Exercise Week 07 – Reference Solution

Sprachverarbeitung (VL + Ü)

Nils Reiter, nils.reiter@uni-koeln.de

May 9, 2023 (Summer term 2023)

1. Create an ARFF file from `titanic.csv`. For this, you need to specify the header with the data types, while the actual data set can remain as it is. You need to make a copy of the file into your own directory first.
   You can test your file by asking the class `weka.core.converters.ArffLoader` to load it:
   `java -cp /tools/weka/weka.jar weka.core.converters.ArffLoader FILE`
   - See below.
2. Alternatively: Use the file `credit.g.arff` to make Weka experiments
3. Train and evaluate a machine learning model with Weka. The simplest command to do that is `java -cp /tools/weka/weka.jar weka.classifiers.Evaluation weka.classifiers.bayes.NaiveBayes -t DATASET`. Please replace DATASET with the filename of your ARFF file. Inspect the output and verify that this is indeed predicting the survival of the passengers. If you leave out the dataset argument, available options are printed on the command line. Alternatively, you can also read about them in the API documentation: `https://weka.sourceforge.io/doc.stable-3-8/`
4. If you're certain Weka does what it's supposed to do, inspect the output of the evaluation and identify parts you don't understand.

Create titanic.arff from titanic.csv

1. There is no single, fully-automatic way to do this.
2. It's a good first step to express the data types of the features in Weka data types:
   - `@attribute PassengerId numeric`
   - `@attribute Survived {0,1}` – I.e., we don't express the feature as numeric, but as a nominal feature, because we want to do classification.
   - `@attribute Pclass numeric` – could also be made into a nominal feature
   - `@attribute Name string`
   - `@attribute Sex {male, female}`
   - `@attribute Age numeric`
   - `@attribute SibSp numeric`
   - `@attribute Parch numeric`
   - `@attribute Ticket string`
   - `@attribute Fare numeric`
   - `@attribute Cabin string`
   - `@attribute Embarked {S, C, Q}`

- Technically, it may be a good strategy to write this first into a separate file, because we may need to manipulate the data set as well. If we have this header in a file called `titanic_header.arff`, we could generate the final file using a pipe: `cat titanic_header.arff titanic.csv > titanic.arff`.

3. We will need to do a few things on the data itself (in file `titanic.csv`):
   - We need to remove the head line. This can be done manually, with a text editor.
   - We need to remove the double double quotes and commas within the name strings.
     - Commas:
       `sed -i -E "s/,\"([[:alpha:] '-]+?), ([[:alpha:]. ()\/-]+?)\",/,\1 \2,/g" titanic.csv`
       (-i allows inline change, i.e., without writing into a different file first)
     - Double double quotes: `sed -i 's/""//g'`
   - Some feature values are missing. Missing values in Weka are to be represented as a single question mark. I.e., we replace all occurrences of `,,` (i.e., two commas) with `,?,`.
     - `sed -i 's/,,/,?,/g' titanic.csv`
     - To take care of missing last values: `sed -i -E 's/,[^,]*$/,?/g' titanic.csv`
   - The ticket id does not seem to follow any logical pattern, and sometimes contains spaces. Strings with spaces need to be enclosed in double quotes in ARFF. We can fix this using a regular expression that is anchored at the end, and identifies the fourth-last column.
     - `sed -i -E 's/,([^,]*?)(,[^,]*){3}$/,"\1"\2/g' titanic.csv`
   - Finally, the cabin feature sometimes also contains spaces. Apparently, some people have used more than one column. I.e., we need to enclose this column in quotes as well.
     - `sed -i -E 's/,([^,?]+?),([^,]*?)$/,"\1",\2/g titanic.csv`
   - After assembling the final arff file, we can load it in Weka.