DSC540 Project ML

Gerardo Palacios

In [1]:
```
import pandas as pd
import numpy as np
data = pd.read_csv('heart.csv')  # Read dataset
```

Data Description

For my final project I will be working with a heart failure dataset. It was retrieved from Kaggle.com https://www.kaggle.com/fedesoriano/heart-failure-prediction. It is a small dataset containing 918 observations, 11 unique associated features and a target class.

In [2]:
```
print('This dataset has {} observations and {} features'.format(*data.shape)) # Describe shape
```

```
This dataset has 918 observations and 12 features
```

It is a very clean dataset with no missing values.

In [3]:
```
data.isna().sum()
```

Out[3]:
```
Age              0
Sex              0
ChestPainType    0
RestingBP        0
Cholesterol      0
FastingBS        0
RestingECG       0
MaxHR            0
ExerciseAngina   0
Oldpeak          0
ST_Slope         0
HeartDisease     0
dtype: int64
```

In [4]:
```
data['FastingBS'] = data['FastingBS'].astype('object') # Convert variable to correct type
data['HeartDisease'] = data['HeartDisease'].astype('object') # Convert variable to correct type
```

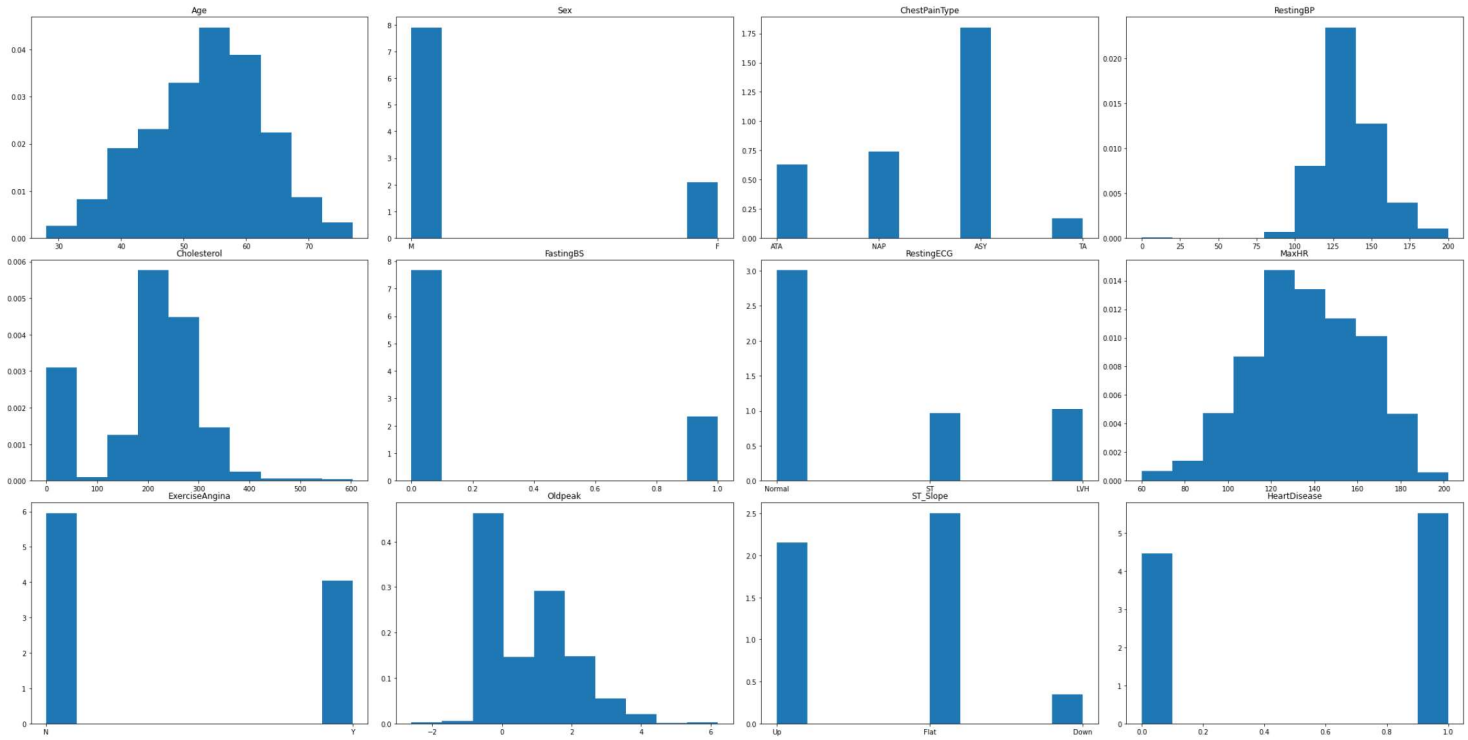
The features included are shown below. It is a slightly unbalanced dataset with the target class having a 55%/44% split between having and not having heart disease. The dataset includes mainly male patients with varying vital measurements and accompanying cardiac symptoms.

Feature	Description	Mean/Most Frequent	Standard Deviation\Ratio
Heart Disease	Target class, 1: Heart Disease 0: Normal	Heart Disease	Heart Disease: 55%, Normal: 44%
Age	Age of the patient	53.5	9.43
RestingBP	Resting blood pressure [mm Hg]	132.39	18.51
Cholesterol	Serum cholesterol	198.79	109.38
Fasting BS	Fasting blood sugar [1: if FastingBS > 120 mg/dl, 0: otherwise]	Normal	0: 76%, 1: 23%
Max HR	Maximum heart rate achieved	136.80	25.46
OldPeak	Numeric value measured in depression	0.88	1.06
Sex	Sex of the patient	Male	M:78%, F:21%
ChestPain Type	Chest pain type [TA: Typical Angina, ATA: Atypical Angina, NAP: Non-Anginal Pain, ASY: Asymptomatic]	Asymptomatic	TA:5%, ATA: 18%, NAP:22%, ASY: 54%
RestingECG	Resting ECF 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]	Normal	Normal: 60%, ST: 19%, LVH: 20%
Excercise Angina	If patient has exercised-induced angina [Y: Yes, N: No]	No	Yes: 40%, No: 59%
ST_Slope	The slope of the peak exercise ST segment [Up: upsloping, Flat: flat, Down: downsloping]	Flat	Up: 43%, Down: 06%, Flat: 50%

The distributions of the features are shown below. Age is the only variable that appears to be nearly normal, while the remaining 4 numerical variables appear to be skewed. All the categorical variables are unbalanced.

In [5]:

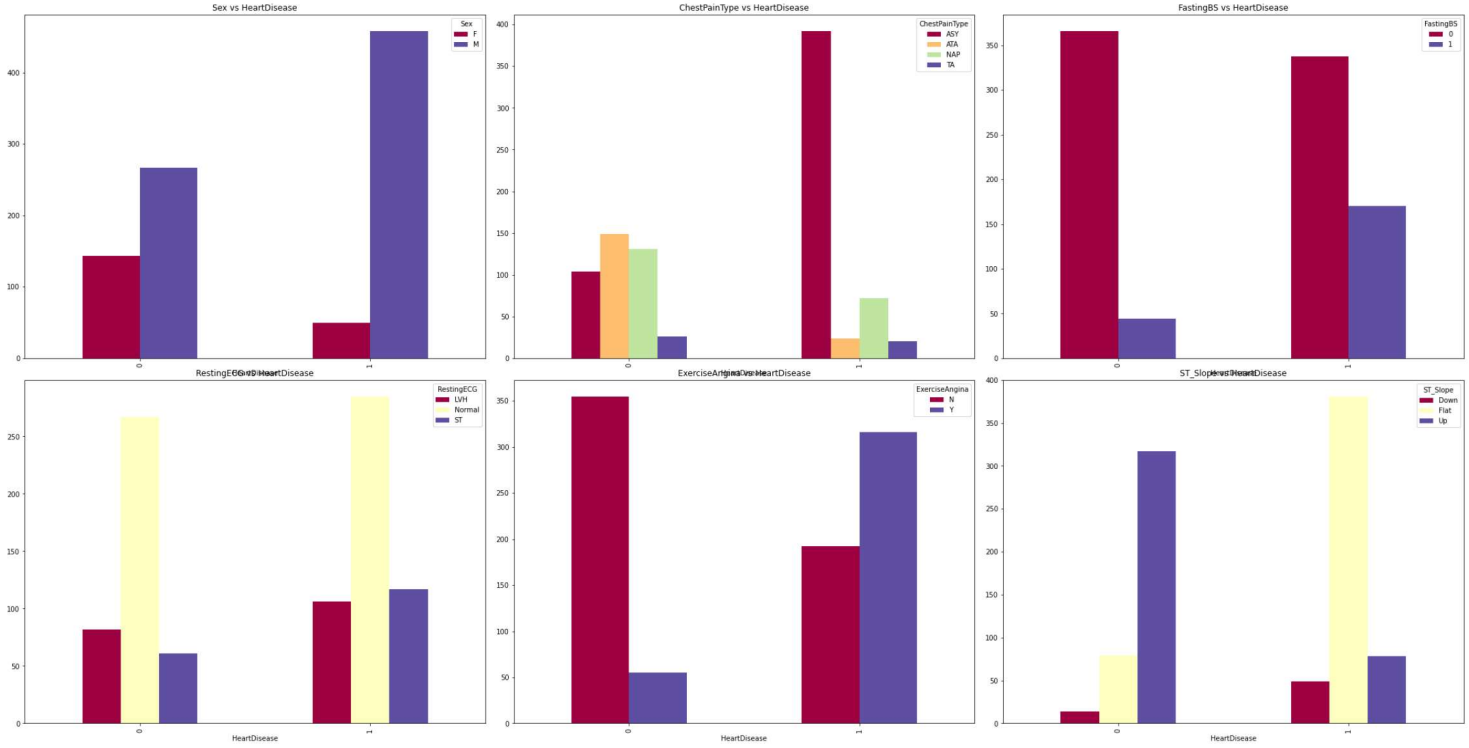
```
import matplotlib.pyplot as plt

rows, columns = 3,4
# Show the distribution for all attributes
fig, axs = plt.subplots(rows,columns, figsize=(30,15))   # initialize 4 by 6 plot grid
fig.tight_layout()                                       # set layout
column_names = [column for column in data]               # Get column names into list
i = 0
for x in range(rows):                                    # Fill plot grid
    for y in range(columns):
        if i < data.shape[1]:
            axs[x,y].hist(data[column_names[i]], density=True)
            axs[x,y].set(title=column_names[i])
            i += 1
```



The following are cross-tabulations between the categorical variables and heart disease. Most of the cross-tabulations do not appear significant. Some of the most alarming are the cross-tabulations between heart disease and chest pain type. There is an alarming majority of patients that were asymptomatic but had heart disease. In addition, there appears to be some relationship between heart disease and having or not having exercise angina as well as having a flat ST Slope. However, that also may be due to the unbalance nature of both of those features.

In [6]:

```
from scipy.stats import chi2_contingency

# Show the cross-tabulations for all categorical attributes with the target class
rows, columns = 2, 3
fig, axs = plt.subplots(rows,columns, figsize=(30,15))   # initialize plot grid
fig.tight_layout()                                       # set layout
i = 0
chi_squared_correlation = []
for x in range(rows):                                       # Fill plot grid
    for y in range(columns):
        if i < data.select_dtypes('object').shape[1] and data.select_dtypes('object').columns[i] != 'HeartDisease':
            cross_tabulation = pd.crosstab(data['HeartDisease'],data.select_dtypes('object').iloc[:,i])
            cross_tabulation.plot.bar(ax=axs[x,y],colormap='Spectral',
                                      title='{} vs HeartDisease'.format(data.select_dtypes('object').columns[i]))
            chi2, p, dof, expected = chi2_contingency(cross_tabulation)
            chi_squared_correlation.append(('{} vs HeartDisease'.format(data.select_dtypes('object').columns[i]),p))

            i +=1
```



However, despite the seemingly unremarkable cross-tabulations, the chi-squared test between the categorical variables and the target class all reject the null hypothesis that the variables are uncorrelated and can accept that the features are in correlated. As shown below, the p-values have 99% confidence that the features are correlated.

In [7]:
```
print(pd.DataFrame(chi_squared_correlation,columns=['Feature Pairs','P-value']))
```

```
                  Feature Pairs       P-value
0            Sex vs HeartDisease  4.597617e-20
1   ChestPainType vs HeartDisease  8.083728e-58
2       FastingBS vs HeartDisease  1.057302e-15
3      RestingECG vs HeartDisease  4.229233e-03
4  ExerciseAngina vs HeartDisease  2.907808e-50
5        ST_Slope vs HeartDisease  5.167638e-78
```

I can compute the point bi-serial correlation to measure the relationship between the continuous variables and the target class. As shown below, all the continuous features appear to have a moderately strong determinant relationship with having heart disease. All of the factors appear significant with heart disease.

In [8]:
```
from scipy.stats import pointbiserialr
biserialcorr = []
i = 0
while i < data.select_dtypes('number').shape[1]:
            biserialcorr.append(('{} vs HeartDisease'.format(data.select_dtypes('number').columns[i]),
                                 *pointbiserialr(data['HeartDisease'], data.select_dtypes('number').iloc[:,i])))
            i +=1
```

In [9]:
```
print(pd.DataFrame(biserialcorr,columns=['Feature Pairs','Bi-Serial Correlations','P-Value']))
```

```
                Feature Pairs  Bi-Serial Correlations       P-Value
0         Age vs HeartDisease                0.282039  3.007953e-18
1   RestingBP vs HeartDisease                0.107589  1.095315e-03
2  Cholesterol vs HeartDisease               -0.232741  9.308309e-13
3       MaxHR vs HeartDisease               -0.400421  1.137786e-36
4   Oldpeak vs HeartDisease                0.403951  2.390772e-37
```

Pre-Processing

In order to prepare the data for classification I'll need to perform a few tasks. First I'll need to encode the categorical variables, and then I have to standardize the dataset. Encoding the categorical variables can be done in two different ways. It can be done using one-hot encoding which converts each categorical variable as its own feature and encodes it with a binary value of 1 or 0. This is done for each categorical variable resulting in a higher-dimensional dataset. The second way is by way of Weight of evidence. Weight of evidence tells the predictive power of an independent categorical variable in relation to the dependent categorical variable. It transforms a categorical feature into a continuous feature in

relation to the dependent variable. In this project, I will experiment with both methodologies and create and compare models in between both sets.

In [10]:
```
from category_encoders.woe import WOEEncoder
data['FastingBS'] = data['FastingBS'].astype('int') # Convert variable to correct type
data['HeartDisease'] = data['HeartDisease'].astype('int') # Convert variable to correct type

y = data.pop('HeartDisease')

# Method 1: One-Hot Encoding
data1 = pd.get_dummies(data)
data2 = data.copy()

# Method 2: Weight of Evidence
columns = [col for col in data2.select_dtypes('object') ]
woe_encoder = WOEEncoder(cols=columns)

data2[columns] = woe_encoder.fit_transform(data2[columns] ,y)
```

In [11]:
```
data1.head()
```

Out[11]:

	Age	RestingBP	Cholesterol	FastingBS	MaxHR	Oldpeak	Sex_F	Sex_M	ChestPainType_ASY	ChestPainType_ATA	ChestPainType_NAP	ChestPainTy
0	40	140	289	0	172	0.0	0	1	0	1	0	
1	49	160	180	0	156	1.0	1	0	0	0	1	
2	37	130	283	0	98	0.0	0	1	0	1	0	
3	48	138	214	0	108	1.5	1	0	1	0	0	
4	54	150	195	0	122	0.0	0	1	0	0	1	

In [12]:
```
data2.head()
```

Out[12]:

	Age	Sex	ChestPainType	RestingBP	Cholesterol	FastingBS	RestingECG	MaxHR	ExerciseAngina	Oldpeak	ST_Slope
0	40	0.324676	-2.005147	140	289	0	-0.148383	172	-0.825628	0.0	-1.605991
1	49	-1.251375	-0.805730	160	180	0	-0.148383	156	-0.825628	1.0	1.350007
2	37	0.324676	-2.005147	130	283	0	0.430163	98	-0.825628	0.0	-1.605991
3	48	-1.251375	1.106462	138	214	0	-0.148383	108	1.520163	1.5	1.350007
4	54	0.324676	-0.805730	150	195	0	-0.148383	122	-0.825628	0.0	-1.605991

Next I will split the data into training and test and transform the data by applying MinMax Transformation. This is done on both sets of data.

In [13]:
```
from sklearn.model_selection import train_test_split
```

In [14]:
```
# Split data into training and testing
X1_train, X1_test, y1_train, y1_test = train_test_split(data1, y, test_size=0.20, random_state=32)
X2_train, X2_test, y2_train, y2_test = train_test_split(data2, y, test_size=0.20, random_state=32)
```

In [15]:
```
from sklearn.preprocessing import MinMaxScaler
# Standardize the dataset
ss = MinMaxScaler()
ss.fit(X1_train)
X1_train = ss.transform(X1_train)
X1_test = ss.transform(X1_test)

ss = MinMaxScaler()
ss.fit(X2_train)
X2_train = ss.transform(X2_train)
X2_test = ss.transform(X2_test)
```

Modeling

Finally, I can begin modeling. In order to compare across models I will be using accuracy and AUC as the comparable performance metric. In order to be thorough, in addition to experimenting with training and tests set, I will be further comparing using 5-fold cross validation using the entire dataset.

In [16]:

```
from sklearn import metrics
from sklearn.model_selection import RandomizedSearchCV
from sklearn.model_selection import StratifiedKFold
import seaborn as sns

# Initialize 5-fold validation
cv = StratifiedKFold(n_splits=5)

def plot_roc(predictions, y_actuals, ax):
    fpr, tpr, threshold = metrics.roc_curve(y_actuals, predictions) # Calculate fpr and tpr
    roc_auc = metrics.auc(fpr, tpr) # Calculate auc
    # Plot ROC
    ax.set_title('Receiver operating characteristic',fontweight="bold", size=20)
    ax.plot(fpr, tpr, 'b', label = 'AUC = {:.2f}'.format(roc_auc))
    ax.legend(loc = 'lower right')
    ax.plot([0, 1], [0, 1],'r--')
    ax.set_xlim([0, 1])
    ax.set_ylim([0, 1])
    ax.set_ylabel('True Positive Rate')
    ax.set_xlabel('False Positive Rate')

def plot_roc_cv(fold_results,ax):
    mean_tpr = 0.0
    mean_fpr = np.linspace(0, 1, 100)
    cv = 0
    for fold in fold_results:
        fpr, tpr = fold['fpr_tpr']
        mean_tpr += np.interp(mean_fpr, fpr, tpr)
        mean_tpr[0] = 0.0


        ax.plot(fpr, tpr, lw=1, label='ROC fold %d (area = %0.2f)' % (fold['fold'], fold['roc']))
        cv += 1

    mean_tpr /= cv
    mean_tpr[-1] = 1.0
    mean_auc = metrics.auc(mean_fpr, mean_tpr)
    ax.plot(mean_fpr, mean_tpr, 'k--',
            label='Mean ROC (area = %0.2f)' % mean_auc, lw=2)
    ax.plot([0, 1], [0, 1], '--', color=(0.6, 0.6, 0.6), label='Luck')
    ax.set_xlim([-0.05, 1.05])
    ax.set_ylim([-0.05, 1.05])
    ax.set_xlabel('False Positive Rate')
    ax.set_ylabel('True Positive Rate')
    ax.set_title('Receiver operating characteristic',fontweight="bold", size=20)

    ax.legend(loc="lower right")

def model_cv(model, X, y, n_splits=5):
    fold_results = []
    i = 1
    for train, test in cv.split(X,y):
        probas_ = model.fit(X.iloc[train], y.iloc[train]).predict_proba(X.iloc[test])
        # Compute ROC curve and area the curve
        fpr, tpr, thresholds = metrics.roc_curve(y.iloc[test], probas_[:, 1])
        roc_auc = metrics.auc(fpr, tpr)
        prediction = model.predict(X.iloc[test])
        accuracy = metrics.accuracy_score(y_true = y.iloc[test], y_pred = prediction)
        fold_results.append({'fold':i, 'fpr_tpr':(fpr,tpr),'accuracy':accuracy,'roc':roc_auc})
        i += 1
    return fold_results


def confusion_matrix(y, predictions, y_train, y_pred, mean, ax):
    cm = metrics.confusion_matrix(y, predictions) # Create confusion matrix
    sns.heatmap(cm, annot=True, fmt=".3f", linewidths=.5, square = True, cmap = 'Blues_r',ax=ax)  # use seaborn heatmap
    ax.set_ylabel('Actual label'); # Add y label
    ax.set_xlabel('Predicted label');  # Add x label
    ax.set_title('5-Fold Accuracy Score: {:.2f}%\nTest Accuracy Score: {:.2f}%\nTrain Accuracy Score: {:.2f}%'.format(
        mean*100,
        metrics.accuracy_score(y_true = y, y_pred = predictions)*100,
        metrics.accuracy_score(y_true = y_train, y_pred = y_pred)*100),fontweight="bold", size=20)


def build_model(model, X_train, y_train,X_test, y_test, X, y, n_folds=5,params=None,label=None):
```

```
    model_type = type(model).__name__
    if params is not None:
         model = RandomizedSearchCV(estimator=model,param_distributions=params ,refit=True,
                                    n_iter=100,scoring='roc_auc',n_jobs=-1,cv=cv,random_state=32)
         model_type = '{} {}'.format(type(model).__name__, type(model.estimator).__name__)

    model.fit(X_train,y_train)
    train_predict = model.predict(X_train)
    test_predict = model.predict(X_test)
    fold_results = model_cv(model,X, y)
    fig, axs = plt.subplots(2,2,figsize=(15,15))
    fig.suptitle('{} {}'.format(model_type,label), fontsize=25)
    mean_accuracy = np.mean([fold['accuracy'] for fold in fold_results])
    plot_roc_cv(fold_results,ax=axs[0,0])
    confusion_matrix(y_test,test_predict,y_train, train_predict,mean_accuracy,ax=axs[0,1])
    plot_roc(test_predict,y_test,ax=axs[1,0])
    plot_distribution(model.predict_proba(X_test)[:, 1], ax=axs[1, 1])
    print_metrics(y_test,test_predict)

def plot_distribution(probabilities, ax):
    sns.histplot(probabilities,ax=ax)
    ax.set_title('Probability Distribution',fontweight="bold", size=20)

def print_metrics(y_val, predictions):
    print(metrics.classification_report(y_val, predictions))
    cm = metrics.confusion_matrix(y_val, predictions) # Create confusion matrix
    TN,FP,FN,TP = cm.ravel()
    # Calculate Specificity
    print('Specificity = {}'.format(TN/(TN+FP)))
    # Calculate Balanced Accuracy
    print('The balanced accuracy score is : {:.2f}%'.format(metrics.balanced_accuracy_score(y_val,predictions)*100))
```

Logistic Regression

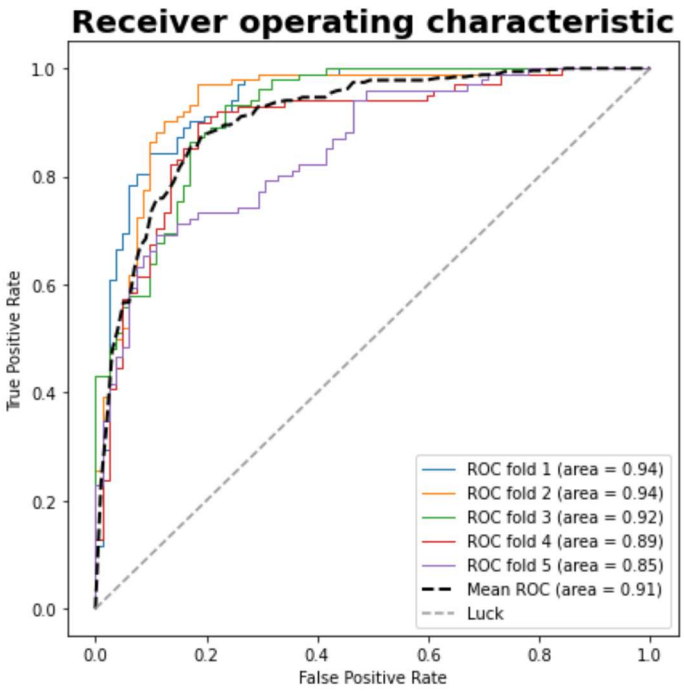
The logistic regression models performed fairly well and will be considered the base model. The logistic regression was trained on the one-hot encoding dataset and the WoE dataset (data1 and data2 respectively). In addition, a randomized search cv was used to do hyper-parameter tuning. As you can see from the following results, all the models scored above 84% accuracy and had mean 5-fold AUC score of at least 0.90. The iterations show little variance and tend to have a slight bias towards heart disease patients as shown by the confusion matrix and the probability distributions.

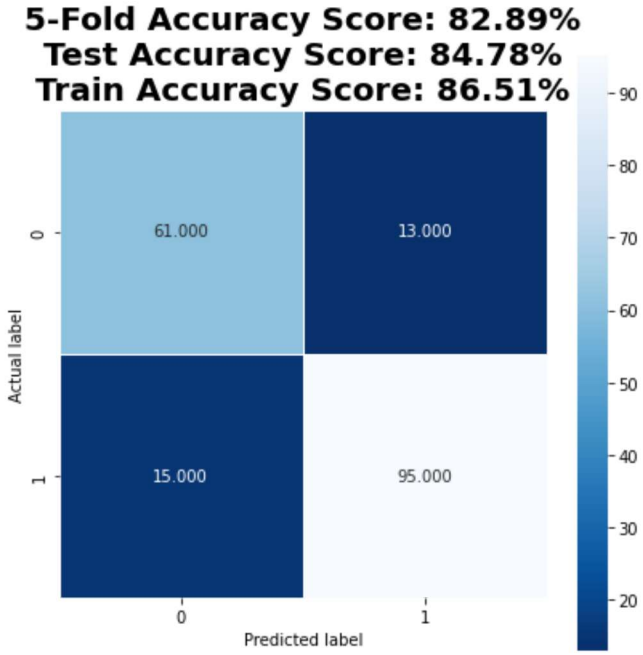
In [17]:

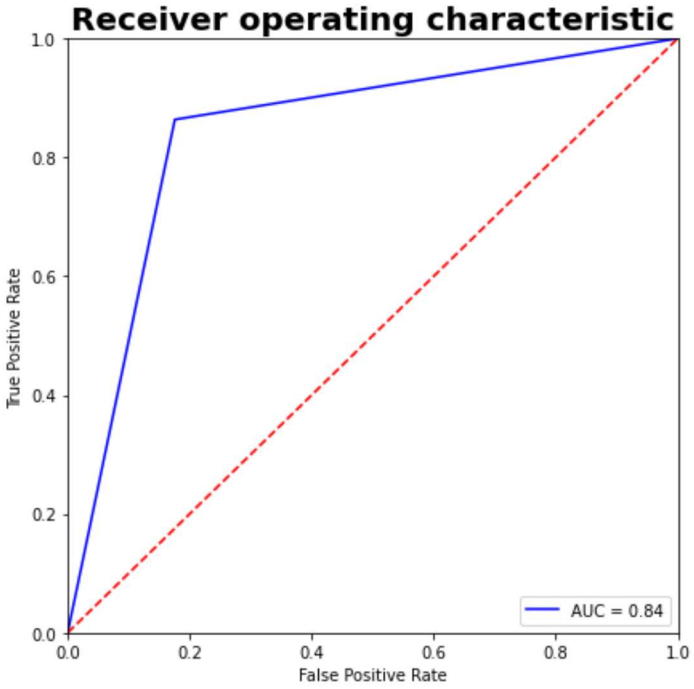
```
from sklearn.linear_model import LogisticRegression

log_reg = LogisticRegression(max_iter=1000000)
build_model(log_reg,X1_train,y1_train,X1_test,y1_test,data1,y,label='Data1 Default')
```

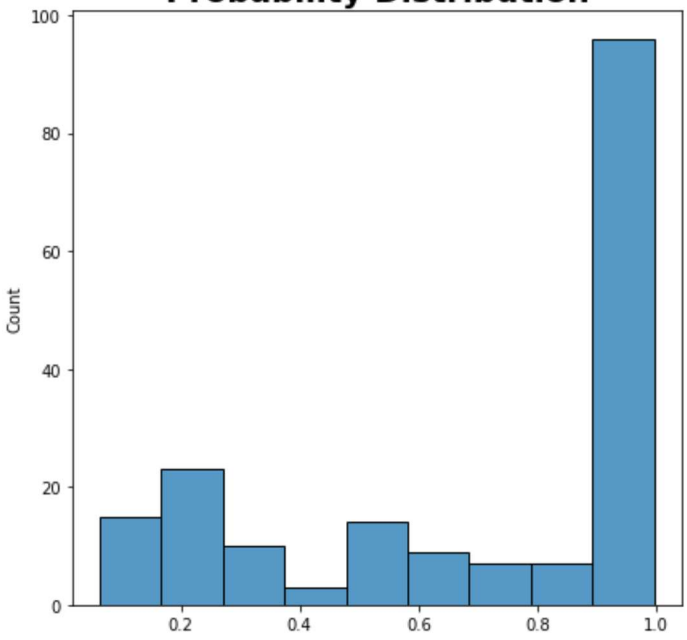
```
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\base.py:445: UserWarning: X does not have valid feature names, bu
t LogisticRegression was fitted with feature names
  warnings.warn(
              precision    recall  f1-score   support

           0       0.80      0.82      0.81        74
           1       0.88      0.86      0.87       110

    accuracy                           0.85       184
   macro avg       0.84      0.84      0.84       184
weighted avg       0.85      0.85      0.85       184

Specificity = 0.8243243243243243
The balanced accuracy score is : 84.40%
```

LogisticRegression Data1 Default







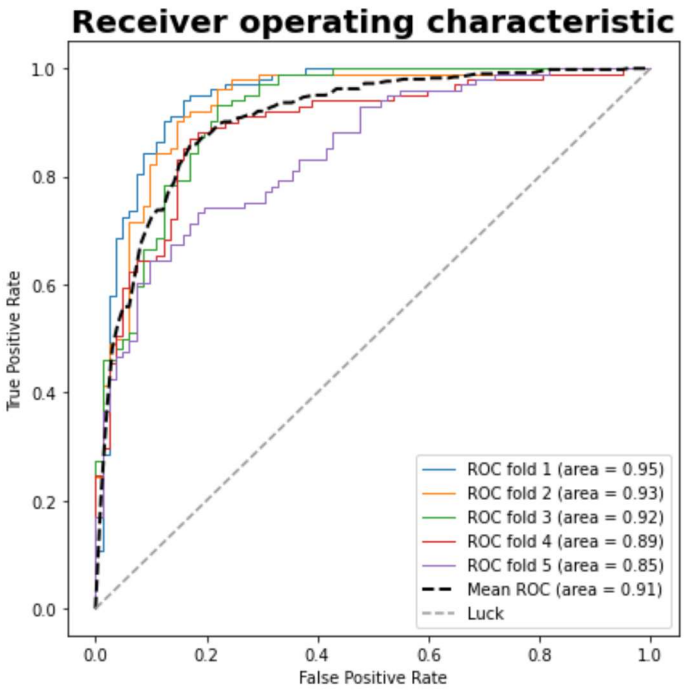


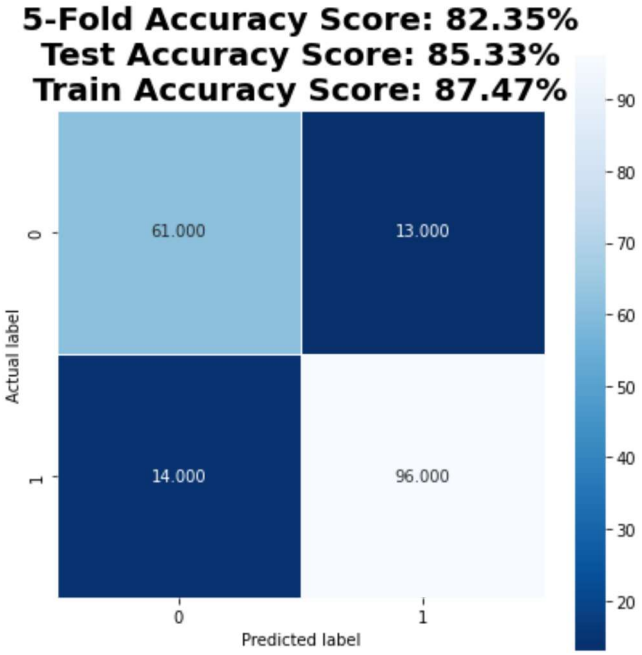
In [18]:

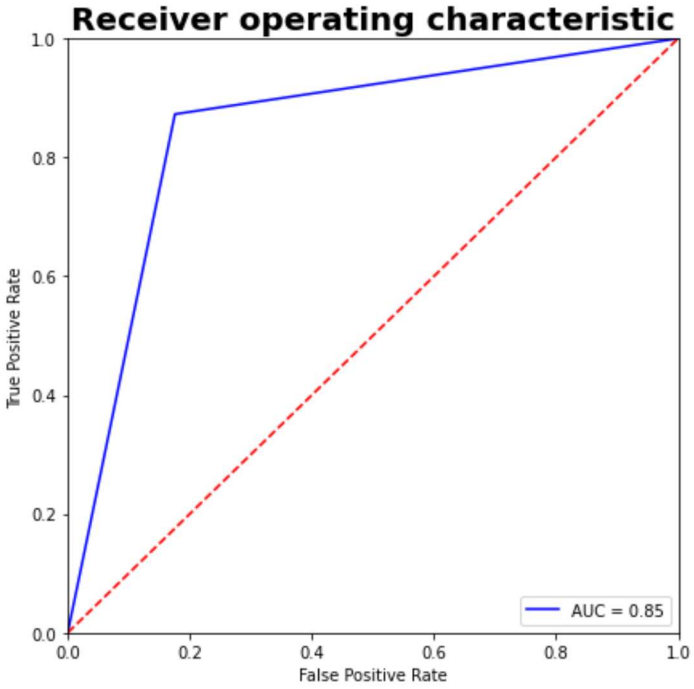
```
log_reg = LogisticRegression(max_iter=1000000)
build_model(log_reg,X2_train,y2_train,X2_test,y2_test,data2,y,label='Data2 Default')
```

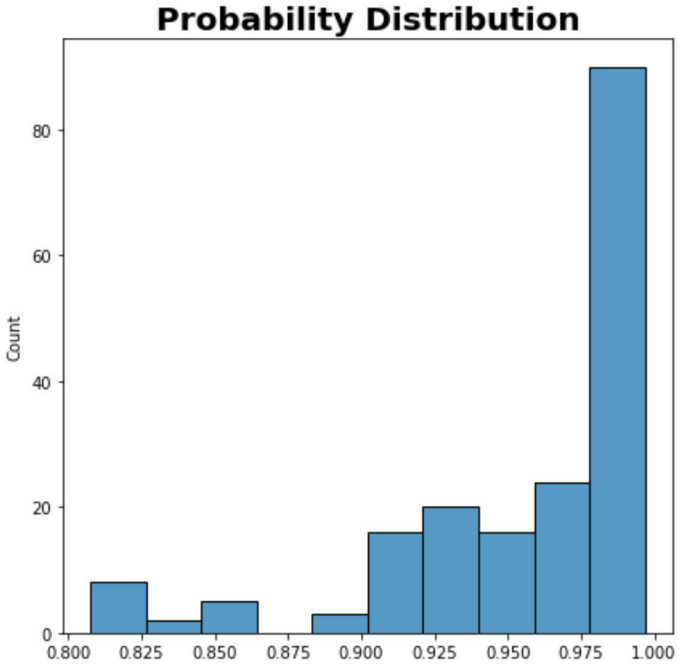
```
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\base.py:445: UserWarning: X does not have valid feature names, bu
t LogisticRegression was fitted with feature names
  warnings.warn(
              precision    recall  f1-score   support

           0       0.81      0.82      0.82        74
           1       0.88      0.87      0.88       110

    accuracy                           0.85       184
   macro avg       0.85      0.85      0.85       184
weighted avg       0.85      0.85      0.85       184

Specificity = 0.8243243243243243
The balanced accuracy score is : 84.85%
```

LogisticRegression Data2 Default







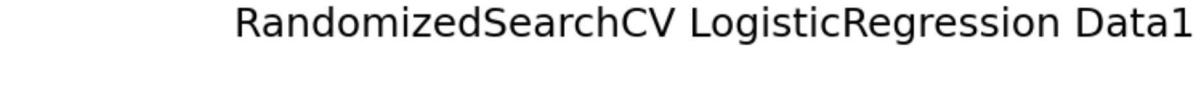


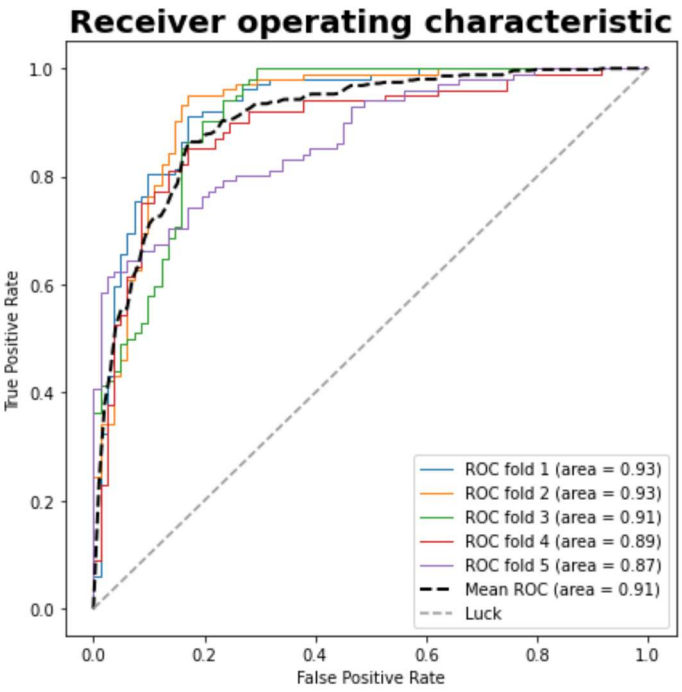
In [19]:

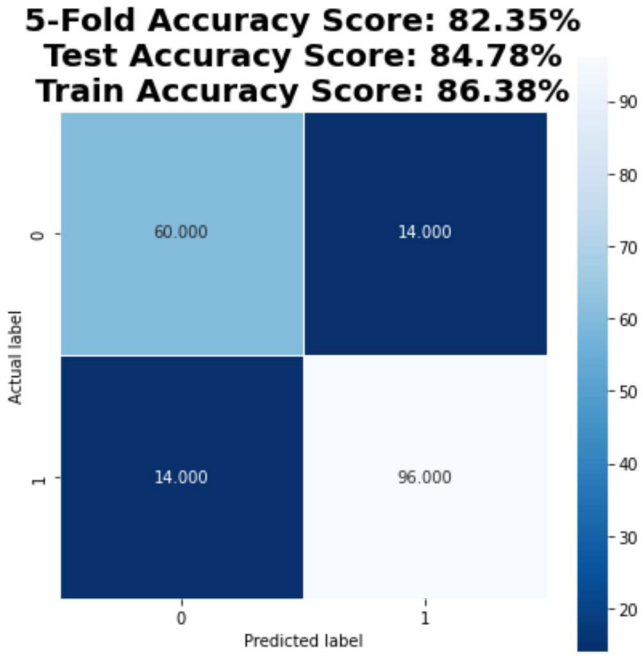
```
logRegGrid = {
    'penalty': ['elasticnet'],
    'solver': ['saga'],
    'fit_intercept': [True,False],
    'class_weight': ['balanced',None],
    'tol': [.0001, .01, 1, 10, 100],
    'C': [.01, 1, 10, 100],
    'l1_ratio': np.linspace(0,1,num=10),
    'n_jobs':[-1]
}

log_reg = LogisticRegression(max_iter=1000000)
build_model(log_reg,X1_train,y1_train,X1_test,y1_test,data1,y,params=logRegGrid,label='Data1')
```

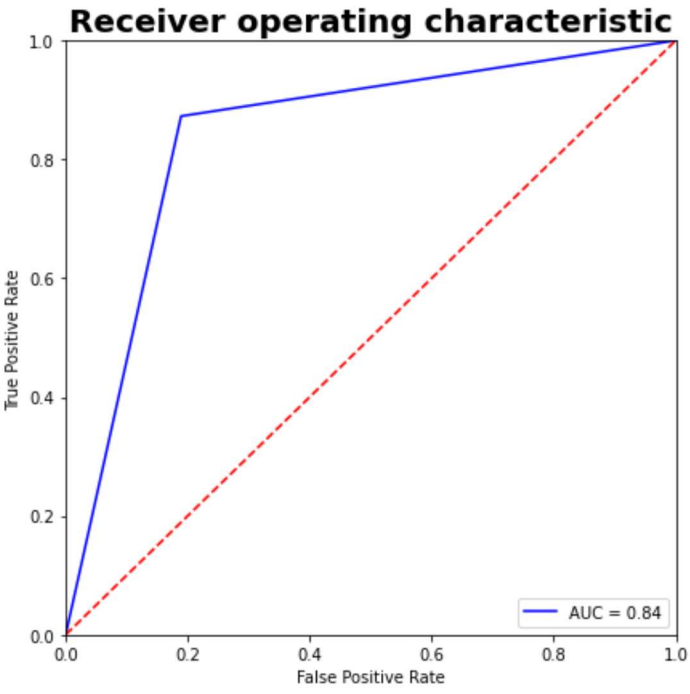
```
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\base.py:445: UserWarning: X does not have valid feature names, bu
t LogisticRegression was fitted with feature names
  warnings.warn(
              precision    recall  f1-score   support
```

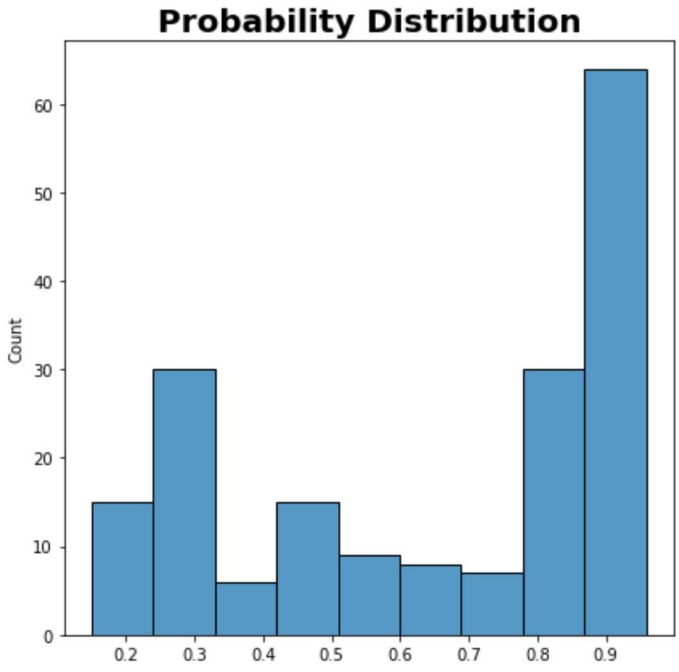
```
              0       0.81      0.81      0.81        74
              1       0.87      0.87      0.87       110

    accuracy                           0.85       184
   macro avg       0.84      0.84      0.84       184
weighted avg       0.85      0.85      0.85       184

Specificity = 0.8108108108108109
The balanced accuracy score is : 84.18%
```









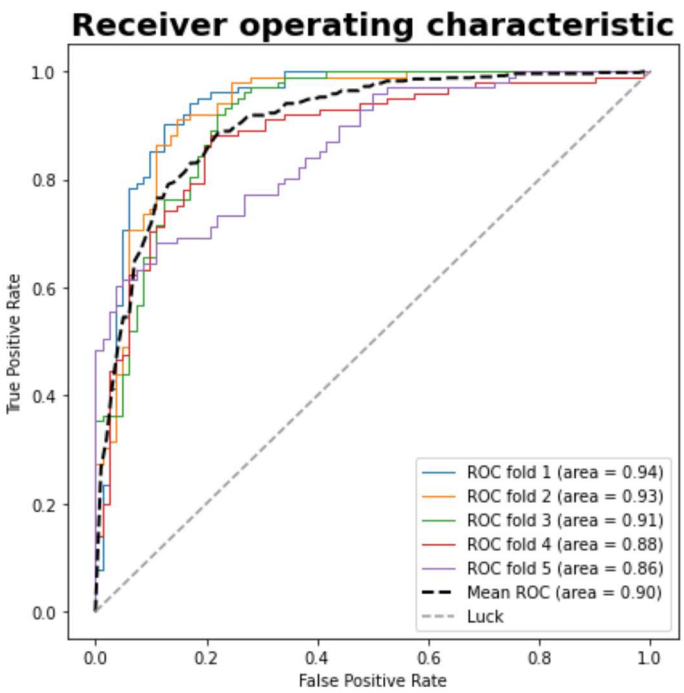


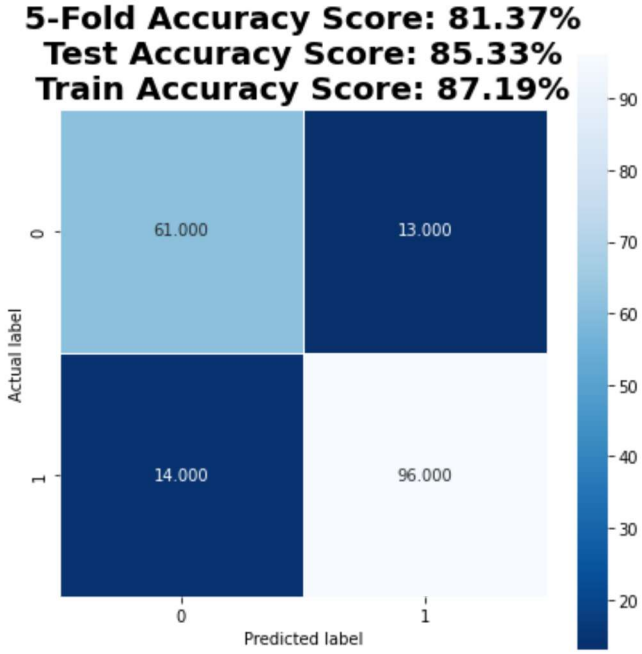
In [20]:
```
log_reg = LogisticRegression(max_iter=1000000)
build_model(log_reg,X2_train,y2_train,X2_test,y2_test,data2,y,params=logRegGrid,label='Data2')
```

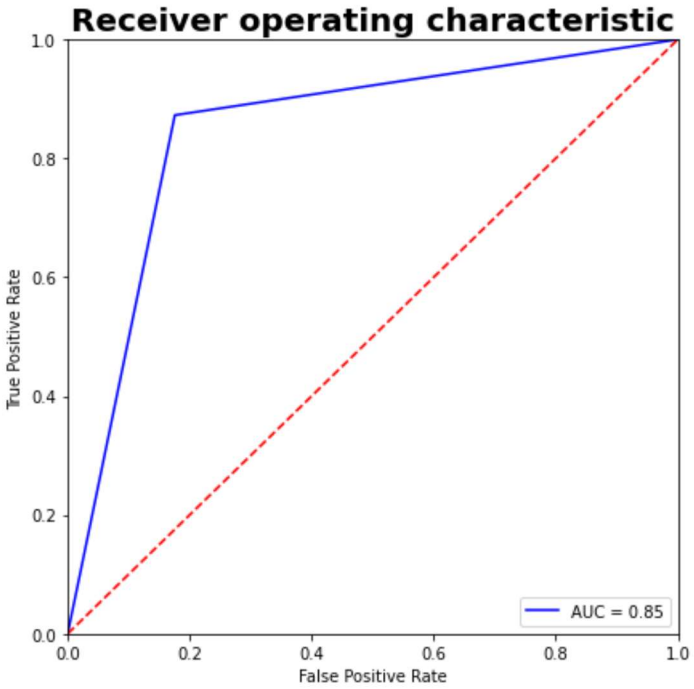
```
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\base.py:445: UserWarning: X does not have valid feature names, bu
t LogisticRegression was fitted with feature names
  warnings.warn(
              precision    recall  f1-score   support

           0       0.81      0.82      0.82        74
           1       0.88      0.87      0.88       110
```

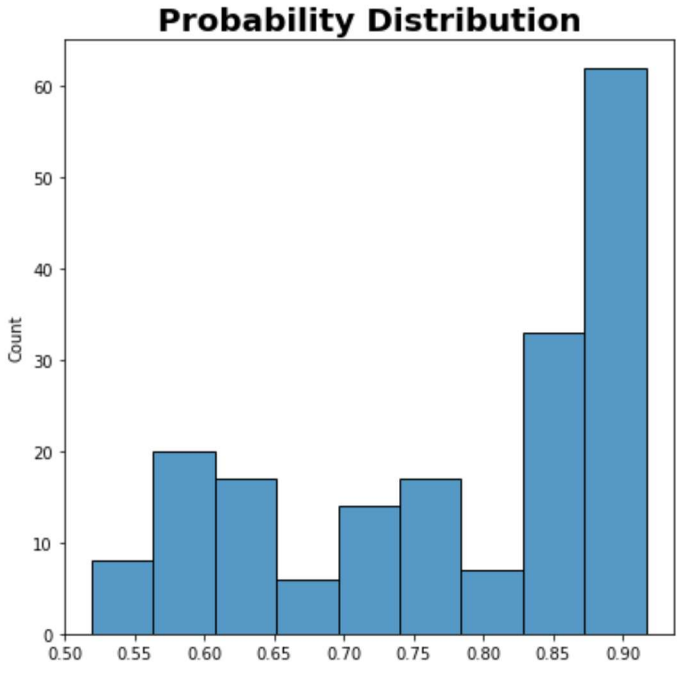
```
    accuracy                           0.85       184
   macro avg       0.85      0.85      0.85       184
weighted avg       0.85      0.85      0.85       184

Specificity = 0.8243243243243243
The balanced accuracy score is : 84.85%
```

RandomizedSearchCV LogisticRegression Data2









Logistic Regression Feature Selection - Sequential Feature Selector

Feature selection was also conducted by way of sequential feature selection. This methodology adds the feature that adds the most in predictive power in a sequential manner until there is no further improvement. As shown below, 5 of 11 features were selected, Age, Sex, ChestPainType, ExcerciseAngina, and ST_Slope as the most significant predictors towards heart disease. In fact when perform a logistic regression on the features alone, it achieves comparable metrics as when using all the available features.

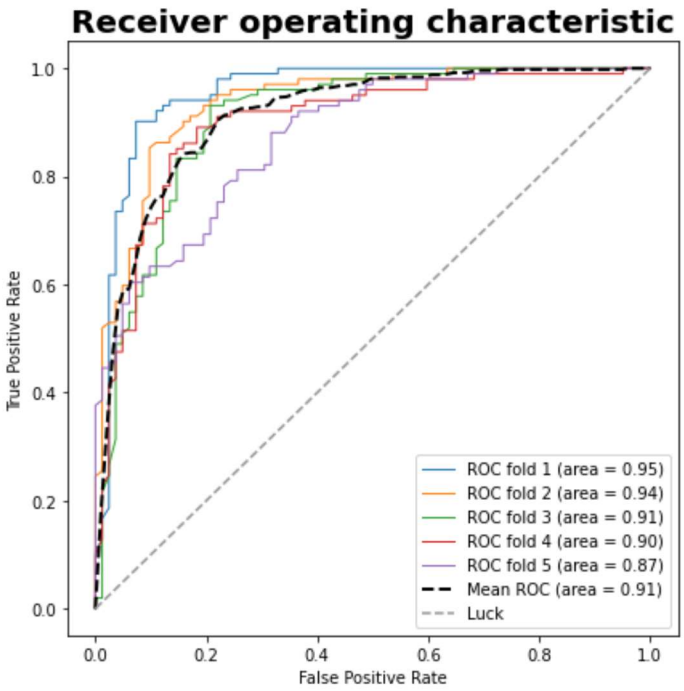
In [21]:

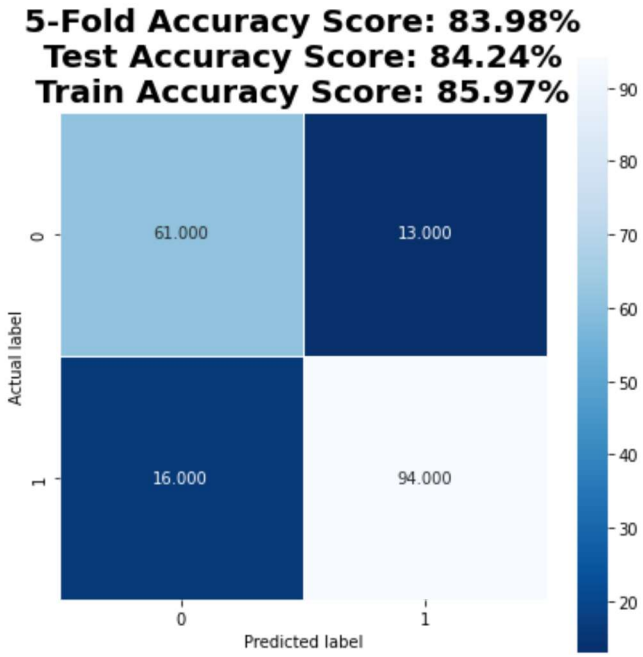
```
from sklearn.feature_selection import SequentialFeatureSelector
sfs = SequentialFeatureSelector(log_reg,n_jobs=-1,cv=cv,scoring='roc_auc')
sfs.fit(data2, y)
print(sfs.get_feature_names_out())
```

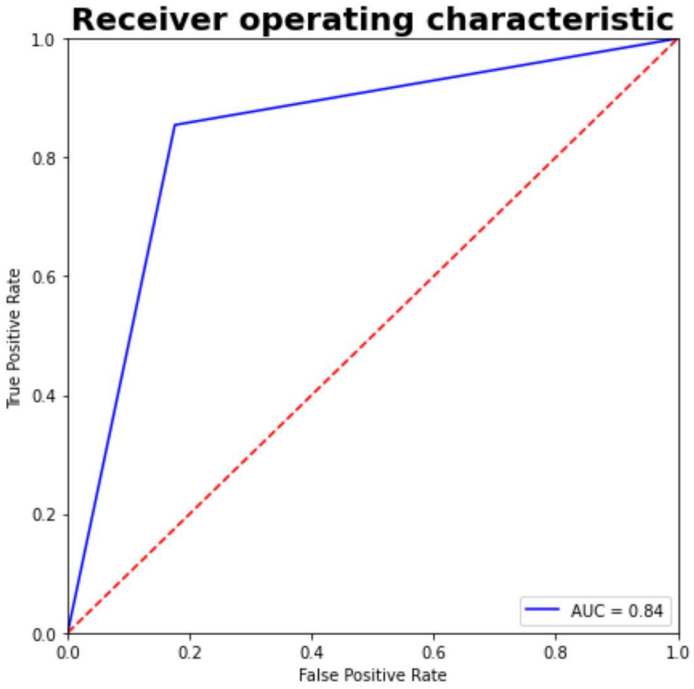
```
build_model(log_reg,sfs.transform(X2_train),y2_train,sfs.transform(X2_test),y2_test,
            pd.DataFrame(sfs.transform(data2)),y,label='Data2 FS')
```

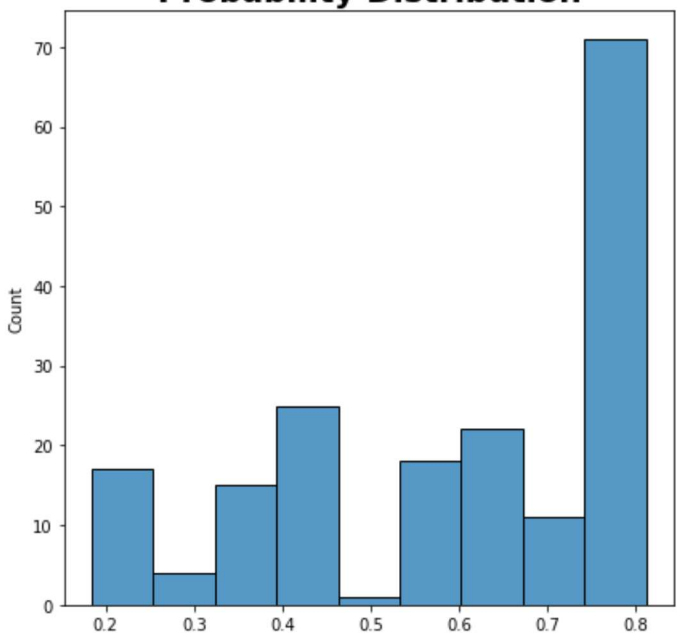
```
['Age' 'Sex' 'ChestPainType' 'ExerciseAngina' 'ST_Slope']
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\base.py:445: UserWarning: X does not have valid feature names, bu
t SequentialFeatureSelector was fitted with feature names
  warnings.warn(
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\base.py:445: UserWarning: X does not have valid feature names, bu
t SequentialFeatureSelector was fitted with feature names
  warnings.warn(
              precision    recall  f1-score   support

           0       0.79      0.82      0.81        74
           1       0.88      0.85      0.87       110

    accuracy                           0.84       184
   macro avg       0.84      0.84      0.84       184
weighted avg       0.84      0.84      0.84       184

Specificity = 0.8243243243243243
The balanced accuracy score is : 83.94%
```

LogisticRegression Data2 FS







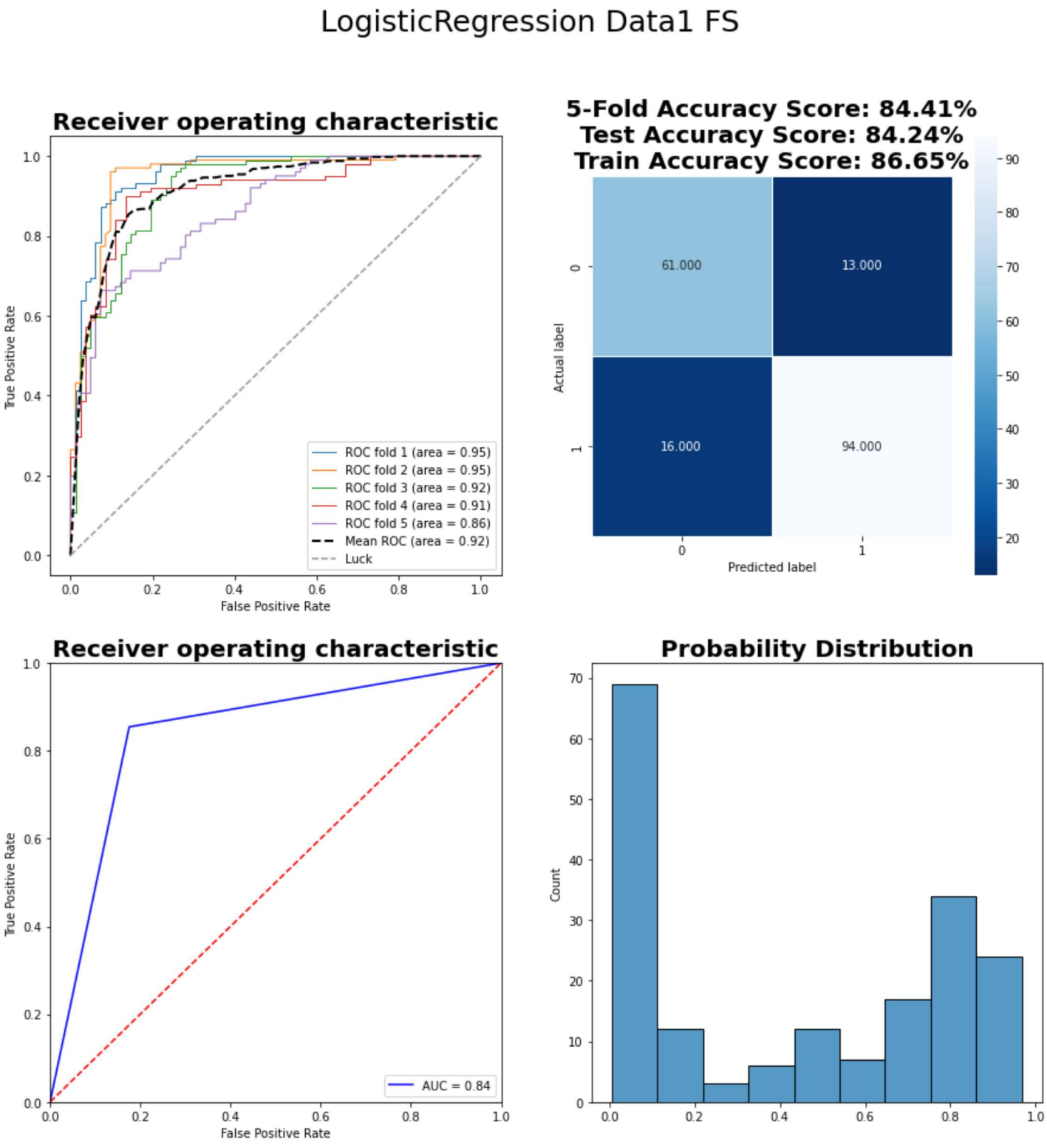


In [22]:

```
sfs = SequentialFeatureSelector(log_reg,n_jobs=-1,cv=cv,scoring='roc_auc')
sfs.fit(data1, y)
print(sfs.get_feature_names_out())
build_model(log_reg,sfs.transform(X1_train),y1_train,sfs.transform(X1_test),y1_test,
            pd.DataFrame(sfs.transform(data1)),y,label='Data1 FS')
```

```
['Age' 'FastingBS' 'Oldpeak' 'Sex_F' 'Sex_M' 'ChestPainType_ASY'
 'ExerciseAngina_N' 'ST_Slope_Down' 'ST_Slope_Flat' 'ST_Slope_Up']
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\base.py:445: UserWarning: X does not have valid feature names, bu
t SequentialFeatureSelector was fitted with feature names
  warnings.warn(
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\base.py:445: UserWarning: X does not have valid feature names, bu
t SequentialFeatureSelector was fitted with feature names
  warnings.warn(
              precision    recall  f1-score   support

           0       0.79      0.82      0.81        74
           1       0.88      0.85      0.87       110
```

```
    accuracy                           0.84       184
   macro avg       0.84      0.84      0.84       184
weighted avg       0.84      0.84      0.84       184

Specificity = 0.8243243243243243
The balanced accuracy score is : 83.94%
```



Random Forest Classifier

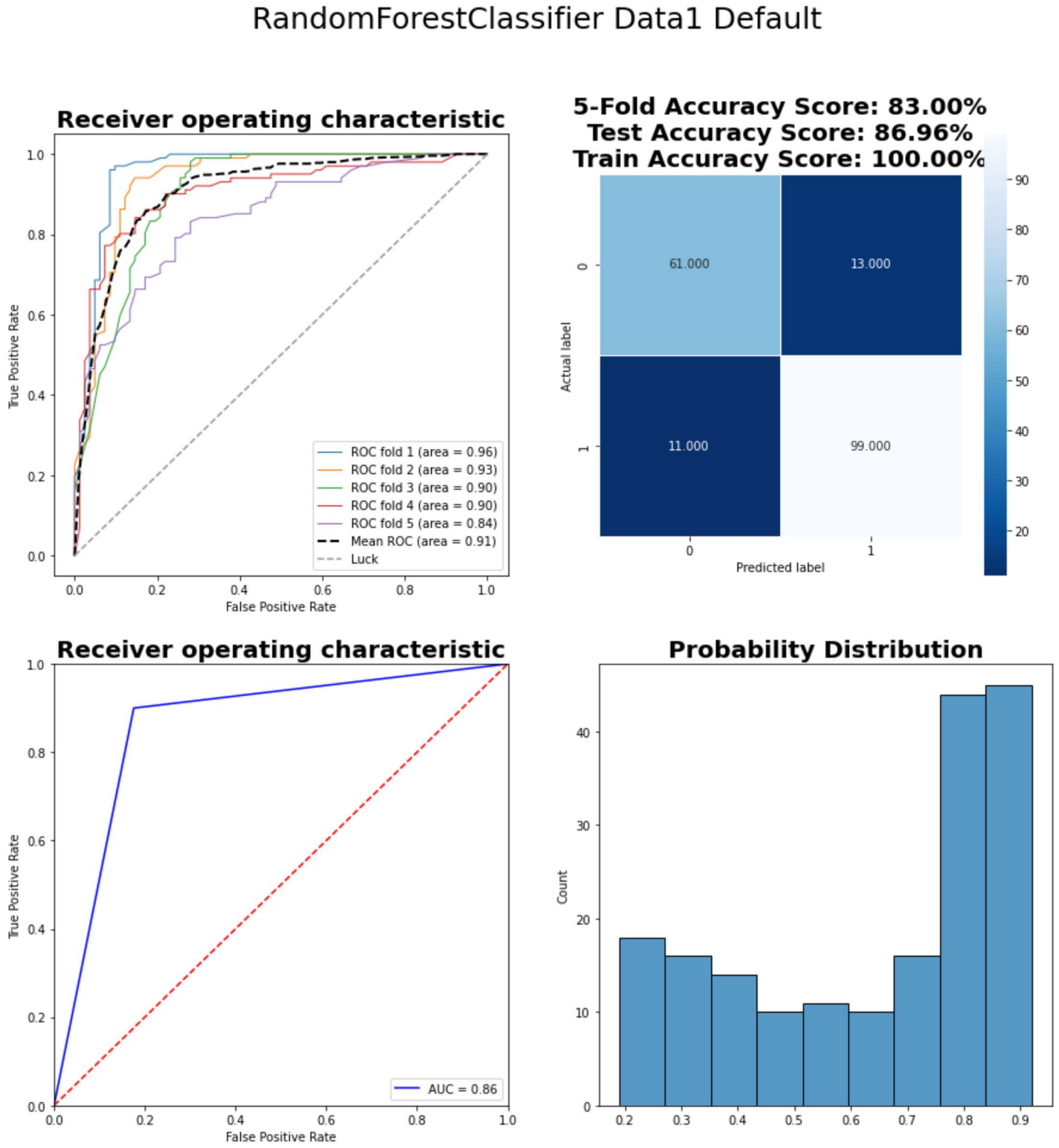
The random forest models performed just as well as the logistic regression models. The test accuracy and AUC scores are comparable to the logistic regression. One advantage over logistic regression is that it seems to have a balanced ratio between predicting heart disease or not. It does not seem to favor one prediction over the other. It seems this is a more generalizable model. This can be further confirmed by the two opposing peaks in the probability distribution.

In [23]:

```
from sklearn.ensemble import RandomForestClassifier

rfc = RandomForestClassifier(n_estimators=100, random_state=32, n_jobs=-1)
```

```
build_model(rfc,X1_train,y1_train,X1_test,y1_test,data1,y,label='Data1 Default')
```

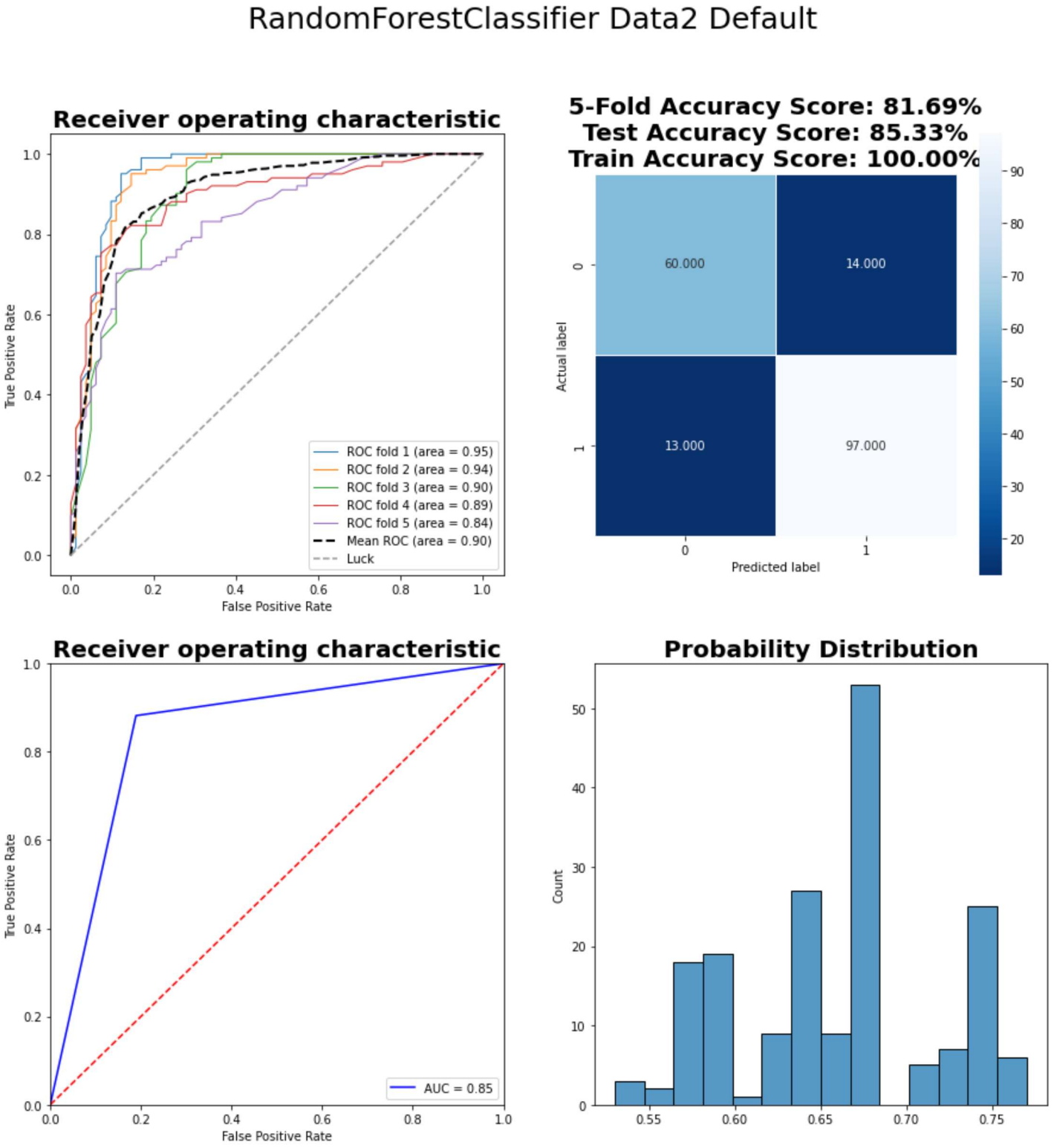
```
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\base.py:445: UserWarning: X does not have valid feature names, bu
t RandomForestClassifier was fitted with feature names
  warnings.warn(
              precision    recall  f1-score   support

           0       0.85      0.82      0.84        74
           1       0.88      0.90      0.89       110

    accuracy                           0.87       184
   macro avg       0.87      0.86      0.86       184
weighted avg       0.87      0.87      0.87       184

Specificity = 0.8243243243243243
The balanced accuracy score is : 86.22%
```



In [24]:
```
rfc = RandomForestClassifier(n_estimators=100, random_state=32, n_jobs=-1)
```

```
build_model(rfc,X2_train,y2_train,X2_test,y2_test,data2,y,label='Data2 Default')
```

```
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\base.py:445: UserWarning: X does not have valid feature names, bu
t RandomForestClassifier was fitted with feature names
  warnings.warn(
              precision    recall  f1-score   support

           0       0.82      0.81      0.82        74
           1       0.87      0.88      0.88       110

    accuracy                           0.85       184
   macro avg       0.85      0.85      0.85       184
weighted avg       0.85      0.85      0.85       184

Specificity = 0.8108108108108109
The balanced accuracy score is : 84.63%
```



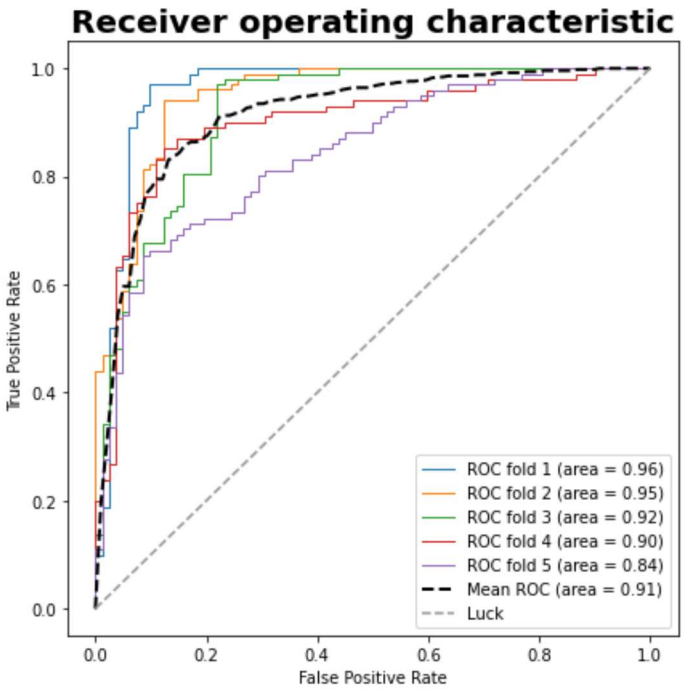
In [25]:

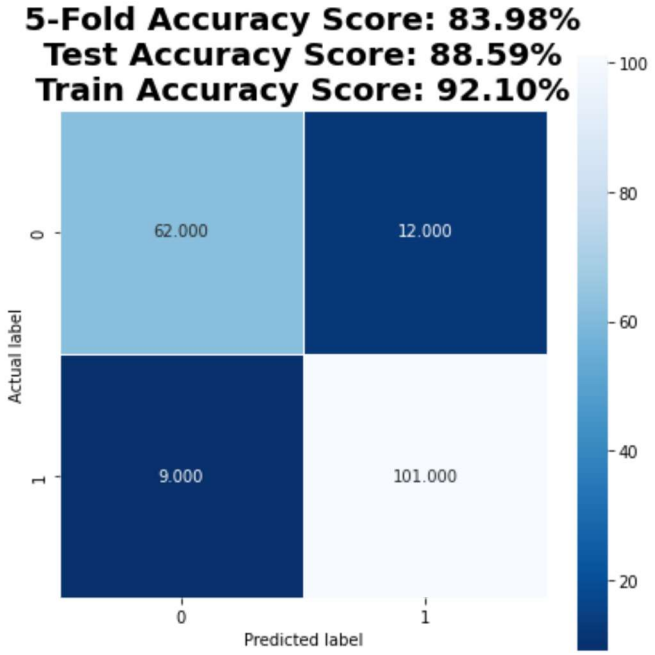
```
rfc_params = {
    'max_depth': [3, 7, 10, 50, 100],
    'min_samples_split' : [ 3, 7, 10, 50, 100],
```

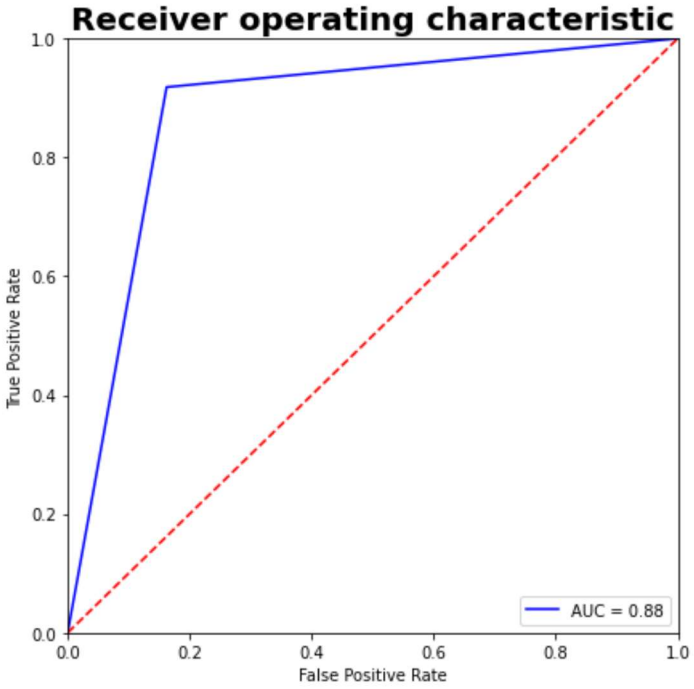
```
    'min_samples_leaf' : [ 3, 7, 10, 50, 100],
    'max_leaf_nodes' : [ 3, 7, 10, 50, 100],
    'criterion': ['entropy','gini']
}

rfc = RandomForestClassifier(n_estimators=100, random_state=32, n_jobs=-1)

build_model(rfc,X1_train,y1_train,X1_test,y1_test,data1,y,params=rfc_params,label='Data1')
```

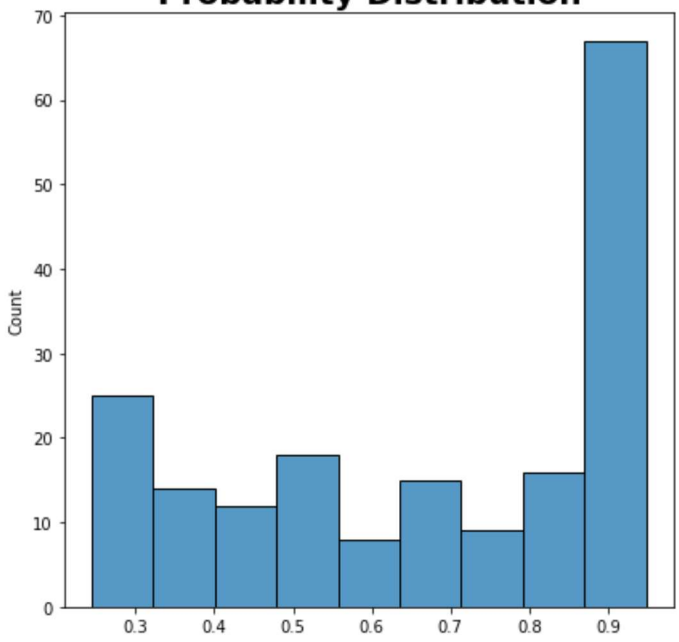
```
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\base.py:445: UserWarning: X does not have valid feature names, bu
t RandomForestClassifier was fitted with feature names
  warnings.warn(
              precision    recall  f1-score   support

           0       0.87      0.84      0.86        74
           1       0.89      0.92      0.91       110

    accuracy                           0.89       184
   macro avg       0.88      0.88      0.88       184
weighted avg       0.89      0.89      0.89       184

Specificity = 0.8378378378378378
The balanced accuracy score is : 87.80%
```

RandomizedSearchCV RandomForestClassifier Data1





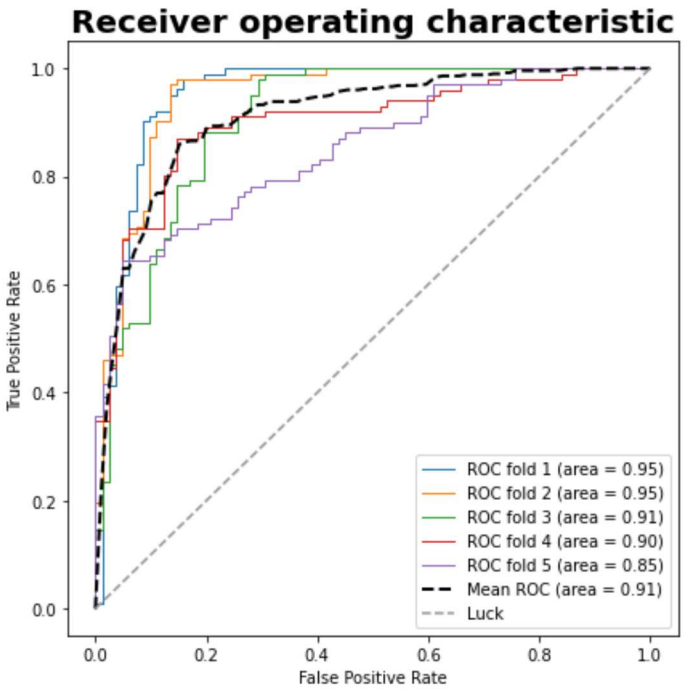


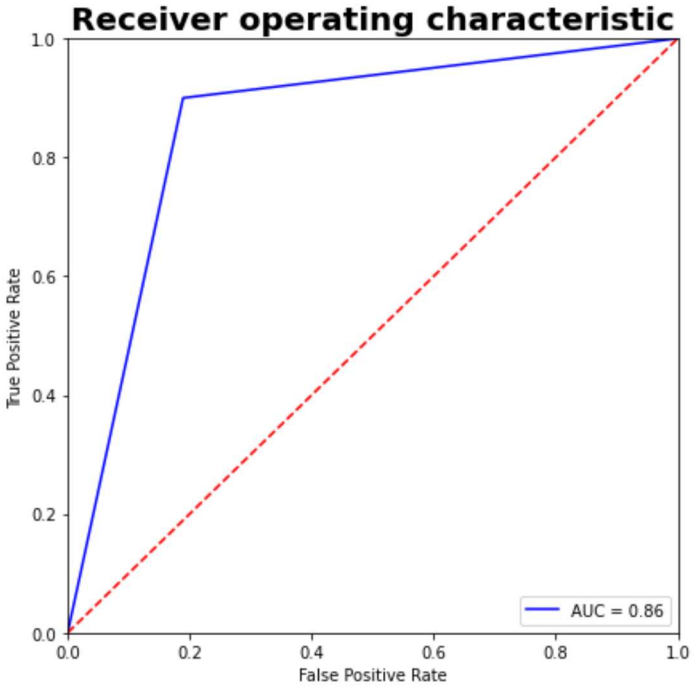


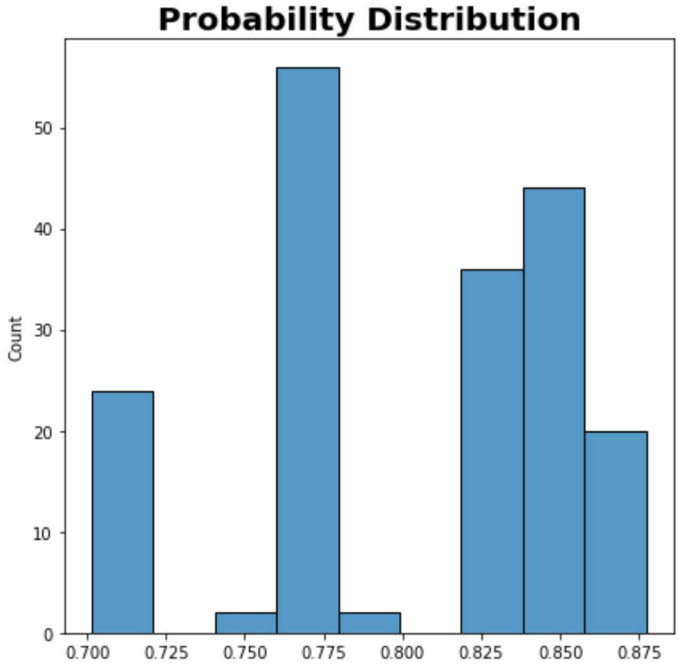
In [26]:
```
rfc = RandomForestClassifier(n_estimators=100, random_state=32, n_jobs=-1)

build_model(rfc,X2_train,y2_train,X2_test,y2_test,data2,y,params=rfc_params,label='Data2')
```

```
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\base.py:445: UserWarning: X does not have valid feature names, bu
t RandomForestClassifier was fitted with feature names
  warnings.warn(
              precision    recall  f1-score   support

           0       0.85      0.81      0.83        74
           1       0.88      0.90      0.89       110

    accuracy                           0.86       184
   macro avg       0.86      0.86      0.86       184
weighted avg       0.86      0.86      0.86       184

Specificity = 0.8108108108108109
The balanced accuracy score is : 85.54%
```

RandomizedSearchCV RandomForestClassifier Data2







Random Forest Feature Selection - Sequential Feature Selection

Feature selection was also conducted using the random forest classifier and a sequential feature selection. There were multiple features that were selected that were also selected in the logistic regression feature selection. Feature selection involved 6 variables, Sex, ChestPainType, FastingBS, OldPeak, and ST_Slope. Sex, ChestPainType and ST_Slope were also selected by the logistic regression model. This adds evidence that these features are important in determining heart disease.

In [27]:
```
sfs = SequentialFeatureSelector(rfc,n_jobs=-1,cv=cv,scoring='roc_auc')
sfs.fit(data2, y)
print(sfs.get_feature_names_out())
build_model(rfc,sfs.transform(X2_train),y2_train,sfs.transform(X2_test),y2_test,
            pd.DataFrame(sfs.transform(data2)),y,label='Data2 FS')
```

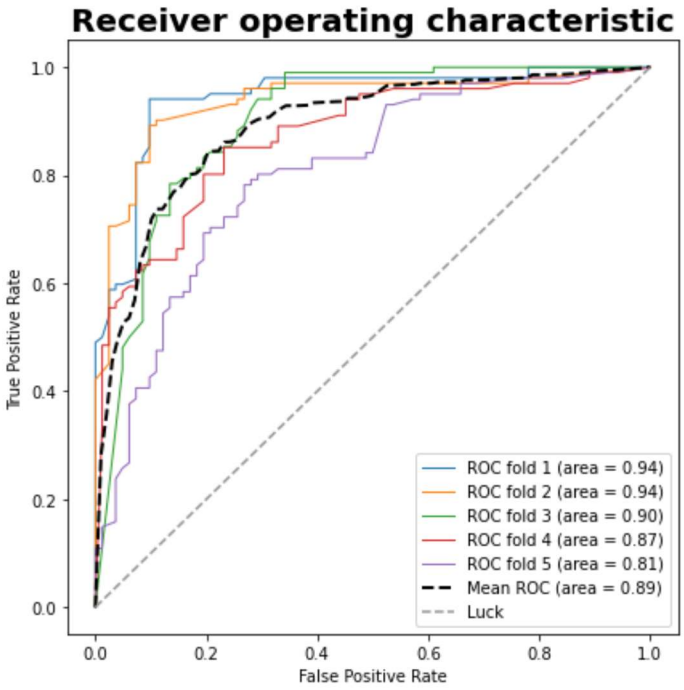
```
['Sex' 'ChestPainType' 'FastingBS' 'Oldpeak' 'ST_Slope']
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\base.py:445: UserWarning: X does not have valid feature names, bu
t SequentialFeatureSelector was fitted with feature names
  warnings.warn(
```

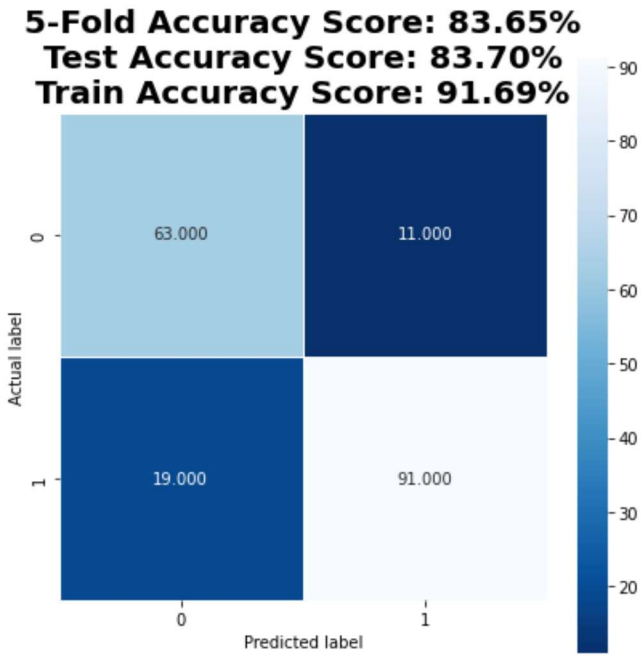
```
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\base.py:445: UserWarning: X does not have valid feature names, but SequentialFeatureSelector was fitted with feature names
  warnings.warn(
```

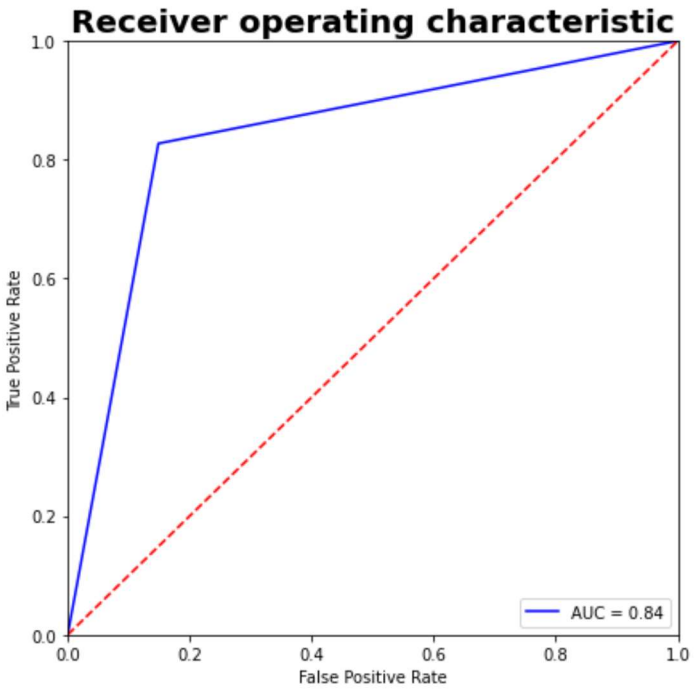
	precision	recall	f1-score	support
0	0.77	0.85	0.81	74
1	0.89	0.83	0.86	110
accuracy			0.84	184
macro avg	0.83	0.84	0.83	184
weighted avg	0.84	0.84	0.84	184

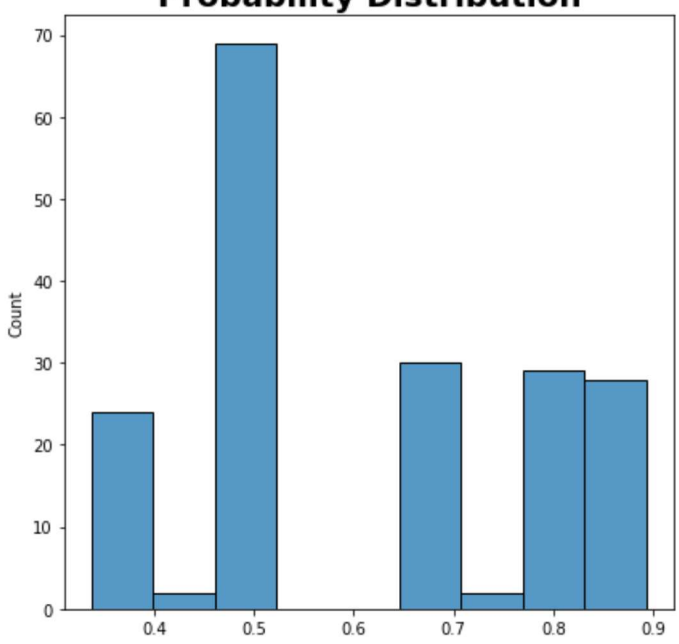
```
Specificity = 0.8513513513513513
The balanced accuracy score is : 83.93%
```

RandomForestClassifier Data2 FS







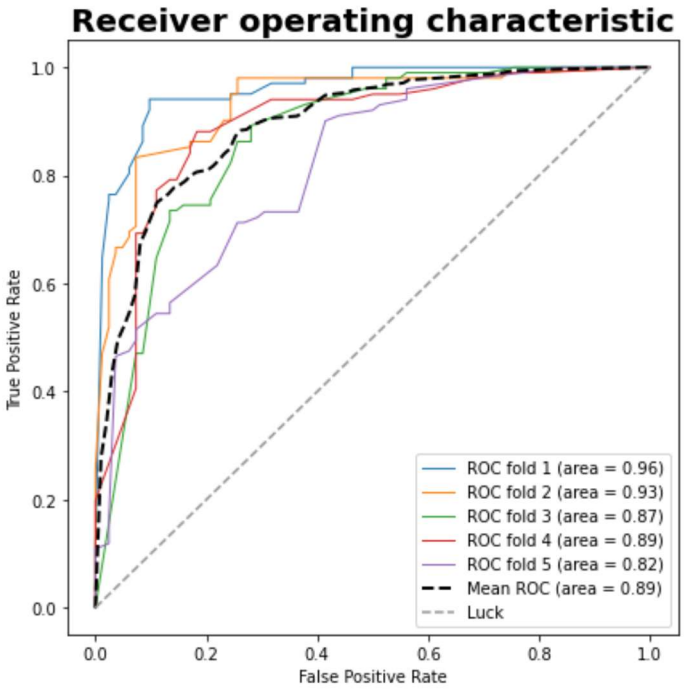


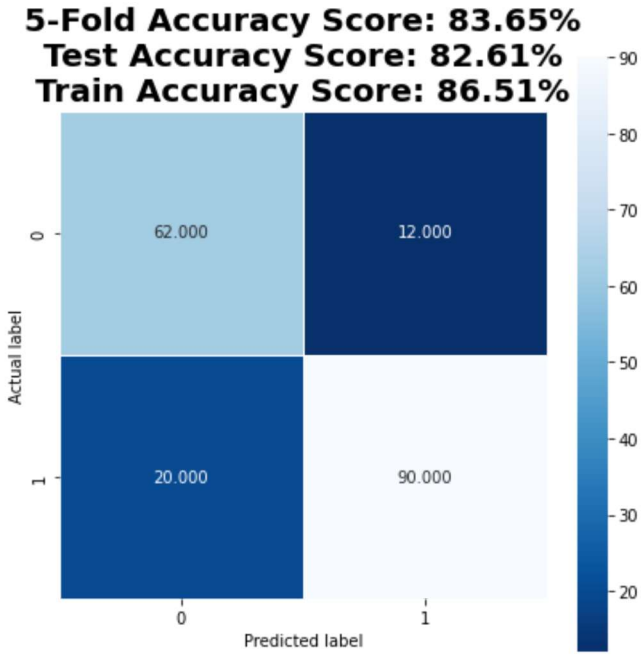
In [28]:

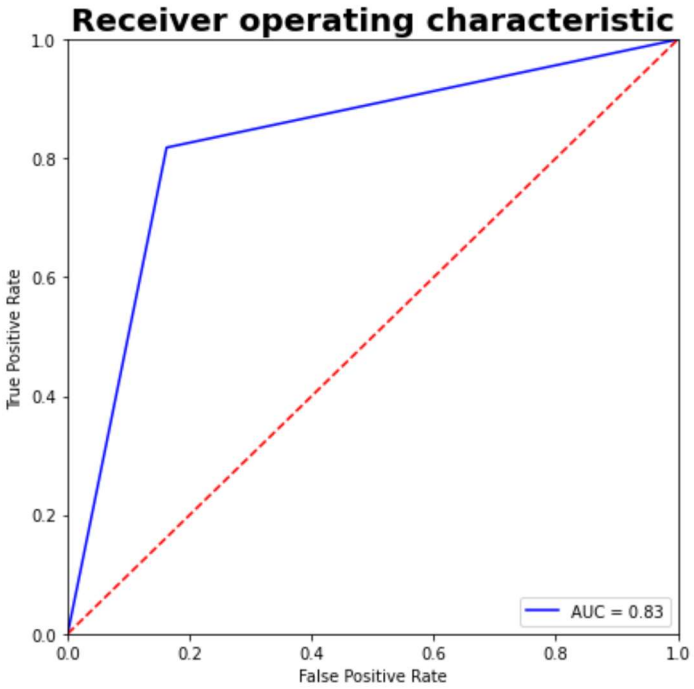
```
sfs = SequentialFeatureSelector(rfc,n_jobs=-1,cv=cv,scoring='roc_auc')
sfs.fit(data1, y)
print(sfs.get_feature_names_out())
build_model(rfc,sfs.transform(X1_train),y1_train,sfs.transform(X1_test),y1_test,
            pd.DataFrame(sfs.transform(data1)),y,label='Data1 FS')
```

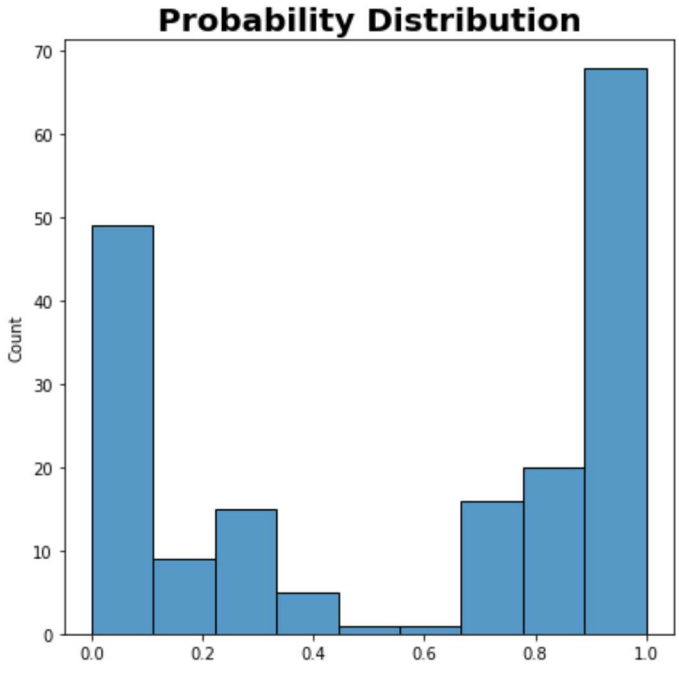
```
['FastingBS' 'Sex_F' 'Sex_M' 'ChestPainType_ASY' 'ChestPainType_ATA'
 'ChestPainType_NAP' 'ChestPainType_TA' 'RestingECG_ST' 'ST_Slope_Down'
 'ST_Slope_Up']
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\base.py:445: UserWarning: X does not have valid feature names, bu
t SequentialFeatureSelector was fitted with feature names
  warnings.warn(
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\base.py:445: UserWarning: X does not have valid feature names, bu
t SequentialFeatureSelector was fitted with feature names
  warnings.warn(
              precision    recall  f1-score   support

           0       0.76      0.84      0.79        74
           1       0.88      0.82      0.85       110

    accuracy                           0.83       184
   macro avg       0.82      0.83      0.82       184
weighted avg       0.83      0.83      0.83       184

Specificity = 0.8378378378378378
The balanced accuracy score is : 82.80%
```

RandomForestClassifier Data1 FS









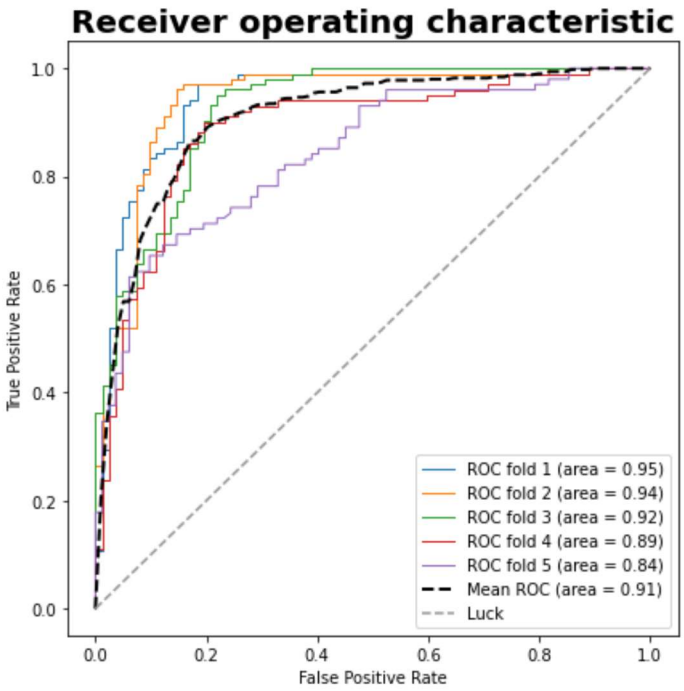
Support Vector Machines

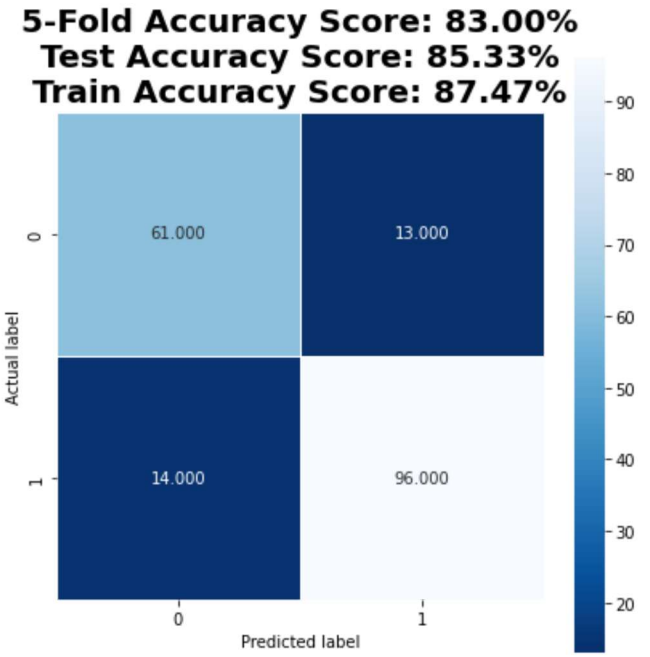
Support Vector machine performed just as well as the previous models. The only downside is that the probabilities are close to the threshold, meaning that each prediction is not strong. This model also took the longest to train without adding much to the accuracy.

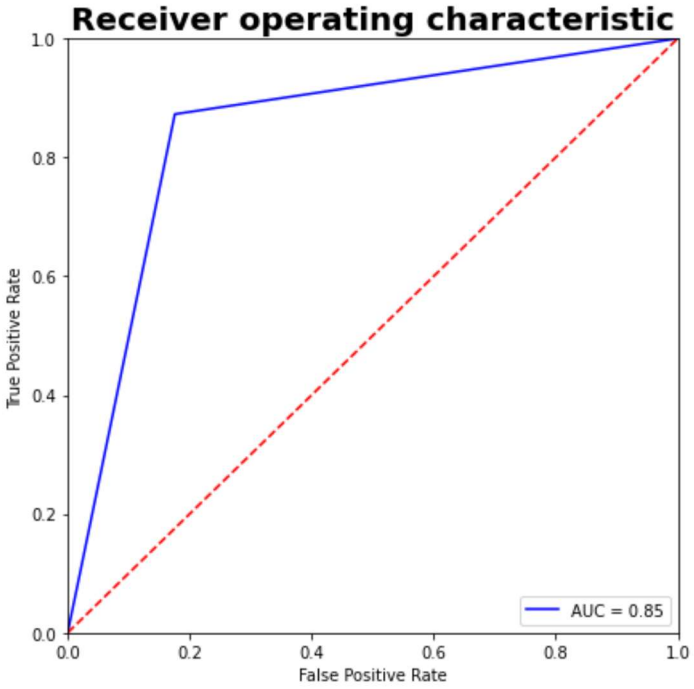
In [29]:
```
from sklearn.svm import SVC

svc = SVC(kernel ='linear',probability=True)
build_model(svc,X1_train,y1_train,X1_test,y1_test,data1,y,label='Data1 Default')
```

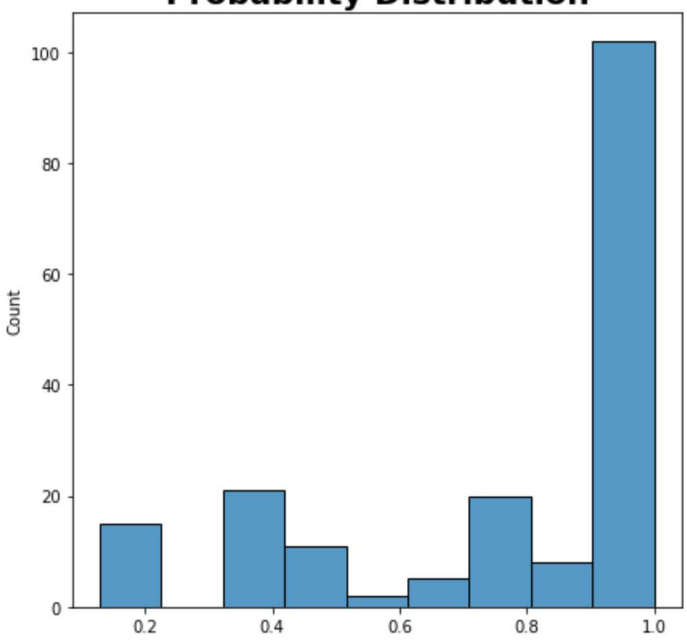
```
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\base.py:445: UserWarning: X does not have valid feature names, bu
t SVC was fitted with feature names
  warnings.warn(
              precision    recall  f1-score   support

           0       0.81      0.82      0.82        74
           1       0.88      0.87      0.88       110

    accuracy                           0.85       184
   macro avg       0.85      0.85      0.85       184
weighted avg       0.85      0.85      0.85       184

Specificity = 0.8243243243243243
The balanced accuracy score is : 84.85%
```

SVC Data1 Default





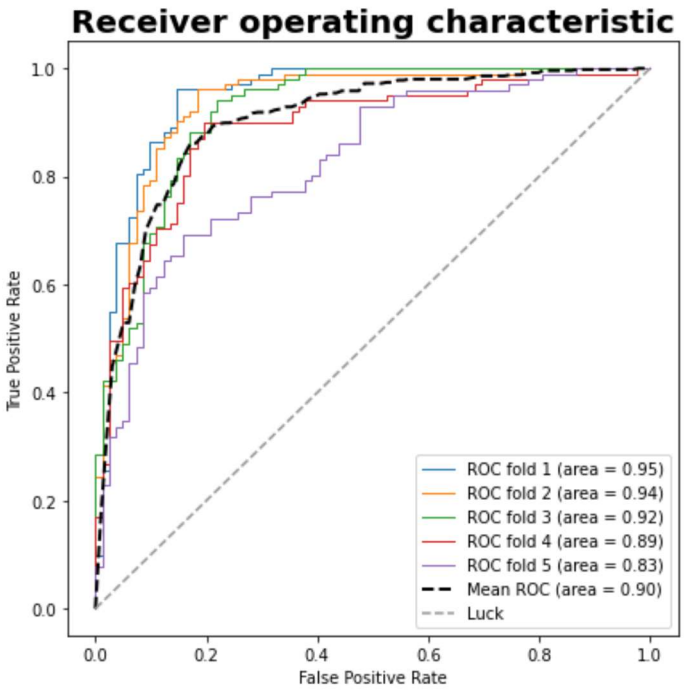


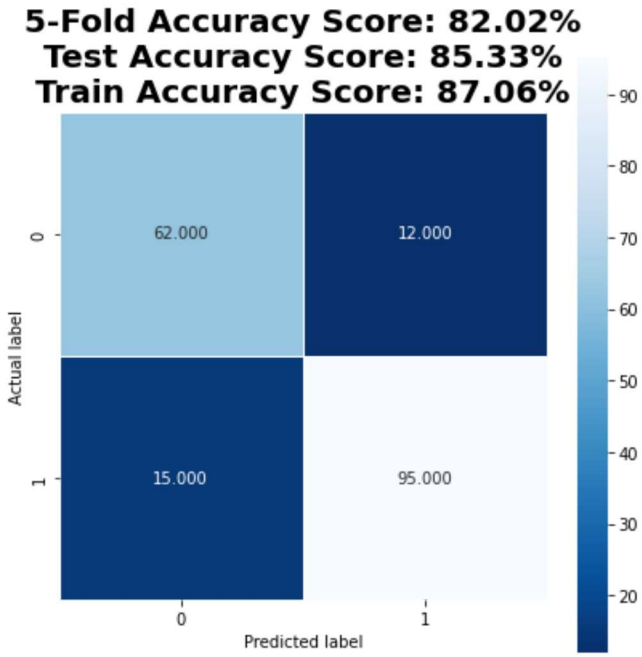


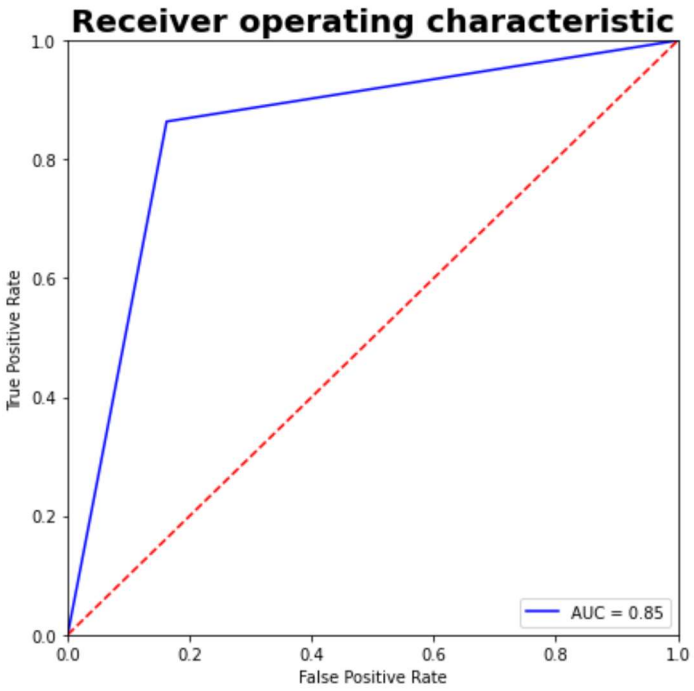
In [30]:
```
svc = SVC(kernel ='linear',probability=True)
build_model(svc,X2_train,y2_train,X2_test,y2_test,data2,y,label='Data2 Default')
```

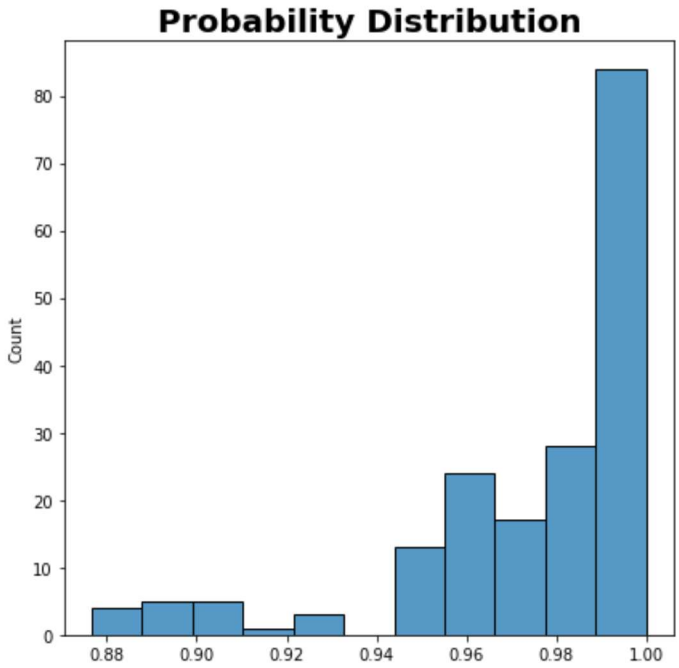
```
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\base.py:445: UserWarning: X does not have valid feature names, bu
t SVC was fitted with feature names
  warnings.warn(
              precision    recall  f1-score   support

           0       0.81      0.84      0.82        74
           1       0.89      0.86      0.88       110

    accuracy                           0.85       184
   macro avg       0.85      0.85      0.85       184
weighted avg       0.85      0.85      0.85       184

Specificity = 0.8378378378378378
The balanced accuracy score is : 85.07%
```

SVC Data2 Default









In [31]:
```
svc_params ={
  'C': [1, 10, 100, 1000],
  'gamma': [0.001, 0.01, 0.1, 1]
}
svc = SVC(kernel='linear',probability=True)
# Takes a long time
# build_model(svc,X1_train,y1_train,X1_test,y1_test,data1,y,label='Data1',params=svc_params)
```

In [32]:
```
svc = SVC(kernel='linear',probability=True)
# Takes a long time
# build_model(svc,X2_train,y2_train,X2_test,y2_test,data2,y,label='Data2',params=svc_params)
```

SVC Feature Selection - Sequential Feature Selection

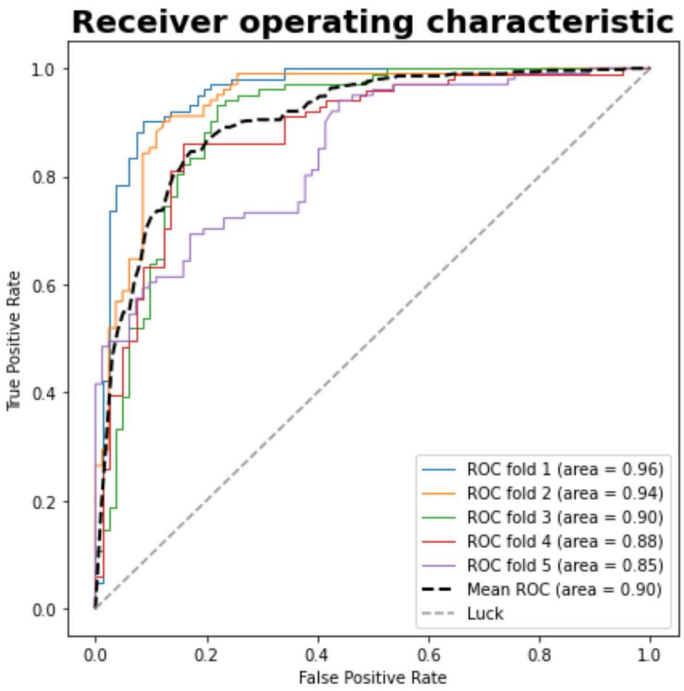
The feature selection keeps selecting the same 3-5 features indicating that these features, Sex, ChestPainType, MaxHR, OldPeak, and ST_Slope are important predictors towards predicting heart disease.

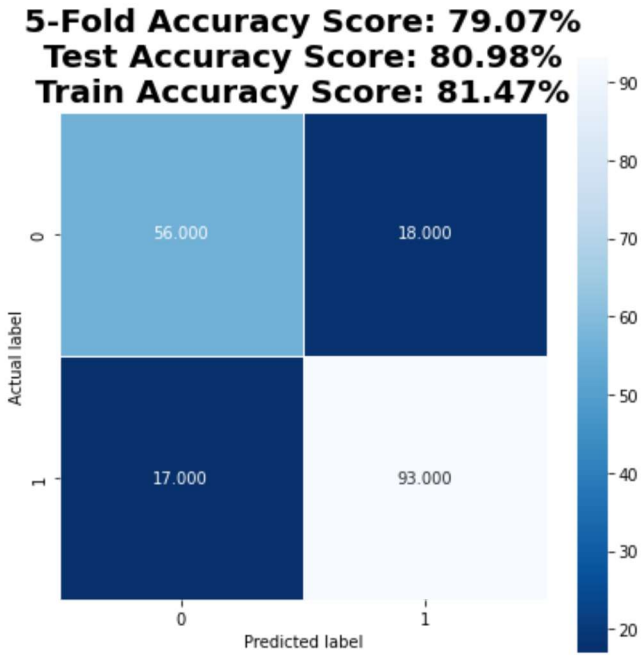
In [33]:

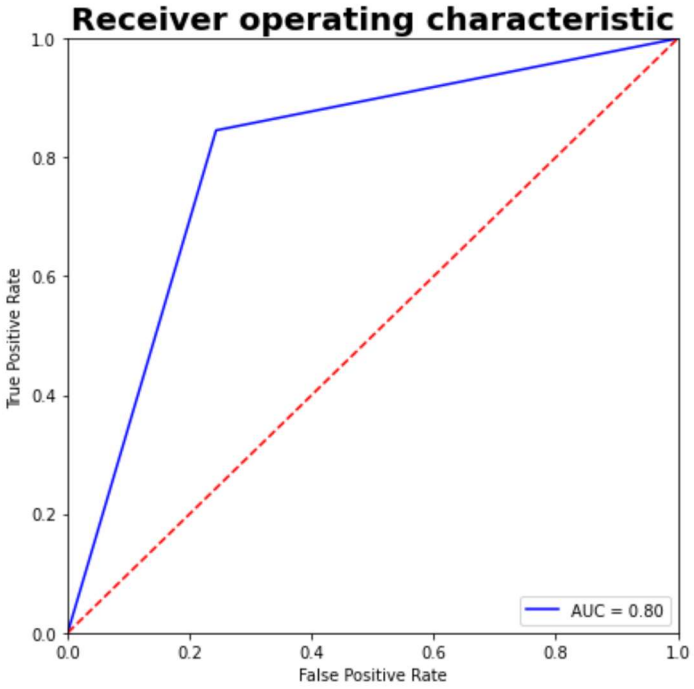
```
svc = SVC(kernel ='linear',probability=True)
sfs = SequentialFeatureSelector(svc,n_jobs=-1,cv=cv,scoring='roc_auc')
sfs.fit(data2, y)
print(sfs.get_feature_names_out())
build_model(svc,sfs.transform(X2_train),y2_train,sfs.transform(X2_test),y2_test,
            pd.DataFrame(sfs.transform(data2)),y,label='Data2 FS')
```

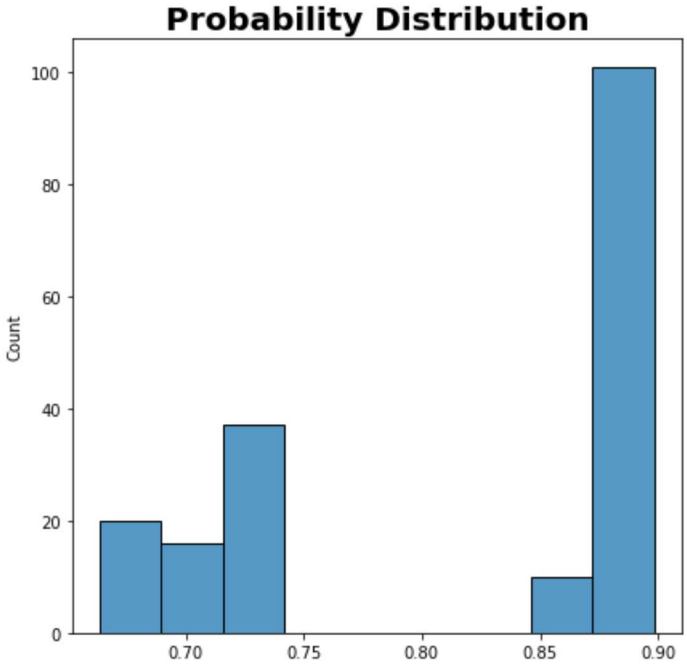
```
['Sex' 'ChestPainType' 'MaxHR' 'Oldpeak' 'ST_Slope']
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\base.py:445: UserWarning: X does not have valid feature names, bu
t SequentialFeatureSelector was fitted with feature names
  warnings.warn(
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\base.py:445: UserWarning: X does not have valid feature names, bu
t SequentialFeatureSelector was fitted with feature names
  warnings.warn(
              precision    recall  f1-score   support

           0       0.77      0.76      0.76        74
           1       0.84      0.85      0.84       110

    accuracy                           0.81       184
   macro avg       0.80      0.80      0.80       184
weighted avg       0.81      0.81      0.81       184

Specificity = 0.7567567567567568
The balanced accuracy score is : 80.11%
```

SVC Data2 FS









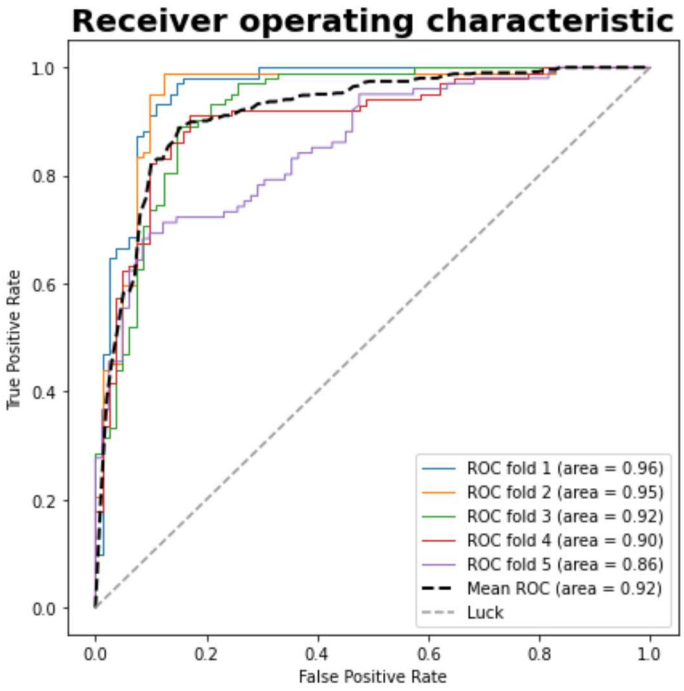
In [34]:

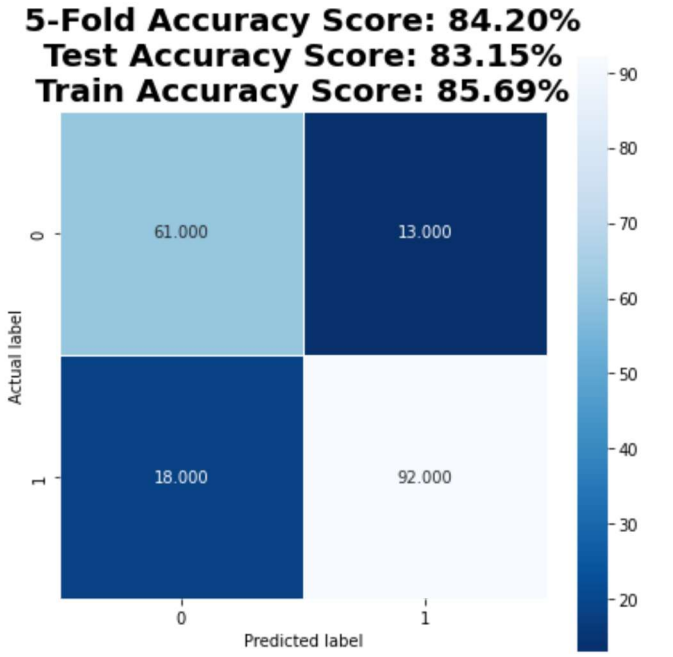
```
svc = SVC(kernel ='linear',probability=True)
sfs = SequentialFeatureSelector(svc,n_jobs=-1,cv=cv,scoring='roc_auc')
sfs.fit(data1, y)
print(sfs.get_feature_names_out())
build_model(svc,sfs.transform(X1_train),y1_train,sfs.transform(X1_test),y1_test,
            pd.DataFrame(sfs.transform(data1)),y,label='Data1 FS')
```

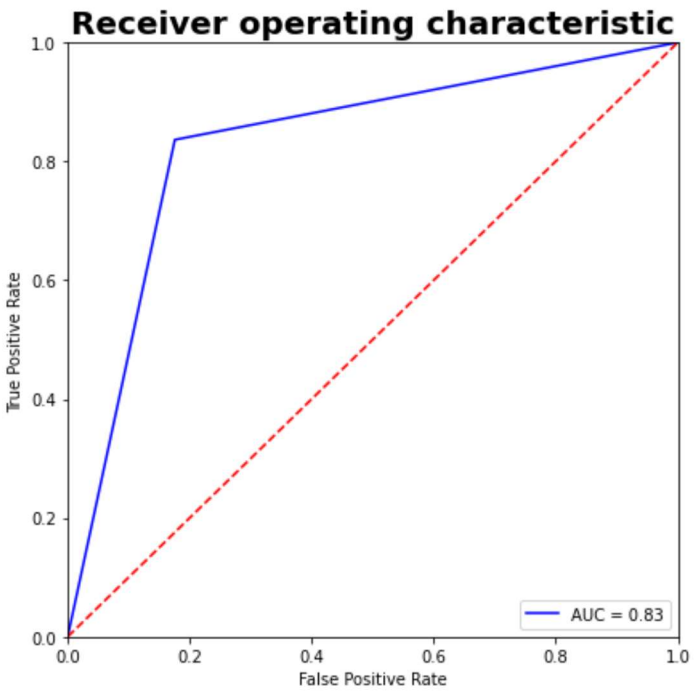
```
['FastingBS' 'MaxHR' 'Oldpeak' 'Sex_F' 'Sex_M' 'ChestPainType_ASY'
 'ExerciseAngina_N' 'ExerciseAngina_Y' 'ST_Slope_Flat' 'ST_Slope_Up']
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\base.py:445: UserWarning: X does not have valid feature names, bu
t SequentialFeatureSelector was fitted with feature names
  warnings.warn(
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\base.py:445: UserWarning: X does not have valid feature names, bu
t SequentialFeatureSelector was fitted with feature names
  warnings.warn(
              precision    recall  f1-score   support

           0       0.77      0.82      0.80        74
           1       0.88      0.84      0.86       110
```

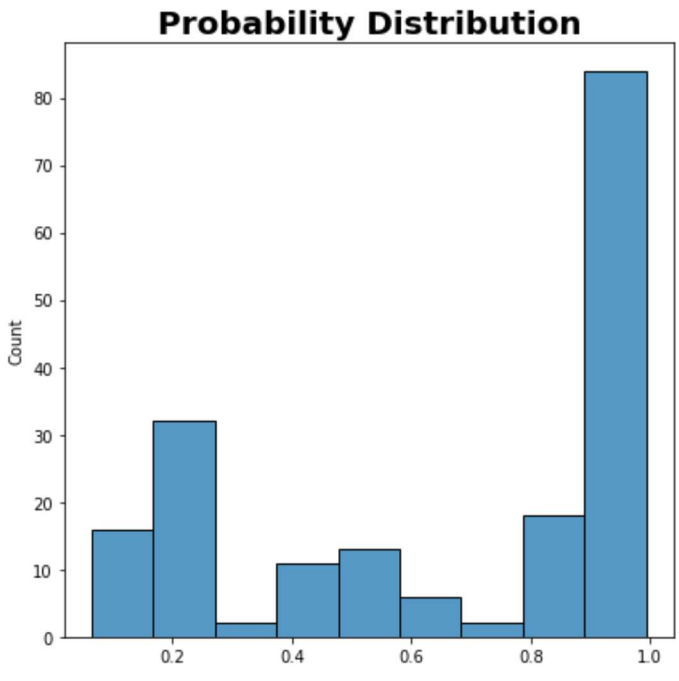
```
    accuracy                           0.83       184
   macro avg       0.82      0.83      0.83       184
weighted avg       0.83      0.83      0.83       184

Specificity = 0.8243243243243243
The balanced accuracy score is : 83.03%
```

SVC Data1 FS









XGBoost

As stated previously in the notebook, the dataset is unbalanced where 55% had heart disease and 44% were normal. One of the benefits of using XGBoost is that it has an attribute that can take the unbalanced nature of the dataset into consideration and yield a more balanced generalizable model. As seen in the results, XGBoost achieves comparable performance metrics with balanced ratio in the confusion matrix between correctly identified classes.

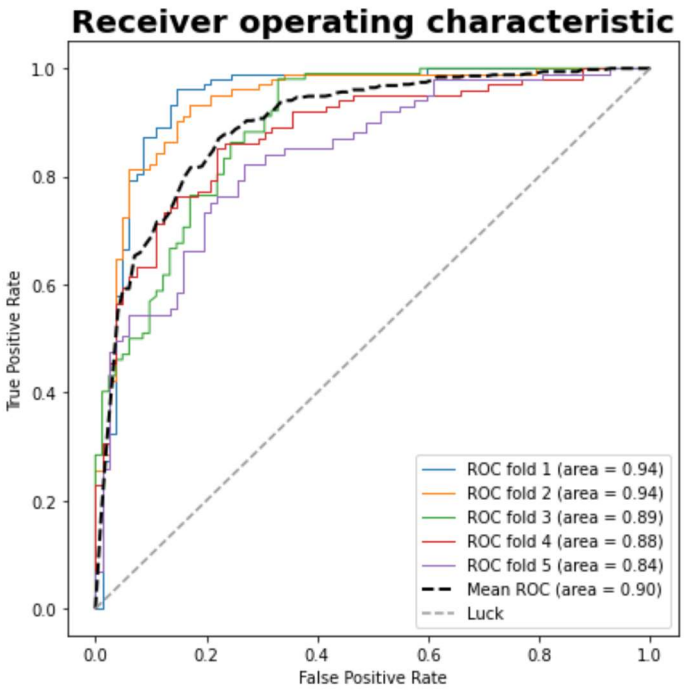
In [35]:

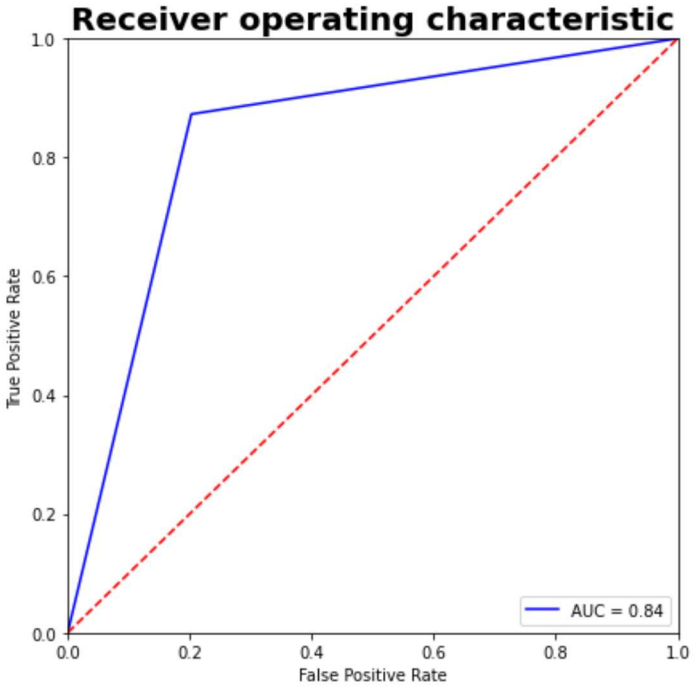
```
from xgboost import XGBClassifier
scale_pos = y.value_counts()[1]/y.value_counts()[0]
```

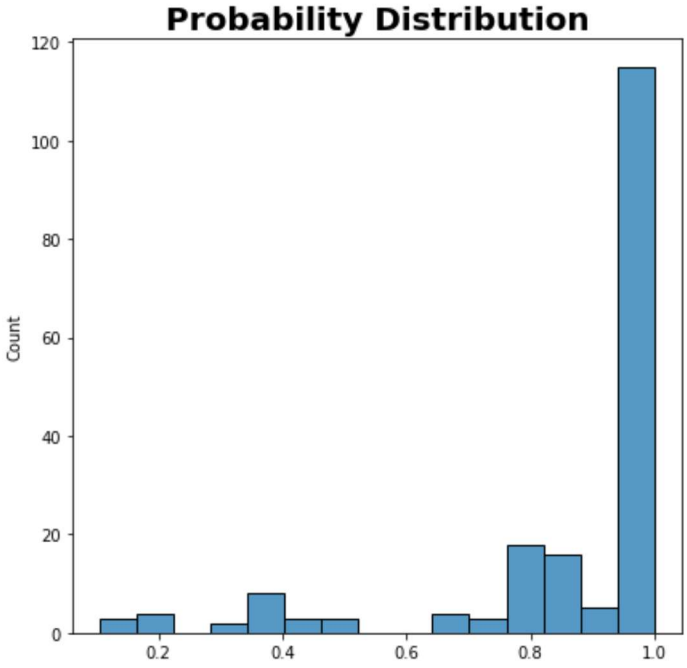
```
xgbc = XGBClassifier(objective='binary:logistic',n_jobs=-1,scale_pos_weight=scale_pos)
build_model(xgbc,X1_train,y1_train,X1_test,y1_test,data1,y,label='Data1 Default')
```

```
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\xgboost\sklearn.py:1224: UserWarning: The use of label encoder in XGBClassifier is deprecated and will be removed in a future release. To remove this warning, do the following: 1) Pass option use_label_encoder=False when constructing XGBClassifier object; and 2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
[20:10:16] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_1.5.0/src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore the old behavior.
[20:10:16] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_1.5.0/src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore the old behavior.
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\xgboost\sklearn.py:1224: UserWarning: The use of label encoder in XGBClassifier is deprecated and will be removed in a future release. To remove this warning, do the following: 1) Pass option use_label_encoder=False when constructing XGBClassifier object; and 2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\xgboost\sklearn.py:1224: UserWarning: The use of label encoder in XGBClassifier is deprecated and will be removed in a future release. To remove this warning, do the following: 1) Pass option use_label_encoder=False when constructing XGBClassifier object; and 2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
[20:10:16] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_1.5.0/src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore the old behavior.
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\xgboost\sklearn.py:1224: UserWarning: The use of label encoder in XGBClassifier is deprecated and will be removed in a future release. To remove this warning, do the following: 1) Pass option use_label_encoder=False when constructing XGBClassifier object; and 2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
[20:10:16] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_1.5.0/src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore the old behavior.
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\xgboost\sklearn.py:1224: UserWarning: The use of label encoder in XGBClassifier is deprecated and will be removed in a future release. To remove this warning, do the following: 1) Pass option use_label_encoder=False when constructing XGBClassifier object; and 2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
[20:10:17] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_1.5.0/src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore the old behavior.
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\xgboost\sklearn.py:1224: UserWarning: The use of label encoder in XGBClassifier is deprecated and will be removed in a future release. To remove this warning, do the following: 1) Pass option use_label_encoder=False when constructing XGBClassifier object; and 2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
[20:10:17] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_1.5.0/src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore the old behavior.
              precision    recall  f1-score   support

           0       0.81      0.80      0.80        74
           1       0.86      0.87      0.87       110

    accuracy                           0.84       184
   macro avg       0.84      0.84      0.84       184
weighted avg       0.84      0.84      0.84       184

Specificity = 0.7972972972972973
The balanced accuracy score is : 83.50%
```

XGBClassifier Data1 Default





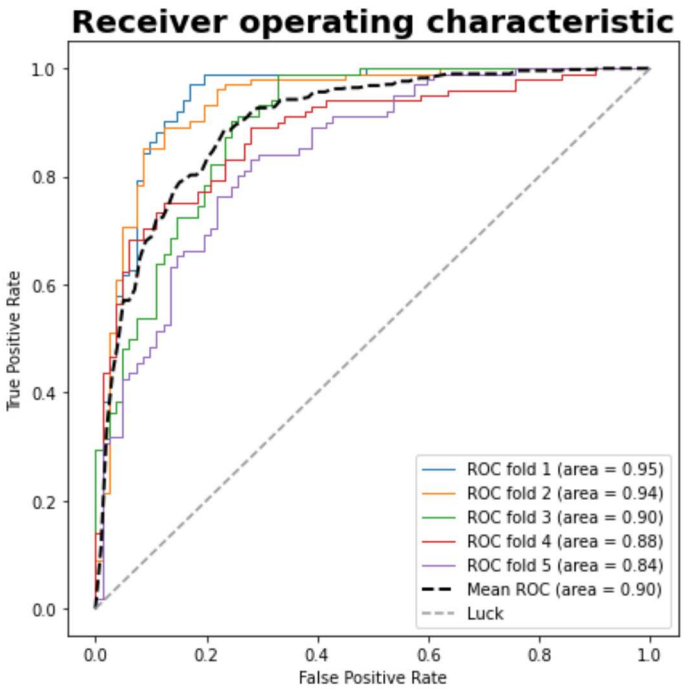


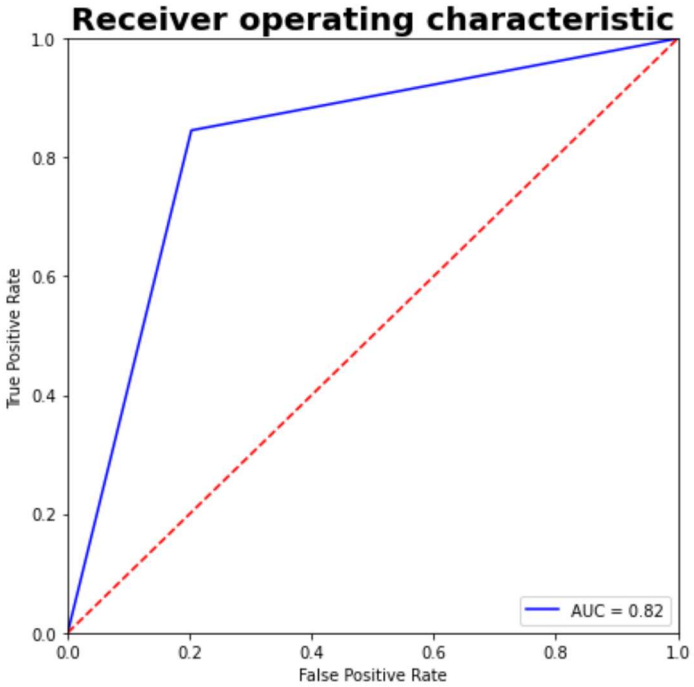
In [36]:
```
xgbc = XGBClassifier(objective='binary:logistic',n_jobs=-1,scale_pos_weight=scale_pos)
build_model(xgbc,X2_train,y2_train,X2_test,y2_test,data2,y,label='Data2 Default')
```

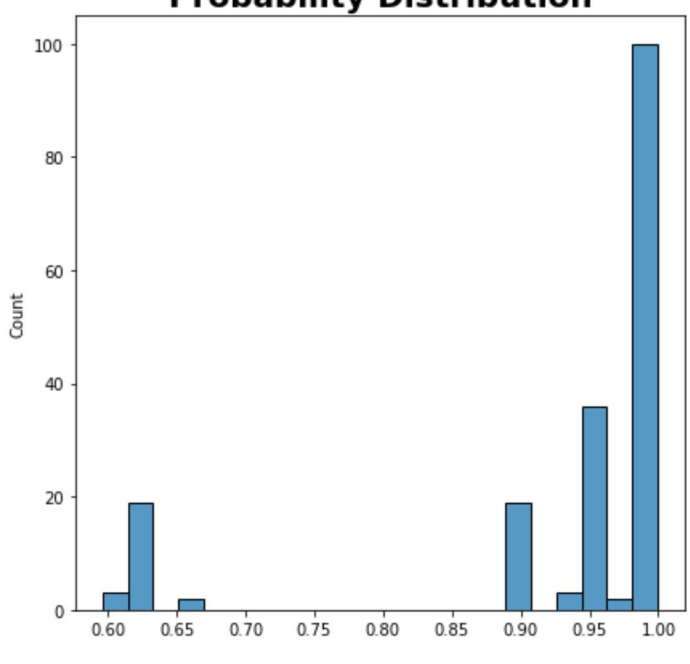
```
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\xgboost\sklearn.py:1224: UserWarning: The use of label encoder in XGBClassifier is deprecated and will be removed in a future release. To remove this warning, do the following: 1) Pass option use_label_encoder=False when constructing XGBClassifier object; and 2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
[20:10:19] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_1.5.0/src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore the old behavior.
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\xgboost\sklearn.py:1224: UserWarning: The use of label encoder in XGBClassifier is deprecated and will be removed in a future release. To remove this warning, do the following: 1) Pass option use_label_encoder=False when constructing XGBClassifier object; and 2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\xgboost\sklearn.py:1224: UserWarning: The use of label encoder in XGBClassifier is deprecated and will be removed in a future release. To remove this warning, do the following: 1) Pass option use_label_encoder=False when constructing XGBClassifier object; and 2) Encode your labels (y) as integers starting with 0, i.e. 0, 1,
```

```
2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
[20:10:19] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_1.5.0/src/learner.cc:1115: Starting in XGBoost 1.3.
0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly se
t eval_metric if you'd like to restore the old behavior.
[20:10:19] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_1.5.0/src/learner.cc:1115: Starting in XGBoost 1.3.
0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly se
t eval_metric if you'd like to restore the old behavior.
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\xgboost\sklearn.py:1224: UserWarning: The use of label encoder in XGBClas
sifier is deprecated and will be removed in a future release. To remove this warning, do the following: 1) Pass option use_labe
l_encoder=False when constructing XGBClassifier object; and 2) Encode your labels (y) as integers starting with 0, i.e. 0, 1,
2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
[20:10:19] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_1.5.0/src/learner.cc:1115: Starting in XGBoost 1.3.
0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly se
t eval_metric if you'd like to restore the old behavior.
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\xgboost\sklearn.py:1224: UserWarning: The use of label encoder in XGBClas
sifier is deprecated and will be removed in a future release. To remove this warning, do the following: 1) Pass option use_labe
l_encoder=False when constructing XGBClassifier object; and 2) Encode your labels (y) as integers starting with 0, i.e. 0, 1,
2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
[20:10:20] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_1.5.0/src/learner.cc:1115: Starting in XGBoost 1.3.
0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly se
t eval_metric if you'd like to restore the old behavior.
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\xgboost\sklearn.py:1224: UserWarning: The use of label encoder in XGBClas
sifier is deprecated and will be removed in a future release. To remove this warning, do the following: 1) Pass option use_labe
l_encoder=False when constructing XGBClassifier object; and 2) Encode your labels (y) as integers starting with 0, i.e. 0, 1,
2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
[20:10:20] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_1.5.0/src/learner.cc:1115: Starting in XGBoost 1.3.
0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly se
t eval_metric if you'd like to restore the old behavior.
              precision    recall  f1-score   support

           0       0.78      0.80      0.79        74
           1       0.86      0.85      0.85       110

    accuracy                           0.83       184
   macro avg       0.82      0.82      0.82       184
weighted avg       0.83      0.83      0.83       184

Specificity = 0.7972972972972973
The balanced accuracy score is : 82.14%
```

XGBClassifier Data2 Default







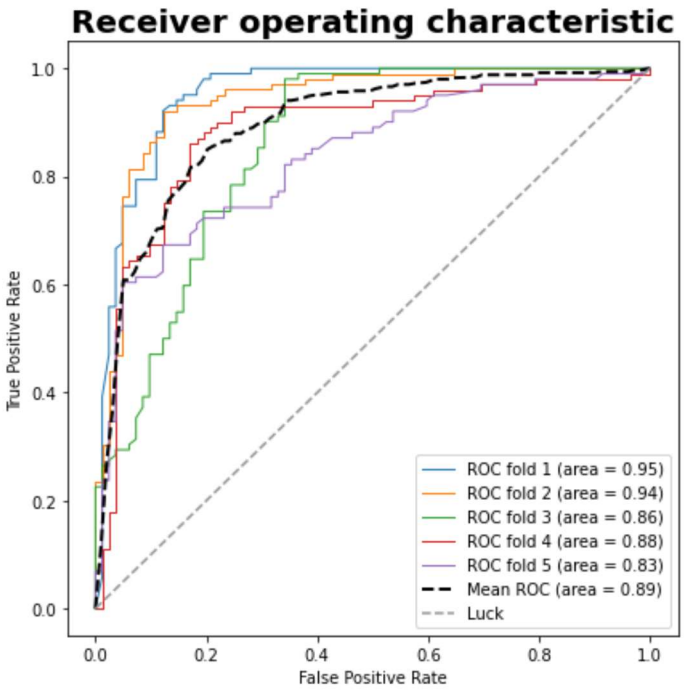
In [37]:

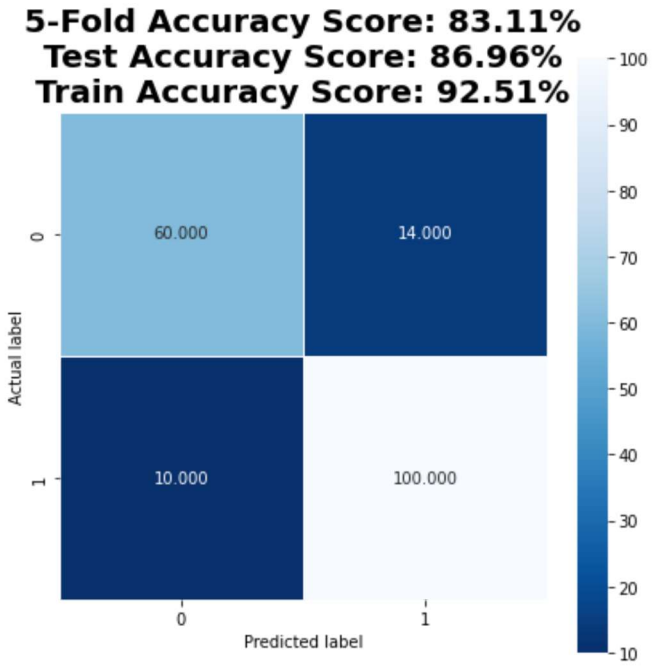
```
xgbc_param = {'gamma': [0,0.1,0.8,1.6,3.2,6.4,12.8,25.6,51.2,102.4, 200],
              'learning_rate': [0.01, 0.03, 0.06, 0.1, 0.15, 0.2, 0.25, 0.4, 0.6, 0.7, 1],
              'max_depth': [5,6,7,8,9,10,11,12,13,14],
              'n_estimators': [50,65,80,100,115,130,150],
              'reg_alpha': [0,0.1,0.8,1.6,3.2,6.4,12.8,51.2,102.4,200],
              'reg_lambda': [0,0.1,0.8,1.6,3.2,6.4,12.8,51.2,102.4,200]}

xgbc = XGBClassifier(objective='binary:logistic',n_jobs=-1,scale_pos_weight=scale_pos)
build_model(xgbc,X1_train,y1_train,X1_test,y1_test,data1,y,label='Data1 Default',params=xgbc_param)
```

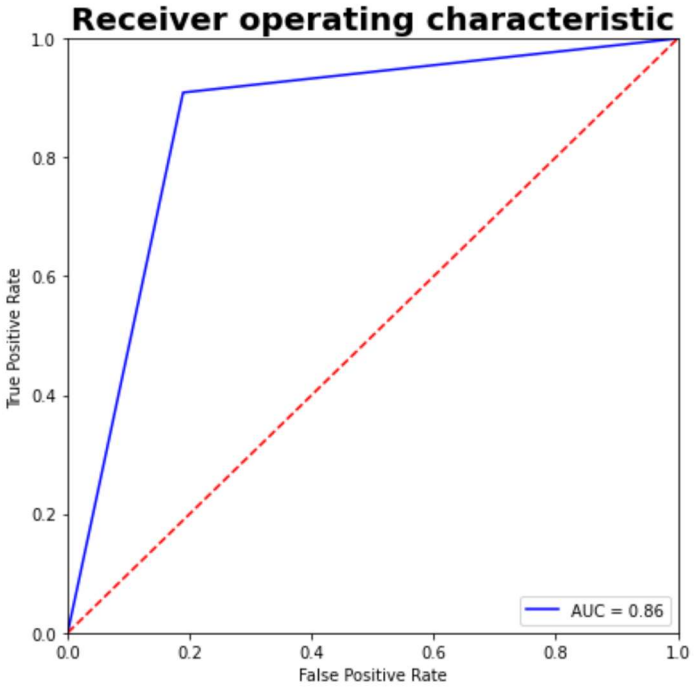
```
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\xgboost\sklearn.py:1224: UserWarning: The use of label encoder in XGBClassifier is deprecated and will be removed in a future release. To remove this warning, do the following: 1) Pass option use_label_encoder=False when constructing XGBClassifier object; and 2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
[20:12:06] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_1.5.0/src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore the old behavior.
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\xgboost\sklearn.py:1224: UserWarning: The use of label encoder in XGBClassifier is deprecated and will be removed in a future release. To remove this warning, do the following: 1) Pass option use_label_encoder=False when constructing XGBClassifier object; and 2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
```

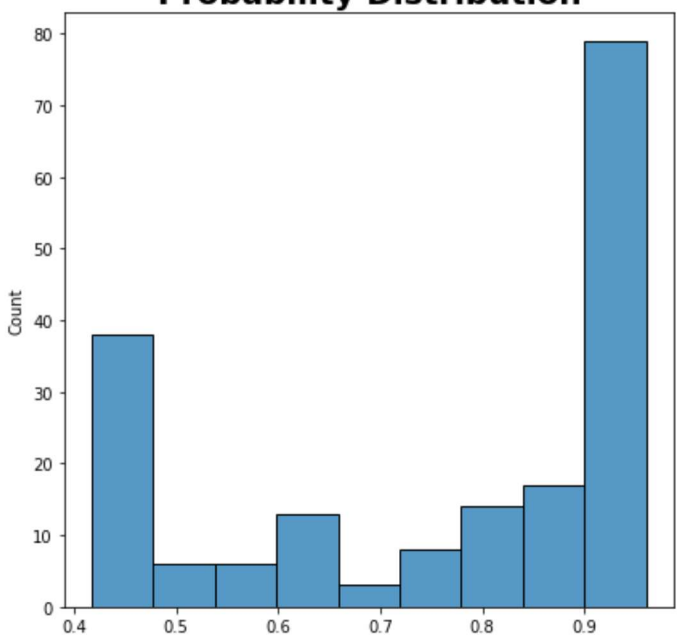
```
sifier is deprecated and will be removed in a future release. To remove this warning, do the following: 1) Pass option use_labe
l_encoder=False when constructing XGBClassifier object; and 2) Encode your labels (y) as integers starting with 0, i.e. 0, 1,
2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
[20:13:21] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_1.5.0/src/learner.cc:1115: Starting in XGBoost 1.3.
0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly se
t eval_metric if you'd like to restore the old behavior.
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\xgboost\sklearn.py:1224: UserWarning: The use of label encoder in XGBClas
sifier is deprecated and will be removed in a future release. To remove this warning, do the following: 1) Pass option use_labe
l_encoder=False when constructing XGBClassifier object; and 2) Encode your labels (y) as integers starting with 0, i.e. 0, 1,
2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
[20:14:28] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_1.5.0/src/learner.cc:1115: Starting in XGBoost 1.3.
0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly se
t eval_metric if you'd like to restore the old behavior.
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\xgboost\sklearn.py:1224: UserWarning: The use of label encoder in XGBClas
sifier is deprecated and will be removed in a future release. To remove this warning, do the following: 1) Pass option use_labe
l_encoder=False when constructing XGBClassifier object; and 2) Encode your labels (y) as integers starting with 0, i.e. 0, 1,
2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
[20:15:33] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_1.5.0/src/learner.cc:1115: Starting in XGBoost 1.3.
0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly se
t eval_metric if you'd like to restore the old behavior.
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\xgboost\sklearn.py:1224: UserWarning: The use of label encoder in XGBClas
sifier is deprecated and will be removed in a future release. To remove this warning, do the following: 1) Pass option use_labe
l_encoder=False when constructing XGBClassifier object; and 2) Encode your labels (y) as integers starting with 0, i.e. 0, 1,
2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
[20:16:10] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_1.5.0/src/learner.cc:1115: Starting in XGBoost 1.3.
0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly se
t eval_metric if you'd like to restore the old behavior.
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\xgboost\sklearn.py:1224: UserWarning: The use of label encoder in XGBClas
sifier is deprecated and will be removed in a future release. To remove this warning, do the following: 1) Pass option use_labe
l_encoder=False when constructing XGBClassifier object; and 2) Encode your labels (y) as integers starting with 0, i.e. 0, 1,
2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
[20:17:29] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_1.5.0/src/learner.cc:1115: Starting in XGBoost 1.3.
0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly se
t eval_metric if you'd like to restore the old behavior.
              precision    recall  f1-score   support

           0       0.86      0.81      0.83        74
           1       0.88      0.91      0.89       110

    accuracy                           0.87       184
   macro avg       0.87      0.86      0.86       184
weighted avg       0.87      0.87      0.87       184

Specificity = 0.8108108108108109
The balanced accuracy score is : 86.00%
```

RandomizedSearchCV XGBClassifier Data1 Default







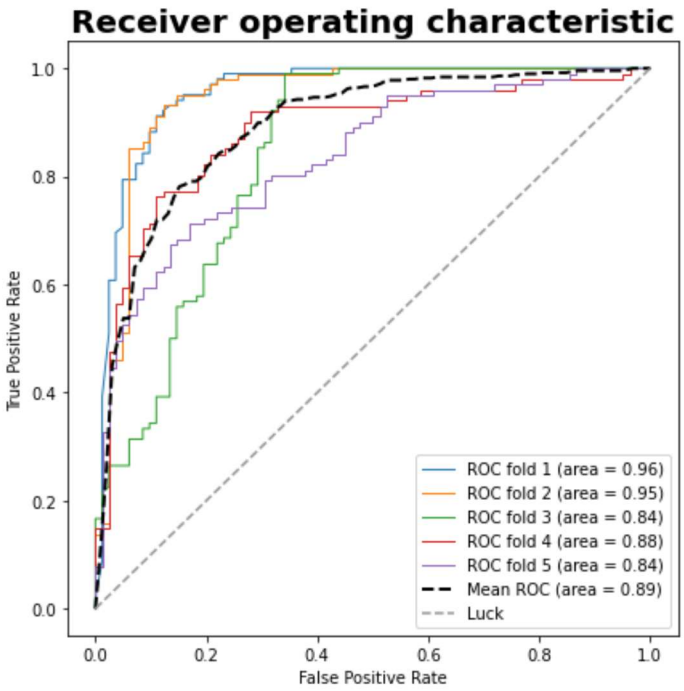


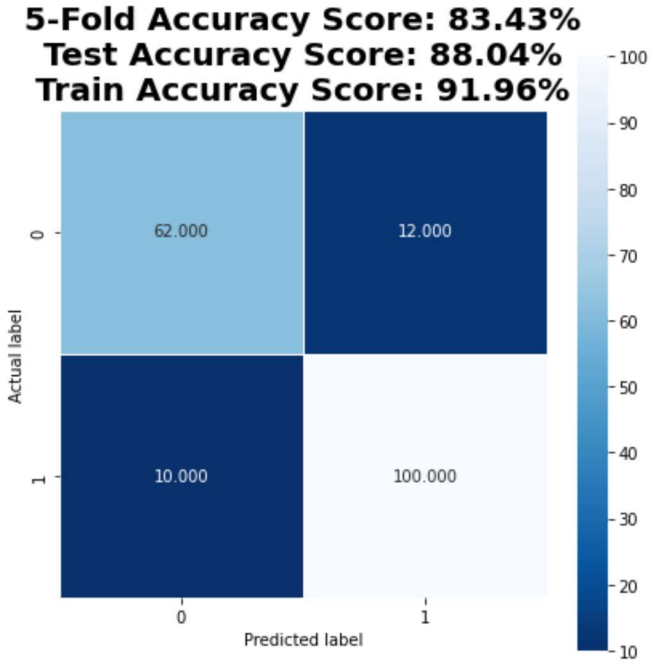
In [38]:
```
xgbc = XGBClassifier(objective='binary:logistic',n_jobs=-1,scale_pos_weight=scale_pos)
build_model(xgbc,X2_train,y2_train,X2_test,y2_test,data2,y,label='Data2 Default',params=xgbc_param)
```

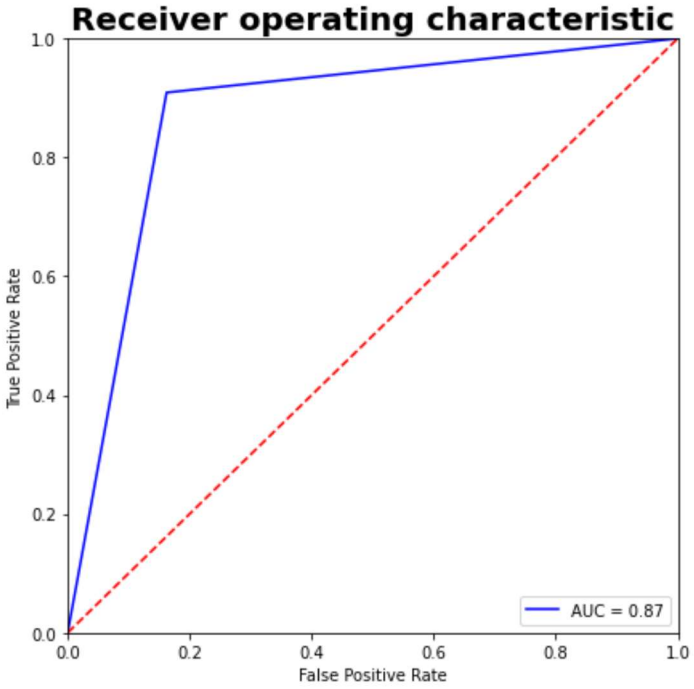
```
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\xgboost\sklearn.py:1224: UserWarning: The use of label encoder in XGBClassifier is deprecated and will be removed in a future release. To remove this warning, do the following: 1) Pass option use_label_encoder=False when constructing XGBClassifier object; and 2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
[20:19:08] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_1.5.0/src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore the old behavior.
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\xgboost\sklearn.py:1224: UserWarning: The use of label encoder in XGBClassifier is deprecated and will be removed in a future release. To remove this warning, do the following: 1) Pass option use_label_encoder=False when constructing XGBClassifier object; and 2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
[20:19:57] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_1.5.0/src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore the old behavior.
```

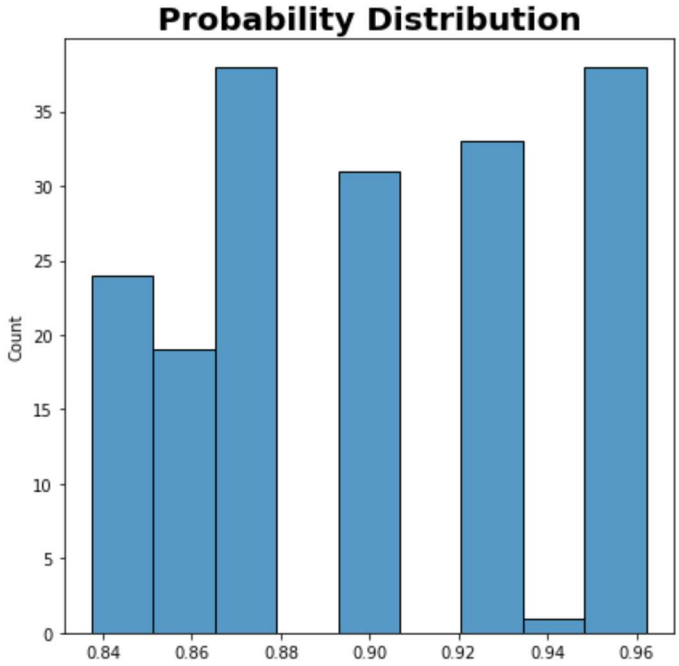
```
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\xgboost\sklearn.py:1224: UserWarning: The use of label encoder in XGBClassifier is deprecated and will be removed in a future release. To remove this warning, do the following: 1) Pass option use_label_encoder=False when constructing XGBClassifier object; and 2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
[20:21:24] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_1.5.0/src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore the old behavior.
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\xgboost\sklearn.py:1224: UserWarning: The use of label encoder in XGBClassifier is deprecated and will be removed in a future release. To remove this warning, do the following: 1) Pass option use_label_encoder=False when constructing XGBClassifier object; and 2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
[20:22:36] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_1.5.0/src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore the old behavior.
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\xgboost\sklearn.py:1224: UserWarning: The use of label encoder in XGBClassifier is deprecated and will be removed in a future release. To remove this warning, do the following: 1) Pass option use_label_encoder=False when constructing XGBClassifier object; and 2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
[20:23:48] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_1.5.0/src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore the old behavior.
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\xgboost\sklearn.py:1224: UserWarning: The use of label encoder in XGBClassifier is deprecated and will be removed in a future release. To remove this warning, do the following: 1) Pass option use_label_encoder=False when constructing XGBClassifier object; and 2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
[20:25:00] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_1.5.0/src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore the old behavior.
              precision    recall  f1-score   support

           0       0.86      0.84      0.85        74
           1       0.89      0.91      0.90       110

    accuracy                           0.88       184
   macro avg       0.88      0.87      0.88       184
weighted avg       0.88      0.88      0.88       184

Specificity = 0.8378378378378378
The balanced accuracy score is : 87.35%
```

RandomizedSearchCV XGBClassifier Data2 Default









XGBClassifier Feature Selection - Sequential Feature Selection

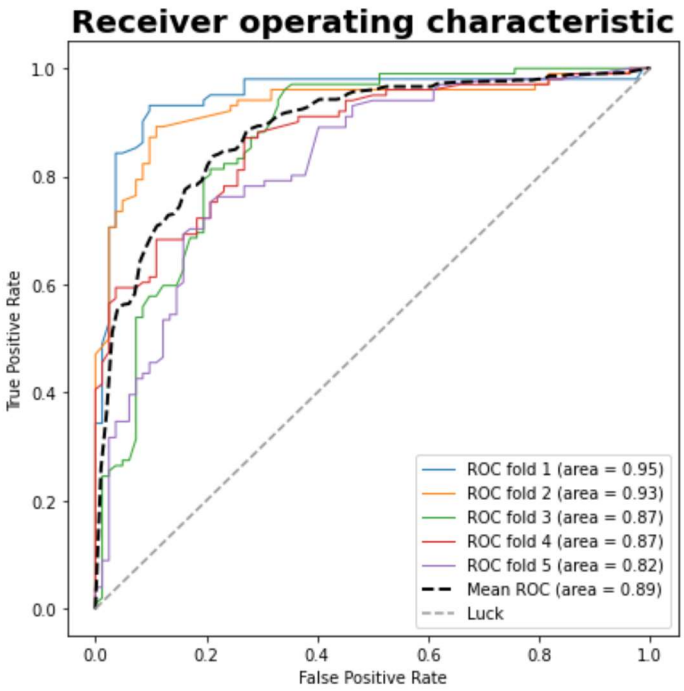
The same five features are selected again by way of gradient boosting.

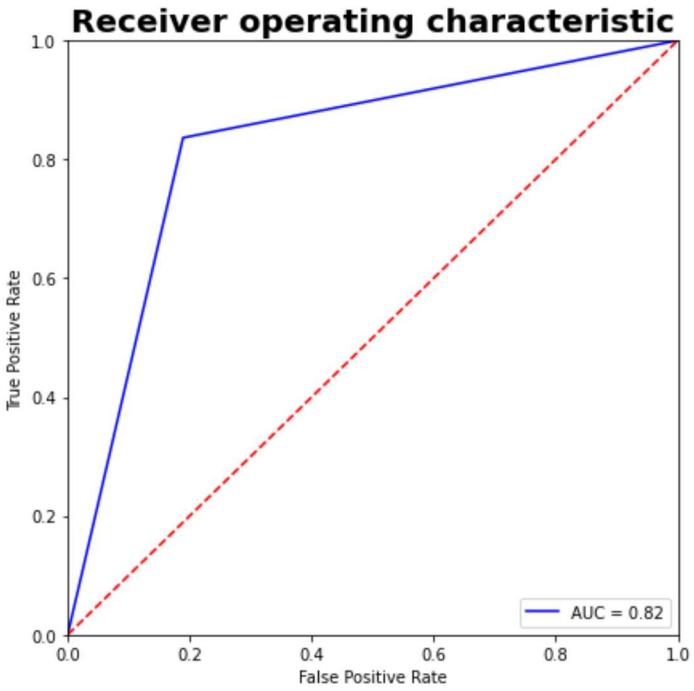
In [39]:
```
xgbc = XGBClassifier(objective='binary:logistic',n_jobs=-1,scale_pos_weight=scale_pos)
sfs = SequentialFeatureSelector(xgbc,n_jobs=-1,cv=cv,scoring='roc_auc')
sfs.fit(data2, y)
print(sfs.get_feature_names_out())
build_model(xgbc,sfs.transform(X2_train),y2_train,sfs.transform(X2_test),y2_test,
            pd.DataFrame(sfs.transform(data2)),y,label='Data2 FS')
```

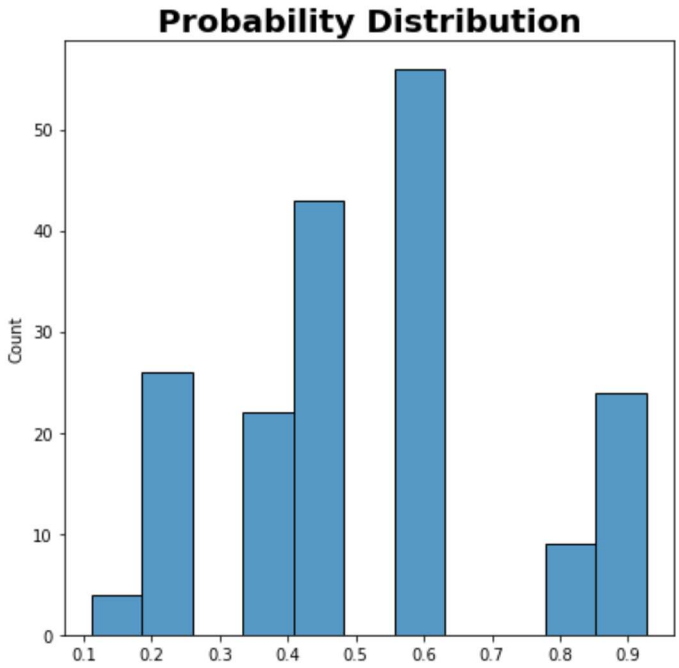
```
['Sex' 'ChestPainType' 'FastingBS' 'Oldpeak' 'ST_Slope']
[20:26:11] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_1.5.0/src/learner.cc:1115: Starting in XGBoost 1.3.
0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly se
t eval_metric if you'd like to restore the old behavior.
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\base.py:445: UserWarning: X does not have valid feature names, bu
t SequentialFeatureSelector was fitted with feature names
  warnings.warn(
```

```
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\base.py:445: UserWarning: X does not have valid feature names, bu
t SequentialFeatureSelector was fitted with feature names
  warnings.warn(
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\xgboost\sklearn.py:1224: UserWarning: The use of label encoder in XGBClas
sifier is deprecated and will be removed in a future release. To remove this warning, do the following: 1) Pass option use_labe
l_encoder=False when constructing XGBClassifier object; and 2) Encode your labels (y) as integers starting with 0, i.e. 0, 1,
2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\xgboost\sklearn.py:1224: UserWarning: The use of label encoder in XGBClas
sifier is deprecated and will be removed in a future release. To remove this warning, do the following: 1) Pass option use_labe
l_encoder=False when constructing XGBClassifier object; and 2) Encode your labels (y) as integers starting with 0, i.e. 0, 1,
2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
[20:26:14] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_1.5.0/src/learner.cc:1115: Starting in XGBoost 1.3.
0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly se
t eval_metric if you'd like to restore the old behavior.
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\xgboost\sklearn.py:1224: UserWarning: The use of label encoder in XGBClas
sifier is deprecated and will be removed in a future release. To remove this warning, do the following: 1) Pass option use_labe
l_encoder=False when constructing XGBClassifier object; and 2) Encode your labels (y) as integers starting with 0, i.e. 0, 1,
2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
[20:26:18] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_1.5.0/src/learner.cc:1115: Starting in XGBoost 1.3.
0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly se
t eval_metric if you'd like to restore the old behavior.
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\xgboost\sklearn.py:1224: UserWarning: The use of label encoder in XGBClas
sifier is deprecated and will be removed in a future release. To remove this warning, do the following: 1) Pass option use_labe
l_encoder=False when constructing XGBClassifier object; and 2) Encode your labels (y) as integers starting with 0, i.e. 0, 1,
2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
[20:26:27] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_1.5.0/src/learner.cc:1115: Starting in XGBoost 1.3.
0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly se
t eval_metric if you'd like to restore the old behavior.
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\xgboost\sklearn.py:1224: UserWarning: The use of label encoder in XGBClas
sifier is deprecated and will be removed in a future release. To remove this warning, do the following: 1) Pass option use_labe
l_encoder=False when constructing XGBClassifier object; and 2) Encode your labels (y) as integers starting with 0, i.e. 0, 1,
2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
[20:26:27] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_1.5.0/src/learner.cc:1115: Starting in XGBoost 1.3.
0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly se
t eval_metric if you'd like to restore the old behavior.
[20:26:27] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_1.5.0/src/learner.cc:1115: Starting in XGBoost 1.3.
0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly se
t eval_metric if you'd like to restore the old behavior.
              precision    recall  f1-score   support

           0       0.77      0.81      0.79        74
           1       0.87      0.84      0.85       110

    accuracy                           0.83       184
   macro avg       0.82      0.82      0.82       184
weighted avg       0.83      0.83      0.83       184

Specificity = 0.8108108108108109
The balanced accuracy score is : 82.36%
```

XGBClassifier Data2 FS







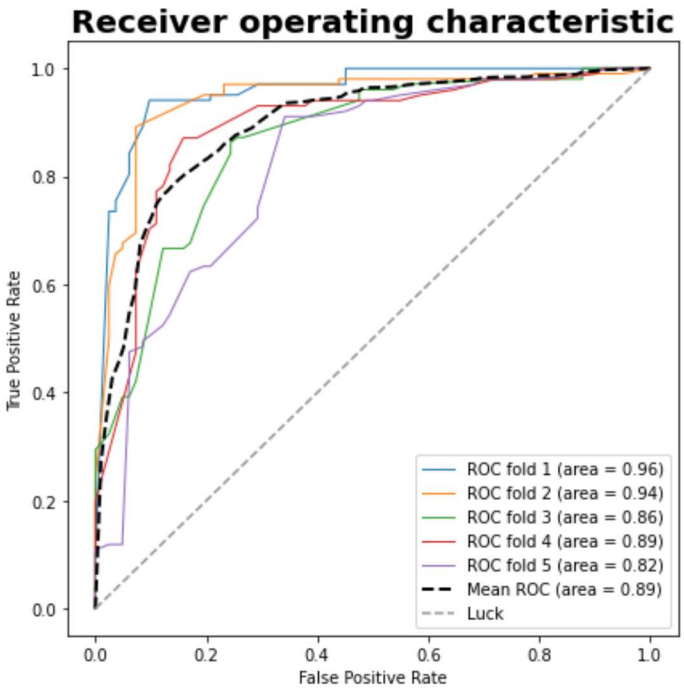
In [40]:

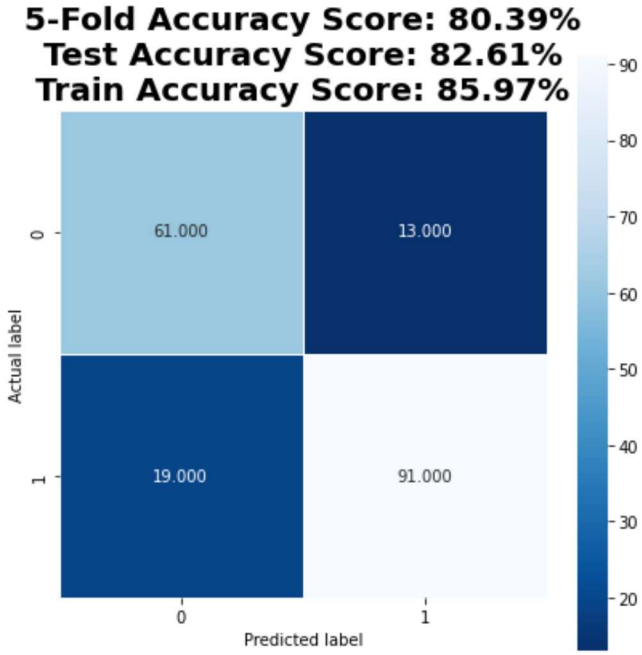
```
xgbc = XGBClassifier(objective='binary:logistic',n_jobs=-1,scale_pos_weight=scale_pos)
sfs = SequentialFeatureSelector(xgbc,n_jobs=-1,cv=cv,scoring='roc_auc')
sfs.fit(data1, y)
print(sfs.get_feature_names_out())
build_model(xgbc,sfs.transform(X1_train),y1_train,sfs.transform(X1_test),y1_test,
            pd.DataFrame(sfs.transform(data1)),y,label='Data1 FS')
```

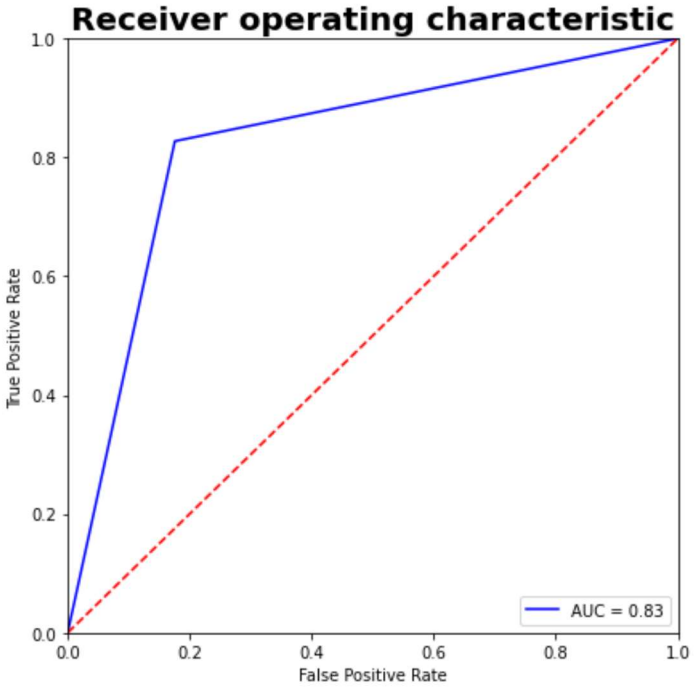
```
['FastingBS' 'Sex_F' 'Sex_M' 'ChestPainType_ASY' 'ChestPainType_ATA'
 'ChestPainType_NAP' 'ChestPainType_TA' 'ST_Slope_Down' 'ST_Slope_Flat'
 'ST_Slope_Up']
[20:28:35] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_1.5.0/src/learner.cc:1115: Starting in XGBoost 1.3.
0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly se
t eval_metric if you'd like to restore the old behavior.
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\base.py:445: UserWarning: X does not have valid feature names, bu
t SequentialFeatureSelector was fitted with feature names
  warnings.warn(
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\base.py:445: UserWarning: X does not have valid feature names, bu
t SequentialFeatureSelector was fitted with feature names
  warnings.warn(
```

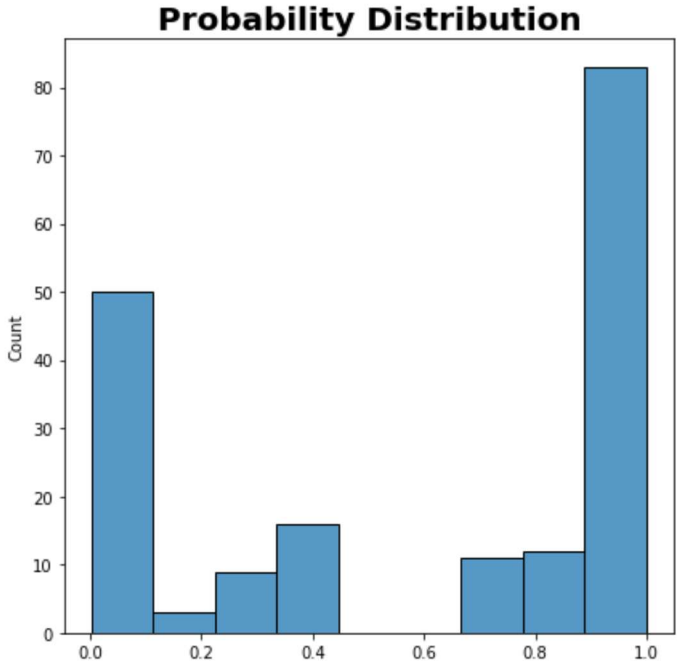
```
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\xgboost\sklearn.py:1224: UserWarning: The use of label encoder in XGBClassifier is deprecated and will be removed in a future release. To remove this warning, do the following: 1) Pass option use_label_encoder=False when constructing XGBClassifier object; and 2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\xgboost\sklearn.py:1224: UserWarning: The use of label encoder in XGBClassifier is deprecated and will be removed in a future release. To remove this warning, do the following: 1) Pass option use_label_encoder=False when constructing XGBClassifier object; and 2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
[20:28:36] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_1.5.0/src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore the old behavior.
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\xgboost\sklearn.py:1224: UserWarning: The use of label encoder in XGBClassifier is deprecated and will be removed in a future release. To remove this warning, do the following: 1) Pass option use_label_encoder=False when constructing XGBClassifier object; and 2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\xgboost\sklearn.py:1224: UserWarning: The use of label encoder in XGBClassifier is deprecated and will be removed in a future release. To remove this warning, do the following: 1) Pass option use_label_encoder=False when constructing XGBClassifier object; and 2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
[20:28:37] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_1.5.0/src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore the old behavior.
[20:28:37] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_1.5.0/src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore the old behavior.
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\xgboost\sklearn.py:1224: UserWarning: The use of label encoder in XGBClassifier is deprecated and will be removed in a future release. To remove this warning, do the following: 1) Pass option use_label_encoder=False when constructing XGBClassifier object; and 2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
[20:28:37] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_1.5.0/src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore the old behavior.
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\xgboost\sklearn.py:1224: UserWarning: The use of label encoder in XGBClassifier is deprecated and will be removed in a future release. To remove this warning, do the following: 1) Pass option use_label_encoder=False when constructing XGBClassifier object; and 2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
[20:28:38] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_1.5.0/src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore the old behavior.
              precision    recall  f1-score   support

           0       0.76      0.82      0.79        74
           1       0.88      0.83      0.85       110

    accuracy                           0.83       184
   macro avg       0.82      0.83      0.82       184
weighted avg       0.83      0.83      0.83       184

Specificity = 0.8243243243243243
The balanced accuracy score is : 82.58%
```

XGBClassifier Data1 FS









LightGBM

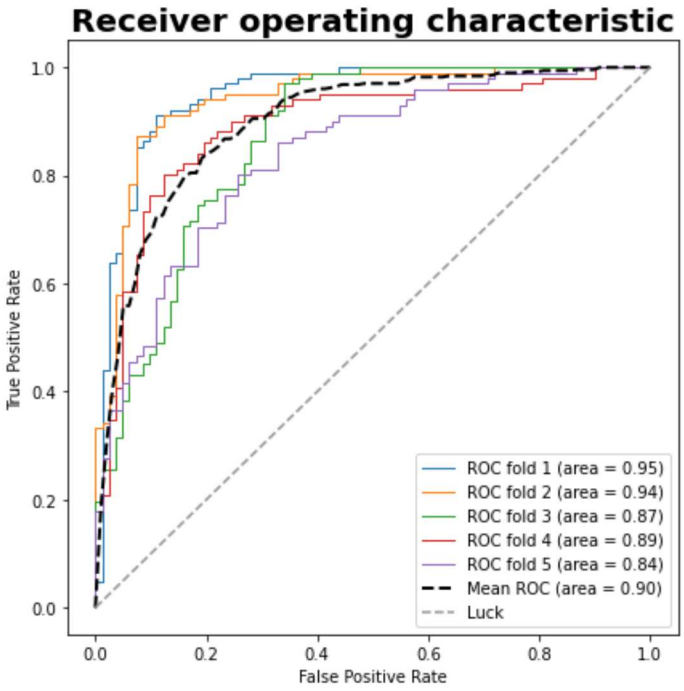
LightGBM is a new classifier that has not been introduced in class. It is a tree based algorithm that focuses on efficiency and accuracy. It was one of the most efficient models with the shorted training time. LightGBM also scores comparably to the other models and also appears to be balanced between both classes as shown in the confusion matrix and probability distribution.

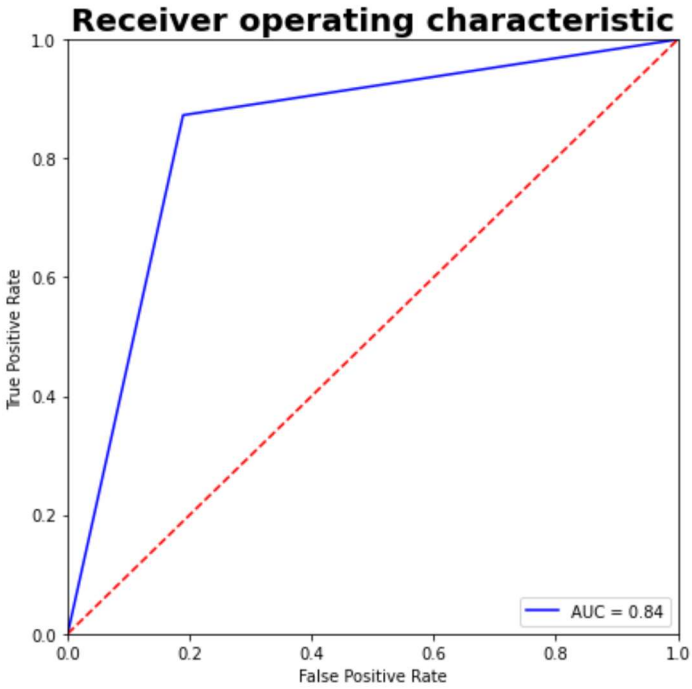
In [41]:
```
from lightgbm import LGBMClassifier

lgbm = LGBMClassifier(n_jobs=-1,random_state=32)
build_model(lgbm,X1_train,y1_train,X1_test,y1_test,data1,y,label='Data1 Default')
```

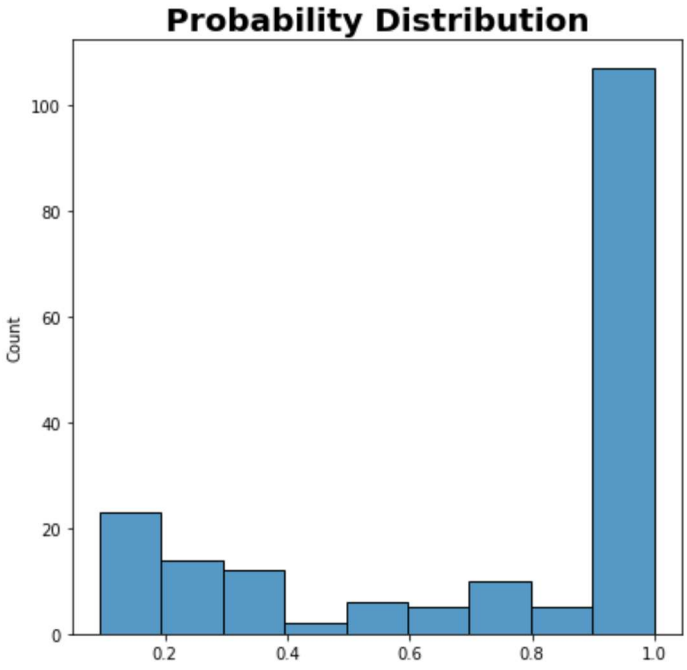
```
              precision    recall  f1-score   support

           0       0.81      0.81      0.81        74
           1       0.87      0.87      0.87       110

    accuracy                           0.85       184
```

```
   macro avg       0.84      0.84      0.84       184
weighted avg       0.85      0.85      0.85       184

Specificity = 0.8108108108108109
The balanced accuracy score is : 84.18%
```

LGBMClassifier Data1 Default



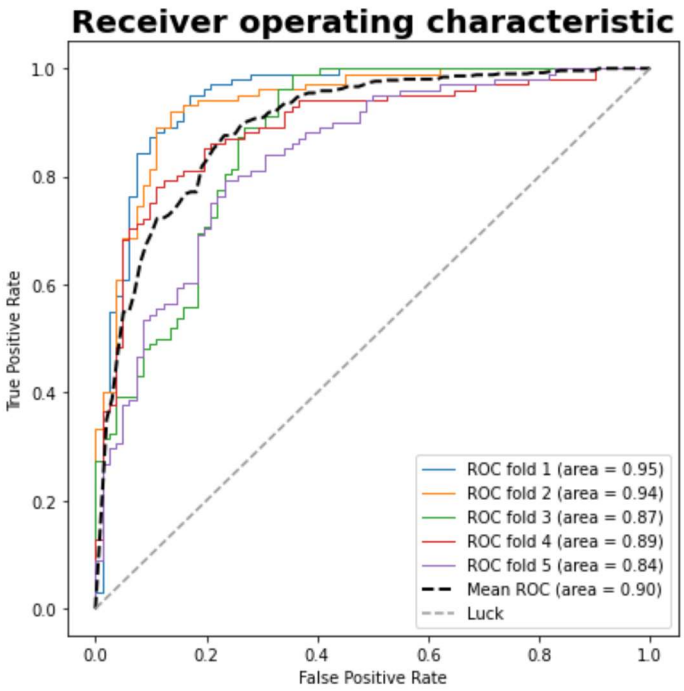


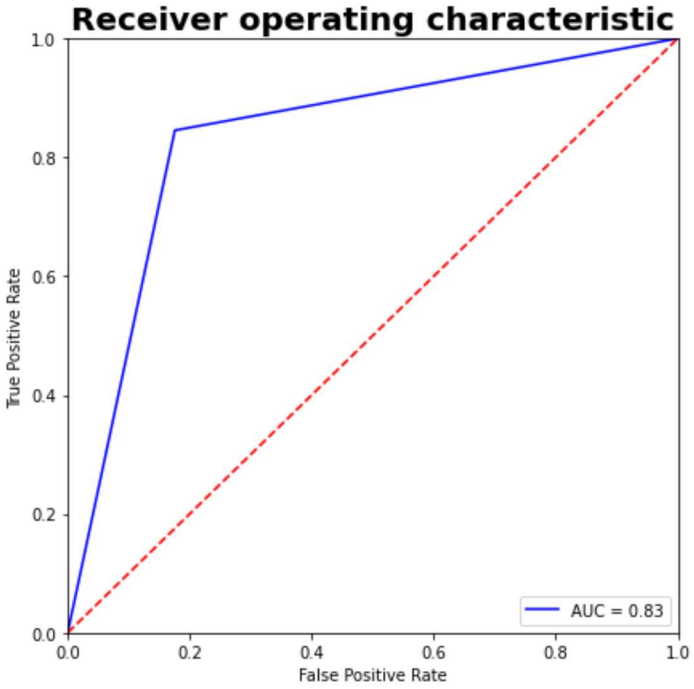


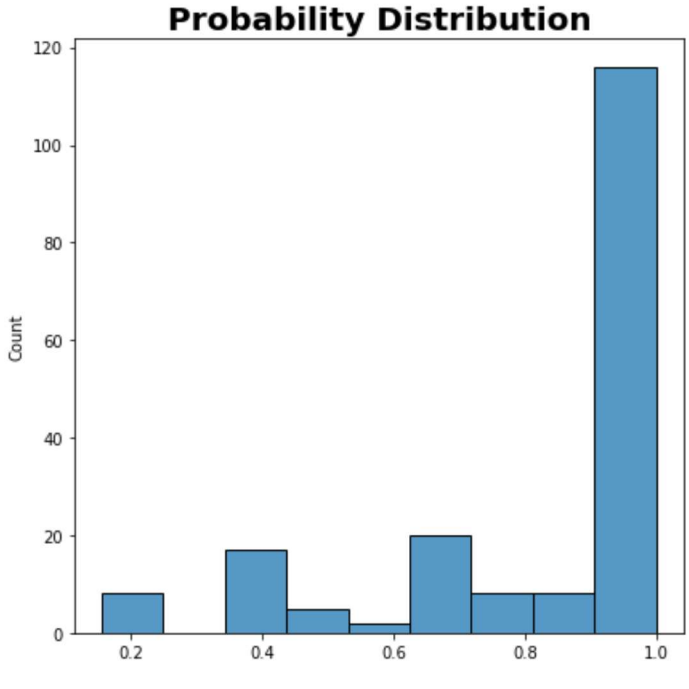
In [42]:
```
lgbm = LGBMClassifier(n_jobs=-1,random_state=32)
build_model(lgbm,X2_train,y2_train,X2_test,y2_test,data2,y,label='Data2 Default')
```

```
              precision    recall  f1-score   support

           0       0.78      0.82      0.80        74
           1       0.88      0.85      0.86       110

    accuracy                           0.84       184
   macro avg       0.83      0.83      0.83       184
weighted avg       0.84      0.84      0.84       184

Specificity = 0.8243243243243243
The balanced accuracy score is : 83.49%
```

LGBMClassifier Data2 Default







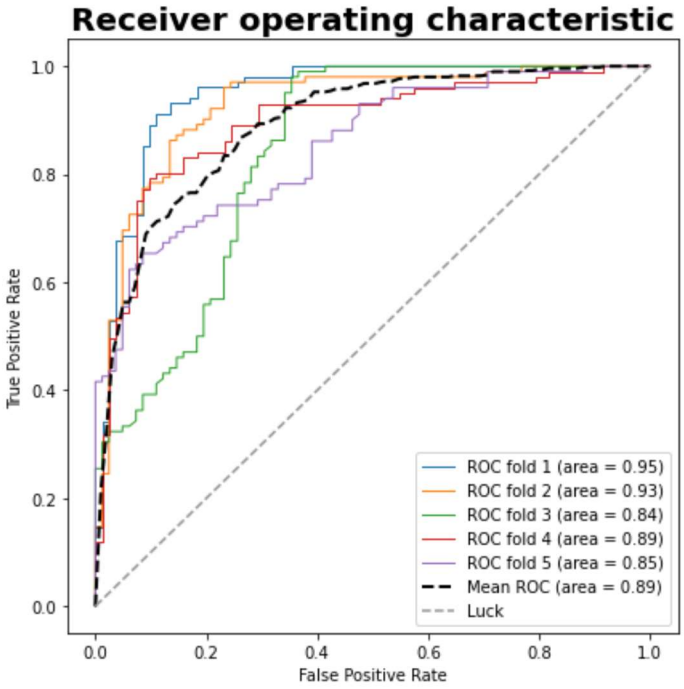
In [43]:

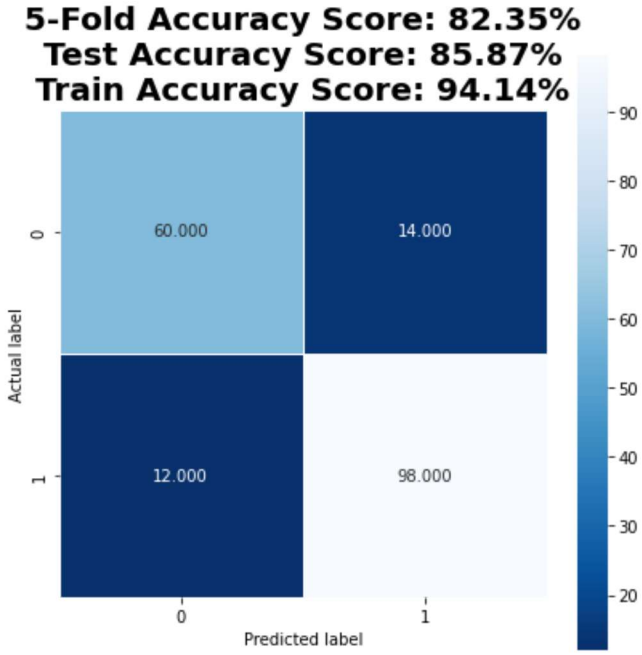
```
lgbm_grid = {
    'num_leaves': [3,10,100,1000],
    'reg_alpha': [0.001,.01,.1,1,10,100],
    'min_data_in_leaf': [1, 5, 10, 50, 100, 250],
    'lambda_l1': [0.001,.01,.1,1,10],
    'lambda_l2': [0.001,.01,.1,1,10]
    }

lgbm = LGBMClassifier(n_jobs=-1,random_state=32)
build_model(lgbm,X1_train,y1_train,X1_test,y1_test,data1,y,label='Data1',params=lgbm_grid)
```

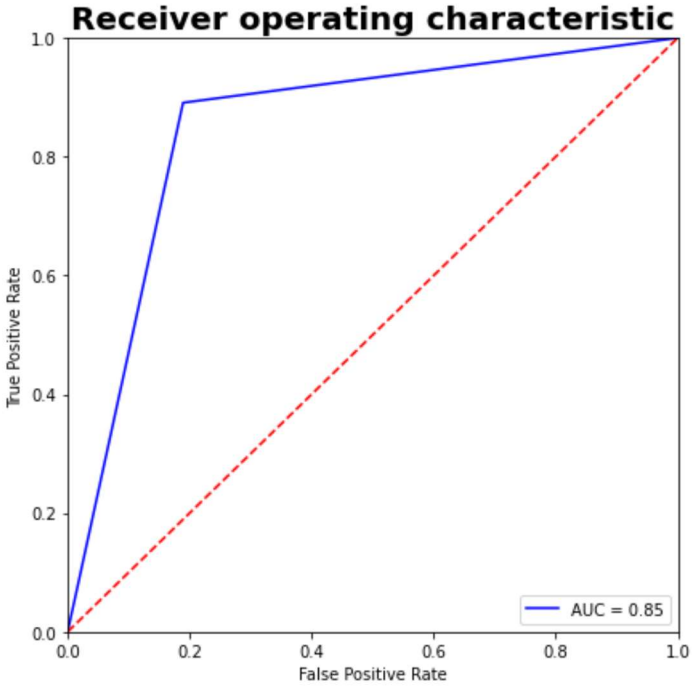
```
[LightGBM] [Warning] min_data_in_leaf is set=10, min_child_samples=20 will be ignored. Current value: min_data_in_leaf=10
[LightGBM] [Warning] lambda_l1 is set=0.1, reg_alpha=1 will be ignored. Current value: lambda_l1=0.1
[LightGBM] [Warning] lambda_l2 is set=10, reg_lambda=0.0 will be ignored. Current value: lambda_l2=10
[LightGBM] [Warning] min_data_in_leaf is set=100, min_child_samples=20 will be ignored. Current value: min_data_in_leaf=100
[LightGBM] [Warning] lambda_l1 is set=0.001, reg_alpha=1 will be ignored. Current value: lambda_l1=0.001
[LightGBM] [Warning] lambda_l2 is set=10, reg_lambda=0.0 will be ignored. Current value: lambda_l2=10
[LightGBM] [Warning] min_data_in_leaf is set=50, min_child_samples=20 will be ignored. Current value: min_data_in_leaf=50
[LightGBM] [Warning] lambda_l1 is set=0.1, reg_alpha=0.001 will be ignored. Current value: lambda_l1=0.1
```

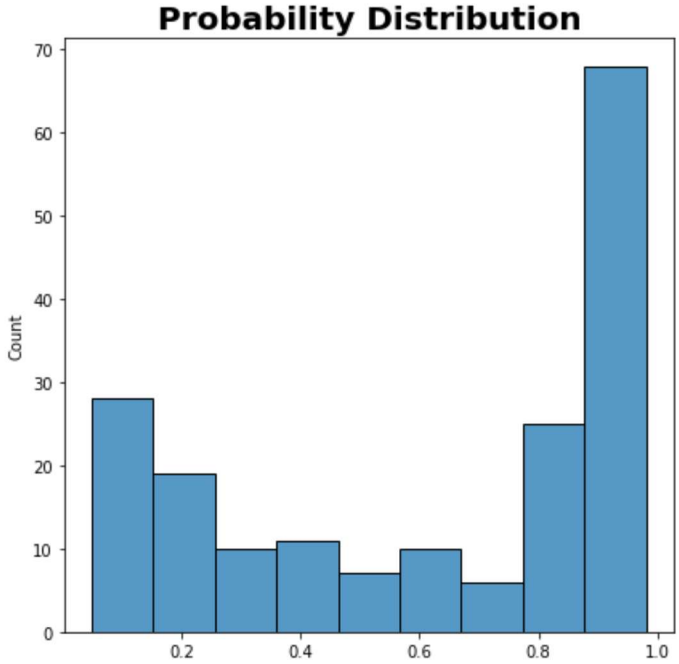
```
[LightGBM] [Warning] lambda_l2 is set=0.001, reg_lambda=0.0 will be ignored. Current value: lambda_l2=0.001
[LightGBM] [Warning] min_data_in_leaf is set=5, min_child_samples=20 will be ignored. Current value: min_data_in_leaf=5
[LightGBM] [Warning] lambda_l1 is set=0.001, reg_alpha=0.01 will be ignored. Current value: lambda_l1=0.001
[LightGBM] [Warning] lambda_l2 is set=0.01, reg_lambda=0.0 will be ignored. Current value: lambda_l2=0.01
[LightGBM] [Warning] min_data_in_leaf is set=5, min_child_samples=20 will be ignored. Current value: min_data_in_leaf=5
[LightGBM] [Warning] lambda_l1 is set=0.1, reg_alpha=10 will be ignored. Current value: lambda_l1=0.1
[LightGBM] [Warning] lambda_l2 is set=1, reg_lambda=0.0 will be ignored. Current value: lambda_l2=1
[LightGBM] [Warning] min_data_in_leaf is set=50, min_child_samples=20 will be ignored. Current value: min_data_in_leaf=50
[LightGBM] [Warning] lambda_l1 is set=1, reg_alpha=0.1 will be ignored. Current value: lambda_l1=1
[LightGBM] [Warning] lambda_l2 is set=0.01, reg_lambda=0.0 will be ignored. Current value: lambda_l2=0.01
              precision    recall  f1-score   support

           0       0.83      0.81      0.82        74
           1       0.88      0.89      0.88       110

    accuracy                           0.86       184
   macro avg       0.85      0.85      0.85       184
weighted avg       0.86      0.86      0.86       184

Specificity = 0.8108108108108109
The balanced accuracy score is : 85.09%
```

RandomizedSearchCV LGBMClassifier Data1







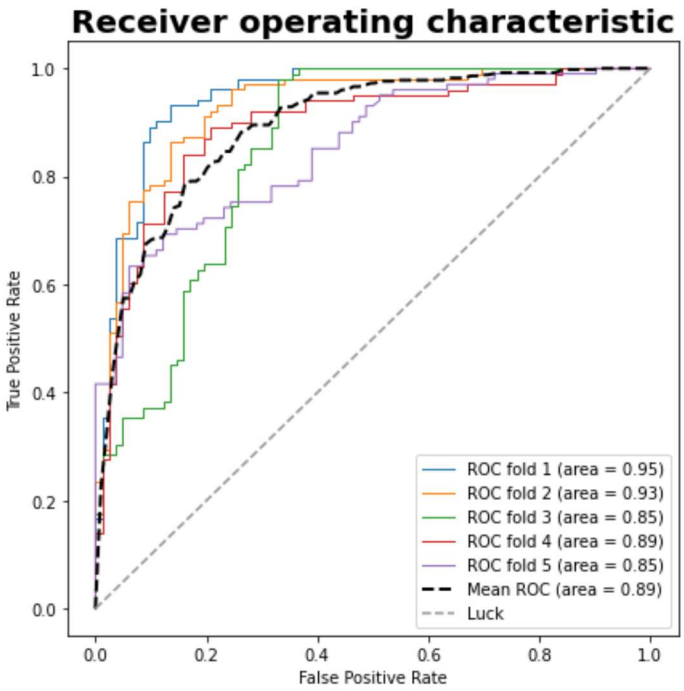


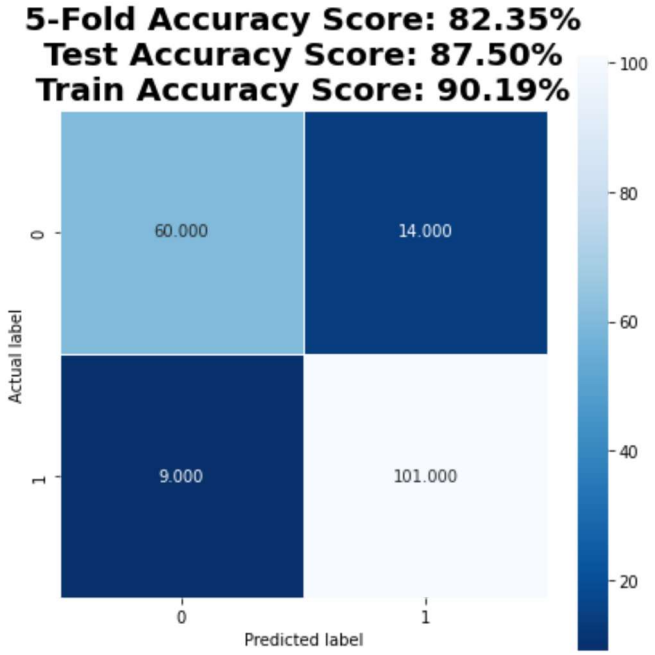
In [44]:

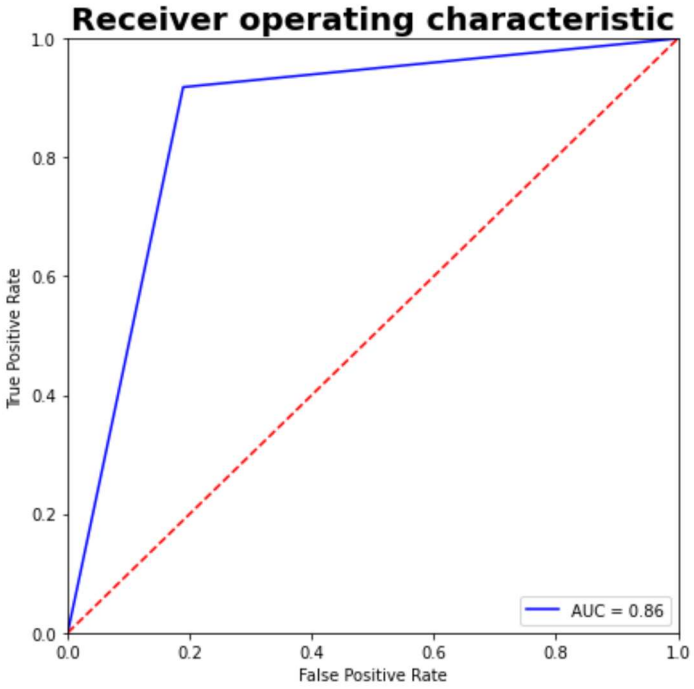
```
lgbm = LGBMClassifier(n_jobs=-1,random_state=32)
build_model(lgbm,X2_train,y2_train,X2_test,y2_test,data2,y,label='Data2',params=lgbm_grid)
```

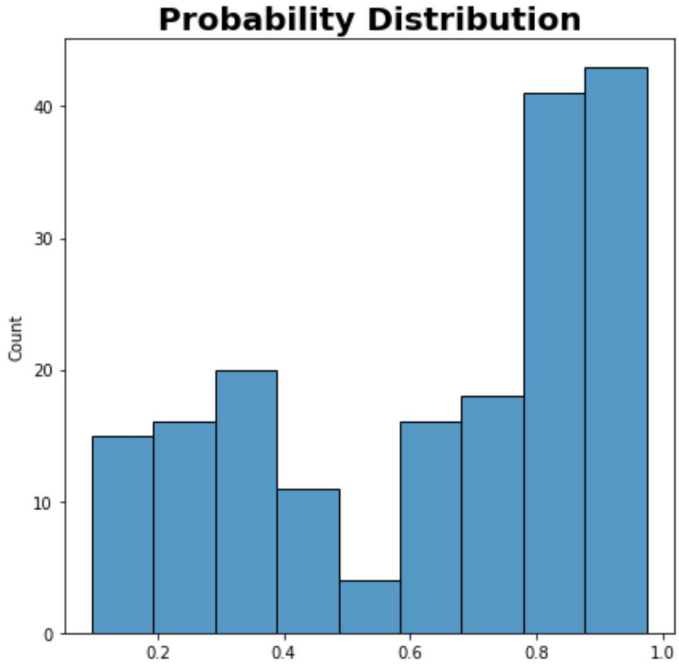
```
[LightGBM] [Warning] min_data_in_leaf is set=50, min_child_samples=20 will be ignored. Current value: min_data_in_leaf=50
[LightGBM] [Warning] lambda_l1 is set=0.001, reg_alpha=100 will be ignored. Current value: lambda_l1=0.001
[LightGBM] [Warning] lambda_l2 is set=0.1, reg_lambda=0.0 will be ignored. Current value: lambda_l2=0.1
[LightGBM] [Warning] min_data_in_leaf is set=100, min_child_samples=20 will be ignored. Current value: min_data_in_leaf=100
[LightGBM] [Warning] lambda_l1 is set=0.001, reg_alpha=1 will be ignored. Current value: lambda_l1=0.001
[LightGBM] [Warning] lambda_l2 is set=10, reg_lambda=0.0 will be ignored. Current value: lambda_l2=10
[LightGBM] [Warning] min_data_in_leaf is set=50, min_child_samples=20 will be ignored. Current value: min_data_in_leaf=50
[LightGBM] [Warning] lambda_l1 is set=0.001, reg_alpha=100 will be ignored. Current value: lambda_l1=0.001
[LightGBM] [Warning] lambda_l2 is set=0.1, reg_lambda=0.0 will be ignored. Current value: lambda_l2=0.1
[LightGBM] [Warning] min_data_in_leaf is set=10, min_child_samples=20 will be ignored. Current value: min_data_in_leaf=10
[LightGBM] [Warning] lambda_l1 is set=0.01, reg_alpha=0.01 will be ignored. Current value: lambda_l1=0.01
[LightGBM] [Warning] lambda_l2 is set=1, reg_lambda=0.0 will be ignored. Current value: lambda_l2=1
[LightGBM] [Warning] min_data_in_leaf is set=50, min_child_samples=20 will be ignored. Current value: min_data_in_leaf=50
[LightGBM] [Warning] lambda_l1 is set=0.001, reg_alpha=100 will be ignored. Current value: lambda_l1=0.001
[LightGBM] [Warning] lambda_l2 is set=0.001, reg_lambda=0.0 will be ignored. Current value: lambda_l2=0.001
[LightGBM] [Warning] min_data_in_leaf is set=50, min_child_samples=20 will be ignored. Current value: min_data_in_leaf=50
[LightGBM] [Warning] lambda_l1 is set=1, reg_alpha=0.1 will be ignored. Current value: lambda_l1=1
[LightGBM] [Warning] lambda_l2 is set=0.01, reg_lambda=0.0 will be ignored. Current value: lambda_l2=0.01
              precision    recall  f1-score   support

           0       0.87      0.81      0.84        74
           1       0.88      0.92      0.90       110

    accuracy                           0.88       184
   macro avg       0.87      0.86      0.87       184
weighted avg       0.87      0.88      0.87       184

Specificity = 0.8108108108108109
The balanced accuracy score is : 86.45%
```

RandomizedSearchCV LGBMClassifier Data2









LightGBM Feature Selection - Sequential Feature Selection

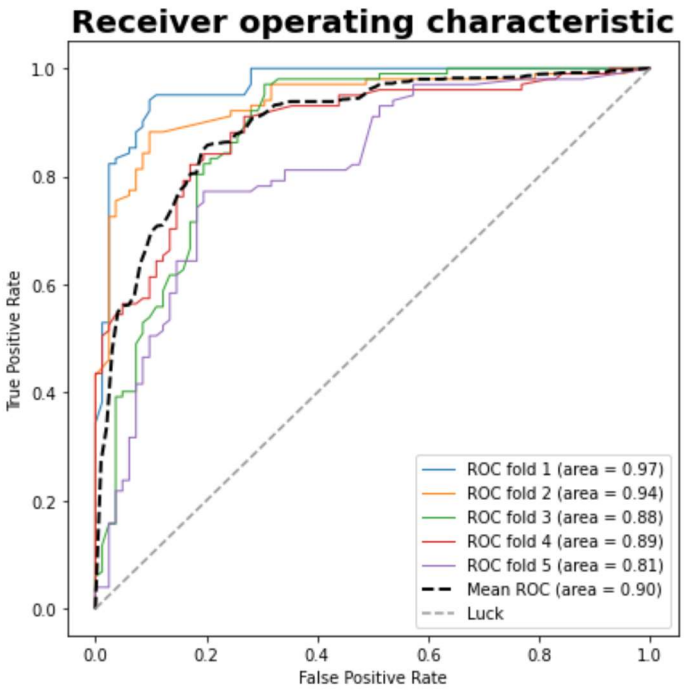
Four of the features were selected that overlapped with the previous iterations of feature selection. This has been confirmed by four other models that these are significant predictors.

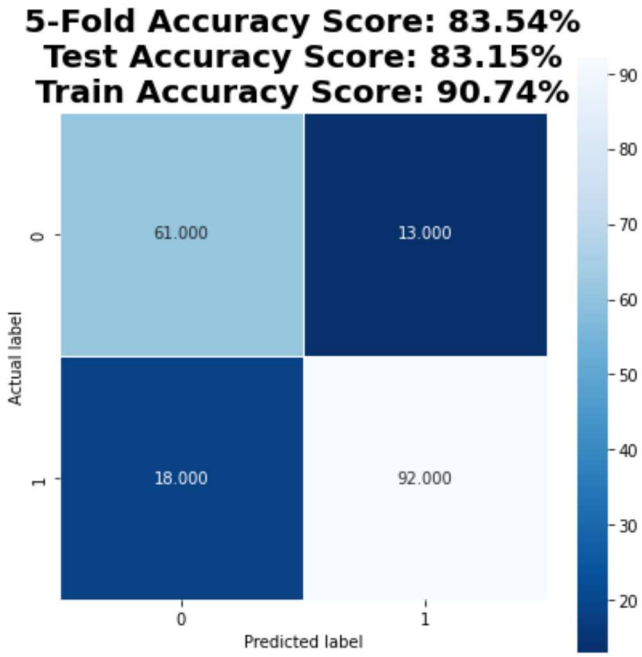
In [45]:
```
lgbm = LGBMClassifier(n_jobs=-1,random_state=32)
sfs = SequentialFeatureSelector(lgbm,n_jobs=-1,cv=cv,scoring='roc_auc')
sfs.fit(data2, y)
print(sfs.get_feature_names_out())
build_model(lgbm,sfs.transform(X2_train),y2_train,sfs.transform(X2_test),y2_test,
            pd.DataFrame(sfs.transform(data2)),y,label='Data2 FS')
```

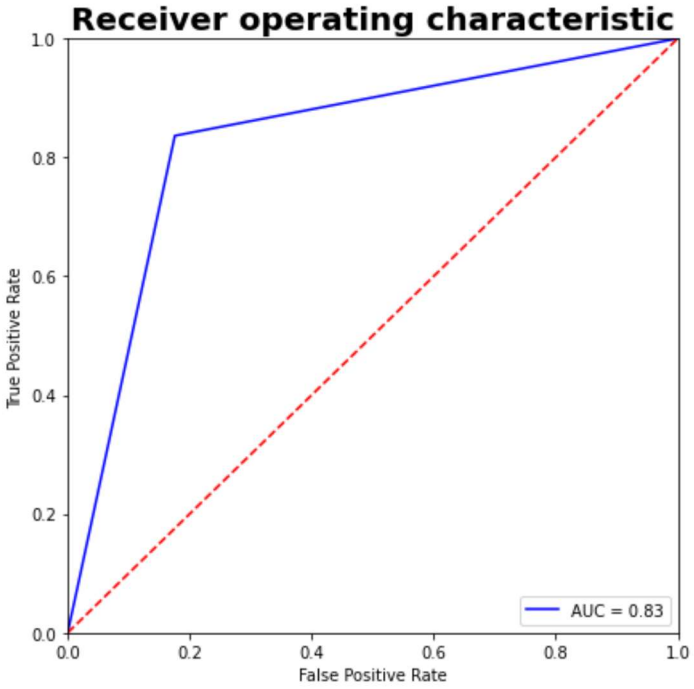
```
['Sex' 'ChestPainType' 'FastingBS' 'Oldpeak' 'ST_Slope']
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\base.py:445: UserWarning: X does not have valid feature names, bu
t SequentialFeatureSelector was fitted with feature names
  warnings.warn(
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\base.py:445: UserWarning: X does not have valid feature names, bu
```

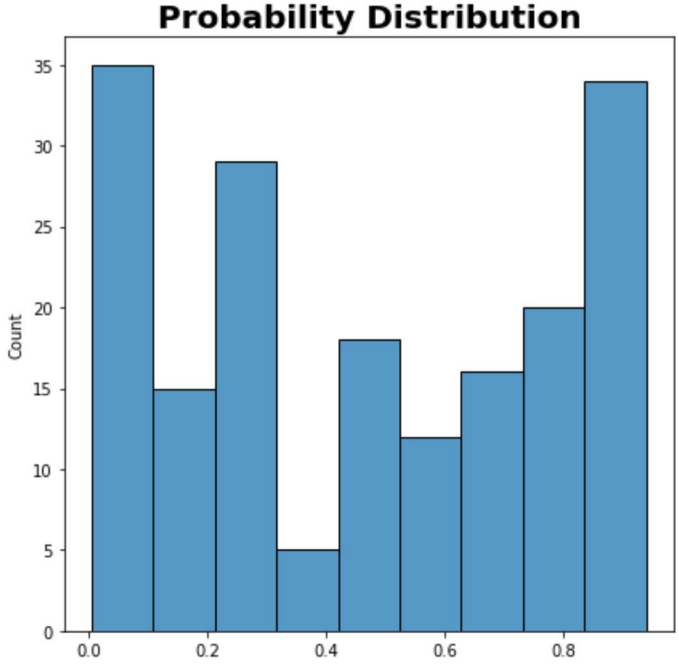
```
t SequentialFeatureSelector was fitted with feature names
  warnings.warn(
              precision    recall  f1-score   support

           0       0.77      0.82      0.80        74
           1       0.88      0.84      0.86       110

    accuracy                           0.83       184
   macro avg       0.82      0.83      0.83       184
weighted avg       0.83      0.83      0.83       184

Specificity = 0.8243243243243243
The balanced accuracy score is : 83.03%
```

LGBMClassifier Data2 FS







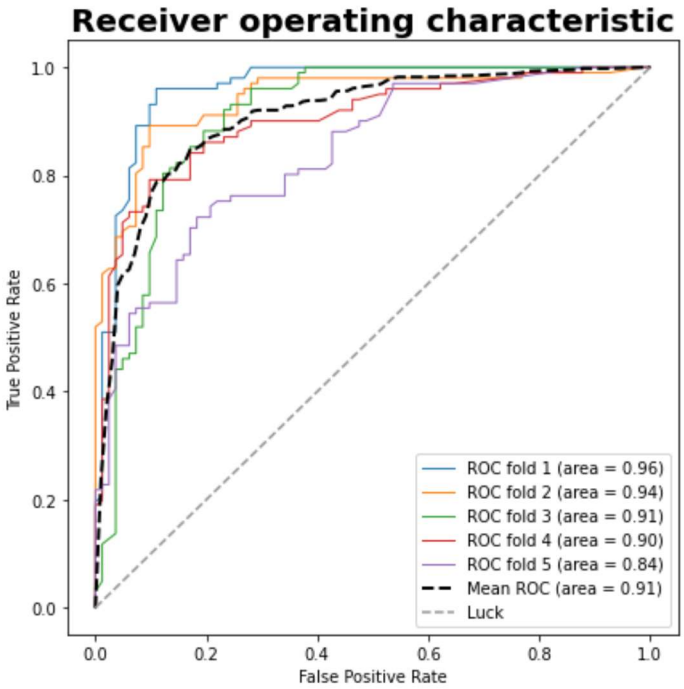


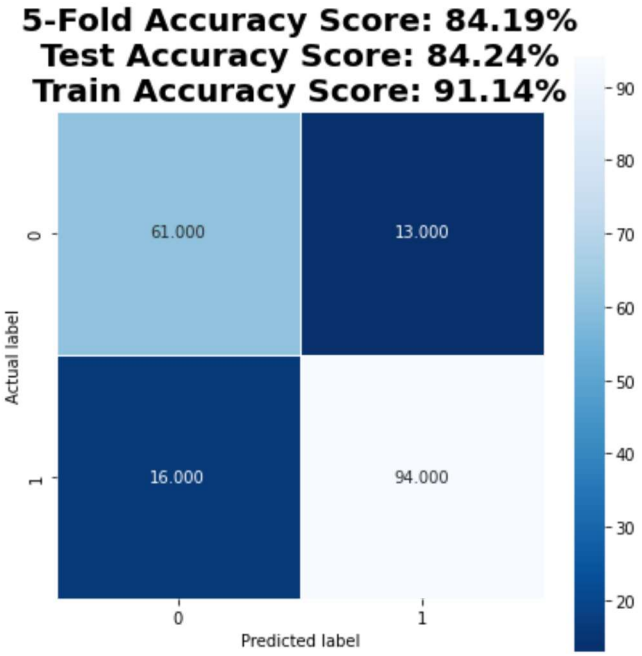
In [46]:

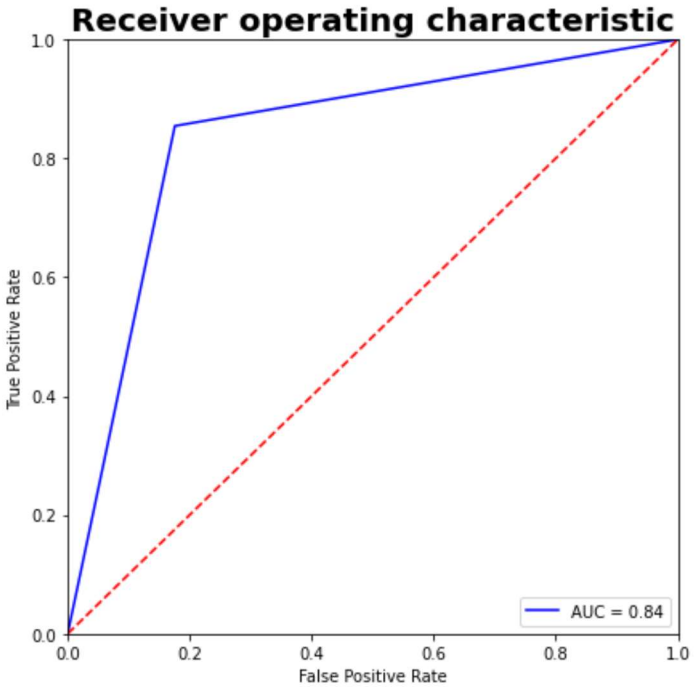
```
lgbm = LGBMClassifier(n_jobs=-1,random_state=32)
sfs = SequentialFeatureSelector(lgbm,n_jobs=-1,cv=cv,scoring='roc_auc')
sfs.fit(data1, y)
print(sfs.get_feature_names_out())
build_model(lgbm,sfs.transform(X1_train),y1_train,sfs.transform(X1_test),y1_test,
            pd.DataFrame(sfs.transform(data1)),y,label='Data1 FS')
```

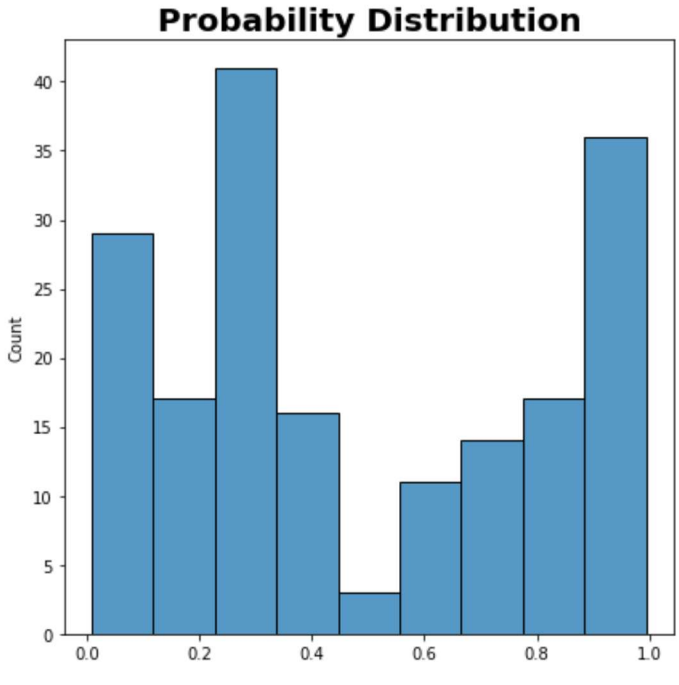
```
['FastingBS' 'Oldpeak' 'Sex_F' 'Sex_M' 'ChestPainType_ASY'
 'ChestPainType_TA' 'RestingECG_ST' 'ExerciseAngina_N' 'ExerciseAngina_Y'
 'ST_Slope_Up']
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\base.py:445: UserWarning: X does not have valid feature names, bu
t SequentialFeatureSelector was fitted with feature names
  warnings.warn(
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\base.py:445: UserWarning: X does not have valid feature names, bu
t SequentialFeatureSelector was fitted with feature names
  warnings.warn(
              precision    recall  f1-score   support

           0       0.79      0.82      0.81        74
           1       0.88      0.85      0.87       110

    accuracy                           0.84       184
   macro avg       0.84      0.84      0.84       184
weighted avg       0.84      0.84      0.84       184

Specificity = 0.8243243243243243
The balanced accuracy score is : 83.94%
```

LGBMClassifier Data1 FS









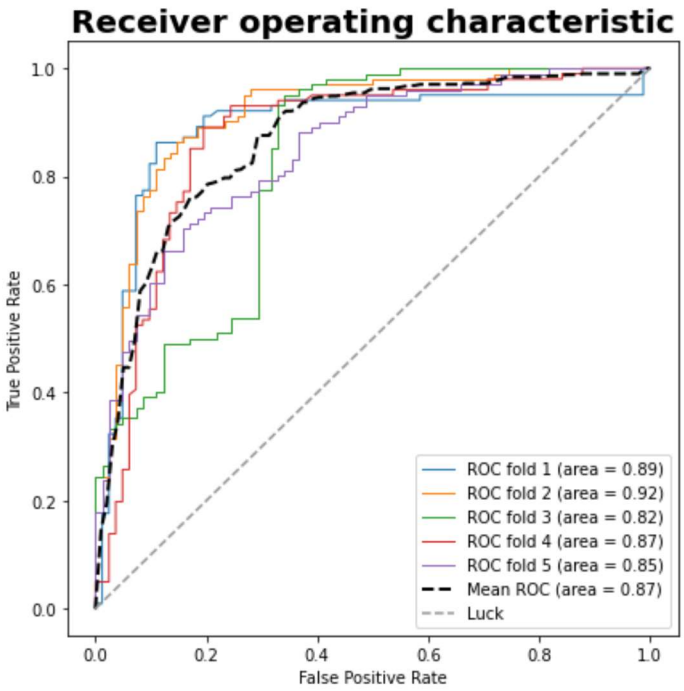
AdaBoost

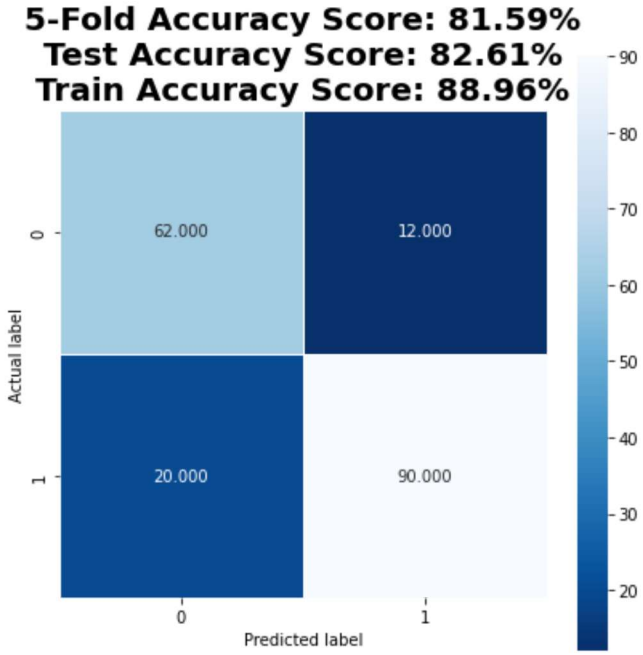
AdaBoost classifier performed marginally worse than the other models. It achieved a much lower accuracy and AUC score. The probability distribution appears to be balanced but many of the predictions are along the threshold. It apperars to have high variance between models as shown in the 5-fold CV ROC plot.

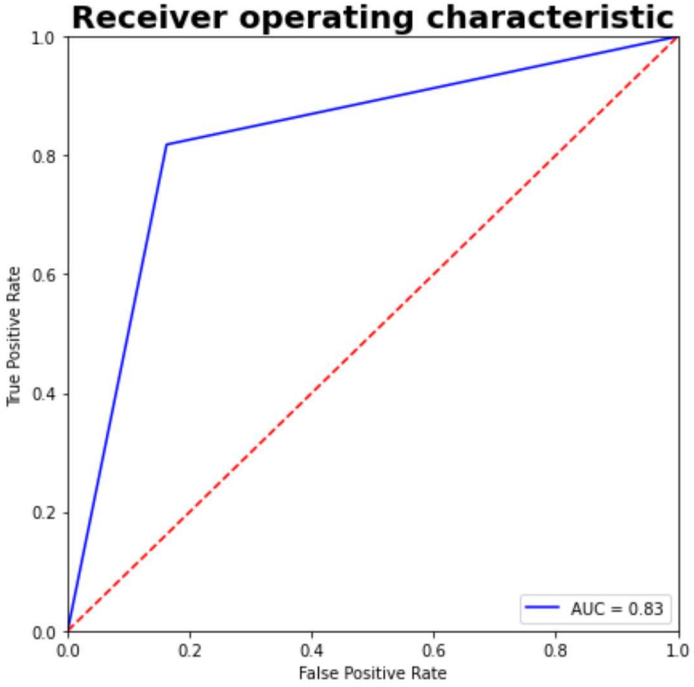
In [47]:
```
from sklearn.ensemble import AdaBoostClassifier

abc = AdaBoostClassifier(random_state=32)
build_model(abc,X1_train,y1_train,X1_test,y1_test,data1,y,label='Data1 Default')
```

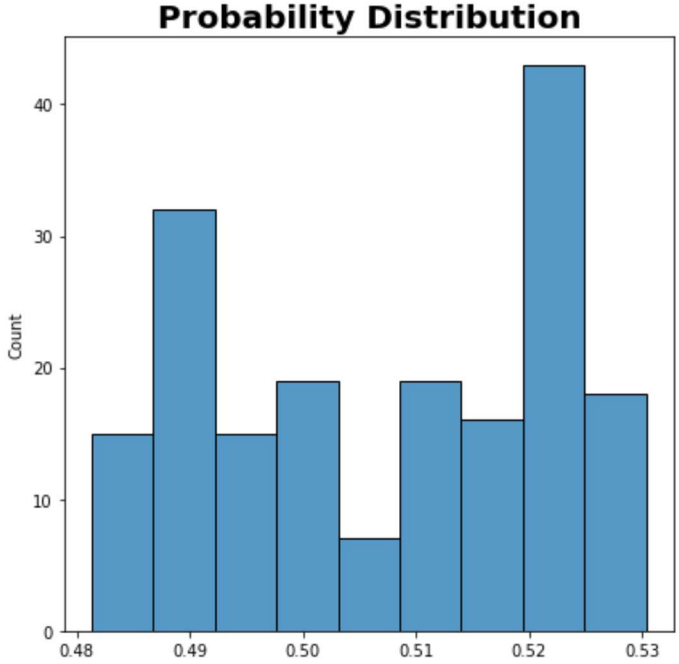
```
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\base.py:445: UserWarning: X does not have valid feature names, bu
t AdaBoostClassifier was fitted with feature names
  warnings.warn(
              precision    recall  f1-score   support

           0       0.76      0.84      0.79        74
           1       0.88      0.82      0.85       110

    accuracy                           0.83       184
   macro avg       0.82      0.83      0.82       184
weighted avg       0.83      0.83      0.83       184

Specificity = 0.8378378378378378
The balanced accuracy score is : 82.80%
```

AdaBoostClassifier Data1 Default



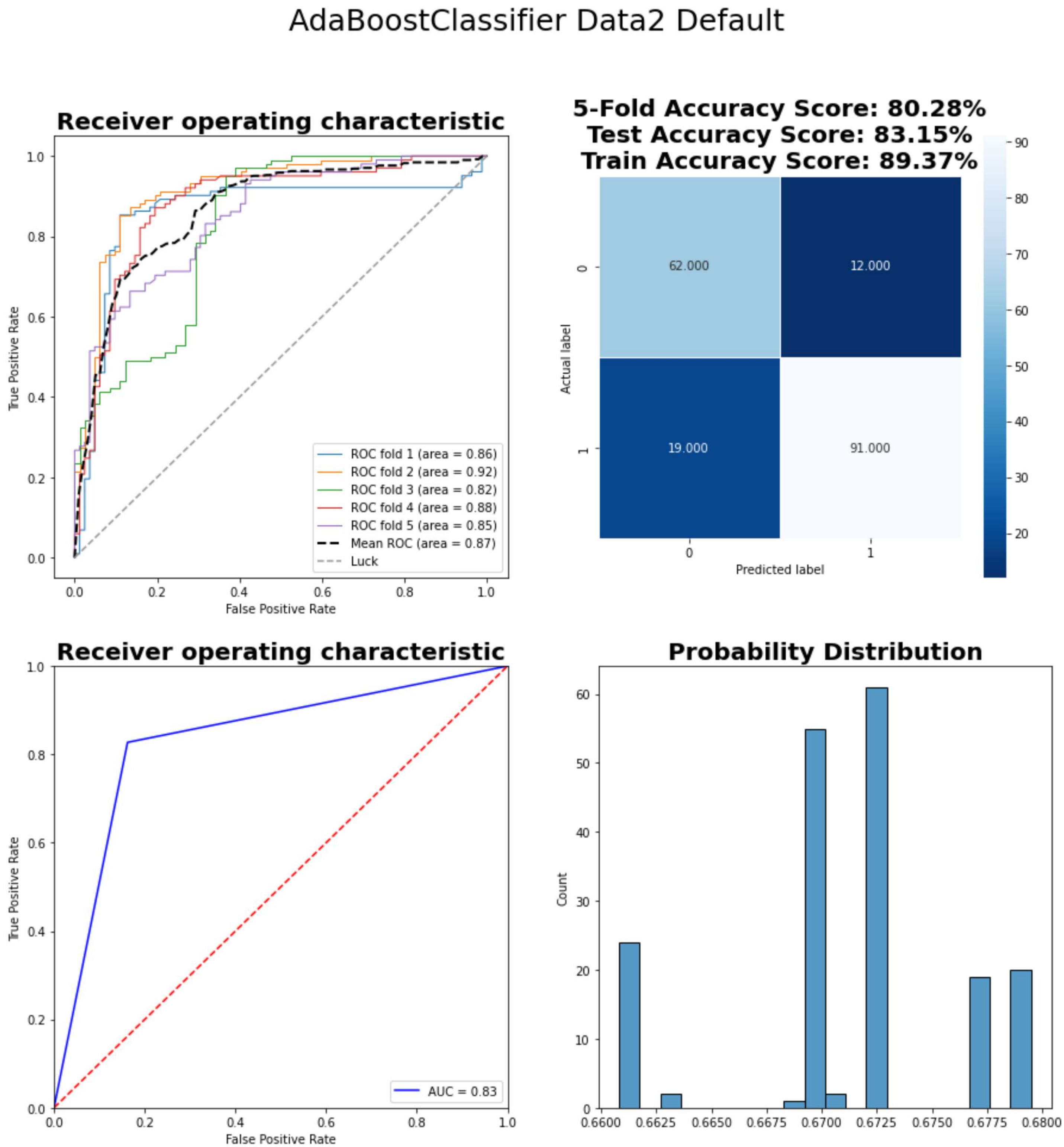






In [48]:
```
abc = AdaBoostClassifier(random_state=32)
build_model(abc,X2_train,y2_train,X2_test,y2_test,data2,y,label='Data2 Default')
```

```
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\base.py:445: UserWarning: X does not have valid feature names, bu
t AdaBoostClassifier was fitted with feature names
  warnings.warn(
              precision    recall  f1-score   support

           0       0.77      0.84      0.80        74
           1       0.88      0.83      0.85       110

    accuracy                           0.83       184
   macro avg       0.82      0.83      0.83       184
weighted avg       0.84      0.83      0.83       184

Specificity = 0.8378378378378378
The balanced accuracy score is : 83.26%
```



In [49]:

```
abc_params = {
    'learning_rate': [0.0001, 0.001, 0.01, 0.1, 1.0],
    'n_estimators':[10, 50, 100, 500],
}
abc = AdaBoostClassifier(random_state=32)
build_model(abc,X1_train,y1_train,X1_test,y1_test,data1,y,label='Data1',params=abc_params)
```

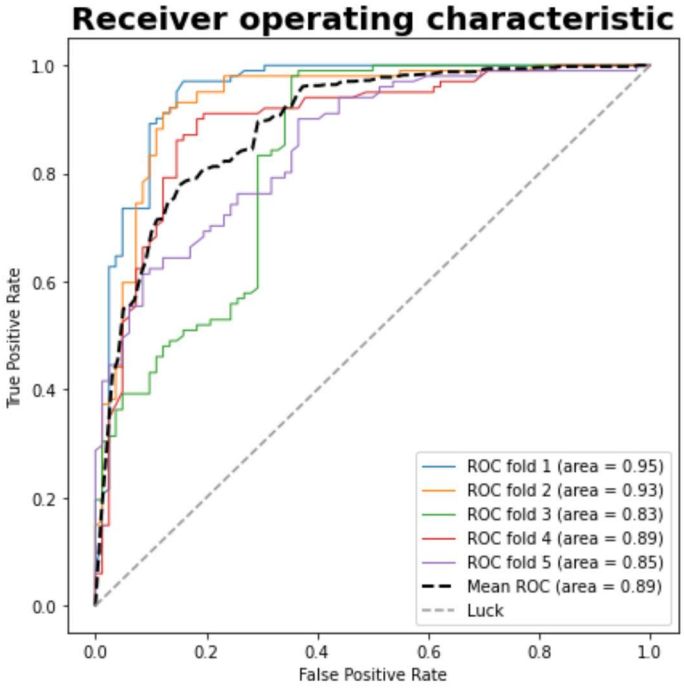
```
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_search.py:292: UserWarning: The total space of p
arameters 20 is smaller than n_iter=100. Running 20 iterations. For exhaustive searches, use GridSearchCV.
  warnings.warn(
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_search.py:292: UserWarning: The total space of p
arameters 20 is smaller than n_iter=100. Running 20 iterations. For exhaustive searches, use GridSearchCV.
  warnings.warn(
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_search.py:292: UserWarning: The total space of p
arameters 20 is smaller than n_iter=100. Running 20 iterations. For exhaustive searches, use GridSearchCV.
  warnings.warn(
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_search.py:292: UserWarning: The total space of p
arameters 20 is smaller than n_iter=100. Running 20 iterations. For exhaustive searches, use GridSearchCV.
  warnings.warn(
```

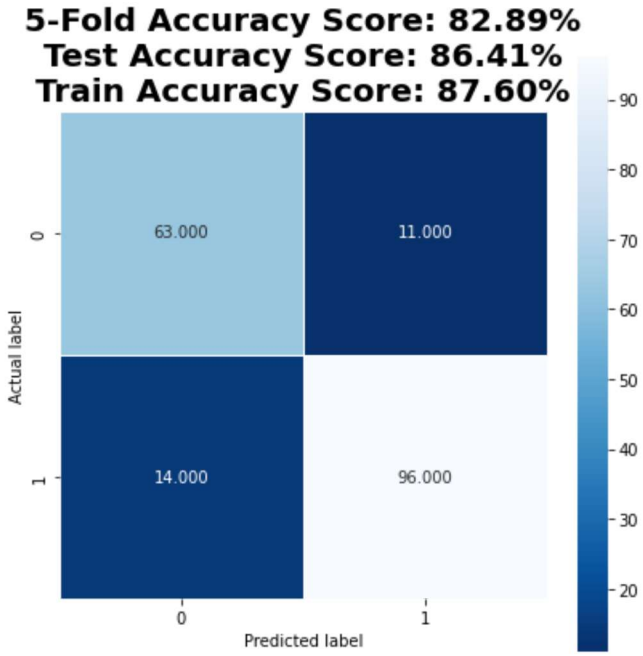
```
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_search.py:292: UserWarning: The total space of parameters 20 is smaller than n_iter=100. Running 20 iterations. For exhaustive searches, use GridSearchCV.
  warnings.warn(
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_search.py:292: UserWarning: The total space of parameters 20 is smaller than n_iter=100. Running 20 iterations. For exhaustive searches, use GridSearchCV.
  warnings.warn(
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\base.py:445: UserWarning: X does not have valid feature names, but AdaBoostClassifier was fitted with feature names
  warnings.warn(
```

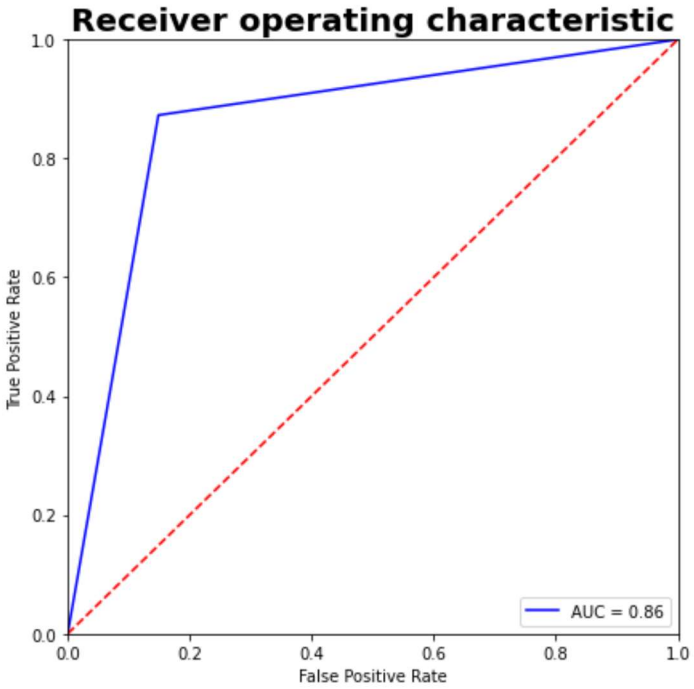
	precision	recall	f1-score	support
0	0.82	0.85	0.83	74
1	0.90	0.87	0.88	110
accuracy			0.86	184
macro avg	0.86	0.86	0.86	184
weighted avg	0.87	0.86	0.86	184

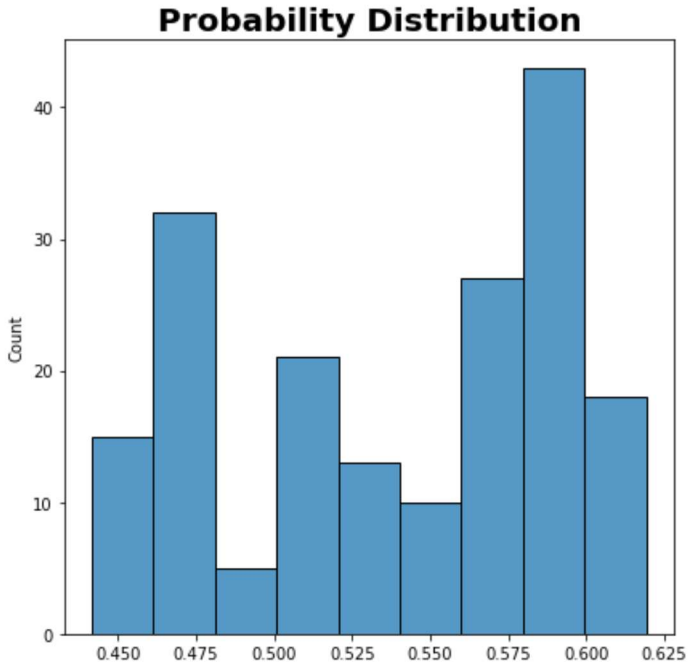
```
Specificity = 0.8513513513513513
The balanced accuracy score is : 86.20%
```

RandomizedSearchCV AdaBoostClassifier Data1





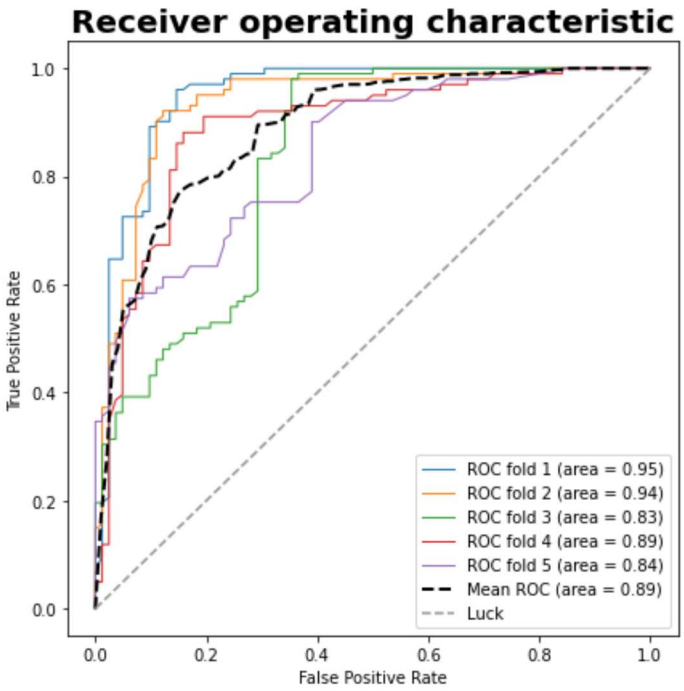


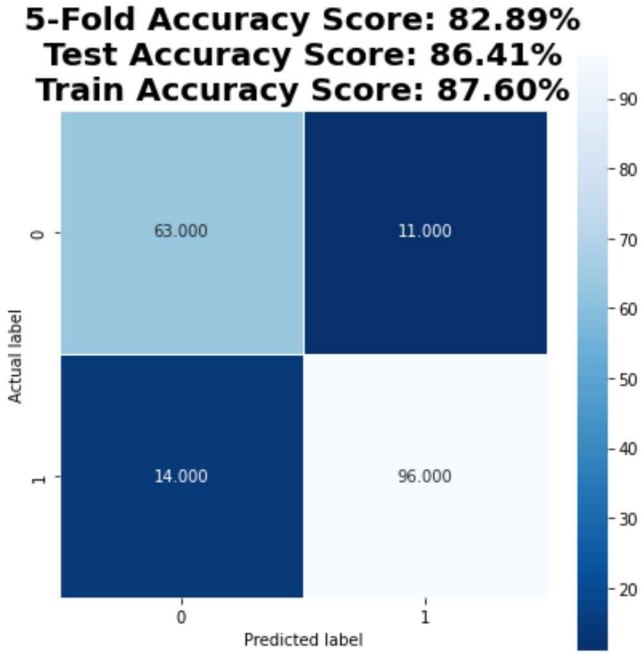


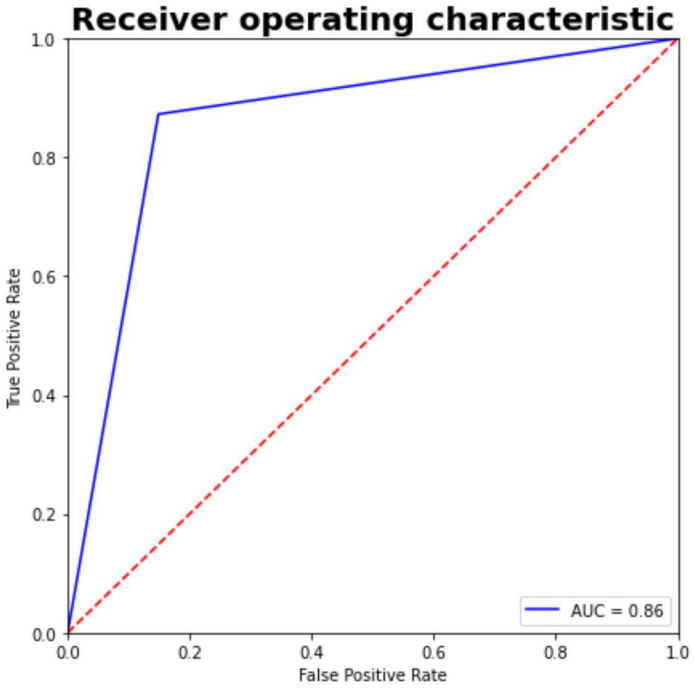
In [50]:
```python
abc = AdaBoostClassifier(random_state=32)
build_model(abc,X2_train,y2_train,X2_test,y2_test,data2,y,label='Data2',params=abc_params)
```

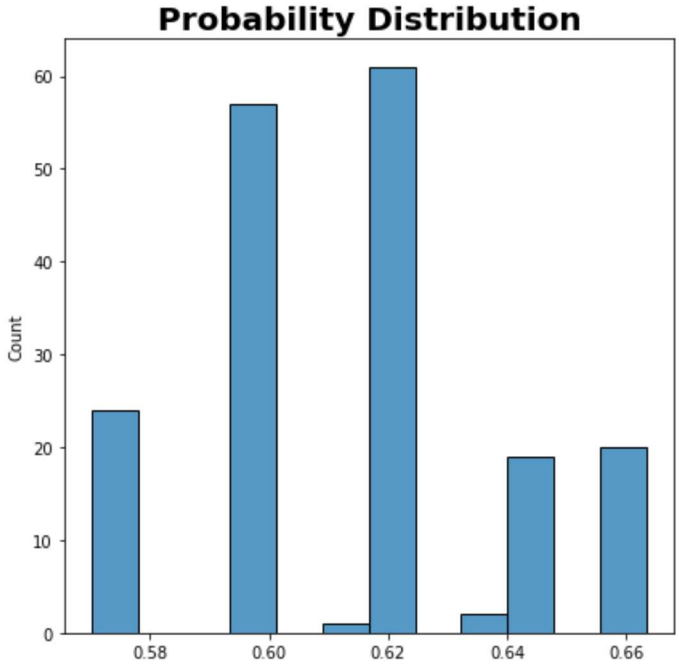
```
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_search.py:292: UserWarning: The total space of p
arameters 20 is smaller than n_iter=100. Running 20 iterations. For exhaustive searches, use GridSearchCV.
  warnings.warn(
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_search.py:292: UserWarning: The total space of p
arameters 20 is smaller than n_iter=100. Running 20 iterations. For exhaustive searches, use GridSearchCV.
  warnings.warn(
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_search.py:292: UserWarning: The total space of p
arameters 20 is smaller than n_iter=100. Running 20 iterations. For exhaustive searches, use GridSearchCV.
  warnings.warn(
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_search.py:292: UserWarning: The total space of p
arameters 20 is smaller than n_iter=100. Running 20 iterations. For exhaustive searches, use GridSearchCV.
  warnings.warn(
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_search.py:292: UserWarning: The total space of p
arameters 20 is smaller than n_iter=100. Running 20 iterations. For exhaustive searches, use GridSearchCV.
  warnings.warn(
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_search.py:292: UserWarning: The total space of p
arameters 20 is smaller than n_iter=100. Running 20 iterations. For exhaustive searches, use GridSearchCV.
  warnings.warn(
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\base.py:445: UserWarning: X does not have valid feature names, bu
t AdaBoostClassifier was fitted with feature names
  warnings.warn(
              precision    recall  f1-score   support

           0       0.82      0.85      0.83        74
           1       0.90      0.87      0.88       110

    accuracy                           0.86       184
   macro avg       0.86      0.86      0.86       184
weighted avg       0.87      0.86      0.86       184

Specificity = 0.8513513513513513
The balanced accuracy score is : 86.20%
```

RandomizedSearchCV AdaBoostClassifier Data2









AdaBoost Feature Selection - Sequential Feature Selection

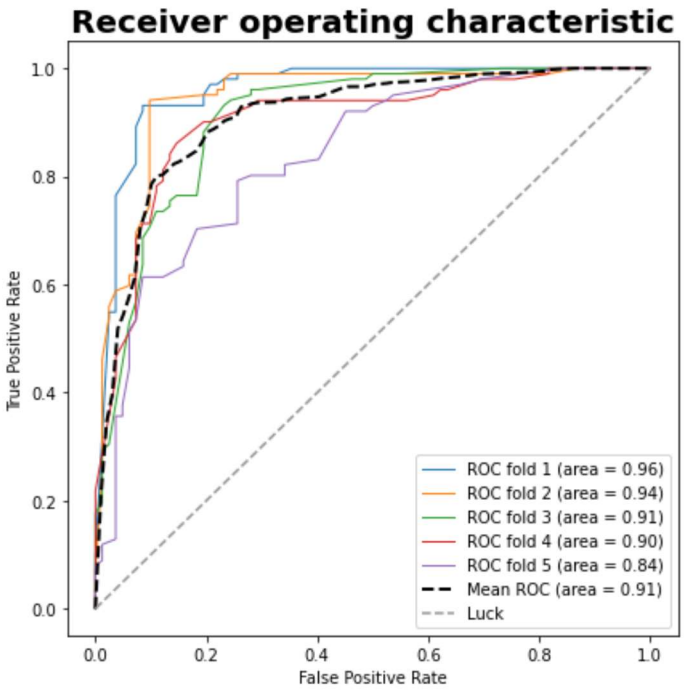
The same 3-5 features are selected once again. However, the predictions achieved are much more varied.

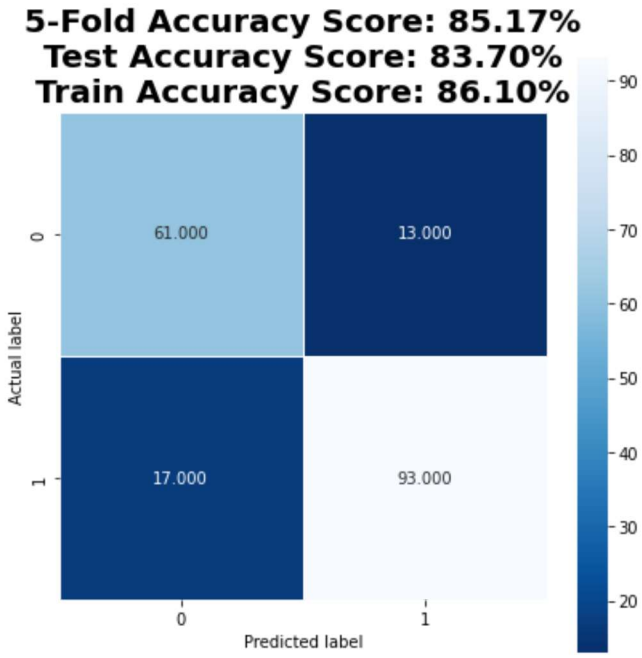
In [51]:
```
abc = AdaBoostClassifier(random_state=32)
sfs = SequentialFeatureSelector(abc,n_jobs=-1,cv=cv,scoring='roc_auc')
sfs.fit(data2, y)
print(sfs.get_feature_names_out())
build_model(abc,sfs.transform(X2_train),y2_train,sfs.transform(X2_test),y2_test,
            pd.DataFrame(sfs.transform(data2)),y,label='Data2 FS')
```

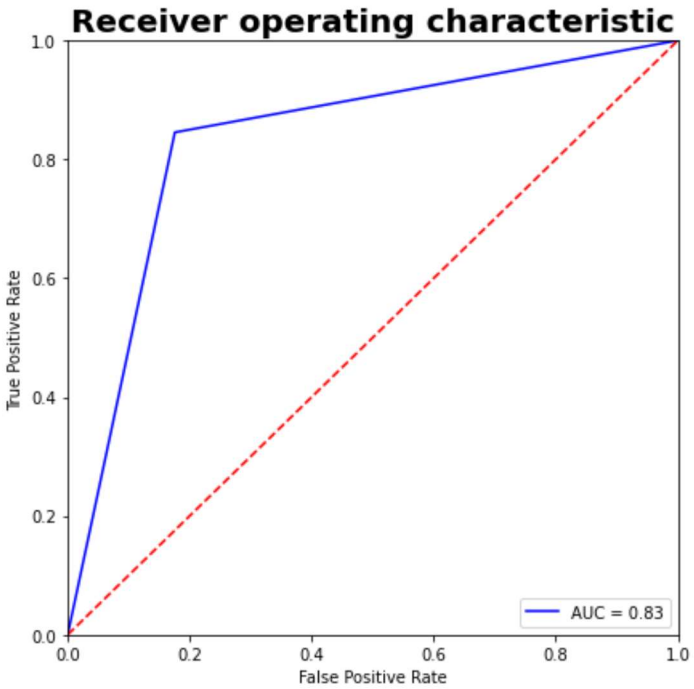
```
['Sex' 'ChestPainType' 'FastingBS' 'ExerciseAngina' 'ST_Slope']
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\base.py:445: UserWarning: X does not have valid feature names, bu
t SequentialFeatureSelector was fitted with feature names
  warnings.warn(
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\base.py:445: UserWarning: X does not have valid feature names, bu
t SequentialFeatureSelector was fitted with feature names
  warnings.warn(
```

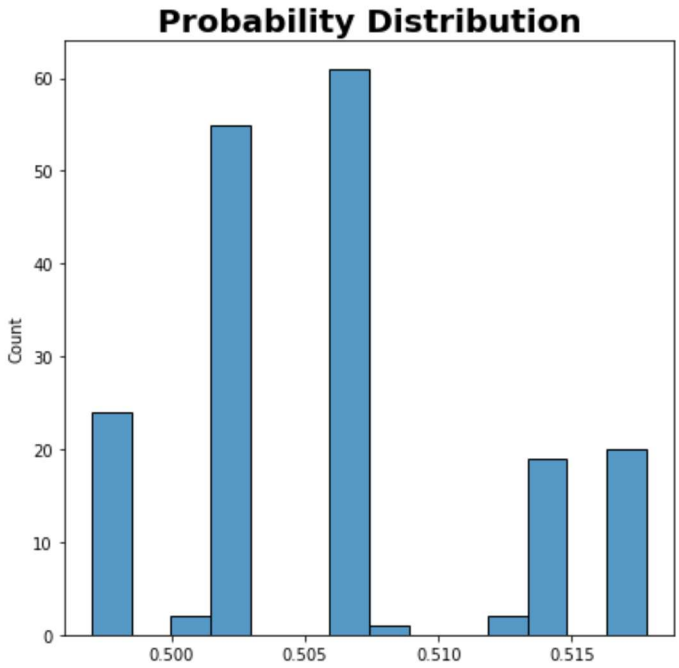
```
              precision    recall  f1-score   support

           0       0.78      0.82      0.80        74
           1       0.88      0.85      0.86       110

    accuracy                           0.84       184
   macro avg       0.83      0.83      0.83       184
weighted avg       0.84      0.84      0.84       184

Specificity = 0.8243243243243243
The balanced accuracy score is : 83.49%
```

AdaBoostClassifier Data2 FS









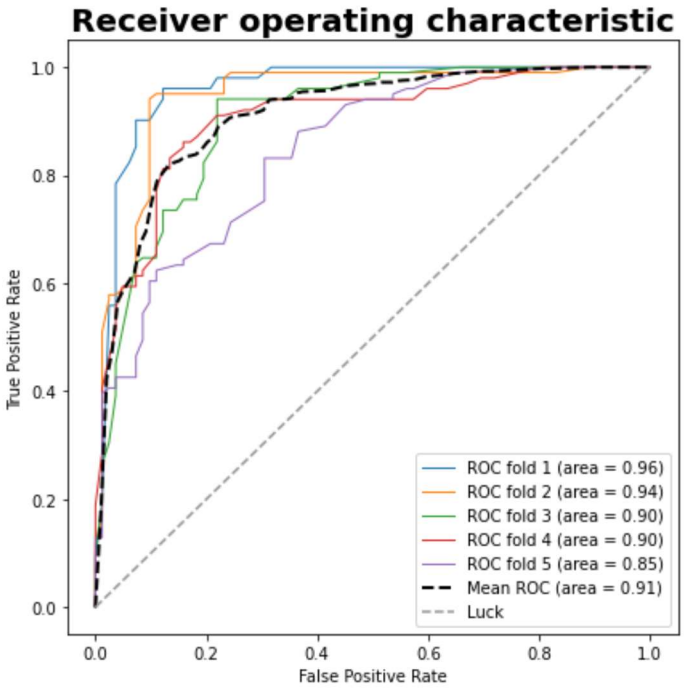
In [52]:

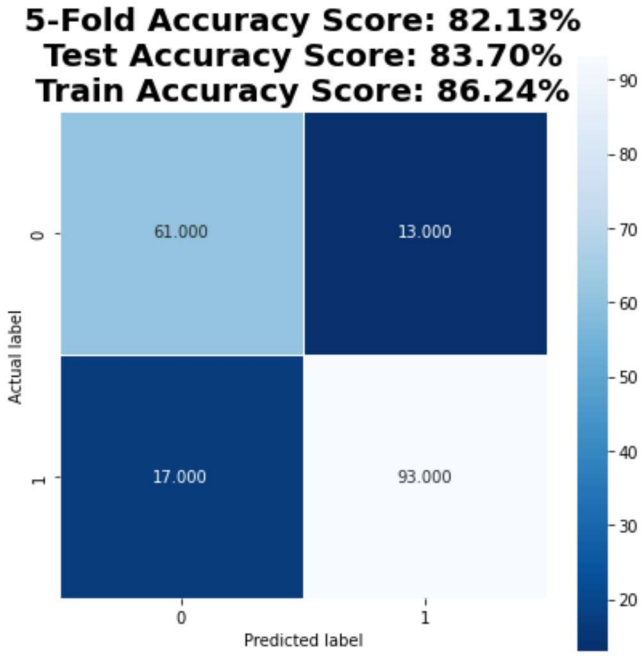
```
abc = AdaBoostClassifier(random_state=32)
sfs = SequentialFeatureSelector(abc,n_jobs=-1,cv=cv,scoring='roc_auc')
sfs.fit(data1, y)
print(sfs.get_feature_names_out())
build_model(abc,sfs.transform(X1_train),y1_train,sfs.transform(X1_test),y1_test,
            pd.DataFrame(sfs.transform(data1)),y,label='Data1 FS')
```

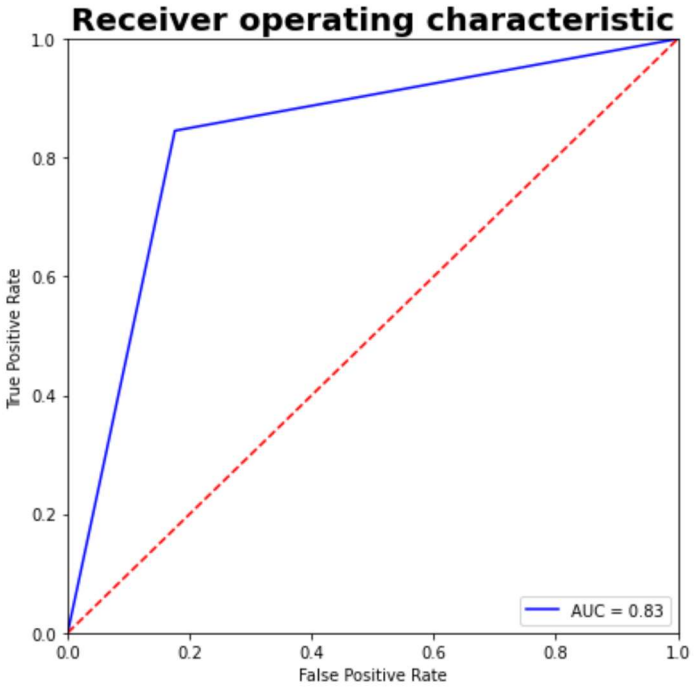
```
['FastingBS' 'Sex_F' 'Sex_M' 'ChestPainType_ASY' 'ChestPainType_TA'
```

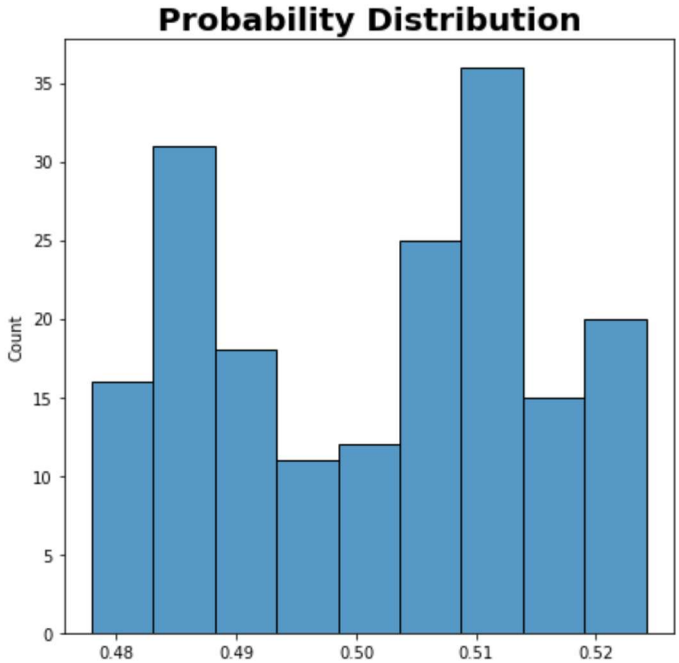
```
 'RestingECG_LVH' 'RestingECG_Normal' 'ExerciseAngina_N'
 'ExerciseAngina_Y' 'ST_Slope_Up']
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\base.py:445: UserWarning: X does not have valid feature names, bu
t SequentialFeatureSelector was fitted with feature names
  warnings.warn(
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\base.py:445: UserWarning: X does not have valid feature names, bu
t SequentialFeatureSelector was fitted with feature names
  warnings.warn(
              precision    recall  f1-score   support

           0       0.78      0.82      0.80        74
           1       0.88      0.85      0.86       110

    accuracy                           0.84       184
   macro avg       0.83      0.83      0.83       184
weighted avg       0.84      0.84      0.84       184

Specificity = 0.8243243243243243
The balanced accuracy score is : 83.49%
```

AdaBoostClassifier Data1 FS









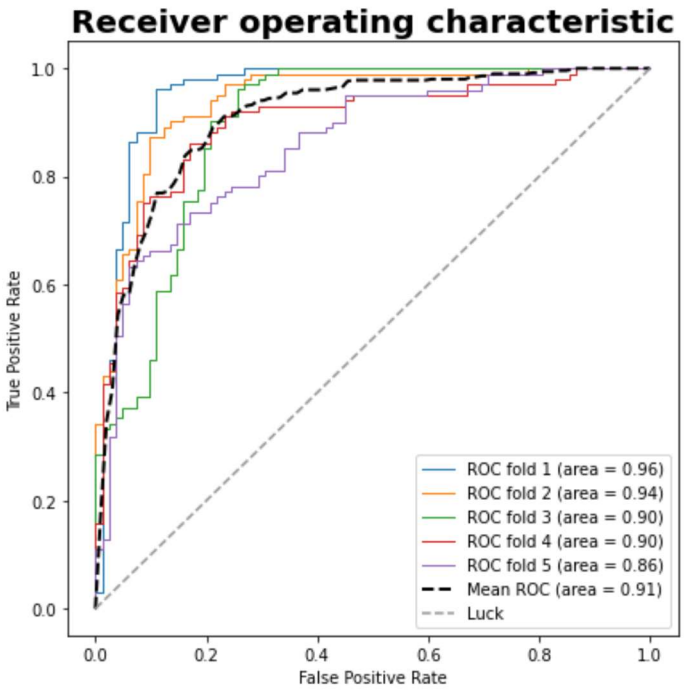
CatBoost

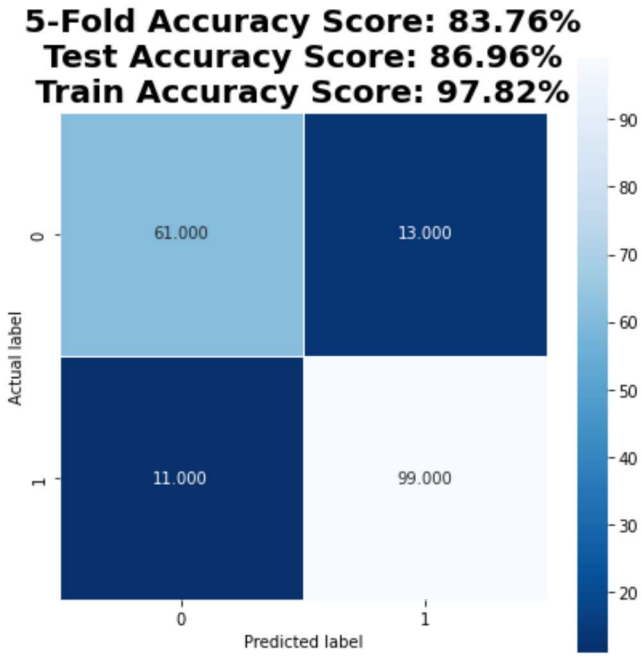
Catboost has been one of the marginally better models compared to the rest of the other ones. The probability distribution is still skewed and tend to favor heart disease class.

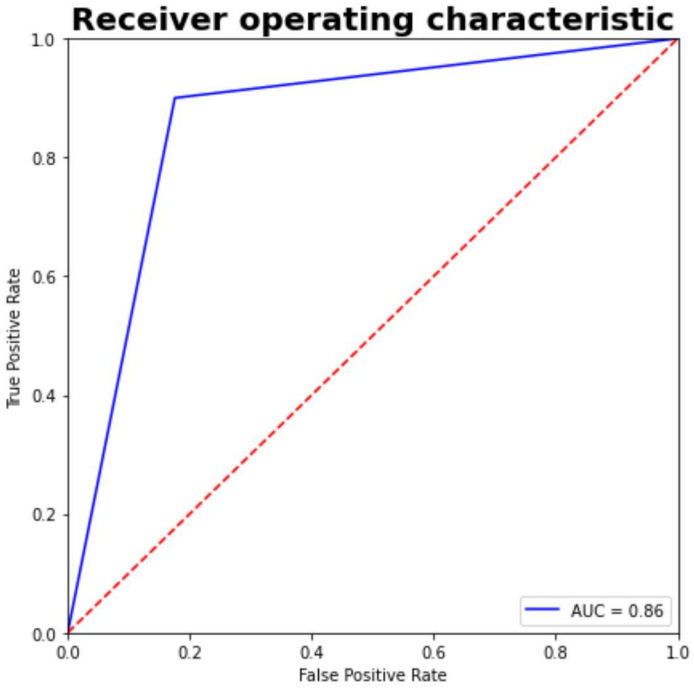
In [53]:
```
from catboost import CatBoostClassifier

cbc = CatBoostClassifier(random_state=32,verbose=False)
build_model(cbc,X1_train,y1_train,X1_test,y1_test,data1,y,label='Data1 Default')
```

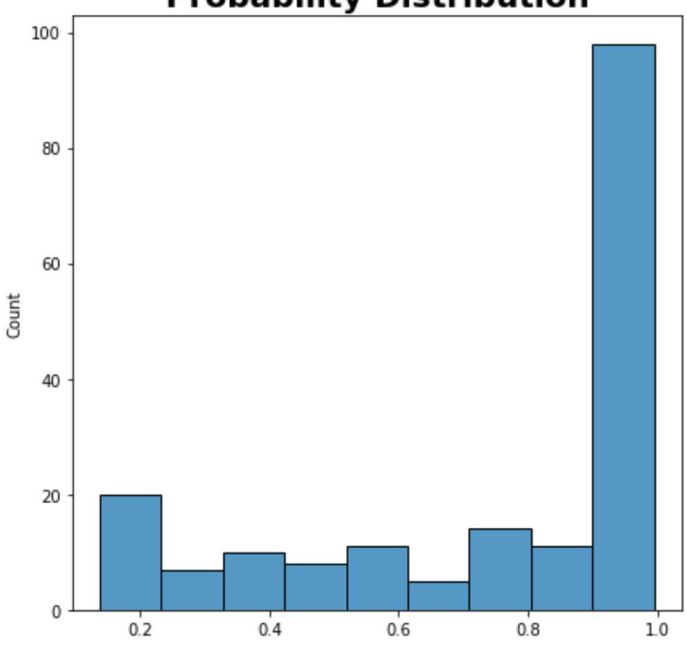
```
              precision    recall  f1-score   support

           0       0.85      0.82      0.84        74
           1       0.88      0.90      0.89       110

    accuracy                           0.87       184
   macro avg       0.87      0.86      0.86       184
weighted avg       0.87      0.87      0.87       184

Specificity = 0.8243243243243243
The balanced accuracy score is : 86.22%
```

CatBoostClassifier Data1 Default





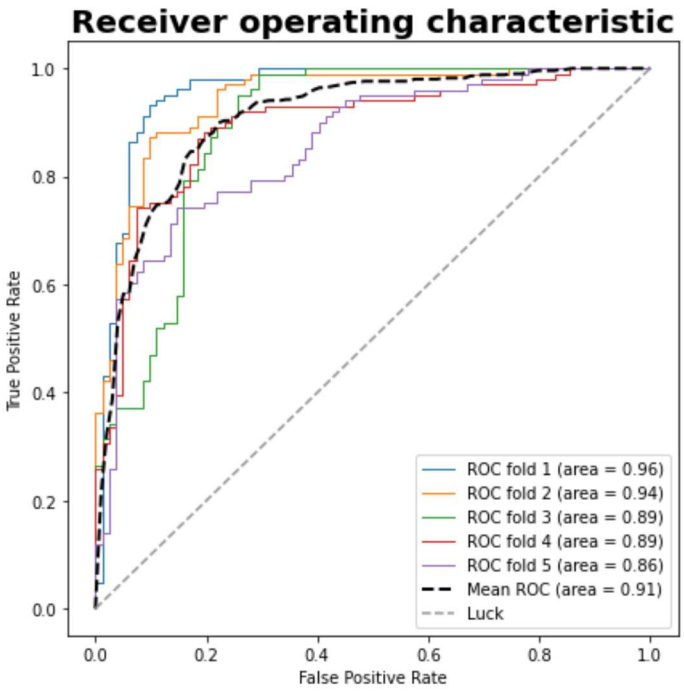


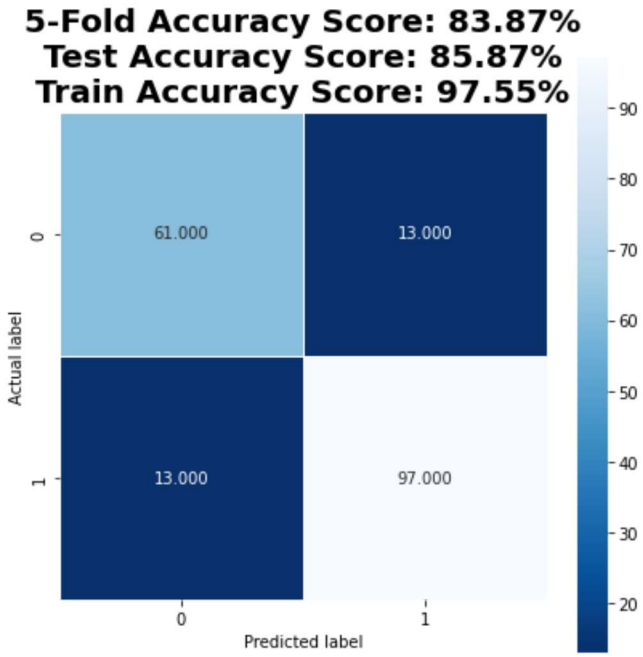


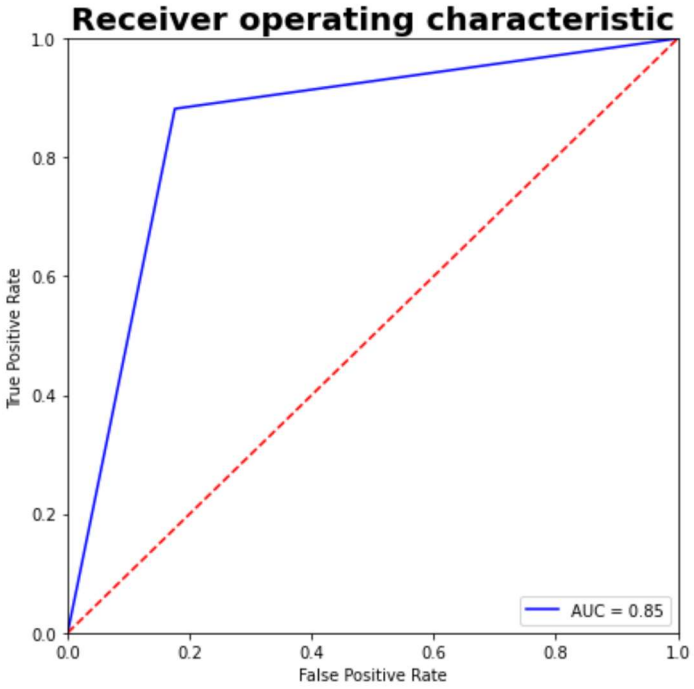
In [54]:
```
cbc = CatBoostClassifier(random_state=32,verbose=False)
build_model(cbc,X2_train,y2_train,X2_test,y2_test,data2,y,label='Data2 Default')
```

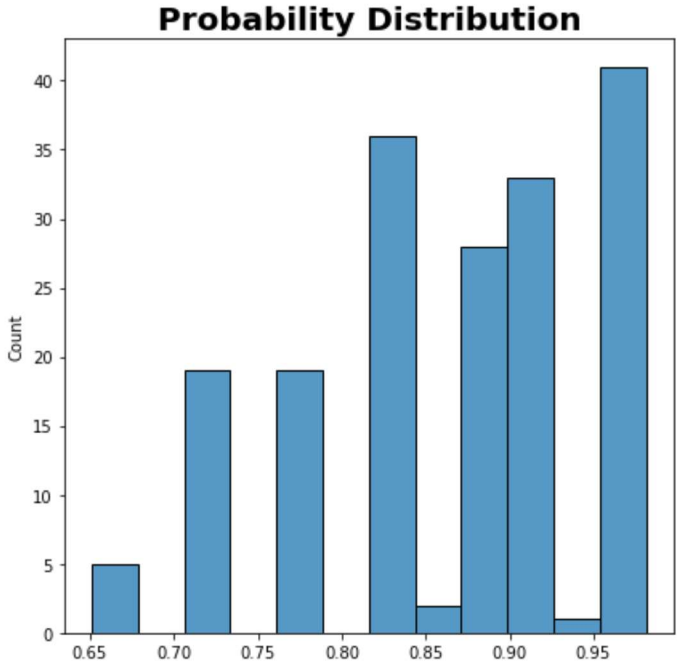
```
              precision    recall  f1-score   support

           0       0.82      0.82      0.82        74
           1       0.88      0.88      0.88       110

    accuracy                           0.86       184
   macro avg       0.85      0.85      0.85       184
weighted avg       0.86      0.86      0.86       184

Specificity = 0.8243243243243243
The balanced accuracy score is : 85.31%
```

CatBoostClassifier Data2 Default









In [55]:

```
cbc_params =    {
    'learning_rate': [0.01,0.02,0.03,0.04],
    'depth': [4, 6,7,8,9,10,15,20],
    'l2_leaf_reg': [0,0.5,1, 3],
    'iterations': [10, 20,30,40,50,60,70,80,90, 100,150]
}
cbc = CatBoostClassifier(random_state=32,verbose=False)
build_model(cbc,X1_train,y1_train,X1_test,y1_test,data1,y,label='Data1 Default',params=cbc_params)
```

```
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_validation.py:372: FitFailedWarning:
80 fits failed out of a total of 500.
The score on these train-test partitions for these parameters will be set to nan.
If these failures are not expected, you can try to debug them by setting error_score='raise'.

Below are more details about the failures:
--------------------------------------------------------------------------------
80 fits failed with the following error:
Traceback (most recent call last):
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_validation.py", line 681, in _fit_and_sc
```

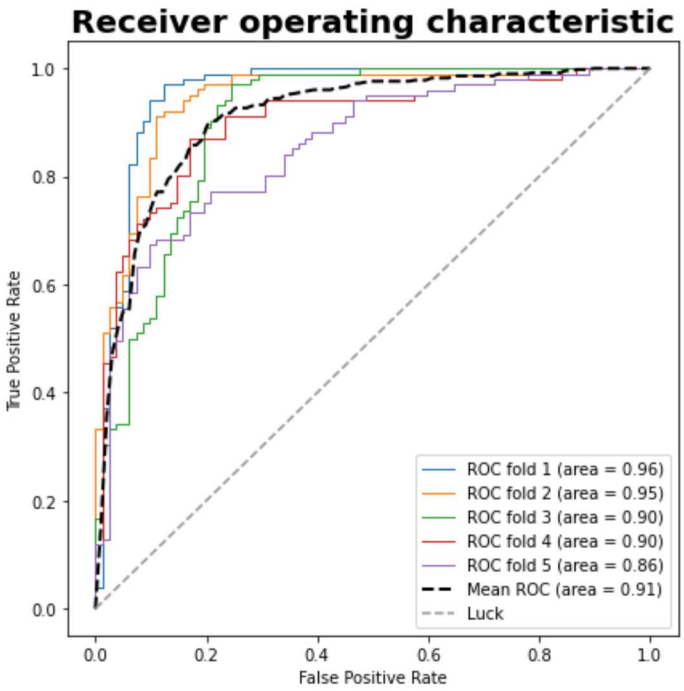
```
ore
    estimator.fit(X_train, y_train, **fit_params)
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\catboost\core.py", line 4716, in fit
    self._fit(X, y, cat_features, text_features, embedding_features, None, sample_weight, None, None, None, None, baseline, use
_best_model,
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\catboost\core.py", line 2021, in _fit
    train_params = self._prepare_train_params(
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\catboost\core.py", line 1953, in _prepare_train_params
    _check_train_params(params)
  File "_catboost.pyx", line 5839, in _catboost._check_train_params
  File "_catboost.pyx", line 5858, in _catboost._check_train_params
_catboost.CatBoostError: C:/Program Files (x86)/Go Agent/pipelines/BuildMaster/catboost.git/catboost/private/libs/options/obliv
ious_tree_options.cpp:122: Maximum tree depth is 16

  warnings.warn(some_fits_failed_message, FitFailedWarning)
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_search.py:969: UserWarning: One or more of the t
est scores are non-finite: [       nan        nan        nan 0.92114908 0.93331689        nan
 0.93580156 0.93243071 0.92219541 0.92444694 0.93403774 0.93503395
 0.91885437 0.93193769 0.93316458 0.93068639        nan 0.93401296
 0.93327608        nan        nan 0.93239751 0.92566375 0.92550597
 0.93220358 0.92922661 0.93059949 0.9300686         nan        nan
 0.90418013 0.92632889 0.93195086 0.93037677 0.93532071 0.92716671
 0.93500767 0.93667021 0.93401018        nan 0.91918427        nan
 0.93625749 0.92554887        nan 0.93408533 0.93481841 0.93266451
 0.92437301 0.92666138        nan 0.92992157 0.93621064 0.93228365
 0.93374557        nan 0.93216674 0.91872473 0.92974553 0.9291179
 0.93235083 0.92873048 0.936429   0.91803439 0.93781798        nan
 0.93394931 0.93621484 0.93508999 0.92916952 0.9349117  0.92877174
 0.92544359        nan 0.92543448 0.93135162 0.9242736  0.93842264
 0.93197356 0.93687648 0.93257188 0.93659342 0.93551283 0.92755293
 0.92694397 0.92574049 0.92060627 0.93378158 0.92887651 0.93571893
 0.93257602 0.93310921 0.93202276 0.92495641 0.93662218 0.93454146
 0.93469566 0.93511116 0.93007921 0.919746  ]
  warnings.warn(
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_validation.py:372: FitFailedWarning:
80 fits failed out of a total of 500.
The score on these train-test partitions for these parameters will be set to nan.
If these failures are not expected, you can try to debug them by setting error_score='raise'.

Below are more details about the failures:
--------------------------------------------------------------------------------
80 fits failed with the following error:
Traceback (most recent call last):
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_validation.py", line 681, in _fit_and_sc
ore
    estimator.fit(X_train, y_train, **fit_params)
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\catboost\core.py", line 4716, in fit
    self._fit(X, y, cat_features, text_features, embedding_features, None, sample_weight, None, None, None, None, baseline, use
_best_model,
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\catboost\core.py", line 2021, in _fit
    train_params = self._prepare_train_params(
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\catboost\core.py", line 1953, in _prepare_train_params
    _check_train_params(params)
  File "_catboost.pyx", line 5839, in _catboost._check_train_params
  File "_catboost.pyx", line 5858, in _catboost._check_train_params
_catboost.CatBoostError: C:/Program Files (x86)/Go Agent/pipelines/BuildMaster/catboost.git/catboost/private/libs/options/obliv
ious_tree_options.cpp:122: Maximum tree depth is 16

  warnings.warn(some_fits_failed_message, FitFailedWarning)
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_search.py:969: UserWarning: One or more of the t
est scores are non-finite: [       nan        nan        nan 0.89004108 0.90879978        nan
 0.9064869  0.90233606 0.89966263 0.89981212 0.90652818 0.9083353
 0.89344522 0.89076137 0.90707137 0.89348308        nan 0.90147921
 0.90615417        nan        nan 0.90236614 0.89846082 0.89793198
 0.90448896 0.89772959 0.90048949 0.8958758         nan        nan
 0.86802487 0.8768471  0.90408952 0.90471491 0.90763342 0.89603081
 0.9056068  0.90318057 0.90361201        nan 0.8613518         nan
 0.90332883 0.89166821        nan 0.90643684 0.90797509 0.90610289
 0.90236538 0.87696814        nan 0.89970407 0.9086737  0.90678664
 0.90728476        nan 0.9044445  0.89085304 0.90228851 0.89511358
 0.90486848 0.89980334 0.90900891 0.88522255 0.90628371        nan
 0.90239583 0.90626549 0.90706217 0.89396993 0.90723645 0.89253528
 0.894869          nan 0.89922244 0.90012213 0.90195356 0.91034881
 0.90332761 0.90785235 0.88776187 0.90606858 0.9010215  0.89146631
 0.89579184 0.89058981 0.88155842 0.90545216 0.90154971 0.90831332
 0.90535833 0.90568801 0.90177931 0.89062266 0.90669957 0.90590525
 0.90472693 0.90147122 0.90432712 0.89385985]
  warnings.warn(
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_validation.py:372: FitFailedWarning:
80 fits failed out of a total of 500.
The score on these train-test partitions for these parameters will be set to nan.
```

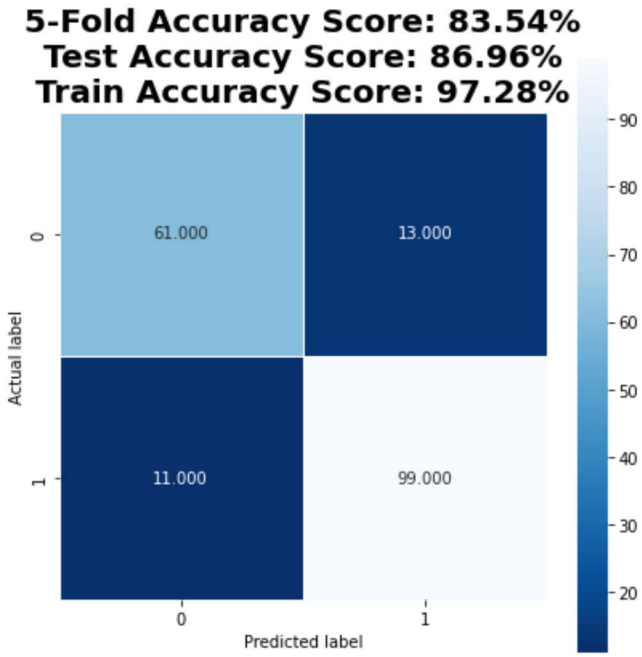
```
If these failures are not expected, you can try to debug them by setting error_score='raise'.

Below are more details about the failures:
--------------------------------------------------------------------------------
80 fits failed with the following error:
Traceback (most recent call last):
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_validation.py", line 681, in _fit_and_sc
ore
    estimator.fit(X_train, y_train, **fit_params)
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\catboost\core.py", line 4716, in fit
    self._fit(X, y, cat_features, text_features, embedding_features, None, sample_weight, None, None, None, None, baseline, use
_best_model,
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\catboost\core.py", line 2021, in _fit
    train_params = self._prepare_train_params(
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\catboost\core.py", line 1953, in _prepare_train_params
    _check_train_params(params)
  File "_catboost.pyx", line 5839, in _catboost._check_train_params
  File "_catboost.pyx", line 5858, in _catboost._check_train_params
_catboost.CatBoostError: C:/Program Files (x86)/Go Agent/pipelines/BuildMaster/catboost.git/catboost/private/libs/options/obliv
ious_tree_options.cpp:122: Maximum tree depth is 16

  warnings.warn(some_fits_failed_message, FitFailedWarning)
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_search.py:969: UserWarning: One or more of the t
est scores are non-finite: [       nan        nan        nan 0.8925835  0.90852678        nan
 0.90861626 0.90080968 0.90165666 0.89972641 0.90709143 0.90980305
 0.90232925 0.88684564 0.90516202 0.89915063        nan 0.90219426
 0.90525766        nan        nan 0.90178069 0.89707163 0.89933892
 0.90797159 0.90486812 0.90301831 0.89138131        nan        nan
 0.86096142 0.88927795 0.90297803 0.90387652 0.90749566 0.90474875
 0.90610061 0.90194952 0.9050457         nan 0.87862307        nan
 0.90466226 0.88160266        nan 0.9086743  0.9073034  0.90768599
 0.90260251 0.88950166        nan 0.90462292 0.90856526 0.90472367
 0.9039623         nan 0.90190183 0.89557473 0.90452216 0.90009238
 0.90839787 0.9005996  0.90891317 0.89391536 0.90396088        nan
 0.90350362 0.90430515 0.90476304 0.90109287 0.90411017 0.88479127
 0.89876769        nan 0.90342109 0.89960289 0.90357267 0.9065758
 0.90052567 0.9084494  0.88998831 0.90759789 0.90378942 0.88517356
 0.89761188 0.89179861 0.87911607 0.90300408 0.9003917  0.90783485
 0.90745034 0.90772972 0.90103551 0.88898423 0.90571224 0.9070353
 0.90694188 0.90459148 0.90327804 0.89523911]
  warnings.warn(
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_validation.py:372: FitFailedWarning:
80 fits failed out of a total of 500.
The score on these train-test partitions for these parameters will be set to nan.
If these failures are not expected, you can try to debug them by setting error_score='raise'.

Below are more details about the failures:
--------------------------------------------------------------------------------
80 fits failed with the following error:
Traceback (most recent call last):
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_validation.py", line 681, in _fit_and_sc
ore
    estimator.fit(X_train, y_train, **fit_params)
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\catboost\core.py", line 4716, in fit
    self._fit(X, y, cat_features, text_features, embedding_features, None, sample_weight, None, None, None, None, baseline, use
_best_model,
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\catboost\core.py", line 2021, in _fit
    train_params = self._prepare_train_params(
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\catboost\core.py", line 1953, in _prepare_train_params
    _check_train_params(params)
  File "_catboost.pyx", line 5839, in _catboost._check_train_params
  File "_catboost.pyx", line 5858, in _catboost._check_train_params
_catboost.CatBoostError: C:/Program Files (x86)/Go Agent/pipelines/BuildMaster/catboost.git/catboost/private/libs/options/obliv
ious_tree_options.cpp:122: Maximum tree depth is 16

  warnings.warn(some_fits_failed_message, FitFailedWarning)
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_search.py:969: UserWarning: One or more of the t
est scores are non-finite: [       nan        nan        nan 0.91170837 0.9215719         nan
 0.92041632 0.91269108 0.90993622 0.91131828 0.91714239 0.92500355
 0.91155106 0.9149677  0.91868252 0.91777753        nan 0.91962131
 0.91747547        nan        nan 0.92089774 0.91380755 0.90737073
 0.92269359 0.91634502 0.91528674 0.91552967        nan        nan
 0.8978479  0.9099864  0.91482701 0.91247667 0.91983445 0.91265607
 0.92045218 0.92326811 0.91592269        nan 0.8944152         nan
 0.91929449 0.9104616         nan 0.91889104 0.92116884 0.91482988
 0.91421558 0.91006694        nan 0.91571699 0.92395213 0.92021043
 0.92017425        nan 0.91325201 0.91253031 0.91668892 0.91071561
 0.91958969 0.91215719 0.92687983 0.90932304 0.92709909        nan
 0.91978533 0.92054325 0.92254685 0.91391639 0.91614067 0.91845189
 0.91053473        nan 0.9096694  0.92034142 0.91276122 0.9232829
 0.91554229 0.92477063 0.9170527  0.9194845  0.92134502 0.91502568
```

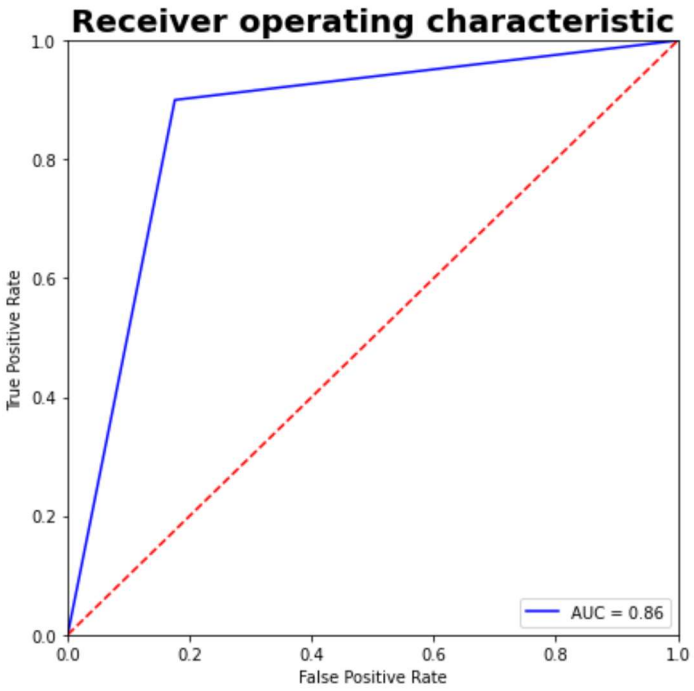
```
 0.910103   0.91523435 0.90838874 0.92186048 0.90967967 0.92237136
 0.92312885 0.91576883 0.91012353 0.91591635 0.91934176 0.92005777
 0.92044026 0.91944999 0.91562746 0.90729338]
  warnings.warn(
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_validation.py:372: FitFailedWarning:
80 fits failed out of a total of 500.
The score on these train-test partitions for these parameters will be set to nan.
If these failures are not expected, you can try to debug them by setting error_score='raise'.

Below are more details about the failures:
--------------------------------------------------------------------------------
80 fits failed with the following error:
Traceback (most recent call last):
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_validation.py", line 681, in _fit_and_sc
ore
    estimator.fit(X_train, y_train, **fit_params)
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\catboost\core.py", line 4716, in fit
    self._fit(X, y, cat_features, text_features, embedding_features, None, sample_weight, None, None, None, None, baseline, use
_best_model,
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\catboost\core.py", line 2021, in _fit
    train_params = self._prepare_train_params(
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\catboost\core.py", line 1953, in _prepare_train_params
    _check_train_params(params)
  File "_catboost.pyx", line 5839, in _catboost._check_train_params
  File "_catboost.pyx", line 5858, in _catboost._check_train_params
_catboost.CatBoostError: C:/Program Files (x86)/Go Agent/pipelines/BuildMaster/catboost.git/catboost/private/libs/options/obliv
ious_tree_options.cpp:122: Maximum tree depth is 16

  warnings.warn(some_fits_failed_message, FitFailedWarning)
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_search.py:969: UserWarning: One or more of the t
est scores are non-finite: [       nan        nan        nan 0.89768374 0.91958721        nan
 0.91992549 0.91002435 0.90485734 0.9076643  0.91167448 0.92090155
 0.90007001 0.89713975 0.91808762 0.89841723        nan 0.91081167
 0.91400285        nan        nan 0.91280131 0.90499632 0.90563725
 0.91568927 0.90942701 0.91080852 0.90260066        nan        nan
 0.87651798 0.90715155 0.91100164 0.9113061  0.9182357  0.9098019
 0.91572454 0.91838613 0.91411396        nan 0.88787157        nan
 0.91449099 0.8994206         nan 0.91377478 0.91658736 0.91193906
 0.90638873 0.90745107        nan 0.91580453 0.91774733 0.91175076
 0.91546087        nan 0.91040026 0.9006619  0.91426169 0.90495588
 0.91722975 0.90792326 0.91917333 0.89884225 0.91744961        nan
 0.91366435 0.91471265 0.91576218 0.90702023 0.91107298 0.90403407
 0.90286063        nan 0.91115264 0.91178739 0.91137856 0.92100917
 0.90829872 0.92378524 0.90070807 0.91883776 0.91268997 0.90407115
 0.9045075  0.91089491 0.89472781 0.91396499 0.90720987 0.91849836
 0.9173749  0.91253864 0.90931196 0.8872248  0.91632324 0.91752612
 0.9151618  0.91238788 0.91140845 0.90518506]
  warnings.warn(
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_validation.py:372: FitFailedWarning:
80 fits failed out of a total of 500.
The score on these train-test partitions for these parameters will be set to nan.
If these failures are not expected, you can try to debug them by setting error_score='raise'.

Below are more details about the failures:
--------------------------------------------------------------------------------
80 fits failed with the following error:
Traceback (most recent call last):
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_validation.py", line 681, in _fit_and_sc
ore
    estimator.fit(X_train, y_train, **fit_params)
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\catboost\core.py", line 4716, in fit
    self._fit(X, y, cat_features, text_features, embedding_features, None, sample_weight, None, None, None, None, baseline, use
_best_model,
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\catboost\core.py", line 2021, in _fit
    train_params = self._prepare_train_params(
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\catboost\core.py", line 1953, in _prepare_train_params
    _check_train_params(params)
  File "_catboost.pyx", line 5839, in _catboost._check_train_params
  File "_catboost.pyx", line 5858, in _catboost._check_train_params
_catboost.CatBoostError: C:/Program Files (x86)/Go Agent/pipelines/BuildMaster/catboost.git/catboost/private/libs/options/obliv
ious_tree_options.cpp:122: Maximum tree depth is 16

  warnings.warn(some_fits_failed_message, FitFailedWarning)
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_search.py:969: UserWarning: One or more of the t
est scores are non-finite: [       nan        nan        nan 0.92245877 0.93207309        nan
 0.92982404 0.92749826 0.92330293 0.92075415 0.92937971 0.93151226
 0.92150097 0.91970989 0.93057709 0.92173436        nan 0.9270518
 0.92746647        nan        nan 0.92076448 0.92469063 0.92191648
 0.92798977 0.92667701 0.92374985 0.91663409        nan        nan
 0.89541873 0.91456097 0.92859015 0.92944858 0.92975214 0.92596721
 0.92914795 0.93158955 0.92532684        nan 0.89656701        nan
```

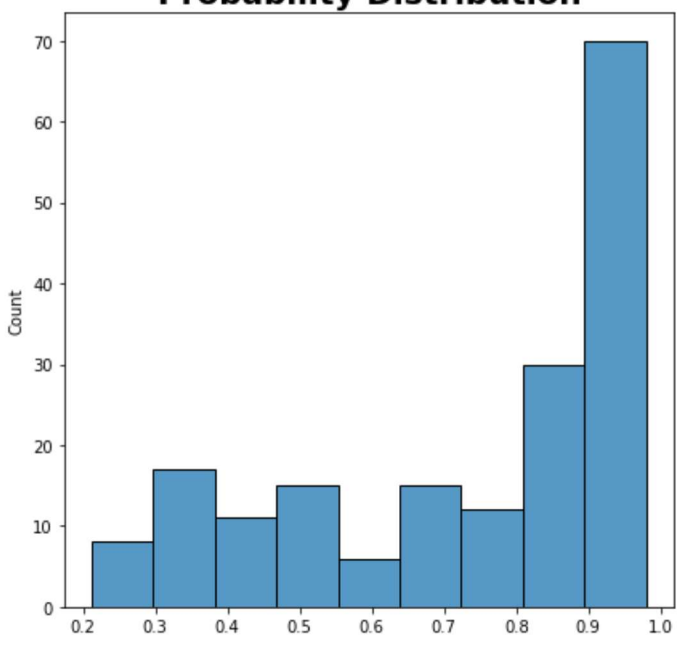
```
 0.92772621 0.90370467        nan 0.92881214 0.93027253 0.92753792
 0.92596418 0.91234933        nan 0.92579487 0.93012704 0.92809886
 0.92753972        nan 0.92749882 0.91813761 0.92772756 0.92190581
 0.92873687 0.92574039 0.92881259 0.91047508 0.92930152        nan
 0.92881461 0.92885259 0.93038577 0.92569815 0.9294506  0.91112904
 0.9246503         nan 0.92255786 0.92061585 0.92386489 0.92633638
 0.92464839 0.93297489 0.91949183 0.93076145 0.92836894 0.91101838
 0.92543818 0.91304633 0.90786551 0.92825176 0.92374827 0.9312842
 0.93064855 0.92809931 0.92611124 0.90927905 0.92967327 0.92900448
 0.92637739 0.9288154  0.92690125 0.92004951]
  warnings.warn(
              precision    recall  f1-score   support

           0       0.85      0.82      0.84        74
           1       0.88      0.90      0.89       110

    accuracy                           0.87       184
   macro avg       0.87      0.86      0.86       184
weighted avg       0.87      0.87      0.87       184

Specificity = 0.8243243243243243
The balanced accuracy score is : 86.22%
```

RandomizedSearchCV CatBoostClassifier Data1 Default









In [56]:

```
cbc = CatBoostClassifier(random_state=32,verbose=False)
build_model(cbc,X2_train,y2_train,X2_test,y2_test,data2,y,label='Data2',params=cbc_params)
```

```
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_validation.py:372: FitFailedWarning:
80 fits failed out of a total of 500.
The score on these train-test partitions for these parameters will be set to nan.
If these failures are not expected, you can try to debug them by setting error_score='raise'.

Below are more details about the failures:
--------------------------------------------------------------------------------
80 fits failed with the following error:
Traceback (most recent call last):
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_validation.py", line 681, in _fit_and_sc
ore
    estimator.fit(X_train, y_train, **fit_params)
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\catboost\core.py", line 4716, in fit
    self._fit(X, y, cat_features, text_features, embedding_features, None, sample_weight, None, None, None, None, baseline, use
_best_model,
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\catboost\core.py", line 2021, in _fit
```

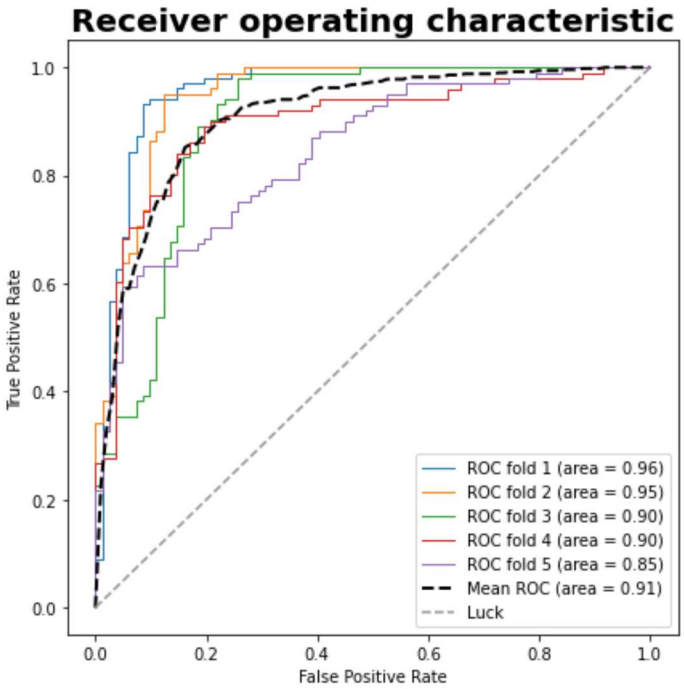
```
    train_params = self._prepare_train_params(
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\catboost\core.py", line 1953, in _prepare_train_params
    _check_train_params(params)
  File "_catboost.pyx", line 5839, in _catboost._check_train_params
  File "_catboost.pyx", line 5858, in _catboost._check_train_params
_catboost.CatBoostError: C:/Program Files (x86)/Go Agent/pipelines/BuildMaster/catboost.git/catboost/private/libs/options/obliv
ious_tree_options.cpp:122: Maximum tree depth is 16

  warnings.warn(some_fits_failed_message, FitFailedWarning)
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_search.py:969: UserWarning: One or more of the t
est scores are non-finite: [       nan        nan        nan 0.92093945 0.92815311        nan
 0.9295349  0.92131478 0.92372443 0.92469102 0.93131305 0.93149398
 0.92032899 0.93012517 0.92779501 0.93217043        nan 0.92658658
 0.93213206        nan        nan 0.92801183 0.92408014 0.92108812
 0.9326774  0.92240673 0.92645353 0.92966479        nan        nan
 0.92330136 0.93242568 0.92794546 0.92173264 0.92917155 0.92202282
 0.92352088 0.93119634 0.92394447        nan 0.92359729        nan
 0.93411337 0.92872187        nan 0.92554114 0.9246784  0.92314847
 0.92599046 0.93424787        nan 0.92491433 0.93314561 0.92771813
 0.92811788        nan 0.92149912 0.92534957 0.92332792 0.92767501
 0.92733366 0.92403538 0.93482438 0.92488097 0.93439762        nan
 0.92757332 0.93168505 0.92588355 0.92727031 0.92950548 0.93456702
 0.92153241        nan 0.92645331 0.92726548 0.92327275 0.93668711
 0.92629149 0.93219777 0.93019451 0.92878686 0.93068206 0.93427079
 0.924719   0.9373548  0.92907397 0.93202446 0.92187548 0.92617539
 0.92857267 0.92336941 0.92183886 0.92753265 0.92887423 0.93295515
 0.92955427 0.93059571 0.92707863 0.92398148]
  warnings.warn(
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_validation.py:372: FitFailedWarning:
80 fits failed out of a total of 500.
The score on these train-test partitions for these parameters will be set to nan.
If these failures are not expected, you can try to debug them by setting error_score='raise'.

Below are more details about the failures:
--------------------------------------------------------------------------------
80 fits failed with the following error:
Traceback (most recent call last):
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_validation.py", line 681, in _fit_and_sc
ore
    estimator.fit(X_train, y_train, **fit_params)
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\catboost\core.py", line 4716, in fit
    self._fit(X, y, cat_features, text_features, embedding_features, None, sample_weight, None, None, None, None, baseline, use
_best_model,
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\catboost\core.py", line 2021, in _fit
    train_params = self._prepare_train_params(
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\catboost\core.py", line 1953, in _prepare_train_params
    _check_train_params(params)
  File "_catboost.pyx", line 5839, in _catboost._check_train_params
  File "_catboost.pyx", line 5858, in _catboost._check_train_params
_catboost.CatBoostError: C:/Program Files (x86)/Go Agent/pipelines/BuildMaster/catboost.git/catboost/private/libs/options/obliv
ious_tree_options.cpp:122: Maximum tree depth is 16

  warnings.warn(some_fits_failed_message, FitFailedWarning)
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_search.py:969: UserWarning: One or more of the t
est scores are non-finite: [       nan        nan        nan 0.89125934 0.90145977        nan
 0.90038617 0.89505072 0.90251371 0.90425635 0.90572281 0.90538288
 0.89267408 0.9039369  0.90364365 0.90284538        nan 0.90155076
 0.90417854        nan        nan 0.90372252 0.89560802 0.89219463
 0.90399346 0.89922758 0.90008844 0.9019841         nan        nan
 0.88627727 0.90652172 0.90119565 0.89969642 0.90164961 0.89763185
 0.8986318  0.90580458 0.90085808        nan 0.88739857        nan
 0.90703286 0.90427508        nan 0.90153067 0.90032713 0.89837574
 0.90027921 0.90637341        nan 0.89947536 0.90442558 0.90192952
 0.90337461        nan 0.89509147 0.90346693 0.89858148 0.89395002
 0.89990418 0.89996352 0.90649185 0.89148372 0.90926936        nan
 0.90282    0.90251765 0.90067763 0.89933659 0.89789268 0.905595
 0.89737371        nan 0.9023602  0.90233191 0.90138306 0.91001233
 0.89709793 0.90729545 0.90452021 0.89872585 0.90507356 0.90488282
 0.89565428 0.90641678 0.89153871 0.90424    0.89426566 0.90190977
 0.9039506  0.89890341 0.89543094 0.90013681 0.90403047 0.90701436
 0.90559884 0.90215526 0.90112638 0.89945886]
  warnings.warn(
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_validation.py:372: FitFailedWarning:
80 fits failed out of a total of 500.
The score on these train-test partitions for these parameters will be set to nan.
If these failures are not expected, you can try to debug them by setting error_score='raise'.

Below are more details about the failures:
--------------------------------------------------------------------------------
80 fits failed with the following error:
Traceback (most recent call last):
```

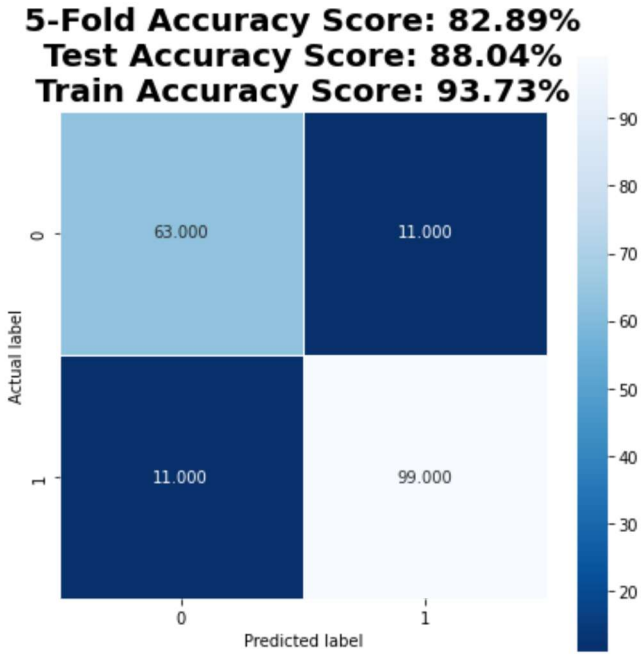
```
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_validation.py", line 681, in _fit_and_sc
ore
    estimator.fit(X_train, y_train, **fit_params)
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\catboost\core.py", line 4716, in fit
    self._fit(X, y, cat_features, text_features, embedding_features, None, sample_weight, None, None, None, None, baseline, use
_best_model,
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\catboost\core.py", line 2021, in _fit
    train_params = self._prepare_train_params(
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\catboost\core.py", line 1953, in _prepare_train_params
    _check_train_params(params)
  File "_catboost.pyx", line 5839, in _catboost._check_train_params
  File "_catboost.pyx", line 5858, in _catboost._check_train_params
_catboost.CatBoostError: C:/Program Files (x86)/Go Agent/pipelines/BuildMaster/catboost.git/catboost/private/libs/options/obliv
ious_tree_options.cpp:122: Maximum tree depth is 16

  warnings.warn(some_fits_failed_message, FitFailedWarning)
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_search.py:969: UserWarning: One or more of the t
est scores are non-finite: [       nan        nan        nan 0.89332173 0.90192714        nan
 0.90280775 0.89573068 0.90291915 0.90205106 0.90082472 0.90506579
 0.89381086 0.90431889 0.90077141 0.90470629        nan 0.90202719
 0.90341477        nan        nan 0.90211643 0.89720899 0.89583652
 0.90255816 0.898815   0.89982829 0.90499789        nan        nan
 0.89661885 0.9024933  0.90146927 0.89426183 0.90505357 0.8974947
 0.89984196 0.90375487 0.899498          nan 0.88709324        nan
 0.90786729 0.89926862        nan 0.89938255 0.90021263 0.89904483
 0.89730105 0.90313067        nan 0.89936497 0.90486638 0.9000209
 0.90203914        nan 0.89653136 0.90339741 0.89922551 0.89642093
 0.90005813 0.89791048 0.90705242 0.89716854 0.90698202        nan
 0.90290896 0.90405533 0.89917879 0.89945207 0.90176241 0.90461596
 0.8969086         nan 0.90389817 0.901621   0.90197583 0.90693266
 0.90143462 0.90476385 0.90442582 0.90024496 0.90279868 0.90396585
 0.89728429 0.90691953 0.89827701 0.90287833 0.89687305 0.90119773
 0.9025284  0.8988961  0.89633725 0.89966549 0.90304086 0.90476802
 0.90186921 0.90211409 0.89981148 0.90260084]
  warnings.warn(
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_validation.py:372: FitFailedWarning:
80 fits failed out of a total of 500.
The score on these train-test partitions for these parameters will be set to nan.
If these failures are not expected, you can try to debug them by setting error_score='raise'.

Below are more details about the failures:
--------------------------------------------------------------------------------
80 fits failed with the following error:
Traceback (most recent call last):
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_validation.py", line 681, in _fit_and_sc
ore
    estimator.fit(X_train, y_train, **fit_params)
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\catboost\core.py", line 4716, in fit
    self._fit(X, y, cat_features, text_features, embedding_features, None, sample_weight, None, None, None, None, baseline, use
_best_model,
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\catboost\core.py", line 2021, in _fit
    train_params = self._prepare_train_params(
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\catboost\core.py", line 1953, in _prepare_train_params
    _check_train_params(params)
  File "_catboost.pyx", line 5839, in _catboost._check_train_params
  File "_catboost.pyx", line 5858, in _catboost._check_train_params
_catboost.CatBoostError: C:/Program Files (x86)/Go Agent/pipelines/BuildMaster/catboost.git/catboost/private/libs/options/obliv
ious_tree_options.cpp:122: Maximum tree depth is 16

  warnings.warn(some_fits_failed_message, FitFailedWarning)
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_search.py:969: UserWarning: One or more of the t
est scores are non-finite: [       nan        nan        nan 0.90687385 0.9126715         nan
 0.91017308 0.90679256 0.91232931 0.91235374 0.91358536 0.91750838
 0.90034873 0.91892938 0.91187883 0.91758355        nan 0.91182083
 0.91847751        nan        nan 0.91423374 0.90555565 0.90787354
 0.91186485 0.90671833 0.90932123 0.91411579        nan        nan
 0.90634812 0.92118562 0.91100257 0.9032334  0.91217429 0.90485038
 0.90913323 0.91439137 0.90913747        nan 0.90538988        nan
 0.91770019 0.91422547        nan 0.90865009 0.91006919 0.90666564
 0.91070634 0.92286567        nan 0.91049408 0.91608959 0.91033664
 0.90949578        nan 0.90668559 0.91129251 0.90758486 0.91098706
 0.91356003 0.90300678 0.92045664 0.90484967 0.91848543        nan
 0.9115713  0.91550123 0.91019078 0.90769635 0.91459679 0.92010517
 0.90376398        nan 0.91227745 0.91246138 0.91115853 0.91993088
 0.90774758 0.91256377 0.91788455 0.90893336 0.91317844 0.91773339
 0.91021651 0.9260837  0.91416896 0.91458553 0.90727344 0.91085572
 0.9159701  0.90734005 0.90716548 0.91144728 0.91064393 0.91564642
 0.91103522 0.91351785 0.9062623  0.91230949]
  warnings.warn(
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_validation.py:372: FitFailedWarning:
80 fits failed out of a total of 500.
```

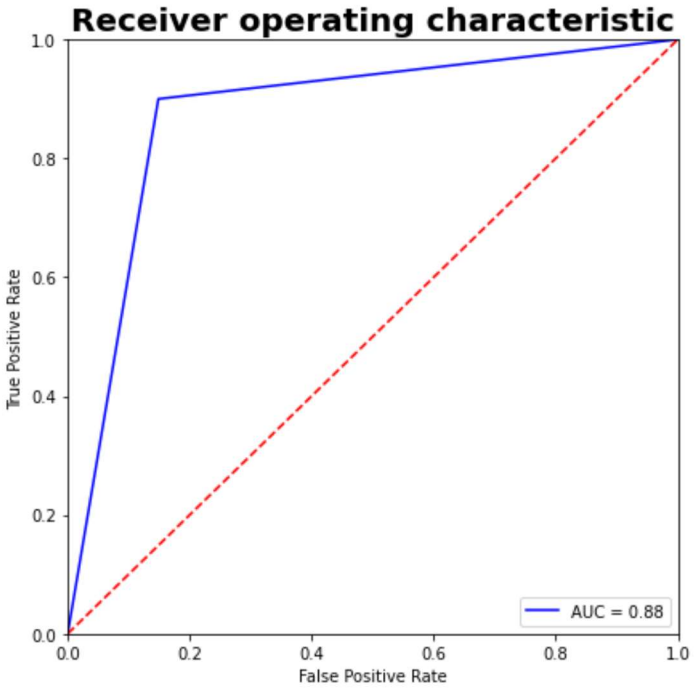
```
The score on these train-test partitions for these parameters will be set to nan.
If these failures are not expected, you can try to debug them by setting error_score='raise'.

Below are more details about the failures:
--------------------------------------------------------------------------------
80 fits failed with the following error:
Traceback (most recent call last):
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_validation.py", line 681, in _fit_and_sc
ore
    estimator.fit(X_train, y_train, **fit_params)
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\catboost\core.py", line 4716, in fit
    self._fit(X, y, cat_features, text_features, embedding_features, None, sample_weight, None, None, None, None, baseline, use
_best_model,
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\catboost\core.py", line 2021, in _fit
    train_params = self._prepare_train_params(
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\catboost\core.py", line 1953, in _prepare_train_params
    _check_train_params(params)
  File "_catboost.pyx", line 5839, in _catboost._check_train_params
  File "_catboost.pyx", line 5858, in _catboost._check_train_params
_catboost.CatBoostError: C:/Program Files (x86)/Go Agent/pipelines/BuildMaster/catboost.git/catboost/private/libs/options/obliv
ious_tree_options.cpp:122: Maximum tree depth is 16

  warnings.warn(some_fits_failed_message, FitFailedWarning)
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_search.py:969: UserWarning: One or more of the t
est scores are non-finite: [       nan        nan        nan 0.90736148 0.91403981        nan
 0.91077684 0.90949756 0.91006389 0.91118825 0.91298971 0.91392387
 0.9077034  0.91145046 0.91088907 0.91474826        nan 0.91092525
 0.9114517         nan        nan 0.91261066 0.91010535 0.90648024
 0.91200983 0.90833781 0.91380905 0.91122364        nan        nan
 0.91002328 0.91576364 0.91010187 0.90759319 0.9113771  0.91036824
 0.90740119 0.91246146 0.91208735        nan 0.90636682        nan
 0.91707528 0.91133655        nan 0.91133992 0.9080747  0.91021421
 0.91077774 0.91651512        nan 0.90888219 0.91823469 0.91216554
 0.91201377        nan 0.90856194 0.91009805 0.91167718 0.91032555
 0.91028544 0.90879124 0.91890921 0.90764273 0.9181984         nan
 0.91328619 0.9124627  0.91193962 0.91131531 0.91156438 0.91527572
 0.90965563        nan 0.91043733 0.91047261 0.91238945 0.92101153
 0.91152663 0.91504676 0.91111208 0.9098028  0.91249944 0.91384949
 0.90653837 0.92018624 0.91272491 0.91534864 0.90801802 0.90893797
 0.91208578 0.91032566 0.90837377 0.91013748 0.91182469 0.91429877
 0.91137497 0.9137367  0.90878933 0.91081212]
  warnings.warn(
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_validation.py:372: FitFailedWarning:
80 fits failed out of a total of 500.
The score on these train-test partitions for these parameters will be set to nan.
If these failures are not expected, you can try to debug them by setting error_score='raise'.

Below are more details about the failures:
--------------------------------------------------------------------------------
80 fits failed with the following error:
Traceback (most recent call last):
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_validation.py", line 681, in _fit_and_sc
ore
    estimator.fit(X_train, y_train, **fit_params)
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\catboost\core.py", line 4716, in fit
    self._fit(X, y, cat_features, text_features, embedding_features, None, sample_weight, None, None, None, None, baseline, use
_best_model,
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\catboost\core.py", line 2021, in _fit
    train_params = self._prepare_train_params(
  File "C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\catboost\core.py", line 1953, in _prepare_train_params
    _check_train_params(params)
  File "_catboost.pyx", line 5839, in _catboost._check_train_params
  File "_catboost.pyx", line 5858, in _catboost._check_train_params
_catboost.CatBoostError: C:/Program Files (x86)/Go Agent/pipelines/BuildMaster/catboost.git/catboost/private/libs/options/obliv
ious_tree_options.cpp:122: Maximum tree depth is 16

  warnings.warn(some_fits_failed_message, FitFailedWarning)
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\model_selection\_search.py:969: UserWarning: One or more of the t
est scores are non-finite: [       nan        nan        nan 0.92813498 0.92866407        nan
 0.92753983 0.92517888 0.92393555 0.92618932 0.92753567 0.92899886
 0.91758504 0.92487285 0.9267879  0.92975124        nan 0.92888685
 0.92618505        nan        nan 0.92978787 0.92678599 0.92397903
 0.92937646 0.92420013 0.92795292 0.93060889        nan        nan
 0.91774845 0.92918783 0.92926265 0.92454211 0.92791315 0.92371502
 0.92663915 0.9298983  0.92716617        nan 0.92393881        nan
 0.92746343 0.92671555        nan 0.9263775  0.92667623 0.92645311
 0.92443077 0.92836164        nan 0.92828985 0.93038689 0.9253647
 0.92727582        nan 0.92536661 0.92277817 0.92465344 0.92551244
 0.9276706  0.92398015 0.93098626 0.92577454 0.93128577        nan
 0.92506328 0.92880956 0.92817952 0.92830867 0.92941083 0.93143755
 0.9231925         nan 0.92360245 0.92952374 0.92450256 0.931246
```

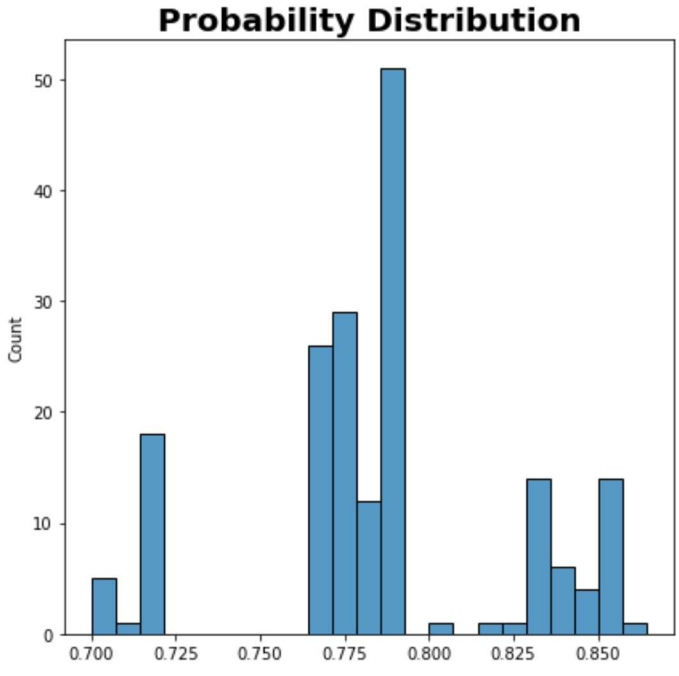
```
0.92633773 0.92941173 0.92787237 0.92828895 0.92547211 0.93038914
0.92446605 0.93136138 0.92230463 0.92993785 0.92398015 0.92645198
0.93023579 0.92622763 0.92442942 0.93035038 0.9269362  0.92712325
0.92772497 0.92738636 0.92499059 0.92393993]
 warnings.warn(
              precision    recall  f1-score   support

           0       0.85      0.85      0.85        74
           1       0.90      0.90      0.90       110

    accuracy                           0.88       184
   macro avg       0.88      0.88      0.88       184
weighted avg       0.88      0.88      0.88       184

Specificity = 0.8513513513513513
The balanced accuracy score is : 87.57%
```

RandomizedSearchCV CatBoostClassifier Data2









CatBoost Feature Selection - Sequential Feature Selection

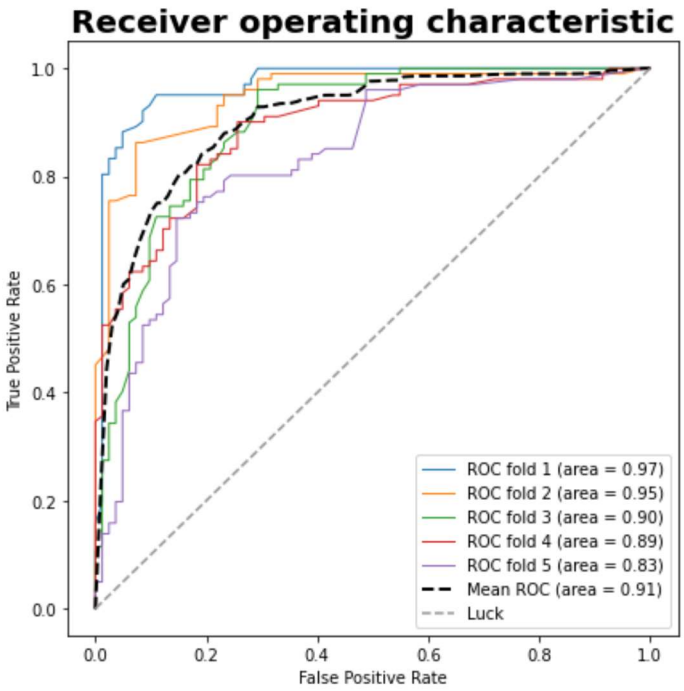
The same 3-5 features are selected once again. However, the predictions achieved are much more varied.

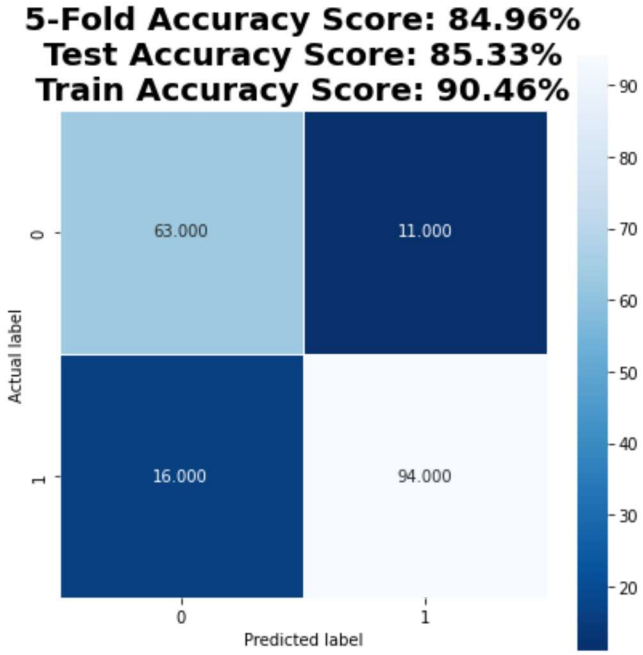
In [57]:

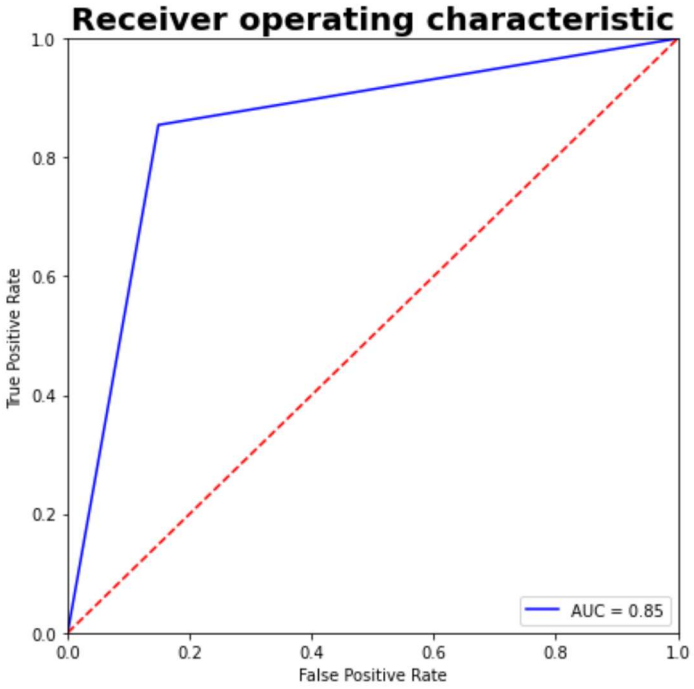
```
cbc = CatBoostClassifier(random_state=32,verbose=False)
sfs = SequentialFeatureSelector(cbc,n_jobs=-1,cv=cv,scoring='roc_auc')
sfs.fit(data2, y)
print(sfs.get_feature_names_out())
build_model(cbc,sfs.transform(X2_train),y2_train,sfs.transform(X2_test),y2_test,
            pd.DataFrame(sfs.transform(data2)),y,label='Data2 FS')
```

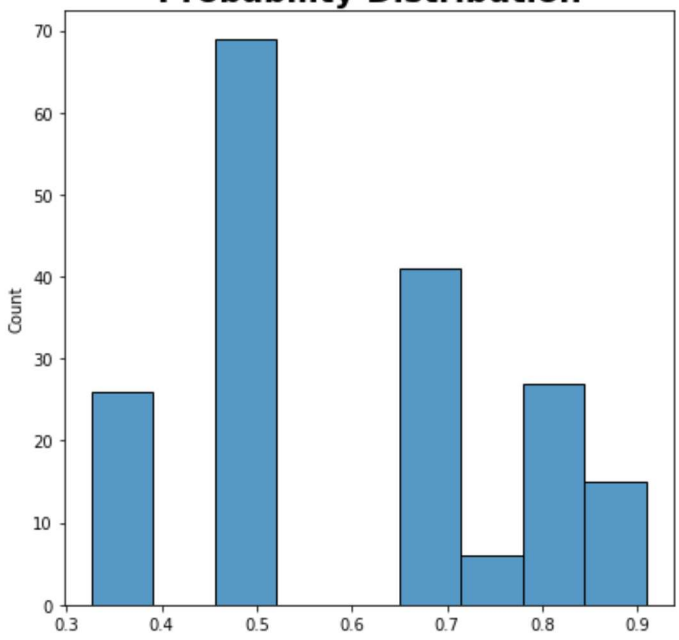
```
['Sex' 'ChestPainType' 'FastingBS' 'Oldpeak' 'ST_Slope']
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\base.py:445: UserWarning: X does not have valid feature names, bu
t SequentialFeatureSelector was fitted with feature names
  warnings.warn(
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\base.py:445: UserWarning: X does not have valid feature names, bu
t SequentialFeatureSelector was fitted with feature names
  warnings.warn(
              precision    recall  f1-score   support

           0       0.80      0.85      0.82        74
           1       0.90      0.85      0.87       110

    accuracy                           0.85       184
   macro avg       0.85      0.85      0.85       184
weighted avg       0.86      0.85      0.85       184

Specificity = 0.8513513513513513
The balanced accuracy score is : 85.29%
```

CatBoostClassifier Data2 FS









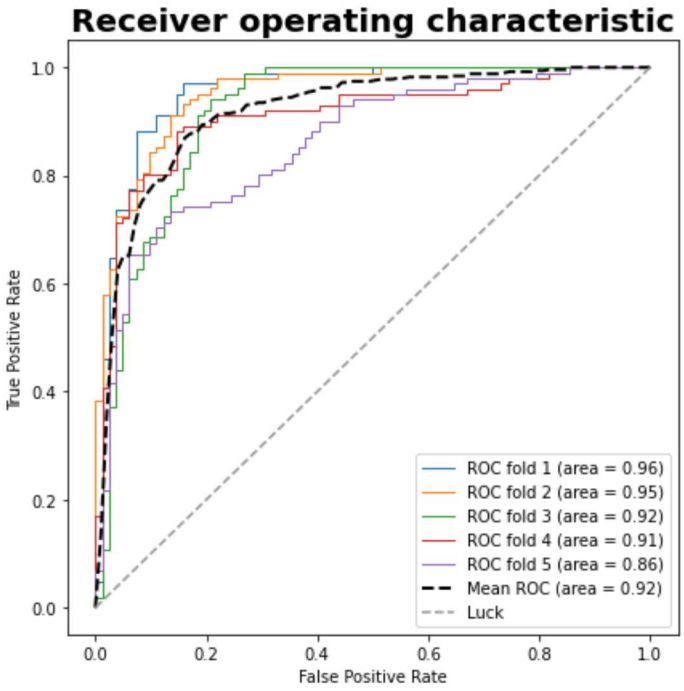
In [58]:

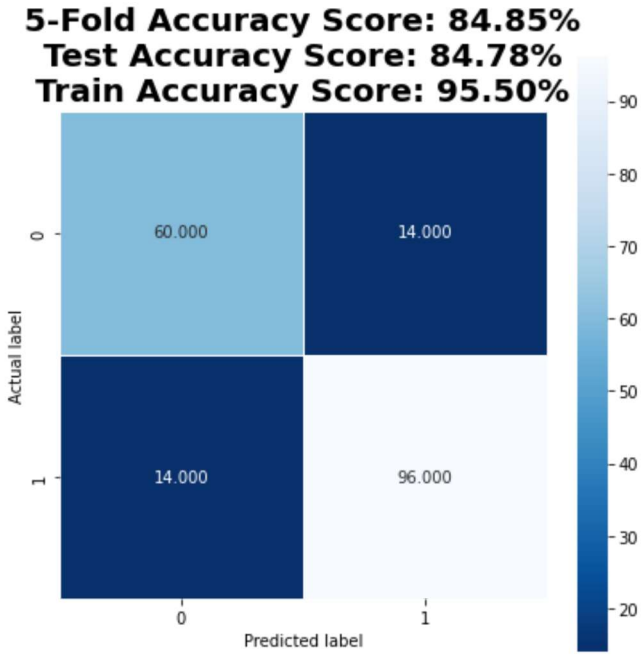
```
cbc = CatBoostClassifier(random_state=32,verbose=False)
sfs = SequentialFeatureSelector(cbc,n_jobs=-1,cv=cv,scoring='roc_auc')
sfs.fit(data1, y)
print(sfs.get_feature_names_out())
build_model(cbc,sfs.transform(X1_train),y1_train,sfs.transform(X1_test),y1_test,
            pd.DataFrame(sfs.transform(data1)),y,label='Data1 FS')
```

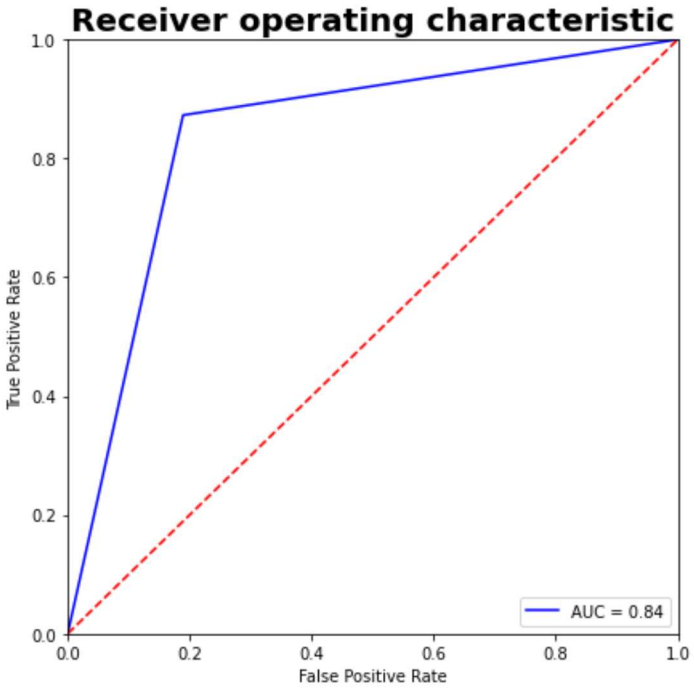
```
['RestingBP' 'FastingBS' 'MaxHR' 'Oldpeak' 'Sex_F' 'Sex_M'
 'ChestPainType_ASY' 'ExerciseAngina_N' 'ST_Slope_Down' 'ST_Slope_Up']
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\base.py:445: UserWarning: X does not have valid feature names, bu
t SequentialFeatureSelector was fitted with feature names
  warnings.warn(
C:\Users\G_MAN\.conda\envs\pymc_env\lib\site-packages\sklearn\base.py:445: UserWarning: X does not have valid feature names, bu
t SequentialFeatureSelector was fitted with feature names
  warnings.warn(
              precision    recall  f1-score   support

           0       0.81      0.81      0.81        74
           1       0.87      0.87      0.87       110
```

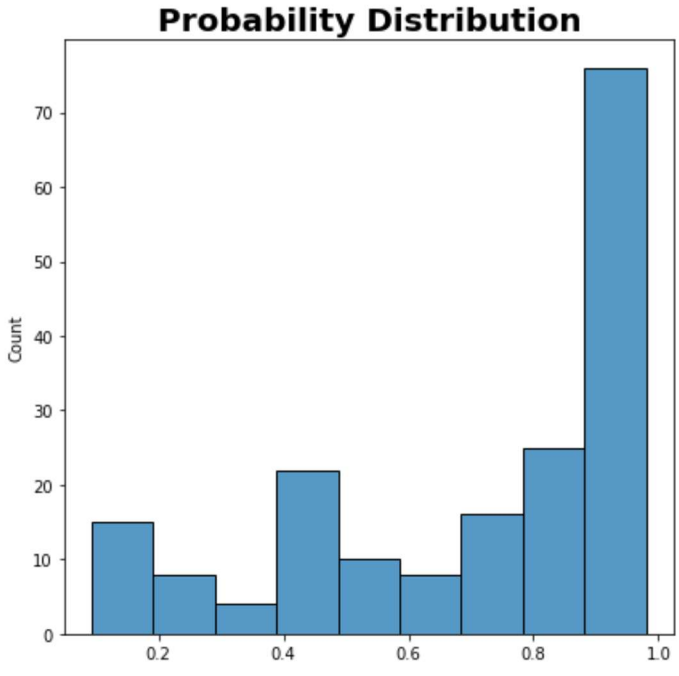
```
    accuracy                           0.85       184
   macro avg       0.84      0.84      0.84       184
weighted avg       0.85      0.85      0.85       184

Specificity = 0.8108108108108109
The balanced accuracy score is : 84.18%
```

CatBoostClassifier Data1 FS









Summary of Results

A total of 42 different models were built around predicting heart disease in a patient. With a total of 11 unique features that were composed of both categorical and continuous values. Two different data set samples were experimented with, one using one-hot encoding for the categorical features and the other using Weight of Evidence transformation. Each of the data samples were trained in three ways. First, by training the model using the default values. Second, by performing a randomized search for hyper-parameters tuning; and lastly by performing feature selection.

What are your findings?

All the models attained a test accuracy score between 82% - 86% and AUC values between 0.82 - 0.86. Shown below are the summarized table of results for all the models trained. Data1 represents the data sample using one-hot encoding transformation while Data2 represents the data sample using Weight of Evidence transformation. The 42 models were trained between 7 different classifiers. Unfortunately there is not one model that truly outperforms the others as all models achieved very similar scores with marginal improvements between each other. What stands out the most is how balanced the models are in predicting either class. Nearly all cases were biased towards patients with heart disease. Meaning, the model would most likely classify as having heart disease. This is due to the unbalanced nature of the data where the split was around 55%/44% having and not having heart disease. Two of the most generalizable models were the random forest classifier and XGBoost classifier. Those two models had the most balanced probability distributions and had balanced metrics between classes. They did not have the highest accuracy or AUC such as Catboost, but they had the most balanced performance metrics as shown by the F-1 scores, confusion matrices and probability distributions. The highest performance model is the Catboost model trained on Data2, Weight of Evidence tranformation. However, as stated before, it is biased towards heart disease. Since the results are only a marginal improvement over all the other models, the final model would have to be either random forest or XGBoost due to their generalizability and efficiency. Additionally, experimenting between one-hot encoding and weight of evidence had marginally similar results where the dataset trained by the data sample transformed by weight of evidence performed better than one-hot encoding. In other words, transforming the categorical variables into weight of evidence enabled a marginally better more parsimonious model. Lastly, another interesting finding between the models was the constant overlap between selected features during feature selection. In nearly every case 3-5 features were consistently selected as important features using sequential feature selection. It appears that sex, chest pain type, fasting blood sugar, old peak, and st slope, are important predictive features in determining heart disease.

If you had more time what do you think can be done further to improve the results?

If given more time, I would perform SMOTE, or over/under sampling of the data in order to achieve a more generalized model. As stated before, the dataset was unbalanced with a 55%/44% split between having and not having heart disease. Although it is not the most unbalanced set, SMOTE, and over/under sampling well help artificially create a balanced dataset which will ultimately train a more generalizable model.

Default Models

Model	Dataset	Train Accuracy	Test Accuracy	5-Fold Test Accuracy	AUC	5-Fold AUC
Logistic Regression	Data1	86.51%	84.78%	82.89%	0.84	0.91
Logistic Regression	Data2	87.47%	85.33%	82.35%	0.85	0.91
Random Forest	Data1	100%	86.96%	83.00%	0.86	0.91
Random Forest	Data2	100%	85.33%	83.69%	0.85	0.90
Support Vector Mch	Data1	87.47%	85.33%	83.00%	0.85	0.91
Support Vector Mch	Data2	87.06%	85.33%	82.02%	0.85	0.90
XGBoost	Data1	100%	84.24%	81.36%	0.84	0.90
XGBoost	Data2	100%	82.61%	82.02%	0.82	0.90
LightGBM	Data1	100%	84.78%	82.67%	0.84	0.90
LightGBM	Data2	100%	83.70%	81.58%	0.83	0.90
AdaBoost	Data1	88.96%	82.61%	81.59%	0.83	0.87
AdaBoost	Data2	89.37%	83.15%	80.28%	0.83	0.87
CatBoost	Data1	97.82%	86.96%	83.76%	0.86	0.91
CatBoost	Data2	97.55%	85.87%	83.87%	0.85	0.91

Randomized Search Models

Model	Dataset	Train Accuracy	Test Accuracy	5-Fold Test Accuracy	AUC	5-Fold AUC
Logistic Regression RS	Data1	86.38%	86.96%	82.35%	0.86	0.91
Logistic Regression RS	Data2	86.24%	85.33%	81.37%	0.85	0.90
Random Forest RS	Data1	92.10%	88.59%	83.98%	0.88	0.91
Random Forest RS	Data2	93.73%	86.41%	82.78%	0.86	0.91
Support Vector Mch RS	Data1	87.47%	85.33%	83.44%	0.85	0.90
Support Vector Mch RS	Data2	87.06%	85.33%	81.47%	0.85	0.90

Model	Dataset	Train Accuracy	Test Accuracy	5-Fold Test Accuracy	AUC	5-Fold AUC
XGBoost RS	Data1	92.51%	86.96%	83.11%	0.86	0.89
XGBoost RS	Data2	91.96%	88.04%	83.43%	0.87	0.89
LightGBM RS	Data1	94.14%	85.87%	82.35%	0.85	0.89
LightGBM RS	Data2	90.19%	87.50%	82.35%	0.86	0.89
AdaBoost RS	Data1	87.60%	86.41%	82.89%	0.86	0.89
AdaBoost RS	Data2	87.60%	86.41%	82.89%	0.86	0.89
CatBoost RS	Data1	97.28%	86.96%	83.54%	0.86	0.91
CatBoost RS	Data2	93.73%	88.04%	82.89%	0.88	0.91

Feature Selected Models

Model	Dataset	Train Accuracy	Test Accuracy	5-Fold Test Accuracy	AUC	5-Fold AUC
Logistic Regression FS	Data1	86.65%	84.24%	84.41%	0.84	0.92
Logistic Regression FS	Data2	85.97%	84.24%	83.98%	0.84	0.91
Random Forest FS	Data1	91.69%	83.70%	83.65%	0.84	0.89
Random Forest FS	Data2	86.51%	82.61%	83.65%	0.83	0.89
Support Vector Mch FS	Data1	85.69%	83.15%	84.20%	0.83	0.92
Support Vector Mch FS	Data2	81.47%	80.98%	79.07%	0.80	0.90
XGBoost FS	Data1	86.51%	82.07%	80.93%	0.82	0.89
XGBoost FS	Data2	91.28%	82.61%	82.99%	0.82	0.89
LightGBM FS	Data1	91.14%	84.24%	84.19%	0.84	0.91
LightGBM FS	Data2	90.74%	83.15%	83.54%	0.83	0.90
AdaBoost FS	Data1	86.24%	83.70%	82.13%	0.83	0.91
AdaBoost FS	Data2	86.10%	83.70%	85.17%	0.83	0.91
CatBoost FS	Data1	95.50%	84.78%	84.85%	0.84	0.92
CatBoost FS	Data2	90.46%	85.33%	84.96%	0.85	0.91

In [ ]: