untitled3

May 5, 2025

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import ttest_ind
```

[2]:
```
df = pd.read_csv("FEV-data-Excel.xlsx.csv")
df.head()
```

[2]:

	Car full name	Make	Model	\
0	Audi e-tron 55 quattro	Audi	e-tron 55 quattro	
1	Audi e-tron 50 quattro	Audi	e-tron 50 quattro	
2	Audi e-tron S quattro	Audi	e-tron S quattro	
3	Audi e-tron Sportback 50 quattro	Audi	e-tron Sportback 50 quattro	
4	Audi e-tron Sportback 55 quattro	Audi	e-tron Sportback 55 quattro	

	Minimal price (gross) [PLN]	Engine power [KM]	Maximum torque [Nm]	\
0	345700	360	664	
1	308400	313	540	
2	414900	503	973	
3	319700	313	540	
4	357000	360	664	

	Type of brakes	Drive type	Battery capacity [kWh]	Range (WLTP) [km]	\
0	disc (front + rear)	4WD	95.0	438	
1	disc (front + rear)	4WD	71.0	340	
2	disc (front + rear)	4WD	95.0	364	
3	disc (front + rear)	4WD	71.0	346	
4	disc (front + rear)	4WD	95.0	447	

	…	Permissable gross weight [kg]	Maximum load capacity [kg]	\
0	…	3130.0	640.0	
1	…	3040.0	670.0	
2	…	3130.0	565.0	
3	…	3040.0	640.0	
4	…	3130.0	670.0	

```
   Number of seats  Number of doors  Tire size [in]  Maximum speed [kph]  \
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]
```

[ ]:
```
#Task 1: Filter EVs by budget and range, then group by manufacturer
Step-by-step:
#a) Filter EVs under 350,000 PLN and range  400 km
```

[3]:
```
filtered_df = df[(df["Minimal price (gross) [PLN]"] <= 350000) & (df["Range␣
↪(WLTP) [km]"] >= 400)]
```

[4]:
```
#Group them by Make
grouped_by_make = filtered_df.groupby("Make")
```

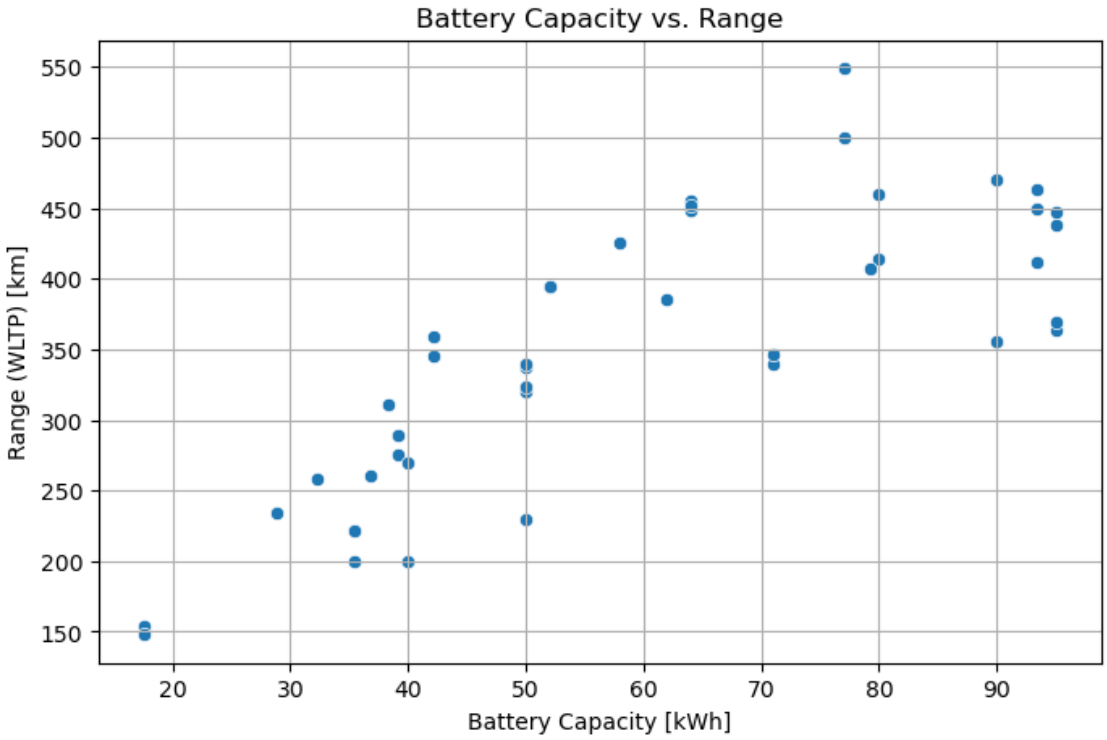
[5]:
```
#Calculate average battery capacity for each manufacturer
avg_battery = grouped_by_make["Battery capacity [kWh]"].mean().reset_index()
print(avg_battery)
```

```
           Make  Battery capacity [kWh]
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
```

[18]:
```
# Use IQR method
#Task 2: Find outliers in mean energy consumption

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

lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

outliers = df[(df['mean - Energy consumption [kWh/100 km]'] < lower_bound) |
              (df['mean - Energy consumption [kWh/100 km]'] > upper_bound)]
print(outliers)
```

```
Empty DataFrame
Columns: [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], Wheelbase [cm], Length [cm], Width [cm], Height [cm], Minimal
empty weight [kg], Permissable gross weight [kg], Maximum load capacity [kg],
Number of seats, Number of doors, Tire size [in], Maximum speed [kph], Boot
capacity (VDA) [l], Acceleration 0-100 kph [s], Maximum DC charging power [kW],
mean - Energy consumption [kWh/100 km]]
Index: []

[0 rows x 25 columns]
```

[19]:
```
plt.figure(figsize=(8,5))
sns.scatterplot(data=df, x="Battery capacity [kWh]", y="Range (WLTP) [km]")
plt.title("Battery Capacity vs. Range")
plt.xlabel("Battery Capacity [kWh]")
plt.ylabel("Range (WLTP) [km]")
plt.grid(True)
plt.show()
```



[ ]:

[26]:
```
##Task 4: EV Recommendation Class
class EVRecommender:
    def __init__(self, data):
        self.data = data

    def recommend(self, budget, min_range, min_battery):
        recommended = self.data[
            (self.data["Minimal price (gross) [PLN]"] <= budget) &
            (self.data["Range (WLTP) [km]"] >= min_range) &
            (self.data["Battery capacity [kWh]"] >= min_battery)
        ]
        return recommended.sort_values(by="Range (WLTP) [km]", ascending=False).
 ↪head(3)

# Usage
recommender = EVRecommender(df)
recommender.recommend(300000, 350, 60)
```

[26]:
```
                Car full name        Make                Model  \
40   Tesla Model 3 Long Range       Tesla   Model 3 Long Range
41  Tesla Model 3 Performance       Tesla  Model 3 Performance
```

```
48      Volkswagen ID.3 Pro S  Volkswagen           ID.3 Pro S

    Minimal price (gross) [PLN]  Engine power [KM]  Maximum torque [Nm]  \
40                       235490                372                  510
41                       260490                480                  639
48                       179990                204                  310

                Type of brakes  Drive type  Battery capacity [kWh]  \
40         disc (front + rear)         4WD                    75.0
41         disc (front + rear)         4WD                    75.0
48  disc (front) + drum (rear)  2WD (rear)                    77.0

    Range (WLTP) [km]  …  Permissable gross weight [kg]  \
40                580  …                           NaN
41                567  …                           NaN
48                549  …                        2280.0

    Maximum load capacity [kg]  Number of seats  Number of doors  \
40                         NaN                5                5
41                         NaN                5                5
48                       412.0                5                5

    Tire size [in]  Maximum speed [kph]  Boot capacity (VDA) [l]  \
40              18                  233                    425.0
41              20                  261                    425.0
48              19                  160                    385.0

    Acceleration 0-100 kph [s]  Maximum DC charging power [kW]  \
40                         4.4                             150
41                         3.3                             150
48                         7.9                             125

    mean - Energy consumption [kWh/100 km]
40                                     NaN
41                                     NaN
48                                    15.9

[3 rows x 25 columns]
```

[27]:
```
#Task 5: Hypothesis Testing (Tesla vs Audi - Engine Power)
#Step-by-step:

tesla_power = df[df["Make"] == "Tesla"]["Engine power [KM]"]
audi_power = df[df["Make"] == "Audi"]["Engine power [KM]"]

t_stat, p_val = ttest_ind(tesla_power, audi_power, equal_var=False)
```

```
print("T-statistic:", t_stat)
print("P-value:", p_val)
```

```
T-statistic: 1.7939951827297178
P-value: 0.10684105068839565
```