Prediction of employee access to systems based on role information

By Sarwar Bhuiyan

Data Science London

August, 2013

*Abstract*

*A common challenge during the onboarding of new employees is to determine which systems in a company network they should be given access to. Although role information such as role, line manager, and reporting hierarchy may be available, the mapping between such information and which systems they should be given access to is left to a manual process where managers have to determine on an on-demand or periodic basis whether their direct reports should have access to certain systems. Amazon presented a challenge on Kaggle (“Amazon.com – Employee Access Challenge) to produce a prediction model for this problem. The competition carried with it a reward of $5000 for the top performer. The available information consisted of 2 years of employee role information and their access to resources (systems) within Amazon. The information was analyzed using various Machine Learning algorithms and their results submitted to Kaggle. The results compared favourably to others submitted on the challenge. A study of the other solutions gave certain insights to the techniques and tricks required for category heavy data sets of this type. The challenge presents many different ways of tackling it and some future opportunities have been identified.*

# Introduction

When a new employee joins a company, they may need to access various resources (systems) within the company. This enabling of access often has to be request at the point of and time is wasted going through approval workflows by various approvers. Amazon presented a challenge on Kaggle.com (<http://www.kaggle.com/c/amazon-employee-access-challenge>), a competition website for machine learning challenges, to produce a prediction model that can predict whether role information can be used to automate the process of giving access to internal systems. The following paper presents the process I used in analyzing this data and the machine learning techniques used to produce predictions for the amazon challenge.

# Motivation

From personal experience in the companies I worked, I noticed that a lot of time was wasted in getting me (and other colleagues) access to specific systems. Often this required the raising of requests through an internal ticketing system and then having to chase multiple times via the ticketing system and email to get approval for access. Before getting approval, employees are blocked and not fully able to carry out their work. Although nobody ever calculates this, the amounts to millions in terms of man-hours wasted in raising requests, manual approvals, escalations, and recertification when people leave. An automated system that can approve most, if not all, requests automatically would be extremely beneficial to large companies with big IT real estates.

# Initial Analysis

An initial look of the data presents a table like the following:



Figure Training Data

The first column contains 0 or 1 depending on whether a user has access or not. The second column contains the IDs of systems that users need access to. The remaining 8 columns contain numeric IDs of role categories within the company such as Manager (MGR\_ID) or Department (ROLE\_DEPTNAME).

Using some simple R commands, I looked at the frequencies of the features of the training and test data set.

> train <- read.csv(“train.csv”,header=True)

> sapply(train, function(x) length(unique(x)))

|  |  |
| --- | --- |
| **Feature** | **Frequency** |
| ACTION | 2 |
| RESOURCE | 7518 |
| MGR\_ID | 4243 |
| ROLE\_ROLLUP\_1 | 128 |
| ROLE\_ROLLUP\_2 | 177 |
| ROLE\_DEPTNAME | 449 |
| ROLE\_TITLE | 343 |
| ROLE\_FAMILY\_DESC | 2358 |
| ROLE\_FAMILY | 67 |
| ROLE\_CODE | 343 |

Figure Frequency of training table features

|  |  |
| --- | --- |
| **Feature** | **Frequency** |
| ACTION | 2 |
| RESOURCE | 4971 |
| MGR\_ID | 4689 |
| ROLE\_ROLLUP\_1 | 126 |
| ROLE\_ROLLUP\_2 | 177 |
| ROLE\_DEPTNAME | 466 |
| ROLE\_TITLE | 351 |
| ROLE\_FAMILY\_DESC | 2749 |
| ROLE\_FAMILY | 68 |
| ROLE\_CODE | 351 |

Figure Frequencey of test table features

# Submission Assessment

The submission data required for the Kaggle.com did not require the predictions. It required calculating a metric known as Area Under Curve of an ROC (Receiver Operator Characteristic). The ROC curve is particularly useful in assessing binary classification problems and represents the relationship between the True Positive Rate (true postives/total positives) versus the False Positive (false positives/all negatives) Rates and the AUC is a number between 0 and 1. A value of 0.5 represents a model that is no better than randomly guessing. Most models will have an AUC between 0.5 and 1.

For more information, see

<https://www.kaggle.com/wiki/AreaUnderCurve>

# Different Techniques Applied

The problem at hand is a classification problem with all features being categorical in nature. The following classifiers from the python scikit-learn package were used:

1. Logistic Regression
2. Decision Tree
3. Random Forest

In order to get the best possible fit, all features were used with the parameters of the classifiers tweaked to achieve the highest AUC in 10-fold cross validation.

The following results were achieved:

|  |  |  |
| --- | --- | --- |
| **Classifier** | **AUC** | **Best Params** |
| Logistic Regression | 0.87057 | C=2.2 |
| Decision Tree | 0.78284 | Criterion=entropy,  Min\_sample\_split=61 |
| Random Forest Classifier | 0.87139 | n\_estimators=200, min\_samples\_split=15, min\_density=0.1,compute\_importances=True |
| Blended (simple sum of 3 classifier AUCs above) | 0.87271 |  |
| Random Forest and Extra Features created by tuples | 0.87678 |  |

The first 4 results were done using very little preprocessing of the initial data. The training and test data were loaded from CSV files and run through the classifiers in the scikit-learn package. The cross\_validation package’s StratifiedKFold was used to run a cross validation loop to calculate the AUC under ROC using the metrics package.

Each classifier produced different AUC with the default parameters. By some tweaking, they were raised to between 0.78 and 0.87.

The Random Forest classifier gave better results as it is using ensemble techniques using Decision Tree underneath the covers.

A slightly higher AUC was achieved by summing the results from the three different classifiers (Logistic, Decision Tree, and Random Forest). This is a rather naïve blending mechanism but illustrated the benefit of combining the results of multiple classifiers.

The last row was achieved by using a technique used by a lot of competitors on the Amazon Kaggle challenge. The idea is to combine the 9 or so existing feature columns in various combinations of 2-tuple or 3-tuple combinations (representing the AND combination of the features and their results). This has the benefit of including the effect of combinations of the features on the result rather than each individual feature.

# Final Best Solution

As seen above the final best solution in the experiments performed above showed that the Random Forest Classifier with the Combined Features performed the best.

The winning submissions on Kaggle used a variety of other to achieve an AUC of slightly higher than 0.92. These included the combination of features to create new features up to many levels (i.e. combinations of combinations) as well as greedy feature selection loops to select the most pertinent combination of features.

# Practical Applications and Future Opportunities

There is definitely potential for these algorithms to in practical applications in industry. There are many certification/recertification tools used by managers to approve (or re-approve) their direct reports for access to various systems in their IT real estate. The current landscape is very much driven by manual business processes, which naturally uses up precious time. There can be efficiencies achieved where perhaps part of the process can be automated where there is a user’s access level can be predicted with a high level of confidence. This can either be built into the approval applications or as separate applications, which can be used to score user’s eligibility for approval. Given a certain threshold, they can even be auto-approved for access to a certain group of applications as soon as they join a company.

# Conclusion

This paper described a problem posed by the Amazon Kaggle competition where a prediction model was required to be able to predict whether employees should have access IT systems based on employee or role attributes. A number of classification algorithms were used and the results submitted to Kaggle. The metric used to evaluate the results was the Area Under Curve of the ROC curve. Some experimentation was carried out with regards to classifier parameters, blending of results, and feature combination. The best result achieved an AUC or 0.87678. There is potential to improve this a lot more by using techniques such as feature selection, blending, and hyper parameter optimization. The prediction models also have the potential to be used in applications in industry.