Final Report

Big Data Platforms & Technologies (D0S06a)

Professor Dr. Seppe vanden Broucke

May 28th, 2017

By Group 8:

Halyna Aleksandrovych (r0648181)

Kexin Huang (r0644406)

Yilin Jiang (r0644880)

Jia Song (r0648524)

Qiaoyubing Sun (r0648527)

Table of Contents

ASSIGNMENT 1

Paper Review—Isolation Forest

1. Paper summary

The paper proposes an anomaly detection method, iForest, that focuses on anomaly isolation rather than normal instance profiling, which is distinguished from existing model-based, distance-based or density based methods. Two concepts are presented: isolation tree (iTree) and path length. Taking advantage of anomalies' nature of 'few and different', iTree isolates anomalies closer to the root of the tree as compared to normal points. Isolation forest builds an ensemble of iTrees for a given dataset and classifies those instances with short average path lengths on the iTree as likely anomalies. The paper shows that iForest's detection performance converges quickly with a very small number of trees, and it only requires a small sub-sampling size to achieve high detection performance with high efficiency. On the other hand, the isolation characteristic of iTrees enables iForest to build partial models and employ only a tiny proportion of training data to build effective models. Through a significantly reduced subsample, iForest has got superior anomaly detection ability in handling swamping and masking. At the same time, iForest has a linear time complexity with a low constant and a low memory requirement which is ideal for high volume data sets.

Four core findings are shown based on the empirical evaluation. First, iForest outperforms ORCA, LOF and RF in terms of AUC and execution time, especially in large dataset. Second, iForest converges quickly with a small ensemble size, which enables it to detect anomalies with high efficiency. Third, for high dimensional problems that contain a large number of irrelevant attributes, iForest can achieve high detection performance quickly with a simple addition of Kurtosis test. Fourth, iForest works well when training set contains only normal instances.

2. Application setting

Given the characteristics of iForest, such as, linear time complexity with a low constant, a low memory requirement, high efficiency and high accuracy especially for large databases, it could be used to detect anomalies in a variety of application areas. Some possible fields are listed below:

- Detect financial fraud (e.g. , detect fraudulent use of credit cards when there are anomalies in transactions)
- In astronomy (e.g., discover a new star or exotic particles in a set of high-energy collisions)
- Detect disease: abnormal patient condition could indicate a disease outbreak in a specific area (e.g., in biopsy data, this algorithm can also be used to detect malignant biopsies)
- Maintain network security (e.g., detect hackings in a set of network events)
- Marketing (e.g., detect churn which could be seen as anomaly clusters and develop targeted marketing campaigns)
- Word processing (e.g., detect structural defects which may point to a systematic problem)

3. Review

Isolation forest has got a lot of advantages compared to existing anomaly detection methods. First, it has a linear time complexity with a low constant and a low memory requirement. This means that it can perform analysis without the need for specialized or extra hardware. Or one can easily host an online anomaly detection system with a minimal memory footprint. Second, compared to existing model-based, distance-based or density-based methods, it has got high accuracy and efficiency, especially when working with a large volume of data. Third, with an additional attribute selector, it could achieve high detection performance in the case of high dimensional data that contains a large number irrelevant attributes. Fourth, it could work well even when no anomalies are present in the training set. Fifth, the method is easy to understand and learn. So for a beginner, it is easy to get started with.

There are also several disadvantages. First, as this algorithm uses global ranking measures based on average path length of each instances, it is proved to be effective in detecting global anomalies. But it fails to detect local anomalies in data sets having multiple clusters of normal instances, because the local anomalies are masked by normal clusters of similar density and thus they become less susceptible to be isolated (Aryal et al., 2014). Second, iForest might fail at high dimensional data with correlated attributes. In the case of using kurtosis as the attribute selector, the ranking might be meaningless and therefore the attribute subspace selection could not proceed. Third, once anomalies are identified, it does not provide much description regarding the way in which an instance is an outlier. For example, it does not immediately inform whether an anomaly is positive or negative, or if two outliers are similar or not.

4. Expansion

One weakness of this method is that it does not provide a measure of similarity for the identified outliers. While residual-based measures can indicate if an outlier scored low or high and on which variables, providing some information, the iForest method does not. To expand upon this technique, we might borrow the concepts from network analysis. For example, one potential solution would be to quantify the number of common paths on iTrees a given instance share with others. For example, if two outliers have no path in common, they could be judged to be more dissimilar than two outliers that split off from one another at the very last step. This would help to label anomaly clusters. For example, with this additional function, we may identify if two financial fraudulent cases are done in the same manner or if two abnormal patient conditions refer to the same disease.

Moreover, in order to overcome the weakness in local anomalies detection, Aryal et al.(2014) suggested to take local data distribution into consideration and replace the global ranking measure based on path length with a local ranking measure based on relative mass to help improve the approach.

Reference

Aryal S., Ting K.M., Wells J.R., Washio T. (2014). Improving iForest with Relative Mass. In: Tseng V.S., Ho T.B., Zhou Z.H., Chen A.L.P., Kao H.Y. (eds). Advances in Knowledge Discovery and Data Mining. PAKDD 2014. Lecture Notes in Computer Science, vol 8444. Springer, Cham.

ASSIGNMENT 2

Predictive Modelling

1. Introduction

The task is to construct a predictive model in order to predict whether customers will churn or not based on the churn dataset. To deal with the class imbalance problem, three popular methods including over-sampling, under-sampling and both-sampling are applied here. Regarding the predictive technique, we use decision tree, logistic regression and random forest. To compare the predicting result, AUC is the main metric besides accuracy and recall.

2. Descriptive statistics and data pre-process

There are 5000 observations and 18 variables in the dataset. The focus of interest is to predict the whether a customer will churn or not, especially capturing the characteristics of those who decide to churn. Therefore, the response variable Churn is of a binary class with 1 denoting churn and 0 denoting non-churn. 17 candidate explanatory variables are available, including usage information of calls, minutes and charge for day, evening, night and international aspects, respectively. Other possible predictors contain area code, whether or not having an international plan and voice mail. Before modelling, some data pre-processes are required and details see below.

2.1. Correlation plot and check for multicollinearity

The correlation plot for the full set of variables is given in Figure 1. It’s obvious that 4 pairs of _charge and _mins variables are highly correlated with correlation coefficients nearly 1. This coincides with the situation in practice where we all know, in the telco industry, the charge is roughly a linear function of the usage minutes.

Besides, variable Vmail and Vmail_message also have a high correlation coefficient of -0.954. Looking in detail we find that, observations with variable Vmail value 1 always have variable Vmail_message equal to 0 without exception. This indicates that variables Vmail and Vmail_ message probably have a huge overlap of information contained, hence one of them should be considered to drop out when multicollinearity is an issue in that case.

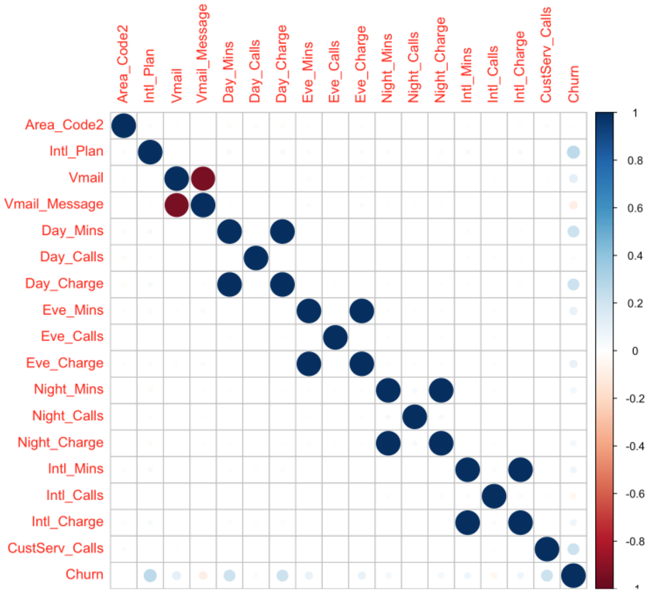


Figure 1 Correlation plot

To further examine the multicollinearity problem, we compute the Variance Inflation Factor (VIF) for each of these candidate predictors. Normally the VIF value for single variable and the average should be around 1. The bigger the value of VIF, a stronger indication for highly correlation. The results are given in Table 1, where three situations are considered:

- Situation A: The full set of variables
- Situation B: deleting all _charge variables and variable Vmail
- Situation C: deleting all _mins variables and variable Vmail

The reason why dropping out variable Vmail rather than Vmail_ message is that variable Vmail _message appears to be more informative because it is a count data while variable Vmail is only a binary indicator.

Table 1 Variance Inflation Factor (VIF) for each candidate predictor

	Area_code2	***Intl_Plan***	***Vmail***	***Vmail_message***	***Day_Mins***	***Day_Calls***
A	*1.00*	*1.00*	*11.28*	*11.27*	*10216572.71*	*1.00*
B	*1.00*	*1.08*	*-*	*1.01*	*1.05*	*1.00*
C	*1.00*	*1.08*	*-*	*1.01*	*-*	*1.00*
	Day_Charge	***Eve_Mins***	***Eve_Calls***	***Eve_Charge***	***Night_Mins***	***Night_Calls***
A	*10216577.85*	*2225603.69*	*1.00*	*2225606.58*	*631941.78*	*1.00*
B	*-*	*1.01*	*1.00*	*-*	*1.01*	*1.00*
C	*1.05*	*-*	*1.00*	*1.01*	*-*	*1.00*
	Night_Charge	***Intl_Mins***	***Intl_Calls***	***Intl_Charge***	***CustServe_Calls***	***Average***
A	*631939.97*	*68264.61*	*1.00*	*68264.73*	*1.00*	*1546165*
B	*-*	*1.01*	*1.00*	*-*	*1.06*	*1.03*
C	*1.01*	*-*	*1.00*	*1.01*	*1.06*	*1.03*

In the full set (situation A), the mean value of VIF is extremely high and this is due to the fact that charge and number of minutes have approximately a linear relation, which a strong indication for the existence of multicollinearity issue. Therefore, in situation B and C, we respectively deleting _charge variables and _mins variables. The resultant average VIF value 1.03 suggests that the multicollinearity issue is solved.

In the following analysis, only in the logistic regression part will we consider the multicollinearity issue and do variable reduction to solve it. In the decision tree and random forest framework, the designs make them less susceptible to multicollinearity, hence we use the full set of covariates.

2.2. Transformation of nominal variables

There are four categorical/nominal variables in this dataset. Table 2 gives the summary of these variables and their corresponding levels.

Table 2 Categorical variables and corresponding levels

Categorical variable	***Level***
Churn (Churn)	*1 = churn; 0 = non-churn;*
Area code (Area_Code2)	*0 = district 0; 1 = district 1; 2 = district 2;*
International plan (Intl_Plan)	*1 = have an international plan; 0 = not have;*
Voice mail (Vmail)	*1 = not use voice mail; 0 = use voice mail;*

In the original dataset, these variables are all in numeric format. To avoid confusion in identification when building the model in R, we use factor() function to convert them into categorical form.

2.3. Create train-test split

To objectively measure the performance of the model we build, a split between training data and test data is needed. In this case, since we have 5000 observations which is quite large, we apply a common 70:30 splitting rule to guarantee that both training and test set are large enough to be valid. To be specific, we randomly assign 70 percent of the original observations into the training dataset while the rest used as hold-out test set.

To ensure each time when running the program we get the same result, a set.seed (947) command is used to fix this randomization. Therefore, in the resulting training set there are 3501 observations while the test set has 1499 observations. For the following model building process, only training set will be used.

2.4. Dealing with class imbalance problem

However, there is a class imbalance problem when checking the number of churners and non-churners in the training set: 496 vs 3006. The latter is approximately 6 times of the former. In such circumstance, an algorithm possibly tends to be lazy and predict all the observations as non-churner to achieve a so-called high accuracy. Apparently, this is not what we want, since our focus of interest is the churners rather than non-churners. Such a lazy algorithm shows little predictive power for the churners.

Therefore, to deal with such problem, two things can be done: first, use AUC rather than accuracy as the evaluation metric; second, use sampling techniques to make the two classes be roughly balanced. Common sampling skills include over-sampling (increase the minority), under-sampling (decrease the majority) and both-sampling (increase minority and decrease majority). The results of the three approaches are shown in Table 3. Again, to make the result reproducible, set.seed() commands are used respectively before the sampling commands.

Table 3 Results of sampling to deal with class imbalance problem

Sampling method	*Number of churner (1)*	*Number of non-churner (0)*
Over-sampling	*3006*	*3006*
Under-sampling	*495*	*495*
Both-sampling	*1442*	*1558*

For over-sampling, the number of minority is lift to be the same as the majority, therefore 3006 observations in each class. For under-sampling, an inverse process is done, resulting 495 observations in each class. For both-sampling, randomization results in unequal but close number in each class.

In the following analysis, we will try all three sampling mechanisms in the decision tree model and choose the best one to do logistic regression and random forest.

2.5. Cross-validation set-up

For the logistic regression part, a three-time repeat of 10-kold cross-validation is used to choose the best set of parameters. This is done inside the training data and details can be seen from the code. Considering the complexity of the work, the decision tree does not involve cross-validation, instead, the model is built based on the whole training set. For random forests part, two functions (randomForest() and caret::train()) are used to construct the model. The former does not involve cross-validation while the latter is performed based on a 10-fold cross-validation. See details in the code. The following content regarding to random forests model is the result of randomForest() function. Both functions reach similar output.

3. Method 1: Decision tree

To construct the decision tree, two popular algorithms are applied here: rpart and C5.0.

3.1. Algorithm 1: recursive partitioning and regression tree (rpart)

First, we run the rpart algorithm on the three different newly generated training sets resulting from over-sampling, under-sampling and both-sampling. Compared the accuracy, recall and AUC calculated on the test data, the classification tree combined with the both-sampling method is the best. Then, according to the cross-validated error result, we select the complexity parameter 0.016 (see Figure 2) and begin to prune the so far optimal tree. The pruned tree, given in Figure 3, has a size of 9, AUC = 0.890, precision = 0.59, recall = 0.78, which is not much different from the unpruned tree (AUC = 0.893, precision = 0.57, recall = 0.83).

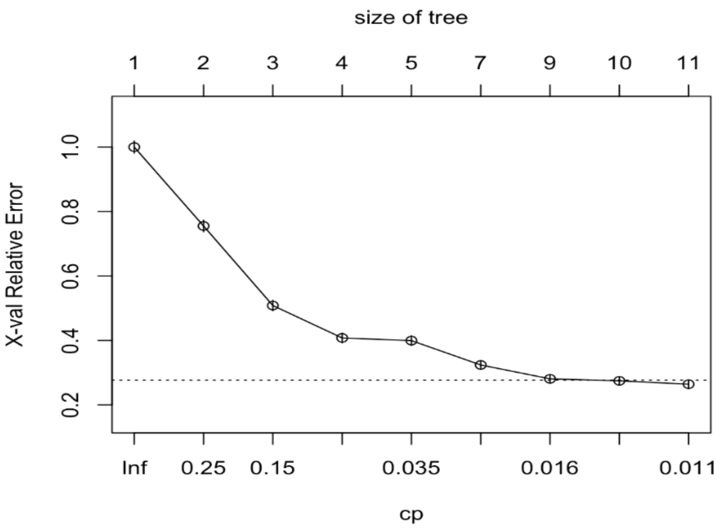


Figure 2 Complexity parameter plot

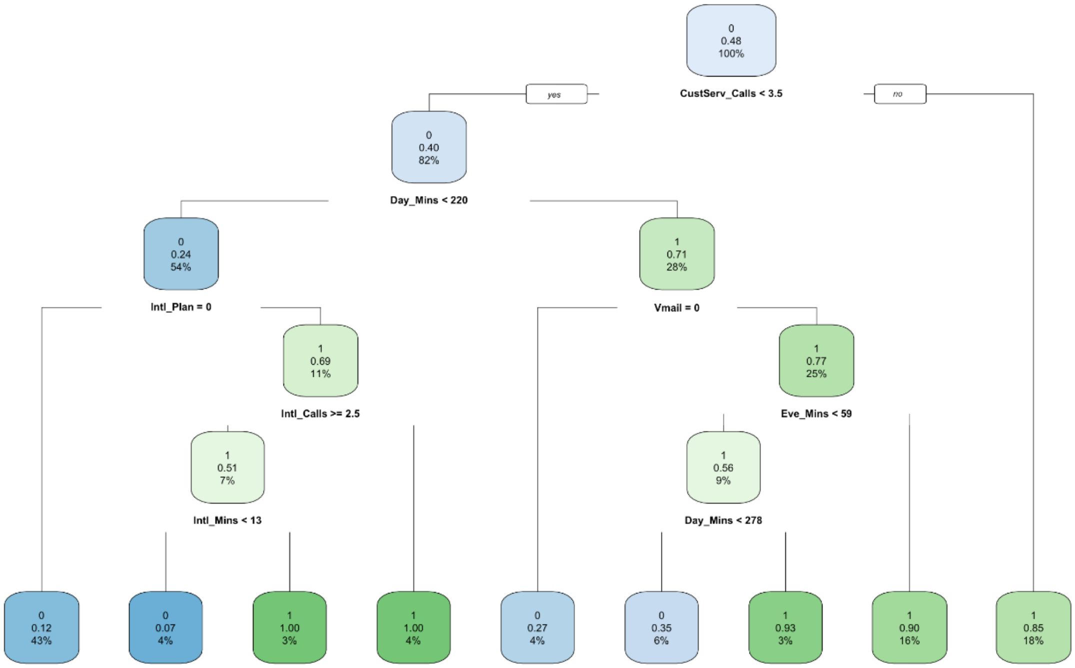


Figure 3 Pruned tree based on rpart and both-sampling

3.2. Algorithm 2: C5.0

Next, we run the C5.0 algorithm. The tree combined with the over-sampling has the highest AUC of 0.891. However, the tree size is 165. Thus, pruning and early stopping are needed here. After trying different settings, we find constraining the minimal cases in at least two of the splits to be 100 gives out the comparatively ideal result, which makes it comparable to the result of the algorithm rpart. The pruned tree, shown in Figure 4, has a size of 9, error rate = 12.3%, accuracy = 0.893, recall = 0.856, AUC = 0.879.

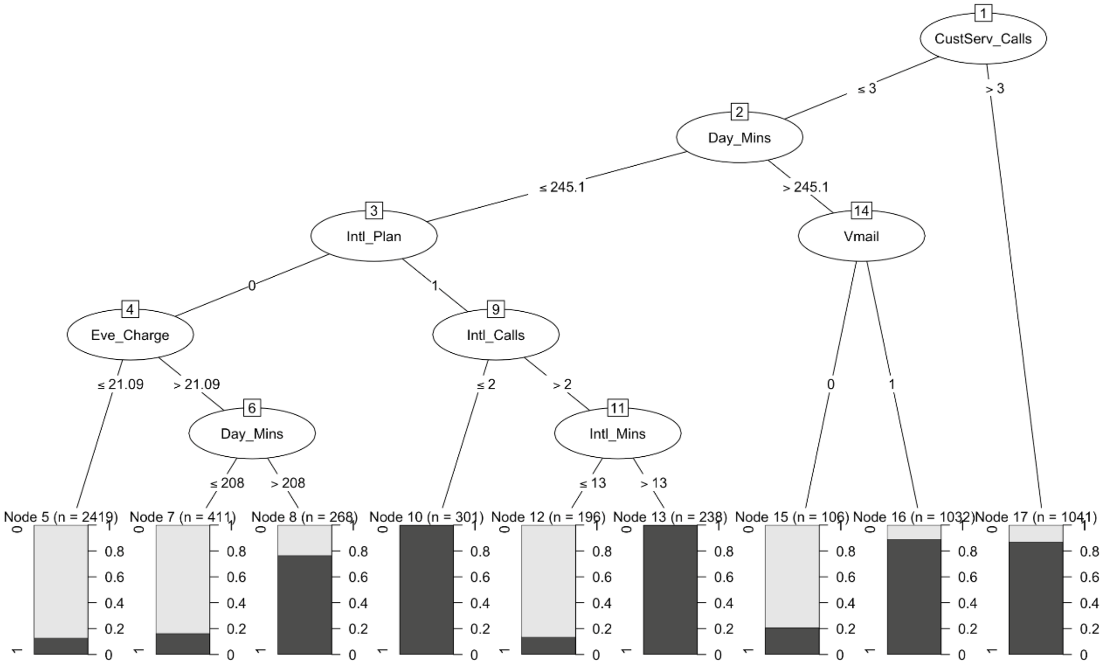


Figure 4 Pruned tree based on C5.0 and over-sampling

3.3. Summary for results of algorithm 1 and algorithm 2

Table 4 gives the summary for the evaluation metrics of two optimal models selected by the two algorithms, respectively. It’s obvious that the optimal model selected by C5.0 has an overwhelming advantage over the model selected by rpart in the accuracy, also an improvement in the recall. As for AUC, both models performed equally well.

Table 4 Accuracy, recall and AUC of optimal models by two algorithms

	Algorithm rpart	***Algorithm C5.0***
Sampling type	*Both-sampling*	*Over-sampling*
Accuracy	*0.589*	*0.893*
Recall	*0.783*	*0.856*
AUC	*0.890*	*0.879*

Since predicting churn is much more important than predicting non-churn in this setting. Given overall consideration, the pruned tree generated by C5.0 algorithm combining with over-sampling is the optimal one. Hence, the over-sampling will be used in the following logistic regression and random forest model. Figure 5 is the ROC curves generated by different combinations of methods.

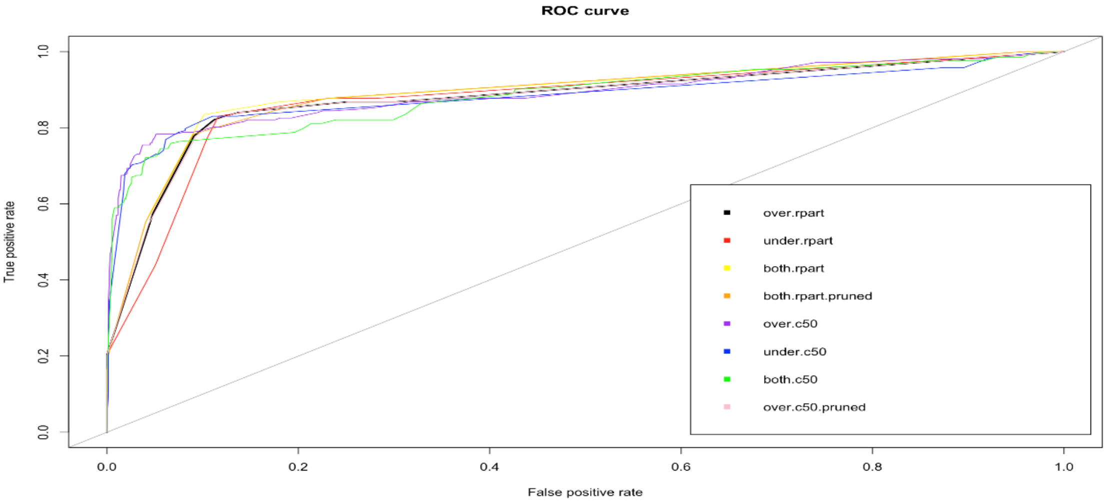


Figure 5 Summary of ROC curves for different combination of methods

4. Method 2: Logistic regression

Another powerful and convenient predictive technique is the logistic regression. Each observation gets a probability score as the output of the model. However, it will be highly affected by the multicollinearity issue such as in the case, resulting unstable parameter estimations. To solve this problem, variable reduction is required to reduce the collinearity among the predictors.

4.1. The full model and subset model

As mentioned in (2.1), two solutions for multicollinearity issue are proved to be valid with a resultant average VIF value of around 1. Therefore, the logistic regression models are built on these two sets of predictor variables, respectively. Furthermore, to reach the best set of parameter estimation, a three-time repeated cross-validation is used on the training data. Table 5 gives the results of estimation for the full model, model without variable Vmail and _Charge terms and model without variable Vmail and _Min terms.

Table 5 Estimated coefficients for the full model, sub-model 1 and sub-model 2

Variable	***Full set***	***Without _Charge terms***	***Without _Min terms***
Intercept	*-9.31 ****	*-6.49 ****	*-6.49 ****
Area_Code21	*0.14*	*0.17 ***	*0.17 ***
Area_Code22	*0.02*	*0.02*	*0.02*
Intl_Plan1	*2.43 ****	*2.40 ****	*2.40 ****
Vmail1	*2.75 ****		
Vmail_message	*0.05 ****	*-0.03 ****	*-0.03 ****
Day_Mins	*7.16 ****	*0.01 ****	
Day_Calls	*0.00*	*0.00*	*0.00*
Day_Charge	*-42.02 ****		*0.08 ****
Eve_Mins	*-0.49*	*0.02 ****	
Eve_Calls	*-0.02*	*-0.02*	*-0.02*
Eve_Charge	*1.80*		*0.07 ****
Night_Mins	*-4.33*	*0.04 ****	
Night_Calls	*0.00*	*-0.00*	*-0.00*
Night_Charge	*9.71*		*0.10 ****
Intl_Mins	*-9.04 ***	*0.08 ****	
Intl_Calls	*-0.07 ****	*-0.07 ****	*-0.07 ****
Intl_Charge	*33.78 ***		*0.30 ****
CustServ_Calls	*0.62 ****	*0.62 ****	*0.62 ****

***** 0.001, ** 0.05, * 0.1 significance level**

4.2. Evaluation on test set

To objectively measure the performance of the model, the evaluation is performed on the hold-out test dataset. The result is given in Table 6. Three models have roughly the same performance and sub-model without _charge terms and sub-model without _min terms have slightly higher AUC value than the full model.

Table 6 Summary of AUC and ACC on test set

	Full set	***Without _Charge terms***	***Without _Min terms***
AUC	*0.817*	*0.822*	*0.822*
ACC (accuracy)	*0.865*	*0.865*	*0.865*

However, based on AUC metrics, the logistic regression model is not as good as the decision tree models.

5. Method 3: Random forests

Random forests are an ensemble learning method for classification, that operate by constructing a multitude of decision trees at the training time and outputting the class that is the mode of the classes.

In the setting, we set the number of trees to be 1000. Automatically the algorithm chooses 4 variables at each split and the resulting out-of-bag error rate is 0.65%. Table 7 gives the confusion matrix where the classification error can be computed to be 1.3%.

Table 7 Confusion matrix of the training data

	Predicted 0	*Predicted 1*
Actual 0	*2967*	*39*
Actual 1	*0*	*3006*

The variable importance plots are shown in Figure 6. Two types of measurement are available: mean decrease in accuracy and mean decrease in node impurity. It seems that variables CustServ_Calls, Intl_Plan, Day_Charge and Day_Mins are key predictors.

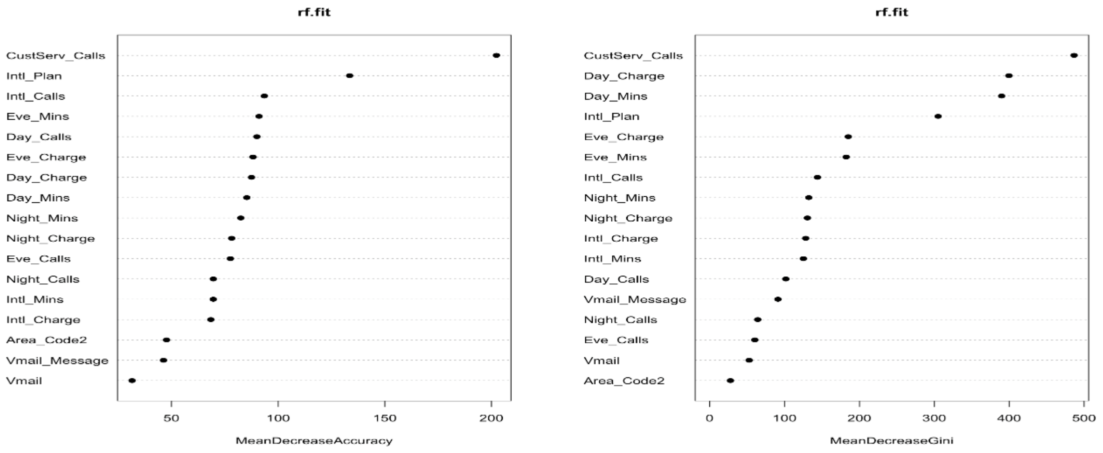


Figure 6 Variable Importance plot

The constructed random forests model is used on the test dataset to measure its performance. Accuracy = 0.958, precision = 0.890, recall = 0.802, AUC= 0.914, which seems to be the best among the three approaches.

6. Conclusion

Through the analysis above, three predictive models are built and the evaluation based on the test dataset is given in Table 8 where accuracy of lazy algorithm means the proportion of non-churner in the test set.

Table 8 Summary of models by three approaches

	Decision tree	*Logistic regression*	*Random forests*	*Lazy algorithm*
Accuracy	*0.893*	*0.865*	*0.958*	*0.859*
AUC	*0.879*	*0.822*	*0.914*	*-*

Both accuracy and AUC are positive metrics, the higher the value, the better the performance of a model. All the three models we build have higher accuracy than the lazy algorithm. Among the three, random forests show an overwhelming good performance over the other two models with accuracy up to 0.958 and AUC up to 0.914. Therefore, it should be used as the final predictive model in this setting. As for predictor variable, variable Customer service calls, International plan, day charge, day minute seem to be relatively important when predicting whether a customer will churn or not.

ASSIGNMENT 3

MapReduce in Hadoop

The third assignment consisted of two Hadoop MapReduce programs - one which would return the minimum, maximum, and average temperatures per Belgian town and month, and the second which would return a list of mutual friends of two individuals.

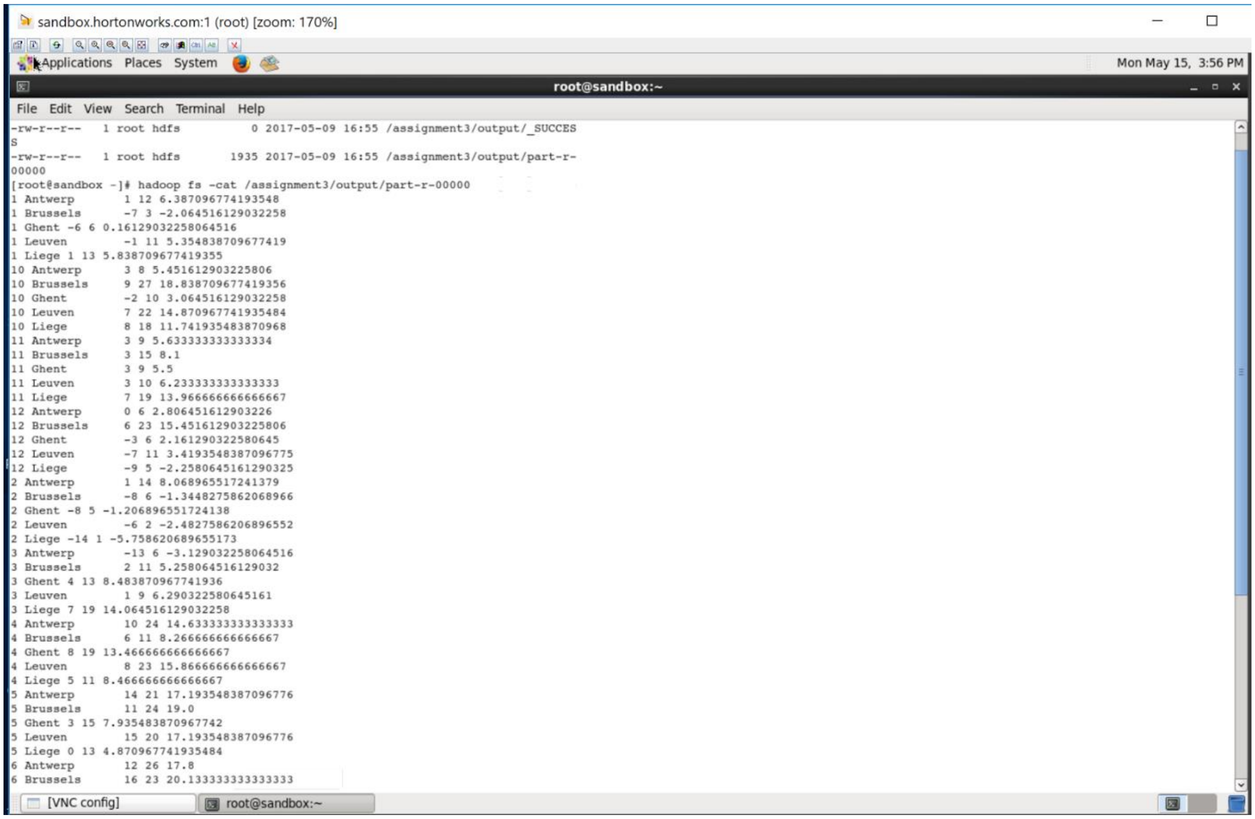
Part I: data “temperatures.txt”

1. Temperature Code:

```
public class MinMaxTempPerMonth {                         // Name the class
public static class MyMapper                              // Nested class (one to map, one to reduce)
extends Mapper<Object, Text, Text, Text> {                // Modifier does not output any values
public void map(Object key, Text value, Context context) throws IOException, InterruptedException {
String[] measurement = value.toString().split("\\t");     // Our input value is a line of measurements, tab
                                                          separated:

int month = Integer.parseInt(measurement[1]);             // Declare the month and temp columns as integer
int temp = Integer.parseInt(measurement[3]);              variables
String place = measurement[2];                            // The variable place is a string type
Text mappedKey = new Text(month+" "+place);               // Make a textual key - the groups for which values will
                                                          be outputted

Text mappedValue = new Text(temp+" "+temp);               // Make textual value - which values will be outputted
context.write(mappedKey, mappedValue);                    for every key
}}                                                        // Close the Map class
public static class MyReducer                             // Nested Reduce class
extends Reducer<Text, Text, Text, Text> {
public void reduce(Text key, Iterable<Text> values, Context context) throws IOException, InterruptedException {
int newMin = 0;                                           // Declare local variables we will need to calculate
int newMax = 0;                                           min,max,avg
int sum = 0;                                              // Avg will be calculated through a sum and count
int n = 0;
boolean firstValueDone = false;                           // Logic variable to be used in min/max loops
for (Text val : values) {
String[] valueMinMax = val.toString().split(" ");         // Convert our two temperatures separated with a
int min = Integer.parseInt(valueMinMax[0]);               space back to integers
int max = Integer.parseInt(valueMinMax[1]);
sum += min;                                               // Add up the temperatures
n++;                                                      // Increment operator to get count
if (min < newMin || !firstValueDone)                      // Go through temp values; if  current value is not
                                                          smaller than the next (statement is false), then take
                                                          on second value as new min
newMin = min;
if (max > newMax || !firstValueDone)                      // Same for max
newMax = max;
firstValueDone = true;
}                                                         // Avg uses decimals
double avg = sum/(double) n;                              // Make textual the reduced value that will be
                                                          outputted
Text reducedValue = new Text(newMin+" "+newMax+" "+avg);  // Output will be key-value pairs; specified what both
context.write(key, reducedValue);                         should be above
}}
public static void main(String[] args) throws Exception {
Configuration conf = new Configuration();
Job job = Job.getInstance(conf, "minmaxtemppermonth");    // Specifying the job with all the created classes
job.setJarByClass(MinMaxTempPerMonth.class);
job.setMapperClass(MyMapper.class);
job.setReducerClass(MyReducer.class);
job.setOutputKeyClass(Text.class);
job.setOutputValueClass(Text.class);
TextInputFormat.addInputPath(job, new Path(args[0]));
Path outputDir = new Path(args[1]);
FileOutputFormat.setOutputPath(job, outputDir);
FileSystem fs = FileSystem.get(conf);
fs.delete(outputDir, true);
System.exit(job.waitForCompletion(true) ? 0 : 1);
}
public Iterable<Integer> convert(int temp) {
// TODO Auto-generated method stub
return null;
}}
```

2. The Results:

First the variables month and place were specified to be used as keys. Next, the method for obtaining the minimum, maximum, and average values was described. To obtain the maximum, the temperature values were gone through one by one, retaining the highest one at each step until all the values were gone through and the same process was used for the minimum. The average was calculated through two variables - a sum and a count. The sum variable started with a temperature value and recursively added them up and was divided by the count to output an average.


sandbox.hortonworks.com:1 (root) [zoom: 170%]
Applications Places System
Mon May 15, 3:56 PM
root@sandbox:~
File Edit View Search Terminal Help
[VNC config]
root@sandbox:~

```
-rw-r--r--   1 root hdfs          0 2017-05-09 16:55 /assignment3/output/_SUCCES
S
-rw-r--r--   1 root hdfs       1935 2017-05-09 16:55 /assignment3/output/part-r-
00000
[root@sandbox ~]# hadoop fs -cat /assignment3/output/part-r-00000
1 Antwerp       1 12 6.387096774193548
1 Brussels      -7 3 -2.064516129032258
1 Ghent -6 6 0.16129032258064516
1 Leuven        -1 11 5.354838709677419
1 Liege 1 13 5.838709677419355
10 Antwerp      3 8 5.451612903225806
10 Brussels     9 27 18.838709677419356
10 Ghent        -2 10 3.064516129032258
10 Leuven       7 22 14.870967741935484
10 Liege        8 18 11.741935483870968
11 Antwerp      3 9 5.633333333333334
11 Brussels     3 15 8.1
11 Ghent        3 9 5.5
11 Leuven       3 10 6.233333333333333
11 Liege        7 19 13.966666666666667
12 Antwerp      0 6 2.806451612903226
12 Brussels     6 23 15.451612903225806
12 Ghent        -3 6 2.161290322580645
12 Leuven       -7 11 3.4193548387096775
12 Liege        -9 5 -2.2580645161290325
2 Antwerp       1 14 8.068965517241379
2 Brussels      -8 6 -1.3448275862068966
2 Ghent -8 5 -1.206896551724138
2 Leuven        -6 2 -2.4827586206896552
2 Liege -14 1 -5.758620689655173
3 Antwerp       -13 6 -3.129032258064516
3 Brussels      2 11 5.258064516129032
3 Ghent 4 13 8.483870967741936
3 Leuven        1 9 6.290322580645161
3 Liege 7 19 14.064516129032258
4 Antwerp       10 24 14.633333333333333
4 Brussels      6 11 8.266666666666667
4 Ghent 8 19 13.466666666666667
4 Leuven        8 23 15.866666666666667
4 Liege 5 11 8.466666666666667
5 Antwerp       14 21 17.193548387096776
5 Brussels      11 24 19.0
5 Ghent 3 15 7.935483870967742
5 Leuven        15 20 17.193548387096776
5 Liege 0 13 4.870967741935484
6 Antwerp       12 26 17.8
6 Brussels      16 23 20.133333333333333
```

Part II: data “friends.txt”

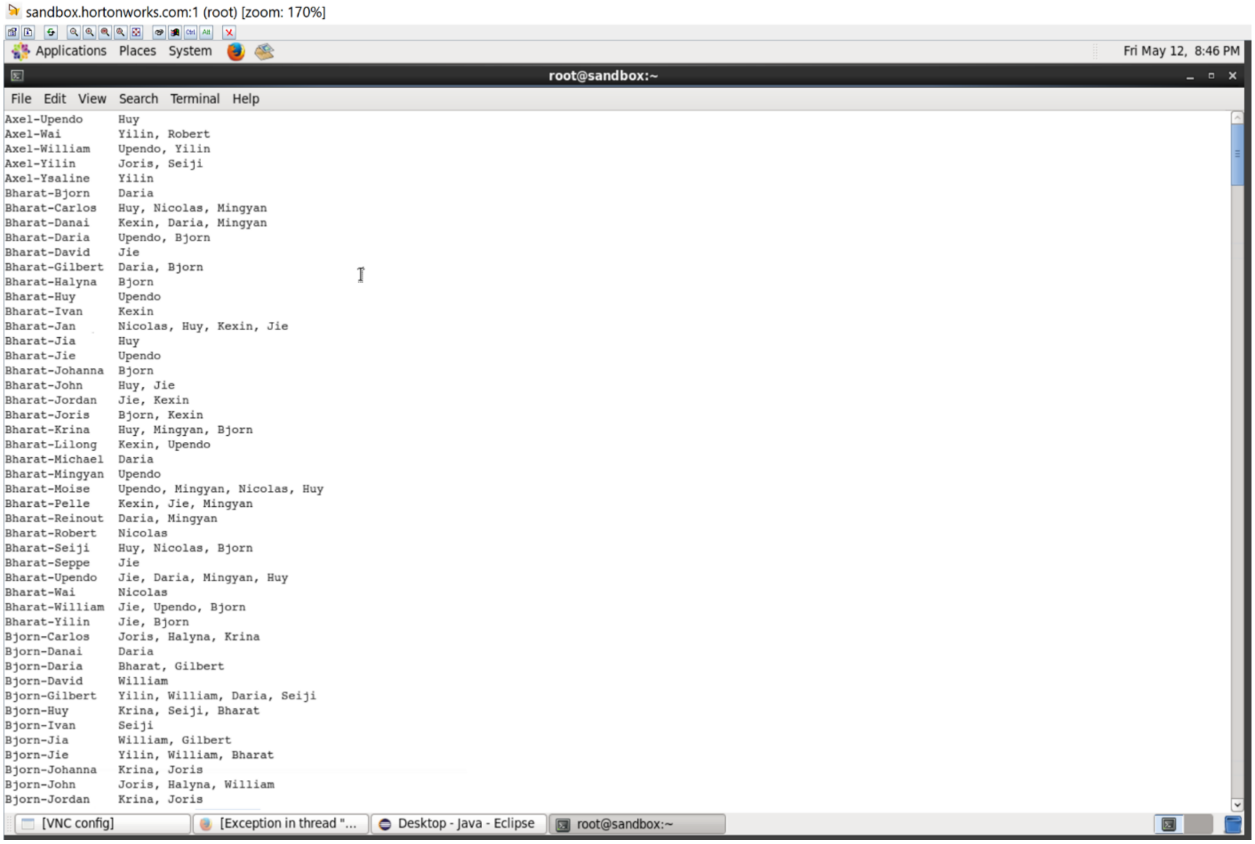
1. Friends Code:

```
public class FindFriend {                                                 // Specifying the overall class and the
                                                                          Mapper
static class MyMapper extends Mapper<LongWritable,Text,Text,Text>{
@Override                                                                 // Overriding the method
protected void map(LongWritable key, Text value, Context context)
throws IOException, InterruptedException {                                // Break up (tokenize) the strings
StringTokenizer stringTokenizer = new StringTokenizer(value.toString());
Text owner = new Text();                                                  // Instantiating variable
Set<String> set = new TreeSet<String>();                                  // Ordered set without duplicates, so names
                                                                          are alphabetically ordered and each listed
owner.set(stringTokenizer.nextToken());                                   only once
while(stringTokenizer.hasMoreTokens()){                                   // Go through names until there are none left
set.add(stringTokenizer.nextToken());
}
String[] friends = new String[set.size()];
friends = set.toArray(friends);                                           // Setting length of friends array
for(int i=0; i<friends.length;i++){                                       // Creating pairs that will be the key
for(int j=i+1; j<friends.length; j++){
String outputkey = friends[i]+friends[j];
context.write(new Text(outputkey), owner);                                // Want to the see the pair as key
}}}}                                                                      // Reducer class
static class MyReducer extends Reducer<Text, Text ,Text, Text>
{
@Override
protected void reduce(Text key, Iterable<Text> values,
Context context)
throws IOException, InterruptedException {
String commonFriends = "";                                                // Print only the parts which are equal to
                                                                          each other
for(Text val:values)                                                      // Creates list of common friends
{
if(commonFriends == ""){
commonFriends = val.toString();
}
else{
commonFriends = commonFriends+"--"+val.toString();
}}
context.write(key, new Text(commonFriends));
}}
public static void main(String[] args) throws Exception {
Configuration conf = new Configuration();
Job job = Job.getInstance(conf, “Find friend”);                           // Specifying the classes - mapper part,
                                                                          reducer part, key, and values
job.setJarByClass(FindFriend.class);
job.setMapperClass(MyMapper.class);
job.setReducerClass(MyReducer.class);
job.setOutputKeyClass(Text.class);
job.setOutputValueClass(Text.class);
TextInputFormat.addInputPath(job, new Path(args[0]));
Path outputDir = new Path(args[1]);
FileOutputFormat.setOutputPath(job, outputDir);
FileSystem fs = FileSystem.get(conf);
fs.delete(outputDir, true);
System.exit(job.waitForCompletion(true) ? 0 : 1);
}}
```

2. The Results

The figure below is a screenshot of the returned result. It can be seen that every pairs of individuals are treated as a key, followed by their common friends.

First, all the names were sorted alphabetically. Next pairs of people were created, avoiding duplicates, to be used as keys. Lastly, the names found in both people's friends lists were printed as the values.



sandbox.hortonworks.com:1 (root) [zoom: 170%]
Applications Places System
Fri May 12, 8:46 PM
root@sandbox:~
File Edit View Search Terminal Help
[VNC config]
[Exception in thread "...
Desktop - Java - Eclipse
root@sandbox:~

```
Axel-Upendo     Huy
Axel-Wai        Yilin, Robert
Axel-William    Upendo, Yilin
Axel-Yilin      Joris, Seiji
Axel-Ysaline    Yilin
Bharat-Bjorn    Daria
Bharat-Carlos   Huy, Nicolas, Mingyan
Bharat-Danai    Kexin, Daria, Mingyan
Bharat-Daria    Upendo, Bjorn
Bharat-David    Jie
Bharat-Gilbert  Daria, Bjorn
Bharat-Halyna   Bjorn
Bharat-Huy      Upendo
Bharat-Ivan     Kexin
Bharat-Jan      Nicolas, Huy, Kexin, Jie
Bharat-Jia      Huy
Bharat-Jie      Upendo
Bharat-Johanna  Bjorn
Bharat-John     Huy, Jie
Bharat-Jordan   Jie, Kexin
Bharat-Joris    Bjorn, Kexin
Bharat-Krina    Huy, Mingyan, Bjorn
Bharat-Lilong   Kexin, Upendo
Bharat-Michael  Daria
Bharat-Mingyan  Upendo
Bharat-Moise    Upendo, Mingyan, Nicolas, Huy
Bharat-Pelle    Kexin, Jie, Mingyan
Bharat-Reinout  Daria, Mingyan
Bharat-Robert   Nicolas
Bharat-Seiji    Huy, Nicolas, Bjorn
Bharat-Seppe    Jie
Bharat-Upendo   Jie, Daria, Mingyan, Huy
Bharat-Wai      Nicolas
Bharat-William  Jie, Upendo, Bjorn
Bharat-Yilin    Jie, Bjorn
Bjorn-Carlos    Joris, Halyna, Krina
Bjorn-Danai     Daria
Bjorn-Daria     Bharat, Gilbert
Bjorn-David     William
Bjorn-Gilbert   Yilin, William, Daria, Seiji
Bjorn-Huy       Krina, Seiji, Bharat
Bjorn-Ivan      Seiji
Bjorn-Jia       William, Gilbert
Bjorn-Jie       Yilin, William, Bharat
Bjorn-Johanna   Krina, Joris
Bjorn-John      Joris, Halyna, William
Bjorn-Jordan    Krina, Joris
```

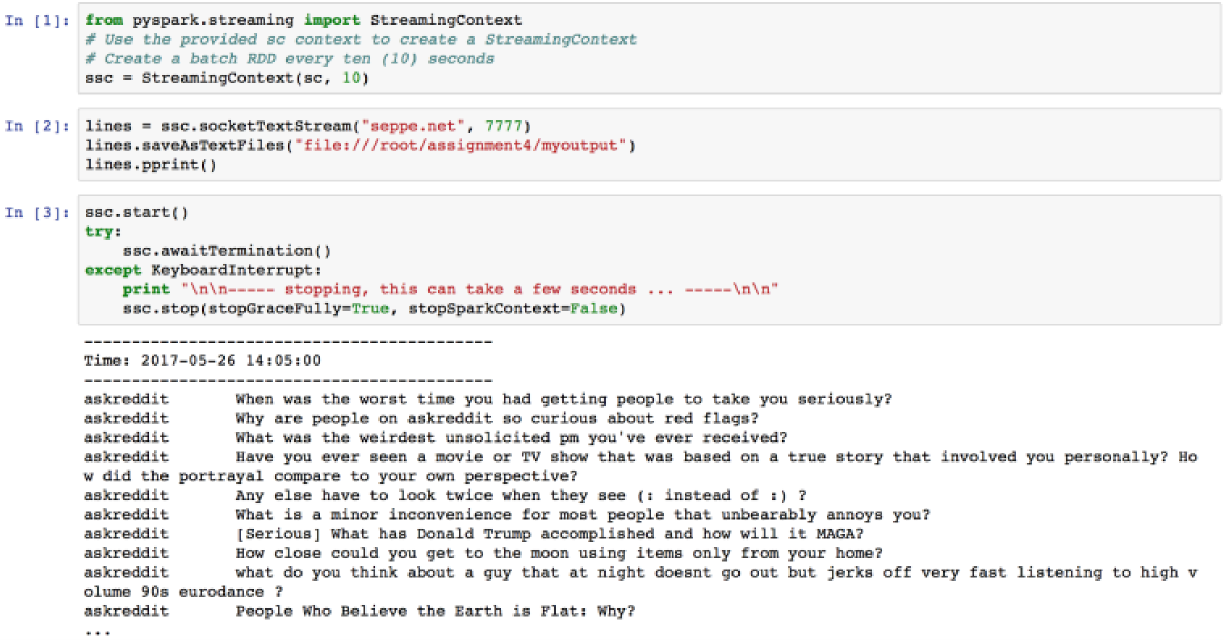
ASSIGNMENT 4

Spark and Spark Streaming

In this assignment, the interest is to construct and analyze a streaming data, which consists of reddit titles and their subreddit, then predict which subreddit the reddit title belongs to.

1. Construction of historical data

As the codes which derive the streaming data set are already given, the streaming data sets are constructed and stored in the folder “assignment4/myoutput” (Figure 1). It could be seen in Jupyter that the titles for each subreddit are separately stored in different .txt file, so they are downloaded from the online notebook and are vertically integrated into one off-line data set, which consists of 425 titles.



```
In [1]: from pyspark.streaming import StreamingContext
        # Use the provided sc context to create a StreamingContext
        # Create a batch RDD every ten (10) seconds
        ssc = StreamingContext(sc, 10)
```

```
In [2]: lines = ssc.socketTextStream("seppe.net", 7777)
        lines.saveAsTextFiles("file:///root/assignment4/myoutput")
        lines.pprint()
```

```
In [3]: ssc.start()
        try:
            ssc.awaitTermination()
        except KeyboardInterrupt:
            print "\n\n----- stopping, this can take a few seconds ... -----\n\n"
            ssc.stop(stopGraceFully=True, stopSparkContext=False)
```

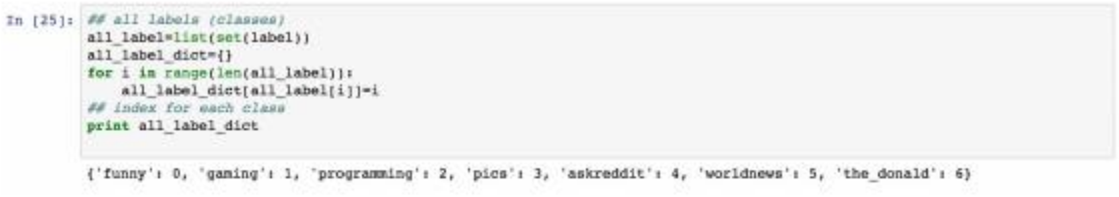
```
-------------------------------------------
Time: 2017-05-26 14:05:00
-------------------------------------------
askreddit       When was the worst time you had getting people to take you seriously?
askreddit       Why are people on askreddit so curious about red flags?
askreddit       What was the weirdest unsolicited pm you've ever received?
askreddit       Have you ever seen a movie or TV show that was based on a true story that involved you personally? Ho
w did the portrayal compare to your own perspective?
askreddit       Any else have to look twice when they see (: instead of :) ?
askreddit       What is a minor inconvenience for most people that unbearably annoys you?
askreddit       [Serious] What has Donald Trump accomplished and how will it MAGA?
askreddit       How close could you get to the moon using items only from your home?
askreddit       what do you think about a guy that at night doesnt go out but jerks off very fast listening to high v
olume 90s eurodance ?
askreddit       People Who Believe the Earth is Flat: Why?
...
```

Figure 1: Screen Capture of the Collected Data Set

2. Off-line Model Using Naive Bayesian Approach

2.1. Featurization and Normalization

Although the off-line dataset is constructed, it is not well structured, so not ready to be analyzed. Hence, efforts are made in splitting titles into words, removing punctuation and converting all the letters into lowercase. These are the most basic and simple approach of featurization and normalization of the text dataset. Subsequently, a numerical label is created for the seven subreddits (shown in Figure 2): 0 for “funny”, 1 for “gaming”, 2 for “programming”, 3 for “pics”, 4 for “askreddit”, 5 for “worldnews” and 6 for “the donald”.



```
In [25]: ## all labels (classes)
         all_label=list(set(label))
         all_label_dict={}
         for i in range(len(all_label)):
             all_label_dict[all_label[i]]=i
         ## index for each class
         print all_label_dict

         {'funny': 0, 'gaming': 1, 'programming': 2, 'pics': 3, 'askreddit': 4, 'worldnews': 5, 'the_donald': 6}
```

Figure 2: Screen Capture of the Created Label for Subreddits

2.2. Construction of a Word List

A word list with n dimensions is build, and the n is determined by the total number of words appeared in all the titles. In this word list, every word appeared are treated as a new attribute, and every title is an instance, then the frequency of each word’s appearance is recorded for each title.

2.3. Naïve Bayesian Approach

As the prediction of subreddits for titles is of interest, the Bayesian theoroem could be of use. The particular classifier is constructed below:

$$\boldsymbol{P(subreddit\ |title) = \frac{P(title\ \left|subreddit\right)*P(subreddit)}{P(title)}} \quad (1)$$

where P(subreddit | title), is the probability of subreddit given the title, the posterior probability here, and is what would be used in predictions given titles; the P(title | subreddit), the prior probability here, is the probability of the title conditioned on the subreddit it belong to; the P(subreddit) is the probability of the subreddit and the P(title) is the probability of the title.

• P(title) is the same for every titles. Here, it is equal to 1/425 (as the sample size of the data set is 425).

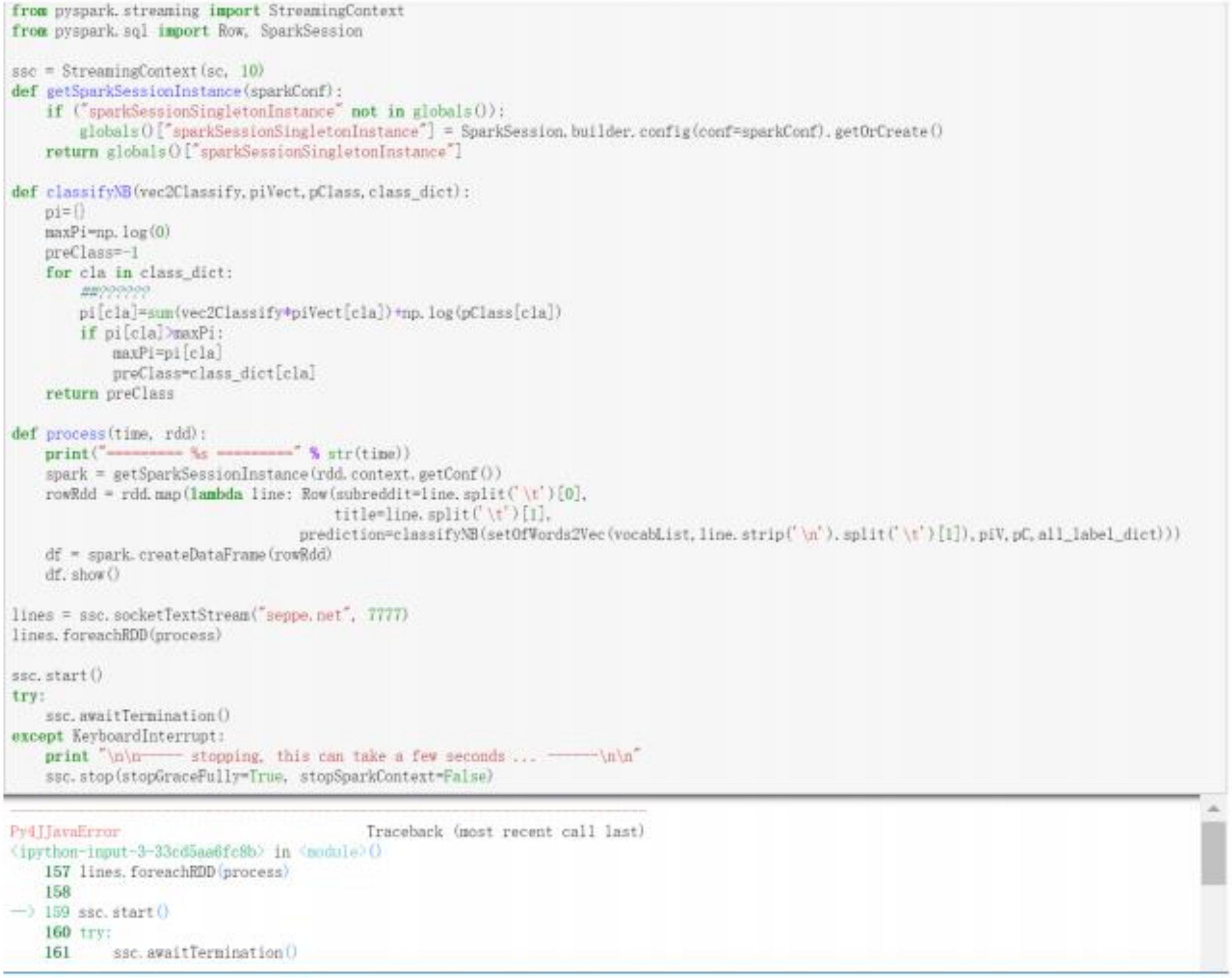
• P(subreddit) could be calculated as the frequency of each subreddit devided by the sample size.

• For P(title | subreddit), the computation is less obvious. we need to first make a strong assumption that the probability of each word’s appearance in the title is independent. Then, it is clear that P(title | subreddit) is the product of P(word | subreddit). However, in the word list constructed previously, some words have 0 frequency, which leads to the corresponding P(word | subreddit) = 0. In order to avoid resulting a P(title | subreddit) = 0, the probability of 0-frequency words are set to be 1, which would make no difference on the multiplication.

As a result, P(subreddit | title) is computed by plugging in every elements of (1). However, it is obvious that each given title has 7 values in P(subreddit | title), which are P(funny | title), P(gaming | title), P(programming | title), P(pics | title), P(askreddit | title), P(worldnews | title) and P(thedonald | title). Among them, the probability with the largest value indicates that the title belongs to that specific subreddit. Therefore, this naïve Bayesian model is built.

3. Prediction in Streaming Setup

When it comes to putting the model online again and applying it to do predictions for streaming data, we failed to combine the codes at hand with the codes in the guide document provided by the professor. The prediction part is replace by ”classifyNB”, a function constructed to calculate P(subreddit|title), based on which the prediction is made. However, the codes is not working and Jupyter gives error warnings as is shown in Figure 3. Many adjustments are done and tried, but the combined codes are not working anyway.

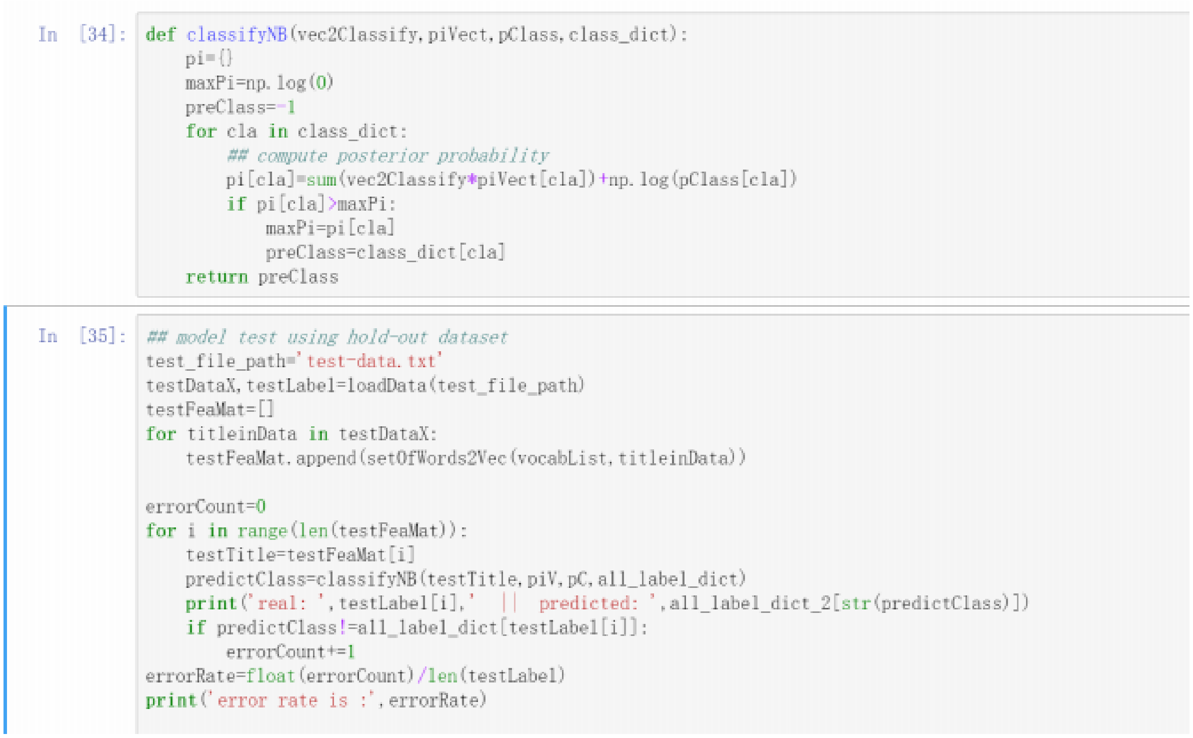


```
from pyspark.streaming import StreamingContext
from pyspark.sql import Row, SparkSession

ssc = StreamingContext(sc, 10)
def getSparkSessionInstance(sparkConf):
    if ("sparkSessionSingletonInstance" not in globals()):
        globals()["sparkSessionSingletonInstance"] = SparkSession.builder.config(conf=sparkConf).getOrCreate()
    return globals()["sparkSessionSingletonInstance"]

def classifyNB(vec2Classify,piVect,pClass,class_dict):
    pi={}
    maxPi=np.log(0)
    preClass=-1
    for cla in class_dict:
        ##??????
        pi[cla]=sum(vec2Classify*piVect[cla])+np.log(pClass[cla])
        if pi[cla]>maxPi:
            maxPi=pi[cla]
            preClass=class_dict[cla]
    return preClass

def process(time, rdd):
    print("========= %s =========" % str(time))
    spark = getSparkSessionInstance(rdd.context.getConf())
    rowRdd = rdd.map(lambda line: Row(subreddit=line.split('\t')[0],
                                      title=line.split('\t')[1],
                                  prediction=classifyNB(setOfWords2Vec(vocabList,line.strip('\n').split('\t')[1]),piV,pC,all_label_dict)))
    df = spark.createDataFrame(rowRdd)
    df.show()

lines = ssc.socketTextStream("seppe.net", 7777)
lines.foreachRDD(process)

ssc.start()
try:
    ssc.awaitTermination()
except KeyboardInterrupt:
    print "\n\n----- stopping, this can take a few seconds ... -----\n\n"
    ssc.stop(stopGraceFully=True, stopSparkContext=False)
```

```
---------------------------------------------------------------------------
Py4JJavaError                             Traceback (most recent call last)
<ipython-input-3-33cd5aa6fc8b> in <module>()
    157 lines.foreachRDD(process)
    158
--> 159 ssc.start()
    160 try:
    161     ssc.awaitTermination()
```

Figure 3: Srceen Capture of the Errors

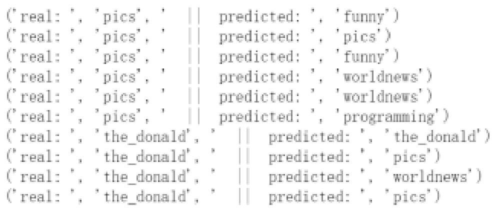
However, the prediction ability of the Bayesian classifier remains our interest. So another data sets are then captured and regarded as the test data set. As these newly constructed data sets are, as before, stored separately according to the seven different subreddits, they are downloaded again and are integrated vertically which results in one dataset as a whole. Now the test data set containing 165 titles is ready. The relevant codes are shown in Figure 4 and the partial results could be seen in Figure 5 and 6. The error rate is 0.5758.



```
In  [34]: def classifyNB(vec2Classify,piVect,pClass,class_dict):
              pi={}
              maxPi=np.log(0)
              preClass=-1
              for cla in class_dict:
                  ## compute posterior probability
                  pi[cla]=sum(vec2Classify*piVect[cla])+np.log(pClass[cla])
                  if pi[cla]>maxPi:
                      maxPi=pi[cla]
                      preClass=class_dict[cla]
              return preClass
```

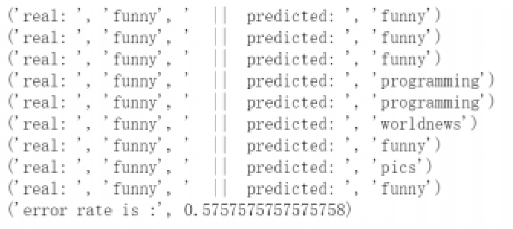
```
In  [35]: ## model test using hold-out dataset
          test_file_path='test-data.txt'
          testDataX,testLabel=loadData(test_file_path)
          testFeaMat=[]
          for titleinData in testDataX:
              testFeaMat.append(setOfWords2Vec(vocabList,titleinData))

          errorCount=0
          for i in range(len(testFeaMat)):
              testTitle=testFeaMat[i]
              predictClass=classifyNB(testTitle,piV,pC,all_label_dict)
              print('real: ',testLabel[i],'  ||  predicted: ',all_label_dict_2[str(predictClass)])
              if predictClass!=all_label_dict[testLabel[i]]:
                  errorCount+=1
          errorRate=float(errorCount)/len(testLabel)
          print('error rate is :',errorRate)
```

Figure 4: Codes for Test Data Set



```
('real: ', 'pics', '  ||  predicted: ', 'funny')
('real: ', 'pics', '  ||  predicted: ', 'pics')
('real: ', 'pics', '  ||  predicted: ', 'funny')
('real: ', 'pics', '  ||  predicted: ', 'worldnews')
('real: ', 'pics', '  ||  predicted: ', 'worldnews')
('real: ', 'pics', '  ||  predicted: ', 'programming')
('real: ', 'the_donald', '  ||  predicted: ', 'the_donald')
('real: ', 'the_donald', '  ||  predicted: ', 'pics')
('real: ', 'the_donald', '  ||  predicted: ', 'worldnews')
('real: ', 'the_donald', '  ||  predicted: ', 'pics')
```

Figure 5: Partial Results-1



```
('real: ', 'funny', '  ||  predicted: ', 'funny')
('real: ', 'funny', '  ||  predicted: ', 'funny')
('real: ', 'funny', '  ||  predicted: ', 'funny')
('real: ', 'funny', '  ||  predicted: ', 'programming')
('real: ', 'funny', '  ||  predicted: ', 'programming')
('real: ', 'funny', '  ||  predicted: ', 'worldnews')
('real: ', 'funny', '  ||  predicted: ', 'funny')
('real: ', 'funny', '  ||  predicted: ', 'pics')
('real: ', 'funny', '  ||  predicted: ', 'funny')
('error rate is :', 0.5757575757575758)
```

Figure 6: Partial Results-2

4. Further Possible Improvements

As is shown from the previous section that the error rate is 0.5758, the prediction accuracy of this model needs to be improved. Here are some aspects under consideration:

- Normalize the words in titles in a more sophistic way, for example, by removing the stop words, stemming the words.

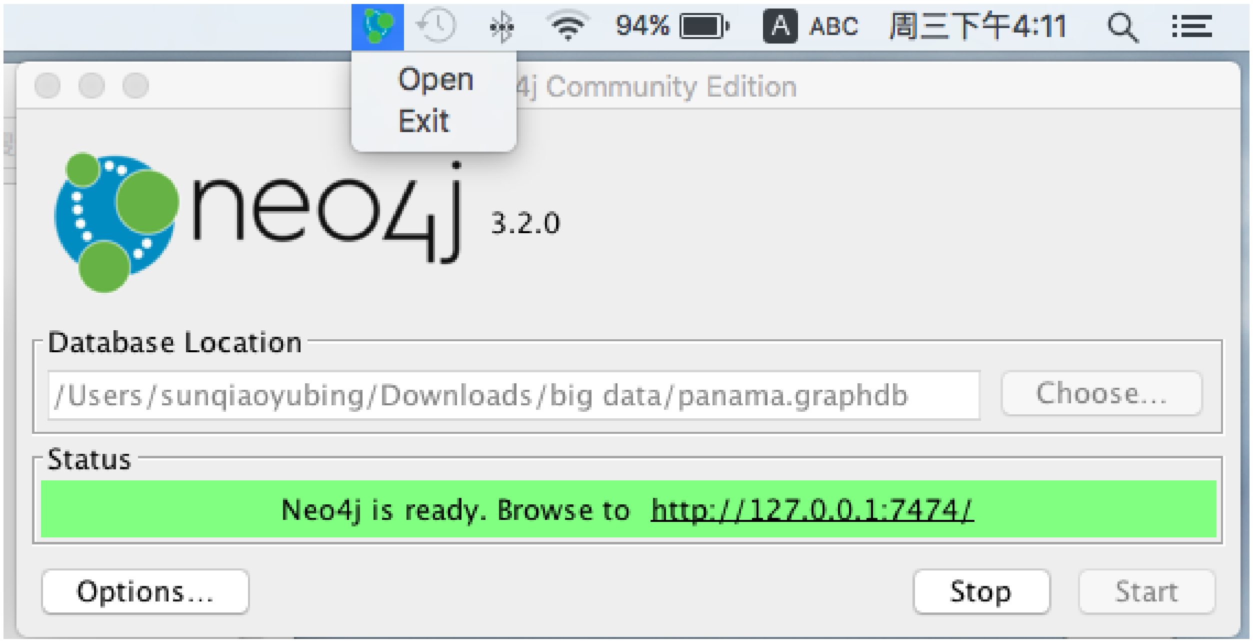
- Omit those words which have too high frequency or too low frequency, because they don’t have strong ability in distinguishing a subreddit from the others. This is tried in our code, but we didn’t manage to finish it.

- The naïve Bayesian approach is applied with the theoretic assumption that the probability of each used word is independent, but this is not always the case in reality. So, taking into account the order of words could also be a way to improve the algorithm of the predictive model.

Neo4j and Gephi

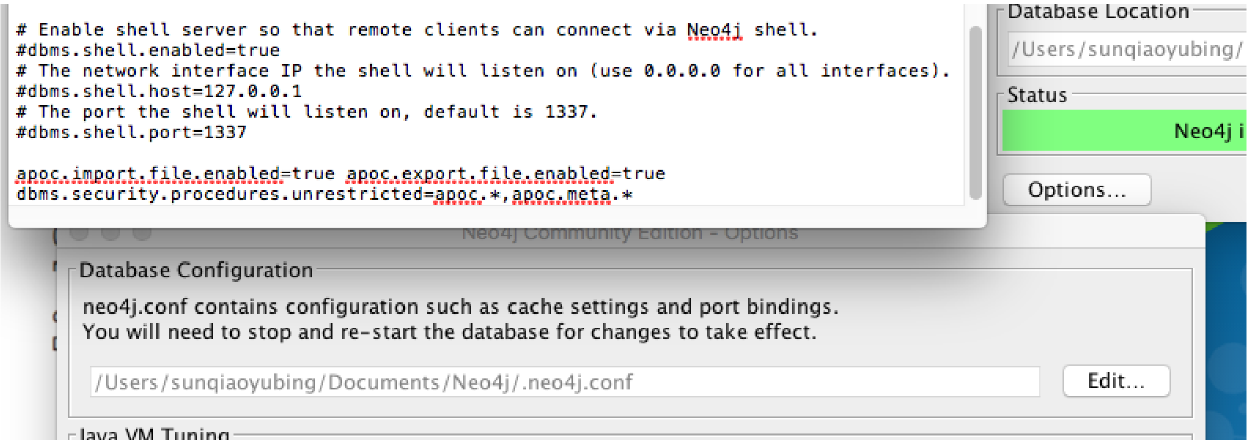
1. Setup of Neo4j, APOC procedure &Gephi

First, we set up Neoj4 step by step following the procedure, then opened it and added the APOC procedure.



/Users/sunqiaoyubing/Downloads/big data/panama.graphdb

Neo4j is ready. Browse to http://127.0.0.1:7474/



```
# Enable shell server so that remote clients can connect via Neo4j shell.
#dbms.shell.enabled=true
# The network interface IP the shell will listen on (use 0.0.0.0 for all interfaces).
#dbms.shell.host=127.0.0.1
# The port the shell will listen on, default is 1337.
#dbms.shell.port=1337

apoc.import.file.enabled=true apoc.export.file.enabled=true
dbms.security.procedures.unrestricted=apoc.*,apoc.meta.*
```

neo4j.conf contains configuration such as cache settings and port bindings.
You will need to stop and re-start the database for changes to take effect.

/Users/sunqiaoyubing/Documents/Neo4j/.neo4j.conf

Then we used APOC plug-in to guide graph and data in Neo4j into Gephi. Gephi is convenient, especially for visualizing complex relationships and making it easier to find and understand entities and their connections.

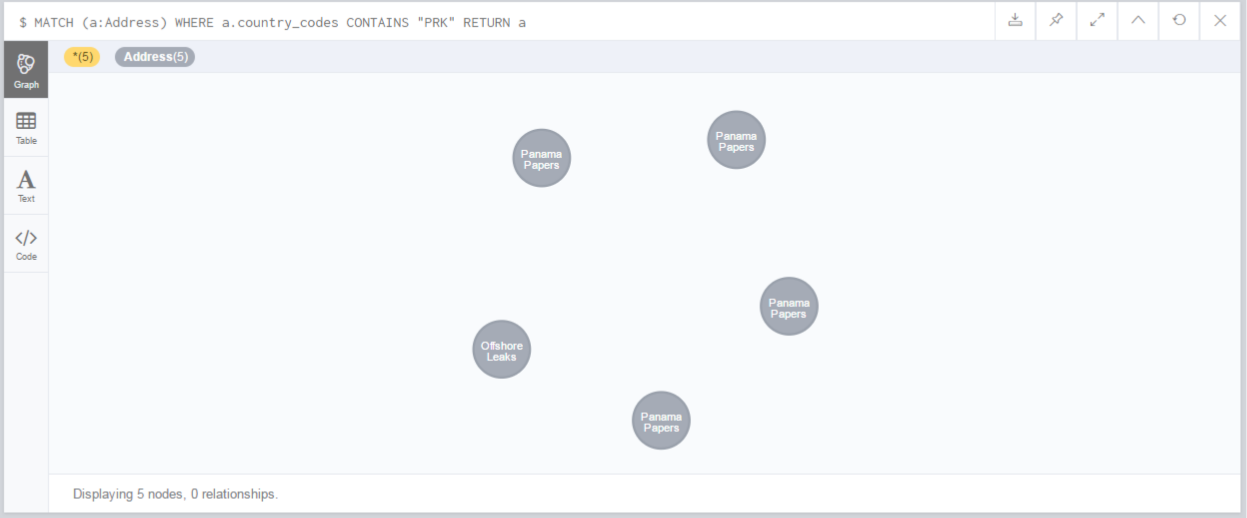
2. Visualization Analysis

2.1. North Korea

In analyzing the Panama Papers we became interested in whether the data contained any information on entities, officers, or intermediaries linked to North Korea. Thus, we began our search with the following:

```
MATCH (a:Address) WHERE a.country_codes CONTAINS "PRK" RETURN a
```

This resulted in five address nodes:



```
$ MATCH (a:Address) WHERE a.country_codes CONTAINS "PRK" RETURN a
```

Displaying 5 nodes, 0 relationships.

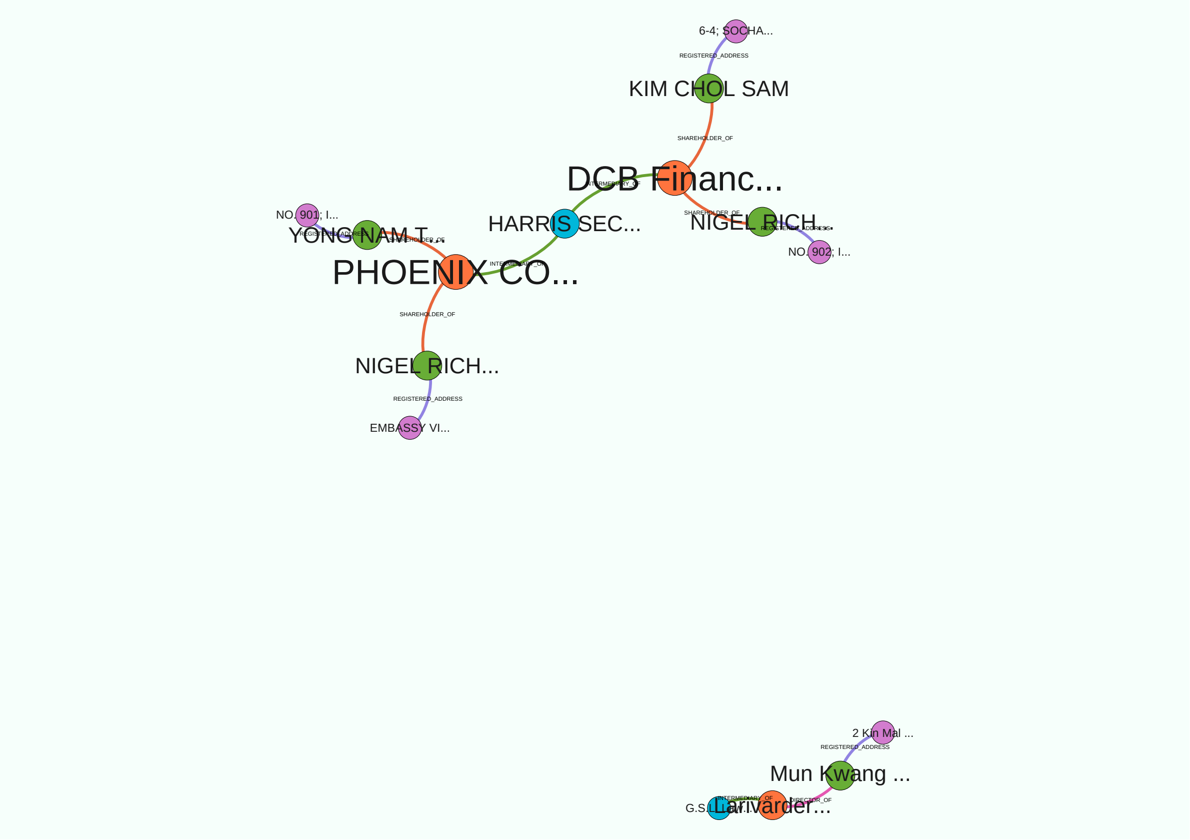
picture 1

Next, they were explored for child relationships with other nodes. All listed Pyongyang addresses, with two of them listing seemingly neighbouring ones – no. 901 and no. 902 of “INTERNATIONAL HOUSE OF CULTURE; CENTRAL DISTRICT; PYONGYANG CITY; D.P.R. OF KOREA”. Furthermore two similar names were registered to two of the addresses – a NIGEL RICHARD JAMES (COWIE) (after a quick google search which revealed him to be a UK banker, those two nodes were treated as the same person).

Interestingly, it was discovered that four of the five officer nodes were assigned with the country and country code labels of Iran. Furthermore, they were all connected through two companies both of which were registered at the same intermediary “HARRIS SECRETARIES LIMITED” in Hong Kong. This made us wonder whether there were any more connections between Iran and North Korea in the Panama Papers.

The following code was used to visualize the network in Gephi:

```
MATCH path= (a:Address)--(o:Officer)--(e:Entity)--(i:Intermediary) WHERE a.country_codes CONTAINS "PRK"
WITH collect(path) AS paths CALL apoc.gephi.add(null, 'workspace1', paths) YIELD nodes, relationships, time
RETURN nodes, relationships, time
```



picture 2

Next, we looked more into officers with the country label of Iran, using intermediaries in Hong Kong, as well as additional companies using the same address as the intermediary HARRIS SECRETARIES LIMITED with the following code:

```
MATCH (o:Officer)--(e:Entity)--(i:Intermediary) WHERE o.country_codes = 'IRN' AND i.country_codes = 'HKG' RETURN o,e,i
```

```
MATCH (e:Entity) WHERE e.address CONTAINS 'HONG KONG TRADE CENTRE  161-167 DES VOEUX ROAD  CENTRAL' RETURN e
```

We discovered 22 officers from Iran linked to 18 entities which use 15 different intermediaries in Hong Kong, as well as two entities registered to the same address as HARRIS SECRETARIES LIMITED.

2.2. Iran

Having exhausted the North Korean side of our search (apart from a fifth address node connected to a South Korean officer tied to a Russian/Cyprus intermediary with too many nodes to analyze and non-North Korea related), we decided to delve more into Iran and see whether any entities are registered there, or if it is the case only with officers, and if so, what is the pattern of their relationships.

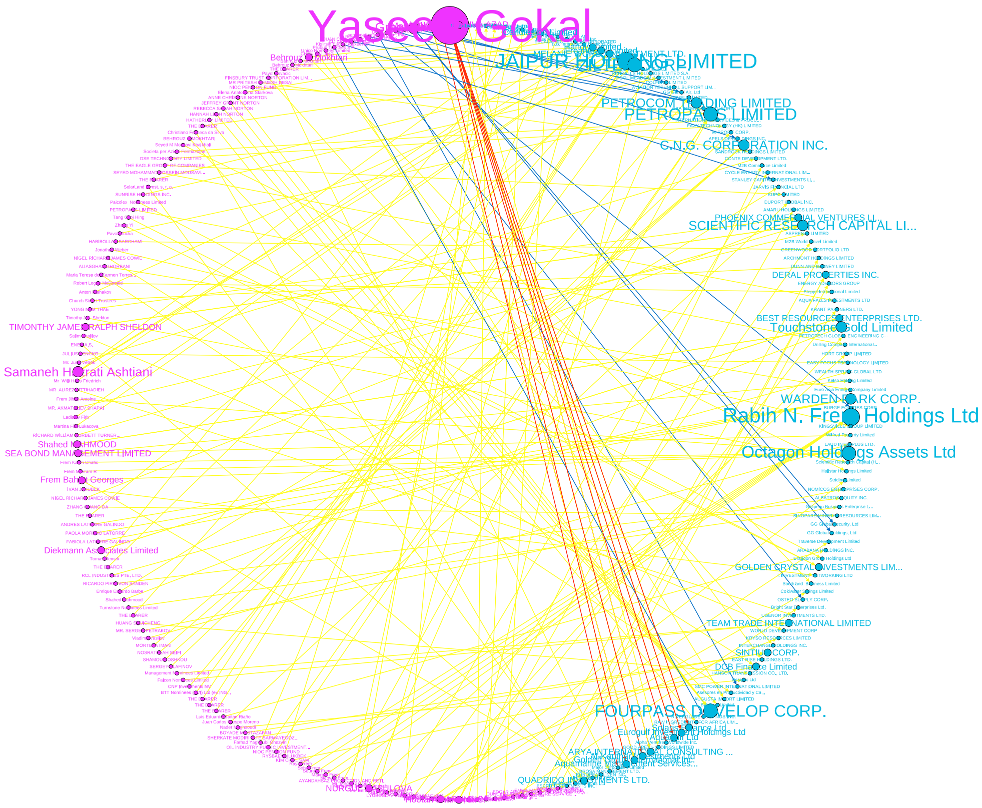
First, we decided to look into every officer labeled in Iran and the companies they are linked to and whether those are also in Iran with the following code:

```
MATCH (o:Officer)--(e:Entity) WHERE o.country_codes = 'IRN' AND NOT
e.country_codes='IRN' RETURN o,e
```

We found that 146 out of 147 officers from Iran were linked to 122 out of 123 overseas companies (last one belonged to a company with no country or country code label). Meaning, officers would use Iran as their address but not conduct any of their business there, according to this data.

The following code was used to visualize the network in Gephi:

```
MATCH path = (o:Officer)--(e:Entity) WHERE o.country_codes = 'IRN' AND NOT
e.country_codes='IRN' WITH collect(path) AS paths
CALL apoc.gephi.add(null, 'workspace2', paths) YIELD nodes, relationships, time
RETURN nodes, relationships, time
```

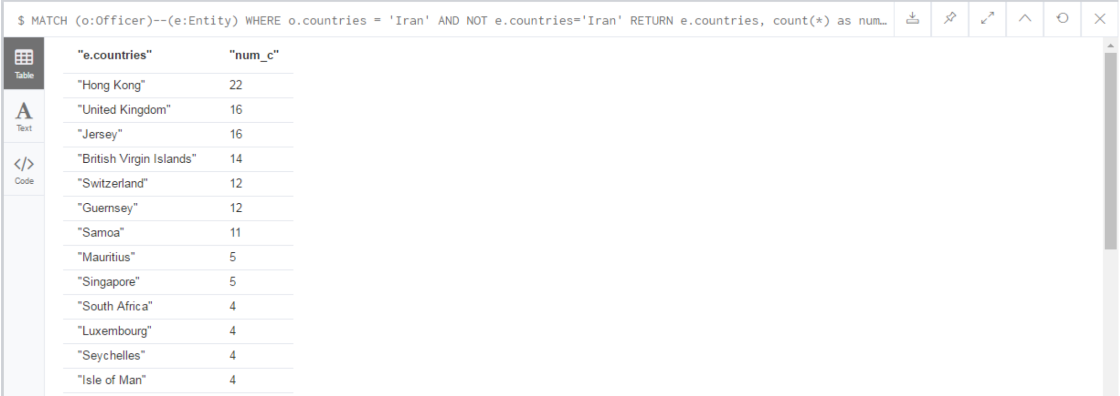

Yaseen Gokal
JAIPUR HOLDING CORP.
LIMITED
PETROCOM HOLDING LIMITED
PETROPARS LIMITED
C.N.G. CORPORATION INC.
PHOENIX COMMERCIAL VENTURES LI...
SCIENTIFIC RESEARCH CAPITAL LI...
DERAL PROPERTIES INC.
BEST RESOURCES ENTERPRISES LTD.
Touchstone Gold Limited
WARDEN PARK CORP.
Rabih N. Frem Holdings Ltd
Octagon Holdings Assets Ltd
GOLDEN CRYSTAL INVESTMENTS LIM...
TEAM TRADE INTERNATIONAL LIMITED
SINTIL CORP.
DCB Finance Limited
FOURPASS DEVELOP CORP.
ARYA INTERNATIONAL CONSULTING ...
QUADRIDO INVESTMENTS LTD.
TIMONTHY JAMES RALPH SHELDON
Samaneh Hezrati Ashtiani
Shahed MAHMOOD
SEA BOND MANAGEMENT LIMITED
Frem Bahige Georges
Diekmann Associates Limited

picture 3

The yellow lines stand for the relationships of shareholders, with the orange and blue lines representing “directors’ of” and “beneficiaries’ of ”relationships. The purple points are Officers and the blue points are Entities. We can see it directly that most of the relationship is shareholders’ relationships.

Since these companies were not registered in Iran, we wanted to know in which countries most of them were:

MATCH (o:Officer)--(e:Entity) WHERE o.countries = 'Iran' AND NOT e.countries='Iran' RETURN e.countries, count(*) as num_c ORDER BY num_c desc



```
$ MATCH (o:Officer)--(e:Entity) WHERE o.countries = 'Iran' AND NOT e.countries='Iran' RETURN e.countries, count(*) as num…
```

"e.countries"	"num_c"
"Hong Kong"	22
"United Kingdom"	16
"Jersey"	16
"British Virgin Islands"	14
"Switzerland"	12
"Guernsey"	12
"Samoa"	11
"Mauritius"	5
"Singapore"	5
"South Africa"	4
"Luxembourg"	4
"Seychelles"	4
"Isle of Man"	4

Graph 1

2.3. Niue

Looking at the above list of countries, we remembered coming across a very obscure location – Niue, a small island nation in the south Pacific with a population of just 1,612 and mostly supported by New Zealand and we decided to look into whether there were any entities there:

MATCH (e:Entity) WHERE e.countries CONTAINS 'NIUE' RETURN e

MATCH (e:Entity) WHERE e.country_codes CONTAINS 'NIU' RETURN e

MATCH (e:Entity) WHERE e.jurisdiction CONTAINS 'NIUE' RETURN e

No entities had the country label Niue, though 372 had the country code NIU, and 9611 fell under the jurisdiction of Niue. Because the last number was so staggeringly high, we decided to investigate the latter. Our thought process was to see whether there was a single officer or intermediary putting all these companies under Niue jurisdiction, so we looked for officers and intermediaries related to the highest number of entities under Niue jurisdiction:

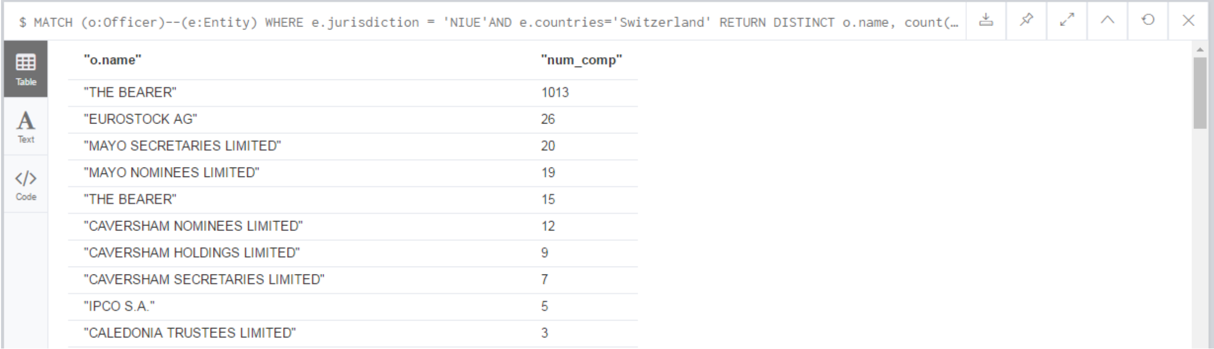
MATCH (o:Officer)--(e:Entity) WHERE e.jurisdiction = 'NIUE' RETURN e.countries, count(*) as num_c ORDER BY num_c desc

MATCH (i:Intermediary)--(e:Entity) WHERE e.jurisdiction = 'NIUE' RETURN e.countries, count(*) as num_c ORDER BY num_c desc

Most officers and intermediaries came from Switzerland, Panama, and Hong Kong. We then focused in on the linked officers and intermediaries from Switzerland:

MATCH (o:Officer)--(e:Entity) WHERE e.jurisdiction = 'NIUE'AND e.countries='Switzerland' RETURN DISTINCT o.name, count(*) as num_comp ORDER BY num_comp desc

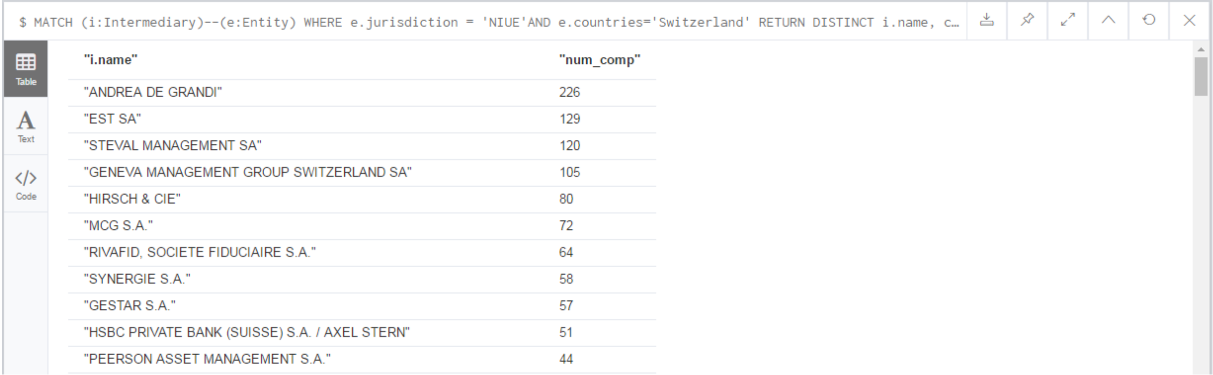
MATCH (i:Intermediary)--(e:Entity) WHERE e.jurisdiction = 'NIUE'AND e.countries='Switzerland' RETURN DISTINCT i.name, count(*) as num_comp ORDER BY num_comp desc



```
$ MATCH (o:Officer)--(e:Entity) WHERE e.jurisdiction = 'NIUE'AND e.countries='Switzerland' RETURN DISTINCT o.name, count(…
```

"o.name"	"num_comp"
"THE BEARER"	1013
"EUROSTOCK AG"	26
"MAYO SECRETARIES LIMITED"	20
"MAYO NOMINEES LIMITED"	19
"THE BEARER"	15
"CAVERSHAM NOMINEES LIMITED"	12
"CAVERSHAM HOLDINGS LIMITED"	9
"CAVERSHAM SECRETARIES LIMITED"	7
"IPCO S.A."	5
"CALEDONIA TRUSTEES LIMITED"	3

Graph 2



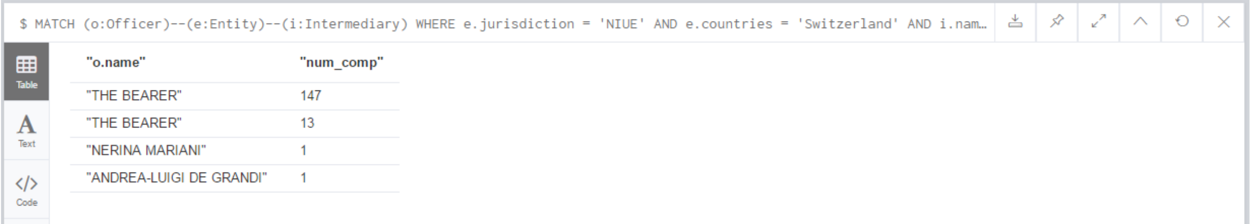
```
$ MATCH (i:Intermediary)--(e:Entity) WHERE e.jurisdiction = 'NIUE'AND e.countries='Switzerland' RETURN DISTINCT i.name, c…
```

"i.name"	"num_comp"
"ANDREA DE GRANDI"	226
"EST SA"	129
"STEVAL MANAGEMENT SA"	120
"GENEVA MANAGEMENT GROUP SWITZERLAND SA"	105
"HIRSCH & CIE"	80
"MCG S.A."	72
"RIVAFID, SOCIETE FIDUCIAIRE S.A."	64
"SYNERGIE S.A."	58
"GESTAR S.A."	57
"HSBC PRIVATE BANK (SUISSE) S.A. / AXEL STERN"	51
"PEERSON ASSET MANAGEMENT S.A."	44

Graph 3

The most common officer name was “THE BEARER” – a common pseudonym likely referring to ‘bearer shares’ which keep the shareholder’s name anonymous, and thus not very useful. We had better luck with intermediaries though, with ANDREA DE GRANDI being linked to 226 companies in Niue and proceeded to see if a single officer was linked to multiple of those:

MATCH (o:Officer)--(e:Entity)--(i:Intermediary) WHERE e.jurisdiction = 'NIUE' AND e.countries = 'Switzerland' AND i.name = 'ANDREA DE GRANDI' RETURN o.name, count(*) as num_comp ORDER BY num_comp desc



```
$ MATCH (o:Officer)--(e:Entity)--(i:Intermediary) WHERE e.jurisdiction = 'NIUE' AND e.countries = 'Switzerland' AND i.nam…
```

"o.name"	"num_comp"
"THE BEARER"	147
"THE BEARER"	13
"NERINA MARIANI"	1
"ANDREA-LUIGI DE GRANDI"	1

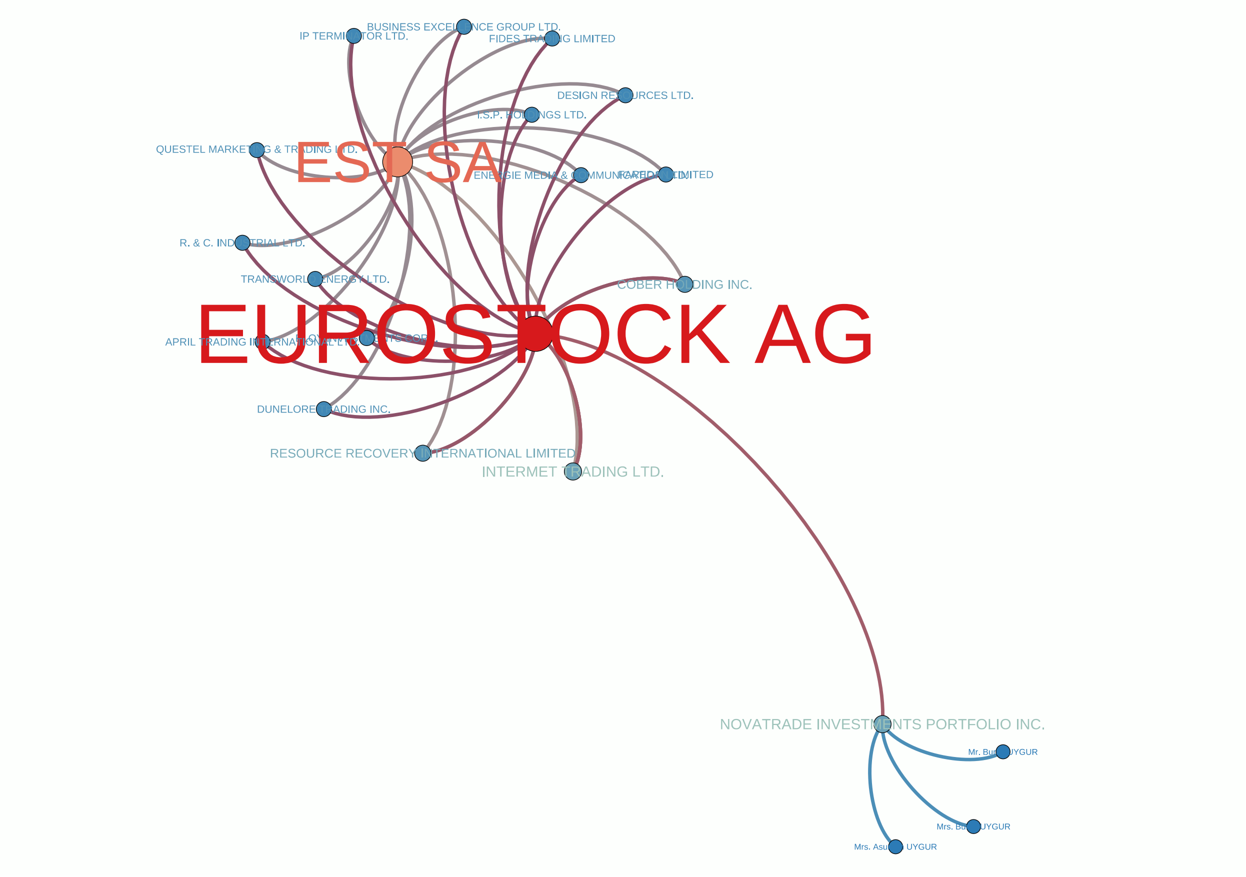
Once again, we had a similar ‘bearer’ problem, so we tried the next largest intermediary as listed above:

MATCH (o:Officer)--(e:Entity)--(i:Intermediary) WHERE e.jurisdiction = 'NIUE' AND e.countries = 'Switzerland' AND i.name = 'EST SA' RETURN o.name,count(*) as num_name ORDER BY num_name desc

We found that EUROSTOCK AG was linked to 19 of the entities (and a seemingly misspelled version to one more entity). By examining those we came across the entity “NOVATRADE INVESTMENTS PORTFOLIO INC” that EUROSTOCK AG was a shareholder of which we found interesting because all three of its beneficiaries shared the same last name of “UYGUR”.

The following code was used to visualize the network in Gephi:

```
MATCH path =  (o1:Officer)--(e1:Entity)--(o2:Officer)--(e2:Entity)--(i:Intermediary) WHERE i.name = 'EST SA' AND o2.name = 'EUROSTOCK AG' AND e1.name = 'NOVATRADE INVESTMENTS PORTFOLIO INC.' AND o1.name CONTAINS 'UYGUR' WITH collect(path) AS paths

CALL apoc.gephi.add(null, 'workspace3', paths) YIELD nodes, relationships, time

RETURN nodes, relationships, time
```



picture 4

Thus, we found that of the thousands of entities under Niue jurisdiction, Switzerland had the most officers and intermediaries linked to them, with ANDREA DE GRANDI being the main intermediary. Furthermore, we discovered that EUROSTOCK AG was linked to 20 of ANDREA DE GRANDI’s Niue entities and an apparent Turkish family linked to some of those as beneficiaries through EUROSTOCK AG and Swiss HORIZON INVESTMENT SA as its intermediary.

APPENDIX:

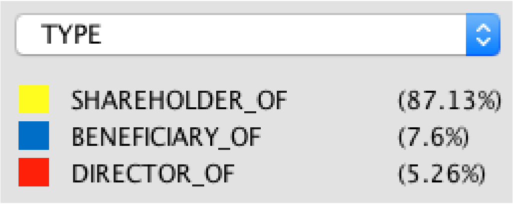
1. Some explanation of Gephi operations:

1.1. The first and third graph use the layout "Force Atlas" because this layout can easily depict the relationship among different subjects. The second graph uses the layout "Circular Layout" because it's an impressive way to express the relations, including the proportions of the relation types.

1.2. For all three graphs, we don't use filter because there aren't too many nodes. Furthermore, all nodes are adjusted by size, meaning the more important the node is, the bigger the node.

1.3. For second graph:

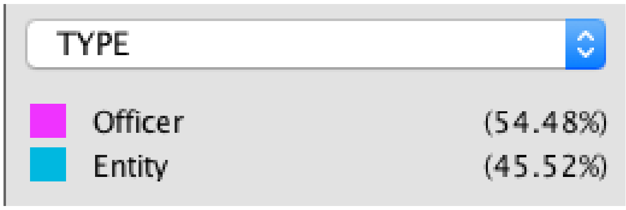
Line Type:



TYPE

🟨	SHAREHOLDER_OF	(87.13%)
🟦	BENEFICIARY_OF	(7.6%)
🟥	DIRECTOR_OF	(5.26%)

Node Type:

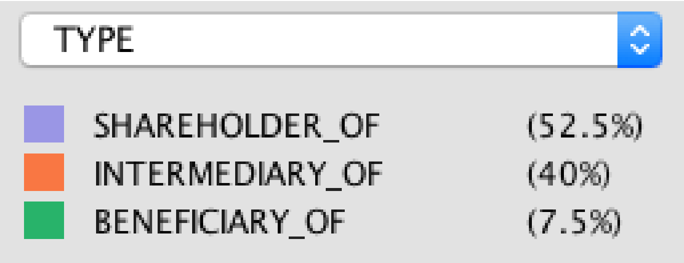


TYPE

🟪	Officer	(54.48%)
🟦	Entity	(45.52%)

1.4. For third graph:

Line Type:



TYPE

🟪	SHAREHOLDER_OF	(52.5%)
🟧	INTERMEDIARY_OF	(40%)
🟩	BENEFICIARY_OF	(7.5%)