ENSEMBLE-BASED NOISE AND OUTLIER DETECTION

Borut Sluban

Doctoral Dissertation
Jožef Stefan International Postgraduate School
Ljubljana, Slovenia, January 2014

Evaluation Board:
Prof. Dr. Sašo Džeroski, Chairman, Jožef Stefan Institute, Ljubljana, Slovenia, and
Jožef Stefan International Postgraduate School, Ljubljana, Slovenia
Prof. Dr. Dunja Mladenić, Member, Jožef Stefan Institute, Ljubljana, Slovenia, and
Jožef Stefan International Postgraduate School, Ljubljana, Slovenia
Prof. Dr. Johannes Fürnkranz, Member, Technische Universität Darmstadt, Germany

MEDNARODNA PODIPLOMSKA ŠOLA JOŽEFA STEFANA
JOŽEF STEFAN INTERNATIONAL POSTGRADUATE SCHOOL

Borut Sluban

ENSEMBLE-BASED NOISE AND OUTLIER DETECTION

Doctoral Dissertation

ANSAMBELSKO ODKRIVANJE ŠUMA IN OSAMELCEV V PODATKIH

Doktorska disertacija

Supervisor: Prof. Dr. Nada Lavrač

Ljubljana, Slovenia, January 2014

Contents

Abstract

The main aim of knowledge discovery and data mining is to discover interesting, new or previously unknown information from the available data and transform it into reusable knowledge about the observed domain. The success of acquiring any knowledge depends, among other things, on the quality of the data at hand. Real-life data inevitably contains errors and unusual instances, which are referred to as *noise*. While the presence of noisy instances usually degrades data mining results, detected unusual or outlier instances may provide new insights into the phenomenon being investigated.

The thesis proposes to improve the process of data cleaning, data understanding and outlier identification, by user-guided ensemble-based noise detection. In this sense the thesis addresses a wider area of noise handling by developing approaches for explicit noise detection, improving the quantitative evaluation of noise detection performance, confirming the practical applicability of the developed noise detection approaches on real-life case studies, and providing an implementation of the developed approaches. Correspondingly, the work in this thesis addresses four main tasks and presents the following.

The first part of the thesis is concerned with the development of explicit noise detection approaches that can be used by various domain experts. The first main contribution is the NoiseRank ensemble-based noise detection and ranking methodology that was developed for user-guided noise detection. The proposed approach enables construction of custom noise detection ensembles, and ranks the detected instances according to the ensemble's confidence of their noisiness. The ranked list of detected instances can be further explored to identify the type of the unusual instances. Additionally, the HARF ensemble-based noise detection algorithm was developed, which demands high agreement among ensemble members to denote an instance as noise, and is suitable for the detection of only the most unusual data instances.

The second part addresses quantitative performance evaluation of noise detection on data with known noisy instances. Improvements over standard evaluation practice are provided by the second main contribution of the thesis, the Viper approach to visual performance evaluation in the precision-recall space. The approach incorporates newly introduced evaluation methods that trade off precision and recall of noise detection. A comparison of standard evaluation approaches and the Viper approach was performed on noiseless and real-life datasets with various levels of randomly injected noise, revealing that the Viper approach enables more intuitive interpretation of achieved results and offers a multifold perspective on noise detection performance.

Real-life applications of ensemble-based noise detection are described in the third part of the thesis. In real-life data, noise appears in different forms, as shown in three use cases: detection of false diagnosis and special cases in a medical domain, identification of atypical documents in a news articles collection, and detection of outlier or borderline documents in a cross domain link discovery problem. The detected noisy and outlier instances were qualitatively evaluated by the corresponding domain experts confirming the usefulness of the proposed noise detection approaches in identifying the most critical noisy instances as well as unusual special cases.

Finally, the implementation of all the developed approaches is provided, enabling public accessibility and repeatability of the presented experiments. The first implementation provides a publicly accessible environment for testing, execution and development of noise detection algorithms, construction of custom ensembles, application of the NoiseRank approach, and quantitative evaluation of noise detection experiments using the Viper methodology. An addition development presents the ViperCharts web-based platform for visual performance evaluation of algorithms for noise detection and also other machine learning tasks, such as information extraction. This platform includes standard as well as novel visual evaluation techniques, and enables to compare algorithm performance according to different evaluation methods, in order to provide diverse perspectives on the performance of evaluated algorithms. These implementations offer wide accessibility of the developed approaches, repeatability of the obtained results, and encourage further use and development of ensemble-based noise and outliers detection methods.

Povzetek

Osrednji namen odkrivanja znanja s pomočjo podatkovnega rudarjenja je odkrivanje zanimivih, novih ali predhodno neznanih informacij v razpoložljivih podatkih in njihovo preoblikovanje v uporabno obliko znanja o obravnavani domeni. Uspešnost metod za odkrivanje znanja je v veliki meri odvisna od kakovosti razpoložljivih podatkov. Podatki, pridobljeni iz realnih problemskih domen, neizogibno vsebujejo napake in neobičajne primere, ki jih imenujemo *šum*. Napake v podatkih imajo negativen vpliv na rezultate rudarjenja podatkov. Primeri, ki so odkriti kot neobičajni ali osamelci v podatkih, pa lahko ponudijo drugačen pogled na raziskovano domeno in celo privedejo do novih spoznanj.

V disertaciji so predlagane izboljšave procesa čiščenja podatkov, razumevanja podatkov in odkrivanja osamelcev v podatkih, z uporabo prilagodljivih ansambelskih metod za odkrivanje šuma. Disertacija obravnava širše področje spopadanja s šumom v podatkih in tako vključuje: razvoj novih metod za neposredno odkrivanje šumnih primerov, izboljšave na področju kvantitativnega vrednotenja uspešnosti odkrivanja šuma, eksperimentalno ovrednotenje razvitih pristopov za odkrivanje šuma v realnih podatkih in implementacijo vseh razvitih metod. Temu ustrezno je vsebina disertacije razdeljena v štiri glavne sklope.

Prvi sklop obravnava razvoj metod za neposredno odkrivanje šuma v podatkih, ki jih lahko uporabljajo različni domenski strokovnjaki. Prvi glavni prispevek je metodologija NoiseRank za ansambelsko odkrivanja šuma in osamelcev v podatkih, ki omogoča uporabniku prilagojeno odkrivanje šuma. Predlagani pristop ponuja uporabo poljubnih, po meri izdelanih ansamblov za odkrivaje šuma in razvrsti odkrite primere glede na njihovo stopnjo šumnosti, ki jo je zaznal ansambel. Rangirane šumne primere je mogoče tudi posamezno preučiti, da se lahko opredeli vrsto zaznane anomalije, ki jo primer predstavlja. Razvit je bil tudi algoritem za ansambelsko odkrivanje šuma, imenovan HARF. Ta pristop zahteva višjo stopnjo strinjanja med algoritmi ansambla pri odkrivanju šuma in je ustrezen za odkrivanje najbolj nenavadnih primerov v podatkih.

Drugi sklop je namenjen kvalitativnemu vrednotenju uspešnosti odkrivanja šuma v podatkih, ki vsebujejo označene ali znane šumne primere. Drugi glavni prispevek te disertacije predstavlja metoda Viper za vizualno vrednotenje uspešnosti z grafično predstavitvijo rezultatov v dvodimenzionalnem prostoru, ki prikazuje razmerje med natančnostjo odkrivanja šuma in stopnjo priklica dejanskega (znanega) šuma v podatkih. Metoda vključuje dve novi meri za vrednotenje uspešnost odkrivanja šuma, ki skušata uravnotežiti razmerje med natančnostjo in priklicem šumnih primerov. Predstavljena je primerjava med standardnim in vizualnim vrednotenjem uspešnosti z metodo Viper na zbirkah podatkov brez lastnega šuma in na zbirkah realnih podatkov, katerim so bile naključno vnesene različne količine šuma. Primerjava je pokazala, da se metoda Viper odlikuje z intuitivno interpretacijo doseženih rezultatov in ponuja večstranski pogled na uspešnost odkrivanja šuma.

Uporabnost ansambelskih metod za odkrivanje šuma v realnih podatkih je bila preverjena v tretjem sklopu disertacije. Metode so bile uporabljene v treh problemskih domenah: za odkrivanje napačno diagnosticiranih in nenavadnih primerov v medicinski domeni, za odkrivanje neobičajnih dokumentov v zbirki časopisnih člankov in za odkrivanje osamelcev ali mejnih dokumentov pri podpori iskanja meddomenskih povezav. Domenski strokovnjaki so preučili odkrite osamelce in šumne primere ter potrdili uporabno vrednost razvitih metod

za odkrivanje najbolj kritičnih šumnih primerov kot tudi neobičajnih posebnih primerov v podatkih.

Nazadnje je predstavljena implementacija vseh opisanih metod v obliki javno dostopnih orodij, ki podpirajo ponovljivost opravljenih eksperimentov. Implementirano je bilo javno dostopno okolje, ki omogoča testiranje, izvajanje in razvoj algoritmov za odkrivanje šuma, izdelavo poljubnih ansamblov, uporabo metodologije NoiseRank in kvantitativno vrednotenje eksperimentov odkrivanja šuma s pomočjo metode Viper. Druga implementacija predstavlja spletno okolje ViperCharts za vizualno vrednotenje uspešnosti algoritmov. Okolje podpira vrednotenje algoritmov za odkrivanje šuma kot tudi drugih algoritmov, ki se uporabljajo v strojnem učenju in podatkovnem rudarjenju. Okolje vključuje standardne in nove metode vizualnega vrednotenja, ki se uporabljajo za primerjavo uspešnosti algoritmov glede na različne mere vrednotenja in tako nudijo raznolik pogled na uspešnost obravnavanih algoritmov. Tovrstna implementacija omogoča široko dostopnost razvitih metod in ponovljivost dobljenih rezultatov ter spodbuja nadaljnjo uporabo in razvoj ansambelskih metod za odkrivanje šuma in osamelcev v podatkih.

1 Introduction

This chapter first introduces the area of data mining and noise handling, and places the research presented in this thesis into a broader data mining context. Next, it presents the motivation for this work and identifies the research problems, states the research hypotheses and the goals of the thesis, and lists the thesis contributions. Finally, the structure of the rest of the thesis is outlined.

1.1 Data mining and noise handling

With the growing amount of data produced daily, automated processing has become a necessity for data interpretation. Data mining is a field of computer science which is concerned with finding new, implicit, previously unknown, interesting and potentially useful information (Witten and Frank, 2005) in the form of extracted patterns and discovered regularities in data. The research area of machine learning provides methods for analyzing large quantities of data, and learning patterns or models representing the data (Mitchell, 1997). Since machine learning methods learn from the available data (experience), which is given in the form of training examples (instances), the process is also referred to as inductive learning. The knowledge induced from the data can then be used for descriptive or predictive purposes.

The success of learning and knowledge discovery from the available data depends on various factors, including data quality. Training data for inductive learning is typically available in the form of a table, where rows present examples in the observed domain, and columns present the properties measured for the observed domain, called domain *attributes*. A special *class* attribute may be available for learning of classification/prediction models. The quality of real-world data is inevitably subject to errors and other irregularities which are usually referred to as *noise*. The term 'noise' applies to all kinds of anomalies in data, from errors to special cases of the observed domain, which make it more difficult to interpret the data as a whole. Examples in data that complicate the learning process and worsen its outcome, or require descriptive models to be more complex and typically less accurate, are called *noisy*, because of their damaging influence on machine learning and data mining results.

Noise in data manifests itself as attribute noise (errors or unusual values), class noise (wrong class label), or a combination of both. Data errors that are a consequence of measurement errors, record keeping, or data preprocessing, represent noise which should best be repaired or discarded from the data. Other irregularities such as outliers, on the other hand, which may not be erroneous but only special cases of regular examples, may lead to new insights, enabling novelty detection and improved domain understanding, or to the discovery of malicious events, like fraudulent transactions or network intrusion (Aggarwal and Yu, 2001; Hodge and Austin, 2004; Chandola et al., 2009; Aggarwal, 2013). Also news and social media content have non-typical or anomalous representatives, which at first seem as noise or outliers, but some of them have proven to be indicators of topic evolution or the beginning of new/emerging topics and trends (Aggarwal, 2012). Since noise may have

adverse effects on the quality of information retrieved from the data, models created from the data, and decisions made based on the data (Zhu and Wu, 2004), noise and outlier handling has been an active area of machine learning and data mining research.

The term 'outlier' does not have a unique definition and is usually interpreted domain-dependently in the scientific literature, sometimes it is even used interchangeably with the term 'noise'. Grubbs (1969) states that an outlier is an example that "appears to deviate markedly from the members of the sample in which it occurs". In this thesis, the term 'outlier' will be used for a very special regular example, and the term 'noise' will be used as its hypernym, denoting that an unusual (noisy) example should be examined by a domain expert to determine whether it is erroneous or it is a very special regular example (i.e., an outlier). Exceptionally, in some application domains the term 'outlier' will be preferred over 'noise' for the purpose of diction, and will serve as its synonym, e.g., a news article may be an outlier with respect to a given article collection, but stating that an article is noisy would be inadequate.

The research topics of this thesis are ensemble-based noise detection, noise ranking and evaluation of noise detection performance. First approaches to noise handling aimed to develop inductive learning algorithms resistant to noise in the data. In order to avoid overfitting noisy data, the constructed models (decision trees and sets of classification rules) were pruned to get a more general form, thereby reducing the chance of overfitting to noise (Gelfand et al., 1991; Quinlan, 1987; Niblett and Bratko, 1987; Mingers, 1989; Fürnkranz, 1997). However, pruning alone cannot avoid the damaging effects of noise and outliers on the structure of the constructed models. Noise filtering or explicit noise detection can avoid this problem. Numerous noise detection approaches found in the literature include classification filtering (Brodley and Friedl, 1999) and saturation filtering (Gamberger and Lavrač, 1997) approaches, rule-based approaches (Khoshgoftaar and Rebours, 2004), itemset-based approaches (Van Hulse and Khoshgoftaar, 2006), distribution-based approaches (Van Hulse et al., 2007) and clustering- or distance-based approaches (Wiswedel and Berthold, 2005; Libralon et al., 2009; Yin et al., 2009; Cai et al., 2011).

Ensemble learning methods are algorithms that construct a set of prediction models (an ensemble) and combine their outputs to a single prediction (Dietterich, 2000). Ensembles are typically used with the purpose of improving the performance of simple or base learning methods. Ensemble techniques have been adopted also for noise handling to obtain more accurate and robust noise detection results than with a single noise detection algorithm (Verbaeten and Van Assche, 2003; Khoshgoftaar et al., 2005, 2006). Another approach to explicit noise detection is noise ranking, which can be obtained by a single noise detection algorithm (Van Hulse et al., 2007) or by aggregated predictions of only one machine learning algorithm trained on bootstrap samples of data (Zhong et al., 2005).

The performance of noise detection is evaluated very differently in the noise handling literature. Commonly the performance of noise detection is evaluated indirectly by measuring the change in classification accuracy, error rates, model complexity, computational complexity, or cluster quality in classification, prediction or clustering tasks (Brodley and Friedl, 1999; Gamberger et al., 2000; Verbaeten, 2002; Zhu et al., 2003; Zhu and Wu, 2004; Khoshgoftaar et al., 2005, 2006; Libralon et al., 2009; Gavriluţ and Ciortuz, 2011; Cai et al., 2011). On the other hand, noise detection performance can be evaluated in a direct way when the main task is noise detection itself. Qualitative performance evaluation of noise detection is commonly used in real-life problems, where the domain expert is asked to evaluate the instances identified by a noise detection algorithm (Gamberger et al., 1999; Loureiro et al., 2004; Van Hulse et al., 2007), thus qualitatively evaluating the significance of each of the detected noisy instances. In contrast, quantitative performance evaluation requires data with known pre-annotated noisy instances. Precision of noise detection and the amount of

retrieved noise (or noise recall) are the most widely used measures, used separately (Khoshgoftaar and Rebours, 2004; Van Hulse and Khoshgoftaar, 2006) or combined with other prediction measures (Verbaeten, 2002; Verbaeten and Van Assche, 2003; Zhu et al., 2003; Miranda et al., 2009; Hido et al., 2011; Dang et al., 2013). Typical presentation of performance results in tabular format proves to be difficult for presenting multiple performance measures at once as well as to compare the performance results of several algorithms.

While numerous noise and outlier detection methods have proven to be beneficial for the quality of data mining results, the research presented in this thesis is focused on explicit noise detection to support user-guided exploration of unusual data, and improvements of quantitative evaluation for the assessment of noise detection performance.

1.2 Motivation

As already mentioned, the data available for data mining influences the quality of information retrieved from the data, models created from the data, and decisions made based on the data (Zhu and Wu, 2004). Noise and outlier detection can be used to improve the quality of data and consequently the quality of data mining results. Since noise handling may represent a demanding step in the data mining process, it is sometimes skipped in the case of abundance of data, where noise presents a relatively small part and its effects on data modeling are outweighed by the great majority of regular instances. However, in the case of limited data resources, which is common in the domains of biology, medical diagnosis, or specific literature mining tasks, it is particularly important to have good quality data, since erroneous examples may seriously affect the results of data mining.

Standard noise handling approaches used in machine learning aim either at improved classification accuracy of models induced from the data, or at noise robustness of the developed algorithms, which should reduce the effect of noise on the algorithm's performance. In contrast, given the importance of every data instance in scarcely represented domains, and the shortage of adequate mechanisms for user-guided detection and analysis of unusual examples in the data, the goal of this work is to develop a methodology enabling the detection of noisy instances to be inspected by human experts in the phase of data cleaning, data understanding, and outlier identification.

Considering the preferences of various domain experts (Gamberger et al., 2003; Loureiro et al., 2004; Van Hulse et al., 2007; Sluban et al., 2014) the proposed approach should result in highest possible precision of noise detection. Moreover, it should detect a relatively small set of data instances that represent noise, outliers or data inconsistencies, worth the expert's inspection time and effort. To this end, the goal of this thesis is not only to develop improved noise detection algorithms but also a methodology which will support the expert in inspecting the data by detecting, ranking and explicitly presenting noisy instances and/or outliers.

In the thesis we propose an ensemble-based noise detection and ranking methodology, called NoiseRank, which ranks noisy instances according to the predictions of different noise detection algorithms thus assuring more reliable results. Furthermore, an environment for construction, execution and evaluation of noise detection ensembles was developed and integrated into a publicly accessible web based platform supporting the construction of data mining workflows.

This thesis is concerned also with quantitative performance evaluation of noise detection. Given that developing a new noise detection algorithm presents a substantial amount of developer's work and that the noise detection performance it achieves is the main indicator of its successful future application, easily understandable and comprehensive presentation of

performance results may significantly support the developer in the evaluation and comparison of results.

In addition, since qualitative examination by domain experts is very demanding and may not always be feasible, quantitative analysis of noise detection results can be performed on data with known or artificially injected noisy instances. Although quantitative evaluation of noise detection is in the literature commonly addressed indirectly by observing how noise removal influences classification and clustering, classification performance measures are not adequate measures for evaluating the success of explicit/actual noise detection. Explicit noise detection can be more appropriately evaluated by performance measures derived from the confusion matrix (e.g., recall, precision, and false positive rate), which measure the relative amounts of retrieved actual noise, and actual or false noise among the detected instances. However, using these measures the evaluation results are typically presented separately or side by side in tables or by simple graphical representations, which are cumbersome/demanding to examine and interpret as a whole; some of these may also offer only little in terms of an aggregated or joint perspective on noise detection performance.

These observations motivated the development of a performance evaluation methodology, which (1) addresses noise detection performance directly by measuring the precision and recall of noise detection on data with known or injected noisy instances, (2) integrates two new evaluation methods that trade off precision and recall: *F-isolines* and the *ε-proximity* evaluation methods, and (3) jointly visualizes these evaluation methods in the two-dimensional *precision-recall space*, allowing simultaneous comparison of three performance measures. This visual performance evaluation methodology can also be used for evaluating information retrieval and entity recognition algorithms, as well as for any other algorithms for which the evaluation in terms of precision, recall and the F-measure is most suitable. The visual performance evaluation approach was implemented in a stand-alone web application, which supports also alternative performance visualizations, as well as incorporated into a web-based data mining platform.

In addition to the developed methods for visual performance evaluation and for ensemble-based noise detection and ranking, the thesis presents the application of ensemble-based noise detection on three real-life use cases involving domain experts, and provides a publicly accessible implementation of the developed approaches enabling the reuse and repeatability of the presented experiments.

1.3 Hypothesis and goals

This thesis is concerned with noise handling for the purpose of data understanding, data cleaning and outlier identification. By using an ensemble-based approach to noise detection, the aim is to improve noise identification performance in various application areas. On the other hand, introducing new evaluation approaches and results visualizations aims to facilitate the validation of the expectedly improved performance of the ensemble-based approach compared to existing noise detection approaches.

The main research hypothesis is that by enabling customized ensemble composition in a publicly accessible environment, the ensemble developer will be able to construct his own ensemble which may outperform the existing noise detection methods and suit best his analytic purposes. Moreover, by enabling to select the desired filtering level (agreement among algorithms in the ensemble) for automated noise filtering, and by inspecting a ranked list of most noisy instances, the working hypothesis is that the user will be able to focus his attention only on noisy and outlier instances of interest.

Considering the performance evaluation of noise detection algorithms, the hypothesis of this thesis is that intuitive and understandable graphical presentation of noise detection per-

formance may enhance the evaluation process by reducing the exploration time compared to commonly used tabular presentations, and by providing the means to simultaneously compare algorithms in terms of several performance measures. Furthermore, different performance visualizations used in the evaluation of machine learning algorithms that depict different aspects of their performance can be beneficial for quantitative evaluation of noise ranking.

The main goals of the dissertation are split into four main tasks, namely: explicit ensemble-based noise detection, visual performance evaluation, applications of ensemble-based noise detection, and implementation and public accessibility of the developed approaches. These four tasks can be further elaborated as follows:

1. Develop an approach to user-guided noise detection for data cleaning, data understanding, and outlier identification.
   (a) In collaboration with domain experts define the requirements of a system aimed at supporting the exploration of unusual data.
   (b) Develop an ensemble-based noise detection and ranking methodology, which will combine and extend the existing approaches to noise ranking and ensemble-based noise detection.
   (c) Design an intuitive presentation of detected noisy instances indicating their degree of noisiness.
   (d) Develop a new ensemble-based noise detection algorithm for explicit noise detection.
2. Improve quantitative evaluation of noise detection performance on datasets with known or artificially injected noisy instances.
   (a) Identify the shortcomings of standard performance evaluation practice.
   (b) Provide alternative evaluation methods.
   (c) Develop a new approach to visual performance evaluation of noise detection, which can overcome the shortcomings of standard performance evaluation practice.
3. Provide evidence of practical applicability of the developed methodology for explicit noise detection on the following real-life case studies:
   (a) Detection of incorrectly diagnosed patients and unusual patient cases in a medical domain.
   (b) Identification of atypical documents in a newspaper articles corpus.
   (c) Detection of domain outlier or borderline documents for supporting cross-domain link discovery.
4. Implement the developed approaches to explicit noise detection and visual performance evaluation in a manner that supports accessibility, repeatability and sharing of experimental results.
   (a) Develop an environment, incorporating all the required building blocks enabling the application, development, testing, and evaluation of noise detection algorithms, as well as their ensembles.
   (b) Provide executable workflows of the developed approaches and experiments.
   (c) Implement a platform for visualization, comparison and sharing of performance results achieved by noise detection and noise ranking algorithms, as well as general classification and information retrieval algorithms.

1.4 Scientific contributions

The scientific contributions of the thesis are listed below.

1. The ensemble-based noise ranking methodology NoiseRank, enabling user-guided explicit noise detection, applicable in the process of data cleaning, data understanding and outlier identification.

2. Developed ensemble-based algorithm HARF achieving increased precision of noise detection.

3. Proposed the F-isoline evaluation method and the ε-proximity evaluation method, which trade off the precision and recall of noise detection.

4. The Viper visual performance evaluation methodology for quantitative performance evaluation of noise detection algorithms on data with known/pre-annotated noisy instances.

5. Experimental evidence of the practical applicability of the proposed noise detection approaches in different domains and different real-life use cases.

6. Implementation of an environment that integrates different noise detection algorithms and supports the construction of custom noise detection ensembles, application of the NoiseRank approach, and comparative evaluation with the developed Viper visual performance evaluation approach.

7. The ViperCharts platform for visualization, comparison and sharing of performance results achieved by noise detection and noise ranking algorithms, as well as general classification and information retrieval algorithms.

The scientific contributions of this thesis were published in the following papers and book chapters:

- Advances is noise detection:

  Sluban, B.; Lavrač, N.; Gamberger, D.; Bauer, A. Experiments with saturation filtering for noise elimination from labeled data. In: *Proceeding of the 12th Multi-conference Information Society, Conference on Data Mining and Data Warehouses, SiKDD 2009.* 240–243 (IJS, Ljubljana, Slovenia, 2009).

  Sluban, B.; Gamberger, D.; Lavrač, N. Advances in class noise detection. In: Coelho, H.; Studer, R.; Wooldridge, M. (eds.) *Proceedings of the 19th European Conference on Artificial Intelligence, ECAI 2010, Lisbon, Portugal, August 16-20, 2010.* Frontiers in Artificial Intelligence and Applications **215**, 1105–1106 (IOS Press, Amsterdam, Netherlands, 2010a).

  Sluban, B.; Gamberger, D.; Lavrač, N. Performance analysis of class noise detection algorithms. In: Ågotnes, T. (ed.) *Proceedings of the Fifth Starting AI Researchers' Symposium, STAIRS 2010, Lisbon, Portugal, 16-20 August, 2010.* Frontiers in Artificial Intelligence and Applications **222**, 303–314 (IOS Press, Amsterdam, Netherlands, 2010b).

- Ensemble-based noise detection, visual performance evaluation, and implementation:

  Sluban, B.; Lavrač, N. ViperCharts: Visual performance evaluation platform. In: Blockeel, H.; Kersting, K.; Nijssen, S.; Zelezný, F. (eds.) *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases, ECML PKDD 2013, Prague, Czech Republic, September 23-27, 2013.* Lecture Notes in Computer Science **8190**, 650–653 (Springer, Heidelberg, Germany, 2013).

  Sluban, B.; Lavrač, N.; Gamberger, D. Ensemble-based noise detection: noise ranking and visual performance evaluation. *Data Mining and Knowledge Discovery* **28**, 265–303 (2014). [IF=2.88]

- Real-life applications of ensemble-based noise detection:

  Sluban, B.; Lavrač, N. Supporting the search for cross-context links by outlier detection methods. *BMC Bioinformatics* **11 (Suppl 5)**, P2 (2010).

  Lavrač, N.; Sluban, B.; Juršič, M. Cross-domain literature mining through outlier document and bridging concept detection. In: *Proceeding of the Workshop on Analysis of Complex NEtworks at ECML PKDD 2010, ACNE 2010.* 35–47 (ACNE Committee, Barcelona, Spain, 2010).

  Sluban, B.; Juršič, M.; Cestnik, B.; Lavrač, N. Evaluating outliers for cross-context link discovery. In: Peleg, M.; Lavrač, N.; Combi, C. (eds.) *Proceedings of the 13th Conference on Artificial Intelligence in Medicine, AIME 2011, Bled, Slovenia, July 2-6, 2011.* **6747**, 343–347 (Springer, Heidelberg, Germany, 2011).

  Juršič, M.; Sluban, B.; Cestnik, B.; Grčar, M.; Lavrač, N. Bridging concept identification for constructing information networks from text documents. In: Berthold, M. R. (ed.) *Bisociative Knowledge Discovery.* Lecture Notes in Computer Science **7250**, 66–90 (Springer, Berlin Heidelberg, Germany, 2012).

  Sluban, B.; Juršič, M.; Cestnik, B.; Lavrač, N. Exploring the power of outliers for cross-domain literature mining. In: Berthold, M. R. (ed.) *Bisociative Knowledge Discovery.* Lecture Notes in Computer Science **7250**, 325–337 (Springer, Berlin Heidelberg, Germany, 2012a).

  Sluban, B.; Pollak, S.; Coesemans, R.; Lavrač, N. Irregularity detection in categorized document corpora. In: Calzolari, N.; Choukri, K.; Declerck, T.; Dogan, M. U.; Maegaard, B.; Mariani, J.; Odijk, J.; Piperidis, S. (eds.) *Proceedings of the Eighth International Conference on Language Resources and Evaluation, LREC-2012, Istanbul, Turkey, May 23-25, 2012.* 1598–1603 (European Language Resources Association (ELRA), Paris, France, 2012b).

1.5 Organization of the thesis

The remainder of the thesis is structured as follows. Chapter 2 presents the related work on noise and outlier detection techniques. It also examines how the performance of noise and outlier detection is evaluated in the related literature. Finally, existing software for noise and outlier detection is reviewed.

Chapter 3 describes the development of the proposed ensemble-based methodology for noise detection and ranking, which enables user-guided data cleaning, exploration of unusual data, and outlier identification. Additionally, a simple and effective ensemble-based algorithm for precise noise detection is presented.

Chapter 4 presents the proposed visual performance evaluation approach for improved quantitative evaluation of noise detection on data with known or injected noisy instances. The approach is compared to standard performance evaluation techniques in an experimental setting with different amounts of injected noisy instances on (initially) noiseless and real-world datasets.

Practical relevance of the developed noise detection approaches is demonstrated in Chapter 5. Ensemble-based noise detection is applied to three real-life use cases in collaboration with the domain experts, addressing the following issues: detection of falsely diagnosed patients and special cases in a medical domain, identification of non-typical news articles, and detection of domain outliers for cross-domain link discovery.

In Chapter 6, the implementation and public availability of the developed approaches is described. An environment is designed that enables the construction, execution, evaluation and sharing of ensemble-based noise detection experiments. Furthermore, a web-based platform for visual performance evaluation of noise and outlier detection and ranking algorithms is presented.

Chapter 7 summarizes the research presented in this thesis, offers a discussion of the work performed, provides the conclusions and lessons learned, and presents directions for further work.

Finally, Appendices A, B and C provide the following additional materials. Appendix A includes the visualization results of the performance evaluation approach, presented in Chapter 4, that were omitted for the conciseness of the chapter. Appendices B and C offer the user manuals for the developed noise detection environment and for the visual performance evaluation platform, respectively.

2 Background and Related Work

Identifying different types of errors and other anomalies in data is a topic widely addressed in the scientific literature. Dealing with errors and anomalies in data, known as noise handling, is an active area of data mining and machine learning research. This section provides an overview of existing approaches to explicit noise and outlier detection, the techniques used for evaluating the performance of such approaches, and available software for data cleaning, noise handling, and outlier detection.

2.1 Background

Various types of errors and anomalies that are encountered in data of different problem domains are usually referred to as *noise*. In the literature the term noise has been used either in its broadest sense as the hypernym of all possible errors and anomalies in data, or very narrowly for specific errors or anomalies in a given domain. The problem in providing a general definition of noise lays in the ambiguity and in the relativity of the "ground truth" according to which a data instance is deemed erroneous or significantly different from the others. Basically, a data instance is considered erroneous or anomalous with respect to a chosen measure or definition of regularity, acceptability or normality. Such measure of regularity is typically obtained from the available data or some prior knowledge about the domain, providing a range of acceptable values according to a calculated or expected distribution of the data. Therefore it is sometimes unclear which instance is indeed erroneous and which is only a very special regular instance.

An instance that deviates greatly from the instances in the sample in which it occurs, is called an *outlier*. This term may apply to regular instances that are very special or to erroneous instances, and its use it typically domain and application dependent. The term is more frequently used in the case of *unlabeled data*, where there are no predefined categories or classes of data instances, and outliers refer to regular special cases as well as to erroneous instances. In this thesis, however, we focus on *labeled data*[1], where outliers are considered to be special cases of regular instances and noise refers to instances with erroneous values in the attributes and/or in the class.

This chapter provides an overview of *explicit* noise and outlier detection approaches, which are employed for identifying and retrieving irregular data instances that can be subsequently analyzed by an expert. This differs from implicit noise handling performed by robust learning algorithms which try to diminish the influence of noise and outliers on the learning task they perform. In the subsequent sections, different noise and outlier detection methods are presented, with the emphasis on explicit noise and outlier detection approaches.

[1] In the analysis of labeled data a special *class* attribute is present that serves as the target attribute for learning, and usually the aim is to predict its values on new instances not seen before in the learning process.

2.2 Noise detection approaches

Noise detection approaches described in this section are presented by roughly following their development over time, and by jointly describing the approaches that use similar techniques for noise identification.

Initial approaches to noise handling aimed to develop inductive learning algorithms resistant to noise in the data. In order to avoid overfitting noisy data, the constructed models (decision trees and rule sets) were pruned to get a more general form, thereby reducing the chance of overfitting to noise in the data (Gelfand et al., 1991; Quinlan, 1987; Niblett and Bratko, 1987; Mingers, 1989; Fürnkranz, 1997). However, pruning alone cannot avoid the damaging effects of noise and outliers on the structure of the constructed models.

Another common approach to noise handling is to eliminate noise by filtering out noisy instances before model construction; this has the advantage that noisy instances will not adversely affect the structure of the induced model. In the classification noise filtering approach by Brodley and Friedl (1999), multiple learning algorithms were applied to induce classifiers which were then used for noise identification. This was achieved in a cross-validation manner where a data instance was identified as noisy if it was incorrectly classified by one or more classifiers. In the case of multiple classifiers used, a majority or consensus scheme can be used, meaning that an instance is identified as noisy only if it is incorrectly classified by the majority or by all of the learned classifiers, respectively. In the thesis, this approach is referred to as the *Classification Filter*. It is commonly adopted in various application areas (Verbaeten, 2002; Miranda et al., 2009) and/or used as a reference noise filtering approach (Gamberger et al., 2000; Khoshgoftaar et al., 2004).

A substantially different noise detection approach was proposed by Gamberger and Lavrač (1997). This approach, called the *Saturation Filter*, is based on the observation that the elimination of noisy examples reduces the so-called *Complexity of the Least Complex correct Hypothesis* (CLCH) value of the training set. The proposed CLCH measure is used to find a saturated training set enabling the induction of a hypothesis which correctly captures the concept represented by the available data; noisy examples are those which are outside the saturated training set.

An approach to classification filtering using different levels of agreement among multiple classifiers, as explored by Khoshgoftaar et al. (2005, 2006) and Verbaeten and Van Assche (2003), is referred to as the *Ensemble Filter*. For large and distributed datasets, the *Partitioning Filter* was proposed by Zhu et al. (2003) which initially splits the dataset into n partitions and a classifier is induced for each of them. The n classifiers are evaluated on the whole dataset, and finally, voting is used to identify noisy examples. Two modifications of the partitioning scheme were introduced by Khoshgoftaar and Rebours (2004): first, the *Multiple-Partitioning Filter* which combines the predictions of multiple classifiers learned on each partition, and second, the *Iterative-Partitioning Filter* which builds only one model on each subset, but iterates the filtering process until a given stopping criterion is satisfied. Another approach which uses ensemble filters sequentially and passes weighted instances to the next iteration of ensemble filters is the *Boosted Noise Filter* (Zhong et al., 2005).

All the above noise filtering algorithms are mainly used for detecting noise which should be removed from the data to improve its quality. Teng (1999) describes a different approach called *Polishing*. When the noisy instances are identified, instead of removing them, they are repaired by replacing the corrupted values with more appropriate ones, and corrected instances are reintroduced into the data set.

Numerous other approaches to noise handling can be found in the literature. A rule-based approach described in (Khoshgoftaar et al., 2004) detects noisy instances by Boolean rules constructed from a set of significant independent variables/attributes. A statistical test

is used to (i) identify candidate significant variables based on their ability to discriminate among the instances belonging to different classes, and to (ii) obtain the critical values of the variables. Each rule is assigned to the target class based on the majority of terms that cover this class. Finally, an iterative process of rule construction and instance classification is used to identify noisy instances.

In (Van Hulse and Khoshgoftaar, 2006), noise is detected by using frequent itemsets. These are sets of instances with common attribute values that satisfy a user-defined minimum support threshold. As such, frequent itemsets contain information about the structure and dependence between attributes. Each frequent itemset is given a class label, based on the proportion of contained instances belonging to each class. If an instance is contained in an itemset that has a large proportion of instances from the other class, it is denoted as noisy. A score indicating the relative likelihood of an instance of being noisy is used for final noise identification.

The distribution-based approach of Van Hulse et al. (2007) compares the distributions of attribute values for groups of data instances to detect noisy examples. Pairwise comparison over all attributes of a dataset is performed. Instances are grouped according to a value range of an attribute and the distribution of values for other attributes is observed. The instances that have a large deviation from normal, given the values of a pair of attributes, are assigned a *noise factor* score which is used for noise identification.

A variety of clustering or distance-based approaches (Dave, 1991; Wiswedel and Berthold, 2005; Libralon et al., 2009; Yin et al., 2009; Cai et al., 2011) use different data representations (feature spaces) and different clustering algorithms to identify noise as small groups of instances that are not close to any large clusters, and/or are dispersed among the clustered instances. The later two types of noise identification are common in the related but more specialized field of outlier detection.

Gavriluţ and Ciortuz (2011) explored various techniques for noise reduction in the training datasets for the task of malware detection. They constructed different noise-reduced training sets to improve malware detection results. The initial training set was filtered by distance-based heuristics removing pairs of data instances which had a small Hamming distance[1] but different labels, and by using a classification algorithm to remove different amounts of misclassified instances closest to the separating hyperplane.

Noise identification at the level of datasets was addressed by Garcia et al. (2013). The authors distinguish between noiseless and noisy datasets based on features constructed from various data complexity measures. These measures assess the class separability, the complexity of the decision borders among classes, and the geometry, topology and density indicators of the dataset. Their experimental datasets, both noiseless and noisy, were converted into feature vectors in the 'complexity measure' feature space and classification algorithms were trained to discriminate between noiseless and noisy datasets.

In this thesis, a selection of classification and saturation filtering approaches to noise detection will be included in the proposed ensemble-based noise detection approach.

2.3 Outlier detection approaches

In the analysis of unlabeled data, the approaches for detecting anomalies in data are typically referred to as outlier detection approaches.[2] An outlier is, strictly statistically speaking, an observation that is numerically distant or lies outside the overall distribution of a sample of observations. Although many definitions can be found in the literature, the most commonly cited definition is the one from Grubbs (1969), which states that:

[1] The Hamming distance of two vectors of equal length is the number of positions at which the corresponding values are different. [2] Note that outlier detection can be also performed on labeled data.

> "*An outlying observation, or outlier, is one that appears to deviate markedly from other members of the sample in which it occurs.*"

The wide range of existing approaches to outlier detection can be grouped in different ways, however most common approaches to outlier detection in data mining are statistical or distribution-based, distance-based, density-based, and clustering-based. Early approaches to outlier detection were developed in the field of statistics (Barnett and Lewis, 1994). They assume that the data is following a distribution model (e.g., Gaussian) and label a data instance as an outlier if it deviates from the model. However, such approaches do not scale well and are limited by the fact that the distribution of real-world datasets may be unknown or is hard to determine.

Distance-based outlier detection approaches identify outliers as the objects lying in the sparsest regions of the feature space. The original approach proposed in (Knorr and Ng, 1998) states that given parameters k and R, an example is a distance-based outlier if less than k examples in the input data set lie within distance R from it. Variations of this definition were suggested to overcome certain difficulties, such as determination of the parameter R, lack of ordering of outliers, and high time complexity (Ramaswamy et al., 2000). Different distance-based outlier detection algorithms are based on computing full dimension distances or finding lower dimensional projections (Aggarwal and Yu, 2001; Angiulli et al., 2006; Zhang et al., 2009).

Density-based approaches were originally proposed in (Breunig et al., 2000a), which compute a local outlier factor (LOF) for each example depending on the density of its local neighborhood. Examples with high LOF have local densities smaller than their neighborhood and typically represent stronger outliers, unlike data points belonging to uniform clusters that usually tend to have lower LOF values. Another approach introduces the relative density factor (RDF) (Ren et al., 2004). The factor indicates the degree at which the density of a data point contrasts those of its neighbors. Based on RDF, data points deep in the clusters are pruned to identify outliers only within the remaining subset of data. A combined statistical and density-based approach is presented in (Hido et al., 2011). Outliers in the test set are detected by using a training set consisting only of inliers. A ratio of densities in the training and test set is computed that serves as an outlier score.

Many clustering algorithms identify outliers as by-products of clustering (Ng and Han, 1994; Ester et al., 1996; Barbará and Chen, 2000; Rehm et al., 2007). Small clusters or individual data instances not belonging to any cluster are identified as outliers. Since the main objective of such methods is clustering, they are not always optimized for outlier detection. A clustering method focused on outlier detection was proposed in (Loureiro et al., 2004). The authors suggest using hierarchical clustering methods, because of the unbalanced distribution of outliers versus "normal" instances. The size of resulting clusters is used to identify groups of instances that are distinct from the majority of the data.

Some other notable methods for outlier detection are the following. The Connectivity outlier factor (COF) (Tang et al., 2002), which separates the notion of density from that of isolation, can detect outliers independently of the densities of the patterns from which they deviate. The Local Correlation Integral (LOCI) algorithm (Papadimitriou et al., 2003) uses data-dictated cut-offs to automatically detect outliers and micro-clusters. Given a radius r and a point p, the algorithm computes the relative deviation of its local neighborhood density from the average local neighborhood density in its r-neighborhood. The approach identifies outliers as data instances with high relative deviations. The LoOP (Local Outlier Probability) outlier detection method (Kriegel et al., 2009) combines local density-based outlier scoring with a probabilistic, statistically-oriented approach to overcome the problem of traditional outlier scores which are not directly comparable for different density regions. A *Probabilistic Local Outlier Factor* (PLOF) is used to obtain the probability value LoOF

which enables to compare the "outlierness" of instances from different density regions of one or multiple datasets.

Schubert et al. (2012) proposed a formalization framework for the analysis of local outlier detection approaches allowing a theoretical comparison and generalization of the approaches. The observed approaches were generalized according to their common algorithmic scheme and can be used as a (general) baseline for comparison with other specialized approaches, as well as for disambiguation of "locality" among the different local outlier detection approaches.

Recently the LODI method with the emphasis on the interpretation of outliers was proposed (Dang et al., 2013). This local outlier detection approach adaptively selects the neighboring instances and uses a learning method to identify the optimal subspace in which an outlier is maximally separated from its neighbors. The results of the optimization problem provide also the relevant features for understanding the exceptional properties of outliers.

Outlier detection on streaming or time series data has recently received a lot of attention in the scientific literature. It aims to develop algorithms that can work on large amounts of data, where new data arrives continuously and is too big to be stored. Therefore, different algorithms working on sliding windows of data are proposed (Angiulli and Fassetti, 2007; Wang et al., 2010), incremental versions of existing approaches (incremental LOF) (Pokrajac et al., 2007) or online clustering algorithms (Marascu and Masseglia, 2009; Cao et al., 2010; Assent et al., 2012) are proposed. However, outlier detection on streaming data is out of the scope of this thesis.

Extensive overviews of outlier detection methodologies are presented in the survey papers by Hodge and Austin (2004) and Chandola et al. (2009), as well as in the chapter by Ben-Gal (2010) and in the book *Outlier Analysis* by Aggarwal (2013).

This section described outlier detection techniques which are typically used on unlabeled data. In this thesis, noise detection approaches described in Section 2.2 will be used for the detection of erroneous and outlier instances on labeled data. Nevertheless, an overview of outlier detection methods was covered for completeness, as outlier identification by domain experts will be presented in the subsequent sections.

2.4 Evaluation of noise and outlier detection performance

The performance of noise detection and noise filtering algorithms is evaluated very differently throughout the literature and it usually depends on the problem for which noise handling is used. Similarly, for evaluating the performance of outlier detection methods different strategies were adopted.

2.4.1 Evaluating noise detection

Commonly the performance of noise detection is evaluated indirectly by measuring the change in performance of machine learning algorithms in classification or prediction tasks (Brodley and Friedl, 1999; Gamberger et al., 2000; Zhu and Wu, 2004; Khoshgoftaar et al., 2006; Libralon et al., 2009; Gavriluţ and Ciortuz, 2011). In these cases, noise detection is considered as a preprocessing step and its performance is evaluated only through its effect on the performance of the main classification/prediction or clustering task, e.g., the change in classification accuracy, error rates, model complexity, computational complexity, or cluster quality (Verbaeten, 2002; Zhu et al., 2003; Khoshgoftaar et al., 2005; Cai et al., 2011).

On the other hand, noise detection performance can be evaluated in a direct way when the main task is noise detection itself. Khoshgoftaar and Rebours (2004) compared the sets of instances detected by different noise filtering algorithms and observed their overlaps to

determine the significance of the detected noisy instances. Qualitative performance evaluation of noise detection is commonly used in real-life problems, where the domain expert is asked to evaluate the instances identified by noise detection algorithms (Gamberger et al., 1999; Van Hulse et al., 2007), by qualitatively evaluating the significance of each of the detected noisy instances. This type of performance evaluation is usually limited to one or a small number of real-life domains and to the evaluation of one or a comparison of only a few noise detection algorithms.

In contrast, quantitative performance evaluation avoids these limitations, but requires data with known pre-annotated noisy instances. When the noisy instances are known in advance, the performance can be evaluated with standard performance measures used in information retrieval (Manning et al., 2008). *Precision* of noise detection and the amount of retrieved noise or the noise *recall* are the most widely used measures, used separately (Khoshgoftaar et al., 2004; Van Hulse and Khoshgoftaar, 2006) or combined with other prediction measures (Verbaeten, 2002; Verbaeten and Van Assche, 2003; Zhu et al., 2003; Miranda et al., 2009). Some evaluation approaches aggregate the precision and recall results, like the F-measure which presents the trade-off between precision and recall of noise detection (Sluban et al., 2010a,b), or the precision-recall curves, which are used for the evaluation of ranked retrieval results, i.e., ranked instances obtained by a noise filtering algorithm (Zhong et al., 2005). Since noise detection can be considered also as a binary classification problem (with classes 'noisy' and 'non-noisy/regular'), classification accuracy was used in (Garcia et al., 2013) to evaluate the identification of noisy datasets.

This thesis will, in addition to the standard evaluation practice, present a visual approach for quantitative performance evaluation of noise detection algorithms on data with labeled noisy instances. Furthermore, qualitative evaluation of detected suspicious instances in real-life applications will be provided from the analysis performed by domain experts.

2.4.2 Evaluating outlier detection

The first approaches that identified outliers only indirectly, as side products of clustering (Ng and Han, 1994; Ester et al., 1996), merely mentioned their existence and were mainly concerned with the evaluation of the algorithms' run time, and the descriptive evaluation of cluster quality.

For many approaches the outlier detection results are inherent to their definition of outliers, and the chosen parameters or thresholds. Therefore, typically descriptive evaluation of outliers detected on a real-world dataset is provided (Knorr and Ng, 1998; Breunig et al., 2000a; Ramaswamy et al., 2000; Papadimitriou et al., 2003). Synthetic as well as real-world datasets are often used to present meaningful outliers that were not identified by previously known approaches (Breunig et al., 2000a; Tang et al., 2002; Papadimitriou et al., 2003), and to show the improvement in execution time over the competing algorithms (Knorr and Ng, 1998; Breunig et al., 2000a; Ramaswamy et al., 2000; Aggarwal and Yu, 2001; Tang et al., 2002; Papadimitriou et al., 2003; Ren et al., 2004).

Qualitative evaluation of outlier detection is performed in terms of the performance measures computed from the confusion matrix. A dataset with labeled outlier instances is a necessary requirement. Loureiro et al. (2004) used a dataset labeled by domain experts and recorded the recall and the size of the obtained outlier set, as desired by the domain experts. Also labeled real-world datasets having an uneven class distribution are used for empirical evaluation of outlier detection. The data instances belonging to the smallest classes, representing the rarest types of data, are denoted as outliers. Outliers were evaluated as a hole set or only the top n suggestions, depending on the type of algorithm. Evaluation was performed in terms of precision (Barbará and Chen, 2000; Aggarwal and Yu, 2001;

Zhang et al., 2009), as well as true positive and false positive rates, ROC analysis, and Area Under Curve (Angiulli et al., 2006; Kriegel et al., 2009; Hido et al., 2011; Dang et al., 2013).

Tang et al. (2002) present an interesting evaluation approach called the *ON-compatibility* (ON stands for Outlier and Non-outliers). They argue that outlier measures (e.g., LOF) are basically functions of the observed data point p and a set of parameters S, i.e., $f(p, S)$. Furthermore, a division of data into outliers and non-outliers is called an interpretation I. The authors propose a definition which states that an outlier measure function $f(p, S)$ is ON-compatible if for an interpretation I there exists a set of parameters S and a value u, such that for each outlier o holds $f(o, S) > u$, and for each non-outlier n holds $f(n, S) < u$. Basically, it indicates that for a single set of parameters an outlier measure function can detect all outliers of a given interpretation.

Another interesting tool for the evaluation of outliers is the LOCI plot presented by Papadimitriou et al. (2003). The plot can be constructed from the values computed by the LOCI algorithm for any point p_i from the dataset. For the selected point p_i it shows (i) the number of neighbors in the r vicinity and (ii) the average number of neighbors that the r-neighbors of p_i have, as well as (iii) the corresponding standard deviation. The LOCI plot gives information about the vicinity of p_i: why it is an outlier with regard to its vicinity, as well as information about nearby clusters and micro-clusters, their diameters and inter-cluster distances.

2.5 Software tools for noise and outlier detection

Identifying different types of anomalies in data, such as noise and outliers, has proved to be beneficial in many application areas, ranging from medicine and biology to mechanical engineering, marketing, and security. Various tools and software applications have been developed for noise identification/data cleaning and outlier detection.

The simplest applications for identifying noisy data instances provide a user interface where the user can find erroneous or atypical data instances. The aim of such applications is to provide help in semi-manual data cleaning. Tools for identification of duplicates and for instance filtering based on attribute range limitation are usually included. Examples of such applications are Wrangler (Kandel et al., 2011), OpenRefine[1], and DataCleaner[2].

Commercial software products for numerical computing and statistical analysis employ more sophisticated approaches to outlier identification. MedCalc[3] uses statistical tests, such as the Grubbs' test (Grubbs, 1969) or the Generalized ESD test (Rosner, 1983), to identify outliers. The IBM SPSS software[4] offers outlier detection based on statistical testing, as well as data modeling with decision trees or clustering methods. Oracle[5] provides data mining solutions as native SQL functions within the Oracle database. Its anomaly detection component uses an One-Class Support Vector Machine (SVM) to find unusual cases within the data. In MATLAB[6], a distance-based outlier measure is used for outlier detection. The Salford Predictive Modeler software[7] can spot outliers and anomalies in data with its Random Forests tool. Other commercial business analytics software also advertise outlier detection functionality in the data preparation phase using statistical tests, e.g., SAS Enterprise Miner[8].

The open source data mining environments, which were mainly developed by the researchers in the field, implement outlier detection methods that were presented in Section 2.3. In the Weka data mining software (Hall et al., 2009), only a statistical *interquartile*

[1] http://openrefine.org/ [2] http://datacleaner.org/ [3] http://www.medcalc.org/
[4] http://www-01.ibm.com/software/analytics/spss/
[5] http://www.oracle.com/technetwork/database/options/advanced-analytics/odm/
[6] http://www.mathworks.com/products/matlab/ [7] http://www.salford-systems.com/products/spm
[8] http://www.sas.com/technologies/analytics/datamining/miner/

data filter is included in the distribution, whereas a LOF filter and an isolation forest classifier for detecting and removing outliers are available as additional packages. The ORANGE data mining environment (Demšar et al., 2013) includes a simple outlier detection component that calculates the z-score[1] for each data point and denotes it as an outlier if the z-score is higher than a predefined threshold. The RapidMinner (Mierswa et al., 2006) system for data-mining offers methods for distance-based and density-based outlier detection, and methods using the LOF and COF factor to identify outliers. In the KNIME (Berthold et al., 2009) data mining environment, outlying values of an attribute can be identified by the *Boxplot* component, which plots the distribution of the values and thereby enables to easily identify outliers. Additionaly, in both KNIME and RapidMiner it is possible to add noise to the dataset (corrupt data values, change data labels, add new features and/or data points), however explicit noise detection other than outlier detection is not provided; it is suggested to overcome the influence of noise by feature selection.

Furthermore, two popular machine learning and data mining software packages/libraries provide outlier detection functionality for their respective programing languages. The first is *scikit-learn*[2] (Pedregosa et al., 2011), a Python[3] library for machine learning. Novelty and outlier detection are performed by a one-class SVM that is used to distinguish between the set of observations belonging to the same distribution and the observations that are different. Another proposed approach is the *ecliptic envelope* which fits a robust covariance estimate to the data, and thus fits an ellipse to the central data points, ignoring points outside the central area. However, the ecliptic envelope approach is suitable only for unimodal data distributions, and fails to produce good results for "multi cluster" data.

The second is *RDataMining*[4] a data mining package for the *R* software for statistical computing and graphics[5]. The package provides a *boxplot* function for visualizing the distribution of a variable and has an *out* component which gives a list of (univariate) outliers, i.e., data points lying beyond the extremes of the whiskers (lines extending vertically from the boxes indicating variability outside the upper and lower quartiles). Functions for computing the local outlier factor using the LOF algorithm are included as well.

More specialized algorithms for noise and outlier detection are usually accessible as packages, scripts, or third-party extensions for the open-source data mining platforms or for different programing languages. Stand-alone tools or applications are also available, like GritBot[6] a tool for improving data quality by searching for anomalies in subsets of the data, or like the SOREX toolkit[7] for subspace outlier ranking (Müller et al., 2010), which enables to explore outliers in subspace projections of the data space. Another example is ELKI (Environment for Developing KDD-Applications Supported by Index-Structures)[8] offering support for various spatial outlier detection techniques which focus on the spatial locality of data (Achtert et al., 2011), and a data visualization technique for unified evaluation of outlier detection models (Achtert et al., 2010).

The reviewed software mainly employs implementations of statistical or distance/density-based outlier detection techniques. None of them use ensemble approaches to noise or outlier detection, as well as very few of them offer users any involvement in the noise or outlier identification process. This thesis will present a user-guided ensemble-based noise detection and ranking approach to be used by the experts for exploration, correction and cleaning of data. Its implementation, which is publicly accessible within a web-based data mining platform, will be presented in detail in Chapter 6.

---

[1] The z-score of an observation is the (signed) number of standard deviations the observation deviates from the mean. [2] `http://scikit-learn.org/` [3] `http://www.python.org/` [4] `http://www.rdatamining.com/` [5] `http://www.r-project.org/` [6] `http://www.rulequest.com/gritbot-info.html` [7] `http://dme.rwth-aachen.de/OpenSubspace/SOREX` [8] `http://elki.dbs.ifi.lmu.de/`

3 Ensemble-based Noise Detection and Ranking

This chapter describes the main approaches to noise detection presented in this thesis. It starts with a motivation for explicit noise detection, which includes facts from the literature as well as our insights from the work with domain experts. Furthermore, the ensemble-based noise detection and ranking methodology, named NoiseRank, is described. Additionally, we present the High Agreement Random Forest noise detection algorithm, called HARF, as a simple and effective approach to precise noise identification.

3.1 Motivation for explicit noise detection

Standard noise handling approaches used in machine learning aim at improved classification accuracy of models induced from the data. They aim to mitigate the influence of noise by removing it from the training data in the model learning phase, or by generalizing model construction procedures to avoid the effects of noise in the data.

Noise filtering and noise detection in data have proven to be of great interest to the scientific community, as well as to the industry. Zhu and Wu (2004) reported a systematic evaluation of the negative effects of noise in machine learning in terms of reduced classification accuracy, increased time for model building and increased model size. However, enhancing data quality for better data modeling and higher predictive performance is only one reason for the interest in the field. The ever more prominent reason is to find interesting data instances that either provide new insight into the data at hand, presenting special cases or novelties, or identify atypical instances that might be cases of abnormal behavior indicating manufacturing errors, changes in the environment, or malicious activity. This is of great interest for real-life applications in security monitoring, transaction surveillance, or business analytics (Aggarwal and Yu, 2001; Hodge and Austin, 2004; Chandola et al., 2009).

The focus of the work presented in this thesis is on providing a noise detection methodology that would enable the inspection of suspicious instances by human experts in the phase of data cleaning, data understanding and outlier identification. The work was motivated by the collaboration with medical experts in a joint effort to improve medical decision making. Examining historic patient records can help identify serious medical mistakes like false diagnosis, as well as obtain new insights in the observed medical problem by identifying complex medical cases which require special attention.

A noise detection methodology that would be most helpful for any domain expert should first of all result in highest possible precision of noise detection, since manual inspection of non-noisy data instances should preferably be avoided. Moreover, the noise detection approach should detect a relatively small set of data instances that represent noise, outliers or data inconsistencies, worth the expert's inspection time and effort. To this end, the goal is not only to develop improved noise detection algorithms but also a new methodology which will support the expert in inspecting the data by detecting, ranking and explicitly presenting noisy instances and/or outliers. This methodology should enable the user to choose the desired approaches to noise detection. Additionally, a predicted degree of "noisiness" should

be clearly indicated for each suspicious instance, and their inspection should be enabled. A scheme of the proposed methodology is shown in Figure 3.1.

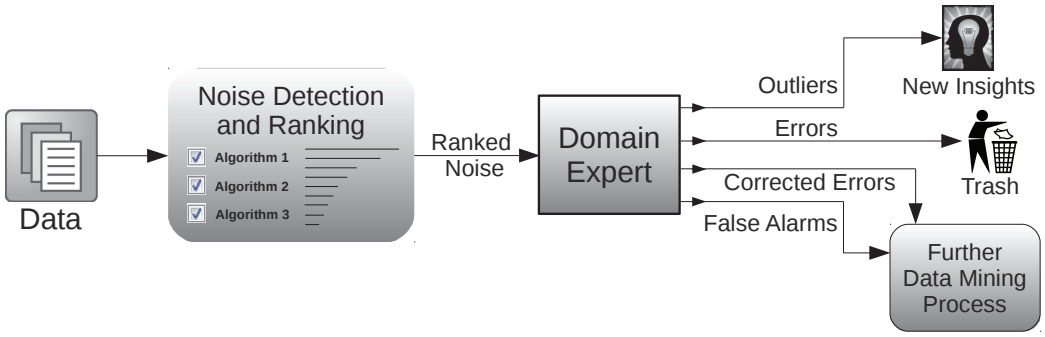


Figure 3.1: Scheme of the proposed methodology for expert-guided noise detection.

The proposed methodology enables expert-guided noise detection. The user can inspect the detected noisy instances and decide whether they are interesting outliers which lead to new insights in domain understanding, erroneous instances which should be removed from the data or false alarms (non-noisy regular instances) and/or instances with minor corrected errors to be reintroduced into the dataset[1].

3.2 NoiseRank

This section presents the NoiseRank ensemble-based noise detection and ranking methodology. It describes the background and the reasoning behind its design, and presents an example of a noise detection ensemble employed by the NoiseRank methodology on a real-world dataset.

3.2.1 NoiseRank methodology for explicit noise detection

As described in the previous section, the goal was to develop a methodology that would enable the detection of noisy data instances for inspection by domain experts or researchers in the process of data cleaning, data exploration, and outlier identification. Developing a methodology like this requires taking into account the suggestions and preferences of domain experts. As suggested in the literature reporting collaboration with domain experts (Gamberger et al., 2003; Loureiro et al., 2004; Van Hulse et al., 2007) and considering our experience from the work with domain experts, an interactive system for noise and outlier identification, applied to a dataset $\mathcal{D}$, should satisfy the requirements outlined below.

Requirements for a noise detection methodology for expert interaction:

1. return a set of suspicious instances $\mathcal{S}_\mathcal{N}$,
2. for each instances in $\mathcal{S}_\mathcal{N}$ there is an indication of its suspiciousness,
3. $\mathcal{S}_\mathcal{N}$ should be of reasonable size for manual inspection,
4. $\mathcal{S}_\mathcal{N}$ should contain as many of the actually suspicious instances contained in $\mathcal{D}$ as possible, and
5. $\mathcal{S}_\mathcal{N}$ should contain as little as possible of correct or normal instances from $\mathcal{D}$.

[1] The latter is referred to as *polishing* by Teng (1999).

The first requirement is naturally satisfied, since the principal task of a noise detection algorithm is to return a set of instances that it identifies as noisy. The second requirement is some kind of a hint desired by the experts to see the degree to which an instance was considered as being noisy by the system. Basically, this means that each instance in the set $\mathcal{S}_\mathcal{N}$ should be accompanied by a measure or score indicating its level of "noisiness", i.e., the system's confidence in declaring an instance noisy.

Considering the terminology of information retrieval, the fifth requirement would translate to high precision of noise detection, meaning the set $\mathcal{S}_\mathcal{N}$ would ideally include only instances that really represent noise and/or outliers. The methodology should point the user to the suspicious examples and avoid causing unnecessary manual inspection of regular data instances. On the other hand, the fourth requirement suggests to retrieve as many of the noisy or outlier instances as there are in $\mathcal{D}$. This is equivalent to high noise recall in terms of information retrieval. However, in real-world scenarios (real-world $\mathcal{D}$), noise recall may be hard to determine, since the actual amount of noise and outliers in a dataset would require manual inspection of a potentially very large dataset. But estimates of the real amount of noise and outliers in a dataset can be induced through manual inspection of a representative sample of the dataset.

Lastly, the third requirement proposed by the domain experts is in a way a trade-off between requirements 4 and 5, precision and recall, respectively. A smaller set $\mathcal{S}_\mathcal{N}$ may promise higher precision of detected noisy instances, but most probably at the price of low recall, and vice versa, in the case of a larger set $\mathcal{S}_\mathcal{N}$. However, if a noise detection approach performed ideally (perfect precision and recall) even on a dataset with a high level of noise, the users would probably not mind inspecting a large set $\mathcal{S}_\mathcal{N}$.

After considering the preferences of the future users of the developing noise detection methodology, their requirements can be summarized into two major objectives of the methodology, outlined below.

Objectives of the noise detection methodology:

(i) maximize the precision of noise detection in trade-off with an acceptable level of recall,

(ii) provide an intuitive measure of confidence for the detected suspicious instances.

The idea of enhancing noise detection performance came from the area of classification, as well as from real-life experience. When we want to make a good decision we ask for advice, e.g., ask several doctors for opinion before agreeing to a medical procedure, or read several user reviews before buying a product. In machine learning, ensemble methods were typically used for the purpose of improving the performance of simple or base learning methods. Ensemble learning methods are algorithms that construct a set of prediction models (an ensemble) and combine their outputs to a single prediction (Dietterich, 2000). Since noise and outlier detection algorithms basically learn to distinguish between regular and non-regular instances, they can be considered as prediction models that classify instances into the classes "noisy" and "non-noisy" (or "regular").

The strength of ensemble methods lies in their ability to correct errors made by some of its members (Polikar, 2009). Therefore, ensemble members have to be diverse in terms of the errors they produce, so that their combination can reduce the total prediction error (Brown et al., 2005). Ensembles with greater diversity among members tend to give higher prediction accuracy (Kuncheva and Whitaker, 2003). Achieving diversity among the members of an ensemble can be achieved in many ways. Popular methods based on boosting (introduced by Schapire (1990)) and bagging (introduced by Breiman (1996)) construct homogeneous ensembles (all members/classifiers use the same (learning) algorithm) and diversify members by training them on differently selected samples/subsets of the training

data. Some approaches prefer to use different parameter settings of algorithms in the training phase to obtain different classifiers. Alternatively, the ensemble can be constructed using entirely different algorithms. In their work (Gashler et al., 2008) the authors have shown that heterogeneous ensembles may lead to better results than homogeneous ensembles.

The use of ensemble methods for noise and outlier detection was chosen to address the objective (i) of the designed noise detection methodology. The proposed methodology therefore enables to construct ensembles from diverse noise detection algorithms that can be selected by the user. Moreover, the user can include standard as well as custom noise detection algorithms in the ensemble.

Ensemble methods provide more reliable results compared to a single algorithm, because they combine the predictions of all member algorithms to construct the final prediction. A variety of different ensemble combination rules can be found in the literature (Kuncheva, 2004), the two main groups being *algebraic combiners* and *voting combiners*. The algebraic combiners operate with continuous prediction algorithm outputs, as probabilities or scores, they include simple or non-trainable combiners, such as the mean, sum, min/max, or median functions, and trainable combiners, such as weighted average or fuzzy integral. The voting combiners, on the other hand, are intended for ensembles where their member algorithms return only label outputs. Most commonly used voting schemes are the following:

- *Consensus (unitary) voting.* If all members of the ensemble agree on a particular label, then it is selected as the ensemble's final decision. Otherwise no label/decision is returned by the ensemble.
- *Majority (plurality) voting.* In the case of a two-class decision problem, the label receiving more than half of the votes from the ensemble members is chosen as the ensemble's final decision. If both classes get the same number of votes, then either one label is chosen at random or no label/decision is returned by the ensemble. For tasks with more than two labels, either the label with the highest number of votes is selected or the label for which more than half of the ensemble members have voted (otherwise no label/decision is returned).
- *Weighted majority voting* is another voting scheme suggested in the case where an ensemble member is believed to be more competent and should have more power in the final decision.

In our case the data analyst can construct the ensemble from different noise detection algorithms. Algebraic combiners working with continuous input values would require scores or probabilities from all ensemble members. Scoring algorithms may return scores from different ranges with different decision thresholds. Proper normalization of scores can overcome this problem, however algebraically combining these scores with the outputs of probabilistic algorithms and algorithms returning neither scores nor probabilities but only labels would still present a challenging task where it remains unclear if this would yield a valid decision/result. Therefore, voting was used in our case, since it presents an alternative that only combines the final decisions of the algorithms in the ensemble.

Furthermore, the results of voting present an easily understandable and very intuitive measure of confidence that can be assigned by the ensemble-based noise detection approach to each detected potentially noisy instance. By ranking the detected instances according to the number of votes obtained by the noise detection ensemble, and additionally labeling which ensemble members denoted an instance as noisy, the objective (ii) for the noise detection methodology is adequately addressed.

Development of a noise detection methodology that satisfies the requirements of domain experts to be used in data exploration, data cleaning and outlier identification, should meet

objectives (i) and (ii). The discussion above described how to achieve high precision of noise detection and showed how to present noise detection results of an ensemble in a way that meaningfully indicates the potential noisiness of the detected instances. With this in mind we can finally summarize the NoiseRank methodology for ensemble-based noise detection and ranking, by listing its main features below.

NoiseRank methodology features:

1. ensemble-based noise detection,
2. custom selection of diverse ensemble members,
3. indication of "noisiness" for a detected instance by ensemble voting results,
4. ranking of detected noisy instances by voting outcomes,
5. display of ensemble members that identified an instance as noisy, and
6. an interactive noise ranking visualization allowing instance selection and exploration.

NoiseRank takes into account predictions of a number of different noise detection approaches to obtain reliable noise detection results: the higher the agreement, the higher is the chance that the detected instance is indeed erroneous or atypical and is therefore worth inspecting. The proposed ranking approach, which allows the expert to inspect a ranked list of potentially noisy instances, proves to be a powerful way of presenting the results of noise detection to the expert, as the goal is to focus the expert's attention on the most relevant noisy instances that are worth his inspection time and effort. The output of the NoiseRank methodology will be presented in Section 3.2.2, after introducing selected methods (in Section 3.2.2) that can be used in the noise detection ensemble.

In contrast to other noise or outlier ranking approaches where the ranking is produced only by one detection algorithm (Van Hulse et al., 2007; Moonesinghe and Tan, 2008; Müller et al., 2011) or by aggregated predictions of only one machine learning algorithm trained on bootstrap samples of data (Zhong et al., 2005), the NoiseRank approach ranks noisy instances according to the predictions of different noise detection algorithms, thus assuring more reliable results. Compared to ensemble filtering by Brodley and Friedl (1999); Verbaeten and Van Assche (2003) and Khoshgoftaar et al. (2005, 2006), it enables the user to construct custom ensembles from different existing algorithms, including his own noise detection algorithms, thereby offering the user to tune the noise detection performance towards increased precision or recall of noise identification, as desired for the given use case. In addition, it allows to select the desired filtering level (agreement among algorithms in the ensemble) for automated noise filtering, or to select only the most interesting instances by inspecting the ranked list of noisy instances provided by an interactive visualization interface.

3.2.2 Selected methods for explicit noise detection

As the aim of NoiseRank is to enable the user to find noise and outliers in the phase of data exploration, methods for explicit noise detection need to be incorporated. As example methods which can be employed in NoiseRank, this section presents only the selected classification and saturation-based noise filtering methods which were used when NoiseRank was applied to a medical domain[1].

[1] Note that the publicly accessible implementation of the NoiseRank methodology described in Section 6.1.1 enables developers to easily incorporate their own algorithms into the noise ranking ensemble.

Classification-based noise detection methods

The idea behind the noise filtering approach presented by Brodley and Friedl (1999) is to use a classifier as a tool for detecting noisy instances in data. Their method, later called the *classification filter*, is performed in a cross-validation manner. They applied 4-fold cross-validation to repeatedly use three folds for training of a classifier and one complementary fold for class prediction where the incorrectly classified instances are identified as noisy. Figure 3.2 depicts the idea of classification filtering.

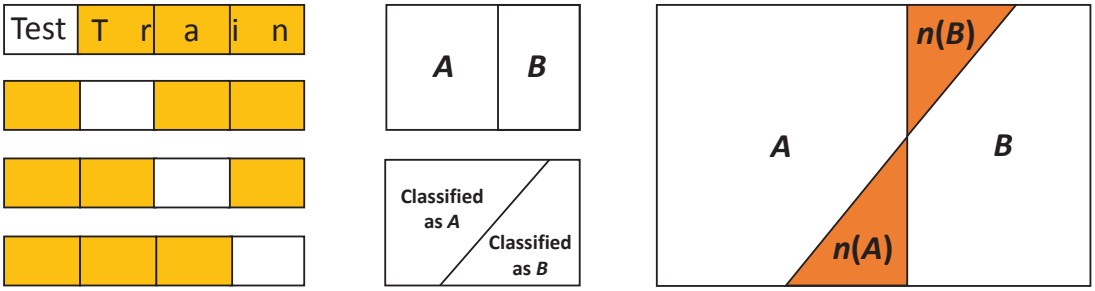


Figure 3.2: Classification filtering using cross-validation. A and B are the class labels of instances in the test fold. The misclassified instances of A and B, denoted with $n(A)$ and $n(B)$, present the noise detected by the classification filter.

Brodley and Friedl (1999) used decision trees, nearest neighbor, and linear machines for noise filtering. In our implementation of classification filters we replaced these simple classifiers by other better performing classifiers. From a variety of available supervised learning algorithms we chose a small and diverse sample of algorithms, each of them employing a different approach to learning: statistics (Naïve Bayes); information gain, sampling and voting (Random Forest); maximum-margin hyperplane computation in high-dimensional feature spaces (Support Vector Machine), and optimization of computational models (networks) by cost function/error minimization (Neural Network). The reason for choosing diverse classifiers is to ensure good performance of ensemble-based noise detection by NoiseRank.

In summary, the following four different classifiers were used in our work:

Bayes: Naïve Bayes (used as a baseline classifier),

RF: Random forest with 500 decision trees (RF500),

SVM: Support vector machine with a radial basis kernel function,

NN: Multilayer Perceptron - feedforward artificial neural network.

In the implementation of NoiseRank, described in detail in Section 6.1.1, we used the existing implementations of Bayes, RF and SVM from Orange (Demšar et al., 2013) and the NN from Weka (Hall et al., 2009).

Saturation-based noise detection methods

In order to contribute to the diversity of noise detection algorithms in the ensemble used by NoiseRank, we selected also the algorithms for explicit noise detection exploiting the notion of a *saturated* training set, proposed by Gamberger and Lavrač (1997). A saturated training set is a subset of training examples which enables the induction of a simple hypothesis correctly capturing the regularity/concept represented by the data. A non-saturated training set can be transformed into a saturated one by the elimination of noisy examples.

A saturation-based approach to noise filtering, called the *saturation filter*, recognizes noisy examples as those whose elimination enables the reduction of the complexity of the induced hypothesis measured by the Complexity of the Least Complex correct Hypothesis (*CLCH* value) of the reduced training set (Gamberger and Lavrač, 1997).

A saturation filter is constructed from two basic methods. The first one is a *saturation test*. At first it computes the complexity of the classification model for the given training set, then it iteratively excludes one training example and computes the complexity of a classification model induced from the rest of the training examples. The examples which have the greatest effect on reducing the complexity of the classification model by their exclusion are labeled as the *most noisy* and are passed on to the second method. The second method, the *noise filter*, randomly chooses one among the most noisy examples and excludes it from the training set, while other examples are returned to the example set. This is repeated as long as the saturation test finds noisy examples, meaning that a saturated subset has not yet been obtained.

A practical problem of the saturation filter implementation is to find a complexity measure for the induced classification model which corresponds to the *CLCH* value and is sensitive enough to enable the noise filtering algorithm to distinguish between the noisy and the non-noisy instances. While Gamberger and Lavrač (1997) use decision rules for modeling the data, in our reimplementation of saturation noise filtering algorithms used in NoiseRank, an unpruned decision tree learner is used to construct the models and the number of all decision tree nodes is used as the *CLCH* measure of model complexity. Hence, the saturation filtering algorithm has to construct as many decision trees as there are instances in the dataset and this is repeated as many times as there are potentially noisy instances. Consequently, this saturation filter implementation, named *SF*, is very time consuming. Therefore we also tested an implementation, named *PruneSF*, which prunes all the leaves of the decision tree supporting only one instance of the first constructed model, removes these instances, and only then starts the filtering phase. This results in a speed up to the existing saturation filter implementation.

In summary, the following two saturation filters were used in our work:

SF: Excludes data instances that have the greatest effect in reducing the complexity of a classification model,

PruneSF: Prunes all the leaves of the first constructed decision tree model supporting only one instance, removes these instances, and only then starts the *SF* phase.

Note that, unlike SF, PruneSF is unable to handle datasets with missing values, which reduces its applicability in real-life domain. Therefore, PruneSF was not used in the application of NoiseRank to a medical domain, described in Section 5.1.

3.2.3 NoiseRank results visualization

The features of the NoiseRank methodology were introduced in Section 3.1 and an example set of diverse noise detection algorithms was described in the previous section. Now the output of the NoiseRank methodology can finally be presented. In order to show the expert a set of potentially noisy instances we decided for a visual approach. The proposed method of ranking the detected noisy instances according to the number of elementary noise detection approaches agreeing that an instance is noisy is shown in Figure 3.3. The figure illustrates the output of NoiseRank in the real-life medical problem of coronary heart disease (CHD) patient analysis, described in detail in Section 5.1. Ranking of the detected instances should help the expert to focus on the instances that are potentially interesting for further analysis, either in terms of erroneousness (noisy instances) or in terms of their

atypicality (outliers). The interactive ranking visualization which allows the user to select and explore the detected noisy instances will be described in more detail in Chapter 6, covering the topic on the implementation and public availability of the developed approach.

```
Rank |  Class   | ID | Detected by:
-------------------------------------------------------------------
  1.   non-CHD    51  __Bayes______RF______SVM_______NN_______SF___
-------------------------------------------------------------------
  2.   CHD       229  ____RF______SVM_______NN_______SF___
-------------------------------------------------------------------
  3.   CHD         0  ___SVM_______NN_______SF___
  3.   non-CHD    27  ____RF_______NN_______SF___
  3.   non-CHD    39  __Bayes_____SVM_______NN___
  3.   CHD       176  __Bayes_____SVM_______NN___
  3.   CHD       194  __Bayes_____SVM_______NN___
  3.   CHD       213  ____RF______SVM_______NN___
-------------------------------------------------------------------
  4.   CHD        42  ___SVM_______NN___
  4.   non-CHD   120  __Bayes_____SVM___
  4.   non-CHD   164  __Bayes______RF___
  4.   non-CHD   173  ____RF_______SF___
  4.   CHD       196  __Bayes_____SVM___
  4.   non-CHD   226  ____RF_______SF___
-------------------------------------------------------------------
  5.   non-CHD    30  ___SVM___
  5.   CHD        45  ____NN___
  5.   non-CHD    59  ____RF___
  5.   non-CHD    62  __Bayes__
  5.   non-CHD    63  ____SF___
  5.   CHD        67  ___SVM___
  5.   non-CHD    72  ___SVM___
  5.   CHD        97  ____SF___
  5.   non-CHD   135  ___SVM___
  5.   non-CHD   177  ____NN___
  5.   CHD       181  __Bayes__
  5.   non-CHD   189  ___SVM___
  5.   CHD       195  ___SVM___
  5.   CHD       200  ____NN___
  5.   CHD       205  ___SVM___
  5.   CHD       231  __Bayes__
```

Figure 3.3: Visual representation of noise detection output. Instances are ranked by the number of noise detection algorithms which identified them as noise. Note that instance IDs are enumerated by zero-based indexing, i.e., starting the count with 0.

3.3 HARF

This section presents the High Agreement Random Forest noise detection approach, called HARF, introduced by Sluban et al. (2010a). It describes the reasons and the idea that led

to the development and application of this algorithm. The section concludes by explaining the HARF algorithm's relevance to this thesis.

3.3.1 Algorithm design

The idea for this simple and efficient approach to noise detection emerged during the study of ensembles for noise detection, ensemble composition with respect to the diversity of ensemble members, and especially while reviewing existing ensemble combination rules. The design of the High Agreement Random Forest noise detection approach (HARF) to noise detection was motivated by the following observations.

The first observation was that classification filters using Random Forest (RF) classifiers (actually) act like an ensemble noise filter. They present homogeneous ensembles composed of randomly generated decision trees, and may be therefore interpreted as individual classification filters using a decision tree learner. However, a classification filter using a RF classifier is not the same as an ensemble of classification filters using a decision tree learner. The important difference is in the "key to success" of the RF classifier. For good ensemble performance the members have to be diverse. In contrast to an ensemble of decision tree noise filters, each decision tree classifier in the RF is learned on a bootstrap sample[1] of the dataset to achieve greater diversity among the members of the ensemble.

The second observation concerned the combination rules that are used for obtaining the final prediction of an ensemble. Different methods for combining the predictions of the members in an ensemble were mentioned already in Section 3.1. Since random forest classifiers use the majority voting scheme to produce their final decision on a data instance, let us consider only the voting combiners. Consensus voting requires all ensemble members to agree on a particular label for it to be chosen as the final decision of the ensemble. The consensus voting scheme is considered to be conservative in the sense that it does not allow any deviations in the predictions for a data instance. This can enhance the performance in terms of precision, but on the other hand it may limit the ensemble's overall performance by refusing to accept that certain ensemble members make prediction errors or are unable to produce the same prediction. Majority voting schemes are more "liberal", therefore they are widely used for combining the predictions of ensemble members (Lam and Suen, 1997; Brodley and Friedl, 1999; Lin et al., 2003; Ruta and Gabrys, 2005; Khoshgoftaar et al., 2005; Mu et al., 2009). They allow some prediction errors, but make their final decision based on the majority of votes.

Majority voting in noise detection ensembles identifies a data instance as noisy if more than half of the ensemble members detect it as noise. Particularly, a classification filter using a RF learner identifies a data instance as noisy if more than half of the decision trees misclassified it. However, if the class which got the maximal number of votes won only by a single vote (over the 50%-50% voting outcome), this result is not reliable, since the decision trees are generated randomly. Therefore, demanding a higher agreement of classifiers (decision trees) when deciding if an data instance is noisy or not, seems as a very natural way to proceed. Already Xu et al. (1992) suggested a more flexible voting scheme for combining classifier predictions, and Khoshgoftaar et al. (2005) examined the influence of different levels of noise filtering (i.e., different levels of agreement in the noise filtering ensemble) on the predictive accuracy of classifiers.

In the standard classification filter using the RF learner a data instance is identified as noise if more than half of the decision trees misclassify it. Hence the decision of noise identification is based on the number of decision trees in the RF that misclassify an instance,

[1] For a given dataset $\mathcal{D}$ of size n, a *bootstrap sample* is a subset of $\mathcal{D}$ obtained by randomly sampling $n' \leq n$ elements with repetition.

or more precisely the sum of misclassifications has to exceed the "50% × ensemble size" threshold of the combination rule that is used to produce the final decision.

The HARF noise detection approach demands a higher agreement among classifiers (decision trees) on the misclassification of a data instance, i.e., an increased noise detection threshold in the voting process. The formal definition of the HARF algorithm is as follows.

Definition 1. The High Agreement Random Forest noise detection approach, called HARF, is a classification noise filter employing a random forest (RF) learning algorithm and using an increased noise identification threshold $T,\ 0.5 < T < 1$. Let n be the size of the ensemble (i.e., RF), $\mathcal{D}$ be a labeled dataset with classes $1, \ldots, k$, $c_j(i)$ be the number of classifiers from RF that classified data instance i into the class j, and let $l(i)$ denote the true label of instance i. For a threshold $T,\ 0.5 < T < 1$, an instance $i \in \mathcal{D}$ that satisfies

$$\sum_{\substack{j=1,\\ j \neq l(i)}}^{k} c_j(i) \geq T \cdot n\,, \quad \text{or equivalently} \quad c_{l(i)}(i) < (1-T) \cdot n\,,$$

is declared noisy by the HARF noise detection algorithm.

The definition states that the HARF algorithm with an increased noise identification threshold $T,\ 0.5 < T < 1$, i.e., demanding high agreement among ensemble members, declares those instances as noisy which were misclassified by at least $T \cdot n$ classifiers of the RF, or were correctly classified by less than $(1-T) \cdot n$ classifiers of the RF.

3.3.2 Threshold parameter setting

In this thesis, the HARF noise detection algorithm is used in the classification filtering manner with the RF classifier with 500 decision trees. Four different noise identification thresholds T were tested: $0.6, 0.7, 0.8$ and 0.9. These thresholds used by the HARF algorithm correspond to four required agreement levels among decision trees in the RF: 60%, 70%, 80% and 90%, and hence the HARF algorithm with one of these agreement levels will be referred to as HARF-60, HARF-70, HARF-80 and HARF-90, respectively.

At this point only the initial performance results are presented to select the most appropriate noise identification threshold(s) to be used in further applications of HARF. These results should also show that the assumptions regarding ensemble methods and the demand for higher agreement in the voting-based decision rule combiners justify the design of the HARF noise detection algorithm. The preliminary performance results were obtained on four datasets, as presented in Section 4.2, and averaged over ten repetitions with 5% randomly introduced noise. Details on the empirical evaluation of noise detection algorithms and the experimental settings will be described in Chapter 4. Figures 3.4, 3.5 and 3.6 show the relation between the required agreement level for noise identification and the precision, recall, and F-score results of noise detection, respectively. At agreement level 50% results of the standard RF are shown. The preliminary results indicate that increasing the noise identification threshold, i.e., demanding higher agreement of misclassification to declare an instance as noisy, as expected increases the precision of noise detection (Figure 3.4) and decreases the noise recall (Figure 3.5). However, the trade-off between precision and recall, as observed in the F-scores results (Figure 3.6), seems to favor T values between 0.7 and 0.8.

In the forthcoming chapters of this thesis, the HARF noise detection algorithm will be presented as an alternative approach for precise noise detection, supported by qualitative as well as empirical results, and used as an example of a new algorithm whose noise detection performance is evaluated and compared to the performance of other existing approaches by the newly proposed visual performance evaluation techniques.

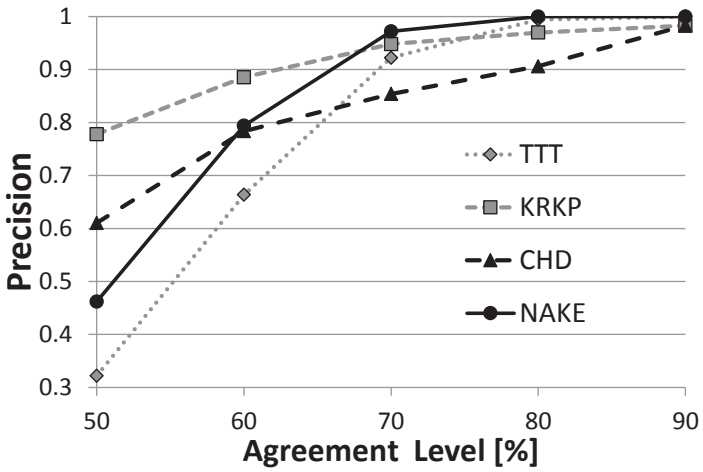


Figure 3.4: Precision of HARF noise detection based on the agreement level for RF on four domains with 5% injected noise.

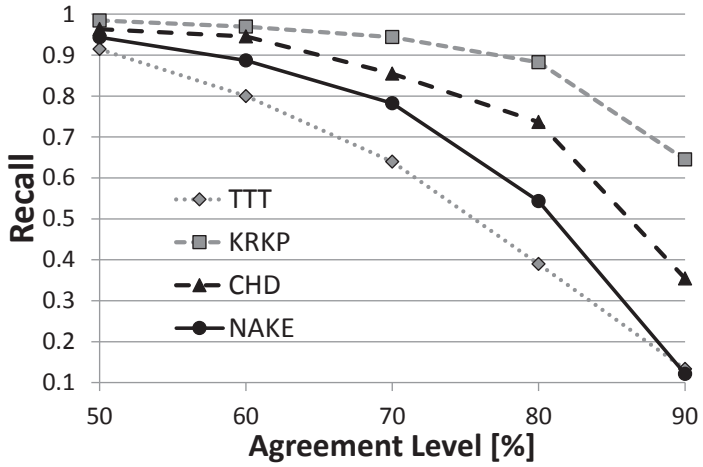


Figure 3.5: $F_{0.5}$-scores of HARF noise detection based on the agreement level for RF on four domains with 5% injected noise.

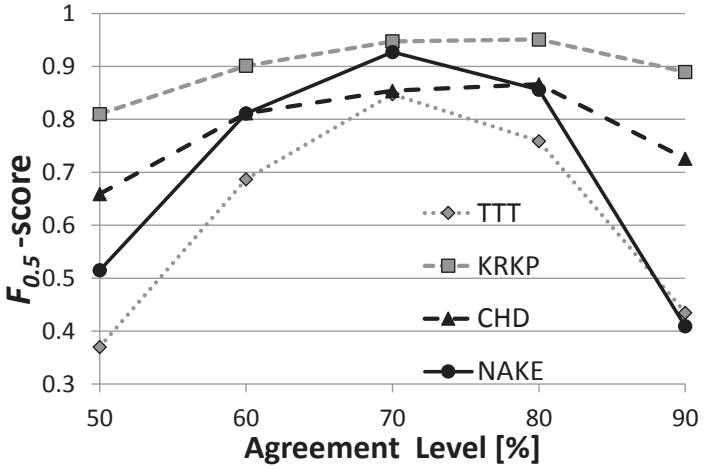


Figure 3.6: $F_{0.5}$-scores of HARF noise detection based on the agreement level for RF on four domains with 5% injected noise.

4 Visual Performance Evaluation

The performance of noise detection algorithms can be evaluated qualitatively or quantitatively by employing various approaches and performance measures. This chapter addresses empirical evaluation of explicit noise detection algorithms using visual performance evaluation approaches. Primarily the focus is on the evaluation of discrete noise detection approaches, which distinguish between noisy and non-noisy (regular) instances and return a set of instances identified as noisy.

First, the motivation for developing a new approach to visual performance evaluation is presented. Next, a description of a simple experimental setting, designed to show how the selected noise detection algorithms can be evaluated, is provided. This setting is used to illustrate the standard performance evaluation approach followed by the presentation of a new approach to visual performance evaluation. Finally, the results of all the evaluation approaches are compared in the described experimental setting.

4.1 Motivation for visual performance evaluation

Developing a new noise detection algorithm presents a substantial amount of developer's work and the noise detection performance it achieves is the main indicator of its successful future application. Therefore, easily understandable and comprehensive presentation of performance results can significantly support the developer in the evaluation and comparison of the results.

Evaluating the performance of noise detection algorithms requires either qualitative examination by domain experts, as will be presented in Chapter 5, or quantitative analysis of noise detection results on data with known or artificially injected noisy instances. The goal of this chapter is to present the developed methods for quantitative performance evaluation, motivated by the following observations.

- First, in the literature, the performance of noise handling approaches has been usually addressed in the context of classification problems and evaluated by its effect on classification performance (Brodley and Friedl, 1999; Gamberger et al., 2000; Zhu and Wu, 2004; Khoshgoftaar et al., 2006; Libralon et al., 2009; Gavriluţ and Ciortuz, 2011; Cai et al., 2011). However, classification performance measures are not adequate measures for evaluating the success of explicit noise detection.

- Second, the actual performance of noise detection can be quantitatively evaluated on domains with annotated or injected noise by the *precision* of noise detection and the *recall* of noisy instances. These two measures known from information retrieval are widely used for noise detection evaluation in addition to other prediction measures (Verbaeten, 2002; Verbaeten and Van Assche, 2003; Zhu et al., 2003; Miranda et al., 2009; Kriegel et al., 2009; Hido et al., 2011; Dang et al., 2013) or just as two separate detection measures (Khoshgoftaar et al., 2004; Van Hulse and Khoshgoftaar, 2006). Their joint or aggregated presentation, on the other hand, has not been reported in the noise handling literature, except for precision-recall curves used for

evaluating ranked noise retrieval results by Zhong et al. (2005) and the F-measure used by Sluban et al. (2010a,b).

- Third, performance results of noise detection are typically presented in tabular format, which proves to be difficult for presenting multiple performance measures at once as well as to compare the performance results of several algorithms. Graphically presenting the results by a bar chart can ease the comparison, but depicting more than one performance measure reduces the chart's understandability.

These observations motivated the development of the VIPER performance evaluation methodology, which:

1. addresses noise detection performance directly by measuring the precision and recall of noise detection on data with known or injected noisy instances,

2. integrates two new evaluation methods that trade off precision and recall: *F-isolines* and the *ε-proximity* evaluation methods, and

3. jointly visualizes these evaluation methods in the two-dimensional *precision-recall space*.

A simplified example of the VIPER visual performance evaluation approach is presented in Figure 4.1.

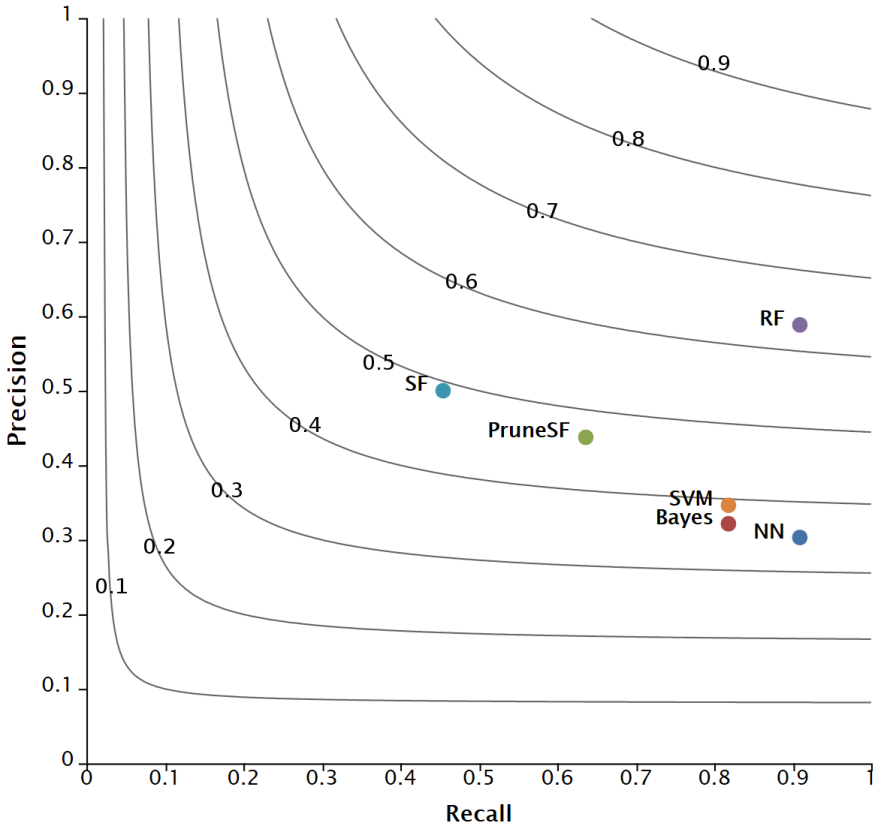


Figure 4.1: Performance results of different noise detection algorithms (presented in Section 3.2.2) in the precision-recall space on the CHD dataset with 5% injected class noise, clearly showing that the RF algorithm outperforms other algorithms in terms of precision (Y-axis) and the F-measure (F-isolines are presented as curves in the precision-recall space).

In contrast to Zhong et al. (2005) who use precision-recall curves for the evaluation of ranked instances retrieved by their noise filtering algorithm, the VIPER methodology aims at the evaluation and comparison of noise detection performance in terms of precision, recall and the F-measure results achieved by different noise detection algorithms. The VIPER methodology presents a novel kind of visual performance evaluation in the precision-recall space, which has—to the best of our knowledge—not been proposed before neither in the noise handling literature nor in the literature presenting the evaluation of information extraction approaches. By employing two new evaluation methods, VIPER enables intuitive evaluation, interpretation and comparison of noise detection performance in terms of the three performance measures at the same time. This visual performance evaluation methodology can also be used for evaluating information retrieval and entity recognition algorithms, as well as for any other algorithms for which the evaluation in terms of precision, recall and the F-measure is most suitable.

4.2 Experimental setting

The performance of different noise detection approaches was evaluated on two noiseless and two real-world datasets, specified in Table 4.1. All the datasets were injected with three different levels of class noise: 2%, 5% and 10%, meaning that the particular percentage of instances of a dataset was randomly selected and the class values were replaced.[1]

Intentionally two artificial noiseless datasets were chosen, Tic Tac Toe (TTT) and King-Rook vs. King-Pawn (KRKP) from the UCI machine learning repository (Bache and Lichman, 2013), as a starting point to evaluate the noise detection algorithms' ability to detect the injected noise. The other two datasets are real-world datasets, containing their own noisy instances: a medical coronary heart disease (CHD) dataset[2], described in Section 5.1.1, and a dataset of News Articles on Kenyan Elections (NAKE), described in Section 5.2.1. All the datasets are two class domains.

Table 4.1: Datasets used in the quantitative evaluation of noise detection algorithms.

Dataset		Instances	Attributes	Type
TTT:	Tic Tac Toe	956	9	noiseless
KRKP:	King-Rook vs. King-Pawn	3,196	36	noiseless
CHD:	Coronary Heart Disease	219	37	real-world
NAKE:	News Articles on Kenyan Elections	464	500	real-world

For each of the four datasets and for each noise level, ten datasets with randomly injected noise were generated. This was done to get more reliable noise detection results, meaning that each result presented in this work for a specific domain and a specific noise level is the average value of the results obtained from ten separate experiments, each on a dataset with different randomly injected noise. In this way each noise detection algorithm was tested not only on four different datasets but actually on 120 datasets, allowing for massive testing of noise detection algorithms despite the relatively small number of selected domains.

Performance evaluation was performed on all elementary noise detection algorithms presented in Section 3.2.2: the classification noise filters (Bayes, RF, SVM and NN) and the

[1] Alternatively, attribute noise could have been inserted, but we were motivated by medical domains where false diagnostics is the main concern. [2] The original CHD dataset includes also 19 instances with missing values, which were removed in the experiments described in this section in order to be able to apply two different saturation filters.

saturation noise filters (SF and PruneSF). Additionally, a number of their ensembles was constructed and evaluated as well. To be more convinced that an instance is noisy, ensemble noise filtering approaches take into account predictions of several different noise filtering algorithms when deciding if an instance should actually be declared as noisy. To increase the diversity among the voters in the ensemble we constructed our ensemble filters from a combination of classification and saturation filters, forming different groups of filtering algorithms. First, we constructed three different ensembles consisting of classification filters:

Ens1: Bayes, RF, SVM,

Ens2: RF, SVM, NN,

Ens3: Bayes, RF, SVM, NN.

RF and SVM were used in all three ensembles, since they are generally accepted as good performing classification algorithms. Bayes and NN, on the other hand, were not used in Ens1 and Ens2, respectively, to see how less favorably performing members of the ensemble affect the ensemble's performance. To observe the influence of saturation filters in the ensembles we constructed ensembles **Ens4**, **Ens5** and **Ens6** by adding SF to Ens1, Ens2 and Ens3, respectively. Like SF, also PruneSF was used to construct ensembles **Ens7**, **Ens8** and **Ens9** from Ens1, Ens2 and Ens3, respectively. The last ensemble **Ens10** is formed from the group of all elementary noise filters explored in this work: Bayes, RF, SVM, NN, SF and PruneSF. All ten ensemble noise filters were used in the consensus voting scheme, denoting an instance as noisy only if all members in the ensemble identified it as noisy. An overview of all constructed ensembles is provided in Table 4.2.

Table 4.2: Ensembles constructed from classification and saturation filters.

Ensemble	Members
Ens1	Bayes, RF, SVM
Ens2	RF, SVM, NN
Ens3	Bayes, RF, SVM, NN
Ens4	Bayes, RF, SVM, SF
Ens5	RF, SVM, NN, SF
Ens6	Bayes, RF, SVM, NN, SF
Ens7	Bayes, RF, SVM, PruneSF
Ens8	RF, SVM, NN, PruneSF
Ens9	Bayes, RF, SVM, NN, PruneSF
Ens10	Bayes, RF, SVM, NN, SF, PruneSF

In addition, to illustrate how a novel noise detection algorithm can be evaluated both in the standard setting and in the VIPER visual performance evaluation setting, the HARF noise detection algorithm, presented in Section 3.3, was included in the experimental evaluation. To this end, two variants of HARF were used, the ones that proved to perform best in the (preliminary) evaluations, as well as in the real life use cases in Chapter 5. These were HARF-70 and HARF-80, which require at least 70% or 80% of classifiers in the ensemble to misclassify an instance in order to denote it as noisy.

4.3 Standard performance evaluation methods

Quantitative performance evaluation of noise detection can be performed only on the data for which the evaluator knows which instances are noisy and which are not. This can be done on data with artificially inserted noise or on data with known noisy instances. This section presents the performance measures that are used for empirical evaluation of explicit noise detection on data with labeled noisy instances. Unlike the common evaluation of noise detection performance through its influence on classification accuracy, a more adequate approach is the one which is standardly used in information retrieval (Manning et al., 2008): using precision and recall of correctly detected noisy instances.

4.3.1 Basic performance measures

A basic measure to be used in the quantitative evaluation of noise detection methods is *precision*, defined as true positives divided by all the predicted positives (i.e., the percentage of correctly detected noisy instances among all the instances identified as noisy by the noise detection algorithm).

$$Precision = \frac{number\ of\ true\ noisy\ instances\ detected}{number\ of\ all\ instances\ identified\ as\ noisy}$$

Another useful measure is *recall*, which is defined as true positives divided by all the positives (i.e., the percentage of correctly detected noisy instances among all the noisy instances inserted into the dataset as random noise).

$$Recall = \frac{number\ of\ true\ noisy\ instances\ detected}{number\ of\ all\ noisy\ instances\ in\ the\ dataset}$$

To model a desired precision-recall trade-off, the so-called F-measure combining precision and recall is used. The general formula for computing the F-measure is the following:

$$F_\beta = (1 + \beta^2)\frac{Precision \cdot Recall}{\beta^2 Precision + Recall} \tag{4.1}$$

where for $\beta = 1$ we get the standard F-measure, also referred to as the F_1 score. By setting the β parameter, the user can assign more importance to either precision or recall in the computation of the F-score.

In some specific use cases other performance measures derived from the confusion matrix may be of interest when evaluating noise detection performance, like the False positive rate (FPR) or the False negative rate (FNR).

4.3.2 Performance visualization of continuous noise detection outputs

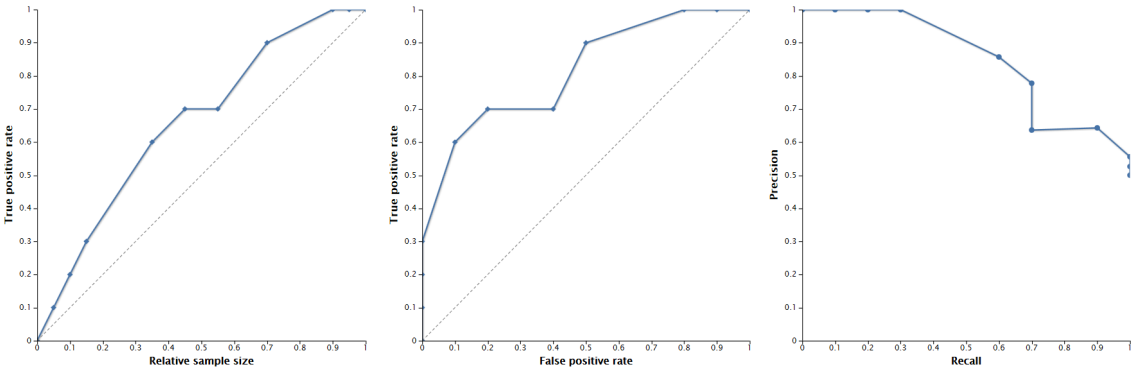
Evaluating the performance of algorithms returning continuous noise detection output, like scores, probabilities, or ranks, is usually reduced to the problem of evaluating a discrete noise detection algorithm by selecting a threshold on the noise prediction scores or ranks for noise identification. Hence, the performance is computed for a range of different feasible thresholds and these results are presented either aggregated or only the best performing results at a specific threshold are selected. Such approaches are common in machine learning and data mining for the evaluation of classification algorithms.

In the noise handling literature, on the other hand, visual performance evaluation techniques for continuous noise detection outputs are rarely used. Notable exceptions are precision-recall curves used by Zhong et al. (2005) for the evaluation of ranking outputs

and the ROC curve analysis performed by Angiulli et al. (2006); Kriegel et al. (2009); Dang et al. (2013) for the evaluation of outlier scores.

Feasible performance visualizations of continuous noise detection outputs are those used in information retrieval, where labeled noisy instances are the 'target' to be retrieved, and those used in the evaluation of classification algorithms, where a noise detection algorithm basically presents a binary classifier for noise and non-noise classification. The visualizations process a list of instances ordered according to the noise detection score or rank assigned to each instance.

The simplest visualization is the *lift chart* (Witten and Frank, 2005), which plots the true positives rate (recall) in relation to the relative sample size of first ranked instances in the ordered input list. Another visualization, commonly used in machine learning, which originates from signal theory, is the receiver operated curve or the ROC chart, which plots the true positive rate relative to the false positive rate (rate of regular instances incorrectly detected as noise) according to a changing decision threshold (Fawcett, 2006; Flach, 2010). However, in our opinion, the most interesting performance visualization for continuous noise detection results is by *precision-recall curves* (PR curves), which show the precision of noise detection relative to the recall of noise according to a specific decision threshold (Manning et al., 2008). Examples of these performance visualizations are shown in Figure 4.2.



Lift curve ROC curve PR curve

Figure 4.2: Examples of performance visualizations for continuous noise detection outputs.

In the evaluation of classification algorithms also other performance visualizations are used. They show the expected loss as a function of skew (fraction of positive examples multiplied by the cost of misclassifying a positive example), like cost curves (Drummond and Holte, 2006), rate-driven curves, or Kendall curves (Hernández-Orallo et al., 2013). However, these visualizations are out of the scope of this thesis in terms of noise detection performance evaluation.

4.3.3 Statistical significance of performance results

Statistical tests are used to verify the significance of the differences between the performance of competing noise detection algorithms. For comparing the performance of several algorithms over multiple datasets, the Friedman test is suggested (Demšar, 2006).

The Friedman test (Friedman, 1940) is a non-parametric statistical test that ranks algorithms for each experimental dataset separately. Ranks are assigned from 1 to k, for k algorithms, where the best algorithm gets the rank 1, the second best rank 2, etc. In case of ties, average ranks are assigned. The test compares the average ranks of algorithms achieved over the experimental datasets, under the null-hypothesis that all algorithms perform equally and therefore also their ranks are equal.

In the case when the null-hypothesis is rejected, additional post-hoc tests can be performed to reveal the significant differences in algorithm performance. Such are the Nemenyi test (Nemenyi, 1963) for comparing all algorithms to each other, and the Bonferroni-Dunn test (Dunn, 1961) for the comparison of a control algorithm to all other algorithms. Demšar (2006) proposed a visual representation of the post-hoc analysis when several algorithms are compared. The *critical difference diagrams* (CD diagrams) present the order of the algorithms in terms of their average ranks, the magnitude of the differences between them (in terms of ranks), and the significance of the observed ranks. An example of the CD diagram for the Nemenyi test is presented in Figure 4.3.

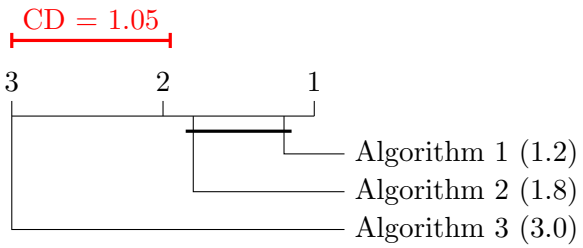


Figure 4.3: Example CD diagram for the Nemenyi test, $\alpha = 0.05$.

Algorithms that are further apart than the specified critical difference (CD) perform statistically significantly differently. The thick black lines connect the groups of algorithms that do not perform significantly differently. An example of the CD diagram for the Bonferroni-Dunn test with the control Algorithm 2 is presented in Figure 4.4.

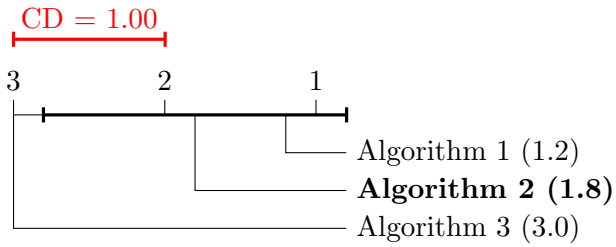


Figure 4.4: Example CD diagram for the Bonferroni-Dunn test, control Alg. 2, $\alpha = 0.05$.

The critical difference for the Bonferroni-Dunn test is different from the one for the Nemenyi test and is depicted to the left and to the right of the selected control algorithm in the CD diagram. It shows that Algorithm 2 performs significantly differently from Algorithm 3, but not from Algorithm 1.

4.4 Performance evaluation with VIPER

This section describes the evaluation of experimental results by the VIPER methodology for visual performance evaluation. In contrast with tabular and simple graphical presentation of results in standard evaluation practice, this methodology visualizes performance results in the precision-recall space enabling intuitive interpretation and simultaneous comparison of three performance measures. First, two new evaluation methods are described followed by the VIPER visualization methodology for evaluation of noise detection algorithms.

4.4.1 *F*-isolines and ε-proximity methods

In this section we present the two evaluation methods which enable the comparison of noise detection performance of different algorithms by modeling their precision-recall trade-off.

_F_-isoline evaluation method

To compare the interaction of different performance measures of noise detection all at once, this method proposes to depict the F-measure values as isolines (contour lines) in the precision-recall space. *F-isolines* are curves in the precision-recall space that satisfy the following condition:

$$F_\beta = (1+\beta^2)\frac{Precision \cdot Recall}{\beta^2 Precision + Recall} = C \ , \tag{4.2}$$

and present all the points in the precision-recall space that have their F_β score equal to the value of C, where $C \in (0,1)$. This allows for intuitive visual interpretation of algorithm performance and its comparison to other noise detection algorithms not only in terms of precision and recall but at the same time also in terms of the F-score.

Additionally, in this way equivalence classes of noise detection algorithms that achieve a similar F-score can be made and thus groups of algorithms with similar noise detection capabilities can be identified.

ε-proximity evaluation method

The designed measure for best noise detection algorithm selection does not insist on maximal precision (Pr) of noise detection at the cost of low recall (Re). Instead, it trades off high precision by maximizing the recall of noise detection in the ε-neighborhood of the most precise noise detection algorithms, where ε is a (small) tolerated proximity of the maximal possible precision achieved by different noise detection algorithms $A_j \in \mathcal{A}$, where $\mathcal{A}$ is the set of all used noise detection algorithms. The formal description of selecting the best noise detection algorithm (abbreviated NDA) according to the ε-proximity evaluation method is provided in Equation (4.3).

$$\begin{aligned} BestNDA(\mathcal{A}) &:= \arg\left(\max_{A_i \in \mathcal{A}\ ;\ Cond}[Re(A_i)]\right) \\ Cond &: \quad Pr(A_i) > \max_{A_j \in \mathcal{A}}[Pr(A_j)] - \varepsilon \ . \end{aligned} \tag{4.3}$$

4.4.2 Viper visualization methodology

The two evaluation methods presented in Section 4.3.3 are jointly used in a novel VIsual PERformance (Viper) evaluation approach. In this approach, noise detection performance results are presented in the precision-recall space, a two-dimensional space where one dimension depicts *recall* values and the other *precision* values, in the following way:

- The noise detection performance result of a noise detection algorithm on a specific domain is presented as a point in the precision-recall space.
- According to the *F-isoline evaluation method*, equal F-scores are depicted in the form of isolines, which enables the comparison of F-measure results of noise detection algorithms in the precision-recall space.
- The ε-proximity of the maximal achieved precision of noise detection is emphasized to enable easy identification of the algorithm with maximal recall as the best performing algorithm in the *ε-proximity* of the algorithm achieving maximal precision of noise detection.

The VIPER visualization enables intuitive interpretation and comparison of performance results of different noise detection algorithms on a selected domain. By integrating the *F-isoline evaluation method* into the precision-recall space, three basic performance evaluation measures (presented in Section 4.3) can be observed at the same time in a two-dimensional space. F-isolines with values $0.1, 0.2, \ldots, 0.9$ can serve as grid lines for the F-measure values, whereas additional F-isolines intersecting specific algorithm performance points may be included to compare the performance among individual noise detection algorithms. The *ε-proximity evaluation method* formally described with Equation 4.3 becomes also easily understandable with its visualization in the precision-recall space. An example of the proposed visualization is presented in Figure 4.5.

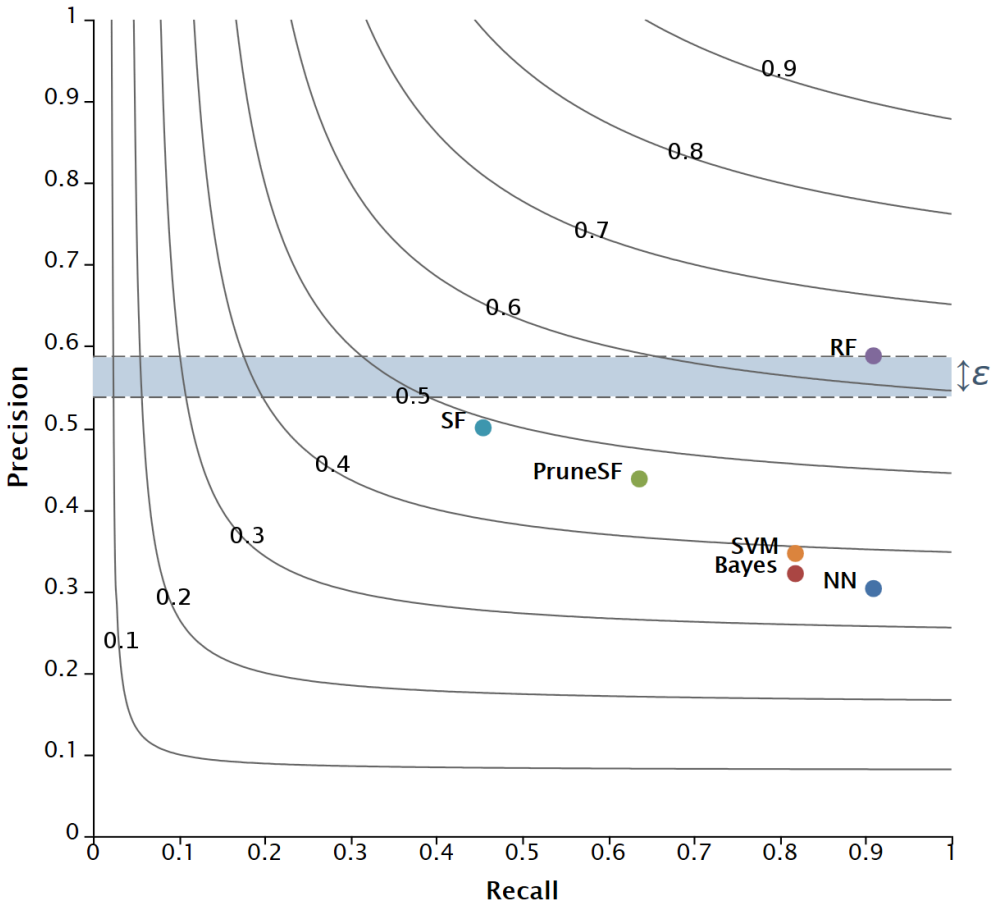


Figure 4.5: Performance results of different noise detection algorithms visualized according to the VIPER methodology in the precision-recall space.

4.5 Experimental results

This section presents the empirical results obtained from noise detection experiments evaluated by performance approaches described in the previous Sections 4.3 and 4.3.3. First, the standard noise detection evaluation practice and the corresponding results are shown in Section 4.5.1. Second, in comparison, the proposed VIPER performance evaluation is presented in Section 4.5.1 to depict the benefits of the intuitive visual performance evaluation approach. Finally, some performance visualizations of quantitative noise ranking results by NOISERANK and HARF are presented in Section 4.5.3.

4.5.1 Standard evaluation of performance results

The evaluation of elementary noise detection algorithms, their ensembles and the HARF noise detection algorithm was performed according to the experimental settings described in Section 4.2 and by using the performance measures described in the Section 4.3. Since precise noise detection is desired by the experts who get to evaluate and identify the output of noise detection algorithms, more importance was attached to precision than to recall in the computation of the F-measure. Therefore the $F_{0.5}$-score performance measure was used for the quantitative evaluation of noise detection algorithms (obtained by using $\beta = 0.5$ in Equation 4.1). The experiments were performed on four datasets with three different levels of randomly injected class noise, each repeated ten times.

Two standard ways of presenting noise detection performance results are presented. First, the tabular results presentation and the accompanying discussion of results is shown, followed by a basic graphical presentation of the same results and an interpretation of results from this graphical presentation.

Tabular results presentation

This section presents and discusses the performance results of different noise detection algorithms and their ensembles evaluated on four domains with three different levels of randomly injected class noise. The precision (Pr) and $F_{0.5}$-score results in Tables 4.3 and 4.4 present averages and standard deviations over ten repetitions of noise detection on a given domain with a given level of randomly injected class noise. Table 4.3 presents performance results of the elementary and the HARF noise detection algorithms, whereas Table 4.4 shows the performance results of the ensemble filters. The best precision and the best $F_{0.5}$-score results in each table are printed in bold.

Once the tabular results have been presented, standard evaluation practice requires elaborate interpretation of these results. To evaluate and compare these results, it is necessary to inspect the entire table(s), which can be quite demanding. The following discussion illustrates the step by step exploration of tabular results presentations.

The results in part (A) of Table 4.3 show that among the classification filters the RF and SVM filters generally perform best, RF on the KRKP and CHD datasets, SVM on the TTT dataset and both RF and SVM on the NAKE dataset. The NN filter is quite good on NAKE at lower noise levels, but otherwise achieves only average performance. Noise detection with the Bayes classification filter shows overall the worst performance results, except for the CHD domain where it outperforms NN and is comparable to SVM.

The performance of the saturation filters presented in part (B) of Table 4.3 is between the worst and the best classification filters. Except for lower noise levels on the KRKP dataset, where SF even outperforms the best classification filter (RF), the saturation filters are not really comparable to the best classification filtering results. On the other hand, among the saturation filters, SF achieves higher or comparable (on the NAKE dataset) precision results like PruneSF, whereas the $F_{0.5}$-scores of PruneSF are not much lower, except for the KRKP dataset, which indicates that PruneSF has higher recall of injected class noise than SF.

Table 4.3: Precision and $F_{0.5}$-score results of elementary and HARF noise detection algorithms on 4 datasets at 3 levels of injected class noise.

	Noise	(A) Classification filters								(B) Saturation filters	
	Level	Bayes		RF500		SVM		NN		SF	
Dataset	(%)	Pr	$F_{0.5}$	Pr	$F_{0.5}$	Pr	$F_{0.5}$	Pr	$F_{0.5}$	Pr	$F_{0.5}$
TTT	2	0.04 ± 0.01	0.05 ± 0.01	0.15 ± 0.01	0.18 ± 0.02	0.54 ± 0.00	0.60 ± 0.00	0.34 ± 0.04	0.39 ± 0.04	0.19 ± 0.03	0.22 ± 0.03
	5	0.11 ± 0.01	0.14 ± 0.01	0.34 ± 0.03	0.39 ± 0.03	0.75 ± 0.01	0.79 ± 0.00	0.42 ± 0.03	0.47 ± 0.03	0.39 ± 0.05	0.42 ± 0.05
	10	0.20 ± 0.01	0.23 ± 0.01	0.46 ± 0.03	0.51 ± 0.03	0.87 ± 0.01	$\mathbf{0.89} \pm 0.00$	0.44 ± 0.02	0.49 ± 0.02	0.50 ± 0.05	0.51 ± 0.04
KRKP	2	0.13 ± 0.00	0.15 ± 0.00	0.60 ± 0.01	0.65 ± 0.01	0.19 ± 0.01	0.22 ± 0.01	0.54 ± 0.03	0.59 ± 0.03	0.81 ± 0.06	0.81 ± 0.06
	5	0.26 ± 0.01	0.31 ± 0.01	0.79 ± 0.02	0.83 ± 0.01	0.38 ± 0.00	0.43 ± 0.00	0.58 ± 0.02	0.63 ± 0.02	0.74 ± 0.05	0.72 ± 0.05
	10	0.42 ± 0.01	0.46 ± 0.01	0.87 ± 0.01	0.89 ± 0.01	0.57 ± 0.01	0.62 ± 0.01	0.56 ± 0.01	0.61 ± 0.01	0.61 ± 0.06	0.57 ± 0.06
CHD	2	0.22 ± 0.03	0.27 ± 0.03	0.37 ± 0.04	0.43 ± 0.04	0.19 ± 0.02	0.22 ± 0.02	0.17 ± 0.02	0.21 ± 0.03	0.32 ± 0.08	0.35 ± 0.09
	5	0.38 ± 0.04	0.42 ± 0.04	0.64 ± 0.04	0.69 ± 0.04	0.35 ± 0.05	0.40 ± 0.05	0.30 ± 0.06	0.34 ± 0.06	0.50 ± 0.06	0.48 ± 0.06
	10	0.51 ± 0.05	0.55 ± 0.06	0.73 ± 0.05	0.77 ± 0.04	0.53 ± 0.07	0.57 ± 0.07	0.39 ± 0.04	0.43 ± 0.04	0.60 ± 0.09	0.54 ± 0.08
NAKE	2	0.18 ± 0.02	0.22 ± 0.02	0.24 ± 0.02	0.29 ± 0.03	0.28 ± 0.03	0.32 ± 0.03	0.33 ± 0.04	0.38 ± 0.04	0.22 ± 0.10	0.25 ± 0.11
	5	0.36 ± 0.02	0.40 ± 0.02	0.52 ± 0.02	0.57 ± 0.02	0.48 ± 0.02	0.54 ± 0.02	0.47 ± 0.05	0.52 ± 0.05	0.37 ± 0.09	0.36 ± 0.08
	10	0.55 ± 0.04	0.60 ± 0.03	0.67 ± 0.03	0.71 ± 0.02	0.70 ± 0.03	0.73 ± 0.03	0.52 ± 0.03	0.57 ± 0.03	0.42 ± 0.12	0.36 ± 0.10
	Noise	(C) HARF with Agreement Level									
	Level	60%		70%		80%		90%		PruneSF	
Dataset	(%)	Pr	$F_{0.5}$	Pr	$F_{0.5}$	Pr	$F_{0.5}$	Pr	$F_{0.5}$	Pr	$F_{0.5}$
TTT	2	0.45 ± 0.09	0.49 ± 0.10	0.87 ± 0.12	$\mathbf{0.80} \pm 0.09$	$\mathbf{1.00} \pm 0.00$	0.73 ± 0.10	$\mathbf{1.00} \pm 0.00$	0.44 ± 0.11	0.14 ± 0.02	0.17 ± 0.02
	5	0.63 ± 0.09	0.66 ± 0.08	0.91 ± 0.05	$\mathbf{0.83} \pm 0.05$	$\mathbf{1.00} \pm 0.01$	0.76 ± 0.06	$\mathbf{1.00} \pm 0.00$	0.45 ± 0.09	0.26 ± 0.01	0.30 ± 0.02
	10	0.70 ± 0.07	0.71 ± 0.06	0.90 ± 0.05	0.81 ± 0.04	0.99 ± 0.01	0.71 ± 0.05	$\mathbf{1.00} \pm 0.00$	0.33 ± 0.05	0.35 ± 0.03	0.38 ± 0.03
KRKP	2	0.78 ± 0.03	0.82 ± 0.03	0.88 ± 0.02	0.90 ± 0.02	0.93 ± 0.01	$\mathbf{0.92} \pm 0.01$	$\mathbf{0.95} \pm 0.02$	0.89 ± 0.02	0.46 ± 0.02	0.51 ± 0.02
	5	0.88 ± 0.01	0.90 ± 0.01	0.94 ± 0.01	0.94 ± 0.01	0.97 ± 0.01	$\mathbf{0.95} \pm 0.01$	$\mathbf{0.98} \pm 0.00$	0.88 ± 0.01	0.48 ± 0.02	0.53 ± 0.02
	10	0.94 ± 0.01	0.94 ± 0.01	0.97 ± 0.01	$\mathbf{0.96} \pm 0.01$	$\mathbf{0.99} \pm 0.00$	0.95 ± 0.01	$\mathbf{0.99} \pm 0.00$	0.83 ± 0.01	0.50 ± 0.01	0.54 ± 0.01
CHD	2	0.54 ± 0.03	0.59 ± 0.04	0.65 ± 0.07	0.69 ± 0.08	0.80 ± 0.07	$\mathbf{0.81} \pm 0.08$	$\mathbf{0.98} \pm 0.08$	$\mathbf{0.81} \pm 0.11$	0.24 ± 0.04	0.27 ± 0.05
	5	0.80 ± 0.04	0.82 ± 0.04	0.82 ± 0.05	0.83 ± 0.05	0.92 ± 0.06	$\mathbf{0.89} \pm 0.05$	$\mathbf{1.00} \pm 0.00$	0.63 ± 0.13	0.44 ± 0.10	0.46 ± 0.10
	10	0.88 ± 0.06	0.88 ± 0.05	0.93 ± 0.04	$\mathbf{0.90} \pm 0.04$	0.96 ± 0.04	0.84 ± 0.05	$\mathbf{1.00} \pm 0.00$	0.47 ± 0.13	0.63 ± 0.06	0.63 ± 0.05
NAKE	2	0.63 ± 0.06	0.67 ± 0.06	0.90 ± 0.05	$\mathbf{0.88} \pm 0.06$	$\mathbf{1.00} \pm 0.00$	$\mathbf{0.88} \pm 0.07$	$\mathbf{1.00} \pm 0.00$	0.59 ± 0.17	0.15 ± 0.03	0.18 ± 0.04
	5	0.76 ± 0.05	0.78 ± 0.04	0.95 ± 0.05	$\mathbf{0.90} \pm 0.05$	$\mathbf{1.00} \pm 0.00$	0.84 ± 0.05	0.90 ± 0.30	0.39 ± 0.17	0.31 ± 0.05	0.34 ± 0.06
	10	0.89 ± 0.05	0.88 ± 0.04	0.99 ± 0.02	$\mathbf{0.91} \pm 0.03$	$\mathbf{1.00} \pm 0.00$	0.74 ± 0.07	0.90 ± 0.30	0.16 ± 0.08	0.45 ± 0.03	0.46 ± 0.03

Table 4.4: Precision and $F_{0.5}$-score results of different noise detection ensembles on four datasets at various levels of injected class noise.

Dataset	Noise Level (%)	Ens1 (B, RF, SVM)		Ens2 (RF, SVM, NN)		Ens3 (B, RF, SVM, NN)		Ens4 (B, RF, SVM, SF)		Ens5 (RF, SVM, NN, SF)	
		Pr	$F_{0.5}$	Pr	$F_{0.5}$	Pr	$F_{0.5}$	Pr	$F_{0.5}$	Pr	$F_{0.5}$
TTT	2	0.56 ± 0.05	0.57 ± 0.05	0.64 ± 0.08	0.67 ± 0.07	0.60 ± 0.09	0.61 ± 0.09	0.90 ± 0.09	0.72 ± 0.11	0.92 ± 0.10	0.78 ± 0.09
	5	0.80 ± 0.02	0.79 ± 0.03	0.84 ± 0.01	0.84 ± 0.01	0.84 ± 0.02	0.81 ± 0.02	0.97 ± 0.04	0.83 ± 0.05	0.97 ± 0.03	0.86 ± 0.05
	10	0.89 ± 0.02	0.83 ± 0.02	0.92 ± 0.01	$\mathbf{0.89} \pm 0.01$	0.91 ± 0.02	0.81 ± 0.03	0.97 ± 0.02	0.75 ± 0.05	0.97 ± 0.02	0.78 ± 0.03
KRKP	2	0.70 ± 0.02	0.72 ± 0.02	0.84 ± 0.02	0.85 ± 0.02	0.87 ± 0.03	0.86 ± 0.03	0.90 ± 0.03	0.85 ± 0.03	0.93 ± 0.02	0.88 ± 0.02
	5	0.87 ± 0.02	0.87 ± 0.02	0.92 ± 0.02	0.91 ± 0.02	0.94 ± 0.01	0.91 ± 0.01	$\mathbf{0.97} \pm 0.02$	0.84 ± 0.03	0.96 ± 0.02	0.86 ± 0.02
	10	0.93 ± 0.01	0.91 ± 0.01	0.95 ± 0.01	$\mathbf{0.93} \pm 0.01$	0.96 ± 0.01	0.92 ± 0.01	0.96 ± 0.01	0.71 ± 0.05	0.96 ± 0.02	0.73 ± 0.04
CHD	2	0.66 ± 0.10	0.70 ± 0.09	0.59 ± 0.12	0.63 ± 0.12	0.71 ± 0.09	0.73 ± 0.09	0.76 ± 0.18	0.69 ± 0.14	0.81 ± 0.16	0.72 ± 0.11
	5	0.87 ± 0.06	$\mathbf{0.84} \pm 0.06$	0.82 ± 0.04	0.81 ± 0.04	0.87 ± 0.06	0.83 ± 0.08	0.89 ± 0.12	0.69 ± 0.11	0.86 ± 0.11	0.67 ± 0.06
	10	0.87 ± 0.06	$\mathbf{0.85} \pm 0.05$	0.85 ± 0.06	0.83 ± 0.06	0.90 ± 0.06	$\mathbf{0.85} \pm 0.06$	0.90 ± 0.10	0.67 ± 0.08	0.92 ± 0.08	0.65 ± 0.08
NAKE	2	0.41 ± 0.03	0.45 ± 0.03	0.48 ± 0.04	0.53 ± 0.04	0.58 ± 0.06	0.62 ± 0.06	0.73 ± 0.15	0.63 ± 0.15	0.90 ± 0.17	0.72 ± 0.15
	5	0.67 ± 0.05	0.70 ± 0.05	0.71 ± 0.06	0.75 ± 0.05	0.78 ± 0.07	$\mathbf{0.79} \pm 0.06$	0.81 ± 0.15	0.62 ± 0.12	0.87 ± 0.09	0.66 ± 0.11
	10	0.81 ± 0.03	0.83 ± 0.02	0.81 ± 0.03	0.82 ± 0.03	0.87 ± 0.04	$\mathbf{0.86} \pm 0.03$	0.86 ± 0.07	0.52 ± 0.11	0.89 ± 0.07	0.53 ± 0.10

Dataset	Noise Level (%)	Ens6 (B, RF, SVM, NN, SF)		Ens7 (B, RF, SVM, PSF)		Ens8 (RF, SVM, NN, PSF)		Ens9 (B, RF, SVM, NN, PSF)		Ens10 (B,RF,SVM,NN,SF,PSF)	
		Pr	$F_{0.5}$	Pr	$F_{0.5}$	Pr	$F_{0.5}$	Pr	$F_{0.5}$	Pr	$F_{0.5}$
TTT	2	0.93 ± 0.09	0.74 ± 0.11	0.86 ± 0.09	0.75 ± 0.07	0.91 ± 0.07	$\mathbf{0.84} \pm 0.07$	0.89 ± 0.07	0.76 ± 0.08	$\mathbf{0.95} \pm 0.09$	0.73 ± 0.10
	5	0.98 ± 0.03	0.82 ± 0.05	0.98 ± 0.02	0.87 ± 0.04	0.98 ± 0.02	$\mathbf{0.91} \pm 0.03$	0.99 ± 0.01	0.87 ± 0.04	$\mathbf{1.00} \pm 0.01$	0.80 ± 0.06
	10	0.97 ± 0.03	0.70 ± 0.05	0.97 ± 0.03	0.81 ± 0.03	0.98 ± 0.02	0.85 ± 0.03	0.98 ± 0.02	0.78 ± 0.05	$\mathbf{0.99} \pm 0.02$	0.69 ± 0.05
KRKP	2	0.93 ± 0.02	0.86 ± 0.02	0.90 ± 0.02	0.87 ± 0.02	0.92 ± 0.01	$\mathbf{0.91} \pm 0.01$	0.93 ± 0.01	0.90 ± 0.02	$\mathbf{0.94} \pm 0.02$	0.86 ± 0.02
	5	$\mathbf{0.97} \pm 0.02$	0.84 ± 0.03	0.95 ± 0.01	0.90 ± 0.01	0.95 ± 0.02	$\mathbf{0.92} \pm 0.01$	0.96 ± 0.01	0.91 ± 0.01	$\mathbf{0.97} \pm 0.02$	0.84 ± 0.03
	10	$\mathbf{0.97} \pm 0.02$	0.70 ± 0.04	0.96 ± 0.01	0.88 ± 0.01	$\mathbf{0.97} \pm 0.01$	0.90 ± 0.01	$\mathbf{0.97} \pm 0.01$	0.88 ± 0.01	$\mathbf{0.97} \pm 0.01$	0.68 ± 0.04
CHD	2	0.83 ± 0.15	0.73 ± 0.09	0.72 ± 0.17	0.71 ± 0.15	0.81 ± 0.15	$\mathbf{0.76} \pm 0.12$	0.81 ± 0.14	$\mathbf{0.76} \pm 0.10$	$\mathbf{0.86} \pm 0.14$	0.75 ± 0.09
	5	0.89 ± 0.12	0.68 ± 0.11	0.88 ± 0.10	0.73 ± 0.12	0.84 ± 0.10	0.71 ± 0.10	0.88 ± 0.10	0.71 ± 0.13	$\mathbf{0.89} \pm 0.12$	0.60 ± 0.14
	10	0.95 ± 0.07	0.65 ± 0.10	0.91 ± 0.06	0.80 ± 0.09	0.91 ± 0.06	0.80 ± 0.08	0.93 ± 0.07	0.78 ± 0.12	$\mathbf{0.96} \pm 0.07$	0.65 ± 0.10
NAKE	2	0.94 ± 0.12	0.75 ± 0.14	0.54 ± 0.06	0.54 ± 0.07	0.84 ± 0.11	0.77 ± 0.11	0.87 ± 0.11	$\mathbf{0.79} \pm 0.10$	$\mathbf{0.98} \pm 0.08$	0.74 ± 0.12
	5	0.89 ± 0.11	0.66 ± 0.11	0.77 ± 0.10	0.70 ± 0.11	0.88 ± 0.06	0.77 ± 0.07	0.88 ± 0.08	0.77 ± 0.09	$\mathbf{0.91} \pm 0.09$	0.66 ± 0.12
	10	0.92 ± 0.07	0.51 ± 0.11	0.87 ± 0.04	0.75 ± 0.05	0.90 ± 0.05	0.76 ± 0.04	0.91 ± 0.04	0.76 ± 0.05	$\mathbf{0.93} \pm 0.06$	0.48 ± 0.11

Ensemble filtering using a majority voting scheme (a simple majority of votes of elementary noise filters in the given ensemble) did not outperform the precision of injected noise detection achieved by the best elementary filtering algorithms alone. Therefore, these results were omitted in this experimental results section. On the other hand, consensus voting significantly improved the precision of noise detection as well as the $F_{0.5}$-scores. Therefore, from now on, with ensemble noise filters it is actually referred to ensemble noise filters using the consensus voting scheme.

Performance results of the ten ensemble noise filters constructed from different elementary noise detection algorithms can be found in Table 4.4. The results show that in terms of precision using many diverse elementary noise detection algorithms in the ensemble proves to be better. Even including a different type of worst performing noise filter like Bayes, this does not reduce the ensemble's precision but it actually improves it, as can be seen by comparing precision results of Ens2 and Ens3, Ens5 and Ens6, as well as Ens8 and Ens9. The highest precision among noise detection ensembles was hence not surprisingly achieved by Ens10 which uses all elementary noise filters examined in this work. However, considering the $F_{0.5}$-scores, it is interesting to notice that ensembles without SF perform best, namely Ens1, Ens2 and Ens3 perform better on all datasets at higher noise levels (5-10%), whereas Ens7, Ens8 and Ens9 are better at lower noise levels (2-5%). This is due to the low noise recall of SF, which reduces the recall of the ensembles and hence leads to lower $F_{0.5}$-scores.

Last, the results presented in part (C) of Table 4.3 are discussed, for the HARF noise detection algorithm (described in more detail in Section 3.3) with four different agreement levels: 60%, 70%, 80% and 90%, also referred to as HARF-60, HARF-70, HARF-80 and HARF-90, respectively. The results in part (C) of Table 4.3 as well as Figures 3.4 and 3.6 in Section 3.3 show that HARF-80 and HARF-90 are the most precise noise detection algorithms presented in Table 4.3 and achieve comparable or higher precision than the most precise ensemble filters in Table 4.4. Furthermore, in terms of the $F_{0.5}$-scores, the best results are achieved by HARF-70 and HARF-80, which shows to be also significantly better than the results achieved by elementary noise detection algorithms, except in one case where SVM outperforms the HARF algorithms on the TTT dataset with 10% noise level. Compared to ensemble noise filters, the HARF-70 and HARF-80 achieve lower $F_{0.5}$-scores only on the TTT dataset, but are slightly better on the KRKP and CHD datasets and better on the NAKE dataset.

At this point it is possible to observe another interesting phenomenon, indicating that classification noise filters are robust noise detection algorithms. The analysis of the results in Tables 4.3 and 4.4 shows that the proportion of detected noise (i.e., the precision of noise detection) increases with the increased abundance of noise (increased levels of artificially inserted noise 2%, 5% 10%). Clearly, with increased noise levels, the classifiers are less accurate (their accuracy drops), therefore leading to the increased number of misclassified instances, i.e., the number of instances which are considered potentially noisy increases. As an observed side effect, the proportion of detected artificially corrupted instances increases as well.

Basic graphical results presentation

In addition to the tabular representation, the usual approach to compare the performance results of different noise detection algorithms is the results visualization by column (bar) charts. We compare the precision of noise detection results, noise recall results and the $F_{0.5}$-scores of different noise detection algorithms on four domains with 5% artificially injected noise in Figures 4.6, 4.7 and 4.8, respectively. The results obtained with 2% and 10% injected noise level are very similar and therefore their presentation is skipped.

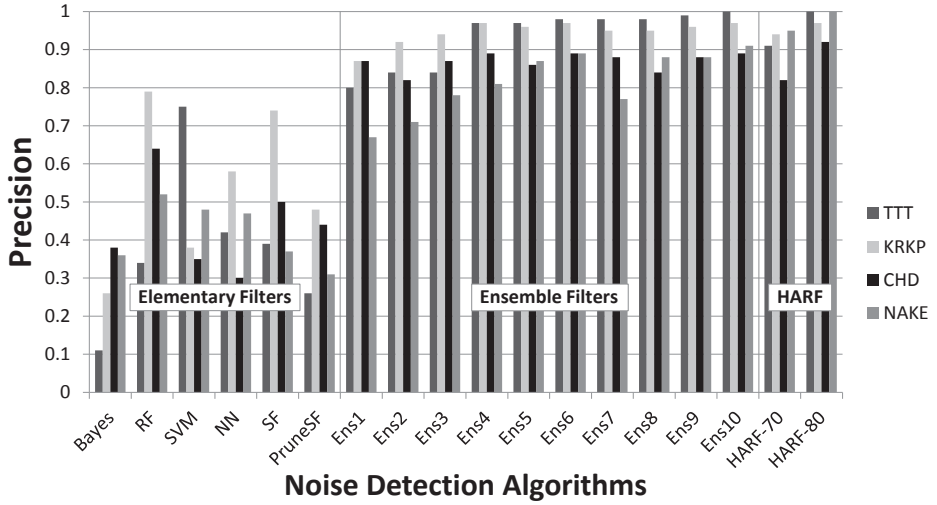


Figure 4.6: Comparison of precision of noise detection results of different noise detection algorithms on four domains with 5% injected noise.

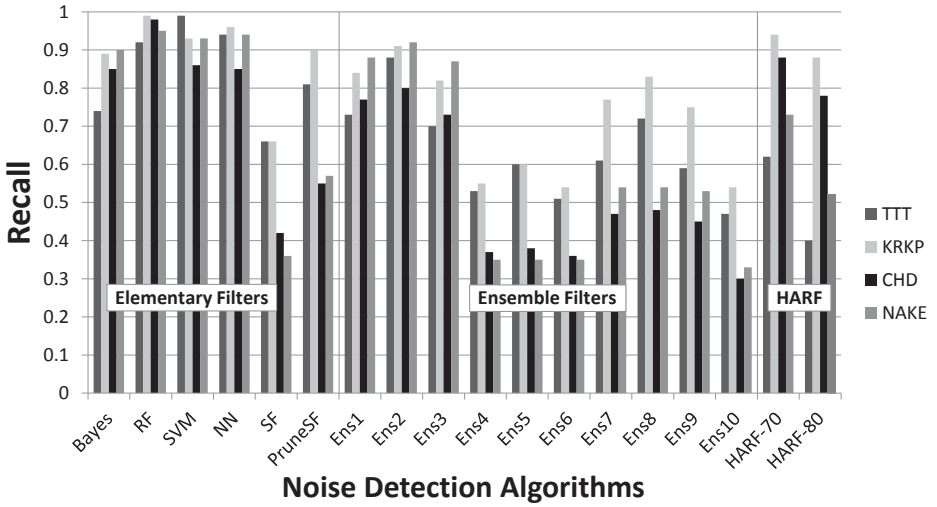


Figure 4.7: Comparison of noise recall results achieved by different noise detection algorithms on four domains with 5% injected noise.

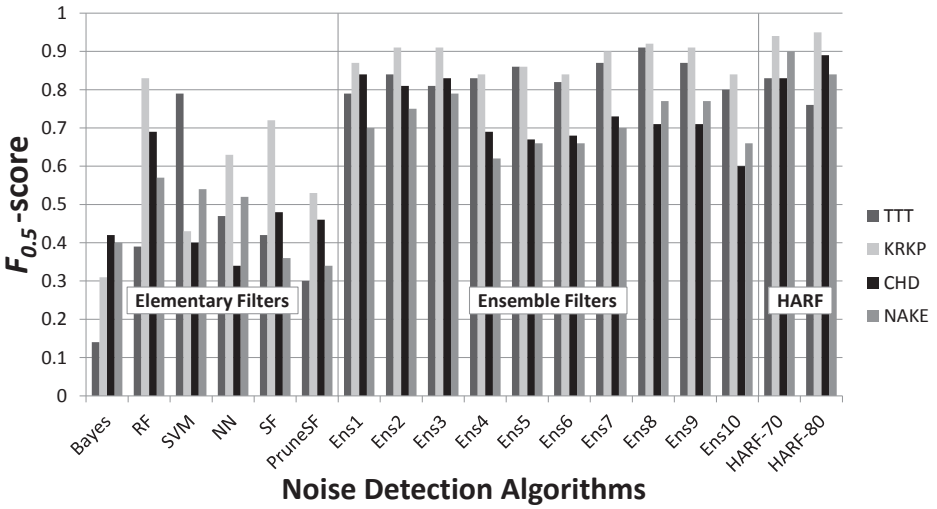


Figure 4.8: Comparison of $F_{0.5}$-scores achieved by different noise detection algorithms on four domains with 5% injected noise.

Interpreting performance results presented with column charts is simpler and yields the following observations. Figure 4.6 shows that ensemble filters and the HARF noise detection algorithm outperform the elementary filters in terms of precision, where Ens10 and HARF-80 are the most precise algorithms on the TTT and KRKP domains and HARF-80 is the most precise noise detection algorithm on the CHD and NAKE domains. The comparison of noise recall illustrated in Figure 4.7 shows that classification filters undoubtedly achieve the highest recall, followed by the ensembles Ens1, Ens2 and Ens3, and HARF-70. In terms of $F_{0.5}$-scores shown in Figure 4.8, the ensembles and HARF algorithms are also significantly better compared to elementary noise detection algorithms. The best $F_{0.5}$-score results are achieved by HARF-80 on all the domains, except on the TTT domain where Ens7 performs best. However, all these results are shown in separate figures for precision, recall and the $F_{0.5}$-score, which makes them hard to interpret as a whole.

Statistical evaluation of performance results

The Friedman test was used for statistical comparison of the performances of all algorithms in the presented experimental setting. This section presents the *critical difference diagrams* (CD diagrams), described in Section 4.3.3, which visualize the average ranks of algorithm performance achieved on the experimental datasets, and the critical difference obtained by the post-hoc tests indicating which algorithms perform significantly different.

The statistical comparison was performed for all elementary noise detection algorithms, their selected ensembles, and two HARF algorithms. The observed performance measure was the $F_{0.5}$-score for which the average ranks over all the experimental datasets were computed. Figure 4.9 presents the CD diagram for the Nemenyi post-hoc test on all noise detection algorithms evaluated in the described experimental setting, for $\alpha = 0.05$.

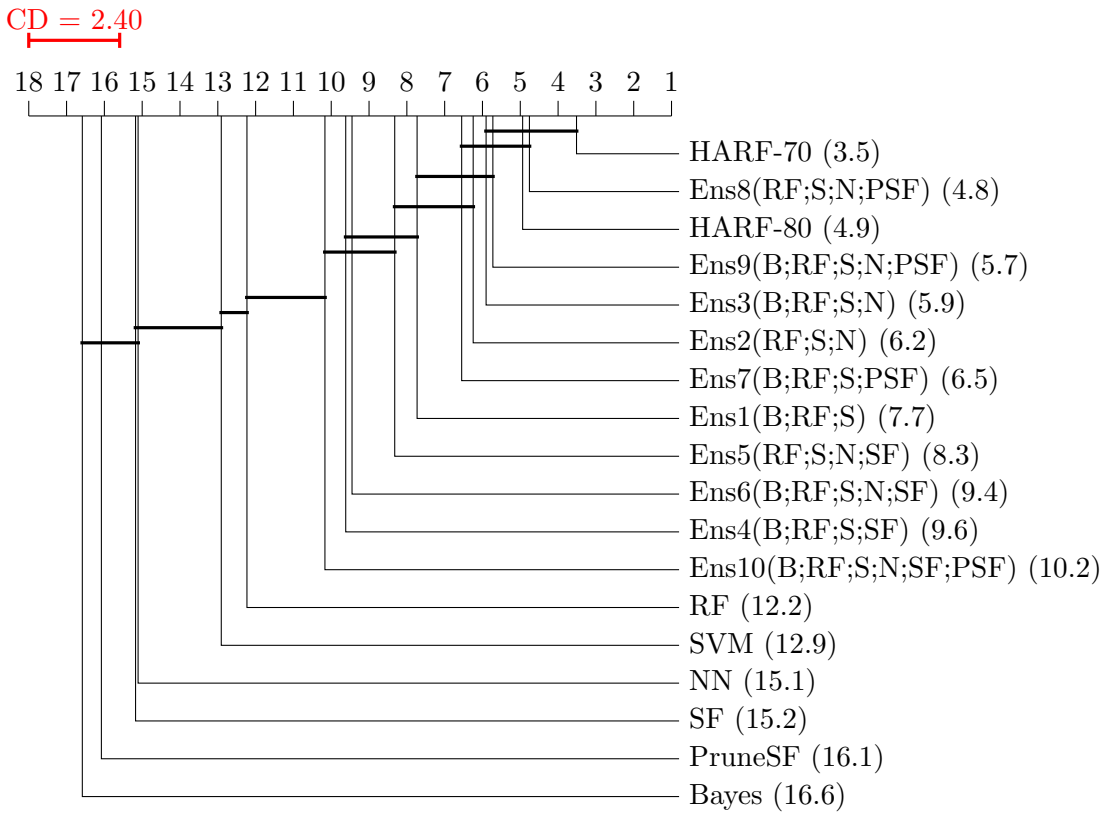


Figure 4.9: Critical difference diagram for the Nemenyi test on all algorithms evaluated in the described experimental setting (120 experiments), for the $F_{0.5}$-scores and $\alpha = 0.05$.

The CD diagram shows that when averaging the performance over all the experiments the highest average rank of 3.5 was achieved by the HARF-70 noise detection algorithm. Although this rank seems to lead with a meaningful difference to the second highest rank 4.8, the Nemenyi statistical post-hoc test reveals that the critical difference of 2.40 among ranks is required to conclude that the difference in performance is statistically significant at $\alpha = 0.05$. Hence, according to this statistical comparison, HARF-70 does not perform significantly better than Ens8, HARF-80, Ens9 and Ens3, but does perform better than all the other algorithms. The CD diagram can be used to identify statistically significant performance differences among any algorithms whose ranks differ more than the critical difference.

The results shown in the CD diagram in Figure 4.9 were obtained from average performance ranks computed over all 120 experiments performed. However, this setting does not entirely comply with the assumptions of the statistical tests which require independence among experiments. It is arguable if two noise detection experiments on the same dataset but each with (the same or different amount of) randomly injected class noise are independent. The above evaluation was performed with the awareness that statistical evaluation on only four datasets does not suffice for any statistically significant results. Nevertheless, for comparison the CD diagram of the Nemenyi test of average performance values on four datasets is presented in Figure 4.10. The diagram shows that this small number of datasets does not suffice to conclude about almost any statistically significant difference among the algorithms, except between the highest ranked (HARF-70) and the lowest four, and the second highest ranked (Ens8) and the algorithm with the lowest rank (Bayes).

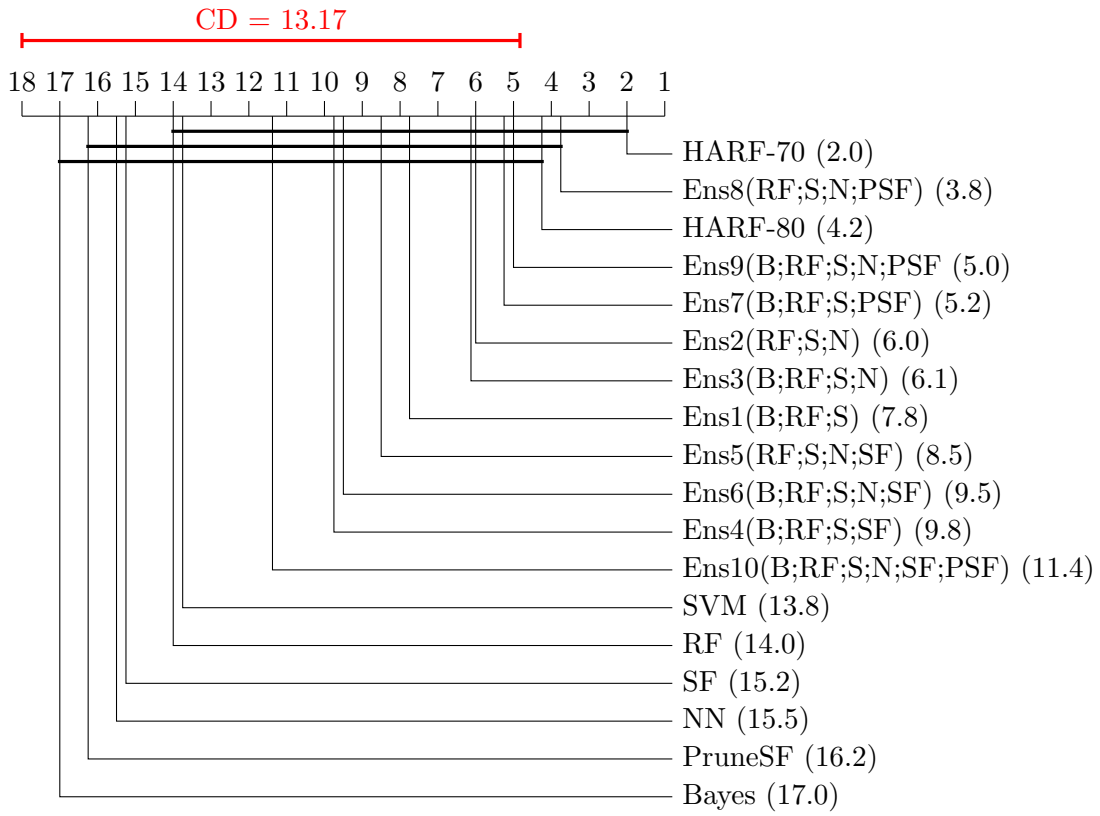


Figure 4.10: Critical difference diagram of all noise detection algorithms evaluated in the described experimental setting (4 exp. datasets), for the $F_{0.5}$-scores and $\alpha = 0.05$.

On the other hand, when using the Bonferroni-Dunn statistical post-hoc test for comparing other algorithms to the highest ranked HARF-70, the critical difference is 2.05. Figure 4.11 presents the critical difference diagram with CD of the Bonferroni-Dunn statistical post-hoc test at $\alpha = 0.05$ for the control algorithm HARF-70. The diagram shows that according to this statistical test, the performance of HARF-70 is not significantly different from Ens8 and HARF-80, but that HARF-70 performs statistically significantly better than all the other 15 evaluated noise detection algorithms.

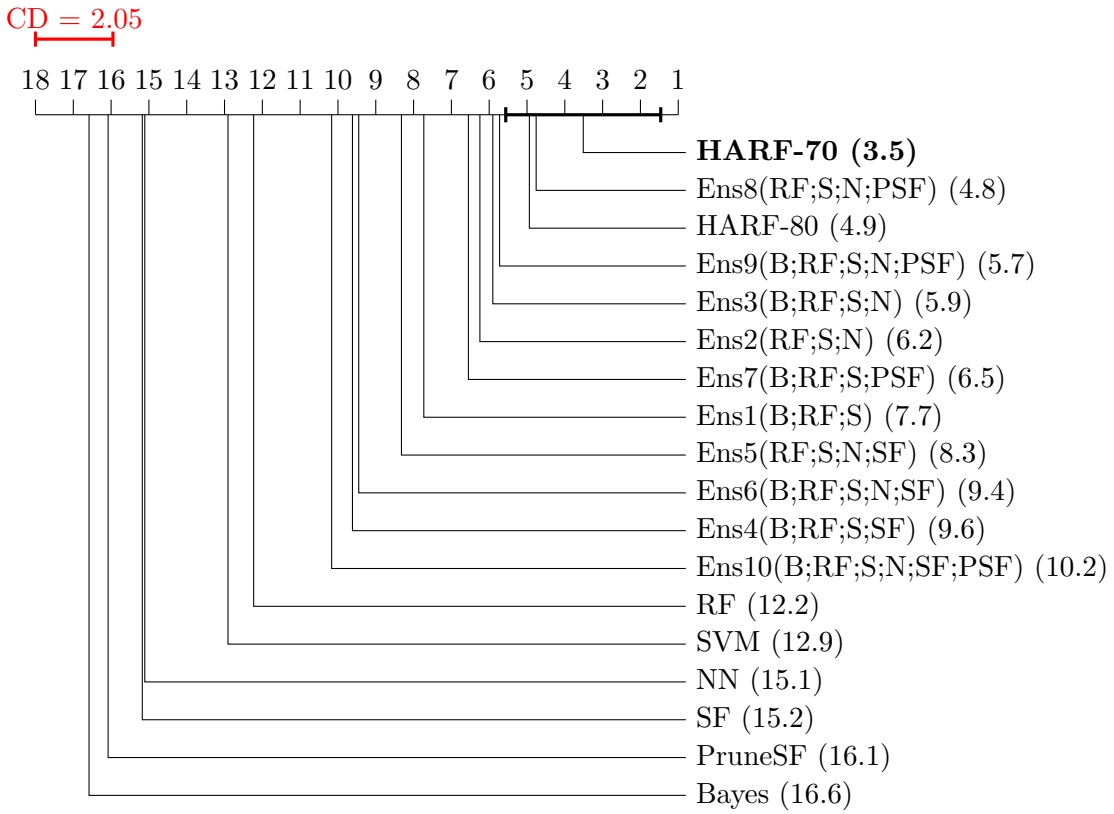


Figure 4.11: Critical difference diagram for the Bonferroni-Dunn test of the HARF-70 vs. all other noise detection algorithms evaluated in the described experimental setting, for the $F_{0.5}$-scores and $\alpha = 0.05$.

4.5.2 Experimental results using the Viper evaluation approach

In the standard evaluation and comparison of noise detection performance results, as presented in Section 4.5.1, inspection of tabular results is needed, which can require quite some effort to compare the results, especially in the case of several performance measures and algorithms used. Furthermore, the graphical presentation with column (bar) charts requires a separate figure for each performance measure, although theoretically more than one measure could be depicted in a column chart, but it would reduce its understandability. On the other hand, the Viper methodology for visual evaluation and comparison of noise detection results enables the visualization of several performance measures and results for several noise detection algorithms in a single figure.

Viper presents the performance results of noise detection algorithms on a given domain as points in the precision-recall space. In the experimental evaluation of noise detection algorithms, the $F_{0.5}$-score was used, which weights precision more than recall. This score is

used in the *F-isoline evaluation method*, where equal $F_{0.5}$-scores in the form of isolines enable to compare noise detection performance in the precision-recall space. When selecting the algorithm with maximal recall in the *ε-proximity* of maximal achieved precision, as defined in Equation 4.3, the proximity value $\varepsilon = 0.05$ was chosen. Using these settings the performance of elementary and ensemble noise detection algorithms was compared with the HARF noise detection algorithm. The performance results of noise detection on four domains with 5% injected class noise and visualized with the proposed VIPER methodology can be found in Figures 4.12, 4.13, 4.14 and 4.15. Similar figures for 2% and 10% noise levels can be found in Appendix A.

The comparison and interpretation of results by the VIPER performance visualization is very intuitive and almost straightforward. The advantage of noise detection algorithm performance evaluation by VIPER visualization in the precision-recall space provides the expert with the means to compare and see the interaction of three performance measures: precision, recall and the F-measure. Furthermore, also the ε-proximity evaluation method for selecting the best noise detection algorithm becomes easily understandable through the visualization of the ε neighborhood of maximal achieved precision.

By inspecting the figures, the following facts can be observed. Elementary classification filters are fairly good at finding most of the injected noise based on their high recall results, however they identified as noisy also a lot of regular (non-noisy) instances and therefore their precision is not as satisfactory, nor are their $F_{0.5}$-scores. In comparison, the saturation filters have lower recall, while the precision and the $F_{0.5}$-score results are between the best and the worst classification filter performance.

Ensemble filters formed from groups of elementary filters, on the other hand, prove to be better suited for precise noise detection. Especially the Ens10 filter, which has the largest variety of different filtering algorithms, achieves the highest precision on three out of four domains among the ensemble filters, but of course at the expense of lower recall and hence also of lower $F_{0.5}$-score. Generally, ensemble filters show great increase in precision of noise detection, however their $F_{0.5}$-scores are (because of their lower recall) in the majority of cases comparable to the performance of best classification filters.

Furthermore, since classification filters achieve good noise recall, also Ens1, Ens2 and Ens3, which are constructed from different groups of classification filters, achieve best recall results among ensembles filters. It is interesting to notice that including a poorly performing noise detection algorithms in the ensemble[1] does reduce the ensemble's recall, but it does not negatively affect its precision, on the contrary it may even increase it, which may lead also to higher F-scores. This can be observed in all three ensemble groups: Ens1-3, Ens4-6 (with SF) and Ens7-9 (with PruneSF), which are all constructed from the same three basic groups of classification filters, i.e. Ens1, Ens2 and Ens3. Another interesting observation, very easy to see with the VIPER visualization, is that the three ensemble groups form cluster-like formations in the precision-recall space. By adding a saturation filter to the Ens1-3 group and obtaining the Ens4-6 or Ens7-9 group, the performance change seems to be almost a translation of the group's cluster formation in the precision-recall space.

The addition of a new noise detection algorithm, like the saturation filter, to an ensemble with a consensus voting scheme translates the ensemble's performance in the direction of the lowest recall of a member in the ensemble and elevates its precision. These observations may suggest that the best approach to constructing an ensemble with a good precision-recall performance could be by using a diverse group of noise detection algorithms, where each member is achieving the highest possible recall.

[1] Bayes added to Ens2 on domains TTT, KRKP and NAKE to get Ens3, or NN added to Ens1 on the CHD domain to get Ens3.

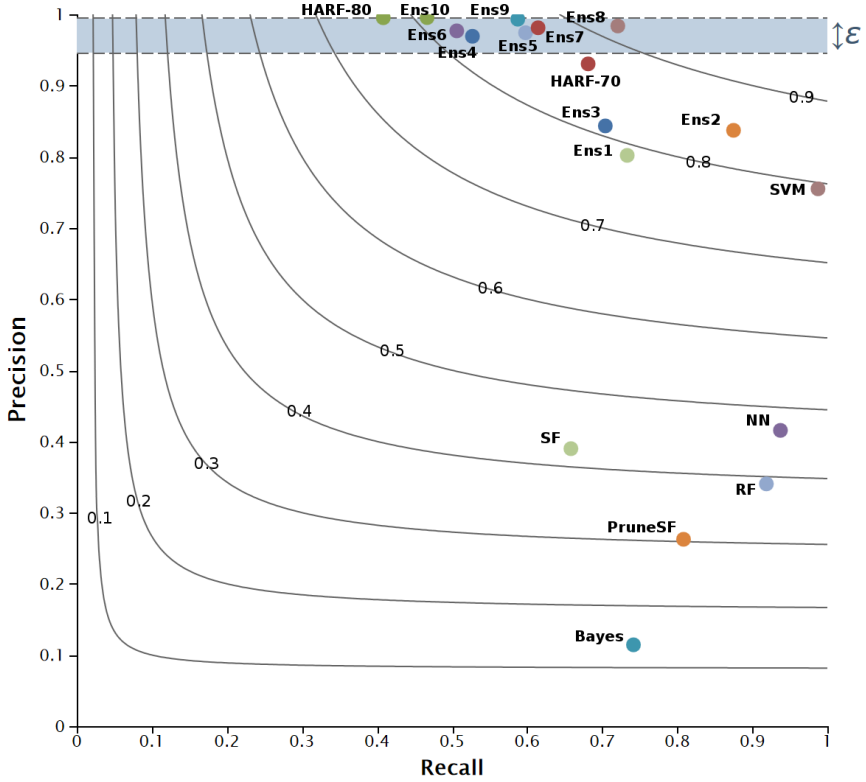


Figure 4.12: Performance results of different noise detection algorithms in the precision-recall space on the Tic Tac Toe dataset with 5% injected noise.

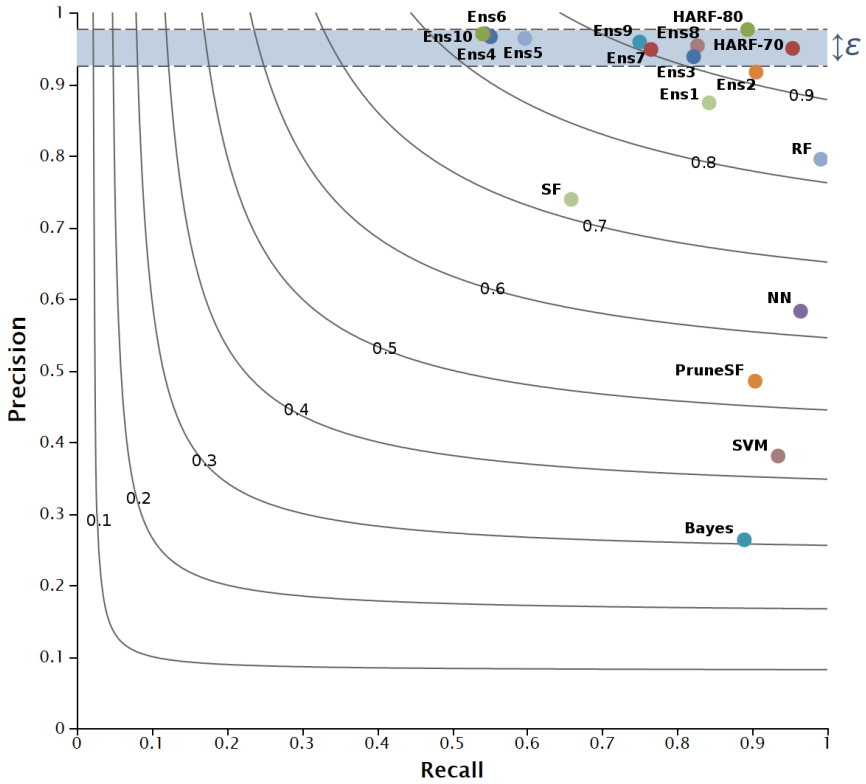


Figure 4.13: Performance results of different noise detection algorithms in the precision-recall space on the King-Rook vs. King-Pawn dataset with 5% injected noise.

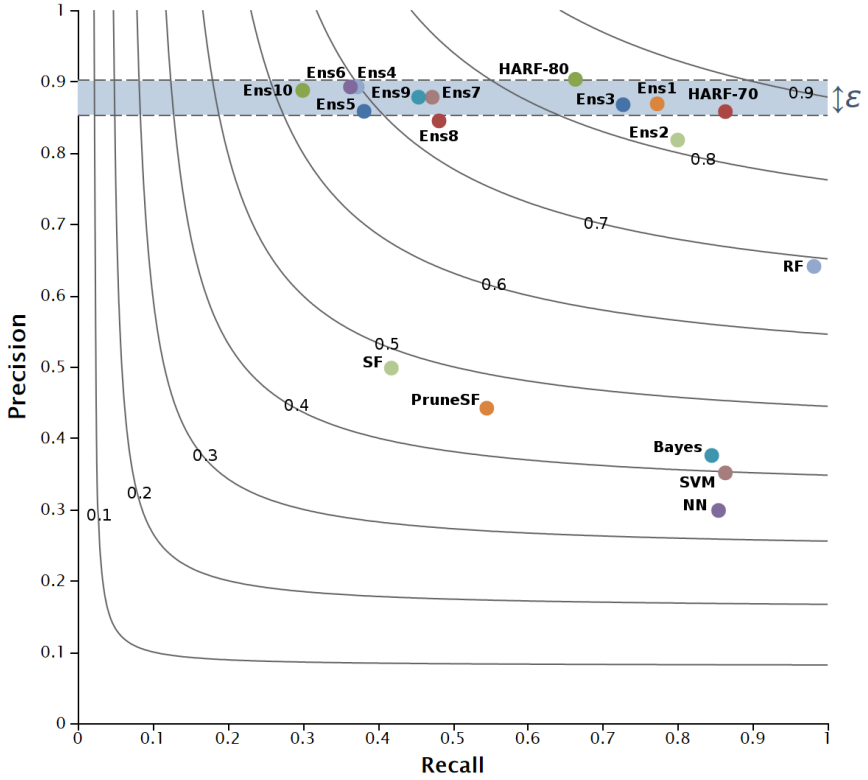


Figure 4.14: Performance results of different noise detection algorithms in the precision-recall space on the Coronary Heart Disease dataset with 5% injected noise.

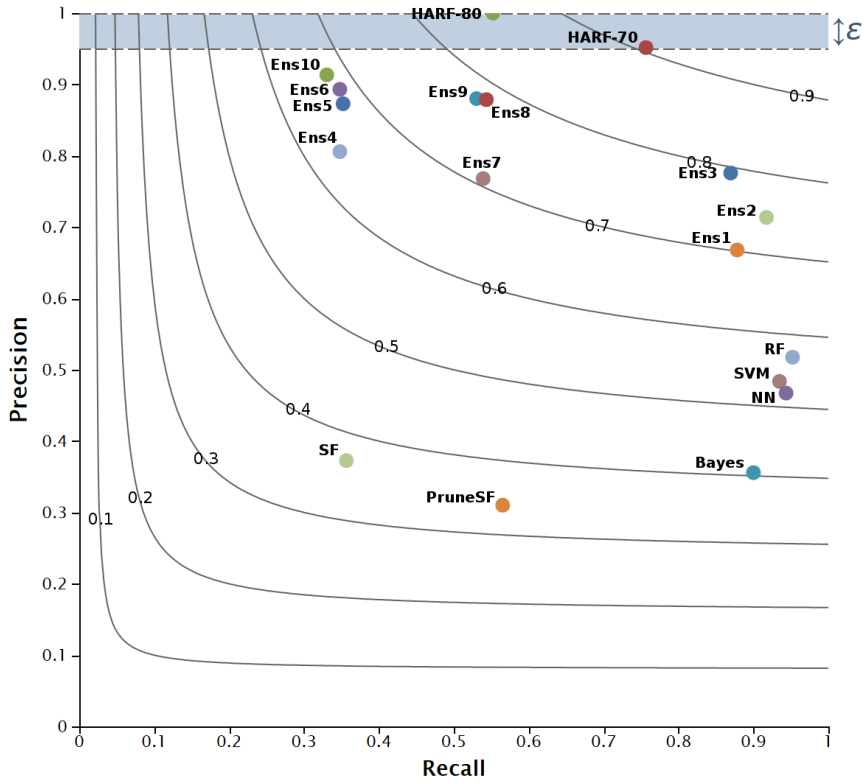


Figure 4.15: Performance results of different noise detection algorithms in the precision-recall space on the News Articles on Kenyan Elections dataset with 5% injected noise.

To depict the VIPER evaluation of a 'new' noise detection algorithm and its comparison to other (existing) approaches the High Agreement Random Forest noise detection algorithms (HARF) with 70% and 80% agreement level (HARF-70 and HARF-80, respectively) were included into the evaluation. Without much effort the performance of the two HARF algorithms is immediately apparent from the visualization. They achieve comparable or higher precision and recall of noise detection compared to ensemble filters and clearly outperform all elementary noise detection algorithms. While HARF-80 is the most precise noise detection algorithm in all four domains, in terms of the $F_{0.5}$-score results HARF-70 performs best on all domains except the TTT, where the ensembles Ens7-9 perform better.

Finally, the selection of the best noise detection algorithm according to the ε-proximity evaluation method, defined by Equation 4.3, depends on the tolerated ε drop of precision. In the case of $\varepsilon = 0.05$ on the TTT domain ensemble filter Ens8 performs best, but on the other three domains the HARF-70 noise detection algorithms perform best. An increase or decrease of the ε value for few percents does not dramatically change the selection of the best noise detection algorithm.

4.5.3 Quantitative evaluation of noise ranking results

Ranked noise detection outputs as produced by NOISERANK can be quantitatively evaluated as described in Section 4.3.2, if the true noisy instances are known in the evaluation process. The experimental settings used in this chapter, described in Section 4.2, allow visual performance evaluation of the ranked noise detection outputs that were obtained when NOISERANK was applied to datasets with artificially injected noisy instances.

The ensemble of the NOISERANK approach was composed of all six elementary noise detection algorithms presented in Section 3.2.2. Consequently, each instance of a dataset could be detected as noisy by a maximum of six and a minimum of zero noise detection algorithm, implying seven possible ranks that can be assigned to an instance. These seven ranks can be viewed as decision threshold for noise identification.

For each dataset and for each noise level the experiments were performed ten times with randomly injected noise. For each of these experiments a ROC curve and a PR curve was computed. The results of ten repetitions on a given dataset with a specific noise level were aggregated by the so-called threshold averaging, as described in Fawcett (2006). This means, that for each threshold (rank) the average performance point over the ten experiments was computed.

This section provides two performance visualizations of NOISERANK's noise detection capabilities evaluated on datasets with known noisy instances. First, the ROC analysis is presented, which originates in signal theory and became widely adopted in the evaluation of classification algorithms. NOISERANK can by considered as a scoring/ranking classifier for noise, its average ROC curves on four datasets with 5% noise are presented in Figure 4.16.

In term of ROC analysis NOISERANK performs quite well on all datasets, since noise recall (true positive rate) increases much faster than the amount of regular instances falsely declared as noise (false positive rate), according to the changing noise identification threshold. However, ROC analysis does not directly address the precision of noise detection. Therefore, the second performance visualization presented in this section focuses exactly on the performance measures of interest when evaluating explicit noise detection. The visualization of threshold averaged PR curves enhanced by the F-isolines evaluation method is presented in Figure 4.17.

Each point on the averaged PR curves presented with a marker/symbol, shows the average performance of NOISERANK at one of the seven possible noise identification thresholds. The most left point of each curve indicates the performance when only first ranked instances

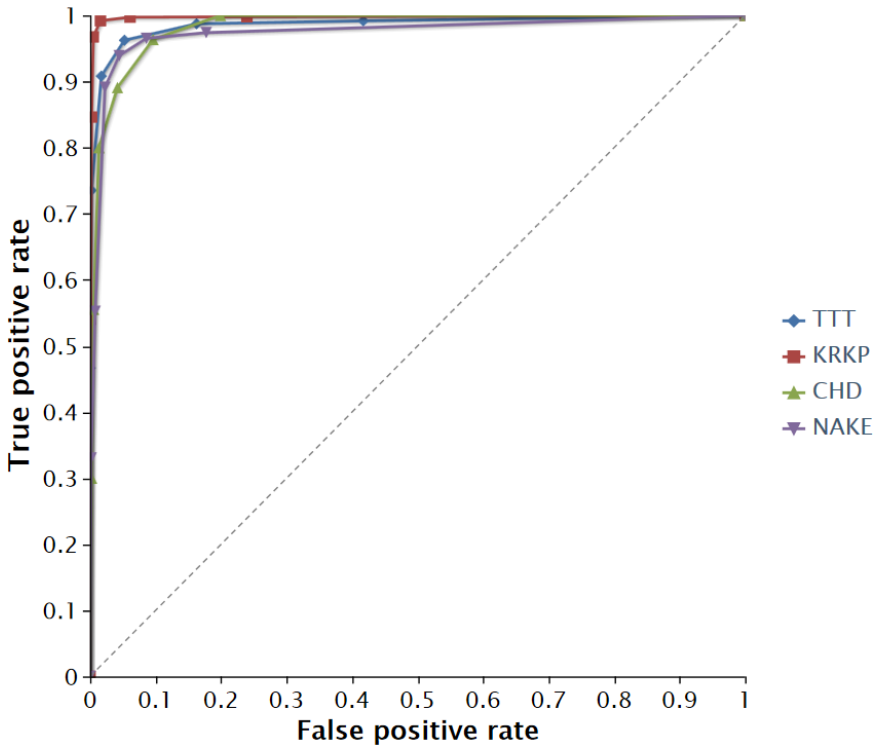


Figure 4.16: Thresholded average ROC curves of NoiseRank on four datasets with 5% injected noise.

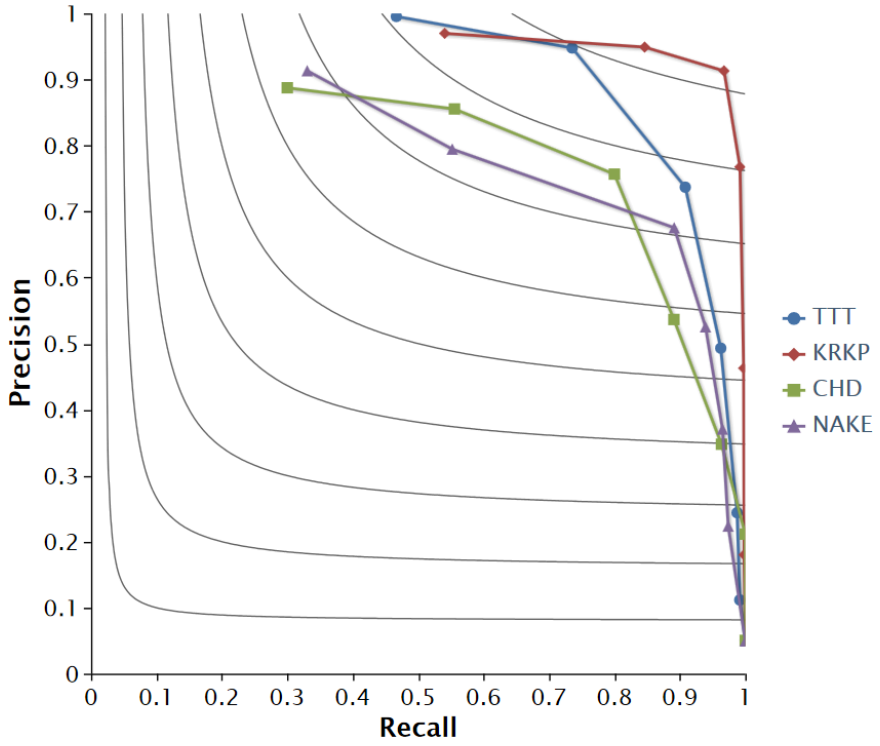


Figure 4.17: Thresholded average PR curves of NoiseRank on four datasets with 5% injected noise.

are considered noisy. Points further to the right indicate the performance at a lower noise identification threshold, i.e., instances ranked lower (second, third, etc.) are included in the performance calculation.

The results in Figure 4.17 show that identifying artificial noise in originally noiseless datasets (TTT and KRKP) seems to be easier than in the CHD and NAKE datasets, which already contain noise and/or outliers from the real world. Furthermore, the highest precision of noise detection can be expectedly observed for the highest noise identification threshold, which actually equals the results of the ensemble filter Ens10 used in a consensus voting scheme, as presented in Section 4.5.1. However, the best results in terms of the precision-recall trade-off are achieved at lower noise identification thresholds, as can be clearly seen with the help of the $F_{0.5}$-isolines.

5 Applications of Ensemble-based Noise Detection

This chapter presents three real-life use cases in which the ensemble-based noise detection approaches presented in Chapter 3 were applied. The obtained results were qualitatively evaluated by the corresponding domain experts. In the first use case, the NoiseRank methodology and the HARF noise detection algorithm were applied to a medical domain, where false diagnosis presents a serious problem and complex medical cases require special attention. Both noise detection approaches were also applied to a news articles domain on Kenyan elections, where the expert in linguistic pragmatics was interested in non-typical articles written either by local or by the Western newspapers. In the last use case, ensemble-based noise detection was used to support cross-domain link discovery by identifying domain outlier documents in a domain-pair dataset which presented a rich source of domain bridging terms.

5.1 Detecting false medical diagnostic cases

In medical diagnosis two major problems are false diagnosis and complex medical cases. In data mining terms these problems correspond to classification noise and outliers, respectively. Therefore a real-life medical domain presents a suitable application area for verifying the practical relevance of ensemble-based noise detection approaches. This section provides a domain expert evaluation of a small set of top-ranked instances detected as potentially noisy by the NoiseRank and HARF noise detection approaches. The coronary heart disease domain is presented, followed by expert evaluation of the detected noisy instances and a comparison among the two noise detection approaches. The section concludes with a summary of the qualitative evaluation of NoiseRank and HARF on a medical domain.

5.1.1 Coronary heart disease domain

Coronary heart disease (CHD) is a disease of diminished blood supply caused by artherosclerotic plaque in coronary arteries. Diminished oxygen supply of the dependent region of the muscular tissue of the heart results in chest pain (angina pectoris). The extreme consequences are myocardial infarction and death. Coronary heart disease is one of the world's most frequent causes of mortality and an important medical problem (Maron et al., 1998).

In the available dataset[1], the instances are records of 238 patients who entered a specialized cardiovascular institution[2] in a few months period. The descriptor set includes 10 anamnestic attributes, 7 attributes describing laboratory test results, ECG at rest (5 attributes), the exercise test data (5 attributes), echocardiography results (2 attributes), vectorcardiography results (2 attributes), and long-term continuous ECG recording data (3 attributes). The patients were classified into two classes: CHD patients diagnosed as those with coronary heart disease and non-CHD patients. The included negative cases (patients

[1] The same dataset was previously used by Gamberger and Lavrač (2002) and Gamberger et al. (2003).
[2] Institute for Cardiovascular Prevention and Rehabilitation, Zagreb, Croatia.

who do not have CHD) are not randomly selected people but individuals with some subjective problems or those considered by the general practitioners as potential CHD patients. As the data was collected at a specialized medical clinic, the database is not an epidemiological CHD database reflecting actual CHD occurrence in a general population, since nearly one half of gathered patient records represent actual CHD patients. More specifically, the dataset consists of 238 patient records: 111 CHD patients (positive cases), and 127 people without CHD (negative cases). Among them there are 177 males (80 positive and 97 negative) and 61 females (31 positive and 30 negative).

5.1.2 Expert evaluation of results obtained by NoiseRank

The medical expert[1] (cardiologist) was shown in total 8 top-ranked examples detected by NoiseRank, as presented in Figure 3.3, for the analysis. In the qualitative evaluation he used all the available retrospective data for these patients to repeat the previous medical reasoning process with the intention to identify why they were detected by NoiseRank as special cases. Since exercise and Holter ST depression values are medically most relevant CHD markers for expert evaluation (Deanfield et al., 1984)[2], Table 5.1 shows the measured data for exercise and Holter ST depression values for the 8 top-ranked patients that were analyzed by the cardiologist. We refer to this table in the analysis of some of the top-ranked instances below.

As a result of the analysis, the expert identified the following reasons why the top-ranked examples are indeed special cases (noise or outliers):

Classification mistake (patient ID 51):
After rechecking the patient data, the cardiologist realized that a mistake in the classification has occurred. In the concrete situation, the mistake was the result of a formal patient classification that did not take into account the actual conditions in which the measurements were performed. The situation is illustrated in the first row of Table 5.1. Patient 51 is classified as non-CHD because exercise ECG ST depression value $0.8mm$ is below the cut-off value of $1mm$ for myocardial ischemia and despite the fact that Holter ECG ST depression value of $1mm$ is actually a borderline value. The mistake was done because it was not realized that exercise ECG ST value 0.8 was measured at only very moderate level of actual exercise stress. It can be assumed that with a normal level of exercise or effort, the expected depression value would have been significantly over $1mm$. Given that this is an actual classification mistake, this type of noise was easily detected by all the noise detection algorithms.

Sportsman (patient ID 0):
The patient is correctly classified as CHD but the critical value $1.1mm$ for exercise ST depression (second row of Table 5.1) was detected under the conditions of relatively high exercise stress. This is due to the excellent physical condition of the patient, having as a consequence that other patient data, including Holter ST depression, have almost normal values. The patient represents an actual outlier and its detection as a noisy example is fully justified.

False positive medical test (patient IDs 27, 194):
It is known that Holter ECG ST depression values (presented in the last column of Table 5.1) are sometimes prone to false positive results. It means that values $\geq 1mm$ do not necessarily mean a positive diagnosis (CHD). The situation is relatively more

[1] Goran Krstačić, MD, PhD, FESC Cardiologist, Institute for Cardiovascular Diseases and Rehabilitation, Zagreb, Croatia. [2] This fact has been also previously confirmed by machine learning approaches on the same CHD domain (Gamberger and Lavrač, 2002).

Table 5.1: Comparison of the exercise (*exST*) and Holter (*holterST*) ST depression values (in millimeters) for the top 8 noisy instances and for the average *CHD* and *non-CHD* instances (patients) in the CHD domain.

Rank	Diagnosis	Patient ID	exST	holterST
1	non-CHD	51	0.80	1.00
2	CHD	229	0.30	0.30
3	CHD	0	1.10	0.50
3	non-CHD	27	0.50	1.00
3	non-CHD	39	0.40	0.50
3	CHD	176	0.80	1.00
3	CHD	194	0.10	1.00
3	CHD	213	0.20	0.20
	Average non-CHD		0.15	0.21
	Average CHD		1.11	1.40

common for female patients. In the cases when the Holter ST depression value is $\geq 1mm$ and at the same time the more reliable test exercise ST depression is $< 1mm$, the medical expert may decide to classify the patient as negative (in the concrete data set for patient ID 27 presented in the fourth row of Table 5.1) or he may decide to classify the patient as positive (patient ID 194 presented in row 7 of Table 5.1). It is very interesting to notice that both situations were successfully recognized by most of the noise detection algorithms. This was possible because Holter ST depression is one of the two main decision attributes used for patient classification.

Other medical problems (patient ID 39):

The patient is correctly classified as non-CHD which is strongly suggested by her exercise and Holter ST depression values (row 5 of Table 5.1). However, this instance was detected as noisy because of significantly non-normal values of other diagnostic parameters like high cholesterol value, low high-density liphoprotein value, and the detected ST depression in the baseline ECG test at rest. Technically, this patient is an example of distributed attribute noise and medically it represents a patient with a higher level of cardiovascular risk factors.

Borderline case (patient ID 176):

The patient has the main decision parameter exercise ECG ST depression value ($0.8mm$) just below the cut-off point of $1.0mm$. Formally, patient 176 (row 6 in Table 5.1) is correctly classified as CHD because of high Holter ECG ST depression value ($1.0mm$). However, its detection as a noisy example is not surprising. It is interesting to notice that although the cardiologists will also in the future classify similar cases in the same way, the result demonstrates that from the technical point of view the classification of patients like the patient with ID 176 into the opposite class would be more appropriate. The result illustrates the complexity of the medical decision making process.

Complex medical case (patient IDs 229, 213):

Although in the majority of examples exercise and Holter ST depression values are used for the diagnosis of CHD patients, there exist situations when the disease is diagnosed from the generally very severe heart disease conditions (severe heart failure) detected by echocardiography examination. It means that patients with IDs 229 and 213,

presented in rows number 2 and 8 of Table 5.1, respectively, are correctly classified but they are detected as noisy because their ST depression values correspond to the description of healthy subjects.

5.1.3 Comparison with HARF noise detection

The HARF noise detection approach, presented in Section 3.3, was applied to the CHD dataset to evaluate its noise detection performance in comparison to NoiseRank. The HARF noise detection approach with required high agreement on misclassification among classifiers in the random forest was used with two agreement levels: 70% and 80%. Thereby obtained noise detection results with HARF-70 and HARF-80, respectively, are summarized in Table 5.2.

Table 5.2: Instances from the CHD dataset detected by HARF-70 and HARF-80, their corresponding agreement level of misclassification, and their corresponding rank as detected by NoiseRank.

Detected by	Diagnosis	Patient ID	Agreement	NoiseRank
HARF-80	non-CHD	51	89.6% (448/500)	1.
HARF-80	CHD	229	81.8% (409/500)	2.
HARF-70	non-CHD	27	77.2% (386/500)	3.

The results show that the HARF noise detection approach with noise identification threshold of 0.7 or 0.8 manages to detect the most significant noisy instances, as obtained also by the more complex NoiseRank approach. HARF-80 returns two instances which were ranked highest by NoiseRank and present the false diagnosis example of patient with ID 51 and the complex medical case example of patient with ID 229. These two data instances correspond to text-book examples of classification noise and an outlier, respectively. The HARF-70 approach denotes one more instance as noisy, the one with patient ID 27, which was identified as a false positive medical test by the domain expert and was ranked third by NoiseRank. The qualitative evaluation of the instances detected by HARF and the comparison to the results produced by NoiseRank verifies that HARF indeed is an approach for precise noise detection.

5.1.4 Medical domain use case lessons learned

In summary, both noise detection approaches, NoiseRank and HARF, proved relevant for data driven elicitation of implicit expert knowledge, by offering to explore a wider range of ranked potentially noisy instances, or returning only the most suspicious noisy instances. The results of medical evaluation of the detected noisy examples show that their case-by-case analysis was worth additional human time and effort. The most important is the example (patient 51) for which the domain expert realized that the original patient classification was inappropriate. This example is important because it demonstrates how explicit noise detection can help increase the quality and reliability of the medical decision making process. Expert analysis of other cases actually confirmed the existence of outliers in medical decision making. The identified reasons for outlier existence proved useful for increasing the expert's awareness of such cases.

5.2 Identification of non-typical news articles

Noise detection approaches identify irregularities and errors in data, therefore they are suitable also for detecting atypical, unusual and/or irregular documents in categorized document corpora. In this use case, the aim is to detect irregular, atypical or outlier documents to be inspected by human experts in the phase of data cleaning and data understanding. To identify only those instances that are indeed noise or outliers and are worth the expert's inspection time and effort, a noise detection approach indicating a level of noisiness would be well suited. Ensemble-based noise detection by NoiseRank uses predictions of several different noise detection approaches to increase the reliability of identifying atypical documents in categorized document corpora, as well as to indicate their noisiness.

First, the news articles domain on Kenyan elections is presented.Second, the top-ranked noisy documents, detected by NoiseRank, were examined by an expert in linguistic pragmatics. Third, a comparison of NoiseRank, HARF, and a baseline method is presented. Finally, the summary of qualitative evaluation of ensemble-based noise detection on a categorized document corpus is provided.

5.2.1 News articles on the Kenyan elections

The ability of the proposed ensemble-based noise detection approaches to obtain interesting documents, which are atypical for a document collection, was tested on a corpus of documents, originating from two different newspaper sources. A subset of articles was selected from a larger corpus of newspaper articles originally collected by the IPrA Research Center, University of Antwerp, as part of the Intertextuality and Flows of Information project[1] in which an ethnographically-supported pragmatic analysis of news discourse was undertaken.

In the experiments, 464 articles (about 320,000 words) were analyzed, originating from six different daily newspapers in English, covering the time period from December 22, 2007 to February 29, 2008, concerning Kenyan presidential and parliamentary elections, held on December 27, 2007, and the crisis following the elections. Articles from the British and US press (*The Independent, The Times, The New York Times* and *The Washington Post*) formed the category "Western" (WE) and articles from Kenyan newspapers *Daily Nation* and *The Standard* were categorized as "Local" (LO). Both categories contain 232 articles. More details on this document collection can be found in Pollak et al. (2011).

These articles were preprocessed in the standard text mining way, using text tokenization, stopword removal, word stemming/lemmatization, and creation of bag-of-words (BoW) vector representations of text using binary term weights. The final dataset for the experiments in this use case was obtained by selecting the 500 most informative features of the BoW space using the χ^2 statistic. A standard workflow of document/text preprocessing for data mining tasks will be provided in Chapter 6.

5.2.2 Expert evaluation of results obtained by NoiseRank

NoiseRank was applied to the dataset of news articles on Kenyan elections (abbreviated NAKE) to obtain suspicious documents that could either present irregularities or errors in the dataset, or atypical or outlier documents of a specific category (class). The NoiseRank methodology employed six noise detection algorithms presented in Section 3.2.2: four classification filters (Bayes, RF, SVM, and NN) and two saturation filters (SF, PruneSF). The output of NoiseRank on the NAKE dataset is shown in Figure 5.1.

The documents from the NAKE dataset that were detected as noisy or outlying by more than half of the algorithms included in the NoiseRank approach were considered for manual

[1] https://www.uantwerp.be/en/rg/ipra/research/projects/

```
Rank |  Class  | ID | Detected by:
-------------------------------------------------------------------
  1.     WE      351  __Bayes_____RF_____SVM______NN______SF___
  1.     WE      356  __Bayes_____RF_____SVM____PruneSF_____SF___
  1.     WE      357  __Bayes_____RF_____SVM____PruneSF_____SF___
-------------------------------------------------------------------
  2.     LO        3  __Bayes_____RF_____SVM____PruneSF_
  2.     LO       24  __Bayes_____RF_____SVM_______NN___
  2.     LO      100  __Bayes_____RF_____SVM_______NN___
  2.     LO      161  __Bayes_____RF_____SVM_______NN___
  2.     LO      172  __Bayes_____RF_____SVM_______NN___
  2.     WE      325  __Bayes_____RF_____SVM_______NN___
  2.     WE      409  __Bayes_____RF_____SVM____PruneSF_
-------------------------------------------------------------------
  3.     LO       60  __Bayes___PruneSF_____SF___
  3.     LO      152  __Bayes_____RF_____SVM___
  3.     LO      347  __Bayes_____RF_____SVM___
  3.     WE      363  __Bayes_____RF_____SVM___
  3.     WE      369  __Bayes_____RF_____SVM___
  3.     WE      463  ____RF_____SVM______NN___
-------------------------------------------------------------------
  4.     LO       20  ____RF_____SVM___
  4.     LO       67  ____RF_____SVM___
  4.     LO      119  __Bayes_____SF___
  4.     LO      157  __Bayes___PruneSF_
  4.     LO      158  _PruneSF_____SF___
  4.     LO      184  __Bayes___PruneSF_
  4.     LO      200  _PruneSF_____SF___
  4.     LO      214  __Bayes_____NN___
  4.     LO      220  _PruneSF_____SF___
  4.     LO      221  __Bayes___PruneSF_
  4.     WE      237  ____RF_____SVM___
  4.     WE      246  __Bayes_____SVM___
  4.     WE      366  __Bayes___PruneSF_
  4.     WE      374  __Bayes___PruneSF_
  4.     WE      384  __Bayes___PruneSF_
  4.     WE      386  ___SVM______NN___
  4.     WE      407  __Bayes___PruneSF_
  4.     WE      412  ____RF______NN___
  4.     WE      415  _PruneSF_____SF___
  4.     WE      420  __Bayes_____SF___
-------------------------------------------------------------------
  5.     LO        7  _PruneSF_
  5.     LO       17  __Bayes__
  5.     LO       70  _PruneSF_
  5.     LO       83  ___SVM___
  5.     LO      129  ____NN___
  5.     LO      131  _PruneSF_
  5.     LO      160  __Bayes__
  5.     LO      180  ___SVM___
  5.     LO      186  __Bayes__
  5.     LO      213  ____NN___
  5.     WE      245  _PruneSF_
  5.     WE      311  ____RF___
  5.     WE      321  _PruneSF_
  5.     WE      327  ____RF___
  ...
```

Figure 5.1: Visual representation of noise detection output obtained by NoiseRank on the NAKE dataset. The figure shows the first 50 instances out of the 69 instances that were detected by at least one noise detection algorithm.

inspection. Hence, detailed analysis of the top 10 documents, which were identified as noise by at least four out of six noise detection algorithms (ranked 1. or 2.), was performed by the domain expert from the field of linguistic pragmatics.[1]

Instead of explaining the examined documents one by one in the order as ranked in Figure 5.1, the articles are presented grouped together by the nature of their outlierness type into the so-called *outlier groups*. Outlier group A contains only one article that in fact turned out to be problematic and should not have been incorporated in the document corpus. The linguists who originally collected the corpus later indeed excluded this article from the corpus. The articles of Outlier group B are the articles that best prove the effectiveness of noise detection methods for outlier identification. This group consists of three articles that were published in the local press, but had been written by 'Western' journalists or talking about 'Western press' coverage of local events. In Outlier group C, there are articles which are specific in their genre, being more opinion type of articles rather than 'hard news' type of articles. The Outlier group D contains three articles that are special because of their extreme document length. One of the detected articles, however, did not belong to any of the uniform outlier types.

Outlier group A: Out of topic

When building the corpus for qualitative linguistic analysis, the articles were collected using the keywords related to the main topic of interest. Noise and outlier detection can be viewed as a very effective way for quickly detecting documents which were included based on the keywords and are in fact out of the main topic of interest: they actually represent mistakes made in the process of document corpus collection.

Article WE 351 which was voted as most atypical (as it was misclassified by all the classifiers and declared as noisy also by the Saturation filter) is out of topic because it is not about the Kenyan elections or post-election violence but about violence in Kenya. It can be explained from the following point of view. When the corpus of newspaper texts on the specific theme of Kenyan elections and post-election crisis was built, the selection was based on several keywords. In this article, the elections are specially mentioned in the content that it is "not connected to post-election violence". It contains the key term but the article describes a criminal attack (a robbery for money), which is neither by the journalist, nor by its context related to the post-election violence. The article was later indeed removed from the corpus used for further linguistic analysis, since it is not about the socio-political climate but about British tourists or expatriates' misfortune.

Another reason for its misclassification might be that it reports on violence in Kenya without making a link to ethnicity, rather focusing on socio-economic motives. This is typical of the Kenyan coverage of the election crisis. While the Western newspapers usually mentioned the ethnicity of perpetrators of violence, the Kenyan newspaper shied away from ethnic labels. Here the ethnicity of the robbers is not mentioned. In Kenyan newspaper references to people involved in violence are often vague, unspecified or general, comparable to this article's reference to "a gang of six men", as well as the use of the term 'community' instead of 'tribe'. Note that in this article the typical local-class word 'community' is used, but it has a different meaning.

Outlier group B: Western journalists

This is a very interesting group of outlier articles. Three local articles which are of great interest are discussed. The first two are interesting because their author is in

[1] Roel Coesemans, PhD, from University of Antwerp, at the time of this study.

fact not a Kenyan journalist, therefore confirming the real 'outlying nature' of the article, while the third article's topic justifies its 'outlying' nature.

Article LO 172 was identified as an outlier, as it can be regarded as being a 'Western article' among the local Kenyan articles category. It is written by a Canadian freelance journalist and travel writer, who at the time worked for the Daily Nation. It was also observed that this journalist does not have the cultural sensitivity or does not follow the editorial guidelines requiring the journalists to be careful when mentioning words like 'tribe' in negative contexts. The journalist has a kind of 'Western' writing style. Although this article is published in the section of national news, traditionally covering factual news, this article also describes a topic of general human interest (mixed marriages).

Also article LO 3 was not written by a Kenyan but by a British psychologist who works in Nairobi. Note also that this article, published in the national news section, is not purely factual, but has an opinionated flavor.

The next example is very interesting. Article LO 161 is not an article by a guest journalist. It is a local Kenyan article about Western news discourse, so it contains a lot of Western voices. It consists of quotes from the international newspapers, such as Financial Times, The Guardian and The Daily Telegraph. It discusses the Western news reporting on the Kenyan election crisis.

Outlier group C: Different genre

Most articles in this corpus are 'hard news' reports, i.e. reports on actual events, but there are also some editorials and opinion articles present. Several of these articles were detected as outliers.

Article LO 100 is an opinion article taking a historical perspective and a macro-perspective. This is not typical of the Kenyan press coverage, which tends to focus on situated, localized incidents rather than putting the events in the macro-frame of nation, history, economics, or—as Western newspapers often do—ethnicity. The Kenyan press did not openly characterize the whole elections as rigged (although they admit rigging in certain districts or constituencies). But in this article, the word "rigged" is prevalent possibly because the author makes a historical comparison with Uganda and the rigging is especially related to Uganda. On the one hand, this article is a bit off the main topic, as it has a large part about Ugandan political history. Moreover, the broad African perspective and comparisons to other African countries (e.g., Rwanda) occurred much more in the Western press than in the local press.

Also article LO 24 is a different subgenre than 'hard news'. It is a personal portrait of the president and a hypothetical analysis of what might happen in some likely election scenarios. The emphasis on personal characteristics is in general more typical of the Western press because of different backgrounds of the readership. Also the mentioned stereotypes about the man (e.g., him being a gentleman of Kenyan politics, a technocratic economist with no feeling for the common man, etc.) feature prominently in the Western press, but rarely in the Kenyan press.

Outlier group D: Extreme document length

Two articles (WE 356, WE 357) are extremely short, containing just two sentences each and appearing in short notices section. On the other hand, one document (WE 409) was included in the corpus by mistake and was in fact a batch of numerous different articles from different sources, comprising more than 400 pages, due to a data collection error.

Uncategorized outlier:

For one article, WE 325, the domain expert was uncertain, whether it is an outlier or not. Nevertheless, the paper written by a Western author resembles the articles of local journalists in the sense that it covers the parliament, while the Western press focused on the presidency (compared to local press which did deal substantially with parliament news coverage). But there are other Western articles which do report on parliament matters.

5.2.3 Comparison to a baseline approach and to HARF

The performance of NoiseRank for outlier document detection on the NAKE domain was compared to the performance of a simple baseline method which uses standard document similarity measures, and to the performance of the HARF noise detection approach.

Cosine similarity based document outlierness

A simple alternative for document outlier detection is to examine how central are documents compared to other articles of the same class (LO and WE, respectively). In this setting, an atypical article for a given class is a document which is more similar to documents of the opposite class than to other documents of its own class. The cosine similarity is used as a baseline similarity measure. Let $\mathbf{a} = \{a_1, \ldots, a_n\}$ and $\mathbf{b} = \{b_1, \ldots, b_n\}$ be vector representations of documents A and B in an n-dimensional feature space of dataset $\mathcal{D}$, then their cosine similarity is the cosine of the angle between their vector representations $\mathbf{a}$ and $\mathbf{b}$.

$$coSim(\mathbf{a}, \mathbf{b}) := \cos(\angle(\mathbf{a}, \mathbf{b})) = \frac{\mathbf{a} \cdot \mathbf{b}}{||\mathbf{a}|| \cdot ||\mathbf{b}||} = \frac{\sum_{i=1}^{n} a_i \cdot b_i}{\sqrt{\sum_{i=1}^{n} a_i^2} \cdot \sqrt{\sum_{i=1}^{n} b_i^2}}$$

For both classes LO and WE their centroids were computed. These are vectors of average values computed for each feature of the feature space over the set of documents $\mathcal{C}$ from the corresponding class.

$$centoid(\mathcal{C}) = \frac{1}{|\mathcal{C}|} \cdot \left(\sum_{\mathbf{v} \in \mathcal{C}} v_1 \ , \ \ldots \ , \ \sum_{\mathbf{v} \in \mathcal{C}} v_n \right)$$

The article's outlierness was modeled by the difference between the article's similarity to the centroid of the opposite class and the similarity to the centroid of its own class.

$$outlierness(\mathbf{v}) := coSim(\mathbf{v}, centroid(\mathcal{C}'; v \notin \mathcal{C}')) - coSim(\mathbf{v}, centroid(\mathcal{C}; v \in \mathcal{C}))$$

If this difference is positive, the article can be considered as atypical or an outlier, since this means that it is more similar to the centroid of the opposite class than to the centroid of its own class, and the larger the difference, the more atypical or outlying is the document for its own class.

To see whether an article is atypical or outlying for the collection of 'local' (LO) and 'Western' (WE) newspaper articles about Kenyan elections and the crisis following the elections, the articles' cosine similarities were compared to the centroid of the LO articles and the centroid of the WE articles. The differences in cosine similarity of an article to the two centroids are presented in Figures 5.2 and 5.3 for articles of the LO and WE class, respectively.

In Figures 5.2 and 5.3, we labeled 9 documents that were identified as most atypical by NoiseRank (achieving a majority of votes) as well as by the baseline cosine similarity based outlierness modeling approach. The article that was not considered outlying by the baseline method, but detected by NoiseRank, was clearly identified by the domain expert

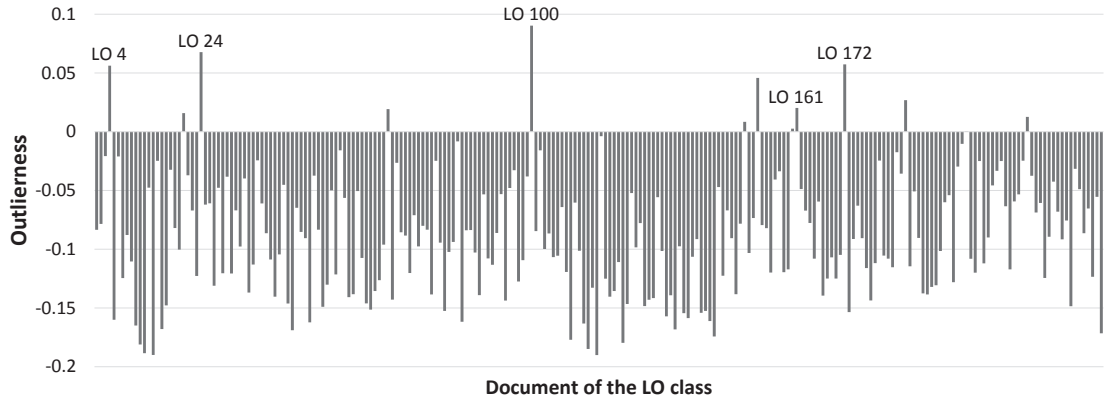


Figure 5.2: Outlierness of the LO class documents, i.e., the difference between their cosine similarity to the WE centroid and to the LO centroid.

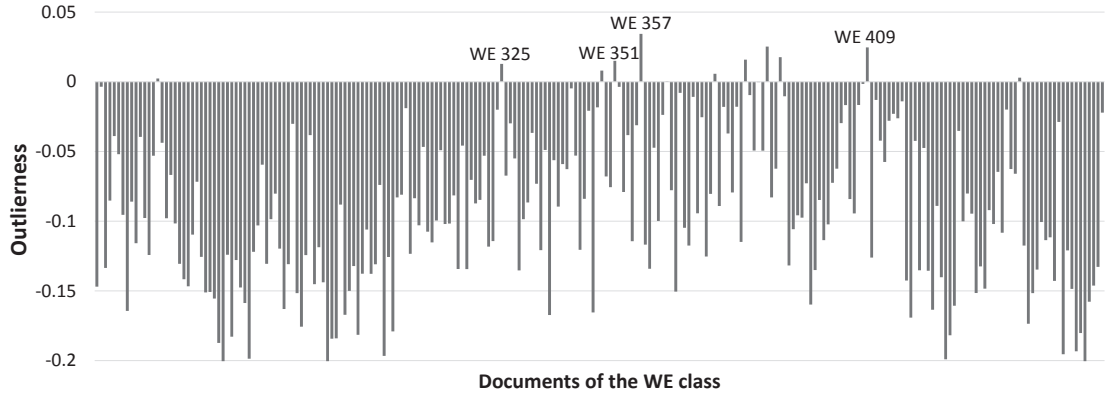


Figure 5.3: Outlierness of the WE class documents, i.e., the difference between their cosine similarity to the LO centroid and to the WE centroid.

as an 'extreme length article' (Outlier group D) that is much too short to be used in further linguistic analysis (WE 356). All other documents (except for one) that are considered as outliers according to their difference in cosine similarity to the centroids were also identified by NoiseRank, but did not achieve a majority of votes. This implies that the ourlierness modeled by the standard cosine similarity measure for document comparison offers only one view on the atypical nature of documents, whereas NoiseRank combines a variety of "opinions" and may thereby promote more significant outliers.

HARF for outlier document detection

Also the HARF noise detection approach was applied to the NAKE dataset, to compare its performance to the one of NoiseRank. In Table 5.3, the results obtained by HARF with agreement levels 70% and 60% are presented. Higher agreement levels did not yield any results.

The article WE 351 that was ranked highest already by NoiseRank, was also the only document detected by HARF-70. This proves once more that HARF used with a sufficiently high noise identification threshold enables precise detection of most significant noisy or outlier instances.

Furthermore, by lowering the required agreement level of misclassification to 60%, HARF identified five more instances as noise. Three of those instances, WE 357, LO 100 and LO 172, were also among the top ranked by NoiseRank, whereas instances LO 20 and WE

Table 5.3: Instances from the NAKE dataset detected by HARF-70 and HARF-60, their corresponding agreement level of misclassification, and their corresponding rank as detected by NoiseRank.

Detected by	Diagnosis	Article ID	Agreement	NoiseRank
HARF-70	WE	351	73.8% (369/500)	1.
HARF-60	LO	20	62.6% (313/500)	4.
HARF-60	WE	357	62.6% (313/500)	1.
HARF-60	LO	100	60.6% (303/500)	2.
HARF-60	WE	412	60.6% (303/500)	4.
HARF-60	LO	172	60.4% (302/500)	2.

412 did not exceed the majority of votes in the ensemble. Nevertheless, article LO 20 was identified as a 'Different genre' outlier by the domain expert, since it is a strongly biased opinion article, almost political propaganda. However, for article WE 412, the domain expert was not certain, whether it is an outlier or not.

5.2.4 News articles use case lessons learned

In summary, the qualitative evaluation of NoiseRank on the news articles on the Kenyan elections showed that the detected documents that were ranked highest were indeed atypical or outlier articles, since they were one of the following: out of topic, local articles written by a Western journalist, different genre articles, or articles of extreme length, i.e., data collection errors. Furthermore, the standard measure for comparison of vector representations of documents, i.e., the cosine similarity, was used to produce a simple baseline model of document outlierness. This baseline approach confirmed that the documents detected by NoiseRank are indeed atypical for their own class, as they are more similar to documents of the other class than to the documents of their own class. Similarly, the ensemble-based HARF-70 detected article WE 351, which was ranked highest by NoiseRank, thereby confirming that the article is highly atypical, as well as confirming that HARF is a precise noise detection approach.

Moreover, the expert evaluation of outlier articles shows that the proposed ensemble-based noise detection approaches are very effective in anomaly detection and can help the domain expert to discover various types of outliers in the data. This enables the expert to better understand the data and thereby supports his or her further decision making in the data analysis process.

5.3 Detecting domain outliers in cross-domain link discovery

In literature-based discovery, introduced by Swanson (1986), the goal is to identify interesting terms or concepts which relate different domains. This section reveals that a majority of these domain bridging concepts can be found in outlier documents which are not in the mainstream domain literature. In this use case, different classification-based noise detection methods were combined to identify domain outlier documents and to explore their potential for supporting the bridging concept discovery process. The experimental evaluation was performed on the classical migraine-magnesium (Swanson, 1988) and the recently explored autism-calcineurin (Petrič et al., 2006) domain pairs.

This section begins with an overview of link discovery in cross-domain literature mining. Next, two domain-pair datasets that were used for bridging concept detection are described, followed by the evaluation of the detected sets of outlier documents as sources of domain bridging terms. A summary on supporting cross-domain link discovery by ensemble-based noise detection concludes this section.

5.3.1 Cross-domain link discovery

Scientific literature serves as a basis of research and discoveries in all scientific domains. In literature-based discovery (Swanson, 1986), one of the interesting goals is to identify terms or concepts which relate different domains, as these terms may represent germs of new scientific discoveries.

The aim of this section is to present the applicability of ensemble-based noise detection to support scientists in their creative knowledge discovery process when analyzing scientific papers of their interest. The presented research follows (Koestler, 1964) stating that scientific discovery requires creative thinking to connect seemingly unrelated information and that domain-crossing associations, called *bisociations*, are a crucial mechanism for progressive insights and paradigm shifts in the history of science.

Based on the definition of bisociations—defined by Koestler (1964) and further refined by Dubitzky et al. (2012)—this section addresses the task of supporting the search for bisociative links that cross different domains. A simplified setting is considered, where a scientist has identified two domains of interest (two different scientific areas or two different contexts) and tries to find concepts that represent potential links between the two different contexts. This simplified cross-context link discovery setting is usually referred to as the *closed discovery* setting (Weeber et al., 2001). Like in Swanson (1986) and Weeber et al. (2001), the problem of literature mining in medical domains was addressed, where papers from two different scientific areas are available, and the task is to support the scientist in cross-context literature mining.

Swanson designed the *ABC model* approach that investigates whether an agent A is connected with a phenomenon C by discovering complementary structures via interconnecting phenomena B (see Figure 5.4)[1]. Two literatures are complementary if one discusses the relations between A and B, while a disparate literature investigates the relations between B and C. If combining these relations suggests a previously unknown meaningful relation between A and C, this can be viewed as a new piece of knowledge that may contribute to a better understanding of phenomenon C.

In a *closed discovery process*, where domains A and C are specified by the expert at the beginning of the discovery process, the goal is to search for bridging concepts (also called b-terms) b in B in order to support the validation of the hypothesized connection between A and C (see Figures 5.4 and 5.5). Smalheiser and Swanson (1998) developed an online system

[1] Uppercase letter symbols A, B and C are used to represent sets of terms, and lowercase symbols a, b and c to represent single terms.

ARROWSMITH, which takes as input two sets of titles from disjoint domains A and C and lists b-terms that are common to literature A and C; the resulting bridging terms (b-terms) are used to generate novel scientific hypotheses. As stated by Swanson et al. (2006), the major focus in literature-based discovery has been on the closed discovery process, where both A and C are specified in advance. Figure 5.5 illustrates the closed discovery process for a real-life case of exploring the migraine-magnesium domain pair.

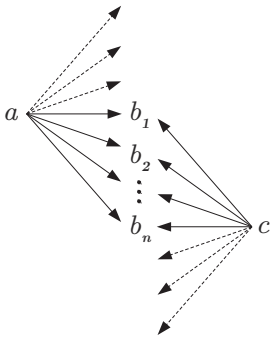


Figure 5.4: Closed discovery process as defined by Weeber et al. (2001).

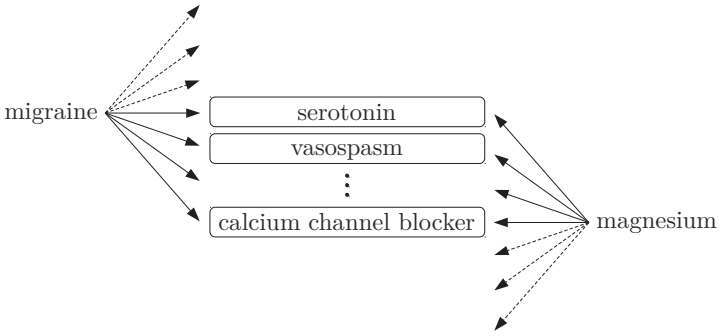


Figure 5.5: Closed discovery when exploring migraine and magnesium documents, with b-terms as identified by Swanson et al. (2006)

Srinivasan (2004) developed an algorithm for bridging concept identification that is claimed to require the least amount of manual work in comparison with other literature-based discovery studies. However, it still needs substantial time and human effort for collecting evidence relevant to the hypothesized connections. In comparison, one of the advantages of the approach presented in this use case is that the domain expert needs to be involved only in exploring the potential b-terms in outlier documents, instead of exploring all the most frequent potential b-terms in all the documents.

Petrič et al. (2010, 2012) have developed a methodology for inspecting outlier documents as a source for speeding up the b-term detection process. Like in this work—and similar to the definition of outliers in statistics where an outlier is defined as an observation that is numerically distant from the rest of the data—they also focus on outlier observations (documents) that lie outside the overall pattern of the given (class) distribution. More specifically, their methodology focuses on the search for b-terms in outlier documents identified by OntoGen, a semi-automated tool for topic ontology construction (Fortuna et al., 2007). Opposed to their approach, which uses k-means clustering in OntoGen to detect outlier documents included in the opposite cluster, our approach uses several classification algorithms to identify misclassified documents as domain outliers, which are inspected for containing domain bridging terms.

In this use case, classification noise filters and their ensembles are used for detecting outlier documents. Documents of a *domain pair* dataset (i.e., the union of two different domain literatures) that are misclassified by a classifier can be considered as domain outliers or borderline documents, since these instances tend to be more similar to regular instances of the opposite domain than to the instances of their own domain. The utility of domain outliers (borderline documents) as relevant sources of domain bridging terms is the topic of study of this section.

5.3.2 Migraine-magnesium and Autism-calcineurin domain pairs

This section shortly describes two datasets which were used to evaluate the proposed outlier detection approach for cross-domain literature mining. Additionally, the description of the

preprocessing techniques is provided and some basic statistics for the reader to get a better insight into the data.

The first dataset, the *migraine-magnesium* domain pair, was previously well researched by different authors (Petrič et al., 2009; Swanson, 1986, 1990; Swanson et al., 2006; Weeber et al., 2001). In the literature-based discovery process Swanson managed to find more than 60 pairs of articles connecting the migraine domain with the magnesium deficiency via several bridging concepts. In this process Swanson identified 43 *b*-terms connecting the two domains of the *migraine-magnesium* domain pair (Swanson et al., 2006).

The second dataset, the *autism-calcineurin* domain pair, was introduced by Petrič et al. (2006, 2007) and Urbančič et al. (2007) and later also researched by Macedoni-Lukšič et al. (2011); Petrič et al. (2009, 2010) and Petrič et al. (2012). Autism belongs to a group of pervasive developmental disorders that are portrayed by an early delay and abnormal development of cognitive, communication and social interaction skills of a person. It is a very complex and not yet sufficiently understood domain, where precise causes are still unknown. Alike Swanson, Petrič et al. (2009) also provide *b*-terms, 13 in total, whose importance in connecting autism to calcineurin (a protein phosphatase) is discussed and confirmed by the domain expert.

The *b*-terms, which were identified in each of the two domain pair datasets, were used as the gold standard to evaluate the utility of domain outlier documents in the cross-context link discovery process. Table 5.4 presents the *b*-terms for the *migraine-magnesium* and the *autism-calcineurin* domain pair datasets used in the experiments of this use case.

Both datasets were retrieved from the PubMed database (`http://www.ncbi.nlm.nih.gov/pubmed`) using the keyword query; however, the migraine-magnesium dataset was selected using an additional filtering condition. It was necessary to select only the articles published before the year 1988 as this was the year when Swanson published his research about this dataset and thus making an explicit connection between migraine and magnesium domain. Preprocessing was done in a standard text mining way, using the preprocessing steps described in Juršič et al. (2012): (a) text tokenization, (b) stopword removal, (c) word stemming/lemmatization using LemmaGen lemmatizer for English (Juršič et al., 2010), (d) construction of N-grams which are terms defined as a concatenation of 1 to N words

Table 5.4: Bridging terms – *b*-terms identified by Swanson et al. (2006) and Petrič et al. (2009) for the *migraine-magnesium* and *autism-calcineurin* domain pair, respectively.

migraine-magnesium	**autism-calcineurin**
serotonin, spread, spread depression, seizure, calcium antagonist, vasospasm, paroxysmal, stress, prostaglandin, reactivity, spasm, inflammatory, anti inflammatory, 5 hydroxytryptamine, calcium channel, epileptic, platelet aggregation, verapamil, calcium channel blocker, nifedipine, indomethacin, prostaglandin e1, anticonvulsant, arterial spasm, coronary spasm, cerebral vasospasm, convulsion, cortical spread depression, brain serotonin, 5 hydroxytryptamine receptor, epilepsy, antimigraine, 5 ht, epileptiform, platelet function, prostacyclin, hypoxia, diltiazem, convulsive, substance p, calcium blocker, prostaglandin synthesis, anti aggregation	synaptic, synaptic plasticity, calmodulin, radiation, working memory, bcl 2, type 1 diabetes, ulcerative colitis, asbestos, deletion syndrome, 22q11 2, maternal hypothyroxinemia, bombesin

Table 5.5: Some basic properties and statistics of the domain pair datasets used in the experimental evaluation.

Dataset name	**migraine-magnesium**	**autism-calcineurin**
Document source	PubMed	PubMed
Query terms	"migraine" |"magnesium" (condition: year<1988)	"autism" |"calcineurin"
Number of retrieved doc.	8,058 (2,425 |5,633)	15,243 (9,365 |5,878)
Part of document used (text)	title	abstract
Average text length	11 words, 12 terms	180 words, 105 terms
Term definition	3-grams, min. freq. 2	1-grams, min. freq. 15
Number of distinct terms	13,524	5,255
Number of *b*-terms	43	13
Num. of doc. with *b*-terms	394 = 4.89%	1672 = 10.97%

that appear consecutively in text with minimum supporting frequency, (e) creation of standard bag-of-words (BoW) representation of text using term-frequency-inverse-document-frequency (tf-idf) or binary (depends on classification algorithm) term weights. Besides this standard workflow, which will be presented in more detail in Section 6.2.2, we additionally removed from the dataset all terms (N-grams) containing words which were used as query terms during document selection. Experiments showed that the correlation between the domain class and the query terms is too high for an outlier detection algorithm to find a reasonable number of high quality outliers. A summary of statistics on the datasets used in our experiments is presented in Table 5.5.

The 43 *b*-terms identified by Swanson in the standard *migraine-magnesium* dataset were retrieved from article titles only (Swanson et al., 2006). Therefore, in the experiments only article titles were used as well. In the preprocessing of this dataset, 3-grams were constructed to obtain more features for each document despite a relatively low average word count. On the other hand, for the *autism-calcineurin* dataset, which contains titles and the abstracts, only 1-grams were used and the minimum supporting frequency of terms was set higher to reduce the number of features due to computational limitations.

5.3.3 Evaluation of outlier documents as sources of domain bridging terms

The aim of this use case is to show that ensemble-based noise detection techniques can be used for supporting the search for cross-domain links between concepts from two disparate literatures A and C. Furthermore, the goal of this section is to provide experimental evidence for the hypothesis that outliers can be used as the focus of exploration to speed-up the search for bridging concepts between different domains of expertise (Petrič et al., 2010, 2012). In other words, this section provides experimental results that verify the assumption that exploring outlier documents simplifies the discovery of *b*-terms (bridging concepts) that establish previously unknown links between literature A and literature C, as the hypothesis of this work is that most bridging concepts occur in outlier documents.

In contrast to previous work in which k-means clustering was used to detect outlier documents (Petrič et al., 2010, 2012), in this use case different classification noise filtering approaches were used for domain outlier detection. Instances of a domain pair dataset that are misclassified by a classifier can be considered as domain outliers, since these instances tend to be more similar to regular instances of the opposite domain than to instances of their own domain. These instances actually present borderline documents between the two domains of a domain-pair dataset. In other words, if an instance of class A is classified in

the opposite class C, it is considered to be an outlier of domain A, and vice versa. The two sets of domain outlier documents are denoted with *O(A)* and *O(C)*, respectively. Figure 5.6 depicts this principle.

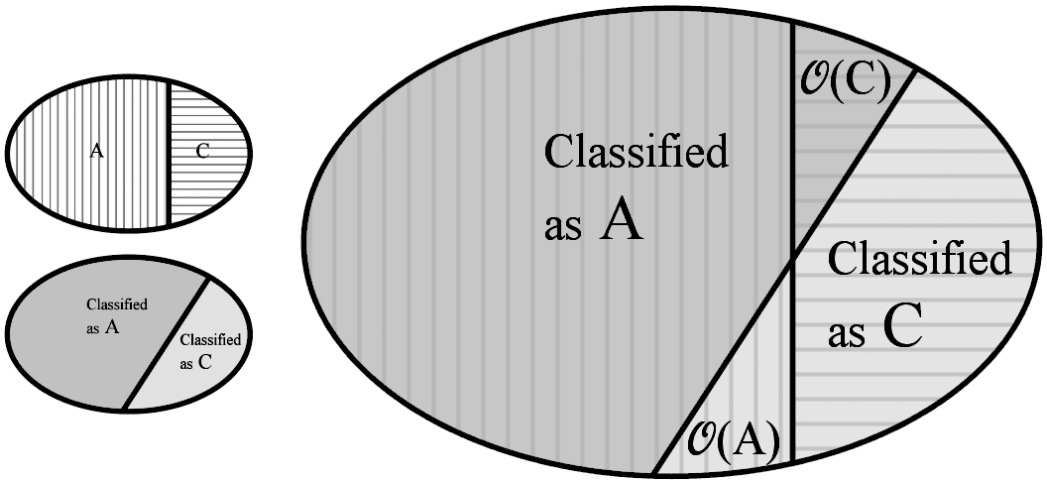


Figure 5.6: Detecting outliers of a domain pair dataset with classification filtering.

The experiments were designed to validate that sets of outlier documents are rich on b-terms and contain significantly more b-terms than sets of arbitrary documents. Outlier detection was performed by classification filtering using three different classifiers: Naïve Bayes (abbreviated: Bayes), Random Forest (RF) and Support Vector Machine (SVM). In addition to the outlier sets obtained by these three classification filters, also the union of these outlier sets was examined and the so-called "Majority" outlier set containing outlier documents that were detected by at least two out of three classification filters. Other noise detection approaches presented in Section 3.2.2 were not used due to computational limitations experienced on the two evaluated domain-pair datasets.

The experiments were performed on the migraine-magnesium and the autism-calcineurin domain pair datasets, described in Section 5.3.2. The relevance of the detected outlier documents, in terms of their potential for containing domain bridging terms, was measured by inspecting 43 terms known as bridging terms appearing in the migraine-magnesium domain pair and 13 known b-terms in the autism-calcineurin domain pair. Tables 5.6 and 5.7 present the size of all examined sets of outlier documents and the amount of b-terms they contain, for the migraine-magnesium and autism-calcineurin dataset, respectively.

Columns of Tables 5.6 and 5.7 present the numbers of outlier documents (and contained b-terms) identified by different outlier detection approaches, together with percentages showing their proportion compared to the given dataset. The rows present these numbers separately for each class, for both classes together, and—for the needs of results validation explained below—for a random sample of documents in the size of the detected outlier set.

These results show that all five outlier subsets[1] of each of the two domain pairs contain from 70% to over 90% (for the "Union" subset) of b-terms, on average in less than 10% of all documents from the migraine-magnesium dataset and in less than 5% of all documents of the autism-calcineurin dataset. This means that by inspecting outlier documents, which represent only a small fraction of the datasets, a great majority of b-terms can be found, which substantially reduces the time and effort needed by the domain expert to discover cross-domain links.

[1] *Outlier subset* is used instead of *outlier set* to emphasize its relation to the entire dataset of documents. The terms are used interchangeably, however they always refer to a set of detected outlier documents that belong to a certain domain pair.

Table 5.6: Numbers of documents and *b*-terms (bT) in different outlier sets, with percentages showing their proportion compared to all documents in the migraine-magnesium domain. The bT percentages can be interpreted as recall of *b*-terms.

Class	All docs		Bayes		RF		SVM		Union		Majority	
	Docs	**bT**	**Docs**	**bT**	**Docs**	**bT**	**Docs**	**bT**	**Docs**	**bT**	**Docs**	**bT**
MIG	2,425	43	248	23	772	26	192	12	895	34	237	20
	(100%)	(100%)	(10%)	(53%)	(32%)	(60%)	(8%)	(28%)	(37%)	(79%)	(10%)	(47%)
MAG	5,633	43	335	27	124	19	170	24	475	33	131	21
	(100%)	(100%)	(6%)	(63%)	(2%)	(44%)	(3%)	(56%)	(8%)	(77%)	(2%)	(49%)
Total	8,058	43	**583**	**32**	**896**	**36**	**362**	**29**	**1,370**	**40**	**368**	**30**
	(100%)	(100%)	**(7%)**	**(74%)**	**(11%)**	**(84%)**	**(4%)**	**(67%)**	**(17%)**	**(93%)**	**(5%)**	**(70%)**
Randomly	8,058	43	**583**	**19**	**896**	**24**	**362**	**14**	**1,370**	**29**	**368**	**14**
sampled	(100%)	(100%)	**(7%)**	**(44%)**	**(11%)**	**(56%)**	**(4%)**	**(33%)**	**(17%)**	**(68%)**	**(5%)**	**(33%)**

Table 5.7: Numbers of documents and *b*-terms (bT) in different outlier sets, with percentages showing their proportion compared to all documents in the autism-calcineurin domain. The bT percentages can be interpreted as recall of *b*-terms.

Class	All docs		Bayes		RF		SVM		Union		Majority	
	Docs	**bT**	**Docs**	**bT**	**Docs**	**bT**	**Docs**	**bT**	**Docs**	**bT**	**Docs**	**bT**
AUT	9,365	13	349	8	77	7	147	6	388	9	139	7
	(100%)	(100%)	(4%)	(62%)	(1%)	(54%)	(2%)	(46%)	4%)	(69%)	(1%)	(54%)
CAL	5,878	13	316	7	65	5	145	7	397	10	98	6
	(100%)	(100%)	(5%)	(54%)	(1%)	(38%)	(2%)	(54%)	(7%)	(77%)	(2%)	(46%)
Total	15,243	13	**665**	**9**	**142**	**9**	**292**	**9**	**785**	**12**	**237**	**9**
	(100%)	(100%)	**(4%)**	**(69%)**	**(1%)**	**(69%)**	**(2%)**	**(69%)**	**(5%)**	**(92%)**	**(2%)**	**(69%)**
Randomly	15,243	13	**665**	**8**	**142**	**5**	**292**	**6**	**785**	**9**	**237**	**6**
sampled	(100%)	(100%)	**(4%)**	**(63%)**	**(1%)**	**(35%)**	**(2%)**	**(46%)**	**(5%)**	**(67%)**	**(2%)**	**(46%)**

To confirm that these results are not due to chance (do not hold for just any arbitrary subset that has the same size as an outlier set), for each of the five outlier sets 1,000 subsets were randomly sampled (all of them having the same size as their corresponding outlier set) in order to present the average *b*-term occurrences in randomly sampled subsets. The last row of Tables 5.6 and 5.7 shows that the sets of outlier documents contain on average more than 30% more of all *b*-terms in the migraine-magnesium dataset and more than 20% more of all *b*-terms in the autism-calcineurin dataset than randomly sampled sets of the same size.

A comparison of the above-discussed results relative to the whole migraine-magnesium and autism-calcineurin datasets is summarized in Figures 5.7 and 5.8, respectively.

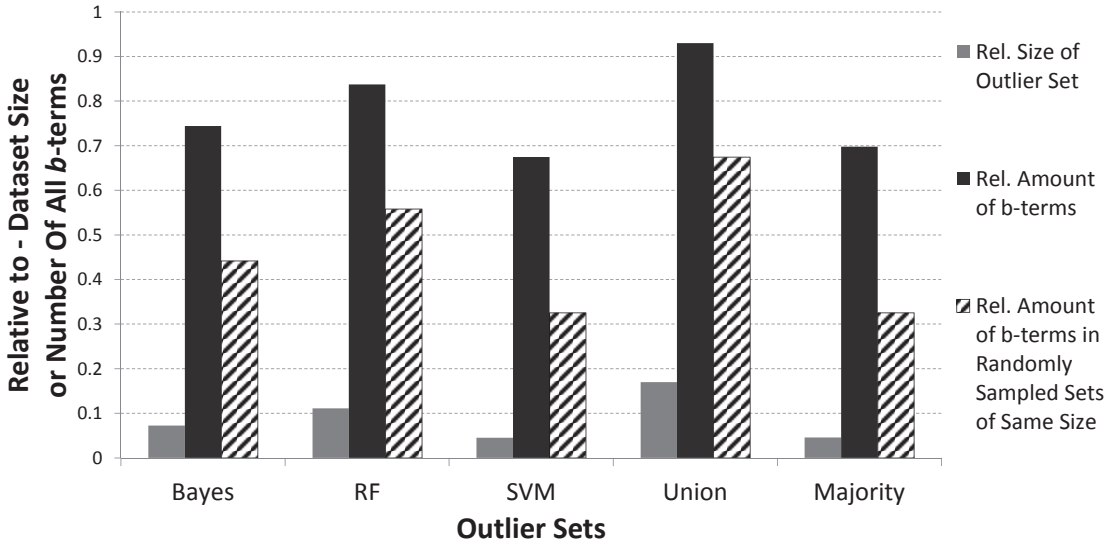


Figure 5.7: Relative size of outlier sets and the amount of *b*-terms for the migraine-magnesium dataset.

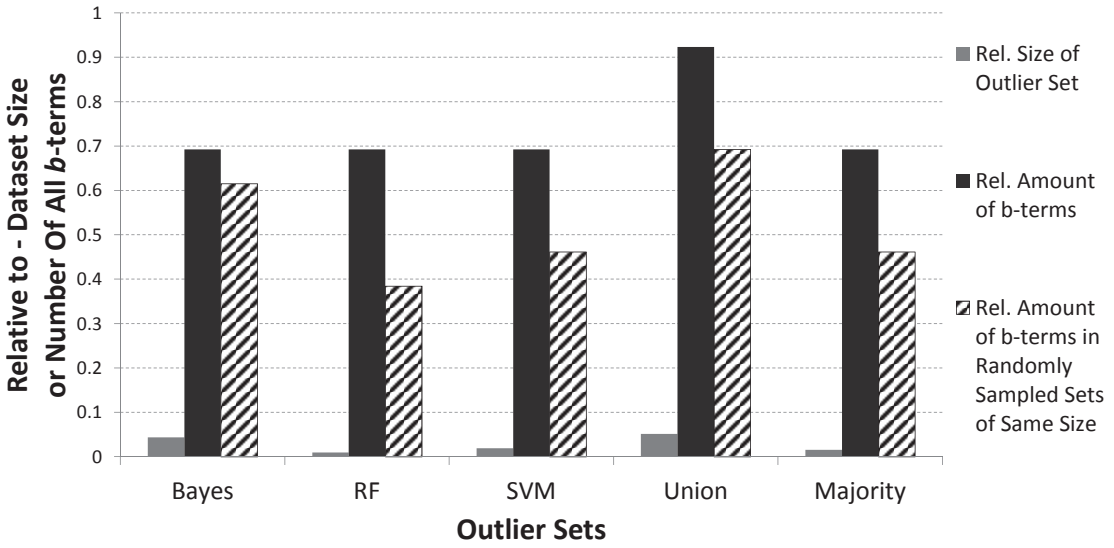


Figure 5.8: Relative size of outlier sets and the amount of *b*-terms for the autism-calcineurin dataset.

Additionally, relative frequencies of *b*-terms in the detected outlier sets were compared to their relative frequencies in the whole dataset, i.e., the fractions of documents containing

a certain b-term among the documents of a chosen set. Figure 5.9 presents the increase of relative frequencies of b-terms in the "Majority" outlier set detected on the migraine-magnesium dataset.

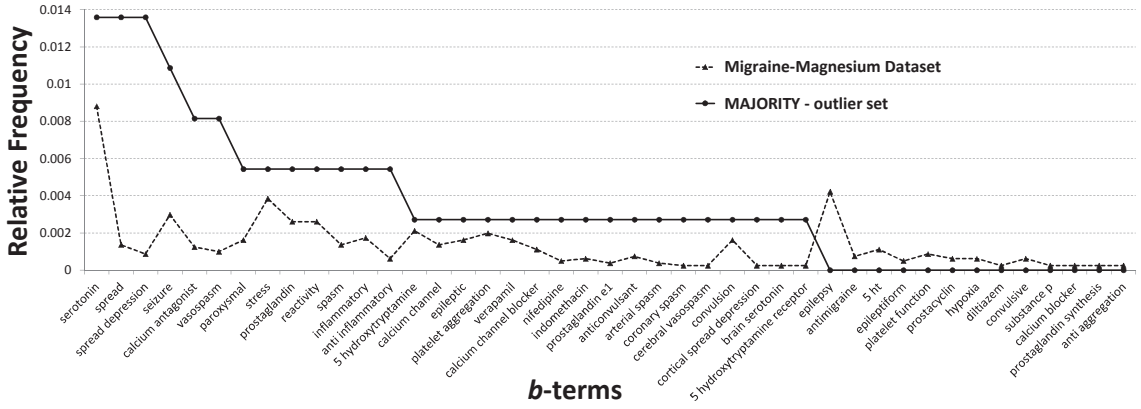


Figure 5.9: Comparison of relative frequencies of bridging terms in the entire migraine-magnesium dataset and in the "Majority" set of outlier documents detected by three different outlier detection methods.

The "Majority" outlier set approach proved to have the greatest potential for bridging concept detection. Firstly, because of the best ratio among the proportion of the size of the outlier subset and the proportion of b-terms which are present in that outlier subset (see Table 5.7 and Figure 5.7), and secondly, because the relative frequency of all the b-terms present in the "Majority" outlier set is higher compared to the entire migraine-magnesium dataset, as can be clearly seen from Figure 5.9.

Similarly, encouraging results for the "Majority" outlier set detected on the autism-calcineurin dataset can be observed in Figure 5.10. Note that the scale of the chart in Figure 5.10 is different from the scale of the chart in Figure 5.9.

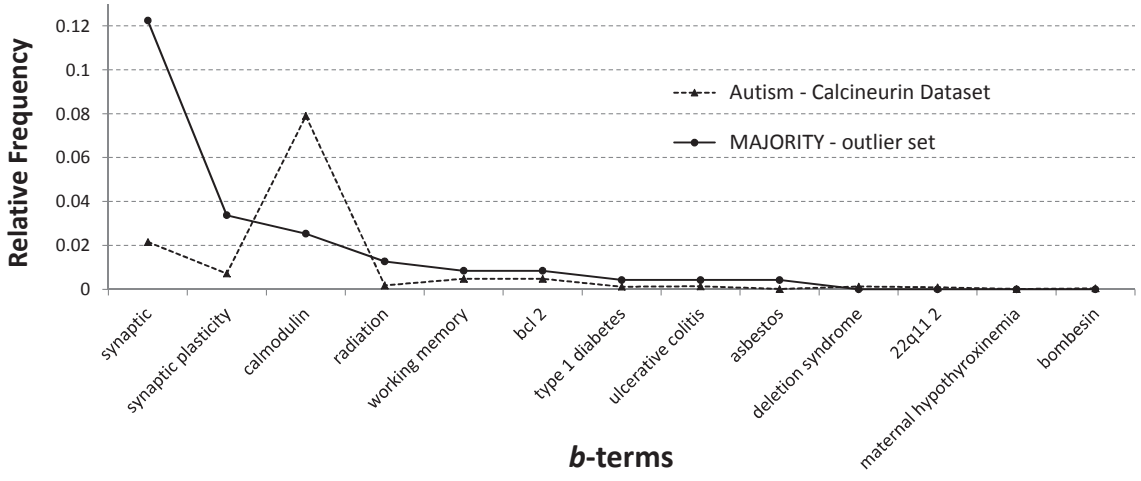


Figure 5.10: Comparison of relative frequencies of bridging terms in the entire autism-calcineurin dataset and in the "Majority" set of outlier documents detected by three different outlier detection methods.

All b-terms that are present in the "Majority" outlier set, except for one ("*calmodulin*"), have a higher relative frequency in the outlier set compared to the relative frequency in the entire dataset. Although (1) the RF outlier set is the best in terms of the ratio among

the proportion of the size of the outlier subset and the proportion of b-terms which are present in that outlier subset, and (2) the "Majority" outlier set is the second best (for the autism-calcineurin dataset), in general we prefer the "Majority" outlier set for bridging concept detection. The majority approach is more likely to give adequate outliers on various datasets, in contrast to a single outlier detection approach, since it reduces the danger of overfitting or bias to a certain domain by requiring the agreement of at least two outlier detection approaches for a document to declare it as a domain outlier.

5.3.4 Cross-domain link discovery use case lessons learned

In summary, this use case investigates the potential of ensemble-based noise detection methods in literature mining for supporting the discovery of bridging concepts between disparate domains. The articles for the migraine-magnesium and the autism-calcineurin domain pairs were retrieved from the PubMed database. In the experiments five sets of outlier documents were obtained for each domain pair by three different outlier detection methods, their union and a majority voting approach. Experimental results show that inspecting outlier documents considerably contributes to the bridging concept discovery process, since it enables the expert to focus only on a small fraction of documents which is rich on concept bridging terms (b-terms). Thus, the effort needed for finding cross-domain links is substantially reduced, as it requires to explore a much smaller subset of documents, where a great majority of b-terms are present and more frequent.

6 Implementation and Public Availability

The goal of this thesis is not only to present the proposed ensemble-based noise detection and ranking methodology, and the developed visual performance evaluation approaches, but also to provide a publicly available infrastructure with the aim of ensuring the accessibility of the developed approaches and the repeatability of the experiments and the obtained results. This chapter covers the implementation of the developed approaches in two main sections. The first section describes the implementation of a *Noise and outlier detection package* into a web-based data mining platform. This package enables to use and reuse the NoiseRank methodology for ensemble-based noise detection and ranking. Furthermore, it includes the implementation of the proposed visual performance evaluation approaches and all the necessary utilities for repeating the presented experiments. The second section describes a newly developed web-based platform solely for visual performance evaluation of machine learning and data mining algorithms. The platform enables to create, save and share standard as well as novel performance visualizations. Its services were made available also via a web API and in the aforementioned web-based data mining platform.

6.1 Noise and outlier detection package

This section presents the implementations of NoiseRank and Viper, enabling their public accessibility, repeatability of the experiments and the obtained results, as well as the integration of user specific noise detection algorithms available as web services. First, a web-based platform for composition and execution of data mining workflows is described in which all the components required for ensemble-based noise ranking and visual performance evaluation of noise detection have been integrated. Second, the implementations of the proposed NoiseRank methodology and Viper performance evaluation methodology are presented. Last, general workflows for experiments on noise detection performance and text preprocessing are described.

6.1.1 Embedding environment

The NoiseRank and Viper approaches were implemented in the ClowdFlows web-based platform for composition and execution of data mining workflows (Kranjc et al., 2012). In contrast to other data mining environments (Weka, Orange, etc.), the ClowdFlows environment does not require any installation, since all the processing load is taken from the users machine to remote servers, and therefore it may entirely be run in any modern web browser. This enables public accessibility of our approaches, repeatability and sharing of the presented experiments and the obtained results, as well as the inclusion of web services allowing the application of new noise detection algorithms.

In this work, the functionality of the ClowdFlows platform was extended by implementing the building blocks, called widgets[1], required for the construction of workflows

[1] Widgets are processing units used as components of a workflow, which—given some input data and/or parameters—perform a certain computation or visualization task.

for explicit noise detection with NoiseRank and the construction of workflows for visual performance evaluation of noise detection with Viper. The implementation ensures the compatibility with algorithms from two known data mining platforms, Weka (Hall et al., 2009) and Orange (Demšar et al., 2013). Implemented were 33 widgets and 11 web services covering the following functionalities.

- Loading datasets (File to Orange Data Table (ODT), UCI dataset to ODT, 26 standard UCI datasets included),
- Data format conversion (ODT to String or ARFF Weka Instances, ODT to TAB file, CSV file or ARFF file)
- Adding class noise,
- Noise filtering (Classification filter, Saturation filter, Ensemble, and HARF widgets),
- Classification algorithms (12 classifier widgets from Orange and 11 classifiers from Weka, implemented as web services),
- Performance evaluation (Evaluate Detection Algorithms, Aggregate Detection Results, Evaluate Repeated Detection, Average and Standard Deviation),
- Visualization (Data Table Information, Data Table Viewer, Evaluation Results Table, Evaluation Results Chart, Viper Visual Performance Evaluation and NoiseRank interactive widgets).

The ClowdFlows platform offers subprocess widgets and the *'for loop'* subprocess widget, which allow repeated experimental evaluation with different initialization parameters. A special widget enables to use 26 standard datasets from the UCI repository (Bache and Lichman, 2013) that were also included into the platform and can be used for extensive experimental evaluation of data mining algorithms. Experiments can be repeated and shared as public workflows accessible through unique URL addresses. The ClowdFlows platform is available at `http://www.clowdflows.org/`.

6.1.2 NoiseRank implementation

The aim of the NoiseRank (ensemble-based noise detection and ranking) methodology is to support domain experts in identifying noisy, outlier or erroneous data instances. The user should be able to select the noise detection algorithms to be used in the ensemble-based noise detection process. Therefore, the NoiseRank methodology was implemented as a workflow in the ClowdFlows platform, which now offers widgets implementing classification and saturation noise filters, and enables the inclusion of external user specific noise detection algorithms available as web services. Figure 6.1 presents the NoiseRank workflow applied to the medical dataset using the noise detection algorithms presented in Section 3.2.2.

The NoiseRank methodology workflow returns a visual representation of a list of potentially noisy or outlier instances ranked according to the decreasing number of elementary noise detection algorithms[1] which identified an instance as noisy. Figure 6.2 shows the actual output of the NoiseRank workflow (widget) when applied to the coronary heart disease (CHD) data presented in Section 5.1.1. This widget opens an interactive window allowing the user to select instances from the ranked list, inspect them, and decide if they are noise due to errors in the data (attribute or class noise), outliers or special cases of the concept represented in the domain, or just false alarms. The NoiseRank workflow presented in Figure 6.1 is publicly accessible at: `http://www.clowdflows.org/workflow/115`.

[1] Note that all the noise detection algorithms described in Section 3.2.2 were used, except for *PruneSF* which is unable to cope with missing values.

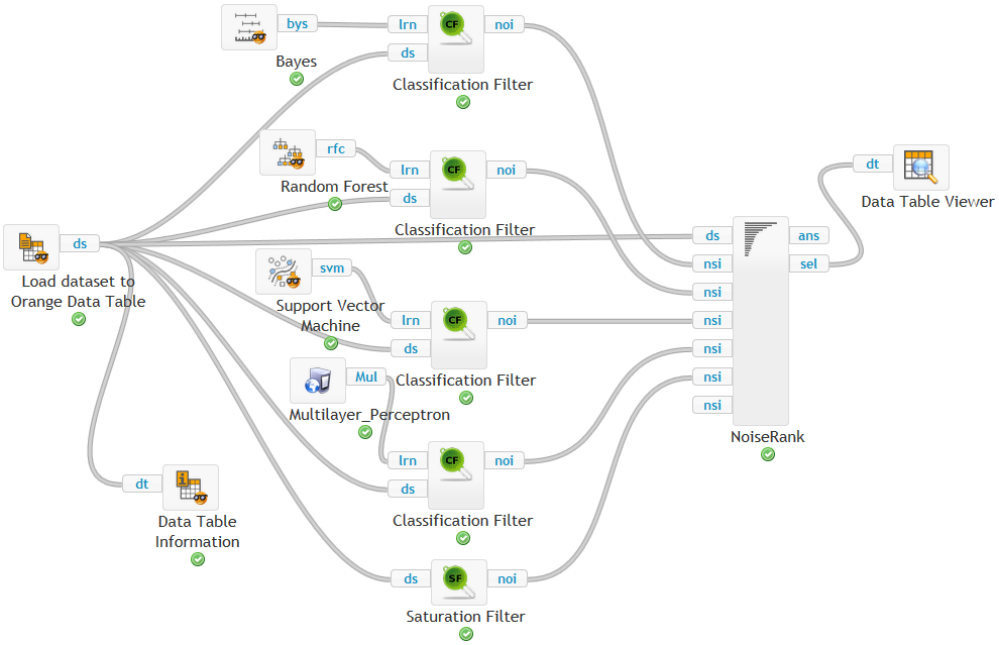


Figure 6.1: Example workflow of the NoiseRank methodology.

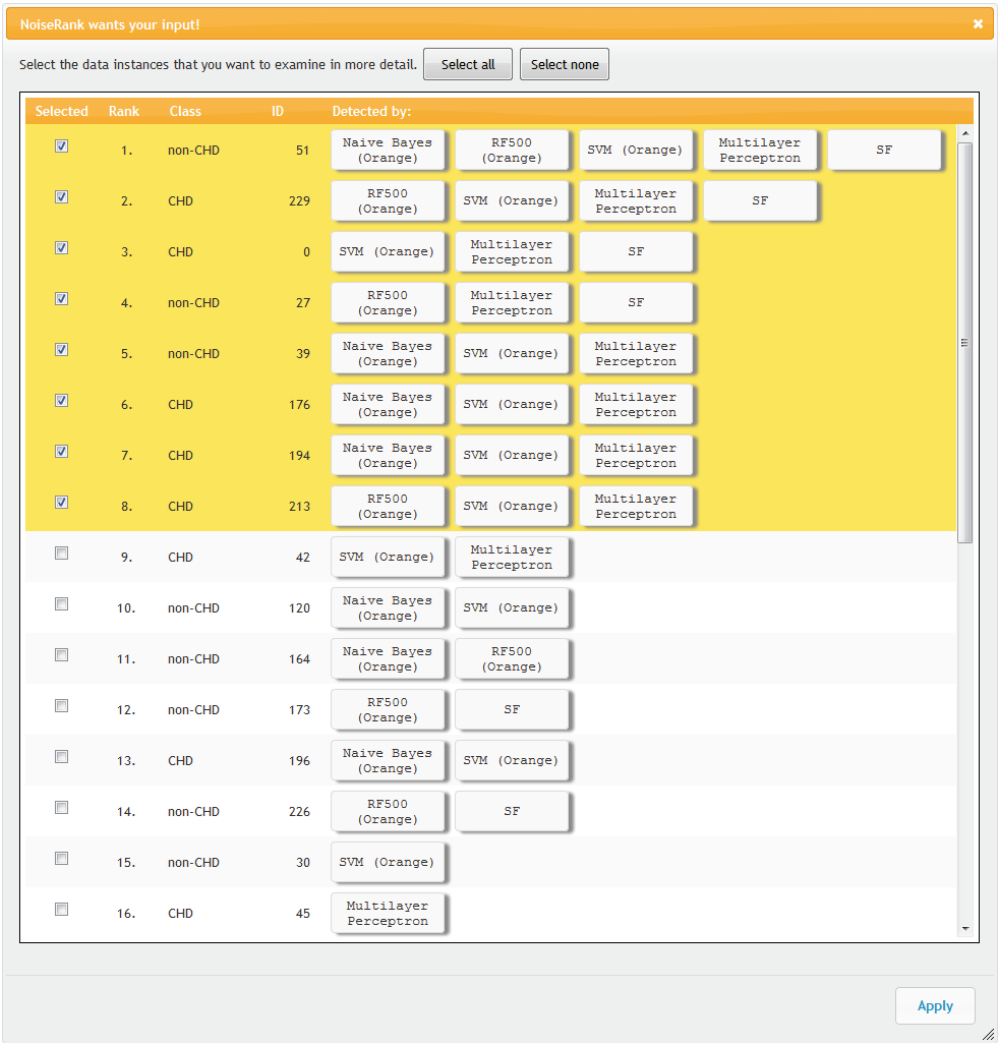


Figure 6.2: Visualization output of the NoiseRank widget.

6.1.3 VIPER implementation

The widgets required for constructing the workflow of the visual performance evaluation methodology VIPER have been also implemented in the CLOWDFLOWS platform. Thereby, the user can evaluate his own noise detection algorithm (or an ensemble of algorithms) and compare its noise detection performance to the performance of other (existing) algorithms in the precision-recall space. User-specific noise detection algorithms that are available as web services can be easily included into the workflow for the VIPER visual performance evaluation. Widgets supporting different input formats are available: string, ORANGE data table and WEKA instances, while the output of noise detection web services should be a simple list of zero-based indices of the detected noisy instances in the initial dataset. Figure 6.3 presents an example workflow of the VIPER visualization approach for performance comparison and evaluation of a sample noise detection algorithm (HARF) against other existing algorithms on a dataset with artificially injected random noise. This workflow is publicly accessible at `http://www.clowdflows.org/workflow/43`.

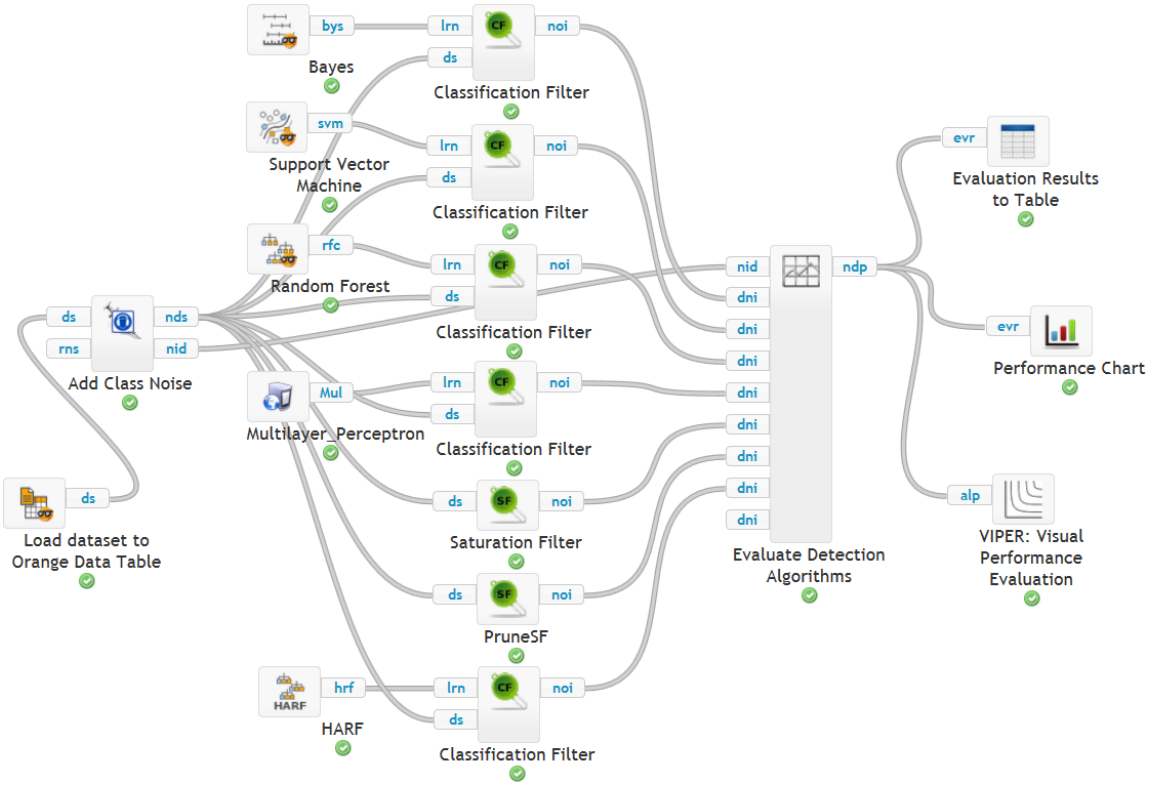


Figure 6.3: An example workflow for VIPER evaluation and comparison of new noise detection algorithms against other algorithms.

At the end of such workflows, which model noise detection performance evaluation experiments, the VIPER *Visual Performance Evaluation* widget is placed. This is the main widget that visualizes the results and enables performance comparison and evaluation in the precision-recall space. The output of the VIPER *Visual Performance Evaluation* widget is an interactive chart[1] visualizing performance results of noise detection algorithms according to the VIPER visual evaluation approach. F-scores and F-isolines are computed and drawn based on the provided β parameter in the *Evaluate* widget (the default value is set to 1). The ε-proximity of maximal achieved precision of noise detection is set as a parameter of the

[1] The implementation uses the HIGHCHARTS charting library, available at `http://www.highcharts.com`.

VIPER widget (by default set to 0.05). Additionally, the user can obtain more information, like algorithm name, precision, recall and F-score results, including standard deviations in case of repeated evaluation, in the following way:

- by hovering over algorithm performance results or points on the F-isolines,
- by clicking on the algorithm performance results which draws their corresponding F-isolines enabling easier comparison, and
- by selecting an area which is zoomed in, thereby enabling the inspection of performance results in a densely 'populated' results area.

An example of the interactive VIPER visualization showing some of its functionality is presented in Figure 6.4.

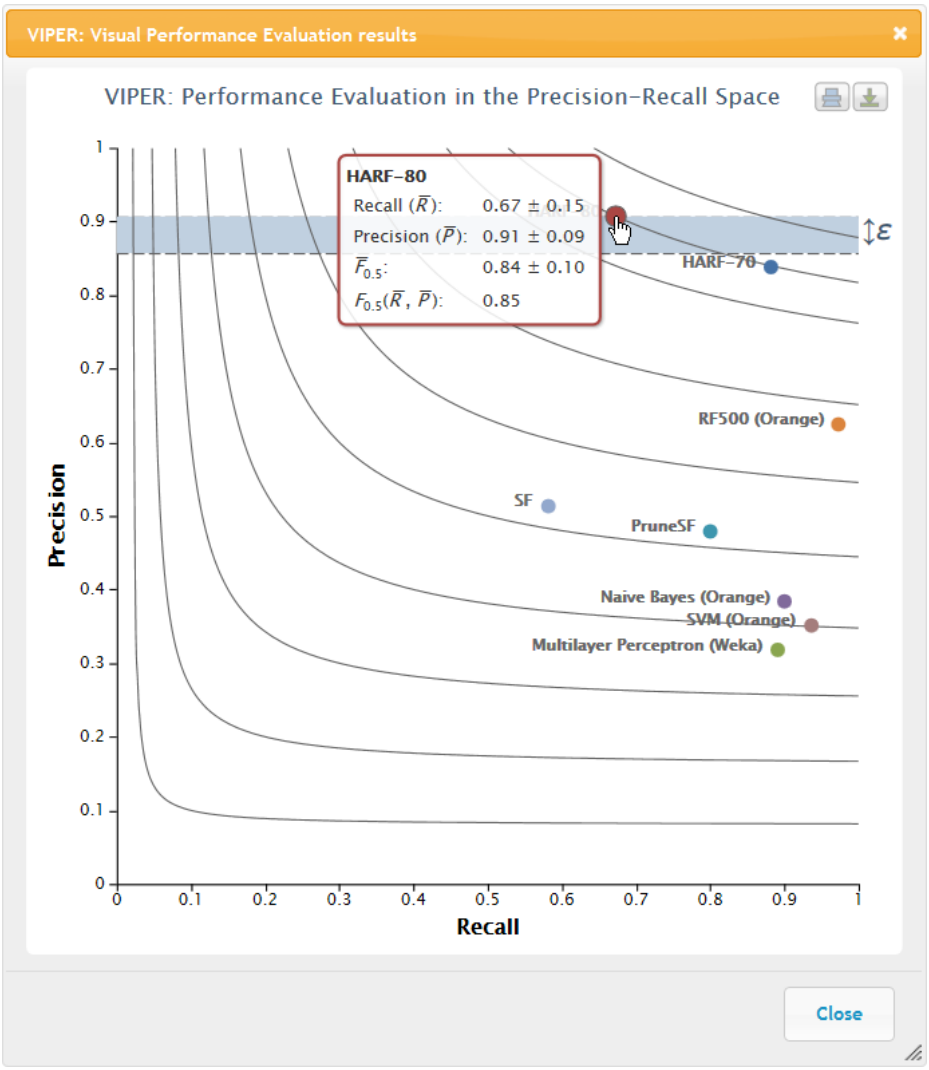


Figure 6.4: Additional functionality of the VIPER interactive visualization. Visualization of selected F-isolines for easier comparison and tooltips with additional information on specific algorithm performance.

6.2 Workflows ensuring experiment repeatability

All the experiments presented in this thesis can be easily repeated within the Clowdflows platform. In addition to the workflows shown in the previous section, this section provides a workflow for quantitative performance evaluation of noise detection algorithms, and the standard text preprocessing workflow for generating datasets suitable for machine learning algorithms from sets of text documents.

6.2.1 Experiments for evaluation of noise detection performance

The relatively simple workflow in Figure 6.1 produced the results that proved to be of significant importance to the domain expert, offering interesting insight into his domain data, as shown in Section 5.1. Similarly, the workflow in Figure 6.3 produces visual performance results that easily convey a first impression of the new algorithm's performance in comparison with the other algorithms. However, the infrastructure that was implemented enables to run also more complex workflows. An example is the repetitive experiment with different initiation parameters for the evaluation of the performance of different noise detection algorithms and their ensembles. Figure 6.5 shows the workflow of the repetitive experiment of noise evaluation, and Figure 6.6 the repeating part of the workflow in the for loop subprocess widget. This is the workflow that produced the noise detection results obtained on the CHD dataset with 5% noise presented in Chapter 4.

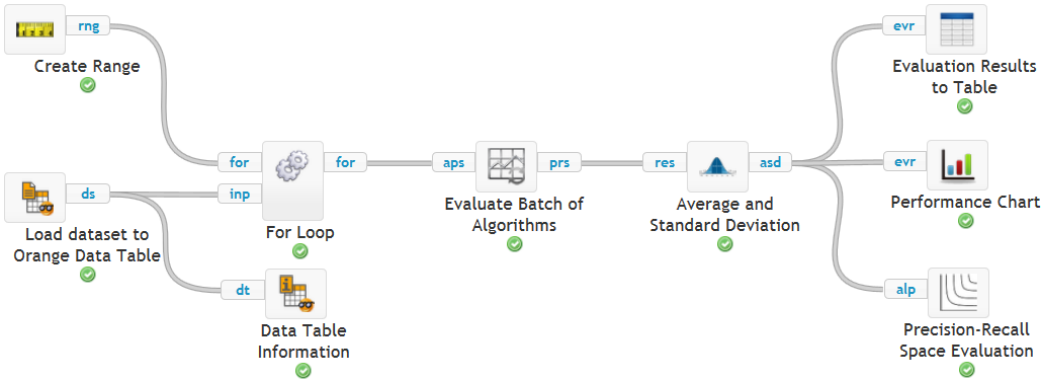


Figure 6.5: Workflow for the computation of average noise detection performance.

6.2.2 Text preprocessing workflow

Applying ensemble-based noise detection to specific real-life problems that work with textual input requires text preprocessing. Converting text data into a suitable input for machine learning algorithms is achieved by the following steps (adopted after Juršič (2013)):

1. text tokenization - the continuous character stream of a document is broken into meaningful tokens, usually words (or terms when a controlled vocabulary is available),
2. stopword removal - removing predefined words from a language that usually carry no relevant semantic information (e.g., articles, prepositions, conjunctions, etc.),
3. stemming or lemmatization - the process that converts each word/token into the morphologically neutral form,
4. creation of Bag-of-Words (BoW) document representation model, a vector of term-frequency-inverse-document-frequency (tf-idf) or binary term weights.

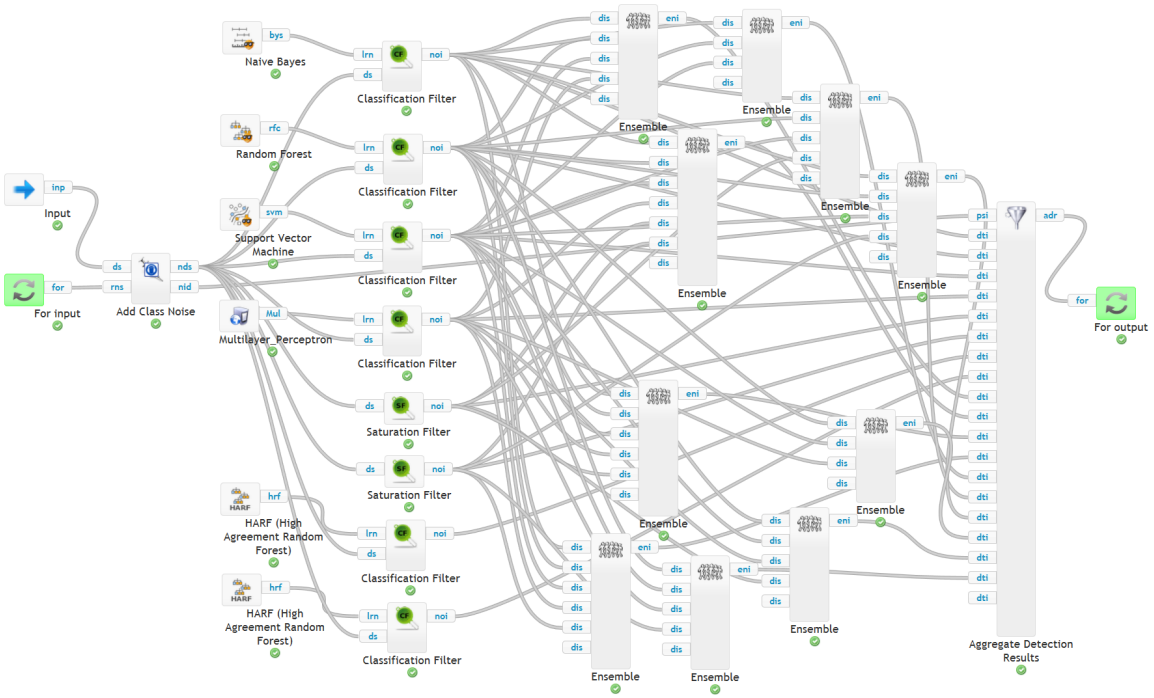


Figure 6.6: The 'for loop' subprocess of the workflow in Figure 6.5, depicting noise detection on a dataset with randomly injected class noise.

The workflow of this process starting with a collection or a file of raw text data is presented in Figure 6.7. This workflow uses text mining tools from LATINO, a link analysis and text mining toolbox[1]. The input to this workflow is a file which must contain in each line the text of one document from the document corpus that the user wants to include in the data/text mining task. The result of the workflow is a sparse dataset of Bag-of-Words text representations of the input documents.

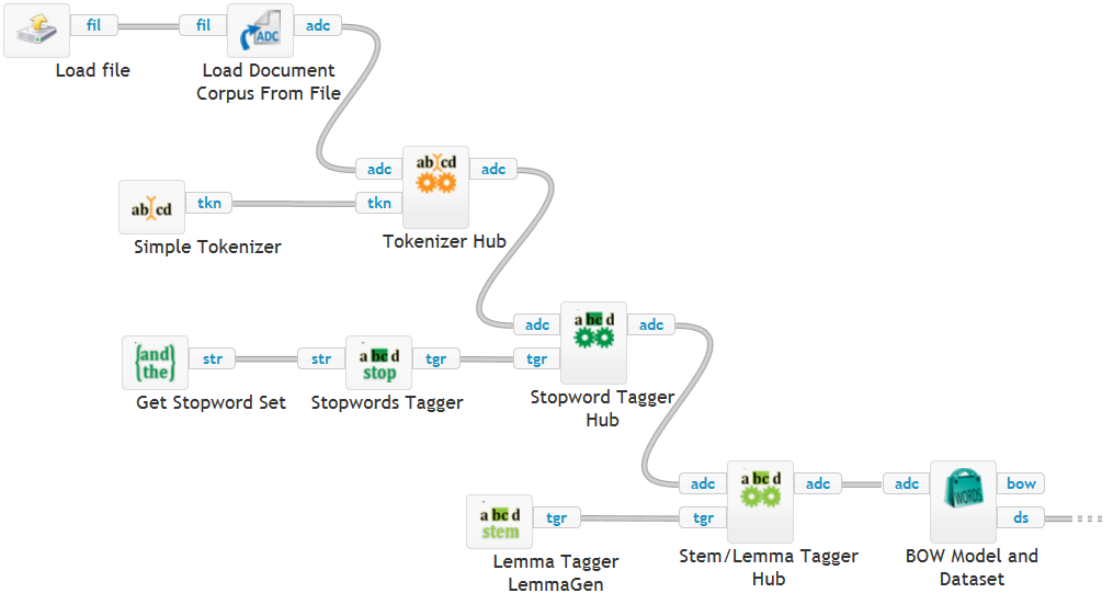


Figure 6.7: Workflow for standard text preprocessing of documents from a file of raw text data.

[1] `http://sourceforge.net/projects/latino/`

6.3 ViperCharts platform

Performance visualization proved to be very helpful in the evaluation of noise detection results, as presented in Chapter 4. However, performance visualizations are not only beneficial for the evaluation of noise detection, but also for evaluating other machine learning tasks.

This section presents the ViperCharts web-based platform that was developed for visual performance evaluation of classification, prediction, noise detection, or information retrieval algorithms used in machine learning and data/text mining. The goal was to provide a web environment which enables intuitive visualization of results and sharing of performance charts that are of interest to machine learning, data/text mining, as well as information retrieval practitioners. The platform enables to create interactive charts for easy and intuitive evaluation of performance results. It includes standard visualizations and extends them by offering alternative evaluation methods, like the Viper approach with F-isolines in the PR space, and by establishing relations between the corresponding presentations like the Precision-Recall and ROC curves. Additionally, the interactive performance charts can be saved, exported to several formats, and shared via unique web addresses. A web API to the service is also available.

This section first reviews existing visualization tools and the common performance visualizations used in machine learning and data/text mining. Then the ViperCharts platform along with its features is presented. Last, the integration of the performance visualizations of ViperCharts in the web-based data mining platform ClowdFlows is described.

6.3.3 Performance visualization

Empirical evaluation of classification algorithms is mainly focused on their ability to correctly predict the desired class of data instances. Hence, several performance measures are derived from the confusion matrix, which provides the distribution of the predicted classes over the actual classes of a dataset. The choice of an evaluation measure depends on the task to be solved. Typically, classification is evaluated by accuracy which is the average of correct predictions over all the classes, however in the case of unbalanced class distributions, medical diagnostic or information retrieval tasks other evaluation measures which focus on a certain class, like precision, recall, false positive rate or specificity, may be more desirable.

Tools for graphical representation of numerical data are provided by different software applications for calculation, numerical computing, statistical computing and statistical analysis, like Microsoft Excel[1], MATLAB[2], R[3] and SPSS[4]. Also several JavaScript, SVG or Flash based charting libraries and charting software, such as Highcharts[5], D3[6] or Google Chart Tools[7], offer data visualizations which can be embedded in web pages. However, among the usually supported chart types, only specific line and scatter charts are interesting from the point of visualizing algorithm performance, since they depict the relation between different variables and/or performance measures of algorithms.

Common visualizations of algorithm performance in machine learning include the Lift curves (Witten and Frank, 2005), ROC curves (Fawcett, 2006; Flach, 2010), Precision-Recall Curves (Manning et al., 2008), and more specific Cost curves (Drummond and Holte, 2006), Rate-driven and Kendall curves (Hernández-Orallo et al., 2013), for probabilistic classifiers or ranking algorithms, and scatter charts in the ROC space or Precision-Recall space (PR space) for discrete classification algorithms. Existing environments for machine learning and data mining, like Weka (Hall et al., 2009), RapidMiner (Mierswa et al., 2006), KNIME (Berthold et al., 2009) and Orange (Demšar et al., 2013), as well as MATLAB and

[1] http://office.microsoft.com/en-us/excel/ [2] http://www.mathworks.com/products/matlab/
[3] http://www.r-project.org/ [4] http://www.ibm.com/software/analytics/spss/
[5] http://www.highcharts.com [6] http://d3js.org [7] https://developers.google.com/chart/

R, only offer support for the computation and visualization of ROC and Lift curves, whereas support for other performance visualizations may be available as third-party packages for different programing languages.

This section presents the ViperCharts platform, which aims to provide charting support for the evaluation of machine learning, data/text mining and information retrieval algorithms. It offers a range of performance visualizations and reveals the relations among different performance measures. Its main distinctive feature is its web-based design which requires no installation on the user's system, enables sharing of performance results, and offers enhanced visual performance evaluation through interactive charts and additional advanced functionality. Furthermore, it provides a unique environment for computing, visualizing and comparing performance results of different algorithms for various evaluation measures. The ViperCharts platform is accessible online at `http://viper.ijs.si`.

6.3.4 Implementation of the platform

The platform is designed to serve as a tool for data mining practitioners with the goal to produce visualizations of performance results achieved by their algorithms. ViperCharts is a web application running in the client's Web browser. The server side is written in Python[1] and uses the Django Web Framework[2].

Creating a chart requires two simple steps. First, the user selects the desired chart type, choosing among Scatter chart, Curve chart, Column chart and Critical difference diagram. Where Scatter charts visualize the performance of discrete prediction algorithms in the ROC or PR space, Curve charts visualize PR, Lift, ROC, Cost, Rate-driven, and Kendall curves, Column charts visualize arbitrary values for a selection of algorithms, and Critical difference diagrams visualize the results of statistical significance tests on the difference in performance of competing algorithms. Second, the required data for the specific chart is copy-pasted into the provided form. Different data input formats are supported: TAB delimited data for copy-pasting from spreadsheet applications, CSV data, and JSON formatted data. Finally, a *Draw chart* button triggers the chart visualization.

The desired performance measures are calculated from the provided data and stored in the platforms' database. In this way the visualization components have access to the data of a specific chart whenever the chart needs to be displayed.

Charts are drawn with the help of a JavaScript charting library, called Highcharts[3]. For each chart type a specific template was created which includes individual functionality available for a certain type of performance visualization. For example, in PR space charts the novel *F-isoline* evaluation approach was included, which enables to simultaneously visually evaluate algorithm performance in terms of recall, precision and the F-measure. An additional novelty of the platform is the possibility to compare different performance visualizations for the same results. PR space chart results can be shown in tabular and column chart presentations, whereas more interestingly for a single curve chart all other corresponding curve chart types can be generated for side by side evaluation comparison. For example PR curves give a more informative picture of the algorithm's performance compared to ROC curves when dealing with highly skewed datasets, which provides additional insight for algorithms design, as discussed by Davis and Goadrich (2006). The Critical difference diagrams offer, in addition to the graphical representation of performance ranks and critical difference intervals, a full results summary including statistical test results and enable selection of different p-values for the tests. Screenshots of the ViperCharts platform showing a chart of the PR space with F-isolines can be found in Figure 6.8, a ROC curve chart in

[1] `http://www.python.org/` [2] `http://www.djangoproject.com` [3] `http://www.highcharts.com`

the tabbed Curve charts panel for switching among curve chart types in Figure 6.9, and a Critical difference diagram in Figure 6.10.

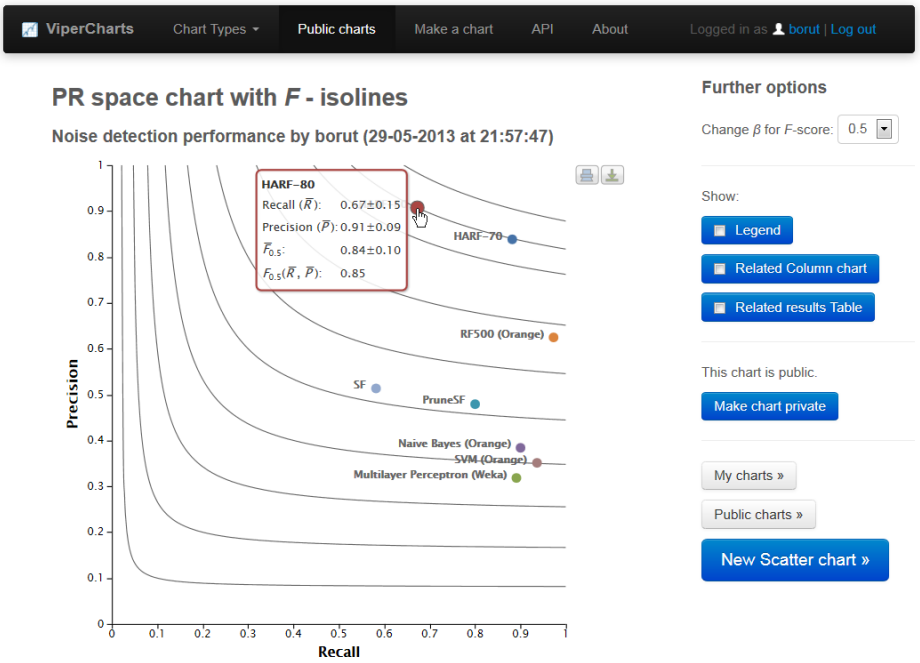


Figure 6.8: A screenshot of the ViperCharts platform showing the performance of discrete prediction algorithms in the PR space enhanced by F-isolines.

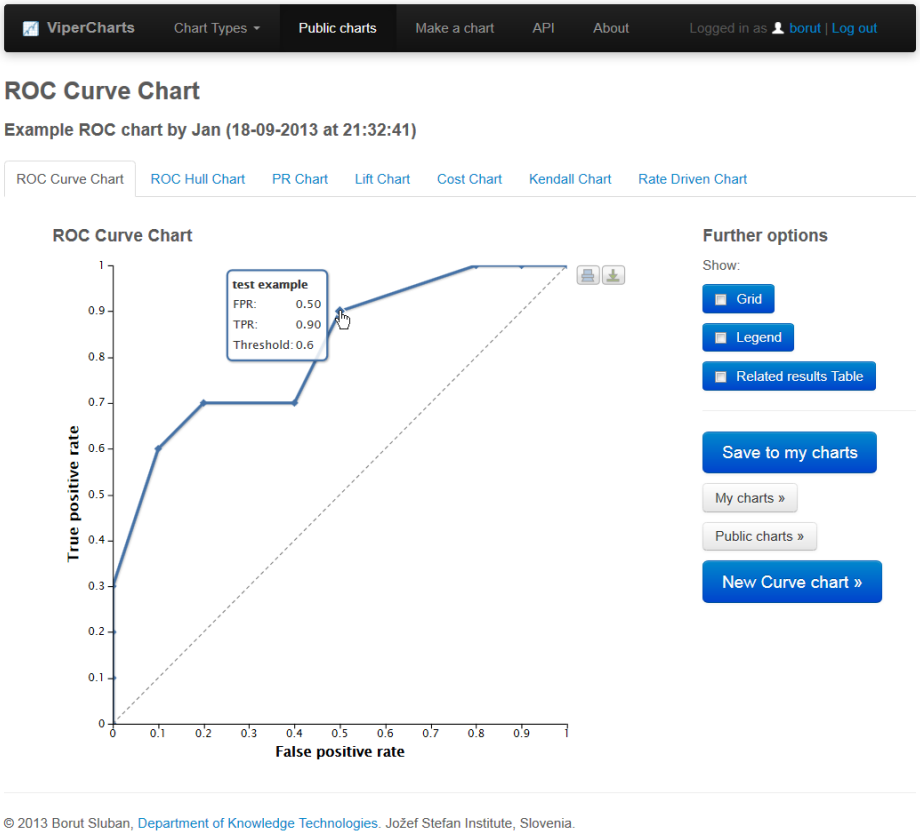


Figure 6.9: A screenshot of the ViperCharts platform showing the ROC curve results in the tabbed Curve charts panel for switching among curve chart types.

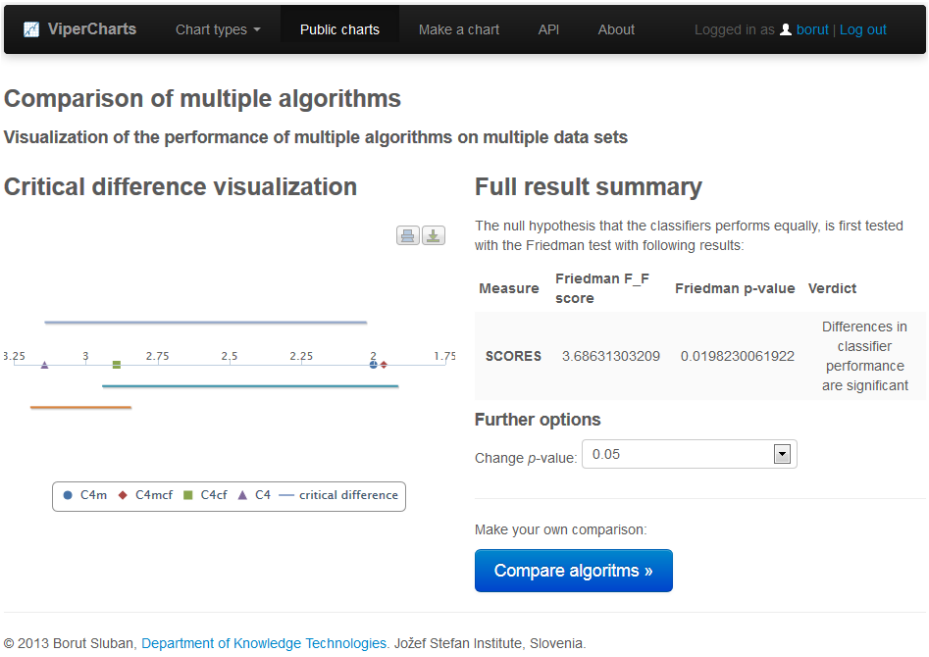


Figure 6.10: A screenshot of the ViperCharts platform showing the Critical difference diagram of a statistical significance test on the performance results of four algorithms.

The ViperCharts platform enables users to make their charts public and share them with research colleagues, export and publish them in research papers, or include them in their web sites. The sharing and embedding of charts is made easy by a unique URL address assigned to each chart. Exporting the charts into SVG, JPG, PNG or PDF formats is taken care of by the exporting module of the charting library that is used for chart visualization.

The charting services of ViperCharts can be accessed also directly via its web API at `http://viper.ijs.si/api/` from the user's favorite programing language. A performance chart can be obtained from the ViperCharts API by following three simple steps.

1. Prepare your data in the required JSON format.
2. Make a POST request to `http://viper.ijs.si/api/` with the appropriate JSON formatted data.
3. Follow the provided link in the response to access your performance chart.

The API can be accessed by executing only few lines of code. An example in Python, showing the first two steps listed above, is provided below.

```
>>> import urllib2, json
>>> url = "http://viper.ijs.si/api/"
>>> data = json.dumps({"chart":"rocc",
                       "data":[
                               { "name":"alg1",
                                 "actual":[1,1,1,0,0,0],
                                 "predicted":[1.2,0.8,0.6,0.8,0.5,0.6]
                               }
                              ]
                      })
>>> req = urllib2.Request(url, data)
>>> req.add_header('Content-Type', 'application/json')
>>> response = urllib2.urlopen(req)
```

The API returns a JSON formatted response with two parameters: a message reporting success or errors, and the URL to the generated performance chart.

```
>>> response.read()
‘{"msg": "ROC curve chart succesfully created.",
  "url":
     "http://viper.ijs.si/api/curve/roc/cfeb5099-7630-4cfe-9ef2-d689347c4bbf/"
}’
```

The last step is to follow the provided URL that points to a plain page displaying only the requested chart. This allows the chart to be embedded into other web sites or displayed in other user applications.

6.3.3 Integration in the ClowdFlows data mining platform

Visual performance evaluation is desired to be ‘close’ to where the experiments are performed, therefore were the ViperCharts visualizations also integrated in the web-based data mining platform ClowdFlows (Kranjc et al., 2012). A *Visual performance evaluation* (ViperCharts) package was added to the ClowdFlows platform, implementing a total of 11 visualization widgets and few data preprocessing widgets. An example of a workflow showing the use of the ViperCharts widgets for visual performance evaluation in the ClowdFlows platform is presented in Figure 6.11.

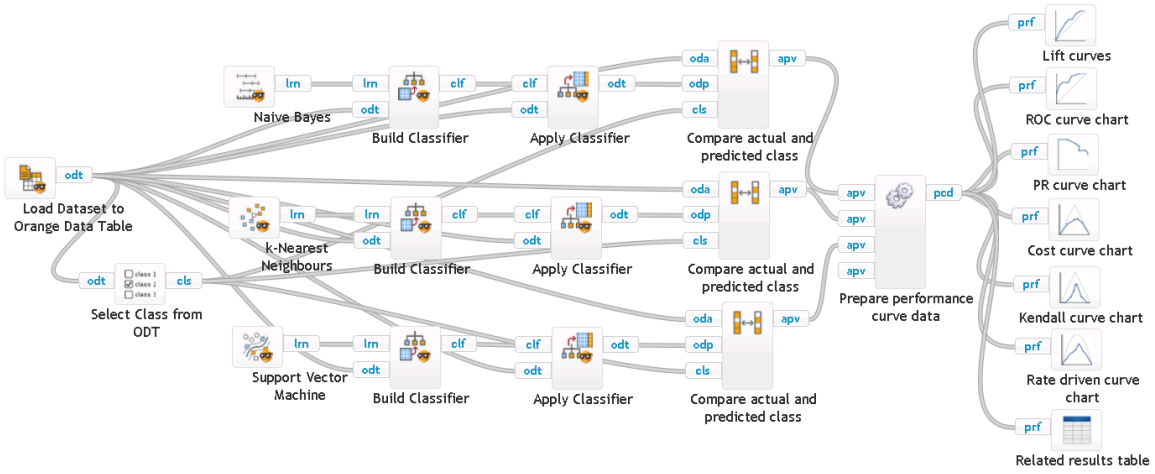


Figure 6.11: Workflow of visual performance evaluation with the ViperCharts widgets in the ClowdFlows platform.

7 Summary and Further Work

The research presented in this thesis addresses the problem of finding unusual instances in data that are either erroneous and may be harmful to data mining in terms of computational, descriptive and predictive performance, or may present special cases in the dataset, which can be interesting for gaining new insights about the observed domain. The main goal of this thesis was to propose a solution enabling the identification of noisy and outlier instances in data, which can be used by the domain experts as well as by the developers of noise detection algorithms. Hence, the thesis presents the developments of a methodology for user-guided detection of unusual data instances, and the design of a visualization approach for easy and intuitive evaluation and comparison of noise detection performance.

The thesis first presents an overview of different noise and outlier detection techniques, inspects the evaluation methods used for assessing the performance of noise and outlier detection, and reviews existing software tools for noise and outlier handling. The first task addressed was the development of a methodology that would yield a noise detection approach to be used by domain experts and algorithm developers, which would enable the user to compose the desired approach to noise detection, return an intuitive display of the obtained results, and allow the user to inspect the presented noisy instances. This was accomplished by the core contribution of this thesis: the developed NoiseRank methodology for user-guided ensemble-based noise detection, which incorporates all the aforementioned specifications. Employing a selection of noise detection algorithms, NoiseRank was successfully applied to several real-life problems as confirmed by the corresponding domain experts. Additionally, a simple yet effective noise detection approach HARF was developed enabling precise identification of most unusual or noisy data instances.

The second task has focused on the quantitative evaluation of noise detection performance. Reviewing the related literature on noise and outlier handling revealed the diversity of evaluation practices and the lack of a simple presentation enabling a multifold perspective on noise detection performance. As the second main contribution of this thesis, the Viper visual performance evaluation approach was developed, enabling the evaluation of noise detection performance in the precision-recall space. It was compared to standard evaluation practice and proved to be very useful, since it provides the means for a side-by-side performance comparison in terms of precision, recall and the F-score, thereby enabling intuitive and straightforward interpretation of noise detection results.

All the developed approaches were implemented in a way that allows public accessibility and repeatability of the presented experiments. The open source web-based data mining platform ClowdFlows was extended by implementing the required building blocks to create an environment that enables the construction, execution, sharing and repetition of noise detection experiments. This environment offers to incorporate the functionality of arbitrary noise detection algorithms available as web services, and the construction of custom noise detection ensembles, which enables to exploit the full potential of the NoiseRank methodology. Also components for standard evaluation and visual performance evaluation with the Viper approach were implemented in the ClowdFlows platform, allowing quantitative performance evaluation of noise detection results. Additionally, a separate web-based plat-

form ViperCharts for visual performance evaluation of machine learning, data mining and information retrieval algorithms was implemented, including the Viper evaluation approach along with other standard performance visualization techniques.

The main contributions of the thesis are summarized below.

1. Development of approaches for explicit noise detection enabling exploration of unusual data instances.

   - The NoiseRank ensemble-based noise detection and ranking methodology, enabling construction of custom noise detection ensembles, and exploration of the detected noise, ranked according to the predictive agreement of ensemble members.
   - The ensemble-based noise detection algorithm HARF, which demands high agreement among the ensemble members to denote an instance as noise. The algorithm achieves high precision of noise detection and is used for the identification of only the most unusual data instances.

2. Improvement of quantitative evaluation of noise detection performance on datasets with known or artificially injected noisy instances, by enabling simultaneous interpretation of results obtained from different performance measures:

   - The proposed F-isoline evaluation method and the ε-proximity evaluation method, which trade off the precision and recall of noise detection.
   - The Viper methodology for visual performance evaluation of noise detection. Combining standard and newly proposed evaluation measures in the precision-recall space offers a visualization for straightforward comparison of performance results and a multifold perspective for improved performance evaluation.

3. Application of ensemble-based noise filtering to real-life use cases, thus providing evidence of the practical applicability of the developed approaches for explicit noise detection on the following real-life case studies:

   - Detection of false diagnosed and outlier patients in a medical domain.
   - Identification of atypical documents in a newspaper articles corpus.
   - Detection of domain outlier or borderline documents for supporting cross-domain link discovery.

4. Implementation of the developed approaches for explicit noise detection and visual performance evaluation, which provides public accessibility of the proposed approaches, and the repeatability of the described experiments.

   - Development of an environment, incorporating all the building blocks for testing, execution, and development of noise detection algorithms, construction of custom ensembles, application of the NoiseRank approach, and visual performance evaluation with the Viper approach.
   - The web-based ViperCharts platform for visual performance evaluation, which offers to compare the performance of noise detection and noise ranking algorithms, as well as general classification and information retrieval algorithms, in terms of various visual evaluation methods.

The presented work covered a range of research challenges concerning the handling of noisy and outlier instances in data, and proposed several solutions and improvements over the existing approaches. Nevertheless, the work also revealed possibilities for improvements and directions for further research.

Since the developed NoiseRank methodology enables the use of arbitrary noise detection algorithms in the noise ranking ensemble, it offers numerous possibilities for further evaluation of different ensemble configurations on different datasets in various application domains. The presented methodology includes a simple combination rule for aggregating noise detection predictions, exploring other mainly algebraic ensemble combination rules in the case of continuous noise detection outputs would also present an interesting research direction. Exploring the relation between different features of the ensembles and their noise detection performance would also be interesting. Examining ensemble diversity measures may be a good starting point for this kind of further research. Moreover, as ensemble-based noise detection approaches proved to obtain the most prominent noisy and outlier instances, further applications to other real-life problems may provide new insightful results.

The developed approaches were intentionally implemented in the open source ClowdFlows platform to offer various possibilities for further work on the development, evaluation and implementation of new noise detection algorithms and their ensembles. The platform can also serve as an experimental environment for practical application of the developed algorithms on new real-life domains.

In terms of performance evaluation, the developed ViperCharts platform is ongoing work for the collection, comparison, development and improvement of visual performance evaluation techniques. Since the main aim of ViperCharts is to offer a platform for comparison of different performance evaluation techniques and measures, in further work we will focus on covering even a wider range of performance visualizations used by machine learning practitioners, as well as including more support for visualizing results of statistical significance tests. We will continue the development on the support for different input data formats and on the relations and conversions between different performance visualizations. We also plan to extend our API to enable integration of the performance visualizations into several existing data mining environments.

8 Acknowledgements

This dissertation would have not been finished without the help of many individuals, their great amount of patience and support, and the funding bodies that have supported my research throughout the time of my PhD studies.

First of all I would like to thank my supervisor Nada Lavrač for guiding my work, fueling my research with enthusiastic ideas and tirelessly going through all my manuscripts, providing valuable suggestions for improvements of my research work as well as this thesis.

Many thanks to the members of my PhD committee: Dunja Mladenić, Sašo Džeroski and Johannes Fürnkranz. I am very thankful for their relevant comments, which increased the quality of the thesis.

I wish to thank the funding bodies that financial supported my studies: the Slovenian research agency, the Jožef Stefan International Postgraduate School, the Department of Knowledge Technologies at the Jožef Stefan Institute, and the European Commission for funding the research projects BISON, FIRST and FOC on which I collaborated.

I am also very grateful to Dragan Gamberger for his inspiring ideas for my research on this thesis, his guidance in the beginning of my research path, and the collaboration in our joint research. Furthermore, I am thankful for very fruitful collaboration on various research papers to Bojan Cestnik, Matjaž Juršič and Senja Pollak, as well as the domain experts Goran Krstačić and Roel Coesemans for their elaborate qualitative evaluation of results.

Thanks to all the colleagues at the Department of Knowledge Technologies at the Jožef Stefan Institute for creating a great working environment. For help with many theoretical and engineering problems I would like to thank Dragi Kocev, Janez Kranjc, Anže Vavpetič and Jan Kralj.

Special thanks go to my office mates Vid Podpečan, Matjaž Juršič and Miha Grčar for their selfless supply of inspiring ideas, programing knowledge, and their valuable time for countless very productive and insightful discussions that greatly motivated my research work.

Finally, I am sincerely thankful to my family for their general support in any of my endeavors, and to my friends for their understanding of my absence. Most importantly, I am grateful to my dearest Melita for her love, encouragement and patience, which in the end contributed the most to the finalization of my thesis.

9 References

Achtert, E.; Hettab, A.; Kriegel, H.-P.; Schubert, E.; Zimek, A. Spatial outlier detection: Data, algorithms, visualizations. In: Pfoser, D.; Tao, Y.; Mouratidis, K.; Nascimento, M. A.; Mokbel, M. F.; Shekhar, S.; Huang, Y. (eds.) *Proceedings of Advances in Spatial and Temporal Databases - 12th International Symposium, SSTD 2011, Minneapolis, MN, USA, August 24-26, 2011*. Lecture Notes in Computer Science **6849**, 512–516 (Springer, 2011).

Achtert, E.; Kriegel, H.-P.; Reichert, L.; Schubert, E.; Wojdanowski, R.; Zimek, A. Visual evaluation of outlier detection models. In: Kitagawa, H.; Ishikawa, Y.; Li, Q.; Watanabe, C. (eds.) *Proceedings of Database Systems for Advanced Applications, 15th International Conference, DASFAA 2010, Tsukuba, Japan, April 1-4, 2010, Part II*. Lecture Notes in Computer Science **5982**, 396–399 (Springer, 2010).

Aggarwal, C. C. Mining text streams. In: Aggarwal, C. C.; Zhai, C. (eds.) *Mining Text Data*. 297–321 (Springer, 2012).

Aggarwal, C. C. *Outlier Analysis* (Springer, New York, USA, 2013).

Aggarwal, C. C.; Yu, P. S. Outlier detection for high dimensional data. In: Sellis, T. (ed.) *Proceedings of the 2001 ACM SIGMOD International Conference on Management of Data*. 37–46 (2001).

Angiulli, F.; Basta, S.; Pizzuti, C. Distance-based detection and prediction of outliers. *IEEE Transactions on Knowledge and Data Engineering* **18**, 145–160 (2006).

Angiulli, F.; Fassetti, F. Detecting distance-based outliers in streams of data. In: Silva, M. J.; Laender, A. H. F.; Baeza-Yates, R. A.; McGuinness, D. L.; Olstad, B.; Olsen, Ø. H.; Falcão, A. O. (eds.) *Proceedings of the Sixteenth ACM Conference on Information and Knowledge Management, CIKM 2007, Lisbon, Portugal, November 6-10, 2007*. 811–820 (ACM, 2007).

Assent, I.; Kranen, P.; Baldauf, C.; Seidl, T. Anyout: Anytime outlier detection on streaming data. In: Lee, S.; Peng, Z.; Zhou, X.; Moon, Y.-S.; Unland, R.; Yoo, J. (eds.) *Proceedings of Database Systems for Advanced Applications - 17th International Conference, DASFAA 2012, Busan, South Korea, April 15-19, 2012, Part I*. Lecture Notes in Computer Science **7238**, 228–242 (Springer, 2012).

Bache, K.; Lichman, M. *UCI machine learning repository* (2013). `http://archive.ics.uci.edu/ml`.

Barbará, D.; Chen, P. Using the fractal dimension to cluster datasets. In: *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*. KDD '00 260–264, KDD '00 (ACM, New York, NY, USA, 2000).

Barnett, V.; Lewis, T. *Outliers in Statistical Data. Wiley Series in Probability & Statistics* (Wiley, 1994).

Ben-Gal, I. Outlier detection. In: Maimon, O.; Rokach, L. (eds.) *Data Mining and Knowledge Discovery Handbook, 2nd ed.* 117–130 (Springer, Heidelberg, Germany, 2010).

Berthold, M. R.; Cebron, N.; Dill, F.; Gabriel, T. R.; Kötter, T.; Meinl, T.; Ohl, P.; Thiel, K.; Wiswedel, B. KNIME - the Konstanz information miner: version 2.0 and beyond. *SIGKDD Explor. Newsl.* **11**, 26–31 (2009).

Breiman, L. Bagging predictors. *Machine Learning* **24**, 123–140 (1996).

Breiman, L. Random forests. *Machine Learning* **45**, 5–32 (2001).

Breunig, M. M.; Kriegel, H.-P.; Ng, R. T.; Sander, J. LOF: Identifying density-based local outliers. In: *Proceedings of the 2000 ACM SIGMOD international conference on Management of data.* SIGMOD '00 93–104, SIGMOD '00 (ACM, New York, NY, USA, 2000a).

Breunig, M. M.; Kriegel, H.-P.; Ng, R. T.; Sander, J. Lof: identifying density-based local outliers. *SIGMOD Rec.* **29**, 93–104 (2000b).

Brodley, C. E.; Friedl, M. A. Identifying mislabeled training data. *Journal of Artificial Intelligence Research* **11**, 131–167 (1999).

Brown, G.; Wyatt, J. L.; Harris, R.; Yao, X. Diversity creation methods: a survey and categorisation. *Information Fusion* **6**, 5–20 (2005).

Cai, X.; Zhang, R.; Gao, D.; Li, W. Simultaneous clustering and noise detection for theme-based summarization. In: *Fifth International Joint Conference on Natural Language Processing, IJCNLP 2011, Chiang Mai, Thailand, November 8-13, 2011.* 491–499 (The Association for Computer Linguistics, 2011).

Cao, H.; Zhou, Y.; Shou, L.; Chen, G. Attribute outlier detection over data streams. In: *Proceedings of the 15th international conference on Database Systems for Advanced Applications - Volume Part II.* DASFAA'10 216–230, DASFAA'10 (Springer-Verlag, Berlin, Heidelberg, 2010).

Chandola, V.; Banerjee, A.; Kumar, V. Anomaly detection: A survey. *ACM Comput. Surv.* **41** (2009).

Dang, X. H.; Micenková, B.; Assent, I.; Ng, R. T. Local outlier detection with interpretation. In: Blockeel, H.; Kersting, K.; Nijssen, S.; Železný, F. (eds.) *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2013, Prague, Czech Republic, September 23-27, 2013, Proceedings, Part III.* Lecture Notes in Computer Science **8190**, 304–320 (Springer, 2013).

Dave, R. N. Characterization and detection of noise in clustering. *Pattern Recognition Letters* **12**, 657 – 664 (1991).

Davis, J.; Goadrich, M. The relationship between Precision-Recall and ROC curves. In: *Proceedings of the 23rd ICML conference.* 233–240 (ACM, New York, NY, USA, 2006).

Deanfield, J.; Shea, M.; Ribiero, P.; de Landsheere, C.; Wilson, R.; Horlock, P.; Selwyn, A. Transient st-segment depression as a marker of myocardial ischemia during daily life. *The American Journal of Cardiology* **54**, 1195–1200 (1984).

Demšar, J. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research* **7**, 1–30 (2006).

Demšar, J.; Curk, T.; Erjavec, A.; Črt Gorup; Hočevar, T.; Milutinovič, M.; Možina, M.; Polajnar, M.; Toplak, M.; Starič, A.; Štajdohar, M.; Umek, L.; Žagar, L.; Žbontar, J.; Žitnik, M.; Zupan, B. Orange: Data mining toolbox in python. *Journal of Machine Learning Research* **14**, 2349–2353 (2013).

Dietterich, T. G. Ensemble methods in machine learning. In: Kittler, J.; Roli, F. (eds.) *Multiple Classifier Systems*. Lecture Notes in Computer Science **1857**, 1–15 (Springer, 2000).

Drummond, C.; Holte, R. C. Cost curves: An improved method for visualizing classifier performance. *Machine Learning* **65**, 95–130 (2006).

Dubitzky, W.; Kötter, T.; Schmidt, O.; Berthold, M. R. Towards creative information exploration based on koestler's concept of bisociation. In: Berthold, M. R. (ed.) *Bisociative Knowledge Discovery*. Lecture Notes in Computer Science **7250**, 11–32 (Springer, Heidelberg, Germany, 2012).

Dunn, O. J. Multiple comparisons among means. *Journal of the American Statistical Association* **56**, 52–64 (1961).

Ester, M.; Kriegel, H.-P.; Sander, J.; Xu, X. A density-based algorithm for discovering clusters in large spatial databases with noise. In: Simoudis, E.; Han, J.; Fayyad, U. M. (eds.) *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96), Portland, Oregon, USA*. 226–231 (AAAI Press, 1996).

Fawcett, T. An introduction to roc analysis. *Pattern Recognition Letters* **27**, 861–874 (2006).

Flach, P. A. Roc analysis. In: Sammut, C.; Webb, G. I. (eds.) *Encyclopedia of Machine Learning*. 869–875 (Springer, 2010).

Fortuna, B.; Grobelnik, M.; Mladenić, D. OntoGen: semi-automatic ontology editor. In: Smith, M.; Salvendy, G. (eds.) *Proceedings of the 12th International Conference on Human-Computer Interaction (HCI 2007)*. 309–318 (2007).

Friedman, M. A comparison of alternative tests of significance for the problem of m rankings. *The Annals of Mathematical Statistic* **11**, 86–92 (1940).

Fürnkranz, J. Pruning algorithms for rule learning. *Machine Learning* **27**, 139–171 (1997).

Fuster, V.; O'Rourke, R. A.; Walsh, R.; Poole-Wilson, P. (eds.) *Hursts The Heart. 2477* (McGraw-Hill, 2007), 12 edition.

Gamberger, D.; Lavrač, N. Conditions for Occam's razor applicability and noise elimination. In: Someren, M. V.; Widmer, G. (eds.) *Lecture Notes in Artificial Intelligence: Machine Learning: ECML-97*. **1224**, 108–123 (Springer-Verlag, 1997).

Gamberger, D.; Lavrač, N. Expert-guided subgroup discovery: Methodology and application. *Journal of Artificial Intelligence Research* **17**, 501–527 (2002).

Gamberger, D.; Lavrač, N.; Džeroski, S. Noise detection and elimination in data preprocessing: Experiments in medical domains. *Applied Artificial Intelligence* **14**, 205–223 (2000).

Gamberger, D.; Lavrač, N.; Grošelj, C. Experiments with noise filtering in a medical domain. In: *Proceedings of 16th International Conference on Machine Learning - ICML*. 143–151 (Morgan Kaufmann, 1999).

Gamberger, D.; Lavrač, N.; Krstačić, G. Active subgroup mining: A case study in a coronary heart disease risk group detection. *Artificial Intelligence in Medicine* **28**, 27–57 (2003).

Garcia, L. P. F.; Carvalho, A. C. P. L. F.; Lorena, A. C. Noisy data set identification. In: Pan, J.-S.; Polycarpou, M. M.; Wozniak, M.; Carvalho, A. C. P. L. F.; Quintián-Pardo, H.; Corchado, E. (eds.) *Proceedings of Hybrid Artificial Intelligent Systems - 8th International Conference, HAIS 2013, Salamanca, Spain, September 11-13, 2013*. Lecture Notes in Computer Science **8073**, 629–638 (Springer, 2013).

Gashler, M.; Giraud-Carrier, C. G.; Martinez, T. R. Decision tree ensemble: Small heterogeneous is better than large homogeneous. In: Wani, M. A.; wen Chen, X.; Casasent, D.; Kurgan, L. A.; Hu, T.; Hafeez, K. (eds.) *Proceedings of the Seventh International Conference on Machine Learning and Applications, ICMLA 2008, San Diego, California, USA, 11-13 December 2008*. 900–905 (IEEE Computer Society, 2008).

Gavriluţ, D.; Ciortuz, L. Dealing with class noise in large training datasets for malware detection. In: Wang, D.; Negru, V.; Ida, T.; Jebelean, T.; Petcu, D.; Watt, S. M.; Zaharie, D. (eds.) *13th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing, SYNASC 2011, Timisoara, Romania, September 26-29, 2011*. 401–407 (IEEE Computer Society, 2011).

Gelfand, S.; Ravishankar, C.; Delp, E. An iterative growing and pruning algorithm for classification tree design. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **13**, 163–174 (1991).

Grubbs, F. E. Procedures for detecting outlying observations in samples. *Technometrics* **11**, 1–21 (1969).

Grčar, M.; Podpečan, V.; Sluban, B.; Mozetič, I. Ontology querying support in semantic annotation process. In: Anthony, P.; Ishizuka, M.; Lukose, D. (eds.) *Proceedings of the 12th Pacific Rim International Conference on Artificial Intelligence, PRICAI 2012, Kuching, Malaysia, September 3-7, 2012*. Lecture Notes in Computer Science **7458**, 76–87 (Springer, Heidelberg, Germany, 2012).

Hall, M.; Frank, E.; Holmes, G.; Pfahringer, B.; Reutemann, P.; Witten, I. H. The WEKA data mining software: an update. *SIGKDD Explorations Newsletter* **11**, 10–18 (2009).

Hernández-Orallo, J.; Flach, P. A.; Ferri, C. ROC curves in cost space. *Machine Learning* **93**, 71–91 (2013).

Hido, S.; Tsuboi, Y.; Kashima, H.; Sugiyama, M.; Kanamori, T. Statistical outlier detection using direct density ratio estimation. *Knowledge and Information Systems* **26**, 309–336 (2011).

Hodge, V.; Austin, J. A survey of outlier detection methodologies. *Artificial Intelligence Review* **22**, 85–126 (2004).

Juršič, M. *Text Mining for Cross-Domain Knowledge Discovery*. Ph.D. thesis, Jožef Stefan International Postgraduate School, Ljubljana, Slovenia (2013).

Juršič, M.; Mozetič, I.; Erjavec, T.; Lavrač, N. Lemmagen: Multilingual lemmatisation with induced ripple-down rules. *Journal of Universal Computer Science* **16**, 1190–1214 (2010).

Juršič, M.; Sluban, B.; Cestnik, B.; Grčar, M.; Lavrač, N. Bridging concept identification for constructing information networks from text documents. In: Berthold, M. R. (ed.) *Bisociative Knowledge Discovery*. Lecture Notes in Computer Science **7250**, 66–90 (Springer, Berlin Heidelberg, Germany, 2012).

Kandel, S.; Paepcke, A.; Hellerstein, J.; Heer, J. Wrangler: interactive visual specification of data transformation scripts. In: Tan, D. S.; Amershi, S.; Begole, B.; Kellogg, W. A.; Tungare, M. (eds.) *Proceedings of the International Conference on Human Factors in Computing Systems, CHI 2011, Vancouver, BC, Canada, May 7-12, 2011*. 3363–3372 (ACM, 2011).

Khoshgoftaar, T.; Seliya, N.; Gao, K. Rule-based noise detection for software measurement data. In: *Information Reuse and Integration, 2004. IRI 2004. Proceedings of the 2004 IEEE International Conference on*. 302 – 307 (2004).

Khoshgoftaar, T. M.; Joshi, V. H.; Seliya, N. Detecting noisy instances with the ensemble filter: a study in software quality estimation. *International Journal of Software Engineering and Knowledge Engineering* **16**, 53–76 (2006).

Khoshgoftaar, T. M.; Rebours, P. Generating multiple noise elimination filters with the ensemble-partitioning filter. In: *Proceedings of the 2004 IEEE International Conference on Information Reuse and Integration*. 369–375 (IEEE Systems, Man, and Cybernetics Society, 2004).

Khoshgoftaar, T. M.; Zhong, S.; Joshi, V. Enhancing software quality estimation using ensemble-classifier based noise filtering. *Intelligent Data Analysis* **9**, 3–27 (2005).

Knorr, E. M.; Ng, R. T. Algorithms for mining distance-based outliers in large datasets. In: *Proceedings of the 24rd International Conference on Very Large Data Bases*. VLDB '98 392–403, VLDB '98 (Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1998).

Koestler, A. *The Act of Creation* (MacMillan Company, New York, 1964).

Kranjc, J.; Podpečan, V.; Lavrač, N. Clowdflows: A cloud based scientific workflow platform. In: Flach, P.; Bie, T.; Cristianini, N. (eds.) *Machine Learning and Knowledge Discovery in Databases*. Lecture Notes in Computer Science **7524**, 816–819 (Springer Berlin Heidelberg, 2012).

Kriegel, H.-P.; Kröger, P.; Schubert, E.; Zimek, A. Loop: local outlier probabilities. In: Cheung, D. W.-L.; Song, I.-Y.; Chu, W. W.; Hu, X.; Lin, J. J. (eds.) *Proceedings of the 18th ACM Conference on Information and Knowledge Management, CIKM 2009, Hong Kong, China, November 2-6, 2009*. 1649–1652 (ACM, 2009).

Kuncheva, L. I. *Combining Pattern Classifiers: Methods and Algorithms* (Wiley-Interscience, New Jersey, USA, 2004).

Kuncheva, L. I.; Whitaker, C. J. Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine Learning* **51**, 181–207 (2003).

Lam, L.; Suen, S. Y. Application of majority voting to pattern recognition: an analysis of its behavior and performance. *Trans. Sys. Man Cyber. Part A* **27**, 553–568 (1997).

Lavrač, N.; Džeroski, S. *Inductive Logic Programming: Techniques and Applications* (Ellis Horwood, New York, USA, 1994).

Lavrač, N.; Sluban, B.; Juršič, M. Cross-domain literature mining through outlier document and bridging concept detection. In: *Proceeding of the Workshop on Analysis of Complex NEtworks at ECML PKDD 2010, ACNE 2010*. 35–47 (ACNE Committee, Barcelona, Spain, 2010).

Libralon, G. L.; Carvalho, A. C. P. L. F.; Lorena, A. C. Ensembles of pre-processing techniques for noise detection in gene expression data. In: *Proceedings of the 15th international conference on Advances in neuro-information processing - Volume Part I*. ICONIP'08 486–493, ICONIP'08 (Springer-Verlag, Berlin, Heidelberg, 2009).

Lin, X.; Yacoub, S.; Burns, J.; Simske, S. Performance analysis of pattern classifier combination by plurality voting. *Pattern Recognition Letters* **24**, 1959–1969 (2003).

Loureiro, A.; Torgo, L.; Soares, C. Outlier detection using clustering methods: a data cleaning application. In: *Proceedings of the Data Mining for Business Workshop* (2004).

Macedoni-Lukšič, M.; Petrič, I.; Cestnik, B.; Urbančič, T. Developing a deeper understanding of autism: Connecting knowledge through literature mining. *Autism Research and Treatment* **2011**, 1–8 (2011).

Manning, C. D.; Raghavan, P.; Schtze, H. *Introduction to Information Retrieval* (Cambridge University Press, New York, NY, USA, 2008).

Marascu, A.; Masseglia, F. Parameterless outlier detection in data streams. In: Shin, S. Y.; Ossowski, S. (eds.) *Proceedings of the 2009 ACM Symposium on Applied Computing (SAC), Honolulu, Hawaii, USA, March 9-12, 2009*. 1491–1495 (ACM, 2009).

Maron, D.; Ridker, P.; Pearson, A. Risk factors and the prevention of coronary heart disease. In: Wayne, A.; Schlant, R.; Fuster, V. (eds.) *HURST'S: The Heart*. 1175–1195 (1998).

Mednick, S. A. The associative basis of the creative process. *Psychological Review* **69**, 219–227 (1962).

Mierswa, I.; Wurst, M.; Klinkenberg, R.; Scholz, M.; Euler, T. YALE: Rapid prototyping for complex data mining tasks. In: Eliassi-Rad, T.; Ungar, L. H.; Craven, M.; Gunopulos, D. (eds.) *Proceedings of the Twelfth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 935–940 (ACM, New York, NY, USA, 2006).

Mingers, J. An empirical comparison of pruning methods for decision tree induction. *Machine Learning* **4**, 227–243 (1989).

Miranda, A.; Garcia, L.; Carvalho, A.; Lorena, A. Use of classification algorithms in noise detection and elimination. In: *Hybrid Artificial Intelligence Systems*. Lecture Notes in Computer Science **5572**, 417–424 (Springer Berlin / Heidelberg, 2009).

Mitchell, T. M. *Machine Learning* (McGraw-Hill, Inc., New York, NY, USA, 1997), 1st edition.

Moonesinghe, H. D. K.; Tan, P.-N. Outrank: a graph-based outlier detection framework using random walk. *International Journal on Artificial Intelligence Tools* **17**, 19–36 (2008).

Mu, X.; Watta, P.; Hassoun, M. H. Analysis of a plurality voting-based combination of classifiers. *Neural Process. Lett.* **29**, 89–107 (2009).

Müller, E.; Schiffer, M.; Gerwert, P.; Hannen, M.; Jansen, T.; Seidl, T. *SOREX*: Subspace outlier ranking exploration toolkit. In: Balcázar, J. L.; Bonchi, F.; Gionis, A.; Sebag, M. (eds.) *Machine Learning and Knowledge Discovery in Databases, European Conference, ECML PKDD 2010*. Lecture Notes in Computer Science **6323**, 607–610 (Springer, 2010).

Müller, E.; Schiffer, M.; Seidl, T. Statistical selection of relevant subspace projections for outlier ranking. In: Abiteboul, S.; Böhm, K.; Koch, C.; Tan, K.-L. (eds.) *Proceedings of the 27th International Conference on Data Engineering, ICDE 2011, April 11-16, 2011, Hannover, Germany*. 434–445 (IEEE Computer Society, Washington, D.C., USA, 2011).

Nemenyi, P. B. *Distribution-free multiple comparisons*. Ph.D. thesis, Princeton University (1963).

Ng, R. T.; Han, J. Efficient and effective clustering methods for spatial data mining. In: *Proceedings of the 20th International Conference on Very Large Data Bases*. VLDB '94 144–155, VLDB '94 (Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1994).

Niblett, T.; Bratko, I. Learning decision rules in noisy domains. In: Bramer, M. (ed.) *Research and Development in Expert Systems* (Cambridge University Press, 1987).

Papadimitriou, S.; Kitagawa, H.; Gibbons, P. B.; Faloutsos, C. LOCI: Fast Outlier Detection Using the Local Correlation Integral. In: *Proceedings of the 19th International Conference on Data Engineering*. 315–326 (IEEE, 2003).

Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; Vanderplas, J.; Passos, A.; Cournapeau, D.; Brucher, M.; Perrot, M.; Duchesnay, E. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research* **12**, 2825–2830 (2011).

Petrič, I.; Cestnik, B.; Lavrac, N.; Urbancic, T. Bisociative knowledge discovery by literature outlier detection. In: Berthold, M. R. (ed.) *Bisociative Knowledge Discovery*. Lecture Notes in Computer Science **7250**, 313–324 (Springer, 2012).

Petrič, I.; Cestnik, B.; Lavrač, N.; Urbančič, T. Outlier detection in cross-context link discovery for creative literature mining. *The Computer Journal* **55**, 47–61 (2010).

Petrič, I.; Urbančič, T.; Cestnik, B. Literature mining: Potential for gaining hidden knowledge from biomedical articles. In: Bohanec, M.; et al. (eds.) *Proceedings of the 9th International Multiconference Information Society*. 52–55 (2006).

Petrič, I.; Urbančič, T.; Cestnik, B. Discovering hidden knowledge from biomedical literature. *Informatica* **31**, 15–20 (2007).

Petrič, I.; Urbančič, T.; Cestnik, B.; Macedoni-Lukšič, M. Literature mining method RaJoLink for uncovering relations between biomedical concepts. *Journal of Biomedical Informatics* **42**, 220–232 (2009).

Pokrajac, D.; Lazarevic, A.; Latecki, L. J. Incremental local outlier detection for data streams. In: *Proceedings of the IEEE Symposium on Computational Intelligence and Data Mining, CIDM 2007, part of the IEEE Symposium Series on Computational Intelligence 2007, Honolulu, Hawaii, USA, 1-5 April 2007*. 504–515 (IEEE, 2007).

Polikar, R. Ensemble learning. *Scholarpedia* **4**, 2776 (2009).

Pollak, S. Text classification of articles on kenyan elections. In: Vetulani, Z. (ed.) *Proceedings of the 4th Language & Technology Conference: Human language technologies as a challenge for computer science and linguistics*. 229–233 (2009).

Pollak, S.; Coesemans, R.; Daelemans, W.; Lavrač, N. Detecting contrast patterns in newspaper articles by combining discourse analysis and text mining. *Pragmatics: Quarterly Publication of the International Pragmatics Association* **21**, 674–683 (2011).

Quinlan, J. R. Simplifying decision trees. *International Journal of Man-Machine Studies* **27**, 221–234 (1987).

Ramaswamy, S.; Rastogi, R.; Shim, K. Efficient algorithms for mining outliers from large data sets. In: Chen, W.; Naughton, J. F.; Bernstein, P. A. (eds.) *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, May 16-18, 2000, Dallas, Texas, USA*. 427–438 (ACM, 2000).

Rehm, F.; Klawonn, F.; Kruse, R. A novel approach to noise clustering for outlier detection. *Soft Computing* **11**, 489–494 (2007).

Ren, D.; Wang, B.; Perrizo, W. RDF: A Density-Based Outlier Detection Method using Vertical Data Representation. In: *Proceedings of the 4th IEEE International Conference on Data Mining (ICDM 2004), 1-4 November 2004, Brighton, UK*. 503–506 (IEEE Computer Society, Los Alamitos, CA, USA, 2004).

Rosner, B. Percentage points for a generalized esd many-outlier procedure. *Technometrics* **25**, 165–172 (1983).

Ruta, D.; Gabrys, B. Classifier selection for majority voting. *Information Fusion* **6**, 63–81 (2005). Diversity in Multiple Classifier Systems.

Schapire, R. E. The strength of weak learnability. *Machine Learning* **5**, 197–227 (1990).

Schubert, E.; Zimek, A.; Kriegel, H.-P. Local outlier detection reconsidered: a generalized view on locality with applications to spatial, video, and network outlier detection. *Data Mining and Knowledge Discovery* 1–48 (2012).

Sluban, B.; Gamberger, D.; Lavrač, N. Advances in class noise detection. In: Coelho, H.; Studer, R.; Wooldridge, M. (eds.) *Proceedings of the 19th European Conference on Artificial Intelligence, ECAI 2010, Lisbon, Portugal, August 16-20, 2010*. Frontiers in Artificial Intelligence and Applications **215**, 1105–1106 (IOS Press, Amsterdam, Netherlands, 2010a).

Sluban, B.; Gamberger, D.; Lavrač, N. Performance analysis of class noise detection algorithms. In: Ågotnes, T. (ed.) *Proceedings of the Fifth Starting AI Researchers' Symposium, STAIRS 2010, Lisbon, Portugal, 16-20 August, 2010*. Frontiers in Artificial Intelligence and Applications **222**, 303–314 (IOS Press, Amsterdam, Netherlands, 2010b).

Sluban, B.; Grčar, M. URL Tree: Efficient Unsupervised Content Extraction from Streams of Web Documents. In: He, Q.; Iyengar, A.; Nejdl, W.; Pei, J.; Rastogi, R. (eds.) *Proceedings of the 22nd ACM International Conference on Information and Knowledge Management, CIKM'13, San Francisco, CA, USA, October 27 - November 1, 2013*. 2267–2272 (ACM, New York, NY, USA, 2013).

Sluban, B.; Juršič, M.; Cestnik, B.; Lavrač, N. Evaluating outliers for cross-context link discovery. In: Peleg, M.; Lavrač, N.; Combi, C. (eds.) *Proceedings of the 13th Conference on Artificial Intelligence in Medicine, AIME 2011, Bled, Slovenia, July 2-6, 2011*. Lecture Notes in Computer Science **6747**, 343–347 (Springer, Heidelberg, Germany, 2011).

Sluban, B.; Juršič, M.; Cestnik, B.; Lavrač, N. Exploring the power of outliers for cross-domain literature mining. In: Berthold, M. R. (ed.) *Bisociative Knowledge Discovery*. Lecture Notes in Computer Science **7250**, 325–337 (Springer, Berlin Heidelberg, Germany, 2012a).

Sluban, B.; Lavrač, N. Supporting the search for cross-context links by outlier detection methods. *BMC Bioinformatics* **11 (Suppl 5)**, P2 (2010).

Sluban, B.; Lavrač, N. ViperCharts: Visual performance evaluation platform. In: Blockeel, H.; Kersting, K.; Nijssen, S.; Zelezný, F. (eds.) *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases, ECML PKDD 2013, Prague, Czech Republic, September 23-27, 2013*. Lecture Notes in Computer Science **8190**, 650–653 (Springer, Heidelberg, Germany, 2013).

Sluban, B.; Lavrač, N.; Gamberger, D. Ensemble-based noise detection: noise ranking and visual performance evaluation. *Data Mining and Knowledge Discovery* **28**, 265–303 (2014). [IF=2.877].

Sluban, B.; Lavrač, N.; Gamberger, D.; Bauer, A. Experiments with saturation filtering for noise elimination from labeled data. In: *Proceeding of the 12th Multi-conference Information Society, Conference on Data Mining and Data Warehouses, SiKDD 2009*. 240–243 (IJS, Ljubljana, Slovenia, 2009).

Sluban, B.; Pollak, S.; Coesemans, R.; Lavrač, N. Irregularity detection in categorized document corpora. In: Calzolari, N.; Choukri, K.; Declerck, T.; Dogan, M. U.; Maegaard, B.; Mariani, J.; Odijk, J.; Piperidis, S. (eds.) *Proceedings of the Eighth International Conference on Language Resources and Evaluation, LREC-2012, Istanbul, Turkey, May 23-25, 2012*. 1598–1603 (European Language Resources Association (ELRA), Paris, France, 2012b).

Smalheiser, N. R.; Swanson, D. R. Using ARROWSMITH: a computer-assisted approach to formulating and assessing scientific hypotheses. *Comput Methods Programs Biomed* **57**, 149–153 (1998).

Srinivasan, P. Text mining: Generating hypotheses from MEDLINE. *Journal of the American Society for Information Science and Technology* **55**, 396–413 (2004).

Swanson, D. R. Undiscovered public knowledge. *Library Quarterly* **56**, 103–118 (1986).

Swanson, D. R. Migraine and magnesium: eleven neglected connections. *Perspectives in Biology and Medicine* **31**, 526–557 (1988).

Swanson, D. R. Medical literature as a potential source of new knowledge. *Bulletin of the Medical Library Association* **78**, 29–37 (1990).

Swanson, D. R.; Smalheiser, N. R.; Torvik, V. I. Ranking indirect connections in literature-based discovery: The role of medical subject headings (mesh). *Journal of the American Society for Information Science and Technology* **57**, 1427–1439 (2006).

Tang, J.; Chen, Z.; Fu, A. W.-C.; Cheung, D. W.-L. Enhancing effectiveness of outlier detections for low density patterns. In: *Proceedings of the 6th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining*. PAKDD '02 535–548, PAKDD '02 (Springer-Verlag, London, UK, UK, 2002).

Teng, C. M. Correcting noisy data. In: *Proceedings of the Sixteenth International Conference on Machine Learning*. 239–248 (1999).

Urbančič, T.; Petrič, I.; Cestnik, B.; Macedoni-Lukšič, M. Literature mining: towards better understanding of autism. In: Bellazzi, R.; Abu-Hanna, A.; Hunter, J. (eds.) *Proceedings of the 11th Conference on Artificial Intelligence in Medicine in Europe*. 217–226 (2007).

Van Hulse, J. D.; Khoshgoftaar, T. M. Class noise detection using frequent itemsets. *Intelligent Data Analysis* **10**, 487–507 (2006).

Van Hulse, J. D.; Khoshgoftaar, T. M.; Huang, H. The pairwise attribute noise detection algorithm. *Knowledge and Information Systems* **11**, 171–190 (2007).

Verbaeten, S. Identifying mislabeled training examples in ilp classification problems. In: *Proceedings of Twelfth Belgian-Dutch Conference on Machine Learning*. 1–8 (2002).

Verbaeten, S.; Van Assche, A. Ensemble methods for noise elimination in classification problems. In: Windeatt, T.; Roli, F. (eds.) *Multiple Classifier Systems*. Lecture Notes in Computer Science **2709**, 317–325 (Springer Berlin / Heidelberg, 2003).

Wang, B.; Yang, X.-C.; Wang, G.-R.; Yu, G. Outlier detection over sliding windows for probabilistic data streams. *J. Comput. Sci. Technol.* **25**, 389–400 (2010).

Weeber, M.; Vos, R.; Klein, H.; de Jong-van den Berg, L. T. W. Using concepts in literature-based discovery: Simulating swanson's raynaud–fish oil and migraine–magnesium discoveries. *Journal of the American Society for Information Science and Technology* **52**, 548–557 (2001).

Wiswedel, B.; Berthold, M. R. Fuzzy clustering in parallel universes with noise detection. In: *Proceedings of the ICDM 2005 Workshop on Computational Intelligence in Data Mining*. 29–37 (2005).

Witten, I. H.; Frank, E. *Data Mining: Practical Machine Learning Tools and Techniques, Second Edition* (Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2005).

Xu, L.; Krzyzak, A.; Suen, C. Methods of combining multiple classifiers and their applications to handwriting recognition. *Systems, Man and Cybernetics, IEEE Transactions on* **22**, 418–435 (1992).

Yin, H.; Dong, H.; Li, Y. A cluster-based noise detection algorithm. *International Workshop on Database Technology and Applications* **0**, 386–389 (2009).

Zhang, K.; Hutter, M.; Jin, H. A new local distance-based outlier detection approach for scattered real-world data. In: *Proceedings of the 13th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining*. PAKDD '09 813–822, PAKDD '09 (Springer-Verlag, Berlin, Heidelberg, 2009).

Zhong, S.; Tang, W.; Khoshgoftaar, T. M. Boosted noise filters for identifying mislabeled data. *Technical Report* (2005). Dept. of Computer Science and Engineering, Florida Atlantic University.

Zhu, X.; Wu, X. Class noise vs. attribute noise: A quantitative study of their impacts. *Artificial Intelligence Review* **22**, 177–210 (2004).

Zhu, X.; Wu, X.; Chen, Q. Eliminating class noise in large datasets. In: Fawcett, T.; Mishra, N. (eds.) *Proceeding of the International Conference on Machine Learning*. 920–927 (AAAI Press, 2003).

List of Figures

List of Tables

Appendix

A Visual Performance Evaluation Results

This appendix includes the performance results visualized by the VIPER evaluation methodology, presented in Chapter 4, which were moved to this appendix for the conciseness of Section 4.5.1. First, the results of average performance of elementary and ensemble noise detection algorithms on the TTT, KRKP, CHD and NAKE datasets with 2% randomly injected noise are visualized in Figures A.1, A.2, A.3 and A.4, respectively. Second, the performance results for the same experimental settings except for a higher noise level of 10% are presented in Figures A.5, A.6, A.7 and A.8.

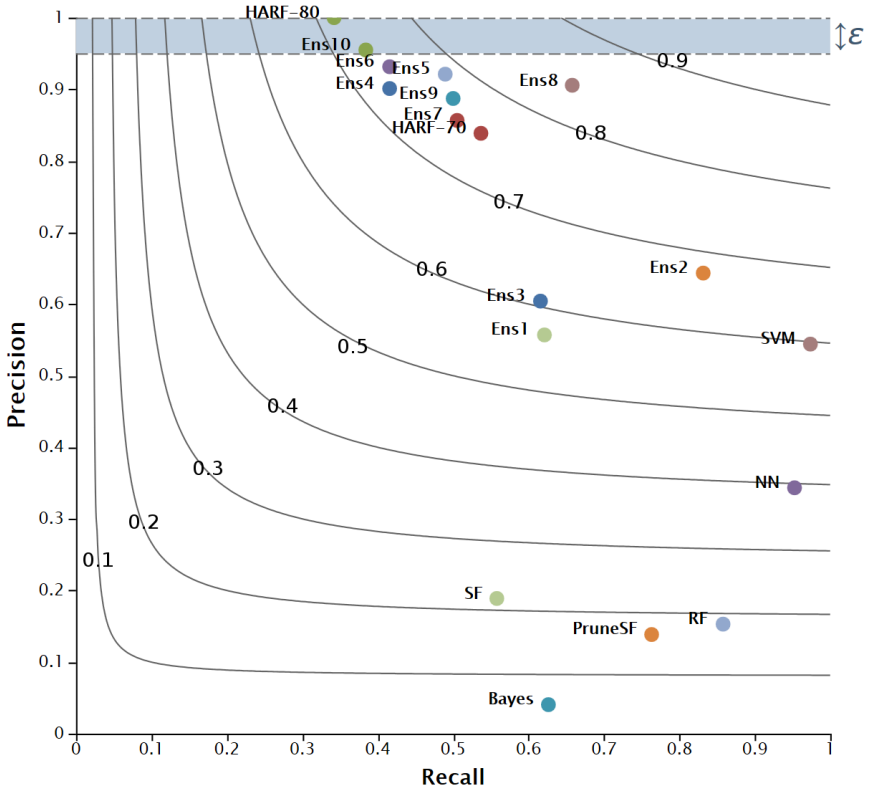


Figure A.1: Performance results of different noise detection algorithms in the precision-recall space on the Tic Tac Toe dataset with 2% injected noise.

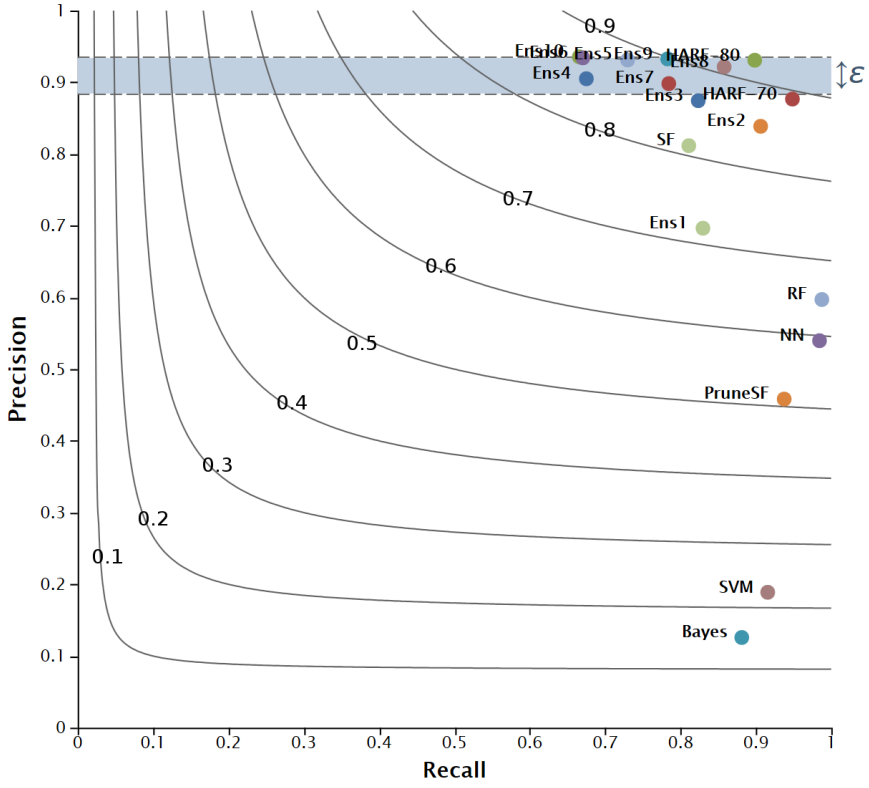


Figure A.2: Performance results of different noise detection algorithms in the precision-recall space on the King-Rook vs. King-Pawn dataset with 2% injected noise.

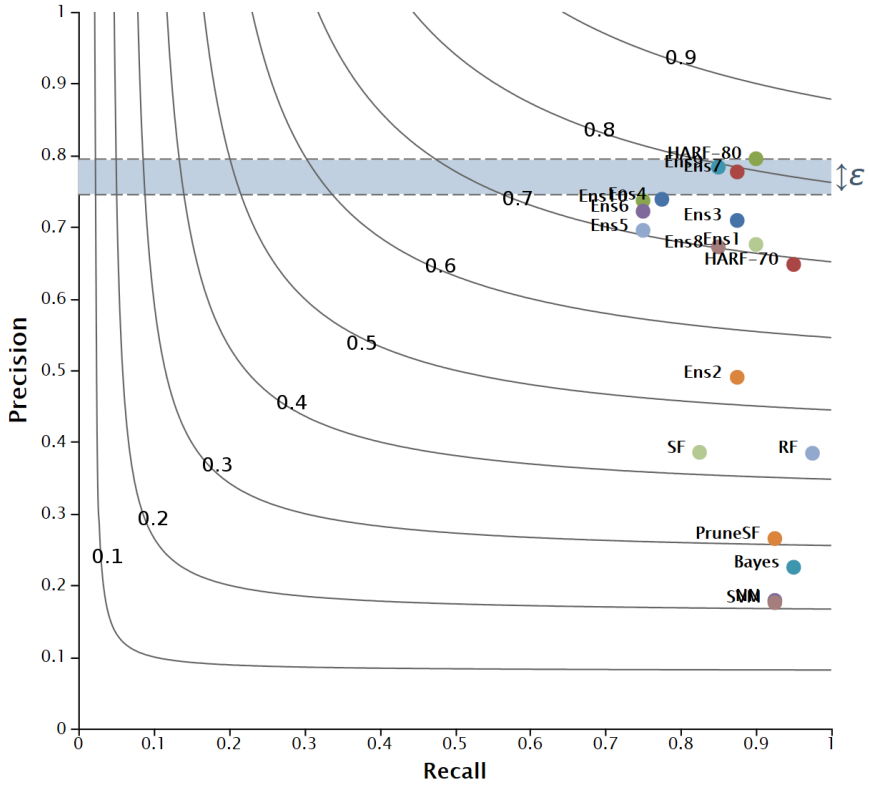


Figure A.3: Performance results of different noise detection algorithms in the precision-recall space on the Coronary Heart Disease dataset with 2% injected noise.

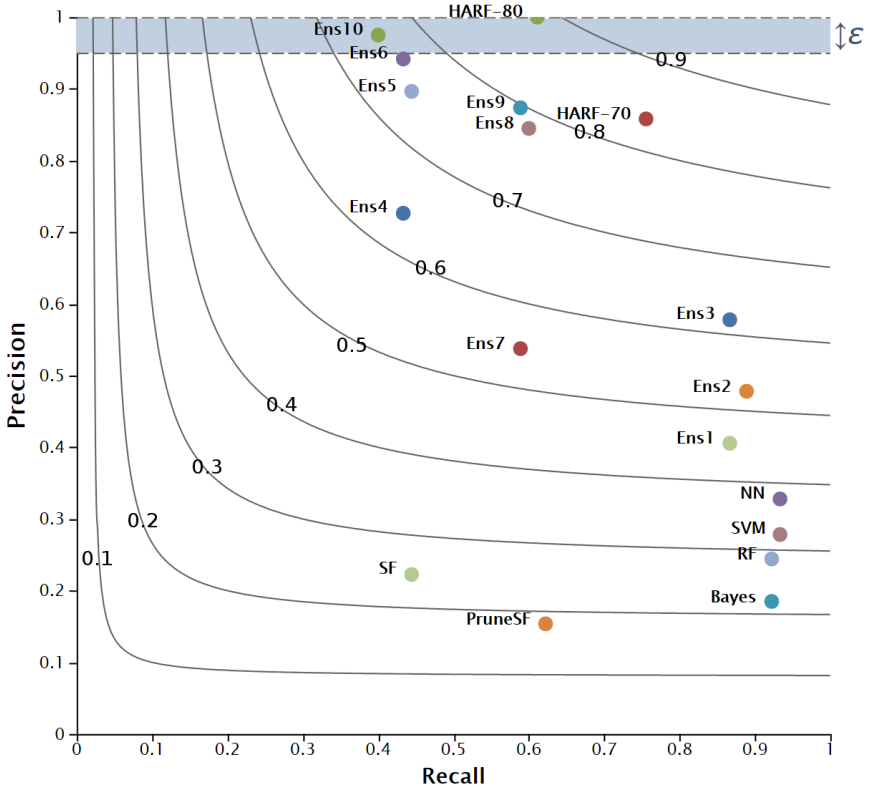


Figure A.4: Performance results of different noise detection algorithms in the precision-recall space on the News Articles on Kenyan Elections dataset with 2% injected noise.

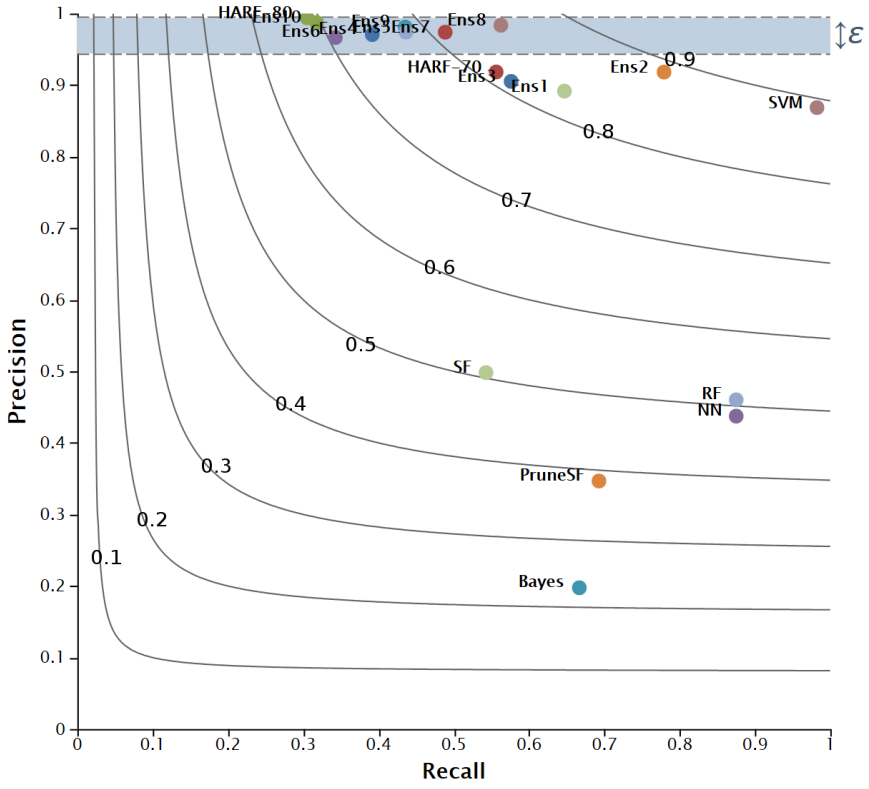


Figure A.5: Performance results of different noise detection algorithms in the precision-recall space on the Tic Tac Toe dataset with 10% injected noise.

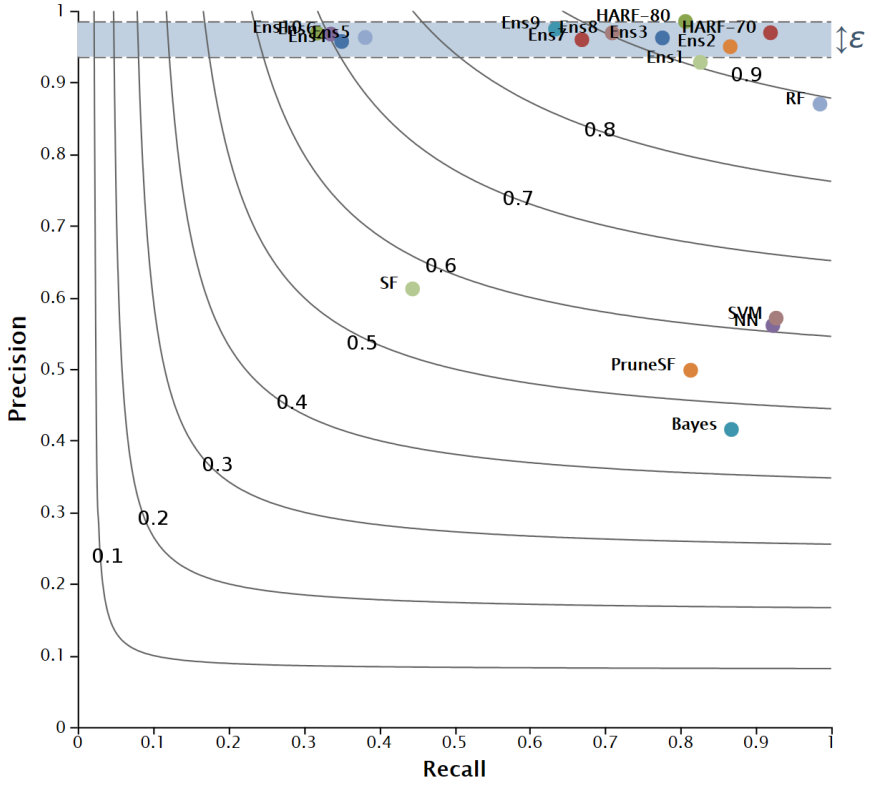


Figure A.6: Performance results of different noise detection algorithms in the precision-recall space on the King-Rook vs. King-Pawn dataset with 10% injected noise.

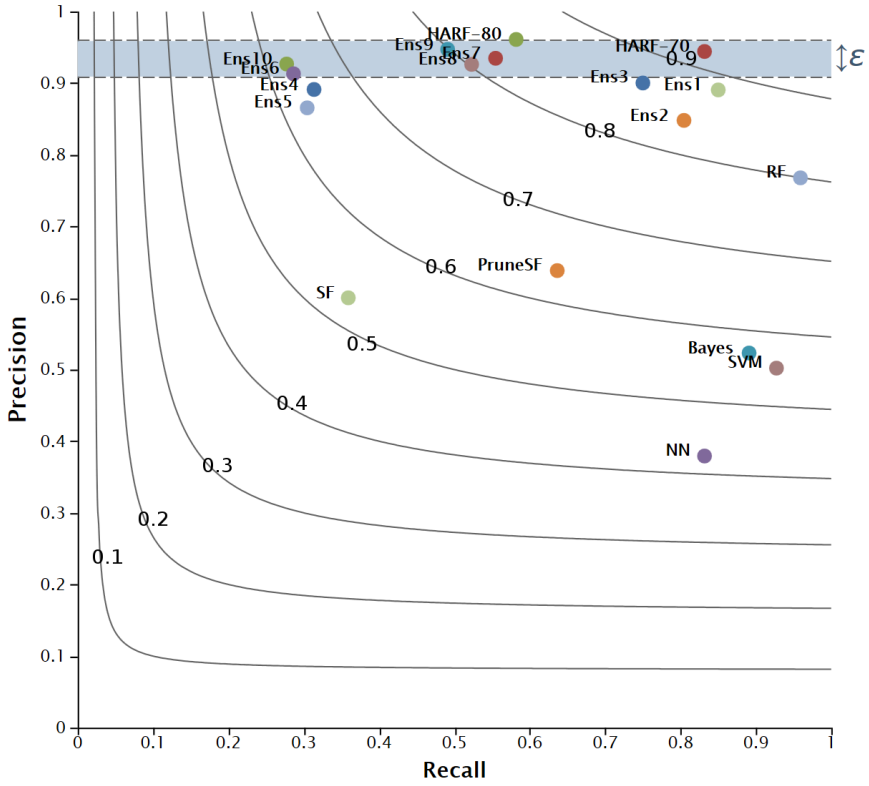


Figure A.7: Performance results of different noise detection algorithms in the precision-recall space on the Coronary Heart Disease dataset with 10% injected noise.

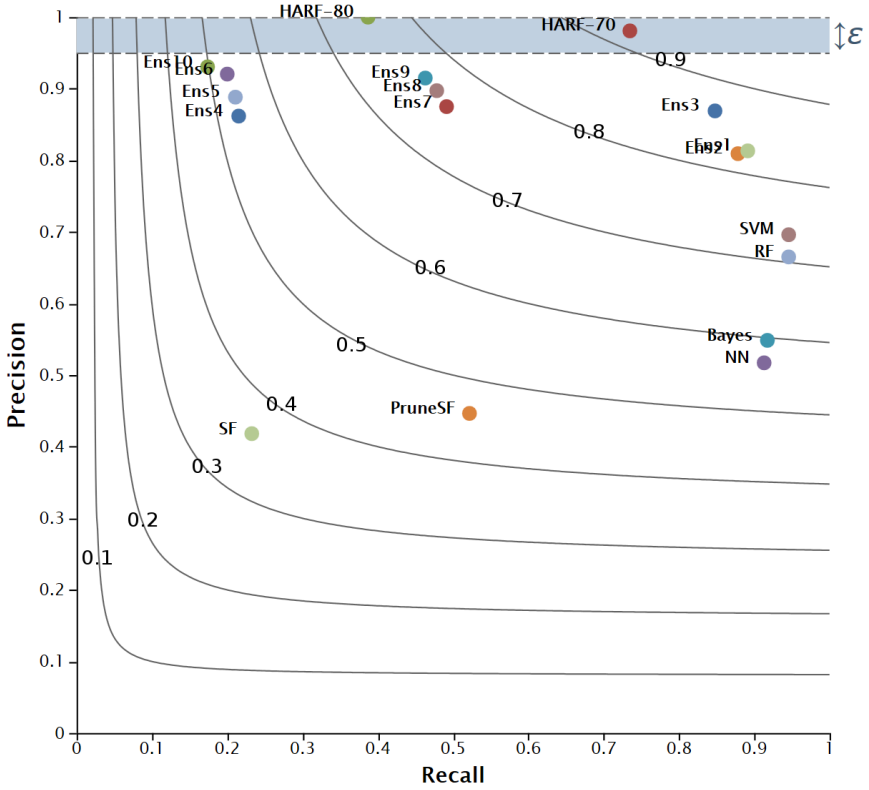


Figure A.8: Performance results of different noise detection algorithms in the precision-recall space on the News Articles on Kenyan Elections dataset with 10% injected noise.

B Noise Handling with ClowdFlows: User Guidelines

This appendix provides the user guidelines for the developed approaches for explicit noise detection and visual performance evaluation, which enable easy public availability and reusability of the developed approaches. For this purpose the developed methodology was implemented in the ClowdFlows cloud-based platform for composition, execution, and sharing of interactive machine learning and data mining workflows (Kranjc et al., 2012). The platform runs in all major web browsers and therefore only an Internet connection is required to access the functionality offered by the platform, as well as the methods and results that can be shared with it.

In this work, the functionality of ClowdFlows was extended by implementing the required building blocks for performing noise detection experiments. This open source platform enables to integrate and join different implementations of algorithms, tools and Web services into a coherent workflow that can be executed in a cloud-based application. The desired experimental process is constructed by the user as a workflow of individual processing components, called widgets. An example of a workflow is presented in Figure B.1.

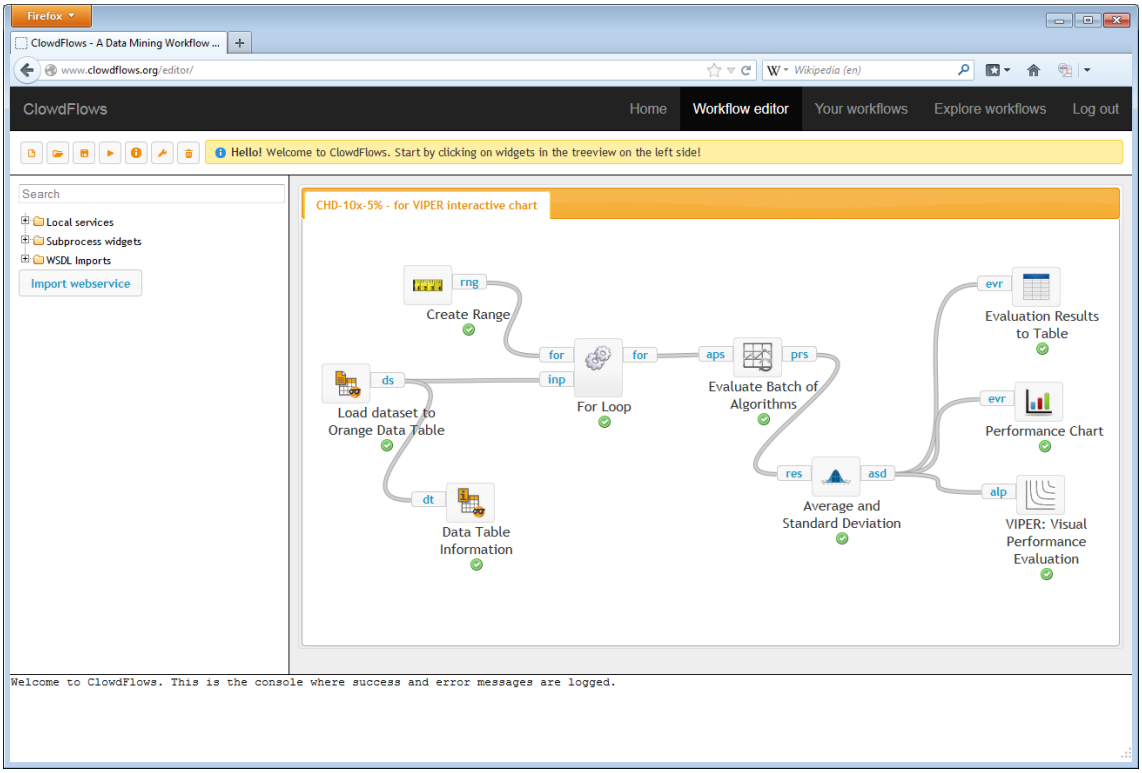


Figure B.1: The workflow editor of the ClowdFlows platform. The workflow management panel at the top, the widget repository on the left, the canvas on the right, and the console at the bottom.

After creating an account, the user can explore existing workflows or go to the *Workflow editor* to construct his own workflows. The Workflow editor view consists of four main parts, as can be observed in Figure B.1. In the top panel control buttons are places, offering basic workflow functionality, like creating new workflows, opening, saving and running workflows, as well as the information about the workflow. On the left hand side, the *widget repository* can be found, which includes different packages with all elementary processing units/widgets that are used for workflow construction. On the right hand side is the *canvas* where the selected widgets can be connected through their inputs and outputs into a workflow of the

desired data mining experiment. The bottom part of the Workflow editor is the *console* where success and error messages of the performed work are logged.

Basic workflow construction

Workflows for noise detection and visual performance evaluation can be easily constructed in only few steps. Let us assume we want to apply NoiseRank to a dataset using three classification noise filters. The following steps describe the construction of this example workflow.

1. The implemented NoiseRank widget can be found in the *Noise Handling* package in the widget repository. To find the widget, the user should expand the local services folder and then the Noise handling package, or use the search functionality at the top of the widget repository to find it. By clicking on the widget's name in the repository, the widget will appear on the canvas.

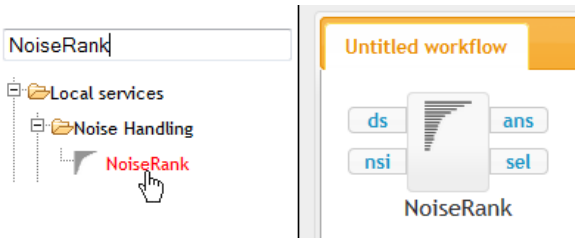


Figure B.2: The widget selected from the widget repository appears on the canvas.

2. The inputs to the NoiseRank widget are the lists of noisy instances detected by different classification filtering algorithms. The implemented widget for classification filter can be found in the *Noise Handling* package under *Noise Filters*. By dragging and dropping, the widgets can be arranged on the canvas as you please. To connect the widgets on the canvas simply click on the output of a Classification filter widget and then click on the "Noisy Instances (nsi)" input of the NoiseRank widget, which will establish a connection. To remove an unwanted connection just select it and press the "Delete" button.

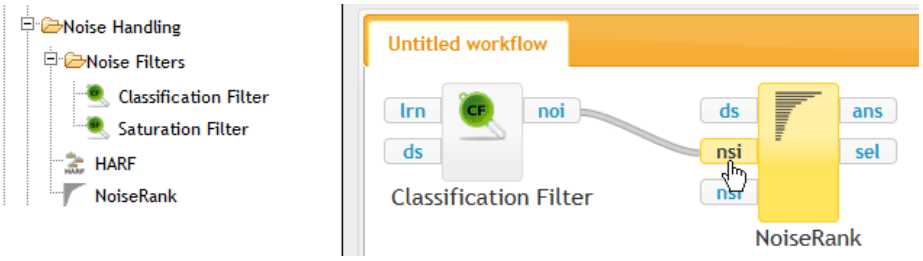


Figure B.3: Connecting two widgets by clicking on the first widget's output connector and then clicking the second widget's input connector.

3. The *Classification filter* widget requires a learning algorithm as input to construct the classifier for noise filtering. The widget supports learning algorithms from the *Orange*[1] package under *Classification and Regresion*, as well as learning algorithms

[1] The Orange data mining environment. `http://orange.biolab.si/`

from the *Weka*[1] package under *Classification*. For each *Classification filter* widget select a learning algorithm widget and connect them.

4. In order to perform classification noise filtering we have to load the desired dataset. The input to the *Classification filter* widgets is data in the form of an *Orange Data Table*. The *Load Dataset to Orange Data Table* widget can convert tab delimited, CSV and ARFF data files to the required data table format. This widget does not have any inputs, but it takes a parameter. By double clicking the widget on the canvas, a dialog box opens and you can enter the required parameter, i.e., the path to your data file. Connect the widget to the *Classification filter* widgets and the NoiseRank widget, and the workflow is complete.

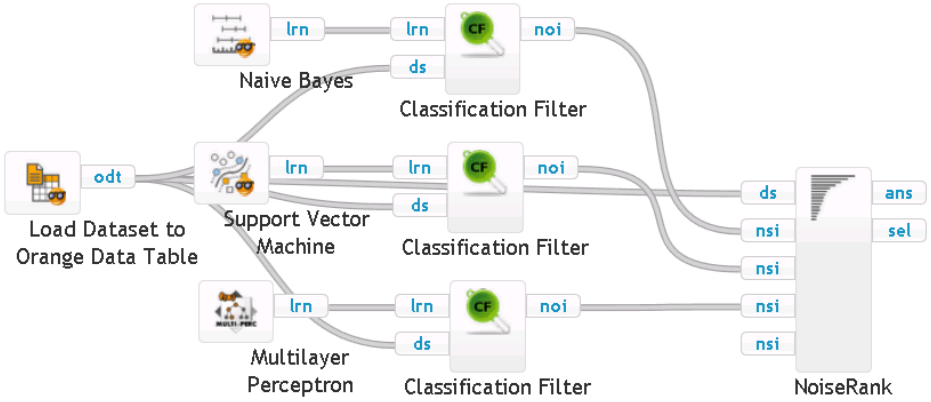


Figure B.4: Example of the NoiseRank workflow constructed by following steps 1 to 4.

5. Finally, the workflow can be executed by clicking the icon that looks like the play button in the top panel of the Workflow editor. Executing individual widgets is also possible by right clicking on the selected widget and using the command "Run" or "Run only this". The first option will automatically run all the required widgets before the selected widget in the workflow, whereas the second option will execute only the selected widget and alert you if any prerequisites for running this widget have not been met. The result of the executed workflow is the NoiseRank visualization presenting a list of noisy instances ranked according to the confidence of the noise detection ensemble, as presented in Figure 6.4 of Chapter 6.

Similarly to the described construction of the NoiseRank workflow, the workflow for performance evaluation of noise detection can be constructed. To enable quantitative performance evaluation, a list of known noisy instances in the evaluated dataset is required. If such a list is not available for a particular dataset, noisy instances can be artificially injected into the dataset. The implemented widget *Add class noise* randomly selects a fraction of the dataset's instances (as set by the widget's parameter) and changes their class labels.

To construct a workflow for visual performance evaluation by the Viper approach, first save the previous workflow, e.g., with the name "NoiseRank example", and then make a copy of it by clicking the *Open icon* in the top panel of the Workflow editor and select the option *Open as new* next to your saved "NoiseRank example" workflow. This copy of the previous workflow can now be saved under a new name, e.g., "Viper example".

For quantitative performance evaluation the NoiseRank widget will not be required and can hence be deleted from the workflow. Now select the *Add class noise* widget from the widget repository, place it on the canvas right after the widget for loading the data, and connect its dataset output to the *Classification filter* widgets. The *Performance evaluation*

[1] The Weka data mining software. `http://www.cs.waikato.ac.nz/ml/weka/`

package in the widget repository holds all the remaining widgets needed for the evaluation of noise detection performance. Use the *Evaluate Detection Algorithm* widget to obtain performance results of the *Classification filter* widgets according to the injected noisy instances by the *Add class noise* widget. Finally, connect the *Evaluation Results to Table*, *Performance Chart* and VIPER*: Visual Performance Evaluation* widgets to the obtained performance results and run the workflow to see the different presentations of noise detection results. The described workflow is shown in Figure B.5.

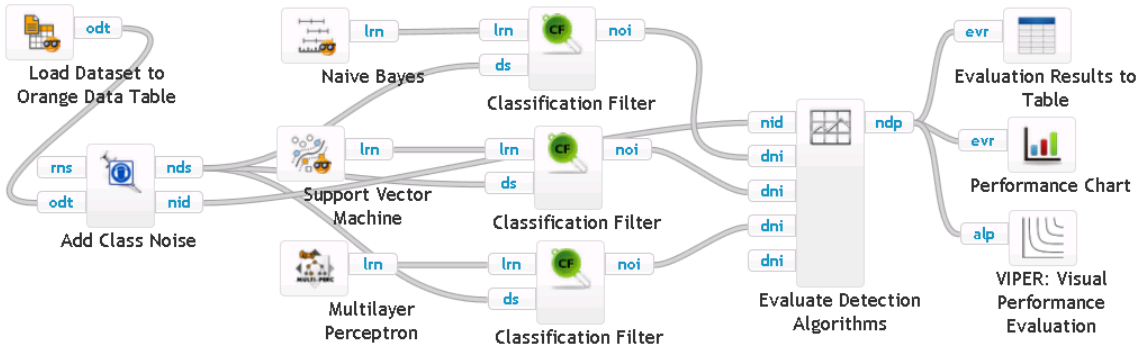


Figure B.5: Example of the VIPER workflow for visual performance evaluation of noise detection.

Including custom noise detection algorithms

The CLOWDFLOWS platform enables to include any algorithm that is available as a WSDL web service[1] into the constructed workflows. Using this functionality, noise detection and visual performance evaluation experiments can be done by incorporating a wide range of different noise detection algorithms. An algorithm available as a web service is imported as a new widget into the widget repository by clicking the *Import webservices* button at the bottom of the *Widget repository*. A dialog box opens and the user can enter the URL to the WSDL file that describes the desired web service(s).

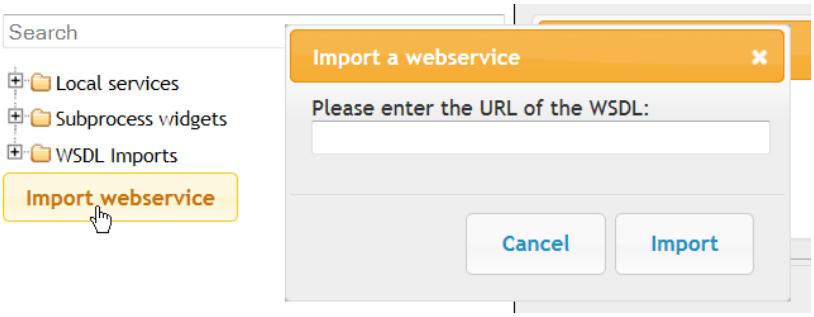


Figure B.6: The dialog box for entering the URL of the WSDL, for importing web services.

The newly added widget of the web service is placed in the widget repository under *WSDL Imports*, and can now be added to any of the user's workflows. However, the user still has to observe the required input and output formats of the widgets and the web services he wants to connect. For example, the inputs of the NOISERANK widget are lists of zero-based indices of the instances in the dataset that were detected by the noise detection

[1] WSDL stands for Web Services Description Language, which is an XML-based interface description language that is used for describing a web service, providing a machine-readable description of how the service can be called, what parameters it expects, and what data structures it returns.

algorithms. Therefore, if a noise detection algorithm in the form of a web service were connected to the NoiseRank widget, its output needs to be in the aforementioned format, or needs to be appropriately converted.

Ensembles and subprocesses

In this section, two additional useful features are presented. Firstly, constructing and using ensembles of noise detection algorithms was simplified by implementing the *Ensemble* widget, which only ensembles the noise detection results of the input algorithms, instead of constructing actual noise detection ensembles. In this way each algorithm is executed only once, rather than within each ensemble, and its results can be used in different ensembles of noise detection results. The *Ensemble* widget is placed in the *Objects* package in the widget repository, since it can be applied generally to list objects and not only to noise detection results. In addition to the list inputs, the widget accepts two parameters, the *ensemble name* for further reference and the *type of voting scheme*, according to which the results are ensembled.

Secondly, the platform offers *Subprocess widgets* that can be used as a *For loop* enabling the repetition of noise detection experiments with different initializing parameters. This can be used to obtain more reliable noise detection results, by averaging the performance over a number of experiments. To support this functionality, the following widgets were implemented in the *Performance evaluation* package: the *Aggregate Detection Results* widget that joins the results of all algorithms into one object, the *Evaluate Repeated Detection* widget that computes the precision, recall and F-measure results for all the experiments, and the *Average and Standard Deviation* widget that averages the results over all experiments and computes the corresponding standard deviations. A workflow example of an experiment using these widgets was provided in Chapter 6 in Figures 6.5 and 6.6.

C ViperCharts Platform: User Guidelines

The ViperCharts platform was developed with the purpose of providing standard and novel performance visualization approaches in an environment that would allow their comparative evaluation and easy access, exporting and sharing of results visualizations. The main menu at the top offers access to all the functionality of the platform, which is divided in the following groups.

- *Chart types* - provides the description of the supported chart types along with an illustrative example.
- *Public charts* - includes publicly accessible charts from the platform's database.
- *Make a chart* - guides the user to create a new performance visualization chart.
- *API* - The provided charting services can also be accessed via the platform's web API.
- *About* - Basic description of the ViperCharts platform.
- *Log in* - Access to user's private charts.

The platform currently supports the following chart types: Scatter charts of performance results in the PR and the ROC space; Curve charts in form of PR curves, lift curves, ROC curves, cost curves, rate-driven curves and Kendall curves; Column charts for the visualization of arbitrary performance measures; and the Critical difference diagrams for the visualization of statistical significance test results. Any of these charts can be created in the *Make a chart* section, by first selecting the desired chart type and then providing the required input data. A view of the data input form is presented in Figure C.1.

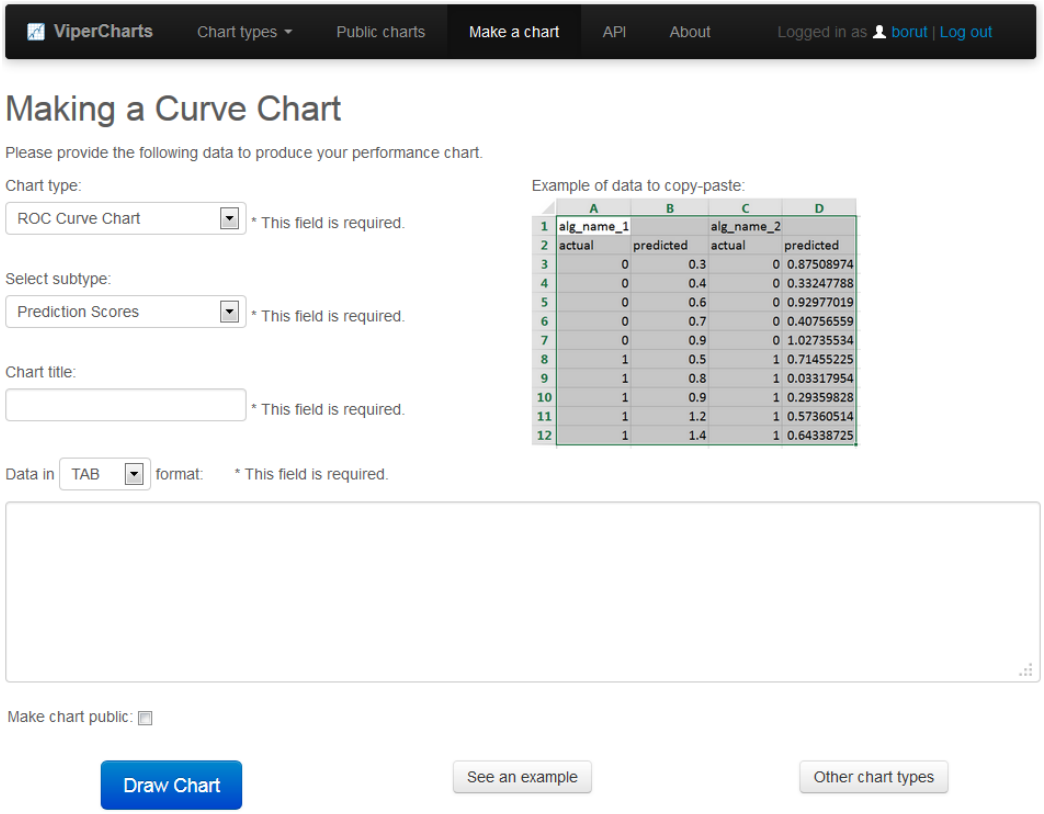


Figure C.1: The data input form for creating a chart in the ViperCharts platform.

Different chart types may require specific input parameters, however all of them have the *Title* field, the *data format* choice, the *input data* field, and the *Make public* checkbox for logged in users. The data can be simply copy-pasted from a spread sheet editor in the form of tab delimited data, from a CSV file, or in the JSON format from the user's favorite programing language. An example of appropriately formated input data is provided on the right hand side of the input form. A specific parameter for the Curve charts worth noticing is the *subtype* parameter with the values: "Prediction scores" meaning higher scores indicate the positive instances, and "Ranks" meaning lower values indicate the positive instances.

The *Draw* button creates the visualization of the provided performance results. The outcome is an interactive chart allowing the user to explore the performance of the evaluated algorithms from different perspectives. For example, the results visualized according to the *Viper* methodology in the PR space with a scatter chart offer additional information by hovering and clicking over specific performance points or the F-isolines. Further visualization options can be enabled on the right hand side of the chart and include: changing the F-measures β parameter, showing the chart's legend, the related column charts visualization and the tabular presentation of the performance results, as shown in Figure C.2.

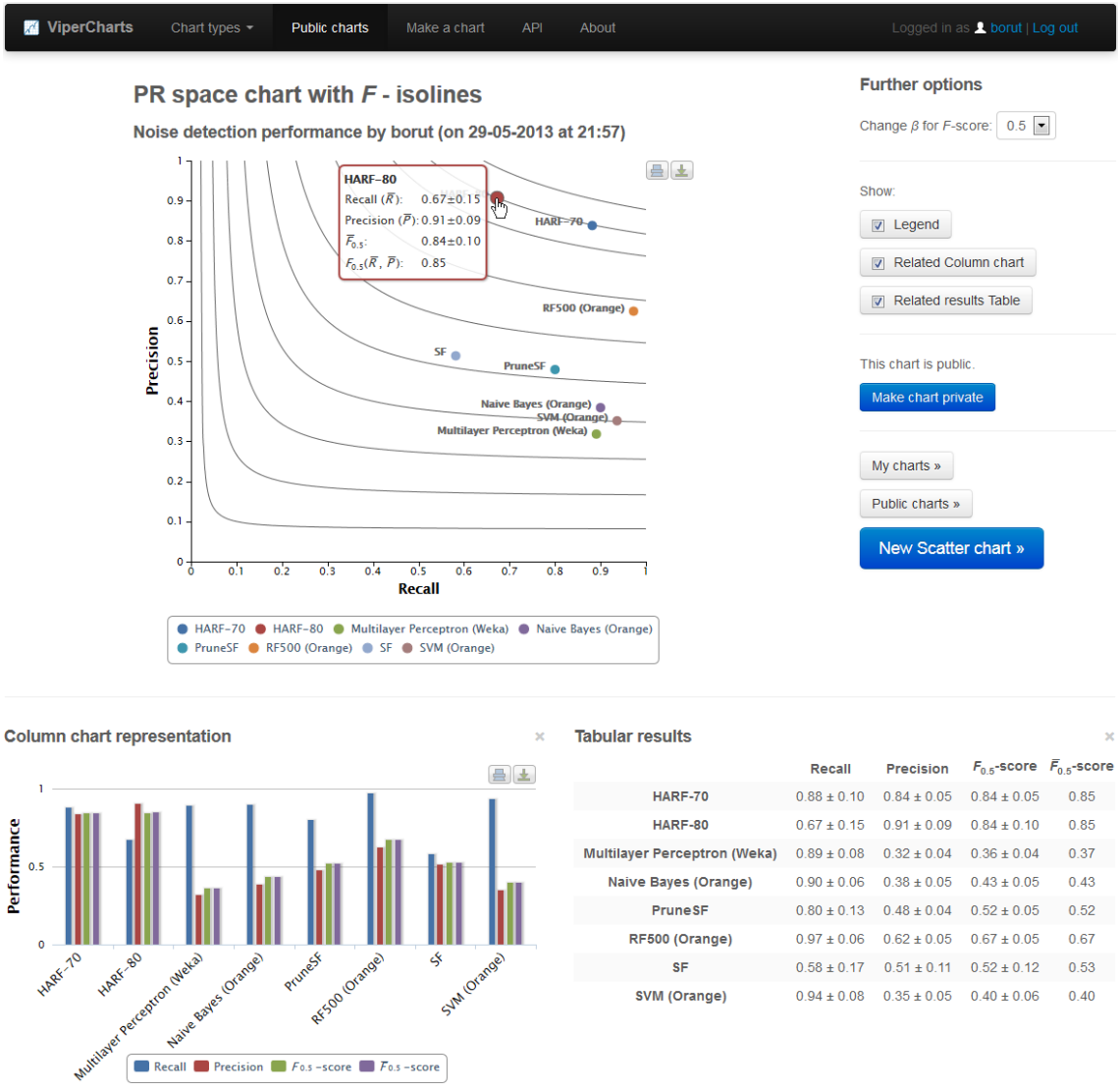


Figure C.2: A scatter chart in the PR space with F-isolines and all other options enabled.

Moreover, any curve chart can be compared side by side with all other curve chart types supported by the platform, since they are all generated from the same input data, but offer a different perspective on the performance of the evaluated algorithms. Further options include showing the legend and the grid of the curve charts, and a table of related performance results, like the area under curve (AUC) results (for ROC, ROC hull and PR curves), the Gini coefficient and the Kendall's tau coefficient.

Performance visualizations can also be created using the ViperCharts web API, as presented in Section 6.3.4. The platform provides a tutorial on the use of the API at `http://viper.ijs.si/api/about/`. Using the API, the supported performance visualizations can be included into other web sites and called from different programing languages.

Author's Bibliography

Publications related to the dissertation

Original scientific article (1.01)[1]

Sluban, B.; Juršič, M.; Cestnik, B.; Lavrač, N. Evaluating outliers for cross-context link discovery. In: Peleg, M.; Lavrač, N.; Combi, C. (eds.) *Proceedings of the 13th Conference on Artificial Intelligence in Medicine, AIME 2011, Bled, Slovenia, July 2-6, 2011*. Lecture Notes in Computer Science **6747**, 343–347 (Springer, Heidelberg, Germany, 2011).

Sluban, B.; Lavrač, N. ViperCharts: Visual performance evaluation platform. In: Blockeel, H.; Kersting, K.; Nijssen, S.; Zelezný, F. (eds.) *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases, ECML PKDD 2013, Prague, Czech Republic, September 23-27, 2013*. Lecture Notes in Computer Science **8190**, 650–653 (Springer, Heidelberg, Germany, 2013).

Sluban, B.; Lavrač, N.; Gamberger, D. Ensemble-based noise detection: noise ranking and visual performance evaluation. *Data Mining and Knowledge Discovery* **28**, 265–303 (2014). [IF=2.88]

Published scientific conference contribution (1.08) - papers

Sluban, B.; Lavrač, N.; Gamberger, D.; Bauer, A. Experiments with saturation filtering for noise elimination from labeled data. In: *Proceeding of the 12th Multi-conference Information Society, Conference on Data Mining and Data Warehouses, SiKDD 2009*. 240–243 (IJS, Ljubljana, Slovenia, 2009).

Sluban, B.; Gamberger, D.; Lavrač, N. Performance analysis of class noise detection algorithms. In: Ågotnes, T. (ed.) *Proceedings of the Fifth Starting AI Researchers' Symposium, STAIRS 2010, Lisbon, Portugal, 16-20 August, 2010*. Frontiers in Artificial Intelligence and Applications **222**, 303–314 (IOS Press, Amsterdam, Netherlands, 2010).

Lavrač, N.; Sluban, B.; Juršič, M. Cross-domain literature mining through outlier document and bridging concept detection. In: *Proceeding of the Workshop on Analysis of Complex NEtworks at ECML PKDD 2010, ACNE 2010*. 35–47 (ACNE Committee, Barcelona, Spain, 2010).

Sluban, B.; Pollak, S.; Coesemans, R.; Lavrač, N. Irregularity detection in categorized document corpora. In: Calzolari, N.; Choukri, K.; Declerck, T.; Dogan, M. U.; Maegaard, B.; Mariani, J.; Odijk, J.; Piperidis, S. (eds.) *Proceedings of the Eighth International Conference on Language Resources and Evaluation, LREC-2012, Istanbul, Turkey, May 23-25, 2012*. 1598–1603 (European Language Resources Association (ELRA), Paris, France, 2012).

[1] Typology of publications according to COBISS bibliography management - `http://www.cobiss.si`, `http://home.izum.si/COBISS/bibliografije/Tipologija_eng.pdf`

Published scientific conference contribution (1.08) - short papers, extended abstracts

Sluban, B.; Lavrač, N. Supporting the search for cross-context links by outlier detection methods. *BMC Bioinformatics* **11 (Suppl 5)**, P2 (2010).

Sluban, B.; Gamberger, D.; Lavrač, N. Advances in class noise detection. In: Coelho, H.; Studer, R.; Wooldridge, M. (eds.) *Proceedings of the 19th European Conference on Artificial Intelligence, ECAI 2010, Lisbon, Portugal, August 16-20, 2010*. Frontiers in Artificial Intelligence and Applications **215**, 1105–1106 (IOS Press, Amsterdam, Netherlands, 2010).

Independent scientific component part or a chapter in a monograph (1.16)

Juršič, M.; Sluban, B.; Cestnik, B.; Grčar, M.; Lavrač, N. Bridging concept identification for constructing information networks from text documents. In: Berthold, M. R. (ed.) *Bisociative Knowledge Discovery*. Lecture Notes in Computer Science **7250**, 66–90 (Springer, Berlin Heidelberg, Germany, 2012).

Sluban, B.; Juršič, M.; Cestnik, B.; Lavrač, N. Exploring the power of outliers for cross-domain literature mining. In: Berthold, M. R. (ed.) *Bisociative Knowledge Discovery*. Lecture Notes in Computer Science **7250**, 325–337 (Springer, Berlin Heidelberg, Germany, 2012).

Publications not related to the dissertation

Original scientific article (1.01)

Grčar, M.; Podpečan, V.; Sluban, B.; Mozetič, I. Ontology querying support in semantic annotation process. In: Anthony, P.; Ishizuka, M.; Lukose, D. (eds.) *Proceedings of the 12th Pacific Rim International Conference on Artificial Intelligence, PRICAI 2012, Kuching, Malaysia, September 3-7, 2012*. Lecture Notes in Computer Science **7458**, 76–87 (Springer, Heidelberg, Germany, 2012).

Sluban, B.; Grčar, M. URL Tree: Efficient Unsupervised Content Extraction from Streams of Web Documents. In: He, Q.; Iyengar, A.; Nejdl, W.; Pei, J.; Rastogi, R. (eds.) *Proceedings of the 22nd ACM International Conference on Information and Knowledge Management, CIKM'13, San Francisco, CA, USA, October 27 - November 1, 2013*. 2267–2272 (ACM, New York, NY, USA, 2013).

Biography

Borut Sluban was born on January 5, 1984 in Maribor, Slovenia. He attended primary and secondary school in Maribor. In 2002, he started his studies at the *Faculty of Mathematics and Physics*, University of Ljubljana, Slovenia. He was enrolled in the BSc program *Applied Mathematics*. He finished his studies and defended his BSc thesis in June 2009 under the supervision of Prof. Nada Lavrač and co-supervision of Prof. Andrej Bauer.

In the fall of 2009 he started his graduate studies at the Jožef Stefan International Postgraduate School, Ljubljana, Slovenia. He enrolled in the PhD program *Information and Communication Technologies* under the supervision of Prof. Nada Lavrač. During his PhD studies he was employed as research assistant at the *Department of Knowledge Technologies* at the Jožef Stefan Institute, Ljubljana, Slovenia. In this time he collaborated on the following projects of the European 7th Framework Programme: BISON (BISOciation Networks for Creative Information Discovery), FIRST (Large Scale Information Extraction and Integration Infrastructure for Supporting Financial Decision Making), and FOC (Forecasting Financial Crisis). His research is in the field of data mining, focusing on the development and application of data mining approaches. His research, presented in this thesis, focused on the development of ensemble-based noise and outlier detection approaches to be used by experts in real-life applications, and on developments of visualization approaches improving the evaluation of noise detection performance.