#

# 1. Downloading Installing and Starting R

# 1.1 Installing the DataSet

#The url can be found here: https://www.kaggle.com/lucidlenn/sloan-digital-sky-survey/version/1

t1 <- Sys.time()

# 1.2 Libraries used

library(tidyverse)

library(data.table)

library(caret)

library(ggplot2)

library(ggcorrplot)

library(RSNNS)

library(randomForest)

library(ggcorrplot)

library(kernlab)

library(cowplot)

# 2. Load The Data

list.files(path = ".") # list files on the current working directory

sky.df <- fread(file = "Skyserver\_SQL2\_27\_2018 6\_51\_39 PM.csv", sep=",")

# 2.3. Create a Validation Dataset

# create a list of 80% of the rows in the original dataset we can use for training

index <- createDataPartition(sky.df$class, p=0.8, list=FALSE)

# select 20% of the data for validation

validation <- sky.df[-index,]

# use the remaining 80% of data to training and testing the models

sky.train <- sky.df[index,]

# 3. Summarize Dataset

str(sky.train)

summary(sky.train)

colSums(is.na(sky.train)) # any NA's?

dim(sky.train)

# 3.1 Types of attributes

sapply(sky.train, class)

# The "class" column is our response variable. Since this is a classification problem, we

# will transform it in a factor with three levels

sky.train$class <- as.factor(sky.train$class)

levels(sky.train$class)

validation$class <- as.factor(validation$class)

levels(validation$class)

# 3.2 Class distribution

# summarize the class distribution

percentage <- prop.table(table(sky.train$class)) \* 100

cbind(freq=table(sky.train$class), percentage=percentage)

# 3.3 Statistical Summary

summary(sky.train)

# We can observe two different things here that data preparation tells us:

# the class distribution is not even and this could be solved using the SMOTE function

# that equalizes the classes.

# The numeric columns are not on the same scale.

# Initially we decided to explore the data "as it is" and later, when we build the models

# can apply SMOTE and normalixze the data.

# 4 Visualize Dataset

# 4.1 Univariate Plots

# The Thuan-Gunn astronomic magnitude system. u, g, r, i, z

# represent the response of the 5 bands of the telescope.

thuan\_gunn <- c("u", "g", "r", "i", "z")

# field are features which describe a field within an image taken by the SDSS.

#A field is basically a part of the entire image corresponding to 2048 by 1489 pixels.

field\_feat <- c("run", "rerun", "camcol", "field")

# Right ascension (abbreviated RA) is the angular distance measured eastward along

# the celestial equator from the Sun at the March equinox to the hour circle of the

# point above the earth in question. When paired with declination (abbreviated dec),

# these astronomical coordinates specify the direction of a point on the celestial

# sphere (traditionally called in English the skies or the sky) in the equatorial

# coordinate system.

skies <- c("ra", "dec")

# remaining features

equipment\_feat <- c("redshift", "plate", "mjd", "fiberid") # These features are related to the measurement equipment

# split input and output

x <- sky.train[,-c(1, 13, 14, 16, 17)]

y <- sky.train$class

ggplot(sky.train, aes(class, fill = class)) +

geom\_bar()

# We need a different approach now: normalize the data to have all the numeric features

# between 0 and 1 to be able to compare them and understand better the data.

set.seed(2205)

sky.train.norm <- normalizeData(sky.train[,-c(1, 13, 14)], type = "0\_1") # using the package 'RSNNS'

sky.train.norm <- as.data.frame(sky.train.norm)

summary(sky.train.norm) # check the normalization

names\_sky.train <- names(sky.train[,-c(1, 13, 14)]) # add the names non-numeric columns back to the df.

names(sky.train.norm) <- names\_sky.train

head(sky.train.norm, 2) # now we check if it worked. That seems OK.

# now we add back the columns that were not included in the normalization.

sky.train.norm <- add\_column(sky.train.norm, objid = sky.train$objid)

sky.train.norm <- add\_column(sky.train.norm, specobjid = sky.train$specobjid)

sky.train.norm <- add\_column(sky.train.norm, class = sky.train$class)

head(sky.train.norm, 2)

# continuing with the normalized data

x <- sky.train.norm[,-c(9, 16:18)]

y <- sky.train.norm$class

featurePlot(x = sky.train.norm[, c("u", "g", "r", "i", "z")], y = y, plot="box", main = "Thuan\_gunn Group")

featurePlot(x = sky.train.norm[, c("run", "camcol", "field")], y = y, plot="box", main = "Field Features (excluded 'rerun')")

featurePlot(x = sky.train.norm[, c("redshift", "plate", "mjd", "fiberid")], y = y, plot="box", main = "Equipment Features")

featurePlot(x = sky.train.norm[, c("ra", "dec")], y = y, plot="box", main = "Skies / Sky Feature")

# This is useful to see that there are clearly different distributions of the

# attributes for each class value and to identify the outliers (noise).

# There seems to be great variability in the 'mjd' and 'plate' parameters.

# The variability found in the 'Thuann\_gunn' group will be kept 'as it is' and we will deal with

# it only in case we need to improve our model.

# Let's run a model with randomForest to check the variables importance

set.seed(2205)

rf.sky.train <- randomForest(class ~ ., data = sky.train[, -c(1, 10, 13, 16, 17)])

imp.df <- importance(rf.sky.train) # importance of the features

imp.df <- data.frame(features = row.names(imp.df), MDG = imp.df[,1])

imp.df <- imp.df[order(imp.df$MDG, decreasing = TRUE),]

ggplot(imp.df, aes(x = reorder(features, MDG), y = MDG, fill = MDG)) +

geom\_bar(stat = "identity") + labs (x = "Features", y = "Mean Decrease Gini (MDG)") +

coord\_flip()

# We have exclude plate and mjd features.

# Plate: Each spectroscopic exposure employs a large, thin, circular metal "plate"

# that positions optical fibers via holes drilled at the locations of the

# images in the telescope focal plane.

# These fibers then feed into the spectrographs.

# Each plate has a unique serial number, which is called plate in views such as SpecObj in the CAS.

## The plate seems to have importance when predicting. This could be a noisy parameter since we

# have different plates measuring the waves.

# MJD: Modified Julian Date, used to indicate the date that a given piece of SDSS data

# (image or spectrum) was taken.

#Let's have a look in the PCA analysis.(we are excluding 2 constant features and the factors)

PCA.sky.train <- prcomp(sky.train[, -c(1, 10, 13, 14)])

summary(PCA.sky.train)

# We notice that the first 6 components respond by 99.99% of the data.

# Considering that the number of features is not so big, we can consider using all the features

# aleready considered for the PCA analysis or use the first 6 features.

## biplot(PCA.sky.train, col = c('red', 'blue'))

# ALthough our suggestion here should be to discard the features obtained from the MDG plot that

# do not contribute to the model accuracy based on the MDG definition:

# "Because Random Forests are an ensemble of individual Decision Trees,

# Gini Importance can be leveraged to calculate Mean Decrease in Gini,

# which is a measure of variable importance for estimating a target variable.

# Mean Decrease in Gini is the average (mean) of a variable’s total decrease

# in node impurity, weighted by the proportion of samples reaching that node in

# each individual decision tree in the random forest. This is effectively a measure

# of how important a variable is for estimating the value of the target variable

# across all of the trees that make up the forest. A higher Mean Decrease in Gini

# indicates higher variable importance. Variables are sorted and displayed in the

# Variable Importance Plot created for the Random Forest by this measure.

# The most important variables to the model will be highest in the plot

# and have the largest Mean Decrease in Gini Values, conversely, the least

# important variable will be lowest in the plot, and have the smallest Mean Decrease in Gini values.

# We have decided to keep all the numeric variables since we have not so many variables that could

# impact on the computational time.

rownames(imp.df) <- NULL

imp.df %>% knitr::kable(caption = "Importance")

# defining the dataset for the model:

sky.model <- sky.train[, -c(1, 10, 13)]

# We can also confirm the importance of the features in this correlation plot:

corr <- round(cor(sky.model[, -7], use = "complete.obs"), 2)

ggcorrplot(corr, hc.order = TRUE,

type = "lower",

lab = TRUE,

lab\_size = 3,

method="circle",

colors = c("red1", "honeydew", "green2"),

title="Correlation of Numeric Features",

ggtheme=theme\_bw)

# Based on the PCA analysis, on the correlation of the numeric features, and on the MDG values

# we decided to select c("redshift", "z", "i", "g", "r", "u") as the most important

# features that explain 99.97% of the dataset.

sky.model <- sky.train[,c("redshift", "z", "i", "g", "r", "u", "class")] # selecting the columns we need

# sky.model <- sky.train[,c("dec", "run", "field", "ra", "z", "i", "g", "r", "u", "class")]

# This was a different approach without the "redshift" feature were we did not achieved the desired

# outcome because the feature has a considerable impact on the dataset.

theme1<- theme(axis.text.x = element\_text(angle = 90, hjust = 1, vjust = 0.5),

legend.position="top")

theme2<- theme(axis.text.x = element\_text(angle = 90, hjust = 1, vjust = 0.5),

legend.position="none")

# a different visualization

# about the interaction between "redshift" and the other selected features

# (note: for better visualization we used the normalized dataset to make it easier the comparison)

theme1<- theme(axis.text.x = element\_text(angle = 90, hjust = 1, vjust = 0.5),

legend.position="top")

theme2<- theme(axis.text.x = element\_text(angle = 90, hjust = 1, vjust = 0.5),

legend.position="none")

plot\_grid(ggplot(sky.train.norm, aes(z, redshift, col = class)) + geom\_point() + theme1,

ggplot(sky.train.norm, aes(i, redshift, col = class)) + geom\_point() + theme1,

ggplot(sky.train.norm, aes(g, redshift, col = class)) + geom\_point() + theme1,

ggplot(sky.train.norm, aes(r, redshift, col = class)) + geom\_point() + theme1,

ggplot(sky.train.norm, aes(u, redshift, col = class)) + geom\_point() + theme1,

align = "h") # Thuann\_Gunn group

# We can see on the plot above that the "redshift"feature is important, especially

# in the low wave length values showing a higher concentration forthe QSO making

# this feature an important observation for the "class" of the objetc to be predicted.

# another comprison between redshift and features that have shown to be correlated in the

# previous analysis that have not been considered for the final model but those features have shown

# some correlation with the redshift feature:

plot\_grid(ggplot(sky.train.norm, aes(dec, redshift, col = class)) + geom\_point() + theme1,

ggplot(sky.train.norm, aes(run, redshift, col = class)) + geom\_point() + theme1,

ggplot(sky.train.norm, aes(field, redshift, col = class)) + geom\_point() + theme1,

ggplot(sky.train.norm, aes(ra, redshift, col = class)) + geom\_point() + theme1,

ggplot(sky.train.norm, aes(plate, redshift, col = class)) + geom\_point() + theme1,

align = "h")

# In all these visualization we have noted the data "redshift" is concentrated for the GALAXY

# and STAR classes and that for the QSO class it covers a broader range showing that the data

# has a greater variability when we consider it as a QSO class feature.

# Despite the small class imbalance we considered this not to be enough difference

# to make use of the SMOTE library from the 'DMwR' package to equalize the classes proportion.

# 5 Evaluate some algorithms

# 5.1 Testing Harness

# Run algorithms using 20-fold cross validation

control <- caret::trainControl(method="cv", number=20)

metric <- "Accuracy"

# 5.2 Build Models

# a) linear algorithms

set.seed(2205)

fit.lda <- caret::train(class ~ . , data=sky.model, method="lda", metric=metric, trControl=control)

# b) nonlinear algorithms

# CART

set.seed(2205)

fit.cart <- caret::train(class ~ . , data=sky.model, method="rpart", metric=metric, trControl=control)

# kNN

set.seed(2205)

fit.knn <- caret::train(class ~ . , data=sky.model, method="knn", metric=metric, trControl=control)

# c) advanced algorithms

# SVM

set.seed(2205)

fit.svm <- caret::train(class ~ . , data=sky.model, method="svmRadial", metric=metric, trControl=control)

# Random Forest

set.seed(2205)

fit.rf <- caret::train(class ~ . , data=sky.model, method="rf", metric=metric, trControl=control)

# 5.3 Select best model

# summarize accuracy of models

results <- resamples(list(lda=fit.lda, cart=fit.cart, knn=fit.knn, svm=fit.svm,

rf=fit.rf))

summary(results)

# compare accuracy of models

dotplot(results)

print(fit.rf) # 0.9915019

plot(fit.rf)

plot(varImp(fit.rf))

# 6 Make Predictions

# estimate skill of RF on the validation dataset

# columns to be discarded:

validation.model <- validation[,c("redshift", "z", "i", "g", "r", "u", "class")]

set.seed(2205)

predictions <- predict(fit.rf, validation.model)

caret::confusionMatrix(predictions, validation.model$class) # 0.992

# Confusion Matrix and Statistics

#Reference

#Prediction GALAXY QSO STAR

#GALAXY 992 3 5

#QSO 6 166 0

#STAR 1 1 825

# The difference between the training and test (validation) sets is 0.088%

# The conclusion we came to is that the validation set cases are not so hard to predict and the

# data is well clustered around the classes.

# We tried to discard the "redshift"feature in order to maximize the weight of the other features

# in previuos models but we only achieved an Accuracy = 0.9305 in the train set and Accuracy = 0.9235

# in the validation set with many cases of False Positives, 10% FP whem predicting STAR (predicted GALAXY)

# and 9% FP when predicting QSO in relation to the other two classes.

# The decision is to keep the "redshift" feature along with the Thuann\_gunn group.

t2 <- Sys.time()

print(t2 - t1) # Time difference of 5.356926 mins