STAFFORDSHIRE UNIVERSITY

DECISION ON OPTIMAL MIX OF ELECTRICAL VEHICLES (USING RANDOM FOREST REGRESSOR FOR PERFORMANCE EVALUATION)

SCHOOL OF DIGITAL, TECHNOLOGY AND ARTS

COMPUTER SCIENCE (BUSINESS COMPUTING)

2022/2023 SESSION

MAY, 2023

Introduction

Studies have shown that transportation contributes immensely to air pollution and this has given transporters a great concern (Union of Concerned Scientists, 2018). Combustion of fossil fuels like gasoline and diesel releases greenhouse gases like carbon dioxide, methane, nitrous oxide (N2O), and hydrofluorocarbons (HFCs) into the atmosphere thereby resulting in changes to the climate (US EPA, 2023). Electric vehicles (EVs) have been largely considered as an effective alternative that will reduce these greenhouse gas emissions and consequently address climate change concerns. Adding even much greater impetus to this study is the Russia-Ukraine war that has impacted negatively on the European energy supply chain.

Seeking an alternative to petrol and diesel as fuels for vehicles is one task, and finding the right mix of vehicles running on the alternative energy source is another herculean task. Businesses operating large fleets of vehicles would have to contend with such complex and challenging decision in their choice of the right mix for the EVs. A2Z Limited, a logistics and transport company that operates a fleet of vehicles in Sane Haven, a major city in the United Kingdom is interested in transitioning to electric vehicles but is unsure about the best mix of EVs to purchase. The company is considering several types of electric vehicles, including sedans, SUVs, and trucks. In addition, the company is also considering certain factors like prices, acceleration, efficiency, powertrain, range, fast charge, top speed, and number of seats, popularity among its target customers and return on investment in the choice of its fleet.

The objectives of this analysis were therefore to:

1. Evaluate the performance of different types of electric vehicles in the context of the stated requirements.
2. Determine the optimal electric vehicle mix for the company, based on a trade-off between cost, range, fast charge, top speed, efficiency, acceleration, power train, and number of seats.

Data Cleaning and Preprocessing

Data set for this analysis was an electric vehicles data obtained from Kaggle. The dataset contains information on the characteristics and performance of each type of electric vehicle comprising 134 rows and 18 columns with object and float data types (Appendix I). Attributes of the datasets include but not limited to brand, model, acceleration, efficiency, fast charge, top speed, range, number of seats, price, power train.

First, Python modules were imported to Jupyter notebook before the electric vehicle dataset was imported using appropriate Python modules like pandas (pd.read_csv()). The dataset was then converted to a dataframe (pd.DataFrame()) inspected and was observed to be replete with some missing or empty values. The df.dropna() method was applied in deleting the null values (Appendix II). Some numeric columns were carrying units or symbols not appropriate or allowable for arithmetic operations in Python. The df[‘column name’].str.replace (‘string’, ‘’) method was used to remove them and then transform them from strings to float data type using the df[‘column name’].astype(float).

Descriptive Statistics and EDA

The descriptive statistics for each numeric attribute in the data across board is as shown below:

Table 1: Summary statistics

	Accelaration	TopSpeed	Range	Efficiency	FastCharge	Seats	Price
count	98.00	98.00	98.00	98.00	98.00	98.00	98.00
mean	7.05	181.65	350.15	189.87	456.73	4.96	57324.68
std	2.48	43.25	118.22	30.06	201.26	0.69	34288.25
min	2.10	123.00	170.00	104.00	170.00	4.00	20129.00
25%	5.10	150.00	258.75	168.00	275.00	5.00	35000.00
50%	7.30	167.00	350.00	181.00	440.00	5.00	45000.00
75%	8.95	200.00	407.50	206.00	560.00	5.00	65465.00
max	14.00	410.00	970.00	273.00	940.00	7.00	215000.00

The range of values (max-min) in the price column is quite wide, hence, its large standard deviation. This is in contrast to the acceleration and number of seats having standard deviation of 2.48 and 0.69 respectively. From the figure above, it is clear that the price column contains outliers. A few outliers were also noted in other columns except in acceleration. Across board, average values of acceleration, top speed, range, efficiency, fast charge, seats and price were respectively 7.05 seconds, 181.65 km/hr, 350.15 km, 189.87, 456.73, 5, and 57324.68 euros respectively.

There are three forms of power train captured in the data, namely rear wheel drive (RWD), front wheel drive (FWD), and all-wheel drive (AWD). Powertrain refers to the system that powers the electric vehicle and could be an essential consideration in the choice of electric vehicles. Powertrain directly affects the acceleration, efficiency, and range of the electric vehicle.

The table below is a dataframe showing the distribution of the vehicle model by power train.

Table2: Vehicle distribution, model by power train

	PowerTrain	Model
0	All Wheel Drive	41
1	Front Wheel Drive	35
2	Rear Wheel Drive	21

What this implies is that most vehicles were all wheel drive (AWD) constituting about 42%, followed by front wheel drive (FWD) with 36%, and then rear wheel drive (RWD) with 22% as shown in the pie chart below.

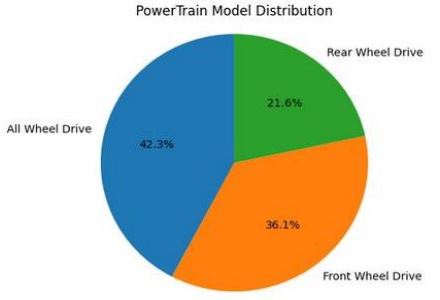


Figure 1: Percentage distribution of power train

From the table below, people seem to prefer AWD vehicles. AWD has many positives, from accelerating in a few seconds, long ranges, fast charging time, high top speed, to high efficiency. However, these vehicles are sold at premium but how then do they still enjoy great patronage? Granularity may explain whether or not every member of that category is expensive. If that is the case, why would anyone go that extra mile to spend that much on vehicles? Perhaps, expanding the context may explain that. For example, if socio-economic and demographic data are brought into the picture, the puzzle may well have been solved. Could these data have been a mere coincidence? This explanation also applies to Rear Wheel Drive and Front Wheel Drive categories. The table shows that RWD vehicles have features of higher quality than those in the FWD category but it seems people were much more swayed by the prices than those features, hence, higher patronage of FWD vehicles. Again, more data is needed to broaden the context so as to gain greater insight.

Table 3: Feature averages by power train

PowerTrain	Accelaration	TopSpeed	Range	Efficiency	FastCharge	Seats	Price
All Wheel Drive	5.0	217.0	426.0	207.0	606.0	5.0	83840.0
Rear Wheel Drive	8.0	168.0	336.0	182.0	446.0	5.0	43274.0
Front Wheel Drive	9.0	149.0	271.0	175.0	293.0	5.0	35323.0

The column chart below indicates the distribution of the vehicle models per brand. Tesla leads with 13 models, followed by Audi with 9, Nissan and Volkswagen with 8. Skoda has 6, while Kia and Porsche have 5 each, and BMW and Ford have 4 each. Opel, Byton, Renault, Mercedes and Hyundai have 3 each. Fiat, Peugeot, and Honda have 2 each, while the remaining brands have 1 model each. The emergence of Tesla as having the highest number of models underscore the fact established earlier about the demographics of EV users, together with their socioeconomic status. Vehicle brands may soon evolve into a major factor for consideration of consumer preference.

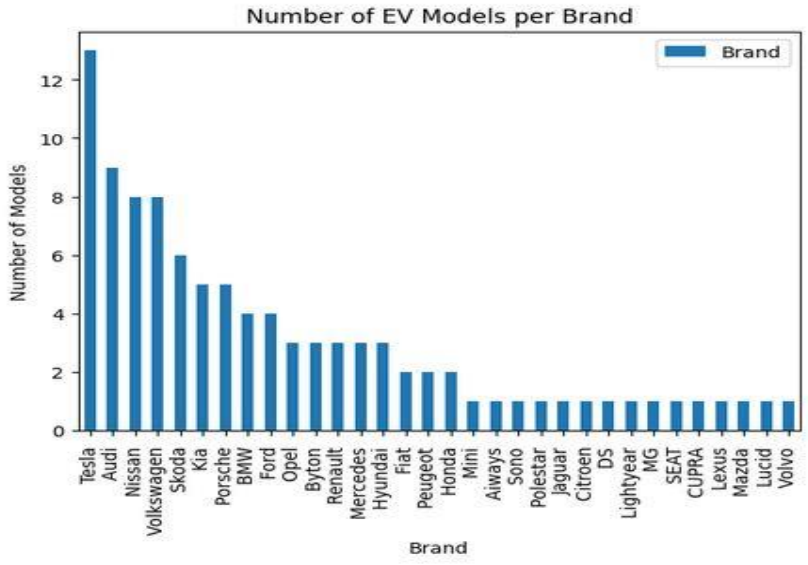


Figure 2: Number of models by brands

Outliers were observed as indicated by the boxplot and histogram subplots. This phenomenon occurred largely with the price attribute. This was, however, expected considering that a few of the vehicles were valued at cut-throat prices, while others were mid-range and low-class vehicles. Hence, discretion was applied in handling the outliers.

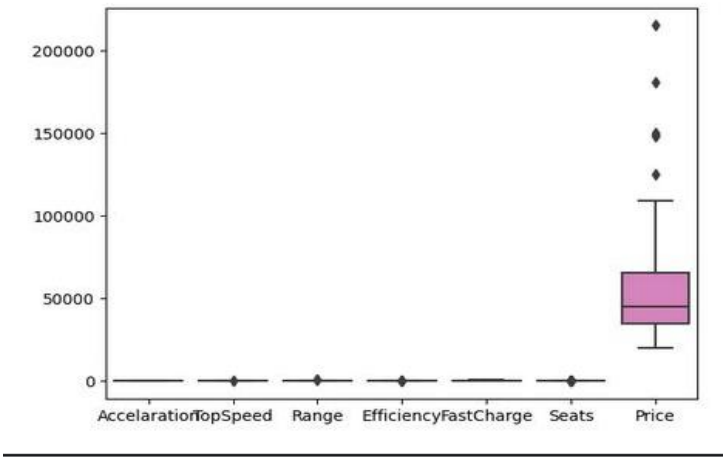


Figure 3: Boxplot indicating distribution of features

Outliers were also found in the number of seats, price, range, efficiency, and top speed. For price, range, seats, and top speed, the histograms were right-skewed, meaning that there were vehicles with extreme values that were either advantageous or disadvantageous, depending on what perspective it was viewed from. For prices and acceleration being right-skewed, vehicles found in such categories may not enjoy wide patronage due to their exorbitant prices and slow acceleration rate. But for the vehicles in the category of range, top speed, seats being right-skewed there would be wide patronage which can only be marred by high prices.

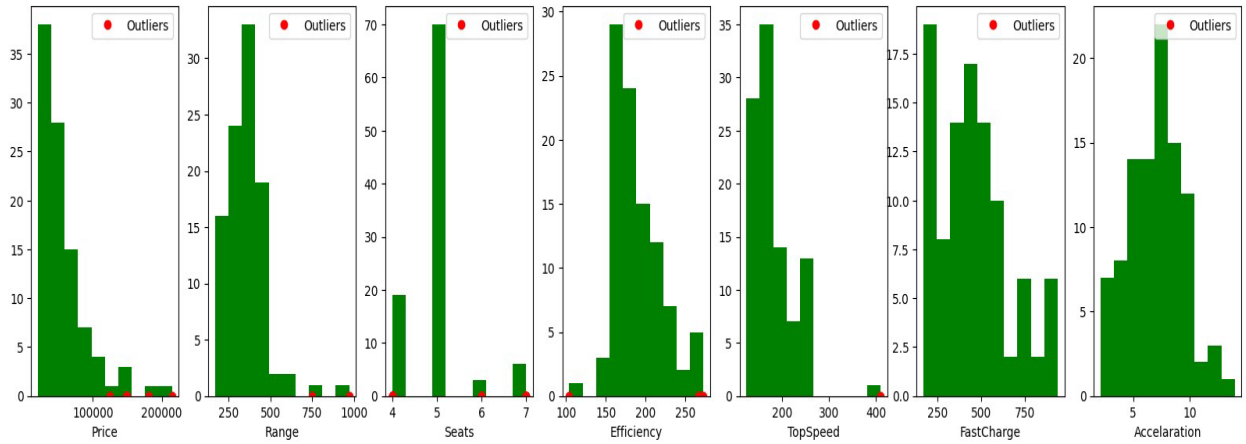


Figure 4: Outliers and their distributions across the features

Outliers largely affect the mean, standard deviation, and median and that may create unnecessary errors in the analysis, hence, the need to normalize the data. Normalization is the process of transforming data to a standard form for easy interpretation by machine learning algorithms. It implies that data having various scales, magnitude or dimensions can be transformed to a common range or scale. There are various forms of normalization but for this project, the MinMax Scaler was used for the normalization. The MinMax Scaler works on the principle of transforming the input to a specified range usually between 0 and 1. To achieve that, the numerical columns were extracted from the rest of the data set and "dataframed" as num_df, short for numeric dataframe as shown below. Columns for the transformation were also called and dataframed as num_cols. The num_cols, together with the numeric dataframe, num_df, was passed into the MinMax Scaler algorithm.

Table 4: Columns for the normalization

```
['Accelaration',
 'TopSpeed',
 'Range',
 'Efficiency',
 'FastCharge',
 'Seats',
 'Price']
```

Table 5: Extracted numerical values for further analysis

	Accelaration	TopSpeed	Range	Efficiency	FastCharge	Seats	Price
0	4.6	233.0	450.0	161.0	940.0	5.0	55480.0
1	10.0	160.0	270.0	167.0	250.0	5.0	30000.0
2	4.7	210.0	400.0	181.0	620.0	5.0	56440.0
3	6.8	180.0	360.0	206.0	560.0	5.0	68040.0
4	9.5	145.0	170.0	168.0	190.0	4.0	32997.0
...	...	...	...	...	...	...	...
98	7.5	160.0	330.0	191.0	440.0	5.0	45000.0
99	4.5	210.0	335.0	258.0	540.0	5.0	96050.0
100	5.9	200.0	325.0	194.0	440.0	5.0	50000.0
101	5.1	200.0	375.0	232.0	450.0	5.0	65000.0
102	7.5	190.0	400.0	238.0	480.0	5.0	62000.0

98 rows × 7 columns

The transformed or normalized data is as shown in the table below, with values ranging between 0 and 1. The normalized data has been built into a new dataframe called norm_df. This norm_df shall be used for correlational analysis and machine learning of the data.

Table 6: Normalized value for further test

	Accelaration	TopSpeed	Range	Efficiency	FastCharge	Seats	Price
0	0.210084	0.383275	0.35000	0.337278	1.000000	0.333333	0.181407
1	0.663866	0.128920	0.12500	0.372781	0.103896	0.333333	0.050654
2	0.218487	0.303136	0.28750	0.455621	0.584416	0.333333	0.186334
3	0.394958	0.198606	0.23750	0.603550	0.506494	0.333333	0.245860
4	0.621849	0.076655	0.00000	0.378698	0.025974	0.000000	0.066033
...	...	...	...	...	...	...	...
93	0.453782	0.128920	0.20000	0.514793	0.350649	0.333333	0.127628
94	0.201681	0.303136	0.20625	0.911243	0.480519	0.333333	0.389596
95	0.319328	0.268293	0.19375	0.532544	0.350649	0.333333	0.153286
96	0.252101	0.268293	0.25625	0.757396	0.363636	0.333333	0.230260
97	0.453782	0.233449	0.28750	0.792899	0.402597	0.333333	0.214865

98 rows × 7 columns

Evaluating the performance of different types of electric vehicles

This is the first objective of this project. The performance evaluation was based on the features of the vehicles, namely acceleration, range, fast charge, top speed, number of seats, efficiency and price. These features were further divided into two categories, make features comprising acceleration, top speed, and price, and battery features comprising efficiency, fast charge, and range for easy visualization.

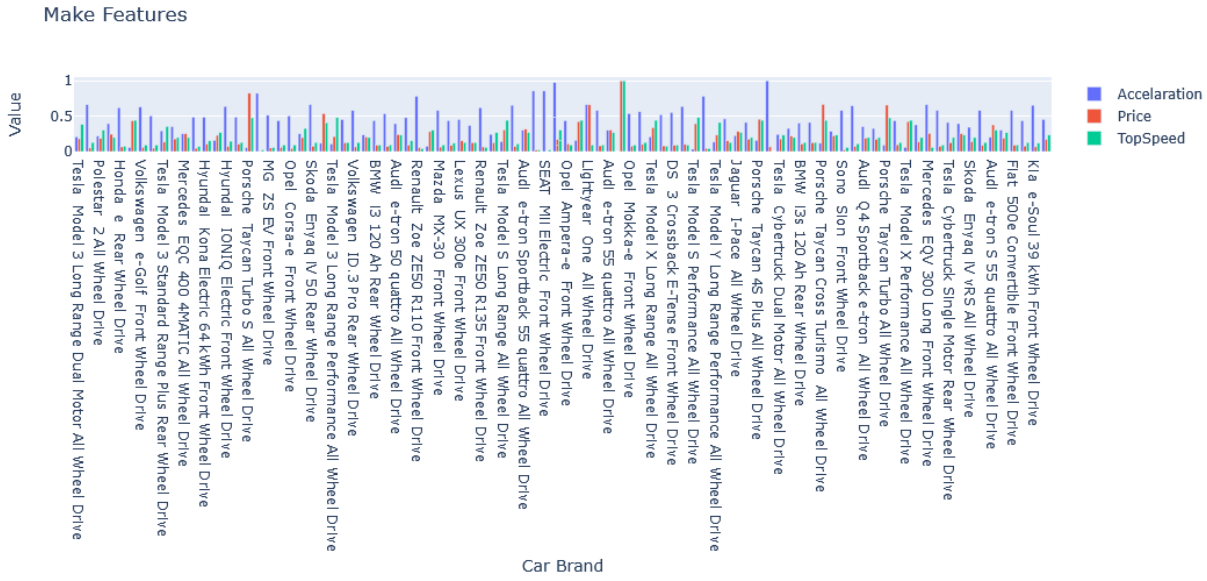
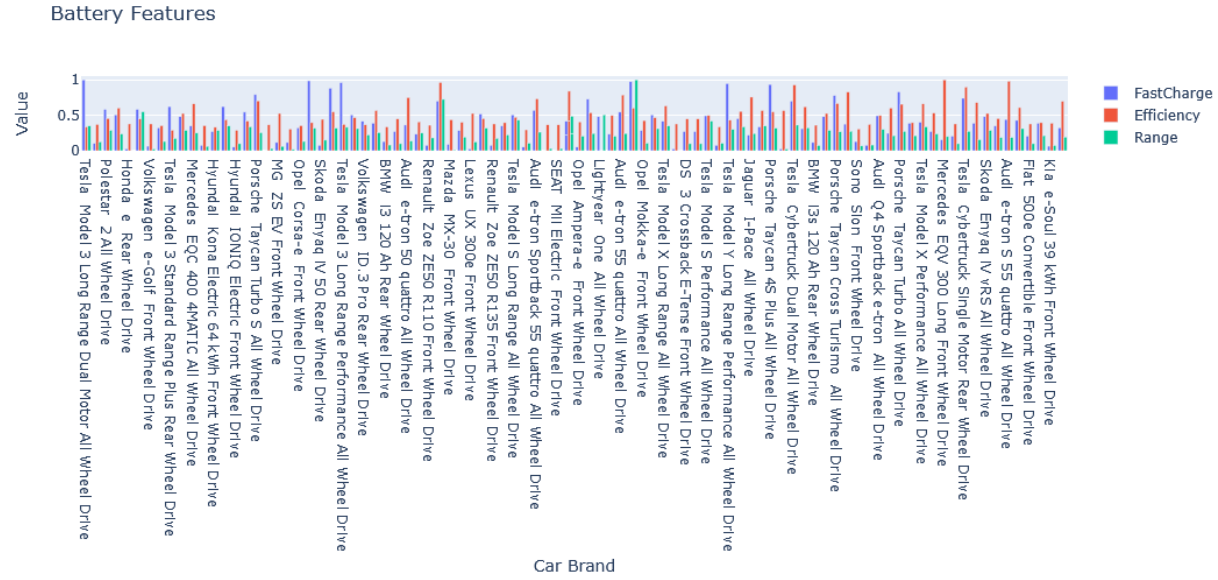


Figure 5: Distribution of features for optimal mix

Number of seats was considered to be an additional feature outside of the kinematic and electrochemical perspectives of the analysis. From the charts, there are vehicles that have the best quality features but at exorbitant prices. There are those with medium quality but with average prices, and there are also those with low quality features but at high or low prices.



range for easy visualization.

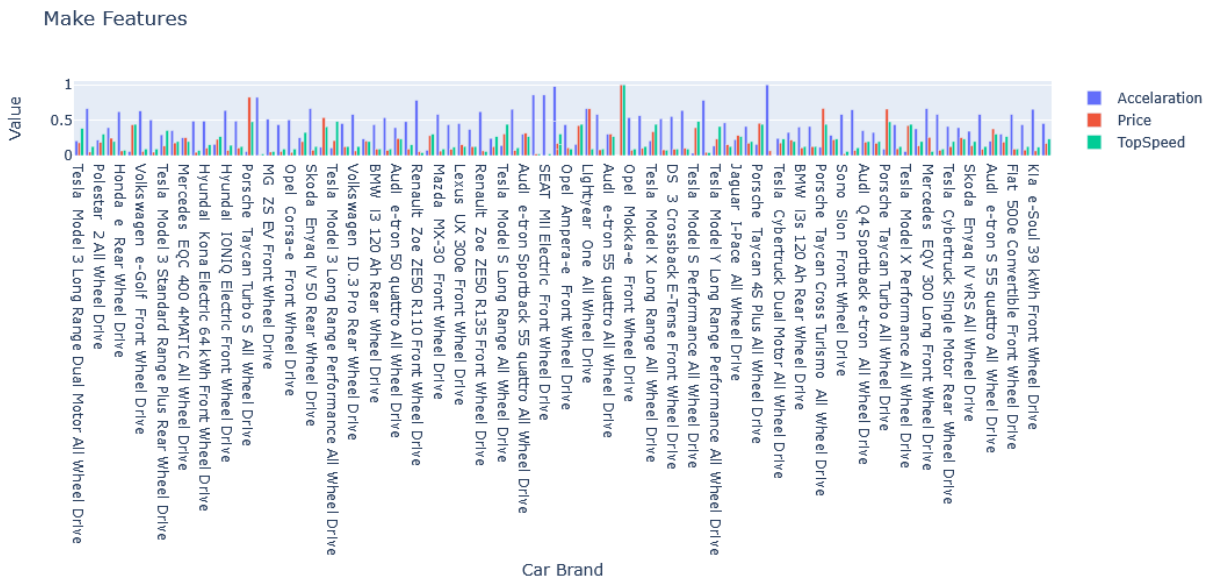


Figure 6: Distribution of features for optimal mix

Having normalized the data and built into a dataframe called norm_df, the requirement for this evaluation was to first conduct correlational analysis. The correlation matrix was obtained with correlational values of each feature with one another displayed. Seaborn heatmap was the library used for the display.

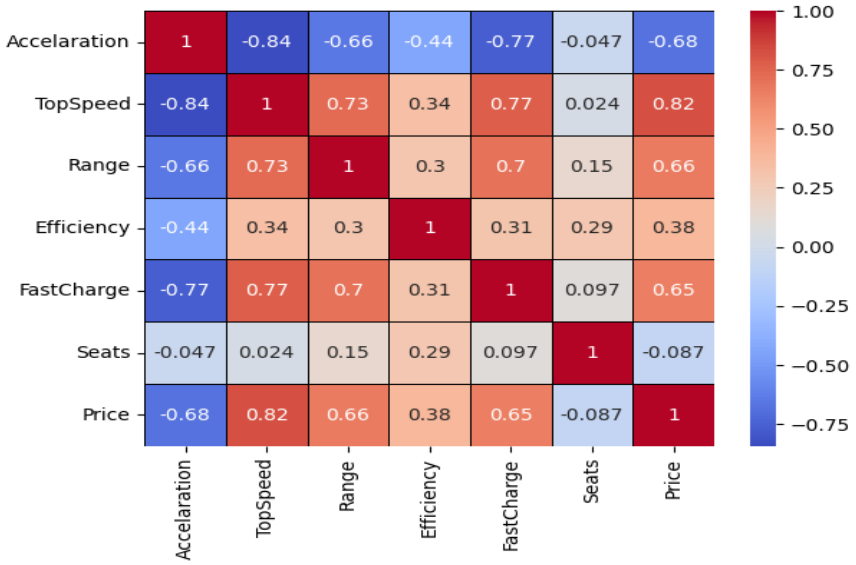


Figure 7: Correlational analysis

Acceleration, usually measured in seconds and rated as 0-60 mph or 0-100 mph, refers to how fast an electric vehicle can increase its speed from a static start or while already in motion. It simply means the rate at which the vehicle would attain its top speed. Acceleration is an essential variable often considered when vehicle performance is the priority and this is huge positive for electric vehicles due to their instant torque delivery resulting in a smooth and seamless driving experience. Top speed is another performance factor considered when rating an electric vehicle, especially when driving on highways or freeways where higher speeds are common. It is the maximum speed that an electric vehicle can reach, measured in miles per hour or kilometers per hour. From the heatmap, it shows that acceleration correlates highest with top speed and that was expected.

However, the chart shows negative correlation (-0.84). The explanation is that, the lower the acceleration values, the higher and the better the rating, implying there is an inverse relationship between acceleration and top speed. The next highest correlation value is between top speed and price (0.82). Not just price and top speed but price and acceleration (0.66). no other features correlated higher than these two. Apparently, acceleration and top speed, the kinematic features of the vehicles may be the key features that drive prices. This is confirmed by features and prices of Tesla cars and if the perspective is broadened, it might be discovered that certain demographics are easily swayed by these particular features regardless of the prices, hence, the high patronage enjoyed by vehicles in this category. Following acceleration and top speed, is fast charge and top speed (0.77) and acceleration and fast charge (-0.77). Fast charging is the ability of an electric vehicle to recharge its battery quickly and is measured in kilowatts. It is related to efficiency by way of minimizing charging time and maximizing driving time. It implies that if a vehicle can reach its top speed in a short instant of time, energy is saved and that in itself is efficiency; thus, acceleration having a negative correlation value (-0.77) with fast charge as was the case between acceleration and top speed implies that the higher the rate of acceleration (low values) the greater the efficiency of the battery. Fast charge is a critical factor to consider when going on long trips or where there is limited access to charging stations.

Range is directly related to efficiency and refers to how far an electric vehicle can travel on a single charge. Measured in miles or kilometers, range is a critical factor to consider when using the electric vehicle for long trips or daily commutes. Efficiency refers to how much of energy is converted from its source (for example, hydrogen or battery) to power the wheels. It is measured in miles (or kilometres) per kilowatt-hour. Electric vehicles are rated at higher efficiency than others if they travel longer distances at same charge. The advantage with highly efficient vehicles is that it minimizes frequency of charging and energy costs. Like efficiency, the range of an electric vehicle will determine how often it needs to be charged and how far the driver can go before the need to recharge arises. From the heatmap above, range also correlated highly with fast charge and top speed (0.7 and 0.73 respectively).

Interestingly, about four of these features correlated highly with the price and that’s a fact to note. The number of seats and efficiency did not show any correlations with others. For efficiency, this might not really pose an issue as range, fast charge and top speed already made up for that, hence, efficiency was ignored. However, efficiency and number of seats will also be factors for consideration when scores for each vehicle would be generated for evaluation. Number of seats could be an essential consideration for transport companies when transporting families or multiple passengers regularly. The number of seats can vary significantly among electric vehicle models, with some offering only two seats and others offering up to seven seats. Each of these factors is critical in the decision-making process when choosing an electric vehicle, depending on the company's needs and preferences.

To generate the scores, the two column charts were visualized. It was deduced from the charts that the features range between 0 and 1 (the effect of normalization). Four categories were created based on the idea of quartiles. For price and acceleration, the higher the values the lower the rating, while for others, the higher the values the higher the rating. The score ranged from 1 to 4, with 4 being the best and 1 the worst. For acceleration and price, 0.1 – 0.25 was rated 4, 0.26-0.50 was rated 3, 0.51-0.75 was rated 2, and 0.76-1.0 was rated 1. The reverse was the case for every other feature apart from acceleration and price, that is, 0.76 - 1.0 was rated 4, 0.51 - 0.75 was rated 3, 0.26 - 0.50 was rated 2, and 0.1 - 0.25 was rated 1.

Having established that background, the def function and for-loop command were utilized in generating the scores. After generating scores for each criterion, the algorithm would pass it to the score column it created in the norm_df. This is how the scores for each vehicle was generated to

ascertain their performance. The performance score ranged from 3 to 22. The table below shows how different vehicles performed with their respective scores sorted in descending order.

Table 9: Vehicle features and their performances

	BrandModel	PowerTrain	Acceleration	TopSpeed	Range	Efficiency	FastCharge	Seats	Price	BrandModel_PowerTrain	Score
0	Renault Zoe ZE40 R110	Front Wheel Drive	0.134454	0.411150	0.30000	0.431953	0.948052	1.000000	0.233442	Renault Zoe ZE40 R110 Front Wheel Drive	22
1	Nissan e-NV200 Evalia	Front Wheel Drive	0.243697	0.233449	0.36250	0.928994	0.701299	0.666667	0.178944	Nissan e-NV200 Evalia Front Wheel Drive	21
2	Tesla Model 3 Long Range Dual Motor	All Wheel Drive	0.210084	0.383275	0.35000	0.337278	1.000000	0.333333	0.181407	Tesla Model 3 Long Range Dual Motor All Wheel...	20
3	Tesla Model X Long Range	All Wheel Drive	0.210084	0.442509	0.35000	0.533136	0.415584	1.000000	0.337972	Tesla Model X Long Range All Wheel Drive	20
4	Tesla Model 3 Long Range Performance	All Wheel Drive	0.109244	0.480836	0.33125	0.372781	0.961039	0.333333	0.212197	Tesla Model 3 Long Range Performance All Whee...	20
...	...	...	...	...	...	...	...	...	...	...	...
88	Honda e Advance	Rear Wheel Drive	0.521008	0.076655	0.00000	0.378698	0.025974	0.000000	0.081038	Honda e Advance Rear Wheel Drive	4
89	Honda e	Rear Wheel Drive	0.521849	0.076655	0.00000	0.378698	0.025974	0.000000	0.066033	Honda e Rear Wheel Drive	4
90	SEAT Mii Electric	Front Wheel Drive	0.857143	0.024390	0.03125	0.366864	0.000000	0.000000	0.000000	SEAT Mii Electric Front Wheel Drive	3
91	Skoda CITIGOe iV	Front Wheel Drive	0.857143	0.024390	0.03125	0.366864	0.000000	0.000000	0.022605	Skoda CITIGOe iV Front Wheel Drive	3
92	Volkswagen e-Up!	Front Wheel Drive	0.823529	0.024390	0.03125	0.366864	0.000000	0.000000	0.006630	Volkswagen e-Up! Front Wheel Drive	3

93 rows × 11 columns

Determination of the optimal electric vehicle mix for the company, based on a trade-off between cost, range, fast charge, top speed, efficiency, acceleration, power train, and number of seats.

Random Forest Regressor was used for this task. The data was split into training and test datasets. X consisted of price, range, acceleration, seats, efficiency, top speed, and fast charge, while the score records were assigned to y. With test_size of 0.2, n_estimators of 100 and random_state of 123, the random forest regressor returned an R^2 of 0.89, mean squared error (MSE) of 3.45 and mean absolute error (MAE) of 1.44. however, with a little adjustment of test_size = 0.25, R^2 was 0.9 and mean squared error (MSE) of 2.91 and mean absolute error (MAE) of 1.28 which is quite an improvement of the model. The MSE of 2.91 and mean absolute error MAE of 1.28 and R^2 of 0.9 are pointers to the high accuracy of the model. The chart below indicates closeness of the predicted scores to the actual scores. Cross-validation was performed to ascertain the accuracy of the model in order to ensure there was not overfitting or underfitting and it returned the following:

Cross-validation scores: [0.80123061 0.81141216 0.80744637 0.80880809]
Average score: 0.807224306885622
Standard deviation: 0.0037423341616152715

With an average score of 0.81, a cross-validaton score of 0.81, and a small standard deviation of approximately 0.004, the model shows consistency across the folds thereby giving confidence to the model's performance.

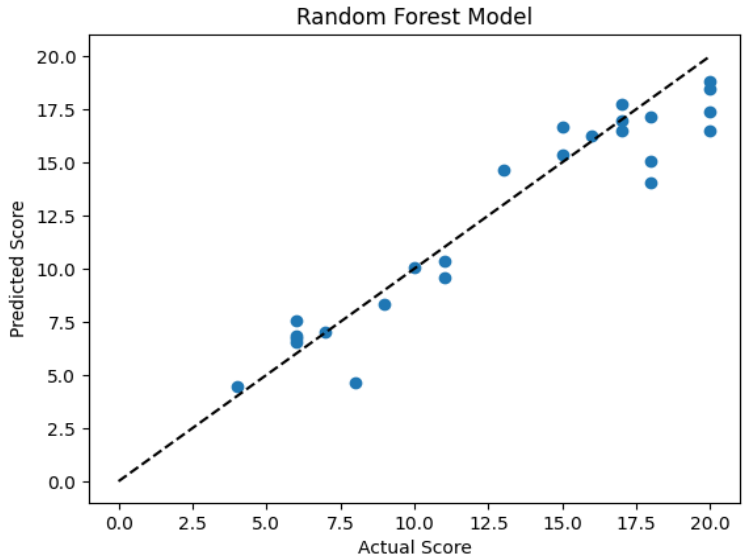


Figure 10: Predicted values against actual values

Having established the accuracy and consistency of the model, the same score criteria was fed into the algorithm to help make decision on vehicle mix to adopt. This was necessary to avoid being betrayed by emotions or being biased. Surprisingly, the model did not return vehicles having scores of 20 and 22, it rather returned those in the class of 18,19. Further investigation revealed that these constituted the best of choices (figure). The score category returned by the algorithm was used to extract the vehicles that are found within those classes and this was built into a dataframe called selected_vehicles as shown in the table below:

Table 11: Vehicle models and their performance scores

	BrandModel	PowerTrain	Score
0	Audi e-tron 55 quattro	All Wheel Drive	18
1	Audi e-tron Sportback 55 quattro	All Wheel Drive	18
2	BMW i4	Rear Wheel Drive	19
3	Byton M-Byte 72 kWh 2WD	Rear Wheel Drive	18
4	Fiat 500e Convertible	Front Wheel Drive	18
5	Ford Mustang Mach-E ER RWD	Rear Wheel Drive	18
6	Polestar 2	All Wheel Drive	19
7	Skoda Enyaq iV 80X	All Wheel Drive	18
8	Skoda Enyaq iV vRS	All Wheel Drive	18
9	Tesla Cybertruck Single Motor	Rear Wheel Drive	18
10	Tesla Cybertruck Tri Motor	All Wheel Drive	18
11	Tesla Model Y Long Range Dual Motor	All Wheel Drive	18

Further deduction resulted in the table below. From the table, it is clear that AWD vehicles would always enjoy high patronage as data has shown that they actually possess high quality features. As noted previously, it was surprising to have observed that FWD vehicles enjoyed higher patronage than their RWD counterparts but after subjecting the data to the machine learning algorithm, it is clear that the RWD drive should actually enjoy higher patronage than the FWD vehicles; thus, it is either that those who patronized the FWD did so with no data to inform them or that the previous observation was by chance.

Table 11: Vehicle models by brand

	PowerTrain	BrandModel
0	All Wheel Drive	7
1	Rear Wheel Drive	4
2	Front Wheel Drive	1

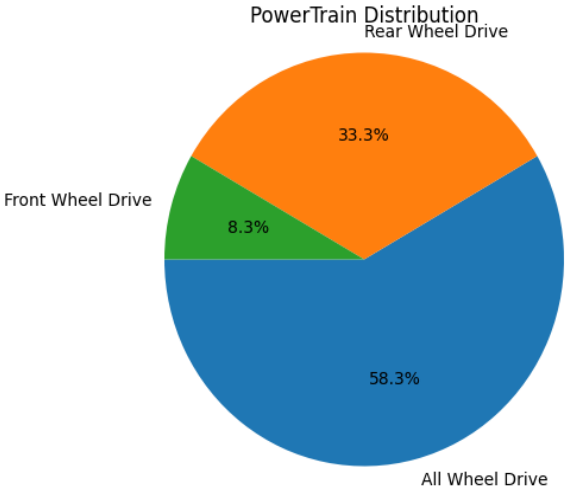


Figure 10: Percentage distribution of features for optimal mix.

The selected vehicles and their ratings shown in the figure below will guide the company in their selection of vehicles in order to obtain optimum mix.

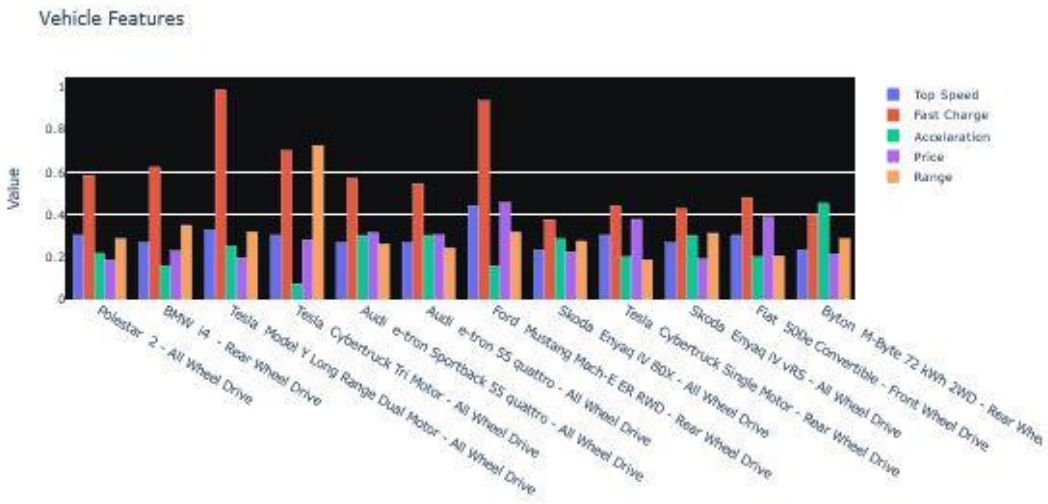


Figure 11a: Selected vehicle (electrochemical) features for optimal mix

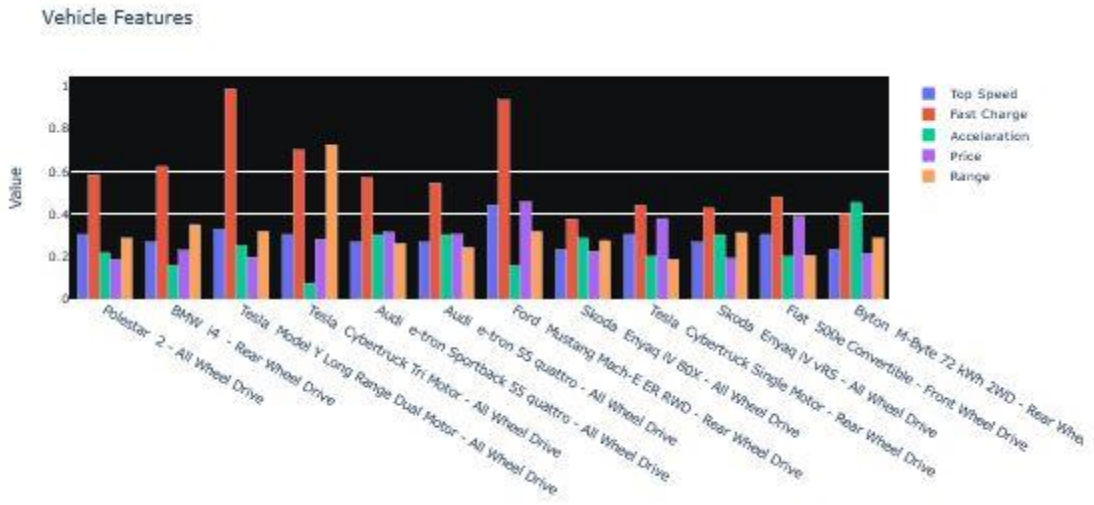


Figure 11b: Selected vehicle (electrochemical) features for optimal mix

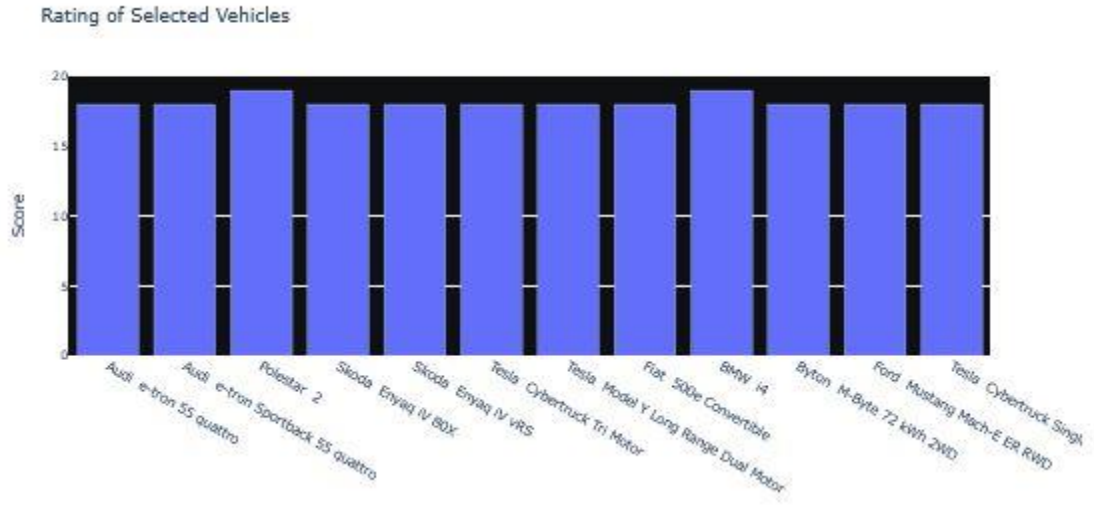


Figure 12: Vehicle performance score for optimal car selection

Conclusion

Transitioning to electric vehicles can offer numerous benefits to A2Z Limited, including reduced operating costs, improved performance, increased customer satisfaction, and a more sustainable environment. By carefully considering the trade-off between cost, range, fast charge, top speed, efficiency, acceleration, powertrain, and number of seats, A2Z Limited can make an informed decision on the optimal electric vehicle mix for their fleet. Each of these factors plays a critical role in the decision-making process when choosing an electric vehicle. The analysis showed that sedans and SUVs perform well in terms of acceleration, efficiency, range, and popularity among customers, while trucks are suitable for transporting goods with their high load capacity and range. While cost-benefit is crucial, even much more critical is the environmental impact of their choice of fleet. This will not only address the business problem but align with their commitment to sustainability and social responsibility, which can also improve their brand image and customer loyalty.

Recommendation

1. Based on the analysis of the different types of electric vehicles, it is recommended that A2Z Limited consider a mix of sedans, SUVs, and trucks.
2. In terms of powertrain, it is highly recommended that vehicles with both rear-wheel drive (RWD) and all-wheel drive (AWD) options should be purchased by A2Z Limited, as they offer better acceleration and efficiency compared to front-wheel drive (FWD) options.
3. To be specific, it is highly recommended that A2Z Limited consider the Polestar 2, BMW i4, Tesla Model Y Long Range Dual Motor, Tesla Cybertruck Tri Motor, Skoda Enyaq IV vRS, and Byton M-Byte 72 KWh 2WD because they offer a good trade-off between price, range, fast charge, top speed, efficiency, acceleration, power train, and number of seats.
4. It is also recommended that A2Z build charging infrastructure in their intended area of operations. This option can be adopted if it is much cheaper to build it than buy vehicles with low efficiency area. But then, downtime due to frequent charging can affect customers' satisfaction.

References

Bryson, A. and Berretta D. (2021). Consumer & societal attitudes towards EVs. Electric Vehicle Growth: the brands, consumers, and EV public opinion trends in 2021. Retrieved from Electric Vehicle Growth: the brands, consumers, and EV public opinion trends in 2021 (pulsarplatform.com) on April 3, 2023.

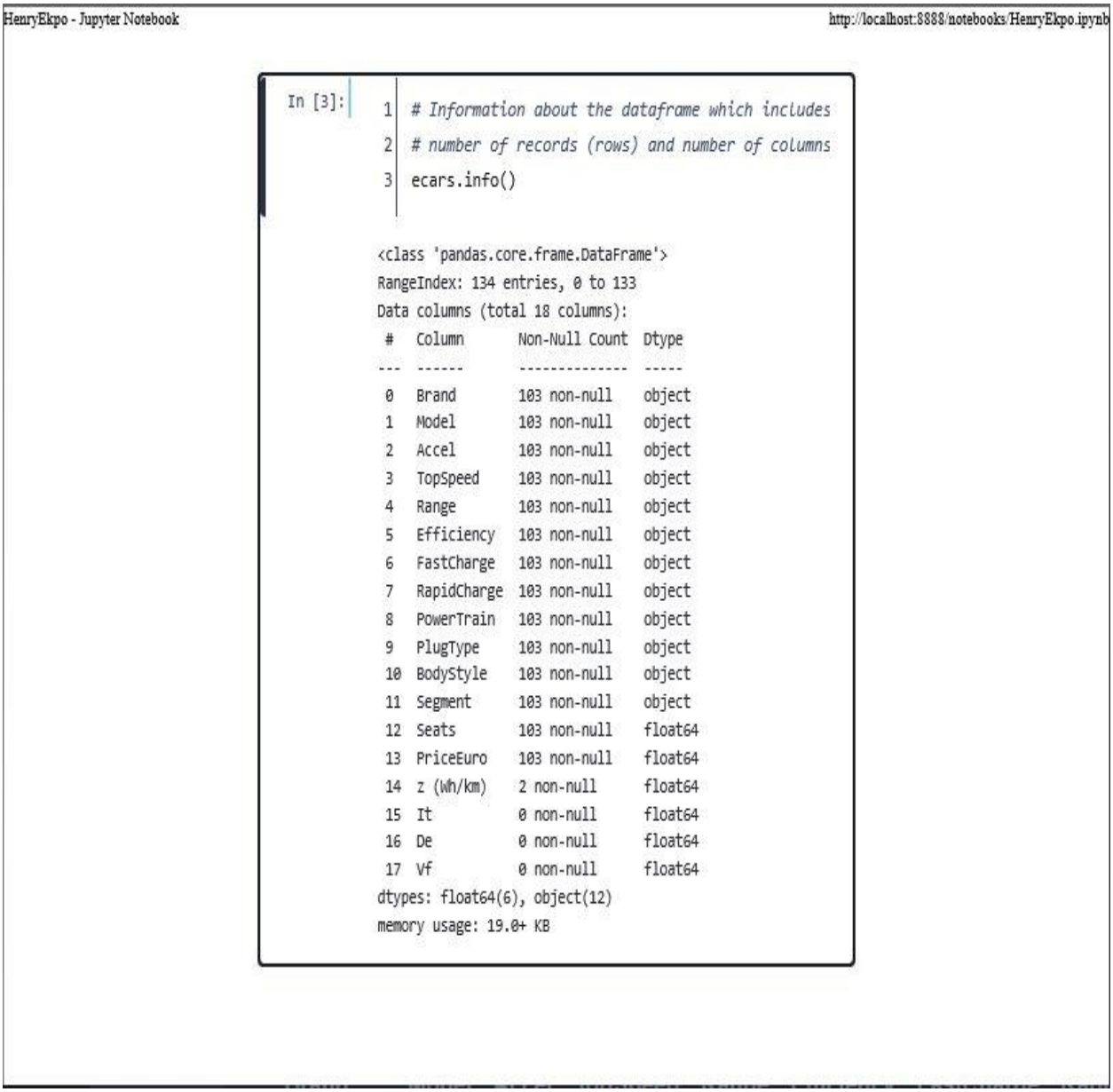
Dans, E. (2021). The five factors driving the mass adoption of electric vehicles. Retrieved from https://www.forbes.com/sites/enriquedans/2021/01/24/the-five-factors-driving-the-mass-adoption-of-electricvehicles/?sh=62149fec39d6 on April 3, 2023.

Kim, S. , Lee, J. and Lee, C. (2017). Does Driving Range of Electric Vehicles Influence Electric Vehicle Adoption? 2017, 9(10), 1783; Retrieved from https://doi.org/10.3390/su9101783 on April 3, 2023.

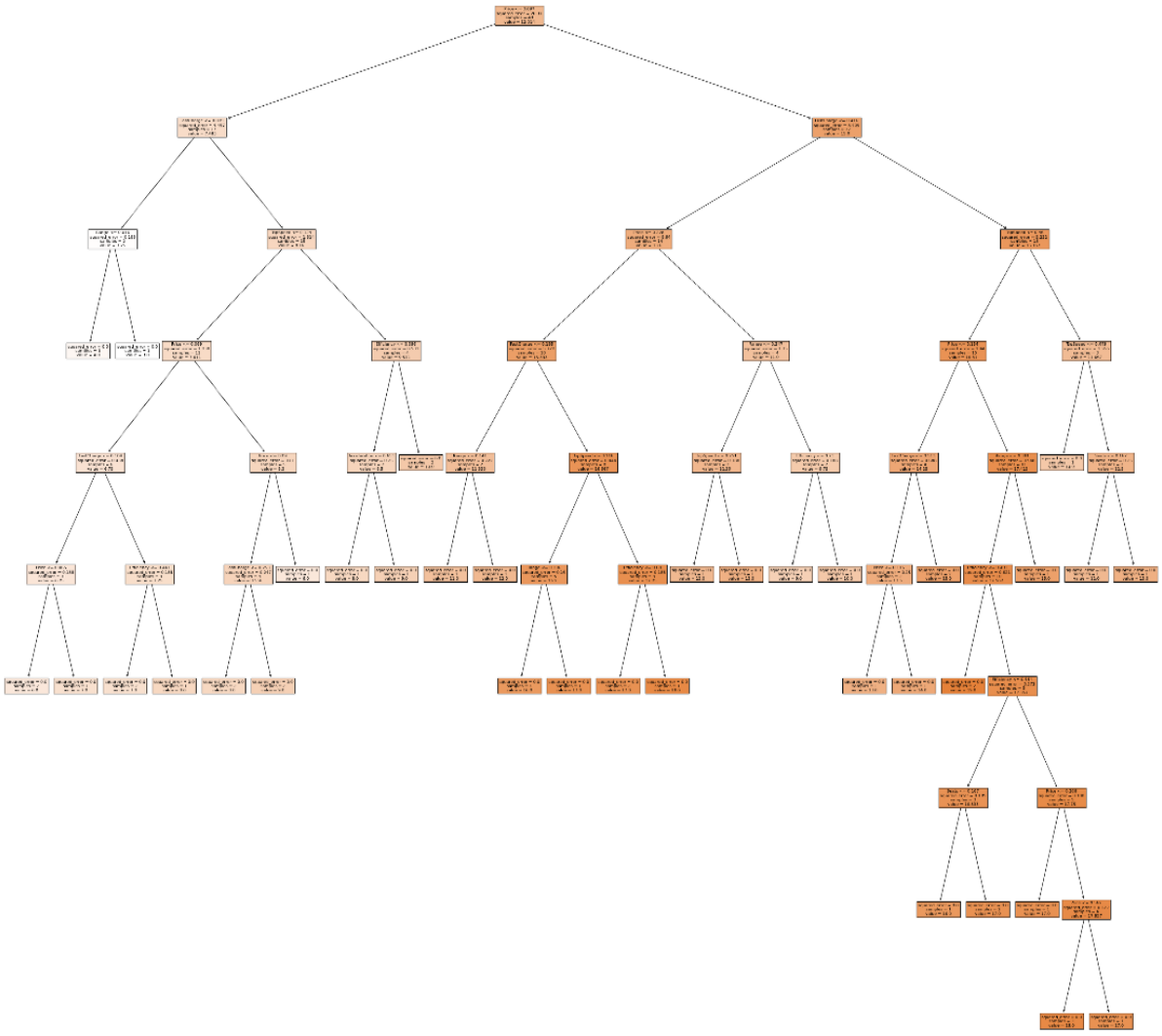
(Union of Concerned Scientists (2018). Cars, trucks, buses and air pollution. Transportation is a major source of air pollution in the United States. Published Jul 23, 2008 Updated Jul 19, 2018. Retrieved from https://www.ucsusa.org/resources/cars-trucks-buses-and-air-pollution on april 3, on April 3, 2023.)

(US EPA (2023) Transportation and climate change. Carbon pollution from transportation. Retrieved from https://www.epa.gov/transportation-air-pollution-and-climate-change/carbon-pollution-transportation 30 on April 3, 2023).

Appendix I: Basic Information on the Data set



Appendix II: Decision Nodes of the Random Forest Regressor



Appendix III: Codes

```
# Invoke required libraries.
#Pandas will handle dataframes and other data manipulations
#Numpy will handle series and other data manipulations
#Matplotlib and Seaborn modules will be used for visualization
#Scipy.stats handles statistics
import pandas as pd
import numpy as np
import matplotlib
import matplotlib.pyplot as pp
import seaborn as sns
import scipy.stats as stats

# Import the data set into the Jupyter notebook using system path
# where the data set is domiciled.
data = pd.read_csv(‘Ecar.csv', encoding='utf-8')
# Change the imported into a dataframe
ecars =pd.DataFrame(data)
# Show the dataframe
ecars

# Information about the dataframe which includes fields,
# number of entries (values/empties), type of data,
# number of records (rows) and number of columns (fields)
ecars.info()

# Remove empties or missing data across rows
ecars.dropna(how='all',inplace=True)
# Remove empties or missing data among columns
ecars.dropna(axis=1,inplace=True)
# Show the updated dataframe
ecars

mask = ecars['FastCharge'].astype(str).str.contains('-')
ecars = ecars[~mask]
ecars

# Remove symbols from numeric data
ecars['Accelaration'] = ecars['Accel'].str.replace('sec','').astype(float)
ecars['TopSpeed'] = ecars['TopSpeed'].str.replace('km/h','').astype(float)
ecars['Range'] = ecars['Range'].str.replace('km','').astype(float)
ecars['Efficiency'] = ecars['Efficiency'].str.replace('Wh/km','').astype(float)
ecars['FastCharge'] = ecars['FastCharge'].astype(str).str.replace('km/h','').astype(float)
ecars['Price'] = ecars['PriceEuro']
ecars

missingvalues = ecars.isnull().sum()
print(missingvalues)

Ecars = ecars.reindex(columns=[
'Brand','Model','Accelaration','TopSpeed','Range','Efficiency','FastCharge',
```

```
'RapidCharge','PowerTrain','PlugType','BodyStyle','Segment','Seats','Price'])
Ecars

# Descriptive statistics for numeric columns of the dataframe
round(Ecars.describe(),2)

counts = Ecars['Brand'].value_counts()
model_count = pd.DataFrame(counts)
model_count

# create bar chart of model counts
model_count.plot(kind='bar')
pp.title('Number of EV Models per Brand')
pp.xlabel('Brand')
pp.ylabel('Number of Models')
pp.show()

avg_price_by_powertrain = round(Ecars.groupby('PowerTrain',as_index=False)
['Accelaration','TopSpeed','Range','Efficiency','FastCharge','Seats','Price'].mean(),0)
avg_price_by_powertrain.sort_values('Price',ascending=False,inplace=True,ignore_index=True)
avg_price_by_powertrain

models_by_powertrain = Ecars.groupby('PowerTrain',as_index=False)['Model'].nunique()
number_of_models_by_powertrain = pd.DataFrame(models_by_powertrain)
number_of_models_by_powertrain

import matplotlib.pyplot as pp
labels = ['All Wheel Drive', 'Front Wheel Drive', 'Rear Wheel Drive']
sizes = [41, 35, 21]
fig, ax = pp.subplots()
ax.pie(sizes, labels=['All Wheel Drive', 'Front Wheel Drive', 'Rear Wheel Drive'],
autopct='%1.1f%%', startangle=90)
ax.axis('equal')
ax.set_title('PowerTrain Model Distribution')
pp.show()

avg_price_by_brand = round(Ecars.groupby('Brand',as_index=False)
[['Accelaration','TopSpeed','Range','Efficiency','FastCharge','Seats','Price']].mean(),0)
avg_price_by_brand.sort_values('Price',ascending=False,inplace=True,ignore_index=True)
avg_price_by_brand

# Check for outliers
outliers = Ecars[['Accelaration','TopSpeed','Range',
'Efficiency','FastCharge','Seats','Price']]
sns.boxplot(data=outliers,showfliers=True)
pp.show()

import matplotlib.pyplot as pp

# Define the variables to plot
variables = ['Price', 'Range', 'Seats', 'Efficiency', 'TopSpeed', 'FastCharge', 'Accelaration']
```

```
# Create the subplots
fig, axs = pp.subplots(nrows=1, ncols=len(variables), figsize=(20, 5))

# Loop through each variable and plot it
for i, variable in enumerate(variables):
    # Get the data for this variable
    data = Ecars[variable]

    # Create the histogram
    axs[i].hist(data, bins=10, color='green')

    # Calculate the median and interquartile range
    median = data.median()
    q1 = data.quantile(0.25)
    q3 = data.quantile(0.75)
    iqr = q3 - q1

    # Define the upper and lower bounds for outliers
    upper_bound = q3 + 1.5 * iqr
    lower_bound = q1 - 1.5 * iqr

    # Plot the outliers as red dots
    outliers = data[(data < lower_bound) | (data > upper_bound)]
    axs[i].plot(outliers, [0] * len(outliers), 'ro', label='Outliers')

    # Set the x-axis label and legend
    axs[i].set_xlabel(variable)
    axs[i].legend()
# Show the plot
pp.show()

# Select only columns with numeric data
num_cols = Ecars.select_dtypes(include=['float64', 'int64']).columns.tolist()
num_cols

# Create a new dataframe with the selected columns
num_df = Ecars[num_cols]
num_df

# Normalize the selected columns using MinMaxScaler
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
norm_df = pd.DataFrame(scaler.fit_transform(num_df), columns=num_cols)
norm_df

# Join strings from the Brand column with those of the Model
# column and create a new column Brandmodel
Ecars['BrandModel'] = Ecars['Brand'].str.cat(Ecars['Model'],sep=' ')
Ecars

# Merge the updated Ecars dataframe with the normalized dataframe
merged = pd.merge(Ecars[['BrandModel','PowerTrain']],
```

```
norm_df, left_index=True, right_index=True)
merged
merged['BrandModel_PowerTrain'] = merged['BrandModel'].str.cat(Ecars['PowerTrain'],sep=' ')
merged

# Create plots for the vehicle features
import plotly.graph_objs as go
# create traces for each bar
trace1 = go.Bar(x=merged['BrandModel_PowerTrain'], y=norm_df['Accelaration'],
name='Accelaration')
trace2 = go.Bar(x=merged['BrandModel_PowerTrain'], y=norm_df['Price'], name='Price')
trace3 = go.Bar(x=merged['BrandModel_PowerTrain'], y=norm_df['TopSpeed'],
name='TopSpeed')
#trace5 = go.Bar(x=e_cars['Brand'], y=e_cars['PriceEuro'], name='Price')
# create layout
layout = go.Layout(title='Make Features',
                   xaxis=dict(title='Car Brand'),
                   yaxis=dict(title='Value'))
# create figure
fig = go.Figure(data=[trace1, trace2, trace3], layout=layout)
# show plot
fig.show()

# create traces for each bar
trace4 = go.Bar(x=merged['BrandModel_PowerTrain'], y=norm_df['FastCharge'],
name='FastCharge')
trace5 = go.Bar(x=merged['BrandModel_PowerTrain'], y=norm_df['Efficiency'],
name='Efficiency')
trace6 = go.Bar(x=merged['BrandModel_PowerTrain'], y=norm_df['Range'], name='Range')

#trace5 = go.Bar(x=e_cars['Brand'], y=e_cars['PriceEuro'], name='Price')
# create layout
layout = go.Layout(title='Battery Features',
                   xaxis=dict(title='Car Brand'),
                   yaxis=dict(title='Value'))
# create figure
fig = go.Figure(data = [trace4, trace5, trace6], layout=layout)
# show plot
fig.show()

# Find correlation among the features
norm_matrix = norm_df.corr()
norm_matrix

# Create correlation matrix using seaborn heatmap
import matplotlib
import matplotlib.pyplot as pp
import seaborn as sns
sns.heatmap(norm_matrix,annot=True,cmap='coolwarm',linewidths=0.7,linecolor='black')
pp.show()

# Select criteria range based on the bar plots on vehicle features above
```

```
# Define the criteria
Range_criteria = [(0.76,1.0,4),(0.51,0.75,3),(0.26,0.50,2),(0.1,0.25,1)]
Acceleration_criteria = [(0.1,0.25,4),(0.26,0.50,3),(0.51,0.75,2),(0.76,1.0,1)]
TopSpeed_criteria = [(0.76,1.0,4),(0.51,0.75,3),(0.26,0.50,2),(0.1,0.25,1)]
Price_criteria = [(0.1,0.25,4),(0.26,0.50,3),(0.51,0.75,2),(0.76,1.0,1)]
FastCharge_criteria = [(0.76,1.0,4),(0.51,0.75,3),(0.26,0.50,2),(0.1,0.25,1)]
Efficiency_criteria = [(0.76,1.0,4),(0.51,0.75,3),(0.26,0.50,2),(0.1,0.25,1)]
Seats_criteria = [(0.76,1.0,4),(0.51,0.75,3),(0.26,0.50,2),(0.1,0.25,1)]

# Define a function to calculate the score for a given value and criteria
def get_score(value, criteria):
    for c in criteria:
        if value >= c[0] and value <= c[1]:
            return c[2]
    return 0

# Compute the score for each record in the DataFrame
scores = []
for i, row in norm_df.iterrows():
    Range_score = get_score(row["Range"], Range_criteria)
    Price_score = get_score(row["Price"], Price_criteria)
    FastCharge_score = get_score(row["FastCharge"], FastCharge_criteria)
    Acceleration_score = get_score(row["Accelaration"], Acceleration_criteria)
    TopSpeed_score = get_score(row["TopSpeed"], TopSpeed_criteria)
    Efficiency_score = get_score(row["Efficiency"], Efficiency_criteria)
    Seats_score = get_score(row["Seats"], Seats_criteria)
    total_score = Range_score + Acceleration_score + TopSpeed_score + Price_score +
FastCharge_score + Efficiency_score + Seats_score
    scores.append(total_score)

# Add the scores as a new column to the DataFrame
norm_df["Score"] = scores
norm_df

merged['Score'] = norm_df['Score']
merged
# Sort the obtained scores
merged_sorted = merged.sort_values('Score', ascending=False,ignore_index=True)
merged_sorted

# Import Python modules for random forest for the training and testing
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_absolute_error, mean_squared_error
import matplotlib.pyplot as pp

# separate target variable and input features
X = norm_df[['Price','Range','Accelaration','Seats', 'Efficiency','TopSpeed','FastCharge']]
y = norm_df['Score']

# split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=123)
```

```
# train Random Forest model
rf_model = RandomForestRegressor(n_estimators=1000, random_state=123)
rf_model.fit(X_train, y_train)

# evaluate model on testing set
y_pred = rf_model.predict(X_test)
r2_score = round(rf_model.score(X_test, y_test),2)
mae = round(mean_absolute_error(y_test, y_pred),2)
mse = round(mean_squared_error(y_test, y_pred),2)

# print performance metrics
print("R-squared score:", r2_score)
print("Mean Absolute Error (MAE):", mae)
print("Mean Squared Error (MSE):", mse)

# Create scatter plot of predicted scores versus actual scores
pp.scatter(y_test, y_pred)
pp.plot([0, 20], [0, 20], '--k')
pp.xlabel('Actual Score')
pp.ylabel('Predicted Score')
pp.title('Random Forest Model')
pp.show()

# Plot decision tree in the random forest
from sklearn.tree import plot_tree
pp.figure(figsize=(40, 40))
plot_tree(rf_model.estimators_[0], feature_names=X.columns, filled=True)
pp.show()

from sklearn.model_selection import cross_val_score
from sklearn.ensemble import RandomForestRegressor

# Load data and split into features and target variable
X = norm_df[['Price','Range','Accelaration','Seats', 'Efficiency','TopSpeed','FastCharge']]
y = norm_df['Score']

# Create random forest model with hyperparameters
rf = RandomForestRegressor(n_estimators=1000,max_depth=100)

# Perform cross-validation with 5 folds
scores = cross_val_score(rf, X, y, cv=4)

# Print the average score and standard deviation
print("Cross-validation scores: {}".format(scores))
print("Average score: {}".format(np.mean(scores)))
print("Standard deviation: {}".format(np.std(scores)))

# Extract only records that meet scores of 18 and 19
desired_cols = ['BrandModel','PowerTrain','Accelaration', 'Price', 'Efficiency',
                'Range', 'Seats', 'TopSpeed','FastCharge', 'Score']
selected_vehicles = merged[merged['Score'].isin([18,19])][desired_cols]
```

```
selected_vehicles

# Join strings from the Brand column with those of the Model column and create a new column Brandmodel
selected_vehicles['BrandModel_PowerTrain'] = selected_vehicles['BrandModel'].str.cat(selected_vehicles['PowerTrain'],sep=' - ')
selected_vehicles

vehicle_counts = pd.DataFrame(selected_vehicles.groupby('PowerTrain',as_index=False)['BrandModel'].nunique())
vehicle_counts.sort_values('BrandModel',ignore_index=True,ascending=False)

vehicle_mix = pd.DataFrame(selected_vehicles.groupby('BrandModel',as_index=False)['PowerTrain','Score'].value_counts())
vehicle_mix

# Create plots for the vehicle features
import plotly.graph_objs as go
# create traces for each bar
trace1 = go.Bar(x=vehicle_mix['BrandModel'], y=vehicle_mix['Score'], name='Score')
#trace5 = go.Bar(x=e_cars['Brand'], y=e_cars['PriceEuro'], name='Price')
# create layout
layout = go.Layout(title='Rating of Selected Vehicles',
                   xaxis=dict(title='Car Brand'),
                   yaxis=dict(title='Score'))
# create figure
fig = go.Figure(data=[trace1], layout=layout)
# show plot
fig.show()

selected_vehicles_powertrain_by_brandmodels.groupby('PowerTrain')['BrandModel'].value_counts()

# Create a DataFrame from the data
df = pd.DataFrame(selected_vehicles_powertrain_by_brandmodels)
# Count the number of each type of PowerTrain
counts = df['PowerTrain'].value_counts()
# Create a pie chart to visualize the distribution of PowerTrain types
fig, ax = pp.subplots()
ax.pie(counts.values, labels=counts.index, autopct='%1.1f%%', startangle=180)
ax.axis('equal')
ax.set_title('PowerTrain Distribution')
pp.show()
```