

Big Data Analytics



Tutorial for Data Mining

Version 3; April 2016

Dr Anil K. Maheshwari

**Table of Contents**

[1 About this Tutorial 3](#_Toc447817980)

[2 Getting Started 3](#_Toc447817981)

[3 Installation 4](#_Toc447817982)

[4 Working on R 7](#_Toc447817983)

[4.1 Get and set working directory 7](#_Toc447817984)

[4.2 List of commands 7](#_Toc447817985)

[4.3 Import a Dataset in R 8](#_Toc447817986)

[4.4 Data visualization 9](#_Toc447817987)

[4.4.1 Ploting a Histogram 9](#_Toc447817988)

[4.4.2 Ploting a Bar Chart 11](#_Toc447817989)

[4.4.3 Ploting charts side by side 12](#_Toc447817990)

[5 Data Mining Techniques 13](#_Toc447817991)

[5.1 Decision Tree (Supervised Learning) 13](#_Toc447817992)

[5.2 Regression (Supervised Learning) 15](#_Toc447817993)

[5.3 Correlation (Supervised Learning) 16](#_Toc447817994)

[5.4 Clustering – Kmeans (Unsupervised Learning) 17](#_Toc447817995)

[5.4.1 Method 1 17](#_Toc447817996)

[5.4.2 Method 2 17](#_Toc447817997)

[6 Big Data Mining 20](#_Toc447817998)

[6.1 WordCloud 20](#_Toc447817999)

[6.2 Twitter Mining 22](#_Toc447818000)

[6.2.1 Steps (Twitter side) 22](#_Toc447818001)

[6.2.2 R Script 23](#_Toc447818002)

[6.3 Page Rank 25](#_Toc447818003)

[7 Additional Documentation 26](#_Toc447818004)

# About this Tutorial

This tutorial was created for use by MSCS Compro students during the course Big Data Analytics. It offers an introduction to R programming script language, what it’s supposed to be a base to learn about Data Mining during this course. I thank my students Tonmay (in 2014), Claire (in 2015), and Mauro (in 2016) for helping enhancing this document.

Where you text/commands with the line below, you can just copy from this document and paste in the R console and type ENTER:

getwd()

If some error occurs when you copy and paste the line codes above, try remove and re-type the double quotes (“) in the console again.

As a recommendation, you can try copy the commands to Notepad (or equivalent) before paste into R Console or R Studio.

# Getting Started

As the R Project website describes, “*R is a free software environment for statistical computing and graphics. It compiles and runs on a wide variety of UNIX platforms, Windows and MacOS.*”.

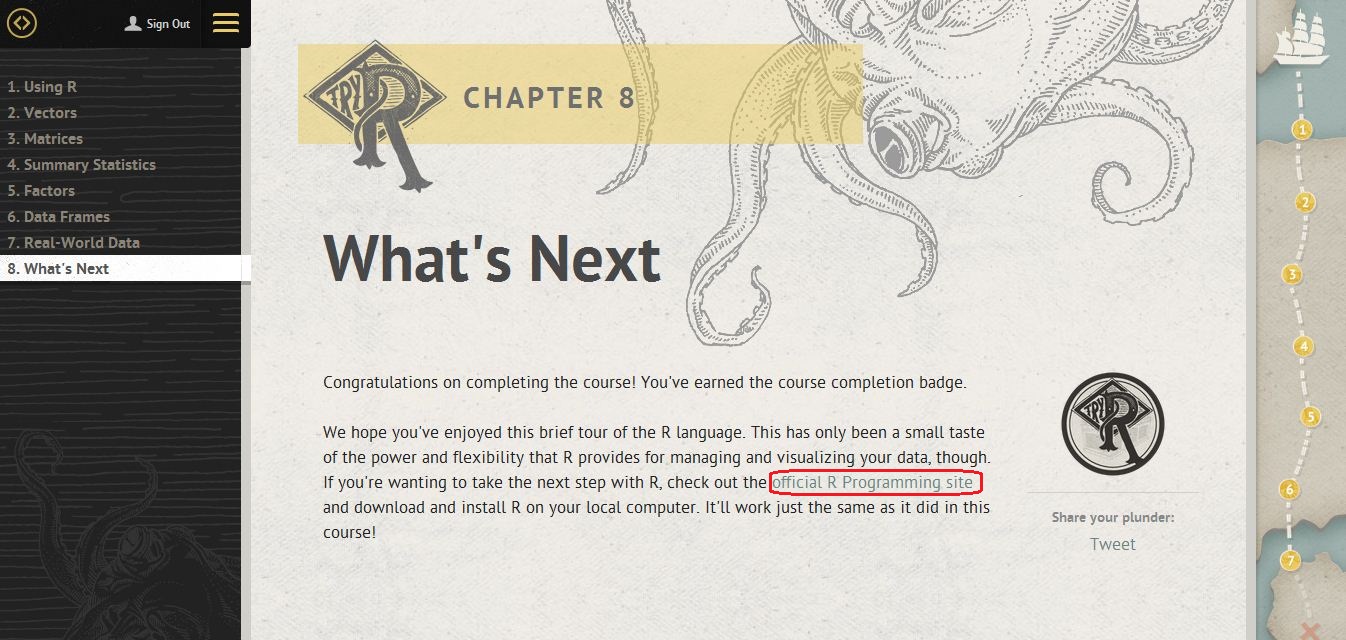
The first step to understand how R works, type at Google “code R” and go to the **R code school website**. You can directly go to <http://tryr.codeschool.com> too.

Sign up/register providing the simple information.

Follow the simple instruction and practice on the given code window.

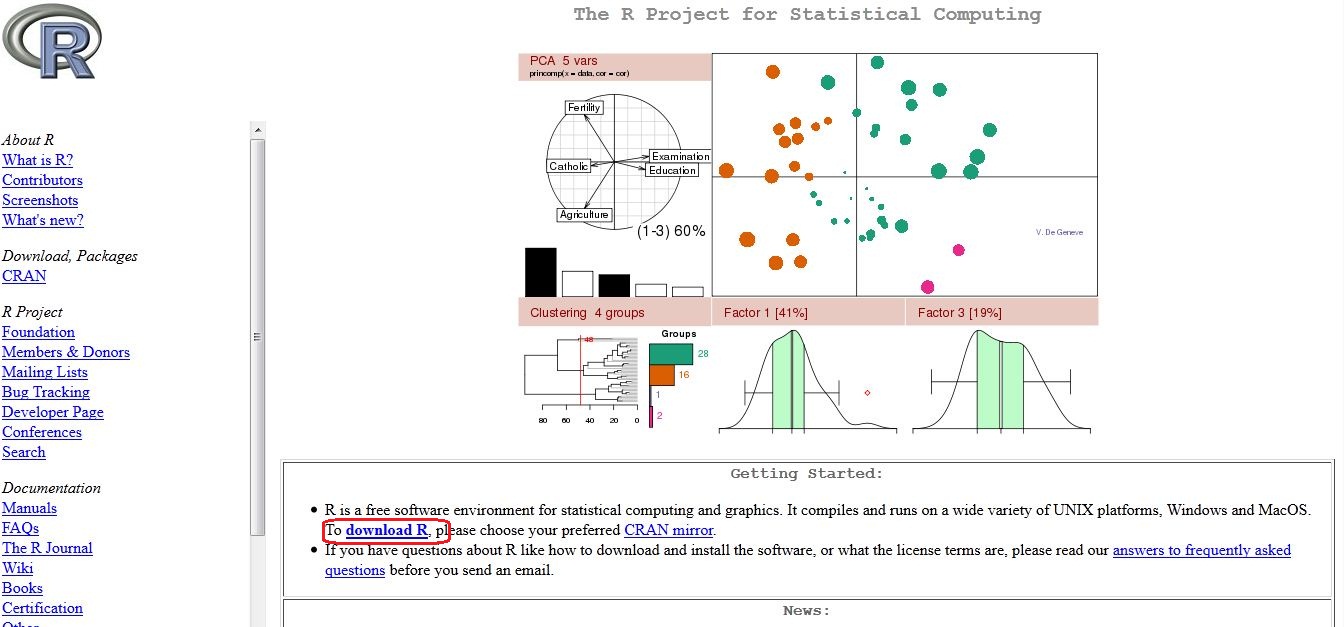
Finish the step and unlock the next steps.

Finish all seven steps and you’ll see a congratulation page like bellow.

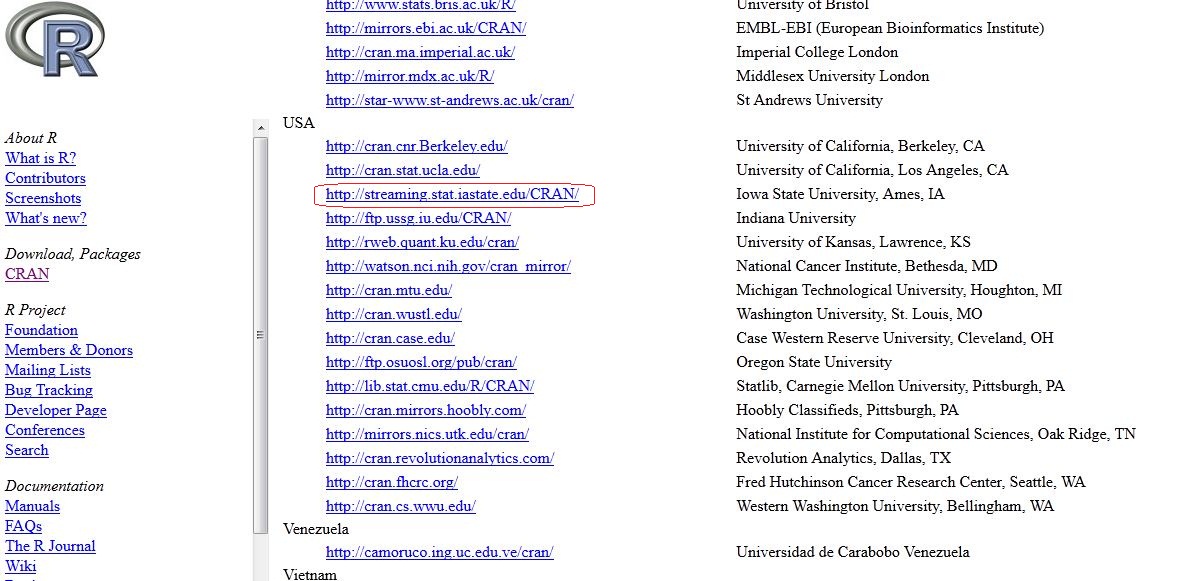


# Installation

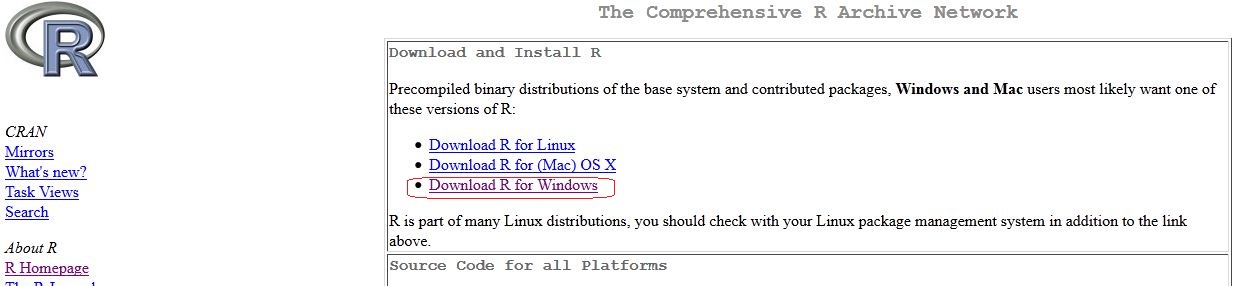
Click on the official R programming site ([http://www.r-project.org](http://www.r-project.org/)). Search for the Download section



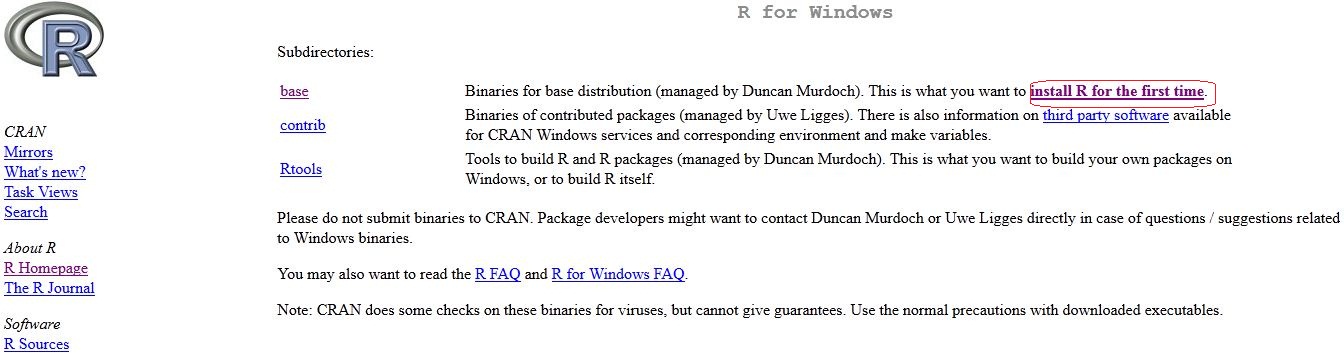
Click “download R” to get the proper mirror. This should take you to a page something like bellow.



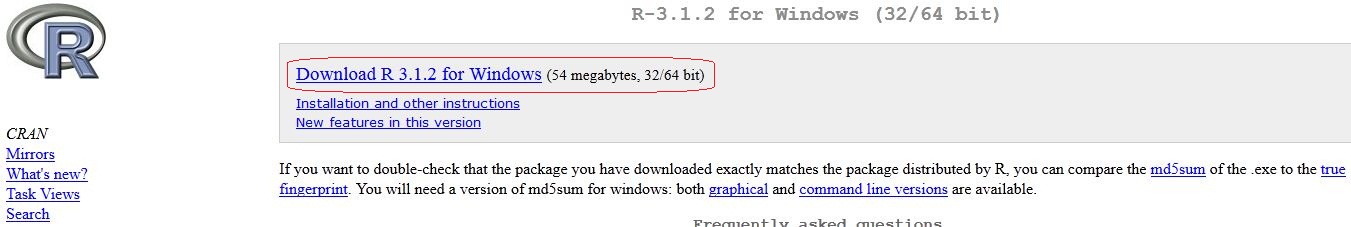
Choose the link for Iowa State University or any other mirror you like.



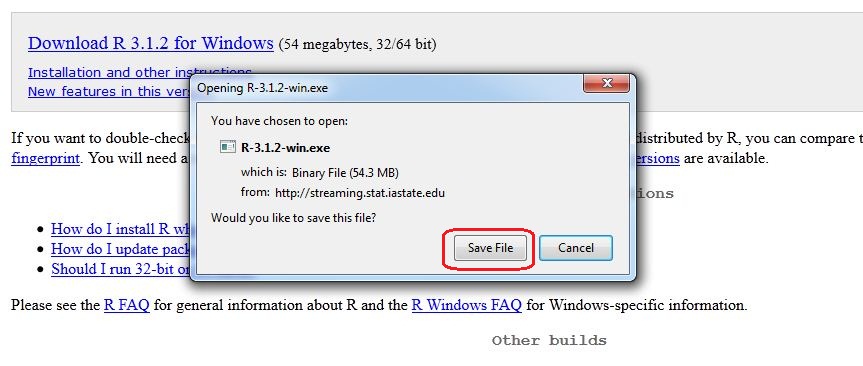
Choose your operating system. In the tutorial case it was selected Microsoft Windows.



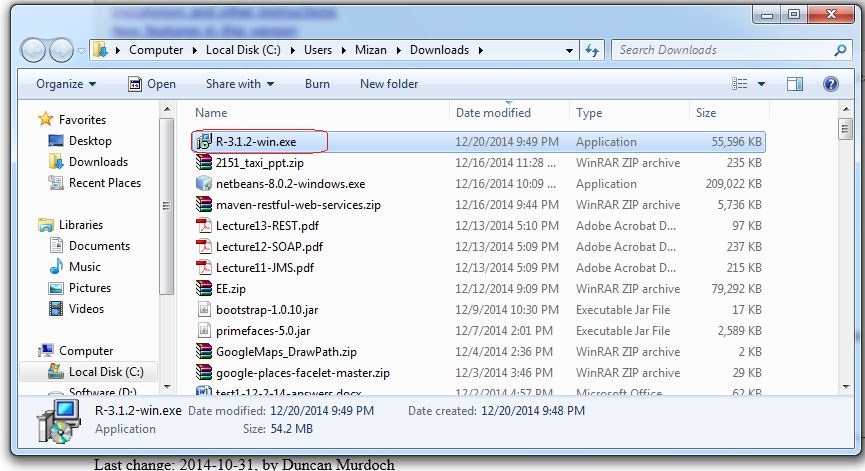
Click “install R for the first time”.



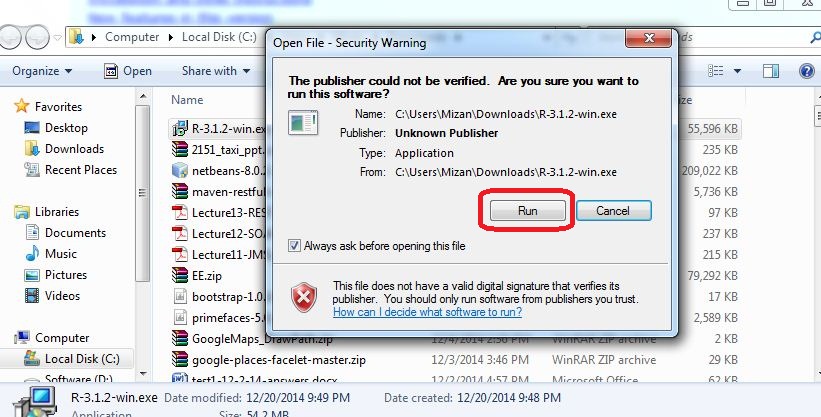
Click on “download” to download the installer (for Windows ).



Click on “Save file” to save the installer to your computer.



Double click the .exe file for installation.



Click “Run” to start the installation. Follow the steps. Click next, accept agreement, select your installation folder and finish the installation.

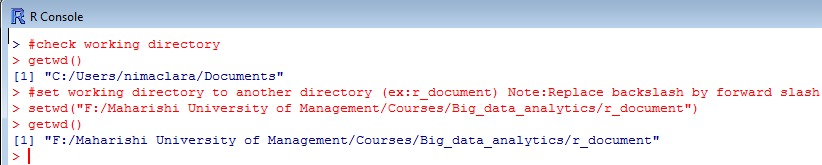
# Working on R

Select the R application from your start menu. All coding style should be same what you practiced on R code school.

## Get and set working directory

getwd()

setwd("/path/to/your/working/directory")



## List of commands

Here are some commands that you can use in your R script.

|  |  |  |
| --- | --- | --- |
| Command/word | Description | Example |
| # | The characters after are considered comment | #this is a comment |
| cat | Print a message to the console (like System.out.println in Java) | cat("Hello World!\n") |
| source | Execute a R script file previously created | source("myScript.r") |
| <- | Assign a value to a variable | one <- 1  Hello <- "Hello" |
| install.packages | Download and install a package | install.packages("rpart") |
| library | Load a installed library to your R script | load(rpart) |

**OBS**: R is case sensitive, so **Variable** is different from **variable**.

## Import a Dataset in R

Before importing your dataset you must convert your .xls or .xlsx file to .csv file and save it in the working directory. Otherwise you will be required to specify its path when you are importing the data in R.

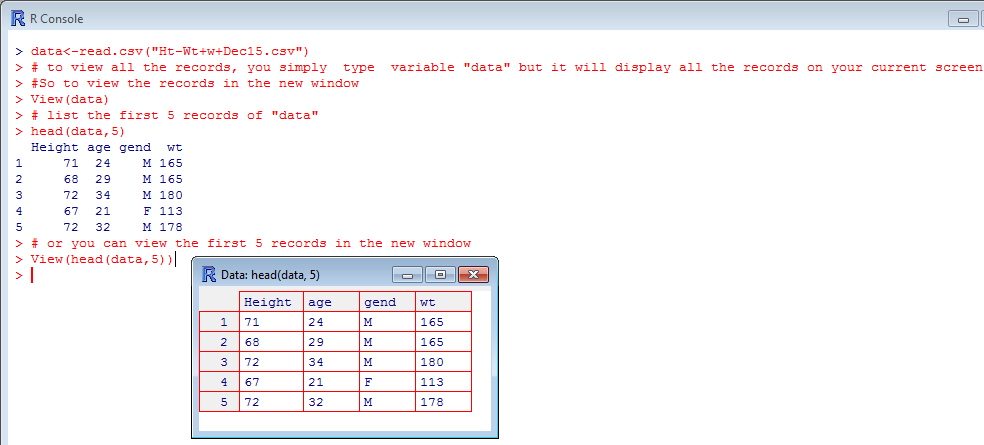
Load the data to a variable using read.csv command. For example, for the screen below we used the following command:

data <- read.csv("Ht-Wt+w+Dec15.csv") #Read data from the csv file and assign to variable “data”

View(data) #show all values from the “data” variable

head(data,5) #print in the console the first 5 rows from “data” variable

View(head(data,5)) #show the data loaded into “data” variable (5 first records)

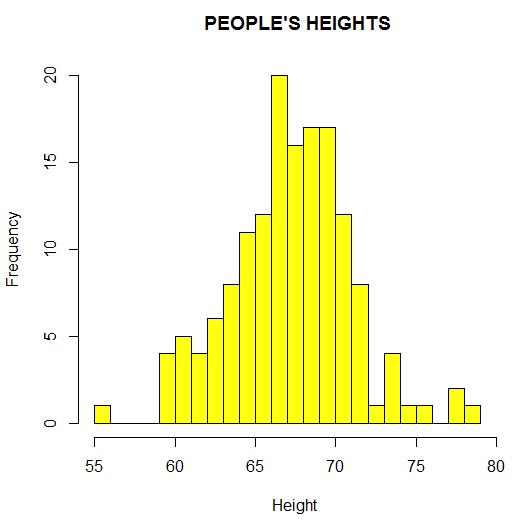


## Data visualization

### Ploting a Histogram

df <- read.csv("Ht-Wt+w+Dec15.csv") #Read data from the csv file and assign to variable “df”

hist(df$Height,ylim(c(0,20)),xlim=c(55,80),main= "PEOPLE’S HEIGHTS", breaks=32,xlab="Heigth", col="#FFFF0F",border="#000000") #plot the histogram height

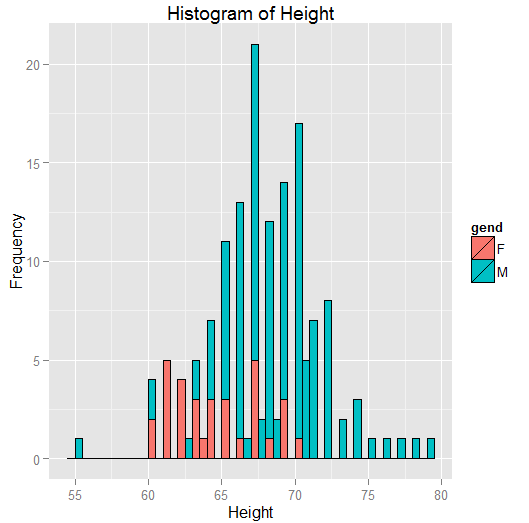


hist(df$wt,ylim(c(0,20)),xlim=c(99,240),main= "PEOPLE’S WEIGHTS", breaks=32,xlab="weight", col="#999999",border="#ffffff") #plot the histogram for weight

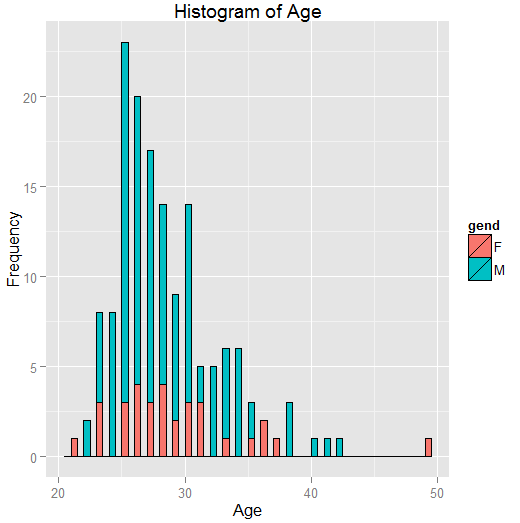


library(ggplot2) #load the library ggplot2, needed for ggplot command. If not installed, try install.packages("ggplot2")

ggplot(df,aes(x=Height,fill=gend))+geom\_bar(position="stack",binwidth=0.5,col="black")+scale\_x\_continuous("Height")+scale\_y\_continuous("Frequency")+ggtitle("Histogram of Height")



ggplot(df,aes(x=age,fill=gend))+geom\_bar(position="stack",binwidth=0.5,col="black")+scale\_x\_continuous("Age")+scale\_y\_continuous("Frequency")+ggtitle("Histogram of Age")



### Ploting a Bar Chart

barplot(table(df$gend),ylab="Frequency",main="GENDER OF PEOPLE",xlab="Gender",ylim(c(0,120)) #plot a bar chart for gender



### Ploting charts side by side

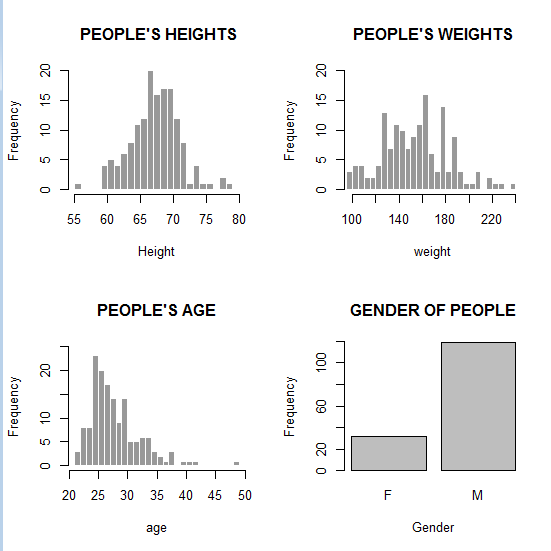
Par(mfrow=c(2,2))

hist(df$Height,ylim(c(0,20)),xlim=c(55,80),main= "PEOPLE’S HEIGHTS", breaks=32,xlab="Heigth", col="#FFFF0F",border="#000000") #plot the histogram height

hist(df$wt,ylim(c(0,20)),xlim=c(99,240),main= "PEOPLE’S WEIGHTS", breaks=32,xlab="weight", col="#999999",border="#ffffff") #plot the histogram for weight

hist(df$age,ylim(c(0,25)),xlim=c(21,49),main= "PEOPLE’S AGE", breaks=32,xlab="age", col="#999999",border="#ffffff") #plot the histogram for age

barplot(table(df$gend),ylab="Frequency",main="GENDER OF PEOPLE",xlab="Gender",ylim(c(0,120)) #plot a bar chart for gender



# Data Mining Techniques

## Decision Tree (Supervised Learning)

library("rpart") #load the library for decision tree

# Draw the tree and assign to any variable (in our case, "myTree”)

# Here Age, Job, House and Credit are column names and we are using decision tree to predict "LoanApproved” based on those columns, loanData is the variable name and

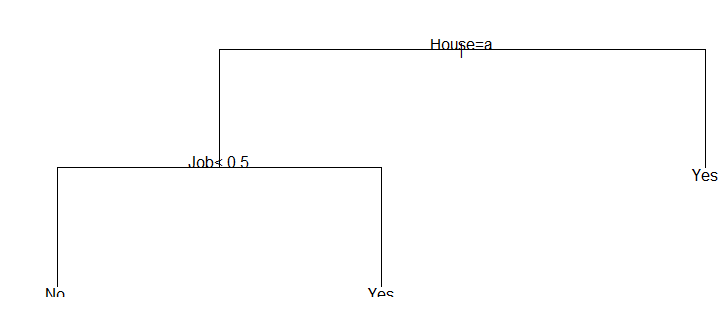
# method=class stands for classification.

loanData <- read.csv("Loan-Approval+dataset.csv”) # read dataset

loan\_tree = rpart(formula = LoanApproved ~ Age + Job + House + Credit, data = loanData, method = "class", control = rpart.control(minsplit = 5))

plot(loan\_tree) # plot the tree

text(loan\_tree) # Insert labels on tree



# To make the tree little bit fancy, install some extra packages

install.packages("rattle”)

install.packages("RColorBrewer") # Don’t forget that R is case sensitive

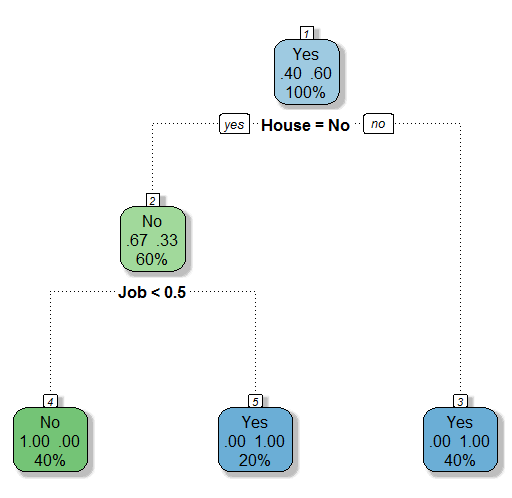
install.packages("rpart.plot”)

library(rattle) # Load rattle library

library(RColorBrewer)

library(rpart.plot)

fancyRpartPlot(loan\_tree) # We are using the same “loan\_tree” variable previously assigned



## Regression (Supervised Learning)

With R is possible work with multiple linear regression.

myData <- read.csv("Ht\_Wt\_wMarch16.csv")

head(myData,5)

Height age gend wt

1 71 24 M 165

2 68 29 M 165

3 72 34 M 180

4 67 21 F 113

5 72 32 M 178

x <- lm(myData$wt ~ myData$Height + myData$age + myData$gend) #Runs the correlation

summary(x)

Call:

lm(formula = myData$wt ~ myData$Height + myData$age + myData$gend)

Residuals:

Min 1Q Median 3Q Max

-46.405 -19.329 -1.992 16.141 65.594

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) -59.7711 37.7863 -1.582 0.115678

myData$Height 2.7103 0.5571 4.865 2.73e-06 \*\*\*

myData$age 0.5700 0.4251 1.341 0.181844

myData$gendM 19.9375 5.1898 3.842 0.000176 \*\*\*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 23.61 on 159 degrees of freedom

Multiple R-squared: 0.3024, Adjusted R-squared: 0.2893

F-statistic: 22.98 on 3 and 159 DF, p-value: 2.068e-12

coefficients(x)

(Intercept) myData$Height myData$age myData$gendM

-59.7710660 2.7102542 0.5700159 19.9374996

## Correlation (Supervised Learning)

In the same way described in decision tree, you can install the necessary library and load the data.

Using cov(data) you can see relation.

Using pairs(data) you can see the regression.

library(rattle) # If not loaded previously

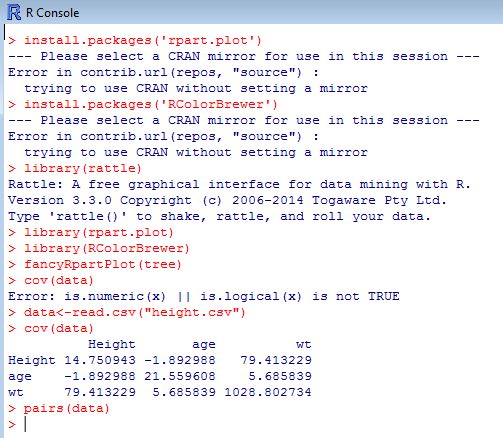
library(rpart.plot)

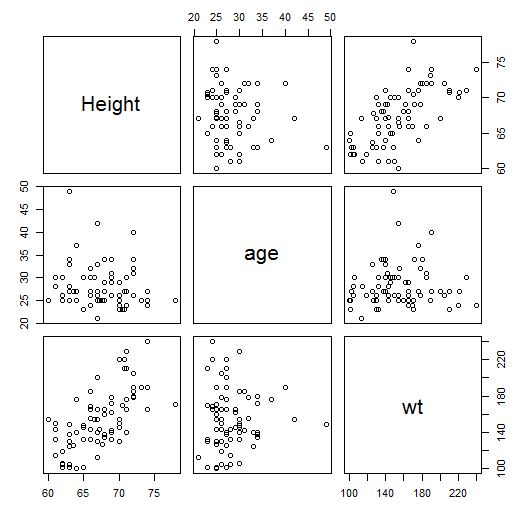
library(RColorBrewer)

data <- read.csv("height.csv")

cov(data)

pairs(data)





## Clustering – Kmeans (Unsupervised Learning)

There are different ways of plotting k-means clustering results

### Method 1

library(datasets)

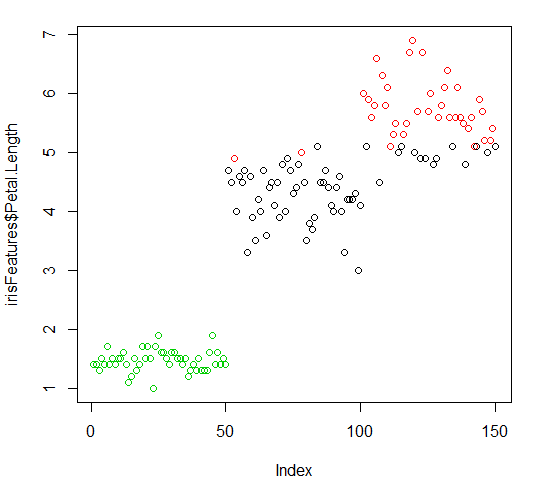
irisFeatures <- iris[-5]

km <- kmeans(irisFeatures,3)

km$size

km$cluster

plot(irisFeatures$Petal.Length, col = km$cluster)

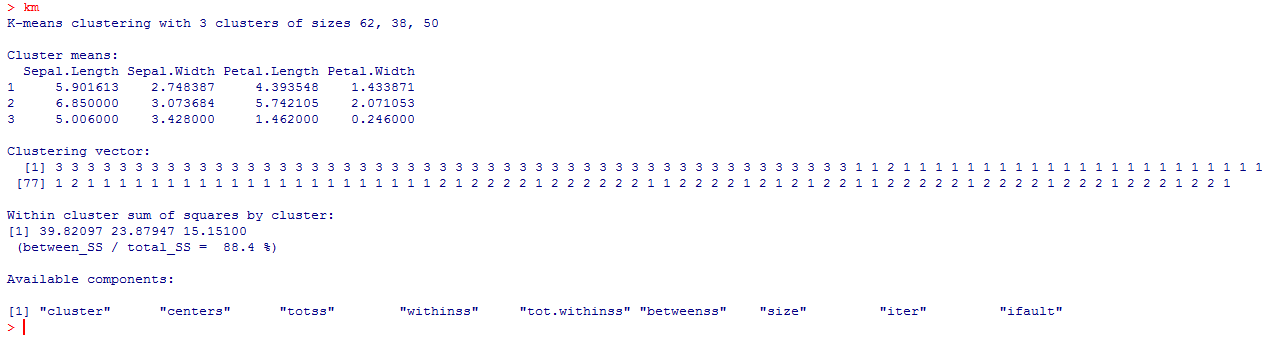


### Method 2

km # Show all the information about km

data(iris) # Using R Iris dataset

head(iris,3) # Just checking the first 3 rows of the dataset



# Iris dataset has 5 columns including 1 target column (species)

# We gonna use only 4 columns (i.e., exclude target column which is the 5th column)

myIris <- iris[,-5] # Assign the first 4 columns of iris dataset to “myIrisData” variable

myIris <- scale(myIrisData) # scaling dataset if they are in different range.

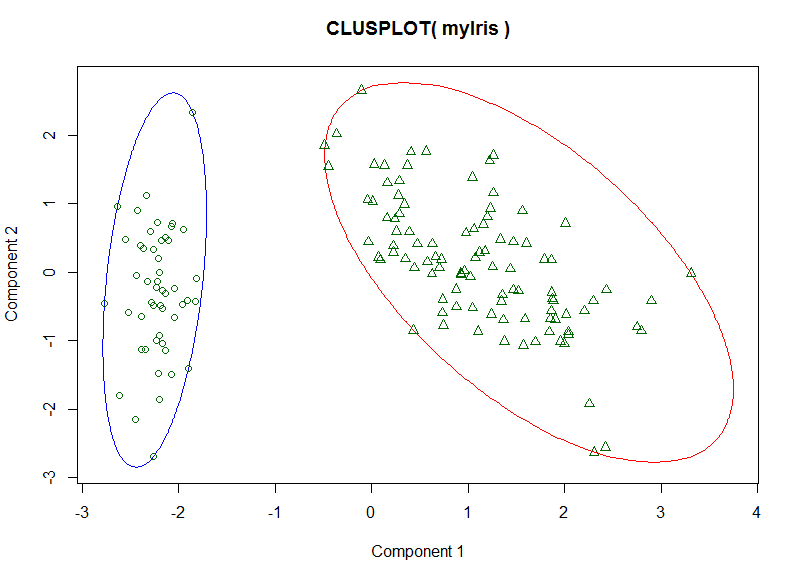
# Now apply k-means algorithm with k = 2 (k is the number of clusters)

irisKM <- kmeans(myIris,2)

library(fpc) # If not installed, try install.packages

library(cluster)

clusplot(myIris, irisKM$cluster, color = TRUE, lines = 0)

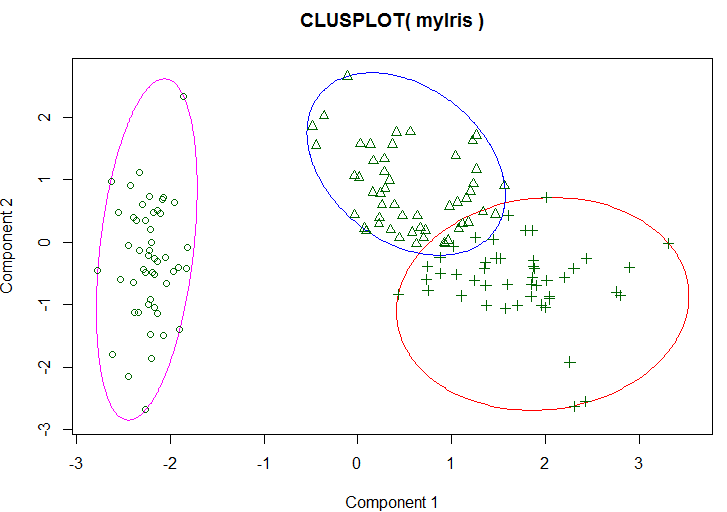


#Now, using different parameters to plot…

# k = 3 with 5 iterations

irisKM <- kmeans(myIris, 3, iter.max = 5)

clusplot(myIris,IrisKM$cluster, color = TRUE, lines = 0)



# Big Data Mining

## WordCloud

Using file contents

# Install the packages TM, wordcloud, RColorBrewer

# install.packages(c("tm", "wordcloud", "RColorBrewer"))

# Import the libraries

library(tm)

library(wordcloud)

library(RColorBrewer)

# select your file name below

#In my case I used obama speech from this link #http://www.cnn.com/2015/12/06/politics/transcript-obama-san-bernardino-isis-address/

articleCorpus <- readLines("obamaSpeech.txt")

# Read text as a vector

# Create corpus

mycorpus = Corpus(VectorSource(articleCorpus))

# Transformation

mycorpus1 = tm\_map(mycorpus, removePunctuation)

mycorpus2 = tm\_map(mycorpus1, removeWords, stopwords("english"))

mycorpus3 = tm\_map(mycorpus2, tolower)

mycorpus4 = tm\_map(mycorpus3, stripWhitespace)

mycorpus5 = tm\_map(mycorpus4, PlainTextDocument)

# Create Term document matrix. A document-term matrix or

# term-document matrix is a mathematical matrix that describes

# the frequency of terms that occur in a collection of documents.

tdm <- TermDocumentMatrix(mycorpus5)

m1 <- as.matrix(tdm)

# Sort data by freqency

v1<- sort(rowSums(m1), decreasing=TRUE)

# create dataframe with data and count/frequency

d1 <- data.frame(word=names(v1), freq=v1)

d1 <- head(d1,100)

# Create word cloud

wordcloud(d1$word,d1$freq,col=brewer.pal(8,"Set2"), min.freq="5",random.order=FALSE)



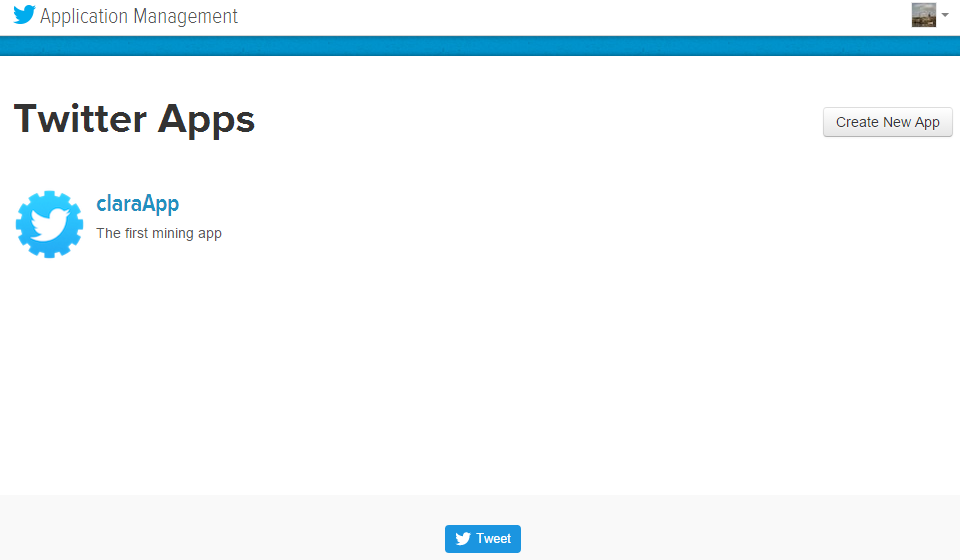
**OBS**: type ?wordcloud in the R Console to get documentation about worldcloud

## Twitter Mining

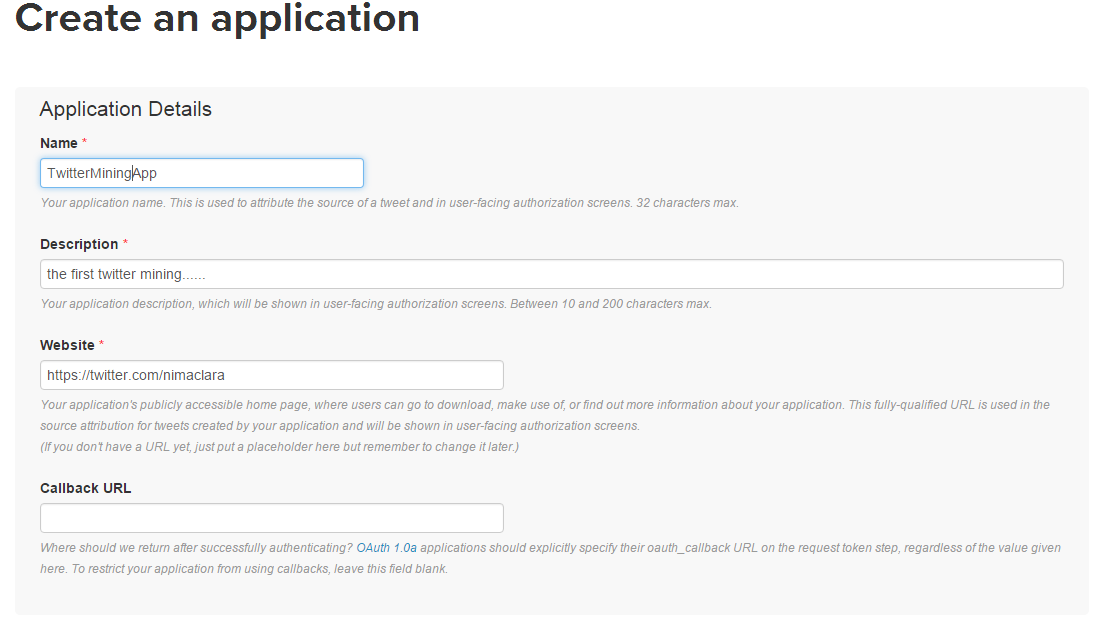
* + Using twitter hashTag mining

### Steps (Twitter side)

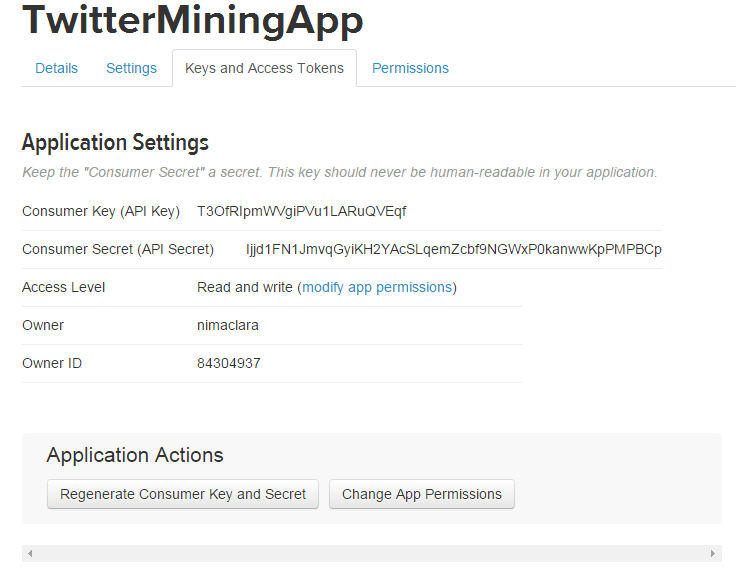
* + Sign in using your twitter account
  + <https://twitter.com/apps>



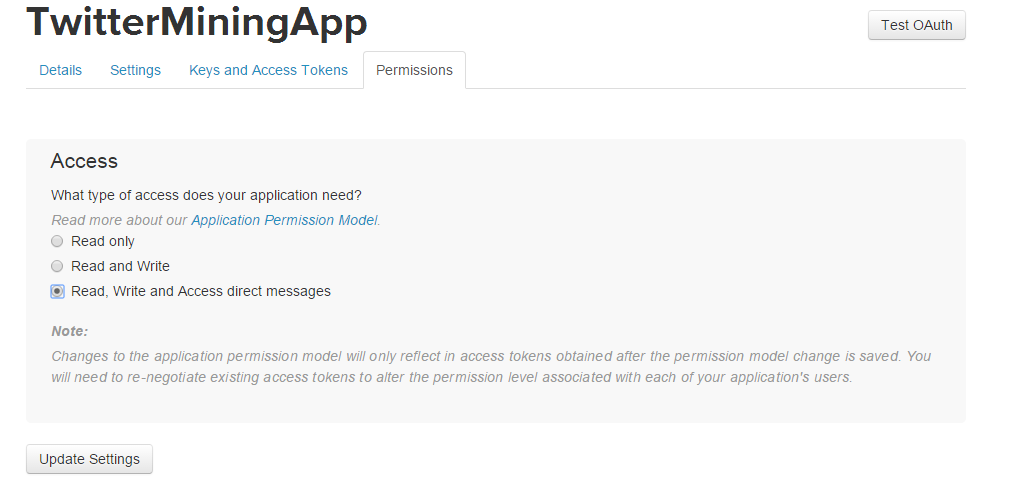
* + Click on create a New App



* + Under "keys and Access Tokens" tab, check the consumer key (API key), consumer secret (API secret), access Token and Access Token Secret.



* + Under "permissions" tab, authorize you application. By default, it is “read and write”. Select “Read, write and Access direct messages”. And then click update settings



### R Script

#twitter mining

install.packages("RCurl")

install.packages("twitteR")

install.packages("ROAuth")

install.packages("base64enc")

library(RCurl)

library(twitteR)

library(ROAuth)

library(base64enc)

# make sure to remove the blank spaces (if any) both at the end of your

# apiKey or apiSecret between the quotations

apiKey <- "T3OfRIpmWVgiPVu1LARuQVEqf"

apiSecret <- "Ijjd1FN1JmvqGyiKH2YAcSLqemZcbf9NGWxP0kanwwKpPMPBCp"

accessToken <- "84304937-Vi1CYqNRbjjsoS7ShrjSY7SZHXfmtIqvS4qdXT3w6"

accessSecret <- "HyAdI4yWEXe00Jk1xdjzCEcxIzCNrH1vmdnvkTuL1tzjP";

setup\_twitter\_oauth(apiKey,apiSecret,access\_token=accessToken,access\_secret=accessSecret)

#install.packages("tm")

#install.packages("wordcloud")

library(tm)

library(wordcloud)

library(RColorBrewer)

hillaryClinton <- searchTwitter("#hillaryclinton",n = 500,lang="en",resultType="recent")

hillaryClinton\_text = sapply(hillaryClinton, function(x) x$getText())

#usableText=str\_replace\_all(tweets$text,"[^[:graph:]]", " ")

hillaryClinton\_corpus = Corpus(VectorSource(hillaryClinton\_text))

#tdm = TermDocumentMatrix(

# hillaryClinton\_corpus,

# control = list(

# removePunctuation = TRUE,

# stopwords = c("hillaryclinton", "HillaryClinton", "hillaryClinton", stopwords("english")),

# removeNumbers = TRUE, tolower = TRUE)

#)

hillary\_clean <- tm\_map(hillaryClinton\_corpus,removePunctuation)

hillary\_clean <- tm\_map(hillary\_clean,content\_transformer(tolower))

hillary\_clean <- tm\_map(hillary\_clean, removeWords, stopwords("english"))

hillary\_clean <- tm\_map(hillary\_clean,removeNumbers)

hillary\_clean <- tm\_map(hillary\_clean,stripWhitespace)

hillary\_clean <- tm\_map(hillary\_clean, removeWords, c("hillaryclinton", "HillaryClinton", "hillaryClinton"))

tdm <- TermDocumentMatrix(hillary\_clean)

m = as.matrix(tdm)

# get word counts in decreasing order

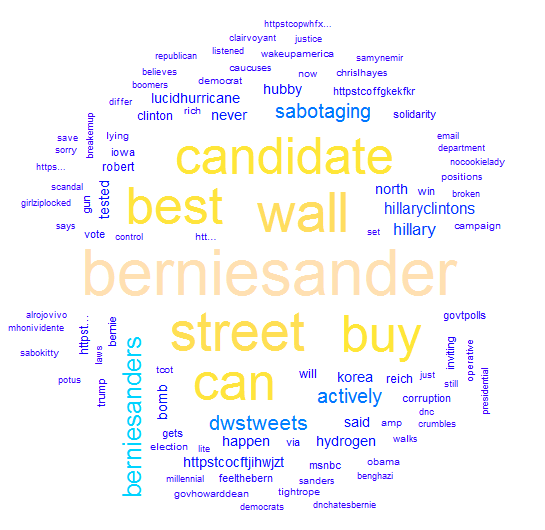
word\_freqs = sort(rowSums(m), decreasing = TRUE)

# create a data frame with words and their frequencies

dm = data.frame(word = names(word\_freqs), freq = word\_freqs)

dm <- dm[1:100,]

wordcloud(dm$word, dm$freq, random.order = FALSE, colors = topo.colors(100))



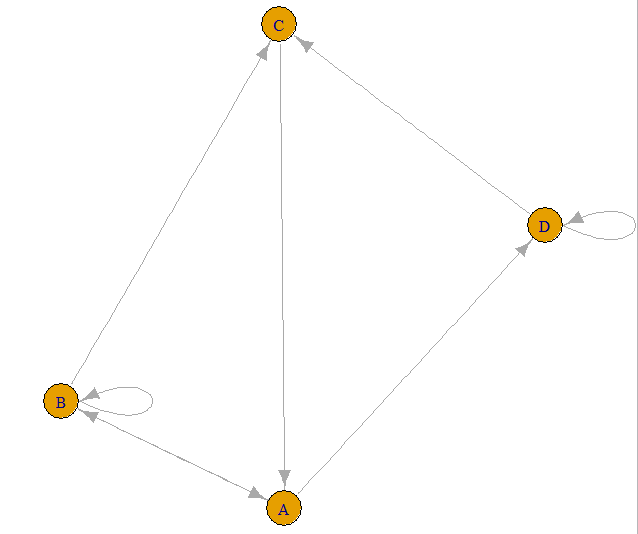
## Page Rank

#install.packages("igraph")

library(igraph)

myGraph <- graph(c( "A", "B", "A", "D", "B", "A", "B", "B", "B", "C", "C", "A", "D", "C", "D", "D"), directed = TRUE)

plot(myGraph)



page.rank(myGraph)$vector



# Additional Documentation

<https://www.r-project.org>

<http://www.rdatamining.com/docs/introduction-to-data-mining-with-r>

<https://www.rstudio.com>

<https://cran.r-project.org/doc/manuals/r-release/fullrefman.pdf>