Project Report on

EMPLOYEE ACCESS CHALLENGE

Team Members:

1) Mohammad Rafi – mxs146030
2) Sai Vamsi Muvva – sxm154231

1. INTRODUCTION

1.1 Motivation and Background:

When an employee at any company starts work, they first need to obtain the computer access necessary to fulfil their role. This access may allow an employee to read/manipulate resources through various applications or web portals.

1.2 Problem Description:

It is assumed that employees fulfilling the functions of a given role will access the same or similar resources. It is often the case that employees figure out the access they need as they encounter roadblocks during their daily work (e.g. not able to log into a reporting portal). A knowledgeable supervisor then takes time to manually grant the needed access in order to overcome access obstacles. As employees move throughout a company, this access discovery/recovery cycle wastes a nontrivial amount of time and money.

2. PROBLEM DEFINITION AND ALGORITHM

2.1 Task Definition:

The objective of this competition is to build a model, learned using historical data, that will determine an employee's access needs, such that manual access transactions(grants and revokes) are minimized as the employee's attributes change over time. The model will take an employee's role information and a resource code and will return whether or not access should be granted.

2.2 Algorithm Definitions

We are planning to model the following the classifiers:

Neural Networks, Naive Bayesian, Random Forest, Bagging, Boosting.

3. EXPERIMENTAL EVALUATION

3.1 Methodology

3.1.1 Criteria to evaluate method:

Different classifiers are trained using the train dataset and will be tested against test dataset and accuracy is found. The specific hypothesis that is tested is the action attribute having numbers 1 if the access is granted and 0 if not. The results can be tested against validation dataset to determine the accuracy and which classifier suits the best.

3.1.2 Data Validation:

To find the null values in a dataset:

```
apply(is.na(dataset),2,sum)
```

To find the redundant data are:

```
datset <- read.table(“path”,header=T)

n_occur <- data.frame(table(datset$id))

n_occur[n_occur$Freq > 1,]

datset[dataset$id %in% n_occur$Var1[n_occur$Freq > 1],]
```

3.1.3 Avoid Overfitting:

Random Forest: To avoid over-fitting in random forest, we need to optimize a tuning parameter that governs the number of features that are randomly chosen to grow each tree from the data with k-fold cross-validation, and choose the tuning parameter that minimizes test sample prediction error. In addition, growing a larger forest will improve predictive accuracy.

In all the classifiers k-fold cross validation can be used to prevent overfitting.

3.1.4 Description of the datasets:

train.csv - The training set. Each row has the ACTION (ground truth), RESOURCE, and information about the employee's role at the time of approval.

Number of instances in train.csv: 32,769

In train.csv, the data is unstable i.e., 96% of the data has class label as 1 and only 4% has class label 0.

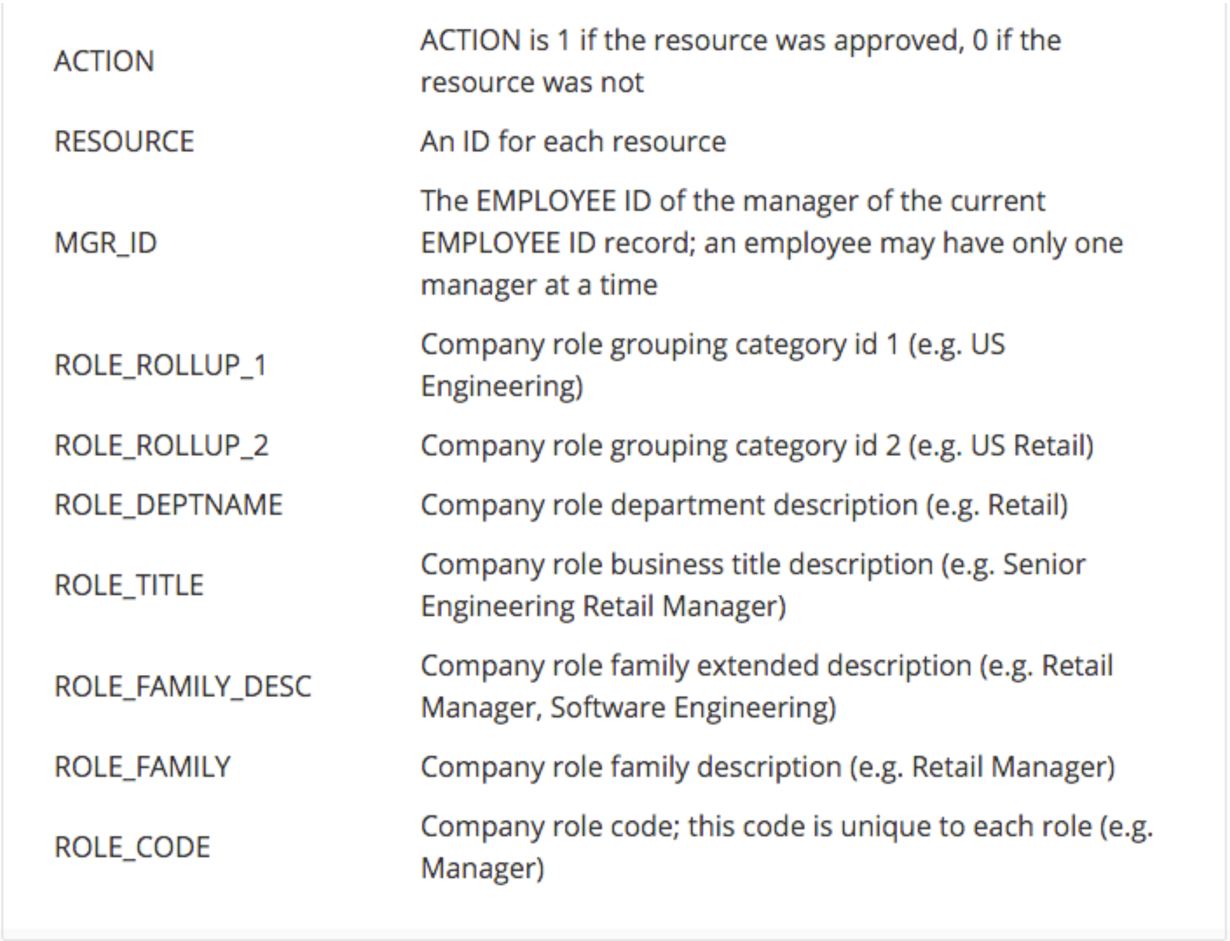
The test.csv has all the class labels as 0. In order to avoid the unstability, we extracted some data from test.csv and combined with the train.csv.

The train.csv is sampled 80/20 for train and test data respectively.

Attribute Description:

Number of attributes: 10.

Output variable: ACTION (real number)



ACTION	ACTION is 1 if the resource was approved, 0 if the resource was not
RESOURCE	An ID for each resource
MGR_ID	The EMPLOYEE ID of the manager of the current EMPLOYEE ID record; an employee may have only one manager at a time
ROLE_ROLLUP_1	Company role grouping category id 1 (e.g. US Engineering)
ROLE_ROLLUP_2	Company role grouping category id 2 (e.g. US Retail)
ROLE_DEPTNAME	Company role department description (e.g. Retail)
ROLE_TITLE	Company role business title description (e.g. Senior Engineering Retail Manager)
ROLE_FAMILY_DESC	Company role family extended description (e.g. Retail Manager, Software Engineering)
ROLE_FAMILY	Company role family description (e.g. Retail Manager)
ROLE_CODE	Company role code; this code is unique to each role (e.g. Manager)

3.1.5 Implementation:

Languages: R.

Packages: klaR, neuralnet, rpart, randomForest, ipred, adabag, caret, ada, gbm, class, e1071, Metrics

3.2 RESULTS

The accuracy for various classifiers using 10 fold cross validation is as shown:

Neural Network Accuracy: 94.2469098122997

SVM Accuracy: 94.2163894399512

Naïve Bayes Accuracy: 90.4776438272547

Random Forest Accuracy: 95.2540820998016

Bagging Accuracy: 94.7352357698764

Boosting Accuracy: 94.2469098122997

3.3 DISCUSSION

The F Scores are calculated for all the classifiers and we can see that the Random Forest has better F Score. The advantage of Random Forest is that it can more effectively account for non-linear dependencies in the data than the other model that we have used. Therefore we think that Random Forest is the promising approach.

4. RELATED WORK

The unbalance problem can cause suboptimal classification performance, which is pervasive and prevalent in many applications including risk management, text classification, and medical diagnosis/monitoring. In recent years, lots of researchers have made efforts to solve this problem both at the data and algorithmic levels. At the data level, different forms of resampling are applied. It is applicable to combine focused over and under sampling. Additionally, more accurate measures such as ROC curves and f-score metrics are applied.

5. CONCLUSION/FUTURE WORK

Simply applying models like Naive Bayes and SVM can only achieve low test accuracies (in our project, they only predict 1 or randomly predict 0 or 1) in an unbalanced set and that the use of On-hot encoder to pre-process dataset can greatly improve their performance. From the above results, we can conclude that the best overall test accuracy we can achieve is around 95% when using the Random Forest (F-score 0.86 is also the highest in this case), but the best test accuracy for the small sample set is only around 21%. To balance the test accuracy between the large and small sample sets, the Naïve Bayes model with combination of good features give us the best prediction. The test accuracy for the small and large sample set is around 78% and 94% respectively. The overall test accuracy is achieved to 90%. To further improve the test accuracy, the future work we can do is combining two/three/more features together and hash them to be a single new feature. In this way, it considers some features together to make a prediction which indicates in reality, some information together about the employee is better to predict the computer access.

Bibliography

https://www.kaggle.com/c/amazon-employee-access-challenge/data

http://cs229.stanford.edu/proj2015/240_report.pdf

http://www.slideshare.net/ShangxuanZhang/kaggle-talk-series-top-02-kaggler-on-amazon-employee-access-challenge