Advanced Machine Learning

Arpita Jena, Gongting Peng
5th March, 2018

Forecasting Parking in San Francisco

Abstract

Parking is one of the biggest concerns for people in San Francisco. This report is a summary of how we managed to predict the availability of parking spots per street in the city of San Francisco using machine learning methods. The report first introduces the data, then summarize the machine learning models we use and present the conclusion at last.

The data used for modeling is acquired from a private source. We have extracted information from three available data sources. The details regarding these can be found in *'Data Description'* section. We have extracted some features from existing data mainly using date, mean encoding of target and geographical location of parking spots. The training data is comparatively small in size and has class imbalance. We have tried to implement logistic regression, random forest and XGBoost for the purpose of prediction after handling the imbalance factor. Finally, we have proposed an ensemble of all three to provide a better prediction accuracy.

Team Details

Name: ParkingSF
Members: Arpita Jena, Gongting Peng
Kaggle IDs: arpitajena, tinapeng

Data Description

To predict the parking spot, there are four available datasets, provided by a private source that help people find street parking. The 'parking record' dataset contains information about latitude longitude of certain parking spots. There is another dataset containing sensor information recorded by all the sensors implemented underground by San Francisco Government years ago. It records whether there is a car in a specific spot, how long that car stays, the occupation rate of the spot, total time recorded, vacant time, etc. The train and test data were manually entered by the employees at the company that include the date and time, street name, where the examined part of the street locates, the length of the examined part, whether there is parking spot at that time, and if yes then the number of parking spots available.

The test data spans over a period of roughly 2.5 years from March, 2014 to November, 2016 where as we only have data for 3 months i.e. from January, 2014 till March, 2014 for training purpose. But the argument for the validity of the model is that the availability pattern of parking spots doesn't change much throughout the years.

Feature Engineering

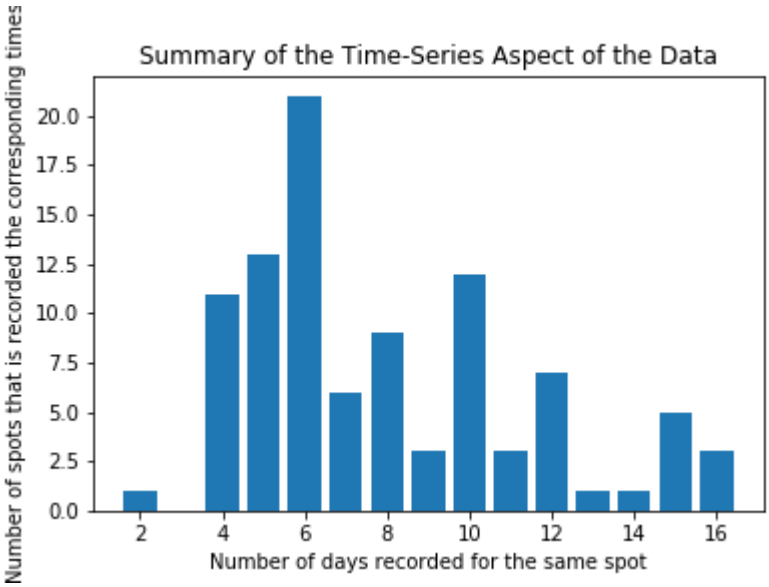
We combined the sensor data with train and test dataset to generate features to process our data for modeling. The features include the following:

- **Time related features:**
    1. isweekday: Whether the day is a weekday, or a weekend
    2. Dayofweek: The day of the week of that specific day
    3. Month: Month of the recording time
    4. Year: Year of the recording time
    5. Dayofyear: The day of the year
- **Geological features:**
    1. The longitude and latitude of the parking spots
    2. The intersections of the parking spots
- **Mean encoding:**
    1. For features like 'from_to, hr, isweekday, Dayofweek, Month' we mean-encoded the variable 'any_spot' and 'real_spot', where the first one indicates whether there is parking spot and the latter one shows how many parking spots are there.
    2. For all possible Null values, we encoded them with the global mean of the entire training data

Exploratory Data Analysis

To understand our model better, we did some exploratory data analysis.

The training data we have is the data from 1/18/2014 to 3/28/2014, which is about 2 months period. There are limited data for the same spot along the time. For the same spot the number of record is as the following graph:



As shown in the graph, for each spot, there are very limited observations: the most common number of records for a spot is 6 (there are 21 spots that are recorded 6 times), while the majority number

of record for each spot is under 10 times. Given the fact that we do not have enough data for each spot along the time, time-series analysis may not a good idea.

Figure below is a simple visualization of how many labels we have for each category: We have 699 data points that shows no parking and have only 401 data points where parking is available.

There is substantial class imbalances as seen above. In order to avoid the classifier learning one class better than the other, sampling was done to even out the class imbalance. We resampled 401 data points randomly from zero-labeled data to make both labels of equal length.

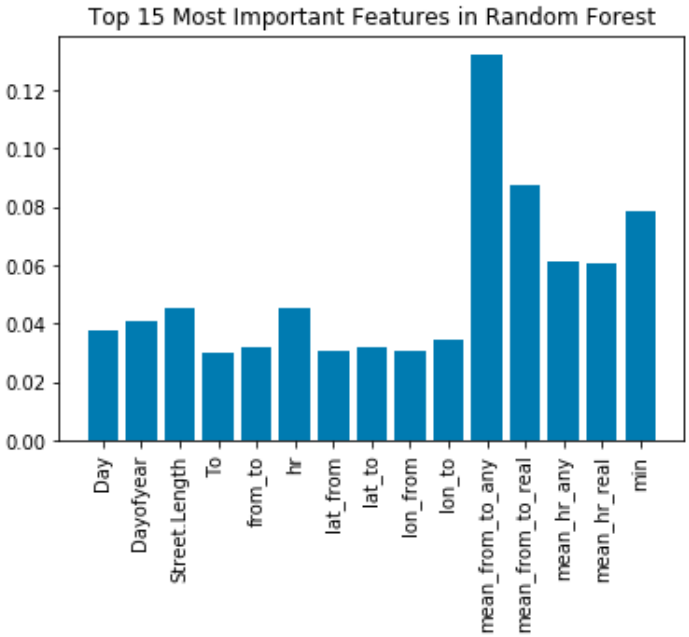
Techniques Overview

We used Regularized Logistic Regression, Random Forest, and XGBoost to predict the parking spot. We splitted data into training and validation set to tune the hyper-parameters in those models. The best result is given by Random Forest. The metric we chose to optimize is F-0.5 score, because we care equally about precision and recall. Then we used ensemble of the results from all three models for better performance.

Experimental Results

- **Random Forest:**

  For Random Forest, we fit a Random Forest Classifier using sklearn.ensemble.RandomForestClassifier with all the available features first and plot the top 15 most importance features given by the model for future use. Figure 3 gives the visualization of the significance of the features:



We chose the features with more than 0.01 importance and fit the Random Forest Classifier again using GridSearch to find the best hyper parameters, then fit the model again with the given best combination by GridSearch.

- **XGBoost:**

Gradient boosting (We have used xgboost package) is a powerful tool in terms of prediction. After tuning the hyper parameters on the training data, we have trained the model to minimize 'eval' error by creating a watchlist for validation error.

- **Logistic Regression:**

Logistic regression (sklearn.linear_model import LogisticRegression is used here) is also a commonly used method for binary classification. To better generalize our model, we chose the penalty method to be 'L2'. To choose the best regularization strength, one of the most important hyper parameters, we tried with difference values of C (inverse of regularization strength; smaller values implying stronger regularization) and the corresponding performance is shown in the following table:

C (Penalty)	0.001	0.010	0.100	1.000	10.000	100.000
Accuracy	0.708075	0.714286	0.745342	0.73913	0.732919	0.73913

Given the accuracy on the validation data, we chose the penalty level to be 0.1 for best performance of the model.

- **Comparison:**

The following table shows the overall performance of the three methods we chose on validation set:

	Random Forest	Logistic Regression	XGBoost
F-0.5 Score	0.724	0.738	0.775
Precision	0.721	0.747	0.759
Recall	0.738	0.738	0.845

Conclusion & Lessons Learned

The data we have is very limited and scattered in terms of location and time. In this case time series may not be the best idea. Tree ensemble as well as regularized logistic regression usually have comparably good accuracy. Huge time lag between train and test also contribute towards low accuracy of the model. The overlap between parking record and train data is very low which restricts us to generate useful information for modeling.

The important lesson we have learned from the project is that not rely too much on the training data when it does not have enough coverage of the situation. Instead, we should think carefully about our approach and stick to the methods we choose. We also learned how to use geolocations as features. Also, to deal with class imbalance we can opt the following methods.

- Random undersampling
- Random oversampling
- Cluster based oversampling

Though each method has its unique advantages and disadvantages, we should try all of them and choose the one with better performance.