Northeastern University

ALY 6015
Assignment 3

OMKAR PRADEEP RAUT

Date of Submission: 02/05/2022

INTRODUCTION

Generalized linear model (GLM) is a generalization of ordinary linear regression that allows for response variables that have error distribution models other than a normal distribution like Gaussian distribution. In R, using lm() is a special case of glm(). lm() fits models following the form Y = Xb + e, where e is Normal (0 , s^2). glm() fits models following the form f(Y) = Xb + e. However, in glm both the function f(Y) (the 'link function') and the distribution of the error term e can be specified. Logistic regression is yet another technique borrowed by machine learning from the field of statistics. It's a powerful statistical way of modeling a binomial outcome with one or more explanatory variables. It measures the relationship between the categorical dependent variable and one or more independent variables by estimating probabilities using a logistic function, which is the cumulative logistic distribution. In this assignment , we will study an example of a binary logistic regression, which we will tackle with the ISLR package, which will provide us with the data set, and the glm() function, which is generally used to fit generalized linear models, will be used to fit the logistic regression model.

Analysis

- Loading Data and necessary libraries

The first thing to do is to install and load the ISLR package, which has all the datasets you're going to use.

```
library(ISLR)
library(caret)
library(ggplot2)
library(gridExtra)
library(pROC)
```

Above are the R libraries which we are going to use in our programming.
Now we will Load the data set which is College dataset which comes in ISLR library.

```
# Dataset
```{r}

collegeData<- College
collegeData
attach(collegeData)
summary(collegeData)
```

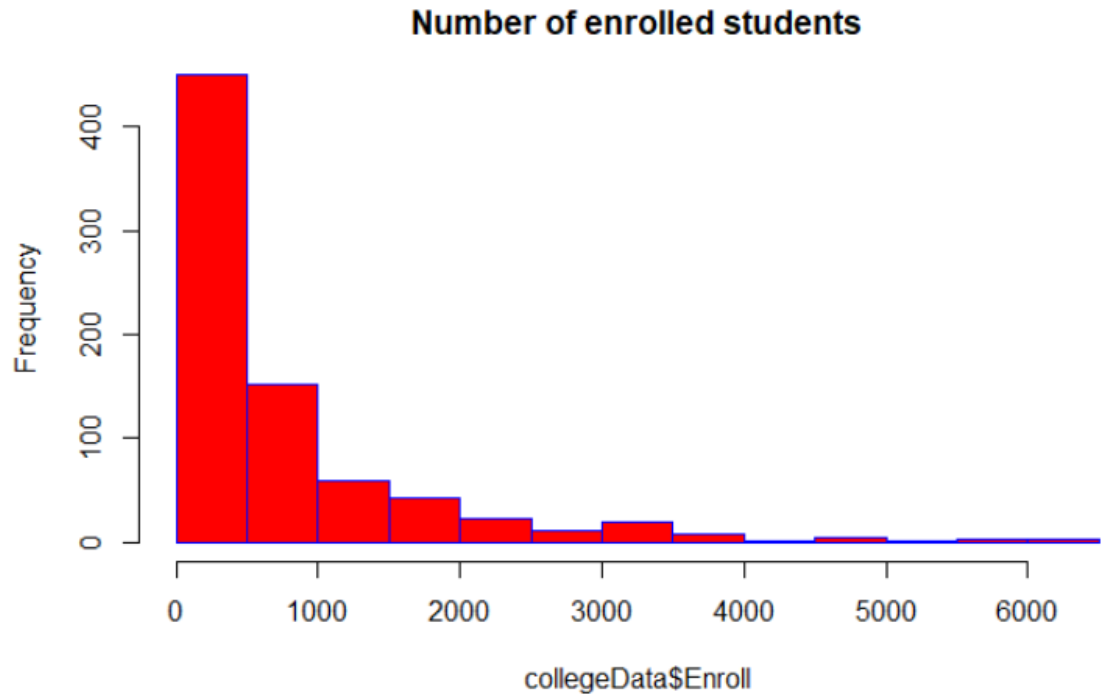
We have used collegeData as a variable to store our dataset. It consist of 777 observation with 18 attributes. Some of the attributes are Private, Accept, Enroll , Top10perc and many more.

Now, we will perform Exploratory Data Analysis by using descriptive statistics. Data visualization is perhaps the fastest and most useful way to summarize and learn more about your data. Following are some of the plot for the data.
```

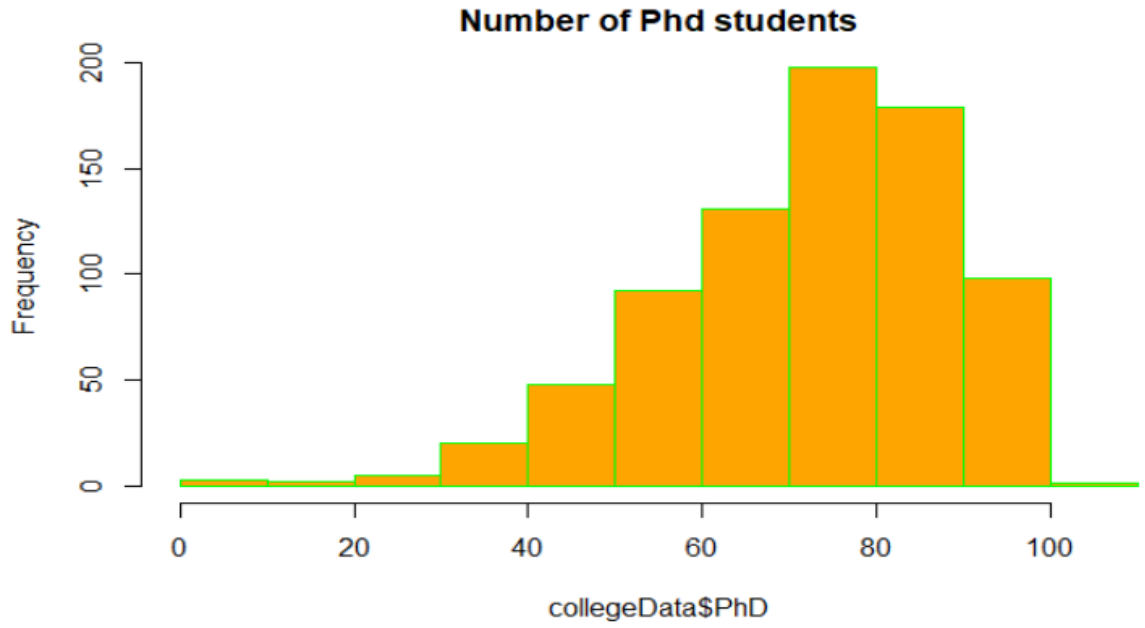
Exploratory Data Analysis

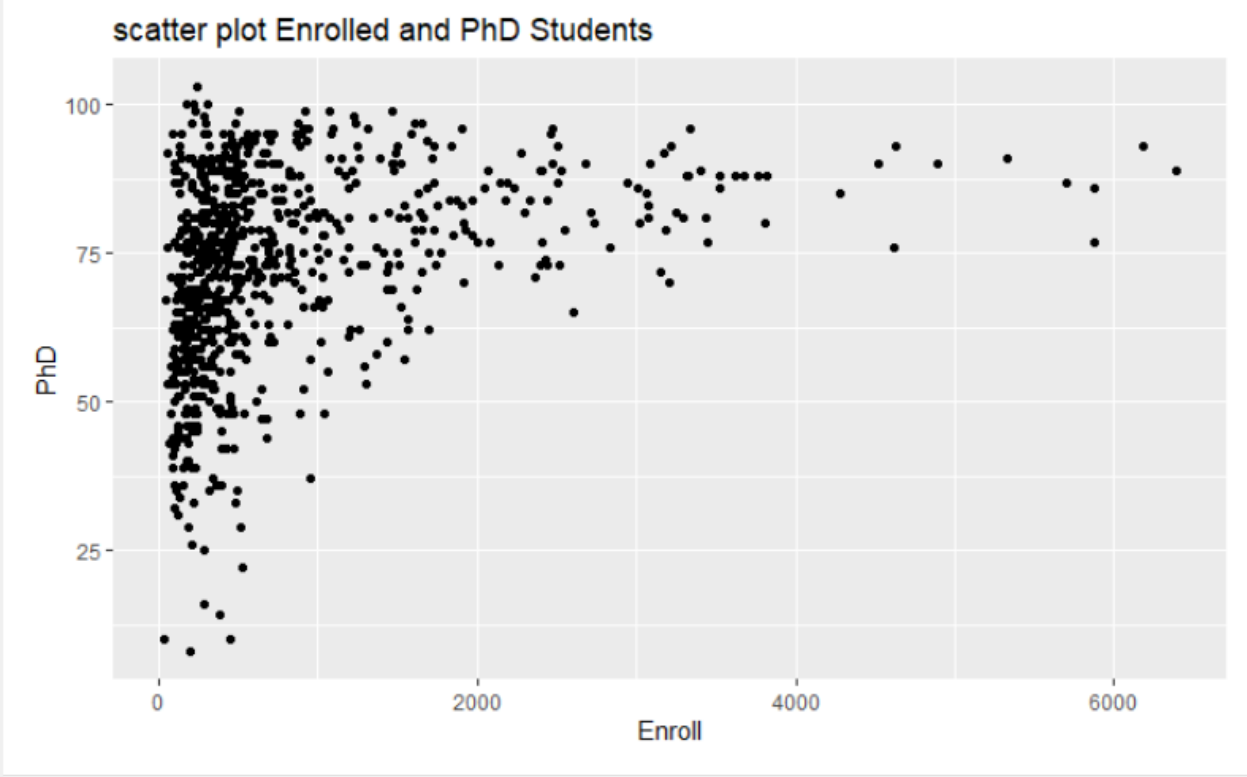
```{r}
#Scatterplot
hist(collegeData$Enroll, main="Number of enrolled students",col="red",  border="blue")
hist(collegeData$PhD, main="Number of Phd students",col="orange",  border="green")
ggplot(collegeData, aes(Enroll, PhD)) + geom_point() +ggtitle('scatter plot Enrolled and PhD Students')
plot(Outstate ~ Private, data = collegeData, col =c("green", "blue"))

```



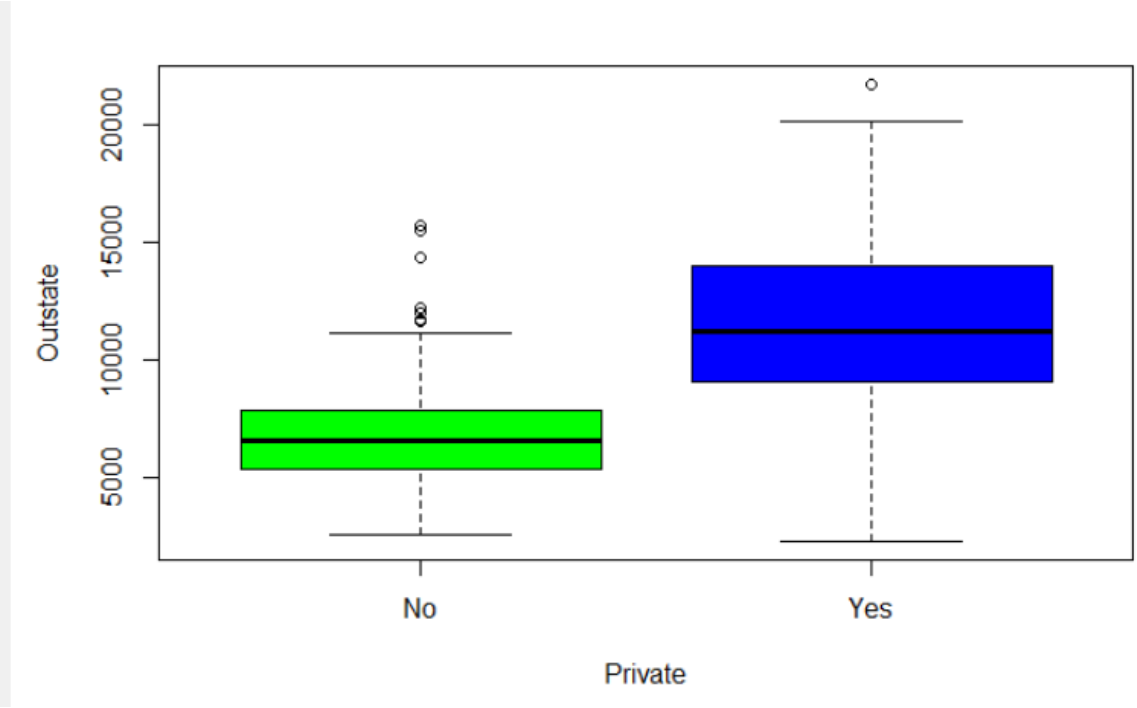
Above is the histogram of the number of Students enrolled.





Above is the ScatterPlot of Enrolled and PhD students.

Following is the boxplot of Outstate versus Private.



In this assignment, we will learn how to split sample into training and test data sets with R.

To maintain same percentage of event rate in both training and validation dataset.

Here we are performing stratified sampling using caret package. The createDataPartition function from caret package generates a stratified random split of the data. In simple words, suppose you have event rate (mean of binary dependent variable) of 10%. The program below makes sure this 10% holds in both training and validation dataset. 70% of data goes in training and remaining 30% in validation dataset.

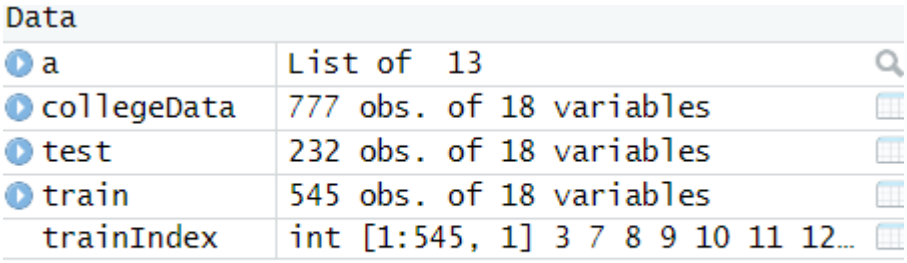
```
# Split Data into Train and Test sets

```{r}

set.seed(123)
trainIndex <- createDataPartition(College$Enroll,p=0.70,list=FALSE)
train <- College[trainIndex,]
test <- College[-trainIndex,]
head(train)

```
```

From the above code, we got 545 training data and 232 test data.



Data	
a	List of 13
collegeData	777 obs. of 18 variables
test	232 obs. of 18 variables
train	545 obs. of 18 variables
trainIndex	int [1:545, 1] 3 7 8 9 10 11 12…

Now we will proceed to out next step which **is Fitting a Logistic Regression Model**. We call glm.fit() function. The first argument that I pass to this function is an R formula. In this case, the formula indicates that Private is the response, while the Enroll and Phd variables are the predictors. As you saw in the introduction, glm is generally used to fit generalized linear models.

```
model2<-glm(Private~Enroll + PhD, data=train, family = binomial(link = "logit"))
summary(model2)
```

```
Call:
glm(formula = Private ~ Enroll + PhD, family = binomial(link = "logit"),
    data = train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.2549  -0.2219   0.5097   0.5883   3.4165

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  1.5017751  0.5064605   2.965  0.00302 **
Enroll      -0.0018131  0.0001987  -9.125  < 2e-16 ***
PhD          0.0135837  0.0073761   1.842  0.06554 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 633.45  on 544  degrees of freedom
Residual deviance: 474.83  on 542  degrees of freedom
AIC: 480.83

Number of Fisher Scoring iterations: 5

 (Intercept)       Enroll          PhD
 1.501775120 -0.001813063  0.013583678
(Intercept)      Enroll         PhD
  4.4896517   0.9981886   1.0136764
```

As you can see, summary() returns the estimate, standard errors, z-score, and p-values on each of the coefficients. Look like none of the coefficients are significant here. It also gives you the null deviance (the deviance just for the mean) and the residual deviance (the deviance for the model with all the predictors). There's a very small difference between the 2, along with 2 degrees of freedom.

.

Making Prediction for Train Data and Test Data

```
probabilities.train <- predict(model2, newdata=train, type = "response")
predicted.classes.min <- as.factor(ifelse(probabilities.train >=0.5, "Yes", "No"))


head(predicted.classes.min)
```

Now I am going to make a prediction of whether the probabilities of train data set is 1 or zero in our case we have use “Yes” and “No” based on the lags and other predictors. In particular, I'll turn the probabilities into classifications by thresholding at 0.5. In order to do so, I use an if else command.

```
        Adrian College   Albertus Magnus College          Albion College 
                   Yes                       Yes                       Yes 
       Albright College Alderson-Broaddus College       Alfred University 
                   Yes                       Yes                       Yes 
Levels: No Yes
```

You can see the probabilities of colleges have probability greater than or equal to 0.5 have responded as YES.

A Confusion matrix is an N x N matrix used for evaluating the performance of a classification model, where N is the number of target classes. The matrix compares the actual target values with those predicted by the machine learning model. This gives us a holistic view of how well our classification model is performing and what kinds of errors it is making.

We are going to check the accuracy of train dataset by using confusion matrix.

```{r}
#Question 4&5:
#Model Accuracy
confusionMatrix(predicted.classes.min, train$Private, positive = "Yes")
```

```
Confusion Matrix and Statistics

          Reference
Prediction  No Yes
       No   67  17
       Yes  79 382

               Accuracy : 0.8239
                 95% CI : (0.7892, 0.8549)
    No Information Rate : 0.7321
    P-Value [Acc > NIR] : 3.017e-07

                  Kappa : 0.4811

 Mcnemar's Test P-Value : 4.791e-10

            Sensitivity : 0.9574
            Specificity : 0.4589
         Pos Pred Value : 0.8286
         Neg Pred Value : 0.7976
             Prevalence : 0.7321
         Detection Rate : 0.7009
   Detection Prevalence : 0.8459
      Balanced Accuracy : 0.7081

       'Positive' Class : Yes
```

As you can see the Accuracy of train model comes around 0.8239 whereas the sensitivity aka Recall is 0.9574. Precision for train model is 0.8286 and Specificity which is a true negative rate is 0.4589.

Now, we are going to check the accuracy of test dataset by using confusion matrix.

```
#Question 6: confusion matrix for test

probabilities.test <- predict(model2, newdata=test, type = "response")
predicted.classes.min <- as.factor(ifelse(probabilities.test >=0.5, "Yes", "No"))

confusionMatrix(predicted.classes.min, test$Private, positive = "Yes")

#calculate total misclassification error rate
misClassError(test$Private, predicted, threshold=optimal)


```

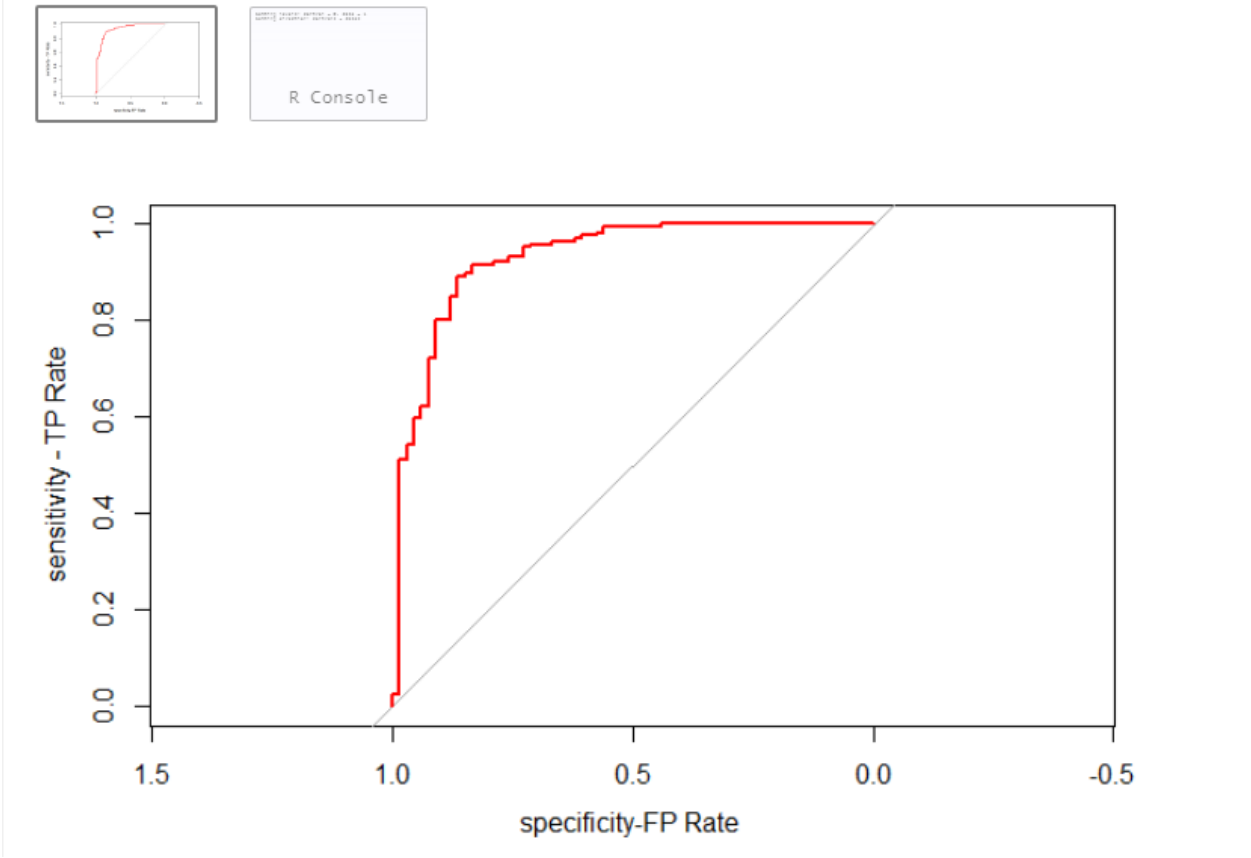
```
Confusion Matrix and Statistics

          Reference
Prediction  No Yes
       No   38   3
       Yes  28 163

               Accuracy : 0.8664
                 95% CI : (0.8157, 0.9074)
    No Information Rate : 0.7155
    P-Value [Acc > NIR] : 3.665e-08

                  Kappa : 0.6295

 Mcnemar's Test P-Value : 1.629e-05

            Sensitivity : 0.9819
            Specificity : 0.5758
         Pos Pred Value : 0.8534
         Neg Pred Value : 0.9268
             Prevalence : 0.7155
         Detection Rate : 0.7026
   Detection Prevalence : 0.8233
      Balanced Accuracy : 0.7788

       'Positive' Class : Yes
```

As you can see the Accuracy of test model comes around 0.8664 whereas the sensitivity aka Recall is 0.9819. Precision for test model which is positive predicted value is 0.8534 and Specificity which is a true negative rate is 0.5758.

Though both of these are a problem, a false negative is more damaging because it lets a problem go undetected, creating a false sense of security

An ROC curve (receiver operating characteristic curve) is a graph showing the performance of a classification model at all classification thresholds. This curve plots two parameters: True Positive Rate. False Positive Rate.

```
ROC1<- roc(test$default, probabilities.test)
plot(ROC1,col="RED", ylab= "sensitivity - TP Rate", xlab="specificity-FP Rate ")
```



A ROC curve is constructed by plotting the true positive rate (TPR) against the false positive rate (FPR). The true positive rate is the proportion of observations that were correctly predicted to be positive out of all positive observations (TP/(TP + FN)). Similarly, the false

positive rate is the proportion of observations that are incorrectly predicted to be positive out of all negative observations (FP/(TN + FP)). The ROC curve shows the trade-off between sensitivity (or TPR) and specificity (1 – FPR). Classifiers that give curves closer to the top-left corner indicate a better performance. As a baseline, a random classifier is expected to give points lying along the diagonal (FPR = TPR). The closer the curve comes to the 45-degree diagonal of the ROC space, the less accurate the test.

Calculating the area under ROC Curve

```{r}
auc<-auc(ROC1)
auc
```

```
Area under the curve: 0.9475
```

To compare different classifiers, it can be useful to summarize the performance of each classifier into a single measure. One common approach is to calculate the area under the ROC curve, which is abbreviated to AUC. It is equivalent to the probability that a randomly chosen positive instance is ranked higher than a randomly chosen negative instance

REFERENCE

- “Logistic Regression in R.” DataCamp Community, https://www.datacamp.com/community/tutorials/logistic-regression-R.
- Zach. (2021, April 1). How to create a confusion matrix in R (step-by-step). Statology. Retrieved February 6, 2022, from https://www.statology.org/confusion-matrix-in-r/
- Valchanov, I. (2021, June 11). False positive and false negative. Medium. Retrieved February 6, 2022, from https://towardsdatascience.com/false-positive-and-false-negative-b29df2c60aca