Machine Learning Model

Module Code: **CS3AI18**
Assignment Report Title: **Machine Learning Model**
Student Numbers: **28010336**
Date Completed: **21/03/2022**
Actual hours spent for the assignment: **24 hours**
Assignment evaluation (3 key points):
1) The number of expected hours spent for this assignment was unrealistic for me personally.
2) I have learnt how to preprocess data.
3) I have gained an understanding on how a machine learning model is created.

Abstract

The report contains the findings and evaluation of the Machine Learning Model. I had extracted a Pandas dataframe from the provided dataset and had prepared the data by applying preprocessing and standardisation techniques. Then data visualisations were produced to determine the most relevant features. Afterwards, I was able to develop a Machine Learning Model and finally was able to train and evaluate it. The model was at least 80% accurate in predicting the values.

Chosen Problem

The problem that I have chosen to solve is to predict the likelihood of a person getting a heart attack. The link to the dataset is included in the References section.

Background

A heart attack is caused by the restriction of blood flow to the heart. High blood pressure is a big risk factor, as it produces excess strain on the coronary arteries. This results in the damaging of coronary arteries. and causes the coronary arteries to narrow from a process called atherosclerosis, which is the build-up of cholesterol and other substances overtime. Eventually, a plaque is formed, hardening the arteries and, in turn increasing the likelihood of blood clots. Atherosclerosis leads to a blockage, interrupting the flow of blood to the heart. Due to the lack of blood, the heart cells are not getting enough oxygen and nutrients and starts to die.

The features that are listed below, contribute to the risk of having a heart attack. In this project, I will determine which factors have the highest correlation coefficient for heart disease, leading to an increase in the risk of a heart attack. The target feature is “Diagnosis” (attribute 14 – In the list below) which was used for data visualisation and supervised training.

Dataset

The dataset, that was chosen for this assignment, is about heart diseases. The data was collected from the Cleveland Clinic Foundation and has been used by ML researchers. They have conducted experiments, focusing on distinguishing the presence of heart disease. The database has 76 attributes but only 14 of them are used in the dataset. Listed below is a brief description of the attributes:

1) Age – In years
2) Sex
    - 0 = Female
    - 1 = Male
3) Chest Pain Type
    - 1 = Typical angina
    - 2 = Atypical angina
    - 3 = Non-anginal pain
    - 4 = Asymptomatic
4) Resting Blood Pressure – In mm Hg on admission to the hospital
5) Serum Cholesterol – In mg/dl
6) Fasting Blood Sugar > 120mg/dl
    - 0 = False
    - 1 = True
7) Resting Electrocardiographic (ECG) Results
    - 0 = Normal
    - 1 = Having ST-T wave abnormality (T wave inversions and/or ST elevation or depression of > 0.05 mV)
    - 2 = Showing probable or definite left ventricular hypertrophy by Estes’ criteria

8) Maximum Heart Rate Achieved
9) Exercise Induced Angina
    - 0 = No
    - 1 = Yes
10) ST Depression – Induced by exercise relative to rest
11) Slope of The Peak – Exercise ST segment
    - 1 = Upsloping (Normal finding during exercise)
    - 2 = Flat (Very typical of Ischemia)
    - 3 = Downsloping (Typical of Ischemia)
12) Number of Major Vessels Blocked – (0-3) coloured by fluoroscopy
13) Thallium Test
    - 3 = Normal
    - 6 = Fixed Defect
    - 7 = Reversable Defect
14) Diagnosis of heart disease – Angiographic Disease Status **(Predicted Attribute)**
    - 0 = <50% diameter narrowing (heart disease is not present).
    - 1 = >50% diameter narrowing (heart disease is present)

Note that, in the dataset, the values from the predicted attribute (#14) have three extra values: 2, 3 and 4. These values are represented as value 1.

- 0 – Absence of heart disease.
- 1 (1-4) – Presence of heart disease.

Machine Learning Model

Approach

PyCharm was the chosen IDE that I had used for most of my coding. I had Jupyter for data visualisations. I had decided that I was going to use a Supervised Learning technique in order to train the machine learning model. But before, the model was created, the data had to be prepared. The dataset was retrieved, and the labels were appended before it was exported to an Excel file. The data was cleansed and scaled before Exploratory Data Analysis (EDA) was performed. At that point, I had a good idea of which features were most relevant and started on developing the Machine Learning Model. The dataframe was split into training and test dataframes and they were trained, using the Logistic Regression technique. Finally, the model was evaluated by producing accuracy values for the predictions made.

Preprocessing

After investigating the dataset, missing values were found which were encoded as '?'. The program went through the values and removed rows which contained missing values, as shown in figure 1. Then the index was reset.

```
def PreprocessData(dfHeartDiseases):
    # Remove missing values encoded as '?'
    dfHeartDiseases = dfHeartDiseases[(dfHeartDiseases.astype(str) != '?').all(axis=1)]
    # Reset index
    dfHeartDiseases = dfHeartDiseases.reset_index(drop=True)
```

Figure 1: Removal of missing values

As shown in figure 2, there are quite a few non numerical values present in the dataset. So, they were converted to the numerical format.

```
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   Age                    303 non-null    float64
 1   Sex                    303 non-null    float64
 2   Chest Pain Type        303 non-null    float64
 3   Resting Blood Pressure 303 non-null    float64
 4   Cholesterol            303 non-null    float64
 5   Fasting Blood Sugar    303 non-null    float64
 6   ECG                    303 non-null    float64
 7   Max Heart Rate         303 non-null    float64
 8   Exercise Angina        303 non-null    float64
 9   ST Depression          303 non-null    float64
 10  Slope                  303 non-null    float64
 11  Major Vessels Blocked  303 non-null    object
 12  Thallium Test          303 non-null    object
 13  Diagnosis              303 non-null    int64
dtypes: float64(11), int64(1), object(2)
memory usage: 33.3+ KB
```

Figure 2: Data Types

```
for oColumn in dfHeartDiseases:
    # Change non numerical values to numerical
    dfHeartDiseases[oColumn] = pd.to_numeric(dfHeartDiseases[oColumn])
```

Figure 3: Conversion of non-numerical values

The IQR method was used to remove outliers. First, the 25th (Q1) and 75th (Q3) percentiles were stored into variables and the IQR was calculated from subtracting Q1 from Q3. Boolean values were then produced by determining whether data values are within the range (IQR). If they were out of range, the values were classed as outliers and rows were deleted.

```python
def RemoveOutliers(dfHeartDiseases):
    # Export statistics of the dataframe to an Excel file
    dfHeartDiseases.describe().to_excel('Statistics.xlsx')
    # Print shape value to the console
    print("\nCHECK || RemoveOutliers() || Shape BEFORE: " + str(dfHeartDiseases.shape))

    # Retrieve Q1 and Q3
    fQ1 = dfHeartDiseases.quantile(0.25)
    fQ3 = dfHeartDiseases.quantile(0.75)
    # Calculate IQR
    fIQR = fQ3 - fQ1
    # Print IQR to console
    print("\nCHECK || RemoveOutliers() || IQR:\n" + str(fIQR))

    # Detect outliers by storing a boolean, produced by dfHeartDiseases >= (fQ3 + 1.5 * fIQR, into a dataframe
    dfUpperBound = dfHeartDiseases >= (fQ3 + 1.5 * fIQR)
    dfLowerBound = dfHeartDiseases <= (fQ1 - 1.5 * fIQR)
    # Print bound dataframes to console
    print("\nCHECK || RemoveOutliers() || \nUpper Bound:\n" + str(dfUpperBound) + "\nLower Bound:\n" + str(dfLowerBound))
    # Export bound dataframes to an Excel file
    dfUpperBound.to_excel('UpperBoundOutliers.xlsx')
    dfLowerBound.to_excel('LowerBoundOutliers.xlsx')

    # Remove outliers by row, using the IQR score
    dfHeartDiseasesClean = dfHeartDiseases[~((dfHeartDiseases > (fQ3 + 1.5 * fIQR)) | (dfHeartDiseases < (fQ1 - 1.5 * fIQR))).any(axis=1)]
    # Print shape of the dataframe to console
    print("\nCHECK || RemoveOutliers() || Shape AFTER: " + str(dfHeartDiseasesClean.shape))

    return dfHeartDiseasesClean
```

Figure 4: Removal of outliers

I had noticed the scale of the values in some columns were quite big, compared to the rest. So, standardisation was applied to some columns, centring the values around the mean with a unit of standard deviation.

```python
# Scale the data
oStandardScaler = StandardScaler()
columns_to_scale = ['Age', 'Resting Blood Pressure', 'Cholesterol', 'Max Heart Rate', 'ST Depression']
dfHeartDiseasesClean[columns_to_scale] = oStandardScaler.fit_transform(dfHeartDiseasesClean[columns_to_scale])
# Write to an Excel file
dfHeartDiseasesClean.to_excel('HeartDiseasesDatasetScale.xlsx')
```

Figure 5: Scaling the data

Data Visualisation and Feature Selection

In Jupyter, Exploratory Data Analysis (EDA) was performed to produce the visualisations of the data. The relationship between all the features and the target feature (Diagnosis) were analysed. The analysis are in the Jupyter notebook under each graph.

I did not need to implement a feature selection method as the provided dataset had all the relevant features.

Model Training and Evaluation

Model Training

The target feature (Diagnosis) was transferred to another dataframe. At this point there is a dataframe containing only the target feature, and another dataframe containing the rest of the features. Both dataframes were split into training and test datasets (75% training data and 25% test data) and Logistic Regression algorithm was applied to create the Machine Learning Model. The model was then trained with the training datasets.

```python
def SupervisedLearning(dfHeartDiseasesPrepared):
    # Store the target feature column into another dataframe
    dfTargetFeature = dfHeartDiseasesPrepared["Diagnosis"]
    # Remove the target feature column from the dataframe
    dfFeatures = dfHeartDiseasesPrepared.drop(["Diagnosis"], axis=1)
    # Export both dataframes to Excel files
    dfTargetFeature.to_excel('TargetFeature.xlsx')
    dfFeatures.to_excel('Features.xlsx')

    # Split dataset into training and testing datasets at 75% training data and 25% test data
    dfFeaturesTrain, dfFeaturesTest, dfTargetFeatureTrain, dfTargetFeatureTest = train_test_split(dfFeatures, dfTargetFeature, test_size=0.25, random_state=0)
    # Print on console
    print("Features Train: " + str(dfFeaturesTrain.shape))
    print("Features Test: " + str(dfFeaturesTest.shape))
    print("Target Feature Train: " + str(dfTargetFeatureTrain.shape))
    print("Target Feature Test: " + str(dfTargetFeatureTest.shape))

    # Model Training
    oModel = LogisticRegression()
    # Train the LogisticRegression Model with Training Data
    oModel.fit(dfFeaturesTrain.values, dfTargetFeatureTrain.values)
```

Figure 6: Supervised Learning Function

Model Evaluation

Model Evaluation was then conducted for both training and test datasets. A prediction value was produced from the Features training dataset and compared the value with the target feature training dataset to produce an accuracy value.

```python
# Model Evaluation - Training Data
dfFeaturesTrainPrediction = oModel.predict(dfFeaturesTrain.values)
fTrainingDataAccuracy = accuracy_score(dfFeaturesTrainPrediction, dfTargetFeatureTrain.values)
print("Training Data Accuracy: " + str(fTrainingDataAccuracy))

# Model Evaluation - Test Data
dfFeaturesTestPrediction = oModel.predict(dfFeaturesTest.values)
fTestDataAccuracy = accuracy_score(dfFeaturesTestPrediction, dfTargetFeatureTest.values)
print("Test Data Accuracy: " + str(fTestDataAccuracy))
```

Figure 7: Model Evaluation

Results and Discussion

As shown in Figure 8, a predictive system was implemented to produce results from a given input data. If the predictive value is 0, that indicates the person does not have a high risk of getting a heart attack as they do not have a heart disease. Otherwise, vice versa is applied.

```python
# Building a Predictive System
# Input data
tpInputData = (58.0,1.0,2.0,120.0,284.0,0.0,2.0,160.0,0.0,1.8,2.0,0.0,3.0)
# Change input data tuple to np array
arInputData = np.asarray(tpInputData)
# Reshape the np array to predict for only 1 instance
arInputDataReshape = arInputData.reshape(1, -1)
lPrediction = oModel.predict(arInputDataReshape)
# Print to console
print(lPrediction)

if lPrediction[0] == 0:
    print("The person does not have a high risk of getting a heart attack")
else:
    print("The person does have a high risk of getting a heart attack")
```

Figure 8: Predictive System

There have been some results produced, where the person has a heart disease does not have a high risk of getting a heart attack.

Conclusion

The aim of this project was to provide a heart attack prediction model for patients with accuracy based on the heart database that was downloaded from the Cleveland site. The data was cleansed by deleting rows containing missing values, and the Logistic Regression technique was used that resulted in a good accuracy. Patients at the risk category

for heart attacks could be recognised with this method which could help lower the rising death rate. This dataset was small, 303 patients. A larger dataset would have given more reliable results. Additional features like Height, Weight, BMI would have helped to detect additional risk factors for heart diseases.

Recommendations/Future Work
To use more classification techniques to get a better comparison by using different algorithms, like Bayes Net, Naïve Bayes, Random Forest, Neural networks and decision trees to predict the heart disease with accuracy. This can be then displayed visually in a bar chart to find the most accurate one.

Use a bigger heart dataset in order to predict with more accuracy

Instead of deleting the missing data values in the dataset, use of an algorithm to fill inexistent values based on the mean/median values

Have a dataset that includes other disorders like diabetes that are clinically related to heart disease

Develop an effective prediction system by using AI to integrating other conditions like environment.

A web app can be developed in the future that is well presented and integrates these methods which will make it easier to predict and treat cardiac abnormalities.

Apply Machine learning approach to alternative datasets of other illnesses like cancer

References

UCI. *Heart Disease Data Set*. https://archive.ics.uci.edu/ml/datasets/Heart+Disease

American Heart Association. *How High Blood Pressure Can Lead to a Heart Attack*. https://www.heart.org/en/health-topics/high-blood-pressure/health-threats-from-high-blood-pressure/how-high-blood-pressure-can-lead-to-a-heart-attack#:~:text=Damage%20can%20build%20over%20time,slow%20process%20is%20called%20atherosclerosis.

NHS. *Heart Attack*. https://www.nhs.uk/conditions/heart-attack/#:~:text=A%20heart%20attack%20(myocardial%20infarction,you%20suspect%20a%20heart%20attack.

Medical News Today. *How to spot and treat a heart attack*. https://www.medicalnewstoday.com/articles/151444

Brownlee, J. (2016, March 16). *Supervised and Unsupervised Machine Learning Algorithms*. https://machinelearningmastery.com/supervised-and-unsupervised-machine-learning-algorithms/

Aniruddha. (2020, April 3). *Feature Scaling for Machine Learning: Understanding the Difference Between Normalization vs. Standardization*. https://www.analyticsvidhya.com/blog/2020/04/feature-scaling-machine-learning-normalization-standardization/