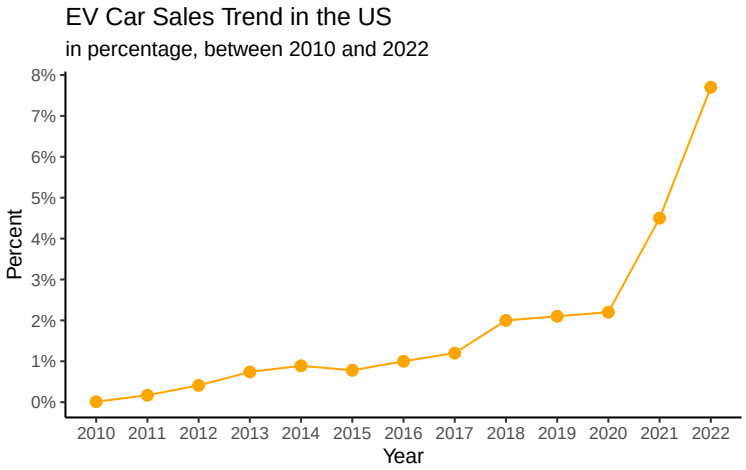
Maximizing Electric Car Value

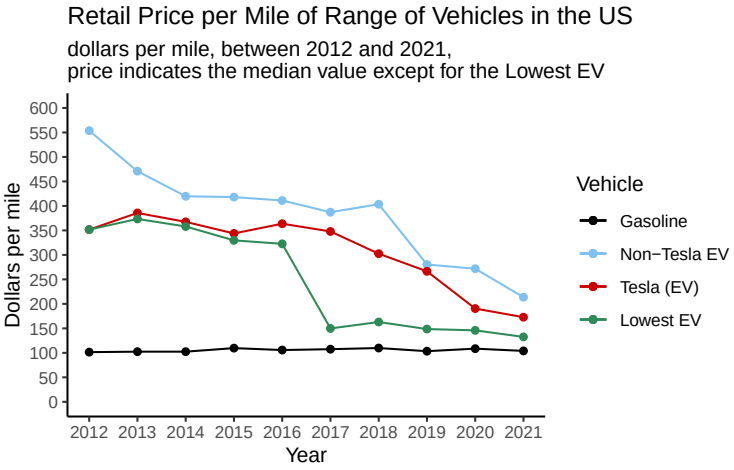
A brief case study on exploring trends with applications of Linear Regression

Matthew Balogh

Report Overview



(a) **How does EV car sales compare to non-EV sales?** EV car sales are on the rise compared to all car sales, especially after 2020 with a steep increase.



(b) **How did retail price per mile of range change over this period?** The indicator for gasoline cars remained steady. While there is still a gap in between EVs and non-EVs, it has decreased substantially over the displyed period.

Sales Trend in the US

The data sources from the Global EV Outlook of IEA.

Question: How does EV car sales compare to non-EV sales?

The results are covered by Figure 1a.

Data Import

The data is imported from a CSV file using the `read_csv` function from the `tidyverse` package.

```
library(tidyverse)

ev_sales <- read_csv(paste("path", "to", "dataset.csv", sep = "/"))
```

Characteristics of the dataset

It contains aggregated historical values for *electrical cars* including **sales** and **sales share** information for various aspects combined in individual rows.

The dataset has the following columns:

```
[1] "region"     "category"   "parameter"  "mode"       "powertrain"
[6] "year"       "unit"       "value"
```

The columns have *syntactic* names which suit the following operations on the dataset without the need of correcting them.

Data Transformation

Simply importing this dataset into `R` would result in `character` and `numerical` variables. However, there are categorical variables with a well-defined set of possible values: *region, category, parameter, mode, powertrain, unit*.

These variables can be converted into a more suitable type called `factor` with the following mutation operation.

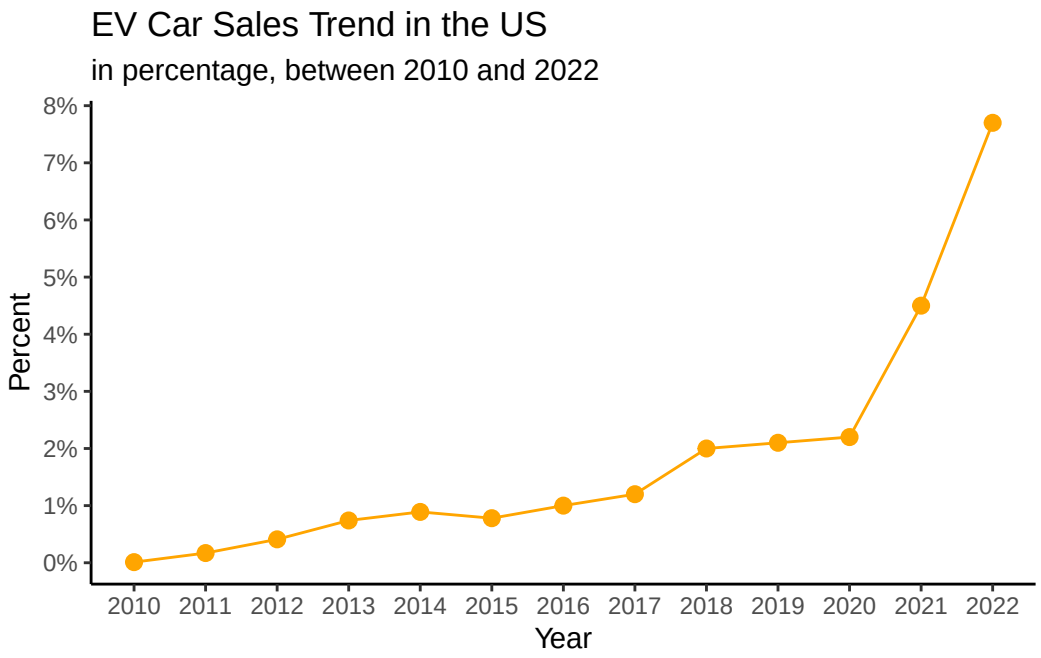
```
ev_sales <- ev_sales |>
    mutate(region = factor(region),
        category = factor(category),
        parameter = factor(parameter),
        mode = factor(mode),
        powertrain = factor(powertrain),
        unit = factor(unit))
```

Data Filtering

Since the dataset contains multiple regions around the world and aspects of sales, we need to filter for the data related to the **US** and **sales share** as follows.

```
us_sales_share <- ev_sales |>
    filter(region == "USA" & parameter == "EV sales share")
```

Data Visualization



To achieve the above graph, I used `ggplot` from the `ggplot2` package with the layers of `geom_point` and `geom_line`.

```
ggplot(us_sale_share, aes(x = year, y = value)) +
    geom_point(col = "orange", size = 2.5) +
    geom_line(col = "orange")
```

As the `mapping` argument of `ggplot` shows above, the abscissa is set to the scale of **year** and the ordinate is to the *value* of **sales share** in *percent*.

Scales of the axes have been set up to show 1-step breaks. To indicate that the y-axis is expressed in percent, I used the `label_percent` function from the package `scales` with a scale of 1 as follows:

```
library(scales)

ggplot(...) +
    ... +
    scale_x_continuous(breaks = seq(2010, 2022)) +
    scale_y_continuous(breaks = seq(0, 8), labels = label_percent(scale = 1))
```

Finally, to give a simple appearance to the visual, I enabled a classic theme.

```
ggplot(...) +
    ... +
    theme_classic()
```

Retail Price per Mile trend in the US

The data sources from the report of Electric vehicles gain ground but still face price, range, charging constraints.

Question: How did retail price per mile of range change over the last decade or so?

The results are covered by Figure 1b.

Data Import

The data is imported from a CSV file just like before.

```
library(tidyverse)

dollars_per_mile <- read_csv(paste("path", "to", "dataset.csv", sep = "/"))
```

Characteristics of the dataset

It contains historical median values of **dollars per mile of range** for different types of vehicles. The last column, however, contains the lowest EV price / miles indicator.

The dataset has the following columns:

```
[1] "Year"
[2] "Gasoline vehicles"
[3] "Electric vehicles (excluding Tesla)"
[4] "Tesla"
[5] "Electric vehicles (lowest MSRP/range)"
```

Most of the columns contain space character, that is, they have *non-syntactic* name, however, no column requires to be transformed into a `factor` variable.

Data Transformation

To make it convenient to work with the dataset, the column names shall be transformed.

There are multiple ways to transform column names into *syntactic* ones. One approach uses an automatic transformation with the help of `janitor::clean_names()`, while the other one is a manual approach with the `rename` function.

```
library(janitor)

dollars_per_mile |>
    janitor::clean_names()
```

```
[1] "year"                                "gasoline_vehicles"
[3] "electric_vehicles_excluding_tesla"   "tesla"
[5] "electric_vehicles_lowest_msrp_range"
```

Since the above column names seem to be too long, I have decided to go with the manual approach.

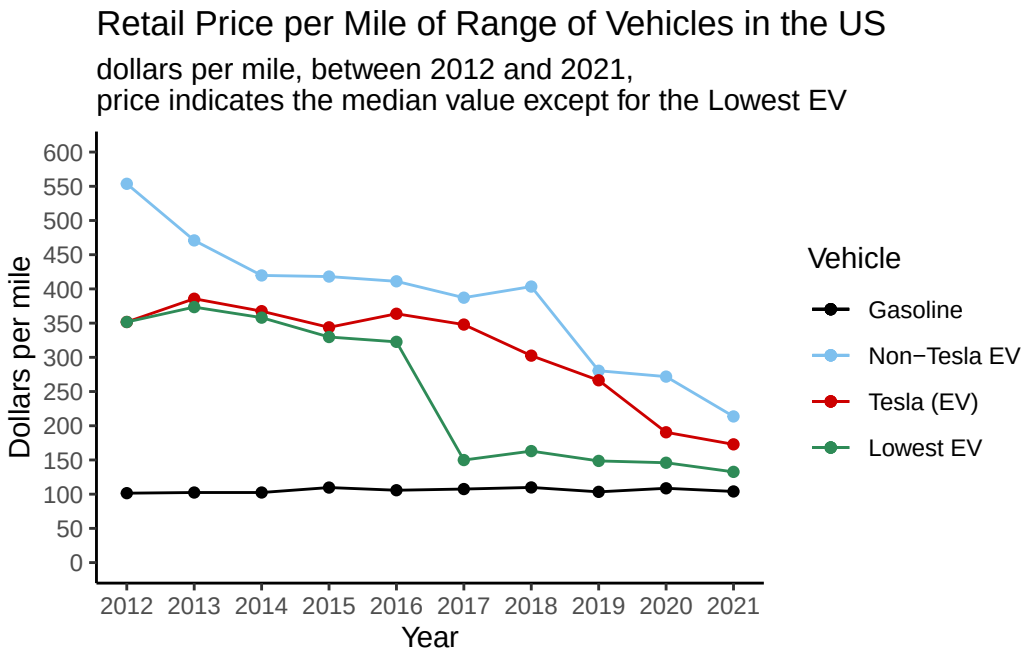
```
dollars_per_mile <- dollars_per_mile |>
    rename(
        year = "Year",
        gasoline = "Gasoline vehicles",
        ev_no_tesla = "Electric vehicles (excluding Tesla)",
        ev_tesla = "Tesla",
        ev_lowest = "Electric vehicles (lowest MSRP/range)"
    )
```

```
[1] "year"        "gasoline"    "ev_no_tesla" "ev_tesla"    "ev_lowest"
```

Data Filtering

Since the data of different vehicle types are separated by columns not rows, there is no need for filtering. When the types are needed to be separated, the geom layer shall use the appropriate variable in its `mapping`.

Data Visualization



To achieve the above plot, I used pairs of layers of `geom_point` and `geom_line` for all the four indicators as follows.

```
ggplot(dollars_per_mile, aes(x = year)) +
    # Gasoline
    geom_point(aes(y = gasoline, col = "1")) +
    geom_line(aes(y = gasoline, col = "1")) +
    # EV non-Tesla
    geom_point(aes(y = ev_no_tesla, col = "2")) +
    geom_line(aes(y = ev_no_tesla, col = "2")) +
    # EV Tesla
    geom_point(aes(y = ev_tesla, col = "3")) +
    geom_line(aes(y = ev_tesla, col = "3")) +
    # Lowest EV indicator
```

```
    geom_point(aes(y = ev_lowest, col = "4")) +
    geom_line(aes(y = ev_lowest, col = "4"))
```

The abscissa is set to the scale of **year** and the ordinate is to the **dollars per mile**. Since the variable on the x scale is universal for all the layers, I specified it in the `mapping` of `ggplot` itself and extended the mapping with the appropriate y variable specification in case of each different indicator.

Scales of the axes have been set up to show 1 and 50-step breaks for x and y, respectively.

```
ggplot(...) +
    ... +
    scale_x_continuous(breaks = seq(2012, 2021)) +
    scale_y_continuous(breaks = seq(0, 600, by = 50), limits = c(0, 600))
```

To highlight the layers with different colors, I specified the `color mapping` attribute of each `geom_point` and `geom_line` with an ordered sequence and then based on that I specified the colors manually. Also, a readable label has been added to each of these colored groups.

```
ggplot(...) +
    ...
    geom_point(aes(..., col = "1")) +
    geom_line(aes(..., col = "1")) +
    ... +
    scale_color_manual(
        values = c("black", "skyblue2", "red3", "seagreen"),
        labels = c("Gasoline", "Non-Tesla EV", "Tesla (EV)", "Lowest EV")
    )
```

Finally, to give a simple appearance to the visual, I enabled a classic theme as I did in the case of the previous case.