

A Machine Learning Approach to Predict Crime Using Time and Location Data

Thesis submitted in partial fulfilment of the requirement for the degree of

Bachelor of Science In Computer Science

Under the Supervision of

Dr. Mahbub Alam Majumdar

By

Nishat Shama (15141009)



School of Engineering & Computer Science

Department of Computer Science & Engineering

BRAC University

Declaration

This is to certify that the research work titled “A Machine Learning Approach to Predict Crime Using Time and Location Data” is submitted by Nishat Shama to the Department of Computer Science & Engineering, BRAC University in partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Science. I hereby declare that this thesis is based on results obtained from my own work. The materials of work found by other researchers and sources are properly acknowledged and mentioned references. This thesis, neither in whole nor in part, has been previously submitted to any other University or Institute for the award of any degree or diploma. I carried my research under the supervision of Dr. Mahbub Alam Majumdar.

Signature of Supervisor:

Dr. Mahbub Alam Majumdar

Department of CSE

BRAC University

Signature of authors:

Nishat Shama

ID - 15141009

BRAC University

FINAL READING APPROVAL

Thesis Title: A Machine Learning Approach to Predict Crime Using Time and Location Data

Date of submission: April 18, 2017

This final report on my research is read and approved by the supervisor Dr. Mahbub Majumdar. Its format, citation and bibliographic style are consistent and acceptable. Its illustrative material, including figures, tables and charts are in place. The final manuscript is satisfactory and is ready for submission to the Department of Computer Science & Engineering, School of Engineering & Computer Science, BRAC University.

Signature of Supervisor:

Dr. Mahbub Alam Majumdar
Supervisor
Department of CSE
BRAC University

Acknowledgements

I would like to offer my sincere gratitude to my thesis supervisor Dr. Mahbub Alam Majumdar for guiding me on how I should approach the problem. His input and guidance and guide has been of tremendous value throughout my research work.

I am grateful to Udacity for their free course materials on machine learning. I would also like to acknowledge numerous people who shared their ideas and work at Kaggle discussion forum.

ABSTRACT

Recognizing the patterns of criminal activity of a place is paramount in order to prevent it. Law enforcement agencies can work effectively and respond faster if they have better knowledge about crime patterns in different geographical points of a city. The aim of this paper is to use machine learning techniques to classify a criminal incident by type, depending on its occurrence at a given time and location. The experimentation is conducted on a dataset containing San Francisco's crime records from 2003 - 2015. For this supervised classification problem, Decision Tree, Gaussian Naive Bayes, k-NN, Logistic Regression, Adaboost, Random Forest classification models were used. As crime categories in the dataset are imbalanced, oversampling methods, such as SMOTE and undersampling methods such as Edited NN, Neighborhood Cleaning Rule were used. Solving the imbalanced class problem, the machine learning agent was able to categorize crimes with approximately 81% accuracy.

Contents

LIST OF FIGURES	8
LIST OF TABLES	9
 CHAPTER 1	
1.1 Introduction	10
1.2 Motivation	11
1.3 Research Goal	11
 CHAPTER 2	
2.1 Literature Review	13
 CHAPTER 3	
3.1 Machine Learning	15
3.2 Supervised Learning	15
3.3 Models	16
3.3.1 Decision Tree Classifier	16
3.3.2 Gaussian Naive Bayes	17
3.3.3 Logistic Regression	17
3.3.4 K-Nearest Neighbor	18
3.3.5 Ensemble Method	18
3.3.5.1 Random Forest	18
3.3.5.2 Adaboost	19
3.4 Performance Matrices	19
3.4.1 Accuracy	19
3.4.2 Log-Loss	20
3.4.3 Confusion matrix	20
 CHAPTER 4	
4.1 Crime Dataset and Attributes	21
4.2 Classification of crimes	22
4.3 Features	23
4.3.1 Time of crime incident	23
4.3.2 Location of crime incident	25

CHAPTER 5

5.1	Preprocessing.....	27
5.2	Training and Testing Dataset.....	27
5.3	Extracting New Features.....	28
5.3.1	New Feature from hour.....	28
5.3.2	Principal Component Analysis.....	29
5.4	Feature Selection	29

CHAPTER 6

6.1	Decision Tree.....	30
6.2	Naive Bayes.....	31
6.3	k-Nearest Neighbor.....	32
6.4	Logistic Regression Classifier	33
6.5	Ensemble Learning.....	33
6.6.1	Adaboost.....	34
6.6.2	Random Forest.....	34

CHAPTER 7

7.1	Imbalanced Dataset	36
7.2	Reducing Classes	36
7.2.1	Result	37
7.2.2	Confusion Matrix	37
7.3	Oversampling And Undersampling	40
7.3.1	Oversampling Dataset	40
7.3.1.1	SMOTE.....	41
7.4	Undersampling Dataset	43
7.4.1	Edited Nearest Neighbors	43
7.4.2	Neighborhood Cleaning Rule	44

CHAPTER 8

8.1	Comparative Analysis of Different Strategies	47
8.1.1	Classification on Original Classes	47
8.1.2	Classification on Balanced Classes.....	47

CHAPTER 9

9.1	Conclusion.....	49
9.2	Future Work.....	49

REFERENCES	50
-------------------------	----

LIST OF FIGURES

Fig-1: Frequency of crime categories	23
Fig-2: Crimes occurring in different months	24
Fig-3: Crimes occurring in different days of the week.....	24
Fig-4: Criminal activities occurring in different hour of day	25
Fig-5: Crimes occurring in different police district.....	26
Fig-6: Criminal activities occurring in different parts of day	28
Fig-7: Class frequencies after SMOTE oversampling.....	41
Fig-8: Class Frequency after ENN undersampling.....	43
Fig-9: Class Frequency after NCL undersampling.....	45

LIST OF TABLES

Table -1: Attributes of the crime dataset	21
Table-2: Frequency of top 15 crimes	22
Table-3: Decision Tree Classification result for different parameters	30
Table-4: Decision Tree Classification result for different features	30
Table-5: Gaussian Naive Bayes Classification result for different features	31
Table-6: KNN result	31
Table-7: Logistic Regression result	32
Table-8: Adaboost result	33
Table-9: Random Forest result	35
Table-10: Classification result for reduced classes	37
Table-11: Confusion matrix for Adaboost	38
Table-12: Confusion matrix for Random Forest	38
Table-13: Confusion matrix for GaussianNB	39
Table-14: Confusion matrix for K-Nearest Neighbors	39
Table-15: Confusion matrix for Decision Tree	39
Table-16: Confusion matrix for Logistic Regression	40
Table-17: Classification result after oversampling with SMOTE	42
Table-18: Classification result after using SMOTE with repeated number of calls	42
Table-19: Classification result after undersampling with ENN	44
Table-20: Classification result after undersampling with NCL	45
Table-21: Combined result of different classifiers on imbalanced dataset	47
Table-22: Result of different classifiers on balanced dataset	48

CHAPTER 1

1.1 Introduction

Criminal activities are present in every region of the world affecting quality of life and socio-economical development. As such, it is a major concern of many governments who are using different advanced technology to tackle such issues. Crime Analysis, a sub branch of criminology, studies the behavioral pattern of criminal activities and tries to identify the indicators of such events.

Machine learning agents work with data and employ different techniques to find patterns in data making it very useful for predictive analysis. Law enforcement agencies use different patrolling strategies based on the information they get to keep an area secure. A machine learning agent can learn and analyze the pattern of occurrence of a crime based on the reports of previous criminal activities and can find hotspots based on time, type or any other factor. This technique is known as classification and it allows to predict nominal class labels. Classification has been used on many different domains such as financial market, business intelligence, healthcare, weather forecasting etc.

In this research, a dataset from San-Francisco Open Data[8] is used which contains the reported criminal activities in the neighborhoods of the city San Francisco for a duration of 12 years. I used different classification techniques like Decision Tree, Naive Bayesian, Logistic Regression, k-Nearest Neighbor, Ensemble Methods to find hotspots of criminal activities based on the time of day. Results of different algorithms have been compared and most the effective approach has also been documented.

1.2 Motivation

Criminal law and sociology scholars have also been studying the underlying pattern of crime and its relation to a region or an area's social or economic development, the characteristics of different groups of people living there, family structure, level of education among other things. Different studies and researches have shown that significant concentration of crime happens at micro level of a region. This clustering is often called - hotspot. Research shows that a good neighborhood often has few streets or locations which have a higher concentration of criminal activities compared to others. It is also true in the opposite case, as a bad neighborhood often has many places which have relatively lower rates of crime.

David Weisburd, a renowned criminologist proposed to change people-centric paradigm of strategy to place-centric paradigm. He identified that geographical topology and microstructures are more important in dealing with hotspots.[15] Machine learning can harmonize the same concept by taking a data driven approach to identify hotspots. The agent can also incorporate social and economic standards at a micro level to find the indications.

The intention of this research has been to explore different machine learning techniques to find clusters and analyze the reported criminal activities to find underlying patterns.

1.3 Research Goal

Criminal activities take place all over the world and law enforcement agencies have to deal with them effectively and efficiently. If enforcement agencies have a prior assumption of the class of the crime, it would give them tactical advantages and help resolve cases faster.

Also, an overall study of criminal activity in a geographic area helps to understand the underlying pattern of the crime the area suffers from.

With our research we hope to find answers to the following questions:

- Does a simple criminal database that contains the geographical location & basic details of the criminal activity have enough indicators to predict a type of crime?
- Given just a geographic location and time, how accurately can we classify the crime?
- Explore different techniques to improve the results.

CHAPTER 2

2.1 Literature Review

As combating criminal activity has always been a priority for governments around the world, many researches has been done to effectively find countermeasures and indicators of crime prior to happening. Criminologists have been pursuing to identify hotspots that need major attention from law enforcement agencies.

Researches have been done to study the relation between criminal activities and socio-economic variables like unemployment [6], income level [10], race [3], level of education [5].

Analyzing the usage of mobile network infrastructure and demographic information of people living in different areas of London, a group of researchers were able to predict if particular areas of London would become a criminal hotspot[2]. They have implied that anonymized data collected by mobile networks contain indicators for predicting crime levels.

Combining two datasets - 1990 US LEMAS and crime data 1995 FBI UCR and applying classification techniques like Decision Tree and Naive Bayesian algorithm, 83.95% accuracy have been achieved when asked to predict a crime category for different states of USA. [7]. However, this paper does not disclose if there were any imbalanced classes of crime category. The same databases were also explored by Somayeh et al [13] who employed a number of machine learning algorithms, where k-Nearest Neighbor algorithm performed better than other algorithms by having an accuracy of 89.50%. They also used the Chi-square feature to improve the feature selection.

Wang et al[14] proposed the Series Finder, a machine learning agent that tried to find patterns in crime committed by same offender or groups of offenders. Clustering has also been used to study patterns of criminal behavior and geographic criminal history. [6]

Remond and Baveja [11] have worked on the data noise problem and studied how some police reports or cases are idiosyncratic and do not contain good indicative matrices. Their proposed system called Case-Based Reasoning (CBR) filtered out these cases, and using this system, they were able to predict better compared to not having any filters on the data.

Social networks have also been used as possible source of criminal activity indicators. Sadhana and Sangareddy [12] have used twitter data and sentiment analysis to predict crime in real time. They also used this data to map the concentration of crime occurrences and find large scale hotspots.

CHAPTER 3

3.1 Machine Learning

Machine Learning is a type of artificial intelligence that recognizes patterns using data analysis. A computer can learn and make predictions from data through machine learning without being explicitly programmed. Machine learning can be divided into three main categories: Supervised Learning, Unsupervised Learning and Reinforcement Learning. In this paper, supervised learning methods are used to predict crime categories.

3.2 Supervised Learning

Supervised learning is a machine learning model that can predict an output from a set of inputs. In supervised learning, the output labels are specifically defined. Input object contains various number of features and usually is represented in a vector form. In the training dataset, each input object is paired with a specific output object. A supervised learning algorithm develops a predictive model using training data and fits new information to the model. Separating training and testing data helps supervised learning models to avoid overfitting. The labels of new information is predicted by the algorithm. Supervised learning models can be implemented on both classification and regression problems. In the crime dataset, the goal is to predict the category of a criminal incident in a given time and place. As the crime categories are discontinuous, this is a supervised classification problem. There are different types of supervised classification models. In this paper Decision Tree, Gaussian

Naive Bayes, Linear Regression K-Nearest Neighbor and two Ensemble Methods - Adaboost and Random Forest are used.

3.3 Models

3.3.1 Decision Tree Classifier

Decision tree classification model forms a tree structure from dataset. Decision tree is built by dividing a dataset into smaller pieces. At each step in the algorithm, a decision tree node is splitted into two or more branches until it reaches leaf nodes. Leaf nodes indicates the class labels or result. At each step, decision tree chooses a feature that best splits the data with the help of two functions: Gini Impurity and Information Gain.

Gini Impurity measures the probability of classifying a random sample incorrectly if the label is picked randomly according to the distribution in a branch.

$$I_G(p) = \sum_{i=1}^k p_i(1 - p_i)$$

Gini Impurity is computed by summing the probability p_i times the probability of mistaking while categorizing an item $(1 - p_i)$.

While building the tree, Information Gain helps to decide which feature to split next at each step. Information Gain can be calculated using entropy, which is a function to calculate expected value at each step. Entropy is defined as :

$$H(T) = I_E = - \sum_{i=1}^k p_i \log_2 p_i \dots \text{(ii)}$$

p_i represents the percentage of each feature being present in the child node after a split. Sum of p_i is always 1. Information Gain can be calculated using the following equation :

$$IG = Entropy(parent) - Weighted Sum of Entropy(children) \dots (iii)$$

At each step, decision tree try to make splits that give the purest child nodes.

3.3.2 Gaussian Naive Bayes

Gaussian Naive Bayes is a supervised classifier that uses naive assumption that there is no dependency between two features. This classifier is implemented by applying Bayesian Theorem. According to the theorem, class y and a dependent feature vector consisting of x_0, x_1, \dots, x_n , has the following relationship:

$$P(y|x_0, x_1, \dots, x_n) = \frac{P(x_0, x_1, \dots, x_n|y)}{P(x_0, x_1, \dots, x_n)} \dots (iv)$$

This probability model, along with a decision rule construct Naive Bayes Classifier. There are different types of Naive Bayes classification algorithms based on data distribution. In this paper, Gaussian Naive Bayes is used, where data is assumed to be distributed according to a Gaussian distribution.

3.3.3 Logistic Regression

Logistic regression uses linear boundaries to classify data into different categories. Logistic regression can work on both binary and multiclass problems. For multiclass dataset, one vs the rest scheme is used. In this method, logistic regression trains separate binary classifiers for each class. Meaning, each class is classified against all other classes, by assuming that all other classes is one category.

3.3.4 K-Nearest Neighbor

Nearest Neighbors method is used in both supervised and unsupervised learning. While testing with new data, KNN looks at k data points in training dataset which are closest to the test data point. k indicates the number of neighbors voting to classify a datapoint. The distance can be measured with various metrics. Euclidean distance is the most common choice.

3.3.5 Ensemble Methods

Ensemble learning is a method of combining multiple learning algorithm together to achieve better performance over a single algorithm. Ensemble methods can be divided into two categories: averaging methods and boosting methods.

Averaging method independently develops different models and averages their predictions. In the combined model the variance gets reduced, and therefore, it gives better result.

Boosting method is built by using separate base models sequentially. The goal of boosting is to reduce biases of the combined model and to build a powerful model from several weak models.

In this paper, two ensemble methods are used: Random Forest, which follows the principle of averaging method and Adaboost which is a boosting model.

3.3.5.1 Random Forest

In this ensemble model several decision trees are built using samples drawn with replacement from the training set. The splitting of each node of a tree is not based on the best

split of all features, rather the best split among a random set of features. The bias of the tree increases due to randomness, but averaging also helps to decrease variance, hence this model often achieves better result.

3.3.5.2 Adaboost

Adaboost or Adaptive Boosting is a boosting algorithm. Adaboost combines several weak learners to produce a stronger model. The final output is obtained from the weighted sum of the weak models. As it is a sequential process, in each step a weak learner is changed in favor of misclassified data points in previous classifiers.

3.4 Performance Metrics

It is important to choose proper metrics to evaluate how well an algorithm is performing. In this paper three metrics are used to measure and compare performance of different models.

3.4.1 Accuracy

Accuracy measures how many predictions are matched exactly with the actual or true label of the testing dataset and returns the percentage of correct results. Accuracy can be calculated using the following equation:

$$accuracy(y, \hat{y}) = \frac{1}{n_{samples}} \sum_{i=0}^{n_{samples}-1} 1(\hat{y}_i = y_i) \dots (v)$$

The fraction of correct prediction over n_{sample} is defined by equation (v), where \hat{y}_i is the predicted value of i th sample and y_i is the corresponding true value.

3.4.2 Log-Loss

Log loss is used to measure performance of classifiers by penalizing false classifications. Therefore, smaller value of log loss means the classifier is more accurate. The log loss value of a perfect classifier is 0. The classifier assigns a probability of predicting each class rather than picking the most probable class. With this probabilities, log loss is measured by the following equation:

$$L_{log}(Y, P) = -\frac{1}{N} \sum_{i=1}^{N-1} \sum_{j=1}^{M-1} y_{ij} \log p_{ij} \dots (vi)$$

Y is an $N \times M$ matrix where M is the number of classes and N is the number of test samples. Y matrix contains true labels in a binary indicator format. P is another $N \times M$ matrix that contains probabilities for each class in M dimension. y_{ij} and p_{ij} are true label and probability for j th class in i th sample.

3.4.3 Confusion Matrix

Confusion matrix is different than the metrics that have been discussed so far. Confusion matrix returns a table layout that helps to visualize the performance of an algorithm rather than producing a numerical value that indicates the goodness of the algorithm.

Confusion matrix is an $X \times Y$ matrix where X dimension indicates the true classes and Y dimension indicates the predicted classes. The value of X and Y is equal, and they indicate the number of classes in a dataset.

CHAPTER 4

4.1 Crime dataset and attributes

The experiment is conducted on a specific dataset. The dataset is provided by SF Opendata from SFPD Crime Incident Reporting System [8]. It provides information on crime incidents that occurred in San Francisco for the period of 1/1/2003 to 5/13/2015. The dataset is a csv file containing 878049 rows. The attributes are given below.

Datetime	A timestamp when the given crime occurred.
Category	Type of crimes. This is the target label for the data. There are 39 types of crime listed in the data
Crime Description	A detailed description of a specific type of crime
Day	Day of week
pDistrict	Name of police department district. There are total 10 Police Districts in the data
Resolution	How the crime was solved. 17 types of resolution
Address	The approximate street address for the incident
X	It signifies the latitude of the location of the crime.
Y	It signifies the longitude of the location of the crime.

Table -1: Attributes of the crime dataset

From the list of attributes in Table-1, the features and label can be determined. The target label that needs to be predicted is the *Category* of a crime incident. The attributes: *Crime Description* and *Resolution* are also related to the target label. Hence, all other attributes apart from these three attributes are used as features.

4.2 Classification of crimes

There are 39 types of crime in the San Francisco Crime Dataset. These types are considered as classes and having 39 classes makes it a multi class problem.

Category	Frequency
LARCENY/THEFT	174900
OTHER OFFENSES	126182
NON-CRIMINAL	92304
ASSAULT	76876
DRUG/NARCOTIC	53971
VEHICLE THEFT	53781
VANDALISM	44725
WARRANTS	42214
BURGLARY	36755
SUSPICIOUS OCC	31414
MISSING PERSON	25989
ROBBERY	23000
FRAUD	16679
FORGERY/COUNTERFEITING	10609
SECONDARY CODES	9985

Table-2: Frequency of top 15 crimes

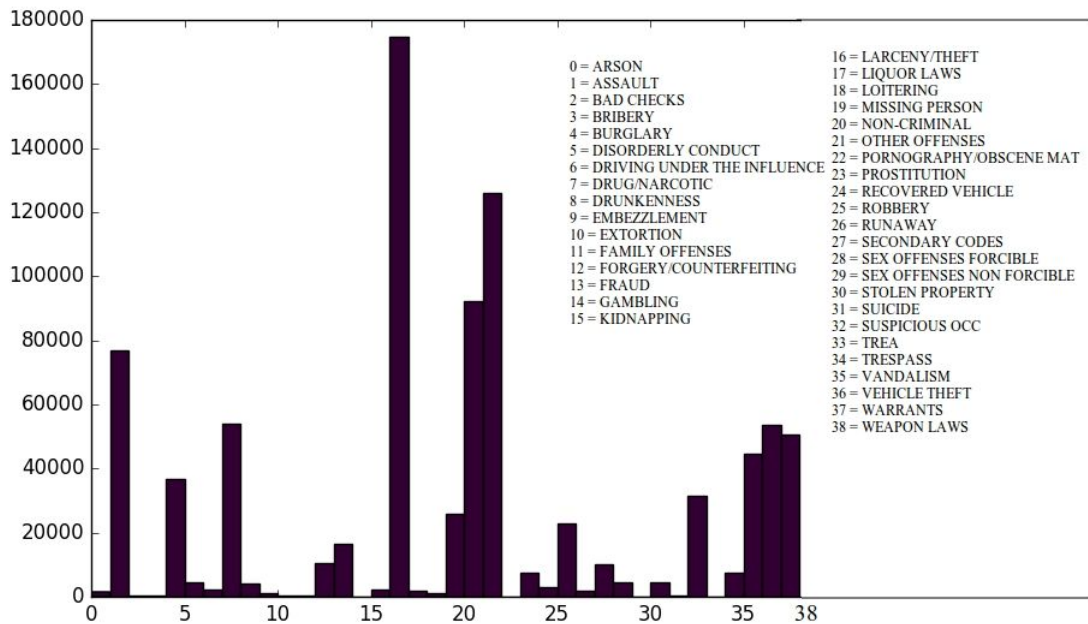


Fig-1: Frequency of crime categories

There are few crimes that occur very frequently, and some crimes are really rare. Larceny/Theft is the most common crime with a frequency of 174900 and Trespassing (TREA) is the least common crime with a frequency of 6. Where there are 14 crime classes that occurred more than 10,000 times, 14 crime classes occurred less than 2,000 times. This shows that the classes are not distributed equally.

4.3 Features

4.3.1 Time of crime incident

From datetime stamp, four four main features are extracted. - year, month, date, hour.

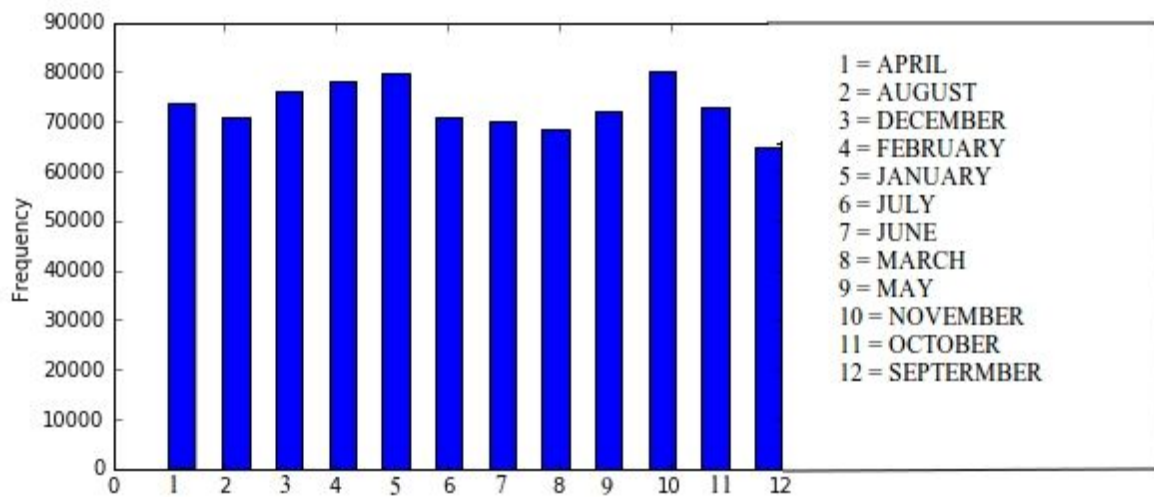


Fig-2: Crimes occurring in different months

Looking at the plots of reported criminal activities occurring throughout of the year, it is apparent that Summer (June to August) and Winter (December to February) has less criminal activities compared to other seasons.

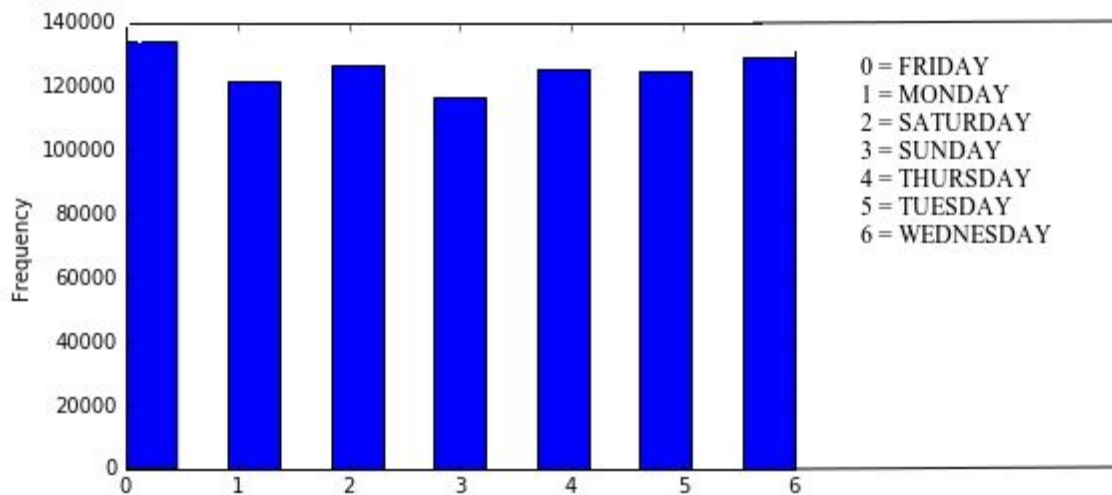


Fig-3: Crimes occurring in different days of the week

Most crime occur on Friday, least crimes occur on Sunday. Crime rate almost gradually increases from Monday to Thursday. This however does not seem indicative of any pattern.

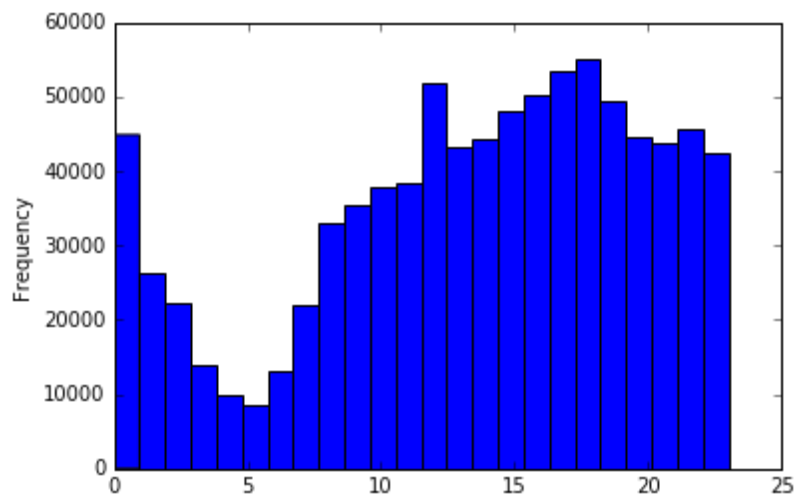


Fig-4: Criminal activities occurring in different hour of day (in 24 hours format)

Taking a look at when crimes occur gives a rather interesting picture. Most crimes occur during afternoon-evening. From midnight to morning, reports of criminal activities are low. There is an upsurge of criminal activities at 6 PM and 8 PM. Criminal activities are drastically reported around 9 AM and it continues to show a gradual increase throughout the day peaking at 6 PM, after which it starts to decrease.

4.3.2 Location of crime incident

‘Latitude’ (X), ‘Longitude’ (Y) have more than 30,000 unique entries of the total 878049 entries, while ‘Address’ has total 23228 unique Entries.

One problem with the location features is, there are 26,533 entries for a specific address. Which is the location of San Francisco Police Officer's Association. This default address gives data a low variance problem.

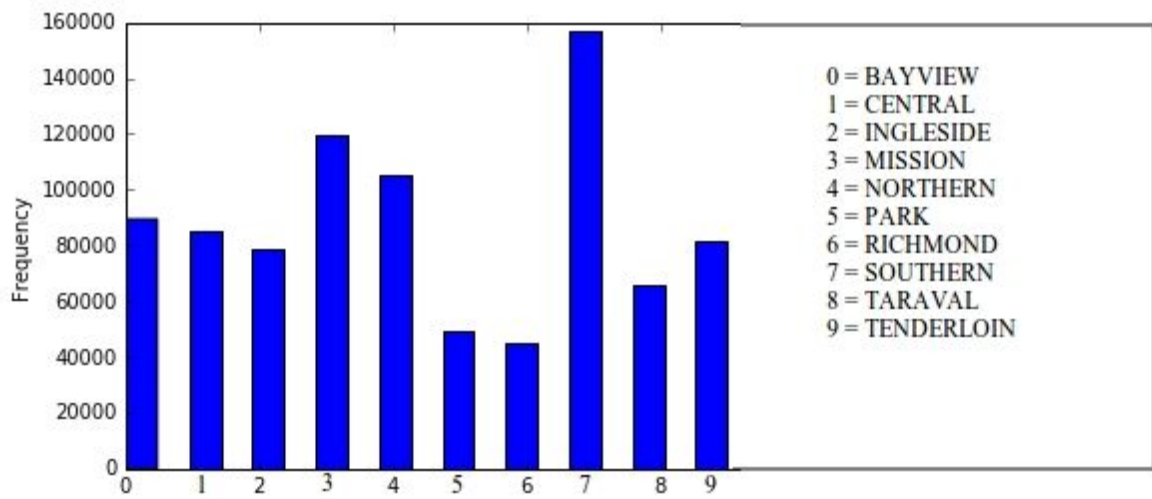


Fig-5: Crimes occurring in different police district

Among the ten police districts, criminal activities in the southern district is higher than any other district.

CHAPTER 5

5.1 Preprocessing

Python library Scikit-learn (sklearn) is used for preprocessing the dataset. Some attributes in the csv files contains string values and others are numeric values. In order to use this dataset in machine learning models, the text features need to be converted into a numeric value.. Python library `numpy` is used to contain both features and label of the dataset after converting them into numeric values.

Attributes with string data type are “Day”, “Category”, “Address” columns. Scikit-learn has a preprocessing package that converts string data into numeric data. This package gives an integer value to each unique item after sorting items in ascending alphabetical order.

Datetime attribute is also a string data type, however this is converted into a datetime object and four different attributes are obtained from it: “Hour”, “Date”, “Month” and “Year”.

5.2 Training and Testing Dataset

To avoid overfitting and getting more realistic accuracy, the dataset is divided into two portion: testing dataset and training dataset. Training dataset contains all features along with the target label. Testing dataset only contains the features from which a machine learning model predicts the target label. Scikit-learn’s `model_selection` package contains a class `test_train_split` that splits the original dataset into testing and training dataset. The default value of the test dataset size is 25% of the original dataset. This default value is used in the conducted experiments.

5.3 Extracting New Features

While the given features give sufficient information about a crime incident, new features can be extracted from the given features which might prove to be useful.

5.3.1 New Feature from hour

There is a specific pattern in occurrence of crime during different part of a day, as shown in previous analysis of different features. A new feature can be extracted by dividing a day in few different parts rather than considering 24 hours. Fig-4 Shows that crime starts falling after midnight and rises gradually from afternoon to evening. Therefore, a day can be divided into following parts:

- Early morning: 1AM-7AM
- Late morning: 8AM-1PM
- Afternoon: 2PM-7PM
- Night 8PM-12AM

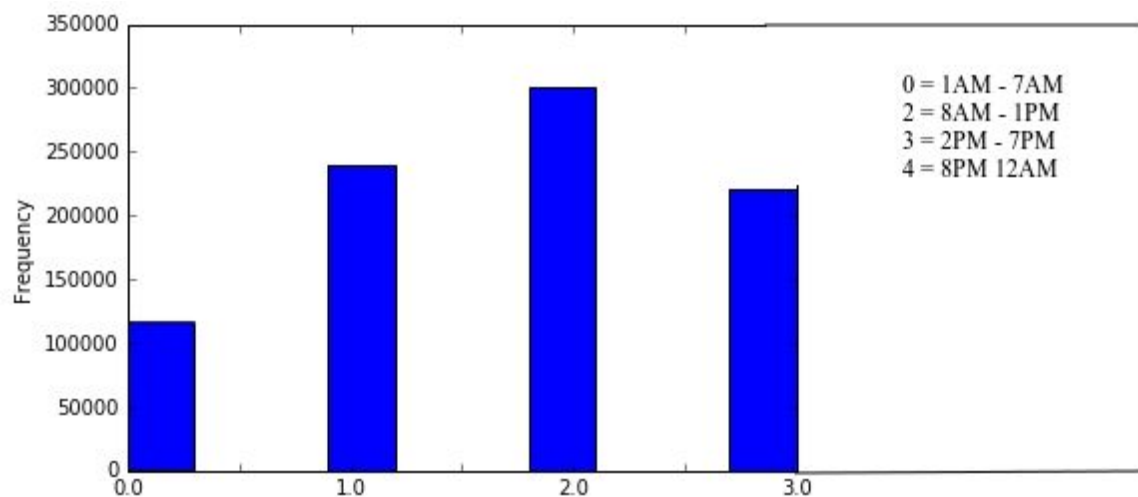


Fig-6: Criminal activities occurring in different parts of day

With blocks of time introduction, a clearer picture of time can be found and it performs better compared to 24 cycle of time. With blocks of time a linear data is achieved compared to an arbitrary sequence of occurrence of criminal activities that does not reflect any indication.

5.3.2 Principal Component Analysis

Principal Component Analysis (PCA) is a method of linear dimensionality reduction. It projects data in a lower dimensional space and maximizes variance. `sklearn.decomposition` package's PCA class is used to obtaining variable features from the existing 9 features. `N_components` in the PCA class indicates the number of new features with lower dimension to be extracted from the old features.

5.4 Feature Selection

Total number of features after preprocessing is 9. One assumption is made before using any predictive model is, some features might be more useful than others. While more feature describes the data better, too many features can make classification complicated and cause overfitting. `sklearn.feature_selection` module uses univariate statistical tests to find features that are best related to the target label. `select_percentile` class of this module takes a percentage input and returns that percentage of best features. For classification problems `f_classif` function of this class is used. Different percentage of features are use in different models to see using how many features gives better performance.

CHAPTER 6

Supervised classification models are applied on San Francisco Crime Dataset to predict the category of a crime incident. In the following part, performance of different classification models are discussed.

6.1 Decision Tree

`Sklearn.tree` module provides `DecisionTreeClassifier` class. Among many parameters of this class, two parameters are useful in this case: `min_samples_split` indicates the number of splits to make at each step of building a decision tree and `criterion` indicates the function to measure the quality of split. As discussed above, there are two types of function to measure quality of split: information gain and impurity. In this class information gain is indicated with *entropy* and impurity is indicated with *gini*.

Split	Function	Accuracy	Log-loss
50	entropy	27.1%	7.99
300	entropy	28.13%	3.33
500	entropy	27.79%	2.92
600	entropy	27.81%	2.83
50	gini	26.49%	8.12
300	gini	28.17%	3.32
500	gini	27.96%	2.92
600	gini	27.89%	2.83

Table-3: Decision Tree Classification result for different parameters

Log loss performance gets better with more sample split for both kernels - gini and entropy. The best accuracy we get using *entropy* is 28.13, while maintaining a log loss score of 3.33. *Gini* performs better by improving log loss performance without sacrificing accuracy using higher splits.

Feature Selection	Accuracy	Log-loss
All features	28.15%	3.33
Select Percentile 70%	28.32%	3.31
Select Percentile 90%	28.14%	3.33
n_components=5 (PCA)	26.23%	3.66
n_components=3 (PCA)	24.77%	3.63

Table-4: Decision Tree Classification result for different features

N_components indicates the number features extracted from principal component analysis. The result shows, selecting 70% of the features give better result than using all features or using reduced dimensionality features.

6.2 Gaussian Naive Bayes

`sklearn.naive_bayes` provides `GaussianNB` class.

Features	Accuracy	Log-Loss
All Features	22.34%	2.58
Select Percentile 70%	22.27%	2.59
Select Percentile 90%	22.30%	2.56
n_components=5	21.85%	2.61

n_components=3	21.41%	2.62
----------------	--------	------

Table-5: Gaussian Naive Bayes Classification result for different features

Although this classifier gives poor accuracy, Log loss measurement is relatively better in this model. Principal component analysis gives poor result in the case of Naive Bayes too.

6.3 k-Nearest Neighbor

`KNearestNeighbors` class in `sklearn.neighbors` module provides supervised nearest neighbors classification models using `k` nearest neighbors. Among different parameters of the class, `n_neighbors` indicates the value of `k`, `metric` indicates the metric used to measure the distance of neighbors. An equivalent of Euclidean distance is used as default.

Features	n_neighbor	Accuracy	Log Loss
All	30	27.85%	6.43
All	60	28.16%	4.505
All	100	28.10%	3.68
All	300	27.57%	3.057
Select 70%	60	28.44%	4.47
Select 70%	100	28.35%	3.67
Select 70%	300	27.58%	2.83
Select 90%	100	28.17%	3.69
n_components=5	100	28.11%	3.69
n_components=3	100	28.12%	3.68

Table-6: KNN result

Log Loss improves as neighbor numbers are increased. However, accuracy also decreases with it. Here, selecting 70% of the features give the best accuracy and log loss. Using features from principal component analysis and using all features gives similar result.

6.4 Logistic Regression Classifier

`sklearn.linear_model.LogisticRegression` class is used for this model. The parameter `multi_class` is set to `ovr` which provides one vs the rest scheme, as multi class model is needed. `Class_weight` parameter makes classes balanced in case of imbalanced classes.

class_weight	Accuracy	Log Loss
None	21.51%	2.62
Balanced	48.67%	3.51

Table-7: Logistic Regression result

This shows that balancing classes gives a boost to accuracy. As the crime classes have large differences in frequency, making them balanced gives better result.

6.5 Ensemble Learning

`Sklearn.ensemble` provides ensemble methods like Adaboost and Random Forest. As base classifier in ensemble method, decision tree classifier, Gaussian Naive Bayes classifier, logistic regression, k-NN are used. In decision tree, `min_samples_split = 300` and `criterion = gini` gives the best result. For most algorithms, using 70% features give better result. Therefore these specifications are used while training ensemble models.

6.5.1 Adaboost

`Adaboost` class has different parameters including `base_estimator` and `n_estimator`. `Base_estimator` indicates which classifier is to boost through `Adaboost`. `N_estimator` indicates the number of weak classifiers to use for boosting.

Classifier	estimator	Accuracy	Log Loss
Decision Tree	100	21.88%	3.66
Decision Tree	10	22.27%	3.66
GaussianNB	10	12.84%	3.66
Logistic Regression	10	60.57%	3.66

Table-8: Adaboost result

Adaboost with different base classifier gives same log loss value. Adaboost try to reduce misclassification, which is the main observation of log loss metrics. However, accuracy measurement shows that Gaussian naive bayes gives a very low accuracy and logistic regression gives a very high accuracy. This shows that when some classifiers perform better after boosting in accuracy metrics, the predictive abilities of different classifiers remain equal.

6.5.2 Random Forest

Parameters of `RandomForestClassifier` class that are used are: `n_estimator`, `min_samples_split`, `criterion`, where `min_samples_split` and `criterion` are the parameters of decision tree of the random forest. `N_estimator` indicates the number of trees to build.

number_of_trees	Accuracy	Log loss
10	28.80%	2.43
50	29%	2.38

Table-9: Random Forest result

Both accuracy and log loss improve while number of trees in random forest are increased. With more tree, Random Forest has samples for averaging, which can decrease variance.

CHAPTER 7

7.1 Imbalanced Dataset

A dataset set is imbalanced when the classification categories are not represented approximately equally [4]. As seen in Fig-1 the crime classes are imbalanced with highest frequency of 174900 and lowest frequency of 6. Two different strategies are used to deal with imbalanced dataset problem. The first one is a brute force method to reduce number of classes by only picking the classes with highest frequencies. The second strategy is to synthesize more samples or reduce samples to make classes more balanced.

7.2 Reducing Classes

Only top 4 crime categories are used in attempt to reduce the number of classes. All other crime classes are labeled by “other crimes”. From Table-2, top 6 crimes are

- LARCENY/THEFT
- OTHER OFFENSE
- NON CRIMINAL
- ASSAULT
- DRUG/NARCOTIC
- VEHICLE THEFT

Other offense and non-criminal category do not give much information. The rest four categories are used as main classes. All previously used algorithm with the best resulting parameters are used with the changed label.

7.2.1 Result

Algorithm	Accuracy	Log-loss
Decision Tree Classifier	58.84%	2.46
Gaussian Naive Bayes	59.02%	1.15
K-Nearest Neighbor	60.40%	1.14
Logistic Regression	59.02%	1.16
Random Forest Classifier	60.62%	1.04
Adaboost Classifier (Dtree)	60.16%	1.18

Table-10: Classification result for reduced classes

Result of all classifiers increases after using less number of classes. Log loss values improve in large margin after reducing classes.

7.2.2 Confusion Matrix

Confusion matrix is used to visualize the results obtained from different classifier.

The classes are indicated with the following integers:

- OTHERS (0)
- ASSAULT (1)
- DRUG/NARCOTIC (2)
- LARCENY/THEFT (3)
- VEHICLE THEFT (4)

True label indicates the true value of a given class, and predicted label indicates the prediction made of that class. True and predicted results are a match on coordinates such as (0,0), (1,1), (2,2) etc.

	True Label					
		0	1	2	3	4
pre	0	119343	18089	11488	32917	11563
dic	1	72	19	0	18	7
ted	2	1394	134	1613	110	32
lab	3	8083	891	331	10501	1169
el	4	679	98	25	337	600

Table-11: confusion matrix for Adaboost

	True Label					
		0	1	2	3	4
pre	0	124574	18723	12429	39245	12709
dic	1	0	0	0	0	0
ted	2	622	49	863	23	8
lab	3	4360	456	165	7586	607
el	4	15	3	0	29	47

Table-12: confusion matrix for Randomforest

		True Label				
		0	1	2	3	4
predicted label	0	129571	19231	13457	43883	13371
	1	0	0	0	0	0
	2	0	0	0	0	0
	3	0	0	0	0	0
	4	0	0	0	0	0

Table-13: confusion matrix for GaussianNB

		True Label				
		0	1	2	3	4
predicted label	0	121271	18330	11714	33982	12260
	1	13	7	8	5	1
	2	1237	122	1416	63	11
	3	6957	766	314	9790	986
	4	93	6	5	43	113

Table-14: confusion matrix for K-Nearest Neighbors

		True Label				
		0	1	2	3	4
predicted label	0	111699	16903	10592	29308	9424
	1	1282	399	67	300	150
	2	2353	247	2222	203	72
	3	11972	1357	484	13049	1915
	4	2265	325	92	1023	1810

Table-15: confusion matrix for Decision Tree

	True Label					
		0	1	2	3	4
pre	0	129571	19231	13457	43883	13371
dic	1	0	0	0	0	0
ted	2	0	0	0	0	0
lab	3	0	0	0	0	0
el	4	0	0	0	0	0

Table-16: confusion matrix for Logistic Regression

The confusion matrix for different classifiers show that even though accuracy and log loss function give good result after reducing classes, most classes are being predicted to be in class 0 (others) as that is the largest class. It also shows that, although gaussian naive bayes and logistic regression has moderately good accuracy and log loss value, they are predicting 0 for most data points. Therefore predicting the highest frequent class for each sample (known as a dumb model) gives similar accuracy as the models that attempt to predict other classes too. As class 0 is quite larger than the rest selected top crimes, the dataset still remains imbalanced.

7.3 Oversampling And Undersampling

7.3.1 Oversampling Dataset

Oversampling method helps to deal with imbalance classes by copying minority class samples or synthesizing new minority class samples to make the dataset more balanced. There are different oversampling methods. In this paper, Synthetic Minority Oversampling Technique (SMOTE) is used.

7.3.1.1 SMOTE

In SMOTE, synthetic minority classes are generated by operating on feature space using k-nearest neighbor [4]. The value of k depends on the number of new samples that need to be created.

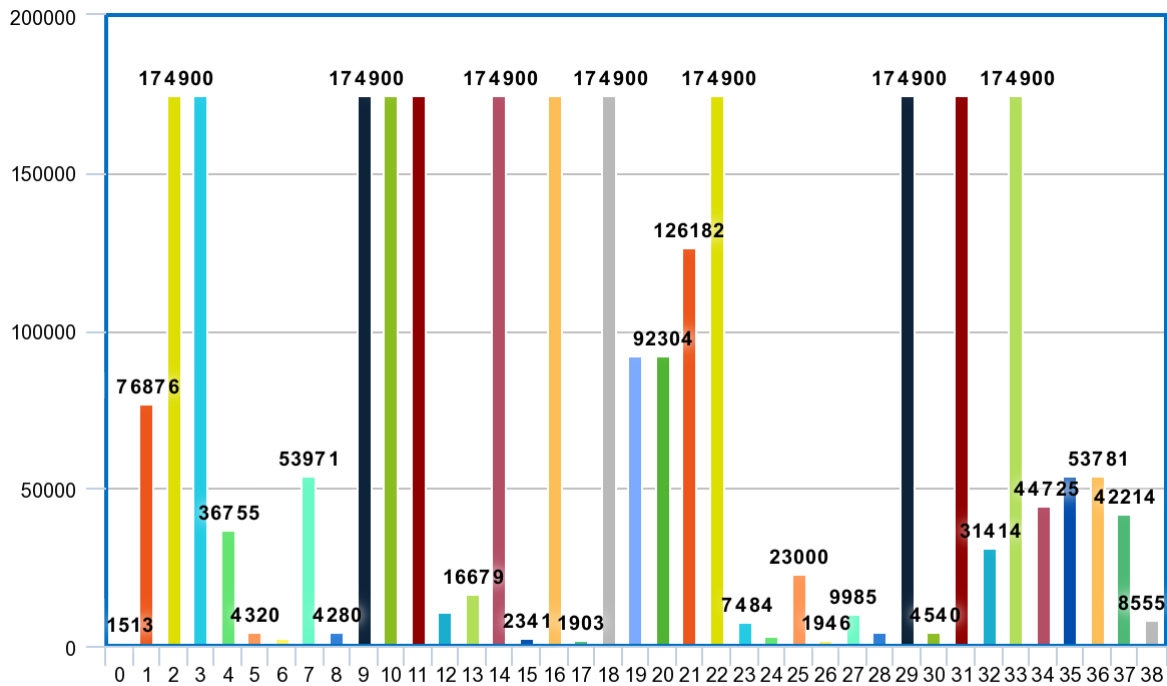


Fig-7: Class frequencies after SMOTE oversampling

Fig-7 shows that differences between most classes are reduced. Although some classes are still small, many classes have become equal using SMOTE. It also shows that total number of data points increased largely after applying SMOTE.

Scikit-learn's contributed package Imbalanced-learn is used for oversampling dataset. The `SMOTE` class of the module is implemented for binary classes. To use it in multiclass problem, the class needs to be called repeatedly to reach a balanced number for all classes. SMOTE is applied on different classifiers with their best resulting parameters.

Classifier	Accuracy	Log loss
Random Forest	40.66%	1.99
Adaboost	36.47%	3.65
Decision Tree	39.84%	2.81
Logistic Regression	27.98%	2.49
K-NN	39.96%	3.12

Table-17: classification result after oversampling with SMOTE

Here, Random forest gives best accuracy and log loss value.

Number of calls	Classifier	Accuracy	Log Loss
1	Decision Tree	39.84%	2.81
5	Decision Tree	63.37%	1.72
10	Decision Tree	66.42%	1.61
15	Decision Tree	63.77%	2.65
10	Adaboost	74.12%	3.64
10	Random Forest	71.43%	1.16
10	k-NN	62.63%	1.85

Table-18: classification result after using SMOTE with repeated number of calls

By repeatedly calling SMOTE function, it eventually makes the dataset balanced by generating enough minority classes. Both ensemble methods: decision tree and adaboost performs well on oversampled data. Results of other classifiers also increase after oversampling. A downside of this is approach is that calling SMOTE multiple times make the dataset very large and it uses more processing power.

7.4 Undersampling dataset

Undersampling is a process where the majority classes in an imbalanced dataset are undersampled so that the classifier can compensate for minority classes. It helps an machine learning agent to not become biased and to not ignore false positives. Scikit-learn's contribution package imbalanced-learn provide various classes for undersampling dataset. Edited Nearest Neighbors and Neighborhood Cleaning Rule methods are used to undersample the crime dataset. One problem with undersampling is, some classes become too small that they do not exist in both testing and training data, which makes log loss calculation difficult.

7.4.1 Edited Nearest Neighbors (ENN)

Edited nearest neighbors is an undersampling method removes instances of majority classes for which the prediction of KNN is different from the majority class [1].

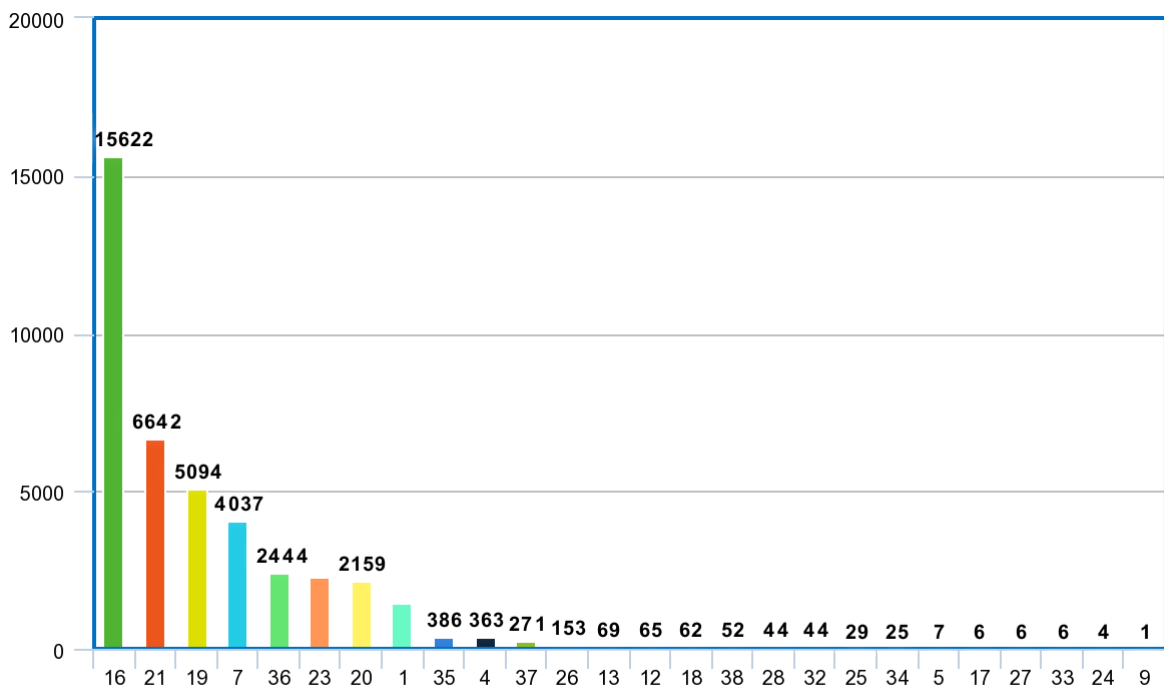


Fig-8: Class Frequency after ENN undersampling

While some categories are removed due to ENN undersampling, The difference between highest and lowest frequency is largely reduced.

`imblearn.EditedNearestNeighbors` class is used to apply undersampling on the dataset. After undersampling, the performance is tested with previously used classification models.

Classifier	Accuracy
Decision Tree (split=300)	71.55%
Decision Tree (split=50)	78.07%
Logistic Regression	49.75%
K nearest Neighbor	70.16%
Random Forest(split=300)	71.13%
Random Forest(split=50)	80.02%
Adaboost(Decision Tree, split=300)	67.53%
Adaboost (Decision Tree, split=50)	81.93%

Table-19: Classification result after undersampling with ENN

ENN undersampling method works better with decision tree and ensemble methods that work with decision tree. As the dataset becomes smaller, small tree split produces better result. Adaboost with decision tree having 50 split gives the best result of 81.93% accuracy.

7.4.2 Neighborhood Cleaning Rule (NCL)

Neighborhood Cleaning Rule is a method of undersampling. This model is an improved version of ENN. In this model, a datapoint is removed, if it belongs to a majority class and there is a prediction error related to its nearest neighbors [1].

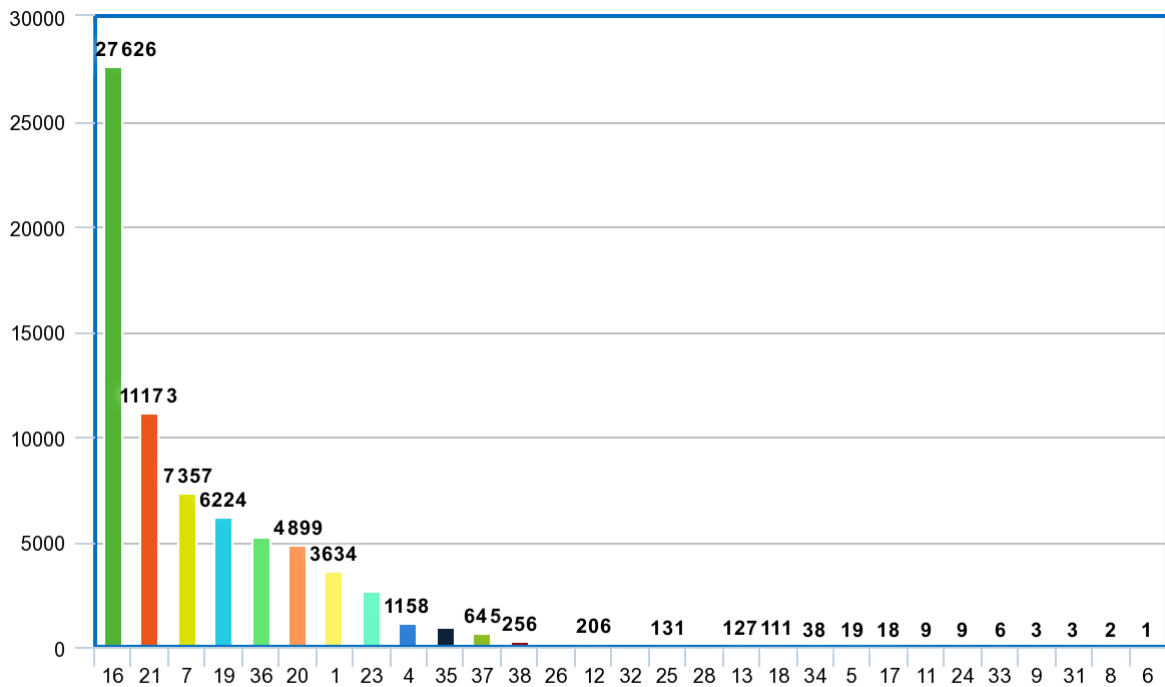


Fig-9: Class frequency after NCL undersampling

Like ENN, NCL also reduces high frequency classes, however the most frequent class 16 (LARCENY/THEFT) is higher in NCL.

`imblearn.NeighborhoodCleaningRule` class is used to undersample dataset and for performance measurement different classification models are used.

Classifier	Accuracy	Log loss
Random Forest(split=300)	65.83%	4.45
Random Forest (split=50)	73.18%	4.33
Decision Tree (split=50)	70.83%	5.22
Decision Tree (split=300)	65.08%	4.71
Adaboost (Dtree split=50)	72.14%	3.31
k-NN	64.50%	4.73
Logistic Regression	46.30%	4.99

Table-20: Classification result after undersampling with NCL

This undersampled dataset also performs better with decision tree having lower split. Log loss value is poor for these samplings. Random Forest and Adaboost gives satisfactory result.

CHAPTER 8

8.1 Comparative analysis of different strategies

Three main approaches are taken to to predict crime category from San Francisco Crime dataset. The original dataset with 39 classes gives both low accuracy and log loss. Log loss and accuracy improves if only 5 classes are considered. However, confusion matrix shows that this model does not predict all classes equally. Systematically undersampling and oversampling dataset gives better result.

8.1.1 Classification on Original Classes

Classifier	Parameters	Accuracy	Log loss
Decision Tree	split=300, criterion= <i>gini</i>	28.32%	3.31
GaussianNB		22.30%	2.56
k-NN	n_neighbor=100	28.10%	3.68
Logistic Regression	class_weight= <i>balanced</i>	48.67%	3.51
Random Forest	number_of_tree=50	29%	2.38
Adaboost	Logistic regression, estimator=10	60.57%	3.66

Table-21: Combined result of different classifiers on imbalanced dataset

The highest accuracy achieved by boosting logistic regression classifier. Logistic regression performs better as it tries to balance classes with `class_weight` parameter. However, random forest gives better log loss value than any other classifier.

8.1.2 Classification on Balanced Classes

The first attempt to make the classes more balance by reducing low frequency classes give good accuracy and log loss value. Random Forest classifier performs best in both metrics with accuracy of 60.62% and log loss value 1.04. However, confusion matrix shows that many classifiers predict the largest class most of the time and achieve good accuracy in spite of that.

Oversampling and undersampling the dataset gives satisfactory result.

Sampling	Method	Classifier	Accuracy	Log loss
Over Sampling	SMOTE	Random Forest	71.43%	1.16
Under Sampling	ENN	Adaboost (Dtree split=50)	81.93%	
Under Sampling	NCL	Adaboost (Dtree split=50)	72.14%	3.31

Table-22: Result of different classifiers on balanced dataset

Classification done on both undersampled and oversampled dataset gives better result over the original dataset. While SMOTE performs better in log loss metrics, Using ENN undersampling, adaboost decision tree with 50 split has been most successful to understand the trend in the data and produced the best accuracy of 81.93%.

CHAPTER 9

9.1 Conclusion

Throughout the research it has been evident that basic details of a criminal activities in an area contains indicators that can be used by machine learning agents to classify a criminal activity given a location and date. Even though the learning agent suffers from imbalanced categories of the dataset, it was able to overcome the difficulty by oversampling and undersampling the dataset.

Through the experiments, it can be seen the imbalanced dataset was benefitted by using ENN undersampling. Using the undersampled data, Adaboost decision tree successfully classified criminal activities based on the time and location. With a accuracy of 81.93%, it was able to outperform other machine learning algorithms.

Imbalanced classes are one of the main hurdles to achieve a better result. Though the machine learning agent was able to predictive model out of simply crime data, a demographic dataset would probably help to further improve the result and solidify it.

9.2 Future Work

For this research, only crime data has been used, but as many researched have showed that a particular area's socio-economic standard is also a key indicator of possible criminal activity. This machine learning agent could incorporate those data and might perform better.

This model can be also used for other geographic locations. This would also help to analyze crimes occurring in different locations and build a better understanding of different crimes and its relation with particular demography.

Also, there are many advanced machine learning approaches that can be explored. Deep Learning & Neural Networks can provide a more balanced understanding of criminal activities. As it has been seen on this research, imbalanced classes has been a major issue in dealing with the particular database. Advanced techniques to deal with imbalanced classes are also something that remains to be explored.

References

1. Beckmann, M., Ebecken, N. F., & de Lima, B. S. P. (2015). A KNN undersampling approach for data balancing. *Journal of Intelligent Learning Systems and Applications*, 7(4), 104.
2. Bogomolov, A., Lepri, B., Staiano, J., Oliver, N., Pianesi, F., & Pentland, A. (2014, November). Once upon a crime: towards crime prediction from demographics and mobile data. In *Proceedings of the 16th international conference on multimodal interaction* (pp. 427-434). ACM.
3. Braithwaite J. Crime, Shame and Reintegration. *Ambridge: Cambridge University Press*, 1989.
4. Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16, 321-357.
5. Ehrlich I.. On the relation between education and crime. 1975.
6. Freeman R. B. The economics of crime. *Handbook of labor economics*, 3:3529–3571, 1999.
7. Iqbal, R., Murad, M. A. A., Mustapha, A., Panahy, P. H. S., & Khanahmadliravi, N. (2013). An experimental study of classification algorithms for crime prediction. *Indian Journal of Science and Technology*, 6(3), 4219-4225.
8. "San Francisco Crime Classification." *San Francisco Crime Classification | Kaggle*. N.p., n.d. Web. 16 Apr. 2017.
9. Maloof, M. A. (2003, August). Learning when data sets are imbalanced and when costs are unequal and unknown. In *ICML-2003 workshop on learning from imbalanced data sets II* (Vol. 2, pp. 2-1).

10. Patterson E. B.. Poverty, income inequality, and community crime rates. *Criminology*, 29(4):755–776, 1991.
11. Redmond M, Baveja A., “A Data-driven Software Tool for Enabling Cooperative Information Sharing Among Police Departments”, *European Journal of Operational Research*, Science Direct, vol. 141, no. 3, pp. 660–678, 2002.
12. Sadhana, C. S. (2015). Survey on Predicting Crime Using Twitter Sentiment and Weather Data israce .2015
13. Shojaee, S., Mustapha, A., Sidi, F., & Jabar, M. A. (2013). A study on classification learning algorithms to predict crime status. *International Journal of Digital Content Technology and its Applications*, 7(9), 361
14. Wang X., Gerber M.S., and BrownD. E. Auto-matic crime prediction using events extracted from twitter posts. *In Social Computing, Behavioral-Cultural Modeling and Prediction*, pages 231–238. Springer, 2012.
15. Weisburd D.. Place-based policing. *Ideas in American Policing*, 9:1–16, 2008.