brca_classification

2024-12-02

Load Libraries

```
library(tidyverse)
library(caret)
library(ggplot2)
library(randomForest)
library(xgboost)
library(rpart); library(rpart.plot)
```

1. Data Import

```
merged_data <- read.csv('r_output/merged_data.csv')
gene_metadata <- read.csv('glycoenzyme_genes.csv')
gene_list <- read.csv('glycoenzyme_gene_list.csv') %>% unlist
```

2. Classification Models

2.1. Data Preparation

```
# Define predictor variables (gene expressions and metadata columns)
predictors <- merged_data[, 2:183]
colnames(merged_data)[183:184] # Should see last enzyme and first metadata
```

```
## [1] "ST6GALNAC4"                          "age_at_initial_pathologic_diagnosis"
```

```
# Define response variable (PAM50)
response <- merged_data$PAM50Call_RNAseq

# cbind genes and predictors for rpart()
rpart.df <- cbind(response, predictors)
```

```
# 4. Split Data into Training and Testing Sets
set.seed(1234)
train_index <- createDataPartition(response, p = 0.8, list = FALSE)

train_data <- predictors[train_index, ]
train_labels <- response[train_index] %>% as.factor

test_data <- predictors[-train_index, ]
test_labels <- response[-train_index] %>% as.factor
```

Check Class Balance

```
# Class balance check
table(response)
```

```
## response
##  Basal   Her2   LumA   LumB Normal
##    141     67    433    194    118
```

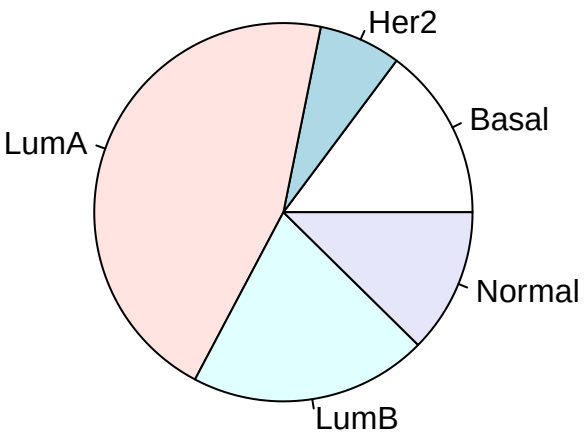
```
table(train_labels)
```

```
## train_labels
##  Basal   Her2   LumA   LumB Normal
##    113     54    347    156     95
```

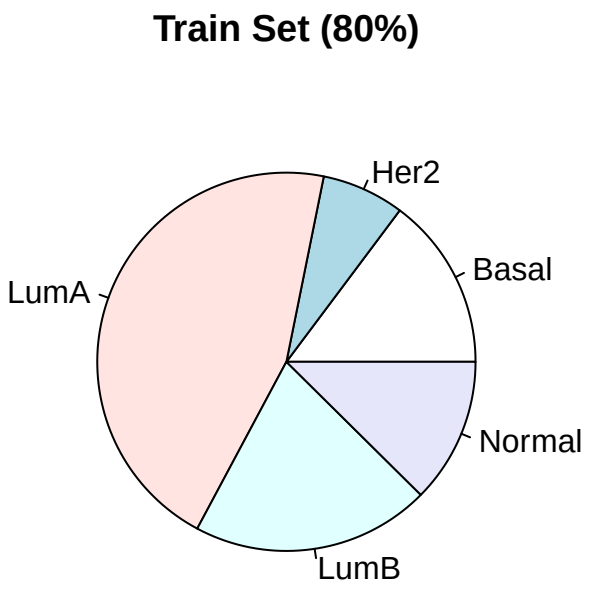
```
table(test_labels)
```

```
## test_labels
##  Basal   Her2   LumA   LumB Normal
##     28     13     86     38     23
```

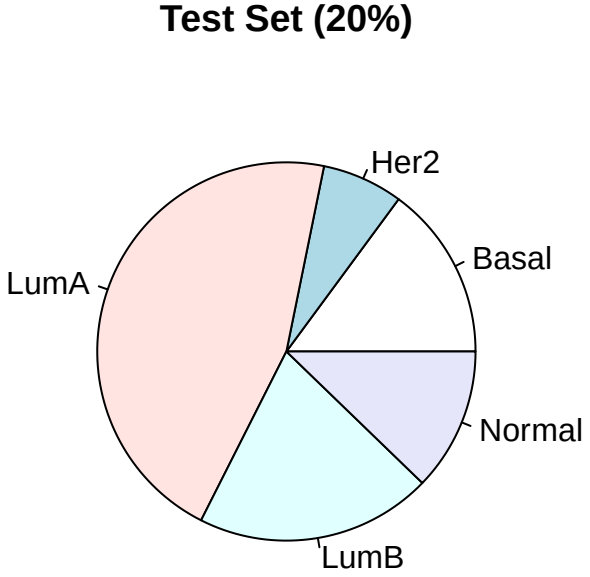
```
table(response) %>% pie(main = "Whole Data (n = 953)")
```




```
table(train_labels) %>% pie(main = "Train Set (80%)")
```




```
table(test_labels) %>% pie(main = "Test Set (20%)")
```



Defining train control

```
tc1 <- trainControl(method = "cv", number = 10)
```

2.2. Random Forest

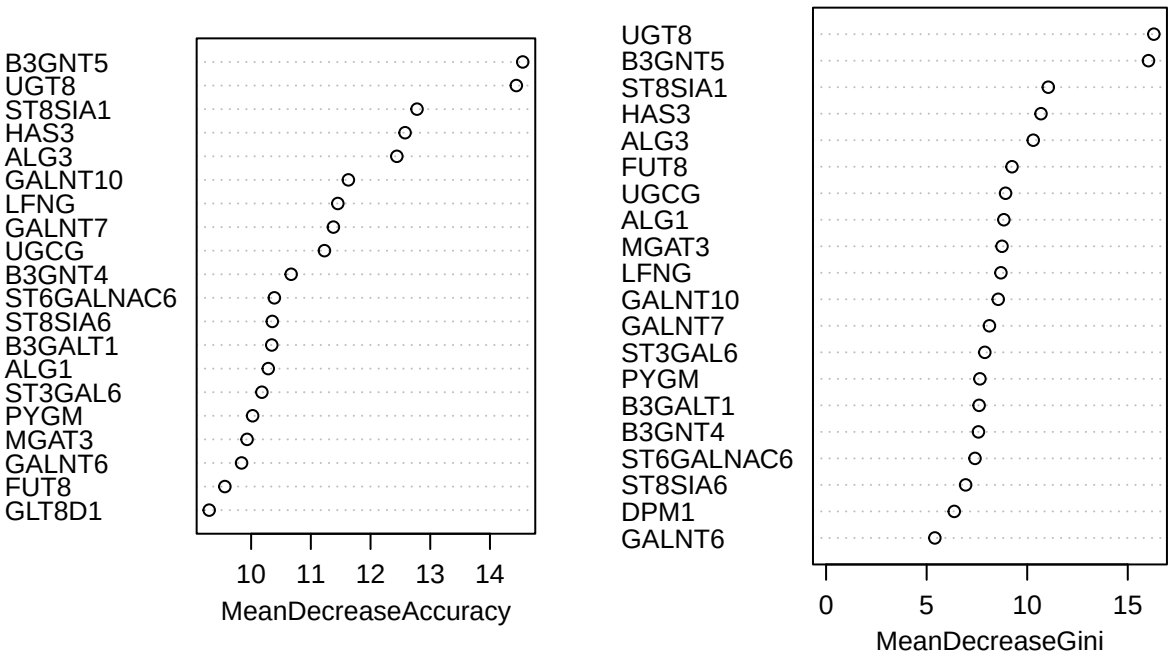
```
# 5. Train Random Forest Model
set.seed(1234)
rf_pre <- randomForest(
  x = train_data,
  y = train_labels,
  ntree = 500,                # Number of trees
  mtry = sqrt(ncol(train_data)), # Number of predictors sampled at each split
  importance = TRUE           # Calculate variable importance
```

```
)
```

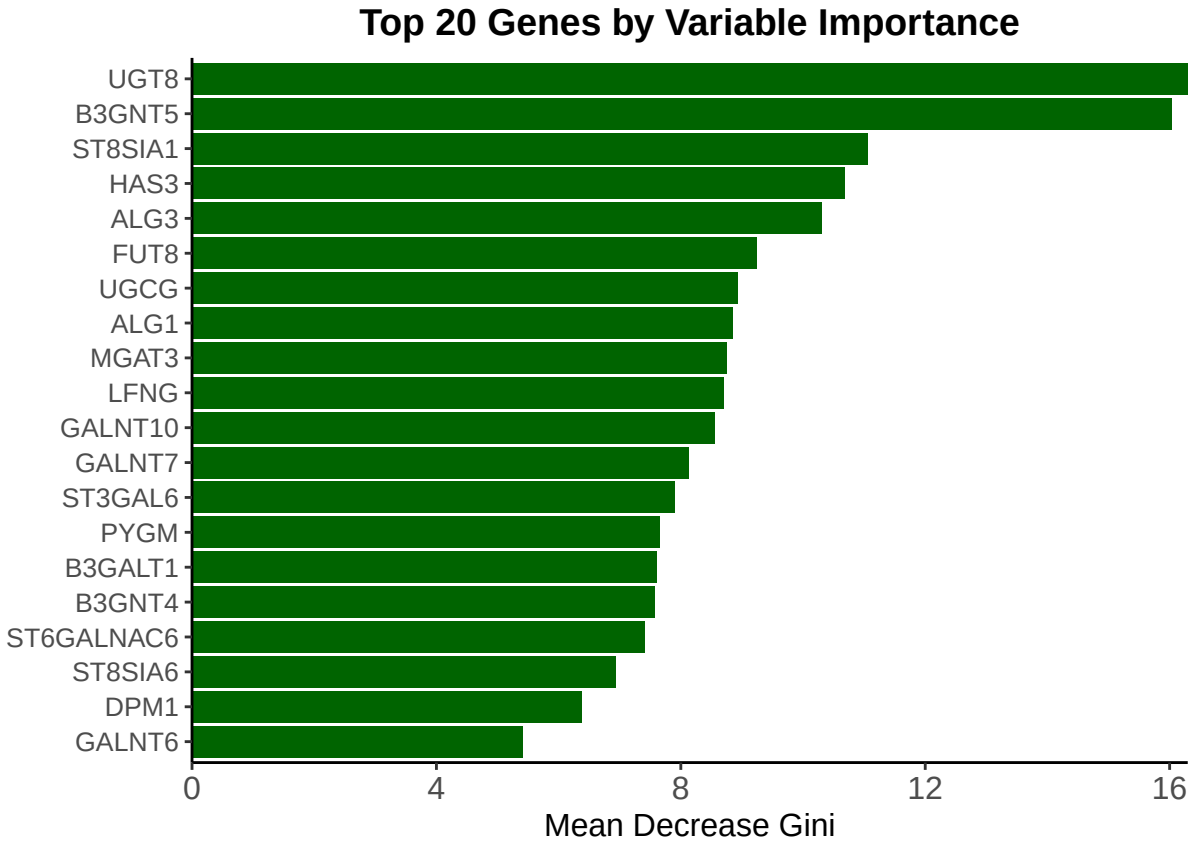
```
# Evaluate the Model
t_predictions <- predict(rf_pre, test_data)

# Confusion Matrix
confusionMatrix(predict(rf_pre, test_data), test_labels)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Basal Her2 LumA LumB Normal
##     Basal     28    1    0    1      2
##     Her2       0    6    0    0      0
##     LumA       0    5   85   17      2
##     LumB       0    1    1   20      0
##     Normal     0    0    0    0     19
##
## Overall Statistics
##
##                Accuracy : 0.8404
##                  95% CI : (0.7801, 0.8897)
##     No Information Rate : 0.4574
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7623
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: Basal Class: Her2 Class: LumA Class: LumB
## Sensitivity                1.0000     0.46154      0.9884      0.5263
## Specificity                0.9750     1.00000      0.7647      0.9867
## Pos Pred Value             0.8750     1.00000      0.7798      0.9091
## Neg Pred Value             1.0000     0.96154      0.9873      0.8916
## Prevalence                 0.1489     0.06915      0.4574      0.2021
## Detection Rate             0.1489     0.03191      0.4521      0.1064
## Detection Prevalence       0.1702     0.03191      0.5798      0.1170
## Balanced Accuracy          0.9875     0.73077      0.8765      0.7565
##                      Class: Normal
## Sensitivity                 0.8261
## Specificity                 1.0000
## Pos Pred Value              1.0000
## Neg Pred Value              0.9763
## Prevalence                  0.1223
## Detection Rate              0.1011
## Detection Prevalence        0.1011
## Balanced Accuracy           0.9130
```

```
# Variable Importance
var_importance <- importance(rf_pre) %>% as.data.frame
varImpPlot(rf_pre, n.var = 20, cex = 0.8, main = "Variable Importance Plot - RF")
```




```r
# Variable importance plot using ggplot
top_genes <- var_importance %>% arrange(desc(MeanDecreaseGini)) %>% head(20)
top_genes$gene <- rownames(top_genes)

# Create horizontal bar chart
ggplot(top_genes, aes(x = reorder(gene, MeanDecreaseGini), y = MeanDecreaseGini)) +
  geom_bar(stat = "identity", fill = "darkgreen") +
  coord_flip() +
  labs(
    title = "Top 20 Genes by Variable Importance",
    x = "",
    y = "Mean Decrease Gini"
  ) +
  theme_classic() +
  theme(
    axis.text.x = element_text(size = 12),
    axis.text.y = element_text(size = 10),
    axis.title = element_text(size = 12),
    plot.title = element_text(hjust = 0.5, size = 14, face = "bold")
  ) +
  scale_y_continuous(expand = c(0, 0))
```

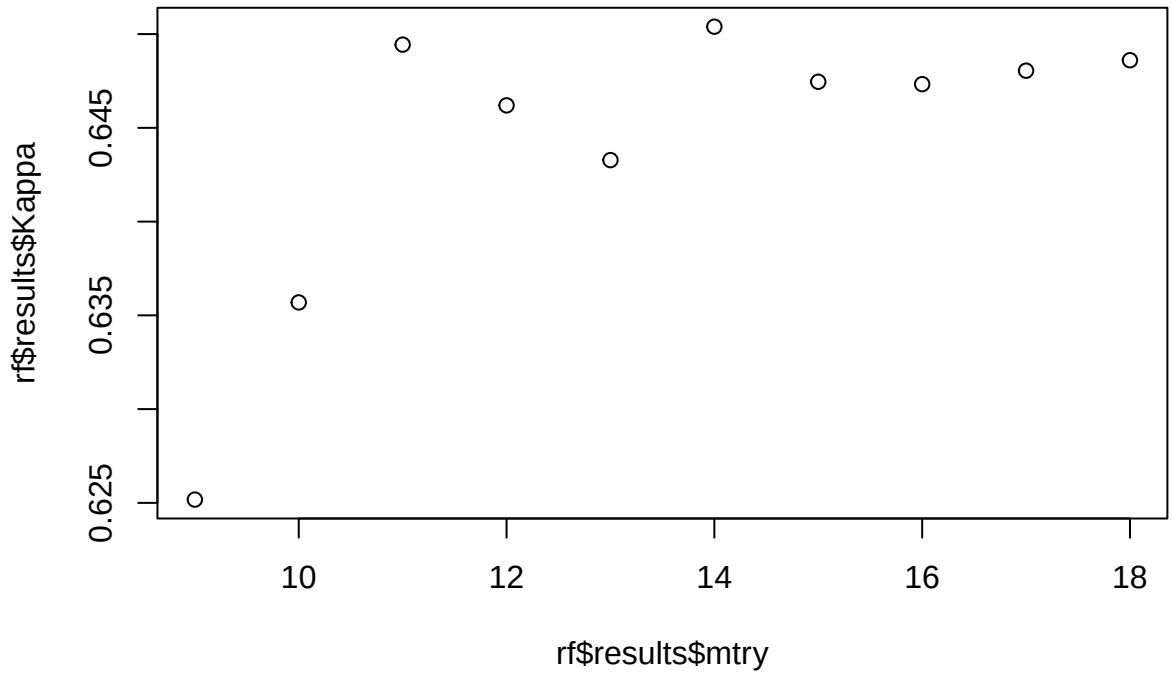



```
set.seed(1234)
rf <- train(x = train_data, y = train_labels,
            method = "rf",
            ntree = 500, tuneGrid = data.frame(mtry = 9:18),
            trControl = tc1)
```

```
rf$results
```

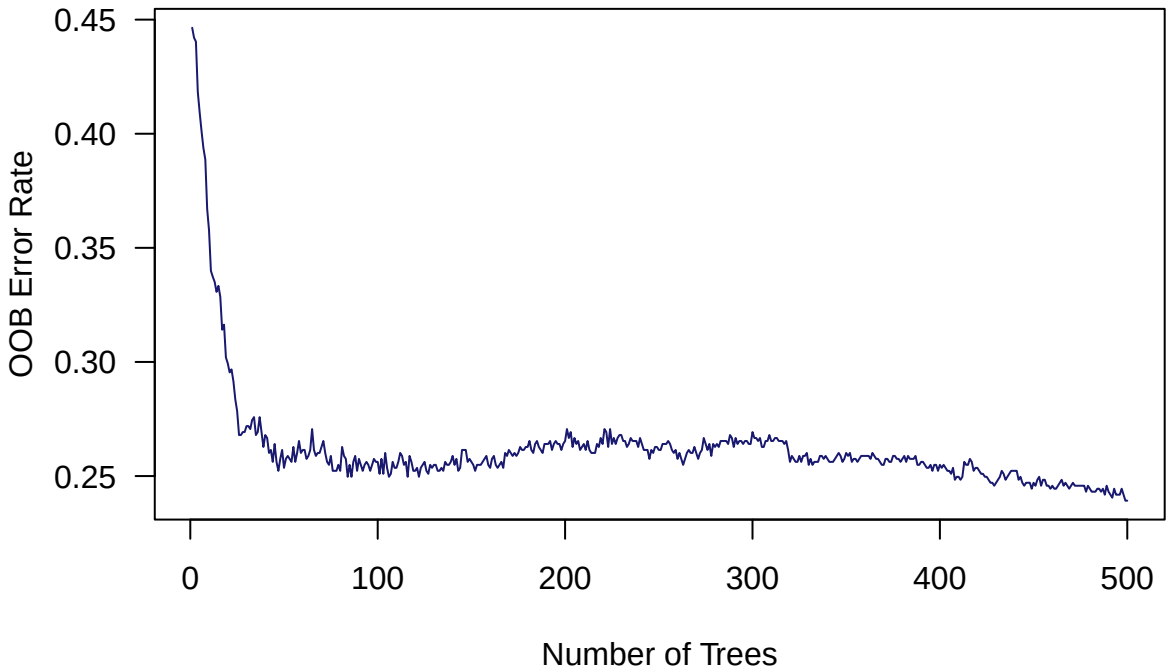
```
##    mtry  Accuracy     Kappa AccuracySD    KappaSD
## 1     9 0.7503515 0.6251753 0.01146662 0.01725901
## 2    10 0.7568634 0.6356897 0.02746549 0.04206456
## 3    11 0.7660964 0.6494375 0.02464839 0.03833859
## 4    12 0.7633797 0.6461985 0.02330685 0.03496256
## 5    13 0.7609138 0.6432781 0.02456293 0.03823207
## 6    14 0.7660437 0.6503948 0.02178500 0.03403484
## 7    15 0.7634131 0.6474575 0.03141803 0.04620822
## 8    16 0.7634819 0.6473285 0.02453084 0.03666983
## 9    17 0.7635836 0.6480488 0.03289227 0.04891232
## 10   18 0.7648998 0.6486068 0.02759232 0.04346279
```

```
plot(rf$results$mtry, rf$results$Kappa)
```



rf determining best n.tree

```
plot(x = c(1:500), y = rf$finalModel$err.rate[, 1],
xlab = "Number of Trees", ylab = "OOB Error Rate",
type = "l", col = "midnightblue", las = 1)
```




```
confusionMatrix(predict(rf, test_data), test_labels)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Basal Her2 LumA LumB Normal
```

```
##     Basal     28    1    0    1      3
##     Her2       0    6    0    0      0
##     LumA       0    5   85   14      2
##     LumB       0    1    1   23      0
##     Normal     0    0    0    0     18
##
## Overall Statistics
##
##                Accuracy : 0.8511
##                  95% CI : (0.792, 0.8987)
##     No Information Rate : 0.4574
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7794
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: Basal Class: Her2 Class: LumA Class: LumB
## Sensitivity                1.0000     0.46154      0.9884      0.6053
## Specificity                0.9688     1.00000      0.7941      0.9867
## Pos Pred Value             0.8485     1.00000      0.8019      0.9200
## Neg Pred Value             1.0000     0.96154      0.9878      0.9080
## Prevalence                 0.1489     0.06915      0.4574      0.2021
## Detection Rate             0.1489     0.03191      0.4521      0.1223
## Detection Prevalence       0.1755     0.03191      0.5638      0.1330
## Balanced Accuracy          0.9844     0.73077      0.8912      0.7960
##                      Class: Normal
## Sensitivity                0.78261
## Specificity                1.00000
## Pos Pred Value             1.00000
## Neg Pred Value             0.97059
## Prevalence                 0.12234
## Detection Rate             0.09574
## Detection Prevalence       0.09574
## Balanced Accuracy          0.89130
```

Performance

```r
# Saving performance metrics
models <- data.frame("Model" = "Random Forest", "Accuracy" = 0.8564, "Kappa" = 0.7884)
```

2.3. Support Vector Machine

```r
# set up tuning grid
tg_svmlin <- expand.grid(C = c(0.001, 0.01, 0.1 , 1, 10, 100))


set.seed(1234)
svmlin <- train(x = train_data, y = train_labels,
method = 'svmLinear', tuneGrid = tg_svmlin,
trControl = tc1)
```

```
svmlin$results
```

```
##       C  Accuracy     Kappa AccuracySD    KappaSD
## 1 1e-03 0.7596151 0.6375590 0.02025877 0.03352202
## 2 1e-02 0.8172847 0.7377489 0.03507766 0.05226883
## 3 1e-01 0.8057560 0.7259253 0.04978846 0.06943702
## 4 1e+00 0.7729738 0.6816854 0.03993984 0.05450255
## 5 1e+01 0.7729738 0.6816854 0.03993984 0.05450255
## 6 1e+02 0.7729738 0.6816854 0.03993984 0.05450255
```

```
confusionMatrix(predict(svmlin, newdata = test_data), test_labels)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Basal Her2 LumA LumB Normal
##     Basal     28    1    0    0      2
##     Her2       0    9    0    2      0
##     LumA       0    2   81   12      1
##     LumB       0    1    5   24      0
##     Normal     0    0    0    0     20
##
## Overall Statistics
##
##                Accuracy : 0.8617
##                  95% CI : (0.804, 0.9076)
##     No Information Rate : 0.4574
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8003
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: Basal Class: Her2 Class: LumA Class: LumB
## Sensitivity                1.0000     0.69231      0.9419      0.6316
## Specificity                0.9812     0.98857      0.8529      0.9600
## Pos Pred Value             0.9032     0.81818      0.8437      0.8000
## Neg Pred Value             1.0000     0.97740      0.9457      0.9114
## Prevalence                 0.1489     0.06915      0.4574      0.2021
## Detection Rate             0.1489     0.04787      0.4309      0.1277
## Detection Prevalence       0.1649     0.05851      0.5106      0.1596
## Balanced Accuracy          0.9906     0.84044      0.8974      0.7958
##                      Class: Normal
## Sensitivity                 0.8696
## Specificity                 1.0000
## Pos Pred Value              1.0000
## Neg Pred Value              0.9821
## Prevalence                  0.1223
## Detection Rate              0.1064
## Detection Prevalence        0.1064
## Balanced Accuracy           0.9348
```

Performance (linear)

```
# Saving performance metrics
models <- data.frame("Model" = "SVM Linear", "Accuracy" = 0.8617,
                     "Kappa" = 0.8003) %>% rbind(models)
```

```
set.seed(1234)
svmpoly <- train(x = train_data, y = train_labels,
method = 'svmPoly', tuneLength = 5,
trControl = tc1)
```

```
svmpoly$results
```

```
##    degree scale    C  Accuracy     Kappa AccuracySD    KappaSD
## 1       1 1e-03 0.25 0.6703405 0.4699451 0.02074663 0.04193416
## 2       1 1e-03 0.50 0.6953954 0.5198216 0.01702726 0.02813840
## 3       1 1e-03 1.00 0.7596151 0.6375590 0.02025877 0.03352202
## 4       1 1e-03 2.00 0.8001439 0.7087882 0.03314568 0.05057065
## 5       1 1e-03 4.00 0.8158690 0.7343100 0.03329950 0.05075243
## 6       1 1e-02 0.25 0.8145344 0.7309370 0.03254721 0.04929481
## 7       1 1e-02 0.50 0.8106203 0.7284222 0.03485764 0.05106069
## 8       1 1e-02 1.00 0.8172847 0.7377489 0.03507766 0.05226883
## 9       1 1e-02 2.00 0.8211480 0.7447916 0.03713087 0.05398954
## 10      1 1e-02 4.00 0.8161381 0.7384860 0.04748589 0.06692023
## 11      1 1e-01 0.25 0.8185164 0.7411406 0.03060279 0.04325947
## 12      1 1e-01 0.50 0.8149394 0.7374198 0.05309663 0.07517639
## 13      1 1e-01 1.00 0.8057560 0.7259253 0.04978846 0.06943702
## 14      1 1e-01 2.00 0.7966621 0.7140324 0.05191341 0.07168271
## 15      1 1e-01 4.00 0.7862370 0.6991405 0.05119163 0.07162191
## 16      1 1e+00 0.25 0.7875006 0.7016576 0.05129850 0.07096686
## 17      1 1e+00 0.50 0.7769734 0.6872594 0.04324568 0.05980399
## 18      1 1e+00 1.00 0.7729738 0.6816854 0.03993984 0.05450255
## 19      1 1e+00 2.00 0.7729738 0.6816854 0.03993984 0.05450255
## 20      1 1e+00 4.00 0.7729738 0.6816854 0.03993984 0.05450255
## 21      1 1e+01 0.25 0.7729738 0.6816854 0.03993984 0.05450255
## 22      1 1e+01 0.50 0.7729738 0.6816854 0.03993984 0.05450255
## 23      1 1e+01 1.00 0.7729738 0.6816854 0.03993984 0.05450255
## 24      1 1e+01 2.00 0.7729738 0.6816854 0.03993984 0.05450255
## 25      1 1e+01 4.00 0.7729738 0.6816854 0.03993984 0.05450255
## 26      2 1e-03 0.25 0.6980446 0.5242853 0.01895395 0.03116196
## 27      2 1e-03 0.50 0.7582989 0.6360200 0.01940039 0.03159510
## 28      2 1e-03 1.00 0.8001093 0.7085987 0.02979500 0.04542089
## 29      2 1e-03 2.00 0.8079729 0.7238669 0.03957032 0.05865008
## 30      2 1e-03 4.00 0.8160018 0.7358928 0.03050783 0.04520573
## 31      2 1e-02 0.25 0.8187548 0.7381368 0.04408286 0.06568793
## 32      2 1e-02 0.50 0.8304962 0.7573352 0.03856143 0.05612795
## 33      2 1e-02 1.00 0.8304309 0.7585935 0.03684338 0.05270523
## 34      2 1e-02 2.00 0.8303809 0.7590561 0.03398619 0.04893502
## 35      2 1e-02 4.00 0.8303809 0.7590561 0.03398619 0.04893502
## 36      2 1e-01 0.25 0.8107246 0.7240673 0.03642956 0.05464839
## 37      2 1e-01 0.50 0.8107246 0.7240673 0.03642956 0.05464839
## 38      2 1e-01 1.00 0.8107246 0.7240673 0.03642956 0.05464839
## 39      2 1e-01 2.00 0.8107246 0.7240673 0.03642956 0.05464839
## 40      2 1e-01 4.00 0.8107246 0.7240673 0.03642956 0.05464839
## 41      2 1e+00 0.25 0.7795949 0.6724016 0.05541629 0.08694211
```

```
## 42      2 1e+00 0.50 0.7795949 0.6724016 0.05541629 0.08694211
## 43      2 1e+00 1.00 0.7795949 0.6724016 0.05541629 0.08694211
## 44      2 1e+00 2.00 0.7795949 0.6724016 0.05541629 0.08694211
## 45      2 1e+00 4.00 0.7795949 0.6724016 0.05541629 0.08694211
## 46      2 1e+01 0.25 0.7624514 0.6455198 0.04435738 0.06799037
## 47      2 1e+01 0.50 0.7624514 0.6455198 0.04435738 0.06799037
## 48      2 1e+01 1.00 0.7624514 0.6455198 0.04435738 0.06799037
## 49      2 1e+01 2.00 0.7624514 0.6455198 0.04435738 0.06799037
## 50      2 1e+01 4.00 0.7624514 0.6455198 0.04435738 0.06799037
## 51      3 1e-03 0.25 0.7387097 0.5998871 0.02468861 0.03900589
## 52      3 1e-03 0.50 0.7962623 0.7011804 0.03241501 0.04968543
## 53      3 1e-03 1.00 0.8145703 0.7321152 0.03767988 0.05662290
## 54      3 1e-03 2.00 0.8185352 0.7390221 0.03346213 0.04971688
## 55      3 1e-03 4.00 0.8266141 0.7515958 0.03557888 0.05253606
## 56      3 1e-02 0.25 0.8210805 0.7424405 0.03496529 0.05134200
## 57      3 1e-02 0.50 0.8183814 0.7396252 0.03160755 0.04641282
## 58      3 1e-02 1.00 0.8183139 0.7398224 0.03150744 0.04611784
## 59      3 1e-02 2.00 0.8183139 0.7398224 0.03150744 0.04611784
## 60      3 1e-02 4.00 0.8183139 0.7398224 0.03150744 0.04611784
## 61      3 1e-01 0.25 0.7622971 0.6400269 0.03498791 0.05194321
## 62      3 1e-01 0.50 0.7622971 0.6400269 0.03498791 0.05194321
## 63      3 1e-01 1.00 0.7622971 0.6400269 0.03498791 0.05194321
## 64      3 1e-01 2.00 0.7622971 0.6400269 0.03498791 0.05194321
## 65      3 1e-01 4.00 0.7622971 0.6400269 0.03498791 0.05194321
## 66      3 1e+00 0.25 0.7386934 0.5987277 0.03558783 0.05460592
## 67      3 1e+00 0.50 0.7386934 0.5987277 0.03558783 0.05460592
## 68      3 1e+00 1.00 0.7386934 0.5987277 0.03558783 0.05460592
## 69      3 1e+00 2.00 0.7386934 0.5987277 0.03558783 0.05460592
## 70      3 1e+00 4.00 0.7386934 0.5987277 0.03558783 0.05460592
## 71      3 1e+01 0.25 0.7360268 0.5940235 0.03592153 0.05474674
## 72      3 1e+01 0.50 0.7360268 0.5940235 0.03592153 0.05474674
## 73      3 1e+01 1.00 0.7360268 0.5940235 0.03592153 0.05474674
## 74      3 1e+01 2.00 0.7360268 0.5940235 0.03592153 0.05474674
## 75      3 1e+01 4.00 0.7360268 0.5940235 0.03592153 0.05474674
```

```
confusionMatrix(predict(svmpoly, newdata = test_data), test_labels)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Basal Her2 LumA LumB Normal
##     Basal     28    0    0    1      1
##     Her2       0   11    0    1      0
##     LumA       0    2   81    9      0
##     LumB       0    0    5   27      0
##     Normal     0    0    0    0     22
##
## Overall Statistics
##
##                Accuracy : 0.8989
##                  95% CI : (0.8467, 0.938)
##     No Information Rate : 0.4574
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8555
```

```
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: Basal Class: Her2 Class: LumA Class: LumB
## Sensitivity                1.0000     0.84615      0.9419      0.7105
## Specificity                0.9875     0.99429      0.8922      0.9667
## Pos Pred Value             0.9333     0.91667      0.8804      0.8438
## Neg Pred Value             1.0000     0.98864      0.9479      0.9295
## Prevalence                 0.1489     0.06915      0.4574      0.2021
## Detection Rate             0.1489     0.05851      0.4309      0.1436
## Detection Prevalence       0.1596     0.06383      0.4894      0.1702
## Balanced Accuracy          0.9938     0.92022      0.9170      0.8386
##                      Class: Normal
## Sensitivity                 0.9565
## Specificity                 1.0000
## Pos Pred Value              1.0000
## Neg Pred Value              0.9940
## Prevalence                  0.1223
## Detection Rate              0.1170
## Detection Prevalence        0.1170
## Balanced Accuracy           0.9783
```

Performance (polynomial)

```
# Saving performance metrics
models <- data.frame("Model" = "SVM Polynomial", "Accuracy" = 0.8989,
                     "Kappa" = 0.8555) %>% rbind(models)
```

```
set.seed(1234)
svmrad <- train(x = train_data, y = train_labels,
method = 'svmRadial', tuneLength = 5,
trControl = tc1)
```

```
svmrad$results
```

```
##         sigma    C  Accuracy     Kappa AccuracySD    KappaSD
## 1 0.003376159 0.25 0.6928485 0.5177307 0.02283392 0.03647516
## 2 0.003376159 0.50 0.7751903 0.6671549 0.02621298 0.04035905
## 3 0.003376159 1.00 0.8146032 0.7315784 0.03005320 0.04538014
## 4 0.003376159 2.00 0.8227198 0.7453955 0.04711652 0.06992023
## 5 0.003376159 4.00 0.8305461 0.7585369 0.04107465 0.05954033
```

```
confusionMatrix(predict(svmrad, newdata = test_data), test_labels)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Basal Her2 LumA LumB Normal
##     Basal     27    0    0    1      2
##     Her2       1   11    0    1      0
##     LumA       0    2   81    9      0
##     LumB       0    0    5   27      0
##     Normal     0    0    0    0     21
##
```

```
## Overall Statistics
##
##                Accuracy : 0.8883
##                  95% CI : (0.8343, 0.9295)
##     No Information Rate : 0.4574
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8403
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: Basal Class: Her2 Class: LumA Class: LumB
## Sensitivity                0.9643     0.84615      0.9419      0.7105
## Specificity                0.9812     0.98857      0.8922      0.9667
## Pos Pred Value             0.9000     0.84615      0.8804      0.8438
## Neg Pred Value             0.9937     0.98857      0.9479      0.9295
## Prevalence                 0.1489     0.06915      0.4574      0.2021
## Detection Rate             0.1436     0.05851      0.4309      0.1436
## Detection Prevalence       0.1596     0.06915      0.4894      0.1702
## Balanced Accuracy          0.9728     0.91736      0.9170      0.8386
##                      Class: Normal
## Sensitivity                 0.9130
## Specificity                 1.0000
## Pos Pred Value              1.0000
## Neg Pred Value              0.9880
## Prevalence                  0.1223
## Detection Rate              0.1117
## Detection Prevalence        0.1117
## Balanced Accuracy           0.9565
```

Performance (radial)

```
# Saving performance metrics
models <- data.frame("Model" = "SVM Radial", "Accuracy" = 0.8883,
                     "Kappa" = 0.8403) %>% rbind(models)
```

2.4. Logistic Regression

```
logit <- train(x = train_data, y = train_labels, method = "multinom",
                   trControl = tc1)
```

```
logit$results
```

```
##   decay  Accuracy     Kappa AccuracySD    KappaSD
## 1 0e+00 0.6875632 0.5643221 0.05125347 0.07339902
## 2 1e-04 0.6967094 0.5756933 0.05158031 0.07464494
## 3 1e-01 0.7780154 0.6857245 0.03782924 0.05486238
```

```
confusionMatrix(predict(logit, newdata = test_data), test_labels)
```

```
## Confusion Matrix and Statistics
##
##           Reference
```

```
## Prediction Basal Her2 LumA LumB Normal
##     Basal     26    1    0    1      2
##     Her2       0   11    0    3      0
##     LumA       0    1   67   12      1
##     LumB       0    0   16   22      1
##     Normal     2    0    3    0     19
##
## Overall Statistics
##
##                Accuracy : 0.7713
##                  95% CI : (0.7045, 0.8293)
##     No Information Rate : 0.4574
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6808
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: Basal Class: Her2 Class: LumA Class: LumB
## Sensitivity                0.9286     0.84615      0.7791      0.5789
## Specificity                0.9750     0.98286      0.8627      0.8867
## Pos Pred Value             0.8667     0.78571      0.8272      0.5641
## Neg Pred Value             0.9873     0.98851      0.8224      0.8926
## Prevalence                 0.1489     0.06915      0.4574      0.2021
## Detection Rate             0.1383     0.05851      0.3564      0.1170
## Detection Prevalence       0.1596     0.07447      0.4309      0.2074
## Balanced Accuracy          0.9518     0.91451      0.8209      0.7328
##                      Class: Normal
## Sensitivity                 0.8261
## Specificity                 0.9697
## Pos Pred Value              0.7917
## Neg Pred Value              0.9756
## Prevalence                  0.1223
## Detection Rate              0.1011
## Detection Prevalence        0.1277
## Balanced Accuracy           0.8979
```

Performance

```
# Saving performance metrics
models <- data.frame("Model" = "Logistic Regression", "Accuracy" = 0.7713,
                     "Kappa" = 0.6808) %>% rbind(models)
```

2.5. Decision Trees

```
set.seed(1234)
cart_pre <- rpart(response ~., data = rpart.df, method = "class",
                  parms = list(split = "gini"))

tg_ctree <- data.frame(cp = cart_pre$cptable[,1])
```

```r
cart <- train(x = train_data, y = train_labels, method = "rpart",
              parms = list(split = "gini"),
              tuneGrid = tg_ctree,
              trControl = trainControl(method = "cv", number = 10,
                                       selectionFunction = "oneSE"))
```

```r
cart$results
```

```
##           cp  Accuracy      Kappa AccuracySD    KappaSD
## 1 0.01000000 0.6379870 0.47811523 0.03853149 0.05110529
## 2 0.01346154 0.6339542 0.46998727 0.02661412 0.03773223
## 3 0.01538462 0.6326726 0.46482158 0.01678283 0.02740563
## 4 0.01923077 0.6313739 0.46510530 0.01803768 0.03191707
## 5 0.02500000 0.6261449 0.45349992 0.02454186 0.03608125
## 6 0.02692308 0.6261449 0.45286916 0.02454186 0.03761483
## 7 0.05000000 0.6327068 0.42419494 0.02860342 0.04368920
## 8 0.12692308 0.6327068 0.41726780 0.02276645 0.04247821
## 9 0.21923077 0.4626965 0.02696477 0.02977519 0.08527009
```

```r
confusionMatrix(predict(cart, newdata = test_data), test_labels)
```

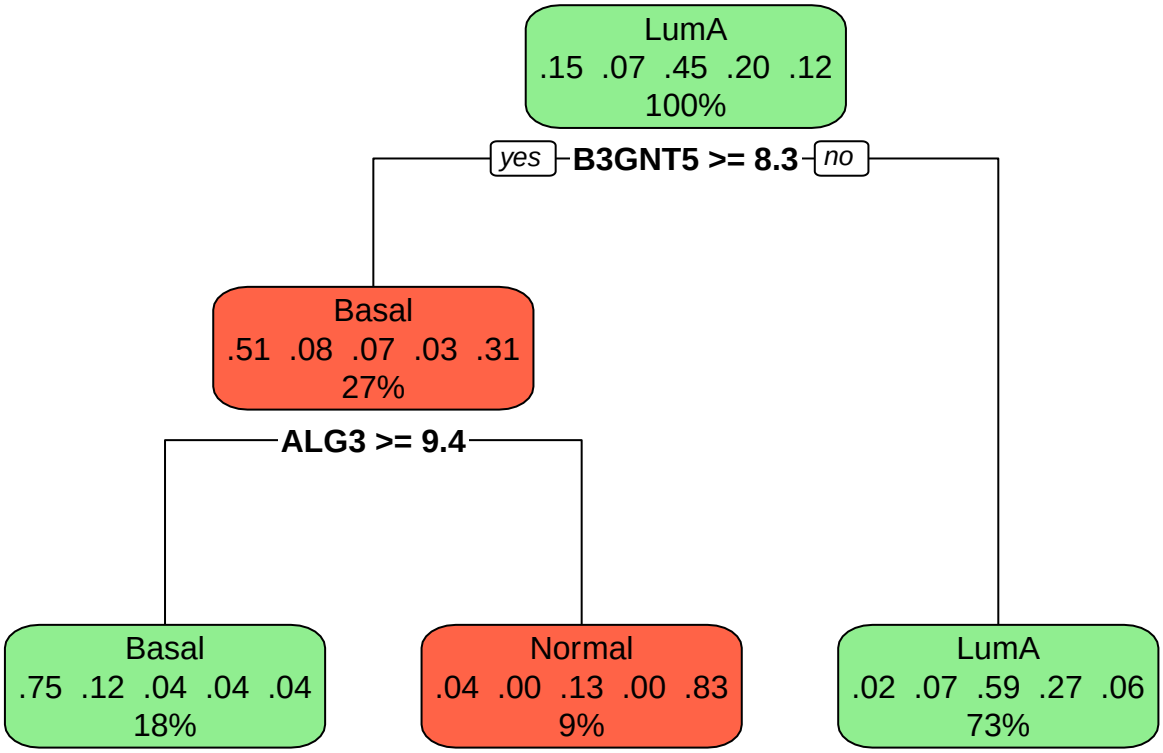
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Basal Her2 LumA LumB Normal
##     Basal     27    4    2    1      5
##     Her2       0    0    0    0      0
##     LumA       0    9   82   37      6
##     LumB       0    0    0    0      0
##     Normal     1    0    2    0     12
##
## Overall Statistics
##
##                Accuracy : 0.6436
##                  95% CI : (0.5707, 0.712)
##     No Information Rate : 0.4574
##     P-Value [Acc > NIR] : 2.151e-07
##
##                   Kappa : 0.4373
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: Basal Class: Her2 Class: LumA Class: LumB
## Sensitivity                0.9643     0.00000      0.9535      0.0000
## Specificity                0.9250     1.00000      0.4902      1.0000
## Pos Pred Value             0.6923         NaN      0.6119         NaN
## Neg Pred Value             0.9933     0.93085      0.9259      0.7979
## Prevalence                 0.1489     0.06915      0.4574      0.2021
## Detection Rate             0.1436     0.00000      0.4362      0.0000
## Detection Prevalence       0.2074     0.00000      0.7128      0.0000
## Balanced Accuracy          0.9446     0.50000      0.7218      0.5000
##                      Class: Normal
## Sensitivity                0.52174
```

```
## Specificity                 0.98182
## Pos Pred Value              0.80000
## Neg Pred Value              0.93642
## Prevalence                  0.12234
## Detection Rate              0.06383
## Detection Prevalence        0.07979
## Balanced Accuracy           0.75178
```

Performance

```
# Saving performance metrics
models <- data.frame("Model" = "Decision Tree", "Accuracy" = 0.6436,
                     "Kappa" = 0.4373) %>% rbind(models)
```

```
rpart.plot(cart$finalModel, box.col = c("lightgreen", "tomato"))
```



2.6. XG Boost

```
set.seed(1234)
xgboost <- train(x = train_data, y = train_labels, method = "xgbTree", trControl = tc1)
```

```
xgboost$results %>% head(5)
```

```
##    eta max_depth gamma colsample_bytree min_child_weight subsample nrounds
## 1  0.3         1     0              0.6                1      0.50      50
## 4  0.3         1     0              0.6                1      0.75      50
## 7  0.3         1     0              0.6                1      1.00      50
## 10 0.3         1     0              0.8                1      0.50      50
## 13 0.3         1     0              0.8                1      0.75      50
##     Accuracy     Kappa AccuracySD    KappaSD
```

```
## 1  0.7791363 0.6800307 0.03342865 0.04830418
## 4  0.7753047 0.6753299 0.03952979 0.05955746
## 7  0.7544181 0.6430033 0.03077115 0.04509411
## 10 0.7832529 0.6855707 0.04673736 0.06863328
## 13 0.7663962 0.6618744 0.03906508 0.05961533
```

```
confusionMatrix(predict(xgboost, newdata = test_data), test_labels)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Basal Her2 LumA LumB Normal
##     Basal     28    2    0    1      2
##     Her2       0    9    0    1      0
##     LumA       0    2   79   11      2
##     LumB       0    0    7   25      0
##     Normal     0    0    0    0     19
##
## Overall Statistics
##
##                Accuracy : 0.8511
##                  95% CI : (0.792, 0.8987)
##     No Information Rate : 0.4574
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7856
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: Basal Class: Her2 Class: LumA Class: LumB
## Sensitivity                1.0000     0.69231      0.9186      0.6579
## Specificity                0.9688     0.99429      0.8529      0.9533
## Pos Pred Value             0.8485     0.90000      0.8404      0.7813
## Neg Pred Value             1.0000     0.97753      0.9255      0.9167
## Prevalence                 0.1489     0.06915      0.4574      0.2021
## Detection Rate             0.1489     0.04787      0.4202      0.1330
## Detection Prevalence       0.1755     0.05319      0.5000      0.1702
## Balanced Accuracy          0.9844     0.84330      0.8858      0.8056
##                      Class: Normal
## Sensitivity                 0.8261
## Specificity                 1.0000
## Pos Pred Value              1.0000
## Neg Pred Value              0.9763
## Prevalence                  0.1223
## Detection Rate              0.1011
## Detection Prevalence        0.1011
## Balanced Accuracy           0.9130
```

Performance

```
# Saving performance metrics
models <- data.frame("Model" = "XGBoost", "Accuracy" = 0.8511,
```

```
                      "Kappa" = 0.7856) %>% rbind(models)
```

2.7. k-Nearest Neighbors

```
knn_grid <- expand.grid(k = seq(1, 25, by = 2))

set.seed(1234)
knn <- train(x = train_data, y = train_labels, method = "knn",
             tuneGrid = knn_grid,
             trControl = tc1)
```

```
knn$results
```

```
##     k  Accuracy     Kappa AccuracySD    KappaSD
## 1   1 0.7293268 0.6024519 0.03280859 0.04820873
## 2   3 0.7333939 0.5983047 0.03355134 0.05419240
## 3   5 0.7452913 0.6135623 0.02806324 0.04902524
## 4   7 0.7322285 0.5913564 0.02486006 0.04239812
## 5   9 0.7267641 0.5805845 0.02713220 0.04722392
## 6  11 0.7175697 0.5664571 0.02846551 0.04905609
## 7  13 0.7032064 0.5447956 0.02255138 0.03415944
## 8  15 0.7084529 0.5532670 0.02906339 0.04382777
## 9  17 0.7071893 0.5490360 0.02344464 0.03586281
## 10 19 0.7165375 0.5637805 0.02121383 0.03713030
## 11 21 0.7020108 0.5398505 0.01851419 0.02858775
## 12 23 0.7033770 0.5397979 0.01274993 0.02302743
## 13 25 0.7032568 0.5398031 0.01679704 0.02454582
```

```
confusionMatrix(predict(knn, newdata = test_data), test_labels)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Basal Her2 LumA LumB Normal
##     Basal     25    0    0    0      1
##     Her2       0    6    0    1      0
##     LumA       2    7   86   25      1
##     LumB       1    0    0   12      0
##     Normal     0    0    0    0     21
##
## Overall Statistics
##
##                Accuracy : 0.7979
##                  95% CI : (0.7333, 0.8528)
##     No Information Rate : 0.4574
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6913
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: Basal Class: Her2 Class: LumA Class: LumB
```

```
## Sensitivity                0.8929     0.46154      1.0000     0.31579
## Specificity                0.9938     0.99429      0.6569     0.99333
## Pos Pred Value             0.9615     0.85714      0.7107     0.92308
## Neg Pred Value             0.9815     0.96133      1.0000     0.85143
## Prevalence                 0.1489     0.06915      0.4574     0.20213
## Detection Rate             0.1330     0.03191      0.4574     0.06383
## Detection Prevalence       0.1383     0.03723      0.6436     0.06915
## Balanced Accuracy          0.9433     0.72791      0.8284     0.65456
##                       Class: Normal
## Sensitivity                  0.9130
## Specificity                  1.0000
## Pos Pred Value               1.0000
## Neg Pred Value               0.9880
## Prevalence                   0.1223
## Detection Rate               0.1117
## Detection Prevalence         0.1117
## Balanced Accuracy            0.9565
```

Performance

```
# Saving performance metrics
models <- data.frame("Model" = "k-Nearest Neighbors", "Accuracy" = 0.7979,
                     "Kappa" = 0.6913) %>% rbind(models)
```

2.11. Gradient Boosted Machines (GBM)

```
set.seed(1234)
bct <- train(x = train_data, y = train_labels, method = "gbm",
             bag.fraction = 0.5,
             tuneLength = 5,
             trControl = tc1)
```

```
bct$results %>% head(5)
```

```
##    shrinkage interaction.depth n.minobsinnode n.trees  Accuracy     Kappa
## 1        0.1                 1             10      50 0.7506548 0.6364129
## 6        0.1                 2             10      50 0.7625001 0.6536494
## 11       0.1                 3             10      50 0.7752218 0.6744070
## 16       0.1                 4             10      50 0.7608608 0.6513009
## 21       0.1                 5             10      50 0.7608476 0.6513624
##    AccuracySD    KappaSD
## 1  0.03643424 0.05451702
## 6  0.04247796 0.06097365
## 11 0.03618091 0.05290703
## 16 0.04265970 0.06331586
## 21 0.04239086 0.06105353
```

```
confusionMatrix(predict(bct, newdata = test_data), test_labels)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Basal Her2 LumA LumB Normal
##     Basal     27    1    0    0      2
##     Her2       0    8    0    0      0
```

```
##     LumA       0    4   80   12      1
##     LumB       0    0    6   26      0
##     Normal     1    0    0    0     20
##
## Overall Statistics
##
##                Accuracy : 0.8564
##                  95% CI : (0.798, 0.9032)
##     No Information Rate : 0.4574
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7916
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: Basal Class: Her2 Class: LumA Class: LumB
## Sensitivity                0.9643     0.61538      0.9302      0.6842
## Specificity                0.9812     1.00000      0.8333      0.9600
## Pos Pred Value             0.9000     1.00000      0.8247      0.8125
## Neg Pred Value             0.9937     0.97222      0.9341      0.9231
## Prevalence                 0.1489     0.06915      0.4574      0.2021
## Detection Rate             0.1436     0.04255      0.4255      0.1383
## Detection Prevalence       0.1596     0.04255      0.5160      0.1702
## Balanced Accuracy          0.9728     0.80769      0.8818      0.8221
##                      Class: Normal
## Sensitivity                 0.8696
## Specificity                 0.9939
## Pos Pred Value              0.9524
## Neg Pred Value              0.9820
## Prevalence                  0.1223
## Detection Rate              0.1064
## Detection Prevalence        0.1117
## Balanced Accuracy           0.9318
```

Performance

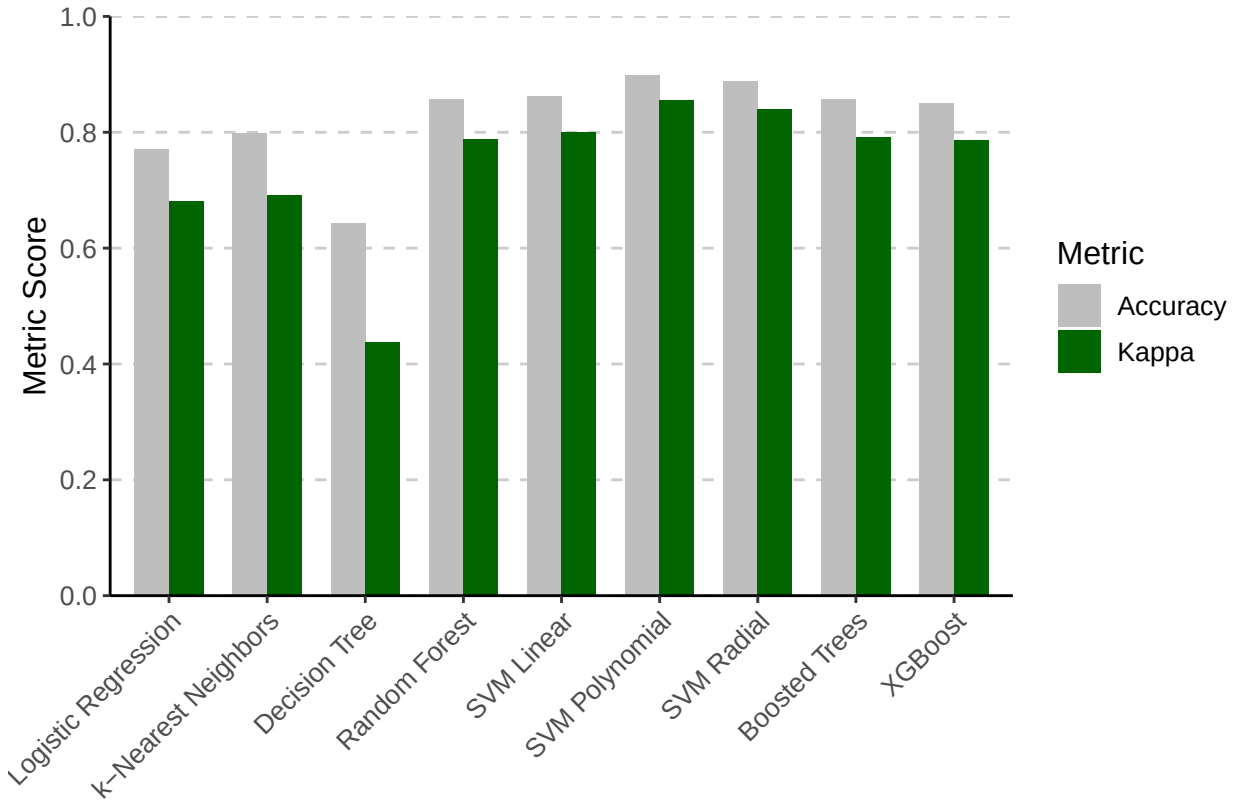
```
# Saving performance metrics
models <- data.frame("Model" = "Boosted Trees", "Accuracy" = 0.8564,
                     "Kappa" = 0.7916) %>% rbind(models)
```

3. Model Outcome Analysis

Performance barchart

```
# Reshape data for easier plotting
models_long <- models %>%
  pivot_longer(cols = c(Accuracy, Kappa), names_to = "Metric", values_to = "Value")

# Order models for plotting
models_long$Model <- factor(models_long$Model, levels = c(
  "Logistic Regression", "k-Nearest Neighbors", "Decision Tree", "Random Forest",
```

```
  "SVM Linear", "SVM Polynomial", "SVM Radial", "Boosted Trees", "XGBoost"
))

# Plot
ggplot(models_long, aes(x = Model, y = Value, fill = Metric)) +
  geom_bar(stat = "identity", position = "dodge", width = 0.7) +
  scale_fill_manual(values = c("Accuracy" = "grey", "Kappa" = "darkgreen")) +
  labs(title = "",
       y = "Metric Score", fill = "Metric") +
  theme_classic() +
  theme(
    axis.text.x = element_text(angle = 45, hjust = 1, size = 10),
    axis.text.y = element_text(size = 10),
    axis.title.x = element_blank(),
    axis.title.y = element_text(size = 12),
    legend.title = element_text(size = 12),
    legend.text = element_text(size = 10),
    panel.grid.major.y = element_line(color = "grey80", linetype = "dashed")
  ) +
   scale_y_continuous(expand = c(0, 0), breaks = seq(0, 1, by = 0.2),
                      limits = c(0, 1))
```



Feature Importance and Family Analysis

```
# Example: Genes from RF feature importance
var_importance <- as.data.frame(var_importance)
var_importance_sorted <- var_importance[order(var_importance$MeanDecreaseGini,
```

```
                                              decreasing = TRUE), ]

rf_genes <- rownames(var_importance_sorted)[1:30]


# Filter metadata for RF genes
mapped_genes <- gene_metadata %>%
  filter(gene %in% rf_genes)

# View the mapped genes
print(mapped_genes)
```

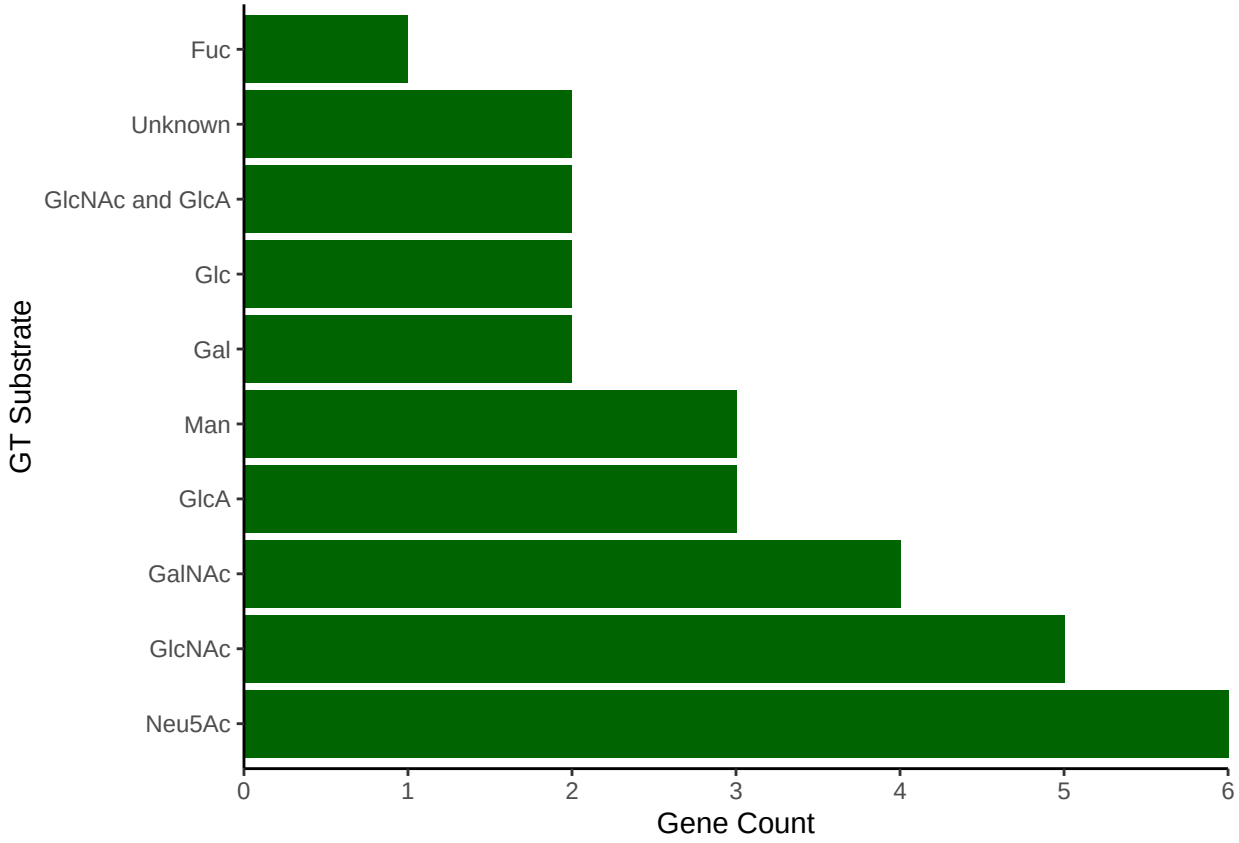
```
##          gene family clan fold       substrate
## 1     UGT2B11    GT1    I GT-B            GlcA
## 2     UGT2B15    GT1    I GT-B            GlcA
## 3      UGT2B7    GT1    I GT-B            GlcA
## 4        UGT8    GT1    I GT-B             Gal
## 5       GCNT4   GT14 <NA> <NA>          GlcNAc
## 6       MGAT3   GT17 <NA> <NA>          GlcNAc
## 7        DPM1    GT2    I GT-A             Man
## 8        HAS3    GT2    I GT-A GlcNAc and GlcA
## 9        UGCG   GT21 <NA> <NA>             Glc
## 10       FUT8   GT23 <NA> <NA>             Fuc
## 11    GALNT10   GT27 <NA> <NA>          GalNAc
## 12     GALNT6   GT27 <NA> <NA>          GalNAc
## 13     GALNT7   GT27 <NA> <NA>          GalNAc
## 14    ST3GAL3   GT29 <NA> <NA>          Neu5Ac
## 15    ST3GAL6   GT29 <NA> <NA>          Neu5Ac
## 16 ST6GALNAC1   GT29 <NA> <NA>          Neu5Ac
## 17 ST6GALNAC6   GT29 <NA> <NA>          Neu5Ac
## 18    ST8SIA1   GT29 <NA> <NA>          Neu5Ac
## 19    ST8SIA6   GT29 <NA> <NA>          Neu5Ac
## 20   B3GALNT2   GT31 <NA> <NA>          GalNAc
## 21    B3GALT1   GT31 <NA> <NA>             Gal
## 22     B3GNT4   GT31 <NA> <NA>          GlcNAc
## 23     B3GNT5   GT31 <NA> <NA>          GlcNAc
## 24       LFNG   GT31 <NA> <NA>          GlcNAc
## 25       ALG1   GT33 <NA> <NA>             Man
## 26       PYGM   GT35   IV GT-B             Glc
## 27      EXTL3   GT47 <NA> <NA> GlcNAc and GlcA
## 28       ALG3   GT58 <NA> <NA>             Man
## 29     GLT8D1    GT8  III GT-A         Unknown
## 30     GLT8D2    GT8  III GT-A         Unknown
```

```
# Count genes per substrate
substrate_counts <- mapped_genes %>%
  count(substrate, name = "gene_count") %>%
  arrange(desc(gene_count))

# View counts per family
print(substrate_counts)
```

```
##          substrate gene_count
## 1           Neu5Ac          6
```

```
## 2           GlcNAc          5
## 3           GalNAc          4
## 4             GlcA          3
## 5              Man          3
## 6              Gal          2
## 7              Glc          2
## 8  GlcNAc and GlcA          2
## 9          Unknown          2
## 10             Fuc          1
```

```r
# Bar plot of family counts
ggplot(substrate_counts, aes(x = reorder(substrate, -gene_count),
                             y = gene_count)) +
  geom_bar(stat = "identity", fill = "darkgreen") +
  theme_classic() +
  labs(x = "GT Substrate", y = "Gene Count") +
  coord_flip() +
  scale_y_continuous(expand = c(0, 0))
```



Clan Enrichment Analysis

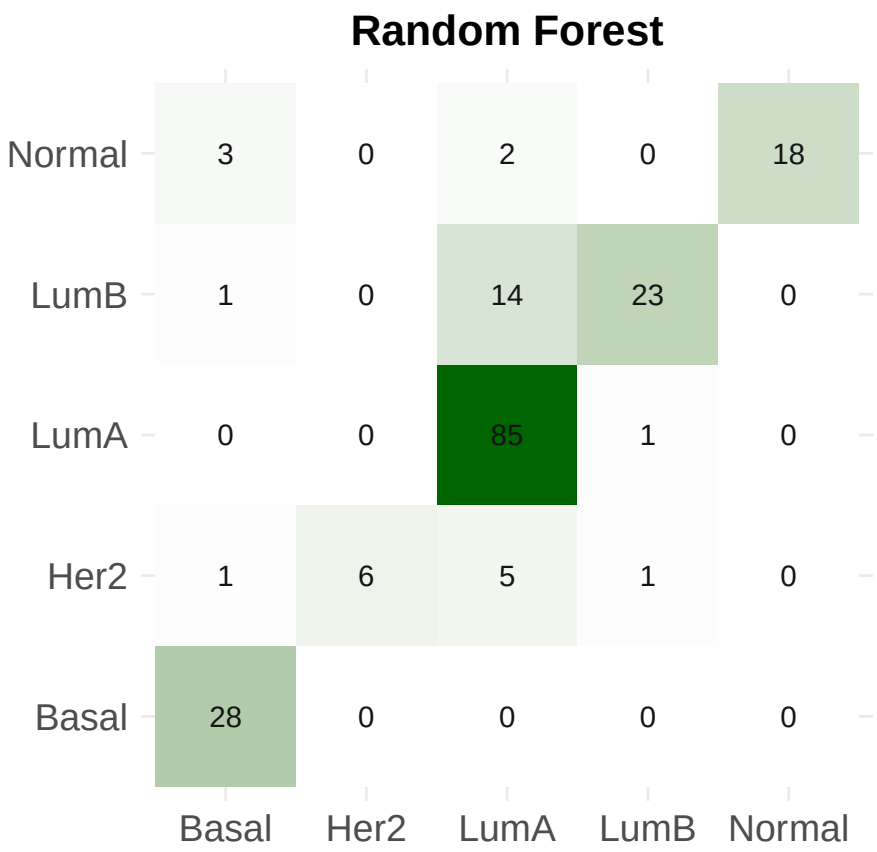
```r
# Load necessary libraries
library(dplyr)
library(ggplot2)

# Define the top 50 significant genes
top_genes <- c("UGT8", "B3GNT5", "ST8SIA1", "HAS3", "ALG3", "FUT8", "UGCG", "ALG1",
```

```
                "MGAT3", "LFNG", "GALNT10", "GALNT7", "ST3GAL6", "PYGM", "B3GALT1",
                "B3GNT4", "ST6GALNAC6", "ST8SIA6", "DPM1", "GALNT6", "GLT8D2",
                "EXTL3", "GLT8D1", "ST3GAL3", "GCNT4", "ST6GALNAC1", "UGT2B11",
                "UGT2B7", "UGT2B15", "B3GALNT2", "C1GALT1", "GYG2", "B3GNT3",
                "B4GALT6", "GALNT4", "CSGALNACT1", "B4GALT3", "B3GNT7", "FUT3",
                "XYLT2", "ST6GALNAC3", "B3GAT1", "PIGV", "EXT1", "FUT4", "CHSY1",
                "B4GALT2", "DPY19L2", "ABO", "B3GNT8")

# Filter metadata for the top genes
enriched_clans <- gene_metadata %>%
  filter(gene %in% top_genes) %>%
  group_by(fold) %>%
  summarise(count = n()) %>%
  arrange(desc(count))

# Plot the clan enrichment
ggplot(enriched_clans, aes(x = reorder(fold, -count), y = count, fill = fold)) +
  geom_bar(stat = "identity") +
  labs(
    title = "Fold Enrichment in Top 50 Glycoenzymes",
    x = "Fold",
    y = "Count of Genes"
  ) +
  theme_minimal() +
  theme(legend.position = "none")
```

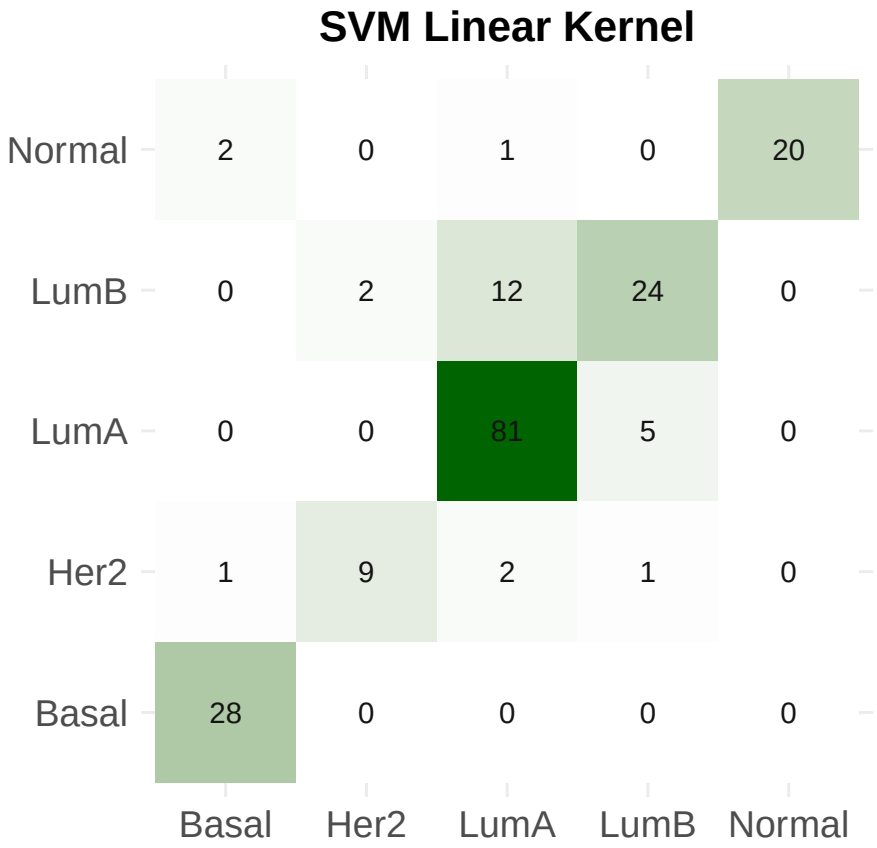
4. Confusion Matrices Visualization

```
# Random Forest
# The confusion matrix from a single assessment set (i.e. fold)
cm_rf <- table(predict(rf, test_data), test_labels) %>% as.data.frame

ggplot(cm_rf, aes(x = Var1, y = test_labels, fill = Freq)) +
  geom_tile() +
  theme_minimal() +
  coord_equal() +
  scale_fill_gradient(low = "white", high = "darkgreen") +
  labs(title = "Random Forest", x = NULL, y = NULL) +
  theme(
    plot.title = element_text(hjust = 0.5, size = 16, face = "bold"),
    axis.text = element_text(size = 14),
    axis.title = element_blank()
  ) +
  guides(fill = FALSE) +
  geom_text(aes(label = Freq), color = "grey8", size = 4)
```

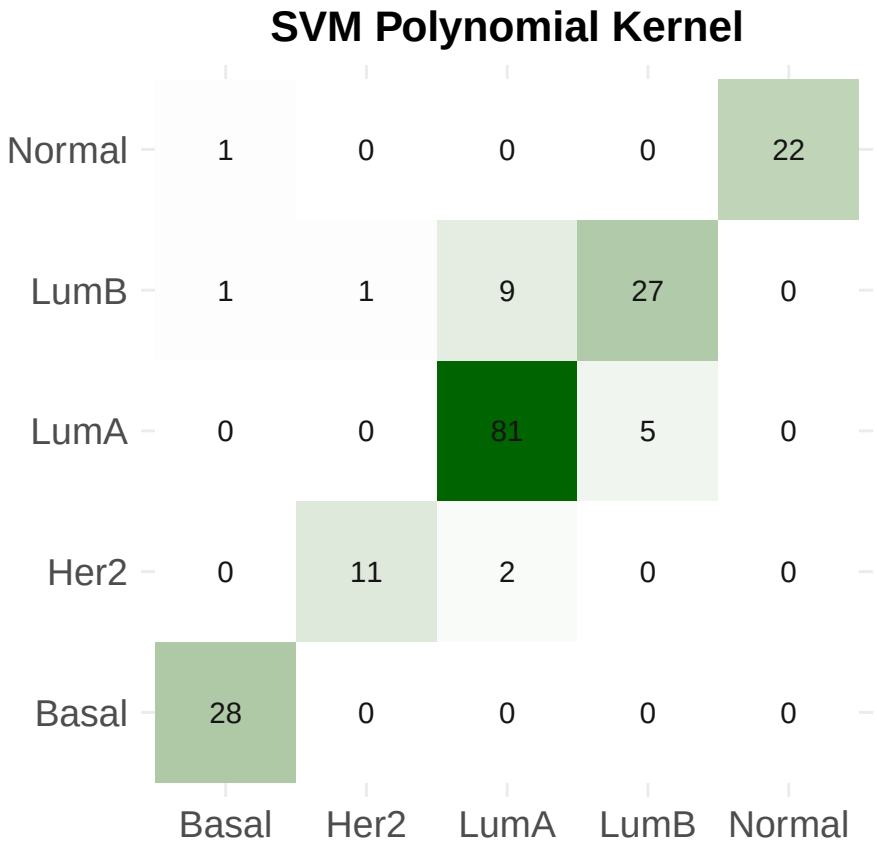
```
## Warning: The `<scale>` argument of `guides()` cannot be `FALSE`. Use "none" instead as
## of ggplot2 3.3.4.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



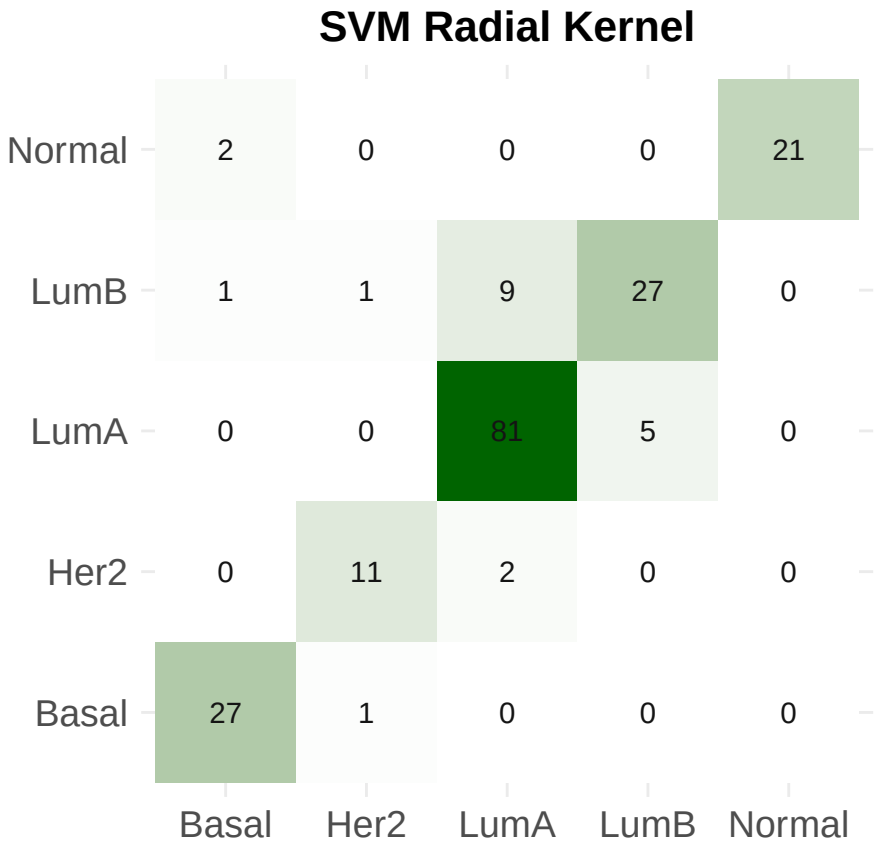

```
# SVM linear
# The confusion matrix from a single assessment set (i.e. fold)
cm_svmlin <- table(predict(svmlin, test_data), test_labels) %>% as.data.frame

ggplot(cm_svmlin, aes(x = Var1, y = test_labels, fill = Freq)) +
  geom_tile() +
  theme_minimal() +
  coord_equal() +
  scale_fill_gradient(low = "white", high = "darkgreen") +
  labs(title = "SVM Linear Kernel", x = NULL, y = NULL) +
  theme(
    plot.title = element_text(hjust = 0.5, size = 16, face = "bold"),
    axis.text = element_text(size = 14),
    axis.title = element_blank()
  ) +
  guides(fill = FALSE) +
  geom_text(aes(label = Freq), color = "grey8", size = 4)
```



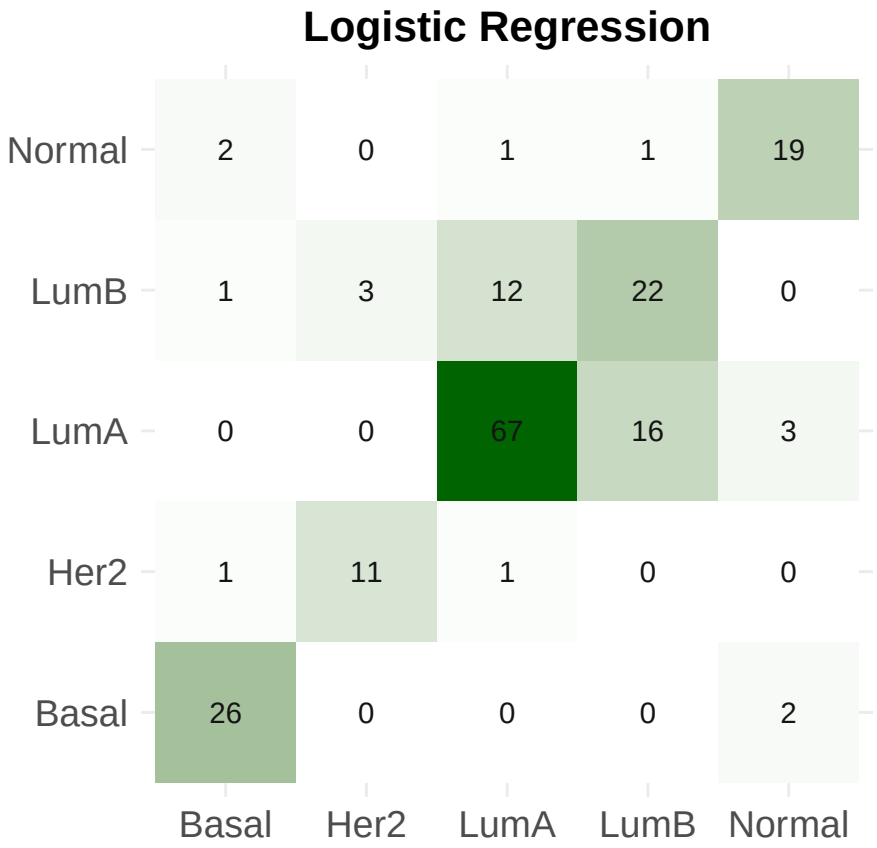

```
# SVM poly
# The confusion matrix from a single assessment set (i.e. fold)
cm_svmpoly <- table(predict(svmpoly, test_data), test_labels) %>% as.data.frame

ggplot(cm_svmpoly, aes(x = Var1, y = test_labels, fill = Freq)) +
  geom_tile() +
  theme_minimal() +
  coord_equal() +
  scale_fill_gradient(low = "white", high = "darkgreen") +
  labs(title = "SVM Polynomial Kernel", x = NULL, y = NULL) +
  theme(
    plot.title = element_text(hjust = 0.5, size = 16, face = "bold"),
    axis.text = element_text(size = 14),
    axis.title = element_blank()
  ) +
  guides(fill = FALSE) +
  geom_text(aes(label = Freq), color = "grey8", size = 4)
```



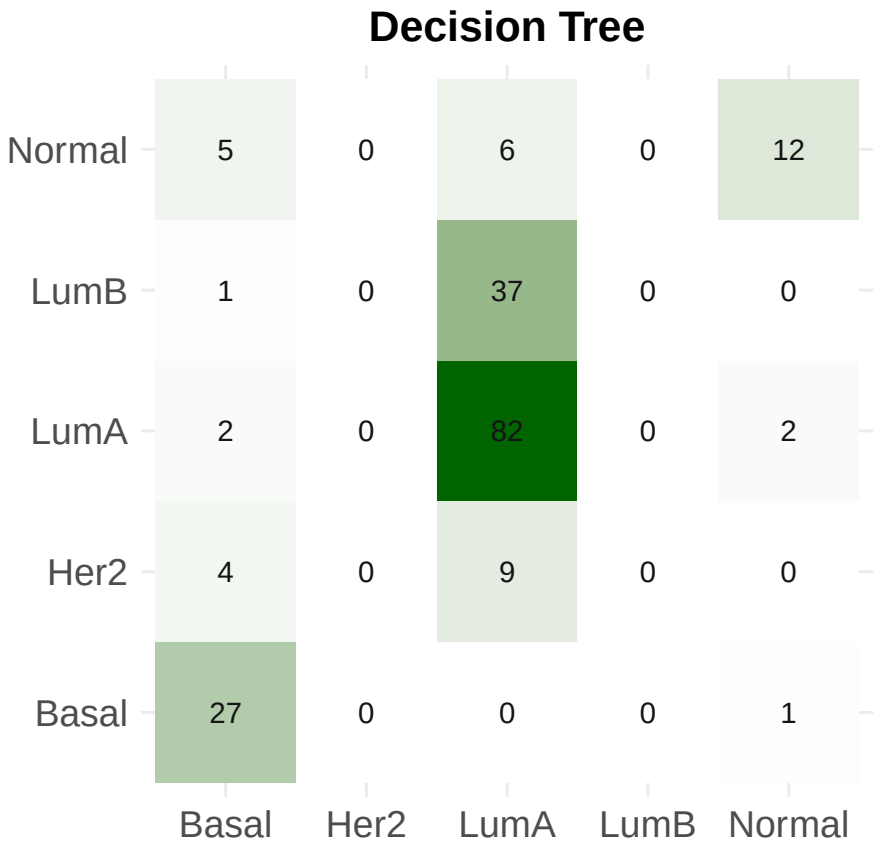

```r
# SVM radial
# The confusion matrix from a single assessment set (i.e. fold)
cm_svmrad <- table(predict(svmrad, test_data), test_labels) %>% as.data.frame

ggplot(cm_svmrad, aes(x = Var1, y = test_labels, fill = Freq)) +
  geom_tile() +
  theme_minimal() +
  coord_equal() +
  scale_fill_gradient(low = "white", high = "darkgreen") +
  labs(title = "SVM Radial Kernel", x = NULL, y = NULL) +
  theme(
    plot.title = element_text(hjust = 0.5, size = 16, face = "bold"),
    axis.text = element_text(size = 14),
    axis.title = element_blank()
  ) +
  guides(fill = FALSE) +
  geom_text(aes(label = Freq), color = "grey8", size = 4)
```



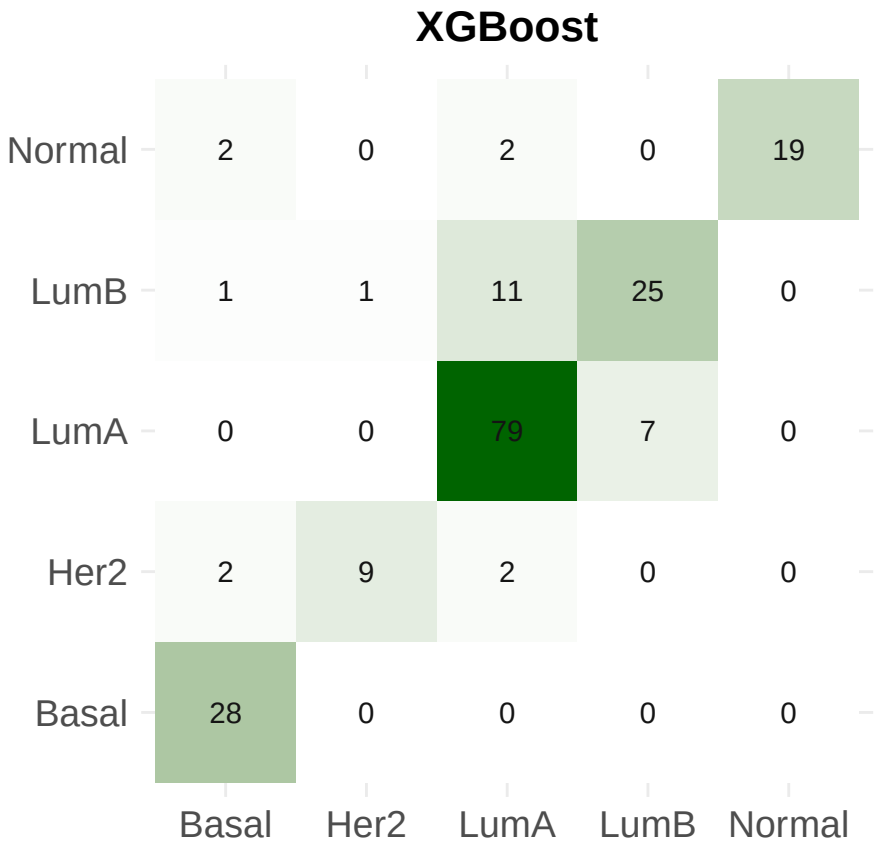

```
# Logistic Regression
# The confusion matrix from a single assessment set (i.e. fold)
cm_logit <- table(predict(logit, test_data), test_labels) %>% as.data.frame

ggplot(cm_logit, aes(x = Var1, y = test_labels, fill = Freq)) +
  geom_tile() +
  theme_minimal() +
  coord_equal() +
  scale_fill_gradient(low = "white", high = "darkgreen") +
  labs(title = "Logistic Regression", x = NULL, y = NULL) +
  theme(
    plot.title = element_text(hjust = 0.5, size = 16, face = "bold"),
    axis.text = element_text(size = 14),
    axis.title = element_blank()
  ) +
  guides(fill = FALSE) +
  geom_text(aes(label = Freq), color = "grey8", size = 4)
```



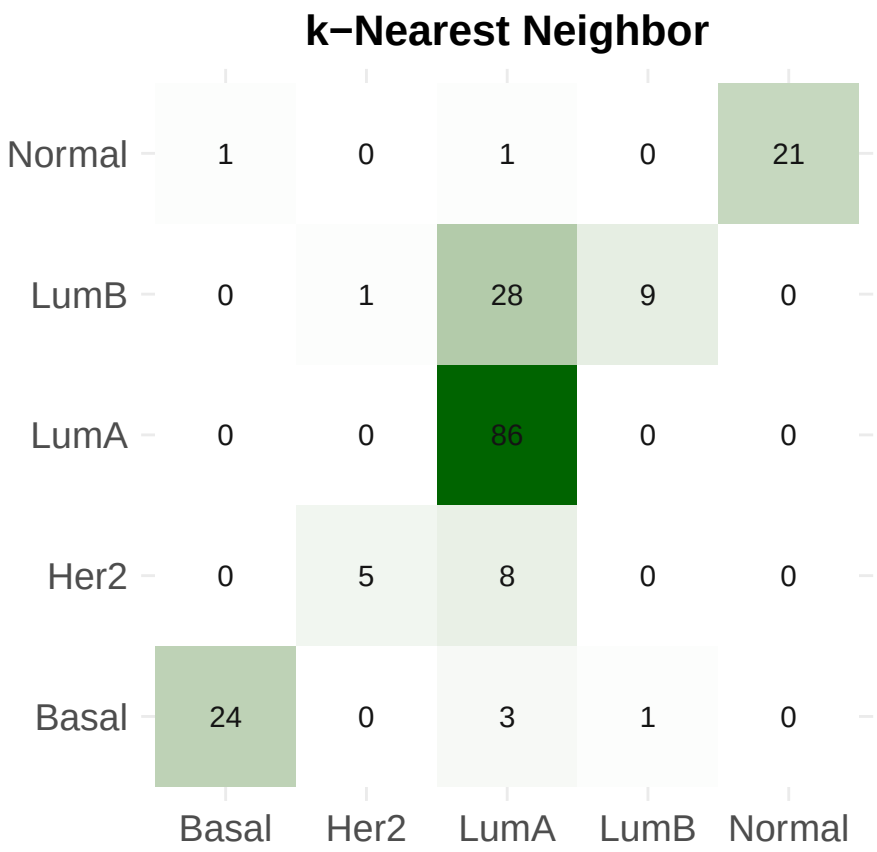

```
# Decision Tree
# The confusion matrix from a single assessment set (i.e. fold)
cm_cart <- table(predict(cart, test_data), test_labels) %>% as.data.frame

ggplot(cm_cart, aes(x = Var1, y = test_labels, fill = Freq)) +
  geom_tile() +
  theme_minimal() +
  coord_equal() +
  scale_fill_gradient(low = "white", high = "darkgreen") +
  labs(title = "Decision Tree", x = NULL, y = NULL) +
  theme(
    plot.title = element_text(hjust = 0.5, size = 16, face = "bold"),
    axis.text = element_text(size = 14),
    axis.title = element_blank()
  ) +
  guides(fill = FALSE) +
  geom_text(aes(label = Freq), color = "grey8", size = 4)
```



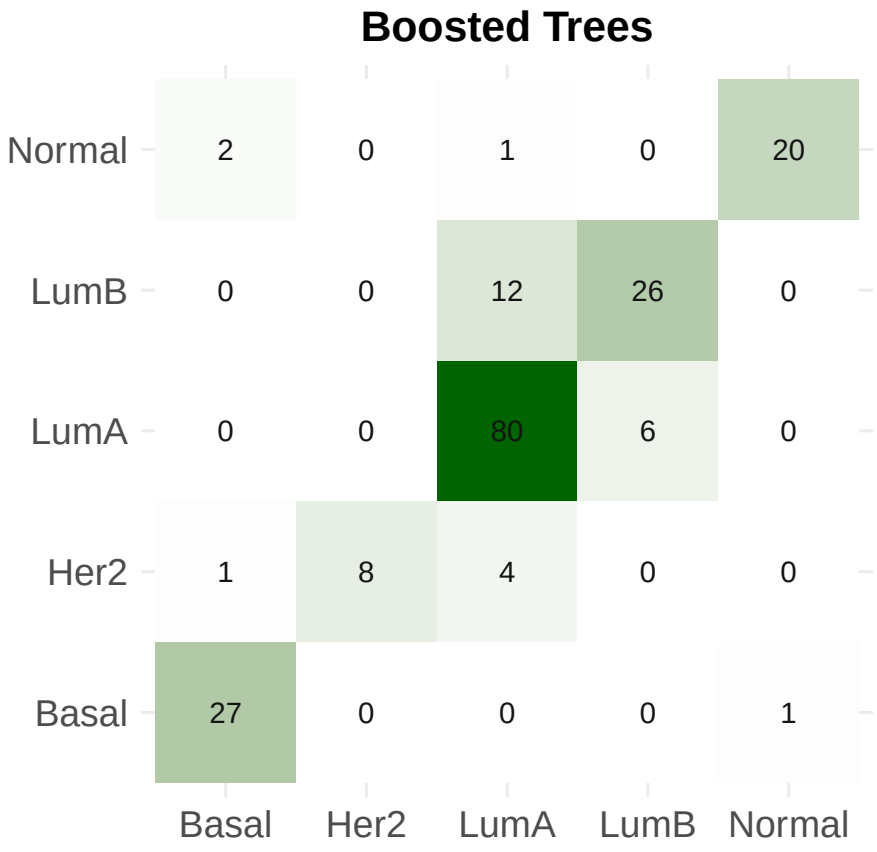

```r
# XGBoost
# The confusion matrix from a single assessment set (i.e. fold)
cm_xgboost <- table(predict(xgboost, test_data), test_labels) %>% as.data.frame

ggplot(cm_xgboost, aes(x = Var1, y = test_labels, fill = Freq)) +
  geom_tile() +
  theme_minimal() +
  coord_equal() +
  scale_fill_gradient(low = "white", high = "darkgreen") +
  labs(title = "XGBoost", x = NULL, y = NULL) +
  theme(
    plot.title = element_text(hjust = 0.5, size = 16, face = "bold"),
    axis.text = element_text(size = 14),
    axis.title = element_blank()
  ) +
  guides(fill = FALSE) +
  geom_text(aes(label = Freq), color = "grey8", size = 4)
```




```
# k-Nearest Neighbor
# The confusion matrix from a single assessment set (i.e. fold)
cm_knn <- table(predict(knn, test_data), test_labels) %>% as.data.frame

ggplot(cm_knn, aes(x = Var1, y = test_labels, fill = Freq)) +
  geom_tile() +
  theme_minimal() +
  coord_equal() +
  scale_fill_gradient(low = "white", high = "darkgreen") +
  labs(title = "k-Nearest Neighbor", x = NULL, y = NULL) +
  theme(
    plot.title = element_text(hjust = 0.5, size = 16, face = "bold"),
    axis.text = element_text(size = 14),
    axis.title = element_blank()
  ) +
  guides(fill = FALSE) +
  geom_text(aes(label = Freq), color = "grey8", size = 4)
```




```
# Boosted Trees
# The confusion matrix from a single assessment set (i.e. fold)
cm_bct <- table(predict(bct, test_data), test_labels) %>% as.data.frame

ggplot(cm_bct, aes(x = Var1, y = test_labels, fill = Freq)) +
  geom_tile() +
  theme_minimal() +
  coord_equal() +
  scale_fill_gradient(low = "white", high = "darkgreen") +
  labs(title = "Boosted Trees", x = NULL, y = NULL) +
  theme(
    plot.title = element_text(hjust = 0.5, size = 16, face = "bold"),
    axis.text = element_text(size = 14),
    axis.title = element_blank()
  ) +
  guides(fill = FALSE) +
  geom_text(aes(label = Freq), color = "grey8", size = 4)
```


Boosted Trees
Normal
LumB
LumA
Her2
Basal
2
0
1
0
20
0
0
12
26
0
0
0
80
6
0
1
8
4
0
0
27
0
0
0
1
Basal
Her2
LumA
LumB
Normal