Wisconsin Diagnostic Breast Cancer (WDBC)

Marina Vallejo Vallés (marina.vallejo01@estudiant.upf.edu (mailto:marina.vallejo01@estudiant.upf.edu))

Last update: 10 abril, 2022

Brief introduction.

The Wisconsin Diagnostic Breast Cancer (WDBC) contains features computed from a digitized image of a fine needle aspirate (FNA) of a breast mass.

In the literature they report that the two diagnosis results: B = benign, M = malignant are linearly separable using all 30 input features available in the data set. They also report that the best predictive accuracy is obtained using one separating plane in the 3-D space of Worst Area, Worst Smoothness and Mean Texture. With an estimated accuracy 97.5% using repeated 10-fold cross-validations.

EXPLORATORY DATA ANALYSIS

Work with the data set.

```
# Access the data:
breast_dataset <- read.csv("data_WDBC.csv", sep=",")
# Summary of the data:
summary(breast_dataset)
```

```
##        id             diagnosis          radius_mean      texture_mean  
##  Min.   :     8670   Length:569         Min.   : 6.981   Min.   : 9.71  
##  1st Qu.:   869218   Class :character   1st Qu.:11.700   1st Qu.:16.17  
##  Median :   906024   Mode  :character   Median :13.370   Median :18.84  
##  Mean   : 30371831                      Mean   :14.127   Mean   :19.29  
##  3rd Qu.:  8813129                      3rd Qu.:15.780   3rd Qu.:21.80  
##  Max.   :911320502                      Max.   :28.110   Max.   :39.28  
##  perimeter_mean     area_mean      smoothness_mean   compactness_mean 
##  Min.   : 43.79   Min.   : 143.5   Min.   :0.05263   Min.   :0.01938  
##  1st Qu.: 75.17   1st Qu.: 420.3   1st Qu.:0.08637   1st Qu.:0.06492  
##  Median : 86.24   Median : 551.1   Median :0.09587   Median :0.09263  
##  Mean   : 91.97   Mean   : 654.9   Mean   :0.09636   Mean   :0.10434  
##  3rd Qu.:104.10   3rd Qu.: 782.7   3rd Qu.:0.10530   3rd Qu.:0.13040  
##  Max.   :188.50   Max.   :2501.0   Max.   :0.16340   Max.   :0.34540  
##  concavity_mean    concave.points_mean symmetry_mean    fractal_dimension_mean
##  Min.   :0.00000   Min.   :0.00000     Min.   :0.1060   Min.   :0.04996       
##  1st Qu.:0.02956   1st Qu.:0.02031     1st Qu.:0.1619   1st Qu.:0.05770       
##  Median :0.06154   Median :0.03350     Median :0.1792   Median :0.06154       
##  Mean   :0.08880   Mean   :0.04892     Mean   :0.1812   Mean   :0.06280       
##  3rd Qu.:0.13070   3rd Qu.:0.07400     3rd Qu.:0.1957   3rd Qu.:0.06612       
##  Max.   :0.42680   Max.   :0.20120     Max.   :0.3040   Max.   :0.09744       
##    radius_se        texture_se      perimeter_se       area_se       
##  Min.   :0.1115   Min.   :0.3602   Min.   : 0.757   Min.   :  6.802  
##  1st Qu.:0.2324   1st Qu.:0.8339   1st Qu.: 1.606   1st Qu.: 17.850  
##  Median :0.3242   Median :1.1080   Median : 2.287   Median : 24.530  
##  Mean   :0.4052   Mean   :1.2169   Mean   : 2.866   Mean   : 40.337  
##  3rd Qu.:0.4789   3rd Qu.:1.4740   3rd Qu.: 3.357   3rd Qu.: 45.190  
##  Max.   :2.8730   Max.   :4.8850   Max.   :21.980   Max.   :542.200  
##  smoothness_se      compactness_se      concavity_se     concave.points_se 
##  Min.   :0.001713   Min.   :0.002252   Min.   :0.00000   Min.   :0.000000  
##  1st Qu.:0.005169   1st Qu.:0.013080   1st Qu.:0.01509   1st Qu.:0.007638  
##  Median :0.006380   Median :0.020450   Median :0.02589   Median :0.010930  
##  Mean   :0.007041   Mean   :0.025478   Mean   :0.03189   Mean   :0.011796  
##  3rd Qu.:0.008146   3rd Qu.:0.032450   3rd Qu.:0.04205   3rd Qu.:0.014710  
##  Max.   :0.031130   Max.   :0.135400   Max.   :0.39600   Max.   :0.052790  
##   symmetry_se       fractal_dimension_se  radius_worst   texture_worst  
##  Min.   :0.007882   Min.   :0.0008948    Min.   : 7.93   Min.   :12.02  
##  1st Qu.:0.015160   1st Qu.:0.0022480    1st Qu.:13.01   1st Qu.:21.08  
##  Median :0.018730   Median :0.0031870    Median :14.97   Median :25.41  
##  Mean   :0.020542   Mean   :0.0037949    Mean   :16.27   Mean   :25.68  
##  3rd Qu.:0.023480   3rd Qu.:0.0045580    3rd Qu.:18.79   3rd Qu.:29.72  
##  Max.   :0.078950   Max.   :0.0298400    Max.   :36.04   Max.   :49.54  
##  perimeter_worst    area_worst     smoothness_worst  compactness_worst
##  Min.   : 50.41   Min.   : 185.2   Min.   :0.07117   Min.   :0.02729  
##  1st Qu.: 84.11   1st Qu.: 515.3   1st Qu.:0.11660   1st Qu.:0.14720  
##  Median : 97.66   Median : 686.5   Median :0.13130   Median :0.21190  
##  Mean   :107.26   Mean   : 880.6   Mean   :0.13237   Mean   :0.25427  
##  3rd Qu.:125.40   3rd Qu.:1084.0   3rd Qu.:0.14600   3rd Qu.:0.33910  
##  Max.   :251.20   Max.   :4254.0   Max.   :0.22260   Max.   :1.05800  
##  concavity_worst  concave.points_worst symmetry_worst   fractal_dimension_worst
##  Min.   :0.0000   Min.   :0.00000      Min.   :0.1565   Min.   :0.05504        
##  1st Qu.:0.1145   1st Qu.:0.06493      1st Qu.:0.2504   1st Qu.:0.07146        
##  Median :0.2267   Median :0.09993      Median :0.2822   Median :0.08004        
##  Mean   :0.2722   Mean   :0.11461      Mean   :0.2901   Mean   :0.08395        
##  3rd Qu.:0.3829   3rd Qu.:0.16140      3rd Qu.:0.3179   3rd Qu.:0.09208        
##  Max.   :1.2520   Max.   :0.29100      Max.   :0.6638   Max.   :0.20750        
##      X          
##  Mode:logical   
##  NA's:569       
##                 
##                 
##                 
##                 
```

The Breast Cancer Dataset contains a total number of 33 variables and 569 observations. The variables `X` and `id` are not interesting for our analysis, so we won't consider them later. There are no missing values in this data set, we don't need to perform imputation. All the variables are numeric, except `Diagnosis` , which is stored as a logical variable, this must be taken into account for future steps.

Check diagnosis:

```
cat(" Number cases with malignant tumor: ",sum(breast_dataset$diagnosis == "M"),"\n", "Number cases with benign tumor: ",sum
(breast_dataset$diagnosis == "B"))
```

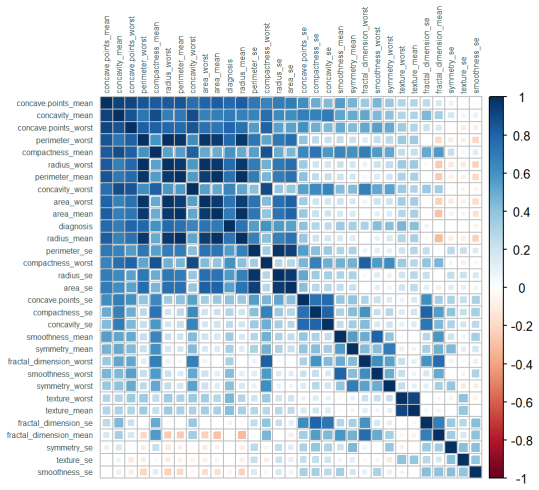
```
##  Number cases with malignant tumor:  212 
##  Number cases with benign tumor:  357
```

```
cat(" From a total number of", nrow(breast_dataset), "tumor cases, ", (sum(breast_dataset$diagnosis == "M")/nrow(breast_data
set)*100), "% are malignant tumors.")
```

```
##  From a total number of 569 tumor cases,  37.25835 % are malignant tumors.
```

Pearson correlation:

```
# Select part of the original data, as some variables are not of our interest:
breast_df <- breast_dataset[,3:ncol(breast_dataset)-1]
breast_df$diagnosis <- as.integer(factor(breast_df$diagnosis))-1

breast_correlations <- cor(breast_df,method="pearson")

# Generate correlation plot:
corrplot(breast_correlations, hclust.method = "ward",method = "square",
          order = "FPC", type = "full", tl.col = "darkslategray",tl.cex=0.5)
```



As it is a huge matrix it is better not to include in the report the table of correlation values.

Note that diagnosis has been changed, now Benign tumors are indicated as 0 and Malignant tumors as 1.

MODEL BUILDING

Before starting with model building, we must pre-process the data. We will change the type of variable diagnosis and normalize data (with the custom made function *normalize*):

```
# Create function:
normalize <- function(x) {
  return ((x - min(x)) / (max(x) - min(x)))
}

# Generate a seed to replicate later the results :
set.seed(123)

# Diagnosis as an integer:
breast_dataset$diagnosis = as.integer(factor(breast_dataset$diagnosis))-1

# Normalize data:
breast_dataset_norm <- as.data.frame(lapply(breast_dataset[2:31], normalize))
```

Now we move to the model building. The first step is to separate the data in training and test:

```
# Generate the split object:
breast_split <- data_split <- initial_split(breast_dataset_norm, prop = 3/4)

# Build the training breast data set (with 3/4 of the data)
breast_training <- breast_split %>% training()
# Obtain labels, it's necessary to keep track of labels but in a separate variable:
breast_training_labels <- breast_training[, 1]
# Remove labels from training set:
breast_training <- as.data.frame(breast_training[-1])

# Build the testing breast data set:
breast_test <- breast_split %>% testing()
# Obtain labels:
breast_test_labels <- breast_test[, 1]
# Remove labels from test set:
breast_test <- as.data.frame(breast_test[-1])
```

KNN

In this part of the report, we will build and test a model based on **K-nearest neighbors (KNN)** algorithm.

KNN is a classification method that estimates the likelihood that a data point will become a member of one group or another, calculating Euclidean distances.

It is a supervised machine learning algorithm, we must provide the labels of the samples, in this case the diagnosis (Malignant (M), Benign (B)).

As an initial parameter we must choose the number of nearest neighbors to include (K). In order to generate a model with high accuracy, we must test different K values. In this report, we have K = 21 as the performance of the model was appropriate (high accuracy and low number of false positive and negative), but other K values that lead to lower accuracy models were tried previously. It was a trial-error process.

```
# Run KNN
set.seed(111)
knn_pred <- knn(train = breast_training, test = breast_test, cl = breast_training_labels, k=21,prob=TRUE)
```

Check the accuracy:

```
cat("The accuracy of the KNN model is:",sum(knn_pred == breast_test_labels)/nrow(breast_test))
```

```
## The accuracy of the KNN model is: 0.9370629
```

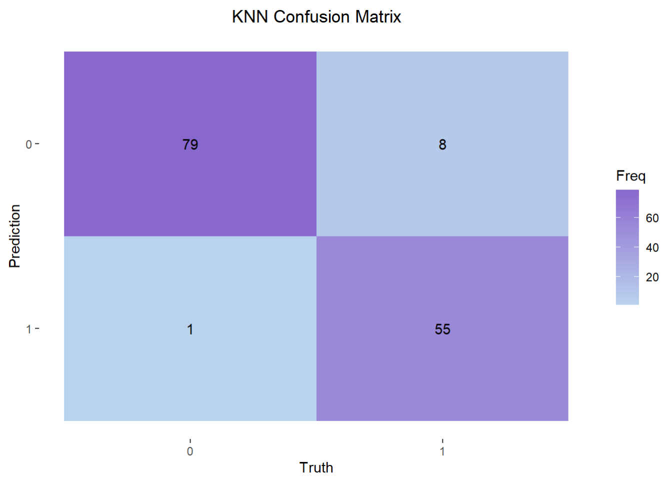
Check results:

```
CrossTable(x = breast_test_labels, y = knn_pred, prop.chisq=FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  143
##
##
##                    | knn_pred
## breast_test_labels |         0 |         1 | Row Total |
## -------------------|-----------|-----------|-----------|
##                  0 |        79 |         1 |        80 |
##                    |     0.988 |     0.012 |     0.559 |
##                    |     0.908 |     0.018 |           |
##                    |     0.552 |     0.007 |           |
## -------------------|-----------|-----------|-----------|
##                  1 |         8 |        55 |        63 |
##                    |     0.127 |     0.873 |     0.441 |
##                    |     0.092 |     0.982 |           |
##                    |     0.056 |     0.385 |           |
## -------------------|-----------|-----------|-----------|
##       Column Total |        87 |        56 |       143 |
##                    |     0.608 |     0.392 |           |
## -------------------|-----------|-----------|-----------|
##
##
```

```
# Generate a Confusion Matrix:
conf_matrix <- data.frame(knn_pred,breast_test_labels)

# Rename Columns
names(conf_matrix) <- c("Predicted", "Actual")

conf_matrix <- conf_mat(conf_matrix,  Actual,Predicted)

autoplot(conf_matrix, type = "heatmap") +
  scale_fill_gradient(low="slategray2",high = "mediumpurple3") +
  theme(legend.position = "right") + labs(title = "KNN Confusion Matrix")+
  theme(plot.title = element_text(hjust = 0.5))
```

```
## Scale for 'fill' is already present. Adding another scale for 'fill', which
## will replace the existing scale.
```



ROC CURVE:

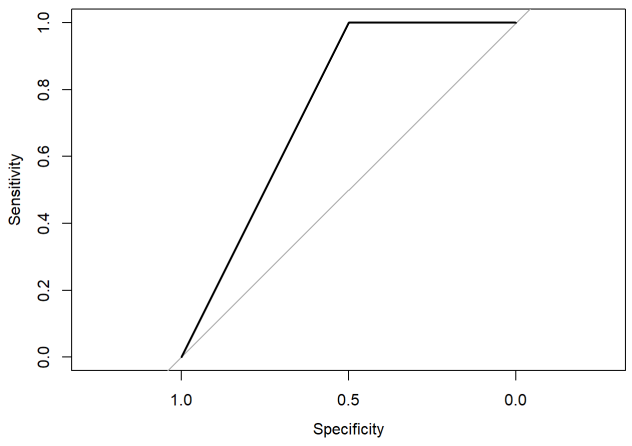
```
knn.ROC <- roc(predictor=breast_test_labels, response=attributes(knn_pred)$prob)
```

```
## Warning in roc.default(predictor = breast_test_labels, response =
## attributes(knn_pred)$prob): 'response' has more than two levels. Consider
## setting 'levels' explicitly or using 'multiclass.roc' instead
```

```
## Setting levels: control = 0.571428571428571, case = 0.619047619047619
```

```
## Setting direction: controls < cases
```

```
plot(knn.ROC)
```



Area Under the Curve (AUC):

```
knn.ROC$auc
```

```
## Area under the curve: 0.75
```

The previous Cross Table and confusion matrix allows to identify True positives (55 cases with malignant tumor), True negatives (79 cases with benignant tumor).

We can also identify False negatives (8 cases that **KNN** predicted as negative, but in fact were positive) and False positives (1 case predicted as positive when it was negative).

The ROC plot and Area Under the curve gives us insights in the model performance. The closer the AUC value to 1, the better prediction. The **KNN** model implemented in the report has an AUC = 0.75, further changes should be done in order to improve it. For example, the model could be improved with a previous pre-processing of data doing a z-score standardization, instead of the normalization done in this report.

Decision Tree

Decision Tree is a classification algorithm that has a flowchart similar to a tree structure.

This model classifies instances by sorting them down the tree from the root to any leaf node, which provides the classification of the instance. It contains internal nodes that denote a test on a specific attribute, while each branch represents an outcome of the test. The terminal nodes contain class labels.

The instances are classified by sorting them from the root to a leaf node, that will provide the classification of the instance.

It is a non-parametric supervised learning method.

In order to generate the Decision Tree model we will use the function C5.0:

```
# Transform labels to factor, necessary for the correct flow of the function:

breast_training$diagnosis<-as.factor(breast_training_labels)

breast_test['diagnosis'] = breast_test_labels
breast_test$diagnosis<-as.factor(breast_test$diagnosis)

# Set seed:
set.seed(1111)
# Generate the model:
dt_model <-C5.0(diagnosis ~., data = breast_training)

# Check:
dt_model
```

```
##
## Call:
## C5.0.formula(formula = diagnosis ~ ., data = breast_training)
##
## Classification Tree
## Number of samples: 426
## Number of predictors: 29
##
## Tree size: 7
##
## Non-standard options: attempt to group attributes
```

Now make predictions with out testing dataset:

```
dt_pred<- predict(dt_model, breast_test, type="class")
```

Check the accuracy:

```
cat("The accuracy of the Decision Tree model is:",sum(dt_pred == breast_test$diagnosis)/nrow(breast_test))
```

```
## The accuracy of the Decision Tree model is: 0.9300699
```

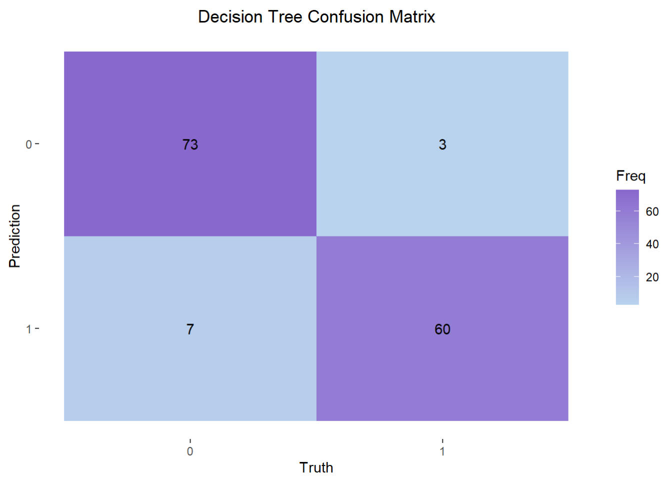
Now we have to compare the predicted labels to the real labels of the test dataset:

```
# Check results
CrossTable(x = breast_test_labels, y = dt_pred, prop.chisq=FALSE)
```

```
## 
## 
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
## 
## 
## Total Observations in Table:  143 
## 
## 
##                    | dt_pred 
## breast_test_labels |         0 |         1 | Row Total | 
## -------------------|-----------|-----------|-----------|
##                  0 |        73 |         7 |        80 | 
##                    |     0.912 |     0.087 |     0.559 | 
##                    |     0.961 |     0.104 |           | 
##                    |     0.510 |     0.049 |           | 
## -------------------|-----------|-----------|-----------|
##                  1 |         3 |        60 |        63 | 
##                    |     0.048 |     0.952 |     0.441 | 
##                    |     0.039 |     0.896 |           | 
##                    |     0.021 |     0.420 |           | 
## -------------------|-----------|-----------|-----------|
##       Column Total |        76 |        67 |       143 | 
##                    |     0.531 |     0.469 |           | 
## -------------------|-----------|-----------|-----------|
## 
## 
```

```
# Generate a Confusion Matrix:
conf_matrix2 <- data.frame(dt_pred,breast_test_labels)

# Rename Columns
names(conf_matrix2) <- c("Predicted", "Actual")

conf_matrix2 <- conf_mat(conf_matrix2,  Actual,Predicted)

autoplot(conf_matrix2, type = "heatmap") +
  scale_fill_gradient(low="slategray2",high = "mediumpurple3") +
  theme(legend.position = "right") + labs(title = "Decision Tree Confusion Matrix")+
  theme(plot.title = element_text(hjust = 0.5))
```

```
## Scale for 'fill' is already present. Adding another scale for 'fill', which
## will replace the existing scale.
```



Comparing the Confusion matrices, for the **Decision Tree model**, we can see that the algorithm has a better performance than **KNN** when detecting positive cases, those that are malignant tumors. From a total number of 63 malignant tumors, it predicted 60. This result is better than KNN, which only predicted 55 cases. But on the other hand, we can see that with Decision Tree has a greater number of false positives (7 vs 1 in KNN), this model is prone to type 1 errors.

Neural Network

Neural Networks consist of an artificial network of functions. These functions also called parameters (or neurons), allow to learn and to modify internal elements by learning from new data. Each neuron after getting an input, produces an output. The outputs are sent to the next of neurons, and so on. The process continues all layers have been considered, and then the terminal neurons then output the final result of the model.

Generate a Neural Network model. The parameters are the ones recommended in literature:

```
set.seed(1111)
nn_model <-nnet(diagnosis ~. , data=breast_training, size=15, rang = 1, decay = 8e-4, maxit = 100)
```

```
## # weights:  466
## initial  value 257.930455 
## iter  10 value 31.020161
## iter  20 value 14.462695
## iter  30 value 5.715691
## iter  40 value 4.426506
## iter  50 value 3.681904
## iter  60 value 3.274219
## iter  70 value 3.084947
## iter  80 value 2.982445
## iter  90 value 2.934833
## iter 100 value 2.902831
## final  value 2.902831 
## stopped after 100 iterations
```

Prediction using the previously generated Neural Network model:

```
nn_pred <- predict(nn_model, breast_test,type = c("class"))
```

Check the accuracy:

```
cat("The accuracy of the Neural Network model is:",sum(nn_pred == breast_test$diagnosis)/nrow(breast_test))
```

```
## The accuracy of the Neural Network model is: 0.972028
```

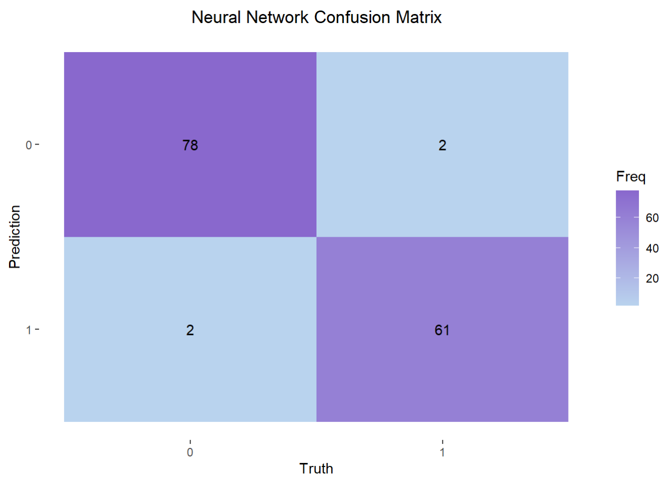
Check the numeric results in a table format:

```
CrossTable(breast_test$diagnosis, nn_pred, prop.chisq = FALSE,prop.c = FALSE, prop.r = FALSE, dnn = c("Actual", "Predicted"
))
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  143
##
##
##              | Predicted
##       Actual |         0 |         1 | Row Total |
## -------------|-----------|-----------|-----------|
##            0 |        78 |         2 |        80 |
##              |     0.545 |     0.014 |           |
## -------------|-----------|-----------|-----------|
##            1 |         2 |        61 |        63 |
##              |     0.014 |     0.427 |           |
## -------------|-----------|-----------|-----------|
## Column Total |        80 |        63 |       143 |
## -------------|-----------|-----------|-----------|
##
##
```

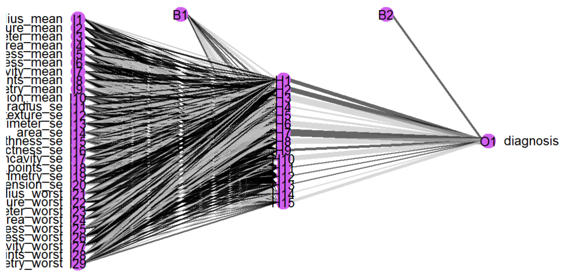
```
# Generate a Confusion Matrix:
conf_matrix3 <- data.frame(nn_pred,breast_test_labels)

# Rename Columns
names(conf_matrix3) <- c("Predicted", "Actual")

conf_matrix3 <- conf_mat(conf_matrix3,  Actual,Predicted)

autoplot(conf_matrix3, type = "heatmap") +
  scale_fill_gradient(low="slategray2",high = "mediumpurple3") +
  theme(legend.position = "right") + labs(title = "Neural Network Confusion Matrix")+
  theme(plot.title = element_text(hjust = 0.5))
```

```
## Scale for 'fill' is already present. Adding another scale for 'fill', which
## will replace the existing scale.
```



Now plot the Neural-network as a diagram:

```
plotnet(nn_model, alpha = 0.6,circle_cex = 2,cex_val = 0.8,circle_col = "mediumorchid2",bord_col = "mediumorchid2", prune_co
l = TRUE)
```



If we compare with the results of **Neural Network** with the previous Confusion Matrices of **KNN** and **Decision Tree**, we can see that this model has the best performance. It has the lowest number of false positives as well as false negatives. Also, we can see that the accuracy is very high.

We can also check the diagram and see the conformation of layers of the model. ## Logistic Regression

Logistic Regression is another classification algorithm that can be used to assign observations to a discrete set.

It is based on probabilities and the cost function is Sigmoid (or Logistic Function).

```
lr_model <-train(diagnosis~.,data=breast_training,method="glm",family=binomial())

varImp(lr_model)
```

```
## glm variable importance
##
##   only 20 most important variables shown (out of 29)
##
##                          Overall
## concavity_se              100.00
## compactness_mean           94.26
## smoothness_mean            84.20
## fractal_dimension_mean     73.76
## concave.points_se          59.38
## radius_mean                58.85
## radius_worst               53.18
## area_worst                 46.93
## smoothness_se              41.62
## symmetry_worst             41.52
## fractal_dimension_se       40.20
## perimeter_mean             36.58
## symmetry_mean              35.86
## symmetry_se                34.51
## compactness_se             32.97
## perimeter_se               30.89
## concavity_worst            24.88
## area_mean                  24.52
## concave.points_mean        15.42
## radius_se                  13.20
```

Only 20 most important variables shown (out of 29).

Prediction:

```
set.seed(11111)
lr_pred<-predict(lr_model,breast_test)
```

Check the accuracy:

```
cat("The accuracy of the Logistic Regression is:",sum(lr_pred == breast_test$diagnosis)/nrow(breast_test))
```

```
## The accuracy of the Logistic Regression is: 0.958042
```

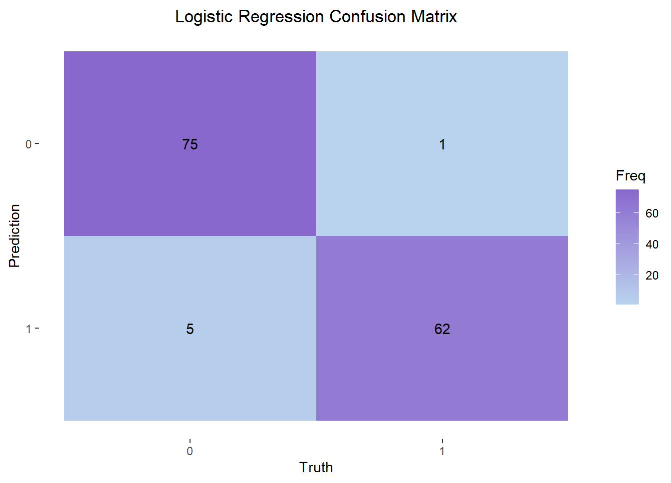
Check the numeric results in a table format:

```
CrossTable(x = breast_test_labels, y = lr_pred, prop.chisq=FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  143
##
##
##                    | lr_pred
## breast_test_labels |         0 |         1 | Row Total |
## -------------------|-----------|-----------|-----------|
##                  0 |        75 |         5 |        80 |
##                    |     0.938 |     0.062 |     0.559 |
##                    |     0.987 |     0.075 |           |
##                    |     0.524 |     0.035 |           |
## -------------------|-----------|-----------|-----------|
##                  1 |         1 |        62 |        63 |
##                    |     0.016 |     0.984 |     0.441 |
##                    |     0.013 |     0.925 |           |
##                    |     0.007 |     0.434 |           |
## -------------------|-----------|-----------|-----------|
##       Column Total |        76 |        67 |       143 |
##                    |     0.531 |     0.469 |           |
## -------------------|-----------|-----------|-----------|
##
##
```

```
# Generate a Confusion Matrix:
conf_matrix4 <- data.frame(lr_pred,breast_test_labels)

# Rename Columns
names(conf_matrix4) <- c("Predicted", "Actual")

conf_matrix4 <- conf_mat(conf_matrix4,  Actual,Predicted)

autoplot(conf_matrix4, type = "heatmap") +
  scale_fill_gradient(low="slategray2",high = "mediumpurple3") +
  theme(legend.position = "right") + labs(title = "Logistic Regression Confusion Matrix")+
  theme(plot.title = element_text(hjust = 0.5))
```

```
## Scale for 'fill' is already present. Adding another scale for 'fill', which
## will replace the existing scale.
```



From the Confusion Matrix we can see that this model has a greater number of false positives than the Neural Network, but a lower number of false negatives (we also have to take into account that the difference is not that big, only by 1 case).

This model is better at not mislabeling malignant tumors as benign, but instead, it mislabels a greater number of benign tumors as malignant.

Which of all models performs better for this data? Discuss.

According to the accuracy, the rank would be (from higher accuracy to lower): Neural Network, Logistic Regression, KNN and Decision Tree.

Of all the models, the **Neural Network** model is the one with a better performance (it has the highest accuracy value, 0.972028).