Hello Future Cheetah,

Congratulations on making it to the exercise!

Before you start, please make sure you understand all the questions and email Rae with inquiries before you begin. Please provide code, and images / plots / tables that you generate. You may use any programming languages you choose to, and can send any type of format, such as Jupyter notebook, RStudio notebook, and / or plain text, etc.

**Problem Statement**

Cheetah has to route many trucks (20-ft refrigerated box trucks) in a geographic area that hold multiple orders each day, where each order is delivered in its entirety by one truck / delivery. For each delivery, the driver needs to park the truck (e.g., on a busy street, close or far away from the restaurant), unload the items with a hand-dolly (e.g., one or multiple loads depending on order volume), deliver the items to the restaurant (e.g., maneuvering through tight spaces or a flight of stairs), obtain delivery signatures, and handle refusal / return whenever required.

We want to route our trucks in a way to ensure on-time delivery. To achieve this, we take into account traffic / driving time between stops, as well as the Service Time at each stop. For this exercise, you want to develop a model to predict the Service Time. This time is calculated based on telemetry and time stamps on the app (*completed\_at* - *arrival\_at*).

1. Filter out bad data. How are you defining this bad data? What do you think could be causing this?

Solution: I could see that there are few records having missing arrived time or completion times . Thereby the difference between the completion times and arrival times are -ve and in few cases there are highly positive. This would eventually effect our prediction models. Hence, I dropped those records. The difference of completion\_time – arrival\_time is in hours. This variable is used as output variable.

Secondly, when merged the orders file with the items file and order\_item file I could see that there are few items and its several attributes are missing these records are close to 250. Hence I dropped them as well.

1. Using your good data and information about the orders and restaurants, build a model to predict the service time. Clearly state the input variables for your model.

Solution:

The following are the predictors used in my model.

Chart

Description automatically generated

Text, table

Description automatically generated

1. What are the limitations of the current approach? And what additional data / variables and models would you want to consider for the next iteration of Service Time prediction?

Solution:

The limitations of the current approach are:

Understanding and analyzing the bottlenecks such as queuing time and quantifying these queuing time for each order would gives more closer prediction to the service time.

* 1. Driver/service men performance could be one thing that we can bring into the existing set of attributes.
  2. Brining the weather conditions is most important thing that we may have to consider for the future predictions. As during the snowfall or heavy rain the service would undergo some delay.
  3. Bringing the traffic congestion level into the data would gives us edge in prediction.

1. How many orders for a new market would you need to train a model? How would you solve the cold start problem when we open a new market?

Solution:

1 year data is good enough to predict the service time for the new market with the same set attributes provided in the current data set.

For the cold start problem, understanding the restaurant/food both B2B and B2C market along with capturing the similar attributes from the same geographical locations which we serve nearby along with understanding the competitor’s data should be good start to predict.

Bringing the seasonality factors can play a huge role in the service time prediction for the cold start.

**Data**

In the file orders.csv, you will find order related data, such as dollar value (*sub\_total*), volume in cubic ft (*volume*), weight in pounds (*weight*), and whether the customer is a subscriber at the time of order (*subscription\_id*). In particular, the driver arrives at the timestamp of *arrived\_at* and completes the delivery at the timestamp of *completed\_at*. In addition, you will also find delivery related information, such as *vehicle\_id*, *driver\_id*, and *bringg\_id* (this field can map to delivery\_notes).

In the file delivery\_notes.csv, you will find delivery related data, such as *bringg\_id*, *user\_id* (who is typically the driver), and *note*. In *note*, the user rates his/her feedback on the restaurant location in light of assist, dock (i.e., loading dock), incline (i.e., street incline), parking, pedestrian (i.e., foot traffic), and stair.

In the file restaurants.csv, you will find customer related data, such as the *city* and *zipcode*, as well as the customer segment (*customer\_segment\_name*) - whether the customer is a restaurant (B2B) or a consumer (B2C). B2C customers are more likely to be located in residential areas.

In the file order\_items.csv, you will find more detailed information about each order, specifically the *item\_id* / *sub\_total* / *quantity*. Some of the items can be purchased in each (*packaging\_type* = 0) rather than case (*packaging\_type* = 1). As an example, in one case, there are 15 cartons of eggs (each carton has 12 counts), and the customers can purchase 1 each (e.g., 1 carton) or 1 case (15 cartons).

In the file items.csv, you will find item related data, such as shelf life (*shelf\_life*), temperature zone (*temp\_zone*), net weight per case (*net\_weight*), gross weight per case with packagings (*gross\_weight*), volume in cubic ft (*volume*), and dimension *length* / *width* / *height* in inch. Some items are sold by pounds, such as ground beef. These items are called catch-weight items (*is\_catchweight* = True). The field *units\_type\_name* indicates the number of units in a Pallet, Layer, Case, Each, and LB for catch-weight items. For example, PA(4800)LA(400)CS(40)EA(40)LB(1) means there are 4,800 Lbs per Pallet, 400 Lbs per Layer, 40 Lbs per Case, and 40 Lbs per Each. In other words, there is 1 each per case. PA(480)LA(120)CS(6)EA(1) means there are 480 units per Pallet, 120 units per Layer, 6 units per Case, and 1 unit per Each.

Enjoy the exercise!