Viz_plot

Pankaj Shah

6/21/2019

Contents

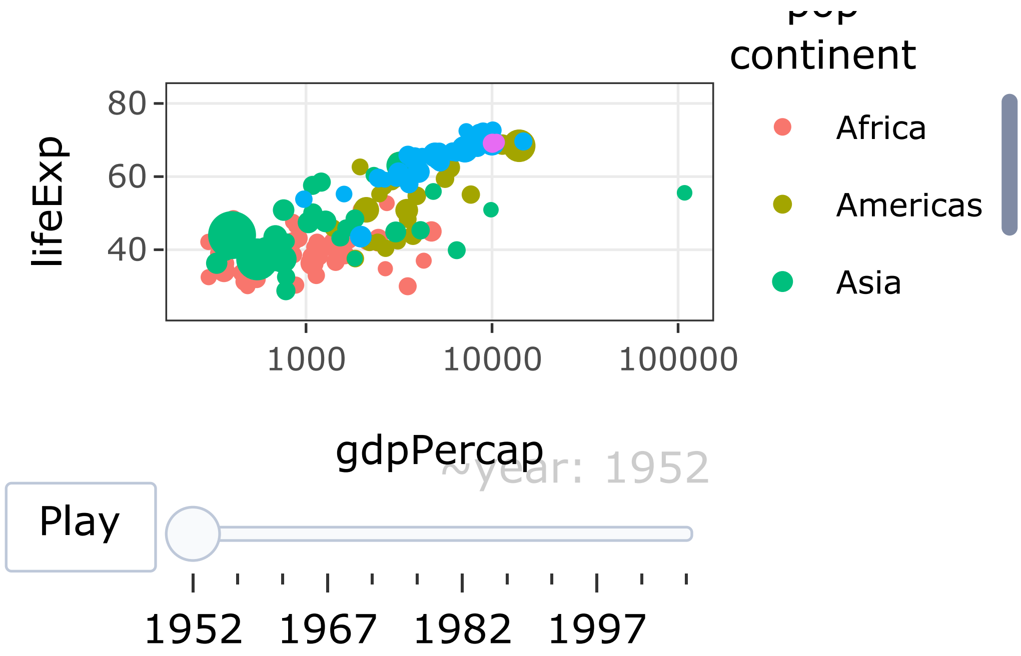
```r
library(ggplot2)
options(scipen=999) # turn-off scientific notation like 1e+48
theme_set(theme_bw()) # pre-set the bw theme.
data("midwest", package = "ggplot2")
# ALT way: midwest <- read.csv("http://goo.gl/G1K41K")
```

- geom_bar(): Bar chart color, fill, alpha
- geom_boxplot(): Box plot color, fill, alpha, notch, width
- geom_density(): Density plot color, fill, alpha, linetype
- geom_histogram(): Histogram color, fill, alpha, linetype, binwidth
- geom_hline(): Horizontal lines color, alpha, linetype, size
- geom_jitter(): Jittered points color, size, alpha, shape
- geom_line(): Line graph colorvalpha, linetype, size
- geom_point(): Scatterplot color, alpha, shape, size
- geom_rug(): Rug plot color, side
- geom_smooth(): Fitted line method, formula, color, fill, linetype, size
- geom_text(): Text annotations Many; see the help for this function
- geom_violin(): Violin plot color, fill, alpha, linetype
- geom_vline(): Vertical lines color, alpha, linetype, size

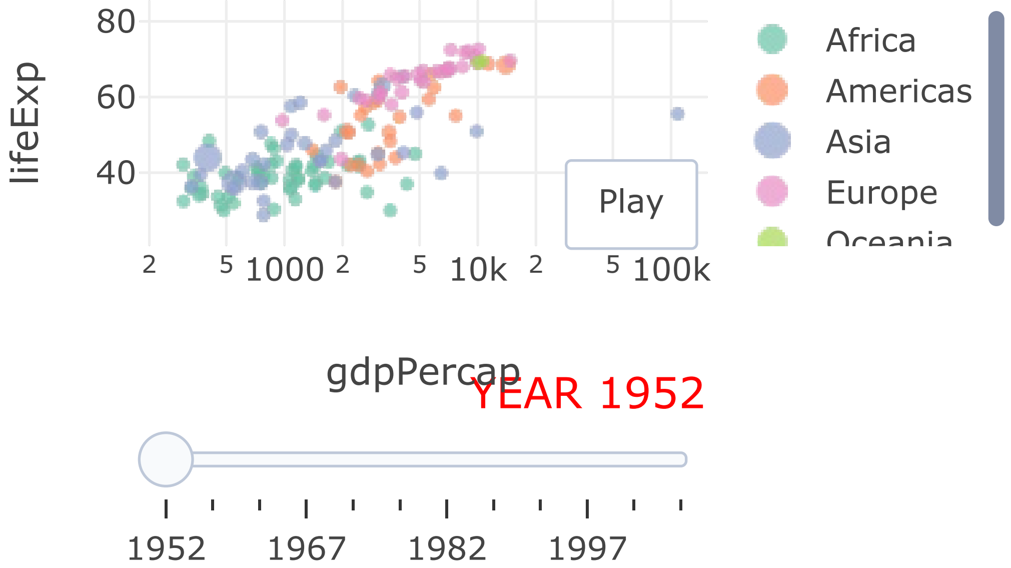
- color: colour of points, lines, and borders around filled regions
- fill: colour of filled areas such as bars and density regions
- alpha: transparency of colors, ranging from 0 (fully transparent) to 1 (opaque)
- linetype: pattern for lines (1 = solid, 2 = dashed, 3 = dotted, 4 = dotdash, 5 = longdash, 6 = * twodash)
- size: point size and line width
- shape: point shapes (same as pch, with 0 = open square, 1 = open circle, 2 = open triangle, and so on)
- position: position of plotted objects such as bars and points.
- dodge : side by side,
- stacked : vertically stacks grouped bar charts,
- fill :vertically stacks grouped bar charts and standardizes their heights to be equal;
- jitter: reduces point overlap
- binwidth: bin width for histograms
- notch: indicates whether box plots should be notched (TRUE/FALSE)
- sides: placement of rug plots on the graph ("b" = bottom, "l" = left, "t" = top, "r" = right, "bl" = both bottom and left, and so on)
- width: width of box plots

Animation Plot

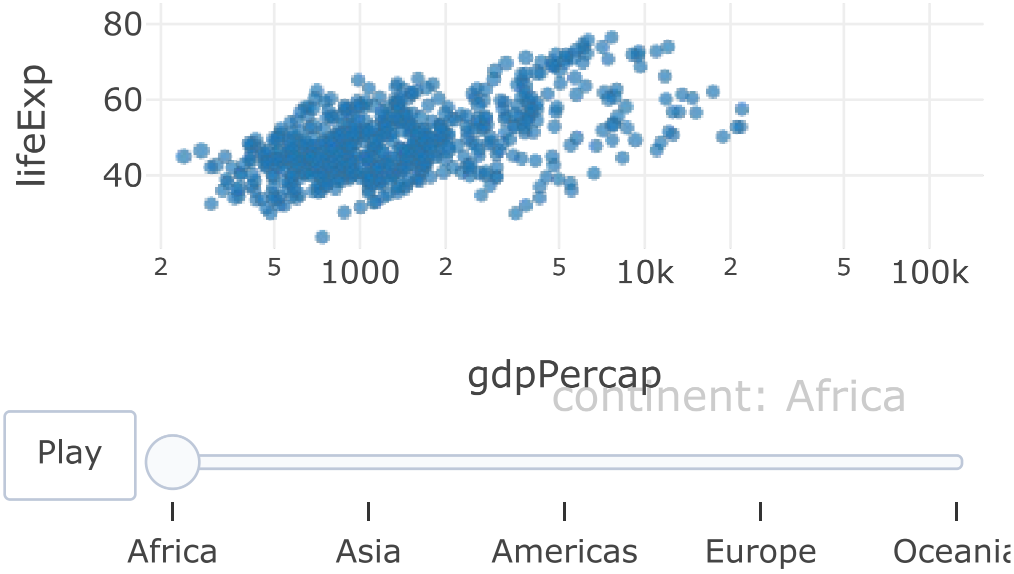
```r
# Library

library(ggplot2)
library(plotly)

#Data
data(gapminder, package = "gapminder")

# Plot
ani_plot <- gapminder %>%
            ggplot(aes(gdpPercap,lifeExp, color = continent ))+
            geom_point(aes(size = pop, frame = year, ids = country)) +
            scale_x_log10()

ggplotly(ani_plot)
```



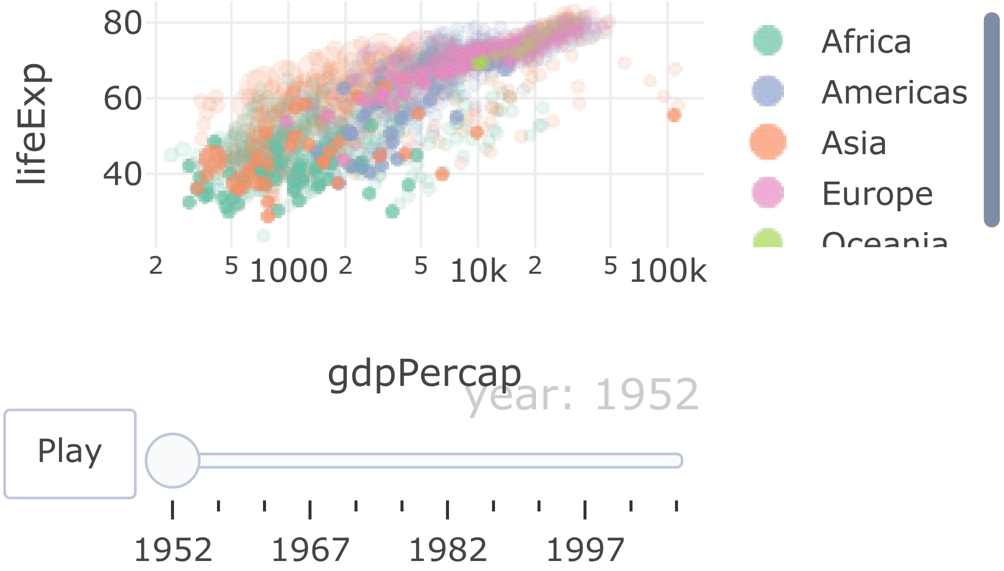
```
gapminder %>%
  plot_ly(x = ~gdpPercap, y = ~lifeExp, size = ~pop,text = ~country, hoverinfo = "text") %>% # Hoverinf
  layout(xaxis = list(type = "log"))%>%
  add_markers(color = ~continent, frame = ~year, ids = ~country) %>% # Markers
  animation_opts(1000, easing = "elastic", redraw = FALSE) %>%
  animation_button(x = 1, xanchor = "right", y = 0, yanchor = "bottom") %>% # Animation Button
  animation_slider(currentvalue = list(prefix = "YEAR ", font = list(color="red"))) # Animation Slider
```



```
# Calculate mean
meanLife <- with(gapminder, tapply(lifeExp, INDEX = continent, mean))

# define continent
gapminder$continent <- factor(gapminder$continent, levels = names(sort(meanLife)))

gapminder %>%
  plot_ly(x = ~gdpPercap, y = ~lifeExp, size = ~pop,text = ~country, hoverinfo = "text") %>%
  layout(xaxis = list(type = "log")) %>%
  add_markers(data = gapminder, frame = ~continent) %>%
  hide_legend() %>%
  animation_opts(frame = 1000, transition = 0, redraw = FALSE)
```

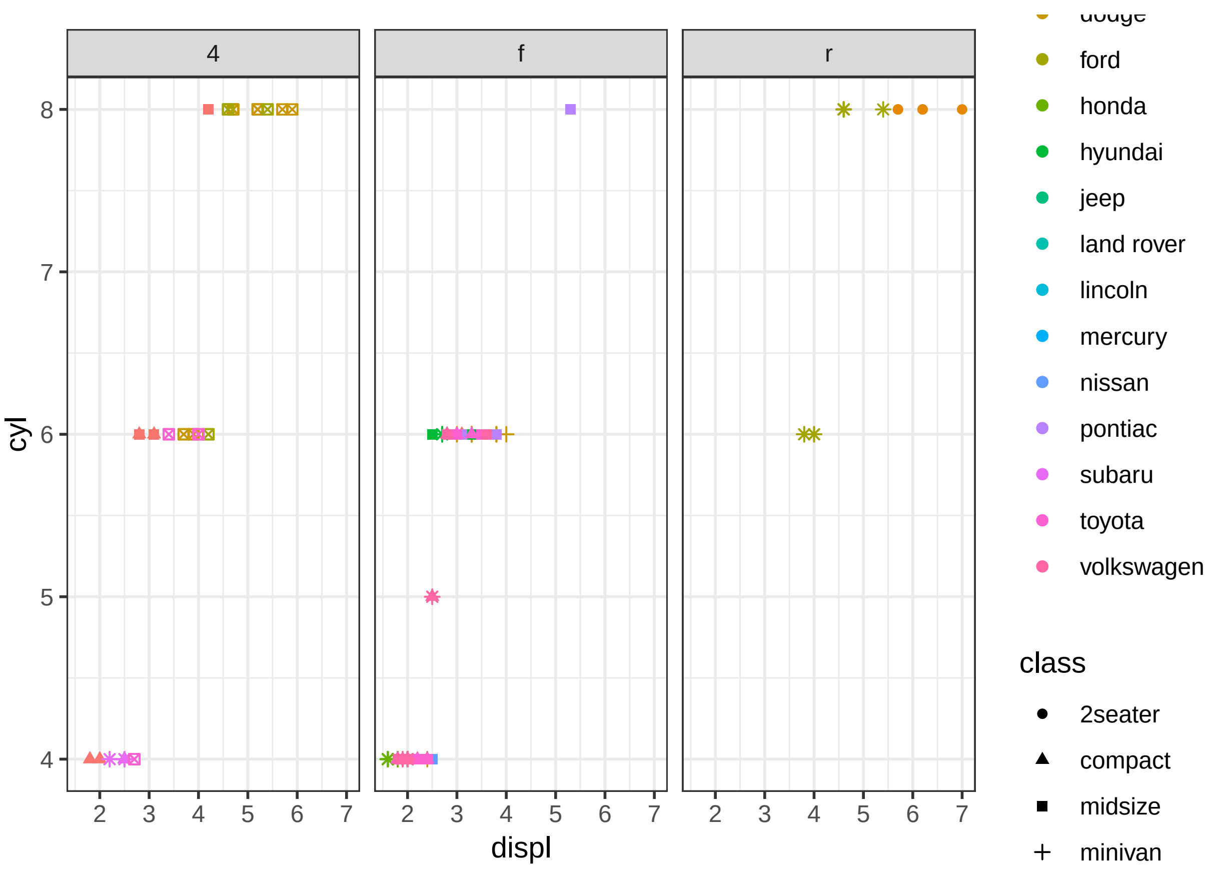


```
gapminder %>%
  plot_ly(x = ~gdpPercap, y = ~lifeExp, size = ~pop,text = ~country, hoverinfo = "text") %>%
  layout(xaxis = list(type = "log")) %>%
  add_markers(color = ~continent, alpha = 0.2, alpha_stroke = 0.2, showlegend = F) %>%
  add_markers(color = ~continent, frame = ~year, ids = ~country) %>%
  animation_opts(1000, redraw = FALSE)
```

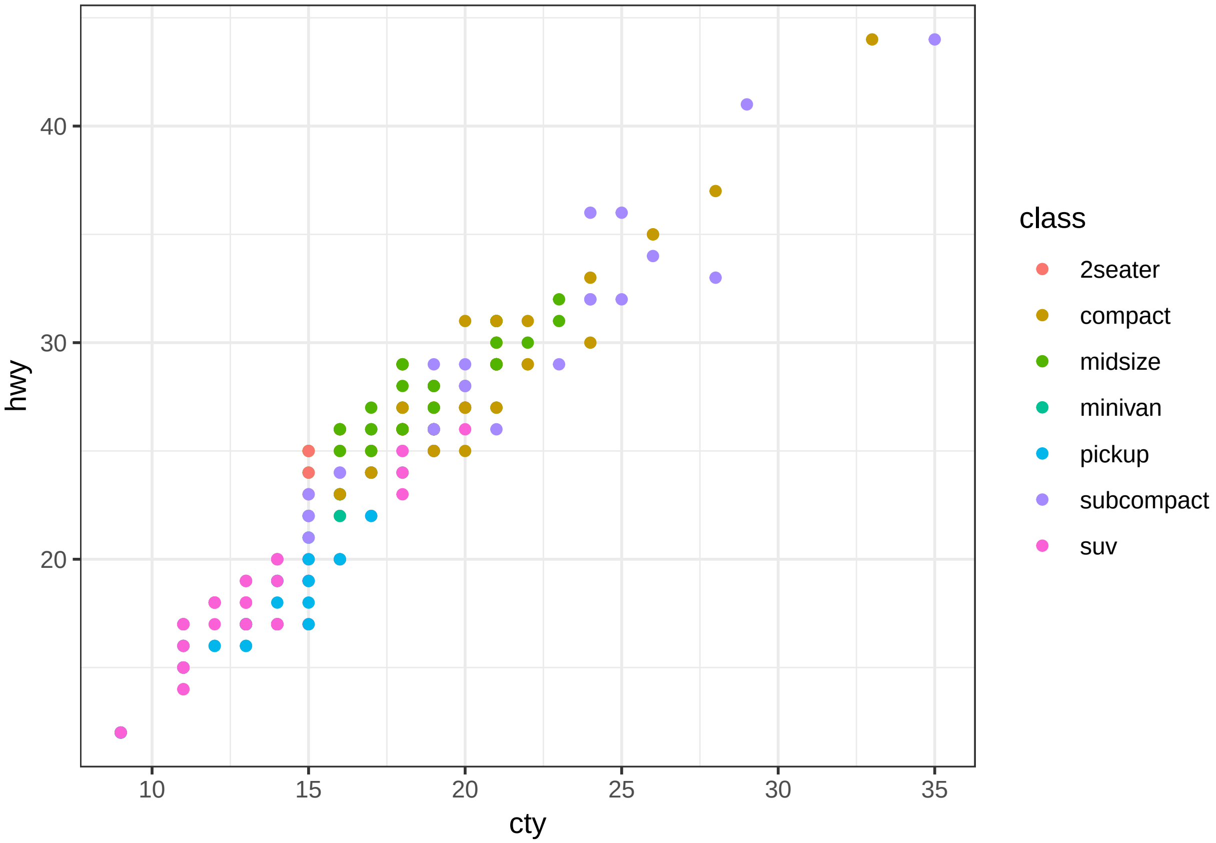


Scatterplot

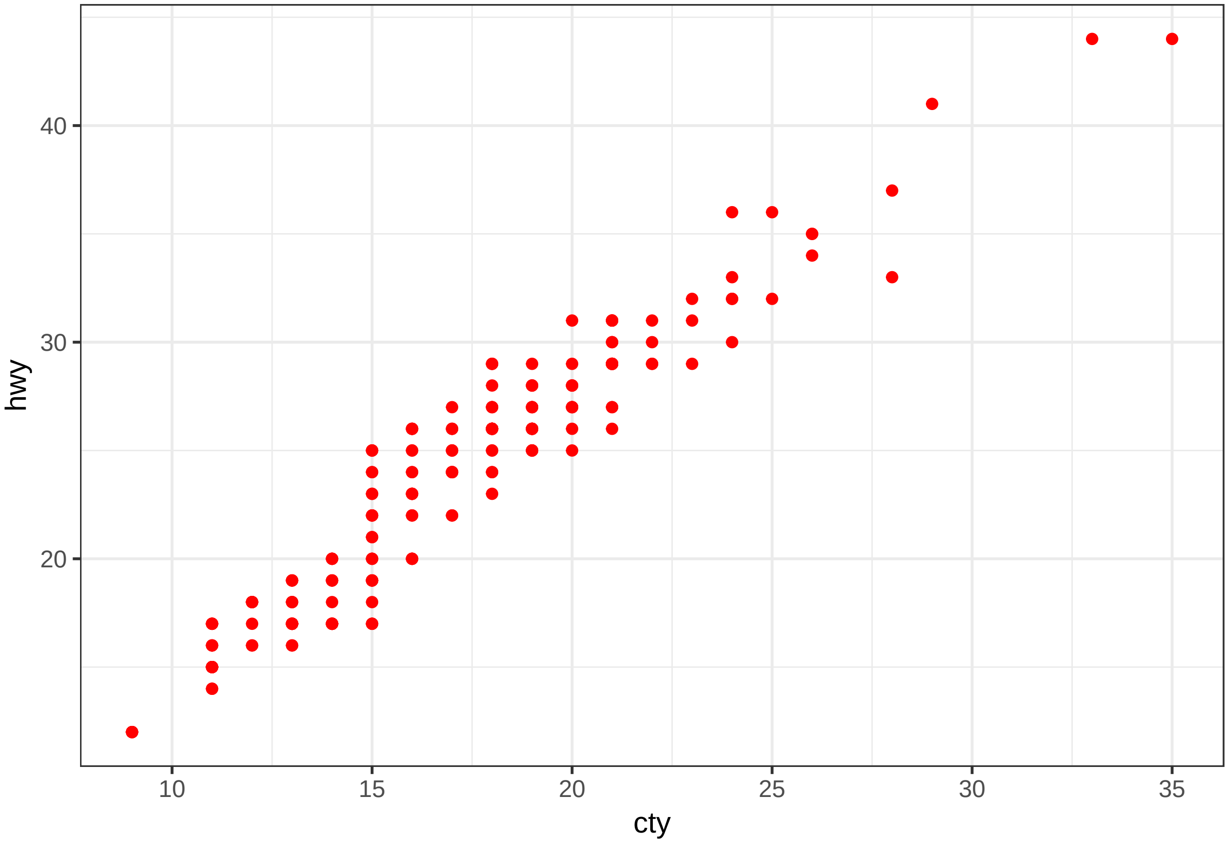
```
# Library
library(ggplot2)

# A:
mpg %>%
ggplot(aes(x=displ, y=cyl, color=manufacturer, shape=class)) +
  geom_point() +
  facet_grid(.~drv)
```



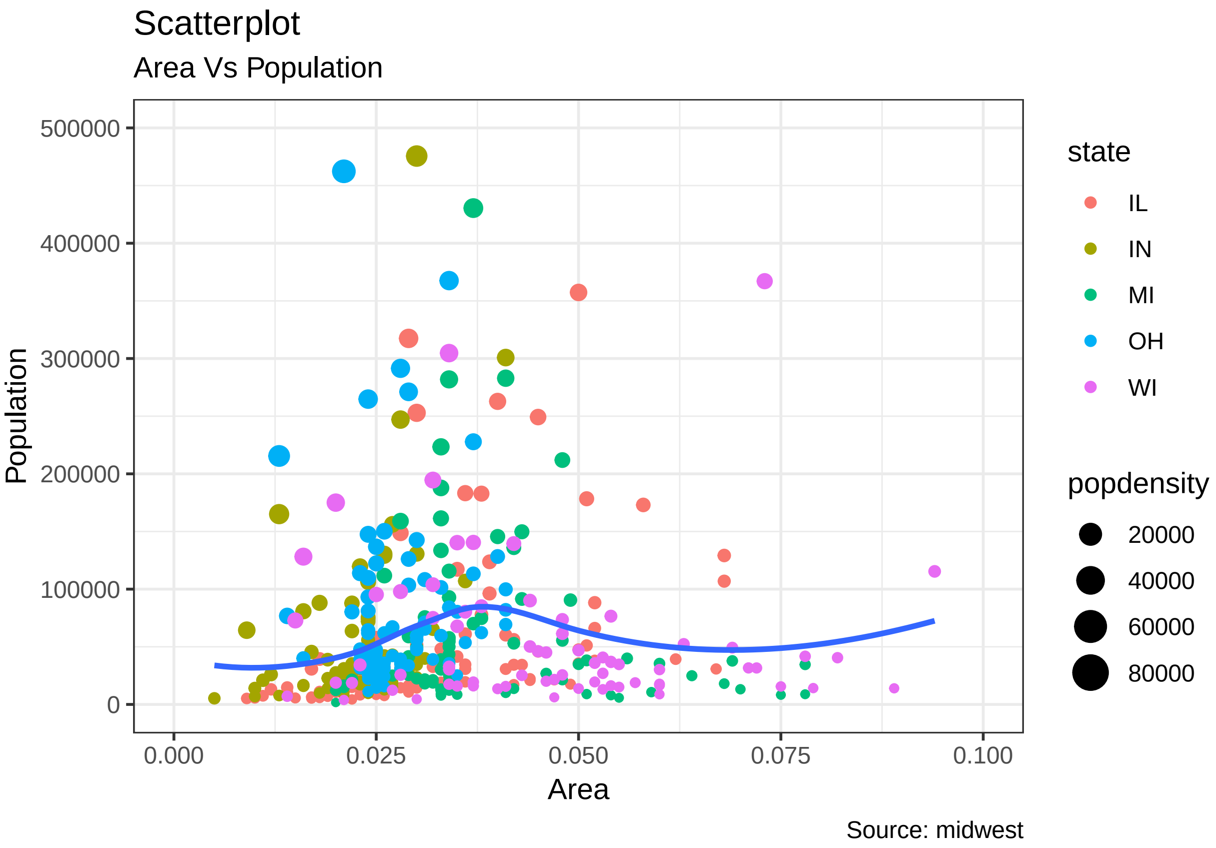
```
# B:
ggplot(mpg, aes(cty, hwy)) + geom_point(aes(colour = class))
```



```
# C:
ggplot(mpg, aes(cty, hwy)) + geom_point(colour = "red") # having one color througout
```



```
# D:
plot_1 <- midwest %>%
            ggplot(aes(x=area, y=poptotal)) +
            geom_point(aes(col=state, size=popdensity)) +
            geom_smooth(method="loess", se=F) +
            xlim(c(0, 0.1)) +
            ylim(c(0, 500000)) +
            labs(subtitle="Area Vs Population", y="Population",
                   x ="Area",
                   title ="Scatterplot",
                    caption = "Source: midwest")
plot(plot_1)
```

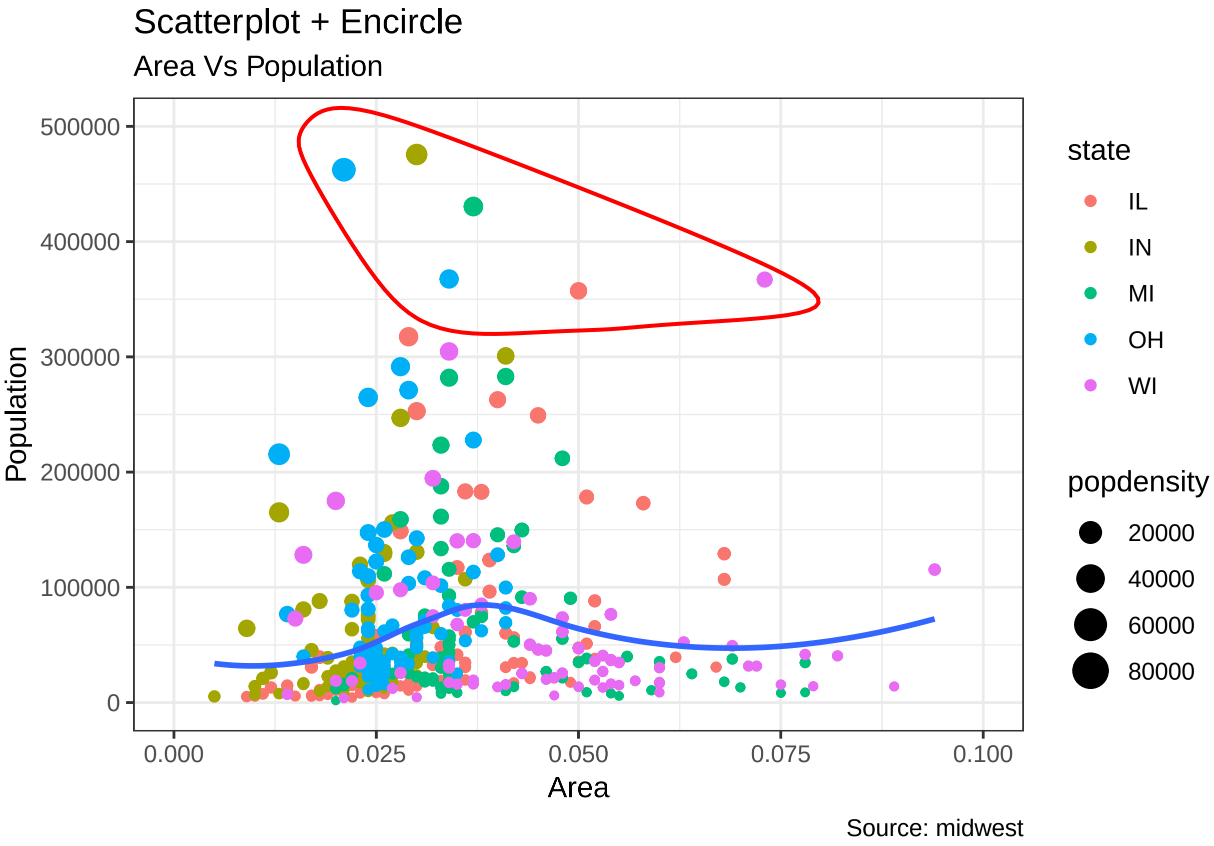


Scatterplot With Encircling

This can be conveniently done using the geom_encircle() in ggalt package.

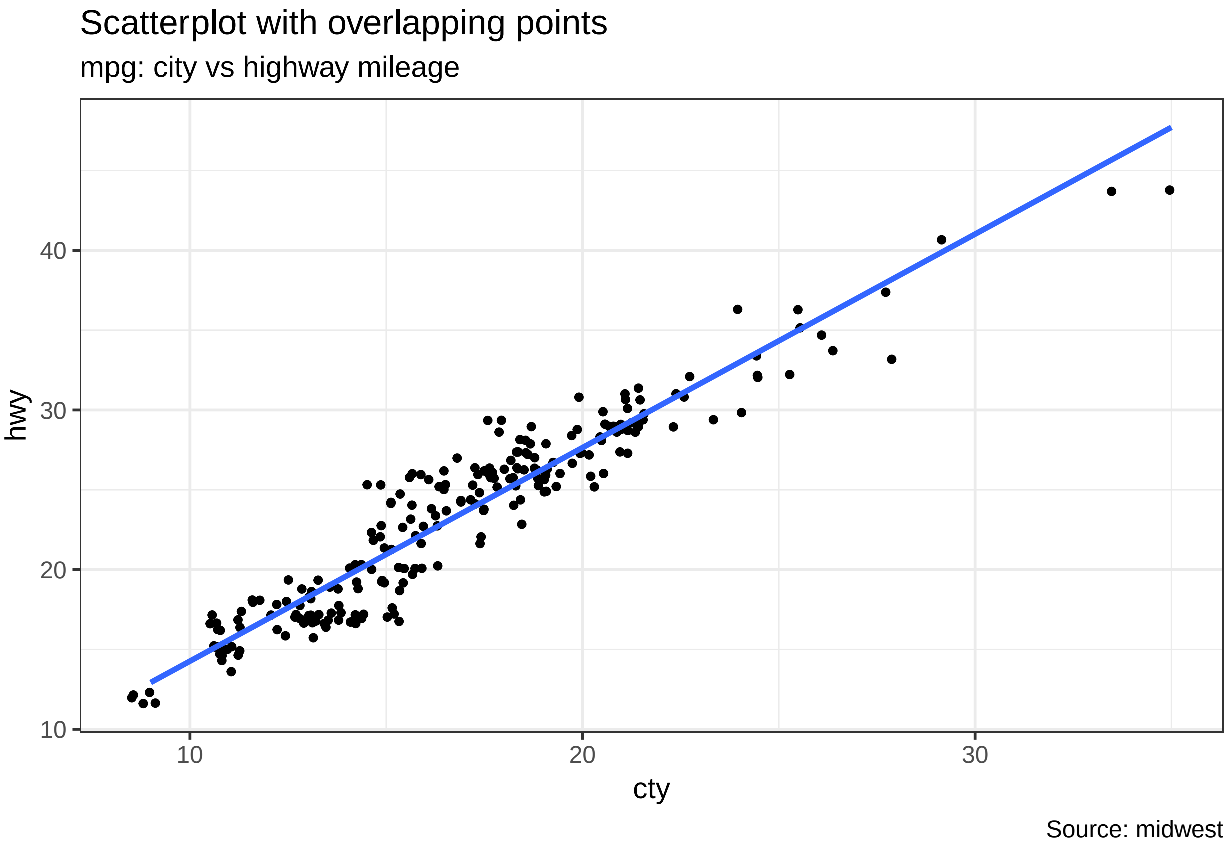
```
options(scipen = 999)

# Library
library(ggplot2)
library(ggalt)

# Data
midwest_select <- midwest %>%
                    filter(poptotal >350000 & poptotal <= 500000 & area > 0.01 & area < 0.1)

# Plot
midwest %>%
  ggplot(aes(x= area, y = poptotal))+
  geom_point(aes(col=state, size=popdensity)) +
  geom_smooth(method="loess", se=F) +
  xlim(c(0, 0.1)) +
  ylim(c(0, 500000)) + # draw smoothing line
  geom_encircle(aes(x=area, y=poptotal),
                data=midwest_select,
                color="red",
                size=2,
                expand=0.08) +    # encircle
```

```
labs(title ="Scatterplot + Encircle",
     subtitle ="Area Vs Population",
     x ="Area",
     y ="Population",
     caption="Source: midwest")
```



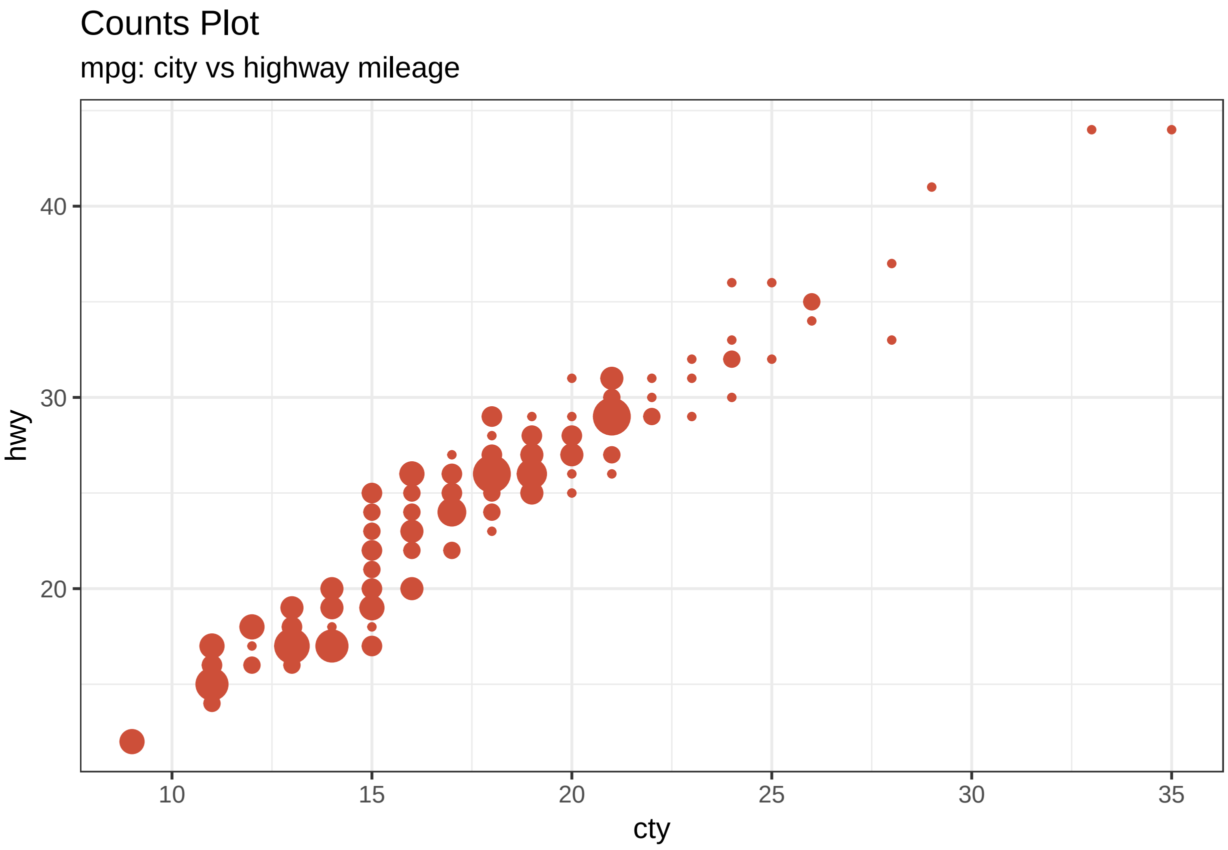
Jitter Plot

```
# Library
library(ggplot2)

# Data
data(mpg, package="ggplot2")

# Theme
theme_set(theme_bw()) # pre-set the bw theme.

# plot
mpg %>%
  ggplot(aes(cty,hwy))+
  geom_jitter(width = .5, size=1) +
  geom_smooth(method="lm", se=F) +
  labs(title="Scatterplot with overlapping points",
       subtitle="mpg: city vs highway mileage",
       x="cty",
       y="hwy",
       caption="Source: midwest")
```



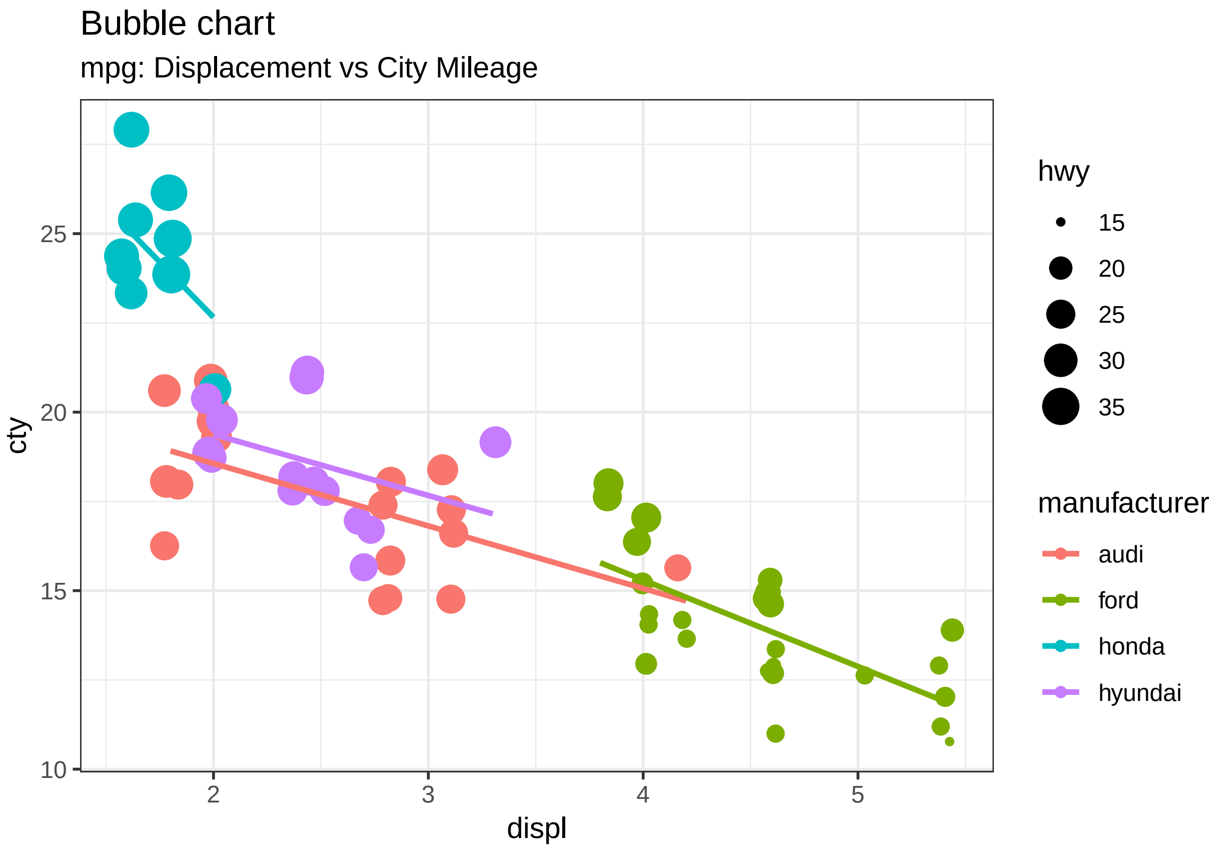
Count Chart

```
# Library
library(ggplot2)

# Data
data(mpg, package="ggplot2")

# Theme
theme_set(theme_bw())

# Scatterplot
mpg %>%
  ggplot(aes(cty, hwy)) +
  geom_count(col="tomato3", show.legend=F) +
  labs(title="Counts Plot",
       subtitle="mpg: city vs highway mileage",
       x="cty",
       y="hwy")
```



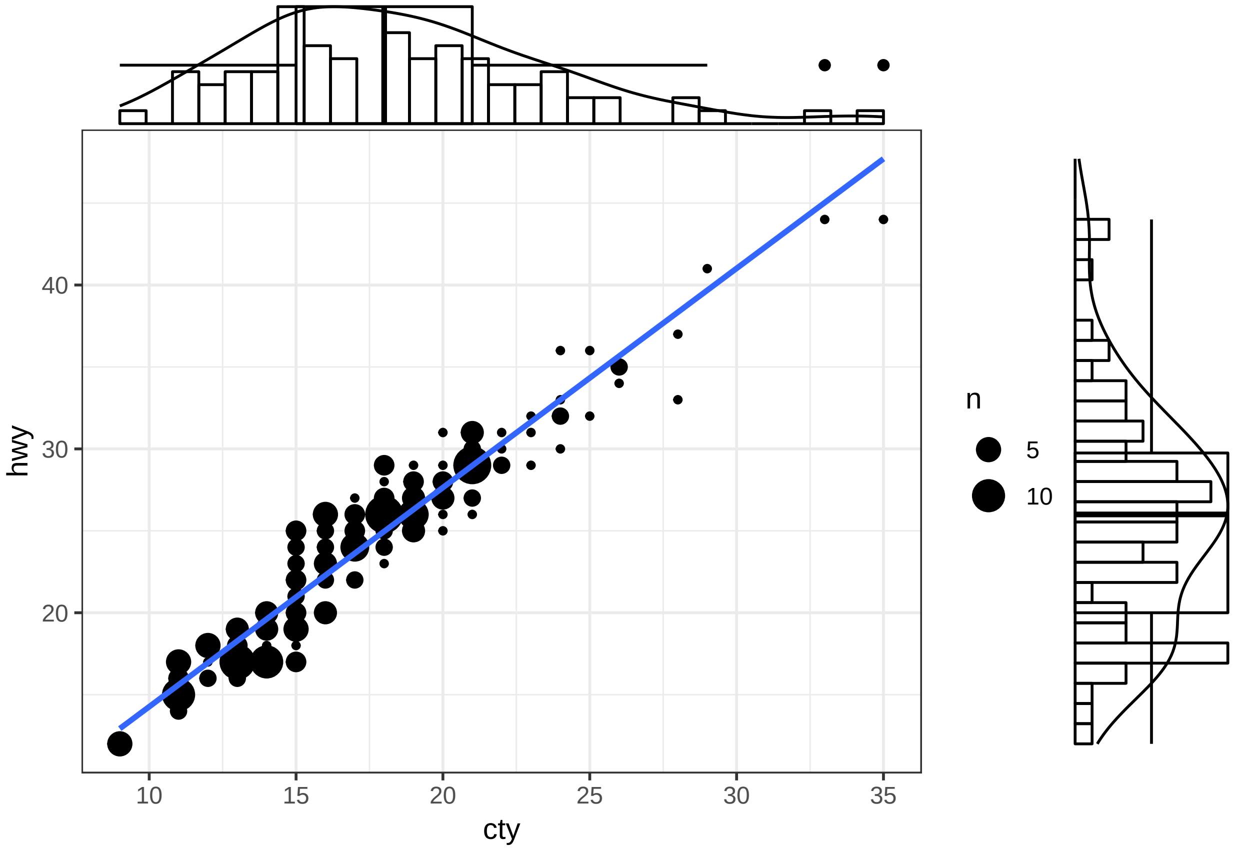
Bubble plot

```
# Library
library(ggplot2)

# Data
data(mpg, package="ggplot2")

mpg_select <- mpg %>%
                dplyr::filter(manufacturer %in% c("audi", "ford", "honda", "hyundai"))

# Plot
mpg_select %>%
  ggplot(aes(displ, cty)) +
  geom_jitter(aes(col=manufacturer, size=hwy)) +
  geom_smooth(aes(col=manufacturer), method="lm", se=F)+
  labs(title="Bubble chart",
       subtitle="mpg: Displacement vs City Mileage")
```

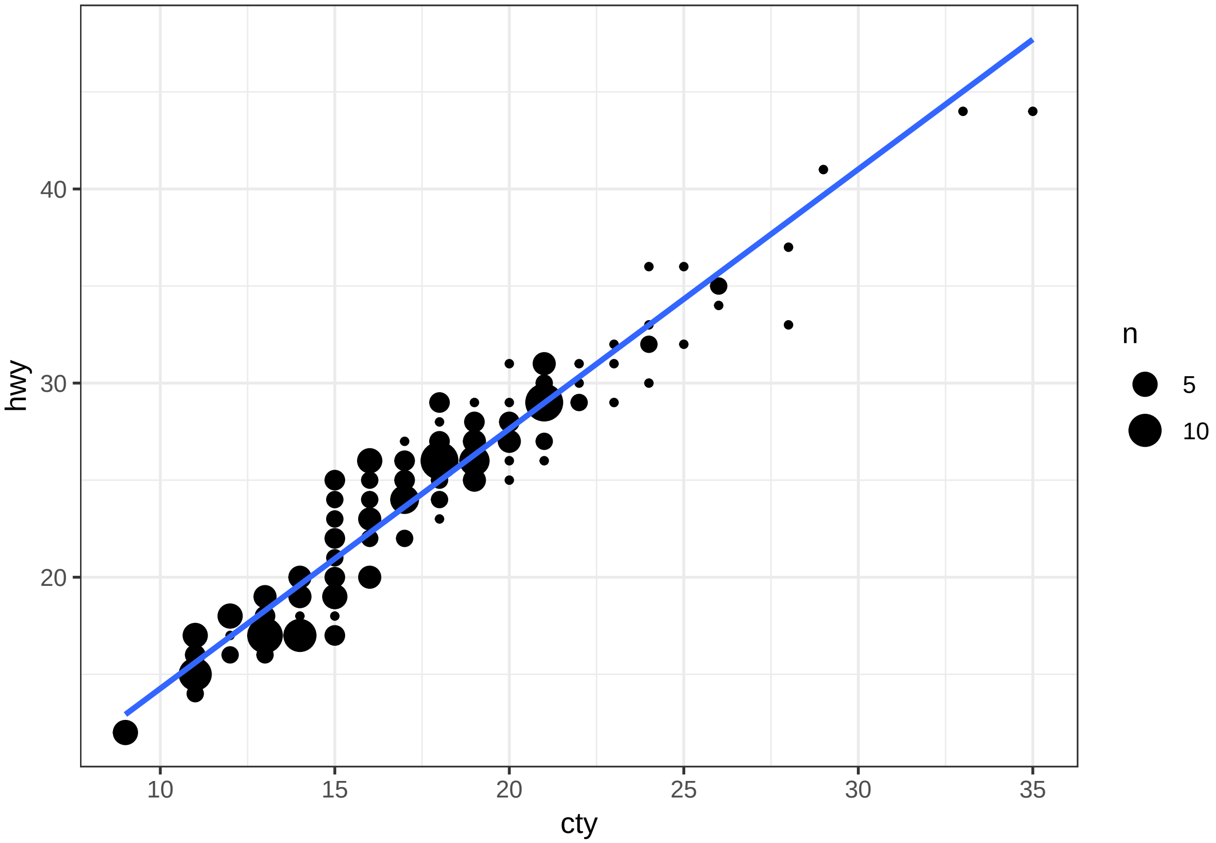


Marginal Histogram / Boxplot

```
# Library
library(ggplot2)
library(ggExtra)

# Data
data(mpg, package="ggplot2")

mpg_select_27_35 <- mpg %>% filter(hwy >= 35 & cty > 27)

# Plot
plot_2 <- mpg %>%
          ggplot(aes(cty, hwy)) +
          geom_count() +
          geom_smooth(method="lm", se=F)

ggMarginal(plot_2 , type = "histogram", fill="transparent")
ggMarginal(plot_2 , type = "boxplot", fill="transparent")
ggMarginal(plot_2 , type = "density", fill="transparent")
```

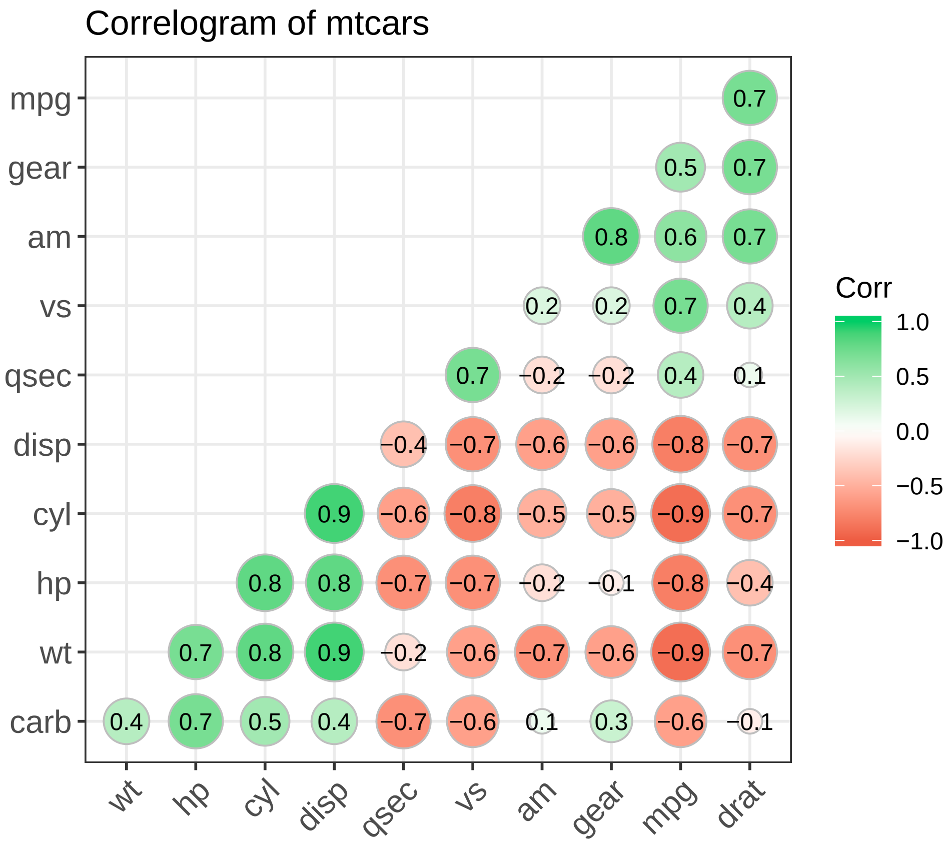


```
plot_2
```



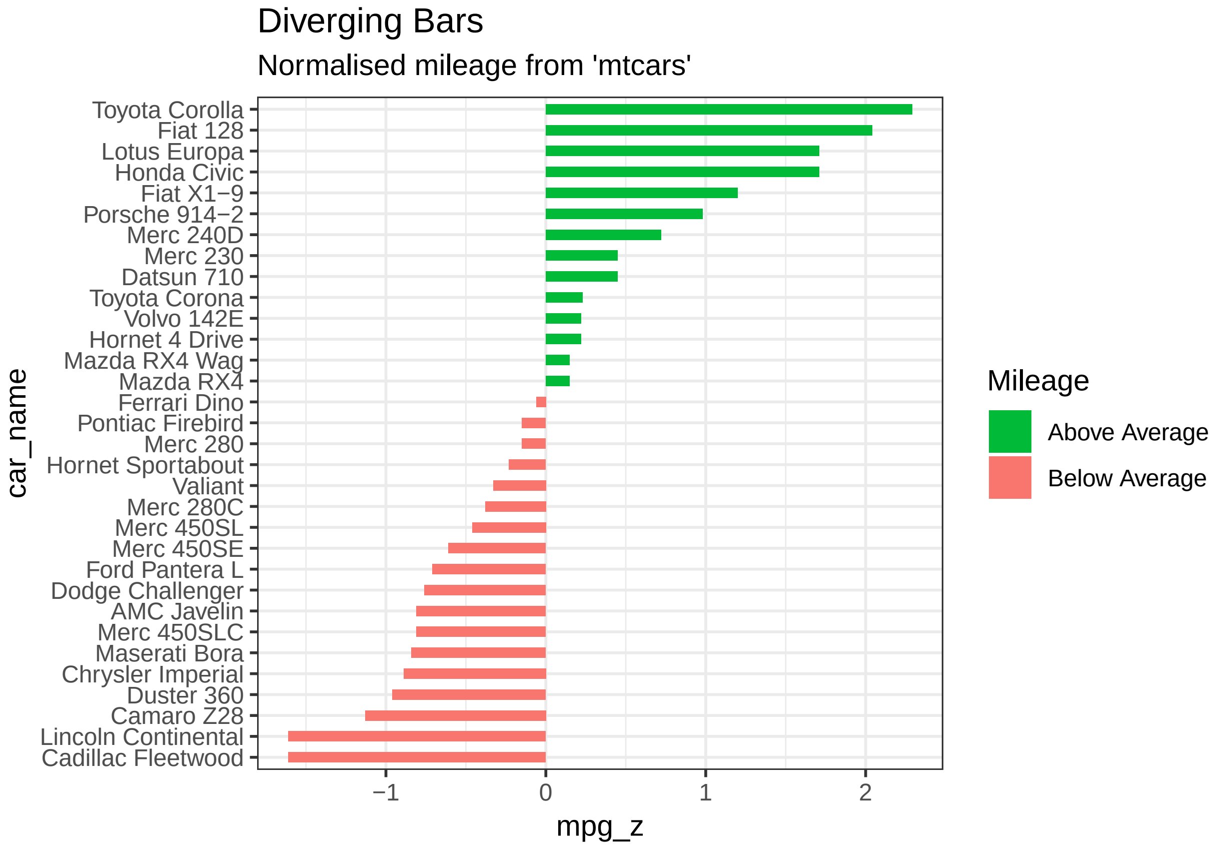
Correlogram

```
# Library
library(ggplot2)
library(ggcorrplot)

corr <- round(cor(mtcars), 1)

# Plot
ggcorrplot(corr, hc.order = TRUE,
                 type = "lower",
                 lab = TRUE,
                 lab_size = 3,
                 method="circle",
                 colors = c("tomato2", "white", "springgreen3"),
                 title="Correlogram of mtcars", ggtheme=theme_bw)
```

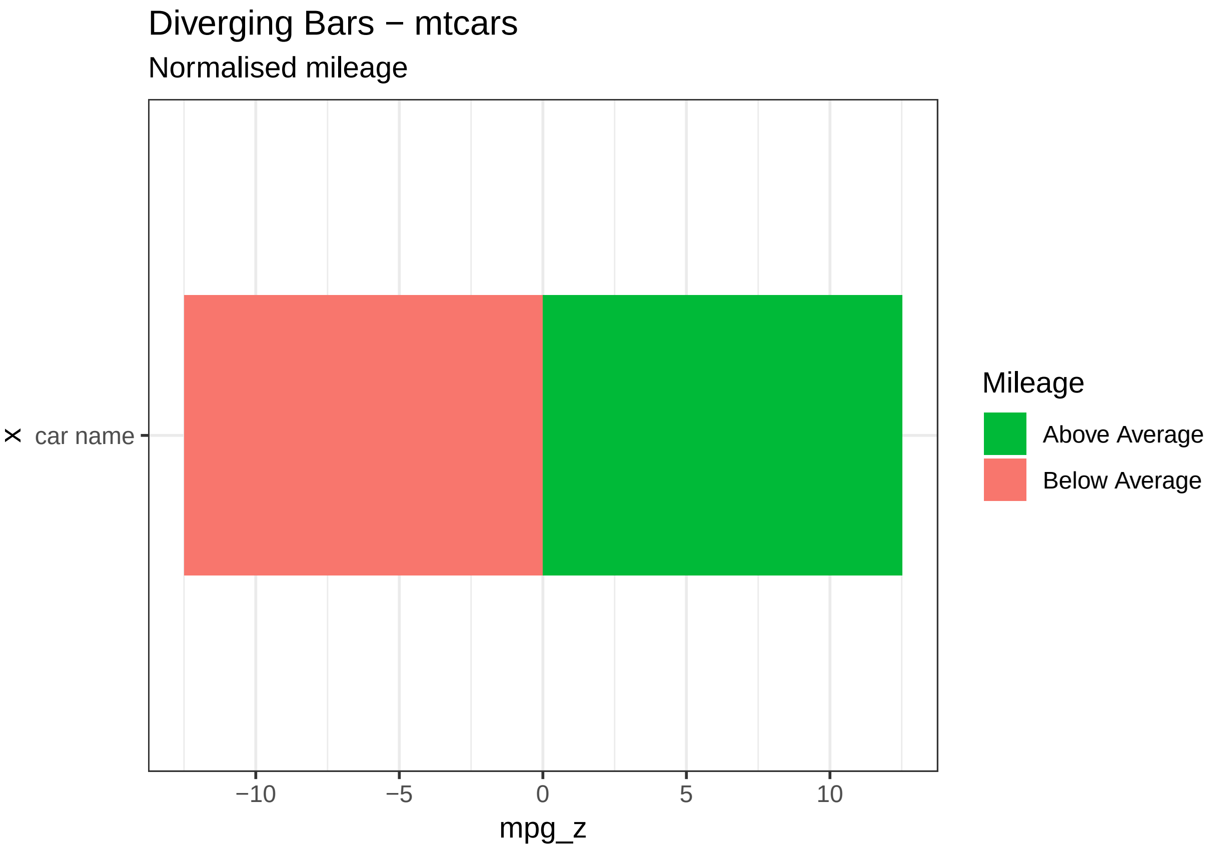


B. Deviation/Diverging Bars

```
# A:

# Library
library(ggplot2)

# Data
data("mtcars")

mtcars$car_name <- rownames(mtcars) # create new column for car names
mtcars$mpg_z <- round((mtcars$mpg - mean(mtcars$mpg))/sd(mtcars$mpg), 2) # compute normalized mpg
mtcars$mpg_type <- ifelse(mtcars$mpg_z < 0, "below", "above") # above / below avg flag

mtcars <- mtcars %>% arrange(mpg_z)
mtcars$car_name <- factor(mtcars$car_name, levels = mtcars$car_name) # convert to factor to retain sort

# Diverging Barcharts

mtcars %>%
  ggplot(aes(x=car_name, y=mpg_z, label=mpg_z)) +
  geom_bar(stat='identity', aes(fill=mpg_type), width=.5) +
  scale_fill_manual(name="Mileage",
                    labels = c("Above Average", "Below Average"),
                    values = c("above"="#00ba38", "below"="#f8766d")) +
  labs(title= "Diverging Bars",
       subtitle="Normalised mileage from 'mtcars'") +
  coord_flip()
```



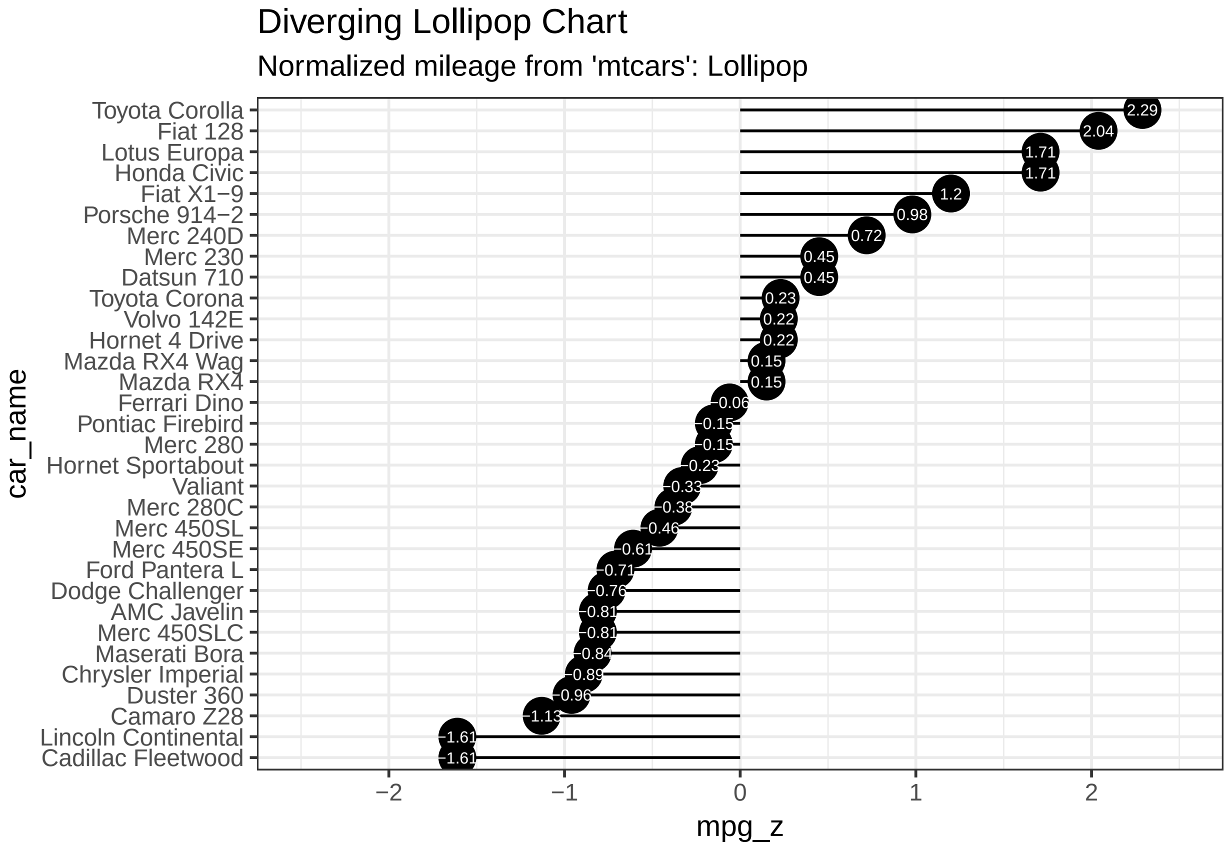
```
# B :

library(ggplot2)
ggplot(mtcars, aes(x='car name', y=mpg_z, label=mpg_z)) +
  geom_bar(stat='identity', aes(fill=mpg_type), width=.5) +
  scale_fill_manual(name="Mileage",
                    labels = c("Above Average", "Below Average"),
                    values = c("above"="#00ba38", "below"="#f8766d")) +
  labs(subtitle="Normalised mileage",
       title= "Diverging Bars - mtcars") +
  coord_flip()
```



Diverging Lollipop Chart

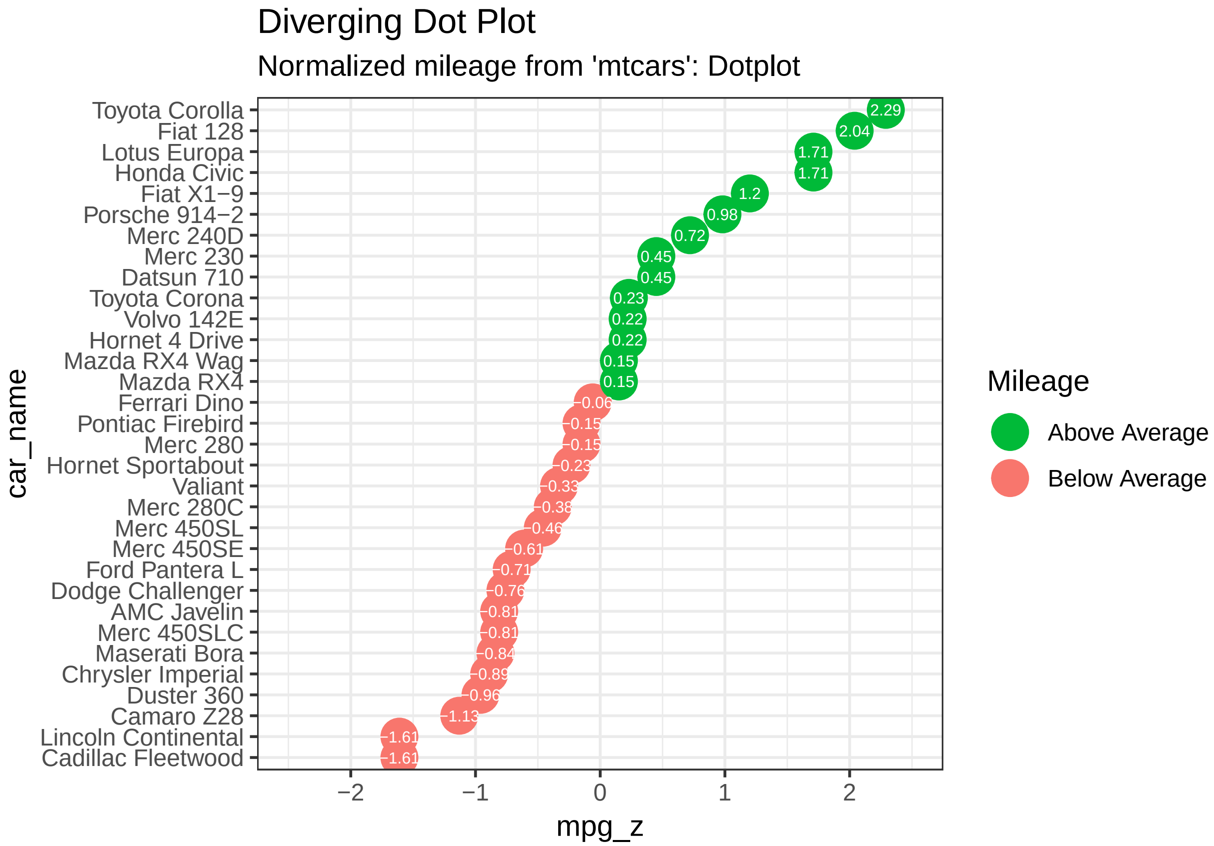
```
# Library
library(ggplot2)

# Data
data("mtcars")

mtcars$car_name <- rownames(mtcars) # create new column for car names
mtcars$mpg_z <- round((mtcars$mpg - mean(mtcars$mpg))/sd(mtcars$mpg), 2) # compute normalized mpg
mtcars$mpg_type <- ifelse(mtcars$mpg_z < 0, "below", "above") # above / below avg flag

mtcars <- mtcars %>% arrange(mpg_z)
mtcars$car_name <- factor(mtcars$car_name, levels = mtcars$car_name) # convert to factor to retain sort

# Plot:
mtcars %>%
ggplot(aes(x=car_name, y=mpg_z, label=mpg_z)) +
  geom_point(stat='identity', fill="black", size=6) +
  geom_segment(aes(y = 0,
                   x = car_name,
                   yend = mpg_z,
                   xend = car_name),
                    color = "black") +
```

```r
  geom_text(color="white", size=2) +
  labs(title="Diverging Lollipop Chart",
       subtitle="Normalized mileage from 'mtcars': Lollipop") +
  ylim(-2.5, 2.5) +
  coord_flip()
```



Diverging Dot Plot

```r
# Library
library(ggplot2)

# Data
data("mtcars")

mtcars$car_name <- rownames(mtcars) # create new column for car names
mtcars$mpg_z <- round((mtcars$mpg - mean(mtcars$mpg))/sd(mtcars$mpg), 2) # compute normalized mpg
mtcars$mpg_type <- ifelse(mtcars$mpg_z < 0, "below", "above") # above / below avg flag

mtcars <- mtcars %>% arrange(mpg_z)
mtcars$car_name <- factor(mtcars$car_name, levels = mtcars$car_name) # convert to factor to retain sort


# Plot
mtcars %>%
  ggplot(aes(x=car_name, y=mpg_z, label=mpg_z)) +
  geom_point(stat='identity', aes(col=mpg_type), size=6) +
  scale_color_manual(name="Mileage",
                     labels = c("Above Average", "Below Average"),
                     values = c("above"="#00ba38", "below"="#f8766d")) +
  geom_text(color="white", size=2) +
  labs(title="Diverging Dot Plot",
       subtitle="Normalized mileage from 'mtcars': Dotplot") +
  ylim(-2.5, 2.5) +
  coord_flip()
```

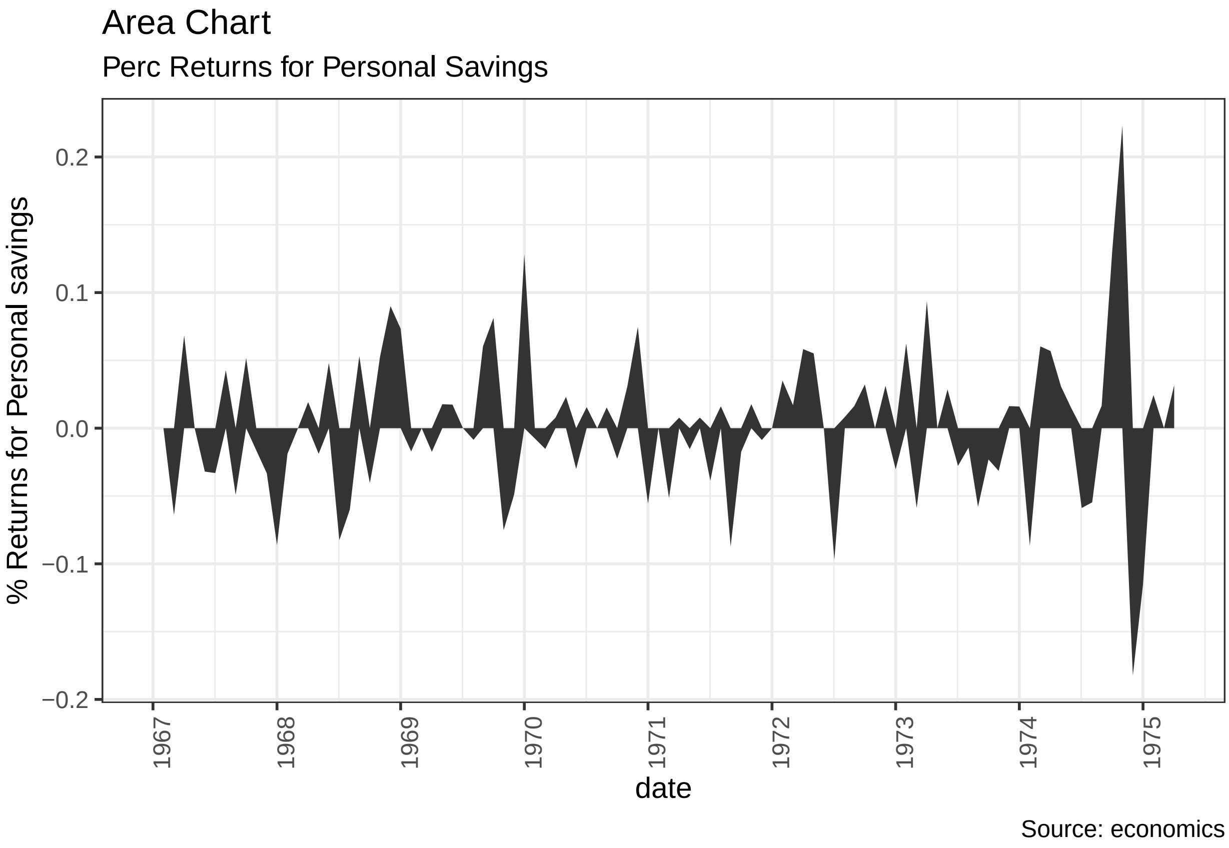


Area Chart

```
data("economics")
economics %>% head()
```

```
## # A tibble: 6 x 6
##   date          pce    pop psavert uempmed unemploy
##   <date>      <dbl>  <int>   <dbl>   <dbl>    <int>
## 1 1967-07-01  507. 198712     12.5     4.5     2944
## 2 1967-08-01  510. 198911     12.5     4.7     2945
## 3 1967-09-01  516. 199113     11.7     4.6     2958
## 4 1967-10-01  513. 199311     12.5     4.9     3143
## 5 1967-11-01  518. 199498     12.5     4.7     3066
## 6 1967-12-01  526. 199657     12.1     4.8     3018
```

```
# Library
library(ggplot2)
library(quantmod)

# Data
data("economics", package = "ggplot2")

# Compute % Returns
economics$returns_perc <- c(0, diff(economics$psavert)/economics$psavert[-length(economics$psavert)]) #

# Create break points and labels for axis ticks
brks <- economics$date[seq(1, length(economics$date), 12)]
lbls <- lubridate::year(economics$date[seq(1, length(economics$date), 12)])

# Plot
economics[1:100, ] %>%
ggplot(aes(date, returns_perc)) +
geom_area() +
scale_x_date(breaks=brks, labels=lbls) +
  theme(axis.text.x = element_text(angle=90)) +
  labs(title="Area Chart",
       subtitle = "Perc Returns for Personal Savings",
       y="% Returns for Personal savings",
       caption="Source: economics")
```

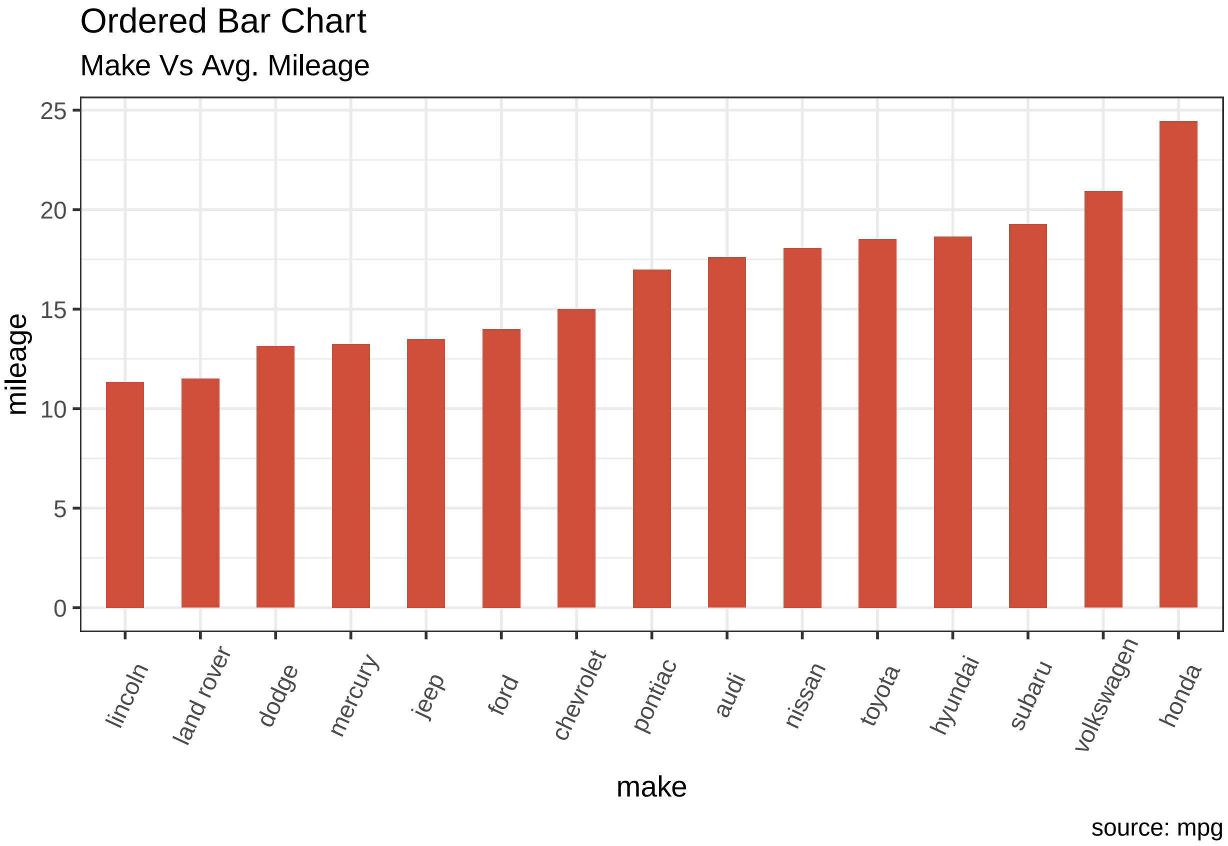

Area Chart

Perc Returns for Personal Savings

Source: economics

Ranking: Ordered Bar Chart

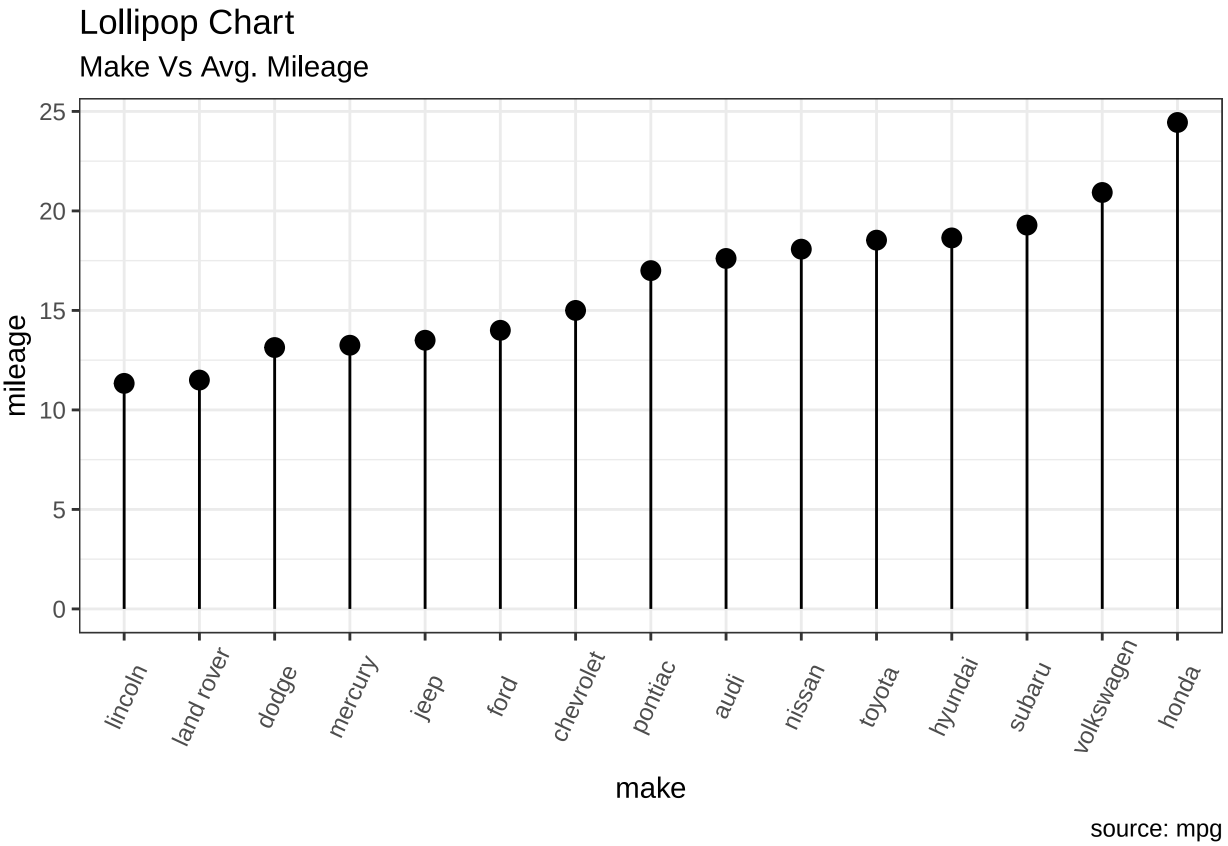
```
library(ggplot2)

# Prepare data: group mean city mileage by manufacturer.
cty_mpg <- aggregate(mpg$cty, by=list(mpg$manufacturer), FUN=mean) # aggregate
colnames(cty_mpg) <- c("make", "mileage") # change column names
cty_mpg <- cty_mpg %>% arrange(mileage) # sort
cty_mpg$make <- factor(cty_mpg$make, levels = cty_mpg$make) # to retain the order in plot.

# plot
cty_mpg %>%
  ggplot(aes(x=make, y=mileage)) +
  geom_bar(stat="identity",width=.5, fill="tomato3") +
  labs(title="Ordered Bar Chart",
       subtitle="Make Vs Avg. Mileage",
       caption="source: mpg") +
theme(axis.text.x = element_text(angle=65, vjust=0.6))
```



Lollipop Chart

```
library(ggplot2)

# Prepare data: group mean city mileage by manufacturer.
```

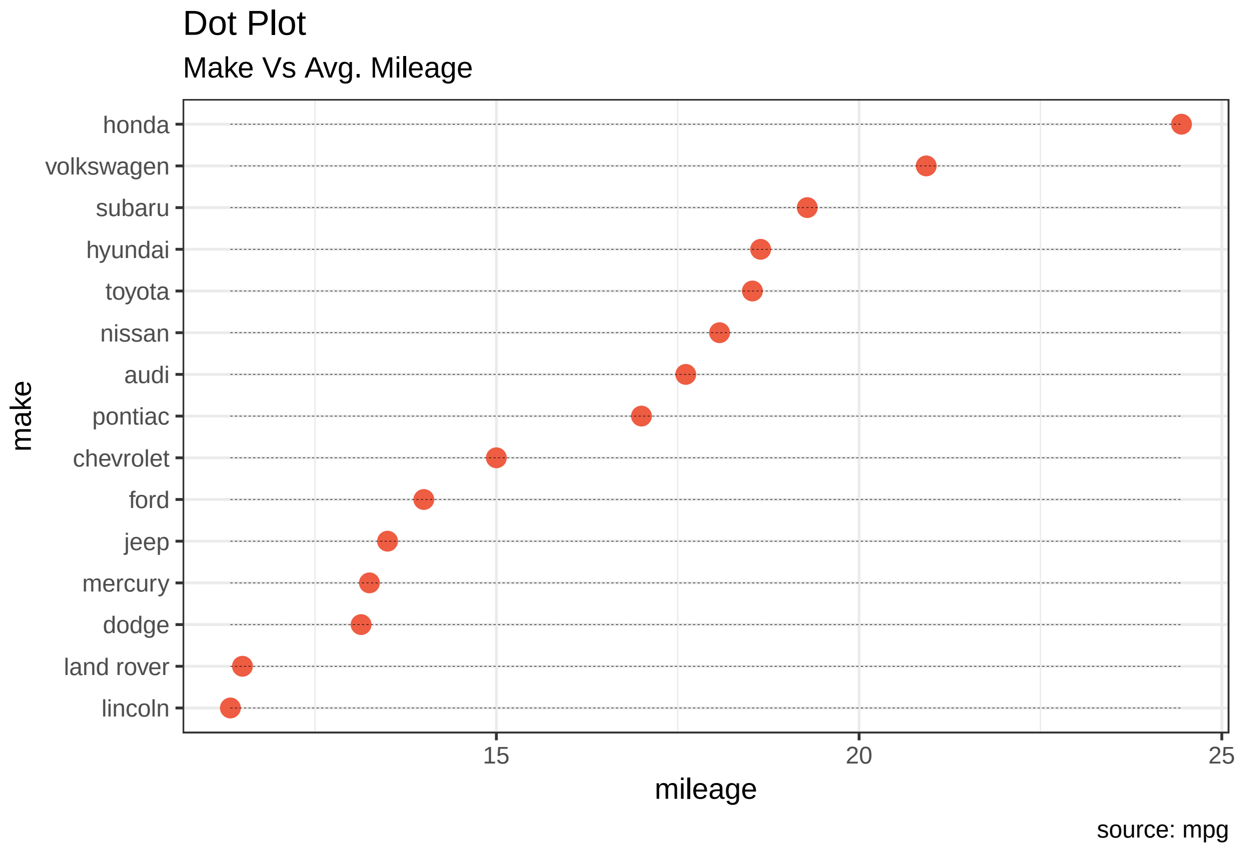
```r
cty_mpg <- aggregate(mpg$cty, by=list(mpg$manufacturer), FUN=mean) # aggregate
colnames(cty_mpg) <- c("make", "mileage") # change column names
cty_mpg <- cty_mpg %>% arrange(mileage) # sort
cty_mpg$make <- factor(cty_mpg$make, levels = cty_mpg$make) # to retain the order in plot.

# Plot
cty_mpg %>%
  ggplot(aes(x=make, y=mileage)) +
  geom_point(size=3) +
  geom_segment(aes(x=make,
                   xend=make,
                   y=0,
                   yend=mileage)) +
  labs(title="Lollipop Chart",
       subtitle="Make Vs Avg. Mileage",
       caption="source: mpg") +
  theme(axis.text.x = element_text(angle=65, vjust=0.6))
```



Dot Plot

```r
# Library
library(ggplot2)
library(scales)
```

```
# Prepare data: group mean city mileage by manufacturer.
cty_mpg <- aggregate(mpg$cty, by=list(mpg$manufacturer), FUN=mean) # aggregate
colnames(cty_mpg) <- c("make", "mileage") # change column names
cty_mpg <- cty_mpg %>% arrange(mileage) # sort
cty_mpg$make <- factor(cty_mpg$make, levels = cty_mpg$make) # to retain the order in plot.

# Plot
cty_mpg %>%
ggplot(aes(x=make, y=mileage)) +
geom_point(col="tomato2", size=3) + # Draw points
geom_segment(aes(x=make, xend=make, y=min(mileage), yend=max(mileage)), linetype="dashed",  size=0.1) +
  labs(title="Dot Plot",
       subtitle="Make Vs Avg. Mileage",
       caption="source: mpg") +
  coord_flip()
```


Dot Plot

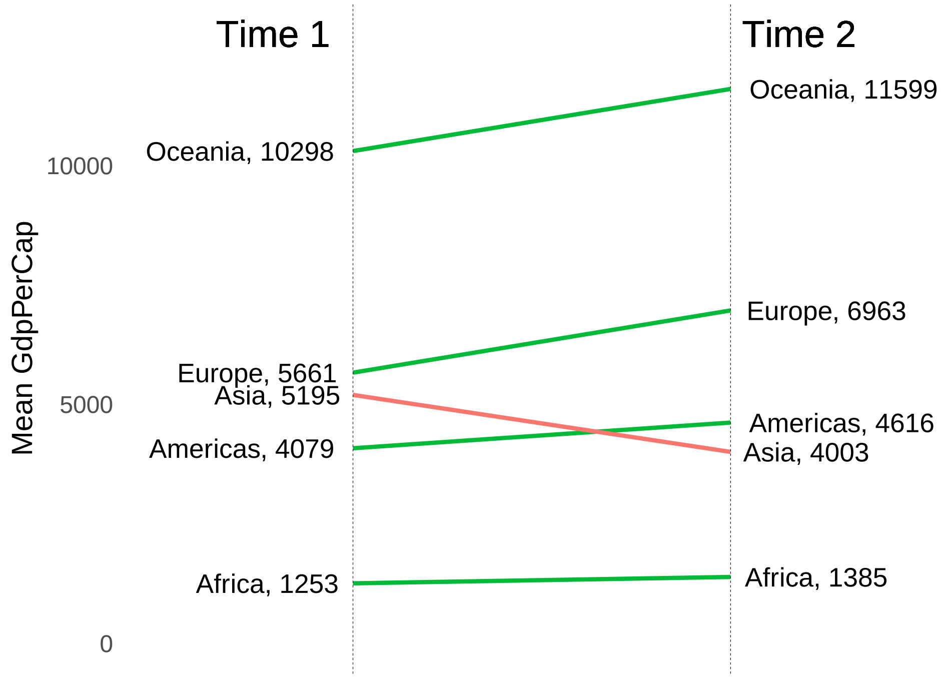
Make Vs Avg. Mileage

source: mpg

Slope Chart:

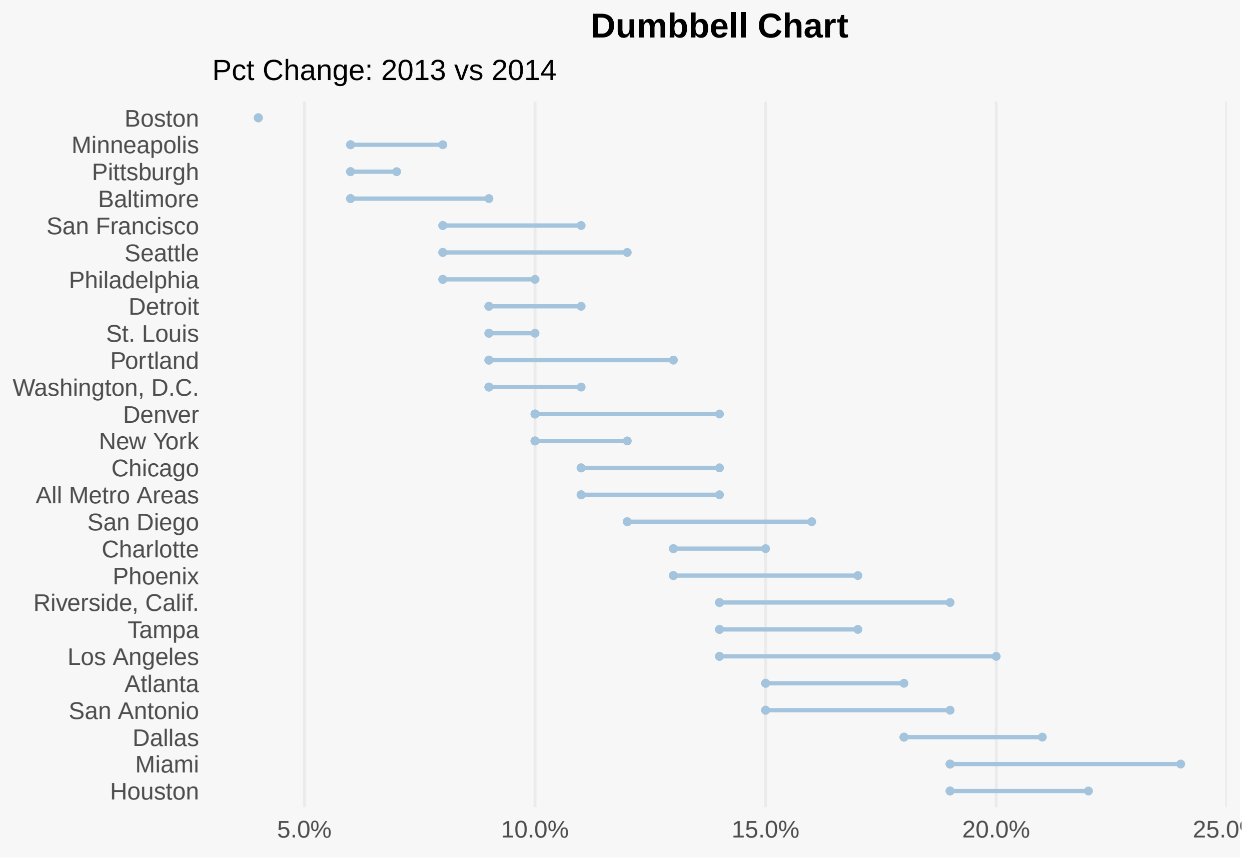
```
library(ggplot2)
library(scales)

# prep data
df <- read.csv("https://raw.githubusercontent.com/shahnp/data/master/gdppercap.txt")
colnames(df) <- c("continent", "1952", "1957")
```

```
left_label <- paste(df$continent, round(df$`1952`),sep=", ")
right_label <- paste(df$continent, round(df$`1957`),sep=", ")

df$class <- ifelse((df$`1957` - df$`1952`) < 0, "red", "green")

# Plot
p <- ggplot(df) +
  geom_segment(aes(x=1, xend=2, y=`1952`, yend=`1957`, col=class), size=.75,show.legend=F) +
  geom_vline(xintercept=1, linetype="dashed", size=.1) +
  geom_vline(xintercept=2, linetype="dashed", size=.1) +
  scale_color_manual(labels = c("Up", "Down"),
                      values = c("green"="#00ba38", "red"="#f8766d")) +
  labs(x="", y="Mean GdpPerCap") + # Axis labels
  xlim(.5, 2.5) +
  ylim(0,(1.1*(max(df$`1952`, df$`1957`)))) # X and Y axis label

# Add texts
p <- p + geom_text(label=left_label, y=df$`1952`, x=rep(1, NROW(df)), hjust=1.1, size=3.5)
p <- p + geom_text(label=right_label, y=df$`1957`, x=rep(2, NROW(df)), hjust=-0.1, size=3.5)
p <- p + geom_text(label="Time 1", x=1, y=1.1*(max(df$`1952`, df$`1957`)), hjust=1.2, size =5) # title
p <- p + geom_text(label="Time 2", x=2, y=1.1*(max(df$`1952`, df$`1957`)), hjust=-0.1, size=5) # title

# Minify theme
p + theme(panel.background = element_blank(),
          panel.grid = element_blank(),
          axis.ticks = element_blank(),
          axis.text.x = element_blank(),
          panel.border = element_blank(),
          plot.margin = unit(c(1,2,1,2), "cm"))
```



Dumbbell Plot

```
# library
library(ggalt)

health <- read.csv("https://raw.githubusercontent.com/shahnp/data/master/health.txt")
health$Area <-factor(health$Area, levels=as.character(health$Area)) # for right ordering of the dumbell
health$Area <- factor(health$Area)

gg <- health %>%
      ggplot(aes(x=pct_2013, xend=pct_2014, y=Area, group=Area)) +
      geom_dumbbell(color="#a3c4dc", size=0.75) +
      scale_x_continuous(label=percent) +
      labs(x=NULL,
           y=NULL,
           title="Dumbbell Chart",
           subtitle="Pct Change: 2013 vs 2014") +
theme(plot.title = element_text(hjust=0.5, face="bold"),
      plot.background=element_rect(fill="#f7f7f7"),
      panel.background=element_rect(fill="#f7f7f7"),
      panel.grid.minor=element_blank(),
      panel.grid.major.y=element_blank(),
      panel.grid.major.x=element_line(),
      axis.ticks=element_blank(),
      panel.border=element_blank(),
      legend.position="top")

plot(gg)
```

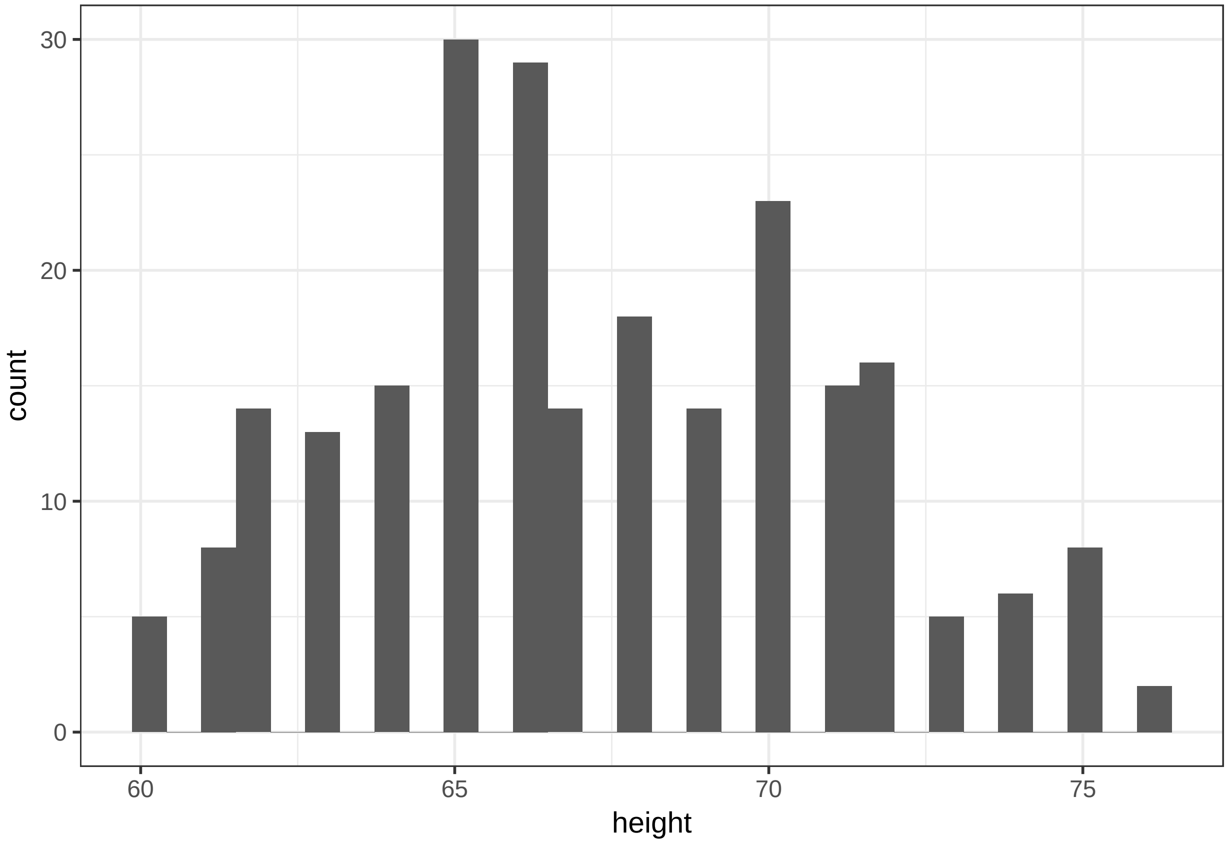


D. Distribution

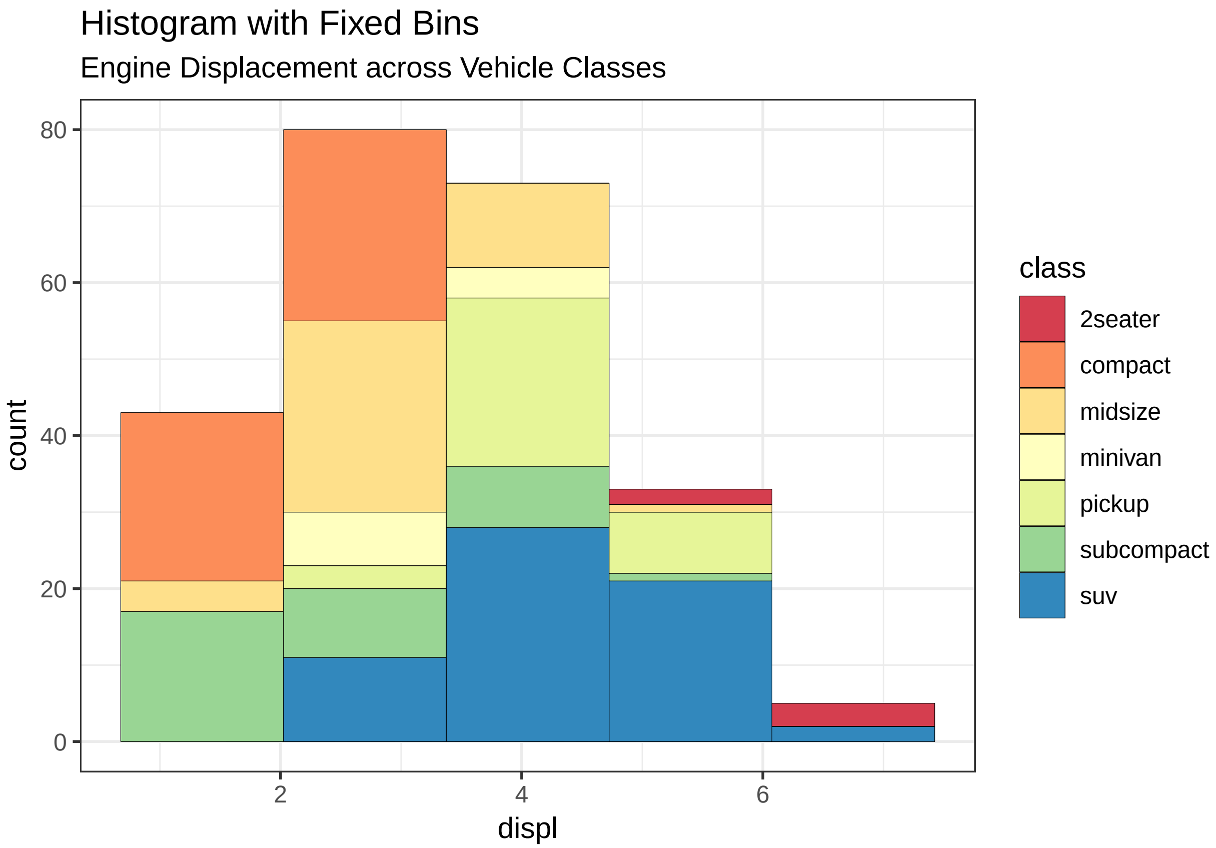
Histogram

```
# Library
library("ggplot2")

# Data
data(singer, package="lattice")

# A:
singer %>%
  ggplot(aes(x=height)) +
  geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

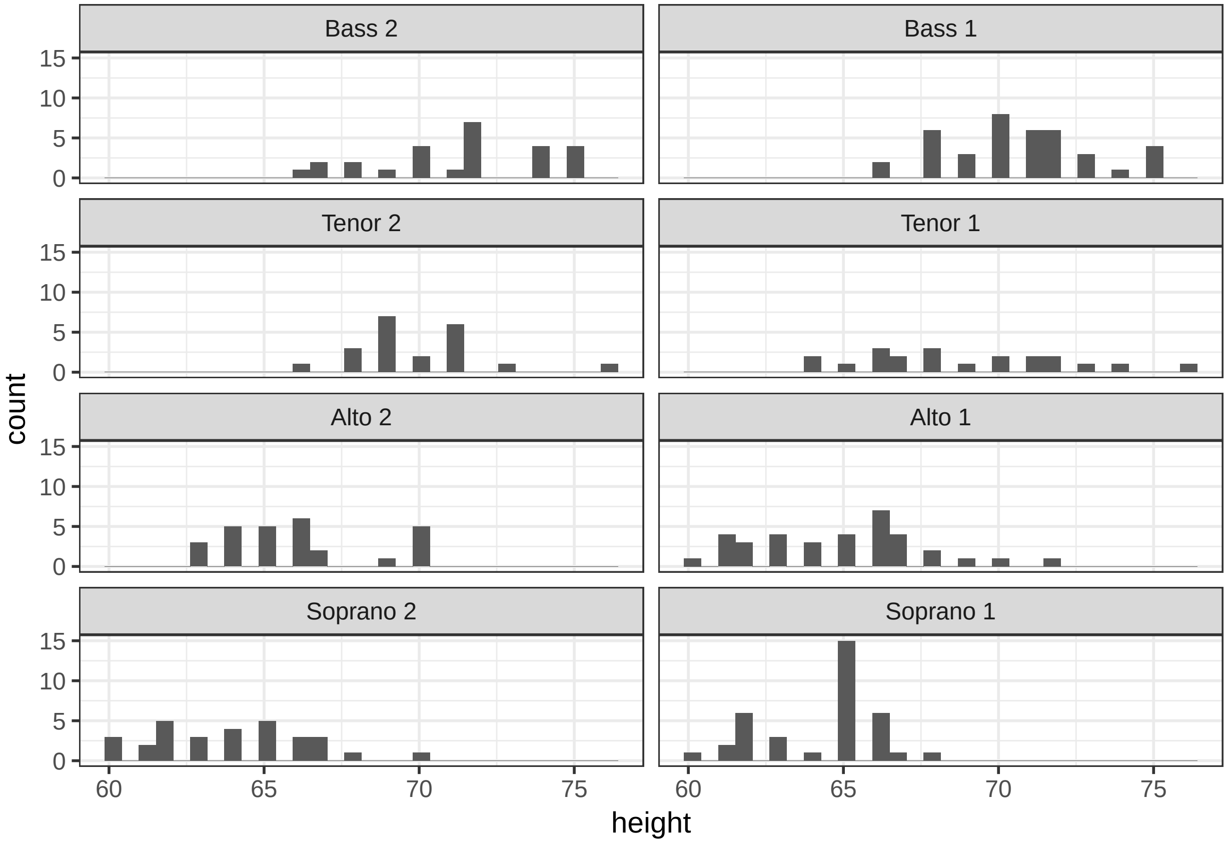


```
# B:
# Histogram on a Continuous (Numeric) Variable
mpg %>%
      ggplot( aes(displ)) +
      scale_fill_brewer(palette = "Spectral")+
      geom_histogram(aes(fill=class), binwidth = .1,col="black",size=.1) + # change binwidth
      labs(title="Histogram with Auto Binning",
           subtitle="Engine Displacement across Vehicle Classes")+
      geom_histogram(aes(fill=class), bins=5,col="black",size=.1) + # change number of bins
      labs(title="Histogram with Fixed Bins",
           subtitle="Engine Displacement across Vehicle Classes")
```

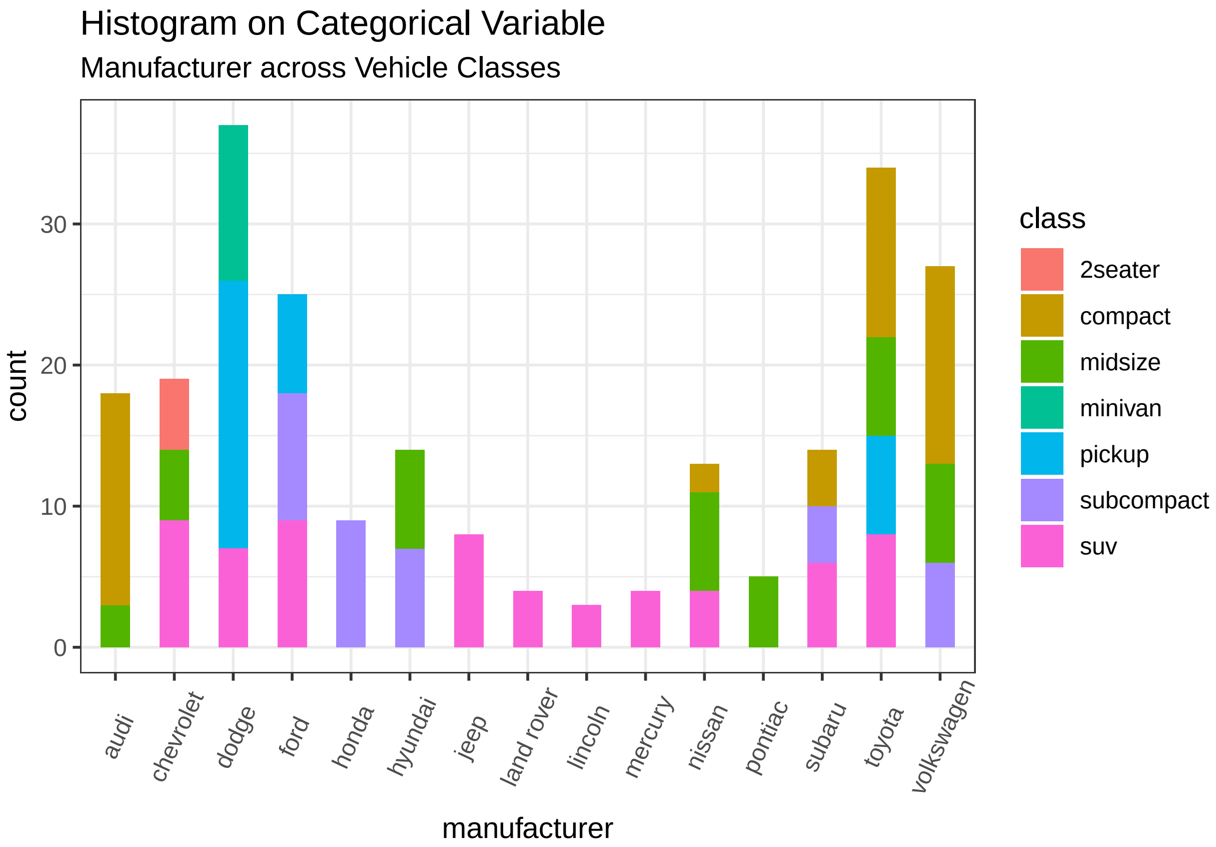


```r
# C:
library(ggplot2)
singer %>%
ggplot(aes(x=height)) +
     geom_histogram() +
     facet_wrap(~voice.part, nrow=4) # 4 plots each row, u can pass 8 and have all in one single column
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

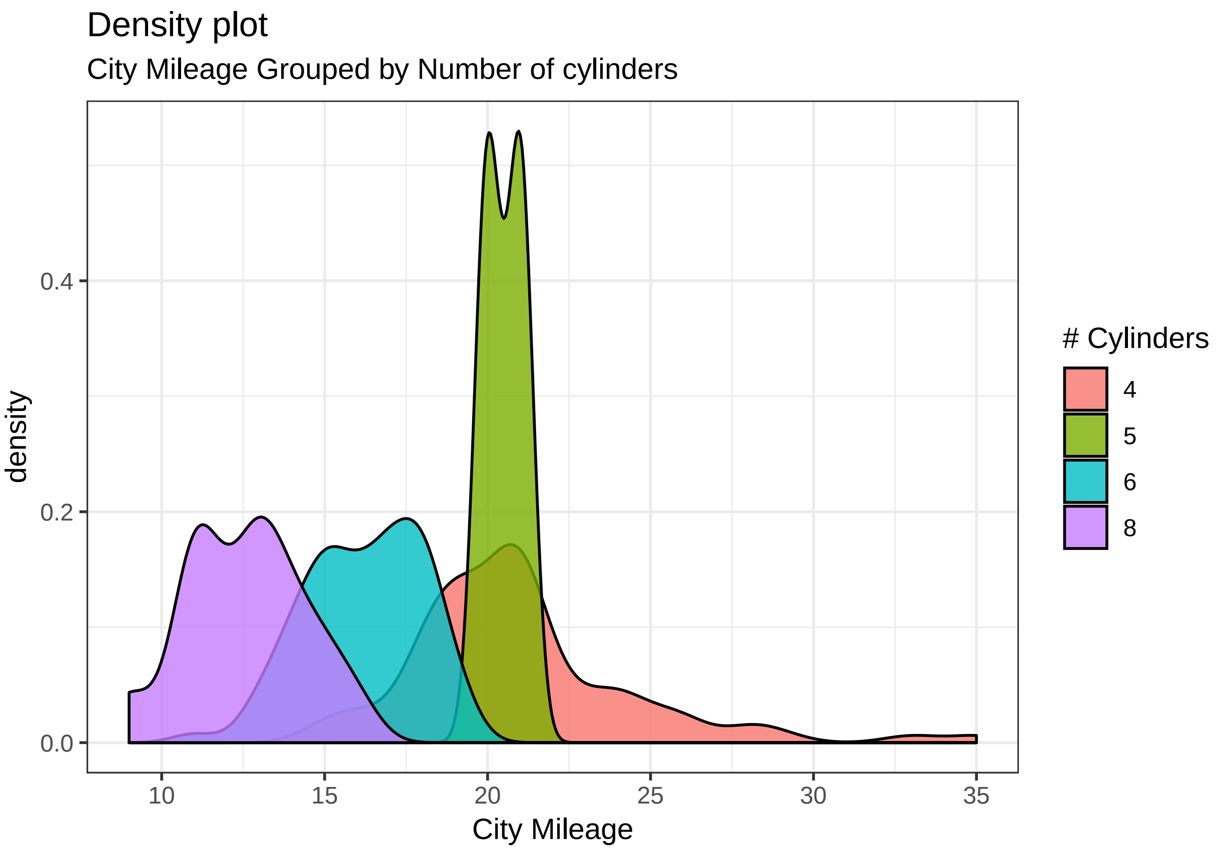


```
# D:

# Histogram on a Categorical variable
mpg %>%
ggplot(aes(manufacturer))+
geom_bar(aes(fill=class), width = 0.5) +
theme(axis.text.x = element_text(angle=65, vjust=0.6)) +
  labs(title="Histogram on Categorical Variable",
       subtitle="Manufacturer across Vehicle Classes")
```



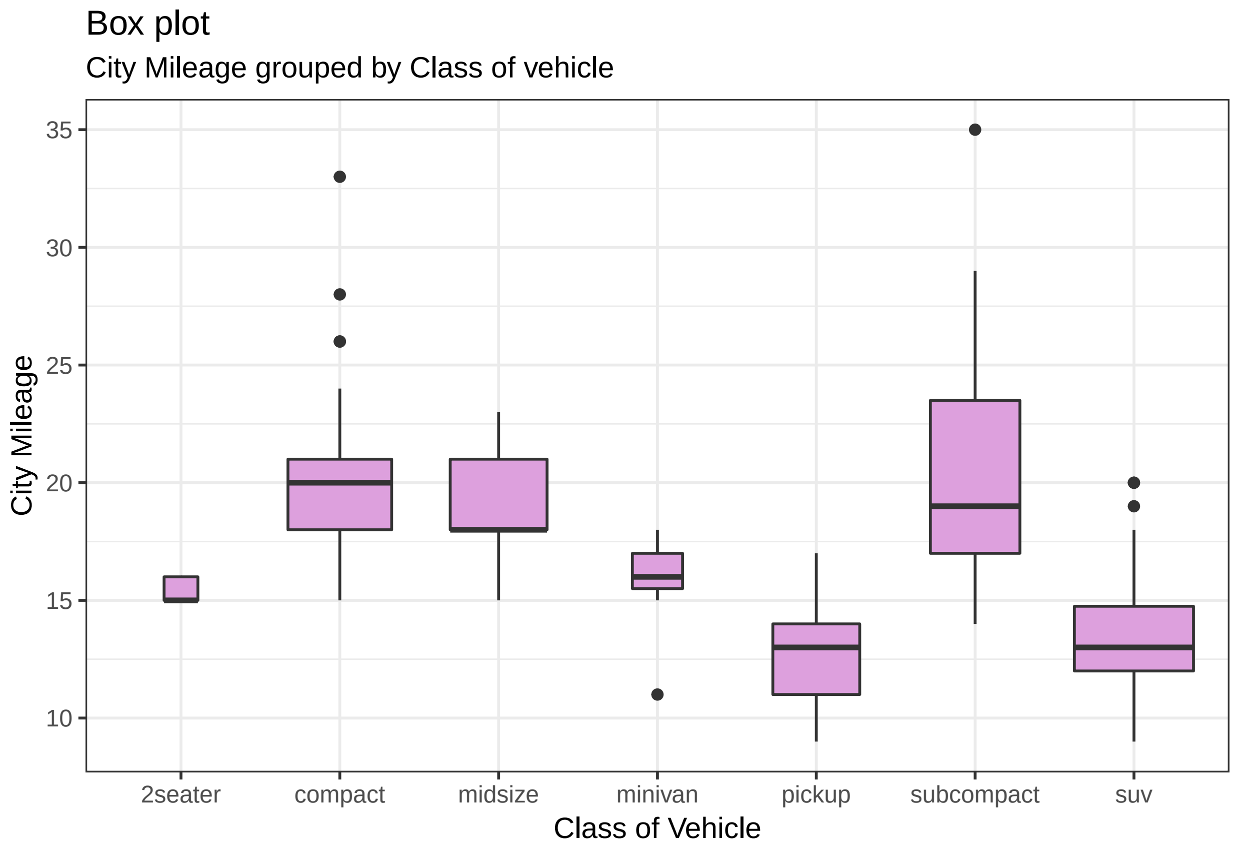
Density plot

```
mpg %>%
ggplot(aes(cty))+
geom_density(aes(fill=factor(cyl)), alpha=0.8) +
labs(title="Density plot",
     subtitle="City Mileage Grouped by Number of cylinders",
      x="City Mileage",
      fill="# Cylinders")
```

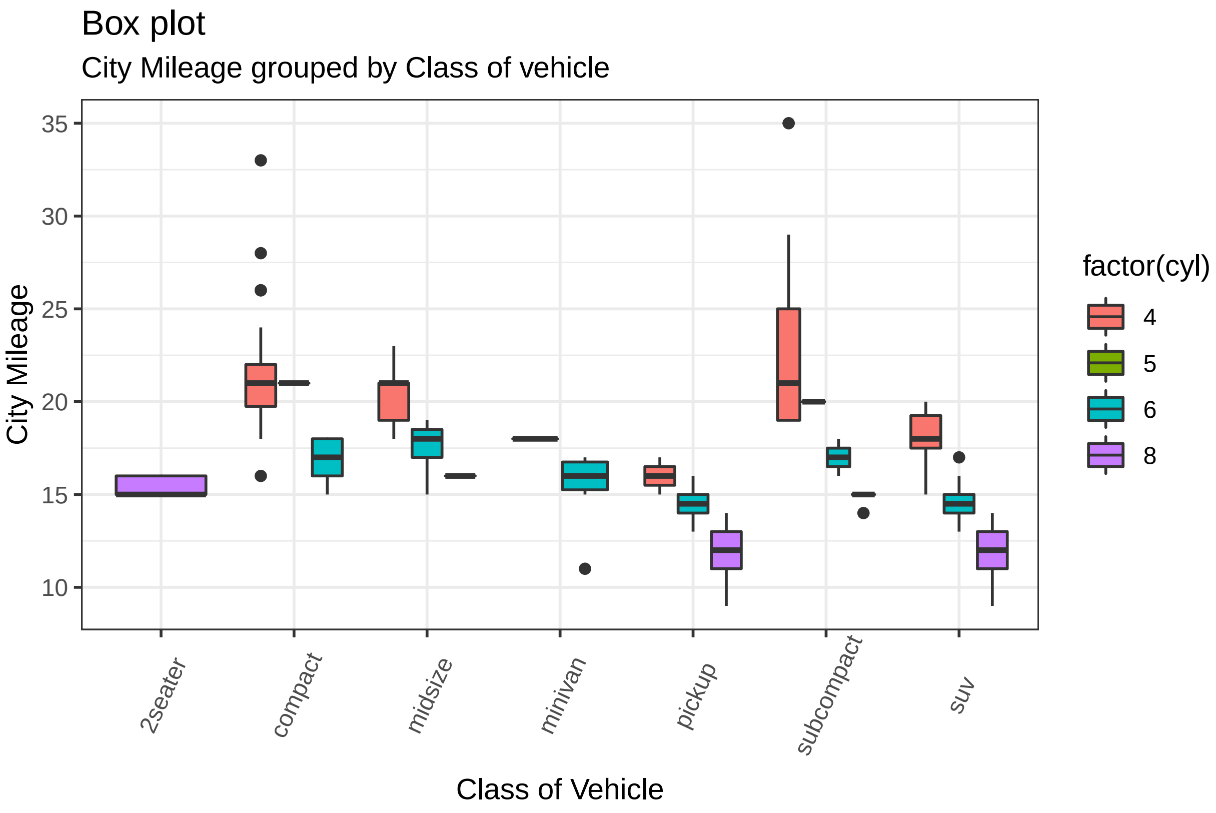


Box Plot

```
# library
library(ggplot2)
library(ggthemes)

# Plot

# A :
mpg %>%
  ggplot(aes(class,cty)) +
  geom_boxplot(varwidth=T, fill="plum") +
  labs(title="Box plot",
       subtitle="City Mileage grouped by Class of vehicle",
        x="Class of Vehicle",
        y="City Mileage")
```



```
# B :
mpg %>%
ggplot(aes(class, cty)) +
geom_boxplot(aes(fill=factor(cyl))) +
theme(axis.text.x = element_text(angle=65, vjust=0.6)) +
  labs(title="Box plot",
       subtitle="City Mileage grouped by Class of vehicle",
       caption="Source: mpg",
       x="Class of Vehicle",
       y="City Mileage")
```

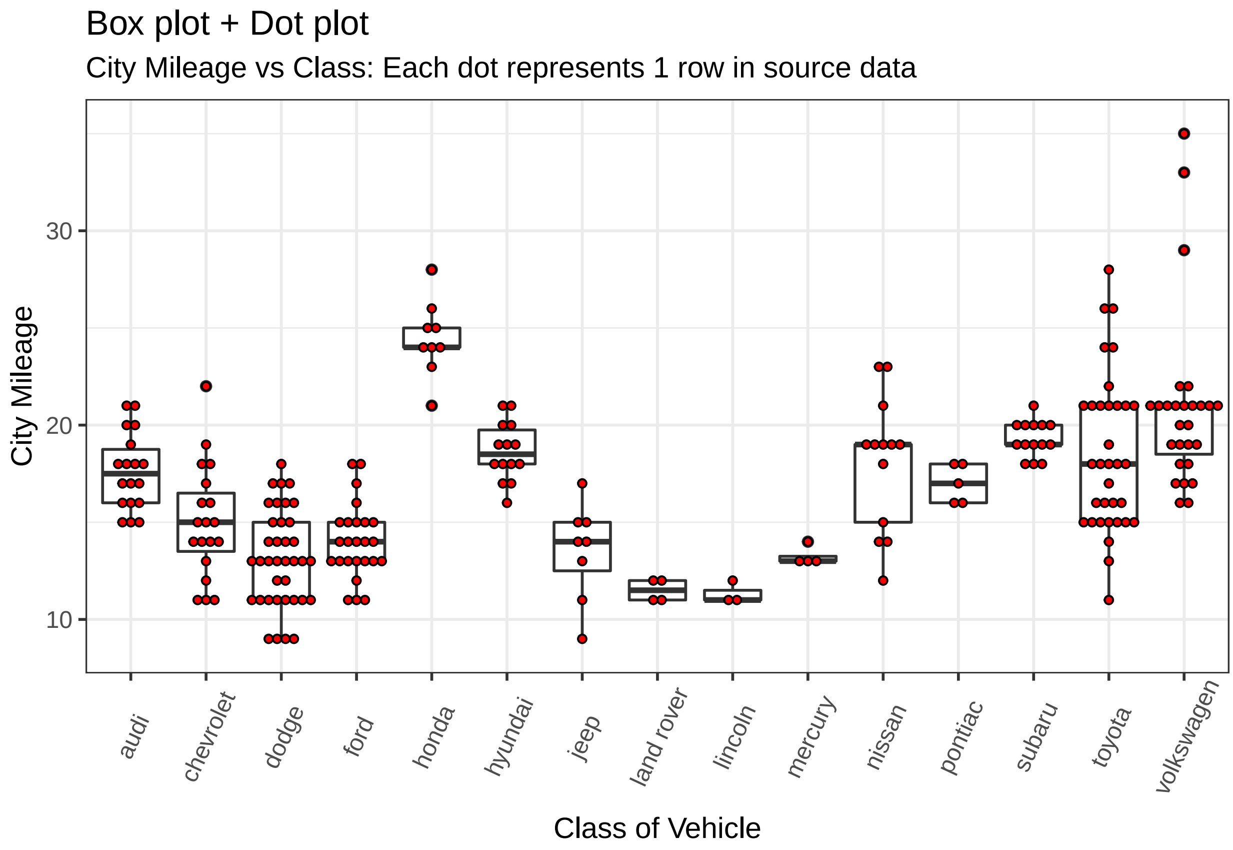


Source: mpg

Dot_Box Plot

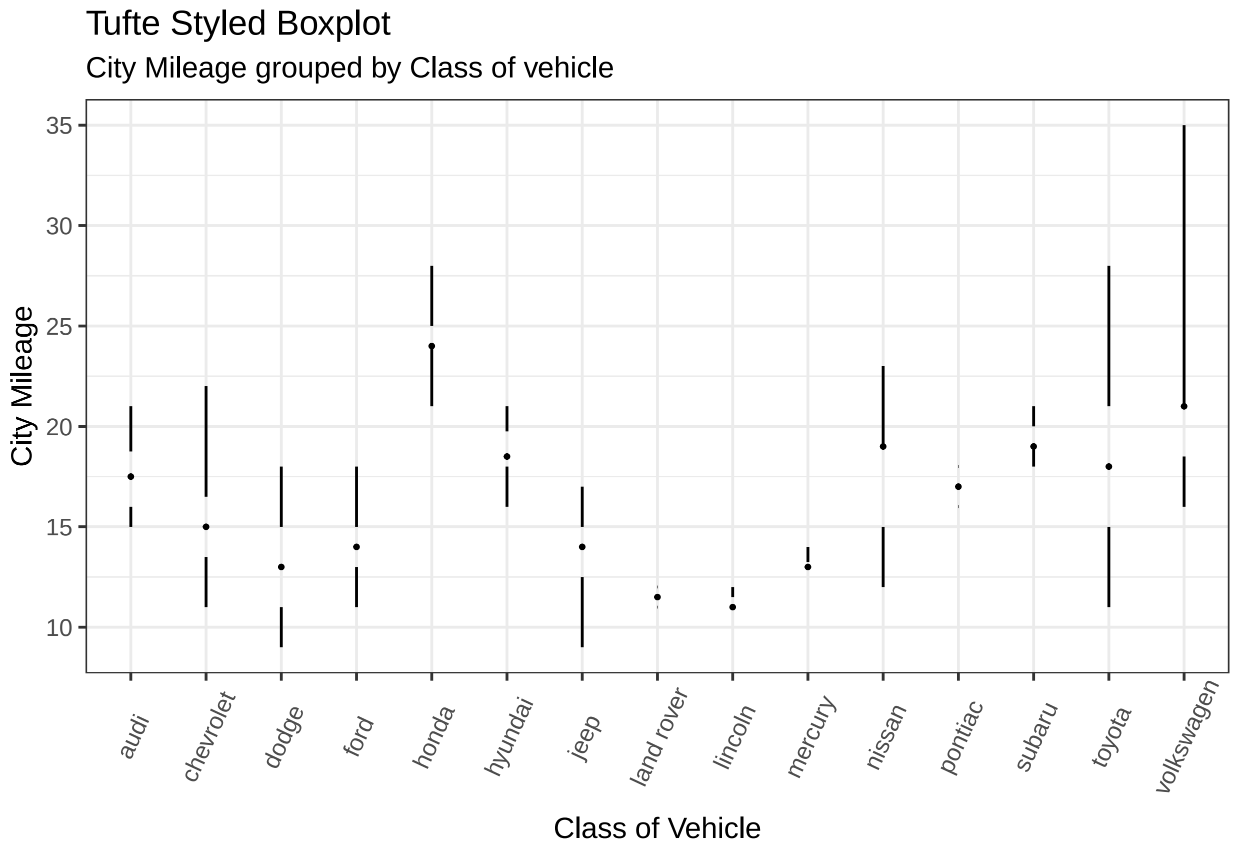
```
library(ggplot2)

# plot
mpg %>%
  ggplot(aes(manufacturer, cty)) +
  geom_boxplot(bins = 30) +
  geom_dotplot(binaxis='y',
               stackdir='center',
               dotsize = .5,
               fill="red") +
  theme(axis.text.x = element_text(angle=65, vjust=0.6)) +
  labs(title="Box plot + Dot plot",
       subtitle="City Mileage vs Class: Each dot represents 1 row in source data",
       x="Class of Vehicle",
       y="City Mileage")
```

```
## `stat_bindot()` using `bins = 30`. Pick better value with `binwidth`.
```



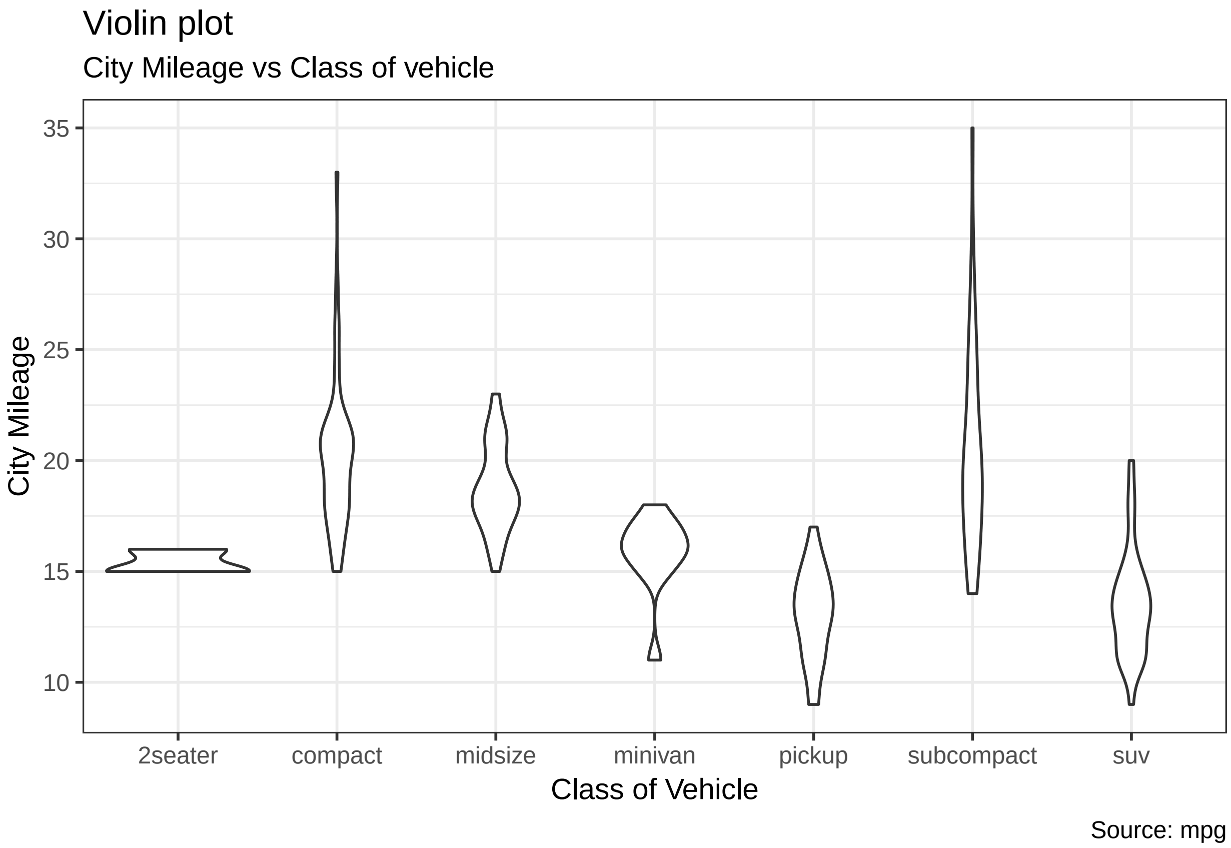
Tufte_Boxplot

```
library(ggthemes)
library(ggplot2)

# plot
mpg %>%
ggplot(aes(manufacturer, cty))+
geom_tufteboxplot() +
theme(axis.text.x = element_text(angle=65, vjust=0.6)) +
  labs(title="Tufte Styled Boxplot",
       subtitle="City Mileage grouped by Class of vehicle",
           x="Class of Vehicle",
           y="City Mileage")
```



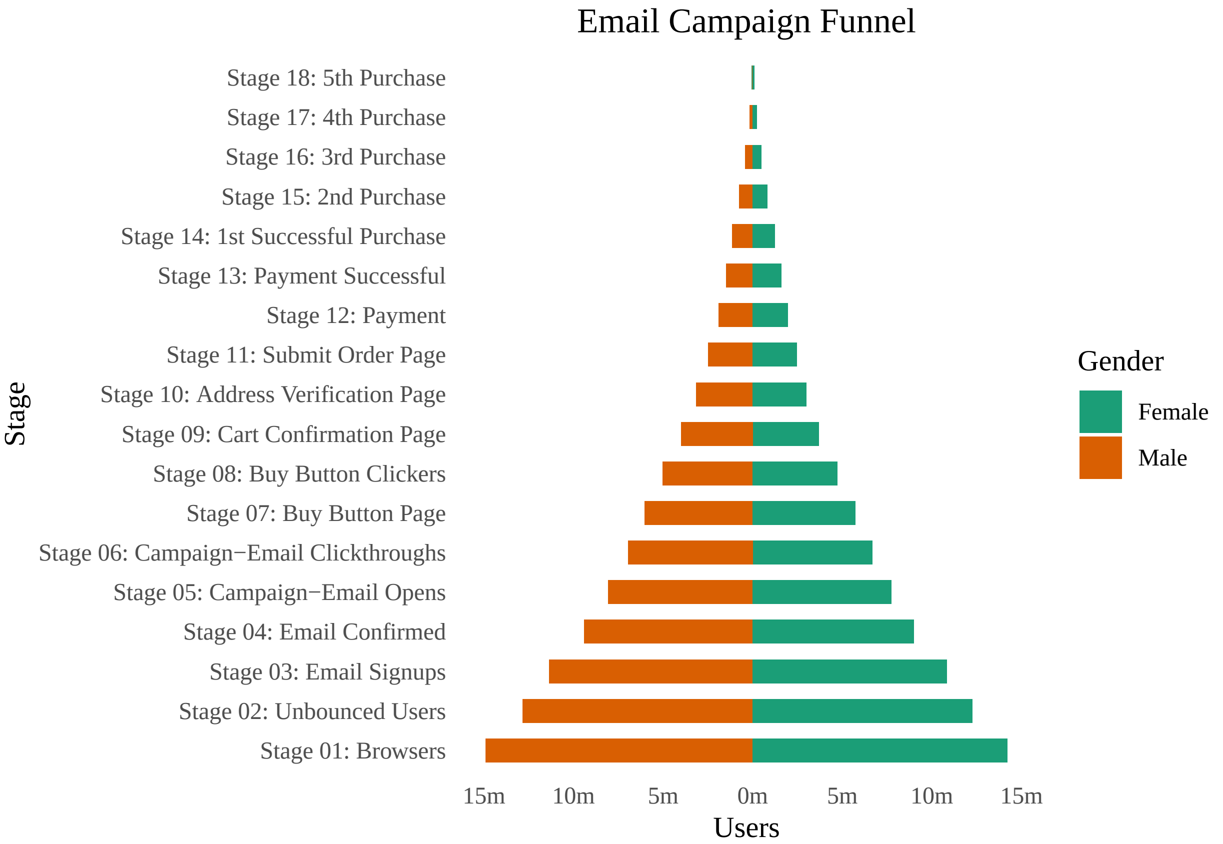
Violin Plot

```
library(ggplot2)

# plot
mpg %>%
ggplot(aes(class, cty)) +
geom_violin() +
labs(title="Violin plot",
     subtitle="City Mileage vs Class of vehicle", caption="Source: mpg",
     x="Class of Vehicle",
     y="City Mileage")
```



Population Pyramid

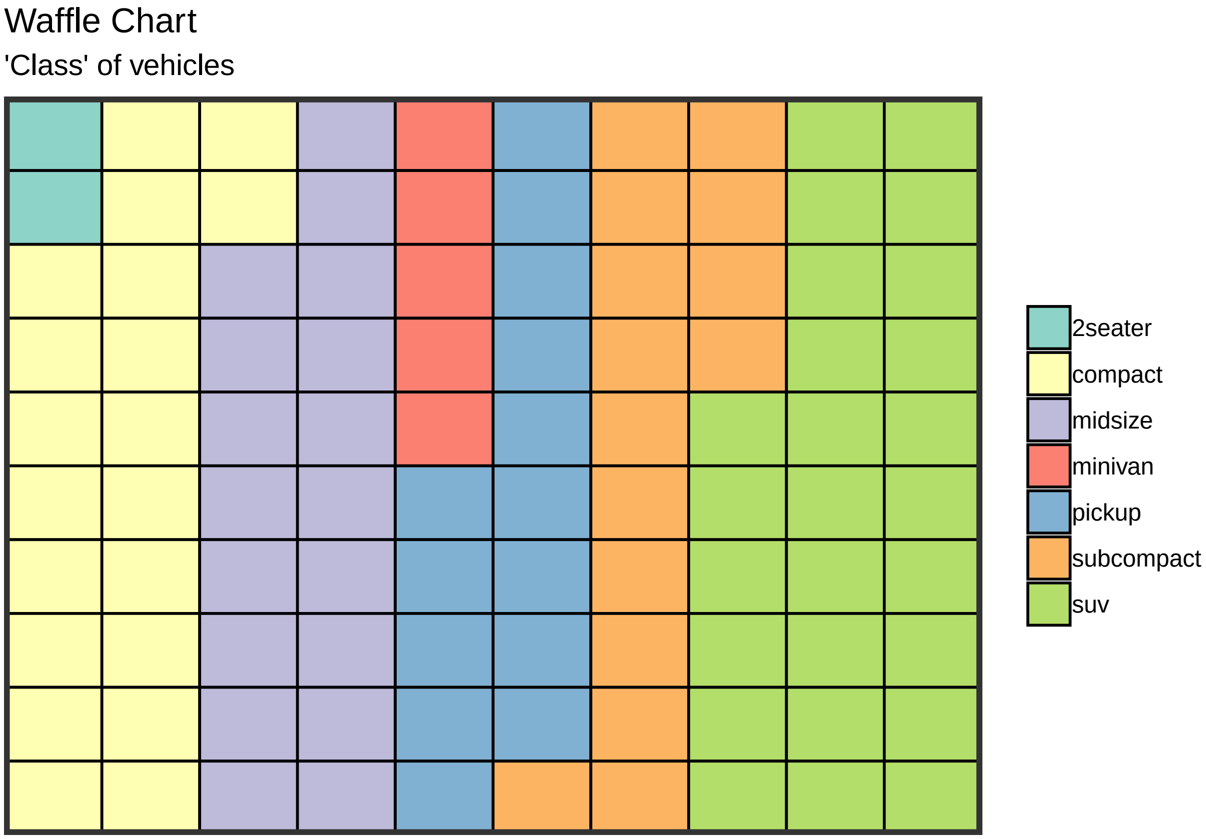
```
options(scipen = 999) # turns of scientific notations like 1e+40

library(ggplot2)
library(ggthemes)

# Read data
email_campaign_funnel <-read.csv("https://raw.githubusercontent.com/shahnp/data/master/email_campaign_fu

# X Axis Breaks and Labels
brks <- seq(-15000000, 15000000, 5000000)
lbls = paste0(as.character(c(seq(15, 0, -5), seq(5, 15, 5))), "m")

# Plot
email_campaign_funnel %>%
  ggplot(aes(x = Stage, y = Users, fill = Gender)) +
  geom_bar(stat = "identity", width = .6) +
  coord_flip() +
  labs(title="Email Campaign Funnel") +
  theme_tufte() + # Tufte theme from ggfortify
  theme(plot.title = element_text(hjust = .5),
  axis.ticks = element_blank()) + # Centre plot title
  scale_y_continuous(breaks = brks,labels = lbls) +  # Breaks  # Labels
  scale_fill_brewer(palette = "Dark2") # Color palette
```



E. Composition

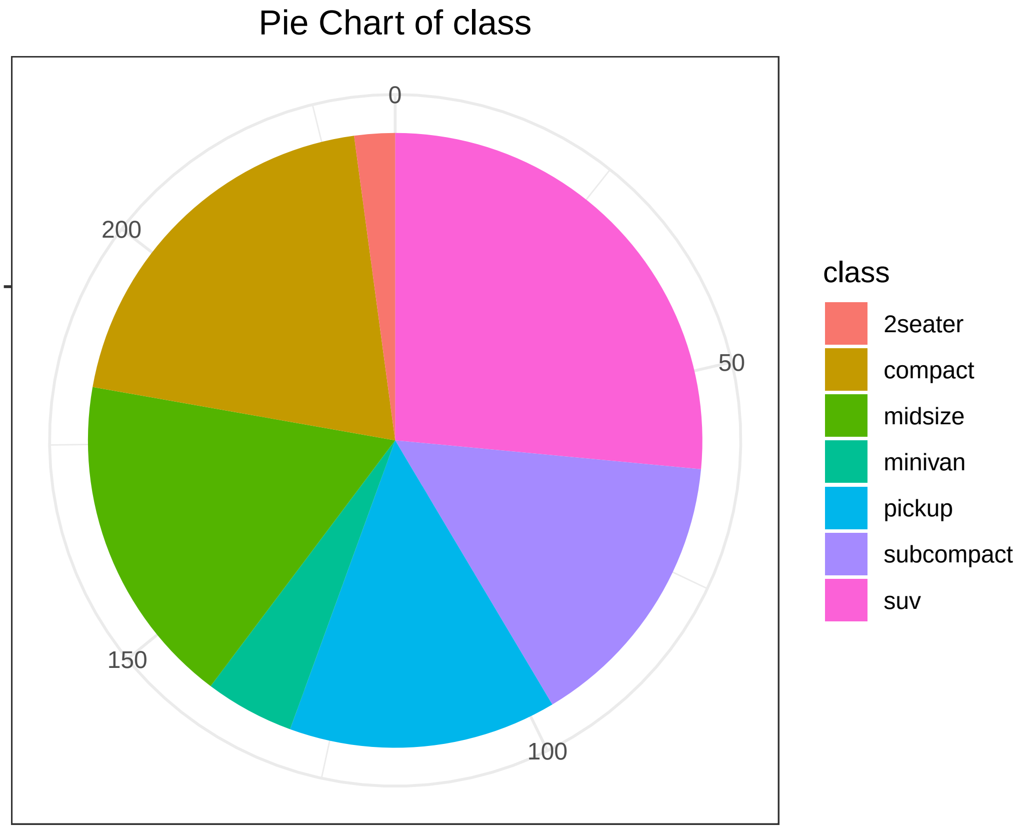
Waffle Chart

```
var <- mpg$class  # the categorical data

## Prep data
nrows <- 10
df <- expand.grid(y = 1:nrows, x = 1:nrows)
categ_table <- round(table(var) * ((nrows*nrows)/(length(var))))

df$category <- factor(rep(names(categ_table), categ_table))

# NOTE: if sum(categ_table) is not 100 (i.e. nrows^2), it will need adjustment to make the sum to 100.

## Plot
df %>%
  ggplot(aes(x = x, y = y, fill = category)) +
  geom_tile(color = "black", size = 0.5) +
  scale_x_continuous(expand = c(0, 0)) +
  scale_y_continuous(expand = c(0, 0), trans = 'reverse') +
  scale_fill_brewer(palette = "Set3") +
  labs(title="Waffle Chart",
       subtitle="'Class' of vehicles") +
  theme(panel.border = element_rect(size = 2),
        plot.title = element_text(size = rel(1.2)),
        axis.text = element_blank(),
        axis.title = element_blank(),
        axis.ticks = element_blank(),
        legend.title = element_blank(),
        legend.position = "right")
```

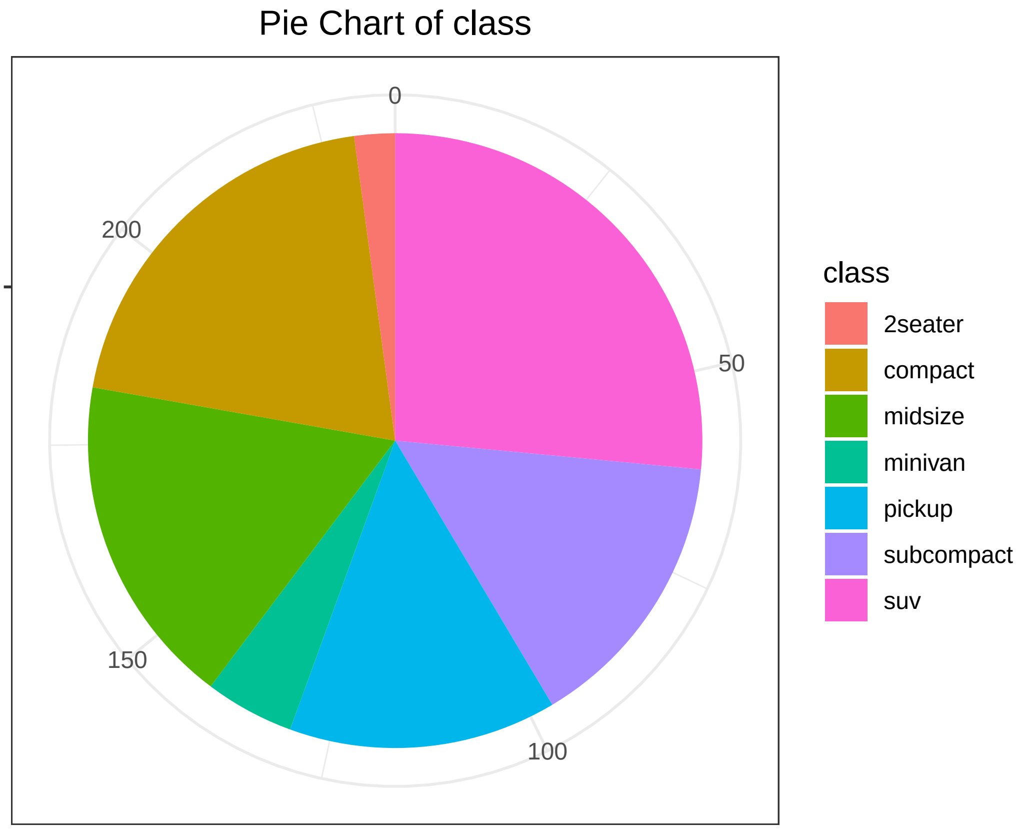


Pie Chart

```
library(ggplot2)

# Source: Frequency table
df <- as.data.frame(table(mpg$class))
colnames(df) <- c("class", "freq")

df %>%
  ggplot(aes(x = "", y=freq, fill = factor(class))) +
  geom_bar(width = 1, stat = "identity") +
  theme(axis.line = element_blank(),
        plot.title = element_text(hjust=0.5)) +
  labs(fill="class",
        x= NULL,
        y= NULL,
        title="Pie Chart of class") +
  coord_polar(theta = "y", start=0)
```

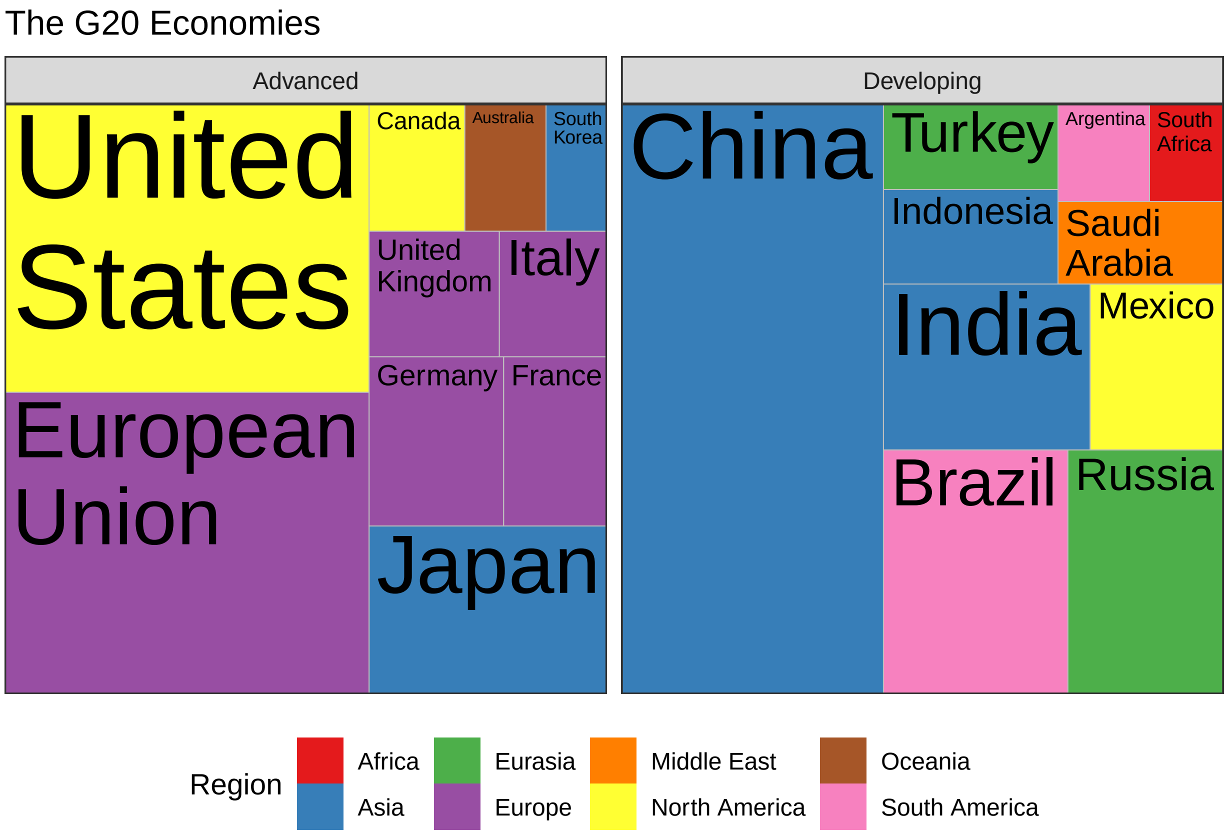


```
# Source: Categorical variable.


mpg %>%
  ggplot(aes(x = "", fill = factor(class))) +
  geom_bar(width = 1) +
  theme(axis.line = element_blank(),
  plot.title = element_text(hjust=0.5)) +
  labs(fill="class",
         x=NULL,
         y=NULL,
         title="Pie Chart of class")+
  coord_polar(theta = "y", start=0)
```



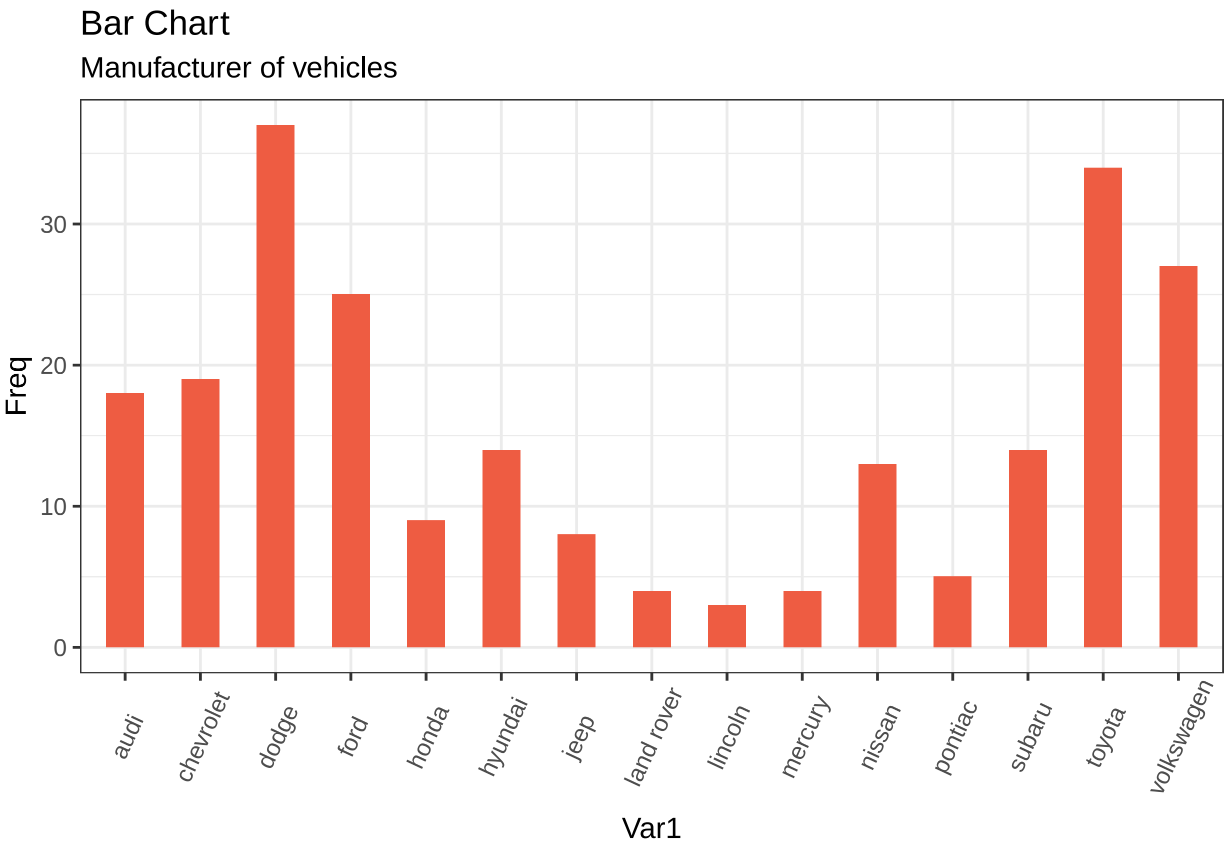
Treemap

```
# Library
library(devtools)
library(treemapify)
library(ggplot2)

# Data
data(G20)

# Plot
G20 %>%
  ggplot(aes(area = gdp_mil_usd, fill= region, label = country)) +
  geom_treemap() +
  geom_treemap_text(grow = T, reflow = T, colour = "black") +
  facet_wrap( ~ econ_classification) +
  scale_fill_brewer(palette = "Set1") +
  theme(legend.position = "bottom") +
  labs(title = "The G20 Economies",
       fill = "Region")
```

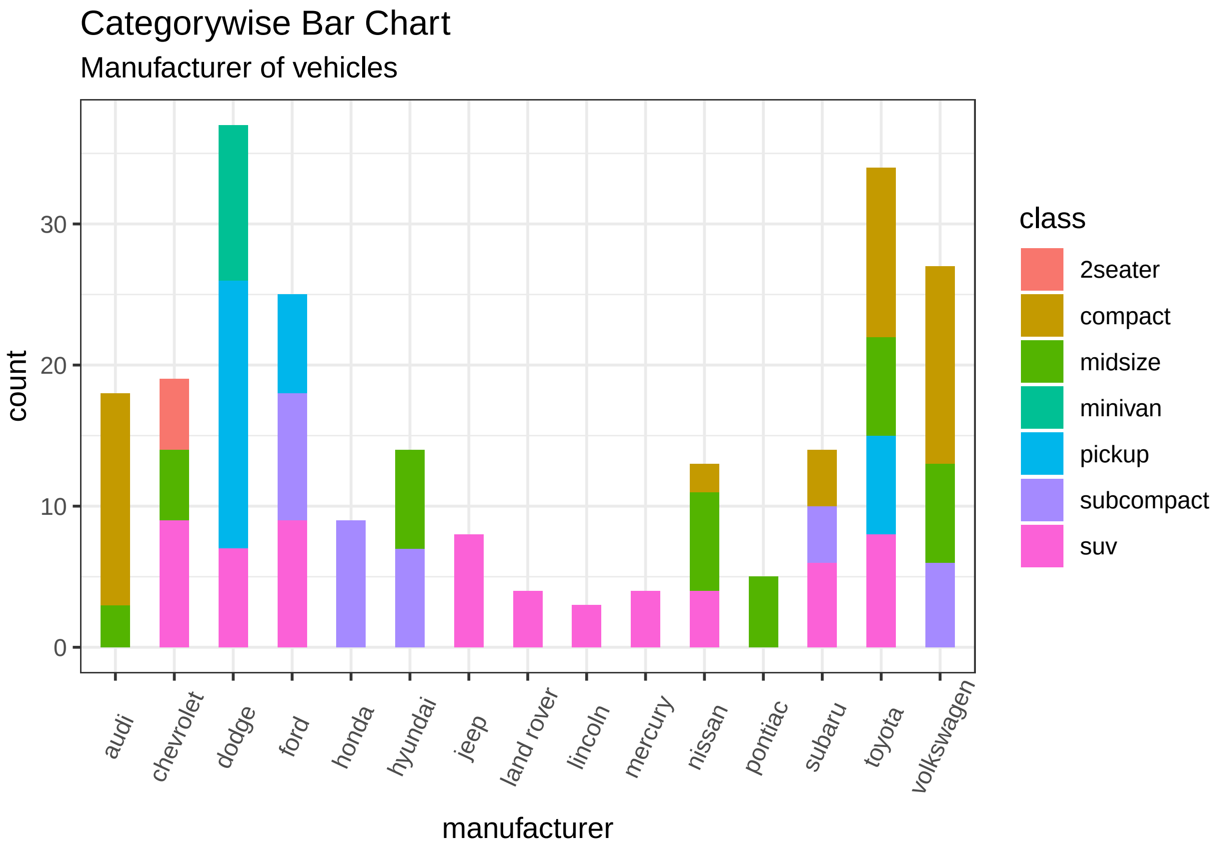


Bar Chart

```
# Library
library(ggplot2)

# Data
freqtable <- table(mpg$manufacturer)
df <- as.data.frame.table(freqtable)

# Plot
df %>%
  ggplot(aes(Var1, Freq))+
  geom_bar(stat="identity", width = 0.5, fill="tomato2") +
  labs(title="Bar Chart",
         subtitle="Manufacturer of vehicles") +
  theme(axis.text.x = element_text(angle=65, vjust=0.6))
```



```
# Categorical variable
mpg %>%
  ggplot(aes(manufacturer))+
  geom_bar(aes(fill=class), width = 0.5) +
  theme(axis.text.x = element_text(angle=65, vjust=0.6)) +
  labs(title="Categorywise Bar Chart",
       subtitle="Manufacturer of vehicles")
```

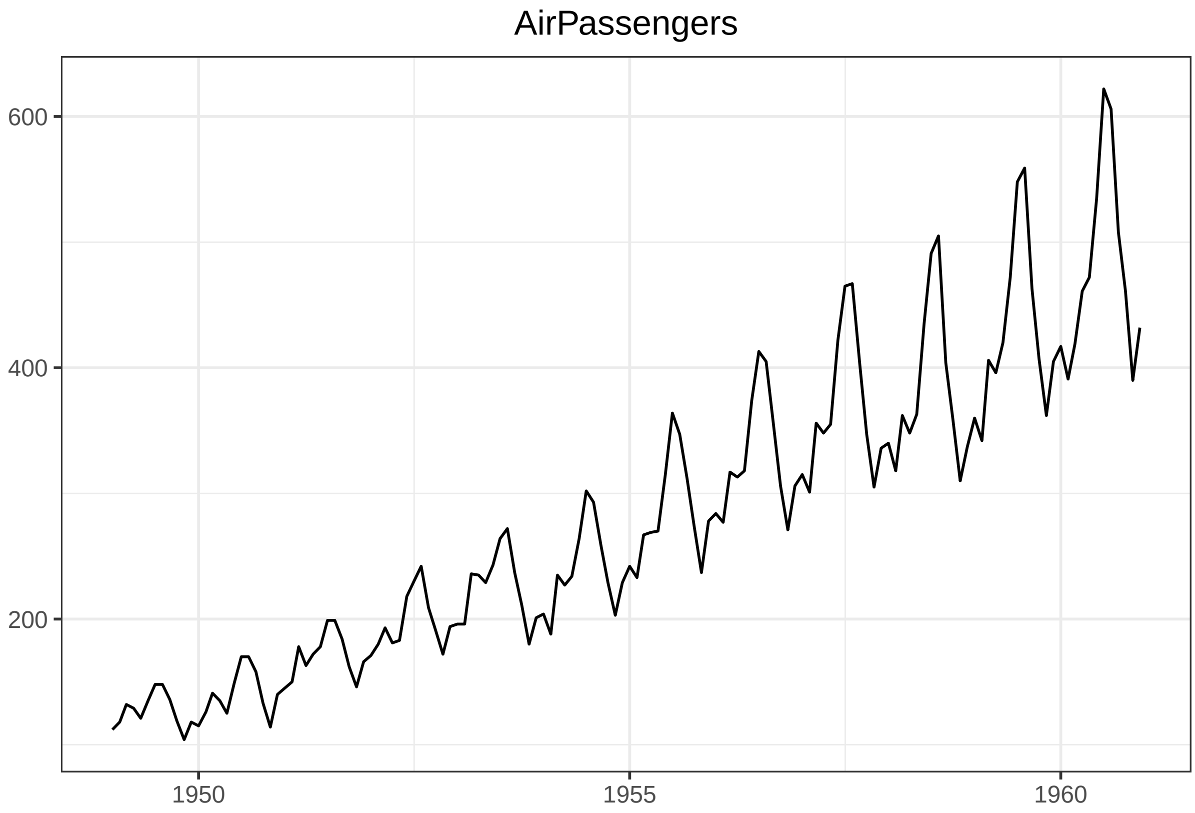


TimeSeries

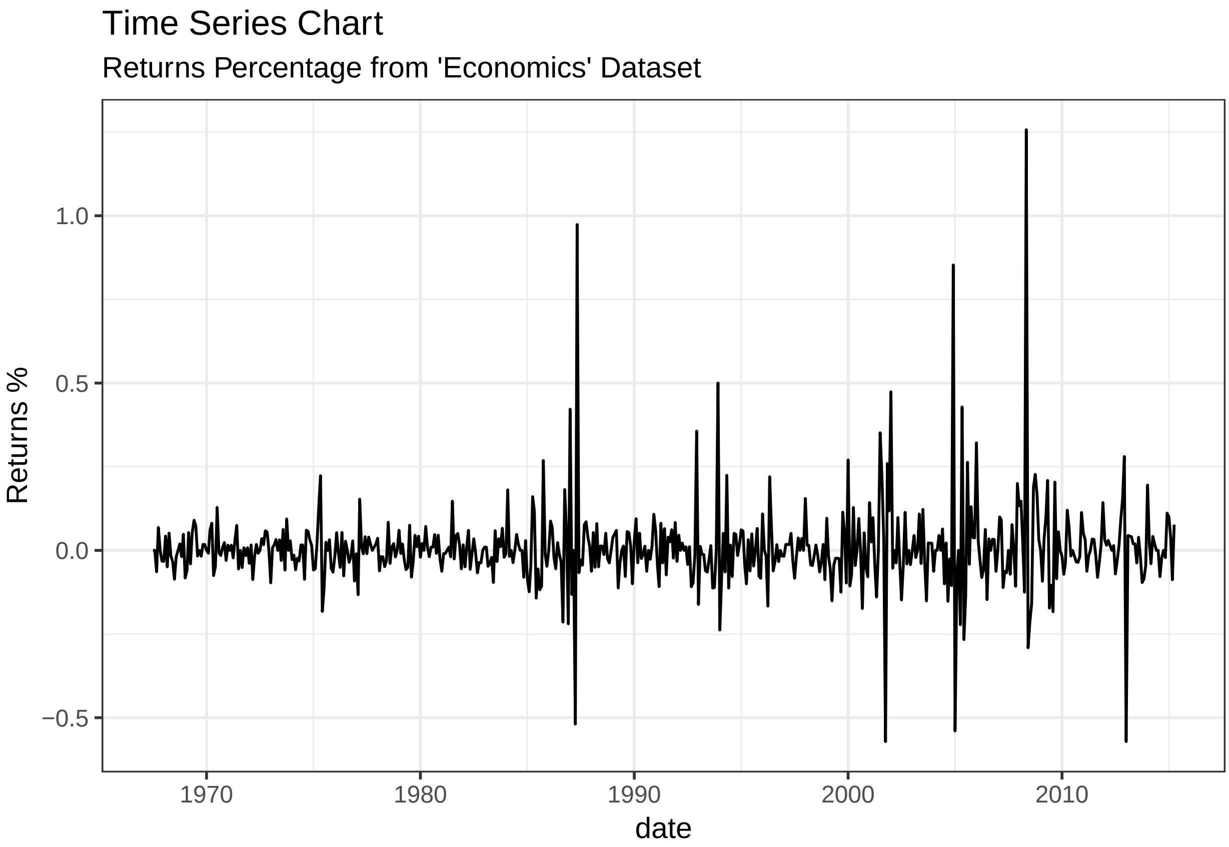
Line graph

Time Series Plot From a Time Series Object (ts)

```
# Library
library(ggplot2)
library(ggfortify)

# Plot A
autoplot(AirPassengers) +
labs(title="AirPassengers") +
  theme(plot.title = element_text(hjust=0.5))
```



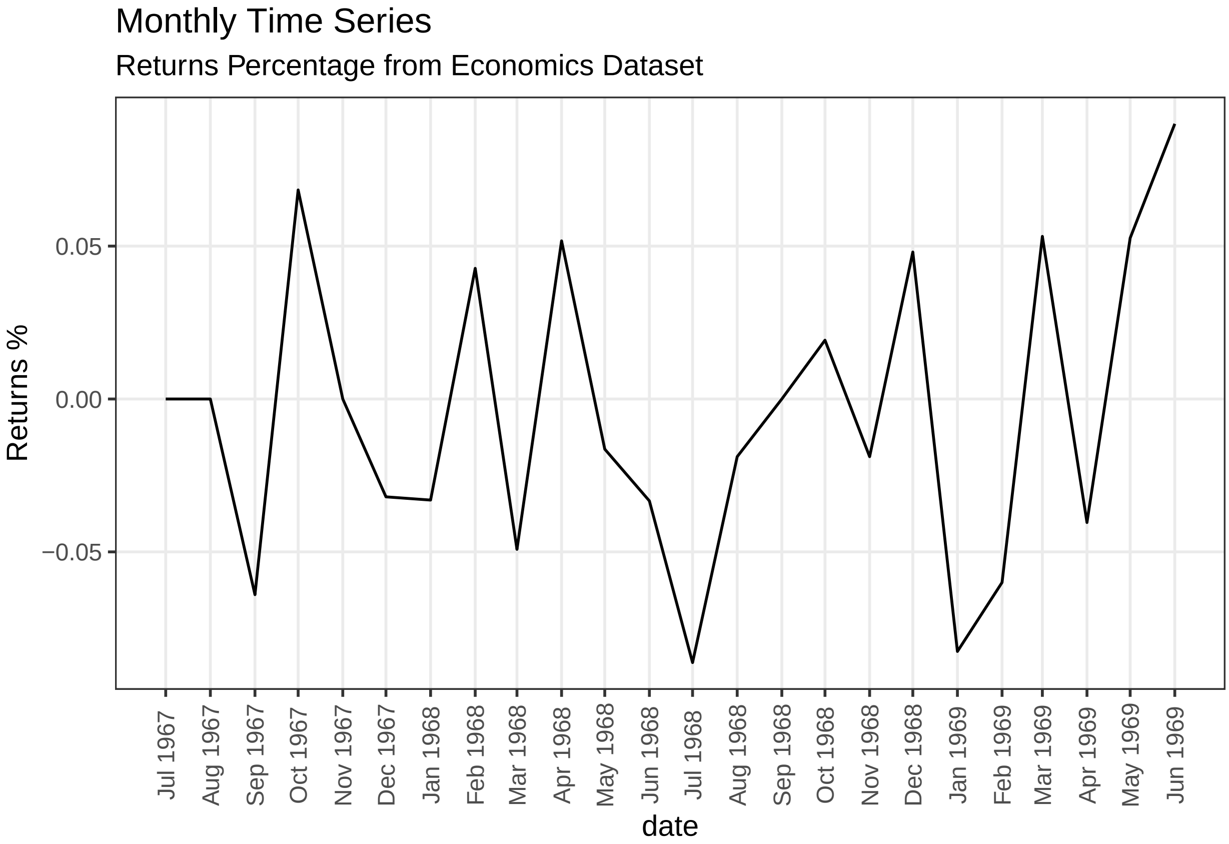
```r
# plot B
economics %>%
  ggplot(aes(x=date)) +
  geom_line(aes(y=returns_perc)) +
    labs(title="Time Series Chart",
       subtitle="Returns Percentage from 'Economics' Dataset",
       y="Returns %")
```



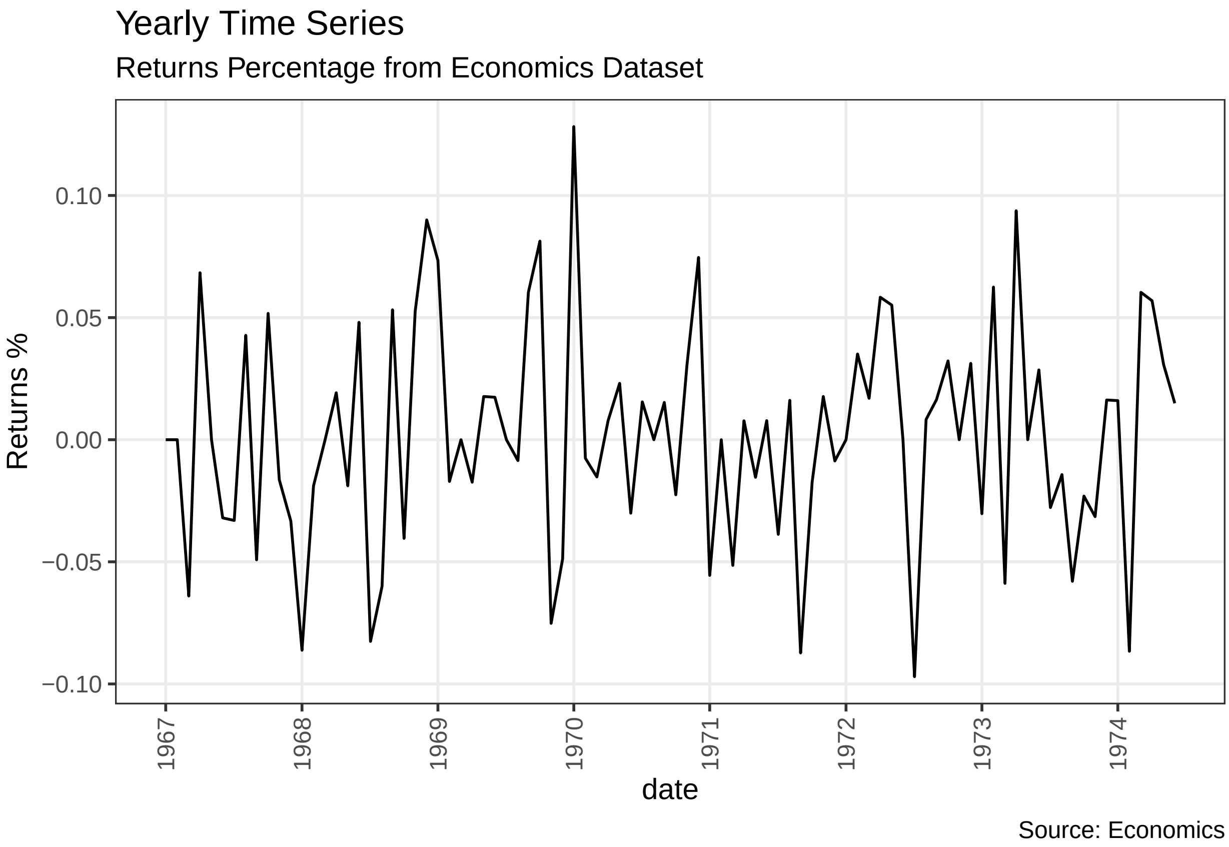
```
# LINE GRAPH

# Plot C:

library(ggplot2)
library(lubridate)
```

```
##
## Attaching package: 'lubridate'

## The following object is masked from 'package:base':
##
##     date
```

```
# Data
economics_m <- economics[1:24, ]

# labels and breaks for X axis text
lbls <- paste0(month.abb[month(economics_m$date)], " ",lubridate::year(economics_m$date))
brks <- economics_m$date

# plot
economics_m %>%
  ggplot(aes(x=date)) +
  geom_line(aes(y=returns_perc)) +
  labs(title="Monthly Time Series",
       subtitle="Returns Percentage from Economics Dataset",
       y="Returns %") +  # title and caption
  scale_x_date(labels = lbls, breaks = brks) + # change to monthly ticks and labels
  theme(axis.text.x = element_text(angle = 90, vjust=0.5), # rotate x axis text
        panel.grid.minor = element_blank()) # turn off minor grid
```



```
# plot D

library(ggplot2)
library(lubridate)

economics_y <- economics[1:90, ]

# labels and breaks for X axis text
brks <- economics_y$date[seq(1, length(economics_y$date), 12)]
lbls <- lubridate::year(brks)

# plot
economics_y %>%
  ggplot(aes(x=date)) +
  geom_line(aes(y=returns_perc)) +
  labs(title="Yearly Time Series",
       subtitle="Returns Percentage from Economics Dataset",
       caption="Source: Economics",
       y="Returns %") +  # title and caption
  scale_x_date(labels = lbls,
               breaks = brks) + # change to monthly ticks and labels
  theme(axis.text.x = element_text(angle = 90, vjust=0.5), # rotate x axis text
      panel.grid.minor = element_blank()) # turn off minor grid
```



Time Series Plot From Long Data Format: Multiple Time Series in Same Dataframe Column
Time Series Plot From Wide Data Format: Data in Multiple Columns of Dataframe

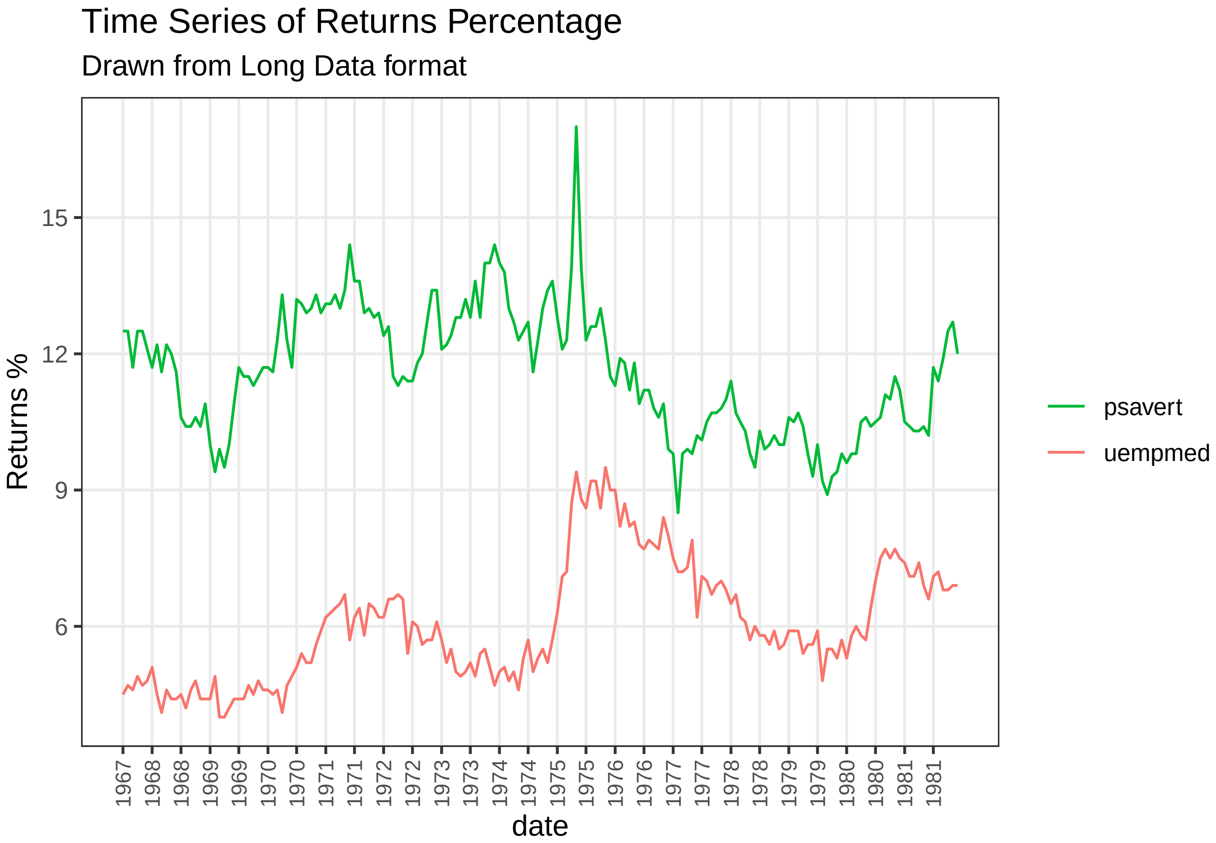
Multiple line graph

```
head(df)
```

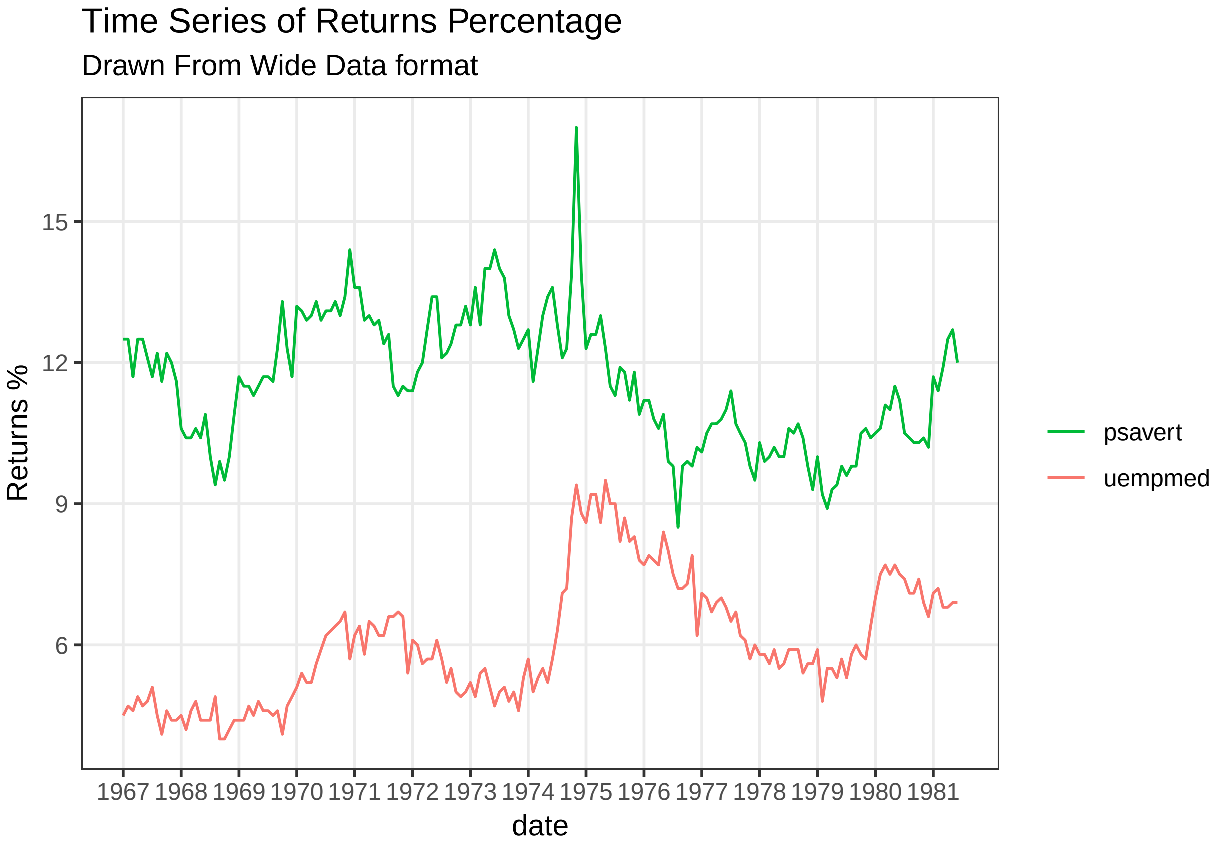
```
##         Var1 Freq
## 1       audi   18
## 2 chevrolet   19
## 3      dodge   37
## 4       ford   25
## 5      honda    9
## 6    hyundai   14
```

```
# Library
library(ggplot2)
library(lubridate)

# Data
data(economics_long, package = "ggplot2")
df <- economics_long[economics_long$variable %in% c("psavert", "uempmed"), ]
df <- df[lubridate::year(df$date) %in% c(1967:1981), ]

brks <- df$date[seq(1, length(df$date), 12)]
lbls <- lubridate::year(brks)
```

```
df %>%
  ggplot(aes(x=date)) +
  geom_line(aes(y=value, col=variable)) +
  labs(title="Time Series of Returns Percentage",
       subtitle="Drawn from Long Data format",
       y="Returns %",
       color=NULL) +  # title and caption
  scale_x_date(labels = lbls, breaks = brks) + # change to monthly ticks and labels
  scale_color_manual(labels = c("psavert", "uempmed"),
  values = c("psavert"="#00ba38", "uempmed"="#f8766d")) + # line color
  theme(axis.text.x = element_text(angle = 90, vjust=0.5, size = 8), # rotate x axis text
  panel.grid.minor = element_blank()) # turn off minor grid
```

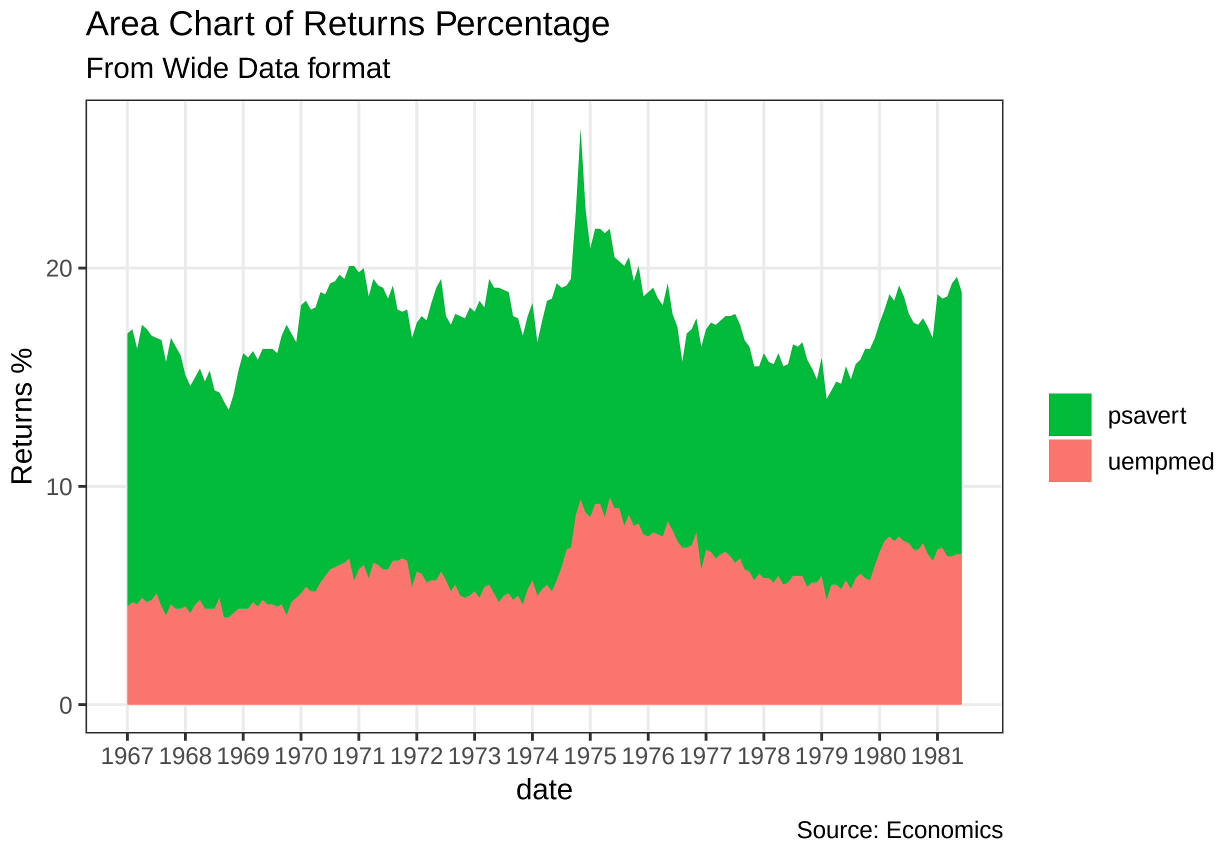


```
# Plot B:

# Time Series Plot From Wide Data Format: Data in Multiple Columns of Dataframe.
library(ggplot2)
library(lubridate)

df <- economics[, c("date", "psavert", "uempmed")]
df <- df[lubridate::year(df$date) %in% c(1967:1981), ]
# labels and breaks for X axis text
brks <- df$date[seq(1, length(df$date), 12)]
lbls <- lubridate::year(brks)

# plot
df %>%
  ggplot(aes(x=date)) +
  geom_line(aes(y=psavert, col="psavert")) +
  geom_line(aes(y=uempmed, col="uempmed")) +
  labs(title="Time Series of Returns Percentage",
       subtitle="Drawn From Wide Data format",
         y="Returns %") + # title and caption
  scale_x_date(labels = lbls, breaks = brks) + # change to monthly ticks and labels
  scale_color_manual(name="",values = c("psavert"="#00ba38", "uempmed"="#f8766d")) + # line color
  theme(panel.grid.minor = element_blank()) # turn off minor grid
```



Stacked Area Chart

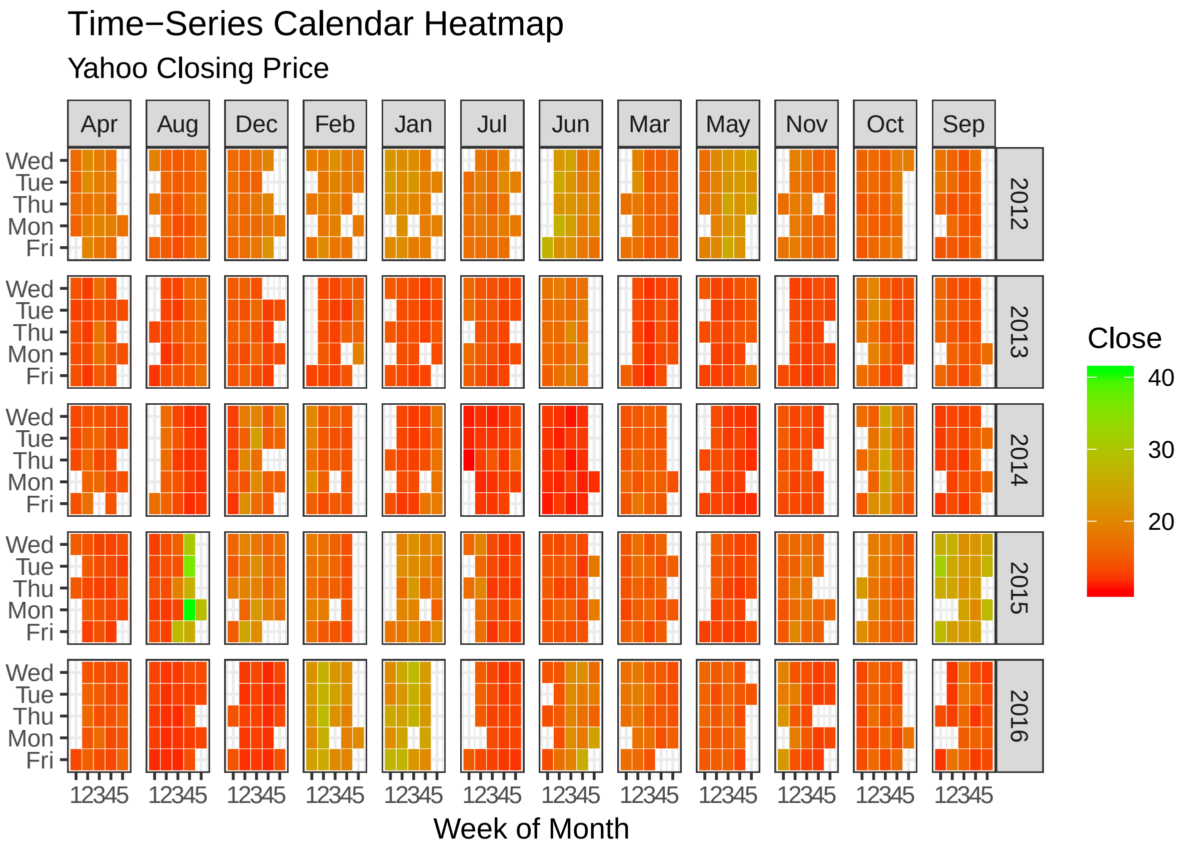
```
library(ggplot2)
library(dplyr)

df <- economics %>% dplyr::select("date", "psavert", "uempmed")
df <- df[lubridate::year(df$date) %in% c(1967:1981), ]

# labels and breaks for X axis text
brks <- df$date[seq(1, length(df$date), 12)]
lbls <- lubridate::year(brks)

# plot
ggplot(df, aes(x=date)) +
geom_area(aes(y=psavert+uempmed, fill="psavert")) +
  geom_area(aes(y=uempmed, fill="uempmed")) +
  labs(title="Area Chart of Returns Percentage",
       subtitle="From Wide Data format",
       caption="Source: Economics",
       y="Returns %") +  # title and caption
scale_x_date(labels = lbls, breaks = brks) + # change to monthly ticks and labels
  scale_fill_manual(name="",values = c("psavert"="#00ba38", "uempmed"="#f8766d")) + # line color
  theme(panel.grid.minor = element_blank()) # turn off minor grid
```



Calendar Heatmap

```
# Library
library(ggplot2)
library(plyr)
library(scales)
library(zoo)

# Data
df <- read.csv("https://raw.githubusercontent.com/shahnp/data/master/yahoo.txt")
df$date <- as.Date(df$date) # format date
df <- df %>% filter(year >= 2012)

# Create Month Week
```

```
df$yearmonth <- as.yearmon(df$date)
df$yearmonth_f <- factor(df$yearmonth)
df <- ddply(df,.(yearmonth_f), transform, monthweek=1+week-min(week)) # compute week numbe r of month
df <- df %>% dplyr::select("year", "yearmonth_f", "monthf", "week", "monthweek", "weekdayf", "VIX.Close

# Plot
df %>%
  ggplot(aes(monthweek, weekdayf, fill = VIX.Close)) +
  geom_tile(colour = "white") +
  facet_grid(year~monthf) +
  scale_fill_gradient(low="red", high="green") +
  labs(x="Week of Month",
       y="",
       title = "Time-Series Calendar Heatmap",
       subtitle="Yahoo Closing Price",
       fill="Close")
```



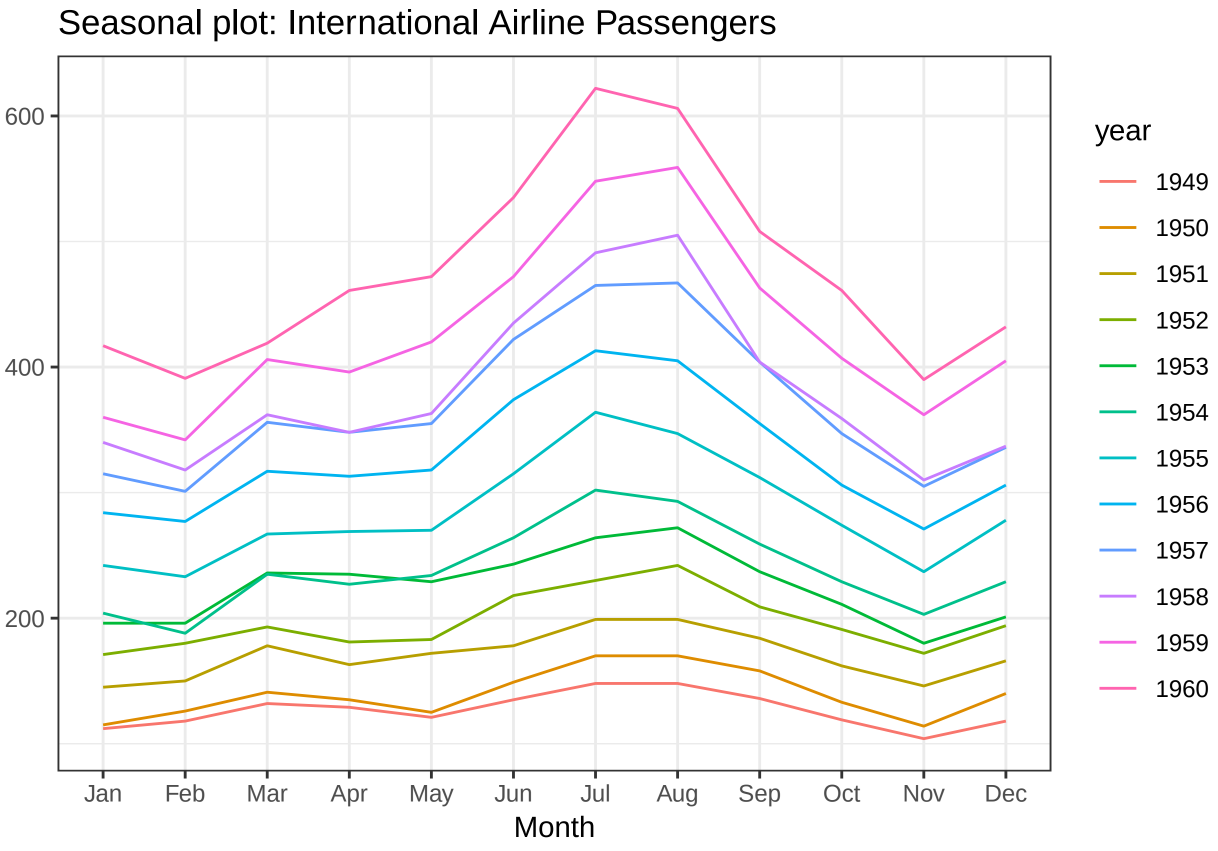
Slope Chart

```
library(dplyr)
library(reshape2)

# Data
source_df <- read.csv("https://raw.githubusercontent.com/shahnp/data/master/cancer_survival_rates.txt")
```

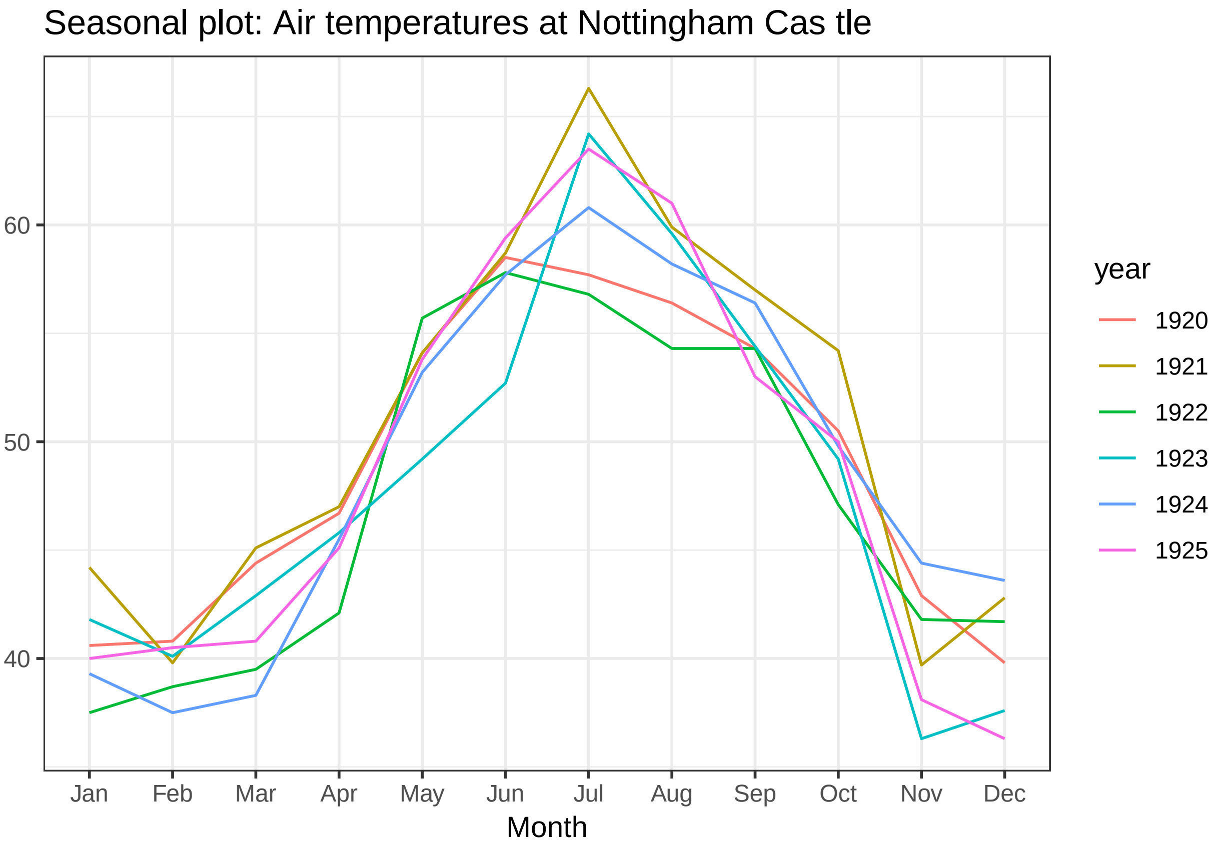
```
tufte_sort <- function(df, x="year", y="value", group="group", method="tufte", min.space=0.05) {
## First rename the columns for consistency
ids <- match(c(x, y, group), names(df))

df <- df %>% dplyr::select(ids)
names(df) <- c("x", "y", "group") # Assisgn colnames

## Expand grid to ensure every combination has a defined value
tmp <- expand.grid(x=unique(df$x), group=unique(df$group))
tmp <- merge(df, tmp, all.y=TRUE)
df <- mutate(tmp, y=ifelse(is.na(y), 0, y))

## Cast into a matrix shape and arrange by first column
tmp <- dcast(df, group ~ x, value.var="y")
ord <- order(tmp[,2])
tmp <- tmp[ord,]
min.space <- min.space*diff(range(tmp[,-1]))
yshift <- numeric(nrow(tmp))

## Start at "bottom" row , Repeat for rest of the rows until you hit the top
for (i in 2:nrow(tmp)) {
## Shift subsequent row up by equal space so gap between ## two entries is >= minimum
mat <- as.matrix(tmp[(i-1):i, -1])
d.min <- min(diff(mat))
yshift[i] <- ifelse(d.min < min.space, min.space - d.min, 0) }
tmp <- cbind(tmp, yshift=cumsum(yshift))
scale <- 1

## Store these gaps in a separate variable so that they can be scaled ypos = a*yshift +y
tmp <- melt(tmp, id=c("group", "yshift"), variable.name="x", value.name="y")
tmp <- transform(tmp, ypos=y + scale*yshift)
return(tmp)
}

plot_slopegraph <- function(df) {
ylabs <- subset(df, x==head(x,1))$group
yvals <- subset(df, x==head(x,1))$ypos
fontSize <- 3

gg <- df %>%
      ggplot(aes(x=x,y=ypos)) +
      geom_line(aes(group=group),colour="grey80") +
      geom_point(colour="white",size=8) +
      geom_text(aes(label=y), size=fontSize, family="American Typewriter") +
      scale_y_continuous(name="", breaks=yvals, labels=ylabs)
      return(gg)
}

## Prepare data
df <- tufte_sort(source_df,
                 x="year",
                 y="value",
                 group="group",
                  method="tufte",
                  min.space=0.05)

df <- transform(df, x=factor(x, levels=c(5,10,15,20),
                       labels=c("5 years","10 years","15 years","20 years")),
                       y=round(y))

## Plot
# plot_slopegraph(df) +
#   labs(title="Estimates of % survival rates") +
# theme(axis.title=element_blank(),
#       axis.ticks = element_blank(),
#       plot.title = element_text(hjust=0.5,family = "American Typewriter",face="bold"),
#       axis.text = element_text(family = "American Typewriter", face="bold"))
```

Seasonal line Chart

```
library(ggplot2)
library(forecast)

# Subset data
nottem_small <- window(nottem, start=c(1920, 1), end=c(1925, 12)) # subset a smaller time window
# Plot
ggseasonplot(AirPassengers) + labs(title="Seasonal plot: International Airline Passengers" )
```

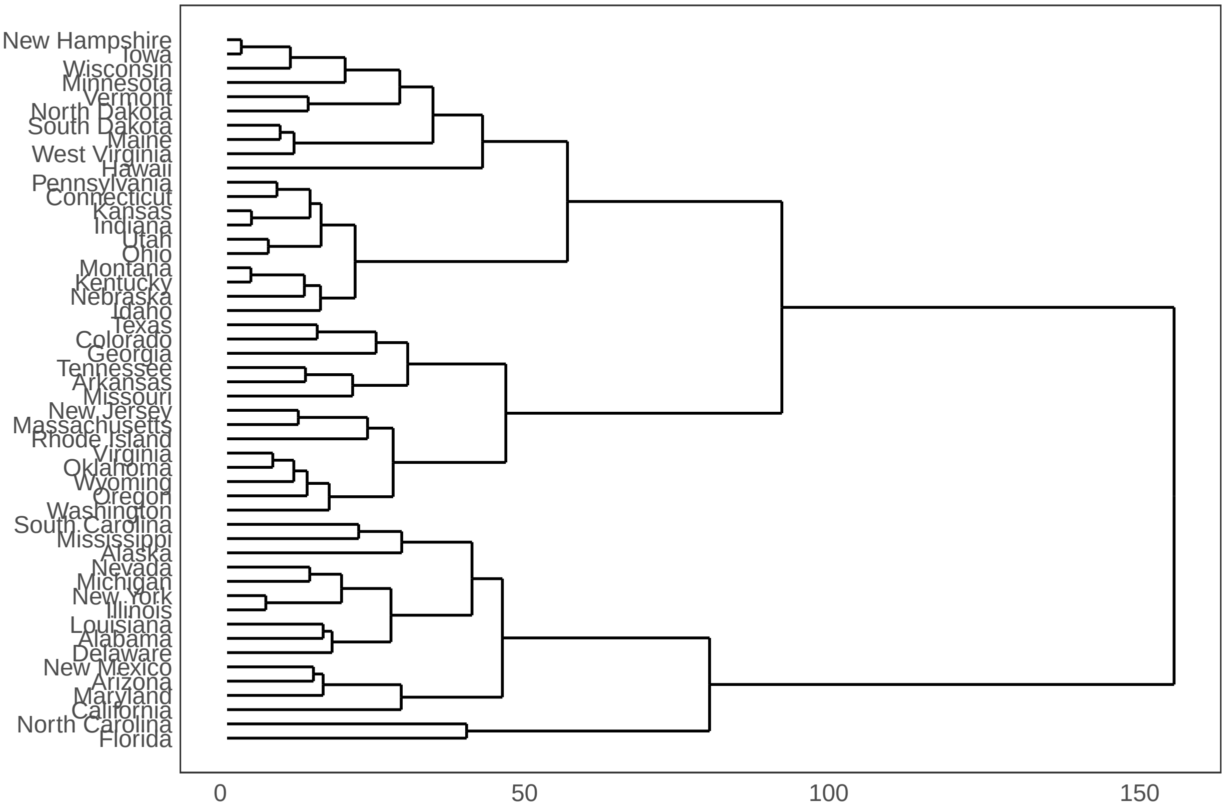


```
ggseasonplot(nottem_small) + labs(title="Seasonal plot: Air temperatures at Nottingham Cas tle")
```



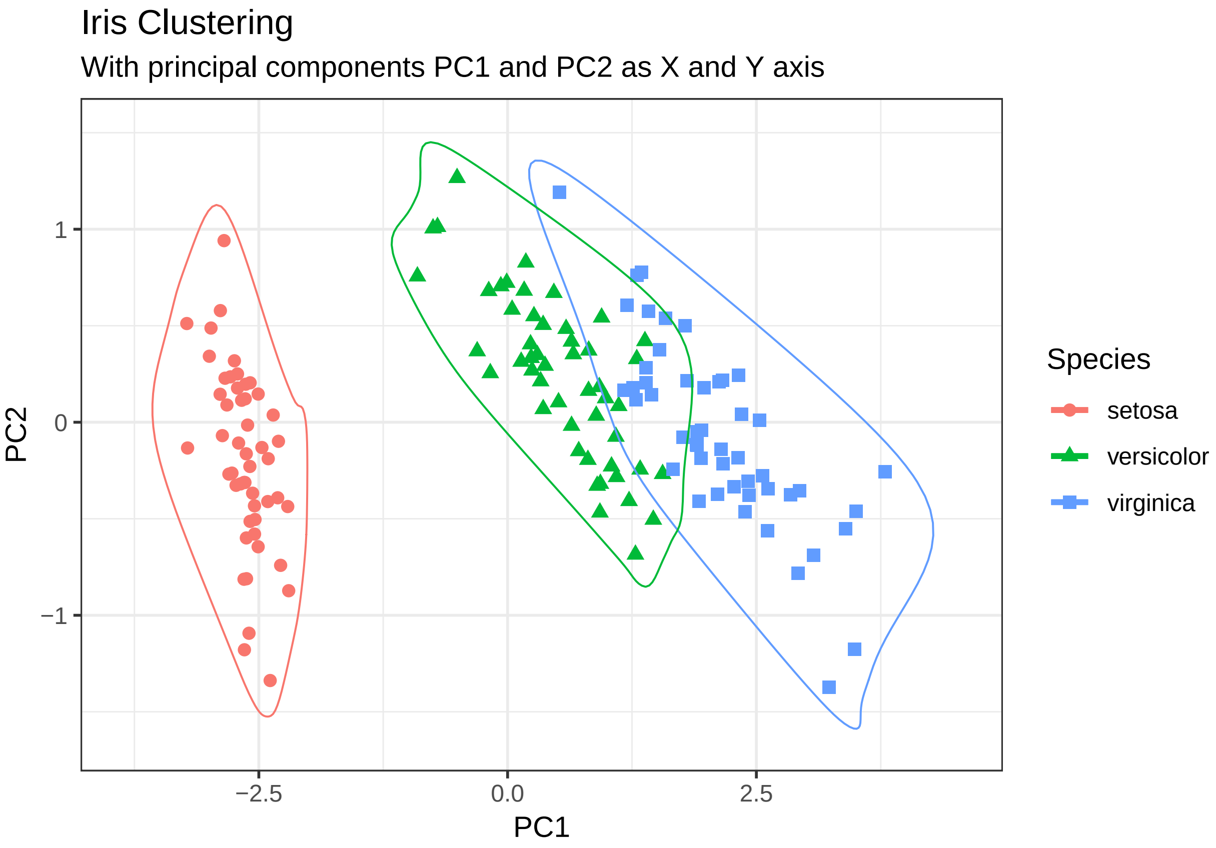
Hierarchical Dendrogram

```
library(ggplot2)
library(ggdendro)

# Data
hc <- hclust(dist(USArrests), "ave") # hierarchical clustering

# plot
ggdendrogram(hc, rotate = TRUE, size = 2)
```



Clusters

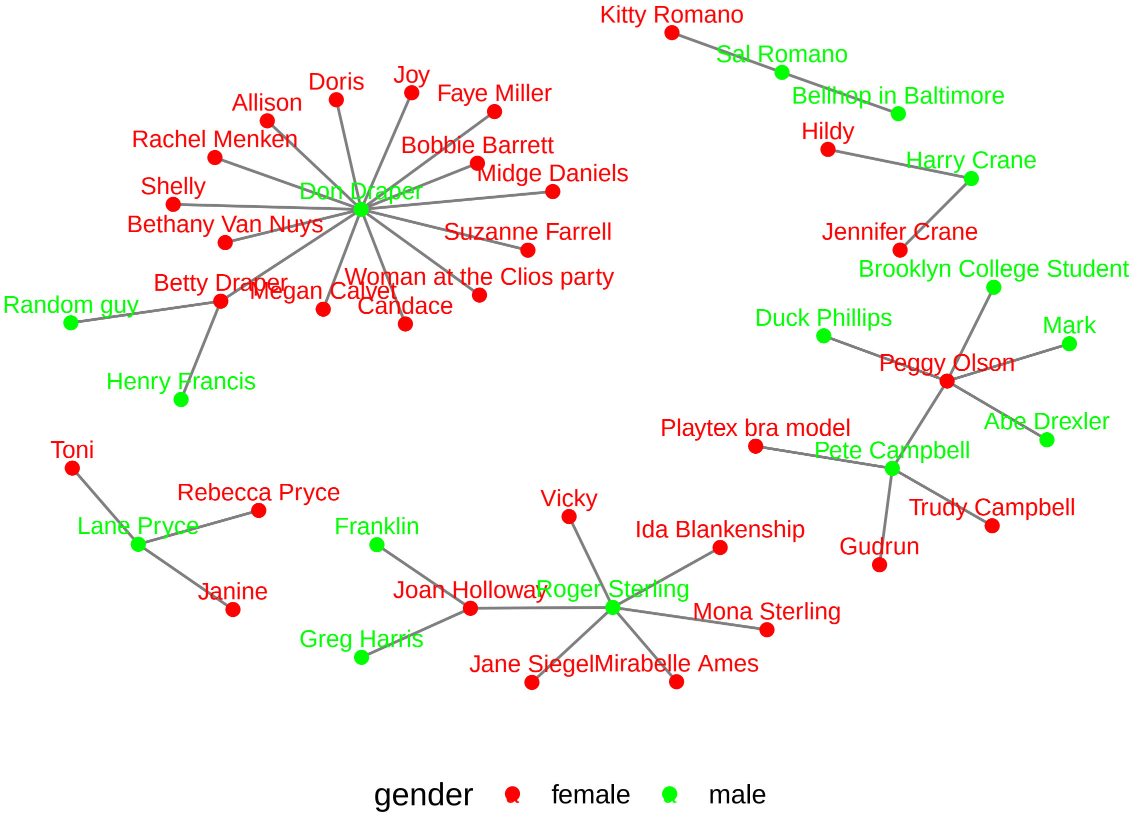
```
library(ggplot2)
library(ggalt)
library(ggfortify)

# Compute data with principal components ------------------
df <- iris[c(1, 2, 3, 4)]
pca_mod <- prcomp(df) # compute principal components

# Data frame of principal components ----------------------
df_pc <- data.frame(pca_mod$x, Species=iris$Species) # dataframe of principal components
df_pc_vir <- df_pc %>% dplyr::filter(Species == "virginica")
df_pc_set <- df_pc %>% dplyr::filter(Species == "setosa") # df for 'setosa'
df_pc_ver <- df_pc %>% dplyr::filter(Species == "versicolor") # df for 'versicolor'

# Plot ----------------------------------------------------
df_pc %>%
  ggplot(aes(PC1, PC2, col=Species)) +
  geom_point(aes(shape=Species), size=2) + # draw points
  labs(title="Iris Clustering",
       subtitle="With principal components PC1 and PC2 as X and Y axis") +
  coord_cartesian(xlim = 1.2 * c(min(df_pc$PC1), max(df_pc$PC1)),
                  ylim = 1.2 * c(min(df_pc$PC2), max(df_pc$PC2))) + # change axis limits
  geom_encircle(data = df_pc_vir, aes(x=PC1, y=PC2)) + # draw circles
  geom_encircle(data = df_pc_set, aes(x=PC1, y=PC2)) +
  geom_encircle(data = df_pc_ver, aes(x=PC1, y=PC2))
```



Network Visulization

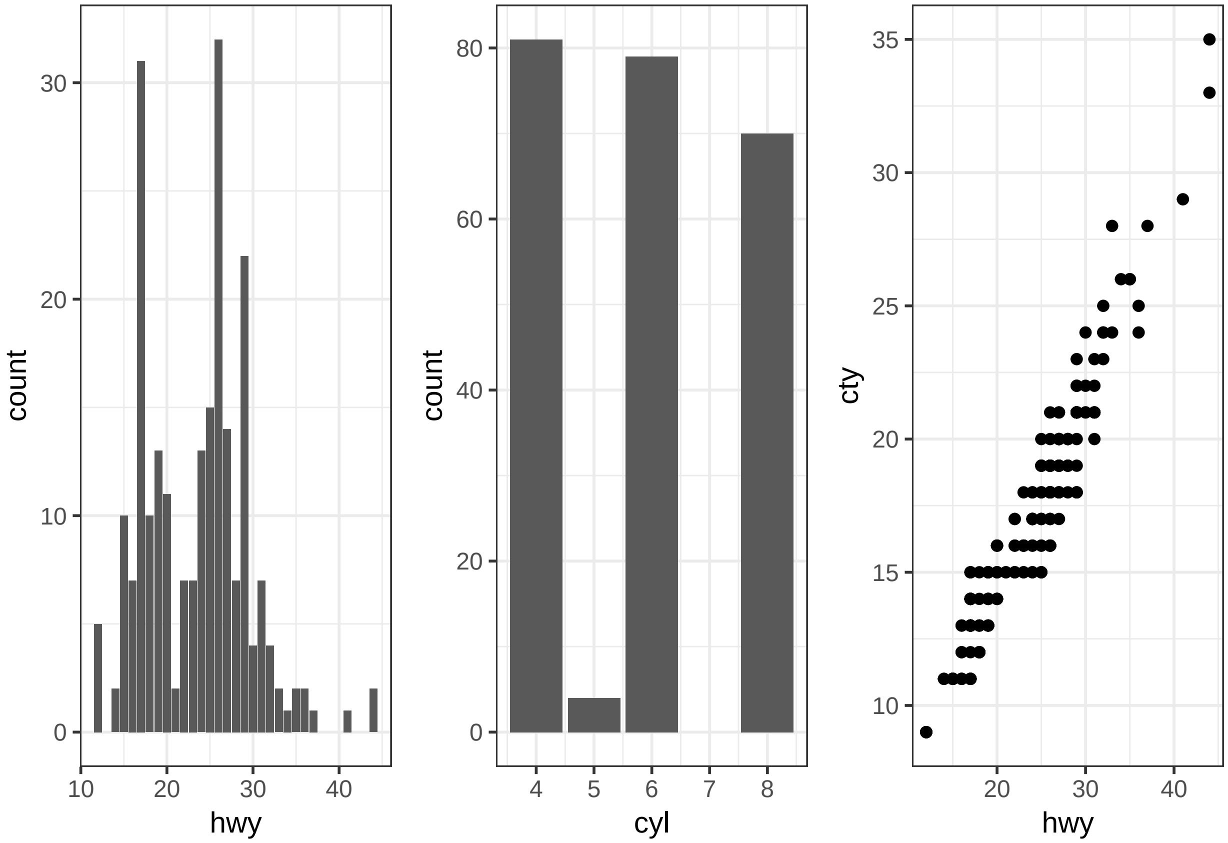
```
# Library
library(ggplot2)
library(ggnetwork)
library(geomnet)
library(network)

# Data
data(madmen, package = 'geomnet')

# create undirected network
mm.net <- network(madmen$edges[, 1:2],directed = FALSE) # mm.net :glance at network object

# create node attribute (gender)
rownames(madmen$vertices) <- madmen$vertices$label
mm.net %v% "gender" <- as.character(madmen$vertices[ network.vertex.names(mm.net),"Gender"])

# gender color palette
mm.col <- c("female" = "#ff0000", "male" = "#00ff00")
set.seed(10052016)
```

```r
ggplot(data = ggnetwork(mm.net, layout = "kamadakawai"),aes(x, y, xend = xend, yend =yend)) +
  geom_edges(color = "grey50") + # draw edge layer
  geom_nodes(aes(colour = gender), size = 2) + # draw node layer
  geom_nodetext(aes(colour = gender,
                    label = vertex.names),
                     size = 3, vjust = -0.6) + # draw node label layer
  scale_colour_manual(values = mm.col) +
  xlim(c(-0.05, 1.05)) +
  theme_blank() +
  theme(legend.position = "bottom")
```



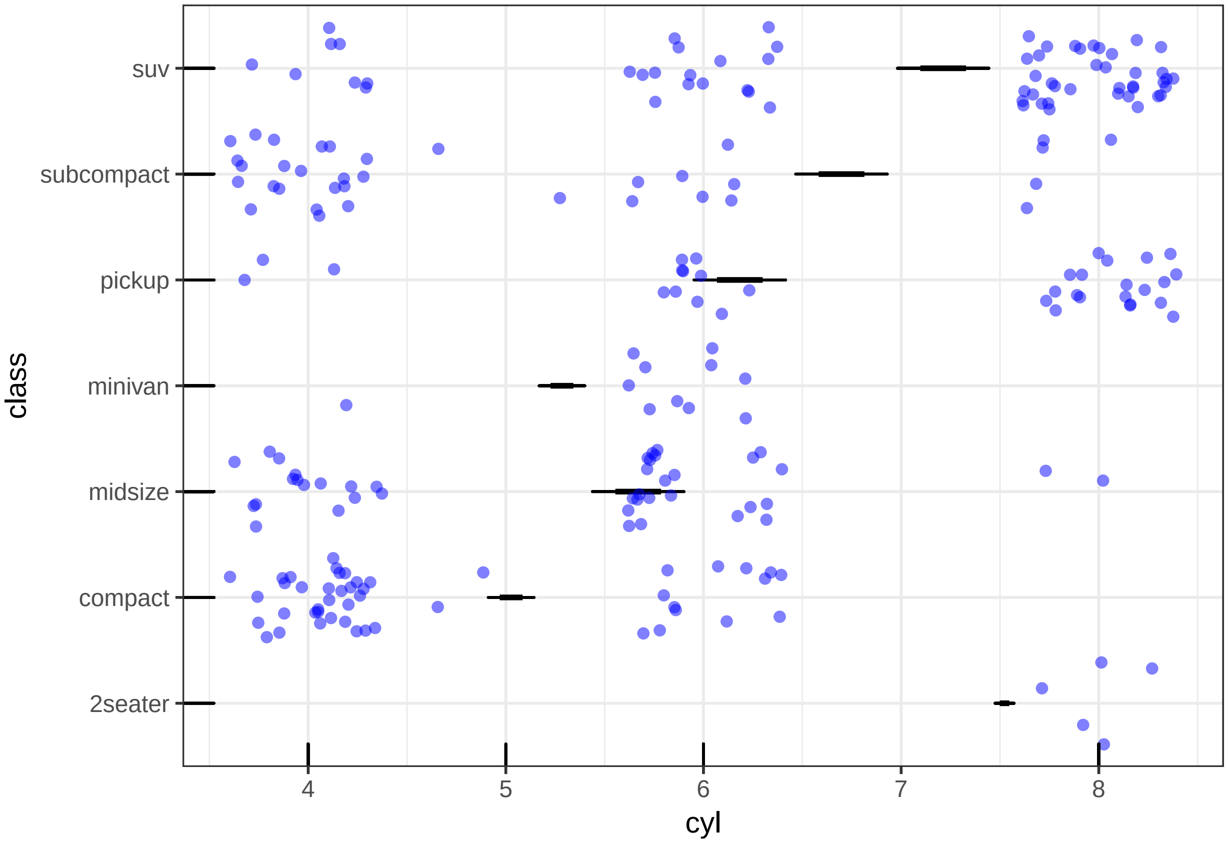
Multiple Graphs per Page

```r
# Library
library(ggplot2)
library(gridExtra)

p1 <- mpg %>% ggplot(aes(x=hwy)) + geom_bar()
p2 <- mpg %>% ggplot(aes(x=cyl)) + geom_bar()
p3 <- mpg %>% ggplot(aes(x=hwy, y=cty)) + geom_point()
grid.arrange(p1, p2, p3, ncol=3)
```



Rug_boxplot

```
library(ggplot2)

mpg %>%
ggplot(aes(x=cyl, y=class)) +
     geom_boxplot(fill="cornflowerblue", color="black", notch=TRUE)+
     geom_point(position="jitter",color="blue", alpha=.5)+
     geom_rug(side="l", color="black")
```



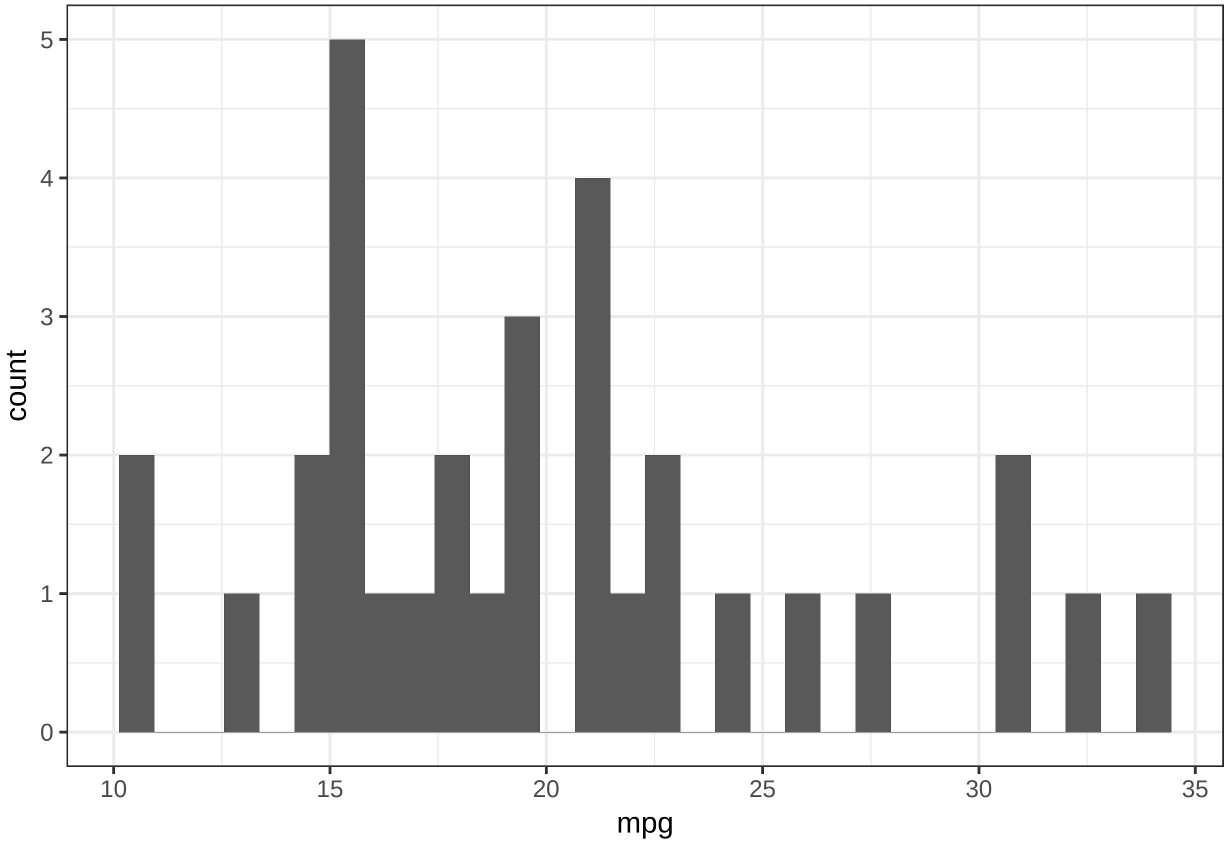
Saving Graphs

The available formats include .ps, .tex, .jpeg, .pdf, .jpg, .tiff, .png, .bmp, .svg, or .wmf (the latter only being available on Windows machines).

SAVE the graph

```
# png
knitr::opts_chunk$set(fig.width=7,fig.height=4)

myplot <- mtcars %>% ggplot(aes(x=mpg)) +
                        geom_histogram(bins = 30)

ggsave(file="mygraph.png", plot=myplot,width=5, height=4)

# PDF

mtcars %>%
  ggplot(aes(x=mpg)) +
  geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



```
ggsave(file="mygraph.pdf")
```

```
## Saving 6.5 x 4.5 in image
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```