



EXPLANATORY DATA ANALYSIS

Monika, Gena, Mustapha, Fazal



CONTENTS:

Section 1: Introduction	2
Section 2: Introduction to the used techniques.....	3
Section 3: Feedback about the provided information.....	5
Section 4: Comparisson between both models.....	9
Section 5: Explanation of the code.....	12

EXPLANATORY DATA ANALYSIS

SECTION 1: INTRODUCTION

Heart disease covers a range of different conditions that could affect the heart. It is one of the most complex diseases to predict, given the number of potential factors in the body that can lead to it. Healthcare organizations collect and produce large volumes of data on daily basis. Information technology allows automatization of processes for extraction of data that help to get useful knowledge and regularities.

Goal

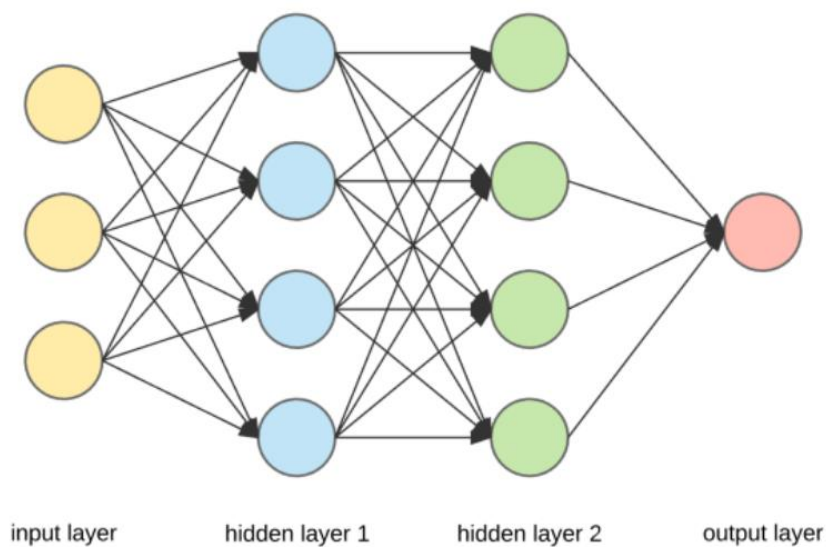
By using this easy-to-use model, you can predict if a patient has heart disease.

The model is based on the Random Forest Classification technique and it achieves accuracy of around 95%. Our objective is to deliver the model, which may save lives and decrease the cost of the healthcare services.

Before being delivered, the functionality of this model was carefully tested. We will proceed with further details about the model, the methods

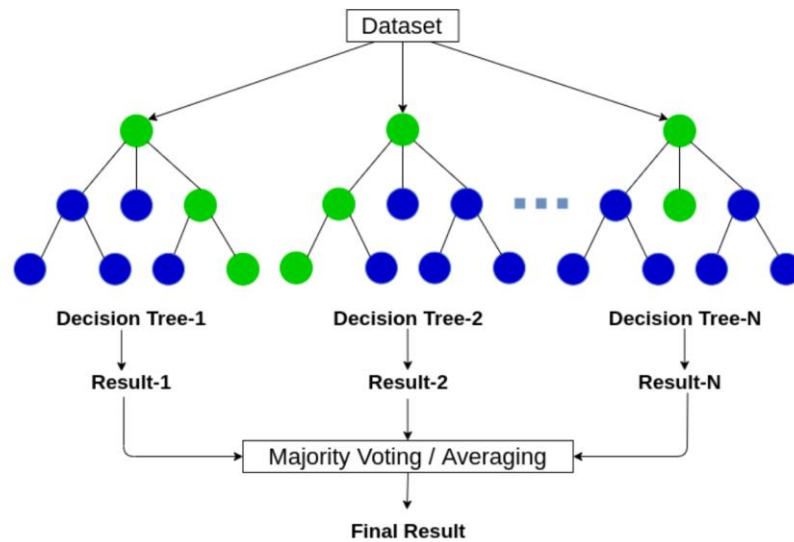
Neural Network

A neural network is a network or circuit of neurons, or in a modern sense, an artificial neural network, composed of artificial neurons or nodes. Thus, a neural network is either a biological neural network, made up of biological neurons, or an artificial neural network, for solving artificial intelligence (AI) problems. Example of an artificial neural network with two hidden layers:




Random Forest Classifier

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time. For classification tasks, the output of the random forest is the class selected by most trees. Example of a random forest classifier, see bellow:



For regression tasks, the mean or average prediction of the individual trees is returned. Random decision forests correct for decision trees' habit of overfitting to their training set. Random forests generally outperform decision trees, but their accuracy is lower than gradient boosted trees. However, data characteristics can affect their performance.

SECTION 3: FEEDBACK ABOUT THE PROVIDED INFORMATION

In this section is explained about the difficulties we encountered while following the pdf instructions “Hear_Disease”, and working with the provided dataset  heart.csv

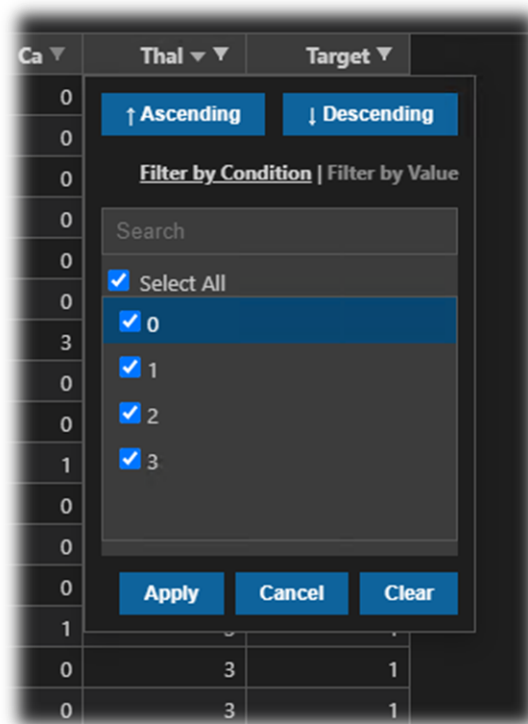
1. In the PDF file with instructions you sent to us: “Hear_Disease”, there is an attribute “thal” that it is with three values (see screenshot of the pdf bellow):

ca: The number of major vessels (0 – 3)

thal: A blood disorder called thalassemia (3 = normal; 6 = fixed defect; 7 = reversable defect)

target: Heart disease (0 = no, 1 = yes)

However, in the dataset itself “heart.csv” in the column “thal” the values are not three, but four (see bellow):

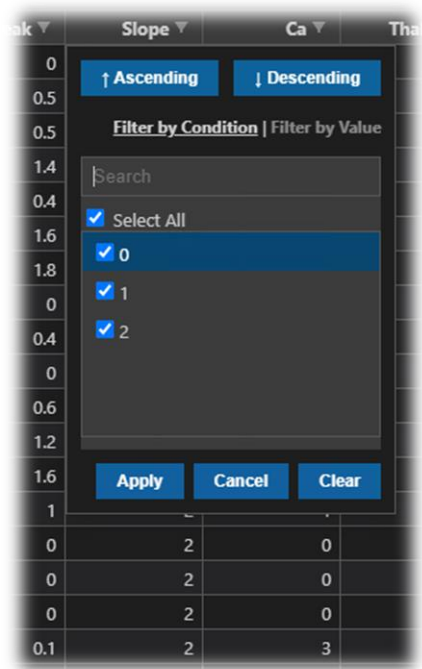


2. In the PDF file with instructions you sent to us: Hear_Disease, there is an attribute “slope” where the numbers are as follows:

ECG plot)

slope: The slope of the peak exercise ST segment (value 1: upsloping, value 2: flat, value 3: downsloping)

However, in the dataset itself “heart.csv”, in the column “slope” the values are 0, 1, 2 instead.

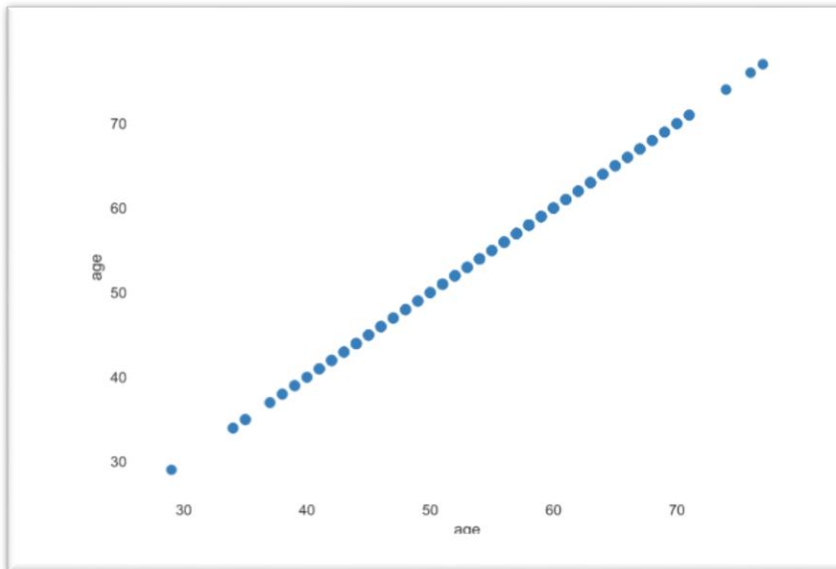


3. Number of records: 303 rows.

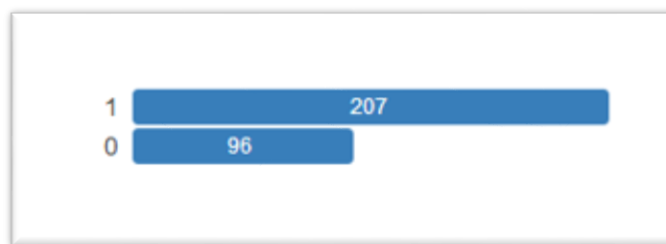
The amount of data is insufficient to extract even more findings. It would be much better if we should receive more data in order to provide better prediction.

Dataset statistics	
Number of variables	14
Number of observations	303

4. Insufficient amount of data between the age groups 0 to 35 and above 70.

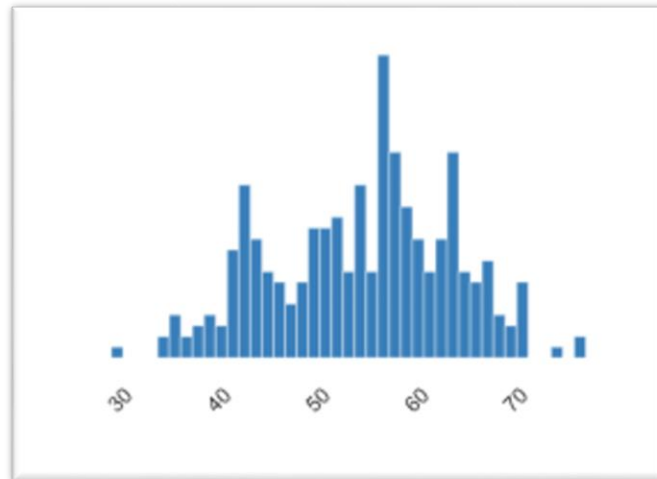


5. Around two-third of the data is for the male gender and the rest – only one-third is for the female gender.



6. About 45% of the data is for patients between the age of 52 to 62. We definitely need a wider age range.

Statistics Histogram Common values Extreme values			
Value	Count	Frequency (%)	
58	19	6.3%	
57	17	5.6%	
54	16	5.3%	
59	14	4.6%	
52	13	4.3%	
51	12	4.0%	
62	11	3.6%	
60	11	3.6%	
44	11	3.6%	
56	11	3.6%	
Other values (31)	168	55.4%	



7. Based on the data the factors below are highly correlated to the heart disease so they should be taken into consideration.
- cp
 - thalach
 - thal
 - exang
8. The attributes “oldpeak” and “slops” are highly related to each other, however they are not strongly correlated to the heart disease.

SECTION 4: COMPARISSON BETWEEN BOTH MODELS

Artificial Neural Network:

The Artificial Neural Network model has been built as follows:

Layers	Neurons	Activation Functions
Single Input layer "layer_1"	13	Sigmoid
One Hidden layer "layer_2"	9	Sigmoid
Output layer "1"	1	Sigmoid

```
network = models.Sequential()

# Input Layer
network.add(layers.Dense(layer_1, activation=activ[0], input_shape =
(X.shape[1],)))

# Hidden Layer
network.add(layers.Dense(layer_2, activation=activ[0]))

# Output Layer
network.add(layers.Dense(1, activation=activ[0]))
```

`X.shape[1]` → The shape of the features we fed to the model = 19

Afterwards, the model has been compiled and fitted as:

```
network.compile(loss='binary_crossentropy',
optimizer=tf.optimizers.Adam(learning_rate=0.001), metrics=['accuracy'])

-----
history = network.fit(X_train, y_train, epochs=epoch, shuffle=True, verbose=1,
validation_split= 0.09, batch_size=5)
```

The maximum test accuracy of the Artificial Neural Network model after multiple tuning of hyperparameters we achieved is 85%.

Test Accuracy: 0.8524590134620667

Our motivation to reach a higher performance lead us to choose another predicting technique. After carefully consideration of various methods, we chose the Machine Learning algorithm Random Forest Classifier.

Random Forest Classifier:

After importing and cleaning the data, we split the data using Sklearn. That means 80 % of the data (X_train) is going to be used for the training of the model and 20% for testing (X_test):

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20)
```

Next, it is necessary to import the Random Forest Classifier library to use the model.

To pick up the best model we run a series of trainings (fitting) and predictions by recording the score of each configuration by the changing the number of trees (n_estimation). This inquiry is done by the following part of the code with a for-loop:

```
from sklearn.metrics import accuracy_score

Estimation_list_min = 1
Estimation_list_max = 5000

scores = []
n_estimation_list = []
max_accuracy = 0

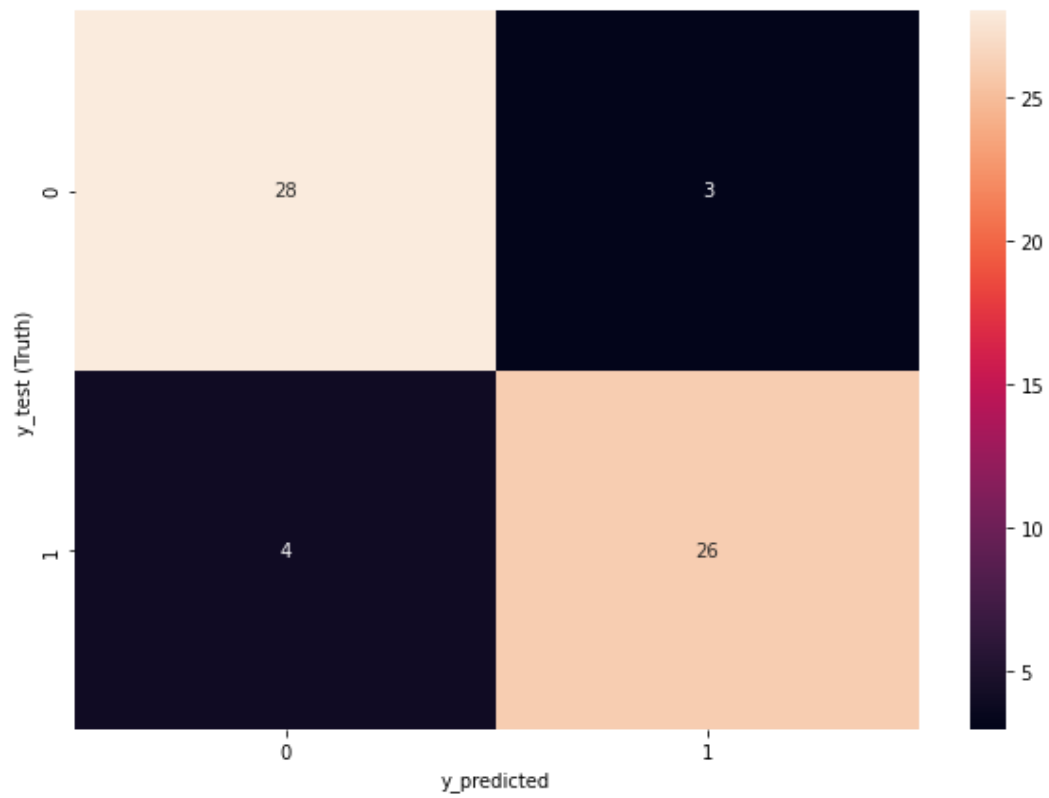
for tree in range(Estimation_list_min, Estimation_list_max):
    model = RandomForestClassifier(random_state=tree)
    model.fit(X_train, y_train)
    y_predicted = model.predict(X_test)
    scores.append(accuracy_score(y_test, y_predicted))
    n_estimation_list.append(tree)

    current_accuracy = round(accuracy_score(y_predicted, y_test)*100,2)
    if(current_accuracy>max_accuracy):
        max_accuracy = current_accuracy
        best_x = tree
```

Using the Random Forest technique, we reached the amazing test accuracy of 95%. This could be clearly seen below under the line of code:

```
print(max_accuracy, best_x)
[343] ✓ 0.6s
... 95.08 4818
```

The confusion matrix below represents the excellent results as well:



SECTION 5: EXPLANATION OF THE CODE

1. Required packages and libraries are being imported at the start of the code. Which includes Pandas, Seaborn, Matplotlib, Keras, Scikitlearn, Tensorflow, Numpy.
2. Read the .csv file with the Pandas and saved it as a data frame.
3. Matplotlib and Seaborn are then used to plot and visualize the dataset.
4. The most important part of visualization is to select the features with the highest correlation with the target column. We plotted a correlation chart to have the insights and then we selected the features accordingly.
5. Useless or less correlated features are dropped with “.drop()” command.
6. As some of the features are categorical. We have converted them to dummy variables, added them to our data frame and then removed the original columns.
7. We defined X (features) and y(targets) to feed it to our model.
8. X and y are then split using “train_test_split()” method between Training data and Test data (X_train, X_test, y_train and y_test).
9. The shape of the parameters is checked once before feeding it to the model.
10. The model is created using “RandomForestClassifier()” command. The model in our case has been placed inside the loop. The purpose of the loop is to run the model numerous times in order to change the hyper-parameters automatically each time it runs. The scores have been saved at every run in a list and with the help of an if-statement, we have saved the best results for the whole loop.
11. A plot has been created for the accuracy vs. the hyper-parameters.
12. A confusion Matrix is plotted to check the predictions.
13. The model is saved with the help of Pickle and can be re-loaded when required.