NYCe Taxi!

INFO 290T SPRING 2015
FINAL REPORT

May 14, 2015

Anand Rajagopal
Anubhav Gupta
Sindhuja Jeyabal

Introduction

This overall objective of this project was to develop an algorithm that can be used to analyze the pattern in which cabs operate in a city to try and predict the tip can be expected per fare.
The idea for this project was based on the assumption that the trip history of cab drivers when combined with demographic data about the model the tipping habits of people at different location, different points of times and for different kinds of trips. Using different machine learning techniques, we wanted to develop a system that would assign a given trip into specific classes designating tip amounts.

We started this work by acquiring data about taxi rides in New York City from 2010 to 2013. This information was released by the NYC Taxi and Limousine Commission and had information covering all the fares and trips made over this period. Using this and associating local demographic data from the US Census, we were able predict tips with a good accuracy. While this problem has been attempted before, we believe that by combining the demographic data our algorithm is little more city agnostic.

At a high level, we used both classification and regression techniques to tackle this problem. We developed classes to aggregate the tip amounts by both their actual denomination and also a percentage of the fare. This was after extensive phases of data cleaning, data exploration and feature engineering where we merged different data sources and visualized the correlation between key variables of our data to identify the most effective features for our algorithm.

We will next describe in more detail the nature of the dataset and our method for dividing the data meaningfully to support our algorithms. After this we will introduce the different paths we followed while exploring this dataset to reach our current hypothesis and the methods we used to analyze this hypothesis. Following which, we go into more detail about the predictive algorithms and the results we had for each of the methods. We will divide the section into different parts based on our individual contributions.

Primary Dataset

Source

The dataset was obtained from the New York City Taxi and Limousine Commission(NYCT&L). It covers four years of taxi operations in New York City and includes 697,622,444 trips.
Source of the dataset: http://publish.illinois.edu/dbwork/open-data/

Size of the data

The data set is consists of four folders each for an year from 2010 to 2013. For the purpose of the project we considered data for the year 2013. The data consists of two parts - Trip data containing trip information like pickup_datetime, trip distance, trip time etc. and Fare data containing fare information like fare amount, tip amount etc. The data from each year consisted of 12 csv files containing data for each month.
trip data size - 116 GB
fare data size - 75 GB
Each month for the year 2013 contained approximately 14 million lines of data.

Nature of the data

Unique fields: These were fields that were unique to a taxi or a driver.
Medallion : the plate or the registration number of the vehicle used for the trip
Hack_Liceense : the license id of the driver who was driving the cab during that trip
Vender_id : The identification number of the vender who owns the vehicle

Trip fields: The different fields that were in the trip data part of the dataset.
Pickup_datetime: the date and time of pickup for a particular trip
Dropoff_datetime: the date and time of dropoff for a particular trip
trip_time: the total time of the trip in seconds
trip_distance: the distance of the trip in miles
pickup_latiude: the latitude of the pickup location
pickup_longitude: the longitude of the pickup location
dropoff_latiude: the latitude of the dropoff location
dropoff_longitude: the longitude of the dropoff location
passenger count: the total number of passengers for a particular trip
store_and_forward_flag

Fare fields: The different fields that were in the fare data part of the dataset.
Fare_amount: the total amount of fare for a particular trip
tip_amount: the tip amount for a particular trip
tolls_amount: The tolls amount that were paid for a particular trip
total_amount: the total amount for a particular trip that consists of fare, tip, tolls and tax
mta_tax: the tax that Metropolitan Transportation Authority collects for a particular trip
payment_type: this is the payment type ie card, cash or others.

As we can see the data set that we used was very large so we had to choose our computational power wisely so that we could perform the necessary processing in a reasonable amount of time. We also had to merge the two datasets together into a single dataset which we did using a combination of three keys medallion, hack_license and pickup_datetime as a primary key that identified the same trip in both the data sets.

Secondary Dataset

We had latitude and longitude information in our trip data. We wanted to get information about the characteristics of the place like affluence, cost of living etc. Our initial plan was to split our latitude and longitude into the different boroughs. But the boroughs extend over a large area, so we instead decided to go with zip codes . We found open data which had zip codes and the latitude-longitude pair for the center of that zip code. We also scraped a website to obtain demographic information of a zip code. We joined these datasets and added in the attributes on demographic information to our merged trip-fare dataset.

Attributes used:
Pickup_Zipcode
Dropoff_Zipcode
Pickup_Cost_of_Living
Dropoff_Cost_of_Living
Pickup_Median_Household_Income: in USD
Dropoff_Median_Household_Income: in USD
Pickup_Population_Density: in thousands per square mile
Dropoff_Population_Density

The cost of living is an index handed out by the Census which has an overall national average of 93 while the NYC average is about 120.

Introduction to approaches

Like mentioned earlier, the dataset consists of trip and fare data. Our initial ideas were to predict peak hours in the city and predict traffic at different times as a function of the average speed. After further analysis and discussion we concluded these problem statements are good visualization projects and we looked at more machine learning based problem statement.

Wait time Prediction

Our next approach was to predict the wait time - the time a taxi driver has to wait in between trips. The dataset does not explicitly have the wait times. We would have had to derive the wait times between trips from the pickup and dropoff timings and the passenger count information. We performed these data operations and figured out the dataset had 98% zero wait time and the rest of it was about an average of just one minute. Since the data was very skewed and since there was not much of a practical significance of the predicting wait time problem we decided to try a explore the dataset with a slightly different angle. We found that our previous analysis could be leveraged to try and solve another challenging problem, that of tip prediction.

Tip Amount Prediction

The problem statement - Predict the tip amount a driver is likely to receive.

Since we found the data to be skewed for wait times, we wanted to confirm that was not the case for tip amounts. We found 50% of the data had zero tip. So, we were safe here. Our generic approach was to visualize the data to understand it better. This would give us information about outliers and incorrect (absurd values) that need to be removed. Apart from data cleaning, the other important affordance of the visualizations is the correlation we can derive between the different attributes and the tip amount.

Once the data was ready, we decided follow two major approaches - Classification and Regression. And since the data had 50% zero tips, we wanted to test the strength of the models on the data with zero tips. We also wanted to see how our models fare in predicting tips as a percentage of the fare data.

Individual Contributions

The tasks involved finding the right problem statement and the dataset. Once that was done, the major tasks included

Data Engineering - Extraction, Infrastructure, Cleaning
Data Visualization - Plotting the attributes against tip data and analyzing the general distribution of the tip
Feature Engineering - Engineer features based on the trip, fare and demographic data
Data Modeling - Classification and Regression models on the different data groups (The total vs the Non-zero tip data)

Work Division
Anand took care of most of feature engineering and the linear regression model while Anubhav and Sindhuja split the Data Engineering, Visualization and Data Modeling. We roughly followed the order above but for the sake of grouping the work in terms of our individual contributions, the tasks might be detailed in a different order.

Feature Engineering
This part of the process involved two major phases - merging the primary and secondary datasets and then engineering composite features and evaluating its impact on the overall efficiency of the algorithm.

In order to merge the secondary data sources, we initially started by looking resources that gave mapping between latitude and longitude information for the different zipcodes in New York City. After finding a dataset which had information about zip codes for the entire country, we filtered that by state and tried to map the different pickup and dropoff latitudes and longitudes to the zipcodes. This led to the discovery of bad data where the latitude and longitude corresponded to areas outside the geographic boundaries of NYC. The zipcodes data was represented in such a way that each zipcode had one

latitude and longitude assigned to it, presumable something pointing towards the center of that area.

in order to match the latitude-longitudes, we developed an algorithm which picked a zipcode based on the zipcode that had the lowest sum of the differences between the corresponding latitudes and longitudes between the individual zipcode and the pickup location. Using this function, we were able to assign a zip code to each pickup and drop off location.

As a composite feature, we wanted to see if the borough to which the zipcode belonged would be useful as a feature. In order to this, we found another dataset that had zipcodes of NYC segregated by boroughs and tried to combine that with the previously assigned zipcodes. However this proved harder than expected since there was a difference in the number of zipcodes present in each dataset. After some cleaning and data manipulation we were able to finalize a dataset and create a new feature. However, we finally decide to exclude this feature from our analysis as we realized that this was strongly tied to NYC and would reduce the city agnostic nature of our algorithm.

In the next stage of this process, we tried to combine data from the US Census to include demographic information about the pickup and dropoff areas. To do this, we matched it by the previously assigned zipcodes and included relevant information as new features. Again this posed challenges because there was no perfect intersection of the zipcodes. After merging this data, we included population density, cost of living index and gender diversity as features. There were other features like unemployment and cost of housing that might have been useful features but there was a lot of missing data for these fields so we decided to exclude them.

The second phase of this analysis was to try and evaluate the different features with the prediction models and trying develop composite features from the existing features. We tried breaking down some of our existing features into smaller sub-features and checking if they have an impact on the overall efficiency. We broke the date-time fields into time of day, month of year, peak/non-peak time and used them as separate features.

Ideally we would have liked to have more time in this phase to also try more combinations of features and their resultant performance but due to time constraint we were forced to restrict ourselves to the more obvious features.

Data Infrastructure

As we mentioned in the section above that our dataset was pretty large even after restricting ourselves to just the year 2013. We had almost 11GB of compressed data that we had to process. We decided to set up a server with the following configuration:
Cloud Service: IBM's Softlayer
OS - Linux 14.04
Computing power - 8 dual core 2.0GHz CPUs(16 cores in total)
Memory - 16 GB (we tried to get the maximum memory that we could so that we could hold the data frames in memory while processing them)
Disk Space - 100 GB (uncompressed data was about 36GB but so we took a 100 GB primary disk)

Data Extraction

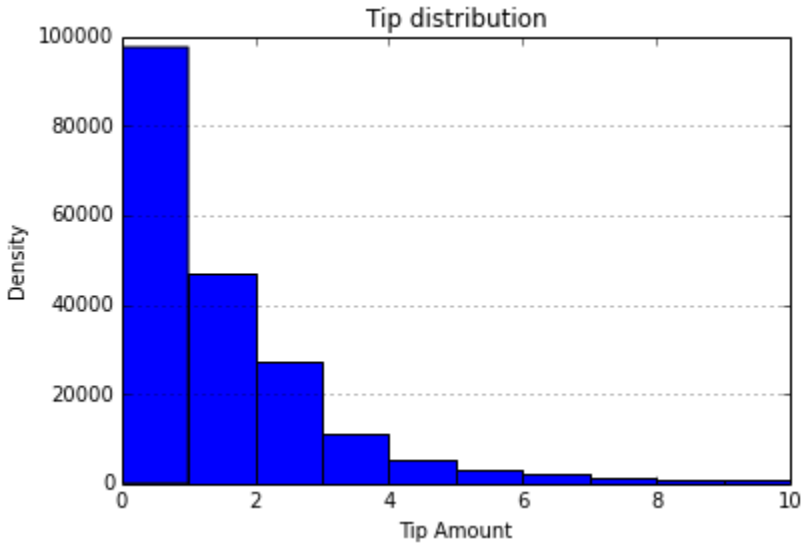
Merging the two data sets: The first step was to merge the trip_data and fare_data on a primary key. Since there was not one key that could identify a trip in both the datasets, we created a primary key by combining Medallion, Hacker_license and pickup_datetime. The merge was performed using bash scripting and we parallelized it using 16 processes(we had 16 cores). We now had the the merged csv files containing the trip and fare attributes but there were still 12 files one for each month.
Slicing the month files and merging them - If we combined all the files, we would roughly have around 150 million records which would take a long time to process so we decided to take 1 million lines from each of the month so that we had a final data file with around 12 million lines of data. So we first shuffled the files using a bash command called shuf and then took out the top 1 million lines from each file and merged them to make a final data file. We then performed a shuffle on this final file so that that data from all the months was properly mixed together.

We installed ipython on the server and hosted the ipython notebook on a particular port which all of us could access through https://ip_addess:port_number on a web browser.

Visualization

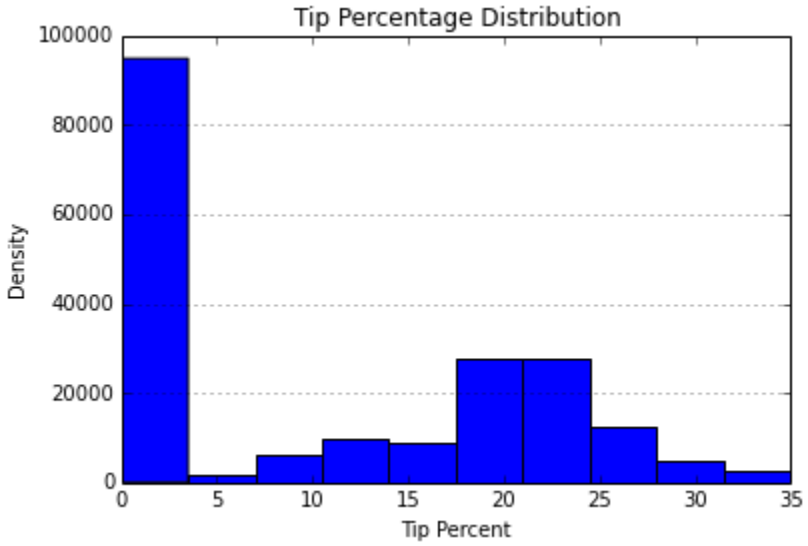
This part of the work is shared with Anubhav. We used matplotlib to plot the tip density and the correlation of different attributes with the tip amount. The aim of this exercise was to understand the data better

We plotted the tip_amount density to identify bins for the tip amount that we could use for our classification model.



We found that a large chunk of data roughly around 47% comprised of ZERO tip so we decided to have Tip == 0 as one of the bins. we also decided to include a classification model with the non zero tip data. Based on the bar graph we made tip bins from 0 - 1, 1 -2, 2 -3 etc.

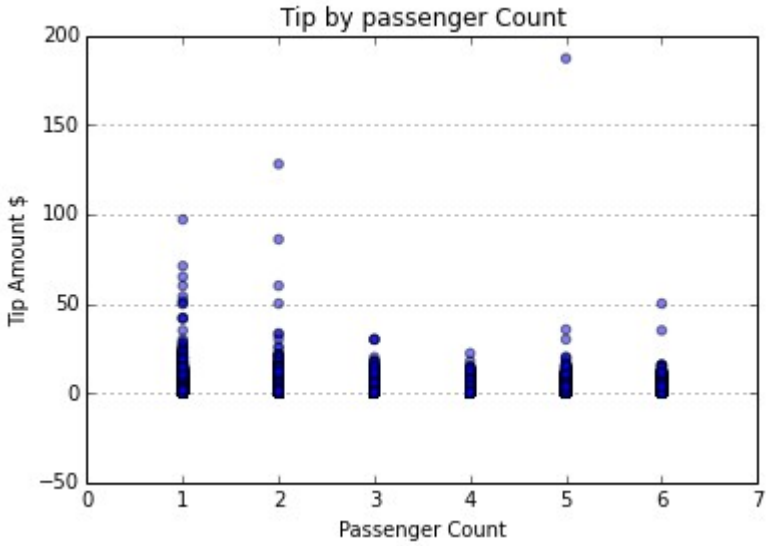
We also plotted the tip percent(as a %age of fare) density function to identify the bins for tip percent for our classification model.



We found that there was a ZERO tip percent bin (because we had a ZERO tip amount concentration) and also a concentration from 17% tp 27%.

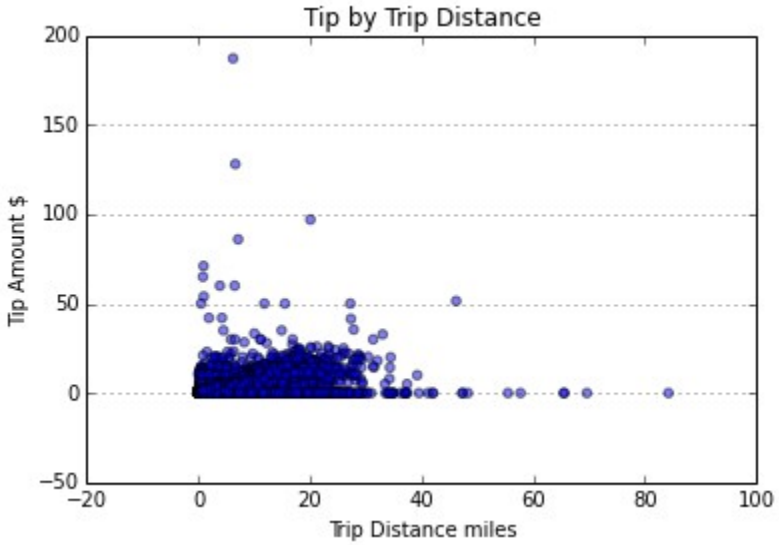
Co relations:

Tip vs Passenger count:



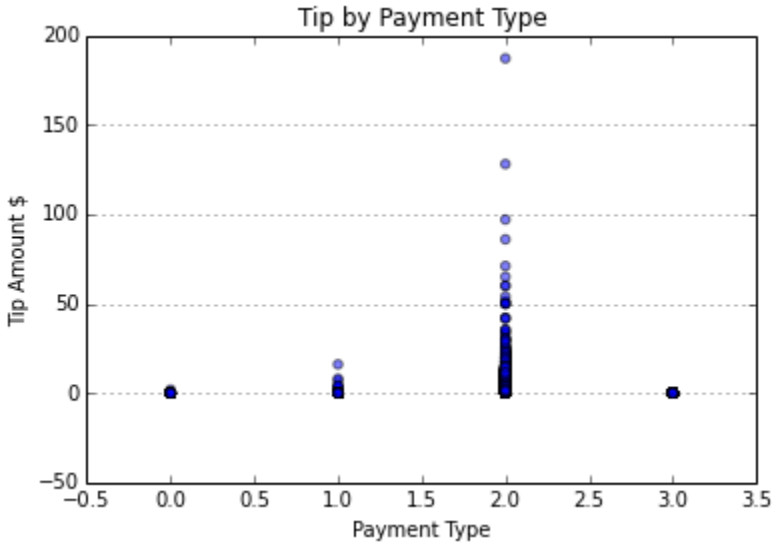
We found that there was slightly higher tip amount for a single passenger however that could be because the number of trips for a single passenger were higher.

Tip vs Trip Distance



We could not find a significant relationship between trip distance and tip though we expected that a higher trip distance would result in a higher tip.

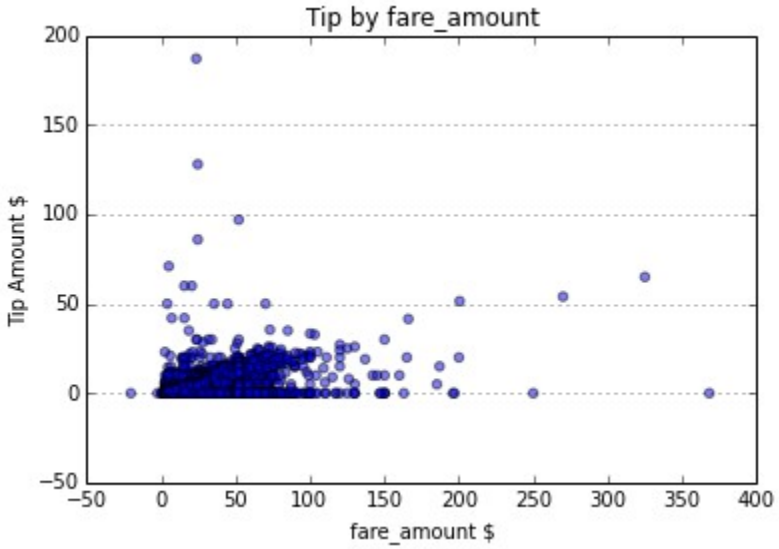
Tip vs Payment Type



{'NOC': 3, 'CSH': 1, 'CRD': 2, 'DIS': 0}

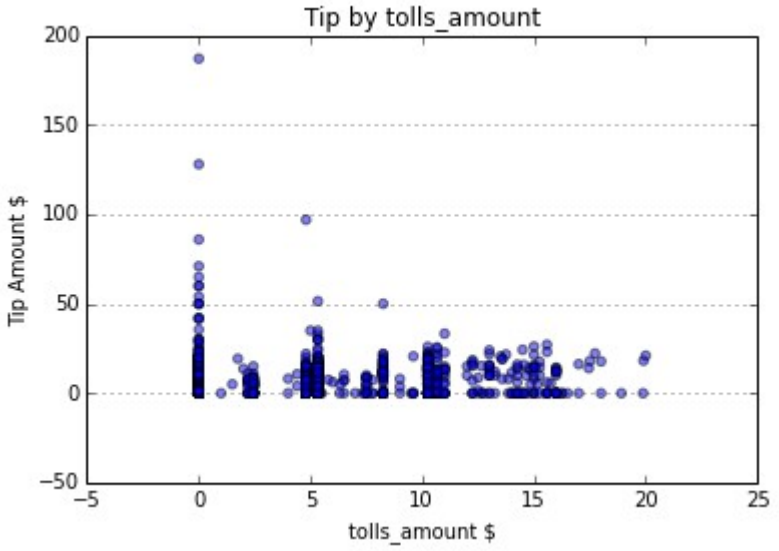
We found that there was a strong correlation between the payment type and the tip amount. The tip amount was generally higher when the payment was made through cash.

Tip vs Fare Amount



As expected we found a correlation between fare amount and the tip because the tip depends on the fare of the trip or a percentage of the fare amount.

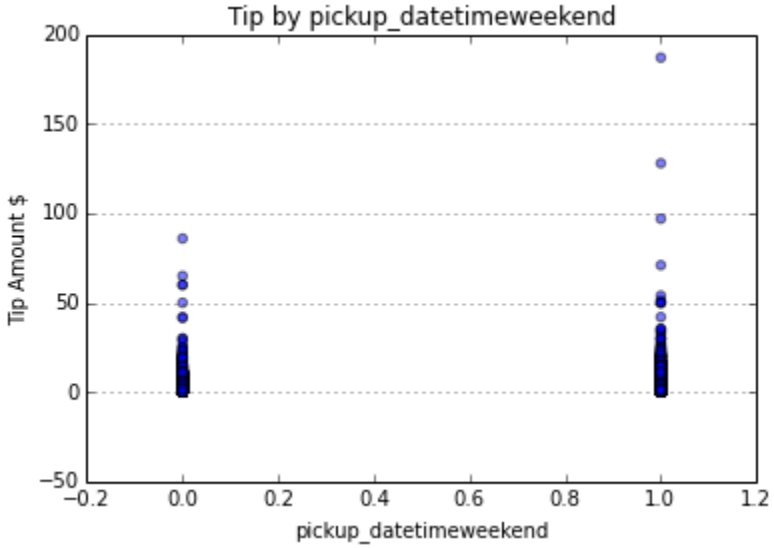
Tip vs Tolls_amount



We found that there was a higher tip amount for a trip that did not have tolls in comparison to one that had tolls.

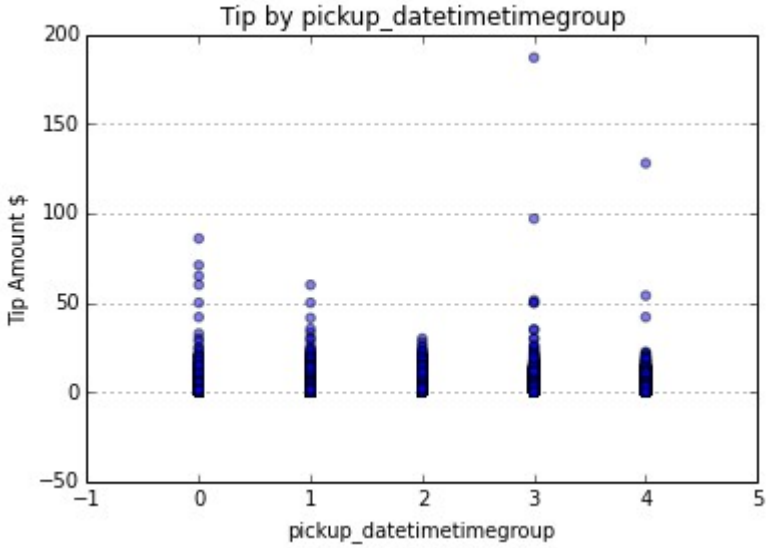
We also tried to plot a graph between the different derived features that we had from feature engineering however none of them had a clear correlation with the tip.

Tip vs pickup_datetimeweekend



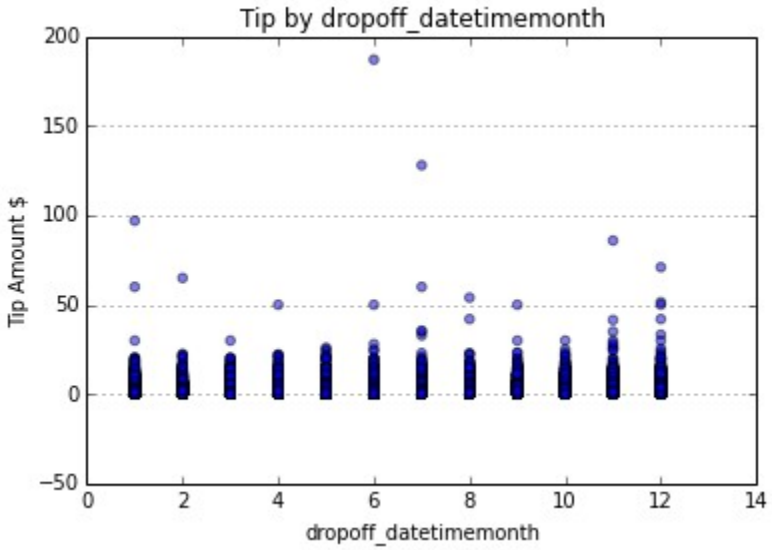
{'weekend': 0, 'weekday': 1}

Tip vs pickup_datetimegroup

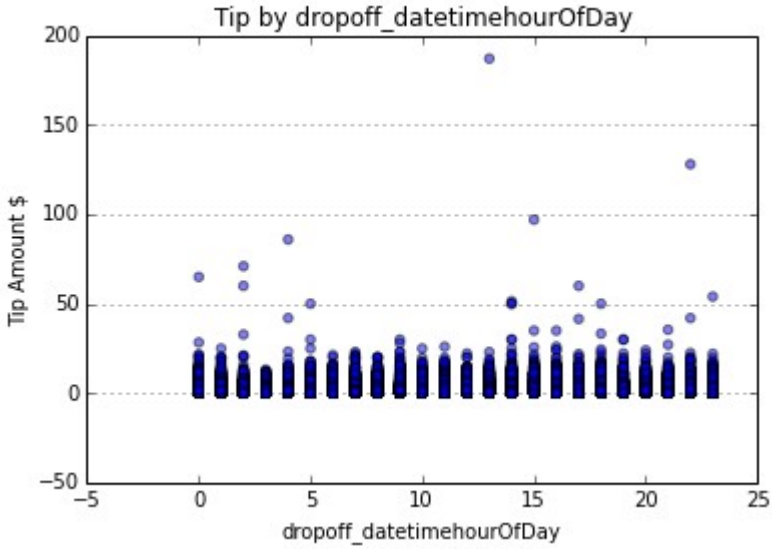


{'early': 0, 'night': 4, 'evening': 1, 'afternoon': 3, 'morning': 2}

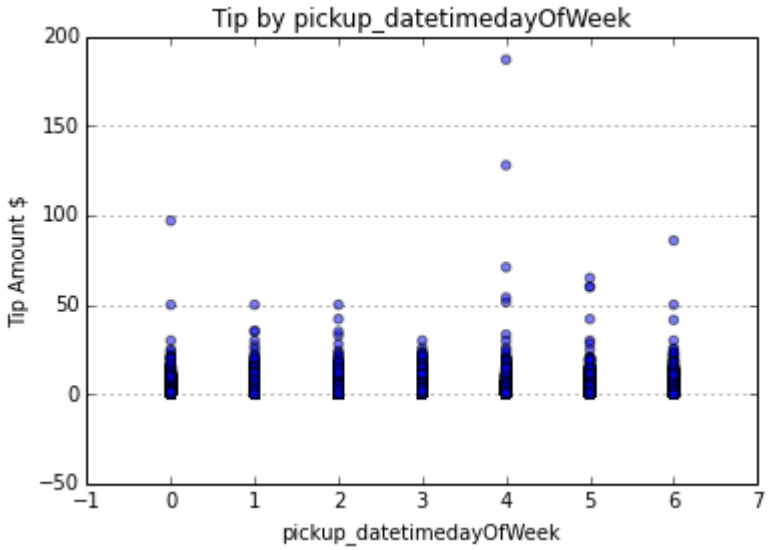
Tip vs dropofftimemonth



Tip vs dropoff_datetimehourofDay



Tip vs pickup_datetimedayofWeek



Data Cleaning

Based on the insights from the visualization and listing of the rows in the dataframe, we figured out missing values, outliers and incorrect values of the attributes.

For data cleaning, we filled in outliers, missing and incorrect values with NA. The following were the conditions we applied for filling with NA.

Passenger counts:

Records which had counts greater than 6. Some records had passenger counts 225!

Trip Distance:

Trip distance greater than 1000 miles and distance equal to zero miles.

Trip Time:

Trip time equal to zero seconds

Fare Amount:

Fare Amount equal to zero

Payment Type:

The records with payment type as ‘UNK’. This was a categorical variable.

We dropped rows containing N values using the dataframe dropna() function in pandas.

Interesting debugging moment:
Dropping the NA values resulted in more than half the rows getting dropped and our models were not performing well. We spent quite some time in analyzing if there was something unusual with the above mentioned fields. But the visualizations did not tell us anything which lead us to doubt the attributes that were cleaned. Upon further debugging, we found there was a 'store_and_forward_flag' which was NA for most rows. And all these were getting dropped on dropna(). We removed this attribute from the original dataset to fix the issue.

Optimization:
Our final set of features were all numerical except for payment type which was a categorical feature. We were using scikit dictvectorizer to convert to numeric features. This was slowing down our development. So, we enumerated the payment_type variable and modified the original dataset and converted it into a numeric variable. Our subsequent code runs were significantly faster.

Tip Bin Classification
For running the classification algorithms, the tip amount had to be classified into different classes. The tip distribution visualization showed that 50% tips were zero and most of the tips ranged from $1 to $20 with a few going higher.

Initially, we tried clustering the tip amounts using k-means clustering and clustered the tip amounts into 5 different classes. But the scikit documentation specified it is not a good practice to use k-means for one dimensional data. So, we adopted the Kernel Density Estimation (KDE) to find gaps in the tip distribution. But both these algorithms, gave centroid clusters that could not be translated meaningfully to the real-world.

So, we finally resorted to manually binning the tip amounts into meaningful bins. We assigned Zero tips to a zero tip class (0) and the other classes were $1- $2, $2-$3, etc and $5-$7.5, $7.5-$10, > $10.

Similarly, we also classified tip percentages (of fare amounts) into different classes.

Models & Algorithms

As mentioned earlier, because of zero tips occupying 50% of the dataset, we worked with multiple dataset to test our models

FEATURE SET

We tried a number of features both location based and time based. But our models performed based (accuracy measure) with the following set of features.

- Fare amount
- Tolls amount
- Payment type - only when zero tip data was present
- Cost of Living Index

For all the classification-models we calculated the baseline accuracy by predicting majority class. And for the regression models, we computed the baseline accuracy using the Mean Absolute Error metric. This measure was computed by predicting the mean for the tip and finding the absolute deviation from mean for each record and computing the mean of all such deviations.

CLASSIFICATION

- Tip/No-Tip Classification
- Tip Class with all the data
- Tip Class with non-zero data
- Tip Percentage class with all the data
- Tip Percentage class with non-zero data

REGRESSION

- Tip amount prediction

We ran the following classification algorithms

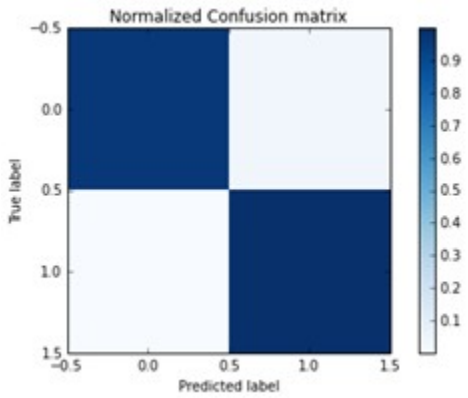
- SVM rbf classification (C = 0.1, gamma = 0.01)
- Decision Tree Classifier
- Random Forest Classifier
- Adaboost Classifier (Decision tree Classifier)

And for regression, we did

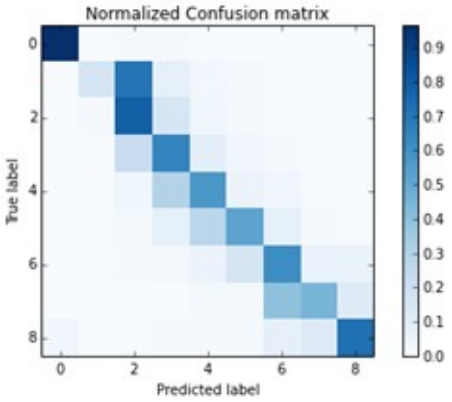
- Linear Regression
- SVM Regression
- Lasso Regression

We plotted confusion matrices for the different models to see where our model was going wrong.
Here are some examples of confusion matrices we had. Following the confusion matrices, we have detailed results on the accuracy scores.

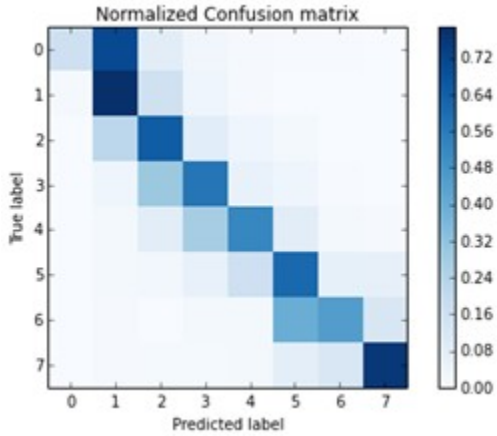
Tip / No Tip Classification



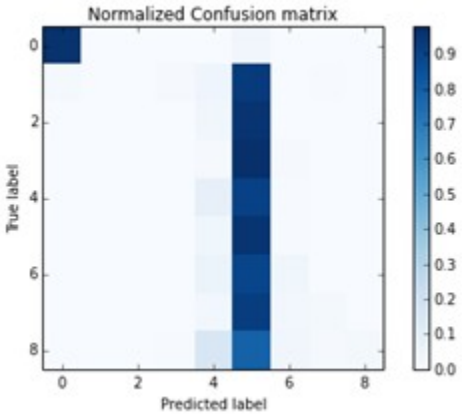
Tip Class for non zero tip data



Tip Class for all the data



Tip Percent Class for all the data



Results

Classification

	Baseline	SVM	Decision Tree	Random Forest	Adaboost
Tip - No Tip	52.33	98.516	98.249	98.259	98.299
Tip Class w/	47.66	81.253	81.565	81.593	81.208
Tip Class w/o	45.07	67.98	68.002	68.002	66.72
Tip % w/	47.66	68.10	68.08	68.097	68.013
Tip % w/o	41.47	41.58	42.12	42.14	41.97

We got a pretty high accuracy with Tip - No Tip and Tip Class w/ Zero values with 98% and 81.5% respectively. The accuracy was considerably above the baseline - 22% for Tip class w/o zero values and Tip% w/ zero values. However we were not able to improve much over the baseline accuracy for the tip% w/o zero values.

Regression

	Baseline(Mean Abs Error)	Linear	SVM	Lasso
Tip	1.38	0.75	0.79	1.254

We got a considerable gain over the baseline accuracy for Linear and SVM regression.

Conclusions

One major learning from this process was that simple features contribute more to the predictions that complicated or composite features. In this case, the tip was primarily a function of three main features - the fare amount, the payment type and whether there was a toll.Other features that had a lesser impact were the cost of living index for a city and the pickup zip code. While these didn't have a very strong effect, we believe that these add weightage in making this algorithm more city independent.

There were other things about the data which we learned in this process. To begin with, regression yielded lower accuracy than classification. This is not a definitive conclusion as we didn't get as much time to try and improve the accuracy of these algorithms. They held a steep learning curve and were also harder to improve because of the inherent skew in the data.

Coming to classification, it was easier to predict if there would be a tip or no tip and this analysis had an extremely high level of accuracy. However when it came to predicting the actual tip amount, it was easier to predict classes made by binning the data on their actual amounts rather than as a percentage of the amount. It was also easier to predict the tip amount when we included the 0$ tip data as part of the dataset. This was again probably because of the skew of the data where there are a large number of records with a 0$ tip, approximately 48% of the data. Therefore, removing the data provides data with a lot more variation in the classes making it harder to predict.

Real world implications

Apart from meeting our main objective of predicting tips, a useful tool to have when deciding how to strategize your trips as a cab driver and analyzing what pattern of driving makes the most sense economically, we were able to draw many conclusion from the data which yield insight into different activities in real life scenarios.

To begin with, the skew in the data is a result of taxi drivers not reporting tips when they receive money via cash transactions. As a result, when the payment type is card, the probability of a tip becomes drastically higher. This data is reported by different vendors and only data from one vendor has tips for cash transactions. Another finding was that the tips was higher for card

and clusterer around 20%. A little analysis showed that this was the default tip on most card machines and a small item that most customers tend to overlook.

Another interesting fact we discovered from our analysis was that taking a route that had tolls seemed to have a negative impact on the tip. This is probably because customers have to pay tolls while taking a toll path and this apparently makes them less likely to pay an additional tip.
The cost of living index of a pickup area seemed to have a little impact whereas the drop area had little effect.

Future Work

Tips only form one part of the overall income of a cab driver. To understand this ecosystem in a better manner, it would be helpful to identify the different parts that contribute to the overall income of the cab driver. Identifying if longer trips are actually more economic versus trips that yield multiple trips within an area of high of cost of living index. Problems like this pose interesting challenges that can explored further in this dataset.