Predicting Heart Disease Using Machine Learning: Comparative Study on the UCI Cleveland Dataset

Manoj Kumar Gunupudi
Student of Data Science
Student ID - 15917205
Coventry University
Coventry, England
gunupudim@coventry.ac.uk

Abstract: **The study is an exploration of predicting heart disease based on the clinical features of the UCI Cleveland dataset. Diagnostic patterns are evaluated by using data preprocessing, exploratory analysis, supervised classification and clustering techniques to evaluate the patterns. "Random Forest" has the highest performance and shows a high level of predictive reliability, and can aid in early detection to facilitate clinical decision-making.**

Keywords: Heart Disease, Machine Learning, Classification, Clustering, Cleveland Dataset, Medical Diagnosis

I. INTRODUCTION

Heart disease in the present clinical world is still considered one of the major causes of death across the globe, and that is the reason there are urgent demands towards accurate and early diagnostic means. Physician expertise is an important part of the traditional clinical evaluation, which might be constrained owing to subjectivity and time. Machine learning (ML) is a data-based method that can help to detect fine details in clinical features that can be challenging to notice with other traditional methods. The most common medical ML research dataset, the UCI "Cleveland Heart Disease" dataset, offers an optimal benchmark of predictive algorithms assessment as it presents a wide range of clinical features. The paper examines the predictive abilities of various machine learning models, which include the "Logistic Regression", "Support Vector Machine" (SVM), "Random Forest", and the "K-Means clustering" to determine the presence of heart disease.

A. Aim and Objectives

Aim

The aim of the study is to develop an accurate as well as reliable diagnostic approach for identifying the presence of heart disease using clinical patient data, enhancing early detection and supporting improved medical decision-making.

Objectives

- To pre-process and analyses the "Cleveland Heart Disease" data to determine the most important clinical features that affect the prediction of heart diseases.
- To carry out and optimize "Logistic Regression", "SVM" and "Random Forest" models for supervised classification.
- To use "K-Means clustering" to examine underlying patterns and groupings in the data.
- To provide a comparison of the performance of all models based on the relevant evaluation metrics to identify the most efficient format to adopt in the detection of early heart diseases.

II. PROBLEM AND DATA SET

A. Problem Definition

Heart disease remains one of the significant challenges to the state of the population since it adds to the rates of morbidity and mortality worldwide [1]. The timely and correct identification is essential to successful treatment, but conventional diagnostic procedures are usually based on a subjective clinical interpretation and can miss unobvious interactions of risk factors. Another potent alternative, which is offered by machine learning, is the discovery of complex and non-linear patterns in medical data, enabling more objective and consistent decision support.

The main issue in the study is the creation of stable computational frameworks that are able to forecast the existence of heart disease based on the patient's clinical characteristics [2]. The study focuses on establishing the most effective machine learning methods to determine the most predictive approach by examining major diagnostic indicators such as "age", "type of chest pain", "blood pressure", "cholesterol level", and "symptoms caused by exercise". The issue is to compare and analyses various algorithms to provide an opportunity to diagnose earlier and minimize the uncertainty of the diagnosis, as well as to improve clinical outcomes.

B. Dataset Description

The study employs the “Cleveland Heart Disease” dataset, which is available on the UCI “Machine Learning Repository”, which is one of the most popular cardiovascular prediction datasets [3]. Even though there are 76 variables in the original database, only 14 major attributes are published in experiments; they include both clinical measurements and patient demographic data. They are “age”, “sex”, “type of chest pain”, “rest blood pressure”, “serum cholesterol”, “fasting blood sugar”, “resting ECG”, “maximum heart rate”, “exercise induced angina”, “oldpeak”, “slope”, “number of major vessels”, and “thalassemia” [4].

The target variable, “num”, measures the heart disease severity as 0-4, but normally is converted into a binary classification problem: 0-absence and 1-4-presence of the disease. The data has a proper structure, it is clinically significant, and it can be used to test various machine learning methods since it contains both categorical and continuous variables.

III. METHODS

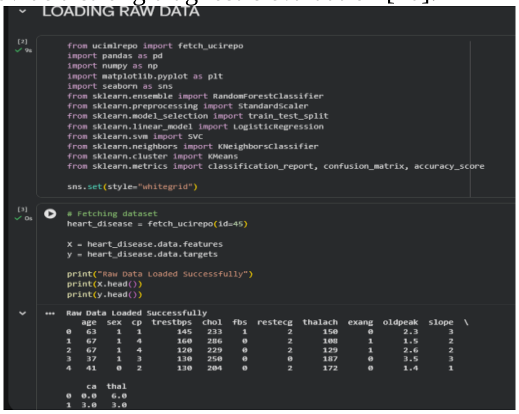
The study used a structured methodological process in its attempt to analyses the clinical features of the UCI “Cleveland Heart Disease” database and generate an effective diagnostic model. The process started with the preprocessing of the data, where missing values are removed and managed based on its suitable strategies of imputation [6]. Categorical variables such as “type of chest pain”, status of “fasting blood sugar level”, and “thalassemia” are coded to make them compatible with numerical learning algorithms. Continuous variables are all standardized to enhance the stability of the model and eliminate bias owing to scale [7].

After preprocessing, the supervised and unsupervised methods of learning are implemented in order to investigate diagnostic patterns in the data. The supervised learning phase consisted of training various classification models to differentiate between the patients that do not have heart disease, and unsupervised clustering is performed to reveal the natural groupings and evaluate the intrinsic structure of the dataset [8].

To make the results objective, a train-test split is taken, and to enhance the reliability of the results, cross-validation techniques are conducted. Metrics such as “accuracy”, “precision”, “recall”, and “F1-score” are used to evaluate the performance, which gives a high-level comparison of the diagnostic effectiveness [9]. It is a rigorous methodology that guarantees a sound and repeatable study on data-driven diagnosis of heart diseases.

IV. EXPERIMENTAL SETUP

The dataset is cleaned, encoded and standardized and then analyzed [5]. The data is divided into training set and a testing set, and cross-validation is used to enhance reliability. Optimized parameters are applied to classification and clustering models, and the models are evaluated based on the performance parameters of “accuracy”, “precision”, “recall”, and “F1-score” to provide a strong diagnostic evaluation [10].



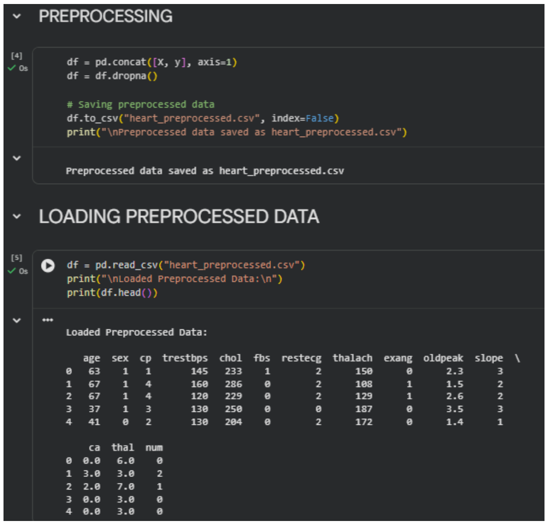
```
from ucimlrepo import fetch_ucirepo
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.cluster import KMeans
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score

sns.set(style="whitegrid")
```

```
# Fetching dataset
heart_disease = fetch_ucirepo(id=45)

X = heart_disease.data.features
y = heart_disease.data.targets

print("Raw Data Loaded Successfully")
print(X.head())
print(y.head())
```

```
Raw Data Loaded Successfully
   age  sex  cp  trestbps  chol  fbs  restecg  thalach  exang  oldpeak  slope  \
0   63    1   1       145   233    1        2      150      0      2.3      3
1   67    1   4       160   286    0        2      108      1      1.5      2
2   67    1   4       120   229    0        2      129      1      2.6      2
3   37    1   3       130   250    0        0      187      0      3.5      3
4   41    0   2       130   204    0        2      172      0      1.4      1

    ca  thal
0  0.0   6.0
1  3.0   3.0
```

Fig 1: Loading row data

The Python code in the Google Colab platform brings libraries of data retrieval, data manipulation, and data visualisation using “ucimlrepo”, “pandas”, “numpy”, “matplotlib”, “seaborn”. It is based on scikit-learn tools to perform preprocessing, model construction such as “Logistic Regression”, “SVM”, “Random Forest”, “KNN”, “K-Means”, train-test splitting, and performance analysis. The database is then accessed and viewed.



```
df = pd.concat([X, y], axis=1)
df = df.dropna()

# Saving preprocessed data
df.to_csv("heart_preprocessed.csv", index=False)
print("\nPreprocessed data saved as heart_preprocessed.csv")
```

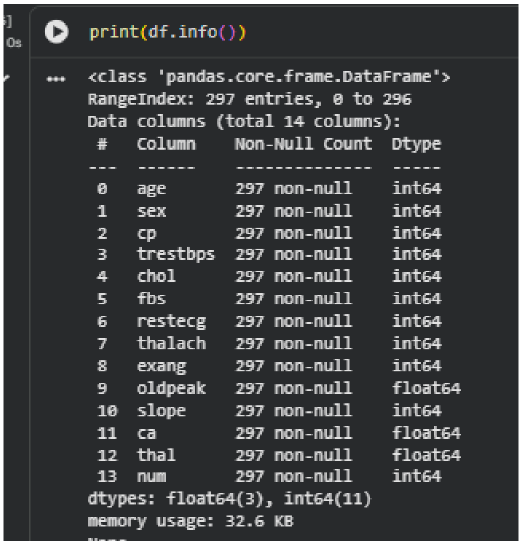
```
Preprocessed data saved as heart_preprocessed.csv
```

```
df = pd.read_csv("heart_preprocessed.csv")
print("\nLoaded Preprocessed Data:\n")
print(df.head())
```

```
Loaded Preprocessed Data:

   age  sex  cp  trestbps  chol  fbs  restecg  thalach  exang  oldpeak  slope  \
0   63    1   1       145   233    1        2      150      0      2.3      3
1   67    1   4       160   286    0        2      108      1      1.5      2
2   67    1   4       120   229    0        2      129      1      2.6      2
3   37    1   3       130   250    0        0      187      0      3.5      3
4   41    0   2       130   204    0        2      172      0      1.4      1

    ca  thal  num
0  0.0   6.0    0
1  3.0   3.0    2
2  2.0   7.0    1
3  0.0   3.0    0
4  0.0   3.0    0
```

Fig 2: Saving the process data and loading the data

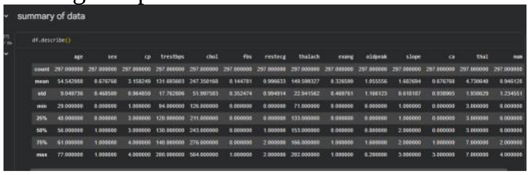
The Python code combines the features and targets, eliminates missing values, saves the cleaned data to a CSV file and then reloads it and shows the first few rows to ensure that it has been preprocessed successfully.



```
print(df.info())

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 297 entries, 0 to 296
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   age       297 non-null    int64
 1   sex       297 non-null    int64
 2   cp        297 non-null    int64
 3   trestbps  297 non-null    int64
 4   chol      297 non-null    int64
 5   fbs       297 non-null    int64
 6   restecg   297 non-null    int64
 7   thalach   297 non-null    int64
 8   exang     297 non-null    int64
 9   oldpeak   297 non-null    float64
 10  slope     297 non-null    int64
 11  ca        297 non-null    float64
 12  thal      297 non-null    float64
 13  num       297 non-null    int64
dtypes: float64(3), int64(11)
memory usage: 32.6 KB
```

Fig 3. Data information

The dataset has 297 complete records, having 14 clinical attributes, most of which are integers, with a few floats. After Preprocessing, there are no missing values. Its balanced structure and homogeneous types of data make it appropriate for statistical analysis, modelling and prediction of heart disease.



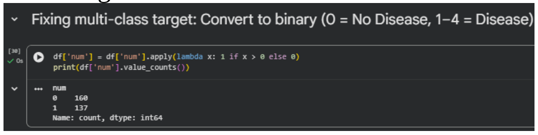
summary of data

```
df.describe()
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	num
count	297.000000	297.000000	297.000000	297.000000	297.000000	297.000000	297.000000	297.000000	297.000000	297.000000	297.000000	297.000000	297.000000	297.000000
mean	54.542088	0.676768	3.158249	131.693603	247.350168	0.144781	0.996633	149.599327	0.326599	1.055556	1.602694	0.676768	4.730640	0.946128
std	9.049736	0.468500	0.964859	17.762806	51.997583	0.352474	0.994914	22.941562	0.469761	1.166123	0.618187	0.938965	1.938629	1.234551
min	29.000000	0.000000	1.000000	94.000000	126.000000	0.000000	0.000000	71.000000	0.000000	0.000000	1.000000	0.000000	3.000000	0.000000
25%	48.000000	0.000000	3.000000	120.000000	211.000000	0.000000	0.000000	133.000000	0.000000	0.000000	1.000000	0.000000	3.000000	0.000000
50%	56.000000	1.000000	3.000000	130.000000	243.000000	0.000000	1.000000	153.000000	0.000000	0.800000	2.000000	0.000000	3.000000	0.000000
75%	61.000000	1.000000	4.000000	140.000000	276.000000	0.000000	2.000000	166.000000	1.000000	1.600000	2.000000	1.000000	7.000000	2.000000
max	77.000000	1.000000	4.000000	200.000000	564.000000	1.000000	2.000000	202.000000	1.000000	6.200000	3.000000	3.000000	7.000000	4.000000

Fig 4. Summary of data

The data summary demonstrates a wide range of ages of the patients, different cholesterol and blood pressure, and equal distributions of features. There is also a significant change in clinical attributes, which can be used to detect patterns effectively in heart disease diagnosis.



Fixing multi-class target: Convert to binary (0 = No Disease, 1–4 = Disease)

```
df['num'] = df['num'].apply(lambda x: 1 if x > 0 else 0)
print(df['num'].value_counts())

num
0    160
1    137
Name: count, dtype: int64
```

Fig 5. Fixing the target column

Digitizing the target provides a clear line between a healthy and a diseased patient. The value counts will verify the distribution of both classes, which can help in the assessment of balance and allow the proper training and assessment of the model.

V. RESULTS

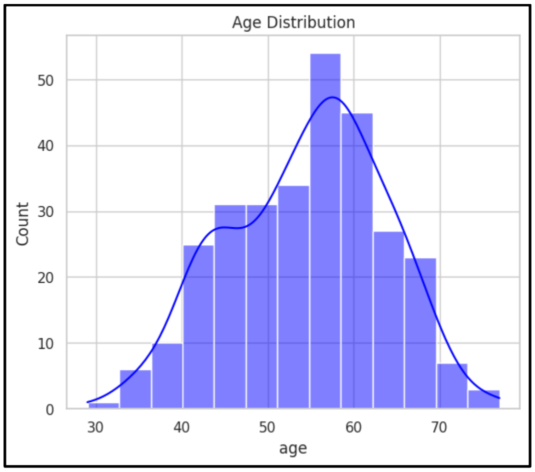


Fig 6.Histogram for age distribution

The age breaks are that the majority of the patients are within the age range of 45 to 65 years, meaning that the older and middle-aged adults are predominant in the dataset. The near-normal distribution indicates smooth age distributions, fewer young and older age groups, and indicates normal patterns of heart disease risks.

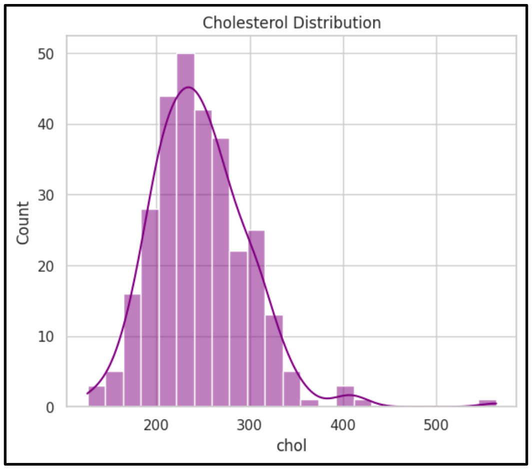


Fig 7. Histogram for cholesterol distribution

The distribution of the cholesterol is skewed right, and the majority of the patients are in the range of 200 and 300mg/dl. There are a few extreme outliers which exceed 400 mg/dL, pointing to the possible high-risk individuals. The spread shows that there is a great variation in lipid profiles between patients, which is applicable in cardiovascular examination.

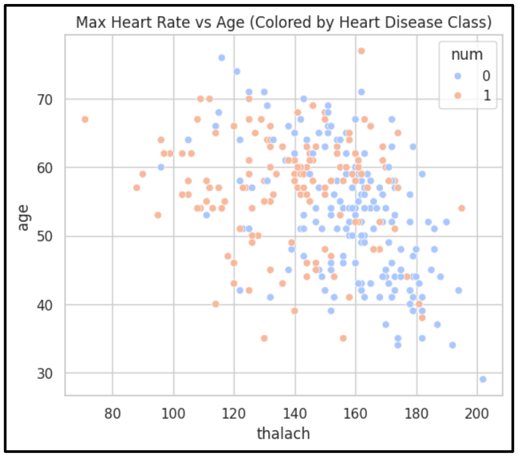


Fig 8. Scatter plot for max heart rate vs age

The scatter plot indicates that there is an inverse correlation between age and maximum heart rate, which is expected of the physiological process. The higher rates are observed in younger individuals, and smaller values are observed in older patients. Individually, colour coding indicates that cases of heart diseases are more compactly concentrated where age increases, and heart rate decreases.

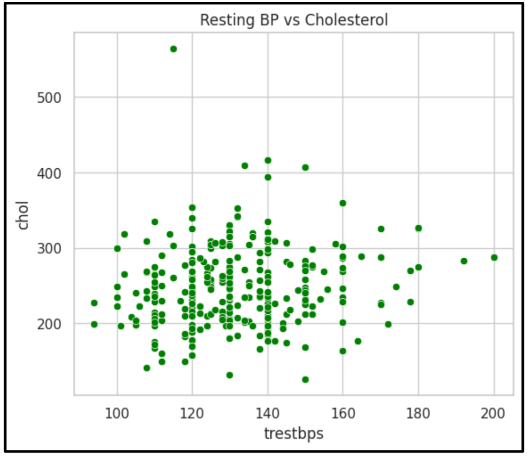
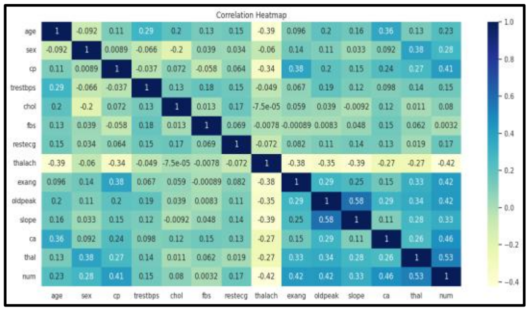


Fig 9. Scatter plot for resting BP vs Cholesterol

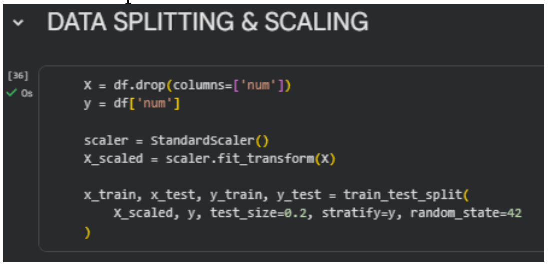
The plot of the resting blood pressure versus cholesterol reveals that the two variables do not exhibit any strong linear association. The two measures are broadly dispersed among the patients, indicating varying cardiovascular risk profiles. Certain groups indicated that those people who have moderately high blood pressure levels tend to have moderate cholesterol levels.



	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	num
age	1	-0.092	0.11	0.29	0.2	0.13	0.15	-0.39	0.096	0.2	0.16	0.36	0.13	0.23
sex	-0.092	1	0.0089	-0.066	-0.2	0.039	0.034	-0.06	0.14	0.11	0.033	0.092	0.38	0.28
cp	0.11	0.0089	1	-0.037	0.072	-0.058	0.064	-0.34	0.38	0.2	0.15	0.24	0.27	0.41
trestbps	0.29	-0.066	-0.037	1	0.13	0.18	0.15	-0.049	0.067	0.19	0.12	0.098	0.14	0.15
chol	0.2	-0.2	0.072	0.13	1	0.013	0.17	-7.5e-05	0.059	0.039	-0.0092	0.12	0.011	0.08
fbs	0.13	0.039	-0.058	0.18	0.013	1	0.069	-0.0078	-0.00089	0.0083	0.048	0.15	0.062	0.0032
restecg	0.15	0.034	0.064	0.15	0.17	0.069	1	-0.072	0.082	0.11	0.14	0.13	0.019	0.17
thalach	-0.39	-0.06	-0.34	-0.049	-7.5e-05	-0.0078	-0.072	1	-0.38	-0.35	-0.39	-0.27	-0.27	-0.42
exang	0.096	0.14	0.38	0.067	0.059	-0.00089	0.082	-0.38	1	0.29	0.25	0.15	0.33	0.42
oldpeak	0.2	0.11	0.2	0.19	0.039	0.0083	0.11	-0.35	0.29	1	0.58	0.29	0.34	0.42
slope	0.16	0.033	0.15	0.12	-0.0092	0.048	0.14	-0.39	0.25	0.58	1	0.11	0.28	0.33
ca	0.36	0.092	0.24	0.098	0.12	0.15	0.13	-0.27	0.15	0.29	0.11	1	0.26	0.46
thal	0.13	0.38	0.27	0.14	0.011	0.062	0.019	-0.27	0.33	0.34	0.28	0.26	1	0.53
num	0.23	0.28	0.41	0.15	0.08	0.0032	0.17	-0.42	0.42	0.42	0.33	0.46	0.53	1

Fig 10. Correlation heatmap

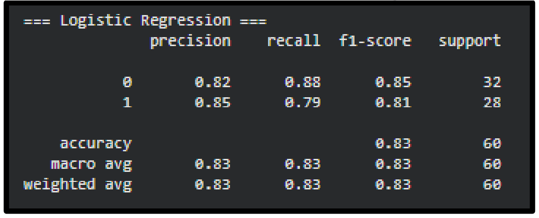
Correlation Heatmap developed in Google Colab indicates the existence of moderate correlations among critical clinical variables and heart disease, including “chest pains”, “slope”, “oldpeak”, and “maximum heart rate”. The negative relationship with “thalach” as well as the positive relationship with “oldpeak” and “ca” show that there are significant diagnostic relationships over which features would be useful in the prediction.



```
X = df.drop(columns=['num'])
y = df['num']

scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

x_train, x_test, y_train, y_test = train_test_split(
    X_scaled, y, test_size=0.2, stratify=y, random_state=42
)
```

Fig 11. Data selection and train-test splitting

The features are then decoupled to the target variable, scaled by using “StandardScaler” that ensures equal scaling and the data is then divided into training and test sets. Stratified sampling maintains the balance in the classes, and a fixed random state guarantees that it can be reproduced later when assessing the model.



```
=== Logistic Regression ===
              precision    recall  f1-score   support

           0       0.82      0.88      0.85        32
           1       0.85      0.79      0.81        28

    accuracy                           0.83        60
   macro avg       0.83      0.83      0.83        60
weighted avg       0.83      0.83      0.83        60
```

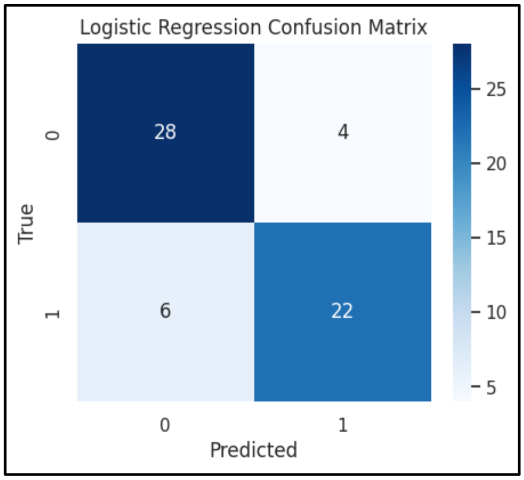
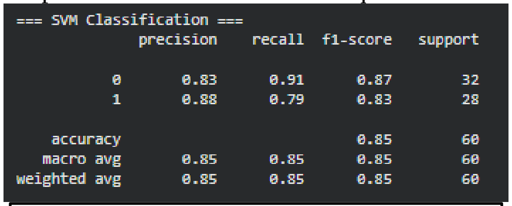


Fig 12. Classification report and Confusion matrix for Logistic regression

The overall accuracy of logistic Regression is 83% indicating even results in both classes. It has a better prediction of non-disease cases, but class 0 is better recalled. The cases of the disease exhibit lower recall of 0.79, which means there are some positives missed. The model has steady precision (0.83) and F1-scores, which signify a stable linear decision-boundary performance that can be used in baseline medical classification. Logistic Regression accurately determines the majority of cases, but cannot identify some cases of the disease. False negatives are only moderate, with good performance but low sensitivity in comparison with ensemble techniques.



```
=== SVM Classification ===
              precision    recall  f1-score   support

           0       0.83      0.91      0.87        32
           1       0.88      0.79      0.83        28

    accuracy                           0.85        60
   macro avg       0.85      0.85      0.85        60
weighted avg       0.85      0.85      0.85        60
```

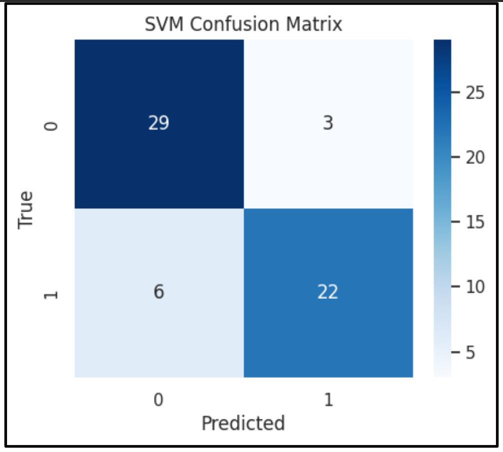
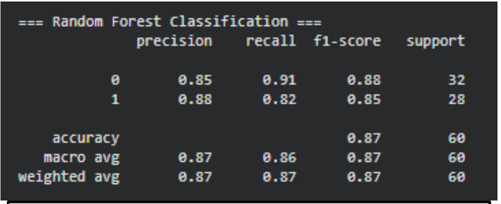


Fig 13. Classification report and Confusion matrix for SVM

The SVMs model is stronger with a 85% accuracy and enhances better discrimination of classes. It has great recall of 0.85 in non-disease cases and an increase in precision (0.85) in disease cases, which is greater separation of boundaries. Class 1 slight recall reduction means that there are slightly missed cases of the disease, but the overall "F1-scores" are high. SVM has better generalization than "Logistic Regression" and is able to deal with complex patterns. SVM enhances the detection of non-diseases with reduced false positives, and it remains steady. False negatives remain, in general classification there is greater boundary separation as compared to "Logistic Regression".



```
=== Random Forest Classification ===
              precision    recall  f1-score   support

           0       0.85      0.91      0.88        32
           1       0.88      0.82      0.85        28

    accuracy                           0.87        60
   macro avg       0.87      0.86      0.87        60
weighted avg       0.87      0.87      0.87        60
```

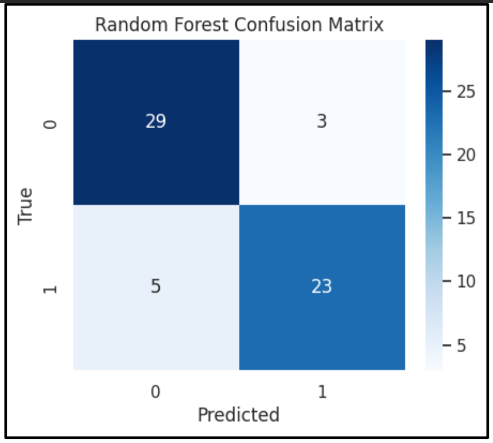
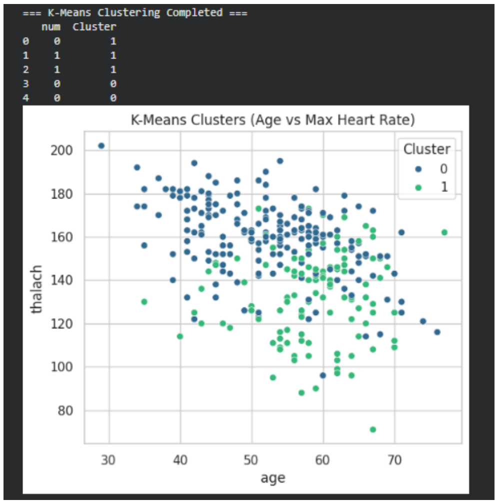


Fig 14. Classification report and Confusion matrix for Random Forest

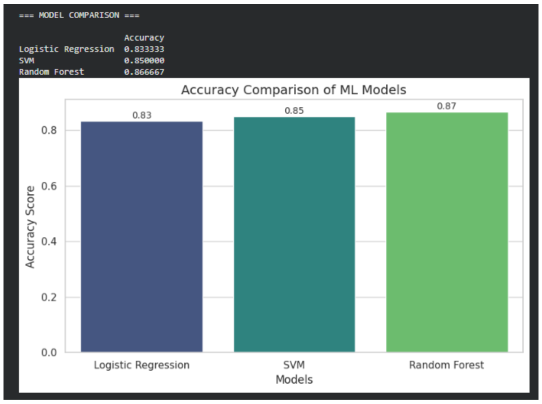
Random Forest is the most balanced and robust since it has the highest accuracy of 87%. It demonstrates high accuracy, recall (0.86) and F1-score (0.87) of the two classes, and better classification of the disease cases than previous models. Its ensemble design is able to capture complex feature interactions thereby minimizing misclassification. It implies that the most efficient and sound model used in predicting heart disease in this study is the "Random Forest". Random Forest provides the most desirable balance that minimizes the number of false positives and false negatives. It is able to capture difficult patterns and provide the most dependable and powerful diagnosis classification.



```
=== K-Means Clustering Completed ===
   num  Cluster
0    0        1
1    1        1
2    1        1
3    0        0
4    0        0
```

Fig 15. K-means clustering

K-Means clustering shows that there are two data sets, as age and maximum heart rate have different clusters. Influential natural patterns are observed in the heart disease risk stratification as younger patients have a higher number of heart rates, and older patients create a distinct cluster.



```
=== MODEL COMPARISON ===

                     Accuracy
Logistic Regression  0.833333
SVM                  0.850000
Random Forest        0.866667
```

Fig 16. Model comparison

Random Forest has an 87% accuracy, which is superior to SVM and "Logistic Regression" and therefore it has the potential to predict better than the other two. SVM is 85th, where Logistic Regression is 83rd, which demonstrates the performance improvement with the more complex models and ensemble techniques.

VI. DISCUSSION AND CONCLUSIONS

A. Discussion

The study compared the prediction of heart disease through "Logistic Regression", "SVM" and "Random Forest" models using standardized clinical features [11].

Model	Accuracy	Strength
"Logistic Regression"	0.83	Interpretable, stable results
"SVM"	0.85	Strong boundary separation
"Random Forest"	0.87	Best performance, robust

Table 1. Summary of Model Performance

"Random Forest" worked the most, as it has an increased accuracy and a balanced classification between classes [12]. The influential variables are identified, and visual exploration showed that age, cholesterol, and maximum heart rate are influential variables. The clustering offered more organization of patient groups, which is helpful in their interpretation [13]. The combination of supervised and unsupervised approaches enhanced the credibility of the analysis and enhanced the interpretation of the patterns of diseases.

B. Conclusion

The analysis proves that "Random Forest" provides the most accurate heart disease forecasts with the knowledge of significant variable relationships and clustering. The integrated analysis method increases the interpretability of the diagnostic and promotes the use of data to make decisions. In the future, further work can be done by adding more clinical attributes or increasing the size of the datasets to enhance the level of prediction further.

VII. REFERENCES

[1] Tsao, C.W., Aday, A.W., Almarzooq, Z.I., Anderson, C.A., Arora, P., Avery, C.L., Baker-Smith, C.M., Beaton, A.Z., Boehme, A.K., Buxton, A.E. and Commodore-Mensah, Y., 2023. Heart disease and stroke statistics—2023 update: a report from the American Heart Association. Circulation, 147(8), pp.e93-e621. https://doi.org/10.1161/CIR.0000000000001123

[2] Shin, K.C., Ali Moussa, H.Y. and Park, Y., 2024. Cholesterol imbalance and neurotransmission defects in neurodegeneration. Experimental & Molecular Medicine, 56(8), pp.1685-1690. https://doi.org/10.1038/s12276-024-01273-4

[3] El Morr, C., Jammal, M., Ali-Hassan, H. and El-Hallak, W., 2022. Data preprocessing. In Machine learning for practical decision making: a multidisciplinary perspective with applications from healthcare, engineering and business analytics (pp. 117-163). Cham: Springer International Publishing. https://doi.org/10.1007/978-3-031-16990-8_4

[4] Pathan, M.S., Nag, A., Pathan, M.M. and Dev, S., 2022. Analyzing the impact of feature selection on the accuracy of heart disease prediction. Healthcare Analytics, 2, p.100060.

[5] Liu, M., Ma, H., Zhang, Y. and Yue, F., 2024, April. Multi-Source Data Preprocessing Method Research Based on Python. In 2024 5th International Conference on Geology, Mapping and Remote Sensing (ICGMRS) (pp. 221-224). IEEE.

[6] Jansen, B.J., Aldous, K.K., Salminen, J., Almerekhi, H. and Jung, S.G., 2023. Data preprocessing. In Understanding Audiences, Customers, and Users via Analytics: An Introduction to the Employment of Web, Social, and Other Types of Digital People Data (pp. 65-75). Cham: Springer Nature Switzerland.

[7] Dasari, D. and Varma, P.S., 2022, December. Employing various data cleaning techniques to achieve better data quality using python. In 2022 6th International Conference on Electronics, Communication and Aerospace Technology (pp. 1379-1383). IEEE. https://doi.org/10.1109/ICECA55336.2022.10009079

[8] Graffelman, J. and De Leeuw, J., 2023. Improved approximation and visualization of the correlation matrix. The American Statistician, 77(4), pp.432-442.

[9] Zabor, E.C., Reddy, C.A., Tendulkar, R.D. and Patil, S., 2022. Logistic regression in clinical studies. International Journal of Radiation Oncology* Biology* Physics, 112(2), pp.271-277.

[10] Elkahwagy, D.M.A.S., Kiriacos, C.J. and Mansour, M., 2024. Logistic regression and other statistical tools in diagnostic biomarker studies. Clinical and translational oncology, 26(9), pp.2172-2180.
[11] Chandra, M.A. and Bedi, S.S., 2021. Survey on SVM and their application in image classification. International Journal of Information Technology, 13(5), pp.1-11.
[12] Iranzad, R. and Liu, X., 2025. A review of random forest-based feature selection methods for data science education and applications. International Journal of Data Science and Analytics, 20(2), pp.197-211.
[13] Salman, H.A., Kalakech, A. and Steiti, A., 2024. Random forest algorithm overview. Babylonian Journal of Machine Learning, 2024, pp.69-79.

APPENDICES

```
!pip install ucimlrepo
from ucimlrepo import fetch_ucirepo
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.cluster import KMeans
from sklearn.metrics import classification_report, confusion_matrix,
accuracy_score

sns.set(style="whitegrid")
# Fetching dataset
heart_disease = fetch_ucirepo(id=45)

X = heart_disease.data.features
y = heart_disease.data.targets

print("Raw Data Loaded Successfully")
print(X.head())
print(y.head())
df = pd.concat([X, y], axis=1)
df = df.dropna()

# Saving preprocessed data
df.to_csv("heart_preprocessed.csv", index=False)
print("\nPreprocessed data saved as heart_preprocessed.csv")
df = pd.read_csv("heart_preprocessed.csv")
print("\nLoaded Preprocessed Data:\n")
print(df.head())
print(df.info())
df.describe()
df['num'] = df['num'].apply(lambda x: 1 if x > 0 else 0)
print(df['num'].value_counts())

#  Age distribution
plt.figure(figsize=(14, 18))
plt.subplot(3, 2, 1)
sns.histplot(df['age'], kde=True, color='blue')
plt.title("Age Distribution")
# Cholesterol distribution
plt.figure(figsize=(14, 18))
plt.subplot(3, 2, 2)
sns.histplot(df['chol'], kde=True, color='purple')
plt.title("Cholesterol Distribution")
# Max Heart Rate vs Disease Class
plt.figure(figsize=(14, 18))
plt.subplot(3, 2, 3)
```

```
sns.scatterplot(data=df, x='thalach', y='age', hue='num',
palette='coolwarm')
plt.title("Max Heart Rate vs Age (Colored by Heart Disease Class)")
#  Resting BP vs Cholesterol
plt.figure(figsize=(14, 18))
plt.subplot(3, 2, 4)
sns.scatterplot(data=df, x='trestbps', y='chol', color='green')
plt.title("Resting BP vs Cholesterol")
#  Heatmap
plt.figure(figsize=(14, 18))
plt.subplot(3, 2, (5, 6))
sns.heatmap(df.corr(), annot=True, cmap="YlGnBu")
plt.title("Correlation Heatmap")

plt.tight_layout()
plt.show()
X = df.drop(columns=['num'])
y = df['num']

scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

x_train, x_test, y_train, y_test = train_test_split(
    X_scaled, y, test_size=0.2, stratify=y, random_state=42
)
def plot_conf_matrix(y_true, y_pred, title):
    plt.figure(figsize=(5, 4))
    cm = confusion_matrix(y_true, y_pred)
    sns.heatmap(cm, annot=True, fmt='d', cmap="Blues")
    plt.title(title)
    plt.xlabel("Predicted")
    plt.ylabel("True")
    plt.show()
results = {}

log_reg = LogisticRegression(max_iter=500)
log_reg.fit(x_train, y_train)
y_pred_lr = log_reg.predict(x_test)

print("\n=== Logistic Regression ===")
print(classification_report(y_test, y_pred_lr))
plot_conf_matrix(y_test, y_pred_lr, "Logistic Regression Confusion Matrix")

results["Logistic Regression"] = accuracy_score(y_test, y_pred_lr)
svm = SVC(kernel='rbf')
svm.fit(x_train, y_train)
y_pred_svm = svm.predict(x_test)

print("\n=== SVM Classification ===")
print(classification_report(y_test, y_pred_svm))
plot_conf_matrix(y_test, y_pred_svm, "SVM Confusion Matrix")

results["SVM"] = accuracy_score(y_test, y_pred_svm)
rf = RandomForestClassifier(
    n_estimators=200,
    max_depth=None,
    min_samples_split=2,
    random_state=42
)

rf.fit(x_train, y_train)
y_pred_rf = rf.predict(x_test)

print("\n=== Random Forest Classification ===")
print(classification_report(y_test, y_pred_rf))
plot_conf_matrix(y_test, y_pred_rf, "Random Forest Confusion Matrix")

results["Random Forest"] = accuracy_score(y_test, y_pred_rf)

kmeans = KMeans(n_clusters=2, random_state=42)
cluster_labels = kmeans.fit_predict(X_scaled)

df['Cluster'] = cluster_labels

print("\n=== K-Means Clustering Completed ===")
print(df[['num', 'Cluster']].head())

plt.figure(figsize=(6, 5))
sns.scatterplot(data=df, x='age', y='thalach', hue='Cluster',
palette='viridis')
plt.title("K-Means Clusters (Age vs Max Heart Rate)")
plt.show()
comparison_df = pd.DataFrame.from_dict(results, orient='index',
columns=['Accuracy'])
print("\n\n=== MODEL COMPARISON ===\n")
print(comparison_df)

plt.figure(figsize=(8,5))
sns.barplot(
    x=comparison_df.index,
    y=comparison_df['Accuracy'],
    hue=comparison_df.index,
    palette="viridis",
    legend=False
)

plt.title("Accuracy Comparison of ML Models", fontsize=14)
plt.xlabel("Models", fontsize=12)
plt.ylabel("Accuracy Score", fontsize=12)

for i, v in enumerate(comparison_df['Accuracy']):
    plt.text(i, v + 0.01, f"{v:.2f}", ha='center', fontsize=10)

plt.tight_layout()
plt.show()
```

Word Count - 3070