Team Box and Whiskers: Sarah Teichman, Tim Lee, and Jonathan Che
STAT-495 Advanced Data Analysis
December 22, 2017

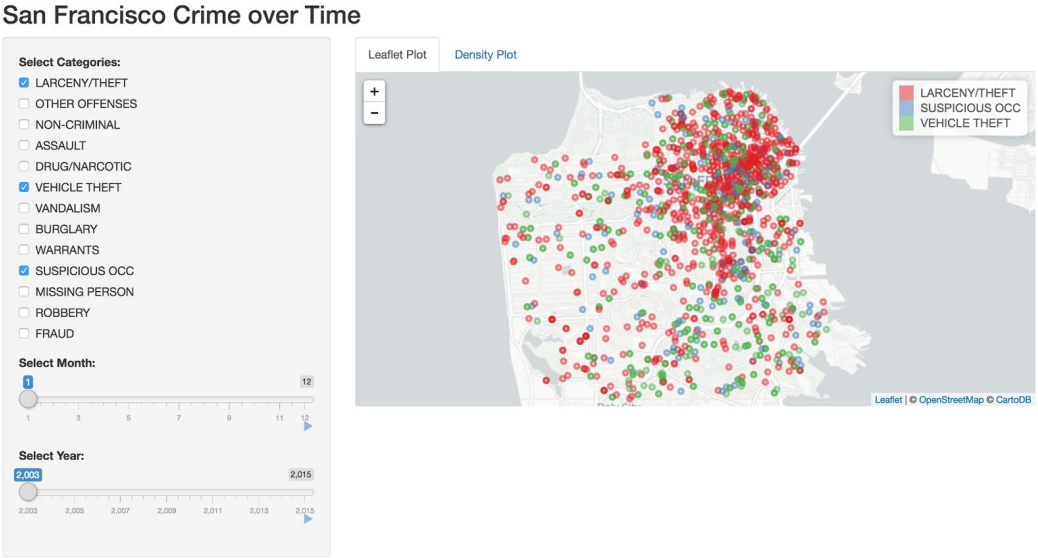
Kaggle: San Francisco Crime Classification

In our project, we classified crime in San Francisco between 2003 and 2015 using variables related to time and location. In our data, we were given the date and location information (latitude and longitude, the police department district, and the address of the crime).

For our exploratory data analysis, we first made several plots to look at time trends. The most interesting trends appeared in the hour and minute information. Most crimes followed common waking/sleeping patterns. For example, larceny and theft (the most common category of crime overall) was the most common crime category during waking hours. Interestingly, late at night on weekends, assault was the most common crime. Also, from midnight through the morning commute, other offenses (usually traffic violations) were the most common.

When we examined the minute data, we noticed peaks in five minute intervals for each category. However, these peaks were less noticeable in certain categories, like drugs and narcotics and warrants. This prompted us to make a variable that encoded whether or not the crime was recorded as happening during a multiple of five minutes. Intuitively, this variable could act as a proxy for how "real-time" the crime was being policed.

To study location/time interactions, we deployed a Shiny app. The app's first tab uses leaflet to plot crime location, which allows users to zoom/drag the map and click on specific points to see the time, address, etc. of the crimes. The app's second tab shows density plots of the location of crime in a month. This enables users to get a higher-level view of overall crime distributions. The app has a sidebar which lets the user choose the crime category, month, and year to view in the app.



We also noticed clusters of points along certain lines, which corresponded to three large streets (Geary, Mission, and Market). We used this address information to create variables that indicated whether the crime occurred on these streets.

Our work exploring and understanding the data helped us build models and predict the category of crime. We ended up focusing on random forests (RF) and multinomial logistic regression. Our best RF submission gave us a Kaggle (multi-class log loss) score of 3.404, and our best multinomial model gave us a Kaggle score of 5.204.

In the project, we ran into many computational issues. For our multinomial logistic regression, we had to balance the complexity of the model with our computational constraints. We used all of the data to train the model, but only fit to the top thirteen categories (predicting 0 for the other 26 categories, which likely inflated the multi-class log loss). If we had wanted to predict for each of the categories, we would have had to sample from our data. Similarly, we were only able to train our RFs on about 100k of the 800k+ possible training observations. A learning curve showed us that training on more data would have likely improved our outcomes by a significant margin. Finally, we initially tried to use k-nearest-neighbors on our data, but found that it would take far too long to predict on the 800k+ test observations that would be needed for a Kaggle submission. These issues could be consequences of our decision to use the R shared project for our project instead of contributing to one repository on our local machines. A server with more computational capacity would have let us use different models on more data.

As a side note, we made a very interesting mistake with our first multinomial logistic regression. Submitting the model gave us a Kaggle score of 2.639, which was the lowest score that we got on any of the models we used! Sadly, we had incorrectly labeled our predictions. When we correctly resubmitted, our Kaggle score ballooned to 13.494. This showed us that "Kaggle scores aren't everything," i.e. that it is possible to randomly stumble across good predictions even if your model is completely incorrect.

Through working on this project, we learned that real data are messy and prediction is difficult. In addition, large datasets are cumbersome in R, and particularly in the shared R server. We had to balance the computational limits of the R server with our goal to create models that would provide the best predictions. Moving forward, we could move the project off of the server to increase computational capacity, engineer more features, and fit spatial models to the data to see how they perform. There are many ways in which we can improve our project, which is exciting because Tim is going to be continuing with it for his statistics comps!