EV_Data_Analysis_Project

June 10, 2025

[3]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import ttest_ind
```

[4]:
```
df = pd.read_excel('FEV-data-Excel.xlsx')
df.head()
```

[4]:

	Car full name	Make	Model	Minimal price (gross) [PLN]	Engine power [KM]	Maximum torque [Nm]	Type of brakes	Drive type	Battery capacity [kWh]	Range (WLTP) [km]	…	Permissable gross weight [kg]	Maximum load capacity [kg]	Number of seats	Number of doors	Tire size [in]	Maximum speed [kph]
0	Audi e-tron 55 quattro	Audi	e-tron 55 quattro	345700	360	664	disc (front + rear)	4WD	95.0	438	…	3130.0	640.0				
1	Audi e-tron 50 quattro	Audi	e-tron 50 quattro	308400	313	540	disc (front + rear)	4WD	71.0	340	…	3040.0	670.0				
2	Audi e-tron S quattro	Audi	e-tron S quattro	414900	503	973	disc (front + rear)	4WD	95.0	364	…	3130.0	565.0				
3	Audi e-tron Sportback 50 quattro	Audi	e-tron Sportback 50 quattro	319700	313	540	disc (front + rear)	4WD	71.0	346	…	3040.0	640.0				
4	Audi e-tron Sportback 55 quattro	Audi	e-tron Sportback 55 quattro	357000	360	664	disc (front + rear)	4WD	95.0	447	…	3130.0	670.0				

```
0                5                5               19                  200
1                5                5               19                  190
2                5                5               20                  210
3                5                5               19                  190
4                5                5               19                  200

   Boot capacity (VDA) [l]  Acceleration 0-100 kph [s]  \
0                    660.0                         5.7
1                    660.0                         6.8
2                    660.0                         4.5
3                    615.0                         6.8
4                    615.0                         5.7

   Maximum DC charging power [kW]  mean - Energy consumption [kWh/100 km]
0                             150                                   24.45
1                             150                                   23.80
2                             150                                   27.55
3                             150                                   23.30
4                             150                                   23.85

[5 rows x 25 columns]
```

[11]:
```python
# Task 1: A customer has a budget of 350,000 PLN and wants an EV with a minimum range of 400 km.

# a) Your task is to filter out EVs that meet these criteria.

filter_df = df[(df['Minimal price (gross) [PLN]'] <= 350000) & (df['Range (WLTP) [km]'] >= 400)]


# b) Group them by the manufacturer (Make).

group_by_make = filter_df.groupby('Make')


# c) Calculate the average battery capacity for each manufacturer

avg_battery = group_by_make['Battery capacity [kWh]'].mean().reset_index()
avg_battery.columns = ['Make', 'Avg Battery Capacity']
print(avg_battery)
```

```
           Make  Avg Battery Capacity
0          Audi             95.000000
1           BMW             80.000000
2       Hyundai             64.000000
3           Kia             64.000000
4  Mercedes-Benz             80.000000
```

```
5          Tesla              68.000000
6     Volkswagen              70.666667
```

[23]:
```
# Task 2: You suspect some EVs have unusually high or low energy consumption. Find the
# outliers in the mean - Energy consumption [kWh/100 km] column.

NewEnergy = 'mean - Energy consumption [kWh/100 km]'
Q1 = df[NewEnergy].quantile(0.25)
Q3 = df[NewEnergy].quantile(0.75)
IQR = Q3 - Q1

outliers =df[(df[NewEnergy] < (Q1 - 1.5 * IQR)) | (df[NewEnergy] > (Q3 + 1.5 * IQR))]
print("Outliers in Energy Consumption:\n", outliers[['Car full name', NewEnergy]])
```

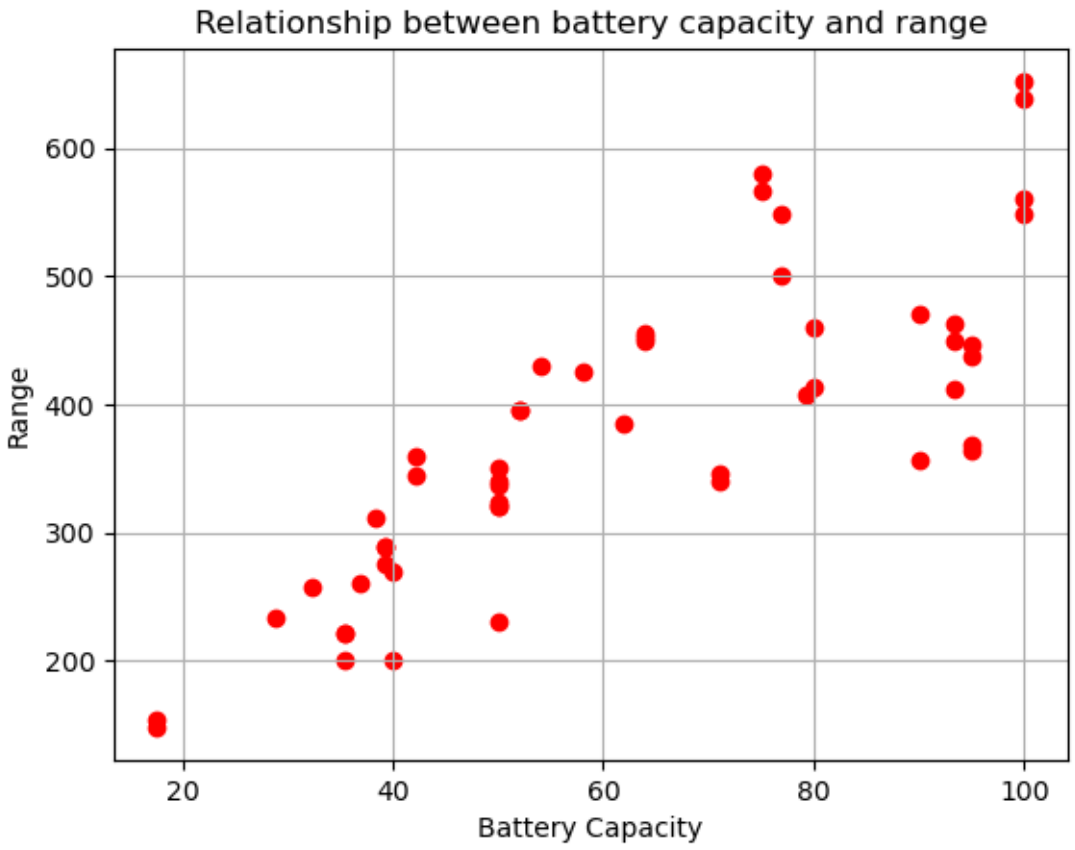
```
Outliers in Energy Consumption:
 Empty DataFrame
Columns: [Car full name, mean - Energy consumption [kWh/100 km]]
Index: []
```

[29]:
```
# Task 3: Your manager wants to know if there's a strong RELATIONSHIP between battery
# capacity and range.
# a) Create a suitable plot to visualize.
# b) Highlight any insights.

import matplotlib.pyplot as plt

plt.scatter(df['Battery capacity [kWh]'],df['Range (WLTP) [km]'],color='red')
plt.title("Relationship between battery capacity and range")
plt.xlabel('Battery Capacity')
plt.ylabel('Range')

plt.grid(True)
plt.show()
print("The majority of points form a diagonal pattern indicating a clear positive correlation between battery capacity and range.")
```



The majority of points form a diagonal pattern indicating a clear positive correlation between battery capacity and range.

[33]:
```
# Task 4: Build an EV recommendation class. The class should allow users to␣
 ↪input their
# budget, desired range, and battery capacity. The class should then return the␣
 ↪top three EVs
# matching their criteria.


class recommendation:
    def __init__(self,df):
        self.df= df




    def EV_rec(self,budget,D_range,B_Capacity):
```

```
        list= self.df[(self.df['Minimal price (gross) [PLN]']<=budget) & (self.
 ↪df['Range (WLTP) [km]']>=D_range) & (self.df['Battery capacity␣
 ↪[kWh]']>=B_Capacity)]
        return list.sort_values(by='Minimal price (gross) [PLN]',␣
 ↪ascending=False).head(3)


r = recommendation(df)
r1 = r.EV_rec(300000, 200, 55)
print("Top 3 EV Recommendations:\n", r1[['Car full name', 'Minimal price␣
 ↪(gross) [PLN]', 'Range (WLTP) [km]','Battery capacity [kWh]']])
```

```
Top 3 EV Recommendations:
                Car full name  Minimal price (gross) [PLN]  Range (WLTP) [km]
\
8                     BMW iX3                       282900                460
41  Tesla Model 3 Performance                       260490                567
40   Tesla Model 3 Long Range                       235490                580

    Battery capacity [kWh]
8                     80.0
41                    75.0
40                    75.0
```

[37]:
```
"""Task 5: Inferential Statistics - Hypothesis Testing: Test whether there is a␣
 ↪significant
difference in the average Engine power [KM] of vehicles manufactured by two␣
 ↪leading
manufacturers i.e. Tesla and Audi. What insights can you draw from the test␣
 ↪results?
Recommendations and Conclusion: Provide actionable insights based on your␣
 ↪analysis.
(Conduct a two sample t-test using ttest_ind from scipy.stats module)"""

from scipy.stats import ttest_ind
t_data = df[df['Make'] == 'Tesla']['Engine power [KM]']
a_data = df[df['Make'] == 'Audi']['Engine power [KM]']

t_stat, p_value = ttest_ind(t_data, a_data, equal_var=False)
print(f"T-Statistic: {t_stat}, P-Value: {p_value}")
if p_value < 0.05:
    print("Conclusion: There is a significant difference in the average engine␣
 ↪power between Tesla and Audi.")
else:
    print("Conclusion:As p value is greater than 0.05.So no significant␣
 ↪difference in the average engine power between Tesla and Audi.")
```

```
T-Statistic: 1.7939951827297178, P-Value: 0.10684105068839565
```

Conclusion:As p value is greater than 0.05.So no significant difference in the average engine power between Tesla and Audi.

[ ]:
```
                                                                # Video Link

            #     https://drive.google.com/file/d/
↪11C5oNSHDWu8IH2bPnYxoHFk2JSToEjyD/view?usp=sharing
```