MS4110 Project - License approval

CE15B035 ME15B082 ME15B086 ME15B087 NA15B047

November 11, 2018

Abstract

The main aim of this project is to predict whether a permit would gain approval given various details about it. This report contains elaborations of our methodology, results, conclusions and observations while solving this problem.

1 Introduction

A building license is an official approval document issued by a governmental agency that allows you to proceed with a construction on a project. Given a dataset consisting of various details like the kind of permit, filing date, location etc, our task is to predict if it is going to be approved or not. This report explains methods used to perform predictive analysis on license approval dataset.

2 Dataset and Preprocessing

2.1 About the dataset

The dataset contains around 170k data points with 25 features like ids, kind of permits, filing date, location, proposed use, units of use and one column indicating the approval status. There are 7 categories of current status of permit : 'complete', 'approved' , 'issued' , 'expired' , 'disapproved', 'evoked' and 'canceled'.

After being filled, a permit is first checked and approved. On approval the permit is issued, after which the construction work starts. The permit can later get expired or the construction may get over, after which the permit is said to be complete. In case of mistake in issuing a permit or in case of a violation of rules, the permit can be revoked.

Therefore we categorized 'complete', 'approved', 'issued' and 'expired' under the super-category 'Approved' and the remaining as 'Disapproved'.

2.2 Feature Selection

We made use of the location coordinates of the buildings. Hence, the address features like street name, street number, block, etc. became obsolete and were removed. The features 'Record Id' and 'Permit number' which were mere numberings were also removed.

In most of the data points the Permit Creation date was the same as the filed date. Moreover, there were countless errors in the Permit Creation dates. Many of the Permit Creation dates were before corresponding Permit Filing dates, which is logically absurd. Hence,we only made use of Permit Filed Dates.

2.3 Feature Preprocessing

The dataset contained numerous redundant features as well as a high number of missing values. The following preprocessing steps were used to deal with these challenges:

1. All categorical variables were converted into one-hot vectors with a separate class for missing data for each variable. The Zip codes were also treated as a categorical variable.
2. The Filed date was converted into three different variables: Filed Day, Filed month and Filed Year
3. The coordinates were converted into two features : Latitude and Longitude.
4. All the continuous variables were imputed using mean imputation and then were standardized

3 Predictive Modeling

3.1 Challenges

The major challenge in Predictive Modeling was the extreme (99:1) class imbalance between approved and disapproved classes. This lead to almost 100% accuracy but a very low f1 score. In order to deal with this issue we changed our model selection metric from accuracy to f1 score. Up sampling of minority class values is done to tackle the issue of high class imbalance using re-sample function from sklearn library.

Ensemble techniques are used for the predictive analysis after performing primary feature selection and data imputations. Observing different aspects of the dataset like sparsity, presence of categorical variables, XGBoost and random forest classifiers are chosen for classification. The chosen ensemble methods were tried over label encoded and label encoded up-sampled data. To increase the score further, one-hot encoded up-sampled data was also used.

3.2 Modeling

Using imbalanced data with label encoding for categorical variables, only random forest performed well. In particular, the F1- score for '0' class was very low. Up-sampling was done over the training data and the score was obtained over the original test data. Random Forest started performed quite well after up-sampling '0' class for categorical variables. Thus, random forest was chosen further for modeling over the new dataset

Furthermore, random forest was used over one-hot encoded up-sampled data which increased F1- score marginally.

3.3 Hyper parameter tuning

Tuning the hyper-parameter n-estimators of the Random Forest Algorithm. Optimal: 50 decision trees.

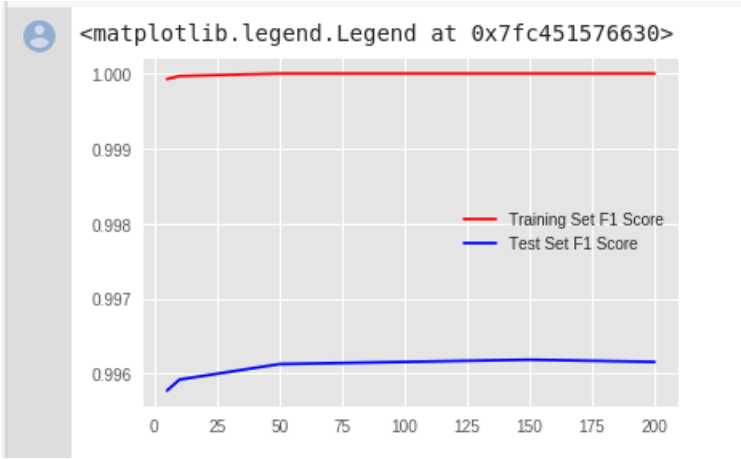

Figure 1: Tuning n-estimators

4 Results

Performance on test split data :

Class	Precision	Recall	F1-score	support
0	0.03	0.48	0.06	312
1	0.99	0.86	0.93	34115
avg/total	0.99	0.86	0.92	34427

Table 1: XGBoost with label encoded data

Class	Precision	Recall	F1-score	support
0	0.75	0.16	0.27	312
1	0.99	1.00	1.00	34115
avg/total	0.99	0.99	0.99	34427

Table 2: Random forest with labeled data

Class	Precision	Recall	F1-score	support
0	0.62	0.21	0.31	312
1	0.99	1.00	1.00	34115
avg/total	0.99	0.99	0.99	34427

Table 3: Random forest with up-sampled data

Class	Precision	Recall	F1-score	support
0	0.61	0.23	0.33	312
1	0.99	1.00	1.00	34115
avg/total	0.99	0.99	0.99	34427

Table 4: Random forest with new data

Confusion matrix:

		Prediction outcome		
		p	**n**	**total**
	p′	71	241	P′
actual value	**n′**	46	34069	N′
	total	P	N	

5 Conclusions:

1. In conclusion, the best performing algorithm is the Random Forest Classifier, trained with One-Hot Encoded Categorical Variables and Random Up-Sampling to solve the issue of Class Imbalance. A comparison was performed between the Random Forest Classifier and the XGBoost Classifier. The Random Forest Classifier did a better job at predicting not approved licenses (class 0) as it is a bagging algorithm, i.e. it uses an ensemble of decision trees that fit the data parallely. The XGBoost CLassifier is a boosting algorithm, it uses an ensemble of decision trees that fit the data sequentially, i.e. on the residuals of each subsequent fitted decision tree , hence it is unable to capture information about class 0 well due to severe class imbalance (1:99 ratio).

2. We obtained a maximum of 0.33 F1 score for '0' Class, i.e. disapproved permit. The prominent culprit behind such a low F1 Score for this particular class is extreme class imbalance. We tried random upsampling of minority class. It was nothing but replication of the examples with '0' class, which increased the probability of the '0' class observations being chosen at each iteration of a bootstrapped dataset in the random forest algorithm. Thus, we could make sure that the at each split in the tree, the algorithm is able to find the observations with '0' class.

3. Alternative approaches for dealing with class imbalance could include have synthesizing new observations with '0' class, which could have given a better F1 score, but at the cost of adding/synthesising new data on the basis of probability distribution (SMOTE Algorithm).