Data Scientist Nano Degree (Capstone Project)

Car Booking Analysis and Prediction

Prepared by: Tarek Abd ElRahman Ahmed ElAyat

Contents

Hi-level description and objectives

In Egypt transportation represent a major challenge, which has been partially relieved after new players entered the market (car booking companies like Uber, Careem, SWVL…etc).

These new players used technology to offer a more reliable, secured and intelligent transportation options.

The competition in this business area is fierce, and the winner will be the one who employ data analytics to provide better customer service with minimal cost.

The objective of this project is to support car booking companies with analysis and prediction to offer better service to the customer.

Unfortunately, dataset from these companies is not publicly available, so in this project I will use NYC Taxi and Limousine Commission dataset as an alternative.

Challenges addressed by the use case

Provide car booking companies with analytics to maximize profit and provide customers with superb customer experience with minimal cost, and also find new business opportunities:

- Fleet management:
    - Deploy the fleet at the most demanding areas at specific time.
    - Deploy the fleet based on driver's preferences for drop off locations (a driver may prefer a drop off near his home)
- Fare prediction:

    Customer need to know the estimated fare for his trip before start

- Trips scheduling:

    Suggest best trip time within a specific time window defined by the customer, this best trip time should have lowest trip duration and fare amount.

- New business opportunities:

    Determine and update best buses routes based on highest pickup and drop-off locations at specific times.

Pain points and impact of each challenge:

Car booking companies needs analytics that help them manage and deploy their fleet in a way that maximizes the ROI and provide better service.

It also to provide customers with high accuracy prediction for their trip conditions/details (i.e. trip duration, estimated fare, best time to take the trip…)

And more over suggestion for bus routes/times for routes with high demand.

Goals, success criteria, constraints and assumptions

- What are the top pick up locations per specific time?
- What are the top pick up locations per specific times and favorite drop off list?
- Predict fare amount.
- Suggest best time to for a specific trip.

Success Criteria -> More than 70% prediction accuracy

Constrains and Assumptions -> prediction algorithms will give close results if applied on Egyptian datasets.

Available data, data sources and required resources:

Unfortunately, dataset is not available for the Egyptian market, so I will use a similar data from other markets to build and test the use case.

The data used were collected and provided to the NYC Taxi and Limousine Commission (TLC) by technology providers authorized under the Taxicab & Livery Passenger Enhancement Programs (TPEP/LPEP).

The dataset is publicly available under the following link:

https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page

I'll use Yellow Taxi trips data from year 2019 to avoid the impact of Covid-19.

Modeling approach for each challenge:

Some of the above business questions/use cases shall be answered by a statistical analysis, and some will need prediction models like (Linear regression and Xgboost)

Application of the use case into operational solution:

Developed algorithms may be used in real-time planning and management of the fleet.

Provide customers with estimated trip details, and may be suggestions for best trip time along the day.

New buses routes suggestions.

Data Exploration and Understanding

A dataset with over 40 million trips was used in the analysis (42821916 trips, with 18 columns)

Data Structure:

The dataset structure and features descriptions are publicly available, and below are the main tables with their data structure.

- Table “yellow_tripdata” (each month has a separate file):

Field Name	Description
VendorID	A code indicating the TPEP provider that provided the record. 1= Creative Mobile Technologies, LLC; 2= VeriFone Inc.
tpep_pickup_datetime	The date and time when the meter was engaged.
tpep_dropoff_datetime	The date and time when the meter was disengaged.
Passenger_count	The number of passengers in the vehicle. This is a driver-entered value.
Trip_distance	The elapsed trip distance in miles reported by the taximeter.
PULocationID	TLC Taxi Zone in which the taximeter was engaged
DOLocationID	TLC Taxi Zone in which the taximeter was disengaged
RateCodeID	The final rate code in effect at the end of the trip. 1= Standard rate 2=JFK 3=Newark 4=Nassau or Westchester 5=Negotiated fare 6=Group ride
Store_and_fwd_flag	This flag indicates whether the trip record was held in vehicle memory before sending to the vendor, aka "store and forward," because the vehicle did not have a connection to the server. Y= store and forward trip N= not a store and forward trip
Payment_type	A numeric code signifying how the passenger paid for the trip. 1= Credit card 2= Cash 3= No charge 4= Dispute 5= Unknown 6= Voided trip
Fare_amount	The time-and-distance fare calculated by the meter.
Extra	Miscellaneous extras and surcharges. Currently, this only includes the $0.50 and $1 rush hour and overnight charges.
MTA_tax	$0.50 MTA tax that is automatically triggered based on the metered rate in use.
Improvement_surcharge	$0.30 improvement surcharge assessed trips at the flag drop. The improvement surcharge began being levied in 2015.
Tip_amount	Tip amount – This field is automatically populated for credit card tips. Cash tips are not included.
Tolls_amount	Total amount of all tolls paid in trip.
Total_amount	The total amount charged to passengers. Does not include cash tips.

- Table “taxi+_zone_lookup”:
  This is the lookup table to map Pickup and DropOff Location IDs to Boroughs, Zones and service_zone names.

Data Exploration:

Upon initial data exploration missing values and un-logical values (-ve values or extreme outliers) were found which affected the data visualization and would affect any conclusion derived from the data.

Moreover, to explore that data, some feature engineering is needed first.

So I started with the basic engineered features and basic data cleaning to be able to visualize the data.

Feature Engineering:

From first look we can notice that the following engineered features can be derived from the Pickup and Dropoff times:

- tripduration_mins
- year, month, day, hour of the trip
- dayofweek
- weekendflag
- holidayflag

Initial Data Cleaning:

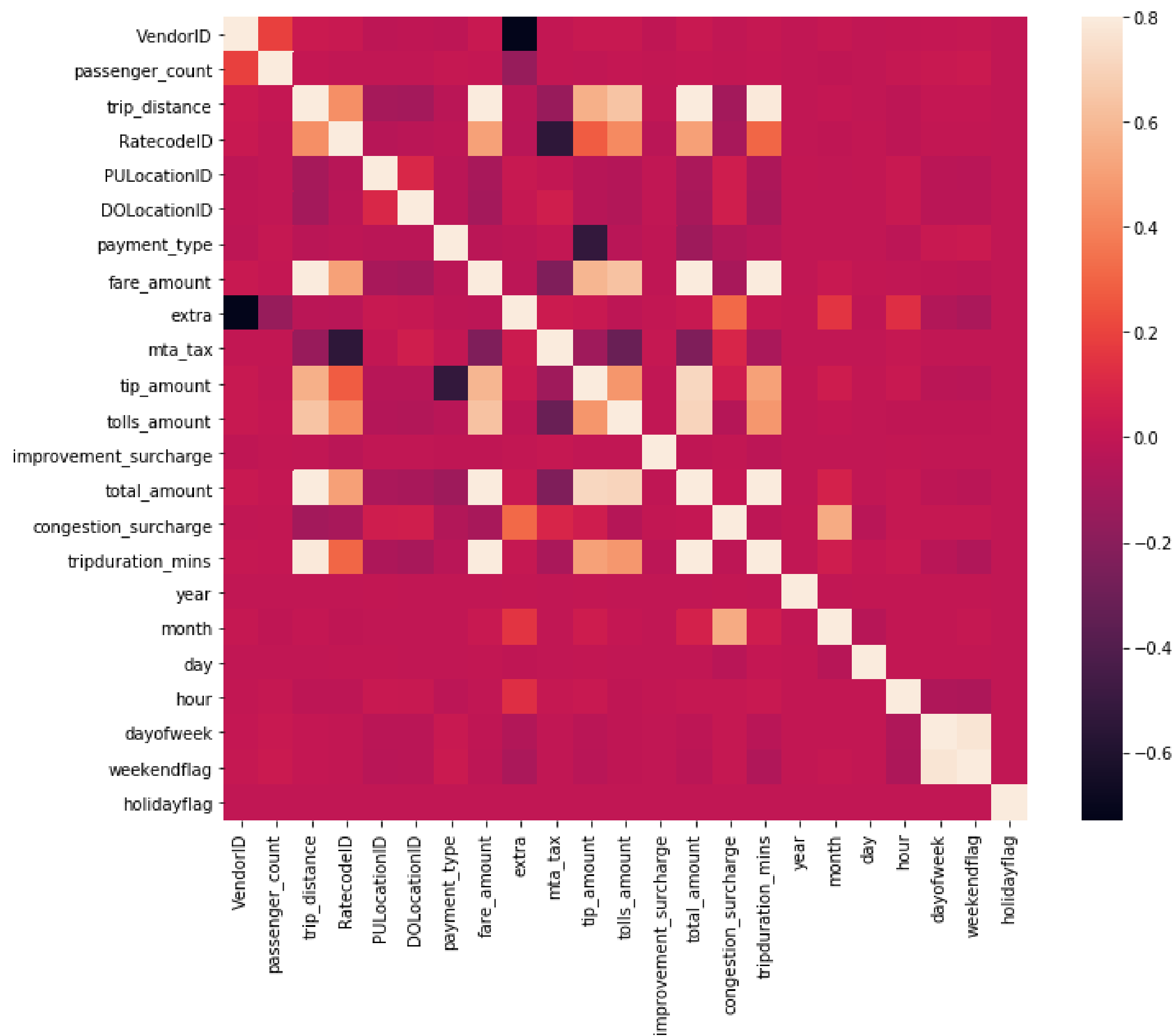
Based on the ranges of the data I have applied the following:

- drop rows with –ve values for 'fare_amount', 'total_amount', 'tripduration_mins', 'tip_amount', 'tolls_amount', 'improvement_surcharge', 'congestion_surcharge', 'trip_distance', 'extra', 'mta_tax'
- drop rows with 'passenger_count' > 6 and 'RatecodeID' > 6
- drop rows with outliers ('fare_amount' > 100, 'tripduration_mins' > 180, 'tip_amount' > 20, 'tolls_amount' > 30, 'extra' > 5, 'mta_tax' > 1, 'improvement_surcharge' > 1 OR 'trip_distance' > 30
- Fill na values in 'congestion_surcharge' with 0.

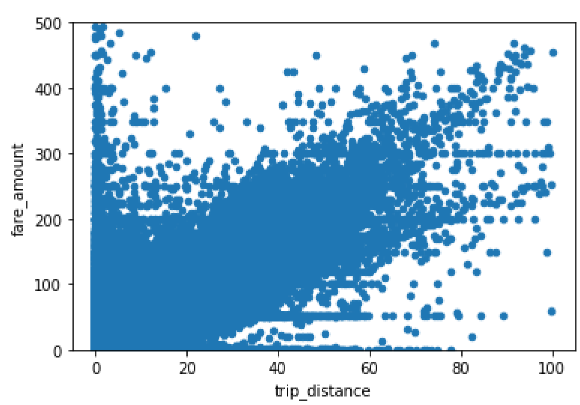
Continue Data Exploration after cleaning:

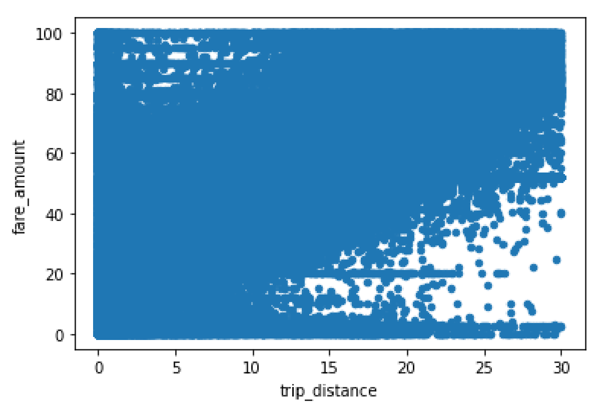
I have started with the Correlation matrix for all features

As expected for example, fare_amount has strong correlation to the trip distance and trip duration.

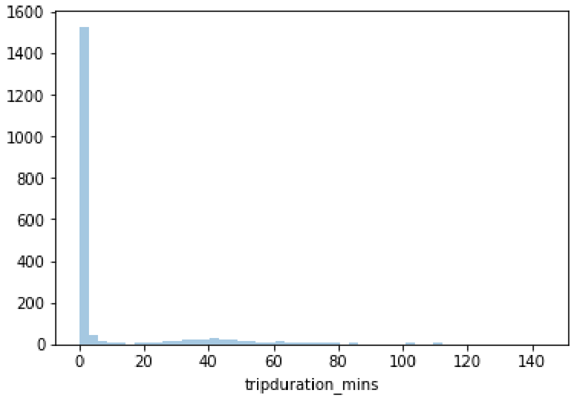


And from the below “Fare vs. Trip Distance” charts we can find that fare amount is correlated to the trip distance with minor exceptions. However, after data cleaning the view is much better and logical.

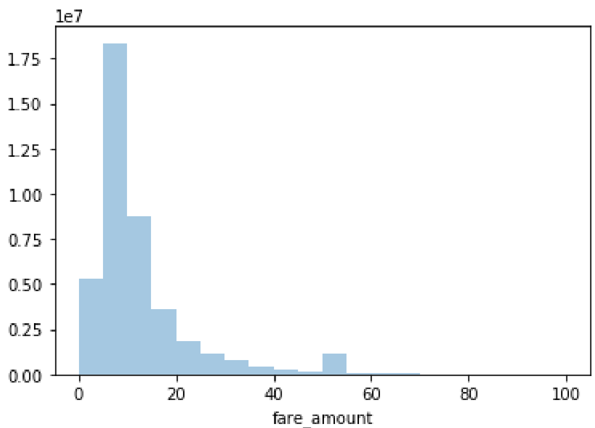


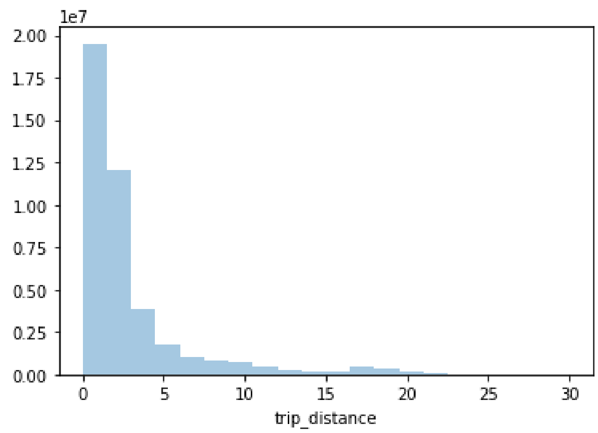


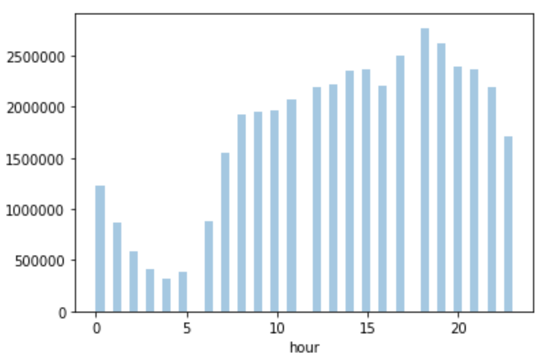
And for the trips with long distance and low fare amount, below its trip time distribution (seems these trips were on free roads and took short time, that’s why the fare was low).

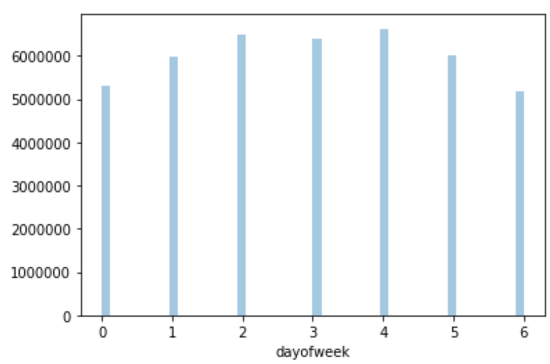


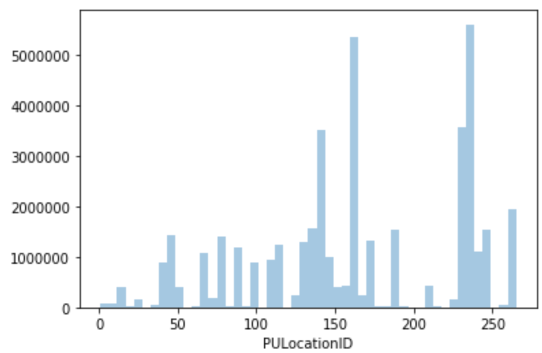
Below charts also show the distribution analysis after data cleaning.

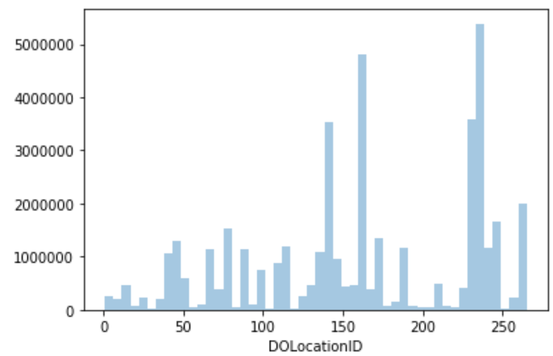








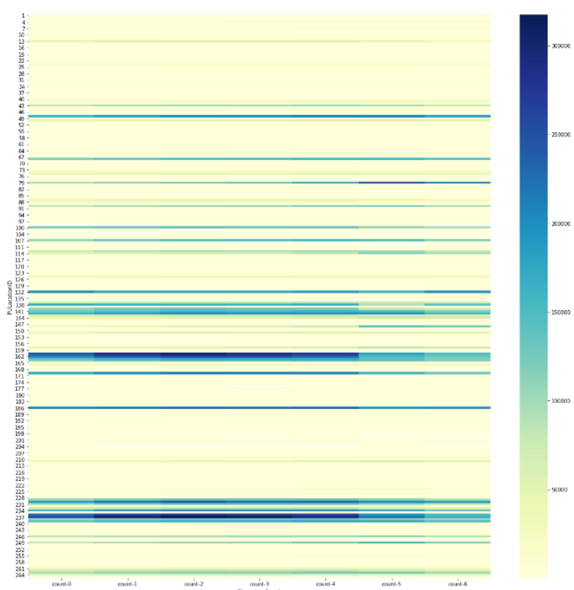


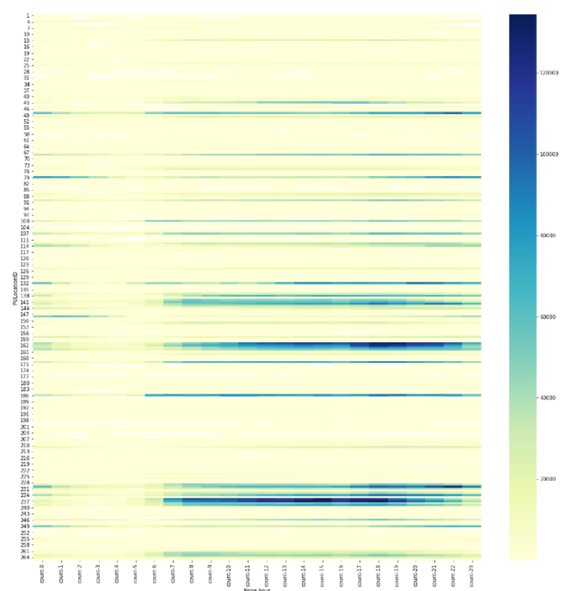


From the above distributions we find:

- The majority of the trips are short (below 5 km) with fare below 20$.
- Sunday and Monday have slightly lower number of trips.
- Distribution over day hours is logical in general (low after mid-night till working hours start)
- Pickup and Drop-off locations almost have the same distribution.

Below as well a heat map for the trips count for each pickup location over days of the week and over day hours as well.

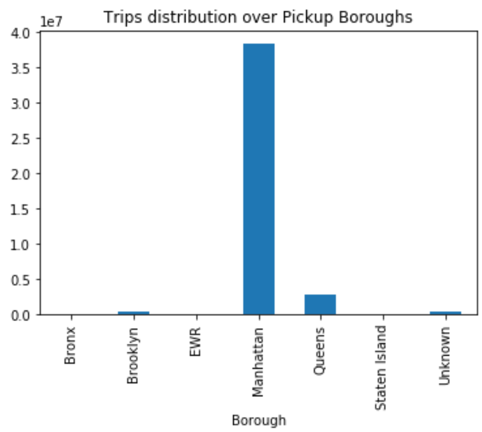


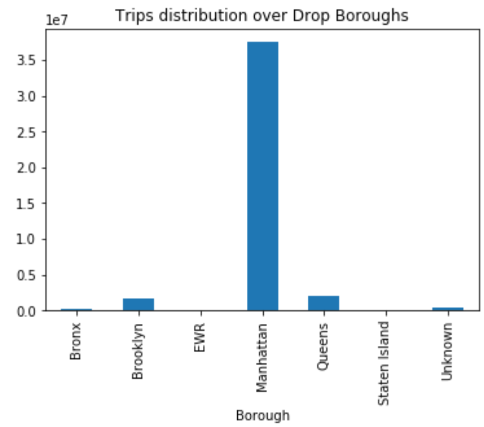


Business questions/use cases answers:

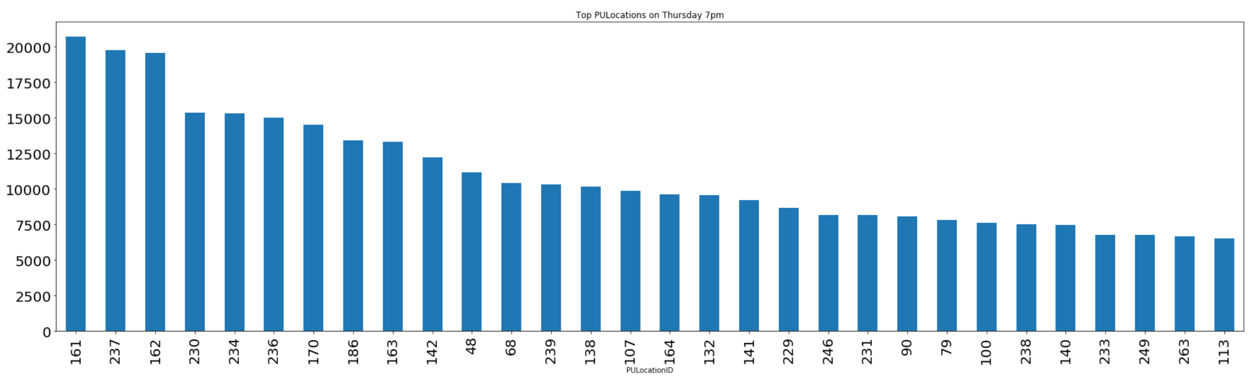
- What are the most demanding areas at specific time?

First let's have a look at the trips distribution over New York Boroughs, we can find that Manhattan has the highest share in our dataset (as expected as per the yellow taxi service areas).



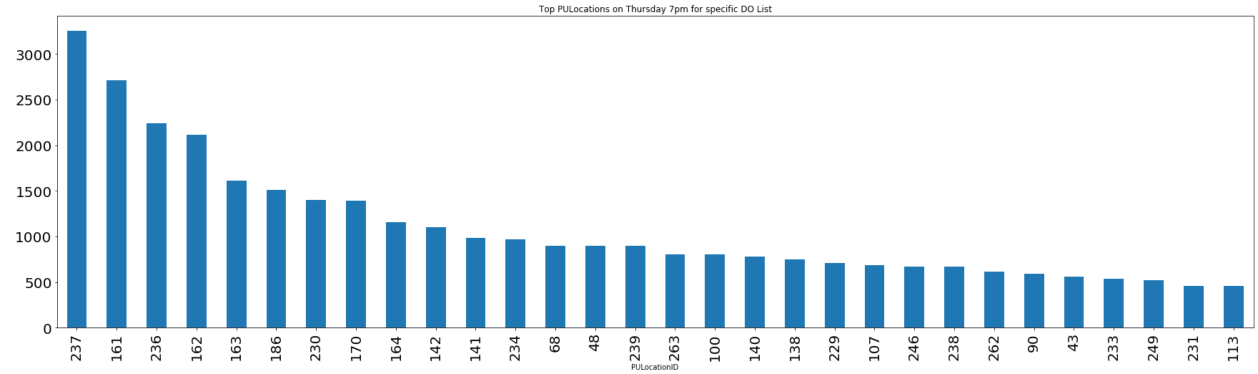


And to answer our question we can build on the history we have in our dataset, for example below are top pick up locations for a specific day and hour (Thursday 7pm).



- Deploy the fleet based on driver's preferences for drop off locations (a driver may prefer a drop off near his home).
  In this case the driver may enter his favorite drop off locations and based on historical data analysis we suggest top locations where he can work for specific time.

  For example, below are top pick up locations for a specific day and hour (Thursday 7pm) which will most likely will lead to the favorite drop off locations [230, 234, 236]



- Fare prediction:

    Please refer to "Modelling: Fare amount prediction" section.

- Trips scheduling:

    Here to suggest best trip time within a specific time window defined by the customer, the system shall predict the trip duration and far amount for each hour in the window specified by the customer and based on this the best suitable time with minimum trip duration and fare amount shall be presented to the customer.

Modelling: Fare amount prediction

Here I am listing all used scenarios for Fare amount prediction along with its R2 and RMSE scores to evaluate them

Scenario 1:

Here I used almost all the features with linear regression model as a baseline:

['trip_distance', 'RatecodeID', 'tripduration_mins', 'month', 'hour', 'weekendflag', 'holidayflag', 'passenger_count', 'PULocationID', 'DOLocationID', 'payment_type']

And I used all features as numerical (no categorical).

The output of this scenario was:

```
The r-squared score for our model on Training data is 0.9408655229497245 o
n 14712202 values.
The mean_squared_error score for our model on Training data is 6.879148895
127688 on 14712202 values.
The root_mean_squared_error score for our model on Training data is 2.6228
13164357631 on 14712202 values.

The r-squared score for our model on Testing data is 0.9409164098556393 on
6305230 values.
The mean_squared_error score for our model on Testing data is 6.8711970854
87989 on 6305230 values.
The root_mean_squared_error score for our model on Testing data is 2.62129
683276961 on 6305230 values.
```

And the coef_weights:

	est_int	coefs	abs_coefs
1	RatecodeID	4.877630	4.877630
0	trip_distance	1.789178	1.789178
6	holidayflag	-0.380822	0.380822
2	tripduration_mins	0.317012	0.317012
5	weekendflag	-0.079736	0.079736
3	month	0.021592	0.021592
4	hour	-0.006859	0.006859
10	payment_type	0.001454	0.001454
9	DOLocationID	-0.000803	0.000803
7	passenger_count	0.000387	0.000387

	est_int	coefs	abs_coefs
8	PULocationID	-0.000010	0.000010

Scenario 2:

Here I excluded some features based on the correlation matrix and scenario 1 coefficients and kept main features plus the datetime engineered features.

['trip_distance', 'RatecodeID', 'tripduration_mins', 'month', 'hour', 'weekendflag', 'holidayflag']

Also treated all features as numerical features (no categorical)

And also used Linear Regression; the output was almost the same

```
The r-squared score for our model on Training data is 0.9408385370477721 o
n 14712202 values.
The mean_squared_error score for our model on Training data is 6.882288181
156071 on 14712202 values.
The root_mean_squared_error score for our model on Training data is 2.6234
11553903823 on 14712202 values.

The r-squared score for our model on Testing data is 0.940890929867919 on
6305230 values.
The mean_squared_error score for our model on Testing data is 6.8741603112
65811 on 6305230 values.
The root_mean_squared_error score for our model on Testing data is 2.62186
19931769505 on 6305230 values.
```

	est_int	coefs	abs_coefs
1	RatecodeID	4.873097	4.873097
0	trip_distance	1.790674	1.790674
6	holidayflag	-0.357203	0.357203
2	tripduration_mins	0.317114	0.317114
5	weekendflag	-0.076037	0.076037
3	month	0.021624	0.021624
4	hour	-0.007014	0.007014

Scenario 3:

Here I used the same features from scenario 2, with linear regression model as well, but handled some features as categorical and applied one-hot-encoding on them.

The output has very minor improvement:

```
The r-squared score for our model on Training data is 0.9432230301501271 o
n 14712202 values.
The mean_squared_error score for our model on Training data is 6.604898679
993168 on 14712202 values.
The root_mean_squared_error score for our model on Training data is 2.5699
997431893196 on 14712202 values.

The r-squared score for our model on Testing data is 0.943321447117103 on
6305230 values.
The mean_squared_error score for our model on Testing data is 6.5915003882
98767 on 6305230 values.
The root_mean_squared_error score for our model on Testing data is 2.56739
1748116903 on 6305230 values.
```

Scenario 4:

Here I used XGBoost Regressor with the same features of scenario 3 and the following hyper parameters: colsample_bytree = 0.3, learning_rate = 0.1, vmax_depth = 5, alpha = 10, n_estimators = 10

and the output was with much lower score as following:

```
The r-squared score for our model on Training data is 0.3242116292704751 o
n 14712202 values.
The mean_squared_error score for our model on Training data is 78.61486320
225254 on 14712202 values.
The root_mean_squared_error score for our model on Training data is 8.8665
02309380659 on 14712202 values.

The r-squared score for our model on Testing data is 0.32425626984223854 o
n 6305230 values.
The mean_squared_error score for our model on Testing data is 78.586428784
23606 on 6305230 values.
The root_mean_squared_error score for our model on Testing data is 8.86489
869001536 on 6305230 values.
```

Scenario 5:

Here I just changed the n_estimators hyper parameter in scenario 4 from 10 to 50 and it gave the highest score among all scenarios as following:

Summary of different models scores:

	Scenario 1	Scenario 2	Scenario 3	Scenario 4	Scenario 5
R2 – on Training	0.9408655	0.9408385	0.9432230	0.3242116	0.9636330
RMSE – On Training	2.6228131	2.6234115	2.5699997	8.8665023	2.0568394
R2 – on Testing	0.9409164	0.9408909	0.9433214	0.3242562	0.9638064
RMSE – On Testing	2.6212968	2.6218619	2.5673917	8.8648986	2.0516272

Scenario #5 XGBoost gives the highest R-squared and lowest RMSE.

And in general Fare_amount prediction will be done with the following features set used for training.

['trip_distance', 'RatecodeID', 'tripduration_mins', 'month', 'hour', 'weekendflag', 'holidayflag']

Conclusion

In this project we have applied different statistical analysis and prediction to answer our business questions that would help car booking companies provide better service to the customers.

The same analysis concepts can be applied when Egyptian car booking dataset is available, which would give insights on the Egyptian market and help solving the transportation problem in Egypt.