Random Forest Classifier

In [1]:
```
#Exp no.:12
```

In [2]:
```
#Aim : Understanding Random Forest Classifier
```

In [3]:
```
#Name:Sahil A. Bankar
#Roll no:04
#Sec:B
#Subject:ET1
#Date:09/10/2025
```

Importing The Libraries

In [4]:
```
import pandas as pd
import numpy as np
```

Data Acquisition using Pandas

In [5]:
```
import os
```

In [6]:
```
os.getcwd()
```

Out[6]:
```
'C:\\Users\\DELL'
```

In [8]:
```
os.chdir('C:\\Users\\DELL\\Desktop')
```

In [9]:
```
data=pd.read_csv("heart.csv")
```

In [10]:
```
data.head()
```

Out[10]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	52	1	0	125	212	0	1	168	0	1.0	2	2	3	0
1	53	1	0	140	203	1	0	155	1	3.1	0	0	3	0
2	70	1	0	145	174	0	1	125	1	2.6	0	0	3	0
3	61	1	0	148	203	0	1	161	0	0.0	2	1	3	0
4	62	0	0	138	294	1	1	106	0	1.9	1	3	2	0

In [12]:
```
data.tail()
```

Out[12]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
1020	59	1	1	140	221	0	1	164	1	0.0	2	0	2	1
1021	60	1	0	125	258	0	0	141	1	2.8	1	1	3	0
1022	47	1	0	110	275	0	0	118	1	1.0	1	1	2	0
1023	50	0	0	110	254	0	0	159	0	0.0	2	0	2	1
1024	54	1	0	120	188	0	1	113	0	1.4	1	1	3	0

In [13]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1025 entries, 0 to 1024
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   age       1025 non-null   int64  
 1   sex       1025 non-null   int64  
 2   cp        1025 non-null   int64  
 3   trestbps  1025 non-null   int64  
 4   chol      1025 non-null   int64  
 5   fbs       1025 non-null   int64  
 6   restecg   1025 non-null   int64  
 7   thalach   1025 non-null   int64  
 8   exang     1025 non-null   int64  
 9   oldpeak   1025 non-null   float64
 10  slope     1025 non-null   int64  
 11  ca        1025 non-null   int64  
 12  thal      1025 non-null   int64  
 13  target    1025 non-null   int64  
dtypes: float64(1), int64(13)
memory usage: 112.2 KB
```

In [14]:

```
data.describe()
```

Out[14]:

	age	sex	cp	trestbps	chol	fbs	restecg	
count	1025.000000	1025.000000	1025.000000	1025.000000	1025.00000	1025.000000	1025.000000	10
mean	54.434146	0.695610	0.942439	131.611707	246.00000	0.149268	0.529756	1
std	9.072290	0.460373	1.029641	17.516718	51.59251	0.356527	0.527878	
min	29.000000	0.000000	0.000000	94.000000	126.00000	0.000000	0.000000	
25%	48.000000	0.000000	0.000000	120.000000	211.00000	0.000000	0.000000	1
50%	56.000000	1.000000	1.000000	130.000000	240.00000	0.000000	1.000000	1
75%	61.000000	1.000000	2.000000	140.000000	275.00000	0.000000	1.000000	1
max	77.000000	1.000000	3.000000	200.000000	564.00000	1.000000	2.000000	2

In [15]:

```
data.shape
```

Out[15]:

```
(1025, 14)
```

In [16]:

```
data.size
```

Out[16]:

```
14350
```

In [17]:

```
data.ndim
```

Out[17]:

```
2
```

Data preprocessing *data cleaning* missing value treatment

In [18]:

```
# check Missing Value by record
data.isna()
```

Out[18]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal
0	False	False	False	False	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...	...	...	...	...	...
1020	False	False	False	False	False	False	False	False	False	False	False	False	False
1021	False	False	False	False	False	False	False	False	False	False	False	False	False
1022	False	False	False	False	False	False	False	False	False	False	False	False	False
1023	False	False	False	False	False	False	False	False	False	False	False	False	False
1024	False	False	False	False	False	False	False	False	False	False	False	False	False

1025 rows × 14 columns

In [19]:

```
data.isna().any()
```

Out[19]:

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

Independent and Dependent Variables

In [20]:

```
x=data.drop("target", axis=1)
y=data["target"]
```

Splitting of DataSet into train and Test

In [21]:

```
# Splitting the data into training and testing data sets
from sklearn.model_selection import train_test_split

x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_stat
```

Random Forest Classifier

In [22]:

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
```

In [23]:

```
rf=RandomForestClassifier()
```

In [24]:

```
rf.fit(x_train, y_train)
```

Out[24]:

```
RandomForestClassifier()
```

In [25]:

```
y_pred5=rf.predict(x_test)
```

In [26]:

```
accuracy_score (y_test,y_pred5)
```

Out[26]:

```
1.0
```

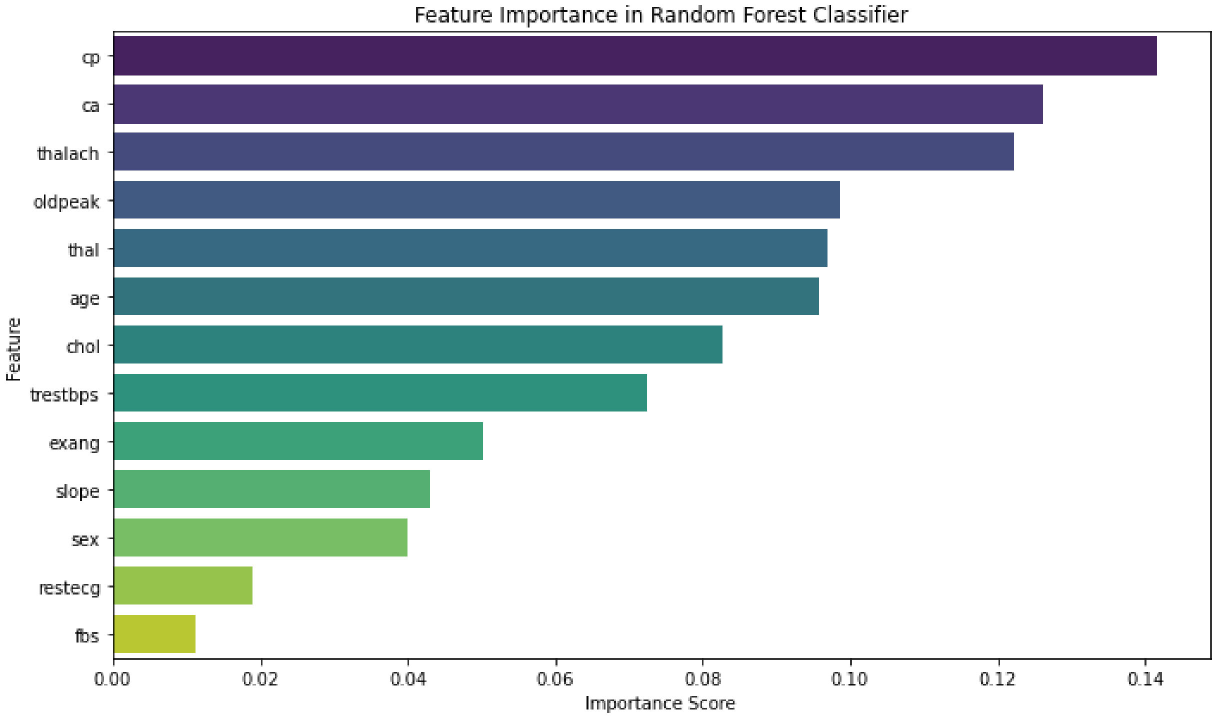
In [28]:

```
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
from sklearn.metrics import confusion_matrix, roc_curve, auc
from sklearn.tree import plot_tree

# Get feature importances from your trained Random Forest model
importances = rf.feature_importances_
features = np.array(x.columns)

# Sort feature importances in descending order
indices = np.argsort(importances)[::-1]

# Plot
plt.figure(figsize=(10, 6))
sns.barplot(x=importances[indices], y=features[indices], palette="viridis", dodge=Fa
plt.title("Feature Importance in Random Forest Classifier")
plt.xlabel("Importance Score")
plt.ylabel("Feature")
plt.tight_layout()
plt.show()
```



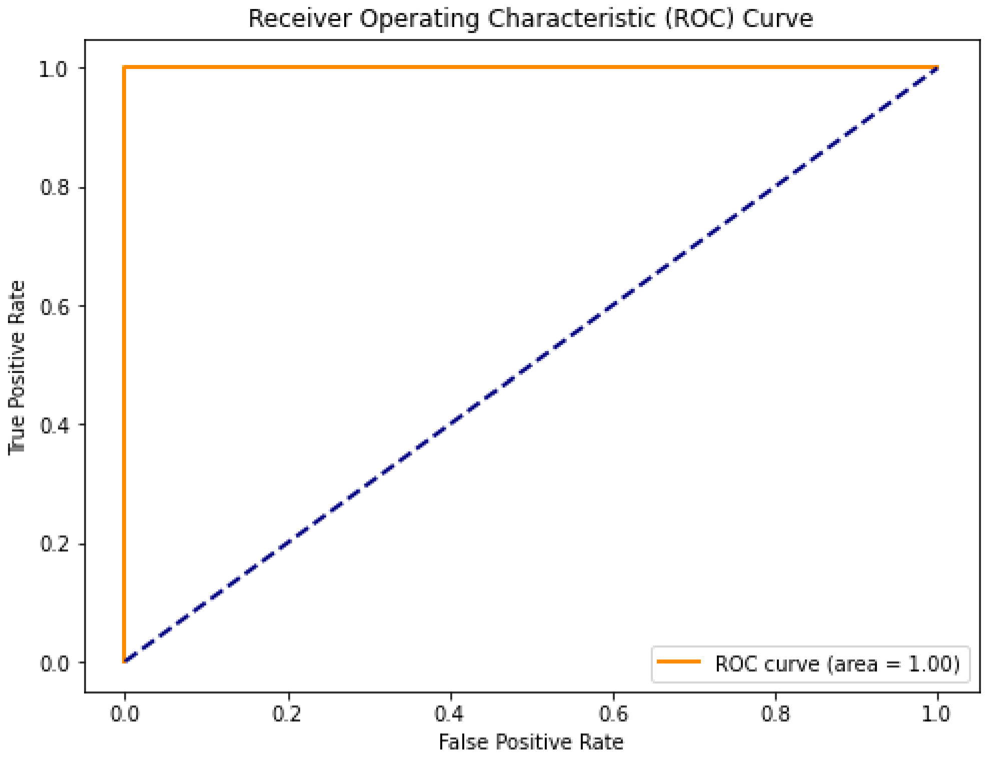
In [29]:

```
from sklearn.metrics import roc_auc_score, roc_curve, auc
import matplotlib.pyplot as plt

# Predict probabilities for ROC curve
y_prob = rf.predict_proba(x_test)[:, 1]

# Compute ROC curve and AUC score
fpr, tpr, thresholds = roc_curve(y_test, y_prob)
roc_auc = auc(fpr, tpr)

# Plot ROC curve
plt.figure(figsize=(8, 6))
plt.plot(fpr, tpr, color='darkorange', lw=2, label=f'ROC curve (area = {roc_auc:.2f}
plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC) Curve')
plt.legend(loc="lower right")
plt.show()
```



Conclusion :

The experiment successfully implemented the Random Forest algorithm, demonstrating its robustness and improved accuracy through ensemble learning. This

In [ ]: