In [ ]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import (
    accuracy_score,
    precision_score,
    recall_score,
    f1_score,
    classification_report,
    confusion_matrix,
    roc_curve,
    auc,
    RocCurveDisplay
)
sns.set_style("ticks")
sns.set_context("paper")
%matplotlib inline
```

In [ ]:
```
df = pd.read_csv("archive/winequalityN.csv", sep=",")
df.head()
```

Out[ ]:

	type	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	a
0	white	7.0	0.27	0.36	20.7	0.045	45.0	170.0	1.0010	3.00	0.45	
1	white	6.3	0.30	0.34	1.6	0.049	14.0	132.0	0.9940	3.30	0.49	
2	white	8.1	0.28	0.40	6.9	0.050	30.0	97.0	0.9951	3.26	0.44	
3	white	7.2	0.23	0.32	8.5	0.058	47.0	186.0	0.9956	3.19	0.40	
4	white	7.2	0.23	0.32	8.5	0.058	47.0	186.0	0.9956	3.19	0.40	

Faça o download da base - esta é uma base real, apresentada no artigo: P. Cortez, A. Cerdeira, F. Almeida, T. Matos and J. Reis. Modeling wine preferences by data mining from physicochemical properties. In Decision Support Systems, Elsevier, 47(4):547-553, 2009.

Ela possui uma variável denominada "quality", uma nota de 0 a 10 que denota a qualidade do vinho. Crie uma nova variável, chamada "opinion" que será uma variável categórica igual à 0, quando quality for menor e igual à 5. O valor será 1, caso contrário. Desconsidere a variável quality para o restante da análise.

In [ ]:
```
df['opinion'] = (df['quality'] > 5).astype(int)
df.head()
```

Out[ ]:

	type	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	a
0	white	7.0	0.27	0.36	20.7	0.045	45.0	170.0	1.0010	3.00	0.45	
1	white	6.3	0.30	0.34	1.6	0.049	14.0	132.0	0.9940	3.30	0.49	
2	white	8.1	0.28	0.40	6.9	0.050	30.0	97.0	0.9951	3.26	0.44	
3	white	7.2	0.23	0.32	8.5	0.058	47.0	186.0	0.9956	3.19	0.40	
4	white	7.2	0.23	0.32	8.5	0.058	47.0	186.0	0.9956	3.19	0.40	

In [ ]:
```
df['type'].value_counts()
```

Out[ ]:
```
type
white    4898
red      1599
Name: count, dtype: int64
```

In [ ]:
```
#criando novo DataFrame vinho branco
df_white = df[df['type']=='white'].copy()
df_white.drop('quality', axis=1, inplace=True)
```

Descreva as variáveis presentes na base. Quais são as variáveis? Quais são os tipos de variáveis (discreta, categórica, contínua)? Quais são as médias e desvios padrões?

In [ ]:
```
#Analise estatistica
df_white.describe()
```

Out[ ]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide
count	4890.000000	4891.000000	4896.000000	4896.000000	4896.000000	4898.000000	4898.000000
mean	6.855532	0.278252	0.334250	6.393250	0.045778	35.308085	138.360657
std	0.843808	0.100811	0.120985	5.072275	0.021850	17.007137	42.498065
min	3.800000	0.080000	0.000000	0.600000	0.009000	2.000000	9.000000
25%	6.300000	0.210000	0.270000	1.700000	0.036000	23.000000	108.000000
50%	6.800000	0.260000	0.320000	5.200000	0.043000	34.000000	134.000000
75%	7.300000	0.320000	0.390000	9.900000	0.050000	46.000000	167.000000
max	14.200000	1.100000	1.660000	65.800000	0.346000	289.000000	440.000000

In [ ]:
```
#Detalhando tipos de variaveis
df_white.dtypes
#object  - nao numerico   - qualitativa nominal
#float64 - numero real    - variavel continua
#int32   - numero inteiro - variavel descreta
```

Out[ ]:
```
type                   object
fixed acidity         float64
volatile acidity      float64
citric acid           float64
residual sugar        float64
chlorides             float64
free sulfur dioxide   float64
total sulfur dioxide  float64
density               float64
pH                    float64
sulphates             float64
alcohol               float64
opinion                 int32
dtype: object
```

4-Com a base escolhida:

a) Descreva as etapas necessárias para criar um modelo de classificação eficiente.

1 - Limpar NaNs e outliers (usando melhor método - moda, média, regressão linear etc.)

2 - Escolha do modelo

3 - Separar conjunto de treino e teste

4 - Treinar modelo

5 - Rodar modelo com variáveis de teste para comparar o yhat/ypred (previsto no teste) com os targets do y de treino (geralmente y_test)

6 - Utilizar f1-score, acurácia, precisão e recall para analisar a eficiência.

b)Treine um modelo de regressão logística usando um modelo de validação cruzada estratificada com k-folds (k=10) para realizar a classificação. Calcule para a base de teste:

i. a média e desvio da acurácia dos modelos obtidos;

ii. a média e desvio da precisão dos modelos obtidos;

iii. a média e desvio da recall dos modelos obtidos;

iv. a média e desvio do f1-score dos modelos obtidos.

In [ ]:
```
#analisando DataFrame
df_white.shape
```

Out[ ]:
```
(4898, 13)
```

In [ ]:
```
df_white.info() # checando dados nulos
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 4898 entries, 0 to 4897
Data columns (total 13 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   type                  4898 non-null   object
 1   fixed acidity         4890 non-null   float64
 2   volatile acidity      4891 non-null   float64
 3   citric acid           4896 non-null   float64
 4   residual sugar        4896 non-null   float64
 5   chlorides             4896 non-null   float64
 6   free sulfur dioxide   4898 non-null   float64
 7   total sulfur dioxide  4898 non-null   float64
 8   density               4898 non-null   float64
 9   pH                    4891 non-null   float64
 10  sulphates             4896 non-null   float64
 11  alcohol               4898 non-null   float64
 12  opinion               4898 non-null   int32
dtypes: float64(11), int32(1), object(1)
memory usage: 516.6+ KB
```

In [ ]:
```
df_white.dropna(inplace=True)
```

In [ ]:
```
print(df_white.isnull().sum()) # print dos NaN
```

```
type                  0
fixed acidity         0
volatile acidity      0
citric acid           0
residual sugar        0
chlorides             0
free sulfur dioxide   0
total sulfur dioxide  0
density               0
pH                    0
sulphates             0
alcohol               0
opinion               0
dtype: int64
```

In [ ]:
```
#resetando o index
df_white.reset_index(inplace=True)
```

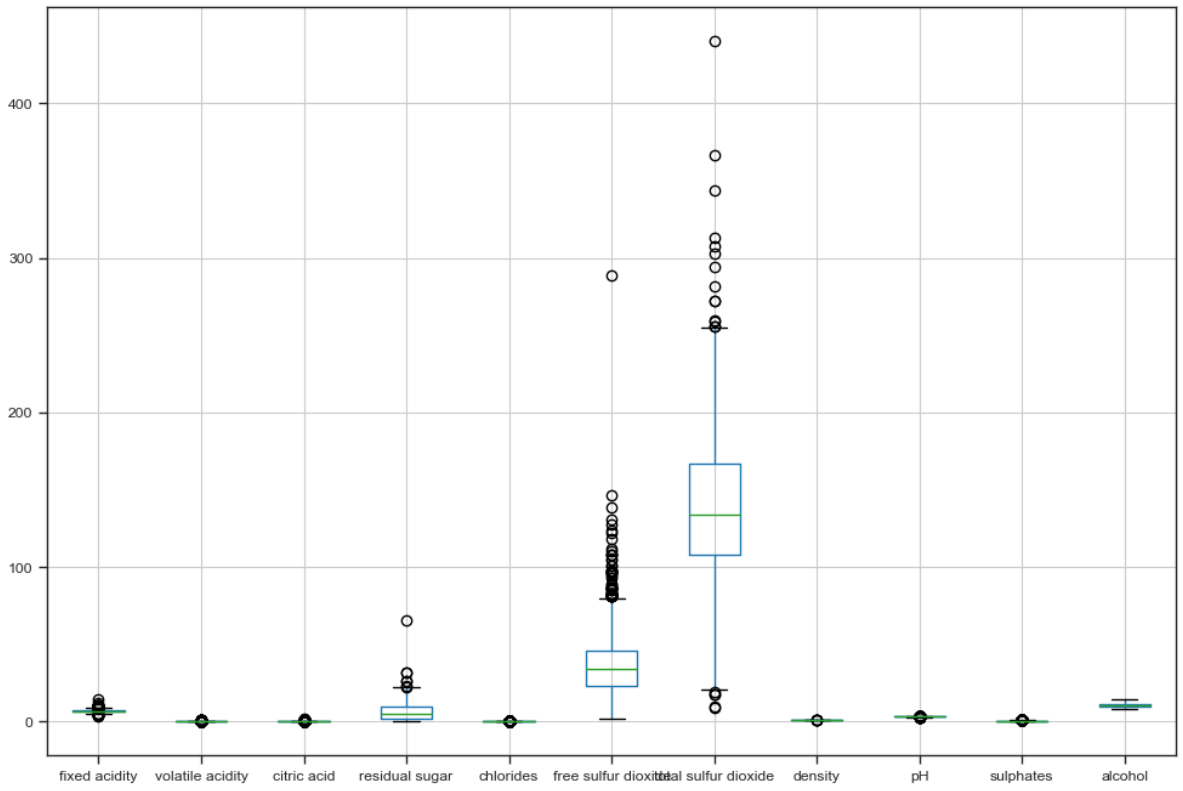
In [ ]:
```
#apagando o index antigo
df_white = df_white.drop('index', axis=1)
df_white.head()
```

Out[ ]:

	type	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	a
0	white	7.0	0.27	0.36	20.7	0.045	45.0	170.0	1.0010	3.00	0.45	
1	white	6.3	0.30	0.34	1.6	0.049	14.0	132.0	0.9940	3.30	0.49	
2	white	8.1	0.28	0.40	6.9	0.050	30.0	97.0	0.9951	3.26	0.44	
3	white	7.2	0.23	0.32	8.5	0.058	47.0	186.0	0.9956	3.19	0.40	
4	white	7.2	0.23	0.32	8.5	0.058	47.0	186.0	0.9956	3.19	0.40	

In [ ]:
```
#checando outliers
df_white.drop('opinion', axis=1).boxplot(figsize=[12,8])
```

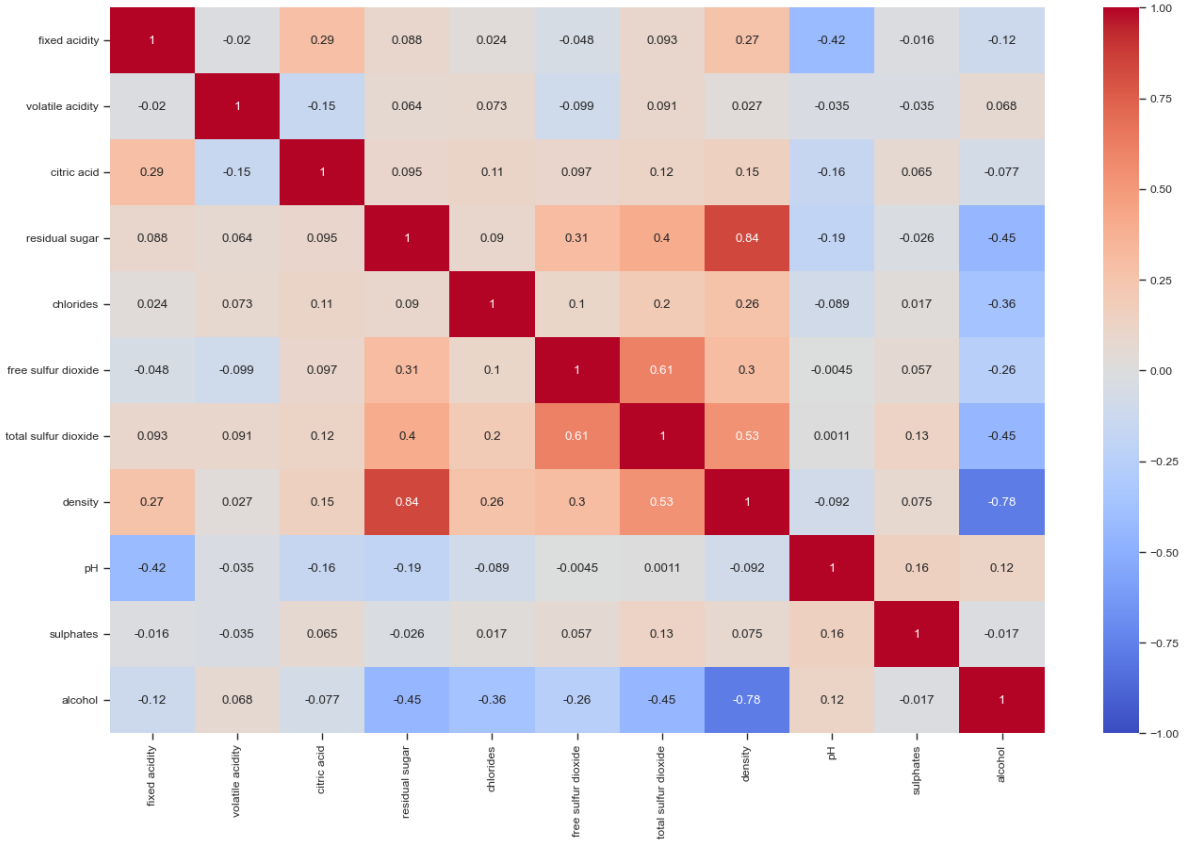
Out[ ]:
```
<Axes: >
```



In [ ]:
```
#limpando outliers free sulfur dioxide > 200
outlier = df_white[df_white['free sulfur dioxide'] > 200 ].index

df_white = df_white.drop(index = outlier)
```

In [ ]:
```
#Criando variaveis regressao logistica
vars = [
   'fixed acidity',
   'volatile acidity',
   'citric acid',
   'residual sugar',
   'chlorides',
   'free sulfur dioxide',
   'total sulfur dioxide',
   'density',
   'pH',
   'sulphates',
   'alcohol'
]
```

In [ ]:
```
#plotando heatmap
fig, ax = plt.subplots(1, 1, figsize=(16, 10))
sns.heatmap(df_white[vars].corr(), vmax=1, vmin=-1, annot=True, ax=ax, cmap="coolw
```



In [ ]:
```
X_train, X_test, y_train, y_test = train_test_split(df_white[vars],
                                                    df_white['opinion'],
                                                    test_size=0.2,
                                                    random_state=42,
                                                    stratify=df_white['opinion'])

scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

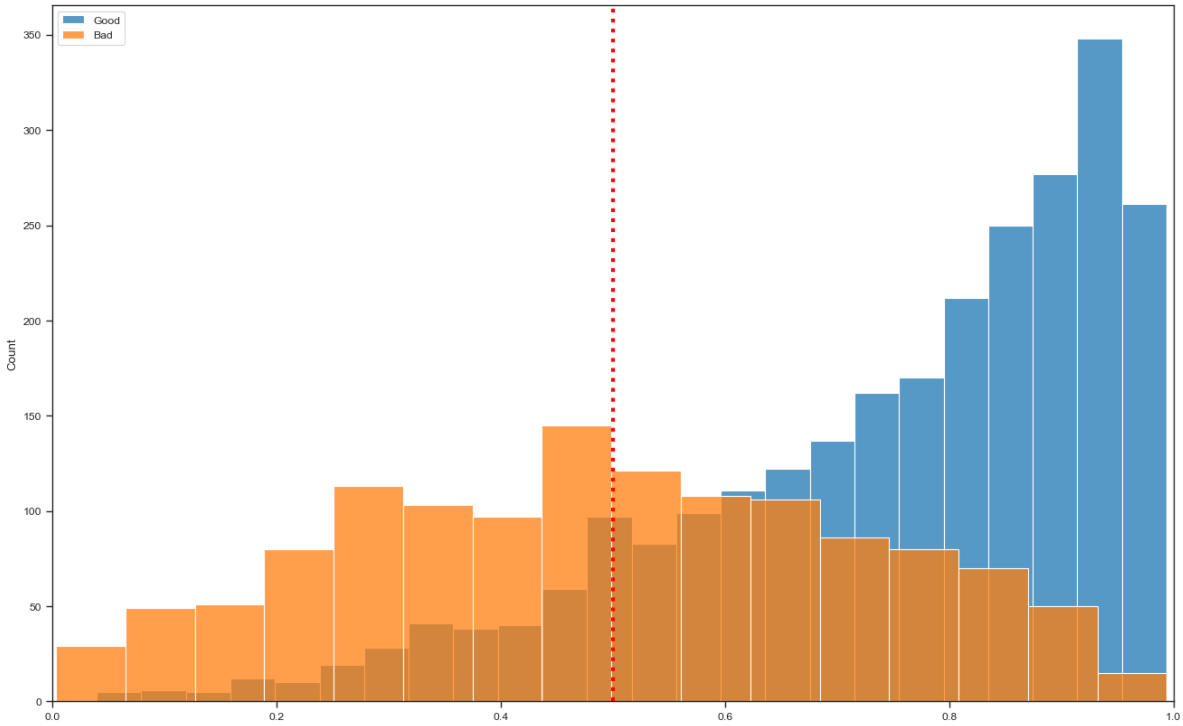
In [ ]:
```
#criando modelo de regressao
logreg = LogisticRegression(max_iter=10000)


logreg.fit(X_train_scaled, y_train)
```

Out[ ]:
```
LogisticRegression
LogisticRegression(max_iter=10000)
```

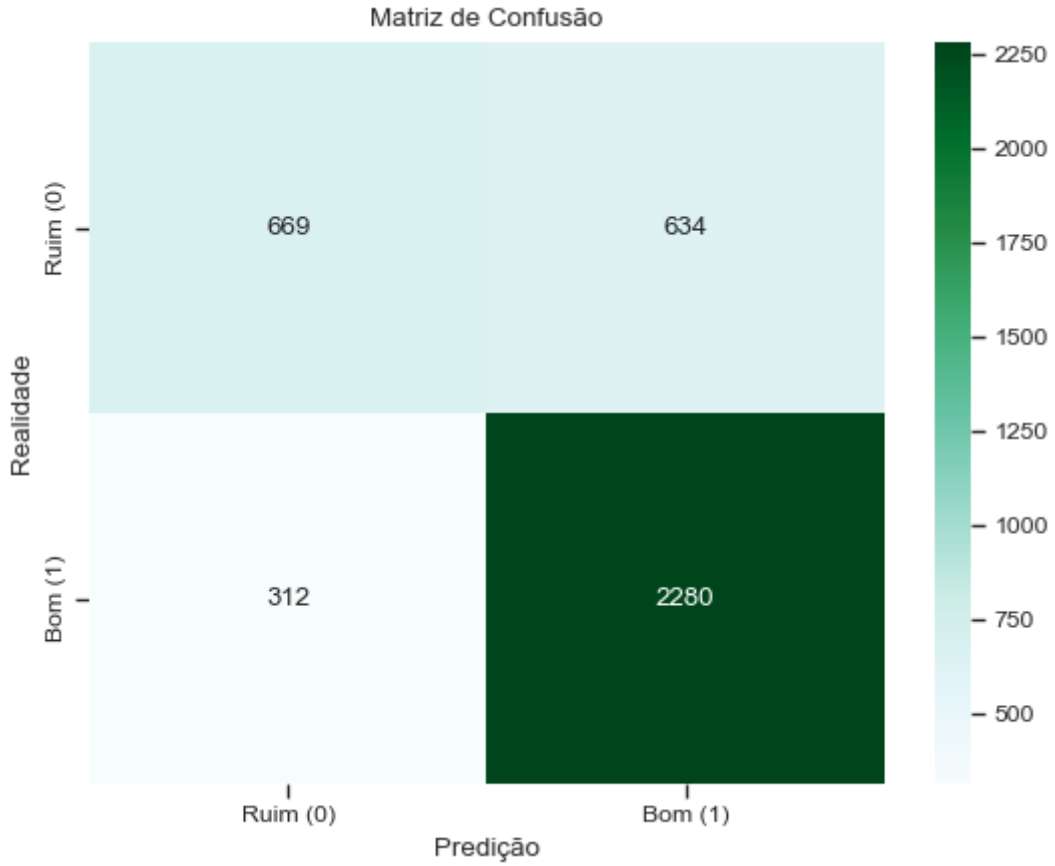
In [ ]:
```
#fazendo predicao
y_hat = logreg.predict_proba(X_train_scaled)
print(y_hat.shape)
```

```
(3895, 2)
```

In [ ]:
```
#plotagem
fig, ax = plt.subplots(1, 1, figsize=(16, 10))
sns.histplot(y_hat[y_train.values == 1, 1], label="Good", ax=ax)
ax.set_xlim([0, 1])
sns.histplot(y_hat[y_train == 0, 1], label="Bad", ax=ax)
ax.legend();
ax.axvline(0.5, color="red", ls=":", lw=3);
```



In [ ]:
```
#matrix de confusao
y_pred = logreg.predict(X_train_scaled)

cm = confusion_matrix(y_train, y_pred)

#cm = np.array([[434, 161], [1168, 516]])

ax = sns.heatmap(cm, cmap="BuGn", annot=True, fmt='g')
ax.set_xlabel("Predição")
ax.set_ylabel("Realidade")
ax.set_title("Matriz de Confusão")

ax.set_xticklabels(["Ruim (0)", "Bom (1)"]);
ax.set_yticklabels(["Ruim (0)", "Bom (1)"]);
```



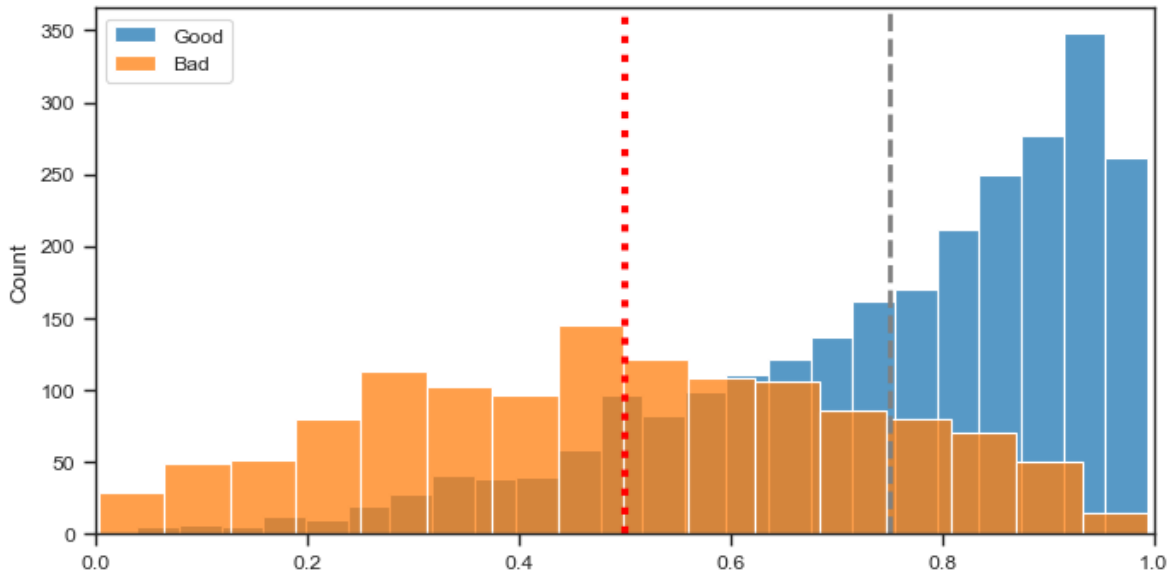
In [ ]:
```
#metricas
precision = cm[1, 1] / (cm[1, 1] + cm[0, 1])

print(f"A precisão é {100*  precision:.2f} %")

accuracy = (cm[0, 0] + cm[1, 1]) / np.sum(cm)

print(f"A acurácia é {100 * accuracy:.2f} %")

# Sensibilidade ou taxa de verdadeiro positivo
sensibility = (cm[1, 1] / (cm[1, 1] + cm[1, 0]))
print(f"A sensibilidade é {100 *  sensibility:.2f} %")

specificity = (cm[0,0] / (cm[0, 0] + cm[0 ,1]))

print(f"A especificidade é {100 * specificity:.2f} %")

# (1 - specificity) ou taxa de falsos positivos

F1_score = 2 *(sensibility * precision) / (sensibility + precision)

print(f"F1 Score =  {F1_score:.2f}")
```

```
A precisão é 78.24 %
A acurácia é 75.71 %
A sensibilidade é 87.96 %
A especificidade é 51.34 %
F1 Score =  0.83
```

In [ ]:
```
#plottagem da distribuicao
def plot_distributions(model, X, y, ax=None):
    y_hat = model.predict_proba(X)
    if ax == None:
        fig, ax = plt.subplots(1, 1, figsize=(8, 4))
    sns.histplot(y_hat[y.values == 1, 1], label="Good", ax=ax)
```

```
    ax.set_xlim([0, 1])
    sns.histplot(y_hat[y == 0, 1], label="Bad", ax=ax)
    ax.legend();
    return ax

ax = plot_distributions(logreg, X_train_scaled, y_train)


ax.axvline(0.5, lw=3, color='red', ls=":");
ax.axvline(0.75, lw=2, color='gray', ls="--")
```

Out[ ]:
```
<matplotlib.lines.Line2D at 0x2a8ec200e80>
```



In [ ]:
```
#metodos de metricas
print(f"A acurácia é {100 * accuracy_score(y_train, y_pred):.2f} %")
print(f"A sensibilidade é {100 *  recall_score(y_train, y_pred):.2f} %")
print(f"A precisão é {100*  precision_score(y_train, y_pred):.2f} %")
```

```
A acurácia é 75.71 %
A sensibilidade é 87.96 %
A precisão é 78.24 %
```

In [ ]:
```
#report
print(classification_report(y_train, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.68      0.51      0.59      1303
           1       0.78      0.88      0.83      2592

    accuracy                           0.76      3895
   macro avg       0.73      0.70      0.71      3895
weighted avg       0.75      0.76      0.75      3895
```

In [ ]:
```
#calculando threshold
thresholds = np.linspace(0, 1, 101)
y_hat = logreg.predict_proba(X_train)


for threshold in thresholds:
    predictions = []
    if (y_hat > threshold).any():
        y_pred = 1.
    else:
```

```
        y_pred = 0.
    predictions.append(y_pred)
```

```
c:\Users\user\miniconda3\envs\Bootcamp\lib\site-packages\sklearn\base.py:432: User
Warning: X has feature names, but LogisticRegression was fitted without feature na
mes
  warnings.warn(
```

In [ ]:
```
#predicao metricas
y_hat = logreg.predict_proba(X_train_scaled)
thresholds = np.linspace(0, 1, 101)

def specificity_score(y, y_pred):
    cm = confusion_matrix(y, y_pred)
    specificity = (cm[0,0] / (cm[0, 0] + cm[0 ,1]))
    return specificity

def predict(model, X, threshold, pos_label=1):
    y_hat = model.predict_proba(X)
    y_pred = []
    for prob_tuple in y_hat:
        prob = prob_tuple[pos_label]
        if (prob > threshold).any():
            y_pred.append(1.)
        else:
            y_pred.append(0.)
    return np.array(y_pred)

#def predict(model, X, threshold, pos_label=1):
#    y_hat = model.predict_proba(X)
#    y_pred = (y_hat[:, pos_label] > threshold)
#    return y_pred.astype(float)
# Probabilidade de ter um vinho bom

recall = []
precision = []
specificity = []
f1 = []
for threshold in thresholds:
    y_pred_thr = predict(logreg, X_train_scaled, threshold)
    recall.append(recall_score(y_train, y_pred_thr))
    precision.append(precision_score(y_train, y_pred_thr))
    specificity.append(specificity_score(y_train, y_pred_thr))
    f1.append(f1_score(y_train, y_pred_thr))
```

```
c:\Users\user\miniconda3\envs\Bootcamp\lib\site-packages\sklearn\metrics\_classifi
cation.py:1344: UndefinedMetricWarning: Precision is ill-defined and being set to
0.0 due to no predicted samples. Use `zero_division` parameter to control this beh
avior.
  _warn_prf(average, modifier, msg_start, len(result))
```

In [ ]:
```
#plotagem ROC     fig, ax = plt.subplots(1, 1, figsize=(10, 6))

ax.plot(thresholds, recall, color="orange", label="recall")
ax.plot(thresholds, precision, color="navy", label="precision")
ax.plot(thresholds, f1, color="olive", label="f1")

f1_max = max(f1)
thr_arg_max = np.argmax(f1)
thr_max = thresholds[thr_arg_max]

ax.axvline(thr_max, color="red", ls=":")
ax.axvline(0.5, color="gray",lw=0.5, ls="--")
ax.axhline(f1_max, color="red", ls=":")
```

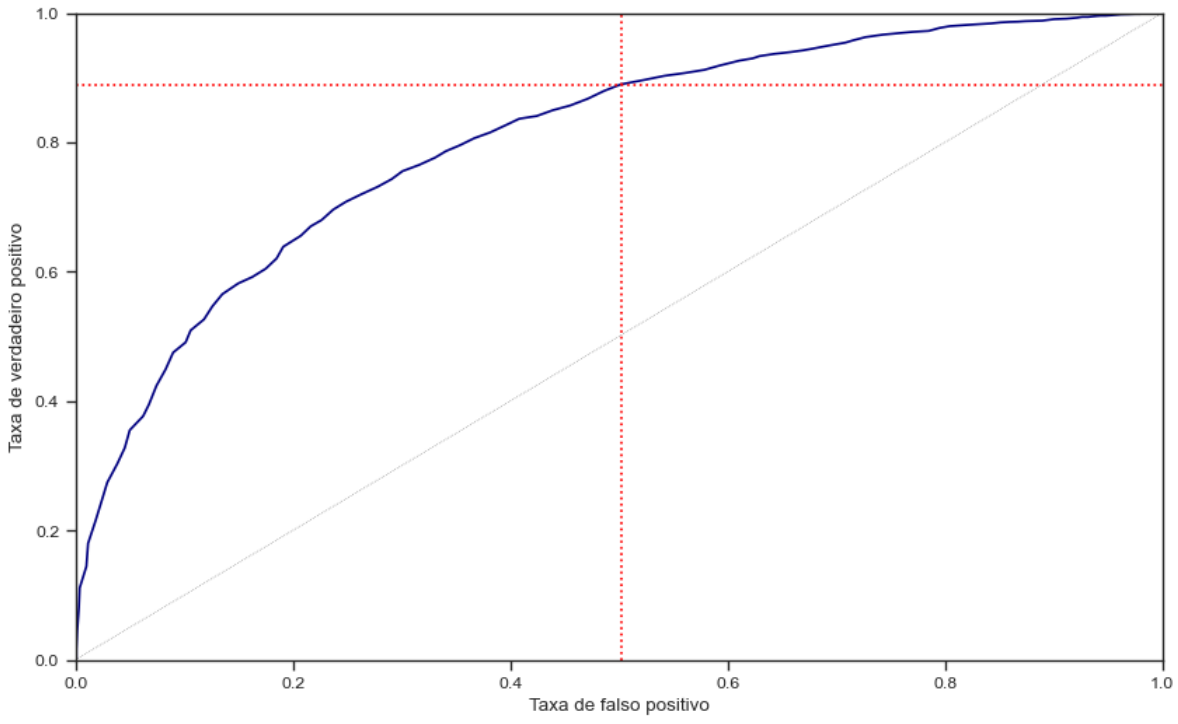
```
ax.legend()
ax.set_ylim([0.5, 1])
print(f"f1 máximo: {f1_max:.2f} - ponto de operação: {thr_max:.2f}")
print(f"Recall: {recall[thr_arg_max]:.2f} - Precision: {precision[thr_arg_max]:.2f
sns.despine(offset=10)
```

```
f1 máximo: 0.83 - ponto de operação: 0.49
Recall: 0.89 - Precision: 0.78
<Figure size 640x480 with 0 Axes>
```

In [ ]:
```
#verrdadeiro e falso positivo
tpr = recall
fpr = [(1 - s) for s in specificity]
```

In [ ]:
```
#plotagem de verdadeiro e falso poositivo
fig, ax = plt.subplots(1, 1, figsize=(10, 6))

ax.plot(fpr, tpr, color="navy")
ax.set_ylabel("Taxa de verdadeiro positivo")
ax.set_xlabel("Taxa de falso positivo")

ax.axvline(fpr[thr_arg_max], color="red", ls=":")
ax.axhline(tpr[thr_arg_max], color="red", ls=":")
ax.plot(thresholds, thresholds, color= "gray", ls=":", lw=0.5)
ax.set_ylim([0., 1.])
ax.set_xlim([0., 1.])
```

Out[ ]:
```
(0.0, 1.0)
```



In [ ]:
```
#area da curva AUC
auc_score = auc(fpr, tpr)
print(f"Area Under Curve (AUC): {auc_score:.2f}")
```

```
Area Under Curve (AUC): 0.80
```

In [ ]:
```
#F1 score
def get_f1_score_list(model, X, y, thresholds):
    list_of_f1 = []
    for threshold in thresholds:
        y_pred = predict(model, X, threshold)
```

```
        f1 = f1_score(y, y_pred)
        list_of_f1.append(f1)
    return list_of_f1

def get_max_f1_score(model, X, y, thresholds):
    list_of_f1 = get_f1_score_list(model, X, y, thresholds)
    f1_max = max(list_of_f1)
    f1_arg_max = np.argmax(list_of_f1)
    threshold_max = thresholds[f1_arg_max]
    return f1_max, threshold_max, f1_arg_max


fpr, tpr, thresholds = roc_curve(y_train, y_hat[:, 1], pos_label=1)
auc_score = auc(fpr, tpr)
f1_max, threshold_max, f1_arg_max =  get_max_f1_score(logreg,
                                                      X_train_scaled,
                                                      y_train,
                                                      thresholds)


print(f"Area Under Curve (AUC): {auc_score:.2f}")
print(f"Maximum F1 : {f1_max:.2f} at {threshold_max:.3f}")
```

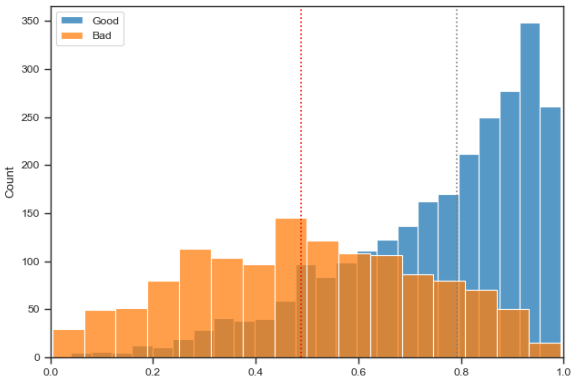
```
Area Under Curve (AUC): 0.80
Maximum F1 : 0.83 at 0.488
```

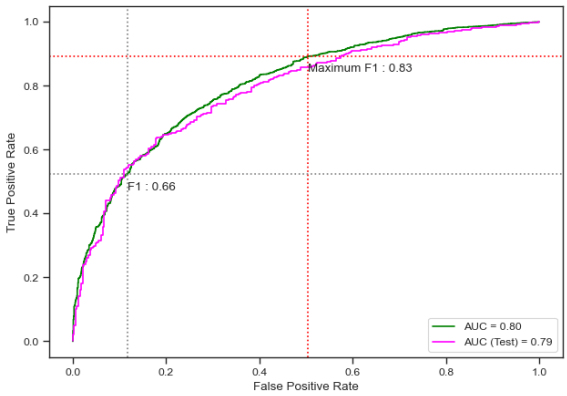
In [ ]:

```
fig, axes = plt.subplots(1, 2, figsize=(16, 5))

threshold_random = 500

# DISTRIBUTION
plot_distributions(logreg, X_train_scaled, y_train, ax=axes[0])
axes[0].axvline(threshold_max, color="red", ls=":")
axes[0].axvline(thresholds[threshold_random], color="gray", ls=":")

# ROC CURVE
RocCurveDisplay(fpr=fpr, tpr=tpr, roc_auc=auc_score).plot(ax=axes[1],color="green"
axes[1].axvline(fpr[f1_arg_max], color="red", ls=":")
axes[1].axhline(tpr[f1_arg_max], color="red", ls=":")
axes[1].annotate(f"Maximum F1 : {f1_max:.2f}", (fpr[f1_arg_max], tpr[f1_arg_max] -


f1_list = get_f1_score_list(logreg, X_train_scaled, y_train, thresholds)


axes[1].axvline(fpr[threshold_random], color="gray", ls=":")
axes[1].axhline(tpr[threshold_random], color="gray", ls=":")

axes[1].annotate(f"F1 : {f1_list[threshold_random]:.2f}",
                 (fpr[threshold_random],
                  tpr[threshold_random] - 0.05))


# ROC CURVE Test
y_hat_test = logreg.predict_proba(X_test_scaled)
fpr_test, tpr_test, thresholds_test = roc_curve(y_test, y_hat_test[:, 1], pos_label
auc_score_test = auc(fpr_test, tpr_test)

RocCurveDisplay(fpr=fpr_test, tpr=tpr_test, roc_auc=auc_score_test).plot(ax=axes[1
                                                                         label=f"AU
                                                                         color="mag
```

Out[ ]:

```
<sklearn.metrics._plot.roc_curve.RocCurveDisplay at 0x2a8e9cee0e0>
```





c)Treine um modelo de árvores de decisão usando um modelo de validação cruzada estratificada com k-folds (k=10) para realizar a classificação. Calcule para a base de teste:

i. a média e desvio da acurácia dos modelos obtidos;

ii. a média e desvio da precisão dos modelos obtidos;

iii. a média e desvio da recall dos modelos obtidos;

iv. a média e desvio do f1-score dos modelos obtidos.

d)Treine um modelo de SVM usando um modelo de validação cruzada estratificada com k-folds (k=10) para realizar a classificação. Calcule para a base de teste:

i. a média e desvio da acurácia dos modelos obtidos;

ii. a média e desvio da precisão dos modelos obtidos;

iii. a média e desvio da recall dos modelos obtidos;

iv. a média e desvio do f1-score dos modelos obtidos.

validacao cruzada

In [ ]:
```
def interpolation(fpr, tpr):
    interp_fpr = np.linspace(0, 1, 100)
    interp_tpr = np.interp(interp_fpr, fpr, tpr)
    interp_tpr[0] = 0.
    return interp_fpr, interp_tpr
```

In [ ]:
```
# Import necessary packages
import pandas as pd
import random
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
from copy import deepcopy as cp
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC

from sklearn.tree import (
    DecisionTreeClassifier,
    plot_tree
)
from sklearn.model_selection import (
```

```
    train_test_split,
    StratifiedKFold)
from sklearn.metrics import (
    accuracy_score,
    precision_score,
    recall_score,
    f1_score,
    classification_report,
    confusion_matrix,
    roc_curve,
    auc,
    RocCurveDisplay
)
sns.set_style("ticks")
sns.set_context("paper")

random_state = 42

X = df_white[vars]
y = df_white['opinion']

X_train_cv, X_test, y_train_cv, y_test = train_test_split(X.values,
                                                          y.values,
                                                          test_size=0.2, # 20 % da
                                                          random_state=42,
                                                          stratify=y)

# Usaremos já na função train a curva ROC média para responder à questão.
def train(X, y, model_klass, model_kwargs = {}):
    cv = StratifiedKFold(n_splits=10) # Esse n_splits é o K, do K folds, que no exe
    f1_score_val_list = []
    f1_score_train_list = []
    model_list = []
    scaler_list = []
    accuracy_score_val_list    = []
    accuracy_score_train_list  = []
    recall_score_val_list      = []
    recall_score_train_list    = []
    precision_score_val_list   = []
    precision_score_train_list = []

    fig, ax = plt.subplots(1, 1, figsize=(8, 8))
    fprs_list = []
    tprs_list = []
    auc_list  = []

    # Validação cruzada só em Training Data
    for fold, (train_idx, val_idx) in enumerate(cv.split(X, y)):
        X_train = X[train_idx, :]
        y_train = y[train_idx]
        X_val = X[val_idx, :]
        y_val = y[val_idx]

        # Escala
        scaler = StandardScaler()
        X_train_scaled = scaler.fit_transform(X_train)
        X_val_scaled = scaler.transform(X_val)

        scaler_list.append(scaler)

        # Treino
        model = model_klass(**model_kwargs)

        model.fit(X_train_scaled, y_train)
```

```
        y_pred = model.predict(X_train_scaled)

        y_pred_val = model.predict(X_val_scaled)
        print(f"======================== FOLD {fold} ========================")
        print(f"Meu resultado para treino de F1-Score é {f1_score(y_train, y_pred)
        print(f"Meu resultado para treino de  Acurácia é de {accuracy_score(y_trai
        print(f"Meu resultado para treino de  Recall é de {recall_score(y_train, y_
        print(f"Meu resultado para treino de  Precision é de {precision_score(y_tr
        f1_score_val_list.append(f1_score(y_val, y_pred_val))
        f1_score_train_list.append(f1_score(y_train, y_pred))
        accuracy_score_val_list.append(accuracy_score(y_val, y_pred_val))
        accuracy_score_train_list.append(accuracy_score(y_train, y_pred))
        recall_score_val_list.append(recall_score(y_val, y_pred_val))
        recall_score_train_list.append(recall_score(y_train, y_pred))
        precision_score_val_list.append(precision_score(y_val, y_pred_val))
        precision_score_train_list.append(precision_score(y_train, y_pred))
        model_list.append(model)
        viz = RocCurveDisplay.from_estimator(
            model,
            X_val_scaled,
            y_val,
            ax = ax,
            alpha=0.3,
            lw=1
        )
        interp_fpr, interp_tpr = interpolation(viz.fpr, viz.tpr)
        fprs_list.append(interp_fpr)
        tprs_list.append(interp_tpr)
        auc_list.append(viz.roc_auc)
    print()
    mean_val = np.mean(f1_score_val_list)
    std_val  = np.std(f1_score_val_list)
    print(f"Meu resultado de F1-Score Médio de treino é {np.mean(f1_score_train_lis
    print(f"Meu resultado de accuracy_score Médio de treino é {np.mean(accuracy_sc
    print(f"Meu resultado de recall_score Médio de treino é {np.mean(recall_score_
    print(f"Meu resultado de precision_score Médio de treino é {np.mean(precision_

    best_model_idx = np.argmax(f1_score_val_list)
    print(f"Meu melhor fold é: {best_model_idx} ")
    best_model = model_list[best_model_idx]

    # Fazer a inferência em Test Data
    best_scaler   = scaler_list[best_model_idx]
    X_test_scaled = best_scaler.transform(X_test)
    y_pred_test   = model.predict(X_test_scaled)

    # Fazer a Curva ROC
    mean_fpr = np.mean(fprs_list, axis=0)
    mean_tpr = np.mean(tprs_list, axis=0)
    mean_auc = np.mean(auc_list)
    std_auc = np.std(auc_list)
    ax.plot(
        mean_fpr,
        mean_tpr,
        color='blue',
        lw=2,
        label=r"Mean ROC (AUC = %.2f $\pm$ %.2f)" %(mean_auc, std_auc)
    )


    ax.plot(np.linspace(0, 1, 100),
            np.linspace(0, 1, 100),
            color='g',
            ls=":",
```

```
            lw=0.5)
    ax.legend()

    print()
    print(f"""
            Meu resultado de F1-Score para o conjunto de teste é : {f1_score(y_tes
            Meu resultado de Accuracy para o conjunto de teste é : {accuracy_score
            Meu resultado de Recall para o conjunto de teste é   : {recall_score(y_
            Meu resultado de Precision para o conjunto de teste é: {precision_scor
    return best_model, mean_val, std_val, best_scaler
```

In [ ]:

```
config = [
    (LogisticRegression, {}),
    (DecisionTreeClassifier, {'min_samples_leaf': 50}),
    (SVC, {'kernel': 'rbf', 'gamma': 2}),

]

#results = []
for model_class, setting in config:
    print(model_class.__name__)
    best_model, mean_val, std_val, best_scaler = train(X_train_cv, y_train_cv, mod
```

```
LogisticRegression
======================== FOLD 0 ========================
Meu resultado para treino de F1-Score é 0.83, Meu resultado para validação de F1-S
core é 0.84
Meu resultado para treino de  Acurácia é de 0.76, Meu resultado para validação de
Acurácia é de 0.77
Meu resultado para treino de  Recall é de 0.89, Meu resultado para validação de Re
call é de 0.89
Meu resultado para treino de  Precision é de 0.78, Meu resultado para validação de
Precision é de 0.79
======================== FOLD 1 ========================
Meu resultado para treino de F1-Score é 0.83, Meu resultado para validação de F1-S
core é 0.83
Meu resultado para treino de  Acurácia é de 0.75, Meu resultado para validação de
Acurácia é de 0.76
Meu resultado para treino de  Recall é de 0.88, Meu resultado para validação de Re
call é de 0.89
Meu resultado para treino de  Precision é de 0.78, Meu resultado para validação de
Precision é de 0.78
======================== FOLD 2 ========================
Meu resultado para treino de F1-Score é 0.82, Meu resultado para validação de F1-S
core é 0.83
Meu resultado para treino de  Acurácia é de 0.75, Meu resultado para validação de
Acurácia é de 0.75
Meu resultado para treino de  Recall é de 0.87, Meu resultado para validação de Re
call é de 0.89
Meu resultado para treino de  Precision é de 0.78, Meu resultado para validação de
Precision é de 0.77
======================== FOLD 3 ========================
Meu resultado para treino de F1-Score é 0.83, Meu resultado para validação de F1-S
core é 0.81
Meu resultado para treino de  Acurácia é de 0.76, Meu resultado para validação de
Acurácia é de 0.73
Meu resultado para treino de  Recall é de 0.88, Meu resultado para validação de Re
call é de 0.86
Meu resultado para treino de  Precision é de 0.79, Meu resultado para validação de
Precision é de 0.76
======================== FOLD 4 ========================
Meu resultado para treino de F1-Score é 0.83, Meu resultado para validação de F1-S
core é 0.78
Meu resultado para treino de  Acurácia é de 0.76, Meu resultado para validação de
Acurácia é de 0.69
Meu resultado para treino de  Recall é de 0.88, Meu resultado para validação de Re
call é de 0.81
Meu resultado para treino de  Precision é de 0.79, Meu resultado para validação de
Precision é de 0.74
======================== FOLD 5 ========================
Meu resultado para treino de F1-Score é 0.83, Meu resultado para validação de F1-S
core é 0.82
Meu resultado para treino de  Acurácia é de 0.76, Meu resultado para validação de
Acurácia é de 0.75
Meu resultado para treino de  Recall é de 0.88, Meu resultado para validação de Re
call é de 0.87
Meu resultado para treino de  Precision é de 0.78, Meu resultado para validação de
Precision é de 0.78
======================== FOLD 6 ========================
Meu resultado para treino de F1-Score é 0.82, Meu resultado para validação de F1-S
core é 0.84
Meu resultado para treino de  Acurácia é de 0.75, Meu resultado para validação de
Acurácia é de 0.78
Meu resultado para treino de  Recall é de 0.88, Meu resultado para validação de Re
call é de 0.86
Meu resultado para treino de  Precision é de 0.78, Meu resultado para validação de
Precision é de 0.82
```

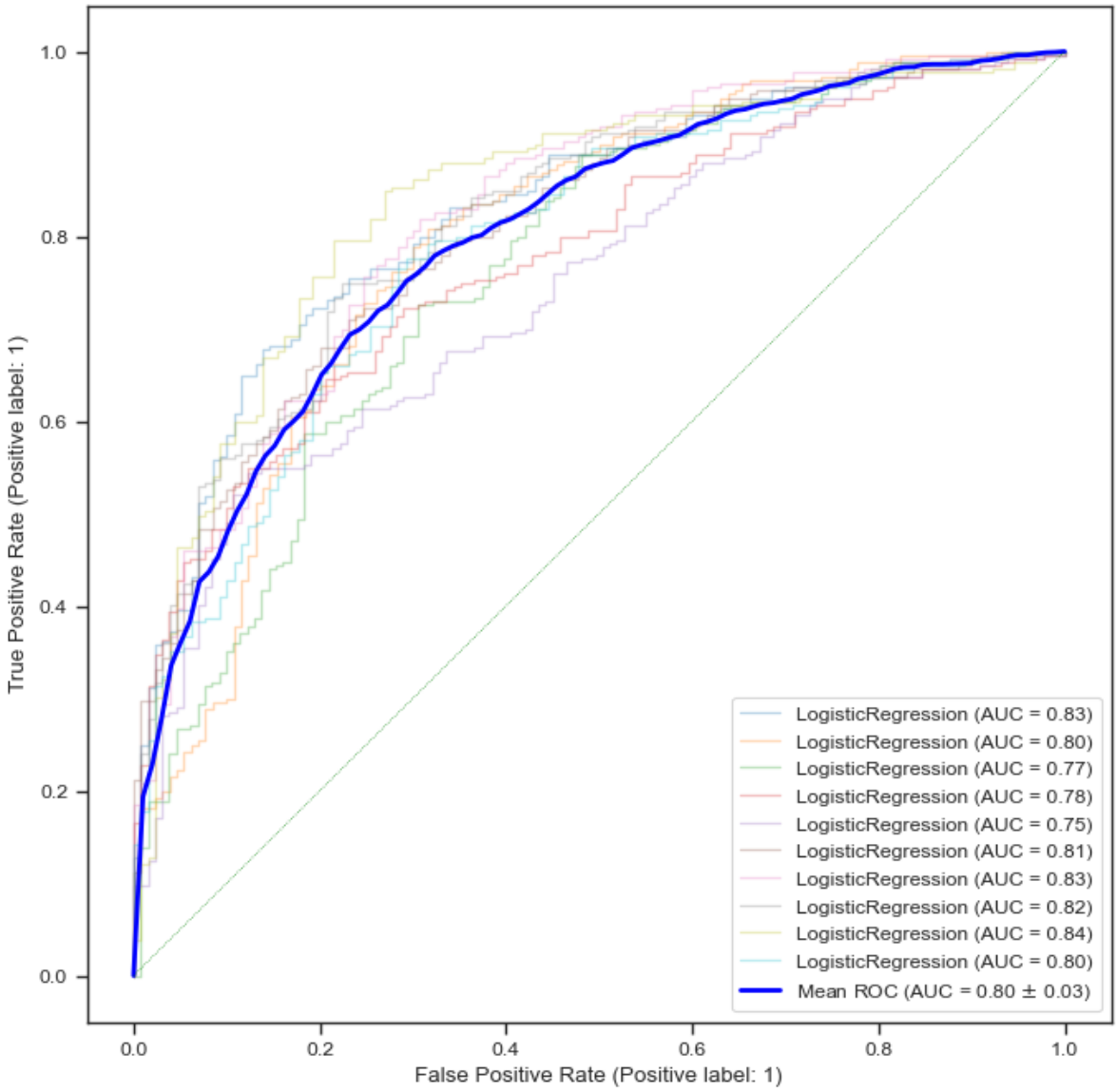
```
======================== FOLD 7 ========================
Meu resultado para treino de F1-Score é 0.83, Meu resultado para validação de F1-S
core é 0.84
Meu resultado para treino de  Acurácia é de 0.76, Meu resultado para validação de
Acurácia é de 0.77
Meu resultado para treino de  Recall é de 0.88, Meu resultado para validação de Re
call é de 0.91
Meu resultado para treino de  Precision é de 0.78, Meu resultado para validação de
Precision é de 0.78
======================== FOLD 8 ========================
Meu resultado para treino de F1-Score é 0.83, Meu resultado para validação de F1-S
core é 0.85
Meu resultado para treino de  Acurácia é de 0.75, Meu resultado para validação de
Acurácia é de 0.79
Meu resultado para treino de  Recall é de 0.88, Meu resultado para validação de Re
call é de 0.91
Meu resultado para treino de  Precision é de 0.78, Meu resultado para validação de
Precision é de 0.8
======================== FOLD 9 ========================
Meu resultado para treino de F1-Score é 0.83, Meu resultado para validação de F1-S
core é 0.83
Meu resultado para treino de  Acurácia é de 0.76, Meu resultado para validação de
Acurácia é de 0.76
Meu resultado para treino de  Recall é de 0.88, Meu resultado para validação de Re
call é de 0.88
Meu resultado para treino de  Precision é de 0.78, Meu resultado para validação de
Precision é de 0.79

Meu resultado de F1-Score Médio de treino é  0.83 +-  0.0023, Meu resultado de F1-
Score Médio de validação é  0.83 +-  0.02
Meu resultado de accuracy_score Médio de treino é  0.76 +-  0.0033, Meu resultado
de accuracy_score Médio de validação é  0.76 +-  0.027
Meu resultado de recall_score Médio de treino é  0.88 +-  0.0036. Meu resultado de
recall_score Médio de validação é  0.88 +-  0.028
Meu resultado de precision_score Médio de treino é  0.78 +-  0.0029, Meu resultado
de precision_score Médio de validação é  0.78 +-  0.02
Meu melhor fold é: 8


            Meu resultado de F1-Score para o conjunto de teste é : 0.81
            Meu resultado de Accuracy para o conjunto de teste é : 0.74
            Meu resultado de Recall para o conjunto de teste é   : 0.87
            Meu resultado de Precision para o conjunto de teste é: 0.77
DecisionTreeClassifier
======================== FOLD 0 ========================
Meu resultado para treino de F1-Score é 0.84, Meu resultado para validação de F1-S
core é 0.83
Meu resultado para treino de  Acurácia é de 0.78, Meu resultado para validação de
Acurácia é de 0.78
Meu resultado para treino de  Recall é de 0.86, Meu resultado para validação de Re
call é de 0.83
Meu resultado para treino de  Precision é de 0.82, Meu resultado para validação de
Precision é de 0.84
======================== FOLD 1 ========================
Meu resultado para treino de F1-Score é 0.84, Meu resultado para validação de F1-S
core é 0.86
Meu resultado para treino de  Acurácia é de 0.79, Meu resultado para validação de
Acurácia é de 0.81
Meu resultado para treino de  Recall é de 0.84, Meu resultado para validação de Re
call é de 0.88
Meu resultado para treino de  Precision é de 0.84, Meu resultado para validação de
Precision é de 0.85
======================== FOLD 2 ========================
Meu resultado para treino de F1-Score é 0.84, Meu resultado para validação de F1-S
```

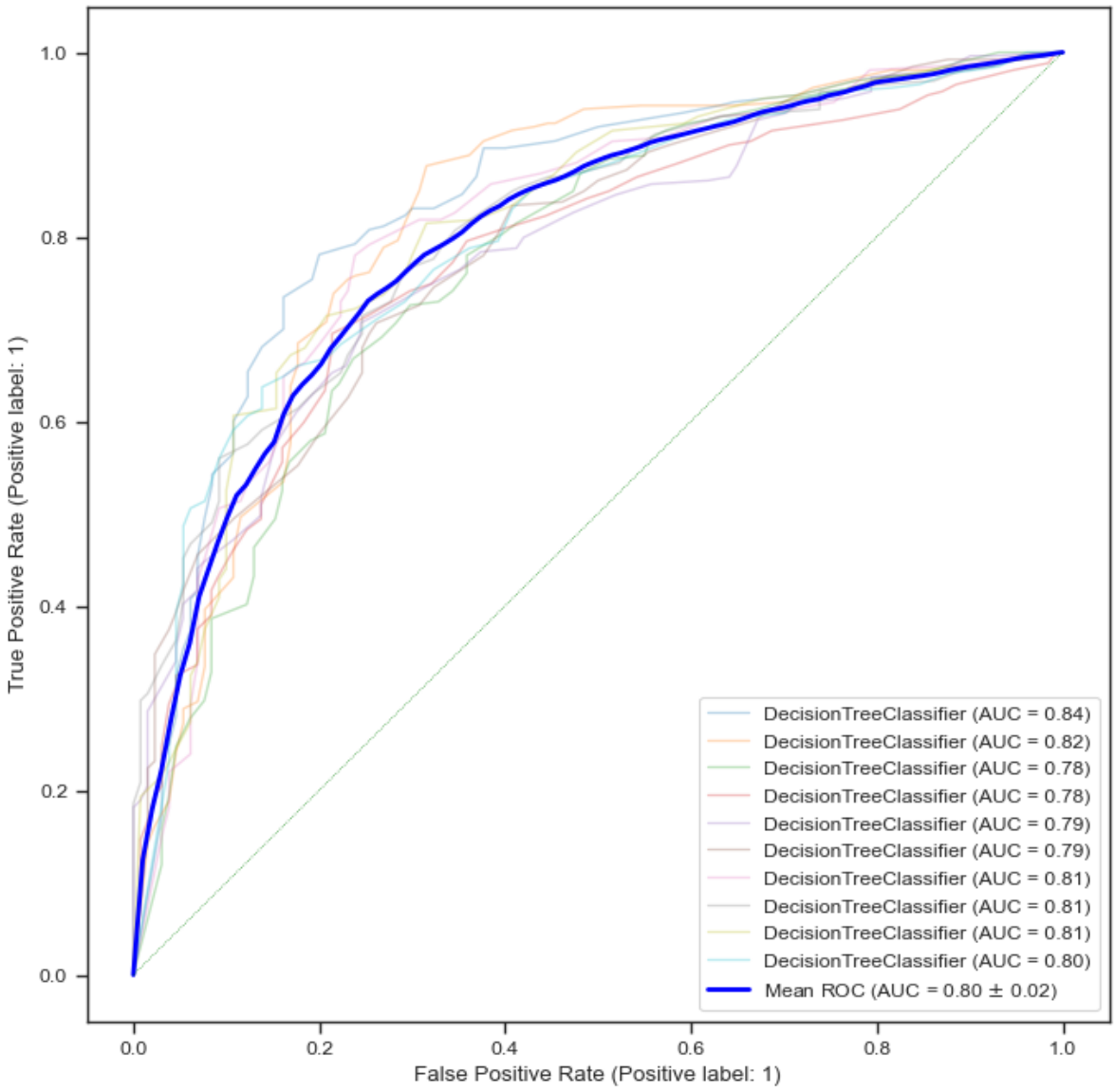
```
core é 0.8
Meu resultado para treino de  Acurácia é de 0.78, Meu resultado para validação de
Acurácia é de 0.74
Meu resultado para treino de  Recall é de 0.85, Meu resultado para validação de Re
call é de 0.8
Meu resultado para treino de  Precision é de 0.83, Meu resultado para validação de
Precision é de 0.8
======================== FOLD 3 ========================
Meu resultado para treino de F1-Score é 0.85, Meu resultado para validação de F1-S
core é 0.8
Meu resultado para treino de  Acurácia é de 0.79, Meu resultado para validação de
Acurácia é de 0.74
Meu resultado para treino de  Recall é de 0.86, Meu resultado para validação de Re
call é de 0.81
Meu resultado para treino de  Precision é de 0.83, Meu resultado para validação de
Precision é de 0.8
======================== FOLD 4 ========================
Meu resultado para treino de F1-Score é 0.84, Meu resultado para validação de F1-S
core é 0.79
Meu resultado para treino de  Acurácia é de 0.79, Meu resultado para validação de
Acurácia é de 0.73
Meu resultado para treino de  Recall é de 0.86, Meu resultado para validação de Re
call é de 0.8
Meu resultado para treino de  Precision é de 0.83, Meu resultado para validação de
Precision é de 0.79
======================== FOLD 5 ========================
Meu resultado para treino de F1-Score é 0.83, Meu resultado para validação de F1-S
core é 0.82
Meu resultado para treino de  Acurácia é de 0.78, Meu resultado para validação de
Acurácia é de 0.75
Meu resultado para treino de  Recall é de 0.82, Meu resultado para validação de Re
call é de 0.83
Meu resultado para treino de  Precision é de 0.85, Meu resultado para validação de
Precision é de 0.8
======================== FOLD 6 ========================
Meu resultado para treino de F1-Score é 0.84, Meu resultado para validação de F1-S
core é 0.82
Meu resultado para treino de  Acurácia é de 0.79, Meu resultado para validação de
Acurácia é de 0.78
Meu resultado para treino de  Recall é de 0.83, Meu resultado para validação de Re
call é de 0.79
Meu resultado para treino de  Precision é de 0.85, Meu resultado para validação de
Precision é de 0.86
======================== FOLD 7 ========================
Meu resultado para treino de F1-Score é 0.84, Meu resultado para validação de F1-S
core é 0.83
Meu resultado para treino de  Acurácia é de 0.79, Meu resultado para validação de
Acurácia é de 0.76
Meu resultado para treino de  Recall é de 0.85, Meu resultado para validação de Re
call é de 0.86
Meu resultado para treino de  Precision é de 0.83, Meu resultado para validação de
Precision é de 0.79
======================== FOLD 8 ========================
Meu resultado para treino de F1-Score é 0.84, Meu resultado para validação de F1-S
core é 0.82
Meu resultado para treino de  Acurácia é de 0.78, Meu resultado para validação de
Acurácia é de 0.76
Meu resultado para treino de  Recall é de 0.85, Meu resultado para validação de Re
call é de 0.82
Meu resultado para treino de  Precision é de 0.83, Meu resultado para validação de
Precision é de 0.82
======================== FOLD 9 ========================
Meu resultado para treino de F1-Score é 0.84, Meu resultado para validação de F1-S
core é 0.8
```

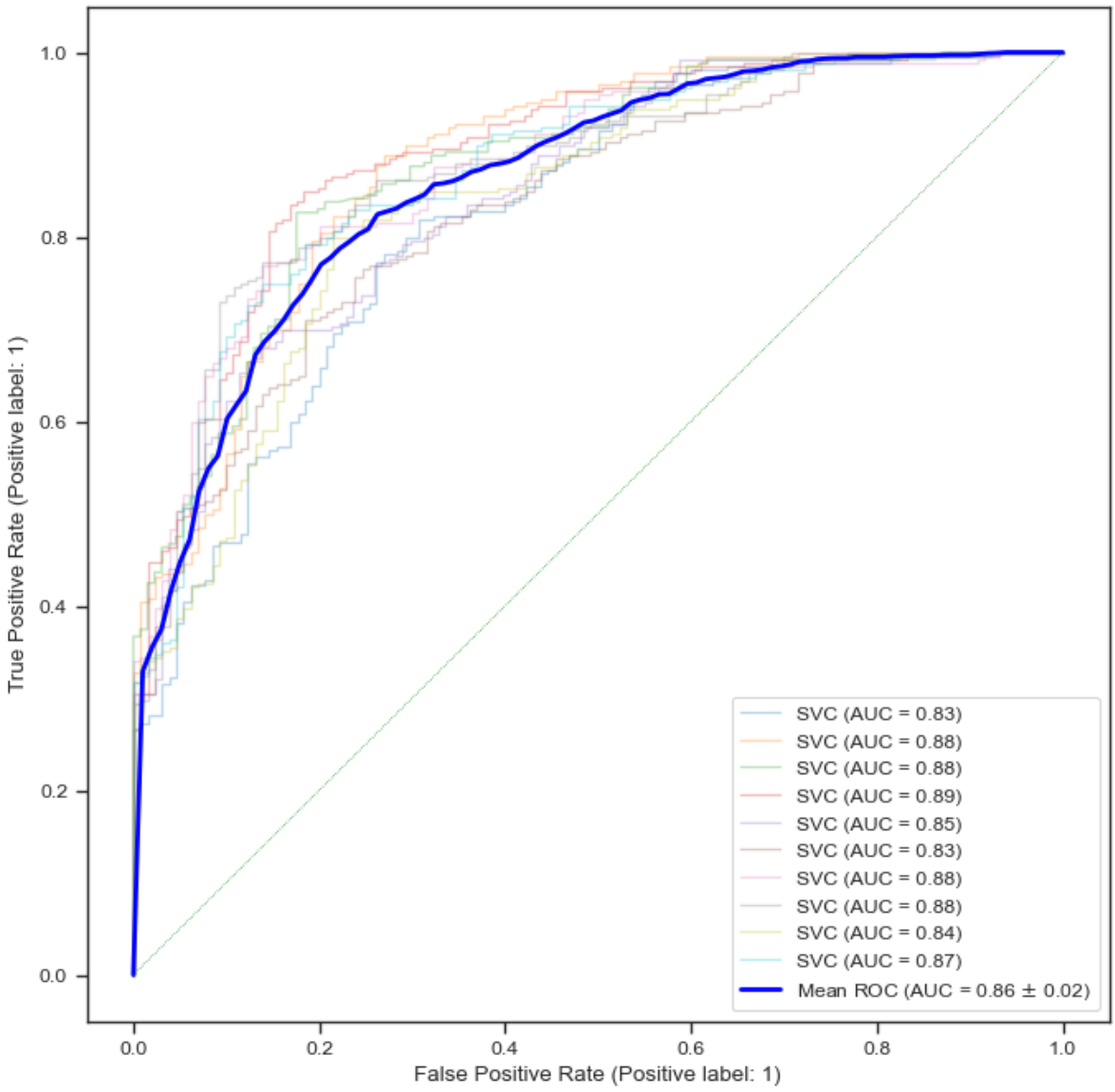
```
Meu resultado para treino de  Acurácia é de 0.78, Meu resultado para validação de
Acurácia é de 0.73
Meu resultado para treino de  Recall é de 0.84, Meu resultado para validação de Re
call é de 0.8
Meu resultado para treino de  Precision é de 0.83, Meu resultado para validação de
Precision é de 0.8

Meu resultado de F1-Score Médio de treino é  0.84 +-  0.0036, Meu resultado de F1-
Score Médio de validação é  0.82 +-  0.019
Meu resultado de accuracy_score Médio de treino é  0.78 +-  0.0038, Meu resultado
de accuracy_score Médio de validação é  0.76 +-  0.025
Meu resultado de recall_score Médio de treino é  0.85 +-  0.013. Meu resultado de
recall_score Médio de validação é  0.82 +-  0.028
Meu resultado de precision_score Médio de treino é  0.83 +-  0.0083, Meu resultado
de precision_score Médio de validação é  0.81 +-  0.024
Meu melhor fold é: 1


            Meu resultado de F1-Score para o conjunto de teste é : 0.79
            Meu resultado de Accuracy para o conjunto de teste é : 0.72
            Meu resultado de Recall para o conjunto de teste é   : 0.81
            Meu resultado de Precision para o conjunto de teste é: 0.78
SVC
========================= FOLD 0 =========================
Meu resultado para treino de F1-Score é 0.99, Meu resultado para validação de F1-S
core é 0.86
Meu resultado para treino de  Acurácia é de 0.99, Meu resultado para validação de
Acurácia é de 0.78
Meu resultado para treino de  Recall é de 1.0, Meu resultado para validação de Rec
all é de 0.98
Meu resultado para treino de  Precision é de 0.99, Meu resultado para validação de
Precision é de 0.77
========================= FOLD 1 =========================
Meu resultado para treino de F1-Score é 0.99, Meu resultado para validação de F1-S
core é 0.86
Meu resultado para treino de  Acurácia é de 0.99, Meu resultado para validação de
Acurácia é de 0.79
Meu resultado para treino de  Recall é de 1.0, Meu resultado para validação de Rec
all é de 0.98
Meu resultado para treino de  Precision é de 0.99, Meu resultado para validação de
Precision é de 0.77
========================= FOLD 2 =========================
Meu resultado para treino de F1-Score é 0.99, Meu resultado para validação de F1-S
core é 0.86
Meu resultado para treino de  Acurácia é de 0.99, Meu resultado para validação de
Acurácia é de 0.79
Meu resultado para treino de  Recall é de 1.0, Meu resultado para validação de Rec
all é de 0.98
Meu resultado para treino de  Precision é de 0.99, Meu resultado para validação de
Precision é de 0.77
========================= FOLD 3 =========================
Meu resultado para treino de F1-Score é 0.99, Meu resultado para validação de F1-S
core é 0.86
Meu resultado para treino de  Acurácia é de 0.99, Meu resultado para validação de
Acurácia é de 0.8
Meu resultado para treino de  Recall é de 1.0, Meu resultado para validação de Rec
all é de 0.97
Meu resultado para treino de  Precision é de 0.99, Meu resultado para validação de
Precision é de 0.78
========================= FOLD 4 =========================
Meu resultado para treino de F1-Score é 0.99, Meu resultado para validação de F1-S
core é 0.86
Meu resultado para treino de  Acurácia é de 0.99, Meu resultado para validação de
Acurácia é de 0.79
```

```
Meu resultado para treino de  Recall é de 1.0, Meu resultado para validação de Rec
all é de 0.99
Meu resultado para treino de  Precision é de 0.99, Meu resultado para validação de
Precision é de 0.76
======================== FOLD 5 ========================
Meu resultado para treino de F1-Score é 0.99, Meu resultado para validação de F1-S
core é 0.84
Meu resultado para treino de  Acurácia é de 0.99, Meu resultado para validação de
Acurácia é de 0.75
Meu resultado para treino de  Recall é de 1.0, Meu resultado para validação de Rec
all é de 0.99
Meu resultado para treino de  Precision é de 0.99, Meu resultado para validação de
Precision é de 0.73
======================== FOLD 6 ========================
Meu resultado para treino de F1-Score é 0.99, Meu resultado para validação de F1-S
core é 0.85
Meu resultado para treino de  Acurácia é de 0.99, Meu resultado para validação de
Acurácia é de 0.77
Meu resultado para treino de  Recall é de 1.0, Meu resultado para validação de Rec
all é de 0.97
Meu resultado para treino de  Precision é de 0.99, Meu resultado para validação de
Precision é de 0.76
======================== FOLD 7 ========================
Meu resultado para treino de F1-Score é 0.99, Meu resultado para validação de F1-S
core é 0.84
Meu resultado para treino de  Acurácia é de 0.99, Meu resultado para validação de
Acurácia é de 0.76
Meu resultado para treino de  Recall é de 1.0, Meu resultado para validação de Rec
all é de 0.98
Meu resultado para treino de  Precision é de 0.99, Meu resultado para validação de
Precision é de 0.74
======================== FOLD 8 ========================
Meu resultado para treino de F1-Score é 0.99, Meu resultado para validação de F1-S
core é 0.84
Meu resultado para treino de  Acurácia é de 0.99, Meu resultado para validação de
Acurácia é de 0.76
Meu resultado para treino de  Recall é de 1.0, Meu resultado para validação de Rec
all é de 0.97
Meu resultado para treino de  Precision é de 0.99, Meu resultado para validação de
Precision é de 0.75
======================== FOLD 9 ========================
Meu resultado para treino de F1-Score é 0.99, Meu resultado para validação de F1-S
core é 0.85
Meu resultado para treino de  Acurácia é de 0.99, Meu resultado para validação de
Acurácia é de 0.77
Meu resultado para treino de  Recall é de 1.0, Meu resultado para validação de Rec
all é de 0.97
Meu resultado para treino de  Precision é de 0.99, Meu resultado para validação de
Precision é de 0.75

Meu resultado de F1-Score Médio de treino é  0.99 +-  0.00041, Meu resultado de F1
-Score Médio de validação é  0.85 +-  0.01
Meu resultado de accuracy_score Médio de treino é  0.99 +-  0.00055, Meu resultado
de accuracy_score Médio de validação é  0.78 +-  0.018
Meu resultado de recall_score Médio de treino é  1.0 +-  0.00051. Meu resultado de
recall_score Médio de validação é  0.98 +-  0.0094
Meu resultado de precision_score Médio de treino é  0.99 +-  0.00077, Meu resultad
o de precision_score Médio de validação é  0.76 +-  0.015
Meu melhor fold é: 1


            Meu resultado de F1-Score para o conjunto de teste é : 0.86
            Meu resultado de Accuracy para o conjunto de teste é : 0.78
```

```
Meu resultado de Recall para o conjunto de teste é   : 0.99
Meu resultado de Precision para o conjunto de teste é: 0.76
```




True Positive Rate (Positive label: 1)
False Positive Rate (Positive label: 1)
0.0
0.2
0.4
0.6
0.8
1.0
DecisionTreeClassifier (AUC = 0.84)
DecisionTreeClassifier (AUC = 0.82)
DecisionTreeClassifier (AUC = 0.78)
DecisionTreeClassifier (AUC = 0.78)
DecisionTreeClassifier (AUC = 0.79)
DecisionTreeClassifier (AUC = 0.79)
DecisionTreeClassifier (AUC = 0.81)
DecisionTreeClassifier (AUC = 0.81)
DecisionTreeClassifier (AUC = 0.81)
DecisionTreeClassifier (AUC = 0.80)
Mean ROC (AUC = 0.80 ± 0.02)



5-Em relação à questão anterior, qual o modelo deveria ser escolhido para uma eventual operação. Responda essa questão mostrando a comparação de todos os modelos, usando um gráfico mostrando a curva ROC média para cada um dos gráficos e justifique.

A mellhor AUC foi a CVM, por isso teve o melhor modelo

6-Com a escolha do melhor modelo, use os dados de vinho tinto, presentes na base original e faça a inferência (não é para treinar novamente!!!) para saber quantos vinhos são bons ou ruins. Utilize o mesmo critério utilizado com os vinhos brancos, para comparar o desempenho do modelo. Ele funciona da mesma forma para essa nova base? Justifique.

In [ ]:
```
df_red = df[df['type'] == 'red'].copy()
df_red.dropna(inplace=True)
```

In [ ]:
```
var = [
   'fixed acidity',
   'volatile acidity',
   'citric acid',
   'residual sugar',
   'chlorides',
```

```
   'free sulfur dioxide',
   'total sulfur dioxide',
   'density',
   'pH',
   'sulphates',
   'alcohol',
]

X_real = df_red[var]
y_real = df_red['opinion']
X_real_scaled = best_scaler.transform(X_real)
y_pred_real   = best_model.predict(X_real_scaled)

print(f"A acurácia real é {100 * accuracy_score(y_real, y_pred_real):.2f} %")
print(f"A sensibilidade real é {100 *  recall_score(y_real, y_pred_real):.2f} %")
print(f"A precisão real é {100*  precision_score(y_real, y_pred_real):.2f} %")
print(f"O F1 Score real =  {f1_score(y_real, y_pred_real):.2f} ")
print(classification_report(y_real, y_pred_real))
```

```
c:\Users\user\miniconda3\envs\Bootcamp\lib\site-packages\sklearn\base.py:432: User
Warning: X has feature names, but StandardScaler was fitted without feature names
  warnings.warn(
A acurácia real é 53.48 %
A sensibilidade real é 100.00 %
A precisão real é 53.45 %
O F1 Score real =  0.70
              precision    recall  f1-score   support

           0       1.00      0.00      0.00       742
           1       0.53      1.00      0.70       851

    accuracy                           0.53      1593
   macro avg       0.77      0.50      0.35      1593
weighted avg       0.75      0.53      0.37      1593
```

In [ ]:
```
print('Vinhos bons brancos: ')
print(df_white.opinion.sum())
print('---------------------')
print('Vinhos bons tintos: ')
print(df_red.opinion.sum())
print('---------------------')
print('Soma: ')
print(df_red.opinion.sum() + df_white.opinion.sum())
```

```
Vinhos bons brancos:
3240
---------------------
Vinhos bons tintos:
851
---------------------
Soma:
4091
```

In [ ]:
```
print(df.opinion.sum()) #resultado incluindo outliers
```

```
4113
```