NYC Taxi Tip Predictor:
Machine Learning for Fare-Based Gratuity

Brijesh Tiwari

Piyush Mishra

Mandar Angchekar

Introduction

The overarching goal of this project was to create an algorithm that can be used to examine the patterns of cab operations in a city and attempt to forecast the tip that can be expected per fare.The premise behind this study was that the trip histories of cab drivers when paired with demographic information, might be used to predict people's propensities to tip differently depending on where they are, when they are
traveling, and what kind of journey they are on. We sought to create a system that would divide a given trip into several groups with varied tipping guidelines using various machine-learning approaches.

Data on taxi travel in New York City from the last 5-10 years was the foundation for this research. The NYC Taxi and Limousine Commission made this information public, and it included data on all the fares and rides taken during this time. We were able to anticipate tips with a high degree of accuracy using this information along with local demographic data from the US Census. Although this issue has been addressed earlier, we think that by merging the demographic data, our method is somewhat more city-independent.

We approached this issue at a high level using both classification and regression methods. We created classes to combine the tips according to both their actual dollar value and their proportionate share of the fare. This came about after lengthy stages of data cleansing, data exploration, and feature engineering in which we combined several data sources and visualized the association between important variables in our data to find the best features for our algorithm.

The dataset's characteristics and the process we used to divide the data in a way that would assist our algorithms will be covered in greater detail below. We will next go over the various routes we took when examining this dataset to arrive at our current hypothesis, as well as the techniques we employed to evaluate it. The findings we obtained using each of the methods are then discussed in more detail, as well as the predictive algorithms. In accordance with our individual contributions, we shall divide the section into various portions.

Primary Dataset Source

The New York City Taxi and Limousine Commission (NYC T&L) provided the dataset. It includes 697,622,444 rides over the course of four years of taxi operations in New York City.

Source of the dataset: https://www.nyc.gov/site/tlc/about/data-and-research.page

Size of the data

There are four folders in the data collection, one for each year from 2014 to 2018. For the project's purposes, we took 2014 data into consideration. The information is divided into two categories: trip data, which contains information about trips such as pickup_datetime, trip distance, trip time, etc., and fare data, which contains information about fares such as fare amount, tip amount, etc. Each year's data was divided into 12 csv files, one for each month.

data size - 80 GB
trip data size - 105 GB fare
In 2014, there were roughly 14 million lines of data per month.

Nature of the data

Unique fields: These were skills that were specific to a driver or taxi.
Hack_Liceense : The taxi driver's license number for that particular journey.
Vendor_id : The vehicle's owner's vendor's identifying number.
Medallion : The license plate or registration number of the travel vehicle

Trip fields: The many fields that made up the dataset's travel data section. Pickup_datetime: The time and date of the pickup for a certain journey.
trip_time: the amount of time spent traveling in seconds
Dropoff_datetime: the time and date of drop-off for a certain journey
pickup_latiude:the pickup location's latitude
trip_distance: the distance of the trip in miles
pickup_longitude: the pickup location's latitude dropoff_latiude: the drop-off location's latitude dropoff_longitude: the drop-off location's longitude
passenger count:the entire group of travelers on a specific trip
store_and_forward_flag

Fare fields: the various fields that made up the dataset's fare data.

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 you can see, the size of the data set we employed required us to choose our computational capacity carefully in order to complete the required processing in a fair amount of time. Additionally, we had to combine the two datasets into a single dataset. To do this, we used a combination of three primary keys—medallion, hack_license, and pickup_datetime—that distinguished the same trip across the two datasets.

Secondary Dataset

Our trip data included latitude and longitude information. We were interested in learning more about the area's qualities, such as its wealth and cost of life. Our initial thought was to divide up the boroughs according to latitude and longitude. The boroughs cover a lot of ground, therefore we choose to use zip codes instead. We discovered available data that had zip codes and the latitude-longitude pair for the zip code's center. In order to gather demographic data for a zip code, we also scraped a website. We combined these datasets, and to our merged trip-fare dataset, we included the properties on demographic data.

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 Census provides a measure of cost of living, with a national average of 93 and a NYC average of roughly 120.

Introduction to approaches

The dataset includes trip and fare data, as was previously described. Our original plans were to forecast the city's busiest moments and traffic flow at various times based on average speed. After more consideration and study, we came to the conclusion that these problem statements would make for effective visualization projects, and we looked at additional machine learning-based problem statements.

Wait time Prediction

The time a taxi driver must wait in between journeys was the focus of our next strategy. The wait times are implicit in the dataset. The pickup and dropoff times as well as the passenger count data would have been used to calculate the wait periods between trips. After doing these data operations, we discovered that 98% of the dataset had zero wait time and the remaining 2% had an average wait time of just one minute.

We chose to try to study the dataset from a slightly different viewpoint because the data was extremely skewed and because the predicting wait time problem did not have much of a practical value. We discovered that our earlier research may be used to try to address the difficult issue of tip prediction.

.

Tip Amount Prediction

The problem statement - Calculate the anticipated tip size for a driver.

We wanted to make sure that the data for tip amounts were not distorted because we discovered that the data for wait times was. 50% of the data had no tip, we discovered. This place was safe for us. In order to better understand the data, our general strategy was to show it. This would advise us of the necessity to eliminate outliers and inaccurate (absurd) values. The association we can draw between the various qualities and the tip amount is the visualizations' other significant benefit in addition to data cleaning.

As soon as the data was prepared, we chose to use two main strategies: classification and regression. We also wanted to test the model's robustness on the data with zero tips because the data had 50% zero tips. Additionally, we were interested in how well our models predicted tips as a proportion of the ticket data.

Individual Contributions

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

Data Visualization - Plotting the attributes against tip data and analyzing the general distribution of the tip.
Data Engineering - Extraction, Infrastructure, Cleaning

Data Modeling - Classification and Regression models on the different data groups (The total vs the Non-zero tip data)
Feature Engineering - Engineer features based on the trip, fare, and demographic data

Work Division

While Piyush and Mandar divided up the Data Engineering, Visualization, and Data Modeling, Brijesh handled the majority of feature engineering and the linear regression model. We essentially followed the above arrangement, but the tasks may be listed in a different order to help organize the work in terms of our individual contributions.

Feature Engineering

The two main stages of this approach were integrating the primary and secondary datasets, creating composite features, and assessing the effect on the algorithm's overall effectiveness.

We first started by looking at resources that provided mapping data between latitude and longitude information for the various zip codes in New York City in order to combine the secondary data sources. After locating a dataset with data on zip codes across the whole nation, we tried to map the various pickup and dropoff latitudes and longitudes to the zip codes by filtering the data by state. As a result, inaccurate data that had latitude and longitude values that were outside of the limits of New York City were found.

Each zip code had a single representation of the zipcode data.It has a latitude and longitude given to it, possibly something pointing in the direction of the region's center.

We created an algorithm that chose a zip code based on the zipcode that had the lowest sum of the discrepancies between the associated latitudes and longitudes between the individual zip code and the pickup location. This was done in order to match the latitude-longitudes. We were able to give each pickup and drop-off site has a zip code by using this function.

We wanted to determine whether the borough that the zip code belonged to would be beneficial as a feature for a composite feature. To do this, we looked for another dataset that had zip codes for New York City divided into its boroughs, and we attempted to integrate it with the previously allocated zip codes. Since each dataset contained a different amount of zip codes, this proved to be more difficult than anticipated. We were able to complete a dataset and develop a new feature after minor cleaning and data manipulation. However, after realizing that this attribute was closely related to New York City and would lessen the city agnostic nature of our algorithm, we ultimately decided to leave it out of our research.

In order to add demographic data about the pickup and drop-off zones, we attempted to merge data from the US Census in the following stage of this process. In order to accomplish this, we compared it to the previously given zip codes and added pertinent data as new characteristics. Once more, this presented difficulties because the zipcodes did not perfectly connect. We combined this data and added factors such as population density, cost of living index, and gender diversity. We decided to remove other features because there was a lot of missing data for variables like unemployment and housing cost that would have been important features.

This study's second phase aimed to assess numerous qualities using prediction models and make an effort to build composite features out of pre-existing features. To explore if it would have an impact on the overall effectiveness, we tried breaking up a few of our present features into smaller sub-sections. The date-time fields were divided into separate fields for the time of day, the month of the year, and peak and off-peak hours.

In this phase, we would have preferred to have more time to test different combinations of features and their performance, but owing to time constraints, we were obliged to focus only on the more obvious features.

Data Infrastructure

Even after limiting our dataset to the year 2014, as was described in the section above, it was still very substantial. We needed to process roughly 11GB of compressed data. We made the decision to configure the server as follows:
Cloud Service: Linux 14.04 from IBM's Softlayer OS
Computer power: 8 dual-core 2.0GHz CPUs (for a total of 16 cores).
RAM: 16 GB (we made every effort to obtain the most RAM possible in order to process the data frames while holding them in memory).
100 GB of disk space was chosen as the primary disk (uncompressed data only took up roughly 36 GB of this space).

Data Extraction

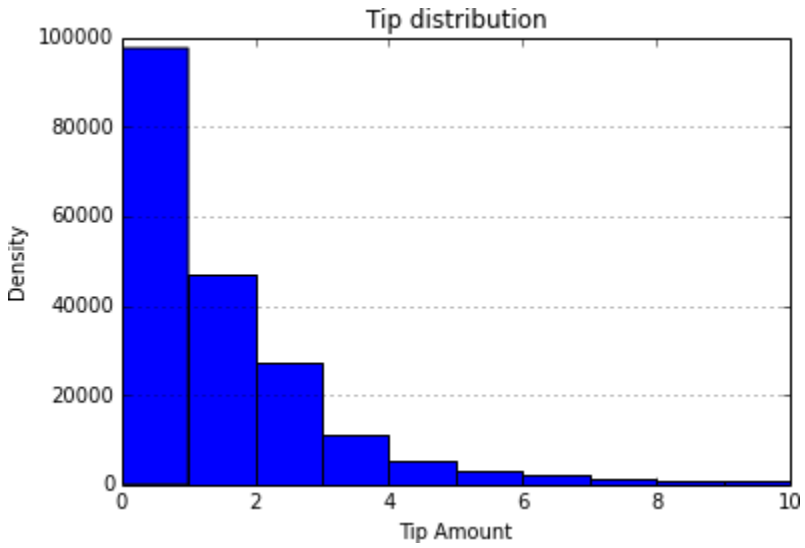
The two data sets are combined: The trip_data and fare_data were initially combined onto a main key. We combined Medallion, Hacker_license, and pickup_datetime to construct a main key because neither dataset had a single key that could uniquely identify a trip. Since we have 16 cores, we parallelized the merge utilizing 16 processes and bash scripting. The trip and fare attributes were now in merged csv files, but there were still 12 files—one for each month.

Combining all of the month files would result in around 150 million records, which would take a long time to process. Instead, we opted to extract 1 million lines from each month, resulting in a final data file with 12 million lines of data. In order to create the final data file, we first shuffled the files using the bash function shuf before removing the top 1 million lines from each file and merging them. After then, we shuffled this final file to ensure that the information from all the months was appropriately combined.

We set up Python on the server and hosted the Python Notebook on a specific port that any of us could view using the web address https://ip_address:port_number.

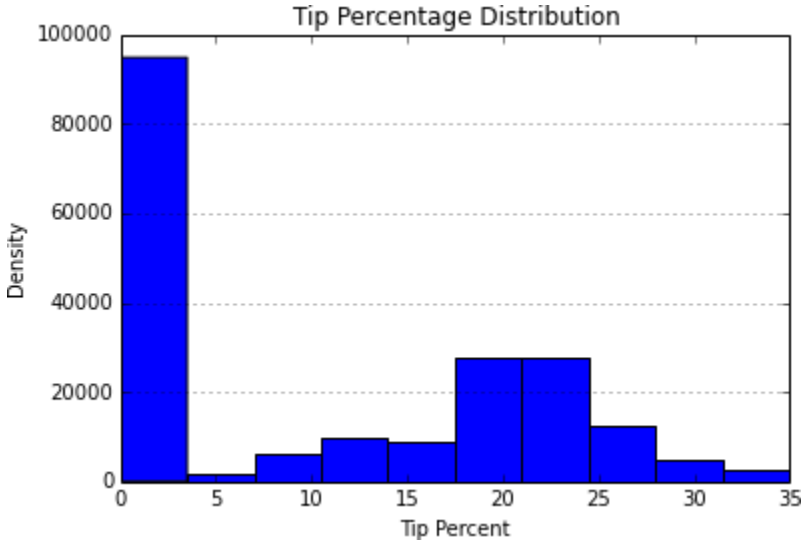
Visualization

With Brijesh, this portion of the labor is shared. We plotted the tip density and the correlation between various variables and tip amount using matplotlib. This activity was designed to help you better grasp the facts. In order to find tip amount bins for our classification model, we displayed the tip_amount density.



We discovered that a significant portion of the data, around 47%, consisted of ZERO tip, thus we chose to include Tip == 0 as one of the bins. The classification model will also be used with the non-zero tip data, we have decided. We created tip bins ranging from 0 to 1, 1 -2, 2 -3, etc. based on the bar graph.

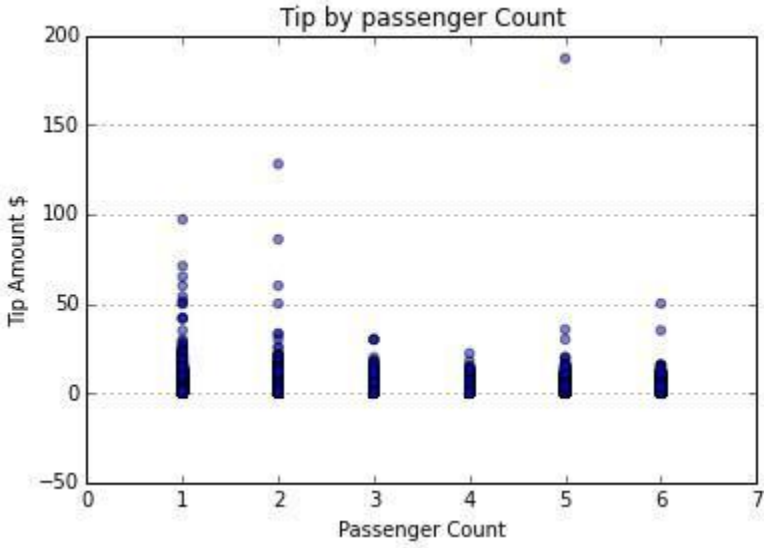
In order to get the tip percent bins for our classification model, we also plotted the tip percent (as a percentage of fare) density function.



Because we had a concentration of ZERO tips, we discovered that there was a ZERO tip percent bin as well as a concentration ranging from 17% to 27%.

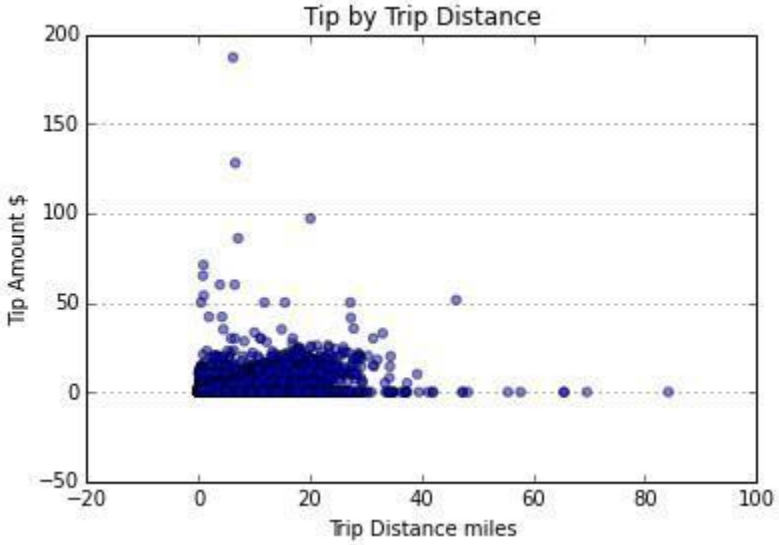
Co relations:

Tip vs Passenger count:



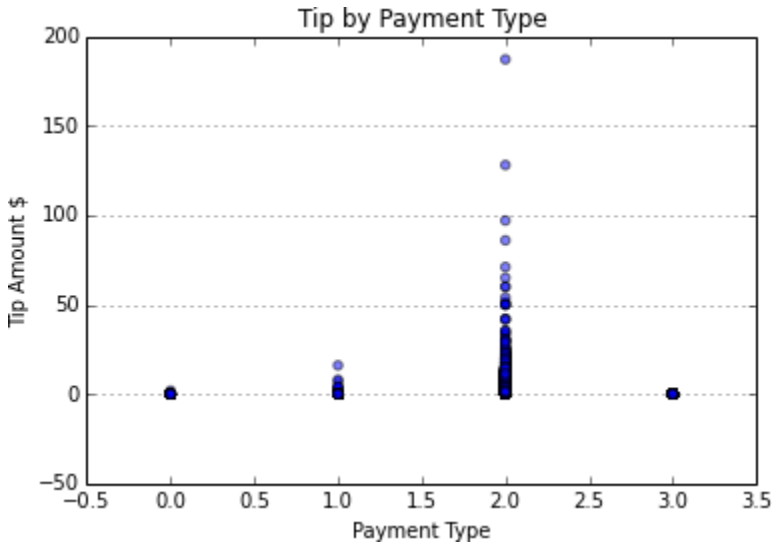
Although we discovered that single passengers left slightly larger tips, this could be due to the fact that they made more journeys.

Tip vs Trip Distance



Despite our expectation that a longer journey distance would translate into a larger tip, we were unable to detect a meaningful link between trip distance and tip.

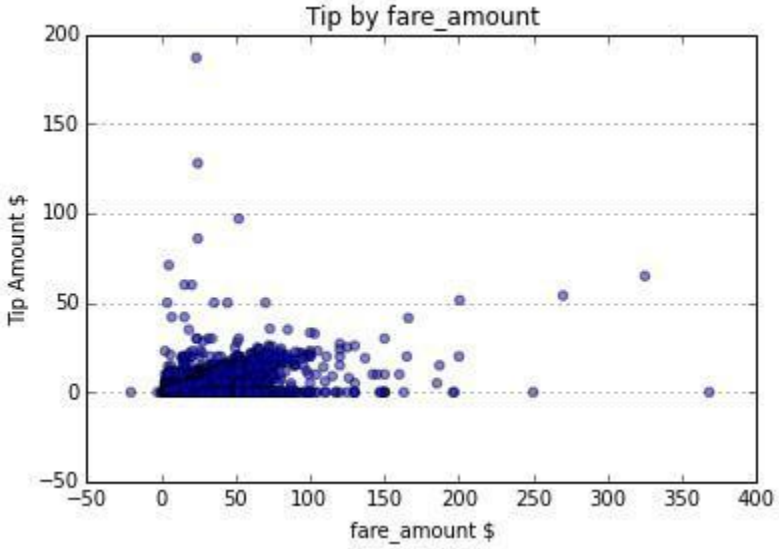
Tip vs Payment Type



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

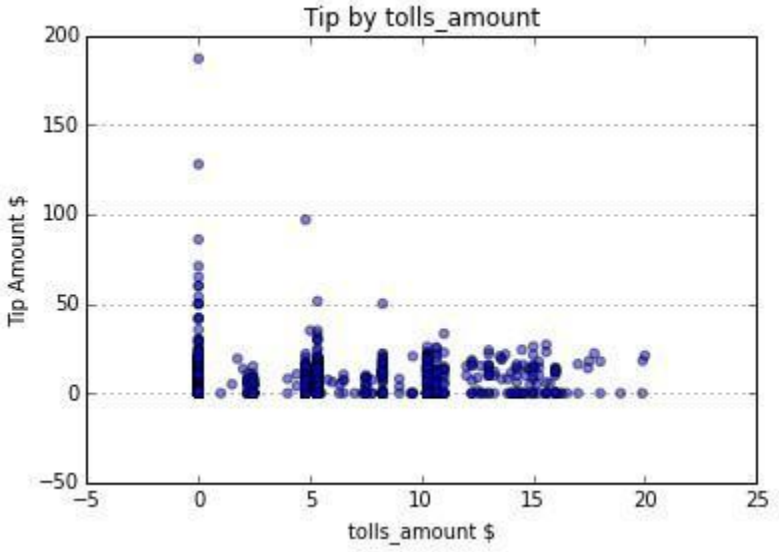
We discovered that the tip quantity and payment type had a significant association. When cash was used for payment, the tip was typically larger.

Tip vs Fare Amount



Because the tip is based on the fare of the trip or a portion of the fare amount, we expected to find a correlation between the fare amount and the tip.

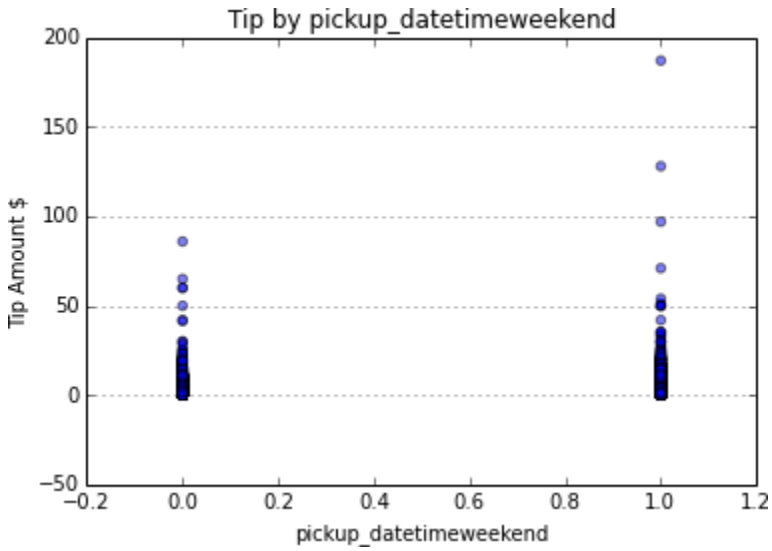
Tip vs Tolls_amount



We discovered that the average gratuity was higher on a trip without tolls compared to one with tolls.

We also attempted to draw a graph between the many features we had obtained through feature engineering, but none of them showed a definite relationship to the tip.

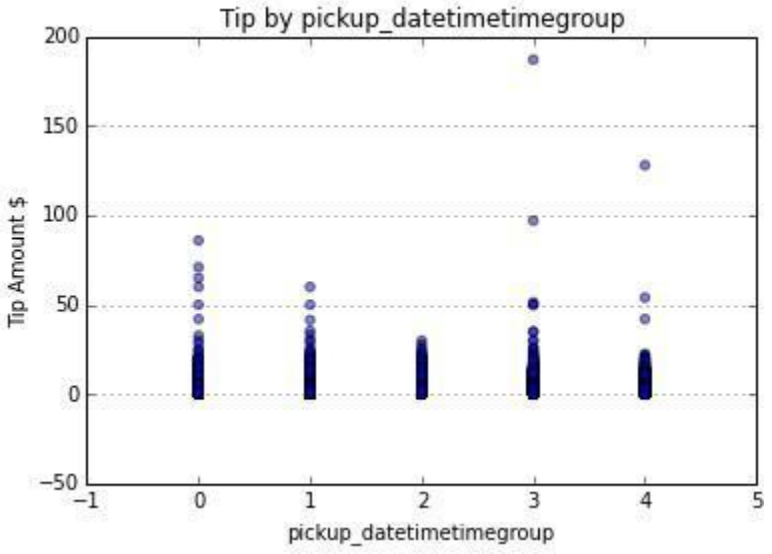
Tip vs pickup_datetimeweekend



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

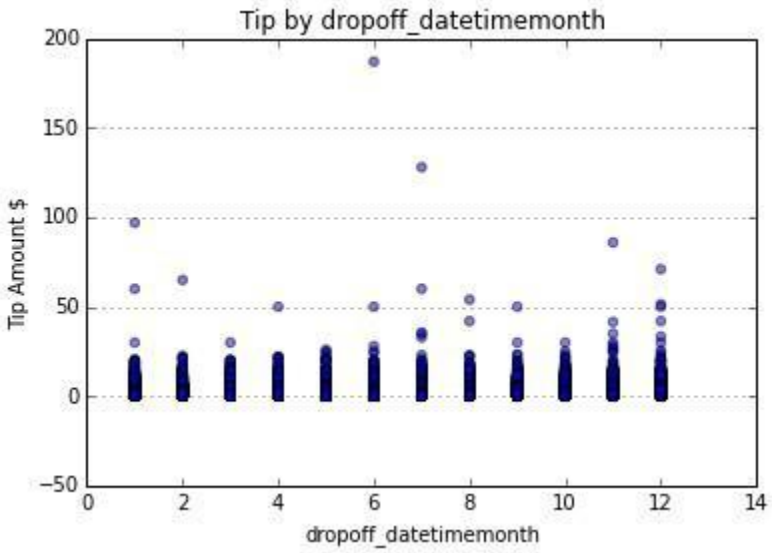
vs

pickup_datetimegroup

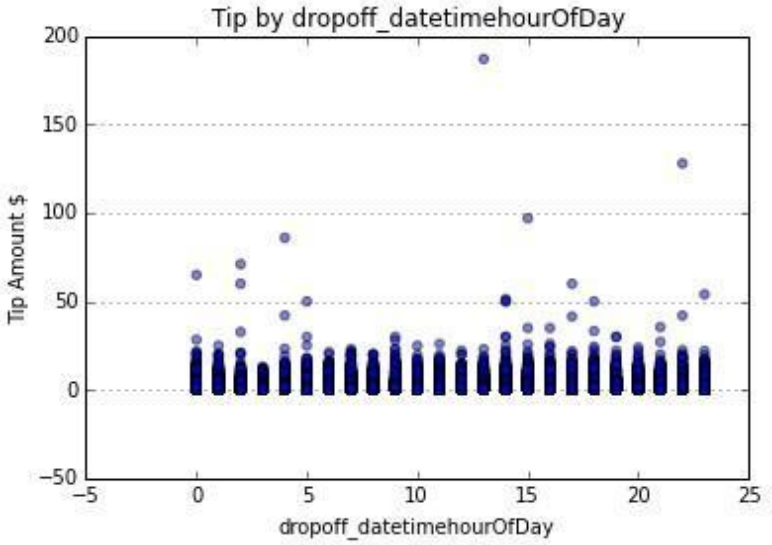


{'early': 0, 'night': 4, 'evening': 1, 'afternoon': 3, 'morning': 2}

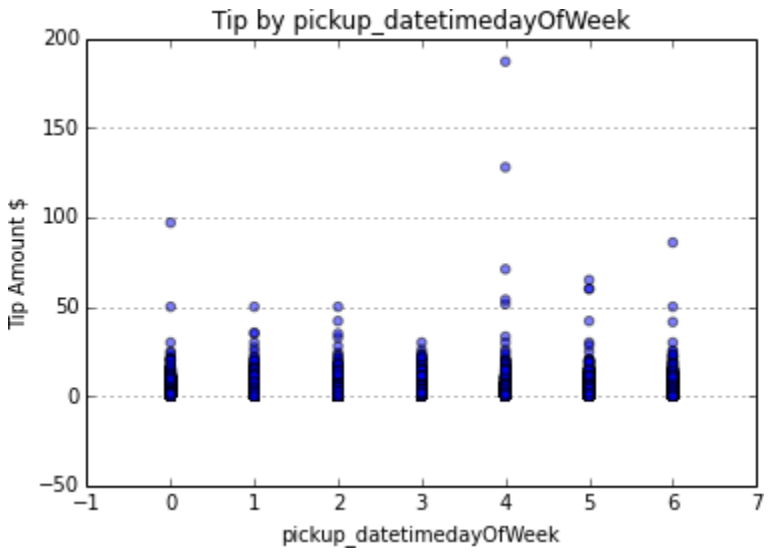
Tip vs dropofftimemonth



Tip vs dropoff_datetimehourofDay



Tip vs pickup_datetimedayofWeek



Data Cleaning

We identified missing values, outliers, and incorrect values of the characteristics based on the insights from the visualization and listing of the rows in the data frame.

In order to clean up the data, we replaced missing, inaccurate, and outlier values with NA. The criteria we used for filling with NA were as follows.

Passenger counts:

Records that contained totals higher than 6. Some records listed 225 passengers!

Trip Distance:

Greater than 1000-mile journeys and journeys covering exactly 0 miles.

Trip Time:

Trip duration is 0 seconds.

Fare Amount: Fare Amount equal to zero

Payment Type:The records with payment type as 'UNK'. This was a categorical variable.

Using the pandas dataframe dropna() function, we eliminated rows that contained N values.

Interesting debugging moment:

When the NA values were removed, more than half of the rows were also removed, and our models started to exhibit poor performance. We took our time examining the aforementioned fields to see if there was anything out of the ordinary. However, the visualizations gave us little information, which made us question the properties that were cleansed. Further investigation revealed a'store_and_forward_flag' that was NA for the majority of the rows. And dropna() was dropping all of these. To resolve the problem, we took this attribute out of the original dataset.

Optimization:

With the exception of payment type, which was a category feature, all of the attributes in our final set were numerical. To convert to numerical characteristics, we were utilizing scikit dictvectorizer. This was holding up our progress. Therefore, we changed the original dataset, enumerated the payment_type field, and transformed it into a numeric variable. Our latter code runs ran noticeably quicker.

Tip Bin Classification

The tip amount required to be divided into various classifications in order to run the classification algorithms. According to the tip distribution visualization, 50% of tips were $0, while the majority of tips fell between $1 and $20, with a handful going higher.

In the beginning, we attempted to group the tip amounts into 5 distinct classes using k-means clustering. But according to the scikit manual, using k-means for one-dimensional data is not a good idea. In order to identify gaps in the tip distribution, we used the Kernel Density Estimation (KDE). However, both of these methods produced centroid clusters that had little real-world relevance.

Therefore, we were forced to manually group the tip amounts into relevant buckets. The other classes were $5-$7.5, $7.5-$10, and > $10. We awarded zero tips to a zero tip class (0) and $1- $2, $2-$3, etc. to the remaining classes.

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

Models & Algorithms

As previously indicated, we used numerous datasets to test our models because zero tips made up 50% of the dataset.

FEATURE SET

We experimented with a variety of time- and location-based features. But using the following collection of features, our models performed well (based on accuracy measure).

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

By forecasting the majority class, we determined the baseline accuracy for each classification model. Additionally, we calculated the baseline accuracy for the regression models using the Mean Absolute Error measure. This measurement was created by calculating the mean of all such deviations, forecasting the mean for the tip, and determining the absolute deviation from the mean for each record.

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

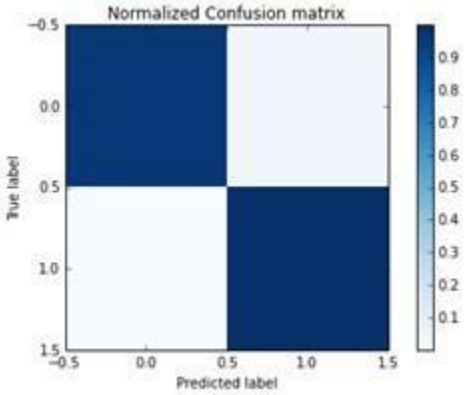
- SVM rbf classification (C = 0.1, gamma = 0.01)
- Decision Tree Classifier
- Random Forest Classifier
- Adaboost Classifier (Decision tree Classifier)

And for regression, we did

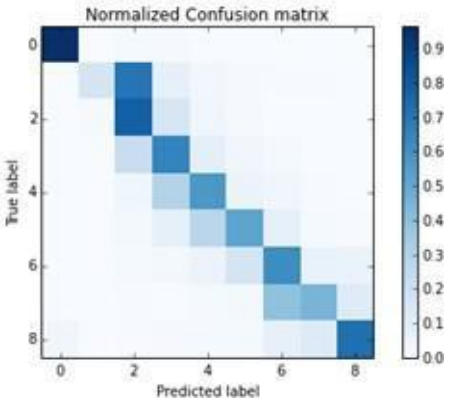
- Linear Regression
- SVM Regression
- Lasso Regression

For each model, we plotted confusion matrices to determine where our model was failing. Here are a few illustrations of the confusion matrices we encountered. We have comprehensive accuracy score findings after the confusion matrices.

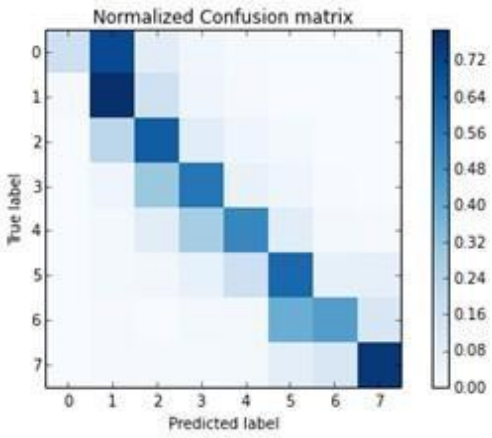
Tip / No Tip Classification



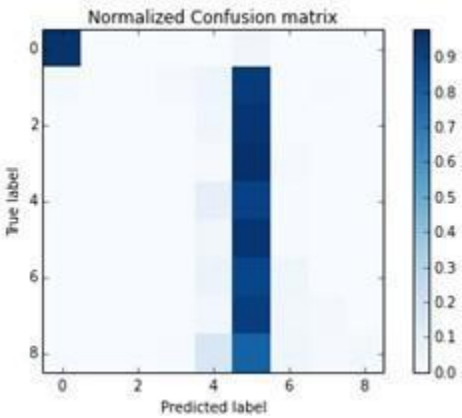
Tip Class for non zero tip data



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

With Tip - No Tip and Tip Class w/ Zero values of 98% and 81.5%, respectively, we achieved a fairly high accuracy. The accuracy was 22% for Tip class without zero values and Tip% without zero values, which was much higher than the baseline.However, for the tip% w/o zero values, we were unable to significantly outperform the baseline accuracy.

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

Simple features contribute more to the forecasts than sophisticated or composite features, which was one of the main lessons learned from this procedure. In this instance, the fare amount, the method of payment, and whether a toll was present all played a significant role in determining the tip.The pickup zip code and the cost of living index for a city were other factors that had less of an effect. Although they didn't have a significant impact, we think that they add weight to the algorithm and help it become more city independent.

We discovered additional information about the data during this process. Regression initially produced results with lesser accuracy than categorization. Since we didn't have enough time to try and increase the accuracy of these algorithms, this is not a conclusive statement. Due to the data's innate bias, they had a steep learning curve and were also more difficult to improve.

In terms of classification, this study had a very high level of accuracy and made it easier to forecast whether there would be a tip or not. However, it was simpler to estimate classes formed by binning the data on their real quantities rather than as a percentage of the amount when it comes to estimating the actual tip amount. When we added the data for tips of zero dollars as part of the dataset, it was also simpler to forecast the tip amount.This was once more most likely due to the skew of the data, as there are roughly 48% of entries with a tip of zero dollars. Therefore, deleting the data results in data that has significantly greater diversity in the classifications, making prediction more difficult.

First off, taxi drivers who receive cash payments do not declare tips, which causes a bias in the figures. As a result, using a credit or debit card significantly boosts your chance of getting a gratuity. Only one of the vendors that provided this data took into account tips for cash purchases. Another finding revealed that car-related tips were greater and tended to cluster at 20%. A quick search indicated that this was the common recommendation for the majority of card readers and a minor point that most customers generally overlook.

Real world implications

In addition to achieving our primary goal of tip prediction, which can be a helpful tool for cab drivers planning their routes and determining the most economical driving pattern, we were able to draw several conclusions from the data that provide insight into various activities in real-life scenarios.Another intriguing finding from our investigation was the finding that choosing a route with tolls appeared to have a negative effect on the tip. This is most likely a result of the fact that consumers who travel on toll roads must pay fees, which reduces their propensity to leave a tip.In contrast to the drop area, the pickup location's cost of living index appeared to have minimal impact.

Future Work

The majority of a cab driver's revenue does not come from tips. It would be beneficial to identify the various components that contribute to the overall income of the cab driver in order to better comprehend this ecosystem.
determining whether lengthier travels are really more cost-effective than journeys that result in multiple excursions within a region with a high cost of living index. Such issues present intriguing difficulties that can be further investigated in this dataset.