**Home Credit Default Risk**

TeamName - Elite

**Swapnil Jain Vinayak Chaturvedi Rushikesh Jachak**

MT2020171 MT2020046 MT2020126

[swapnil.jain@iiitb.org](mailto:shreyas.gupta@iiitb.org) [vinayak.chaturvedi@iiitb.org](mailto:shreyas.gupta@iiitb.org) [rushikesh.jachak@iiitb.org](mailto:shreyas.gupta@iiitb.org)

***Abstract*—This is a detailed report on our work on classifica- tion to ensure that clients capable of repayment are not rejected and that loans are given with a principal, maturity, and repayment calendar that will empower their clients to be successful.**1

***Index Terms*—Feature Engineering, Geohashing, logarithmic odds, XGBoost, One vs Rest classifiers, Grid Search, Cross Validation, Logistic Regression, EasyEnsembleCLassifier, Naive Bayes, Random Forest Classifier, SGD Classifier**

PROBLEM STATEMENT

Many people struggle to get loans due to insufficient or non-existent credit histories. And, unfortunately, this population is often taken advantage of by untrustworthy lenders.

Home Credit strives to broaden financial inclusion for the unbanked population by providing a positive and safe borrowing experience. In order to make sure this underserved population has a positive loan experience, Home Credit makes use of a variety of alternative data--including telco and transactional information--to predict their clients' repayment abilities.

DATASET

This dataset contains 7 tables viz.

* application{train/test}.csv :This is the main table, broken into two files for Train (with TARGET) and Test (without TARGET).
* bureau.csv: All client's previous credits provided by other financial institutions was in this table.
* bureau\_balance.csv: Monthly balances of previous credits in Credit Bureau.
* POS\_CASH\_balance.csv: monthly balance snapshots of previous POS (point of sales) and cash loans that the applicant had with Home Credit.
* credit\_card\_balance.csv: Monthly balance snapshots of previous credit cards that the applicant has.
* previous\_application.csv: All previous applications for Home Credit loans of clients who have loans in our sample.
* installments\_payments.csv: Repayment history for the previously disbursed credits in Home Credit related to the loans in our sample.

1The problem statement is a contest hosted on Kaggle.

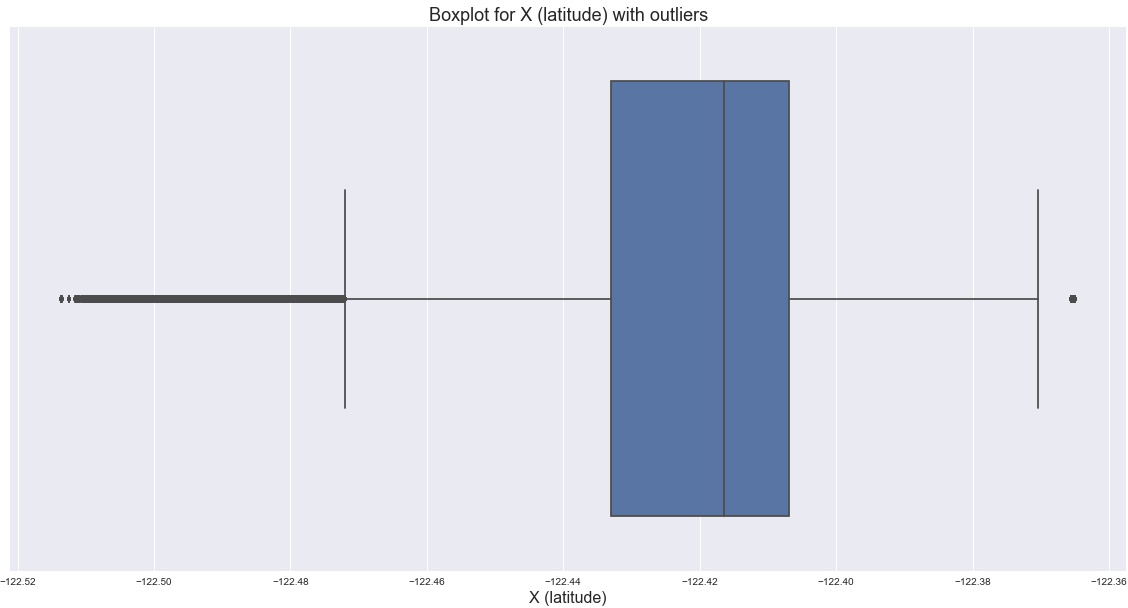
* 1. INTRODUCTION

With advancements in corrections, surveillance systems, weapons and armory, strict laws, frequency of crime and other illegal activities have gradually decreased in the past couple of decades. Our streets and societies are safer, but this forces us to ponder over our methodology. Punishing outlaws and criminals post-crime increases the tensions surrounding that crime and above all increases fear among people but it still doesn’t prevent the crime from happening.

The concept of somehow predicting the crime before it happens is surreal but has been explored for decades. One of the first references to this idea was in the book *Mindhunter: Inside the FBI’s Elite Serial Crime Unit* by Joe Penhall. The novel follows the life of two FBI agents and a Psychologist who interview dozens of serial killers and mass murderers to create a psychiatric profile for each, with hopes to use it to understand them, generalize it and eventually use it to prevent murders from happening.

The final model we’ll generate won’t directly help predict crimes, but give additional information to police personnel about the crime scene they are visiting. For example, 911 center sends a broadcast of the location of the crime scene and additional details they acquired from the reporter. Our model can help the police personnel better prepare for the crime scene before hand by giving them a probabilistic score of what type of crime it could possibly be.

* 1. DATA PRE-PROCESSING



As the dataset contains only two numerical columns, X(latitude) and Y(longitude). Using boxplot, we observed the

distribution, identified the outliers and removed them.

We also used StandardScaler2 to standardize the X and Y. The standard scalar removes the mean and scales all values to unit variance.

For categorical columns, we used one-hot encoding to con- vert them into numerical columns. Though simple numerical encoding would also give approximately the same results without drastically increasing the data dimensions, we would like to make sure our model doesn’t fall to the shortcomings

175000

150000

125000

100000

Count

75000

50000

25000

0

LARCENY/THEFT

OTHER OFFENSES

NON-CRIMINAL

ASSAULT

DRUG/NARCOTIC

VEHICLE THEFT

VANDALISM

WARRANTS

BURGLARY

SUSPICIOUS OCC

MISSING PERSON

ROBBERY

FRAUD

FORGERY/COUNTERFEITING

SECONDARY CODES

WEAPON LAWS

Category distribution

Category

PROSTITUTION

TRESPASS

STOLEN PROPERTY

SEX OFFENSES FORCIBLE

DISORDERLY CONDUCT

DRUNKENNESS

RECOVERED VEHICLE

KIDNAPPING

DRIVING UNDER THE INFLUENCE

RUNAWAY

LIQUOR LAWS

ARSON

LOITERING

EMBEZZLEMENT

SUICIDE

FAMILY OFFENSES

BAD CHECKS

BRIBERY

EXTORTION

SEX OFFENSES NON FORCIBLE

GAMBLING

PORNOGRAPHY/OBSCENE MAT

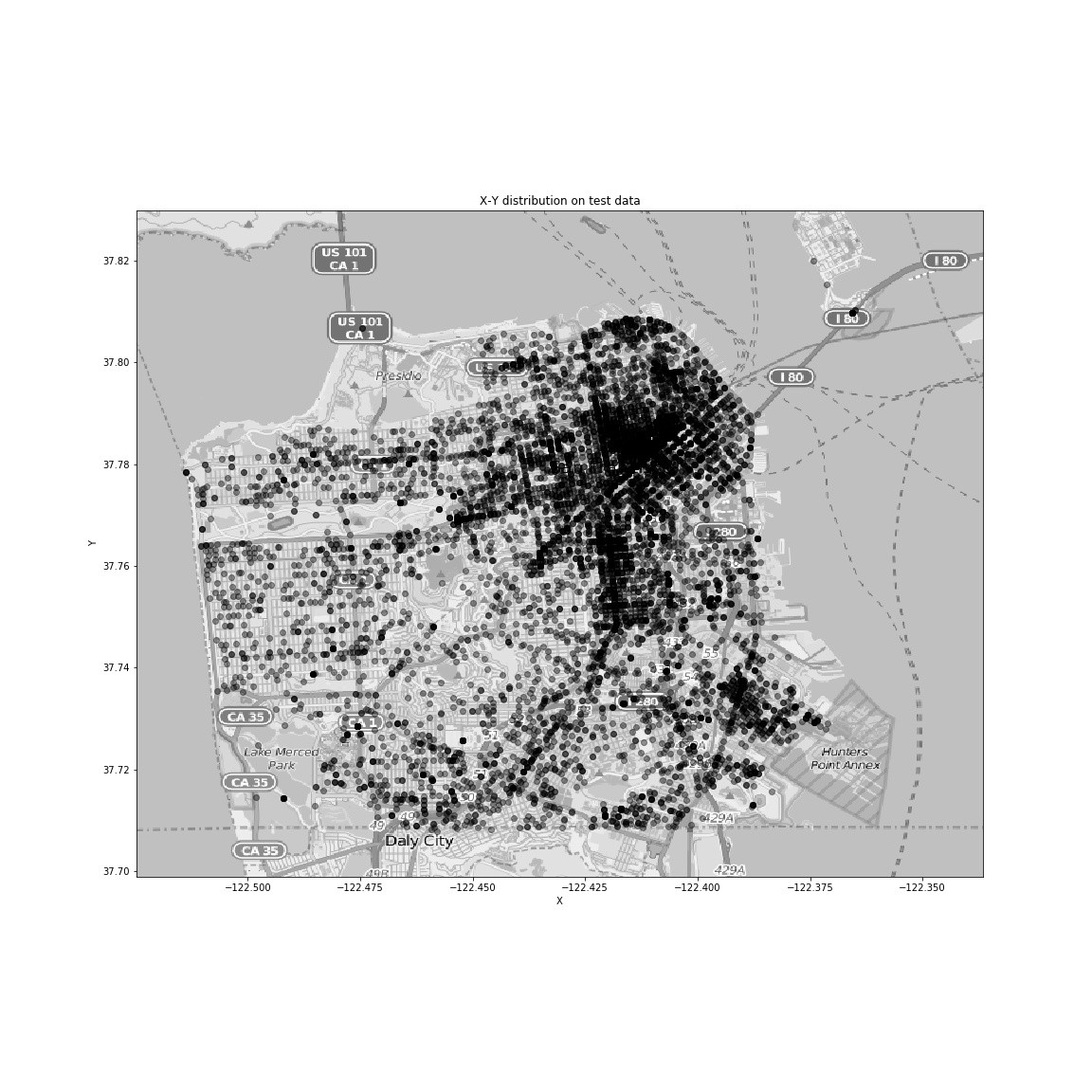
TREA

of numerical encoding.

* 1. DATA VISUALIZATION

There are a total of 36 crime categories. We observed that the distribution for these categories is heavily skewed. There are only two numerical columns X and Y, which are latitudes and longitudes (not Cartesian coordinates). The test data does not contain the Descript and Resolution columns because in practice these columns won’t be present when we are trying to predict crimes. The Address column is the city block number and the name of the street/alley/avenue etc.

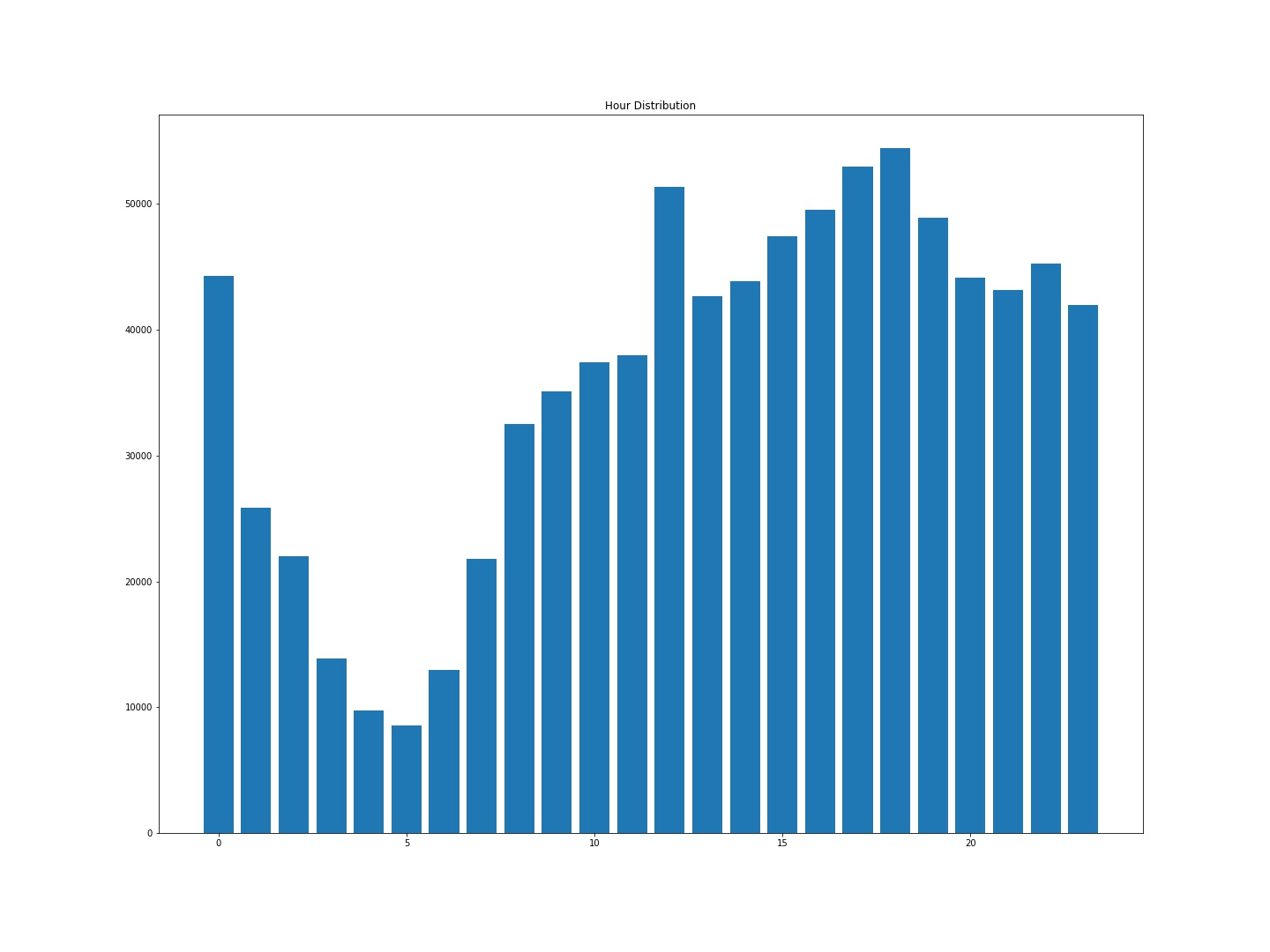
We first observed the distribution of the crimes on the San Francisco map



We first observe the distribution of the classes

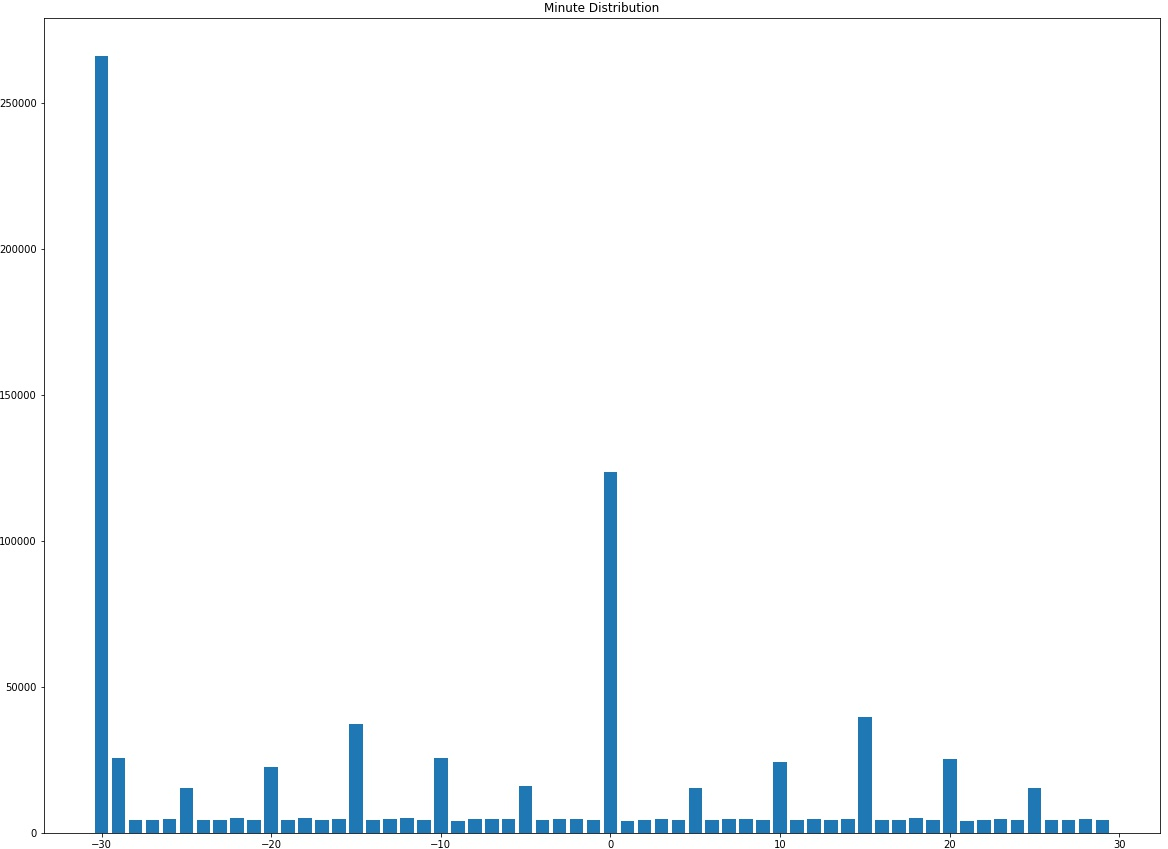
2StandardScaler function from sklearn’s [1] preprocessing module

The class distribution reflects the skewness of the categories in our dataset.

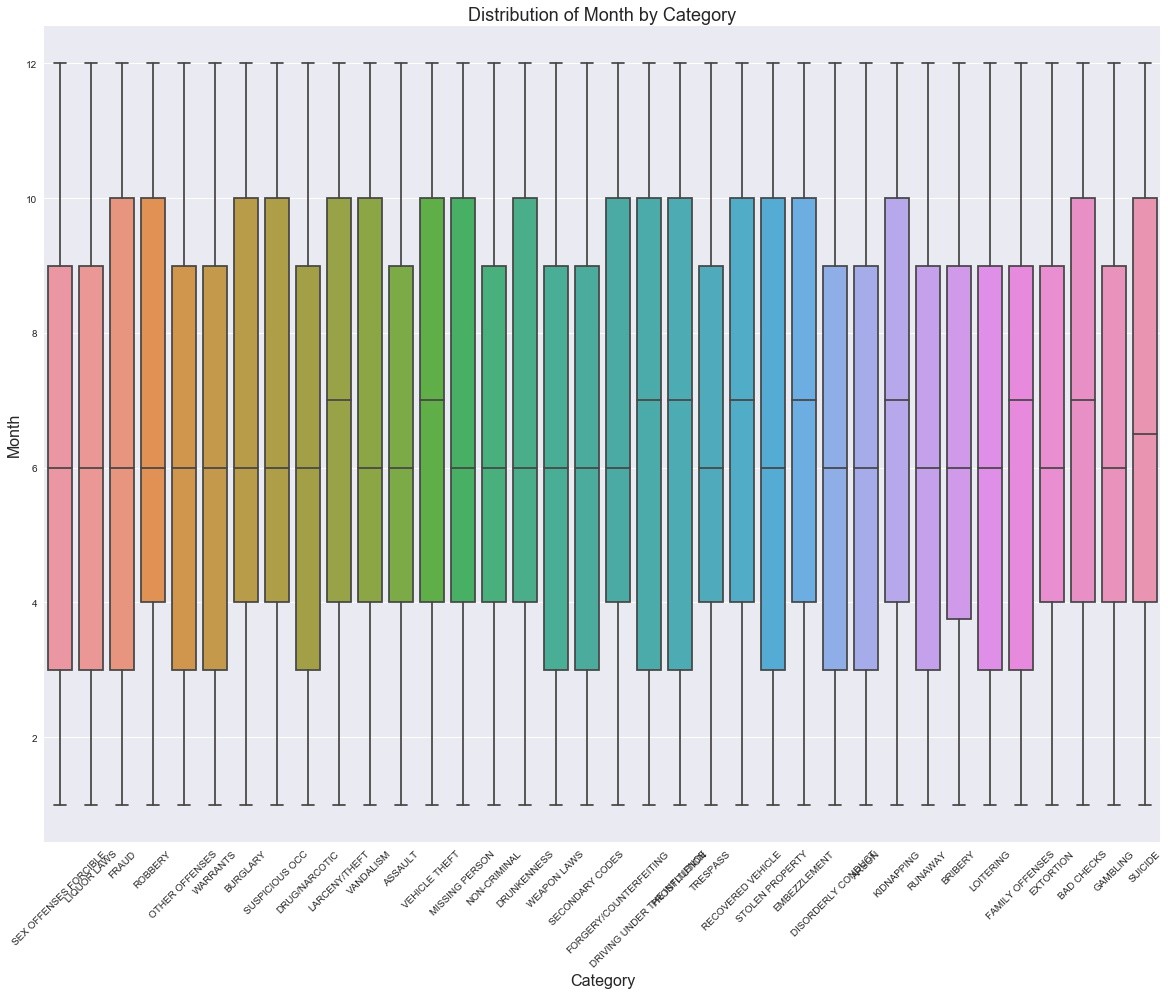
We then tried plotting histograms for year, month, hour and other time based feature to decide which ones are relevant.

We can observed that there is a dip in the histogram of hour. This is because the number of crimes which occur at night time are significantly less than day. We also saw that there is a small peak at around midnight.

We also saw a very peculiar crime distribution based on minutes. The x-ticks tell us that the crimes have been recorded at integer multiples of 5, and most of them have been rounded off to 0 or 30 minutes. This is exactly why this feature is useful, though it is technically not supposed to be.



The month feature is also useful as it divides the data into proper 12 parts as seen below



* 1. FEATURE ENGINEERING

The X and Y features are great, but they don’t make a lot of sense unless observed together. X and Y together give us the the precise location of the crime.

In order to tackle this issue, we use geohashing. Geohashing

[2] is a method of creating hashes of X and Y together. The physical interpretation of these hashes would be regions on the map. As we increase the precision of the hashing function, we increase the number of discreet regions. We found a sweet spot at precision 8 in the pygeohash module. We then one-hot these hashes which will then finally tell us in which hash/region a particular X and Y coordinate lies.

We converted the date column into a python datetime object and extracted key features like month, year, dayofweek, day, hour, minute. These columns are really helpful in getting

intrinsic data like patterns in daily crime frequency, patterns in weekend/weekday crime frequency, as seen in the data visualization section.

Further, in order to generalize the data, we create new columns called seasons(summer, winter, autumn, spring) [3] and time of the day(morning, afternoon, evening, night). This breakdown was done by referring to this [4] for knowing the seasonal and daily weather patterns.

The address column has a lot of data which we can extract. The data is of two types, one with two street names which refers to a road crossing and another where the block number of a area is mentioned. Blocks is a systematic way of categorizing localities in San Francisco. We used regex(regular expressions) to extract the names of the streets and check if an address is a crossing. We also use regex to extract the type of street which is also present in the address column in the form of abbreviations like AV, ST etc.

Another interesting feature is logodds(logarithmic odds). Logarithmic odds gives us the odds of a particular event to happen given an event. We find logodds for the categorical crime to occur given the address of a datapoint. This adds 36 new columns to our DataFrame, with each column giving us the logodds of a particular crime, for each row.

In our initial test data, the descript and resolution columns were also present. Though it technically doesn’t make sense to have them in our test data and also to even try working around with these columns, we tried tweaking with it. We applied TF-IDF (term frequency - inverse document frequency) with ngram range - (1, 2). We got an astonishing

* 1. logarithmic loss. This is clearly because the descript column contains clear description of the category, which will obviously include the category in it.
  2. TRAINING AND RESULTS

The size of the dataset after feature engineering increased 10 fold. It’s hard for machine learning models to converge properly when the data is of high dimensions. In order to reduce the size of the dataset, we project it down to lower dimensions with the help of PCA3 to transform our dataset to a lower dimension.

We trained our data on the following algorithms

* + - LogisticRegression (LR)
    - XGBoost (XGB)
    - SGDclassifier (SGD)
    - EasyEnsembleCLassifier (EEC)
    - bernoulliNB (bNB)

The logarithmic loss on these models were the following

3PCA from sklearn’s decomposition module

TABLE I

LOGLOSS ON VARIOUS ALGORITHMS

|  |  |  |
| --- | --- | --- |
| S.no | Algorithm | logloss |
| 1 | LogisticRegression | 2.362 |
| 2 | XGBoost | 2.366 |
| 3 | SGDclassifier | 2.41 |
| 4 | EasyEnsembleCLassifier | 2.356 |
| 5 | bernoulliNB | 2.643 |

* 1. CONCLUSION

With a logloss of around 2, we see that it is quite difficult to predict the crime based on just spatial and temporal data, accurately. We think that 911 transcripts which include data about how the crime was reported, assuming it was reported via a 91 call, will be helpful.

ACKNOWLEDGMENT

We would like to thank our teaching assistant Ravi Teja for helping us out in the initial stages in numerous occasions, Tejas Kotha for helping us out with different ensemble techniques and also with the report. Special thanks to Raghavan sir for, well everything. His highly detailed lectures on various topics helped us understand what we were actually doing, rather than just copy pasting code from tutorial sites.

REFERENCES

1. F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion,

O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vander- plas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, “Scikit-learn: Machine learning in Python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

1. W. McGinnis, “Geohashing,” 2015, [First release - Jan 7, 2016]. [Online]. Available: \url{https://github.com/wdm0006/pygeohash}
2. Y. Abouelnaga, “San francisco crime classification,” *CoRR*, vol. abs/1607.03626, 2016. [Online]. Available: [http://arxiv.org/abs/1607.](http://arxiv.org/abs/1607) 03626
3. timeanddate, “timeanddate,” 2018, []. [Online]. Available: \url{https:

[//www.timeanddate.com/sun/usa/san-francisco}](http://www.timeanddate.com/sun/usa/san-francisco)

* 1. PROJECT FILE LINK

https://drive.google.com/open?id=1re04PjcLHnEihsySEhZA4dvrVlV0ZUsI

This link contains the .csv files required to test the pickle file. Just download the files and extract them into the project folder.