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

**CS 6375.501 – Machine Learning**

**Project Report**

*Asha Mary Thomas - axt143530*

*Biligiri Vasan S R - bxs152830*

*Somasundaram Ardhanareeswaran - sxa146230*

Contents

[**San Francisco Crime Classification** 1](#_Toc449292184)

[**CS 6375.501 – Machine Learning** 1](#_Toc449292185)

[**1.** **Introduction** 3](#_Toc449292186)

[**2.** **Problem Definition and Algorithm** 3](#_Toc449292187)

[**Problem Description:** 3](#_Toc449292188)

[**Algorithms used to build Models:** 3](#_Toc449292189)

[**Naïve Bayes Classifier** 3](#_Toc449292190)

[**Ada Boosting** 4](#_Toc449292191)

[**Random Forest** 4](#_Toc449292192)

[**3.** **Experimental Evaluation** 4](#_Toc449292193)

[**Dataset Description** 4](#_Toc449292194)

[**Methodology:** 5](#_Toc449292195)

[**Dataset Analysis & Feature Selection and Feature Engineering:** 5](#_Toc449292196)

[**Eliminating Outliers:** 5](#_Toc449292197)

[**Clustering Details and Techniques:** 7](#_Toc449292198)

[**Filtering Less Frequent Classes:** 8](#_Toc449292199)

[**Accuracy and Log Loss calculation:** 9](#_Toc449292200)

[**Results** 10](#_Toc449292201)

[**Predictions of Categories on the Test Data:** 11](#_Toc449292202)

[**Discussion** 13](#_Toc449292203)

[**4.** **Related Work** 14](#_Toc449292204)

[**5.** **Future Work** 14](#_Toc449292205)

[**6.** **Conclusion** 15](#_Toc449292206)

[**7.** **References** 16](#_Toc449292207)

# **Introduction**

The project is based on Kaggle Competition to predict the category of crimes that occurred in the San Francisco city. SF city is known for its tech scene. But, with rising wealth inequality, housing shortages, etc. there is no scarcity of crime in the city. The Kaggle competition's dataset provides almost around 12 years of crime reports from across all of San Francisco's neighborhoods. The aim of the project is to predict the category of crime that occurred given time and location.

In this project we are training the data on three classifiers and comparing the results such as log loss and accuracy noticed with each of the classifiers. As the test data provided in the competition does not have the target category, we would be predicting the category of crime associated with it given the data.

# **Problem Definition and Algorithm**

## **Problem Description:**

The dataset contains incidents derived from SFPD Crime Incident Reporting system. The data ranges from 1/1/2003 to 5/13/2015. The training set and test set rotate every week, meaning week 1,3,5,7 belongs to test set and week 2,4,6,8 belongs to training set. The aim is to predict the category of crime that occurred given time and location.

## **Algorithms used to build Models:**

### **Naïve Bayes Classifier**

This classifier makes and assumption of independence between every pair of features. The performance of Naive Bayes learners and classifiers is fast compared to more sophisticated methods. The idea behind is that the decoupling of the class conditional feature distributions helps to estimate each distribution independently as a one dimensional distribution and thus helping to alleviate problems stemming from the curse of dimensionality.

We have used Gaussian Naive Bayes classifier to build the classifier for the project. It implements the Gaussian Naive Bayes algorithm for classification and the likelihood of the features is assumed to be Gaussian:

P(x_i \mid y) &= \frac{1}{\sqrt{2\pi\sigma^2_y}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma^2_y}\right)

### **Ada Boosting**

The AdaBoost classifier begins by fitting a classifier on the original dataset and then fits additional copies of the classifier on the same dataset. The weights of incorrectly classified instances are adjusted so that subsequent classifiers focus more on the incorrect cases. The AdaBoost classifier used in the project implements the algorithm known as AdaBoost-SAMME.

### **Random Forest**

In random forests classifier, each tree in the ensemble is built from a sample drawn with replacement from the training set. When splitting a node during the construction of the tree, the split that is chosen is no longer the best split among all features. But, the split is the best split among a random subset of the features. Due to this randomness, the bias of the forest usually slightly increases but, due to averaging, its variance also decreases, and compensates for the increase in bias, hence giving an overall better model. The scikit-learn implementation used in the project combines classifiers by averaging their probabilistic prediction, instead of letting each classifier vote for a single class.

# **Experimental Evaluation**

## **Dataset Description**

The total number of training data available is approximately around 878 thousand records and each records has 9 columns associated with it. The testing data has approximately 884 thousand records.

***Total instances:***

Training Data: **878049**

Test Data: **884262**

***Columns present are:***

**Dates**: timestamp of the crime incident

**Category**: category of the crime incident (only in train.csv). Class variable we are going to predict.

**Descript**: detailed description of the crime incident (only in train.csv)

**DayOfWeek**: the day of the week

**PdDistrict**: name of the Police Department District

**Resolution**: how the crime incident was resolved (only in train.csv)

**Address**: the approximate street address of the crime incident

**X**: Longitude

**Y**: Latitude

## **Methodology:**

### **Dataset Analysis & Feature Selection and Feature Engineering:**

The fields Descript and Resolution is not present in the test data and so we have adopted proper representations of Dates, DayOfWeek, PdDistrict, X, Y as our feature set. Since, X and Y coordinates along with the Police department district can together serve as a feature equivalent to the address found in the dataset, we have eliminated “address”.

* **Dates**: We extracted year, month, day and hour from this field and used these representation as our features.
* **Day of Week**: We represented this column with numerical value 1 – 7
* **PdDistrict**: We have represented the 10 police department districts with numerical values ranging from 1 to 10.
* **X, Y**: The X and Y coordinates indicate the latitude and longitude for a crime location. As the X and Y values is varying lot for small decimal points, we have clustered that data into 1000 clusters. The X, Y fields are then replaced with the cluster value in the dataset identified by name “**cluster\_ids**”.

### **Eliminating Outliers:**

The dataset has few outliers in the latitude(Y) and longitude(X) that does not belong to the SFO neighborhood. We have cleansed the data to eliminate these outliers using simple filtering techniques of latitudes and longitudes that do not belong to the SFO region.

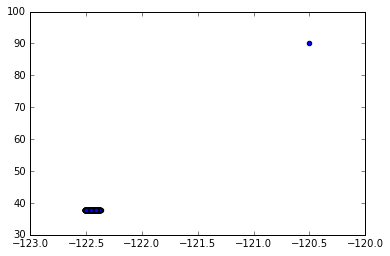


Figure 1: Plotting location coordinates indicating the outliers.

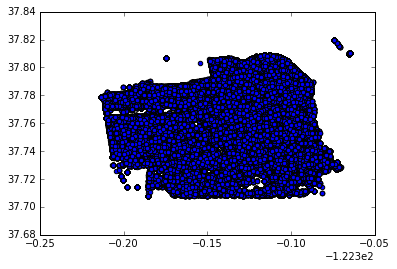


Figure 2: Plotting location coordinates after removing outliers.

In the first plot, we notice that the outliers existence diminished plot of the coordinates of the actual SFO region. However, after filtering, we can observe the outline of the SFO region clearly for the same plot.

### **Clustering Details and Techniques:**

The X and Y values were clustered together using the K-Means clustering algorithm. The number of clusters were chosen to be 1000, since this value gave better accuracy when compared to lower and higher cluster centers.

The clusters were built on the training data and the cluster IDs were predicted using the same model for the testing data.

Following are the plots of the X and Y values before and after clustering:

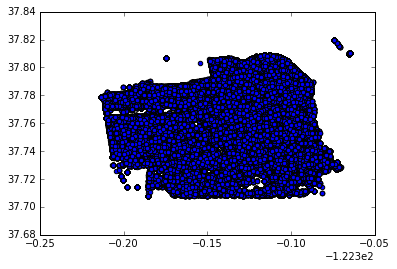


Figure : Plot of the coordinates of the SFO region before clustering.

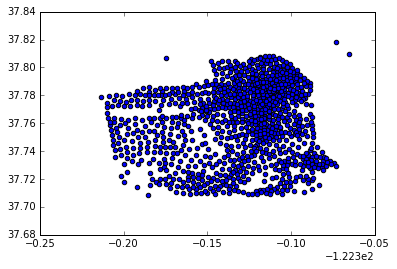


Figure : Plot of the coordinates of the SFO region after clustering.

### **Filtering Less Frequent Classes:**

There are **39 types** (classes) of crimes in total. We could analyze that the distribution of crimes satisfies a property that several most common crime categories make up the majority of all crimes. We have seen that the top **22 categories constitute approximately 98%** of the data and so we can safely remove the other less frequent classes from our dataset.

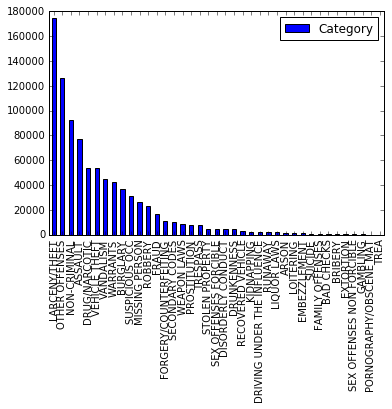
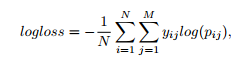


Figure : List of categories in the descending order of their occurrence

### **Accuracy and Log Loss calculation:**

The technique used for calculation of accuracy is k-Fold Cross Validation. In this technique, the original sample is randomly partitioned into *k* equal sized subsamples. Of the *k* subsamples, a single subsample is retained as the validation data for testing the model, and the remaining *k* − 1 subsamples are used as training data. The cross-validation process is then repeated *k* times (the *folds*), with each of the *k* subsamples used exactly once as the validation data. The *k* results from the folds can then be averaged (or otherwise combined) to produce a single estimation. The advantage of this method over repeated random sub-sampling (see below) is that all observations are used for both training and validation, and each observation is used for validation exactly once. 10-fold cross-validation is commonly used, but in general *k* remains an unfixed parameter.

We have evaluated the performance of our models on Kaggle, which measures the prediction error by multi-class logarithmic loss. Each case is labeled with one true category. For each record, we calculated a set of predicted probabilities (one for every class). The formula is then,



where N is the number of incidents in the test set, M is the number of class labels, log is the natural logarithm, yij is 1 if observation i is in class j and 0 otherwise, and pij is the predicted probability that observation i belongs to class j. We have used the inbuilt metrics.log\_loss() function to perform this calculation for our project.

## **Results**

Following are the accuracies and Log Loss values obtained after applying various classifiers on the training dataset:

|  |  |  |  |
| --- | --- | --- | --- |
| **Classifier** | **Random Forest** | **AdaBoost** | **Naïve Bayes** |
| **Accuracy** | 29.29% | 23.82% | 21.58% |
| **Log Loss** | 2.34 | 3.06 | 2.54 |

Below are plots comparing the accuracies and Log Loss values for each for the classifiers:

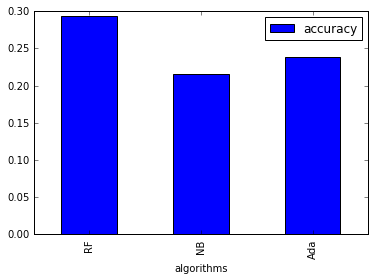


Figure : Accuracy of classifiers

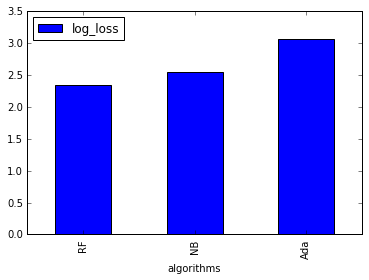


Figure : Log Loss values of classifiers

### **Predictions of Categories on the Test Data:**

Following are the results of the various classifiers used:

#### **Random Forest:**

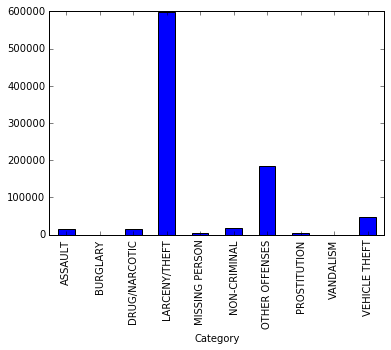


Figure : Plot of RF classifier prediction

#### **AdaBoost**

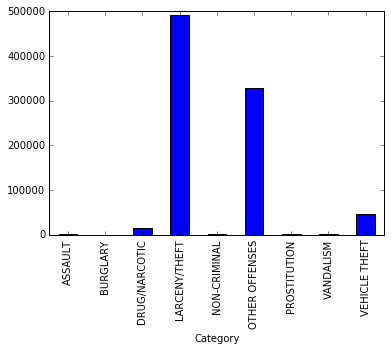


Figure : Plot of AdaBoost classifier prediction

#### **Naïve Bayes**

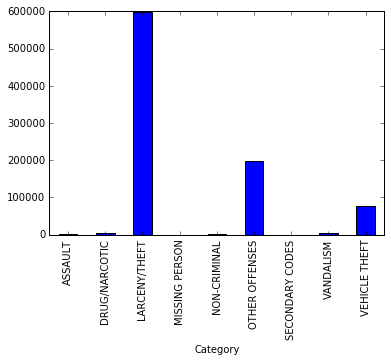


Figure : Plot of Naïve Bayes classifier prediction

As we can see from the predictions, most of the crime classifications can be directly correlated to the trend we observed in the training data, i.e. we are getting high number of predictions for categories like “LARCENY/THEFT”, “OTHER OFFENCES”, and “ASSAULT” which are significantly high in our training data (observed in Figure 5).

## **Discussion**

The hypothesis that was proposed consisted of a dataset where the address column was eliminated since the location coordinates along with the police department districts served the same purpose. This helped us reduce the redundant data from the training set, thereby reducing the run time of the classifiers.

We have reduced the categories from 39 to 22 by filtering out the least occurring categories from the training set. These categories hold no significant importance when compared to those with most occurrences. We observed a rise in accuracy by 1.5% after translating the 39-way prediction problem to a 22-way prediction problem.

We also clustered the location coordinates to 1000 clusters which significantly reduced the dense distribution of the data and helped us wrap around close by locations. It helped in the reduction of levels in the factors for classifiers like Random Forest and also improved the run time and the accuracy further by 1% which is significant in a 22-way prediction problem.

By observing the results obtained, we can conclude that Random Forest has performed better than AdaBoost and Naïve Bayes classifiers. Both the average accuracy and the Log Loss value is found to be better than the other two classifiers used.

Based on the discussion forum on Kaggle, we are at par with the top 250 submissions by comparing the Log Loss values found on the leaderboard for this competition with our best performing classifier. We also observe that the accuracy obtained by cross validation is close to the test accuracy recorded in paper RoboCop[5].

# **Related Work**

The Kaggle competition has about 1840 teams that are participating. The competition is still on-going. The best leaderboard score till date is log loss score of **2.05079**.

The work presented in [4] has compared the results of multiple classifiers. They have used only 5 top categories from the training dataset which constitutes approximately only 66% of the data. The log loss score for their submission was 2.39383.

The work presented in [5] has used a Random Forest model with added demographic data about the location of a crime’s occurrence to get a 39-way classification accuracy of 31.84%. The accuracy is calculated based on the data hold out method from the training dataset.

# **Future Work**

We have noticed from the Kaggle submissions doing a dummy encoding for few of the features (DayOfWeek, PdDistrict) and the class attribute helps to slightly decrease the log loss value. The dummy encoding would essentially convert the problem to a binary classification. The dummy encoding requires us to build a matrix representation of the data which is very data intensive and hence we could not do in the current project.

We are also considering to attempt to build a model using higher order features to reduce the bias of our model. Because of the large size of dataset and consequent long training times with the initial set of features, we could not accomplish it in the current version.

# **Conclusion**

In the project we have accomplished to do a **22-way classification** and have the best classifier accuracy of **29.29%.** The best log loss value is around **2.34** and it can be compared to the top 100 submissions in Kaggle among the 1840 teams. The worst log loss value in Kaggle submissions for this competition is **34.5**. Also, we could see that the accuracy we have computed using cross validation is on par with the accuracy observed by [4] and [5]. We can notice that the prediction accuracy is good when compared to a random 22-way classification which would have a correct classification probability of only about 4.5%.

# **References**

[1] <http://scikit-learn.org/stable/modules/ensemble.html>

[2] <http://scikit-learn.org/stable/modules/ensemble.html>

[3] <http://scikit-learn.org/stable/modules/naive_bayes.html>

[4] San Francisco Crime Classification, Junbo Ke, Xinyue Li, Jiajia Chen

<http://cseweb.ucsd.edu/~jmcauley/cse255/reports/fa15/012.pdf>

[5] RoboCop: Crime Classification and Prediction in San Francisco, John Cherian and Mitchell Dawson

<http://cs229.stanford.edu/proj2015/254_report.pdf>