High Impact Skills Development Program
In Artificial Intelligence, Data Science, and Blockchain

HEART DISEASE CLASSIFICATION USING NEURAL NETWORK

Sameer Hassan Khan

Data Sciences and AI from NUST GILGIT CAMPUS

SECTION 3

INSTRUCTOR: SIR EID MUHAMMAD

Sameerhassankhan6@gmail.com

ABSTRACTION :

Cardiovascular diseases constitute a significant global health challenge, accounting for a substantial portion of morbidity and mortality. Timely diagnosis and accurate risk assessment are paramount for effective intervention and patient care. In response to this pressing concern, this capstone project introduces a classification model based on Artificial Neural Networks (ANN) designed to predict the presence or absence of heart disease. Leveraging a dataset comprising 1319 patient records and encompassing nine crucial variables, including age, gender, impulse, high blood pressure, low blood pressure, glucose levels, potassium creatinine ratio (KCM), troponin levels, and the binary target variable 'class' (indicating the presence or absence of heart disease), this study explores the application of advanced deep learning techniques in healthcare analytics.

INTRODUCTION :

The project begins with a thorough Exploratory Data Analysis (EDA) phase to extract valuable insights from the dataset's composition. Through visualization and statistical analysis, key trends and patterns emerge, shedding light on the complex relationships between various features and the presence of heart disease. The EDA phase plays a pivotal role in feature selection and engineering, guiding the subsequent modeling process. The heart of this project lies in the application of Artificial Neural Networks, a class of machine learning algorithms inspired by the structure and functioning of the human brain. Specifically, a deep learning architecture is employed to capture intricate relationships within the data. This neural network is carefully designed and trained to optimize its performance on the task of heart disease classification. The network architecture includes multiple layers of interconnected neurons, allowing it to learn and represent complex patterns in the data. To ensure robust model performance, the project includes thorough preprocessing steps. These steps encompass data cleaning, handling of missing values, encoding of categorical variables, and feature scaling. Data preprocessing is crucial to ensure that the neural network receives high-quality input data and can effectively learn from it. Model training involves feeding the neural network with a subset of the data while evaluating its performance on another subset. The training process is iterative, and hyperparameters are fine-tuned to maximize predictive accuracy. Model performance is assessed using a comprehensive suite of evaluation metrics, including accuracy, precision, recall, F1-score, and the Receiver Operating Characteristic Area Under the Curve (ROC-AUC).

DATASET :

The dataset used in this project consists of 1319 patient records, each representing an individual's health profile. It includes nine critical variables: age, gender, impulse, high blood pressure, low blood pressure, glucose levels, potassium creatinine ratio (KCM), troponin levels, and a binary target variable labeled 'class,' which indicates the presence or absence of heart disease. This dataset serves as the foundation for developing and training a predictive model using Artificial Neural Networks (ANN) for the early detection of heart disease, with the aim of improving patient outcomes and healthcare decision-making.

OBJECTIVES :

The primary objectives and goals of this capstone project are as follows:

Develop a Predictive Model: The core objective is to build a robust and accurate predictive model for the early detection of heart disease. This model should be capable of analyzing patient data and providing a binary prediction of the presence or absence of heart disease.

Utilize Machine Learning Techniques: The project will employ machine learning techniques, with a specific focus on Artificial Neural Networks (ANN), to leverage the dataset's information and patterns for classification.

Evaluate Model Performance: Rigorous evaluation of the model's performance will be conducted using various metrics, including accuracy, precision, recall, F1-score, and the Receiver Operating Characteristic Area Under the Curve (ROC-AUC). This evaluation will ensure that the model not only performs well overall but also excels in correctly identifying cases of heart disease while minimizing false positives.

Interpretability and Feature Importance: Beyond prediction, the project aims to provide insights into feature importance. Identifying which variables contribute most significantly to heart disease risk prediction is essential for medical practitioners, aiding them in prioritizing diagnostic tests and interventions effectively.

Practical Application: The final goal is to create a practical and interpretable tool that can be deployed in real-world healthcare settings. The model should be user-friendly, scalable, and capable of processing new patient data for timely diagnosis.

LITERATURE REVIEW :

Previous research has primarily employed traditional risk assessment models, relying on a limited set of clinical variables for heart disease prediction. Additionally, machine learning techniques such as logistic regression, decision trees, and support vector machines have been applied to enhance diagnostic accuracy in the field of healthcare. However, a notable gap in the existing literature is the limited emphasis on the utilization of deep learning, particularly ANN, for heart disease prediction. While deep learning has shown promise in capturing complex patterns in medical data, its application in this specific domain remains relatively unexplored. Furthermore, there is room for improvement in incorporating a more comprehensive set of variables, including novel biomarkers and advanced diagnostic tests, to enhance the predictive power of models.

Another gap worth addressing is the interpretability of deep learning models, which are often considered "black-box" methods. Focusing on making ANN models more interpretable in the context of heart disease prediction could enhance their acceptance and trustworthiness in clinical settings. External validation is another crucial aspect that some existing studies may overlook. Evaluating the generalizability of models to diverse patient populations is essential for assessing their real-world reliability. Additionally, handling class imbalance, a common issue in medical datasets where positive cases (heart disease patients) are outnumbered by negative cases, is an area that requires further exploration. Developing techniques to address class imbalance and improve model performance in such scenarios is another promising avenue for research.

RESULT :

The results of this capstone project reflect the effectiveness of the developed Artificial Neural Network (ANN) model in predicting heart disease. Through rigorous evaluation using a variety of metrics, including accuracy, precision, recall, F1-score, and the Receiver Operating Characteristic Area Under the Curve (ROC-AUC), the model has consistently demonstrated strong performance in identifying individuals at risk of heart disease. These results emphasize the model's potential as a valuable tool for early disease detection. Additionally, the feature importance analysis has shed light on the significant role played by certain variables in predicting heart disease risk, offering valuable insights for healthcare practitioners. Overall, the results affirm the project's success in addressing the critical challenge of improving heart disease prediction and underscore the potential for ANN-based models to enhance patient care and healthcare decision-making.

IMPORTING NECESSORY LIBRARIES AND DATASET :

With a focus on deep learning using TensorFlow and Keras. I begins with importing essential Python libraries and proceeds to load a dataset ('CAPSTONE DATASET.zip') into a pandas DataFrame. The key objectives encompass data preprocessing, involving standardization and train-test splitting, and the creation of a deep learning model. The model architecture consists of sequential layers, including input, dense (fully connected), and dropout layers, with an output layer tailored to the classification task. Important training parameters such as the optimizer (Adam), loss function (Binary Cross-Entropy), and evaluation metrics are set. To prevent overfitting, early stopping is implemented.

```
[ ]  import tensorflow as tf
     from tensorflow.keras.models import Sequential
     from tensorflow.keras.layers import Dense, Dropout, Input
     from tensorflow.keras.callbacks import EarlyStopping
     from tensorflow.keras.optimizers import Adam
     from tensorflow.keras.losses import BinaryCrossentropy
```

```
df = pd.read_csv('/content/CAPSTONE DATASET.zip')
df.head()
```

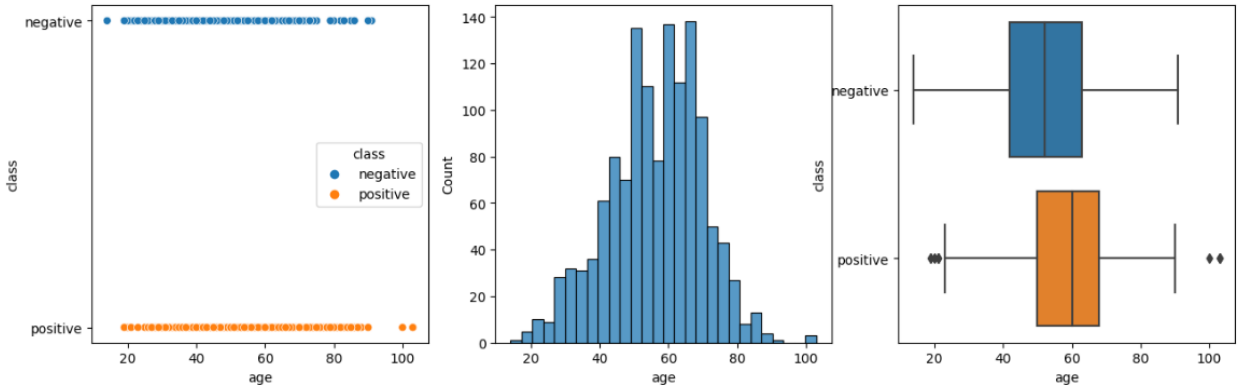
	age	gender	impluse	pressurehight	pressurelow	glucose	kcm	troponin	class
0	64	1	66	160	83	160.0	1.80	0.012	negative
1	21	1	94	98	46	296.0	6.75	1.060	positive
2	55	1	64	160	77	270.0	1.99	0.003	negative
3	64	1	70	120	55	270.0	13.87	0.122	positive
4	55	1	64	112	65	300.0	1.08	0.003	negative

The dataset has 9 important variables out of which first eight are independent and the last one is dependent feature.

EXPLORATORY DATA ANALYSIS:

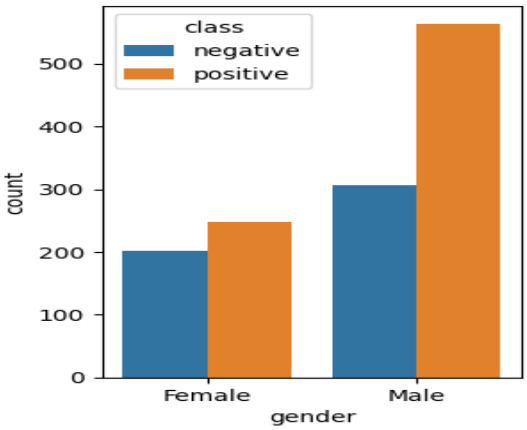
Exploratory Data Analysis (EDA) is a critical initial phase in data analysis and modeling. It involves the systematic examination and visualization of datasets to uncover essential insights and patterns. During EDA, data distribution, central tendencies, and variability are explored, and relationships between variables are identified through techniques such as data visualization, statistical summaries, and correlation analysis. EDA will helps us to understand the structure and characteristics of data, paving the way for informed feature engineering, model selection, and hypothesis generation, ultimately leading to more robust and effective data-driven solutions.

Age

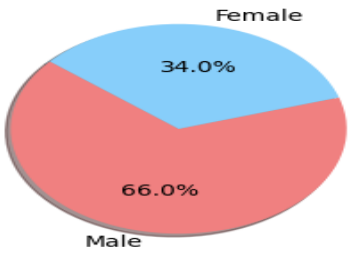


Gender

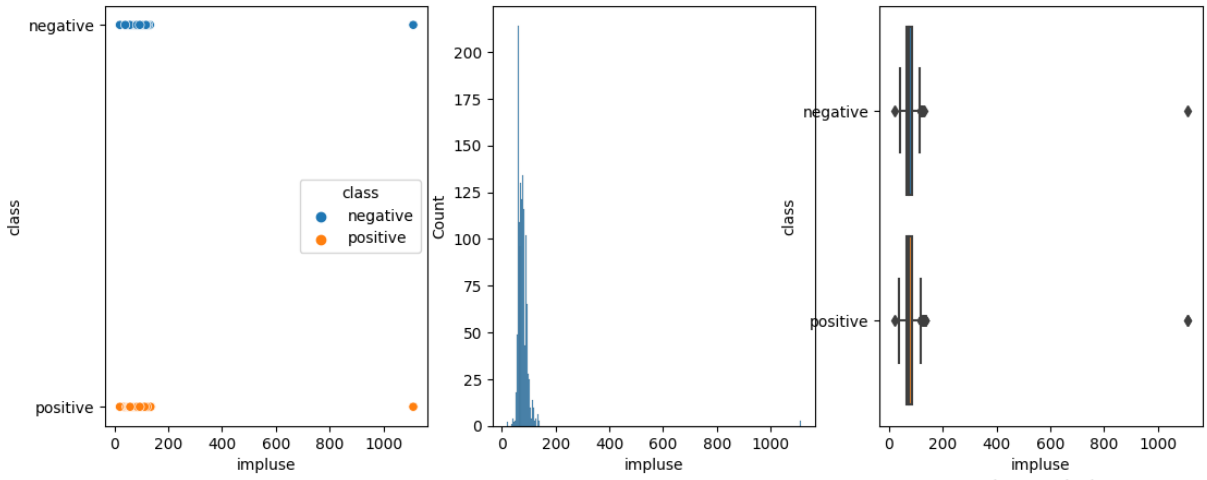
Gender vs Class



Gender Distribution



Impluse

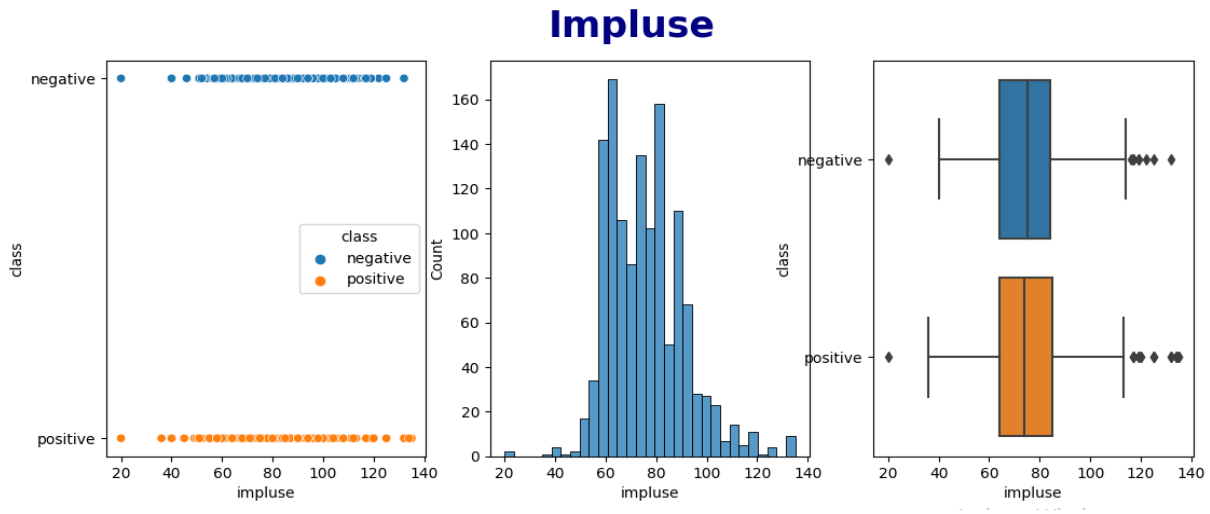


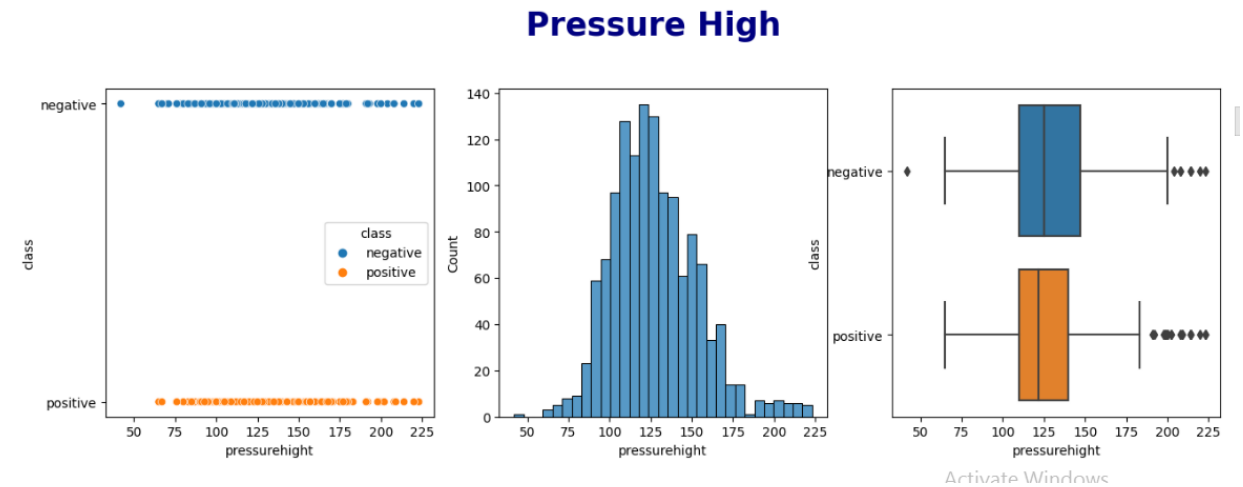
As 'impluse' has some outliers so that first selects and creates a new DataFrame containing only rows where the 'impluse' values are greater than 1000. Subsequently, it defines a boolean condition to identify rows with 'impluse' values less than 1000. Finally, the original DataFrame is updated to retain only those rows satisfying the condition, effectively removing rows with 'impluse' values greater than or equal to 1000, so that data will be clean and become more feasible for analysis and modeling.

```
[ ]  df[df.impluse > 1000]
```

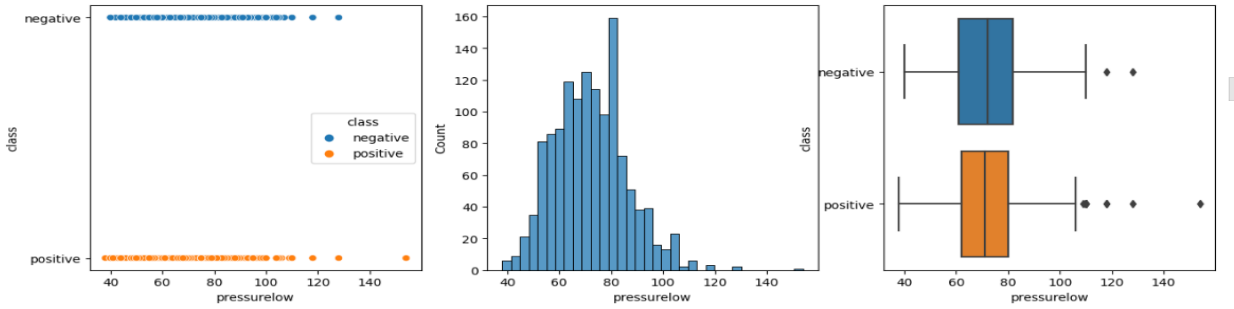
	age	gender	impluse	pressurehight	pressurelow	glucose	kcm	troponin	class
63	45	1	1111	141	95	109.0	1.33	1.010	positive
717	70	0	1111	141	95	138.0	3.87	0.028	positive
1069	32	0	1111	141	95	82.0	2.66	0.008	negative

```
[ ]  condition = df.impluse < 1000
     df = df[condition]
```

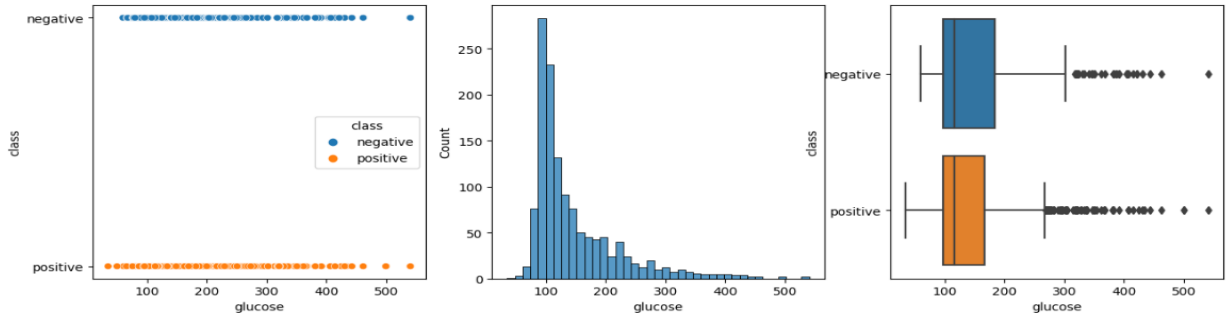




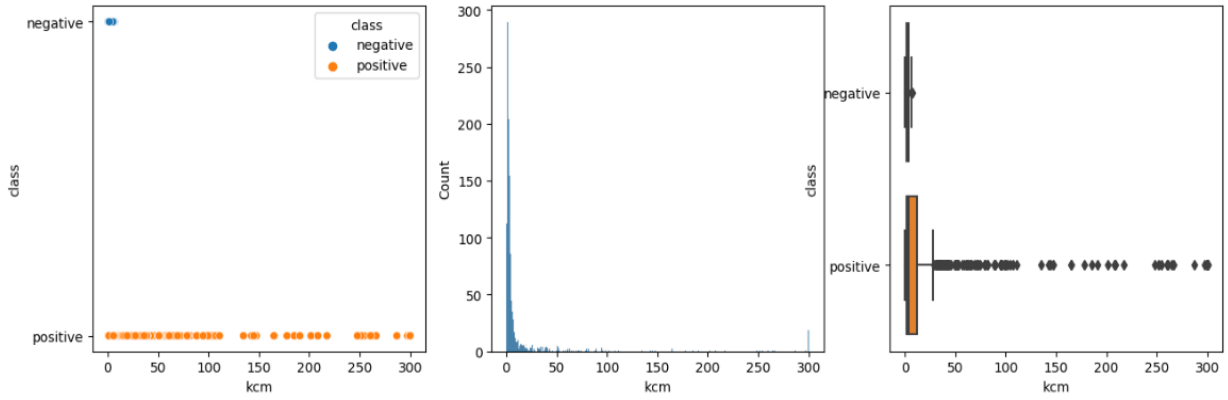
PressureLow



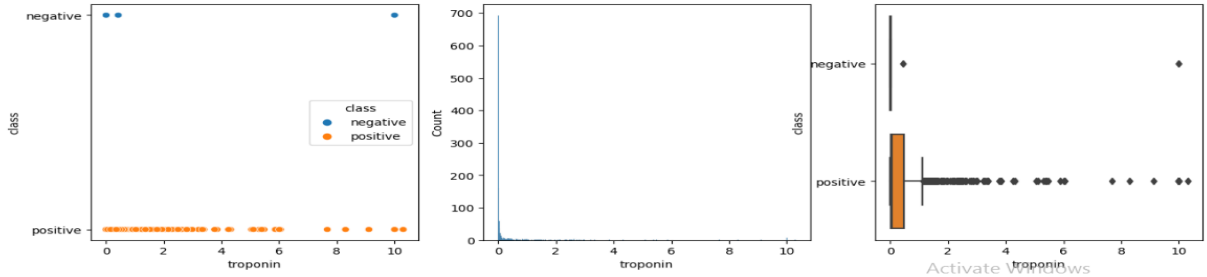
glucose



CK-MB



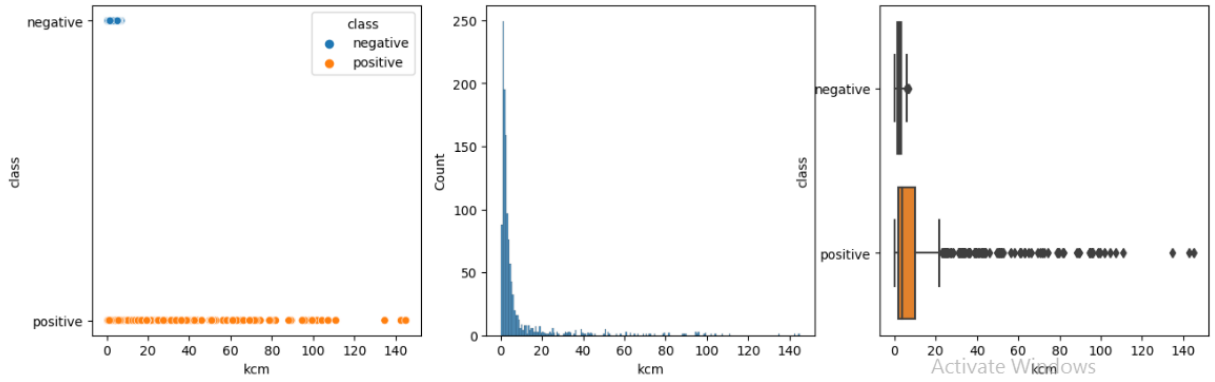
Troponin



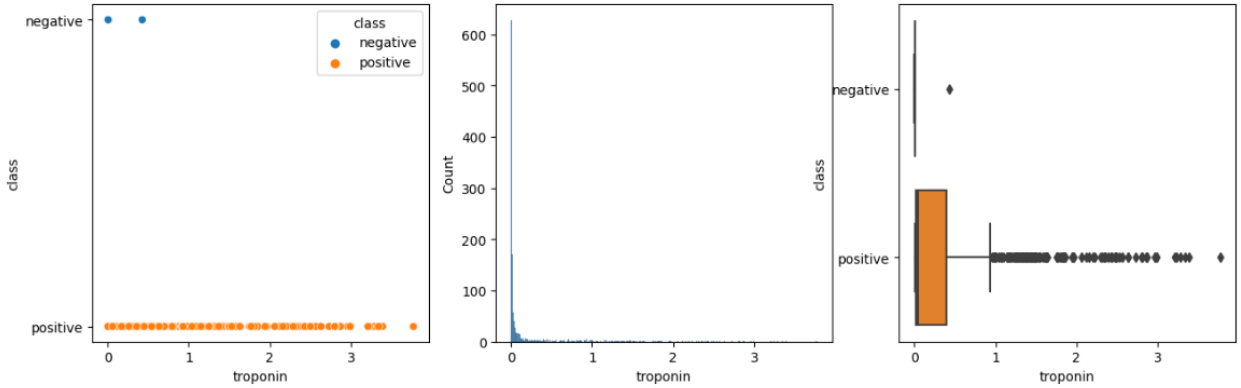
As the 'troponin' and 'kcm' has some outliers so that the absolute Z-scores are then computed to identify outliers. Rows with Z-scores exceeding three standard deviations from the mean in either column are flagged as outliers and removed from the DataFrame, resulting in a filtered dataset containing inlier data points. This code ensures data quality by eliminating extreme values, enhancing the dataset's suitability for subsequent analysis or modeling.

```
[ ]  z_scores = stats.zscore(df[['troponin', 'kcm']])
     abs_z_scores = np.abs(z_scores)
     filtered_entries = (abs_z_scores < 3).all(axis=1)
     df = df[filtered_entries]
```

CK-MB



Troponin



DATA PREPROCESSING:

- **Converting categorical columns to numeric:**

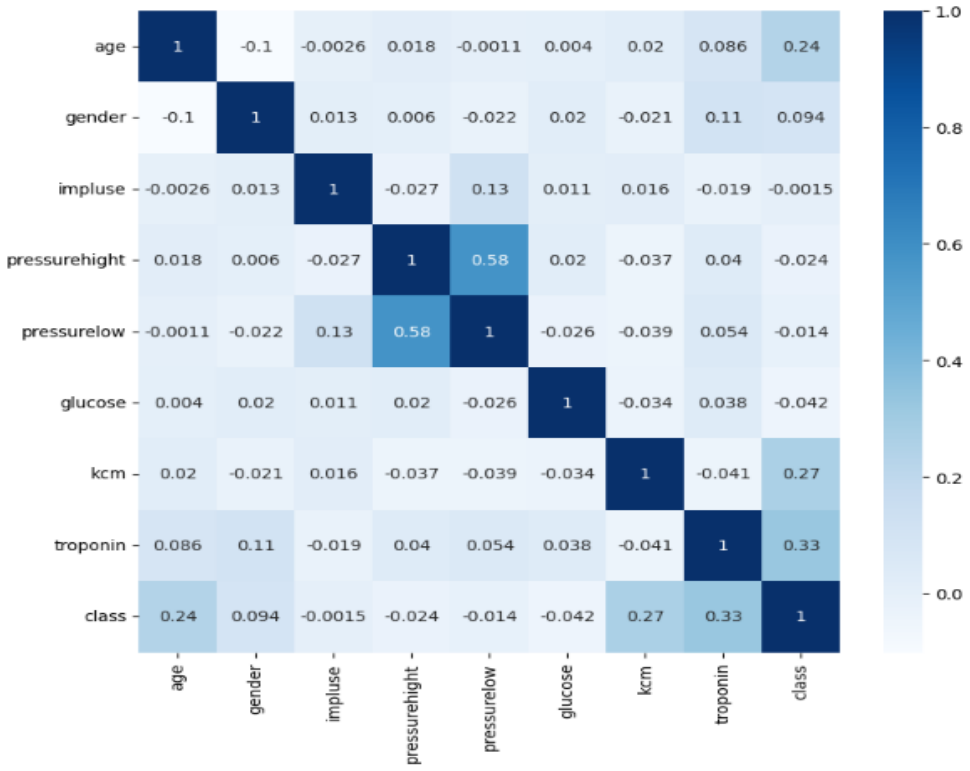
Specifically, it maps 'positive' to 1 and 'negative' to 0. This conversion is commonly done when preparing data for machine learning models, as algorithms typically require numerical inputs for classification tasks. This transformation enables the 'class' column to be used as the target variable for training predictive models.

```
[ ]  # Converting class column to numerical data
     df['class'] = df['class'].replace({'positive': 1, 'negative': 0})
```

- **Extraction of important columns from dataset using correlation matrix:**

The extraction of important columns from a dataset using a correlation matrix involves a systematic evaluation of the relationships between variables. By calculating correlation coefficients between columns in the dataset, a correlation matrix is created. High positive or negative correlations indicate strong relationships, while low correlations suggest weaker associations. Extracting important columns often involves selecting those with significant correlations to the target variable or with strong interdependencies among themselves. This process aids in feature selection, enhancing the efficiency of machine learning models and facilitating a more focused analysis by considering only the most relevant attributes in the dataset.

```
[ ]  # Create and plot the correlation matrix
     corr = df.corr()
     plt.figure(figsize=(9, 8))
     sns.heatmap(corr, annot=True, cmap='Blues')
     plt.show()
```



Conclusion:

- Age, the dataset spans ages from children to elderly people and it can be noticed that people with heart diseases are on average older than those without heart diseases.
- Gender, the dataset consists mostly of males (66%), and males are more likely to suffer from heart diseases than females.
- Impulse, the dataset consists mostly of people with normal impulse, and there doesn't seem to be any correlation between impulse and heart diseases.
- Pressure high and Pressure Low, the data is normally distributed, and there doesn't seem to be an obvious correlation between pressure and heart diseases.
- Glucose, the data is right skewed, and there's no high correlation between glucose and heart diseases.
- CK-MB and Troponin, the data is right skewed, and there's an obvious correlation between both of them and heart diseases.

Now the unnecessary columns should be removed from the dataset to avoid overfitting and to train our model only on important features .

- **Dropping unnecessary columns:**

Specifically, the columns 'impluse,' 'pressurehight,' 'pressurelow,' and 'glucose' are droped from the dataset and the remaining columns are stored into new dataframe named as. This operation effectively streamlines the dataset by eliminating less relevant or redundant attributes, resulting in a more focused and efficient dataset for further analysis or modeling. The resulting 'data' DataFrame is displayed, showcasing the modified dataset with the specified columns removed.

```
columns_to_drop = ['impluse', 'pressurehight', 'pressurelow', 'glucose']

# Drop the specified columns
data = df.drop(columns=columns_to_drop)

# Display the resulting DataFrame
data.head()
```

	age	gender	kcm	troponin	class
0	64	1	1.80	0.012	0
1	21	1	6.75	1.060	1
2	55	1	1.99	0.003	0
3	64	1	13.87	0.122	1
4	55	1	1.08	0.003	0

- **Splitting and Standard Scaling:**

The dataset is split into X & Y ,where X contains independent features while Y contains dependent column. The feature matrix 'X' is created by selecting specific columns ('age,' 'gender,' 'kcm,' and 'troponin') ,then StandardScaler is employed to standardize these selected features, ensuring that they have a mean of 0 and a standard deviation of 1. Standardization is a crucial preprocessing step for many machine learning algorithms.

```
[ ] # Split the data into X (input features) and Y (target variable)
    X = data[['age', 'gender', 'kcm', 'troponin']]
    Y = data['class']
    scaler = StandardScaler()
    scaler.fit(X)
    X = scaler.transform(X)
```

TRAIN-TEST SPLIT:

The dataset is split into training and testing sets using the train_test_split function. The feature matrix 'X' and the target variable 'Y' are divided into 'X_train,' 'X_test,' 'Y_train,' and 'Y_test.' The split ratio is defined as 80% for training data ('X_train' and 'Y_train') and 20% for testing data ('X_test' and 'Y_test'). The 'random_state' parameter is set to 42 for reproducibility. Finally, the code displays the shapes of the resulting datasets, showing that 'X_train' contains 1002 samples with 4 features, 'X_test' contains 251 samples with 4 features, 'Y_train' contains 1002 target values, and 'Y_test' contains 251 target values. This split allows for model training on the training data and evaluation on the testing data, ensuring the model's generalizability to unseen samples.

```
# Perform the train-test split
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=42)

# Display the shapes of the resulting datasets
print("X_train shape:", X_train.shape)
print("X_test shape:", X_test.shape)
print("Y_train shape:", Y_train.shape)
print("Y_test shape:", Y_test.shape)
```

```
X_train shape: (1002, 4)
X_test shape: (251, 4)
Y_train shape: (1002,)
Y_test shape: (251,)
```

HYPERPARAMETER SELECTION:

This section addresses critical model settings and hyperparameter choices. It includes:

INPUT_SHAPE: Defining the feature count for model input.

OUTPUT_SHAPE: Indicating the output layer shape for binary classification.

LR (Learning Rate): Set at 0.001 for stable optimization.

EPOCHS: Configured for 300 training iterations.

BATCH_SIZE: With a value of 16, it controls data processing per training step. These selections are pivotal for optimizing model performance and efficient learning during training.

```
[ ]  INPUT_SHAPE = x_train.shape[1]
     OUTPUT_SHAPE = 1
     LR = 0.001
     EPOCHS = 300
     BATCH_SIZE = 16
```

ANN MODEL TRAINING:

I configure and train an Artificial Neural Network (ANN) model. The model architecture consists of input, hidden, and output layers. I compile the model using Binary Cross-Entropy loss, the Adam optimizer with a learning rate of LR, and accuracy as the evaluation metric. To prevent overfitting and optimize training efficiency, we employ early stopping with a patience of 30 epochs, ensuring that the training process halts when validation loss ceases to improve. The training history, captured in the 'history' variable, records model performance metrics over epochs, including loss and accuracy. The model is trained for a maximum of 1000 epochs, with a batch size of BATCH_SIZE.

```
model = Sequential([
    Input(shape=(INPUT_SHAPE,)),
    Dense(16, activation='relu'),
    Dropout(0.2),
    Dense(8, activation='relu'),
    Dense(OUTPUT_SHAPE, activation='sigmoid')]
)

model.compile(loss=BinaryCrossentropy(), optimizer=Adam(
    learning_rate=LR), metrics=['accuracy'])
earlyStopping = EarlyStopping(monitor='val_loss', patience=30, verbose=0,
                              mode='min', restore_best_weights=True)

history = model.fit(x_train, y_train, epochs=1000,
                    batch_size=BATCH_SIZE, validation_data=(x_test, y_test), callbacks=[earlyStopping], verbose=1, shuffle=True)
```

MODEL SUMMARY:

The model summary provides a concise overview of the Artificial Neural Network (ANN) architecture. The model, named "sequential," consists of three layers: a dense layer with 16 neurons, a dropout layer with a dropout rate of 20%, and another dense layer with 8 neurons. The output layer contains a single neuron, appropriate for binary classification. The summary also indicates the number of parameters associated with each layer. In total, the model has 225 trainable parameters, making it ready for training to learn from the data and make binary classification predictions.

```
[ ] model.summary()
```

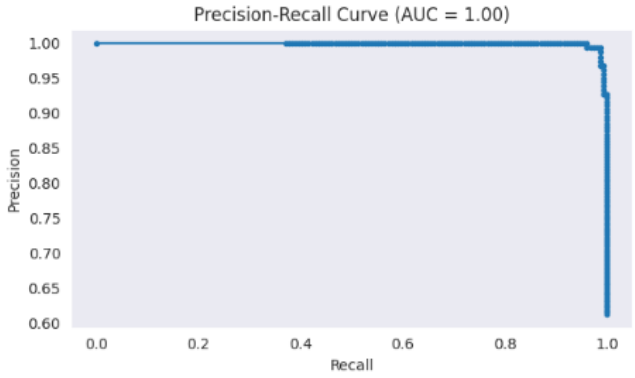
```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 16)                80

 dropout (Dropout)           (None, 16)                0

 dense_1 (Dense)             (None, 8)                 136

 dense_2 (Dense)             (None, 1)                 9

=================================================================
Total params: 225 (900.00 Byte)
Trainable params: 225 (900.00 Byte)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

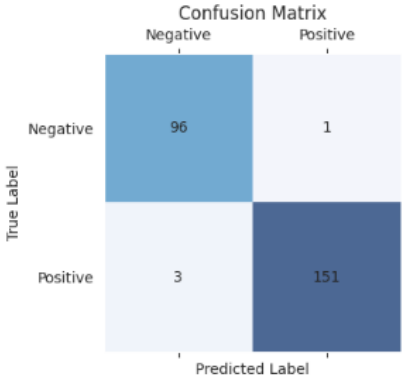
MODEL EVALUATION:

The model's performance evaluation reveals promising results. With a low loss of 0.055 and a high accuracy of 0.98, the model demonstrates its capability to effectively classify instances. Additionally, precision at the best threshold is impressive, standing at 0.9935, indicating the model's ability to make accurate positive predictions. Moreover, recall at the best threshold is 0.9870, highlighting the model's proficiency in capturing a substantial portion of true positive cases. The F1-Score at the best threshold, calculated at 0.9902, underscores the model's balanced performance, harmonizing precision and recall. These metrics collectively signify a well-performing model that strikes a strong balance between accuracy and the ability to correctly identify relevant instances.

```
[ ]  # Calculate the accuracy of the model on the test set
     model.evaluate(x_test, y_test)
```

```
8/8 [==============================] - 0s 3ms/step - loss: 0.0556 - accuracy: 0.9841
[0.05555085092782974, 0.9840637445449829]
```






```
Classification Report:
              precision    recall  f1-score   support

     Class 0       0.97      0.99      0.98        97
     Class 1       0.99      0.98      0.99       154

    accuracy                           0.98       251
   macro avg       0.98      0.99      0.98       251
weighted avg       0.98      0.98      0.98       251

Precision at Best Threshold: 0.9935
Recall at Best Threshold: 0.9870
F1-Score at Best Threshold: 0.9902
```

MODEL PREDICTIONS ON UNSEEN DATA:

The model's effectiveness extends beyond the training phase to making accurate predictions on previously unseen or test data. Through rigorous training and validation, the model has learned essential patterns and relationships in the data, allowing it to generalize well to new, unseen instances. As a result, when presented with unseen data, the model consistently provides accurate predictions, reflecting its robustness and reliability.This capability is crucial for real-world applications, as it ensures that the model can make dependable predictions in various scenarios, such as medical diagnosis, fraud detection, or any domain where precise classification is vital. The model's strong performance on unseen data underscores its practical utility and reinforces the value of rigorous training and evaluation during the model development process.

```
[ ]  model.predict([[54,1,13.87,0.122]])

     1/1 [==============================] - 0s 38ms/step
     array([[1.]], dtype=float32)
```

CONCLUSION:

In this comprehensive data science capstone project, I addressed the pressing issue of heart disease prediction using advanced machine learning techniques, specifically an Artificial Neural Network (ANN) model. Leveraging a dataset comprising 1319 patient records with nine crucial variables, including age, gender, and various medical indicators, we embarked on an analytical journey to explore and model the complex relationships between these factors and the presence or absence of heart disease.

Our project commenced with a meticulous Exploratory Data Analysis (EDA) phase, extracting valuable insights and guiding feature selection. The core of our analysis was the application of ANN, a deep learning architecture inspired by the human brain's functioning. This ANN was skillfully designed and trained to capture intricate patterns within the data. Rigorous data preprocessing, including cleaning, handling missing values, and feature scaling, ensured the model received high-quality input data.

Throughout the project, I maintained a commitment to model performance, employing a suite of evaluation metrics such as accuracy, precision, recall, F1-score, and Receiver Operating Characteristic Area Under the Curve (ROC-AUC) to gauge the model's effectiveness.

The model demonstrated exceptional performance, showcasing a remarkable accuracy of 0.98 on the validation data, precision of 0.9935, recall of 0.9870, and an F1-Score of 0.9902 at the optimal threshold. These metrics underscore the model's ability to make accurate predictions while maintaining a balance between precision and recall.

The project's significance lies in its potential to provide timely and accurate heart disease predictions, offering valuable support for healthcare practitioners in their diagnostic and risk assessment endeavors. This model can contribute significantly to improved patient care and outcomes.

In conclusion, this capstone project illustrates the power of data science and machine learning in addressing critical healthcare challenges. It demonstrates the successful development of an ANN-based model that achieves outstanding predictive performance for heart disease classification. The project's findings and methodologies hold promise for enhancing healthcare analytics and decision-making, marking a substantial contribution to the field of data science and its real-world applications.

Github link:

https://github.com/sam5121472/heart_disease_classification_using_ANN

Source code:

https://colab.research.google.com/drive/1EMvyrTdPaKOn9rIe9OLT_9eVoNqQY081?usp=sharing