Han Chen, Borui Wang

Prof. Pujara

TIM245

March 9, 2016

# San Francisco Crime Classification

**ABSTRACT**

The goal is to classify the type of crimes committed with City San Francisco, based on the data points given on the dataset. This study will help police officers identify top possibilities of categories with information of time, areas, coordinates and so on. Another focus on this project is to visualise dataset using Python Matplotlib and Google Map.

During the modelling, Bogomolov's[1] paper has enlighten us to Random Forest Model(RF). RF has the best performance among all models, and also better computation time compared with kNN classifiers.

## 1. INTRODUCTION

As in all cities, crime is a reality San Francisco. The challenge we tackle today involves attempting to guess the class of a crime committed within the city, given the time and location it took place. Such studies are representative of efforts by many police forces today: Using machine learning approaches, one can get an improved understanding of which crimes occur where and when in a city — this then allows for better, dynamic allocation of police resources. To aid in the SF challenge, Kaggle[2] has provided about 12 years of crime reports from all over the city — a data set that is pretty interesting to comb through.

Here, we outline our approach to tackling this problem. The main steps in this project includes Data-Processing, Building Models, Comparison between Classifiers, and Data Visualisation. We use multiple classifiers including Naive Bayes Classifier, KNN Classifier, Random Forest Classifier, SVM and Logistic Regression to predict. The main comparison method is log loss.

The difficulty of this project is at

a. The complexity and large amounts of data points. (Totally 1,600K data points, with half in training dataset, half in test dataset.

b. Low accuracy after classifying. (Best log loss result at kaggle.com is 2.05.)

c. Computation time through all data points. (Take 9 hours for a single classifier among all data points for training).

Our progress since presentation:

a. Improve log loss from 2.58 to 2.32.

b. Creating new attributes to the datasets, including isNight, StreetNo, ContainOf, PCA_X, PCA_Y, Year, Month.

## 2. Summary of Dataset

The dataset includes data from January 2013 to May 2015, in format of .csv file.

For training.csv, the total number is 878,049, and it include all odd weeks.

The attributes are as following,

| | |
|---|---|
| a. Date | — Year/Month/Day Hour/Minute/Second |
| b. Category | — Kinds of Crime |
| c. Descript | — Description of the crime |
| d. DayofWeek | — Monday, Tuesday, … Sunday |
| e. PdDescript | — Police Description about the area |
| f. Resolution | — Results |
| g. Address | — Street Address |
| h. X | — Coordinates |
| i. Y | — Coordinates |

The amounts between different categories are very large, with top 5 categories occupy nearly 67% precent of total crimes. And the least one only has 51 cases in 12 years. The detail information is given in Appendix Figure 1.

Besides, we also notice crimes amount also differ significantly in different hours. The detail information is given in Appendix Figure 2.

Another observation for the dataset is that the attribute 'Address' can mainly two kinds. One contains 'of', one doesn't contain 'of'. Meanwhile, we noticed that number of crimes doesn't change much on different weeks.

## 3. Data Processing

a. Handling Missing Value and Noise

For the purpose of analysis, we use train_test_split in Scikit-learn module, to separate train.csv into data_train dataFrame and data_test dataFrame, with test_size = 0.2.

For missing values, there're a few instance named 'None', and we treat it as a category during analysis.

For coordinates out of city San Francisco, we eliminate data points from dataset.

b. Features.

5 new features are created as following.

| | |
|---|---|
| AddressContainOf | — Binary attribute. 1 means 'Address' contains 'of'. |
| StreetNo | — Street number in attribute 'Address'. |
| Hour, Year, Month | — Hour when crime happens. 'Dates' -> hour, Year, Month. |
| Evening | — Binary attribute. (If hour is in [18,19,20,21,22,23,0,1,2,3,4,5,6]) |
| PCA_X, PCA_Y | — Coordinates after PCA |

## 4. Evaluation

The evaluation metric used by Kaggle is the Multi-class Log Loss. Based on the definition given on https://www.kaggle.com/wiki/LogarithmicLoss.

The logarithm of the likelihood function for a Bernouli random distribution.In plain English, this error metric is used where contestants have to predict that something is true or false with a probability (likelihood) ranging from definitely true (1) to equally true (0.5) to definitely false(0). The use of log on the error provides extreme punishments for being both confident and wrong. In the worst possible case, a single prediction that something is definitely true (1) when it is actually false will add infinite to your error score and make every other entry pointless. In Kaggle competitions, predictions are bounded away from the extremes by a small value in order to prevent this.

$$logloss = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{ij}\log(p_{ij})$$

$$logloss = -\frac{1}{N}\sum_{i=1}^{N}(y_i\log(p_i) + (1-y_i)\log(1-p_i))$$

## 5. Exploration

Before the whole data processing steps, in order to get insight of our model and dataset, we did some simple experiments of our dataset. We only include days, district and hour as our features, because the other instances are very distributed. We used logistic regression which is good for multi-class classification problem and Naive Bayes which is a faster learner and random forest which is good for multi-class classification to built model. But the result is terrible. We can see the result from the following table.

| | Logistic Regression | Naive Bayes | Random Forest |
|---|---|---|---|
| **Accuracy** | 0.22 | 0.22 | 0.27 |
| **Log-loss** | 2.58 | 2.58 | 14.7 |

**Table1. Initial model's results**

After that, we started thinking what was the issue for pool preforms. We think the reason is that we didn't use enough data for our model, such as location X, Y and the information of street. So we should include those features into our training set.

Therefore, we added new features, as mentioned in data processing part, according to the conclusion we got from data visualization. Because X, Y are longitude and latitude which are high dimensional, so we used PCA to principal components. After adding those features, we did the experiment again.

## 6. Modeling

### 6.1 Naive Bayes Classifier

Naive Bayes Classifier is used as the first model for its high efficiency and speed.

We find that as the size of training dataset increases from 300,000 to 800,000, the performance for Naive Bayes drops. We also find if the features just include 'Day', 'Month', 'Year', 'Hour', 'PdDescript', which is used at the presentation, Naive Bayes Classifier has a much better prediction(log loss=2.58).

One possible reason is that the features are correlated, while Naive Bayes Classifier initially treat all features independent.

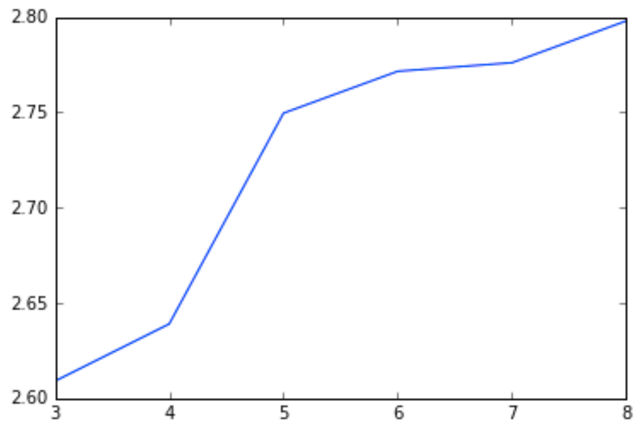


**Figure 4: Naive Bayes Classifier Plot**

### 6.2 Random Forest

A random forest is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and use averaging to improve the predictive accuracy and control over-fitting.

Our purpose using random forest classifier is to decrease overfitting, and the most important parameter is max_depth to hyperplane when training. We selectively choose different max_depth in each training, and compare the results. The results are giving as following.

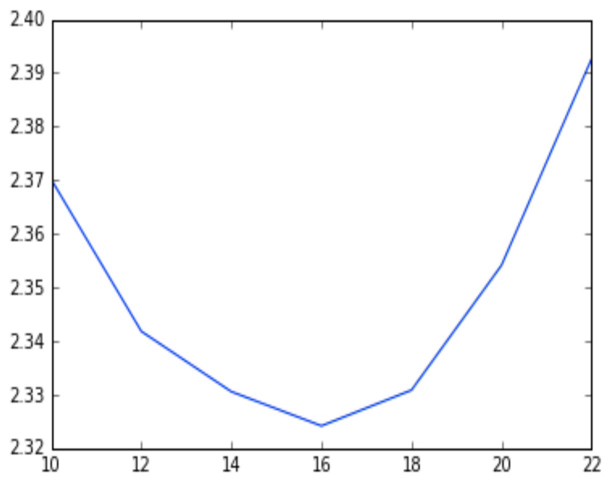


**Figure 5. RandomForest Plot**

We noticed that as max_depth increases, log loss decreases at the beginning, and then log loss increases. The best result is at max_depth=16, log loss=2.32417

### 6.3 KNN Classifier

In this model, classification is based on the majority vote of its k nearest neighbour. In this case, KNN will classifier based on different features in Part. 3. However, our result shows poor performance. Compared with Random Forest, KNN takes much longer time to compute, and KNN's log loss is higher than Random Forest or Naive Bayes. A possible reason is that our dataset is highly crewed and large, which means the distribution between different crime categories is concentrated on top 5 crimes. Also, the features 'X' and 'Y' also lower the computation speed. The last reason is because KNN can't decide which feature is more important.

The following plot shows the performance for KNN, from training dataset size range from 150,000 to 250,000.

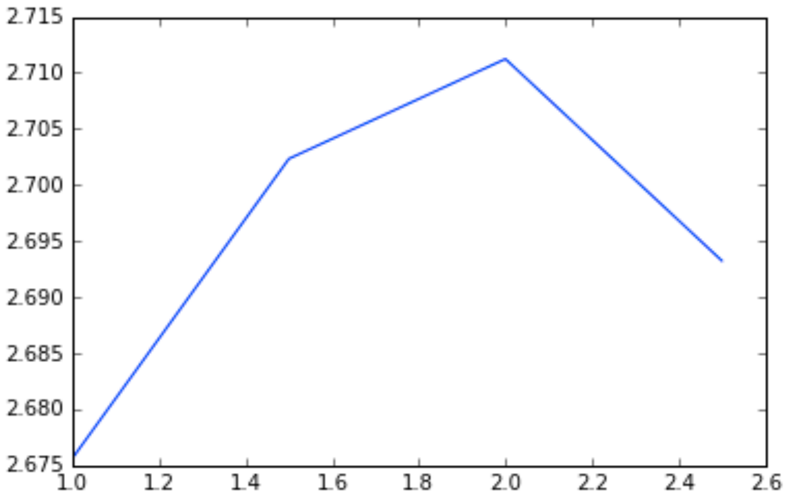


**Figure 6: KNN Classifier Plot**

### 6.4 Logistic Regression Classifier

Logistic regression is a statistical method for analyzing a dataset in which there are one or more independent variables that determine an outcome. The log-loss of this module is 2.46. The reason why logistic regression cannot get better performance is because our data set is highly crewed and large, and logistic regression cannot handle large number of categorical features/ variables well.

### 6.5 SVM

SVM is also a classical classification algorithm. Given a set of training examples, each marked for belonging to one of two categories, an SVM training algorithm builds a model that assigns new examples into one category or the other.

We used SVC in sklearn model to train our model and the log-loss of SVM is 2.4.

## 7. Experiment Result

This part mainly focus on analysis the result of Random Forest, which has the best performance among all classifiers. Because our training set has lots of instances, and random forest has benefit of learning large instances, creating a high accuracy model. From the graph we can see, if the depth of tress is too large, the performance will decrease, in this case, we need to use pruning for our decision tree to prevent overfitting. In our case, depth 16 has the best performance.

The performance metrics are Accuracy and Log Loss, with a overview table as following.

| | Naive Bayes | Random Forest | KNN | Logistic regression | SVM |
|---|---|---|---|---|---|
| **Log-loss** | 2.58 | 2.32 | 2.65 | 2.46 | 2.4 |

**Table 2: Final result**

From the table we can conclude, random forest has the best performance from the models we picked.

## 8. Data Visualisation

Despite the plots used in data processing, I have developed a .html file for data visualisation on Google Map[3]. For the reason that data points on one day are too many for showing, the map will only select top 5 popular crimes.

The steps for building Google Map is as following,

a. Extract ‘Descript’, ‘Category’, ‘Y’, ‘X’ from dataset to a single .coor file.

b. Handling format using .cpp program

c. Paste .coor to .html file, where I use Google Map API.

d. Open in safari, or any browser.

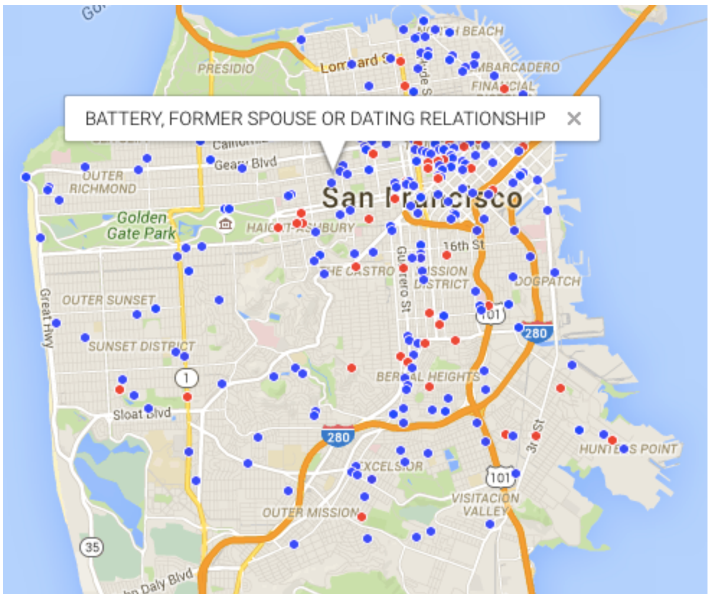


**Figure 7: GoogleMap Sample**

For detail, please follow the steps to open the .html file using terminal.

1. cd HanChenWork/GoogleMap
2. open data_20150511.html

This will give you a GoogleMap for data on 2015/05/11, which there'e data on 2015/05/12 and 2015/05/13. By clicking on the colourful dots, a piece of information will display.

| | |
|---|---|
| Red Dot: | WARRANTS |
| Green Dot: | LARCENY/THEFT |
| Black Dot: | ASSAULT |
| Yellow Dot: | OTHER OFFENCES |
| Blue Dot: | DRUG/NARCOTIC |

Plots for different features are as following,

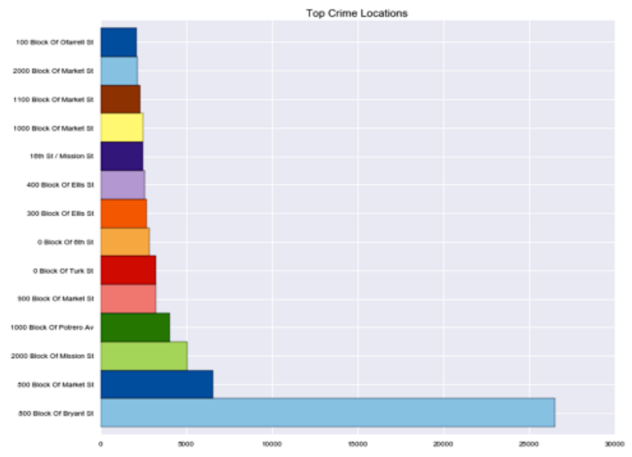


**Figure 8: Top Crime Locations**

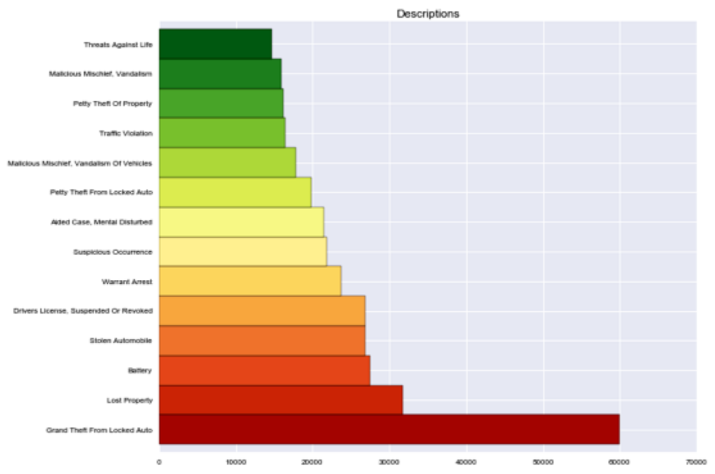
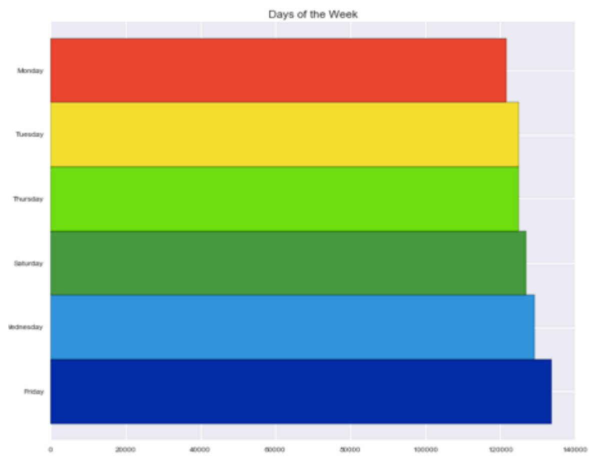


**Figure 9: Descriptions**



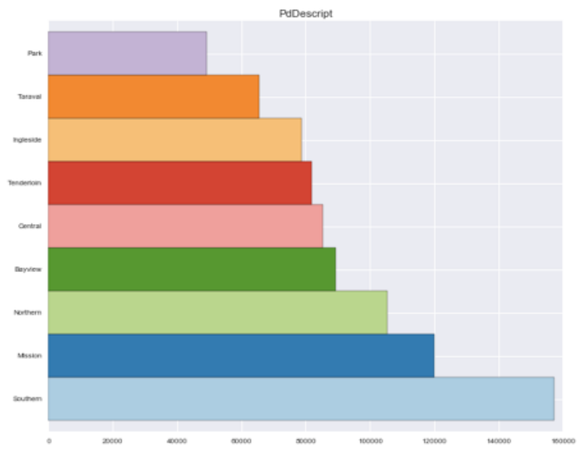


**Figure 10: Days of Week** **Figure 11: PdDescript**

## 8. Conclusion

In our project, we tried four different machine-learning algorithms to train our data and compare the result among them. The motivation for us to pick those four algorithms are:

1. Too learn a model faster. Because our training data set is too large, if we use complex algorithm will take lots of time.

2. Our problem is classification problem, and Navies Bayes, Random Forest, KNN and logistic regression are classical algorithms for classification problem.

The data processing part is pretty important in whole data mining process. As the two result shows, if we don't pick correct features, the performance is pretty bad. And also the PCA is a good tool for find out the principal components. If we didn't use PCA to process the X, Y in our data set, the performance is even worse. So before we choose algorithm, we should handle our data carefully and choose correct features to built the model.

By comparing those algorithms, we can deeper insight of each algorithm. Navies Bayes can learn a model really fast and the accuracy is not bad. Random forest is good for large instances and multi-class classification. KNN is not good for highly crewed and large data set. Logistic regression can be used for multi-class classification problem but the accuracy in large instances data set is worse than random forest.

For random forest we can see, the depth of the tree has huge influence of performance. If we need to prevent overfitting, we should consider pre-pruning and post-pruning strategy.

We upload all our source code and documents on Borui's repository. Here is the link: https://github.com/boruiwang/SanFranciscoCrimeClassification

## 9. Reference

[1]Once Upon a Crime: Towards Crime Prediction from Demographics and Mobile Data. Andrey Bogomolov. http://arxiv.org/pdf/1409.2983.pdf

[2]San Francisco Crime Classification. https://www.kaggle.com/c/sf-crime

[3]Google Map Javascript API https://developers.google.com/maps/documentation/javascript/examples/#basics

## Appendix

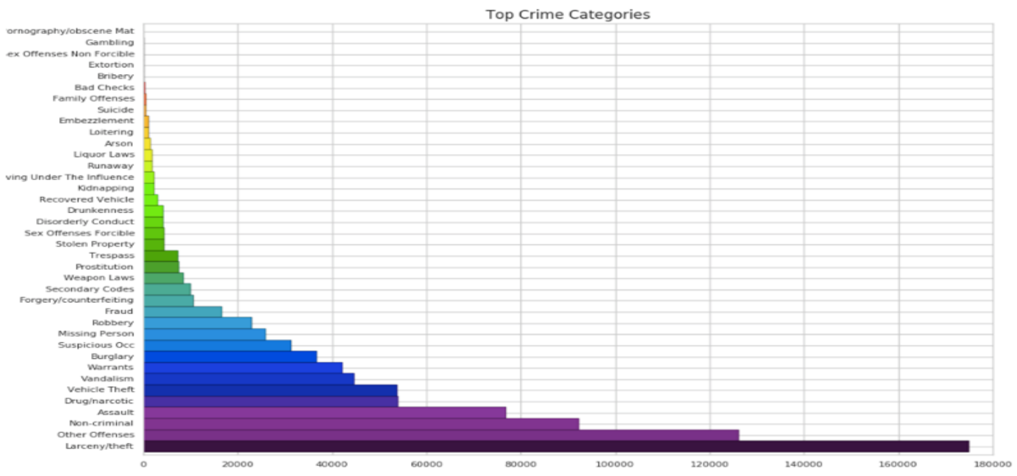
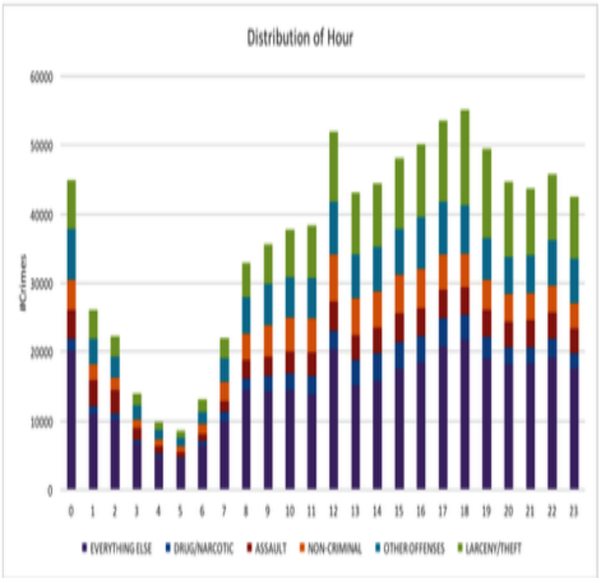


**Figure 1. Top Crime Categories**



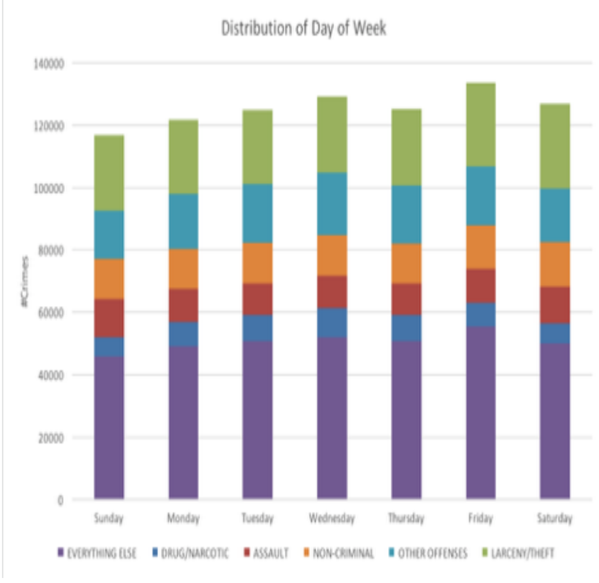


**Figure 2. Distribution of Hour**

**Figure 3. Distribution of Day of Week**