STELLAR CLASSIFICATION USING SPECTRAL CHARACTERISTICS

Kristina Levina

16/09/2022

Summary: In this study, Stellar Classification Dataset - SDSS17 was explored. This is a multi-class classification problem with three classes: galaxies, stars, and quasars. The logistic regression model with `multinom` function from `nnet` package was utilised to solve this problem. The obtained accuracy for distinguishing stars from other classes is 99.4%. The obtained ϕ coefficient for distinguishing quasars from other classes is 89.2%. The obtained accuracy for distinguishing galaxies from other classes is 96.2%.

Introduction

Stellar Classification Dataset - SDSS17 [1] provides data for classification of stars, galaxies, and quasars based on their spectral characteristics. This dataset contains 18 attributes, one of which is `class`. Other 17 attributes include various spectral characteristics, object identifiers, and measurement-related parameters. Overall, the dataset contains 100000 observations of space taken by the Sloan Digital Sky Survey (SDSS).

The distribution of stars, quasars, and galaxies is shown in Fig. 1.

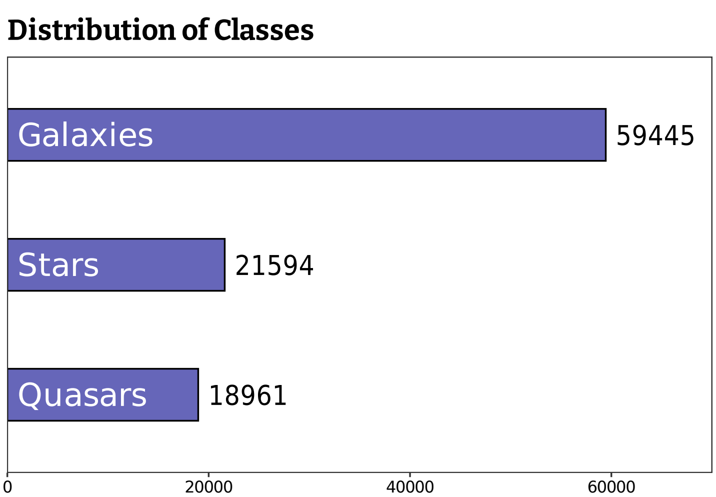

Figure 1: Distribution of stars, galaxies, and quasars in Stellar Classification Dataset - SDSS17

As one can observe from Fig. 1, the problem is quite imbalanced because the number of quasars is low with respect to the number of galaxies. The same concerns the number of stars.

Feature Selection

Full description of the attributes can be found in [1]. We need to select meaningful attributes for use in the model. To this, exploratory analysis is performed.

First, `obj-ID` is the object identifier, the unique value that identifies the object in the image catalog used by the CAS. Therefore, it should not be used in further analysis because it is not a spectral characteristic.

Second, `alpha` and `delta` specify the right ascension and declination angles, respectively. These angles specify the position of the observed object in the sky, making their use irrelevant for the model.

Third, `u`, `r`, `g`, `i`, `redshift`, and `z` are spectral characteristics. They should be included into the model.

Fourth, `run-ID`, `rerun-ID`, `cam-col`, `fiber-ID`, `plate`, `MJD`, and `field-ID` are features related to the the measurement setup, which makes them irrelevant for stellar classification based on spectroscopic data.

Fifth, `spec-obj-ID` is related to spectroscopic observations. Hence, we will use this feature in the model.

Let us now calculate the correlation between the selected features to understand their relationship between each other. The result is visualised in Fig. 2

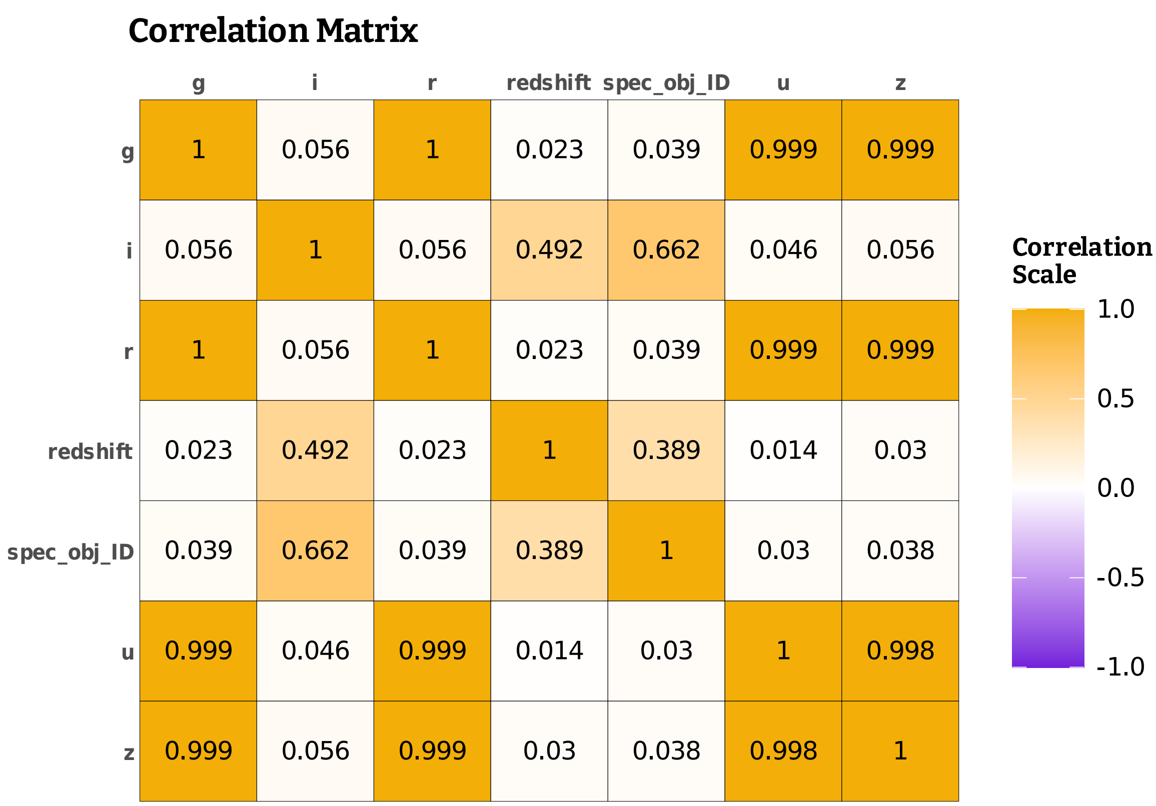

Figure 2: Correlation matrix between spectral features in the stellar classification dataset

We observe that `r`, `g`, `u`, and `z` are strongly correlated. However, further analysis of the model performance has shown that inclusion of all these features into the model improves the results.

Data Preprocessing

The selected features have been scaled so that they all have a mean of 0 and a standard deviation of 1. The dataset is split to 80% train and 20% test data randomly.

Problem Visualisation

We selected two meaningful features `spec-obj-ID` and `redshift` and visualised the distribution of classes. The results are shown in Fig. 3.

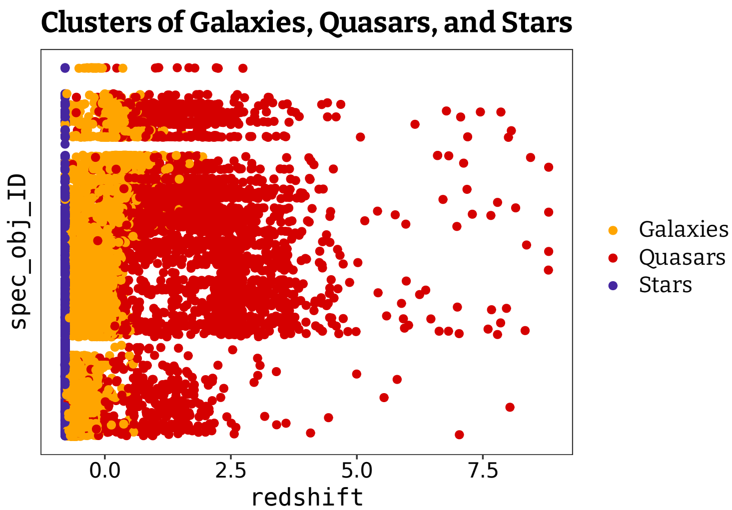

Figure 3: Visualised distribution of classes in the coordinate system of spec-obj-ID and redshift

Figure 3 shows three distinct clusters that should be easily identified using logistic regression. We see that the red shift value that corresponds to quasars is the highest because quasars are the most distant objects humans can observe. Stars have the lowest red shift value. The red shift value of galaxies is in between.

Model and Results

We use the logistic regression model with `multinom` function from `nnet` package. To evaluate the model performance, we use one-versus rest approach and investigate the following three cases: 1) galaxies-versus-rest classification, 2) quasars-versus-rest classification, and 3) stars-versus-rest classification. The performance of the model is assessed on the test data.

For the classification of stars versus other objects, the obtained train and test accuracies are 99.4% and 99.5%, respectively. Figure 4 shows the corresponding confusion matrices. The accuracy metric is a good choice in this case because the number of misclassification cases is low. Hence, stars are easy to distinguish from galaxies and quasars.

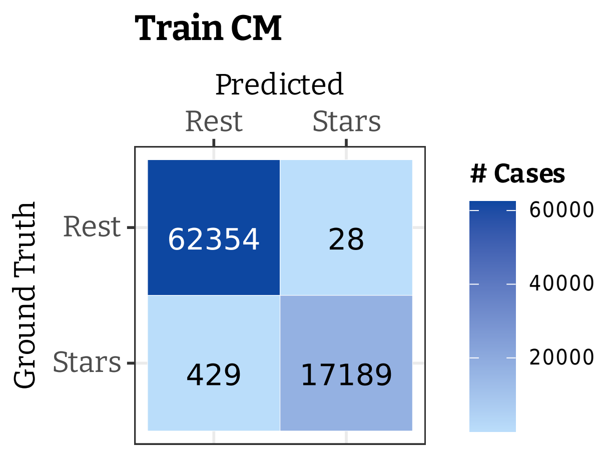

Figure 4: Confusion matrices for train and test data for classification of stars versus other classes

For the classification of quasars versus other objects, the obtained train and test ϕ coefficients are 89.2% and 89.2%, respectively. Figure 5 shows the corresponding confusion matrices. The ϕ coefficient metric is a good choice in this case because of the considerable imbalance of quasars with respect to other classes and difficulty in distinguishing quasars from galaxies based on the studied spectral characteristics.

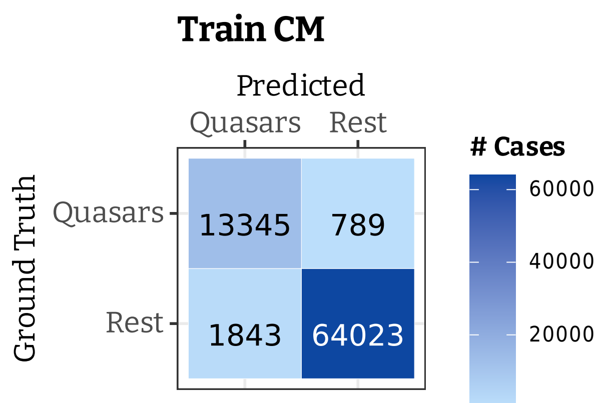

Figure 5: Confusion matrices for train and test data for classification of quasars versus other classes

Considering the considerable imbalance, the model performance is decent in distinguishing quasars from other classes.

Finally, we assess the model performance in distinguishing galaxies from other classes.

For the classification of galaxies versus other objects, the obtained train and test accuracies are 96.2% and 96.2%, respectively. Figure 6 shows the corresponding confusion matrices. The accuracy metric is a good choice in this case because the number of false positives and false negatives is similar and the number of galaxies is quite balanced with respect to the number of other classes.

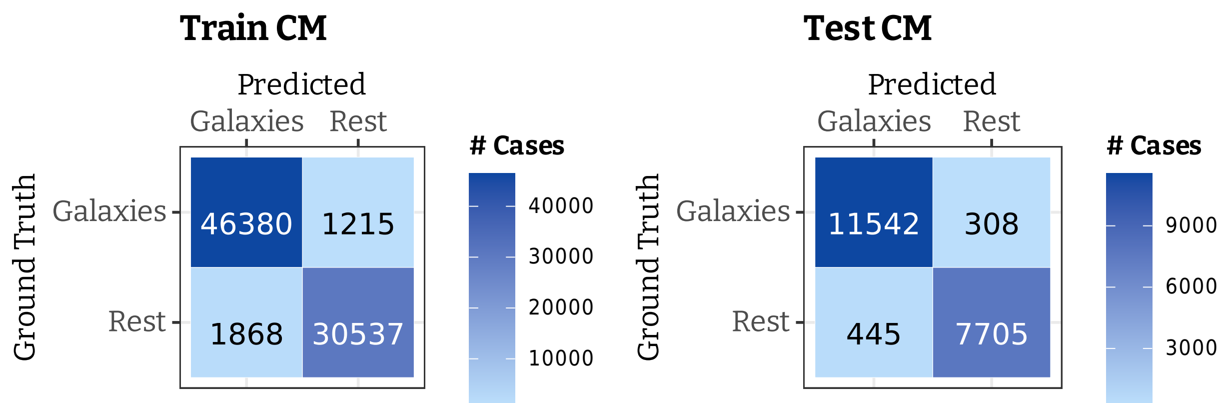

Figure 6: Confusion matrices for train and test data for classification of galaxies versus other classes

Conclusion

Summary: In this study, Stellar Classification Dataset - SDSS17 was explored. This is a multi-class classification problem with three classes: galaxies, stars, and quasars. The logistic regression model with `multinom` function from `nnet` package was utilised to solve this problem. The obtained accuracy for distinguishing stars from other classes is 99.4%. The obtained ϕ coefficient for distinguishing quasars from other classes is 89.2%. The obtained accuracy for distinguishing galaxies from other classes is 96.2%.

Strengths and limitations: We can observe that the model performance is excellent for distinguishing stars from other classes. The model performance is slightly worse but still good for distinguishing galaxies from other classes. In contrast, the model performance is bad for distinguishing quasars from other classes.

Future scope: We will attempt to increase the performance of quasars' classification without compromising the performance of stars' and galaxies' classification by improving the model. Furthermore, the performance of the model selected herein will be compared with those of other models, and the best model will be used. In addition, the features used in the model should be investigated further; for example, basis expansion can be used, and the correlated features should be explored further. Finally, the model evaluation criteria will be adjusted; for example, ROC and AUC characteristics will be explored.

References

[1] https://www.kaggle.com/datasets/fedesoriano/stellar-classification-dataset-sdss17 (The data released by the SDSS is under public domain.)

Appendix

Full code

```
#------------------------------------------------------------------#
# LIBRARIES                                                        #
#------------------------------------------------------------------#
```

```r
library(nnet) # logistic regression with multiple classes
library(ggplot2)
library(sysfonts)

## ggplot2 common theme
dejavu_layer <- list(
  theme_bw(),
  theme(
    text = element_text(family = "Bitter", face = "bold", size = 11),
    title = element_text(size = 12),
    legend.text = element_text(face = "plain", size = 11),
    axis.text = element_text(size = 10, family = "DejaVuSans", face = "plain"),
    axis.title = element_text(size = 11, face = "plain")
  )
)

#------------------------------------------------------------------#
# READING THE DATA                                                 #
#------------------------------------------------------------------#

data <- read.csv("data/star_classification.csv", stringsAsFactors = TRUE)

#------------------------------------------------------------------#
# FEATURE SELECTION                                                #
#------------------------------------------------------------------#

# Only spectroscopic features are selected
# Let us see the correlation between spectroscopic features
features <- data.frame(u = data$u,
                       r = data$g,
                       g = data$g,
                       i = data$i,
                       z = data$z,
                       spec_obj_ID = data$spec_obj_ID,
                       redshift = data$redshift)
target <- data$class

# Compute the correlation matrix
cor_mat <- cor(features)
print(round(cor_mat, 3))

# Prepare df for plotting
n_features <- dim(features)[2] # number of features
row_tiles <- factor(rep(colnames(cor_mat), n_features)) # row grid for tiles
col_tiles <- matrix(colnames(features), #col grid values for tiles
                             ncol = n_features,
                             nrow = n_features,
                             byrow = T)
col_tiles <- factor(c(col_tiles))

values <- round(c(cor_mat), 3)
df <- data.frame(row_tiles, col_tiles, values)
```

```r
plot <- ggplot(data =  df, mapping = aes(x = col_tiles, y = row_tiles)) +
  theme_minimal() +
  geom_tile(aes(fill = values), color = "black", size = 0.1) +
  geom_text(
    aes(label = values),
    vjust = 0.5, family = "DejaVuSans", size = 4
  ) +
  labs(
    title = "Correlation Matrix",
    x = NULL,
    y = NULL,
    fill = "Correlation\nScale"
  ) +
  scale_fill_gradient2(
    low = "#7321db",
    mid = "white",
    high = "#f3ae07",
    breaks = seq(-1, 1, 0.5), # control label values
    limits = c(-1, 1) # set limits feasible for correlation
  ) +
  scale_x_discrete(position = "top") +
  scale_y_discrete(limits = rev) +
  theme(
    text = element_text(family = "Bitter", face = "bold"),
    axis.text = element_text(
      family = "DejaVuSansMono",
      size = 13, face = "bold"),
    title = element_text(size = 16),
    legend.title = element_text(size = 13, margin = margin(b = 3)),
    legend.text = element_text(size = 12,
                               face = "plain",
                               family = "DejaVuSans"),
    panel.grid.major = element_blank(),
    axis.text.x.top = element_text(margin = margin(b = -3)),
    axis.text.y.left = element_text(margin = margin(r = -5)),
    legend.key.width = unit(1, 'cm'),
    legend.key.height = unit(1, "cm")
  )

print(plot)


# We observe that r, g, u, and z are strongly correlated.

#-------------------------------------------------------------------#
# DATA PREPROCESSING                                                #
#-------------------------------------------------------------------#

# Scale the features to zero mean and 1 std
features <- scale(features)

# Name preprocessed data as data_prep
data_prep <- as.data.frame(cbind(target, features))
```

```r
# Data division to train and test
set.seed(12345)
n <- dim(data_prep)[1]
tr_ind <- sample(1:n, floor(0.8*n)) # 80% training data and 20% test data
tr <- data_prep[tr_ind, ]
te <- data_prep[-tr_ind, ]

#--------------------------------------------------------------------#
# DATA VISUALISATION                                                 #
#--------------------------------------------------------------------#

# Let us visualise the distribution of data points as spec_obj_ID versus
# red_shift colored by class

plot <- ggplot(data = te) + dejavu_layer +
  geom_point(
    mapping = aes(x = redshift, y = spec_obj_ID, color = as.factor(target)),
    size = 2
  ) +
  scale_color_manual(
    values = c("orange", "#d50000", "#4527a0"),
    name = NULL,
    labels = c("Galaxies", "Quasars", "Stars")
  ) +
  labs(title = "Clusters of Galaxies, Quasars, and Stars") +
  theme(
    panel.grid.major = element_blank(),
    panel.grid.minor = element_blank(),
    axis.text.y = element_blank(),
    axis.ticks.y = element_blank(),
    axis.title = element_text(family = "DejaVuSansMono", face = "plain"),
    axis.text = element_text(color = "black")
  )

print(plot)

# We can observe three distinct clusters

#--------------------------------------------------------------------#
# LOGISTIC REGRESSION                                                #
#--------------------------------------------------------------------#

logreg <- multinom(formula = target ~ ., data = tr)

# predictions
pred_tr <- predict(logreg, tr, type = "class")
pred_te <- predict(logreg, te, type = "class")

#--------------------------------------------------------------------#
# PERFORMANCE EVALUATION                                             #
#--------------------------------------------------------------------#

CM_tr <- table(pred_tr, tr$target, deparse.level = 0)
```

```r
CM_te <- table(pred_te, te$target, deparse.level = 0)

rownames(CM_te) <- c("GAL_pr", "QUA_pr", "STA_pr")
colnames(CM_te) <- c("GAL_tr", "QUA_tr", "STA_tr")

cat("The confusion matrix for train data is\n"); print(CM_tr)
cat("The confusion matrix for test data is\n"); print(CM_te)

# Factors: 1 is GALAXY, 2 is QUASAR, 3 is STAR

# Stars-versus-rest CM
CM_tr_stars_vs_rest <- matrix(c(CM_tr[1, 1] + CM_tr[1, 2] +
                                  CM_tr[2, 1] + CM_tr[2, 2],
                                CM_tr[2, 3] + CM_tr[1, 3],
                                CM_tr[3, 1] + CM_tr[3, 2],
                                CM_tr[3, 3]), byrow = TRUE, ncol = 2)

colnames(CM_tr_stars_vs_rest) <- c("rest_tr", "STA_tr")
rownames(CM_tr_stars_vs_rest) <- c("rest_pr", "STA_pr")

print("For train data")
print(CM_tr_stars_vs_rest)

# Prepare df for plotting
GT <- factor(c("Rest", "Stars", "Rest", "Stars"))
predicted <- factor(c("Rest", "Rest", "Stars", "Stars"))
values <- c(CM_tr_stars_vs_rest)
df <- data.frame(GT, predicted, values)

# Function for plotting a confusion matrix
plot_CM <- function(df, boundary, title){
  plot <- ggplot(data =  df, mapping = aes(x = predicted, y = GT)) +
    dejavu_layer +
    geom_tile(aes(fill = values), color = "white") +
    geom_text(
      data = subset(df, values > boundary),
      aes(label = values),
      vjust = 1, color = "white", family = "DejaVuSans"
    ) +
    geom_text(
      data = subset(df, values < boundary),
      aes(label = values),
      vjust = 1, color = "black", family = "DejaVuSans"
    ) +
    labs(
      title = title,
      x = "Predicted",
      y = "Ground Truth",
      fill = "# Cases"
    ) +
    scale_fill_gradient(low = "#bbdefb", high = "#0d47a1") +
    scale_x_discrete(position = "top") +
    scale_y_discrete(limits = rev) +
```

```r
    theme(
      axis.text = element_text(family = "Bitter", size = 11)
    )

  return(list(plot))
}

# Visualise this CM
plot <- plot_CM(df = df, boundary = 40000, title = "Train CM")
print(plot)

CM_te_stars_vs_rest <- matrix(c(CM_te[1, 1] + CM_te[1, 2] +
                                  CM_te[2, 1] + CM_te[2, 2],
                                CM_te[2, 3] + CM_te[1, 3],
                                CM_te[3, 1] + CM_te[3, 2],
                                CM_te[3, 3]), byrow = TRUE, ncol = 2)

colnames(CM_te_stars_vs_rest) <- c("rest_tr", "STA_tr")
rownames(CM_te_stars_vs_rest) <- c("rest_pr", "STA_pr")

print("For test data")
print(CM_te_stars_vs_rest)

# Prepare df for plotting
GT <- factor(c("Rest", "Stars", "Rest", "Stars"))
predicted <- factor(c("Rest", "Rest", "Stars", "Stars"))
values <- c(CM_te_stars_vs_rest)
df <- data.frame(GT, predicted, values)

# Visualise this CM
plot <- plot_CM(df = df, boundary = 10000, title = "Test CM")
print(plot)

cat("Train accuracy for distinguishing between stars and other classes\n")
numerator <- CM_tr_stars_vs_rest[1, 1] + CM_tr_stars_vs_rest[2, 2]
MCR_tr <- numerator / sum(CM_tr_stars_vs_rest)
cat(round(MCR_tr*100, 1), "%", sep="")

cat("\nTest accuracy for distinguishing between stars and other classes\n")
numerator <- CM_te_stars_vs_rest[1, 1] + CM_te_stars_vs_rest[2, 2]
MCR_te <- numerator / sum(CM_te_stars_vs_rest)
cat(round(MCR_te*100, 1), "%", sep="")

# Now, let us consider distinguishing quasars from other data. The confusion
# matrix is as follows:
CM_tr_quasars_vs_rest <- matrix(c(CM_tr[1, 1] + CM_tr[1, 3] +
                                    CM_tr[3, 1] + CM_tr[3, 3],
                                CM_tr[1, 2] + CM_tr[3, 2],
                                CM_tr[2, 1] + CM_tr[2, 3],
                                CM_tr[2, 2]), byrow = TRUE, ncol = 2)

colnames(CM_tr_quasars_vs_rest) <- c("rest_tr", "QUA_tr")
rownames(CM_tr_quasars_vs_rest) <- c("rest_pr", "QUA_pr")
```

```r
print("For train data")
print(CM_tr_quasars_vs_rest)

# Prepare df for plotting
GT <- factor(c("Rest", "Quasars", "Rest", "Quasars"))
predicted <- factor(c("Rest", "Rest", "Quasars", "Quasars"))
values <- c(CM_tr_quasars_vs_rest)
df <- data.frame(GT, predicted, values)

# Visualise this CM
plot <- plot_CM(df = df, boundary = 40000, title = "Train CM")
print(plot)

CM_te_quasars_vs_rest <- matrix(c(CM_te[1, 1] + CM_te[1, 3] +
                                    CM_te[3, 1] + CM_te[3, 3],
                                CM_te[1, 2] + CM_te[3, 2],
                                CM_te[2, 1] + CM_te[2, 3],
                                CM_te[2, 2]), byrow = TRUE, ncol = 2)

colnames(CM_te_quasars_vs_rest) <- c("rest_tr", "QUA_tr")
rownames(CM_te_quasars_vs_rest) <- c("rest_pr", "QUA_pr")

print("For test data")
print(CM_te_quasars_vs_rest)

# Prepare df for plotting
GT <- factor(c("Rest", "Quasars", "Rest", "Quasars"))
predicted <- factor(c("Rest", "Rest", "Quasars", "Quasars"))
values <- c(CM_te_quasars_vs_rest)
df <- data.frame(GT, predicted, values)

# Visualise this CM
plot <- plot_CM(df = df, boundary = 6000, title = "Test CM")
print(plot)

# Here, we observe the large number of misclassification cases. To assess the
# model performance, considering the data imbalance (low number of quasars wrt
# other classes, as shown in Figure X), we use the Phi coefficient, which is a
# recommended choice for binary classification in the case of imbalanced data

phi_coef <- function(M){
  numerator <- (M[1, 1]*M[2, 2] - M[1, 2]*M[2, 1])

  MCC <- numerator/sqrt(M[1, 1] + M[1, 2])
  MCC <- MCC/sqrt(M[2, 1] + M[2, 2])
  MCC <- MCC/sqrt(M[1, 1] + M[2, 1])
  MCC <- MCC/sqrt(M[1, 2] + M[2, 2])
  return(MCC)
}

cat("Train Phi coeff for distinguishing between quasars and other classes\n",
    round(phi_coef(CM_tr_quasars_vs_rest)*100, 1), "%", sep = "")
cat("\nTest Phi coeff for distinguishing between quasars and other classes\n",
```

```
    round(phi_coef(CM_te_quasars_vs_rest)*100, 1), "%", sep = "")

# Considering the considerable imbalance, the model performance is decent in
# distinguishing quasars from other classes

# Finally, we want to assess the model performance in distinguishing galaxies
# from other classes.

CM_tr_galaxies_vs_rest <- matrix(c(CM_tr[2, 2] + CM_tr[2, 3] +
                                     CM_tr[3, 2] + CM_tr[3, 3],
                                   CM_tr[1, 2] + CM_tr[1, 3],
                                   CM_tr[2, 1] + CM_tr[3, 1],
                                   CM_tr[1, 1]), byrow = TRUE, ncol = 2)

colnames(CM_tr_galaxies_vs_rest) <- c("rest_tr", "GAL_tr")
rownames(CM_tr_galaxies_vs_rest) <- c("rest_pr", "GAL_pr")

print("For train data")
print(CM_tr_galaxies_vs_rest)

# Prepare df for plotting
GT <- factor(c("Rest", "Galaxies", "Rest", "Galaxies"))
predicted <- factor(c("Rest", "Rest", "Galaxies", "Galaxies"))
values <- c(CM_tr_galaxies_vs_rest)
df <- data.frame(GT, predicted, values)

# Visualise this CM
plot <- plot_CM(df = df, boundary = 30000, title = "Train CM")
print(plot)

CM_te_galaxies_vs_rest <- matrix(c(CM_te[2, 2] + CM_te[2, 3] +
                                     CM_te[3, 2] + CM_te[3, 3],
                                   CM_te[1, 2] + CM_te[1, 3],
                                   CM_te[2, 1] + CM_te[3, 1],
                                   CM_te[1, 1]), byrow = TRUE, ncol = 2)

colnames(CM_te_galaxies_vs_rest) <- c("rest_tr", "GAL_tr")
rownames(CM_te_galaxies_vs_rest) <- c("rest_pr", "GAL_pr")

print("For test data")
print(CM_te_galaxies_vs_rest)

# Prepare df for plotting
GT <- factor(c("Rest", "Galaxies", "Rest", "Galaxies"))
predicted <- factor(c("Rest", "Rest", "Galaxies", "Galaxies"))
values <- c(CM_te_galaxies_vs_rest)
df <- data.frame(GT, predicted, values)

# Visualise this CM
plot <- plot_CM(df = df, boundary = 6000, title = "Test CM")
print(plot)

cat("Train accuracy for distinguishing between galaxies and other classes\n")
```

```r
numerator <- CM_tr_galaxies_vs_rest[1, 1] + CM_tr_galaxies_vs_rest[2, 2]
MCR_tr <- numerator / sum(CM_tr_galaxies_vs_rest)
cat(round(MCR_tr*100, 1), "%", sep="")

cat("\nTest accuracy for distinguishing between galaxies and other classes\n")
numerator <- CM_te_galaxies_vs_rest[1, 1] + CM_te_galaxies_vs_rest[2, 2]
MCR_te <- numerator / sum(CM_te_galaxies_vs_rest)
cat(round(MCR_te*100, 1), "%", sep="")

#-------------------------------------------------------------------#
# CLASS DISTRIBUTION                                                #
#-------------------------------------------------------------------#

# Visualise the distribution of classes
classes = c("GALAXY", "STAR", "QSO")
counts = numeric(3)
for (i in 1:3){
  counts[i] = length(which(data$class == classes[i]))
}

df_classes <- data.frame(classes = classes, counts = counts)

plot <- ggplot(data = df_classes,
               mapping = aes(x = counts, y = reorder(classes, counts))) +
  dejavu_layer +
  geom_bar(stat = "identity", fill = "darkblue", alpha = .6, width = .4,
           color = "black") +
  labs(title = "Distribution of Classes", x = "", y = "") +
  theme(
    panel.grid.major = element_blank(),
    panel.grid.minor = element_blank(),
    axis.text.y = element_blank(),
    axis.ticks.y = element_blank(),
    axis.text = element_text(color = "black")
  ) +
  scale_x_continuous(expand = c(0, 10), limits = c(0, 70000)) +
  geom_text(
    aes(1000, y = classes, label = c("Galaxies", "Stars", "Quasars")),
    hjust = 0,
    nudge_x = 0,
    color = "white",
    size = 7,
    family = "DejaVuSans"
  ) +
  geom_text(
    aes(counts + 1000, y = classes, label = counts),
    hjust = 0,
    nudge_x = 0.3,
    colour = "black",
    size = 6,
    family = "DejaVuSansCondensed"
  )
```

```
print(plot)
```