Muhammad\_Khan\_Capstone\_Project(CMK-136)

Muhammad Khan

July 28, 2018

## R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

1: Data Preperation

1.1 Install packeges and load Libraries

# -- I will use the following package to load the file from http url. ---  
#install.packages('RCurl')  
#install.package("GGally")  
#library(GGally)  
#install.packages("class")  
#install.packages("gmodels")  
library(RCurl) # getURL

## Warning: package 'RCurl' was built under R version 3.4.3

## Loading required package: bitops

library(ggplot2)

## Warning: package 'ggplot2' was built under R version 3.4.4

library(e1071)

## Warning: package 'e1071' was built under R version 3.4.4

library(caret)

## Warning: package 'caret' was built under R version 3.4.4

## Loading required package: lattice

library(boot) ## for Linear Model Validation

## Warning: package 'boot' was built under R version 3.4.4

##   
## Attaching package: 'boot'

## The following object is masked from 'package:lattice':  
##   
## melanoma

# Needed to grow a tree  
library(rpart)  
# To draw a pretty tree (fancyRpartPlot function)  
library(rattle)

## Warning: package 'rattle' was built under R version 3.4.4

## Rattle: A free graphical interface for data science with R.  
## Version 5.1.0 Copyright (c) 2006-2017 Togaware Pty Ltd.  
## Type 'rattle()' to shake, rattle, and roll your data.

library(randomForest)

## Warning: package 'randomForest' was built under R version 3.4.4

## randomForest 4.6-14

## Type rfNews() to see new features/changes/bug fixes.

##   
## Attaching package: 'randomForest'

## The following object is masked from 'package:rattle':  
##   
## importance

## The following object is masked from 'package:ggplot2':  
##   
## margin

library("xlsx")

## Warning: package 'xlsx' was built under R version 3.4.4

library(ROCR)

## Warning: package 'ROCR' was built under R version 3.4.4

## Loading required package: gplots

## Warning: package 'gplots' was built under R version 3.4.4

##   
## Attaching package: 'gplots'

## The following object is masked from 'package:stats':  
##   
## lowess

1.2 All Functions used in this RMD File

#--------Function for Testing NA,NAN and empty values of each attributes of Data Frame---- #  
check\_Data = function(pattr\_name,df){  
pattr\_data = df[,pattr\_name]  
mretna = 0  
mretnan = 0  
mretempty = 0  
mretnull = 0  
 mtotal\_rows\_in\_data = nrow(df)  
 mretempty = sum(pattr\_data =="")  
 mretna = sum(is.na(pattr\_data))  
 mretnan = sum(is.nan(pattr\_data))  
 mretnull = sum(is.null(pattr\_data))  
   
   
 mretna = if(!is.na(mretna)) {mretna} else {0}  
 mretnan = if(!is.na(mretnan)) {mretnan} else {0}  
 mretnull = if(!is.na(mretnull)) {mretnull} else {0}  
 mretempty = if(!is.na(mretempty)) {mretempty} else {0}  
   
 mgooddata = mtotal\_rows\_in\_data-(mretempty+mretna+mretnan+mretnull)  
   
 mPerEmpty = ((mretempty/mtotal\_rows\_in\_data) \* 100)  
 mPerna = ((mretna/mtotal\_rows\_in\_data) \* 100)   
 mPernan = ((mretnan/mtotal\_rows\_in\_data) \* 100)  
 mPernull = ((mretnull/mtotal\_rows\_in\_data) \* 100)  
 mPergood = ( 100- (mPerEmpty+mPerna+mPernan+mPernull) )   
   
 mPerna = if(!is.na(mPerna)) {mPerna} else {0}  
 mPernan = if(!is.na(mPernan)) {mPernan} else {0}  
 mPernull = if(!is.na(mPernull)) {mPernull} else {0}  
 mPerEmpty = if(!is.na(mPerEmpty)) {mPerEmpty} else {0}  
   
   
return(data.frame( Attribute\_Name = c(pattr\_name), Descriptions = c("Total\_No\_Of\_Rows\_In\_DataSet","Good Data","Empty","NA","NAN","Null"), Count = c(mtotal\_rows\_in\_data,mgooddata,mretempty,mretna,mretnan,mretnull),Percentage = c(100,mPergood,mPerEmpty,mPerna,mPernan,mPernull),stringsAsFactors = FALSE))  
}  
  
# ------- Function for plotting a graph of coorelatoin among the attributes ---- #  
panel.cor <- function(x, y, digits=2, prefix="", cex.cor)   
{  
 usr <- par("usr"); on.exit(par(usr))   
 par(usr = c(0, 1, 0, 1))   
 r <- abs(cor(x, y))   
 txt <- format(c(r, 0.123456789), digits=digits)[1]   
 txt <- paste(prefix, txt, sep="")   
 if(missing(cex.cor)) cex <- 0.8/strwidth(txt)   
   
 test <- cor.test(x,y)   
 # borrowed from printCoefmat  
 Signif <- symnum(test$p.value, corr = FALSE, na = FALSE,   
 cutpoints = c(0, 0.001, 0.01, 0.05, 0.1, 1),  
 symbols = c("\*\*\*", "\*\*", "\*", ".", " "))   
   
 text(0.5, 0.5, txt, cex = cex \* r)   
 text(.8, .8, Signif, cex=cex, col=2)   
}  
#-----End Of Correlatoin Test graph--- #  
  
glm.tune <- function(model, dataset) {  
 results <- data.frame()  
 for (q in seq(0.02, 0.65, by = 0.02)) {  
 fitted\_values <- model$fitted.values  
 prediction <- ifelse(fitted\_values >= q, "1", "0")   
 cm <- confusionMatrix(prediction, dataset$Rained)  
 accuracy <- cm$overall["Accuracy"]  
 specificity <- cm$byClass["Specificity"]  
 results <- rbind(results, data.frame(cutoff=q, accuracy=accuracy, specificity = specificity))  
 }  
 rownames(results) <- NULL  
 results  
}  
##----- End of glm\_tun function --------#

1.3 Load data file

# --- Load data file stored in GitHub repository for this project ---  
FileURL <- getURL("https://raw.githubusercontent.com/muhammadBePatience/Capstone/master/Daily\_Weather\_Toronto.csv")  
data <- read.csv(text = FileURL,stringsAsFactors = FALSE)  
colnames(data)

## [1] "Date.Time" "Year" "Month"   
## [4] "Day" "Data\_Quality" "Max\_Temp"   
## [7] "Max\_Temp\_Flag" "Min\_Temp" "Min\_Temp\_Flag"   
## [10] "Mean\_Temp" "Mean\_Temp\_Flag" "Heat\_Deg\_Days"   
## [13] "Heat\_Deg\_Days\_Flag" "Cool\_Deg\_Days" "Cool\_Deg\_Days\_Flag"   
## [16] "Total\_Rain" "Total\_Rain\_Flag" "Total\_Snow"   
## [19] "Total\_Snow\_Flag" "Total\_Precip" "Total\_Precip\_Flag"   
## [22] "Snow\_on\_Grnd" "Snow\_on\_Grnd\_Flag" "Dir\_of\_Max\_Gust"   
## [25] "Dir\_of\_Max\_Gust\_Flag" "Spd\_of\_Max\_Gust" "Spd\_of\_Max\_Gust\_Flag"

1.4 Initial checkup of Data

#Check class of each attribute of Data set  
  
(attributes\_types <- sapply(data,class))

## Date.Time Year Month   
## "character" "integer" "integer"   
## Day Data\_Quality Max\_Temp   
## "integer" "logical" "numeric"   
## Max\_Temp\_Flag Min\_Temp Min\_Temp\_Flag   
## "character" "numeric" "character"   
## Mean\_Temp Mean\_Temp\_Flag Heat\_Deg\_Days   
## "numeric" "logical" "numeric"   
## Heat\_Deg\_Days\_Flag Cool\_Deg\_Days Cool\_Deg\_Days\_Flag   
## "logical" "numeric" "logical"   
## Total\_Rain Total\_Rain\_Flag Total\_Snow   
## "numeric" "character" "numeric"   
## Total\_Snow\_Flag Total\_Precip Total\_Precip\_Flag   
## "logical" "numeric" "character"   
## Snow\_on\_Grnd Snow\_on\_Grnd\_Flag Dir\_of\_Max\_Gust   
## "integer" "character" "integer"   
## Dir\_of\_Max\_Gust\_Flag Spd\_of\_Max\_Gust Spd\_of\_Max\_Gust\_Flag   
## "logical" "character" "logical"

summary(data)

## Date.Time Year Month Day   
## Length:57343 Min. :1846 Min. : 1.000 Min. : 1.00   
## Class :character 1st Qu.:1885 1st Qu.: 4.000 1st Qu.: 8.00   
## Mode :character Median :1924 Median : 7.000 Median :16.00   
## Mean :1924 Mean : 6.523 Mean :15.73   
## 3rd Qu.:1963 3rd Qu.:10.000 3rd Qu.:23.00   
## Max. :2002 Max. :12.000 Max. :31.00   
##   
## Data\_Quality Max\_Temp Max\_Temp\_Flag Min\_Temp   
## Mode:logical Min. :-25.0 Length:57343 Min. :-32.800   
## NA's:57343 1st Qu.: 3.3 Class :character 1st Qu.: -2.800   
## Median : 12.2 Mode :character Median : 3.900   
## Mean : 12.3 Mean : 3.832   
## 3rd Qu.: 22.2 3rd Qu.: 12.200   
## Max. : 40.6 Max. : 26.400   
##   
## Min\_Temp\_Flag Mean\_Temp Mean\_Temp\_Flag Heat\_Deg\_Days   
## Length:57343 Min. :-27.200 Mode:logical Min. : 0.00   
## Class :character 1st Qu.: 0.300 NA's:57343 1st Qu.: 0.90   
## Mode :character Median : 8.300 Median : 9.70   
## Mean : 8.079 Mean :10.69   
## 3rd Qu.: 17.100 3rd Qu.:17.70   
## Max. : 33.100 Max. :45.20   
##   
## Heat\_Deg\_Days\_Flag Cool\_Deg\_Days Cool\_Deg\_Days\_Flag Total\_Rain   
## Mode:logical Min. : 0.0000 Mode:logical Min. : 0.000   
## NA's:57343 1st Qu.: 0.0000 NA's:57343 1st Qu.: 0.000   
## Median : 0.0000 Median : 0.000   
## Mean : 0.7653 Mean : 1.826   
## 3rd Qu.: 0.0000 3rd Qu.: 0.500   
## Max. :15.1000 Max. :98.600   
##   
## Total\_Rain\_Flag Total\_Snow Total\_Snow\_Flag Total\_Precip   
## Length:57343 Min. : 0.0000 Mode:logical Min. : 0.000   
## Class :character 1st Qu.: 0.0000 TRUE:4748 1st Qu.: 0.000   
## Mode :character Median : 0.0000 NA's:52595 Median : 0.000   
## Mean : 0.4094 Mean : 2.231   
## 3rd Qu.: 0.0000 3rd Qu.: 1.400   
## Max. :48.3000 Max. :98.600   
## NA's :182 NA's :182   
## Total\_Precip\_Flag Snow\_on\_Grnd Snow\_on\_Grnd\_Flag Dir\_of\_Max\_Gust  
## Length:57343 Min. : 0.00 Length:57343 Min. : 5.0   
## Class :character 1st Qu.: 0.00 Class :character 1st Qu.: 9.0   
## Mode :character Median : 0.00 Mode :character Median :27.0   
## Mean : 1.73 Mean :21.7   
## 3rd Qu.: 0.00 3rd Qu.:27.0   
## Max. :65.00 Max. :36.0   
## NA's :40155 NA's :57306   
## Dir\_of\_Max\_Gust\_Flag Spd\_of\_Max\_Gust Spd\_of\_Max\_Gust\_Flag  
## Mode:logical Length:57343 Mode:logical   
## NA's:57343 Class :character NA's:57343   
## Mode :character   
##   
##   
##   
##

#check each Atrribute of Data Set for missing values , Row counts, empty values etc...---  
cc <- (data.frame(colnames(data[0,])))   
colnames(cc) = as.character("attribute")  
cc$attribute = as.character(cc$attribute)  
#class(cc)  
  
(Attribute\_Status <- apply(cc ,1, function(x,y) check\_Data(x,data)))

## Attribute\_Name Descriptions Count Percentage  
## 1 Date.Time Total\_No\_Of\_Rows\_In\_DataSet 57343 100  
## 2 Date.Time Good Data 57343 100  
## 3 Date.Time Empty 0 0  
## 4 Date.Time NA 0 0  
## 5 Date.Time NAN 0 0  
## 6 Date.Time Null 0 0  
##   
## [[2]]  
## Attribute\_Name Descriptions Count Percentage  
## 1 Year Total\_No\_Of\_Rows\_In\_DataSet 57343 100  
## 2 Year Good Data 57343 100  
## 3 Year Empty 0 0  
## 4 Year NA 0 0  
## 5 Year NAN 0 0  
## 6 Year Null 0 0  
##   
## [[3]]  
## Attribute\_Name Descriptions Count Percentage  
## 1 Month Total\_No\_Of\_Rows\_In\_DataSet 57343 100  
## 2 Month Good Data 57343 100  
## 3 Month Empty 0 0  
## 4 Month NA 0 0  
## 5 Month NAN 0 0  
## 6 Month Null 0 0  
##   
## [[4]]  
## Attribute\_Name Descriptions Count Percentage  
## 1 Day Total\_No\_Of\_Rows\_In\_DataSet 57343 100  
## 2 Day Good Data 57343 100  
## 3 Day Empty 0 0  
## 4 Day NA 0 0  
## 5 Day NAN 0 0  
## 6 Day Null 0 0  
##   
## [[5]]  
## Attribute\_Name Descriptions Count Percentage  
## 1 Data\_Quality Total\_No\_Of\_Rows\_In\_DataSet 57343 100  
## 2 Data\_Quality Good Data 0 0  
## 3 Data\_Quality Empty 0 0  
## 4 Data\_Quality NA 57343 100  
## 5 Data\_Quality NAN 0 0  
## 6 Data\_Quality Null 0 0  
##   
## [[6]]  
## Attribute\_Name Descriptions Count Percentage  
## 1 Max\_Temp Total\_No\_Of\_Rows\_In\_DataSet 57343 100  
## 2 Max\_Temp Good Data 57343 100  
## 3 Max\_Temp Empty 0 0  
## 4 Max\_Temp NA 0 0  
## 5 Max\_Temp NAN 0 0  
## 6 Max\_Temp Null 0 0  
##   
## [[7]]  
## Attribute\_Name Descriptions Count Percentage  
## 1 Max\_Temp\_Flag Total\_No\_Of\_Rows\_In\_DataSet 57343 1.000000e+02  
## 2 Max\_Temp\_Flag Good Data 1 1.743892e-03  
## 3 Max\_Temp\_Flag Empty 57342 9.999826e+01  
## 4 Max\_Temp\_Flag NA 0 0.000000e+00  
## 5 Max\_Temp\_Flag NAN 0 0.000000e+00  
## 6 Max\_Temp\_Flag Null 0 0.000000e+00  
##   
## [[8]]  
## Attribute\_Name Descriptions Count Percentage  
## 1 Min\_Temp Total\_No\_Of\_Rows\_In\_DataSet 57343 100  
## 2 Min\_Temp Good Data 57343 100  
## 3 Min\_Temp Empty 0 0  
## 4 Min\_Temp NA 0 0  
## 5 Min\_Temp NAN 0 0  
## 6 Min\_Temp Null 0 0  
##   
## [[9]]  
## Attribute\_Name Descriptions Count Percentage  
## 1 Min\_Temp\_Flag Total\_No\_Of\_Rows\_In\_DataSet 57343 1.000000e+02  
## 2 Min\_Temp\_Flag Good Data 4 6.975568e-03  
## 3 Min\_Temp\_Flag Empty 57339 9.999302e+01  
## 4 Min\_Temp\_Flag NA 0 0.000000e+00  
## 5 Min\_Temp\_Flag NAN 0 0.000000e+00  
## 6 Min\_Temp\_Flag Null 0 0.000000e+00  
##   
## [[10]]  
## Attribute\_Name Descriptions Count Percentage  
## 1 Mean\_Temp Total\_No\_Of\_Rows\_In\_DataSet 57343 100  
## 2 Mean\_Temp Good Data 57343 100  
## 3 Mean\_Temp Empty 0 0  
## 4 Mean\_Temp NA 0 0  
## 5 Mean\_Temp NAN 0 0  
## 6 Mean\_Temp Null 0 0  
##   
## [[11]]  
## Attribute\_Name Descriptions Count Percentage  
## 1 Mean\_Temp\_Flag Total\_No\_Of\_Rows\_In\_DataSet 57343 100  
## 2 Mean\_Temp\_Flag Good Data 0 0  
## 3 Mean\_Temp\_Flag Empty 0 0  
## 4 Mean\_Temp\_Flag NA 57343 100  
## 5 Mean\_Temp\_Flag NAN 0 0  
## 6 Mean\_Temp\_Flag Null 0 0  
##   
## [[12]]  
## Attribute\_Name Descriptions Count Percentage  
## 1 Heat\_Deg\_Days Total\_No\_Of\_Rows\_In\_DataSet 57343 100  
## 2 Heat\_Deg\_Days Good Data 57343 100  
## 3 Heat\_Deg\_Days Empty 0 0  
## 4 Heat\_Deg\_Days NA 0 0  
## 5 Heat\_Deg\_Days NAN 0 0  
## 6 Heat\_Deg\_Days Null 0 0  
##   
## [[13]]  
## Attribute\_Name Descriptions Count Percentage  
## 1 Heat\_Deg\_Days\_Flag Total\_No\_Of\_Rows\_In\_DataSet 57343 100  
## 2 Heat\_Deg\_Days\_Flag Good Data 0 0  
## 3 Heat\_Deg\_Days\_Flag Empty 0 0  
## 4 Heat\_Deg\_Days\_Flag NA 57343 100  
## 5 Heat\_Deg\_Days\_Flag NAN 0 0  
## 6 Heat\_Deg\_Days\_Flag Null 0 0  
##   
## [[14]]  
## Attribute\_Name Descriptions Count Percentage  
## 1 Cool\_Deg\_Days Total\_No\_Of\_Rows\_In\_DataSet 57343 100  
## 2 Cool\_Deg\_Days Good Data 57343 100  
## 3 Cool\_Deg\_Days Empty 0 0  
## 4 Cool\_Deg\_Days NA 0 0  
## 5 Cool\_Deg\_Days NAN 0 0  
## 6 Cool\_Deg\_Days Null 0 0  
##   
## [[15]]  
## Attribute\_Name Descriptions Count Percentage  
## 1 Cool\_Deg\_Days\_Flag Total\_No\_Of\_Rows\_In\_DataSet 57343 100  
## 2 Cool\_Deg\_Days\_Flag Good Data 0 0  
## 3 Cool\_Deg\_Days\_Flag Empty 0 0  
## 4 Cool\_Deg\_Days\_Flag NA 57343 100  
## 5 Cool\_Deg\_Days\_Flag NAN 0 0  
## 6 Cool\_Deg\_Days\_Flag Null 0 0  
##   
## [[16]]  
## Attribute\_Name Descriptions Count Percentage  
## 1 Total\_Rain Total\_No\_Of\_Rows\_In\_DataSet 57343 100  
## 2 Total\_Rain Good Data 57343 100  
## 3 Total\_Rain Empty 0 0  
## 4 Total\_Rain NA 0 0  
## 5 Total\_Rain NAN 0 0  
## 6 Total\_Rain Null 0 0  
##   
## [[17]]  
## Attribute\_Name Descriptions Count Percentage  
## 1 Total\_Rain\_Flag Total\_No\_Of\_Rows\_In\_DataSet 57343 100.000000  
## 2 Total\_Rain\_Flag Good Data 4673 8.149207  
## 3 Total\_Rain\_Flag Empty 52670 91.850793  
## 4 Total\_Rain\_Flag NA 0 0.000000  
## 5 Total\_Rain\_Flag NAN 0 0.000000  
## 6 Total\_Rain\_Flag Null 0 0.000000  
##   
## [[18]]  
## Attribute\_Name Descriptions Count Percentage  
## 1 Total\_Snow Total\_No\_Of\_Rows\_In\_DataSet 57343 100.0000000  
## 2 Total\_Snow Good Data 57161 99.6826117  
## 3 Total\_Snow Empty 0 0.0000000  
## 4 Total\_Snow NA 182 0.3173883  
## 5 Total\_Snow NAN 0 0.0000000  
## 6 Total\_Snow Null 0 0.0000000  
##   
## [[19]]  
## Attribute\_Name Descriptions Count Percentage  
## 1 Total\_Snow\_Flag Total\_No\_Of\_Rows\_In\_DataSet 57343 100.000000  
## 2 Total\_Snow\_Flag Good Data 4748 8.279999  
## 3 Total\_Snow\_Flag Empty 0 0.000000  
## 4 Total\_Snow\_Flag NA 52595 91.720001  
## 5 Total\_Snow\_Flag NAN 0 0.000000  
## 6 Total\_Snow\_Flag Null 0 0.000000  
##   
## [[20]]  
## Attribute\_Name Descriptions Count Percentage  
## 1 Total\_Precip Total\_No\_Of\_Rows\_In\_DataSet 57343 100.0000000  
## 2 Total\_Precip Good Data 57161 99.6826117  
## 3 Total\_Precip Empty 0 0.0000000  
## 4 Total\_Precip NA 182 0.3173883  
## 5 Total\_Precip NAN 0 0.0000000  
## 6 Total\_Precip Null 0 0.0000000  
##   
## [[21]]  
## Attribute\_Name Descriptions Count Percentage  
## 1 Total\_Precip\_Flag Total\_No\_Of\_Rows\_In\_DataSet 57343 100.00000  
## 2 Total\_Precip\_Flag Good Data 7882 13.74536  
## 3 Total\_Precip\_Flag Empty 49461 86.25464  
## 4 Total\_Precip\_Flag NA 0 0.00000  
## 5 Total\_Precip\_Flag NAN 0 0.00000  
## 6 Total\_Precip\_Flag Null 0 0.00000  
##   
## [[22]]  
## Attribute\_Name Descriptions Count Percentage  
## 1 Snow\_on\_Grnd Total\_No\_Of\_Rows\_In\_DataSet 57343 100.00000  
## 2 Snow\_on\_Grnd Good Data 17188 29.97402  
## 3 Snow\_on\_Grnd Empty 0 0.00000  
## 4 Snow\_on\_Grnd NA 40155 70.02598  
## 5 Snow\_on\_Grnd NAN 0 0.00000  
## 6 Snow\_on\_Grnd Null 0 0.00000  
##   
## [[23]]  
## Attribute\_Name Descriptions Count Percentage  
## 1 Snow\_on\_Grnd\_Flag Total\_No\_Of\_Rows\_In\_DataSet 57343 100.000000  
## 2 Snow\_on\_Grnd\_Flag Good Data 1488 2.594911  
## 3 Snow\_on\_Grnd\_Flag Empty 55855 97.405089  
## 4 Snow\_on\_Grnd\_Flag NA 0 0.000000  
## 5 Snow\_on\_Grnd\_Flag NAN 0 0.000000  
## 6 Snow\_on\_Grnd\_Flag Null 0 0.000000  
##   
## [[24]]  
## Attribute\_Name Descriptions Count Percentage  
## 1 Dir\_of\_Max\_Gust Total\_No\_Of\_Rows\_In\_DataSet 57343 100.000000  
## 2 Dir\_of\_Max\_Gust Good Data 37 0.064524  
## 3 Dir\_of\_Max\_Gust Empty 0 0.000000  
## 4 Dir\_of\_Max\_Gust NA 57306 99.935476  
## 5 Dir\_of\_Max\_Gust NAN 0 0.000000  
## 6 Dir\_of\_Max\_Gust Null 0 0.000000  
##   
## [[25]]  
## Attribute\_Name Descriptions Count Percentage  
## 1 Dir\_of\_Max\_Gust\_Flag Total\_No\_Of\_Rows\_In\_DataSet 57343 100  
## 2 Dir\_of\_Max\_Gust\_Flag Good Data 0 0  
## 3 Dir\_of\_Max\_Gust\_Flag Empty 0 0  
## 4 Dir\_of\_Max\_Gust\_Flag NA 57343 100  
## 5 Dir\_of\_Max\_Gust\_Flag NAN 0 0  
## 6 Dir\_of\_Max\_Gust\_Flag Null 0 0  
##   
## [[26]]  
## Attribute\_Name Descriptions Count Percentage  
## 1 Spd\_of\_Max\_Gust Total\_No\_Of\_Rows\_In\_DataSet 57343 100.0000000  
## 2 Spd\_of\_Max\_Gust Good Data 92 0.1604381  
## 3 Spd\_of\_Max\_Gust Empty 57251 99.8395619  
## 4 Spd\_of\_Max\_Gust NA 0 0.0000000  
## 5 Spd\_of\_Max\_Gust NAN 0 0.0000000  
## 6 Spd\_of\_Max\_Gust Null 0 0.0000000  
##   
## [[27]]  
## Attribute\_Name Descriptions Count Percentage  
## 1 Spd\_of\_Max\_Gust\_Flag Total\_No\_Of\_Rows\_In\_DataSet 57343 100  
## 2 Spd\_of\_Max\_Gust\_Flag Good Data 0 0  
## 3 Spd\_of\_Max\_Gust\_Flag Empty 0 0  
## 4 Spd\_of\_Max\_Gust\_Flag NA 57343 100  
## 5 Spd\_of\_Max\_Gust\_Flag NAN 0 0  
## 6 Spd\_of\_Max\_Gust\_Flag Null 0 0

#check individually each column of Data Set ---  
#check\_Data("Total\_Precip",Actual\_Data)  
#check\_Data("Total\_Rain",Actual\_Data)  
#check\_Data("Max\_Temp",Actual\_Data)  
#check\_Data("Min\_Temp",Actual\_Data)  
#check\_Data("Mean\_Temp",Actual\_Data)  
#check\_Data("Total\_Precip",Actual\_Data)  
#check\_Data("Heat\_Deg\_Days",Actual\_Data)  
#check\_Data("Dir\_of\_Max\_Gust",Actual\_Data)  
#check\_Data("Spd\_of\_Max\_Gust",Actual\_Data)  
  
  
#First make a copy of Original dataset.#  
Actual\_Data <-data

1.5 Concrete decisions for data set

# Based on the above information, i will do following additions and deletions of attributes of DataSet   
  
#Based on this check\_data function result, i will remove couple of variables..  
Actual\_Data <- Actual\_Data[c("Date.Time","Year","Month","Max\_Temp","Min\_Temp","Mean\_Temp",  
 "Heat\_Deg\_Days","Total\_Rain","Total\_Precip")]  
  
#Rename Date.Time attribute to Date and change data type to Date data type  
dates = trimws(Actual\_Data$Date.Time)  
dates = as.Date(dates)  
dates = data.frame(Date = dates)  
  
colnames(Actual\_Data)[1] ="Date"  
Actual\_Data$Date = dates$Date  
  
  
  
# NAs check #  
#Baesd on the above check\_Data function result, I found NAs in one of the attribute "Total\_Precip", It should be fix   
#Updating Total\_Percip attribute for NA values with mean(Total\_Percip)  
Actual\_Data$Total\_Precip[is.na(data$Total\_Precip)] <- mean(Actual\_Data$Total\_Precip,na.rm =TRUE)  
  
  
## In order to help visualization of Data Set,i decieded to add an extra variable called "Season", based on existing Date attribute, it will help to visualize the data, Its value depend upon the Date columns, Data type will be Factor, there will be four(4) level of this attribute, (Winter,Spring,Summer,Autumn)  
d = function(month\_day) which(lut$month\_day == month\_day)  
lut = data.frame(all\_dates = as.POSIXct("2012-01-01") + ((0:365) \* 3600 \* 24),  
 season = NA)  
lut = within(lut, { month\_day = strftime(all\_dates, "%b-%d") })  
lut[c(d("Jan-01"):d("Mar-20"), d("Dec-21"):d("Dec-31")), "season"] = "Winter"  
lut[c(d("Mar-21"):d("Jun-20")), "season"] = "Spring"  
lut[c(d("Jun-21"):d("Sep-20")), "season"] = "Summer"  
lut[c(d("Sep-21"):d("Dec-20")), "season"] = "Autumn"  
rownames(lut) = lut$month\_day  
  
dat = data.frame(dates = Actual\_Data$Date + (0:11)\*30)

## Warning in unclass(e1) + unclass(e2): longer object length is not a  
## multiple of shorter object length

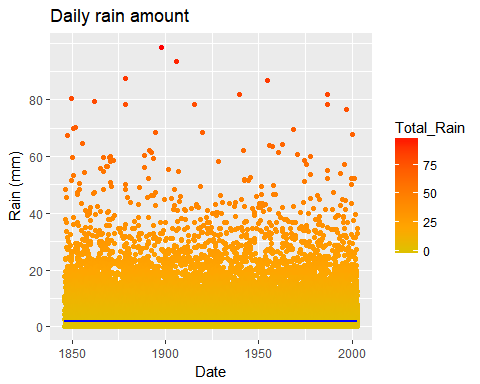
dat = within(dat, {   
 season = lut[strftime(dates, "%b-%d"), "season"]   
})  
  
Actual\_Data$Season = (dat$season)  
  
## Will add a varriable called month\_name , based on the Month Varriable ##   
monthnames = c("Jan","Feb","Mar","Apr","May",  
 "Jun","Jul","Aug","Sep","Oct",  
 "Nov","Dec")  
Actual\_Data$Month\_Name<- as.factor(monthnames[Actual\_Data$Month])  
  
#Reordring the column names of Data Set   
Actual\_Data <- Actual\_Data[c(1,2,3,11,10,4,5,6,7,8,9)]

1.6 Data Visualization

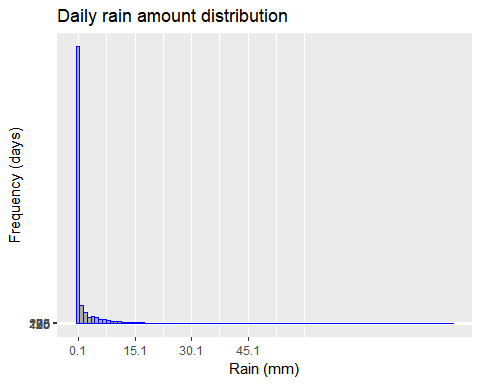
1.6.1 Lets visualize the data,so get more clear picture of data and then i will take some other decisions

ggplot(Actual\_Data, aes(Date,Total\_Rain)) +  
 geom\_point(aes(colour = Total\_Rain)) +  
 geom\_smooth(colour = "blue", size = 1) +  
 scale\_colour\_gradient2(low = "green", mid = "orange",high = "red", midpoint = 20) +  
 scale\_y\_continuous(breaks = seq(0,80,20)) +  
 xlab("Date") +  
 ylab("Rain (mm)") +  
 ggtitle("Daily rain amount")

## `geom\_smooth()` using method = 'gam'



## Histogram Of Total\_Rain Variable ##   
ggplot(Actual\_Data ,aes(Total\_Rain)) +   
 geom\_histogram(binwidth = 1,colour = "blue", fill = "darkgrey") +  
 scale\_x\_continuous(breaks = seq(0.1,58,15)) +  
 scale\_y\_continuous(breaks = seq(0,225,25)) +  
 xlab("Rain (mm)") +  
 ylab ("Frequency (days)") +  
 ggtitle("Daily rain amount distribution")



#Becuase attribute Total\_Rain is left skewed, i will add a binary variable based on the Total\_Rain attribute.  
  
#some more for data is balance or skewed ##  
  
#Heavily Left-skewed distribution  
(summary(Actual\_Data$Total\_Rain))

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0.000 0.000 0.000 1.826 0.500 98.600

# Left-skewness is still there after removing all the dry days  
summary(subset(Actual\_Data, Total\_Rain > 0)$Total\_Rain)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0.200 1.000 3.300 6.423 8.400 98.600

skewness(Actual\_Data$Total\_Rain)

## [1] 5.210701

skewness(subset(Actual\_Data, Total\_Rain >0)$Total\_Rain)

## [1] 2.915972

nrow(subset(Actual\_Data,Total\_Rain==0) )

## [1] 41044

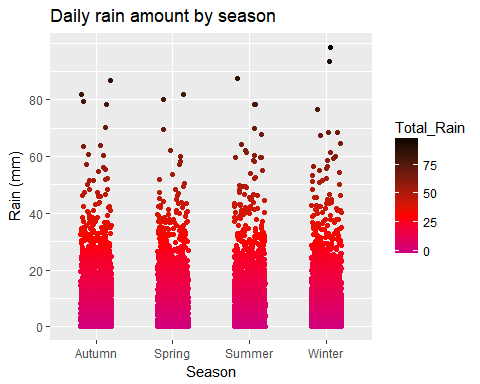
nrow(subset(Actual\_Data,Total\_Rain>0) )

## [1] 16299

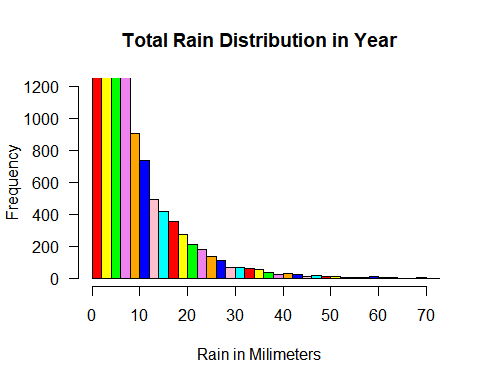
Actual\_Data$Rained <- as.factor(ifelse(Actual\_Data$Total\_Rain >= 1, 1, 0))

1.6.2 More visualization of Data attributes

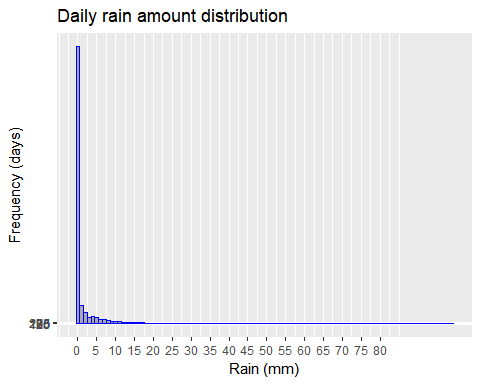
#Total\_Rain by Season   
ggplot(Actual\_Data, aes(Season,Total\_Rain)) +  
 geom\_jitter(aes(colour=Total\_Rain), position = position\_jitter(width = 0.2)) +  
 scale\_colour\_gradient2(low = "blue", mid = "red",high = "black", midpoint = 30) +  
 scale\_y\_continuous(breaks = seq(0,80,20)) +  
 xlab("Season") +  
 ylab ("Rain (mm)") +  
 ggtitle("Daily rain amount by season")



#Historgram of Daily Rain  
colors = c("red", "yellow", "green", "violet", "orange", "blue", "pink", "cyan")  
hist(Actual\_Data$Total\_Rain,  
 right=FALSE, col=colors,   
 main = "Total Rain Distribution in Year",  
 xlab = "Rain in Milimeters",  
 xlim=c(0,70), ylim=c(0,1200),  
 las=1,   
 breaks=c(50)  
)



ggplot(Actual\_Data,aes(Total\_Rain)) +   
 geom\_histogram(binwidth = 1,colour = "blue", fill = "darkgrey") +  
 scale\_x\_continuous(breaks = seq(0,80,5)) +  
 scale\_y\_continuous(breaks = seq(0,225,25)) +  
 xlab("Rain (mm)") +  
 ylab ("Frequency (days)") +  
 ggtitle("Daily rain amount distribution")



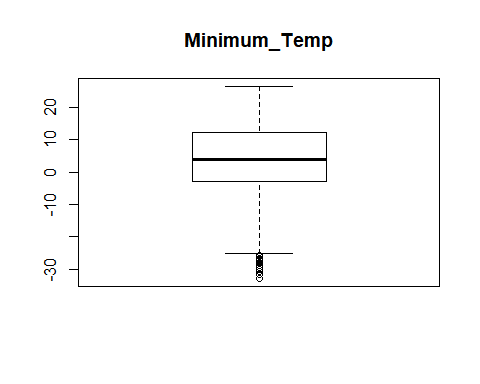
1.7 Check Dry and Wet Days Ratio and outlier in attributes

#Dry and wet days (absolute)  
#table(rained = Actual\_Data$Rained)   
# Dry and wet days (relative)  
prop.table(table(rained = Actual\_Data$Rained))

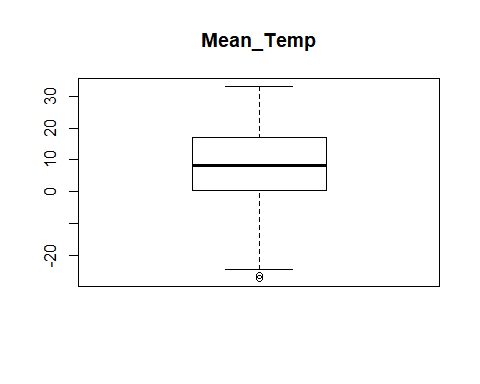
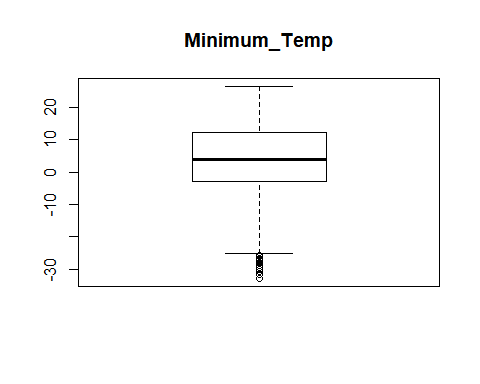
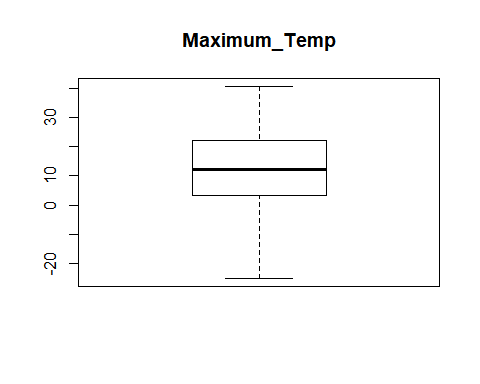
## rained  
## 0 1   
## 0.777305 0.222695

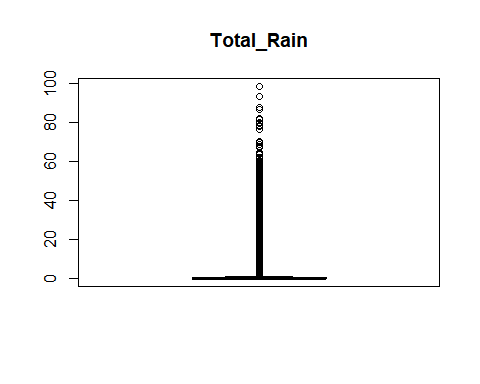
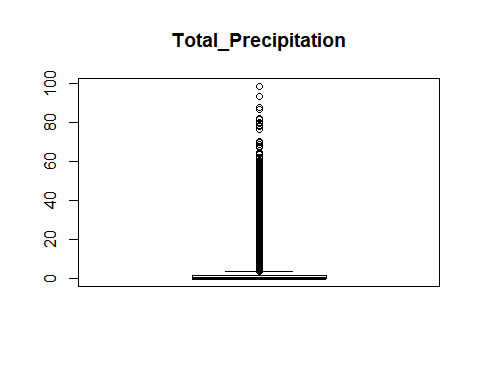
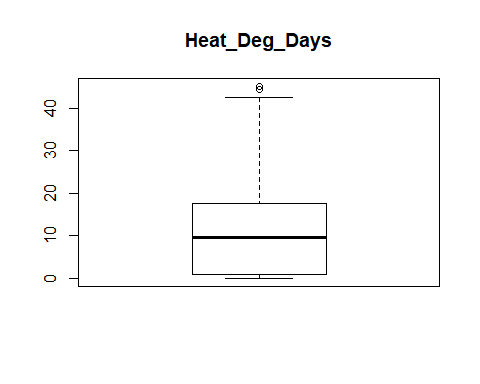
1.8 Box Plots for varriables to see the outliner

boxplot(Actual\_Data$Min\_Temp,data=Actual\_Data ,main="Minimum\_Temp")

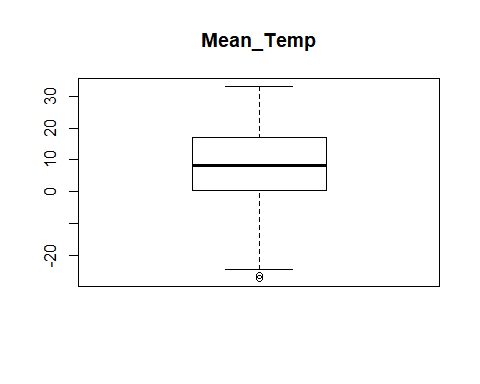


boxplot(Actual\_Data$Max\_Temp,data=Actual\_Data, main="Maximum\_Temp")

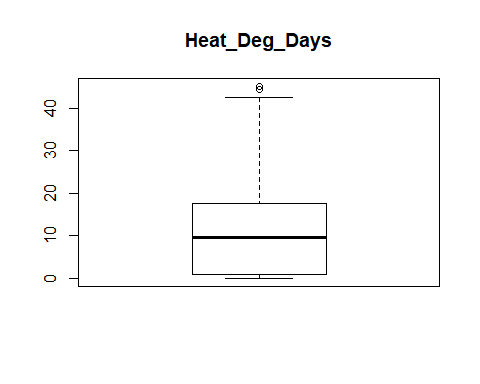




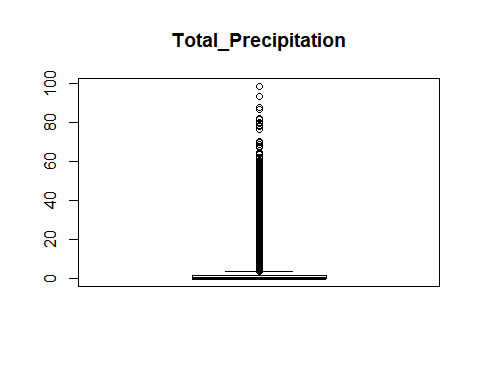
boxplot(Actual\_Data$Mean\_Temp,data=Actual\_Data,main="Mean\_Temp")



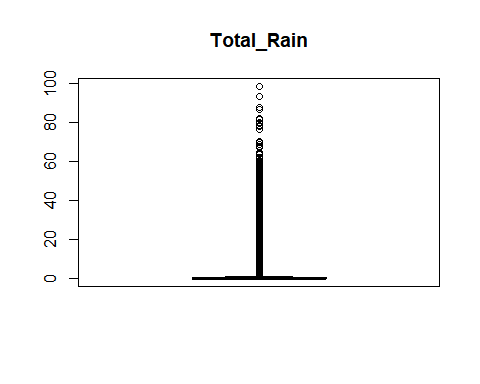
boxplot(Actual\_Data$Heat\_Deg\_Days,data=Actual\_Data,main="Heat\_Deg\_Days")



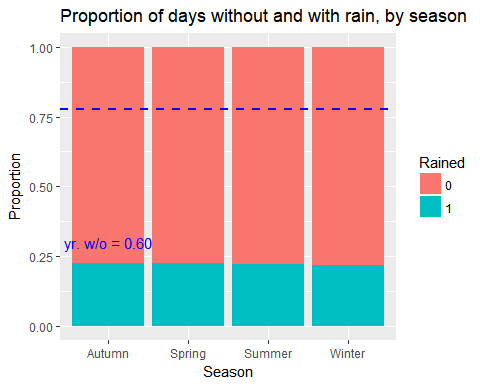
boxplot(Actual\_Data$Total\_Precip,data=Actual\_Data,main="Total\_Precipitation")



boxplot(Actual\_Data$Total\_Rain,data=Actual\_Data,main="Total\_Rain")



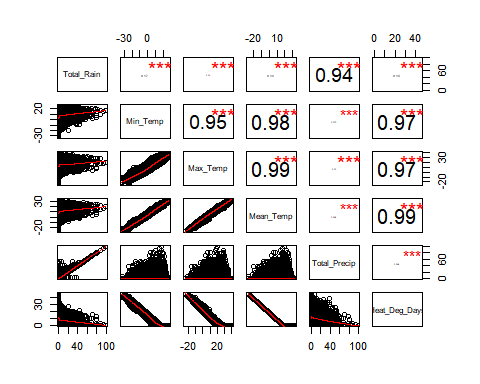
Actual\_Data\_Featuers <- Actual\_Data[,c("Rained","Season")]  
ggplot(Actual\_Data,aes(Season)) +  
 geom\_bar(aes(fill = Rained), position = "fill") +  
 geom\_hline(aes(yintercept = prop.table(table(Actual\_Data$Rained))["0"]),  
 colour = "blue",linetype = "dashed", size = 1) +  
 annotate("text", x = 1, y = 0.30, label = "yr. w/o = 0.60", colour = "blue") +  
 xlab("Season") +  
 ylab ("Proportion") +  
 ggtitle("Proportion of days without and with rain, by season")



#round(prop.table(table(Season = Actual\_Data\_Final$Season, Rained= #Actual\_Data\_Final$Rained),1),2)

1.9 Plotting a graph for checking a relationship among attributes of data

#pairs(Actual\_Data, lower.panel=panel.smooth, upper.panel=panel.cor)  
Actual\_Data\_Featuers <- Actual\_Data[,c("Total\_Rain","Min\_Temp","Max\_Temp","Mean\_Temp","Total\_Precip","Heat\_Deg\_Days")]  
pairs(Actual\_Data\_Featuers, lower.panel=panel.smooth, upper.panel=panel.cor)



#View(Actual\_Data\_Final)

1.10 Coorelaton Test

#Note: Above plots shows the +ve coorleation, so do some coorelaton test #   
  
cor.test(Actual\_Data$Total\_Rain, Actual\_Data$Min\_Temp,data=Actual\_Data)

##   
## Pearson's product-moment correlation  
##   
## data: Actual\_Data$Total\_Rain and Actual\_Data$Min\_Temp  
## t = 40.51, df = 57341, p-value < 2.2e-16  
## alternative hypothesis: true correlation is not equal to 0  
## 95 percent confidence interval:  
## 0.1588334 0.1747476  
## sample estimates:  
## cor   
## 0.1668014

cor.test(Actual\_Data$Total\_Rain, Actual\_Data$Max\_Temp,data=Actual\_Data)

##   
## Pearson's product-moment correlation  
##   
## data: Actual\_Data$Total\_Rain and Actual\_Data$Max\_Temp  
## t = 27.059, df = 57341, p-value < 2.2e-16  
## alternative hypothesis: true correlation is not equal to 0  
## 95 percent confidence interval:  
## 0.1041958 0.1203591  
## sample estimates:  
## cor   
## 0.1122849

cor.test(Actual\_Data$Total\_Rain, Actual\_Data$Mean\_Temp,data=Actual\_Data)

##   
## Pearson's product-moment correlation  
##   
## data: Actual\_Data$Total\_Rain and Actual\_Data$Mean\_Temp  
## t = 33.886, df = 57341, p-value < 2.2e-16  
## alternative hypothesis: true correlation is not equal to 0  
## 95 percent confidence interval:  
## 0.1320802 0.1481285  
## sample estimates:  
## cor   
## 0.1401136

cor.test(Actual\_Data$Total\_Rain, Actual\_Data$Total\_Precip,data=Actual\_Data)

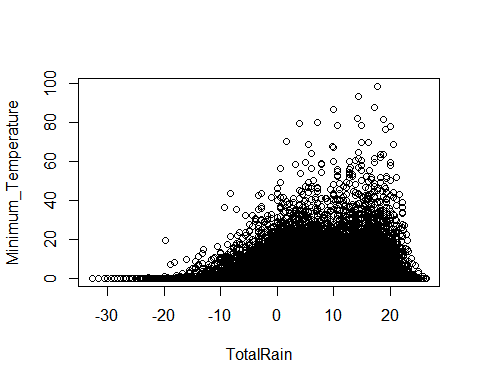
##   
## Pearson's product-moment correlation  
##   
## data: Actual\_Data$Total\_Rain and Actual\_Data$Total\_Precip  
## t = 638.53, df = 57341, p-value < 2.2e-16  
## alternative hypothesis: true correlation is not equal to 0  
## 95 percent confidence interval:  
## 0.9353068 0.9373253  
## sample estimates:  
## cor   
## 0.9363238

cor.test(Actual\_Data$Total\_Rain, Actual\_Data$Heat\_Deg\_Days,data=Actual\_Data)

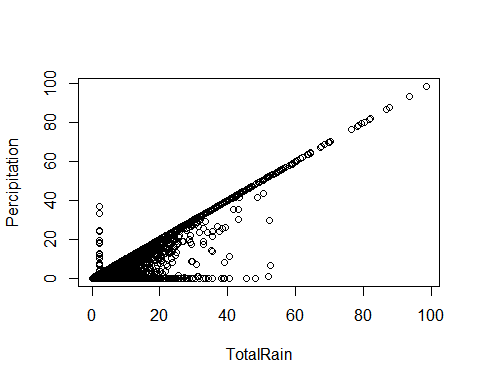
##   
## Pearson's product-moment correlation  
##   
## data: Actual\_Data$Total\_Rain and Actual\_Data$Heat\_Deg\_Days  
## t = -35.364, df = 57341, p-value < 2.2e-16  
## alternative hypothesis: true correlation is not equal to 0  
## 95 percent confidence interval:  
## -0.1540964 -0.1380761  
## sample estimates:  
## cor   
## -0.1460958

1.11 Visualize Data

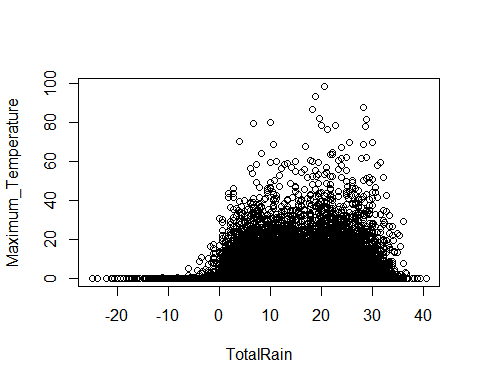
plot(Actual\_Data$Total\_Rain~Actual\_Data$Min\_Temp,xlab="TotalRain",ylab="Minimum\_Temperature")



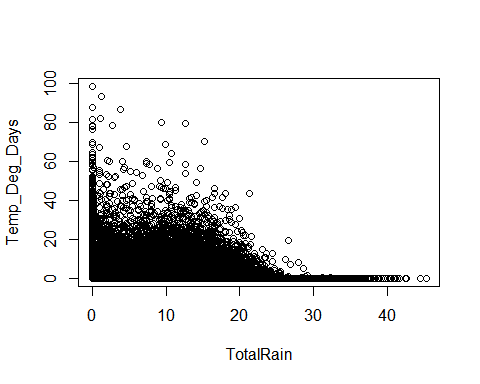
plot(Actual\_Data$Total\_Rain~Actual\_Data$Total\_Precip,xlab="TotalRain",ylab="Percipitation")



plot(Actual\_Data$Total\_Rain~Actual\_Data$Max\_Temp,xlab="TotalRain",ylab="Maximum\_Temperature")



plot(Actual\_Data$Total\_Rain~Actual\_Data$Heat\_Deg\_Days,xlab="TotalRain",ylab="Temp\_Deg\_Days")



1.12 Final Copy of Data Set

Actual\_Data\_Final = Actual\_Data[c("Date","Year","Month","Max\_Temp","Min\_Temp","Mean\_Temp",  
 "Heat\_Deg\_Days","Total\_Rain","Total\_Precip","Rained")]

1. Algorithm Preperatoin

2.1 Preperatoin of Training and Testing Data Set

set.seed(1235)  
  
Lin\_Reg\_Data <- Actual\_Data\_Final[c("Month","Max\_Temp","Min\_Temp","Mean\_Temp","Heat\_Deg\_Days",  
 "Total\_Precip","Total\_Rain")]  
index <- sample(1:nrow(Lin\_Reg\_Data),size = 0.7\*nrow(Lin\_Reg\_Data))   
  
# subset weather to include only the elements in the index  
train <- Lin\_Reg\_Data[index,]   
  
# subset weather to include all but the elements in the index  
test <- Lin\_Reg\_Data [-index,]   
  
nrow(train)

## [1] 40140

nrow(test)

## [1] 17203

2.2 Linear Regression Base Line Model

best.guess <- mean(train$Total\_Rain)   
  
RMSE.baseline <- sqrt(mean((best.guess-test$Total\_Rain)^2))  
RMSE.baseline

## [1] 5.381145

#5.381145  
  
MAE.baseline <- mean(abs(best.guess-test$Total\_Rain))  
MAE.baseline

## [1] 2.838545

#2.838545

2.2 Linear Regression Model

lin.reg <- lm(log(Total\_Rain+1) ~ Mean\_Temp+Max\_Temp+Min\_Temp+Heat\_Deg\_Days+Total\_Precip, data = train)  
# Inspect the model  
summary(lin.reg)

##   
## Call:  
## lm(formula = log(Total\_Rain + 1) ~ Mean\_Temp + Max\_Temp + Min\_Temp +   
## Heat\_Deg\_Days + Total\_Precip, data = train)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -7.6082 -0.2582 -0.1316 0.1572 3.0779   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 0.7720896 0.0308341 25.04 <2e-16 \*\*\*  
## Mean\_Temp 1.2767181 0.0871155 14.65 <2e-16 \*\*\*  
## Max\_Temp -0.6629791 0.0434318 -15.27 <2e-16 \*\*\*  
## Min\_Temp -0.6302972 0.0435142 -14.48 <2e-16 \*\*\*  
## Heat\_Deg\_Days -0.0320808 0.0016976 -18.90 <2e-16 \*\*\*  
## Total\_Precip 0.1196852 0.0004398 272.14 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.4855 on 40134 degrees of freedom  
## Multiple R-squared: 0.6779, Adjusted R-squared: 0.6779   
## F-statistic: 1.689e+04 on 5 and 40134 DF, p-value: < 2.2e-16

exp(lin.reg$coefficients["Total\_Precip"])

## Total\_Precip   
## 1.127142

#Total\_Precip   
# 1.127142   
  
exp(lin.reg$coefficients["Min\_Temp"])

## Min\_Temp   
## 0.5324336

# Min\_Temp   
# 0.5324336   
  
  
# Apply the model to the testing data (i.e., make predictions) ...  
# (Don't forget to exponentiate the results to revert the log transformation)  
test.pred.lin <- exp(predict(lin.reg,test))-1  
  
RMSE.lin.reg <- sqrt(mean((test.pred.lin-test$Total\_Rain)^2))  
RMSE.lin.reg

## [1] 459.4112

MAE.lin.reg <- mean(abs(test.pred.lin-test$Total\_Rain))  
MAE.lin.reg

## [1] 10.05387

2.3 Applied Decision Tree Model

##-- apply the decision tree here ##  
library(rpart)  
library(rattle)  
rt <- rpart(Total\_Rain ~ Month + Max\_Temp + Min\_Temp + Mean\_Temp+Total\_Precip+Heat\_Deg\_Days, data=train)  
test.pred.rtree <- predict(rt,test)  
  
RMSE.rtree <- sqrt(mean((test.pred.rtree-test$Total\_Rain)^2))  
RMSE.rtree

## [1] 1.627743

#1.627743  
MAE.rtree <- mean(abs(test.pred.rtree-test$Total\_Rain))  
MAE.rtree

## [1] 0.6907401

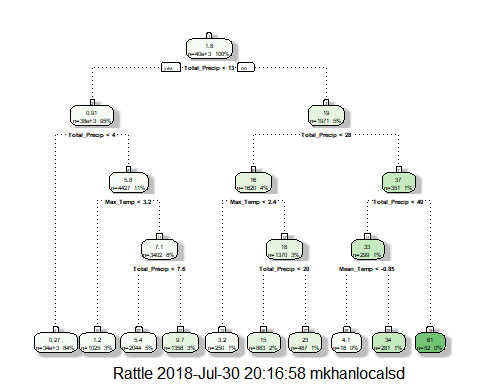
#0.6907401  
  
printcp(rt)

##   
## Regression tree:  
## rpart(formula = Total\_Rain ~ Month + Max\_Temp + Min\_Temp + Mean\_Temp +   
## Total\_Precip + Heat\_Deg\_Days, data = train)  
##   
## Variables actually used in tree construction:  
## [1] Max\_Temp Mean\_Temp Total\_Precip  
##   
## Root node error: 1129974/40140 = 28.151  
##   
## n= 40140   
##   
## CP nsplit rel error xerror xstd  
## 1 0.571516 0 1.000000 1.000043 0.0322814  
## 2 0.113850 1 0.428484 0.432006 0.0161795  
## 3 0.104627 2 0.314634 0.316064 0.0097544  
## 4 0.041195 3 0.210006 0.215717 0.0094529  
## 5 0.031458 4 0.168812 0.173271 0.0090357  
## 6 0.024286 5 0.137353 0.149551 0.0065169  
## 7 0.014936 6 0.113067 0.119146 0.0060939  
## 8 0.013809 7 0.098132 0.100195 0.0056597  
## 9 0.013075 8 0.084323 0.099456 0.0055868  
## 10 0.010000 9 0.071248 0.081189 0.0048498

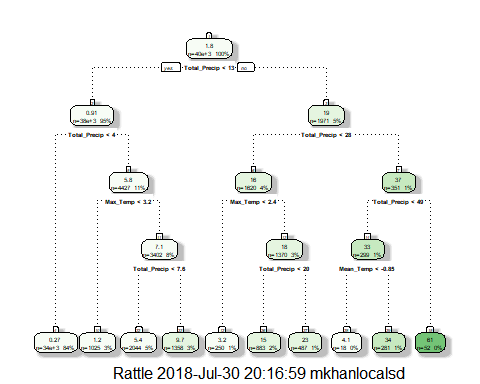
# Get the optimal CP programmatically...  
min.xerror <- rt$cptable[which.min(rt$cptable[,"xerror"]),"CP"]  
min.xerror

## [1] 0.01

# 0.01  
#Plot the Prune Tree #  
fancyRpartPlot(rt)



# ...and use it to prune the tree  
rt.pruned <- prune(rt,cp = min.xerror)   
  
#Plot the Prune Tree #  
fancyRpartPlot(rt.pruned)



# Evaluate the new pruned tree on the test set  
test.pred.rtree.p <- predict(rt.pruned,test)  
RMSE.rtree.pruned <- sqrt(mean((test.pred.rtree.p-test$Total\_Rain)^2))  
RMSE.rtree.pruned

## [1] 1.627743

# 1.627743  
  
MAE.rtree.pruned <- mean(abs(test.pred.rtree.p-test$Total\_Rain))  
MAE.rtree.pruned

## [1] 0.6907401

# 0.6907401

2.4 Applied Random Forest Alogorithm

##-- apply the random forest here ##  
library(randomForest)  
set.seed(123)  
  
# Create a random forest with 1000 trees  
rf <- randomForest(Total\_Rain ~ Month+ Max\_Temp + Min\_Temp + Mean\_Temp+Total\_Precip+  
 Heat\_Deg\_Days, data = train, importance = TRUE, ntree=1000)  
   
# Find out how many trees are needed to reach the minimum error estimate?   
which.min(rf$mse)

## [1] 488

#[1] 93  
  
  
# Using the importance() function to calculate the importance of each variable  
imp <- as.data.frame(sort(importance(rf)[,1],decreasing = TRUE),optional = T)  
names(imp) <- "% Inc MSE"  
imp

## % Inc MSE  
## Total\_Precip 539.786101  
## Month 19.267820  
## Max\_Temp 13.421982  
## Min\_Temp 13.055725  
## Heat\_Deg\_Days 10.926238  
## Mean\_Temp 9.205103

# As usual, predict and evaluate on the test set  
test.pred.forest <- predict(rf,test)  
RMSE.forest <- sqrt(mean((test.pred.forest-test$Total\_Rain)^2))  
RMSE.forest

## [1] 1.006345

#1.006345  
   
MAE.forest <- mean(abs(test.pred.forest-test$Total\_Rain))  
MAE.forest

## [1] 0.2182462

#0.2182462  
  
  
#The Total\_Precip was, once again, considered the most important predictor;   
#it is estimated that, in the absence of that variable, the error would increase.

1. Model Assesment and Comparison

3.1 Create a data frame with the error metrics for each method and Compare

accuracy <- data.frame(Method = c("Baseline","Linear Regression","Full tree","Pruned tree","Random forest"),  
 RMSE = c(RMSE.baseline,RMSE.lin.reg,RMSE.rtree,RMSE.rtree.pruned,RMSE.forest),  
 MAE = c(MAE.baseline,MAE.lin.reg,MAE.rtree,MAE.rtree.pruned,MAE.forest))   
  
  
# Round the values and print the table  
accuracy$RMSE <- round(accuracy$RMSE,2)  
accuracy$MAE <- round(accuracy$MAE,2)   
accuracy

## Method RMSE MAE  
## 1 Baseline 5.38 2.84  
## 2 Linear Regression 459.41 10.05  
## 3 Full tree 1.63 0.69  
## 4 Pruned tree 1.63 0.69  
## 5 Random forest 1.01 0.22

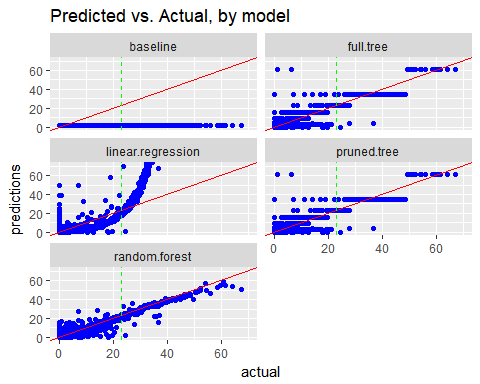
# Create a data frame with the predictions for each method  
all.predictions <- data.frame(actual = test$Total\_Rain,  
 baseline = best.guess,  
 linear.regression = test.pred.lin,  
 full.tree = test.pred.rtree,  
 pruned.tree = test.pred.rtree.p,  
 random.forest = test.pred.forest)  
  
  
#head(all.predictions)  
  
  
# Needed to melt the columns with the gather() function   
# tidyr is an alternative to the reshape2 package (see the end of Part3a)   
library(tidyr)

## Warning: package 'tidyr' was built under R version 3.4.4

##   
## Attaching package: 'tidyr'

## The following object is masked from 'package:RCurl':  
##   
## complete

# Gather the prediction variables (columns) into a single row (i.e., wide to long)  
# Recall the ggplot2 prefers the long data format  
all.predictions <- gather(all.predictions,key = model,value = predictions,2:6)  
   
#head(all.predictions)  
  
#tail (all.predictions)  
  
  
  
#Predicted vs. actual for each model  
ggplot(data = all.predictions,aes(x = actual, y = predictions)) +   
 geom\_point(colour = "blue") +   
 geom\_abline(intercept = 0, slope = 1, colour = "red") +  
 geom\_vline(xintercept = 23, colour = "green", linetype = "dashed") +  
 facet\_wrap(~ model,ncol = 2) +   
 coord\_cartesian(xlim = c(0,70),ylim = c(0,70)) +  
 ggtitle("Predicted vs. Actual, by model")



1. Logistics Regression Applied

4.1 Apply Logistics Regression to same dataset and see how it behaves

## Start Sampling Data here for Algorithim application ##  
# randomly pick 70% of the number of observations (365)  
set.seed(1234)  
  
#LR\_Data  
LR\_Data <- Actual\_Data\_Final[c("Date","Year","Month","Max\_Temp","Min\_Temp","Mean\_Temp","Heat\_Deg\_Days","Total\_Precip","Total\_Rain","Rained")]  
  
#LR\_Data$Month <- as.factor(LR\_Data$Month)  
#check cross tabulation of the Data  
#xtabs(~Rained+Month, data=LR\_Data)  
  
  
#str(LR\_Data)  
index <- sample(1:nrow(LR\_Data),size = 0.7\*nrow(LR\_Data),replace = TRUE)   
  
# subset weather to include only the elements in the index  
train <- LR\_Data[index,]   
  
# subset weather to include all but the elements in the index  
test <- LR\_Data [-index,]   
  
#check the row counts for test and train data  
nrow(train)

## [1] 40140

nrow(test)

## [1] 28513

# ----Apply Model here ---- #  
#Note: Because of fitted probabilities numerically 0 or 1 occured warning, i go step by step and learned that Total\_Precip is causing an issue. Therefore i am removing this variable from formulla   
  
model <- glm(Rained ~ Max\_Temp + Min\_Temp + Mean\_Temp + Heat\_Deg\_Days , data = train, family = binomial)  
  
# --- check Model Summary --- #  
summary(model)

##   
## Call:  
## glm(formula = Rained ~ Max\_Temp + Min\_Temp + Mean\_Temp + Heat\_Deg\_Days,   
## family = binomial, data = train)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.4821 -0.7639 -0.5643 -0.2423 2.9835   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 2.033441 0.154164 13.19 <2e-16 \*\*\*  
## Max\_Temp -2.690172 0.228631 -11.77 <2e-16 \*\*\*  
## Min\_Temp -2.371862 0.228899 -10.36 <2e-16 \*\*\*  
## Mean\_Temp 4.994315 0.458202 10.90 <2e-16 \*\*\*  
## Heat\_Deg\_Days -0.153161 0.008587 -17.84 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 42771 on 40139 degrees of freedom  
## Residual deviance: 39001 on 40135 degrees of freedom  
## AIC: 39011  
##   
## Number of Fisher Scoring iterations: 5

model

##   
## Call: glm(formula = Rained ~ Max\_Temp + Min\_Temp + Mean\_Temp + Heat\_Deg\_Days,   
## family = binomial, data = train)  
##   
## Coefficients:  
## (Intercept) Max\_Temp Min\_Temp Mean\_Temp Heat\_Deg\_Days   
## 2.0334 -2.6902 -2.3719 4.9943 -0.1532   
##   
## Degrees of Freedom: 40139 Total (i.e. Null); 40135 Residual  
## Null Deviance: 42770   
## Residual Deviance: 39000 AIC: 39010

4.2 Prediction and Misclassificaiton

4.2.1 Prediction and Misclassification for Training Data

## Validation Starts here for Train Data ##  
##--- Prediction and Misclassification for Training Data   
predicted\_values <- predict(model, train,type = "response")  
  
prediction\_train <- data.frame("RAIN" = c(1:nrow(train)))  
prediction\_train$RAIN <- "1"  
  
#Before deciding the cutt off values, let see the optimum cut off value.  
glm.tune(model, train)

## cutoff accuracy specificity  
## 1 0.02 0.2355506 0.998558599  
## 2 0.04 0.2667912 0.989799313  
## 3 0.06 0.3027653 0.979820379  
## 4 0.08 0.3396612 0.967956536  
## 5 0.10 0.3833582 0.952988136  
## 6 0.12 0.4216243 0.931034483  
## 7 0.14 0.4612606 0.899878035  
## 8 0.16 0.5012706 0.859407917  
## 9 0.18 0.5402591 0.816942011  
## 10 0.20 0.5745142 0.769819270  
## 11 0.22 0.6133284 0.710832687  
## 12 0.24 0.6428500 0.652289611  
## 13 0.26 0.6739661 0.586428651  
## 14 0.28 0.7002990 0.526444173  
## 15 0.30 0.7211261 0.464685664  
## 16 0.32 0.7353014 0.395498392  
## 17 0.34 0.7490284 0.340170751  
## 18 0.36 0.7551071 0.293380641  
## 19 0.38 0.7634280 0.242266327  
## 20 0.40 0.7704285 0.199356913  
## 21 0.42 0.7751868 0.157888901  
## 22 0.44 0.7766816 0.129726134  
## 23 0.46 0.7779023 0.108548620  
## 24 0.48 0.7781764 0.083157778  
## 25 0.50 0.7787245 0.061647633  
## 26 0.52 0.7789736 0.048120634  
## 27 0.54 0.7775536 0.032708726  
## 28 0.56 0.7772048 0.022286285  
## 29 0.58 0.7768560 0.015522785  
## 30 0.60 0.7762830 0.009424548  
## 31 0.62 0.7756602 0.004878590  
## 32 0.64 0.7755107 0.001884910

prediction\_train$RAIN[ predicted\_values < 0.50] <- "0"  
  
prediction\_train$RAIN <- as.factor(prediction\_train$RAIN)  
  
p\_tab\_train<- table(prediction\_train$RAIN, train$Rained)  
#Confusion Matrix  
tab\_train <-confusionMatrix(prediction\_train$RAIN, train$Rained)  
  
tab\_train

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 30702 8463  
## 1 419 556  
##   
## Accuracy : 0.7787   
## 95% CI : (0.7746, 0.7828)  
## No Information Rate : 0.7753   
## P-Value [Acc > NIR] : 0.05111   
##   
## Kappa : 0.0705   
## Mcnemar's Test P-Value : < 2e-16   
##   
## Sensitivity : 0.98654   
## Specificity : 0.06165   
## Pos Pred Value : 0.78391   
## Neg Pred Value : 0.57026   
## Prevalence : 0.77531   
## Detection Rate : 0.76487   
## Detection Prevalence : 0.97571   
## Balanced Accuracy : 0.52409   
##   
## 'Positive' Class : 0   
##

#Misclassification Error#  
(Misclassificatoin\_Error <- 1-sum(diag(p\_tab\_train))/sum(p\_tab\_train))

## [1] 0.2212755

4.2.2 Prediction and Misclassification for Testing Data

##--- Prediction and Misclassification for Testing Data   
predicted\_values <- predict(model, test, type = "response")  
prediction\_test <- data.frame(c(1:nrow(test)))  
colnames(prediction\_test) <- c("RAIN")  
  
  
prediction\_test$RAIN <- "1"  
  
prediction\_test$RAIN[ predicted\_values < 0.50] <- "0"  
   
prediction\_test$RAIN <- as.factor(prediction\_test$RAIN)  
   
p\_tab\_test<- table(prediction\_test$RAIN, test$Rained)  
tab\_test <-confusionMatrix(prediction\_test$RAIN, test$Rained)  
tab\_test

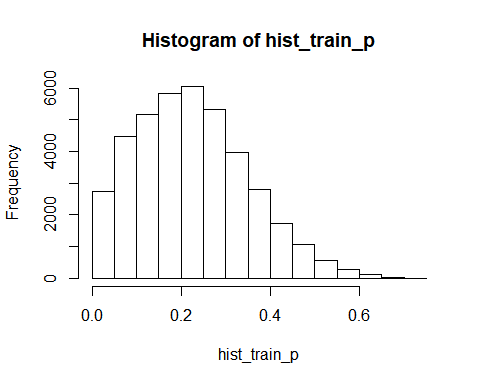
## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 21953 5885  
## 1 308 367  
##   
## Accuracy : 0.7828   
## 95% CI : (0.778, 0.7876)  
## No Information Rate : 0.7807   
## P-Value [Acc > NIR] : 0.2013   
##   
## Kappa : 0.0661   
## Mcnemar's Test P-Value : <2e-16   
##   
## Sensitivity : 0.9862   
## Specificity : 0.0587   
## Pos Pred Value : 0.7886   
## Neg Pred Value : 0.5437   
## Prevalence : 0.7807   
## Detection Rate : 0.7699   
## Detection Prevalence : 0.9763   
## Balanced Accuracy : 0.5224   
##   
## 'Positive' Class : 0   
##

1-sum(diag(p\_tab\_test))/sum(p\_tab\_test)

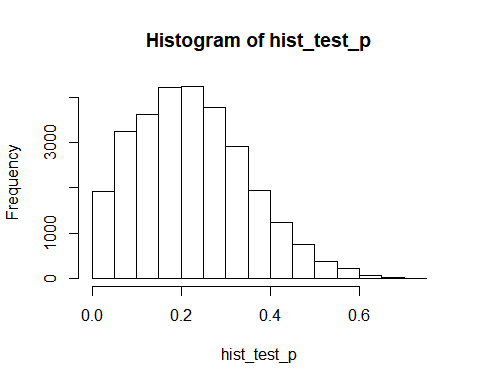
## [1] 0.2171992

f 4.2.3 Visulaize the Prediction

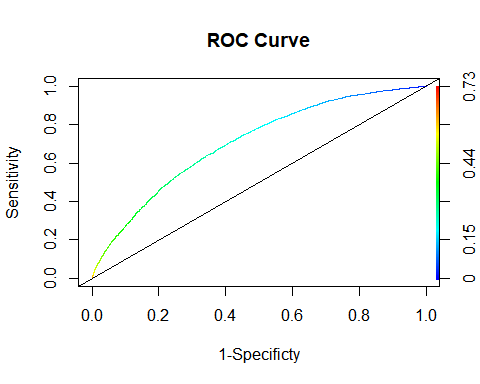
## Check the histogram of Probability for Training and Testing Data  
hist\_train\_p <- predict(model,train,type="response")  
hist\_test\_p <- predict(model,test,type="response")  
hist(hist\_train\_p)



hist(hist\_test\_p)



#ROC Curve  
#True positive rate   
 # 1- Sensitivity  
 # 2- specificity  
  
p <- predict(model,train,type="response")  
pred <- prediction(p, train$Rained)  
roc <-performance(pred,"tpr","fpr")  
plot(roc, colorize=T, main="ROC Curve", xlab="1-Specificty", ylab = "Sensitivity")  
# make a line  
abline(a=0,b=1)



## Check the chances of Rain ##  
chance\_of\_rain <- function(model, data\_record){  
 chance\_frac <- predict(model, data\_record, type="response") ## "1" = "Yes"  
 paste(round(chance\_frac\*100), "%", sep="")  
}  
  
chance\_of\_rain(model, test[1:10,])

## [1] "4%" "19%" "19%" "18%" "16%" "30%" "17%" "22%" "17%" "5%"

chance\_of\_rain(model, test[100:110,])

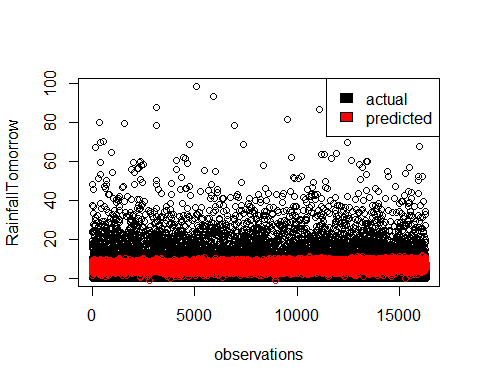
## [1] "15%" "13%" "18%" "37%" "28%" "34%" "39%" "24%" "16%" "26%" "22%"

1. Apply linear Regression only To Rain Data when Rain is happened, it means Rained = 1

LR\_Data\_For\_LM = LR\_Data[LR\_Data$Total\_Rain > 0,]  
rf\_fit <- lm(Total\_Rain ~ Min\_Temp + Max\_Temp + Mean\_Temp + Heat\_Deg\_Days -1, data = LR\_Data\_For\_LM)  
summary(rf\_fit)

##   
## Call:  
## lm(formula = Total\_Rain ~ Min\_Temp + Max\_Temp + Mean\_Temp + Heat\_Deg\_Days -   
## 1, data = LR\_Data\_For\_LM)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -11.244 -4.970 -2.718 2.123 90.145   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## Min\_Temp -0.69214 1.20098 -0.576 0.564   
## Max\_Temp -1.32560 1.19867 -1.106 0.269   
## Mean\_Temp 2.50429 2.39554 1.045 0.296   
## Heat\_Deg\_Days 0.42993 0.01017 42.258 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 8.314 on 16295 degrees of freedom  
## Multiple R-squared: 0.3808, Adjusted R-squared: 0.3807   
## F-statistic: 2506 on 4 and 16295 DF, p-value: < 2.2e-16

lm\_pred <- predict(rf\_fit, LR\_Data\_For\_LM)  
plot(x = seq\_along(LR\_Data\_For\_LM$Total\_Rain), y = LR\_Data\_For\_LM$Total\_Rain, type='p', xlab = "observations", ylab = "RainfallTomorrow")  
legend("topright", c("actual", "predicted"), fill = c("black", "red"))  
points(x = seq\_along(LR\_Data\_For\_LM$Total\_Rain), y = lm\_pred, col='red')



1. Final Report

weather\_report <- function(today\_record, rain\_tomorrow\_model, cutoff) {  
 # RainTomorrow prediction  
 rainTomorrow\_prob <- predict(rain\_tomorrow\_model, today\_record, type="response")  
 rainTomorrow\_pred = ifelse(rainTomorrow\_prob >= cutoff, "1", "0")  
   
 # Rainfall prediction iff RainTomorrow prediction is Yes; chance of rain probability  
 rainfall\_pred <- NA  
 chance\_of\_rain <- NA  
 if (rainTomorrow\_pred == "1") {  
 rainfall\_pred <- round(predict(rf\_fit, today\_record), 1)  
 chance\_of\_rain <- round(rainTomorrow\_prob\*100)  
 }  
   
 # converting all numeric predictions to strings  
 if (is.na(rainfall\_pred)) {  
 rainfall\_pred\_str <- "< 1 mm"  
 } else {  
 rainfall\_pred\_str <- paste(rainfall\_pred, "mm", sep = " ")  
 }  
   
 if (is.na(chance\_of\_rain)) {  
 chance\_of\_rain\_str <- ""  
 } else {  
 chance\_of\_rain\_str <- paste(chance\_of\_rain, "%", sep="")  
 }  
   
   
   
 report <- data.frame(Rainfall = rainfall\_pred\_str,ChanceOfRain = chance\_of\_rain\_str)  
 report  
}  
  
(tomorrow\_report <- weather\_report(LR\_Data[73,], model, 0.25))

## Rainfall ChanceOfRain  
## 1 6.1 mm 25%

(tomorrow\_report <- weather\_report(LR\_Data[32,], model, 0.25))

## Rainfall ChanceOfRain  
## 1 < 1 mm

(tomorrow\_report <- weather\_report(LR\_Data[50,], model, 0.25))

## Rainfall ChanceOfRain  
## 1 < 1 mm

(tomorrow\_report <- weather\_report(LR\_Data[100,], model, 0.25))

## Rainfall ChanceOfRain  
## 1 < 1 mm

(tomorrow\_report <- weather\_report(LR\_Data[115,], model, 0.25))

## Rainfall ChanceOfRain  
## 1 5.6 mm 25%

(tomorrow\_report <- weather\_report(LR\_Data[253,], model, 0.25))

## Rainfall ChanceOfRain  
## 1 7.4 mm 56%

(tomorrow\_report <- weather\_report(LR\_Data[311,], model, 0.25))

## Rainfall ChanceOfRain  
## 1 < 1 mm