Uber_data_Analysis

December 1, 2024

1 Importing package

[2]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

[3]:
```
#Importing files from CSV
df = pd.read_csv("UberDataset.csv")
df
```

[3]:

	START_DATE	END_DATE	CATEGORY	START	STOP	MILES	PURPOSE
0	01-01-2016 21:11	01-01-2016 21:17	Business	Fort Pierce	Fort Pierce	5.1	Meal/Entertain
1	01-02-2016 01:25	01-02-2016 01:37	Business	Fort Pierce	Fort Pierce	5.0	NaN
2	01-02-2016 20:25	01-02-2016 20:38	Business	Fort Pierce	Fort Pierce	4.8	Errand/Supplies
3	01-05-2016 17:31	01-05-2016 17:45	Business	Fort Pierce	Fort Pierce	4.7	Meeting
4	01-06-2016 14:42	01-06-2016 15:49	Business	Fort Pierce	West Palm Beach	63.7	Customer Visit
…	…	…	…	…	…	…	…
1151	12/31/2016 13:24	12/31/2016 13:42	Business	Kar?chi	Unknown Location	3.9	Temporary Site
1152	12/31/2016 15:03	12/31/2016 15:38	Business	Unknown Location	Unknown Location	16.2	Meeting
1153	12/31/2016 21:32	12/31/2016 21:50	Business	Katunayake	Gampaha	6.4	Temporary Site
1154	12/31/2016 22:08	12/31/2016 23:51	Business	Gampaha	Ilukwatta	48.2	Temporary Site
1155	Totals	NaN	NaN	NaN	NaN	12204.7	NaN

```
[1156 rows x 7 columns]
```

[4]:
```
#Getting information about the types of data and the number of columns
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1156 entries, 0 to 1155
Data columns (total 7 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   START_DATE  1156 non-null   object 
 1   END_DATE    1155 non-null   object 
 2   CATEGORY    1155 non-null   object 
 3   START       1155 non-null   object 
 4   STOP        1155 non-null   object 
 5   MILES       1156 non-null   float64
 6   PURPOSE     653 non-null    object 
dtypes: float64(1), object(6)
memory usage: 63.3+ KB
```

[5]:
```
#The general stats of the data provided
df.describe()
```

[5]:

	MILES
count	1156.000000
mean	21.115398
std	359.299007
min	0.500000
25%	2.900000
50%	6.000000
75%	10.400000
max	12204.700000

[6]:
```
#shape of the dataset
df.shape
```

[6]:
```
(1156, 7)
```

2 Data Preprocessing

[7]:
```
#Where ever the Purpose is unknouwn(NaN), filling it with NOT.
df['PURPOSE'].fillna("NOT",inplace = True)
df.head()
```

```
C:\Users\SK\AppData\Local\Temp\ipykernel_11096\1507545567.py:1: FutureWarning: A
value is trying to be set on a copy of a DataFrame or Series through chained
assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work
```

```
because the intermediate object on which we are setting values always behaves as
a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using
'df.method({col: value}, inplace=True)' or df[col] = df[col].method(value)
instead, to perform the operation inplace on the original object.
```

```
df['PURPOSE'].fillna("NOT",inplace = True)
```

[7]:

	START_DATE	END_DATE	CATEGORY	START	STOP	MILES	PURPOSE
0	01-01-2016 21:11	01-01-2016 21:17	Business	Fort Pierce	Fort Pierce	5.1	Meal/Entertain
1	01-02-2016 01:25	01-02-2016 01:37	Business	Fort Pierce	Fort Pierce	5.0	NOT
2	01-02-2016 20:25	01-02-2016 20:38	Business	Fort Pierce	Fort Pierce	4.8	Errand/Supplies
3	01-05-2016 17:31	01-05-2016 17:45	Business	Fort Pierce	Fort Pierce	4.7	Meeting
4	01-06-2016 14:42	01-06-2016 15:49	Business	Fort Pierce	West Palm Beach	63.7	Customer Visit

[8]:
```
#Changing the format of the START_Date and END_Date to Datetime format and
#also inmaking it permanent.
df['START_DATE'] = pd.to_datetime(df['START_DATE'],errors = 'coerce')
df['END_DATE'] = pd.to_datetime(df['END_DATE'],errors = 'coerce')
df
```

[8]:

	START_DATE	END_DATE	CATEGORY	START	STOP	MILES	PURPOSE
0	2016-01-01 21:11:00	2016-01-01 21:17:00	Business	Fort Pierce	Fort Pierce	5.1	Meal/Entertain
1	2016-01-02 01:25:00	2016-01-02 01:37:00	Business	Fort Pierce	Fort Pierce	5.0	NOT
2	2016-01-02 20:25:00	2016-01-02 20:38:00	Business	Fort Pierce	Fort Pierce	4.8	Errand/Supplies
3	2016-01-05 17:31:00	2016-01-05 17:45:00	Business	Fort Pierce	Fort Pierce	4.7	Meeting
4	2016-01-06 14:42:00	2016-01-06 15:49:00	Business	Fort Pierce	West Palm Beach	63.7	Customer Visit
…	…	…	…	…	…	…	…
1151	NaT	NaT	Business	Kar?chi	Kar?chi	7.1	Errand/Supplies
1152	NaT	NaT	Business	Unknown Location	Unknown Location	0.0	Temporary Site
1153	NaT	NaT	Business	Katunayake	Katunayake	3.2	Meeting
1154	NaT	NaT	Business	Gampaha	Gampaha	6.4	Errand/Supplies
1155	NaT	NaT	NaN	NaN	NaN	30.5	NaN

```
4      West Palm Beach     63.7   Customer Visit
…                  …      …              …
1151  Unknown Location      3.9   Temporary Site
1152  Unknown Location     16.2          Meeting
1153           Gampaha      6.4   Temporary Site
1154         Ilukwatta     48.2   Temporary Site
1155               NaN  12204.7              NOT

[1156 rows x 7 columns]
```

[9]:
```
#Checking the data type of columns after changing it.
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1156 entries, 0 to 1155
Data columns (total 7 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   START_DATE  421 non-null    datetime64[ns]
 1   END_DATE    420 non-null    datetime64[ns]
 2   CATEGORY    1155 non-null   object
 3   START       1155 non-null   object
 4   STOP        1155 non-null   object
 5   MILES       1156 non-null   float64
 6   PURPOSE     1156 non-null   object
dtypes: datetime64[ns](2), float64(1), object(4)
memory usage: 63.3+ KB
```

[10]:
```
#Extracting the Date and Time of the START_DATE for finding the
#solution of the questions.
df['Date'] = pd.DatetimeIndex(df['START_DATE']).date
df['Time'] = pd.DatetimeIndex(df['START_DATE']).hour
df.head()
```

[10]:

	START_DATE	END_DATE	CATEGORY	START	STOP	MILES	PURPOSE	Date	Time
0	2016-01-01 21:11:00	2016-01-01 21:17:00	Business	Fort Pierce	Fort Pierce	5.1	Meal/Entertain	2016-01-01	21.0
1	2016-01-02 01:25:00	2016-01-02 01:37:00	Business	Fort Pierce	Fort Pierce	5.0	NOT	2016-01-02	1.0
2	2016-01-02 20:25:00	2016-01-02 20:38:00	Business	Fort Pierce	Fort Pierce	4.8	Errand/Supplies	2016-01-02	20.0
3	2016-01-05 17:31:00	2016-01-05 17:45:00	Business	Fort Pierce	Fort Pierce	4.7	Meeting	2016-01-05	17.0
4	2016-01-06 14:42:00	2016-01-06 15:49:00	Business	Fort Pierce	West Palm Beach	63.7	Customer Visit	2016-01-06	14.0

[11]:
```
#Marking partitions of the time from 0 to 10 am as Morning,10 to 15 as␣
 ↪afternoon,
#15 to 19 as evening and 19 to 24 as night.
#Mapping these partitions to the time columns of the dataset.
df['Day_Night'] = pd.cut(x=df['Time'],bins = [0,10,15,19,24],
                         labels = ['Morning','Afternoon','Evening','Night'])
df.head()
```

[11]:

	START_DATE	END_DATE	CATEGORY	START	\
0	2016-01-01 21:11:00	2016-01-01 21:17:00	Business	Fort Pierce	
1	2016-01-02 01:25:00	2016-01-02 01:37:00	Business	Fort Pierce	
2	2016-01-02 20:25:00	2016-01-02 20:38:00	Business	Fort Pierce	
3	2016-01-05 17:31:00	2016-01-05 17:45:00	Business	Fort Pierce	
4	2016-01-06 14:42:00	2016-01-06 15:49:00	Business	Fort Pierce	

	STOP	MILES	PURPOSE	Date	Time	Day_Night
0	Fort Pierce	5.1	Meal/Entertain	2016-01-01	21.0	Night
1	Fort Pierce	5.0	NOT	2016-01-02	1.0	Morning
2	Fort Pierce	4.8	Errand/Supplies	2016-01-02	20.0	Night
3	Fort Pierce	4.7	Meeting	2016-01-05	17.0	Evening
4	West Palm Beach	63.7	Customer Visit	2016-01-06	14.0	Afternoon

[12]:
```
#Dropping all the rows which has empty values(Nan)
df.dropna(inplace = True)
#Checking the shape of the dataset after droping Nan values.
df.shape
```

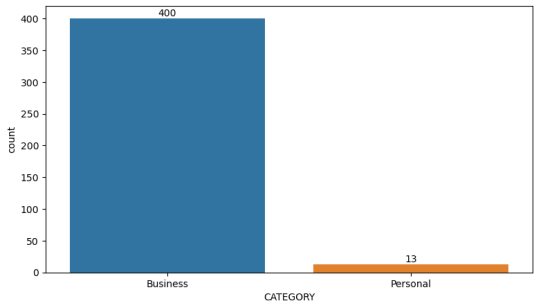
[12]:
```
(413, 10)
```

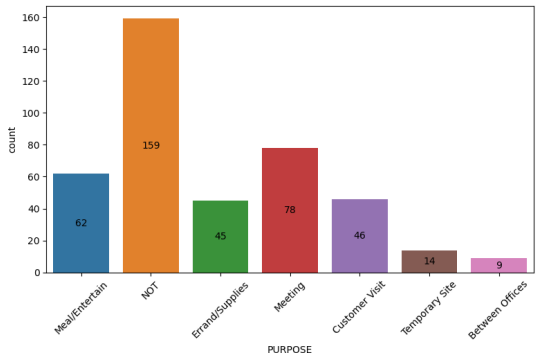
3 Question 1. In which category do people book the most Uber rides?

4 Question 2. For which purpose do people book Uber rides the most?

[29]:
```
#Answer 1: Figure one shows the number of people booked Uber rides for business␣
 ↪and personal use.
plt.figure(figsize = (20,5))
plt.subplot(1,2,1)
ax=sns.countplot(data = df, x = "CATEGORY")
ax.bar_label(ax.containers[0])
#Answer 2: Figure 2 Shows the number of people booked uber for these Purposes.
plt.subplot(1,2,2)
ax2=sns.countplot(data = df,x = "PURPOSE")
ax2.bar_label(ax2.containers[0],label_type = 'center')
plt.xticks(rotation=45)
```

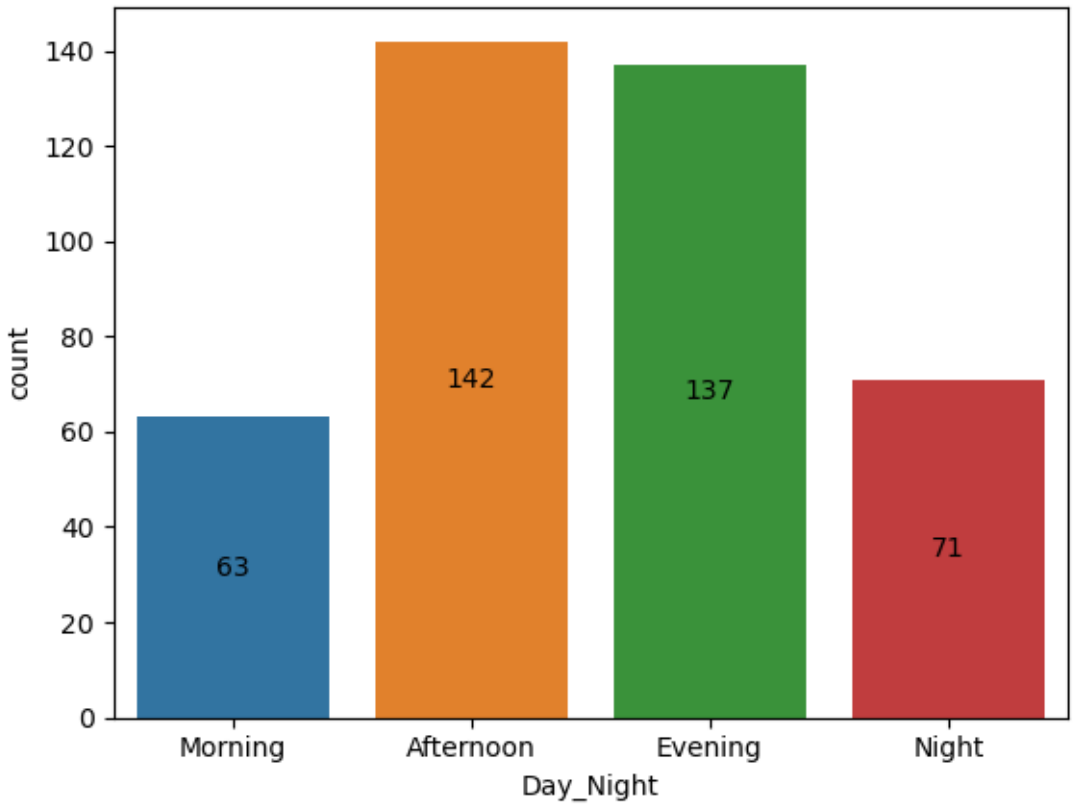
```
plt.show()
```





5 Question 3. At what time do people book cabs the most from Uber?

[33]:
```
# Below figure shows at what time people book Uber rised the most
fig = sns.countplot(data = df, x="Day_Night")
fig.bar_label(fig.containers[0],label_type = 'center')
plt.show()
```



[40]:
```
#Making a new column Month and mapping to the concerned months and
#counting the number of times the nohts appeared
df['Month'] = pd.DatetimeIndex(df['START_DATE']).month
month_label = {1.0:'Jan',2.0:'Feb',3.0:'Mar',4.0:'Apr',
               5.0:'May',6.0:'Jun',7.0:'Jul',8.0:'Aug',
                9.0:'Sept',10.0:'Oct',11.0:'Nov',12.0:'Dec',
               }
df['Month'] = df.Month.map(month_label)
mon = df.Month.value_counts(sort = False)
df.head()
```

[40]:

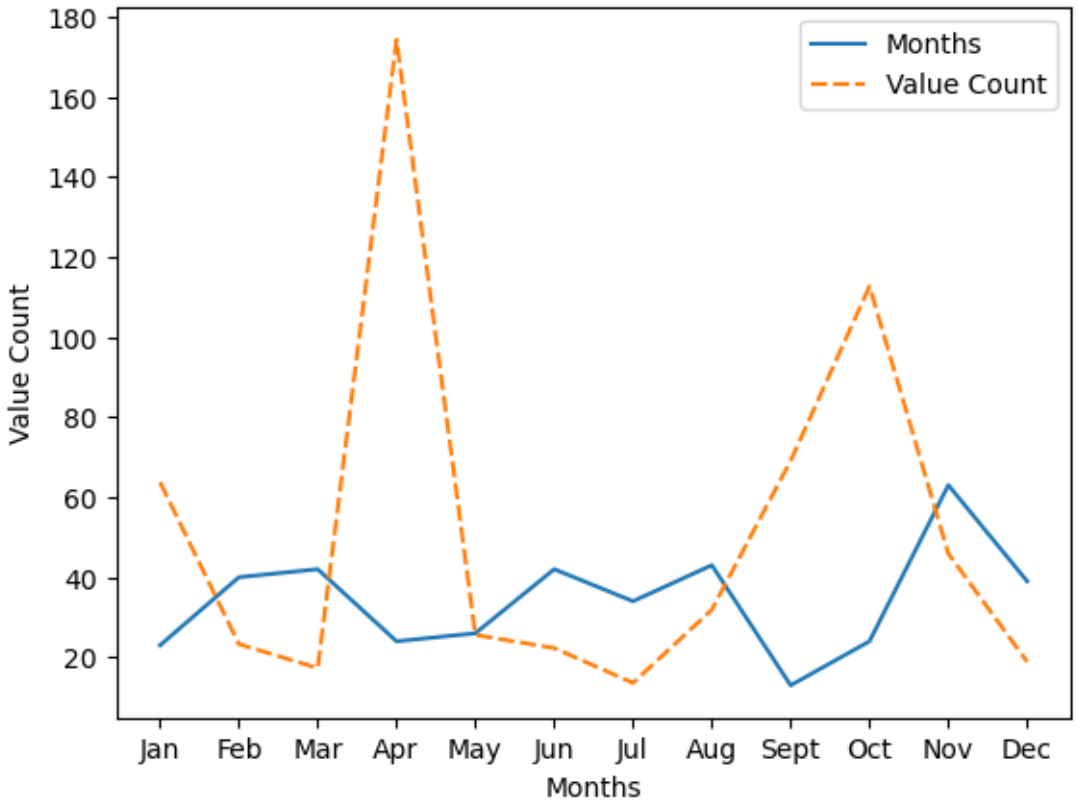
	START_DATE	END_DATE	CATEGORY	START	STOP	MILES	PURPOSE	Date	Time	Day_Night	Month
0	2016-01-01 21:11:00	2016-01-01 21:17:00	Business	Fort Pierce	Fort Pierce	5.1	Meal/Entertain	2016-01-01	21.0	Night	Jan
1	2016-01-02 01:25:00	2016-01-02 01:37:00	Business	Fort Pierce	Fort Pierce	5.0	NOT	2016-01-02	1.0	Morning	Jan
2	2016-01-02 20:25:00	2016-01-02 20:38:00	Business	Fort Pierce	Fort Pierce	4.8	Errand/Supplies	2016-01-02	20.0	Night	Jan
3	2016-01-05 17:31:00	2016-01-05 17:45:00	Business	Fort Pierce	Fort Pierce	4.7	Meeting	2016-01-05	17.0	Evening	Jan
4	2016-01-06 14:42:00	2016-01-06 15:49:00	Business	Fort Pierce	West Palm Beach	63.7	Customer Visit	2016-01-06	14.0	Afternoon	Jan

6 Question 4. In which months do people book Uber rides less frequently?

[45]:
```
#Answer 4: Making a plot to understand on which month there is a declining trend
# Making a line plot with count of ride and and miles covered
df2= pd.DataFrame({
    "Months":mon.values,
    "Value Count":df.groupby('Month',sort = False)['MILES'].max()
})

fig2 = sns.lineplot(data = df2)
fig2.set(xlabel = "Months",ylabel = "Value Count")
plt.show()
```

```
C:\ProgramData\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed in a
future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\ProgramData\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119:
```

```
FutureWarning: use_inf_as_na option is deprecated and will be removed in a
future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



[47]:
```
#Extractign the weekdays and mapping it with the day
df['DAY'] = df.START_DATE.dt.weekday

day_label = { 0:'Sun',1:'Mon',2:'Tues',3:'Wed',4:'Thus',5:'Fri',6:'Sat'
}

df['DAY'] = df['DAY'].map(day_label)
df.head()
```

[47]:

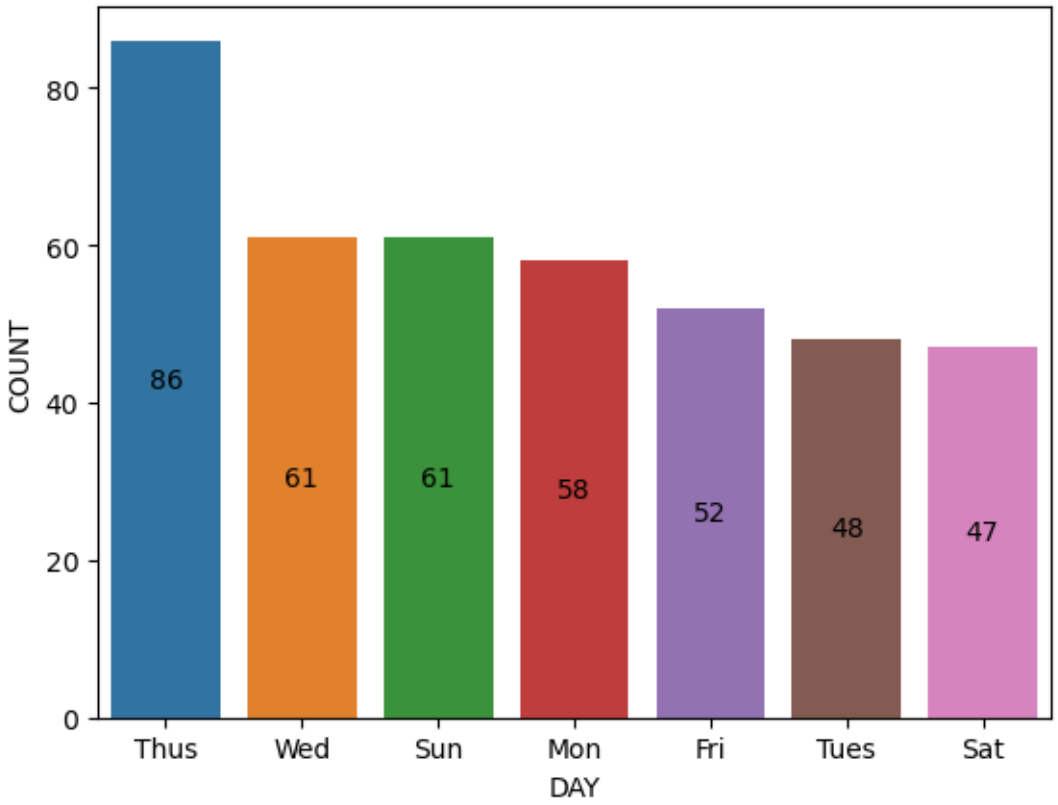
	START_DATE	END_DATE	CATEGORY	START	\
0	2016-01-01 21:11:00	2016-01-01 21:17:00	Business	Fort Pierce	
1	2016-01-02 01:25:00	2016-01-02 01:37:00	Business	Fort Pierce	
2	2016-01-02 20:25:00	2016-01-02 20:38:00	Business	Fort Pierce	
3	2016-01-05 17:31:00	2016-01-05 17:45:00	Business	Fort Pierce	
4	2016-01-06 14:42:00	2016-01-06 15:49:00	Business	Fort Pierce	

	STOP	MILES	PURPOSE	Date	Time	Day_Night	Month	\

```
0      Fort Pierce    5.1   Meal/Entertain  2016-01-01  21.0      Night   Jan
1      Fort Pierce    5.0              NOT  2016-01-02   1.0    Morning   Jan
2      Fort Pierce    4.8  Errand/Supplies  2016-01-02  20.0      Night   Jan
3      Fort Pierce    4.7          Meeting  2016-01-05  17.0    Evening   Jan
4  West Palm Beach   63.7   Customer Visit  2016-01-06  14.0  Afternoon   Jan

    DAY
0  Thus
1   Fri
2   Fri
3   Mon
4  Tues
```

7 Question 5. On which days of the week do people book Uber rides the most?

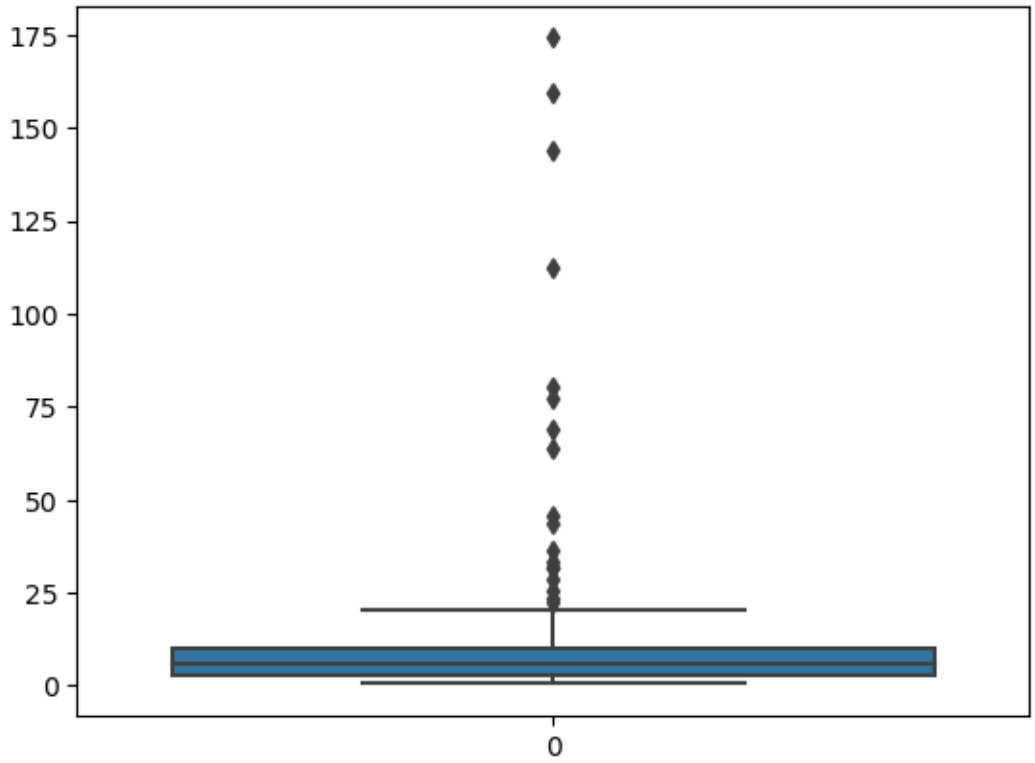
[51]:
```
# Finding the count of ride booked in the weekdays and ploting it againts the␣
 ↪count
day_label = df.DAY.value_counts()

fig3 = sns.barplot(x= day_label.index, y = day_label)
fig3.bar_label(fig3.containers[0],label_type = 'center')
plt.xlabel('DAY')
plt.ylabel('COUNT')
plt.show()
```



8 Question 6. How many miles do people usually book a cab for through Uber?

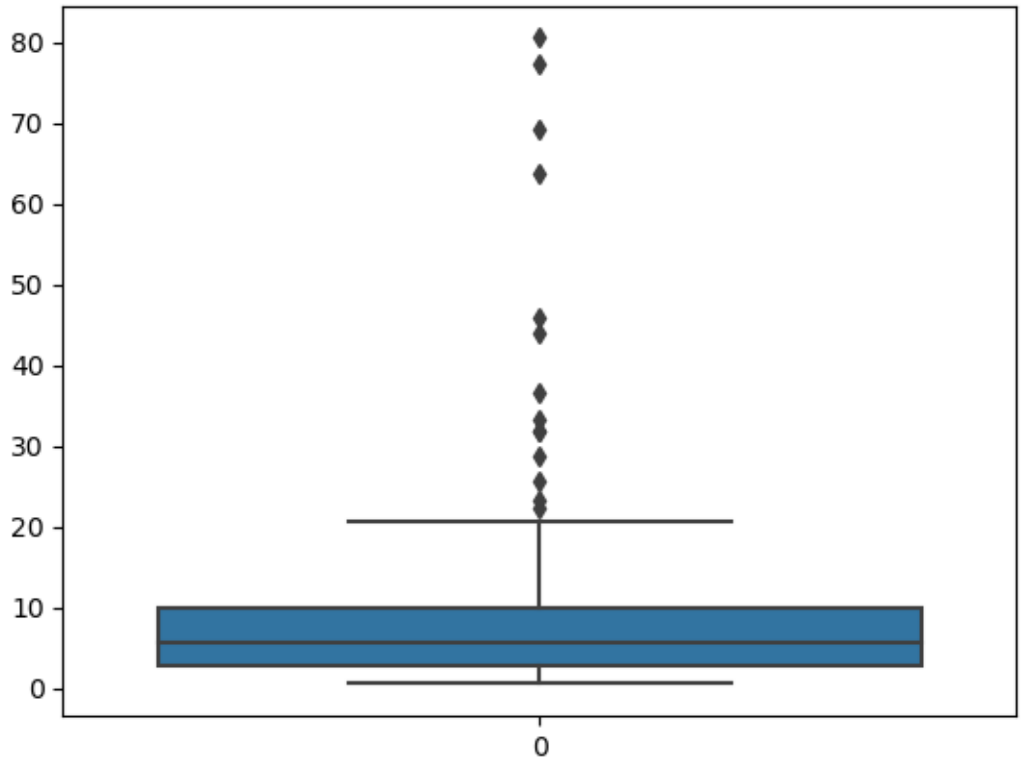
[52]:
```
#Plot the box plot understand the anomalies and the desnity of rides
#with respect to miles.
sns.boxplot(df['MILES'])
```

[52]: <Axes: >



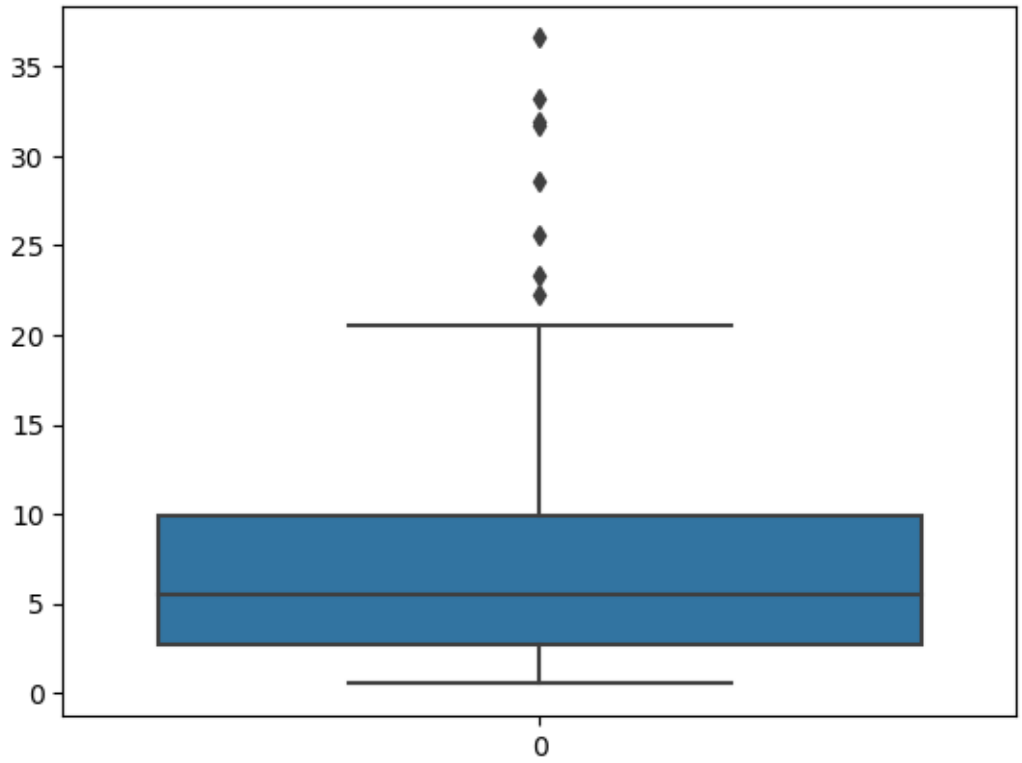
[53]:
```
#Restricting it to 100 miles to understand the density.
sns.boxplot(df[df['MILES']<100]['MILES'])
```

[53]: <Axes: >



[54]:
```
#Restricting it to 40 miles to understand the density.
sns.boxplot(df[df['MILES']<40]['MILES'])
```

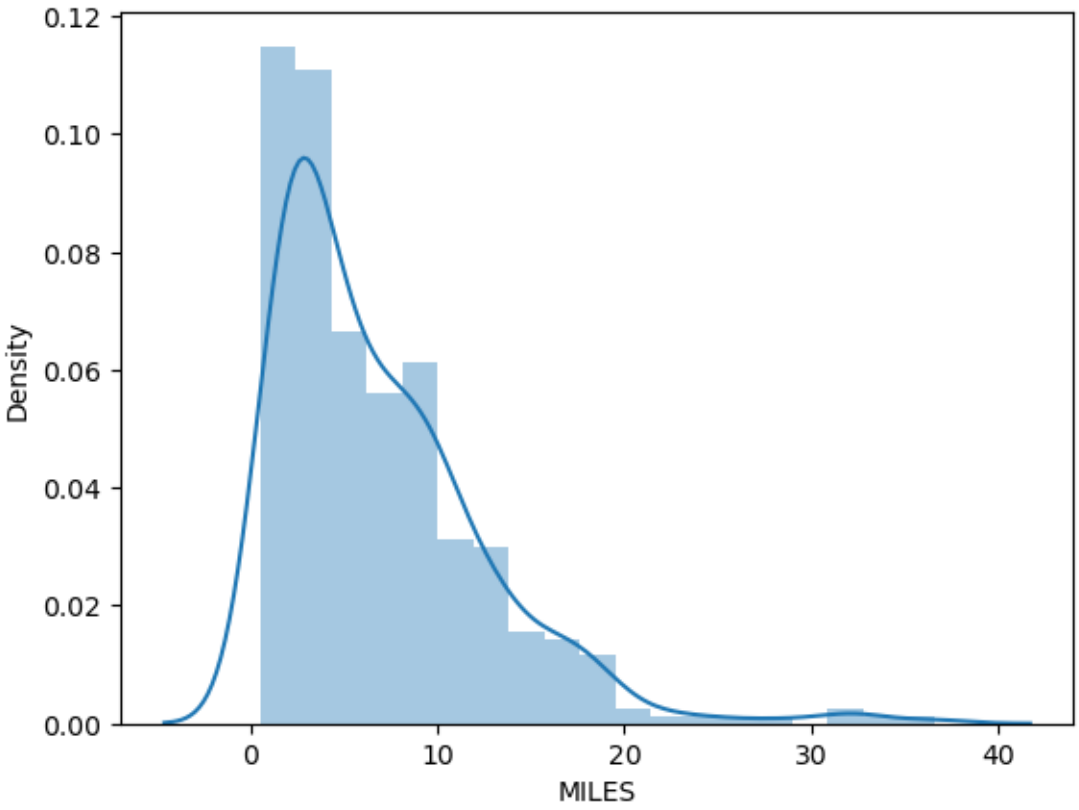
[54]: <Axes: >



[55]:
```
#Using the density chart to better visualize the given senario.
sns.distplot(df[df['MILES']<40]['MILES'])
```

```
C:\Users\SK\AppData\Local\Temp\ipykernel_11096\1171915261.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df[df['MILES']<40]['MILES'])
C:\ProgramData\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed in a
future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```

[55]: `<Axes: xlabel='MILES', ylabel='Density'>`



9 Thank You

[ ]: