# Waze Project

**Course 2 - Get Started with Python**

Welcome to the Waze Project!

Your Waze data analytics team is still in the early stages of their user churn project. Previously, you were asked to complete a project proposal by your supervisor, May Santner. You have received notice that your project proposal has been approved and that your team has been given access to Waze's user data. To get clear insights, the user data must be inspected and prepared for the upcoming process of exploratory data analysis (EDA).

A Python notebook has been prepared to guide you through this project. Answer the questions and create an executive summary for the Waze data team.

# 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

- How can you best prepare to understand and organize the provided 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

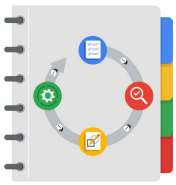
**Part 3:** Understand the variables

- Use insights from your examination of the summary data to guide deeper investigation into 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 compile summary information



## 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 stages: 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 driver data?

*Begin by exploring your dataset and consider reviewing the Data Dictionary.*

By examining the Data Dictionary and identifying variable types, you can quickly determine if the data types are appropriate. Additionally, using functions in languages like Python, R, or others can provide valuable insights into descriptive statistics.

# PACE: Analyze

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

## Task 2a. Imports and data loading

Start by importing the packages that you will need to load and explore the dataset. Make sure to use the following import statements:

- `import pandas as pd`
- `import numpy as np`

In [2]:
```
# Import packages for data manipulation
import numpy as np
import pandas as pd
```

Then, load the dataset into a dataframe. Creating a dataframe will help you conduct data manipulation, exploratory data analysis (EDA), and statistical activities.

**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 [3]:
```
# Load dataset into dataframe
df = pd.read_csv('waze_dataset.csv')
```

## Task 2b. Summary information

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

1. df.head(10)
2. df.info()

*Consider the following questions:*

1. When reviewing the `df.head()` output, are there any variables that have missing values?

2. When reviewing the `df.info()` output, what are the data types? How many rows and columns do you have?

3. Does the dataset have any missing values?

In [4]:
```
df.head()
```

Out[4]:

| | ID | label | sessions | drives | total_sessions | n_days_after_onboarding | total_navigations_ |
|---|---|---|---|---|---|---|---|
| **0** | 0 | retained | 283 | 226 | 296.748273 | 2276 | |
| **1** | 1 | retained | 133 | 107 | 326.896596 | 1225 | |
| **2** | 2 | retained | 114 | 95 | 135.522926 | 2651 | |
| **3** | 3 | retained | 49 | 40 | 67.589221 | 15 | |
| **4** | 4 | retained | 84 | 68 | 168.247020 | 1562 | |

In [5]:
```
df.info()
df.isnull().sum()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 14999 entries, 0 to 14998
Data columns (total 13 columns):
 #   Column                   Non-Null Count  Dtype  
---  ------                   --------------  -----  
 0   ID                       14999 non-null  int64  
 1   label                    14299 non-null  object 
 2   sessions                 14999 non-null  int64  
 3   drives                   14999 non-null  int64  
 4   total_sessions           14999 non-null  float64
 5   n_days_after_onboarding  14999 non-null  int64  
 6   total_navigations_fav1   14999 non-null  int64  
 7   total_navigations_fav2   14999 non-null  int64  
 8   driven_km_drives         14999 non-null  float64
 9   duration_minutes_drives  14999 non-null  float64
 10  activity_days            14999 non-null  int64  
 11  driving_days             14999 non-null  int64  
 12  device                   14999 non-null  object 
dtypes: float64(3), int64(8), object(2)
memory usage: 1.5+ MB
```

Out[5]:
```
ID                           0
label                      700
sessions                     0
drives                       0
total_sessions               0
n_days_after_onboarding      0
total_navigations_fav1       0
total_navigations_fav2       0
driven_km_drives             0
duration_minutes_drives      0
activity_days                0
driving_days                 0
device                       0
dtype: int64
```

1. Inspecting only the output of df.head() does not immediately reveal any missing values.
2. The data types are int64, object, and float64. The dataset contains 13 columns and 14,999 rows.
3. The column 'label' and seems to have missing values.

## Task 2c. Null values and summary statistics

Compare the summary statistics of the 700 rows that are missing labels with summary statistics of the rows that are not missing any values.

**Question:** Is there a discernible difference between the two populations?

In [6]:
```
# Isolate rows with null values
nulls = df[df.isnull().any(axis=1)]
```

```
# Display summary stats of rows with null values
nulls.describe()
```

Out[6]:

| | ID | sessions | drives | total_sessions | n_days_after_onboarding | total_ |
|---|---|---|---|---|---|---|
| **count** | 700.000000 | 700.000000 | 700.000000 | 700.000000 | 700.000000 | |
| **mean** | 7405.584286 | 80.837143 | 67.798571 | 198.483348 | 1709.295714 | |
| **std** | 4306.900234 | 79.987440 | 65.271926 | 140.561715 | 1005.306562 | |
| **min** | 77.000000 | 0.000000 | 0.000000 | 5.582648 | 16.000000 | |
| **25%** | 3744.500000 | 23.000000 | 20.000000 | 94.056340 | 869.000000 | |
| **50%** | 7443.000000 | 56.000000 | 47.500000 | 177.255925 | 1650.500000 | |
| **75%** | 11007.000000 | 112.250000 | 94.000000 | 266.058022 | 2508.750000 | |
| **max** | 14993.000000 | 556.000000 | 445.000000 | 1076.879741 | 3498.000000 | |

In [7]:

```
# Isolate rows without null values

# non_nulls = df.dropna()
non_nulls = df[~df.isnull().any(axis=1)]

# Display summary stats of rows without null values
non_nulls.describe()
```

Out[7]:

| | ID | sessions | drives | total_sessions | n_days_after_onboarding | t |
|---|---|---|---|---|---|---|
| **count** | 14299.000000 | 14299.000000 | 14299.000000 | 14299.000000 | 14299.000000 | |
| **mean** | 7503.573117 | 80.623820 | 67.255822 | 189.547409 | 1751.822505 | |
| **std** | 4331.207621 | 80.736502 | 65.947295 | 136.189764 | 1008.663834 | |
| **min** | 0.000000 | 0.000000 | 0.000000 | 0.220211 | 4.000000 | |
| **25%** | 3749.500000 | 23.000000 | 20.000000 | 90.457733 | 878.500000 | |
| **50%** | 7504.000000 | 56.000000 | 48.000000 | 158.718571 | 1749.000000 | |
| **75%** | 11257.500000 | 111.000000 | 93.000000 | 253.540450 | 2627.500000 | |
| **max** | 14998.000000 | 743.000000 | 596.000000 | 1216.154633 | 3500.000000 | |

Comparing the summary statistics, such as mean, standard deviation, and inter-quartile range, for the data with and without null values, we find a negligible difference. This suggests that the rows containing missing values may have a minimal impact.

## Task 2d. Null values - device counts

Next, check the two populations with respect to the `device` variable.

**Question:** How many iPhone users had null values and how many Android users had null values?

In [8]:
```
# Get count of null values by device
nulls['device'].value_counts()
```

Out[8]:
```
device
iPhone     447
Android    253
Name: count, dtype: int64
```

Of all iPhone and Android users, 447 were iPhone and 253 were Android.

Now, of the rows with null values, calculate the percentage with each device—Android and iPhone. You can do this directly with the `value_counts()` function.

In [9]:
```
# Calculate % of iPhone nulls and Android nulls
nulls['device'].value_counts(normalize=True)
```

Out[9]:
```
device
iPhone     0.638571
Android    0.361429
Name: proportion, dtype: float64
```

How does this compare to the device ratio in the full dataset?

In [10]:
```
# Calculate % of iPhone users and Android users in full dataset
df['device'].value_counts(normalize=True)
```

Out[10]:
```
device
iPhone     0.644843
Android    0.355157
Name: proportion, dtype: float64
```

The percentage of missing values by each device is consistent with their representation in the data overall.

There is nothing to suggest a non-random cause of the missing data.

Examine the counts and percentages of users who churned vs. those who were retained. How many of each group are represented in the data?

In [11]:
```
# Calculate counts of churned vs. retained
print(df['label'].value_counts())
print()
print(df['label'].value_counts(normalize=True))
```

```
label
retained    11763
churned      2536
Name: count, dtype: int64

label
retained    0.822645
churned     0.177355
Name: proportion, dtype: float64
```

This dataset contains 82% retained users and 18% churned users.

Next, compare the medians of each variable for churned and retained users. The reason for calculating the median and not the mean is that you don't want outliers to unduly affect the portrayal of a typical user. Notice, for example, that the maximum value in the `driven_km_drives` column is 21,183 km. That's more than half the circumference of the earth!

In [12]:
```
# Calculate median values of all columns for churned and retained users
df.groupby('label').median(numeric_only=True)
```

Out[12]:

| label | ID | sessions | drives | total_sessions | n_days_after_onboarding | total_navigations |
|---|---|---|---|---|---|---|
| **churned** | 7477.5 | 59.0 | 50.0 | 164.339042 | 1321.0 | |
| **retained** | 7509.0 | 56.0 | 47.0 | 157.586756 | 1843.0 | |

This offers an interesting snapshot of the two groups, churned vs. retained:

Users who churned averaged ~3 more drives in the last month than retained users, but retained users used the app on over twice as many days as churned users in the same time period.

The median churned user drove ~200 more kilometers and 2.5 more hours during the last month than the median retained user.

It seems that churned users had more drives in fewer days, and their trips were farther and longer in duration. Perhaps this is suggestive of a user profile. Continue exploring!

Calculate the median kilometers per drive in the last month for both retained and churned users.

Begin by dividing the `driven_km_drives` column by the `drives` column. Then, group the results by churned/retained and calculate the median km/drive of each group.

In [13]:
```
# Add a column to df called `km_per_drive`
df['km_per_drive'] = df['driven_km_drives'] / df['drives']
```

```
# Group by `label`, calculate the median, and isolate for km per drive
df.groupby('label').median(numeric_only=True)[['km_per_drive']]
```

Out[13]:

| label | km_per_drive |
|---|---|
| **churned** | 74.109416 |
| **retained** | 75.014702 |

The median retained user drove about one more kilometer per drive than the median churned user. How many kilometers per driving day was this?

To calculate this statistic, repeat the steps above using `driving_days` instead of `drives`.

In [14]:
```
# Add a column to df called `km_per_driving_day`
df['km_per_driving_day'] = df['driven_km_drives'] / df['driving_days']

# Group by `label`, calculate the median, and isolate for km per driving day
df.groupby('label').median(numeric_only=True)[['km_per_driving_day']]
```

Out[14]:

| label | km_per_driving_day |
|---|---|
| **churned** | 697.541999 |
| **retained** | 289.549333 |

Now, calculate the median number of drives per driving day for each group.

In [15]:
```
# Add a column to df called `drives_per_driving_day`
df['drives_per_driving_day'] = df['drives'] / df['driving_days']

# Group by `label`, calculate the median, and isolate for drives per driving day
df.groupby('label').median(numeric_only = True)[['drives_per_driving_day']]
```

Out[15]:

| label | drives_per_driving_day |
|---|---|
| **churned** | 10.0000 |
| **retained** | 4.0625 |

The median user who churned drove 698 kilometers each day they drove last month, which is almost ~240% the per-drive-day distance of retained users. The median churned user had a similarly disproporionate number of drives per drive day compared to retained users.

It is clear from these figures that, regardless of whether a user churned or not, the users represented in this data are serious drivers! It would probably be safe to assume that this data does not represent typical drivers at large. Perhaps the data—and in particular the sample of churned users—contains a high proportion of long-haul truckers.

In consideration of how much these users drive, it would be worthwhile to recommend to Waze that they gather more data on these super-drivers. It's possible that the reason for their driving so much is also the reason why the Waze app does not meet their specific set of needs, which may differ from the needs of a more typical driver, such as a commuter.

Finally, examine whether there is an imbalance in how many users churned by device type.

Begin by getting the overall counts of each device type for each group, churned and retained.

In [16]:
```
# For each label, calculate the number of Android users and iPhone users
df.groupby(['label','device']).size()
```

Out[16]:
```
label     device
churned   Android     891
          iPhone     1645
retained  Android    4183
          iPhone     7580
dtype: int64
```

Now, within each group, churned and retained, calculate what percent was Android and what percent was iPhone.

In [17]:
```
# For each label, calculate the percentage of Android users and iPhone users
df.groupby('label')['device'].value_counts(normalize=True)
```

Out[17]:
```
label     device
churned   iPhone     0.648659
          Android    0.351341
retained  iPhone     0.644393
          Android    0.355607
Name: proportion, dtype: float64
```

The ratio of iPhone users and Android users is consistent between the churned group and the retained group, and those ratios are both consistent with the ratio found in the overall dataset.

## 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:

## Task 3. Conclusion

Recall that your supervisor, May Santer, asked you to share your findings with the data team in an executive summary. Consider the following questions as you prepare to write your summary. Think about key points you may want to share with the team, and what information is most relevant to the user churn project.

**Questions:**

1. Did the data contain any missing values? How many, and which variables were affected? Was there a pattern to the missing data?
2. What is a benefit of using the median value of a sample instead of the mean?
3. Did your investigation give rise to further questions that you would like to explore or ask the Waze team about?
4. What percentage of the users in the dataset were Android users and what percentage were iPhone users?
5. What were some distinguishing characteristics of users who churned vs. users who were retained?
6. Was there an appreciable difference in churn rate between iPhone users vs. Android users?

1. There were 700 rows missing data which impacted the label column, however, after investigation, it was determined that these missing values did not have a significant impact on the overall dataset. There did not appear to be a pattern associated with the missing values either.

2. The median value is far less impacted by major outliers and a more robust metric compared to the mean, which is highly influenced by extreme outliers.

3. The investigation reveals that churned and retained users exhibit significantly higher 'km_per_driving_day' and 'drives_per_driving_day' metrics. This suggests that the data may not accurately represent the typical driver population, potentially being skewed by the inclusion of data from long-haul truckers and similar individuals who frequently drive long distances. To ensure a more comprehensive understanding, it would be beneficial to collect additional data that encompasses a wider range of driver profiles. Further investigation should be pursued regarding how the data in this sample was collected.

4. iPhone = 65%, Android = 35%.

5. Although the total kilometers driven by both churned and retained drivers are substantial, churned users have driven significantly more. Their average of 10 'drives_per_driving_day' and 698 'km_per_driving_day' suggests driving habits that considerably surpass those of the average driver.

6. While more iPhone users appear to churn than Android users, it's important to note that the dataset contains a larger proportion of iPhone users. Given this disparity, the observed higher churn rate among iPhone users is not entirely unexpected and therefore does not appear correlated to device type.

**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.