MEDNARODNA
PODIPLOMSKA ŠOLA
JOŽEFA STEFANA

INFORMATION AND COMMUNICATION TECHNOLOGIES
Master study programme

# Data and Text Mining

Petra Kralj Novak

October 23, 2019

http://kt.ijs.si/petra_kralj/dmkd.html

# Data and Text Mining

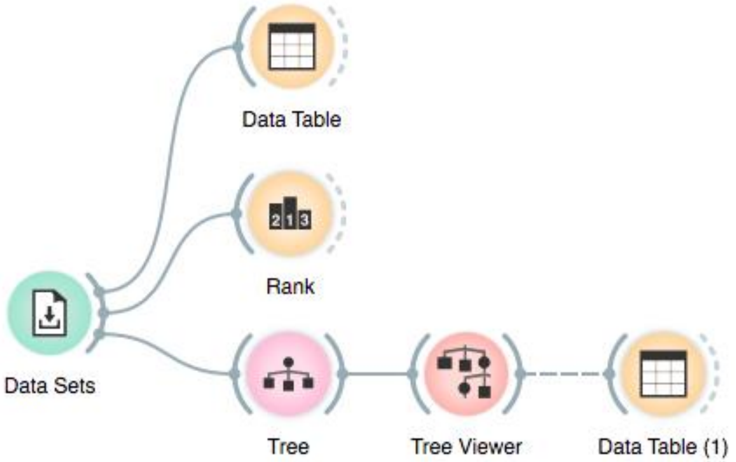
Course scope:

| | |
|---|---|
| - Data preprocessing | Prof. dr. Bojan Cestnik |
| - Data mining | Prof. dr. Nada Lavrač |
| | Doc. dr. Petra Kralj Novak |
| - Text Mining | Prof. dr. Dunja Mladenić |
| | Erik Novak |

Literature: Max Bramer: Principles of data mining (2007)

- Skip Chapter 5
- Additional material on selected topics

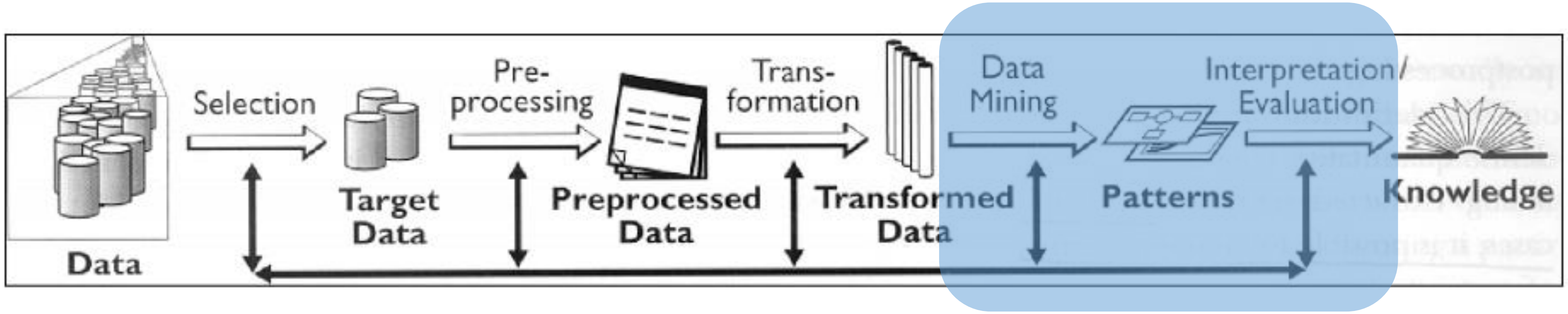
- Theory and exercises
- Hands-on orange
  - Open source machine learning and data visualization
  - Interactive data analysis workflows with a large toolbox
  - Visual programming
- Machine learning in Python with **scikit-learn**
  - The gold standard of Python machine learning
  - Simple and efficient tools for data mining and data analysis
  - Well documented



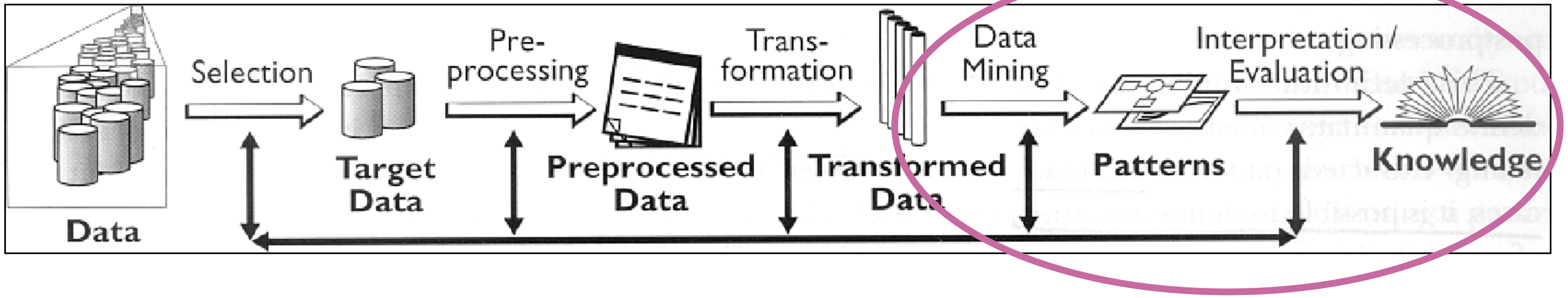

```
# ---------------------------------------------------------------------------------------------
print("Train and test classification models")
classifiers = [
    # ("Naive Bayes", naive_bayes.MultinomialNB()),
    ("Logistic regression", linear_model.LogisticRegression(C=1e5, solver='lbfgs', multi_class='multinomial', max_iter=600)),
    ("MultinomialNB", MultinomialNB()),
    ("SVC", svm.LinearSVC()),
    ("SVC-RBF", svm.SVC(gamma='scale', decision_function_shape='ovo'))]


for name, classifier in classifiers:
    classifier.fit(train_data, y_train)
    predictions = classifier.predict(test_data)
    classifier.confusion_matrix = metrics.confusion_matrix(predictions, y_test, labels=["negative", "neutral", "positive"])
    classifier.accuracy = metrics.accuracy_score(predictions, y_test)
    print(name, classifier.accuracy, "\n Confusion matrix: \n", classifier.confusion_matrix)
    pickle_clf(classifier, path="./models/"+name+".pkl")
```

# KDD vs. ML/DM

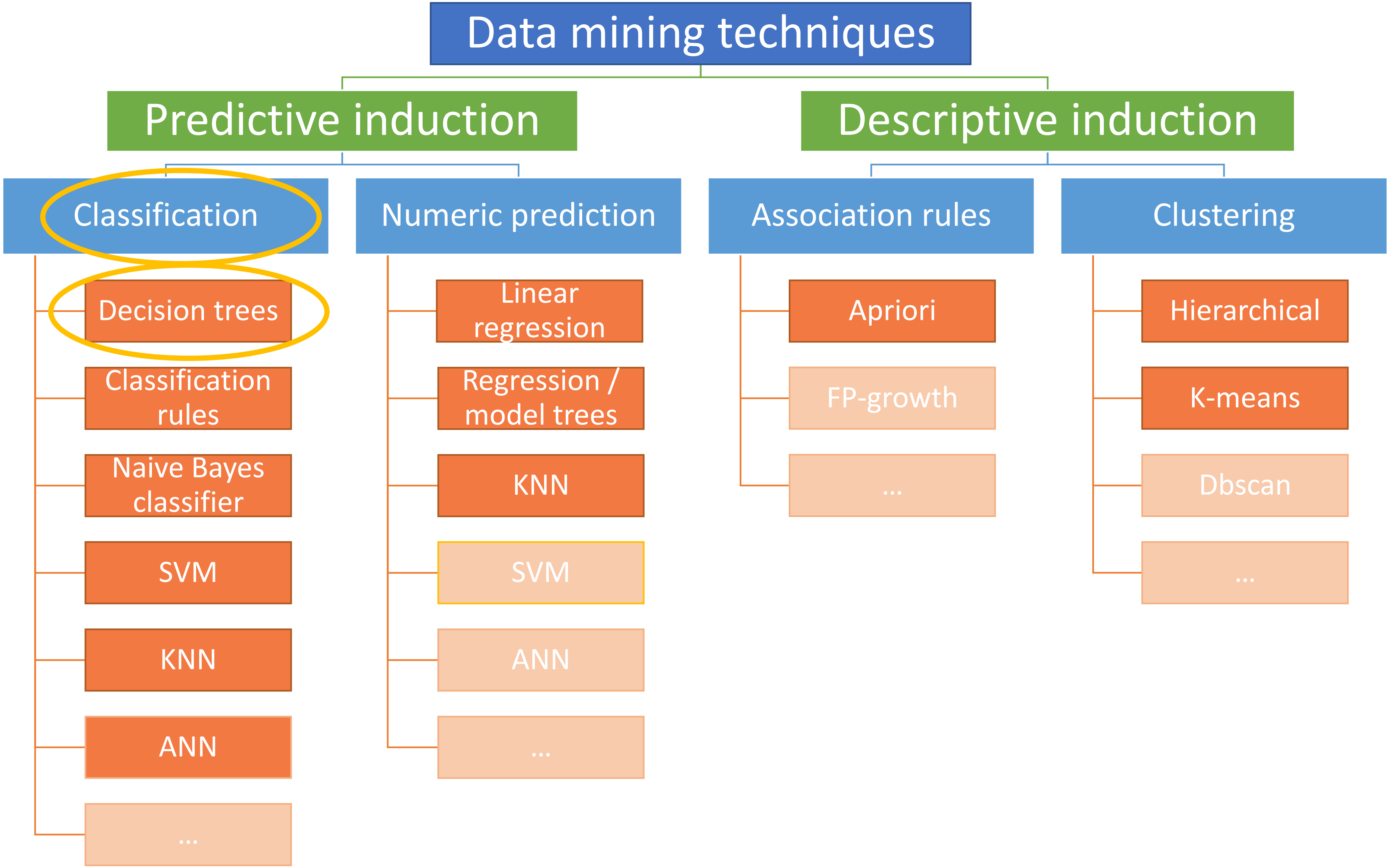
- Knowledge Discovery from Databases vs. Machine Learning/Data Mining



# Keywords




- Data
  - Attribute, example, attribute-value data, target variable, class, discretization, market basket data
- Algorithms
  - Decision tree induction, ID3, entropy, information gain, overfitting, Occam's razor, model pruning, naïve Bayes classifier, KNN, association rules, support, confidence, classification rules, Laplace estimate, numeric prediction, regression tree, model tree, hierarchical clustering, dendrogram, k-means clustering, centroid, DB-scan, silhouette coefficient, Apriori, heuristics vs. exhaustive search, predictive vs. descriptive DM, language bias, artificial neural networks, deep learning, backpropagation,…
- Evaluation
  - Train set, test set, accuracy, confusion matrix, cross validation, true positives, false positives, ROC space, AUC, error, precision, recall, F1, MSE, RMSE, rRMSE, support, confidence


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
…

# Data for Data Mining

Max Bramer: Principles of data mining (2007)

Chapter 1: Data for Data Mining

# Types of attributes

- Categorical
  - Nominal (Colors: red, blue, green )
  - Binary (Gender: male, female)
  - Ordinal (Size: small, medium, large)
- Numerical
  - Integer (Number of car sits: 2, 5, …)
  - Real (Temperature in degrees: 21℃, 23.4℃,…)
  - Ordinal
  - Binary
- Complex types (time series, text, graphs, images, …)

# Mining complex data types

- Time series analysis
  - Financial time series, heart-rate monitoring,…
- Text mining
  - News, comments, Wikipedia, books, … for content, sentiment, style, word meaning…
- Graph mining
  - Maps, molecules, citation networks, hyperlinks, …. for classification, patterns,…
- Social media mining (graphs + text)
  - Facebook, Twitter, …. Information spreading, hate speech…
- Images
  - Image classification

# Classification

# Classification problem

- Goal: Assign each example a category
  - Magazine reader (or not)
  - Patients at risk for acquiring a certain illness
  - A patient needing antibiotics (or not)
  - Customers who are likely buyers
  - People who are likely to vote for a political party
  - Churn prediction
  - …

# Classification problem

- Goal: Identifying to which one of a number of mutually exhaustive and exclusive categories (known as classes) an object belongs to.

  - Given a dataset of examples (described by attributes).
  - The target variable is a attribute that we are interested in predicting. In classification, the target is categorical.
  - The values of the target variable are called classes.
  - We train a model on the data that will predict the classes of new examples as accurately as possible.

# Attribute-value data for classification

(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

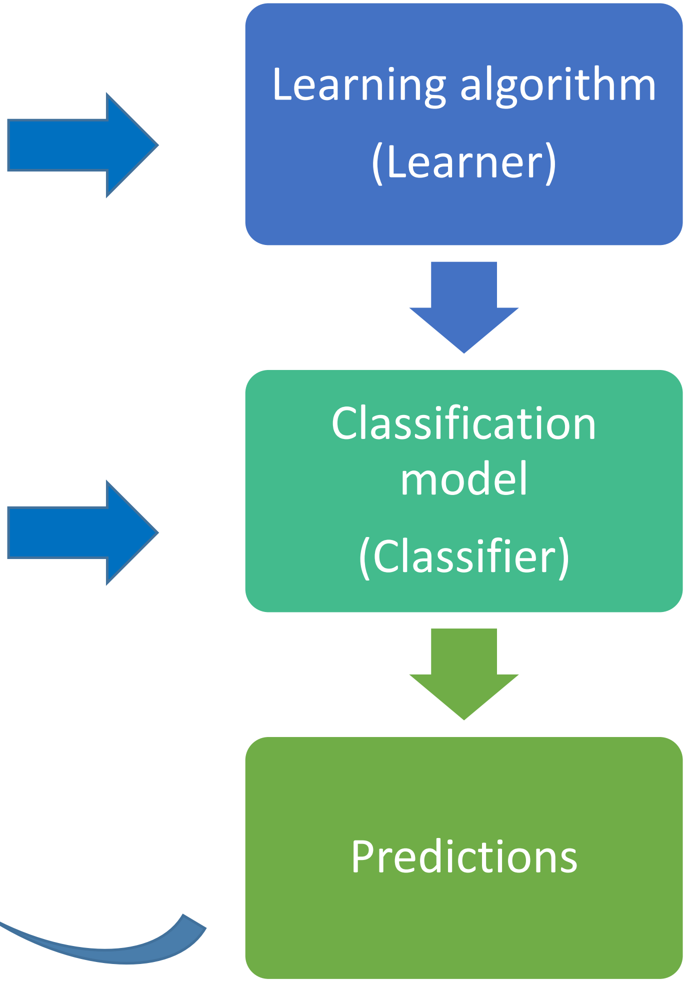
# The basic classification schema

| Št | Atrib1 | Atrib2 | Atrib3 | Clasa |
|---|---|---|---|---|
| 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 | Clasa |
|---|---|---|---|---|
| 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?

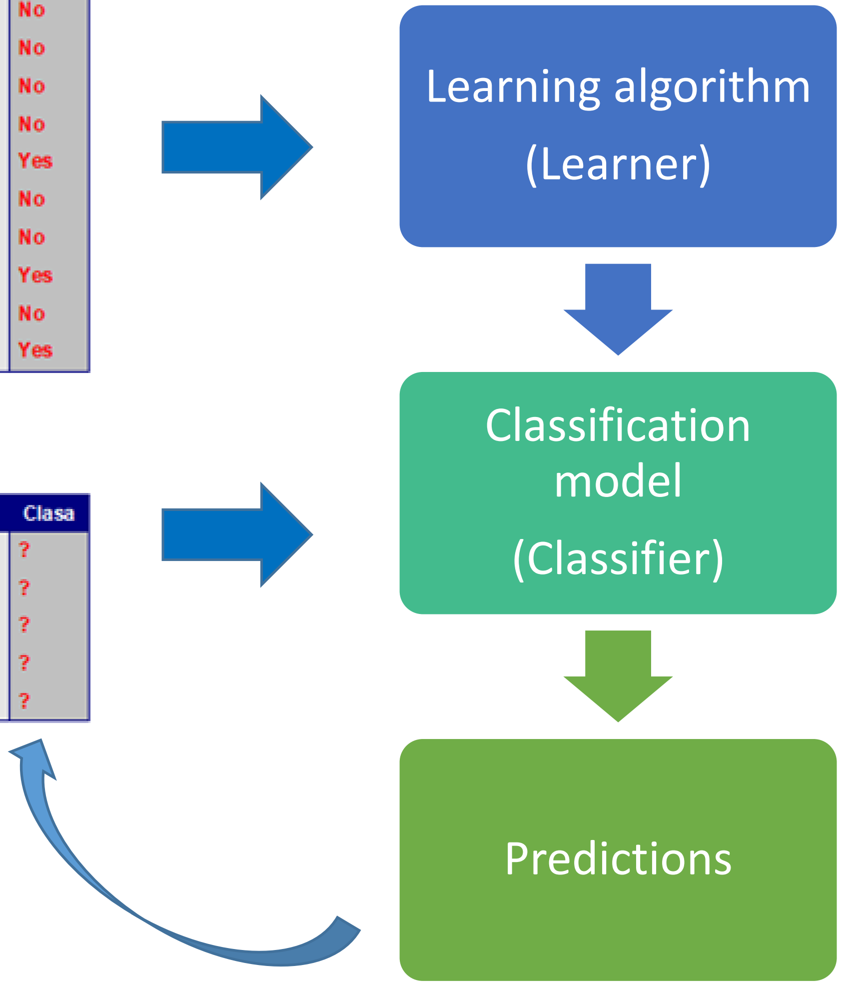
# The basic classification schema

| Šr | Atrib1 | Atrib2 | Atrib3 | Clasa |
|---|---|---|---|---|
| 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**

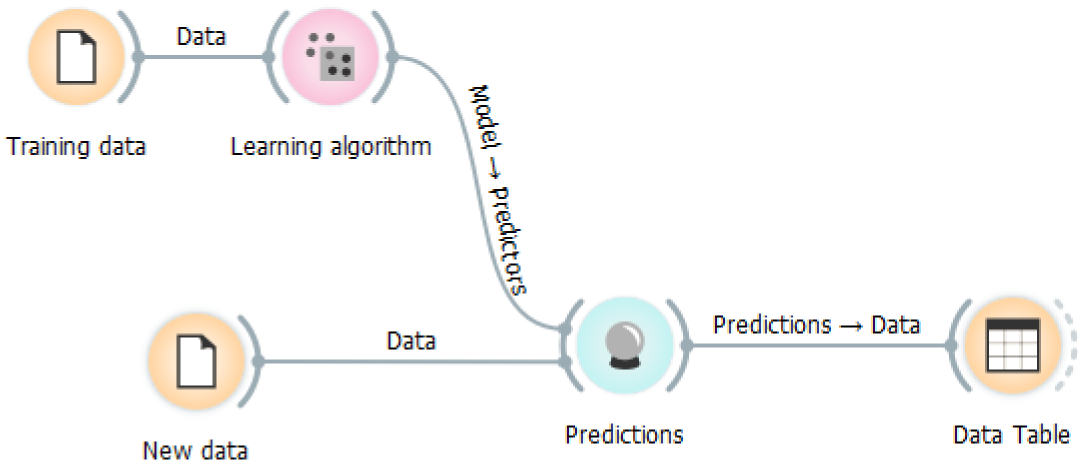
| Šr | Atrib1 | Atrib2 | Atrib3 | Clasa |
|---|---|---|---|---|
| 11 | No | Small | 55K | ? |
| 12 | Yes | Medium | 80K | ? |
| 13 | Yes | Large | 110K | ? |
| 14 | No | Small | 95K | ? |
| 15 | No | Large | 67K | ? |

**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 $Y_p$.

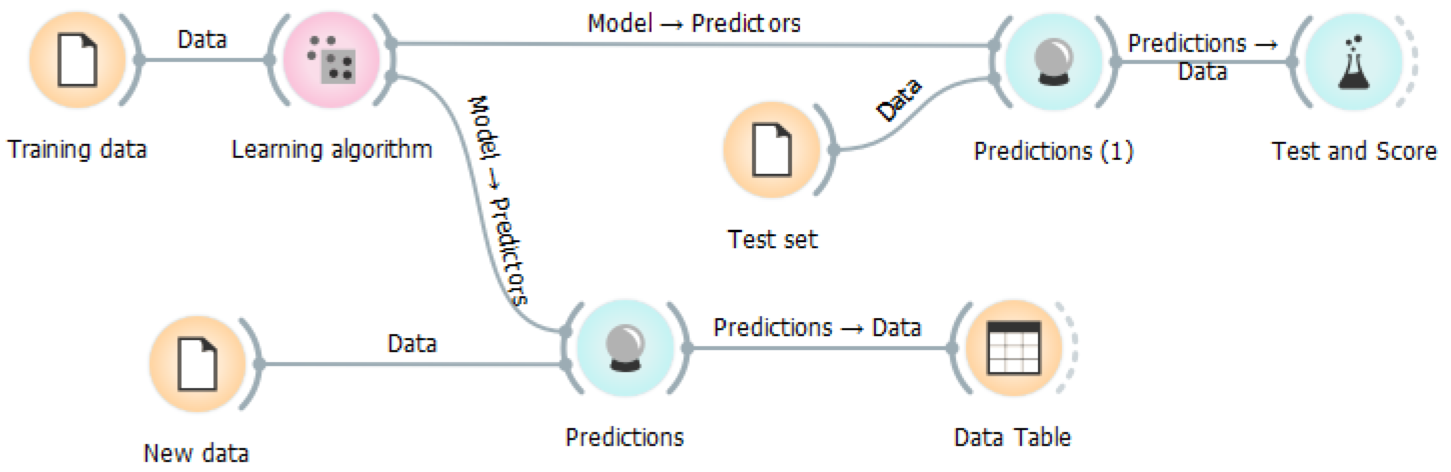
# Basic classification schema in Orange




- 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)
  - ....

# Classification with evaluation

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



# Example: “titanic” dataset

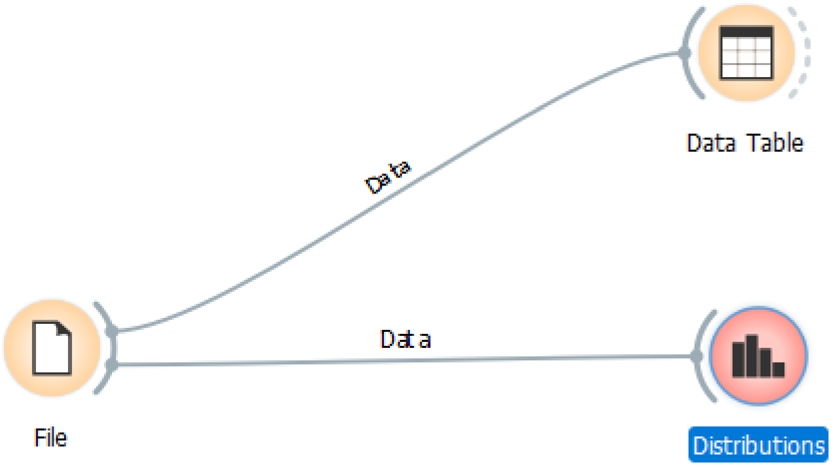
**Target variable**

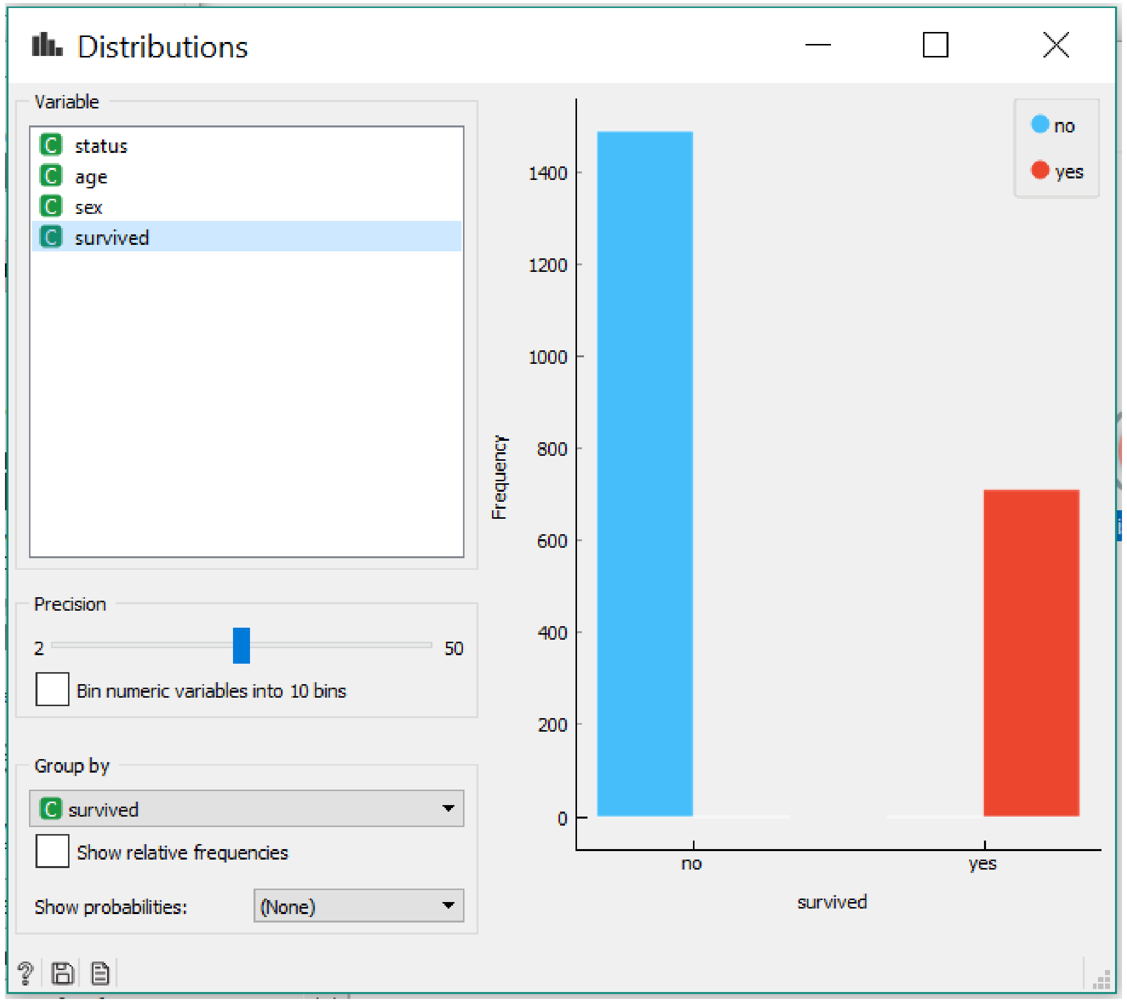
Attributes

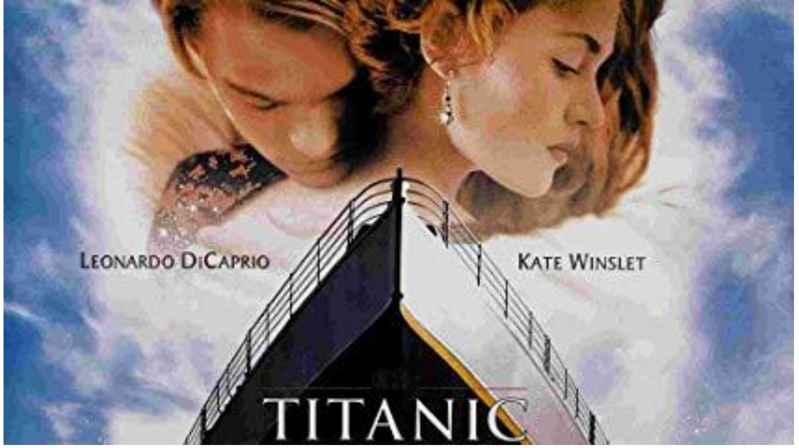
Examples

| | 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 |

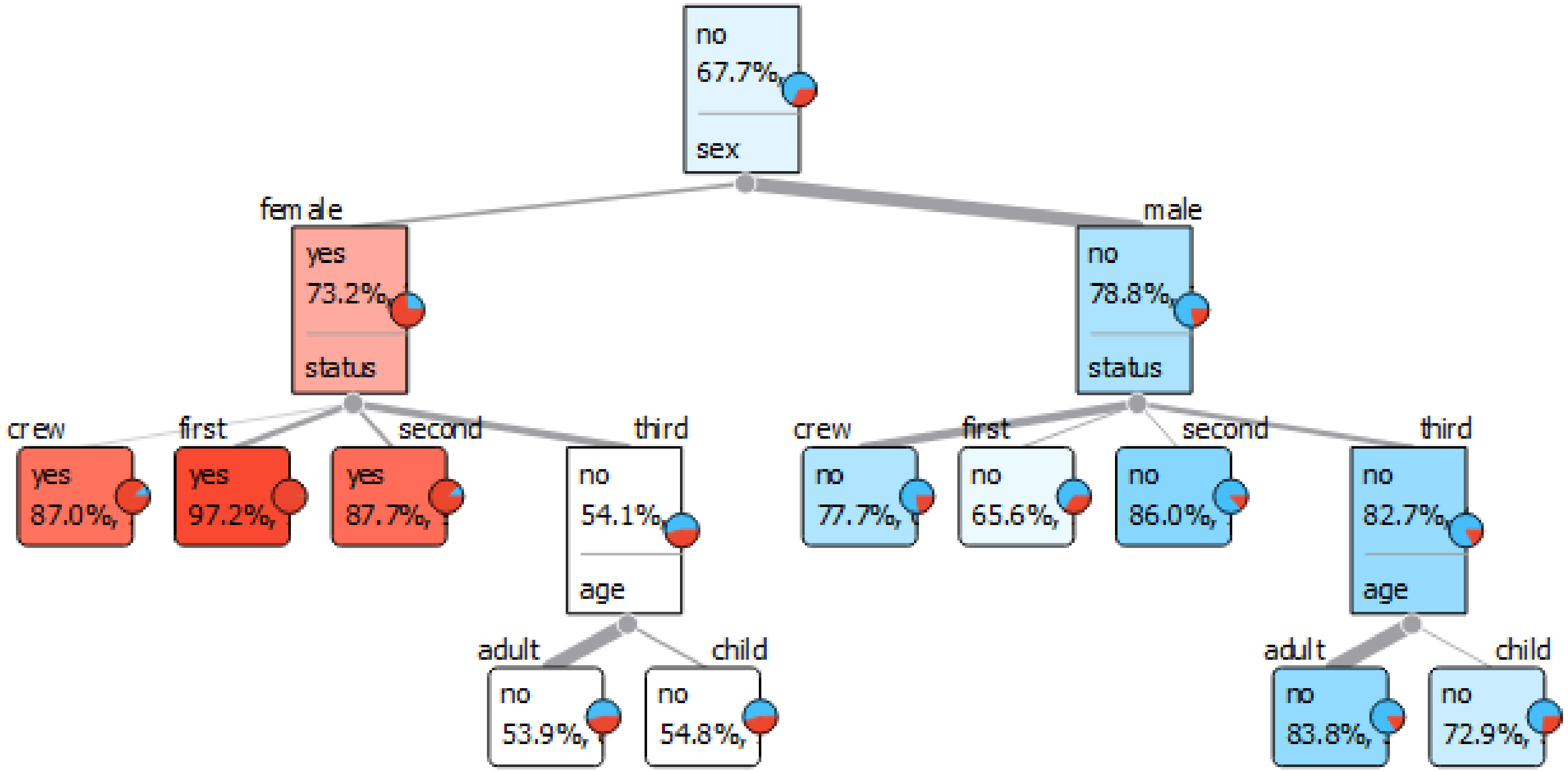
# Classification: distribution of the target variable



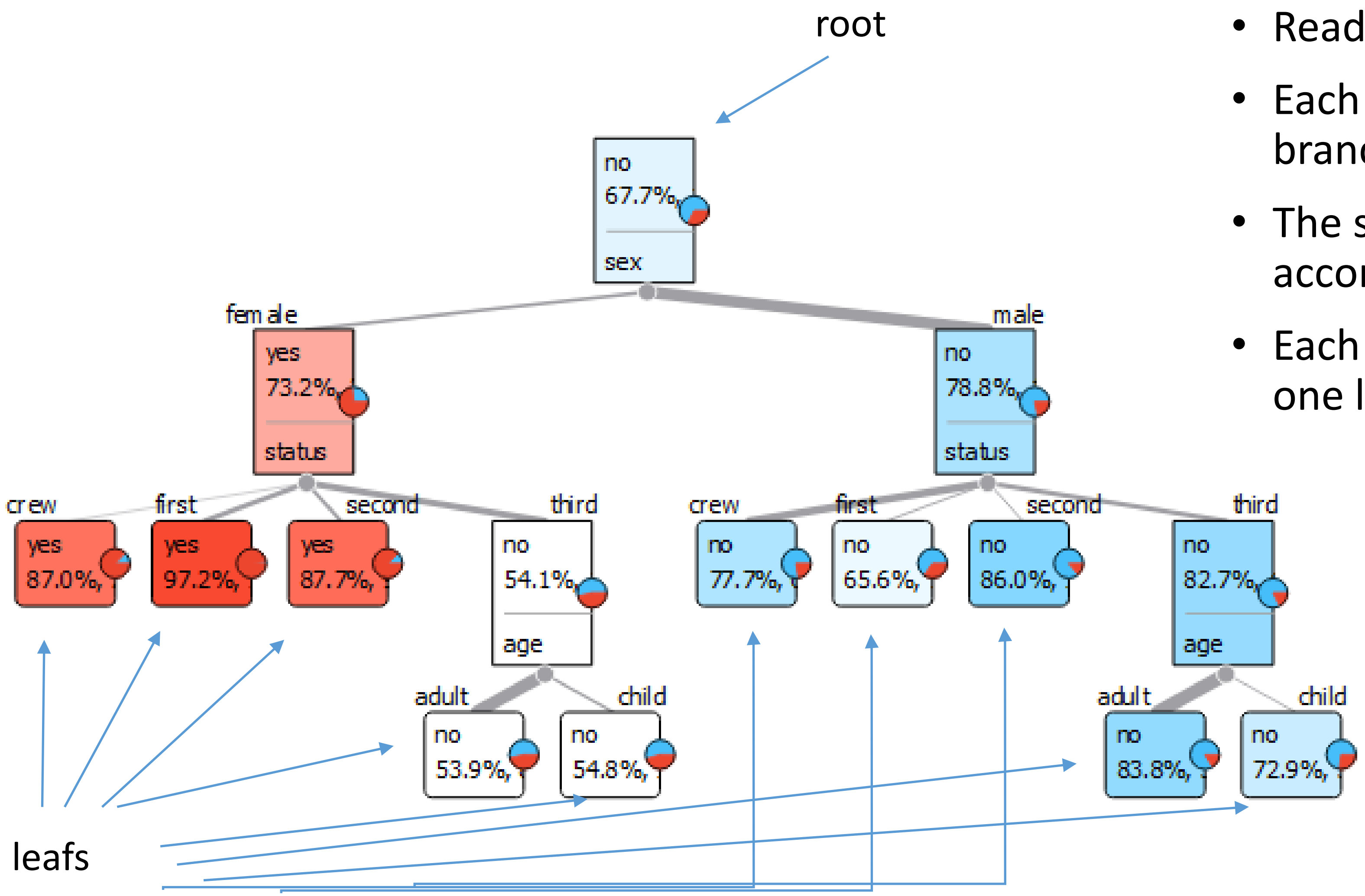






# Who survived on the Titanic?

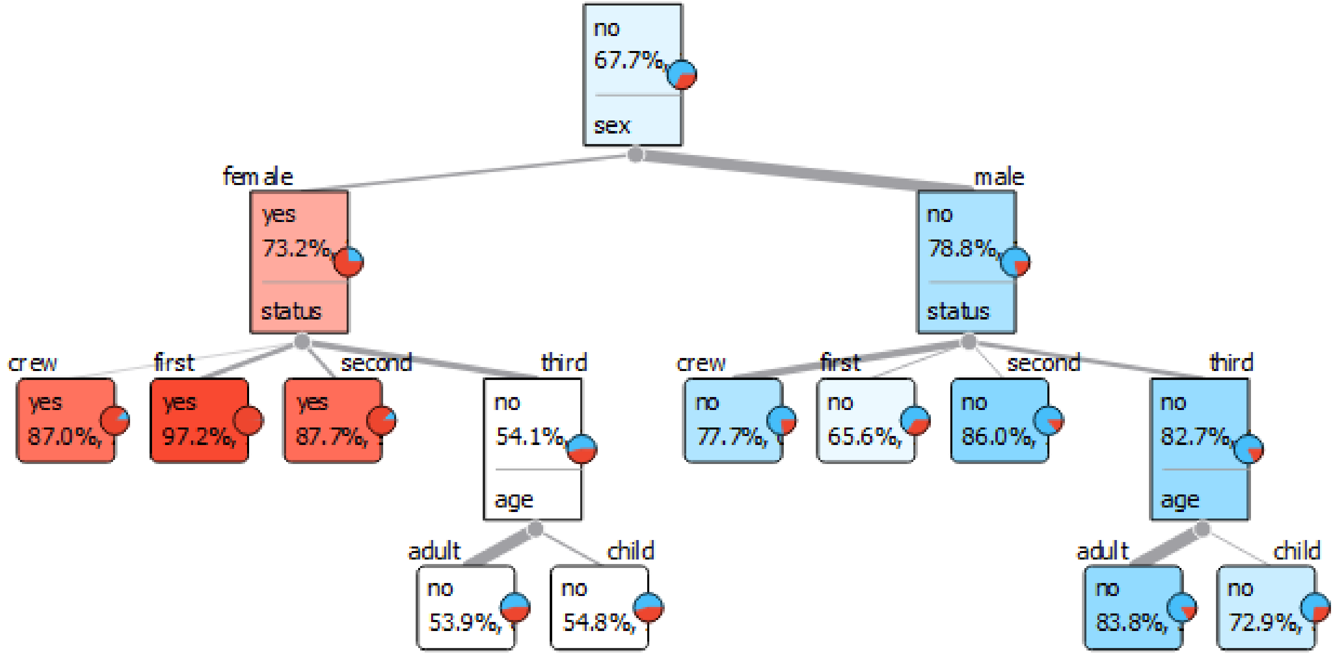



# Decision tree



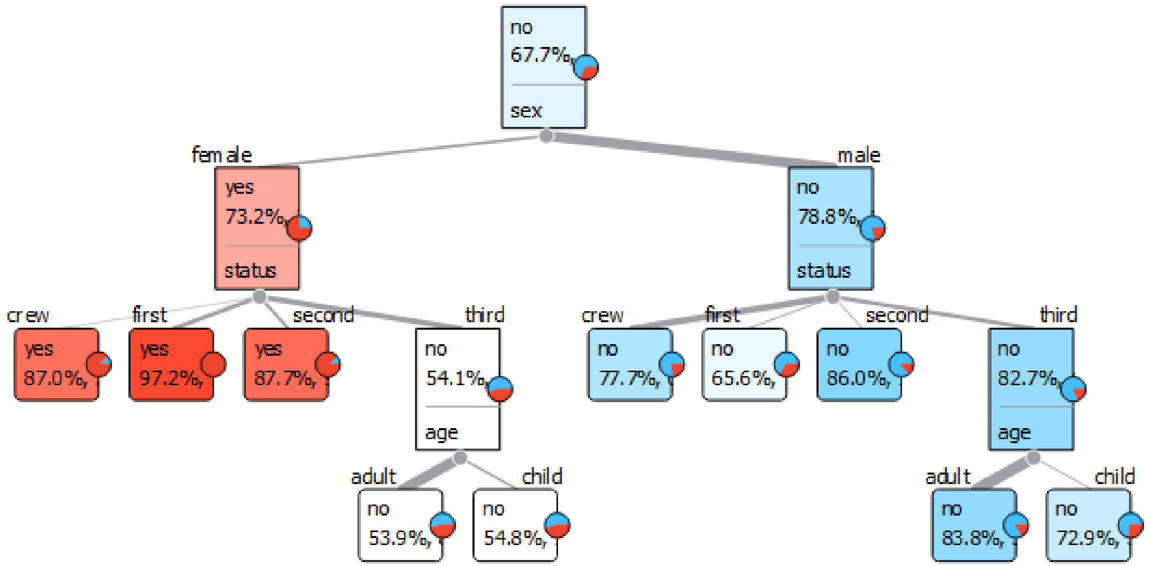

- Read top-down
- Each node is an attribute which branches according to its values
- The set of examples splits according to attribute values
- Each example end up in exactly one leaf

# Exercise: Classify the data instances



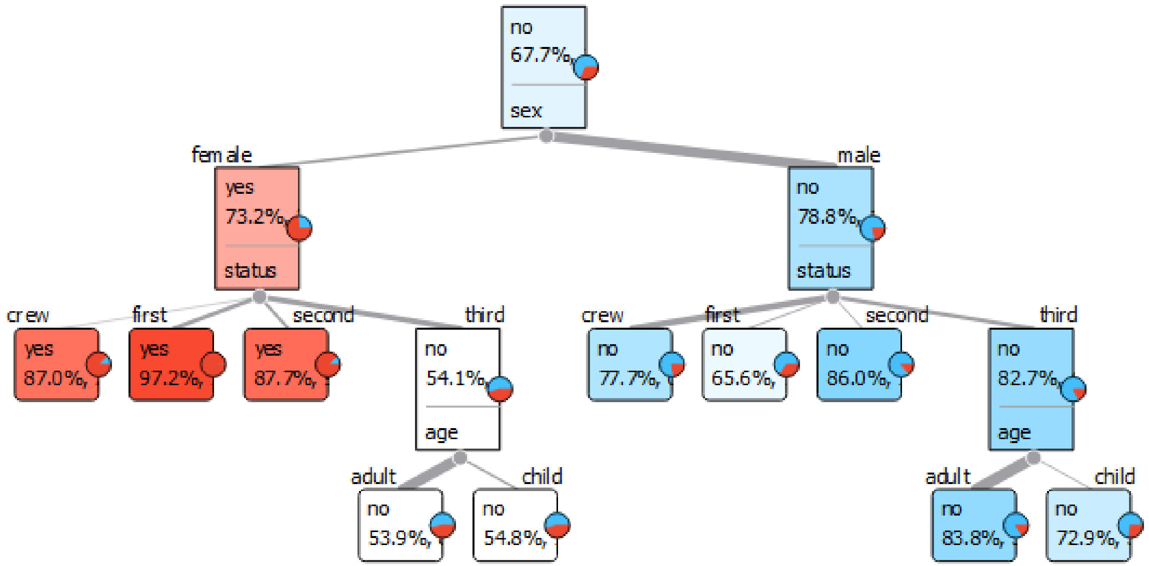

| | status | age | sex | survived? |
|---|---|---|---|---|
| 1 | third | child | male | |
| 2 | third | child | female | |
| 3 | crew | adult | male | |
| 4 | first | adult | male | |
| 5 | second | adult | male | |
| 6 | third | adult | male | |
| 7 | first | adult | female | |
| 8 | second | adult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

# We can rewrite the tree as a set of rules




- One rule for each leaf

# We can rewrite the tree as a set of rules

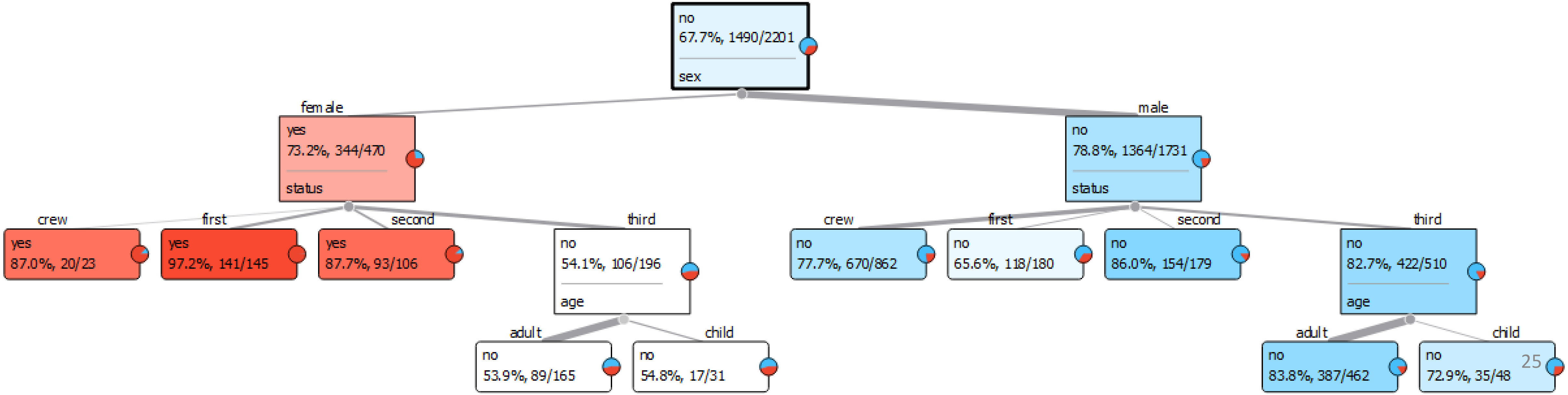



- sex = female & status = crew → survived = yes
- sex = female & status = first → survived = yes
- sex = female & status = second → survived = yes
- sex = female & status = third & age = adult → survived = no
- sex = female & status = third & age = child → survived = no
- sex = male & status = crew →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

- Rule: a path from root leaf
- Each example *fires* exactly one rule

# We can interpret decision trees

- Which is the most informative attribute?
- Visualization in orange:
  - The number of examples in each node
  - Percentage of examples belonging to the majority class
  - Colour intensity = certainty of the prediction
  - Thickness of the branch proportional to the number of examples



# TDIDT
# Top Down Induction of Decision Trees

# TDIDT – Top Down Induction of Decision Trees

- We induce decision trees top-down
- There is many possible decision trees for a given dataset
- It is very important which attribute we choose as the root
- Heuristic: we choose the attribute which **best separates** the classes

↑

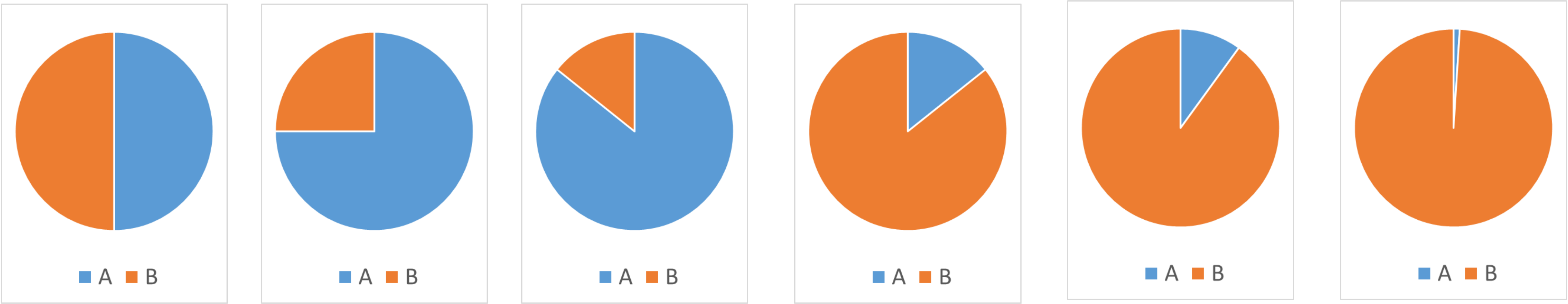
Information gain

↑

Entropy

# Entropy

- Entropy (information theory) is a measure of uncertainty.




½ ¼ 1/7 6/7 9/10 99/100

# Entropy

$$E(S) = -\sum_{c=1}^{N} p_c \cdot \log_2 p_c$$

- Calculate:

  E (0 , 1) = 0

  E (1/2 , 1/2) = 1

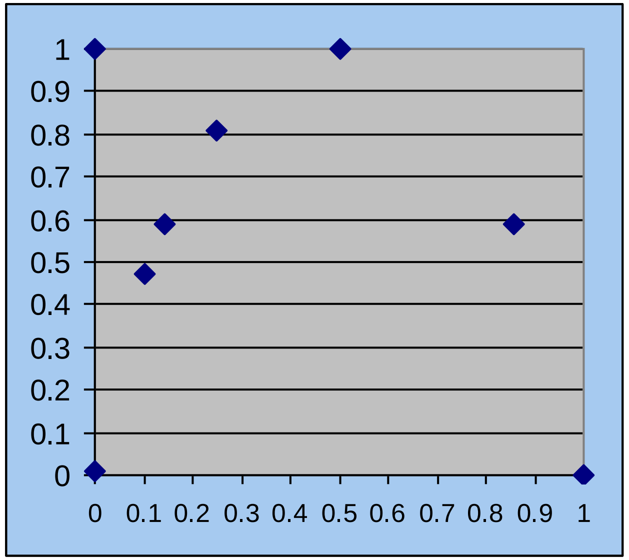
  E (1/4 , 3/4) = 0.81

  E (1/7 , 6/7) = 0.59

  E (6/7 , 1/7) = 0.59

  E (0.1 , 0.9) = 0.47

  E (0.001 , 0.999) = 0.01



# Entropy

$$E(S) = -\sum_{c=1}^{N} p_c \cdot \log_2 p_c$$

- Calculate:

  E (0 , 1) = 0

  E (1/2 , 1/2) = 1

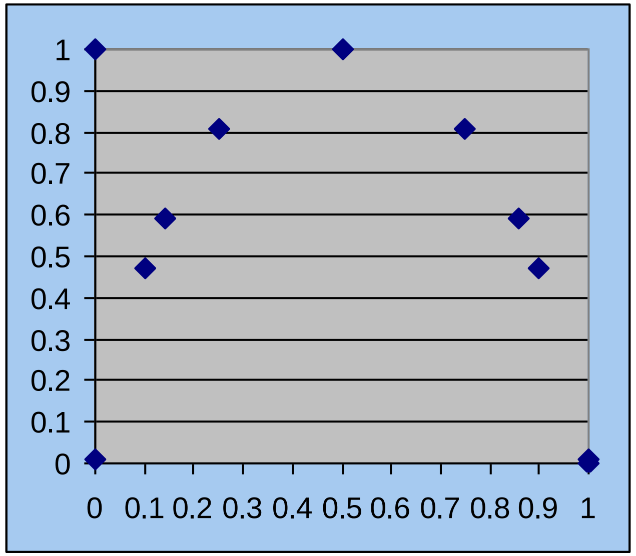
  E (1/4 , 3/4) = 0.81

  E (1/7 , 6/7) = 0.59

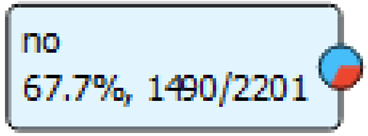
  E (6/7 , 1/7) = 0.59

  E (0.1 , 0.9) = 0.47

  E (0.001 , 0.999) = 0.01



# Example: entropy of a dataset




Titanic survivers

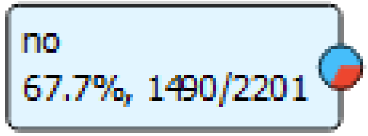
- All passengers: 2201
- Survivers: 721

$$E(S) = -\sum_{c=1}^{N} p_c \cdot \log_2 p_c$$

- The entire dataset 2201 instances
- 1490 classified NO
- 721 classified YES

We compute the entropy

# Example: entropy of a dataset




Titanic survivers

- All passengers: 2201
- Survivers: 721

$$E(S) = -\sum_{c=1}^{N} p_c \cdot \log_2 p_c$$

- The entire dataset 2201 instances
- 1490 classified NO
- 721 classified YES

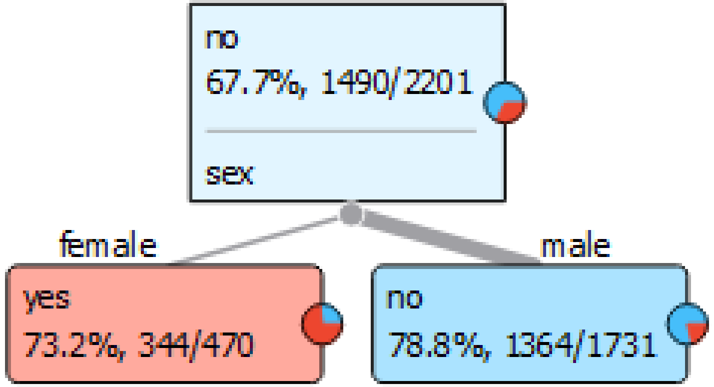
We compute the entropy

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

# Information gain (of an attribute)

Information gain (IG) measures how much "information" a feature gives us about the class.

= How much the entropy is reduced by splitting the data according to the attribute



# Information Gain

number of examples in the subset $S_v$

(probability of the branch)

set S

attribute A

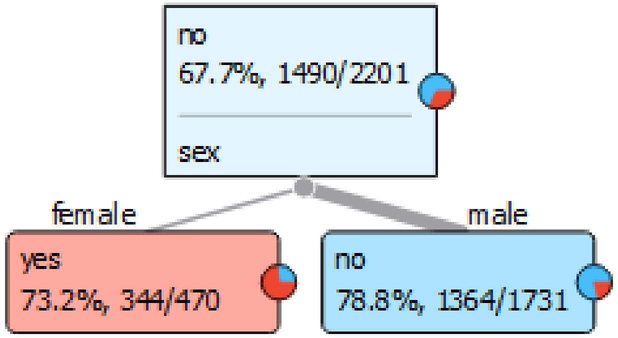
$$Gain(S, A) = E(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} \cdot E(S_v)$$

Entropy of the set S

Entropy of the subset $S_v$

number of examples in set S

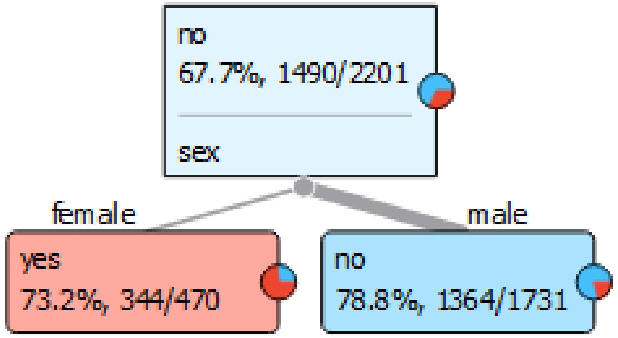
# Information gain: example




1. Compute the entropy of the entire set
2. The attribute “sex” splits the dataset into two subsets :
   - **female** with 470 instances (344 survived)
   - **male** with 1731 instances (1364 died)
3. Compute the entropy of each subset
4. Compute the Information gain

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

# Information gain: example

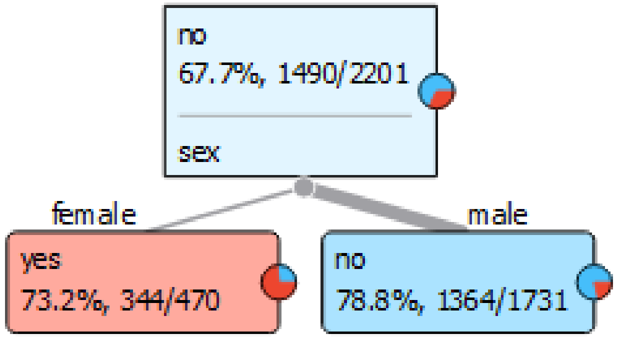



1. Compute the entropy of the entire set
2. The attribute “sex” splits the dataset into two subsets :
   - **female** with 470 instances (344 survived)
   - **male** with 1731 instances (1364 died)
3. Compute the entropy of each subset
4. Compute the Information gain

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

| | NO | YES | total |
|---|---|---|---|
| | 1490 | 720 | 2210 |
| | | | |
| Class probability pi | 0,674 | 0,326 | |
| | | | |
| pi * log (pi, 2) | -0,38 | -0,53 | |
| | | | |
| entropy | 0,911 | | |

# Information gain: example



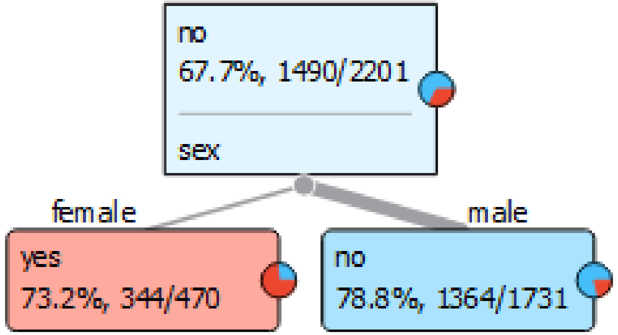

1. Compute the entropy of the entire set
2. The attribute “sex” splits the dataset into two subsets :
   - **female** with 470 instances (344 survived)
   - **male** with 1731 instances (1364 died)
3. Compute the entropy of each subset
4. Compute the Information gain

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

| 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 | | |

# Information gain: example




1. Compute the entropy of the entire set
2. The attribute “sex” splits the dataset into two subsets :
   - **female** with 470 instances (344 survived)
   - **male** with 1731 instances (1364 died)
3. Compute the entropy of each subset
4. Compute the Information gain

| 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 | | |

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

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

# TDIDT – Top Down Induction of Decision Trees

- We induce decision trees top-down
- There is many possible decision trees for a given dataset
- It is very important which attribute we choose as the root
- Heuristic: we choose the attribute which **best separates** the classes

Information gain

Entropy

# Decision tree induction
# ID3 Algorithm

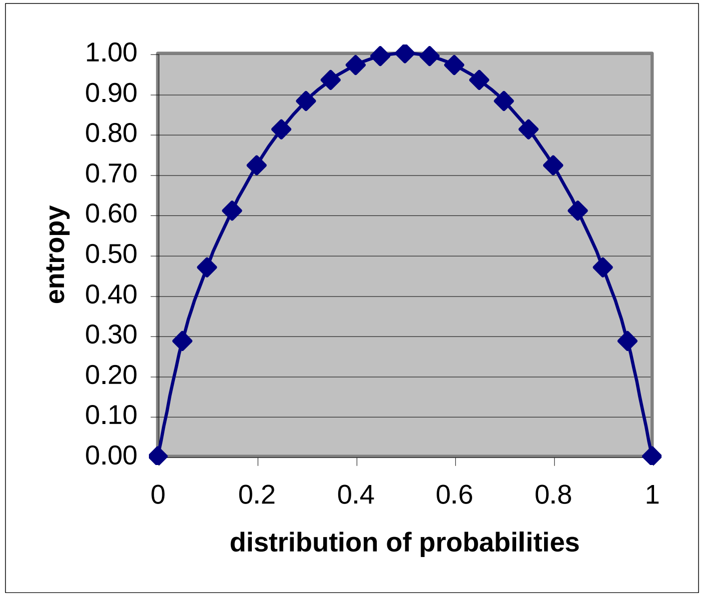
Induce a decision tree on set S:

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 $S_i$

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

# Entropy and information gain

| probability of class 1 $p_1$ | probability of class 2 $p_2 = 1-p_1$ | entropy $E(p_1, p_2) = -p_1 \cdot \log_2(p_1) - p_2 \cdot \log_2(p_2)$ |
|---|---|---|
| 0 | 1 | 0.00 |
| 0.05 | 0.95 | 0.29 |
| 0.10 | 0.90 | 0.47 |
| 0.15 | 0.85 | 0.61 |
| 0.20 | 0.80 | 0.72 |
| 0.25 | 0.75 | 0.81 |
| 0.30 | 0.70 | 0.88 |
| 0.35 | 0.65 | 0.93 |
| 0.40 | 0.60 | 0.97 |
| 0.45 | 0.55 | 0.99 |
| 0.50 | 0.50 | 1.00 |
| 0.55 | 0.45 | 0.99 |
| 0.60 | 0.40 | 0.97 |
| 0.65 | 0.35 | 0.93 |
| 0.70 | 0.30 | 0.88 |
| 0.75 | 0.25 | 0.81 |
| 0.80 | 0.20 | 0.72 |
| 0.85 | 0.15 | 0.61 |
| 0.90 | 0.10 | 0.47 |
| 0.95 | 0.05 | 0.29 |
| 1 | 0 | 0.00 |




number of examples in the subset

probability of the "branch"

attribut A

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

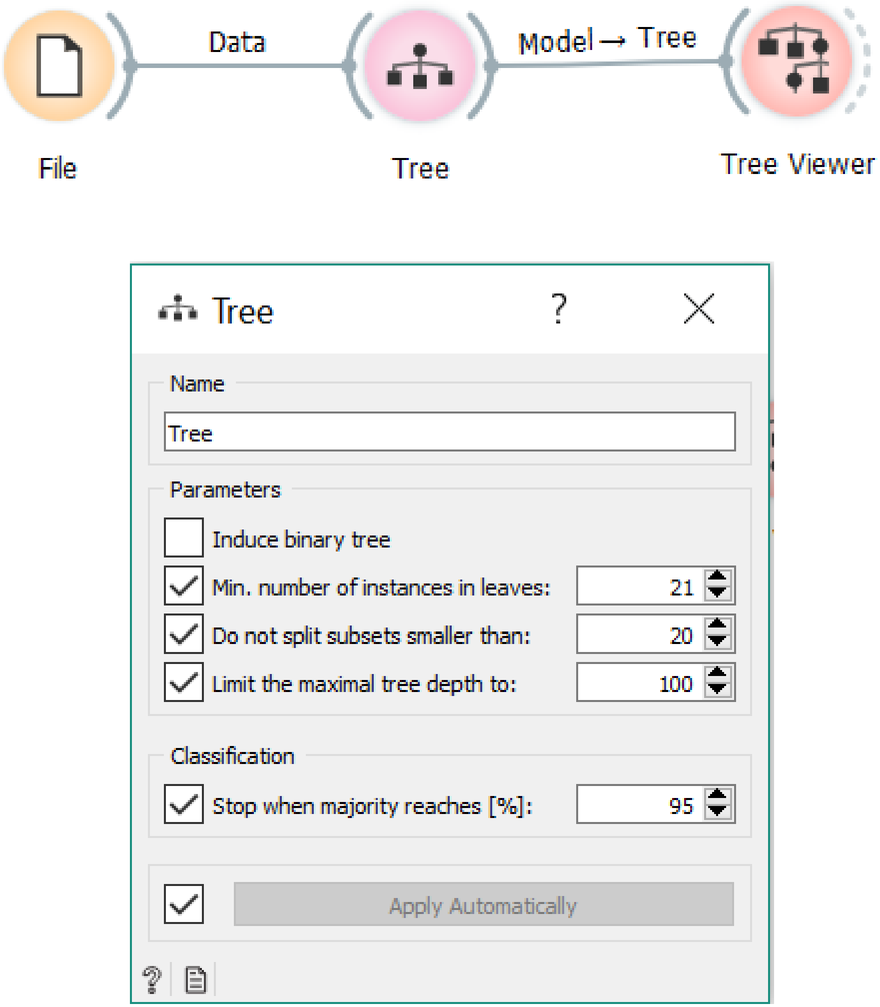
set S

number of examples in set S

# Lab exercise 1

Decision trees in Orange

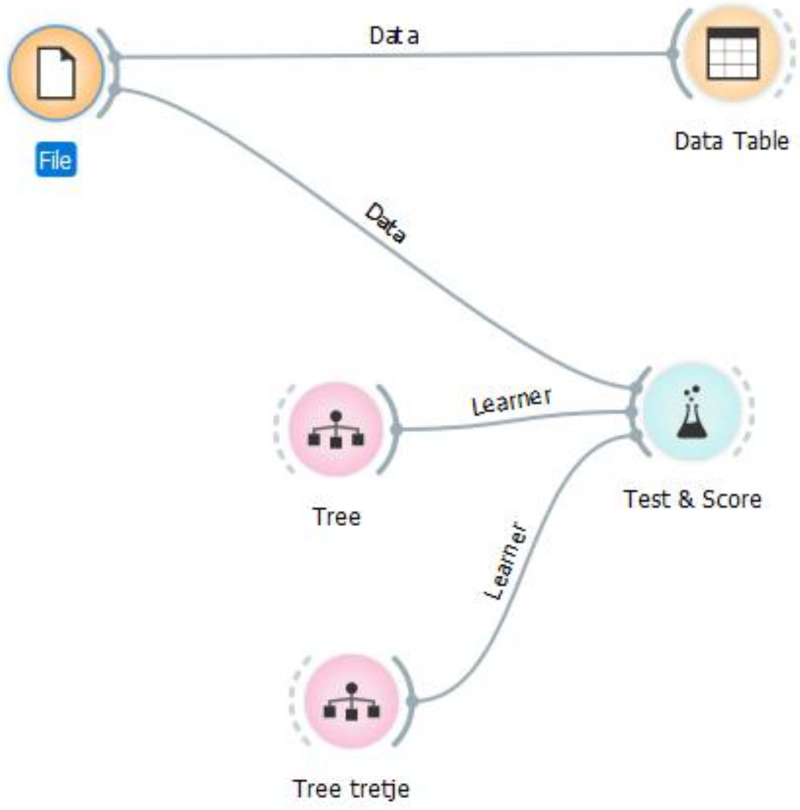
# Exercise 1: Induce a decision tree




- Dataset: “titanic”
- Play with tree parameters

- Repeat with the “adult” dataset

# Exercise 2: Evaluate the decision tree



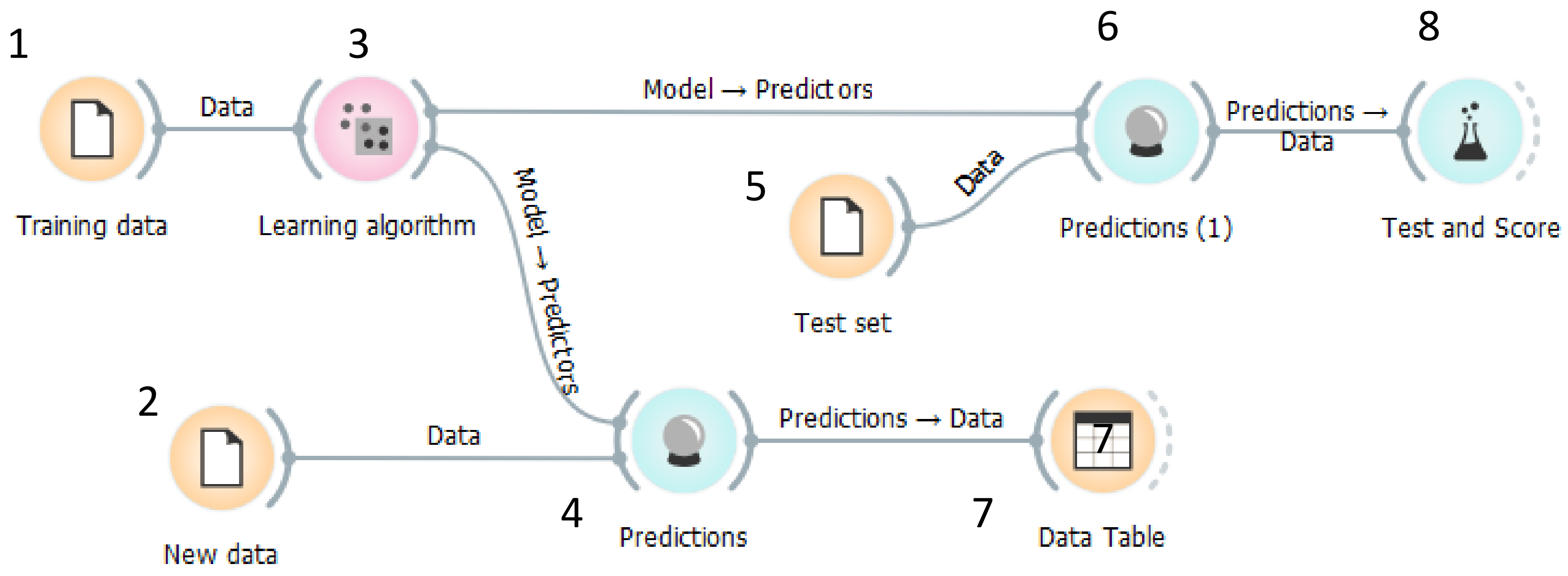

- Dataset: “zoo”
- Compare tree classifiers with different parameter values

# Discussion points

- How do we compute entropy for a target variable that has three values? Lenses = {hard=4, soft=5, none=13}
- What are the stopping criteria for building a decision tree? What other criteria could be used?
- How would you compute the information gain for a numeric attribute?

# Classification

1. Train the model on train data: 1, 3
2. Test the model on test data: 5, 6, 8
3. Classify new data with the model: 2, 4, 7

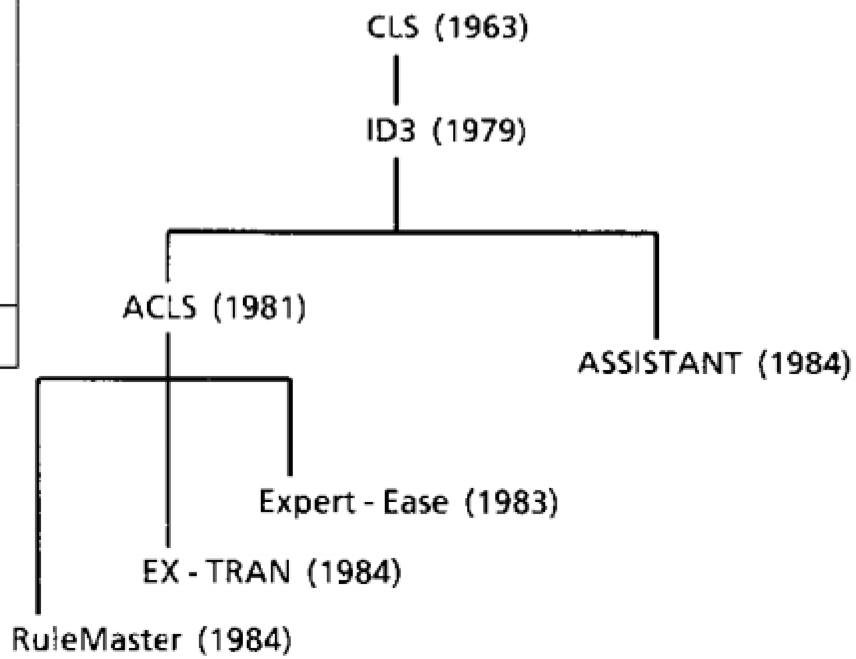


# The TDIDT family of learning systems

TDIDT: BASIC ALGORITHM

IF all the instances in the training set belong to the same class
THEN return the value of the class
ELSE (a) Select an attribute $A$ to split on$^{+}$
(b) Sort the instances in the training set into subsets, one for each value of attribute $A$
(c) Return a tree with one branch for each *non-empty* subset, each branch having a descendant subtree or a class value produced by applying the algorithm recursively

$^{+}$ Never select an attribute twice in the same branch



# Decision tree induction with ID3

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 $S_i$

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

# Exercise: Train and test a decision tree (ID3)

| 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** |

# Split the dataset into a training and a 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

# Information gain

number of examples in the subset $S_v$

set S

attribute A

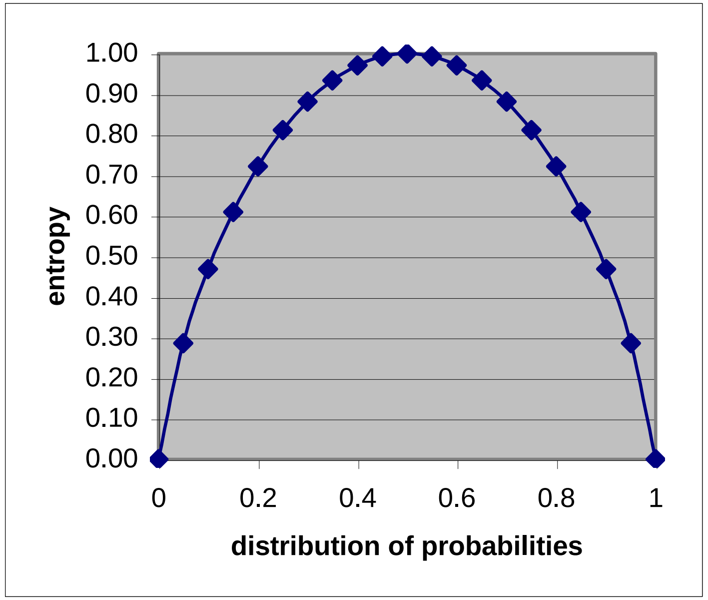
$$Gain(S,A) = E(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} \cdot E(S_v)$$

number of examples in set S

Weight = probability of a branch

# Entropy and information gain

| probability of class 1 | probability of class 2 | |
|---|---|---|
| $p_1$ | $p_2 = 1\text{-}p_1$ | **entropy $E(p_1, p_2)$ =** $-p_1 * \log_2(p_1) - p_2 * \log_2(p_2)$ |
| 0 | 1 | 0.00 |
| 0.05 | 0.95 | 0.29 |
| 0.10 | 0.90 | 0.47 |
| 0.15 | 0.85 | 0.61 |
| 0.20 | 0.80 | 0.72 |
| 0.25 | 0.75 | 0.81 |
| 0.30 | 0.70 | 0.88 |
| 0.35 | 0.65 | 0.93 |
| 0.40 | 0.60 | 0.97 |
| 0.45 | 0.55 | 0.99 |
| 0.50 | 0.50 | 1.00 |
| 0.55 | 0.45 | 0.99 |
| 0.60 | 0.40 | 0.97 |
| 0.65 | 0.35 | 0.93 |
| 0.70 | 0.30 | 0.88 |
| 0.75 | 0.25 | 0.81 |
| 0.80 | 0.20 | 0.72 |
| 0.85 | 0.15 | 0.61 |
| 0.90 | 0.10 | 0.47 |
| 0.95 | 0.05 | 0.29 |
| 1 | 0 | 0.00 |




number of examples in the subset

probability of the "branch"

attribut A

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

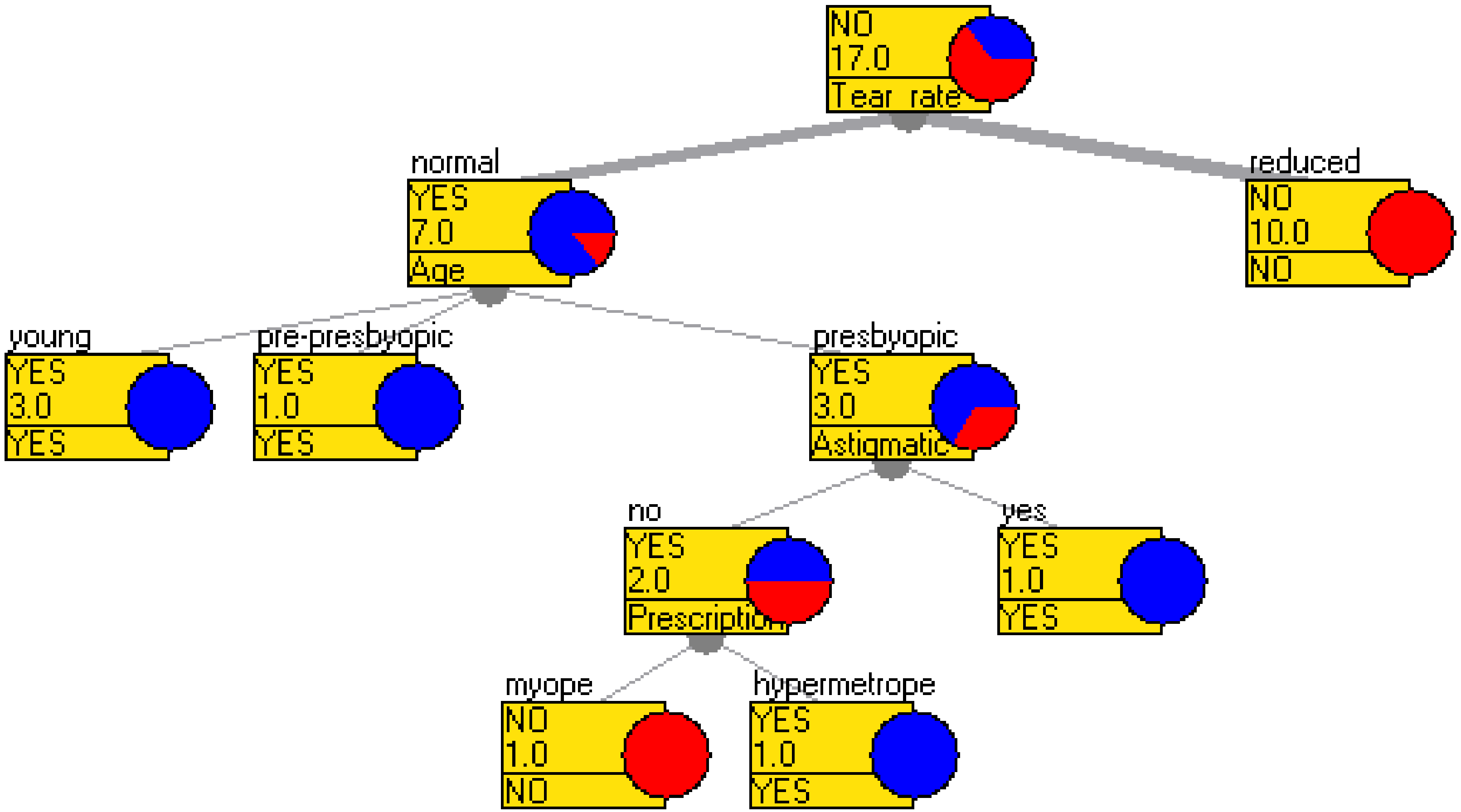
set S

number of examples in set S

# Exercise: Induce a decision tree on this dataset

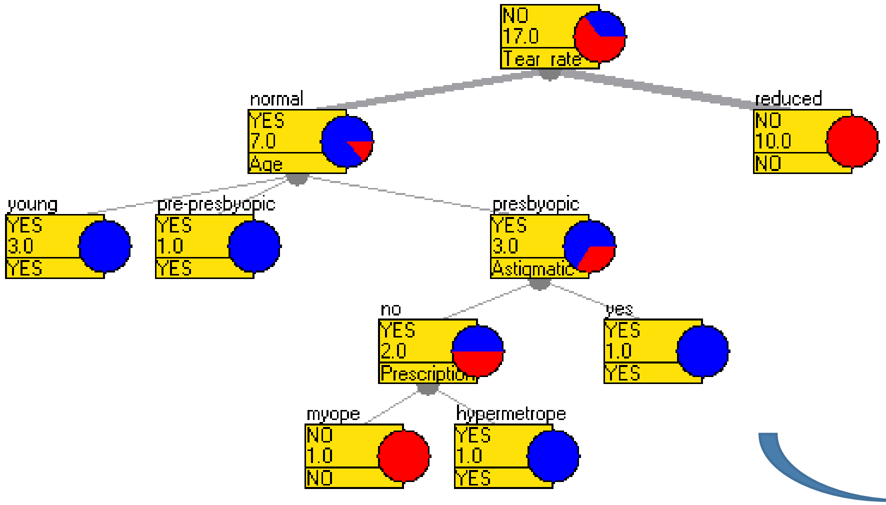
| Person | Age | Prescription | Astigmatic | Tear_Rate | **Lenses** |
|---|---|---|---|---|---|
| P1 | young | myope | no | normal | **YES** |
| P2 | young | myope | no | reduced | **NO** |
| P4 | young | hypermetrope | no | reduced | **NO** |
| P5 | young | myope | yes | normal | **YES** |
| P6 | young | myope | yes | reduced | **NO** |
| P7 | young | hypermetrope | yes | normal | **YES** |
| P8 | young | hypermetrope | yes | reduced | **NO** |
| P10 | pre-presbyopic | myope | no | reduced | **NO** |
| P11 | pre-presbyopic | hypermetrope | no | normal | **YES** |
| P14 | pre-presbyopic | myope | 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** |
| P24 | presbyopic | hypermetrope | yes | reduced | **NO** |

# The induced decision tree



# Classification with the tree

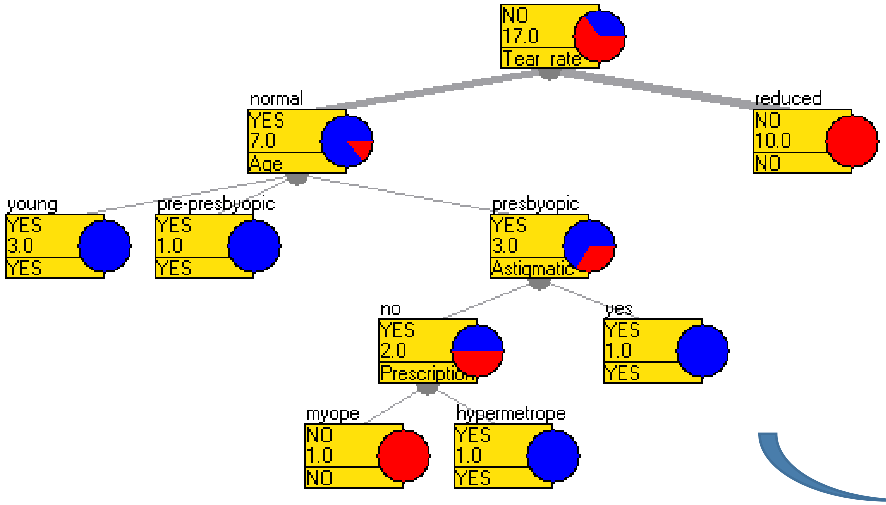
| Person | Age | Prescription | Astigmatic | Tear_Rate | **Lenses** |
|---|---|---|---|---|---|
| P3 | young | hypermetrope | no | normal | **YES** |
| P9 | pre-presbyopic | myope | no | normal | **YES** |
| P12 | pre-presbyopic | hypermetrope | no | reduced | **NO** |
| P13 | pre-presbyopic | myope | yes | normal | **YES** |
| P15 | pre-presbyopic | hypermetrope | yes | normal | **NO** |
| P16 | pre-presbyopic | hypermetrope | yes | reduced | **NO** |
| P23 | presbyopic | hypermetrope | yes | normal | **NO** |




| | Predicted „YES“ | Predicted „NO“ |
|---|---|---|
| Actual „YES“ | | |
| ACTUAL „NO“ | | |

# Classification with the tree

| Person | Age | Prescription | Astigmatic | Tear_Rate | **Lenses** |
|---|---|---|---|---|---|
| P3 | young | hypermetrope | no | normal | **YES** |
| P9 | pre-presbyopic | myope | no | normal | **YES** |
| P12 | pre-presbyopic | hypermetrope | no | reduced | **NO** |
| P13 | pre-presbyopic | myope | yes | normal | **YES** |
| P15 | pre-presbyopic | hypermetrope | yes | normal | **NO** |
| P16 | pre-presbyopic | hypermetrope | yes | reduced | **NO** |
| P23 | presbyopic | hypermetrope | yes | normal | **NO** |




Classification accuracy= (3+2)/ (3+2+2+0) = 71%

| | Predicted „YES“ | Predicted „NO“ |
|---|---|---|
| Actual „YES“ | TP=3 | FN=0 |
| ACTUAL „NO“ | FP=2 | TN=2 |

# Questions

- Construct an attribute with Information gain =1.
- Construct an attribute with Information gain =0.
- Compute the Information gain of the attribute “Person”.
- How would you compute the information gain of a numeric attribute.

# Lab exercise 2

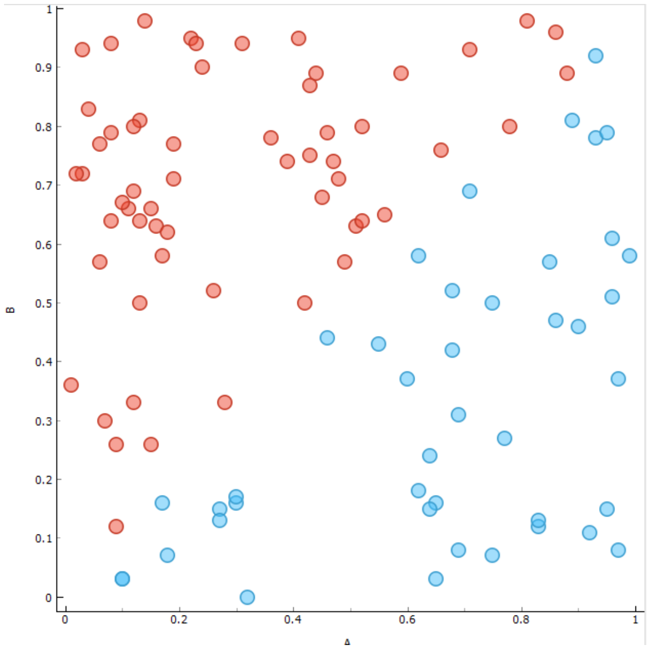
Language bias of decision trees

# Lab exercise: Decision trees & Language bias

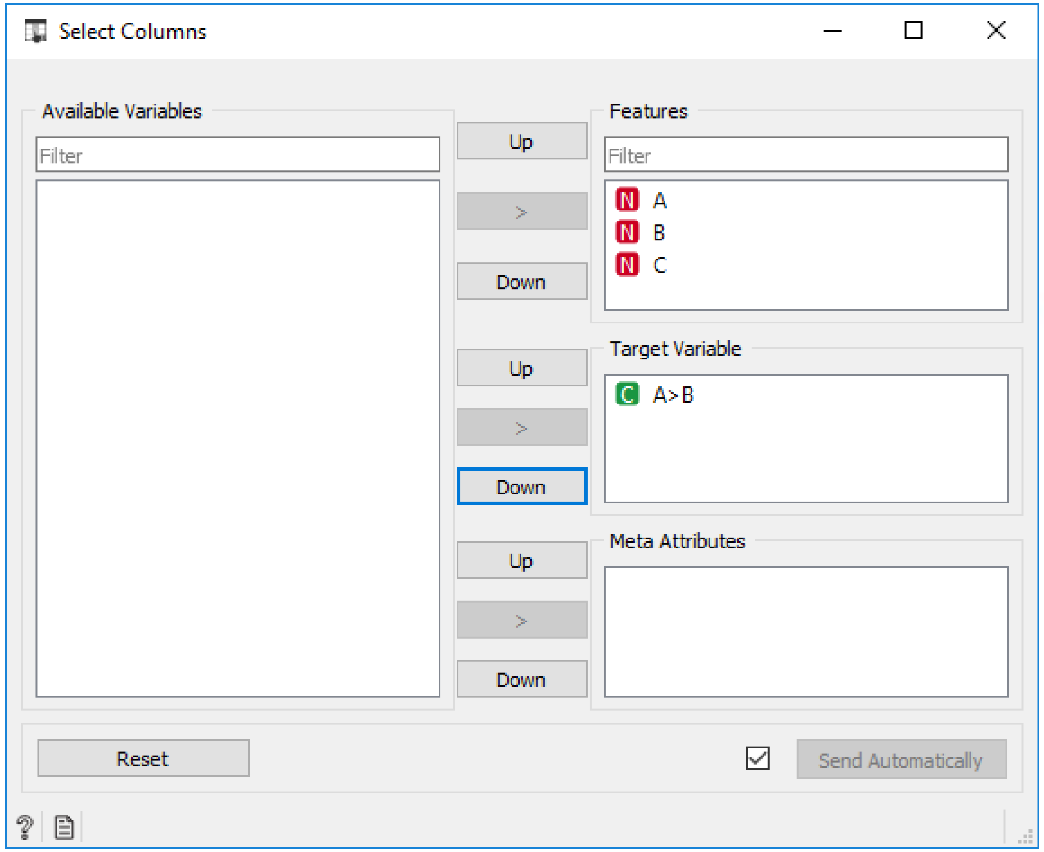
- Use a spreadsheet program (e.g. MS Excel) to generate 1000 examples:
    - Attributes A, B and C should have random values
    - Target variable „A>B“, should have value „true“ if A>B else “false”
    - Save the file
- Use Orange trees to predict „A>B“ from the attributes A, B in C
    - Set the target variable
    - Use separate test set for validation
    - Plot the training and classified data in “Scatter Plot”
- How good is your model?
- How does the training set size influence the model performance?
- MS Excel hints:

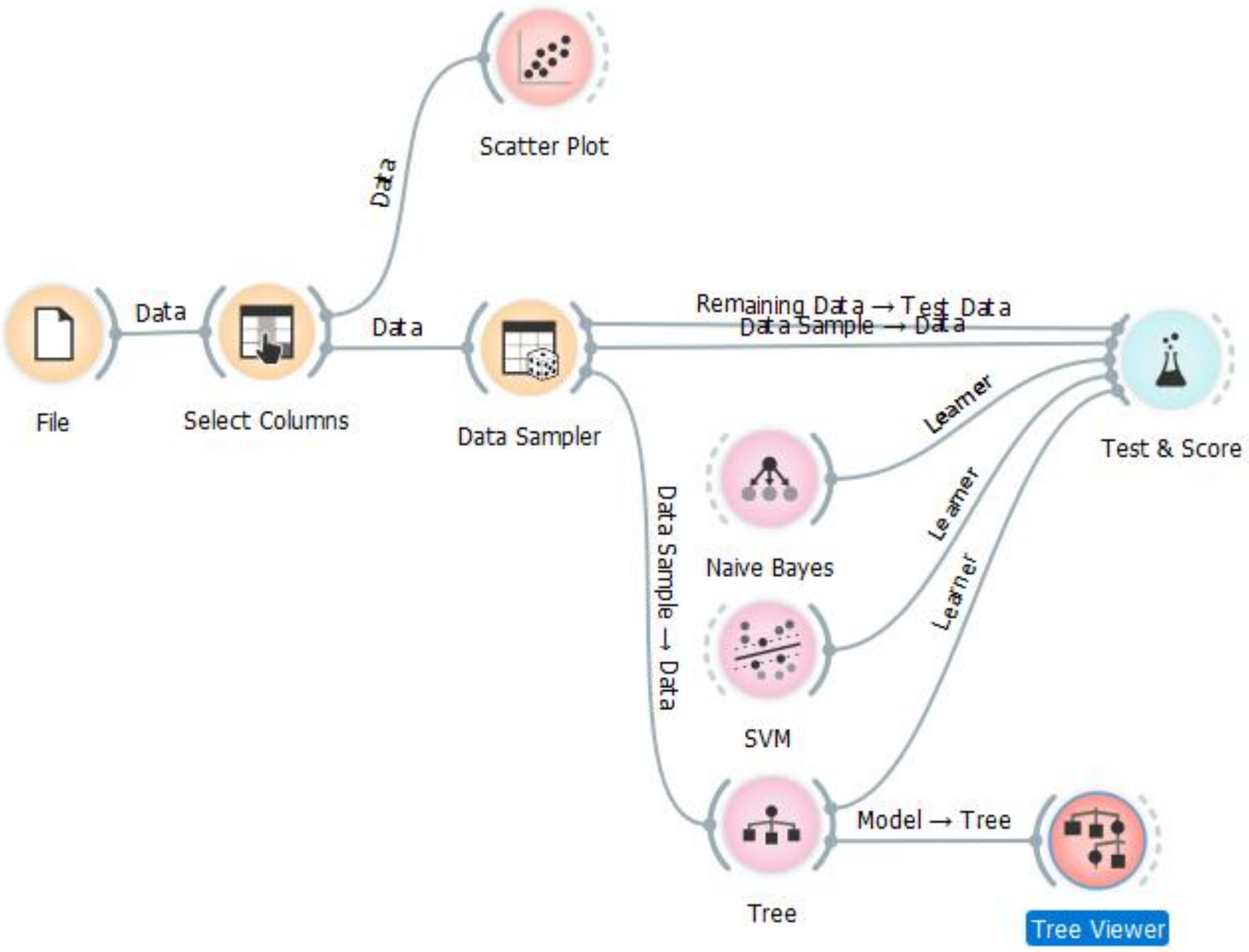
  = RAND()

  = IF(A2>B2,“true",“false")

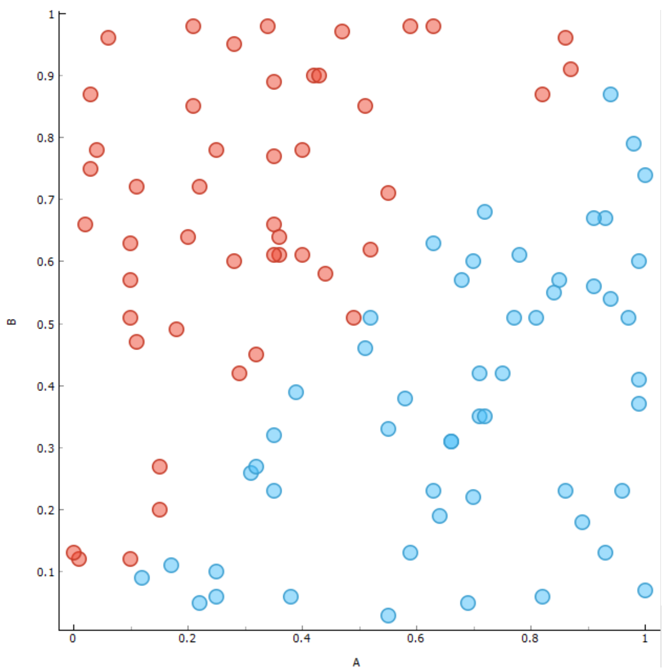


# Lab exercise: Decision trees & Language bias

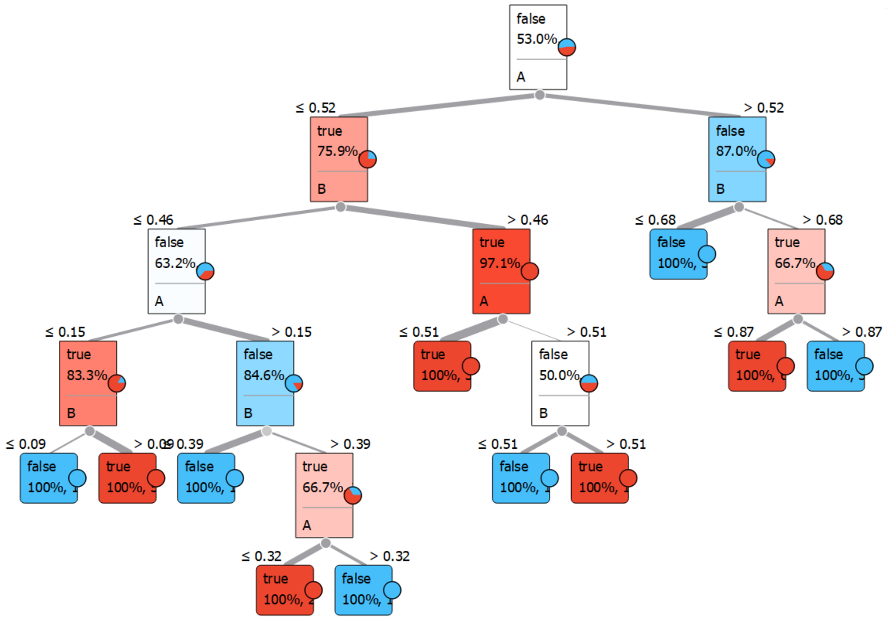


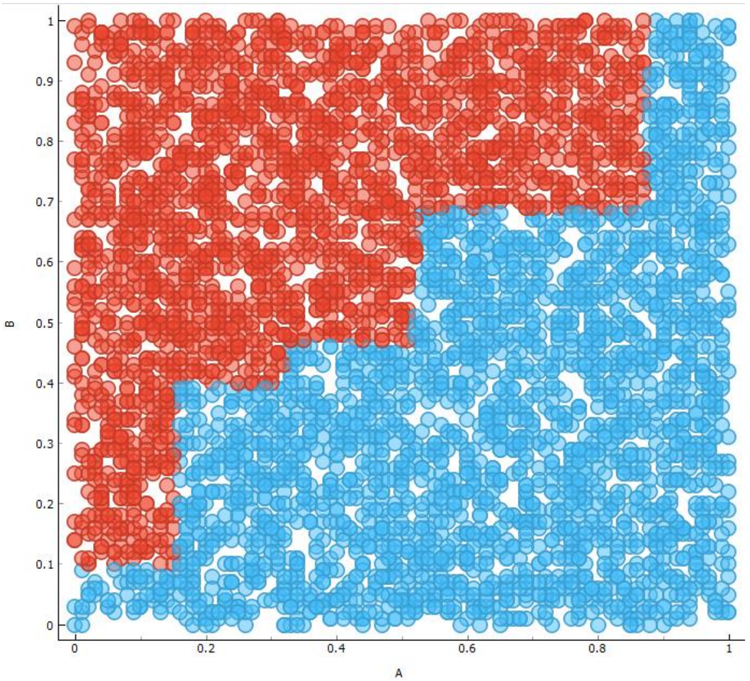
# Lab exercise: Decision trees & Language bias



Training set

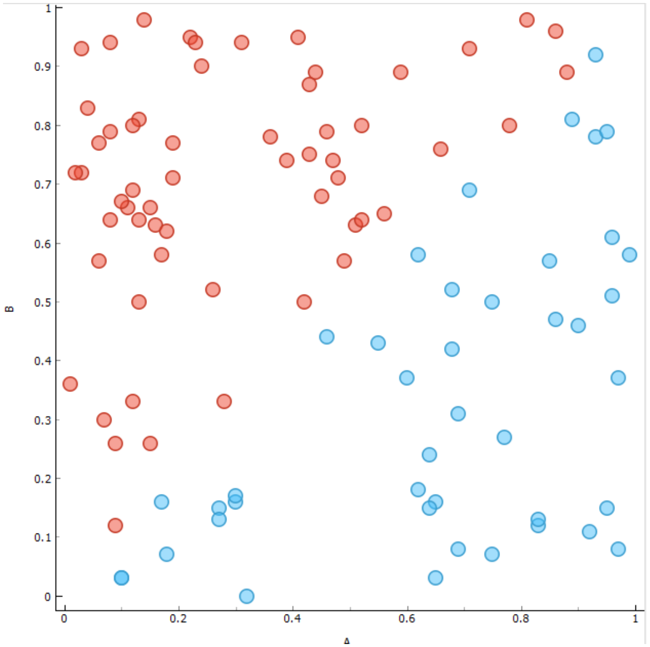



Decision tree

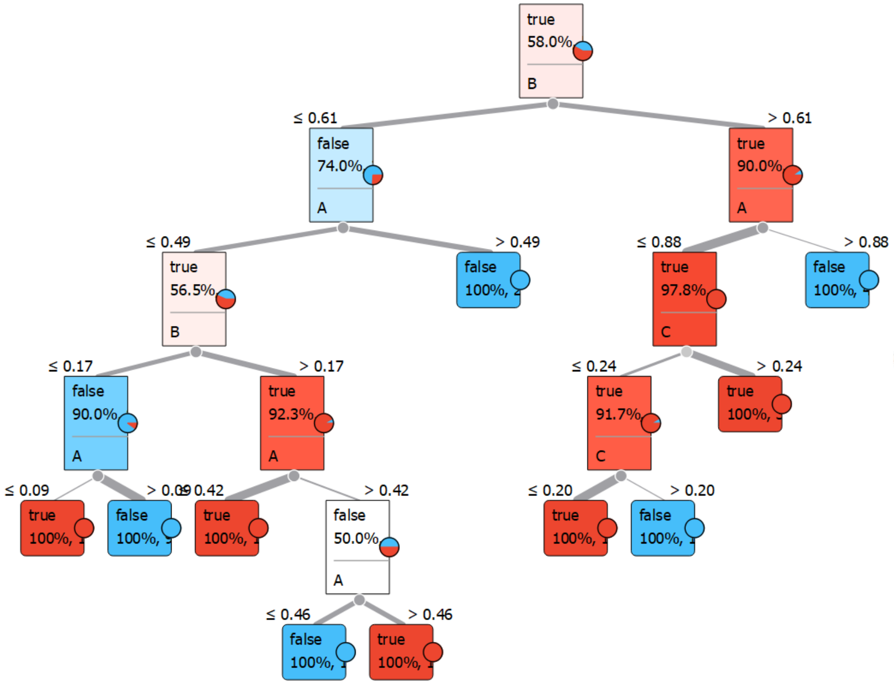


Test set

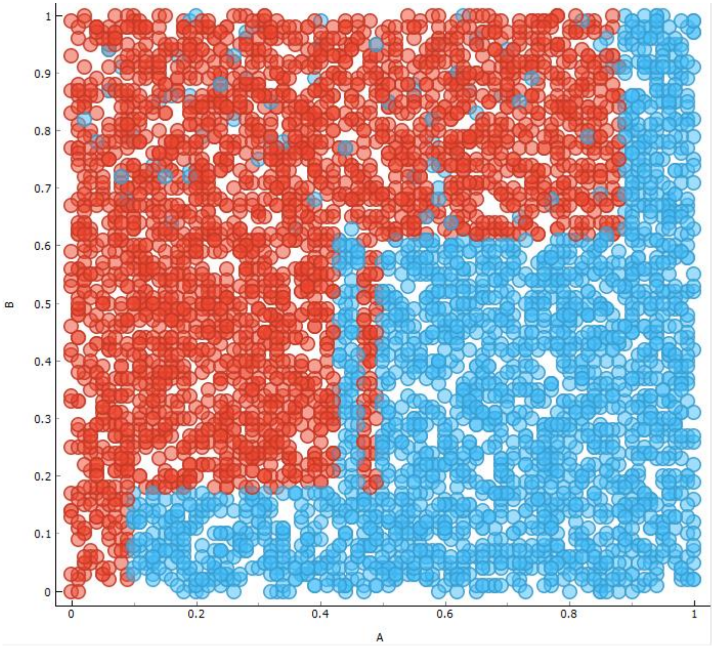
# Same program, different random seed



Training set




Decision tree



Test set

# How to overcome this

- Feature engineering
  - Create a new feature A>B
  - Examples
    - We have a person’s height and body mass
      → Create a new attribute BMI (bod mass index)
    - We have income and outcome data
      → Create a new attribute “profit”

$$BMI = \frac{Weight\ (kg)}{[Height(m)]^2}$$

- Ensemble
  - We build more models that vote for the final classification
  - Random forest: Several trees built on different subsets od the training set
  - On this example, decision trees achieve CA 88,2% while random forest 90,8%
  - As a general rule, classifier ensembles always outperform single classifiers

# Evaluation

How good is the model

!!!

# Evaluation goal

- How good is the model

- Method
  - HOW we measure

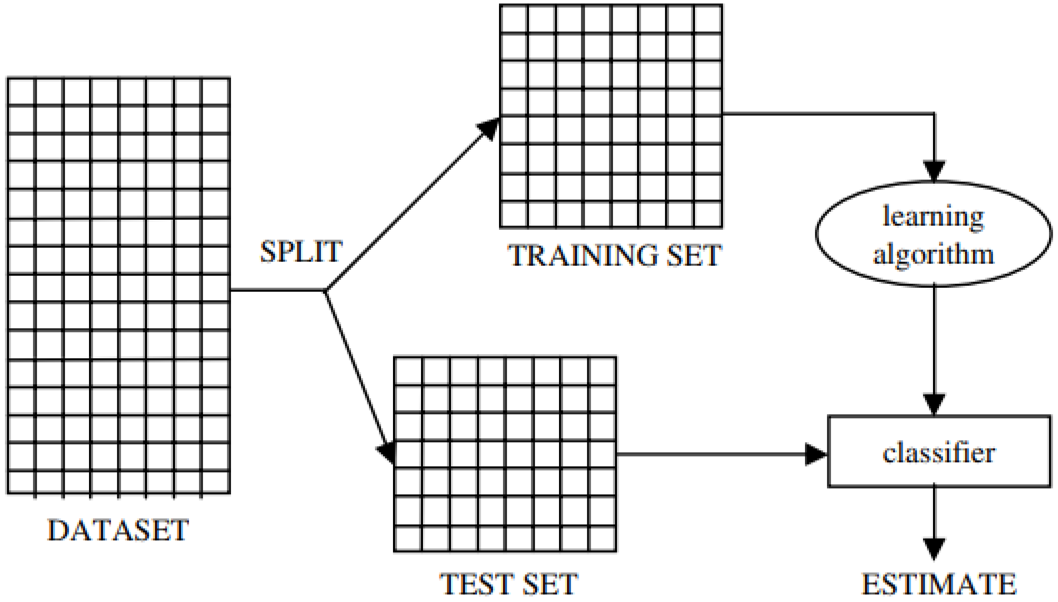
- Measure
  - WHAT me measure

# 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

# Method: Test on a separate test set



# Stratified sampling

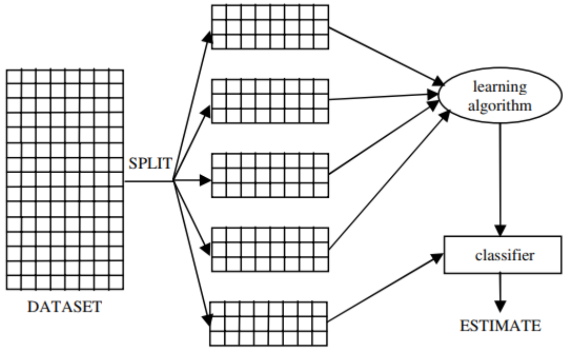
- Stratified sampling aims at splitting one data set so that each split are similar with respect to the target variable distribution.

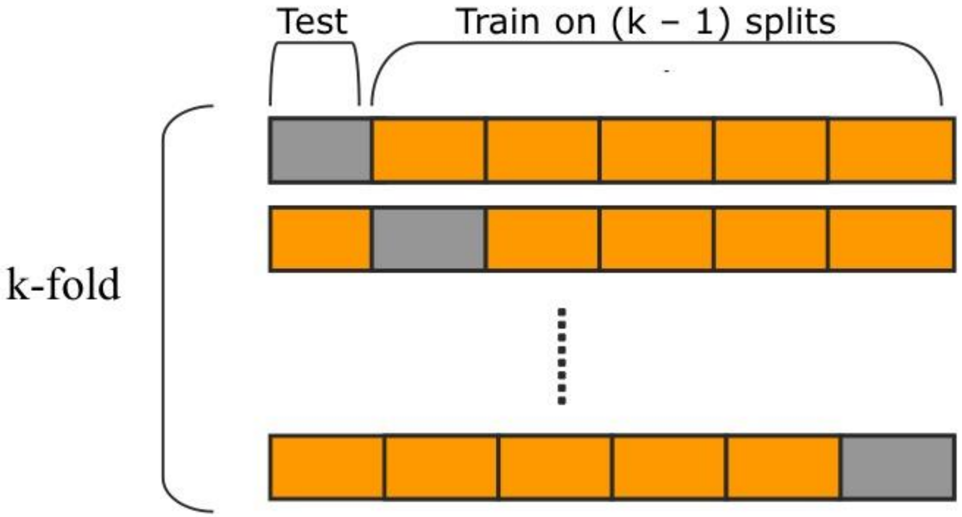
# Method: Random sampling

- Repeat several times „Test on a separate test set“ with different test set selections
- Compute the mean, variance on the results …
- The evaluation is more robust as it does not depend so much on a single random split

# Method: *K*-fold cross validation

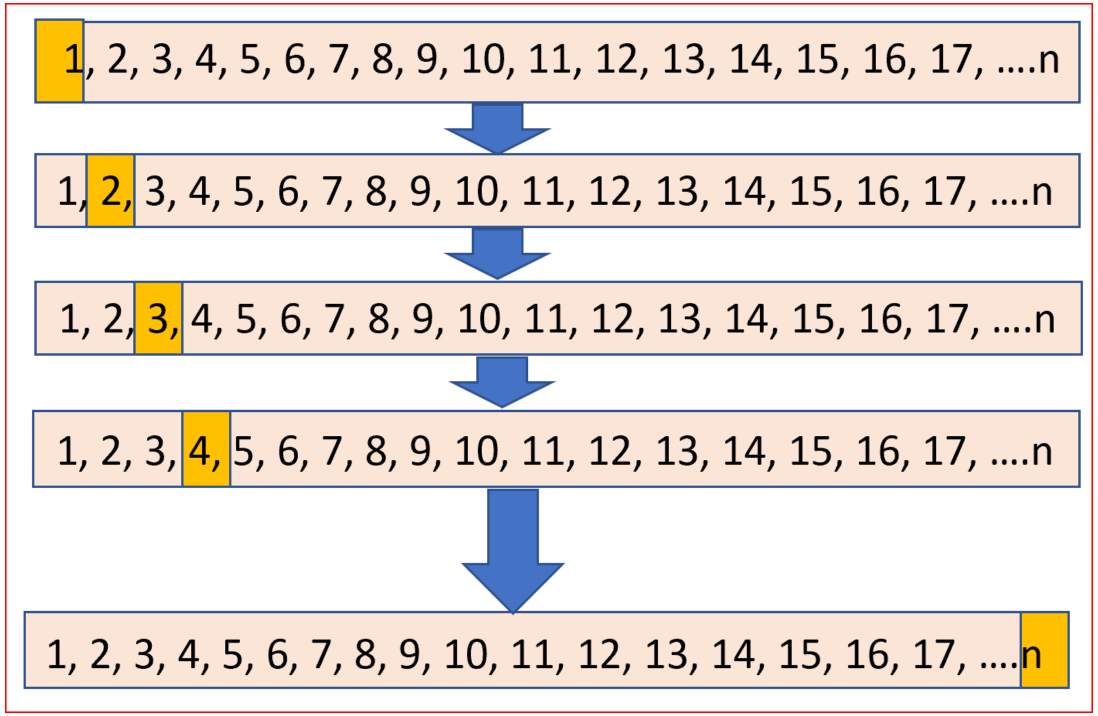
- Most commonly used in machine learning
- Split the dataset into *k* (disjunctive) subsets
- Repeat *k*-times:
  - Use a different subset for testing
  - Use all the other data for training
- Each example is in the test set just once



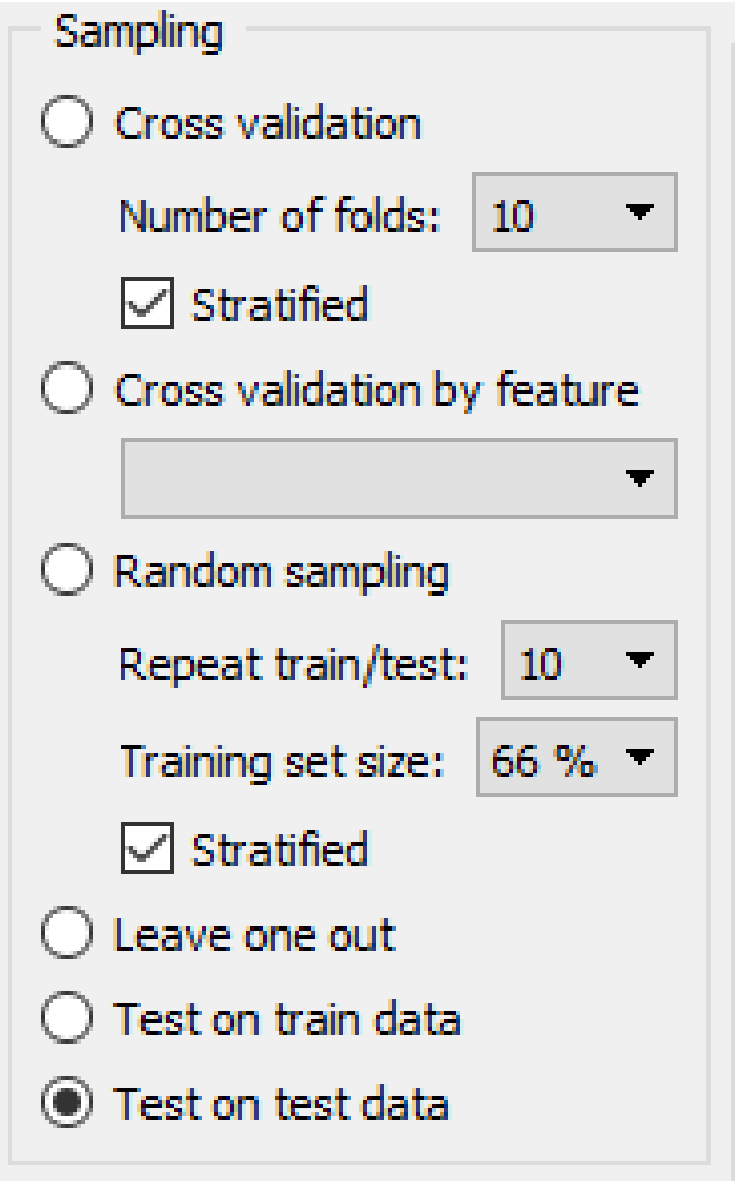
# Method: Leave one out (N-fold cross-validation)

- For small datasets
- Similar to cross validation with test set size =1
- Repeat the training *N*-times if there is *N* examples in the dataset



# Evaluation methods in Orange


- Cross validation
- Random sampling
- Leave one out
- Test on train data
- Test on test data



# Questions

- What are properties of the results of testing on the training set?

# Classification quality measures

# Confusion matrix (error matrix)

Breakdown of the classifier's performance, i.e. how frequently instances of class X were correctly classified as class X or misclassified as some other class.

Primer: car

| 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 |

Primer: titanic

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

# Confusion matrix

- Matrix of correct and incorrect classifications
  - Rows are actual values
  - Columns are predicted values
  - Correct classifications are on the diagonal

| 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 |

# Confusion matrix for two classes

Predicted

Actual

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

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

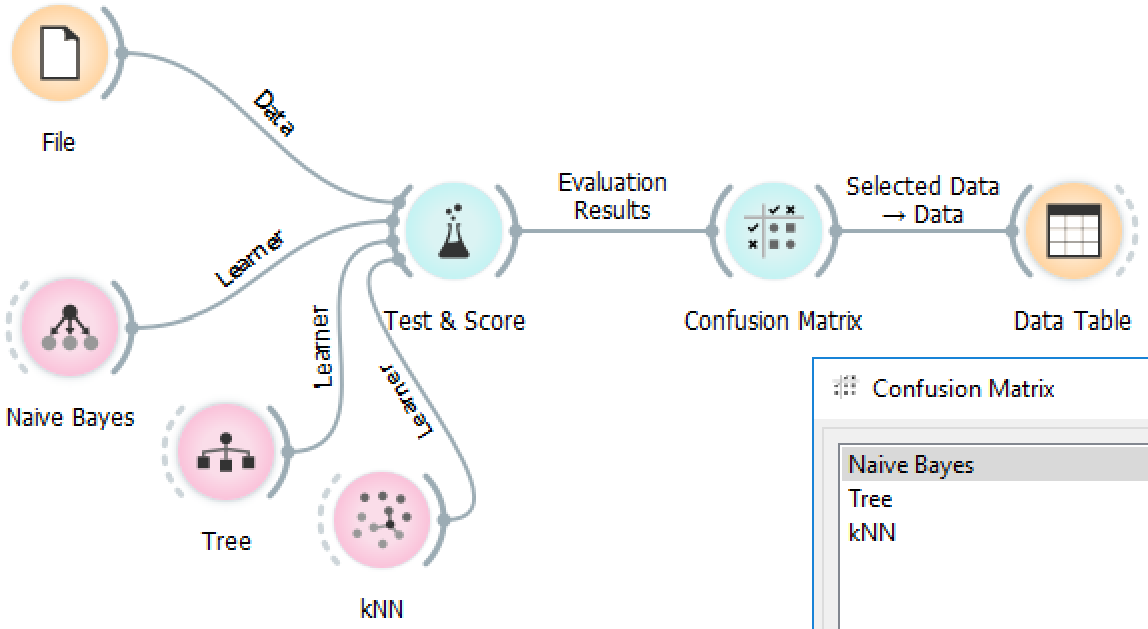
TN: true negatives
The number of negative instances that are classified as negative

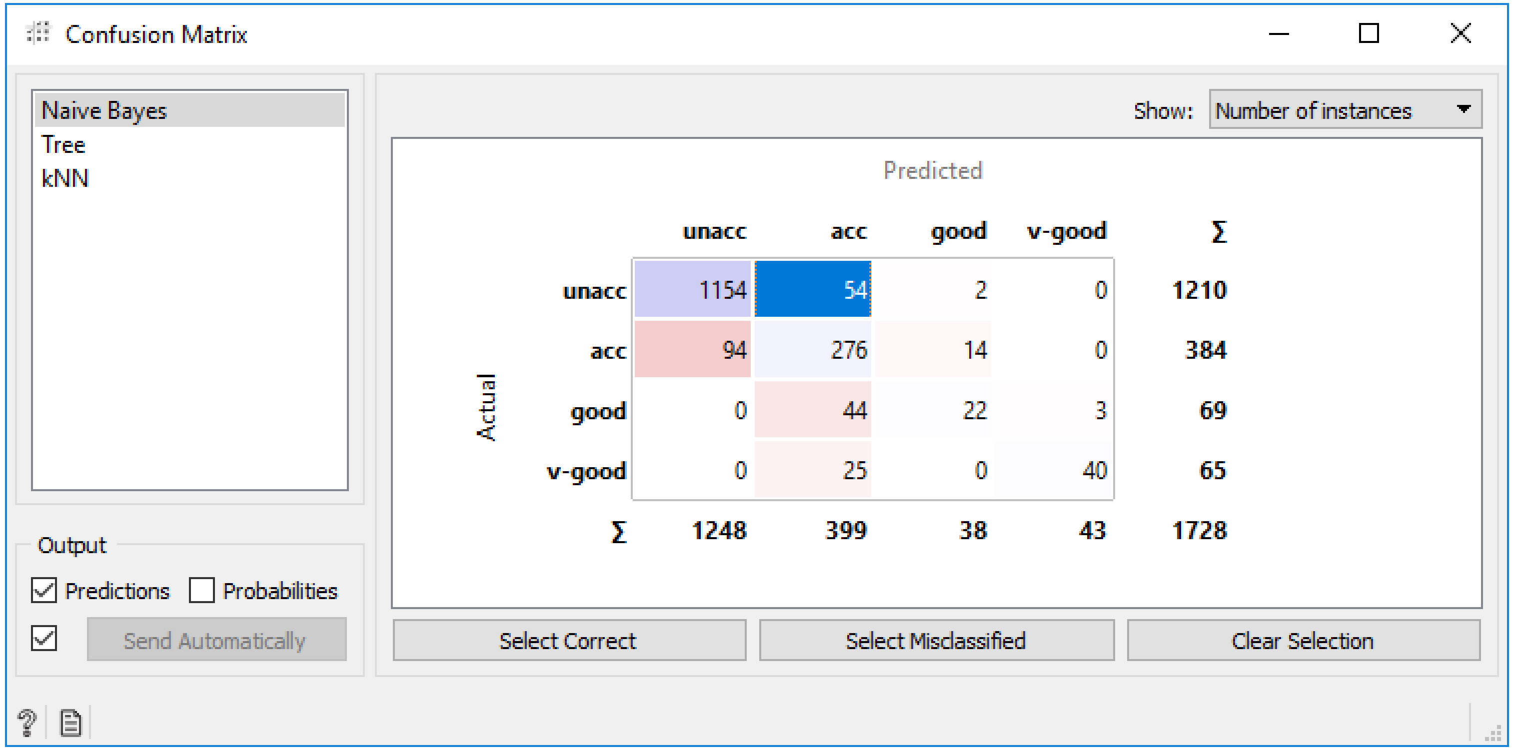
- Diagonal: correct classifications
- Outside: misclassifications
- Classification accuracy =

= |correct classifications| / |all examples|

= |correct classifications| / (|correct classifications| + |misclassifications|)

# In Orange, the confusion matrix is interactive



# Classification accuracy

- Percentage of correctly classified examples

Classification accuracy =
= |correct classifications| / |all examples|
= |correct classifications| / (|correct classifications| + |misclassifications|)

# Exercise: Confusion matrix

**Titanic**

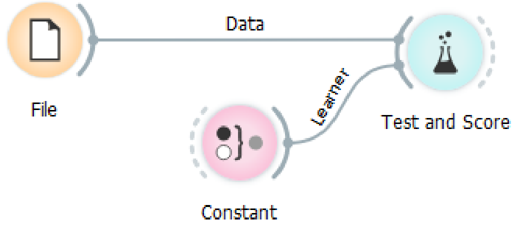
| Actual \ Predicted | no | yes | Σ |
|---|---|---|---|
| no | 1364 | 126 | 1490 |
| yes | 362 | 349 | 711 |
| Σ | 1726 | 475 | 2201 |

**Car**

| 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 |

| | Titanic | Car |
|---|---|---|
| Number of examples | | |
| Number of classes | | |
| Number of examples in each class | | |
| Number of examples classified in individual classes | | |
| Number of misclassified examples | | |
| Classification accuracy | | |

# Majority class classifier (Constant)




Predicted

| Actual | 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 |

Predicted

| Actual | no | yes | Σ |
|---|---|---|---|
| no | 1364 | 126 | 1490 |
| yes | 362 | 349 | 711 |
| Σ | 1726 | 475 | 2201 |

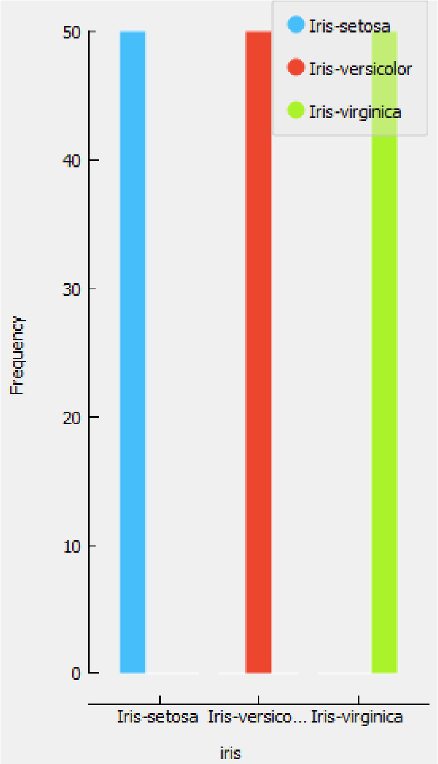
- What is the classification accuracy of a classifier that classifies all the examples in the majority class?
- Car: 70% Titanic: 68%

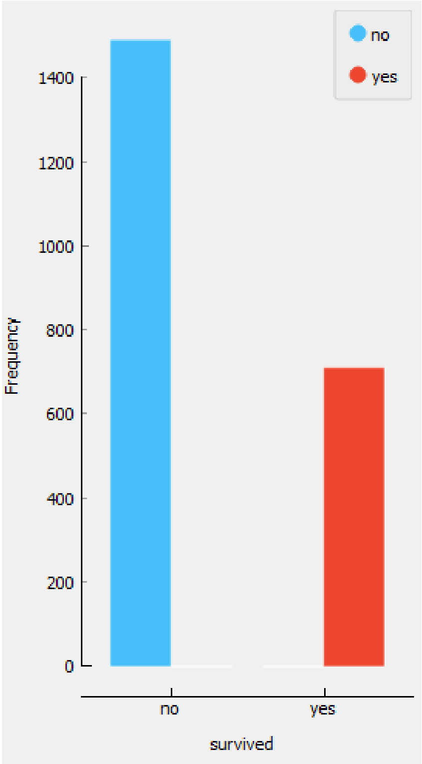
# Question

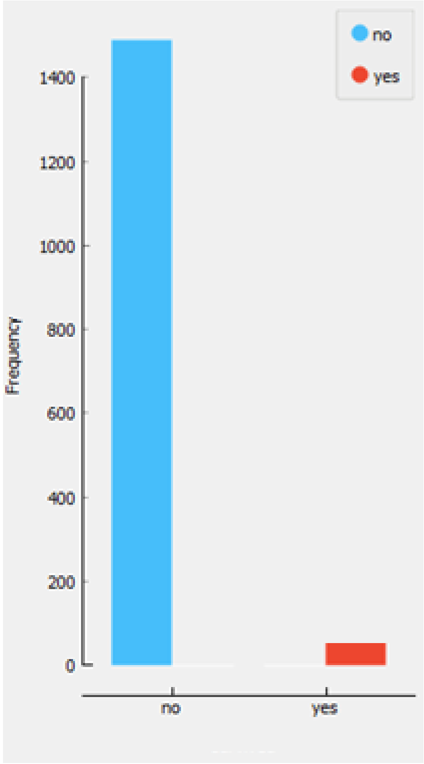
- When is classification accuracy “good”?

# Imbalanced Data and Unequal Misclassification Costs

- Imbalanced dataset: One class is minority compared to the other(s)
  - The minority class is usually the one of interest



# Imbalanced Data and Unequal Misclassification Costs

- Imbalanced dataset: One class is minority compared to the other(s)
  - The minority class is usually the one of interest
- Unequal misclassification costs:
  - Some errors are more costly (have more severe consequences)
- Examples:
  - Screening tests (nuchal scan, Zora, Dora, Svit, …)
  - Intrusion detection
  - Credit card fraud

# Exercise: Credit card fraud

*„FED report notes the fraud rate for debit and prepaid signature transactions in 2012 was approximately 4.04 basis points (bps), or about **four per every 10,000 transactions**.“*

- What is the classification accuracy of a classifier that classifies all the examples a „not fraudulent“?
  - Answer: 99.96%
- Can a classifier with a 98% accuracy be “better” then the one with classification accuracy 99.96%?

https://www.pymnts.com/in-depth/2014/a-tale-of-two-fraud-stats/

# Exercise: Credit card fraud

**Two confusion matrices for two classifiers**

| | | Predicted | | |
|---|---|---|---|---|
| | | Fraud | Not Fraud | |
| Actual | Fraud | 0 | 4 | 4 |
| | Not fraud | 0 | 9996 | 9996 |
| | | 0 | 10000 | 10000 |

| | | Predicted | | |
|---|---|---|---|---|
| | | Fraud | Not Fraud | |
| Actual | Fraud | 4 | 0 | 4 |
| | Not fraud | 300 | 9696 | 9996 |
| | | 304 | 9696 | 10000 |

**Classification accuracy**

- CA = (0 + 9996)/10000

  = 99,96%

- CA = (4 + 9696)/10000

  = 97,00%

The model with lower classification accuracy is better.

# Precision & Recall

- Class-specific metrics
  - Precision (Positive Predictive Value)
    - Proportion of instances classified as positive that are really positive
  - Recall (True Positive Rate, TP Rate, Hit Rate, Sensitivity)
    - The proportion of positive instances that are correctly classified as positive
- Exercise: write down the formulas for precision and recall

| | | Predicted class | | Total instances |
|---|---|---|---|---|
| | | + | − | |
| Actual class | + | TP | FN | P |
| | − | FP | TN | N |

# Precision, Recall & F1

- Class-specific metrics
  - Precision (Positive Predictive Value)
    - Proportion of instances classified as positive that are really positive
  - Recall (True Positive Rate, TP Rate, Hit Rate, Sensitivity)
    - The proportion of positive instances that are correctly classified as positive
  - F1
    - Harmonic mean of precision and recall

$$F_1 = 2 * \frac{precision * recall}{precision + recall}$$

- We can average the metrics over the classes (macro average) or weigh them by the number of examples (micro average)

# Precision, recall, F1

| | | Predicted class | | Total instances |
|---|---|---|---|---|
| | | + | − | |
| Actual class | + | TP | FN | P |
| | − | FP | TN | N |

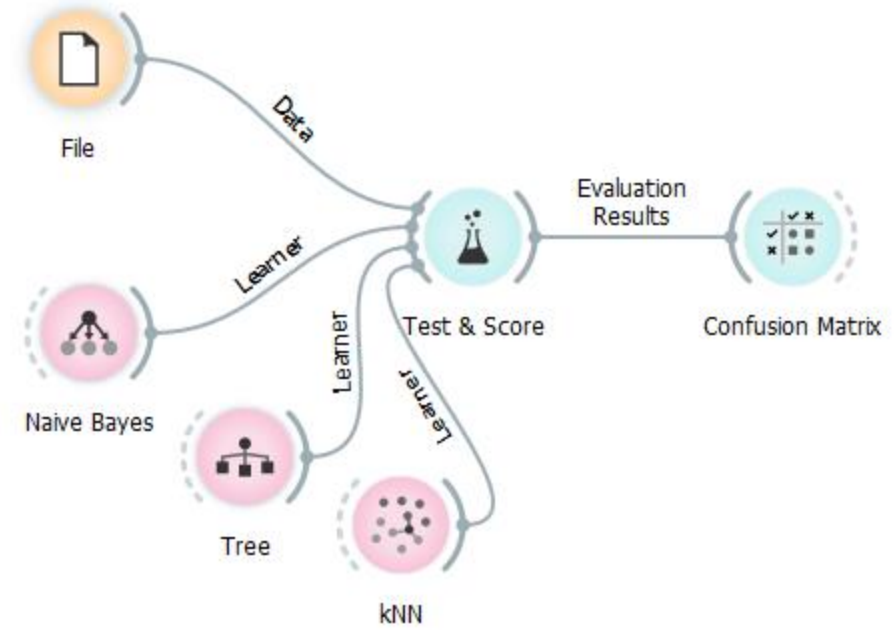
| | | | |
|---|---|---|---|
| **True Positive Rate** or Hit Rate or Recall or Sensitivity or TP Rate | TP/P | The proportion of positive instances that are correctly classified as positive | Priklic |
| **Precision** or Positive Predictive Value | TP/(TP+FP) | Proportion of instances classified as positive that are really positive | Natančnost |
| **F1 Score** | (2 × Precision × Recall) /(Precision + Recall) | A measure that combines Precision and Recall | Mera F1 |
| **Accuracy** or Predictive Accuracy | (TP + TN)/(P + N) | The proportion of instances that are correctly classified | Klasifikacijska točnost |

# Classification evaluation in Orange

- AUC
  - Area under curve
  - AUROC
  - Površina pod ROC krivuljo
- CA – classification accuracy
  - Klasifikacijska točnost
- F1 – harmonično povprečje priklica in natančnosti
- Precision – natančnost
- Recall - priklic

Evaluation Results

| Method | AUC | CA | F1 | Precision | Recall |
|---|---|---|---|---|---|
| kNN | 0.951 | 0.845 | 0.823 | 0.835 | 0.845 |
| Naive Bayes | 0.971 | 0.863 | 0.858 | 0.859 | 0.863 |
| Tree | 0.991 | 0.951 | 0.951 | 0.951 | 0.951 |



# Lab exercise 3

Classifier evaluation

# Lab exercise

- Compare three evaluation methods
  - Train (70%) test (30%) split
  - Cross validation
  - Random sampling
- Test three models:
  - Decision trees
  - Random forest
  - Naïve Bayes classifier
- Metrics
  - Classification accuracy (CA)
  - Micro and macro Average F1
  - Area under curve (AUC) *– more about this next time*
- Use the dataset „car“

# Literature

- Max Bramer: Principles of data mining (2007)
  - 1. Data for Data Mining
  - 2. Introduction to Classification: Naive Bayes and Nearest Neighbour
  - 3. Using Decision Trees for Classification
  - 4. Decision Tree Induction: Using Entropy for Attribute Selection
  - We skip 5
  - 6. Estimating the Predictive Accuracy of a Classifier
  - 8. Avoiding Overfitting of Decision Trees
  - 11. Measuring the Performance of a Classifier