Apollo ST310 Group Project

21-03-2023

Import libraries

EDA & Data Manipulation

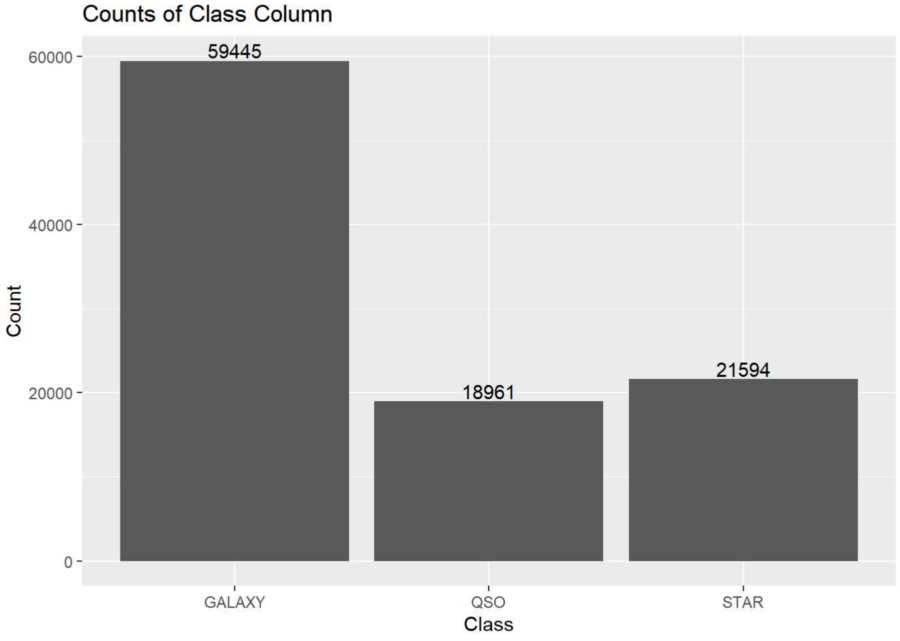
Import data

```r
# Import and view head of training data
path <- "../Data/Stellar-Classification-Dataset.csv"
raw_df <- read.csv(path)
```

```r
# Make "Class" the first column
#reordered_raw_df <- raw_df[,c(14,1:13, 15:ncol(raw_df))]
#head(reordered_raw_df)
```

Distribution of `class`

```r
ggplot(raw_df,aes(x=factor(class))) +
geom_bar() +
labs(title="Counts of Class Column", x="Class", y = "Count")+
geom_text(aes(label=..count..),stat='count', vjust=-0.2)
```



```
raw_df <- raw_df[,c(14,1:13, 15:ncol(raw_df))]
head(raw_df)
```

```
##    class      obj_ID    alpha      delta        u        g        r        i
## 1 GALAXY 1.23766e+18 135.6891 32.4946318 23.87882 22.27530 20.39501 19.16573
## 2 GALAXY 1.23766e+18 144.8261 31.2741849 24.77759 22.83188 22.58444 21.16812
## 3 GALAXY 1.23766e+18 142.1888 35.5824442 25.26307 22.66389 20.60976 19.34857
## 4 GALAXY 1.23766e+18 338.7410 -0.4028276 22.13682 23.77656 21.61162 20.50454
## 5 GALAXY 1.23768e+18 345.2826 21.1838656 19.43718 17.58028 16.49747 15.97711
## 6    QSO 1.23768e+18 340.9951 20.5894763 23.48827 23.33776 21.32195 20.25615
##          z run_ID rerun_ID cam_col field_ID spec_obj_ID  redshift plate   MJD
## 1 18.79371   3606      301       2       79 6.54378e+18 0.6347936  5812 56354
## 2 21.61427   4518      301       5      119 1.17601e+19 0.7791360 10445 58158
## 3 18.94827   3606      301       2      120 5.15220e+18 0.6441945  4576 55592
## 4 19.25010   4192      301       3      214 1.03011e+19 0.9323456  9149 58039
## 5 15.54461   8102      301       3      137 6.89186e+18 0.1161227  6121 56187
## 6 19.54544   8102      301       3      110 5.65898e+18 1.4246590  5026 55855
##   fiber_ID
## 1      171
## 2      427
## 3      299
## 4      775
## 5      842
## 6      741
```

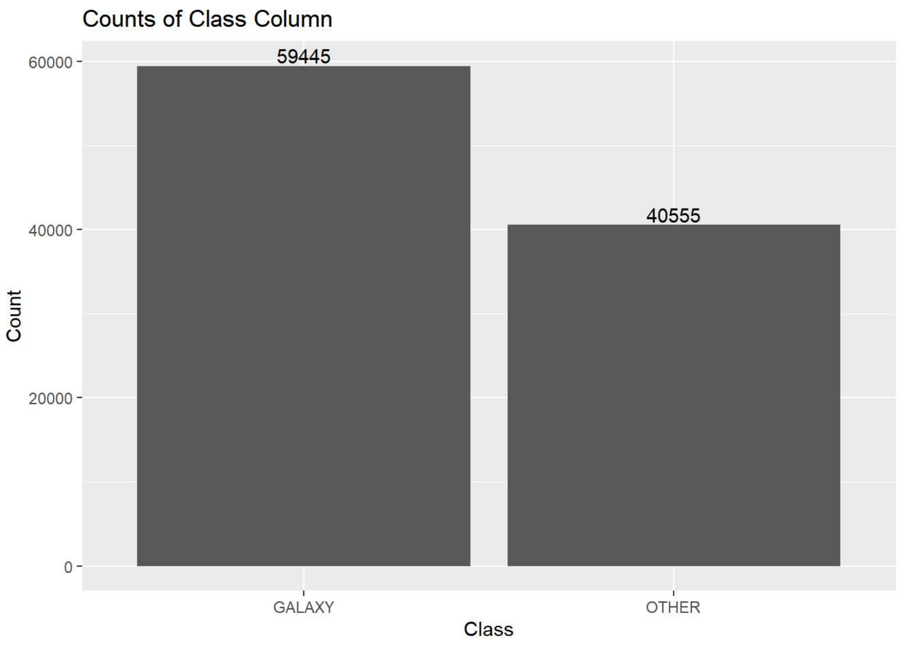
```
# Filter the data by class and subsample
#galaxy_df <- reordered_raw_df[reordered_raw_df$class == "GALAXY", ]
#subsampled_galaxy_df <- galaxy_df[sample(nrow(galaxy_df), size = 1000, replace = FALSE),]

#qso_df <- reordered_raw_df[reordered_raw_df$class == "QSO", ]
#subsampled_qso_df <- qso_df[sample(nrow(qso_df), size = 1000, replace = FALSE),]

#star_df <- reordered_raw_df[reordered_raw_df$class == "STAR", ]
#subsampled_star_df <- star_df[sample(nrow(star_df), size = 1000, replace = FALSE),]
```

```
# Create new DataFrame
#new_df <- rbind(subsampled_galaxy_df, subsampled_qso_df, subsampled_star_df)
#head(new_df)
```

Re-labelling `class`

```
raw_df[raw_df == 'QSO' | raw_df == 'STAR'] <- 'OTHER'
```

```
ggplot(raw_df,aes(x=factor(class))) +
geom_bar() +
labs(title="Counts of Class Column", x="Class", y = "Count")+
geom_text(aes(label=..count..),stat='count', vjust=-0.2)
```



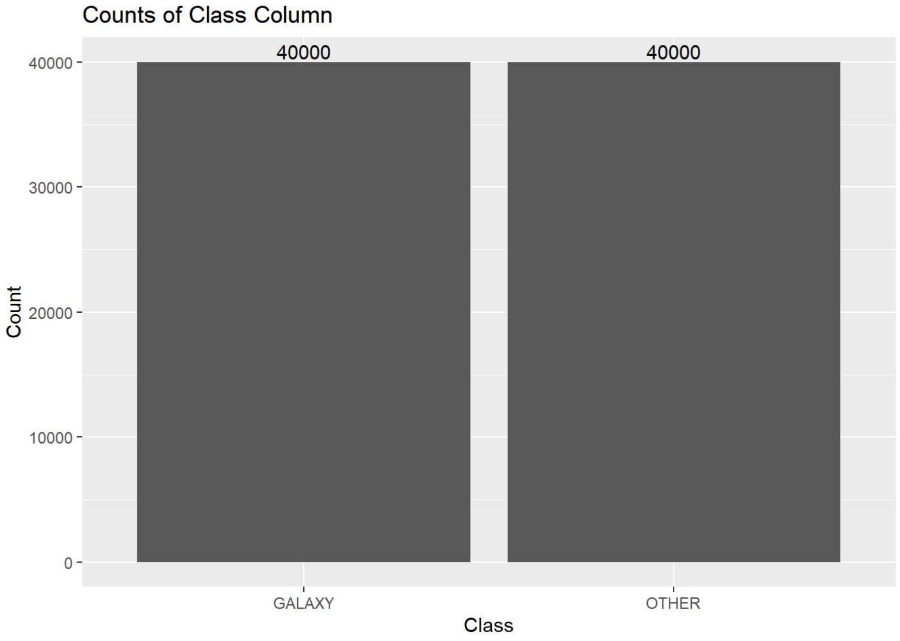
Subsample the data

```
# Filter the data by class and subsample
galaxy_df <- raw_df[raw_df$class == "GALAXY", ]
subsampled_galaxy_df <- galaxy_df[sample(nrow(galaxy_df), size = 40000, replace = FALSE),]

other_df <- raw_df[raw_df$class == "OTHER", ]
subsampled_other_df <- other_df[sample(nrow(other_df), size = 40000, replace = FALSE),]
```

```
# Create new DataFrame
df <- rbind(subsampled_galaxy_df, subsampled_other_df)
dim(df)
```

```
## [1] 80000    18
```

```
ggplot(df,aes(x=factor(class))) +
geom_bar() +
labs(title="Counts of Class Column", x="Class", y = "Count")+
geom_text(aes(label=..count..),stat='count', vjust=-0.2)
```



```
# Export data to .csv file
write.csv(df, "../Data/Binary-Subsampled-Data.csv", row.names=FALSE)
```

Summary statistics

```
df %>%
  skimr::skim(colnames(df))
```

Data summary

Name	Piped data
Number of rows	80000
Number of columns	18
________________________	
Column type frequency:	
character	1
numeric	17
________________________	
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
class	0	1	5	6	0	2	0

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hi
obj_ID	0	1	1.237687e+18	2.300960e+14	1.23765e+18	1.237660e+18	1.237660e+18	1.23767e+18	1.24000e+18	▇
alpha	0	1	1.774600e+02	9.700000e+01	1.00000e-02	1.265300e+02	1.805700e+02	2.34690e+02	3.60000e+02	▅
delta	0	1	2.429000e+01	1.972000e+01	-1.87900e+01	5.140000e+00	2.410000e+01	4.02900e+01	8.30000e+01	▅
u	0	1	2.184000e+01	3.550000e+01	-9.99900e+03	2.028000e+01	2.201000e+01	2.35400e+01	3.27800e+01	▁
g	0	1	2.044000e+01	3.548000e+01	-9.99900e+03	1.897000e+01	2.100000e+01	2.20300e+01	3.06100e+01	▁
r	0	1	1.966000e+01	1.850000e+00	9.82000e+00	1.821000e+01	2.013000e+01	2.10600e+01	2.95700e+01	▁▂
i	0	1	1.914000e+01	1.770000e+00	9.47000e+00	1.782000e+01	1.946000e+01	2.04900e+01	3.21400e+01	▁▃
z	0	1	1.872000e+01	3.546000e+01	-9.99900e+03	1.756000e+01	1.907000e+01	2.00600e+01	2.87900e+01	▁
run_ID	0	1	4.471560e+03	1.967950e+03	1.09000e+02	3.180000e+03	4.188000e+03	5.33000e+03	8.16200e+03	▂
rerun_ID	0	1	3.010000e+02	0.000000e+00	3.01000e+02	3.010000e+02	3.010000e+02	3.01000e+02	3.01000e+02	▁
cam_col	0	1	3.510000e+00	1.590000e+00	1.00000e+00	2.000000e+00	4.000000e+00	5.00000e+00	6.00000e+00	▇
field_ID	0	1	1.850700e+02	1.477700e+02	1.10000e+01	8.200000e+01	1.460000e+02	2.39000e+02	9.89000e+02	▇
spec_obj_ID	0	1	5.853644e+18	3.343783e+18	2.99519e+17	2.902725e+18	5.646555e+18	8.38805e+18	1.41269e+19	▇
redshift	0	1	6.100000e-01	8.000000e-01	-1.00000e-02	0.000000e+00	4.100000e-01	7.70000e-01	7.01000e+00	▇
plate	0	1	5.198970e+03	2.969860e+03	2.66000e+02	2.578000e+03	5.015000e+03	7.45000e+03	1.25470e+04	▇
MJD	0	1	5.562663e+04	1.806680e+03	5.16080e+04	5.438000e+04	5.589400e+04	5.69470e+04	5.89320e+04	▂
fiber_ID	0	1	4.487200e+02	2.722300e+02	1.00000e+00	2.210000e+02	4.320000e+02	6.44000e+02	1.00000e+03	▇

Model Recipe

Train/Test Split

```
set.seed(222)
# Put 80% of the data into the training set
data_split <- initial_split(df, prop = 0.8)

# Create data frames for the two sets:
train_data <- training(data_split)
test_data  <- testing(data_split)
```

Create our recipe

```
# Declare the ID variables
IDs <- c("obj_ID", "run_ID", "rerun_ID", "cam_col", "field_ID", "spec_obj_ID", "plate", "MJD", "fiber_ID")
```

```
# Define our model, and exclude the ID variables
recipe <-
  recipe(class ~ ., data = train_data) %>%
  update_role(all_of(IDs), new_role = "ID")
```

```
# Summary of the recipe
summary(recipe)
```

```
## # A tibble: 18 × 4
##    variable    type    role      source
##    <chr>       <chr>   <chr>     <chr>
##  1 obj_ID      numeric ID        original
##  2 alpha       numeric predictor original
##  3 delta       numeric predictor original
##  4 u           numeric predictor original
##  5 g           numeric predictor original
##  6 r           numeric predictor original
##  7 i           numeric predictor original
##  8 z           numeric predictor original
##  9 run_ID      numeric ID        original
## 10 rerun_ID    numeric ID        original
## 11 cam_col     numeric ID        original
## 12 field_ID    numeric ID        original
## 13 spec_obj_ID numeric ID        original
## 14 redshift    numeric predictor original
## 15 plate       numeric ID        original
## 16 MJD         numeric ID        original
## 17 fiber_ID    numeric ID        original
## 18 class       nominal outcome   original
```

Logisitic Regression Model

```
# Define our logistic regression model
logistic_model <-
  logistic_reg() %>%
  set_engine("glm")
```

Workflow

```
# Create our workflow
logistic_wflow <-
  workflow() %>%
  add_model(logistic_model) %>%
  add_recipe(recipe)

logistic_wflow
```

```
## ══ Workflow ════════════════════════════════════════════════════════════════
## Preprocessor: Recipe
## Model: logistic_reg()
##
## ── Preprocessor ────────────────────────────────────────────────────────────
## 0 Recipe Steps
##
## ── Model ───────────────────────────────────────────────────────────────────
## Logistic Regression Model Specification (classification)
##
## Computational engine: glm
```

Fit and explore the model

```
# Fit the model
logistic_fit <-
  logistic_wflow %>%
  fit(data = train_data)
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
logistic_fit %>%
  extract_fit_parsnip() %>%
  tidy()
```

```
## # A tibble: 9 × 5
##   term          estimate std.error statistic  p.value
##   <chr>            <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept) -0.375      0.123       -3.05  2.33e- 3
## 2 alpha       -0.0000600  0.000104    -0.575 5.66e- 1
## 3 delta        0.00471    0.000510     9.24  2.52e-20
## 4 u            0.0468     0.0109       4.32  1.59e- 5
## 5 g           -1.88       0.0318     -59.1   0
## 6 r            0.995      0.0521      19.1   2.53e-81
## 7 i            0.885      0.0492      18.0   2.72e-72
## 8 z            0.0670     0.0302       2.22  2.65e- 2
## 9 redshift     0.268      0.0166      16.1   1.77e-58
```

```
logistic_augment <-
  augment(logistic_fit, test_data, type = "prob")
```

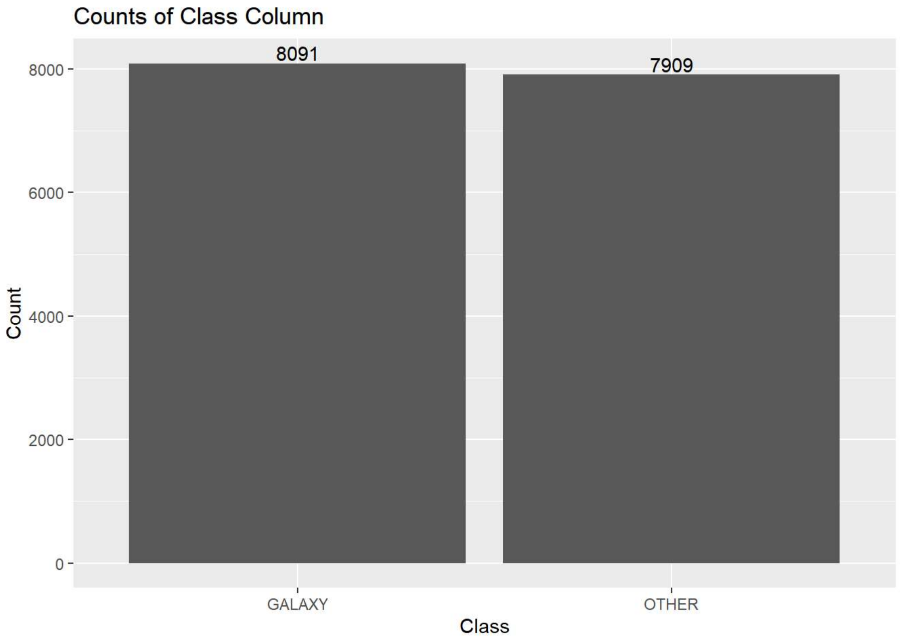
Predictions

```
# DataFrame of prediction probabilities
pred_df <- logistic_augment %>%
  select(class, .pred_class, .pred_GALAXY)
```

```
pred_df
```

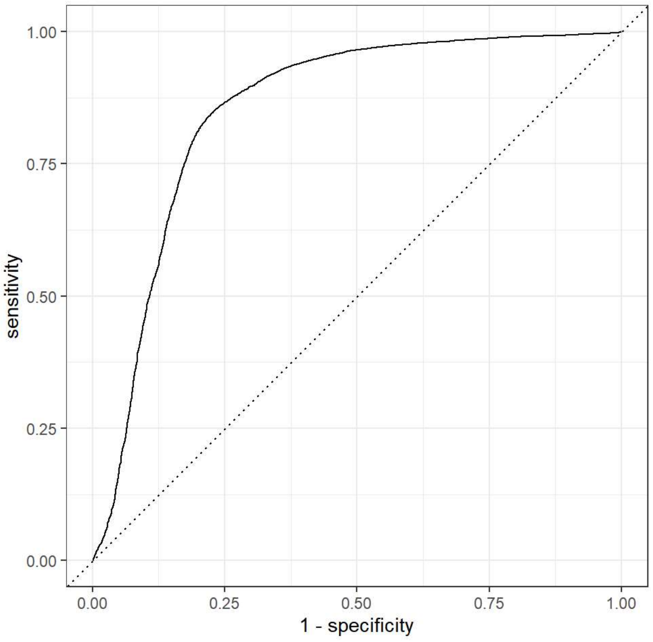
```
## # A tibble: 16,000 × 3
##    class  .pred_class .pred_GALAXY
##    <chr>  <fct>              <dbl>
##  1 GALAXY GALAXY             0.648
##  2 GALAXY GALAXY             0.972
##  3 GALAXY GALAXY             0.582
##  4 GALAXY GALAXY             0.948
##  5 GALAXY GALAXY             0.705
##  6 GALAXY GALAXY             0.582
##  7 GALAXY GALAXY             0.657
##  8 GALAXY GALAXY             0.797
##  9 GALAXY GALAXY             0.881
## 10 GALAXY GALAXY             0.828
## # … with 15,990 more rows
```

```
# Distribution of predictions
ggplot(pred_df,aes(x=factor(class))) +
geom_bar() +
labs(title="Counts of Class Column", x="Class", y = "Count")+
geom_text(aes(label=..count..),stat='count', vjust=-0.2)
```



Evaluation metric: ROC/AUC

```
logistic_augment %>%
  roc_curve(truth = as.factor(class), .pred_GALAXY) %>%
  autoplot()
```



```
# Area Under Curve (AUC)
logistic_augment %>%
  roc_auc(truth = as.factor(class), .pred_GALAXY)
```

```
## # A tibble: 1 × 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.851
```

Evaluation Metric: Accuracy Score

```
accuracy(pred_df, as.factor(class), as.factor(.pred_class))
```

```
## # A tibble: 1 × 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.803
```

Random Forest Model

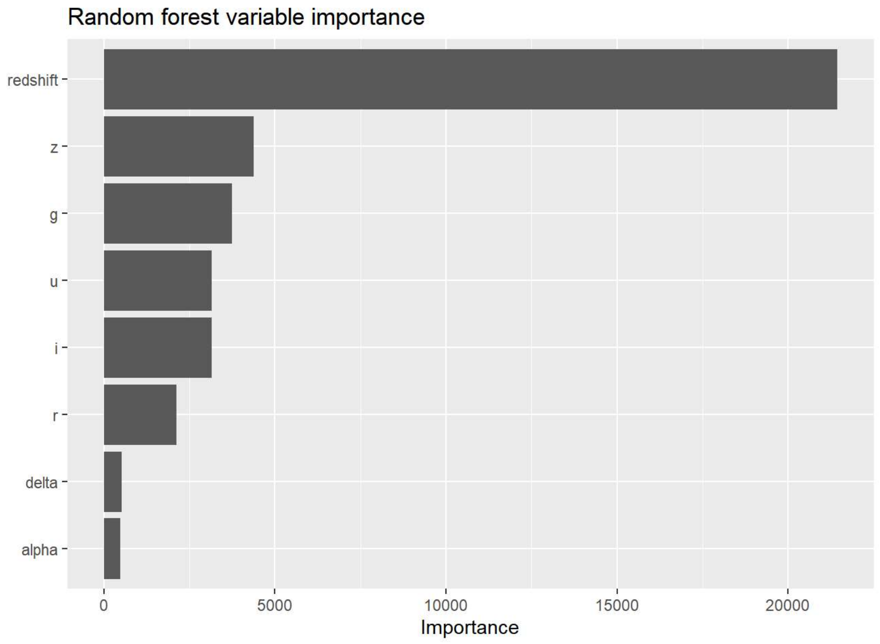
Create model

```
rf_model <- rand_forest(mode = "classification", trees = 20) %>%
  set_engine("ranger", importance = "impurity")
```

```
rf_workflow <-
  workflow() %>%
  add_model(rf_model) %>%
  add_recipe(recipe)
```

Fit model and examine variable importance

```
rf_workflow %>%
  fit(df) %>%
  extract_fit_parsnip() %>%
  vip(num_features = 8) +
  labs(title = "Random forest variable importance")
```


Random forest variable importance

Predictions

```
# Fit the model
rf_fit <-
  rf_workflow %>%
  fit(data = train_data)
```

```
rf_augment <-
  augment(rf_fit, test_data, type = "prob")
```

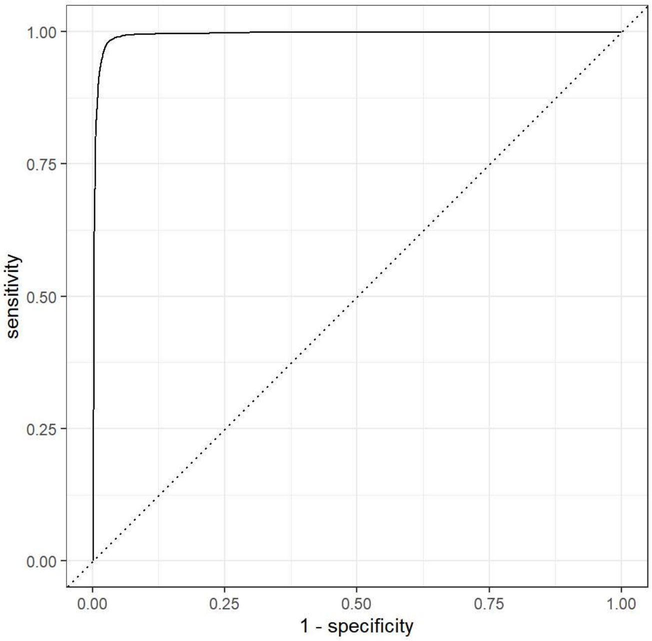
```
# DataFrame of prediction probabilities
rf_pred_df <- rf_augment %>%
  select(class, .pred_class, .pred_GALAXY)
```

```
rf_pred_df
```

```
## # A tibble: 16,000 × 3
##    class  .pred_class .pred_GALAXY
##    <chr>  <fct>              <dbl>
##  1 GALAXY GALAXY             0.938
##  2 GALAXY GALAXY             1
##  3 GALAXY GALAXY             0.929
##  4 GALAXY GALAXY             1
##  5 GALAXY GALAXY             1
##  6 GALAXY GALAXY             0.983
##  7 GALAXY GALAXY             1
##  8 GALAXY GALAXY             0.983
##  9 GALAXY GALAXY             1
## 10 GALAXY GALAXY             1
## # … with 15,990 more rows
```

Evaluation metric: ROC/AUC

```
rf_augment %>%
  roc_curve(truth = as.factor(class), .pred_GALAXY) %>%
  autoplot()
```



```
# Area Under Curve (AUC)
rf_augment %>%
  roc_auc(truth = as.factor(class), .pred_GALAXY)
```

```
## # A tibble: 1 × 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.993
```

Evaluation Metric: Accuracy Score

```
accuracy(rf_pred_df, as.factor(class), as.factor(.pred_class))
```

```
## # A tibble: 1 × 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.975
```

10-fold Cross-Validation

```
folds <- vfold_cv(train_data, v = 10)
```

```
rf_random_samples <-
  rf_workflow %>%
  fit_resamples(folds)
```

```
rf_random_samples
```

```
## # Resampling results
## # 10-fold cross-validation
## # A tibble: 10 × 4
##    splits               id     .metrics         .notes
##    <list>               <chr>  <list>           <list>
##  1 <split [57600/6400]> Fold01 <tibble [2 × 4]> <tibble [0 × 3]>
##  2 <split [57600/6400]> Fold02 <tibble [2 × 4]> <tibble [0 × 3]>
##  3 <split [57600/6400]> Fold03 <tibble [2 × 4]> <tibble [0 × 3]>
##  4 <split [57600/6400]> Fold04 <tibble [2 × 4]> <tibble [0 × 3]>
##  5 <split [57600/6400]> Fold05 <tibble [2 × 4]> <tibble [0 × 3]>
##  6 <split [57600/6400]> Fold06 <tibble [2 × 4]> <tibble [0 × 3]>
##  7 <split [57600/6400]> Fold07 <tibble [2 × 4]> <tibble [0 × 3]>
##  8 <split [57600/6400]> Fold08 <tibble [2 × 4]> <tibble [0 × 3]>
##  9 <split [57600/6400]> Fold09 <tibble [2 × 4]> <tibble [0 × 3]>
## 10 <split [57600/6400]> Fold10 <tibble [2 × 4]> <tibble [0 × 3]>
```

```
collect_metrics(rf_random_samples)
```

```
## # A tibble: 2 × 6
##   .metric  .estimator  mean     n  std_err .config
##   <chr>    <chr>      <dbl> <int>    <dbl> <chr>
## 1 accuracy binary     0.975    10 0.000971 Preprocessor1_Model1
## 2 roc_auc  binary     0.993    10 0.000409 Preprocessor1_Model1
```