Data Mining Lab 1:

Data sets: characteristics, formats, repositories

Introduction to Rattle (R), Scikit-learn (Python) and Weka (Java)

I. Data sets

I.1. Data sets characteristics and formats

The data to be processed can be

- **structured** (e.g. data matrix, tables from relational databases) ← used during this lab
- semi-structured (e.g. XML files, web logs)
- unstructured (e.g. text documents)

Structured data:

- set of instances (records)
- each instance contains values corresponding to several attributes (features)
- the attributes can be of different types:
    - qualitative or categorical (their values are symbolic objects, e.g. symbols or strings):
        - nominal (e.g. nationality, gender, religion, marital status etc)
        - logical/ binary (e.g. presence or absence of a specific characteristic)
        - ordinal (e.g. level of satisfaction (“low”, “medium”, “high”), qualitative grade (“insufficient”, “sufficient”, ”good”, “very good”, “excellent”)
    - quantitative (their values are numbers taking values in discrete sets or continuous intervals)
        - integer (e.g. number of children, age in years, number of hits of a web page – usually values obtained by counting)
        - real (e.g. temperature, height, weight – usually values obtained by measuring)
- possible operations on the values of the attributes:
    - equality check, count of occurrences: nominal, logical and binary, ordinal, integer, real
    - comparison, ranking: ordinal, integer, real
    - comparison, ranking , addition, subtraction: integer, real (these are also called interval data)
    - comparison, ranking , addition, subtraction, multiplication, ratio: real (these are also called ratio data)

I.2. Data sets repositories

UCI Machine Learning repository (http://archive.ics.uci.edu/ml/)

- around 500 datasets grouped by categories
- a dataset usually contains files with the description of data (“names”) and csv files containing the instances (“data”)

Kaggle Platform (https://www.kaggle.com/)

- more than 29000 datasets (around 14000 datasets one year ago) of different sizes (from several hundreds of KB to more than on GB), various formats (csv, json, sqLite, BiqQuery), different domains
- solutions (kernels): Python notebooks, R scripts
- forum of discussions
- active competitions

KDD competitions (http://www.kdd.org/kdd-cup)

- data for annual Data Mining and Knowledge Discovery competitions organized by ACM

Exercise 1:

Select and analyze the characteristics (number of instances, number and types of attributes, tasks for which it has been used) of a data set from the UCI. Criteria to be used in the selection:

a. Size:
    - Small number of attributes of instances (e.g. Iris dataset)
    - Small number of attributes and a large number of instances (e.g. DBWorld emails dataset)
    - Large number of attributes and a large number of instances

b. Problem category:
    - Classification
    - Clustering
    - Regression

Exercise 2: Analysis of some Kaggle datasets:

a. Titanic data set (https://www.kaggle.com/c/titanic). Identify: number of instances, number and types of attributes. Examples of processing steps applied to the dataset
    - https://www.kaggle.com/hiteshp/head-start-for-data-scientist
      https://www.kaggle.com/vin1234/best-titanic-survival-prediction-for-beginners

b. Google jobs (https://www.kaggle.com/niyamatalmass/google-job-skills). What kind of knowledge could we extract from the dataset

c. Visit the current (and recent) competitions at Kaggle

II. Rattle = R Analytical Tool To Learn Easily =A Graphical User Interface for Data Mining using R (https://rattle.togaware.com/)

Rattle is a small platform, implemented in R (and based on existing R packages) used to learn how to apply basic data mining methods (without the need to search for the corresponding R package). It offers a simple to use GUI.

Rattle allows to:

- Load datasets in various formats (e.g. csv, dataset from R, arff (Attribute-Relation File Format - Weka standard) etc.)
- Attribute and instance selection
- Data exploration (e.g. summarization, vizualization)
- Data transformation
- Construct/ evaluate/ export models for classification, clustering, association analysis

Installation: `>install.packages("rattle")`

Usage:

```
>library(rattle)
>rattle()
```

Processing:

Data loading: Data ->fill Filename + upload -> Execute
Setting attributes: input/ target

Data exploration: Explore
Summary – statistical measues
Distributions – various visualization (BoxPlot, Histogram etc
Correlation – analysis of the correlation between attribues (based on correlation coefficients: Pearson, Spearman, Kendall)
Principal Components – identification of maximal variance directions

Test: statistical tests

Transform: scaling(Rescale) / fill in missing values (Impute) / type conversions (Recode) / instance removal (Cleanup)

Model: classification and regression models based on: decision trees (Tree), ensemble of decision trees (Forest), support vector machines (SVM), logistic regression (Linear), neural networks (Neural Net)

Cluster: data clustering – partitional algorithms (Kmeans), agglomerative hierarchical algorithms (Hierarchical), biclustering (BiCluster)

Associate: extraction of association rules

Remarks:

- After a processing task is selected and configured press Execute
- Some tasks requires the installation of R packages (Rattle will do this)
- All processing steps can be exported in R scripts which can be used later in R (see Log tab)

Exercise 3:

1. Load the Iris dataset in Rattle and analyze the summary:
   a. Number of instances
   b. Number of attributes
   c. For each attribute: possible values and frequency
2. Use View/Edit to visualize the data matrix and make changes

3. Use Explore to analyze:
   a. The distribution of values of each attribute in each class
   b. The correlation between the attributes; Which attributes are strongly correlated? How could be used this information?

Exercise 4: For the datasets from files carr.arff and autoMPG.arff analyze:

a. The types of the attributes

b. The distribution of data per classes
c. The distribution of attributes' values per classes
d. The correlation between the numerical / ordinal attributes (where it is possible)

III. Introduction to Scikit-learn (https://scikit-learn.org/)

Scikit-learn is an open-source Python package (which relies on numpy, scipy şi matplotlib) and offers a set of useful functions for data mining and machine learning

Installing (it requires numpy and scipy) – it can be done using pip or conda:

```
pip install -U scikit-learn
```

or

```
conda install scikit-learn
```

It contains various functions for:

- Data pre-processing:
  - Conversions (discretizations)
  - Transformations (scaling, standardization, normalization)
  - Feature extraction/engineering (e.g. Principal Component Analysis)
- Classification and Regression:
  - Decision Trees
  - Instance-based classifiers (e.g. k Nearest Neigbours)
  - Probabilistic classifiers (e.g. Naïve Bayes)
  - Classification and regression models relying on support vectors (Support Vector Machines)
  - Classification and regression models relying on neural networks(Multilayer Perceptron)
  - Meta-models (ensemble models)
- Clustering
  - Partitional methods (e.g. kMeans, fuzzy c-Means)
  - Hierarchical models (agglomerative algorithms)
  - Density based methods (e.g. DBSCAN)
  - Spectral methods
- Model evaluation and selection

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

IV.1. What is Weka?

WEKA = Waikato Environment for Knowledge Analysis

Free open-source software workbench developed at the Waikato University incorporating several implementations of machine learning algorithms to be used for different data mining tasks:

- Data visualization
- Data pre-processing (75 implemented algorithms)
- Attribute selection (around 25 implemented algorithms)
- Classification (more than 100 algorithm currently implemented)
- Clustering (around 20 implemented algorithms)
- Association rules

It is implemented in Java and it runs on: Windows, Linux, Mac

IV.2.How can it be installed?

Just download from http://www.cs.waikato.ac.nz/ml/weka/downloading.html
Last stable version: 3.8.* (currently 3.8.2)

IV.3.Which are the main components / ways of using Weka?

- **Graphical User Interface:**
  - **Explorer –** used to apply specific data mining tasks to datasets
  - **Experimenter –** used to conduct comparative analysis of different methods on different datasets
  - **KnowledgeFlow –** graphical interface used to define flows of tasks
  - **Workbench –** it subsumes all the major Weka GUIs
- **Simple CLI -** command line interface
- **Java API**

IV.3.1. Explorer:

- The data to be processed can be loaded:
  - From file (Open File) – common formats: arff (Weka format), csv (comma separated values)
  - From a web site (Open URL) or from a database (Open DB)
- Some examples of data could be generated randomly by using Generate
- The loaded dataset can be visualized as a data table and edited using Edit
- Categories of processing tasks:
  - Visualization
  - Data preprocessing
  - Attribute selection
  - Classification
  - Clustering
  - Association rules

IV.3.2. Experimenter:

- Allows the statistical comparison of several methods associated to a given task (e.g. classification) by using several data sets
- Provides the result of paired t-test (with corrections for multiple comparisons)

IV.3.3. Knowledge Flow:

- **Typical workflow:** “data source” ->”filter”->”classifier”->”evaluator”
- **Example**
  - **Arff loader -** data set connection to
  - **Cross validation foldMaker –** training set, test set connection to
  - **Naïve Bayes –** batch classifier connection to
  - **Classifier Performance Evaluation –** text connection to
  - **Text viewer**

  Remark: the connections are selected by right button click on the node icon
- **Flow activation:**

- Right button click on Arff and Start loading

IV.3.4. Command line interface:

- **Set the weka environment variable for java**
    - setenv WEKAHOME c:\Program Files\Weka-3-6
    - setenv CLASSPATH $WEKAHOME/weka.jar:$CLASSPATH
- **use a weka function**
    - java weka.classifiers.j48.J48 -t $WEKAHOME/data/iris.arff

IV.3.5. Java API:

- Add weka.jar to class path
- Example of using the J48 classifier

```
import weka.core.Instances;
import weka.classifiers.trees.J48;
...
Instances data = ... // from somewhere
String[] options = new String[1];
options[0] = "-U"; // unpruned tree
J48 tree = new J48(); // new instance of tree
tree.setOptions(options); // set the options
tree.buildClassifier(data); // build classifier
```

IV.3.6. ARFF data format – Attribute Relation File Format (this is the standard format of datasets to be processed in Weka)

- Header:
    - Comments (%)
    - Identifier of the data set: @relation dataset_name
    - List of attributes (each attribute is characterized by a name and a type): @attribute attr_name type
      Remark: in the case of discrete attributes (which do not have many values) the type is the set of possible values; in the case of discrete numeric attributes the type may be specified by integer; in the case of continuous numeric attributes the type is specified by real
- Data matrix:
    - specified by @data
    - each row contains a data instance; the values of the attributes are separated by comma

Example 1:

```
@relation car
@attribute buying {vhigh,high,med,low}
@attribute maint {vhigh,high,med,low}
@attribute doors {2,3,4,5more}
@attribute persons {2,4,more}
@attribute lug_boot {small,med,big}
@attribute safety {low,med,high}
@attribute class {unacc,acc,good,vgood}
@data
vhigh,vhigh,2,2,small,low,unacc
vhigh,vhigh,2,2,small,med,unacc
vhigh,vhigh,2,2,small,high,unacc
```

```
vhigh,vhigh,2,2,med,low,unacc
vhigh,vhigh,2,2,med,med,unacc
```

Example 2:

```
@relation 'autoPrice.names'
@attribute symboling real
@attribute normalized-losses real
@attribute wheel-base real
@attribute length real
@attribute width real
@attribute height real
@attribute curb-weight real
@attribute engine-size real
@attribute bore real
@attribute stroke real
@attribute compression-ratio real
@attribute horsepower real
@attribute peak-rpm real
@attribute city-mpg real
@attribute highway-mpg real
@attribute class real
@data
2,164,99.8,176.6,66.2,54.3,2337,109,3.19,3.4,10,102,5500,24,30,13950
2,164,99.4,176.6,66.4,54.3,2824,136,3.19,3.4,8,115,5500,18,22,17450
1,158,105.8,192.7,71.4,55.7,2844,136,3.19,3.4,8.5,110,5500,19,25,17710
1,158,105.8,192.7,71.4,55.9,3086,131,3.13,3.4,8.3,140,5500,17,20,23875
```

Exercise 5:

a. Load the Iris data set in Weka (by clicking Open file button) and analyze its summary:
    - Number of instances
    - Number of attributes
    - For each attribute: possible values and their corresponding counts

b. Use Edit to see the data matrix and to make changes on the data values.

c. Use Visualize All to see the distribution of the values of each of the attributes in different classes (the class attribute is implicitly the last attribute). Identify a pair of attributes which are able to discriminate the classes (e.g. (sepal width, petal length)). How can be intuitively estimated the discriminative ability?

d. Use the appropriate Weka filter to remove the attributes which are not considered relevant for the classification.

Exercise 6:

1. Follow the same steps for the car data set (car.arff)
   Remark: The main difference between iris and car datasets is the fact that the first one contains numerical attributes while the second one contains nominal/ordinal attributes.
2. Load the autoMpg.arff file (it contains information on cars and on their consumption (mpg=miles per gallon)) and analyze its content using Edit.

Remark 1: Most of attributes (including the class) are numeric; if the class attribute is numeric then the data are appropriate for prediction tasks (estimate the values of “miles per gallon” depending on the characteristics of the care).
Remark 2: The empty (gray) fields in the table correspond to missing values.

a) Use Visualize All to see which of the attributes is correlated with the mpg value

3. Load the supermarket.arff file (it contains data useful for market-basket analysis)

Exercise 7: compare the performance of several classification tools on two datasets (more details on the classification methods will be presented in Lecture 3 and Lab 3)

1. Select Experimenter from the starting Weka panel
2. Click New button to create a new experiment
3. Add datasets: iris.arff and breast-cancer.arff
4. Add algorithms: zeroR, oneR, naiveBayes, J48 (oneR and zeroR are from the “Rules” group, naiveBayes is from the “Bayes” group and J48 is from the “Tree” group)
5. Run the experiment
6. Analyze the results:
   a. Click on Experiment
   b. Perform the statistical test
   c. Interpret the results of the statistical test (using zeroR as reference method)
      i. v/ /* should be interpreted as: v= number of cases (datasets) on which the current method is better than the reference method; / /= number of cases (datasets) on which the current method is not significantly different than the reference method; *= number of cases (datasets) on which the current method is wors than the reference method;