MEDNARODNA
PODIPLOMSKA ŠOLA
JOŽEFA STEFANA

Data and Text Mining: Hands-on Labs

2024 / 2025

Blaž Škrlj

Department of Knowledge Technologies

Jožef Stefan Institute

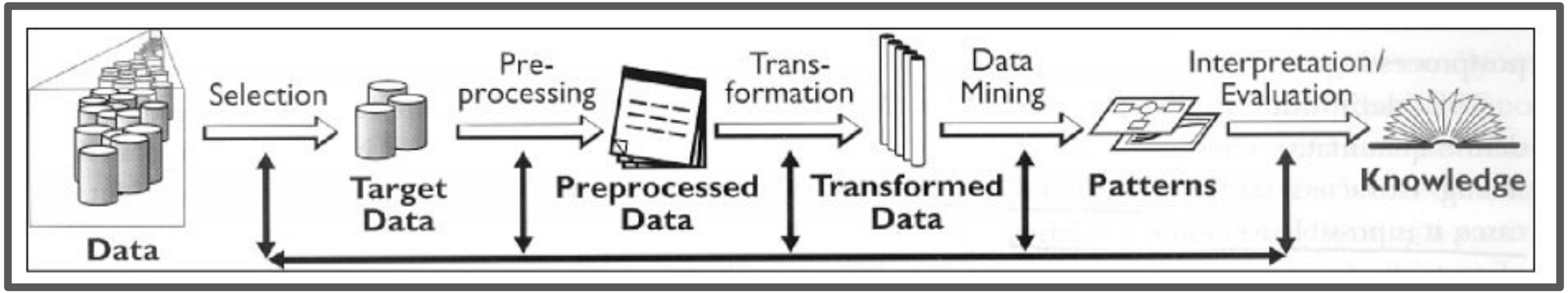
Ljubljana, Slovenia

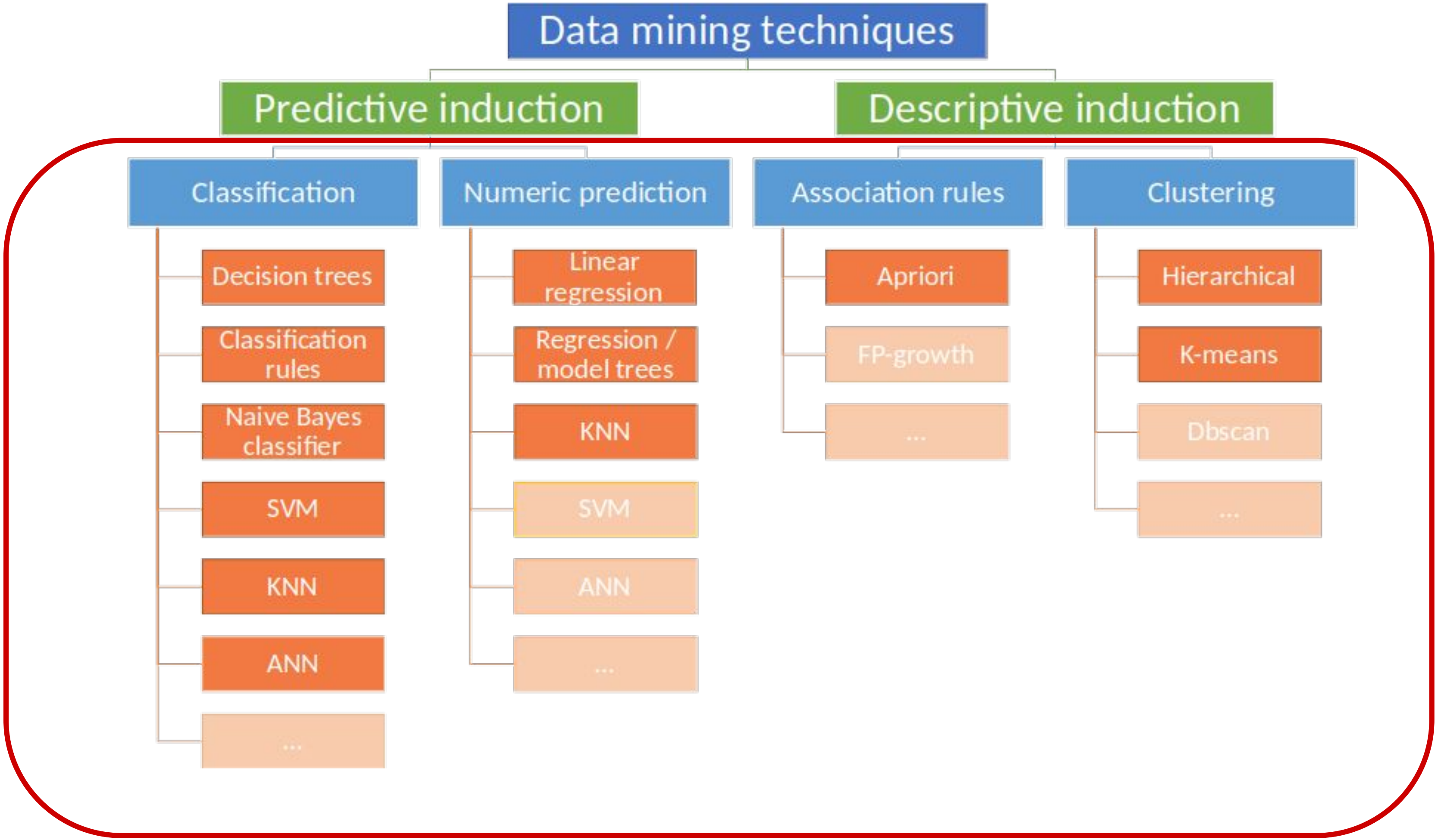
Overview

- Hands-on Orange3
  - Machine learning and Data Visualization
  - Interactive Analysis
  - **Visual Programming**

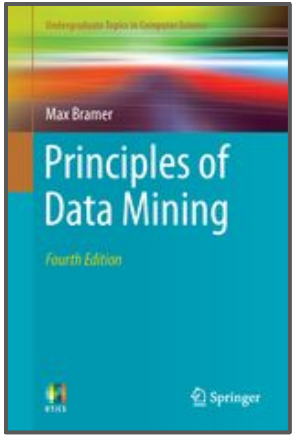
- Main goals
  - Understanding of the **data mining process**
  - Being able to perform analysis on your own
  - Critical evaluation of the results

KDD vs. ML/DM


Data
Selection
Target Data
Pre-processing
Preprocessed Data
Trans-formation
Transformed Data
Data Mining
Patterns
Interpretation/ Evaluation
Knowledge


Data mining techniques
Predictive induction
Descriptive induction
Classification
Numeric prediction
Association rules
Clustering
Decision trees
Classification rules
Naive Bayes classifier
SVM
KNN
ANN
...
Linear regression
Regression / model trees
KNN
SVM
ANN
...
Apriori
FP-growth
...
Hierarchical
K-means
Dbscan
...

Max Bramer: Principles of data mining (2007)



Attribute Types

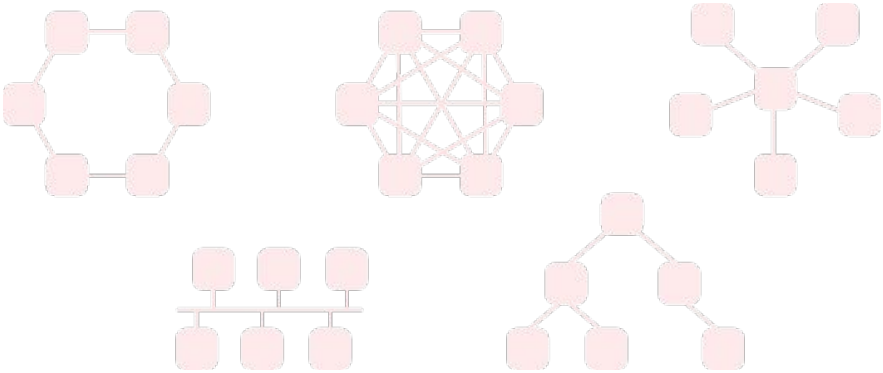
1. Categorical
   a. Nominal (e.g., Colors -> R,G,B)
   b. Binary (e.g., class presence -> yes, no)
   c. Ordinal (e.g., Size -> small, medium, large, …)

2. Numerical
   a. Integer (Number of pets)
   b. Real (wavelength, temperature etc.)

Why is this relevant?

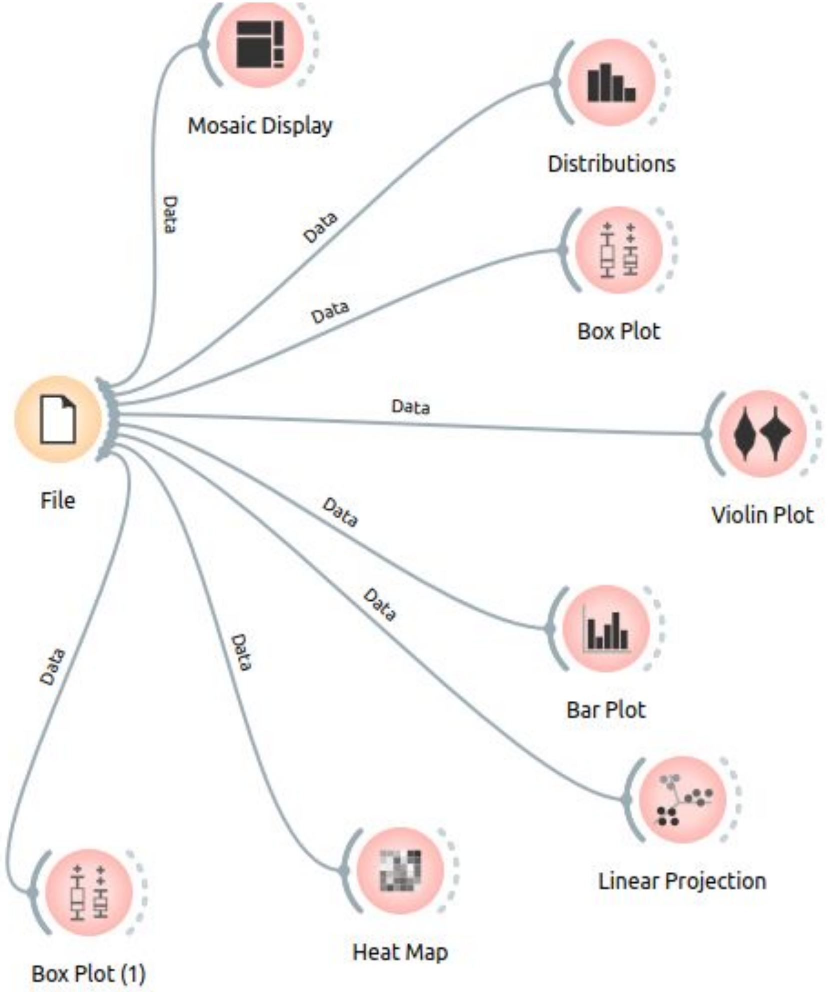
Complex data types

1. Time series - *data in time*
2. Texts - *instances are documents*
3. Graphs - *instances are related (explicitly)*
4. Images - *instances are images*
5. Multi-modal data - *combined spaces*



Data visualization

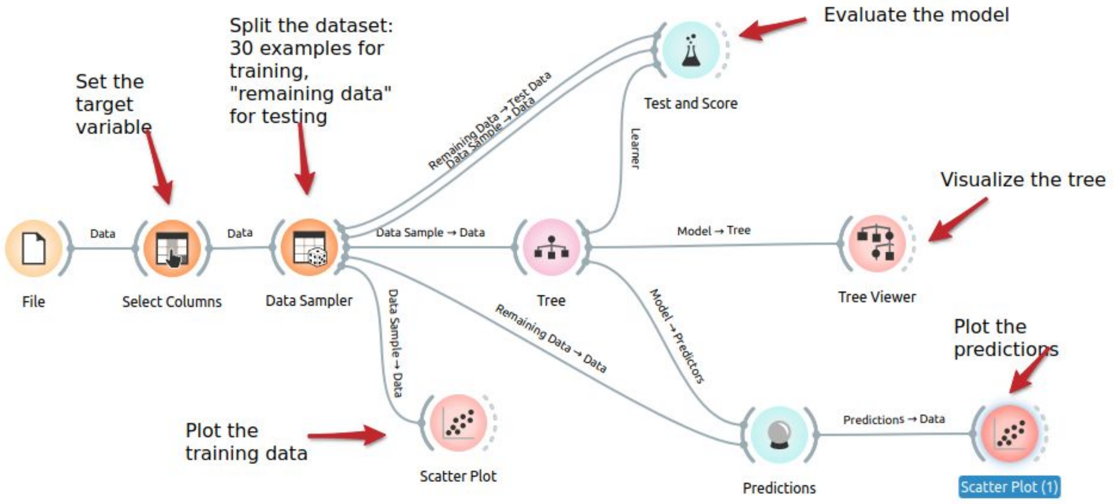
Wf1 - visualization



Classification

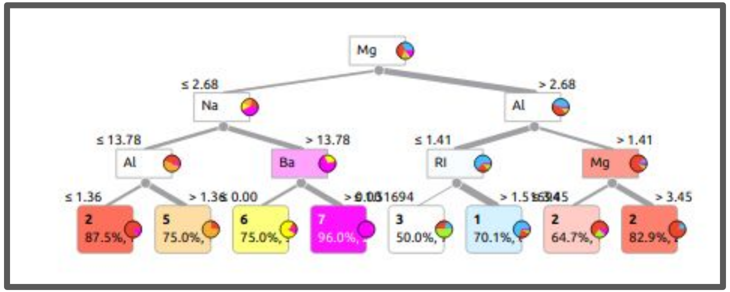
The classification problem

- Given a collection of examples, assign them to categories.
  - Magazine reader (or not)
  - A patient at risk of falling ill
  - Likely buyers
  - Types of plants
  - Gene functions



More formally

1. Given a collection of instances X with corresponding labels Y, identify f: X -> Y.
    a. Instances are described by attributes
    b. The target variable is an attribute we are interested in (e.g., illness, categorical)
    c. The values of the target variable are called labels.
    d. The goal is to assign labels to new instances, as accurately as possible.



(nominal) target variable

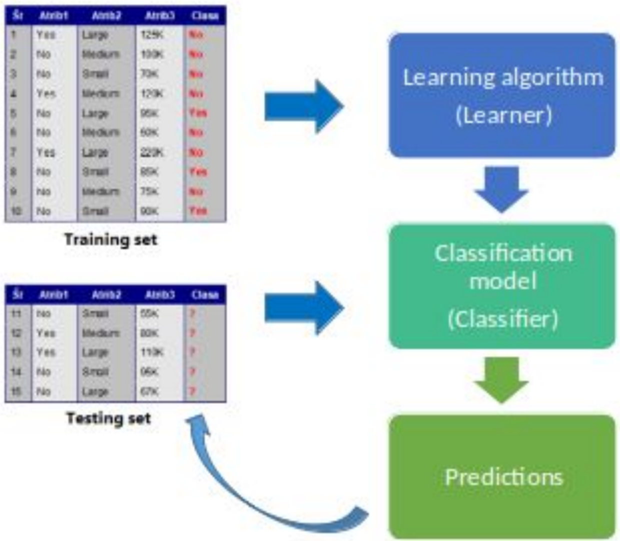
attributes

Examples or instances

Person	Age	Prescription	Astigmatic	Tear_Rate	**Lenses**
P1	young	myope	no	normal	**YES**
P2	young	myope	no	reduced	**NO**
P3	young	hypermetrope	no	normal	**YES**
P4	young	hypermetrope	no	reduced	**NO**
P5	young	myope	yes	normal	**YES**
P6	young	myope	yes	reduced	**NO**
P7	young	hypermetrope	yes	normal	**YES**
P8	young	hypermetrope	yes	reduced	**NO**
P9	pre-presbyopic	myope	no	normal	**YES**
P10	pre-presbyopic	myope	no	reduced	**NO**
P11	pre-presbyopic	hypermetrope	no	normal	**YES**
P12	pre-presbyopic	hypermetrope	no	reduced	**NO**
P13	pre-presbyopic	myope	yes	normal	**YES**
P14	pre-presbyopic	myope	yes	reduced	**NO**
P15	pre-presbyopic	hypermetrope	yes	normal	**NO**
P16	pre-presbyopic	hypermetrope	yes	reduced	**NO**
P17	presbyopic	myope	no	normal	**NO**
P18	presbyopic	myope	no	reduced	**NO**
P19	presbyopic	hypermetrope	no	normal	**YES**
P20	presbyopic	hypermetrope	no	reduced	**NO**
P21	presbyopic	myope	yes	normal	**YES**
P22	presbyopic	myope	yes	reduced	**NO**
P23	presbyopic	hypermetrope	yes	normal	**NO**
P24	presbyopic	hypermetrope	yes	reduced	**NO**

classes = values of the (nominal) target variable

The basic classification schema



Šr	Atrib1	Atrib2	Atrib3	Class
1	Yes	Large	125K	No
2	No	Medium	100K	No
3	No	Small	70K	No
4	Yes	Medium	120K	No
5	No	Large	95K	Yes
6	No	Medium	60K	No
7	Yes	Large	220K	No
8	No	Small	85K	Yes
9	No	Medium	75K	No
10	No	Small	90K	Yes

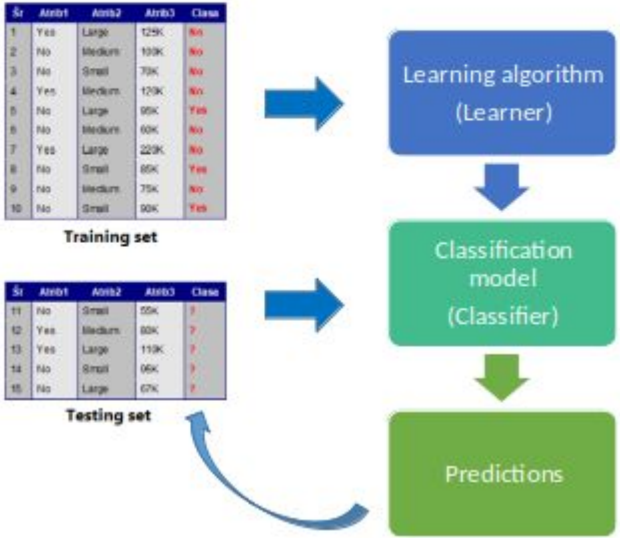
Training set

Šr	Atrib1	Atrib2	Atrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

Testing set

- A classifier is a function that maps from the attributes to the classes
  - Classifier(attributes) = Classes
  - $f(X) = Y$
- In training, the attributes and the classes are known (training examples) and we are learning a mapping function *f* (the clasifier)
  - $?(X) = Y$
- When predicting, the attributes and the classifier are known and we are assigning the classes
  - $f(X) = ?$
- What about evaluation?

The basic classification schema



Št	Atrib1	Atrib2	Atrib3	Class
1	Yes	Large	125K	No
2	No	Medium	100K	No
3	No	Small	70K	No
4	Yes	Medium	120K	No
5	No	Large	95K	Yes
6	No	Medium	60K	No
7	Yes	Large	220K	No
8	No	Small	85K	Yes
9	No	Medium	75K	No
10	No	Small	90K	Yes

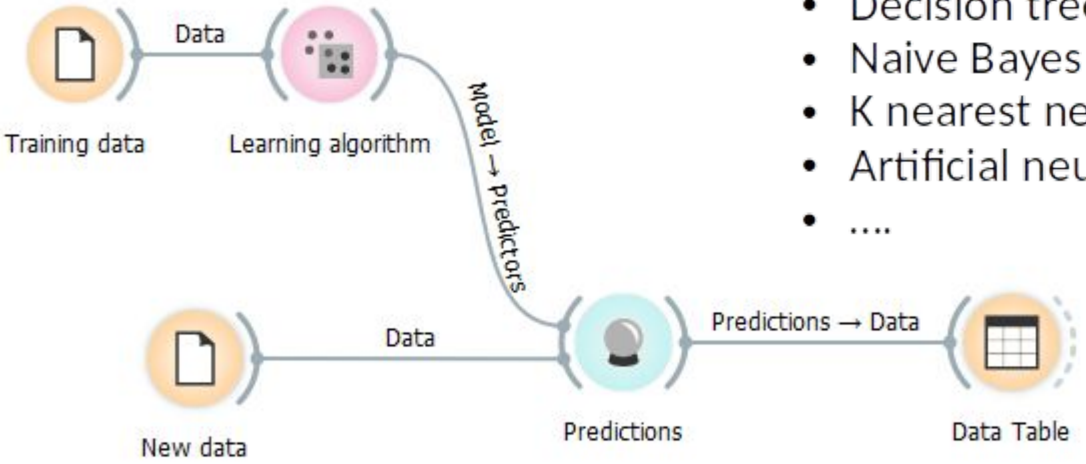
Training set

Št	Atrib1	Atrib2	Atrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

Testing set

- A classifier is a function that maps from the attributes to the classes
  - Classifier(attributes) = Classes
  - $f(X) = Y$
- In training, the attributes and the classes are known (training examples) and we are learning a mapping function *f* (the clasifier)
  - $?(X) = Y$
- When predicting, the attributes and the classifier are known and we are assigning the classes
  - $f(X) = ?$
- When evaluating, *f*, *X* and *Y* are known. We compute the predictions $Y_p = f(X)$ and evaluate the difference between *Y* and Y_p.

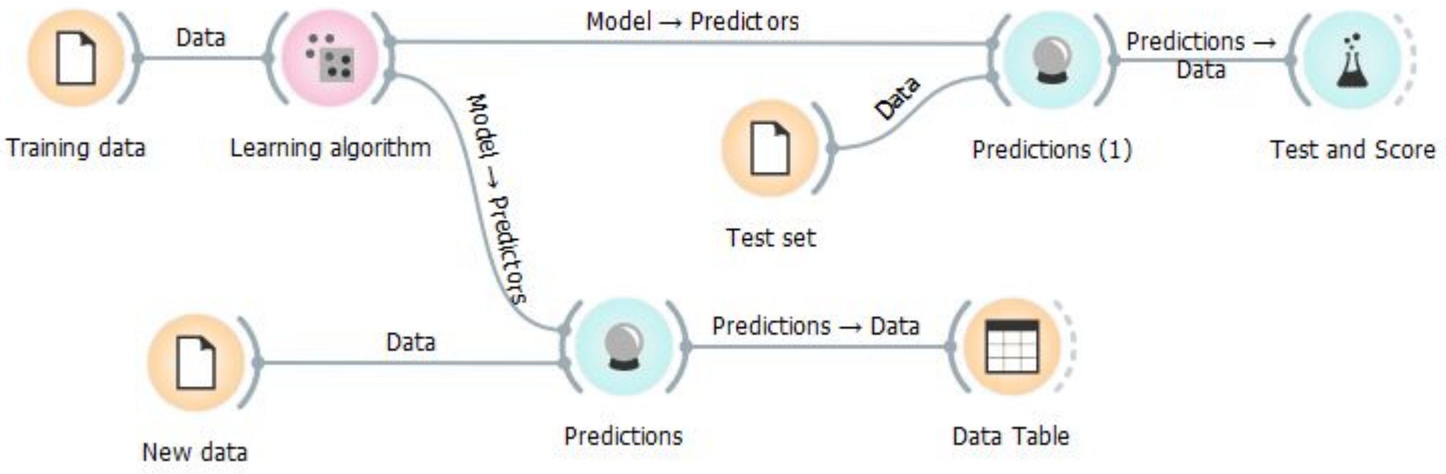
Basic classification schema



- We train the model on the train set
- We predict the target for the new instances
- There are several classification algorithms:
    - Decision trees
    - Naive Bayes classifier
    - K nearest neighbors (KNN)
    - Artificial neural networks (ANN)
    - ….

What about evaluation

- We train the model on the train set
- We evaluate on the test set
- We classify the new instances



A recap

Target variable

Descriptive variables

Examples

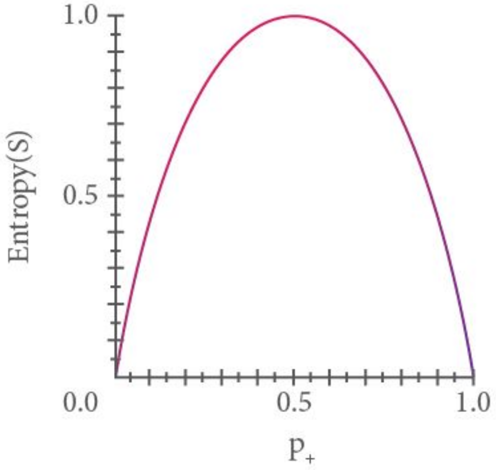
	survived	status	age	sex
1281	no	third	child	male
1282	no	third	child	male
1283	no	third	child	male
1284	no	third	child	male
1285	no	third	child	male
1286	yes	third	child	female
1287	yes	third	child	female
1288	yes	third	child	female
1289	yes	third	child	female
1290	yes	third	child	female
1291	yes	third	child	female
1292	yes	third	child	female
1293	yes	third	child	female
1294	yes	third	child	female
1295	yes	third	child	female
1296	yes	third	child	female
1297	yes	third	child	female
1298	yes	third	child	female
1299	yes	third	child	female
1300	no	third	child	female

Algorithm 1: Decision trees

Trees: Algorithmic background

Induce a decision tree on set S:

1. Compute the **entropy** E(S) of the set S
2. **IF** E(S) = 0
3. The current set is “clean” and therefore a leaf in our tree
4. **IF** E(S) > 0
5. Compute the **information gain** of each attribute Gain(S, A)
6. The attribute A with the highest information gain becomes the root
7. Divide the set S into subsets S_i according to the values of A
8. Repeat steps 1-7 on each S_i



https://www.hackerearth.com/practice/machine-learning/machine-learning-algorithms/ml-decision-tree/tutorial/

Quinlan, J. R. 1986. Induction of Decision Trees. Mach. Learn. 1, 1 (Mar. 1986), 81-106

Decision Trees - intuition - Information Gain

number of examples in the subset S_v

(probability of the branch)

set S

attribute A

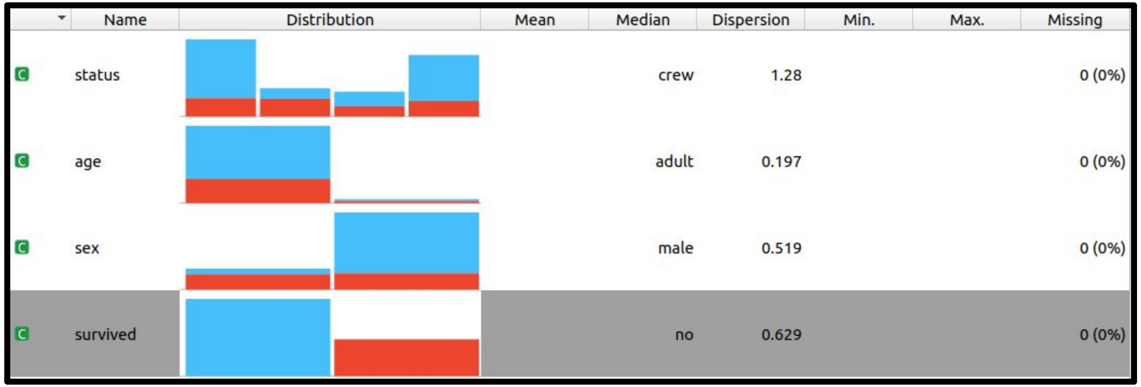
$$Gain(S, A) = E(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} \cdot E(S_v)$$

Entropy of the set S

Entropy of the subset S_v

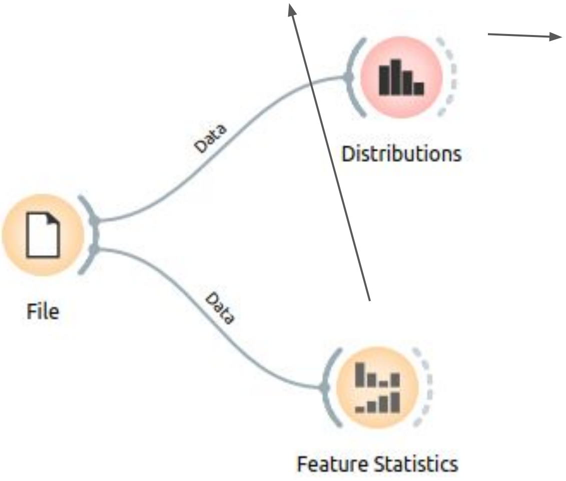
number of examples in set S

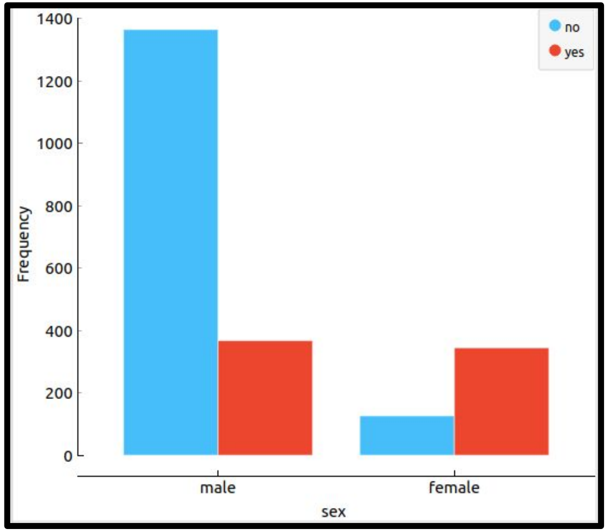
Warm up



	Name	Distribution	Mean	Median	Dispersion	Min.	Max.	Missing
C	status			crew	1.28			0 (0%)
C	age			adult	0.197			0 (0%)
C	sex			male	0.519			0 (0%)
C	survived			no	0.629			0 (0%)

Select: titanic.tab





Illustrative example (IG)

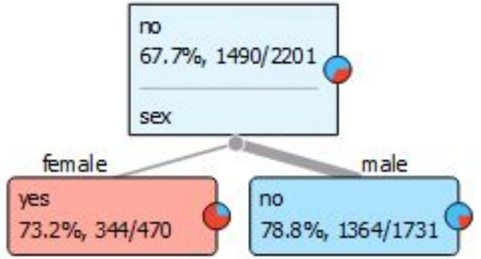
1. Compute entropy of the entire set
2. Identify subsets based on a given attribute's values
3. Compute entropy of each subset
4. Compute the IG

Higher gain = better!

	NO	YES	total
	1490	721	2211
class probability	0.674	0.326	
pi * log (pi, 2)	-0.384	-0.527	
entropy	0.911		

female	NO	YES	total
	136	334	470
Class probability pi	0,289	0,711	
pi * log (pi, 2)	-0,52	-0,35	
entropy	0,868		

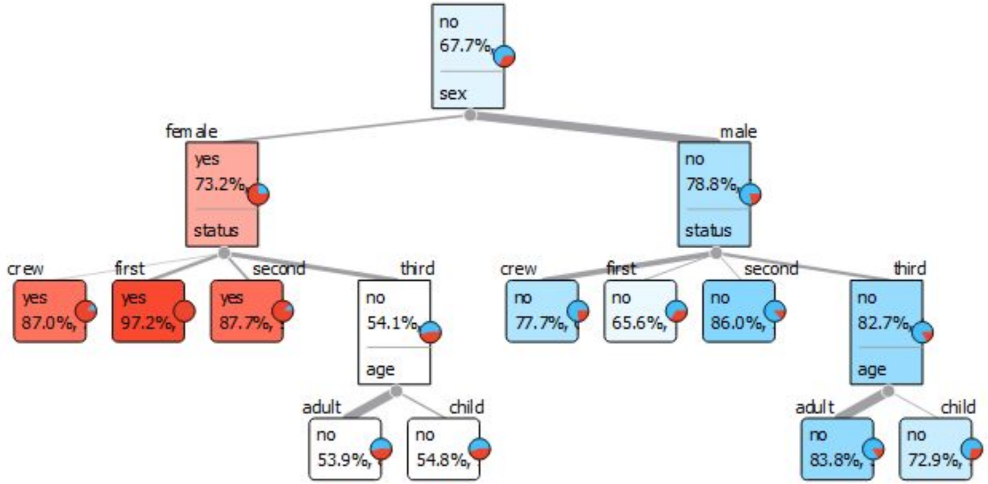
male	NO	YES	total
	1364	367	1731
Class probability pi	0,788	0,212	
pi * log (pi, 2)	-0,27	-0,47	
entropy	0,745		



$$Gain(S,A) = E(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} \cdot E(S_v)$$

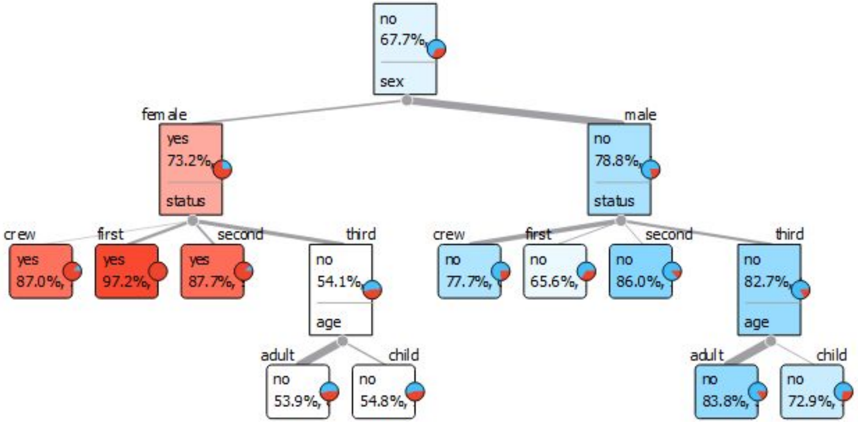
$$Gain(S, Sex) = 0{,}911 - \left(\frac{470}{2201} \cdot 0{,}868 + \frac{1731}{2201} \cdot 0{,}745 \right) = 0{,}166$$

Classification via traversal



	status	age	sex	survived?
1	third	child	male	
2	third	child	female	
3	crew	adult	male	
4	first	adult	male	
5	second	adult	male	
6	third	adult	male	
7	first	adult	female	
8	second	adult	female	
9	third	adult	female	
10	third	child	male	

From trees to rules

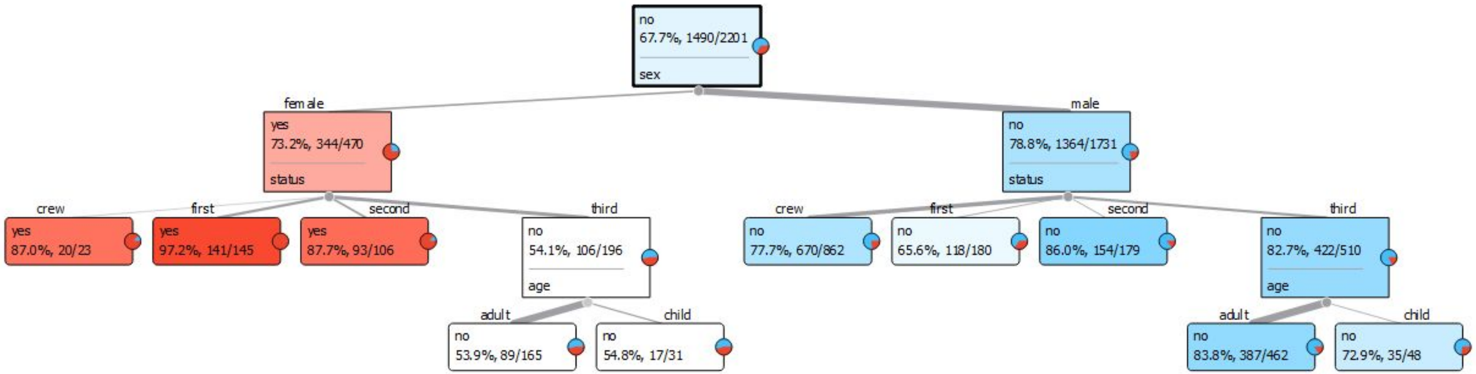


- sex = female & status = crew → survived = yes
- sex = female & status = first → survived = yes
- sex = female & status = second → survived = yes
- sex = female & status = third & age = adult → survived = no
- sex = female & status = third & age = child → survived = no
- sex = male & status = crew → survived = no
- sex = male & status = first → survived = no
- sex = male & status = second → survived = no
- sex = male & status = third & age = adult → survived = no
- sex = male & status = third & age = child → survived = no

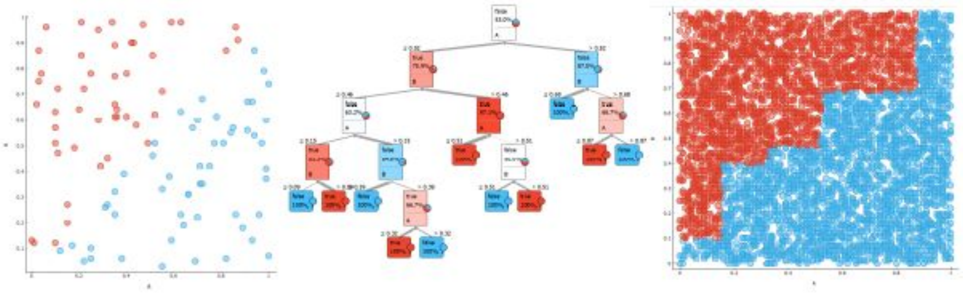
One example -> one rule -> one path!

A note on interpretability

- Attribute importance and its placement
- Visualization interpretation



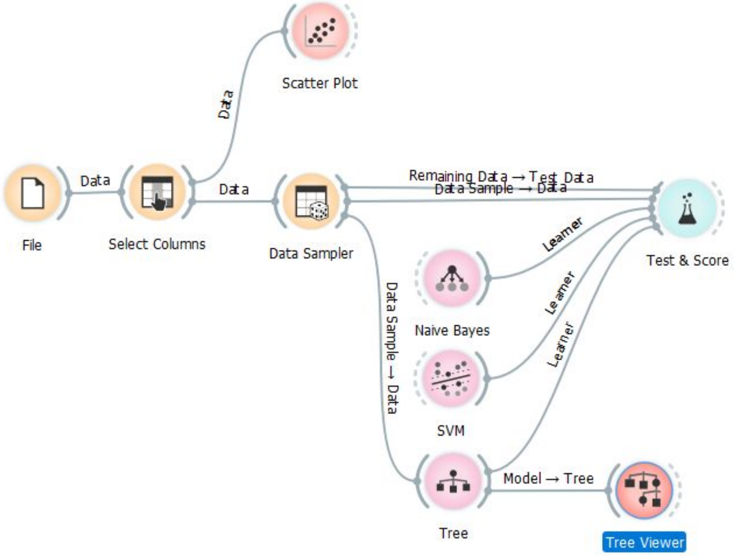
A note on language bias



Training set

Decision tree

Test set



Algorithm 2: Rules

CN2 Rule Induction

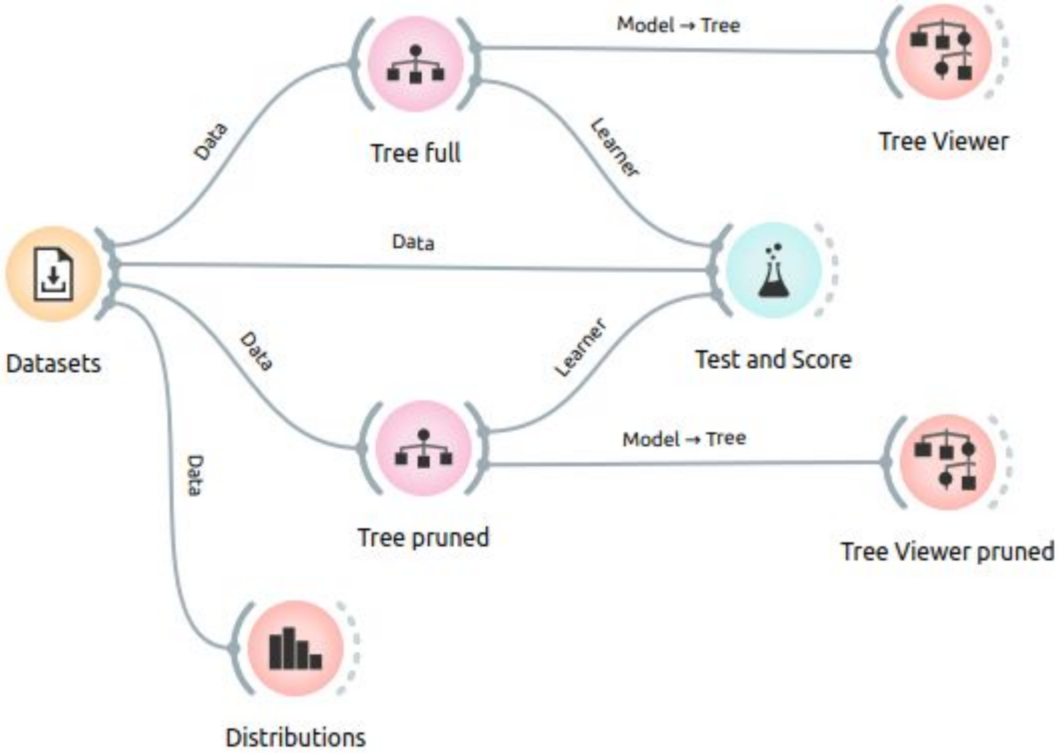
- Based on the "covering" principle
- Dependent on the BEST_CPX
  - Generates candidate rules
  - Tests their "significance"
  - Prunes the complex space
- Why is this different to tree chopping?
  - Not as greedy
  - Effectively upgrades a rule list

```
Let: E be a set of training examples;

Procedure CN2(E) returning RULE_LIST:

let RULE_LIST be the empty list;
repeat
    let BEST_CPX be Find_Best_Complex(E);
    if BEST_CPX is not nil then
        Let E' be the examples covered by BEST_CPX;
        Remove from E the examples E' covered by BEST_CPX;
        Let C be the most common class of examples in E';
        Add the rule 'if BEST_CPX then class=C' to the end of RULE_LIST,
until BEST_CPX is nil or E is empty.
return RULE_LIST.

Procedure Find_Best_Complex(E) returning BEST_CPX:
let the set STAR contain only the empty complex;
let BEST_CPX be nil;
let SELECTORS be the set of all possible selectors;
while STAR is not empty,
    specialize all complexes in STAR as follows:
    let NEWSTAR be the set {x ∧ y|x ∈ STAR, y ∈ SELECTORS};
    Remove all complexes in NEWSTAR that are either in STAR (i.e., the
        unspecialized ones) or are null (e.g. big = y ∧ big = n)
    for every complex Ci in NEWSTAR:
        if Ci is statistically significant when tested on E and better than
            BEST_CPX according to user-defined criteria when tested on E,
        then  replace the current value of BEST_CPX by Ci;
    repeat remove worst complexes from NEWSTAR
        until size of NEWSTAR is ≤ user-defined maximum;
    let STAR be NEWSTAR;
return BEST_CPX.
```

Clark, P. and Niblett, T (1989) The CN2 induction algorithm. Machine Learning 3(4):261-283.

Practice time - basic tree learning

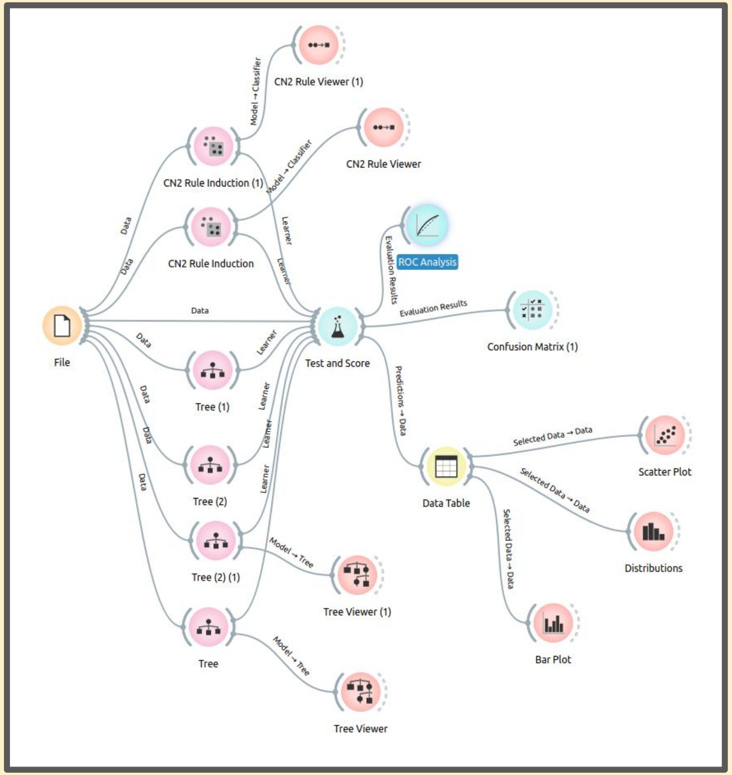
1. Open Orange3
2. Load the data
3. Explore the data
4. Build and visualize a tree
5. Generate rules (CN2)



HW questions

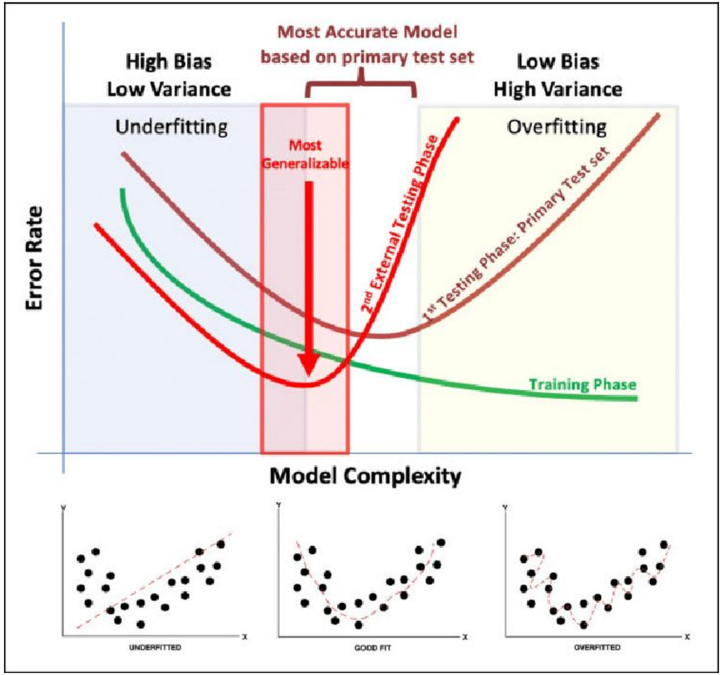
1. How does an attribute with IG=~1 look like?
2. What about IG ~ 0?
3. How would you compute the information gain of a numeric attribute (Hint: See Kullback-Leibler Divergence)
4. Explain the difference between TDIDT- and CN2-based rules.
   a. How they are constructed
   b. How much time does it take to construct them (based on pseudocode)
   c. Are some more greedy than others? Why?

Evaluation

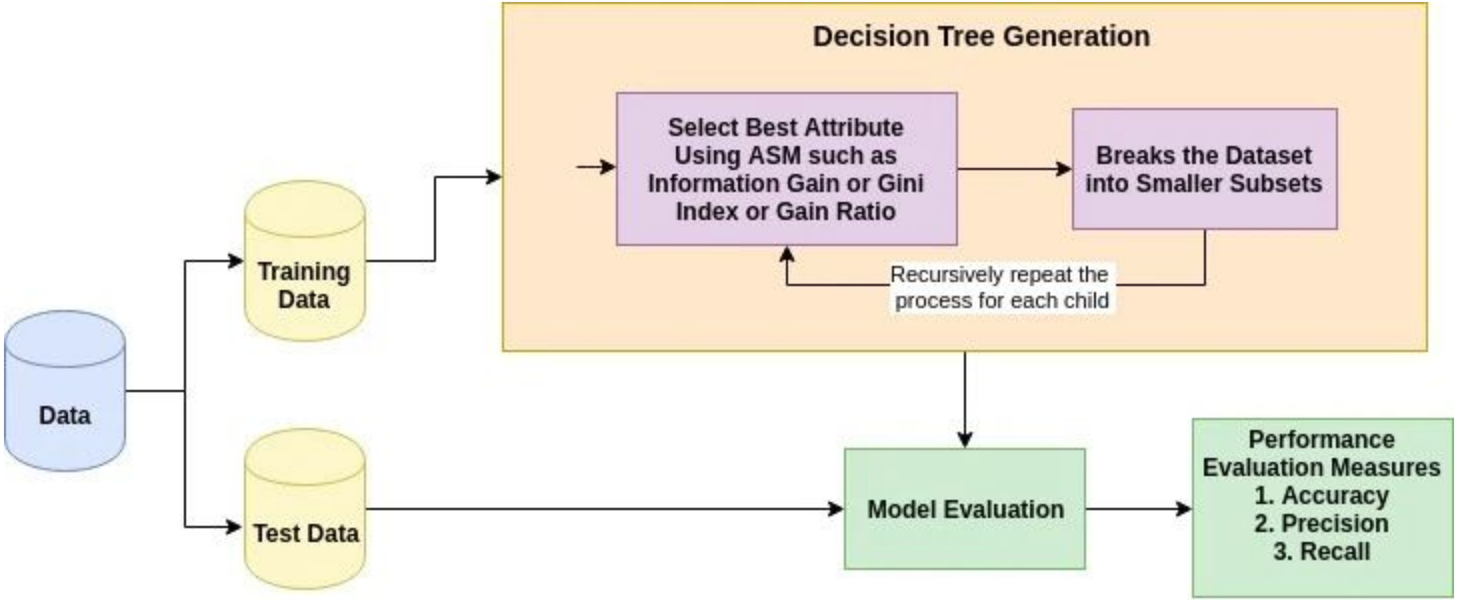

CN2 Rule Viewer (1)
CN2 Rule Viewer
CN2 Rule Induction (1)
CN2 Rule Induction
ROC Analysis
File
Test and Score
Confusion Matrix (1)
Tree (1)
Tree (2)
Tree (2) (1)
Tree
Data Table
Scatter Plot
Distributions
Tree Viewer (1)
Tree Viewer
Bar Plot
Data
Learner
Model → Classifier
Evaluation Results
Predictions → Data
Selected Data → Data
Model → Tree

(over) Fitting a model on the whole data set is … not ok.

- Does the model **generalize**?
- How do we **measure** the performance?
- What are we measuring, really?



https://www.researchgate.net/figure/Bias-variance-trade-off-in-machine-learning-This-figure-illustrates-the-trade-off_fig2_335604816



https://www.datacamp.com/community/tutorials/decision-tree-classification-python

Key take-away message

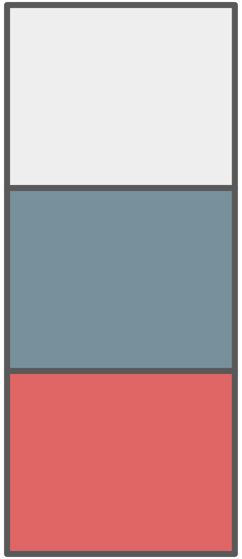
Test on a separate test set

Person	Age	Prescription	Astigmatic	Tear_Rate	Lenses
P1	young	myope	no	normal	**YES**
P2	young	myope	no	reduced	**NO**
P3	young	hypermetrope	no	normal	**YES**
P4	young	hypermetrope	no	reduced	**NO**
P5	young	myope	yes	normal	**YES**
P6	young	myope	yes	reduced	**NO**
P7	young	hypermetrope	yes	normal	**YES**
P8	young	hypermetrope	yes	reduced	**NO**
P9	pre-presbyopic	myope	no	normal	**YES**
P10	pre-presbyopic	myope	no	reduced	**NO**
P11	pre-presbyopic	hypermetrope	no	normal	**YES**
P12	pre-presbyopic	hypermetrope	no	reduced	**NO**
P13	pre-presbyopic	myope	yes	normal	**YES**
P14	pre-presbyopic	myope	yes	reduced	**NO**
P15	pre-presbyopic	hypermetrope	yes	normal	**NO**
P16	pre-presbyopic	hypermetrope	yes	reduced	**NO**
P17	presbyopic	myope	no	normal	**NO**
P18	presbyopic	myope	no	reduced	**NO**
P19	presbyopic	hypermetrope	no	normal	**YES**
P20	presbyopic	hypermetrope	no	reduced	**NO**
P21	presbyopic	myope	yes	normal	**YES**
P22	presbyopic	myope	yes	reduced	**NO**
P23	presbyopic	hypermetrope	yes	normal	**NO**
P24	presbyopic	hypermetrope	yes	reduced	**NO**

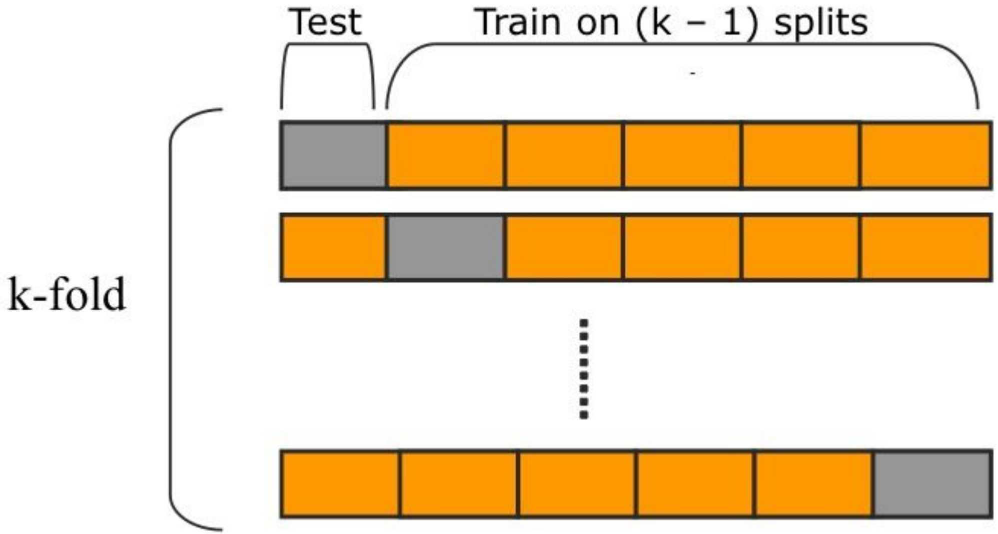
30% of examples are (randomly) selected for testing

Common evaluation scenarios

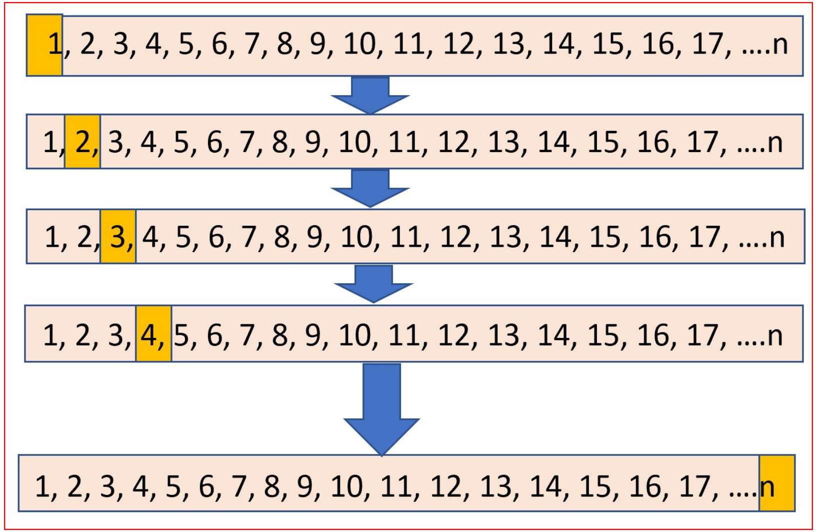
1. Train-val-test split
2. K-fold cross validation
3. Leave-one-out validation
4. Random sampling (and averaging)
5. Stratified splits



K-fold Cross Validation

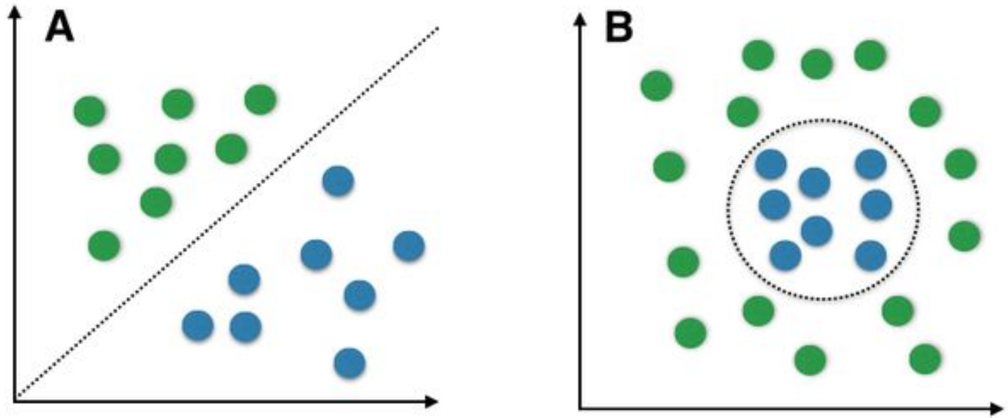


Leave-one-out



Classification Quality

- Splitting the data is the first step
- The second one involves computing a score (on unseen samples)



Confusion matrix

Actual \ Predicted	unacc	acc	good	v-good	Σ
unacc	1154	54	2	0	1210
acc	94	276	14	0	384
good	0	44	22	3	69
v-good	0	25	0	40	65
Σ	1248	399	38	43	1728

Actual \ Predicted	no	yes	Σ
no	1364	126	1490
yes	362	349	711
Σ	1726	475	2201

Matrix of correct/incorrect classifications

- Rows = actual
- Columns = predicted
- Correct = diagonal

Accuracy

TP: true positives
The number of positive instances that are classified as positive

FP: false positives
The number of negative instances that are classified as positive

FN: false negatives
The number of positive instances that are classified as negative

TN: true negatives
The number of negative instances that are classified as negative

Correct classification	Classified as	
	+	−
+	true positives	false negatives
−	false positives	true negatives

Acc = sum(diag)/sum(all)

Baselines

- Knowing that we are able to classify with a certain accuracy **is fine**, but is that good/bad?
- **Baselines are crucial** in machine learning - comparing your method against other methods is the most credible way to prove its actual performance.

A simple baseline: Majority class **#most frequent class / all**

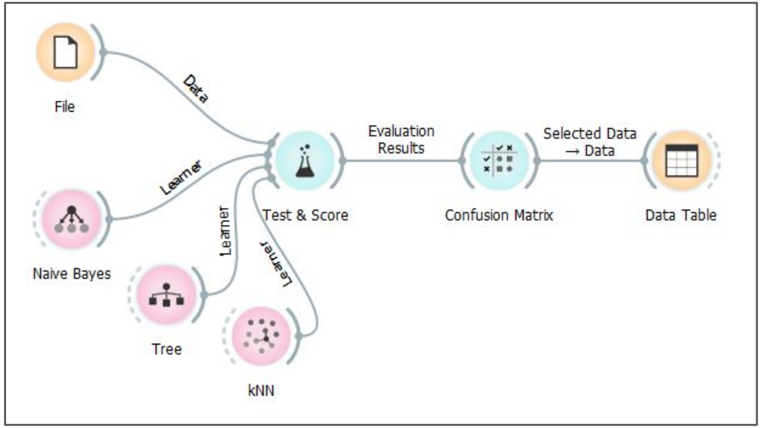
		Predicted		
		no	yes	Σ
Actual	no	1364	126	**1490**
	yes	362	349	**711**
	Σ	**1726**	**475**	**2201**

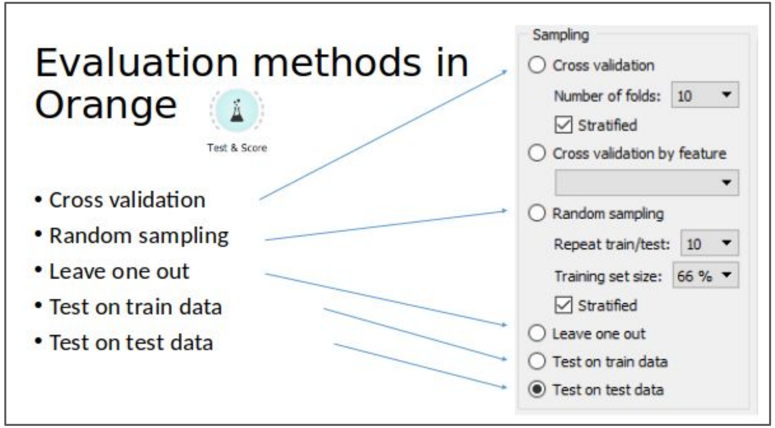
= 68%

Class-specific metrics

True Positive Rate or Hit Rate or Recall or Sensitivity or TP Rate	TP/P	The proportion of positive instances that are correctly classified as positive
Precision or Positive Predictive Value	TP/(TP+FP)	Proportion of instances classified as positive that are really positive
F1 Score	$(2 \times \text{Precision} \times \text{Recall}) / (\text{Precision} + \text{Recall})$	A measure that combines Precision and Recall
Accuracy or Predictive Accuracy	(TP + TN)/(P + N)	The proportion of instances that are correctly classified

Practice time: evaluation and Orange3





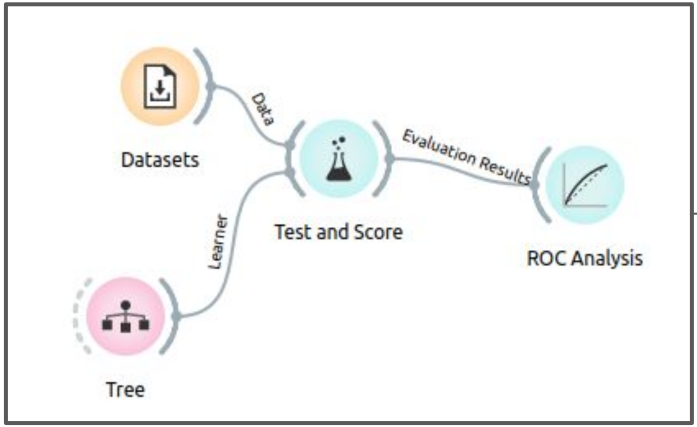
ROC curves

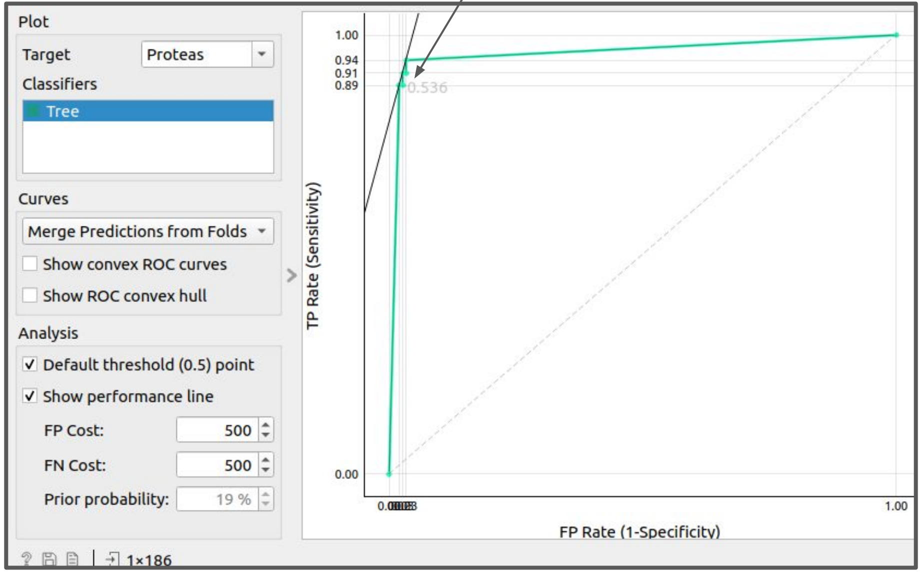
Key point: Varying discrimination threshold has great impact on classification!

True positive rate = $\boxed{\frac{TP}{TP + FN}}$

False positive rate = 1 - Specificity = $\boxed{\frac{FP}{FP + TN}}$

Different thresholds!



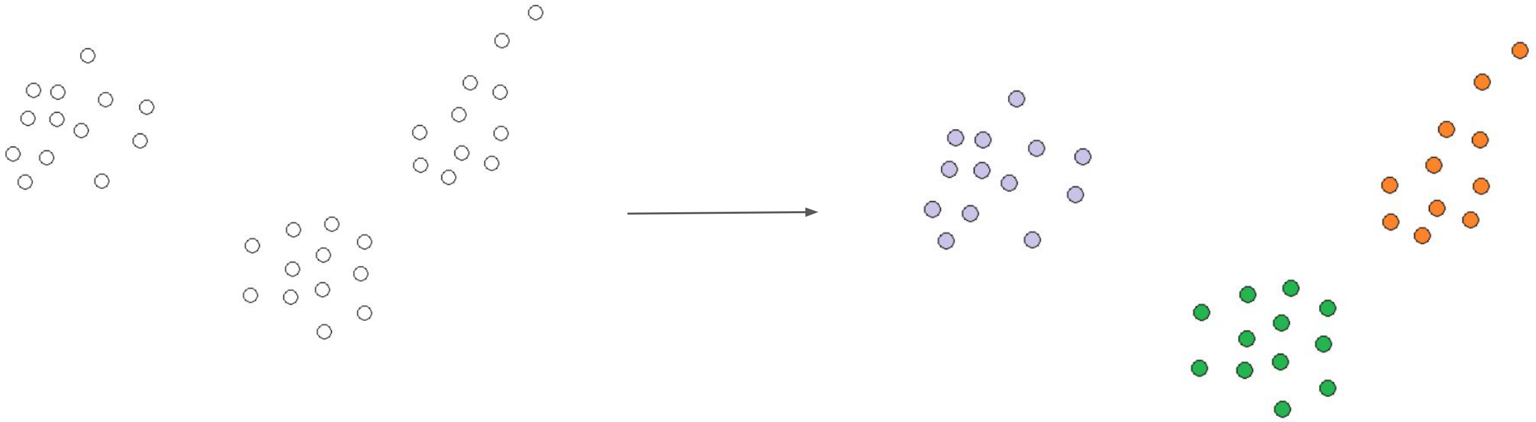


HW questions

1. For a given classifier, obtain its confusion matrix (Iris data set, 70:30 split)
2. Compute the Precision and Recall for the most frequent class
3. Compute F1
4. Compute the precision of the Majority classifier. What do you observe?

Unsupervised learning: clustering

End-goal



Clustering

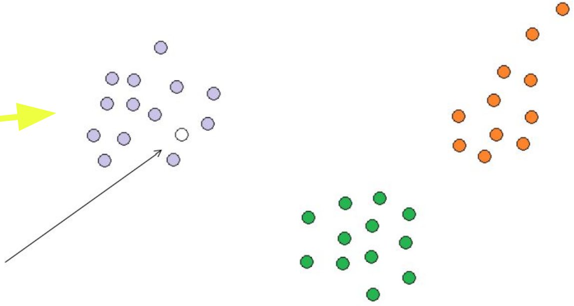
… is the process of grouping the data instances into clusters so that objects within a cluster have high similarity but are very dissimilar to objects in other clusters.

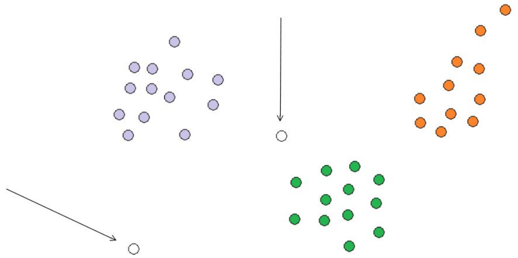
Wish list:

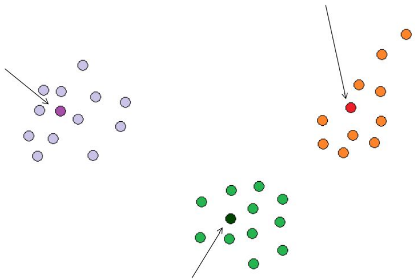
- Identity clusters irrespective of their shapes
- Scalability
- Ability to deal with noisy data
- Insensitivity to the order of input records

Some applications

1. Label assignment (cluster-based classification
2. Data summarization (e.g., via centroids, medoids)
3. Outlier detection

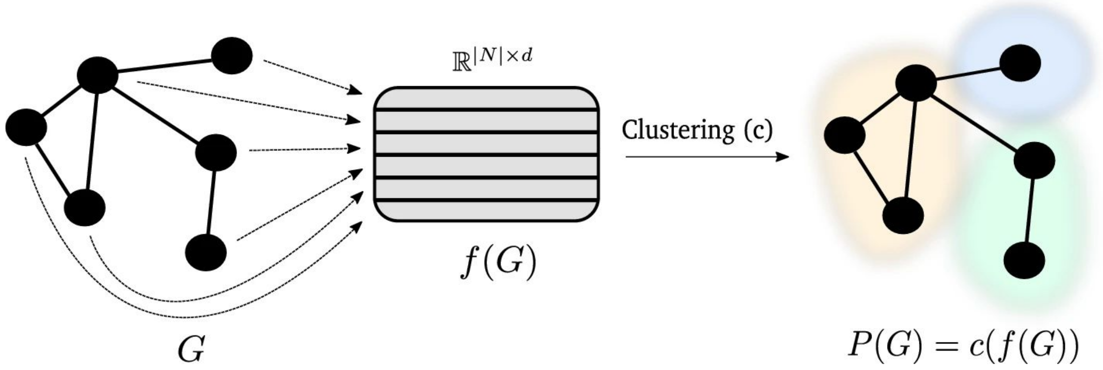






More applications

1. Customer segmentation and collaborative filtering
2. Text applications
3. Social network analysis



https://link.springer.com/article/10.1007/s10994-020-05882-8/figures/1

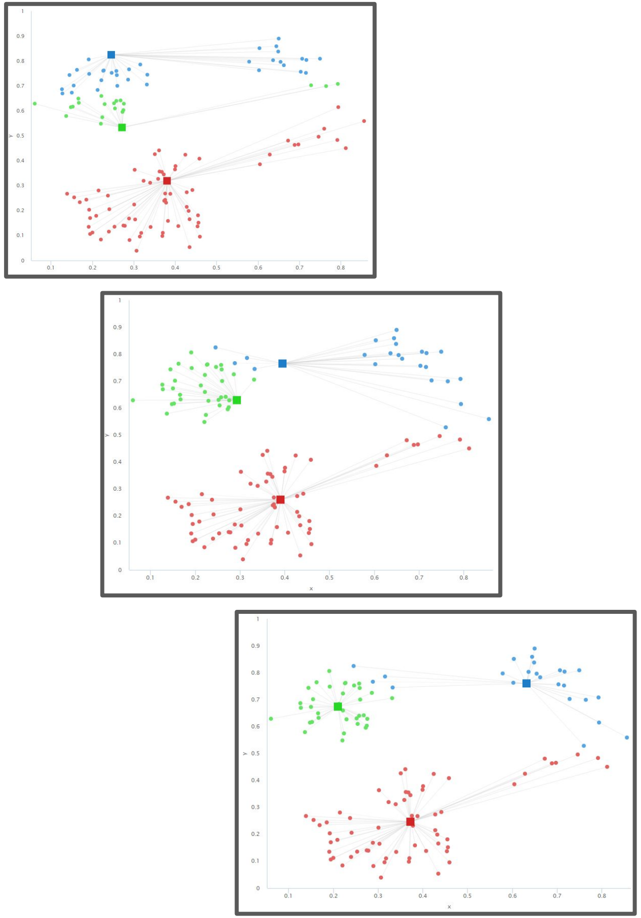
Clustering algorithm types

- Partitioning-based
  - K-means, k-medoids, k-modes
- Hierarchical
  - Agglomerative
- Grid-based
  - Multi-resolution grid structure
  - Efficient and scalable
- Density based
  - Low/high density regions of space (DBSCAN, OPTICS, DenClue)

k-Means

1. Choose **k random instances** as cluster centers
2. Assign each instance to its **closest cluster** center
3. Re-compute cluster centers by computing the average (*aka centroid*) of the instances pertaining to each cluster
4. If cluster centers have moved, go back to Step 2
5. (Equivalent termination criterion: stop when assignment of instances to cluster centers has not changed)

Alternatives: *K-medoids, K-modes*



Some properties of k-Means

- The number of clusters **k** is fixed in advance
- It is fast, it always converges
- Can converge into a local minima (bad solution because of unlucky start)
- Finds “spherical” shaped clusters
- k-Means will cluster the data even if it can’t be clustered (e.g. data that comes from uniform distributions)

How many clusters?

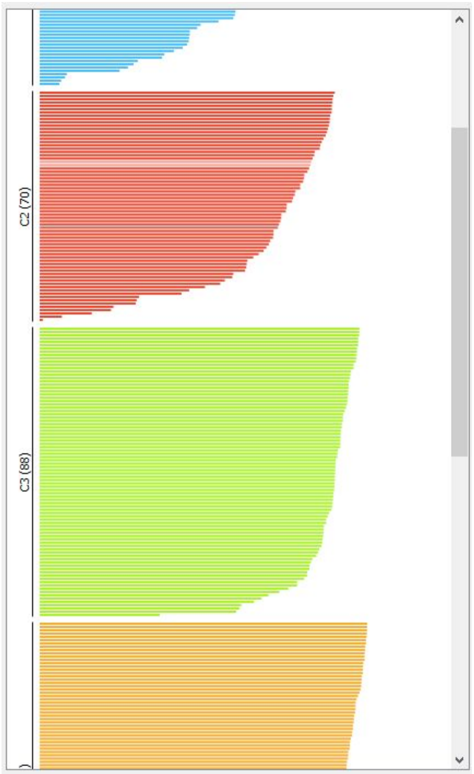
The silhouette value is a measure of how similar an object is to its own cluster (cohesion) compared to other clusters (separation).

- For example xi , its silhouette coefficient is $s_i = (b_i - a_i) / \max(a_i, b_i)$

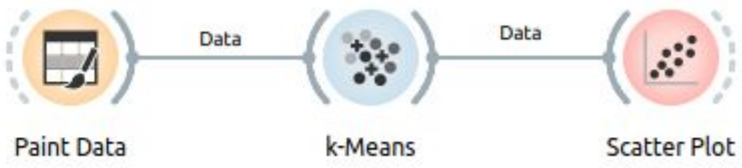
  ai average distance between xi to all other examples in its cluster.

  bi average distance between xi to the examples in the “closet neighboring” cluster

The overall silhouette coefficient is the average of the data point-specific coefficients.

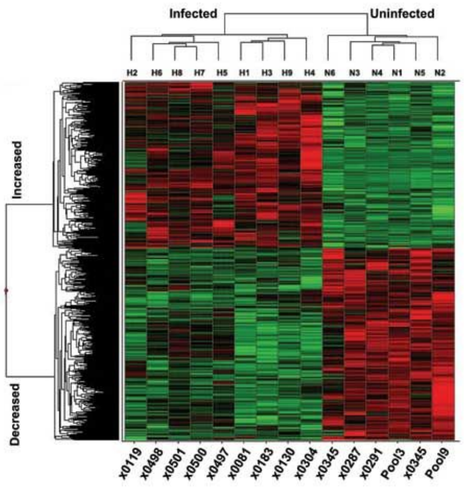


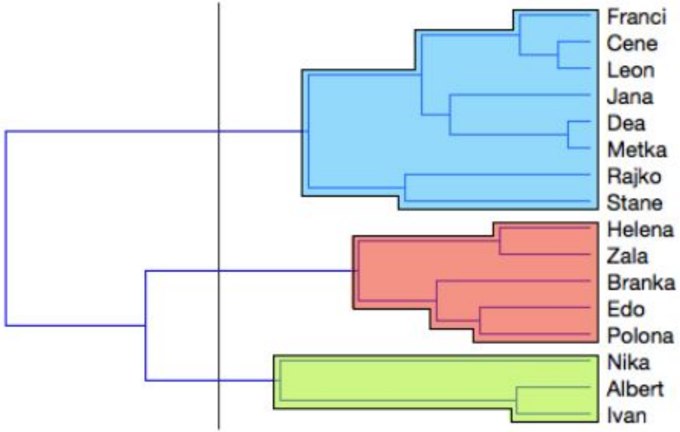
Practice: Number of clusters and custom data



Agglomerative clustering

- Start with a collection C of n singleton clusters
  - Each cluster contains one data point ci ={xi}
- Repeat until only one cluster is left:
  - Find a pair of clusters that is closest: min D(ci, cj)
  - Merge the clusters ci and cj into ci+j
  - Remove ci and cj from the collection C, add ci+j





Discuss: time&space complexity?

Linkage functions and metrics

Two main parameters: The metric (e.g., Manhattan, Euclidean etc.),

and **linkage:**

- **Single Linkage**
- **Average Linkage**
- **Ward Linkage**
- **Complete Linkage**

$$D(X,Y) = \min_{x \in X, y \in Y} d(x,y)$$

$$D(X,Y) = \frac{1}{N_X \times N_Y} \sum_{i=1}^{N_X} \sum_{j=1}^{N_Y} d(x_i, y_j);$$
$$x_i \in X,\ y_i \in Y,$$

$$ESS(X) = \sum_{i=1}^{N_X} \left| x_i - \frac{1}{N_X} \sum_{j=1}^{N_X} x_j \right|^2$$

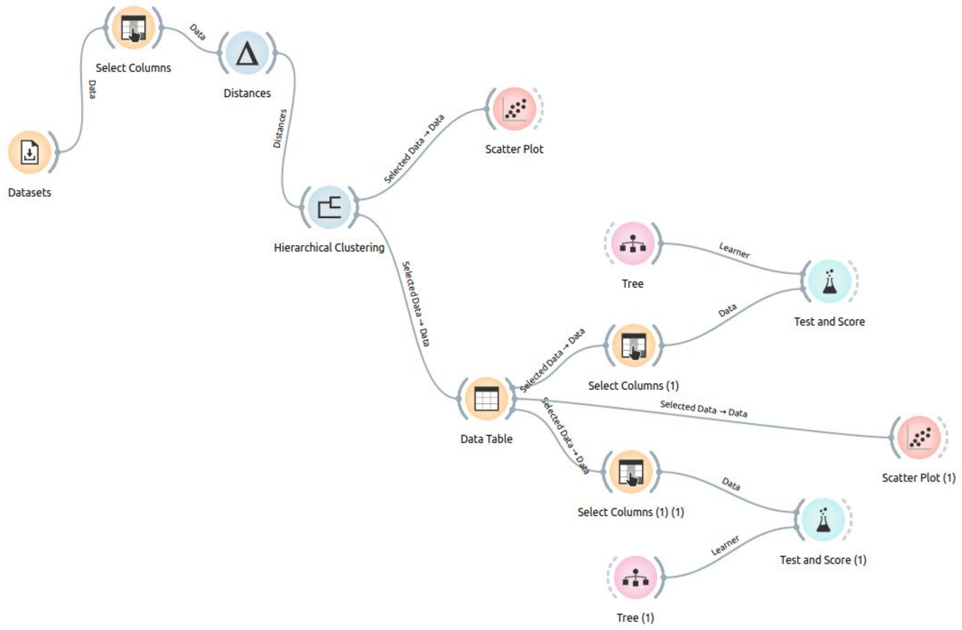
$$D(X,Y) = ESS(XY) - [ESS(X) + ESS(Y)]$$

$$D(X,Y) = \max_{x \in X, y \in Y} d(x,y)$$

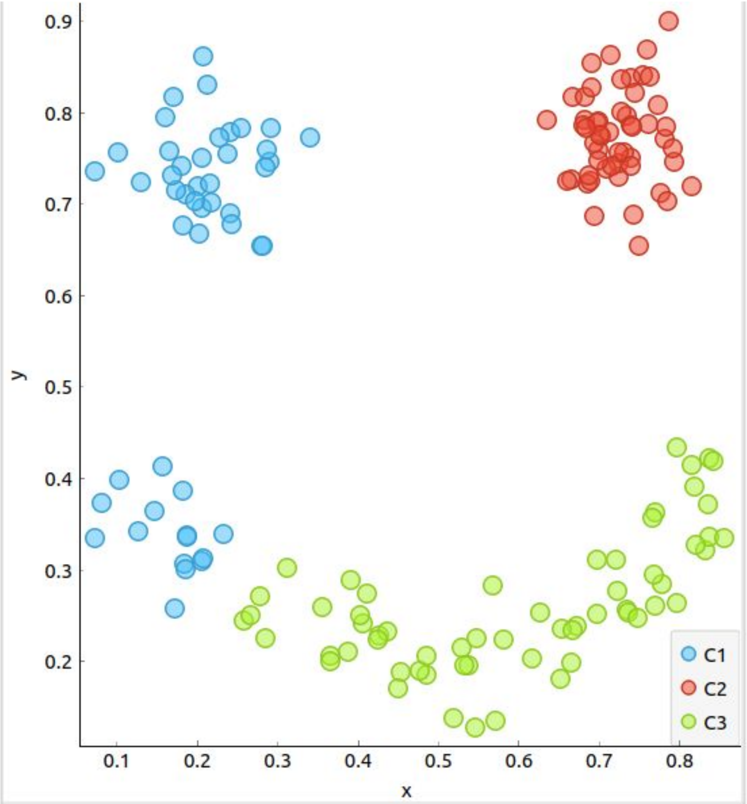
Find more details at: https://www.statistics.com/glossary/linkage-function/

Practice

- Language bias of clustering algorithms
- The **clustering hypothesis**



Practice - algorithmic language bias



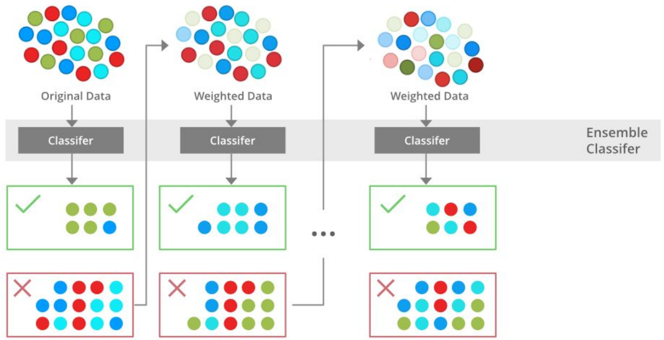
Interesting things to try

1. Write your version of Single-linkage Hierarchical Clustering. Try to skip a step or two during cluster generalization. What do you observe?
2. Implement k-medoids algorithm and try to estimate cluster uncertainty by performing multiple initializations (and implementing their combination)

State-of-the-art classification

Extreme Gradient Boosting

1. Decision tree ensembles
2. Build a tree
    a. Identify where loss is highest
    b. Use this information when building subsequent trees
3. Final ensemble of trees used for classification



Python Notebook

1. Benchmark:
   a. Tree
   b. Extreme gradient boosting
2. Accuracy