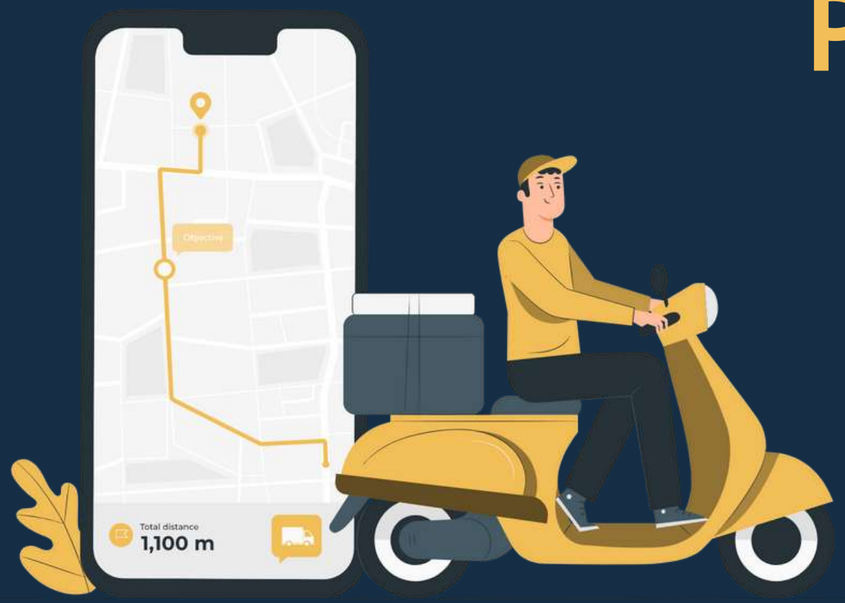
Food Delivery Time Prediction



✓ PRESENTED BY: 1.Deepali Ghumare
2.Manisha Shinde
3.Pallavi Kale
4.Akash Satapute
5.Vaibhav Hegadkar

Introduction

Food delivery is a service that allows customers to order food from restaurants and have it delivered to their doorstep.

More businesses are moving online these days, and consumers are ordering online instead of traveling to the store to buy. Zomato and Swiggy are popular online platforms for ordering food products. Other examples are Uber Eats, Food Panda, and Deliveroo, which also have similar services. They provide food delivery options.

If the order is complete, a partner will pick up and deliver the meal to the given address via a delivery service. In online food-ordering businesses, delivery time is critical. As a result, estimated food delivery time prediction to reach the buyer's location is critical.

Objectives

1. What factors can affect the time in delivering food from the restaurant to the destination location?
2. How much food delivery time prediction accuracy performance?

Data Preparation

General Info

```
5]: food_delivery_data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11397 entries, 0 to 11396
Data columns (total 20 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   ID                           11397 non-null  object
 1   Delivery_person_ID           11397 non-null  object
 2   Delivery_person_Age          11397 non-null  object
 3   Delivery_person_Ratings      11397 non-null  object
 4   Restaurant_latitude          11397 non-null  float64
 5   Restaurant_longitude         11397 non-null  float64
 6   Delivery_location_latitude   11397 non-null  float64
 7   Delivery_location_longitude  11397 non-null  float64
 8   Order_Date                   11397 non-null  object
 9   Time_Orderd                  11397 non-null  object
 10  Time_Order_picked            11397 non-null  object
 11  Weatherconditions            11397 non-null  object
 12  Road_traffic_density         11397 non-null  object
 13  Vehicle_condition            11397 non-null  int64
 14  Type_of_order                11397 non-null  object
 15  Type_of_vehicle              11397 non-null  object
 16  multiple_deliveries          11397 non-null  object
 17  Festival                     11397 non-null  object
 18  City                         11397 non-null  object
 19  Time_taken(min)              11397 non-null  object
dtypes: float64(4), int64(1), object(15)
memory usage: 1.7+ MB
```

- Dataset consists 11397 rows and 20 columns. Then the dataset also consists of 5 numerical data and 15 categorical data
- The problem faced is a regression problem, namely predicting the time needed to deliver food (Time taken)
- There are no missing value, duplicated value, and odd data in this dataset
- All numerical values contained in the dataset are quite reasonable

Data Preparation

Distance Calculation

```
In [36]: #Calculate distance between restaurant location & delivery location
         def calculate_distance(df):
             df['distance']=np.zeros(len(df))
             restaurant_coordinates=df[['Restaurant_latitude','Restaurant_longitude']].to_numpy()
             delivery_location_coordinates=df[['Delivery_location_latitude','Delivery_location_longitude']].to_numpy()
             df['distance'] = np.array([geodesic(restaurant, delivery) for restaurant, delivery in zip(restaurant_coordinates, delivery_l
             df['distance']= df['distance'].astype("str").str.extract('(\d+)').astype("int64")

         calculate_distance(food_delivery_data)
         food_delivery_data.head()
```

	ID	Delivery_person_ID	Delivery_person_Age	Delivery_person_Rating	Distance	Type_of_order	Type_of_vehicle	Time_taken(min)
0	4607	INDORES13DEL02	37	4.	3.025149	Snack	motorcycle	24
1	B379	BANGRES18DEL02	34	4.	20.183530	Snack	scooter	33
2	5D6D	BANGRES19DEL01	23	4.	1.552758	Drinks	motorcycle	26
3	7A6A	COIMBRES13DEL02	38	4.	7.790401	Buffet	motorcycle	21
4	70A2	CHENRES12DEL01	32	4.	6.210138	Snack	scooter	30

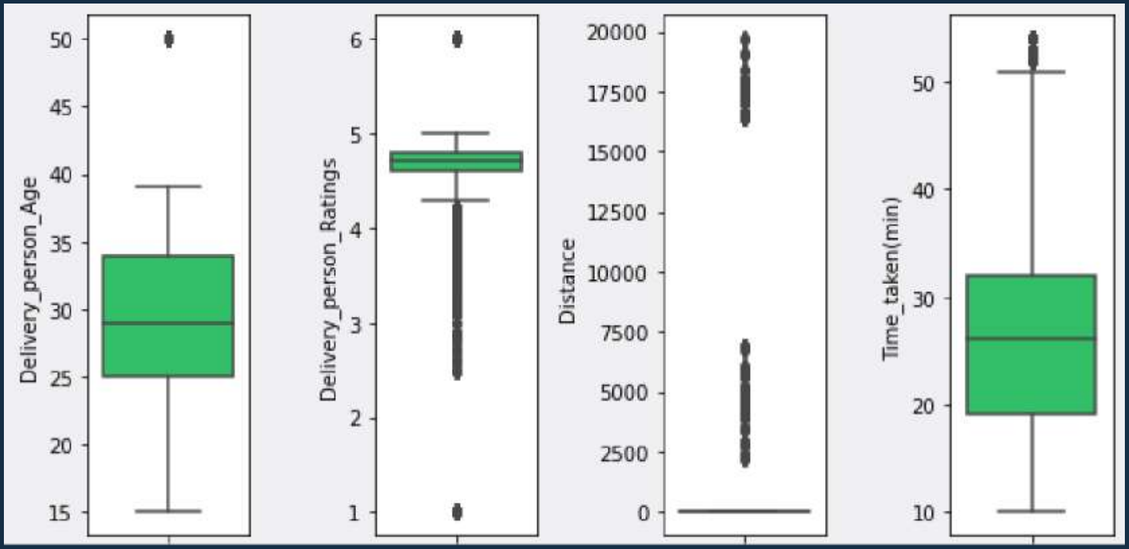
To get the time needed to deliver food, the distance between the restaurant and the delivery location is needed. To get the required distance, we use **geodesic function** which can be used to calculate the distance between 2 locations by utilizing longitudes and latitudes.

Geodesic Function

Library- geopy.distance.geodesic

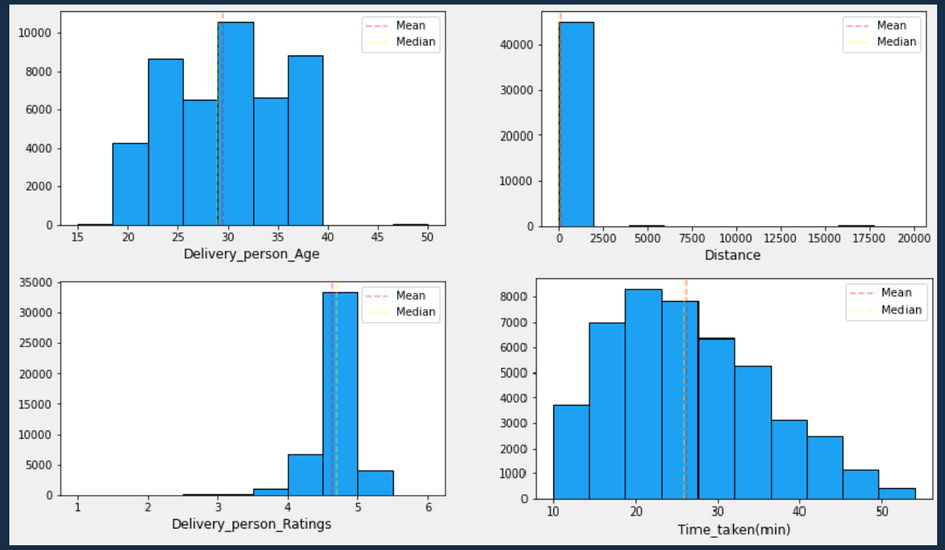
Exploratory Data Analysis

Univariate Analysis



Outliers Check

Extreme outliers at Delivery_person_Ratings column are on lower boundary (< 3.9), meanwhile distance column have extreme outliers in upper boundary (> 31.9)

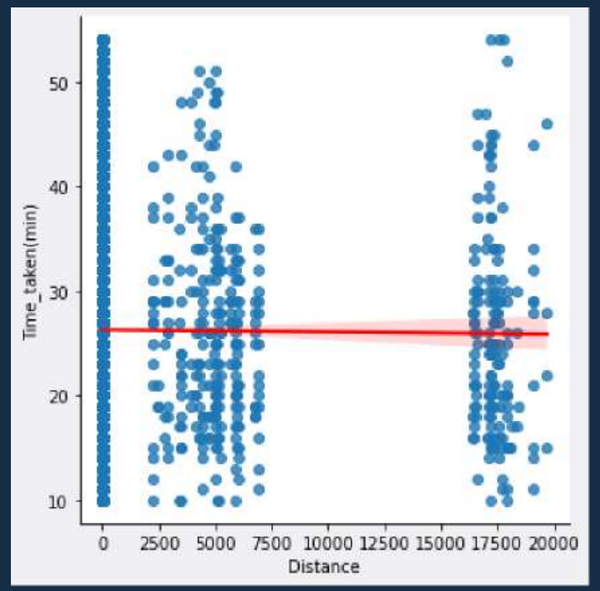


Data Distribution Check

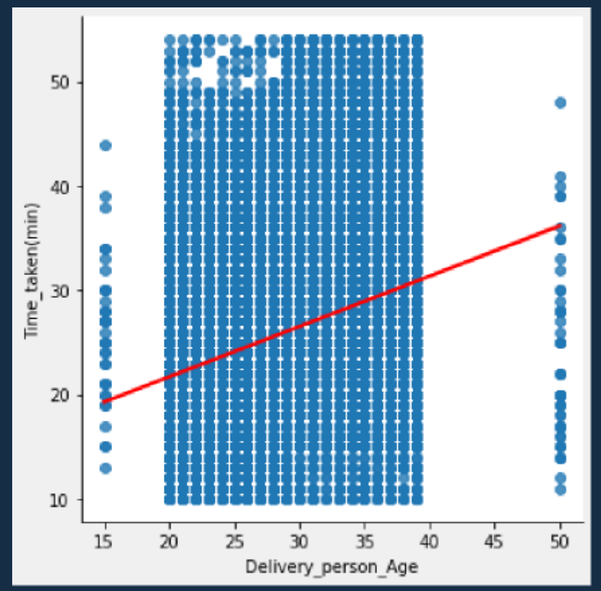
- Delivery_person_Age and Time_taken(min) column have relatively normal data distribution
- Delivery_person_Ratings has negative skew data distribution, meanwhile distance column has positive skew data distribution

Exploratory Data Analysis

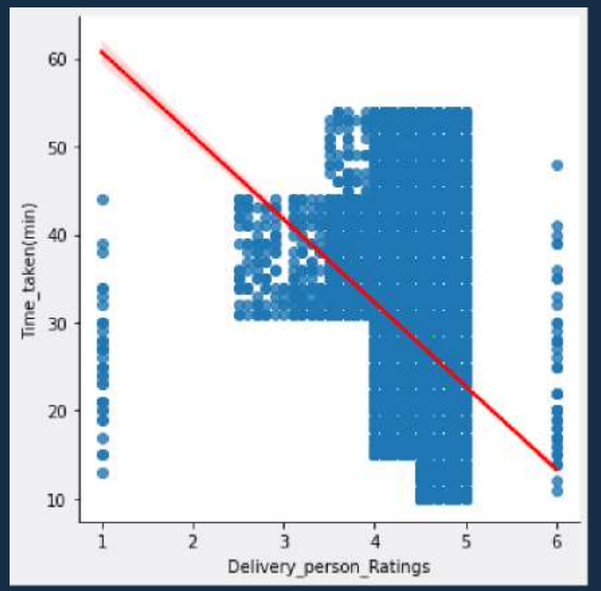
Bivariate Analysis



There is consistent relationship between the time taken and the distance travelled to deliver the food. It looks like majority food delivered within 25-27 minutes regardless of distance



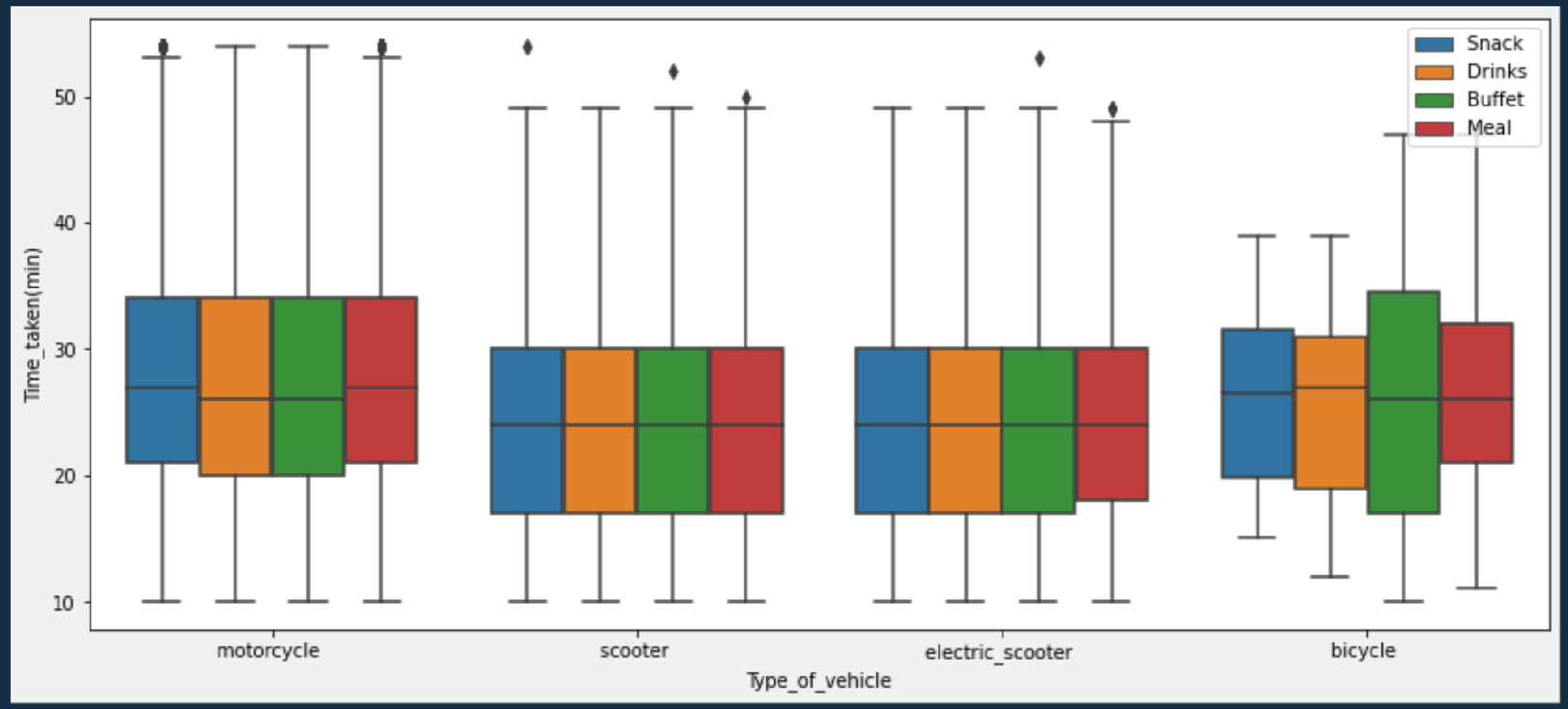
There is a linear relationship between the time taken to deliver the food and the age of the person who delivering the food. It looks like person with the young age able to take less time than person with old age to deliver the food to customers



There is an inverse linear relationship between the time taken to deliver the food and the delivery person ratings. It looks like person with the higher ratings take a less time to deliver the food than person with low ratings

Exploratory Data Analysis

Bivariate Analysis



It looks like there is not much difference between the time taken depending on the vehicle they are driving and the type of food they are delivering

Data Pipeline Architecture

1. AWS Instance Creation:

- Provision an Amazon EC2 instance to serve as the infrastructure for our data pipeline.

2. Zookeeper and Kafka Setup:

- Install and configure Apache Zookeeper for distributed coordination.
- Install and configure Apache Kafka for real-time data streaming.

3. S3 Bucket Creation:

- Create an Amazon S3 bucket to store and manage data.

4.Python Data Loading:

- Develop a Python script to load data into the S3 bucket.
- Utilize the Boto3 library to interact with AWS services programmatically.

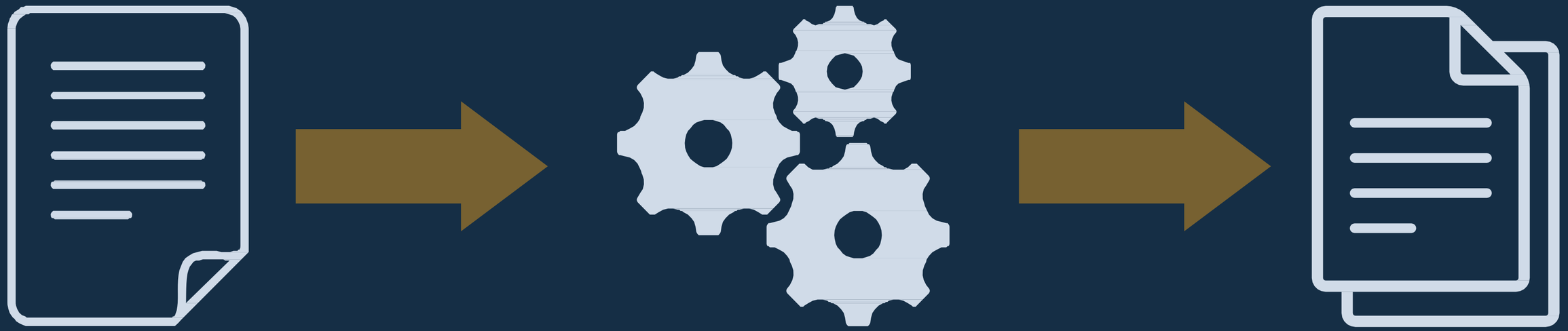
5.Athena Data Querying:

- Set up an Amazon Athena database to query data stored in the S3 bucket using SQL-like queries.
- Define tables and partitions to optimize query performance.

6.Power BI Connection through ODBC:

- Install and configure the Athena ODBC driver on the EC2 instance.
- Set up an ODBC data source to connect Power BI to Amazon Athena.

Feature Engineering



Initial Dataset

Feature Engineering

- Unused Feature Drop
- Outliers Handling
- One Hot Encoding
- Train Test Data Split (70:30)

Final Dataset (Train & Test Data)

Modeling and Evaluation

Algorithms Training

```
LinearRegression:
Best parameters: {}
Best R2 score: 0.42461058199128565

DecisionTreeRegressor:
Best parameters: {'max_depth': 7}
Best R2 score: 0.7338950553018575

RandomForestRegressor:
Best parameters: {'n_estimators': 200}
Best R2 score: 0.8131771162980966

XGBRegressor:
Best parameters: {'max_depth': 9, 'n_estimators': 20}
Best R2 score: 0.81867302298811
```

```
Mean Absolute Error (MAE): 3.17
Mean Squared Error (MSE): 15.95
Root Mean Squared Error (RMSE): 3.99
R-squared (R2) Score: 0.82
```

Based on each **RMSE (Root Squared Mean Error)** result from 4 trained algorithms, best algorithm is XGBRegressor because have the smallest error than other algorithms. Performance of the best model will try to be improved with hyperparameter tuning.

RMSE (Root Squared Mean Error) Formula

$$RMSE = \text{sqrt}\left(\frac{\Sigma(\text{actual} - \text{prediction})^2}{\text{Number of observations}}\right)$$

Modeling and Evaluation

Hyperparameter Tuning

```
param_grid = [
    {},
    {'max_depth': [3, 5, 7]},
    {'n_estimators': [100, 200, 300]},
    {'n_estimators': [20, 25, 30], 'max_depth': [5, 7, 9]},
]
```

Parameters to be tuned

Parameter tune using GridSearch CV

```
XGBRegressor:
Best parameters: {'max_depth': 9, 'n_estimators': 20}
```

Best parameter

Model train using best parameter

```
R-squared (R2) Score: 0.82
```

Final model performance

Conclusion and Recommendation

Conclusion

1. Rating of person in previous delivers become is the most influential factor on the delivery time of food to the destination location. Person with the higher ratings take a less time to deliver the food than person with low ratings
2. Model has RMSE score 7.28 and that means error between delivery time prediction and delivery time actual is 7.28 minutes

Recommendation

The rating obtained by the deliveryman is a representation of the deliveryman's performance in delivering food to the intended location in terms of delivery time. Of course this is a potential loss of customers if this continues to happen. Delivery time performance needs to be maintained so that the rating obtained is high and customer trust can still be maintained.
The RMSE value of the model can be used as a guarantee of delivery time performance which can be given to the customer so that as much as possible the delivery time is not more than the existing RMSE score (delay in delivery time of not more than 7.28 minutes).

Documentation : Github Repository

THANK YOU

+62 878 7375 6695

foood@yahoo.com

fooddeliveryx456