Development of Predictive Models for Battery Electric Vehicles (BEVs) with Best Mileage Range

Student Name: Prativa Basnet

Course Name: STAT 7240 – Applied Data Mining

Professor Name: Dr. Paul Johnson

Fall 2023

Table of Contents

List of Tables

List of Figures

a) Abstract

This project uses data from the electric vehicle population, explicitly focusing on Battery Electric Vehicles (BEVs), sourced from the “Electric Vehicle Population” dataset provided by the Washington State Department of Licensing. The primary objective of this project is to develop predictive models to identify the Make, Model, and Model Year with the best electric mileage range using data mining algorithms. The dataset extracted from Washington State Department of Licensing contains Battery Electric Vehicles (BEVs) as well as Plug-in Hybrid Electric Vehicles (PHEVs). However, this project only focuses to Battery Electric Vehicles (BEVs). Three models are developed to predict the vehicles with the best electric mileage range. The initial model employs Multiple Regression analysis, while the subsequent two models use the Regression Random Forest algorithm. The goal of this project is also to select the most effective model that enhances our understanding of the factors influencing electric mileage range and ensures the validation of predictive model accuracy.

b) Introduction

The public interest in purchase of Electric Vehicles (EVs) is rapidly increasing as we shift towards sustainable transportation. Therefore, understanding and optimizing the factors influencing the selection of the electric vehicles with the best electric mileage range becomes paramount as the adoption of EVs rises.

The main aim of this project is to develop predictive models capable of effectively identifying the Make, Model, and Model Year that yield the most favorable optimal electric mileage range. This study seeks to contribute insights into the factors influencing EV range by employing data mining algorithms and validation processes for Battery electric vehicles. This study uses data from the “Electric Vehicle Population” the Washington State Department of Licensing (DOL) provided.

The motivation for this project is to find the vehicle with best electric mileage range so that consumers experience no range anxiety when transitioning to BEVs. The most critical concern for the consumers of electric vehicles is range anxiety, the fear of running out of battery power before reaching their destination. Therefore, predicting electric mileage ranges of vehicles to eliminate range anxiety accurately and promote widespread BEV adoption is crucial. This project specifically focuses on identifying the Make, Model, and Model Year for the vehicles with the best electric mileage range.

c) Methodology

The description of dataset and methods used in this project are described in detail below.

a. Data Description

The electric vehicle population data used in this report is extracted from **data.gov**. The dataset "Electric Vehicle Population" contained in **data.gov** was initially created on April 16, 2019 and updated on September 14, 2023. This dataset is registered through the Washington State Department of Licensing (DOL). There are a total of 143,269 registered electric vehicles in the dataset. The size of this dataset is reduced to 110,652 by selecting the vehicles registered only in Washington State and ignoring Plug-in Hybrid Electric Vehicles. There are total of 17 variables in the dataset.

The dataset contains parameters such as Make, Model, Model Year, Electric range, Electric Vehicle Type, etc. The predictor parameters are Make, Model, and Model Year, and the target parameter is Electric Range. The electric range not researched is entered as zero in the dataset and therefore excluded from the study. The study uses total of 46,811 data for predicting the best electric mileage range using predictor parameters Make, Model, and Model Year from 1997 through 2021.

Error! Reference source not found. below shows the variable types and names for the electric ehicle population data used in this study. The "Electric range" and "Model Year" are quantitative variables, and the "Make" and "Model" are qualitative variables.

Table 1: Description of Variables

Variable Names	Variable Type	Descriptions
Electric Range	Quantitative	How far a vehicle travel on it electric charge
Make	Qualitative	Manufacture of the vehicle
Model	Qualitative	Model of the vehicle
Model Year	Quantitative	Model year of vehicle

The "Electric Range" variable is defined as how far a vehicle travels with one time fully electric charge. The "Make" is name of the manufacture of vehicles, "Model" is the model of the vehicles, and "Model Year" is the built year. The dataset contains the vehicles' Make, Model, and Model year, as shown below in Table 2.

Table 2: Make, Model, and Model Year of BEVs

Makes	Models	Model Year
AUDI	E-TRON, E-TRON SPORTBACK	2019, 2020, 2021
AZURE DYNAMICS	TRANSIT CONNECT ELECTRIC	2011, 2012
BMW	I3	2014 to 2020
CHEVROLET	BOLT EV, SPARK, S-10 PICKUP	1997, 2014 to 2020
FIAT	500	2013 to 2019
FORD	RANGER , FOCUS	1998 to 2000, 2012 to 2018
HYUNDAI	IONIQ, KONA	2017 to 2020
JAGUAR	I-PACE	2019, 2020
KIA	NIRO, SOUL, SOUL EV	2015 to 2020
MERCEDES-BENZ	B-CLASS	2014 to 2017
MINI	HARDTOP	2021
MITSUBISHI	I-MIEV	2012, 2014,2016,2017
NISSAN	LEAF	2011 to 2020
POLESTAR	PS2	2021
PORSCHE	TAYCAN	2020,2021
SMART	EQ FORTWO, FORTWO, FORTWO ELECTRIC DRIVE	2013 to 2019
TESLA	MODEL S, MODEL 3, MODEL X, MODEL Y,ROADSTER	2008, 2010 to 2020
TH!NK	CITY	2011
TOYOTA	RAV4	2002,2003, 2012-2014
VOLKSWAGEN	E-GOLF	2015 -2019

The “Make” and “Model” qualitative variables are transformed into factors. This conversion enables a more effective exploration of the relationship between the Make and Model and their respective electric mileage ranges. The dataset has thoroughly been examined for the presence of missing values and outliers, and it has been verified that the dataset is free from any missing values or outliers.

As previously described, the dataset for this project is based on electric vehicles registered through the Washington State Department of Licensing (DOL) and built between 1997 and 2021. The best electric mileage ranges for vehicles built between 1997 and 2021 are computed for various Makes and Models. This computation involves carefully considering the electric mileage achieved by each electric vehicle, allowing for comparative analysis across different Make and Model combinations.

The scatter plot shown in Figure 1 shows the electric mileage of Makes by “Model Year.” This scatter plot displays the electric mileage range of different makes (Tesla, Audi, Nissan, etc.) and Model Years (1997 through 2021).

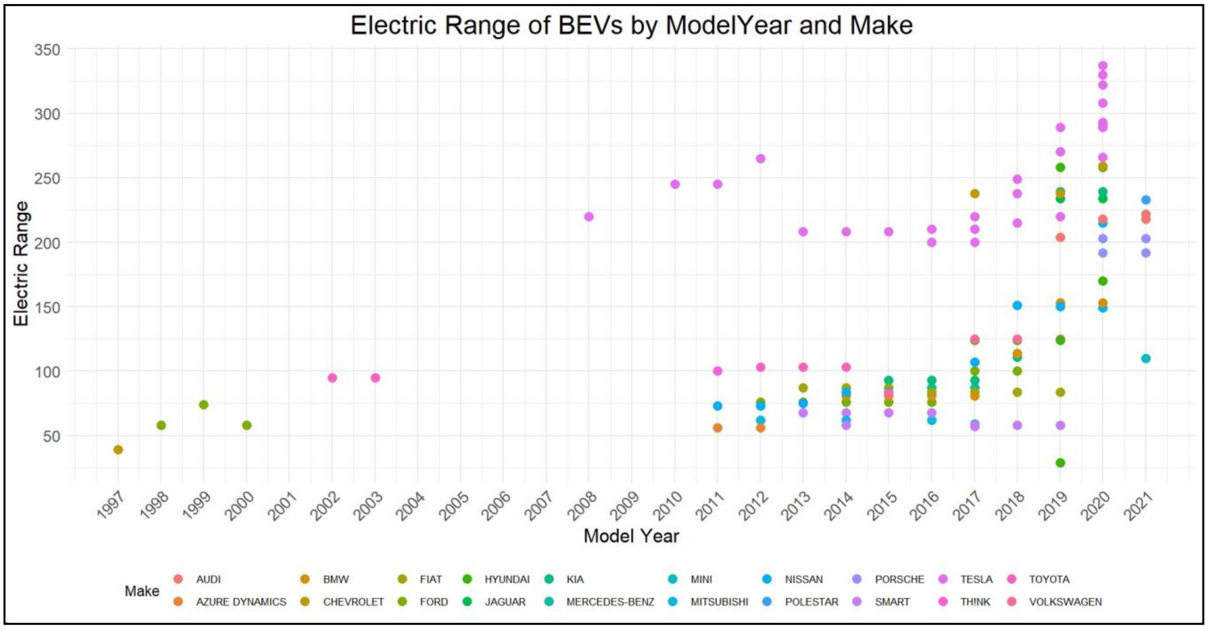


Figure 1: Scatter Plot of Electric Range by Model Year & Make

The scatter plot shown in Figure 2 shows the electric mileage of Models by "Model Year." This scatter plot displays the electric mileage range of different models (Model S, E-Tron, Leaf, etc.) and Model years (1997 through 2021).

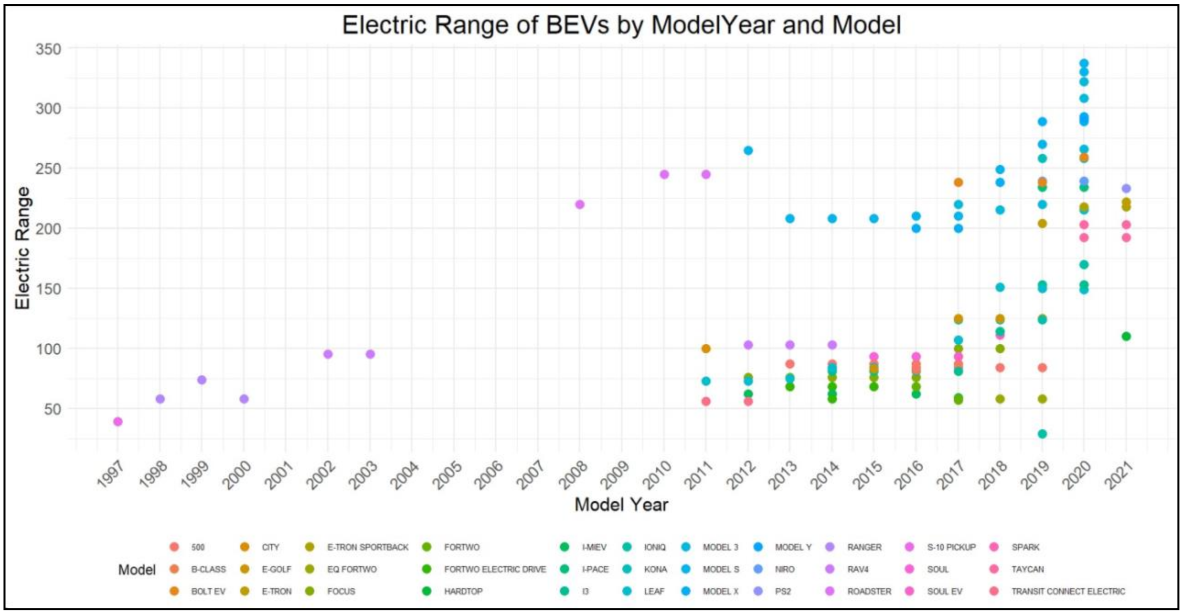


Figure 2: Scatter Plot of Electric Range by Model Year & Model

Each data point on the scatter plot corresponds to a unique combination of Make, Model, and Model Year, providing a detailed visualization of electric mileage trends across different vehicle specifications. The scatter plot shows patterns, trends, or insights related to electric mileage, considering both Make and Model factors and their evaluation over the years.

There are 46,811 observations (data) used in the development of models after pre-processing (data cleanup) the given dataset. The predictive models contain 46,811 data points after conducting the data cleanup.

The cleaned up dataset is split into training and testing datasets using the 80/20 split ratio to ensure the model's effectiveness. Using 80/20 split rule, there are total of 37,448 data (80%) for training the model and the remaining 9,363 (20%) for validating the model. This testing set serves a crucial role in the unbiased evaluation of the model's performance, allowing it to assess its generalization capabilities.

b. Methods

The two distinct Multiple Regression and Regression Random Forest methodologies are used to predict the vehicles with best electric mileage range. The Multiple Regression used in first model. Similarly, the Regression Random Forest is used in 2nd and 3rd models.

The multiple regressions (1st) model is used to understand how various factors such as "Make", "Model," and "Model Year" influence the predictions. The linear interdependence between the predictors is assessed using the linear model's alias function. The alias function employed on the linear model reveals instances of linear interdependence, as presented in Table 3. A value of 0 denotes no aliasing, while a value of -1 or 1 indicates an alias relationship.

Table 3: Linear Interdependence of the Predictor Make and Model

```
Model :
Electric_Range ~ Model_Year + Make + Model

Complete :
                             (Intercept) Model_Year MakeAZURE DYNAMICS MakeBMW
ModelB-CLASS                  0           0          0                  0
ModelCITY                     0           0          0                  0
ModelE-GOLF                   0           0          0                  0
ModelE-TRON SPORTBACK         1           0         -1                 -1
ModelFORTWO ELECTRIC DRIVE    0           0          0                  0
ModelHARDTOP                  0           0          0                  0
ModelI-MIEV                   0           0          0                  0
ModelI-PACE                   0           0          0                  0
ModelI3                       0           0          0                  1
ModelKONA                     0           0          0                  0
ModelLEAF                     0           0          0                  0
ModelPS2                      0           0          0                  0
ModelRANGER                   0           0          0                  0
ModelRAV4                     0           0          0                  0
ModelROADSTER                 0           0          0                  0
ModelSOUL EV                  0           0          0                  0
ModelSPARK                    0           0          0                  0
ModelTAYCAN                   0           0          0                  0
ModelTRANSIT CONNECT ELECTRIC 0           0          1                  0
                             MakeCHEVROLET MakeFIAT MakeFORD MakeHYUNDAI
ModelB-CLASS                  0             0        0        0
ModelCITY                     0             0        0        0
ModelE-GOLF                   0             0        0        0
ModelE-TRON SPORTBACK        -1            -1       -1       -1
ModelFORTWO ELECTRIC DRIVE    0             0        0        0
ModelHARDTOP                  0             0        0        0
ModelI-MIEV                   0             0        0        0
ModelI-PACE                   0             0        0        0
ModelI3                       0             0        0        0
ModelKONA                     0             0        0        1
ModelLEAF                     0             0        0        0
ModelPS2                      0             0        0        0
ModelRANGER                   0             0        1        0
ModelRAV4                     0             0        0        0
ModelROADSTER                 0             0        0        0
ModelSOUL EV                  0             0        0        0
ModelSPARK                    1             0        0        0
ModelTAYCAN                   0             0        0        0
ModelTRANSIT CONNECT ELECTRIC 0             0        0        0
                             MakeJAGUAR MakeKIA MakeMERCEDES-BENZ MakeMINI
ModelB-CLASS                  0          0       1                 0
ModelCITY                     0          0       0                 0
ModelE-GOLF                   0          0       0                 0
ModelE-TRON SPORTBACK        -1         -1      -1                -1
ModelFORTWO ELECTRIC DRIVE    0          0       0                 0
ModelHARDTOP                  0          0       0                 1
```

This analysis shows a linear interdependence between Make and Model predictor parameters. Hence, the second and third models are developed to deal with linear interdependence issues using Regression Random Forest. The second model uses “Make” and “Model Year” as predictor parameters, while the third model uses “Model” and “Model Year”.

The hyper-parameter tuning is conducted for both 2nd and 3rd models to optimize performance. The Regression Random Forest employs a tuning grid with parameters such as “variance” as splitrule, a minimum node size of “default 5”, and mtry (randomly selected predictor) as “square root of the total predictors”.

In the hyper-parameter process for the second and third models, the tuning grid includes “variance” for splitrule, predictors 1 and 2 for mtry, and 5, 7, 9, & 11 for min. node size. The Root Mean Squared Error (RMSE) is an evaluation metric. The cross-validation method with 10-fold approach is implemented as part of the training control to ensure robust evaluation.

The Mean Absolute Error (MAE) and R-Squared are also used to evaluate the performance of the developed models. The primary objective of this analysis is to identify the most effective model (either Model 2 or Model 3) for predicting the electric range of BEVs. This determination is based on the specified predictors and tuning parameters, with the performance metrics serving as critical benchmarks for model effectiveness. The output from the Regression Random Forest method for Model 2 is shown below in Table 4. The smallest value of RMSE was used to select the optimal model. The final values used for Model 2 are mtry of 2, splitrule of variance, and min.node size of 9.

Table 4: Model 2 Output Result using Regression Random Forest

```
Random Forest

37448 samples
    2 predictor

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 33703, 33704, 33704, 33703, 33702, 33703, ...
Resampling results across tuning parameters:

  mtry  min.node.size  RMSE      Rsquared   MAE
  1      5             67.89868  0.8156184  55.46462
  1      7             67.92569  0.8177308  55.54364
  1      9             68.07595  0.8204772  55.68554
  1     11             67.92678  0.8207422  55.50646
  2      5             56.32249  0.8408087  44.37303
  2      7             56.50795  0.8443263  44.63037
  2      9             56.05729  0.8394628  44.06063
  2     11             56.74498  0.8425738  44.83132

Tuning parameter 'splitrule' was held constant at a value of variance
RMSE was used to select the optimal model using the smallest value.
The final values used for the model were mtry = 2, splitrule = variance
 and min.node.size = 9.
```

The output from the Regression Random Forest method for Model 3 is shown below in Table 5. The smallest value of RMSE was used to select the optimal model. The final values used for Model 3 are mtry of 2, splitrule of variance, and min.node size of 7.

Table 5: Model 3 Output Result using Regression Random Forest

```
Random Forest

37448 samples
    2 predictor

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 33703, 33704, 33704, 33703, 33702, 33703, ...
Resampling results across tuning parameters:

  mtry  min.node.size  RMSE      Rsquared   MAE
  1      5             72.62407  0.7796764  60.04164
  1      7             72.40219  0.7764834  59.82097
  1      9             72.57225  0.7814325  59.97018
  1     11             72.54376  0.7759017  59.95174
  2      5             70.97736  0.8080661  58.49837
  2      7             70.86523  0.8047507  58.40790
  2      9             70.96467  0.8118751  58.51141
  2     11             70.88566  0.7957194  58.43301

Tuning parameter 'splitrule' was held constant at a value of variance
RMSE was used to select the optimal model using the smallest value.
The final values used for the model were mtry = 2, splitrule = variance
 and min.node.size = 7.
```

d) Results and Discussion

The outcomes from the three distinct models, Models 1, 2, and 3, are examined carefully to understand their efficacy in explaining the electric mileage range variation in battery electric vehicles.

The detailed results of the Multiple Regressions Model (1^{st} Model) are shown below in Table 6. The Multiple Regressions Model (MLR) shows predictor parameters such as Make, Model, and Model Year explained 86% of electric mileage range variation. The model exhibited a significant p-value of $2.2e^{-16}$, indicating its statistical significance. However, it’s important to note that a linear interdependence exists between predictor parameters “Make” and “Model”. Hence, the second and third models are needed to be developed.

Table 6: Result from Multiple Regression

MLR Results	Predictors: Make, Model, Model Year
Residual Standard Error	21.6
Degree of Freedom	37414
R-Squared	0.860
F-statistic	12020
P-value	$2.2e^{-16}$

Models 2 and 3 use the Regression Random Forest method and the output results from both models are shown in Table 7. The Root Mean Square Error (RMSE) and Absolute Mean Error (MAE) values provide insights into the predictive accuracy of the models. Model 2 displayed robust predictive capabilities, especially during the training phase, with lower Root Mean Square Error (RMSE) and Absolute Mean Error (MAE) compared to Model 3. The R-Squared values further support Model 2 efficacy, indicating strong predictive capabilities, especially concerning the influential predictors of Make and Model Year.

Model 3, however, showed diminished performance, particularly in the testing set where the R-Squared value dropped substantially. The comprehensive analysis suggests that Model 2, employing the Regression Random Forest approach, stands out as a robust predictor for estimating the best electric range of battery vehicles, with the most influential predictors Make and Model Year.

The analysis indicates Model 2 demonstrates strong predictive capabilities for estimating the best electric range of Battery vehicles with the most influential predictors Make and Model Year.

Table 7: Results for Model 2 and 3

Result	Model	Root Mean Square Error (RMSE)	Absolute Mean Error (MAE)	R-Squared
Training	Model 2	56.05	44.06	0.8394
Testing	Model 2	57.45	45.43	0.4039
Training	Model 3	70.86	58.49	0.8047
Testing	Model 3	72.00	59.41	0.0639

The essential pivotal factors influencing the prediction of best electric mileage range vehicles are shown in Figure 3. The most important scaled factors influencing the prediction of best electric mileage are **Make**, notably Tesla and Nissan, and the Model Year of the vehicles. This graphical representation shows the significance of these specific factors in determining the optimal electric range for vehicles in the study. The prominence of Make, focusing on Tesla and Nissan and including Model Year, suggests that electric mileage is notably influenced by these manufacturers' inherent characteristics and specifications of vehicles.

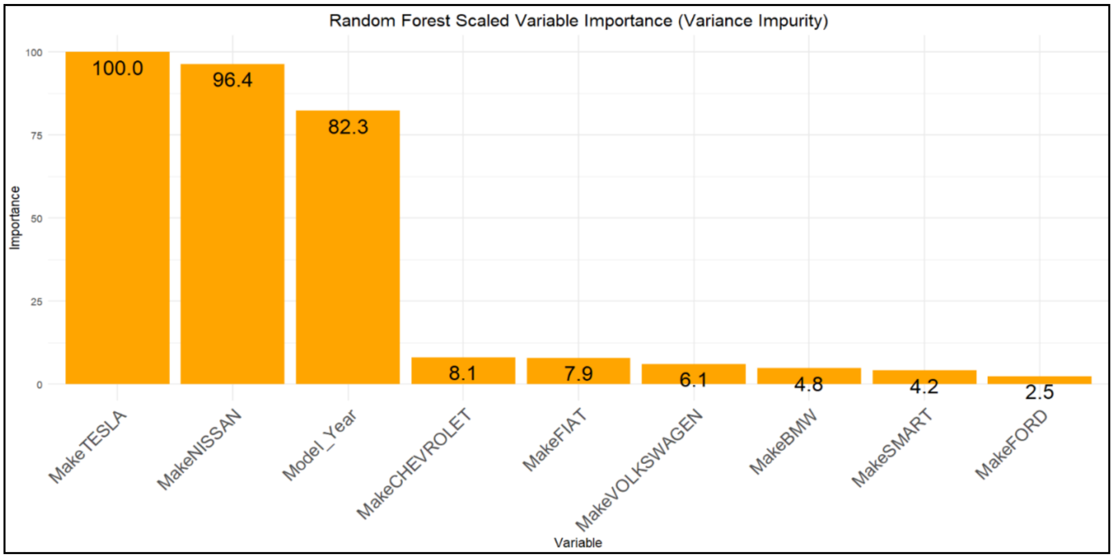


Figure 3: Bar Graph with the Random Forest Scaled Variable Importance

The results presented in Table 7 above show that the performance metrics of Model 2 have an RMSE of 56.05 and MAE of 44.06 for the training set and an RMSE of 57.45 and MAE of 45.43 for the testing set. The RMSE and MAE values from the testing set are close and comparable to the training set, indicating that the predictors' Make and Model Year accurately predict the electric mileage range. The R-squared value for the training set stands at 83.94%, signifying that the predictors in the model account for approximately 83.94% of the variability in the electric range. However, the test set R-Squared value is notably low at 40.39%, indicating a substantial decrease in explanatory power. This discrepancy implies that while the predictors Make and Model Year demonstrate proficiency in explaining variability in the training set, their effectiveness diminished when applied to the test set, highlighting a limitation in capturing overall variability in the electric mileage range.

Hence, adjusting the model is imperative to enhance predictive accuracy and comprehensively account for variability in electric mileage. Additional predictors such as "climate condition", "battery capacity", "driving condition", and "driving habits" should be incorporated. These additional factors are expected to contribute crucial insights and refine the model, ensuring a more robust and encompassing representation of the diverse factors influencing the electric mileage range.

e) Conclusion

The study provides valuable insights into predicting the best electric mileage range of BEVs, utilizing a comprehensive dataset and advanced predictive modeling techniques. The initial model (Model 1), employing multiple regression analysis, reveals significant explanatory power, explaining approximately 86% of the variation in the electric mileage range. However, the presence of linear interdependencies between “Make” and “Model” necessitated the development of the subsequent two models. The regression random forest models (Models 2 and 3) emerged as critical advancements, effectively addressing the issues posed by linear interdependence.

Model 2, in particular, has demonstrated robust predictive capabilities during both training and testing phases, surpassing the performance of Model 3. The graphical representation further explains the significance of predictors such as “Make” (highlighting Tesla and Nissan) and “Model Year”.

This study requires the need for continuous refinement. The lower R-Squared value in the testing set implies a limitation in capturing overall variability affecting electric mileage. In order to enhance predictive accuracy, future iterations should consider incorporating additional predictors such as “climate condition”, “battery capacity”, “driving condition”, and “driving habits”. These refinements promise to yield a more comprehensive model, ensuring a nuanced understanding of the diverse factors shaping electric vehicle performance.

f) References

Electric Vehicle Dataset, Washington. (2022) Data.gov. https://catalog.data.gov/dataset/electric-vehicle-population-data/resource/fa51be35-691f-45d2-9f3e-535877965e69

H. Ribberink and E. Entchev, *"Electric vehicles — A 'one-size-fits-all' solution for emission reduction from transportation?,"* (2013) World Electric Vehicle Symposium and Exhibition (EVS27), Barcelona, Spain, 2013, pp. 1-7, doi: 10.1109/EVS.2013.6914837.
Lars-Henrik Björnsson, Sten Karlsson, "Electrification of the two-car household: PHEV or BEV?", Transportation Research Part C: Emerging Technologies, Volume 85, (2017), https://doi.org/10.1016/j.trc.2017.09.021.

Plötz, P., Funke, S.A., Jochem, P. et al. CO2 Mitigation Potential of Plug-in Hybrid Electric Vehicles larger than expected. Sci Rep 7, 16493 (2017). https://doi.org/10.1038/s41598-017-16684-9

Lin, Z., & Greene, D. L. (2011). Promoting the Market for Plug-In Hybrid and Battery Electric Vehicles: Role of Recharge Availability. Transportation Research Record, 2252(1), 49-56. https://doi.org/10.3141/2252-07

143269

KENNESAW STATE UNIVERSITY
COLLEGE OF COMPUTING AND SOFTWARE ENGINEERING
School of Data Science and Analytics

BEV and PHEV Usage in WA as a Model for Addressing Electric Vehicle Adoption Obstacles

Prativa Basnet & Brandi Jones

Faculty Advisor: Dr. Austin Brown

R Studio

KENNESAW STATE UNIVERSITY
COLLEGE OF COMPUTING AND SOFTWARE ENGINEERING
School of Data Science and Analytics

INTRODUCTION

- The urgent global threat of climate change makes finding wide-ranging mitigation efforts crucial. Electric vehicles have the potential to greatly reduce greenhouse gases, energy security, and air pollutant concerns.
- Issues with range, cost, and fast-charge availability have traditionally been obstacles to EV adoption. Battery Electric Vehicles (BEV) and Plug-In Hybrid Vehicles (PHEV) present two different strategies of electrification and thus address adoption obstacles in alternate ways.
- BEV usage can reduce both reliance on fossil fuel energy and gas emissions but suffers from range limitations. PHEV does not suffer from overall range limits, but the fuel-engine powertrain adds substantial consumer cost.
- The purpose of the current study is to evaluate BEV and PHEV usage in Washington State in an effort to understand public demand and market potential for increased adoption.

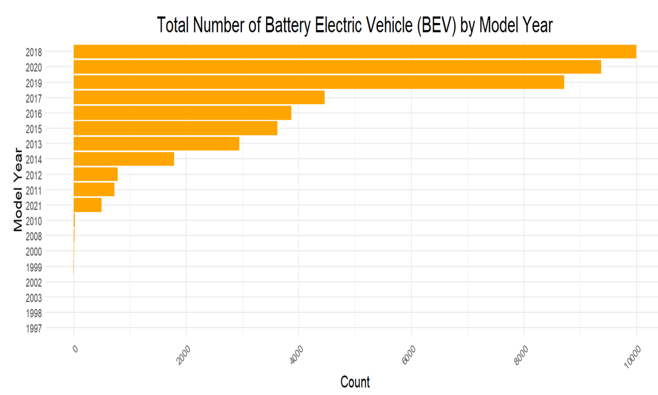
METHODS

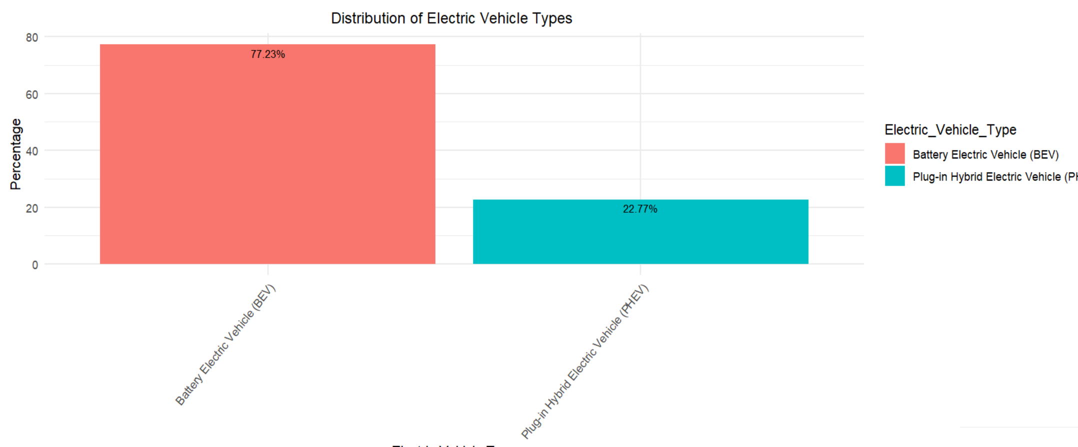
- Electric Vehicle Population dataset showing the Battery Electric Vehicles (BEVs) and Plug-in Hybrid Electric Vehicles (PHEVs) that are currently registered through Washington State Department of Licensing (DOL) was obtained from data.gov.
- R Studio was used for all data manipulation and visualizations.
- Data was separated into BEV and PHEV subsets for market demand exploration
- Average and maximum electric ranges were calculated by model and make for model years 2008-2024.
- Simple Linear Regression model used to evaluate PHEV vs. BEV electric range.
- Multiple Linear Regression model was created to examine BEV makes across model years.
- Shapiro-Wilk, Breusch-Pagan tests and visualizations were used to check for regression assumption violations.

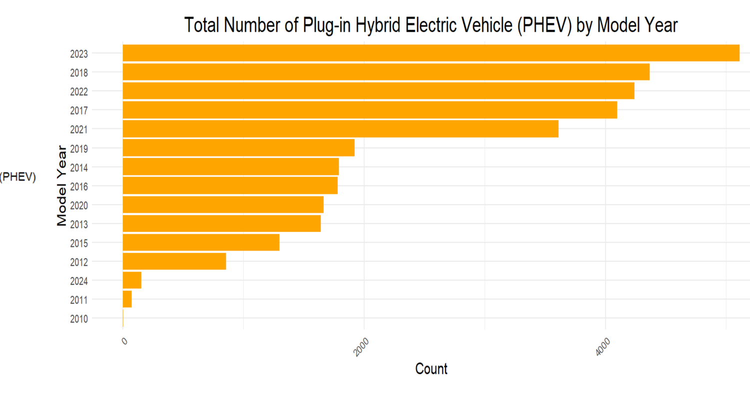
RESULTS

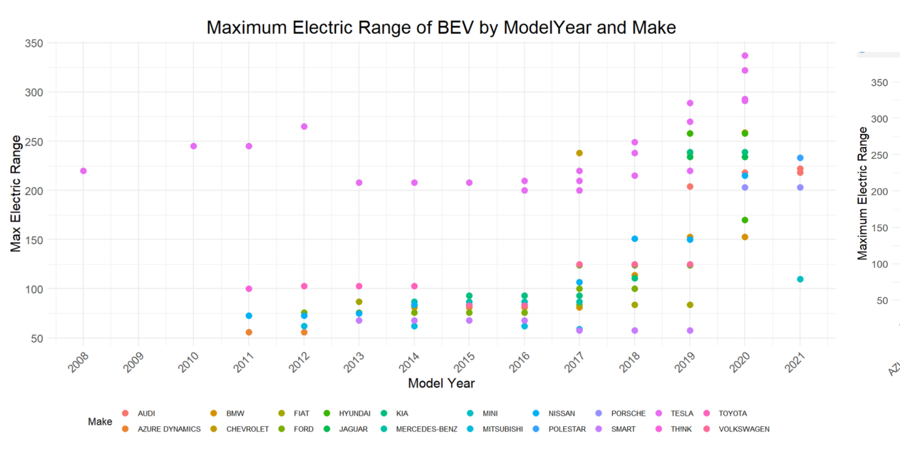
- 143,269 electric vehicles registered in WA were included in the study. Of those, 77% or 110,652 were BEVs and 23% or 32,615 were PHEVs.
- Various makes and models of both EV types were evaluated for avg and max electric travel range. Tesla's 2022 BEV Model Y has the highest avg and 2022 Model S has the highest maximum, 337 miles, electric range overall EV types. Wheego Electric Cars produces the highest max and BMW's 2021 I3 has the highest avg electric range vehicle for PHEVs.
- The number of available models has increased sharply since 2010 for both BEV and PHEVs, with BEVs having the larger model selection of the two.
- An SLR of BEVs versus PHEVs demonstrated a weak positive association for electric range by BEV type.
- MLR of Model Year and Make explained 86% of electric range variation.

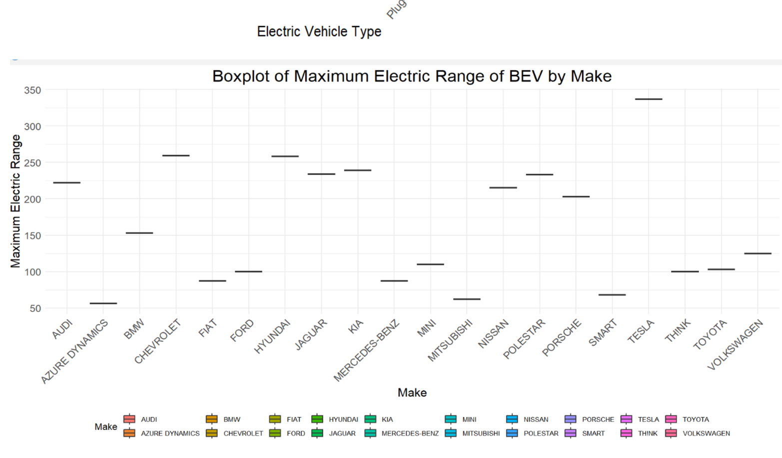
Scan for References

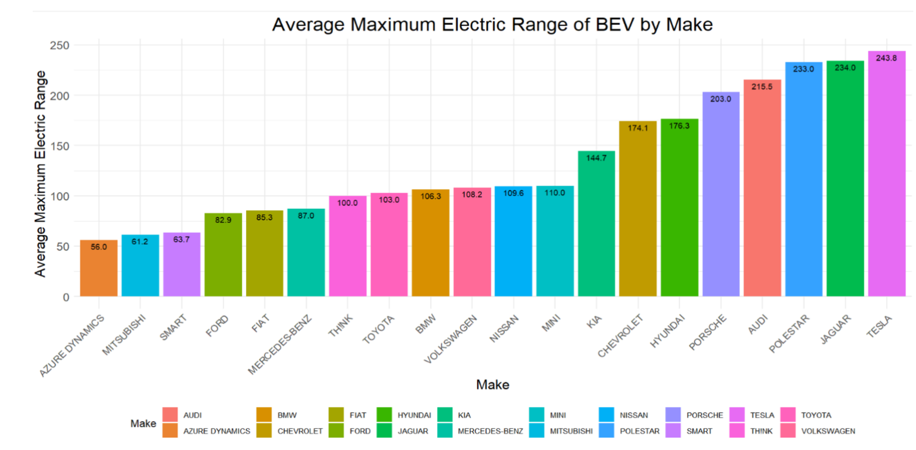


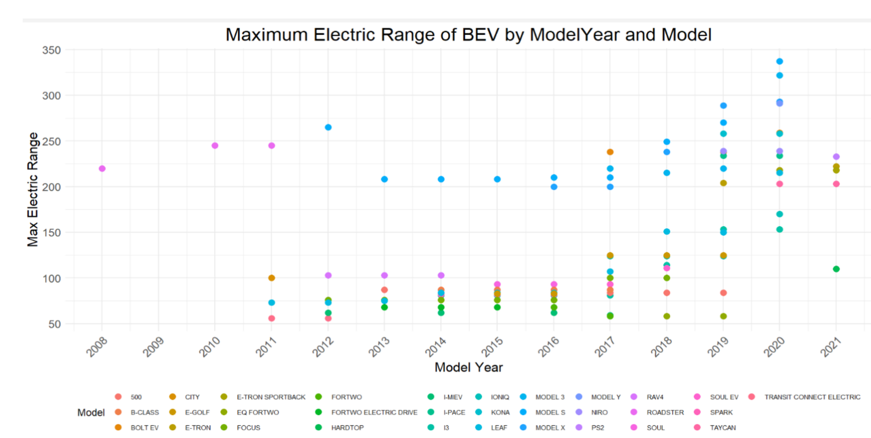


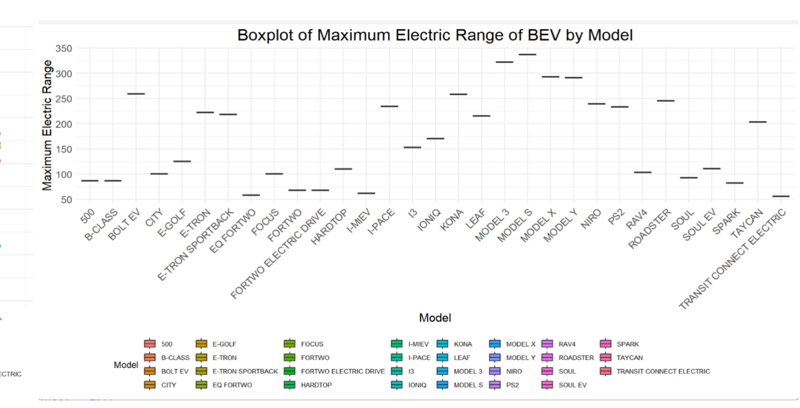


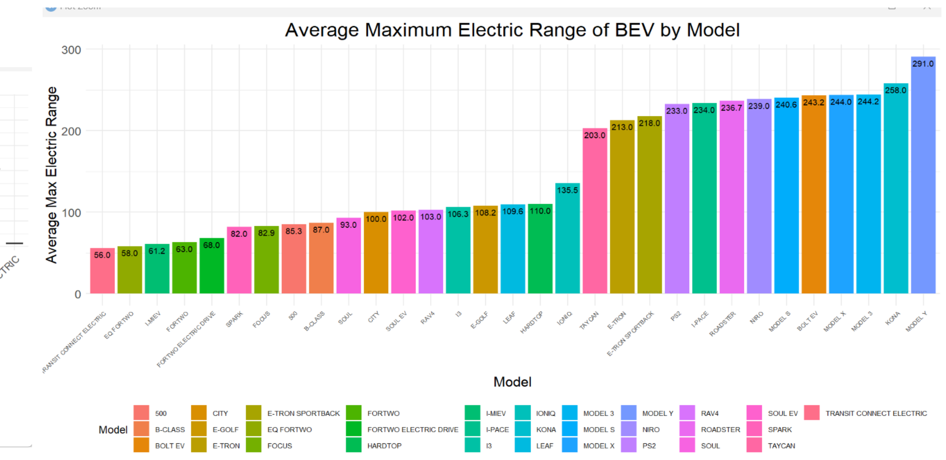


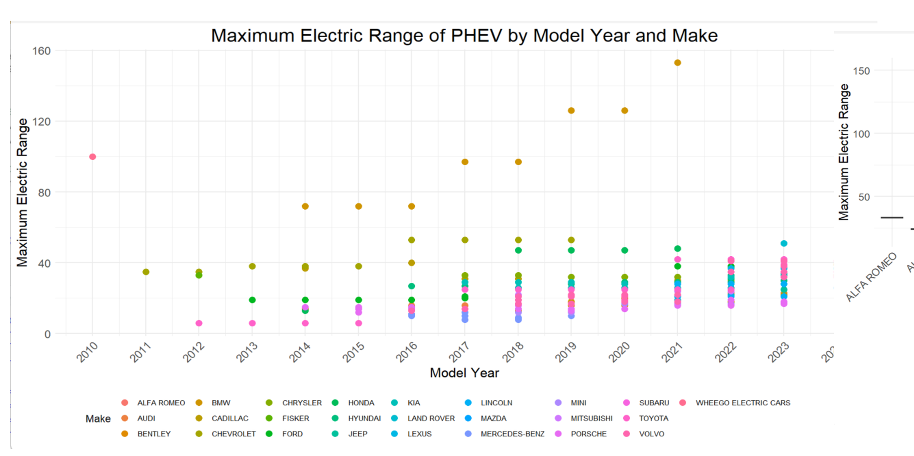


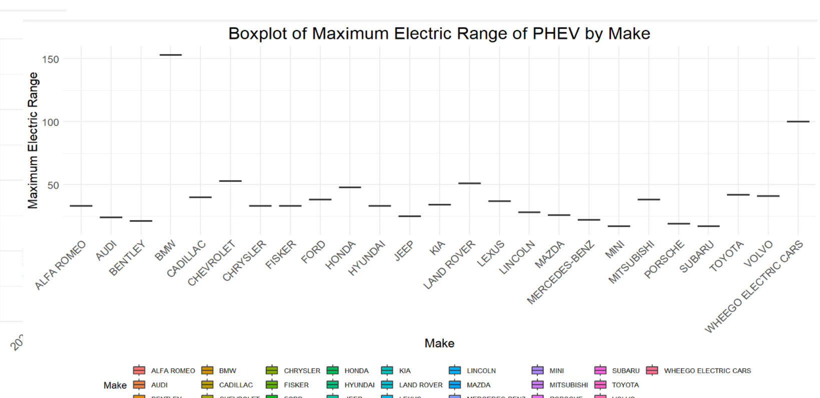


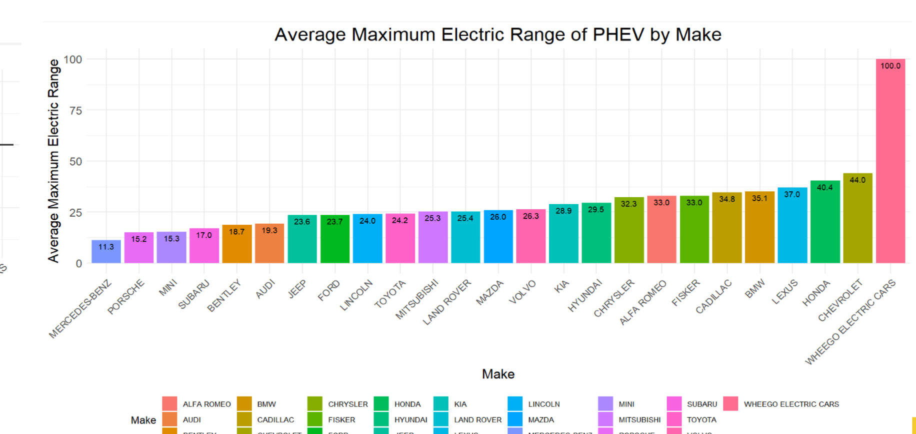


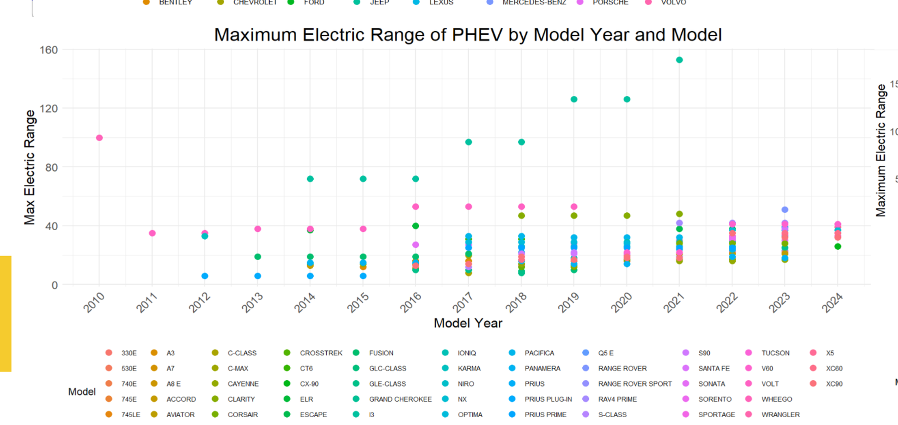


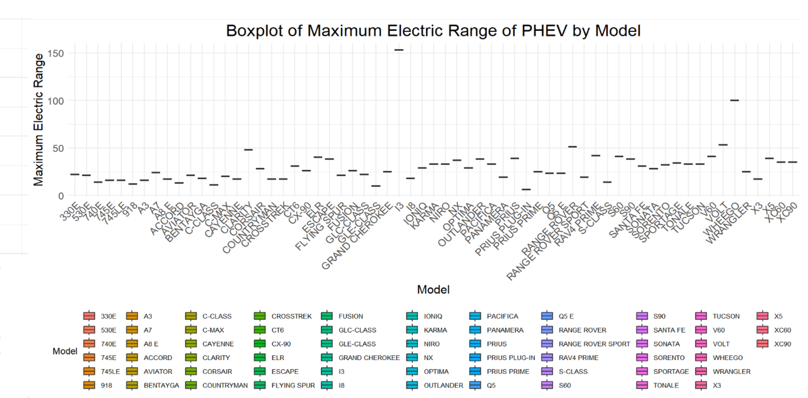


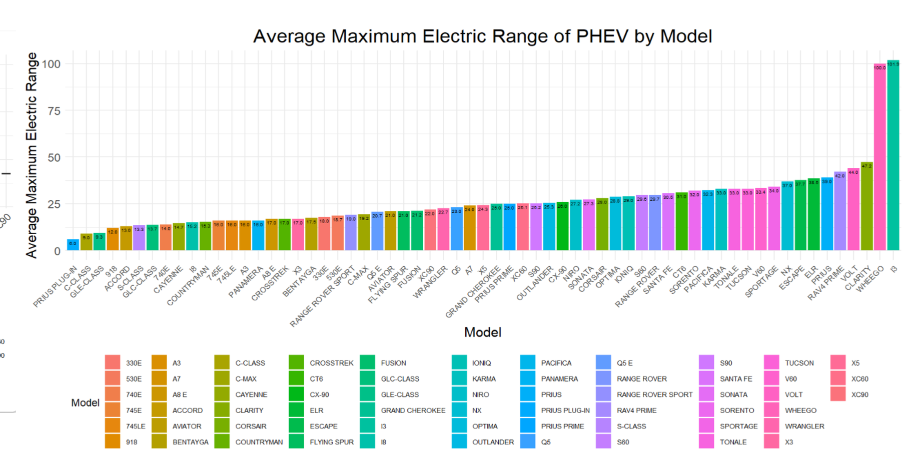






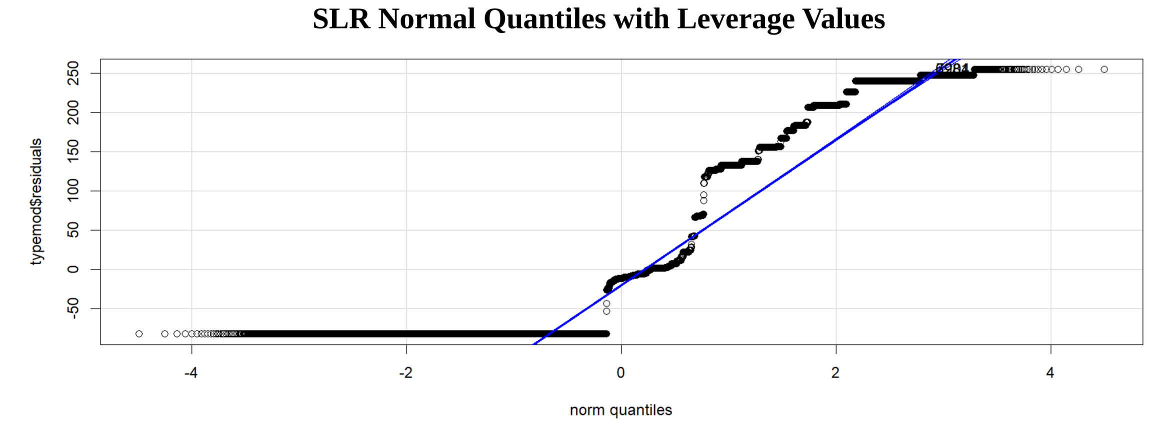




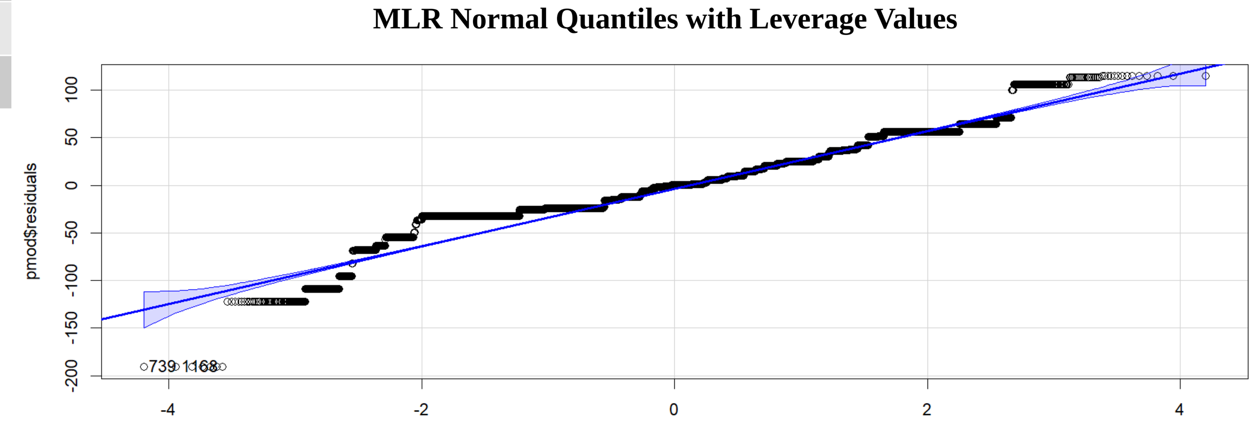


Regression Model	X=EV Type (BEV or PHEV) Y=Max Electric Range
Residual Standard Error	94.68
Degrees of Freedom	143267
Adjusted R-Squared	0.05
F-Statistic	7478
P-Value	2.2e-16

SLR Normal Quantiles with Leverage Values



MLR Normal Quantiles with Leverage Values



MLR Results	X=Make, Model Year Y=Max Electric Range
Residual Standard Error	27.3
Degrees of Freedom	37413
Adjusted R-Squared	0.8633
F-Statistic	11820
P-Value	2.2e-16
VIF Value	1.404989

DISCUSSION

- Our findings indicate that a majority of EV owners in WA own BEV rather than PHEV, suggesting the possible benefit of focusing innovation efforts and marketing on this EV type.
- While growth has occurred with both EV types, model choice is also greater in BEV manufacturers than in PHEVs. The battery range of BEV is greater on average and measured as a maximum value. These measurements of range do not consider the extended non-electric range of PHEV's fuel engine component, however. A more thorough study of EV market demand will necessitate the addition of sample data outside of WA.
- The two linear regression models we built indicated that model year and make of EV are more reliable predictors of electric range than EV type alone, but both regression models indicate issues with constant variance that will need to be addressed, however these assumption violations are less pertinent as our sample size is very large. The inclusion of additional predictors, like model cost (initial and maintenance) in order to properly evaluate the potential role of each EV type in market demand could also help the model.
- The emerging nature of new EV make/models, and relative novel technology in both BEV and PHEV options, suggests continued study is needed to truly assess the viability of both EV options and the role they can play in efforts to combat climate change.
- Future studies should incorporate machine learning

R CODE

```
## Select the variables of interest using "select" functionelectric_veh_washington1 <- electric_veh_washington %>% select(Model_Year,Make,Model,Electric_Vehicle_Type,Electric_Range)
# Convert categorical variables to factors using "lapply" functioncategorical_vars <- c("Make","Model","Electric_Vehicle_Type")electric_veh_washington1[categorical_vars] <- lapply(electric_veh_washington1[categorical_vars], as.factor)

# Create a scatter plot of Model_Year and Make for BEV
p3 <- max_range_Bev %>%ggplot(aes(x = Model_Year, y = Max_Electric_Range_Bev, color = Make)) +  geom_point(size=3) +  labs(title = "Maximum Electric Range of BEV by ModelYear and Make",x = "Model Year",y = "Max Electric Range") + scale_x_continuous(breaks = seq(2008, 2024, by = 1)) +  scale_y_continuous(breaks = seq(0, 400, by = 50)) +  theme_minimal() +  theme(plot.title = element_text(hjust = 0.5,size=20),  axis.title = element_text(size = 15),  legend.text = element_text(size = 8), axis.text.x = element_text( angle = 45,hjust = 1,size=12),  axis.text.y = element_text(size=12),  legend.position = 'bottom')+  guides(color = guide_legend(ncol = 11))

## Fit full Model on training dataframe
pmod<-lm(Electric_Range~Model_Year+Make, data=train_data)pmod# Summary of the regression model

#dummy code cat type var
electric_veh_washington1$dummy_code <- ifelse(electric_veh_washington1$Electric_Vehicle_Type == "Battery Electric Vehicle (BEV)", 1, 0)
#slr for vehicle type bev vs phev
typemod<- lm(Electric_Range~dummy_code, data=electric_veh_washington1)summary(typemod)
```


Prativa Basnet
MSAS
Graduation Spring 2024

Brandi Jones
MSAS
Graduation Fall 2024

````
---
title: "Project"
author: "Prativa Basnet"
date: "December 6, 2023"
output:
  word_document: default
  html_document: default
  pdf_document: default
editor_options:
  chunk_output_type: console
---

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
```


```{r setup, include=FALSE}
# Install packages tinytex to knit the code with result

options(repos = c(CRAN = "https://cran.rstudio.com/"))
library(tinytex)
```

```{r}
# Packages
library(tidyverse)
library(car)
library(ranger)
library(caret)
```

```{r}
# Read the dataset
electric_veh<- readr::read_csv("Electric_Vehicle_Population_Data_Project.csv")
dplyr::glimpse(electric_veh)
summary(electric_veh)

# Filter vehicles registered in Washington State
electric_veh_washington <- electric_veh %>%
  filter(State == "WA")

glimpse(electric_veh_washington)
summary(electric_veh_washington)
```




```{r}
````
````

````r
## Select the variables of interest using "select" function
electric_veh_washington1 <- electric_veh_washington %>%
  select(Model_Year,Make,Model,Electric_Vehicle_Type,Electric_Range)


# Check missing values by variable in electric_veh_washington1 dataset
colSums(is.na(electric_veh_washington1))


#check the number of observations
num_observations <- nrow(electric_veh_washington1)
num_observations

```


```{r}
# Convert categorical variables to factors using "lapply" function
categorical_vars <- c("Make","Model","Electric_Vehicle_Type")
electric_veh_washington1[categorical_vars] <-
lapply(electric_veh_washington1[categorical_vars], as.factor)

glimpse(electric_veh_washington1)
summary(electric_veh_washington1)

# Check number of levels for each variable in data frame
sapply(electric_veh_washington1, function(x) length(unique(x)))
```



```{r}
# Data Visualization for electric vehicle type
electric_veh_count <- electric_veh_washington1 %>%
  group_by(Electric_Vehicle_Type) %>%
  count()


# Create a bar chart for Electric Vehicle Types
p1<-electric_veh_count %>%
  ggplot(aes(x = Electric_Vehicle_Type , y = n,    fill=Electric_Vehicle_Type)) +
  geom_bar(stat = 'identity') +
  geom_text(aes(label = sprintf("%.0f", n)), size = 6, color = 'black', vjust = 1.5)
+
  labs(title = "Distribution of Electric Vehicle Type ",
       x = "Electric Vehicle Type",
       y = "Count") +
  theme_minimal()+
  theme(plot.title = element_text(hjust = 0.5,size=12),
        axis.text.x = element_text(angle = 20, hjust = 1,size=15),
        axis.text.y = element_text(hjust = 1))
````

~~~~
p1
```

```{r}
# Filter the data for Battery Electric Vehicles (BEV)
Bev_data <- electric_veh_washington1 %>%
  filter(Electric_Vehicle_Type == "Battery Electric Vehicle (BEV)")
summary(Bev_data)


#check the number of observations
num_observations <- nrow(Bev_data)
num_observations
```

```{r}
# Filter out rows with electric_range not equal to Zero (electric range has not been
researched.)
Bev_data1 <- Bev_data %>%
  filter(Electric_Range!=0)

summary(Bev_data1)
```




```{r}
write.csv(Bev_data1, "Bev_data1.csv")
```
#Checking for outliers
```{r}
# Calculate the IQR and identify outliers
q1_BEV <- quantile(Bev_data1$Electric_Range, 0.25)
q3_BEV <- quantile(Bev_data1$Electric_Range, 0.75)
iqr_BEV <- q3_BEV - q1_BEV
lower_bound_BEV <- q1_BEV - 1.5 * iqr_BEV
upper_bound_BEV <- q3_BEV + 1.5 * iqr_BEV

# Identify outliers
outliers_BEV <- Bev_data1$Electric_Range[Bev_data1$Electric_Range < lower_bound_BEV
| Bev_data1$Electric_Range > upper_bound_BEV]
outliers_BEV
```

# Scatterplot of Model_Year and Make
```{r}
# Create a scatter plot of Model_Year and Make for BEV
p2 <- Bev_data1 %>%
ggplot(aes(x = Model_Year, y = Electric_Range, color = Make)) +
  geom_point(size=3) +
~~~~

````
  labs(title = "Electric Range of BEVs by ModelYear and Make",x = "Model Year",y = 
"Electric Range") +
  scale_x_continuous(breaks = seq(1997, 2021, by = 1)) +
  scale_y_continuous(breaks = seq(0, 400, by = 50)) +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5,size=20),
  axis.title = element_text(size = 15),
  legend.text = element_text(size = 8),
  axis.text.x = element_text( angle = 45,hjust = 1,size=12),
  axis.text.y = element_text(size=12),
  legend.position = 'bottom')+
  guides(color = guide_legend(ncol = 11))
p2
```

# Scatterplot of Model_Year and Model
```{r}
# Create a scatter plot of Model_Year and Make for BEV
p3 <- Bev_data1 %>%
ggplot(aes(x = Model_Year, y = Electric_Range, color = Model)) +
  geom_point(size=3) +
  labs(title = "Electric Range of BEVs by ModelYear and Model",x = "Model Year",y = 
"Electric Range") +
  scale_x_continuous(breaks = seq(1997, 2021, by = 1)) +
  scale_y_continuous(breaks = seq(0, 400, by = 50)) +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5,size=20),
  axis.title = element_text(size = 15),
  legend.text = element_text(size = 6),
  axis.text.x = element_text( angle = 45,hjust = 1,size=12),
  axis.text.y = element_text(size=12),
  legend.position = 'bottom')+
  guides(color = guide_legend(ncol = 11))
p3
```


# Total number of Battery Electric Vehicle (BEV) for different Model_Year
```{r}
# Count the number of vehicles in each year
vehicle_counts <- Bev_data1 %>%
  group_by(Model_Year) %>%
  summarize(Count = n())

# View the vehicle counts
print(vehicle_counts)

# Create a horizontal bar chart BEV vs Year
p2 <- vehicle_counts %>%
  ggplot( aes(x=Count,y=reorder(Model_Year,Count)))+
````

````
  geom_bar(stat = 'identity', fill="orange") +
  #geom_text(aes(label= Count), size = 2,color='black',hjust=1.0) +
  labs(title = "Total Number of Battery Electric Vehicle (BEV) by Model Year",
       x = "Count",
       y = "Model Year") +
  scale_x_continuous(breaks = seq(0, 26000, by = 2000)) +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5,size=20),
  axis.title = element_text(size = 15),
  axis.text.x = element_text(angle = 45, hjust = 1,size=10),
  axis.text.y = element_text(size=10))

p2
```



```{r}
#write.csv(max_range_Bev, "max_range_Bev.csv")
```



#Full regression model
```{r}
Analy_data_BEV_Full<-Bev_data1 %>%
  select(Model_Year,Make,Model,Electric_Range)
glimpse(Analy_data_BEV_Full)
```


#Split dataset
```{r}
## Split the Data into a 80/20 Training/Testing Set ##
set.seed(123)
p_split_BEV_Full <- rsample::initial_split(Analy_data_BEV_Full,prop=0.80)
train_data_BEV_Full <- rsample::training(p_split_BEV_Full)
test_data_BEV_Full <- rsample::testing(p_split_BEV_Full)

```


```{r}
#Multiple Regression
## Fit full Model on training dataframe ##
pmod<-lm(Electric_Range~., data=train_data_BEV_Full)
# Summary of the regression model
summary(pmod)

```




```{r}
````

````
## Check Assumptions ##

## Normality ##
pmod$residuals |>
  ggpubr::ggqqplot() +
  ggtitle("QQ Plot of Residuals") +
  xlab("Theoretical") +
  ylab("Sample")

#pmod$residuals |>
  #rstatix::shapiro_test()



## Constant Variance ##

ggplot() +
  geom_point(aes(x=fitted(pmod),y=rstudent(pmod))) +
  geom_hline(yintercept=3,color='red') +
  geom_hline(yintercept=-3,color='red') +
  geom_hline(yintercept=0,color='blue') +
  labs(y="Studentized Residuals",
       x="Fitted Values") +
  theme_classic()

#studentized Breusch-Pagan test
lmtest::bptest(pmod)

## VIF ##

#car::vif(pmod)

# Check for linear interdependence
aliased_terms <- alias(pmod)
aliased_terms
```

#BEVs range predictor by Model_Year,Make
```{r}
Analy_data_BEV_M1<-Bev_data1 %>%
  select(Model_Year,Make,Electric_Range)
glimpse(Analy_data_BEV_M1)
```

#Split dataset
```{r}
## Split the Data into a 80/20 Training/Testing Set ##
set.seed(123)
p_split_BEV_M1 <- rsample::initial_split(Analy_data_BEV_M1,prop=0.80)
train_data_BEV_M1 <- rsample::training(p_split_BEV_M1)
````

````
test_data_BEV_M1 <- rsample::testing(p_split_BEV_M1)

```
# Random Forest
```{r}

## Identify hyper-parameters
# 1.mtry (Randomly selected predictor)
# 2.splitrule
# 3.min.node size
# Specify values for hyper-parameters
# mtry --> generally start with around square-root of variables (2 parameters)
sqrt(2) #1.4

my.mtry = c(1,2)
#splitrule
my.rule = "variance"
# min.node.size --> default is 5 for regression
my.nodes = c(5,7,9,11)
# create tuning grid
my.grid = expand.grid(mtry = my.mtry,
                      splitrule = my.rule,
                      min.node.size = my.nodes)
my.grid

#Select an appropriate evaluation metric
my.metric = "RMSE"

#Train model over hyperparameters
set.seed(2)
rf.tune_BEV_M1 = caret::train(Electric_Range ~ ., data = train_data_BEV_M1,
                       method = "ranger",
                       metric = my.metric ,
                       importance = "impurity",
                       trControl = trainControl(method = "cv", number = 10) ,
                       tuneGrid = my.grid ) # grid of hyperparameters
rf.tune_BEV_M1

rf.tune_BEV_M1$results %>% arrange(RMSE)

## Select best hyperparameters based on evaluation metric
rf.tune_BEV_M1$bestTune

```


```{r}
# Random forest model variable importance scores were based on improvement in
variance impurity
````

```
# Extract scaled variable importance scores
rf_importance_BEV_M1 <- varImp(rf.tune_BEV_M1)
print(rf_importance_BEV_M1)
vip::vip(rf.tune_BEV_M1)
```

```{r}
# Create a data frame for Random Forest variable importance
rf_importance_BEV_M1_df <- data.frame( Variable =
rownames(rf_importance_BEV_M1$importance), Importance =
rf_importance_BEV_M1$importance$Overall)

# Arrange in order
rf_importance_BEV_M1_df<-rf_importance_BEV_M1_df %>%
  arrange(desc(Importance))
```

```{r}
p <- rf_importance_BEV_M1_df %>%
  filter(Importance > 1) %>%  # Filter rows where Importance is greater than 1
  ggplot(aes(x = reorder(Variable, desc(Importance)), y = Importance)) +
  geom_bar(stat = "identity", fill = "orange") +
  geom_text(aes(label = ifelse(Importance > 0, sprintf("%.1f", Importance), "")),
size = 6, color = 'black', vjust = 1.5) +
  labs(title = "Random Forest Scaled Variable Importance (Variance Impurity)", x =
"Variable", y = "Importance") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5, size = 15),
        axis.text.x = element_text(angle = 45, hjust = 1,size=15))

print(p)

```

```{r}
## Use tuned model to predict test sample
pred_test_sample_BEV_M1 = predict(rf.tune_BEV_M1, test_data_BEV_M1)
head(pred_test_sample_BEV_M1)
```

```{r}
## Graphically evaluate the predicted & observed values ##

ggplot() +
  geom_point(aes(pred_test_sample_BEV_M1,test_data_BEV_M1$Electric_Range )) +
  labs(title = "Observed & Predicted values with Predictor Model_Year and Make",
       x = "Predicted Values",
```

````
       y = "Observed Values") + theme_classic()
```

```{r}
## MSE,RMSE, Rsquared,and MAE ##

MSE <- mean((test_data_BEV_M1$Electric_Range  - pred_test_sample_BEV_M1)^2)

RMSE <- sqrt(MSE)
Rsquared <- 1 - MSE/var(test_data_BEV_M1$Electric_Range )

MAE <- mean(abs(test_data_BEV_M1$Electric_Range  - pred_test_sample_BEV_M1))

c(MSE,RMSE,MAE,Rsquared)

```


#Battery Electric Vehicles for predictor Model_Year,Model

```{r}
Analy_data_BEV_M2 <- Bev_data1  %>%
  ungroup() %>%
  select(Model_Year, Model, Electric_Range)

glimpse(Analy_data_BEV_M2)

# Check missing values by variable
colSums(is.na(Analy_data_BEV_M2))

```




#Split dataset
```{r}
## Split the Data into a 80/20 Training/Testing Set ##
set.seed(123)
p_split_BEV_M2 <- rsample::initial_split(Analy_data_BEV_M2,prop=0.80)
train_data_BEV_M2 <- rsample::training(p_split_BEV_M2)
test_data_BEV_M2 <- rsample::testing(p_split_BEV_M2)
```


# Random Forest
```{r}
## Identify hyper-parameters
# 1.mtry (Randomly selected predictor)
# 2.splitrule
````

````
# 3.min.node size
# Specify values for hyper-parameters
# mtry --> generally start with around square-root of variables (2 parameters)
sqrt(2) #1.4

my.mtry = c(1,2)
#splitrule
my.rule = "variance"
# min.node.size
my.nodes = c(5,7,9,11)
# create tuning grid
my.grid = expand.grid(mtry = my.mtry,
                      splitrule = my.rule,
                      min.node.size = my.nodes)
my.grid

#Select an appropriate evaluation metric
my.metric = "RMSE"
## 5. Train model over hyperparameters
#?caret::train
set.seed(2)
rf.tune_BEV_M2 = caret::train(Electric_Range~ ., data = train_data_BEV_M2,
                       method = "ranger",
                       metric = my.metric ,
                       importance = "impurity",
                       trControl = trainControl(method = "cv", number = 10) ,
                       tuneGrid = my.grid ) # grid of hyperparameters
rf.tune_BEV_M2

rf.tune_BEV_M2$results %>% arrange(RMSE)

## Select best hyperparameters based on evaluation metric
rf.tune_BEV_M2$bestTune

```

```{r}
# Random forest model variable importance scores were based on improvement in gini
impurity # Extract scaled variable importance scores
rf_importance_BEV_M2 <- varImp(rf.tune_BEV_M2)
print(rf_importance_BEV_M2)
vip::vip(rf.tune_BEV_M2)
```

```{r}
# Create a data frame for Random Forest variable importance
rf_importance_BEV_M2_df <- data.frame( Variable =
rownames(rf_importance_BEV_M2$importance), Importance =
rf_importance_BEV_M2$importance$Overall)
````

````
# Arrange in order
rf_importance_BEV_M2_df<-rf_importance_BEV_M2_df %>%
  arrange(desc(Importance))
```

```{r}
p <- rf_importance_BEV_M2_df %>%
  filter(Importance > 2) %>%  # Filter rows where Importance is greater than 1
  ggplot(aes(x = reorder(Variable, desc(Importance)), y = Importance)) +
  geom_bar(stat = "identity", fill = "orange") +
  geom_text(aes(label = ifelse(Importance > 0, sprintf("%.1f", Importance), "")),
size = 2.5, color = 'black', vjust = 1.0) +
  labs(title = "Random Forest Scaled Variable Importance (Variance Impurity ", x =
"Variable", y = "Importance") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5, size = 15),
        axis.text.x = element_text(angle = 45, hjust = 1))

print(p)
```




```{r}
## Use tuned model to predict test sample
pred_test_sample_BEV_M2 = predict(rf.tune_BEV_M2, test_data_BEV_M2)
head(pred_test_sample_BEV_M2)
```



```{r}
## Graphically evaluate the predicted & observed values ##

ggplot() +
  geom_point(aes(pred_test_sample_BEV_M2,test_data_BEV_M2$Electric_Range)) +
  labs(title = "Observed & Predicted values with Predictor Model_Year and Model",
       x = "Predicted Values",
       y = "Observed Values") +
  theme_classic()+
  theme(plot.title = element_text(hjust = 0.5, size = 10))
```

```{r}
## MSE,RMSE, Rsquared,and MAE ##

MSE <- mean((test_data_BEV_M2$Electric_Range - pred_test_sample_BEV_M2)^2)

RMSE <- sqrt(MSE)
Rsquared <- 1 - MSE/var(test_data_BEV_M2$Electric_Range)
````

MAE <- mean(abs(test_data_BEV_M2$Electric_Range - pred_test_sample_BEV_M2))

c(MSE,RMSE,MAE,Rsquared)

```

# Plug-in Hybrid Electric Vehicle (PHEV)
```{r}
# Filter the data for Plug-in Hybrid Electric Vehicle (PHEV)
PHEV_data <- electric_veh_washington1 %>%
  filter(Electric_Vehicle_Type == "Plug-in Hybrid Electric Vehicle (PHEV)")
summary(PHEV_data)

#check the number of observations
num_observations <- nrow(PHEV_data)
num_observations
```

```{r}
write.csv(PHEV_data, "PHEV_data.csv")
```

#Checking for outliers
```{r}
# Calculate the IQR and identify outliers
q1_PHEV <- quantile(PHEV_data$Electric_Range, 0.25)
q3_PHEV <- quantile(PHEV_data$Electric_Range, 0.75)
iqr_PHEV <- q3_PHEV - q1_PHEV
lower_bound_PHEV <- q1_PHEV - 1.5 * iqr_PHEV
upper_bound_PHEV <- q3_PHEV + 1.5 * iqr_PHEV

# Identify outliers
outliers_PHEV <- PHEV_data$Electric_Range[PHEV_data$Electric_Range <
lower_bound_PHEV | PHEV_data$Electric_Range > upper_bound_PHEV]
outliers_PHEV
```

Use median value to take care of outliers
```{r}
# Calculate the median
median_range <- median(PHEV_data$Electric_Range)
#median_range<- 2*median_range
median_range

````
# Filter the data for Electric_Range >median_range (26)
PHEV_data1 <- PHEV_data %>%
  filter(Electric_Range > median_range)
summary(PHEV_data1)
```

Rechecking outliers
```{r}
# Calculate the IQR and identify outliers
q1_PHEV1 <- quantile(PHEV_data1$Electric_Range, 0.25)
q3_PHEV1 <- quantile(PHEV_data1$Electric_Range, 0.75)
iqr_PHEV1 <- q3_PHEV1 - q1_PHEV1
lower_bound_PHEV1 <- q1_PHEV1 - 1.5 * iqr_PHEV1
upper_bound_PHEV1 <- q3_PHEV1 + 1.5 * iqr_PHEV1

# Identify outliers
outliers_PHEV1 <- PHEV_data1$Electric_Range[PHEV_data1$Electric_Range <
lower_bound_PHEV1 | PHEV_data1$Electric_Range > upper_bound_PHEV1]
outliers_PHEV1
```

Handling the outliers
```{r}
# Filter out rows with electric_range less than 126
#PHEV_data2 <- PHEV_data1 %>%
  #filter(Electric_Range< 126)
#glimpse(PHEV_data2)
```


```{r}

write.csv(PHEV_data2, "PHEV_data2.csv")

```
Scatterplot of Model_Year and Make
```{r}
# Create a scatter plot of Model_Year and Make for BEV
p3 <- PHEV_data1 %>%
ggplot(aes(x = Model_Year, y = Electric_Range, color = Make)) +
  geom_point(size=3) +
  labs(title = "Maximum Electric Range of BEV by ModelYear and Make",x = "Model
Year",y = "Max Electric Range") +
  scale_x_continuous(breaks = seq(2010, 2019, by = 1)) +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5,size=20),
  axis.title = element_text(size = 15),
  legend.text = element_text(size = 8),
  axis.text.x = element_text( angle = 45,hjust = 1,size=12),
````

````
  axis.text.y = element_text(size=12),
  legend.position = 'bottom')+
  guides(color = guide_legend(ncol = 11))
p3
```

# Scatterplot of Model_Year and Maodel
```{r}
# Create a scatter plot of Model_Year and Make for BEV
p3 <- PHEV_data1 %>%
ggplot(aes(x = Model_Year, y = Electric_Range, color = Model)) +
  geom_point(size=3) +
  labs(title = "Maximum Electric Range of BEV by ModelYear and Make",x = "Model
Year",y = "Max Electric Range") +
  scale_x_continuous(breaks = seq(1997, 2024, by = 1)) +
  scale_y_continuous(breaks = seq(0, 400, by = 50)) +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5,size=20),
  axis.title = element_text(size = 15),
  legend.text = element_text(size = 8),
  axis.text.x = element_text( angle = 45,hjust = 1,size=12),
  axis.text.y = element_text(size=12),
  legend.position = 'bottom')+
  guides(color = guide_legend(ncol = 11))
p3
```



```{r}
write.csv(PHEV_data2, "PHEV_data2.csv")
```




```{r}
# Count the number of vehicles in each year
PHEV_counts <- PHEV_data1 %>%
  group_by(Model_Year) %>%
  summarize(Count = n())

# View the vehicle counts
print(PHEV_counts)
```

```{r}
# Create a horizontal bar chart PHEV vs Year
p9 <- PHEV_counts %>%
  ggplot( aes(x=Count,y=reorder(Model_Year,Count)))+
  geom_bar(stat = 'identity', fill="orange") +
  #geom_text(aes(label= Count), size = 2,color='black',hjust=1.0) +
````

````
  labs(title = "Total Number of  Plug-in Hybrid Electric Vehicle (PHEV) by Model
Year",
       x = "Count",
       y = "Model Year") +
  scale_x_continuous(breaks = seq(0, 5000, by = 1000)) +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5,size=20),
  axis.title = element_text(size = 15),
  axis.text.x = element_text(angle = 45, hjust = 1,size=10),
  axis.text.y = element_text(size=10))

p9
```


#Full regression model
```{r}
Analy_data_PHEV_Full<-PHEV_data1 %>%
  select(Model_Year,Make,Model,Electric_Range)
glimpse(Analy_data_PHEV_Full)

# Check missing values by variable in train_data and test_data
colSums(is.na(Analy_data_PHEV_Full))

```


#Split dataset
```{r}
## Split the Data into a 80/20 Training/Testing Set ##
set.seed(123)
p_split_PHEV_Full <- rsample::initial_split(Analy_data_PHEV_Full,prop=0.80)
train_data_PHEV_Full <- rsample::training(p_split_PHEV_Full)
test_data_PHEV_Full <- rsample::testing(p_split_PHEV_Full)
```

```{r}
#Multiple Regression
## Fit full Model on training dataframe ##
pmod<-lm(Electric_Range~., data=train_data_PHEV_Full)
# Summary of the regression model
summary(pmod)

```




```{r}
## Check Assumptions ##

## Normality ##
````

````
pmod$residuals |>
  ggpubr::ggqqplot()

#pmod$residuals |>
  #rstatix::shapiro_test()

## Constant Variance ##
ggplot() +
  geom_point(aes(x=fitted(pmod),y=rstudent(pmod))) +
  geom_hline(yintercept=3,color='red') +
  geom_hline(yintercept=-3,color='red') +
  geom_hline(yintercept=0,color='blue') +
  labs(y="Studentized Residuals",
       x="Fitted Values") +
  theme_classic()

#studentized Breusch-Pagan test
lmtest::bptest(pmod)

## VIF ##

car::vif(pmod)

```

#Plug-in Hybrid Electric Vehicles from Year 2010 to 2024 for predictor Model_Year
and Make
```{r}
Analy_data_PHEV_M1<-PHEV_data1 %>%
  select(Model_Year,Make,Electric_Range)
glimpse(Analy_data_PHEV_M1)

# Check missing values by variable in train_data and test_data
colSums(is.na(Analy_data_PHEV_M1))

```



#Split dataset
```{r}
## Split the Data into a 80/20 Training/Testing Set ##
set.seed(123)
p_split_PHEV_M1 <- rsample::initial_split(Analy_data_PHEV_M1,prop=0.80)
train_data_PHEV_M1 <- rsample::training(p_split_PHEV_M1)
test_data_PHEV_M1 <- rsample::testing(p_split_PHEV_M1)

# Check missing values by variable in train_data and test_data
colSums(is.na(train_data_PHEV_M1))
````

~~~
colSums(is.na(test_data_PHEV_M1))
```


# Random Forest
```{r}
## Identify hyper-parameters
# 1.mtry (Randomly selected predictor)
# 2.splitrule
# 3.min.node size
# Specify values for hyper-parameters # mtry --> generally start with around
square-root of variables (parameters)
sqrt(2) #1.4

my.mtry = c(1,2)
#splitrule
my.rule = "variance"
# min.node.size
my.nodes = c(5,7,9,11)
# create tuning grid
my.grid = expand.grid(mtry = my.mtry,
                      splitrule = my.rule,
                      min.node.size = my.nodes)
my.grid

#Select an appropriate evaluation metric
my.metric = "RMSE"

## 5. Train model over hyperparameters
set.seed(2)
rf.tune_PHEV_M1 = caret::train(Electric_Range~ ., data = train_data_PHEV_M1,
                       method = "ranger",
                       metric = my.metric ,
                       importance = "impurity",
                       trControl = trainControl(method = "cv", number = 10) ,
                       tuneGrid = my.grid ) # grid of hyperparameters
rf.tune_PHEV_M1

rf.tune_PHEV_M1$results %>% arrange(RMSE)

## Select best hyperparameters based on evaluation metric
rf.tune_PHEV_M1$bestTune

```

```{r}
# Random forest model variable importance scores were based on improvement in gini
impurity # Extract scaled variable importance scores
rf_importance_PHEV_M1 <- varImp(rf.tune_PHEV_M1)
print(rf_importance_PHEV_M1)
~~~

````
vip::vip(rf.tune_PHEV_M1)
```

```{r}
# Create a data frame for Random Forest variable importance
rf_importance_PHEV_M1_df <- data.frame( Variable =
rownames(rf_importance_PHEV_M1$importance), Importance =
rf_importance_PHEV_M1$importance$Overall)

#Arrange in order
rf_importance_PHEV_M1_df<-rf_importance_PHEV_M1_df %>%
  arrange(desc(Importance))
```

```{r}
p <- rf_importance_PHEV_M1_df %>%
  filter(Importance > 1.2) %>%  # Filter rows where Importance is greater than 1
  ggplot(aes(x = reorder(Variable, desc(Importance)), y = Importance)) +
  geom_bar(stat = "identity", fill = "orange") +
  geom_text(aes(label = ifelse(Importance > 0, sprintf("%.1f", Importance), "")),
size = 2.5, color = 'black', vjust = 1.0) +
  labs(title = "Random Forest Scaled Variable Importance (Variance Impurity ", x =
"Variable", y = "Importance") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5, size = 15),
        axis.text.x = element_text(angle = 45, hjust = 1))

print(p)
```



```{r}
## Use tuned model to predict test sample
pred_test_sample_PHEV_M1 = predict(rf.tune_PHEV_M1, test_data_PHEV_M1)
head(pred_test_sample_PHEV_M1)
```

```{r}
## Graphically evaluate the predicted & observed values ##

ggplot() +
geom_point(aes(pred_test_sample_PHEV_M1,test_data_PHEV_M1$Electric_Range)) +
  labs(title = "Observed & Predicted values with Predictor Model_Year and Model",
       x = "Predicted Values",
       y = "Observed Values") +
  theme_classic()+
  theme(plot.title = element_text(hjust = 0.5, size = 10))
```

```{r}
````

````
## MSE/RMSE, Rsquared,MAE ##

MSE <- mean((test_data_PHEV_M1$Electric_Range - pred_test_sample_PHEV_M1)^2)

RMSE <- sqrt(MSE)
Rsquared <- 1 - MSE/var(test_data_PHEV_M1$Electric_Range)

MAE <- mean(abs(test_data_PHEV_M1$Electric_Range - pred_test_sample_PHEV_M1))

c(MSE,RMSE,MAE,Rsquared)

```


#Plug-in Hybrid Electric Vehicles from Year 2010 to 2024 for predictor Model_Year and Model
```{r}
Analy_data_PHEV_M2<-PHEV_data1  %>%
  ungroup() %>%
  select(Model_Year,Model,Electric_Range)
glimpse(Analy_data_PHEV_M2)

# Check missing values by variable in train_data and test_data
colSums(is.na(Analy_data_PHEV_M2))

```




#Split dataset
```{r}
## Split the Data into a 80/20 Training/Testing Set ##
set.seed(123)
p_split_PHEV_M2 <- rsample::initial_split(Analy_data_PHEV_M2,prop=0.80)
train_data_PHEV_M2 <- rsample::training(p_split_PHEV_M2)
test_data_PHEV_M2 <- rsample::testing(p_split_PHEV_M2)

```



# Random Forest
```{r}
## Identify hyper-parameters
# 1.mtry (Randomly selected predictor)
# 2.splitrule
# 3.min.node size
# Specify values for hyper-parameters
# mtry
sqrt(2) #1.4
````

~~~
my.mtry = c(1,2)
#splitrule
my.rule = "variance"
# min.node.size
my.nodes = c(5,7,9,11)

# create tuning grid
my.grid = expand.grid(mtry = my.mtry,
                      splitrule = my.rule,
                      min.node.size = my.nodes)
my.grid

#Select an appropriate evaluation metric
my.metric = "RMSE"

## 5. Train model over hyperparameters
#?caret::train
set.seed(2)
rf.tune_PHEV_M2 = caret::train(Electric_Range~ ., data = train_data_PHEV_M2,
                       method = "ranger",
                       metric = my.metric ,
                       importance = "impurity",
                       trControl = trainControl(method = "cv", number = 10) ,
                       tuneGrid = my.grid ) # grid of hyperparameters
rf.tune_PHEV_M2

rf.tune_PHEV_M2$results %>% arrange(RMSE)

## Select best hyperparameters based on evaluation metric
rf.tune_PHEV_M2$bestTune

```

```{r}
# Random forest model variable importance scores were based on improvement in gini
impurity # Extract scaled variable importance scores
rf_importance_PHEV_M2 <- varImp(rf.tune_PHEV_M2)
print(rf_importance_PHEV_M2)
vip::vip(rf.tune_PHEV_M2)
```

```{r}
# Create a data frame for Random Forest variable importance
rf_importance_PHEV_M2_df <- data.frame( Variable =
rownames(rf_importance_PHEV_M2$importance), Importance =
rf_importance_PHEV_M2$importance$Overall)
# Arrange in order
rf_importance_PHEV_M2_df<-rf_importance_PHEV_M2_df %>%
  arrange(desc(Importance))
```
~~~

````
```{r}
p <- rf_importance_PHEV_M2_df %>%
  filter(Importance > 1.2) %>%  # Filter rows where Importance is greater than 1
  ggplot(aes(x = reorder(Variable, desc(Importance)), y = Importance)) +
  geom_bar(stat = "identity", fill = "orange") +
  geom_text(aes(label = ifelse(Importance > 0, sprintf("%.1f", Importance), "")),
size = 2.5, color = 'black', vjust = 1.0) +
  labs(title = "Random Forest Scaled Variable Importance (Variance Impurity)", x =
"Variable", y = "Importance") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5, size = 15),
        axis.text.x = element_text(angle = 45, hjust = 1))

print(p)
```

```{r}
## Use tuned model to predict test sample
pred_test_sample_PHEV_M2 = predict(rf.tune_PHEV_M2, test_data_PHEV_M2)
head(pred_test_sample_PHEV_M2)
```

```{r}
## Graphically evaluate the predicted & observed values ##

ggplot() +
  geom_point(aes(pred_test_sample_PHEV_M2,test_data_PHEV_M2$Electric_Range)) +
  labs(title = "Pred and Obs values of Reg RF Model with Predictor Model_Year and
Model",
       x = "Predicted Values",
       y = "Observed Values") +
  theme_classic()+
  theme(plot.title = element_text(hjust = 0.5, size = 10))
```

```{r}
## MSE,RMSE, Rsquared,MAE ##

MSE <- mean((test_data_PHEV_M2$Electric_Range - pred_test_sample_PHEV_M2)^2)

RMSE <- sqrt(MSE)
Rsquared <- 1 - MSE/var(test_data_PHEV_M2$Electric_Range)

MAE <- mean(abs(test_data_PHEV_M2$Electric_Range - pred_test_sample_PHEV_M2))

c(MSE,RMSE,MAE,Rsquared)
````
````

````
```


################################################################################
# BEV Predictor Model_Year, Model,Make
```{r}
Analy_data_BEV_MMM<-Bev_data1 %>%
  select(Model_Year,Make,Model,Electric_Range)
glimpse(Analy_data_BEV_MMM)
```

#Split dataset
```{r}
## Split the Data into a 80/20 Training/Testing Set ##
set.seed(123)
p_split_BEV_MMM <- rsample::initial_split(Analy_data_BEV_MMM,prop=0.80)
train_data_BEV_MMM <- rsample::training(p_split_BEV_MMM)
test_data_BEV_MMM <- rsample::testing(p_split_BEV_MMM)

# Check missing values by variable in train_data and test_data
colSums(is.na(train_data_BEV_MMM))

colSums(is.na(test_data_BEV_MMM))
```


Random Forest
```{r}
# Random Forest Model on the training sample
#rang = ranger(Electric_Range~ ., data = train_data)
#rang

## Identify hyper-parameters
# 1.mtry (Randomly selected predictor)
# 2.splitrule
# 3.min.node size
# Specify values for hyper-parameters # mtry --> generally start with around
square-root of variables (3 parameters)
sqrt(3) #1.7

my.mtry = c(1,2,3)
#splitrule
my.rule = "variance"
# min.node.size
my.nodes = c(5,7,9,11)

# create tuning grid
my.grid = expand.grid(mtry = my.mtry,
````

````
                      splitrule = my.rule,
                      min.node.size = my.nodes)
my.grid

#Select an appropriate evaluation metric
my.metric = "RMSE"

## 5. Train model over hyperparameters
#?caret::train
set.seed(2)
rf.tune_BEV_MMM = caret::train(Electric_Range~ ., data = train_data_BEV_MMM,
                       method = "ranger",
                       metric = my.metric ,
                       importance = "impurity",
                       trControl = trainControl(method = "cv", number = 25) ,
                       tuneGrid = my.grid ) # grid of hyperparameters
rf.tune_BEV_MMM

rf.tune_BEV_MMM$results %>% arrange(RMSE)

## Select best hyperparameters based on evaluation metric
rf.tune_BEV_MMM$bestTune

```

```{r}
# Random forest model variable importance scores were based on improvement in gini
impurity # Extract scaled variable importance scores
rf_importance_BEV_MMM <- varImp(rf.tune_BEV_MMM)
print(rf_importance_BEV_MMM)
vip::vip(rf.tune_BEV_MMM)
```

```{r}
# Create a data frame for Random Forest variable importance
rf_importance_BEV_MMM_df <- data.frame( Variable =
rownames(rf_importance_BEV_MMM$importance), Importance =
rf_importance_BEV_MMM$importance$Overall)

# Arrange in order
rf_importance_BEV_MMM_df<-rf_importance_BEV_MMM_df %>%
  arrange(desc(Importance))
```

```{r}
p <- rf_importance_BEV_MMM_df %>%
  filter(Importance > 1.2) %>%  # Filter rows where Importance is greater than 1
  ggplot(aes(x = reorder(Variable, desc(Importance)), y = Importance)) +
  geom_bar(stat = "identity", fill = "yellow") +
  geom_text(aes(label = ifelse(Importance > 0, sprintf("%.1f", Importance), "")),
````

````
size = 2.5, color = 'black', vjust = 1.0) +
  labs(title = "Random Forest Variable Importance (Variance Impurity) for scaled ",
x = "Variable", y = "Importance") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5, size = 10),
        axis.text.x = element_text(angle = 45, hjust = 1))

print(p)
```




```{r}
## Use tuned model to predict test sample
pred_test_sample_BEV_MMM = predict(rf.tune_BEV_MMM, test_data_BEV_MMM)
head(pred_test_sample_BEV_MMM)
```



```{r}
## Graphically evaluate the predicted & observed values ##

ggplot() +
geom_point(aes(pred_test_sample_BEV_MMM,test_data_BEV_MMM$Electric_Range)) +
  labs(x = "Predicted Values",
       y = "Observed Values") + theme_classic()
```

```{r}
## MSE,RMSE, Rsquared,MAE ##

MSE <- mean((test_data_BEV_MMM$Electric_Range - pred_test_sample_BEV_MMM)^2)

RMSE <- sqrt(MSE)
Rsquared <- 1 - MSE/var(test_data_BEV_MMM$Electric_Range)

MAE <- mean(abs(test_data_BEV_MMM$Electric_Range - pred_test_sample_BEV_MMM))

c(MSE,RMSE,MAE,Rsquared)


```

# PHEV Predictor Model_Year, Model,Make
```{r}
Analy_data_PHEV_MMM<-PHEV_data2%>%
  select(Model_Year,Make,Model,Electric_Range)
glimpse(Analy_data_PHEV_MMM)
```
````
````

````
#Split dataset
```{r}
## Split the Data into a 80/20 Training/Testing Set ##
set.seed(123)
p_split_PHEV_MMM <- rsample::initial_split(Analy_data_PHEV_MMM,prop=0.80)
train_data_PHEV_MMM <- rsample::training(p_split_PHEV_MMM)
test_data_PHEV_MMM <- rsample::testing(p_split_PHEV_MMM)

# Check missing values by variable in train_data and test_data
colSums(is.na(train_data_PHEV_MMM))

colSums(is.na(test_data_PHEV_MMM))
```


Random Forest
```{r}
# Random Forest Model on the training sample
#rang = ranger(Electric_Range~ ., data = train_data)
#rang

## Identify hyper-parameters
# 1.mtry (Randomly selected predictor)
# 2.splitrule
# 3.min.node size
# Specify values for hyper-parameters # mtry --> generally start with around
square-root of variables (8 parameters)
sqrt(3) #1.7

my.mtry = c(1,2,3)
#splitrule
my.rule = "variance"
# min.node.size
my.nodes = c(5,7,9,11)

# create tuning grid
my.grid = expand.grid(mtry = my.mtry,
                      splitrule = my.rule,
                      min.node.size = my.nodes)
my.grid

#Select an appropriate evaluation metric
my.metric = "RMSE"

## 5. Train model over hyperparameters
#?caret::train
set.seed(2)
rf.tune_PHEV_MMM = caret::train(Electric_Range~ ., data = train_data_PHEV_MMM,
                       method = "ranger",
````
````

````
                       metric = my.metric ,
                       importance = "impurity",
                       trControl = trainControl(method = "cv", number = 10) ,
                       tuneGrid = my.grid) # grid of hyperparameters
rf.tune_PHEV_MMM

rf.tune_PHEV_MMM$results %>% arrange(RMSE)

Select best hyperparameters based on evaluation metric
rf.tune_PHEV_MMM$bestTune

```


```{r}
# Random forest model variable importance scores were based on improvement in gini
impurity # Extract scaled variable importance scores
rf_importance_PHEV_MMM <- varImp(rf.tune_PHEV_MMM)
print(rf_importance_PHEV_MMM)
vip::vip(rf.tune_PHEV_MMM)
```



```{r}
## Use tuned model to predict test sample
pred_test_sample_PHEV_MMM = predict(rf.tune_PHEV_MMM, test_data_PHEV_MMM)
head(pred_test_sample_PHEV_MMM)
```



```{r}
## Graphically evaluate the predicted & observed values ##

ggplot() +
geom_point(aes(pred_test_sample_PHEV_MMM,test_data_PHEV_MMM$Electric_Range)) +
  labs(x = "Predicted Values",
       y = "Observed Values") +
  theme_classic()
```


```{r}
## MSE,RMSE, Rsquared,MAE ##

MSE <- mean((test_data_PHEV_MMM$Electric_Range - pred_test_sample_PHEV_MMM)^2)

RMSE <- sqrt(MSE)
Rsquared <- 1 - MSE/var(test_data_PHEV_MMM$Electric_Range)

MAE <- mean(abs(test_data_PHEV_MMM$Electric_Range - pred_test_sample_PHEV_MMM))

c(MSE,RMSE,MAE,Rsquared)
````

```
```

````
---
title: "Project"
author: "Prativa Basnet and Brandi"
date: "October 20, 2023"
output:
  word_document: default
  html_document: default
  pdf_document: default
editor_options:
  chunk_output_type: console
---

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
```


```{r setup, include=FALSE}
# Install packages tinytex to knit the code with result

options(repos = c(CRAN = "https://cran.rstudio.com/"))
library(tinytex)
```

```{r}
# Packages
library(tidyverse)
library(schrute)
library(patchwork)
library(gganimate)
library(car)
library(rpart)
library(rpart.plot)
library(olsrr)
```

```{r}
# Read the dataset
electric_veh<- readr::read_csv("Electric_Vehicle_Population_Data_Project.csv")
dplyr::glimpse(electric_veh)
summary(electric_veh)

# Filter vehicles registered in Washington State
electric_veh_washington <- electric_veh %>%
  filter(State == "WA")

glimpse(electric_veh_washington)
summary(electric_veh_washington)
```
````

````
```{r}
## Select the variables of interest using "select" function
electric_veh_washington1 <- electric_veh_washington %>%

select(Model_Year,Make,Model,Electric_Vehicle_Type,Electric_Range)#select(County,Cit
y,Postal_Code,Model_Year,Make,Model,Electric_Vehicle_Type,Vehicle_Location,Electric_
Range)

# Check for missing values
#which(is.na(electric_veh_washington1))

# Check missing values by variable in electric_veh_washington1 dataset
colSums(is.na(electric_veh_washington1))

# Variable "Vehicle_Location" has 3 missing observation so it is removed
#electric_veh_washington2 <- electric_veh_washington1 %>%
  #filter(!is.na(Vehicle_Location))

# Recheck missing values
#colSums(is.na(electric_veh_washington2))

#check the number of observations
num_observations <- nrow(electric_veh_washington1)
num_observations

```


```{r}
# Convert categorical variables to factors using "lapply" function
categorical_vars <- c("Make","Model","Electric_Vehicle_Type")
electric_veh_washington1[categorical_vars] <-
lapply(electric_veh_washington1[categorical_vars], as.factor)

glimpse(electric_veh_washington1)
summary(electric_veh_washington1)

# Check number of levels for each variable in data frame
sapply(electric_veh_washington1, function(x) length(unique(x)))
```


```{r}
# Data Visualization for electric vehicle type
electric_veh_count <- electric_veh_washington1 %>%
  group_by(Electric_Vehicle_Type) %>%
  count()
````
````

````
# Calculate percentages
electric_veh_count <- electric_veh_count %>%
  ungroup() %>%
  mutate(percentage = (n / sum(n)) * 100)

# Create a bar chart for Electric Vehicle Types
p1<-electric_veh_count %>%
  ggplot(aes(x = Electric_Vehicle_Type , y = percentage,
fill=Electric_Vehicle_Type)) +
  geom_bar(stat = 'identity') +
  geom_text(aes(label= sprintf("%.2f%%", percentage)), size =
3,color='black',vjust=1.5) +
  labs(title = "Distribution of Electric Vehicle Types ",
       x = "Electric Vehicle Type",
       y = "Percentage") +
  theme_minimal()+
  theme(plot.title = element_text(hjust = 0.5,size=12),
        axis.text.x = element_text(angle = 50, hjust = 1),
        axis.text.y = element_text(hjust = 1))

p1
```



```{r}
# Filter the data for Battery Electric Vehicles (BEV)
Bev_data <- electric_veh_washington1 %>%
  filter(Electric_Vehicle_Type == "Battery Electric Vehicle (BEV)")

# Recheck missing values
colSums(is.na(Bev_data))

#check the number of observations
num_observations <- nrow(Bev_data)
num_observations
```

```{r}
# Filter out rows with electric_range not equal to 0
Bev_data1 <- Bev_data %>%
  filter(Electric_Range!=0)
```



```{r}
# Count the number of vehicles in each year
vehicle_counts <- Bev_data1 %>%
  group_by(Model_Year) %>%
  summarize(Count = n())
````

````
# View the vehicle counts
print(vehicle_counts)
```


# Total number of Battery Electric Vehicle (BEV) for Model_Year
```{r}
# Create a horizontal bar chart BEV vs Year
p2 <- vehicle_counts %>%
  ggplot( aes(x=Count,y=reorder(Model_Year,Count)))+
  geom_bar(stat = 'identity', fill="orange") +
  #geom_text(aes(label= Count), size = 2,color='black',hjust=1.0) +
  labs(title = "Total Number of Battery Electric Vehicle (BEV) by Model Year",
       x = "Count",
       y = "Model Year") +
  scale_x_continuous(breaks = seq(0, 26000, by = 2000)) +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5,size=20),
  axis.title = element_text(size = 15),
  axis.text.x = element_text(angle = 45, hjust = 1,size=10),
  axis.text.y = element_text(size=10))

p2
```

```{r}
#Model_Year between 1997 to 2007 have only few vehicles less than 9
# Select Model_Year from 2008 to 2024
Bev_data2 <- Bev_data1 %>%
  filter(Model_Year >= 2008)
head(Bev_data2)
```

# Max electric range
```{r}
# Group the data by Model_Year, Make, and Model, and find the maximum electric range
max_range_Bev <- Bev_data2 %>%
  group_by(Model_Year, Make, Model) %>%
  summarize(Max_Electric_Range_Bev = max(Electric_Range))


#check the number of observations
num_observations <- nrow(max_range_Bev)
num_observations

#Print the first 10 observation
head(max_range_Bev, n = 10)
```
````
````

````
Scatterplot of Model_Year and Make
```{r}
# Create a scatter plot of Model_Year and Make for BEV
p3 <- max_range_Bev %>%
ggplot(aes(x = Model_Year, y = Max_Electric_Range_Bev, color = Make)) +
  geom_point(size=3) +
  labs(title = "Maximum Electric Range of BEV by ModelYear and Make",x = "Model
Year",y = "Max Electric Range") +
  scale_x_continuous(breaks = seq(2008, 2024, by = 1)) +
  scale_y_continuous(breaks = seq(0, 400, by = 50)) +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5,size=20),
  axis.title = element_text(size = 15),
  legend.text = element_text(size = 8),
  axis.text.x = element_text( angle = 45,hjust = 1,size=12),
  axis.text.y = element_text(size=12),
  legend.position = 'bottom')+
  guides(color = guide_legend(ncol = 11))
p3
```



```{r}
#Pick the Model Year greater than or equal to 2015 for boxplot
Boxplot_data <- max_range_Bev %>%
  filter(Model_Year >= 2015)
```


Boxplot of Model_Year and Make
```{r}
# Create a boxplot
p4 <- Boxplot_data %>%
  ggplot(aes(x = Model_Year, y = Max_Electric_Range_Bev, color = Make)) +
  geom_boxplot(width = 0.00001) +
  labs(title = "Boxplot of Maximum Electric Range of BEV by Model Year(2015-2021)
and Make",
       x = "Model Year",
       y = "Maximum Electric Range") +
 scale_x_continuous(breaks = seq(2015, 2021, by = 1)) +
 scale_y_continuous(breaks = seq(0, 400, by = 50)) +
 theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5,size=20),
  axis.title = element_text(size = 15),
  legend.text = element_text(size = 8),
  axis.text.x = element_text( hjust = 1,size=12),
  axis.text.y = element_text(size=12),
  legend.position = 'bottom')+
  guides(color = guide_legend(ncol = 10))

p4
````
````

````
# Save the plot to a file
ggsave("my_plot.png", p4, width = 8, height = 6)
```

# Scatterplot of Model_Year and Model
```{r}
# Create a scatter plot of Model_Year and Model for BEV
p5 <- max_range_Bev %>%
ggplot(aes(x = Model_Year, y = Max_Electric_Range_Bev, color = Model)) +
  geom_point(size=3) +
  labs(title = "Maximum Electric Range of BEV by ModelYear and Model",
       x = "Model Year",y = "Max Electric Range") +
  scale_x_continuous(breaks = seq(1997, 2021, by = 1)) +
  scale_y_continuous(breaks = seq(0, 400, by = 50)) +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5,size=20),
  axis.title = element_text(size = 15),
  legend.text = element_text(size = 7),
  axis.text.x = element_text( angle = 45,hjust = 1,size=12),
  axis.text.y = element_text(size=12),
  legend.position = 'bottom')+
  guides(color = guide_legend(ncol = 11))

p5
```


Boxplot of Model_Year and Model
```{r}
# Create a boxplot Model_Year and Model for BEV
p6 <- Boxplot_data %>%
  ggplot(aes(x = Model_Year, y = Max_Electric_Range_Bev, color = Model)) +
  geom_boxplot(width = 0.001) +
  labs(title = "Maximum Electric Range of BEV by Model Year(2015-2021) and Model",
       x = "Model Year",
       y = "Maximum Electric Range") +
 scale_x_continuous(breaks = seq(2015, 2021, by = 1)) +
 scale_y_continuous(breaks = seq(0, 400, by = 50)) +
 theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5,size=20),
  axis.title = element_text(size = 15),
  legend.text = element_text(size = 8),
  axis.text.x = element_text( hjust = 1,size=12),
  axis.text.y = element_text(size=12),
  legend.position = 'bottom')+
  guides(color = guide_legend(ncol = 10))

p6

```
````
````

````
# Average electric range
```{r}
# Calculate average maximum electric range for each Model Year and Make for BEV
avg_range_Bev_MYrs <- max_range_Bev %>%
  group_by(Model_Year,Make) %>%
  summarize(Avg_Electric_Range = mean(Max_Electric_Range_Bev))
```


# Need to find a way to show average number on each bar for different make
```{r}
# Create a bar chart
p7 <- avg_range_Bev_MYrs %>%
  ggplot( aes(x=Model_Year,y=Avg_Electric_Range,fill=Make))+
  geom_bar(stat = 'identity') +
  geom_text(aes(label=sprintf("%.1f", Avg_Electric_Range)), size =
2,color='Black',vjust=1.5) +
  labs(title = "Average Maximum Electric Range of BEV by Model Year",
       x = "Model Year",
       y = "Average Max Electric Range") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5,size=10),
  axis.title = element_text(size = 7),
  legend.text = element_text(size = 6),
  axis.text.x = element_text(angle = 45, hjust = 1,size=7),
  axis.text.y = element_text(size=7),
  legend.position = 'bottom')+
  guides(fill = guide_legend(ncol = 10))

p7
```


```{r}
# Calculate average maximum electric range for each Make for BEV
avg_range_Bev_make <- max_range_Bev %>%
  group_by(Make) %>%
  summarize(Avg_Max_Electric_Range = mean(Max_Electric_Range_Bev))
head(avg_range_Bev_make)
```

```{r}
# Create a bar chart
p8 <- avg_range_Bev_make %>%
  ggplot(aes(x = reorder(Make,Avg_Max_Electric_Range), y = Avg_Max_Electric_Range,
fill=Make)) +
  geom_bar(stat = 'identity') +
  geom_text(aes(label=sprintf("%.1f", Avg_Max_Electric_Range)), size =
3,color='Black',vjust=1.5) +
````
````

```
  labs(title = "Average Maximum Electric Range of BEV by Make",
       x = "Make",
       y = "Average Maximum Electric Range") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5,size=20),
  axis.title = element_text(size = 15),
  legend.text = element_text(size = 8),
  axis.text.x = element_text(angle = 45, hjust = 1,size=10),
  axis.text.y = element_text(size=12),
  legend.position = 'bottom')+
  guides(fill = guide_legend(ncol = 10))

p8
```

```{r}
# Calculate average maximum electric range for each Model for BEV
avg_range_Bev_model <- max_range_Bev %>%
  group_by(Model) %>%
  summarize(Avg_Max_Electric_Range_Mod = mean(Max_Electric_Range_Bev))
head(avg_range_Bev_model)
```

```{r}
# Create a bar chart
p9 <- avg_range_Bev_model %>%
  ggplot(aes(x =reorder(Model,Avg_Max_Electric_Range_Mod), y =
Avg_Max_Electric_Range_Mod, fill=Model)) +
  geom_bar(stat = 'identity') +
  geom_text(aes(label=sprintf("%.1f", Avg_Max_Electric_Range_Mod)), size =
3,color='Black',vjust=1.5) +
  labs(title = "Average Maximum Electric Range of BEV by Model",
       x = "Model",
       y = "Average Max Electric Range") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5,size=20),
  axis.title = element_text(size = 15),
  legend.text = element_text(size = 7),
  axis.text.x = element_text(angle = 45, hjust = 1,size=6),
  axis.text.y = element_text(size=12),
  legend.position = 'bottom')+
  guides(fill = guide_legend(ncol = 11))

p9
```

#Split dataset

```{r}
# select data
```

````
Aanaly_data<-Bev_data2 %>%
  select(Model_Year,Model,Make,Electric_Range)

Linear combinations
caret::findLinearCombos(dplyr::select(Aanaly_data, -c(Electric_Range))) #none

Split the Data into a 80/20 Training/Testing Set
set.seed(123)
p_split <- rsample::initial_split(Aanaly_data,prop=0.80)
train_data <- rsample::training(p_split)
test_data <- rsample::testing(p_split)

# Check missing values by variable in train_data and test_data
colSums(is.na(train_data))

colSums(is.na(test_data))


```


#Multiple Regression
```{r}
## Fit full Model on training dataframe ##
pmod<-lm(Electric_Range~., data=train_data)
pmod
# Summary of the regression model
summary(pmod)
## Stepwise Selection ##
stepwise_slect <- olsrr::ols_step_both_p(pmod)
plot(stepwise_slect)
```

Decision Tree method
```{r}
# Default tree
tree = rpart(Electric_Range~., data = train_data, method = "anova")
tree
# Visualizing the tree (best aspect of basic decision trees)
rpart.plot(tree)

### Tree PRUNING #####################################################
#CP --> complexity parameter !!!!!!!
#nsplit --> number of non-terminal nodes
#rel error --> training error
#xerror --> cross-validation error !!!!!!!!
#xst --> std.dev of the cv error

# Complexity Parameter (CP)
tree$cptable
````

````
#default CP
tree$control$cp #0.01 --> second lowest xerror

# Select best CP
lambda = tree$cptable[which.min(tree$cptable[,"xerror"]), "CP"]
lambda #0.01 --> lowest xerror

# Since the default decision tree has uesd the lowest cross-validation it is not
necessary to Prune tree model (based on best CP)
#tree.prune = rpart::prune(tree, lambda)
#tree.prune

#Predict the "target" variable in the test dataset "train_data"
# Predictions
pred = predict(tree , train_data, type = "anova") #"prob" for probability of being
in class

test_data <- test_data %>% select(-Electric_Range)  # Remove the outcome variable
predictions <- predict(model, newdata = test_data)

# Evaluate the model's performance (for regression)
rmse <- sqrt(mean((predictions - test_data$Electric_Range)^2))  # Root Mean Squared
Error

# Display the RMSE
cat("Root Mean Squared Error:", rmse, "\n")

# Visualize the decision tree (optional)
plot(model)
text(model)

```



```{r}
correlation_matrix <- cor(train_data[, c("Model_Year", "Make", "Model")])

## Fit full Model on training dataframe ##
pmod<-lm(Electric_Range~., data=train_data)

# Summary of the regression model
summary(pmod)
```




```{r}
# Assuming your data is stored in 'Bev_data2'
data <- Bev_data2
````

````
# Define the outcome variable and predictor variables
outcome_variable <- "Electric_Range"
predictor_variables <- c("Model_Year", "Make", "Model")

# Fit a linear regression model
model <- lm(formula = paste(outcome_variable, "~", paste(predictor_variables,
collapse = "+")), data = data)

# Summary of the regression model
summary(model)
?rpart
# Make predictions
predictions <- predict(model, newdata = data)

# Optionally, you can add the predictions back to the original data
data$Predicted_Range <- predictions

# View the model summary and results
print(summary(model))

```


#ANCOVA Model
```{r}
# select data
ancova_data<-Bev_data2 %>%
  select(Model_Year,Model,Make,Electric_Range)
# Fit an ANCOVA model for BEV
ancova_model_BEV <- lm(Electric_Range ~ ., data = ancova_data)


# View the summary of the ANCOVA model
summary(ancova_model_BEV)

# Check for aliased coefficients
alias_info <- alias(ancova_model_BEV)
print(alias_info)
# Calculate VIF for your regression model
vif_values <- vif(ancova_model_BEV)
```


stress %>%
  anova_test(
    score ~ age + treatment + exercise +
     treatment*exercise + age*treatment +
     age*exercise + age*exercise*treatment
  )

#Homogeneity of regression slopes (checks the interaction between independent
variables)
````
````

````
```{r}
# Fit your linear regression model
ancova_model_BEV <- lm(Electric_Range ~ Model_Year + Model + Make + Model_Year *
Model + Model_Year * Make + Model * Make + Model_Year * Model * Make, data =
ancova_data)

# Perform ANOVA test for the entire model
anova_result <- anova(ancova_model_BEV)


```


# Fit the model, the covariate goes first
model <- lm(score ~ age + treatment*exercise, data = stress)
# Inspect the model diagnostic metrics
model.metrics <- augment(model) %>%
  select(-.hat, -.sigma, -.fitted, -.se.fit) # Remove details
head(model.metrics, 3)

Plug-in Hybrid Electric Vehicle (PHEV)
```{r}
# Filter the data for Plug-in Hybrid Electric Vehicle (PHEV)
plug_data <- electric_veh_washington2 %>%
  filter(Electric_Vehicle_Type == "Plug-in Hybrid Electric Vehicle (PHEV)")
```

```{r}

# Count the number of vehicles in each year
plug_vehicle_counts <- plug_data %>%
  group_by(Model_Year) %>%
  summarize(Count = n())

# View the vehicle counts
print(plug_vehicle_counts)
```

```{r}
# Create a horizontal bar chart BEV vs Year
p10 <- plug_vehicle_counts %>%
  ggplot( aes(x=Count,y=reorder(Model_Year,Count)))+
  geom_bar(stat = 'identity', fill="orange") +
  #geom_text(aes(label= Count), size = 2,color='black',hjust=1.0) +
  labs(title = "Total number of  Plug-in Hybrid Electric Vehicle (PHEV) by Model
Year",
       x = "Count",
````
````

````
       y = "Model Year") +
  scale_x_continuous(breaks = seq(0, 5000, by = 1000)) +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5,size=20),
  axis.title = element_text(size = 15),
  axis.text.x = element_text(angle = 45, hjust = 1,size=10),
  axis.text.y = element_text(size=10))

p10
```

# Max electric range for Plug-in Hybrid Electric Vehicle (PHEV)
```{r}
# Group the data by Model_Year, Make, and Model, and find the maximum electric range
max_range_plug <- plug_data %>%
  group_by(Model_Year, Make, Model) %>%
  summarize(Max_Electric_Range_plug = max(Electric_Range))

# Filter out rows with max_range_Bev not equal to 0
max_range_plug1 <- max_range_plug %>%
  filter(Max_Electric_Range_plug  > 0)

#check the number of observations
num_observations <- nrow(max_range_plug1)
num_observations

# Print top 10
head(max_range_plug1, n = 10)
```


Scatterplot of Model_Year and Make
```{r}
# Create a scatterplot of Model_Year and Make for PHEV
p11<- max_range_plug1 %>%
  ggplot(aes(x = Model_Year, y = Max_Electric_Range_plug, color = Make)) +
  geom_point(size=3) +
  labs(title = "Maximum Electric Range of PHEV by Model Year and Make",
       x = "Model Year", y = "Max Electric Range") +
  scale_x_continuous(breaks = seq(2010, 2024, by = 1)) +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5,size=20),
  axis.title = element_text(size = 15),
  legend.text = element_text(size = 8),
  axis.text.x = element_text( angle = 45,hjust = 1,size=12),
  axis.text.y = element_text(size=12),
  legend.position = 'bottom')+
  guides(color = guide_legend(ncol = 11))

p11
````

````
```

# Boxplot of Model_Year and Make
```{r}
# Create a boxplot
p12 <- max_range_plug1 %>%
  ggplot(aes(x = Model_Year, y = Max_Electric_Range_plug, color = Make)) +
  geom_boxplot(width = 0.001) +
  labs(title = "Boxplot of Maximum Electric Range of PHEV by Model Year(2010-2024)
and Make",
       x = "Model Year",
       y = "Maximum Electric Range") +
 scale_x_continuous(breaks = seq(2010, 2024, by = 1)) +
 theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5,size=20),
  axis.title = element_text(size = 15),
  legend.text = element_text(size = 8),
  axis.text.x = element_text( hjust = 1,size=12),
  axis.text.y = element_text(size=12),
  legend.position = 'bottom')+
  guides(color = guide_legend(ncol = 10))

p12
```

# Scatterplot of Model_Year and Model
```{r}
# Create a scatterplot of Model_Year and Model for PHEV
p13<- max_range_plug1 %>%
  ggplot(aes(x = Model_Year, y = Max_Electric_Range_plug, color = Model)) +
  geom_point(size=3) +
  labs(title = "Maximum Electric Range of PHEV by Model Year and Model",
       x = "Model Year", y = "Max Electric Range") +
  scale_x_continuous(breaks = seq(2010, 2024, by = 1)) +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5,size=20),
  axis.title = element_text(size = 15),
  legend.text = element_text(size = 7),
  axis.text.x = element_text( angle = 45,hjust = 1,size=12),
  axis.text.y = element_text(size=12),
  legend.position = 'bottom')+
  guides(color = guide_legend(ncol = 12))

p13
```


# Boxplot of Model_Year and Model
```{r}
# Create a boxplot Model_Year and Model for PHEV
````

````
p14 <- max_range_plug1 %>%
  ggplot(aes(x = Model_Year, y = Max_Electric_Range_plug, color = Model)) +
  geom_boxplot(width = 0.001) +
  labs(title = "Maximum Electric Range of PHEV by Model Year(2010-2024) and Model",
       x = "Model Year",
       y = "Maximum Electric Range") +
 scale_x_continuous(breaks = seq(2010, 2024, by = 1)) +
 theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5,size=20),
  axis.title = element_text(size = 15),
  legend.text = element_text(size = 8),
  axis.text.x = element_text( hjust = 1,size=12),
  axis.text.y = element_text(size=12),
  legend.position = 'bottom')+
  guides(color = guide_legend(ncol = 12))
p14
```


# Average electric range
```{r}
# Calculate average maximum electric range for each Model Year and Make for BEV
avg_range_Phev_MYrs <- max_range_plug1 %>%
  group_by(Model_Year,Make) %>%
  summarize(Avg_Electric_Range = mean(Max_Electric_Range_plug))
```


# Need to find a way to show average number on each bar for different make
```{r}
# Create a bar chart
p15 <- avg_range_Phev_MYrs %>%
  ggplot( aes(x=Model_Year,y=Avg_Electric_Range,fill=Make))+
  geom_bar(stat = 'identity') +
  geom_text(aes(label=sprintf("%.1f", Avg_Electric_Range)), size =
2,color='Black',vjust=1.5) +
  labs(title = "Average Maximum Electric Range of PHEV by Model Year",
       x = "Model Year",
       y = "Average Max Electric Range") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5,size=20),
  axis.title = element_text(size = 15),
  legend.text = element_text(size = 6),
  axis.text.x = element_text(angle = 45, hjust = 1,size=10),
  axis.text.y = element_text(size=10),
  legend.position = 'bottom')+
  guides(fill = guide_legend(ncol = 14))

p15
```
````
````

````
Barchart of Make and Average max electric range
```{r}
# Calculate average maximum electric range for each Make
avg_range_plug_make <- max_range_plug1 %>%
  group_by(Make) %>%
  summarize(Avg_Max_Electric_Range = mean(Max_Electric_Range_plug))
head(avg_range_plug_make)
```

```{r}
# Create a bar chart
p16 <- avg_range_plug_make %>%
  ggplot(aes(x =reorder(Make,Avg_Max_Electric_Range),y =
Avg_Max_Electric_Range,fill=Make)) +
  geom_bar(stat = 'identity') +
  geom_text(aes(label=sprintf("%.1f", Avg_Max_Electric_Range)), size =
3,color='Black',vjust=1.5) +
  labs(title = "Average Maximum Electric Range of PHEV by Make",
       x = "Make",
       y = "Average Maximum Electric Range") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5,size=20),
  axis.title = element_text(size = 15),
  legend.text = element_text(size = 8),
  axis.text.x = element_text(angle = 45, hjust = 1,size=10),
  axis.text.y = element_text(size=12),
  legend.position = 'bottom')+
  guides(fill = guide_legend(ncol = 10))

p16
```


Barchart of Model and Average max electric range
```{r}
# Calculate average maximum electric range for each Model for PHEV
avg_range_plug_model <- max_range_plug1 %>%
  group_by(Model) %>%
  summarize(Avg_Max_Electric_Range = mean(Max_Electric_Range_plug))
head(avg_range_plug_model)
```


```{r}
# Create a bar chart
p17 <- avg_range_plug_model  %>%
  ggplot(aes(x = reorder(Model,Avg_Max_Electric_Range),y = Avg_Max_Electric_Range,
fill=Model)) +
  geom_bar(stat = 'identity') +
````
````

````
  geom_text(aes(label=sprintf("%.1f", Avg_Max_Electric_Range)), size =
1.75,color='Black',vjust=1.5) +
  labs(title = "Average Maximum Electric Range of PHEV by Model",
       x = "Model",
       y = "Average Maximum Electric Range") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5,size=20),
  axis.title = element_text(size = 15),
  legend.text = element_text(size = 7),
  axis.text.x = element_text(angle = 45, hjust = 1,size=8),
  axis.text.y = element_text(size=12),
  legend.position = 'bottom')+
  guides(fill = guide_legend(ncol = 12))

p17
```

```{r}
# select data
ancova_data<-Bev_data %>%
  select(Model_Year,Model,Make,Electric_Range)
# Fit an ANCOVA model for BEV
ancova_model_BEV <- lm(Electric_Range ~ ., data = ancova_data)

# View the summary of the ANCOVA model
summary(ancova_model_BEV)
```


```{r}
# Fit an ANCOVA model
ancova_model <- lm(Electric_Range ~ ., data = electric_veh_washington2)

# View the summary of the ANCOVA model
summary(ancova_model)
```


##########################Need to see these#############################
```{r}
boxplot(electric_veh_washington1$Electric_Range ~ electric_veh_washington1$Make,
        xlab = "Make", ylab = "Electric Range", main = "Box Plot by Make")
```


```{r}
# Extract latitude and longitude from the text in 'Vehicle_Location' column
electric_veh_washington2 <- electric_veh_washington2 %>%
  mutate(
    Longitude = as.numeric(sub(".*\\((.*)\\s.*", "\\1", Vehicle_Location)),
````

````
    Latitude = as.numeric(sub(".*\\s(.*)\\)", "\\1", Vehicle_Location)))


# Create spatial points using the longitude and latitude
electric_veh_washington2 <- st_as_sf(electric_veh_washington2, coords =
c("Longitude", "Latitude"), crs = 4326)

# Create a plot using geom_sf
ggplot(electric_veh_washington2) +
  geom_sf() +
  labs(title = "Spatial Plot of Electric Vehicles",
       x = "Longitude",
       y = "Latitude") +
  theme_minimal()

# Assuming you have latitude and longitude columns in your data frame
# Convert them to a suitable spatial object
electric_veh_washington2 <- electric_veh_washington2 %>%
  st_as_sf(coords = c("Longitude", "Latitude"), crs = 4326)

# Create a plot using geom_sf
ggplot(electric_veh_washington2, aes(x = Longitude, y = Latitude, color =
Electric_Vehicle_Type)) +
  geom_sf() +
  labs(title = "Spatial Plot of Electric Vehicles",
       x = "Longitude",
       y = "Latitude") +
  theme_minimal()
###############################delete#################
# Remove the temporary 'Longitude' and 'Latitude' columns
electric_veh_washington2 <- electric_veh_washington2 %>%
  select(-Longitude, -Latitude)
############################delete########################
```


```{r}
# Convert the Vehicle_Location column to a spatial object
electric_veh_washington1 <- electric_veh_washington1 %>%
  mutate(Vehicle_Location = st_as_text(st_point(x =
st_coordinates(st_sfc(st_geometry(electric_veh_washington1$Vehicle_Location))))))

# Create an sf data frame
electric_sf <- st_as_sf(electric_veh_washington1, coords = c("Vehicle_Location"),
crs = 4326)

# Plot the sf data frame with ggplot2
ggplot(electric_sf) +
  geom_sf()
```
````
````

```{r}
# Assuming you have a data frame called 'electric_veh_data' with latitude and
longitude
# You need to create a spatial object using 'st_as_sf'
electric_sf <- st_as_sf(electric_veh_washington1, coords =
c("longitude_column_name", "latitude_column_name"), crs = 4326)

# Create a plot using 'geom_sf'
ggplot(electric_sf) +
  geom_sf(aes(color = City)) +
  labs(title = "Electric Vehicle Locations", subtitle = "City, County, Model, Make,
Model Year, and Postal Code") +
  theme_minimal()
```

```{r}
# Fit an ANCOVA model
ancova_model <- lm(Electric_Range ~ ., data = electric_veh_washington1)

# View the summary of the ANCOVA model
summary(ancova_model)
```

target ~ .,

```{r}
# Calculate the correlation matrix
correlation_matrix <- cor(electric_veh_washington1)
```

```{r}
# Calculate the correlation matrix
correlation_matrix <- cor(electric_veh_washington)

# Set a correlation threshold
correlation_threshold <- 0.7

# Find highly correlated variables
highly_correlated_vars <- findCorrelation(correlation_matrix, cutoff =
correlation_threshold)
```

```
# Subset the data frame to include only important variables
selected_data <- electric_veh_washington[, -highly_correlated_vars]
```

````
---
title: "Project"
author: "Prativa Basnet and Brandi"
date: "October 20, 2023"
output:
  word_document: default
  html_document: default
  pdf_document: default
editor_options:
  chunk_output_type: console

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
```


```{r setup, include=FALSE}
# Install packages tinytex to knit the code with result

options(repos = c(CRAN = "https://cran.rstudio.com/"))
library(tinytex)
```

```{r}
# Packages
library(tidyverse)
library(schrute)
library(patchwork)
library(gganimate)
library(car)
library(olsrr)
library(randomForest)
library(ranger)
```

```{r}
# Read the dataset
electric_veh<- readr::read_csv("Electric_Vehicle_Population_Data_Project.csv")
dplyr::glimpse(electric_veh)
summary(electric_veh)

# Filter vehicles registered in Washington State
electric_veh_washington <- electric_veh %>%
  filter(State == "WA")

glimpse(electric_veh_washington)
summary(electric_veh_washington)
```
````

````
```{r}
## Select the variables of interest using "select" function
electric_veh_washington1 <- electric_veh_washington %>%
  select(Model_Year,Make,Model,Electric_Vehicle_Type,Electric_Range)

# Check for missing values
#which(is.na(electric_veh_washington1))

# Check missing values by variable in electric_veh_washington1 dataset
colSums(is.na(electric_veh_washington1))


#check the number of observations
num_observations <- nrow(electric_veh_washington1)
num_observations

```


```{r}
# Convert categorical variables to factors using "lapply" function
categorical_vars <- c("Make","Model","Electric_Vehicle_Type")
electric_veh_washington1[categorical_vars] <-
lapply(electric_veh_washington1[categorical_vars], as.factor)

glimpse(electric_veh_washington1)
summary(electric_veh_washington1)

# Check number of levels for each variable in data frame
sapply(electric_veh_washington1, function(x) length(unique(x)))
```



```{r}
# Data Visualization for electric vehicle type
electric_veh_count <- electric_veh_washington1 %>%
  group_by(Electric_Vehicle_Type) %>%
  count()

# Calculate percentages
electric_veh_count <- electric_veh_count %>%
  ungroup() %>%
  mutate(percentage = (n / sum(n)) * 100)
```

```{r}
# Create a bar chart for Electric Vehicle Types
p1<-electric_veh_count %>%
````
````

````
  ggplot(aes(x = Electric_Vehicle_Type , y = percentage,
fill=Electric_Vehicle_Type)) +
  geom_bar(stat = 'identity') +
  geom_text(aes(label= sprintf("%.2f%%", percentage)), size =
3,color='black',vjust=1.5) +
  labs(title = "Distribution of Electric Vehicle Types ",
       x = "Electric Vehicle Type",
       y = "Percentage") +
  theme_minimal()+
  theme(plot.title = element_text(hjust = 0.5,size=12),
        axis.text.x = element_text(angle = 50, hjust = 1),
        axis.text.y = element_text(hjust = 1))

p1
```


```{r}
# Filter the data for Battery Electric Vehicles (BEV)
Bev_data <- electric_veh_washington1 %>%
  filter(Electric_Vehicle_Type == "Battery Electric Vehicle (BEV)")


#check the number of observations
num_observations <- nrow(Bev_data)
num_observations
```

```{r}
# Filter out rows with electric_range not equal to 0
Bev_data1 <- Bev_data %>%
  filter(Electric_Range!=0)
```


```{r}
# Count the number of vehicles in each year
vehicle_counts <- Bev_data1 %>%
  group_by(Model_Year) %>%
  summarize(Count = n())

# View the vehicle counts
print(vehicle_counts)
```


Total number of Battery Electric Vehicle (BEV) for different Model_Year
```{r}
# Create a horizontal bar chart BEV vs Year
p2 <- vehicle_counts %>%
````

````
  ggplot( aes(x=Count,y=reorder(Model_Year,Count)))+
  geom_bar(stat = 'identity', fill="orange") +
  #geom_text(aes(label= Count), size = 2,color='black',hjust=1.0) +
  labs(title = "Total Number of Battery Electric Vehicle (BEV) by Model Year",
       x = "Count",
       y = "Model Year") +
  scale_x_continuous(breaks = seq(0, 26000, by = 2000)) +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5,size=20),
  axis.title = element_text(size = 15),
  axis.text.x = element_text(angle = 45, hjust = 1,size=10),
  axis.text.y = element_text(size=10))

p2
```

```{r}
#Model_Year between 1997 to 2007 have only few vehicles less than 9
# Select Model_Year from 2008 to 2024
Bev_data2 <- Bev_data1 %>%
  filter(Model_Year >= 2008)
head(Bev_data2)
```

# Max electric range
```{r}
# Group the data by Model_Year, Make, and Model, and find the maximum electric range
max_range_Bev <- Bev_data2 %>%
  group_by(Model_Year, Make, Model) %>%
  summarize(Max_Electric_Range_Bev = max(Electric_Range))


#check the number of observations
num_observations <- nrow(max_range_Bev)
num_observations

#Print the first 10 observation
head(max_range_Bev, n = 10)
```

# Scatterplot of Model_Year and Make
```{r}
# Create a scatter plot of Model_Year and Make for BEV
p3 <- max_range_Bev %>%
ggplot(aes(x = Model_Year, y = Max_Electric_Range_Bev, color = Make)) +
  geom_point(size=3) +
  labs(title = "Maximum Electric Range of BEV by ModelYear and Make",x = "Model
Year",y = "Max Electric Range") +
  scale_x_continuous(breaks = seq(2008, 2024, by = 1)) +
  scale_y_continuous(breaks = seq(0, 400, by = 50)) +
````

````
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5,size=20),
  axis.title = element_text(size = 15),
  legend.text = element_text(size = 8),
  axis.text.x = element_text( angle = 45,hjust = 1,size=12),
  axis.text.y = element_text(size=12),
  legend.position = 'bottom')+
  guides(color = guide_legend(ncol = 11))
p3
```

```{r}

max_range_Make <- max_range_Bev %>%
  group_by(Make) %>%
  summarize(Max_Electric_Range= max(Max_Electric_Range_Bev))
```

# Barplot for max range
```{r}
# Create a Barplot of Make vs max electric range for BEV
p <- max_range_Make %>%
ggplot(aes(x = reorder(Make,Max_Electric_Range), y = Max_Electric_Range, fill =
Make)) +
  geom_bar(stat = 'identity') +
  geom_text(aes(label=sprintf("%.1f", Max_Electric_Range)), size =
2,color='Black',vjust=1.5) +
  labs(title = "Maximum Electric Range of BEV by Make",x = "Make",y = "Maximum
Electric Range") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5,size=20),
  axis.title = element_text(size = 15),
  legend.text = element_text(size = 8),
  axis.text.x = element_text( angle = 45,hjust = 1,size=12),
  axis.text.y = element_text(size=12),
  legend.position = 'bottom')+
  guides(fill = guide_legend(ncol = 11))
p
```

# Boxplot for max range of Make
```{r}
# Create a boxplot
p4 <- max_range_Make %>%
  ggplot(aes(x = Make, y = Max_Electric_Range, fill = Make)) +
  geom_boxplot() +
  labs(title = "Boxplot of Maximum Electric Range of BEV by Make",
       x = "Make",
       y = "Maximum Electric Range") +
 scale_y_continuous(breaks = seq(0, 400, by = 50)) +
````

````
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5,size=20),
  axis.title = element_text(size = 15),
  legend.text = element_text(size = 8),
  axis.text.x = element_text( angle = 45,hjust = 1,size=12),
  axis.text.y = element_text(size=12),
  legend.position = 'bottom')+
  guides(fill = guide_legend(ncol = 10))

p4
```


# Scatterplot of Model_Year and Model
```{r}
# Create a scatter plot of Model_Year and Model for BEV
p5 <- max_range_Bev %>%
ggplot(aes(x = Model_Year, y = Max_Electric_Range_Bev, color = Model)) +
  geom_point(size=3) +
  labs(title = "Maximum Electric Range of BEV by ModelYear and Model",
       x = "Model Year",y = "Max Electric Range") +
  scale_x_continuous(breaks = seq(1997, 2021, by = 1)) +
  scale_y_continuous(breaks = seq(0, 400, by = 50)) +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5,size=20),
  axis.title = element_text(size = 15),
  legend.text = element_text(size = 7),
  axis.text.x = element_text( angle = 45,hjust = 1,size=12),
  axis.text.y = element_text(size=12),
  legend.position = 'bottom')+
  guides(color = guide_legend(ncol = 11))

p5
```


```{r}
max_range_Bev_Model <- max_range_Bev %>%
  group_by(Model) %>%
  summarize(Max_Electric_Range= max(Max_Electric_Range_Bev))
```

# Boxplot for max range of Model
```{r}
# Create a boxplot
p6 <- max_range_Bev_Model %>%
  ggplot(aes(x = Model, y = Max_Electric_Range, fill = Model)) +
  geom_boxplot() +
  labs(title = "Boxplot of Maximum Electric Range of BEV by Model",
       x = "Model",
````

````
       y = "Maximum Electric Range") +
 scale_y_continuous(breaks = seq(0, 400, by = 50)) +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5,size=20),
  axis.title = element_text(size = 15),
  legend.text = element_text(size = 8),
  axis.text.x = element_text( angle = 45,hjust = 1,size=12),
  axis.text.y = element_text(size=12),
  legend.position = 'bottom')+
  guides(fill= guide_legend(ncol = 10))

p6
```

# Average maximum electric range for Make
```{r}
# Calculate average maximum electric range for each Make for BEV
avg_range_Bev_make <- max_range_Bev %>%
  group_by(Make) %>%
  summarize(Avg_Max_Electric_Range = mean(Max_Electric_Range_Bev))
head(avg_range_Bev_make)
```

```{r}
# Create a bar chart
p7 <- avg_range_Bev_make %>%
  ggplot(aes(x = reorder(Make,Avg_Max_Electric_Range), y = Avg_Max_Electric_Range,
fill=Make)) +
  geom_bar(stat = 'identity') +
  geom_text(aes(label=sprintf("%.1f", Avg_Max_Electric_Range)), size =
3,color='Black',vjust=1.5) +
  labs(title = "Average Maximum Electric Range of BEV by Make",
       x = "Make",
       y = "Average Maximum Electric Range") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5,size=20),
  axis.title = element_text(size = 15),
  legend.text = element_text(size = 8),
  axis.text.x = element_text(angle = 45, hjust = 1,size=10),
  axis.text.y = element_text(size=12),
  legend.position = 'bottom')+
  guides(fill = guide_legend(ncol = 10))

p7
```

# Average maximum electric range for Model
```{r}
# Calculate average maximum electric range for each Model for BEV
avg_range_Bev_model <- max_range_Bev %>%
````

````
  group_by(Model) %>%
  summarize(Avg_Max_Electric_Range_Mod = mean(Max_Electric_Range_Bev))
head(avg_range_Bev_model)
```

```{r}
# Create a bar chart
p8 <- avg_range_Bev_model %>%
  ggplot(aes(x =reorder(Model,Avg_Max_Electric_Range_Mod), y =
Avg_Max_Electric_Range_Mod, fill=Model)) +
  geom_bar(stat = 'identity') +
  geom_text(aes(label=sprintf("%.1f", Avg_Max_Electric_Range_Mod)), size =
3,color='Black',vjust=1.5) +
  labs(title = "Average Maximum Electric Range of BEV by Model",
       x = "Model",
       y = "Average Max Electric Range") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5,size=20),
  axis.title = element_text(size = 15),
  legend.text = element_text(size = 7),
  axis.text.x = element_text(angle = 45, hjust = 1,size=6),
  axis.text.y = element_text(size=12),
  legend.position = 'bottom')+
  guides(fill = guide_legend(ncol = 11))

p8
```



# Plug-in Hybrid Electric Vehicle (PHEV)
```{r}
# Filter the data for Plug-in Hybrid Electric Vehicle (PHEV)
plug_data <- electric_veh_washington1 %>%
  filter(Electric_Vehicle_Type == "Plug-in Hybrid Electric Vehicle (PHEV)")

#check the number of observations
num_observations <- nrow(plug_data)
num_observations
```

```{r}
# Filter out rows with electric_range not equal to 0
plug_data1 <- plug_data %>%
  filter(Electric_Range!=0)
```


```{r}

# Count the number of vehicles in each year
````

````
plug_vehicle_counts <- plug_data1 %>%
  group_by(Model_Year) %>%
  summarize(Count = n())

# View the vehicle counts
print(plug_vehicle_counts)
```

```{r}
# Create a horizontal bar chart PHEV vs Year
p9 <- plug_vehicle_counts %>%
  ggplot( aes(x=Count,y=reorder(Model_Year,Count)))+
  geom_bar(stat = 'identity', fill="orange") +
  #geom_text(aes(label= Count), size = 2,color='black',hjust=1.0) +
  labs(title = "Total Number of  Plug-in Hybrid Electric Vehicle (PHEV) by Model
Year",
       x = "Count",
       y = "Model Year") +
  scale_x_continuous(breaks = seq(0, 5000, by = 1000)) +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5,size=20),
  axis.title = element_text(size = 15),
  axis.text.x = element_text(angle = 45, hjust = 1,size=10),
  axis.text.y = element_text(size=10))

p9
```

# Max electric range for Plug-in Hybrid Electric Vehicle (PHEV)
```{r}
# Group the data by Model_Year, Make, and Model, and find the maximum electric range
max_range_plug <- plug_data1 %>%
  group_by(Model_Year, Make, Model) %>%
  summarize(Max_Electric_Range= max(Electric_Range))


#check the number of observations
num_observations <- nrow(max_range_plug)
num_observations

# Print top 10
head(max_range_plug , n = 10)
```


# Scatterplot of Model_Year and Make
```{r}
# Create a scatterplot of Model_Year and Make for PHEV
p10<- max_range_plug %>%
  ggplot(aes(x = Model_Year, y = Max_Electric_Range, color = Make)) +
````

````
  geom_point(size=3) +
  labs(title = "Maximum Electric Range of PHEV by Model Year and Make",
       x = "Model Year", y = "Maximum Electric Range") +
  scale_x_continuous(breaks = seq(2010, 2024, by = 1)) +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5,size=20),
  axis.title = element_text(size = 15),
  legend.text = element_text(size = 8),
  axis.text.x = element_text( angle = 45,hjust = 1,size=12),
  axis.text.y = element_text(size=12),
  legend.position = 'bottom')+
  guides(color = guide_legend(ncol = 11))

p10
```

```{r}

max_range_plug2 <- max_range_plug  %>%
  group_by(Make) %>%
  summarize(Max_Electric_Range= max(Max_Electric_Range))
```

# Barplot for max range of Make
```{r}
# Create a Barplot of Make vs max electric range for BEV
pbar <- max_range_plug2 %>%
ggplot(aes(x = reorder(Make,Max_Electric_Range), y = Max_Electric_Range, fill =
Make)) +
  geom_bar(stat = 'identity') +
  geom_text(aes(label=sprintf("%.1f", Max_Electric_Range)), size =
2,color='Black',vjust=1.5) +
  labs(title = "Maximum Electric Range of PHEV by Make",x = "Make",y = "Maximum
Electric Range") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5,size=20),
  axis.title = element_text(size = 15),
  legend.text = element_text(size = 8),
  axis.text.x = element_text( angle = 45,hjust = 1,size=12),
  axis.text.y = element_text(size=12),
  legend.position = 'bottom')+
  guides(fill = guide_legend(ncol = 11))
pbar
```

# Boxplot for max range of Make
```{r}
# Create a boxplot
p11 <- max_range_plug2 %>%
  ggplot(aes(x = Make, y = Max_Electric_Range, fill = Make)) +
````

~~~
  geom_boxplot() +
  labs(title = "Boxplot of Maximum Electric Range of PHEV by Make",
       x = "Make",
       y = "Maximum Electric Range") +
 scale_y_continuous(breaks = seq(0, 400, by = 50)) +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5,size=20),
  axis.title = element_text(size = 15),
  legend.text = element_text(size = 8),
  axis.text.x = element_text( angle = 45,hjust = 1,size=12),
  axis.text.y = element_text(size=12),
  legend.position = 'bottom')+
  guides(fill = guide_legend(ncol = 10))

p11
```



# Scatterplot of Model_Year and Model
```{r}
# Create a scatterplot of Model_Year and Model for PHEV
p12<- max_range_plug %>%
  ggplot(aes(x = Model_Year, y = Max_Electric_Range, color = Model)) +
  geom_point(size=3) +
  labs(title = "Maximum Electric Range of PHEV by Model Year and Model",
       x = "Model Year", y = "Max Electric Range") +
  scale_x_continuous(breaks = seq(2010, 2024, by = 1)) +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5,size=20),
  axis.title = element_text(size = 15),
  legend.text = element_text(size = 7),
  axis.text.x = element_text( angle = 45,hjust = 1,size=12),
  axis.text.y = element_text(size=12),
  legend.position = 'bottom')+
  guides(color = guide_legend(ncol = 12))

p12
```



```{r}

max_range_Phev_Model <- max_range_plug %>%
  group_by(Model) %>%
  summarize(Max_Electric_Range= max(Max_Electric_Range))
```

# Boxplot for max range of Model
```{r}
# Create a boxplot
~~~

````
p13 <- max_range_Phev_Model %>%
  ggplot(aes(x = Model, y = Max_Electric_Range, fill = Model)) +
  geom_boxplot() +
  labs(title = "Boxplot of Maximum Electric Range of PHEV by Model",
       x = "Model",
       y = "Maximum Electric Range") +
 scale_y_continuous(breaks = seq(0, 400, by = 50)) +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5,size=20),
  axis.title = element_text(size = 15),
  legend.text = element_text(size = 8),
  axis.text.x = element_text( angle = 45,hjust = 1,size=12),
  axis.text.y = element_text(size=12),
  legend.position = 'bottom')+
  guides(fill= guide_legend(ncol = 12))

p13
```



# Average electric range Of Make
```{r}
# Calculate average maximum electric range for each Make
avg_range_plug_make <- max_range_plug %>%
  group_by(Make) %>%
  summarize(Avg_Max_Electric_Range = mean(Max_Electric_Range))
head(avg_range_plug_make)
```

```{r}
# Create a bar chart
p14 <- avg_range_plug_make %>%
  ggplot(aes(x =reorder(Make,Avg_Max_Electric_Range),y =
Avg_Max_Electric_Range,fill=Make)) +
  geom_bar(stat = 'identity') +
  geom_text(aes(label=sprintf("%.1f", Avg_Max_Electric_Range)), size =
3,color='Black',vjust=1.5) +
  labs(title = "Average Maximum Electric Range of PHEV by Make",
       x = "Make",
       y = "Average Maximum Electric Range") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5,size=20),
  axis.title = element_text(size = 15),
  legend.text = element_text(size = 8),
  axis.text.x = element_text(angle = 45, hjust = 1,size=10),
  axis.text.y = element_text(size=12),
  legend.position = 'bottom')+
  guides(fill = guide_legend(ncol = 10))

p14
````

````
```


# Barchart of Model and Average max electric range
```{r}
# Calculate average maximum electric range for each Model for PHEV
avg_range_plug_model <- max_range_plug %>%
  group_by(Model) %>%
  summarize(Avg_Max_Electric_Range = mean(Max_Electric_Range))
head(avg_range_plug_model)
```


```{r}
# Create a bar chart
p17 <- avg_range_plug_model  %>%
  ggplot(aes(x = reorder(Model,Avg_Max_Electric_Range),y = Avg_Max_Electric_Range,
fill=Model)) +
  geom_bar(stat = 'identity') +
  geom_text(aes(label=sprintf("%.1f", Avg_Max_Electric_Range)), size =
1.75,color='Black',vjust=1.5) +
  labs(title = "Average Maximum Electric Range of PHEV by Model",
       x = "Model",
       y = "Average Maximum Electric Range") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5,size=20),
  axis.title = element_text(size = 15),
  legend.text = element_text(size = 7),
  axis.text.x = element_text(angle = 45, hjust = 1,size=8),
  axis.text.y = element_text(size=12),
  legend.position = 'bottom')+
  guides(fill = guide_legend(ncol = 12))

p17
```

# Analysis Part

#Split dataset
```{r}
# Since Model and Make variable have multicollinearity so use only use Model_Year
and Make as predictor variable
Analy_data<-Bev_data2 %>%
  select(Model_Year,Make,Electric_Range)
head(Analy_data)




## Split the Data into a 80/20 Training/Testing Set ##
````

~~~~
set.seed(123)
p_split <- rsample::initial_split(Analy_data,prop=0.80)
train_data <- rsample::training(p_split)
test_data <- rsample::testing(p_split)

# Check missing values by variable in train_data and test_data
colSums(is.na(train_data))

colSums(is.na(test_data))
```

#Multiple Regression
```{r}
## Fit full Model on training dataframe ##
pmod<-lm(Electric_Range~., data=train_data)
# Summary of the regression model
summary(pmod)

```




```{r}
## Check Assumptions ##

## Normality ##

pmod$residuals |>
  ggpubr::ggqqplot()

pmod$residuals |>
  rstatix::shapiro_test()

## Constant Variance ##

ggplot() + geom_point(aes(fitted(pmod),rstudent(pmod))) + theme_classic()

#or
#Residual plots
ggplot() +
  geom_point(aes(x=fitted(pmod),y=rstudent(pmod))) +
  geom_hline(yintercept=3,color='red') +
  geom_hline(yintercept=-3,color='red') +
  geom_hline(yintercept=0,color='blue') +
  labs(y="Studentized Residuals",
       x="Fitted Values") +
  theme_classic()

#studentized Breusch-Pagan test
lmtest::bptest(pmod)
~~~~

````
## VIF ##

car::vif(pmod)

```




```{r}
# Model check
pmod |>
  moderndive::get_regression_summaries()

pmod  |>
  moderndive::get_regression_table()


```


# Random Forest
```{r}
# Split the data into training and testing sets
#set.seed(123)  # For reproducibility
#train_index <- sample(1:nrow(data), 0.7 * nrow(data))
#train_data <- data[train_index, ]
#test_data <- data[-train_index, ]

# Train a Random Forest model
model <- randomForest(
  formula = Electric_Range ~ .,
  data = train_data,
  ntree = 100,  # Number of trees in the forest (you can adjust this)
  mtry = sqrt(ncol(train_data) - 1)  # Number of variables randomly sampled at each
split (suggested value)
)

#Model
set.seed(12) #note ... this only works if not using parallel processing
rang = ranger(Electric_Range ~ ., data = train_data)
rang

## Train ranger model using caret
set.seed(12) #note ... this only works if not using parallel processing
rf = caret::train(Electric_Range ~ ., data = train_data, method = "ranger")
rf
rf$bestTune
````

````
## Confusion Matrix
pred = predict(rf, sonar.test, type = "raw") #ranger version of 'class'
conf = table(actual = sonar.test$Class, pred)
conf
# TP FN
# FP TN

# Accuracy (test sample)
acc = sum(diag(conf))/sum(conf)
acc

# Confusion Matrix ... the easy way (using caret)
?caret::confusionMatrix
conf2 = caret::confusionMatrix(data = pred, reference = sonar.test$Class)
conf2
# Nice ... it gives you a lot of info
# But NOOO ... the confusion matrix is flipped ... always pay attention
t(conf2$table) # just take the transpose

# Make predictions on the test data
test_data <- test_data %>% select(-Electric_Range)  # Remove the outcome variable
predictions <- predict(model, newdata = test_data)

# Evaluate the model's performance (for regression)
rmse <- sqrt(mean((predictions - test_data$Electric_Range)^2))  # Root Mean Squared
Error

# Display the RMSE
rmse

```
````
````