Indian Institute of Technology Delhi

MTL782 Data Mining: Assignment

Deepak: 2019MT10685

Aakash Goel: 2019MT10668

Question 1

In this part, we have performed multi class classification using various classification techniques such as Decision Tree, Random Forest, Naive Bayes Classifier and KNN Classifier to classify the bitcoin transaction into different types of Ransomware or white(non-ransomware).

Data Description

We have used the Bitcoin Heist Ransomware Address Dataset Data Set from UCL Machine Learning Repository. This dataset contains the entire Bitcoin transaction graph (using a time interval of 24 hour and network edges greater than 80.3) from 2009 January to 2018 December. The dataset has a total of 29,16,697 instances and 10 attributes: address(String, Bitcoin address), year(Integer.Year), day(Integer,Day of the year: 1 is the first day, 365 is the last day), length(Integer), weight(Float), count(Integer), looped(Integer), neighbors(Integer), income(Integer. Satoshi amount), label(String, Name of the ransomware family (e.g., Cryptxxx, cryptolocker etc) or white (i.e., not known to be ransomware)).

Benefits we hope to get from Data Mining

- We hope to analyse this dataset and find patterns that may allow us to flag certain transactions as being ransomware and classify it into different ransomware families.
- This may aid law enforcement authorities to detect fraud and prevent cyber crime.
- Apply the data mining techniques for computing the best trained model with less computation cost and high testing accuracy.

Problems with the data

- The dataset has several redundant features which we need to drop of before training otherwise it would cause over fitting of model.
- Names and string data is not good for modelling purposes and needs to encoded for better efficiency.
- Attributes in the data are measured at different scales and therefore they may not contribute equally to the model fitting model learned function and this might end up creating a bias.

Appropriate response to quality Issues

- We have done feature selection to remove the redundant features(contain no information that is useful for the data mining task at hand) such as the attribute address in the dataset.
- We have used label encoding to handle the catagorical features.
- We have used Feature scaling (min max) to overcome the issue of variables being measued at different scales

Pre-processing the Data

We have applied several pre-processing techniques before feeding the data into the classifier

- **Handling missing values or duplicates :** The data did not contain any missing values or duplicates.
- **Normalisation and Feature Scaling :** It is essential to normalise or scale the data before feeding it into the classifier because without scaling, large values can introduce bias in the results. Thus the features are usually scaled down to the range [-1,1] or [0,1]. We used the normalise method from the preprocessing library of sklearn and applied MinMax scaling on top of it.Thus, the range off feature values was first brought down to [-1,1] and then to [0,1]
- **Label Encoding :** Inorder to handle the categorical features, we have used label encoding technique to convert them to integral values using the label encoder of sklearn preprocessing.
- **Feature Subset selection :** Since the address attribute was irrelevant to the type of Ransomware and including it would have caused overfitting of the model. Therefore, we have dropped this feature
- **K-fold Cross validation :** We have used the k-fold cross validation technique to split our data into training and test data and computed the model accuracy.

Decision Tree Classifier

Decision Tree is a Supervised Machine Learning Algorithm that uses a set of rules to make decisions. On using decision tree classifier to classify the bitcoin transactions, the training error accuracy was much greater as compared to testing accuracy. Therefore, overfitting has occurred.
Training Accuracy = 99.96 %
Testing Accuracy = 93.53 %

Random Forest Classifier

Random forests or random decision forests is an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time. For classifying Bitcoin transactions, optimal tuned random forest classifier has a maximum depth of 3 nodes and 1 random state
Training Accuracy = 98.51 %
Testing Accuracy = 98.53 %

Naive Bayes Classifier

Naive Bayes is a simple technique for constructing classifiers: models that assign class labels to problem instances, represented as vectors of feature values, where the class labels are drawn from some finite set. We have used the Multinomial Naive Bayes classifier to classify the bitcoin transactions.
Training Accuracy = 98.58 %
Testing Accuracy = 98.58 %

KNN Classifier

KNN is a type of classification technique where the function is only approximated locally using the information obtained from its nearest neighbours. For classifying Bitcoin transactions,

optimal tuned KNN classifier works with 3 nearest neighbours.
Training Accuracy = 97.28 %
Testing Accuracy = 96.16 %

Question 2

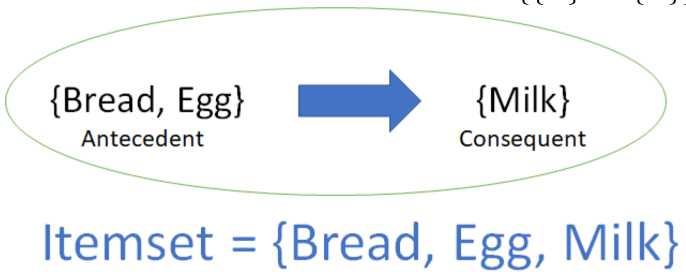
In the second question, we look for patterns in a transactional database, such as retail shopping statistics, to find correlation between different items.

Association Rule Mining

Association Rule Mining is one of the ways to find patterns in data. It finds:

- features (dimensions) which occur together
- features (dimensions) which are "correlated"

Apriori algorithm and FP-Growth algorithm are two of the methods to mine frequent itemsets and divise association rules of the form {{X} -> {Y}} from a given transactional database.



The measures of effectiveness of the rule are:

- Support : This measure gives an idea of how frequent an itemset is in all the transactions.

$$Support(\{X\} \rightarrow \{Y\}) = \frac{Transactions\ containing\ both\ X\ and\ Y}{Total\ number\ of\ transactions}$$

- Confidence : This measure defines the likeliness of occurrence of consequent on the cart given that the cart already has the antecedents.

$$Confidence(\{X\} \rightarrow \{Y\}) = \frac{Transactions\ containing\ both\ X\ and\ Y}{Transactions\ containing\ X}$$

- Lift : This is the rise in probability of having {Y} on the cart with the knowledge of {X} being present over the probability of having {Y} on the cart without any knowledge about presence of {X}.

$$Lift(\{X\} \rightarrow \{Y\}) = \frac{(Transactions\ containing\ both\ X\ and\ Y)/(Transactions\ containing\ X)}{Fraction\ of\ transactions\ containing\ Y}$$

Data Description

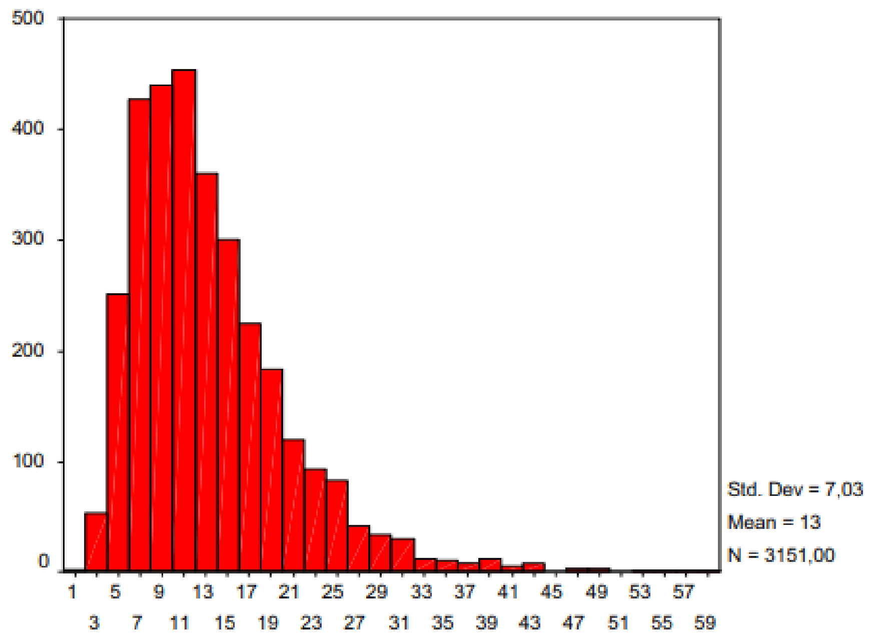
The association models are trained using the dataset from http://fimi.uantwerpen.be/data/retail.dat, which contains the (anonymized) retail market basket data from an anonymous Belgian retail store.

The data are provided 'as is'.

The dataset contains approximately 5 months of data. The total amount of receipts being collected equals 88,163.

Figure 1 shows the average number of distinct items purchased per shopping visit. The average number of distinct items (i.e. different products) purchased per shopping visit equals 13 and most customers buy between 7 and 11 items per shopping visit.

Figure 1: Average number of distinct items purchased per visit

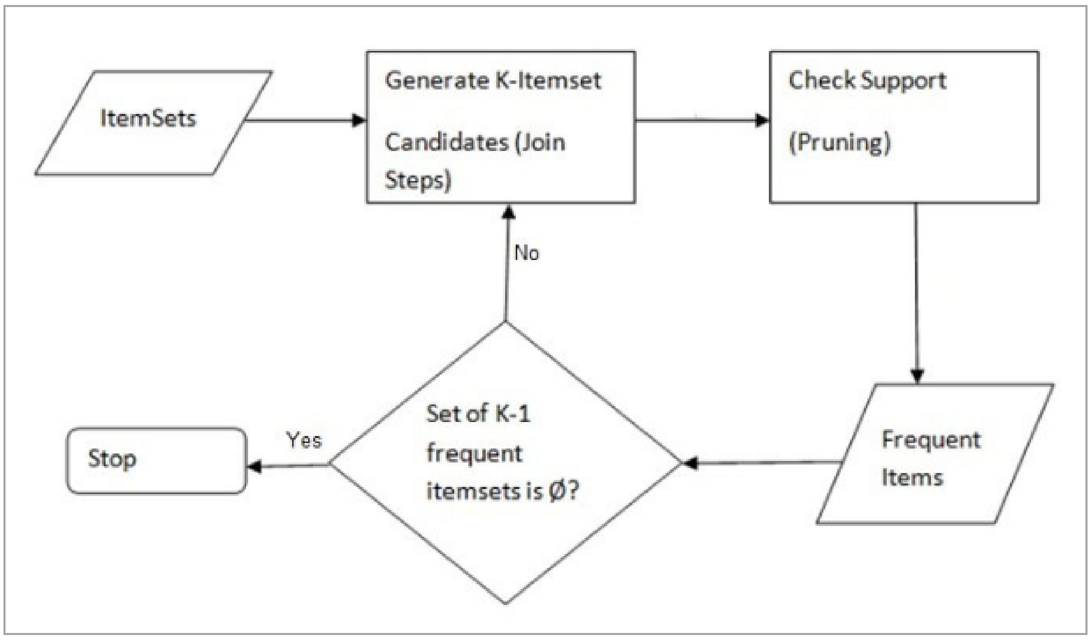


The data has already been pre-processed into labelled records, and can be fed directly to the frequency rule generator.

Apriori

The following are the main steps of the apriori algorithm:

- Calculate the support of item sets (of size k = 1) in the transactional database (note that support is the frequency of occurrence of an itemset). This is called generating the candidate set.
- Prune the candidate set by eliminating items with a support less than the given threshold.
- Join the frequent itemsets to form sets of size k + 1, and repeat the above sets until no more itemsets can be formed. This will happen when the set(s) formed have a support less than the given support.


ItemSets
Generate K-Itemset Candidates (Join Steps)
Check Support (Pruning)
No
Set of K-1 frequent itemsets is Ø?
Yes
Stop
Frequent Items

Modifications

- Transaction reduction: Traditional Apriori algorithm involves the generation of many candidate item sets consisting of many infrequent and unnecessary item sets, and a large number of combinations. The algorithm was modified to disregard itemsets with frequency less than a threshold using minimum support and minimum confidence criteria, and only most frequent itemsets were considered for further rule generation, greatly increasing the efficiency of the algorithm.
- Sampling: The algorithm when run on randomly selected sample of the entire database gave similar results, as itemsets frequent in the entire database are likely to be frequent in the sample as well.

FP Growth

Frequent Pattern Tree is a tree-like structure that is made with the initial itemsets of the database. The purpose of the FP tree is to mine the most frequent pattern. Each node of the FP tree represents an item of the itemset.

The root node represents null while the lower nodes represent the itemsets. The association of the nodes with the lower nodes that is the itemsets with the other itemsets are maintained while forming the tree.

Running the algorithm

The functions apriori and fpgrowth each takes in the dataset and thresholds for support and confidence as parameters and returns the most frequent itemsets along with the association rules derived from those itemsets.

For hyperparameters min_support and min_confidence as 0.01 and 0.2 respectively, frequency sets having upto 5 unique items were discovered satisfying the bounds of support and confidence.

FP Growth was found to be much faster (by upto 20 times for the given dataset) even after making modifications to the apriori algorithm. This is most likely because the construction of fp-tree requires only one pass through the database as compared to multiple scans for apriori and is thus more efficient.

```
Scan  1
```

```
Size of freq set:  78
Scan  2
Size of freq set:  80
Scan  3
Size of freq set:  46
Scan  4
Size of freq set:  12
Scan  5
Size of freq set:  1
Scan  6
Size of freq set:  0
```