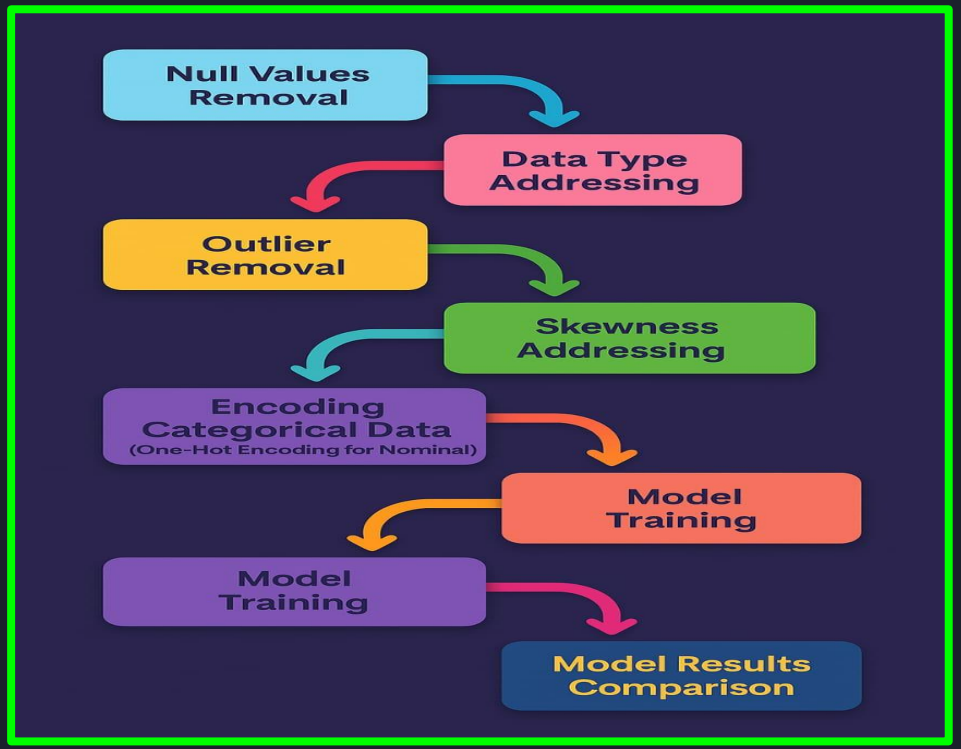
HEART DISEASE PREDICTION

Utilizing Machine Learning For Early Diagnosis And Prediction Of Heart Diseases.

COURSE OF ACTION



INTRODUCTION

Machine Learning (ML) is playing a vital role in the early detection and diagnosis of **heart diseases**. By analyzing clinical data, patient history, lifestyle factors, and physiological measurements, ML models can accurately predict the likelihood of heart-related conditions. These intelligent systems support healthcare professionals in making timely, data-driven decisions, which is crucial for effective treatment and prevention. Commonly used techniques include **logistic regression**, **decision trees**, **random forests**, **and** **neural networks**. Overall, ML enhances the precision, speed, and reliability of heart disease prediction and risk assessment.

Heart Disease Dataset - Variable Summary

No.	Variable	Full Form / Description	Data Type	Variable Type
1	age	Age of the patient	Integer	Continuous (Numerical)
2	sex	Sex (1 = male; 0 = female)	Integer	Categorical (Binary)
3	cp	Chest pain type	Integer	Nominal (Categorical)
4	trestbps	Resting blood pressure (mm Hg)	Integer	Continuous (Numerical)
5	chol	Serum cholesterol (mg/dl)	Integer	Continuous (Numerical)
6	fbs	Fasting blood sugar > 120 mg/dl	Integer	Categorical (Binary)
7	restecg	Resting electrocardiographic results	Integer	Nominal (Categorical)
8	thalach	Max heart rate achieved	Integer	Continuous (Numerical)
9	exang	Exercise induced angina	Integer	Categorical (Binary)
10	oldpeak	ST depression from exercise	Float	Continuous (Numerical)
11	slope	Slope of peak exercise ST segment	Integer	Ordinal (Categorical)
12	ca	No. of major vessels (0-3)	Integer	Ordinal (Categorical)
13	thal	Thalassemia	Integer	Nominal (Categorical)
14	target	Heart disease presence	Integer	Categorical (Binary)

DATA SET OVERVIEW

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
count	287.000000	291.000000	286.000000	287.000000	290.000000	288.000000	294.000000	286.000000	295.000000	290.000000	288.000000	275.000000	292.000000	285.000000
mean	54.571429	0.683849	1.000000	131.383275	246.575862	0.145833	0.530612	149.772727	0.332203	1.056207	1.399306	0.727273	2.315068	0.554386
std	8.994615	0.465774	1.042938	17.725159	51.414081	0.353553	0.526514	22.520184	0.471804	1.177696	0.622124	1.026200	0.617649	0.497908
min	29.000000	0.000000	0.000000	94.000000	126.000000	0.000000	0.000000	88.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	48.000000	0.000000	0.000000	120.000000	211.250000	0.000000	0.000000	134.500000	0.000000	0.000000	1.000000	0.000000	2.000000	0.000000
50%	56.000000	1.000000	1.000000	130.000000	240.500000	0.000000	1.000000	152.000000	0.000000	0.800000	1.000000	0.000000	2.000000	1.000000
75%	61.000000	1.000000	2.000000	140.000000	274.000000	0.000000	1.000000	166.000000	1.000000	1.750000	2.000000	1.000000	3.000000	1.000000
max	77.000000	1.000000	3.000000	200.000000	564.000000	1.000000	2.000000	202.000000	1.000000	6.200000	2.000000	4.000000	3.000000	1.000000

CHECK FOR NULL VALUES

```
df.isnull().sum()
```

	0
age	16
sex	12
cp	17
trestbps	16
chol	13
fbs	15
restecg	9
thalach	17
exang	8
oldpeak	13
slope	15
ca	28
thal	11
target	18

dtype: int64

DATATYPE MATCHING(NUMERICAL AND CATEGORICAL)

BEFORE

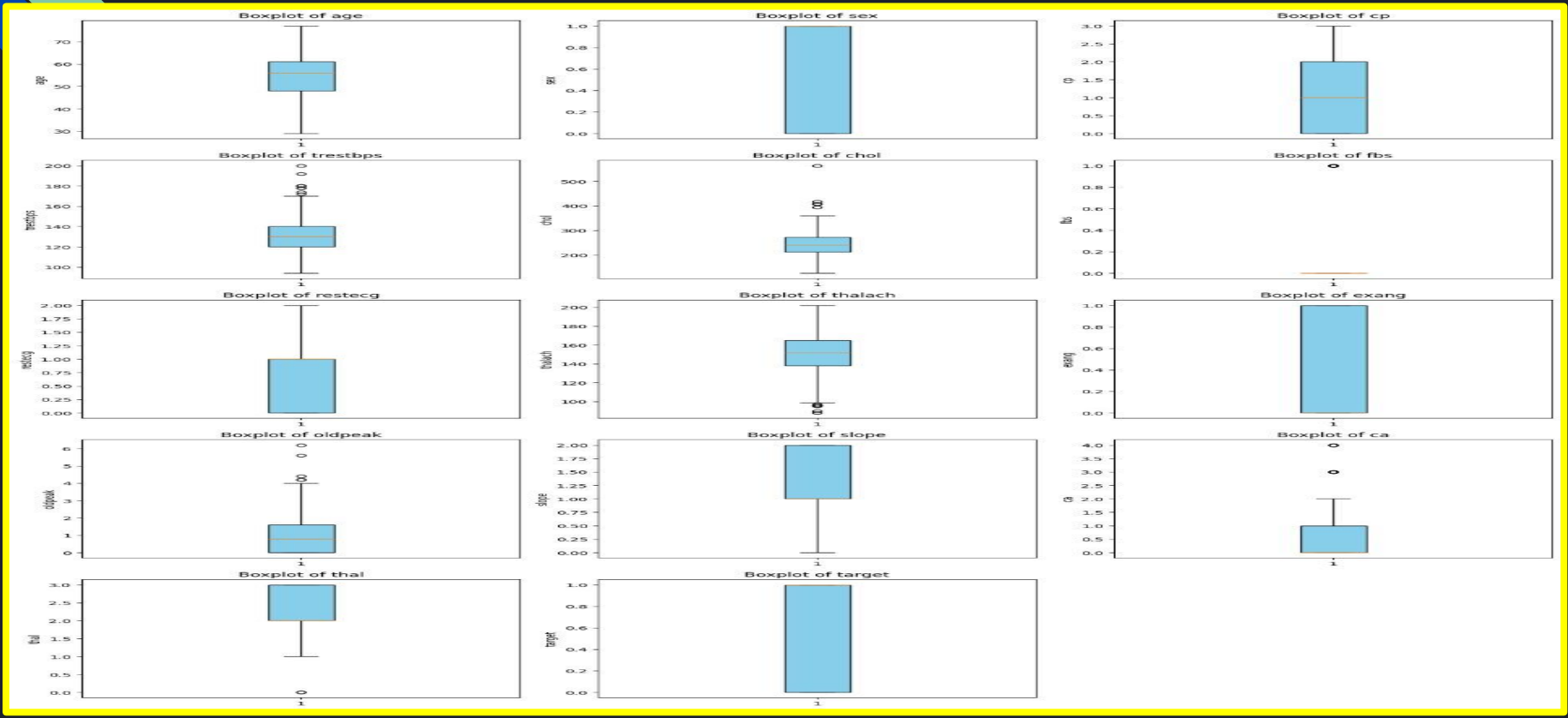
```
print(df.dtypes)
```

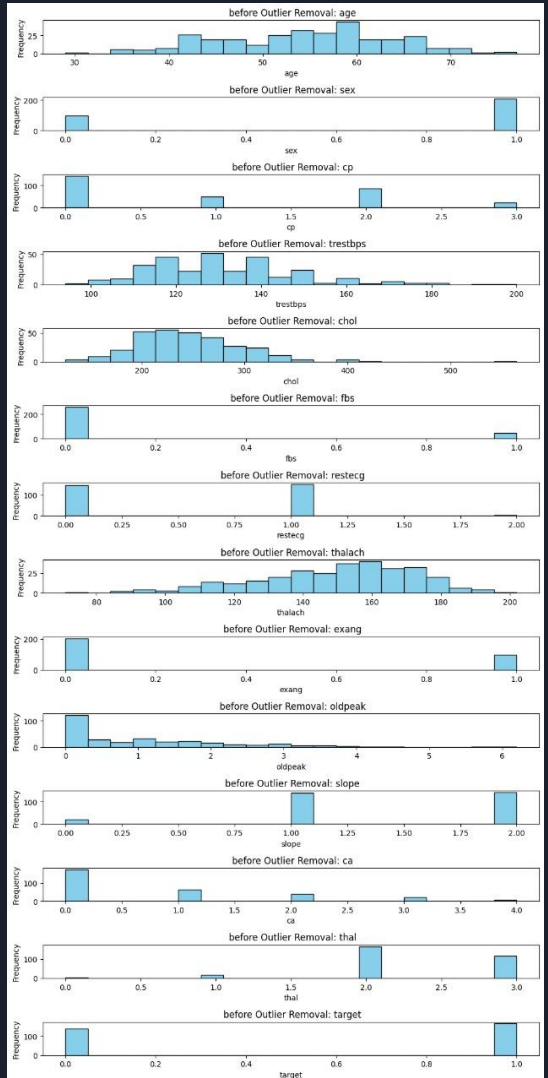
```
age         float64
sex         float64
cp          float64
trestbps    float64
chol        float64
fbs         float64
restecg     float64
thalach     float64
exang       float64
oldpeak     float64
slope       float64
ca          float64
thal        float64
target      float64
dtype: object
```

AFTER

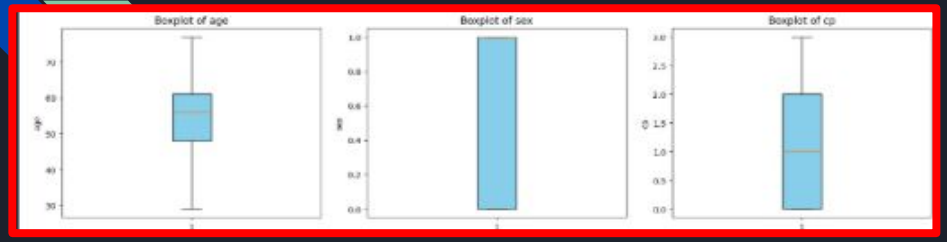
```
age         float64
sex           int64
cp            int64
trestbps    float64
chol        float64
fbs           int64
restecg       int64
thalach     float64
exang         int64
oldpeak     float64
slope         int64
ca            int64
thal          int64
target      float64
dtype: object
```

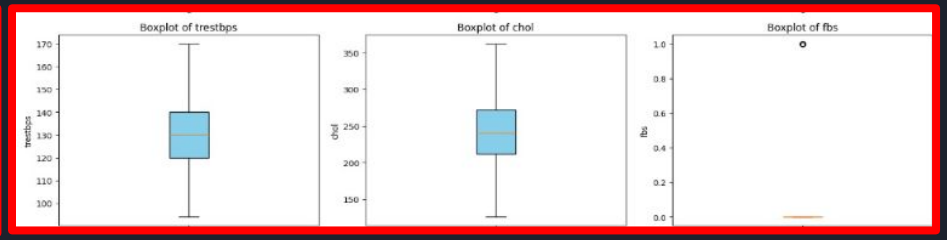
OUTLIER REMOVAL BOX PLOT(BEFORE)

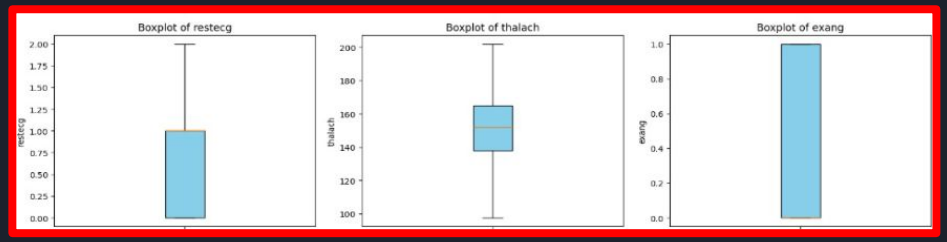


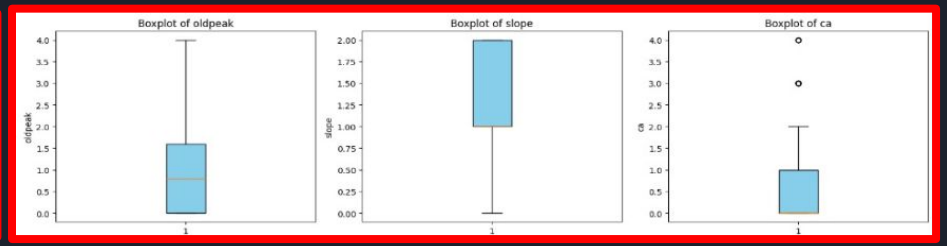

before Outlier Removal: age
Frequency
age
before Outlier Removal: sex
Frequency
sex
before Outlier Removal: cp
Frequency
cp
before Outlier Removal: trestbps
Frequency
trestbps
before Outlier Removal: chol
Frequency
chol
before Outlier Removal: fbs
Frequency
fbs
before Outlier Removal: restecg
Frequency
restecg
before Outlier Removal: thalach
Frequency
thalach
before Outlier Removal: exang
Frequency
exang
before Outlier Removal: oldpeak
Frequency
oldpeak
before Outlier Removal: slope
Frequency
slope
before Outlier Removal: ca
Frequency
ca
before Outlier Removal: thal
Frequency
thal
before Outlier Removal: target
Frequency
target

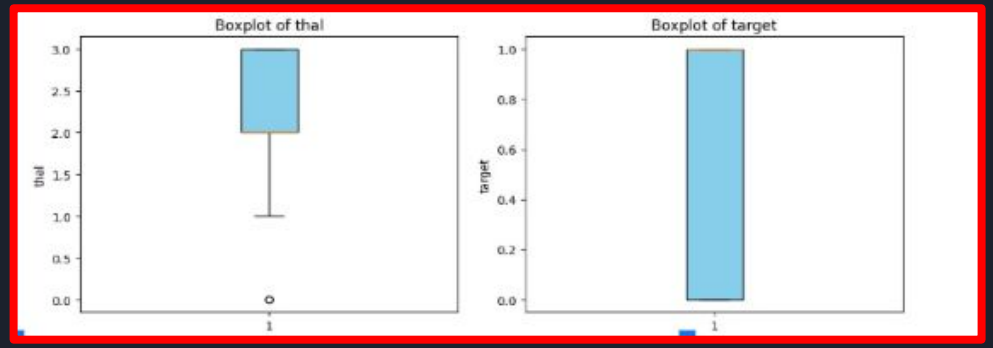
AFTER REMOVAL

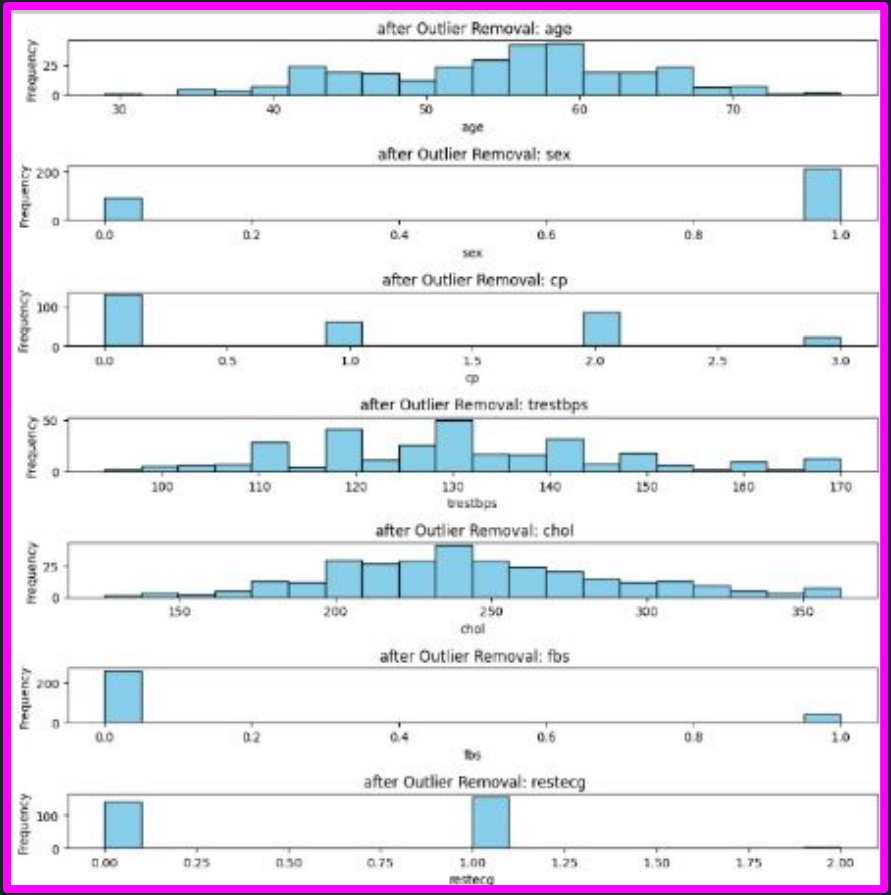


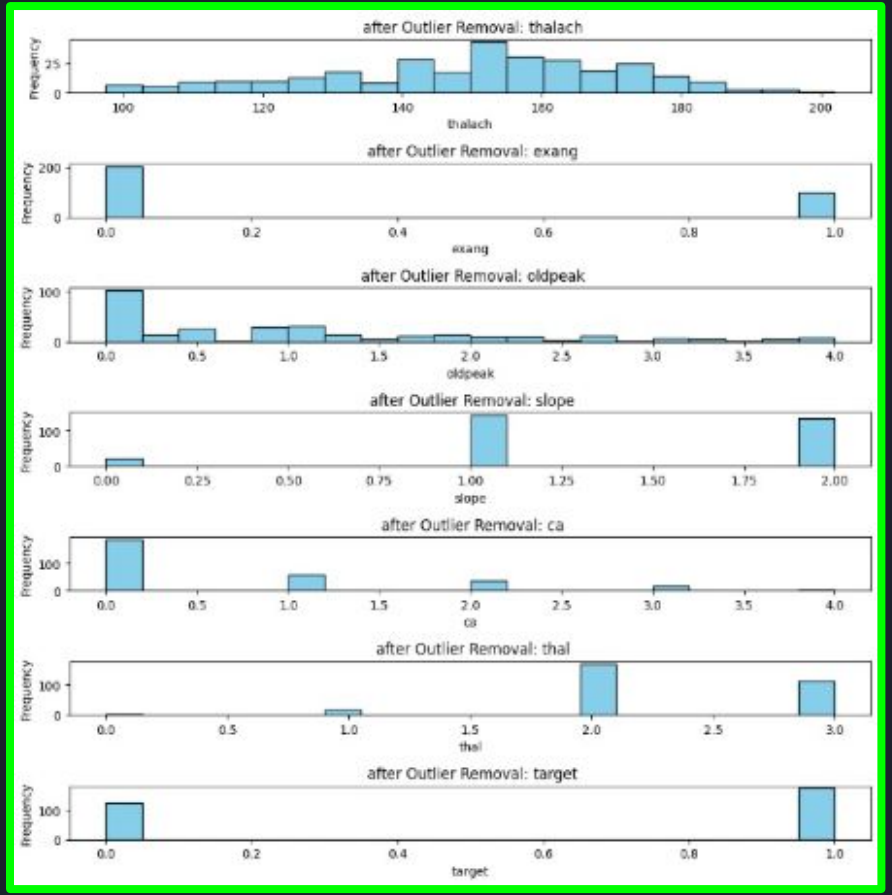




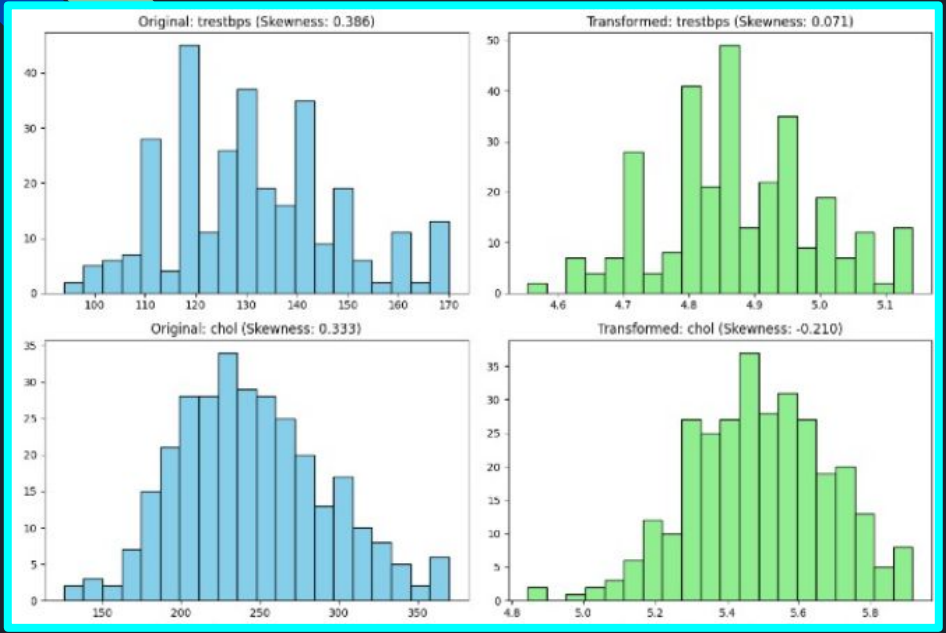


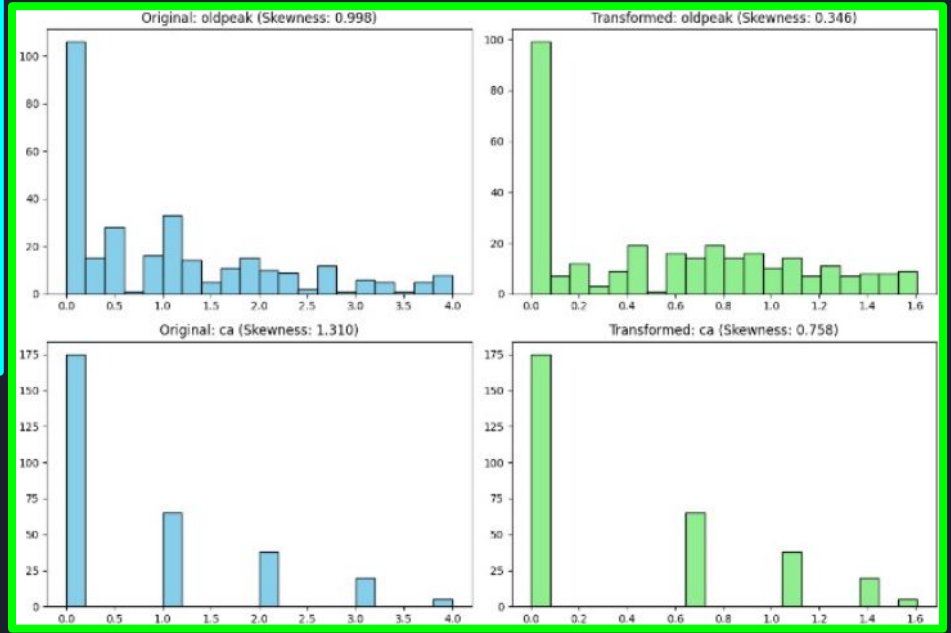



after Outlier Removal: age
Frequency
age
after Outlier Removal: sex
Frequency
sex
after Outlier Removal: cp
Frequency
cp
after Outlier Removal: trestbps
Frequency
trestbps
after Outlier Removal: chol
Frequency
chol
after Outlier Removal: fbs
Frequency
fbs
after Outlier Removal: restecg
Frequency
restecg

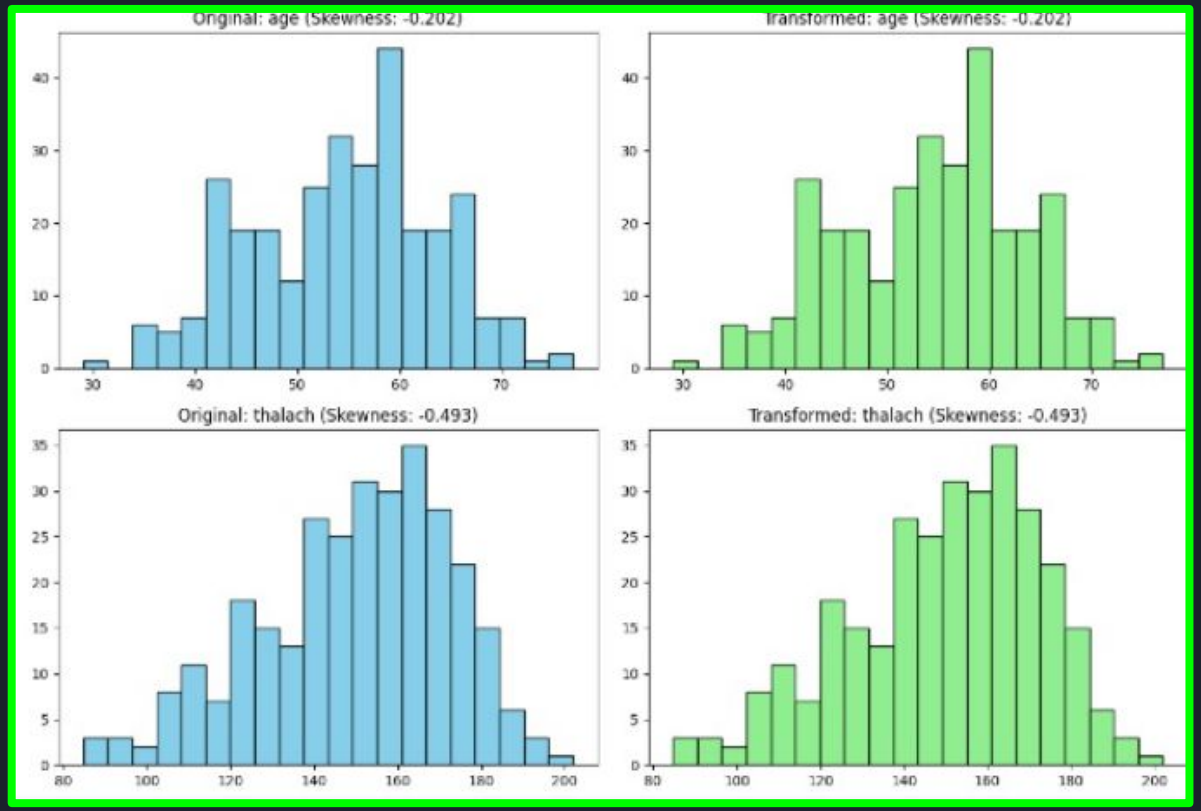

after Outlier Removal: thalach
Frequency
thalach
after Outlier Removal: exang
Frequency
exang
after Outlier Removal: oldpeak
Frequency
oldpeak
after Outlier Removal: slope
Frequency
slope
after Outlier Removal: ca
Frequency
ca
after Outlier Removal: thal
Frequency
thal
after Outlier Removal: target
Frequency
target

SKEWNESS ADDRESSING

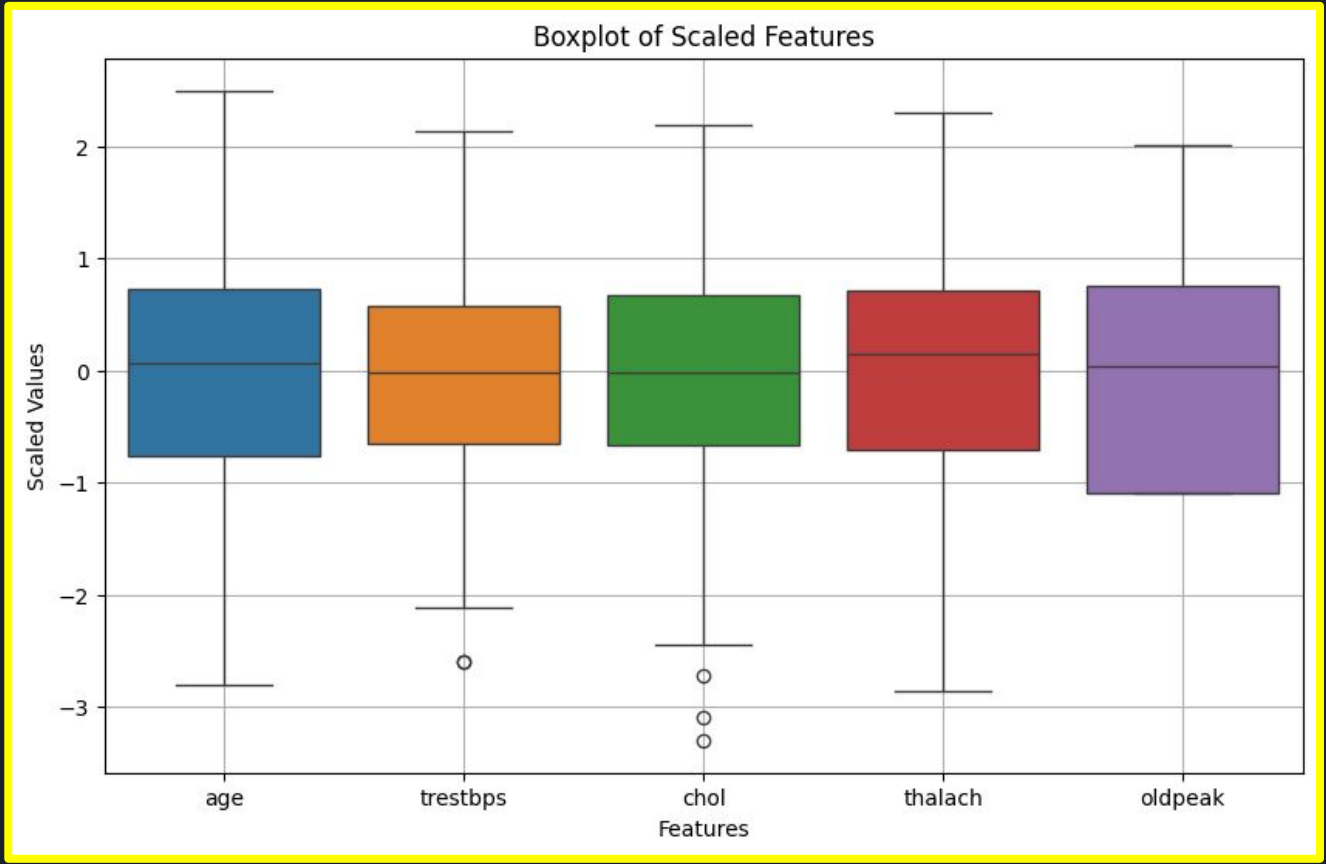




SKEWNESS ADDRESSING (CONT.)



STANDARDIZATION



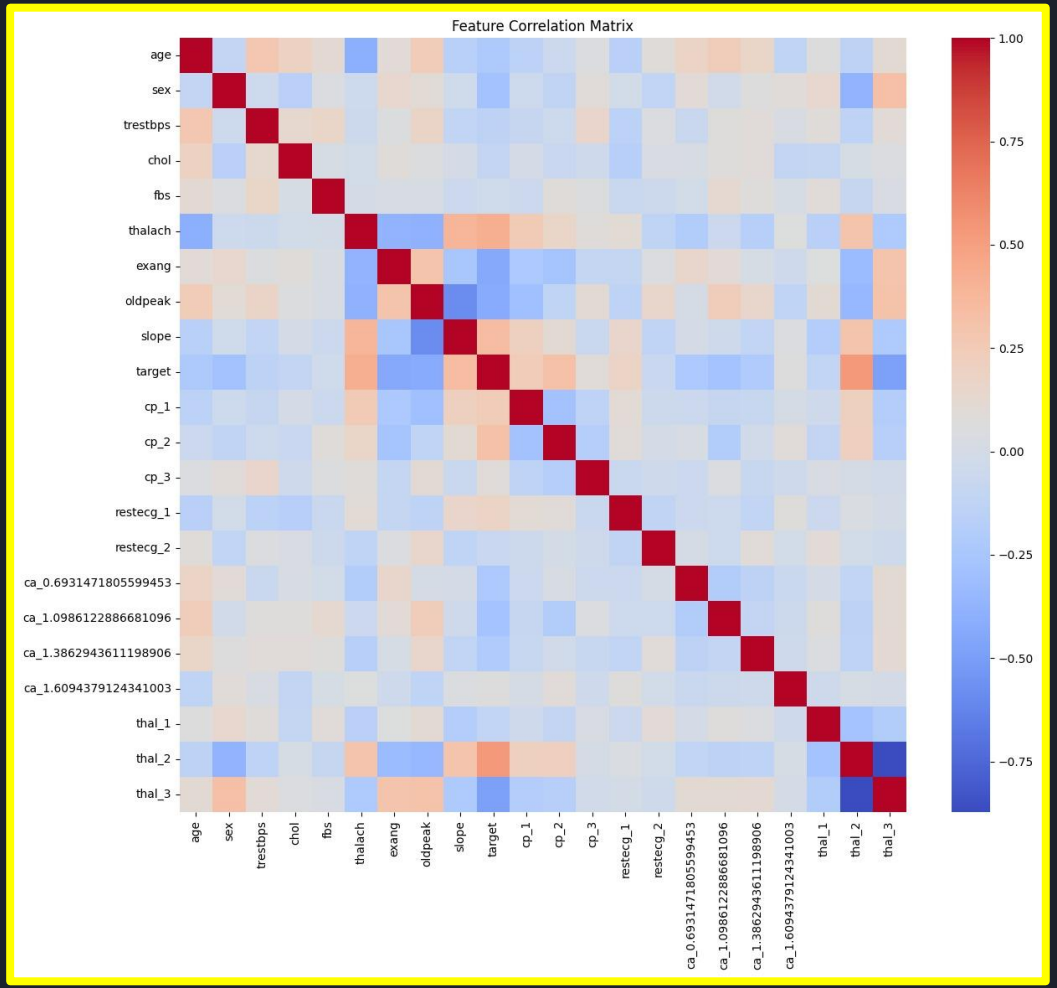
ENCODING(One Hot Encoding(Nominal) And Ordinal Encoding)

```
[ ] dfe.head()
```

	age	sex	trestbps	chol	fbs	thalach	exang	oldpeak	slope	target	...	cp_3	restecg_1	restecg_2	ca_0.6931471805599453	ca_1.0986122886681096	ca_1.3862943611198906	ca_1.6094379124341003	thal_1	thal_2	thal_3
0	0.952197	1	0.856223	-0.162807	1	0.013543	0	1.210817	0	1	...	True	False	False	False	False	False	False	True	False	False
1	-1.915313	1	-0.015954	0.196559	0	1.641748	0	1.810138	0	1	...	False	True	False	False	False	False	False	False	True	False
2	-1.474158	0	-0.015954	-0.840788	0	0.981665	0	0.595461	2	1	...	False	False	False	False	False	False	False	False	True	False
3	0.180175	1	-0.654800	-0.097530	0	1.245698	0	0.039565	2	1	...	False	True	False	False	False	False	False	False	True	False
4	0.290464	0	-0.654800	1.972919	0	0.585615	1	-0.188031	2	1	...	False	True	False	False	False	False	False	False	True	False

5 rows × 22 columns

CORRELATION MATRIX



MODEL TRAINING

ALgorithms Used

Logistic Regression

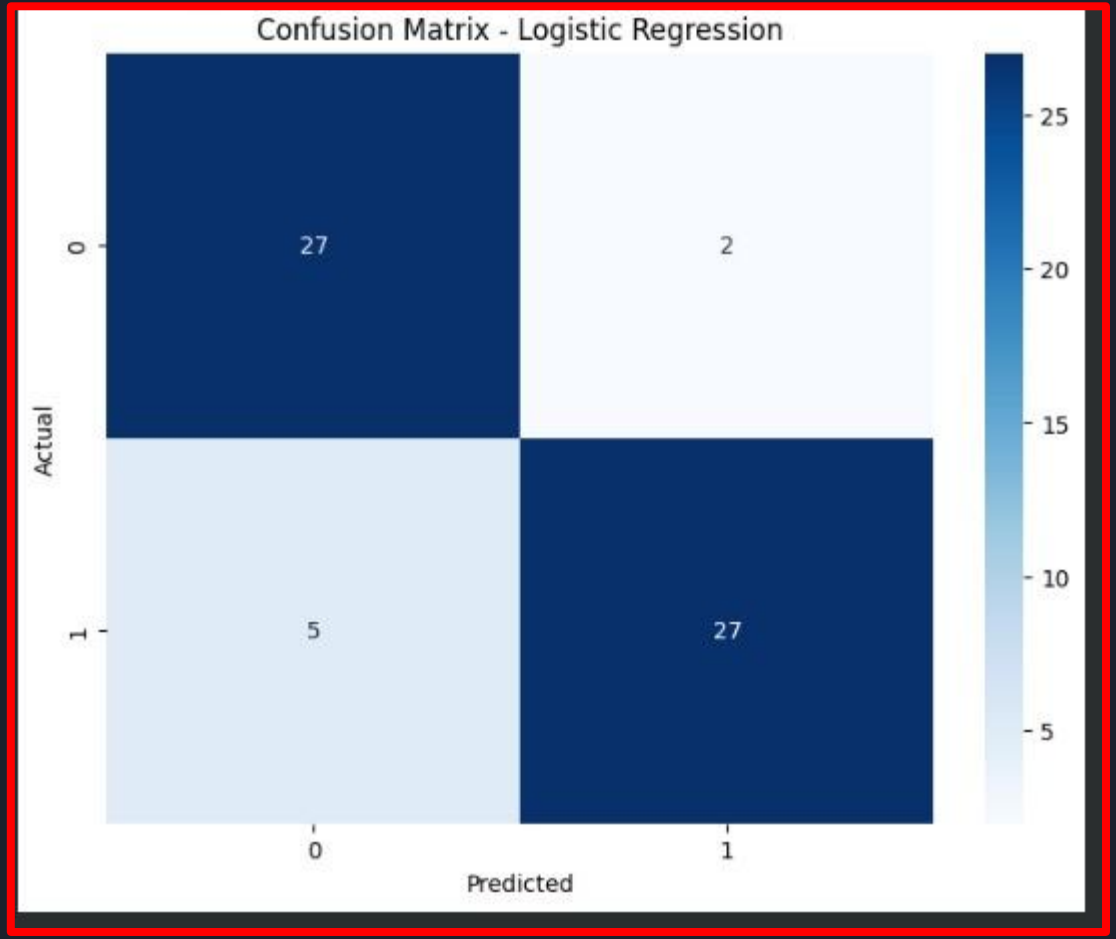
K nearest neighbour

LOGISTIC REGRESSION RESULT

Logistic Regression Performance:

	precision	recall	f1-score	support
0	0.84	0.93	0.89	29
1	0.93	0.84	0.89	32
accuracy			0.89	61
macro avg	0.89	0.89	0.89	61
weighted avg	0.89	0.89	0.89	61

ROC AUC Score: 0.9386

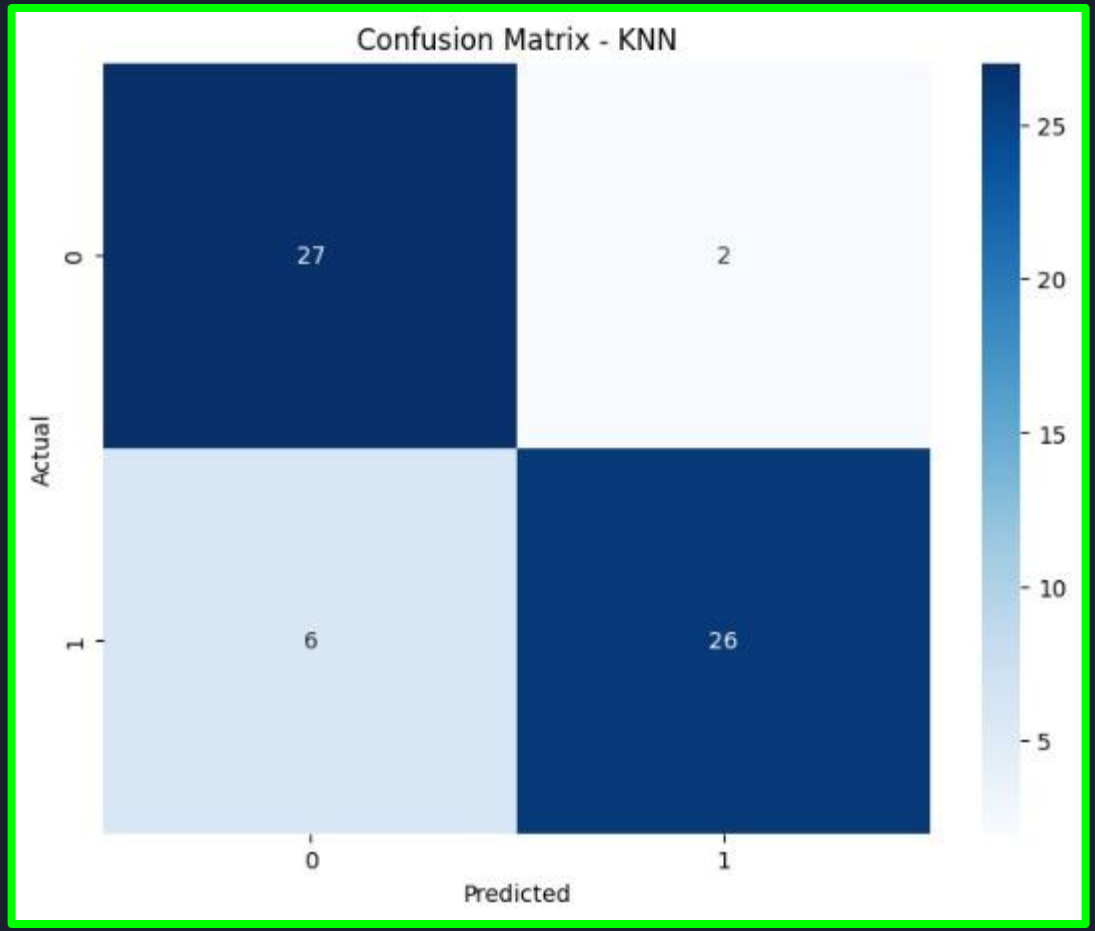

Confusion Matrix - Logistic Regression
Actual
Predicted
0
1
27
2
5
27
25
20
15
10
5

K-Nearest Neighbors (KNN) RESULT

KNN Performance:

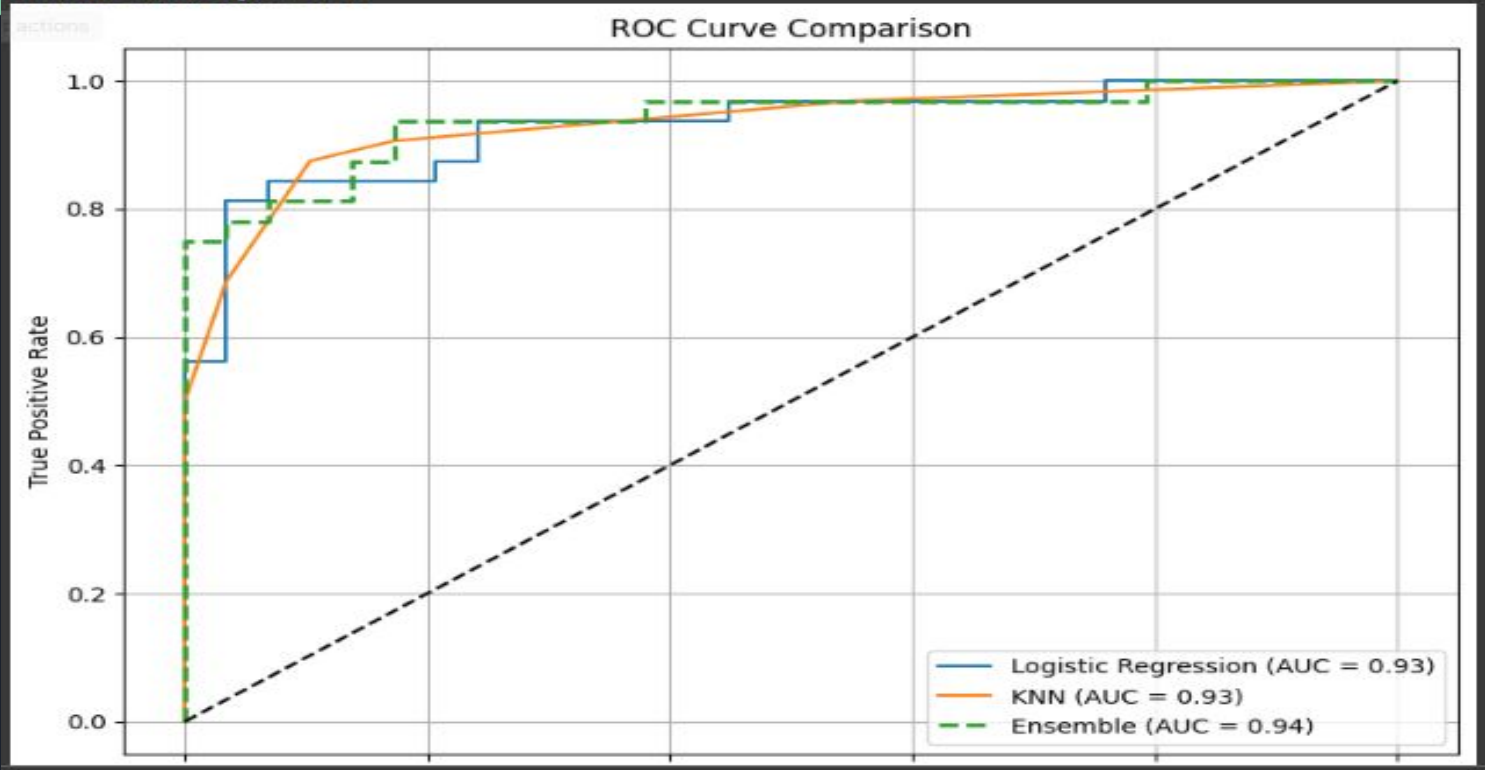
	precision	recall	f1-score	support
0	0.82	0.93	0.87	29
1	0.93	0.81	0.87	32
accuracy			0.87	61
macro avg	0.87	0.87	0.87	61
weighted avg	0.88	0.87	0.87	61

ROC AUC Score: 0.8992


Confusion Matrix - KNN
27
2
6
26
0
1
Actual
0
1
Predicted
25
20
15
10
5

ROC Curve Comparison (Logistic VS KNN)

```
=== Ensemble (Average of Logistic + KNN probabilities) ===
Ensemble Accuracy: 0.8361
```



THANK YOU

SUBMITTED BY:
VEDANT SINGH: 23/SE/169