Tutorial 2
Data Visualization using R

R Visualization Packages

- R currently has one of the most advanced graphical libraries
- **Graphic systems in R include:**
  - **Base R graphics (or traditional graphics):** Typically the default graphics system built into R & easiest to learn, e.g., plot, bar, line, scatter
  - **Grid graphics:** provide functions for drawing complete plots
  - **Lattice graphics:**
    - Based on the grid package & implemented in the lattice package
    - Provides alternative implementation of standard plotting functions available in base graphics, including scatterplots, bar charts, boxplots, histograms & QQ-plots
  - **ggplot2:**
    - Most used visualization package by data scientists
    - Similar to lattice graphics, but with a fundamentally different structure

ggplot2

- Most popular visualization package in R
- Developed by Hadley Wickham in 2005 for creating presentation-quality visualization

- ***"gg"*** stands for "***grammar of graphics***":
  - Term coined by Leland Wilkinson to define a system of plotting theory & nomenclature
  - Breaks down the notion of a graphic into its constituent parts (the data, scales, coordinates, geometries, aesthetics)

- **Graphing functions in ggplot2:** qplot & ggplot

ggplot Representation

- Every ggplot2 plot is based on 3 key components:
  - **Data**
  - A set of **aesthetic mappings** between variables in the data & visual properties
  - **Geom** layer, which describes how to render each observation

- Basic ggplot object can be represented by:

ggplot(dataset_name, mapping) +
layer(stat = “ ”, geom = “ ", position = “ ”)

mapping = *aes (x = dataset_name$VarX, y =dataset_name$VarY)*

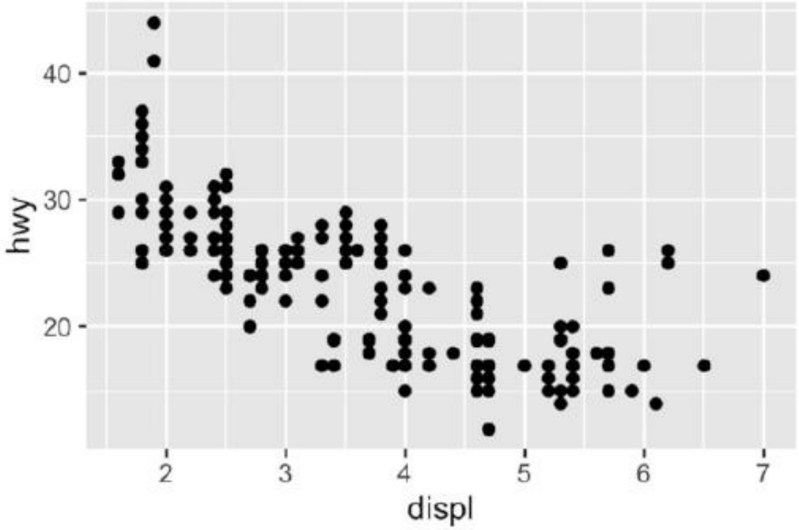
where varX & varY are the variables for plotting on the x- and y-axes

Example: ggplot Representation

ggplot(mpg, aes(x = displ, y = hwy)) +

geom_point()

- Dataset produces a scatterplot defined by:
  1. **Data:** *mpg dataset in ggplot*
  2. **Aesthetic mapping:** engine size mapped to *x* position, fuel economy to *y* position.
  3. **Layer:** points
- **Note:** Data & aesthetic mappings are supplied in ggplot(), then layers are added on with +



Parameters for Graphing Functions

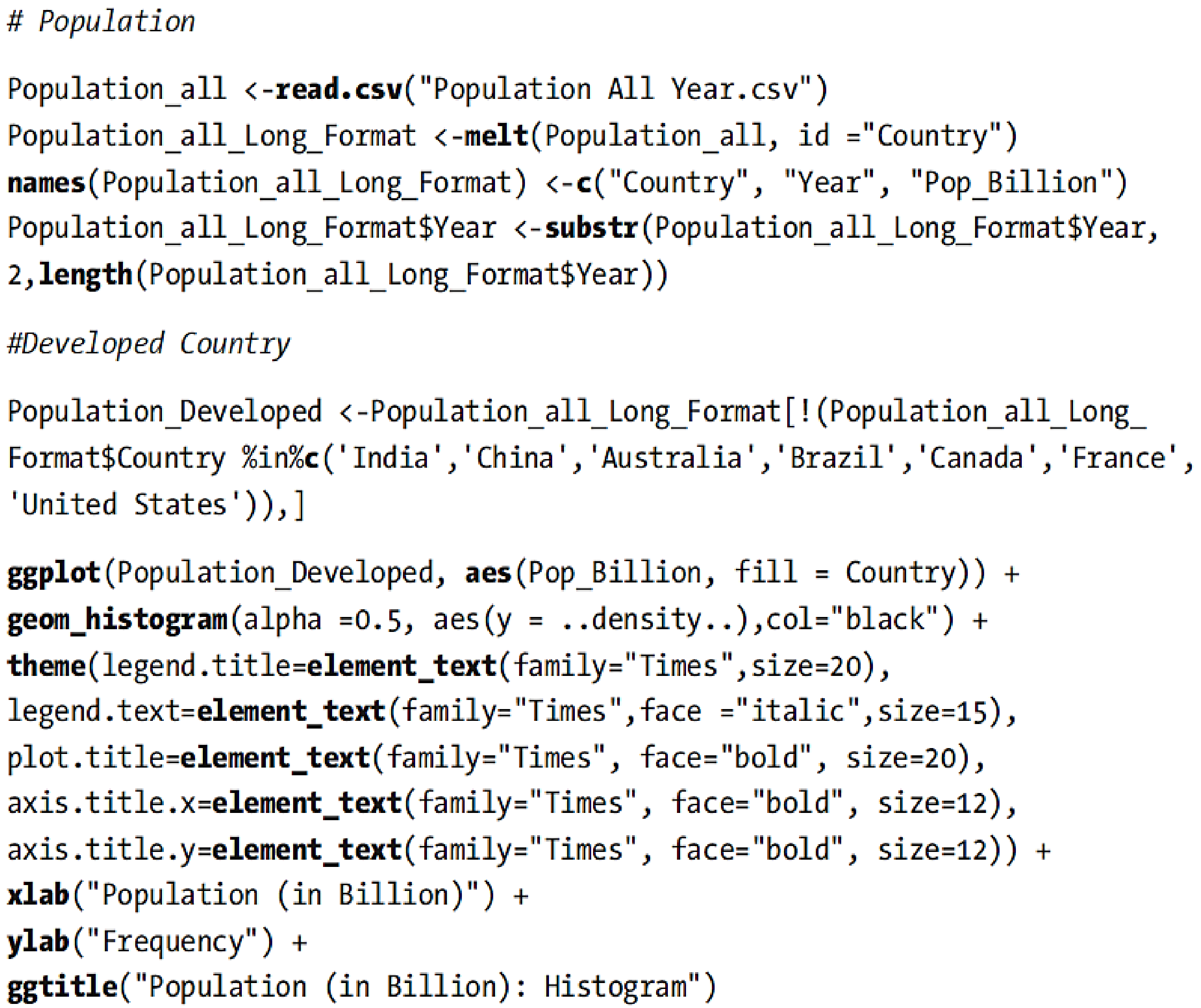
- ***geom_:***
  - Geoms, short for geometric objects, describe the type of plot
  - Ex: *geom_point, geom_histogram, geom_bar & geom_line*
- ***stat_:***
  - Statistical functions that transform the data in the plot before plotting
  - Ex: *stat_abline, stat_smooth & stat_box_plot*
- ***scale_:***
  - Control how data are mapped to the plot on the basis of esthetics
  - Ex: *scale_size & scale_gradient*

Parameters for Graphing Functions

- ***coord_:***
  - Changes the coordinate systems of plots to the 2D plane of the computer screen
  - Ex: *coord_cartesian, coord_flip*
- ***facet_:***
  - Allow subsets of data to be displayed in different panels for emphasis or to focus on particular parts of data and graph
  - Ex: *facet_grid*
- ***position_:***
  - Adjust the position of points in a plot & allow for fine-tuning
  - Ex: dodging, jittering & stacking

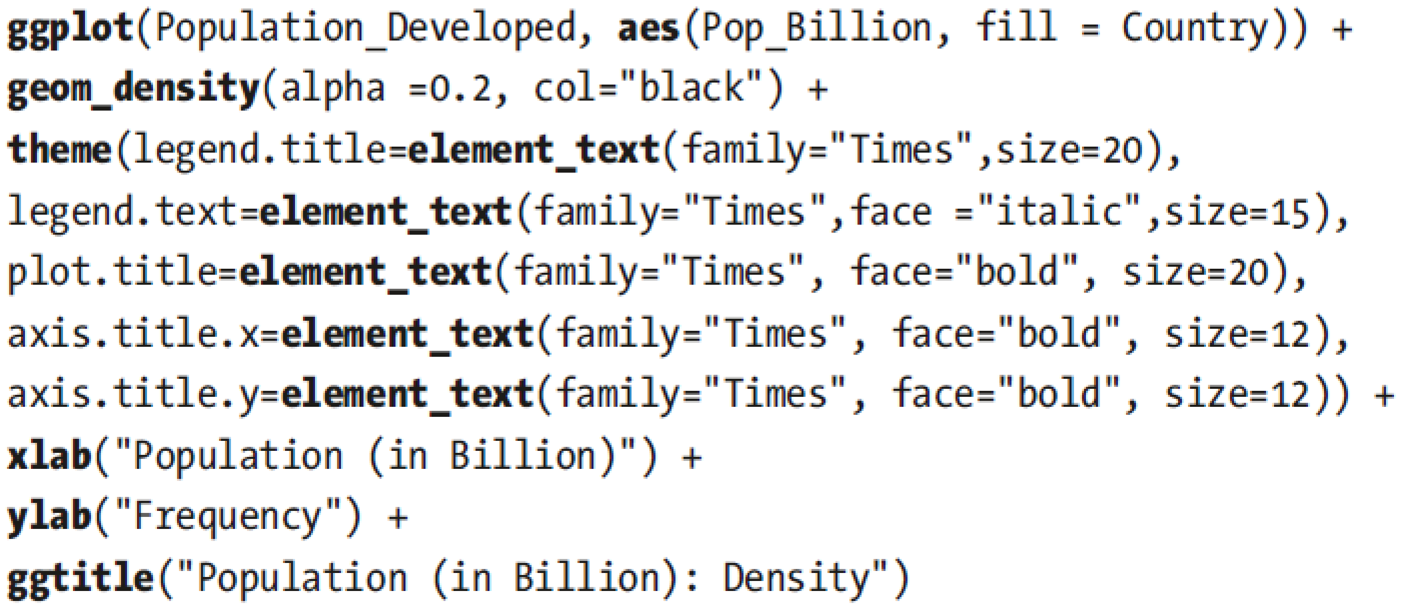
Histograms:

install.packages("reshape"); library(reshape)



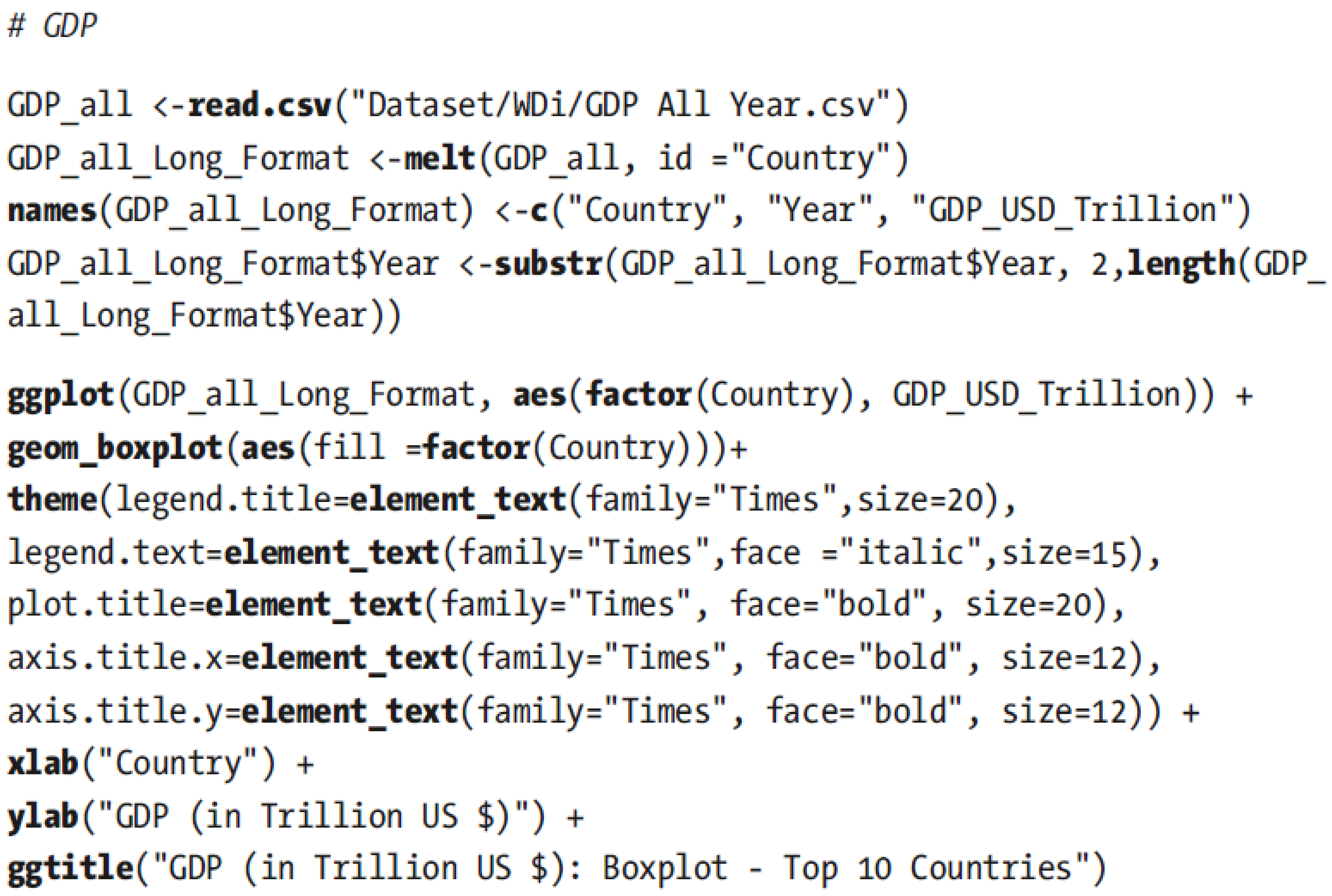
```
# Population

Population_all <-read.csv("Population All Year.csv")
Population_all_Long_Format <-melt(Population_all, id ="Country")
names(Population_all_Long_Format) <-c("Country", "Year", "Pop_Billion")
Population_all_Long_Format$Year <-substr(Population_all_Long_Format$Year,
2,length(Population_all_Long_Format$Year))

#Developed Country

Population_Developed <-Population_all_Long_Format[!(Population_all_Long_
Format$Country %in%c('India','China','Australia','Brazil','Canada','France',
'United States')),]

ggplot(Population_Developed, aes(Pop_Billion, fill = Country)) +
geom_histogram(alpha =0.5, aes(y = ..density..),col="black") +
theme(legend.title=element_text(family="Times",size=20),
legend.text=element_text(family="Times",face ="italic",size=15),
plot.title=element_text(family="Times", face="bold", size=20),
axis.title.x=element_text(family="Times", face="bold", size=12),
axis.title.y=element_text(family="Times", face="bold", size=12)) +
xlab("Population (in Billion)") +
ylab("Frequency") +
ggtitle("Population (in Billion): Histogram")
```

Density Plots



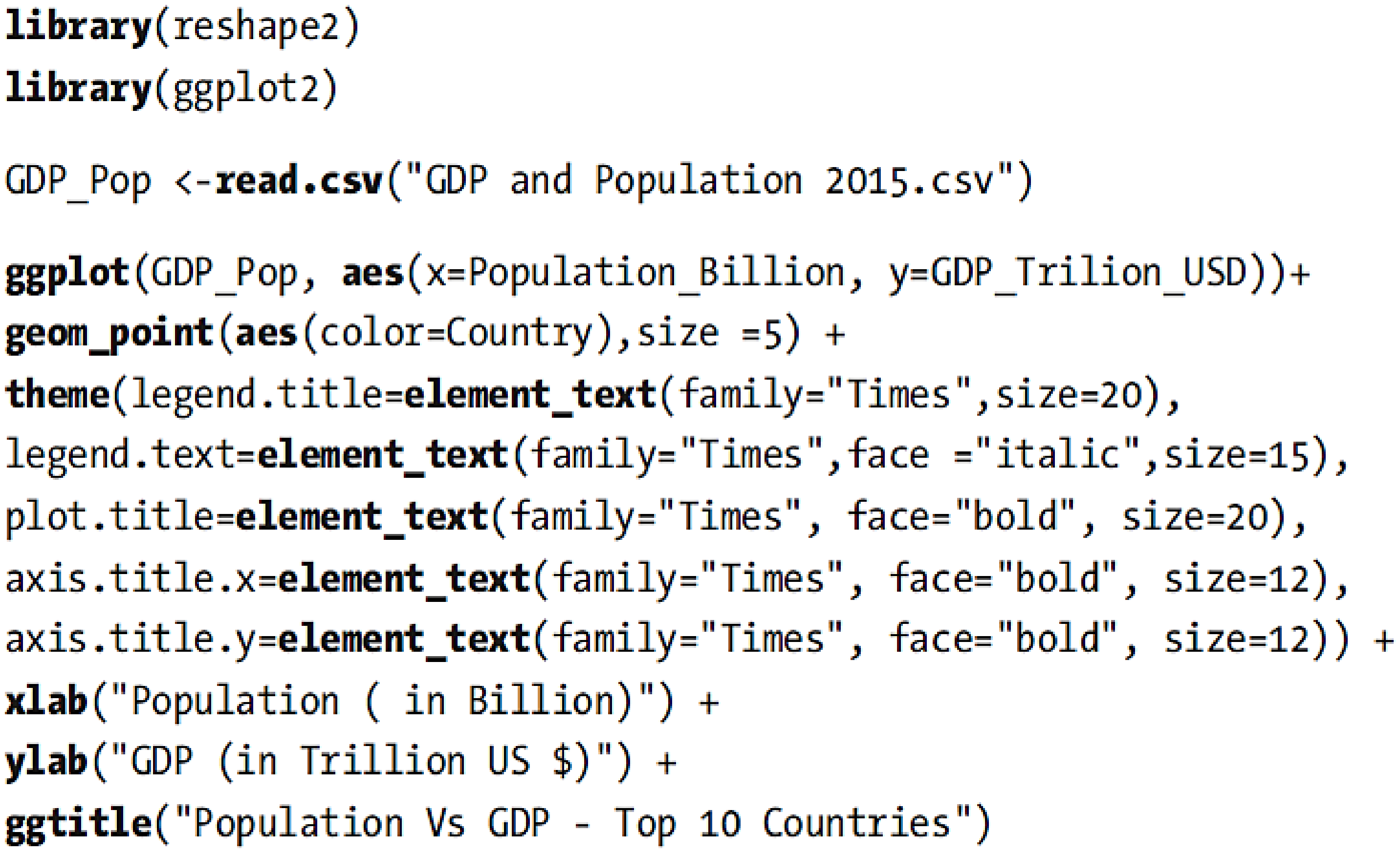
```
ggplot(Population_Developed, aes(Pop_Billion, fill = Country)) +
geom_density(alpha =0.2, col="black") +
theme(legend.title=element_text(family="Times",size=20),
legend.text=element_text(family="Times",face ="italic",size=15),
plot.title=element_text(family="Times", face="bold", size=20),
axis.title.x=element_text(family="Times", face="bold", size=12),
axis.title.y=element_text(family="Times", face="bold", size=12)) +
xlab("Population (in Billion)") +
ylab("Frequency") +
ggtitle("Population (in Billion): Density")
```

Boxplots



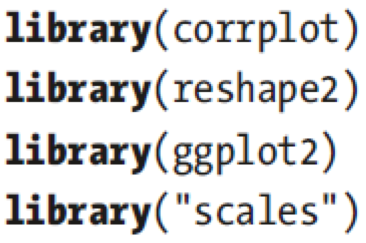
```
# GDP

GDP_all <-read.csv("Dataset/WDi/GDP All Year.csv")
GDP_all_Long_Format <-melt(GDP_all, id ="Country")
names(GDP_all_Long_Format) <-c("Country", "Year", "GDP_USD_Trillion")
GDP_all_Long_Format$Year <-substr(GDP_all_Long_Format$Year, 2,length(GDP_
all_Long_Format$Year))

ggplot(GDP_all_Long_Format, aes(factor(Country), GDP_USD_Trillion)) +
geom_boxplot(aes(fill =factor(Country)))+
theme(legend.title=element_text(family="Times",size=20),
legend.text=element_text(family="Times",face ="italic",size=15),
plot.title=element_text(family="Times", face="bold", size=20),
axis.title.x=element_text(family="Times", face="bold", size=12),
axis.title.y=element_text(family="Times", face="bold", size=12)) +
xlab("Country") +
ylab("GDP (in Trillion US $)") +
ggtitle("GDP (in Trillion US $): Boxplot - Top 10 Countries")
```

scatterplots

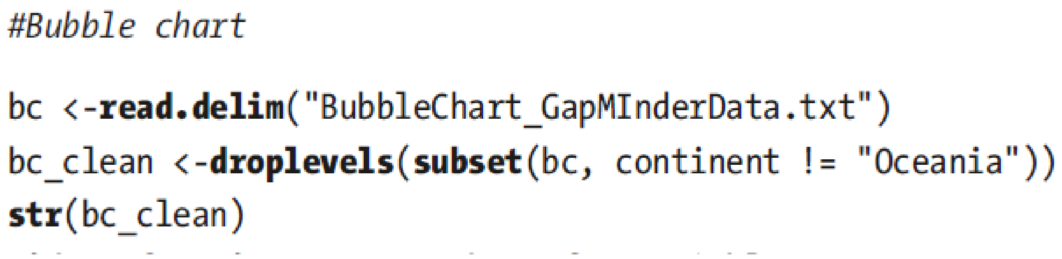


```
library(reshape2)
library(ggplot2)

GDP_Pop <-read.csv("GDP and Population 2015.csv")

ggplot(GDP_Pop, aes(x=Population_Billion, y=GDP_Trilion_USD))+
geom_point(aes(color=Country),size =5) +
theme(legend.title=element_text(family="Times",size=20),
legend.text=element_text(family="Times",face ="italic",size=15),
plot.title=element_text(family="Times", face="bold", size=20),
axis.title.x=element_text(family="Times", face="bold", size=12),
axis.title.y=element_text(family="Times", face="bold", size=12)) +
xlab("Population ( in Billion)") +
ylab("GDP (in Trillion US $)") +
ggtitle("Population Vs GDP - Top 10 Countries")
```

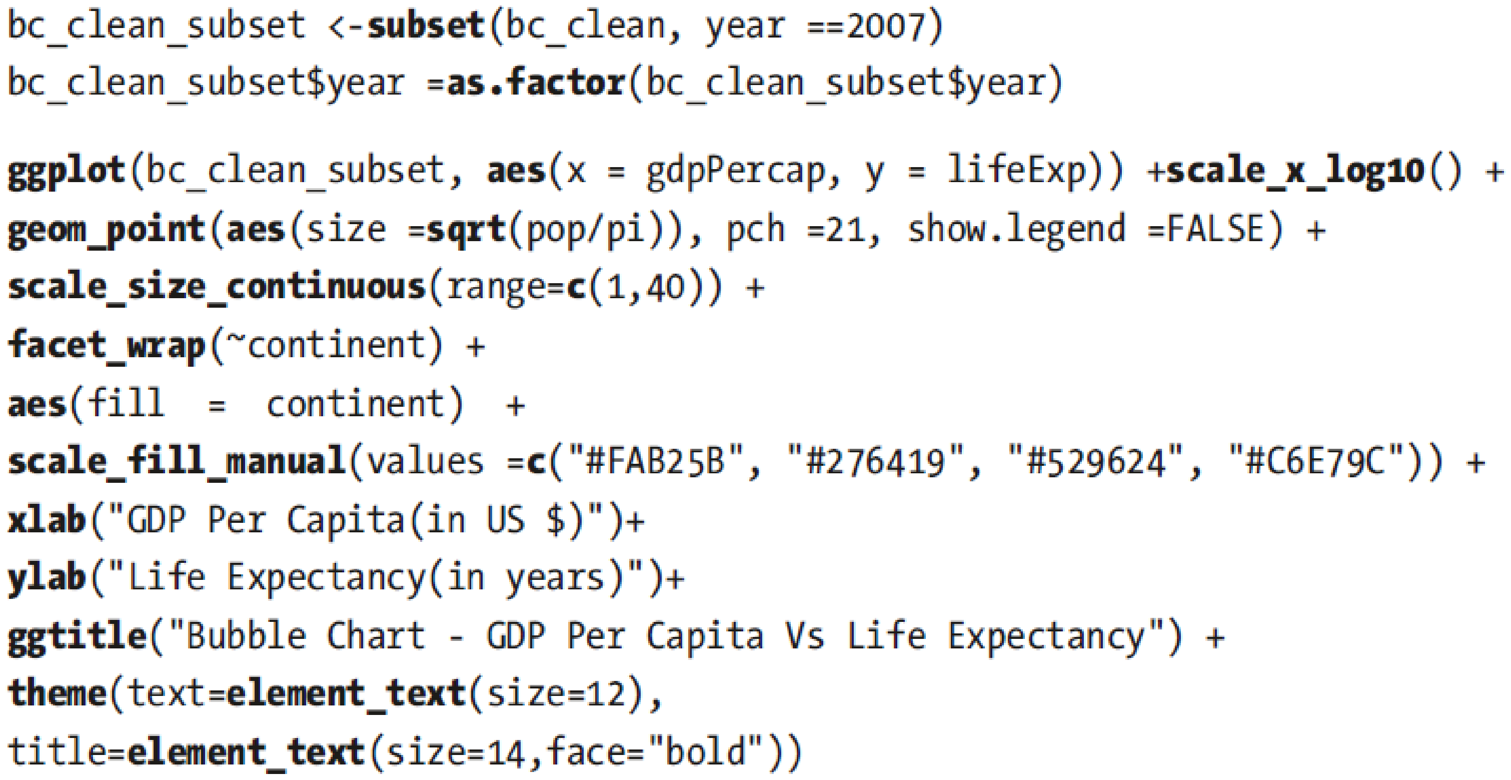
For jitters: geom_jitter(position=position_jitter(0.02)) + …



```
library(corrplot)
library(reshape2)
library(ggplot2)
library("scales")
```



```
#Bubble chart

bc <-read.delim("BubbleChart_GapMInderData.txt")
bc_clean <-droplevels(subset(bc, continent != "Oceania"))
str(bc_clean)
```

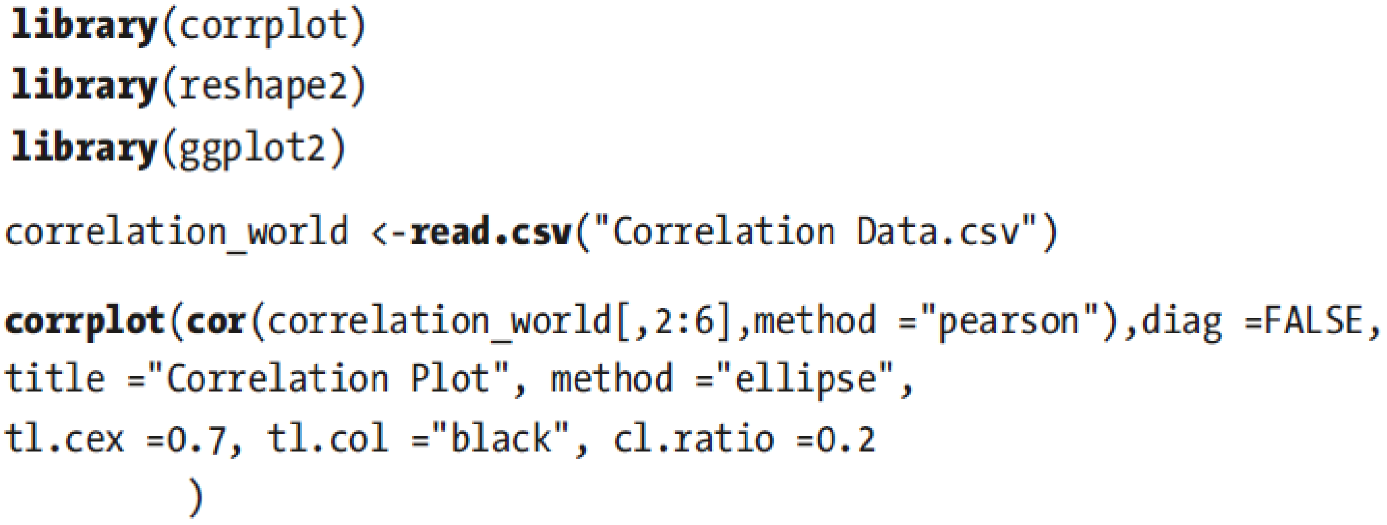


```
bc_clean_subset <-subset(bc_clean, year ==2007)
bc_clean_subset$year =as.factor(bc_clean_subset$year)

ggplot(bc_clean_subset, aes(x = gdpPercap, y = lifeExp)) +scale_x_log10() +
geom_point(aes(size =sqrt(pop/pi)), pch =21, show.legend =FALSE) +
scale_size_continuous(range=c(1,40)) +
facet_wrap(~continent) +
aes(fill  =  continent)  +
scale_fill_manual(values =c("#FAB25B", "#276419", "#529624", "#C6E79C")) +
xlab("GDP Per Capita(in US $)")+
ylab("Life Expectancy(in years)")+
ggtitle("Bubble Chart - GDP Per Capita Vs Life Expectancy") +
theme(text=element_text(size=12),
title=element_text(size=14,face="bold"))
```

Correlation Plots

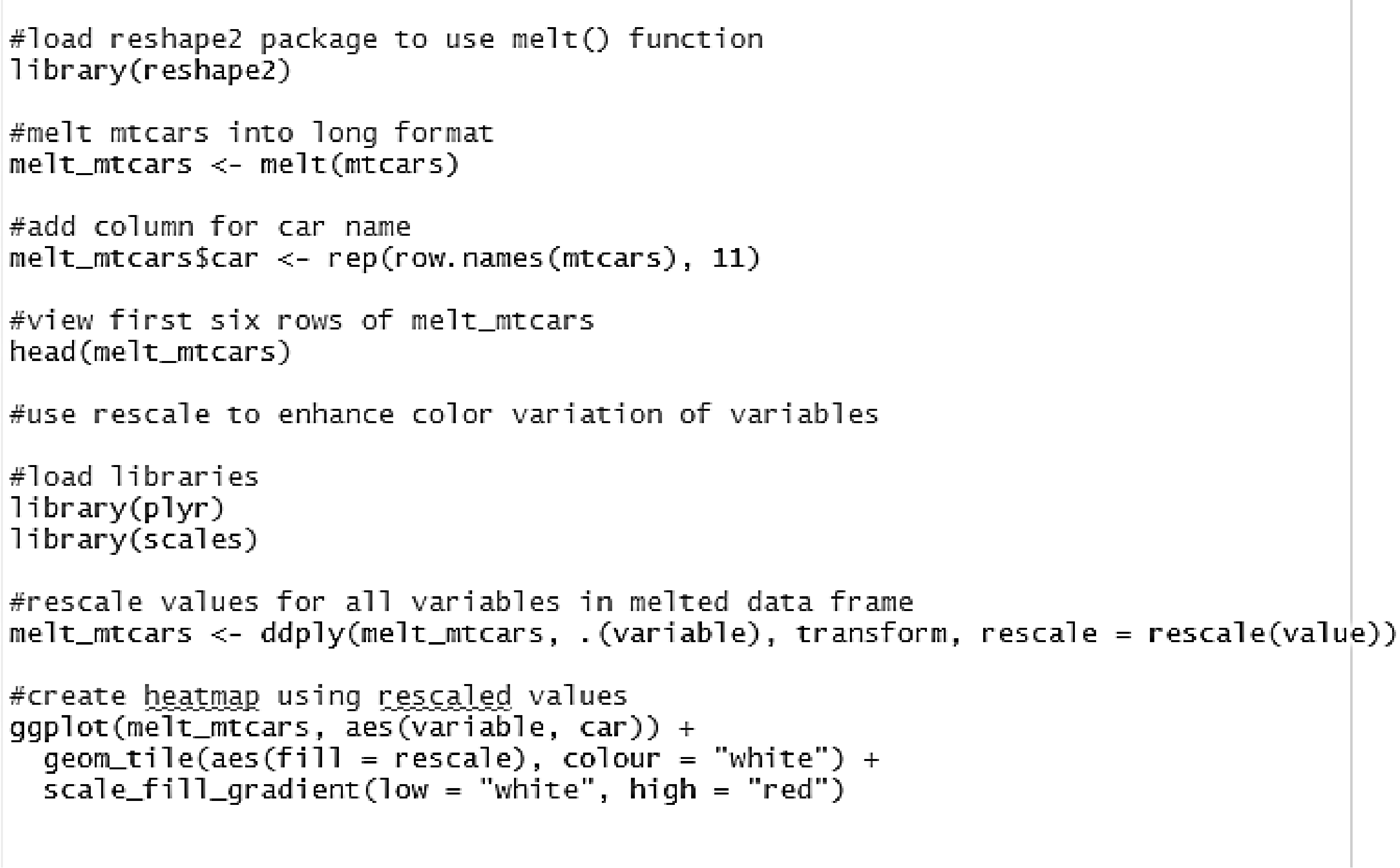
- *Corrplot()* is a R package that can be used for graphical display of a correlation matrix

install.packages("corrplot")



```
library(corrplot)
library(reshape2)
library(ggplot2)

correlation_world <-read.csv("Correlation Data.csv")

corrplot(cor(correlation_world[,2:6],method ="pearson"),diag =FALSE,
title ="Correlation Plot", method ="ellipse",
tl.cex =0.7, tl.col ="black", cl.ratio =0.2
         )
```

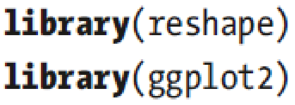
Heat Maps



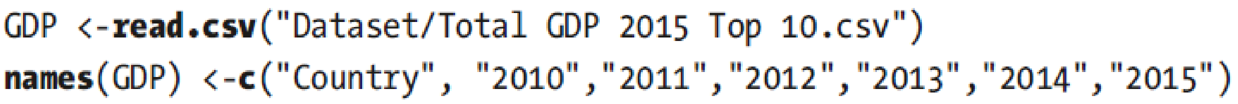
```
#load reshape2 package to use melt() function
library(reshape2)

#melt mtcars into long format
melt_mtcars <- melt(mtcars)

#add column for car name
melt_mtcars$car <- rep(row.names(mtcars), 11)

#view first six rows of melt_mtcars
head(melt_mtcars)

#use rescale to enhance color variation of variables

#load libraries
library(plyr)
library(scales)

#rescale values for all variables in melted data frame
melt_mtcars <- ddply(melt_mtcars, .(variable), transform, rescale = rescale(value))

#create heatmap using rescaled values
ggplot(melt_mtcars, aes(variable, car)) +
  geom_tile(aes(fill = rescale), colour = "white") +
  scale_fill_gradient(low = "white", high = "red")
```

```r
library(plyr)

World_Comp_GDP <-read.csv("World GDP and Sector.csv")

World_Comp_GDP_Long_Format <-melt(World_Comp_GDP, id ="Sector")
names(World_Comp_GDP_Long_Format) <-c("Sector", "Year", "USD")

World_Comp_GDP_Long_Format$Year <-substr(World_Comp_GDP_Long_Format$Year,
2,length(World_Comp_GDP_Long_Format$Year))

# calculate midpoints of bars

World_Comp_GDP_Long_Format_Label <-ddply(World_Comp_GDP_Long_Format, .(Year),
   transform, pos =cumsum(USD) -(0.5 *USD))

ggplot(World_Comp_GDP_Long_Format_Label, aes(x = Year, y = USD, fill =
Sector)) +
geom_bar(stat ="identity") +
geom_text(aes(label = USD, y = pos), size =3) +
theme(legend.title=element_text(family="Times",size=20),
legend.text=element_text(family="Times",face ="italic",size=15),
plot.title=element_text(family="Times", face="bold", size=20),
axis.title.x=element_text(family="Times", face="bold", size=12),
axis.title.y=element_text(family="Times", face="bold", size=12)) +
xlab("Year") +
ylab("% of GDP") +
ggtitle("Contribution of various sector in the World GDP")
```

Line charts

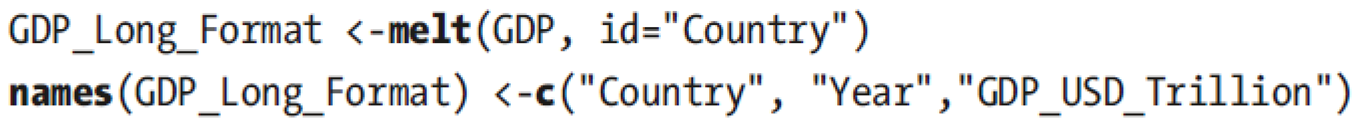


```
library(reshape)
library(ggplot2)
```

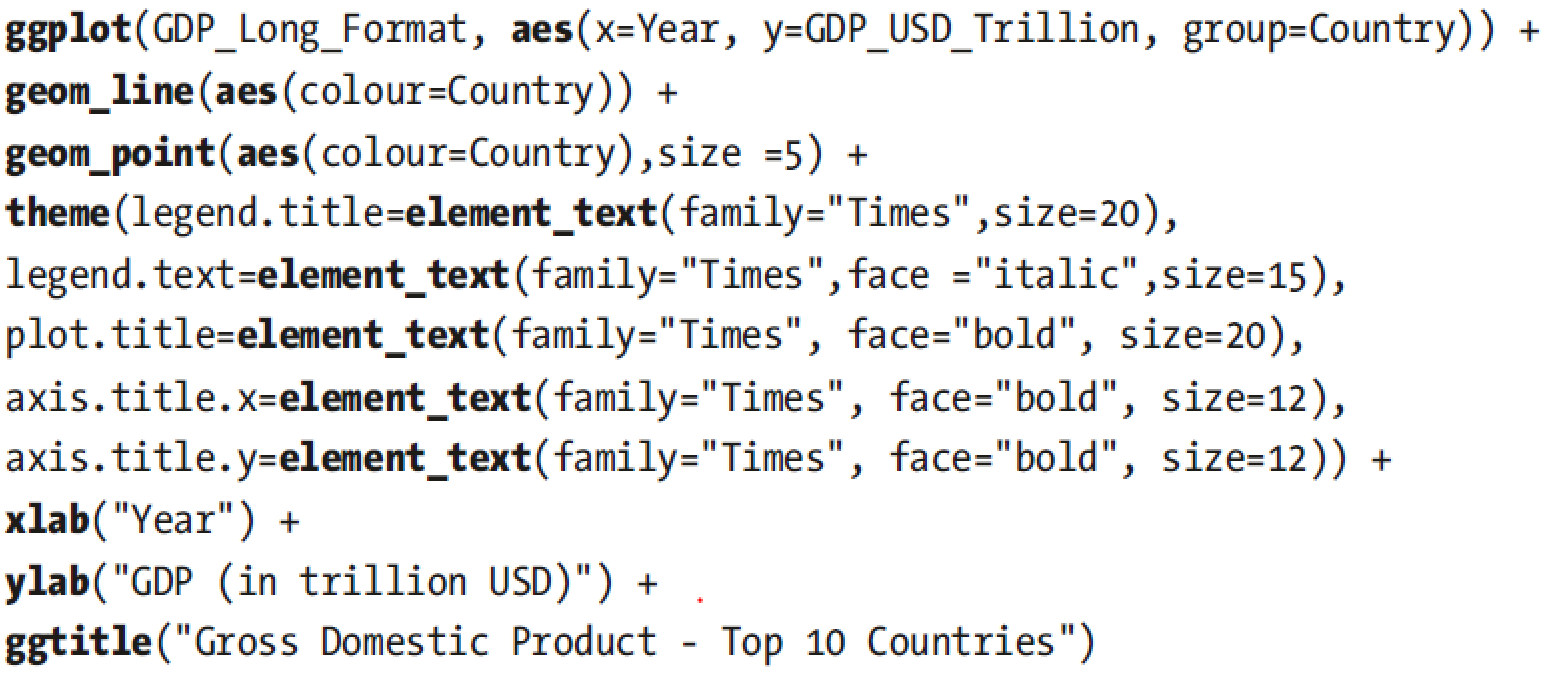


```
GDP <-read.csv("Dataset/Total GDP 2015 Top 10.csv")
names(GDP) <-c("Country", "2010","2011","2012","2013","2014","2015")
```

Melt() function available in reshape package takes data in wide formats & stacks a set of columns into a single column of data.

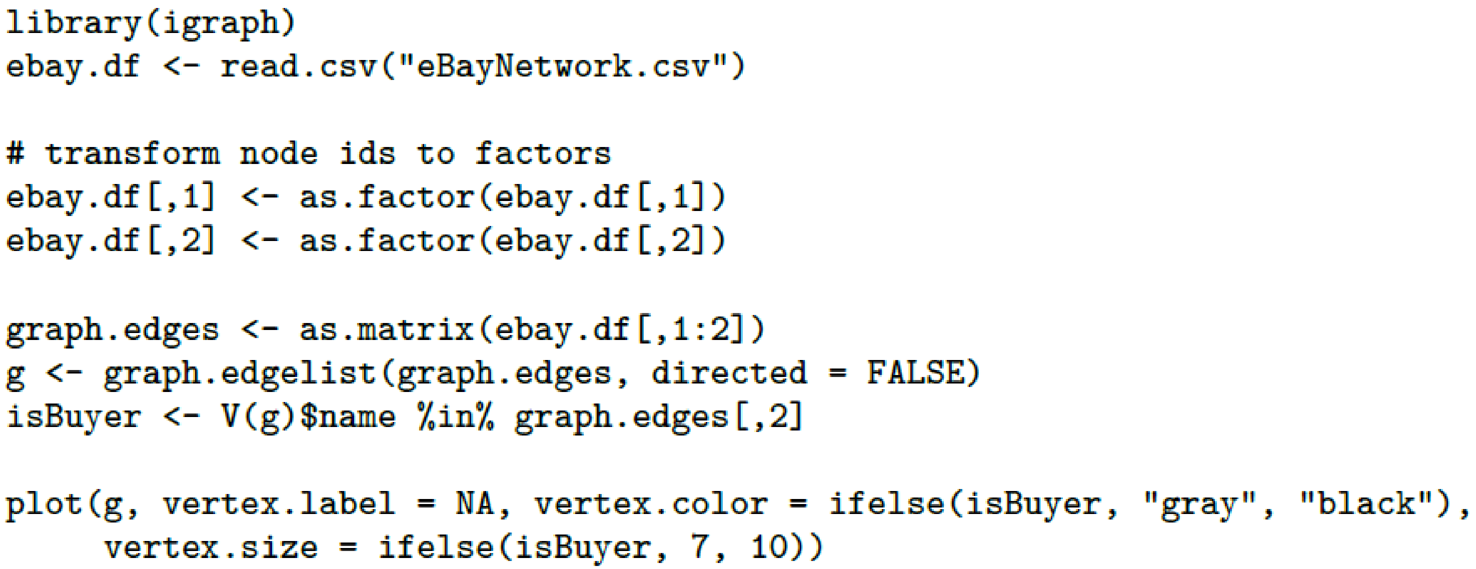


```
GDP_Long_Format <-melt(GDP, id="Country")
names(GDP_Long_Format) <-c("Country", "Year","GDP_USD_Trillion")
```



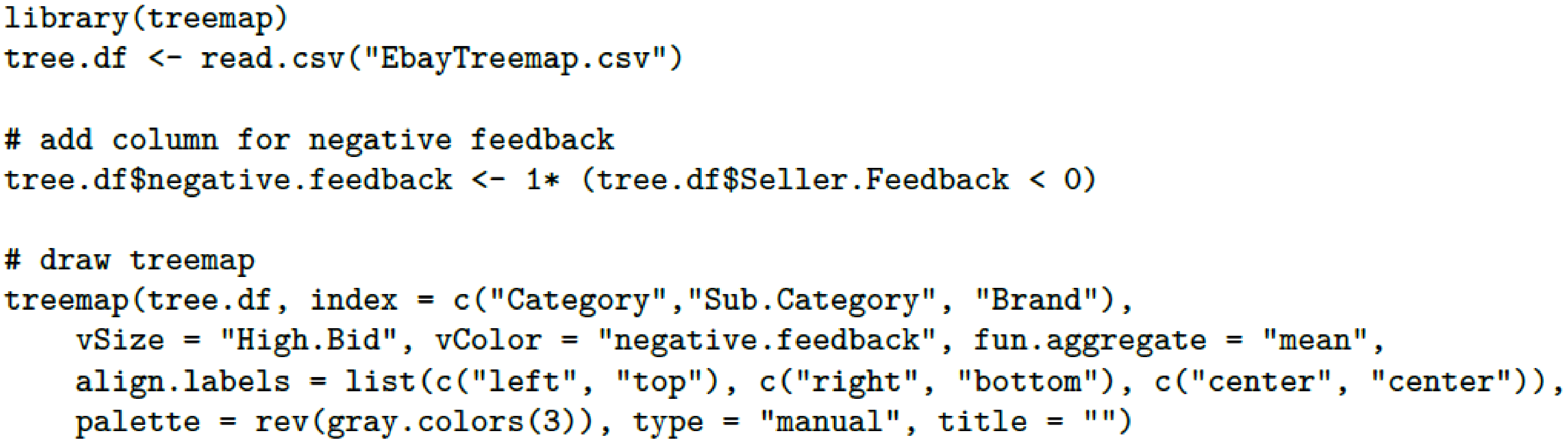
```
ggplot(GDP_Long_Format, aes(x=Year, y=GDP_USD_Trillion, group=Country)) +
geom_line(aes(colour=Country)) +
geom_point(aes(colour=Country),size =5) +
theme(legend.title=element_text(family="Times",size=20),
legend.text=element_text(family="Times",face ="italic",size=15),
plot.title=element_text(family="Times", face="bold", size=20),
axis.title.x=element_text(family="Times", face="bold", size=12),
axis.title.y=element_text(family="Times", face="bold", size=12)) +
xlab("Year") +
ylab("GDP (in trillion USD)") +
ggtitle("Gross Domestic Product - Top 10 Countries")
```

Networked Data



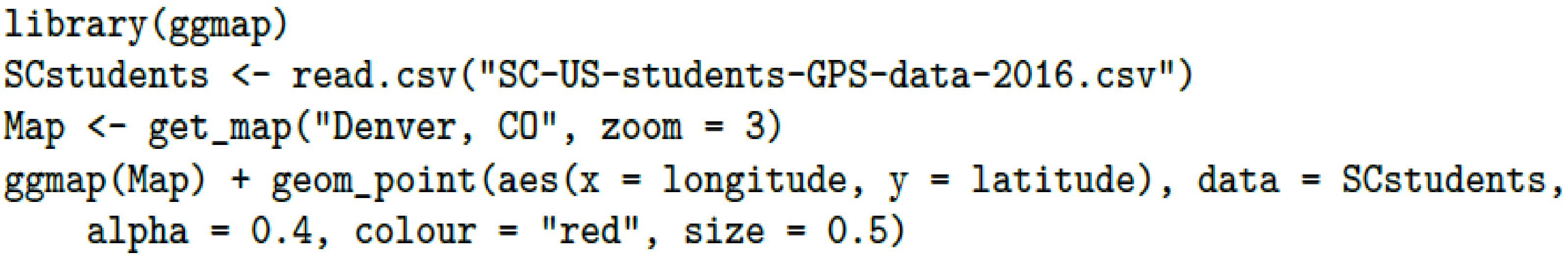
```
library(igraph)
ebay.df <- read.csv("eBayNetwork.csv")

# transform node ids to factors
ebay.df[,1] <- as.factor(ebay.df[,1])
ebay.df[,2] <- as.factor(ebay.df[,2])

graph.edges <- as.matrix(ebay.df[,1:2])
g <- graph.edgelist(graph.edges, directed = FALSE)
isBuyer <- V(g)$name %in% graph.edges[,2]

plot(g, vertex.label = NA, vertex.color = ifelse(isBuyer, "gray", "black"),
     vertex.size = ifelse(isBuyer, 7, 10))
```

Hierarchical Data: Tree Maps



```
library(treemap)
tree.df <- read.csv("EbayTreemap.csv")

# add column for negative feedback
tree.df$negative.feedback <- 1* (tree.df$Seller.Feedback < 0)

# draw treemap
treemap(tree.df, index = c("Category","Sub.Category", "Brand"),
    vSize = "High.Bid", vColor = "negative.feedback", fun.aggregate = "mean",
    align.labels = list(c("left", "top"), c("right", "bottom"), c("center", "center")),
    palette = rev(gray.colors(3)), type = "manual", title = "")
```

Geographical Data: Spatial Maps



```
library(ggmap)
SCstudents <- read.csv("SC-US-students-GPS-data-2016.csv")
Map <- get_map("Denver, CO", zoom = 3)
ggmap(Map) + geom_point(aes(x = longitude, y = latitude), data = SCstudents,
    alpha = 0.4, colour = "red", size = 0.5)
```