Brayden Cleary
Saturday, August 22, 15
GA Data Science Summer ‘15

Predicting SF Crime Category and Smoothie Sales

Problem statement and hypothesis

Fiction: San Francisco is a magical land filled with benevolent unicorns building payment-processing startups. Fact: Crime occurs in San Francisco. To be sure, from January 1st, 2001 to May 10th, 2015, the SFPD documented over 1.5 million crimes and that figure doesn’t even include the guy I see defecating on the street 4 nights a week.

Looking at crime data is like having drinks with a co-worker that shares too much. You’re excited about what you might learn but also scared to learn too much. If a crime were to take place in my neighborhood, what kind would it be? What are the violent/non-violent crime hot spots in the city? What night of the week is it safest to walk through the tenderloin at 2am? Curious to learn more about the city I live in, I decided to compete in Kaggle’s San Francisco Crime Classification challenge for the first half of my project.

The second half of my project focuses on a personal business goal: getting rich off selling smoothies. On June 1st, 2015, I founded a startup called Redwagon Smoothies (RWS) that has yet to start. RWS uses the latest in wagon and blender technology to bring smoothies in mason jars to a park near you. Although the inputs to starting RWS are low, the safety risks are high. A customer writing a fraudulent check in exchange for one of my delicious smoothies, my wagon getting hit by a drunk driver, or my wagon being stolen by some crazed thief are all risks I face before hitting the pavement. Using a subset of the Kaggle SF crime data, I aimed to predict which SF zip codes I should target and which ones I should avoid at any given time.

Description of your data set and how it was obtained

Kaggle provided a train and test dataset, each with about 850,000 crime entries. The columns in the training dataset are: Dates, Category, Descript, DayOfWeek, PdDistrict, Resolution, Address, X (long), and Y (lat). Since Descript and Resolution aren’t present in the test dataset and Category is the value I'm predicting, initial possible features were: Dates, DayOfWeek, PdDistrict, Address, X (long), and Y (lat).

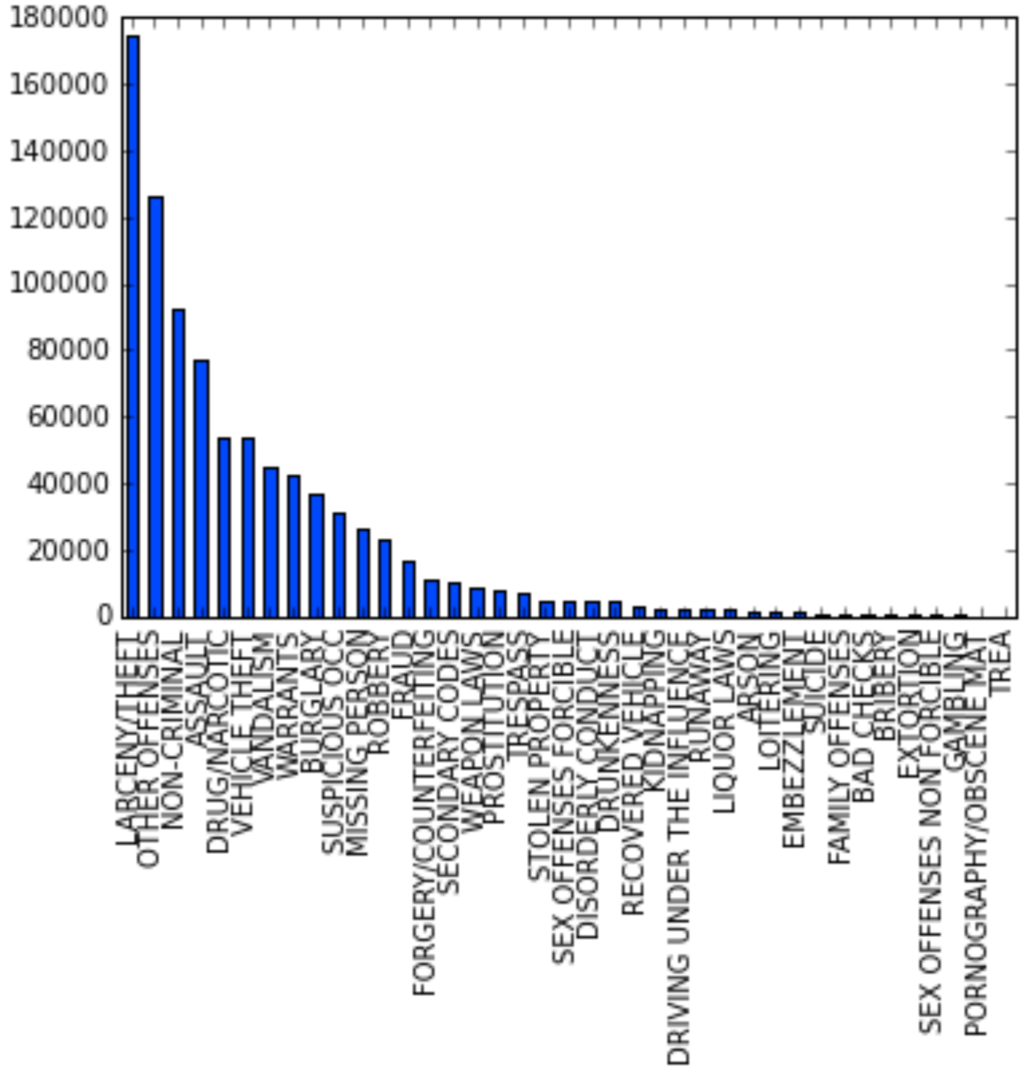
Description of any pre-processing steps you took

To be able to complete the second half of my project, I had to find a way to logically group crimes by area. The askgeo api provided an easy way for me to translate a lat/long point to a zip code and pull in about 160 datapoints based on the 2010 census with it. Although all of these features didn't prove useful in the development of my models, some did prove to be more important than the initial features Kaggle provided.

In terms of pre-processing, askgeo wasn't able to translate 67 of the lat/long cords to a zip code so I dropped those. I also renamed X and Y to long and lat respectively. For day of week and pd district, I created dummy columns for all of the unique values and I also created 5 time-related columns: is_weekend, day_of_month, month_of_year, year, and time_of_day_bucket (values 1 through 8 where 1 is early morning and 8 is late night).

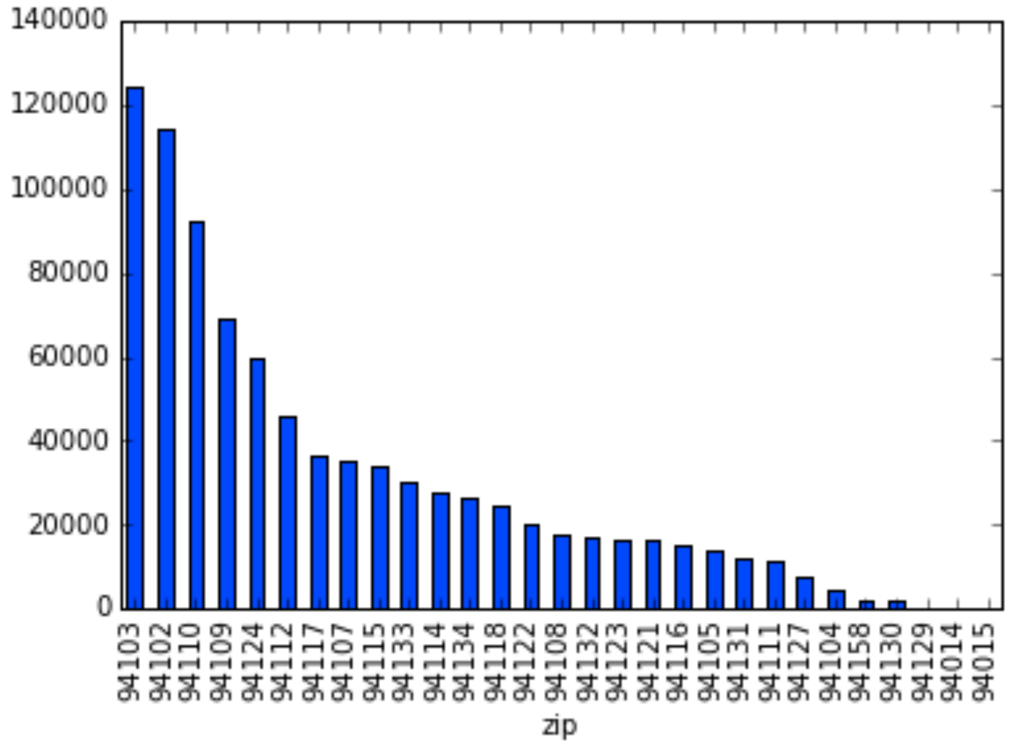
What you learned from exploring the data, including visualizations

Looking at the values counts of categories in the training dataset, a handful of categories make up for the majority of crimes. With the presence of such a long tail, I knew the accuracy of my model would likely be low.



Three other simple plots that I found interesting were values counts for time of day bucket (there were drastically fewer crimes taking place in buckets 1 and 8 which represenet 2-7:59am), day of month (the 1st is by far the most popular date for crime occurance), and month of year (October has most and December has fewest).

A final plot that helped introduce me to the dataset was a value count by zip to see where the most crime-heavy areas of SF are and where my neighborhood stands. The Tenderloin and Soma are the most dangerous by quite some margin and my neighborhood comes in 7th. The fact that the Tenderloin is dangerous doesn't surprise me but I was surprised to find out that I might need to watch my back a bit more in my neighborhood. It was also telling to group by zip and category to get a feel for which crimes were popular where.



How you chose which features to use in your analysis

I used a random forest sort my features by importance. My top 5 features were: day_of_month, time_of_day_bucket, month_of_year, year, DayOfWeekFriday. This helped validate the dummy columns I added and the time_of_day_bucket column I created. Some census features in my top-25 include: CensusRacePercentOtherOnly, CensusHispanicPercentMexican, CensusAgePercent15To19, CensusAge15To17, and CensusAgeUnder18. The fact that a handful of census features were more important than features Kaggle provided also helped validate looking to external sources for feature development.

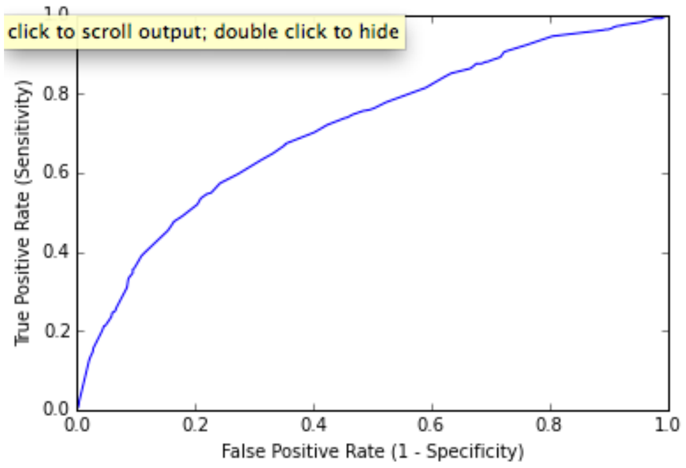
Details of your modeling process, including how you selected your models and validated them

For the first half of my project, I faced a multiclass classification problem. The three models that I chose to explore were Knn, Decision Tree, and Naïve Bayes. I used sklearn's gridsearch to tune paramenters for each of these models and played with the number of features given to the model. If I predicted every crime to be Larceny/Theft in my training set, I would've obtained an accuracy of 19.92% of the time so I took this to be a baseline effectiveness measure for my model. If it can predict better than 19.92% accuracy, my model is effective at some level.

For Knn, I used my top 5 features and a k value of 27 to obtain an accuracy of 20.02%. This model took the longest to fit and during development, I decided to take a random 10,000 subsample in order to speed the process up. Working with a smaller dataset allowed me to tune the number and set of features I passed to train my final model in a reasonable amount of time. In order to improve my accuracy using Knn, I decided to normalize my features using z score. After normalizing my features, the accuracy of my model decreased by -3.1%. I never figured out why my accuracy decreased but my initial thought is that maybe I need to include a different set of features after normalizing. For my Decision Tree, I used my top 10 features and a max depth of 2 to obtain an accuracy of 23.45%. Naïve Bayes yielded a null accuracy rate lower than my null accuracy rate of 19.93%, giving me an accuracy of 17% when using my top 18 features.

The second half of my project was a binary classification problem. The two possible results were -1 (bad for selling smoothies) and 1 (good for selling smoothies. The crimes that I defined as helpful to smoothie selling were: arson, drunkenness, drugs/narcotic, gambling, and loitering. And the crimes that I defined as harmful to smoothie selling were: vehicle theft, recovered vehicle, stolen property, robbery, and driving under the influence. The web app that this model powers displays a map of San Francisco with each zip code highlighted as either red (bad to sell smoothies) or green (good to sell smoothies). The features my app feeds to my model are: day_of_month, time_of_day_bucket, month_of_year, year, DayOfWeek_Friday, DayOfWeek_Wednesday, DayOfWeek_Tuesday, DayOfWeek_Thursday, DayOfWeek_Monday, DayOfWeek_Saturday, DayOfWeek_Sunday, is_weekend, and zip. If I chose -1 for all of my samples, my accuracy would've been 58.65% so I defined that as my baseline effectiveness measure.

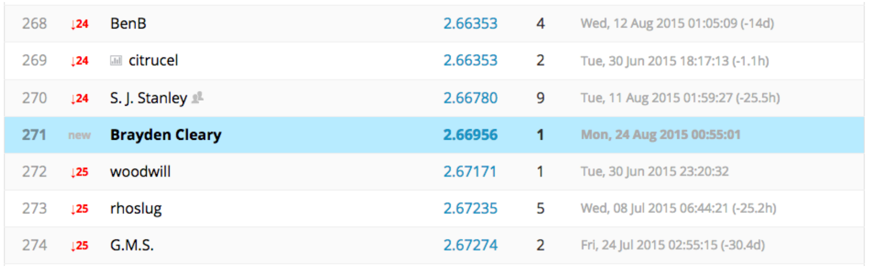
After experimenting with the same models as the first half of my project and adding a logistic regression, svm, and random forest, I settled on using a decision tree to power my smoothie selling predictor. The model had an overall accuracy of 72.6 %, with a specificity of 74.25%, a sensitivity of 71.91%, and an AUC of 78.65%.



Your challenges and successes

My accuracies for both problems were low. For the multiclass classification problem, I'm only slightly better off than picking Larceny/Theft and for the binary classification problem, I'm far away from maximizing my smoothie selling effectiveness. Working with a large-ish (850,000) rows and 160 columns presented some processing challenges, especially when working with computationally intense models like Knn. To overcome this challenge, I sub-sampled a random set of 10,000 rows and developed my model using that data. Two of the most useful features Kaggle provided in the train dataset weren't present in the test set (description and result). These two columns provided a text description of the crime and a one word description of the result (eg 'Arrested'). Although non of my Kaggle competitors could use those features either, it was still a mental challenge being presented with such awesome features and then not being able to train on them.

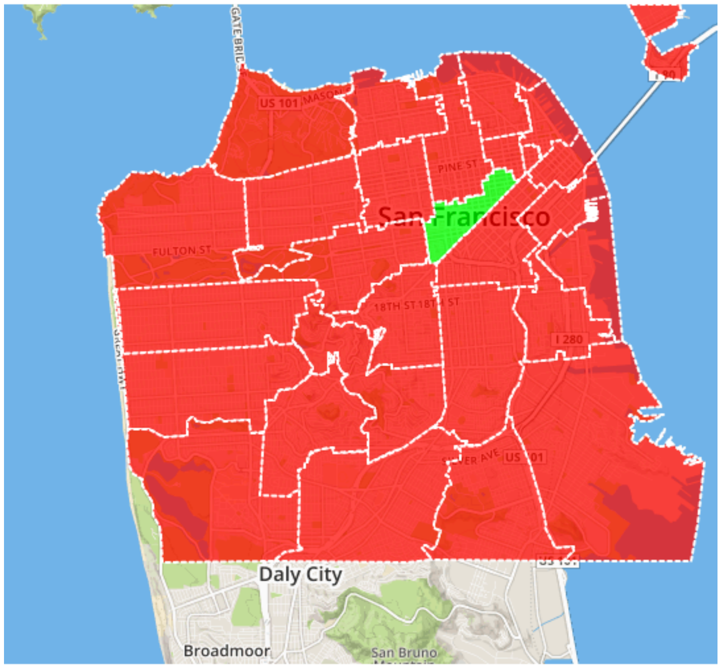
As far as successes, I'm 260th out of 420 in the Kaggle competition, which is not as horrible as I would expect after my first submission. The process of prepping the CSV for submission was a challenge in and of itself, so



268	↓24	BenB	2.66353	4	Wed, 12 Aug 2015 01:05:09 (-14d)
269	↓24	citrucel	2.66353	2	Tue, 30 Jun 2015 18:17:13 (-1.1h)
270	↓24	S. J. Stanley	2.66780	9	Tue, 11 Aug 2015 01:59:27 (-25.5h)
271	new	**Brayden Cleary**	**2.66956**	**1**	**Mon, 24 Aug 2015 00:55:01**
272	↓25	woodwill	2.67171	1	Tue, 30 Jun 2015 23:20:32
273	↓25	rhoslug	2.67235	5	Wed, 08 Jul 2015 06:44:21 (-25.2h)
274	↓25	G.M.S.	2.67274	2	Fri, 24 Jul 2015 02:55:15 (-30.4d)

I'm excited I overcame that barrier and that future submissions will be much easier. Pulling in zip/census data from ask geo was another success I can hang my hat on. There's an abundance of interesting data to explore beyond the problems that I chose to explore, so I look forward to doing that soon. And finally, using EC2 to explore my data and develop my models was significantly faster than using my personal computer and saved me a bunch of time. Investing time in environment/infrastructure setup isn't always thrilling but this one proved useful now and will prove as useful moving forward.

Conclusions and key learnings



Red Wagon Smoothies is just getting started. We're a young company with a bright future and we'd like to power our selling decisions based on a variety of data. This is a step in the right direction and will serve as one indicator for where we should/shouldn't sell smoothies. As chief data scientist, I know that smoothie consumption data is tough to come across. Gathering Twitter, Instagram, and Foursquare data on smoothie mentions or places where people might consume smoothies are only estimators for smoothie consumption, just like crimes are. To be fair, crimes are a bit more outlandish than social media data but the point remains, our selling predictions will have some level of uncertainty no matter the data-source until we start collecting our own data.

In using smoothiecrime.herokuapp.com, the Tenderloin is generally the best place for me to sell smoothies. This conclusion is slightly worrisome due to the dangerous environment in which I'd be selling smoothies, but I'm sure it would also keep things exciting. In the future, I'd like to display the results of different models, display a list of parks that would be ideal to sell at, and add in some weather data to make my predictions more dynamic.