Final Project

Kalyan B

2024-08-06

```
install.packages(c("dplyr", "ggplot2", "tidyr", "stringr","caret","gbm","rpart","DescTools","forcats","
```

```
##
## The downloaded binary packages are in
##  /var/folders/8g/tqfmt0ys1w9dhfqph70wgyjr0000gn/T//RtmpUyZPAr/downloaded_packages
```

```
library(dplyr)
library(ggplot2)
library(tidyr)
library(stringr)
library(caret)
library(gbm)
library(rpart)
library(DescTools)
```

```
## Warning: package 'DescTools' was built under R version 4.3.3
```

```
##
## Attaching package: 'DescTools'
```

```
## The following objects are masked from 'package:caret':
##
##     MAE, RMSE
```

```
library(forcats)
library(tidyverse)
```

```
#load the dataset
dataset <- read_excel("/Users/Shared/final_proj/true_car_listings.xlsx")


# Displaying the original number of rows
original_row_count <- nrow(dataset)
cat("Original number of rows: ", original_row_count, "\n")
```

```
## Original number of rows:  121347
```

```
#Data Cleaning
# Randomly sampling it to 10,000 rows
set.seed(123) # For reproducibility
data <- dataset %>% sample_n(10000)

head(data)
```

```
## # A tibble: 6 x 8
##   Price  Year Mileage City         State Vin               Make      Model
```

```
##   <dbl> <dbl>   <dbl> <chr>        <chr> <chr>             <chr>     <chr>
## 1 39888  2015   31700 High Point   NC    5UXKR0C53F0K64021 BMW       X5xDrive35i
## 2 18697  2017   24538 Union City   GA    1G4PR5SK9H4102851 Buick     VeranoSport
## 3 10888  2009  142100 Vallejo      CA    19UUA86249A021349 Acura     TL4dr
## 4 17900  2011   71578 Bentonville  AR    WBAPM5C5XBE578813 BMW       3
## 5 14289  2015    9824 Cincinnati   OH    1G1PC5SB5F7259142 Chevrolet Cruze1LT
## 6 10900  2013   96577 Indianapolis IN    1G11C5SAXDU140675 Chevrolet Malibu1LT
```

```r
summary(data)
```

```
##      Price             Year         Mileage           City
##  Min.   :  1718   Min.   :1997   Min.   :     5   Length:10000
##  1st Qu.: 13995   1st Qu.:2012   1st Qu.: 21640   Class :character
##  Median : 19000   Median :2014   Median : 37512   Mode  :character
##  Mean   : 22803   Mean   :2014   Mean   : 46902
##  3rd Qu.: 28995   3rd Qu.:2016   3rd Qu.: 64884
##  Max.   :234995   Max.   :2018   Max.   :292415
##     State               Vin                Make               Model
##  Length:10000       Length:10000       Length:10000       Length:10000
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
```

```r
str(data)
```

```
## tibble [10,000 x 8] (S3: tbl_df/tbl/data.frame)
##  $ Price  : num [1:10000] 39888 18697 10888 17900 14289 ...
##  $ Year   : num [1:10000] 2015 2017 2009 2011 2015 ...
##  $ Mileage: num [1:10000] 31700 24538 142100 71578 9824 ...
##  $ City   : chr [1:10000] "High Point" "Union City" "Vallejo" "Bentonville" ...
##  $ State  : chr [1:10000] "NC" "GA" "CA" "AR" ...
##  $ Vin    : chr [1:10000] "5UXKR0C53F0K64021" "1G4PR5SK9H4102851" "19UUA86249A021349" "WBAPM5C5XBE578
##  $ Make   : chr [1:10000] "BMW" "Buick" "Acura" "BMW" ...
##  $ Model  : chr [1:10000] "X5xDrive35i" "VeranoSport" "TL4dr" "3" ...
```

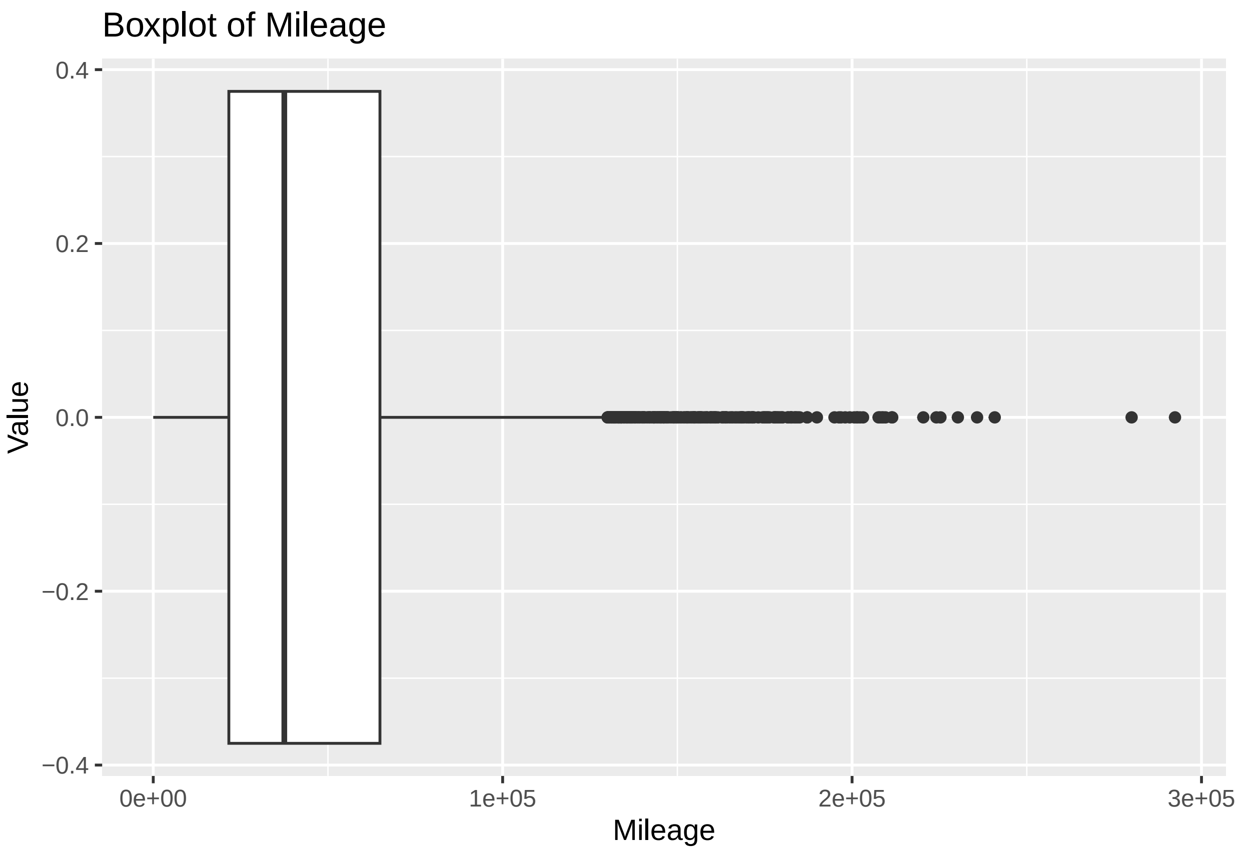
```r
# Checking if there are missing values
sum(is.na(data))
```

```
## [1] 0
```

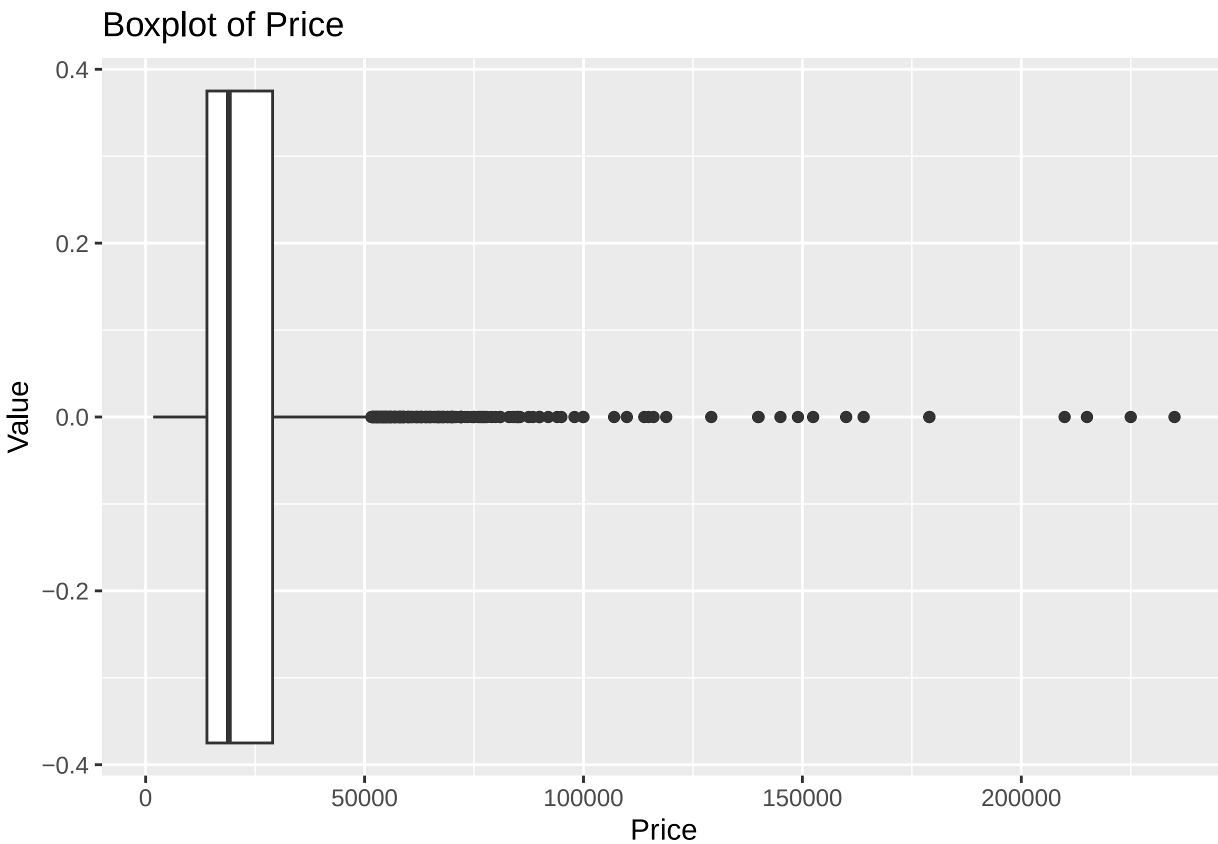
```r
# Convert categorical variables to factors
data$City <- as.factor(data$City)
data$State <- as.factor(data$State)
data$Make <- as.factor(data$Make)
data$Model <- as.factor(data$Model)

# Remove unnecessary columns because vin number is not needed for the prediction
data <- data %>% select(-Vin)
```

```r
#check for outliers
# Plot to check for outliers in Mileage
ggplot(data, aes(x = Mileage)) +
  geom_boxplot() +
  ggtitle("Boxplot of Mileage") +
  xlab("Mileage") +
```

```
  ylab("Value")
```




```
# Plot to check for outliers in Price
ggplot(data, aes(x = Price)) +
  geom_boxplot() +
  ggtitle("Boxplot of Price") +
  xlab("Price") +
  ylab("Value")
```




```r
# Calculate IQR for Mileage
iqr_mileage <- IQR(data$Mileage, na.rm = TRUE)
q1_mileage <- quantile(data$Mileage, 0.25, na.rm = TRUE)
q3_mileage <- quantile(data$Mileage, 0.75, na.rm = TRUE)
lower_bound_mileage <- q1_mileage - 1.5 * iqr_mileage
upper_bound_mileage <- q3_mileage + 1.5 * iqr_mileage

# Identify outliers in Mileage
outliers_mileage <- data %>%
  filter(Mileage < lower_bound_mileage | Mileage > upper_bound_mileage)

# Print outliers for Mileage
print("Outliers in Mileage:")
```

```
## [1] "Outliers in Mileage:"
```

```r
print(outliers_mileage)
```

```
## # A tibble: 303 x 7
##    Price  Year Mileage City          State Make      Model
##    <dbl> <dbl>   <dbl> <fct>         <fct> <fct>     <fct>
##  1 10888  2009  142100 Vallejo       CA    Acura     TL4dr
##  2  9520  2008  156170 Hazelwood     MO    Acura     MDX4WD
##  3  8500  2011  138711 St. Peters    MO    Chevrolet EquinoxFWD
##  4  5933  2002  178714 Hayward       CA    Acura     RSXType-S
##  5  7985  2007  144204 Cherry Hill   NJ    BMW       5
##  6  3999  1998  130565 Waite Park    MN    Buick     Park
##  7 11690  2007  145737 Fox Lake      IL    Acura     MDX4WD
```

```
## 8  3100  2002  137945 Portsmouth   VA    Buick     Park
## 9  9960  2012  131589 Mt. Pocono   PA    Chevrolet EquinoxAWD
## 10  7995  2008  168975 Jacksonville FL    Acura     MDX4WD
## # i 293 more rows
```

```
# Calculate IQR for Price
iqr_price <- IQR(data$Price, na.rm = TRUE)
q1_price <- quantile(data$Price, 0.25, na.rm = TRUE)
q3_price <- quantile(data$Price, 0.75, na.rm = TRUE)
lower_bound_price <- q1_price - 1.5 * iqr_price
upper_bound_price <- q3_price + 1.5 * iqr_price

# Identify outliers in Price
outliers_price <- data %>%
  filter(Price < lower_bound_price | Price > upper_bound_price)

# Print outliers for Price
print("Outliers in Price:")
```

```
## [1] "Outliers in Price:"
```

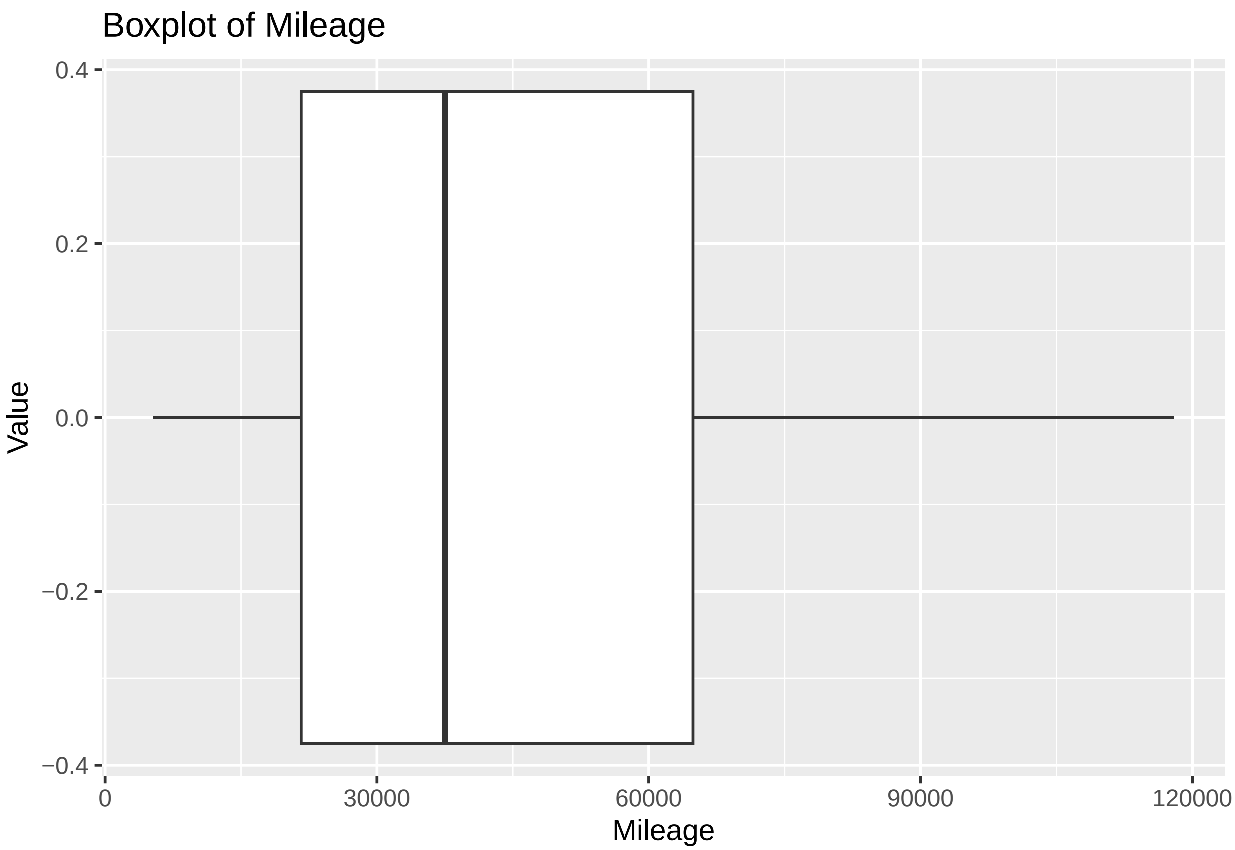
```
print(outliers_price)
```

```
## # A tibble: 308 x 7
##    Price  Year Mileage City             State Make     Model
##    <dbl> <dbl>   <dbl> <fct>            <fct> <fct>    <fct>
##  1 53700  2015   31436 Little Rock      AR    BMW      7
##  2 64200  2017   23085 Raleigh          NC    Cadillac Escalade
##  3 56855  2017    8229 Des Plaines      IL    Audi     Q73.0T
##  4 55777  2014    8082 Woburn           MA    Cadillac Escalade
##  5 55222  2015   43102 Phoenix          AZ    Cadillac Escalade
##  6 62000  2016   12738 Beaverton        OR    BMW      5
##  7 55395  2018     500 Spring Valley    NY    BMW      5
##  8 68900  2016   12088 Englewood Cliffs NJ    Cadillac Escalade4WD
##  9 62776  2016   26636 Brigham City     UT    Cadillac Escalade
## 10 52990  2014   24728 Shoreline        WA    Cadillac CTS-V
## # i 298 more rows
```

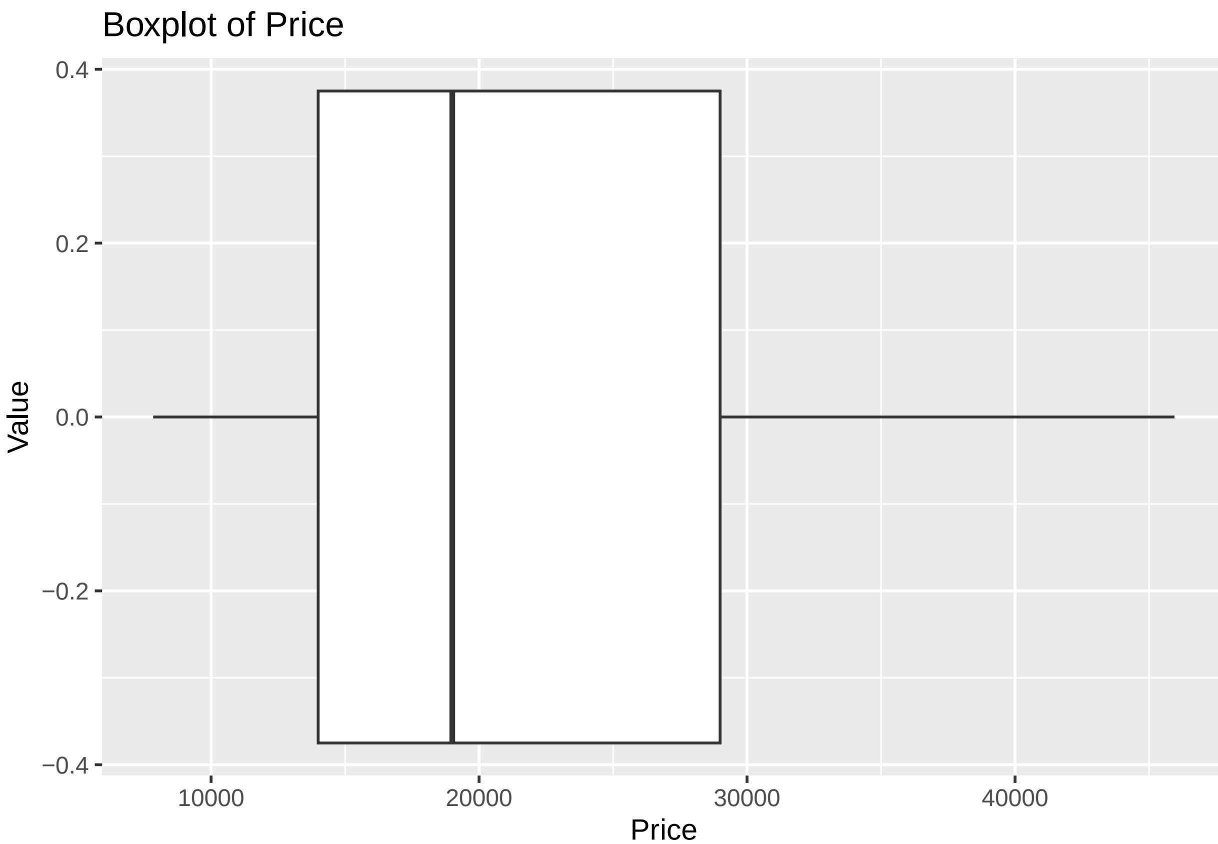
```
# Calculate the percentiles for Winsorization
lower_bound_mileage <- quantile(data$Mileage, 0.05, na.rm = TRUE)
upper_bound_mileage <- quantile(data$Mileage, 0.95, na.rm = TRUE)

# Winsorize Mileage
data$Mileage <- pmin(pmax(data$Mileage, lower_bound_mileage), upper_bound_mileage)

# Calculate the percentiles for Price
lower_bound_price <- quantile(data$Price, 0.05, na.rm = TRUE)
upper_bound_price <- quantile(data$Price, 0.95, na.rm = TRUE)

# Winsorize Price
data$Price <- pmin(pmax(data$Price, lower_bound_price), upper_bound_price)
```

```
# Plot to check if outliers still exist after Winsorization
ggplot(data, aes(x = Mileage)) +
  geom_boxplot() +
  ggtitle("Boxplot of Mileage") +
```

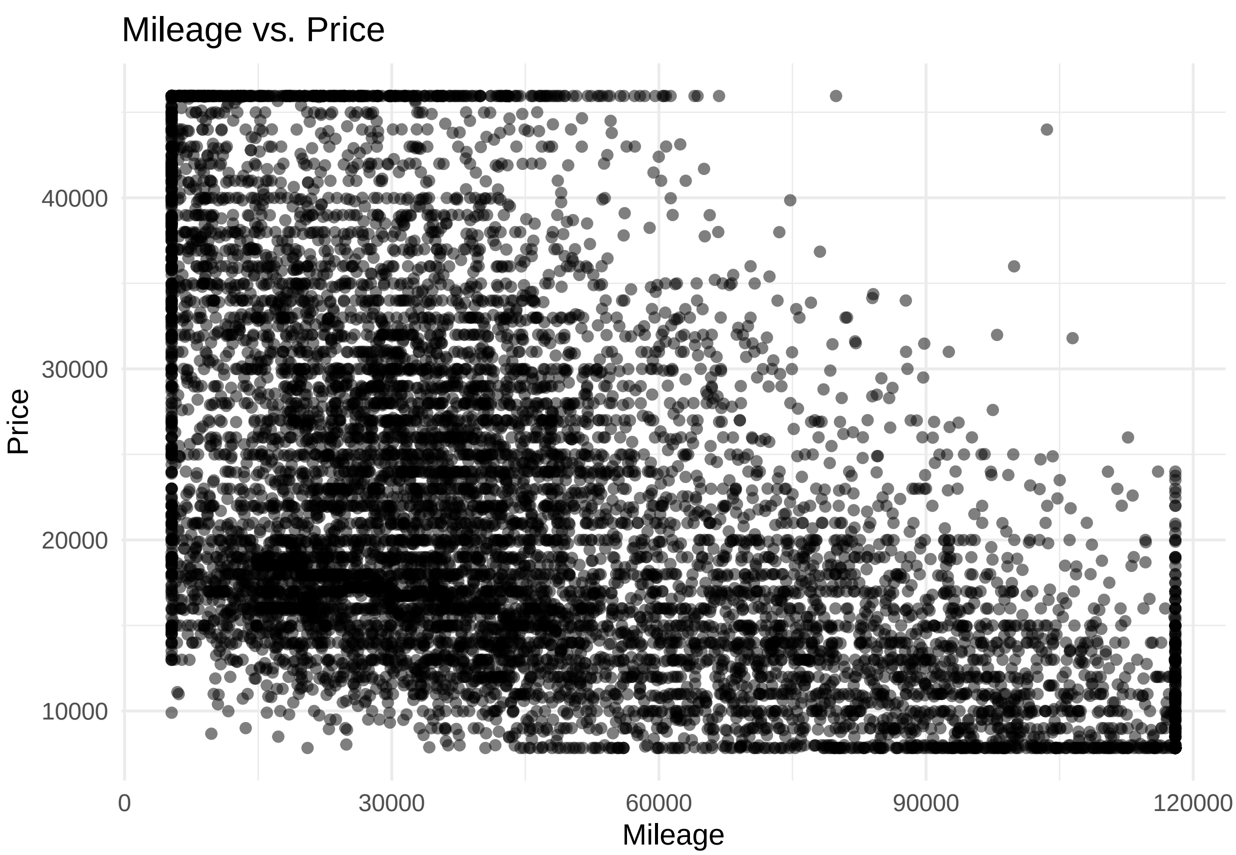
```r
  xlab("Mileage") +
  ylab("Value")
```



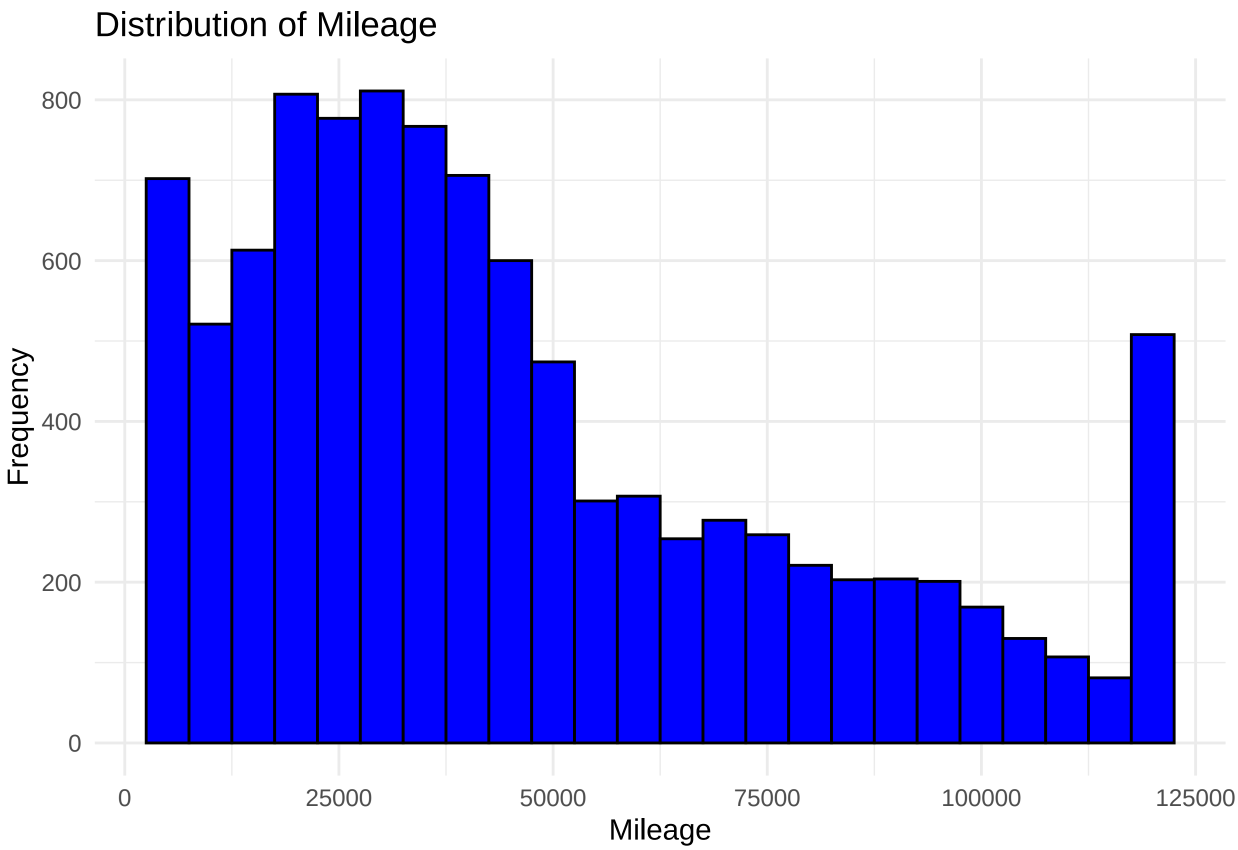

```r
# Plot to check for outliers in Price
ggplot(data, aes(x = Price)) +
  geom_boxplot() +
  ggtitle("Boxplot of Price") +
  xlab("Price") +
  ylab("Value")
```



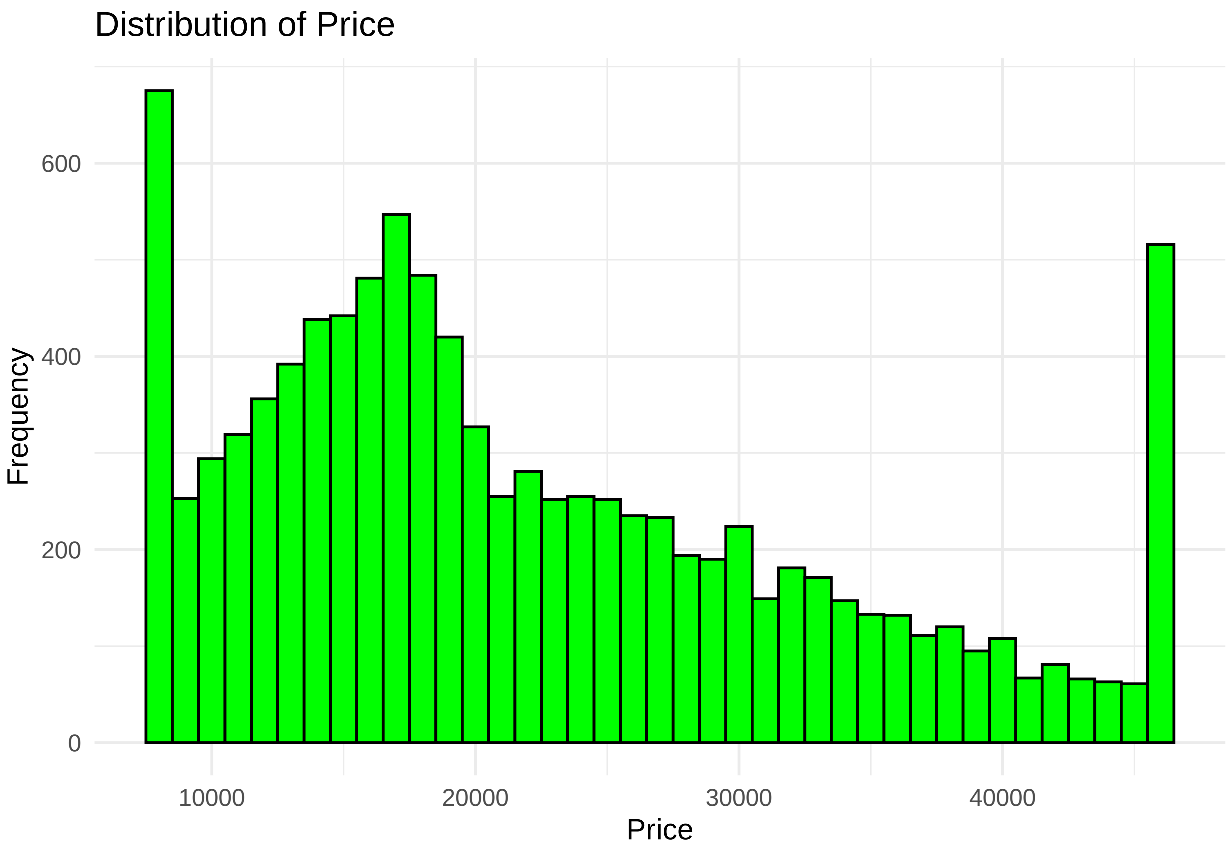

```r
#Scatter plots can help understand the relationship between Mileage and Price.
ggplot(data, aes(x = Mileage, y = Price)) +
  geom_point(alpha = 0.5) +
  labs(title = "Mileage vs. Price", x = "Mileage", y = "Price") +
  theme_minimal()
```



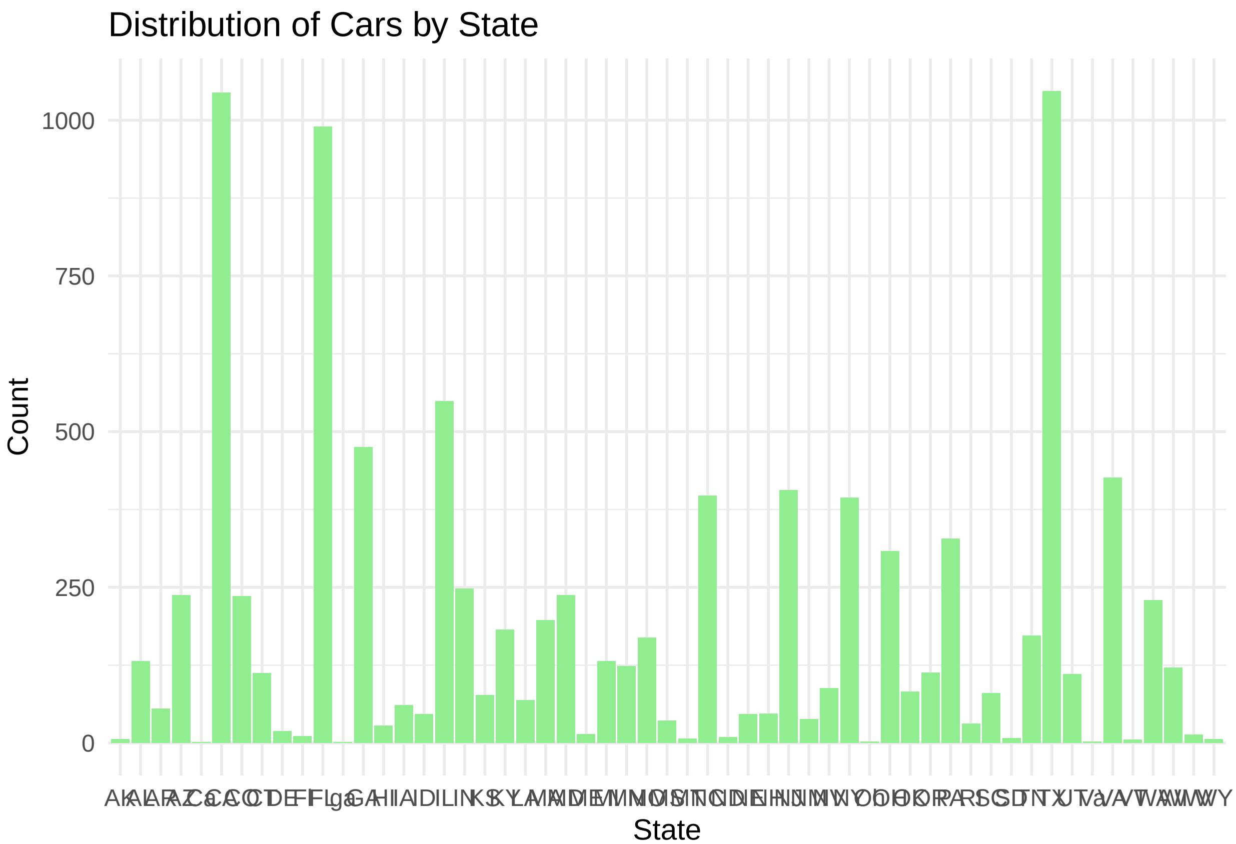

```r
#Histograms help visualize the distribution of continuous variables.
# Histogram for Mileage
ggplot(data, aes(x = Mileage)) +
  geom_histogram(binwidth = 5000, fill = "blue", color = "black") +
  labs(title = "Distribution of Mileage", x = "Mileage", y = "Frequency") +
  theme_minimal()
```




```r
# Histogram for Price
ggplot(data, aes(x = Price)) +
  geom_histogram(binwidth = 1000, fill = "green", color = "black") +
  labs(title = "Distribution of Price", x = "Price", y = "Frequency") +
  theme_minimal()
```




```r
# Bar plot for State
ggplot(data, aes(x = State)) +
  geom_bar(fill = "lightgreen") +
  labs(title = "Distribution of Cars by State", x = "State", y = "Count") +
  theme_minimal()
```




```r
#Split the data into training and testing sets
set.seed(123)  # For reproducibility
trainIndex <- createDataPartition(data$Price, p = 0.8, list = FALSE)
dataTrain <- data[trainIndex, ]
dataTest <- data[-trainIndex, ]

# Combine rare levels in City
dataTrain$City <- fct_lump(dataTrain$City, n = 50)  # Keep top 50 levels, lump the rest
dataTest$City <- fct_lump(dataTest$City, n = 50)    # Apply same to test data
```

```r
#Linear Regression
lm_model <- lm(Price ~ Mileage + Year + City + State + Make + Model, data = dataTrain)
```

```r
# Ensure the levels of factors in the test set match those in the training set
dataTest$City <- factor(dataTest$City, levels = levels(dataTrain$City))
dataTest$State <- factor(dataTest$State, levels = levels(dataTrain$State))
dataTest$Make <- factor(dataTest$Make, levels = levels(dataTrain$Make))
dataTest$Model <- factor(dataTest$Model, levels = levels(dataTrain$Model))
```

```r
#Decision Tree Model
tree_model <- rpart(Price ~ Mileage + Year + City + State + Make + Model, data = dataTrain)

# Make predictions on the test dataset
tree_predictions <- predict(tree_model, newdata = dataTest)

# Calculate performance metrics
mae_tree <- mean(abs(dataTest$Price - tree_predictions))
mse_tree <- mean((dataTest$Price - tree_predictions)^2)
```

```r
rsq_tree <- 1 - sum((dataTest$Price - tree_predictions)^2) / sum((dataTest$Price - mean(dataTest$Price)

# Print the performance metrics
print(paste("MAE:", mae_tree))
```

```
## [1] "MAE: 3566.25086375629"
```

```r
print(paste("MSE:", mse_tree))
```

```
## [1] "MSE: 22644828.5752094"
```

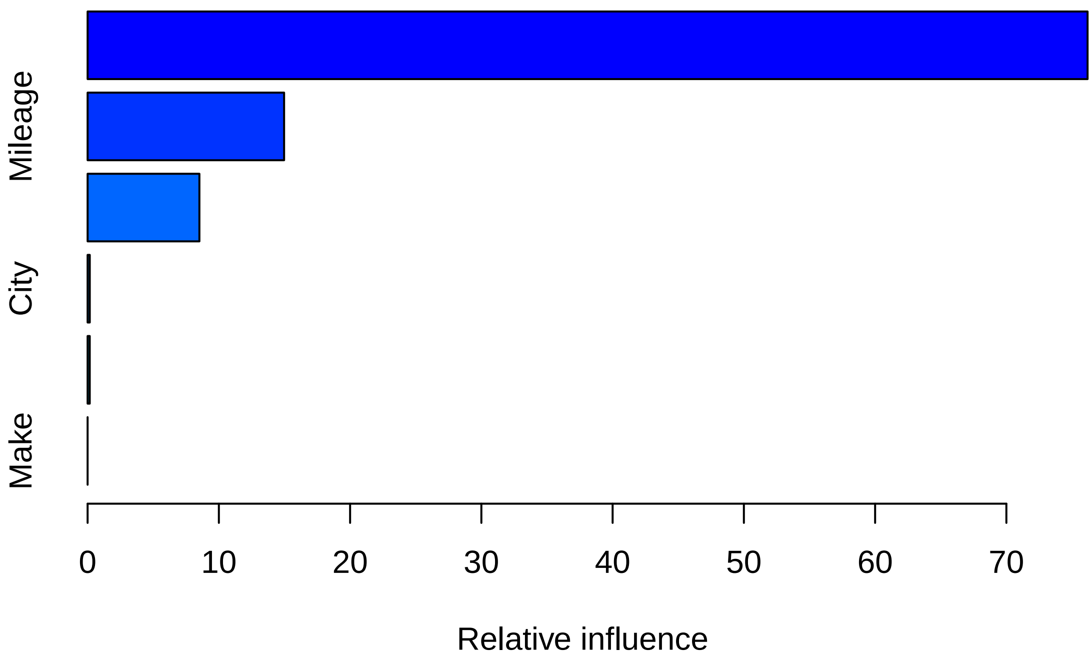
```r
print(paste("R-squared:", rsq_tree))
```

```
## [1] "R-squared: 0.79914982027344"
```

```r
#Gradient Boosting Machine
gbm_model <- gbm(Price ~ Mileage + Year + City + State + Make + Model,
                 data = dataTrain,
                 distribution = "gaussian",
                 n.trees = 100,
                 interaction.depth = 3)
summary(gbm_model)
```

```
##             var    rel.inf
## Model     Model 76.1865391
## Mileage Mileage 14.9696036
## Year       Year  8.5181789
## City       City  0.1637060
## State     State  0.1619723
## Make       Make  0.0000000
```

```r
# Extract and print the variable importance
variable_importance <- summary(gbm_model)
```




```r
# Sort the variable importance scores
variable_importance <- variable_importance[order(-variable_importance$rel.inf), ]
```

```r
# Print the variable importance
print(variable_importance)
```

```
##             var    rel.inf
## Model     Model 76.1865391
## Mileage Mileage 14.9696036
## Year       Year  8.5181789
## City       City  0.1637060
## State     State  0.1619723
## Make       Make  0.0000000
```

```r
#predicting using gbm model on testdata
gbm_pred <- predict(gbm_model, newdata = dataTest, n.trees = 100)

# Calculate performance metrics
mae <- mean(abs(dataTest$Price - gbm_pred))
mse <- mean((dataTest$Price - gbm_pred)^2)
rsq <- 1 - sum((dataTest$Price - gbm_pred)^2) / sum((dataTest$Price - mean(dataTest$Price))^2)

print(paste("MAE:", mae))
```

```
## [1] "MAE: 2069.9477655695"
```

```r
print(paste("MSE:", mse))
```

```
## [1] "MSE: 9068276.87086404"
```

```r
print(paste("R-squared:", rsq))
```

```
## [1] "R-squared: 0.91956816836683"
```

```r
#since gbm model is performing better than tree model
#making predictions on custom data using gbm model
custom_data <- data.frame(
  Mileage = c(30000, 50000, 100000),
  Year = c(2020, 2018, 2015),
  City = c("San Francisco", "Los Angeles", "Chicago"),
  State = c("CA", "CA", "IL"),
  Make = c("Toyota", "Honda", "Ford"),
  Model = c("Camry", "Civic", "Focus")
)

# Ensure factor levels match those in the training data
custom_data$City <- factor(custom_data$City, levels = levels(dataTrain$City))
custom_data$State <- factor(custom_data$State, levels = levels(dataTrain$State))
custom_data$Make <- factor(custom_data$Make, levels = levels(dataTrain$Make))
custom_data$Model <- factor(custom_data$Model, levels = levels(dataTrain$Model))

# Predict on the custom data
custom_predictions <- predict(gbm_model, newdata = custom_data)
```

```
## Using 100 trees...
```

```r
# Print the predictions
print(custom_predictions)
```

```
## [1] 22563.98 22563.98 21315.51
```

```r
# Calculate errors between predictions and actual values
errors <- abs(dataTest$Price - gbm_pred)

# Define a threshold for acceptable error (e.g., 10% of the actual price)
threshold <- 0.10
acceptable_error <- dataTest$Price * threshold

# Calculate percentage of predictions within the acceptable error range
correct_predictions <- sum(errors <= acceptable_error)
total_predictions <- length(errors)
percentage_correct <- (correct_predictions / total_predictions) * 100

# Print the percentage of correct predictions
print(paste("Percentage of Correct Predictions:", round(percentage_correct, 2), "%"))
```

```
## [1] "Percentage of Correct Predictions: 62.11 %"
```