

Identify data types and relevant variables using Python

Build dataframe

```
In [1]: import pandas as pd          #library exercise for buidling dataframes
import numpy as np                #numpy is imported with pandas

df = pd.read_csv('2017_Yellow_Taxi_Trip_Data.csv')
```

```
In [2]: df.head(10)
```

```
Out[2]:
```

| | Unnamed: 0 | VendorID | tpep_pickup_datetime | tpep_dropoff_datetime | passenger_count | trip_distance | RatecodeID | store_and_fwd_flag | PULocationID | DOLocationID | payment_type | fare_amount | extra | mta_tax | tip_amount | tolls_amount | improvement_surcharge | total_amount |
|---|------------|----------|------------------------|------------------------|-----------------|---------------|------------|--------------------|--------------|--------------|--------------|-------------|-------|---------|------------|--------------|-----------------------|--------------|
| 0 | 24870114 | 2 | 03/25/2017 8:55:43 AM | 03/25/2017 9:09:47 AM | 6 | 3.34 | 1 | N | 100 | 231 | 1 | 13.0 | 0.0 | 0.5 | 2.76 | 0.0 | 0.3 | 16.56 |
| 1 | 35634249 | 1 | 04/11/2017 2:53:28 PM | 04/11/2017 3:19:58 PM | 1 | 1.80 | 1 | N | 186 | 43 | 1 | 16.0 | 0.0 | 0.5 | 4.00 | 0.0 | 0.3 | 20.80 |
| 2 | 106203690 | 1 | 12/15/2017 7:26:56 AM | 12/15/2017 7:34:08 AM | 1 | 1.00 | 1 | N | 262 | 236 | 1 | 6.5 | 0.0 | 0.5 | 1.45 | 0.0 | 0.3 | 8.75 |
| 3 | 38942136 | 2 | 05/07/2017 1:17:59 PM | 05/07/2017 1:48:14 PM | 1 | 3.70 | 1 | N | 188 | 97 | 1 | 20.5 | 0.0 | 0.5 | 6.39 | 0.0 | 0.3 | 27.69 |
| 4 | 30841670 | 2 | 04/15/2017 11:32:20 PM | 04/15/2017 11:49:03 PM | 1 | 4.37 | 1 | N | 4 | 112 | 2 | 16.5 | 0.5 | 0.5 | 0.00 | 0.0 | 0.3 | 17.80 |
| 5 | 23345809 | 2 | 03/25/2017 8:34:11 PM | 03/25/2017 8:42:11 PM | 6 | 2.30 | 1 | N | 161 | 236 | 1 | 9.0 | 0.5 | 0.5 | 2.06 | 0.0 | 0.3 | 12.36 |
| 6 | 37660487 | 2 | 05/03/2017 7:04:09 PM | 05/03/2017 8:03:47 PM | 1 | 12.83 | 1 | N | 79 | 241 | 1 | 47.5 | 1.0 | 0.5 | 9.86 | 0.0 | 0.3 | 59.16 |
| 7 | 69059411 | 2 | 08/15/2017 5:41:06 PM | 08/15/2017 6:03:05 PM | 1 | 2.98 | 1 | N | 237 | 114 | 1 | 16.0 | 1.0 | 0.5 | 1.78 | 0.0 | 0.3 | 19.58 |
| 8 | 8433159 | 2 | 02/04/2017 4:17:07 PM | 02/04/2017 4:29:14 PM | 1 | 1.20 | 1 | N | 234 | 249 | 2 | 9.0 | 0.0 | 0.5 | 0.00 | 0.0 | 0.3 | 9.80 |
| 9 | 95294817 | 1 | 11/10/2017 3:20:29 PM | 11/10/2017 3:40:55 PM | 1 | 1.60 | 1 | N | 239 | 237 | 1 | 13.0 | 0.0 | 0.5 | 2.75 | 0.0 | 0.3 | 16.55 |

```
In [3]: # ==> EXEMPLAR CODE and Output
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 [4]: # ==> EXEMPLAR CODE and OUTPUT
df.describe()
```

```
Out[4]:
```

| | Unnamed: 0 | VendorID | passenger_count | trip_distance | RatecodeID | PULocationID | DOLocationID | payment_type | fare_amount | extra | mta_tax | tip_amount | tolls_amount | improvement_surcharge | total_amount |
|-------|---------------|--------------|-----------------|---------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|-----------------------|--------------|
| count | 2.269900e+04 | 22699.000000 | 22699.000000 | 22699.000000 | 22699.000000 | 22699.000000 | 22699.000000 | 22699.000000 | 22699.000000 | 22699.000000 | 22699.000000 | 22699.000000 | 22699.000000 | 22699.000000 | 22699.000000 |
| mean | 5.675849e+07 | 1.556236 | 1.642319 | 2.913313 | 1.043394 | 162.412353 | 161.527997 | 1.336887 | 13.026629 | 0.333275 | 0.497445 | 1.835781 | 0.312542 | 0.299551 | 16.310502 |
| std | 3.2744493e+07 | 0.496838 | 1.285231 | 3.653171 | 0.708391 | 66.633373 | 70.139691 | 0.496211 | 13.243791 | 0.039465 | 2.800626 | 1.399212 | 0.015673 | 16.097295 | |
| min | 1.212700e+04 | 1.000000 | 0.000000 | 0.000000 | 1.000000 | 1.000000 | 1.000000 | -120.000000 | -1.000000 | -0.500000 | 0.000000 | 0.000000 | -0.300000 | -120.300000 | |
| 25% | 2.852056e+07 | 1.000000 | 1.000000 | 0.990000 | 1.000000 | 114.000000 | 112.000000 | 1.000000 | 6.500000 | 0.000000 | 0.500000 | 0.000000 | 0.000000 | 0.300000 | 8.750000 |
| 50% | 5.673150e+07 | 2.000000 | 1.000000 | 1.610000 | 1.000000 | 162.000000 | 162.000000 | 1.000000 | 9.500000 | 0.000000 | 0.500000 | 1.350000 | 0.000000 | 0.300000 | 11.800000 |
| 75% | 8.537452e+07 | 2.000000 | 2.000000 | 3.060000 | 1.000000 | 233.000000 | 233.000000 | 2.000000 | 14.500000 | 0.500000 | 0.500000 | 2.450000 | 0.000000 | 0.300000 | 17.800000 |
| max | 1.134863e+08 | 2.000000 | 6.000000 | 33.960000 | 99.000000 | 265.000000 | 265.000000 | 4.000000 | 999.990000 | 4.500000 | 0.500000 | 200.000000 | 19.100000 | 0.300000 | 1200.290000 |

```
In [5]: # ==> EXEMPLAR CODE and OUTPUT

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

df_sort = df.sort_values(by=['trip_distance'],ascending=False)
df_sort.head(10)
```

```
Out[5]:
```

| | Unnamed: 0 | VendorID | tpep_pickup_datetime | tpep_dropoff_datetime | passenger_count | trip_distance | RatecodeID | store_and_fwd_flag | PULocationID | DOLocationID | payment_type | fare_amount | extra | mta_tax | tip_amount | tolls_amount | improvement_surcharge | total_amount |
|-------|------------|----------|------------------------|------------------------|-----------------|---------------|------------|--------------------|--------------|--------------|--------------|-------------|-------|---------|------------|--------------|-----------------------|--------------|
| 9280 | 51810714 | 2 | 06/18/2017 11:33:25 PM | 06/19/2017 12:12:38 AM | 2 | 33.96 | 5 | N | 132 | 265 | 2 | 150.00 | 0.0 | 0.0 | 0.00 | 0.00 | 0.3 | 150.30 |
| 13861 | 40523668 | 2 | 05/19/2017 8:20:21 AM | 05/19/2017 9:20:30 AM | 1 | 33.92 | 5 | N | 229 | 265 | 1 | 200.01 | 0.0 | 0.5 | 51.64 | 5.76 | 0.3 | 258.21 |
| 6064 | 49894023 | 2 | 06/13/2017 12:30:22 PM | 06/13/2017 1:37:51 PM | 1 | 32.72 | 3 | N | 138 | 1 | 1 | 107.00 | 0.0 | 0.0 | 55.50 | 16.26 | 0.3 | 179.06 |
| 10291 | 76319330 | 2 | 09/11/2017 11:41:04 AM | 09/11/2017 12:18:58 PM | 1 | 31.95 | 4 | N | 138 | 265 | 2 | 131.00 | 0.0 | 0.5 | 0.00 | 0.00 | 0.3 | 131.80 |
| 29 | 94052446 | 2 | 11/06/2017 8:30:50 PM | 11/07/2017 12:00:00 AM | 1 | 30.83 | 1 | N | 132 | 23 | 1 | 80.00 | 0.5 | 0.5 | 18.56 | 11.52 | 0.3 | 111.38 |
| 18130 | 90375786 | 1 | 10/26/2017 2:45:01 PM | 10/26/2017 4:12:49 PM | 1 | 30.50 | 1 | N | 132 | 220 | 1 | 90.50 | 0.0 | 0.5 | 19.85 | 8.16 | 0.3 | 119.31 |
| 5792 | 68023798 | 2 | 08/11/2017 2:14:01 PM | 08/11/2017 3:17:31 PM | 1 | 30.33 | 2 | N | 132 | 158 | 1 | 52.00 | 0.0 | 0.5 | 14.64 | 5.76 | 0.3 | 73.20 |
| 15350 | 77309977 | 2 | 09/14/2017 1:44:44 PM | 09/14/2017 2:34:29 PM | 1 | 28.23 | 2 | N | 13 | 132 | 1 | 52.00 | 0.0 | 0.5 | 4.40 | 5.76 | 0.3 | 62.96 |
| 10302 | 43431843 | 1 | 05/15/2017 8:11:34 AM | 05/15/2017 9:03:16 AM | 1 | 28.20 | 2 | N | 90 | 132 | 1 | 52.00 | 0.0 | 0.5 | 11.71 | 5.76 | 0.3 | 70.27 |
| 2592 | 51094874 | 2 | 06/16/2017 6:51:20 PM | 06/16/2017 7:41:42 PM | 1 | 27.97 | 2 | N | 261 | 132 | 2 | 52.00 | 4.5 | 0.5 | 0.00 | 5.76 | 0.3 | 63.06 |

```
In [6]: # ==> EXEMPLAR CODE and OUTPUT

# Sort the data by total amount and print the top 20 values
total_amount_sorted = df.sort_values(
    ['total_amount'], ascending=False)['total_amount']
total_amount_sorted.head(20)
```

```
Out[6]:
```

| | |
|-------|---------|
| 8476 | 1200.29 |
| 20312 | 450.30 |
| 13861 | 258.21 |
| 12511 | 233.74 |
| 15474 | 211.80 |
| 6064 | 179.06 |
| 16379 | 157.06 |
| 3582 | 152.30 |
| 11269 | 151.82 |
| 9280 | 150.30 |
| 1928 | 137.80 |
| 10291 | 131.80 |
| 6708 | 126.00 |
| 11608 | 123.30 |
| 908 | 121.56 |
| 7281 | 120.96 |
| 18130 | 119.31 |
| 13621 | 115.94 |
| 13359 | 111.95 |
| 29 | 111.38 |

```
Name: total_amount, dtype: float64
```

```
In [7]: # ==> EXEMPLAR CODE and OUTPUT

# Sort the data by total amount and print the bottom 20 values
total_amount_sorted.tail(20)
```

```
Out[7]:
```

| | |
|-------|---------|
| 14283 | 0.31 |
| 19067 | 0.30 |
| 10506 | 0.00 |
| 5722 | 0.00 |
| 4402 | 0.00 |
| 22566 | 0.00 |
| 1646 | -3.30 |
| 18565 | -3.80 |
| 314 | -3.80 |
| 5758 | -3.80 |
| 5448 | -4.30 |
| 4423 | -4.30 |
| 10281 | -4.30 |
| 8294 | -4.80 |
| 20317 | -4.80 |
| 11204 | -5.30 |
| 14714 | -5.30 |
| 17602 | -5.80 |
| 20698 | -5.80 |
| 12944 | -120.30 |

```
Name: total_amount, dtype: float64
```

```
In [8]: # ==> EXEMPLAR CODE and OUTPUT

# How many of each payment type are represented in the data?
df['payment_type'].value_counts()
```

```
Out[8]:
```

| | |
|---|-------|
| 1 | 15265 |
| 2 | 7267 |
| 3 | 121 |
| 4 | 46 |

```
Name: payment_type, dtype: int64
```

```
In [9]: # ==> EXEMPLAR CODE and OUTPUT

# What is the average tip for trips paid for with credit card?
avg_cc_tip = df[df['payment_type']==1]['tip_amount'].mean()
print('Avg. cc tip:', avg_cc_tip)

# What is the average tip for trips paid for with cash?
avg_cash_tip = df[df['payment_type']==2]['tip_amount'].mean()
print('Avg. cash tip:', avg_cash_tip)

Avg. cc tip: 2.7298001965279934
Avg. cash tip: 0.0
```

```
In [10]: # ==> EXEMPLAR CODE and OUTPUT

# How many times is each vendor ID represented in the data?
df['VendorID'].value_counts()
```

```
Out[10]:
```

| | |
|---|-------|
| 2 | 12626 |
| 1 | 10073 |

```
Name: VendorID, dtype: int64
```

```
In [11]: # ==> EXEMPLAR CODE and OUTPUT

# What is the mean total amount for each vendor?
df.groupby(['VendorID']).mean(numeric_only=True)[['total_amount']]
```

```
Out[11]:
```

| | total_amount |
|----------|--------------|
| VendorID | |
| 1 | 16.298119 |
| 2 | 16.320382 |

```
In [12]: # ==> EXEMPLAR CODE and OUTPUT

# Filter the data for credit card payments only
credit_card = df[df['payment_type']==1]

# Filter the data for passenger count only
credit_card['passenger_count'].value_counts()
```

```
Out[12]:
```

| | |
|---|-------|
| 1 | 10977 |
| 2 | 2168 |
| 5 | 775 |
| 3 | 600 |
| 6 | 451 |
| 4 | 267 |
| 0 | 27 |

```
Name: passenger_count, dtype: int64
```

```
In [13]: # ==> EXEMPLAR CODE and OUTPUT

# Calculate the average tip amount for each passenger count (credit card payments only)
credit_card.groupby(['passenger_count']).mean(numeric_only=True)[['tip_amount']]
```

```
Out[13]:
```

| | tip_amount |
|-----------------|------------|
| passenger_count | |
| 0 | 2.610370 |
| 1 | 2.714681 |
| 2 | 2.829949 |
| 3 | 2.726800 |
| 4 | 2.607753 |
| 5 | 2.762645 |
| 6 | 2.643326 |