Classification of Stars, Galaxies and Quasars

Enio Linhares Junior

5/19/2019

Contents

1 Introduction

1.1 Content

The Sloan Digital Sky Survey (SDSS) offers public data of space observations and the task here is to build a model that is able to predict the different classes of objects (Stars, Galaxies and Quasars) based on the data acquired through the scientific equipment. The data consists of 10,000 observations of space taken by the SDSS. Every observation is described by 17 feature columns and 1 class column which identifies it to be either a star, galaxy or quasar.
Our model achieved an **accuray** of 0.9915019

1.2 Feature Description

The table results from a query which joins two tables (actuaclly views): "PhotoObj" which contains photometric data and "SpecObj" which contains spectral data.

During our data exploratory analysis we will be explaining the features as they appear.

The data released by the SDSS is under public domain. Its taken from the current data release RD14.

2 Downloading Installing and Starting R

2.1 Installing the DataSet

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

2.2 Libraries used

```
if(!require(tidyverse)) install.packages("tidyverse", repos = "http://cran.us.r-project.org")
if(!require(caret)) install.packages("caret", repos = "http://cran.us.r-project.org")
if(!require(data.table)) install.packages("data.table", repos = "http://cran.us.r-project.org")
if(!require(ggplot2)) install.packages("ggplot2", repos = "http://cran.us.r-project.org")
```

```
if(!require(ggcorrplot)) install.packages("ggcorrplot", repos = "http://cran.us.r-project.org")
if(!require(RSNNS)) install.packages("RSNNS", repos = "http://cran.us.r-project.org")
if(!require(randomForest)) install.packages("randomForest", repos = "http://cran.us.r-project.org")
if(!require(kernlab)) install.packages("kernlab", repos = "http://cran.us.r-project.org")
if(!require(cowplot)) install.packages("cowplot", repos = "http://cran.us.r-project.org")
```

```
## Warning: package 'cowplot' was built under R version 3.5.2
```

```
library(tidyverse)
library(data.table)
library(caret)
library(ggplot2)
library(ggcorrplot)
library(RSNNS)
library(randomForest)
library(ggcorrplot)
library(kernlab)
library(cowplot)
```

3 Load The Data

```
sky.df <- fread(file = "Skyserver_SQL2_27_2018 6_51_39 PM.csv", sep=",") # load the data and save for u
```

3.1 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,]
```

4 Summarize Dataset

4.1 Structure, summary, NA's check, and dimensions

```
str(sky.train)
```

```
## Classes 'data.table' and 'data.frame':   8001 obs. of  18 variables:
##  $ objid    :integer64 1237648704577142859 1237648704577273907 1237648704577273970 1237648704577273995
##  $ ra       : num  184 184 184 184 184 ...
##  $ dec      : num  0.1353 0.0499 0.1737 0.0192 0.1875 ...
##  $ u        : num  18.7 17.8 19.4 19.4 19 ...
##  $ g        : num  17.2 16.6 18.5 17.9 17.8 ...
##  $ r        : num  16.7 16.2 18.2 17.1 17.4 ...
##  $ i        : num  16.5 16 18 16.7 17.2 ...
##  $ z        : num  16.4 15.9 18 16.4 17.1 ...
##  $ run      : int  752 752 752 752 752 752 752 752 752 752 ...
##  $ rerun    : int  301 301 301 301 301 301 301 301 301 301 ...
##  $ camcol   : int  4 4 4 4 4 4 4 4 4 4 ...
```

```
##  $ field    : int  267 269 269 269 269 270 270 270 271 271 ...
##  $ specobjid: chr  "363814405953054720" "3722365362331820032" "364954875244603392" "3232869639541493
##  $ class    : chr  "STAR" "STAR" "STAR" "GALAXY" ...
##  $ redshift : num  -5.49e-05 -1.11e-04 3.15e-04 1.00e-01 3.15e-04 ...
##  $ plate    : int  323 3306 324 287 3306 323 288 3306 3306 3306 ...
##  $ mjd      : int  51615 54922 51666 52023 54922 51615 52000 54922 54922 54922 ...
##  $ fiberid  : int  541 510 594 559 515 595 400 506 547 546 ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

```
summary(sky.train)
```

```
##      objid                           ra               dec
##  Min.   :1237646798137852371   Min.   :  8.235   Min.   :-5.3826
##  1st Qu.:1237648705652129817   1st Qu.:157.348   1st Qu.:-0.5486
##  Median :1237648722294341764   Median :180.491   Median : 0.3869
##  Mean   :1237649689210512330   Mean   :175.418   Mean   :14.7823
##  3rd Qu.:1237651190821290109   3rd Qu.:201.511   3rd Qu.:34.9444
##  Max.   :1237651540334280888   Max.   :260.884   Max.   :68.5423
##        u                g                r                i
##  Min.   :12.99   Min.   :12.80   Min.   :12.43   Min.   :11.95
##  1st Qu.:18.18   1st Qu.:16.81   1st Qu.:16.17   1st Qu.:15.85
##  Median :18.85   Median :17.49   Median :16.86   Median :16.55
##  Mean   :18.62   Mean   :17.37   Mean   :16.84   Mean   :16.58
##  3rd Qu.:19.26   3rd Qu.:18.01   3rd Qu.:17.51   3rd Qu.:17.26
##  Max.   :19.60   Max.   :19.74   Max.   :24.80   Max.   :28.18
##        z              run             rerun         camcol
##  Min.   :11.61   Min.   : 308.0   Min.   :301   Min.   :1.00
##  1st Qu.:15.62   1st Qu.: 752.0   1st Qu.:301   1st Qu.:2.00
##  Median :16.39   Median : 756.0   Median :301   Median :4.00
##  Mean   :16.42   Mean   : 981.3   Mean   :301   Mean   :3.63
##  3rd Qu.:17.15   3rd Qu.:1331.0   3rd Qu.:301   3rd Qu.:5.00
##  Max.   :22.83   Max.   :1412.0   Max.   :301   Max.   :6.00
##      field        specobjid            class              redshift
##  Min.   : 11.0   Length:8001        Length:8001        Min.   :-0.004136
##  1st Qu.:184.0   Class :character   Class :character   1st Qu.: 0.000082
##  Median :298.0   Mode  :character   Mode  :character   Median : 0.042371
##  Mean   :302.4                                         Mean   : 0.142588
##  3rd Qu.:415.0                                         3rd Qu.: 0.092275
##  Max.   :768.0                                         Max.   : 5.353854
##      plate           mjd           fiberid
##  Min.   : 266   Min.   :51578   Min.   :  1.0
##  1st Qu.: 301   1st Qu.:51900   1st Qu.:184.0
##  Median : 442   Median :51999   Median :348.0
##  Mean   :1465   Mean   :52949   Mean   :351.6
##  3rd Qu.:2559   3rd Qu.:54468   3rd Qu.:510.0
##  Max.   :8410   Max.   :57481   Max.   :998.0
```

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

```
##     objid        ra       dec         u         g         r         i
##         0         0         0         0         0         0         0
##         z       run     rerun    camcol     field specobjid     class
##         0         0         0         0         0         0         0
##  redshift     plate       mjd   fiberid
##         0         0         0         0
```

```
dim(sky.train)
```

```
## [1] 8001   18
```

4.2 Types of attributes

```
sapply(sky.train, class) # checking the class of every feature
```

```
##       objid          ra         dec           u           g           r
## "integer64"   "numeric"   "numeric"   "numeric"   "numeric"   "numeric"
##           i           z         run       rerun      camcol       field
##   "numeric"   "numeric"   "integer"   "integer"   "integer"   "integer"
##   specobjid       class    redshift       plate         mjd     fiberid
## "character" "character"   "numeric"   "integer"   "integer"   "integer"
```

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)
```

```
## [1] "GALAXY" "QSO"    "STAR"
```

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

```
## [1] "GALAXY" "QSO"    "STAR"
```

4.3 Class distribution

Summarize the class distribution:

```
percentage <- prop.table(table(sky.train$class)) * 100
cbind(freq=table(sky.train$class), percentage=percentage)
```

```
##        freq percentage
## GALAXY 3999  49.981252
## QSO     680   8.498938
## STAR   3322  41.519810
```

4.4 Statistical Summary

```
summary(sky.train)
```

```
##      objid                          ra              dec
##  Min.   :1237646798137852371   Min.   :  8.235   Min.   :-5.3826
##  1st Qu.:1237648705652129817   1st Qu.:157.348   1st Qu.:-0.5486
##  Median :1237648722294341764   Median :180.491   Median : 0.3869
##  Mean   :1237649689210512330   Mean   :175.418   Mean   :14.7823
##  3rd Qu.:1237651190821290109   3rd Qu.:201.511   3rd Qu.:34.9444
##  Max.   :1237651540334280888   Max.   :260.884   Max.   :68.5423
##        u               g               r               i
##  Min.   :12.99   Min.   :12.80   Min.   :12.43   Min.   :11.95
##  1st Qu.:18.18   1st Qu.:16.81   1st Qu.:16.17   1st Qu.:15.85
```

```
##  Median :18.85   Median :17.49   Median :16.86   Median :16.55
##  Mean   :18.62   Mean   :17.37   Mean   :16.84   Mean   :16.58
##  3rd Qu.:19.26   3rd Qu.:18.01   3rd Qu.:17.51   3rd Qu.:17.26
##  Max.   :19.60   Max.   :19.74   Max.   :24.80   Max.   :28.18
##        z             run             rerun         camcol
##  Min.   :11.61   Min.   : 308.0   Min.   :301   Min.   :1.00
##  1st Qu.:15.62   1st Qu.: 752.0   1st Qu.:301   1st Qu.:2.00
##  Median :16.39   Median : 756.0   Median :301   Median :4.00
##  Mean   :16.42   Mean   : 981.3   Mean   :301   Mean   :3.63
##  3rd Qu.:17.15   3rd Qu.:1331.0   3rd Qu.:301   3rd Qu.:5.00
##  Max.   :22.83   Max.   :1412.0   Max.   :301   Max.   :6.00
##      field        specobjid            class         redshift
##  Min.   : 11.0   Length:8001        GALAXY:3999   Min.   :-0.004136
##  1st Qu.:184.0   Class :character   QSO   : 680   1st Qu.: 0.000082
##  Median :298.0   Mode  :character   STAR  :3322   Median : 0.042371
##  Mean   :302.4                                    Mean   : 0.142588
##  3rd Qu.:415.0                                    3rd Qu.: 0.092275
##  Max.   :768.0                                    Max.   : 5.353854
##      plate           mjd           fiberid
##  Min.   : 266   Min.   :51578   Min.   :  1.0
##  1st Qu.: 301   1st Qu.:51900   1st Qu.:184.0
##  Median : 442   Median :51999   Median :348.0
##  Mean   :1465   Mean   :52949   Mean   :351.6
##  3rd Qu.:2559   3rd Qu.:54468   3rd Qu.:510.0
##  Max.   :8410   Max.   :57481   Max.   :998.0
```

We observe here two different things that the data preparation and exploratory analysis tells us:
- The class distribution is not even and this could be solved later using the SMOTE function which equalizes the classes proportion.
- The numeric columns are not on the same scale.
As a first step we decided to explore the data "as it is" and later, when we build the model, we evaluate the use of the SMOTE function and equalize the classes.

5 Visualization

5.1 Grouped Features

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")
```

- u = better of DeV/Exp magnitude fit
- g = better of DeV/Exp magnitude fit
- r = better of DeV/Exp magnitude fit
- i = better of DeV/Exp magnitude fit
- z = better of DeV/Exp magnitude fit

Field:
Run, rerun, camcol and 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. A field can be identified by: - run number, which identifies the specific scan, - the camera column, or "camcol," a number from 1 to 6, identifying the scanline within the run, and - the field number. The field number typically starts at 11 (after an initial rampup time), and can be as large as 800 for particularly long runs. - An additional number, rerun, specifies how the image was processed.

```
field_feat <- c("run", "rerun", "camcol", "field")
```

- run = Run Number
- rereun = Rerun Number
- camcol = Camera column
- field = Field number

Skies or Sky:
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. Source.

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

Remaining features:

- redshift = Final Redshift
- plate = plate number
- mjd = MJD of observation
- fiberid = fiber ID
  In physics, redshift happens when light or other electromagnetic radiation from an object is increased in wavelength, or shifted to the red end of the spectrum.

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.

Modified Julian Date, used to indicate the date that a given piece of SDSS data (image or spectrum) was taken.

The SDSS spectrograph uses optical fibers to direct the light at the focal plane from individual objects to the slithead. Each object is assigned a corresponding fiberID.

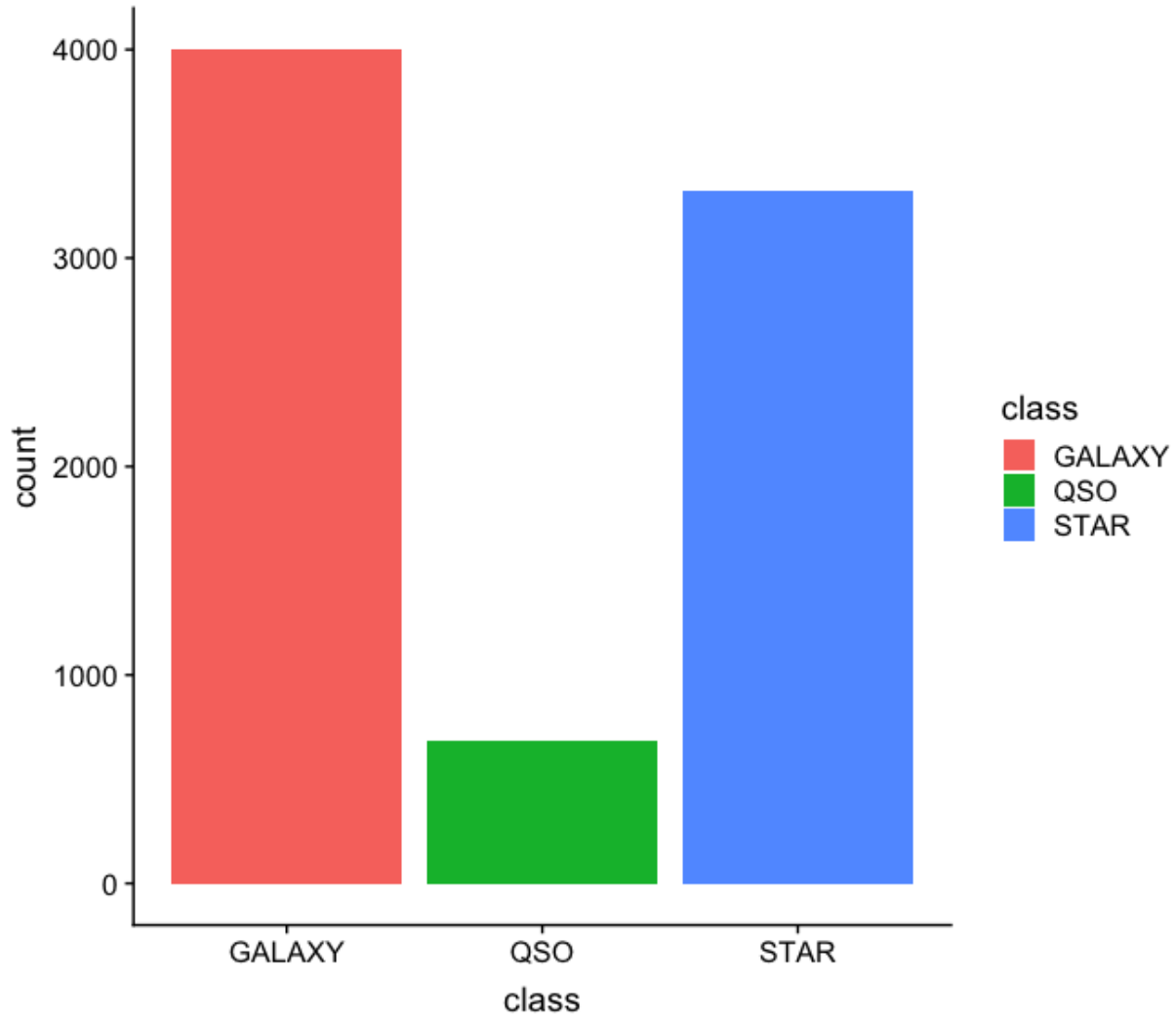
```
equipment_feat <- c("redshift", "plate", "mjd",
                    "fiberid") # These features are related to the measurement equipment
```

Class and Identification:
The class identifies an object to be either a galaxy, star or quasar. This will be the response variable which we will be trying to predict. - View “SpecObj” - specobjid = Object Identifier - class = object class (galaxy, star or quasar object)

5.2 Univariate Plots

The *class* distribution can now be visualized on the training set:

```
x <- sky.train[,-c(1, 13, 14, 16, 17)]
y <- sky.train$class # split input and output
ggplot(sky.train, aes(class, fill = class)) +
  geom_bar() # plot the classes
```



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.

```
# normalize using the package 'RSNNS'
set.seed(2205)
sky.train.norm <- normalizeData(sky.train[,-c(1, 13, 14)], type = "0_1")
sky.train.norm <- as.data.frame(sky.train.norm)
summary(sky.train.norm) # check the normalization
```

```
##        V1                V2                V3                V4
##  Min.   :0.0000   Min.   :0.00000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.5902   1st Qu.:0.06539   1st Qu.:0.7850   1st Qu.:0.5787
##  Median :0.6818   Median :0.07805   Median :0.8869   Median :0.6766
##  Mean   :0.6617   Mean   :0.27278   Mean   :0.8518   Mean   :0.6587
##  3rd Qu.:0.7650   3rd Qu.:0.54551   3rd Qu.:0.9481   3rd Qu.:0.7503
##  Max.   :1.0000   Max.   :1.00000   Max.   :1.0000   Max.   :1.0000
##        V5                V6                V7                V8
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.3022   1st Qu.:0.2405   1st Qu.:0.3569   1st Qu.:0.4022
##  Median :0.3578   Median :0.2838   Median :0.4256   Median :0.4058
##  Mean   :0.3563   Mean   :0.2855   Mean   :0.4285   Mean   :0.6099
```

```
##  3rd Qu.:0.4109   3rd Qu.:0.3275   3rd Qu.:0.4932   3rd Qu.:0.9266
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
##        V9            V10              V11              V12
##  Min.   :301   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000000
##  1st Qu.:301   1st Qu.:0.2000   1st Qu.:0.2285   1st Qu.:0.0007872
##  Median :301   Median :0.6000   Median :0.3791   Median :0.0086800
##  Mean   :301   Mean   :0.5259   Mean   :0.3849   Mean   :0.0273843
##  3rd Qu.:301   3rd Qu.:0.8000   3rd Qu.:0.5337   3rd Qu.:0.0179940
##  Max.   :301   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000000
##       V13                V14               V15
##  Min.   :0.000000   Min.   :0.00000   Min.   :0.0000
##  1st Qu.:0.004298   1st Qu.:0.05455   1st Qu.:0.1836
##  Median :0.021611   Median :0.07132   Median :0.3480
##  Mean   :0.147193   Mean   :0.23222   Mean   :0.3517
##  3rd Qu.:0.281557   3rd Qu.:0.48958   3rd Qu.:0.5105
##  Max.   :1.000000   Max.   :1.00000   Max.   :1.0000
```

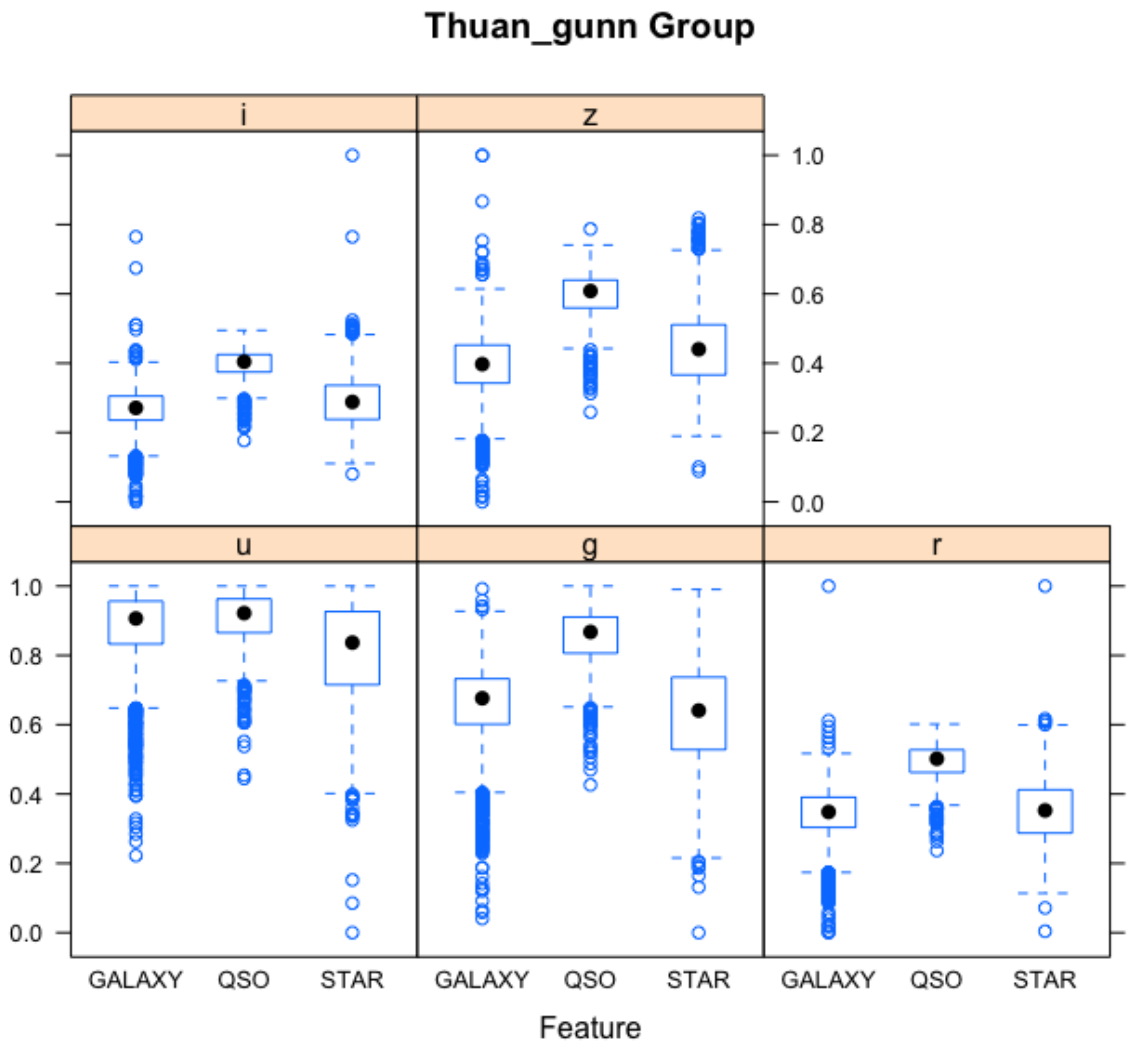
Tidying the normalized data:

```
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
# 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)
```

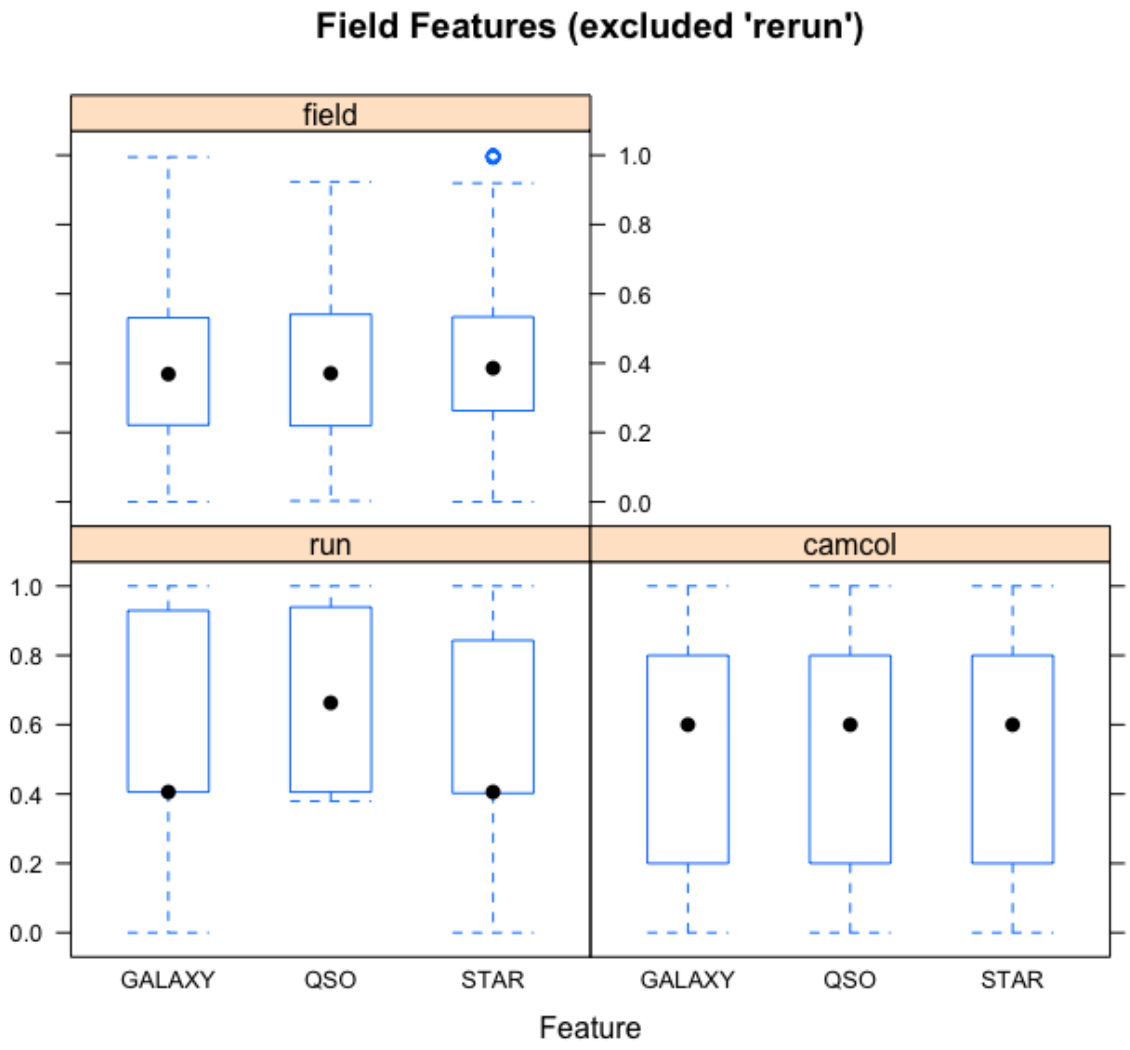
```
##          ra        dec         u         g         r        i         z
## 1 0.6940976 0.07464221 0.8582499 0.6362373 0.3431382 0.279811 0.4260215
## 2 0.6951749 0.07348733 0.7224990 0.5480751 0.3014897 0.248584 0.3826164
##         run rerun camcol    field     redshift       plate         mjd
## 1 0.4021739   301    0.6 0.338177 0.0007616973 0.006999018 0.006267999
## 2 0.4021739   301    0.6 0.340819 0.0007513008 0.373280943 0.566491614
##     fiberid               objid           specobjid class
## 1 0.5416249 1237648704577142859  363814405953054720  STAR
## 2 0.5105316 1237648704577273907 3722365362331820032  STAR
```

plotting with the normalized data:

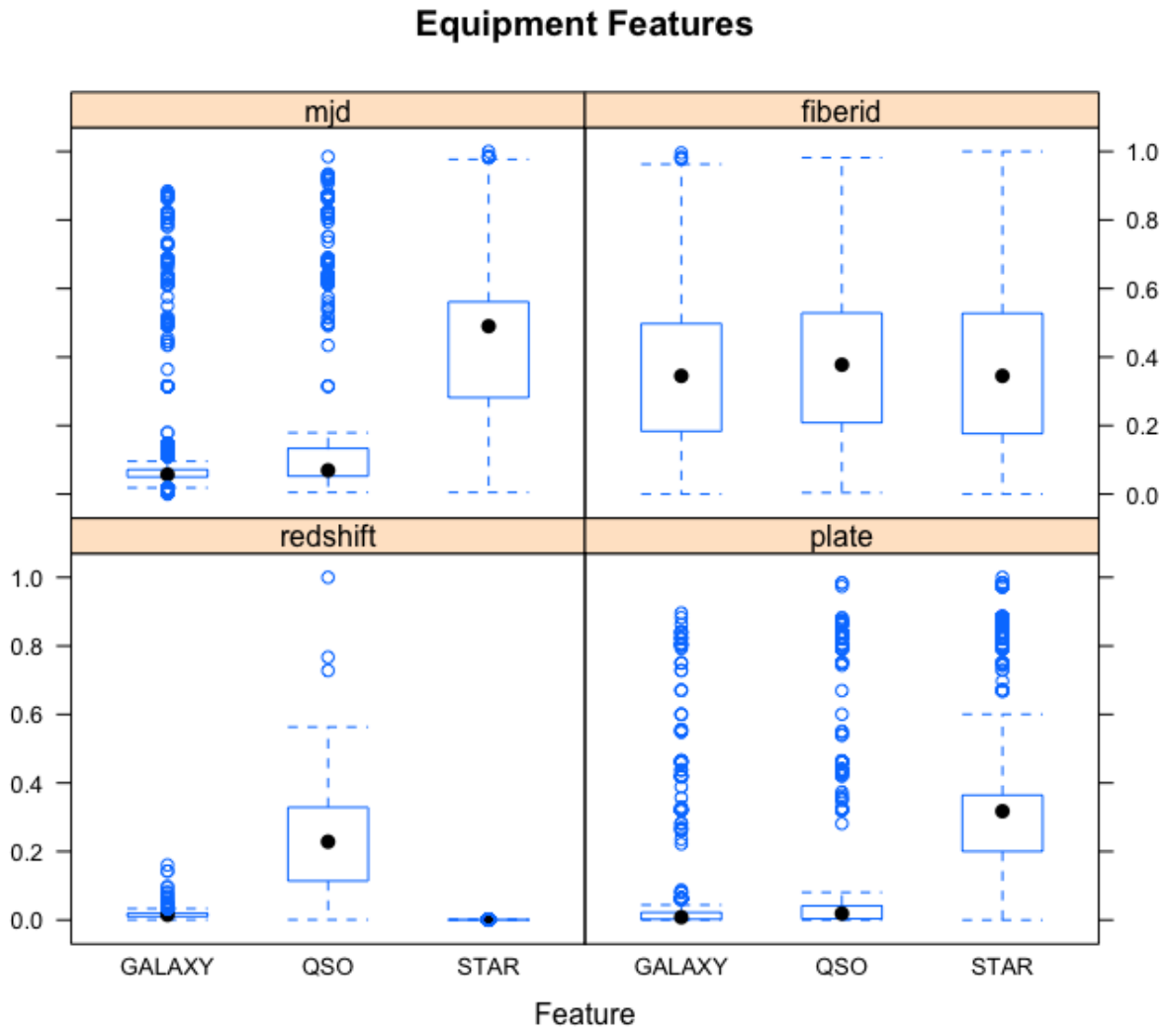
```
x <- sky.train.norm[,-c(9, 16:18)]
y <- sky.train.norm$class
featurePlot(x = sky.train.norm[, c("u", "g", "r", "i", "z")], y = y, plot="box",
            main = "Thuan_gunn Group")
```



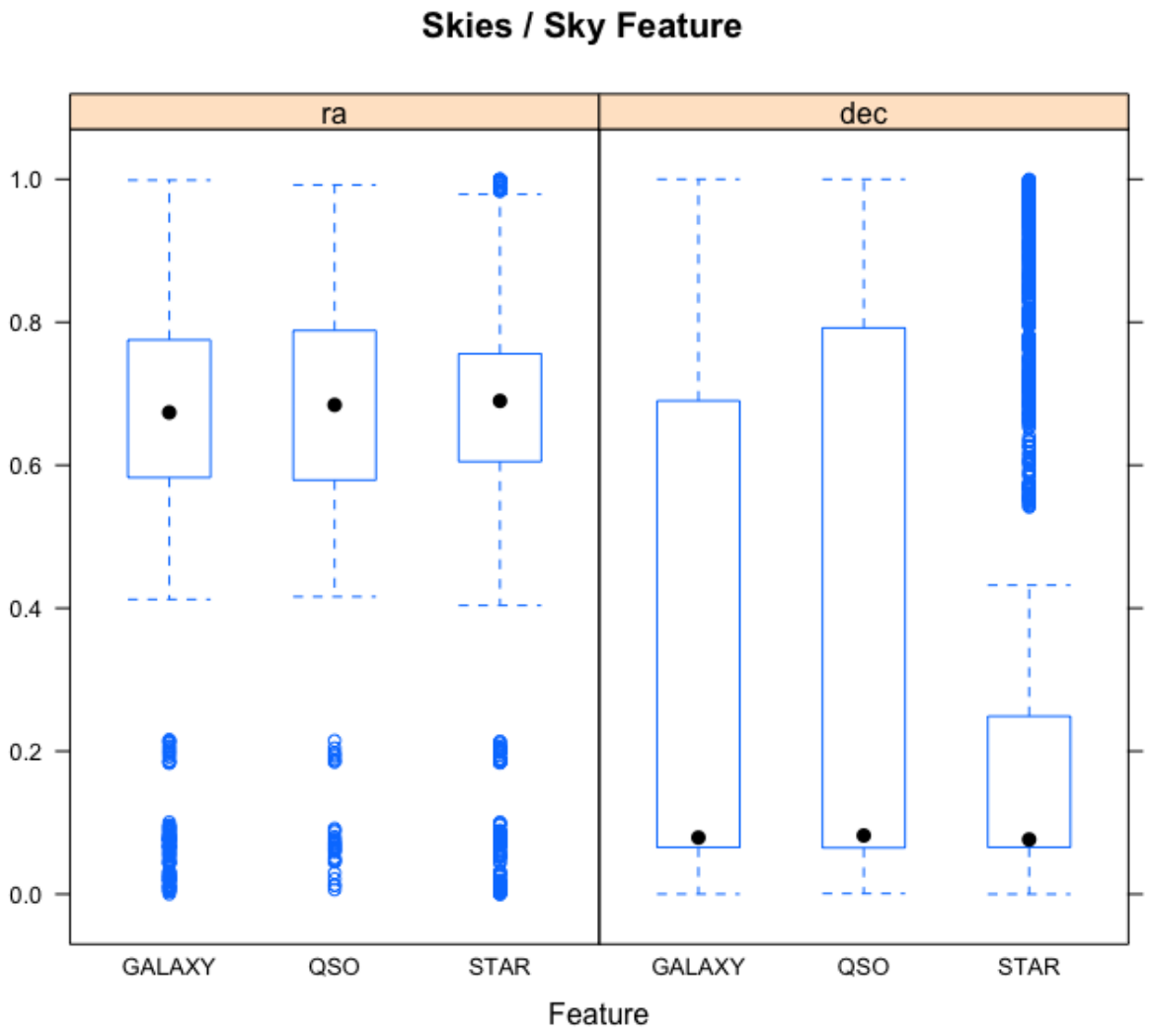

```
featurePlot(x = sky.train.norm[, c("run", "camcol", "field")], y = y, plot="box",
            main = "Field Features (excluded 'rerun')")
```




```
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")
```

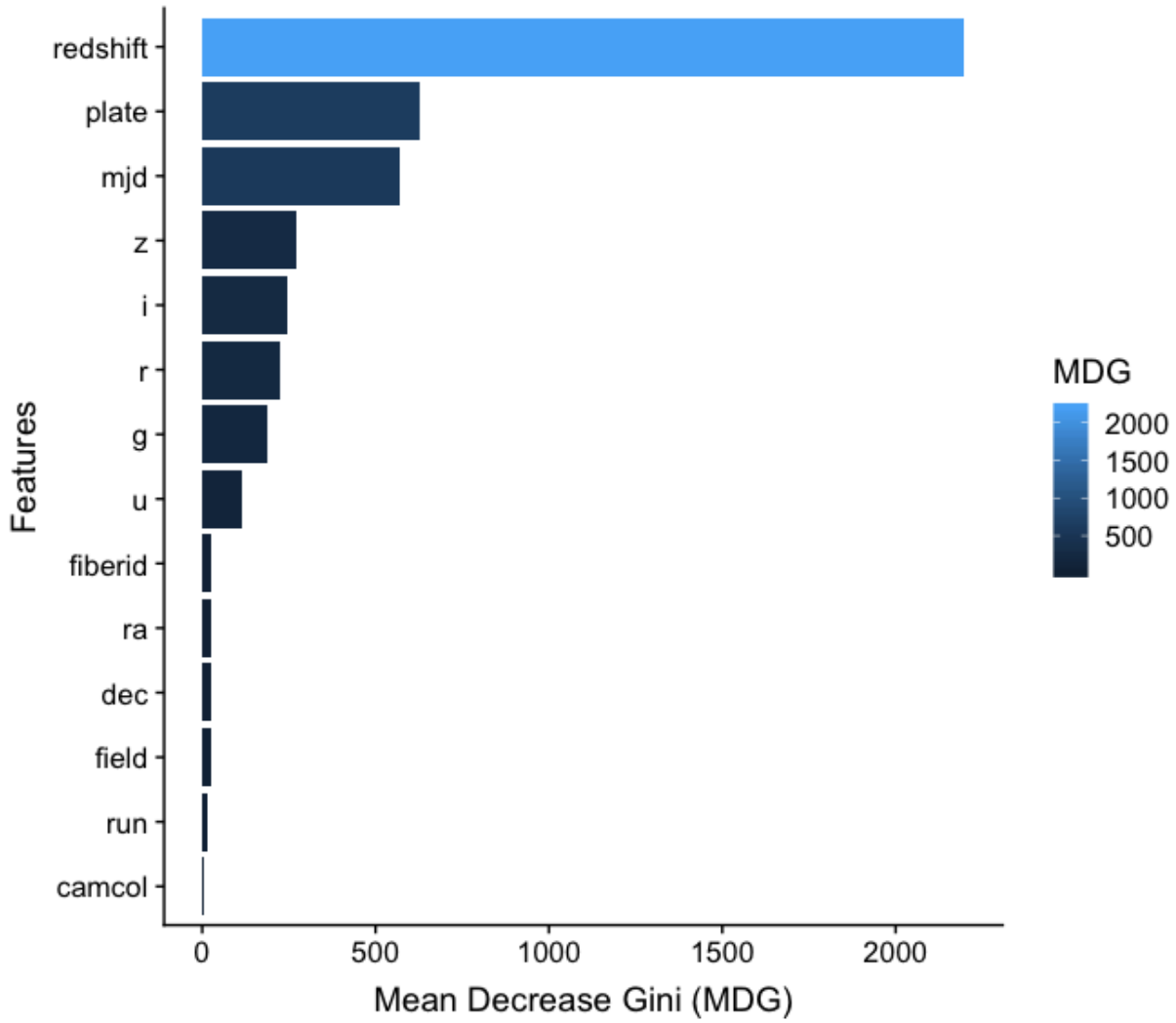


This visualization is useful for us to notice that there are clearly different distributions of the attributes for each class value and to identify the outliers (noise).
There seems to be great variability in the 'mjd' and 'plate' parameters.
The variability found in the 'Thuann_gunn' group will be kept 'as it is' and we will deal with it only in case we need to improve our model.

5.3 Variables importance

Running a model with *randomForest* to check the variables importance:

```
set.seed(2205)
rf.sky.train <- randomForest(class ~ ., data = sky.train[, -c(1, 10, 13)])
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()
```



The *plate* seems to have importance when predicting. This could be a noisy parameter since we have different plates measuring the waves.
The *MJD* seems to be important as well. Could those two features make the difference in the model?

Running a PCA analysis to check the variables importance (we are excluding 2 constant features and the factors)

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

```
## Importance of components:
##                             PC1       PC2       PC3      PC4       PC5
## Standard deviation     2323.0608 310.08422 277.55847 197.0730 140.06275
## Proportion of Variance    0.9586   0.01708   0.01368   0.0069   0.00348
## Cumulative Proportion     0.9586   0.97566   0.98935   0.9962   0.99973
##                             PC6      PC7   PC8   PC9 PC10   PC11   PC12
## Standard deviation     36.31792 13.92237 2.208 1.388 0.66 0.3447 0.1537
## Proportion of Variance  0.00023  0.00003 0.000 0.000 0.00 0.0000 0.0000
## Cumulative Proportion   0.99996  1.00000 1.000 1.000 1.00 1.0000 1.0000
##                            PC13    PC14
## Standard deviation     0.1354 0.08124
```

```
## Proportion of Variance 0.0000 0.00000
## Cumulative Proportion  1.0000 1.00000
```

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 could use all the features already considered by the PCA analysis or use only the first 6 features.

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 / list 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.

```
rownames(imp.df) <- NULL
imp.df %>% knitr::kable(caption = "Importance")
```

Table 1: Importance

features	MDG
redshift	2198.928548
plate	625.431866
mjd	570.201435
z	272.272961
i	245.159960
r	224.532084
g	187.561329
u	112.309808
fiberid	28.234315
ra	26.911081
dec	25.798185
field	23.307735
run	16.300995
camcol	6.936498

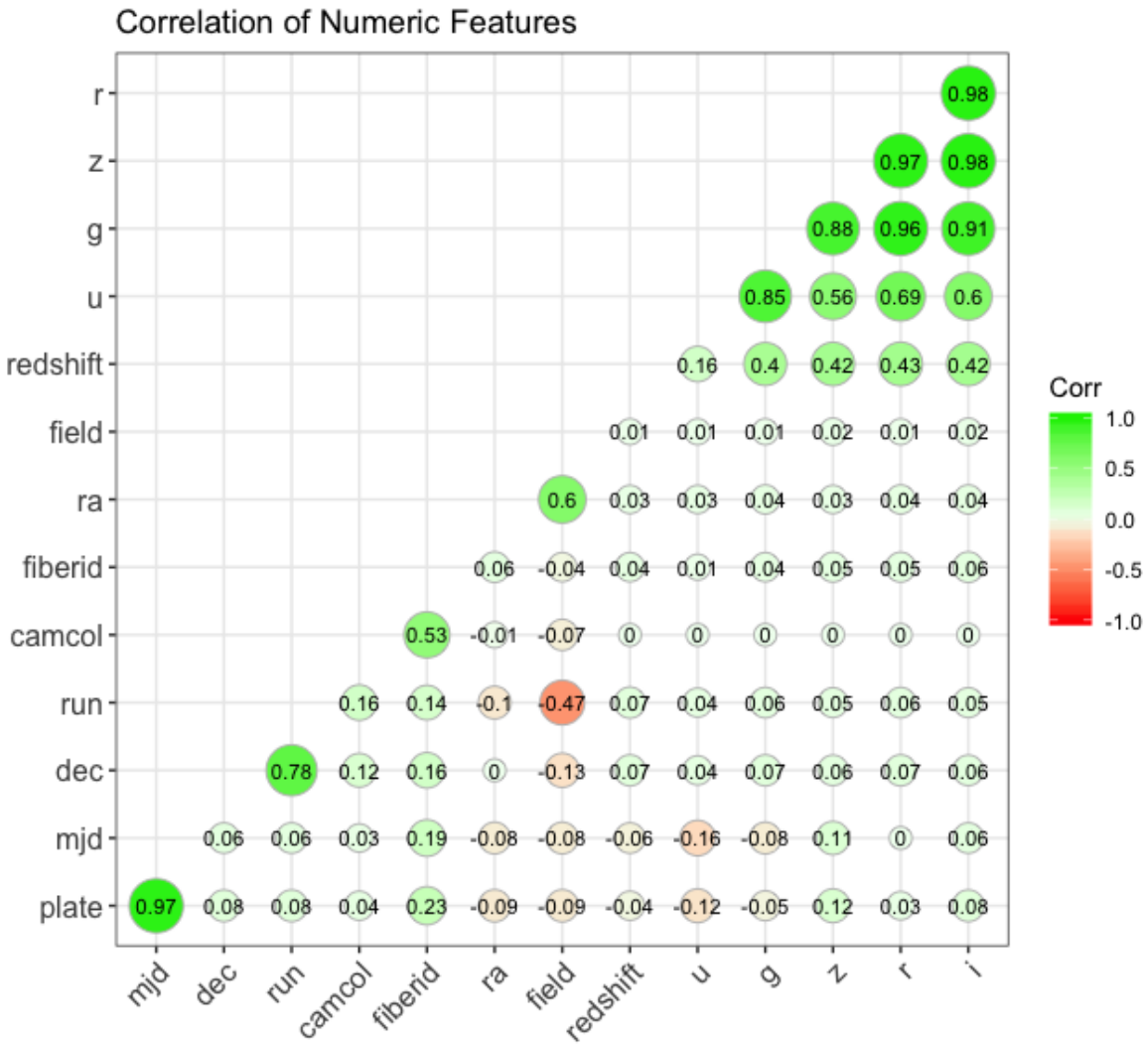
6 Defining the dataset for the model:

6.1 Features Selection

We will select the features to be part of the dataset that will be evaluated.

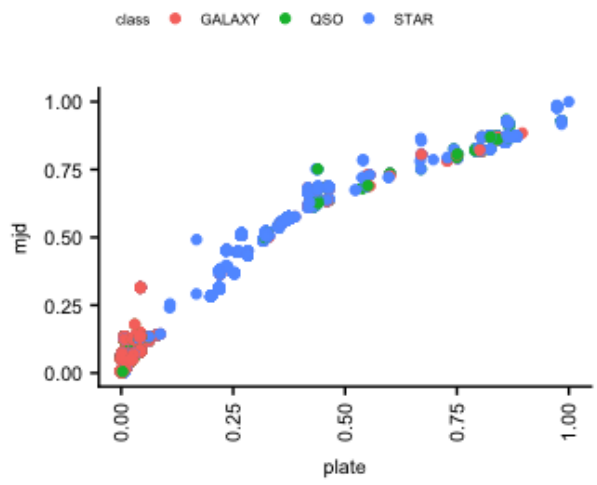
```
sky.model <- sky.train[, -c(1, 10, 13)]
corr <- round(cor(sky.model[, -11], use = "complete.obs"), 2)
ggcorrplot(corr, hc.order = TRUE,
           type = "lower",
           lab = TRUE,
           lab_size = 3,
           method="circle",
```

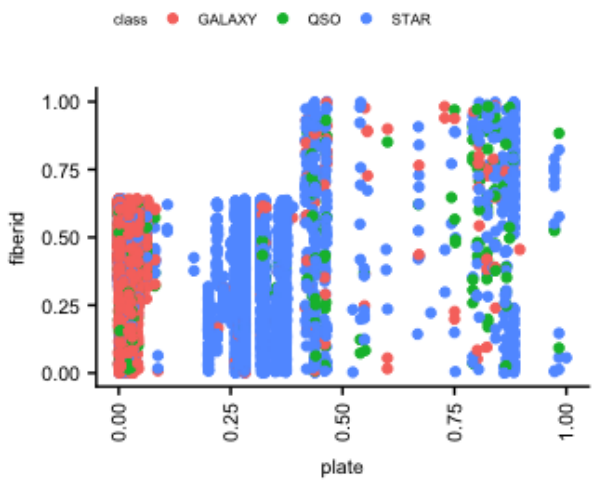
```
          colors = c("red1", "honeydew", "green2"),
          title="Correlation of Numeric Features",
          ggtheme=theme_bw)
```

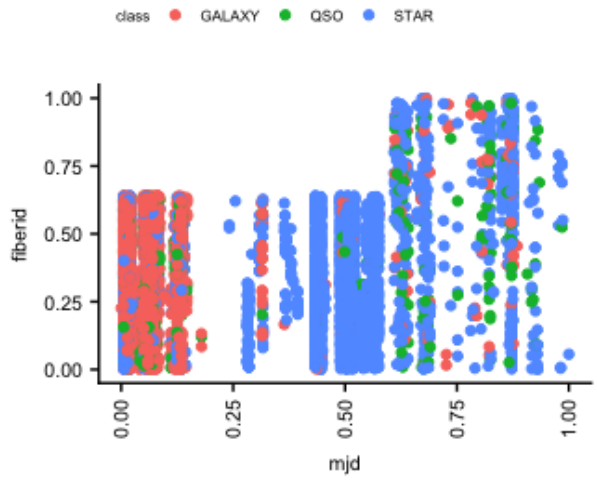


Although the *plate* and *mjd* feature show with a high importance in the MDG evaluation, their correlation is based on the multicollinearity adn they are not independent because every *plate* is linked to a *fiberid* and a *mjd*.
For this reason we decided to **exclude** these features from the dataset that will be used on the model.

```
theme1 <- theme(axis.text.x = element_text(size = 8, angle = 90, hjust = 0.5, vjust = 0.5),
                axis.text.y = element_text(size = 8, angle = 0, hjust = 0.5, vjust = 0.5),
                axis.title.x = element_text(size = 8, angle = 0, hjust = 0.5, vjust = 0.5),
                axis.title.y = element_text(size = 8, angle = 90, hjust = 0.5, vjust = 0.5),
               legend.position="top", legend.text = element_text(size = 6),
               legend.title = element_text(size = 6))
plot_grid(ggplot(sky.train.norm, aes(plate, mjd, col = class)) + geom_point() + theme1,
          ggplot(sky.train.norm, aes(plate, fiberid, col = class)) + geom_point() + theme1,
          ggplot(sky.train.norm, aes(mjd, fiberid, col = class)) + geom_point() + theme1,
          align = "h")
```





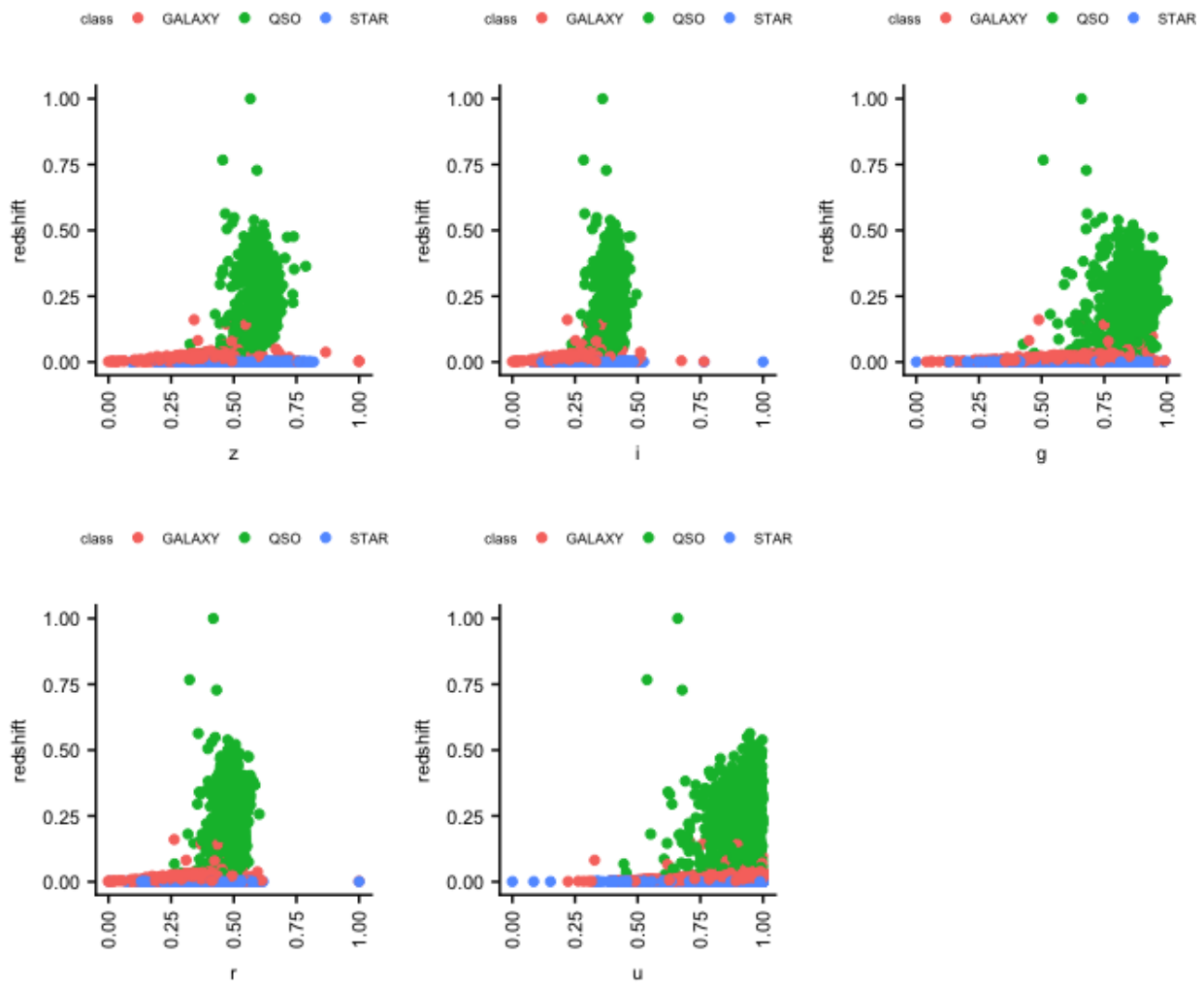


Selecting the columns:

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

As a last check we will visualize the **redshift** variable because it showed a greater importance compared to the other variables.
This is the interaction between "redshift" and the other selected features (note: for better visualization we used the normalized dataset to make it easier the comparison).

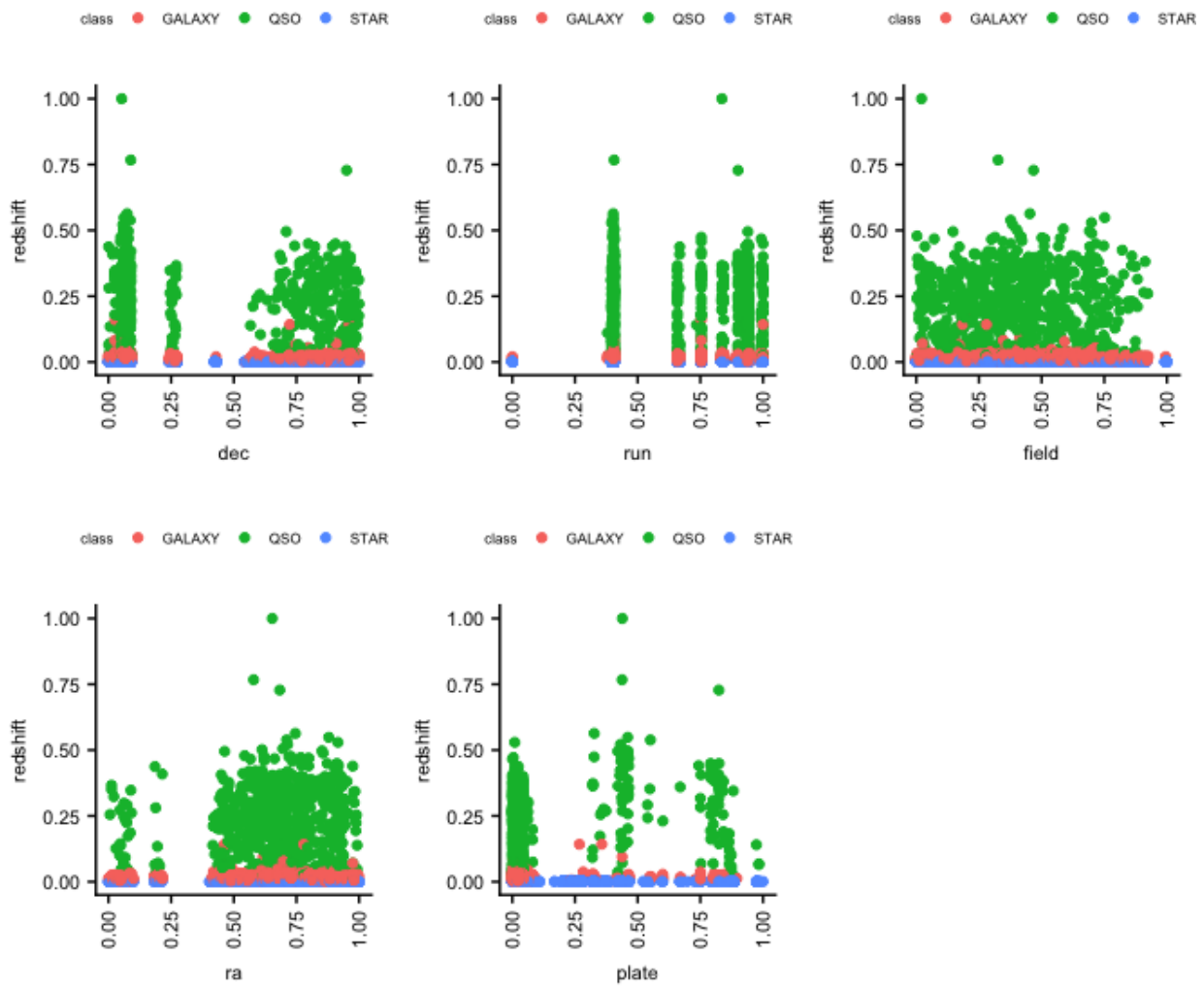
```
plot_grid(ggplot(sky.train.norm, aes(z, redshift, col = class)) + geom_point() + theme1,
          ggplot(sky.train.norm, aes(i, redshift, col = class)) + geom_point() + theme1,
          ggplot(sky.train.norm, aes(g, redshift, col = class)) + geom_point() + theme1,
          ggplot(sky.train.norm, 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" how important it is, especially in the low wave length values showing a higher concentration forthe QSO making this feature an important observation for the "class" of the object 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 visualizations 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.

7 Evaluate some algorithms

7.1 Testing Harness

Run algorithms using 20-fold cross validation

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

7.2 Build Models

The model to be evaluated will be:
- Linear - nonLinear - Advanced

```
# 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=contro
# Random Forest
set.seed(2205)
fit.rf <- caret::train(class ~ . , data=sky.model, method="rf", metric=metric, trControl=control)
```

7.3 Select best model

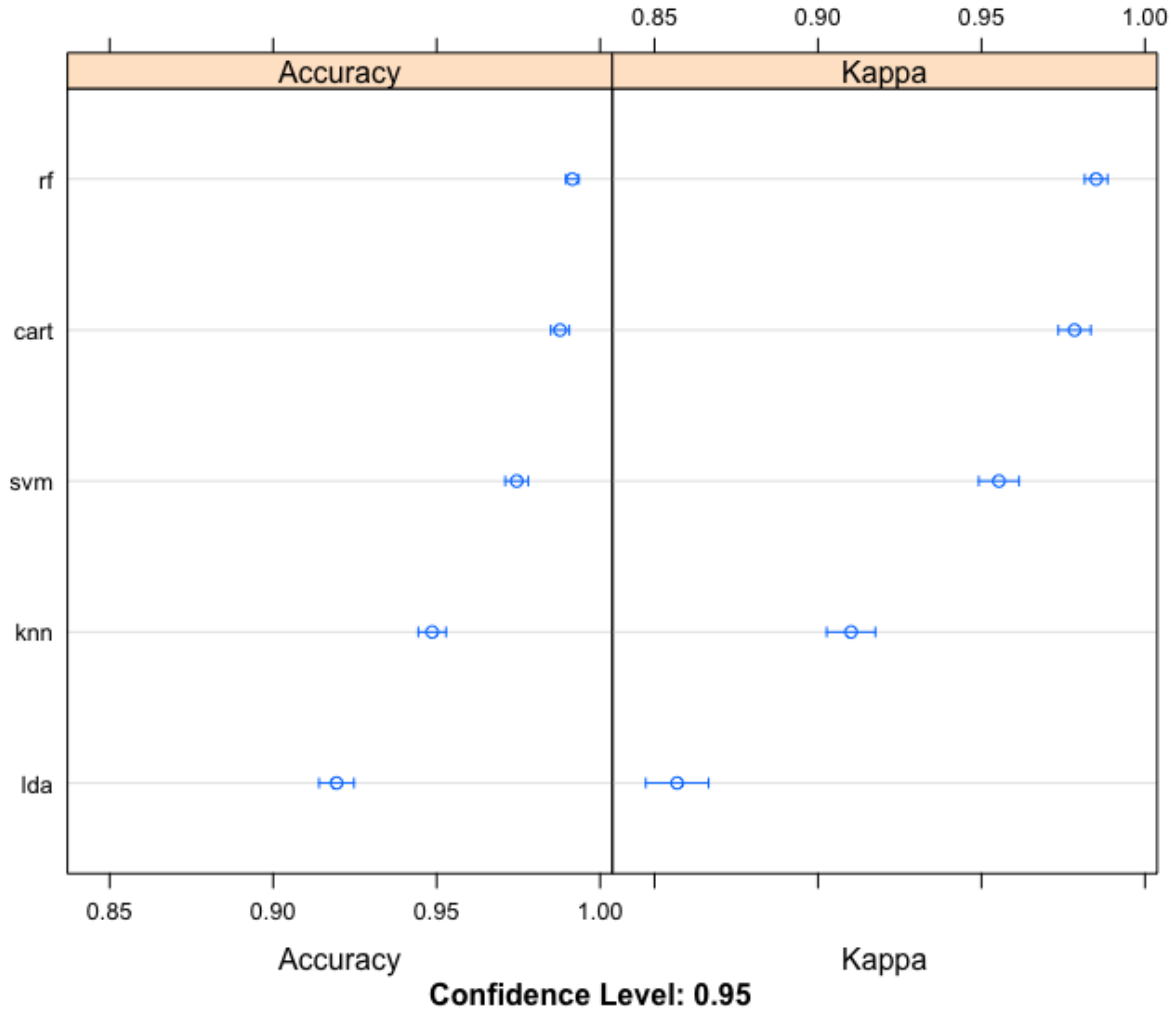
Summarizing the accuracy of the models:

```
results <- resamples(list(lda=fit.lda, cart=fit.cart, knn=fit.knn, svm=fit.svm,
                          rf=fit.rf))
summary(results)
```

```
##
## Call:
## summary.resamples(object = results)
##
## Models: lda, cart, knn, svm, rf
## Number of resamples: 20
##
## Accuracy
##        Min.   1st Qu.    Median      Mean 3rd Qu.      Max. NA's
## lda  0.9000 0.9126637 0.9175000 0.9193884 0.92375 0.9425000    0
## cart 0.9750 0.9843750 0.9875000 0.9877528 0.99250 0.9974937    0
## knn  0.9325 0.9437500 0.9512500 0.9486321 0.95500 0.9625000    0
## svm  0.9575 0.9718750 0.9762500 0.9745034 0.98000 0.9875000    0
## rf   0.9800 0.9900000 0.9912625 0.9915016 0.99500 0.9975000    0
##
## Kappa
##           Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## lda  0.8208287 0.8448042 0.8541122 0.8569261 0.8646839 0.8983425    0
## cart 0.9557972 0.9724931 0.9780355 0.9784554 0.9868669 0.9956020    0
## knn  0.8813552 0.9011663 0.9148438 0.9101043 0.9213392 0.9345178    0
## svm  0.9253535 0.9507732 0.9582954 0.9553256 0.9649156 0.9780278    0
## rf   0.9646893 0.9824316 0.9846596 0.9850665 0.9912157 0.9956262    0
```

Compare the accuracy of the models:

```
dotplot(results)
```



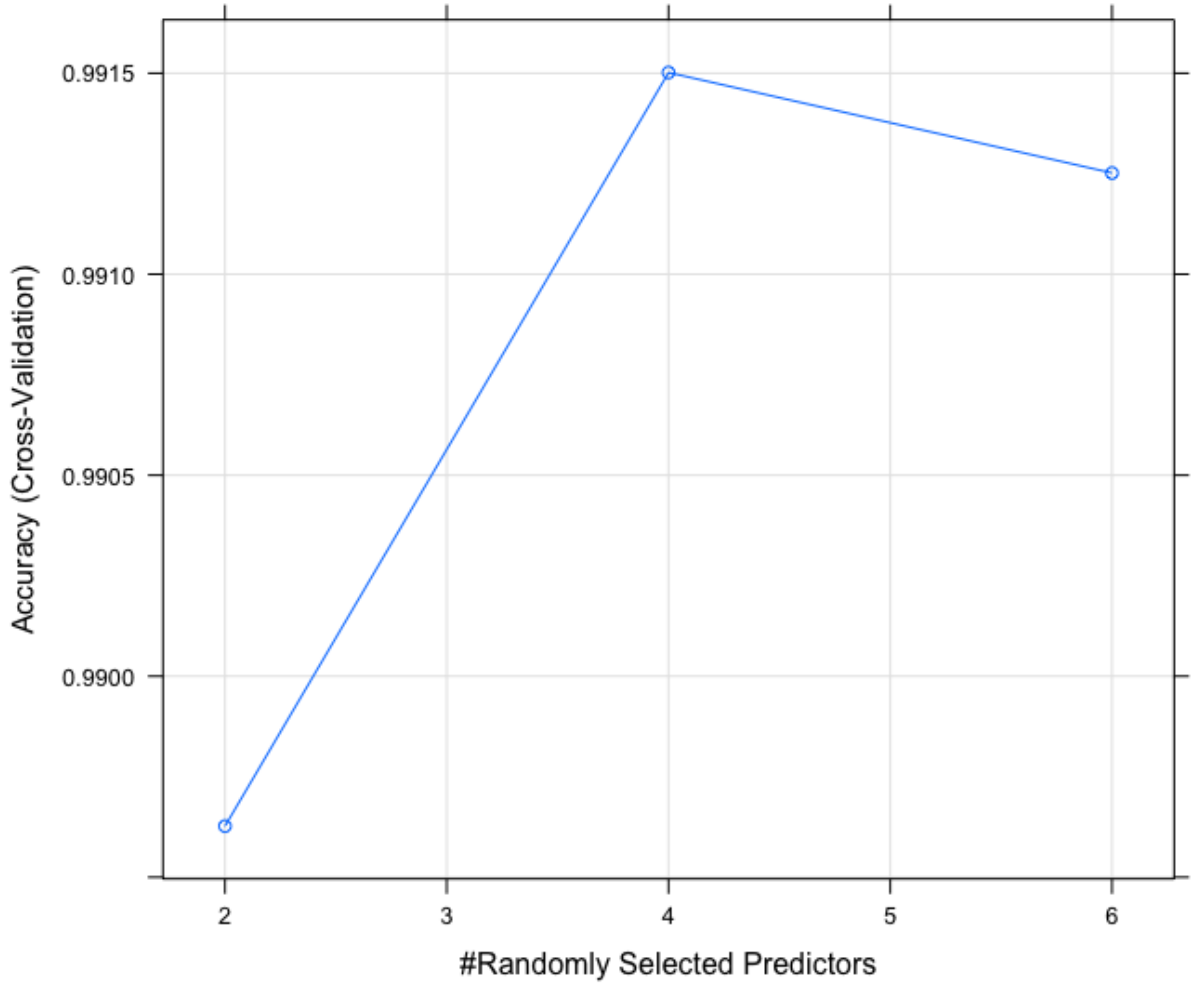

```
print(fit.rf)
```

```
## Random Forest
##
## 8001 samples
##    6 predictor
##    3 classes: 'GALAXY', 'QSO', 'STAR'
##
## No pre-processing
## Resampling: Cross-Validated (20 fold)
## Summary of sample sizes: 7601, 7601, 7601, 7601, 7601, 7601, ...
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##   2     0.9896272  0.9817734
##   4     0.9915016  0.9850665
##   6     0.9912516  0.9846349
##
```

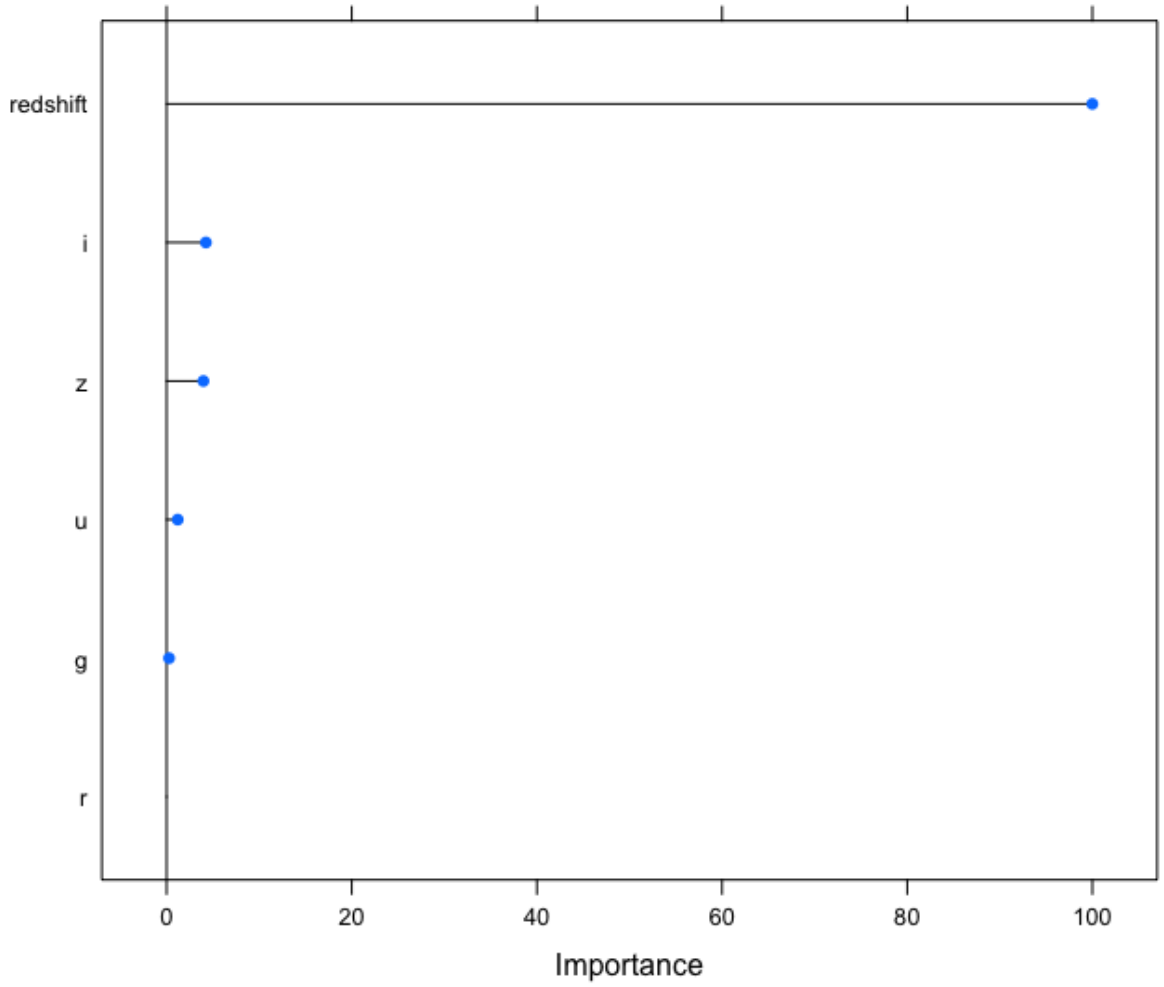
```
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 4.
```

The Accuracy vs. predictors and variable importance:

```
plot(fit.rf)
```




```
plot(varImp(fit.rf))
```



8 Make Predictions

8.1 Estimating the skill of RF (randomForest) on the validation dataset.

columns to be discarded:

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

Predict:

```
set.seed(2205)
predictions <- predict(fit.rf, validation.model)
caret::confusionMatrix(predictions, validation.model$class)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction GALAXY QSO STAR
##     GALAXY    991   7    0
##     QSO         8 163    0
```

```
##     STAR        0   0  830
##
## Overall Statistics
##
##                Accuracy : 0.9925
##                  95% CI : (0.9877, 0.9958)
##     No Information Rate : 0.4997
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9869
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: GALAXY Class: QSO Class: STAR
## Sensitivity                 0.9920    0.95882      1.0000
## Specificity                 0.9930    0.99563      1.0000
## Pos Pred Value              0.9930    0.95322      1.0000
## Neg Pred Value              0.9920    0.99617      1.0000
## Prevalence                  0.4997    0.08504      0.4152
## Detection Rate              0.4957    0.08154      0.4152
## Detection Prevalence        0.4992    0.08554      0.4152
## Balanced Accuracy           0.9925    0.97722      1.0000
```

9 Results and Conclusion

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 (test) set having many cases of False Positives around 10% FP's when predicting STAR (predicted GALAXY instead) and 9% FP's when predicting QSO in relation to the other two classes.
The decision is to keep the "redshift" feature along with the Thuann_gunn group.
The final Accuracy is 0.9915019

Notes:
- Time to run the whole code 6.31 mins
- Computer used:
- MacBook Pro
- Processor: 2.4 GHz Intel Core i5
- Memory: 8 GB 1600 MHz DDR3
- Graphics: Intel Iris 1536 MB