Data Visualization with R

Here, I have tried to include most of the visualizations which are useful in data analysis.

Loading libraries

In [2]:

```
#Loading some libraries
library(readr)
library(ggplot2)
library(dplyr)
library(yarrr)
library(e1071)
library(caret)
library(corrgram)
library(usmap)
library(maps)
library(mapdata)
library(ggmap)
library(tmap)
library(sp)
library(leaflet)
library(tidyverse)
library(ggfittext)
library(treemapify)
library(ggridges)
library(ggcorrplot)
library(vcd)
library(scales)
library(rmarkdown)
library(rgdal)
```

Importing data

In [3]:

```
# Importing data
df = read.csv('raw_automobile_data.csv')
# getting only 5 rows
head(df, 5)
```

symboling	normalized_losses	make	fuel_type	aspiration	num_of_doors	body_style	drive_w
3	NA	alfa-romero	gas	std	two	convertible	
3	NA	alfa-romero	gas	std	two	convertible	
1	NA	alfa-romero	gas	std	two	hatchback	
2	164	audi	gas	std	four	sedan	
2	164	audi	gas	std	four	sedan	

In [4]:

```
#Getting total missing values in each column
colSums(is.na(df))
```

symboling
0
normalized_losses
41
make
0
fuel_type
0
aspiration
0
num_of_doors
0
body_style
0
drive_wheels
0
engine_location
0
wheel_base
0
length

0
width
0
height
0
curb_weight
0
engine_type
0
num_of_cylinders
0
engine_size
0
fuel_system
0
bore
4
stroke
4
compression_ratio
0
horsepower
2
peak_rpm
2
city_mpg
0
highway_mpg
0
price
4

Replacing the missing Values

In [5]:

```
#Replacing normalized_locess by mean
df$normalized_losses[is.na(df$normalized_losses)] <- mean(df$normalized_losses,
na.rm = TRUE)
```

In [6]:

```
#Replacing stroke by mean
df$stroke[is.na(df$stroke)] <- mean(df$stroke, na.rm = TRUE)
```

In [7]:

```
#Replacing bore by mean
df$bore[is.na(df$bore)] <- mean(df$bore, na.rm = TRUE)
```

In [8]:

```
#Replacing horsepower by mean
df$horsepower[is.na(df$horsepower)] <- mean(df$horsepower, na.rm = TRUE)
```

In [9]:

```
#Replacing peak_rpm by mean
df$peak_rpm[is.na(df$peak_rpm)] <- mean(df$peak_rpm, na.rm = TRUE)
```

In [10]:

```
#Replacing peak_rpm by Mode
df$num_of_doors[is.na(df$num_of_doors)] <- 'four'
```

In [11]:

```
#Deleting missing values in Price
df <- na.omit(df)
```

In [12]:

```
head(df)
```

symboling	normalized_losses	make	fuel_type	aspiration	num_of_doors	body_style	drive_w
3	122	alfa-romero	gas	std	two	convertible	
3	122	alfa-romero	gas	std	two	convertible	
1	122	alfa-romero	gas	std	two	hatchback	
2	164	audi	gas	std	four	sedan	
2	164	audi	gas	std	four	sedan	
2	122	audi	gas	std	two	sedan	

In [13]:

```
#Getting structure of data frame - name, type and preview of data in each column
str(df)
```

```
'data.frame':   201 obs. of  26 variables:
 $ symboling        : int  3 3 1 2 2 2 1 1 1 2 ...
 $ normalized_losses: num  122 122 122 164 164 122 158 122 158 192 .
..
 $ make             : Factor w/ 22 levels "alfa-romero",..: 1 1 1 2
2 2 2 2 2 3 ...
 $ fuel_type        : Factor w/ 2 levels "diesel","gas": 2 2 2 2 2 2
2 2 2 2 ...
 $ aspiration       : Factor w/ 2 levels "std","turbo": 1 1 1 1 1 1
1 1 2 1 ...
 $ num_of_doors     : Factor w/ 3 levels "","four","two": 3 3 3 2 2
3 2 2 2 3 ...
 $ body_style       : Factor w/ 5 levels "convertible",..: 1 1 3 4 4
4 4 5 4 4 ...
 $ drive_wheels     : Factor w/ 3 levels "4wd","fwd","rwd": 3 3 3 2
1 2 2 2 2 3 ...
 $ engine_location  : Factor w/ 2 levels "front","rear": 1 1 1 1 1 1
1 1 1 1 ...
 $ wheel_base       : num  88.6 88.6 94.5 99.8 99.4 ...
 $ length           : num  169 169 171 177 177 ...
 $ width            : num  64.1 64.1 65.5 66.2 66.4 66.3 71.4 71.4 7
1.4 64.8 ...
 $ height           : num  48.8 48.8 52.4 54.3 54.3 53.1 55.7 55.7 5
5.9 54.3 ...
 $ curb_weight      : int  2548 2548 2823 2337 2824 2507 2844 2954 3
086 2395 ...
 $ engine_type      : Factor w/ 7 levels "dohc","dohcv",..: 1 1 6 4
4 4 4 4 4 4 ...
 $ num_of_cylinders : Factor w/ 7 levels "eight","five",..: 3 3 4 3
2 2 2 2 2 3 ...
 $ engine_size      : int  130 130 152 109 136 136 136 136 131 108 .
..
 $ fuel_system      : Factor w/ 8 levels "1bbl","2bbl",..: 6 6 6 6 6
6 6 6 6 6 ...
 $ bore             : num  3.47 3.47 2.68 3.19 3.19 3.19 3.19 3.19 3
.13 3.5 ...
 $ stroke           : num  2.68 2.68 3.47 3.4 3.4 3.4 3.4 3.4 3.4 2.
8 ...
 $ compression_ratio: num  9 9 9 10 8 8.5 8.5 8.5 8.3 8.8 ...
 $ horsepower       : num  111 111 154 102 115 110 110 110 140 101 .
..
 $ peak_rpm         : num  5000 5000 5000 5500 5500 5500 5500 5500 5
500 5800 ...
 $ city_mpg         : int  21 21 19 24 18 19 19 19 17 23 ...
 $ highway_mpg      : int  27 27 26 30 22 25 25 25 20 29 ...
 $ price            : int  13495 16500 16500 13950 17450 15250 17710
18920 23875 16430 ...
 - attr(*, "na.action")= 'omit' Named int  10 45 46 130
  ..- attr(*, "names")= chr  "10" "45" "46" "130"
```

In [14]:

```
# Summary statistics of all varibles in the data frame
summary(df)
```

```
   symboling       normalized_losses         make     fuel_type   as
piration
 Min.   :-2.0000   Min.   : 65       toyota    :32   diesel: 20   st
d  :165
 1st Qu.: 0.0000   1st Qu.:101       nissan    :18   gas   :181   tu
rbo: 36
 Median : 1.0000   Median :122       mazda     :17
 Mean   : 0.8408   Mean   :122       honda     :13
 3rd Qu.: 2.0000   3rd Qu.:137       mitsubishi:13
 Max.   : 3.0000   Max.   :256       subaru    :12
                                     (Other)   :96
 num_of_doors       body_style drive_wheels engine_location   wheel_
base
     :  2     convertible: 6   4wd:  8      front:198       Min.   :
86.6
 four:113     hardtop    : 8   fwd:118      rear :  3       1st Qu.:
94.5
 two : 86     hatchback  :68   rwd: 75                      Median :
97.0
              sedan      :94                                Mean   :
98.8
              wagon      :25                                3rd Qu.:
102.4
                                                            Max.   :
120.9

     length          width           height       curb_weight   engi
ne_type
 Min.   :141.1   Min.   :60.30   Min.   :47.80   Min.   :1488   dohc
: 12
 1st Qu.:166.8   1st Qu.:64.10   1st Qu.:52.00   1st Qu.:2169   dohc
v:  0
 Median :173.2   Median :65.50   Median :54.10   Median :2414   l
: 12
 Mean   :174.2   Mean   :65.89   Mean   :53.77   Mean   :2556   ohc
:145
 3rd Qu.:183.5   3rd Qu.:66.60   3rd Qu.:55.50   3rd Qu.:2926   ohcf
: 15
 Max.   :208.1   Max.   :72.00   Max.   :59.80   Max.   :4066   ohcv
: 13
                                                              roto
r:  4
 num_of_cylinders  engine_size     fuel_system      bore            s
troke
 eight :  4       Min.   : 61.0   mpfi   :92   Min.   :2.540   Min.
```

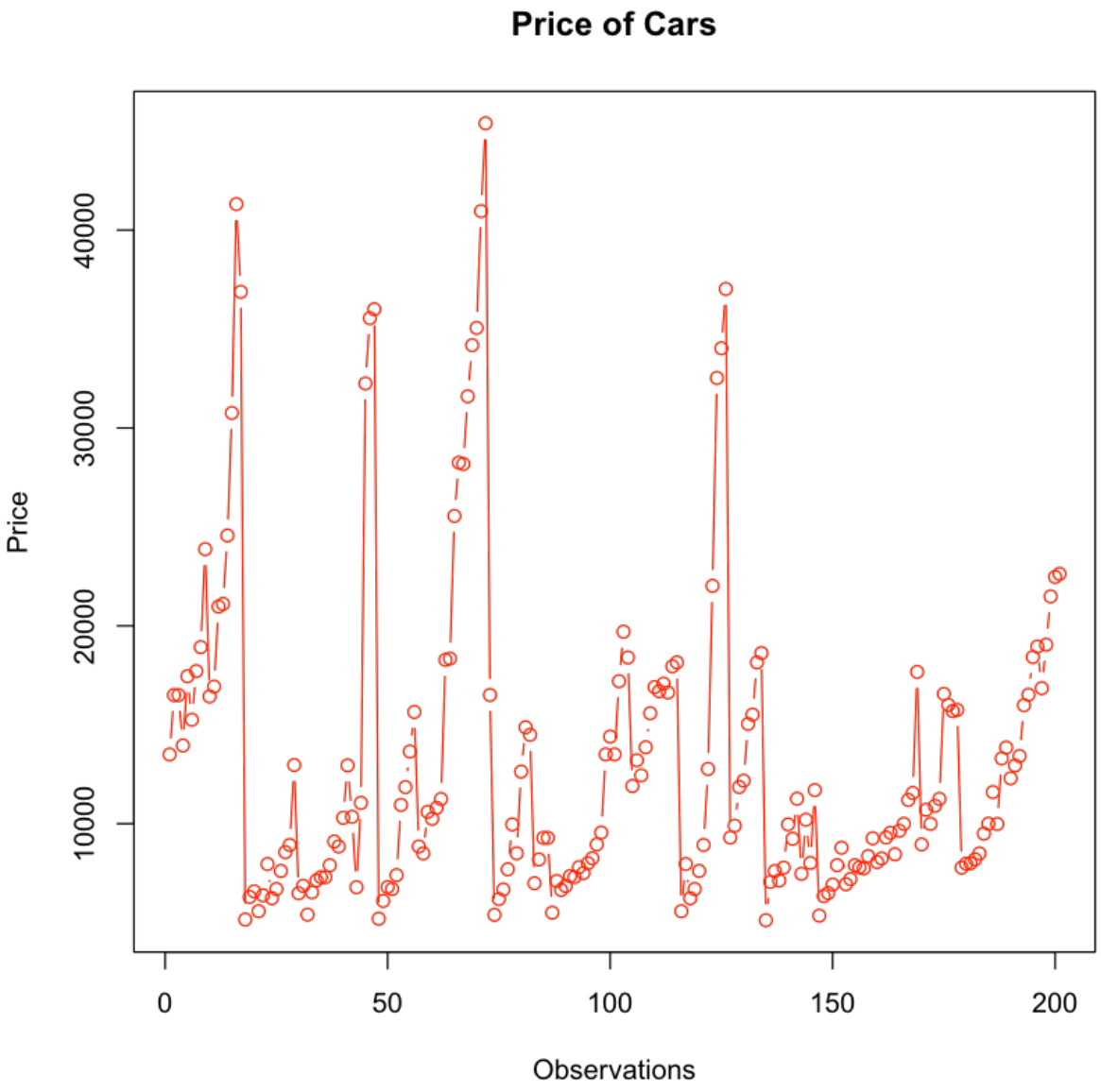
```
:2.070
 five  : 10       1st Qu.: 98.0   2bbl   :64   1st Qu.:3.150   1st Q
u.:3.110
 four  :157       Median :120.0   idi    :20   Median :3.310   Media
n :3.290
 six   : 24       Mean   :126.9   1bbl   :11   Mean   :3.331   Mean
:3.257
 three :  1       3rd Qu.:141.0   spdi   : 9   3rd Qu.:3.580   3rd Q
u.:3.410
 twelve:  1       Max.   :326.0   4bbl   : 3   Max.   :3.940   Max.
:4.170
 two   :  4                       (Other): 2
 compression_ratio   horsepower       peak_rpm       city_mpg
 Min.   : 7.00     Min.   : 48.0   Min.   :4150   Min.   :13.00
 1st Qu.: 8.60     1st Qu.: 70.0   1st Qu.:4800   1st Qu.:19.00
 Median : 9.00     Median : 95.0   Median :5125   Median :24.00
 Mean   :10.16     Mean   :103.4   Mean   :5118   Mean   :25.18
 3rd Qu.: 9.40     3rd Qu.:116.0   3rd Qu.:5500   3rd Qu.:30.00
 Max.   :23.00     Max.   :262.0   Max.   :6600   Max.   :49.00

  highway_mpg        price
 Min.   :16.00   Min.   : 5118
 1st Qu.:25.00   1st Qu.: 7775
 Median :30.00   Median :10295
 Mean   :30.69   Mean   :13207
 3rd Qu.:34.00   3rd Qu.:16500
 Max.   :54.00   Max.   :45400
```

Univariate Plots

Points and Lines Plot

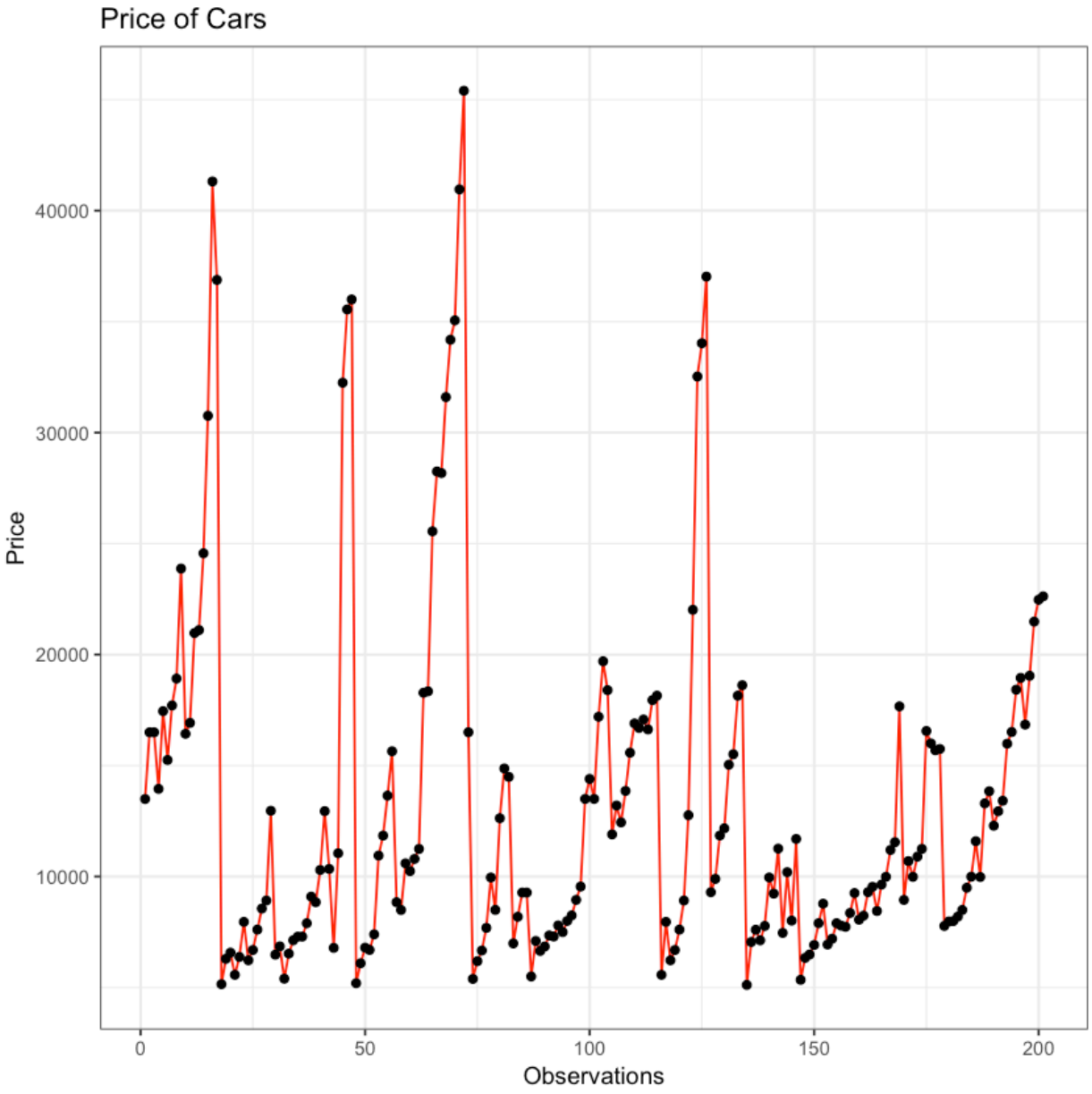
In [15]:

```
# Points and lines plot
 plot(df$price, type= "b",
      main = "Price of Cars", xlab ='Observations', ylab = 'Price', col = 'red')
```


Price of Cars
Price
10000
20000
30000
40000
0
50
100
150
200
Observations

In [16]:

```
# Point Lines plot
ggplot(df, aes(x = 1:201, y=price)) +  geom_line(color = 'red') +
       geom_point()+
       theme_bw()+
       labs(x = 'Observations', y = 'Price', title = 'Price of Cars' )
```



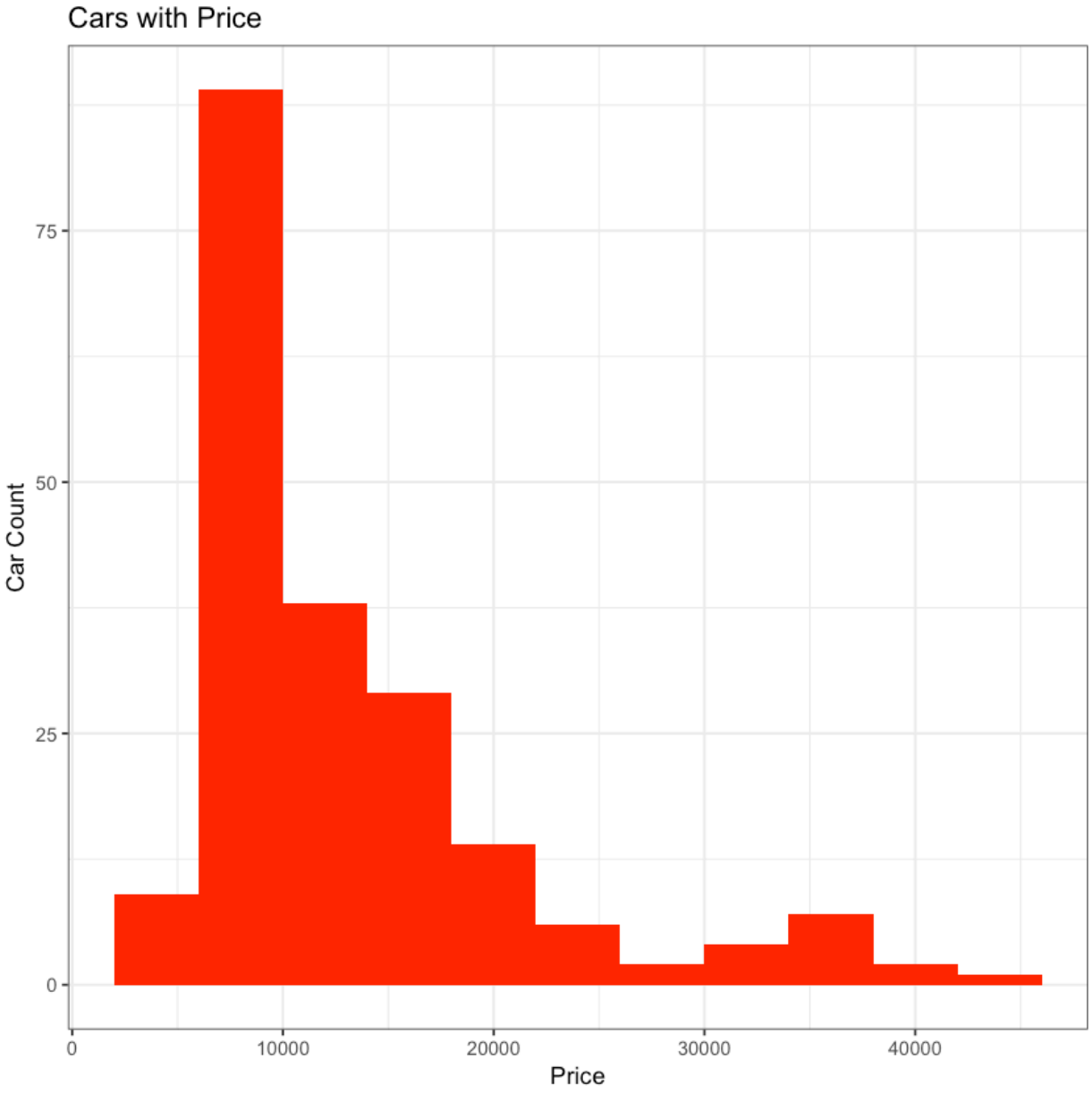
In [17]:

```
# Line Plot
plot(df$price, type = 'h',
     main = "Price of Cars", xlab ='Observations', ylab = 'Price', col = 'red')
```

Histogram

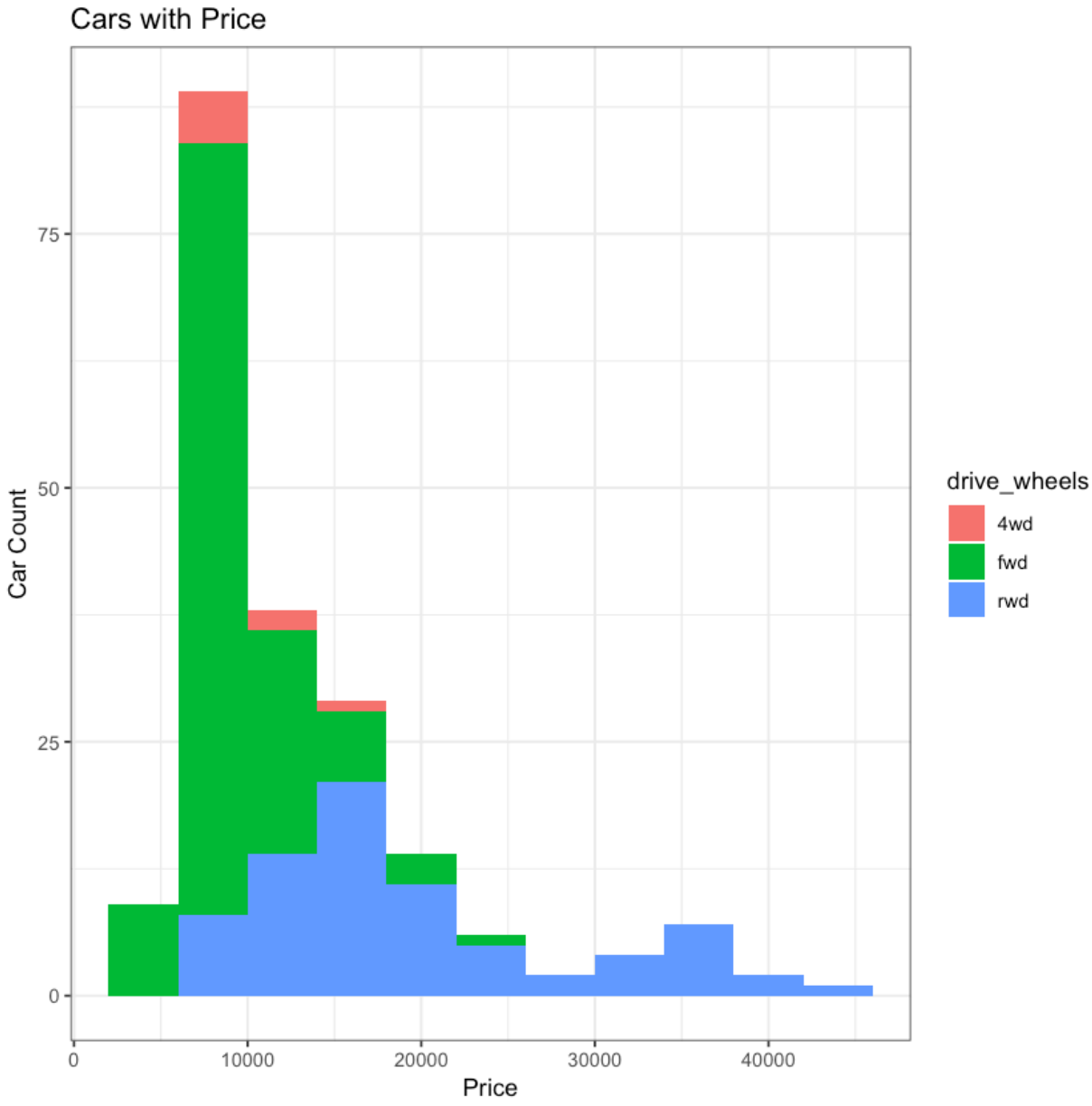
In [18]:

```
# Simple HISTOGRAM
ggplot(df, aes(x = price)) +  geom_histogram(fill = 'red', binwidth = 4000) +
       theme_bw()+
       labs(x = 'Price', y = 'Car Count', title = 'Cars with Price' )
```



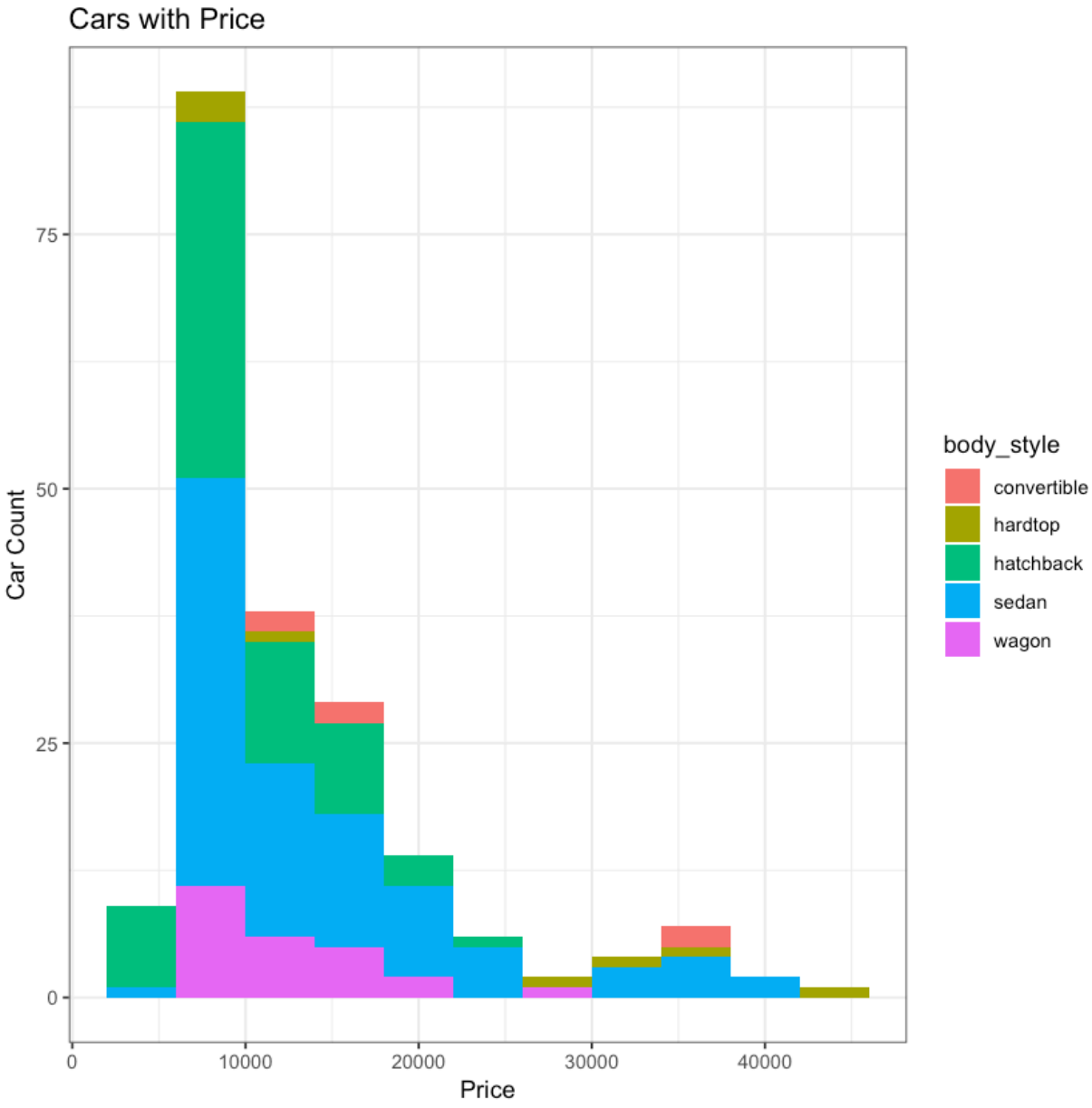
In [19]:

```
# HISTOGRAM with Category
ggplot(df, aes(x = price, fill= drive_wheels)) +  geom_histogram(binwidth = 4000
) +
       theme_bw()+
       labs(x = 'Price', y = 'Car Count', title = 'Cars with Price' )
```



In [20]:

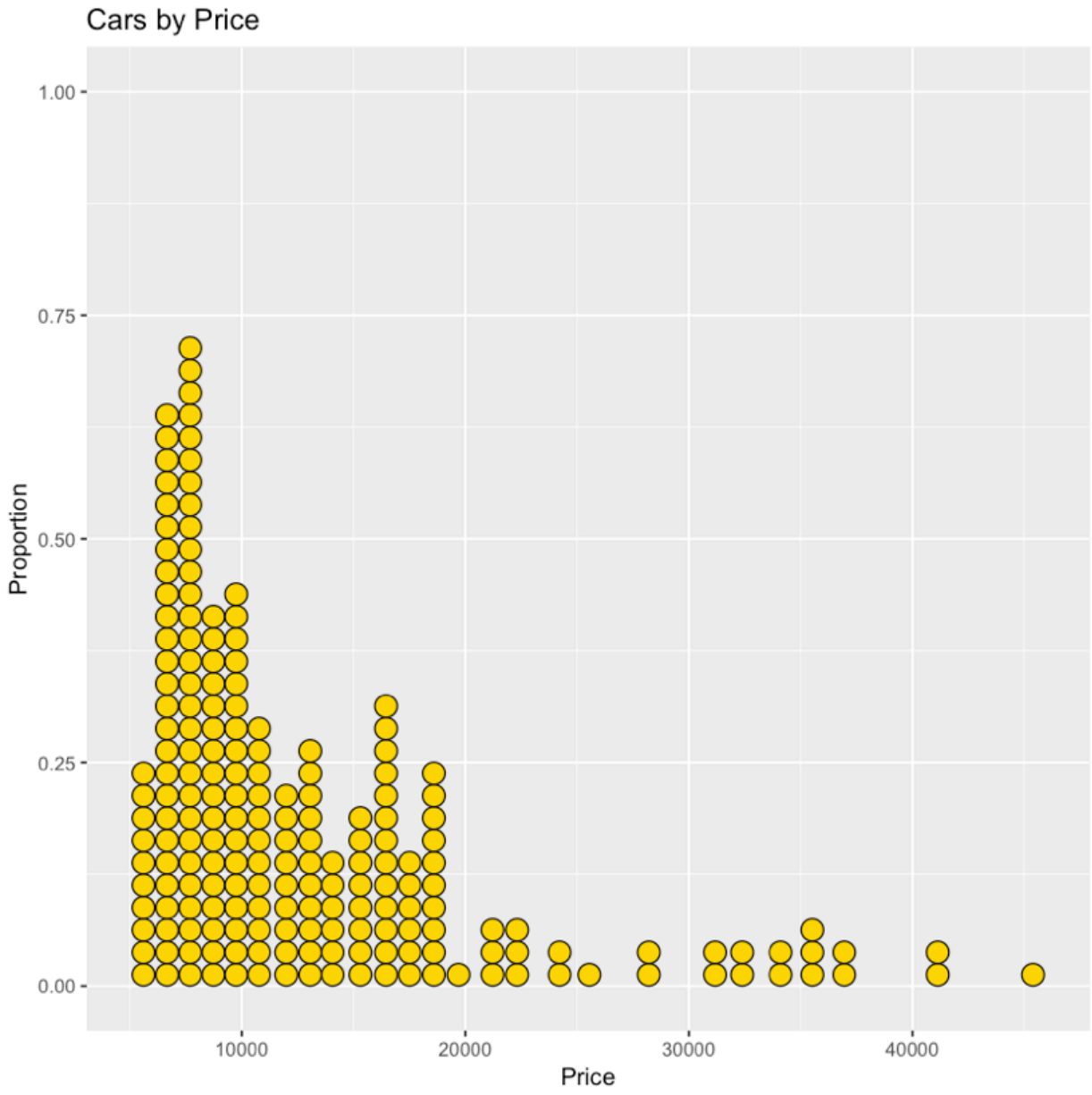
```
#HISTOGRAM with Category
ggplot(df, aes(x = price, fill= body_style)) +  geom_histogram(binwidth = 4000)
+
       theme_bw()+
       labs(x = 'Price', y = 'Car Count', title = 'Cars with Price' )
```



Dot Plot

In [21]:

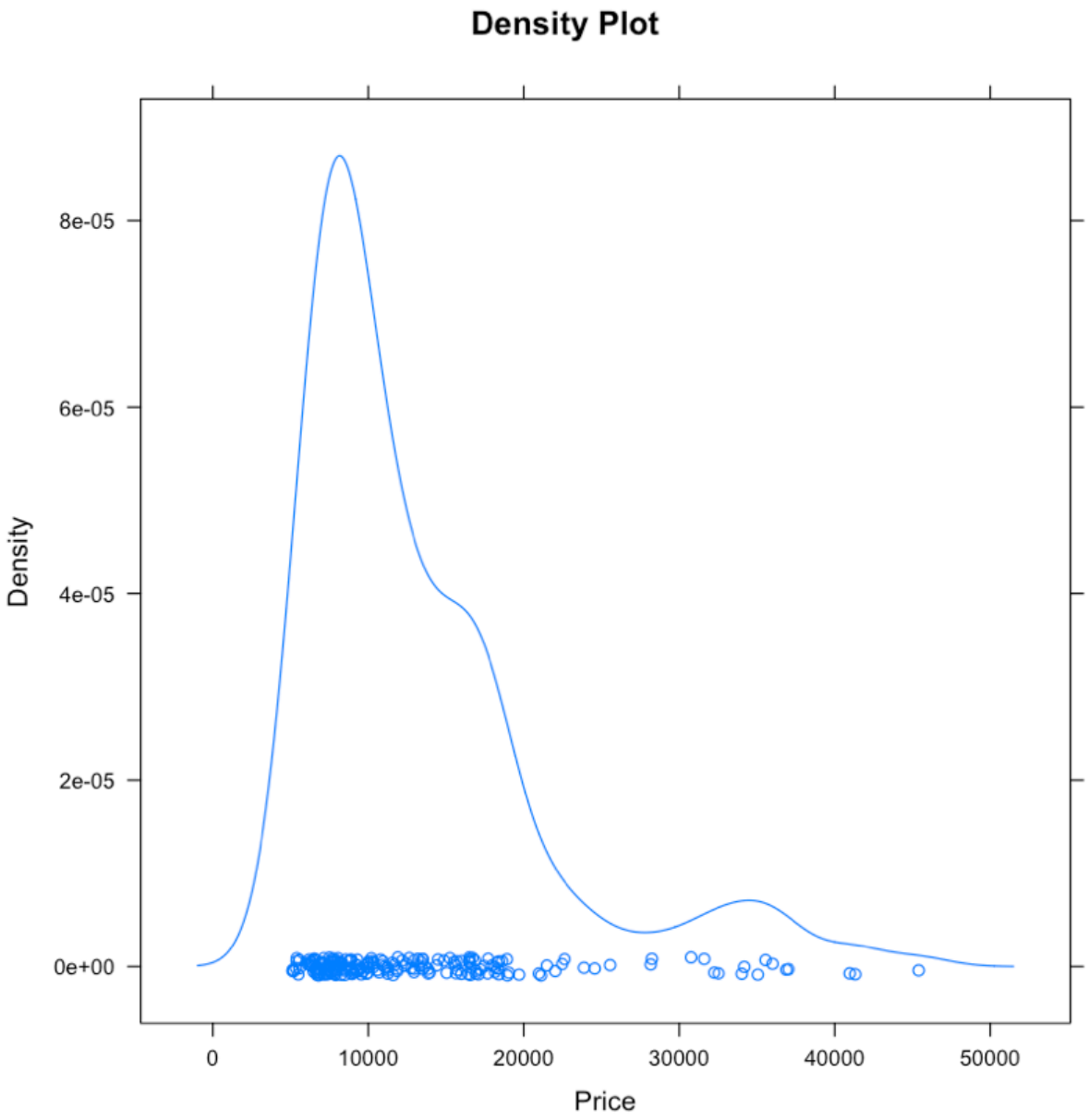
```
# Dot plot
ggplot(df, aes(x = price)) +
       geom_dotplot(fill = "gold", color = "black", binwidth = 1000) +
       labs(title = "Cars by Price", y = "Proportion", x = "Price")
```



Density Plot

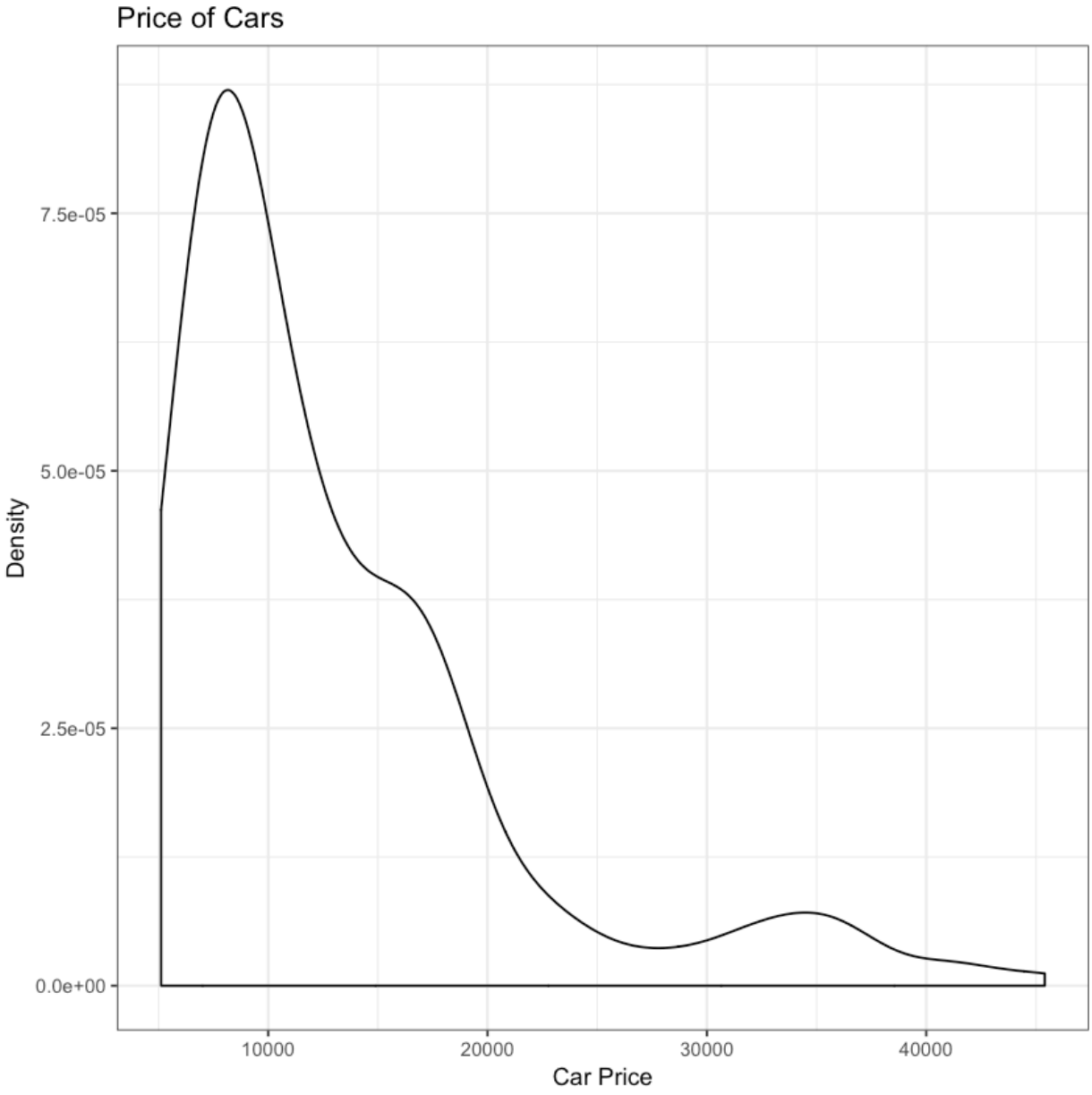
In [22]:

```
# Density plot
densityplot(~df$price, main="Density Plot",  xlab="Price")
```



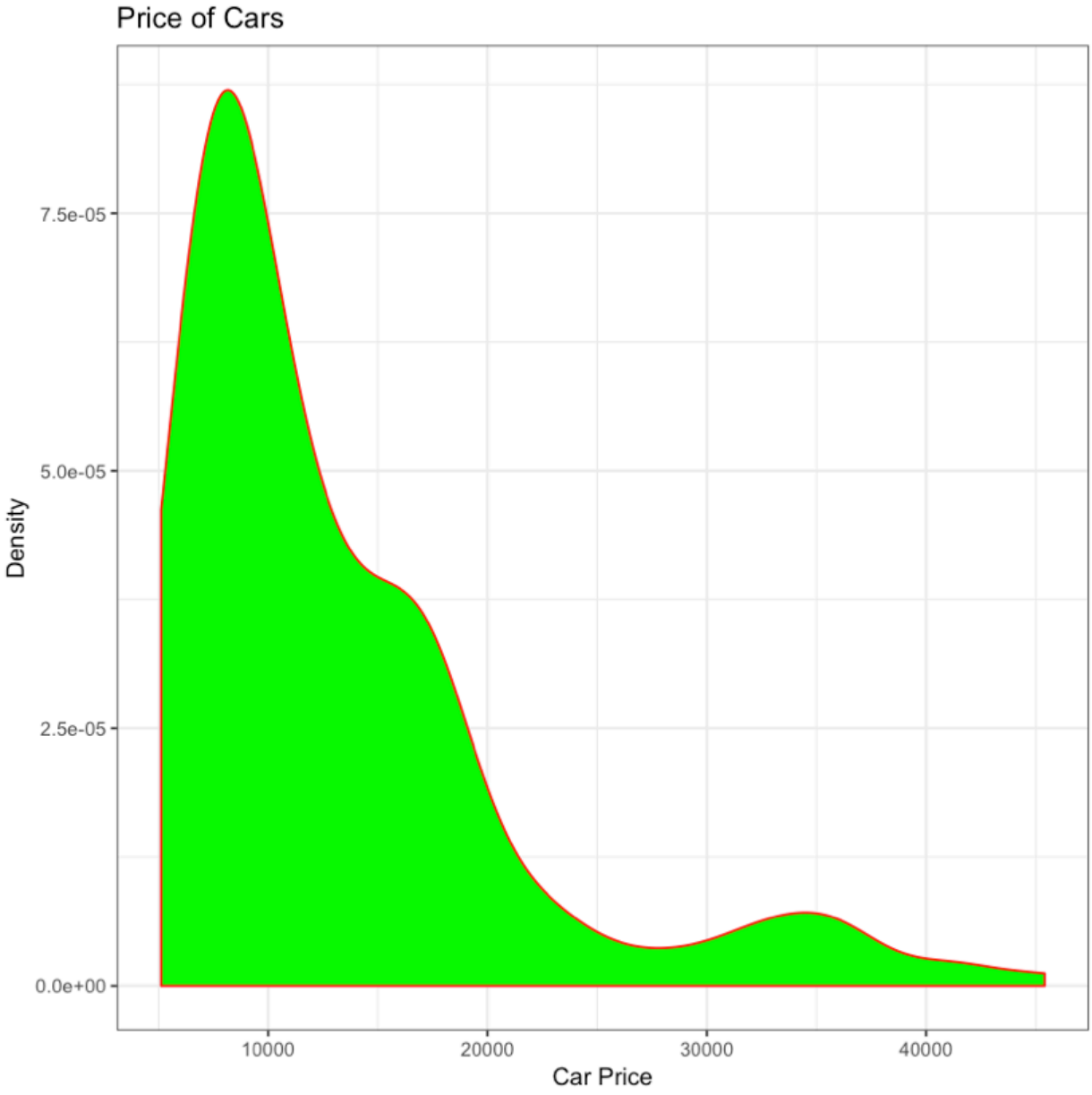
In [23]:

```
# Density plot
ggplot(df, aes(x = price)) +  geom_density() +
       theme_bw()+
       labs(x = 'Car Price', y = 'Density', title = 'Price of Cars' )
```



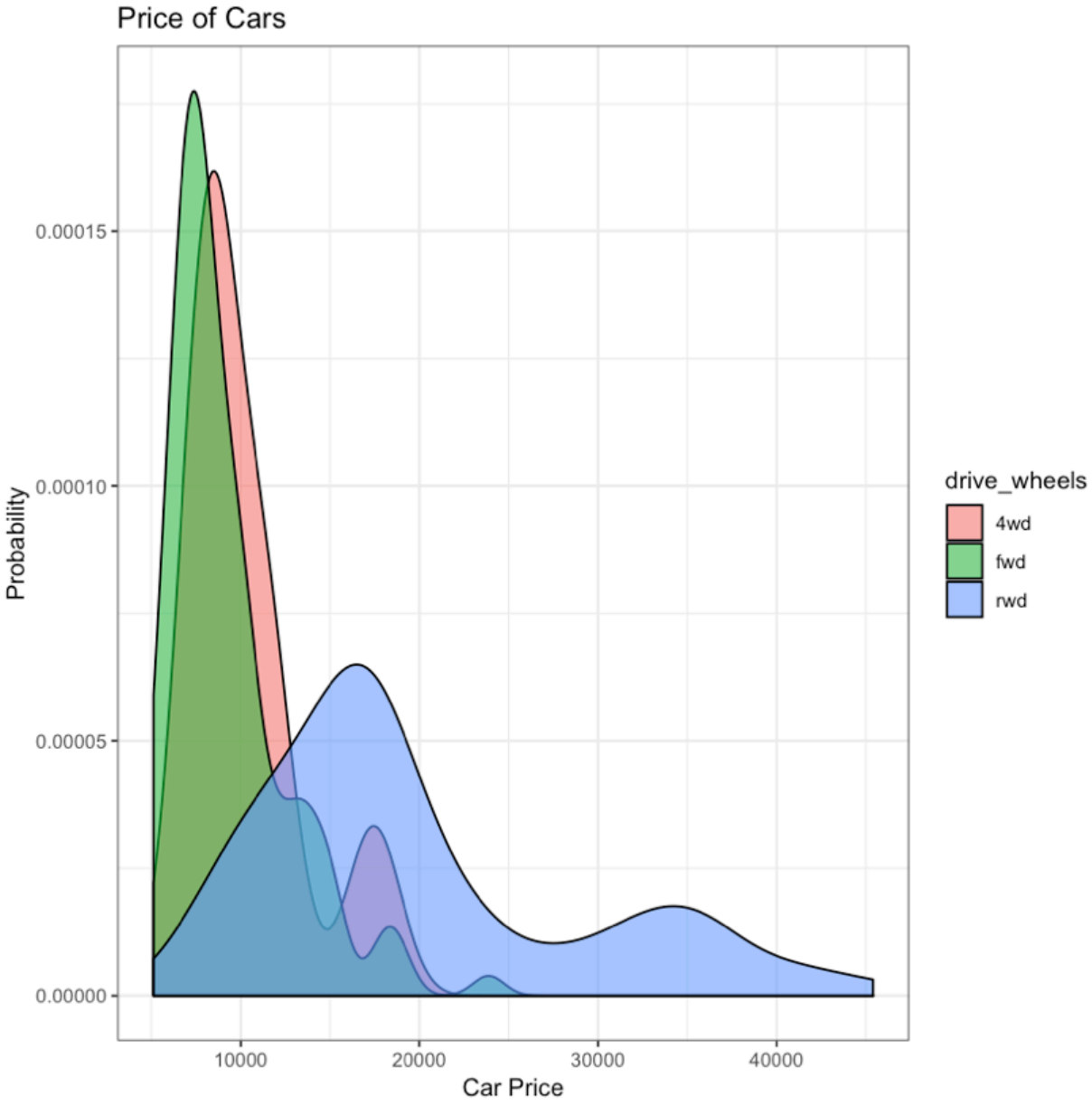
In [24]:

```
# Density plot
ggplot(df, aes(x = price)) +  geom_density(color = 'red', fill = 'green') +
       theme_bw()+
       labs(x = 'Car Price', y = 'Density', title = 'Price of Cars' )
```



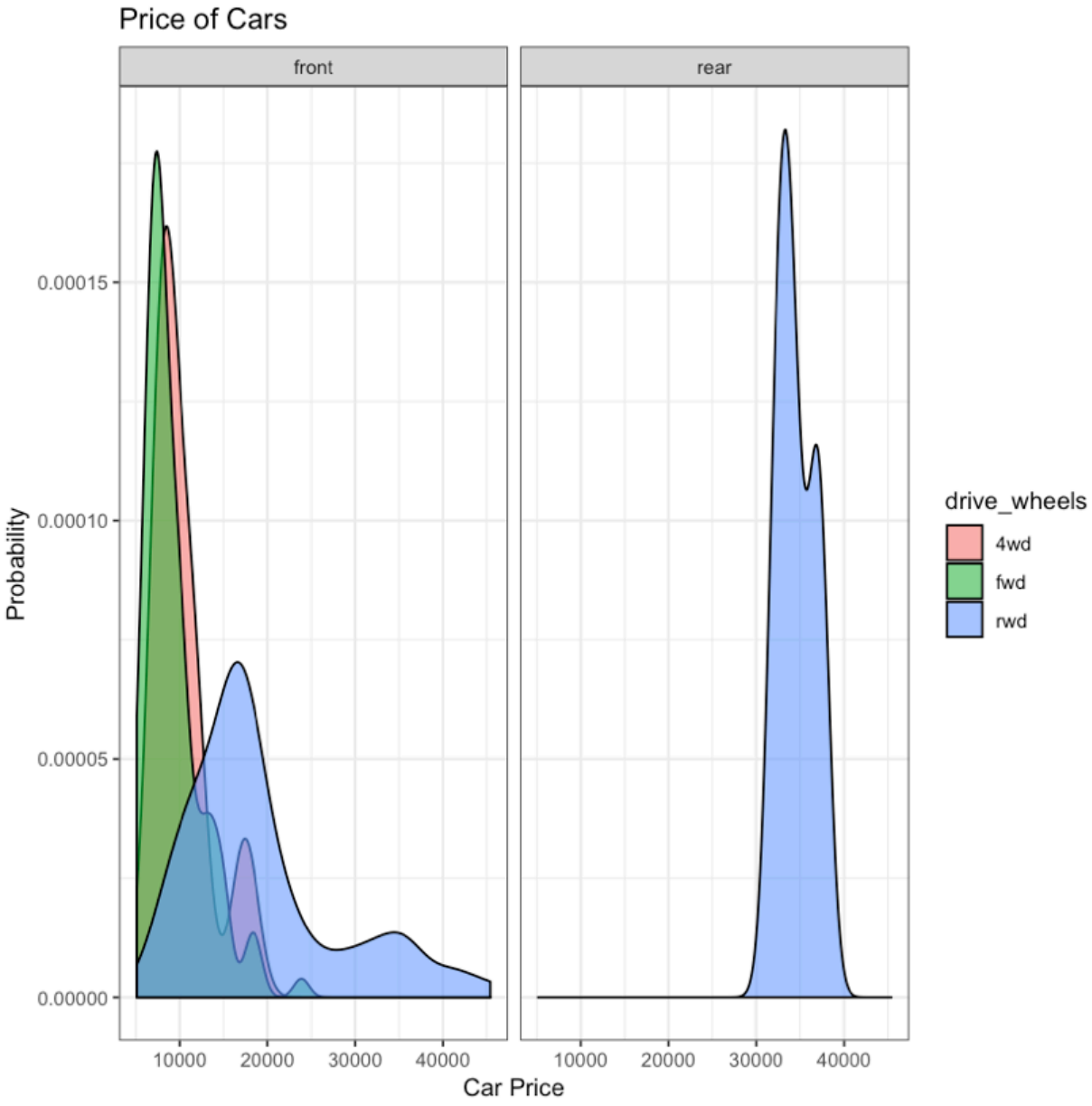
In [25]:

```
# Density plot fill by another categorical variable
ggplot(df, aes(x = price, fill = drive_wheels)) +  geom_density(alpha = 0.6) +
       theme_bw()+
       labs(x = 'Car Price', y = 'Probability', title = 'Price of Cars' )
```



In [26]:

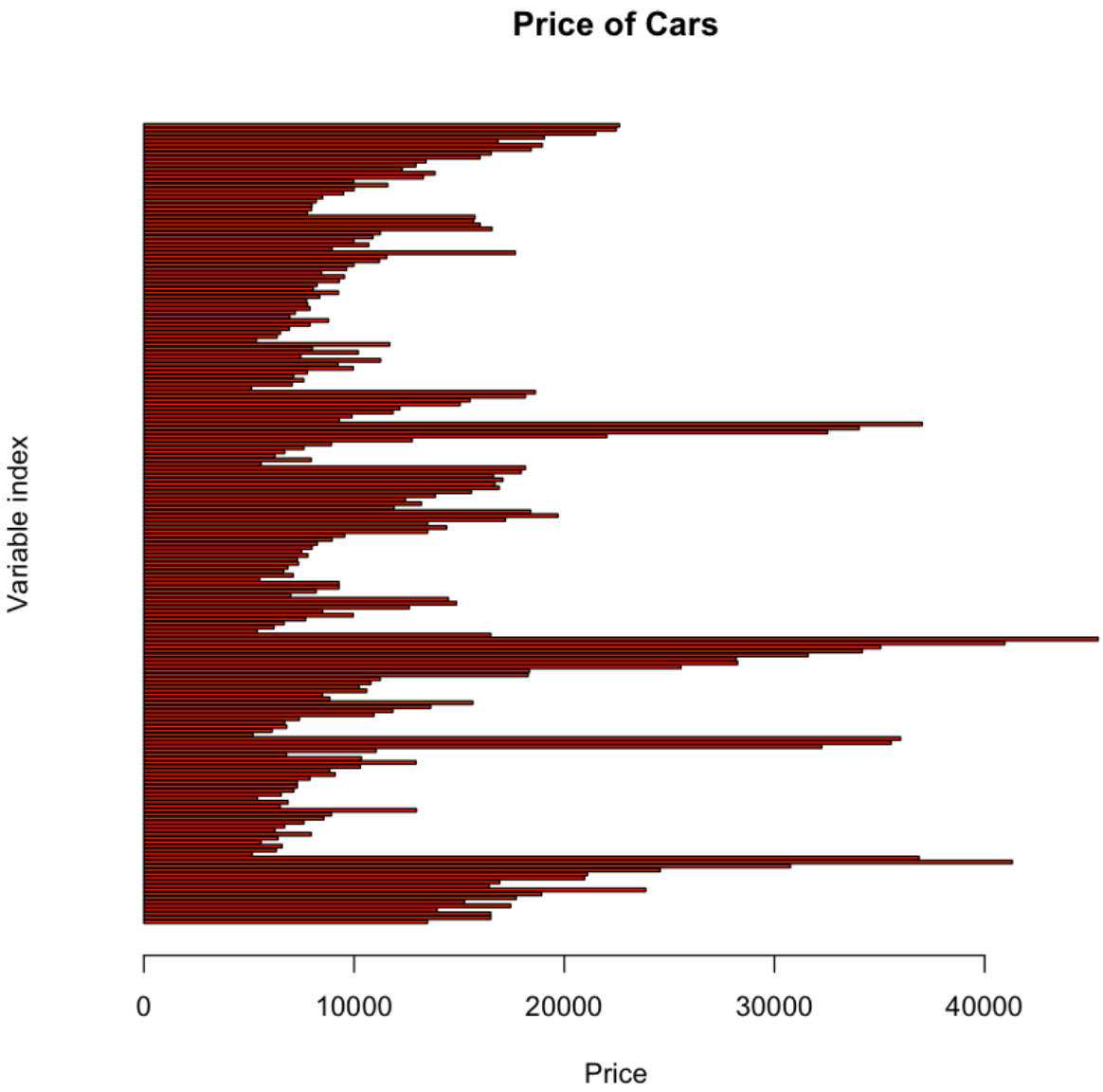
```
# Density plot fill by another categorical variable
ggplot(df, aes(x = price, fill = drive_wheels)) +  geom_density(alpha = 0.6) +
       theme_bw()+
       facet_wrap(~engine_location)+
       labs(x = 'Car Price', y = 'Probability', title = 'Price of Cars' )
```



Bar Plot

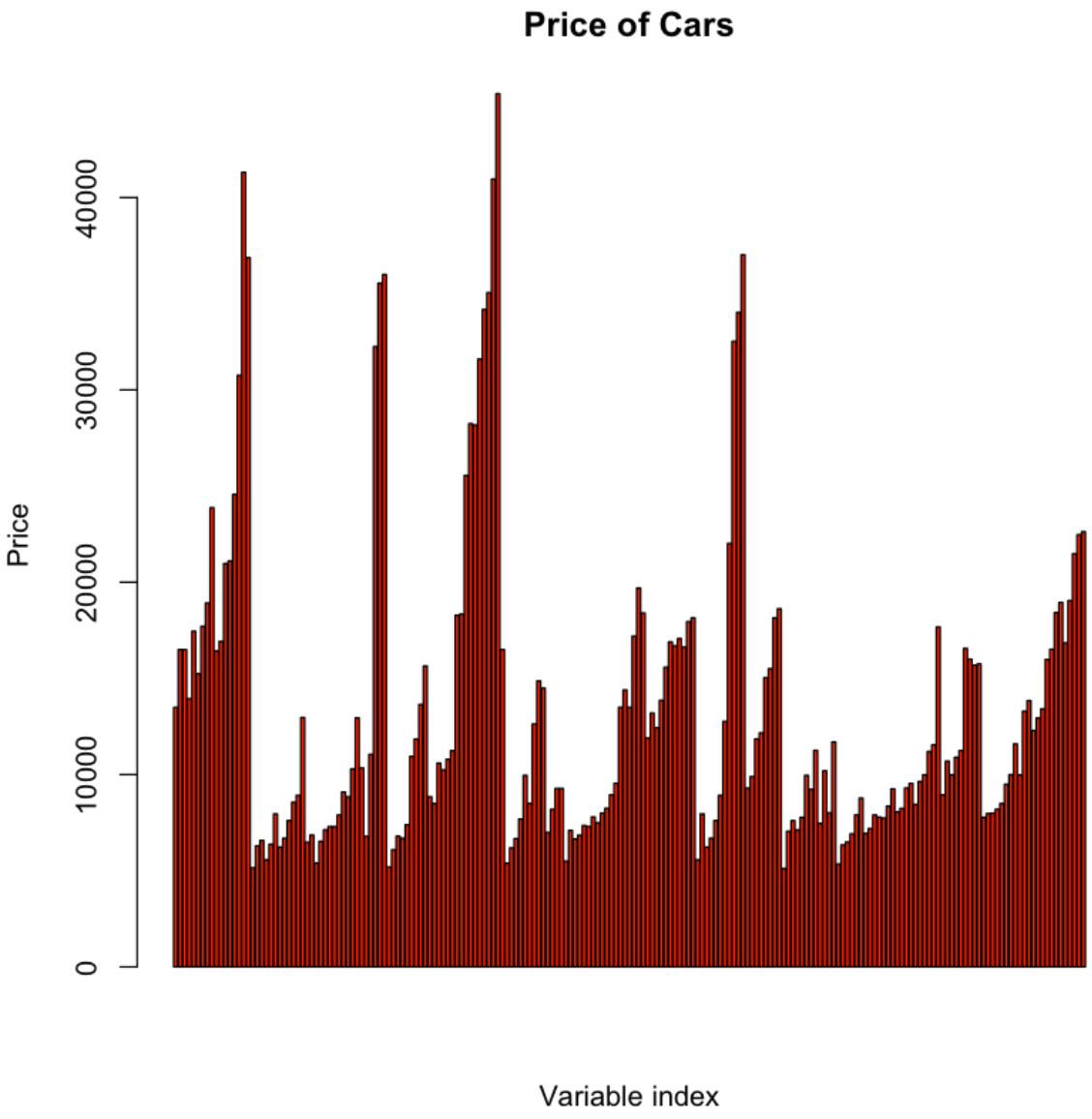
In [27]:

```
# Simple Bar plot
barplot(df$price, ylab = 'Variable index', xlab = 'Price',
    main = 'Price of Cars', col = 'red', horiz = TRUE)
```



In [28]:

```
# Bar plot
barplot(df$price, xlab = 'Variable index', ylab = 'Price',
    main = 'Price of Cars', col = 'red', horiz = FALSE)
```



In [29]:

```
#Creating a frequency table using count function in dplyr
ftable = count(df, body_style)
ftable
```

body_style	n
convertible	6
hardtop	8
hatchback	68
sedan	94
wagon	25

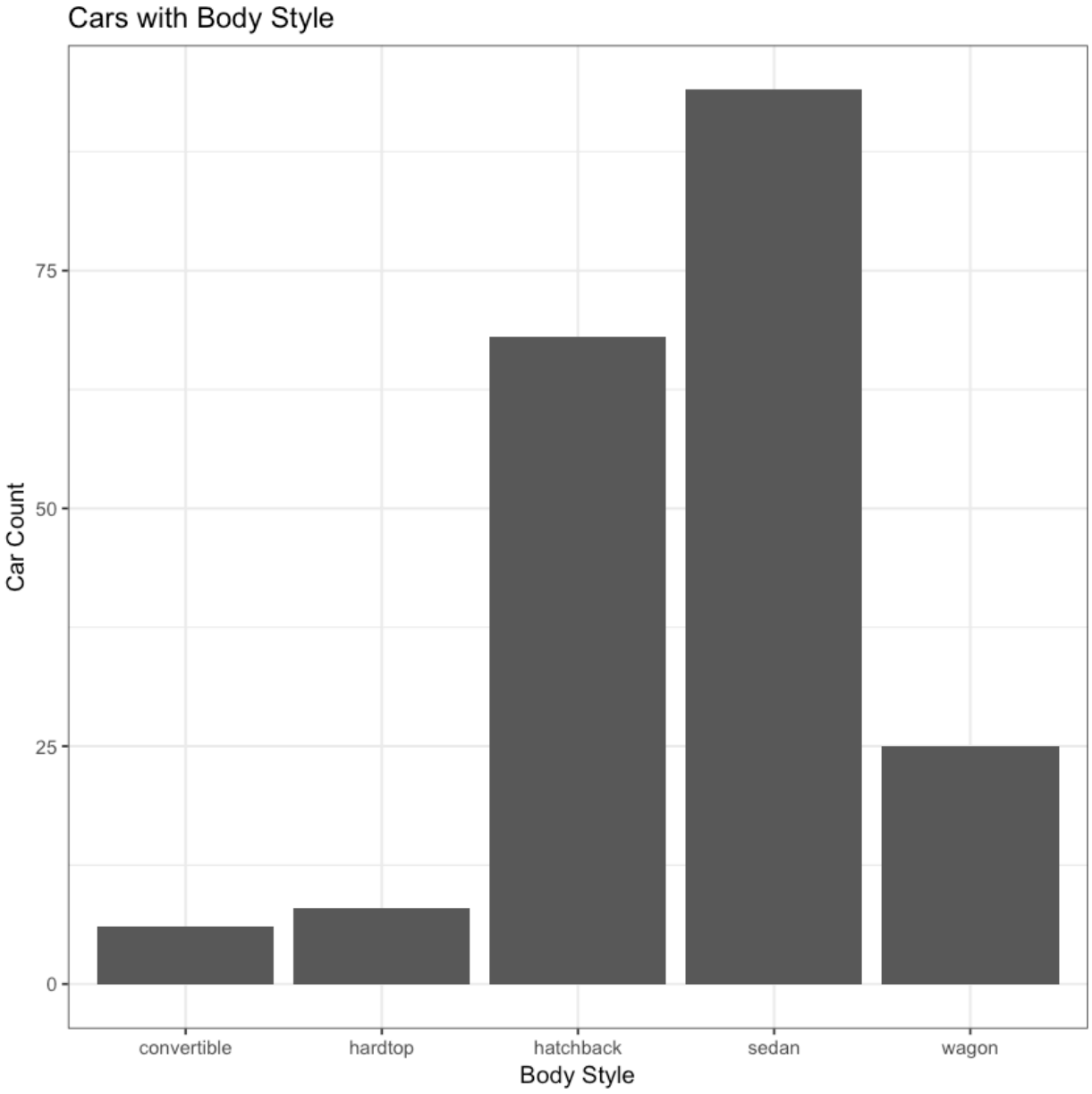
In [30]:

```
dim(ftable)
```

5 2

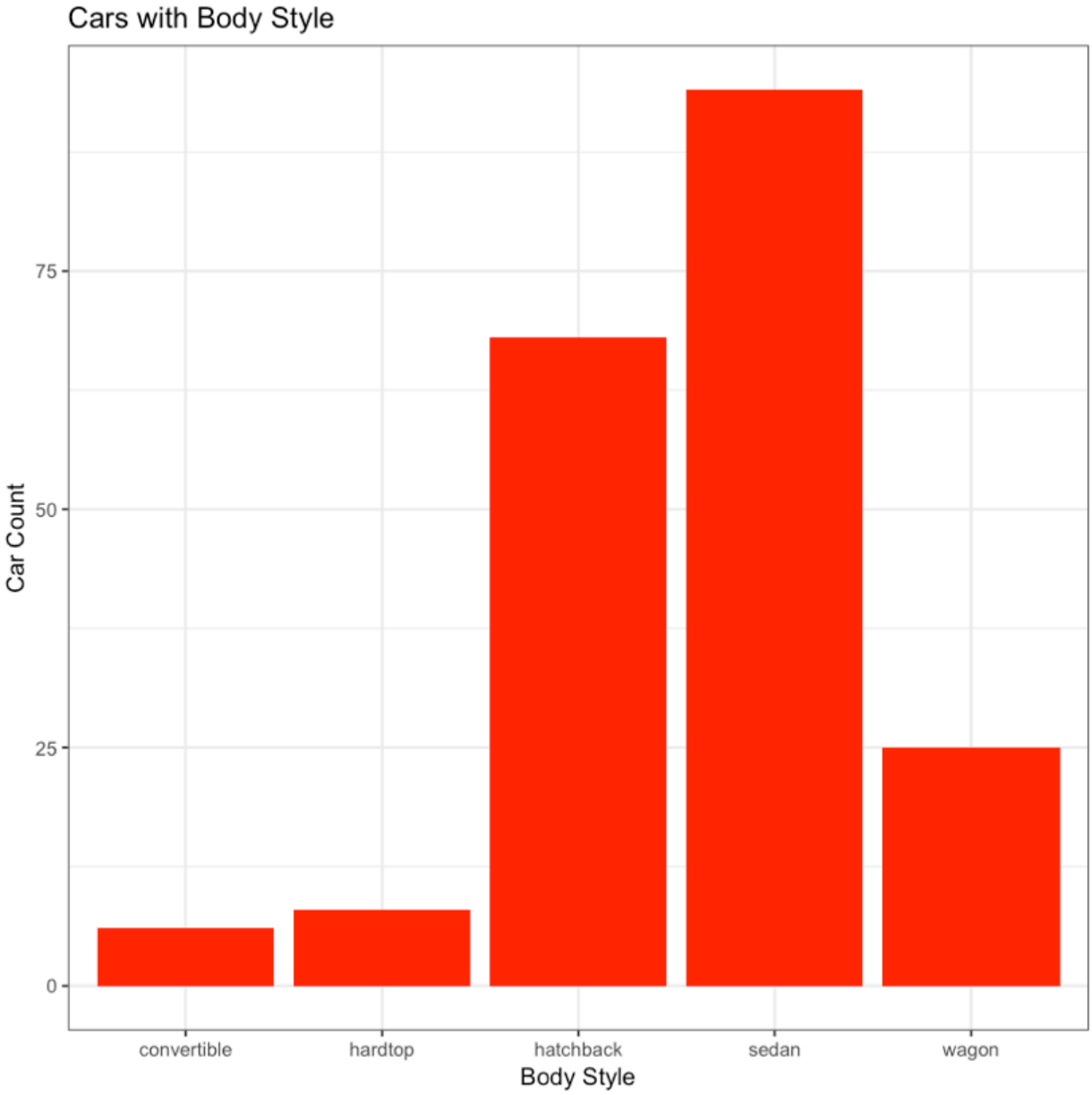
In [31]:

```
# Bar plot
ggplot(df, aes(x = body_style)) +  geom_bar() +
       theme_bw()+
       labs(x = 'Body Style', y = 'Car Count', title = 'Cars with Body Style' )
```


Cars with Body Style
Car Count
75
50
25
0
convertible
hardtop
hatchback
sedan
wagon
Body Style

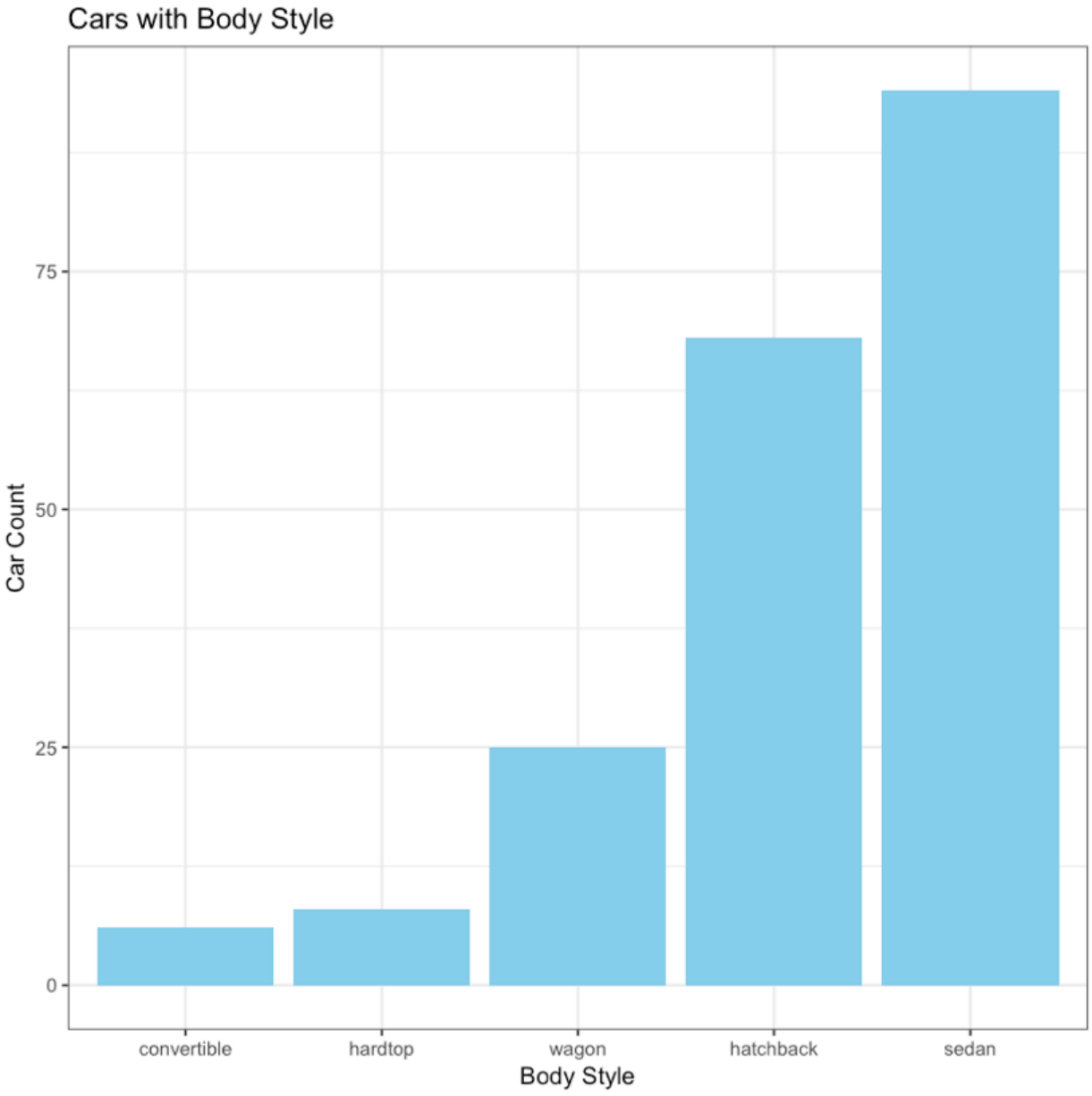
In [32]:

```
# Bar plot
ggplot(df, aes(x = body_style)) +  geom_bar(fill = 'red') +
       theme_bw()+
       labs(x = 'Body Style', y = 'Car Count', title = 'Cars with Body Style' )
```



In [33]:

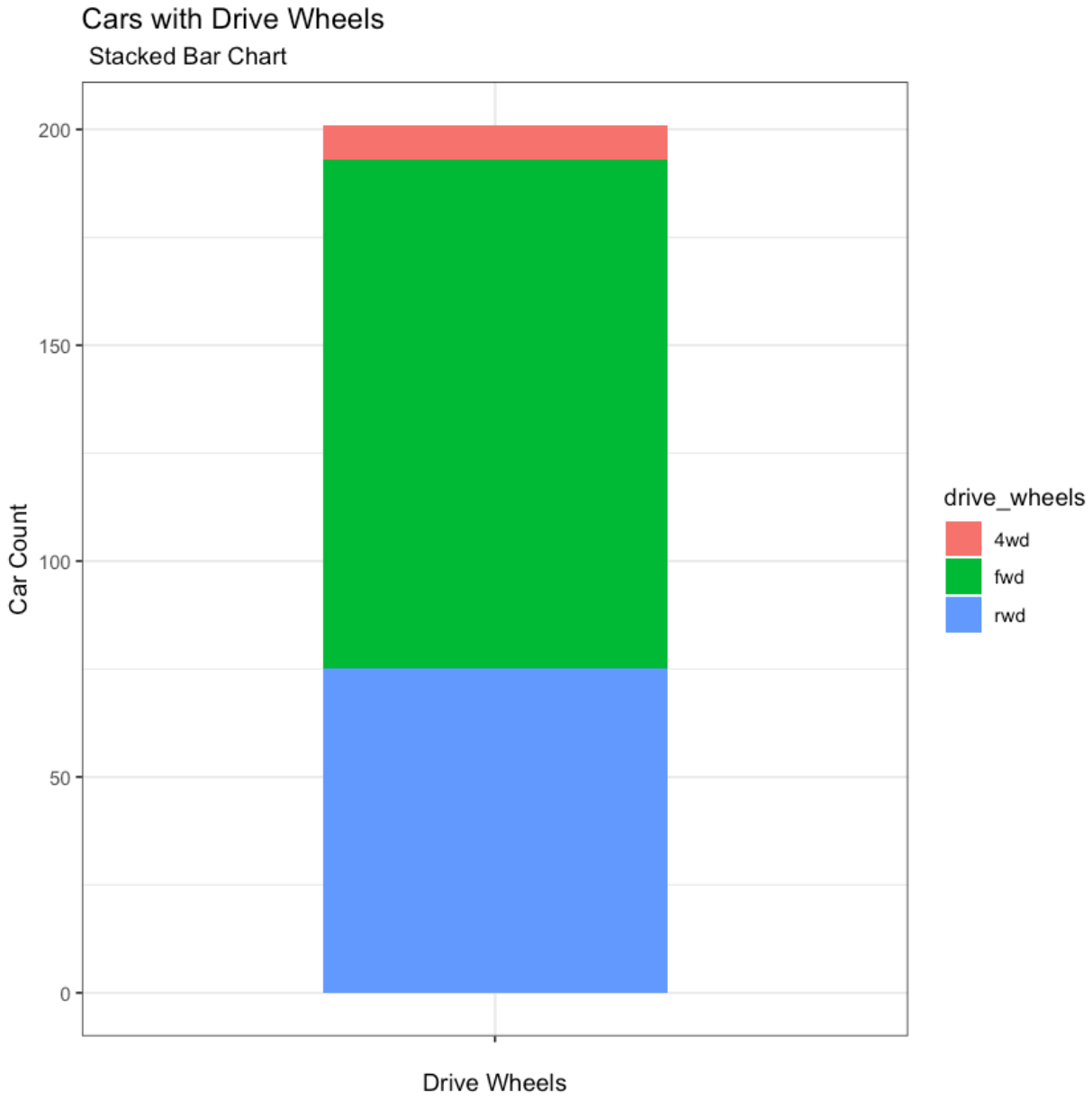
```
# plot the bars in ascending order
ggplot(ftable, aes(x = reorder(body_style, n),  y = n)) +
       geom_bar(stat = "identity", fill = 'skyblue') +
       theme_bw()+
       labs(x = 'Body Style', y = 'Car Count', title = 'Cars with Body Style' )
```



Stacked Bar plot

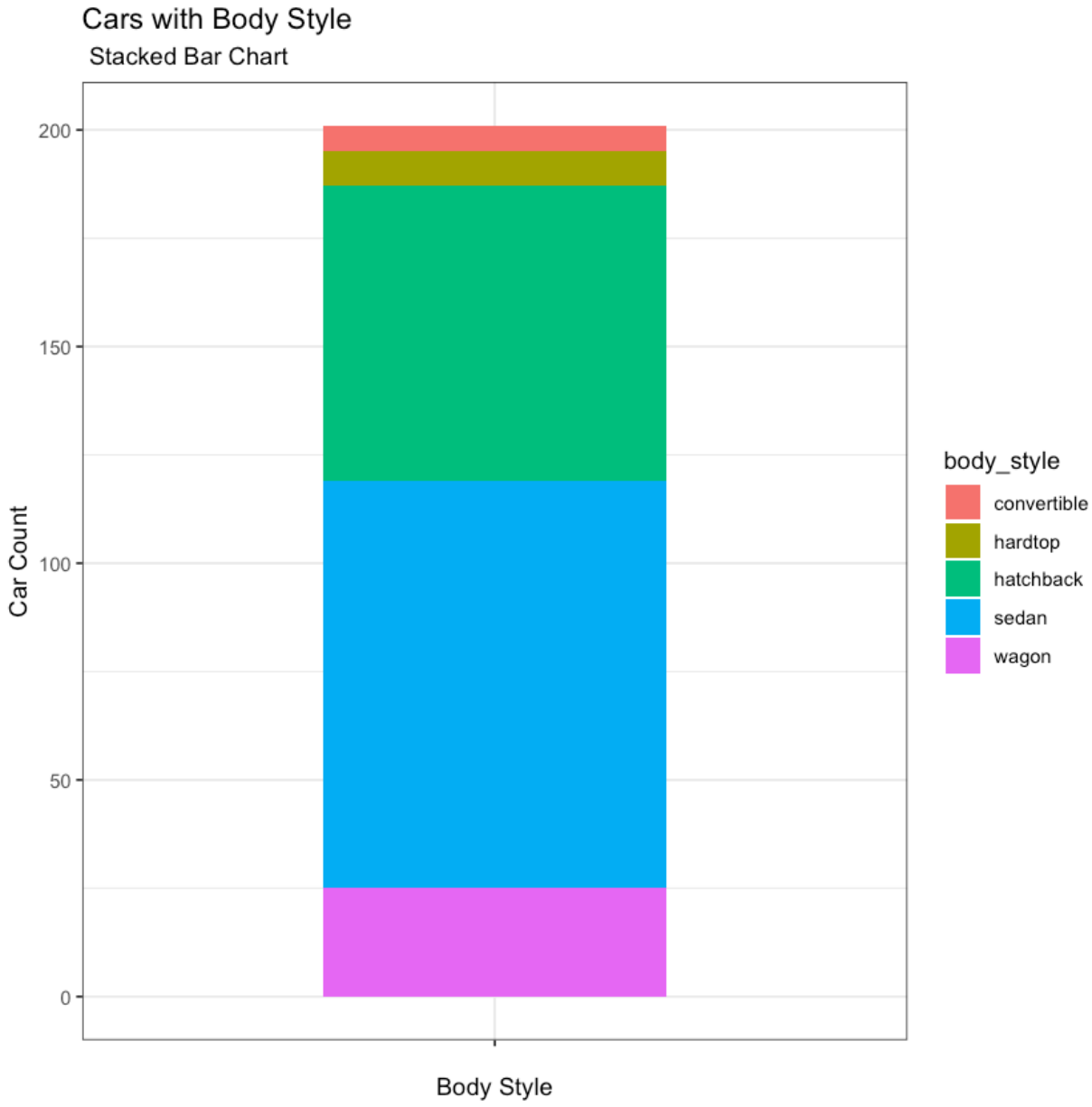
In [34]:

```
# Stacked Bar plot fill by another categorical variable
ggplot(df, aes(x = "", fill = drive_wheels)) +  geom_bar(width = 0.5) +
       theme_bw()+
       labs(x = 'Drive Wheels', y = 'Car Count',
            title = 'Cars with Drive Wheels', subtitle =' Stacked Bar Chart')
```



In [35]:

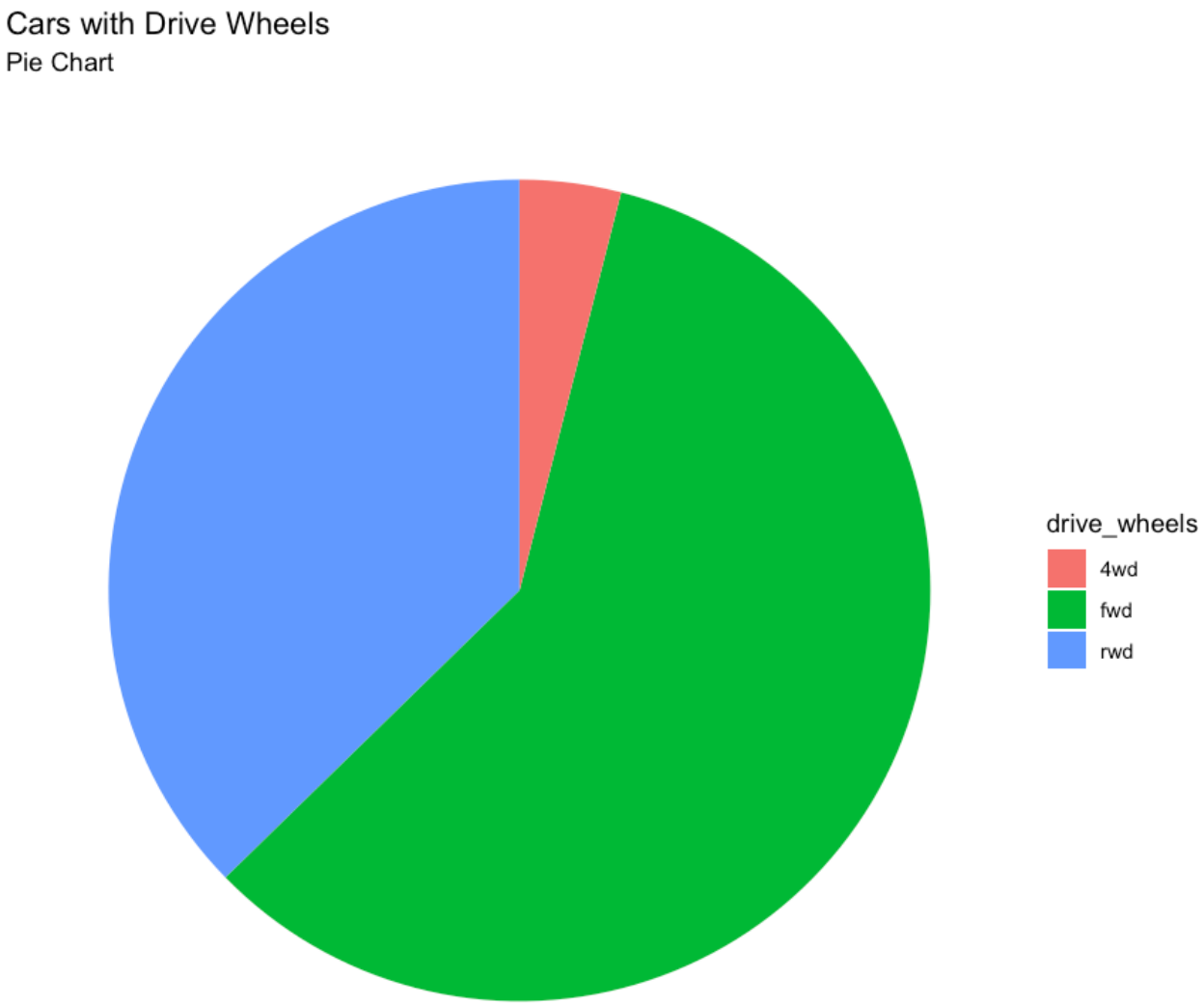
```
# Stacked Bar plot fill by another categorical variable
ggplot(df, aes(x = "", fill = body_style)) +  geom_bar(width = 0.5) +
       theme_bw()+
       labs(x = 'Body Style', y = 'Car Count',
            title = 'Cars with Body Style', subtitle =' Stacked Bar Chart')
```



Pie Chart

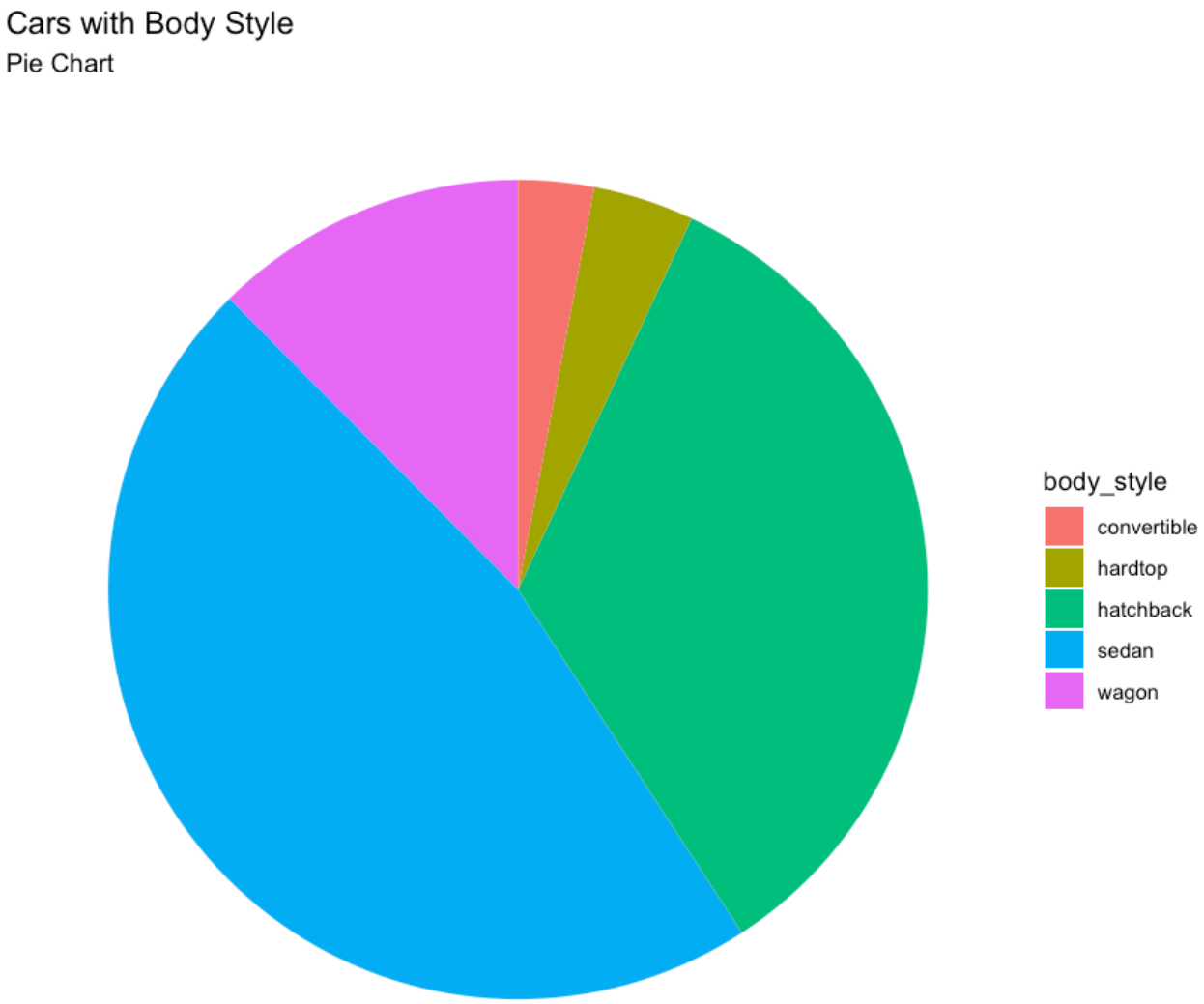
In [36]:

```
# Creating Pie chart
ggplot(df, aes(x=factor(3), fill= drive_wheels))+
  geom_bar()+
  coord_polar("y", start = 0,   direction = -1)+
  theme_void()+
  labs(x = '', y = '',
       title = 'Cars with Drive Wheels', subtitle ='Pie Chart')
```



In [37]:

```
# Creating Pie chart as bar in polar coordinates
ggplot(df, aes(x=factor(5), fill= body_style))+
  geom_bar()+
  coord_polar("y",start = 0,   direction = -1)+
  theme_void()+
  labs(x = '', y = '',
       title = 'Cars with Body Style', subtitle ='Pie Chart')
```



In [38]:

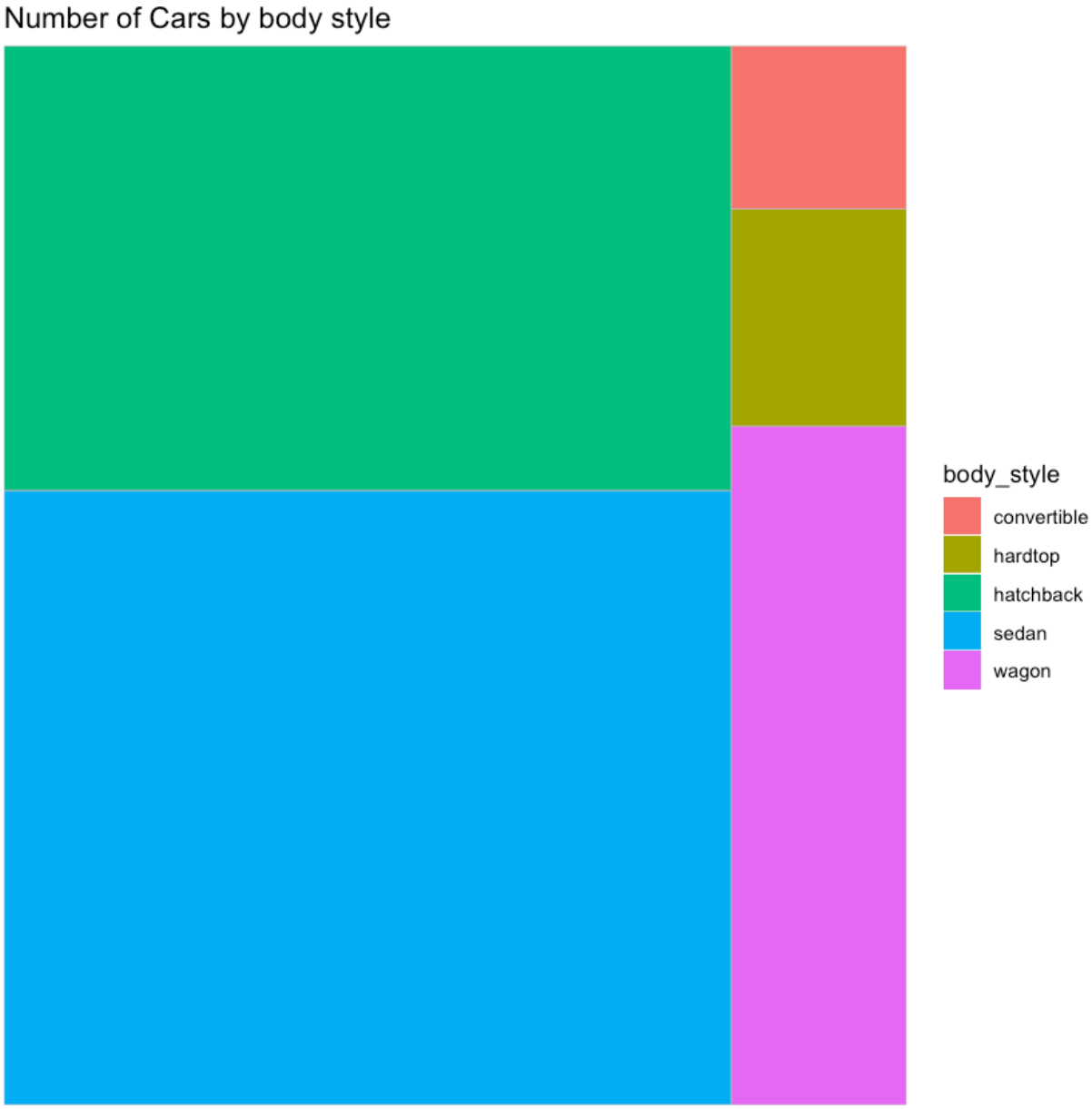
```
ftable
```

body_style	n
convertible	6
hardtop	8
hatchback	68
sedan	94
wagon	25

Tree Map

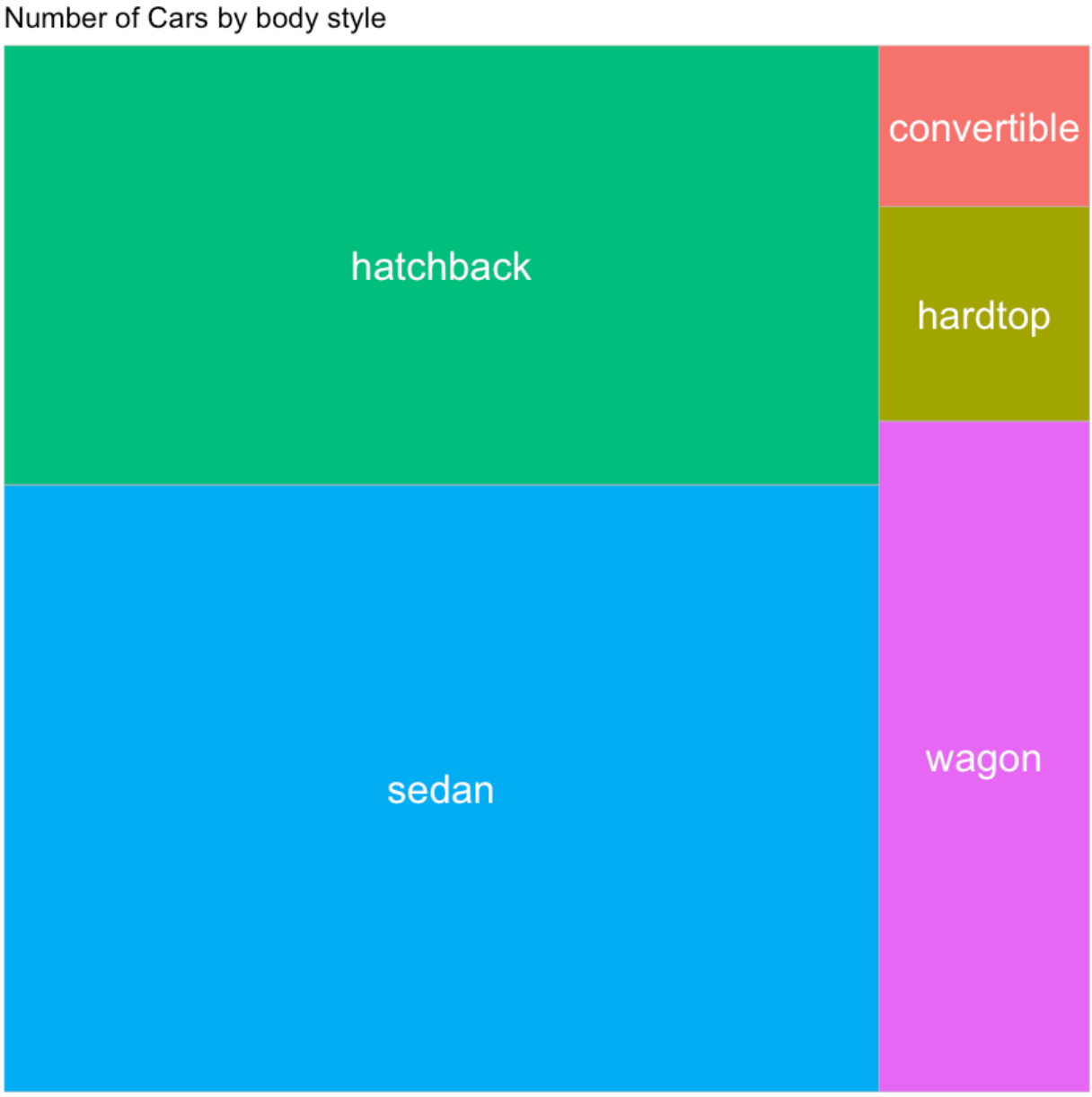
In [39]:

```
# create a treemap
# area = numeric , fill = categorical
ggplot(ftable, aes(area = n, fill = body_style)) +
       geom_treemap() +
       labs(title = "Number of Cars by body style")
```


Number of Cars by body style
body_style
convertible
hardtop
hatchback
sedan
wagon

In [40]:

```
# create a treemap with tile labels
# area = numeric , fill = categorical
ggplot(ftable, aes(area = n, fill = body_style, label = body_style)) +
       geom_treemap() +
       geom_treemap_text(colour = "white",  place = "centre") +
       theme(legend.position = "none")+
       labs(title = "Number of Cars by body style")
```



Bivariate Graphs

In [41]:

```
# Normalizing columns "length", "width" and "height" using scale() function
df$length<-scale(df$length, center = TRUE, scale = TRUE)
df$width<-scale(df$width, center = TRUE, scale = TRUE)
df$height<-scale(df$height, center = TRUE, scale = TRUE)

df[1:5,]
```

symboling	normalized_losses	make	fuel_type	aspiration	num_of_doors	body_style	drive_w
3	122	alfa-romero	gas	std	two	convertible	
3	122	alfa-romero	gas	std	two	convertible	
1	122	alfa-romero	gas	std	two	hatchback	
2	164	audi	gas	std	four	sedan	
2	164	audi	gas	std	four	sedan	

In [42]:

```
#Getting structure of data frame - name, type and preview of data in each column
str(df)
```

```
'data.frame':   201 obs. of  26 variables:
 $ symboling        : int  3 3 1 2 2 2 1 1 1 2 ...
 $ normalized_losses: num  122 122 122 164 164 122 158 122 158 192 .
..
 $ make             : Factor w/ 22 levels "alfa-romero",..: 1 1 1 2
2 2 2 2 2 3 ...
 $ fuel_type        : Factor w/ 2 levels "diesel","gas": 2 2 2 2 2 2
2 2 2 2 ...
 $ aspiration       : Factor w/ 2 levels "std","turbo": 1 1 1 1 1 1
1 1 2 1 ...
 $ num_of_doors     : Factor w/ 3 levels "","four","two": 3 3 3 2 2
3 2 2 2 3 ...
 $ body_style       : Factor w/ 5 levels "convertible",..: 1 1 3 4 4
4 4 5 4 4 ...
 $ drive_wheels     : Factor w/ 3 levels "4wd","fwd","rwd": 3 3 3 2
1 2 2 2 2 3 ...
 $ engine_location  : Factor w/ 2 levels "front","rear": 1 1 1 1 1 1
1 1 1 1 ...
 $ wheel_base       : num  88.6 88.6 94.5 99.8 99.4 ...
 $ length           : num [1:201, 1] -0.438 -0.438 -0.244 0.195 0.19
5 ...
  ..- attr(*, "scaled:center")= num 174
  ..- attr(*, "scaled:scale")= num 12.3
```

```
 $ width            : num [1:201, 1] -0.851 -0.851 -0.185 0.148 0.24
3 ...
  ..- attr(*, "scaled:center")= num 65.9
  ..- attr(*, "scaled:scale")= num 2.1
 $ height           : num [1:201, 1] -2.029 -2.029 -0.558 0.218 0.21
8 ...
  ..- attr(*, "scaled:center")= num 53.8
  ..- attr(*, "scaled:scale")= num 2.45
 $ curb_weight      : int  2548 2548 2823 2337 2824 2507 2844 2954 3
086 2395 ...
 $ engine_type      : Factor w/ 7 levels "dohc","dohcv",..: 1 1 6 4
4 4 4 4 4 4 ...
 $ num_of_cylinders : Factor w/ 7 levels "eight","five",..: 3 3 4 3
2 2 2 2 2 3 ...
 $ engine_size      : int  130 130 152 109 136 136 136 136 131 108 .
..
 $ fuel_system      : Factor w/ 8 levels "1bbl","2bbl",..: 6 6 6 6 6
6 6 6 6 6 ...
 $ bore             : num  3.47 3.47 2.68 3.19 3.19 3.19 3.19 3.19 3
.13 3.5 ...
 $ stroke           : num  2.68 2.68 3.47 3.4 3.4 3.4 3.4 3.4 3.4 2.
8 ...
 $ compression_ratio: num  9 9 9 10 8 8.5 8.5 8.5 8.3 8.8 ...
 $ horsepower       : num  111 111 154 102 115 110 110 110 140 101 .
..
 $ peak_rpm         : num  5000 5000 5000 5500 5500 5500 5500 5500 5
500 5800 ...
 $ city_mpg         : int  21 21 19 24 18 19 19 19 17 23 ...
 $ highway_mpg      : int  27 27 26 30 22 25 25 25 20 29 ...
 $ price            : int  13495 16500 16500 13950 17450 15250 17710
18920 23875 16430 ...
 - attr(*, "na.action")= 'omit' Named int  10 45 46 130
  ..- attr(*, "names")= chr  "10" "45" "46" "130"
```

In [43]:

```
#Correlation matrix of numeric variables
cor(df[ , c(1, 2,10,11, 12, 13, 14, 17, 19, 20, 21, 22, 23, 24, 25, 26)])
```

	symboling	normalized_losses	wheel_base	length	width	
symboling	1.000000000	0.46626376	-0.53598680	-0.36540436	-0.24242260	-0.5
normalized_losses	0.466263758	1.00000000	-0.05666124	0.01942356	0.08680206	-0.3
wheel_base	-0.535986803	-0.05666124	1.00000000	0.87602389	0.81450665	0.5
length	-0.365404363	0.01942356	0.87602389	1.00000000	0.85717032	0.4
width	-0.242422604	0.08680206	0.81450665	0.85717032	1.00000000	0.3
height	-0.550159864	-0.37373695	0.59074167	0.49206255	0.30600216	1.0
curb_weight	-0.233118485	0.09940425	0.78209724	0.88066479	0.86620110	0.3
engine_size	-0.110580556	0.11236002	0.57202669	0.68502476	0.72943564	0.0
bore	-0.140019370	-0.02986248	0.49324419	0.60897097	0.54488546	0.1
stroke	-0.008152842	0.05504531	0.15801767	0.12395231	0.18882198	-0.0
compression_ratio	-0.182196158	-0.11471325	0.25031309	0.15973311	0.18986712	0.2
horsepower	0.075818883	0.21729943	0.37114668	0.57982145	0.61507674	-0.0
peak_rpm	0.279739659	0.23954349	-0.36030453	-0.28596960	-0.24580014	-0.3
city_mpg	-0.035527043	-0.22501573	-0.47060641	-0.66519239	-0.63353064	-0.0
highway_mpg	0.036232811	-0.18187718	-0.54330447	-0.69814185	-0.68063521	-0.1
price	-0.082391187	0.13399873	0.58464182	0.69062838	0.75126534	0.1

In [44]:

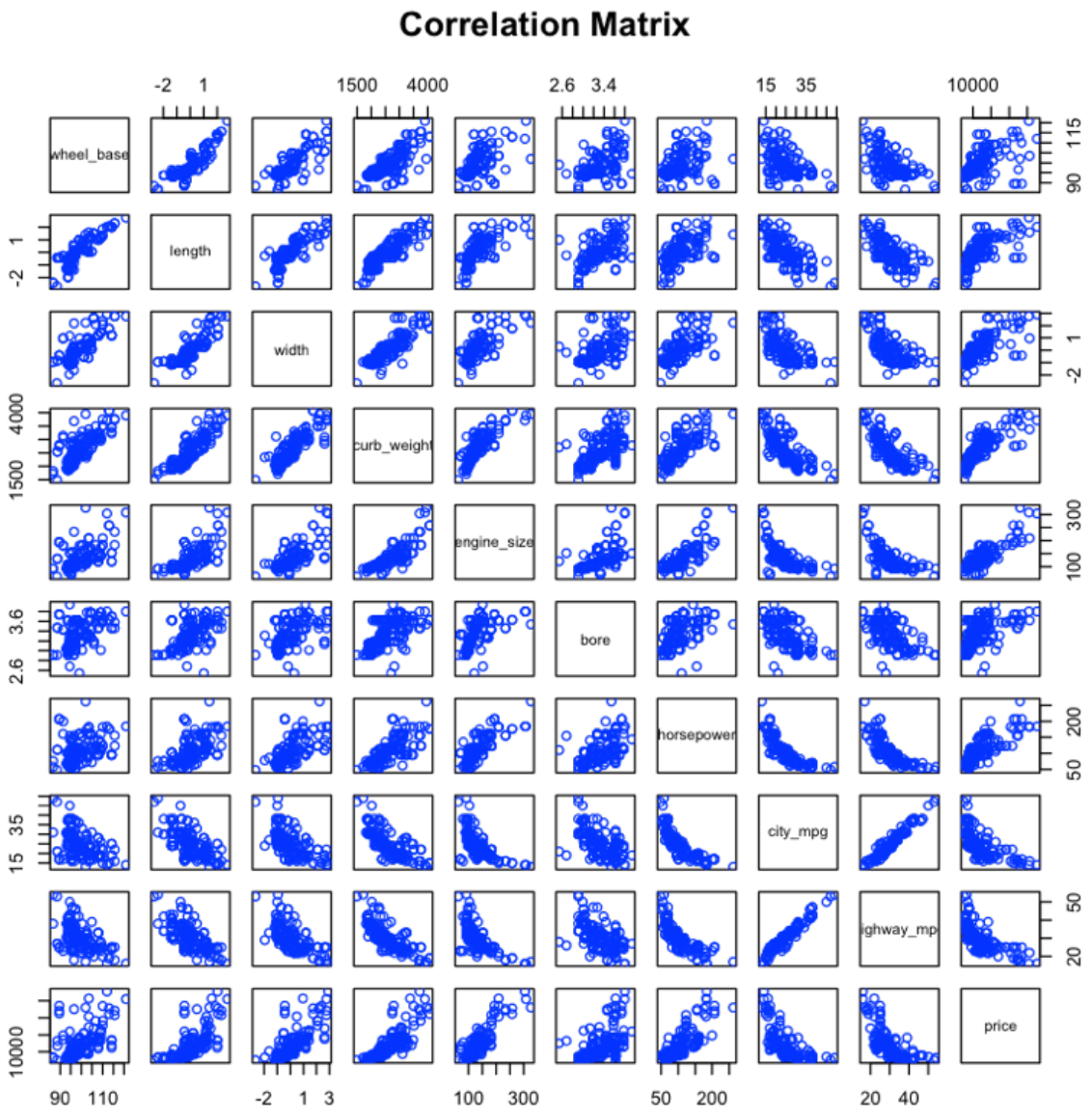
```
#Highly Correlated variables with price
cor(df[ , c(10,11, 12, 14, 17, 19, 22, 24, 25, 26)])
```

	wheel_base	length	width	curb_weight	engine_size	bore	hors
wheel_base	1.0000000	0.8760239	0.8145067	0.7820972	0.5720267	0.4932442	0.3
length	0.8760239	1.0000000	0.8571703	0.8806648	0.6850248	0.6089710	0.5
width	0.8145067	0.8571703	1.0000000	0.8662011	0.7294356	0.5448855	0.6
curb_weight	0.7820972	0.8806648	0.8662011	1.0000000	0.8490717	0.6440605	0.7
engine_size	0.5720267	0.6850248	0.7294356	0.8490717	1.0000000	0.5726093	0.8
bore	0.4932442	0.6089710	0.5448855	0.6440605	0.5726093	1.0000000	0.5
horsepower	0.3711467	0.5798215	0.6150767	0.7579756	0.8226756	0.5669355	1.0
city_mpg	-0.4706064	-0.6651924	-0.6335306	-0.7495431	-0.6505460	-0.5820270	-0.8
highway_mpg	-0.5433045	-0.6981418	-0.6806352	-0.7948889	-0.6795713	-0.5913092	-0.8
price	0.5846418	0.6906284	0.7512653	0.8344145	0.8723352	0.5431554	0.8

Correlation matrix of scatter Plot

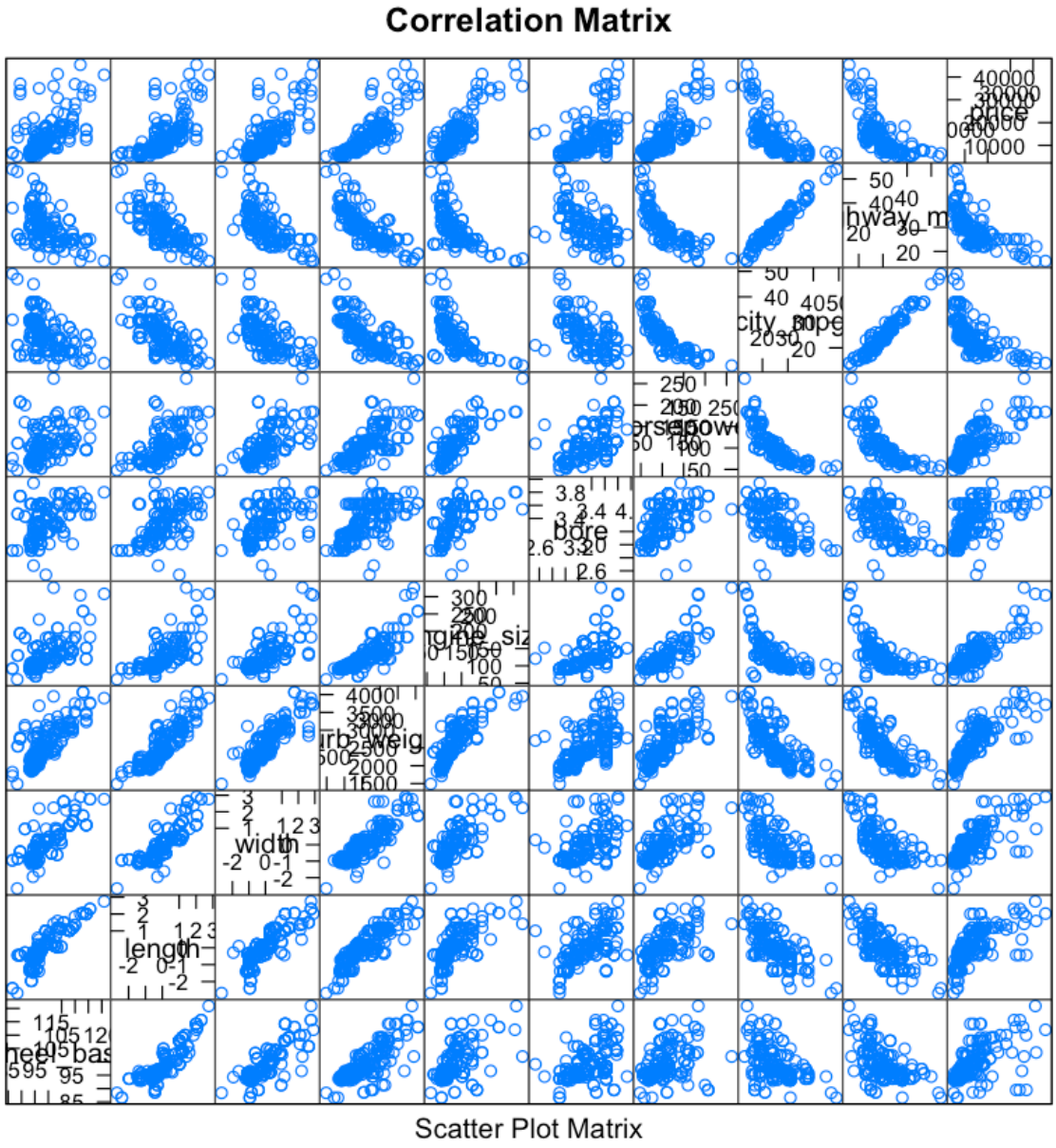
In [45]:

```
# Correlation matrix of scatter Plot of numeric variables
plot(df[ , c(10,11, 12, 14, 17, 19, 22, 24, 25, 26)], col = 'blue', main = 'Corr
elation Matrix')
```


Correlation Matrix
wheel_base
length
width
curb_weight
engine_size
bore
horsepower
city_mpg
highway_mpg
price

In [46]:

```
# scatterplot matrix
splom(df[c(10,11, 12, 14, 17, 19, 22, 24, 25, 26)], main="Correlation Matrix")
```

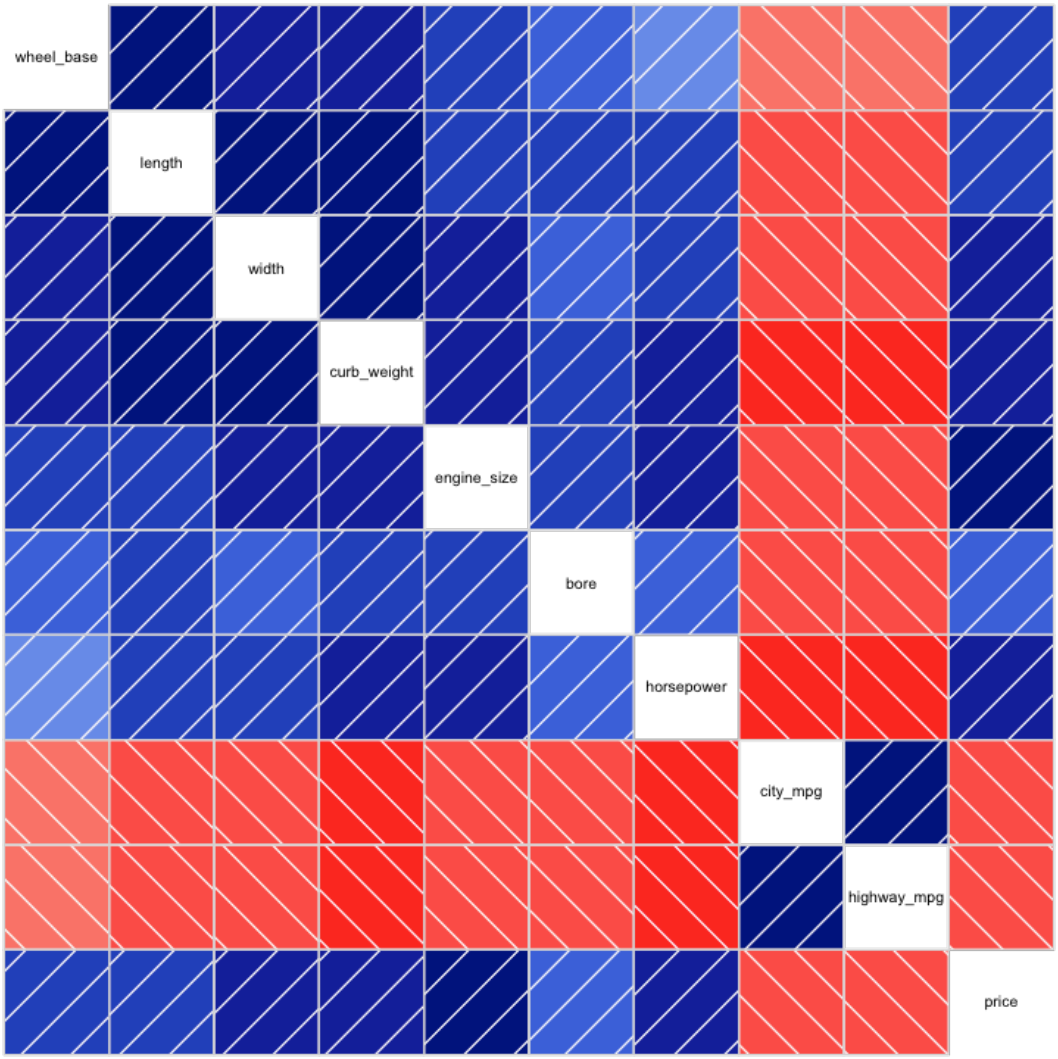


Correlogram

The corrgram function produces a graphical display of a correlation matrix, called a correlogram. The cells of the matrix can be shaded or colored to show the correlation value.

In [47]:

```
corrgram(df[ , c(10,11, 12, 14, 17, 19, 22, 24, 25, 26)])
```



Correlation plots

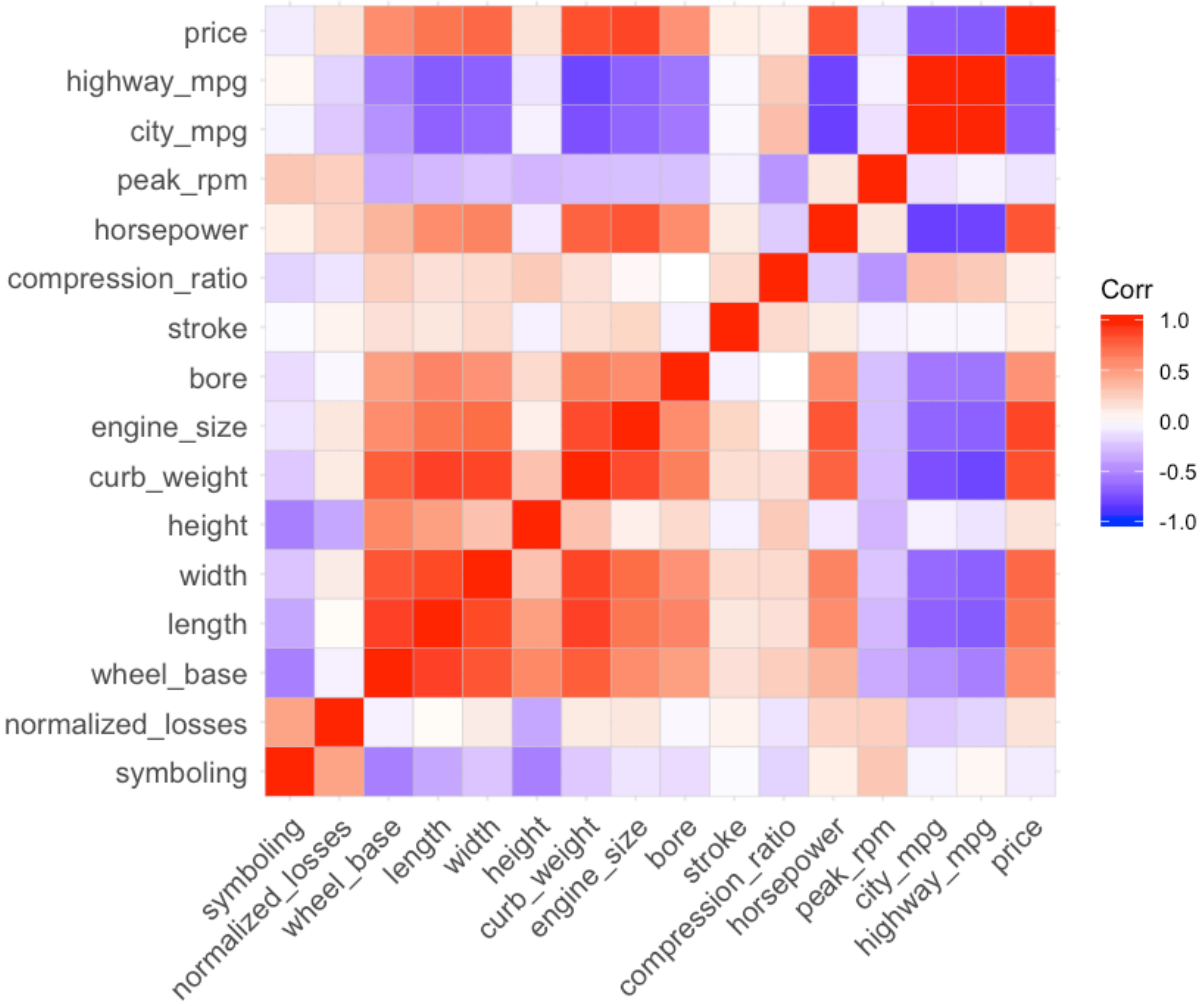
In [48]:

```
df2 <- dplyr::select_if(df, is.numeric)
# calulate the correlations
corln <- cor(df2, use="complete.obs")
round(corln, 2)
```

	symboling	normalized_losses	wheel_base	length	width	height	curb_wei
symboling	1.00	0.47	-0.54	-0.37	-0.24	-0.55	-0
normalized_losses	0.47	1.00	-0.06	0.02	0.09	-0.37	0
wheel_base	-0.54	-0.06	1.00	0.88	0.81	0.59	0
length	-0.37	0.02	0.88	1.00	0.86	0.49	0
width	-0.24	0.09	0.81	0.86	1.00	0.31	0
height	-0.55	-0.37	0.59	0.49	0.31	1.00	0
curb_weight	-0.23	0.10	0.78	0.88	0.87	0.31	1
engine_size	-0.11	0.11	0.57	0.69	0.73	0.07	0
bore	-0.14	-0.03	0.49	0.61	0.54	0.18	0
stroke	-0.01	0.06	0.16	0.12	0.19	-0.06	0
compression_ratio	-0.18	-0.11	0.25	0.16	0.19	0.26	0
horsepower	0.08	0.22	0.37	0.58	0.62	-0.09	0
peak_rpm	0.28	0.24	-0.36	-0.29	-0.25	-0.31	-0
city_mpg	-0.04	-0.23	-0.47	-0.67	-0.63	-0.05	-0
highway_mpg	0.04	-0.18	-0.54	-0.70	-0.68	-0.10	-0
price	-0.08	0.13	0.58	0.69	0.75	0.14	0

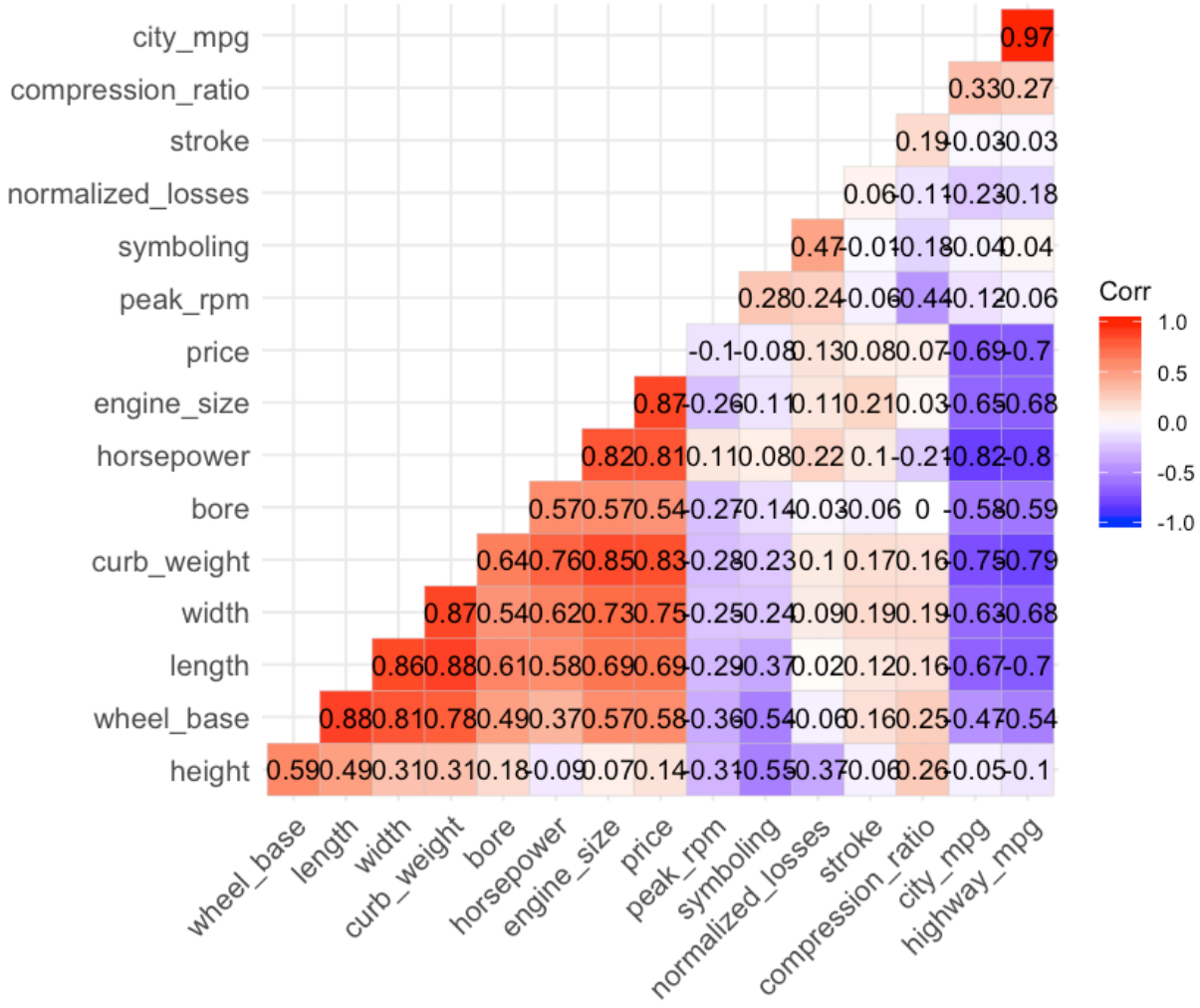
In [49]:

```
# Correlation Plot
ggcorrplot(corln)
```


price
highway_mpg
city_mpg
peak_rpm
horsepower
compression_ratio
stroke
bore
engine_size
curb_weight
height
width
length
wheel_base
normalized_losses
symboling
symboling
normalized_losses
wheel_base
length
width
height
curb_weight
engine_size
bore
stroke
compression_ratio
horsepower
peak_rpm
city_mpg
highway_mpg
price
Corr
1.0
0.5
0.0
-0.5
-1.0

In [50]:

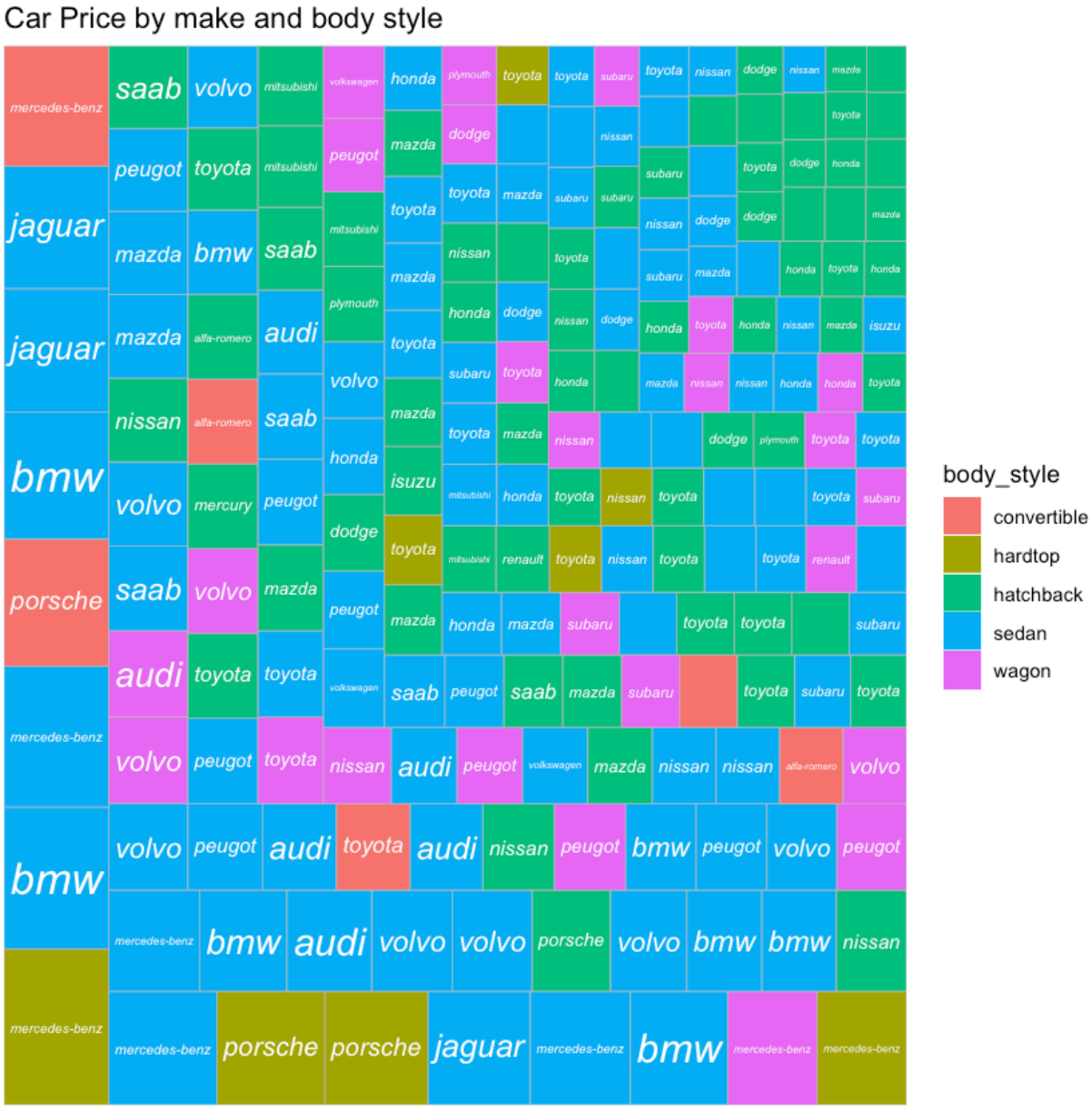
```
ggcorrplot(corln, hc.order = TRUE, type = "lower", lab = TRUE)
```



Tree Map

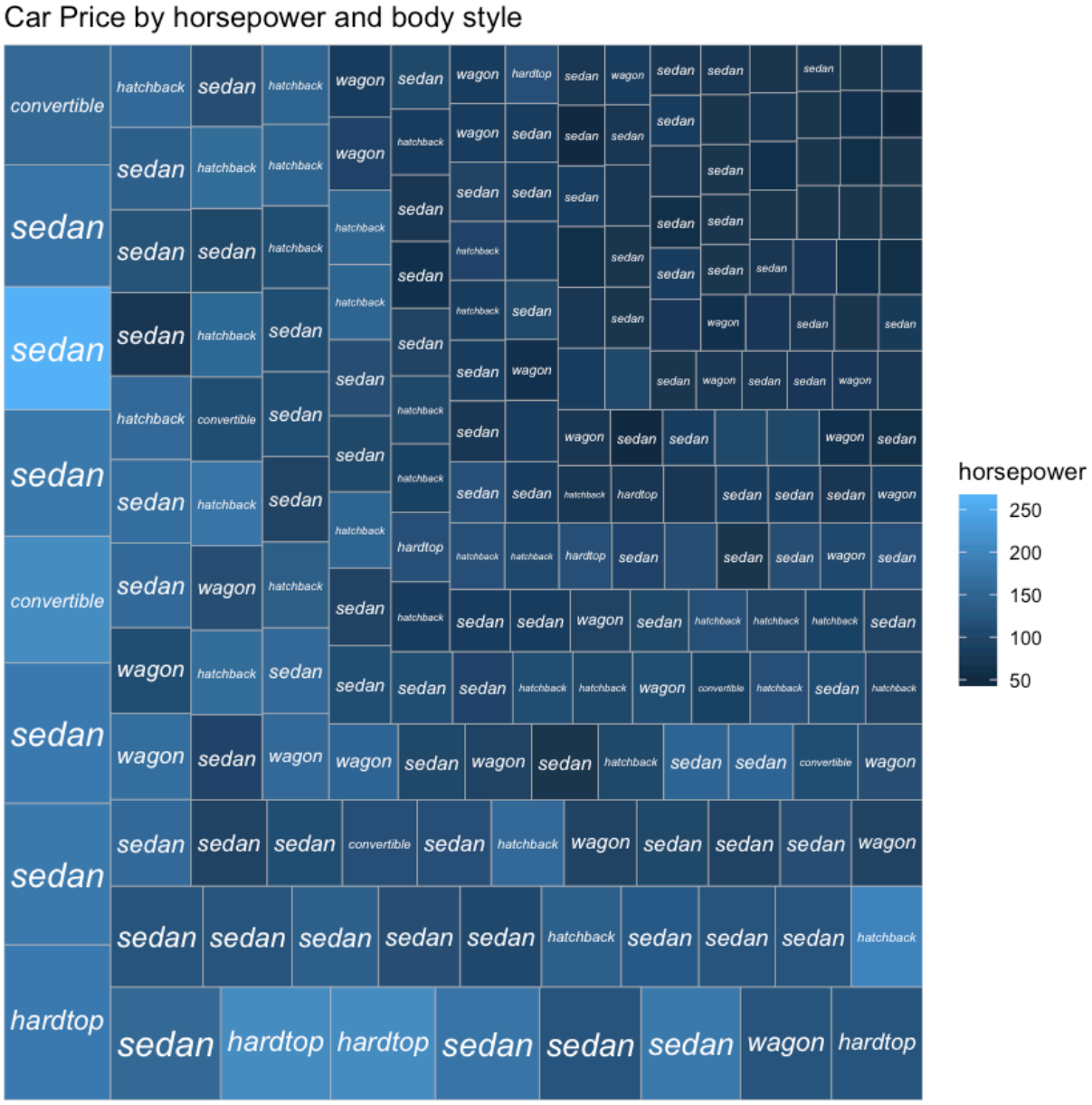
In [51]:

```
# create a treemap with tile labels
# area = numeric , fill = numeric or categorical, label = categorical variable
ggplot(df, aes(area = price, fill = body_style, label = make)) +
       geom_treemap() +
       geom_treemap_text(fontface = "italic", colour = "white", place = "centre"
, grow = TRUE)+
       labs(title = "Car Price by make and body style")
```



In [52]:

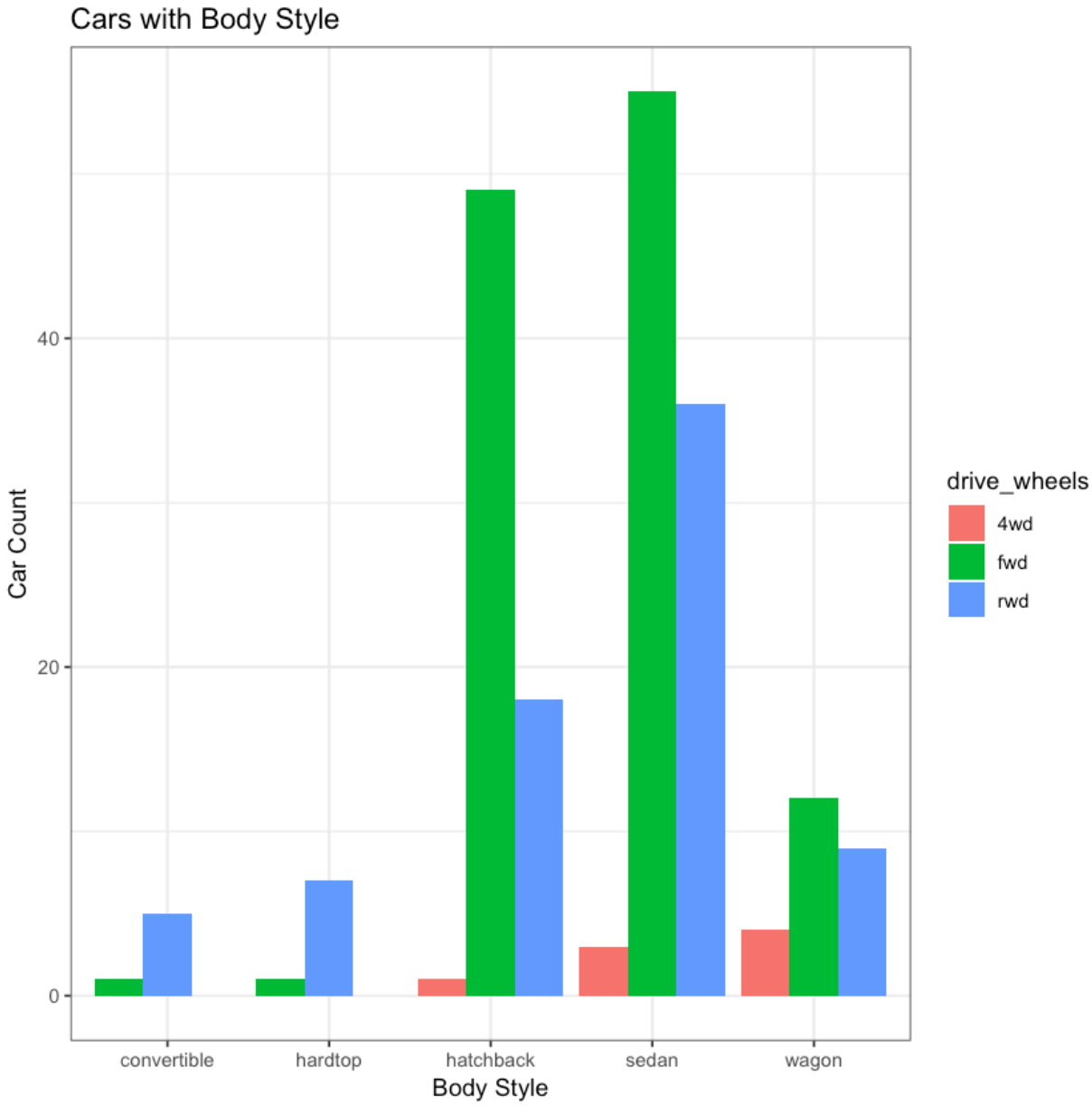
```
# create a treemap with tile labels
ggplot(df, aes(area = price , fill = horsepower, label = body_style)) +
       geom_treemap() +
       geom_treemap_text(fontface = "italic", colour = "white", place = "centre"
, grow = TRUE) +
       labs(title = "Car Price by horsepower and body style")
```



Multiple Bar Chart

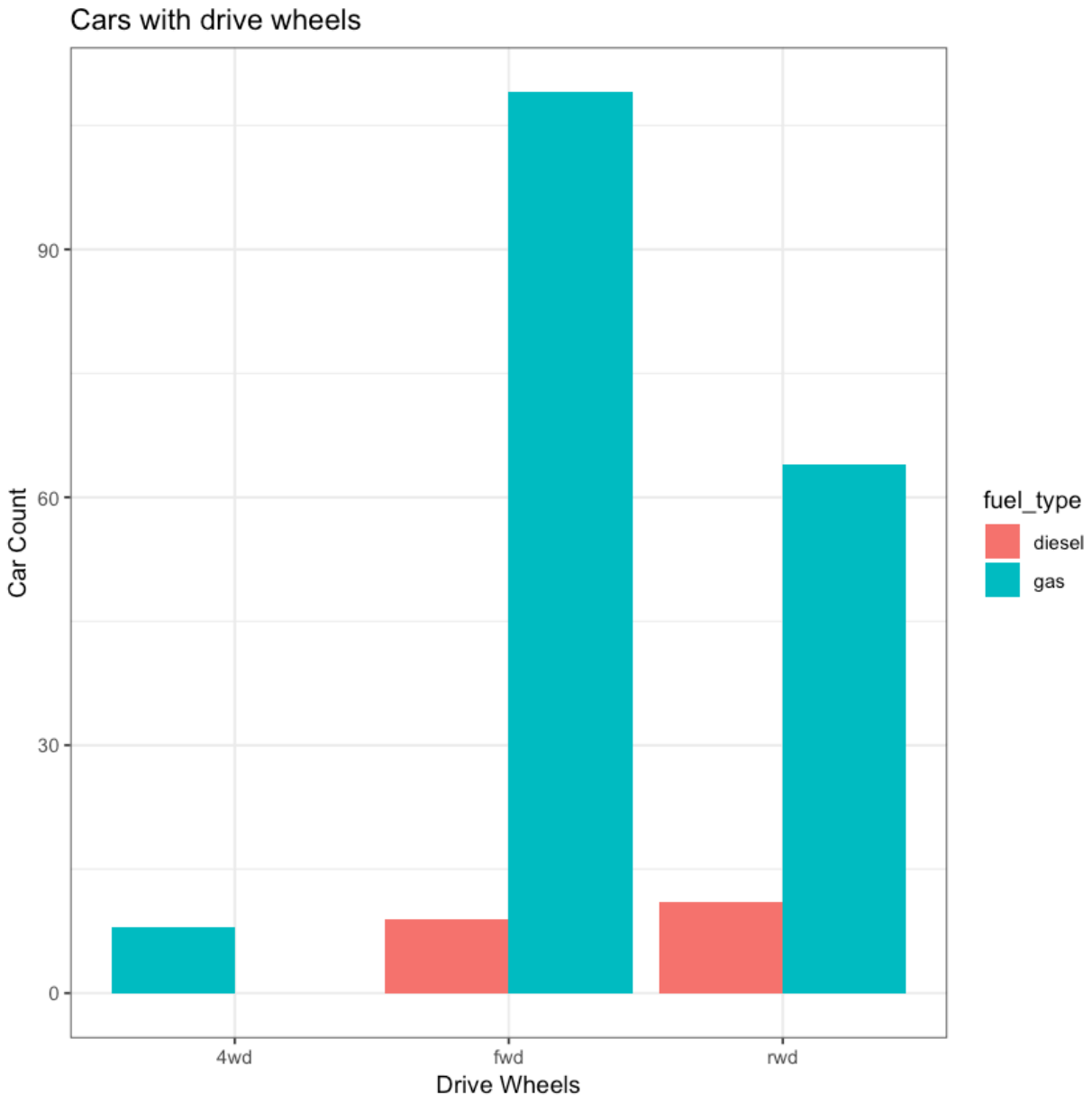
In [53]:

```
# grouped bar plot preserving zero count bars (Multiple bar chart)
ggplot(df, aes(x = body_style, fill = drive_wheels)) +
       geom_bar(position = position_dodge(preserve = "single"))+
       theme_bw()+
       labs(x = 'Body Style', y = 'Car Count', title = 'Cars with Body Style' )
```



In [54]:

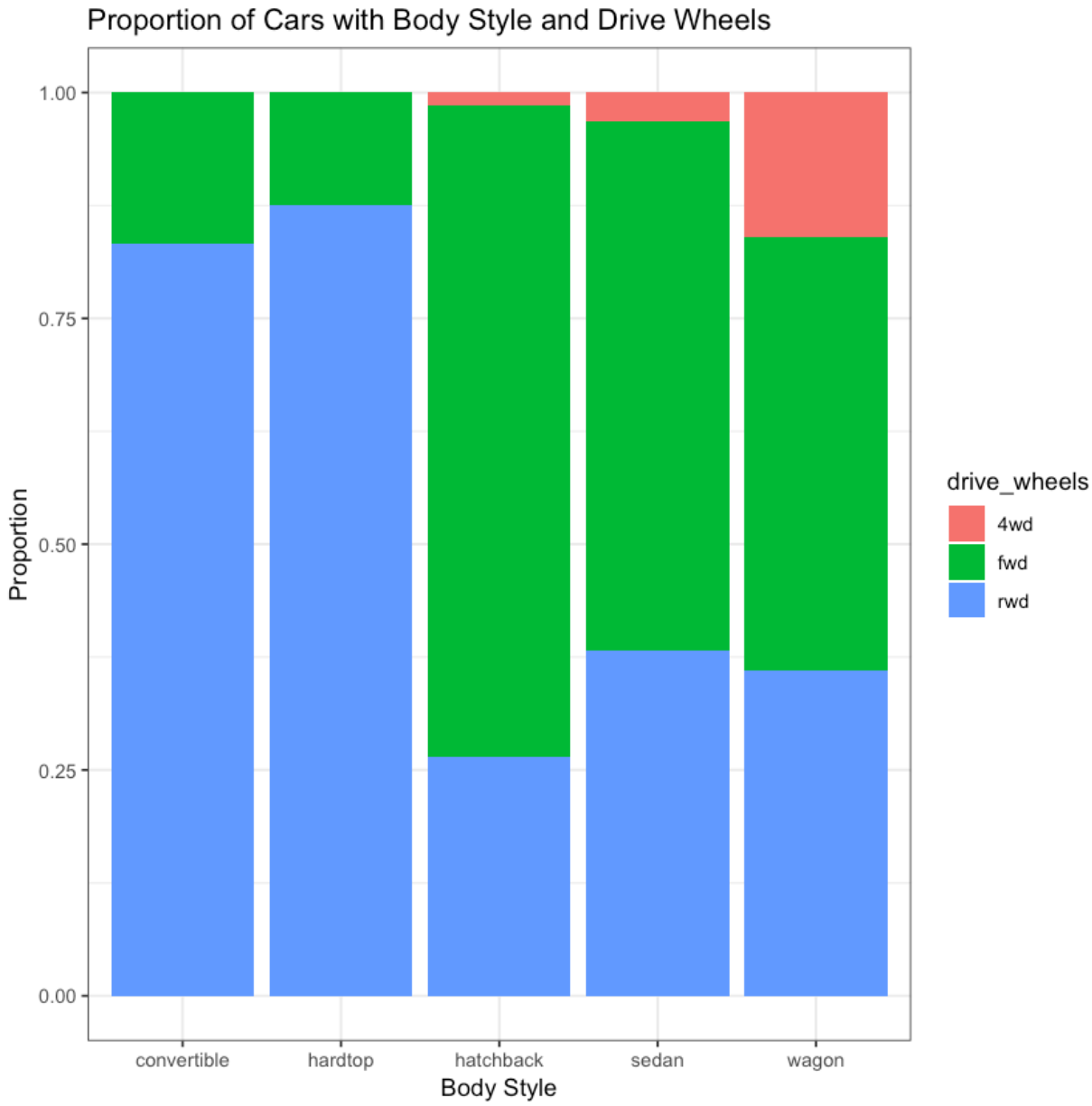
```
# grouped bar plot preserving zero count bars (Multiple bar chart)
ggplot(df, aes(x = drive_wheels, fill = fuel_type)) +
       geom_bar(position = position_dodge(preserve = "single"))+
       theme_bw()+
       labs(x = 'Drive Wheels', y = 'Car Count', title = 'Cars with drive wheels
' )
```



Multiple Stacked Bar Chart

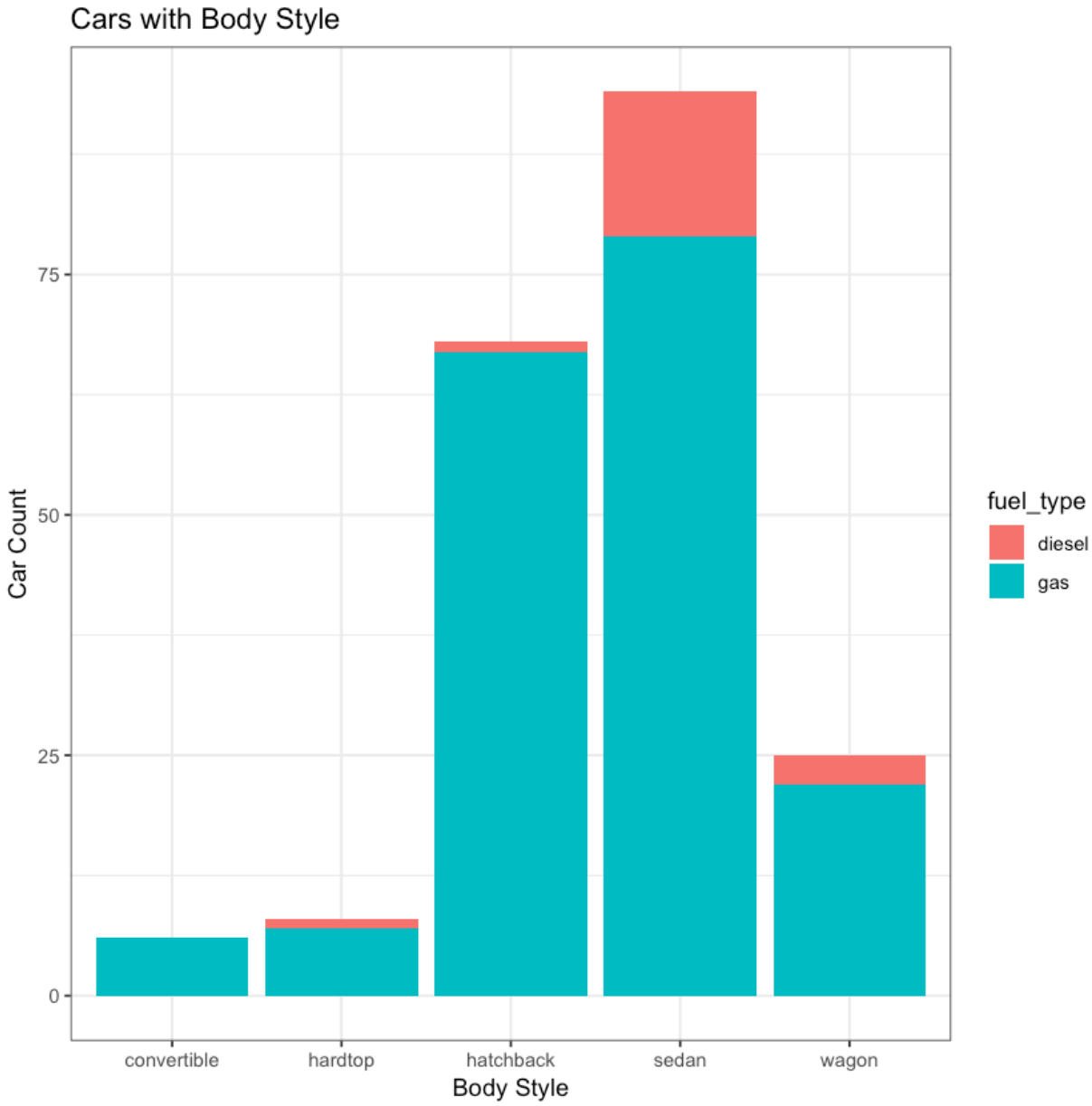
In [55]:

```
# bar plot, with each bar representing 100% (Stacked or subdivided bar chart)
ggplot(df, aes(x = body_style, fill = drive_wheels)) +
      geom_bar(position = "fill") +
      theme_bw()+
      labs(x = 'Body Style', y = 'Proportion',
           title = 'Proportion of Cars with Body Style and Drive Wheels' )
```



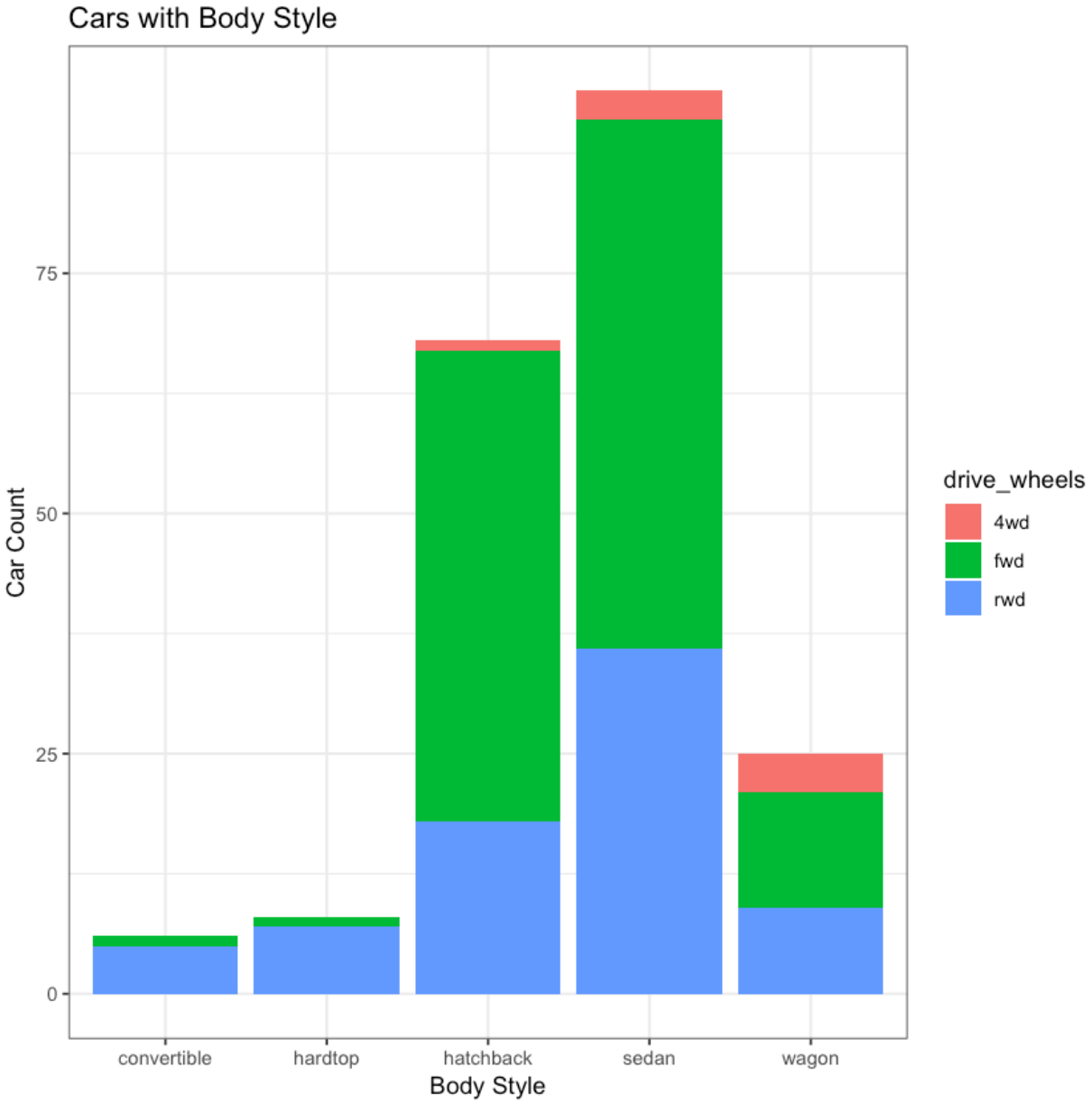
In [56]:

```
# Bar plot fill by another categorical variable
ggplot(df, aes(x = body_style, fill = fuel_type  )) +
       geom_bar() +
       theme_bw()+
       labs(x = 'Body Style', y = 'Car Count', title = 'Cars with Body Style' )
```



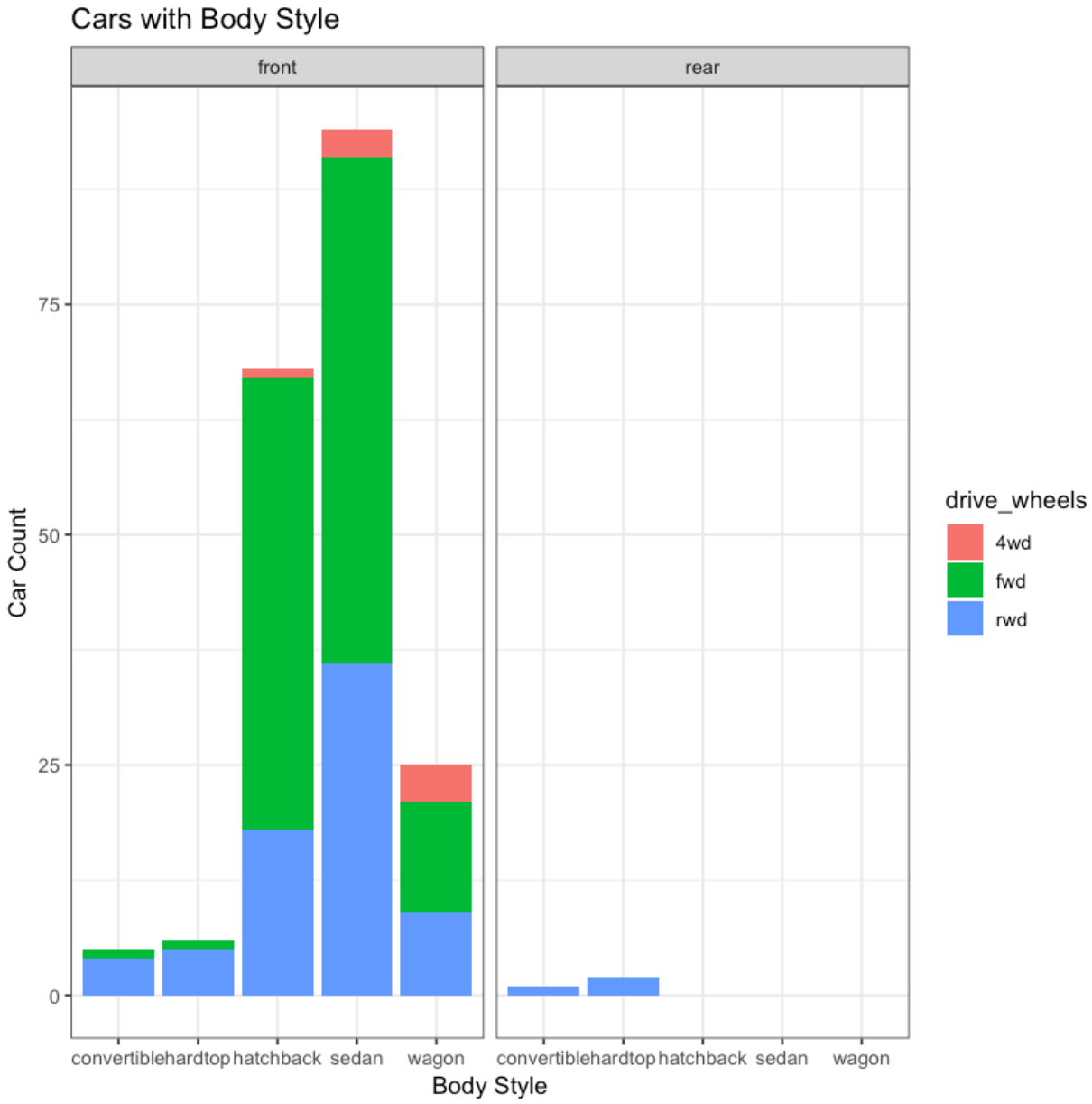
In [57]:

```
# Bar plot fill by another categorical variable
ggplot(df, aes(x = body_style, fill = drive_wheels)) +  geom_bar() +
       theme_bw()+
       labs(x = 'Body Style', y = 'Car Count', title = 'Cars with Body Style' )
```



In [58]:

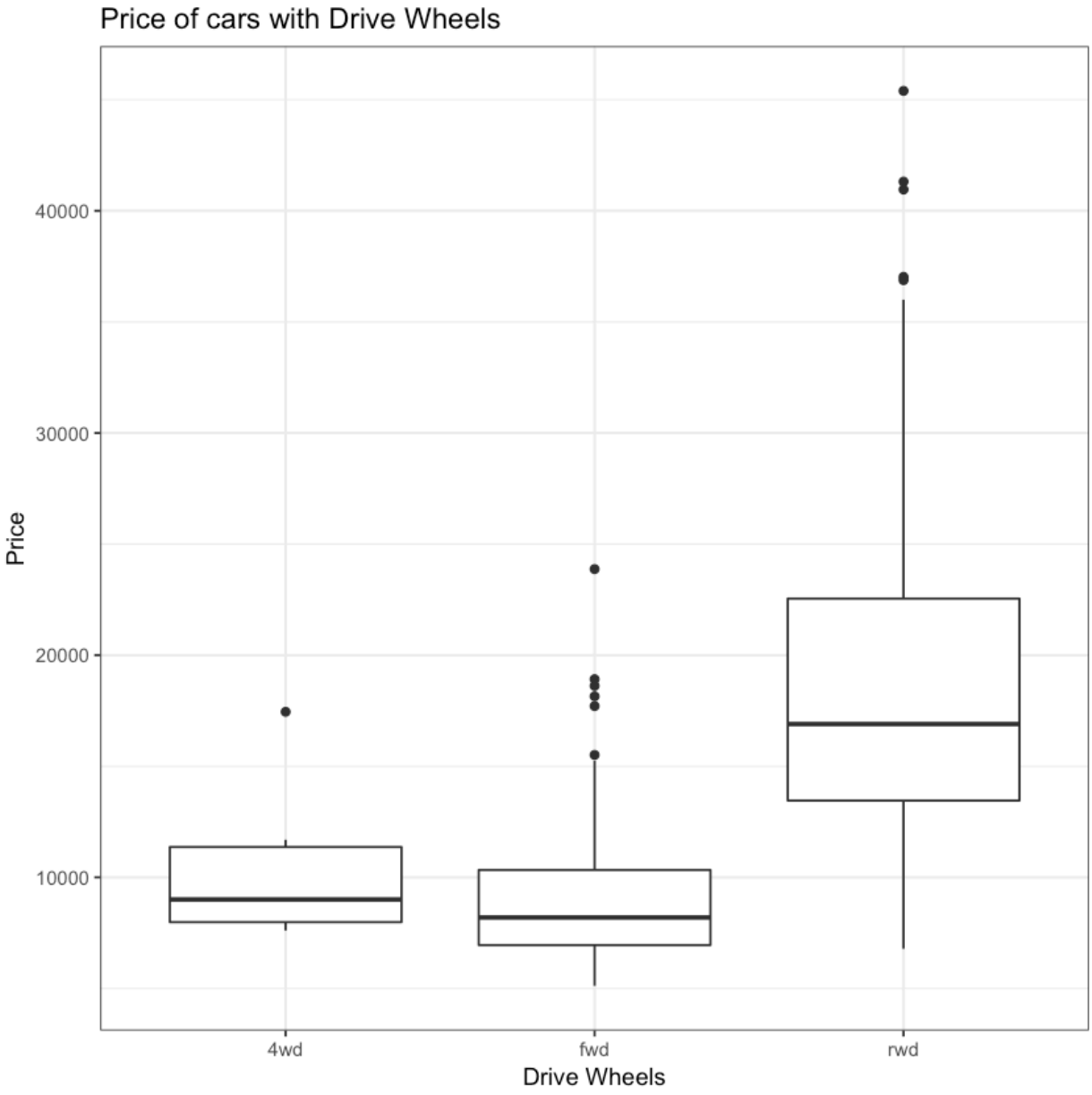
```
# Bar plot fill by another categorical variable
ggplot(df, aes(x = body_style, fill = drive_wheels)) +  geom_bar() +
       theme_bw()+
       facet_wrap(~engine_location)+
       labs(x = 'Body Style', y = 'Car Count', title = 'Cars with Body Style' )
```



Box Plot

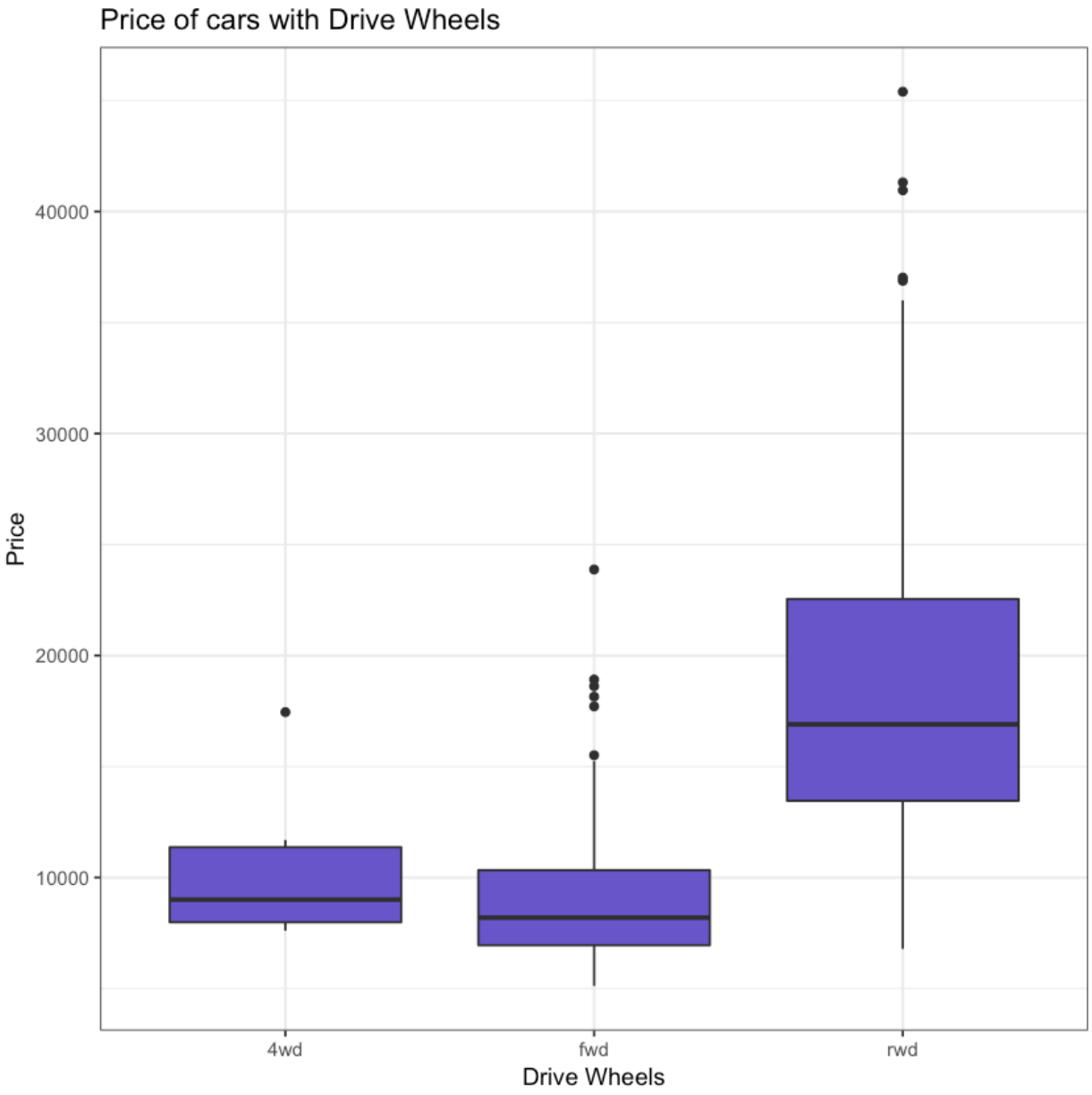
In [59]:

```
#Box Plot
ggplot(df, aes(x = drive_wheels, y = price)) +  geom_boxplot() +
       theme_bw()+
       labs(x = 'Drive Wheels', y = 'Price', title = 'Price of cars with Drive W
heels' )
```



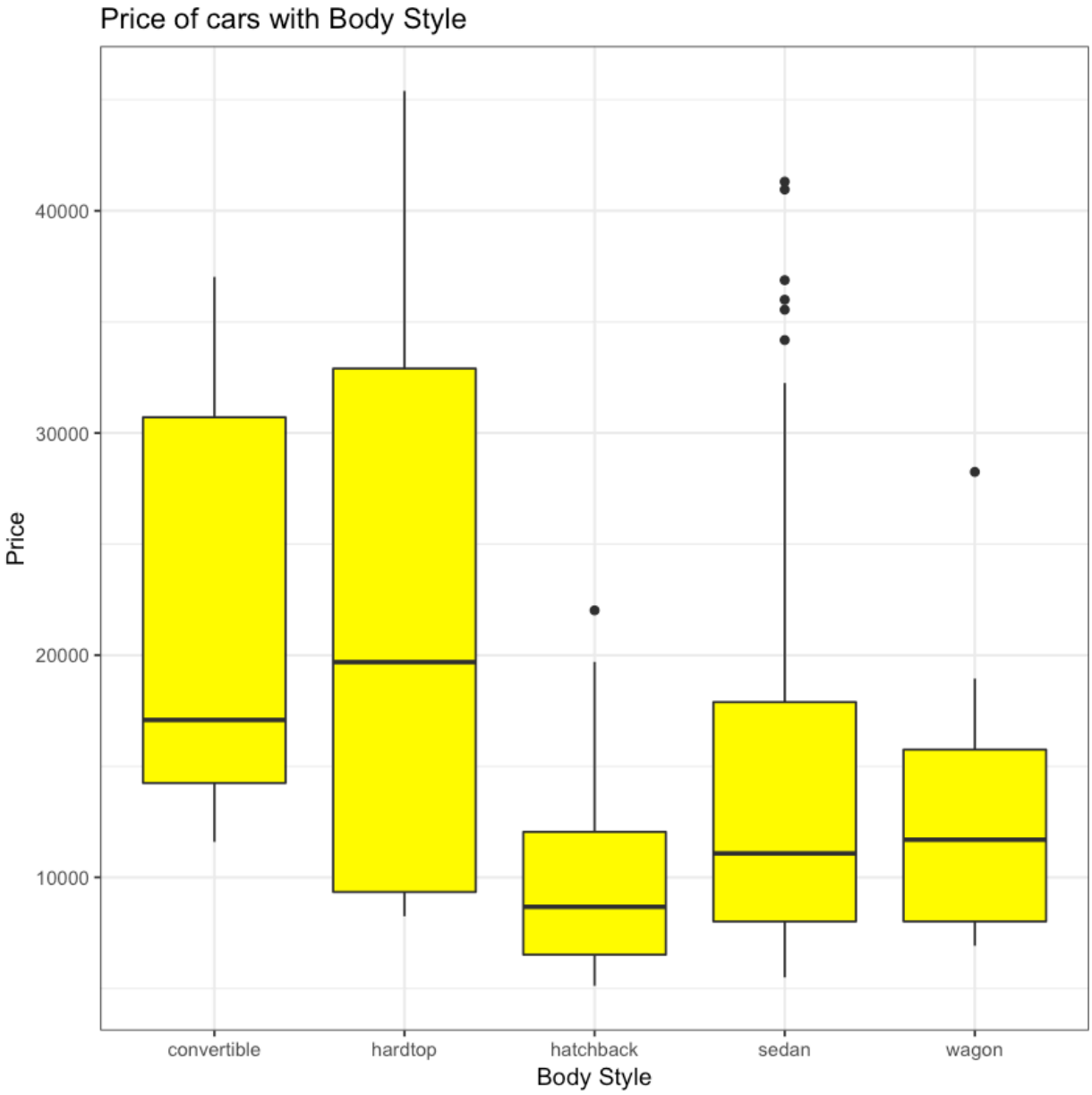
In [60]:

```
#Box Plot
ggplot(df, aes(x = drive_wheels, y = price)) +  geom_boxplot(fill = 'slateblue')
+
       theme_bw()+
       labs(x = 'Drive Wheels', y = 'Price', title = 'Price of cars with Drive W
heels' )
```



In [61]:

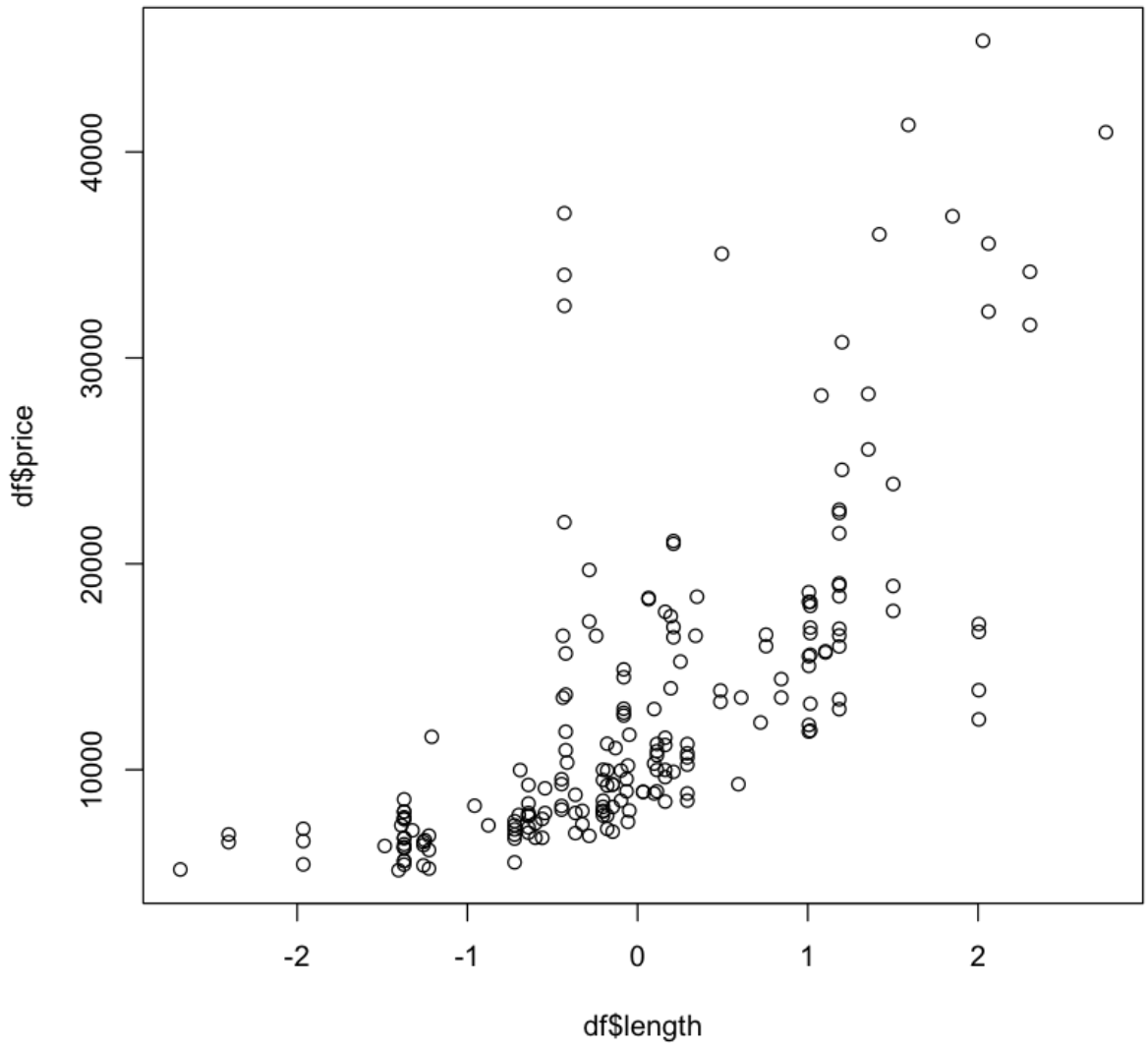
```
#Box Plot
ggplot(df, aes(x = body_style, y = price)) +  geom_boxplot(fill = 'yellow') +
       theme_bw()+
       labs(x = 'Body Style', y = 'Price', title = 'Price of cars with Body Styl
e' )
```



Scatter Plot

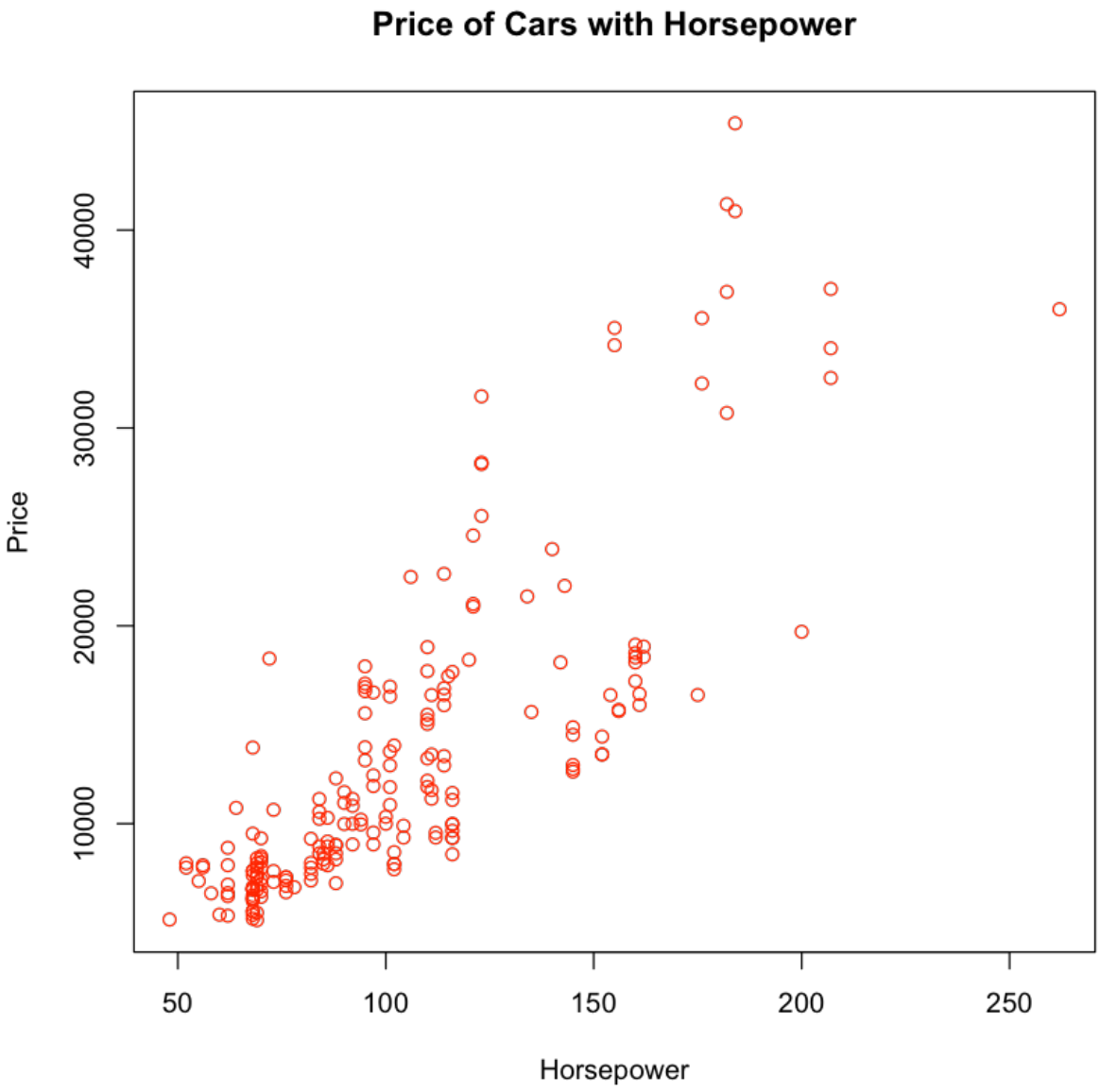
In [63]:

```
#Scatter Plot
plot(df$length, df$price, )
```



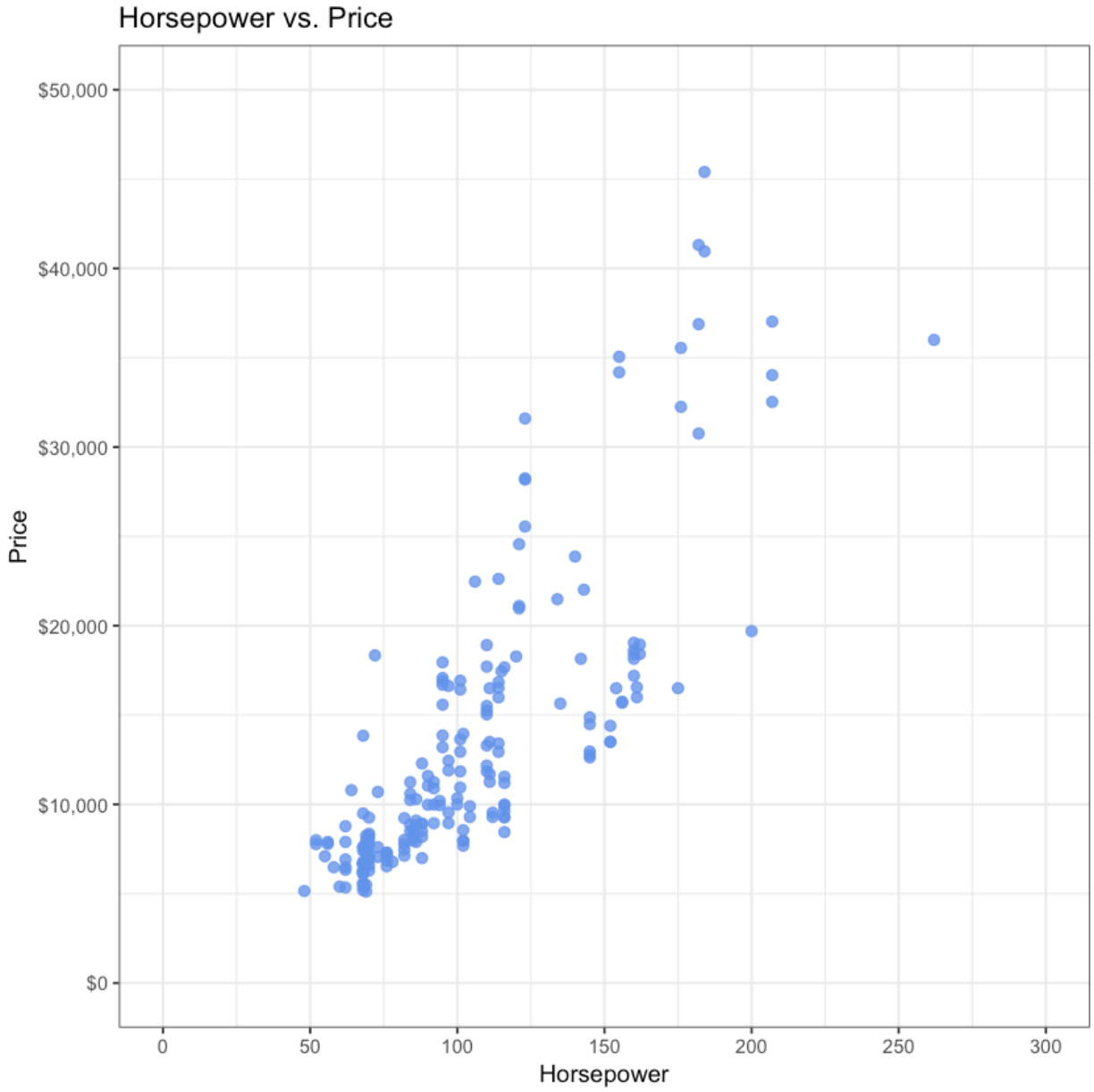
In [65]:

```
#Scatter Plot
plot(df$horsepower, df$price, xlab = 'Horsepower', ylab = 'Price',
    main = 'Price of Cars with Horsepower', col = 'red')
```



In [66]:

```
# enhanced scatter plot
ggplot(df, aes(x = horsepower, y = price)) +
     geom_point(color="cornflowerblue", size = 2, alpha=.8) +
     theme_bw()+
     scale_y_continuous(label = scales::dollar, limits = c(0, 50000)) +
     scale_x_continuous(breaks = seq(0, 300, 50), limits=c(0, 300)) +
     labs(x = "Horsepower", y = "Price", title = "Horsepower vs. Price")
```



In [67]:

```
# Enhanced Linear Regression
ggplot(df, aes(x = horsepower, y = price)) +
     geom_point(color="steelblue", size = 1) +
     geom_smooth(method = "lm", color = 'red')+
     scale_y_continuous(label = scales::dollar, limits = c(0, 50000)) +
     scale_x_continuous(breaks = seq(0, 300, 50), limits=c(0, 300)) +
     labs(x = "Horsepower", y = "Price", title = "Horsepower vs. Price")
```

In [68]:

```
# Enhanced Polynomial Regression
ggplot(df, aes(x = horsepower, y = price)) +
     geom_point(color="steelblue", size = 1) +
     geom_smooth(method = "lm",formula = y ~ poly(x, 3), color = 'red')+
     scale_y_continuous(label = scales::dollar, limits = c(0, 50000)) +
     scale_x_continuous(breaks = seq(0, 300, 50), limits=c(0, 300)) +
     labs(x = "Horsepower", y = "Price", title = "Horsepower vs. Price")
```

Mosaic plots

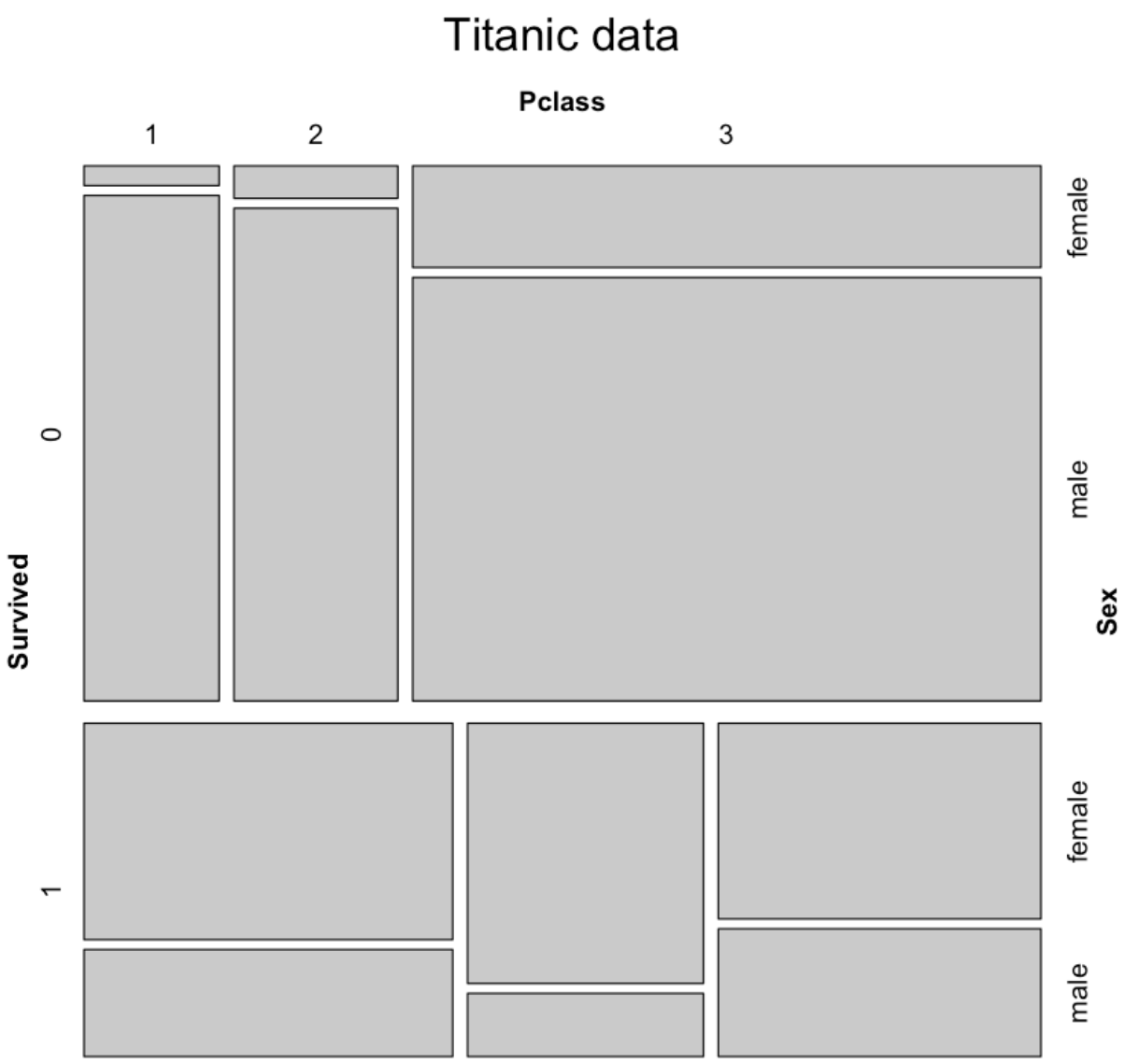
In [69]:

```
# input data
titanic <- read.csv("titanic.csv")

# create a table
tbl <- xtabs(~Survived + Pclass + Sex, titanic)
ftable(tbl)
```

```
                Sex female male
Survived Pclass
0        1                3   77
         2                6   91
         3               72  300
1        1               91   45
         2               70   17
         3               72   47
```

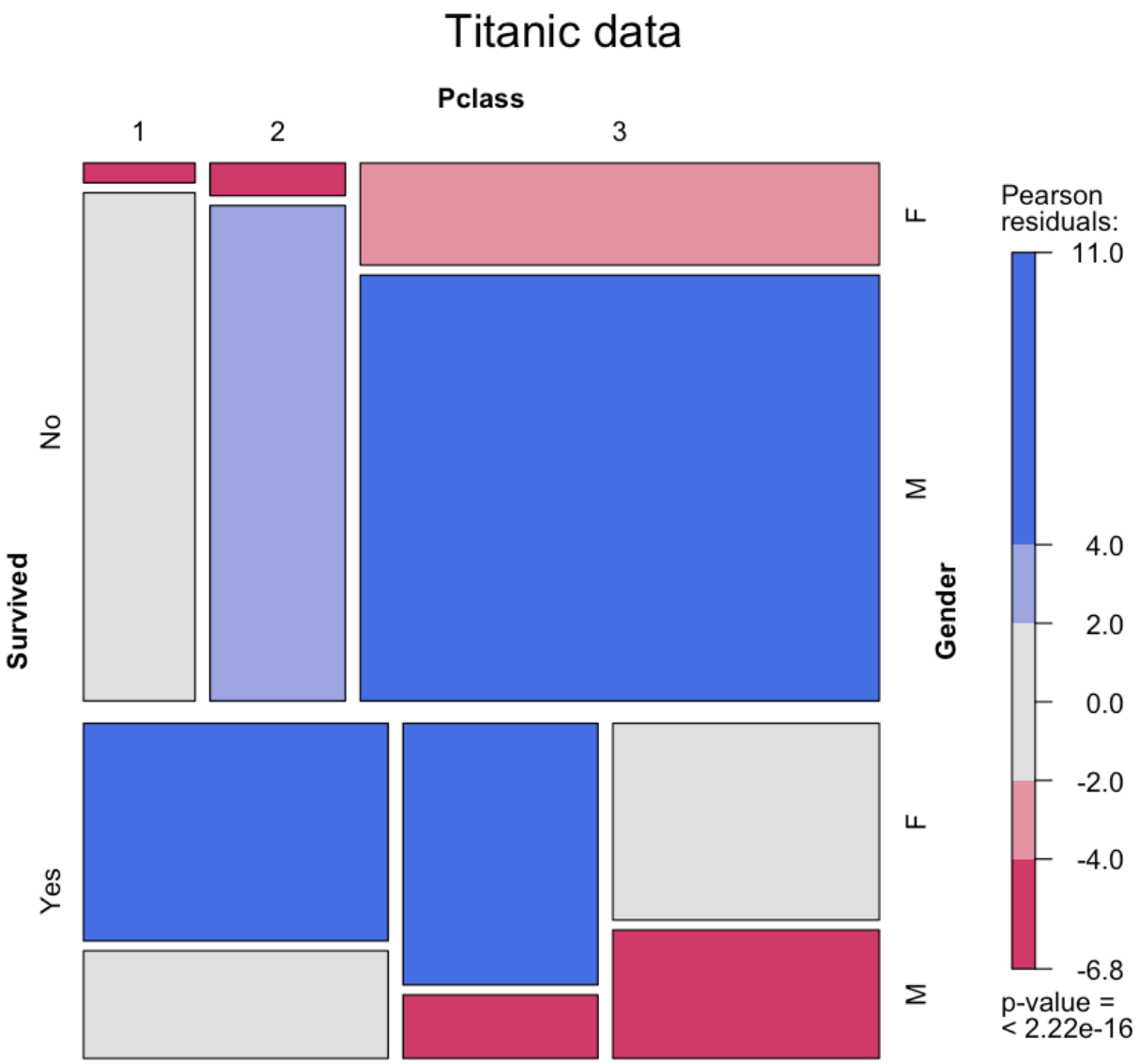
In [70]:

```
#Mosaic plots
mosaic(tbl, main = "Titanic data")
```


Titanic data
Pclass
1
2
3
Survived
0
1
Sex
female
male
female
male

In [71]:

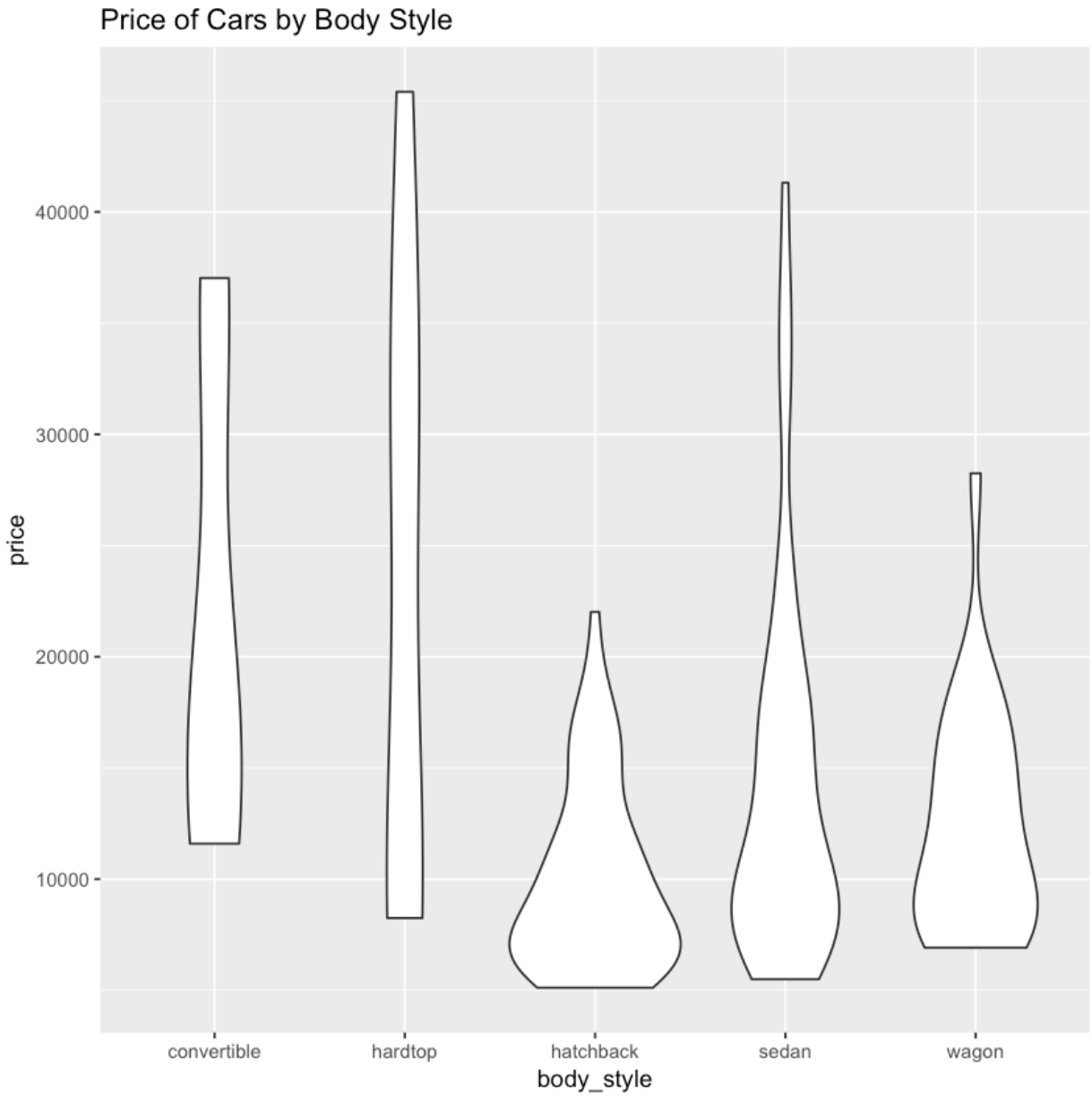
```
#Mosaic plots
mosaic(tbl, shade = TRUE, legend = TRUE,
       labeling_args = list(set_varnames = c(Sex = "Gender",
                                             Survived = "Survived",
                                             Class = "Passenger Class")),
       set_labels = list(Survived = c("No", "Yes"),
                         Class = c("1st", "2nd", "3rd", "Crew"),
                         Sex = c("F", "M")),
       main = "Titanic data")
```



Vilon Plot

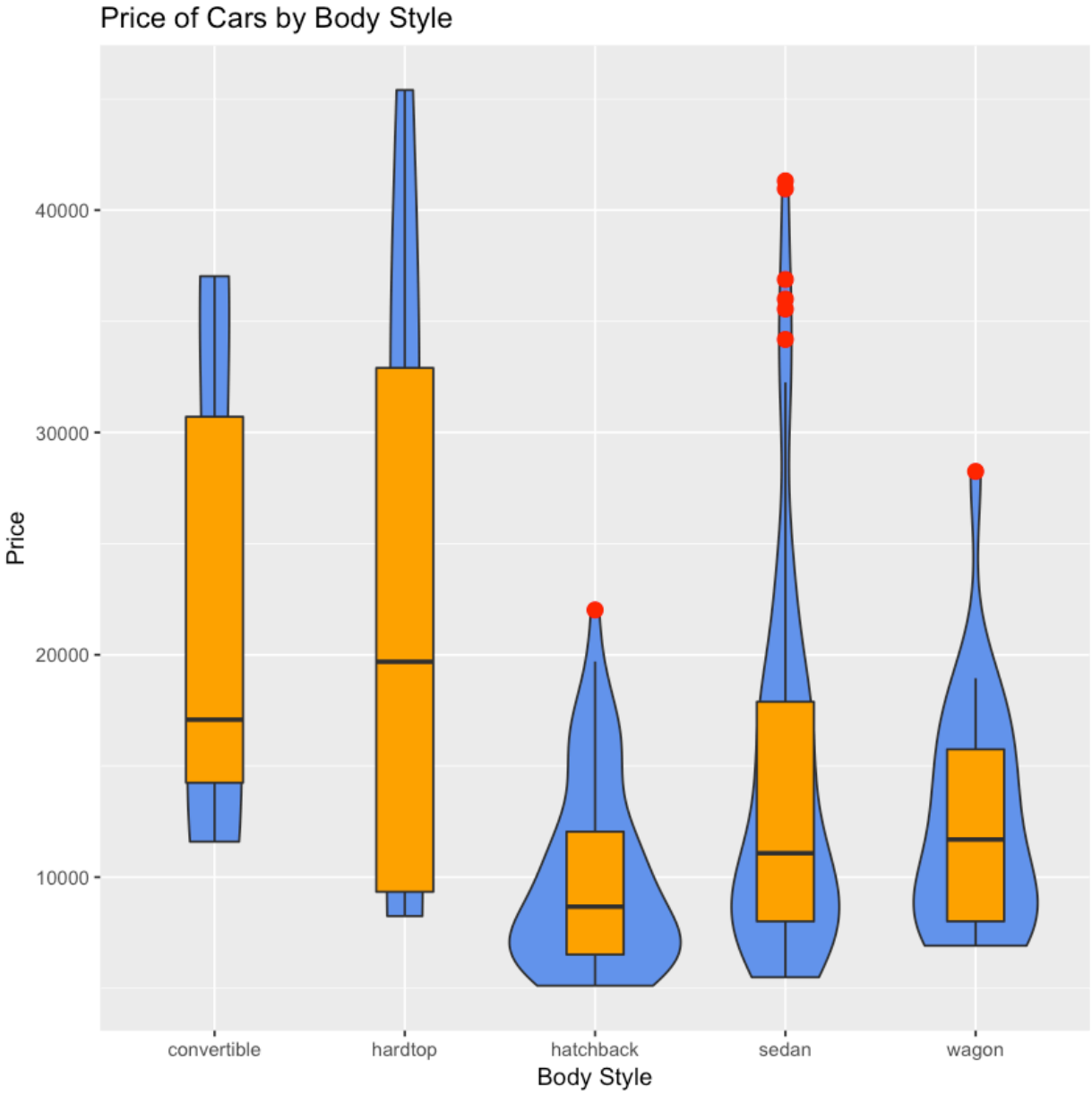
In [72]:

```
# Vilon Plot
ggplot(df, aes(x = body_style, y = price)) +
       geom_violin() +
       labs(title = "Price of Cars by Body Style")
```



In [73]:

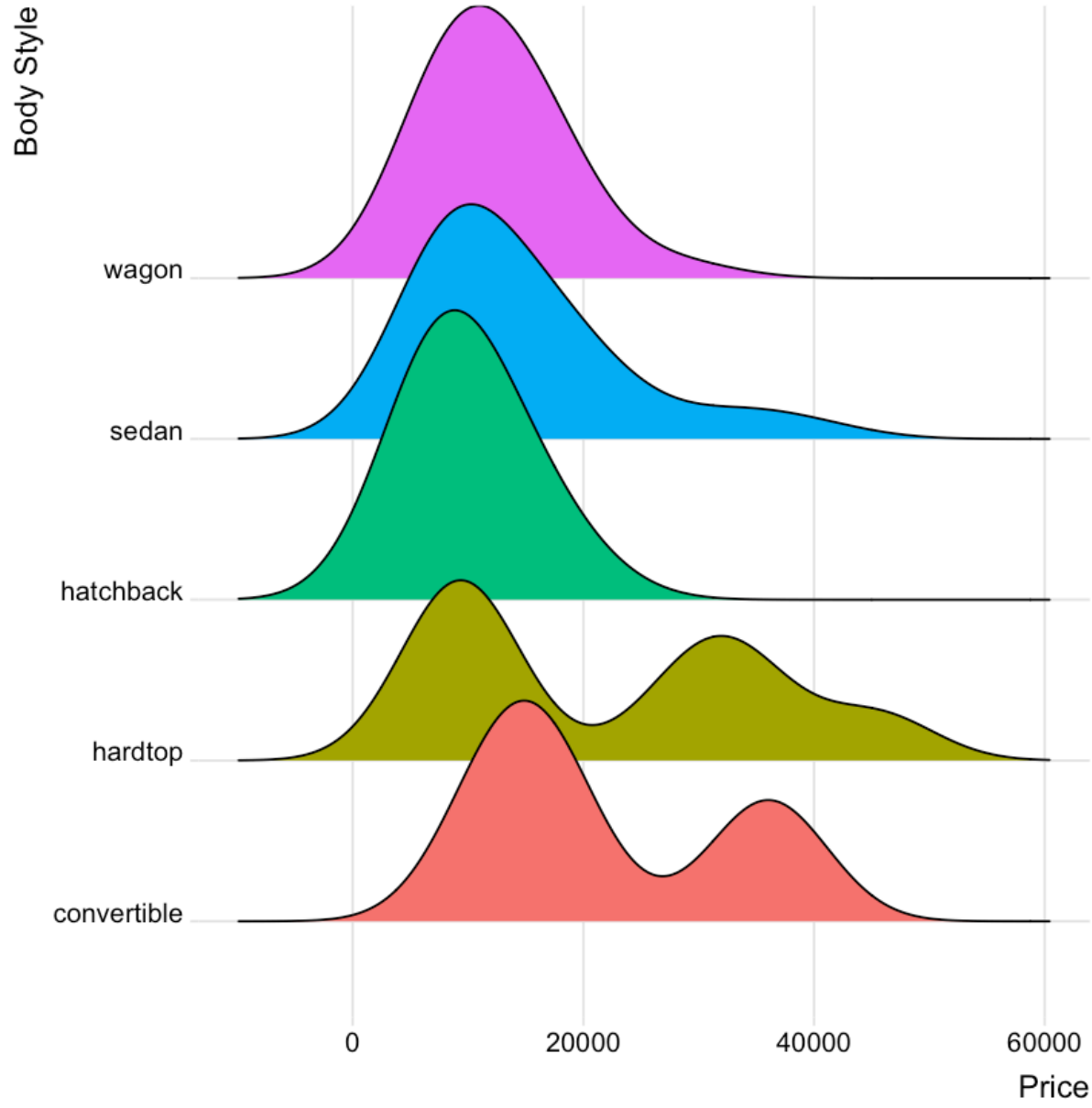
```
# Vilon Plot
ggplot(df, aes(x = body_style, y = price)) +
       geom_violin(fill = "cornflowerblue") +
       geom_boxplot(width = .3, fill = "orange", outlier.color = "red", outlier.
size = 3) +
       labs(x = "Body Style", y = "Price", title = "Price of Cars by Body Style"
)
```



Ridgeline plot or Joyplot)

In [74]:

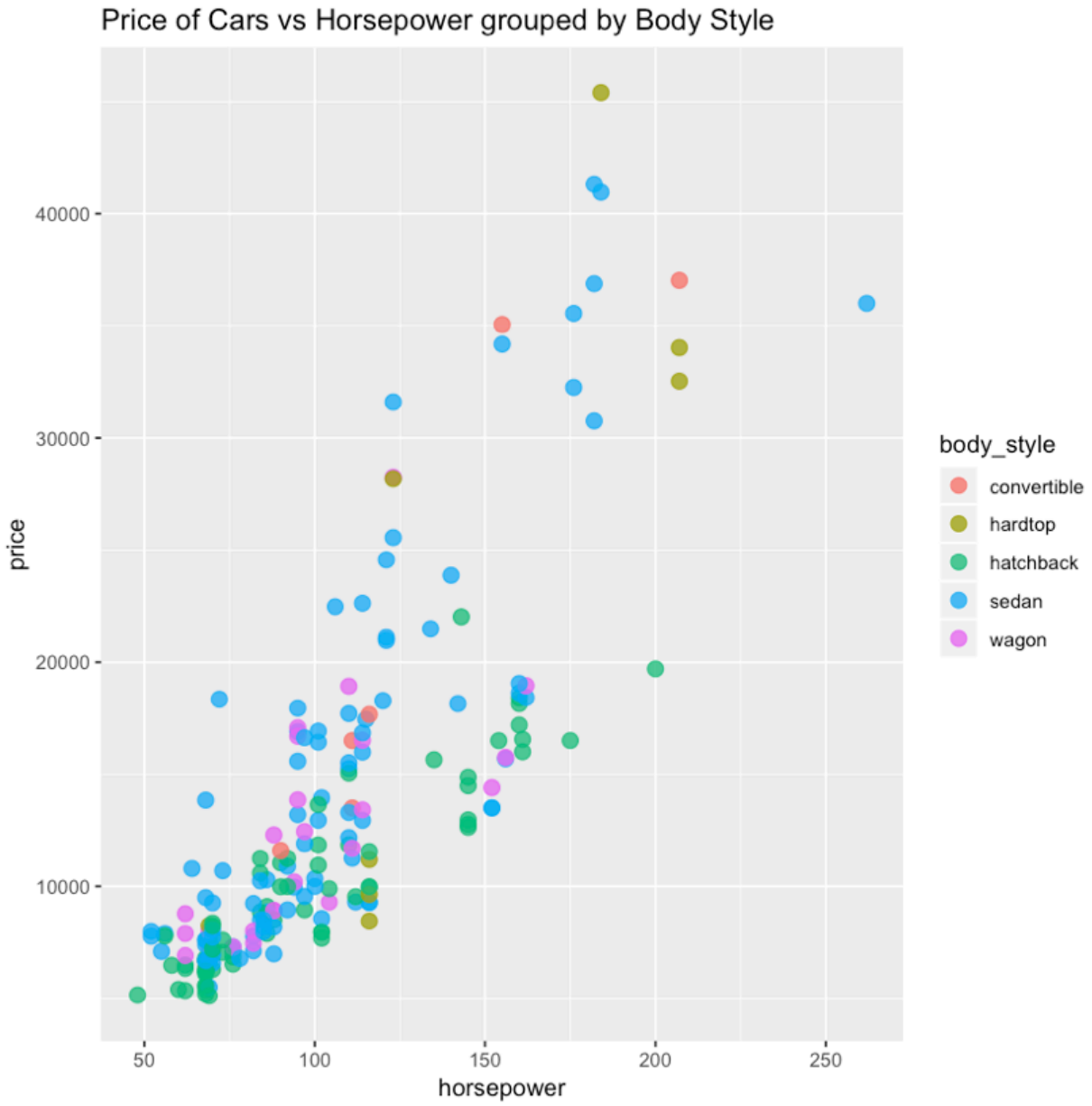
```
#ridgeline plot or joyplot)
ggplot(df, aes(x = price, y = body_style, fill = body_style )) +
      geom_density_ridges(bandwidth = 5000) +
      theme_ridges() +
      labs(x = 'Price', y = 'Body Style') +
      theme(legend.position = "none")
```



Grouped scatter plot

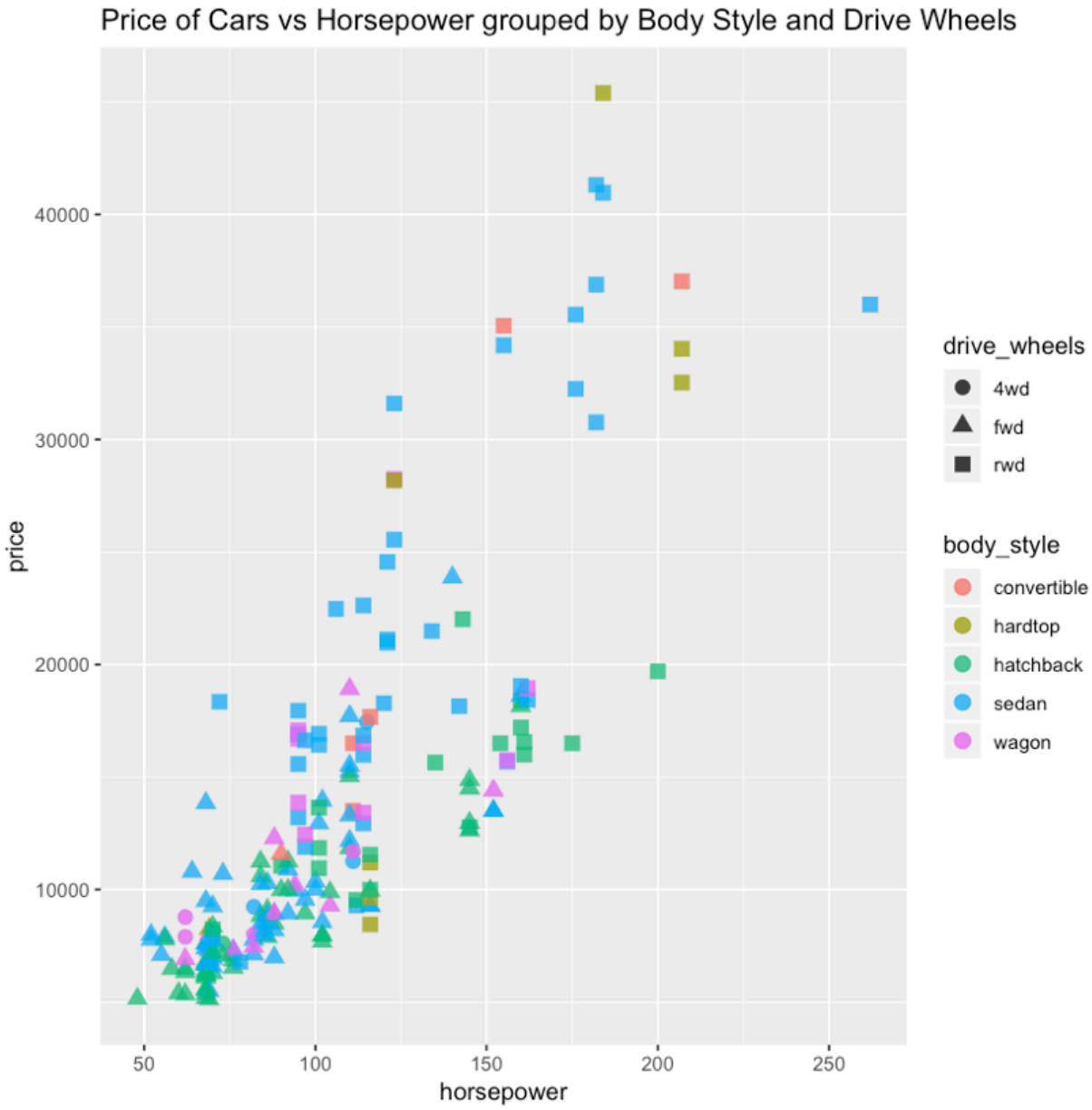
In [75]:

```
# grouped scatter plot
ggplot(df, aes(x = horsepower, y = price, color=body_style)) +
      geom_point(size = 3, alpha = .8) +
      labs(title = "Price of Cars vs Horsepower grouped by Body Style ")
```



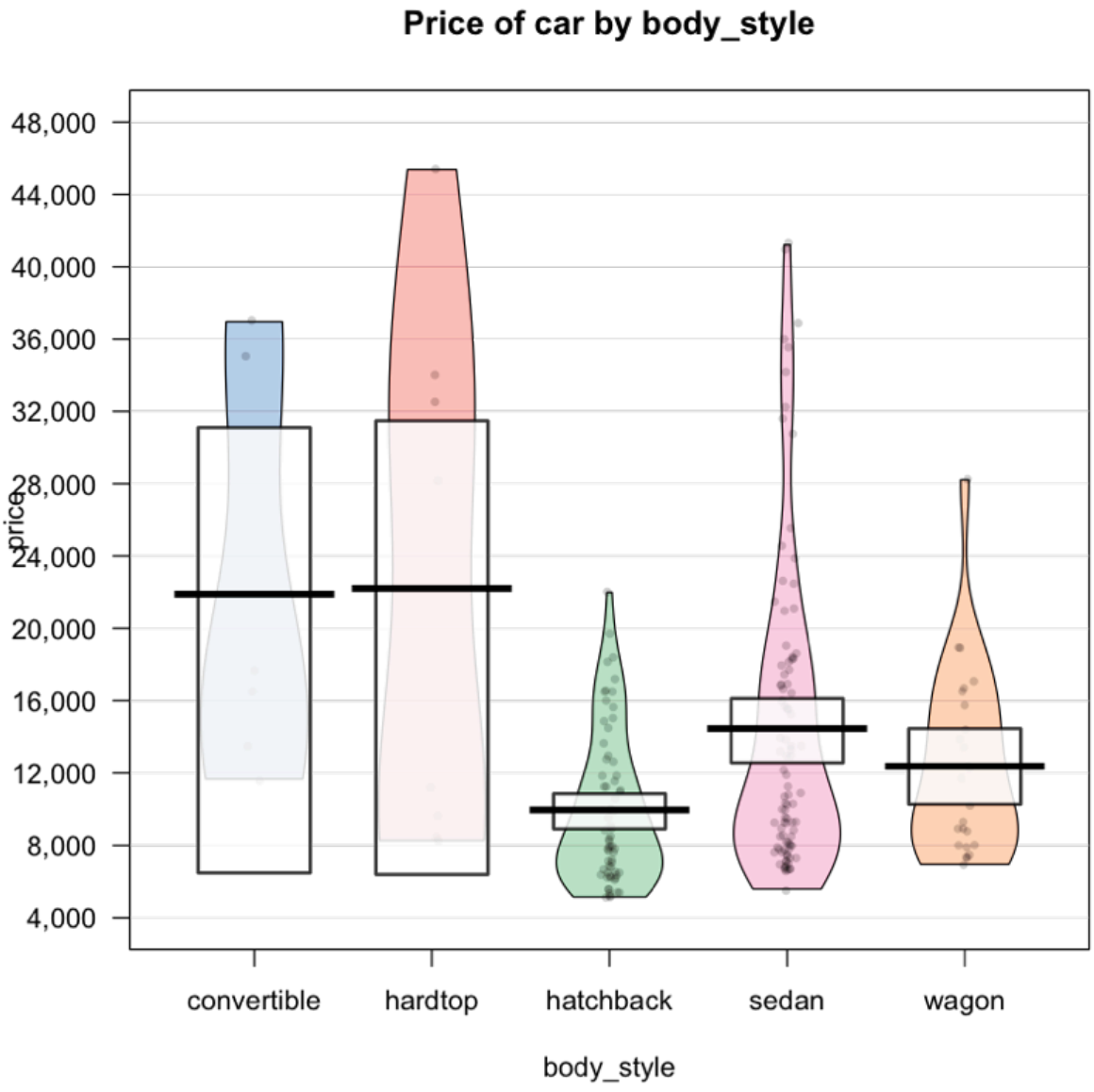
In [76]:

```
# grouped scatter plot
ggplot(df, aes(x = horsepower, y = price, color=body_style, shape = drive_wheels
)) +
      geom_point(size = 3, alpha = .8) +
      labs(title = "Price of Cars vs Horsepower grouped by Body Style and Drive
Wheels")
```



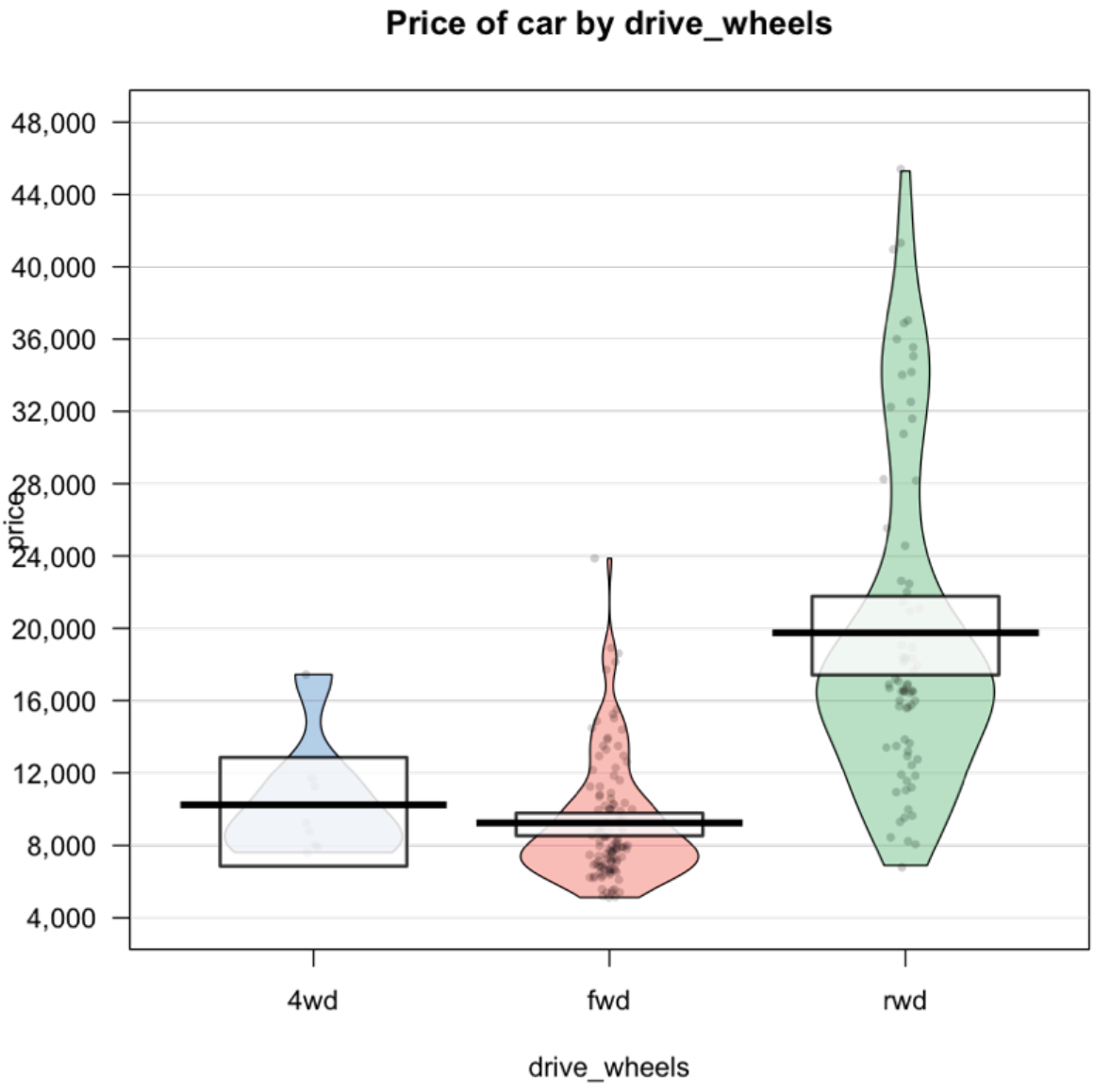
In [77]:

```
pirateplot(formula = price ~ body_style,
           data = df,
           main = "Price of car by body_style",
           xlab = "body_style",
           ylab = 'price',
           theme = 1
          )
```



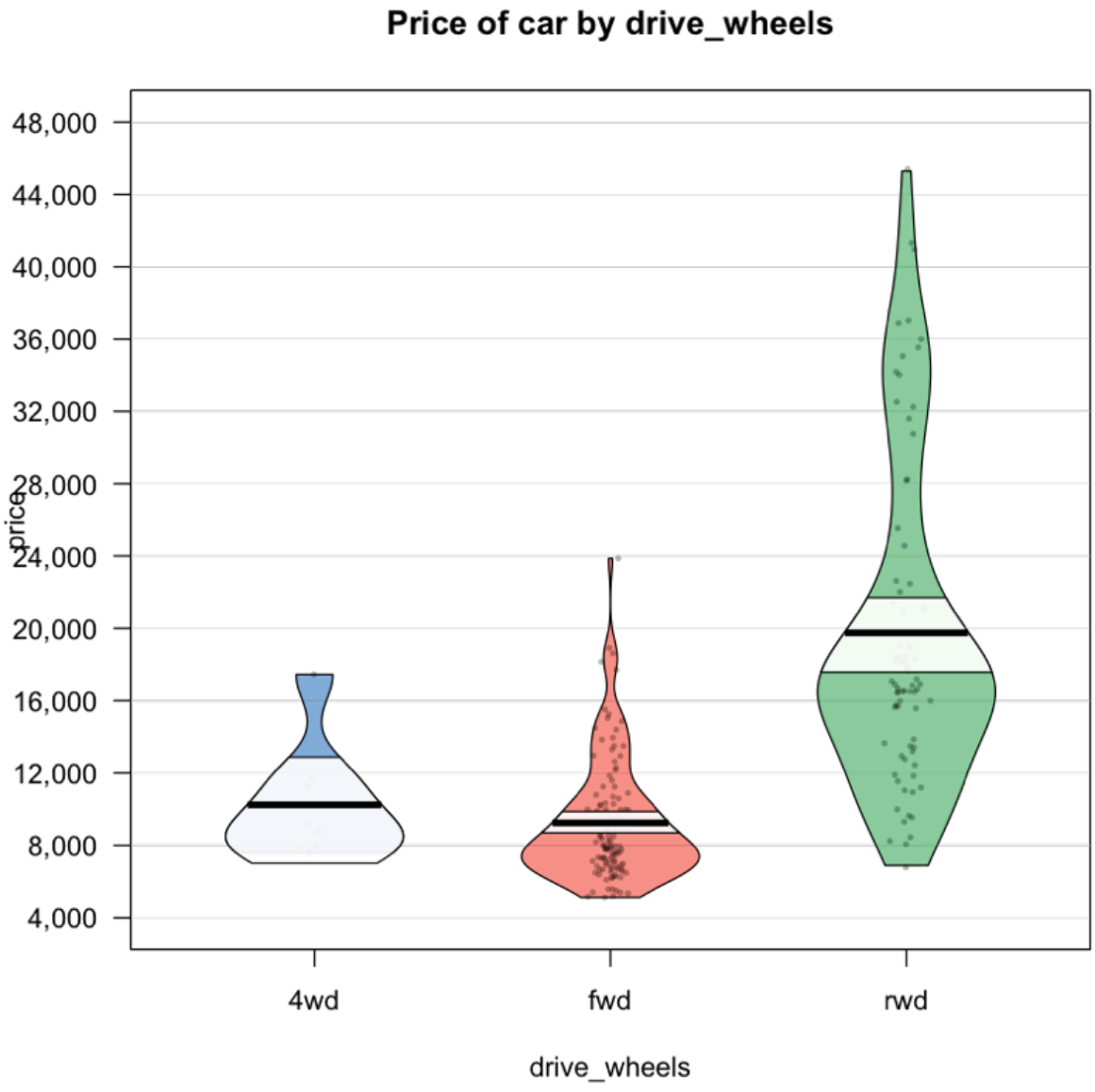
In [78]:

```
pirateplot(formula = price ~ drive_wheels,
           data = df,
           main = "Price of car by drive_wheels",
           xlab = "drive_wheels",
           ylab = 'price',
           theme = 1
          )
```



In [79]:

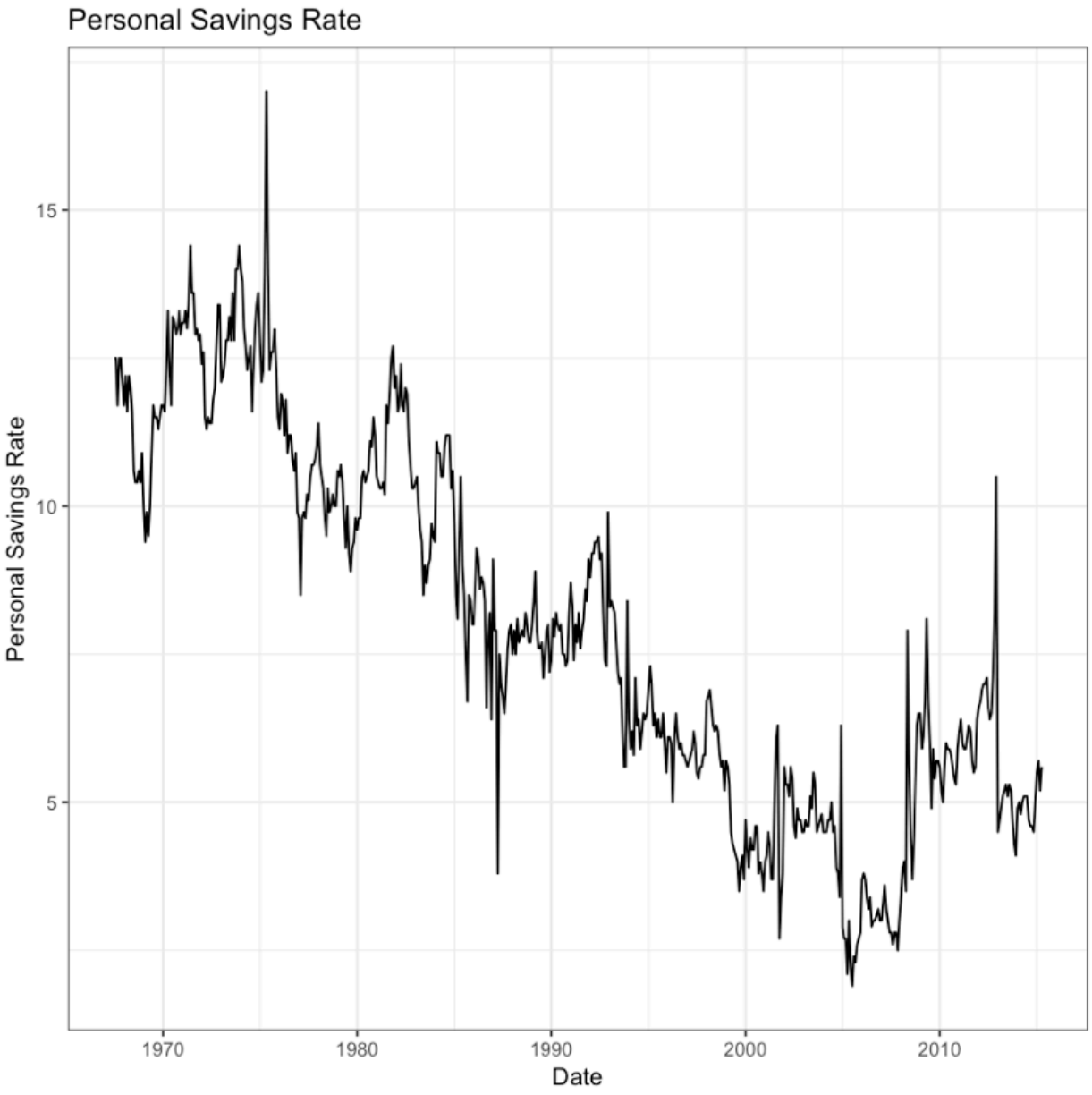
```
pirateplot(formula = price ~ drive_wheels,
           data = df,
           main = "Price of car by drive_wheels",
           xlab = "drive_wheels",
           ylab = 'price',
           theme = 3
          )
```



Time series

In [80]:

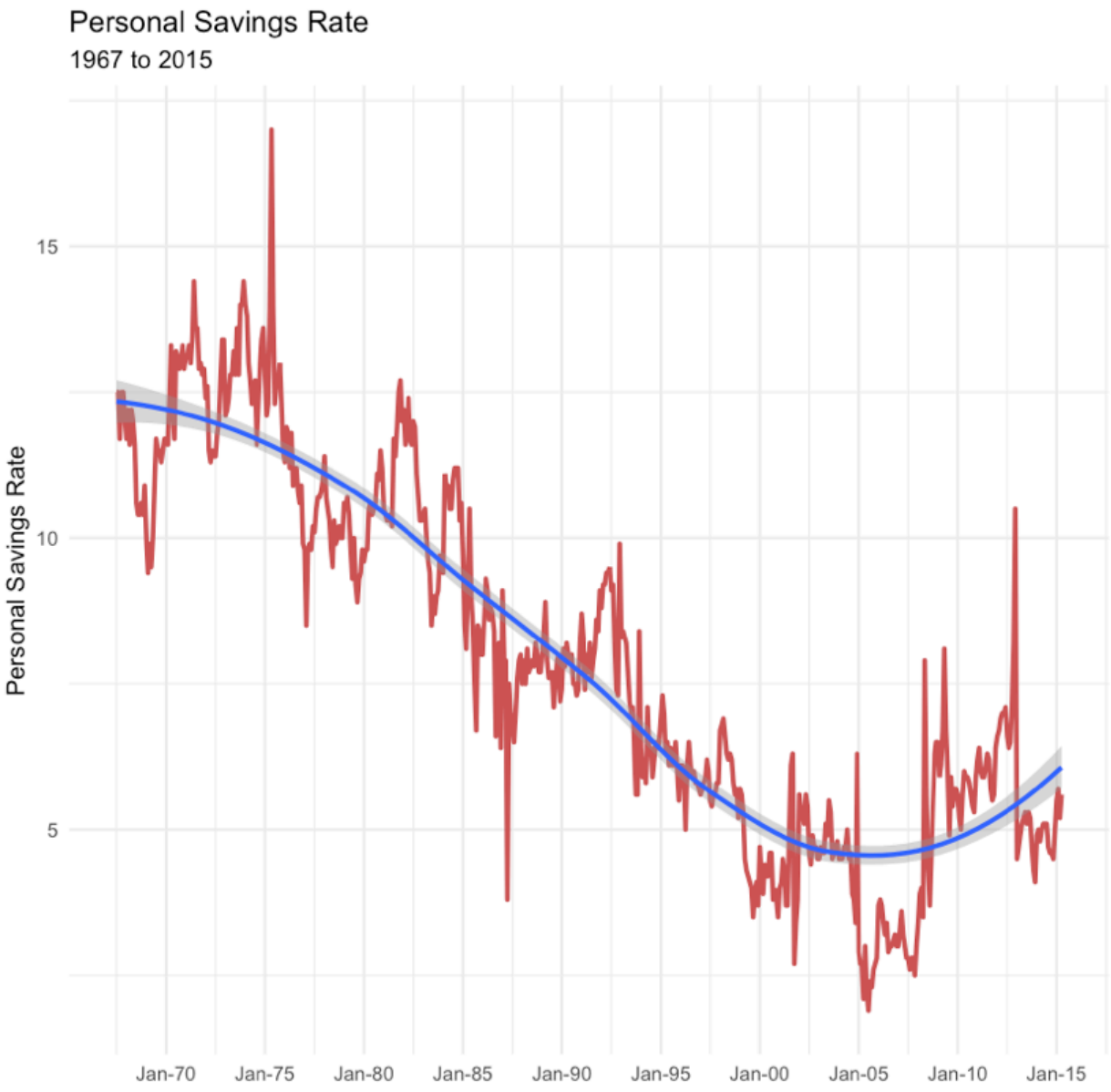
```
data(economics, package="ggplot2")
ggplot(economics, aes(x = date, y = psavert)) +
       geom_line() +
       theme_bw()+
       labs(title = "Personal Savings Rate", x = "Date", y = "Personal Savings R
ate")
```



In [81]:

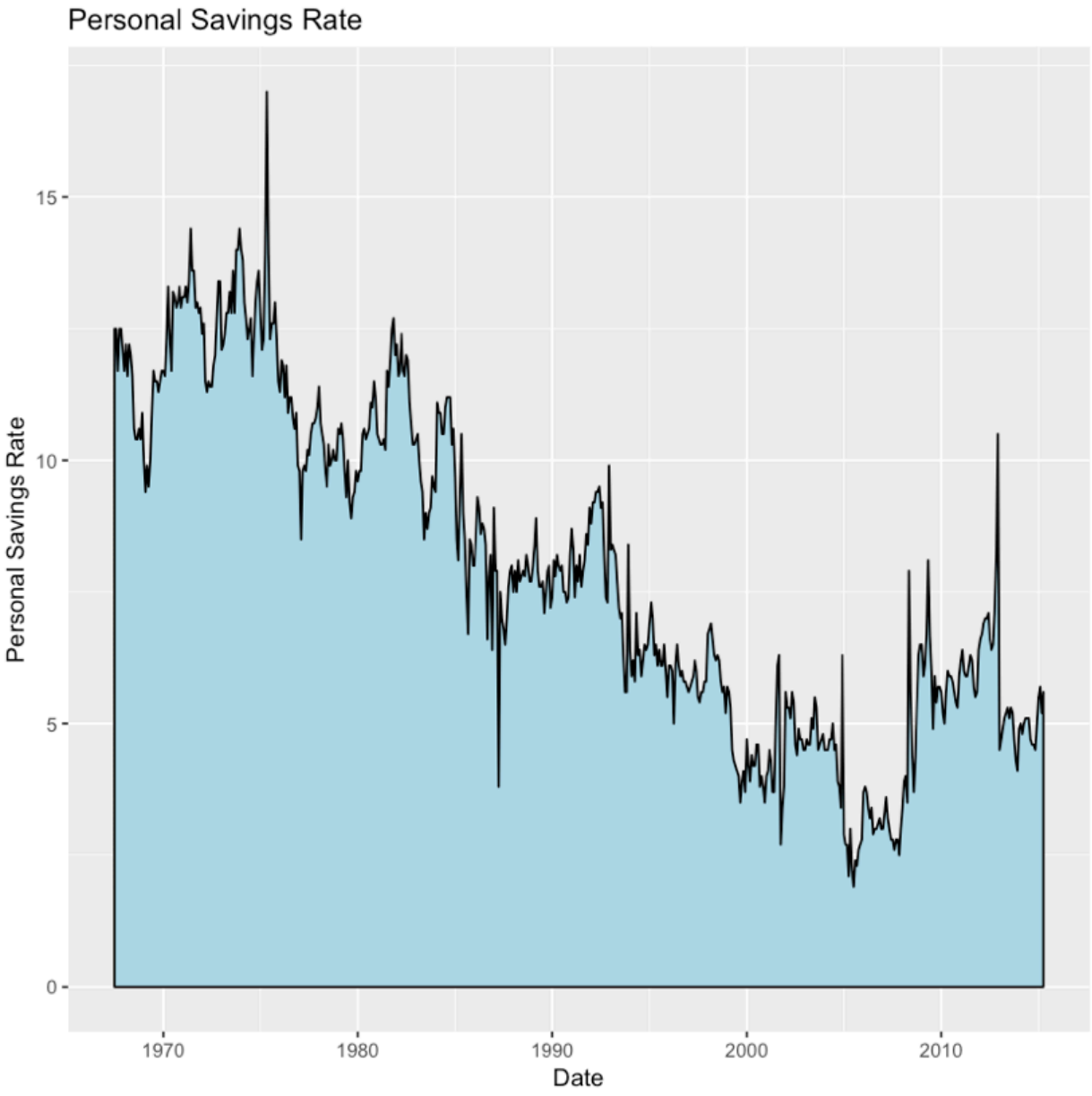
```
# Time Series
ggplot(economics, aes(x = date, y = psavert)) +
       geom_line(color = "indianred3", size=1 ) +
       geom_smooth(formula = y ~ x) +
       scale_x_date(date_breaks = '5 years', labels = date_format("%b-%y")) +
       labs(title = "Personal Savings Rate", subtitle = "1967 to 2015", x = "",
            y = "Personal Savings Rate") +
       theme_minimal()
```

```
`geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



In [82]:

```
# Area Chart
ggplot(economics, aes(x = date, y = psavert)) +
  geom_area(fill="lightblue", color="black") +
  labs(title = "Personal Savings Rate",
       x = "Date",
       y = "Personal Savings Rate")
```



Maps

Basic maps

In [4]:

```
library(readr)
library(ggplot2)
library(dplyr)
library(yarrr)
library(e1071)
library(usmap)
library(maps)
library(ggmap)
```

```
Google's Terms of Service: https://cloud.google.com/maps-platform/te
rms/.
Please cite ggmap if you use it! See citation("ggmap") for details.
```

In [6]:

```
# Map of usa
map(database='state')
title('Map of the United States')
```


Map of the United States

In [7]:

```
map('state', col = "darkgray", fill = TRUE, border = "white")
# add a title to your map
title('Map of the United States')
```

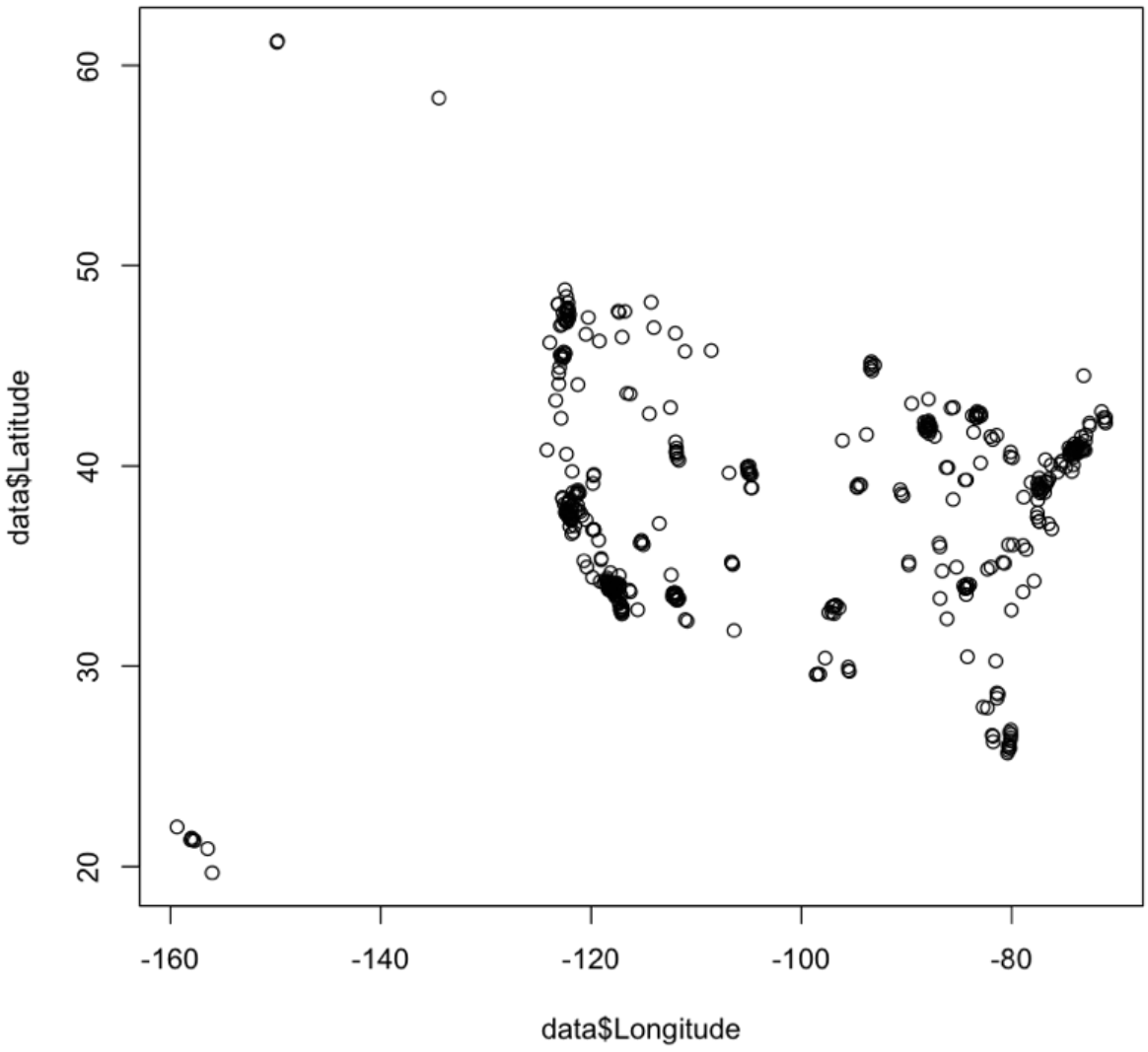
In [8]:

```
data <- read.csv('ABC_locations.csv')
head(data)
```

Address	City	State	Zip.Code	Latitude	Longitude
1205 N. Memorial Parkway	Huntsville	Alabama	35801-5930	34.74309	-86.60096
3650 Galleria Circle	Hoover	Alabama	35244-2346	33.37765	-86.81242
8251 Eastchase Parkway	Montgomery	Alabama	36117	32.36389	-86.15088
5225 Commercial Boulevard	Juneau	Alaska	99801-7210	58.35920	-134.48300
330 West Dimond Blvd	Anchorage	Alaska	99515-1950	61.14327	-149.88422
4125 DeBarr Road	Anchorage	Alaska	99508-3115	61.21081	-149.80434

In [8]:

```
plot(data$Longitude, data$Latitude)
```


data$Latitude
60
50
40
30
20
-160
-140
-120
-100
-80
data$Longitude

In [9]:

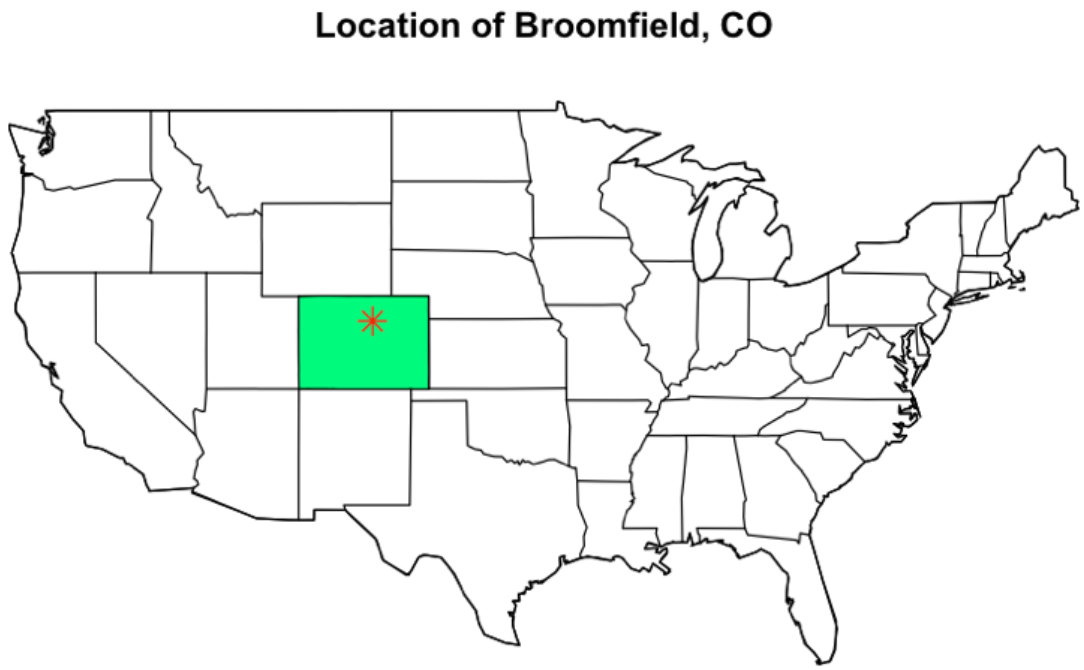
```
# Map of usa
usmap::plot_usmap("states", labels = TRUE)
```

In [10]:

```
map('state')
#make changes on the map of usa
map(database = "usa", col = "green", lwd = 1, add = TRUE)
# add locations
title("Location of super Stores in US")
# add the x, y for the location of the stores
#pch = shape, cex = size
points(x = data$Longitude, y = data$Latitude, pch = 1, col = "red", cex = 0.5)
```

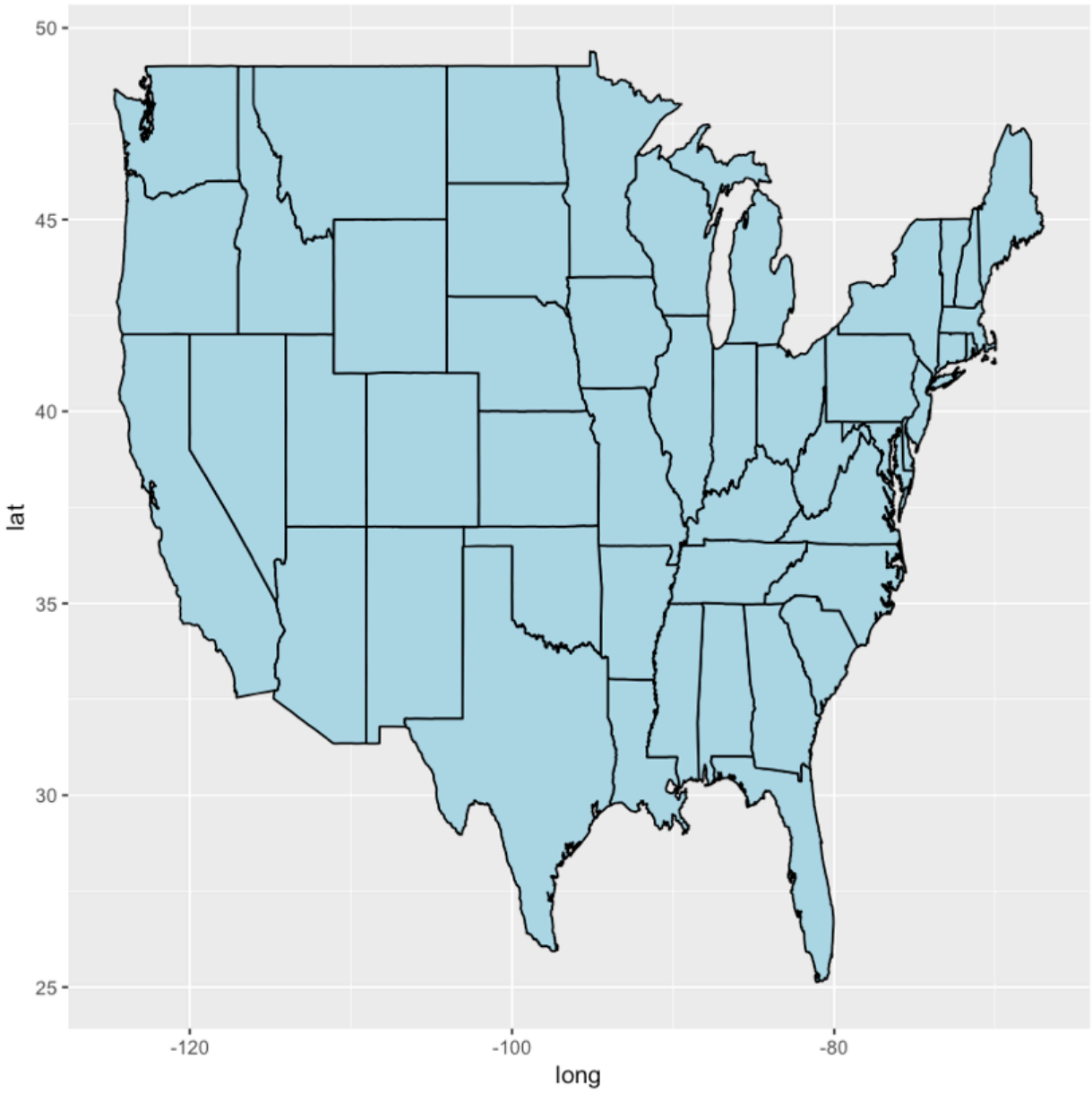
In [11]:

```
map('state')
map(database = "usa", lwd = 1, add = TRUE)
# add the adjacent parts of the US; can't forget my homeland
map("state", "colorado", col = "springgreen",
    lwd = 1, fill = TRUE, add = TRUE)
# add gage location
title("Location of Broomfield, CO")
# add the x, y for the location of Broomfield CO
#pch = shape, cex = size
points(x = -105.086647, y = 39.920540, pch = 8, col = "red", cex = 1.5)
```



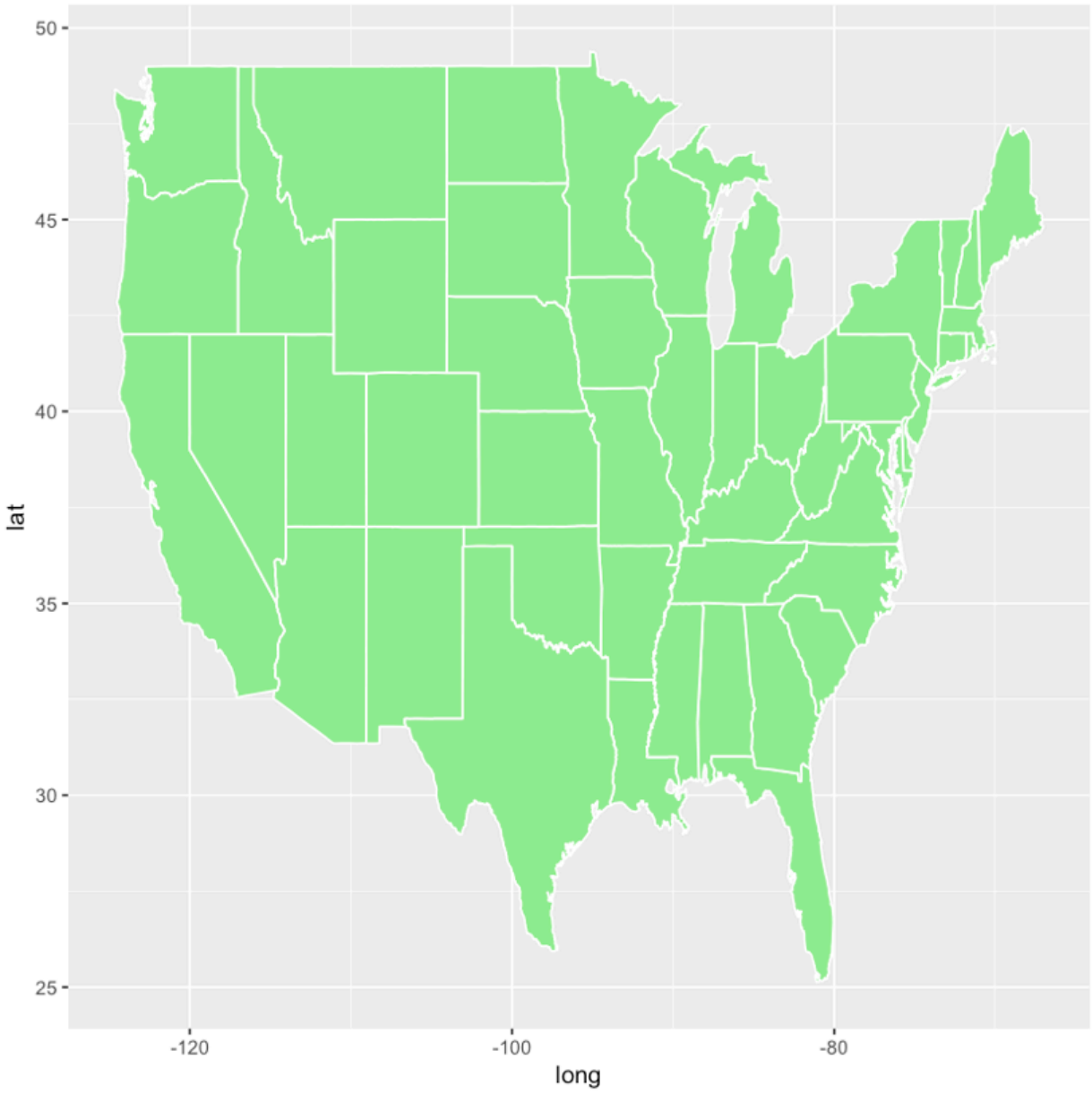
In [12]:

```
ggplot() +
  geom_polygon( data=map_data("state"), aes(x=long, y=lat, group=group),
                color="black", fill="lightblue" )
```



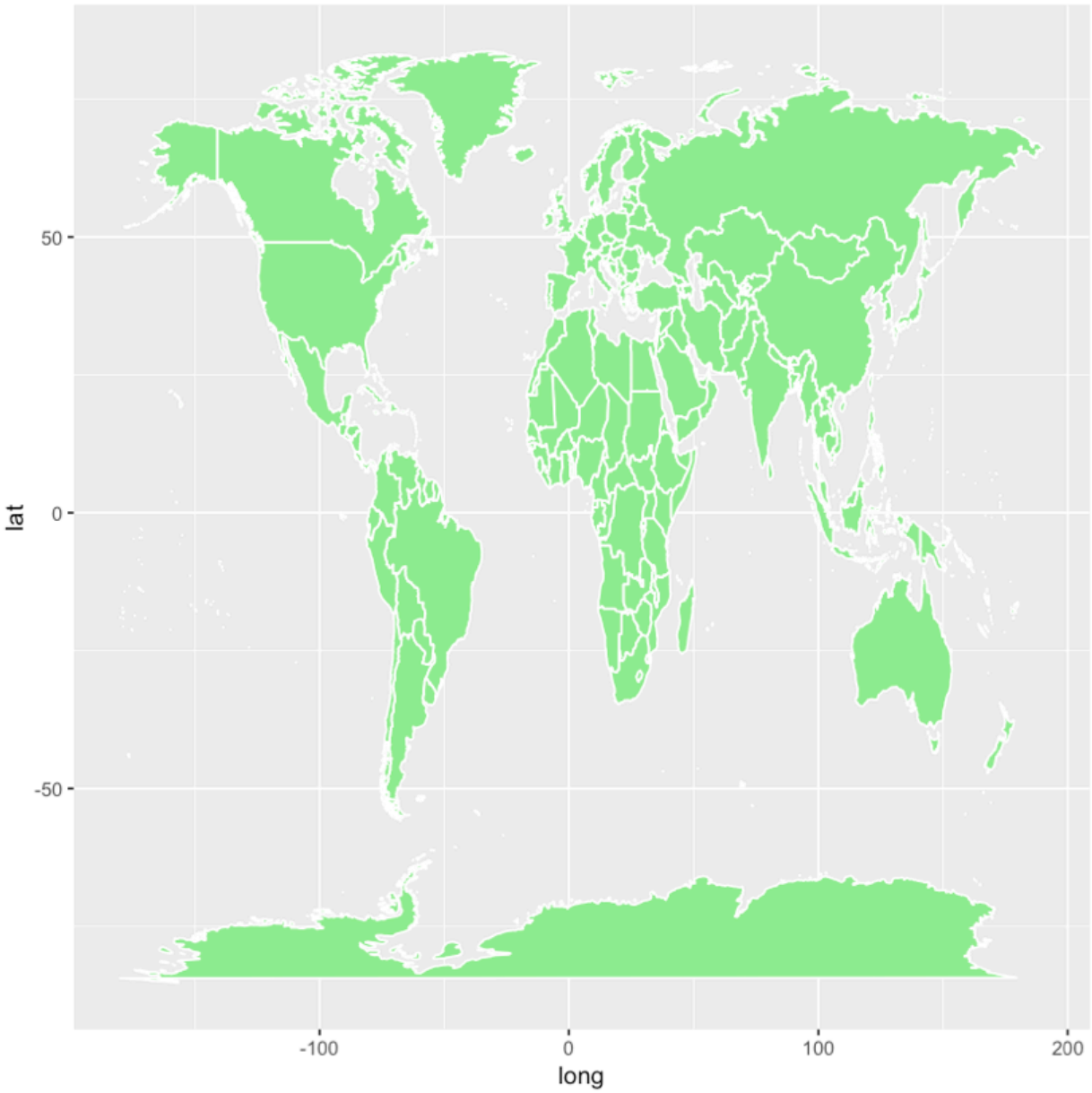
In [13]:

```
us<-map_data("state")
ggplot(us, aes(x = long, y = lat, group = group)) +
     geom_polygon(fill="lightgreen", colour = "white")
```



In [14]:

```
world_map <- map_data("world")
ggplot(world_map, aes(x = long, y = lat, group = group)) +
     geom_polygon(fill="lightgreen", colour = "white")
```



In [ ]:

```

```