Project Report
Predicting hourly customer footfall for Uber

1. Problem statement

The ability to meet the demand when it surges, and manage drivers' expectations when the demand decreases yields higher satisfaction among customers and drivers, higher operational efficiency, and increased revenue.

The goal of this project is to improve Uber's ability to meet demand by predicting the number of Uber requests for each NY taxi zone based on date, hour, and weather conditions.

I'm assuming that some of the important factors that impact demand are:

- Time of the day: during rush hours (morning & evening) the demand may increase due to the need to commute to and from the office;
- Day of the week: on the weekends and public holidays the demand may be higher due to the increased number of tourists;
- Location: downtown and areas with places of cultural interest may have higher demand;
- Weather conditions: people may prefer taking an Uber instead of driving themselves (or walking, or biking) when it's cold, snowing, or raining.

2. Data wrangling

Data used: New York for-hire vehicles trip data for 2021 from Taxi and Limousine Commission: https://www.kaggle.com/datasets/shuhengmo/uber-nyc-forhire-vehicles-trip-data-2021/data?select=fhvhv_tripdata_2021-02.parquet

This data contains:

- 12 parquet files with trips data: each row represents a trip request, each file has 1 month worth of data, ~12mln rows;
- shapefile with geometry of each taxi zone;
- csv taxi zones lookup file with zone id, zone name and the borough name where zone is located;
- csv file with daily records for weather in New York.

Target feature: Trip requests.

Given the size of data, I worked in Google collab notebooks and stored raw and pre-processed data in GCP.

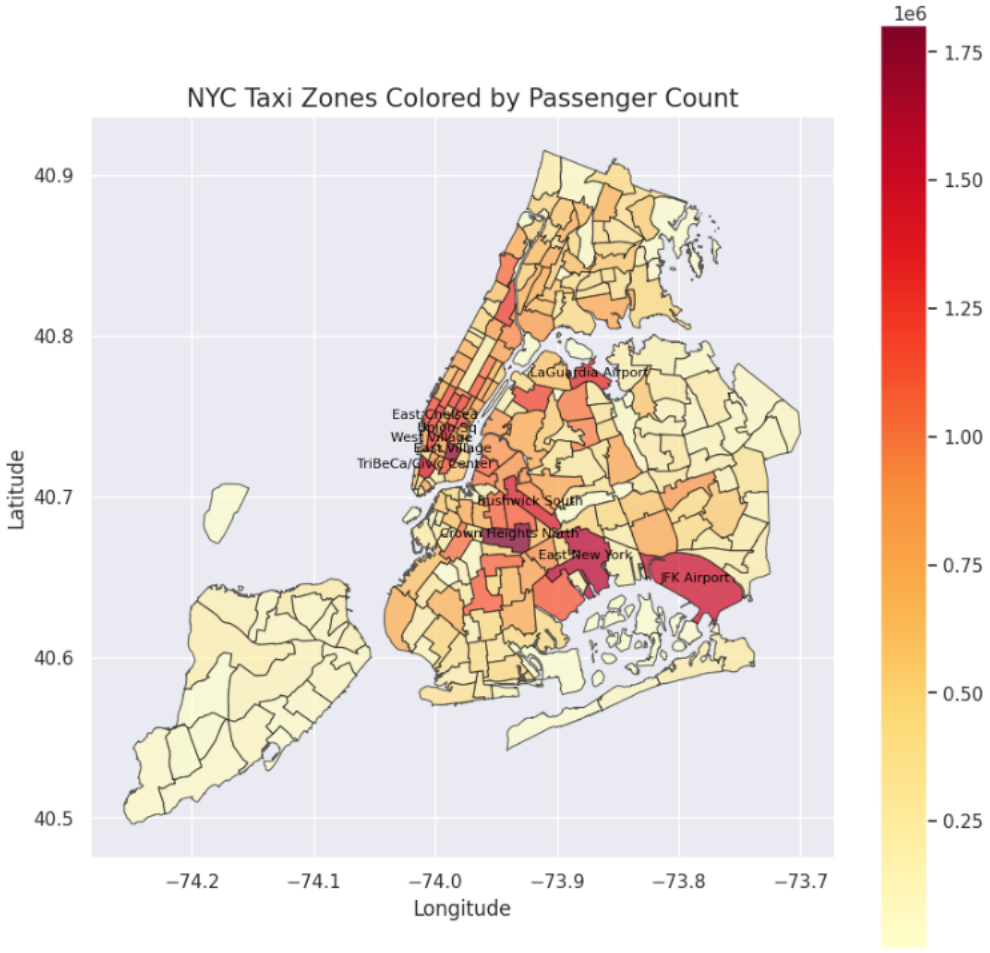
Data preparation steps:

- I aggregated trip requests data by date and hour, filtered data to include Uber only, concatenated monthly files into one data frame.
- Dropped duplicates.
- Noted on some anomalies in data where pickup time is earlier than request time. I decided to keep these records since I won't be using exact request times in my model (data is aggregated on hourly level).

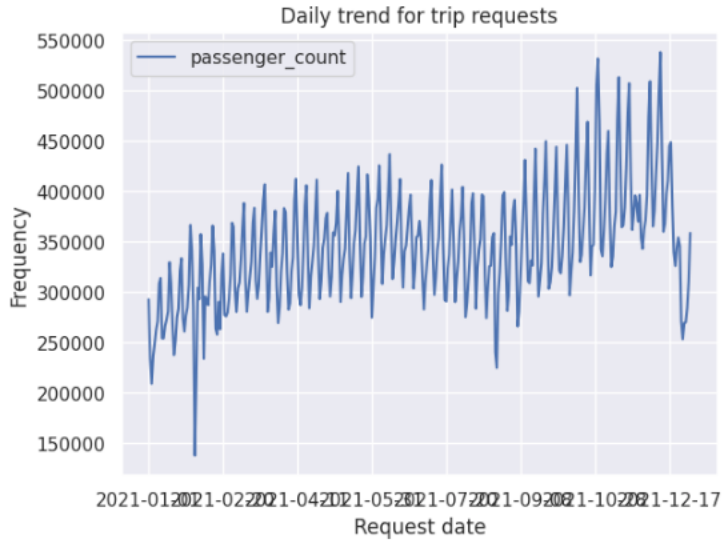
- Added weather features: temperature, dew, humidity, precipitation (snow, rain), windspeed & cloud cover (merged with trips data on trip request date).
- Added additional features: request month, request day, request day of week, flags for morning time request, daytime request, evening time request and nighttime request. This data frame will be used for modelling.
- Merged taxi zones names from lookup file with zones' geometry data. One zone id (265) has no name.
- Merged aggregated trips data with taxi zones geometry data for data exploration purposes.

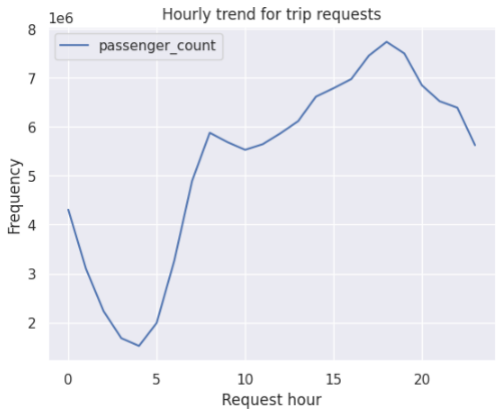
3. Exploratory Data Analysis

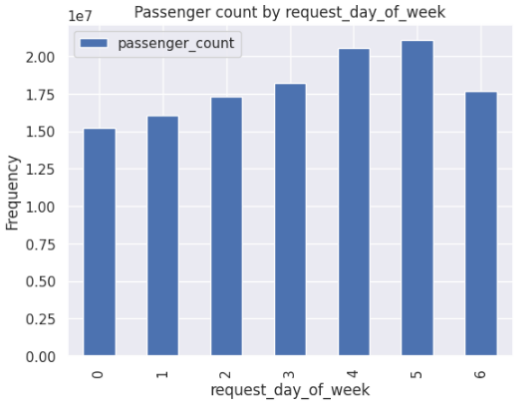
- There is a huge difference in demand between request zones. The highest number of uber requests are in Crown Heights North, East Village, East New York, JFK Airport, and Civic Center.



- There are few seasonality patterns with the number of request being the lowest during nighttime and in the beginning of the week, and the highest after work hours (5-7pm) on Fridays and on Saturdays. Also, there's an increase in demand in the last 3 months of the year.





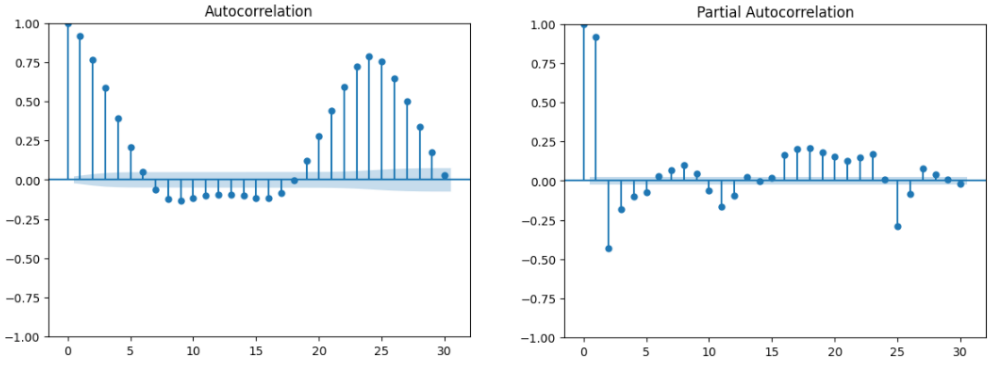


3. Preprocessing and modelling

Since my data is time series with clear seasonality patterns, I decided to try SARIMAX model (including weather as exogenous features), Linear regression and XGBoost Regression.

SARIMAX model:

- I split data into train (first 90% of records) and test (remaining 10% of records).
- I checked if my data was stationary using Dickey-Fuller test – my data was stationary (d=0).
- My ACF and PACF plots revealed that there is a strong autocorrelation in data, meaning that current values are strongly influenced by the previous values.



- I created Fourier terms for daily, weekly and yearly seasonality and exogenous features that include Fourier terms and weather features (temperature, dew, humidity, cloud cover, rain, snow).
- Initial SARIMAX model was created with order = (1, 0, 1) and seasonal order = (1, 1, 1, 24). In case this initial model looks promising, the order parameters can be optimized.
- The model didn't produce good results – it overestimates the demand on high demand days and overestimates on low demand days.

Linear Regression:

- Since there is an autocorrelation in data I created two lagged features: number of requests in the previous hour and number of requests 24 hours ago.
- I scaled the data using Standard Scaler.
- I split the data into train (first 90% of the records) and test (remaining 10%).
- The model produced very good results: R-squared of 0.91. However, in real life scenario it may be costly (or even impossible) to feed the model with previous hour's data. Due to this consideration I decided to create Linear regression #2 with only one lagged feature – the number of requests 24 hours ago.
- As expected, Linear regression #2 with only one lagged feature had a worse performance than the first linear regression, but it was still much better than SARIMA's performance (lower AIC, MAE & RMSE).

XGBoost Regression:

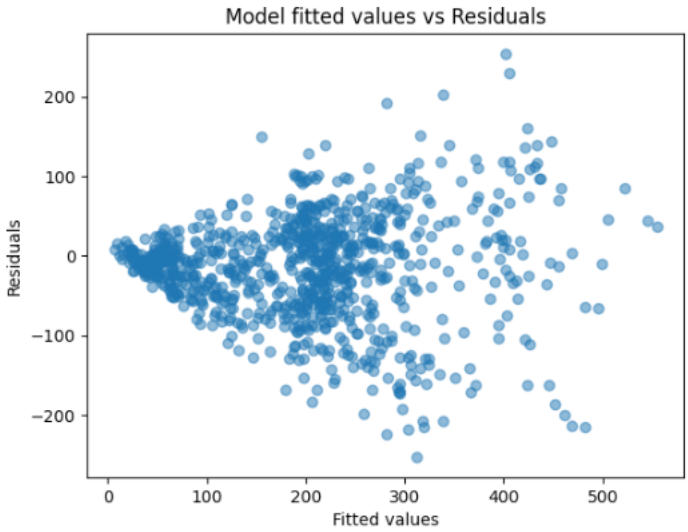
- I used the same data pre-processing steps as for the Linear regression #2.
- XGBoost Regression performed better than the Linear regression #2 (lower MAE, MSE, RMSE and higher R-squared).

Hyperparameters optimization:

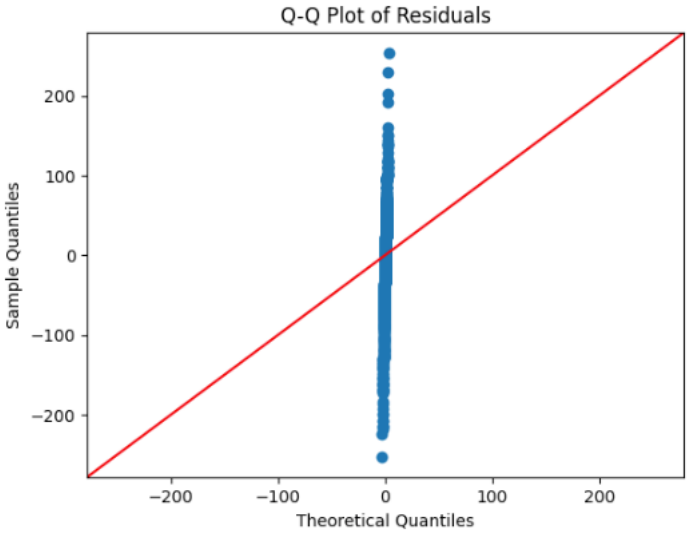
- I used randomized search with cross-fold validation to find the best values for parameters:
    - 'colsample_bytree': 0.8452413703502375,
    - 'gamma': 0.31164906341377896,
    - 'learning_rate': 0.10926940745579475,
    - 'max_depth': 9,
    - 'min_child_weight': 7,
    - 'n_estimators': 111,
    - 'reg_alpha': 0.32518332202674705,
    - 'reg_lambda': 0.7296061783380641,
    - 'subsample' : 0.8187787356776066

Model diagnostics:

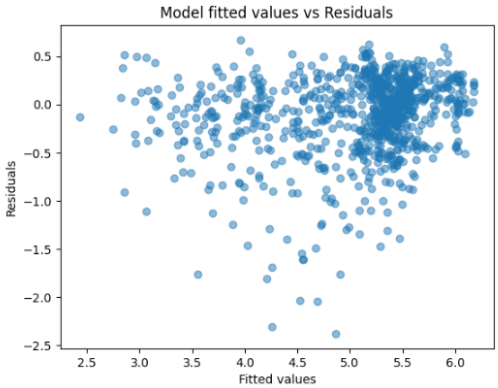
- I created Fitted values vs Residuals plot to check if residuals were randomly scattered. The plot revealed funnel shape, meaning that the data had a heteroscedasticity problem.

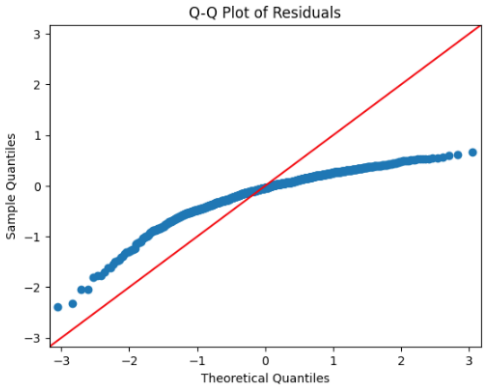


- I created a QQ plot of residuals to check if they were normally distributed. The residuals were not normally distributed.



- To achieve homoscedasticity and residuals normal distribution I log-transformed my target feature. Below are the diagnostic plots after the target feature transformation.





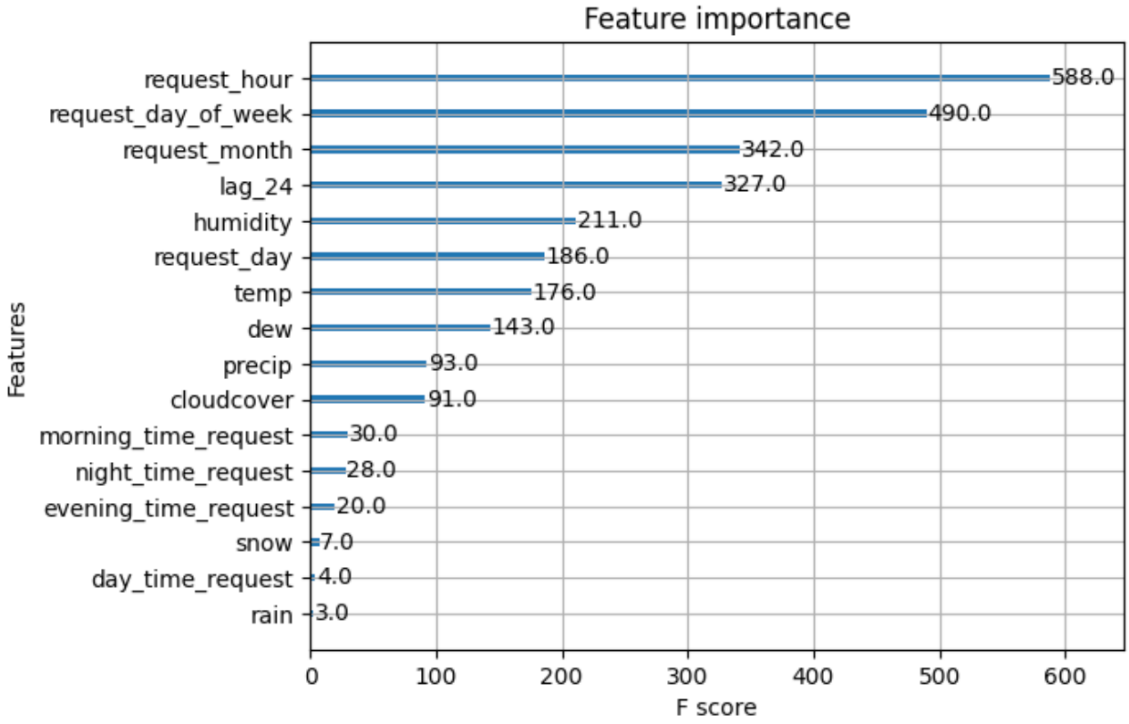
Summary of all models' performance:

Model	MAE	MSE	RMSE	AIC	R-squared
1. SARIMAX	77.6		99.5	74,997	n/a
2. Linear regression with 2 lagged features	27.6	1,393	37.3	8,843	0.91
3.Linear regression with one lagged feature	55	5,288	72.7	10,007	0.67
4.XGBoost regression	54	4,918	70	n/a	0.69
5.XGBoost regression with optimized hyperparameters	50	4,590	67.7	n/a	0.71
6.XGBoost regression after log-transformation of target variable	0.3	0.19	0.44	n/a	0.76

My best performing model is Linear regression with two lagged features (model number 2), but it may be costly to 'feed' the model with last hour's data and retrain the model. Due to this reason, my final chosen model is XGBoost regression with optimized hyperparameters and log-transformed target feature.

4. Conclusions

Based on feature importance graph, hour of the day, week, and month of request, along with number of trip requests at specified hour on a previous day are the most important features to predict trip requests. Weather features play only moderate role in predictions with humidity being the most important one.



5. Future research

Given the huge size of data and the amount of computational resources needed, for the purpose of this project I created a model for one zone only. Different zones (like, airport

zones vs tourist attraction zones vs suburban zones) will probably have different hourly and weekly trends. In real-life scenario, to account for this we could either:

- Create dummy variables for each zone and include them as predictors in the model; or
- Train a separate model for each zone.