Data Mining II 2016/2017
Rita P. Ribeiro DCC - FCUP

Exercises on Sequence Mining and Outlier Detection

Sequence Mining

Start R and load the package: `arulesSequences`

1. Load the transactions dataset `zaki`.

(a) Inspect the content of `zaki`.

(b) Mine the frequent sequences obtained from `zaki`. Use the function `cspade()` [1] for that purpose and set the minimum support to 0.4. What is the class of the obtained result? How many sequences were obtained?

(c) Use the function `summary()` over the obtained sequences. What information does it give to you?

(d) Use the function `as()` to coerce the result to a `data.frame`.

Outlier Detection

Load the following packages: `e1071`, `DMwR2`, `dbscan`, `UBL`

Unsupervised Approaches

2. Load the dataset `iris`, remove the attribute `Species` and apply the following clustering techniques.

(a) Start by applying the clustering algorithm DBSCAN, available through the function `dbscan()` of the package `dbscan`. Use `eps=.4`. Interpret the obtained result.

(b) Vary the values for the parameters `eps` and `minPts` and see the impact it has on the result.

(c) Regarding the first clustering model, obtain the index of the observations identified as outliers.

(d) Use the function `pairs` over the dataset with the color based on the index given by the cluster assigned to each observation to see where the identified outliers are. (Alternatively, you can also explore the function `fviz_cluster()` from the package `factoextra`)

(e) Run the LOF algorithm, available through the function `lof()` of the package `dbscan`. Use `k=5`. Interpret the obtained result.

(f) Select the top N outliers identified by LOF where N is the number of outliers identified by the DBSCAN. Are there any common observations?

(g) Run the OR_H algorithm, available through the function `outliers.ranking()` of the package `DMwR2`. Interpret the obtained result.

[1] SPADE is an Apriori-based approach to mine frequent sequences. More details available in https://link.springer.com/article/10.1023/A:1007652502315

(h) Select the top N outliers identified by OR_H where N is the number of outliers identified by the DBSCAN. Are there any common observations?

(i) Obtain the observations that are identified as outliers by the three approaches.

Semi-supervised Approaches

3. Load the dataset `iris`. Filter all the observations of the Species `"setosa"` to a data frame called `irisNormal` and apply the following semi-supervised technique.

(a) From the dataset `irisNormal`, randomly select 70% of the observations for training and 30% for testing.

(b) Obtain the one-class svm model on the training data, using the function `svm()` of the package `e1071`, with `type='one-classification'`.

(c) Use the obtained model to get the predictions over the training data.

(d) Use the function `table()` to get the confusion matrix with the predictions made by the model and the true values of the Species information.

(e) Obtain the confusion matrix for the test dataset.

(f) Obtain the confusion matrix considering, now, all the instances of iris dataset which were not given to the one-class svm algorithm. What can you conclude from the results?

Supervised Approaches

4. Load the dataset `ImbC` from the package `UBL` and apply the following pre-processing techniques.

(a) Start by inspecting the distribution of classes in the dataset.

(b) Remove from the dataset all the observations from class `"rare1"`. (We now have a binary-classification problem, where the positive classe is the minority class `"rare2"`.)

(c) From the previous dataset, randomly select 70% of the observations for training and 30% for testing.

(d) Build an svm model over the training data and obtain the confusion matrix with the test data. Evaluate the performance of model by $accuracy = (TP + TN)/(TP + FN + TN + FP)$, $precision = TP/(TP + FP)$ and $recall = TP/(TP + FN)$. What can you conclude?

(e) From the original training data, obtain a new training dataset using the function `RandUnderClassif()` with its default parameters. Inspect the distribution of classes on the new dataset.

(f) Build an svm model over the new training data and obtain the confusion matrix with the test data. Evaluate the performance of the model.

(g) From the original training data, obtain a new training dataset using the function `RandOverClassif()` with its default parameters. Inspect the distribution of classes on the new dataset.

(h) Build an svm model over the new training data and obtain the confusion matrix with the test data. Evaluate the performance of the model.

(i) From the original training data, obtain a new training dataset using the function `SmoteClassif()` with its default parameters, but with `dist='HEOM'`, so that it can compute the distance between nominal attributes. Inspect the distribution of classes on the new dataset.

(j) Build an svm model over the new training data and obtain the confusion matrix with the test data. Evaluate the performance of the model.