stem-fuel

July 23, 2021

1 Vehicle Fuel Economy Estimates Data Set

Use this notebook to practice your exploratory data analysis and visualization skills.

EXPLORE: Click on the Edit App button in the top right corner of this page, to interact with this Jupyter notebook To navigate through **this notebook** simply press the shift + Enter keys to run each **block of code**.

You can include new blocks of code or text throughout the notebook to continue exploring and programming with R.

(New to Jupyter notebooks? See the help page for more on working with Jupyter notebooks)

The **original dataset** is obtained from FuelEconomy.gov Web Services. The 1984-2017 fuel economy data is produced during vehicle testing at the **Environmental Protection Agency's (EPA) National Vehicle and Fuel Emissions Laboratory** in Ann Arbor, Michigan, and by vehicle manufacturers with EPA oversight. Check also the data in this Kaggle page.

The version of the data used in this notebook is also available in this repo.

1.1 Load packages

```
In [151]: # use data science tools from the tidyverse
          library(tidyverse)
```

1.2 Read the data

The adapted dataset used in this notebook contains more than 38,000 observations and 81 variables are available! (We will focus on a small subset of the attributes for this initial exploration). A related data dictionary can be found at https://www.fueleconomy.gov/feg/ws/

EPA's fuel economy values are good estimates of the fuel economy a typical driver will achieve under average driving conditions and provide a good basis to compare one vehicle to another. Fuel economy varies, sometimes significantly, based on driving conditions, driving style, and other factors.

Below we read the `.csv` file using `readr::read_csv()` (the `readr` package is part of the `tidyverse`)

```
In [152]: fuel <- read_csv("../data/fuel.csv", col_types = cols())
```

Check the dimensions of the dataset:

```
In [153]: dim(fuel)
```

1. 38113 2. 81

1.3 Data Exploration

```
In [154]: # random sample of the data
          set.seed(217)       # this sets a random seed for reproducibility
          fuel %>%
             sample_n(7)
```

vehicle_id	year	make	model	class	drive
13264	1997	Honda	Del Sol	Two Seaters	Front-Wheel D
4074	1987	Jeep	Cherokee/Wagoneer 4WD	Special Purpose Vehicles	4-Wheel or Al
8495	1991	Ford	Bronco 4WD	Special Purpose Vehicles	4-Wheel or Al
32292	2012	Porsche	New 911 Carrera S	Minicompact Cars	Rear-Wheel D
1038	1985	Chevrolet	S10 Blazer 2WD	Special Purpose Vehicle 2WD	Rear-Wheel D
23313	2007	Audi	S4 Avant	Small Station Wagons	4-Wheel or Al
24783	2008	BMW	335ci	Subcompact Cars	Rear-Wheel D

We can see the range (minimum and maximum) of a variable using the `range()` function:

```
In [155]: range(fuel$year)
```

1. 1984 2. 2017

We can also use the `dplyr::sumarize()` function to get some summaries for certain variables:

```
In [156]: fuel %>%
            summarize(minmax_year = range(year),
                      minmax_fuel_cost = range(annual_fuel_cost_ft1),
                      minmax_barrels = range(annual_consumption_in_barrels_ft1))
```

minmax_year	minmax_fuel_cost	minmax_barrels
1984	500	0.06000
2017	6050	47.08714

For variables that are encoded as categorical, we can also get counts. First, below is a trick to find which variables are encoded as *character* (this will help you determine which ones are actually categorical variables: for example an email is stored as a character, but we may not treat is a category since it may be unique, while colors and brands could be treated as categorical):

```
In [157]: # select variables that are of type character
          fuel %>%
             select_if(is.character)
```

make	model	class	drive
Alfa Romeo	GT V6 2.5	Minicompact Cars	NA
Alfa Romeo	GT V6 2.5	Minicompact Cars	NA
Alfa Romeo	Spider Veloce 2000	Two Seaters	NA
Alfa Romeo	Spider Veloce 2000	Two Seaters	NA
AM General	DJ Po Vehicle 2WD	Special Purpose Vehicle 2WD	2-Wheel Driv
AM General	DJ Po Vehicle 2WD	Special Purpose Vehicle 2WD	2-Wheel Driv
AM General	FJ8c Post Office	Special Purpose Vehicle 2WD	2-Wheel Driv
AM General	FJ8c Post Office	Special Purpose Vehicle 2WD	2-Wheel Driv
American Motors Corporation	Eagle 4WD	Special Purpose Vehicle 4WD	4-Wheel or A
American Motors Corporation	Eagle 4WD	Special Purpose Vehicle 4WD	4-Wheel or A
American Motors Corporation	Eagle 4WD	Special Purpose Vehicle 4WD	4-Wheel or A
American Motors Corporation	Eagle 4WD	Special Purpose Vehicle 4WD	4-Wheel or A
American Motors Corporation	Eagle 4WD	Special Purpose Vehicle 4WD	4-Wheel or A
American Motors Corporation	Eagle 4WD	Special Purpose Vehicle 4WD	4-Wheel or A
American Motors Corporation	Eagle 4WD	Special Purpose Vehicle 4WD	4-Wheel or A
American Motors Corporation	Eagle 4WD	Special Purpose Vehicle 4WD	4-Wheel or A
American Motors Corporation	Eagle 4WD	Special Purpose Vehicle 4WD	4-Wheel or A
American Motors Corporation	Eagle SX/4 4WD	Special Purpose Vehicle 4WD	4-Wheel or A
American Motors Corporation	Eagle SX/4 4WD	Special Purpose Vehicle 4WD	4-Wheel or A
American Motors Corporation	Eagle SX/4 4WD	Special Purpose Vehicle 4WD	4-Wheel or A
American Motors Corporation	Eagle SX/4 4WD	Special Purpose Vehicle 4WD	4-Wheel or A
American Motors Corporation	Eagle SX/4 4WD	Special Purpose Vehicle 4WD	4-Wheel or A
American Motors Corporation	Eagle SX/4 4WD	Special Purpose Vehicle 4WD	4-Wheel or A
American Motors Corporation	Eagle SX/4 4WD	Special Purpose Vehicle 4WD	4-Wheel or A
American Motors Corporation	Eagle SX/4 4WD	Special Purpose Vehicle 4WD	4-Wheel or A
American Motors Corporation	Eagle SX/4 4WD	Special Purpose Vehicle 4WD	4-Wheel or A
American Motors Corporation	Eagle SX/4 4WD	Special Purpose Vehicle 4WD	4-Wheel or A
American Motors Corporation	Eagle SX/4 4WD	Special Purpose Vehicle 4WD	4-Wheel or A
Aston Martin	Lagonda	Subcompact Cars	NA
Aston Martin	Lagonda	Subcompact Cars	NA
Volkswagen	Jetta	Compact Cars	Front-Wheel
Volkswagen	Jetta	Compact Cars	Front-Wheel
Volkswagen	Jetta	Compact Cars	Front-Wheel
Volkswagen	Passat	Midsize Cars	Front-Wheel
Volkswagen	Passat	Midsize Cars	Front-Wheel
Volkswagen	Tiguan	Small Sport Utility Vehicle 2WD	Front-Wheel
Volkswagen	Tiguan 4motion	Small Sport Utility Vehicle 4WD	All-Wheel D
Volkswagen	Touareg	Standard Sport Utility Vehicle 4WD	All-Wheel D
Volvo	S60 AWD	Compact Cars	All-Wheel D
Volvo	S60 AWD	Compact Cars	All-Wheel D
Volvo	S60 CC AWD	Compact Cars	All-Wheel D
Volvo	S60 FWD	Compact Cars	Front-Wheel
Volvo	S60 Inscription AWD	Compact Cars	All-Wheel D
Volvo	S60 Inscription FWD	Compact Cars	Front-Wheel
Volvo	S60 Polestar AWD	Compact Cars	All-Wheel D
Volvo	S90 AWD	Midsize Cars	All-Wheel D
Volvo	S90 FWD	Midsize Cars	Front-Wheel
Volvo	V60 AWD	Small Station Wagons	All-Wheel D
Volvo	V60 AWD	Small Station Wagons	All-Wheel D
Volvo	V60 CC AWD	Small Station Wagons	All-Wheel D
Volvo	V60 FWD	Small Station Wagons	Front-Wheel

Let us select check the number of observations for each class of vehicle (`class`)

```
In [158]: fuel %>%
            group_by(class) %>%
            count()
          # you could also use group_by() followed by summarize() where the
          # summary counts the number of rows using the n() function
```

class	n
Compact Cars	5508
Large Cars	1891
Midsize Cars	4395
Midsize Station Wagons	523
Midsize-Large Station Wagons	656
Minicompact Cars	1260
Minivan - 2WD	342
Minivan - 4WD	47
Small Pickup Trucks	538
Small Pickup Trucks 2WD	436
Small Pickup Trucks 4WD	218
Small Sport Utility Vehicle 2WD	403
Small Sport Utility Vehicle 4WD	526
Small Station Wagons	1499
Special Purpose Vehicle	1
Special Purpose Vehicle 2WD	613
Special Purpose Vehicle 4WD	302
Special Purpose Vehicles	1455
Special Purpose Vehicles/2wd	2
Special Purpose Vehicles/4wd	2
Sport Utility Vehicle - 2WD	1627
Sport Utility Vehicle - 4WD	2082
Standard Pickup Trucks	2354
Standard Pickup Trucks 2WD	1177
Standard Pickup Trucks 4WD	986
Standard Pickup Trucks/2wd	4
Standard Sport Utility Vehicle 2WD	182
Standard Sport Utility Vehicle 4WD	434
Subcompact Cars	4872
Two Seaters	1886
Vans	1141
Vans Passenger	2
Vans, Cargo Type	438
Vans, Passenger Type	311

```
In [159]: # alternative: using the group_by + summarize combination
          fuel %>%
            group_by(fuel_type) %>%
            summarize(n = n())
```

fuel_type	n
CNG	60
Diesel	1014
Electricity	133
Gasoline or E85	1223
Gasoline or natural gas	20
Gasoline or propane	8
Midgrade	77
Premium	10133
Premium and Electricity	25
Premium Gas or Electricity	18
Premium or E85	122
Regular	25258
Regular Gas and Electricity	20
Regular Gas or Electricity	2

When working with larger datasets like this one, chances are that several observations have missing values (NA) in some of the attributes available in the dataset. It is good practice to get a sense of the proportion of missing values for different variables. This may help you make design choices when exploring predictive models (e.g., how and what type of data imputation to incorporate - if any -, or deciding which variables have enough variation and are good choices for further analysis).

Below is a trick to easily get this information using tools from `dplyr`:

```
In [160]: fuel %>%
          summarize_all(~sum(is.na(.))/n())
```

vehicle_id	year	make	model	class	drive	transmission	transmission_type	engine_index
0	0	0	0	0	0.0311967	0.0002886154	0.6052528	0

The code above tells you that we have no missing values for the variables `year`, `make`, `model` and others; and it also indicates that the attribute `range_ft2` is an empty column (all observations have a missing value there). **Quick explanation**: `is.na()` returns either `TRUE` if the element is missing, and `FALSE` otherwise. When combined with the function `sum()`, any value of `TRUE` will be understood as a `1`, and instances of `FALSE` as `0` (this is known as ***coercion***). Therefore, adding all the `1`s will tell you how many observations have a missing value, and dividing by the number of observations (i.e., using `n()`) will give the proportion. Documentation for the `summarize_all()` function (and other similar functions) can be found here. Again, this shows the power of `dplyr`: just a few lines of code can give you very good information.

Practice: which other type of summaries can you create? Try grouping by *multiple* variables to analyze that set of observationss (e.g., grouping by `make` and `transmission` to analyze the fuel efficiency of cars in the different groups)

1.4 Some data visualizations

There are many observations and attributes (variables) available in this dataset. We will generate some data visualizations in this notebook that can help us confirm some of the things we would expect from the evolution and progress made in car manufacturing in recent years.

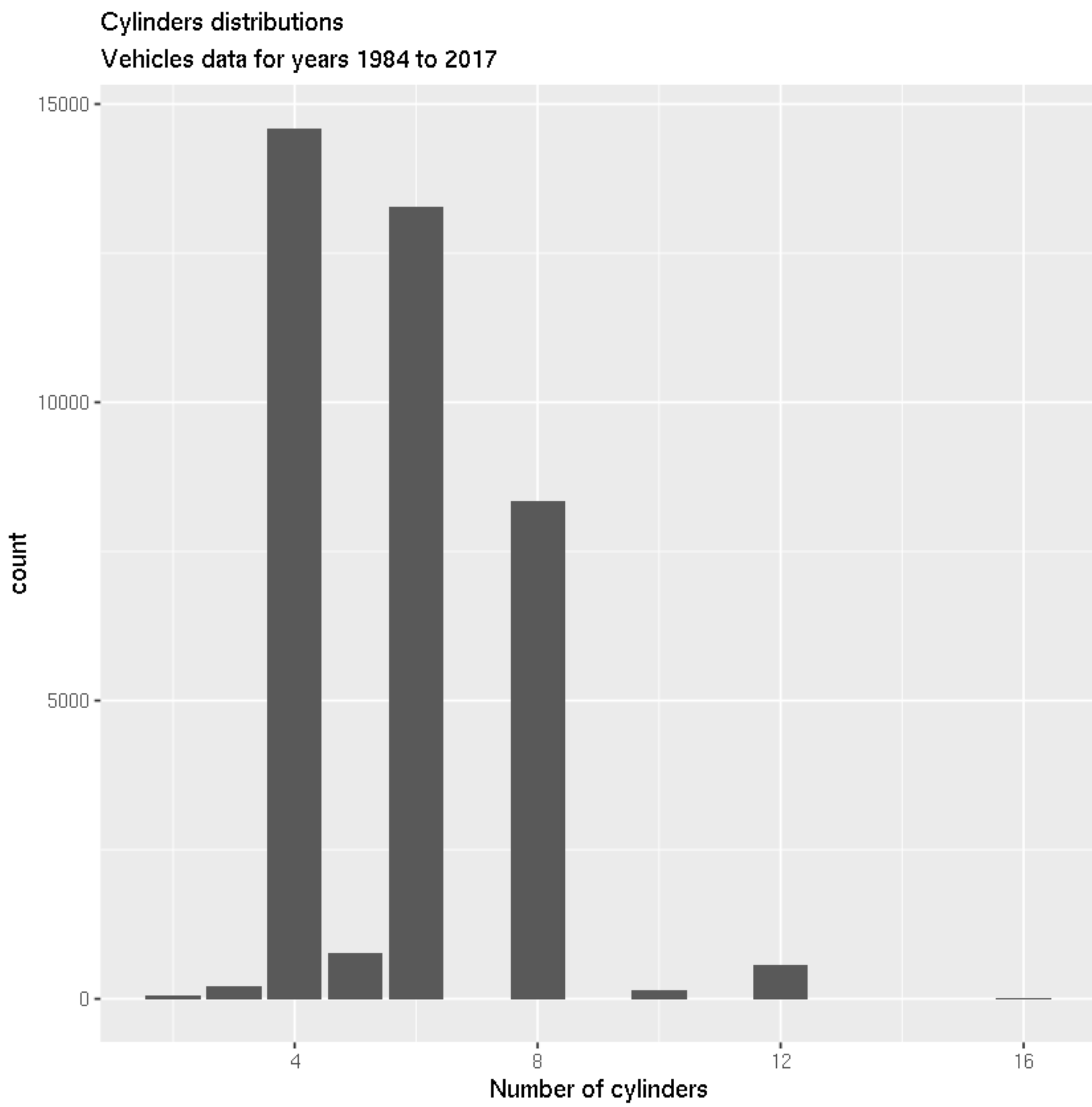
The purpose of EPA's fuel economy estimates is to provide a reliable basis for comparing vehicles. Most vehicles in the database (other than plug-in hybrids) have three fuel economy estimates: - a "city" estimate that represents urban driving, in which a vehicle is started in the morning (after being parked all night) and driven in stop-and-go traffic;

- a "highway" estimate that represents a mixture of rural and interstate highway driving in a warmed-up vehicle, typical of longer trips in free-flowing traffic;
- and a "combined" estimate that represents a combination of city driving (55%) and highway driving (45%). Estimates for all vehicles are based on laboratory testing under standardized conditions to allow for fair comparisons.

The database also provides annual fuel cost estimates, rounded to the nearest \$50, for each vehicle. The estimates are based on the assumptions that you travel 15,000 miles per year (55% under city driving conditions and 45% under highway conditions) and that fuel costs \$2.33/gallon for regular unleaded gasoline, \$2.58/gallon for mid-grade unleaded gasoline, and \$2.82/gallon for premium.

```
In [161]: ggplot(data = fuel) +
            geom_bar(aes(x = engine_cylinders )) +
            labs(x = "Number of cylinders", title = "Cylinders distributions",
                subtitle = "Vehicles data for years 1984 to 2017")
```

```
Warning message:
Removed 136 rows containing non-finite values (stat_count).
```



If you look to the far right in the above plot, you will notice some vehicles with 16 cylinders. Let us use the `dplyr::filter()` function to find those observations:

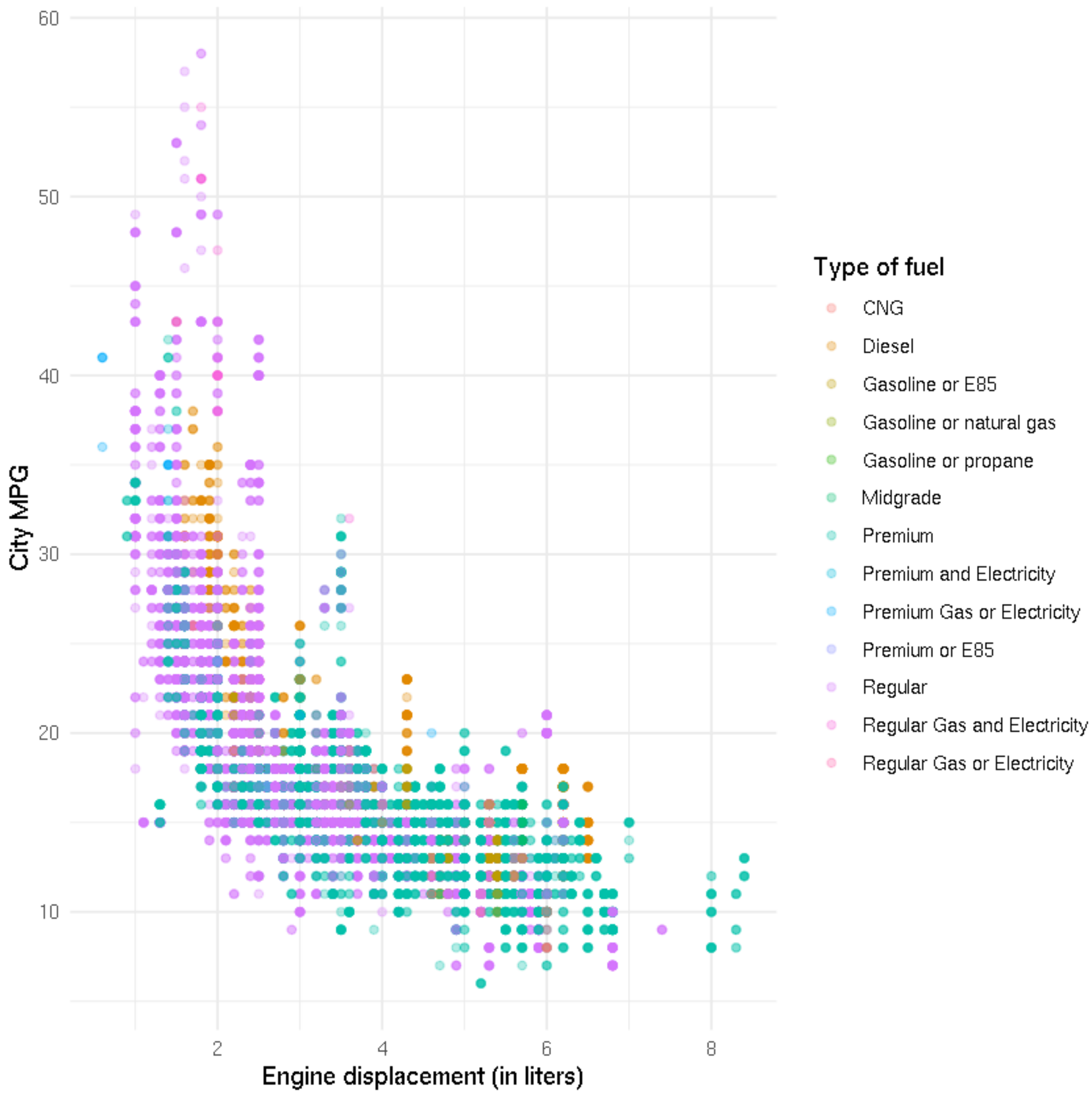
```
In [162]: fuel %>%
            filter(engine_cylinders == 16) %>%
            select(year, make, model, drive, transmission,
                   engine_cylinders, fuel_type, city_mpg_ft1)
```

year	make	model	drive	transmission	engine_cylinders	fuel_type	ci
2006	Bugatti	Veyron	4-Wheel or All-Wheel Drive	Automatic (S6)	16	Premium	8
2008	Bugatti	Veyron	4-Wheel or All-Wheel Drive	Automatic (S6)	16	Premium	8
2010	Bugatti	Veyron	All-Wheel Drive	Automatic (S7)	16	Premium	8
2011	Bugatti	Veyron	All-Wheel Drive	Automatic (S7)	16	Premium	8
2012	Bugatti	Veyron	All-Wheel Drive	Automatic (S7)	16	Premium	8
2013	Bugatti	Veyron	All-Wheel Drive	Auto(AM-S7)	16	Premium	8
2014	Bugatti	Veyron	All-Wheel Drive	Auto(AM-S7)	16	Premium	8
2015	Bugatti	Veyron	All-Wheel Drive	Auto(AM-S7)	16	Premium	8

The Bugatti Veyron: million dollars cars! Learn more about this car here.

In the set of slides for `ggplot2` we studied the relationship between the engine size (`engine_displacement`) and the fuel efficiency. Let us do something similar here:

```
In [163]: fuel %>%
            filter(engine_displacement > 0 ) %>% # make sure we have engine size values
            ggplot() +
            geom_point(aes(x = engine_displacement,
                           y = city_mpg_ft1, color = fuel_type),
                       alpha = 0.3) +
            theme_minimal() +
            labs(x = "Engine displacement (in liters)",
                 y = "City MPG",
                  color = "Type of fuel")
```



1.5 Renewable energy

Let us aggregate data by fuel type and create a new variable called to identify if the energy source is renewable or not. We can also generate an estimate of efficiency and tailpipe carbon dioxide (CO2) (`tailpipe_co2_in_grams_mile_ft1`) averages by fuel type (`fuel_type`).

A good reference to learn more about this can be found in the (US Department of Energy) *Energy Efficiency and Renewable Energy* page: https://afdc.energy.gov/fuels/

```
In [164]: fuel %>%
            group_by(fuel_type) %>%
            count()
```

fuel_type	n
CNG	60
Diesel	1014
Electricity	133
Gasoline or E85	1223
Gasoline or natural gas	20
Gasoline or propane	8
Midgrade	77
Premium	10133
Premium and Electricity	25
Premium Gas or Electricity	18
Premium or E85	122
Regular	25258
Regular Gas and Electricity	20
Regular Gas or Electricity	2

Below we use `dplyr::mutate()` to create a new column, based on the value of `fuel_type` containing the word "Electricity" or "E85". This is done with the help of the `str_detect()` function from the `stringr` package (part of the `tidyverse`)

```
In [165]: fuel %>%
            mutate(renewable = case_when(
                      str_detect(fuel_type, pattern = "Elect") ~ "Yes",
                      str_detect(fuel_type, pattern = "E85") ~ "Yes",
                      TRUE ~ "No"
                     )
                   ) %>%
            group_by(renewable) %>%
            count()
```

renewable	n
No	36570
Yes	1543

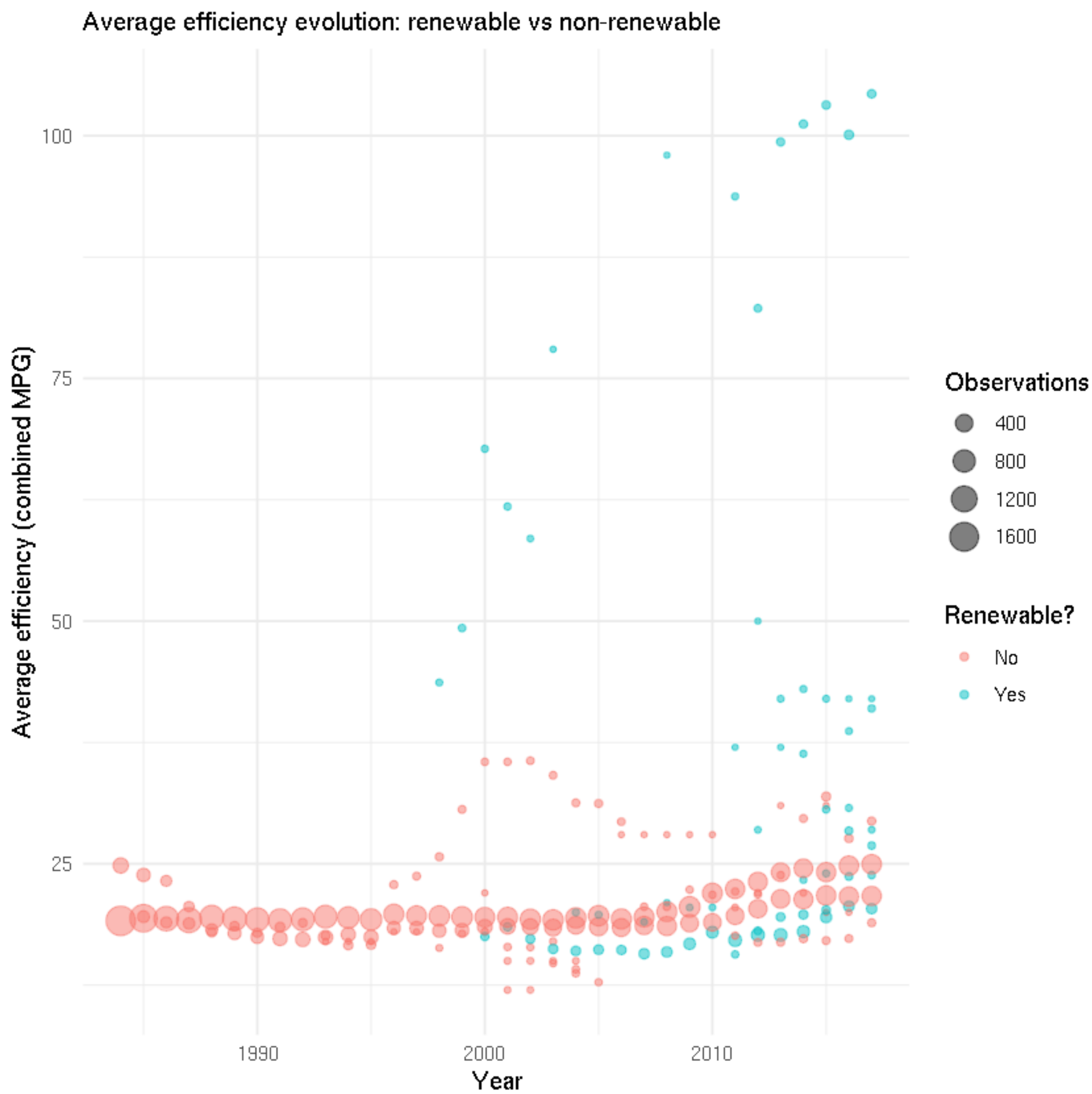
```
In [166]: # average tailpipe_co2_in_grams_mile_ft1
          fuel %>%
            mutate(renewable = case_when(
                      str_detect(fuel_type, pattern = "Elect") ~ "Yes",
                      str_detect(fuel_type, pattern = "E85") ~ "Yes",
                      TRUE ~ "No"
                     )
                   ) %>%
            group_by(renewable) %>%
            summarize(average_co2 = mean(tailpipe_co2_in_grams_mile_ft1, na.rm = T),
                      average_efficiency = mean(combined_mpg_ft1, na.rm = T))
```

renewable	average_co2	average_efficiency
No	473.3132	20.01709
Yes	459.6839	24.93195

We will create a simple plot comparing the available observations and the variable `renewable`:

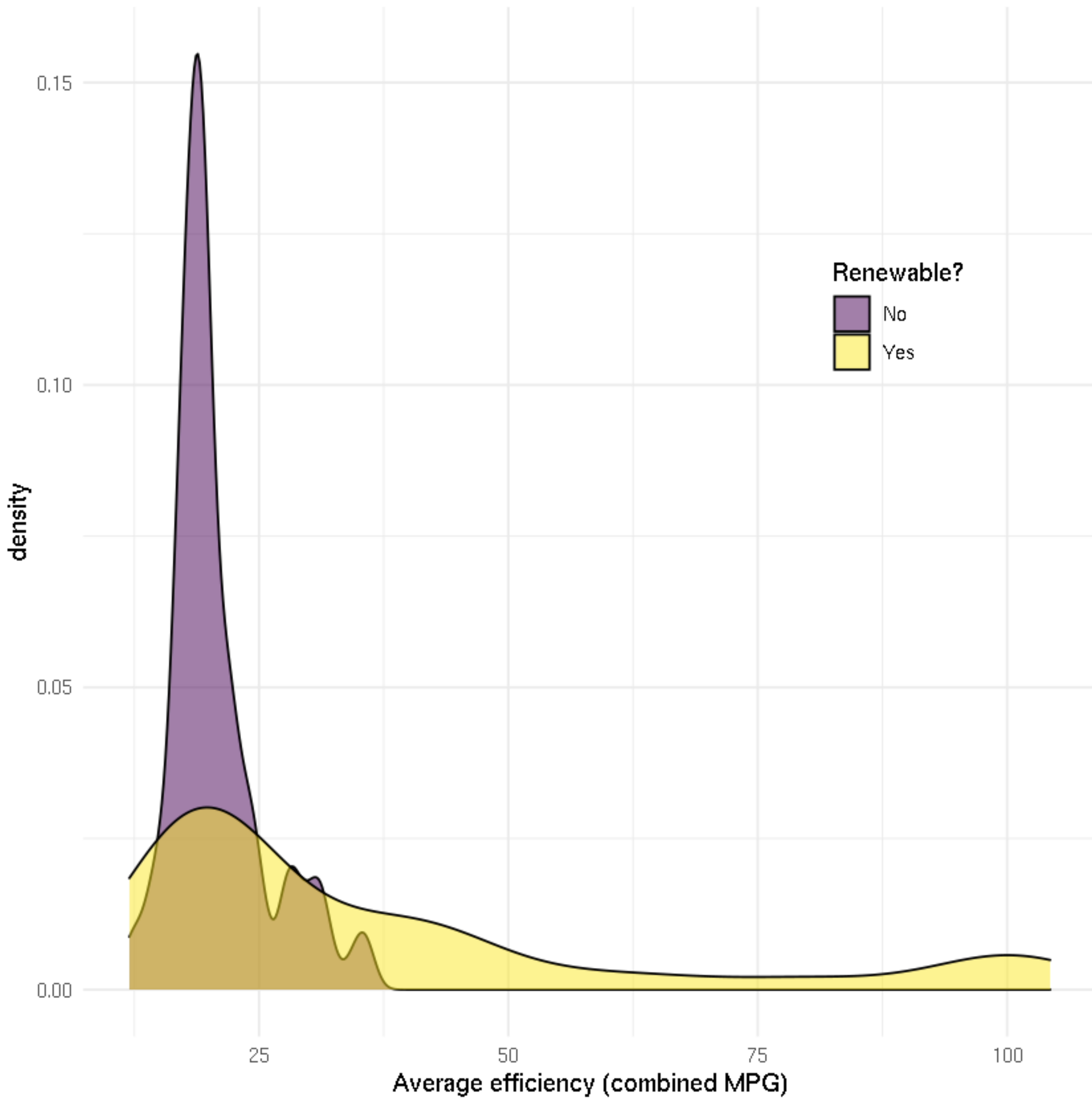
In [167]:
```
# create auxiliary dataframe with summaries
fuel_ave <- fuel %>%
  mutate(renewable = case_when(
           str_detect(fuel_type, pattern = "Elect") ~ "Yes",
           str_detect(fuel_type, pattern = "E85") ~ "Yes",
           TRUE ~ "No"
          )
        ) %>%
  group_by(fuel_type, year, renewable) %>%
  summarize(average_co2 = mean(tailpipe_co2_in_grams_mile_ft1, na.rm = T),
            average_efficiency = mean(combined_mpg_ft1, na.rm = T),
            quantity = n(),
           .groups = "drop")
```

In [168]:
```
ggplot(data = fuel_ave) +
  geom_point(aes(x = year, y = average_efficiency,
                 color = renewable, size = quantity),
             alpha = 0.5) +
  labs(x = "Year", y = "Average efficiency (combined MPG)",
       title = "Average efficiency evolution: renewable vs non-renewable",
       color = "Renewable?", size = "Observations") +
  theme_minimal()
```



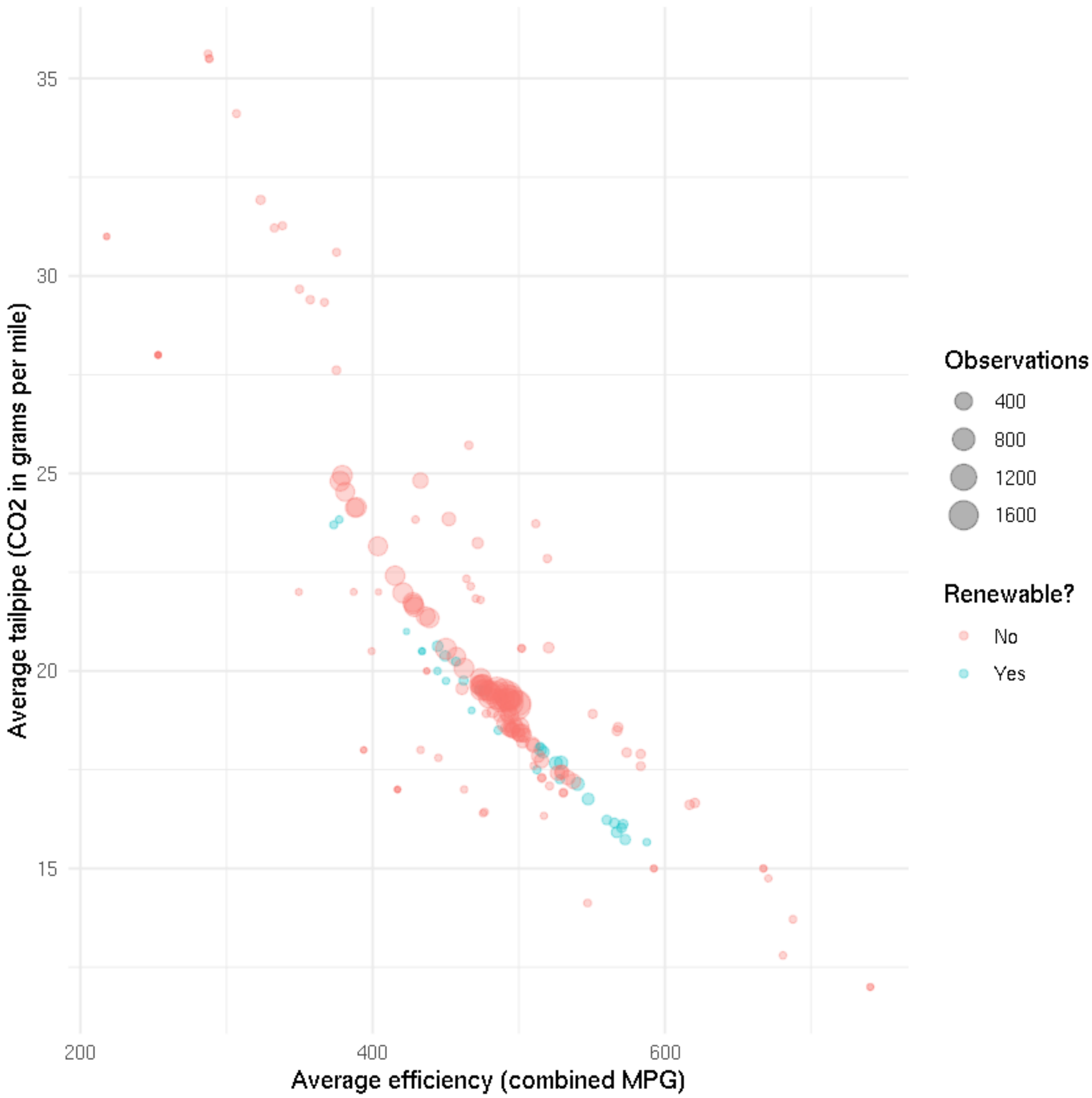
We can also check density plots:

```
In [169]: ggplot(data = fuel_ave) +
            geom_density(aes(x = average_efficiency, fill = renewable),
                         alpha = 0.5) +
            labs(x = "Average efficiency (combined MPG)",
                 fill = "Renewable?") +
            scale_fill_viridis_d() +
            theme_minimal() +
            theme(legend.position = c(0.8,0.7)) # adjust the legend position
```



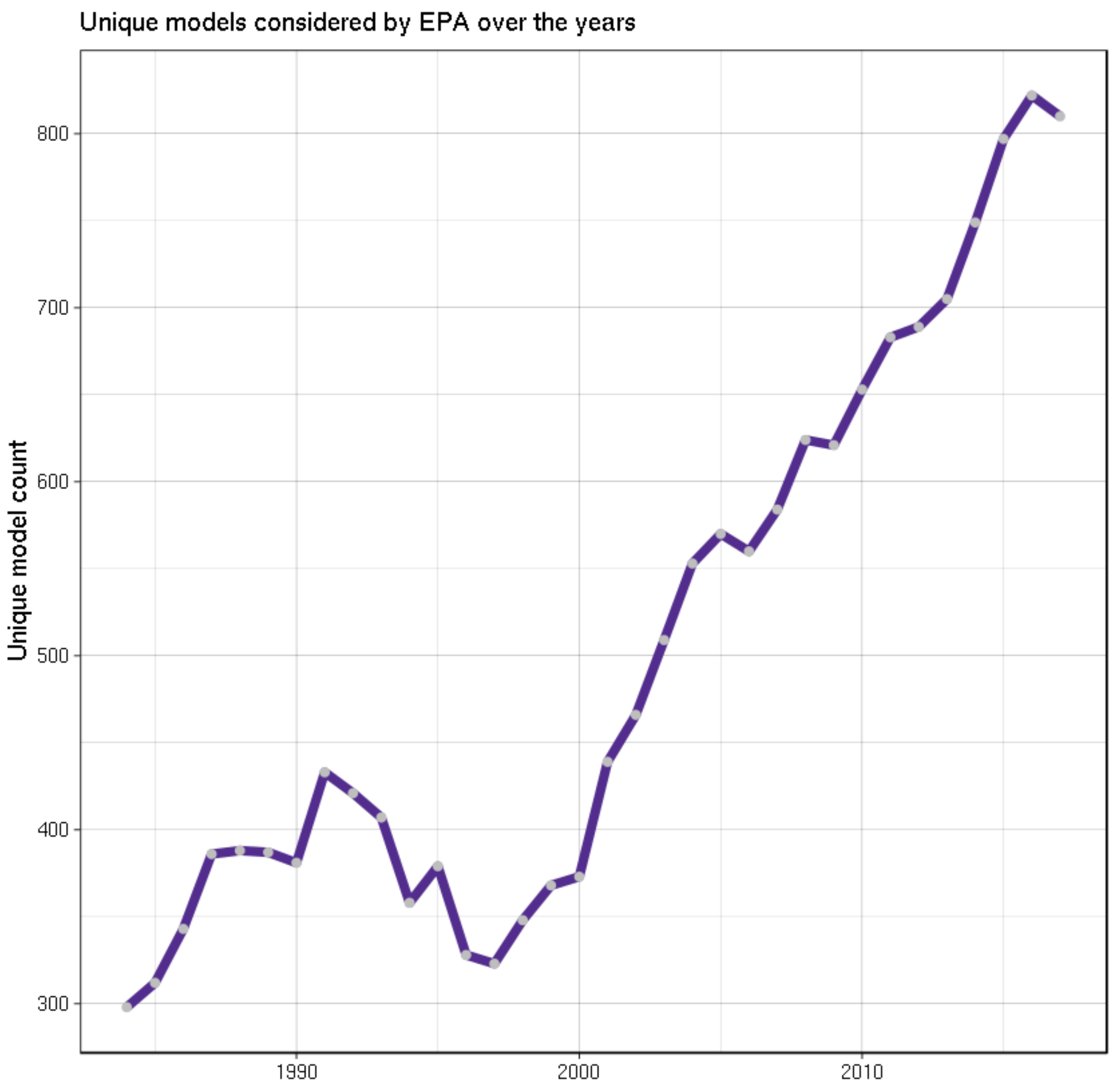
Finally, we analyze the relationship between gas emissions and fuel efficiency (ignoring electric cars):

```
In [170]: fuel_ave %>%
            filter(!str_detect(fuel_type, "Elect")) %>%
            ggplot() +
            geom_point(aes(x = average_co2, y = average_efficiency,
                           color = renewable, size = quantity),
                       alpha = 0.3) +
            labs(x = "Average efficiency (combined MPG)",
                 y = "Average tailpipe (CO2 in grams per mile)",
                 color = "Renewable?", size = "Observations") +
            theme_minimal()
```



Practice: explore the relationship between other variables. Can you characterize the trends you observe? How about the number of unique models over the years?

```
In [171]: # sample solution: notice the custom colors using HEX codes
          fuel %>%
            group_by(year) %>%
            summarize(diff_model = length(unique(model))
                     ) %>%
            ggplot(aes(x = year, y = diff_model)) +
            geom_line(color = "#532d8e", size = 2) +
            geom_point(color = "grey") +
            labs(x = "",  y = "Unique model count",
                 title = "Unique models considered by EPA over the years") +
            theme_linedraw()
```



Success! You can now return to the main page to continue learning.