|  |
| --- |
| **PROJECT TITLE:**  **PREDICTING CARDIOVASCULAR DISEASES: A BIG DATA APPROACH**  **BIG DATA ANALYTICS AND DATA VISUALIZATION 7153CEM**  **ENOMFON A. OBOT – 14243972**  **FACULTY OF ENGINEERING, ENVIRONMENT AND COMPUTING, COVENTRY UNIVERSITY** |

**TABLE OF CONTENTS**

Abstract………………………………………………………………………………… 2

Introduction……………………………………………………………………………. 2

Background/Related Work……………………………………………………………… 3

Dataset Description………………………………………………………………….….. 4

Methodology……………………………………………………………………………. 6

Experimental Section…………………………………………………..………….……. 8

Tableau Visualization……………………………………………………………………9

Result Discussion……………………………………………………………………….10

Conclusion………………………………………………………………………………10

Future Works…………………………………………………………………………....11

Social and Ethical Impact ………………………………………………………………11

References ………………………………………………………………………………12

Appendix - A……………………………………………………..……….……………..14

Appendix - B………………………………………………………………………….…24

Appendix - C……………………………………………………………………….……25

Appendix - D………………………………………………………………………….…26

1. **ABSTRACT**

This study is geared toward predicting heart failure disease using big data methodology. The data used was the result of the combination of different data. The data had been cleaned, as duplicates had been removed prior, exploratory data analysis was applied to the data.

While some studies showed that Random Forest (RF) and Linear Support Vector (SVM) are the recommended models for predicting heart disease, in this study we have compared both models stated above with other models such as Decision Tree (DT), Gradient Boost (GB) and Linear Regression (LR). The RF had the highest AUC score; however, the GB model had the highest score when all the evaluation metrics in this study used are averaged.

Keywords: Heart Failure dataset, Logistic Regression, Support Vector Machine, Decision Tree, Random Forest, Gradient Boost, Encoding, Vector Assembler, Cardiovascular Disease.

1. **INTRODUCTION**

On a worldwide scale, cardiovascular diseases (CVDs) result in 17.9 million deaths each year, making them the top cause of death. CVDs are a term that encompasses all illnesses related to the heart and blood vessels like rheumatic heart disease, coronary heart disease and cerebrovascular disease. Heart attacks and strokes make up more than four in every five deaths from CVD with one third of these occurring too soon for people aged less than 70 years old (World Health Organization: WHO, 2019).

In contrast, men are prone to experience coronary heart disease (CHD) and cardiovascular diseases (CVD) at an earlier age than women. But women tend to be more at risk of suffering a stroke which usually occurs later in life. (Bots et al., 2017)

Many variables fall into one of these two risk factor types, with different levels of impact on developing cardiovascular disease. Non-modifiable factors include things like age, gender, and hereditary characteristics. But, fasting blood sugar, high blood pressure, serum cholesterol along with smoking habit as well as dietary inclination towards items such as red meat and processed foodstuffs are placed together in the other category of concerns known as modifiable risk factors (Baghdadi et al., 2023).

The heart failure dataset, which is a combination of many datasets. For this project the goal of research work is to create a holistic strategy for predicting heart disease which includes making basic programming environments such as Spark, exploring and displaying data with Tableau, setting up training and reporting classification models. This study builds predictive models using machine learning classification algorithms to identify important signs of heart disease through deep investigation. The goal is to give important viewpoints about early spotting and dealing with cardiovascular disease.

1. **BACKGROUND/RELATED WORK**

Big Data's rise and revolution has led to the digitalization of healthcare data. The recent years have seen a tremendous increase in data, which has prompted the introduction of the big data domain. Within the field of information technology, massive amounts of data that are too large and challenging for typical databases to handle are referred to as "big data." Intelligent healthcare systems, which include big data analytics, provide innovative and mobile health while reducing medical expenses and increasing productivity (Awrahman et al., 2022).

People may stop the advancement of CVD by identifying changeable risk elements and trying to transform lifestyle-based danger factors into healthy ones. At first, those who suffer from heart disease might not show any symptoms until it becomes advanced sometimes leading to a situation where treatment or control becomes impossible (Baghdadi et al., 2023).

Research demonstrates the valuable predictive capabilities of machine learning methods, surpassing traditional statistical models in their ability to capture complex connections and nonlinear relationships between variables. According to the results of several studies, support vector machines (SVM) and random forests (RF) outperformed conventional models (Subramani et al., 2023).

A study involving over 378,000 UK patients found that machine-learning algorithms can significantly improve cardiovascular risk prediction accuracy. The study evaluated four algorithms: random forests, logistic regression, gradient boosting machines, and neural networks. Neural networks achieved the highest accuracy, and all machine-learning algorithms performed better than defined requirements. The study suggests that machine learning algorithms can identify patients who would benefit from cardiovascular disease prevention and prevent unnecessary medical interventions. (Weng et al., 2017).

Machine learning techniques were used to predict the risk of cardiovascular disease, using three classifiers compared to the HellenicSCORE risk assessment. The random forest classifier achieved the best outcomes, outperforming the risk tool. The study suggests that machine learning offers benefits over existing methods, such as access to larger datasets and the ability to identify hidden correlations between data (Dimopoulos et al., 2018).

* 1. **Limitations and Challenges:**

The rapid expansion of medical data across multiple domains has compelled computational specialists to devise methods for interpreting and analysing vast quantities of information. Data storage, search, capture, sharing, and analysis are some of the challenges associated with big data. In addition to these challenges are the difficulties encountered with real-time processing, data quality, privacy and security, and heterogeneous data. Maintaining high standards for medical data is also a prominent big data analytics challenge in healthcare systems. (Awrahman et al., 2022).

1. **DATASET DESCRIPTION**

Numerous elements in the dataset utilized for this investigation indicate risk factors that influence the development of cardiovascular disease. It was produced by merging many distinct, openly accessible datasets. This dataset is the largest heart disease dataset currently available for research purposes because it includes five distinct heart datasets that were pooled based on eleven similar traits. The following five distinct datasets were each used in its curation:

Cleveland: 303 observations

Hungarian: 294 observations

Switzerland: 123 observations

Long Beach VA: 200 observations

Stalog (Heart) Data Set: 270 observations

Total: 1190 observations

Duplicates: 272 observations

Final dataset: 918 observations, 11 features, 1 target

**Source of Dataset:** The dataset was obtained from Kaggle via the link <https://www.kaggle.com/datasets/fedesoriano/heart-failure-prediction/download?datasetVersionNumber=1>

The datatypes included in the dataset are:

|  |  |
| --- | --- |
| Integer | Age, Cholesterol |
| String | Sex, RestingECG |
| Double | Oldpeak |

* 1. **Data Pre-Processing and Analysis**

This phase is critical to ensure that necessary adjustments are made to the data before feeding the dataset into the machine learning model. If the data preprocessing step is skipped and there are missing values, the model would not be very functional, the results produced would also be affected if there are outliers present, is high dimensionality, and the data is noisy (García et al., 2014). Therefore, by preprocessing the data, the dataset would become complete and more accurate.

* 1. **Description of Dataset Attributes**

|  |  |  |
| --- | --- | --- |
| Feature | Explanation | Measurements |
| Age | The candidate’s age | Years |
| Sex | The candidate’s sex.  [F : female, and M : male] | Binary |
| ChestPainType | The candidate’s chest pain type.  [TA is typical angina, ATA is atypical angina, NAP is non-anginal pain,  ASY is Asymptomatic pain] |  |
| RestingBloodPressure | Normal blood pressure with no exercise | mmHg |
| Cholesterol | Serum cholesterol | mm/dl |
| FastingBloodsugar | Fasting blood sugar> 120 mg/dl (1= true;0 =false). It is blood sugar taken after a long gap between a meal and the test. Typically, it’s taken before any meal in the morning. | Boolean |
| RestingECG | Resting electrocardiogram results [Normal: Normal, ST: having ST-T wave abnormality (T wave inversions and/or ST elevation or depression of > 0.05 mV), LVH: showing probable or definite left ventricular hypertrophy by Estes' criteria] |  |
| MaximumHeartRate | Maximum heart rate achieved [Numeric value between 60 and 202] | bpm |
| ExerciseInduced Angina | exercise-induced angina [Y: Yes, N: No]. It is chest pain while exercising or doing any physical activity. | Boolean |
| Oldpeak | Oldpeak = ST. It is the difference between the value of ECG at rest and after exercise |  |
| ST\_Slope | the slope of the peak exercise ST segment [Up: upsloping, Flat: flat, Down: downsloping] |  |
| HeartDisease | Diagnosis of heart disease: output class [1: heart disease, 0: Normal] | Boolean |

* 1. **Machine Learning**

Machine Learning (ML) is defined as the area of Artificial Intelligence (AI) where a model learns from previous cases to determine future cases without exact programming. Several ML classifiers employ medical data for disease categorization and predicting. Biomarkers of heart disease termed features are learned by the machine in a supervised learning approach with the input data labelled for training purposes. The results obtained are also labelled after the input patient data is received (Pal et al., 2022).

1. **METHODOLOGY**

In carrying out this study, several ML techniques were used to model the data. The ML techniques include:

* 1. **Logistic Regression (LR)**

Utilizing the available historical data, logistic regression calculates the likelihood that an event will occur. Its objective is to determine the coefficient values for each input variable's weight. The logistic function, a nonlinear function, is used to alter the output prediction. When dealing with binary classification dataset, which include only two class values, this approach is advised (Le, 2024, A Guide to the Types of Machine Learning Algorithms, n.d.)(Figure 16).

* 1. **Gradient Boost (GB)**

With the help of multiple weak classifiers, boosting is an ensemble technique that aims to produce a strong classifier. To do this, a model is first constructed using the training set of data, and then an attempt is made to rectify the faults in the first model by developing a second model. Until the maximum number of models have been added, or the training set is properly predicted, new models are added (Le, 2024) (Figure 17).

* 1. **Linear Support Vector Classifier (LSVC)**

By class (either class 0 or class 1), a hyperplane is chosen in support vector machines (SVM) to best divide the points in the input variable space. In essence, SVM divides data into two groups. This is accomplished by giving training samples, each of which is designated as falling into one of the two categories. Using an algorithm, a model that gives new values to one or the other category is constructed (Le, 2024, A Guide to the Types of Machine Learning Algorithms, n.d.) (Figure 18).

* 1. **Random Forest (RDF)**

Known as "random decision forests," random forests are an ensemble learning technique that combine several algorithms to produce improved outcomes for tasks like regression and classification. An input is entered at the top of the "decision tree," which is the algorithm's initial representation of a tree-like graph or model of decisions. The data is then divided into smaller and smaller sets based on variables as it moves down the tree (A Guide to the Types of Machine Learning Algorithms, n.d.) (Figure 19).

* 1. **Decision Trees (DT)**

A decision tree is a tree structure that resembles a flow chart and employs branching to show all potential results. Every branch in the tree is the result of a test conducted on a particular variable, and each node in the tree represents a test on that variable. (A Guide to the Types of Machine Learning Algorithms, n.d.) (Figure 20).

Several evaluation metrics were also used to assess the results of the techniques used to model the data. These metrics include:

* 1. **Area under the curve (AUC)**

The possibility that a randomly selected positive sample would be ranked higher than a randomly selected negative sample is the classifier's area under the curve (AUC) (Mishra, 2021).

* 1. **Accuracy**

The ratio of the number of correct predictions to the total number of input samples gives the accuracy of the model (Rainio et al., 2024, Mishra, 2021).

It is calculated using.

Acc. = TP + TN

------------------------------ ∈ [0, 1]

TP + TN + FP + FN

* 1. **Precision**

A model's performance is measured by precision, which is the proportion of positive predictions the model makes that come true (GfG, 2023). Mathematically, it is:

Precision = TP

---------------------------

TP + FP

* 1. **Sensitivity**

Represented mathematically as:

Precision = TP

---------------------------

TP + FN

* 1. **F1-score**

F1 Score is the Harmonic Mean between precision and recall. It tells you how precise your classifier is (how many instances it classifies correctly), as well as how robust it is (it does not miss a significant number of instances). The greater the F1 Score, the better is the performance of our model (Rainio et al., 2024, Mishra, 2021).

It is represented mathematically as

F1 = 2 \* Precision \* Recall

---------------------------------------- ∈ [0, 1]

Precision + Recall

1. **EXPERIMENTAL SECTION**

Apache Spark python API -Pyspark was set up on the Ubuntu environment that had been installed prior, using the VMware application, please see figure 3. To analyse and experiment with the data, a version of Pyspark was installed on Jupyter notebook (Appendix A). Following which, all the requisite libraries were loaded, and a spark session created, please see Appendix A. Immediately the environment was fully setup and all required libraries had been loaded, the dataset, heart.csv, was loaded into the spark session and the first few rows shown (Figure 5). Figure 6 shows the dimensions of the dataset.

Data entries that are nil or have no assigned value are referred to as missing values. Using data with missing values can lead to skewed or incorrect results during modelling. To begin the data pre-processing stage, the dataset was first scanned for missing values (Figure 6). While no missing values were found in the dataset, about five categorical variable columns (Figure 7a and 7b) were found.

Using PySpark's “StringIndexer” the identified columns were encoded, with each individual column string value mapped to a unique numerical index based on their frequency (Jagdeesh, 2023) via a pipeline (Figure 8). A DataFrame was also created with the newly encoded columns added (Figure 10). To avoid obtaining skewed results and encountering difficulty modelling the data, the original columns were dropped leaving just the encoded columns.

The labelled column was dropped from the DataFrame and assigned to a different variable called Target; this assignment was done to view the distribution of the different classes in the target variable (Figure 11). While the two classes were not equal, the difference was negligible, hence, no class imbalance technique was applied to the dataset.

The Pyspark Vector Assembler was then used to transform the remaining columns in the dataset as well as the target variable column (Figure 12). The vector assembler converts data columns into vectors and further assembles them for further processing. Figure 13 shows the data being standardized to prevent the model(s) from producing skewed results because the data used is not scaled down to a measurement common to all columns. The target column was renamed label to allow for further processing (Figure 14).

Splitting the data into two parts allows for the model(s) to be trained and tested. The dataset used in this study was split into two parts with 80% of the data used for training the model and 20% used for testing the model. A seed of 42 was implemented to ensure that every time the data is run through the code, it is split the exact same way with the same seed count. This promotes consistency in the results being produced (Figure 15).

The data was modelled using five different ML techniques. These models include the Logistics Regression model, the Gradient Boost model, the Linear Support Vector Classifier model, the Random Forest model, and the Decision Tree model. The metrics used to analyse the performance of these models include Area Under the Curve (AUC-ROC), Accuracy, Precision, Sensitivity, and the F1-Score. However, for the purpose of this study, the AUC is focused on as the main performance evaluator of the models created (Figures 16, 17, 18, 19, 20).

1. **TABLEAU VISUALIZATION**

Tableau was installed on my computer to aid with visualization (Figure 4). Figures 21a and 21b show the percentage distribution of the dataset classes and furthermore the gender distribution within those classes. The visualization shows that there are more heart disease class observations than non-heart disease, there are also more male observations than female observations recorded for both classes in the dataset.

In figure 22, the image shows that men within the 54-57 age bracket are more like to suffer from heart disease while the age bracket for women is between 58-61. Therefore, men are more prone to heart disease at a younger age than women.

The visualization in figure 23 shows the relationship between heart disease and resting ECG. It suggests that the correlation between resting ECG and heart disease is not strong as most people who had heart disease also had normal resting ECG.

Figure 24 shows the relationship between heart disease and exercise induced angina. From the image, there is a strong relationship between these two variables as the larger number of men and women who experience exercise induced angina have heart disease and vice versa.

1. **RESULT DISCUSSION**

The results of this study have been summarized in the table below.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Results** | LR | GB | LSVC | RDF | DT |
| AUC | 0.91549 | 0.91440 | 0.91422 | 0.92325 | 0.87685 |
| Accuracy | 0.85234 | 0.88590 | 0.85906 | 0.85906 | 0.87248 |
| Precision | 0.86073 | 0.88886 | 0.86362 | 0.86024 | 0.87377 |
| Sensitivity | 0.85234 | 0.88590 | 0.85906 | 0.85906 | 0.87248 |
| F1 score | 0.85080 | 0.88539 | 0.85813 | 0.85867 | 0.87213 |

Logistic Regression: This model achieved an AUC score of 0.91 which is quite a high score and further confirms why LR is the recommended model for linear classification purposes.

Gradient Boost: With its ensemble-based way of modelling data, the GB model achieved an AUC score of 0.91 while not the highest, the GB model scored the highest in every other metric used to evaluate the model. Thus, it can be recommended as a good ML technique for modelling classification data.

Linear Support Vector Machine: Attaining an AUC score of 0.91 as well, the LSVM model has proven itself capable of modelling classification data accurately.

Random Forest: This model achieved the highest AUC score of 0.92. In RF, to improve the estimation of the true output value when a prediction needs to be made for fresh data, the average of the predictions from each model is used, making it an in-depth prediction model.

Decision Trees: Despite attaining the lowest AUC score, the DT model with a score 0.87 overall did well in predicting whether a subject has or does not have heart disease.

1. **CONCLUSION**

Based on the results, the Random Forest model with the highest AUC score is the preferred model for predicting heart failure disease based on the features used and preprocessing carried out. However, clinical adoption and interpretability should also be considered when adopting ML techniques. To obtain better accuracy the models could be combined to create an ensemble model, however, this could reduce the interpretability of the model due to the added complexity of the ensemble model.

The heart dataset was modelled with five different machine learning classifiers. With the highest AUC, the Random Forest model was the most accurate in predicting heart disease, followed closely by the Gradient Boost model. Both models seem to have done better by virtue of being ensemble models.

The Decision Tree model had the lowest AUC score of 0.87 which is still relatively good score for a model. However, other models seem to be better suited for modelling this data.

1. **FUTURE WORKS**

Obtaining more data especially standard health data will improve accuracy of research in this study area. Building predictive models based on ensemble techniques are also another way of improving the accuracy of results. Emphasis on building easily interpretable models would also aid in promoting clinical adoption of ML models to predict diseases.

1. **SOCIAL AND ETHICAL IMPACT**

There are a lot of ethical considerations to be made when handling medical data, hence, in utilizing ML to predict heart disease, all ethics governing the use of the data should be strictly adhered. CVD is the most common cause of death in women and men today, being able to improve the predictive ability of clinicians using ML techniques would aid in the reduction of the mortality rate of CVDs.

**REFERENCES**

World Health Organization: WHO. (2019, June 11). *Cardiovascular diseases*. https://www.who.int/health-topics/cardiovascular-diseases#tab=tab\_1

Bots, S. H., Peters, S. A., & Woodward, M. (2017). Sex differences in coronary heart disease and stroke mortality: a global assessment of the effect of ageing between 1980 and 2010. *BMJ Global Health*, *2*(2), e000298. https://doi.org/10.1136/bmjgh-2017-000298

Baghdadi, N. A., Abdelaliem, S. M. F., Malki, A., Gad, I., Ewis, A. A., & Atlam, E. (2023). Advanced machine learning techniques for cardiovascular disease early detection and diagnosis. *Journal of Big Data*, *10*(1). https://doi.org/10.1186/s40537-023-00817-1

Subramani, S., Varshney, N., Anand, M. V., Soudagar, M. E. M., Al-Keridis, L. A., Upadhyay, T. K., Alshammari, N., Saeed, M., Subramanian, K., Anbarasu, K., & Rohini, K. (2023). Cardiovascular diseases prediction by machine learning incorporation with deep learning. *Frontiers in Medicine*, *10*. https://doi.org/10.3389/fmed.2023.1150933

Awrahman, B. J., Fatah, C. A., & Hamaamin, M. Y. (2022). A review of the role and challenges of big data in healthcare informatics and analytics. *Computational Intelligence and Neuroscience (Print)*, *2022*, 1–10. <https://doi.org/10.1155/2022/5317760>

García, S., Luengo, J., & Herrera, F. (2014). Data preprocessing in data mining. Springer.

Pal, M., Parija, S., Panda, G., Dhama, K., & Mohapatra, R. K. (2022). Risk prediction of cardiovascular disease using machine learning classifiers. *Open Medicine*, *17*(1), 1100–1113. https://doi.org/10.1515/med-2022-0508

Weng, S., Reps, J., Kai, J., Garibaldi, J. M., & Qureshi, N. (2017). Can machine-learning improve cardiovascular risk prediction using routine clinical data? *PloS One*, *12*(4), e0174944. https://doi.org/10.1371/journal.pone.0174944

Dimopoulos, A. C., Νικολαϊδου, Μ., Caballero, F. F., Engchuan, W., Sánchez-Niubò, A., Arndt, H., Ayuso-Mateos, J. L., Haro, J. M., Chatterji, S., Georgousopoulou, E., Pitsavos, C., & Panagiotakos, D. B. (2018). Machine learning methodologies versus cardiovascular risk scores, in predicting disease risk. *BMC Medical Research Methodology (Online)*, *18*(1). https://doi.org/10.1186/s12874-018-0644-1

Le, J. (2024, March 5). *The top 10 machine learning algorithms to know*. Built In. https://builtin.com/data-science/tour-top-10-algorithms-machine-learning-newbies

*A guide to the types of machine learning algorithms*. (n.d.-a). SAS UK. https://www.sas.com/en\_gb/insights/articles/analytics/machine-learning-algorithms.html

Rainio, O., Teuho, J., & Клен, Р. (2024). Evaluation metrics and statistical tests for machine learning. *Scientific Reports (Nature Publishing Group)*, *14*(1). https://doi.org/10.1038/s41598-024-56706-x

Mishra, A. (2021, December 29). Metrics to Evaluate your Machine Learning Algorithm. *Medium*. https://towardsdatascience.com/metrics-to-evaluate-your-machine-learning-algorithm-f10ba6e38234

GfG. (2023, May 5). *Evaluation metrics in machine learning*. GeeksforGeeks. https://www.geeksforgeeks.org/metrics-for-machine-learning-model/

Jagdeesh. (2023, May 8). *PySpark StringIndexer – A Comprehensive Guide to master PySpark StringIndexer*. Machine Learning Plus. https://www.machinelearningplus.com/pyspark/pyspark-stringindexer/

**APPENDIX A**

A screenshot of a computer program

Description automatically generated

**Figure 1.** Hadoop Installation

A screenshot of a computer

Description automatically generated

**Figure 2.** Calling the HDFS

A screenshot of a computer program

Description automatically generated

**Figure 3.** Spark Installation

A screenshot of a computer

Description automatically generated

**Figure 4.** Tableau Installation

A screenshot of a computer

Description automatically generated

**Figure 5.** Loading the dataset

A screenshot of a computer code

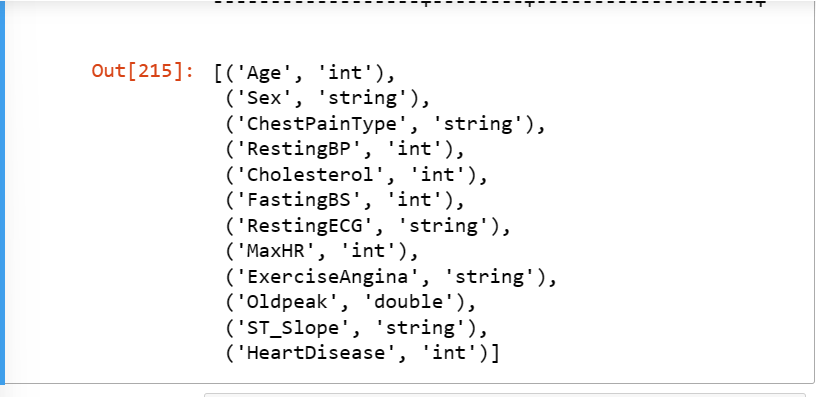
Description automatically generated

**Figure 6.** Checking the dimensionality of the dataset and for missing values

A screenshot of a computer code

Description automatically generated

**Figure 7a.** Checking the column datatypes.



**Figure 7b.** Output from checking the column datatypes.

A close-up of a computer screen

Description automatically generated

**Figure 8.** Encoding Categorical Variables

A screenshot of a computer program

Description automatically generated

**Figure 9.** New DataFrame with encoded columns added.

A screenshot of a computer program

Description automatically generated

**Figure 10.** Original columns dropped in favour of encoded columns.

A screen shot of a computer

Description automatically generated

**Figure 11.** Checking for class imbalance

A screenshot of a computer

Description automatically generated

**Figure 12.** Applying the VectorAssembler function to the dataset.

A screenshot of a computer

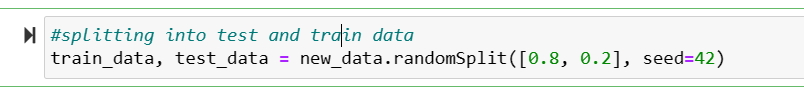
Description automatically generated

**Figure 13.** Standardizing the data

A screenshot of a computer

Description automatically generated

**Figure 14.** Renaming the Target column



**Figure 15.** Splitting the dataset.

A screenshot of a computer program

Description automatically generated

**Figure 16.** Logistics Regression model with AUC output

A screenshot of a computer program

Description automatically generated

**Figure 17.** Gradient Boost model with AUC output

A screenshot of a computer code

Description automatically generated

**Figure 18.** Linear Support Vector Classifier Model with AUC output

A screen shot of a computer code

Description automatically generated

**Figure 19.** Random Forest Model with AUC score

A screenshot of a computer code

Description automatically generated

**Figure 20.** Decision Tree Model with AUC score.

A screenshot of a graph

Description automatically generated A graph of a heart disease

Description automatically generated

**Figure 21a.** Data Distribution **Figure 21b.** Gender Distribution of dataset

A graph of age and age

Description automatically generated

**Figure 22.** Age distribution in the dataset

A graph of a number and number

Description automatically generated with medium confidence

**Figure 23.** Relationship between resting ECG and Heart Disease

A graph with numbers and a number

Description automatically generated

**Figure 23:** Relationship between heart disease and Exercise angina

**APPENDIX B**: Installing PySpark and calling the required libraries.

#to install pyspark

#pip install pyspark

# Import necessary libraries

**from** **pyspark.sql** **import** SparkSession

**from** **pyspark.ml.stat** **import** Correlation

**from** **pyspark.ml.feature** **import** VectorAssembler,StringIndexer,StandardScaler

**from** **pyspark.mllib.evaluation** **import** BinaryClassificationMetrics

**from** **pyspark.ml.evaluation** **import** MulticlassClassificationEvaluator

**from** **pyspark.ml.evaluation** **import** BinaryClassificationEvaluator

**from** **pyspark.ml.classification** **import** LogisticRegression

**from** **pyspark.ml.classification** **import** RandomForestClassifier

**from** **pyspark.ml.classification** **import** GBTClassifier

**from** **pyspark.ml.classification** **import** LinearSVC

**from** **pyspark.ml.classification** **import** DecisionTreeClassifier

# Create a SparkSession

spark = SparkSession.builder.appName("HeartFailurePredictor").master("local").getOrCreate()

**APPENDIX C**: Importing the Dataset and Data Preprocessing

# Load the dataset

data = spark.read.csv("C:/Users/eobot/OneDrive - Coventry University/Desktop/7153/Assessment/Coursework/heart.csv", header=True, inferSchema=True)

data.show(**10**)

**import** **pyspark.sql.functions** **as** **fc**

**print**((data.count(), len(data.columns)))

data.describe().show()

# check for null values in each column

data\_null = data.agg(\*[fc.count(fc.when(fc.isnull(c), c)).alias(c) **for** c **in** data.columns])

data\_null.show() # no null values

data.summary

data.describe().show()

data.dtypes

**from** **pyspark.ml** **import** Pipeline

**from** **pyspark.ml.feature** **import** StringIndexer

#label encoding of categorical columns

categorical\_cols = ["Sex","ChestPainType","RestingECG","ExerciseAngina","ST\_Slope"]

label\_encoders = [StringIndexer(inputCol=col, outputCol=col + "\_encoded").fit(data) **for** col **in** categorical\_cols]

pipeline = Pipeline(stages=label\_encoders)

data = pipeline.fit(data).transform(data)

#drop duplicate columns

new\_data = data.drop("Sex","ChestPainType","RestingECG","ExerciseAngina","ST\_Slope")

new\_data.dtypes

FeaturesData = new\_data.drop("HeartDisease")

**print**((FeaturesData.count(), len(FeaturesData.columns)))

Target = new\_data.drop("Age", "RestingBP","Cholesterol","FastingBS","MaxHR","Oldpeak","ExerciseAngina\_encoded","Sex\_encoded","ChestPainType\_encoded","RestingECG\_encoded","ST\_Slope\_encoded")

**print**((Target.count(), len(Target.columns)))

Target.groupBy('HeartDisease').count().orderBy('count').show()

features\_columns = FeaturesData.columns

**print**(features\_columns)

VAssembler = VectorAssembler(inputCols=features\_columns, outputCol="VAfeatures")

new\_data = VAssembler.transform(new\_data)

new\_data = new\_data.select("VAfeatures", "HeartDisease")

new\_data.show(**5**)

#standardizing the data

scaled\_data = StandardScaler(inputCol="VAfeatures", outputCol="features")

new\_data = scaled\_data.fit(new\_data).transform(new\_data)

new\_data.show(**5**)

new\_data = new\_data.select("features", "HeartDisease")

new\_data = new\_data.withColumnRenamed("HeartDisease","label")

new\_data.show(**5**)

#splitting into test and train data

train\_data, test\_data = new\_data.randomSplit([**0.8**, **0.2**], seed=**42**)

**APPENDIX D** – Classification Models

# Logistic Regression Model

LR=LogisticRegression().fit(train\_data)

#Get Predictions for Logistic Regression Model

prediction = LR.transform(test\_data)

multi\_evaluator = MulticlassClassificationEvaluator(labelCol="label", predictionCol="prediction")

evaluator = BinaryClassificationEvaluator(rawPredictionCol="rawPrediction", labelCol="label")

#Metrics for evaluation

auc = evaluator.evaluate(prediction)

accuracy = multi\_evaluator.evaluate(prediction, {multi\_evaluator.metricName: "accuracy"})

precision = multi\_evaluator.evaluate(prediction, {multi\_evaluator.metricName: "weightedPrecision"})

sensitivity = multi\_evaluator.evaluate(prediction, {multi\_evaluator.metricName: "weightedRecall"})

f1score = multi\_evaluator.evaluate(prediction, {multi\_evaluator.metricName: "f1"})

**print**("AUC-ROC: ", auc)

**print**("Accuracy: ", accuracy)

**print**("Precision: ", precision)

**print**("Sensitivity: ", sensitivity)

**print**("F1-Score: ", f1score)

#Display the Logistic Regression predictions

prediction.show()

cnd;hd

# Gradient Boost Model

gradient\_boost\_class = GBTClassifier(labelCol="label", featuresCol="features")

model = gradient\_boost\_class.fit(train\_data)

#Get predictions for Gradient Boost model

predictionsGBT = model.transform(test\_data)

#Metrics for evaluation

auc = evaluator.evaluate(predictionsGBT)

accuracy = multi\_evaluator.evaluate(predictionsGBT, {multi\_evaluator.metricName: "accuracy"})

precision = multi\_evaluator.evaluate(predictionsGBT, {multi\_evaluator.metricName: "weightedPrecision"})

sensitivity = multi\_evaluator.evaluate(predictionsGBT, {multi\_evaluator.metricName: "weightedRecall"})

f1score = multi\_evaluator.evaluate(predictionsGBT, {multi\_evaluator.metricName: "f1"})

**print**("AUC-ROC: ", auc)

**print**("Accuracy: ", accuracy)

**print**("Precision: ", precision)

**print**("Sensitivity: ", sensitivity)

**print**("F1-Score: ", f1score)

#Display the Gradient Boost predictions

predictionsGBT.show()

#Linear Support Vector Model

lsvc = LinearSVC(labelCol="label", maxIter=**50**)

model = lsvc.fit(train\_data)

#Get predictions for Support Vector Machine Classifier

predictionsLSVC = model.transform(test\_data)

#Metrics for evaluation

auc = evaluator.evaluate(predictionsLSVC)

accuracy = multi\_evaluator.evaluate(predictionsLSVC, {multi\_evaluator.metricName: "accuracy"})

precision = multi\_evaluator.evaluate(predictionsLSVC, {multi\_evaluator.metricName: "weightedPrecision"})

sensitivity = multi\_evaluator.evaluate(predictionsLSVC, {multi\_evaluator.metricName: "weightedRecall"})

f1score = multi\_evaluator.evaluate(predictionsLSVC, {multi\_evaluator.metricName: "f1"})

**print**("AUC-ROC: ", auc)

**print**("Accuracy: ", accuracy)

**print**("Precision: ", precision)

**print**("Sensitivity: ", sensitivity)

**print**("F1-Score: ", f1score)

#Display the Linear Support Vector Classifier predictions

predictionsLSVC.show()

#Random Forest Model

random\_forest = RandomForestClassifier(labelCol="label", featuresCol="features")

model = random\_forest.fit(train\_data)

#Get predictions for Random Forest model

predictionsRDF = model.transform(test\_data)

#Metrics for evaluation

auc = evaluator.evaluate(predictionsRDF)

accuracy = multi\_evaluator.evaluate(predictionsRDF, {multi\_evaluator.metricName: "accuracy"})

precision = multi\_evaluator.evaluate(predictionsRDF, {multi\_evaluator.metricName: "weightedPrecision"})

sensitivity = multi\_evaluator.evaluate(predictionsRDF, {multi\_evaluator.metricName: "weightedRecall"})

f1score = multi\_evaluator.evaluate(predictionsRDF, {multi\_evaluator.metricName: "f1"})

**print**("AUC-ROC: ", auc)

**print**("Accuracy: ", accuracy)

**print**("Precision: ", precision)

**print**("Sensitivity: ", sensitivity)

**print**("F1-Score: ", f1score)

#Display the Random Forest predictions

predictionsRDF.show()

#Decision Tree Model

dt = DecisionTreeClassifier(labelCol="label", featuresCol="features", maxDepth = **7**)

model = dt.fit(train\_data)

#Get predictions for Decision Tree model

predictionsDT = model.transform(test\_data)

#Metrics for evaluation

auc = evaluator.evaluate(predictionsDT)

accuracy = multi\_evaluator.evaluate(predictionsDT, {multi\_evaluator.metricName: "accuracy"})

precision = multi\_evaluator.evaluate(predictionsDT, {multi\_evaluator.metricName: "weightedPrecision"})

sensitivity = multi\_evaluator.evaluate(predictionsDT, {multi\_evaluator.metricName: "weightedRecall"})

f1score = multi\_evaluator.evaluate(predictionsDT, {multi\_evaluator.metricName: "f1"})

**print**("AUC-ROC: ", auc)

**print**("Accuracy: ", accuracy)

**print**("Precision: ", precision)

**print**("Sensitivity: ", sensitivity)

**print**("F1-Score: ", f1score)

#Display the Decision Tree predictions

predictionsDT.show()

A