Park-Safe (16-791 Course Project)

Vivek Gupta, Aashi Gupta, Gautham Kumar Vedam

Carnegie Mellon University, {vivekgup, aashig, gvedam}@andrew.cmu.edu

Executive Summary

In this **Park-Safe** project, we analyzed different factors affecting vehicle crimes in the city of San Francisco. Vehicle thefts are rampant in SF and through this project, we aim to provide a data-driven solution to combat this problem. Specifically, we want to recommend safe parking spots to people looking to park their vehicle.

We worked with 4 datasets capturing different factors which might potentially impact vehicle thefts. Exploratory analysis reveals that vehicle thefts are more likely to happen between 6-8 PM and on Fridays. Populous neighborhoods like 'Mission' and 'South of Market' neighborhoods experience higher crime incident rate.

Next, we used these features to construct ML models which predict the probability of vehicle theft at a given spot and at a particular time. For this task, we treated theft incidents as the positive class and sampled records uniformly at random to form the negative class. We experimented with 4 ML techniques and performed feature selection and hyperparameter optimization to maximize the performance of our methods.

The best performing model (Random Forest in our case) is then utilized to build a webapp which suggests safe parking spots in real-time around user's location. The app inputs the current location co-ordinates and radius of search. It then plots the parking spots on the map which are color-coded to represent their safety level. The spots can be further investigated to reveal more information.

1 Introduction

Park-Safe is data-powered application which provides real-time recommendations of safe parking spots to users in the city of San Francisco, California. The goal of this project is to use historical vehicular crime data to analyze the time and location demographics of the incidents. The result of this analysis is encapsulated in the form of an interactive application which, provided the time and location information, evaluates the vehicle theft safety of nearby metered parking spots. We present a web-based prototype of the application.

2 Motivation

San Francisco city is the cultural, commercial, and financial center of Northern California. The city is plagued by a car break-in epidemic with 25,677 car break-in reports in 2019, according to data from the San Francisco Police Department [1]. The police department only makes arrests in about 1% of cases. Vehicle thefts shot up almost 21% in San Francisco from April to May in 2020, compared with the same period last year, in spite of nearly non existent traffic due to Coronavirus [2].

Identifying safe parking spots will assist the city locals,tourists and car rentals to park their vehicles securely. In a broader context, it may also help people looking to purchase or rent property in a safer neighborhood.

3 Data-sets

We work with the following 4 data-sets:

- **Police Department Incident Reports: 2018 to Present** [3] published by San Francisco Police Department, is the main data-set of vehicular thefts
- **Parking Meters** data [4] published by the Municipal Transportation Agency of San Francisco, includes a list of all parking meters in San Francisco
- **Income by Location in San Francisco** data [5]

provided by the U.S. census contains median household income for all SF census tracts in 2018

- **Population by Neighborhood in San Francisco** data [6] provided by the U.S. census contains density and absolute population for all SF neighborhoods in 2018

3.1 Description and Filtration

The Incident Report dataset [3] contains $\sim 393k$ records. We filter out records with `Incident Description` beginning with ”Vehicle, Stolen”. This reduces the data to 14872 records. We also filter the following features – `Incident Datetime, Analysis Neighborhood, Point`

The Parking Meters dataset [4] contains $\sim 33k$ parking meters. We filter out the active meters which reduces the size to 27274 meters. We keep the following features – `Parking_ID, Point`

3.2 Cleaning and Preparation

Original `Analysis Neighborhood` feature is noisy – many records have multiple neighborhoods clubbed together like ‘Financial District/South Beach’ and ‘Oceanview/Merced/Ingleside’. This leads to only 41 unique neighborhoods in the dataset. Using San Francisco neighborhoods geojson information, we map each incident spot to its correct neighborhood using the location co-ordinates, resulting in 116 neighborhoods.

4 Factor Analysis

We study the distribution of vehicle crimes along 4 dimensions – time, population, household income and location of the incident.

4.1 Time of Incident

We study the time distribution of vehicle thefts at hourly, daily and monthly frequency. We find that most crimes occur in the evening from 6 PM to 8 PM. This implies that – *vehicle thefts are NOT more likely to occur during wee hours of the day*. Vehicle thefts are also more likely to transpire on Fridays and in the months of January, July and August. This might relate to tourists/locals going for outdoor activities on weekends.

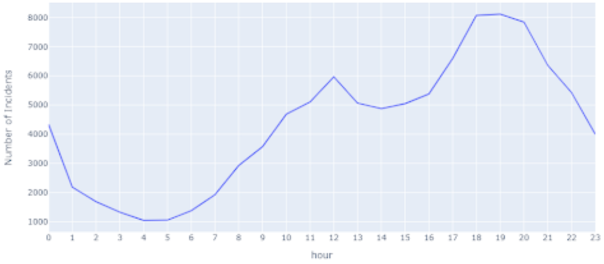


Figure 1: Distribution of crime by **hour**

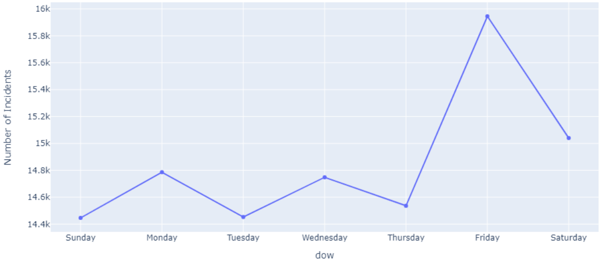


Figure 2: Distribution of crime by **day of week**

4.2 Population

We calculate the Pearson correlation coefficient between the population of a neighborhood and the number of thefts in that neighborhood to be **0.54**. Conducting a permutation test reveals an empirical **p-value of 0.001** of obtaining this result. This implies that – *more populous neighborhoods tend to have higher crime rates.*

4.3 Household Income

We calculate the Pearson correlation coefficient between the median household income of a neighborhood and the number of thefts in that neighborhood to be **0.37**. Conducting a permutation test reveals an empirical **p-value of 0.02** of obtaining this result. This implies that – *more wealthy neighborhoods tend to have higher crime rates* although the relationship is not very strong.

4.4 Location

We perform the location-based study of vehicle theft at 2 levels of granularity – neighborhood level and smaller regions around the incident spots. We believe location to be an important factor since certain areas are notorious for being unsafe.

4.4.1 Neighborhood level

Exploratory analysis reveals that **Mission** and **South of Market** neighborhoods have significantly higher incident rates. This observation is consistent across different time frames in the dataset.

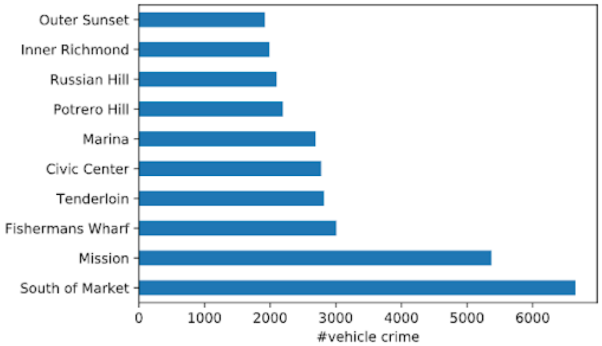


Figure 3: Distribution of crime by neighborhood

4.4.2 Incident-spot level – crime hot-spots

Neighborhoods cover geographic areas that are still large to accurately characterize vehicle thefts. These thefts are often localized to specific streets, markets, etc (hot-spots). Therefore, we need to conduct our location-based analysis at a higher granular level.

We model our theft-incident spots as a connected undirected graph with incident-spots as nodes and edge weights measuring the geographic closeness of the connecting spots. Performing a graph clustering algorithm yields groups of closely connected incident-spots which we can treat as hotspots.

Technical details: Since the graph has a large number of nodes (14000+), treating it as densely connected would render the clustering algorithm computationally infeasible (for the project scope). Therefore, we only consider edges between spots which lie within 500 meters of each other. We transform the distance into a similarity metric using negative exponentiation. This is required for the clustering algorithm. We perform **Markov Clustering** [7] on the graph which yields 594 clusters. Dropping singleton clusters reduces the count to 574. The largest 20 clusters obtained are visualized in Figure 4

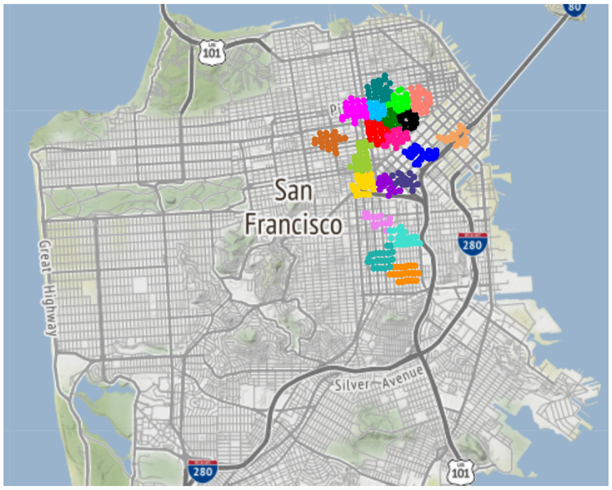


Figure 4: Top 20 largest clusters

5 Modeling

The goal of this project is to suggest safe parking spots. We model it as a classification task – given the time and location information, predict if the sample is a crime-incident or not. Once we have a trained ML model, we estimate the safeness of parking spots (nearby our query location) by predicting the probability of those spots being crime-incidents. This will assign a $[0-1]$ rating to each parking spot which reflects the chance of a vehicle being stolen from that parking spot.

5.1 Data-set Creation

The Incident Report data-set contains vehicular crime records. We assign positive (+) label to these crime-incident records. We need to artificially insert negative (-) records. **We randomly sample 6 negative data-points for every positive data-point**. The features sampled are latitude, longitude, hour_of_day, day_of_week, month. The coordinates are ensured to lie within the SF region. The ratio of 6:1 is kept to ensure the dataset size remains within manageable computational limit. Figure 5 shows the distribution of (positive) and (negative) data-points.

5.2 Feature Engineering

We use the following features from the constructed data-set directly – latitude, longitude, hour_of_day, day_of_week, month, neighborhood population. We also use our cluster analysis from section 4.4.2 to engineer a feature `Location_Prob` measuring location-based probability of crime. We describe the computation of this attribute below.

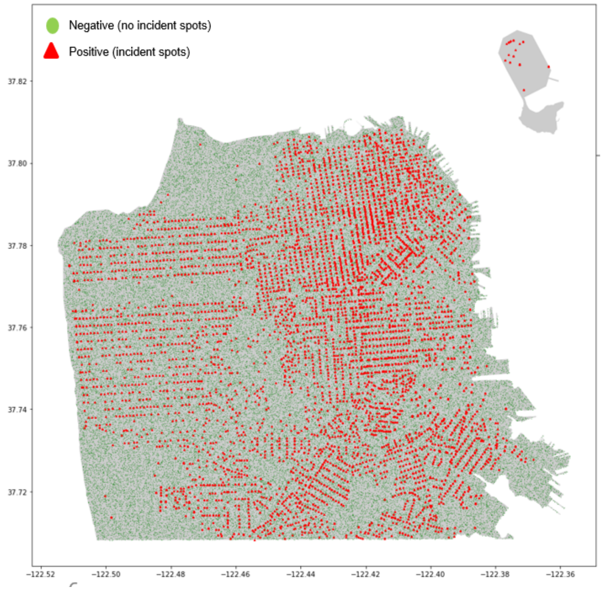


Figure 5: Distribution of Data-Points

Location_Prob: The clusters obtained are filtered to keep only the ones with size at least 30. This removes locations which do not report frequent repeated thefts and therefore, are not typical crime hot-spots. This reduces the number of clusters to 149. We model each cluster as a bi-variate normal distribution with latitude and longitude as the two axes. Using this modeling scheme, we can compute the probability of theft at a given location as the weighted sum of probabilities output by each Gaussian. The weight of a distribution corresponding to a cluster is the ratio of size of the cluster to the total size of all clusters. To compute the probability from one Gaussian, we consider a 6×6 square meter area around the desired location. This represents the typical size of a car or a parking spot.

5.3 Training

We train the following 4 ML models for the classification task – Logistic Regression, Naive Bayes, Decision Tree, Random Forest. We choose these models as they are popular standard approaches to classification. Contrary to deep neural networks, these ML techniques are very interpretable and offer sufficient power (especially RF) needed for our task.

5.4 Feature Selection

We employ Recursive Feature Elimination (RFECV) technique to identify the most useful classification features from our ML models. Default *sklearn* hyperparamter setting is used (num_fold = 5, etc) except the `scoring` method which we customize to select features maximizing TNR at 0.8 recall. The results are outlined in Table 1

Model	Selected Features
LR	longitude
NB	longitude, hour, month, latitude, dow
DT	latitude, longitude
RF	latitude, longitude

Table 1: Features selected using RFECV

5.5 Hyperparameter Optimization

We experiment with different hyperparameter configurations for all 4 ML models and compare the TNR of the resulting models at 0.8 recall. This enables us to identify the best version of each model as shown in Table 3. The classes are weighed inversely to their frequency (class weight = 'balanced') to compensate for the class skewness.

5.6 Evaluation

Using the features selected by RFECV, we train the final version of our ML models. We perform a stratified 10-fold cross-validation to evaluate model performance. The ROC curve obtained is visualized in Figure 6. Since we are interested in identifying crime-incidents (positive samples) correctly, i.e. we want a high recall (TPR), we zoom in the high TPR region of the ROC by applying a log-transform to FNR and plotting it against TNR in Figure 7.

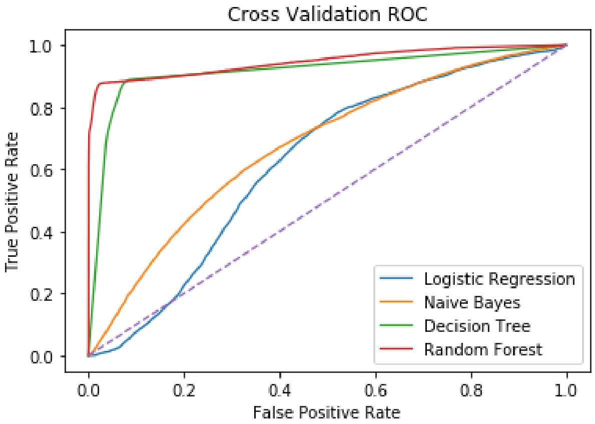


Figure 6: ROC of ML models

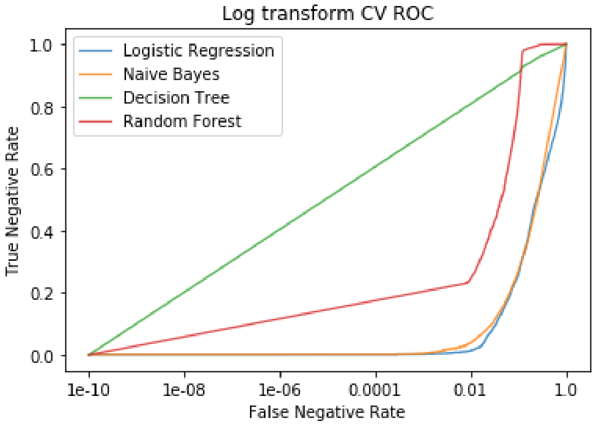


Figure 7: ROC - Log transform

We visualize the decision boundaries learned by the Decision Tree model overlayed on the map of SF in Figure 8. The green areas denote safe regions while the red pockets denote crime-prone regions. The yellow spots are the actual crime incident spots from the training data.

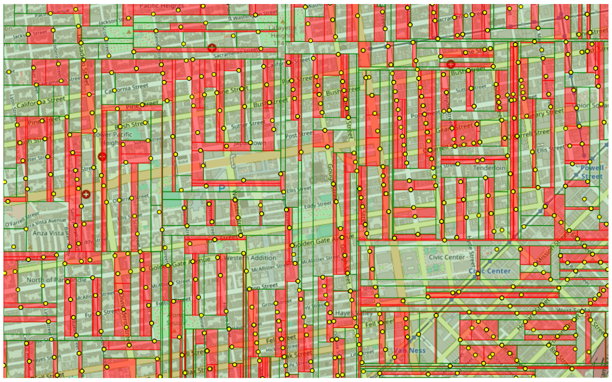


Figure 8: Decision Tree – decision boundary

To compare the classification performance of different models, we need to set a probability threshold for classifying a prediction as crime-incident or not. The threshold value is dictated by the application stakeholders and business use-case. For our application, we set a threshold which provides 80% recall. The AUC, precision, TNR and accuracy for different ML models are outlined in Table 2. We observe that **Random Forest is the best performing model at 0.8 recall with a precision of 93.1%**

6 Web Application

We use our final Random Forest model to build a *Flask* based web-based application for suggesting safe parking spots. The app is fed the geographic coordinates (which can be obtained via GPS) and it displays all parking spots within a user specified radius (up to 1000 metre). The spots are color-coded (unsafe and safe) and can be further explored to view the ”safety index” (1-theft probability) along with other metadata like meter-ID, distance, etc. We show a snapshot of the application in Figure 9.

Model	AUC	Precision	TNR	Accuracy
LR	0.623	0.197	0.456	0.505
NB	0.676	0.190	0.430	0.483
DT	0.916	0.694	0.940	0.923
RF	**0.947**	**0.931**	**0.990**	**0.963**

Table 2: Model performance at 0.8 recall

Model	Hyperparameter setting	TNR
LR	**L2 penalty**	**0.456**
	L1 or elasticnet penalty	–
NB		**0.430**

DT	min_samples_leaf = 1
	max_depth = 20
	min_samples_split = 2
	min_impurity_decrease = 1e-6
		0.830

	min_samples_leaf = 2 **max_depth = 38** **min_samples_split = 4** **min_impurity_decrease = 1e-7**	**0.930**

	min_samples_leaf = 3
	max_depth = 40
	min_samples_split = 6
	min_impurity_decrease = 1e-5
		0.910

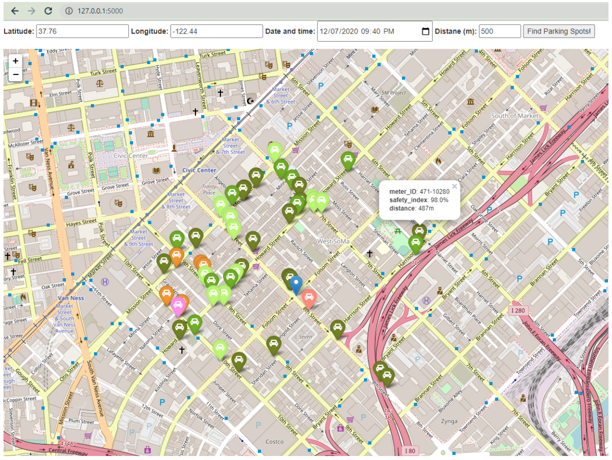
RF	n_estimators = 50
	min_samples_leaf = 2
	max_depth = 38
	min_samples_split = 4
	min_impurity_decrease = 1e-7
		0.989

	n_estimators = 100 **other hyperparameters same**	**0.990**
	n_estimators = 150 other hyperparameters same	0.990

Table 3: TNR@ 0.8 recall for varying hyperparameter configurations of ML models

7 Business Implication

As motivated in section 2, car theft is a serious problem in San Francisco. Providing a solution to combat this issue is a promising endeavour. The application interface shown in Figure 9 provides an intuitive, user-friendly way to find safe parking spots near a given location. However, this is a web-based prototype. To turn this project into a successful business model, the client needs to develop a cellphone app which integrates the ML model with the geo-coordinate info from the GPS sensor of the device. A phone app would make the product accessible to a much larger customer base.



Latitude: 37.76 Longitude: -122.44 Date and time: 12/07/2020 09:40 PM Distane (m): 500

meter_ID: 471-10280
safety_index: 98.0%
distance: 487m

Figure 9: WebApp Interface

8 Risk and Mitigation Plans

There are several assumptions made as part of our analysis which might not hold true in reality:

- **Ratio of negative to positive data samples**: We kept this ratio at 6:1 to keep the data size within compute constraints. However, the actual ratio is likely to be much higher than this value. This ratio captures the prior odds of a vehicle theft.

- **Random sampling of features**: All time and location based features are sampled uniformly at random from their possible ranges. There might exist more appropriate sampling strategies which take crime events data into account.

- **Operating point**: The optimal operating point, i.e. the threshold for binary classification decision, depends on the relative cost of false positive error versus false negative error. As noted in section 5.6, setting the operating point to obtain 0.8 recall is a business decision which affects our model choice. Recall of 0.8 effectively implies that we want our model to correctly identify 80% of the crime incidents. The optimal operating point can vary from user to user (clients).

9 Future Work

The scope of this project can be expanded beyond SF to other cities. If compute is not a hindrance, the dataset size can be scaled up to provide a more accurate representation of reality. Other features like "tourist spot" can be tested to check if they add value. More sophisticated modeling techniques like deep neural networks can be experimented with to keep pace with increasing data complexity. Error analysis can be conducted to check if the model has systematic error patterns. It might also be interesting to understand why some features like time do not add improve model performance. Finally, the webapp can be transformed into a mobile app which can be used by drivers looking to safely park their vehicles.

References

[1] https://www.businessinsider.com/san-francisco-is-proposing-reimbursing-car-break-in-victims-2020-2.

[2] https://www.ktvu.com/news/vehicle-theft-spikes-during-covid-19- pandemic.

[3] *DataSF - Police Department Incident Reports: 2018 to Present*. https://data.sfgov.org/Public-Safety/Police-Department-Incident-Reports-2018-to-Present/wg3w-h783.

[4] *DataSF - Parking Meters*. https://data.sfgov.org/Transportation/Parking-Meters/8vzz-qzz9.

[5] *DATA USA - San Francisco, CA*. `https://datausa.io/profile/geo/san-francisco-ca#income_geo`.

[6] *Statistical Atlas*. https://statisticalatlas.com/neighborhood/California/San-Francisco/Financial-District/Population.

[7] S. Van Dongen, *Graph Clustering by Flow Simulation*. PhD thesis, University of Utrecht, The Netherlands, 2000.