Automatidata project

Course 2 - Get Started with Python

Welcome to the Automatidata Project!

You have just started as a data professional in a fictional data consulting firm, Automatidata. Their client, the New York City Taxi and Limousine Commission (New York City TLC), has hired the Automatidata team for its reputation in helping their clients develop data-based solutions.

The team is still in the early stages of the project. Previously, you were asked to complete a project proposal by your supervisor, DeShawn Washington. You have received notice that your project proposal has been approved and that New York City TLC has given the Automatidata team access to their data. To get clear insights, New York TLC's data must be analyzed, key variables identified, and the dataset ensured it is ready for analysis.

A notebook was structured and prepared to help you in this project. Please complete the following questions.

Course 2 End-of-course project: Inspect and analyze data

In this activity, you will examine data provided and prepare it for analysis. This activity will help ensure the information is,

1. Ready to answer questions and yield insights
2. Ready for visualizations
3. Ready for future hypothesis testing and statistical methods

The purpose of this project is to investigate and understand the data provided.

The goal is to use a dataframe contructed within Python, perform a cursory inspection of the provided dataset, and inform team members of your findings.

This activity has three parts:

Part 1: Understand the situation

- Prepare to understand and organize the provided taxi cab dataset and information.

Part 2: Understand the data

- Create a pandas dataframe for data learning, future exploratory data analysis (EDA), and statistical activities.
- Compile summary information about the data to inform next steps.

Part 3: Understand the variables

- Use insights from your examination of the summary data to guide deeper investigation into specific variables.

Follow the instructions and answer the following questions to complete the activity. Then, you will complete an Executive Summary using the questions listed on the PACE Strategy Document.

Be sure to complete this activity before moving on. The next course item will provide you with a completed exemplar to compare to your own work.

Identify data types and relevant variables using Python

PACE stages

Throughout these project notebooks, you'll see references to the problem-solving framework PACE. The following notebook components are labeled with the respective PACE stage: Plan, Analyze, Construct, and Execute.

PACE: Plan

Consider the questions in your PACE Strategy Document and those below to craft your response:

Task 1. Understand the situation

- How can you best prepare to understand and organize the provided taxi cab information?

- Understand the goal or output required for the project - in this case it is to create a dataframe for future analysis for the TLC data to achieve or calculate fares before trip
- REsearch and familiarize with the trip data table shared
- Assess the stakeholder needs and create the dataframe accordingly
- Format the data , create a dataframe using jupyter notebook to inspect and analyse the data and share the initial observations . The next step would be to learn more about the data and check for any anomalies

PACE: Analyze

Consider the questions in your PACE Strategy Document to reflect on the Analyze stage.

Task 2a. Build dataframe

Create a pandas dataframe for data learning, and future exploratory data analysis (EDA) and statistical activities.

Code the following,

- import pandas as `pd` . pandas is used for buidling dataframes.
- import numpy as `np` . numpy is imported with pandas
- `df = pd.read_csv('Datasets\NYC taxi data.csv')`

Note: pair the data object name `df` with pandas functions to manipulate data, such as `df.groupby()` .

Note: As shown in this cell, the dataset has been automatically loaded in for you. You do not need to download the .csv file, or provide more code, in order to access the dataset and proceed with this lab. Please continue with this activity by completing the following instructions.

In [33]:
```
#Import libraries and packages listed above
import pandas as pd
import numpy as np

# Load dataset into dataframe
df = pd.read_csv('2017_Yellow_Taxi_Trip_Data.csv')
print("done")
```

```
done
```

Task 2b. Understand the data - Inspect the data

View and inspect summary information about the dataframe by coding the following:

1. `df.head(10)`
2. `df.info()`
3. `df.describe()`

Consider the following two questions:

Question 1: When reviewing the `df.info()` output, what do you notice about the different variables? Are there any null values? Are all of the variables numeric? Does anything else stand out?

Question 2: When reviewing the `df.describe()` output, what do you notice about the distributions of each variable? Are there any questionable values?

1. from info() output I noticed all the variables have non-null values. The variables are of either int,flaot or Object data type. The date-time values are also object data type - Also noticed , the first Column/variable which stores the trip identification number doesnot have a Column Name. 2.df.describe() returns descriptive statistics of the dataframe, including the minimum, maximum, mean, and percentile values of its numeric columns. There were some negative values for fare amount, total_amount and other payments in 'min' statistics which required further clarification.The maximum fare amount is a much larger value ($1000) than the 25-75 percent range of values. Also, it's questionable

how there are negative values for fare amount. Regarding trip distance, most rides are between 1-3 miles, but the maximum is over 33 miles.

In [34]:
```
df.head(10)
```

Out[34]:

	Unnamed: 0	VendorID	tpep_pickup_datetime	tpep_dropoff_datetime	passenger_count	trip_dista
0	24870114	2	03/25/2017 8:55:43 AM	03/25/2017 9:09:47 AM	6	
1	35634249	1	04/11/2017 2:53:28 PM	04/11/2017 3:19:58 PM	1	
2	106203690	1	12/15/2017 7:26:56 AM	12/15/2017 7:34:08 AM	1	
3	38942136	2	05/07/2017 1:17:59 PM	05/07/2017 1:48:14 PM	1	
4	30841670	2	04/15/2017 11:32:20 PM	04/15/2017 11:49:03 PM	1	
5	23345809	2	03/25/2017 8:34:11 PM	03/25/2017 8:42:11 PM	6	
6	37660487	2	05/03/2017 7:04:09 PM	05/03/2017 8:03:47 PM	1	1
7	69059411	2	08/15/2017 5:41:06 PM	08/15/2017 6:03:05 PM	1	
8	8433159	2	02/04/2017 4:17:07 PM	02/04/2017 4:29:14 PM	1	
9	95294817	1	11/10/2017 3:20:29 PM	11/10/2017 3:40:55 PM	1	

In [35]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 22699 entries, 0 to 22698
Data columns (total 18 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   Unnamed: 0             22699 non-null  int64
 1   VendorID               22699 non-null  int64
 2   tpep_pickup_datetime   22699 non-null  object
 3   tpep_dropoff_datetime  22699 non-null  object
 4   passenger_count        22699 non-null  int64
 5   trip_distance          22699 non-null  float64
 6   RatecodeID             22699 non-null  int64
 7   store_and_fwd_flag     22699 non-null  object
 8   PULocationID           22699 non-null  int64
 9   DOLocationID           22699 non-null  int64
 10  payment_type           22699 non-null  int64
 11  fare_amount            22699 non-null  float64
 12  extra                  22699 non-null  float64
 13  mta_tax                22699 non-null  float64
 14  tip_amount             22699 non-null  float64
 15  tolls_amount           22699 non-null  float64
 16  improvement_surcharge  22699 non-null  float64
 17  total_amount           22699 non-null  float64
dtypes: float64(8), int64(7), object(3)
memory usage: 3.1+ MB
```

In [36]:
```
df.describe()
```

Out[36]:

	Unnamed: 0	VendorID	passenger_count	trip_distance	RatecodeID	PULocationID
count	2.269900e+04	22699.000000	22699.000000	22699.000000	22699.000000	22699.000000
mean	5.675849e+07	1.556236	1.642319	2.913313	1.043394	162.412353
std	3.274493e+07	0.496838	1.285231	3.653171	0.708391	66.633373
min	1.212700e+04	1.000000	0.000000	0.000000	1.000000	1.000000
25%	2.852056e+07	1.000000	1.000000	0.990000	1.000000	114.000000
50%	5.673150e+07	2.000000	1.000000	1.610000	1.000000	162.000000
75%	8.537452e+07	2.000000	2.000000	3.060000	1.000000	233.000000
max	1.134863e+08	2.000000	6.000000	33.960000	99.000000	265.000000

Task 2c. Understand the data - Investigate the variables

Sort and interpret the data table for two variables: `trip_distance` and `total_amount` .

Answer the following three questions:

Question 1: Sort your first variable (`trip_distance`) from maximum to minimum value, do the values seem normal?

Question 2: Sort by your second variable (`total_amount`), are any values unusual?

Question 3: Are the resulting rows similar for both sorts? Why or why not?

1. After analyzing the data sorted from max to min by trip distance, I noticed there were few values that are 0.0(miles) but a total amount was shown. Also as mentioned before longest ride was 33 miles
2.
   - after sorting the data by total_amount in descending , I noticed few of the values had a significantly high fare but the total miles traveled is much less or 0 . The data seemed a bit unusual in these cases
   - The data with bottom 20 values sorted by total_amount , noticed all the bottom 20 values are either negative numbers or 0 even with a trip distance greater than 0.
3. The results in both these sorts are not similar because there are other factors than trip distance that influence total_amt like tip_amount , surcharges, taxes , location, rate codes etc..

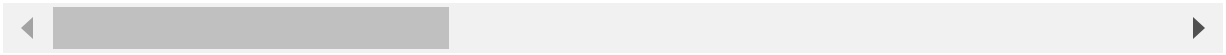
In [37]:
```
sort_tripdistance = df.sort_values(by=['trip_distance'], ascending=False)
sort_tripdistance

# Sort the data by trip distance from maximum to minimum value
```

Out[37]:

	Unnamed: 0	VendorID	tpep_pickup_datetime	tpep_dropoff_datetime	passenger_count	trip_
9280	51810714	2	06/18/2017 11:33:25 PM	06/19/2017 12:12:38 AM	2	
13861	40523668	2	05/19/2017 8:20:21 AM	05/19/2017 9:20:30 AM	1	
6064	49894023	2	06/13/2017 12:30:22 PM	06/13/2017 1:37:51 PM	1	
10291	76319330	2	09/11/2017 11:41:04 AM	09/11/2017 12:18:58 PM	1	
29	94052446	2	11/06/2017 8:30:50 PM	11/07/2017 12:00:00 AM	1	
...	...	...	...	...	...	...
2440	63574825	1	07/26/2017 10:26:58 PM	07/26/2017 10:26:58 PM	1	
15916	47368116	1	06/29/2017 7:30:30 PM	06/29/2017 7:43:29 PM	1	
1350	91619825	2	10/30/2017 8:20:29 AM	10/30/2017 8:20:38 AM	1	
246	78660848	1	09/18/2017 8:50:53 PM	09/18/2017 8:51:03 PM	1	
17788	58079289	1	07/08/2017 12:54:02 AM	07/08/2017 12:55:03 AM	2	

22699 rows × 18 columns



In [38]:
```
sort_totalamt_desc = df.sort_values(by=['total_amount'], ascending=False)
sort_totalamt_desc.head(20)

# Sort the data by total amount and print the top 20 values
```

Out[38]:

	Unnamed: 0	VendorID	tpep_pickup_datetime	tpep_dropoff_datetime	passenger_count	trip_
8476	11157412	1	02/06/2017 5:50:10 AM	02/06/2017 5:51:08 AM	1	
20312	107558404	2	12/19/2017 9:40:46 AM	12/19/2017 9:40:55 AM	2	
13861	40523668	2	05/19/2017 8:20:21 AM	05/19/2017 9:20:30 AM	1	
12511	107108848	2	12/17/2017 6:24:24 PM	12/17/2017 6:24:42 PM	1	
15474	55538852	2	06/06/2017 8:55:01 PM	06/06/2017 8:55:06 PM	1	
6064	49894023	2	06/13/2017 12:30:22 PM	06/13/2017 1:37:51 PM	1	
16379	101198443	2	11/30/2017 10:41:11 AM	11/30/2017 11:31:45 AM	1	
3582	111653084	1	01/01/2017 11:53:01 PM	01/01/2017 11:53:42 PM	1	
11269	51920669	1	06/19/2017 12:51:17 AM	06/19/2017 12:52:12 AM	2	
9280	51810714	2	06/18/2017 11:33:25 PM	06/19/2017 12:12:38 AM	2	
1928	51087145	1	06/16/2017 6:30:08 PM	06/16/2017 7:18:50 PM	2	
10291	76319330	2	09/11/2017 11:41:04 AM	09/11/2017 12:18:58 PM	1	
6708	91660295	2	10/30/2017 11:23:46 AM	10/30/2017 11:23:49 AM	1	
11608	107690629	2	12/19/2017 5:00:56 PM	12/19/2017 6:41:56 PM	2	
908	25075013	2	03/27/2017 1:01:38 PM	03/27/2017 1:38:44 PM	2	
7281	111091850	2	01/01/2017 3:02:53 AM	01/01/2017 3:03:02 AM	1	
18130	90375786	1	10/26/2017 2:45:01 PM	10/26/2017 4:12:49 PM	1	
13621	93330154	1	11/04/2017 1:32:14 PM	11/04/2017 2:18:50 PM	2	
13359	3055315	1	01/12/2017 7:19:36 AM	01/12/2017 7:19:56 AM	1	
29	94052446	2	11/06/2017 8:30:50 PM	11/07/2017 12:00:00 AM	1	

In [39]:
```
sort_totalamt_desc.tail(20)

# Sort the data by total amount and print the bottom 20 values
```

Out[39]:

	Unnamed: 0	VendorID	tpep_pickup_datetime	tpep_dropoff_datetime	passenger_count	trip_
14283	37675840	1	05/03/2017 7:44:28 PM	05/03/2017 7:44:38 PM	1	
19067	58713019	1	07/10/2017 2:40:09 PM	07/10/2017 2:40:59 PM	1	
10506	26005024	2	03/30/2017 3:14:26 AM	03/30/2017 3:14:28 AM	1	
5722	49670364	2	06/12/2017 12:08:55 PM	06/12/2017 12:08:57 PM	1	
4402	108016954	2	12/20/2017 4:06:53 PM	12/20/2017 4:47:50 PM	1	
22566	19022898	2	03/07/2017 2:24:47 AM	03/07/2017 2:24:50 AM	1	
1646	57337183	2	07/05/2017 11:02:23 AM	07/05/2017 11:03:00 AM	1	
18565	43859760	2	05/22/2017 3:51:20 PM	05/22/2017 3:52:22 PM	1	
314	105454287	2	12/13/2017 2:02:39 AM	12/13/2017 2:03:08 AM	6	
5758	833948	2	01/03/2017 8:15:23 PM	01/03/2017 8:15:39 PM	1	
5448	28459983	2	04/06/2017 12:50:26 PM	04/06/2017 12:52:39 PM	1	
4423	97329905	2	11/16/2017 8:13:30 PM	11/16/2017 8:14:50 PM	2	
10281	55302347	2	06/05/2017 5:34:25 PM	06/05/2017 5:36:29 PM	2	
8204	91187947	2	10/28/2017 8:39:36 PM	10/28/2017 8:41:59 PM	1	
20317	75926915	2	09/09/2017 10:59:51 PM	09/09/2017 11:02:06 PM	1	
11204	58395501	2	07/09/2017 7:20:59 AM	07/09/2017 7:23:50 AM	1	
14714	109276092	2	12/24/2017 10:37:58 PM	12/24/2017 10:41:08 PM	5	
17602	24690146	2	03/24/2017 7:31:13 PM	03/24/2017 7:34:49 PM	1	
20698	14668209	2	02/24/2017 12:38:17 AM	02/24/2017 12:42:05 AM	1	
12944	29059760	2	04/08/2017 12:00:16 AM	04/08/2017 11:15:57 PM	1	

In [40]:
```
df.groupby('payment_type').agg({'payment_type':['count']})


# How many of each payment type are represented in the data?
```

Out[40]:

	payment_type
	count
payment_type	
1	15265
2	7267
3	121
4	46

According to the data dictionary, the payment method was encoded as follows:

1 = Credit card
2 = Cash
3 = No charge
4 = Dispute
5 = Unknown
6 = Voided trip

In [41]:
```
# What is the average tip for trips paid for with credit card?

tip_by_creditcard = df['payment_type'] == 1
mask_creditcard = df[tip_by_creditcard]
print(mask_creditcard['tip_amount'].agg(['mean']))

# What is the average tip for trips paid for with cash?
tip_by_cash = df['payment_type'] == 2
mask_cash = df[tip_by_cash]
print(mask_cash['tip_amount'].agg(['mean']))
```

```
mean    2.7298
Name: tip_amount, dtype: float64
mean    0.0
Name: tip_amount, dtype: float64
```

In [42]:
```
df.groupby('VendorID').agg({'VendorID':['count']})

# How many times is each vendor ID represented in the data?
```

Out[42]:

	VendorID
	count
VendorID	
1	10073
2	12626

In [43]:
```
df.groupby('VendorID').agg({'total_amount':['mean']})

# What is the mean total amount for each vendor?
```

Out[43]:

	total_amount
	mean
VendorID	
1	16.298119
2	16.320382

In [44]:
```
data_creditcard = df['payment_type'] == 1
mask_cc = df[data_creditcard]

# Filter the data for credit card payments only

mask_cc['passenger_count'].value_counts()

# Filter the credit-card-only data for passenger count only
```

Out[44]:
```
1    10977
2     2168
5      775
3      600
6      451
4      267
0       27
Name: passenger_count, dtype: int64
```

In [ ]:
```
mask_cc.groupby('passenger_count').agg({'tip_amount':['mean']})

# Calculate the average tip amount for each passenger count (credit card payments only)
```

PACE: Construct

Note: The Construct stage does not apply to this workflow. The PACE framework can be adapted to fit the specific requirements of any project.

PACE: Execute

Consider the questions in your PACE Strategy Document and those below to craft your response.

Given your efforts, what can you summarize for DeShawn and the data team?

Note for Learners: Your notebook should contain data that can address Luana's requests. Which two variables are most helpful for building a predictive model for the client: NYC TLC?

==> After looking at the dataset, the two variables that are most likely to help build a predictive model for taxi ride fares are total_amount and trip_distance because those variables show a picture of a taxi cab ride.

Congratulations! You've completed this lab. However, you may not notice a green check mark next to this item on Coursera's platform. Please continue your progress regardless of the check mark. Just click on the "save" icon at the top of this notebook to ensure your work has been logged.