Breast Cancer Detection

Şeyma Günönü

Supervised Learning: Classification Task

Classification task is the another task of supervised learning models. In this task, we will interest the classification problem about detecting women with breast cancer. In this classification task; Y: Status (Our target variable which has two values are dead and alive), X1: Age, X2: Race, X3: Marital Status, … (These are our features.).

Packages

We need to install some packages.

```
 #install.packages("DALEX")
 #install.packages("caret")
 #install.packages("ROCR")
 #install.packages("readr")
library(DALEX)
library(caret)
library(ROCR)
library(readr)
```

Dataset

We are using a data set about female patients with breast cancer diagnosed in 2006-2010 from Kaggle. The data set is a csv file, so we can use readr package to import the data set.

```
breast_cancer <- read_csv("Breast_Cancer.csv")
```

After adding our data set, we can use `str()` function to see details about the data set.

```
str(breast_cancer)
```

```
spc_tbl_ [4,024 x 16] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
 $ Age                   : num [1:4024] 68 50 58 58 47 51 51 40 40 69 ...
 $ Race                  : chr [1:4024] "White" "White" "White" "White" ...
 $ Marital Status        : chr [1:4024] "Married" "Married" "Divorced" "Married" ...
 $ T Stage               : chr [1:4024] "T1" "T2" "T3" "T1" ...
 $ N Stage               : chr [1:4024] "N1" "N2" "N3" "N1" ...
 $ 6th Stage             : chr [1:4024] "IIA" "IIIA" "IIIC" "IIA" ...
 $ differentiate         : chr [1:4024] "Poorly differentiated" "Moderately differentiated" "
 $ Grade                 : chr [1:4024] "3" "2" "2" "3" ...
 $ A Stage               : chr [1:4024] "Regional" "Regional" "Regional" "Regional" ...
 $ Tumor Size            : num [1:4024] 4 35 63 18 41 20 8 30 103 32 ...
 $ Estrogen Status       : chr [1:4024] "Positive" "Positive" "Positive" "Positive" ...
 $ Progesterone Status   : chr [1:4024] "Positive" "Positive" "Positive" "Positive" ...
 $ Regional Node Examined: num [1:4024] 24 14 14 2 3 18 11 9 20 21 ...
 $ Reginol Node Positive : num [1:4024] 1 5 7 1 1 2 1 1 18 12 ...
 $ Survival Months       : num [1:4024] 60 62 75 84 50 89 54 14 70 92 ...
 $ Status                : chr [1:4024] "Alive" "Alive" "Alive" "Alive" ...
 - attr(*, "spec")=
  .. cols(
  ..   Age = col_double(),
  ..   Race = col_character(),
  ..   `Marital Status` = col_character(),
  ..   `T Stage` = col_character(),
  ..   `N Stage` = col_character(),
  ..   `6th Stage` = col_character(),
  ..   differentiate = col_character(),
  ..   Grade = col_character(),
  ..   `A Stage` = col_character(),
  ..   `Tumor Size` = col_double(),
  ..   `Estrogen Status` = col_character(),
  ..   `Progesterone Status` = col_character(),
  ..   `Regional Node Examined` = col_double(),
  ..   `Reginol Node Positive` = col_double(),
  ..   `Survival Months` = col_double(),
  ..   Status = col_character()
  .. )
 - attr(*, "problems")=<externalptr>
```

Our variables are;

1. Age
2. Race
3. Marital Status
4. T Stage
5. N Stage
6. 6th Stage
7. differentiate
8. Grade
9. A Stage
10. Tumor Size
11. Estrogen Status
12. Progesterone Status
13. Regional Node Examined
14. Regional Node Positive
15. Survival Months
16. Status

When we look at the data set, we have just 5 numeric variables and the others are character. In this data set, our target variable is Status, because we want to detect the status of women with breast cancer.

```
breast_cancer <- na.exclude(breast_cancer)
```

First, we have to exclude all the NA's on the data set. After that we can check the dimension of the data set by using `dim()` function.

```
dim(breast_cancer)
```

```
[1] 4024   16
```

It has 4024 observations and 16 variables. Excluding 'NA' values did not affect the dimension, so we can say that the data set has no 'NA' values.

Training

We are checking the status of women with breast cancer, so we have a categorical variables which is our target variable. We have two values in the Status variable which are dead and alive. We have to assign them 0 and 1. After assigning them character, we should change their class to factor with two levels. It will be needed for training the model.

```r
breast_cancer["Status"][breast_cancer["Status"] == "Alive"] <- "1"
breast_cancer["Status"][breast_cancer["Status"] == "Dead"] <- "0"
```

```r
breast_cancer$Status <- as.factor(breast_cancer$Status)
class(breast_cancer$Status)
```

```
[1] "factor"
```

1 - Splitting the Data set

After editing our data set, we have to split the data set to detect target variable. First, we have to split the data set as a two subset by using `sample()` function. But first, we have to set a seed to get same observations. We create a sample that assigned as index observation. This sample starts with a sequence from 1 to number of row of breast_cancer, and we should split 80% of the data set as a train set and the remain part will be the test set.

```r
set.seed(123)
index <- sample(1 : nrow(breast_cancer), round(nrow(breast_cancer) * 0.80))
train <- breast_cancer[index, ]
test <- breast_cancer[-index, ]
```

2 - Training a logistic regression model

In this part, we want to train our model by using `glm()` function. We should pay attention to 3 things, when using the function. First, we should write the model formula which is target variable and the features as a '~'. Second, we use the train model that we set the previous part. The third one is adding the family distribution of the target variable. We are interesting the binary logistic regression that has a only two outcomes, so it will be set as a 'binomial'.

```r
lr_model <- glm(Status ~ ., data = train, family = "binomial")
lr_model
```

```
Call:  glm(formula = Status ~ ., family = "binomial", data = train)

Coefficients:
                        (Intercept)                                 Age
                          -1.294415                           -0.031739
                          RaceOther                           RaceWhite
                           0.867929                            0.515519
```

```
          `Marital Status`Married            `Marital Status`Separated
                         0.272381                           -0.620519
           `Marital Status`Single              `Marital Status`Widowed
                         0.218506                            0.109418
                      `T Stage`T2                         `T Stage`T3
                        -0.409576                           -0.967772
                      `T Stage`T4                         `N Stage`N2
                        -1.651890                           -0.813027
                      `N Stage`N3                       `6th Stage`IIB
                        -0.240321                           -0.212427
                  `6th Stage`IIIA                     `6th Stage`IIIB
                         0.544258                            0.274990
                  `6th Stage`IIIC  differentiatePoorly differentiated
                                NA                           -0.381933
     differentiateUndifferentiated    differentiateWell differentiated
                         -1.758900                            0.726406
                            Grade2                              Grade3
                                NA                                  NA
        Gradeanaplastic; Grade IV                   `A Stage`Regional
                                NA                           -0.227727
                    `Tumor Size`           `Estrogen Status`Positive
                         0.004066                            0.416672
      `Progesterone Status`Positive            `Regional Node Examined`
                         0.534972                            0.028956
           `Reginol Node Positive`                   `Survival Months`
                        -0.086619                            0.063085

Degrees of Freedom: 3218 Total (i.e. Null);  3193 Residual
Null Deviance:      2780
Residual Deviance: 1786     AIC: 1838
```

When we look at the output, we can saw that model formula, trained data, and coefficients which is estimated values of model parameters (betas). The coefficients show us the log odds ratios of the features. When we look at the Age feature, we cannot easily understand it. So interpret them easier, we can take odds ratios of these features for taking exponential of them by using `exp()` function.

```r
exp(lr_model$coefficients)
```

```
                    (Intercept)                             Age
                      0.2740580                       0.9687596
```

```
                     RaceOther                          RaceWhite
                     2.3819719                          1.6745066
       `Marital Status`Married          `Marital Status`Separated
                     1.3130875                          0.5376655
        `Marital Status`Single            `Marital Status`Widowed
                     1.2442167                          1.1156289
                   `T Stage`T2                        `T Stage`T3
                     0.6639317                          0.3799284
                   `T Stage`T4                        `N Stage`N2
                     0.1916873                          0.4435134
                   `N Stage`N3                       `6th Stage`IIB
                     0.7863758                          0.8086190
                `6th Stage`IIIA                      `6th Stage`IIIB
                     1.7233285                          1.3165177
               `6th Stage`IIIC differentiatePoorly differentiated
                            NA                          0.6825409
differentiateUndifferentiated   differentiateWell differentiated
                     0.1722342                          2.0676356
                        Grade2                             Grade3
                            NA                                 NA
    Gradeanaplastic; Grade IV                  `A Stage`Regional
                            NA                          0.7963416
                 `Tumor Size`          `Estrogen Status`Positive
                     1.0040741                          1.5169043
`Progesterone Status`Positive           `Regional Node Examined`
                     1.7074013                          1.0293791
      `Reginol Node Positive`                  `Survival Months`
                     0.9170264                          1.0651176
```

After taking exponential of them, we can tell that if we increase the value of age by 1 unit, it explains the target variable as approximately 0.96. Shortly, we multiply the odds ratio of target variable by 0.96.

```
summary(lr_model)
```

```
Call:
glm(formula = Status ~ ., family = "binomial", data = train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
```

```
-3.3413   0.1348   0.2623   0.4676   2.3556

Coefficients: (4 not defined because of singularities)
                                  Estimate Std. Error z value Pr(>|z|)
(Intercept)                     -1.294415   0.659200  -1.964  0.04957 *
Age                             -0.031739   0.007393  -4.293 1.76e-05 ***
RaceOther                        0.867929   0.318963   2.721  0.00651 **
RaceWhite                        0.515519   0.207703   2.482  0.01306 *
`Marital Status`Married          0.272381   0.185532   1.468  0.14208
`Marital Status`Separated       -0.620519   0.529481  -1.172  0.24122
`Marital Status`Single           0.218506   0.229882   0.951  0.34185
`Marital Status`Widowed          0.109418   0.287810   0.380  0.70382
`T Stage`T2                     -0.409576   0.267279  -1.532  0.12543
`T Stage`T3                     -0.967772   0.421580  -2.296  0.02170 *
`T Stage`T4                     -1.651890   0.654122  -2.525  0.01156 *
`N Stage`N2                     -0.813027   0.322273  -2.523  0.01164 *
`N Stage`N3                     -0.240321   0.410256  -0.586  0.55802
`6th Stage`IIB                  -0.212427   0.307610  -0.691  0.48983
`6th Stage`IIIA                  0.544258   0.396693   1.372  0.17007
`6th Stage`IIIB                  0.274990   0.759259   0.362  0.71722
`6th Stage`IIIC                        NA         NA      NA       NA
differentiatePoorly differentiated -0.381933   0.138259  -2.762  0.00574 **
differentiateUndifferentiated   -1.758900   0.845417  -2.081  0.03748 *
differentiateWell differentiated  0.726406   0.238369   3.047  0.00231 **
Grade2                                 NA         NA      NA       NA
Grade3                                 NA         NA      NA       NA
Gradeanaplastic; Grade IV              NA         NA      NA       NA
`A Stage`Regional               -0.227727   0.363445  -0.627  0.53094
`Tumor Size`                     0.004066   0.005289   0.769  0.44207
`Estrogen Status`Positive        0.416672   0.258087   1.614  0.10643
`Progesterone Status`Positive    0.534972   0.171746   3.115  0.00184 **
`Regional Node Examined`         0.028956   0.008816   3.284  0.00102 **
`Reginol Node Positive`         -0.086619   0.020392  -4.248 2.16e-05 ***
`Survival Months`                0.063085   0.003146  20.051  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2780.2  on 3218  degrees of freedom
Residual deviance: 1786.4  on 3193  degrees of freedom
AIC: 1838.4
```

```
Number of Fisher Scoring iterations: 6
```

We use `summary()` function to see more details, and the output shows us the model formula, residuals, and coefficients. We can see the results and make inferences using these statistics for each parameter. İf any class of a categorical variable is significant that means the others are significant too. (P-value is lower than the significance level (0.05).) When we look at the N Stage, we can see just N2 is significant, but it does not mean the others not significant. It means that N Stage is significant.

3 - Measuring model performance

After looking the summary of the trained data, we can check the model performance of the model on test set. First, we have to calculate the predicted value of target variable on test set. We should not forget to exclude the target variable on test set. We determine the type as a response, because we can calculate the predicted values of the target variable like that. Then predicting the values, we can check the first six values using `head()` function.

```
predicted_probs <- predict(lr_model, test[ ,-16], type = "response")
head(predicted_probs)
```

```
        1         2         3         4         5         6
0.8432563 0.9899364 0.6367093 0.7404053 0.9491666 0.6107392
```

This vector has a probability of women with breast cancer. After that we should transform these probabilities to classes by using `ifelse()` function. We set the condition like that. If there is greater than 0.5, it means "Alive". If it is smaller than 0.05, it assigned "dead". And it returns us the first six values, and it will be "1" because these are the alive women values with breast cancer.

```
predicted_classes <- ifelse(predicted_probs > 0.5, 1, 0)
head(predicted_classes)
```

```
1 2 3 4 5 6
1 1 1 1 1 1
```

We can create confusion matrix using the metrics which is based on the confusion matrix. We should assign the positive and negative classes as a 1 and 0.

```
TP <- sum(predicted_classes[which(test$Status == "1")] == 1)
FP <- sum(predicted_classes[which(test$Status == "1")] == 0)
TN <- sum(predicted_classes[which(test$Status == "0")] == 0)
```

```r
FN <- sum(predicted_classes[which(test$Status == "0")] == 1)

recall      <- TP / (TP + FN)
specificity <- TN / (TN + FP)
precision   <- TP / (TP + FP)
accuracy    <- (TN + TP) / (TP + FP + TN + FN)

recall
```

```
[1] 0.9102041
```

```r
specificity
```

```
[1] 0.7142857
```

```r
precision
```

```
[1] 0.9709724
```

```r
accuracy
```

```
[1] 0.8931677
```

When we look at the calculated model performance metrics, especially the precision is the highest one, and the model classify the 97% of the model. According to accuracy of the model is classified 89% correctly. If precision approaches to 1, it means the model is perfect. Sometimes it may cause a misleading, because it can understand the model is accurate. We are not sure about that, so we should check the imbalancedness of the classes.

```r
table(train$Status) / dim(train)[1]
```

```
        0         1
0.1553277 0.8446723
```

Checking the imbalancedness of the classes is important, it may cause the model learn less from the minority or the majority class and the model cannot learn the correctly way. In this set, we can tell that the classes of the target variable is not balanced, and it may cause a problem.

```
confusionMatrix(table(ifelse(test$Status == "1", "1", "0"),
                      predicted_classes),
                positive = "1")
```

```
Confusion Matrix and Statistics

   predicted_classes
      0   1
  0  50  66
  1  20 669

               Accuracy : 0.8932
                 95% CI : (0.8697, 0.9137)
    No Information Rate : 0.913
    P-Value [Acc > NIR] : 0.978

                  Kappa : 0.4814

 Mcnemar's Test P-Value : 1.219e-06

            Sensitivity : 0.9102
            Specificity : 0.7143
         Pos Pred Value : 0.9710
         Neg Pred Value : 0.4310
             Prevalence : 0.9130
         Detection Rate : 0.8311
   Detection Prevalence : 0.8559
      Balanced Accuracy : 0.8122

       'Positive' Class : 1
```

The confusion matrix shows us the how many observation values the data set has. We have observations of predicted classes, accuracy, precision, and the others. We set the positive as "1", because 1 which means alive is our intended class. The accuracy shows us the overall performance of the model, and it is 89%. Positive prediction value means that precision, and it

is 97%. The other important one is balanced accuracy. It gives us the average of the accuracy in each classes independently.

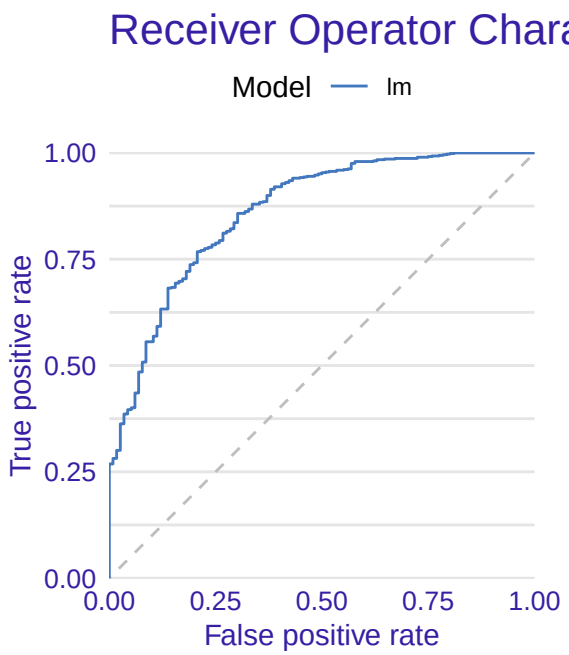
ROC Curve and AUC

We can construct the metrics using ROC curve. First, we need to create an explainer.

```
explain_lr <- explain(model = lr_model,
                      data = test[ , -16],
                      y = test$Status == "1",
                      type = "classification",
                      verbose = FALSE)
```

After that, we can draw ROC curve.

```
performance_lr <- model_performance(explain_lr)
plot(performance_lr, geom = "roc")
```



When we look at the graph the ROC curve should be above the dashed line. If the ROC curve is close to this line that means the model is a random model and it does not perform well. But we can tell that our model is not close, so it is good.

```
performance_lr
```

```
Measures for:  classification
recall     : 0.9709724
precision  : 0.9102041
f1         : 0.9396067
accuracy   : 0.8931677
auc        : 0.8623943

Residuals:
          0%          10%          20%          30%          40%          50%
-0.982202733 -0.302637308  0.006436929  0.011499261  0.020552443  0.035012123
         60%          70%          80%          90%         100%
 0.056762992  0.088278876  0.139532283  0.242106512  0.866188429
```

Calculating the area under curve (auc) value, we can check the performance_lr. The auc value is 86%. Model performance would have been excellent if its exact or approximate value had been 1. Our model is 0.86, so it is close to the perfect.

Checking Imbalancedness Problem

We mentioned above the codes about the imbalancedness. When we check the target variables observations on the train set, it gives us the dead ones is 500, and the alive ones is 2719.

```
table(train$Status)
```

```
   0    1
 500 2719
```

We should balance them using under or over-sampling. In the below section, we used under-sampling method. Under-sampling brings the alive ones to the bottom, and it will be equal with dead ones. After that the problem will be solved.

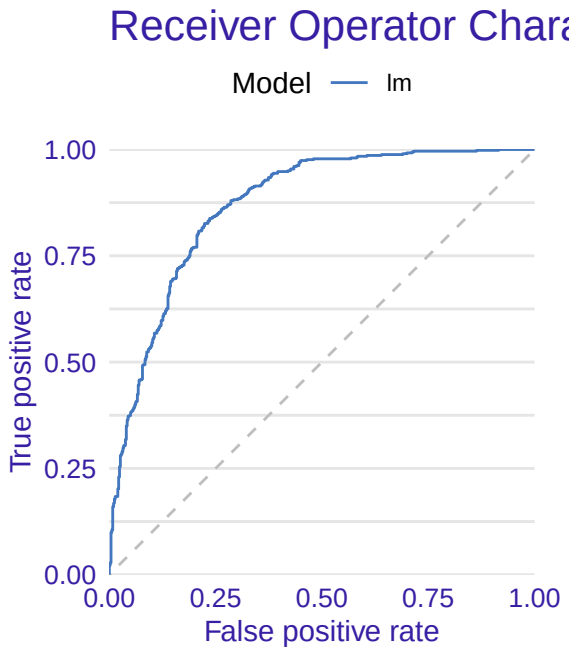
```
under_train <- downSample(x = train[, -ncol(train)],
                          y = train$Status)
table(under_train$Class)
```

```
  0   1
500 500
```

We look at the imbalancedness problem, but we need to check the auc for the changes.

```
explain_under <- explain(model = lr_model,data = under_train[, -16],
                         y = under_train$Class == "1",
                         type = "classification",
                         verbose = FALSE)
```

```
performance_under <- model_performance(explain_under)
plot(performance_under, geom = "roc")
```




```
performance_under
```

```
Measures for:  classification
recall     : 0.978
precision  : 0.6400524
f1         : 0.7737342
accuracy   : 0.714
```

```
auc        : 0.870092

Residuals:
          0%          10%          20%          30%          40%          50%
-0.996235152 -0.863983086 -0.678415561 -0.411516090 -0.155747212 -0.001130646
         60%          70%          80%          90%         100%
 0.014720768  0.034983570  0.073933681  0.151601406  0.937620151
```

The auc value is 87%, and it close the previous one. When we look at the accuracy, it decreases. We can see clearly, the precision is decreased too. The previous one is close to 1, bu this one is not that perfect.