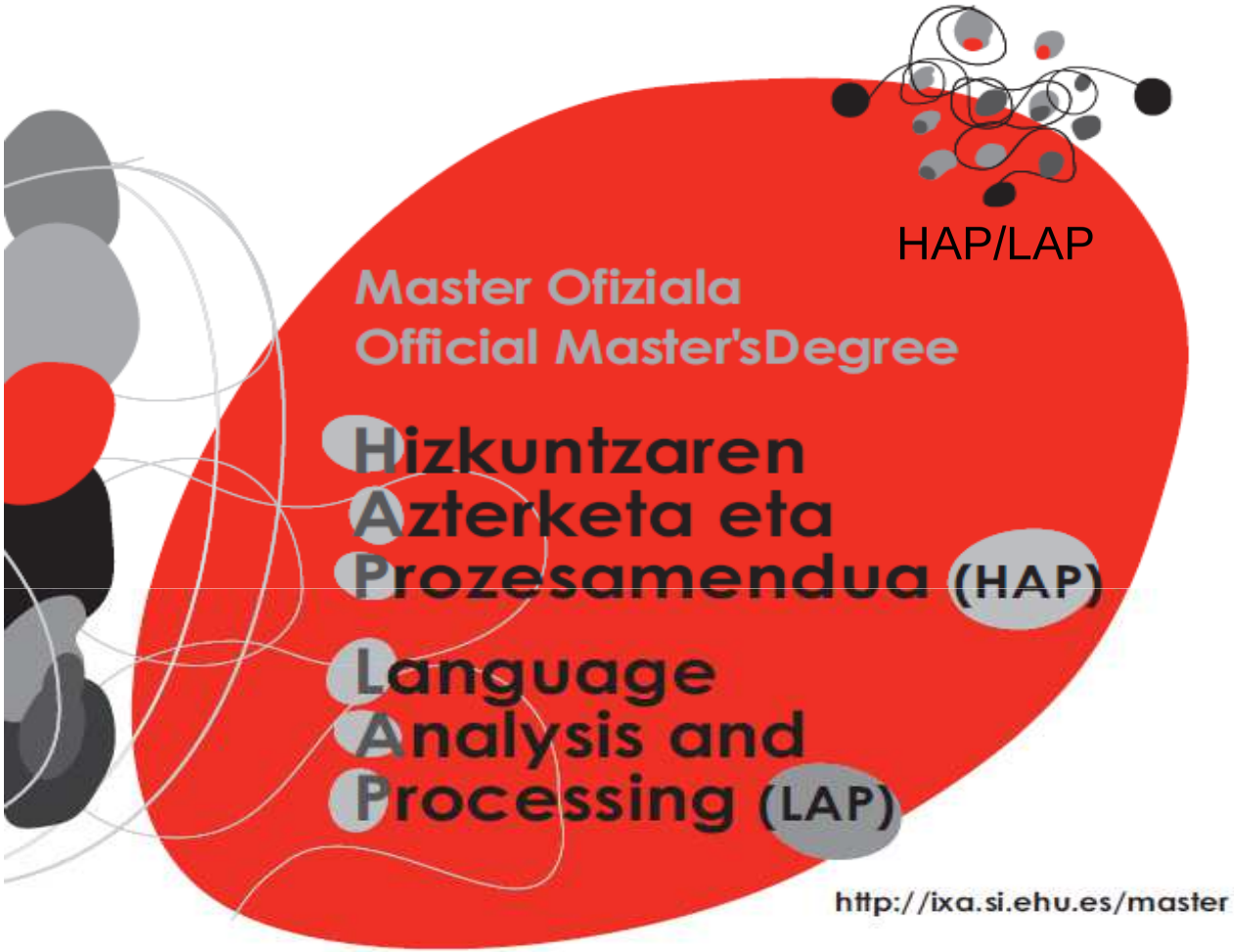
Erasmus Mundus

Language and Communication Technologies



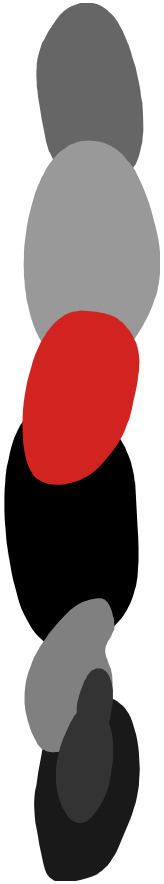
ixa

ueu udako euskal unibertsitatea

Universidad del País Vasco · Euskal Herriko Unibertsitatea · MDe Master eta Doktorego Eskola · Escuela de Máster y Doctorado · Master and Doctoral School

lct

Erasmus Mundus



HAP/LAP

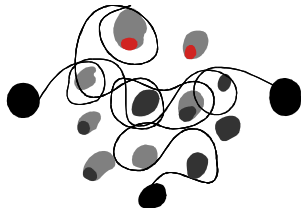
Statistical methods and text corpora

Introduction to Machine Learning
for Natural Language Processing

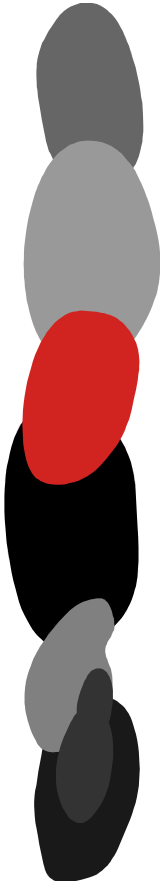
Olatz Arbelaitz: olatz.arbelaitz@ehu.eus

lct

Erasmus Mundus



Topics

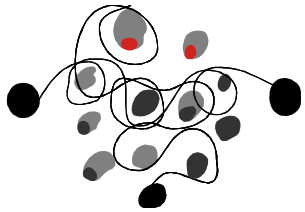


1.- Machine Learning forLNP

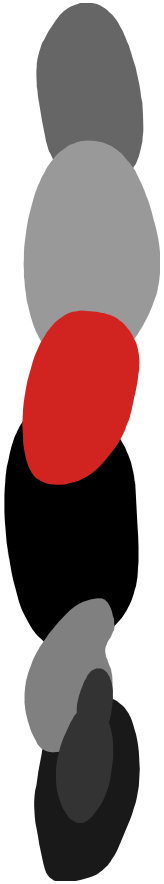
2.- WEKA software:

- 2.1.-Attribute (feature) selection
- 2.3.- Basic algorithms: Naive Bayes, K-NN, Decision Trees, Rules, …
- 2.3.-Evaluation

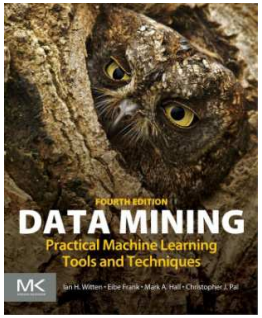
Bibliography



HAP/LAP

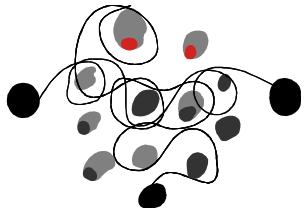


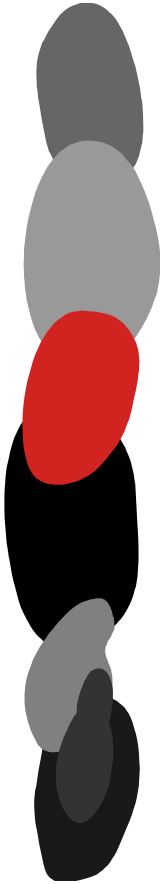
- Data Mining. Mark Hall, Ian Witten and Eibe Frank (3rd Edition) The Morgan Kaufmann, 2011. http://www.cs.waikato.ac.nz/ml/weka



- How to do Linguistics with R. Data exploration and statistical analysis. Natalia Levshina. John Benjamins Publishing Company, 2015
- Fundamentals of Predictive Text Mining (2nd Edition). Weiss, Sholom M., Indurkhya, Nitin, Zhang, Tong. Springer-Verlag London, 2015
- Text Mining. Predictive Methods for Analyzing Unstructured Information. S. M. Weiss, N. Indurkhya, T. Zhang, F. J. Damerau. Springer, 2005

Bibliography



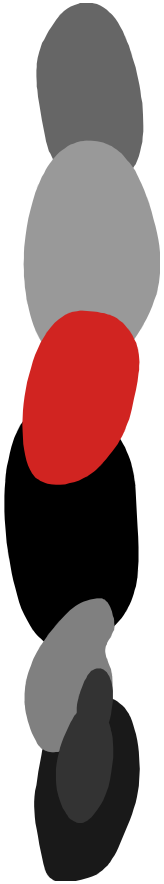


- Survey of Text Mining II. Clustering, Classification, and Retrieval. Michael W. Berry. Springer, 2008. http://www.springerlink.com/content/978-1-84800-045-2

- Text Mining: Applications and Theory (1st Edition). Michael W. Berry , Jacob Kogan . Wiley, 2010
- Aprendizaje automático: conceptos básicos y avanzados. Coord.: B. Sierra. Pearson, Prentice Hall, 2006
- Speech and Language Processing: an introduction to natural language processing, computational linguistics, and speech recognition. 2nd Edition. D. Jurafsky & J.H. Martin. Pearson, Prentice Hall, 2009

  http://www.cs.colorado.edu/%7Emartin/slp2.html

Software



HAP/LAP

Weka
http://www.cs.waikato.ac.nz/ml/weka

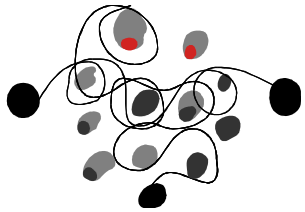
GNU Octave (high-level interpreted language)
http://www.gnu.org/software/octave/

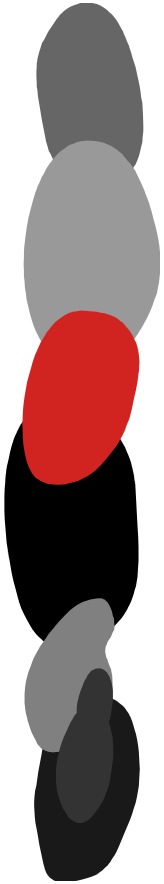
R (an integrated suite of software facilities for data manipulation, calculation and graphical display)
http://www.r-project.org/

RapidMiner (text mining, multimedia mining, feature engineering)
http://rapid-i.com/content/blogcategory/10/69/

Text mining
http://data-miner.com/

Software





C4.5 decision trees
http://www2.cs.uregina.ca/~dbd/cs831/notes/ml/dtrees/c4.5/tutorial.html

SVM light (Support Vector Machine)
http://svmlight.joachims.org/

Kernel (LIBSVM, …)
www.kernel-machines.org
http://mloss.org/software/view/33/

Machine Learning (other)
LIONsolver, KNIME, ODM, Shogun toolbox,

Semi Supervised Learning (Semil)
http://www.learning-from-data.com/te-ming/semil.htm

Software: Text mining

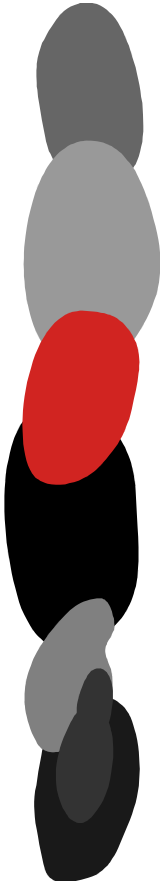
HAP/LAP

Top15: free software

1 General Architecture for Text Engineering – GATE	2 RapidMiner Text Mining Extension	3 KH Coder	4 Coding Analysis Toolkit	5 TAMS
6 VisualText	7 QDA Miner Lite	8 Datumbox	9 Natural Language Toolkit	10 Apache Mahout
11 Twinword	12 Apache UIMA	13 Pattern	14 LingPipe	15 Gensim

https://www.predictiveanalyticstoday.com/top-free-software-for-text-analysis-text-mining-text-analytics/

Courses (on-line)



https://www.coursera.org/

Stanford Unibersity:

http://see.stanford.edu/see/courses.aspx

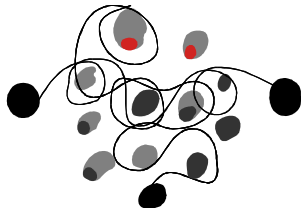
- Natural Language Processing
  - https://www.coursera.org/course/nlp
  - Instructor: Jurafsky and Manning
- Machine Learning
  - https://www.coursera.org/course/ml
  - Instructor: Ng, Andrew

University of Toronto:

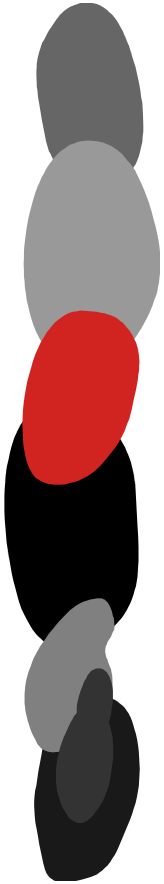
- Natural Language Processing
  - https://www.coursera.org/course/nlangp
    - Instructor: Michael Collins
- Neural Networks for Machine Learning
  - https://www.coursera.org/course/neuralnets
  - Instructor: Geoffrey Hintonek

HAP/LAP

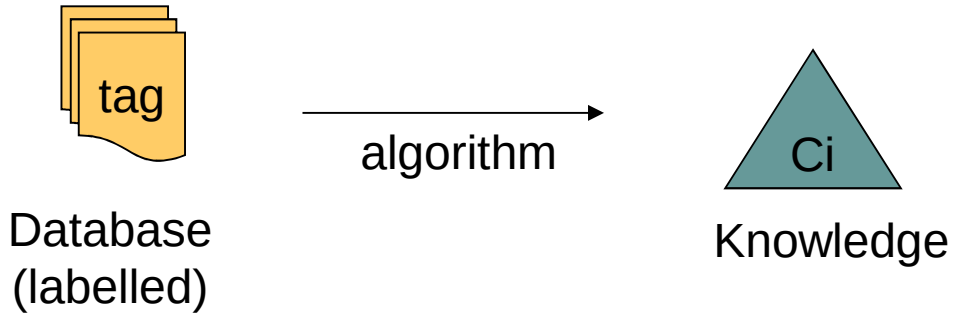
1-Introduction



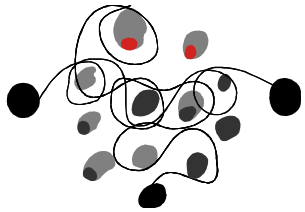
What is machine learning?

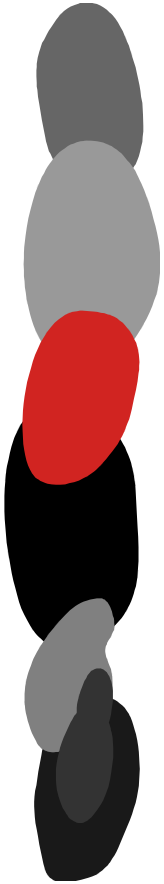


- It is a branch of artificial Intelligence.
- Main objective: to make the computers learn in an automatic way
- To extract knowledge in an automatic way from a concrete data domain
- Computers learn form the information given as input
- To obtain knowledge, it needs a **dataset** as input.
- Analysing data, **using induction,** the computer will obtain conclusions nad build a knowledge pattern **(C_i)**.
- This will be made using **programs or algorithms**.

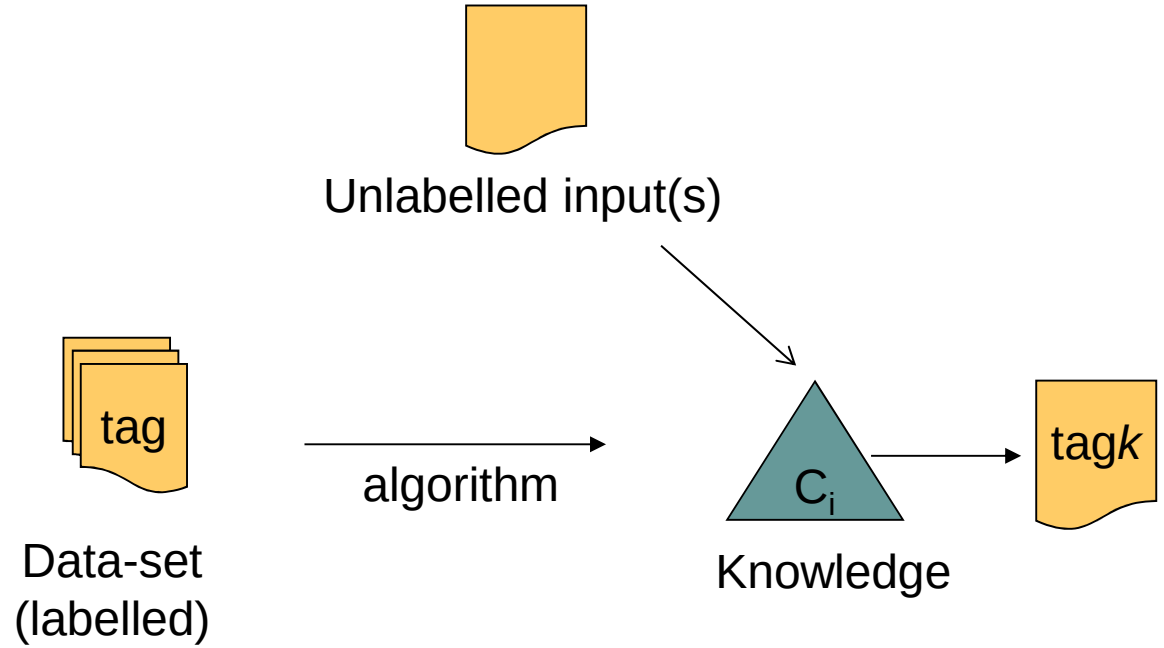


Introduction

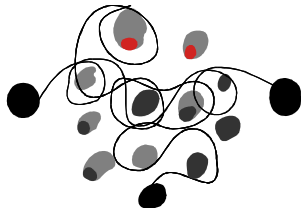




- The algorithms obtain input data, generalize the information found there, and generate models able to **make decisions based on "experience",** to solve the objective problem.
- The generated models can be used to label new examples

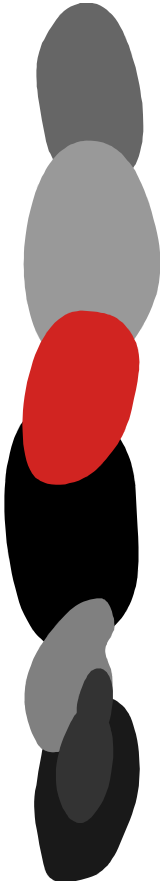


Introduction



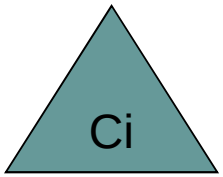
HAP/LAP

Training→ inductive generalization of the input data



Manually labelled
Learning data

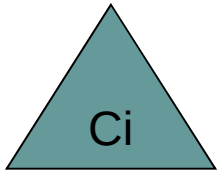
algorithm



Knowledge
(classifier)

Decide how good is the built model *(test)*

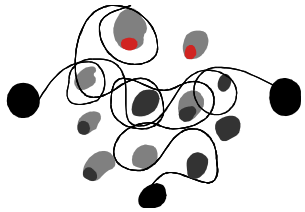
Manually labelled
test data (test)



Knowledge
(Compare
real with
predicted)

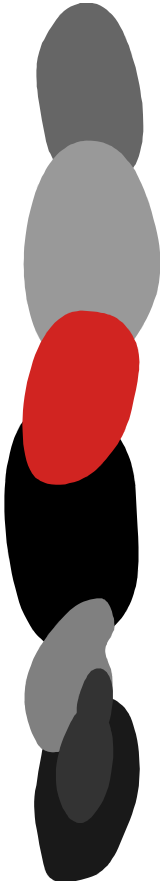
Hit rate (%)

Introduction

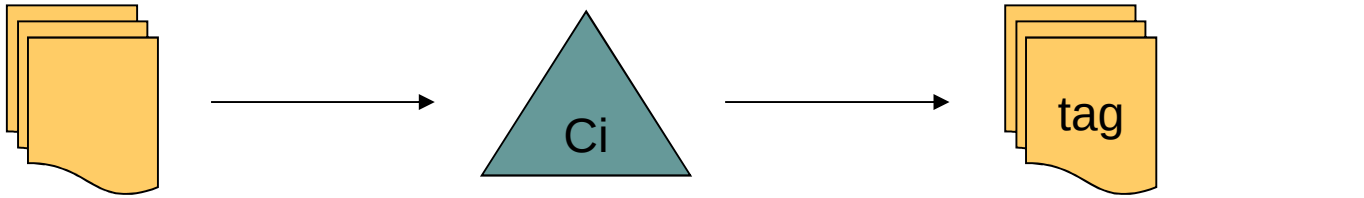


HAP/LAP

Aim→ make decisions about unlabelled data in applications



obtain results

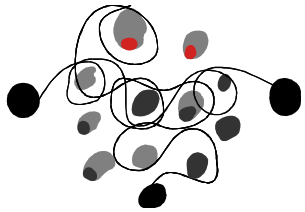


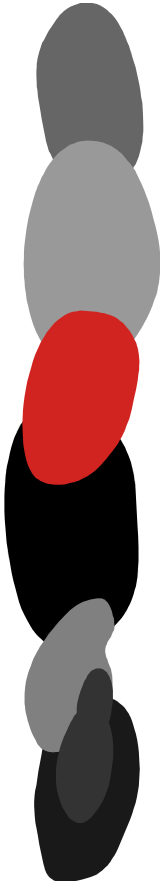
Unlabelled input(s)

Knowledge (classifier)

Automatically labelled data

Introduction

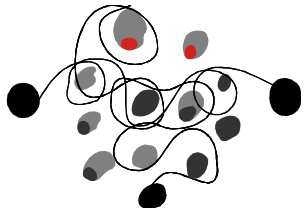




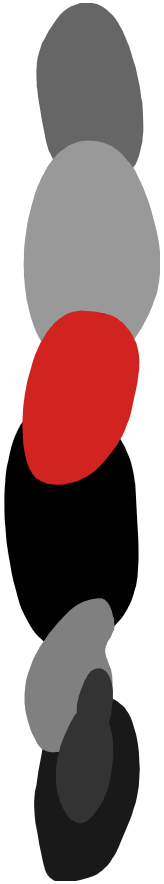
Applications:

- Adaptive websites
- Bioinformatics, Cheminformatics
- Classifying DNA sequences
- Computational anatomy
- Computer vision, including object recognition
- Detecting credit card fraud
- Game playing
- Fraud detection
- Marketing, Online advertising, Stock market analysis
- Medical diagnosis
- Robot locomotion

Introduction

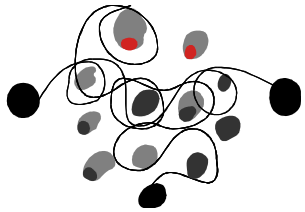


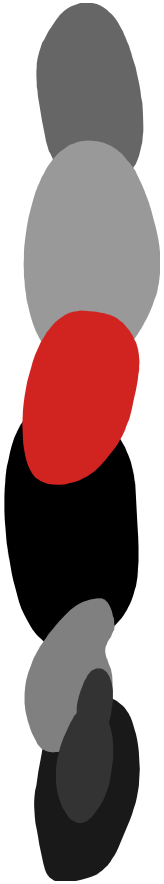
Applications related to Language Processing:



- Information retrieval
- Natural language understanding
- Search engines
- Sentiment analysis (or opinion mining)
- Speech and handwriting recognition
- Syntactic pattern recognition

Introduction





LT Modules

Lexical / Morphological Analysis
PoS Tagging
Chunking
Syntactic Analysis
Word Sense Disambiguation
Semantic Role Labelling
Named Entity Recognition
Semantic Analysis
Coreference Resolution
Discourse Analysis

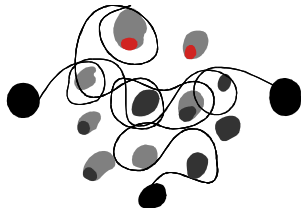
Texts

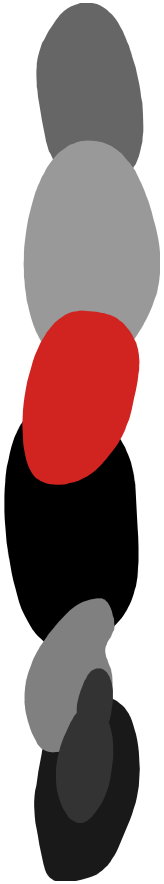
Knowledge

Applications

OCR
Spelling Error Correction
Information retrieval
Document Classification
Grammar Checking
Information Extraction
Summarization
Question Answering
Ontology Extraction
Dialogue Systems
Machine Translation

Introduction





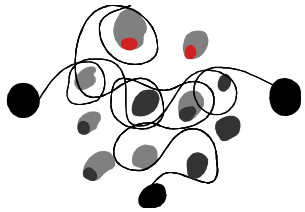
Why it is so useful nowadays:

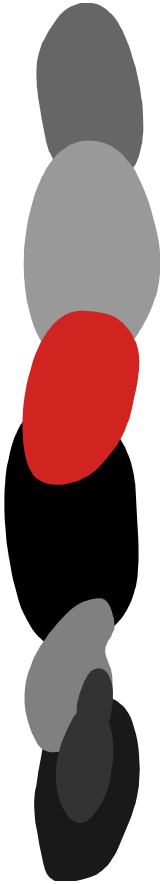
- a lot of data available
- for 2011, 600 Exabytes = 600 x 10^{12} MB
- Only % 0,007 is in paper
-H igh computation capacity

Problems:

- data are not always clean
- complex algorithms, difficult to understand
- a lot and different features to learn → **not enough space in memory**

2.- Machine learning for NLP



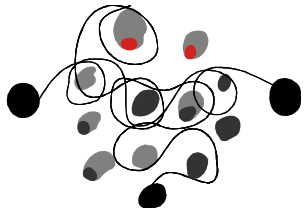


- Machine Learning *(ML)* for Natural Language Processing (NLP)
  - Used in **different parts** of the natural language processing (preprocessing, different tasks…)
  - Process (understand) **natural language**
  - Input data: texts, voice …

To use **machine learning:**

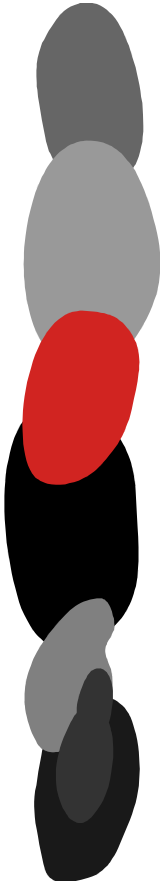
- The objective must be defined: **what to learn**
- Data to learn is required **from where to learn**
  - which corpus
  - which features to use to learn
- A methodology is required **how to learn**
  - Select the right algorithm
- Analyse if what we learnt is correct: **how to evaluate**

Machine learning for NLP



Example: document classification

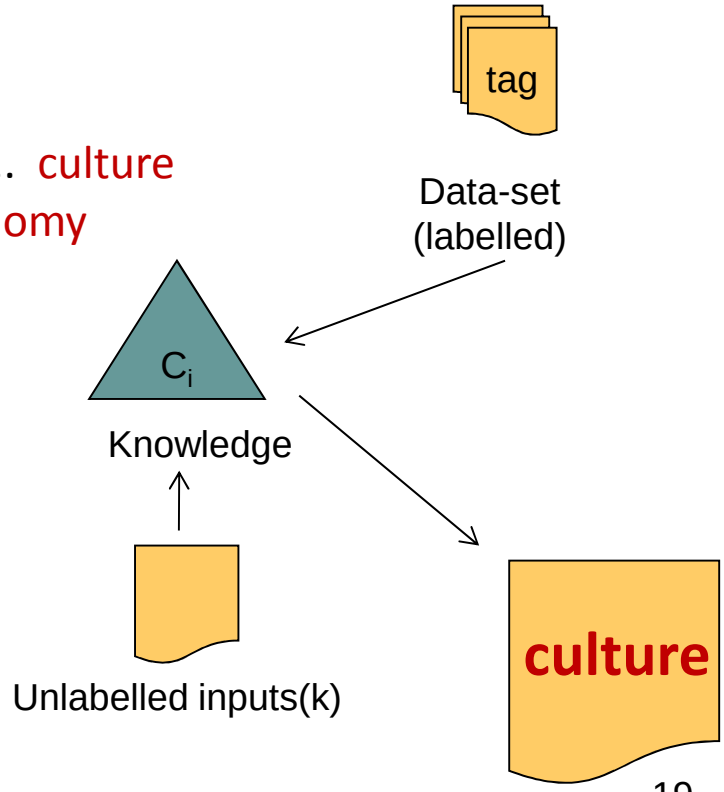
What to learn: category (culture, politics, sports, …)
From where to learn: labelled documents
(newspapers, journals, news agencies…)
How to learn: rules, probabilities, …



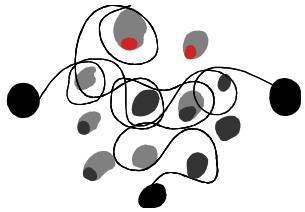
Input (labelled documents)
Generation of a film library in the museum... culture
The plan against the economic crisis economy

Classifier (inductive generalisation)
Film, library, book, art museufm ... culture
Crisis, euro, bank, work. .. economy

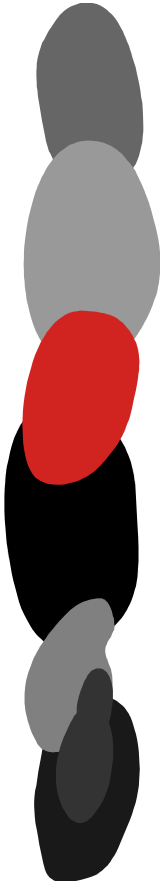
Output (label new unlabelled data)
The film festival will start today with...



Problems



- We will find problems to process and understand language:
  - Mistakes in the input: not found in the dictionary



- Exchanged characters: lsat → last
- Missing characters: prgram→ program

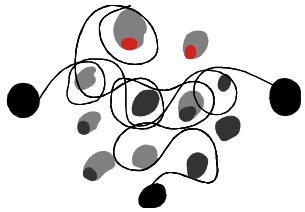
  - Not a lot of importance is given to tokenisation but

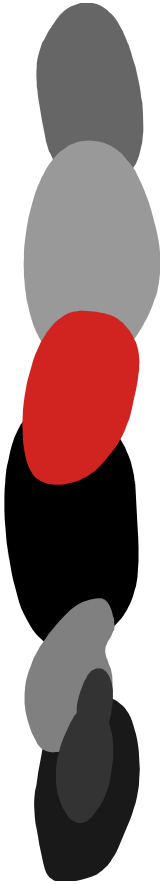
- Blank spaces are not enough to tokenise
- Percentages: 25 % (two tokens?)
- Initials: J.M. / E.H.U.
- Others: in sections a.1, a.2 and a.3

- Ambiguity
  - *I saw the woman with the telescope (2 meanings)*
  - *I saw the woman on the hill with the telescope (4 meanings)*
  - *I saw the woman on the hill in Texas with the telescope (8 meanings)*

We have a very wide knowledge. We use further information than the one we see.

Problems





Ambiguity wherever:

Speech recognition demo

"Recognize speech" vs. "Wreck a nice beach"

"vice president Gore" vs. "dice precedent core"

Syntactic analysis

I ate rice with tomato, vs, I ate rice with fork

Semantic analysis

Michel runs a company vs. Michel runs a marathon

Pragmatic analysis

– Aitor: Does your dog bite?

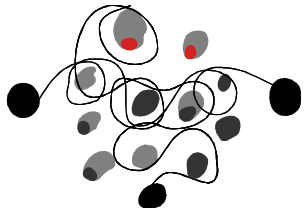
– Mikel: Not.

Aitor is gone to play with the dog and the dog bites him.

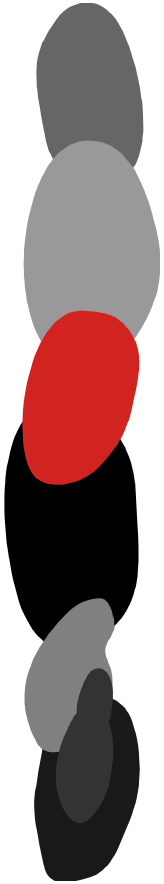
– Aitor: You told me that your dog does not bite

Mikel: That one is not my dog.

Problems



The aim in many tasks is to solve ambiguity



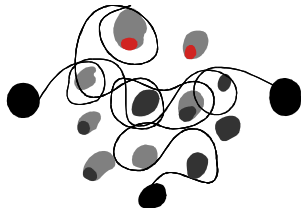
For example to obtain correct translations:
We need to solve syntactic and semantic ambiguity

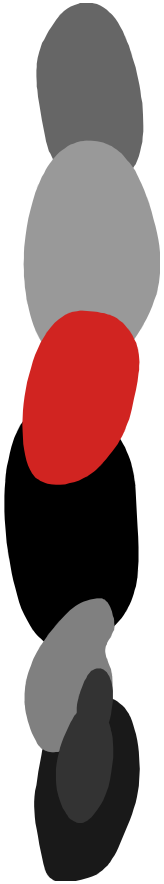
- John plays the guitar → John toca la guitarra
- John plays soccer → John juega al futbol

Knowledge to select the correct interpretation:

- Phonetic/phonologic
- Morphologic/syntactic
- Semantic
- Knowledge of the word/ know how
  - Fork tool to eat
  - Tomato type of food
  - Guitar is a type of musical instrument
  - Football is a game ....

Problems





We might be able to read the following text but a machine…li

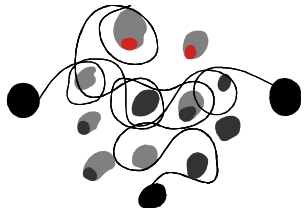
f y cn rd ths thn y r dng btr thn
ny autmtc txt nrmlztion prgrm cn do.

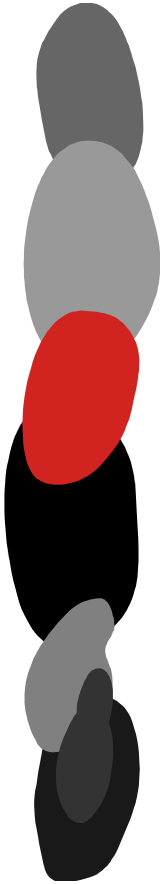
if you can read this then you are doing better than
any automatic text normalization program can do.

And also the next one:

Aoccdrnig to a rscheearch at Cmabrigde Uinervtisy, it deosn't mttaer in what oredr the ltteers in a wrod are, the olny iprmoetnt tihng is taht the frist and lsat ltteer be at the rghit pclae. The rset can be a total mses and you can sitll raed it wouthit porbelm. Tihs is bcuseae the huamn mnid deos not raed ervey lteter by istlef, but the wrod as a wlohe.

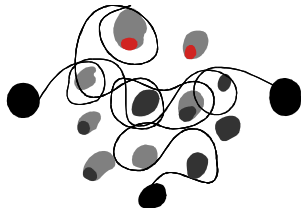
Machine learning for NLP





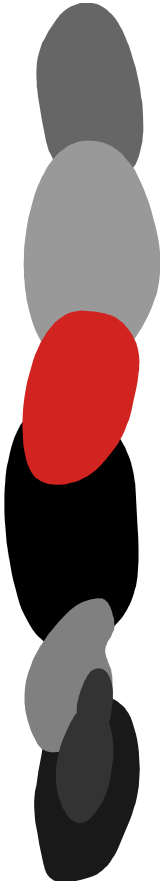
- We need to answer some questions:
  - **What** to learn? We need to have an **objective**
  - **Where** to learn it **from**? **Data** is required
    - Corpora
  - **How** to **represent** it?
    - Which characteristics to use to learn features, weight, …
  - **How** to **learn**? A methodology is required
    - Selection of the adequate algorithm
  - **How** to **evaluate**? Is it correct what we learnt?

What to learn?

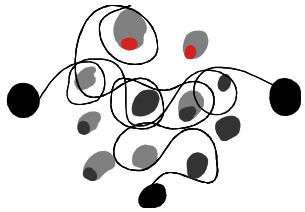


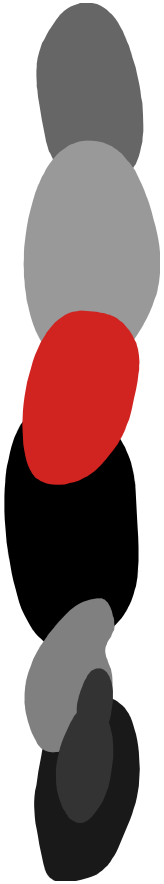
- The program will always have an **objective**:
  - Category of the text
    - Health, economy, politics, …
  - Syntactical function of the words
    - Subject, object, verb,
  - Word sense disambiguation
    - Glass = material
    - Glass = container for drinks
  - Finding and classifying entities
    - place, person,
  - Number classification
    - dates, %, roman, …

Classification problems



From where to learn? Corpora

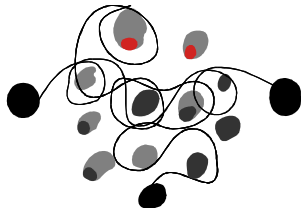


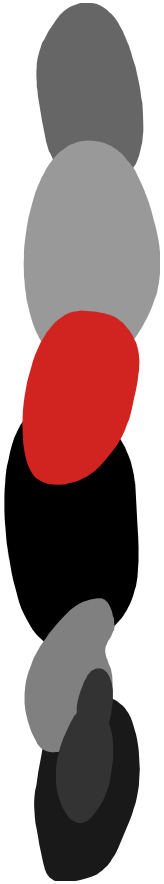


- The machine needs corpora (labelled data) to learn:
  - For each task data needs specific labels
  - When bigger the amount of data better will the system learn
  - The situation is very different depending on the language
    - English (great volume)
    - Basque limited (small community language)
    - Generating corpora is expensive (prepare and validate)
  - Generated for competitions (task dependent)
    - SemEval (**Sem**antic **Eval**uation)
    - TREC (**T**ext **RE**trieval **C**onference)
    - MUC (**M**essage **U**nderstanding **C**onference)
    - TAC (**T**ext **A**nalysis **C**onference)

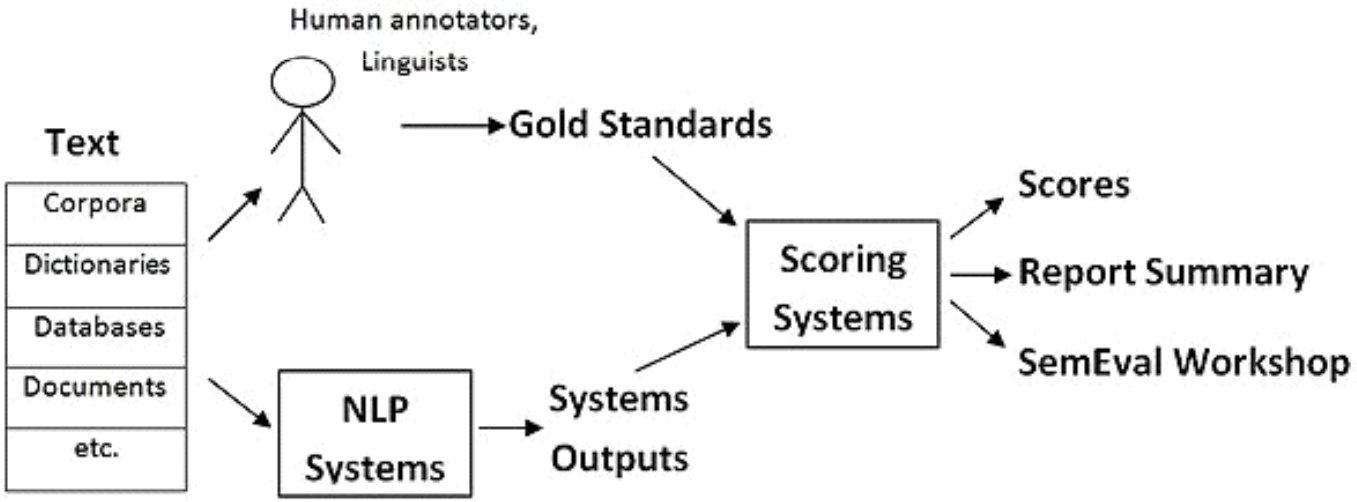
  Corpus generation requires craft work (linguists)

From where to learn? Corpora

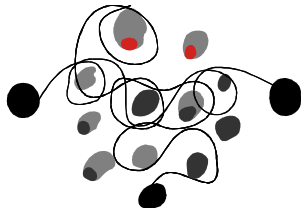




- Often generated for contexts (depend on tasks)
  - SemEval (**Sem**antic **Eval**uation)
  - Craft work to generate copora

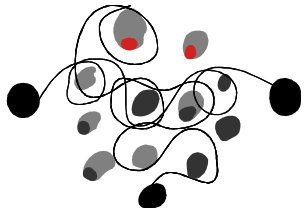


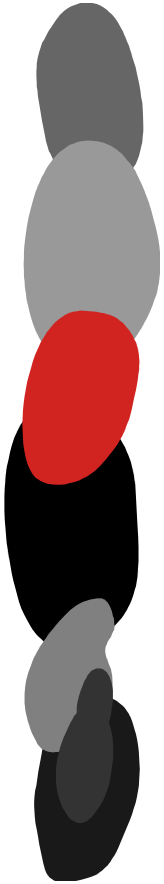
From where to learn? corpora



- Reuters Corpora (Reuters news agency)
- http://trec.nist.gov/data/reuters/reuters.html
  - Very used in text classification
    - RCV1 (Reuters Collection Volumen 1) 1996-1997
      - English
      - 810,000 Reuters News stories.
      - It requires about 2.5 GB for storage of the uncompressed files.
    - RCV2 (Reuters Collection Volume 2) 1996-1997
      - Multilingual
      - 487,000 Reuters News stories
      - Thirteen languages (Dutch, French, German, Chinese, Japanese, Russian, Portuguese, Spanish, Latin American Spanish, Italian, Danish, Norwegian, Swedish).
      - The stories are NOT PARALLEL, but are written by local reporters in each language.
    - TRC2 (Thomson Reuters Text Research Collection) 2008-2009
      - 1,800,370 news stories

From where to learn? corpora





TREC: Text Retrieval and Evaluation Conferences collections
http://trec.nist.gov/

- Selections from the Wall Street Journal, the New York Times, Ziff-Davis Public. …
- Documents for different tasks
- Question-answering, Spam, …

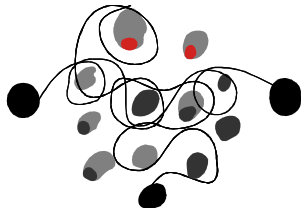
Text Analysis Conference-2015

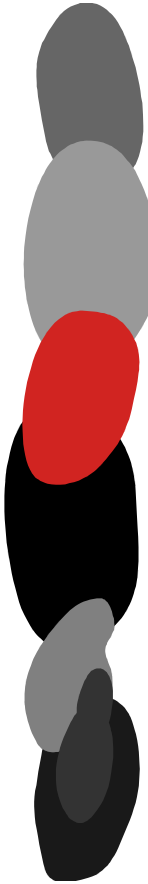
- Entity-Discovery and Linking: English, Chinese, Spanish
- Slot-Filling: : English, Spanish

SemEval-2016-2017

- Textual Similarity and Question Answering
- Sentiment Analysis
- Semantic Parsing, Semantic Analysis, Semantic Taxonomy
- Learning sense of humor

From where to learn? corpora

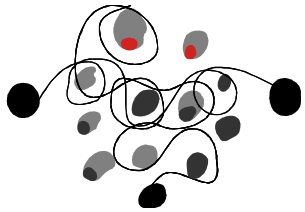


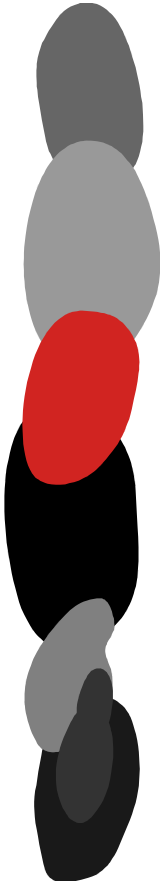


- **MEDLINE/P:** Data set from the National Institute of Health
  - abstracts on medical subjects parsed and indexed

  http://www.bioinformatics.nl/biometa/medlineparsed.html
- **LDC:** The Linguistic Data Consortium
  - An open consortium of universities, companies and government research laboratories.
  - It creates, collects and distributes speech and text databases, lexicons, and other resources for research and development purposes.
  - https://catalog.ldc.upenn.edu/LDC2013T19
- **Penn Tree Bank:**
  - Manually parsed sentences from the Wall Street Journal
  - http://www.cis.upenn.edu/~treebank/

From where to learn? corpora



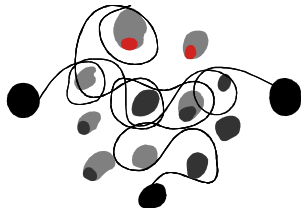


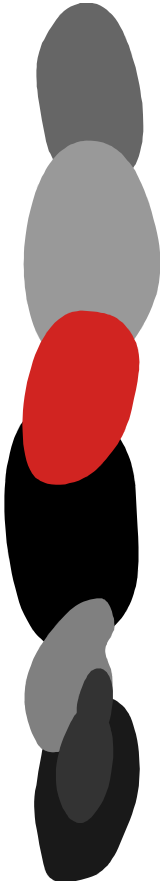
Problems

- Size of corpora → many features
  - Lack of labelled corpora → need of labelling manually?
  - Where to get the data from? publishers, television, web...

    Be careful! permissions, types of licences,...
  - The method used to learn affects to the result
- *data-sparseness*
  - Requires a lot of memory and the information is not always meaningful (half of the different words we can find in a book appears only once)
- Quality of the corpora:
  - Many not meaningful features
  - Errors and noise
  - Examples not in the corpora: probability 0 ???

From where to learn: instances





- What does corpora have?
  - **Instances** or examples of what we want to learn
  - Depending of the aim of the learning process the instances will have different information

    Text Classification → documents

    Word Sense Disambiguation → paragraphs or parts of documents containing the word to be disambiguated

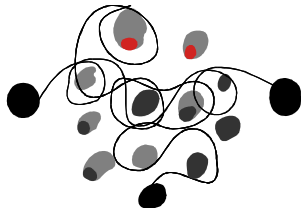
    Named Entity → entities and surrounding words

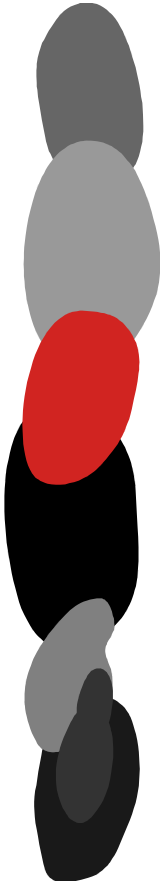
    Syntactic Function → words do not have enough information, the same word can have different SF. Additional information is required : PoS, NE, …

    …

    Semantic Role Labeling, Coreference, temporal structures, …

From where to learn: features





What do instances have?

Instances are composed of **features**, that is, the data or characteristics we use to define the example.

The **Class** or the category is required

Depending on the learning objective:

Text Classification → documents→ words, lemmas, …

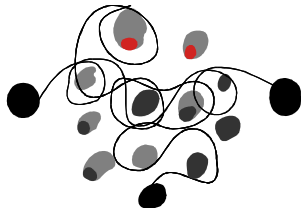
Syntactic Function → phrases→ words, cat, ...

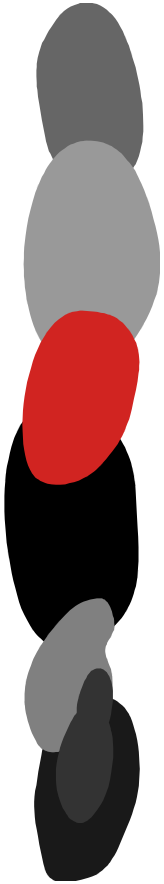
Input: (TC features--> words)

The cyclist of Sky answerd to the questions …

Cylce, Sky, quest, …. **sports**

From where to learn: features





features

Original: (BoW, phonemes, ...)

Texts are not directly treated, they require conversion:

Calculations: obtained preprocessing the original corpus

lemma (stemming), category, case, capital letter, …

Distance from the verb, length, ...

Complete text : BoW (TC)

Text windows : global or local context (NE, WSD, punctuation, ...)

w_{-2}, w_{-1}, w_0, w_{+1}, w_{+2}

Feature value vectors

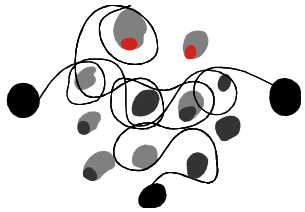
Category=NAME, ADJ, VERB,

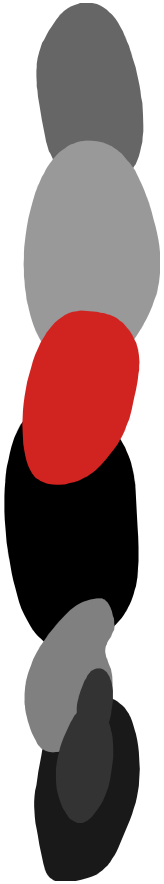
subcategory=COM,

number=S, P, UNC

…

From where to learn: features





- Document classification.

  Input (labelled documents)

  The cyclist of Sky answered to the questions of the press in the hospital room... SPORT

- Finding phrases within a text.

  Input (labelled documents)

  [The cyclist of Sky]$_{NPH}$ answered to [the questions of the press]$_{NPH}$ in [the hospital room]$_{NPH}$ ...

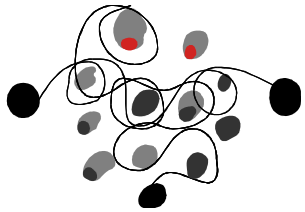
- Identification of the syntactic function of the words

  Input (labelled documents)

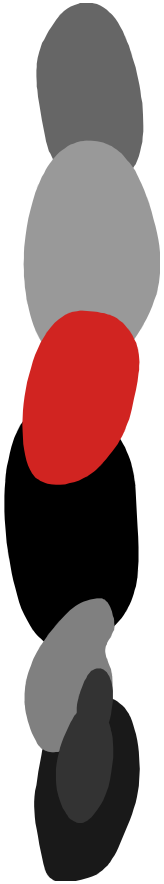
  [The cyclist of Sky]$_{SUBJ}$ answered to [the questions of the press]$_{OBJ}$ in the hospital room ...

- Word sense disambiguation
- Web page classification

3.-WEKA

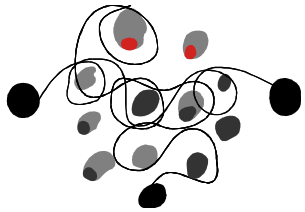


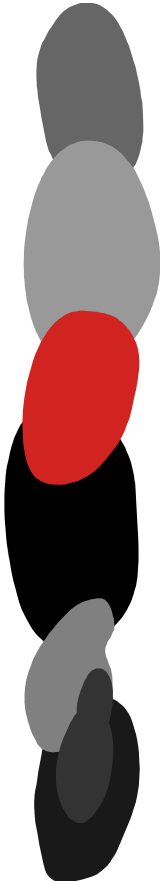
WEKA: tool for machine learning



- 3.0 Introduction
- 3.1 Feature selection
- 3.2 Basic classifiers
  - Naive Bayes,
  - K-NN
  - Classification trees
  - Rules
  - …

WEKA software

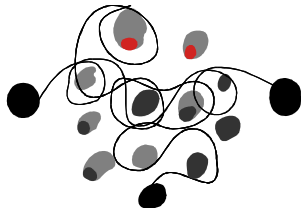




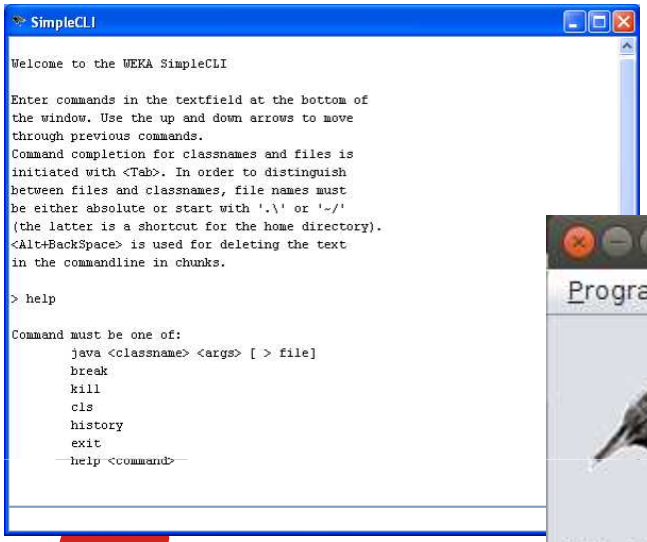
- Machine Learning tool written in Java (GNU license)
- Many types of algorithms:
  - Directly applied to a set of data
  - Possible to call them from your own java program

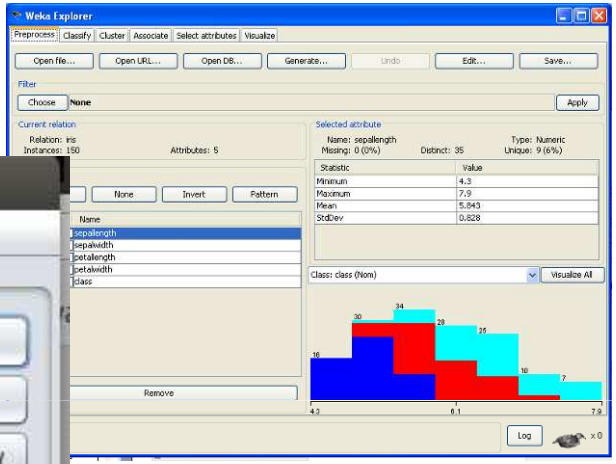
- Main characteristics:
  - Graphical interface
  - Option to preprocess data (feature selection, …)
  - A lot of learning algorithms (classification, clustering, …)
  - Different evaluation methods ...
- Parts:
  - Data processing part Explorer
  - Part to start experiments Knowledge Flow
  - Method comparison part Experimenter
  - All together Workbench

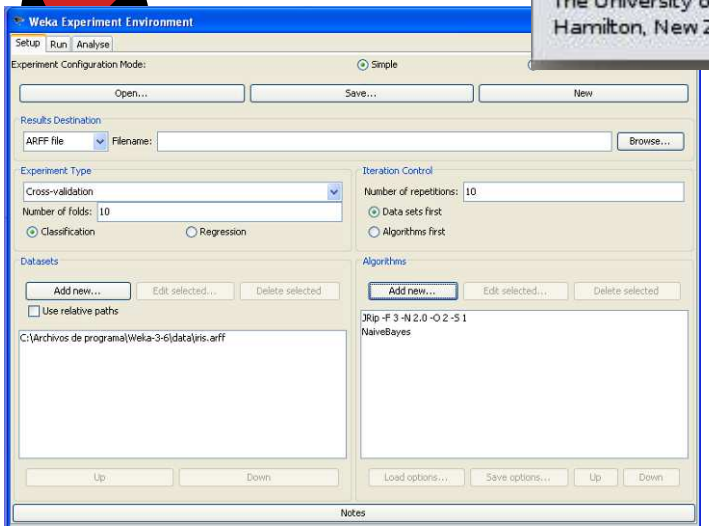
WEKA: tool for machine learning

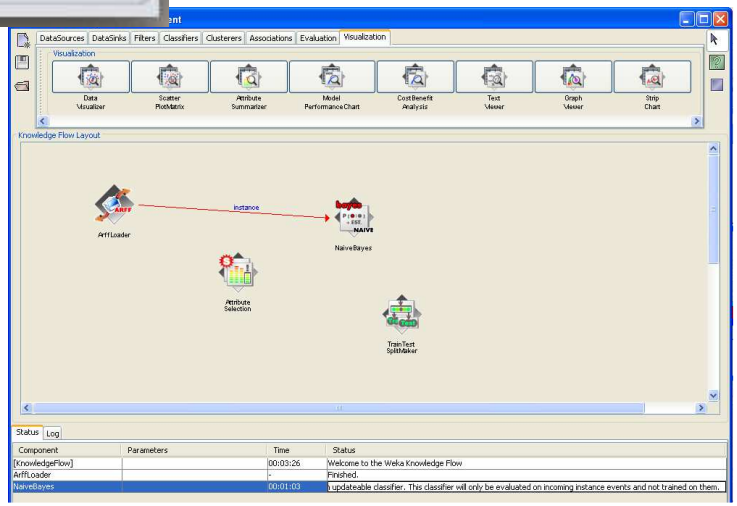


HAP/LAP

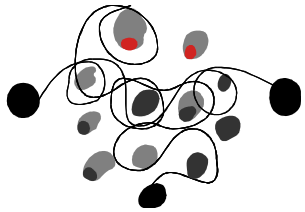


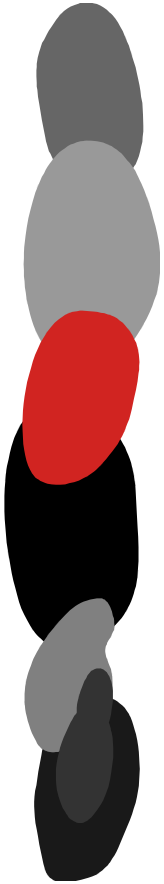






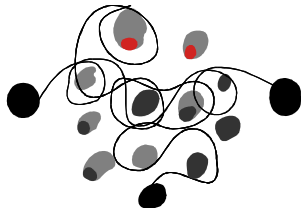
WEKA: bibliography

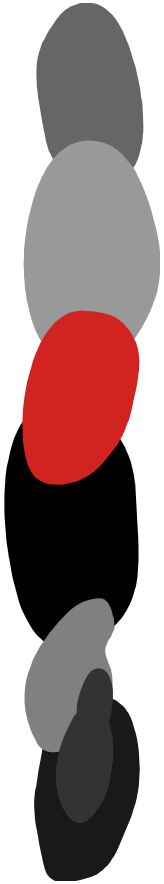




- Book:
  Data Mining Ian H. Witten, Eibe Frank , Mark A, Hall (3. arg)
- Software
  http://www.cs.waikato.ac.nz/ml/weka
- Weka's wiki
  http://weka.wikispaces.com/
- Programs, data-files, …
  http://www.hakank.org/weka/index.html
  (arff files, programs, ...)
  http://archive.ics.uci.edu/ml/
  (264 databases for machine learning)
  http://www.kdnuggets.com/datasets/index.html
  (websites with databases)

WEKA: introduction





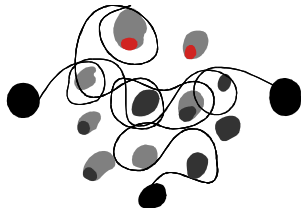
- Input files:
  - Format: ARFF
  - Accessible in the Internet:
    - http://www.hakank.org/weka/index.html
    - http://archive.ics.uci.edu/ml/

Data Set Characteristics:	Multivariate	**Number of Instances:**	569	**Area:**	Life
Attribute Characteristics:	Real	**Number of Attributes:**	32	**Date Donated**	1995-11-01
Associated Tasks:	Classification	**Missing Values?**	No	**Number of Web Hits:**	579838

  - Data from a URL (address where the dataset is stored)
  - From SQL datsets (weka/experimenter)

Data files:

http://www.hakank.org/weka/index.html



HAP/LAP

ARFF data files

The data file normally used by Weka is in ARFF file format, which consist of special tags to indicate different things in the data file (foremost: attribute names, attribute types, attribute values and the data).

Here is a list of some ARFF-file you can use, many are standard data sets often used in the machine learning community. Most of them are available from the Weka site. Many of them are also described and downloadable from http://www.ics.uci.edu/~mlearn/MLRepository.html.

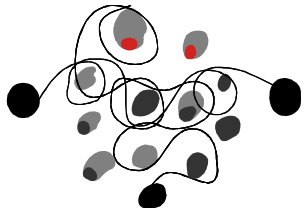
If you click on the link in the list below you can see for yourself what the data set looks like. Please note that some files are quite big, and for some algorithms it will take a lot of time (often **a lot** of time!). The number in parenthesis is the size in bytes. In some of the files there are quite good comments for the data set, other has no explanation at all (they are probably converted from some other source by myself).

One more thing: The class attribute (i.e. the attribute we want to learn) must be the last.

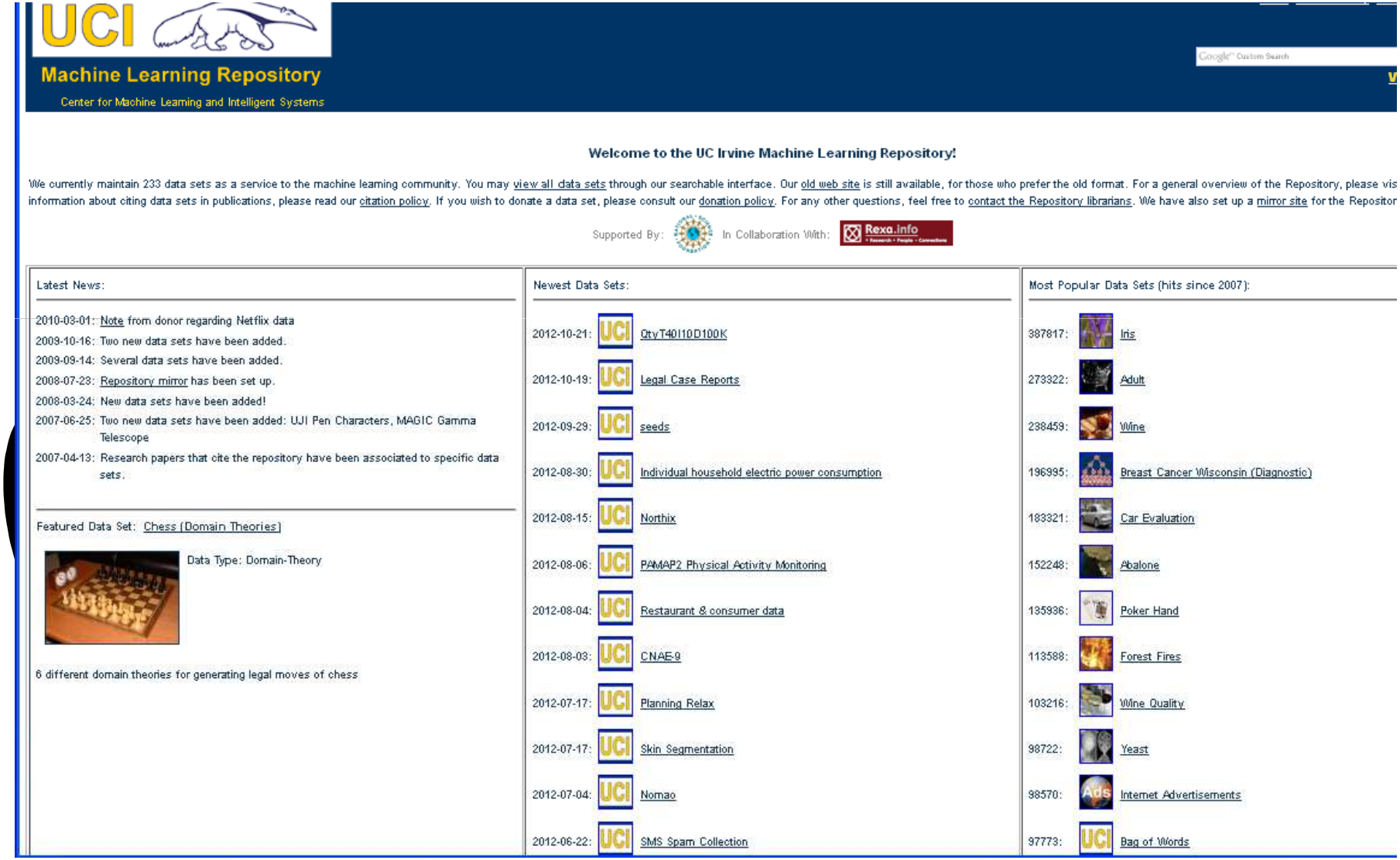
- http://www.hakank.org/weka/zoo2_x.arff (6296)
- http://www.hakank.org/weka/golf.arff (383)
- http://www.hakank.org/weka/cpu.arff (6936)
- http://www.hakank.org/weka/sunburn.arff (573)
- http://www.hakank.org/weka/wine.arff (13790)
- http://www.hakank.org/weka/iris_discretized.arff (12390)
- http://www.hakank.org/weka/shape.arff (296)
- http://www.hakank.org/weka/titanic.arff (42322)
- http://www.hakank.org/weka/disease.arff (457)
- http://www.hakank.org/weka/labor_discretized.arff (9595)
- http://www.hakank.org/weka/zoo.arff (9408)
- http://www.hakank.org/weka/monk3.arff (1944)

Data files:

http://archive.ics.uci.edu/ml/



HAP/LAP



UCI

Machine Learning Repository

Center for Machine Learning and Intelligent Systems

Welcome to the UC Irvine Machine Learning Repository!

We currently maintain 233 data sets as a service to the machine learning community. You may view all data sets through our searchable interface. Our old web site is still available, for those who prefer the old format. For a general overview of the Repository, please vis
information about citing data sets in publications, please read our citation policy. If you wish to donate a data set, please consult our donation policy. For any other questions, feel free to contact the Repository librarians. We have also set up a mirror site for the Repositor

Supported By: In Collaboration With: Rexa.info

Latest News:

2010-03-01: Note from donor regarding Netflix data
2009-10-16: Two new data sets have been added.
2009-09-14: Several data sets have been added.
2008-07-23: Repository mirror has been set up.
2008-03-24: New data sets have been added!
2007-06-25: Two new data sets have been added: UJI Pen Characters, MAGIC Gamma Telescope
2007-04-13: Research papers that cite the repository have been associated to specific data sets.

Featured Data Set: Chess (Domain Theories)

Data Type: Domain-Theory

6 different domain theories for generating legal moves of chess

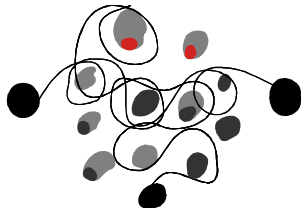
Newest Data Sets:

2012-10-21: QtyT40I10D100K
2012-10-19: Legal Case Reports
2012-09-29: seeds
2012-08-30: Individual household electric power consumption
2012-08-15: Northix
2012-08-06: PAMAP2 Physical Activity Monitoring
2012-08-04: Restaurant & consumer data
2012-08-03: CNAE-9
2012-07-17: Planning Relax
2012-07-17: Skin Segmentation
2012-07-04: Nomao
2012-06-22: SMS Spam Collection

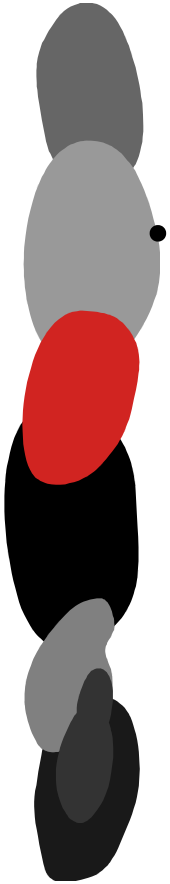
Most Popular Data Sets (hits since 2007):

387817: Iris
273322: Adult
238459: Wine
196995: Breast Cancer Wisconsin (Diagnostic)
183321: Car Evaluation
152248: Abalone
135936: Poker Hand
113588: Forest Fires
103216: Wine Quality
98722: Yeast
98570: Internet Advertisements
97773: Bag of Words

ARFF format

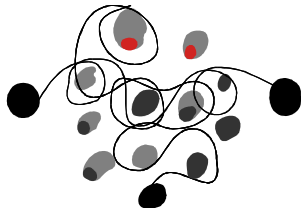


- Input files:
  - Format: ARFF (from excel: http://exceltoarffconv.sourceforge.net/)
  - Data can be read from URLs (address where the dataset is stored) or SQL data-bases (weka/experimenter)



- ARFF (Attribute-Relation File Format)
  - **Header**:
    - Title @relation <relation-name>
    - features, feature-type @attribute <attribute-name> <datatype>
      - *numeric* (*integer, real*)
      - {value1, value2, …}: nominal, list of possible values
      - *string*: to work with text files (filters are required)
      - *date:* dates
    - The last feature is the category or class @attribute <attribute-name> <datatype>
  - **Data**
    - Instances or examples @data

WEKA: input files (.arff)



```
@relation heart-disease-simplified

@attribute age numeric
@attribute sex {female, male}
@attribute chest_pain_type {typ_angina, asympt,
   non_anginal, atyp_angina}
@attribute cholesterol numeric
@attribute exercise_induced_angina {no, yes}
@attribute class {present, not_present}

@data
63,male,typ_angina,233,no,not_present
67,male,asympt,286,yes,present
67,male,asympt,229,yes,present
38,female,non_anginal,?,no,not_present
```

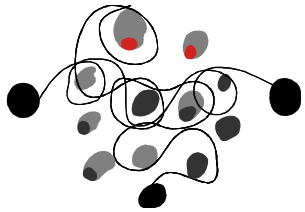
Numeric features

Nominal

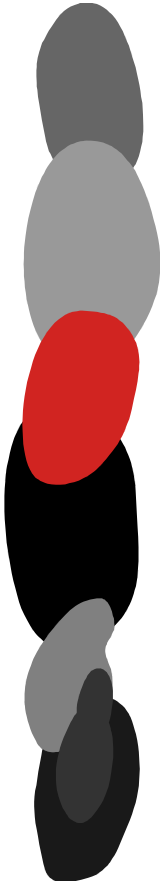
Class

Instances

WEKA: example(.arff)

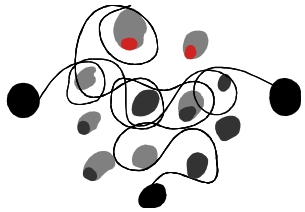


HAP/LAP



```
Euskaltel "Euskaltel" IZE LIB ENTI_HAS_ORG HAS_MAI KM> %SIH
Euskadiko "Euskadi" IZE LIB GEL NUMS ENTI_BUK_ORG HAS_MAI IZLG> %ZERO
txirrindulariak  "txirrindulari" IZE ARR ERG NUMS SUBJ %SIB
prentsaurrekoa "prentsaurreko" IZE ARR ABS NUMS OBJ %SINT
eman "eman" ADI SIN PART BURU NOTDEK -JADNAG %ADIKATHAS
zuen "*edun" ADL B1 NOR_NORK NR_HURA NK_HARK +JADLAG %ADIKATBU
atzo "atzo" ADB ARR ZERO ADLG %SINT

 @relation  Izen sintagmak
 @attribute lema string
 @attribute kategoria {IZE, ADI, ADB, ADJ, ADL, ...}
 @attribute azpikategoria {ARR, LIB, B1 ...}
 @attribute numeroa {MUGM, NUMS, NUMP, ...}
 @attribute funtzio_sintaktikoa {SUBJ, OBJ, IZLG, KM, JADNAG ...}
 @attribute class {SINT, SIH, SIB, ZERO}

@data
   'Euskaltel' IZE LIB ? KM> SIH
   'Euskadi' IZE LIB NUMS ZERO
   'txirrindulari' IZE ARR NUMS SUBJ SIB
   'prentsaurreko' IZE ARR NUMS OBJ SINT
```

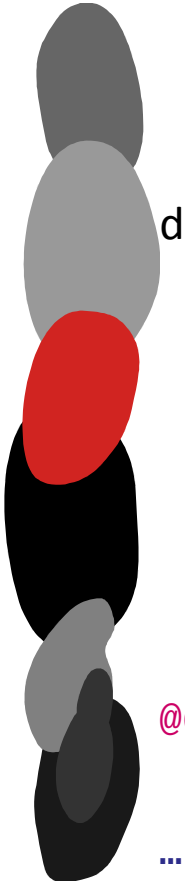
WEKA: example(.arff)



HAP/LAP

doc1:

German Chancellor Angela Merkel has re-affirmed her country's responsibility for the Holocaust, following controversial comments by Israel's prime minister.
Benjamin Netanyahu was criticized for saying Adolf Hitler had only wanted to expel Jews from Europe but that a Palestinian leader, the Grand Mufti of Jerusalem Haj Amin al-Husseini, told him to "burn **POLITICS**



doc2:

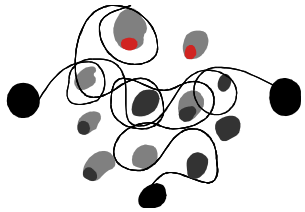
Britain's defending Tour de France champion Chris Froome has welcomed a "great" 2016 route which should help his bid for a third victory.
The course for the 2,187-mile race, which runs from 2-24 July, was announced in Paris on Tuesday. **SPORTS**

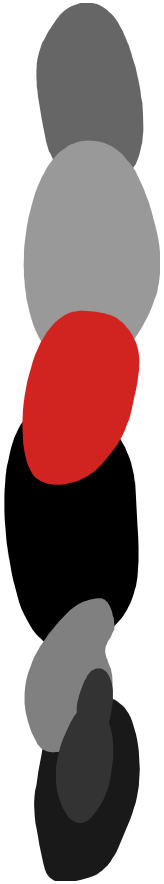
Which are the features?

How are they defined?

```
@relation  topics
@attribute doc string
@attribute class {sports, politics}

@data
    'German Chancellor Angela Merkel has re-affirmed...'  POLITICS
    'Britain's defending Tour de France champion Chris Froome …'  SPORTS
…
```

WEKA: simple example(.arff)



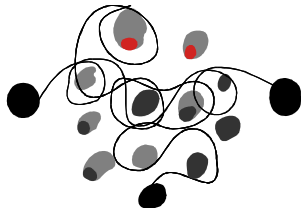


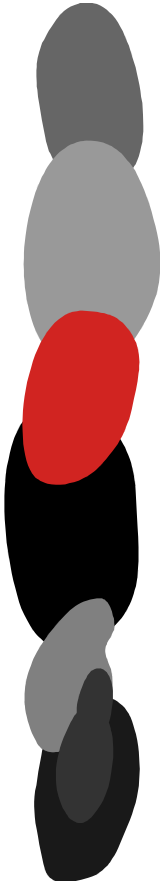
```
@RELATION iris
@ATTRIBUTE sepallength      REAL
@ATTRIBUTE sepalwidth       REAL
@ATTRIBUTE petallength      REAL
@ATTRIBUTE petalwidth       REAL
@ATTRIBUTE class {Iris-setosa,Iris-versicolor,Iris-virginica}
@DATA
5.1,3.5,1.4,0.2,Iris-setosa
4.9,3.0,1.4,0.2,Iris-setosa
...
7.0,3.2,4.7,1.4,Iris-versicolor
6.4,3.2,4.5,1.5,Iris-versicolor
...
6.3,3.3,6.0,2.5,Iris-virginica
5.8,2.7,5.1,1.9,Iris-virginica
...
```

sepallength

class

iris.arff (summary)



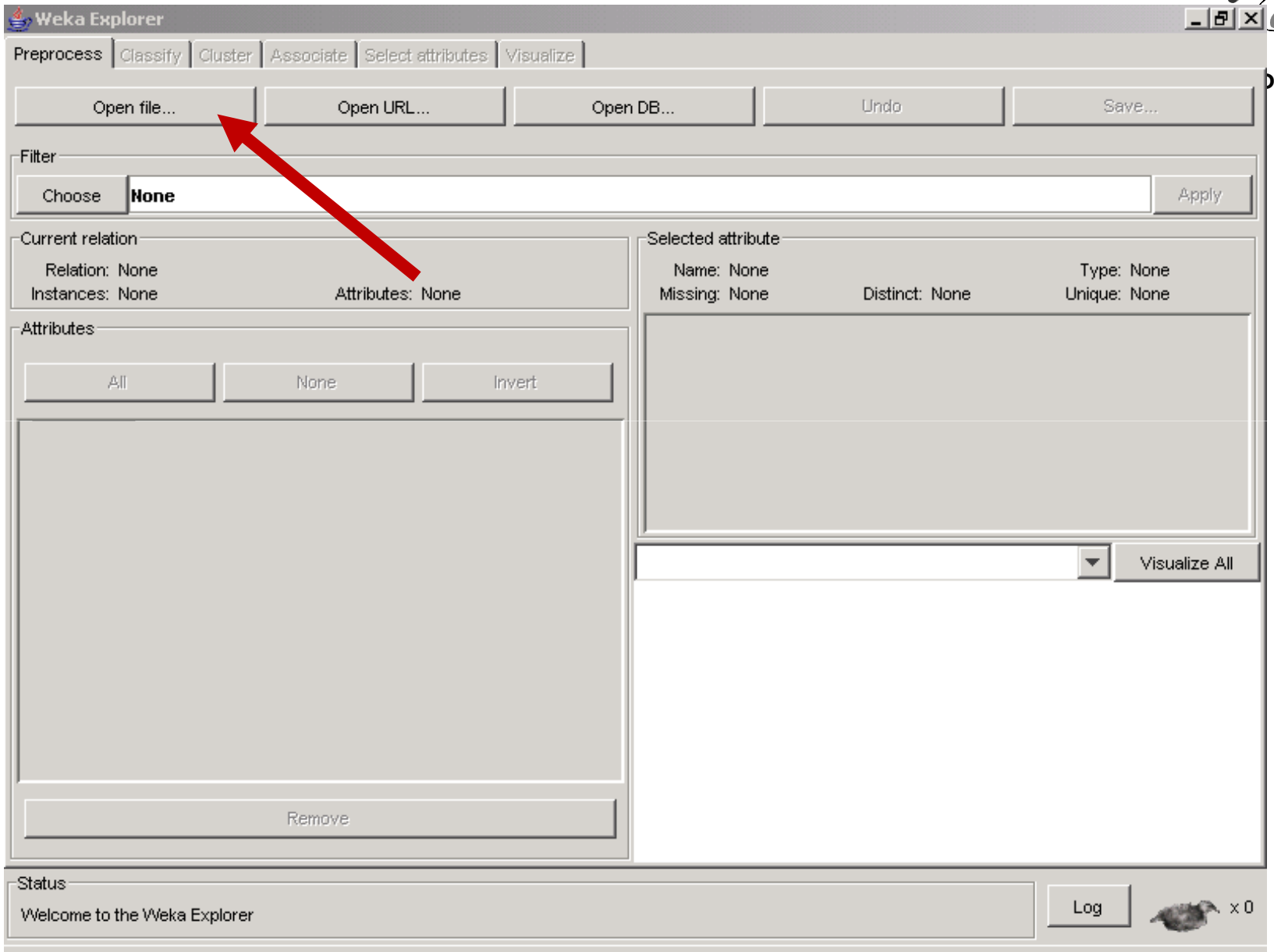


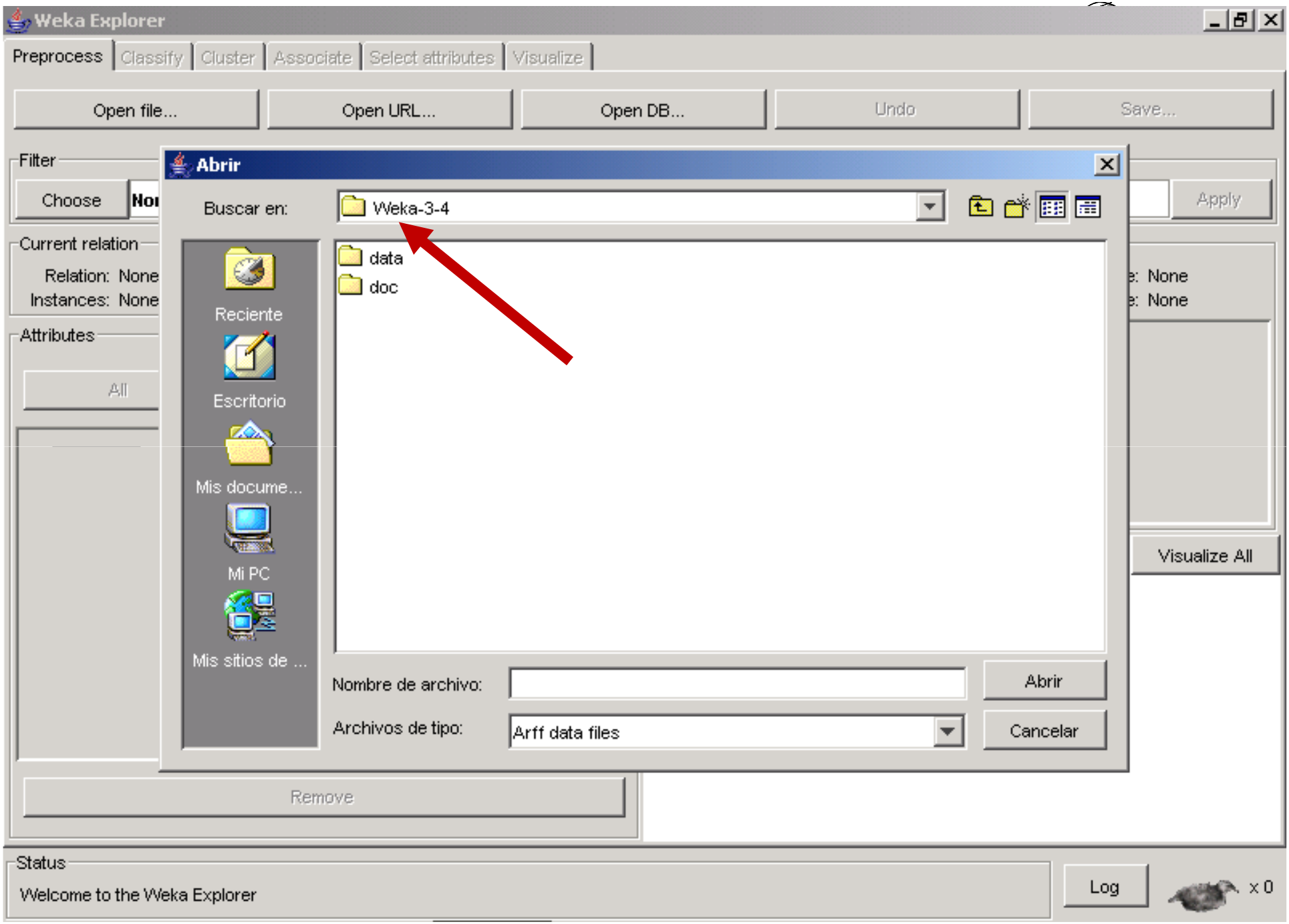
```
Attribute Information:
   1. sepal length     in cm
   2. sepal width      in cm
   3. petal length     in cm
   4. petal width      in cm
   5. class: Iris Setosa  Iris Versicolour Iris Virginica
```

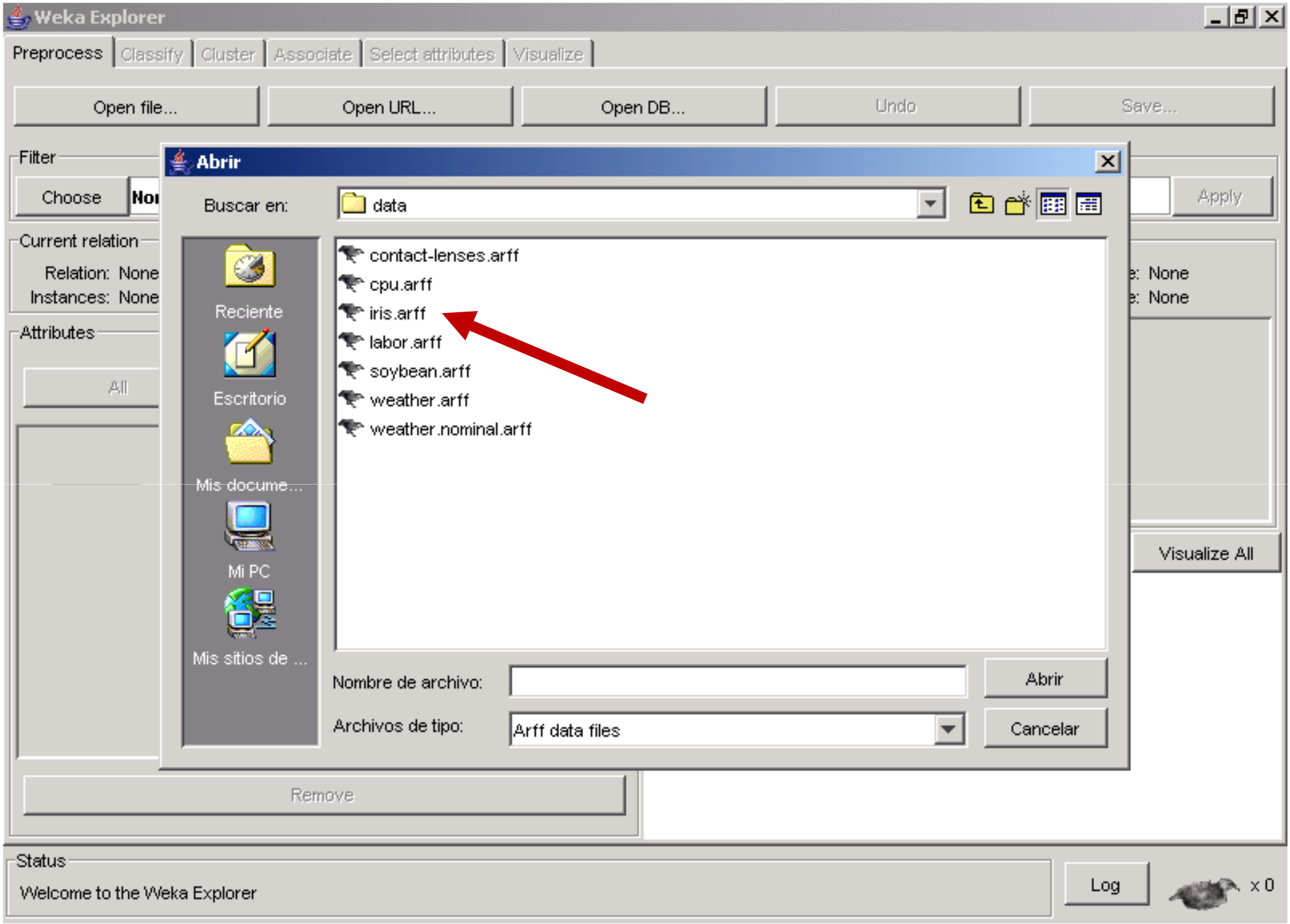
4 features
3 classes
150 instances (examples)

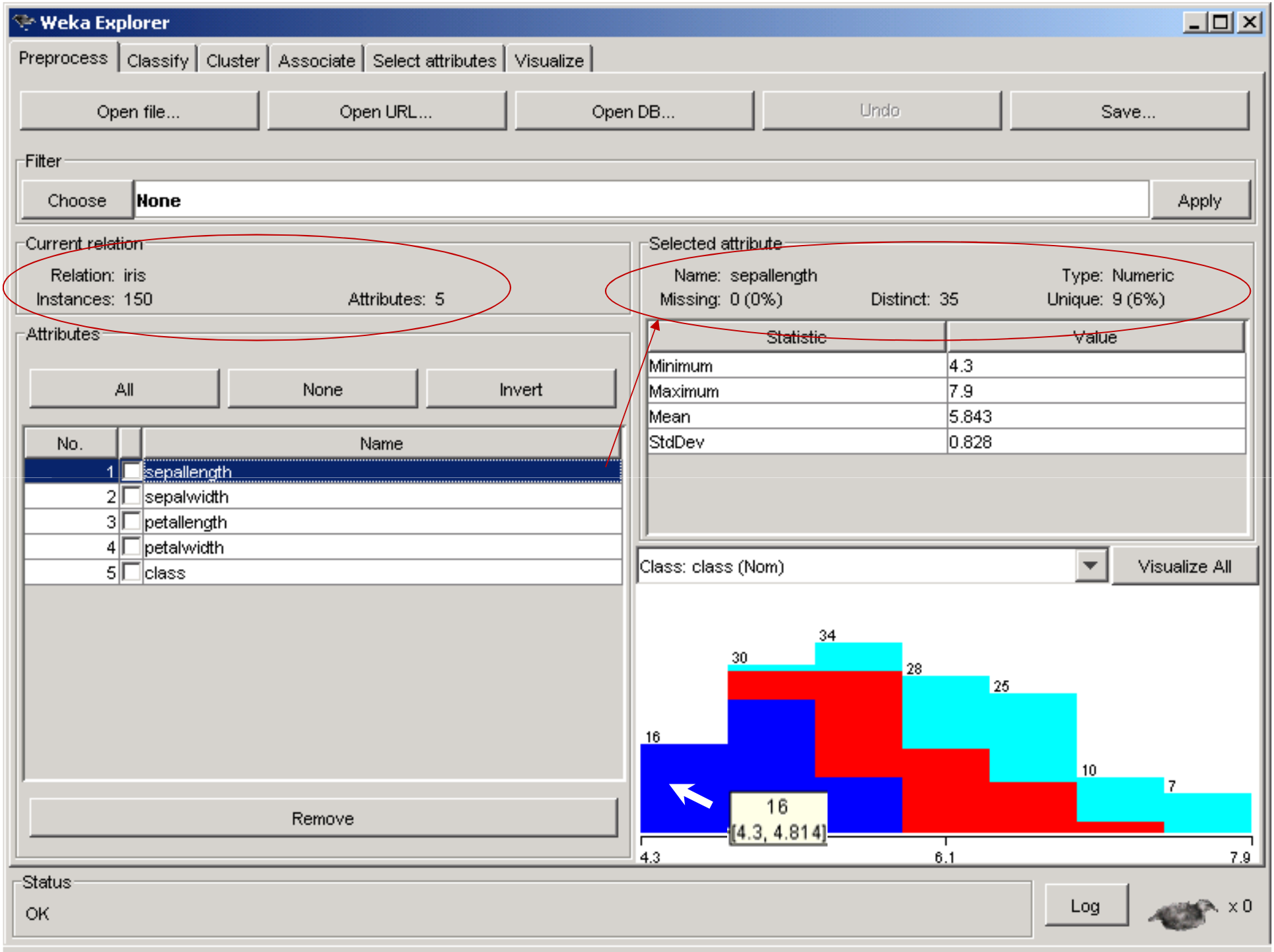
```
Summary Statistics:
                 Min   Max   Mean    SD  Class Correlation
sepal length:    4.3   7.9   5.84   0.83    0.7826
sepal width:     2.0   4.4   3.05   0.43   -0.4194
petal length:    1.0   6.9   3.76   1.76    0.9490 (high!)
petal width:     0.1   2.5   1.20   0.76    0.9565 (high!)
Class Distribution:
   33.3% for each of 3 classes
```

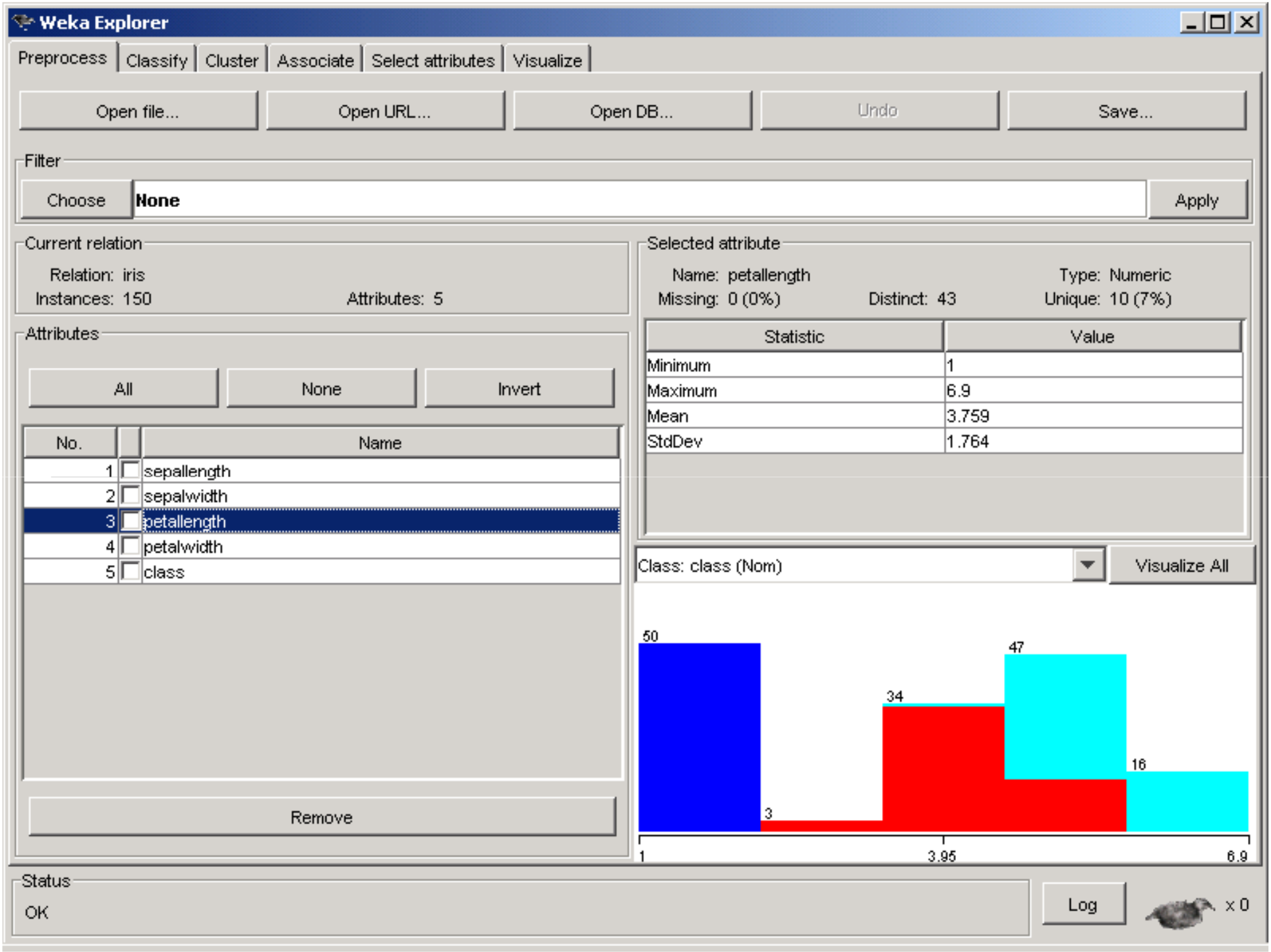
- **Open files:** Preprocess / Open file



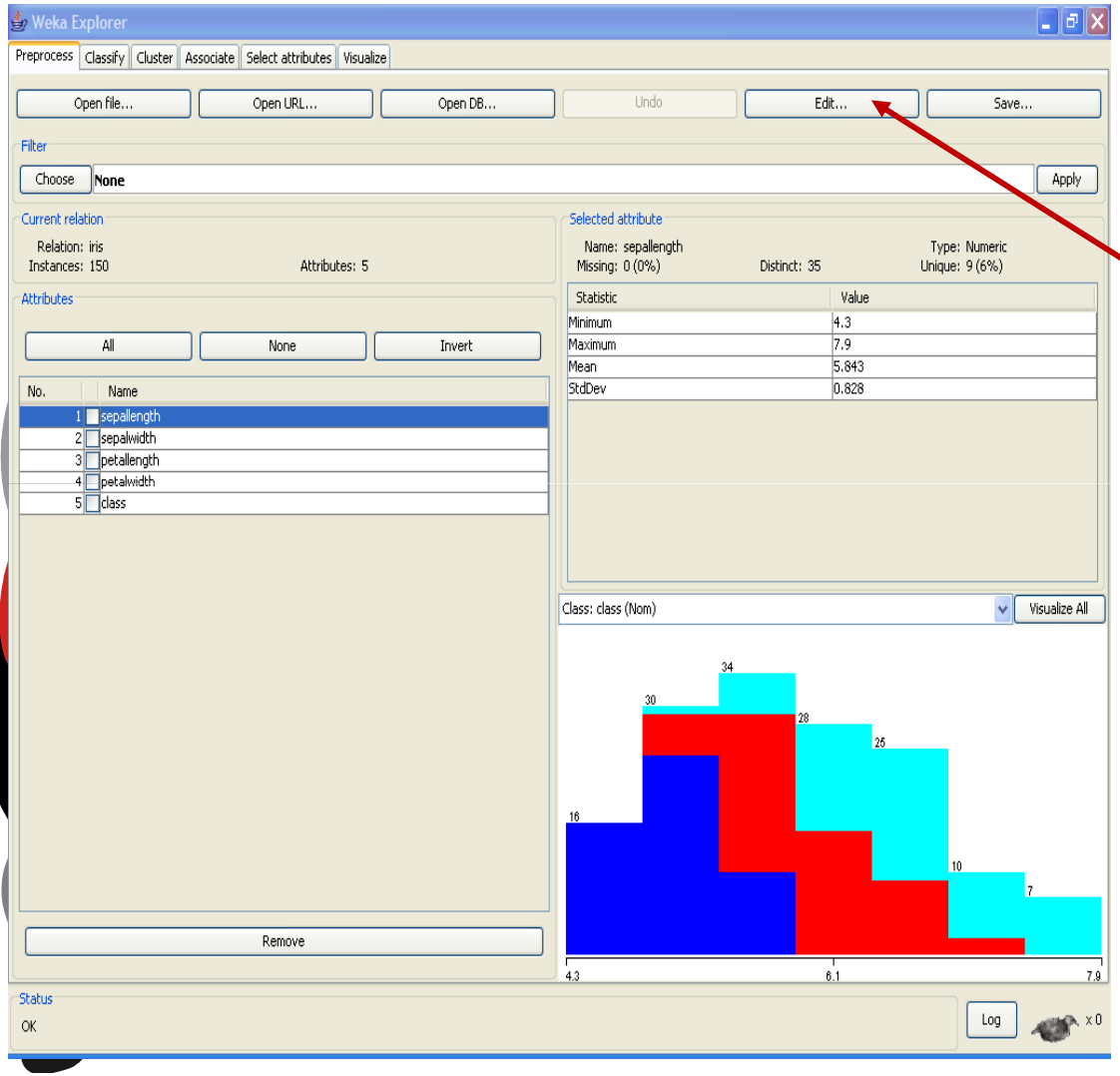

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Open file...
Open URL...
Open DB...
Undo
Save...
Filter
Choose
Apply
Current relation
Relation: None
Instances: None
Attributes
All
Abrir
Buscar en:
Weka-3-4
data
doc
Reciente
Escritorio
Mis docume...
Mi PC
Mis sitios de ...
Nombre de archivo:
Abrir
Archivos de tipo:
Arff data files
Cancelar
Visualize All
Remove
Status
Welcome to the Weka Explorer
Log
x 0

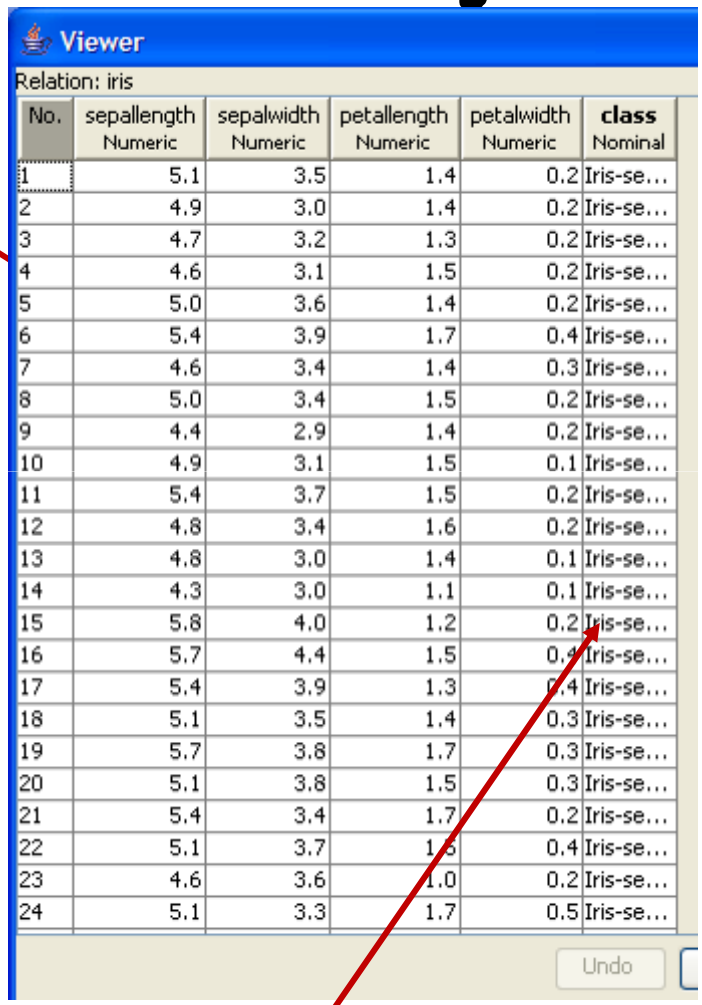

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Open file...
Open URL...
Open DB...
Undo
Save...
Filter
Choose
Apply
Current relation
Relation: None
Instances: None
Attributes
All
Remove
Visualize All
Abrir
Buscar en:
data
Reciente
Escritorio
Mis docume...
Mi PC
Mis sitios de ...
contact-lenses.arff
cpu.arff
iris.arff
labor.arff
soybean.arff
weather.arff
weather.nominal.arff
Nombre de archivo:
Abrir
Archivos de tipo:
Arff data files
Cancelar
Status
Welcome to the Weka Explorer
Log
x 0


Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Open file...
Open URL...
Open DB...
Undo
Save...
Filter
Choose
None
Apply
Current relation
Relation: iris
Instances: 150
Attributes: 5
Selected attribute
Name: sepallength
Type: Numeric
Missing: 0 (0%)
Distinct: 35
Unique: 9 (6%)
Statistic
Value
Minimum
4.3
Maximum
7.9
Mean
5.843
StdDev
0.828
Attributes
All
None
Invert
No.
Name
1 sepallength
2 sepalwidth
3 petallength
4 petalwidth
5 class
Remove
Class: class (Nom)
Visualize All
34
30
28
25
16
10
7
16
[4.3, 4.814]
4.3
6.1
7.9
Status
OK
Log
x 0

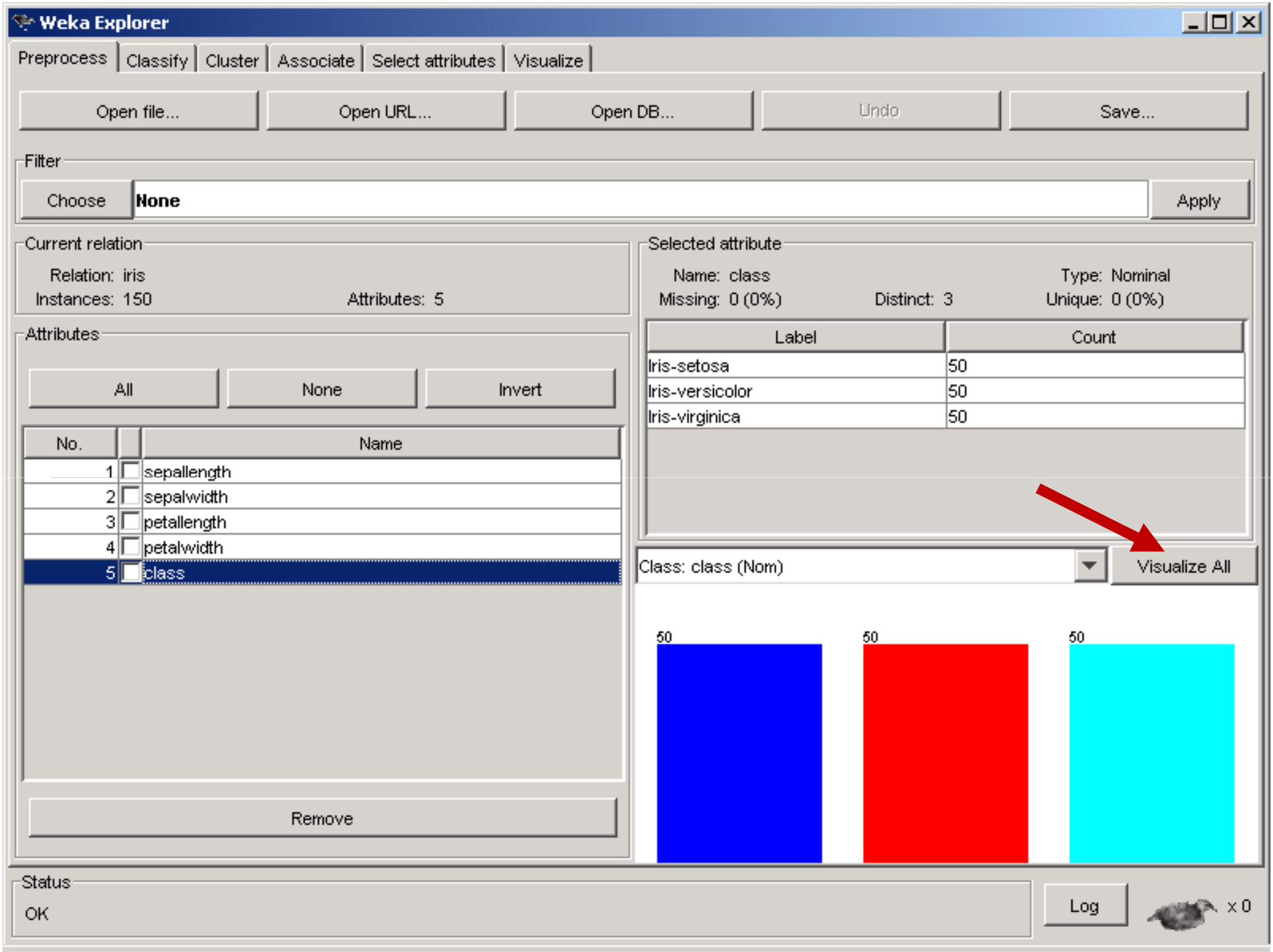

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Open file...
Open URL...
Open DB...
Undo
Save...
Filter
Choose
None
Apply
Current relation
Relation: iris
Instances: 150
Attributes: 5
Attributes
All
None
Invert
No.
Name
1 sepallength
2 sepalwidth
3 petallength
4 petalwidth
5 class
Remove
Selected attribute
Name: petallength
Type: Numeric
Missing: 0 (0%)
Distinct: 43
Unique: 10 (7%)
Statistic
Value
Minimum 1
Maximum 6.9
Mean 3.759
StdDev 1.764
Class: class (Nom)
Visualize All
50
3
34
47
16
1
3.95
6.9
Status
OK
Log
x 0

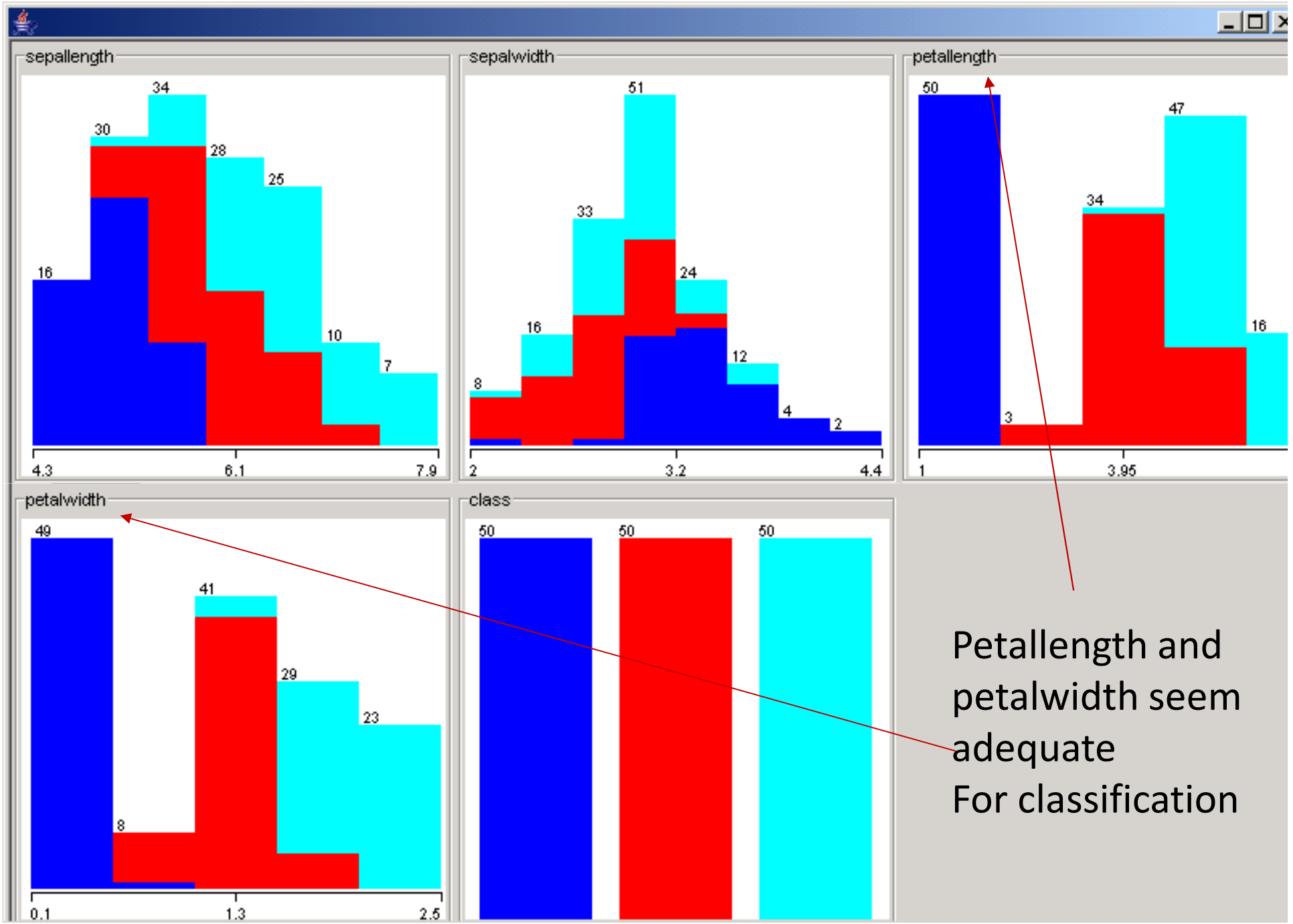
Edit files



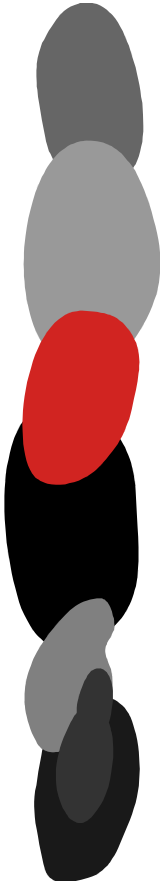


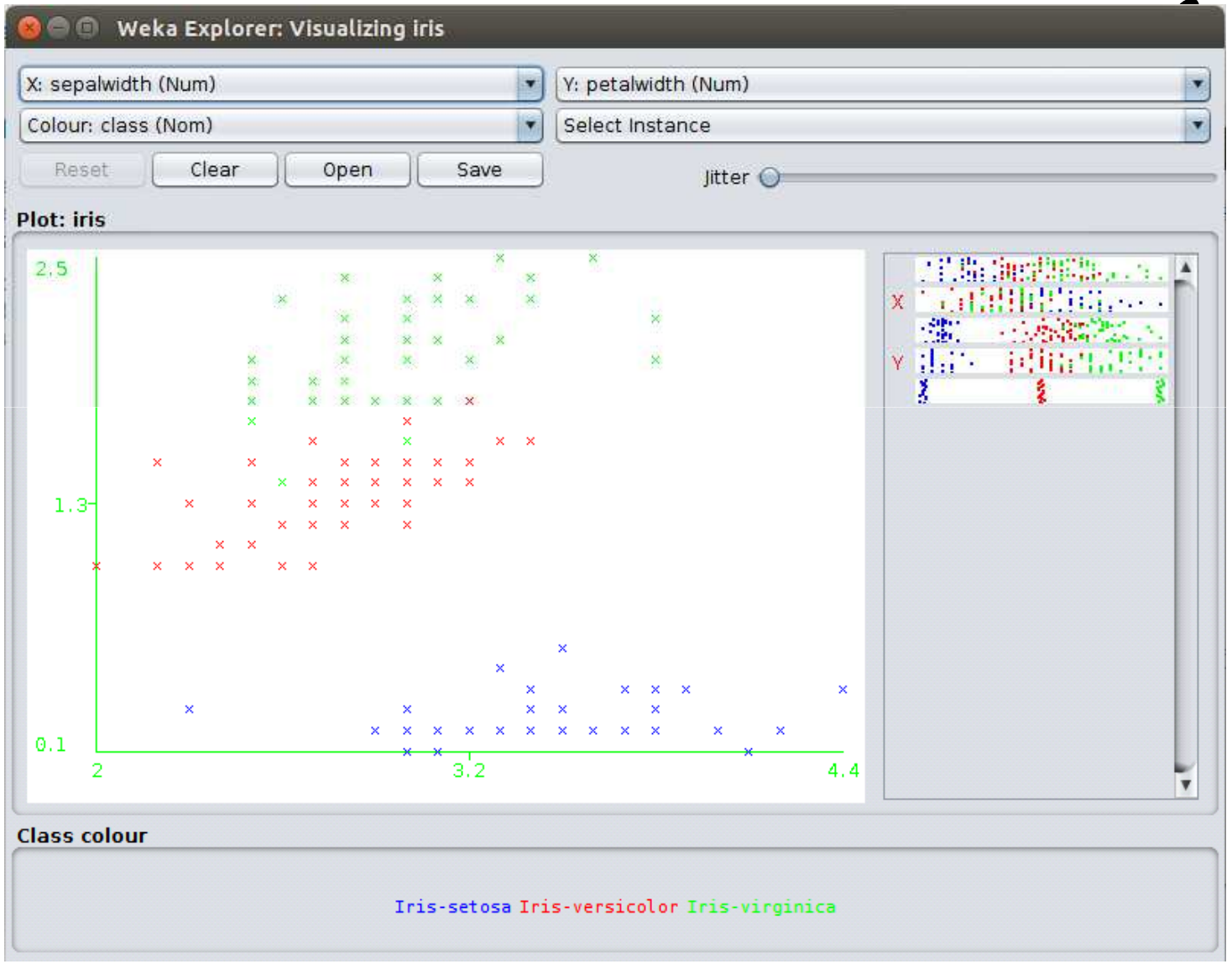
 Change files


Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Open file...
Open URL...
Open DB...
Undo
Save...
Filter
Choose
None
Apply
Current relation
Relation: iris
Instances: 150
Attributes: 5
Selected attribute
Name: class
Type: Nominal
Missing: 0 (0%)
Distinct: 3
Unique: 0 (0%)
Label
Count
Iris-setosa
50
Iris-versicolor
50
Iris-virginica
50
Attributes
All
None
Invert
No.
Name
1 sepallength
2 sepalwidth
3 petallength
4 petalwidth
5 class
Class: class (Nom)
Visualize All
50
50
50
Remove
Status
OK
Log
x 0


sepallength
34
30
28
25
16
10
7
4.3
6.1
7.9
sepalwidth
51
33
24
16
12
8
4
2
2
3.2
4.4
petallength
50
47
34
16
3
1
3.95
petalwidth
49
41
29
23
8
0.1
1.3
2.5
class
50
50
50
Petallength and
petalwidth seem
adequate
For classification

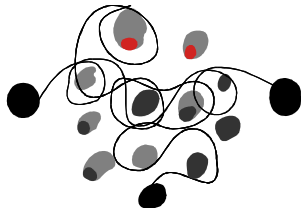
Visualize

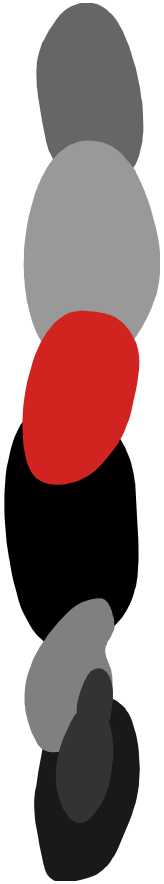



Weka Explorer: Visualizing iris
X: sepalwidth (Num)
Y: petalwidth (Num)
Colour: class (Nom)
Select Instance
Reset
Clear
Open
Save
Jitter
Plot: iris
2.5
1.3
0.1
2
3.2
4.4
X
Y
Class colour
Iris-setosa Iris-versicolor Iris-virginica

soybean.arff

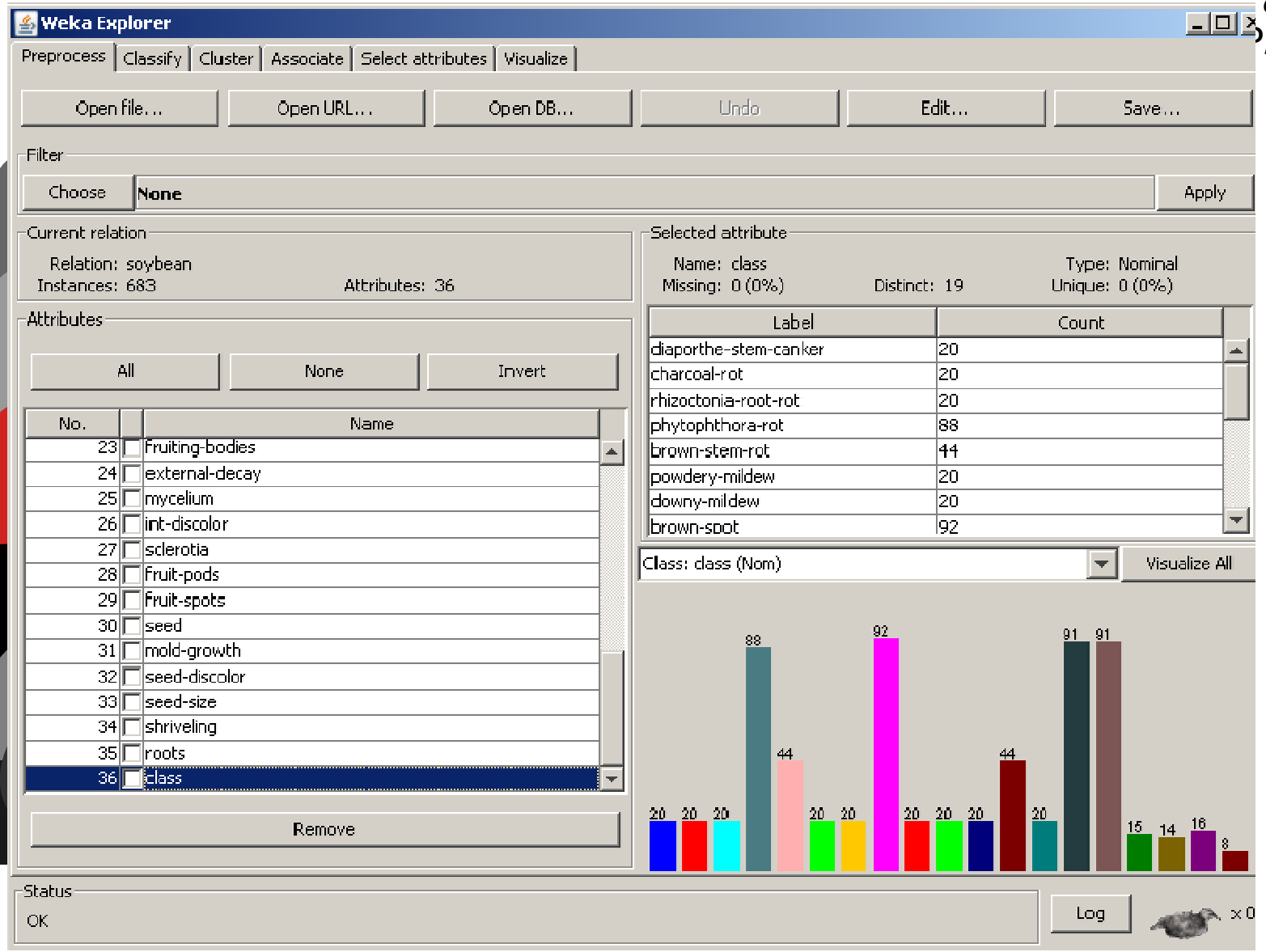
35 features 683 instances 19 classes

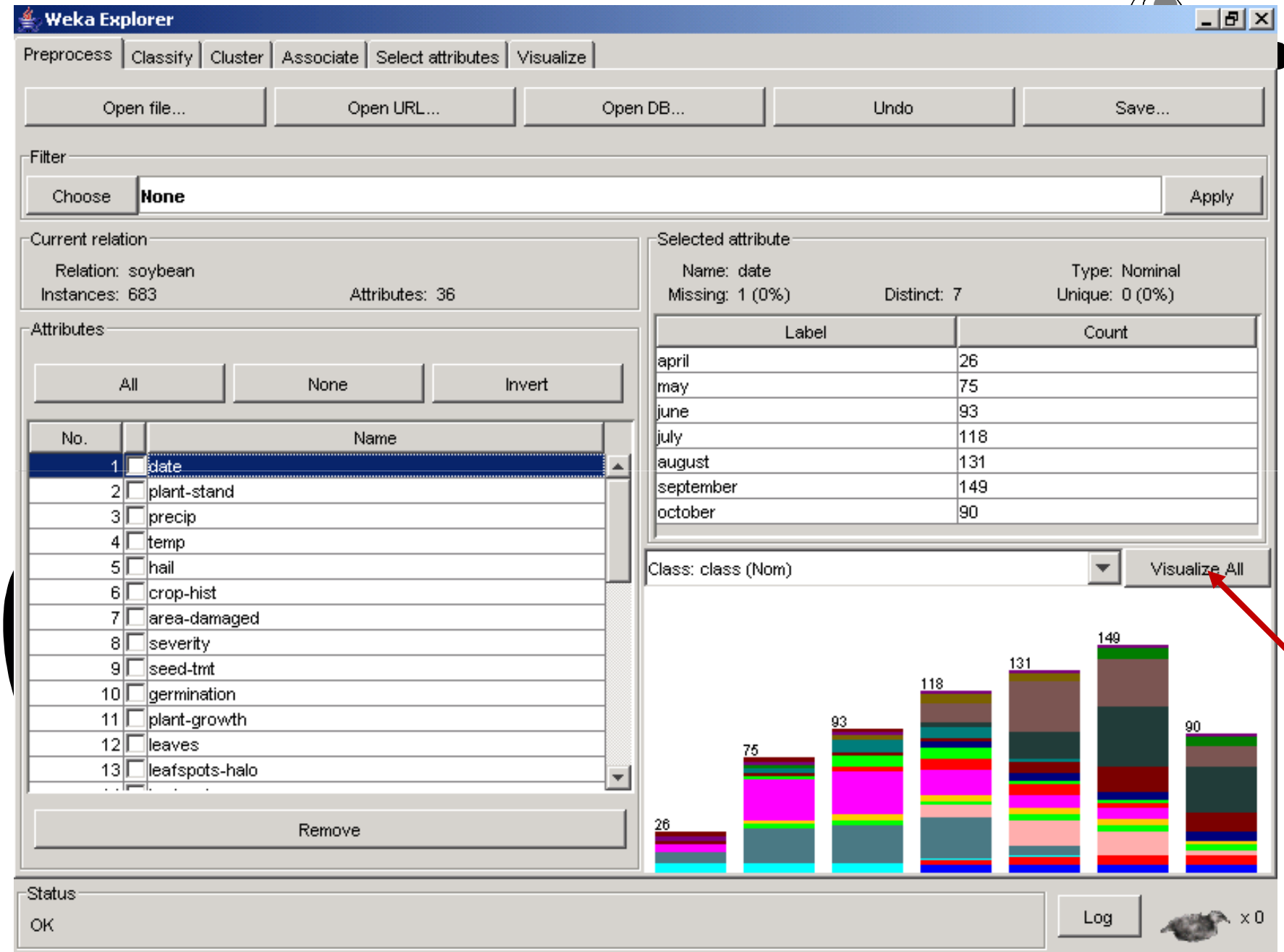


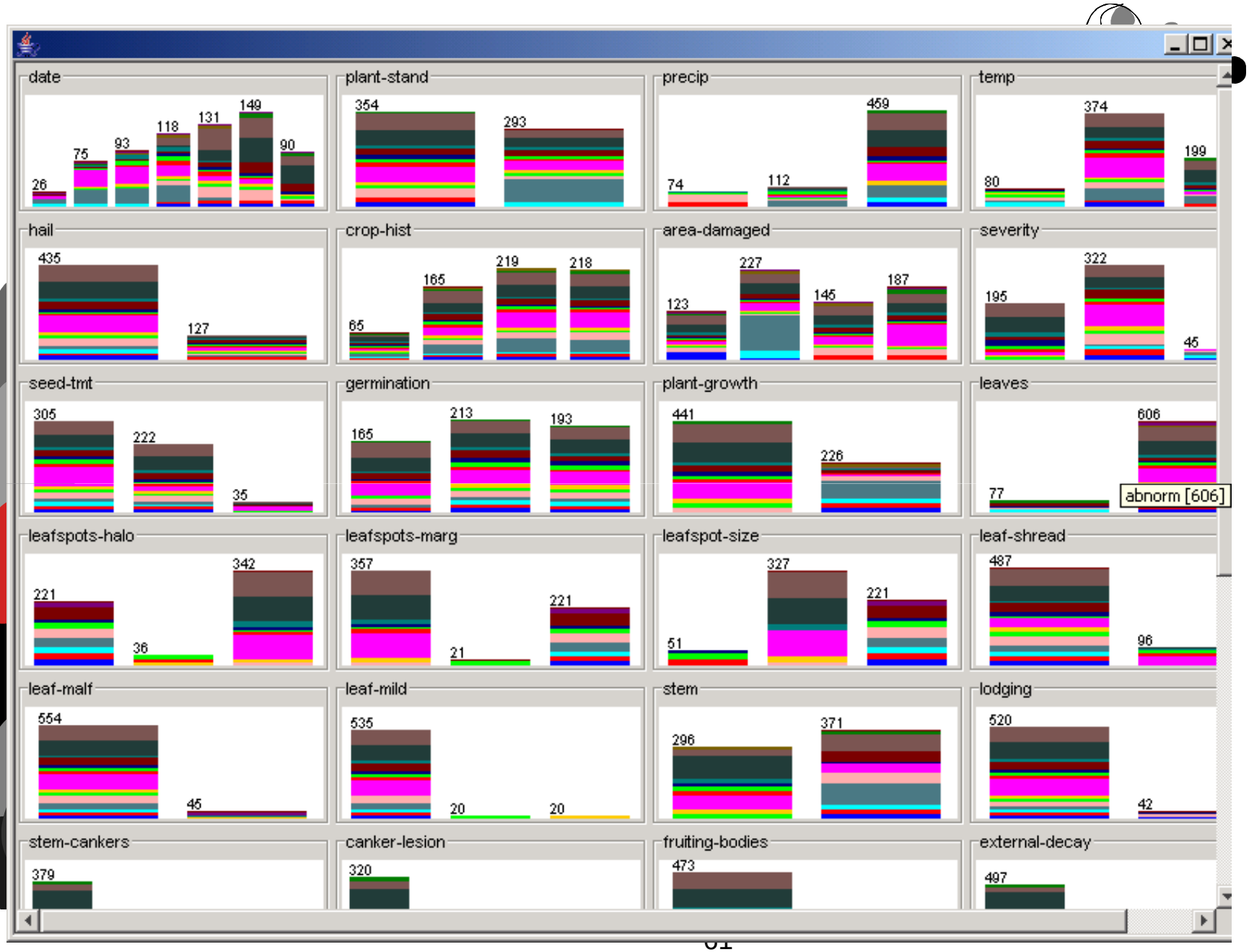


```
@RELATION soybean
@ATTRIBUTE date              {april,may,june,july,august,september,october}
@ATTRIBUTE plant-stand       {normal,lt-normal}
@ATTRIBUTE precip            {lt-norm,norm,gt-norm}
@ATTRIBUTE temp              {lt-norm,norm,gt-norm}
…
@ATTRIBUTE class {diaporthe-stem-canker, charcoal-rot, rhizoctonia-root-rot,
    phytophthora-rot, brown-stem-rot, powdery-mildew, downy-mildew, brown-
    spot, bacterial-blight, bacterial-pustule, purple-seed-stain, anthracnose,
    phyllosticta-leaf-spot, alternarialeaf-spot, frog-eye-leaf-spot,
    diaporthe-pod-&-stem-blight, cyst-nematode, 2-4-d-injury, herbicide-
    injury}

@DATA
october, normal, gt-norm, norm, yes, same-lst-yr, low-areas, pot-severe, none,
    90-100, abnorm, abnorm, absent, dna, dna, absent, absent, absent, abnorm,
    no, above-sec-nde, brown, present, firm-and-dry, absent, none, absent,
    norm, dna, norm, absent, absent, norm, absent, norm, diaporthe-stem-canker
august, normal, gt-norm, norm, yes, same-lst-two-yrs, scattered, severe,
    fungicide, 80-89, abnorm, abnorm, absent, dna, dna, absent, absent,
    absent, abnorm, yes, above-sec-nde, brown, present, firm-and-dry, absent,
    none, absent, norm, dna, norm, ....
```

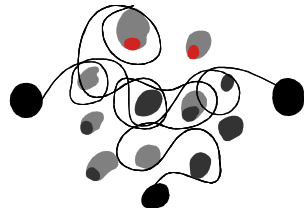
soybean.arff




Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Open file...
Open URL...
Open DB...
Undo
Save...
Filter
Choose
None
Apply
Current relation
Relation: soybean
Instances: 683
Attributes: 36
Attributes
All
None
Invert
No.
Name
1 date
2 plant-stand
3 precip
4 temp
5 hail
6 crop-hist
7 area-damaged
8 severity
9 seed-tmt
10 germination
11 plant-growth
12 leaves
13 leafspots-halo
Remove
Selected attribute
Name: date
Type: Nominal
Missing: 1 (0%)
Distinct: 7
Unique: 0 (0%)
Label
Count
april 26
may 75
june 93
july 118
august 131
september 149
october 90
Class: class (Nom)
Visualize All
26
75
93
118
131
149
90
Status
OK
Log
x 0


date
26
75
93
118
131
149
90
plant-stand
354
293
precip
74
112
459
temp
80
374
199
hail
435
127
crop-hist
65
165
219
218
area-damaged
123
227
145
187
severity
195
322
45
seed-tmt
305
222
35
germination
165
213
193
plant-growth
441
226
leaves
77
606
abnorm [606]
leafspots-halo
221
36
342
leafspots-marg
357
21
221
leafspot-size
51
327
221
leaf-shread
487
96
leaf-malf
554
45
leaf-mild
535
20
20
stem
296
371
lodging
520
42
stem-cankers
379
canker-lesion
320
fruiting-bodies
473
external-decay
497

Visualize. Texts

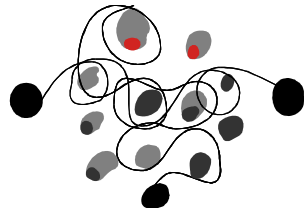


Reuters_Corn

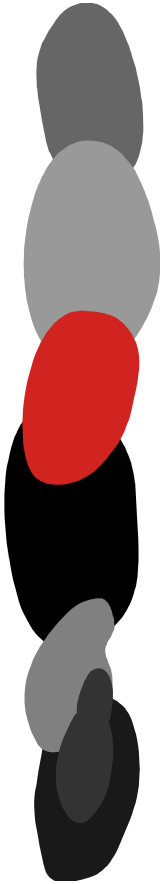
```
@attribute Text string
@attribute class-att {0,1}

@data
'ASIAN EXPORTERS … dispute.\n REUTER\n&#3;',0
'CHINA DAILY … details.\n REUTER\n&#3;',0
'THAI TRADE …  35 pct.\n REUTER\n&#3;',1
```

Assignment: try to see reuters_corn_test database with weka
- Visualize and edit

Texts



`Reuters_Corn`: special characters\n, <, -, numbers



Viewer

Relation: Reuters-21578 Corn ModApte Test-weka.filters.unsupervised.attribute.NumericToBinary-weka.filters.unsupervised.inst...

No.	1: Text String	2: class-at Nominal
1	ASIAN EXPORTERS FEAR DAMAGE FROM U.S.-JAPAN RIFT Mounting trade friction between the\nU.S. And Japan	0
2	CHINA DAILY SAYS VERMIN EAT 7-12 PCT GRAIN STOCKS A survey of 19 provinces and seven cities\nshowed	0
3	JAPAN TO REVISE LONG-TERM ENERGY DEMAND DOWNWARDS The Ministry of International Trade	0
4	THAI TRADE DEFICIT WIDENS IN FIRST QUARTER Thailand's trade deficit widened to 4.5\nbillion baht in the	0
5	INDONESIA SEES CPO PRICE RISING SHARPLY Indonesia expects crude palm oil (CPO)\nprices to rise sharply	0
6	AUSTRALIAN FOREIGN SHIP BAN ENDS BUT NSW PORTS HIT Tug crews in New South Wales (NSW),\nVictoria	0
7	INDONESIAN COMMODITY EXCHANGE MAY EXPAND The Indonesian Commodity Exchange is\nlikely to start	0
8	SRI LANKA GETS USDA APPROVAL FOR WHEAT PRICE Food Department officials said the U.S.\nDepartment of	0
9	WESTERN MINING TO OPEN NEW GOLD MINE IN AUSTRALIA Western Mining Corp Holdings Ltd\n<WMNG.S)	0
10	SUMITOMO BANK AIMS AT QUICK RECOVERY FROM MERGER Sumitomo Bank Ltd <SUMI.T) is certain to\nlose	0
11	SUBROTO SAYS INDONESIA SUPPORTS TIN PACT EXTENSION Mines and Energy Minister Subroto\nconfirmed	0
12	BUNDESBANK ALLOCATES 6.1 BILLION MARKS IN TENDER The Bundesbank accepted bids for 6.1\nbillion	0
13	BOND CORP STILL CONSIDERING ATLAS MINING BAIL-OUT Bond Corp Holdings Ltd <BONA.S) and	0
14	CHINA INDUSTRIAL OUTPUT RISES IN FIRST QUARTER China's industrial output rose 14.1 pct\nin the first	0
15	JAPAN MINISTRY SAYS OPEN FARM TRADE WOULD HIT U.S. Japan's Agriculture Ministry, angered by\nU.S.	0
16	AMATIL PROPOSES TWO-FOR-FIVE BONUS SHARE ISSUE Amatil Ltd <AMAA.S) said it proposes to\nmake a	0
17	BOWATER 1986 PRETAX PROFITS RISE 15.6 MLN STG Shr 27.7p vs 20.7p\n Div 6.0p vs 5.5p making 10.0p vs	0
18	U.K. MONEY MARKET DEFICIT FORECAST AT 250 MLN STG The Bank of England said it forecast a\nshortage of	0
19	SOUTH KOREA MOVES TO SLOW GROWTH OF TRADE SURPLUS South Korea's trade surplus is growing	0
20	FINNS AND CANADIANS TO STUDY MTBE PRODUCTION PLANT Finland's national oil company Neste	0
21	CRA SOLD FORREST GOLD FOR 76 MLN DLRS - WHIM CREEK <Whim Creek Consolidated NL) said	0
22	GERMAN INDUSTRIAL EMPLOYMENT SEEN STAGNATING The number of workers employed in\nthe West	0
23	BOWATER INDUSTRIES PROFIT EXCEED EXPECTATIONS Bowater Industries Plc <BWTR.L) 1986\npretax profits	0

Add instance | Undo | OK | Cancel

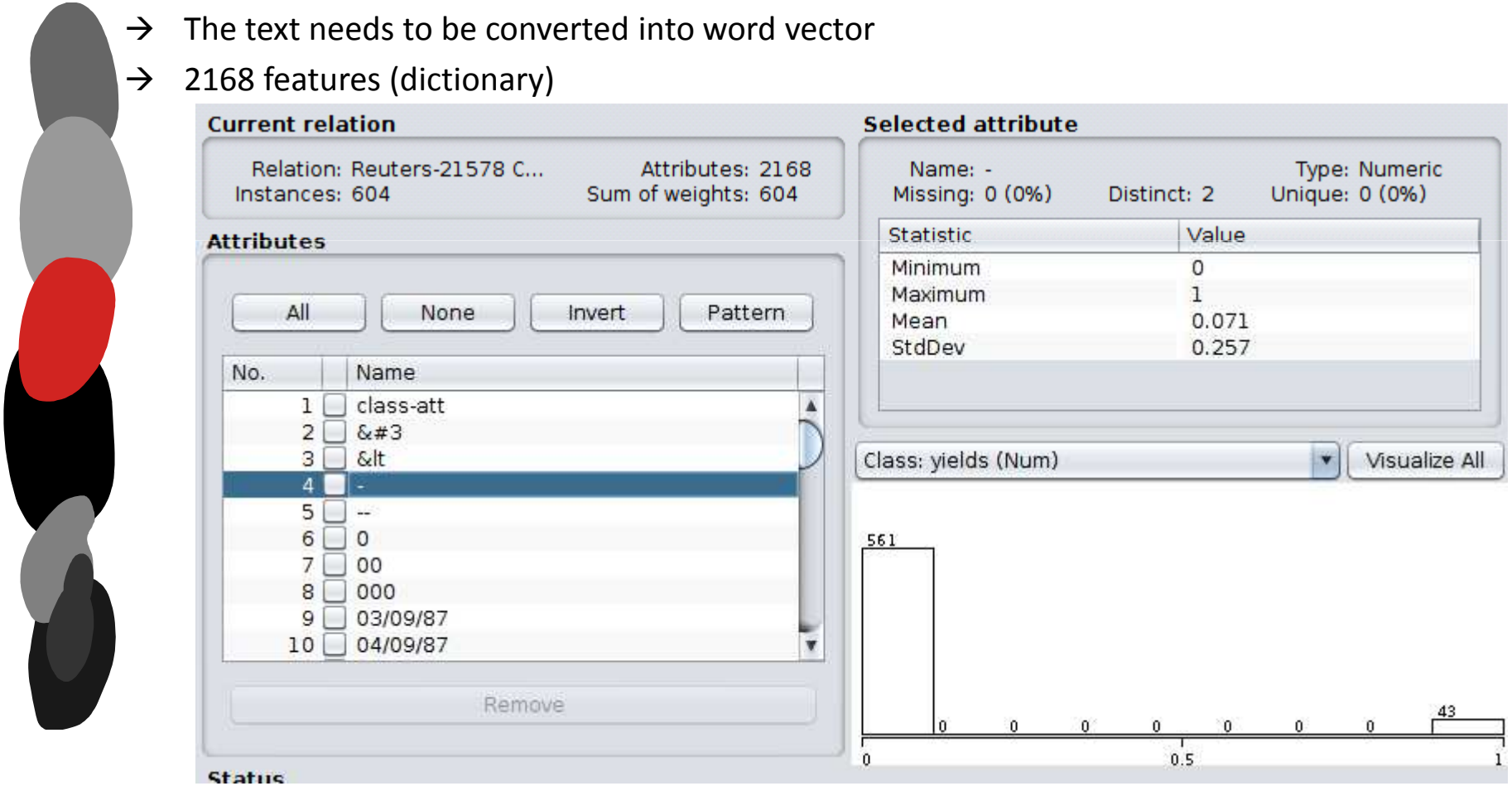
Texts

Reuters_Corn

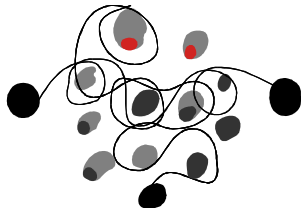
Weka can not treat the data as it is represented here: they are not numeric attributes
neither nominal

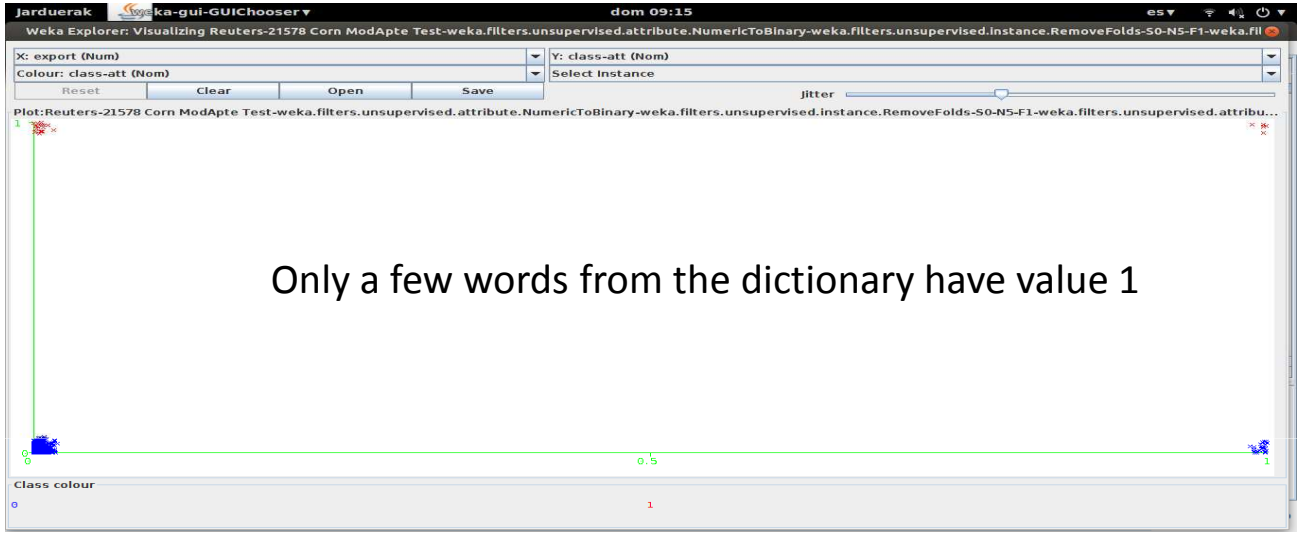
→ The text needs to be converted into word vector

→ 2168 features (dictionary)

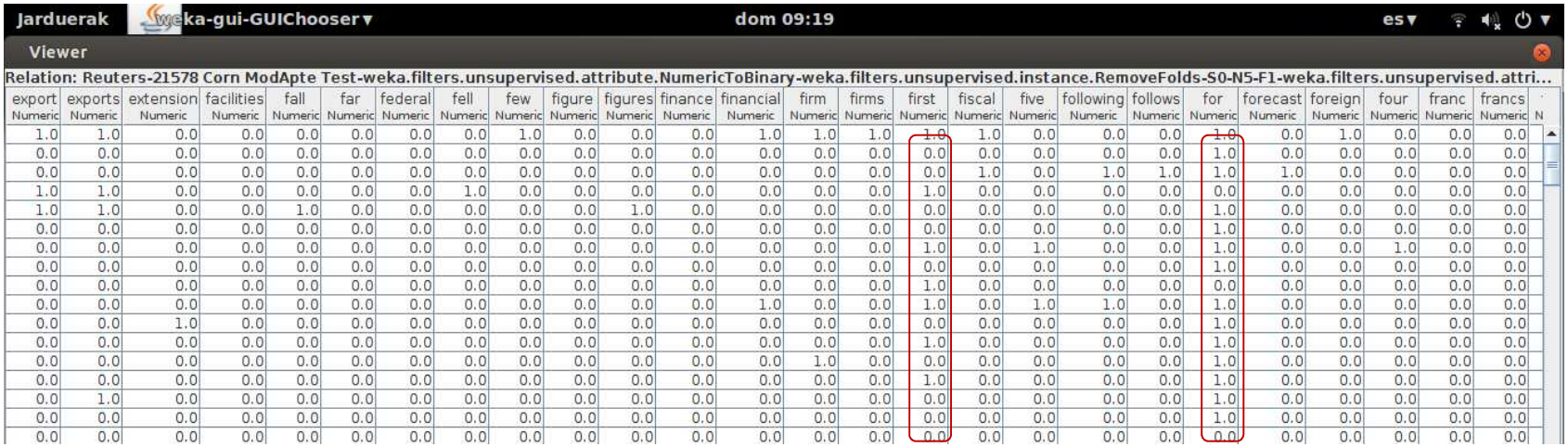


Weka. Texts





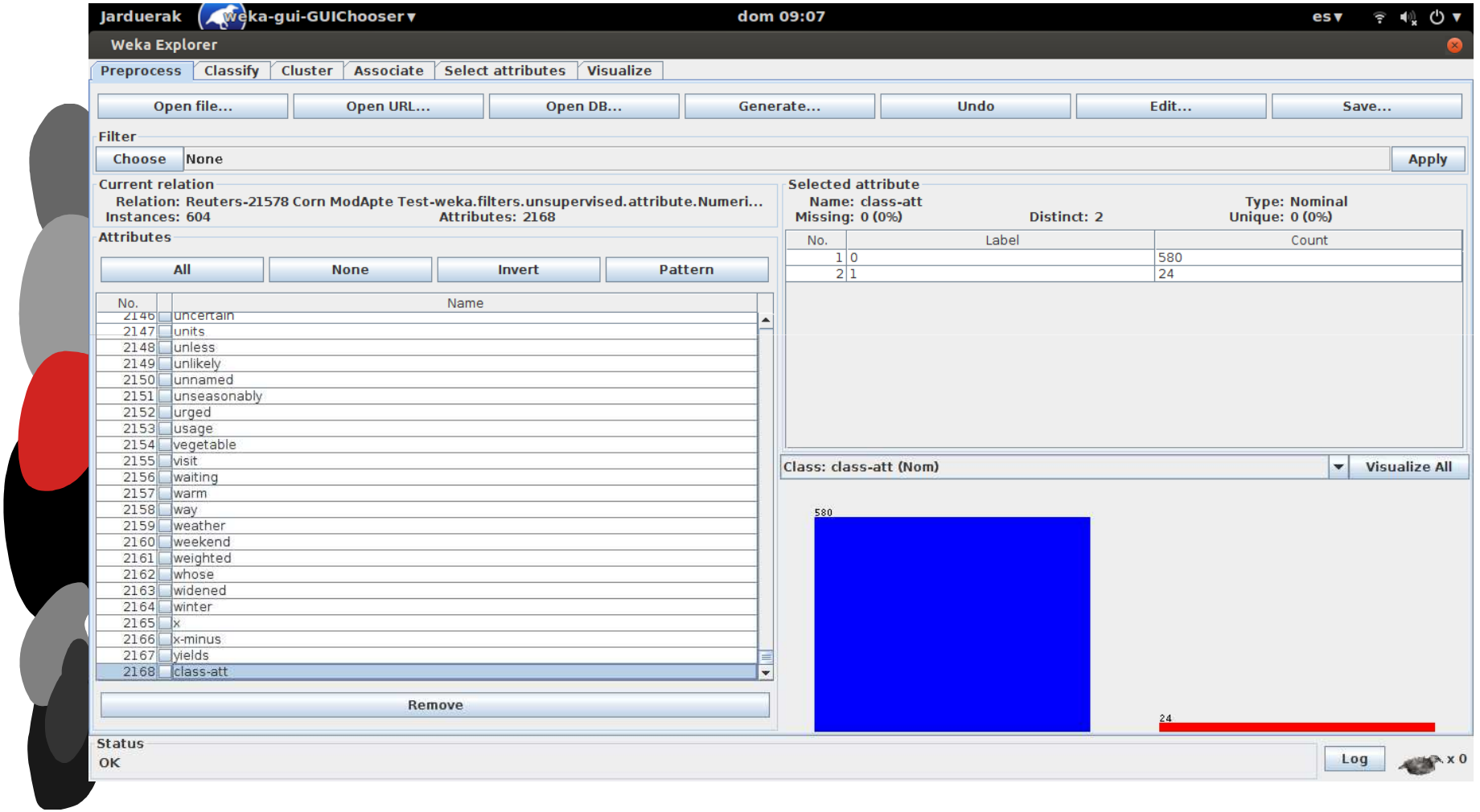
Only a few words from the dictionary have value 1



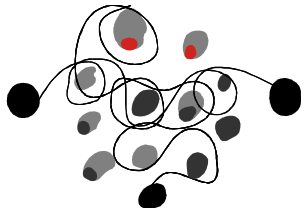
export Numeric	exports Numeric	extension Numeric	facilities Numeric	fall Numeric	far Numeric	federal Numeric	fell Numeric	few Numeric	figure Numeric	figures Numeric	finance Numeric	financial Numeric	firm Numeric	firms Numeric	first Numeric	fiscal Numeric	five Numeric	following Numeric	follows Numeric	for Numeric	forecast Numeric	foreign Numeric	four Numeric	franc Numeric	francs Numeric	N
1.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	1.0	1.0	1.0	1.0	1.0	0.0	0.0	0.0	1.0	0.0	1.0	0.0	0.0	0.0	
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	1.0	1.0	1.0	1.0	0.0	0.0	0.0	0.0	
1.0	1.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
1.0	1.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	1.0	0.0	0.0	1.0	0.0	0.0	1.0	0.0	0.0	
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	1.0	0.0	1.0	1.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	
0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	
0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	

Weka. Texts

Reuters_Corn: unbalanced

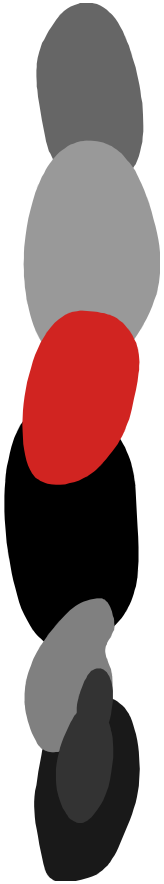


Modifications in features/instances



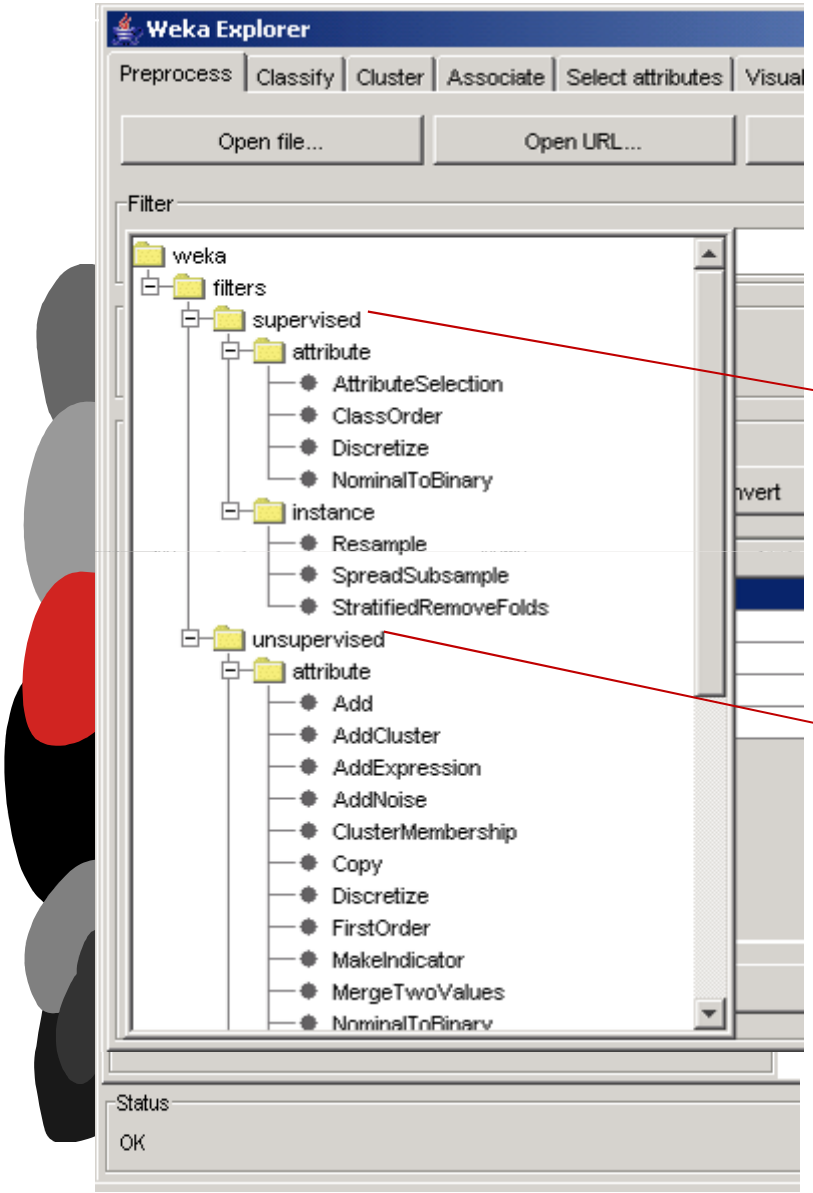
Instance or feature modifications are often required:

- To adapt to the needs of algorithms (numbers)
- Remove no meaningful features (stop word vector)
- Balance corpora (re-sampling) → delete instances
- …



- **Feature selection** (supervised/unsupervised)
- Discretize features (discretize)
- **Trasnform words in numbers**(String to word vector)
- Normalize words (String to nominal)
- Convert nominal to number(Nominal to binary)
- Normalize numeric features (normalize)
- Clean numeric features (delete very big or very small values)
- Generate/change features with matematical operations
- Principal compoonent analysis (PCA)
- …

Filters



HAP/LAP

Filters:

to make "changes" in the features or instances

Supervised

based on the data

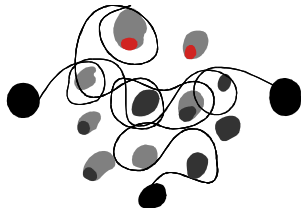
- features
- Instances

Unsupervised

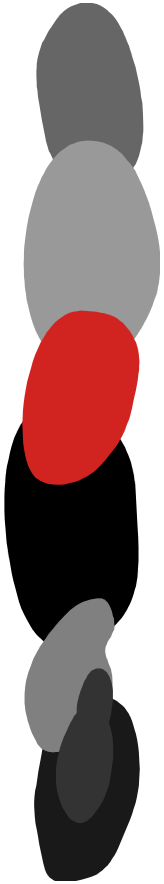
use the information given by the user

- features
- Instances

Filters (attribute selection)

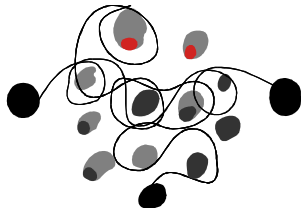


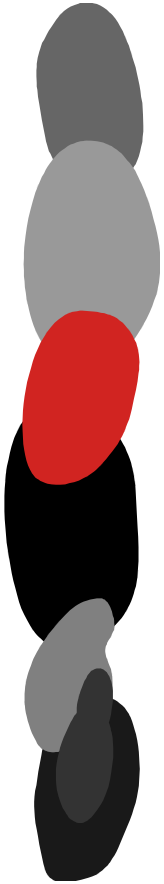
Feature selection



- **Attribute selection**:
  - **Objective**: to find the most significant feature or set of features to do the prediction
  - It often happens to have features in the data set that hardly affect to the classification
  - **Theoretically** classifiers do not take these features into account but practically they do
  - Doing the selection the efficiency of the classifier is improved. Time reduction
  - The wrong selection can reduce the classifier's performance in a % 5 - % 10
- **Options:**
  - ***Filter***: selection before the learning process starts
  - ***Wrapper***: the learning algorithm participates in the selection

Filters (attribute selection)





Two phases:

Evaluation: measure the quality or the adequacy of the feature

1.- analyse set of features (...SubsetEval)

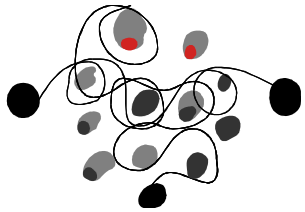
2.- analyse features one by one (...AttributeEval)

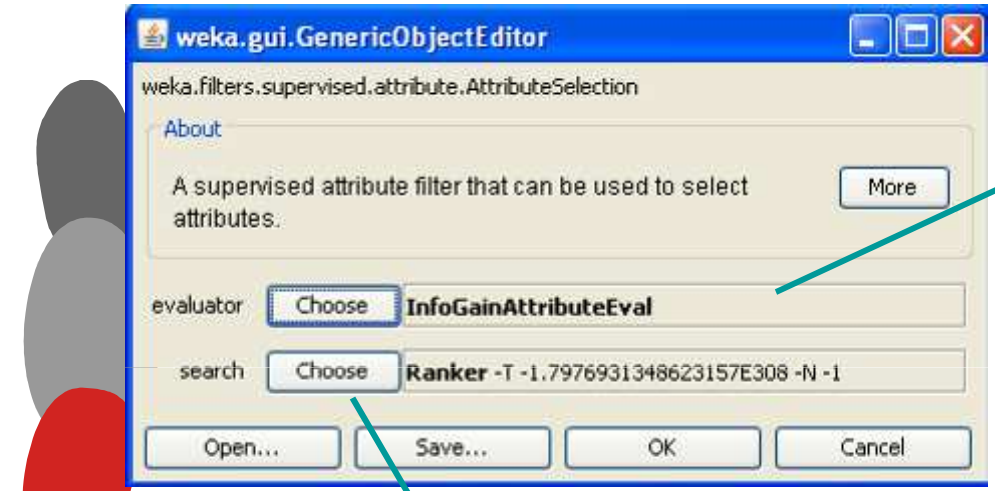
Search: to search for the most adequate set of features in the feature space

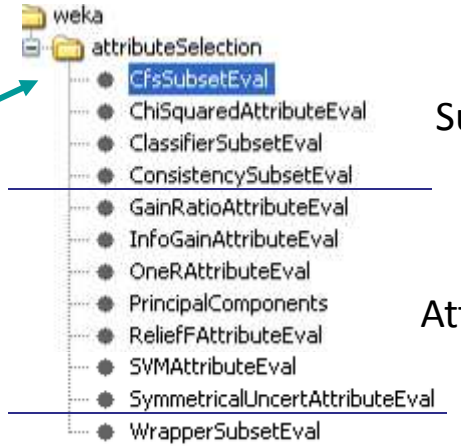
- Ranker
- Best First….

Not every evaluation and search combination is possible

Filters (attribute selection)
Evaluation + search



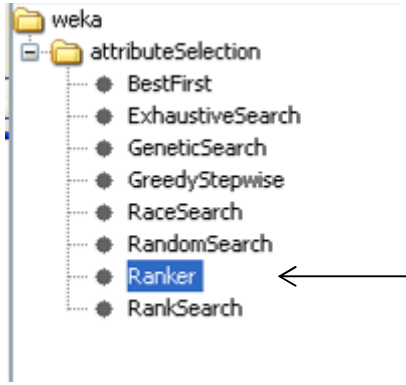




SubsetEval

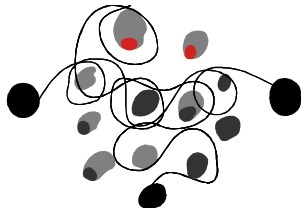
AttributeEval

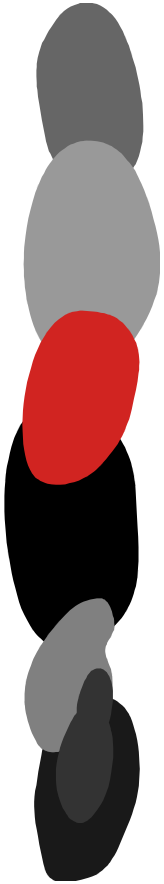
Evaluation



Search

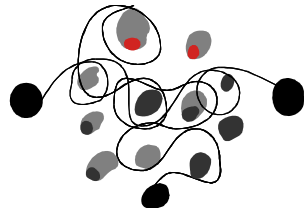
Filters (attribute selection) Search. Weka

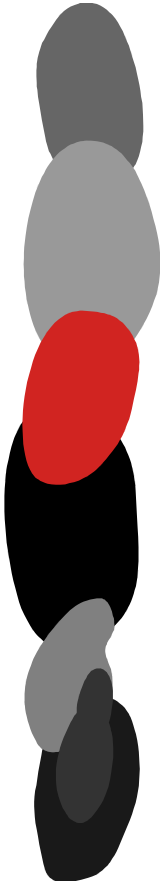




- BestFirst: If the evaluation is worse when expanding a set, the process continues with the best subset of the previous stage. (backtracking)
- ExhaustiveSearch: starts from the empty set tries every possibility (expensive)
- GreedyStepwise: includes each feature independently to the set and selects the one with the best evaluation
- RandomSearch: selects randomly the starting point.
- GeneticSearch: simple genetic algorithm (Goldberg 1989)
- Ranker: the cheapest. It just orders the features without searching for the best order.

Filters (attribute selection) Evaluation. Weka





Sets of features

CfsSubsetEval: selets features with high correlation with the class.

ClassifierSubsetEval: uses a classifier for selection.

WrapperSubsetEval: similar to the previous one with Cross-validation.

ConsistencySubsetEval: measures consistency of sets of features with class.

Individual features:

InfoGainAttributeEval:

Evaluates the quality of the features by analysing the information gain obtained for the class
InfoGain(Class, Attribute) = H (Class | Attribute) where H is entropy

GainRatioAttributeEval:

Evaluates the quality of the features by analysing the information gain ratio obtained for the class
InfoGain(Class, Attribute) = H(class)-H (Class | Attribute) where H is entropy

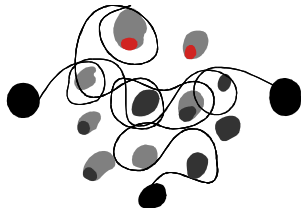
ChiSquaredAttributeEval: c2.

Symmetrical Uncertainty.

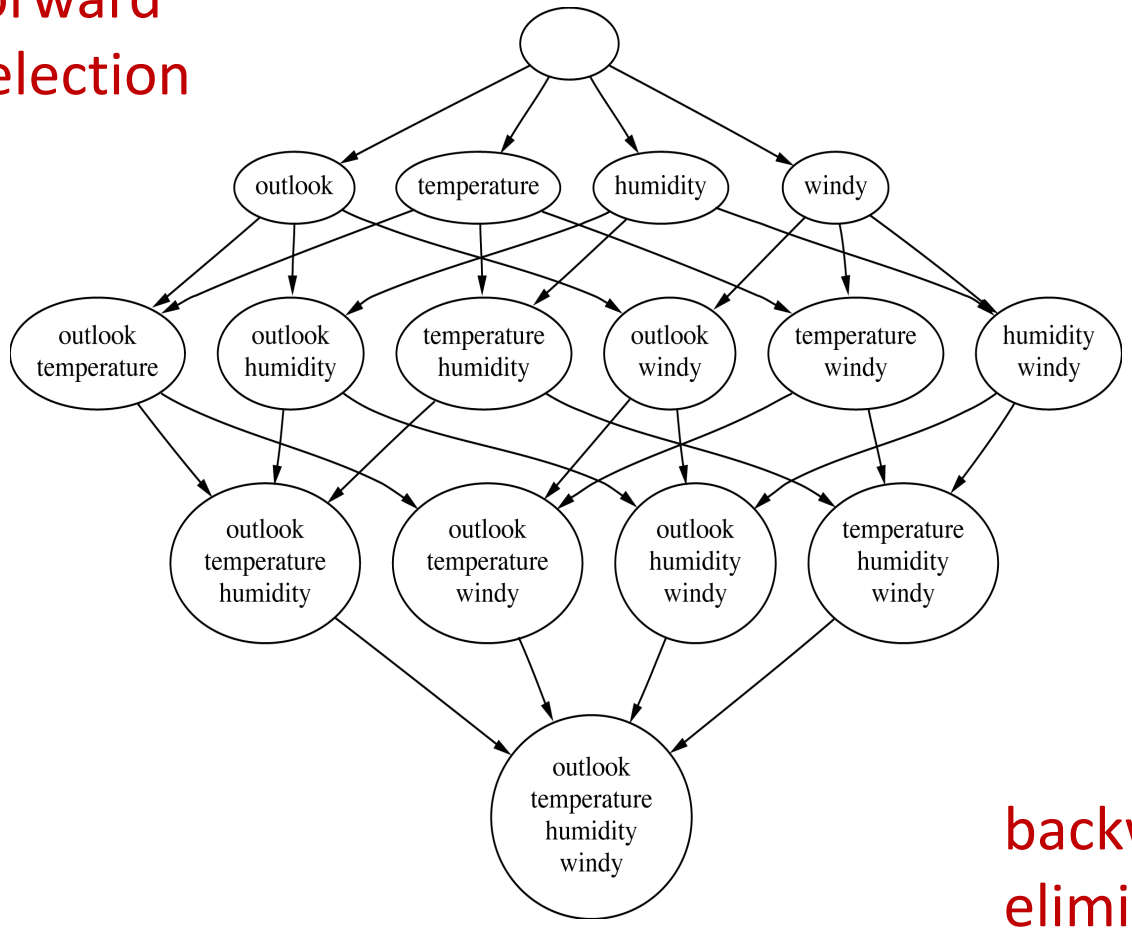
SVMAttributeEval: uses linear SVM to calculate feature values

Filters (attribute selection)

Best first



forward selection



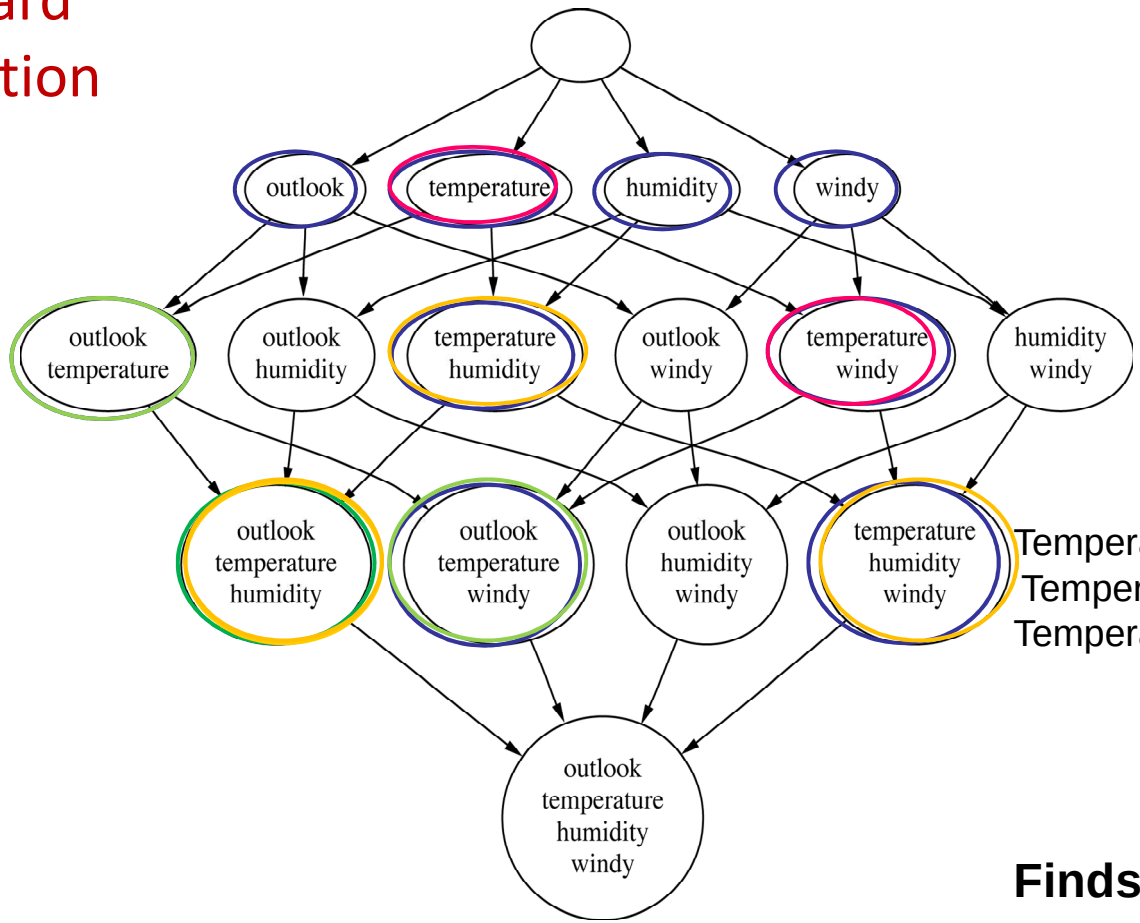
1. Quality of the subset evaluated when a new feature is inncluded/deleted.
2. Select the best one and continue with the next feature.
3. If no improvement is obtained finish with the search.

backward elimination

Filters (attribute selection)

Best first

forward selection



Temperature + windy > others

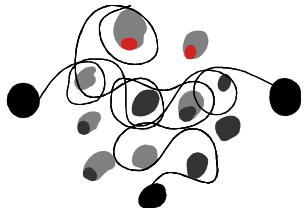
Temperature + windy > (o+t+w) and (t+h+w)
Temperature + windy > (o+t+h) and (t+h+w)
Temperature + windy > (o+t+h) and (o+t+w)

↓

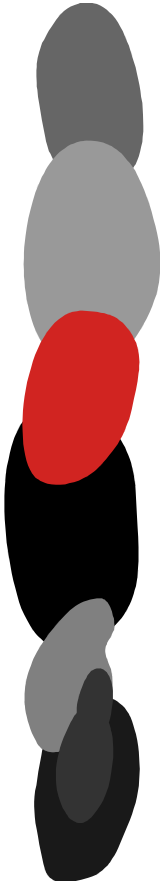
Temperature + windy

Finds the local optimum

Filters (attribute selection) Search. Weka



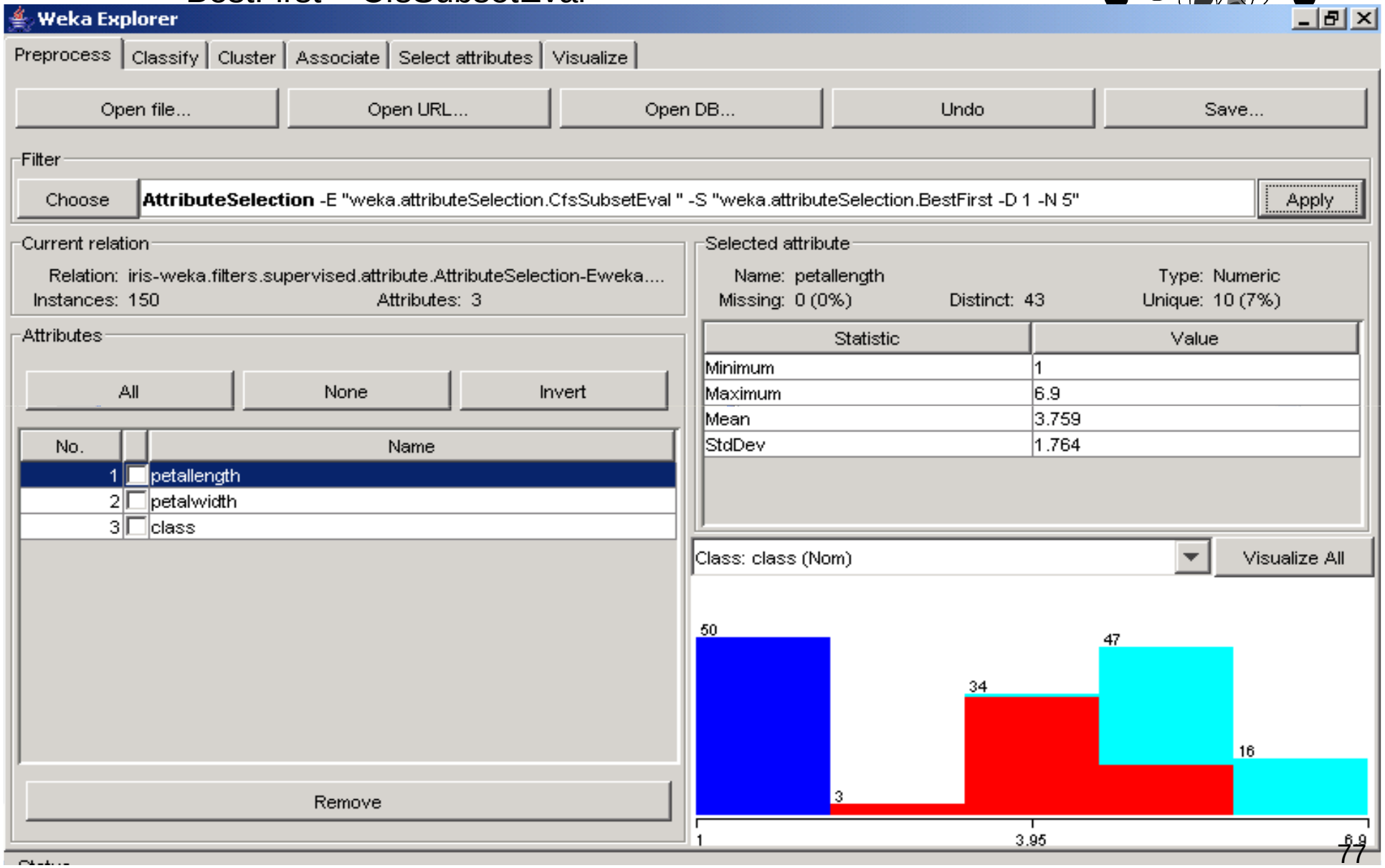
Usual parameters:



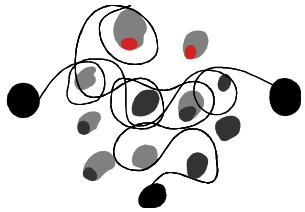
- direction: direction of the search ↑, ↓, ↕
  - forward → starts from the empty set
  - backward → starts with the complete set
  - bi-directional →start in any point and expand in the two
- numToSelect: final number of features
- searchPercent: % of the search space analysed
- startSet: fix the starting point of the search. Indicate the feature set. Ex: 1,2,5-9,17.
- searchTermination: maximum number of nodes that do not improve the evaluation by backtracking
- lookupCacheSize: number of subsets to maintain

AttributeSelection

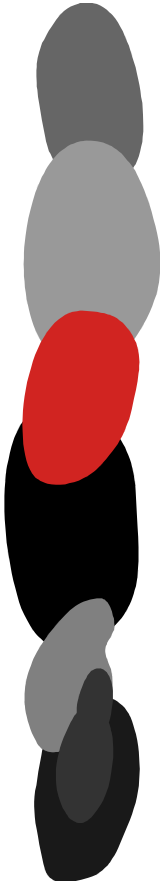
BestFirst + CfsSubsetEval

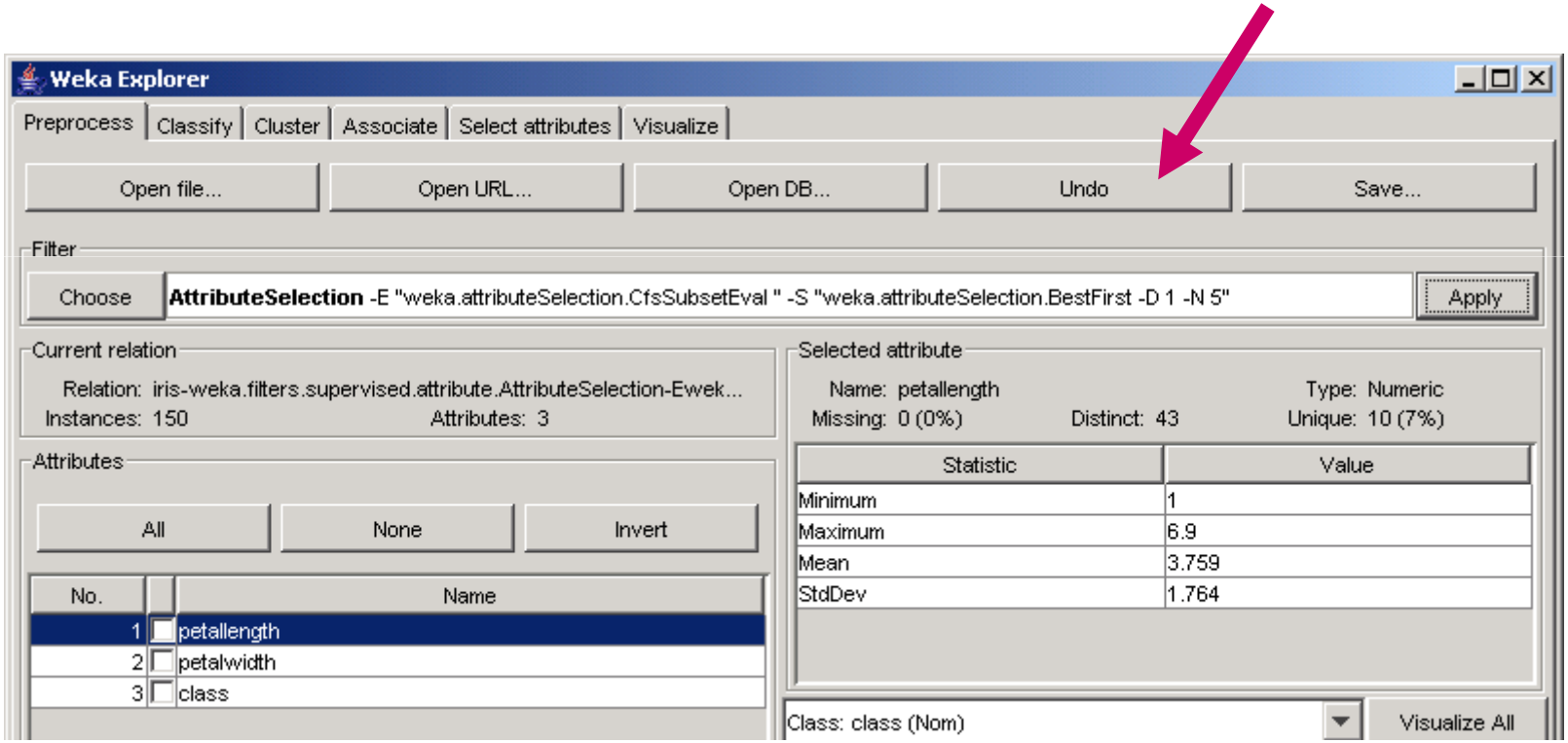


AttributeSelection

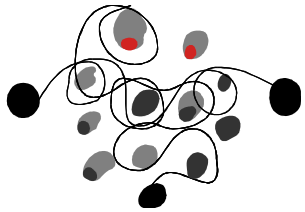


- UNDO back to the initial situation

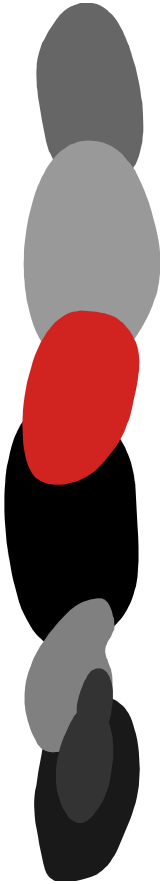


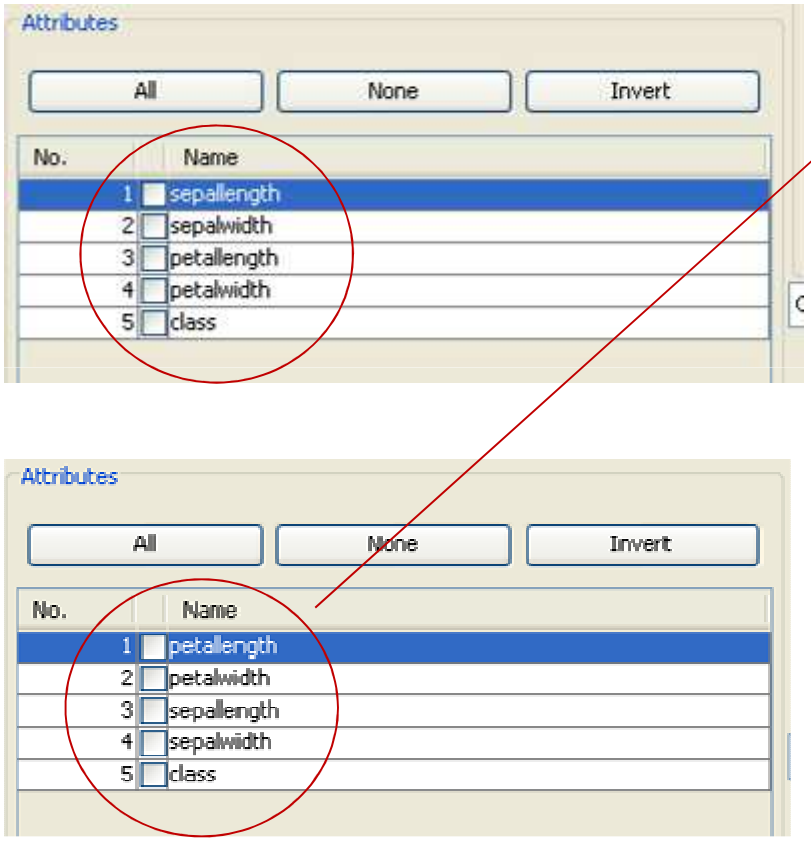


AttributeSelection



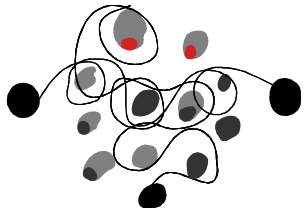
InfoGain (evaluation) + Ranker (search)



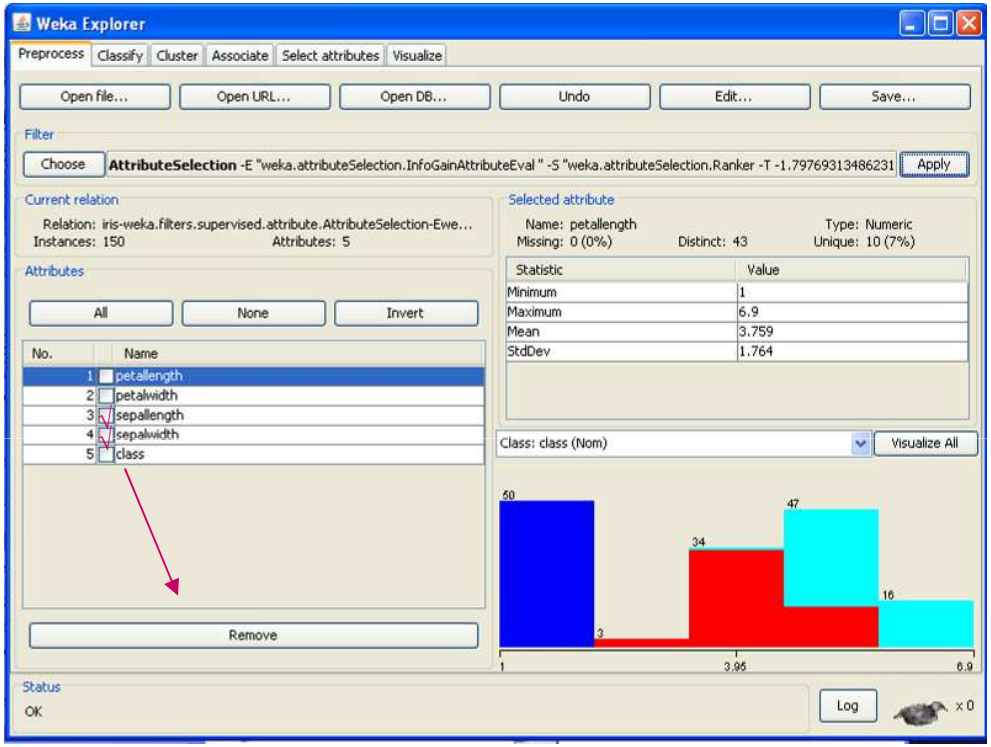


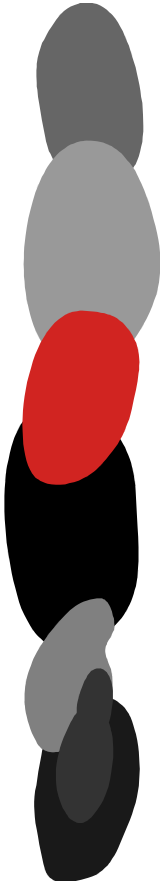
- Features ordered according to the evaluation method
- Organized in the new order
- If we want to delete them:
  - manually (remove)
  - use threshold (menu)
  - limit number (numToSelect)

Search+ Evaluation



InfoGain + Ranker

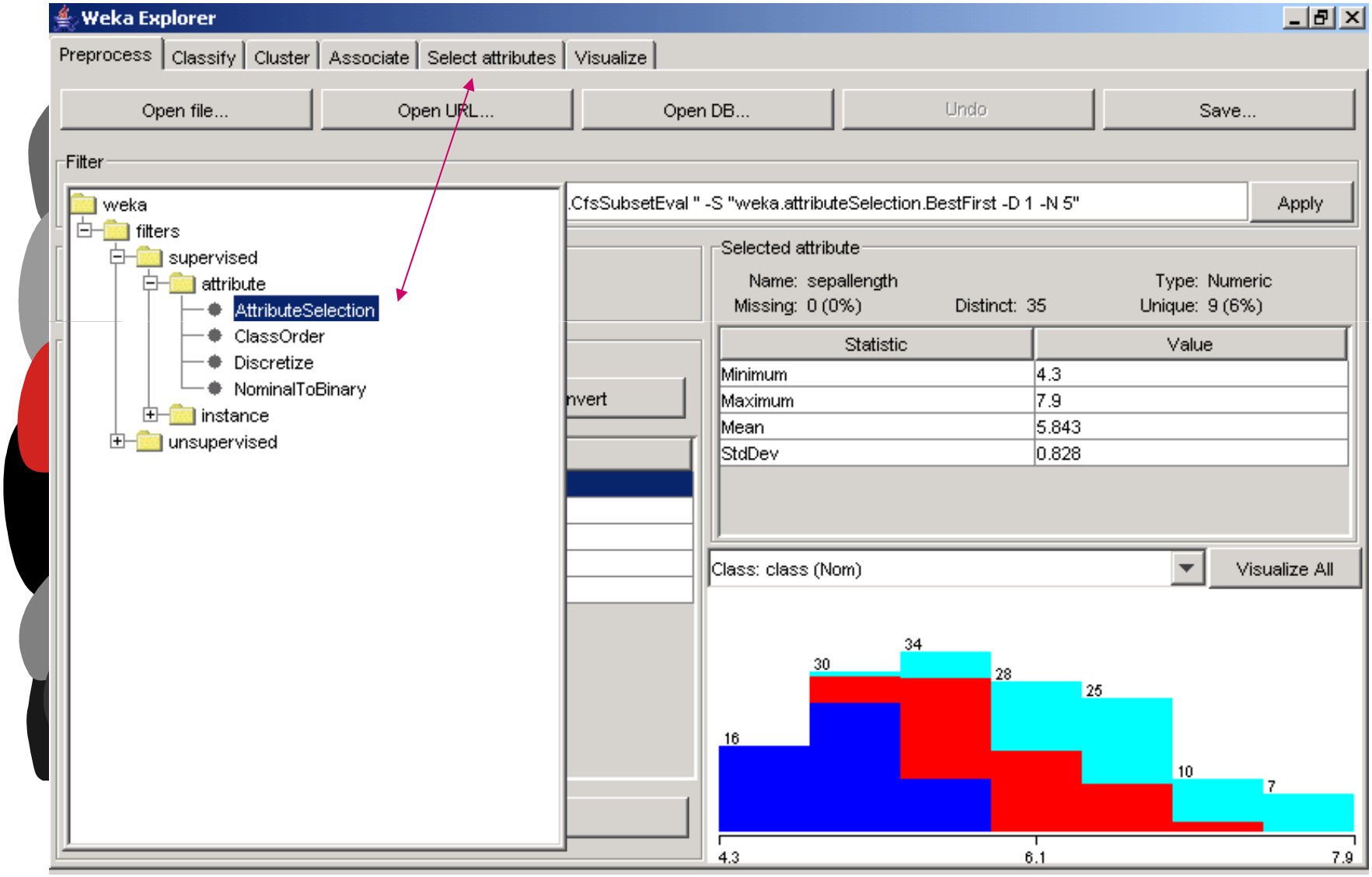




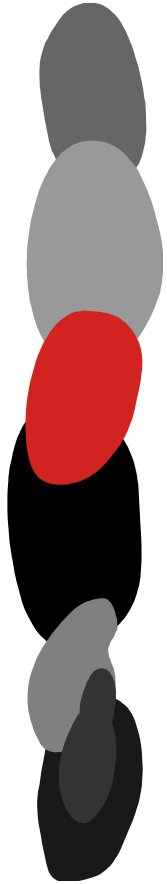
Filters: Remove

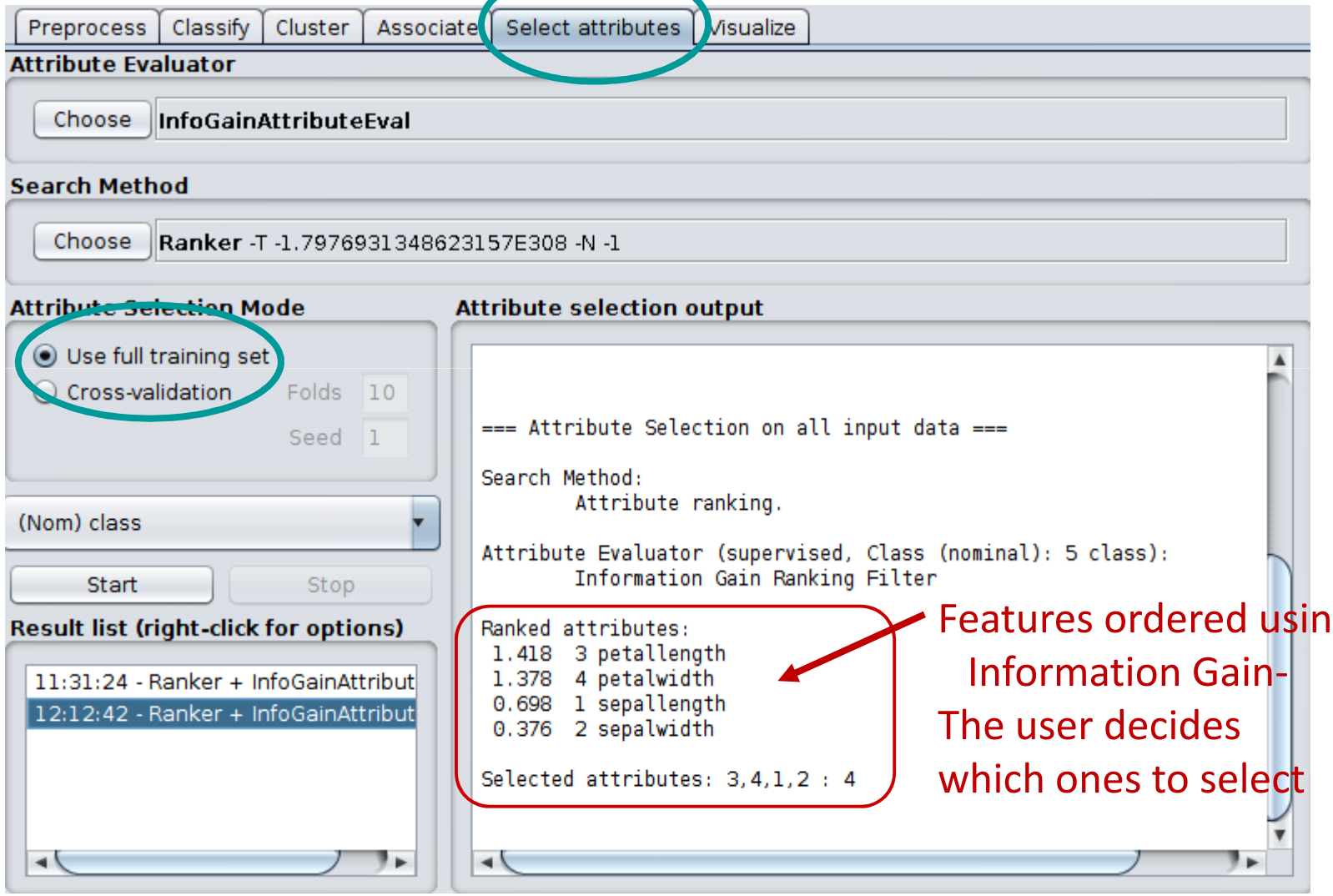
Filter/unsupervided/attribute/remove `attributeIndices`

Menu. Select attributes



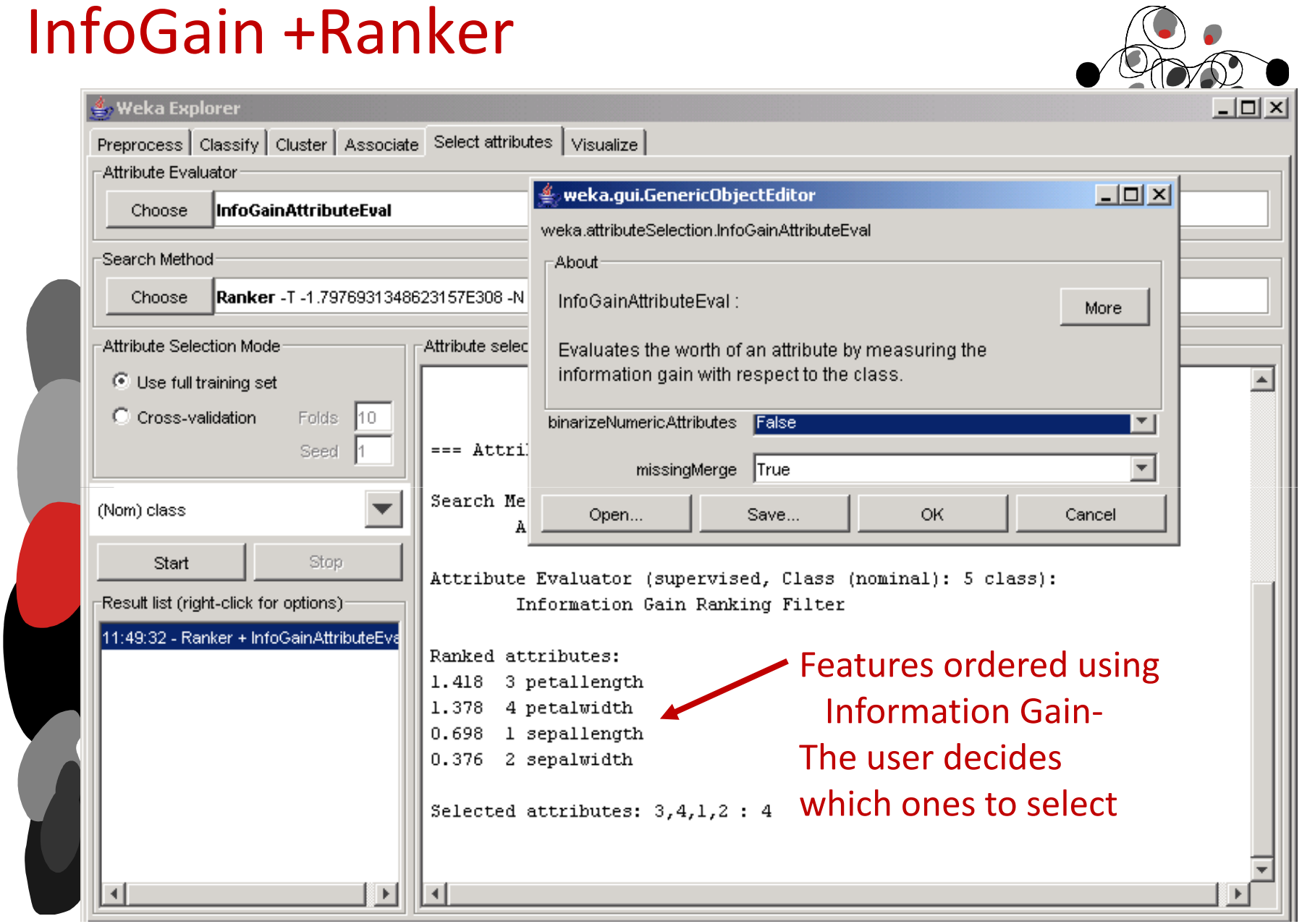
Menu. Select attributes



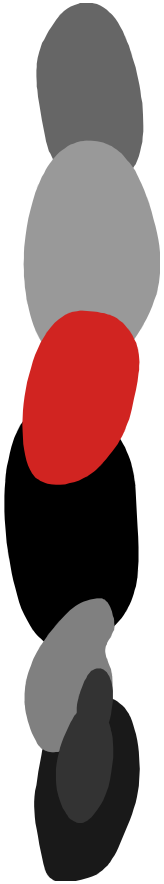


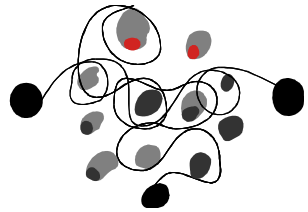
Features ordered using Information Gain- The user decides which ones to select

InfoGain +Ranker



Attribute selection
Example



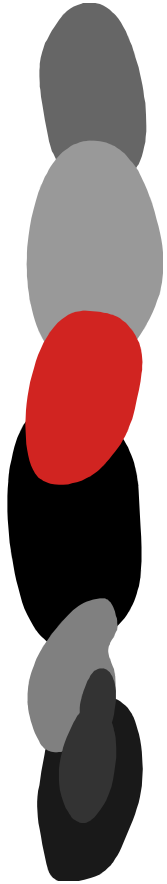


Iris:
4 features (% 96) → 2 features (%96)

Labor:
17 features (% 73,6) → 8 features (% 77,2)

Hit rates with j48 algorithm

Assignment: Soybean



Algorithm: J48

Percentaje split: % 66

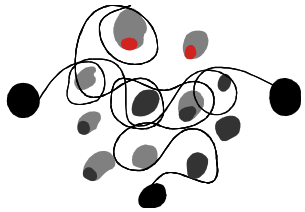
- All features Correctly classified instances = % 90,5

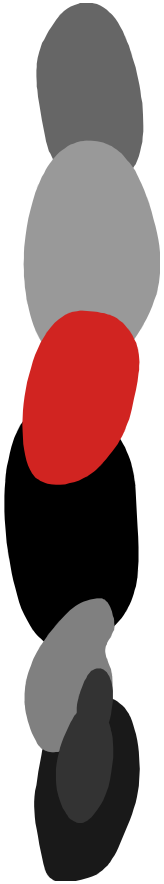
-Feature selection:

- `BestFirst+CfsSubsetEval`
- `InfoGain + Ranker` changing threshold

	Threshold	**N-feat.**	**%**	**F_1**
BF + CSE				
IF + Rank				
IF + Rank				
IF + Rank				

Assignment: Reuters_Grain_Test_WV





Algorithm: J48

Percentaje split: % 66

- All features

  Correctly classified instances = % 95,122 (F_1 = 64,3)

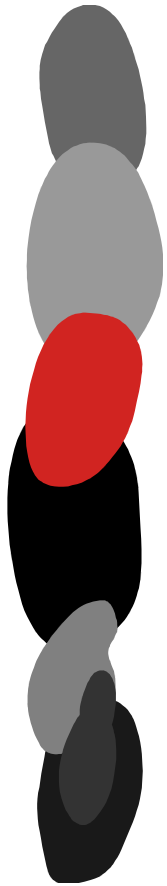
- Feature selection:

  `BestFirst+CfsSubsetEval`

  `InfoGain + Ranker` (changing *Threshold*)

	Threshold	**N-feat.**	**%**	**F_1**
BF + CSE				
IF + Rank				
IF + Rank				
IF + Rank				

Filters: Discretize



What does it do? Convert numeric data into discrete

Ex: age (5, 7, 18, 46, ….93)

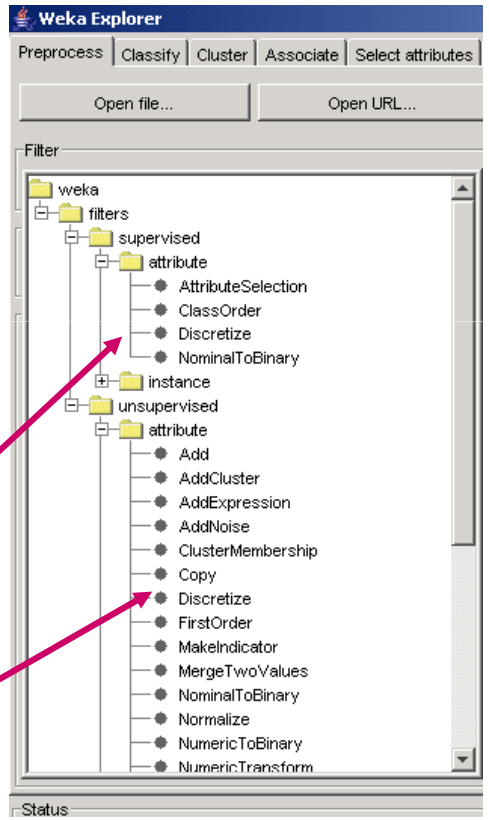
age {child, young, adult, elder}

What for?

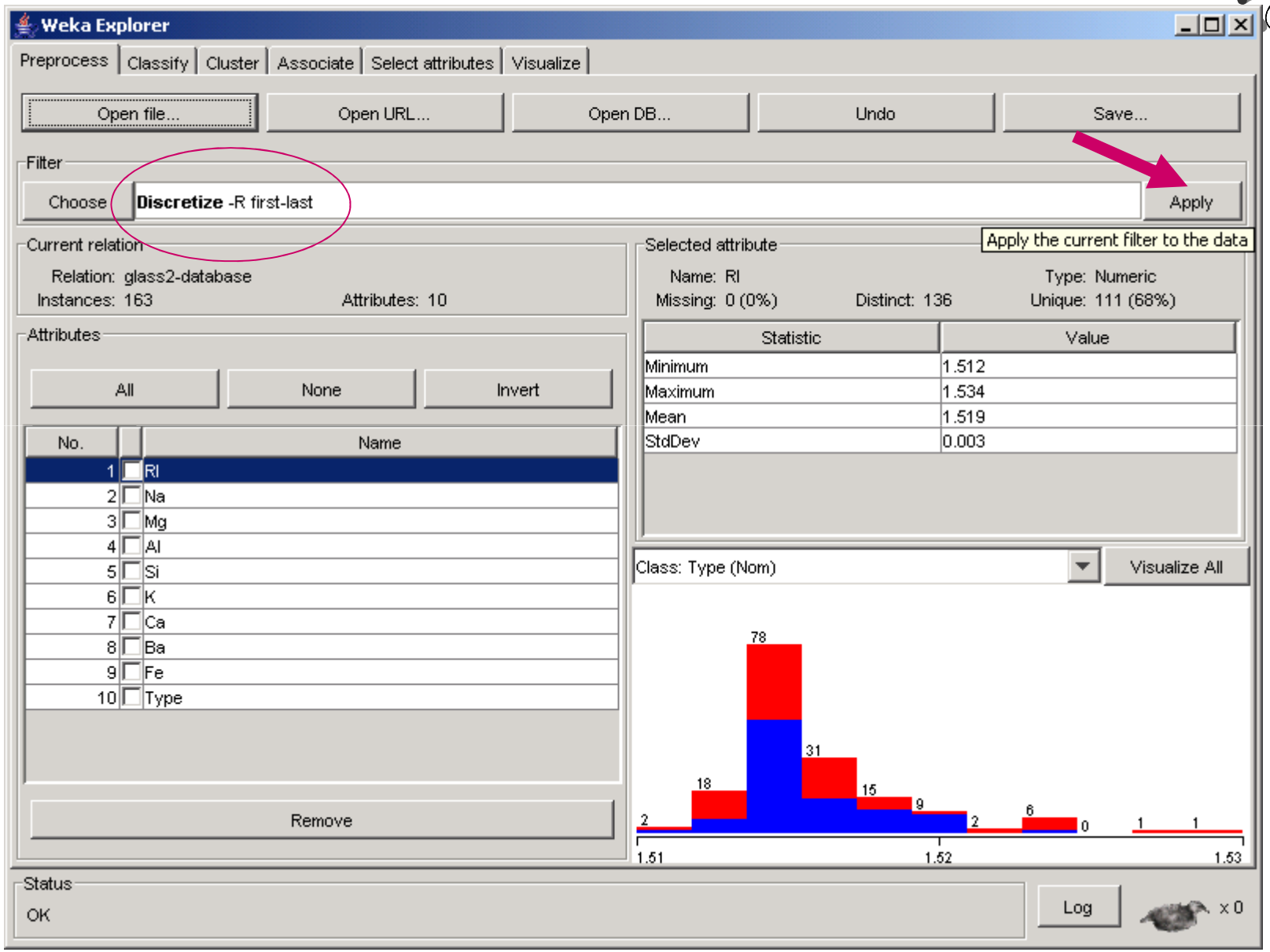
- Some classifiers require discrete features.
- Some methods (trees and rules) work slower with numeric features.
- Some clustering algorithms and NB consider that numeric features have a normal distribution.

Supervised: taking into account the class

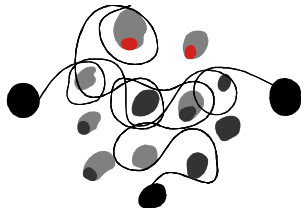
Unsupervised : without taking into account the class



Example: G2.arff



Filters: Discretize



Value

Number of instances

Selected attribute

Name: Rl Type: Nominal
Missing: 0 (0%) Distinct: 3 Unique: 0 (0%)

Label	Count
'(-inf-1.517155]'	53
'(1.517155-1.517985]'	42
'(1.517985-inf)'	68

Rl

Selected attribute

Name: Na Type: Nominal
Missing: 0 (0%) Distinct: 1 Unique: 0 (0%)

Label	Count
'All'	163

Na

Selected attribute

Name: Mg Type: Nominal
Missing: 0 (0%) Distinct: 2 Unique: 0 (0%)

Label	Count
'(-inf-2.495]'	13
'(2.495-inf)'	150

Mg

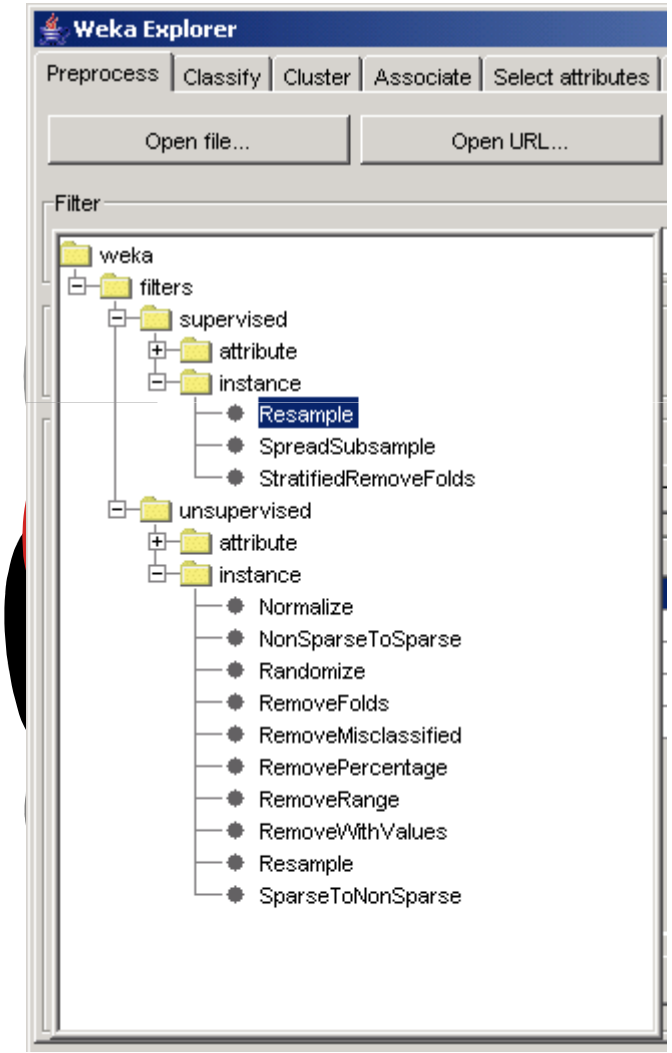
Selected attribute

Name: Al Type: Nominal
Missing: 0 (0%) Distinct: 2 Unique: 0 (0%)

Label	Count
'(-inf-1.42]'	108
'(1.42-inf)'	55

Al

Filters



HAP/LAP

Supervised

Instances

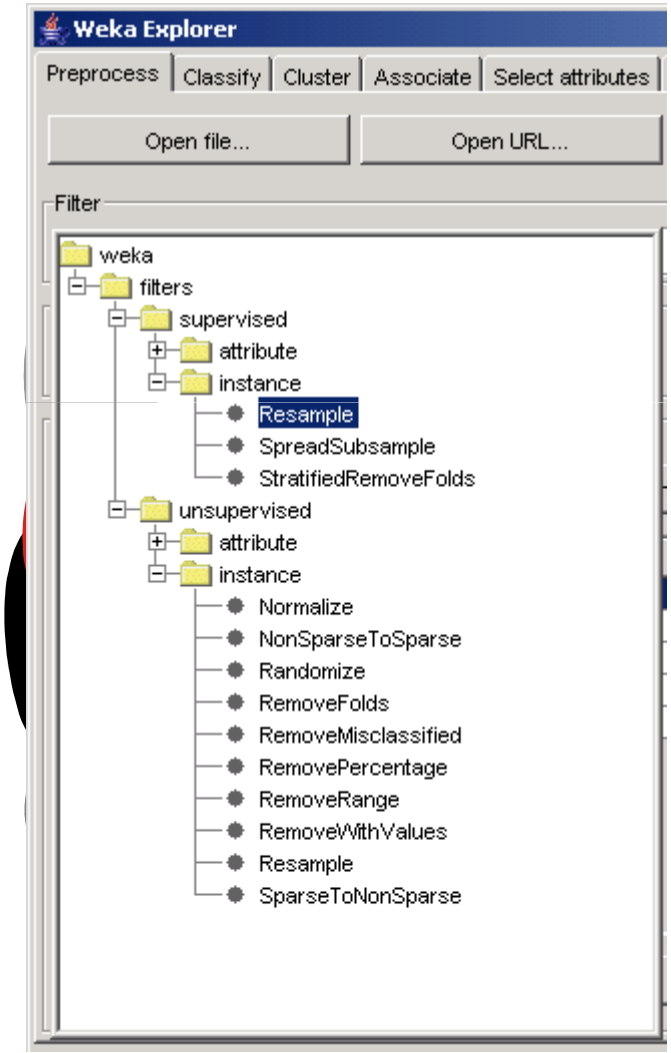
(re)sampling

Delete instances to maintain distribution when the number of examples in the categories is unbalanced, *sampling*

- **StratifiedRemoveFolds**

(for crossvalidation)

Filters. Resample

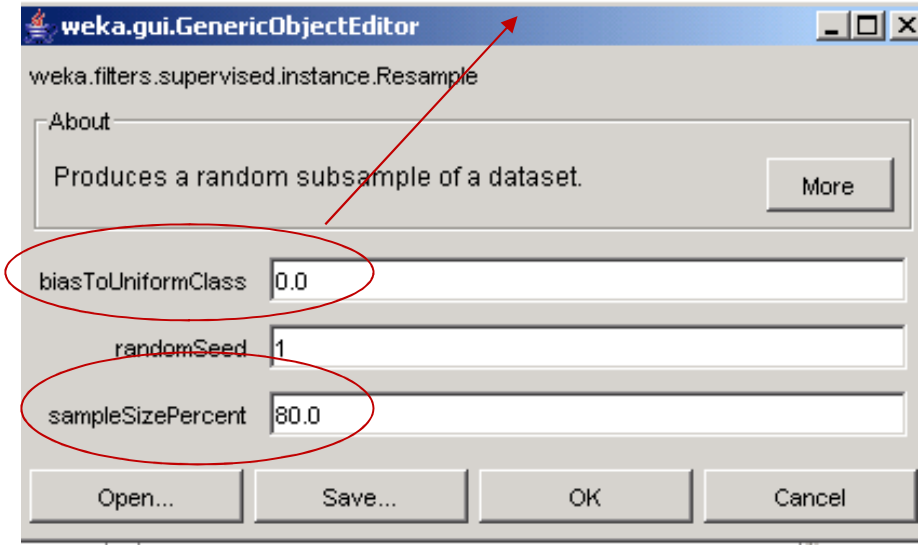


HAP/LAP

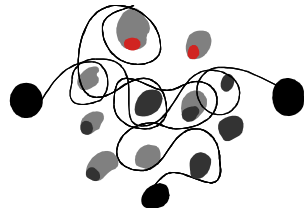
resample to generate subsets of data

Delete instances to maintain the distribution by categories (when the number of examples in unbalanced)

Maintain the class distribution or uniformimze



Filters. Resample

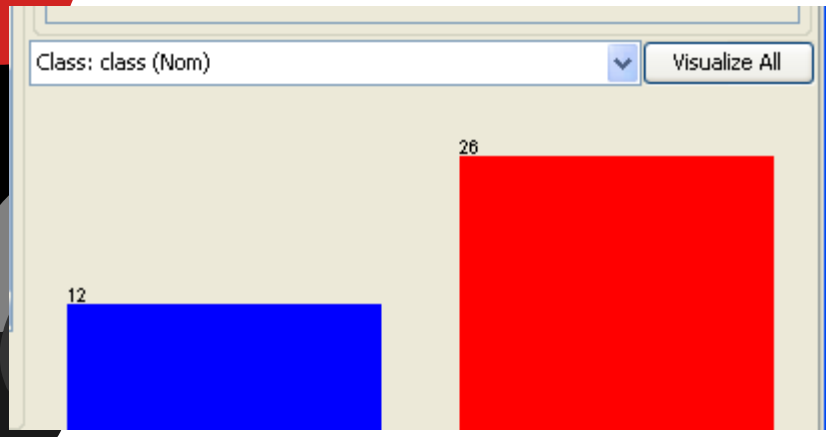


HAP/LAP

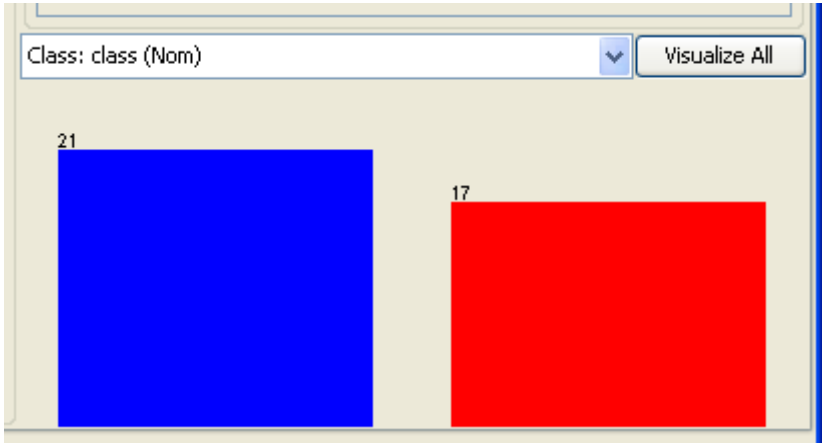
Labor

- Original: 57 instances
- Resample (% 67): 38 instances

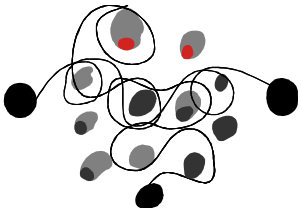
BiasToUniformClass = 0

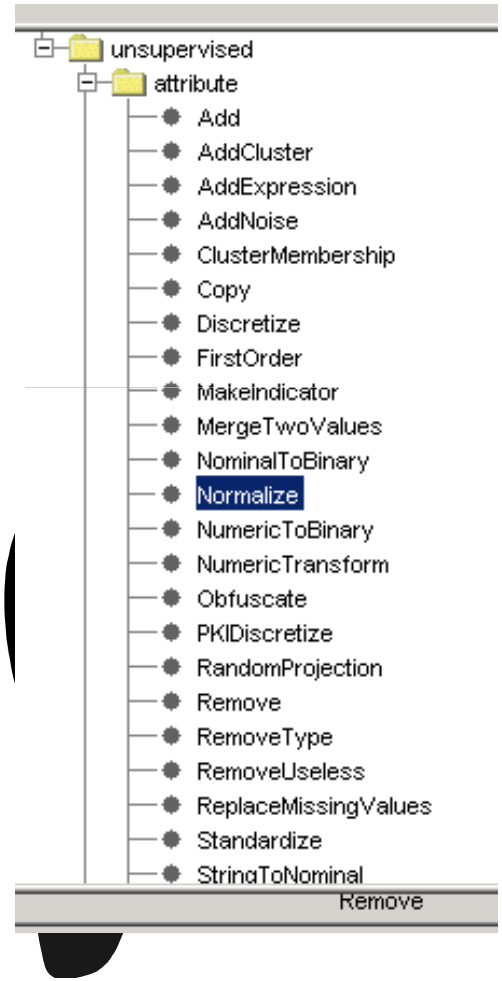


BiasToUniformClass = 1



Filters



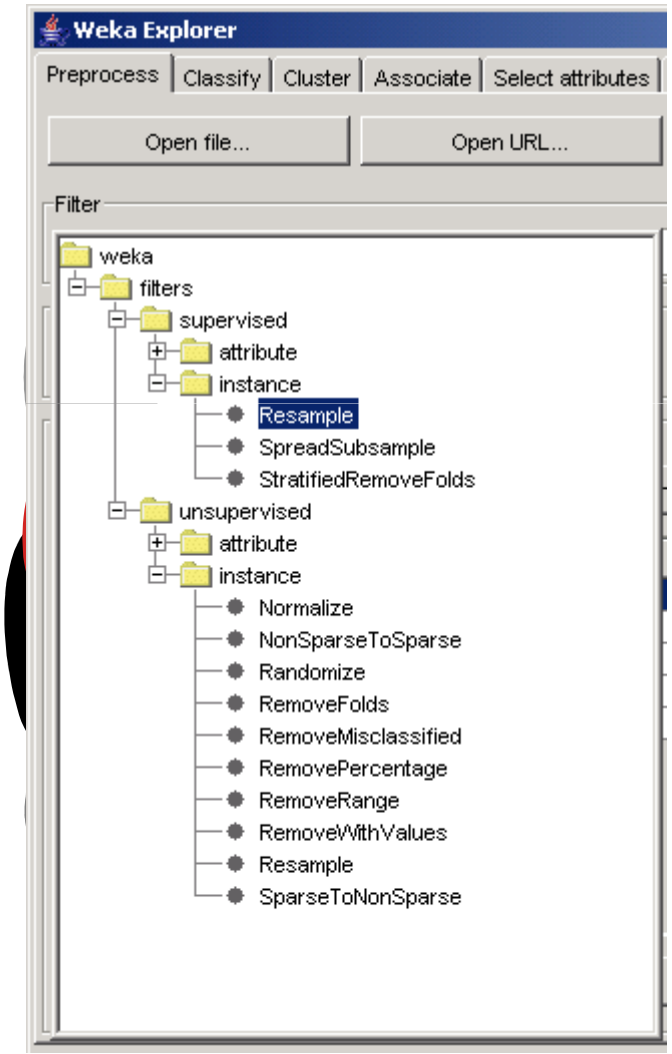


Unsupervised

Attribute:

- **Add:** add feature
- **AddExpresion:** generate new feature based on operation between features
- **AddNoise:** add noyse in attribute values
- **Copy:** copy attributes
- **Discretize**
- **Normalize**
- **NumericToBinary**
- **Remove:** remove features
- **StringToWordVector:** to work with texts

Filters



Unsupervised

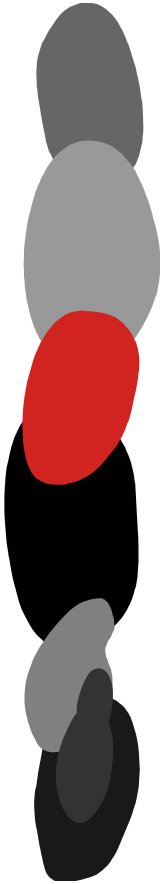
Instances

- **Remove** (delete)
  - RemoveFolds: delet a fold
  - RemoveMisclassified: delete wrongly classified

    (requires selecting classifier)
  - RemovePercentage: delete %

    → for **train/test division**
  - RemoveRange: which instances to delete

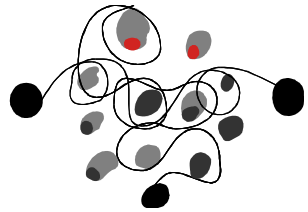
Working with texts: attribute selection



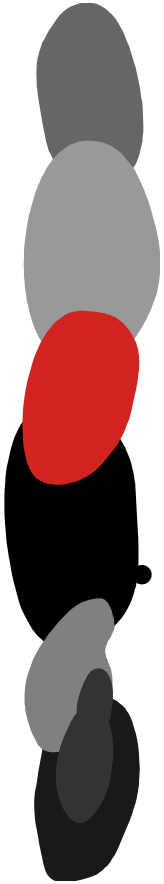
Selecting features

- Stop word-list:
  - Delete functional words
  - Delete very frequent words (they do not inform)
  - Delete rare words (errors, …)
- Use of a function to select features:
  - *InformationGain*
  - *Mutual Information*
  - *Chi-square*, ...
- Use of more complex techniques
  - (SVD) Singular Value Decomposition

Working with texts: features



- Problems with words:
  - Many algorithms do not treat them as texts
  - Transformation. Convert into number: bag of words (BoW) or vector space model (VSM)



document *x* is represented by vector $\phi(x)$ (pre)defined dictionary

- Ex: document classification

  d_i = book presentation in Koldo Mitxelenan …

  d_j = the writer will talk about the book in Koldo M.

  d_k = Europes' economy … the euro has risen …

- Representation (lemmas)

  d_i = book present Koldo Mitxelena …

  d_j = writer talk about book Koldo Mitxelena …

  d_k = Europe economy … euro rise…

- Features (dictionary):

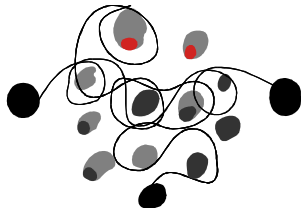
  {writer, book, present, author, novel, …, read, Koldo, Mitxelena, have, Europe, economy, euro, rise}

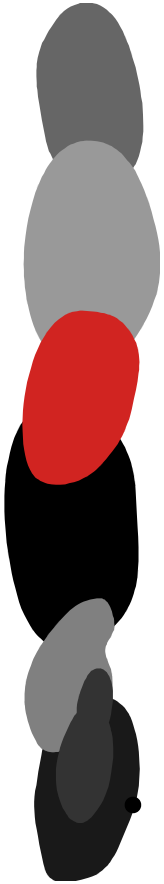
$d_1 = x \rightarrow \phi(x) = \{0,1,1,0,0,....,0,1,1,0,0,0,0,0\}$

$d_2 = z \rightarrow \phi(z) = \{1,1,0,0,0,....,0,1,1,1,0,0,0,0\}$

$d_3 = v \rightarrow \phi(x) = \{0,0,0,0,0,....,0,0,0,1,1,1,1,1\}$

Working with texts : features





- The vectors can be **binary** or **weighted**

  Binary: the term appears (1), does not appear (0).

  → default option

  Term frequency (tf): relative number of occurrences of the term

  → f_{ij} (where f_{ij} is the frequency of word i in document j).

  Document frequency (df) : number of training documents where the term t_k appears at least once

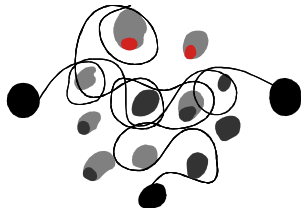
  Inverse document frequency (df) : to decrease the weight of words that appear very often in the corpus and increase it for words appearing few times

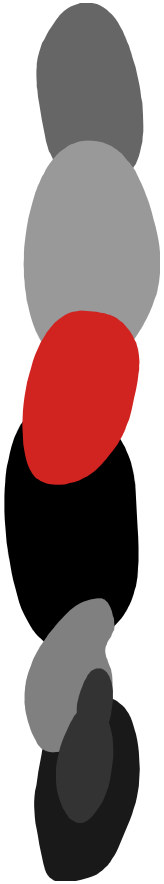
  term frecuency – inverse document frecuency

  tf-idf: if the term appears often→ it is meaningful

  if it appears in many texts → it is not discriminator

Working with texts : features



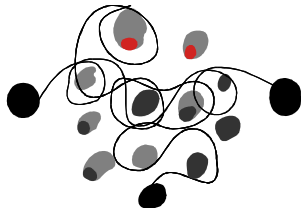


- **Document frequency #Tr(t_k):** in how many documents the word appears
  - It is possible to reduce the size of the documents 10 times
  - No efficiency lost
  - If #Tr is very small or very big, the word is not important.
  - Delete words appearing in 1 or two documents
- **Mutual Information**:

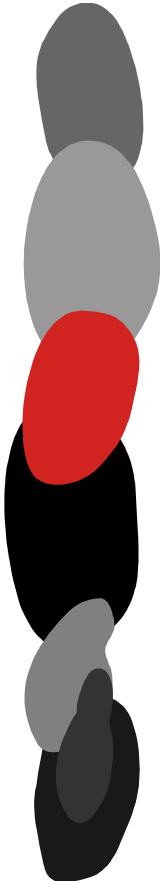
  relationship between the word (t_k) and the category (c_i)

  Chi-square, Information gain:
  - It takes into account the distribution of the word in different categories
  - It is possible to reduce the size of the documents 100 times

Working with texts : features



Term frecuency – inverse document frecuency (tf-idf)



tf-idf: if it appears often in the text→ is meaningful

if it appears in many texts → it is not discriminant

$$tfidf(t_k, d_j) = \#(t_k, d_j).\log \frac{|Tr|}{\#Tr(t_k)}$$

$\#(t_k, d_j)$ → number of occurrences of term *k* in document / num documents

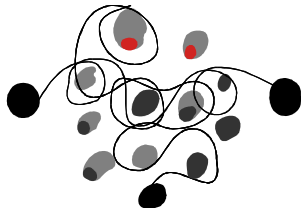
$\#Tr(t_k)$ → number of documents in train where t_k appears at least once

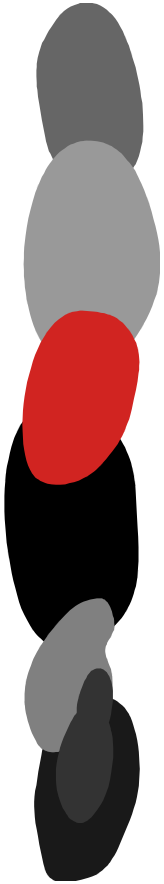
(often 1 + $\#Tr(t_k)$ to avoid divisions with 0

Tfidf

	t1	t2	t1	t1	t1	t1
Tr	100	100	100	100	100	100
# (tk, dj)	3	12	3	3	20	20
# Tr (tk)	50	50	90	10	10	90
tfidf (t1, dj)						
	t1 appears few times	t2 appears often	t1 appears few times in many documents	t1 appears few times in few documents	t1 appears many times in few documents	t1 appears many times in manydocuments

Filters: StringToWordVector





- Options of the filter:
  - Word frequency: weight for the words:
  - Number of occurrences
  - Capital letters

    Tokenizer: divide the words of the test

    limits: blank. , ; : ‘ “ ( ) ? !
  - Stemmer: root of the word

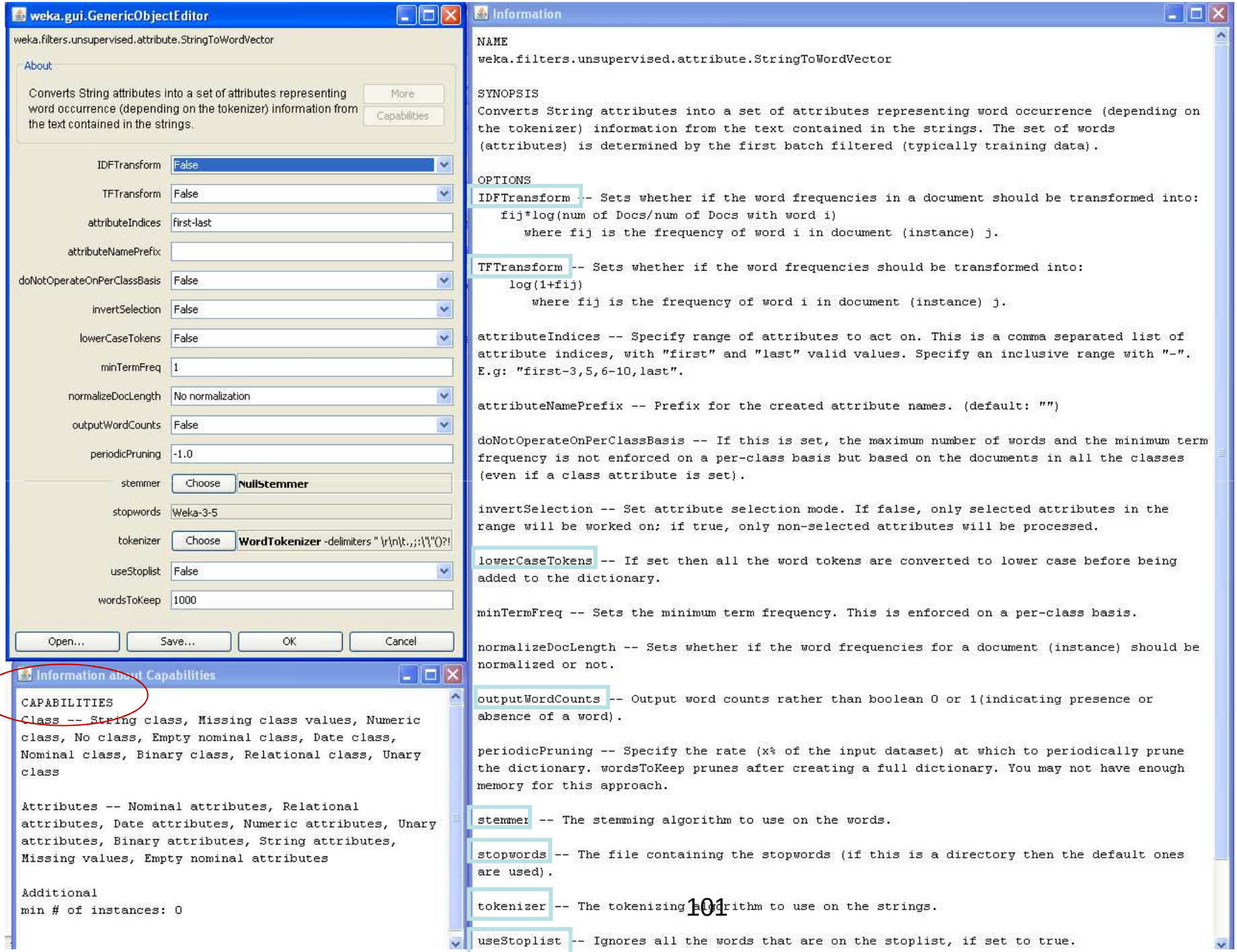
    Porter algorithms (most used)

    http://snowball.tartarus.org/algorithms/porter/stemmer.html

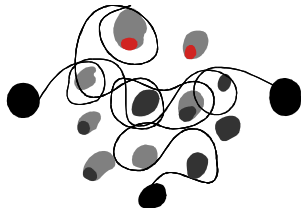
    http://weka.wikispaces.com/Stemmers
  - StopList: words without “value”, functional words (depends on the task)

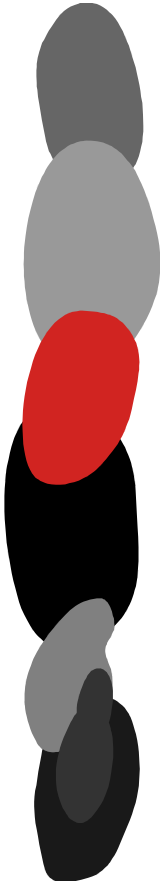
    or, and, is, …

    http://www.ranks.nl/stopwords


weka.gui.GenericObjectEditor
weka.filters.unsupervised.attribute.StringToWordVector
About
Converts String attributes into a set of attributes representing word occurrence (depending on the tokenizer) information from the text contained in the strings.
More
Capabilities
IDFTransform
False
TFTransform
False
attributeIndices
first-last
attributeNamePrefix
doNotOperateOnPerClassBasis
False
invertSelection
False
lowerCaseTokens
False
minTermFreq
1
normalizeDocLength
No normalization
outputWordCounts
False
periodicPruning
-1.0
stemmer
Choose
NullStemmer
stopwords
Weka-3-5
tokenizer
Choose
WordTokenizer -delimiters " \r\n\t.,;:\'\"()?!
useStoplist
False
wordsToKeep
1000
Open...
Save...
OK
Cancel
Information about Capabilities
CAPABILITIES
Class -- String class, Missing class values, Numeric class, No class, Empty nominal class, Date class, Nominal class, Binary class, Relational class, Unary class
Attributes -- Nominal attributes, Relational attributes, Date attributes, Numeric attributes, Unary attributes, Binary attributes, String attributes, Missing values, Empty nominal attributes
Additional
min # of instances: 0
Information
NAME
weka.filters.unsupervised.attribute.StringToWordVector
SYNOPSIS
Converts String attributes into a set of attributes representing word occurrence (depending on the tokenizer) information from the text contained in the strings. The set of words (attributes) is determined by the first batch filtered (typically training data).
OPTIONS
IDFTransform -- Sets whether if the word frequencies in a document should be transformed into:
fij*log(num of Docs/num of Docs with word i)
where fij is the frequency of word i in document (instance) j.
TFTransform -- Sets whether if the word frequencies should be transformed into:
log(1+fij)
where fij is the frequency of word i in document (instance) j.
attributeIndices -- Specify range of attributes to act on. This is a comma separated list of attribute indices, with "first" and "last" valid values. Specify an inclusive range with "-". E.g: "first-3,5,6-10,last".
attributeNamePrefix -- Prefix for the created attribute names. (default: "")
doNotOperateOnPerClassBasis -- If this is set, the maximum number of words and the minimum term frequency is not enforced on a per-class basis but based on the documents in all the classes (even if a class attribute is set).
invertSelection -- Set attribute selection mode. If false, only selected attributes in the range will be worked on; if true, only non-selected attributes will be processed.
lowerCaseTokens -- If set then all the word tokens are converted to lower case before being added to the dictionary.
minTermFreq -- Sets the minimum term frequency. This is enforced on a per-class basis.
normalizeDocLength -- Sets whether if the word frequencies for a document (instance) should be normalized or not.
outputWordCounts -- Output word counts rather than boolean 0 or 1(indicating presence or absence of a word).
periodicPruning -- Specify the rate (x% of the input dataset) at which to periodically prune the dictionary. wordsToKeep prunes after creating a full dictionary. You may not have enough memory for this approach.
stemmer -- The stemming algorithm to use on the words.
stopwords -- The file containing the stopwords (if this is a directory then the default ones are used).
tokenizer -- The tokenizing algorithm to use on the strings.
useStoplist -- Ignores all the words that are on the stoplist, if set to true.

Example

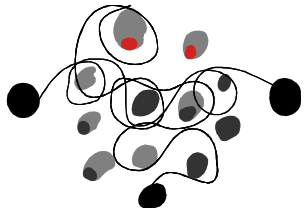




```
@relation tc

@attribute dok string
@attribute kat {kirola kultura politika}

@data
'Euskaltel Euskadi txirrindulari prentsaurreko' kirola
'liburu aurkeztu koldo mitxelena' kultura
'liburu berri koldo mitxelena idazle' kultura
'liburu idazle koldo mitxelena' kultura
'baloia Euskadi partida' kirola
'baloia partida' kirola
'lehendakaria hauteskundeak' politika
```

weka.filters.unsupervised.attribute.StringToWordVector -R first -W 1000 -prune-rate -1.0 -N 0 -stemmer weka.core.stemmers.NullStemmer -M 1 -tokenizer "weka.core.tokenizers.WordTokenizer -delimiters \" \\r\\n\\t.,;:\\\'\\\"()?!\""

Example



HAP/LAP

```
@relation tc

@attribute dok string
@attribute kat {kirola kultura politika}

@data
'Euskaltel Euskadi txirrindulari
prentsaurreko' kirola
'liburu aurkeztu koldo mitxelena' kultura
'liburu berri mitxelena idazle' kultura
'liburu idazle mitxelena' kultura
'baloia Euskadi partida' kirola
'baloia partida' kirola
'lehendakaria hauteskundeak' politika
```

```
   @relation 'tc-weka.filters.unsupervised…

   @attribute kat {kirola,kultura,politika}
0  @attribute Euskadi numeric
1  @attribute Euskaltel numeric
2  @attribute baloia numeric
3  @attribute partida numeric
4  @attribute prentsaurreko numeric
5  @attribute txirrindulari numeric
6  @attribute aurkeztu numeric
7  @attribute berri numeric
8  @attribute idazle numeric
9  @attribute koldo numeric
10 @attribute liburu numeric
11 @attribute mitxelena numeric
12 @attribute hauteskundeak numeric
13 @attribute lehendakaria numeric
```

binary

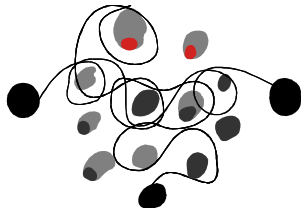
```
@data
{0 1,1 1,4 1,5 1}
{6 1,9 1,10 1,11 1,14 kultura}
{7 1,8 1,10 1,11 1,14 kultura}
{8 1,10 1,11 1,14 kultura}
{0 1,2 1,3 1}
{2 1,3 1}
{12 1,13 1,14 politika}
```

0 ezaugarria “Euskadi” ageri da

tfidf

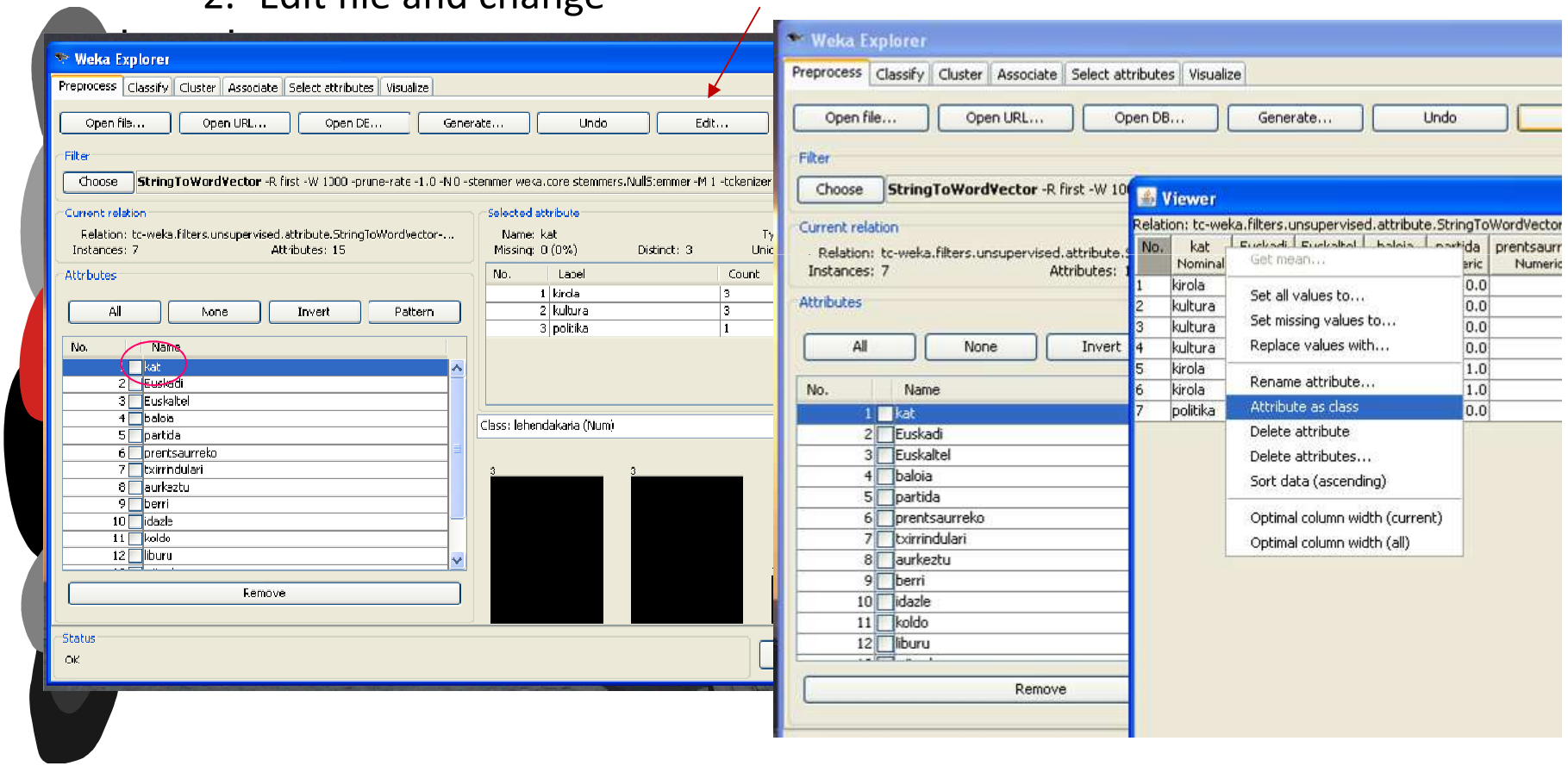
```
@data
{0 0.868349,1 1.348802,4 1.348802,5 1.348802}
{6 1.348802,9 1.348802,10 0.587302,11 0.587302,14 kultura}
{7 1.348802,8 0.868349,10 0.587302,11 0.587302,14 kultura}
{8 0.868349,10 0.587302,11 0.587302,14 kultura}
{0 0.868349,2 0.868349,3 0.868349}
{2 0.868349,3 0.868349}
{12 1.348802,13 1.348802,14 politika}
```

Filters: StringToWordVector

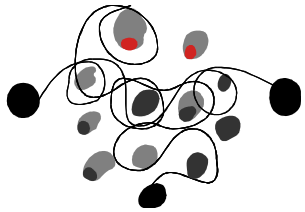


HAP/LAP

- To move the category to the last position:
  1.- weka.filters.unsupervised.attribute.Reorder (Reorder 2-last,1)
  2.- Edit file and change



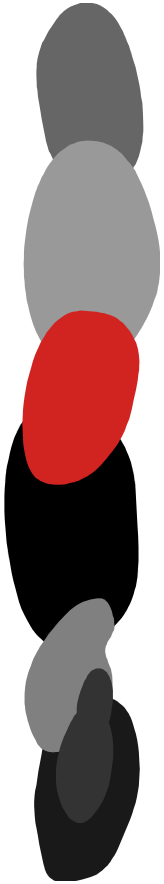
StringToWordVector



HAP/LAP

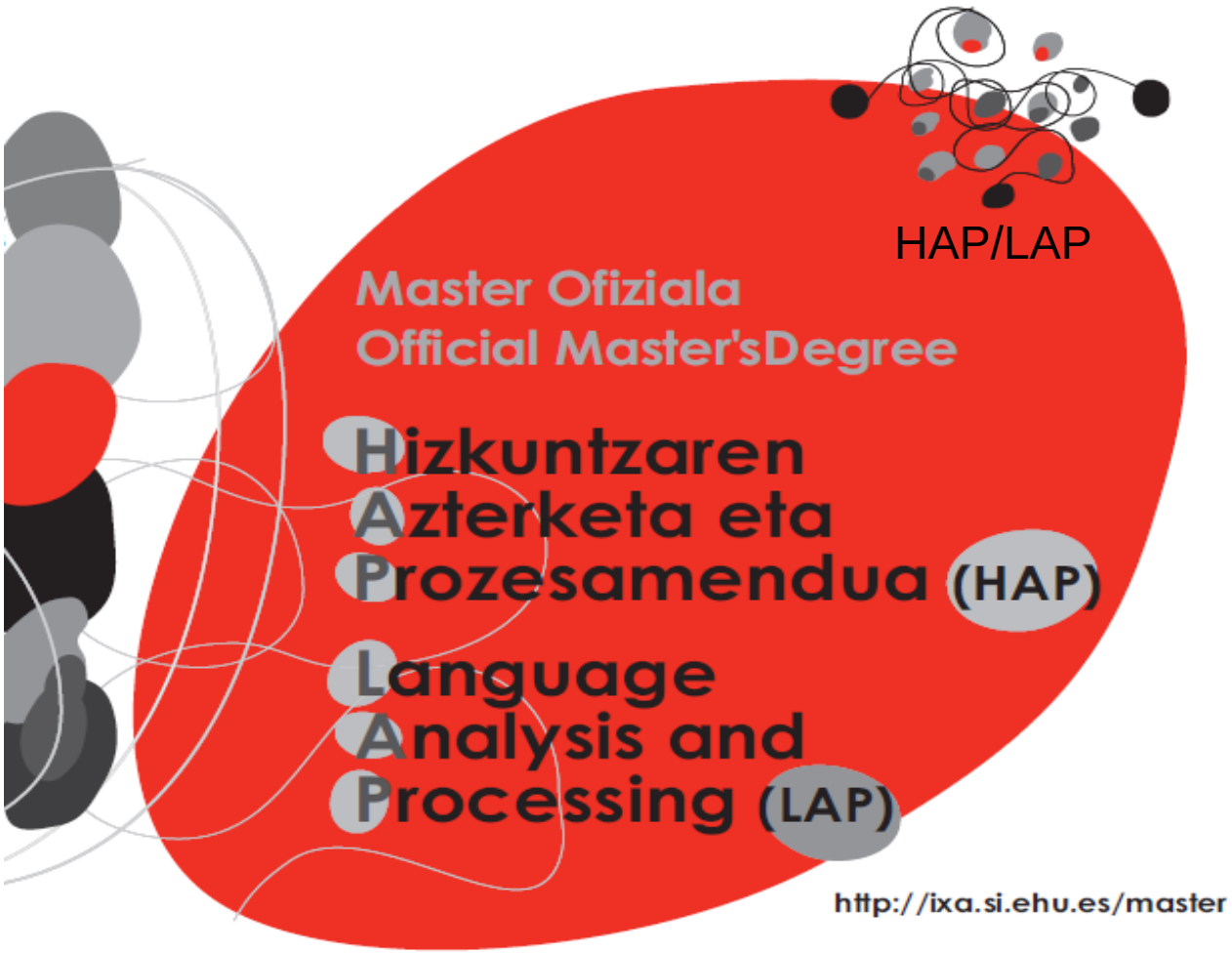
Assignment

- Open ReutersGrainTrain
- Apply filter StringToWordVector with default values
- Move the class to the end
- Try J48 classifier (train-test option: *Percentage split)*
- Apply filter StringToWordVector including stopword list
- Move the class to the end
- Try J48 classifier
- Select attributes (different options)
- Try J48



	Threshold	N_features	%	F_1
BF + CSE				
I_Gain+Ranker				
I_Gain+Ranker				
SymmetricalUn…				
SymmetricalUn…				

Erasmus Mundus



Language and Communication Technologies

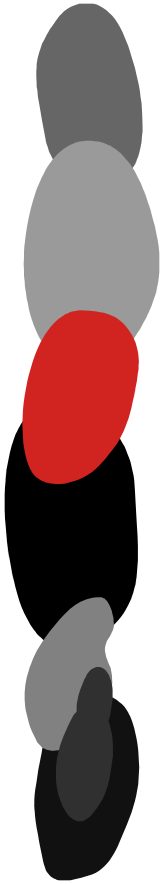
ixa

ueu
udako
euskal unibertsitatea

Universidad del País Vasco
Euskal Herriko Unibertsitatea
MDe
Master eta Doktorego Eskola
Escuela de Máster y Doctorado
Master and Doctoral School

lct

Erasmus Mundus



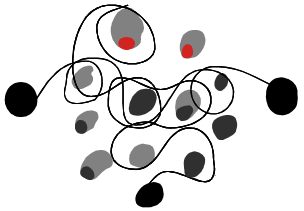
HAP/LAP

Statistical methods and text corpora

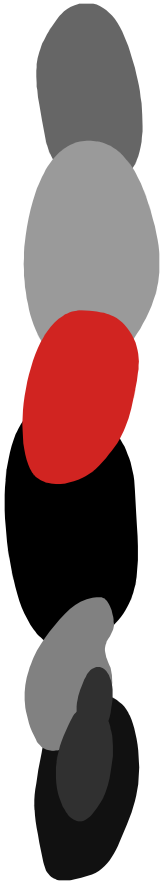
Introduction to Machine Learning for Natural Language Processing

Olatz Arbelaitz: olatz.arbelaitz@ehu.eus

Erasmus Mundus



Topics

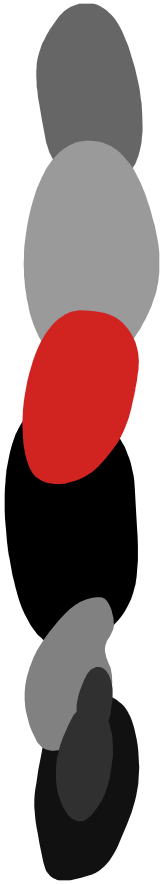


1.- Machine Learning in LNP

2.- WEKA software:

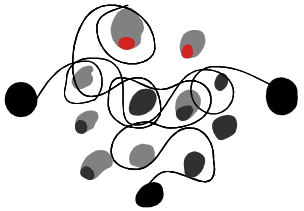
- 2.1.-Attribute (feature) selection
- 2.3.- **Basic algorithms: Naive Bayes, K-NN, Decision Trees, Rules, …**
- 2.3.-Evaluation

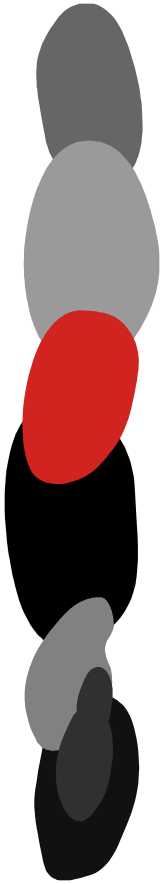
Classification process



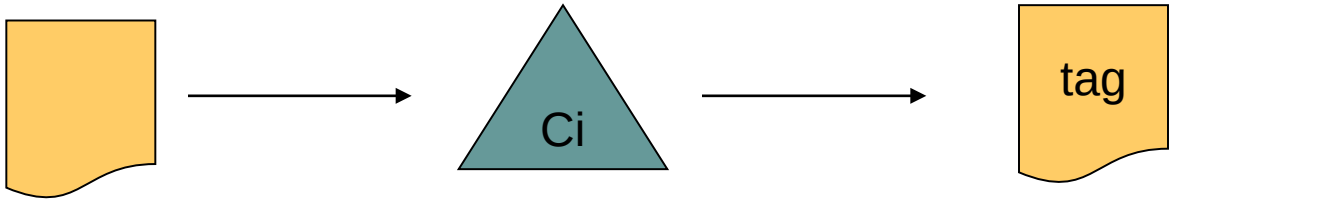
- Division of the corpora (*Test options*)
  - train / test
  - Cross-validation
- Classifier (*Classify*)
  - Set parameters
- Evaluation (*Classifier Output*)
  - Confusion matrix
  - Precision/recall
  - Microaveraging/macroaveraging

Division of the corpora



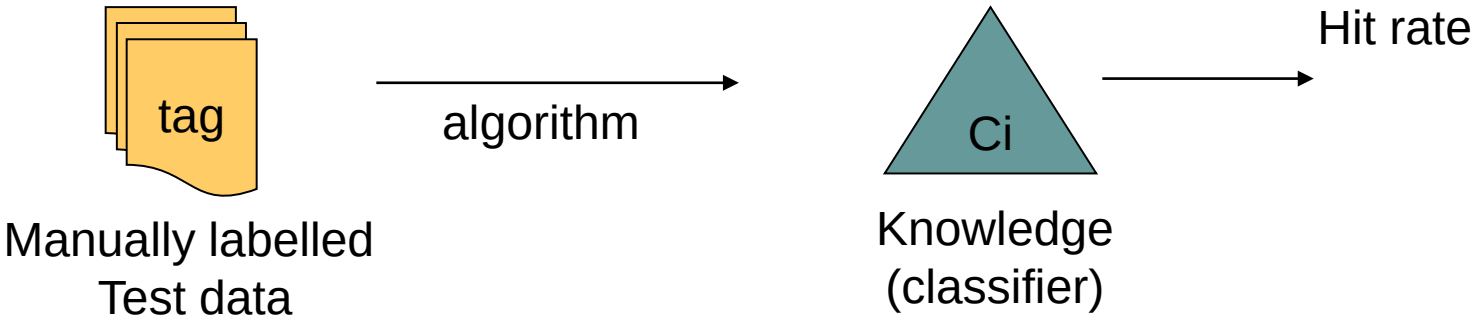


After learning → Make decisions, **obtain results**

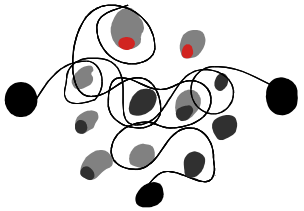


Is the obtained result good?

Test → to measure the quality of the built classifier



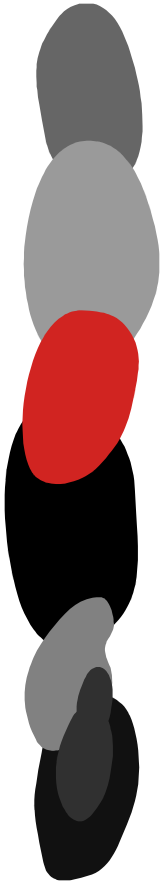
Learning Process

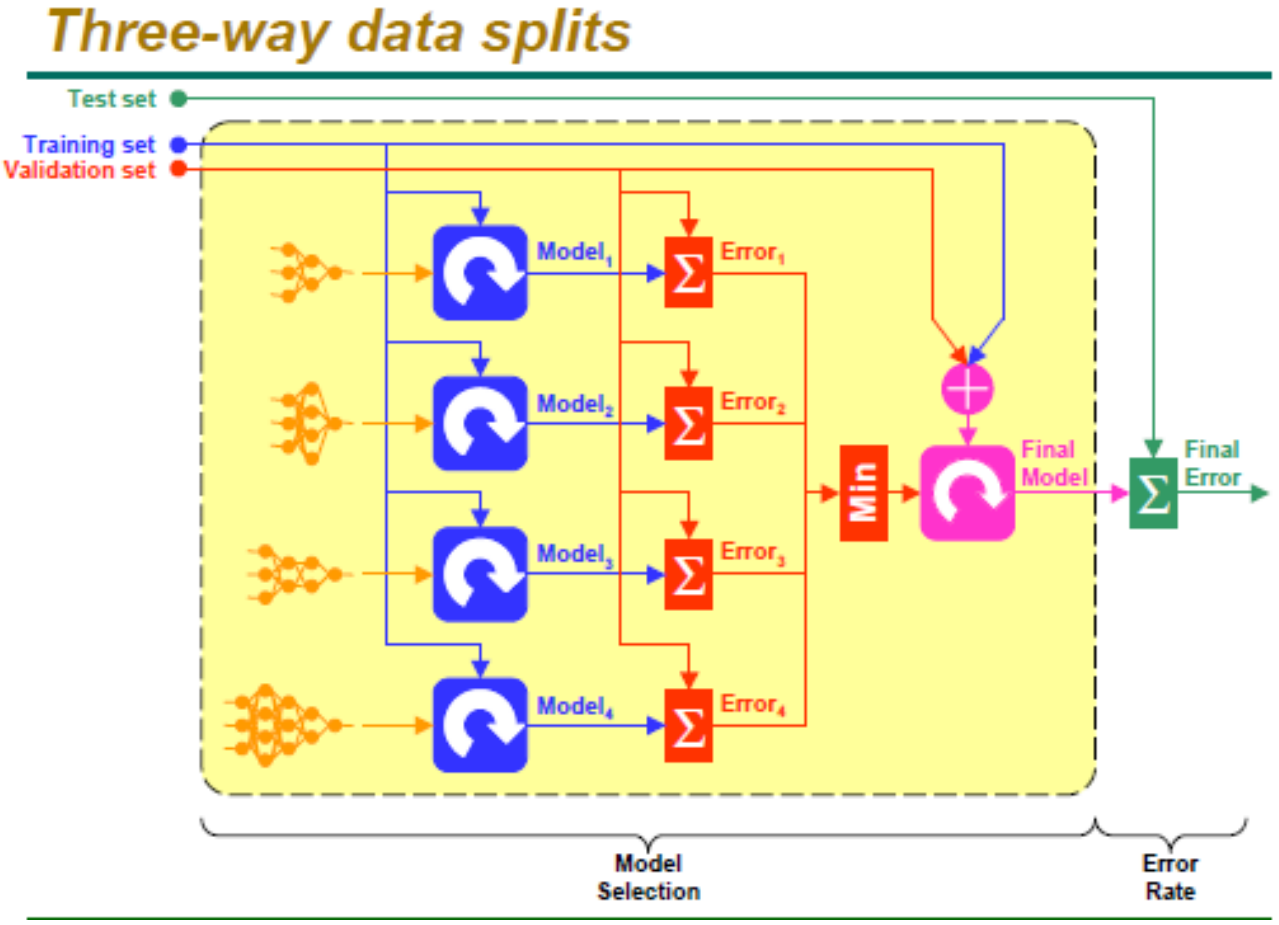


1. Divide the available data into training, validation and test set
2. Select architecture and training parameters
3. Train the model using the **training set**
4. Evaluate the model using the **validation set**
5. Repeat steps 2 through 4 using different architectures and training parameters
6. Select the best model and train it using data from the training and validation set
7. Assess this final model using the **test set**

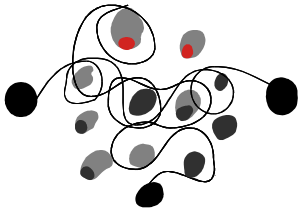
The error rate obtained with the validation set is often smaller than the real error because this set has bee used to select the model.

Learning Process

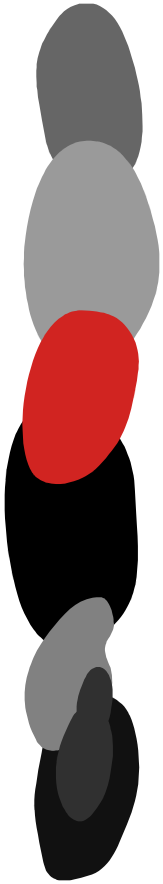




Division of the Corpora



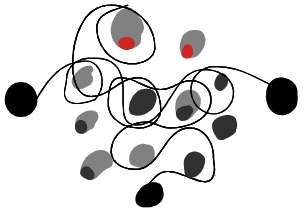
- **Train / Test**
  - Train: % 80 - % 66 → learning
  - Validation: part of the training (% 15 - % 20) → tuning
  - Test: % 20 - % 33 → evaluate (solution must be known)



Results can change depending on the division

Train (%66)		**Test (% 33)**
Training set	Validation set	Test set

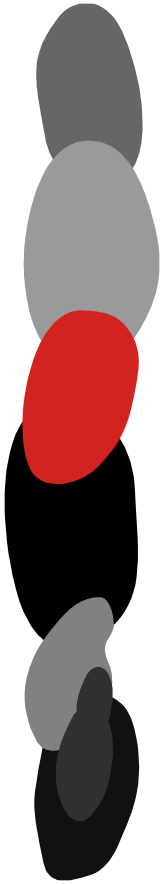
Division of the Corpora



- **Cross-validation:** for small corpora
  - Division in *k* folds (10)
  - *k*-1 train and 1 test
  - Learn *k* times
  - Average error rate: errors in the *k* iterations/*k*

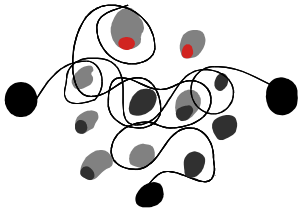
    10-folds or 2 times 5-folds

    5-folds (when the corpora is small)

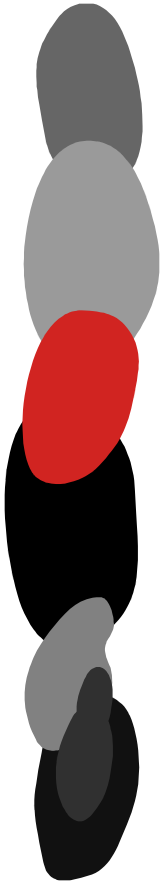


Exp. 1					**Test set**
Exp. 2				**Test set**	
Exp. 3			**Test set**		
Exp. 4		**Test set**			
Exp. 5	**Test set**				

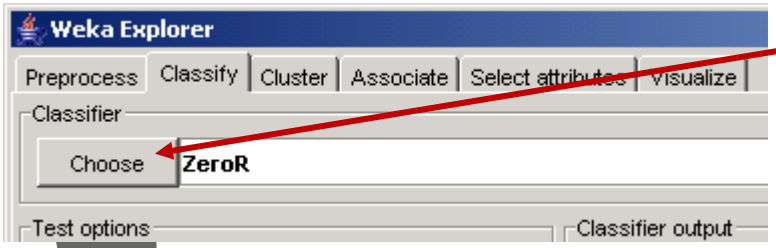
Division of the Corpora

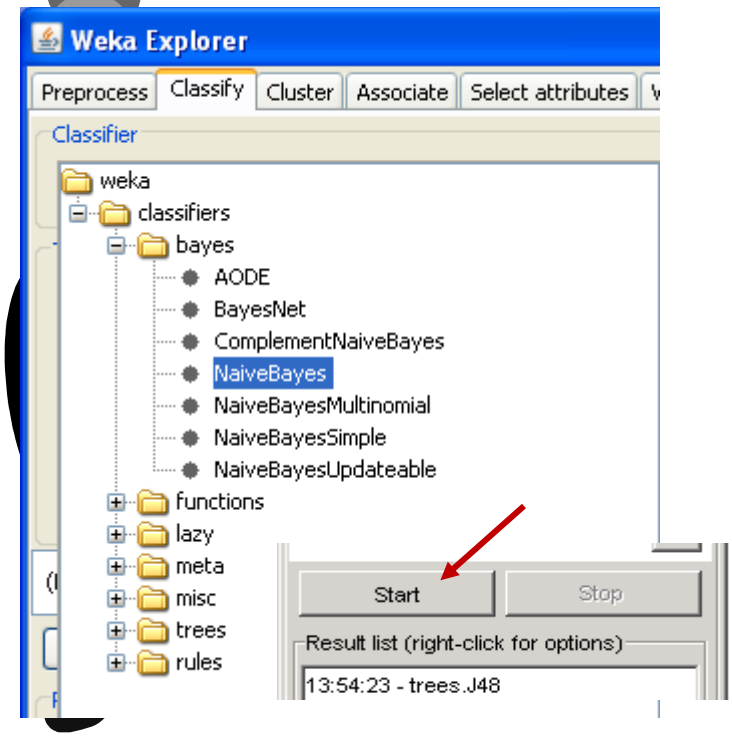


- **Leave-one-out** cross-validation:
  - Similar to cross-validation but a single example used for testing in each iteration and the rest of the examples are used for training



Types of classifiers





Lazy: based on instances (knn)

IB1, IBk

Bayes: based on Bayes theorem

NaiveBayes

Trees: based on trees

J48, RandomForest

Rules:

PART, OneR

Functions: based on linear functions

MultilayerPerceptron,
SMO, Winnow

Meta: classifier combination

AdaBoostM1, vote, stacking

HAP/LAP

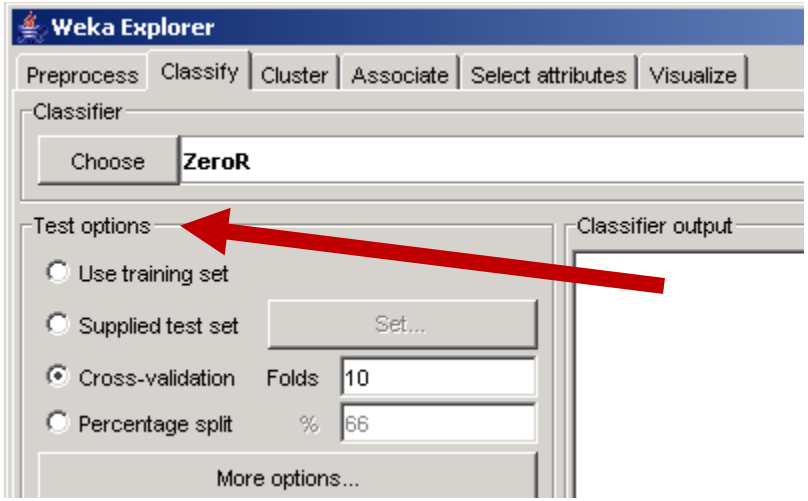
Division of the Corpora

HAP/LAP

To divide the corpora:

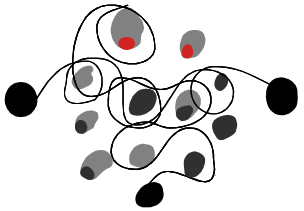
Test options:

- Use training set: use all the examples used for learning → NO

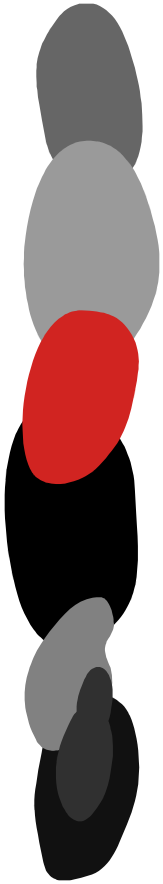


- Supplied test set: provide test file
  - press *Set* and select file
- Cross - validation (*fold* indicate number of folds)
- Percentage split (percentage used for learning)

For train/test division

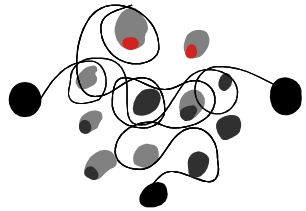


The class is not taken into account in the division!

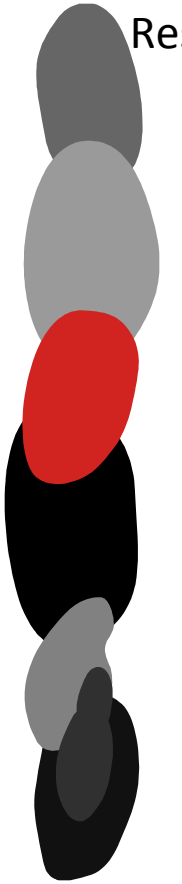


- RemovePercentage: for dividing train/test
  - training set:
    - Open complete file
    - Select **RemovePercentage** filter
    - Write percentage desired for division
    - Apply filter
    - Save the generated dataset as a new file
  - test set:
    - Open complete file (or use undo)
    - Select **RemovePercentage** filter
    - Activate **invertSelection** in the filter
    - Apply filter
    - Save the generated dataset as a new file

For train/test division



To divide taking into account the class



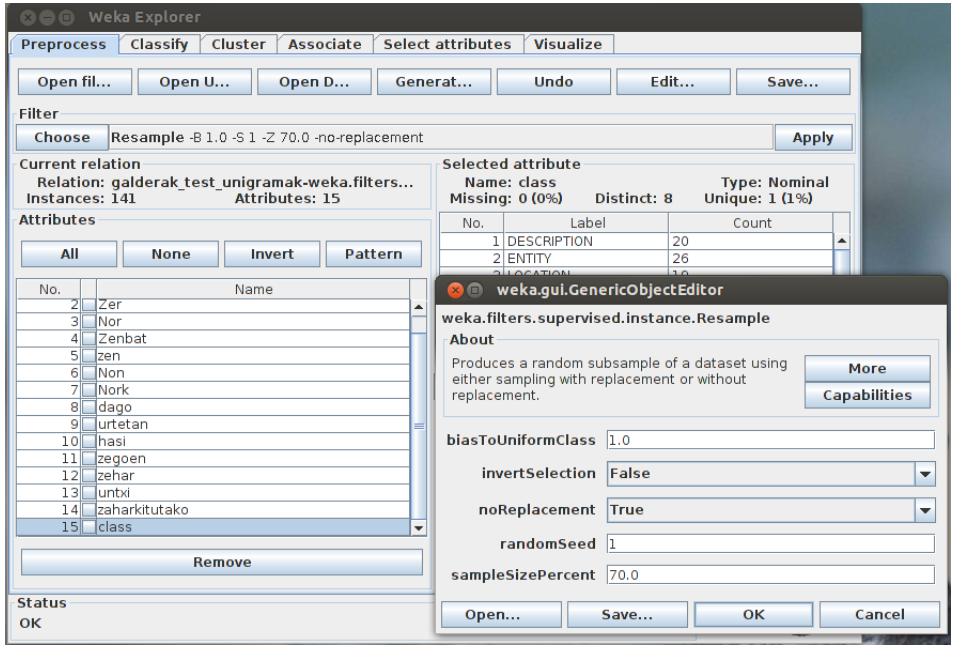
Resample

training set:

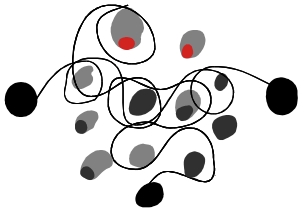
- Open complete file
- Select **Resample (supervised)** filter
- Write desired percentage
- Select BiasToUniformClass
- noReplacement: true
- Apply filter
- Save the generated dataset as a new file

test set:

- Open complete file (or undo)
- Select **invertSelection** in filter
- Apply filter
- Save the generated dataset as a new file



How to evaluate?



Binary classifier:

Real→ C = 1 positive class C = 0 negative class

Prediction→ C_M = 1 positive class predicted and C_M = 0 negative class predicted

		Real class	
		C = 1	C = 0
Predicted Class	C_M = 1	TP	FP
	C_M = 0	FN	TN

(Confusion Matrix)

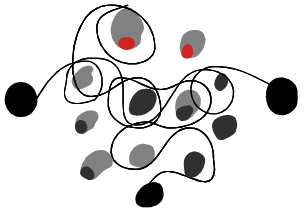
TP = True Positive

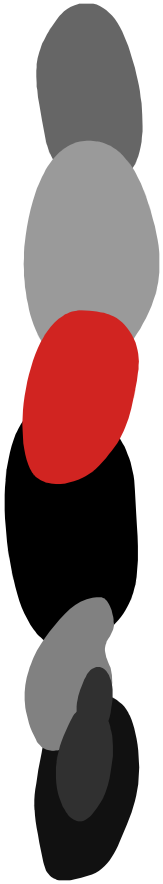
TN = True Negative

FP = Fase Positive

FN = False Negative

How to evaluate?

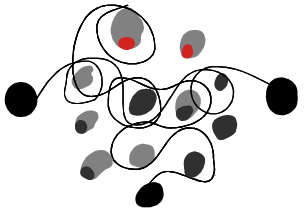




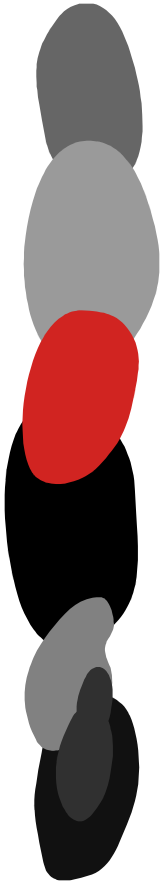
- When evaluating we take into account:
  - **Precision**: taking into account the examples classified into a category, number of hits among them
  - **Recall**: among the examples of a category, number of hits
  - Combination of both metrics (*F-score/F-measure*)
- Which is more important?
  - **precision**: what the system says is always correct (→ although it might say it fewer times)

    or,
  - **recall**: although with some mistakes to say it more

How to evaluate?



Binary classifier:



Hit rate (accuracy): proportion of examples with correct prediction among the tested examples.

$$Accuracy(AC) = \frac{TP + TN}{TP + TN + FP + FN}$$

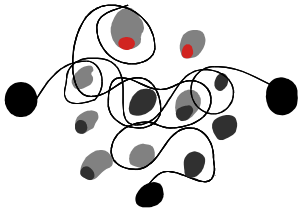
Precision taking into account the examples classified as being of the positive class, number of hits among them

$$\Pr ecision = \frac{TP}{TP + FP}$$

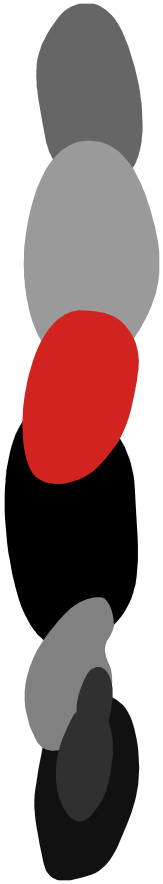
Recall: number of hits among the examples of the positive class which where tested,

$$\mathrm{Re}\, call = \frac{TP}{TP + FN}$$

How to evaluate?



Binary classifier:



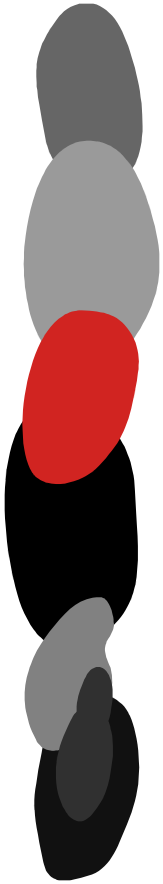
F_1-score (F-measure): Harmonic mean of the precision and

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

Error: proportion of examples with wrong prediction among the tested examples

$$Error(1-AC) = \frac{FP + FN}{TP + TN + FP + FN}$$

How to evaluate?



- Example: *weather.arff*
  - 4 features, 100 instances, class={**good, bad**}
  - Classifier: *rules* **JRip**
  - Test options: Percentage split (% 66) → 34 instan.

    - Classifier Output

    === Run information ===

    Scheme: weka.classifiers.rules.JRip -F 3 -N 2.0 -O 2 -S 1

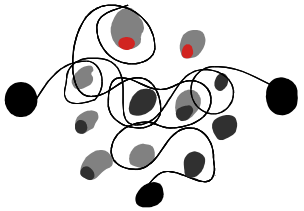
    Relation: weather

    Instances: 100

    Attributes: 5 outlook, temperature, humidity, windy, play

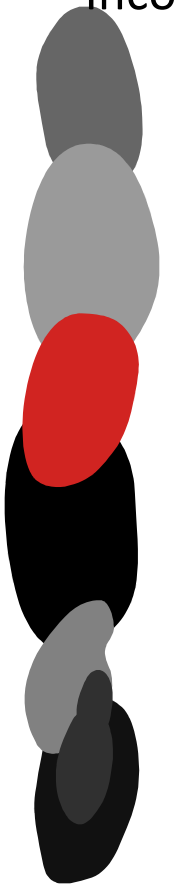
    Test mode: split 66.0% train, remainder test

How to evaluate?



Correctly Classified Instances	28	82.3529 %	
Incorrectly Classified Instances	6	17.6471 %	percentage

28*100/34
6*100/34



=== Confusion Matrix ===

a	b	classified as
18	1	a = good
5	10	b = bad

Category		classified as	
		YES	**NO**
real	**YES**	TP	FN
	NO	FP	TN

n

category; **good**

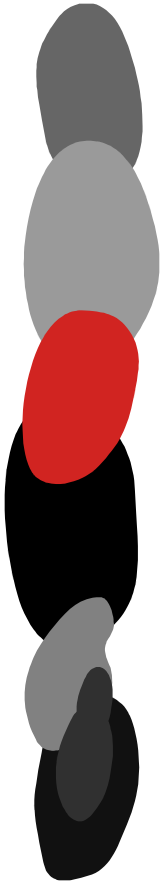
TP = true positive = 18 FN = false negative = 1

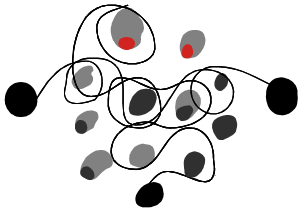
FP = false positive = 5 TN = true negative = 10

Correctly Classified Instances = **TP + TN** = 18 + 10 = 28

Incorrectly Classified Instances = **FN + FP** = 1 + 5 = 6

How to evaluate?





=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	Class
	0.947	0.333	0.783	0.947	0.857	0.811	good
	0.667	0.053	0.909	0.667	0.769	0.811	bad
Weighted Avg.	0.824	0.209	0.838	0.824	0.818	0.811	

=== Confusion Matrix ===

a	**b**	**classified as**
18	1	a = good
5	10	b = bad

Category		classified as	
		YES	**NO**
real	**YES**	TP	FN
	NO	FP	TN

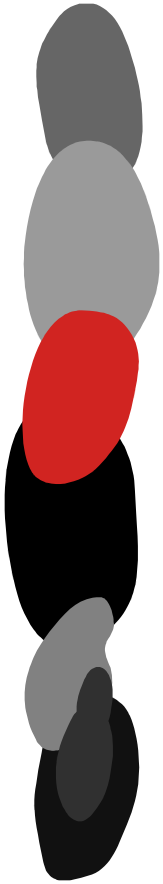
n

TP Rate = TP/(TP+FN) = 18/(18+1) = 0.947 → good

FP Rate = FP/(FP+TN) = 5/(5+10) = 0.333

↓ the probability of falsely rejecting the null hypothesis for a particular test.

How to evaluate?



Category good		classified as	
		YES	**NO**
real	**YES**	18	1
	NO	5	10

n

TP = true positive = 18
FN = false negative = 1
FP = false positive = 5
TN = true negative = 10

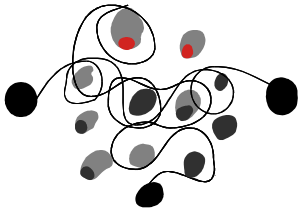
Category bad		classified as	
		YES	**NO**
real	**YES**	10	5
	NO	1	18

n

TP = true positive = 10
FN = false negative = 5
FP = false positive = 1
TN = true negative = 18

Weighted Average:
TP Rate = [TP Rate (good) * 19 + TP Rate (bad) * 15] / 34
FP Rate = [FP Rate (good) * 19 + FP Rate (bad) * 15] / 34

How to evaluate?



Category good		classified as	
		YES	**NO**
real	**YES**	18	1
	NO	5	10

n

Category bad		classified as	
		YES	**NO**
real	**YES**	10	5
	NO	1	18

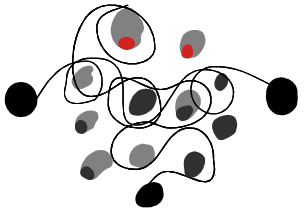
n

TP = true positive = 18
FN = false negative = 1
FP = false positive = 5
TN = true negative = 10

TP = true positive = 10
FN = false negative = 5
FP = false positive = 1
TN = true negative = 18

Precision (good) = [TP / (TP + FP)] = 18 / (18 + 5) = 0,783
Recall (good) = [TP / (TP + FN)] = 18 / (18 + 1) = 0,947
F-measure (good) = [2·Pr·Rc / (Pr + Rc)] = 0,857

How to evaluate?



Multiclass classifier:

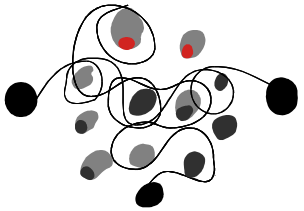
Real → C = {y_1, y_2… y_e}

		Real class	
		C = y_j	C = y_j
Predicted class	C_M = y_j	TP_j	FP_j
	C_M = y_j	FN_j	TN_j

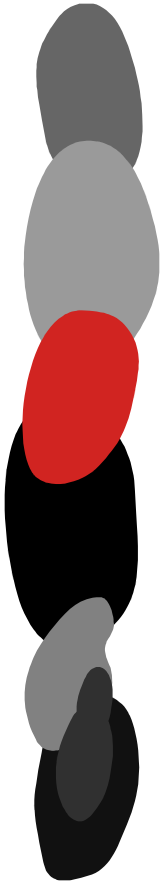
Confusion matrix for class y_j

$$\text{Precision}_j = \frac{TP_j}{TP_j + FP_j} \qquad \text{Recall}_j = \frac{TP_j}{TP_j + FN_j} \qquad F_1 = \frac{2 \times precision \times recall}{precision + recall}$$

How to evaluate?



- Performance metrics:
  - **Precision** (Pr): if example d_j has been classified into category c_i, probability of the decision to be correct
  - **Recall** (Re): if example d_j belongs to category c_i, , probability of being classified that way. “coverage” of the classifier.



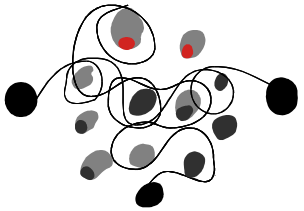
Category Ci		Classifier	
		YES	**NO**
Real	**YES**	TPi	FNi
	NO	FPi	TNi

$$Pr_i = \frac{TP_i}{TP_i + FP_i}$$

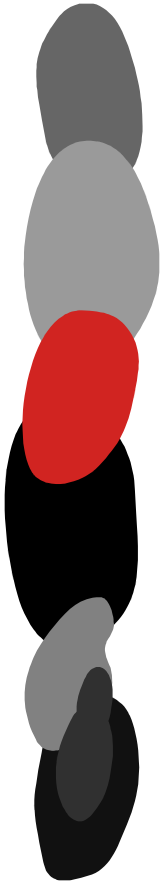
$$Re_i = \frac{TP_i}{TP_i + FN_i}$$

$$\text{F-Measure} = \frac{2 \cdot Pr \cdot Re}{Pr + Re}$$

How to evaluate?



Example: iris.arff NaiveBayes



Time taken to build model: 0.02 seconds

=== Detailed Accuracy By Class ===

TP Rate	FP Rate	Precision	Recall	F-Measure	Class
1	0	1	1	1	Iris-setosa
0.96	0.04	0.923	0.96	0.941	Iris-versicolor
0.92	0.02	0.958	0.92	0.939	Iris-virginica

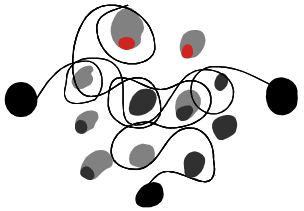
=== Confusion Matrix ===

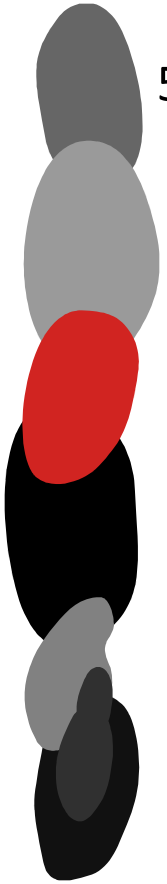
a	b	c	classified as
50	0	0	a = Iris-setosa
0	48	2	b = Iris-versicolor
0	4	46	c = Iris-virginica

All correct

4 incorrect

How to evaluate?





=== Confusion Matrix ===

a	b	c	classified as
50	0	0	a = Iris-setosa
0	48	2	b = Iris-versicolor
0	4	46	c = Iris-virginica

Category Ci		Classified as	
		Yes	**No**
Real	**Yes**	TPi	FNi
	No	FPi	TNi

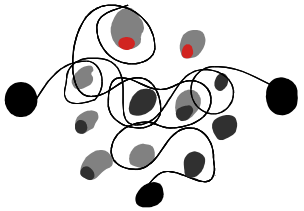
- Iris-setosa

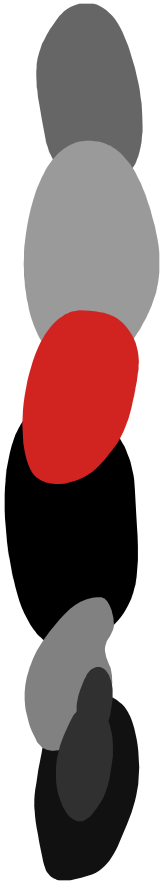
  TP = 50, FP = 0, FN = 0, TN = 48 + 2 + 4 + 46 = 100
- Iris-versicolor

  TP = 48, FP = 4, FN = 2, TN = 50+46 = 96
- Iris-virginica

  TP = 46, FP = 2, FN = 4, TN = 50+48 = 98

How to evaluate?





Microaveraging global addition

$$\widehat{\mathrm{Re}}^{\mu} = \frac{TP}{TP + FN} = \frac{\Sigma_{i=1}^{m} TP_i}{\Sigma_{i=1}^{m} (TP_i + FN_i)}$$

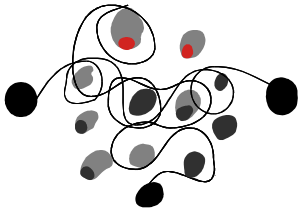
$$\widehat{\mathrm{Pr}}^{\mu} = \frac{TP}{TP + FP} = \frac{\Sigma_{i=1}^{m} TP_i}{\Sigma_{i=1}^{m} (TP_i + FP_i)}$$

Macroaveraging: precision and recall are locally calculated for each category and then added

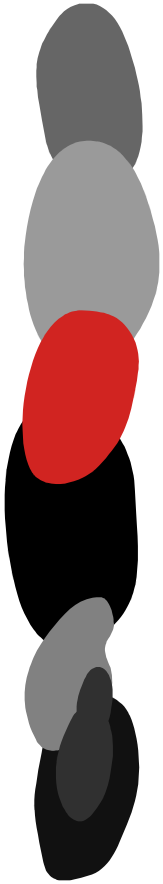
$$\widehat{\mathrm{Pr}}^{M} = \frac{\Sigma_{i=1}^{m} \widehat{\mathrm{Pr}}_i}{m} \qquad \widehat{\mathrm{Re}}^{M} = \frac{\Sigma_{i=1}^{m} \widehat{\mathrm{Re}}_i}{m}$$

Note: good classifiers will obtain good microaverage values for categories with many test instances

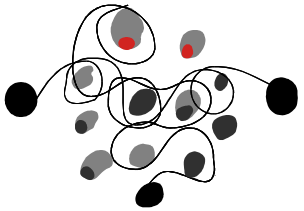
How to evaluate?



- What is more interesting, high Precision (Pr) or high Recall (Re)?
  - Both
  - Medicine: does she/he have cancer?
    - When we diagnose it as positive to be true (Pr): precision
    - To diagnose cancer when it exists (Rc): recall
  - Classify texts from newspapers
    - The classifier is not Used on its own. The classifier helps but final classification is manual
  - Web page classification in search engines
    - Wrong classifications are not that important
  - ...



Assignment



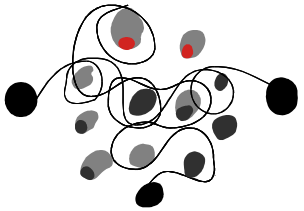

```
=== Confusion Matrix ===

   a   b   c   <-- classified as
   5   0   0 |   a = soft
   0   3   1 |   b = hard
   1   2 12 |   c = none
```

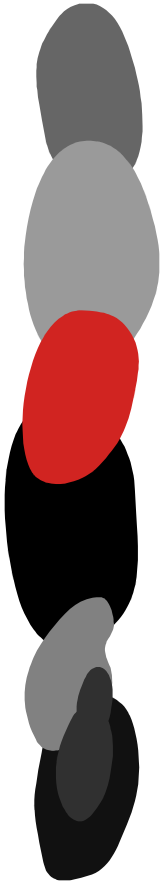
-Which is the most trustful category?

-If we take randomly an example of each category which will the one correctly classified with higher probability?

Class	TP	FP	FN	TN	Pr	Rc	F-Measure
soft							
hard							
none							
				Micro			
				Macro			

Classifier





Lazy: based on instances (knn)
IB1, IBk

Bayes: based on Bayes theorem
NaiveBayes

Trees: based on trees
J48, RandomForest

Rules:
PART, OneR

Functions: based on linear functions
MultilayerPerceptron,
SMO, Winnow

Meta: classifier combination
AdaBoostM1, vote, stacking

Similarity function

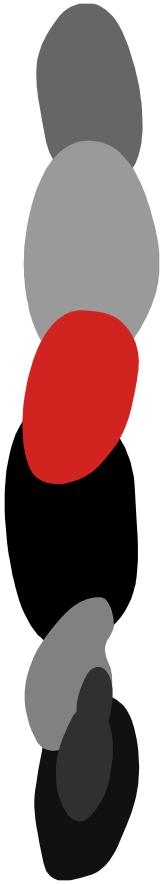
Measuring probabilities

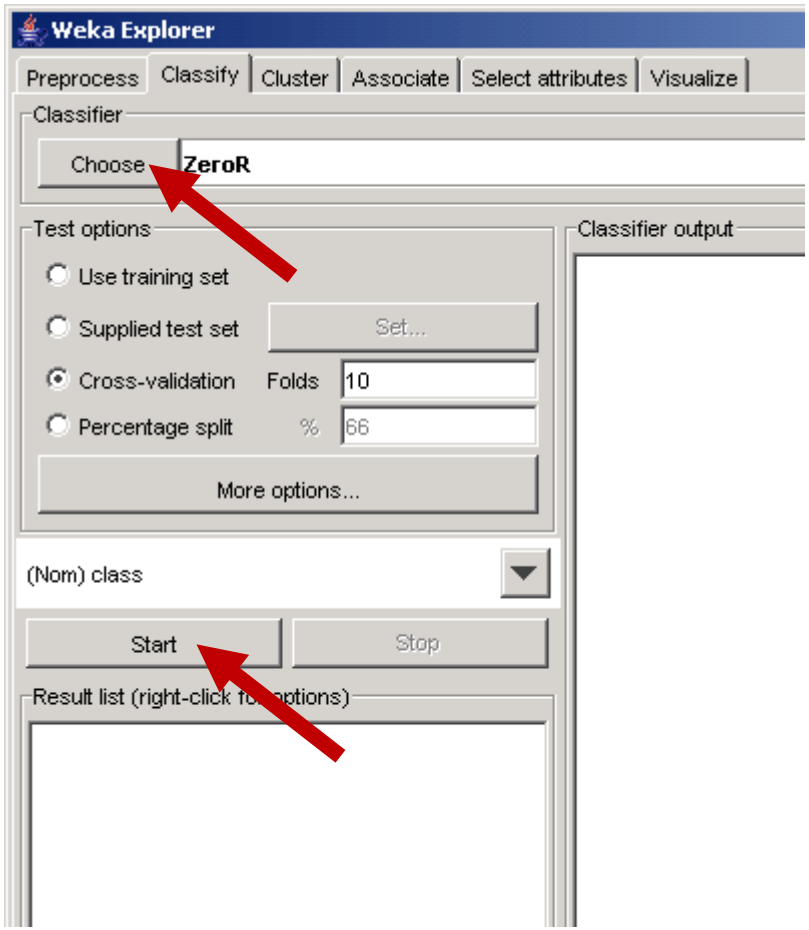
Typical methods of Artificial Intelligence

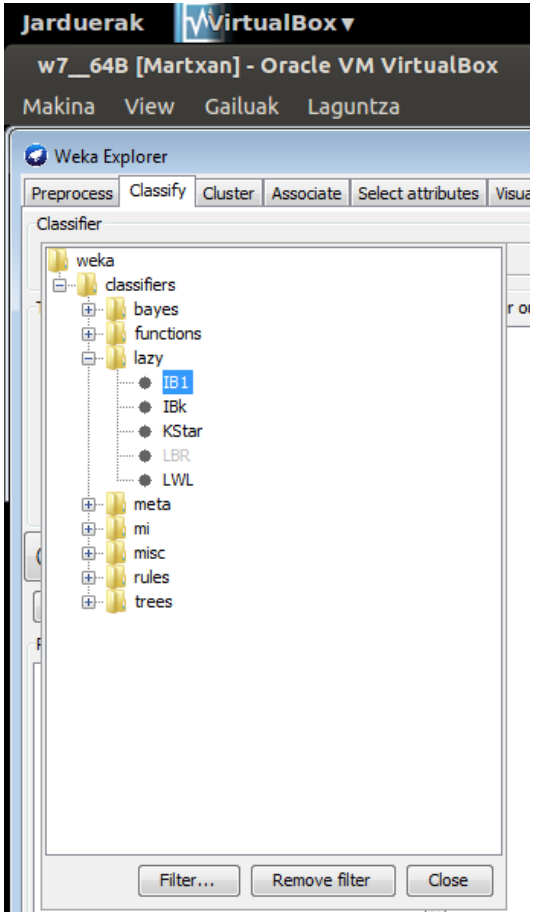
Calculating functions

Combination of simple classifiers

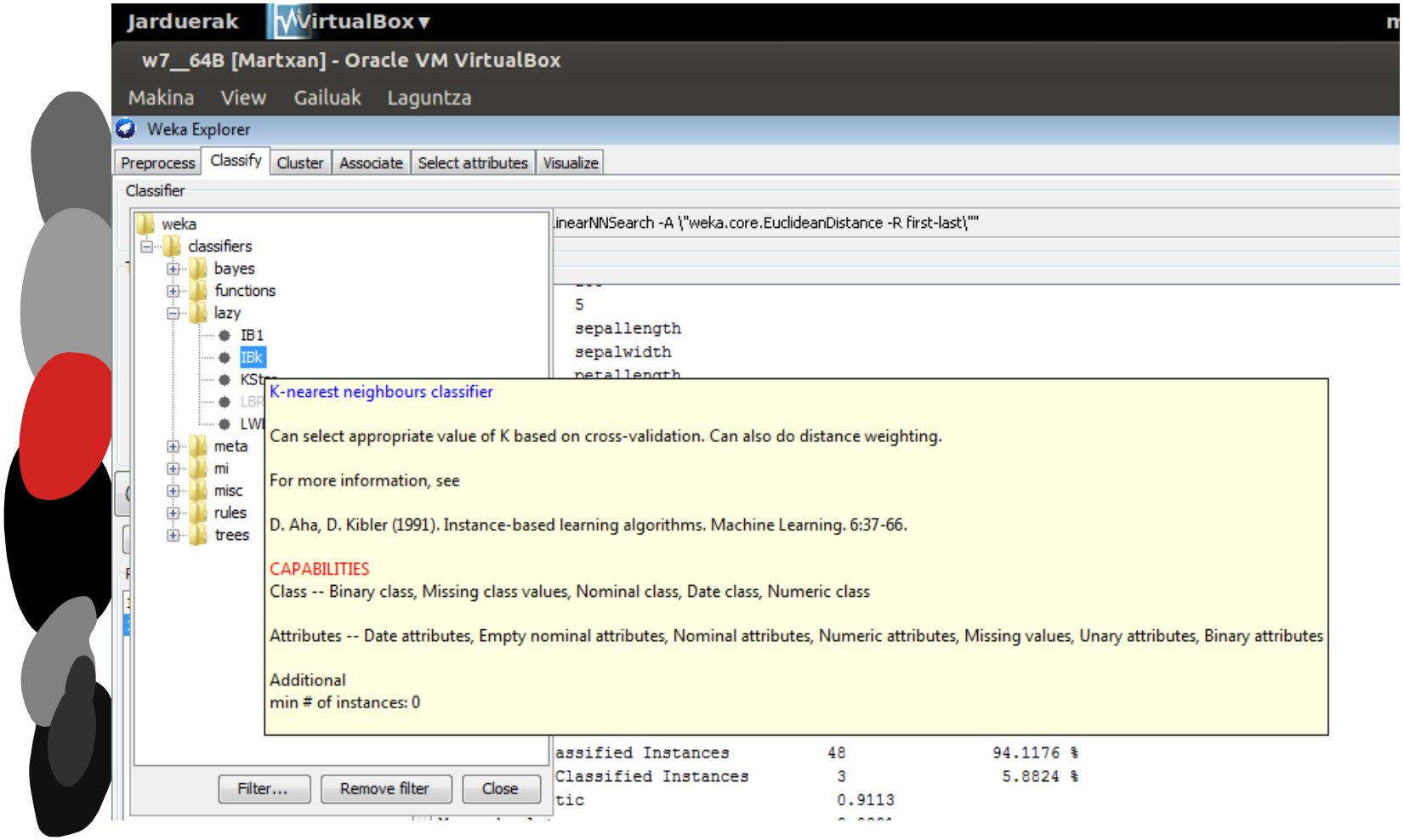
Classifier



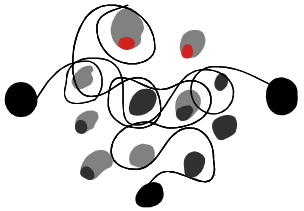

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Classifier
Choose
ZeroR
Test options
Use training set
Supplied test set
Set...
Cross-validation
Folds
10
Percentage split
%
66
More options...
Classifier output
(Nom) class
Start
Stop
Result list (right-click fo ptions)


Jarduerak
VirtualBox
w7_64B [Martxan] - Oracle VM VirtualBox
Makina
View
Gailuak
Laguntza
Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Classifier
weka
classifiers
bayes
functions
lazy
IB1
IBk
KStar
LBR
LWL
meta
mi
misc
rules
trees
Filter...
Remove filter
Close

Classifier


Jarduerak
VirtualBox
w7_64B [Martxan] - Oracle VM VirtualBox
Makina View Gailuak Laguntza
Weka Explorer
Preprocess Classify Cluster Associate Select attributes Visualize
Classifier
weka
classifiers
bayes
functions
lazy
IB1
IBk
meta
mi
misc
rules
trees
inearNNSearch -A \"weka.core.EuclideanDistance -R first-last\""
5
sepallength
sepalwidth
K-nearest neighbours classifier
Can select appropriate value of K based on cross-validation. Can also do distance weighting.
For more information, see
D. Aha, D. Kibler (1991). Instance-based learning algorithms. Machine Learning. 6:37-66.
CAPABILITIES
Class -- Binary class, Missing class values, Nominal class, Date class, Numeric class
Attributes -- Date attributes, Empty nominal attributes, Nominal attributes, Numeric attributes, Missing values, Unary attributes, Binary attributes
Additional
min # of instances: 0
assified Instances 48 94.1176 %
Classified Instances 3 5.8824 %
tic 0.9113
Filter...
Remove filter
Close

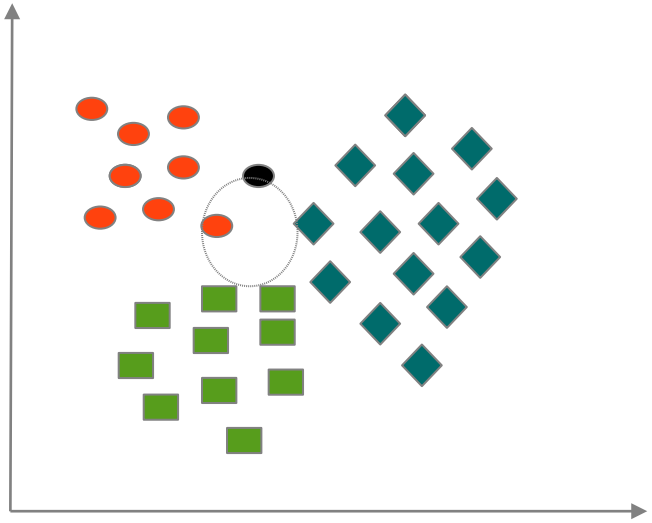
K-NN classifier



k Nearest Neigbour (**k**-NN)

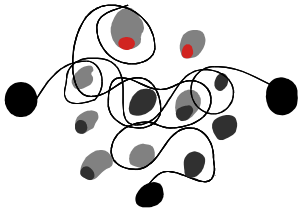
k-NN is a **lazy** classifier. The classifier is based on the learning instances stored in memory, there is not model of the categories built

1-NN The class of the new instance to be classified will be the class of its nearest neighbour in the reference pattern set (RPS)

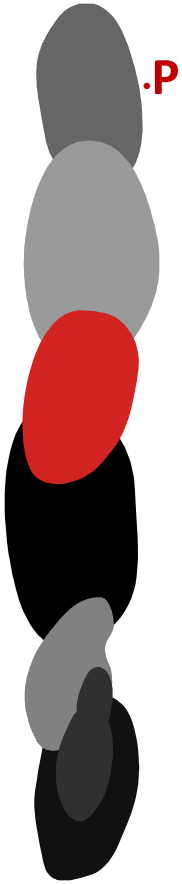


- Instances of the RPS
  - Class 1
  - Class 2
  - Class 3

K-NN classifier



k Nearest Neighbour (**k**-NN): the majority class within the K nearest instances in the RPS is chosen



.**Procedure** to classify new instances

.1.- Calculate the **distance** between the instance to classify and all the instances in the RPS
For example Euclidean distance (n dimensions)

.2.- Select the **k nearest** instances (smallest distance)

.3.- Assign as class the **majority class** within the k instances

Main parameters

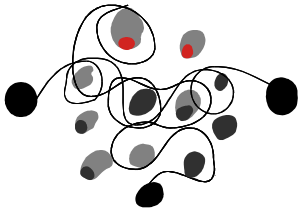
Value of **k, number of examples** used to classify

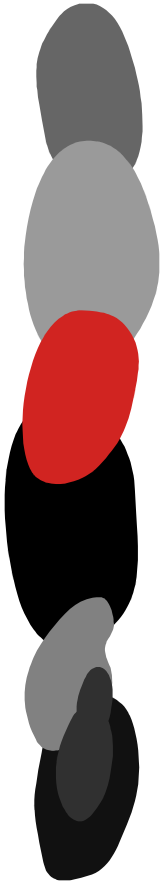
Distance or similarity measure used to compare instances

Criteria **to select** the k nearest instances

Criteria **to decide** the class of the new instances

k-NN. Example





Example (text classification) **category**

d_1: ill(3), player, doctor(5), health(2) **health**

d_2: nurse, play(3), doctor(4), health(2) **health**

d_3: player(4), play(2), doctor(2), ball(3) **sport**

d_t: **ill(3), play(4), doctor(2), health** **???**

dictionary
Nurse
Ill
Player
Play
Doctor
Health
Ball

$$|d_j, d_z| = \sqrt{\sum_{i=1}^{n} (w_{ji} - w_{zi})^2}$$

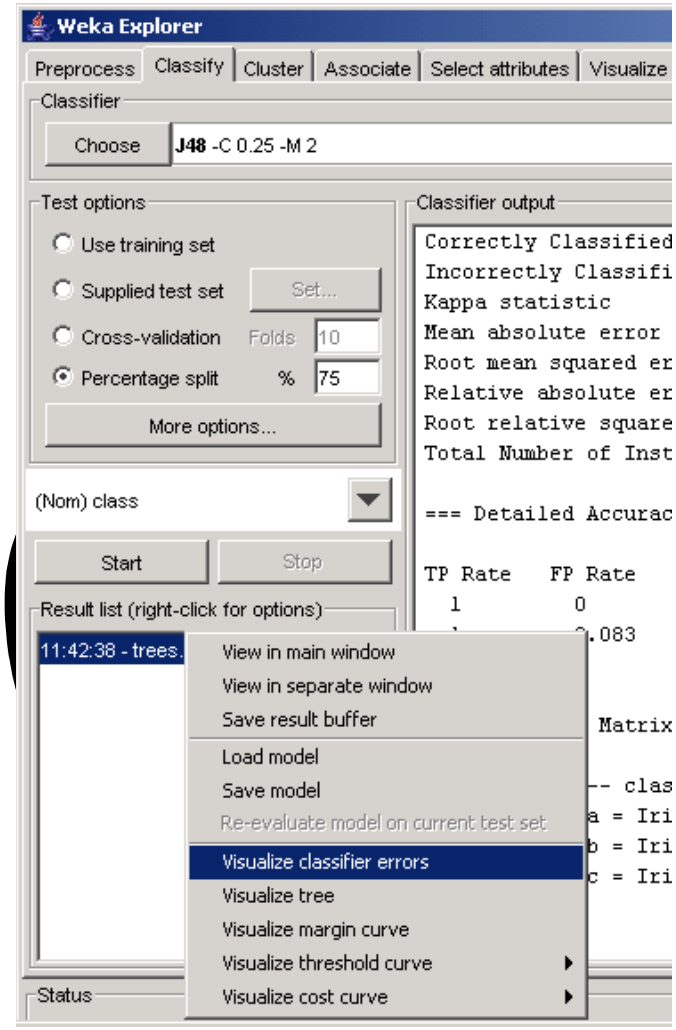
d_1: 0 3 1 0 5 2 0 $|d_1-d_t| = \sqrt{0^2+0^2+1^2+4^2+3^2+1^2+0^2} = 5{,}19$

d_2: 1 0 0 3 4 2 0 $|d_2-d_t| = 4$

d_3: 0 0 4 2 2 0 3 $|d_3-d_t| = 6{,}24$

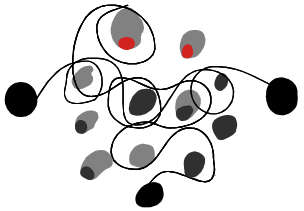
d_t: **0 3 0 4 2 1 0**

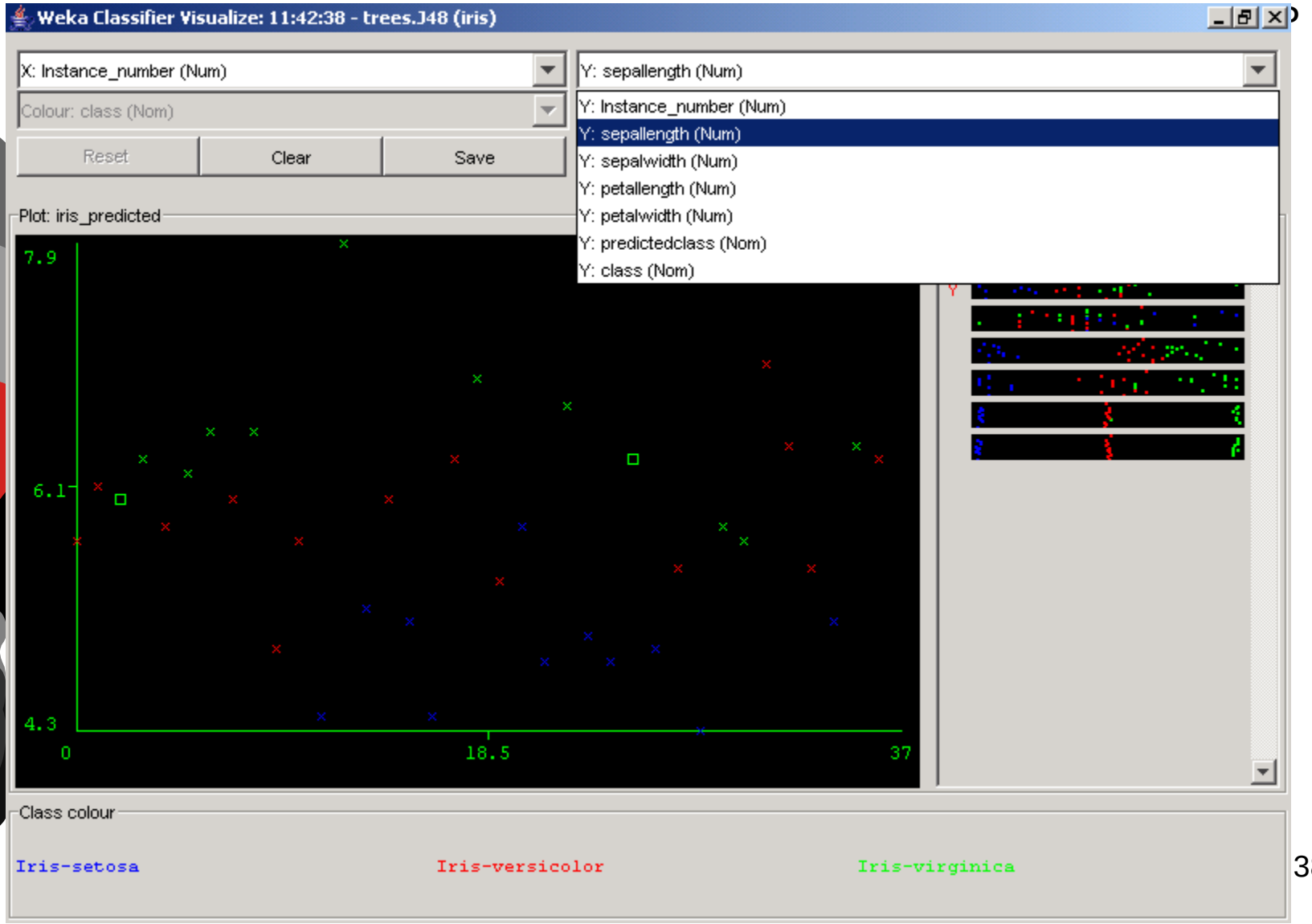
Visualize classifier errors



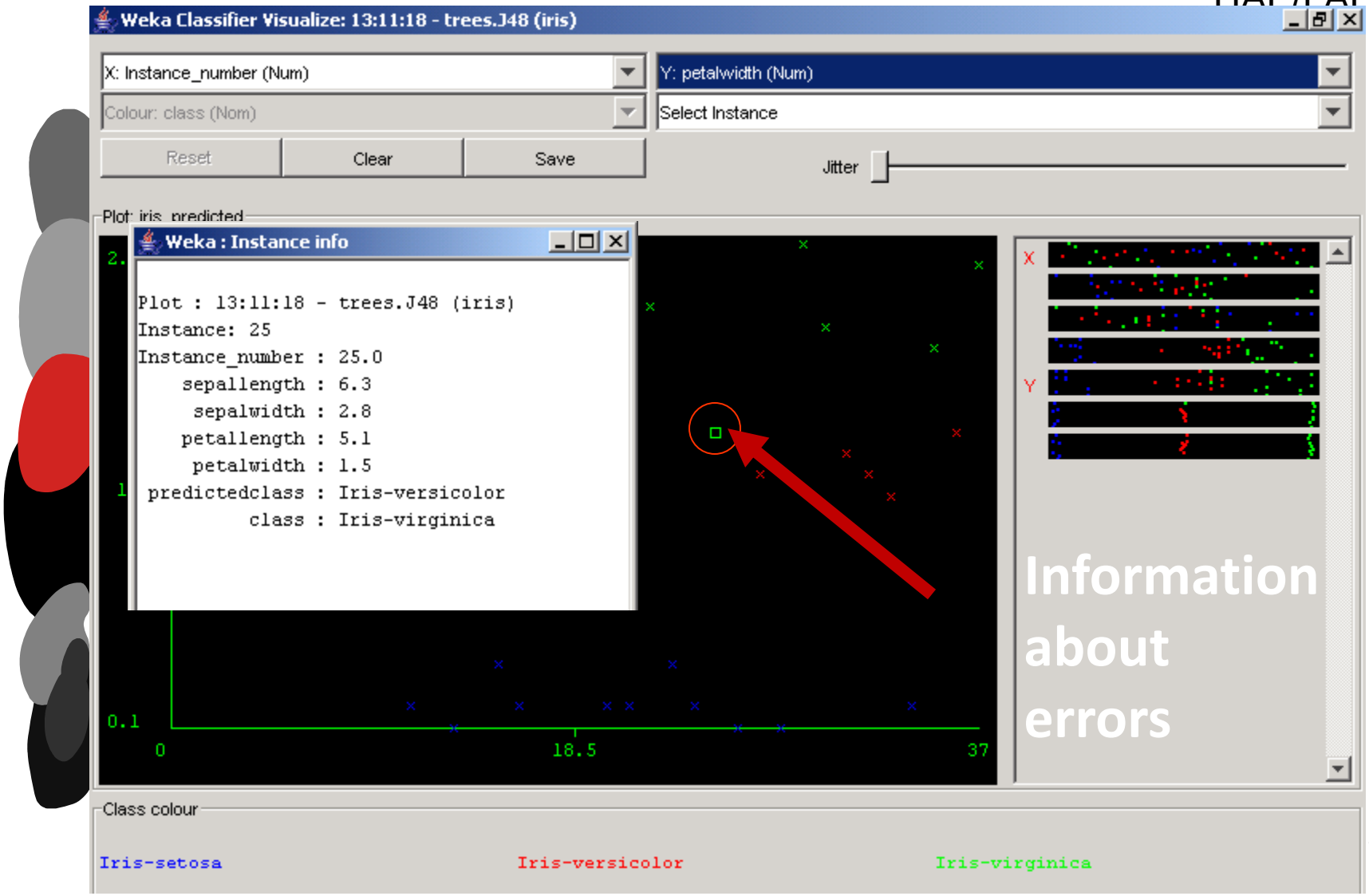
- Instance distribution in the space.
- It is possible to select attributes in axes X and Y.
- Information about errors

Visualize classifier errors

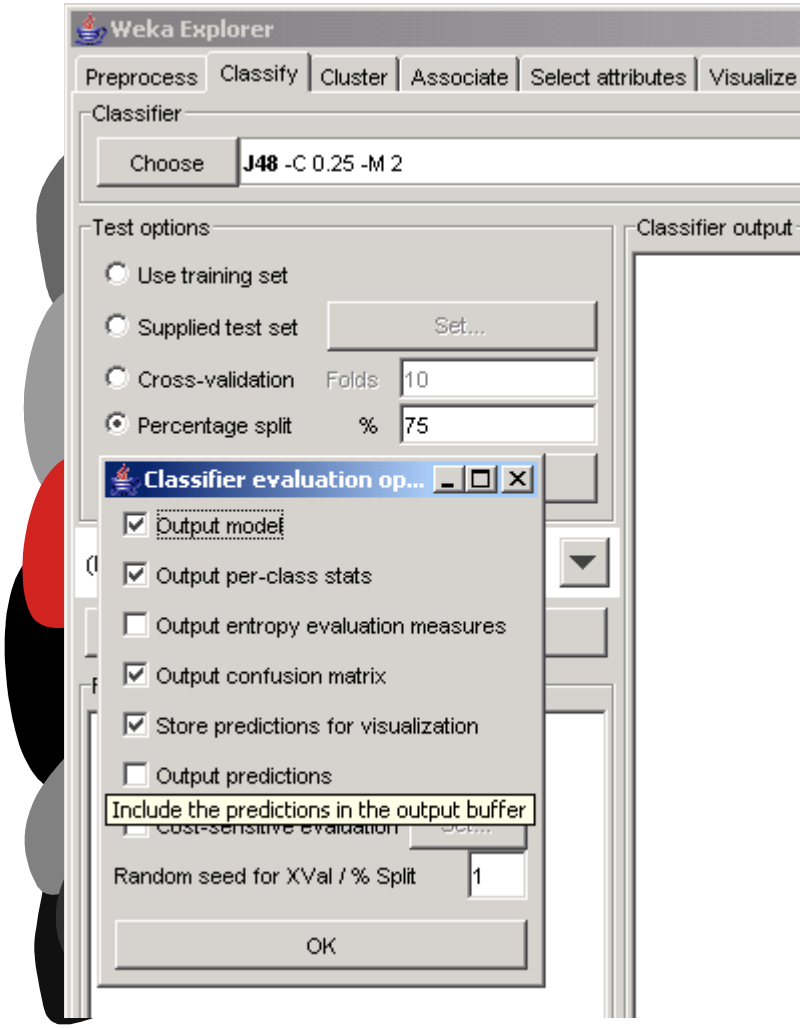




Visualize classifier errors



More Options: Output predictions



instance

class

prediction

probability

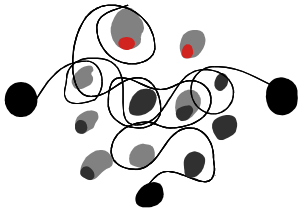
error

HAP/LAP

Classifier output

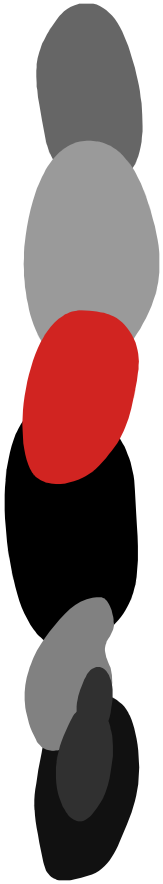
```
=== Predictions on test split ===

 inst#,     actual, predicted, error, probability distribution
     1 2:Iris-ver 2:Iris-ver           0      *0.972  0.028
     2 2:Iris-ver 2:Iris-ver           0      *0.972  0.028
     3 3:Iris-vir 2:Iris-ver     +     0      *0.972  0.028
     4 3:Iris-vir 3:Iris-vir           0       0.029 *0.971
     5 2:Iris-ver 2:Iris-ver           0      *0.972  0.028
     6 3:Iris-vir 3:Iris-vir           0       0.029 *0.971
     7 3:Iris-vir 3:Iris-vir           0       0.029 *0.971
     8 2:Iris-ver 2:Iris-ver           0      *0.972  0.028
     9 3:Iris-vir 3:Iris-vir           0       0.029 *0.971
    10 2:Iris-ver 2:Iris-ver           0      *0.972  0.028
    11 2:Iris-ver 2:Iris-ver           0      *0.972  0.028
    12 1:Iris-set 1:Iris-set          *1       0       0
    13 3:Iris-vir 3:Iris-vir           0       0.029 *0.971
    14 1:Iris-set 1:Iris-set          *1       0       0
    15 2:Iris-ver 2:Iris-ver           0      *0.972  0.028
    16 1:Iris-set 1:Iris-set          *1       0       0
    17 1:Iris-set 1:Iris-set          *1       0       0
    18 2:Iris-ver 2:Iris-ver           0      *0.972  0.028
    19 3:Iris-vir 3:Iris-vir           0       0.029 *0.971
    20 2:Iris-ver 2:Iris-ver           0      *0.972  0.028
    21 1:Iris-set 1:Iris-set          *1       0       0
```

k-NN. Example



Iris.arff **numeric**

Soybean.arff **nominal**

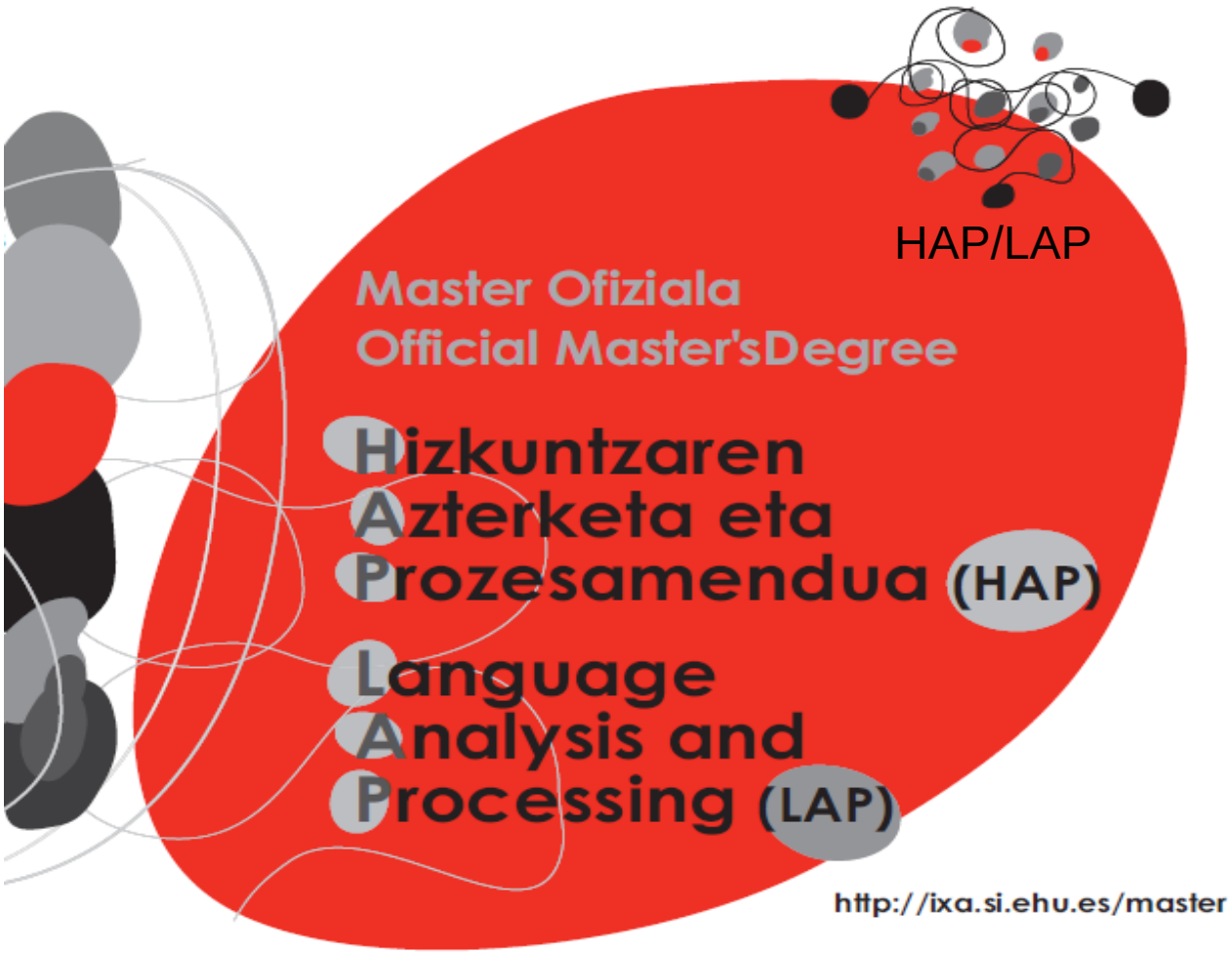


	k = 1	k = 3	k =5
iris.arff			
soybean.arrf			

Select the best (k>1) and apply distanceWeighting

	1/distance	1-distance
iris.arff	(k=)	(k=)
soybean.arff	(k=)	(k=)

Erasmus Mundus



Language and Communication Technologies

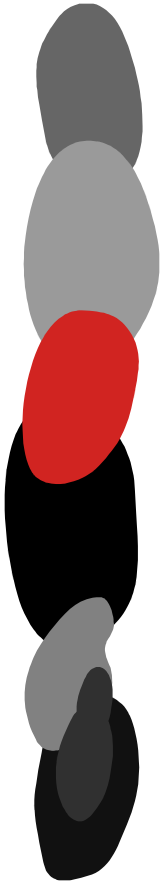
ixa

ueu udako euskal unibertsitatea

Universidad del País Vasco Euskal Herriko Unibertsitatea | MDe Master eta Doktorego Eskola Escuela de Máster y Doctorado Master and Doctoral School

Lct

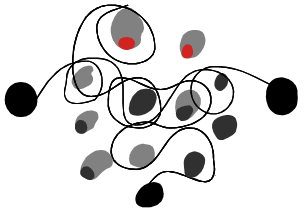
Erasmus Mundus



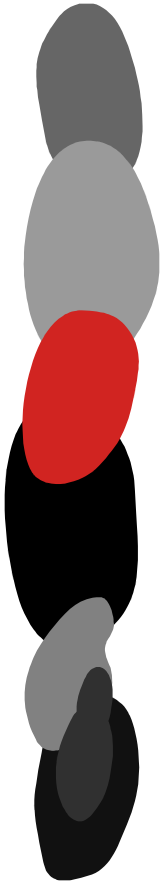
Statistical methods and text corpora

Introduction to Machine Learning for Natural Language Processing

Olatz Arbelaitz: olatz.arbelaitz@ehu.eus



Topics

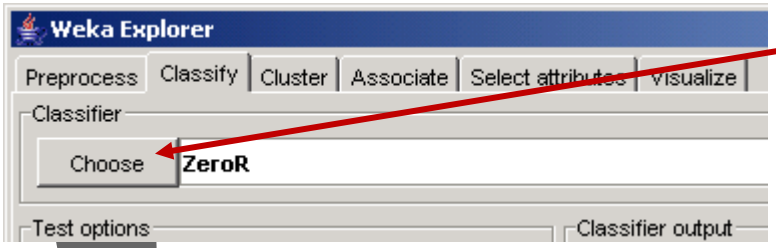


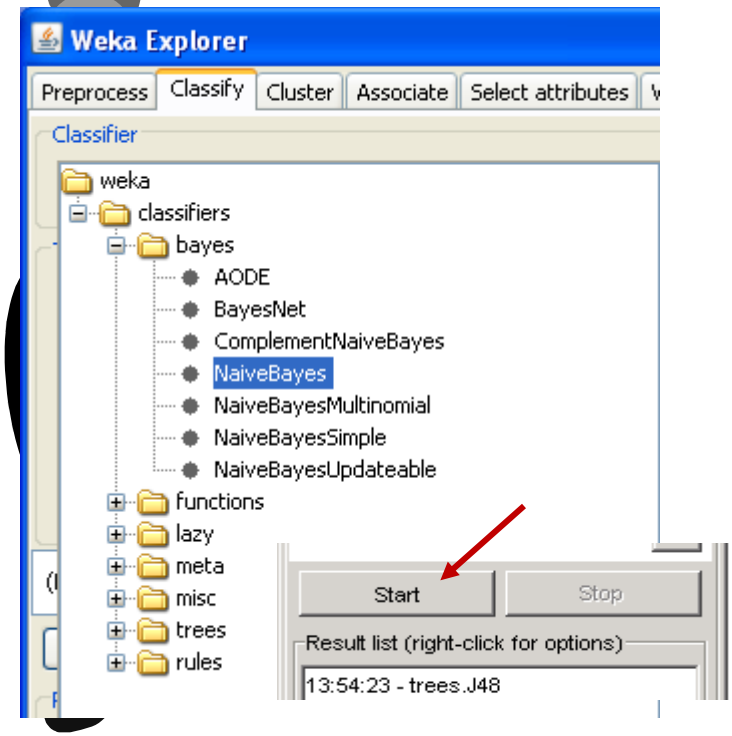
1.- Machine Learning in LNP

2.- WEKA software:

- 2.1.-Attribute (feature) selection
- 2.3.- **Basic algorithms: Naive Bayes, K-NN, Decision Trees, Rules, …**
- 2.3.-Evaluation

Types of classifiers





Lazy: based on instances (knn)

IB1, IBk

Bayes: based on Bayes theorem

NaiveBayes

Trees: based on trees

J48, NBTree, RandomForest

Rules:

PART, OneR

Functions: based on linear functions

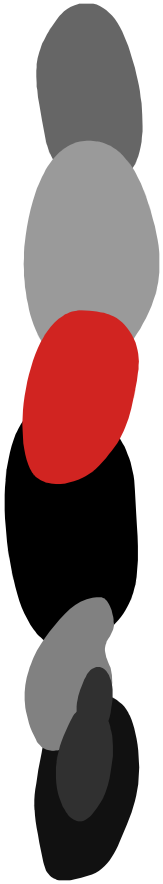
MultilayerPerceptron,
SMO, Winnow

Meta: classifier combination

AdaBoostM1, vote, stacking

HAP/LAP

Naive Bayes



Classifier based on Bayes Theorem making calculations simpler when:

The database has many features

There are not enough examples to calculate probabilities of all feature combinations.

Revision:

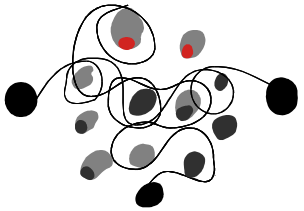
P(wi): prior probability of class wi
(quantifies the probability of a class without any extra information)

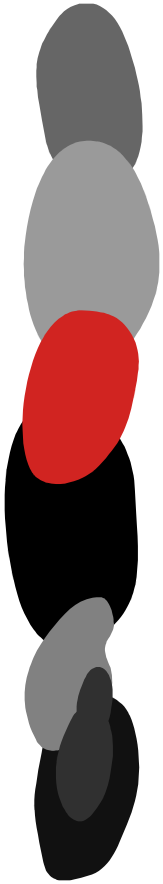
P(x|wi): density function probability conditioned to the class
(quantifies the probability of **x** having a concrete value knowing the class it belongs to)

P(wi|x): **posterior probability**
(quantifies the probability of an instance to belong to a class)

p(x): probability of instance x (unconditional)
(distribution of the instances)

Naive Bayes





Bayes theorem

$0 <= P(w_i) <= 1$ $\quad \sum_{(i=1,\ldots,c)} P(w_i) = 1$ $\quad \sum_{(i=1,\ldots,c)} P(x|w_i) = 1$

Given the prior probabilities and the density functions the **posterior probability** can be calculated **(≈classification):**

$$P(\omega_i|x) = \frac{P(x|\omega_i) * P(\omega_i)}{P(x)}$$

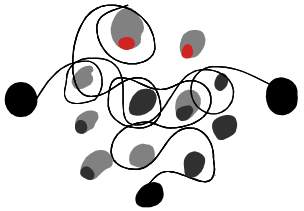
where

$$P(x) = \sum_{i=1}^{c} P(x|\omega_i) * P(\omega_i)$$

Classification with Naive Bayes

$$w_{NB} = \arg\max_{w_i \in C} P(w_i) \prod_{k=1..F} P(x_f \mid w_i)$$

Naive Bayes

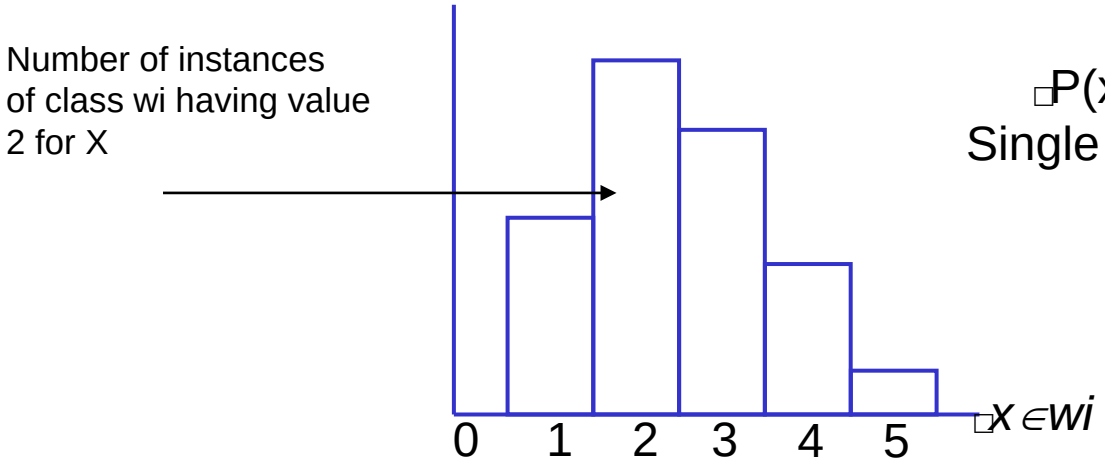


- Features are supposed to be **independent**

  As a consequence the probability for many features can be calculated as follows

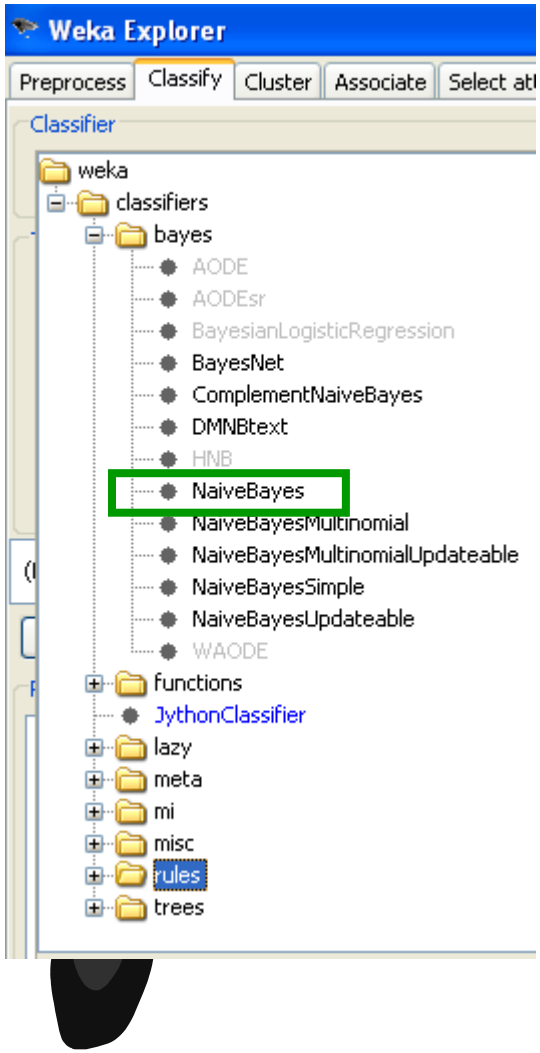
- $P(x1,\ldots,xF|wi) = P(x1|wi)\cdot\ldots\cdot P(xF|wi)$

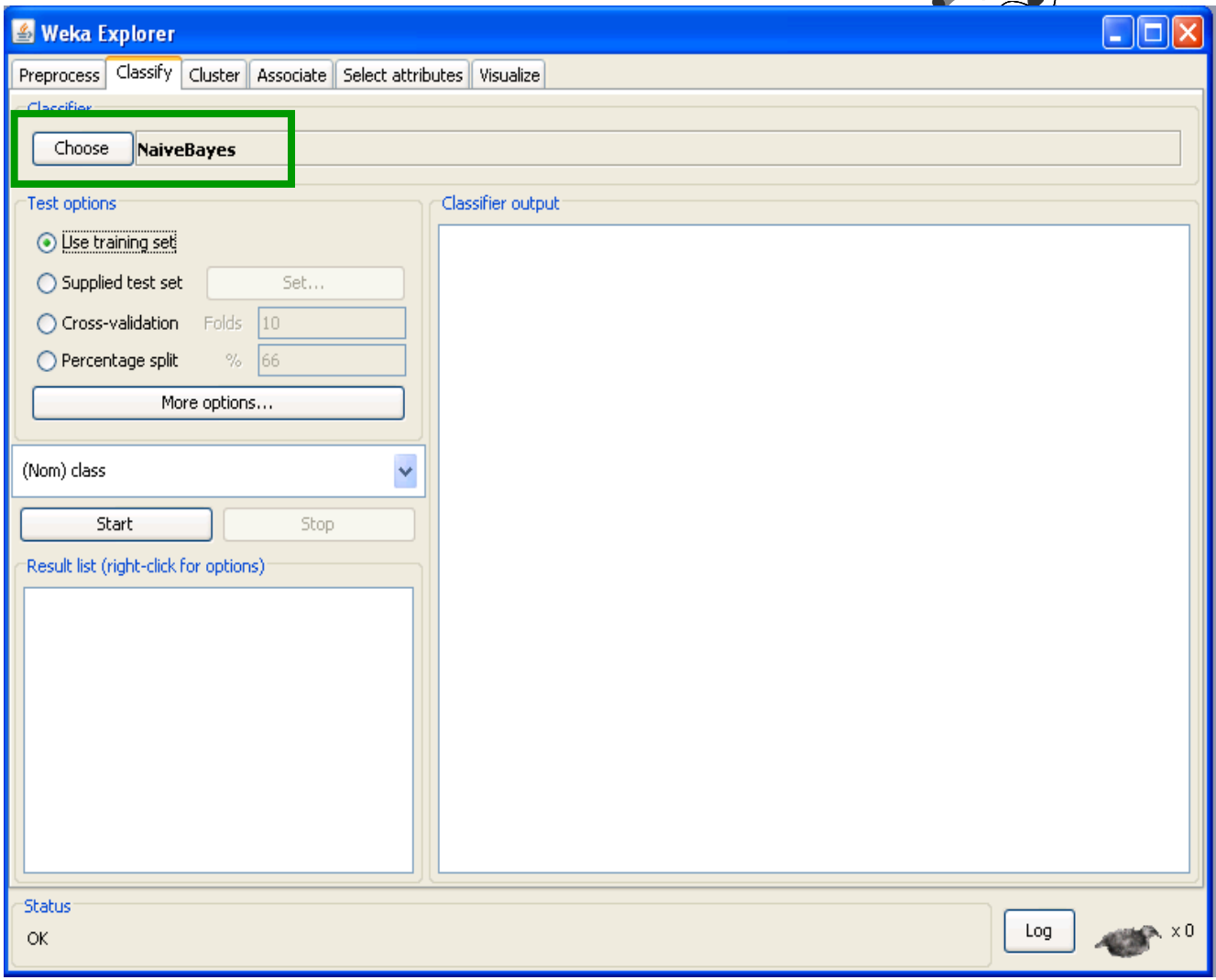
- If xf is **discrete** P(xf|wi) is estimated based on the relative instance frequency of class wi taking value xf (mass function, histograms)



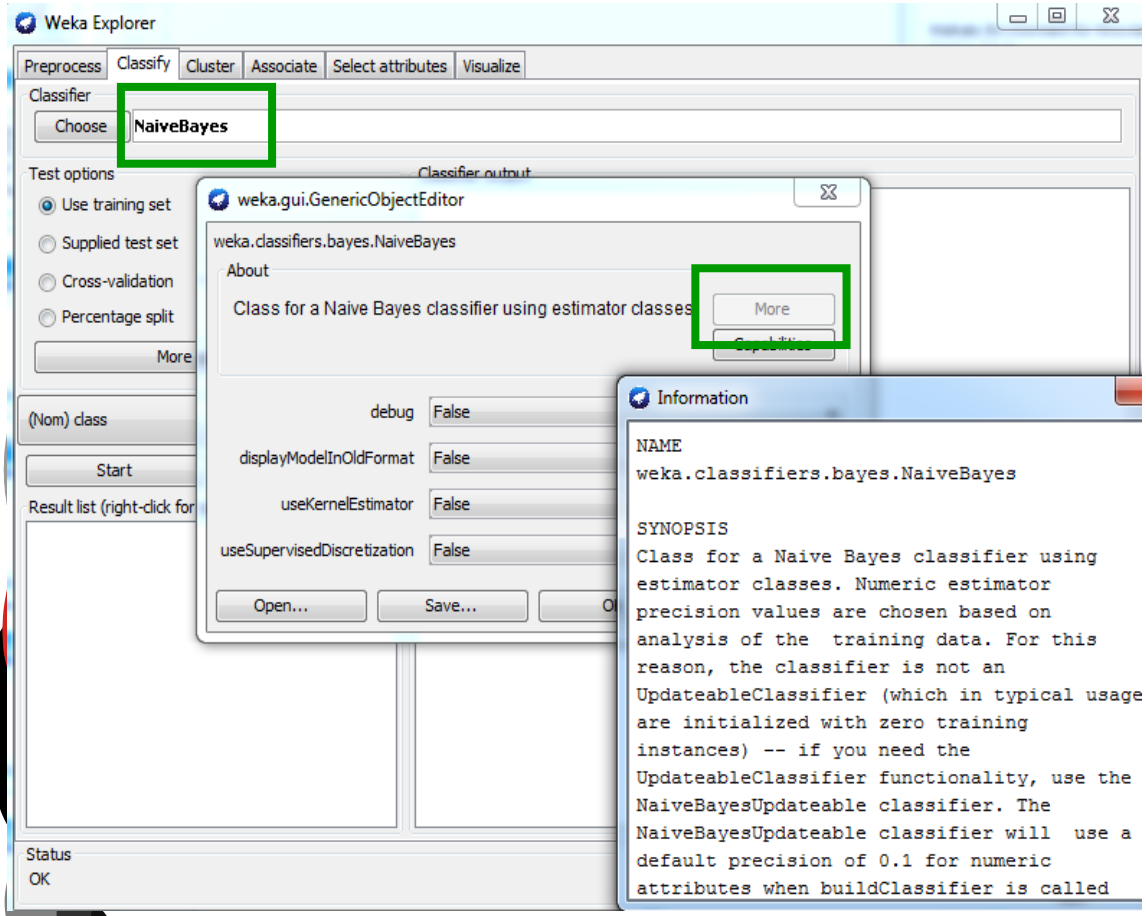
- If xf is **continuous**
  1. Discretize and treat it as discrete
  2. estimate P(xf|wi) assuming a gaussian (normal) distribution- (only mean and variance required)

Naive Bayes


Weka Explorer
Preprocess
Classify
Cluster
Associate
Select at
Classifier
weka
classifiers
bayes
AODE
AODEsr
BayesianLogisticRegression
BayesNet
ComplementNaiveBayes
DMNBtext
HNB
NaiveBayes
NaiveBayesMultinomial
NaiveBayesMultinomialUpdateable
NaiveBayesSimple
NaiveBayesUpdateable
WAODE
functions
JythonClassifier
lazy
meta
mi
misc
rules
trees


Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Classifier
Choose
NaiveBayes
Test options
Use training set
Supplied test set
Set...
Cross-validation
Folds
10
Percentage split
%
66
More options...
Classifier output
(Nom) class
Start
Stop
Result list (right-click for options)
Status
OK
Log
x 0

Naive Bayes



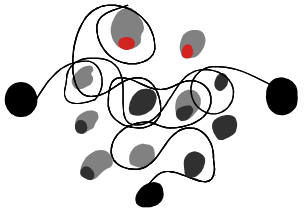
HAP/LAP

To see the concrete **Parameters** of a classifier and a short description about them click in the name of the classifier in bold and then click the "*More*" button

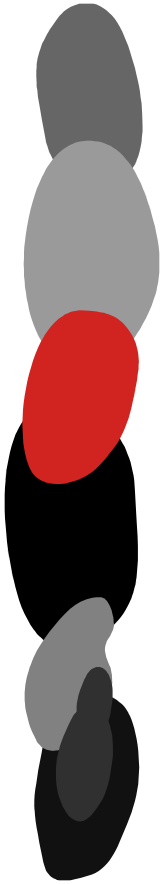
Example: document classification(spam)

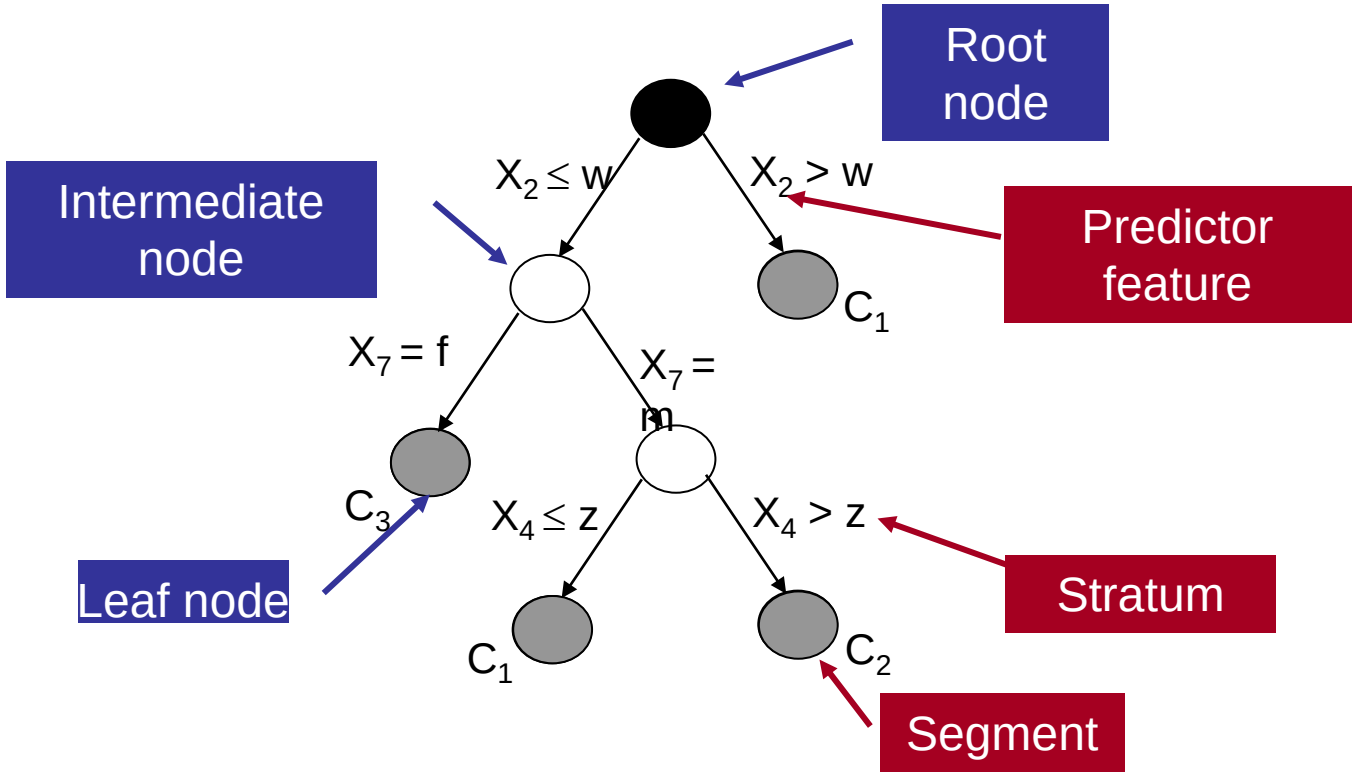
http://en.wikipedia.org/wiki/Naive_Bayes_classifier#Document_Classification

Decision trees

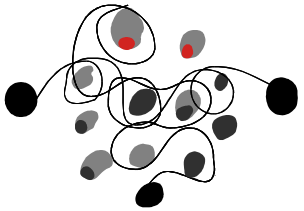


- Based on "Divide and conquer" algorithm
- A classifier in the form of a tree structure
  - Decision node: specifies a test on a single attribute
  - Leaf node: indicates the value of the target attribute
  - Arc/edge: split of one attribute
  - Path: a disjunction of test to make the final decision

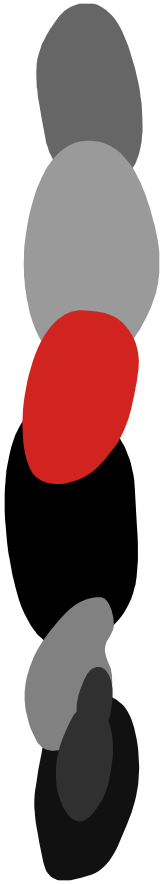


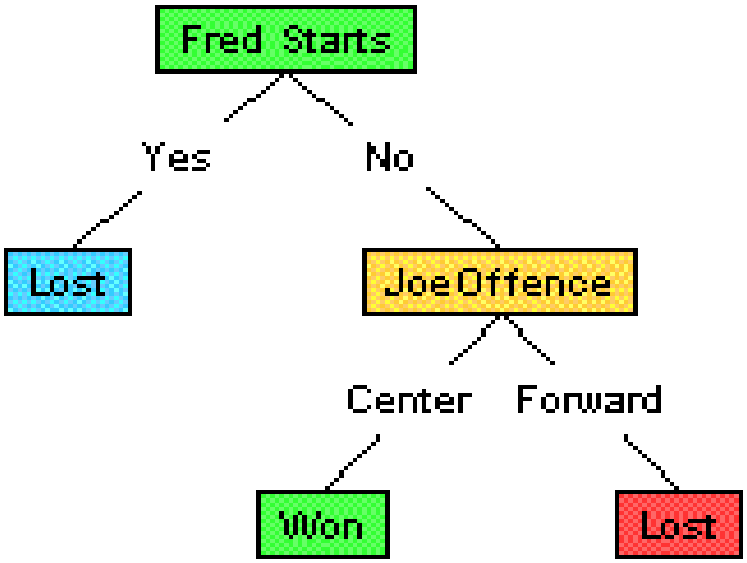


Decision trees



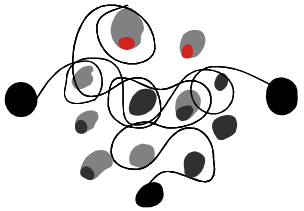
- Decision trees classify instances or examples by starting at the root of the tree and moving through it until a leaf node.

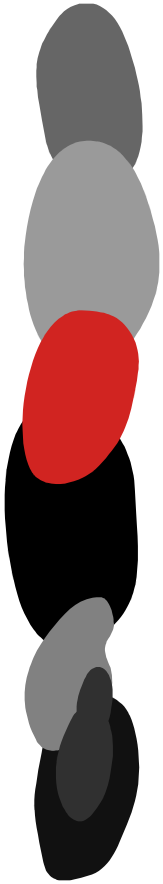




- Problem: Overfitting (good in learning, worse in generalization)

Decision trees

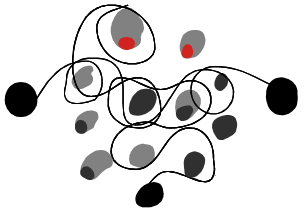


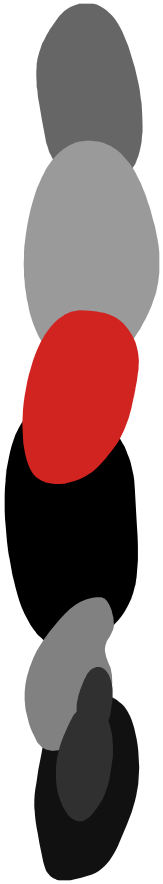


Input: Training set E (labelled instances)
Output: Decision tree (T)
Algorithm

```
begin
  If all the examples in E are of the same category Cj
    then Result simple node labelled as Cj
  else
    begin
    Select a feature Xi with values xi1,...,xil
     Partition E in E1,...,El according to the values of Xi
   Build subtrees T1,...,Tl forE1,...,El
   The result is a tree with root Xi and subtrees T1,...,Tl
   The branches between Xi and the subtrees are labelled with xi1,...,xil
  end
end
```

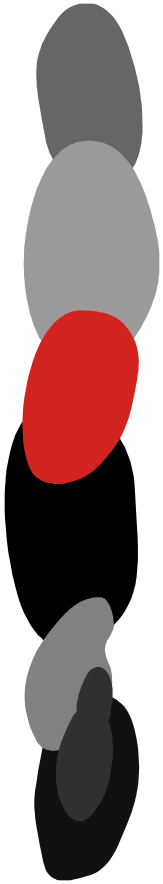
Decision trees

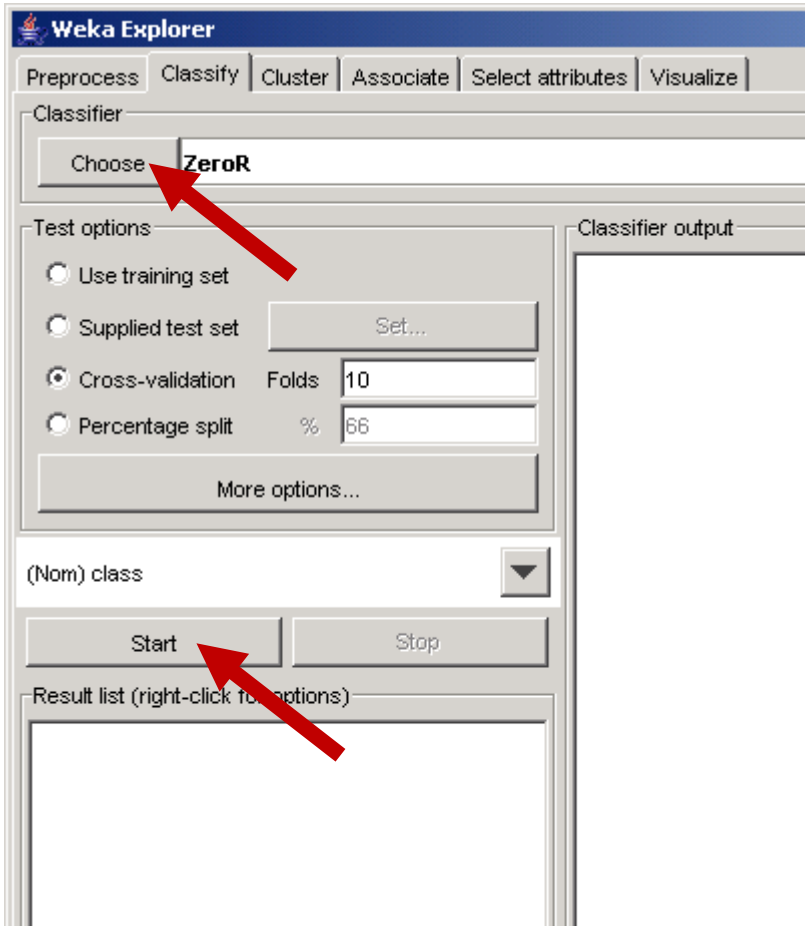


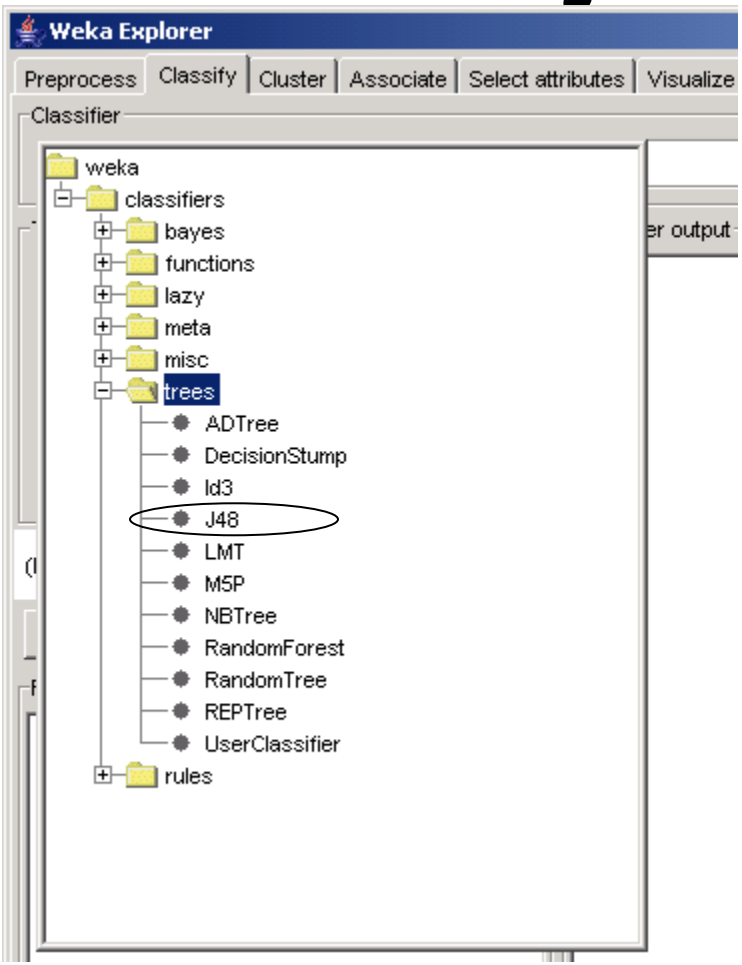


- **At each node:** selection of an attribute to split- choosing the most useful attribute for classifying examples.
  - Nominal features: as many branches as values
  - Numeric features: ≤, > // >, = , < // <, segment, >
- Example: information gain
  - measures how well a given attribute separates the training examples according to their target classification
  - At each node, choose to divide the attribute with the largest information gain
- Stopping rule
  - Every attribute has already been included along this path through the tree, or
  - The training examples associated with this leaf node all have the same target attribute value (i.e., their entropy is zero).

Classifier







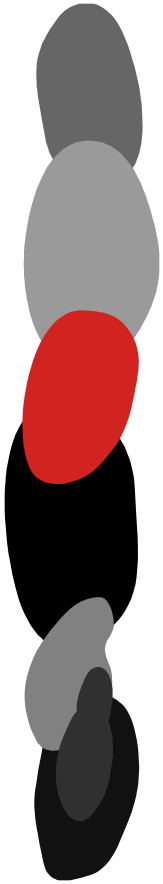
Implementation of C4.5: J48

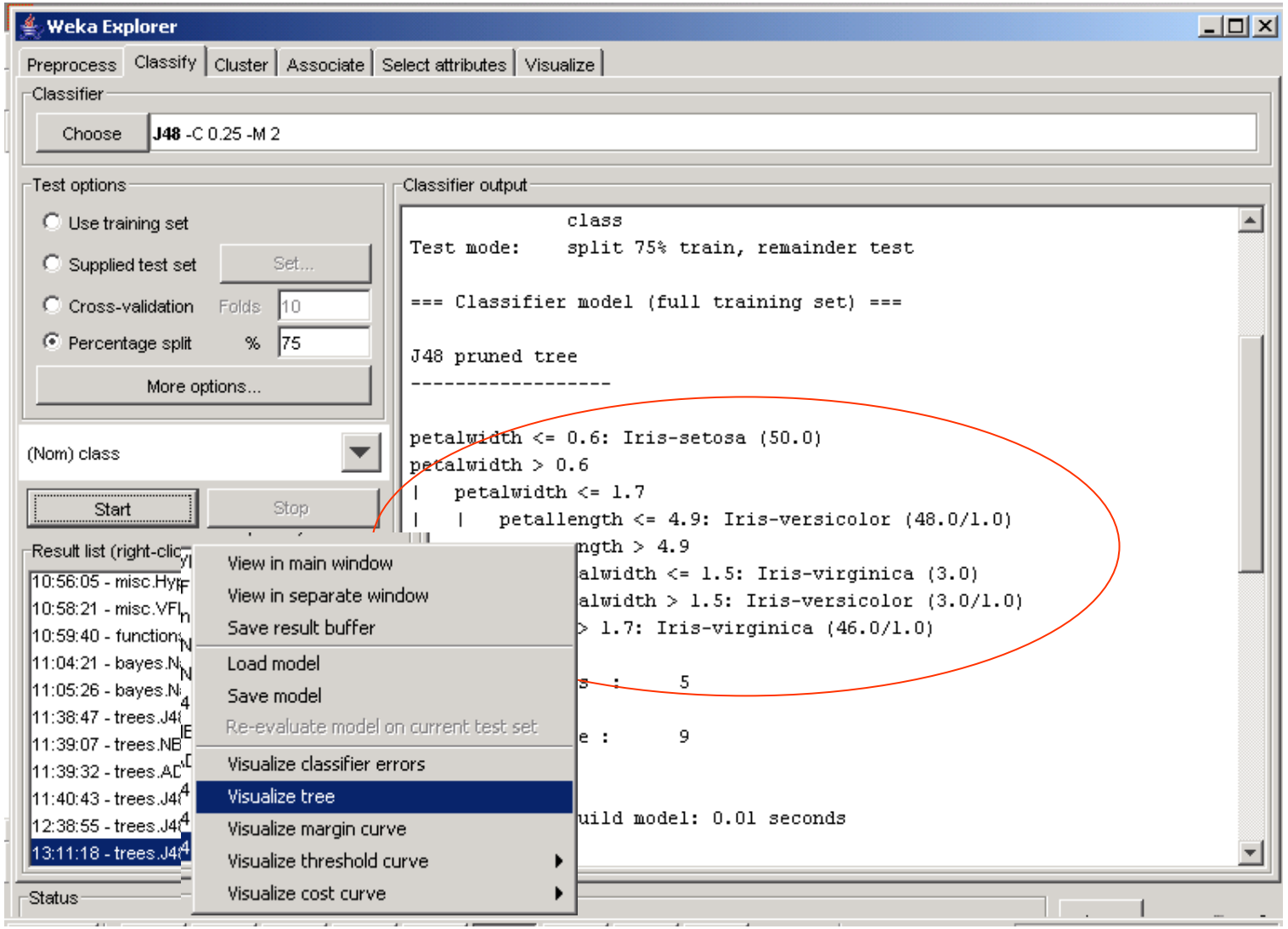
Two branches in numeric attributes (≤, >) see *iris.arff*

With nominal attributes: every value seen*soybean.arff*

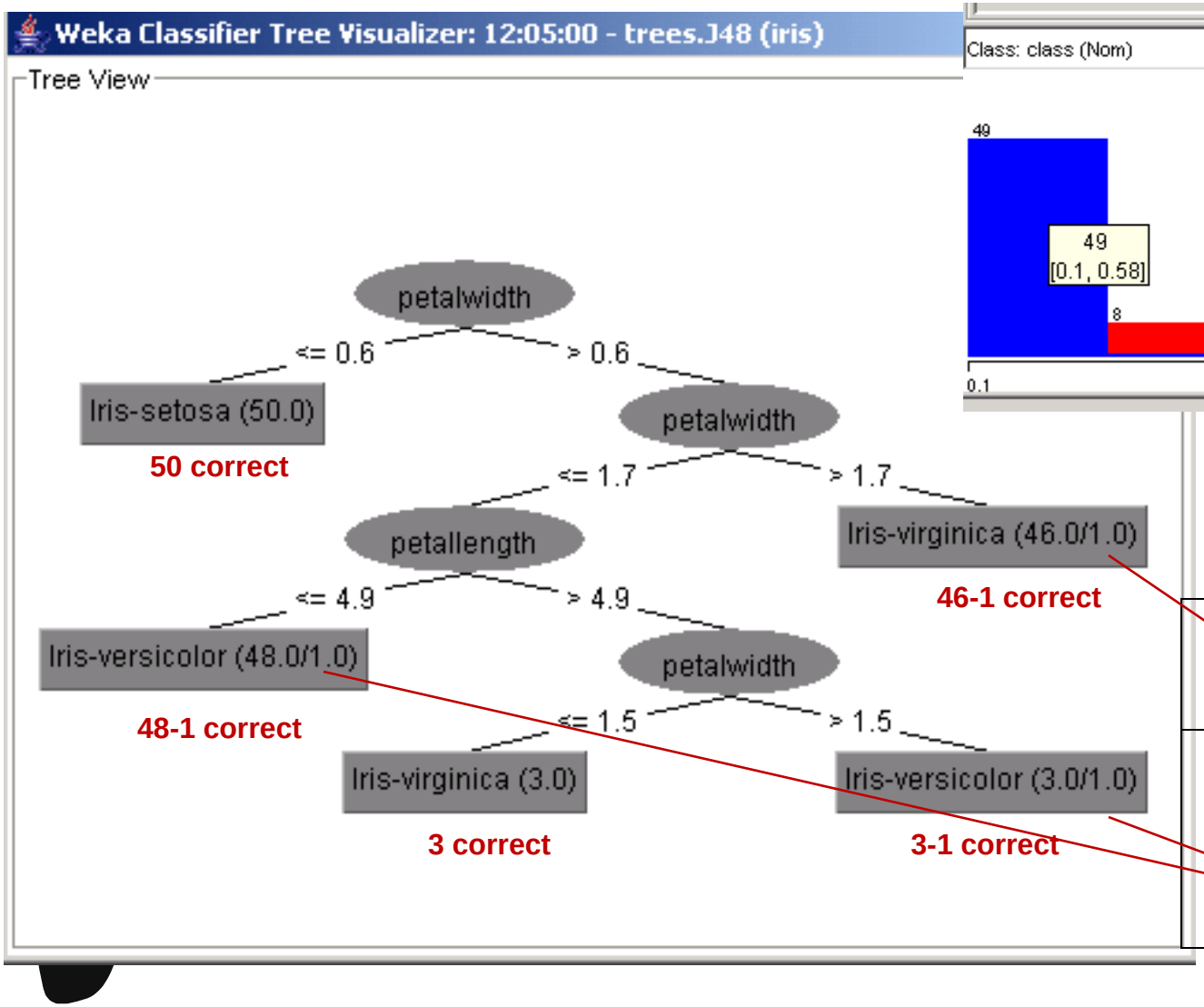
Decision trees: J48

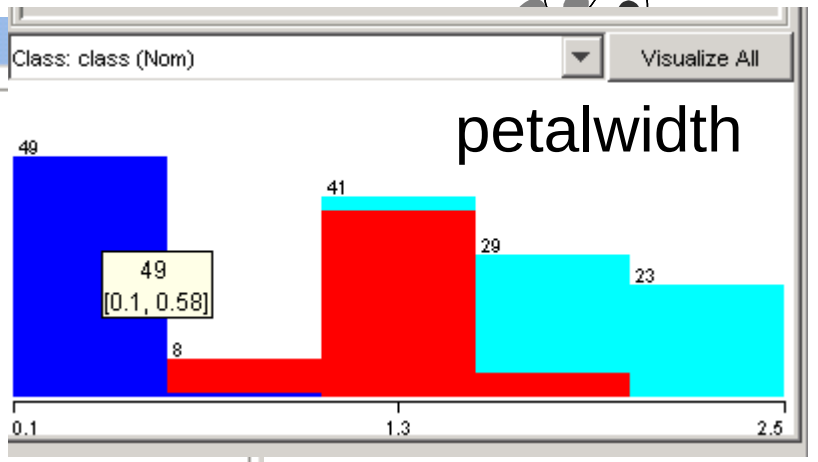
HAP/LAP





Visualize tree

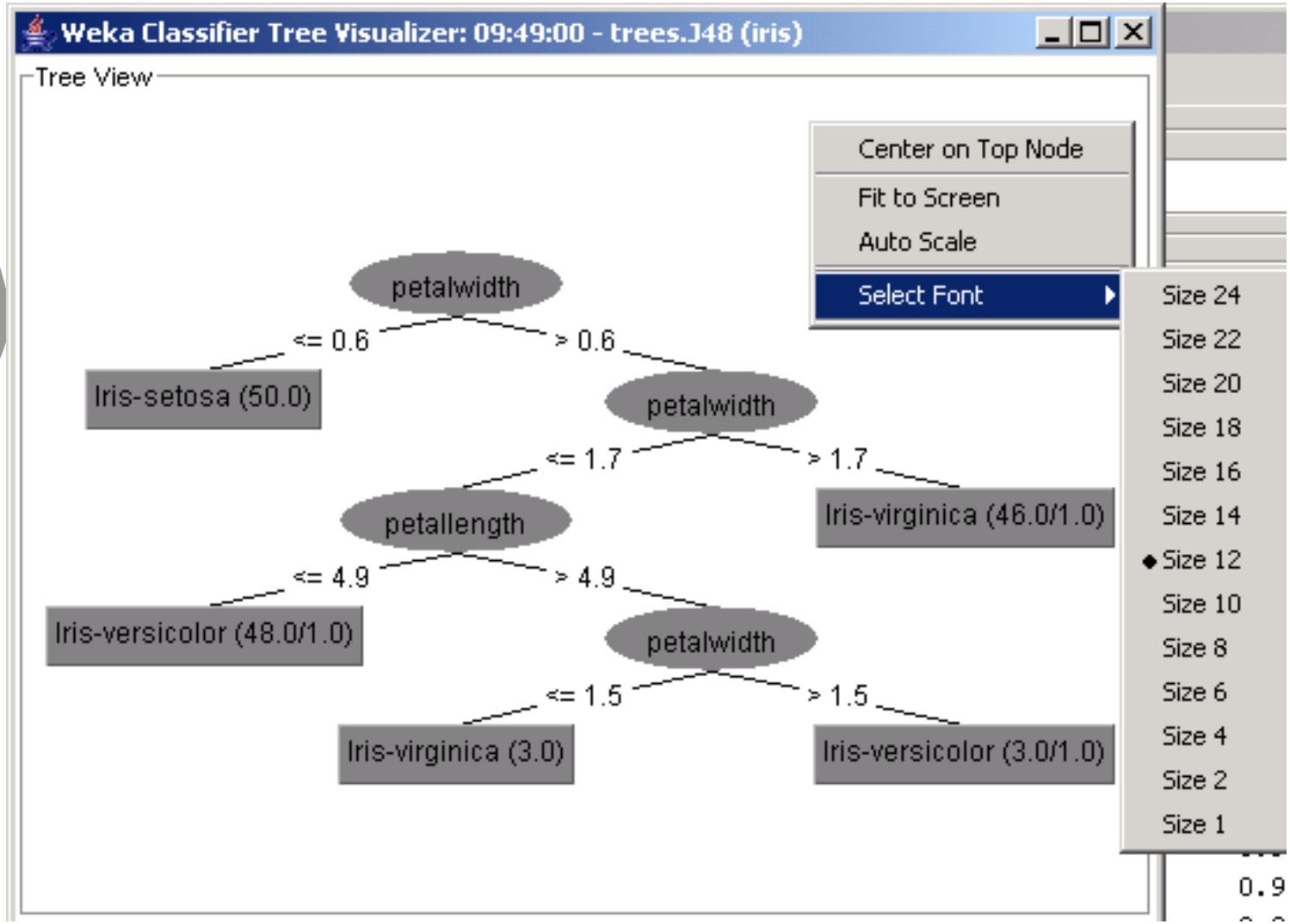




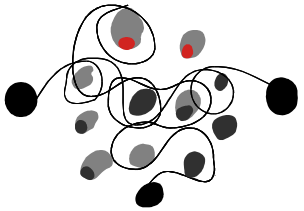
Classified as			
seto	versi	virgi	
50	0	0	set
0	49	1	ver
0	2	48	vir

J48
iris.arff

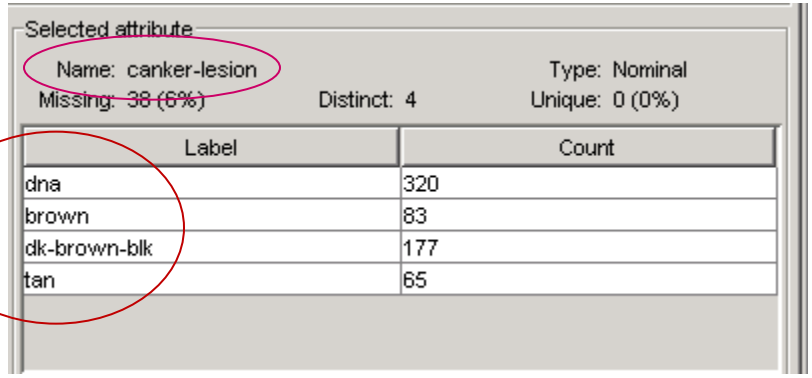
HAP/LAP

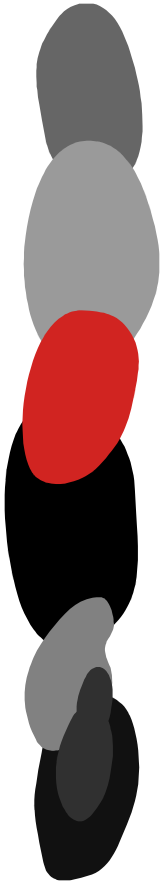


J48
soybean.arff



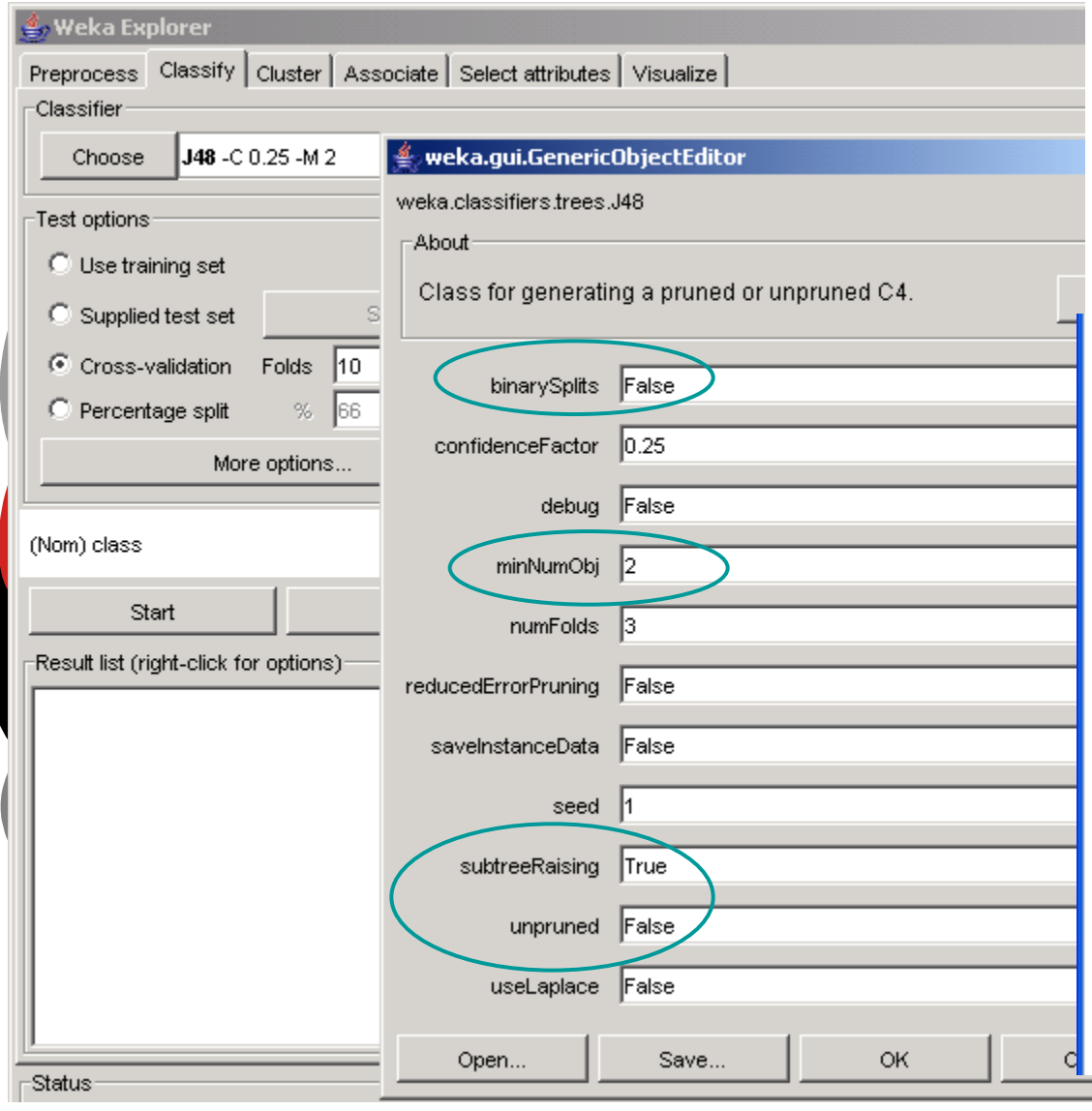
HAP/LAP





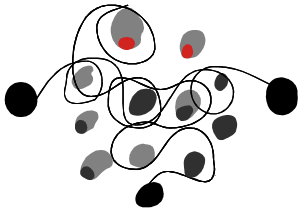
```
leafspot-size = lt-1/8
    canker-lesion = dna
|   |   leafspots-marg = w-s-marg
|   |   |   seed-size = norm: bacterial-blight (21.0/1.0)
|   |   |   seed-size = lt-norm: bacterial-pustule (3.23/1.23)
|   |   leafspots-marg = no-w-s-marg: bacterial-pustule (17.91/0.91)
|   |   leafspots-marg = dna: bacterial-blight (0.0)
|   canker-lesion = brown: bacterial-blight
    canker-lesion = dk-brown-blk: phytophthora-rot (4.78/0.1
    canker-lesion = tan: purple-seed-stain (11.23/0.23)
....
```

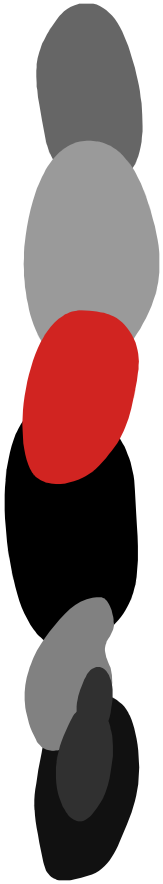
J48. Options




```
OPTIONS
binarySplits -- Whether to use binary splits
on nominal attributes when building the
trees.

confidenceFactor -- The confidence factor
used for pruning (smaller values incur more
pruning).

debug -- If set to true, classifier may
output additional info to the console.

minNumObj -- The minimum number of instances
per leaf.

numFolds -- Determines the amount of data
used for reduced-error pruning.  One fold is
used for pruning, the rest for growing the
tree.

reducedErrorPruning -- Whether reduced-error
pruning is used instead of C.4.5 pruning.

saveInstanceData -- Whether to save the
training data for visualization.

seed -- The seed used for randomizing the
data when reduced-error pruning is used.

subtreeRaising -- Whether to consider the
subtree raising operation when pruning.

unpruned -- Whether pruning is performed.

useLaplace -- Whether counts at leaves are
smoothed based on Laplace.
```

J48. Options





Pruning

Prepruning (forward pruning)

decide where to cut when building the tree

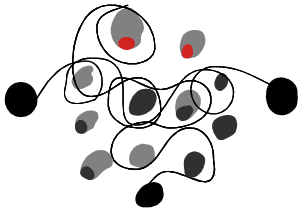
Postpruning (backward pruning)

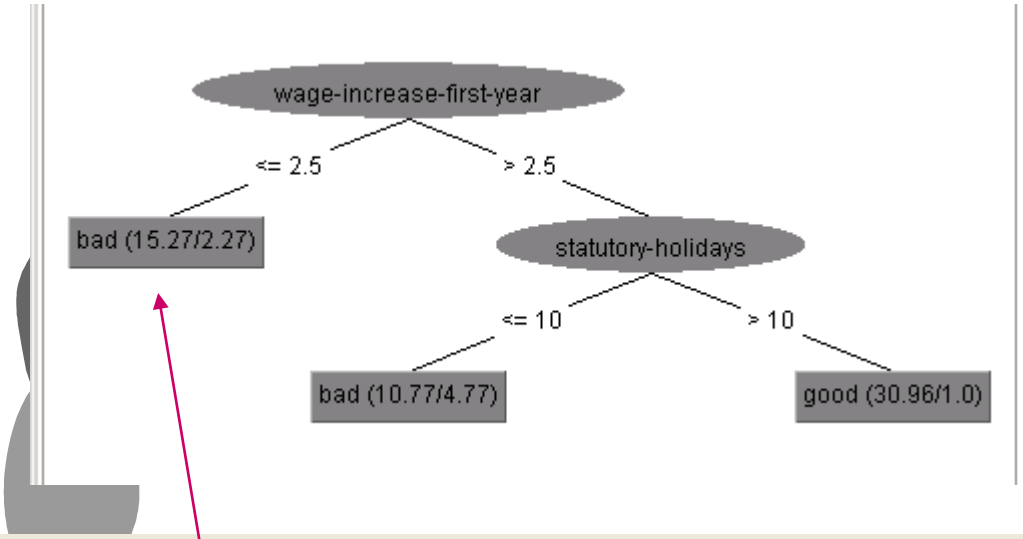
- generate the tree and then analyze to decide where to cut
- most used option
- Two options in each node

  subtree replacement → replace with leaves

  subtree raising → remove subtrees and replace with the lower part → reclassify→ time

J48. Options. *labor.arff*



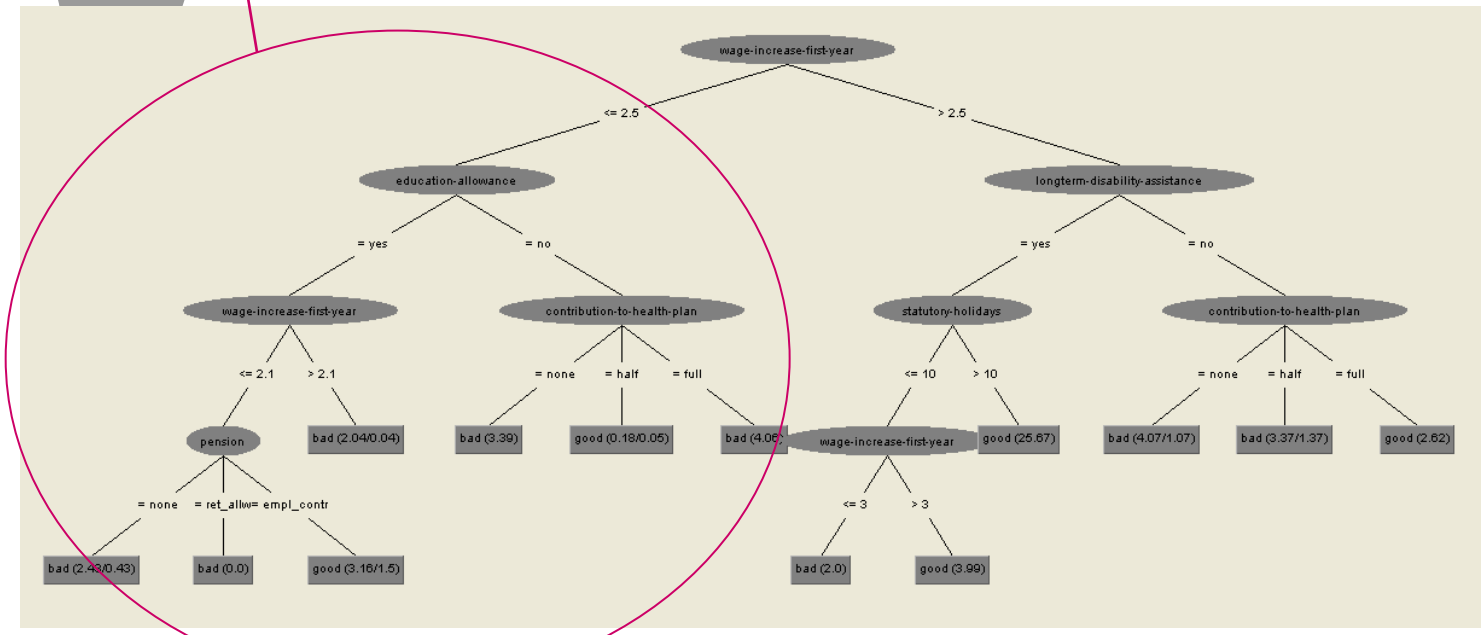


Subtree replacement

5 leaves bad → bad

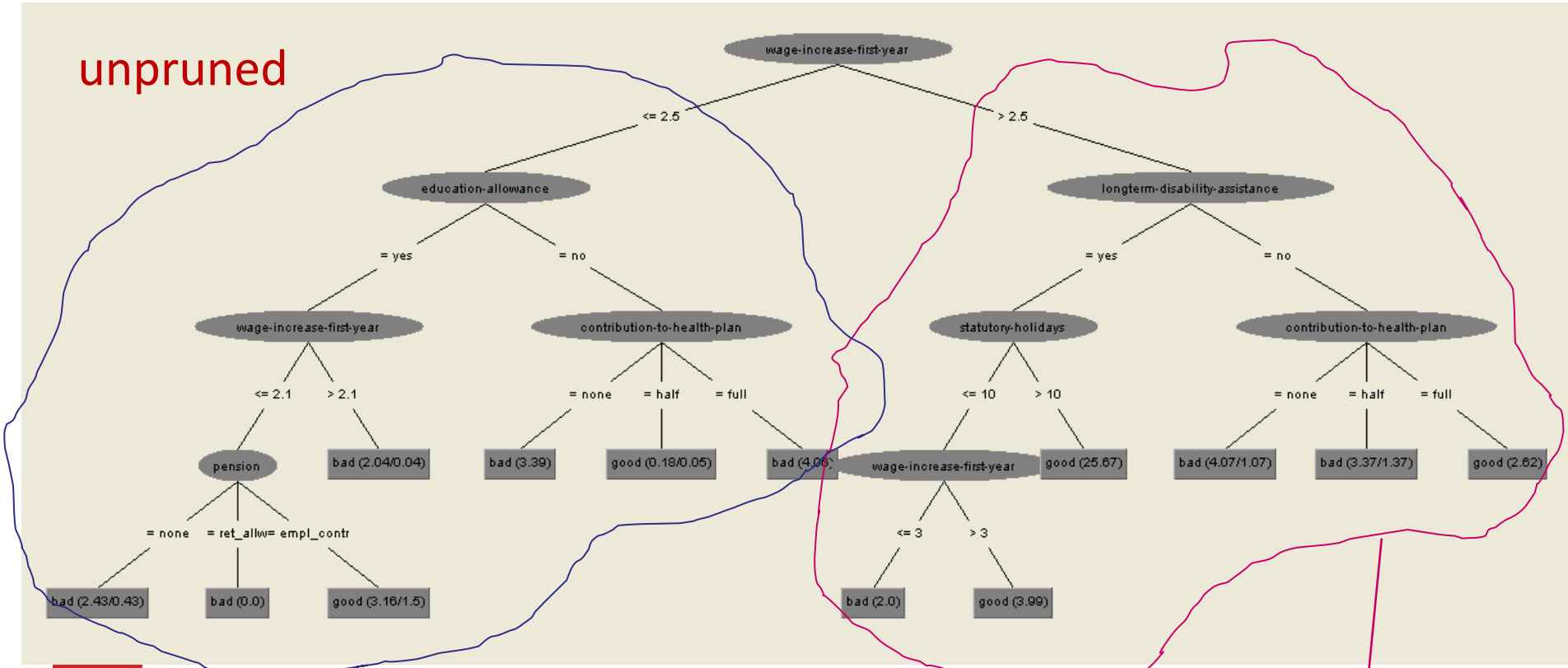
2 leaves good

F-measure = 0.89



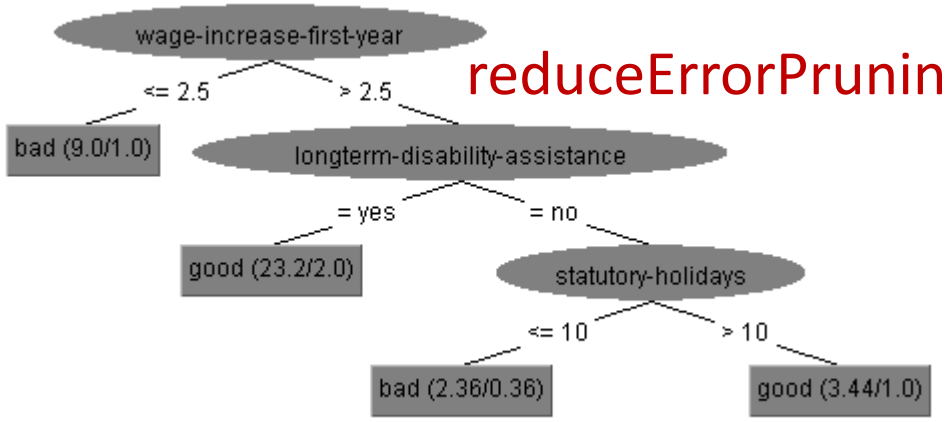
unpruned

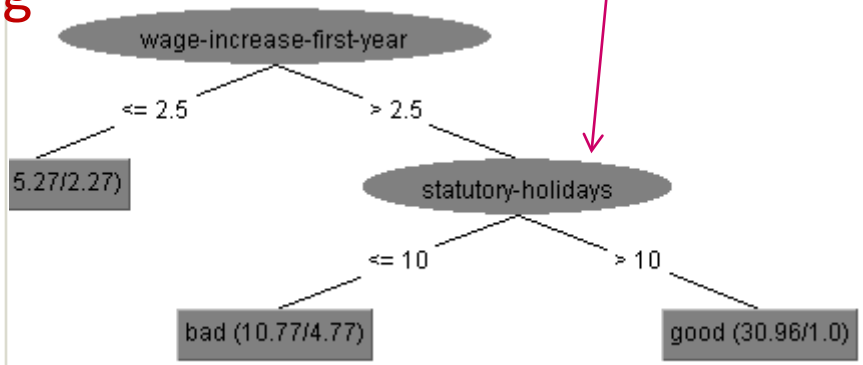
F-measure = 0.89



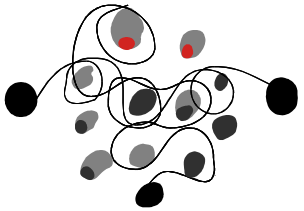
Subtree Raising

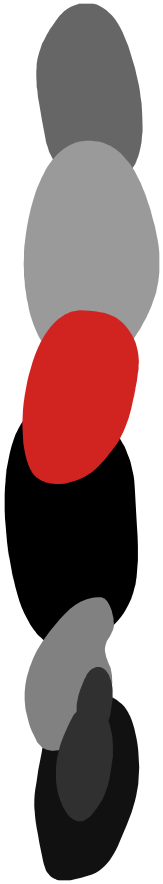
reduceErrorPruning





Other tree options



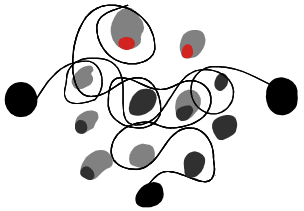


- ***Random Tree***:
  - K features are selected randomly in each node (try *soybean.arff*)
- ***ADTree***: *Freund, Y., Mason, L (1999)*
  - bi kategoriko atazak (ad. labor.arff)
  - Boosting iterations: adds 3 nodes in each iteration
- ***NBTrees:*** *Ron Kohavi (1996)*
  - NB classiefiers in leaf nodes
- ***DecisionStump:***
  - Generates binary trees of a single level.
  - To use in Boosting methods
- ***Id3:*** *R. Quinlan (1986)*
  - Only nominal attributes
  - Information gain → Gain Ratio
- ***Random Forest***:
  - forest of random trees
  - Bagging (multiple classifier system)

NBTrees

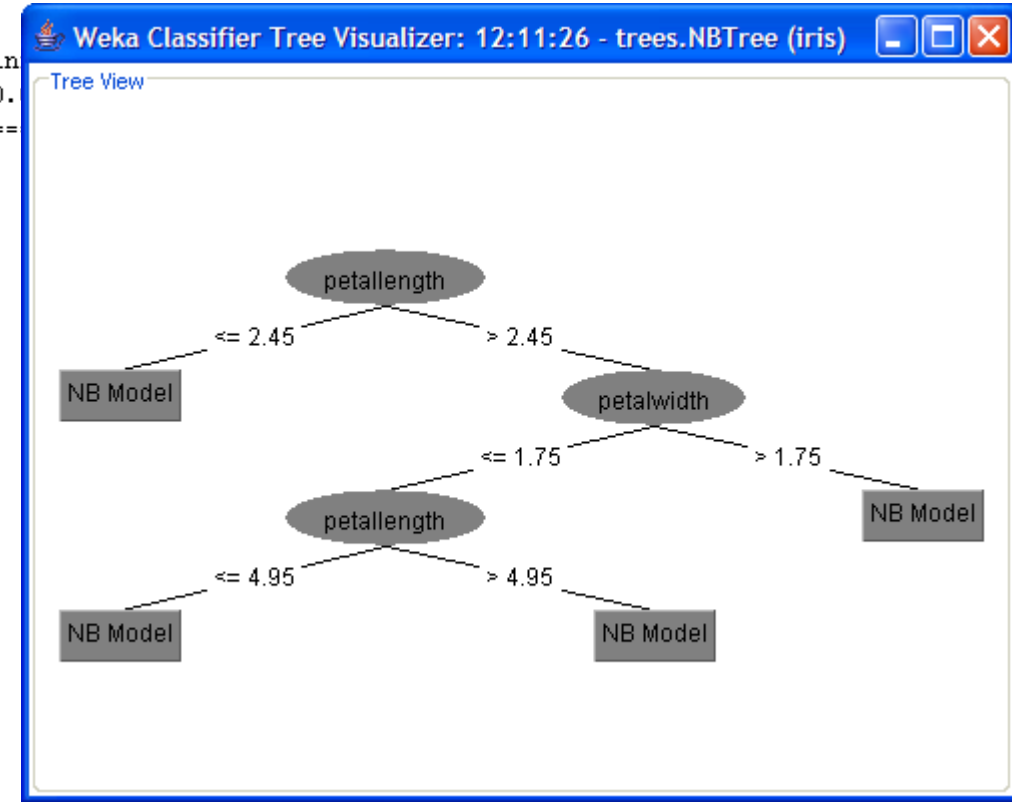
Corpusa

Ebaluazioa
1.5 Clustering (Cluster)
1.6 Atributuen arteko erlazioak (Associate)

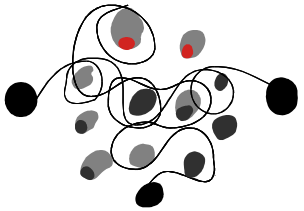


HAP/LAP

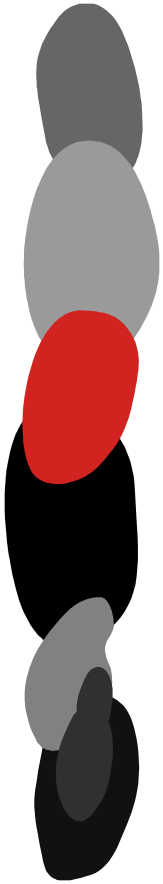
```
petallength <= 2.45: NB 1
petallength > 2.45
|   petalwidth <= 1.75
|   |   petallength <= 4.95: NB 4
|   |   petallength > 4.95: NB 5
|   petalwidth > 1.75: NB 6

Leaf number: 1 Naive Bayes Classifier

                         Class
Attribute          Iris-setosa Iris-versicolor  Iris-virgin
                       (0.96)          (0.02)          (0.
=============================================================
sepallength
  'All'                  51.0             1.0
  [total]                51.0             1.0

sepalwidth
  'All'                  51.0             1.0
  [total]                51.0             1.0

petallength
  'All'                  51.0             1.0
  [total]                51.0             1.0

petalwidth
  'All'                  51.0             1.0
  [total]                51.0             1.0



Leaf number: 4 Naive Bayes Classifier
```



Decision rules



- Divide-and-conquer technique
- Find rules that group instances of a class and discard those that are not of the class
- Overfitting (good in learning but not generalization capacity)



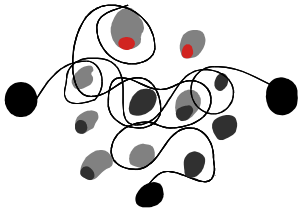
- Maximize the ratio: *p/t*

  t: number of examples coverd by the rule and *p* those that are positive among them
- Information gain: *p[log p/t – log P/T]*

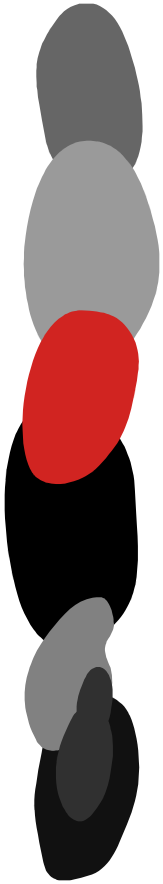
  gain with the new rule

  t and *p* the same, and *P* and *T* same value after introducing the new rule
- To introduce new rules:
  - as many positive examples as possible
  - as few negative as possible

Decision rules



- Rule induction



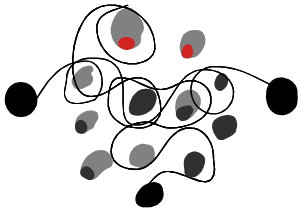
Example iris: 3 rules

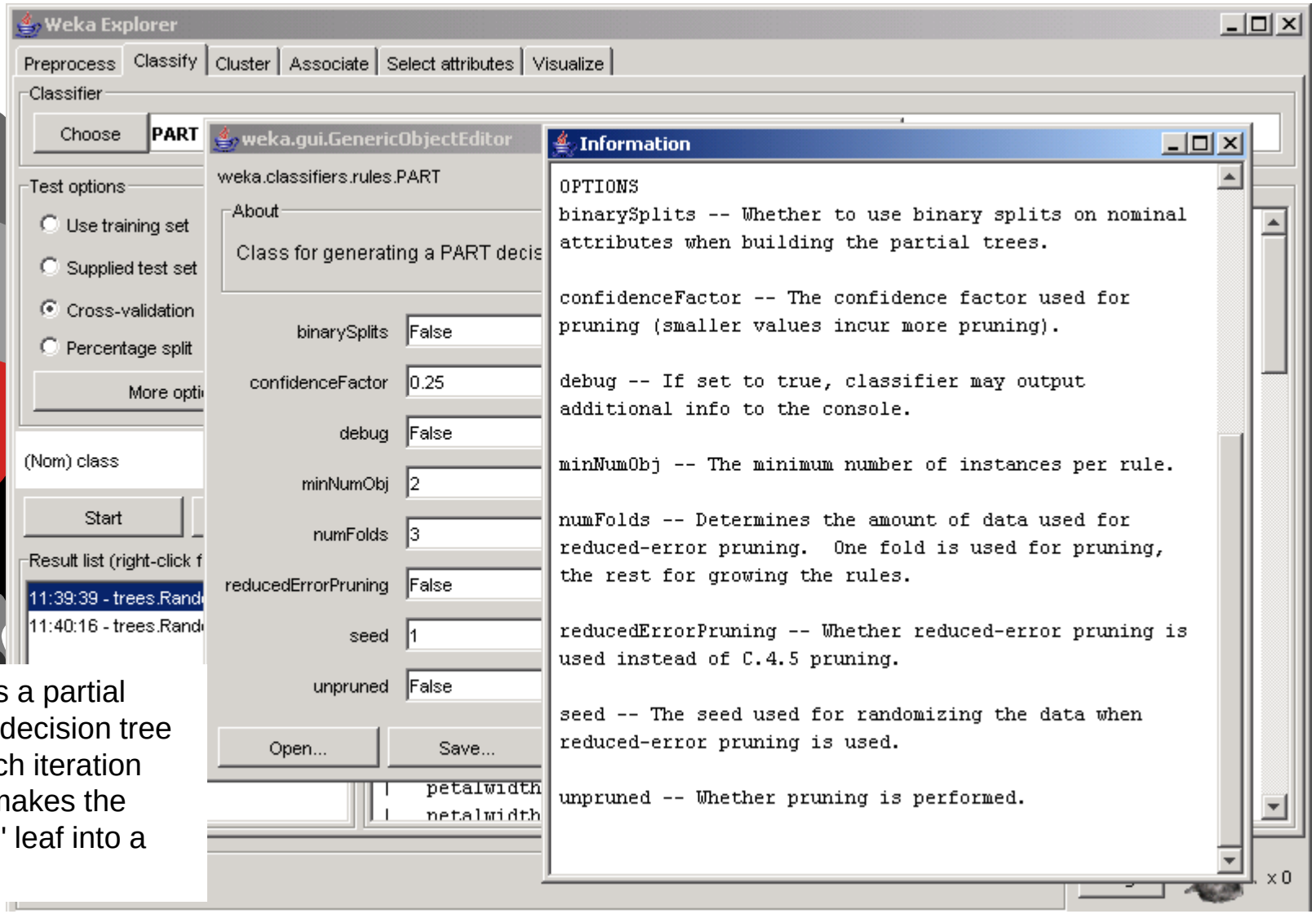
```
if petalwidth <= 0.6 then Iris-setosa
else    if petalwidth <= 1.7 AND petallength <= 4.9
  then Iris-versicolor
        else Iris-virginica
```

Example TC

```
  cyclist & Sky & … then sport
else    if crisis & euro & … then economy
     else …
```

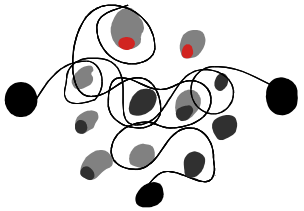
Decision rules: PART



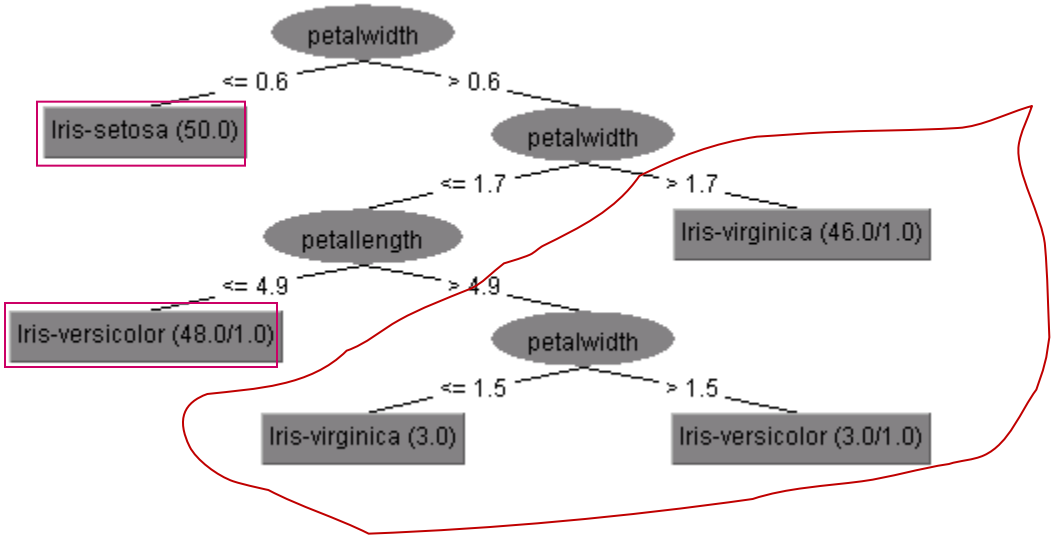


Builds a partial C4.5 decision tree in each iteration and makes the "best" leaf into a rule

Decision rules: PART

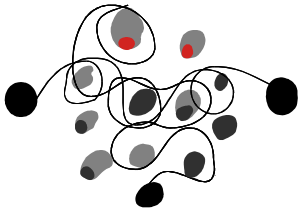


PART decision list---------- 3 rules
petalwidth <= 0.6: Iris-setosa (50.0)
petalwidth <= 1.7 AND petallength <= 4.9: Iris-versicolor (48.0/1.0)
: Iris-virginica (52.0/3.0)

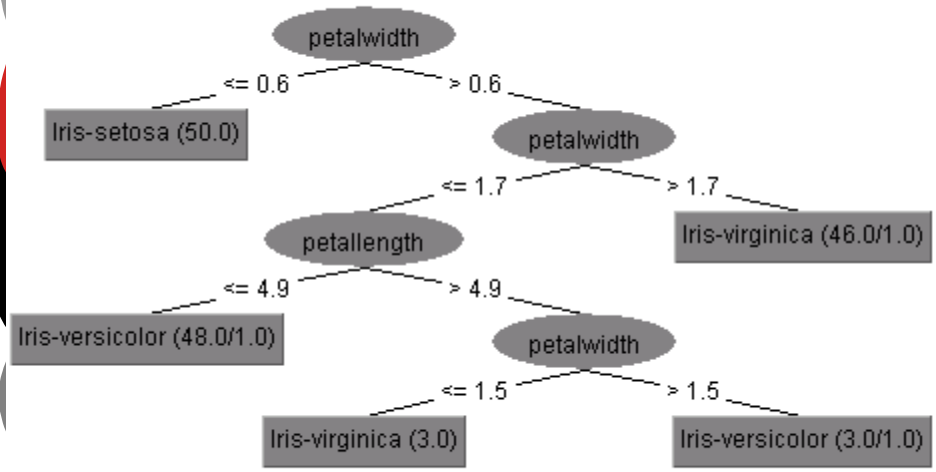


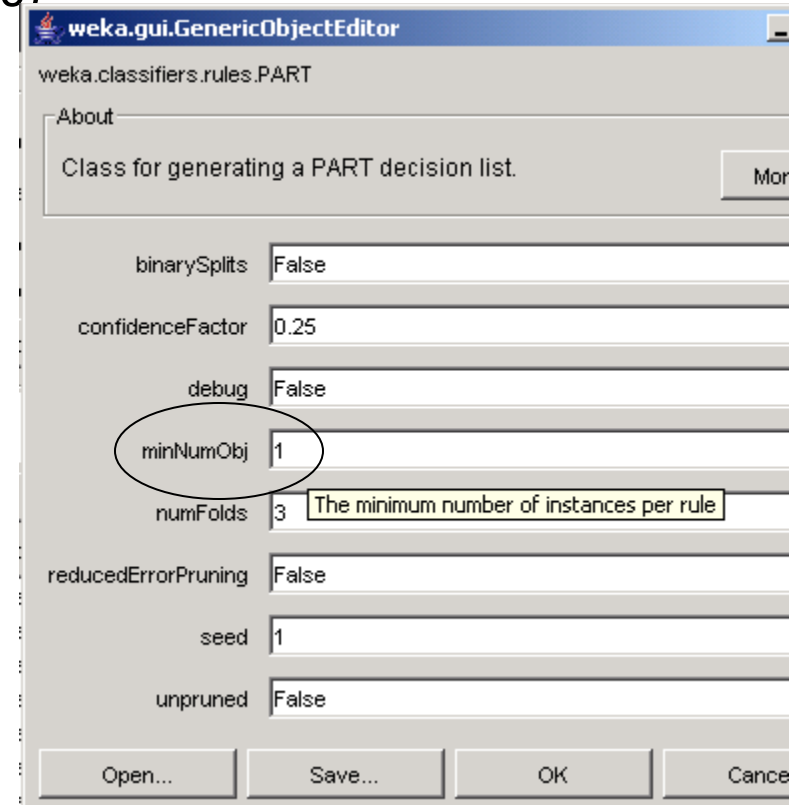
Iris-virginica

Decision rules: PART

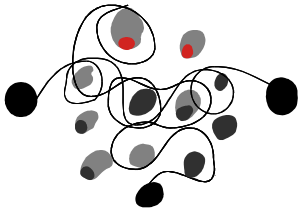


petalwidth <= 0.6: Iris-setosa (50.0)
petalwidth > 1.7 AND petallength > 4.8: Iris-virginica (43.0)
petallength <= 4.7 AND petalwidth <= 1.5: Iris-versicolor (42.0)
sepalwidth <= 3: Iris-virginica (11.0/4.0)
: Iris-versicolor (4.0)
Number of Rules : 5





Decision rules: other options



JRIP: similar to RIPPER (Repeated Incremental Pruning to Produce Error Reduction), in accuracy, number of rules and time. Not in memory

OneR: *R.C. Holte (1993)*

Generates a single rule

Prediction based on the feature with minimum error

Discretizes numeric attributes

Simple rules obtain often better results

DecisionTable: Ron Kohavi (1995)

Uses Best-first search to explore feature set

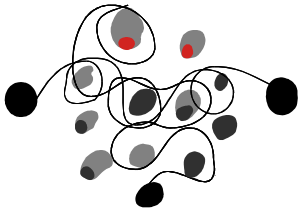
Cross-validation for evaluation

1-nn can be used when instances can not be classified with the information in the table

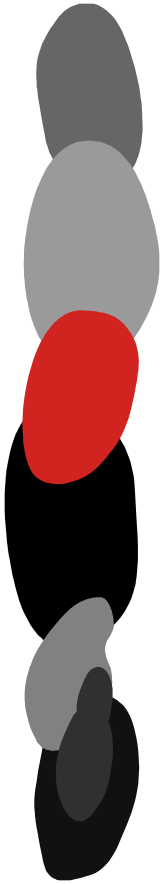
Rules:

petalwidth	class
'(1.75-inf)'	Iris-virginica
'(0.8-1.75]'	Iris-versicolor
'(-inf-0.8]'	Iris-setosa

Functions



Classifiers that can be classified with mathematical equations:



Regression

To predict based on numeric classes and attributes

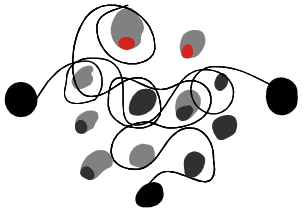
Ex.: cost of a flat based on size, number of bedrooms, …

Size (feet^2)	Number of bedrooms	Number of floors	Age of home (years)	Price ($1000)
2104	5	1	45	460
1416	3	2	40	232
1534	3	2	30	315
852	2	1	36	178
…	…	…	…	…

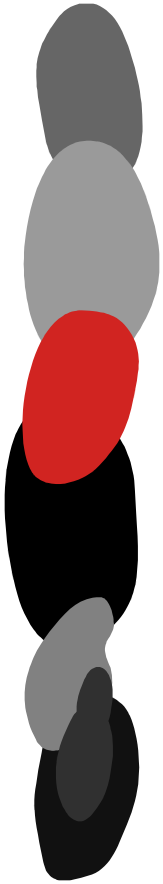
Classification

The values of the class are discrete

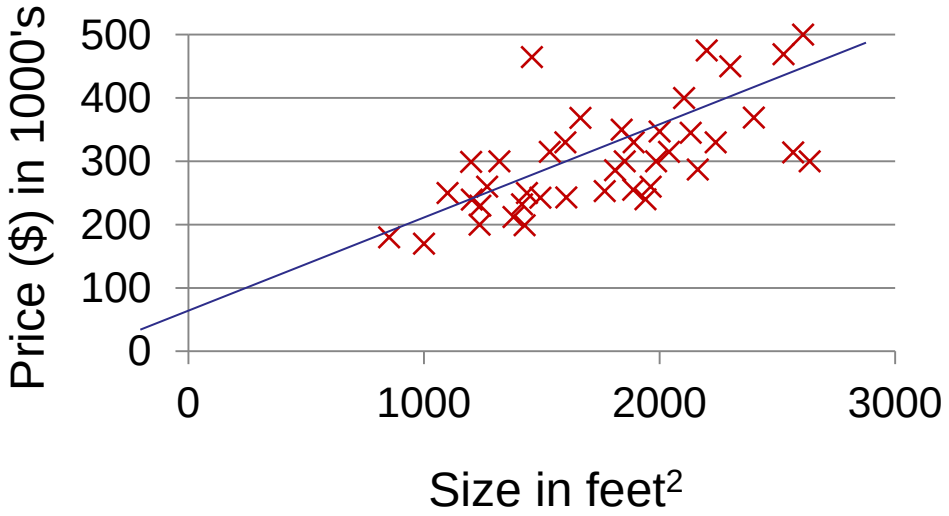
Functions



Example: cost of the flat according to size
A single attribute: size(x)



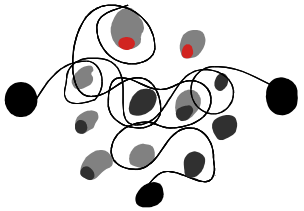
Size in feet² (x)	Price ($) in 1000's (y)
2104	460
1416	232
1534	315
852	178
…	…



Linear function (hypothesis)

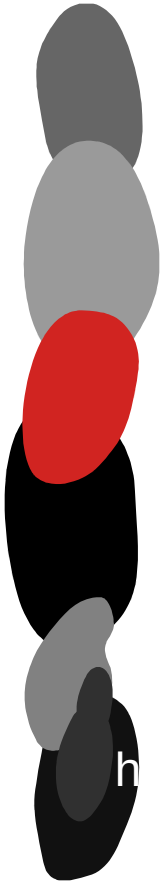
$h_w(x) = w_0 + w_1 x$ (w_i = weights)

Functions

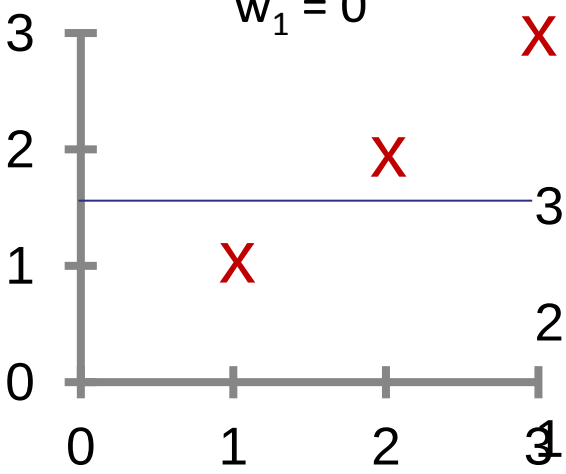


HAP/LAP

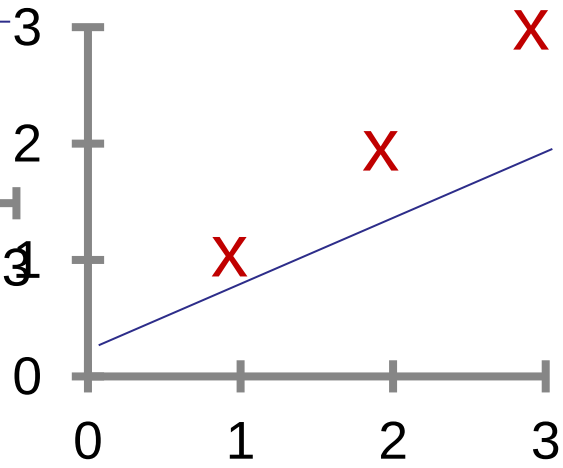
Example: $h_w(x) = w_0 + w_1 x$



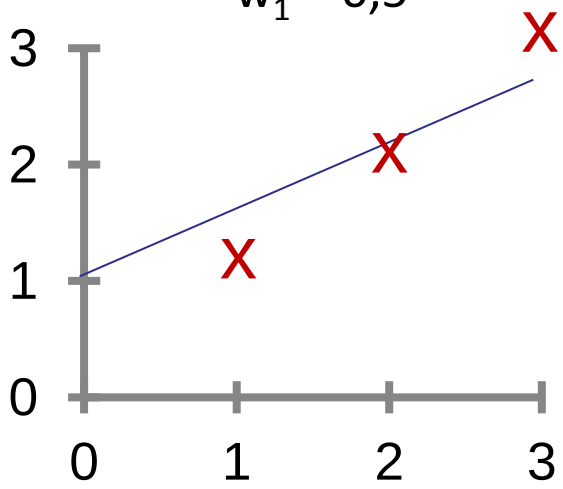
$w_0 = 1{,}5$
$w_1 = 0$



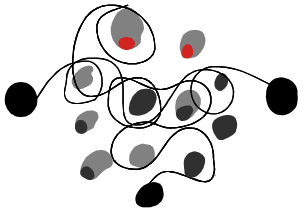
$w_0 = 0$
$w_1 = 0{,}5$



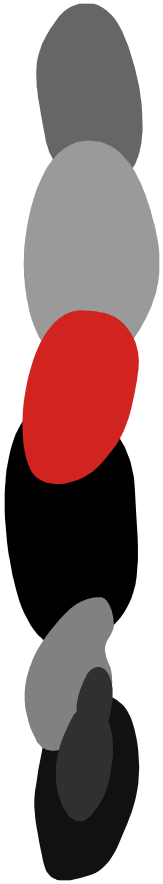
$w_0 = 1$
$w_1 = 0{,}5$

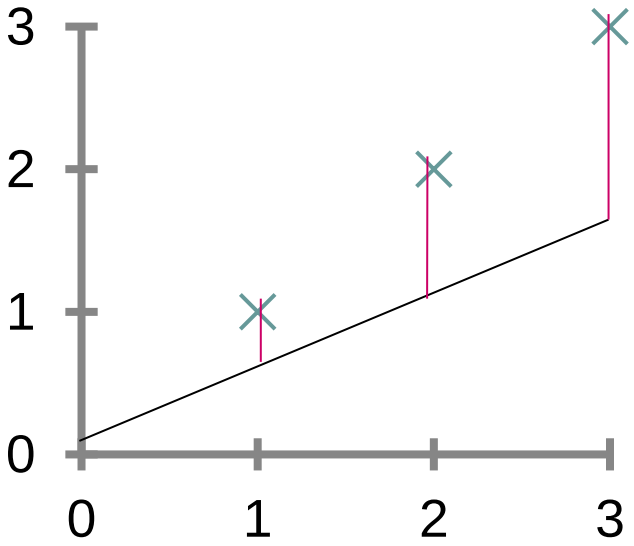


Functions



How to calculate the cost of the prediction?





Linear function

$h_w(x) = w_0 + w_1 x_1$

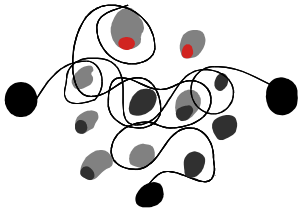
w_i = weights

Cost function:

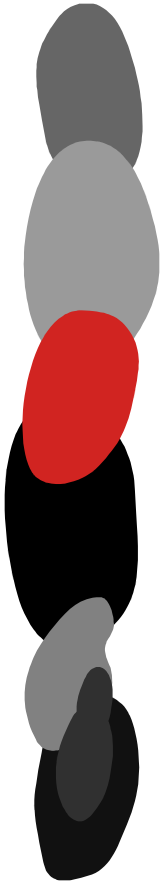
$$J(w_0, w_1) = \frac{1}{2m} \sum_{i=1}^{m} (h_w(x^{(i)}) - y^{(i)})^2$$

m: number of instances

Functions



Linear regression



To predict with numeric classes and attributes.

- **The class** is represented as a linear combination of the features with predicted weights

$$C = w_0 + w_1x_1 + w_2x_2 + \ldots + w_nx_n$$

weights are calculated from the training set

class weights features

- **The predicted class** for each example is calculated in the following way: ($x_0 = 1$)

$$w_0x_0^{(1)} + w_1x_1^{(1)} + w_2x_2^{(1)} + \ldots + w_nx_n^{(1)} = \sum w_jx_j^{(1)}$$

find w_j coefficients that minimize the squared difference between the predicted value and the real value

Predicted value

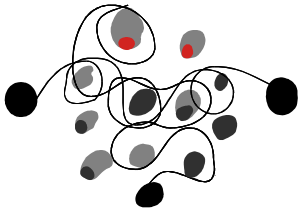
$$\sum_{i=1}^{m} (k^{(i)} - \sum_{j=0}^{n} w_ja_j^{(i)})^2$$

m: instances

n: number of features

Real value

Example: CPU.arff



Data

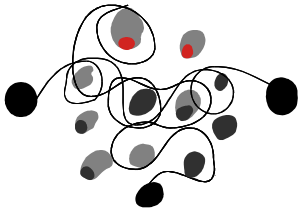
MYCT	MMIN	MMAX	CACH	CHMIN	CHMAX	class
125,	256,	6000,	256,	16,	128,	198
29,	8000,	32000,	32,	8,	32,	269
29,	8000,	32000,	32,	8,	32,	220
29,	8000,	32000,	32,	8,	32,	172
….						

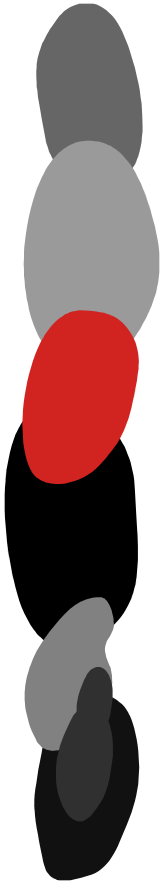
To calculate w coefficients, the following values need to be minimized:

$$(198 - \sum w_j x_j)^2 + (269 - \sum w_j x_j)^2 + (220 - \sum w_j x_j)^2 + (172 - \sum w_j x_j)^2 + \ldots$$

```
class =

      0.0491 * MYCT +
      0.0152 * MMIN +
      0.0056 * MMAX +
      0.6298 * CACH +
      1.4599 * CHMAX +
    -56.075
```

Functions: Artificial Neural Networks





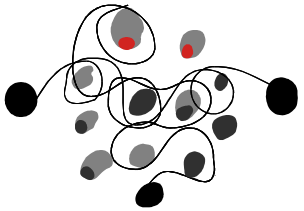
Idea:

- model mathematically the human intellectual capacities.
- massively parallel computation schemas
- Unstable classifiers appropriate for multiple classifier systems

Universal approach property:

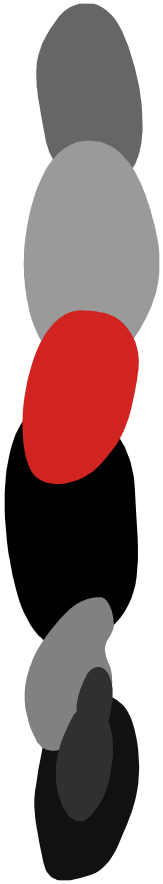
Some of the models, (Multilayer Perceptron (MLP), Radial Basis Function (RBF)), if an infinite number of patterns can be used, have the capacity to approach any discriminate function with a certain precision.

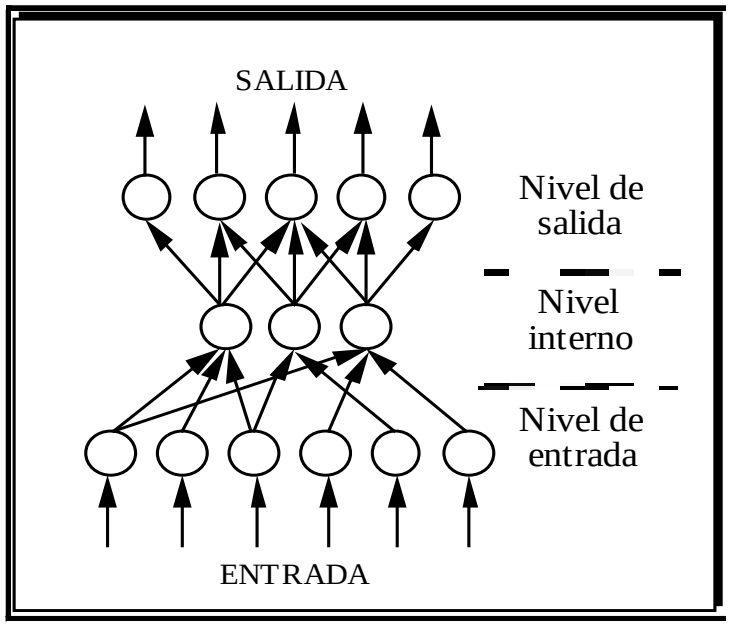
Functions: Artificial Neural Networks



Multilayer Perceptron (MLP)

Is a feedforward network where every neuron in a layer is connected to every neuron in next layer. The structure would be the following:

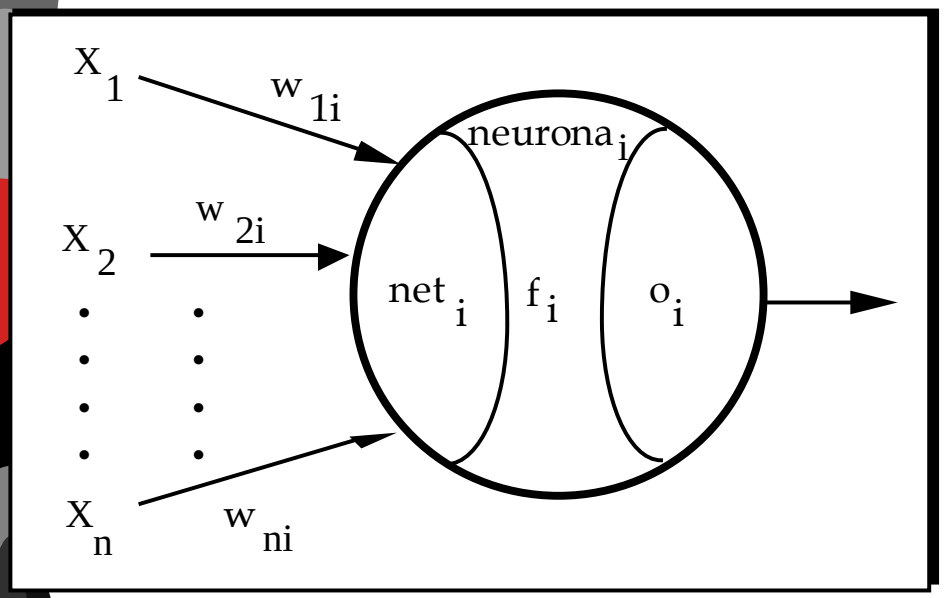




Functions: Artificial Neural Networks

The basic structure of an ANN is a single neuron

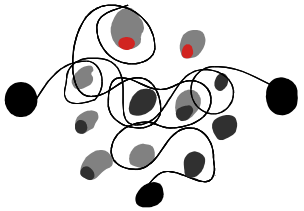
When the ANN has a single neuron it is called simple linear perceptron.



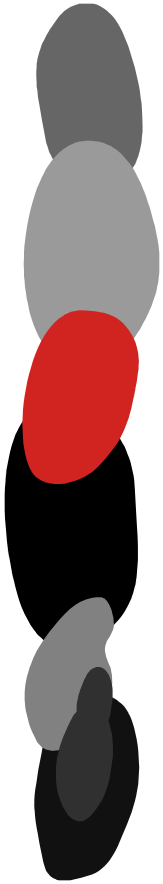
$$o = f\left(\sum_{j=1}^{N} w_j * x_j + w_0\right)$$

f is an identity type function, sign, sigmoid, etc.

Functions



Perceptron



At the beginning, all the weights of the features of category c_i are identical: w_{ki}

For new examples to learn (d_j) the classifier classifies with the weights the classifier has and:

If the classification is correct: no action

If the classification is not correct :

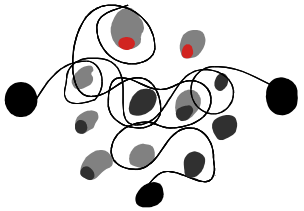
If $d_j \in c_i$ then $w_{ki} := w_{ki} + \alpha$ ($\alpha > 0$)

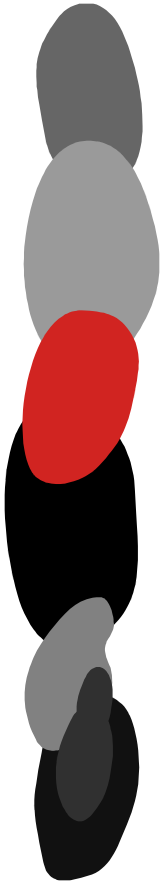
If $d_j \notin c_i$ then $w_{ki} := w_{ki} - \alpha$ ($\alpha > 0$)

($w_{ki} \rightarrow$ all t_k where $w_{kj} = 1$)

If the weight of the feature (w_{ki}) diminishes in the learning process, feature (t_k) is not useful for classification and it can be removed (on-the-fly term space reduction)

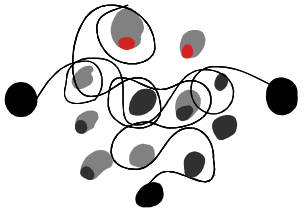
Perceptron. Example

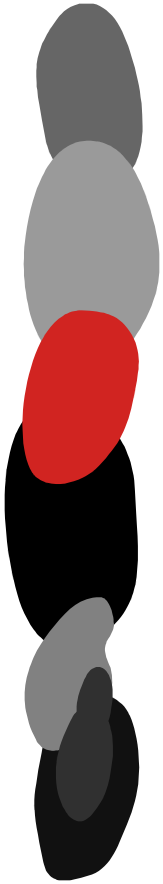




- Document classification
  - d_i = … book presentation in Koldo Mitxelenan …
  - d_l = … Europes´ economy… the euro has risen
  - d_j = … the writer will talk about the book in Koldo Mitxelena
- Representation (lemmas)
  - d_i = … book present Koldo Mitxelena …
  - d_l = … Europe economy euro have rise…
  - d_j = … writer talk about book Koldo Mitxelena …
- Features:
  - **{writer, book, present, novel, … , read, Koldo, Mitxelena, have, Europe, economy, euro, rise, …}**
- Category:
  - **Culture**

Perceptron. Example





- {x_1=writer, x_2=book, x_3=present, x_4=novel, x_5=read, x_6=Koldo, x_7=Mitxelena, x_8=have , x_9=Europe, x_{10}=economy, x_{11}=euro, x_{12}=rise, …}

  $d_i = (x_1, x_2, \ldots, x_{12}) = \{0,1,1,0,0,1,1,0,0,0,0,0\}$ **Culture**

  $d_l = \{0,0,0,0,0,0,0,1,1,1,1,1\}$ **Economy**

  $d_j = \{1,1,0,0,0,1,1,1,0,0,0,0,0\}$ **Culture**

- Algorithm

  Beginning $w_k = 0.1$; $x_0 = 1$; $\alpha = 0.8$

  $d_i \rightarrow f(x) = x_0w_0 + x_1w_1 + x_2w_2 + \ldots + x_{12}w_{12} = 0.5$ $f(x) > 0 \rightarrow$ culture YES

  $d_l \rightarrow f(x) = x_0w_0 + x_1w_1 + x_2w_2 + \ldots + x_{12}w_{12} = 0.6$ $f(x) > 0 \rightarrow$ culture YES

  Error → recalculating weights ($-\alpha$) $w_0 - w_7 = 0.1$; $w_8 - w_{12} = -0.7$

  $d_j \rightarrow f(x) = x_0w_0 + x_1w_1 + x_2w_2 + \ldots + x_{12}w_{12} = -0.2$ $f(x) < 0 \rightarrow$ culture NO

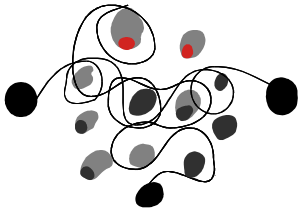
  Error → recalculating weights ($+\alpha$) $w_0 - w_2 = 0.9$; $w_3 - w_5 = 0.1$;

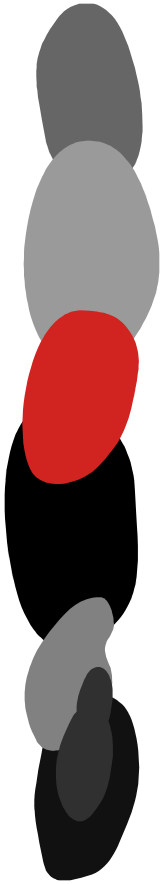
  $w_6 - w_7 = 0.9$; $w_8 = 0.1$; $w_9 - w_{12} = -0.7$

- Classifier

  hyperplane → weight vector

Functions





Winnow

To recalculate weights ($\alpha > 1$, $0 < \beta < 1$)

Multiplication instead of addition subtraction

Positive winnow

If the classification is correct: no action

If the classification is not correct

If $d_j \in c_i$ then $w_{ki} := w_{ki} \times \alpha$ $(\alpha > 1)$

If $d_j \notin c_i$ then $w_{ki} := w_{ki} \times \beta$ $(0 < \beta < 1)$

Balanced winnow

Two weights for each term (+ and -)

If the classification is not correct:

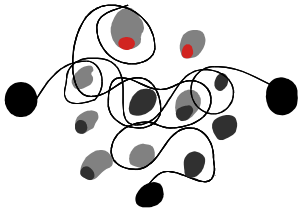
if $dj \in ci$ then $w_{ki} := w_{ki}^{+} \times \alpha$ $(\alpha > 1)$

$w_{ki} := w_{ki}^{-} \times \beta$ $(0 < \beta < 1)$

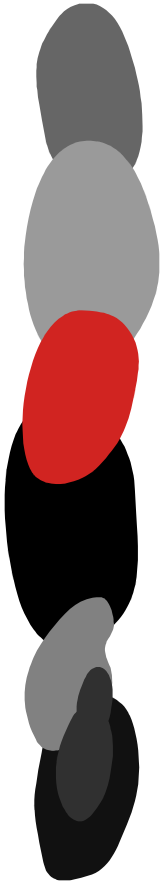
if $dj \notin ci$ then $w_{ki} := w_{ki}^{-} \times \alpha$

$w_{ki} := w_{ki}^{+} \times \beta$

Functions



SVM (Support Vector Machine)



Problems of linear classifiers:

- The boundaries between two classes are linear
- Too simple for many practical applications

SVM- uses linear models to define non linear boundaries between classes

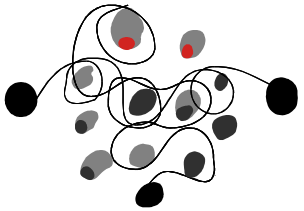
- Projects the input data using non linear mapping
- Converts the instance space to another space
- In the new space, the classes are linearly separable but in the original they are not.

Linear model: *maximun margin hyperplane*

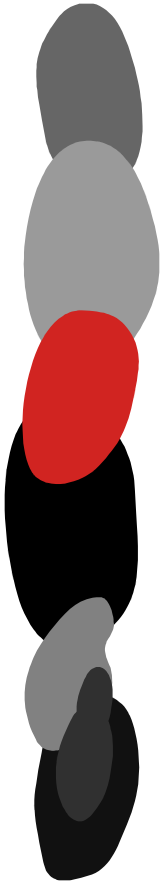
***Weka*: functions/SMO classifier** it is slow, CacheSize = 0

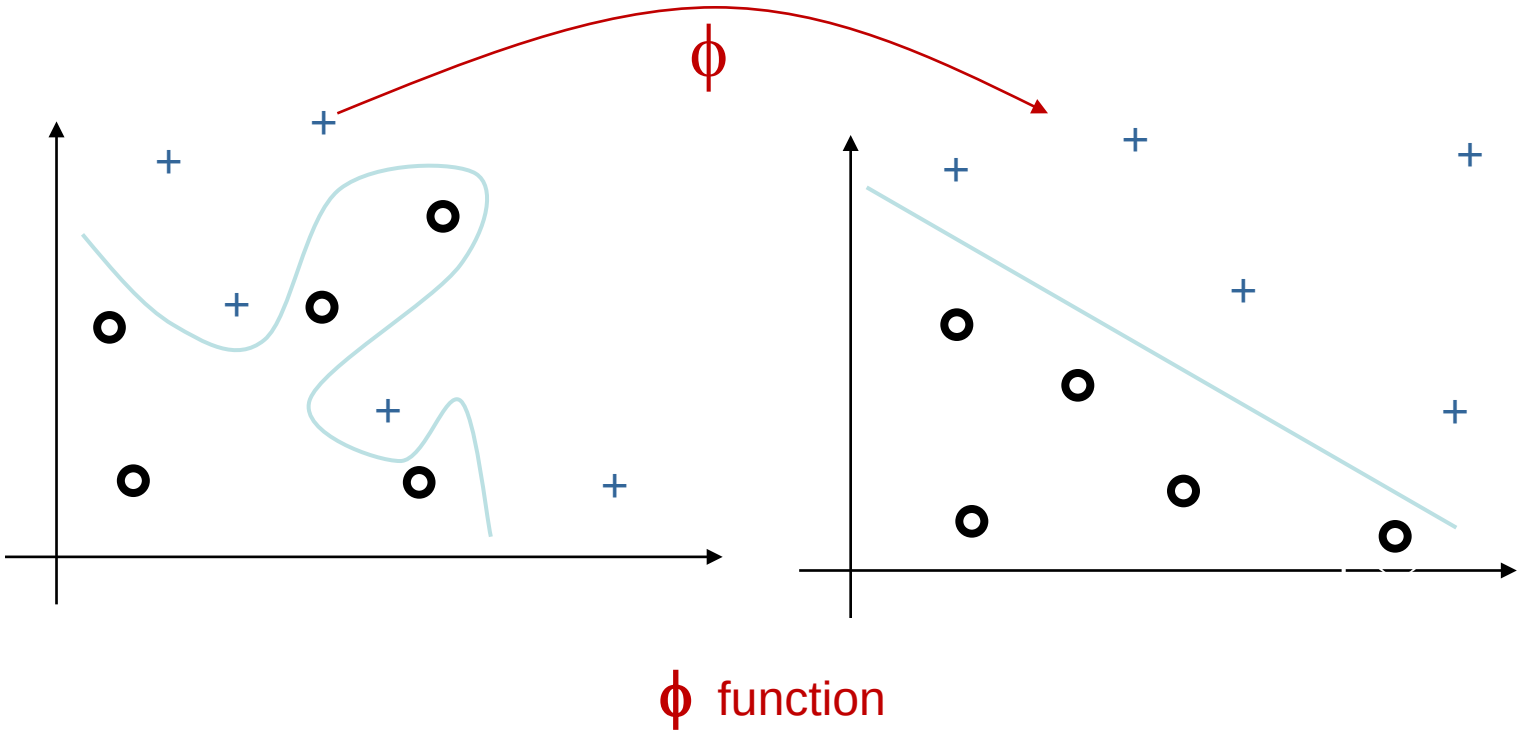
Call weka with: java -Xms512M -jar weka.jar

Functions

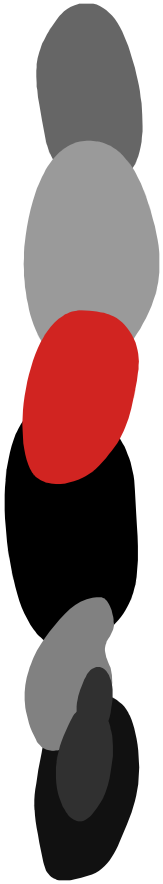


- Hyperplane discriminators: separate positive and negative examples by an hyperplane





Kernels

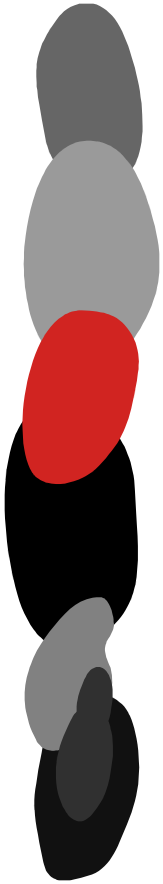


- **Kernel** Function defined in the new space

  any x, z $\in$ X $\qquad K = \langle \phi(\mathrm{x}) \cdot \phi(\mathrm{z}) \rangle$

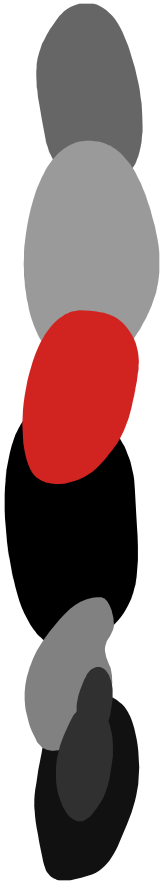
- The Kernel represents the similarity between two objects
- Depending on function f different kernels can be generated different learning algorithms can be used
- Advantages:
  - Not necessary to maintain the feature vectors of the data
  - Adequate to work with complex spaces
  - Good result with high dimensional problems(TC)
  - Efficient ways of calculating internal product exist

Kernels



- Document classification
  - d_i = … book presentation in Koldo Mitxelenan …
  - d_l = … Europes' economy… the euro has risen
  - d_j = … the writer will talk about the book in Koldo Mitxelenan
- Representation (lemas)
  - d_i = … book present Koldo Mitxelena …
  - d_l = … Europe economy euro have rise…
  - d_j = … writer talk about book Koldo Mitxelena …
- Features:
  - **{writer, book, present, nobel, … , read, Koldo, Mitxelena, have, Europe, economy, euro, rise, …}**

Kernels



$d_1 = x \rightarrow \phi(x) = \{0,1,1,0,0,....,0,1,1,0,0,0,0,0\}$
$d_2 = z \rightarrow \phi(z) = \{1,1,0,0,0,....,0,1,1,1,0,0,0,0\}$
$d_3 = v \rightarrow \phi(x) = \{0,0,0,0,0,....,0,0,0,1,1,1,1,1\}$

Kernel function to measure similarity between d_1 and d_2

$d_1 = x \rightarrow \phi(x) = \{0,1,1,0,0,....,0,1,1,0,0,0,0,0\}$
$d_2 = z \rightarrow \phi(z) = \{1,1,0,0,0,....,0,1,1,1,0,0,0,0\}$

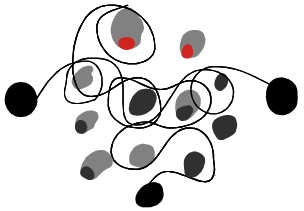
$k(x,z) = <\phi(x) \cdot \phi(z)> =$ **3 identical words**

Kernel function to measure similarity between d_2 and d_3

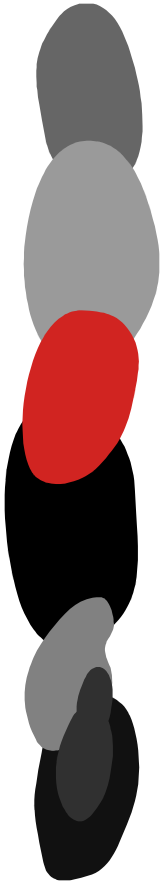
$d_2 = z \rightarrow \phi(z) = \{1,1,0,0,0,....,0,1,1,1,0,0,0,0\}$
$d_3 = x \rightarrow \phi(x) = \{0,0,0,0,0,....,0,0,0,1,1,1,1,1\}$

$k(x,z) = <\phi(x) \cdot \phi(z)> = 1 \rightarrow$ **a single word**

Kernels



- In the new space of the learning set, the **similarity** of each document with the rest of documents is represented.

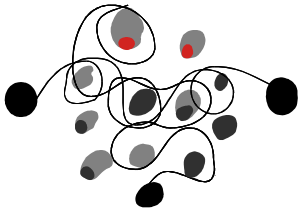


$$K_{training} = \begin{bmatrix} k(\mathbf{x}_1,\mathbf{x}_1) & k(\mathbf{x}_1,\mathbf{x}_2) & \dots & k(\mathbf{x}_1,\mathbf{x}_m) \\ k(\mathbf{x}_2,\mathbf{x}_1) & k(\mathbf{x}_2,\mathbf{x}_2) & \dots & k(\mathbf{x}_2,\mathbf{x}_m) \\ \dots & \dots & \dots & \dots \\ k(\mathbf{x}_m,\mathbf{x}_1) & k(\mathbf{x}_m,\mathbf{x}_2) & \dots & k(\mathbf{x}_m,\mathbf{x}_m) \end{bmatrix}$$

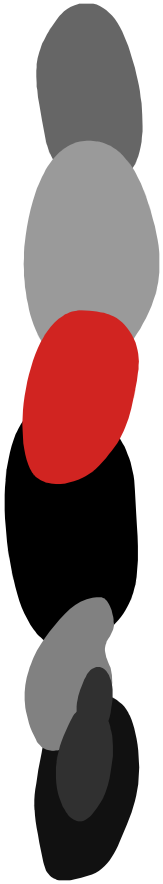
Kernel Gram Matrix

- It is not necessary to maintain all the original information

Kernels



- Three documents (d1, d2, and d3) and 13 words in the dictionary (t1, t2, … t12,t13)



term by document

$$D = [d1, d2, d3] = \begin{array}{c} \\ t1 \\ t2 \\ \dots \\ t13 \end{array} \begin{array}{c} \begin{array}{ccc} d1 & d2 & d3 \end{array} \\ \begin{pmatrix} 0 & 1 & 0 \\ 1 & 1 & 0 \\ \dots & \dots & \dots \\ 0 & 0 & 1 \end{pmatrix} \end{array}$$

doc by doc

d1 and d2-k share 3 words

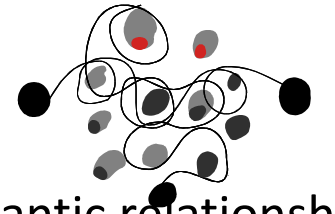
$$G = D'D = \begin{pmatrix} 0 & 1 & \dots & 0 \\ 1 & 1 & \dots & 0 \\ 0 & 0 & \dots & 1 \end{pmatrix} \cdot \begin{pmatrix} 0 & 1 & 0 \\ 1 & 1 & 0 \\ \dots & \dots & \dots \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 4 & 3 & 0 \\ 3 & 5 & 1 \\ 0 & 1 & 5 \end{pmatrix}$$

K (Gram)

- Size: 500 doc x 10.000 word→ 5.000.000 elements in matrix D

  250.000 elements in matrix G

Kernels



- The inner product does not take into account the semantic relationship between terms:

  d1 = health, hospital, doctor… d2 = hospital, pills, …

  $k(d1,d2) = \langle \phi(d1) \bullet \phi(d2) \rangle = 0$ identical words

- It is possible to measure similarity between words instead of document similarity
  - d1 = x → $\phi(x)$ = {0,1,1,0,0,….,0,1,1,0,0,0,0,0}
  - d2 = z → $\phi(z)$ = {1,1,0,0,0,….,0,1,1,1,0,0,0,0}
  - d3 = v → $\phi(x)$ = {0,0,0,0,0,….,0,0,0,1,1,1,1,1}

term by term

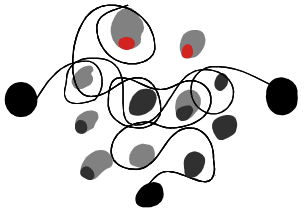
0 docs with words t1 and t13

$$DD' = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 1 & 0 \\ \dots & \dots & \dots \\ 1 & 1 & 0 \\ \dots & \dots & \dots \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 0 & 1 & \dots & 0 \\ 1 & 1 & \dots & 0 \\ 0 & 0 & \dots & 1 \end{pmatrix} = \begin{pmatrix} 1 & 1 & \dots & \textcircled{0} \\ 1 & 2 & \dots & 1 \\ \dots & \dots & \dots & \dots \\ 1 & \textcircled{2} & & \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & 1 \end{pmatrix} \quad T$$

2 docs with words t2 and t7

Similarity between words. Different words but they appear together

Kernels



- Identity (linear kernel) $K(\mathbf{x},\mathbf{z}) = \langle \mathbf{x}\cdot\mathbf{z} \rangle$
- Polynomial (polinomy of degree d) $K(\mathbf{x},\mathbf{z}) = \left(\langle \mathbf{x}\cdot\mathbf{z} \rangle + c\right)^d$

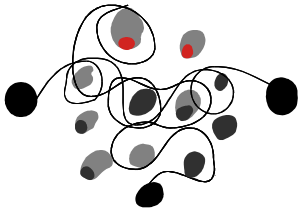
$$\langle \mathbf{x}\cdot\mathbf{z} \rangle^2 = \left(\sum_{i=1}^N x_i z_i\right)^2 = \left(\sum_{i=1}^N x_i z_i\right)\cdot\left(\sum_{j=1}^N x_j z_j\right) = \sum_{i=1}^N \sum_{j=1}^N x_i x_j z_i z_j = \sum_{i,j=1}^N (x_i x_j)(z_i z_j)$$

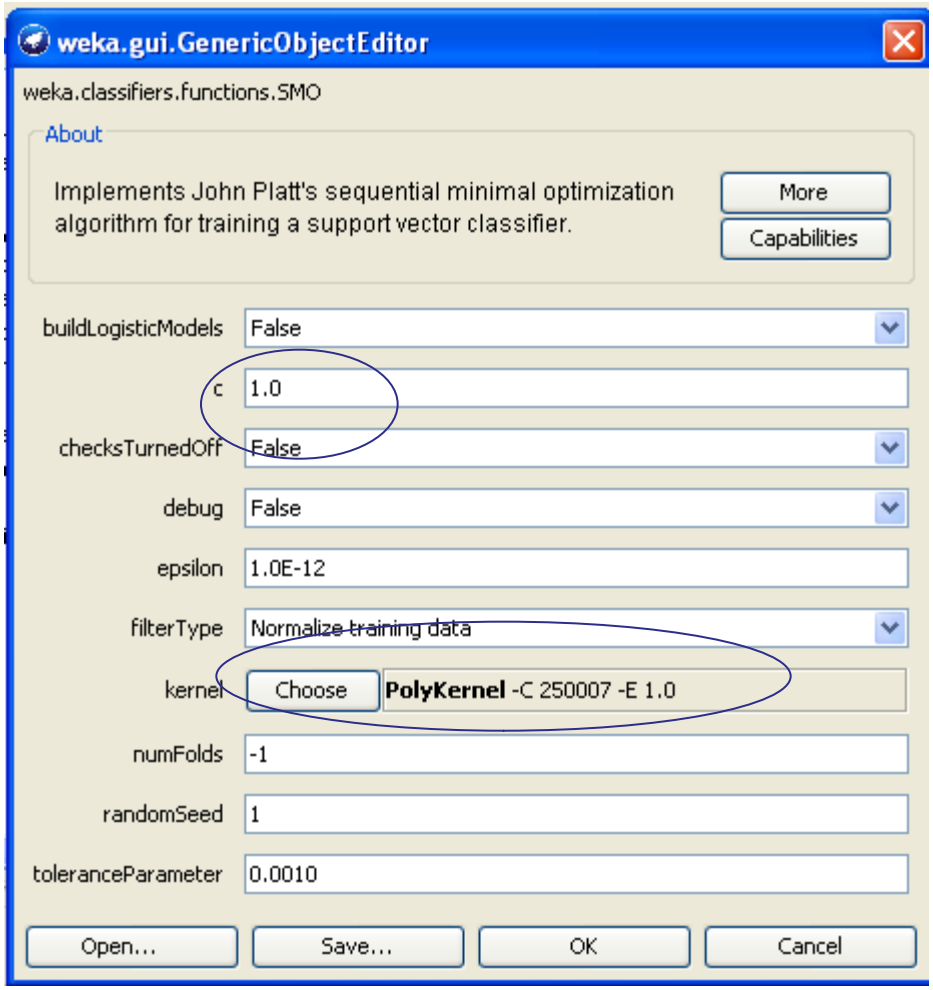
- Gaussian kernel (Radial Basis Function, RBF)

$$K(\mathbf{x},\mathbf{z}) = \exp\left(-\|\mathbf{x}-\mathbf{z}\|^2 / 2\sigma^2\right)$$

- Sigmoid $K(\mathbf{x},\mathbf{z}) = \tanh\left(\kappa\langle \mathbf{x}\cdot\mathbf{z} \rangle + \vartheta\right)$

SMO (sequential minimal optimization)



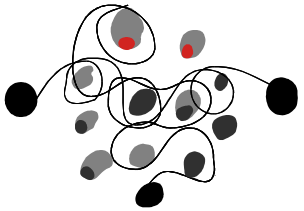


Weka
- Normalizes attributes
- Converts nominal attributes to binary

Select C
- small C : great error tolerance → many training examples missclassified
- bigger C: results will improve
- very big C : great importance to the training data → overfitting

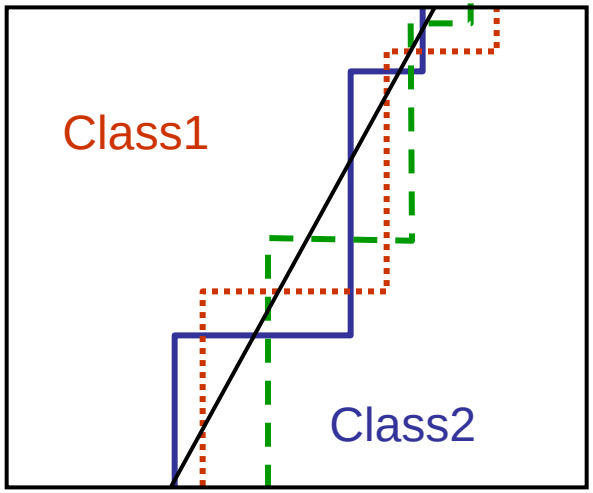
Select Kernel
- Polinomial (PolyKernel): berretzailea
- Gaussian kernel (RBF)

Multiple classifier systems

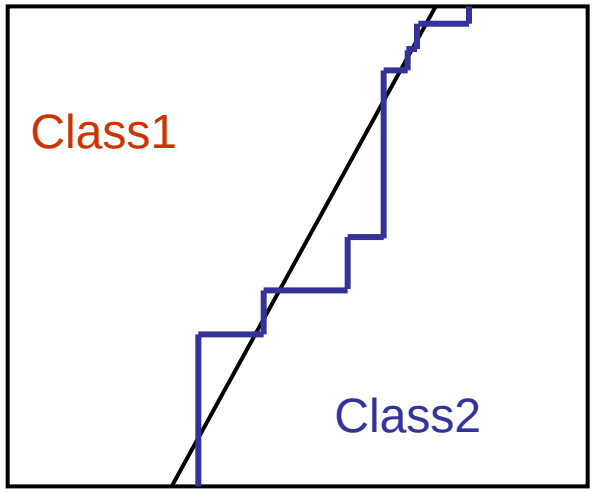


Motivation*:*

- **Statistical**: For small training samples the algorithm might find different hypothesis with the same performance. Taking into account several classifiers reduces the risk of selecting the wrong classifier.
- **Representational**: In many learning problems the objective function can not be represented for none of the classifiers.
- **DT example:**

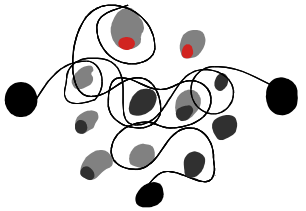


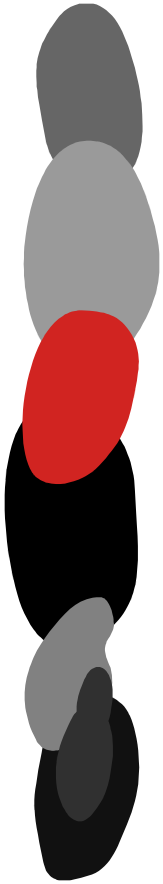
Decision Boundary (DB) of 3 individual DTs



Corresponding DB of the voting classifier

Multiple classifier systems





T classifiers $\Phi_1, \Phi_2, \ldots, \Phi_T$, to solve the same task

Classification based on different learning methods

result: combination of the result of *k* classifiers

- Voting in classification tasks
- Average result if it is numeric

T classifiers of the same type

Bagging: same classifier different subsamples. All classifiers same weight

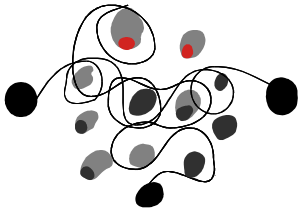
Boosting: complementary classifiers. (sequential learning)

Random subespace methods: classifiers built with different sets of features

Weka: Vote (select T different classifiers)

AdaBoost, Bagging, Random subespace (select a classifier)

Multiple classifier systems: bagging

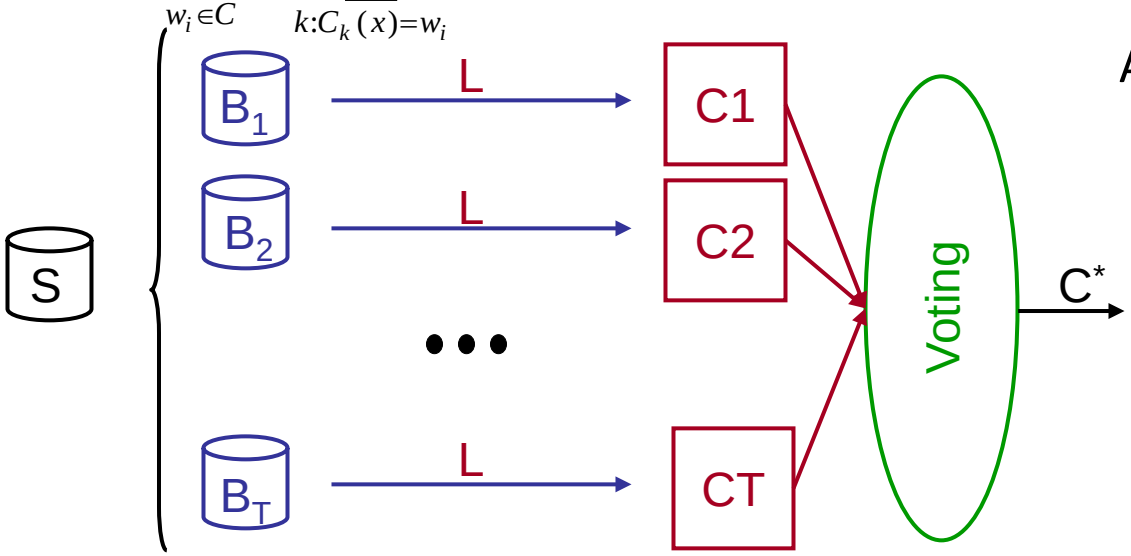


Bagging: Bootstrap aggregating, (Breiman en 1996)

Classifiers built using bootstrap samples (with replacement)
Generates T subsamples of size n' (n' ≤n) from S (learning sample of size n)
Builds T classifiers and combines the outputs

Final decision: voting of all individual classifiers for classificatio
Average for regression

$$C^*(x) = \arg\max_{w_i \in C} \sum_{k:C_k(x)=w_i} 1$$ /* The most voted class*/



Avoids overfitting

Multiple classifier systems: boosting

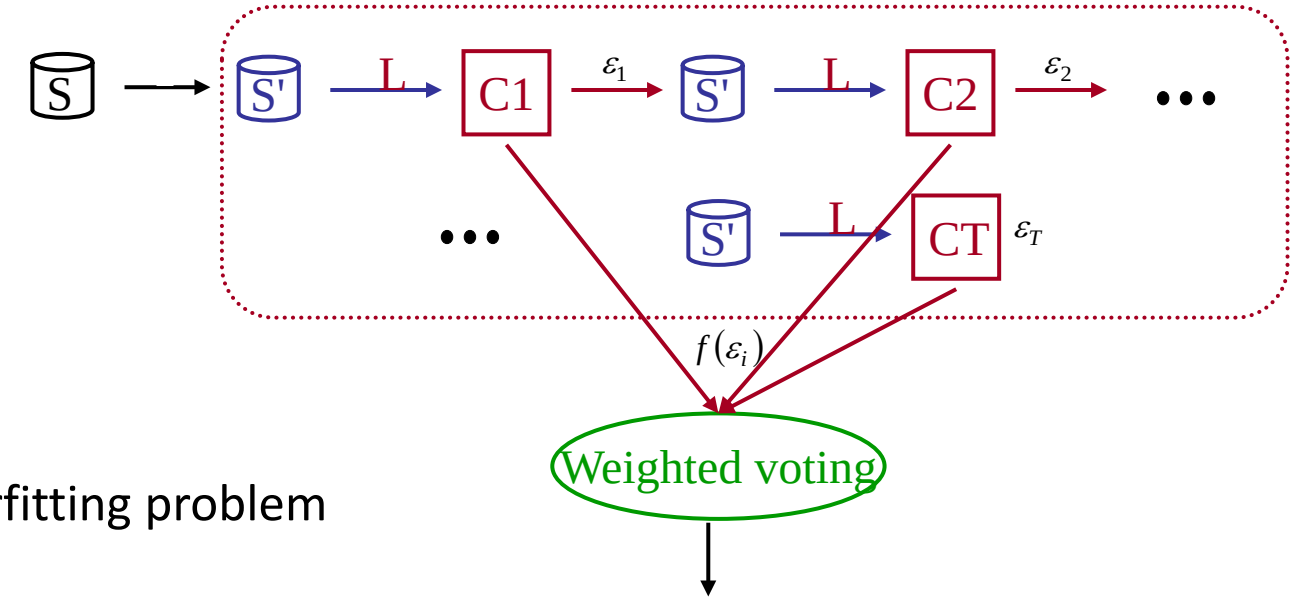
Boosting: proposed with the aim of stregthening of weak classifiers (Schapire 1990) AdaBoost (Adaptive Boosting) introduced in by 1996 Freund&Schapire.

T classifiers are built sequentially. Each of the instances in the sample has a weght which changes depending on whether its classification is correct or incorrect.

resampling vs *reweighting*

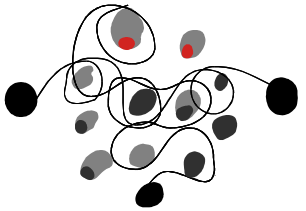
- Weight increments for incorrectly classified examples
- Weight decrements for correctly classified examples

Final decision: weighted voting



Overfitting problem

Multiple classifier systems: RSM



Random subespace method (RSM) (Ho 1995: *Random Decision Forests*)

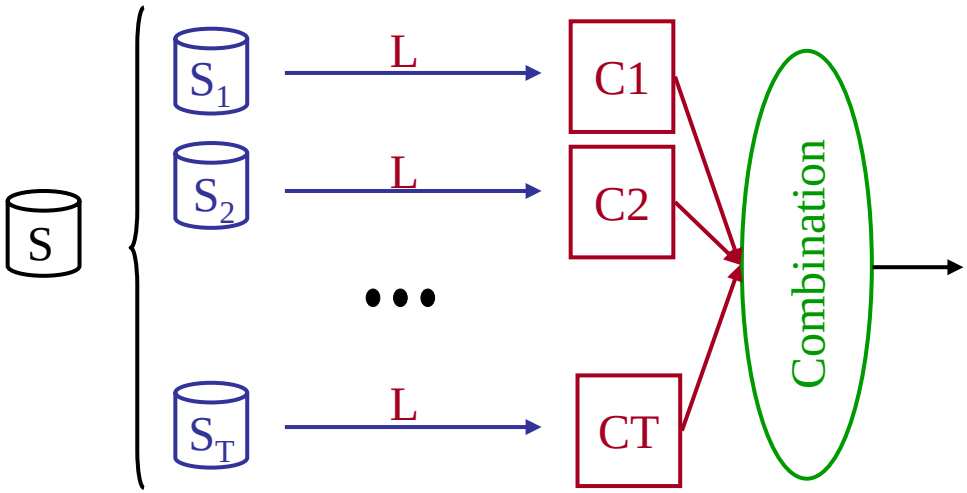
To build individual classifiers based on different subespaces

Subespace: classification space based on a subset of the original set of features

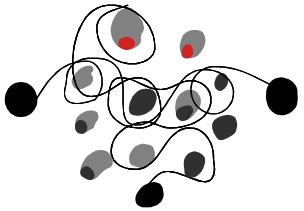
The methodology is applicable to any type ofclassifier

Final decision: average of class memebership probabilities produced by each individual classifier.

$$C^*(x) = \arg\max_{w_i \in C} \frac{1}{T} \sum_{j=1}^{T} \hat{P}_j(w_i \mid x)$$



Example



- *soybean.arff* (%66)
  - J48; RandomTree; IB1
  - AdaBoost_J48; RandomTree;
  - Bagging_J48;
  - Vote (J48+RandomTree); (IB1+J48+RandomTree)

	Basic	AdaBoost	Bagging	Vote
IB1				
J48				
RandomTree				
J48+RTree				
IB1+J48+RTree				