Analysing used car sales data for Pakistan

Zahid Asghar, SOE, QAU

Table of contents

. 2

Why R **3**

Installing and loading packages 3

Data Overview 5

Key components of handling data **7**

Information in **pakwheels** data 7

dplyr features **9**

Data Filtering 10

Filtering with respect to two variables 12

Filtering data for Honda 12

Sort data with `arrange` 12

Change an existing variable or create a new one with mutate 13

Top bottom prices by companies for pkw data 13

How to calculate new variables 15

Ordering 16

Summarising data **16**

Summarising data by groups 16

if_else command alongwith mutate 17

Total number of cars for sale year-wise 17

Percentiles 18

Data visualization 19

Visualizing data to get data insight 20

3 Essential components of `ggplot2` 20

Scatter plot 20

Bar plot 42

Lollipop chart 44

Figure 1: used cars for sale

zasghar@qau.edu.pk zahedasghar zahidasghar.com

```
library(tidyverse)
library(gapminder)
library(kableExtra)
library(patchwork)
library(fontawesome)
library(gapminder)
library(scales)
library(httr)
clrs <- MetBrewer::met.brewer(name = "Java")
```

```
clrs_lt <- colorspace::lighten(clrs, 0.9)
knitr::opts_chunk$set(fig.retina = 3, collapse = TRUE)
options(digits = 3, width = 75)
```

Why R

There is an increasing recognition of reproducibility of research, though it has limited recognition in social sciences. The document in your hand is written in `Quarto`. `Quarto` which can be used for pdf, html, word, PowerPoint/Slidy/Beamer Presentations, Webpages, LaTex and many others. Besides learning basics of R-coding, another objective of this workshop is understanding the importance of reproducibility. Building this new habit of reproducible work at times maybe little challenging occasionally. Getting rid of culture of copying and pasting, and sparing this time for doing data analysis and research is one of the objectives of this or coming workshops. Purpose is to help you to get away from this tedious activity so that you can spend more time **doing science**.

Installing and loading packages

The first thing we need to do is install and then load the `tidyverse` set of R packages to provide us with lots of extra functionality. You only need to install this once: once it's installed we can simply load it into the workspace using the library() function each time we open a new R session.

Understanding data sets requires many hours/days or in some cases weeks.There are many commercially available software but open source community based software have now dominated and R is one of these. R makes data understanding process as easy as possible through the `dplyr` package. It is one of the easiest solution for code-based data analysis. We will learn in this training how to do it. In case, you need more information, you can watch my videos.

I have discussed the Gapminder dataset in my videos, you can watch those videos. `gapminder` package is available through CRAN, so make sure to install it. Here's how to load in all required packages:

```
library(tidyverse)
library(knitr)
library(kableExtra)
#install.packages("gapminder")
library(hrbrthemes)
library(viridis)
library(kableExtra)
```

```r
options(knitr.table.format = "html")
library(plotly)
library(gridExtra)
library(ggrepel)
```

**The dataset to be used is pakwheels data obtained from kaggle and you can download data from the link data download or from my github repository. We read `data` as follows

```r
#pakwheels_11Jul2020 <- read_csv("C:/Users/92300/Downloads/archive/pakwheels-11Jul2020.csv
#pakwheels<-saveRDS(pakwheels_11Jul2020,file = "pakwheels.rds")

pakwheels <- readRDS("D:/RepTemplates/data_analytics/pakwheels.rds")

pakwheels
## # A tibble: 56,186 x 16
##    `Ad No` Name       Price Model~1 Locat~2 Mileage Regis~3 Engin~4 Engin~5
##      <dbl> <chr>      <chr>   <dbl> <chr>     <dbl> <chr>   <chr>   <chr>
##  1 4096758 Toyota Vi~ 2385~    2017 G- 8, ~    9869 Un-Reg~ Petrol  1000 cc
##  2 4168305 Toyota Co~ 1110~    2019 Peshaw~   11111 Islama~ Petrol  1300 cc
##  3 4168298 Suzuki Al~ 1530~    2019 Akora ~   17500 Un-Reg~ Petrol  660 cc
##  4 4168307 Suzuki Al~ 1650~    2019 Abdull~    9600 Lahore  Petrol  660 cc
##  5 4168306 Toyota Co~ 1435~    2010 9th Av~  120000 Islama~ Petrol  1300 cc
##  6 4168303 Honda Civ~ 3850~    2017 Peshaw~   22000 Islama~ Petrol  1500 cc
##  7 4168304 Suzuki Wa~ 1440~    2017 Gulber~   31000 Lahore  Petrol  1000 cc
##  8 4168309 Mitsubish~ 1425~    2012 Askari~  101000 Lahore  Petrol  1000 cc
##  9 4168310 Toyota Pr~ 2650~    1998 Sargod~  110000 Rawalp~ Diesel  3000 cc
## 10 4168311 Honda Civ~ 3350~    2017 Air Av~   60000 Lahore  Petrol  1800 cc
## # ... with 56,176 more rows, 7 more variables: Transmission <chr>,
## #   Color <chr>, Assembly <chr>, `Body Type` <chr>, Features <chr>,
## #   `Last Updated` <chr>, URL <chr>, and abbreviated variable names
## #   1: `Model Year`, 2: Location, 3: `Registered City`, 4: `Engine Type`,
## #   5: `Engine Capacity`
```

First few rows of data are displayed here and there `56186` observations in total. We will start from scratch and end up a sophisticated analysis. Firs step in dealing with data is to clean up the data and bringing it in workable format. It is said that every `tidy` data is alike but every messy data is messy in its own way. So lets see whether this data are in neat and clean format.

Data Overview

Pipe operator %>% or `|>` play very nicely with dplyr and make our code very easy to understand. For this lets have an overview of data for which one can use `glimpse()` or `str()` for structure of data and to view entire spreadsheet use `View()`.`View` command opens data in new worksheet while glimpse lists nature of variables (numeric/character/factor…) and total number of rows and columns.To see first 6 and last 6 observations use `head()` and `tail()` ,respectively.

```
pakwheels|>glimpse()
## Rows: 56,186
## Columns: 16
## $ `Ad No`           <dbl> 4096758, 4168305, 4168298, 4168307, 4168306, 41~
## $ Name              <chr> "Toyota Vitz F 1.0 2017", "Toyota Corolla GLi A~
## $ Price             <chr> "2385000.0", "111000.00000000001", "1530000.0",~
## $ `Model Year`      <dbl> 2017, 2019, 2019, 2019, 2010, 2017, 2017, 2012,~
## $ Location          <chr> "G- 8, Islamabad Islamabad", "Peshawar KPK", "A~
## $ Mileage           <dbl> 9869, 11111, 17500, 9600, 120000, 22000, 31000,~
## $ `Registered City` <chr> "Un-Registered", "Islamabad", "Un-Registered", ~
## $ `Engine Type`     <chr> "Petrol", "Petrol", "Petrol", "Petrol", "Petrol~
## $ `Engine Capacity` <chr> "1000 cc", "1300 cc", "660 cc", "660 cc", "1300~
## $ Transmission      <chr> "Automatic", "Automatic", "Automatic", "Manual"~
## $ Color             <chr> "Silver", "White", "White", "White", "Black", "~
## $ Assembly          <chr> "Imported", "Local", "Local", "Local", "Local",~
## $ `Body Type`       <chr> "Hatchback", "Sedan", "Hatchback", "Hatchback",~
## $ Features          <chr> "ABS, AM/FM Radio, Air Bags, Air Conditioning, ~
## $ `Last Updated`    <chr> "Jul 11, 2020", "Jul 12, 2020", "Jul 12, 2020",~
## $ URL               <chr> "https://www.pakwheels.com/used-cars/toyota-vit~
head(pakwheels)
## # A tibble: 6 x 16
##   `Ad No` Name        Price Model~1 Locat~2 Mileage Regis~3 Engin~4 Engin~5
##     <dbl> <chr>       <chr>   <dbl> <chr>     <dbl> <chr>   <chr>   <chr>
## 1 4096758 Toyota Vit~ 2385~    2017 G- 8, ~    9869 Un-Reg~ Petrol  1000 cc
## 2 4168305 Toyota Cor~ 1110~    2019 Peshaw~   11111 Islama~ Petrol  1300 cc
## 3 4168298 Suzuki Alt~ 1530~    2019 Akora ~   17500 Un-Reg~ Petrol  660 cc
## 4 4168307 Suzuki Alt~ 1650~    2019 Abdull~    9600 Lahore  Petrol  660 cc
## 5 4168306 Toyota Cor~ 1435~    2010 9th Av~  120000 Islama~ Petrol  1300 cc
## 6 4168303 Honda Civi~ 3850~    2017 Peshaw~   22000 Islama~ Petrol  1500 cc
## # ... with 7 more variables: Transmission <chr>, Color <chr>,
## #   Assembly <chr>, `Body Type` <chr>, Features <chr>,
## #   `Last Updated` <chr>, URL <chr>, and abbreviated variable names
```

```
## #   1: `Model Year`, 2: Location, 3: `Registered City`, 4: `Engine Type`,
## #   5: `Engine Capacity`
tail(pakwheels)
## # A tibble: 6 x 16
##   `Ad No` Name        Price Model~1 Locat~2 Mileage Regis~3 Engin~4 Engin~5
##     <dbl> <chr>       <chr>   <dbl> <chr>     <dbl> <chr>   <chr>   <chr>
## 1 3339018 Nissan Kix~ 1900~    2012 Faisal~   42000 Faisal~ Petrol  660 cc
## 2 3349017 Honda Civi~ 3250~    2015 Lahore~  125000 Lahore  Petrol  1800 cc
## 3 3722322 Toyota Pri~ 4000~    2015 Peshaw~   35000 Lahore  Hybrid  1800 cc
## 4 3748215 Toyota Aqu~ 3000~    2016 Gujran~   60000 Lahore  Petrol  1500 cc
## 5 3785520 Honda Veze~ Call~    2015 Multan~   45000 Un-Reg~ Hybrid  1500 cc
## 6 3806951 Toyota Cor~ 2250~    2015 Gujran~   77000 Gujran~ Petrol  1300 cc
## # ... with 7 more variables: Transmission <chr>, Color <chr>,
## #   Assembly <chr>, `Body Type` <chr>, Features <chr>,
## #   `Last Updated` <chr>, URL <chr>, and abbreviated variable names
## #   1: `Model Year`, 2: Location, 3: `Registered City`, 4: `Engine Type`,
## #   5: `Engine Capacity`
```

One can observe that `Price` and `Engine Capacity` are character variables while we know Price is numeric variable and should fall in cateogry of `dbl` used in `R` for numeric variable. Similarly `Engine Capacity` can be made numeric if we remove cc from it. In the follow `chunk` I am going to convert these two variables as `numeric`

```
pakwheels$price<-as.numeric(pakwheels$Price) ## To convert price as numeric, R-base comman
pakwheels$hp <- as.numeric(gsub("([A-Za-z]+).*", "", pakwheels$`Engine Capacity`)) ## To t
pakwheels$company <- gsub("([A-Za-z]+).*", "\\1", pakwheels$Name)  ## To take first word f
```

So far so good. We have converted now three new variables `price`, `hp` and `company`. Variable names should preferably not have space between their names. Better use one word or use hyphen or underscore. Lets have a look at data again.

```
pakwheels|>glimpse()
## Rows: 56,186
## Columns: 19
## $ `Ad No`          <dbl> 4096758, 4168305, 4168298, 4168307, 4168306, 41~
## $ Name             <chr> "Toyota Vitz F 1.0 2017", "Toyota Corolla GLi A~
## $ Price            <chr> "2385000.0", "111000.00000000001", "1530000.0",~
## $ `Model Year`     <dbl> 2017, 2019, 2019, 2019, 2010, 2017, 2017, 2012,~
## $ Location         <chr> "G- 8, Islamabad Islamabad", "Peshawar KPK", "A~
## $ Mileage          <dbl> 9869, 11111, 17500, 9600, 120000, 22000, 31000,~
## $ `Registered City` <chr> "Un-Registered", "Islamabad", "Un-Registered", ~
```

```
## $ `Engine Type`     <chr> "Petrol", "Petrol", "Petrol", "Petrol", "Petrol~
## $ `Engine Capacity` <chr> "1000 cc", "1300 cc", "660 cc", "660 cc", "1300~
## $ Transmission      <chr> "Automatic", "Automatic", "Automatic", "Manual"~
## $ Color             <chr> "Silver", "White", "White", "White", "Black", "~
## $ Assembly          <chr> "Imported", "Local", "Local", "Local", "Local",~
## $ `Body Type`       <chr> "Hatchback", "Sedan", "Hatchback", "Hatchback",~
## $ Features          <chr> "ABS, AM/FM Radio, Air Bags, Air Conditioning, ~
## $ `Last Updated`    <chr> "Jul 11, 2020", "Jul 12, 2020", "Jul 12, 2020",~
## $ URL               <chr> "https://www.pakwheels.com/used-cars/toyota-vit~
## $ price             <dbl> 2385000, 111000, 1530000, 1650000, 1435000, 385~
## $ hp                <dbl> 1000, 1300, 660, 660, 1300, 1500, 1000, 1000, 3~
## $ company           <chr> "Toyota", "Toyota", "Suzuki", "Suzuki", "Toyota~
```

So now we have 19 variables and `price` and `hp` are numeric variables.

Key components of handling data

- View, glimpse, structure
- head, tail
- Column Selection
- Data Filtering
- Data Ordering
- Creating Derived Columns
- Calculating Summary Statistics
- Grouping

Information in pakwheels data

To rename a variable, there are various ways. I am using a command `rename(new_va=old_var)`.

```
pakwheels<-pakwheels|>rename(year=`Model Year`)
#kbl()#|>
 # kable_styling(bootstrap_options = "striped", full_width = F)
#View(pakwheels)
pakwheels|>count(year)|>arrange(desc(year))# How many cars by year model are listed for sa
# A tibble: 30 x 2
    year     n
   <dbl> <int>
 1  2019  3166
 2  2018  3820
```

```
 3  2017  4628
 4  2016  4484
 5  2015  4900
 6  2014  4430
 7  2013  3342
 8  2012  3146
 9  2011  2621
10  2010  2117
# ... with 20 more rows
```

As there are a large number of observations and it is not possible to find out through scrolling how many missing observations in the data. We use a command `na.omit()` to find out how many missing observations and give a **`new name`** to our data without losing our original data as follows: glimpse(gapminder) # We see that there are 1704 rows for 6 columns and also tells nature of variable #View(gapminder) # This opens up full data in a new window

```
pkw<- pakwheels|>na.omit()
pkw|>glimpse()
## Rows: 44,917
## Columns: 19
## $ `Ad No`           <dbl> 4096758, 4168305, 4168298, 4168307, 4168306, 41~
## $ Name              <chr> "Toyota Vitz F 1.0 2017", "Toyota Corolla GLi A~
## $ Price             <chr> "2385000.0", "111000.00000000001", "1530000.0",~
## $ year              <dbl> 2017, 2019, 2019, 2019, 2010, 2017, 2017, 2012,~
## $ Location          <chr> "G- 8, Islamabad Islamabad", "Peshawar KPK", "A~
## $ Mileage           <dbl> 9869, 11111, 17500, 9600, 120000, 22000, 31000,~
## $ `Registered City` <chr> "Un-Registered", "Islamabad", "Un-Registered", ~
## $ `Engine Type`     <chr> "Petrol", "Petrol", "Petrol", "Petrol", "Petrol~
## $ `Engine Capacity` <chr> "1000 cc", "1300 cc", "660 cc", "660 cc", "1300~
## $ Transmission      <chr> "Automatic", "Automatic", "Automatic", "Manual"~
## $ Color             <chr> "Silver", "White", "White", "White", "Black", "~
## $ Assembly          <chr> "Imported", "Local", "Local", "Local", "Local",~
## $ `Body Type`       <chr> "Hatchback", "Sedan", "Hatchback", "Hatchback",~
## $ Features          <chr> "ABS, AM/FM Radio, Air Bags, Air Conditioning, ~
## $ `Last Updated`    <chr> "Jul 11, 2020", "Jul 12, 2020", "Jul 12, 2020",~
## $ URL               <chr> "https://www.pakwheels.com/used-cars/toyota-vit~
## $ price             <dbl> 2385000, 111000, 1530000, 1650000, 1435000, 385~
## $ hp                <dbl> 1000, 1300, 660, 660, 1300, 1500, 1000, 1000, 3~
## $ company           <chr> "Toyota", "Toyota", "Suzuki", "Suzuki", "Toyota~
```

So now we have 44,917 observations aftere liminating missing observations.

dplyr features

One of the most widely used `package` in R for data wrangling is `dplyr` which is under `tidyverse` or you can simply recall `dplyr`.

1. `filter()` to keep selected observations
2. `select()` to keep selected variables
3. `arrange()` to reorder observations by a value
4. `mutate()` to create new variables
5. `summarize()` to create summary statistics
6. `group_by()` for performing operations by group

Now I shall mention some of the powerful but very simple to use features of dplyr. ## Column Selection

More often than not, you don't need all columns of a data set for your analysis. For example `PDHS` files have more than 5000 columns in some files and maybe 40 or 50 or even fewer than that are needed for your analysis. `Select()` function of R's dplyr is used to select columns of your interest Three selected columns are selected as follows. You can give new name to this data.

```
pkw %>% select(price, hp, company)
## # A tibble: 44,917 x 3
##      price    hp company
##      <dbl> <dbl> <chr>
##  1 2385000  1000 Toyota
##  2  111000  1300 Toyota
##  3 1530000   660 Suzuki
##  4 1650000   660 Suzuki
##  5 1435000  1300 Toyota
##  6 3850000  1500 Honda
##  7 1440000  1000 Suzuki
##  8 1425000  1000 Mitsubishi
##  9 2650000  3000 Toyota
## 10 3350000  1800 Honda
## # ... with 44,907 more rows
```

In case you want to select most of the variables and drop one or two, you may proceed as follows

```
pkw |> select(-URL)
## # A tibble: 44,917 x 18
##    `Ad No` Name         Price  year Locat~1 Mileage Regis~2 Engin~3 Engin~4
```

```
##      <dbl> <chr>        <chr> <dbl> <chr>     <dbl> <chr>   <chr>   <chr>
##  1 4096758 Toyota Vitz~ 2385~  2017 G- 8, ~  9869 Un-Reg~ Petrol  1000 cc
##  2 4168305 Toyota Coro~ 1110~  2019 Peshaw~ 11111 Islama~ Petrol  1300 cc
##  3 4168298 Suzuki Alto~ 1530~  2019 Akora ~ 17500 Un-Reg~ Petrol  660 cc
##  4 4168307 Suzuki Alto~ 1650~  2019 Abdull~  9600 Lahore  Petrol  660 cc
##  5 4168306 Toyota Coro~ 1435~  2010 9th Av~ 120000 Islama~ Petrol  1300 cc
##  6 4168303 Honda Civic~ 3850~  2017 Peshaw~ 22000 Islama~ Petrol  1500 cc
##  7 4168304 Suzuki Wago~ 1440~  2017 Gulber~ 31000 Lahore  Petrol  1000 cc
##  8 4168309 Mitsubishi ~ 1425~  2012 Askari~ 101000 Lahore  Petrol  1000 cc
##  9 4168310 Toyota Prad~ 2650~  1998 Sargod~ 110000 Rawalp~ Diesel  3000 cc
## 10 4168311 Honda Civic~ 3350~  2017 Air Av~ 60000 Lahore  Petrol  1800 cc
## # ... with 44,907 more rows, 9 more variables: Transmission <chr>,
## #   Color <chr>, Assembly <chr>, `Body Type` <chr>, Features <chr>,
## #   `Last Updated` <chr>, price <dbl>, hp <dbl>, company <chr>, and
## #   abbreviated variable names 1: Location, 2: `Registered City`,
## #   3: `Engine Type`, 4: `Engine Capacity`
```

So `url` column is now not shown above.

Data Filtering

Filtering is another very important task one has to do in one's analysis. Sometimes, one has to select sale related to a particular city or agent or quarter. Here is how one uses `filter()` command for data with a condition. We are using here command only to select data for year 2007 for all the countries. I am going to explain `filter` variable of dplyr. `filter` is used only to select rows for a given condition. I am going to select data only for year 2007.

```
pkw_szk<- pkw %>% filter(company=="Suzuki")|> select(`year`, price, hp,Color,Assembly, Tra
pkw_szk|>glimpse()
## Rows: 14,199
## Columns: 9
## $ year              <dbl> 2019, 2019, 2017, 2012, 2016, 2018, 2000, 2016,~
## $ price             <dbl> 1530000, 1650000, 1440000, 920000, 1625000, 840~
## $ hp                <dbl> 660, 660, 1000, 1000, 660, 800, 1000, 660, 660,~
## $ Color             <chr> "White", "White", "White", "Grey", "Brown", "Wh~
## $ Assembly          <chr> "Local", "Local", "Local", "Local", "Imported",~
## $ Transmission      <chr> "Automatic", "Manual", "Manual", "Manual", "Aut~
## $ `Engine Type`     <chr> "Petrol", "Petrol", "Petrol", "Petrol", "Petrol~
## $ Mileage           <dbl> 17500, 9600, 31000, 83000, 45000, 55000, 90000,~
## $ `Registered City` <chr> "Un-Registered", "Lahore", "Lahore", "Lahore", ~
kbl(pkw_szk[1:10,])%>%kable_styling(fixed_thead=T)
```

year	price	hp	Color	Assembly	Transmission	Engine Type	Mileage	Registered City
2019	1530000	660	White	Local	Automatic	Petrol	17500	Un-Registered
2019	1650000	660	White	Local	Manual	Petrol	9600	Lahore
2017	1440000	1000	White	Local	Manual	Petrol	31000	Lahore
2012	920000	1000	Grey	Local	Manual	Petrol	83000	Lahore
2016	1625000	660	Brown	Imported	Automatic	Petrol	45000	Un-Registered
2018	840000	800	White	Local	Manual	Petrol	55000	Islamabad
2000	400000	1000	Assembly	Local	Manual	Petrol	90000	Multan
2016	1445000	660	White	Imported	Automatic	Petrol	65000	Lahore
2013	1495000	660	White	Local	Automatic	Petrol	85000	Islamabad
2016	1495000	660	Burgundy	Local	Automatic	Petrol	63000	Karachi

Compared to previous one, `pkw_szk` is showing data only for There are 14,199 cars. The tibble (name used for data in tidyverse form) `pkw` is being piped into the function `filter()`. The argument `company == "Suzuki"` tells filter() that it should find all the rows such that the logical condition `year == "Suzuki` is TRUE.

`Have we accidently deleted all other rows? Answer is no.`

Nope: we haven't made any changes to gapminder at all. If you don't believe me try entering `pkw` at the console. All that this command does is display a subset of gapminder. If we wanted to store the result of running this command, we'd need to assign it to a variable, for example if you are not sure, lets type

```
pkw |> filter(company=="Suzuki")
## # A tibble: 14,199 x 19
##    `Ad No` Name         Price  year Locat~1 Mileage Regis~2 Engin~3 Engin~4
##      <dbl> <chr>        <chr> <dbl> <chr>     <dbl> <chr>   <chr>   <chr>
##  1 4168298 Suzuki Alto~ 1530~  2019 Akora ~   17500 Un-Reg~ Petrol  660 cc
##  2 4168307 Suzuki Alto~ 1650~  2019 Abdull~    9600 Lahore  Petrol  660 cc
##  3 4168304 Suzuki Wago~ 1440~  2017 Gulber~   31000 Lahore  Petrol  1000 cc
##  4 4168320 Suzuki Cult~ 9199~  2012 Bismil~   83000 Lahore  Petrol  1000 cc
##  5 4168327 Suzuki Wago~ 1625~  2016 Sui No~   45000 Un-Reg~ Petrol  660 cc
##  6 4168332 Suzuki Mehr~ 8400~  2018 Sargod~   55000 Islama~ Petrol  800 cc
##  7 4168333 Suzuki Khyb~ 4000~  2000 MDA Ch~   90000 Multan  Petrol  1000 cc
##  8 4134011 Suzuki Alto~ 1445~  2016 Johar ~   65000 Lahore  Petrol  660 cc
##  9 4168228 Suzuki Wago~ 1495~  2013 Hospit~   85000 Islama~ Petrol  660 cc
## 10 4168257 Suzuki Alto~ 1495~  2016 Karach~   63000 Karachi Petrol  660 cc
## # ... with 14,189 more rows, 10 more variables: Transmission <chr>,
## #   Color <chr>, Assembly <chr>, `Body Type` <chr>, Features <chr>,
## #   `Last Updated` <chr>, URL <chr>, price <dbl>, hp <dbl>, company <chr>,
## #   and abbreviated variable names 1: Location, 2: `Registered City`,
```

```
## #   3: `Engine Type`, 4: `Engine Capacity`
```

Filtering with respect to two variables

One can apply multiple `filters`

```
pkw %>% filter(year=="2019",company=="Toyota") ## year 2019 and company is Toyota
```

Now we are selecting multiple years for `Toyota`.

```
pkw %>% filter(year %in%c(2015,2016,2017,2018,2019), company=="Toyota")
```

Filtering data for Honda

```
pkw_hnd<-pkw |> filter(company=="Honda")
```

Sort data with `arrange`

Sort data with arrange Suppose we wanted to sort `pkw` data for `Honda` by `Color`. To do this we can use the arrange command along with the pipe `|>` as follows:

```
pkw |>filter(company=="Honda") |> count(Color)|>arrange(n)## This sorts in asceding order
## # A tibble: 23 x 2
##    Color         n
##    <chr>     <int>
##  1 Magenta       1
##  2 Orange        3
##  3 Yellow       10
##  4 Navy         17
##  5 Turquoise    17
##  6 Pink         20
##  7 Purple       24
##  8 Indigo       25
##  9 Unlisted     29
## 10 Beige        42
## # ... with 13 more rows
```

Descending order requires `arrange(desc())` comand

```
pkw |>filter(company=="Honda") |> count(Color)|>arrange(desc(n))## This sorts in asceding
## # A tibble: 23 x 2
##    Color        n
##    <chr>    <int>
##  1 White     3425
##  2 Black     1848
##  3 Silver    1603
##  4 Grey       992
##  5 Blue       315
##  6 Assembly   271
##  7 Maroon     162
##  8 Red        150
##  9 Gold       140
## 10 Green      128
## # ... with 13 more rows
```

The logic is very similar to what we saw above for filter. Here, I use another important function `arrange`. The argument `count(Color)` tells `arrange()` that we want to sort by Color for Honda company cars. Note that by default `arrange()` sorts in ascending order. If we want to sort in descending order, we use the function `desc()`.

Change an existing variable or create a new one with mutate

It's a little hard to read the column `peice` in `pkw` data since there are so many digits. Suppose that, instead of `price in Rs.`, we wanted to display `price` in millions of rupees. This requires us to divide `price` by 1000000, which we can do using the function `mutate()` from dplyr as follows:

```
pkw<- pkw %>% mutate(price_m=price/1000000)
```

Top bottom prices by companies for pkw data

What are the five lowest and highest car prices for 1300 cc for model year 2017?

```
pkw |> filter(year=="2017",hp==1300) ## year 2017 and engine capacity 1300
## # A tibble: 758 x 20
##    `Ad No` Name         Price  year Locat~1 Mileage Regis~2 Engin~3 Engin~4
##      <dbl> <chr>        <chr> <dbl> <chr>     <dbl> <chr>   <chr>   <chr>
##  1 4168312 Honda City ~ 1989~  2017 Lahore~   75000 Lahore  Petrol  1300 cc
```

```
##  2 4168141 Toyota Coro~ 2595~  2017 Islama~   41000 Islama~ Petrol  1300 cc
##  3 4168148 Toyota Coro~ 2480~  2017 North ~   38000 Karachi Petrol  1300 cc
##  4 4168061 Honda City ~ 2145~  2017 Bahria~   70000 Islama~ Petrol  1300 cc
##  5 4145330 Toyota Coro~ 2540~  2017 Karach~   36000 Karachi Petrol  1300 cc
##  6 4145974 Suzuki Swif~ 1570~  2017 Garden~   56000 Islama~ Petrol  1300 cc
##  7 4167831 Toyota Coro~ 2400~  2017 Bahria~   50000 Islama~ Petrol  1300 cc
##  8 3911337 Suzuki Swif~ 1575~  2017 A.F.O.~   34000 Karachi Petrol  1300 cc
##  9 4167800 Honda City ~ 2050~  2017 Taj Pu~   85000 Lahore  Petrol  1300 cc
## 10 4167728 Honda City ~ 2025~  2017 Rawalp~   80000 Islama~ Petrol  1300 cc
## # ... with 748 more rows, 11 more variables: Transmission <chr>,
## #   Color <chr>, Assembly <chr>, `Body Type` <chr>, Features <chr>,
## #   `Last Updated` <chr>, URL <chr>, price <dbl>, hp <dbl>, company <chr>,
## #   price_m <dbl>, and abbreviated variable names 1: Location,
## #   2: `Registered City`, 3: `Engine Type`, 4: `Engine Capacity`
pkw|>glimpse()
## Rows: 44,917
## Columns: 20
## $ `Ad No`           <dbl> 4096758, 4168305, 4168298, 4168307, 4168306, 41~
## $ Name              <chr> "Toyota Vitz F 1.0 2017", "Toyota Corolla GLi A~
## $ Price             <chr> "2385000.0", "111000.00000000001", "1530000.0",~
## $ year              <dbl> 2017, 2019, 2019, 2019, 2010, 2017, 2017, 2012,~
## $ Location          <chr> "G- 8, Islamabad Islamabad", "Peshawar KPK", "A~
## $ Mileage           <dbl> 9869, 11111, 17500, 9600, 120000, 22000, 31000,~
## $ `Registered City` <chr> "Un-Registered", "Islamabad", "Un-Registered", ~
## $ `Engine Type`     <chr> "Petrol", "Petrol", "Petrol", "Petrol", "Petrol~
## $ `Engine Capacity` <chr> "1000 cc", "1300 cc", "660 cc", "660 cc", "1300~
## $ Transmission      <chr> "Automatic", "Automatic", "Automatic", "Manual"~
## $ Color             <chr> "Silver", "White", "White", "White", "Black", "~
## $ Assembly          <chr> "Imported", "Local", "Local", "Local", "Local",~
## $ `Body Type`       <chr> "Hatchback", "Sedan", "Hatchback", "Hatchback",~
## $ Features          <chr> "ABS, AM/FM Radio, Air Bags, Air Conditioning, ~
## $ `Last Updated`    <chr> "Jul 11, 2020", "Jul 12, 2020", "Jul 12, 2020",~
## $ URL               <chr> "https://www.pakwheels.com/used-cars/toyota-vit~
## $ price             <dbl> 2385000, 111000, 1530000, 1650000, 1435000, 385~
## $ hp                <dbl> 1000, 1300, 660, 660, 1300, 1500, 1000, 1000, 3~
## $ company           <chr> "Toyota", "Toyota", "Suzuki", "Suzuki", "Toyota~
## $ price_m           <dbl> 2.385, 0.111, 1.530, 1.650, 1.435, 3.850, 1.440~
pkw |> filter(year=="2017"&hp==1300)|>select(company, price, Assembly,Mileage,Color,Transm
## # A tibble: 10 x 6
##    company   price Assembly Mileage Color  Transmission
##    <chr>     <dbl> <chr>      <dbl> <chr>  <chr>
```

```
##  1 Toyota  3220000 Local      40000 Grey   Automatic
##  2 Toyota  3150000 Local      40000 White  Manual
##  3 Toyota  3150000 Local      40000 White  Manual
##  4 Suzuki  3000000 Local      35000 White  Manual
##  5 Toyota  2950000 Local      50000 Silver Automatic
##  6 Toyota  2890000 Local      40000 Silver Automatic
##  7 Toyota  2850000 Local      33000 Bronze Automatic
##  8 Toyota  2800000 Local      14000 Grey   Automatic
##  9 Toyota  2800000 Local      15000 Silver Automatic
## 10 Toyota  2800000 Local      20000 Silver Manual
```

Bottom 10 are reported as follows

```
pkw |> filter(year=="2017"&hp==1300)|>select(company, price, Assembly,Mileage,Color,Transm
## # A tibble: 10 x 6
##    company   price Assembly Mileage Color  Transmission
##    <chr>     <dbl> <chr>      <dbl> <chr>  <chr>
##  1 FAW     1220000 Imported   30000 Gold   Manual
##  2 FAW     1215000 Local      91000 White  Manual
##  3 FAW     1200000 Local      57000 Gold   Manual
##  4 FAW     1200000 Imported   50000 White  Manual
##  5 FAW     1200000 Imported   54000 Black  Manual
##  6 FAW     1100000 Imported   24000 White  Manual
##  7 Suzuki  1100000 Local      67000 White  Manual
##  8 FAW     1080000 Local      82000 Silver Manual
##  9 FAW     1075000 Imported   90000 Silver Manual
## 10 Toyota   800000 Local      62800 Grey   Automatic
```

How to calculate new variables

As mentioned above, `mutate` is used to calculate new variable. Here,we calculate a new variable `price_mill` (price in million of Rs.) and then `arranged` data and selected `top_n(10)` cars `transmute()` keeps only the derived column. Let's use it in the example from above:

```
pkw %>% filter(year==2017) %>%
  transmute(price_million=price/1000000) %>%
  arrange(desc(price_million)) %>%
  top_n(10, price_million)
## # A tibble: 11 x 1
##    price_million
```

```
##            <dbl>
##  1          47
##  2          45
##  3          44
##  4          43.5
##  5          42.5
##  6          42.5
##  7          42.5
##  8          42.5
##  9          42
## 10          41.5
## 11          41.5
```

Ordering

If one wants to have ordered data with respect to specific column(s), `arrange()` function is used in dplyr. To arrange data by life expectancy, we use `arrange()` function

If one wants order from top to bottom, then use `arrange(desc())` command as follows:

Summarising data

Another feature of dplyr is `summarise` data

```
pkw |> filter(year==2018,hp==1300) |> summarise(mean=mean(price),min=min(price),max=max(pr
## # A tibble: 1 x 3
##       mean    min     max
##      <dbl>  <dbl>   <dbl>
## 1 2393775. 239000 3285000
```

Summarising data by groups

```
pkw |> filter(year==2018,hp==1300) |> group_by(company) |> summarise(mean=mean(price),min=
## # A tibble: 4 x 4
##   company     mean     min     max
##   <chr>      <dbl>   <dbl>   <dbl>
## 1 FAW     1305136.  700000 1500000
## 2 Honda   2301686.  239000 2750000
```

```
## 3 Suzuki  1820240  1350000 3285000
## 4 Toyota  2661864.  560000 3200000
```

if_else command alongwith mutate

```
pkw |>
  filter(year == 2017) |>
  group_by(company) |>
  mutate(under_25 = if_else(Mileage<25000, "Y", "N"))|>
  summarize(avg_price = mean(price))
```

Total number of cars for sale year-wise

```
pkw
## # A tibble: 44,917 x 20
##    `Ad No` Name         Price  year Locat~1 Mileage Regis~2 Engin~3 Engin~4
##      <dbl> <chr>        <chr> <dbl> <chr>     <dbl> <chr>   <chr>   <chr>
##  1 4096758 Toyota Vitz~ 2385~  2017 G- 8, ~    9869 Un-Reg~ Petrol  1000 cc
##  2 4168305 Toyota Coro~ 1110~  2019 Peshaw~   11111 Islama~ Petrol  1300 cc
##  3 4168298 Suzuki Alto~ 1530~  2019 Akora ~   17500 Un-Reg~ Petrol  660 cc
##  4 4168307 Suzuki Alto~ 1650~  2019 Abdull~    9600 Lahore  Petrol  660 cc
##  5 4168306 Toyota Coro~ 1435~  2010 9th Av~  120000 Islama~ Petrol  1300 cc
##  6 4168303 Honda Civic~ 3850~  2017 Peshaw~   22000 Islama~ Petrol  1500 cc
##  7 4168304 Suzuki Wago~ 1440~  2017 Gulber~   31000 Lahore  Petrol  1000 cc
##  8 4168309 Mitsubishi ~ 1425~  2012 Askari~  101000 Lahore  Petrol  1000 cc
##  9 4168310 Toyota Prad~ 2650~  1998 Sargod~  110000 Rawalp~ Diesel  3000 cc
## 10 4168311 Honda Civic~ 3350~  2017 Air Av~   60000 Lahore  Petrol  1800 cc
## # ... with 44,907 more rows, 11 more variables: Transmission <chr>,
## #   Color <chr>, Assembly <chr>, `Body Type` <chr>, Features <chr>,
## #   `Last Updated` <chr>, URL <chr>, price <dbl>, hp <dbl>, company <chr>,
## #   price_m <dbl>, and abbreviated variable names 1: Location,
## #   2: `Registered City`, 3: `Engine Type`, 4: `Engine Capacity`
pkw|>filter(hp==1300)|>group_by(year,company)|>
  count(Color)|>summarise(total=sum(n))|>arrange(desc(year))
## # A tibble: 135 x 3
## # Groups:   year [30]
##    year company total
##   <dbl> <chr>   <int>
```

```
##  1  2019 FAW         5
##  2  2019 Honda     228
##  3  2019 Suzuki     84
##  4  2019 Toyota    345
##  5  2018 FAW        22
##  6  2018 Honda     334
##  7  2018 Suzuki    100
##  8  2018 Toyota    418
##  9  2017 FAW        19
## 10  2017 Honda     237
## # ... with 125 more rows
```

Percentiles

In general, higher the Mileage (more the age of a car) , lower the price. To test this assumption, lets calculate percentiles of price. This will indicate how many have ranking lower than the current country.

```
pkw %>% select(price,year, Mileage) %>%
  filter(year == 2017) %>%
  mutate(percentile = ntile(Mileage, 100)) %>%
  arrange(desc(price))
## # A tibble: 4,038 x 4
##       price  year Mileage percentile
##       <dbl> <dbl>   <dbl>      <int>
##  1 47000000  2017   18000         11
##  2 45000000  2017   20000         12
##  3 44000000  2017     700          1
##  4 43500000  2017   13000          6
##  5 42500000  2017      50          1
##  6 42500000  2017   25000         17
##  7 42500000  2017   60000         75
##  8 42500000  2017   18000         10
##  9 42000000  2017   16000          9
## 10 41500000  2017   14000          7
## # ... with 4,028 more rows
```

.

So it makes sense that higher the Mileage, lower the price. This is not formal testing but exploratory data makes lot of sense here.

Data visualization

Filtering data as done in introductory analysis seems quite difficult if you are not familiar with these simple things. But if you are working with dplyr for quite sometime, there is not anything very advanced or difficult.

For example, let's say you have to find out the top 10 countries in the 90th percentile regarding life expectancy in 2007. You can reuse some of the logic from the previous sections, but answering this question alone requires `multiple filtering` and `subsetting`:

```
pkw %>% filter(year==2017) %>%
  mutate(percentile=ntile(price,100)) %>%
  filter(percentile>90) %>%
  arrange(desc(percentile)) %>%
  top_n(10,wt=percentile) %>%
  select(company,Mileage,price,hp)
## # A tibble: 40 x 4
##    company  Mileage    price    hp
##    <chr>      <dbl>    <dbl> <dbl>
##  1 Mercedes   11000 15500000  2000
##  2 Toyota        50 42500000  4608
##  3 Toyota      4000 26500000  2700
##  4 Toyota     25000 42500000  4608
##  5 Audi       21000 13500000  1800
##  6 Toyota       700 44000000  4600
##  7 Mercedes   12900 17500000  2000
##  8 Toyota     60000 38000000  4608
##  9 Toyota     16500 25500000  2700
## 10 Toyota     20000 45000000  4600
## # ... with 30 more rows
```

In case you are interested in bottom 10 (lowest price cars), use `top_n` with `-10`.

```
pkw %>% filter(year==2017) %>%
  mutate(percentile=ntile(price,100)) %>%
  filter(percentile<10) %>%
  arrange(desc(percentile)) %>%
  top_n(-10,wt=percentile) %>%
  select(company,Mileage,price,hp)
## # A tibble: 41 x 4
##    company Mileage  price    hp
##    <chr>     <dbl>  <dbl> <dbl>
```

```
##  1 Toyota    97000 123000  2700
##  2 Suzuki    77928 725000   800
##  3 Suzuki    40000 700000   800
##  4 Suzuki    49000 760000   800
##  5 Suzuki    57000 750000   800
##  6 Suzuki    74000 740000   800
##  7 Suzuki    60000 640000   800
##  8 Suzuki    95000 700000  1000
##  9 Suzuki    39000 730000   800
## 10 Suzuki    68000 730000   800
## # ... with 31 more rows
```

Visualizing data to get data insight

Visualizing data is one of the most important aspect of getting data insight and may provide a better data insight than a complicated model. Visualizing large data sets were not an easy task, so researchers relied on mathematical and core econometric/regression models. `ggplot2` which is a set of `tidyverse` package is probably one of the greatest tool for data visualization used in `R`. In the following sections we are going to visualize `gapminder` data.

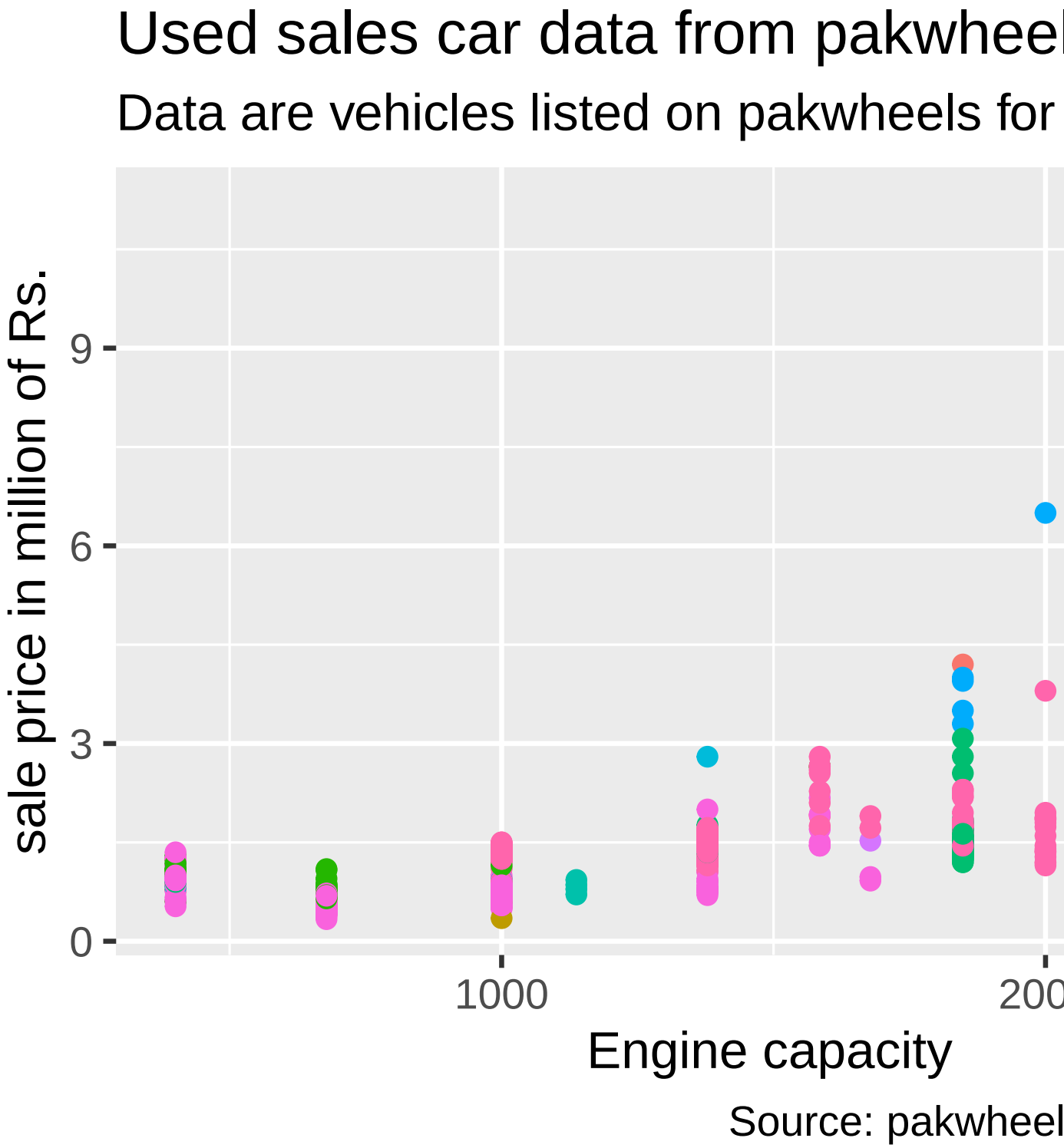
Stat graphics is a mapping of variable to `aes`thetic attributes of `geo`metric objects.

3 Essential components of `ggplot2`

- data: dataset containing the variables of interest
- geom: geometric object in question line, point, bars
- aes: aesthetic attributes of an object x/y position, colors, shape, size

Scatter plot

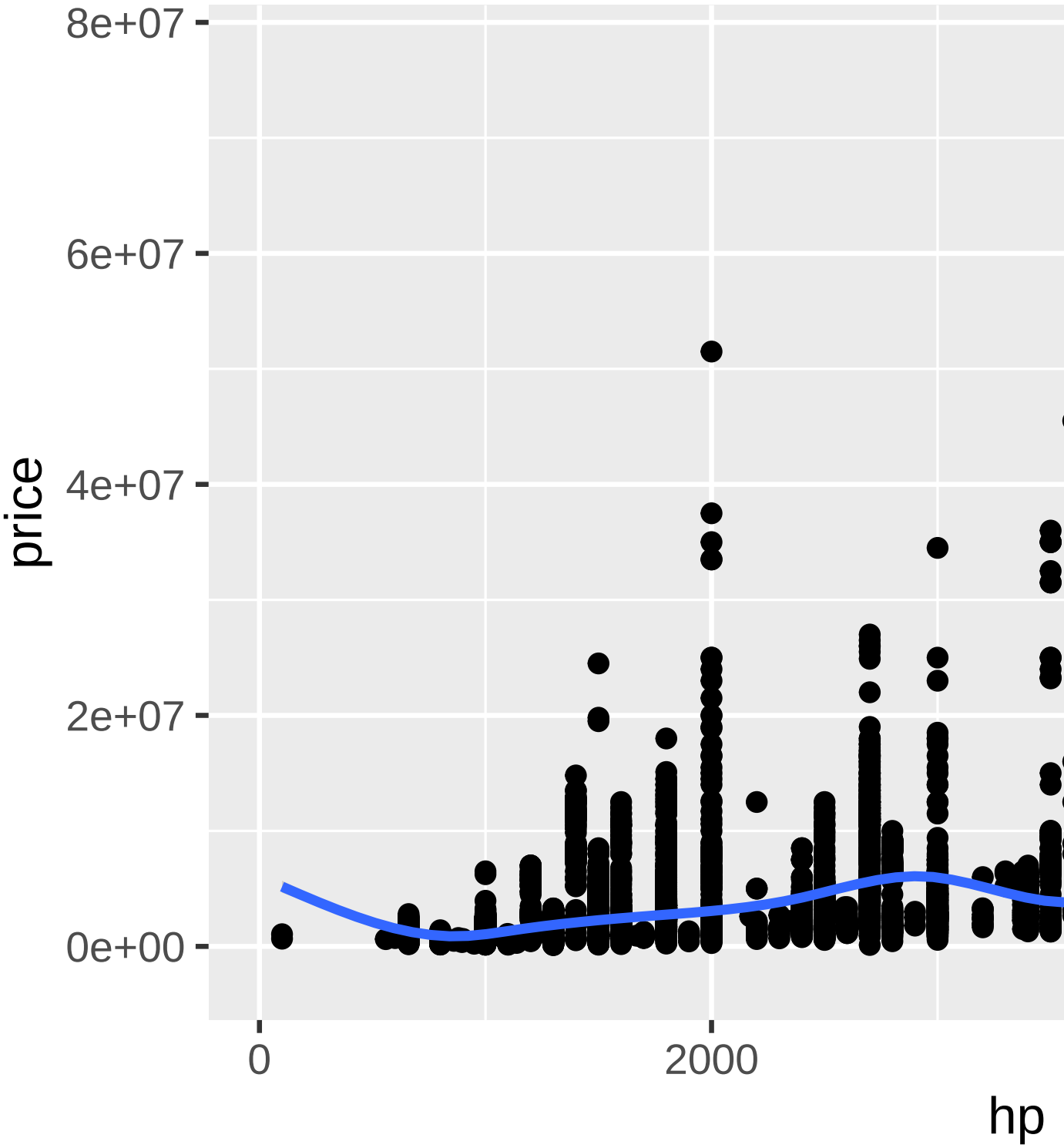
```
p1<-pkw|>filter(hp>=600 & hp<=3000,`year`==2009)|> mutate(price_m=price/1000000)|>
  ggplot(aes(x=hp,y=price_m,color=company))+geom_point()+scale_x_log10()
p1+  labs(x = "Engine capacity", y = "sale price in million of Rs.",
       title = "Used sales car data from pakwheels",
       subtitle = "Data are vehicles listed on pakwheels for sale in 2020",
       caption = "Source: pakwheels, By Zahid Asghar")
```


Used sales car data from pakwheel
Data are vehicles listed on pakwheels for
sale price in million of Rs.
9
6
3
0
1000
200
Engine capacity
Source: pakwheel

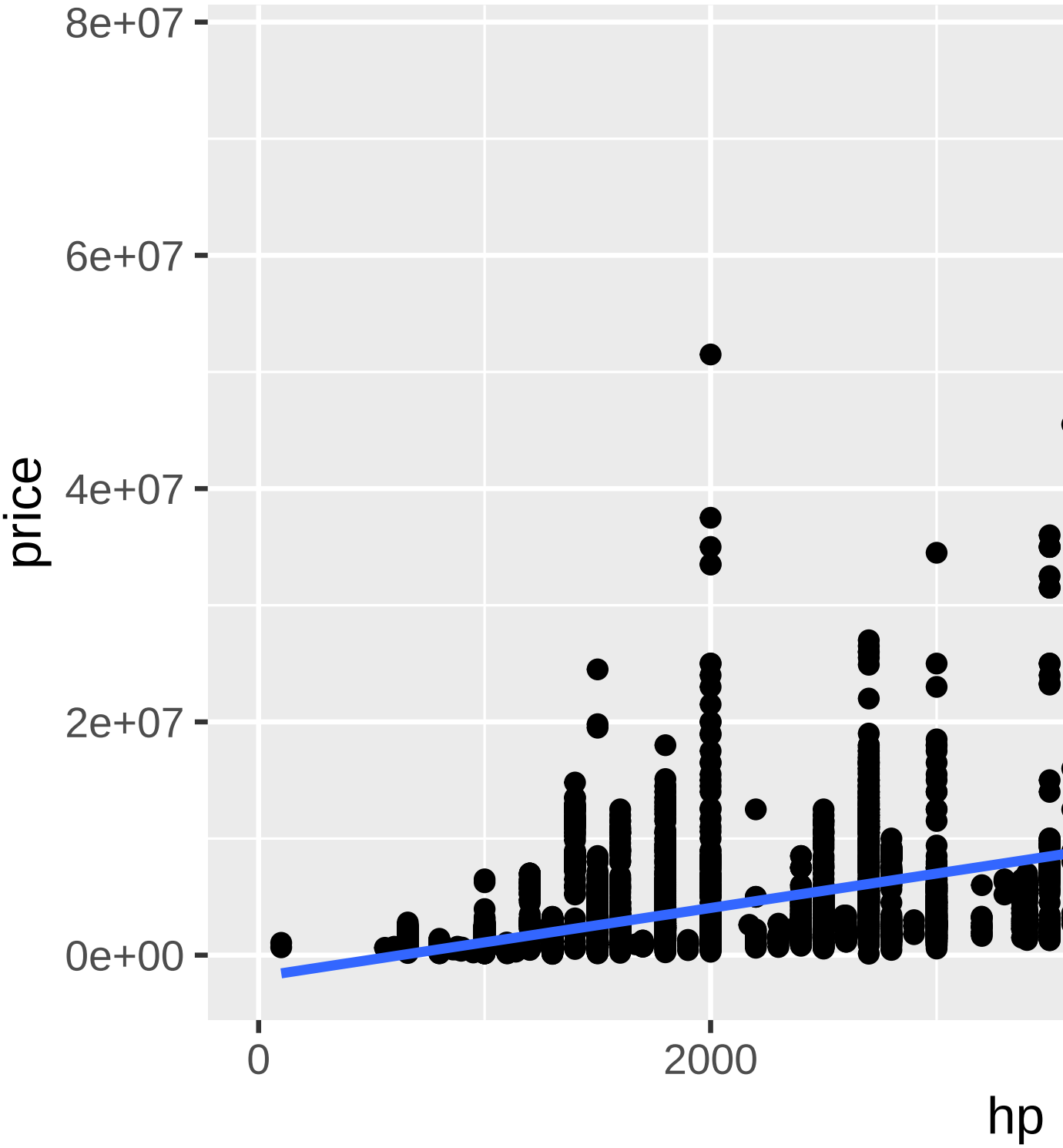
If you just want to highlight the relationship between gbp per capita and life Expectancy you've probably done most of the work now. However, it is a good practice to highlight a few interesting dots in this chart to give more insight to the plot:

```
#| eval: false
#tmp <- pkw %>%
 #mutate(
  # annotation = case_when(
   # hp < 2000 & price < 700000 ~ "yes",
    #price < 3000000 ~ "yes",
     #Mileage > 15000 ~ "yes"
    #)
#) %>% mutate(Mileage=Mileage/10000)
 # arrange(desc(price))

# Plot
#ggplot( tmp, aes(x=hp, y=price, size =Mileage , color = company)) +
 #   geom_point(alpha=0.7) +
  #  scale_size(range = c(1.4, 19), name="Price in Million of Rupees") +
   # scale_color_viridis(discrete=TRUE) +
    #theme_ipsum() +
    #theme(legend.position="none") +
    #geom_text_repel(data=tmp %>% filter(annotation=="yes"), aes(label=company), size=4 )
```

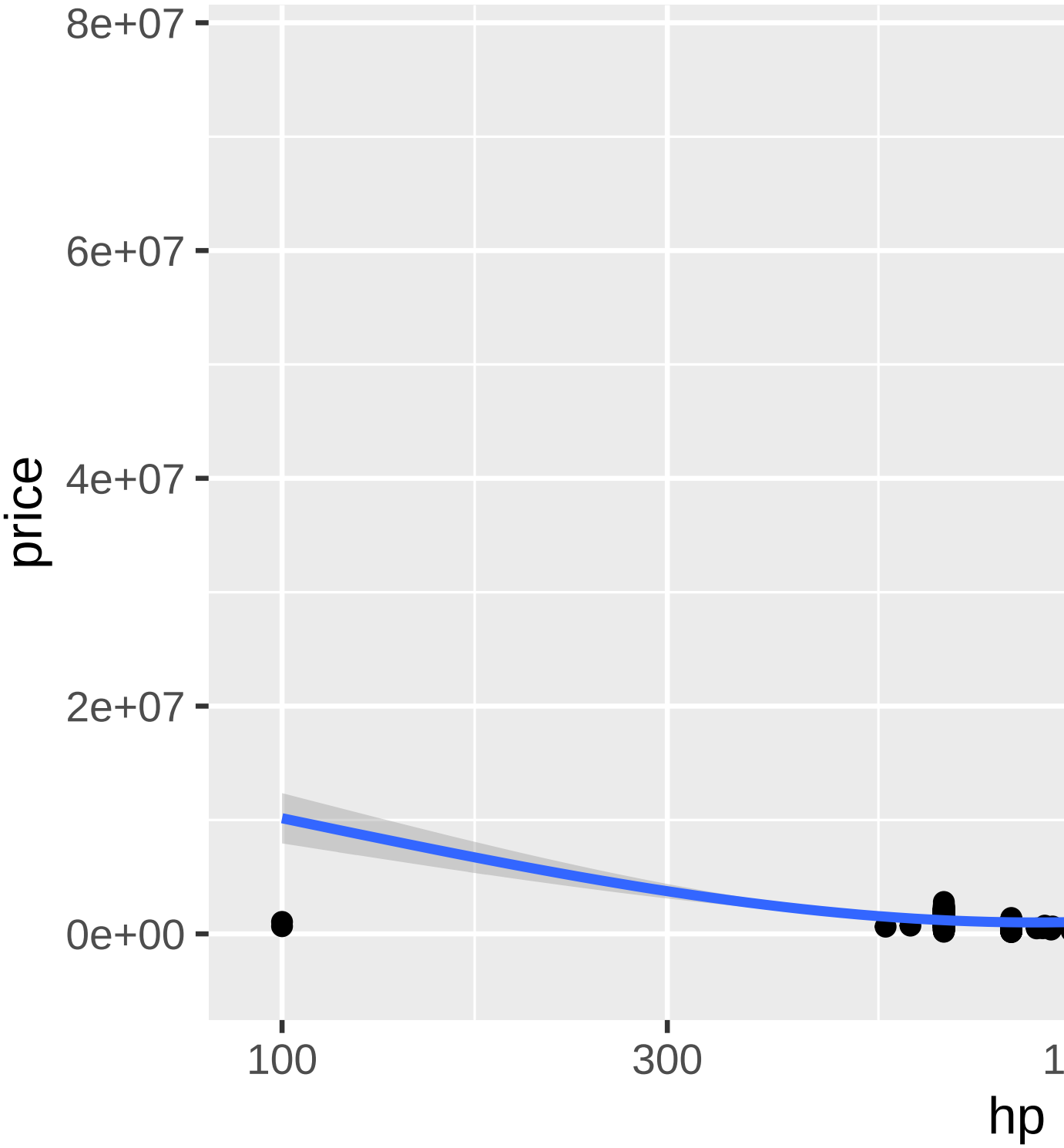
```
##This is a table of data about a large number of countries, each observed over several ye
#| eval: false
P<-ggplot(data=pkw,mapping = aes(x=hp,y=price))

P+geom_point()+geom_smooth()
```


8e+07
6e+07
4e+07
2e+07
0e+00
price
0
2000
hp

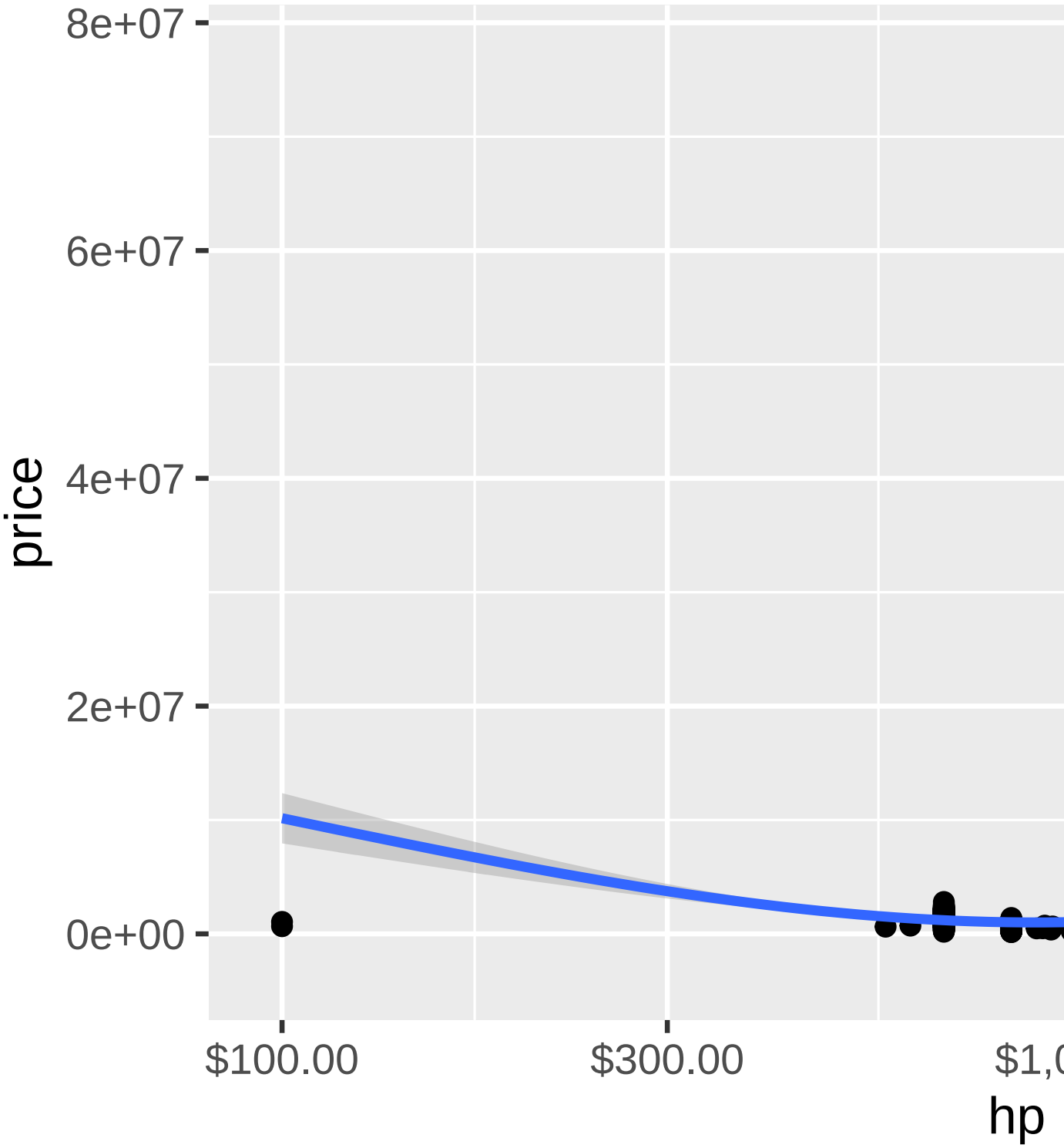
```
P+geom_point()+geom_smooth(method = "lm")
```


8e+07
6e+07
4e+07
2e+07
0e+00
price
0
2000
hp

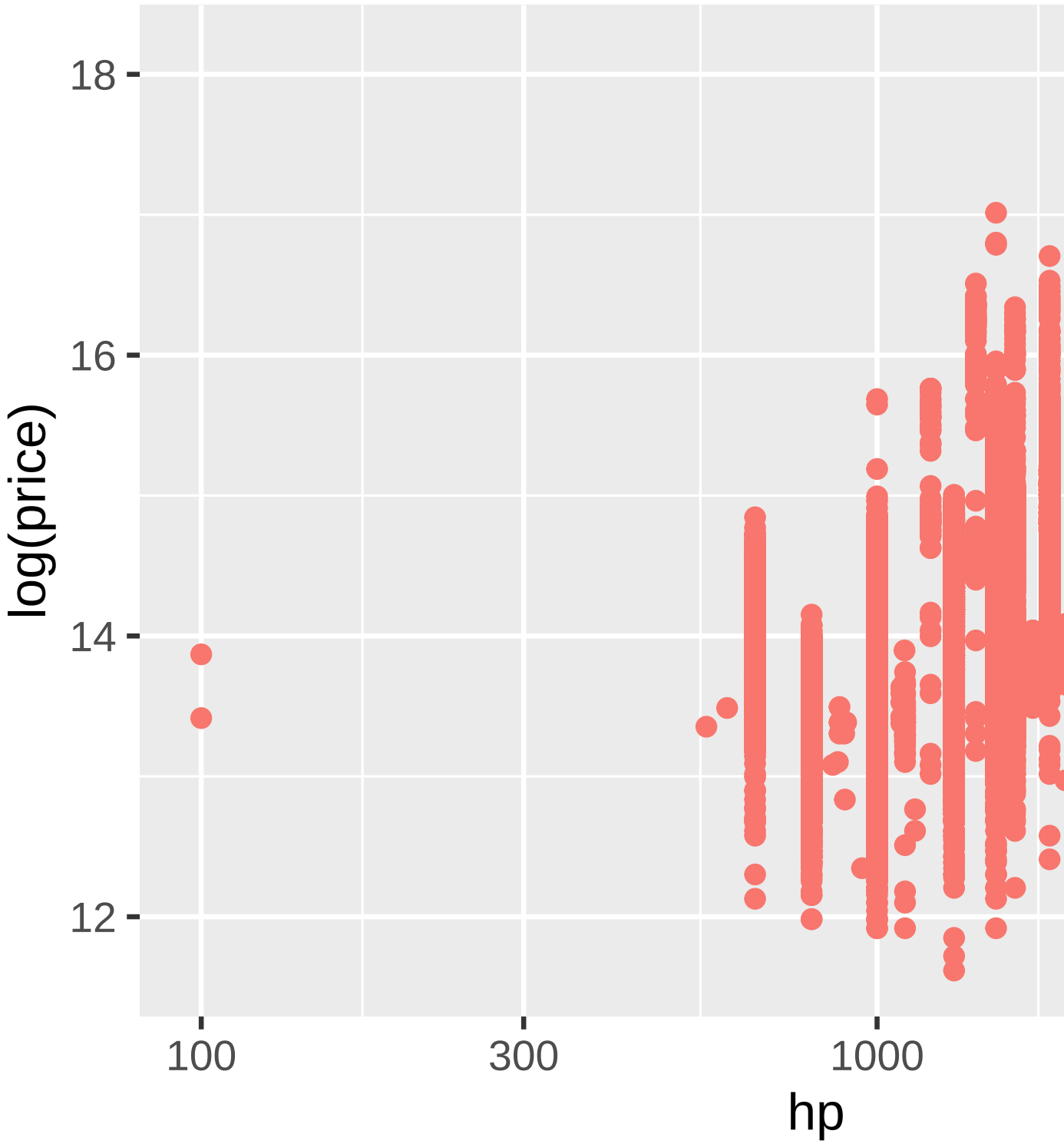
```
P+geom_point()+geom_smooth(method = "gam")+scale_x_log10()
```


8e+07
6e+07
4e+07
2e+07
0e+00
price
100
300
hp

```
P+geom_point()+geom_smooth(method = "gam")+scale_x_log10(labels=scales::dollar)
```

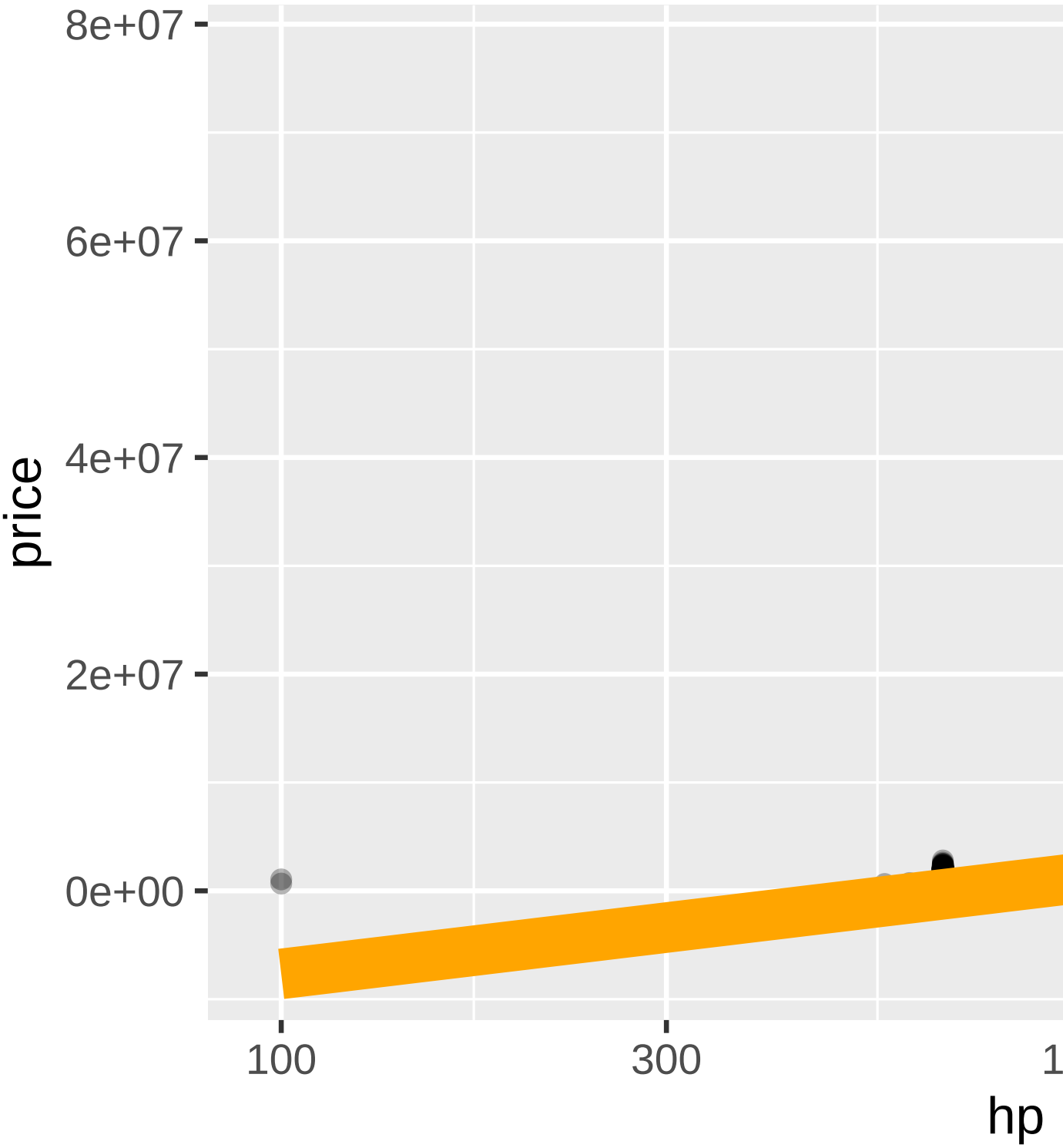

8e+07
6e+07
4e+07
2e+07
0e+00
price
$100.00
$300.00
hp

```
P<-ggplot(data=pkw,mapping = aes(hp,y=log(price),color="purple"))
P+geom_point()+geom_smooth(method = "loess")+scale_x_log10()
```

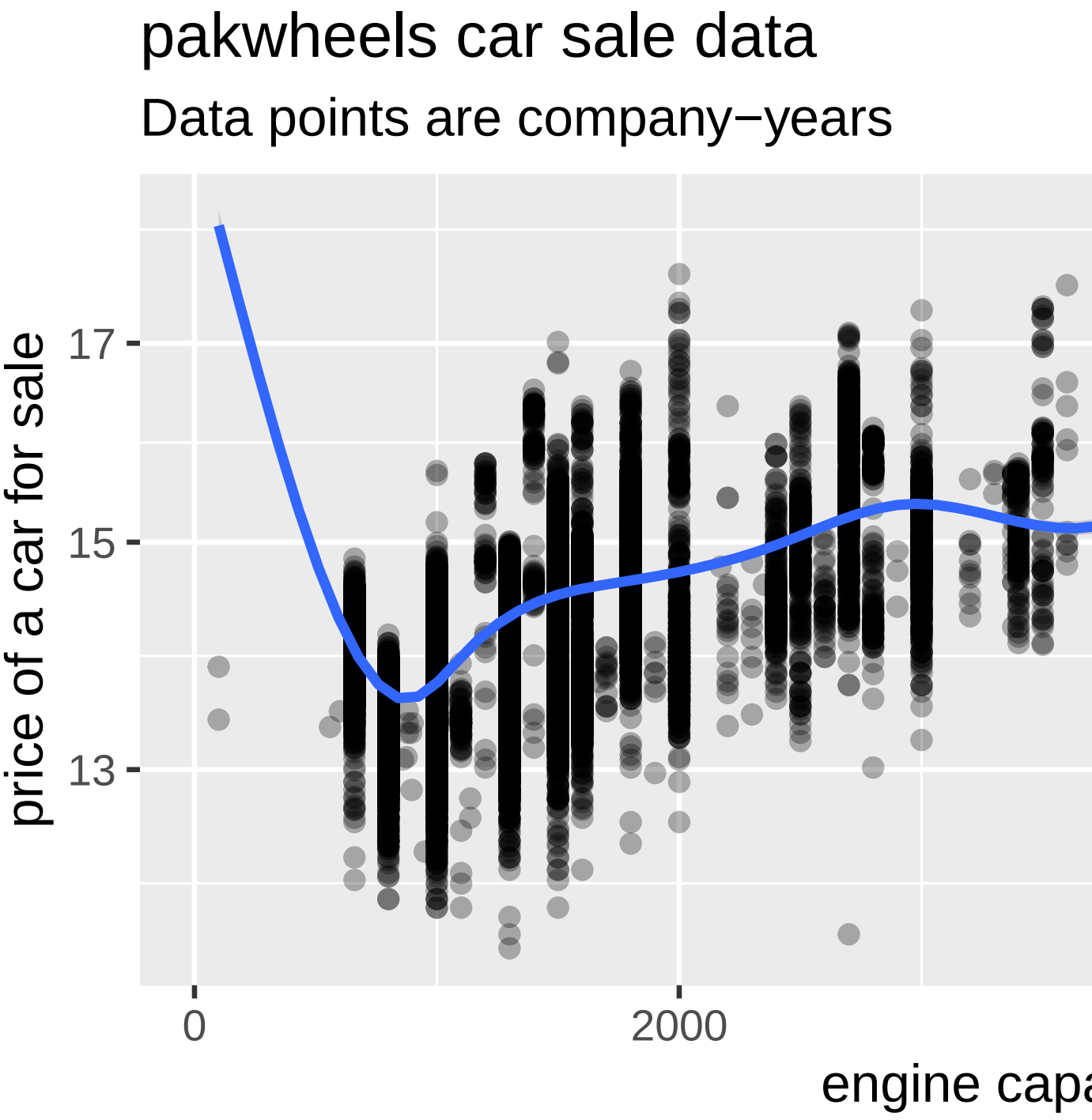

log(price)
18
16
14
12
100
300
1000
hp

```r
##aes() is for variables
P<-ggplot(data=pkw,mapping = aes(hp,y=log(price)))
P+geom_point(color="purple")+geom_smooth(method = "loess")+scale_x_log10()
```

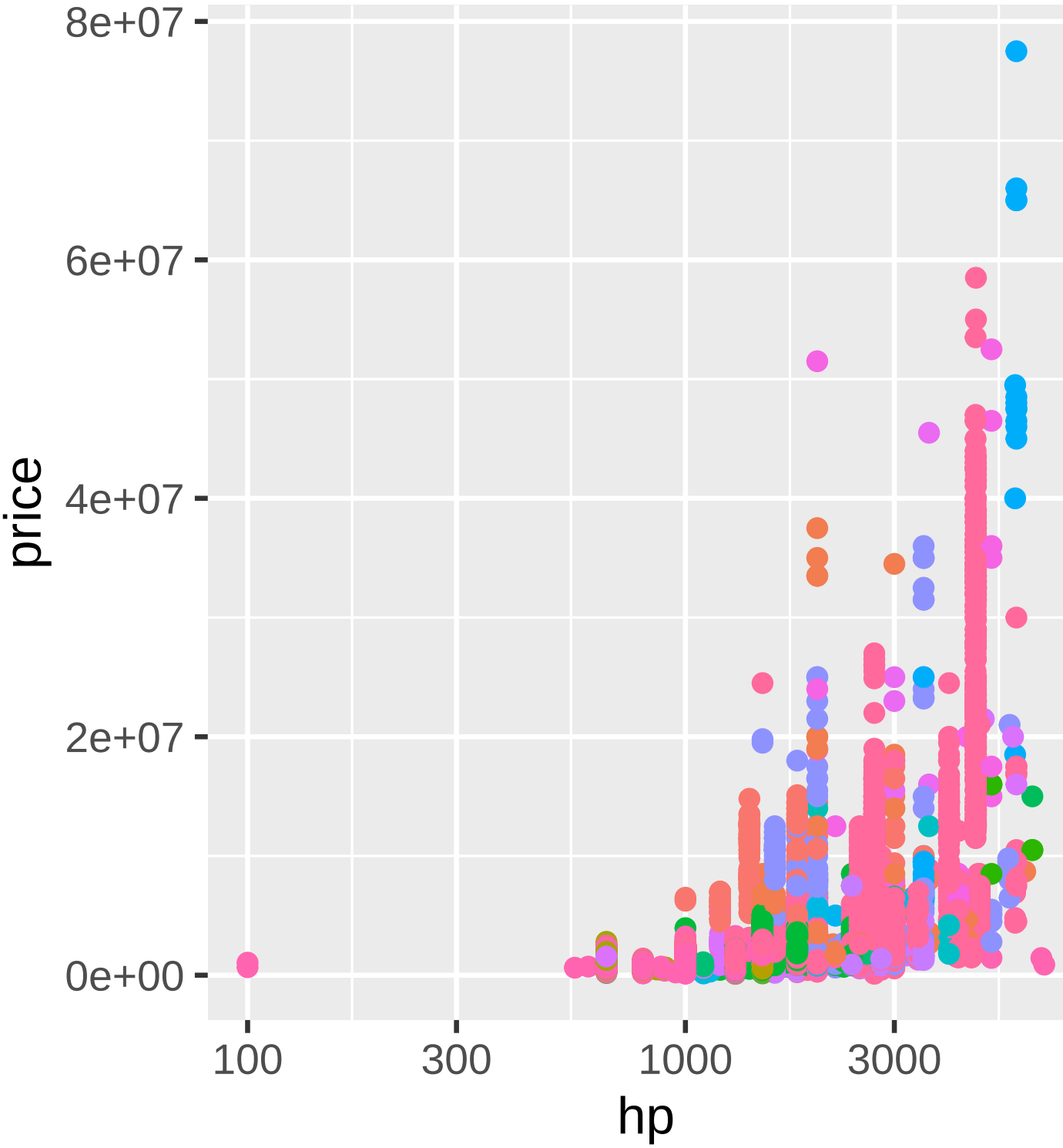
```r
P<-ggplot(data=pkw,mapping = aes(hp,y=price))
P+geom_point(alpha=0.3)+
  geom_smooth(color="orange",se=FALSE,size=5,method = "lm")+
  scale_x_log10()
```


8e+07
6e+07
4e+07
2e+07
0e+00
price
100
300
hp

```r
#With proper title
P<-ggplot(data=pkw,mapping = aes(hp,y=log(price)))
P+geom_point(alpha=0.3)+
  geom_smooth(method = "gam")+
  scale_y_log10()+
  labs(x = "engine capacity", y = "price of a car for sale",
       title = "pakwheels car sale data",
       subtitle = "Data points are company-years",
       caption = "Source: pakwheels.")
```

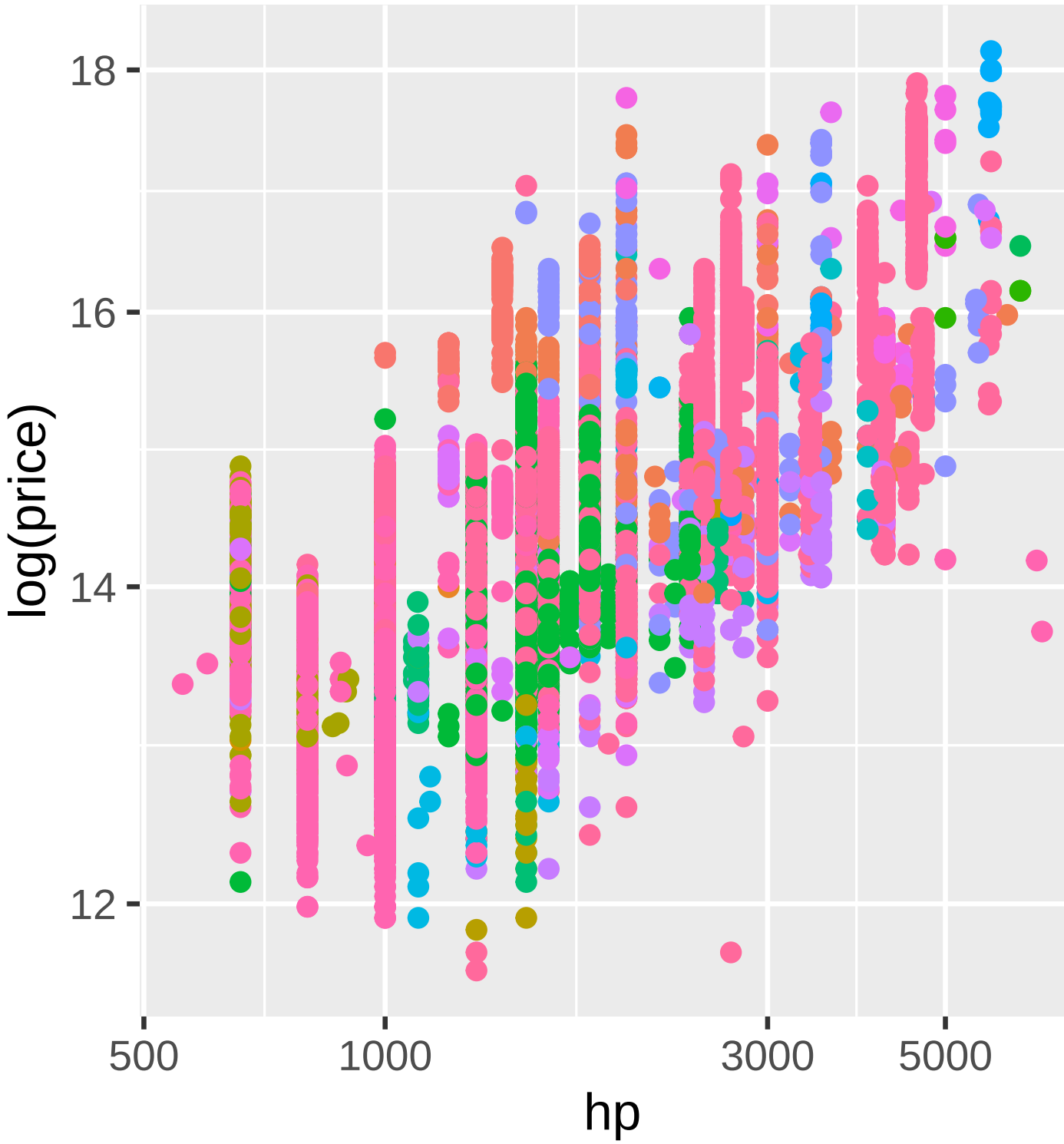

pakwheels car sale data
Data points are company−years
price of a car for sale
17
15
13
0
2000
engine capa

```
##Continent wise
#| eval: false

p <- ggplot(data = pkw,
            mapping = aes(x = hp,
                          y = price,
                          color = company))
p + geom_point() +
  geom_smooth(method = "loess") +
  scale_x_log10()
```

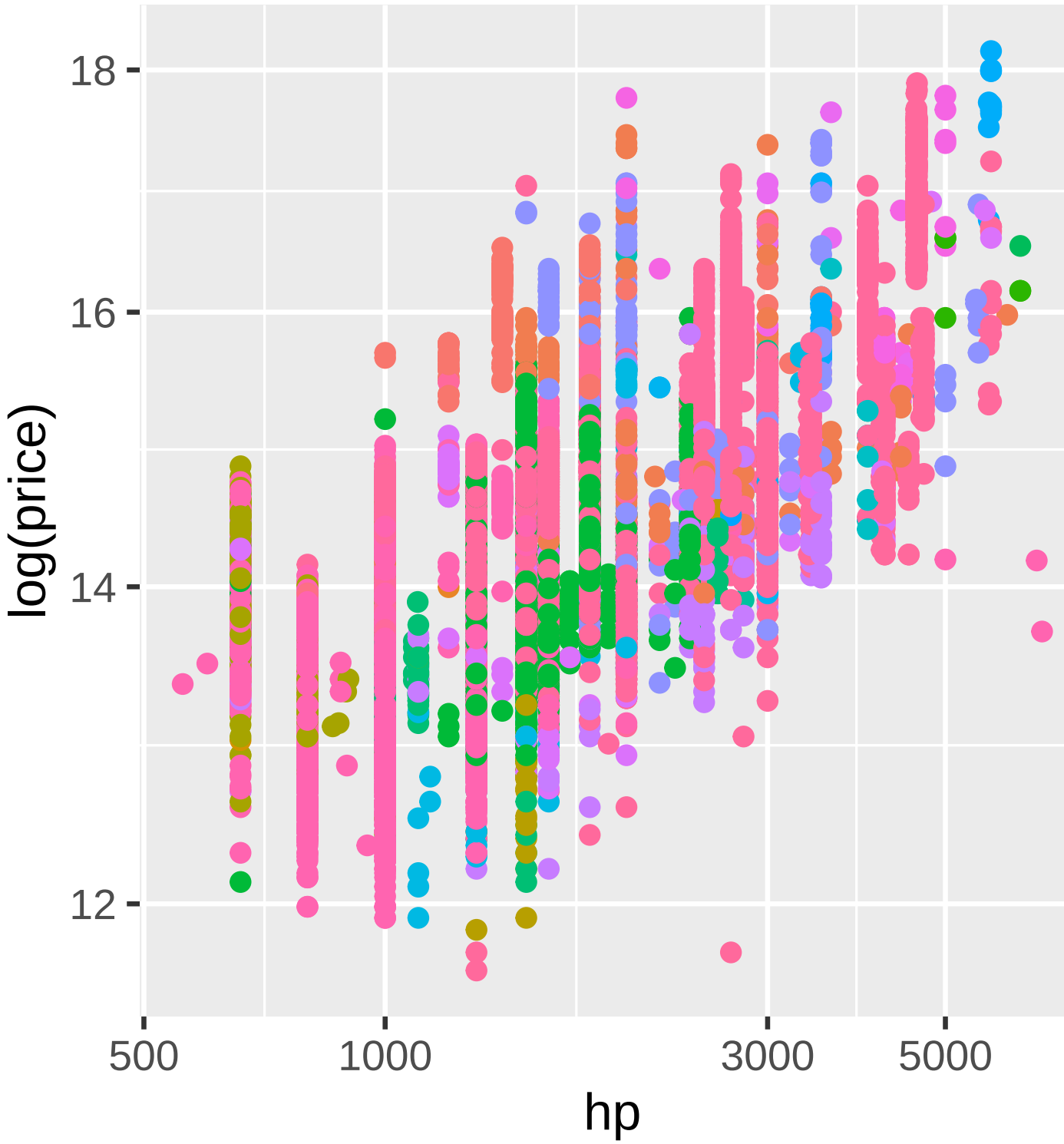

8e+07
6e+07
4e+07
2e+07
0e+00
price
100
300
1000
3000
hp

```
p <-pkw|> filter(hp>300)|> ggplot(aes(x = hp,
                          y = price,
                          color = Assembly,
                          fill = Assembly))
p + geom_point() +
  geom_smooth(method = "loess") +
  scale_x_log10()+scale_y_continuous()
```

```
##Aesthetics can be mapped per geom
#| eval: false
p <-pkw|>filter(hp>500)|> ggplot(aes(x = hp, y = log(price)))
p + geom_point(mapping = aes(color = company)) +
  geom_smooth(method = "loess") +scale_y_log10()+
  scale_x_log10()
```

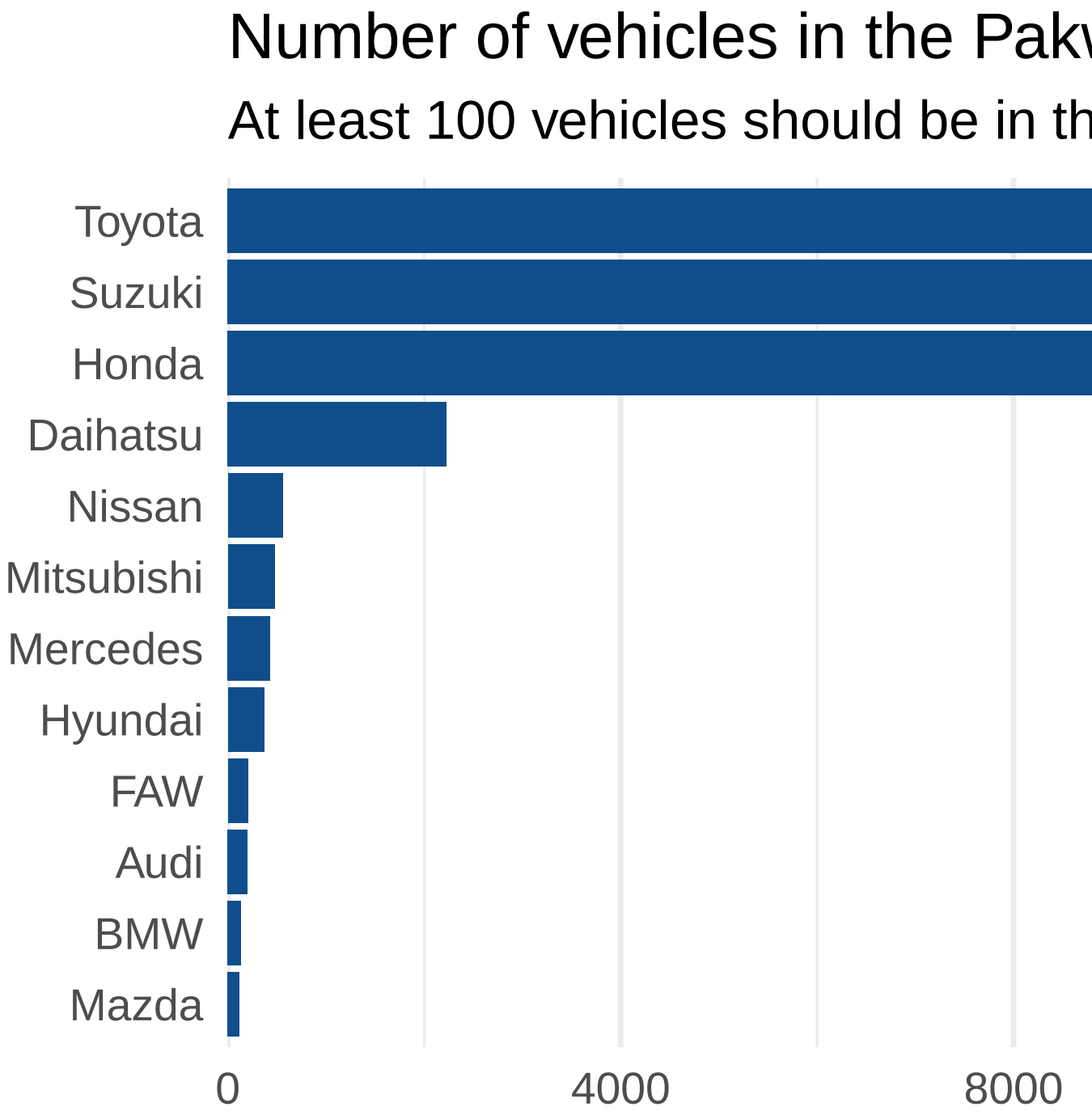

18
16
14
12
log(price)
500
1000
3000
5000
hp

```
p + geom_point(mapping = aes(color = company)) +
  scale_x_log10()+scale_y_log10()
```


18
16
14
12
log(price)
500
1000
3000
5000
hp

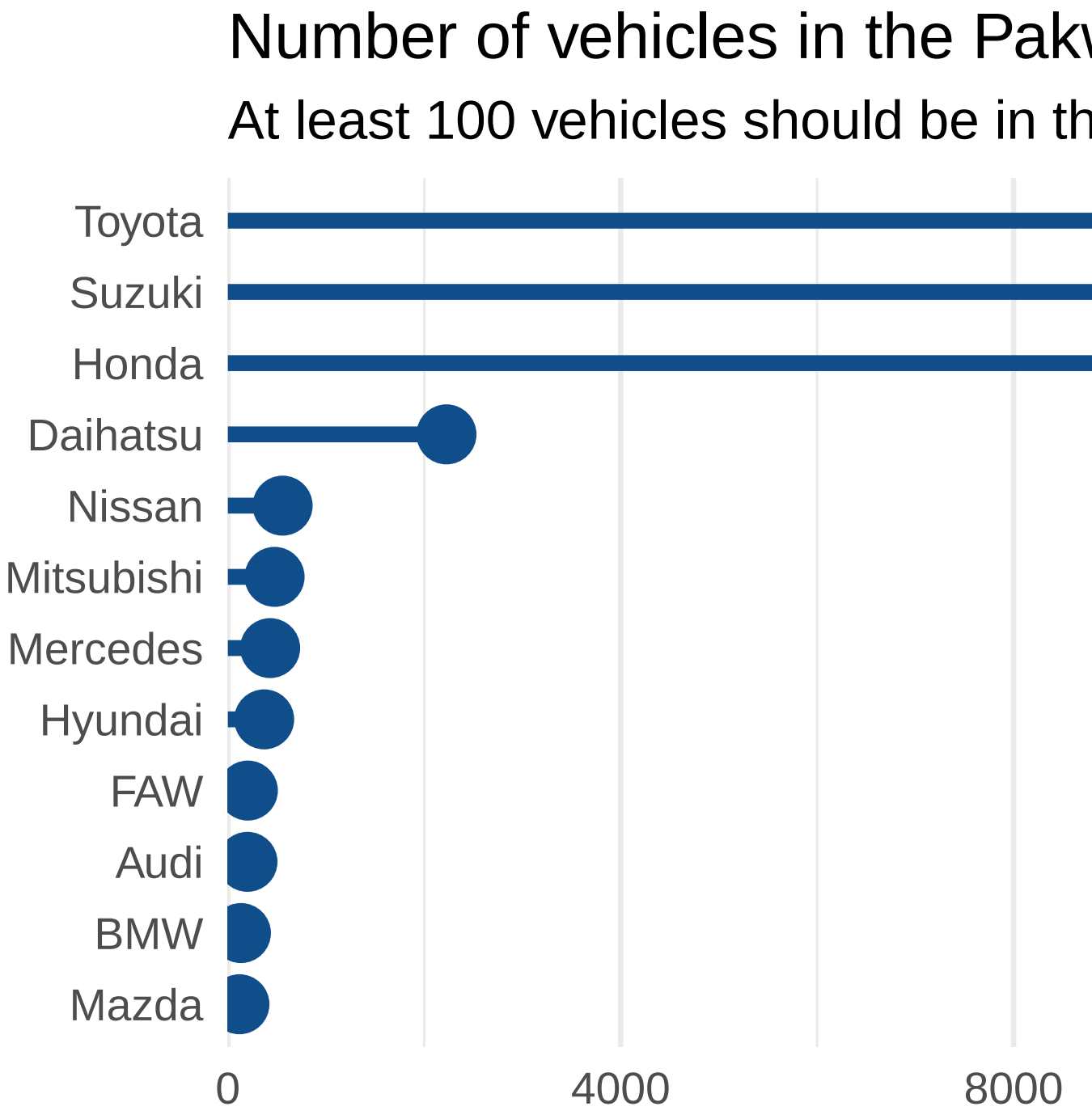
Bar plot

```
manufacturers <- pkw |>
  count(company, sort = TRUE) |>
  mutate(
    manufacturer = str_to_title(company),
    manufacturer = fct_reorder(company, n)
  )|>na.omit()
manufacturers |> filter(n>100)|>
  ggplot(aes(y = manufacturer, x = n)) +
  geom_col(fill = 'dodgerblue4') +
  theme_minimal() +
  scale_x_continuous(
    expand = expansion(mult = c(0, 0.05))
  ) +
  labs(
    x = element_blank(),
    y = element_blank(),
    title = 'Number of vehicles in the Pakwheels data set',
    subtitle = "At least 100 vehicles should be in the data to be included in graph",
    caption = "Source: Pakwheels| Zahid Asghar "
  ) +
  theme(
    panel.grid.major.y = element_blank()
  )
```


Number of vehicles in the Pak
At least 100 vehicles should be in th
Toyota
Suzuki
Honda
Daihatsu
Nissan
Mitsubishi
Mercedes
Hyundai
FAW
Audi
BMW
Mazda
0
4000
8000

Lollipop chart

```
manufacturers |> filter(n>100)|>
  ggplot(aes(y = manufacturer, x = n)) +
  geom_point(col = 'dodgerblue4', size = 5) +
  geom_segment(
    aes(x = 0, xend = n, y = manufacturer, yend = manufacturer),
    linewidth = 1.5,
    col = 'dodgerblue4'
  ) +
  theme_minimal() +
  scale_x_continuous(
    expand = expansion(mult = c(0, 0.05))
  ) +
  labs(
    x = element_blank(),
    y = element_blank(),
    title = 'Number of vehicles in the Pakwheels data set',
    subtitle = "At least 100 vehicles should be in the data to be included in graph",
    caption = "Source: Pakwheels| Zahid Asghar "
  ) +
  theme(
    panel.grid.major.y = element_blank()
  )
```


Number of vehicles in the Pak
At least 100 vehicles should be in th
Toyota
Suzuki
Honda
Daihatsu
Nissan
Mitsubishi
Mercedes
Hyundai
FAW
Audi
BMW
Mazda
0
4000
8000