Packages to include

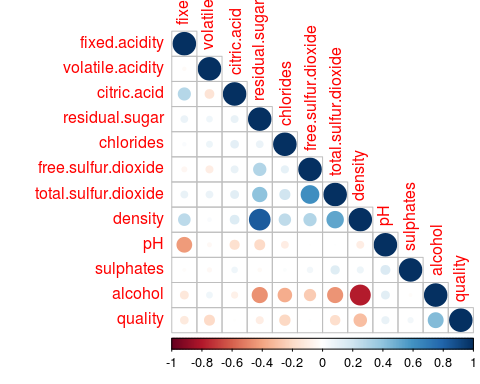
setwd("/home/roshan/workspace/R Studio/R-Pr/")  
library(tensorflow)  
use\_virtualenv("/home/roshan/.virtualenvs/r-tensorflow")  
library(reticulate)  
wn <- import("warnings")  
wn$filterwarnings("ignore")  
library(keras)  
library(caTools)  
library(ggplot2)  
library(corrplot)

Import Data Set

data <- read.csv(file = "wineQualityWhites.csv", sep = ",", header = TRUE)

Feature Selection using Correlation Plots

cr <- cor(data[c(-1)])  
corrplot(cr, type = "lower")



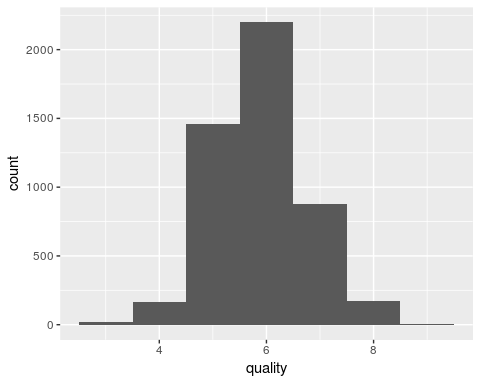
Find Quality Levels

data$quality.factor <- as.factor(data$quality)  
  
levels(data$quality.factor)

## [1] "3" "4" "5" "6" "7" "8" "9"

Histogram of Quality

ggplot(data = data, aes(x = quality)) +  
 geom\_histogram(binwidth = 1)



Add Binary Ordered Quality Attribute

for(i in 1:nrow(data)){  
 if(data$quality[[i]] < 6){  
 data$quality.order[[i]] <- "Bad"  
 }else{  
 data$quality.order[[i]] <- "Good"  
 }   
}

Count

i <- j <- 0  
for(q in data$quality.order){  
 if(q == "Bad"){  
 i <- i+1   
 }else{  
 j <- j+1  
 }  
}  
print(i)

## [1] 1640

print(j)

## [1] 3258

data$quality.order <- factor(factor(data$quality.order), levels = c("Bad", "Good"))

Convert Quality To Binary Attribute

for(i in 1:nrow(data)){  
 if(data$quality.order[[i]] == "Bad"){  
 data$quality.num[[i]] <- 0  
 }else{  
 data$quality.num[[i]] <- 1  
 }   
}

Train and Testing Data Sets

value <- sample.split(data$X, SplitRatio = 0.7)  
train.data <- subset(data, value == TRUE)  
test.data <- subset(data, value == FALSE)  
  
write.csv(train.data, file = "train\_data.csv")  
write.csv(test.data, file = "test\_data.csv")

Loading Existing training and testing data sets

train.data <- read.csv(file = "train\_data.csv", sep = ",", header = TRUE)   
test.data <- read.csv(file = "test\_data.csv", sep = ",", header = TRUE)   
  
train.data <- train.data[,-1]  
test.data <- test.data[,-1]  
  
x\_train <- as.matrix(train.data[,c(2,3,5,6,8:12)])  
y\_train <- train.data[,16]  
x\_test <- as.matrix(test.data[,c(2,3,5,6,8:12)])  
y\_test <- test.data[,16]  
  
y\_train <- to\_categorical(y\_train,2)  
y\_test <- to\_categorical(y\_test,2)

Creating Model

model <- NULL  
model <- keras\_model\_sequential()  
model %>%  
 layer\_dense(units = 256, input\_shape = c(9), activation = "relu") %>%  
 layer\_dropout(0.2) %>%  
 layer\_dense(units = 128, activation = "relu") %>%  
 layer\_dropout(0.3) %>%   
 layer\_dense(units = 2, activation = "softmax")

Compile the Model

model %>% compile(  
 loss = "binary\_crossentropy",  
 optimizer = optimizer\_adam(0.001),  
 metrics = c('accuracy')  
)

Train the Model

model %>% fit(  
 x\_train, y\_train,   
 validation\_split = 0.3,   
 epochs = 100,   
 batch\_size = 87  
)  
  
save\_model\_hdf5(model, "model\_name.h5")

Load the data model

model <- load\_model\_hdf5("model8\_tracc\_0.7850.h5")

Evaluate the Quality of Wine for testing data set

model %>% evaluate(x\_test,y\_test)

## $loss  
## [1] 0.5228955  
##   
## $acc  
## [1] 0.785034

Predict Values and plot Confusion Matrix

pred <- model %>% predict\_classes(x\_test)  
table(Actual = test.data[,16], Predicted = pred)

## Predicted  
## Actual 0 1  
## 0 299 193  
## 1 123 855