Predicting prices of ride-share services

Ubercool, kyoshii + khachisu + gchen49 + cporter5

Goal

Ride-hailing services have gained popularity in recent years due to attributes such as reduced travel costs, traffic congestion, and emissions. With the rise of Uber, Lyft, and other ride-hailing services, the demand for ride-sharing services has also increased. Moreover, Covid-19 has brought a surge in ride-sharing demand. Our goal is to build models that can predict the price of ride-sharing services in Chicago based on the Covid-19 cases/deaths and the demand for ride-sharing services impacted by it.

Data

We used two publicly available datasets, both of which are provided by the Chicago Data Portal. Transportation Network Providers is the first dataset we used. It captures all trips, starting from November 2018, reported by Transportation Network Providers to the City of Chicago. This dataset includes important features such as the total price of a ride and a timestamp. The second dataset we used is COVID-19 Daily Cases, Deaths, and Hospitalizations which includes the number of deaths and daily cases per day. We joined these two datasets on timestamp. Our final dataset had Total Price as a target attribute and the following features: timestamp, trip_pooled, trip_miles, cases_total, deaths_total, hospitalizations_total, cases_age_18_29, cases_age_60_69, year, and month.

Model+Evaluation Setup

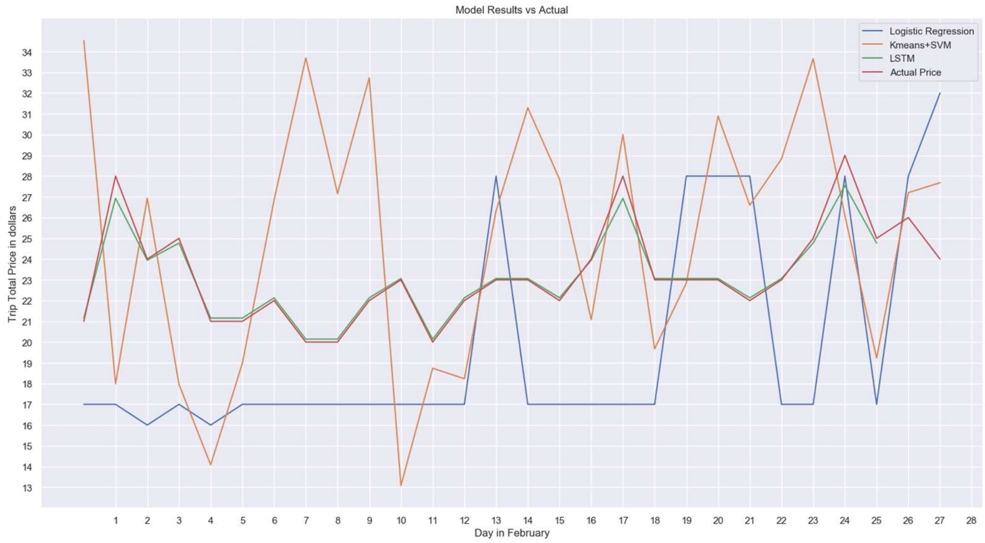
We intend to predict the price of a ride-share service for a day (in dollars) given the features in our dataset (i.e. “cases_total”, “hospitalizations_total”, and so on). We used Feb 2022 price data (the latest available data) to test our model. Since this was the case, we used MSE to score the performance of different models (in our case three) between their predictions and the ground truth price labels. To decide on specific models we wanted to use, we used PCA to learn the distribution of our dataset. From it, we learned that the data points were clustered in one big place, so we decided to use KMeans + SVM and a deep learning model, in addition to the baseline regression model. Finally, we train these models using data from 2020-01 to 2022-01, allowing the models to learn the price difference between pre-Covid and post-Covid. We trained our models on 730 data points (one data point for each day) and test it on 28 data points (for Feb 2022). Since it was a time series dataset, we could not use K-Fold. Instead, we used forward changing and report the MSEs at each stage and took the average of them to compare the model performances.

Results and Analysis

Claim #1: Deep Learning model performed the best in predicting the prices for February.
Support for Claim #1: The Table below shows the MSE for each of the three model on testing dataset. As you can see, LSTM model did an excellent job predicting the price as it handles time series data exceptionally well.

Model	MSE
Linear Regression (Baseline)	**37.6785**
KMeans + SVM	**25.4223**
LSTM	**0.1860**

Claim #2: Each model captures different aspects of the dataset.
Support for Claim #2:



As for linear regression, you can see that it is sometimes capturing the sudden spikes in the actual price. We then observe that KMeans + SVM does a fairly good job capturing the general shape of the actual price. At some points, it spikes more, and we believe this is because we train SVM model for each cluster. Hence, if we assign a data point to a “25-30” range cluster for example, the model has some freedom of picking number within that range. Finally, we see that LSTM model is almost identical to the actual price. We fear this might be because our model is overfitting on our small dataset.

Claim #3: Most of the predictive power comes from features such as “cases_total” and “trip_miles”.
Support for Claim #3: The table below shows how our KMeans + SVM model performed with different subset of features. We can immediately see that “cases_total” feature weighs a lot in the model’s decision making. We can also see that features such as “trip_total” and “trip_miles” contribute non trivial amount to our model’s decision-making.

Features	MSE
Just covid cases (cases_total)	30.6433
Just covid cases + hospitalization total + deaths total + cases age (60-69 and 18-29)	29.1432
All features	25.4223

Hypothesis

We wanted to know (1) if the two attributes “cases_total” and “trip_total” were independent of each other or not, (2) if our time series data is stationary (not affected by time) or not, and finally (3) if our time series data has a Gaussian distribution or not, and finally. Testing the above three hypotheses helped us determine which machine learning models we wanted to use for the task we have.

Findings (p = 0.05)

Claim #1: “cases_total” and “trip_total” are independent of each other
Support for Claim #1: To test this hypothesis, we used Chi squared test of independence. We did a cross-tabulation between the two variables. From this test, we obtained the p-value of 0.0001. Since it is smaller than the threshold p-value, we failed to regect the null hypothesis. Hence, “cases_total” and “trip_total” are dependent.

Claim #2: The time series data for “trip_total” we have is stationary (it is not affected by time)
Support for Claim #2: To test for the stationarity of our dataset, we used Augmented Dickey-Fuller Test which is one of the tests that check if a dataset has unit root. We use ADF because of the nature of our high dimensional dataset and since it allows us include high-order regressive process. Since we obtained the p value of 0.6064, we reject the null hypothesis, which says that our dataset is stationary.

Claim #3: Our time series dataset has normal distribution
Support for Claim #3: Since our dataset has more than two attributes, to check if our dataset has normal distribution or not, we needed to check all possible combinations of attributes. To this end, we used the Multivariate Normality Test that determines if the given group of variables come from normal distribution or not. Since we obtained the p-value of 0.0012 we reject the null hypothesis. Our multivariate dataset is not normally distributed.