

Food Delivery Time Prediction using LSTM Python

```
In [1]: import pandas as pd
import numpy as np
import plotly.express as px

data = pd.read_csv("deliverytime.txt")

print(data.head())
   ID Delivery_person_ID Delivery_person_Age Delivery_person_Ratings \
0  4607  INDRES130E102          37             4.9
1  B379  BA0RES120E102          34             4.5
2  5000  BANGRES120E101          25             4.4
3  7464  COTMRES130E102          38             4.7
4  7042  CHNRES120E101          32             4.6

   Restaurant_latitude Restaurant_longitude Delivery_location_latitude \
0    22.745809           75.912471            22.765848
1    12.915041           77.681237            13.843041
2    12.914264           77.678400            12.924264
3   11.003669           76.976494            11.053669
4    12.972793           80.249982            13.012793

   Delivery_location_longitude Type_of_order Type_of_vehicle Time_taken(min)
0           75.912471      Snack     motorcycle            24
1           77.681237      Snack     scooter              33
2           77.678400     Drinks     motorcycle            26
3           77.026494     Buffet     motorcycle            21
4           80.249982      Snack     scooter              30
```

Let's have a look at the column insights before moving forward:

```
In [2]: data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 45593 entries, 0 to 45592
Data columns (total 11 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   ID               45593 non-null   object 
 1   Delivery_person_ID 45593 non-null   object 
 2   Delivery_person_Age 45593 non-null   int64  
 3   Delivery_person_Ratings 45593 non-null   float64
 4   Restaurant_latitude 45593 non-null   float64
 5   Restaurant_longitude 45593 non-null   float64
 6   Delivery_location_latitude 45593 non-null   float64
 7   Delivery_location_longitude 45593 non-null   float64
 8   Type_of_order       45593 non-null   object 
 9   Type_of_vehicle     45593 non-null   object 
 10  Time_taken(min)    45593 non-null   int64  
dtypes: float64(5), int64(2), object(4)
memory usage: 3.8+ MB
```

Now let's have a look at whether this dataset contains any null values or not:

```
In [3]: data.isnull().sum()

Out[3]: ID          0
Delivery_person_ID 0
Delivery_person_Age 0
Delivery_person_Ratings 0
Restaurant_latitude 0
Restaurant_longitude 0
Delivery_location_latitude 0
Delivery_location_longitude 0
Type_of_order       0
Type_of_vehicle     0
Time_taken(min)    0
dtype: int64
```

Calculating Distance Between Two Latitudes and Longitudes

The dataset doesn't have any feature that shows the difference between the restaurant and the delivery location. All we have are the latitude and longitude points of the restaurant and the delivery location. We can use the haversine formula to calculate the distance between two locations based on their latitudes and longitudes.

Below is how we can find the distance between the restaurant and the delivery location based on their latitudes and longitudes by using the haversine formula:

```
In [4]: # Set the earth's radius (in kilometers)
R = 6371

# Convert degrees to radians
def deg_to_rad(degrees):
    return degrees * (np.pi/180)

# Function to calculate the distance between two points using the haversine formula
def distcalculate(lat1, lon1, lat2, lon2):
    d_lat = deg_to_rad(lat2-lat1)
    d_lon = deg_to_rad(lon2-lon1)
    a = np.sin(d_lat/2)**2 + np.cos(deg_to_rad(lat1)) * np.cos(deg_to_rad(lat2)) * np.sin(d_lon/2)**2
    c = 2 * np.arctan2(np.sqrt(a), np.sqrt(1-a))
    return R * c

# Calculate the distance between each pair of points
data['distance'] = np.nan

for i in range(len(data)):
    data.loc[i, 'distance'] = distcalculate(data.loc[i, 'Restaurant_latitude'],
                                             data.loc[i, 'Restaurant_longitude'],
                                             data.loc[i, 'Delivery_location_latitude'],
                                             data.loc[i, 'Delivery_location_longitude'])
```

We have now calculated the distance between the restaurant and the delivery location. We have also added a new feature in the dataset as distance. Let's look at the dataset again:

```
In [5]: print(data.head())
   ID Delivery_person_ID Delivery_person_Age Delivery_person_Ratings \
0  4607  INDRES130E102          37             4.9
1  B379  BA0RES120E102          34             4.5
2  5000  BANGRES120E101          25             4.4
3  7464  COTMRES130E102          38             4.7
4  7042  CHNRES120E101          32             4.6

   Restaurant_latitude Restaurant_longitude Delivery_location_latitude \
0    22.745809           75.912471            22.765848
1    12.915041           77.681237            13.843041
2    12.914264           77.678400            12.924264
3   11.003669           76.976494            11.053669
4    12.972793           80.249982            13.012793

   Delivery_location_longitude Type_of_order Type_of_vehicle Time_taken(min)
0           75.912471      Snack     motorcycle            24
1           77.681237      Snack     scooter              33
2           77.678400     Drinks     motorcycle            26
3           77.026494     Buffet     motorcycle            21
4           80.249982      Snack     scooter              30

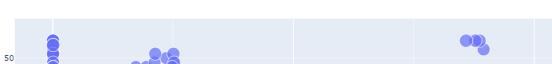
   distance
0  3.025149
1  1.000000
2  1.552758
3  7.798401
4  6.210138
```

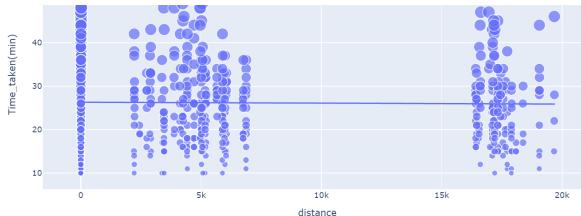
Data Exploration

Now let's explore the data to find relationships between the features. I'll start by looking at the relationship between the distance and time taken to deliver the food:

```
In [6]: figure = px.scatter(data_frame = data,
                        x="distance",
                        y="Time_taken(min)",
                        size="Time_taken(min)",
                        trendline="ols",
                        title = "Relationship Between Distance and Time Taken")
figure.show()
```

Relationship Between Distance and Time Taken





There is a consistent relationship between the time taken and the distance travelled to deliver the food. It means that most delivery partners deliver food within 25 minutes, regardless of distance.

Now let's have a look at the relationship between the time taken to deliver the food and the age of the delivery partner:

```
In [7]: figure = px.scatter(data_frame = data,
x="Delivery_person_Age",
y="Time_taken(min)",
size="Time_taken(min)",
color = "distance",
trendline="ols",
title = "Relationship Between Time Taken and Age")
figure.show()
```

Relationship Between Time Taken and Age



There is a linear relationship between the time taken to deliver the food and the age of the delivery partner. It means young delivery partners take less time to deliver the food compared to the elder partners.

Now let's have a look at the relationship between the time taken to deliver the food and the ratings of the delivery partner:

```
In [8]: figure = px.scatter(data_frame = data,
x="Delivery_person_Ratings",
y="Time_taken(min)",
size="Time_taken(min)",
color = "distance",
trendline="ols",
title = "Relationship Between Time Taken and Ratings")
figure.show()
```

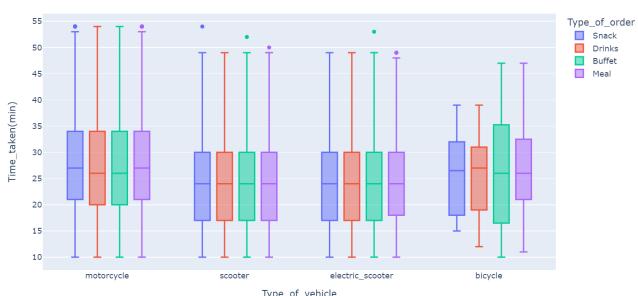
Relationship Between Time Taken and Ratings



There is an inverse linear relationship between the time taken to deliver the food and the ratings of the delivery partner. It means delivery partners with higher ratings take less time to deliver the food compared to partners with low ratings.

Now let's have a look if the type of food ordered by the customer and the type of vehicle used by the delivery partner affects the delivery time or not:

```
In [9]: fig = px.box(data,
x="Type_of_vehicle",
y="Time_taken(min)",
color="Type_of_order")
fig.show()
```



```
In [10]: # pip install tensorflow
```

```
Defaulting to user installation because normal site-packages is not writable
Requirement already satisfied: tensorflow in c:\users\pavan\appdata\roaming\python\python310\site-packages (2.12.0)
Requirement already satisfied: tensorflow-intel==2.12.0 in c:\users\pavan\appdata\roaming\python\python310\site-packages (from tensorflow) (2.12.0)
Requirement already satisfied: tensorflow-metal==2.12.0 in c:\users\pavan\appdata\roaming\python\python310\site-packages (from tensorflow) (2.12.0)
Requirement already satisfied: tensorflow-text==2.12.0 in c:\users\pavan\appdata\roaming\python\python310\site-packages (from tensorflow) (2.12.0)
Requirement already satisfied: tensorflow-addons==0.11.0 in c:\users\pavan\appdata\roaming\python\python310\site-packages (from tensorflow-addons==2.12.0>tensorflow) (0.2.0)
Requirement already satisfied: tensorflow-intel==2.12.0>tensorflow (65.6.3)
Requirement already satisfied: astunparse==1.16.0 in c:\users\pavan\appdata\roaming\python\python310\site-packages (from tensorflow-intel==2.12.0>tensorflow) (1.6.3)
Requirement already satisfied: h5py==2.9.0 in d:\anaconda3\lib\site-packages (from tensorflow-intel==2.12.0>tensorflow) (3.7.0)
Requirement already satisfied: libclang==13.0.0 in c:\users\pavan\appdata\roaming\python\python310\site-packages (from tensorflow-intel==2.12.0>tensorflow) (16.0.0)
```

```
In [13]: import tensorflow as tf  
from tensorflow import keras  
from tensorflow.keras import layers
```

So there is not much difference between the time taken by delivery partners depending on the vehicle they are driving and the type of food they are delivering.

- So the features that contribute most to the food delivery time

 - age of the delivery partner
 - ratings of the delivery partner
 - distance between the restaurant and the delivery location

In the section below, we will learn how to train a Machine Learning model for food delivery time prediction.

Food Delivery Time Prediction Model

Now let's train a Machine Learning model using an LSTM neural network model for the task of food delivery time prediction.

```
In [12]: #splitting data
from sklearn.model_selection import train_test_split
x = np.array(data[["Gender","Person_Age",
                   "Deliver_person_Ratings",
                   "distance"]])
y = np.array(data[["Time_taken(min)"]])
xtrain, xtest, ytrain, ytest = train_test_split(x, y,
                                                test_size=0.10,
                                                random_state=42)

# creating the LSTM neural network model
from keras.models import Sequential
from keras.layers import Dense, LSTM
model = Sequential()
model.add(LSTM(128, return_sequences=True, input_shape=(xtrain.shape[1], 1)))
model.add(LSTM(64, return_sequences=False))
model.add(Dense(25))
model.add(Dense(1))
model.summary()

Model: "sequential"
=====
Layer (type)                 Output Shape              Param #
=====
lstm (LSTM)                  (None, 3, 128)          66560
lstm_1 (LSTM)                (None, 64)               49408
dense (Dense)                (None, 25)              1625
dense_1 (Dense)              (None, 1)               26
=====
Total params: 117,619
Trainable params: 117,619
Non-trainable params: 0
```

```
In [14]: # training the model
model.compile(optimizer='adam', loss='mean_squared_error')
model.fit(xtrain, ytrain, batch_size=1, epochs=9)

Epoch 1/9
41833/41833 [=====] - 136s 3ms/step - loss: 69.12
Epoch 2/9
41833/41833 [=====] - 137s 3ms/step - loss: 63.82
Epoch 3/9
41833/41833 [=====] - 145s 4ms/step - loss: 61.21
Epoch 4/9
41833/41833 [=====] - 137s 3ms/step - loss: 61.10
Epoch 5/9
41833/41833 [=====] - 129s 3ms/step - loss: 60.07
Epoch 6/9
41833/41833 [=====] - 129s 3ms/step - loss: 59.42
Epoch 7/9
41833/41833 [=====] - 149s 4ms/step - loss: 59.37
Epoch 8/9
41833/41833 [=====] - 146s 3ms/step - loss: 59.18
Epoch 9/9
41833/41833 [=====] - 142s 3ms/step - loss: 58.60
```

: <keras.callbacks.History at 0x2521ef70a0>

```
In [15]: print("Food Delivery Time Prediction")
a = int(input("Age of Delivery Partner: "))
b = float(input("Ratings of Previous Deliveries: "))
c = float(input("Total Distance: "))

features = np.array([[a, b, c]])
print("Predicted Delivery Time in Minutes = ", model.predict(features))

Food Delivery Time Prediction
Age of Delivery Partner: 34
Ratings of Previous Deliveries: 4
Total Distance: 8
[1]: [oooooooooooooooooooo] - 1 is 543ms/step
Predicted.Delivery.Time.in.Minutes = [[14.46319461]]
```

So this is how you can use Machine Learning for the task of food delivery time prediction using the Python programming language.