```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
```

```
dados = pd.read_csv('test.csv')
```

```
print(dados.head())
```

```
    age  anaemia  creatinine_phosphokinase  diabetes  ejection_fraction  \
0  75.0        0                       582         0                 20   
1  55.0        0                      7861         0                 38   
2  65.0        0                       146         0                 20   
3  50.0        1                       111         0                 20   
4  65.0        1                       160         1                 20   

   high_blood_pressure  platelets  serum_creatinine  serum_sodium  sex  \
0                    1  265000.00               1.9           130    1   
1                    0  263358.03               1.1           136    1   
2                    0  162000.00               1.3           129    1   
3                    0  210000.00               1.9           137    1   
4                    0  327000.00               2.7           116    0   

   smoking  time  DEATH_EVENT  
0        0     4            1  
1        0     6            1  
2        1     7            1  
3        0     7            1  
4        0     8            1  
```

```
X = dados.drop(columns=['DEATH_EVENT'])
y = dados['DEATH_EVENT']
```

```
from sklearn.metrics import confusion_matrix
y_true = [1, 0, 1, 0, 1, 0, 0, 1]
y_pred = [1, 1, 0, 0, 1, 1, 0, 1]
matriz_confusao = confusion_matrix(y_true, y_pred)
print(matriz_confusao)
```

```
[[2 2]
 [1 3]]
```

```
from sklearn.model_selection import train_test_split

X = dados.drop(columns=['DEATH_EVENT'])
y = dados['DEATH_EVENT']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

modelo.fit(X_train, y_train)
```

```
▾ RandomForestClassifier
RandomForestClassifier()
```

```
from sklearn.metrics import precision_score, recall_score, f1_score
```

```
y_pred = modelo.predict(X_test)
```

```
precisao = precision_score(y_test, y_pred)
recall = recall_score(y_test, y_pred)
f1 = f1_score(y_test, y_pred)
```

```
print(f'Precisão: {precisao}, Recall: {recall}, F1-score: {f1}')
```

```
Precisão: 0.75, Recall: 0.48, F1-score: 0.5853658536585366
```

```
from sklearn.model_selection import GridSearchCV
from sklearn.ensemble import RandomForestClassifier
```

```
parametros = {'n_estimators': [50, 100, 150], 'max_depth': [None, 10, 20]}
```

```
modelo = RandomForestClassifier()
```

```
grid_search = GridSearchCV(modelo, parametros, cv=5)
```

```
grid_search.fit(X_train, y_train)
```

```
▸           GridSearchCV
▸ estimator: RandomForestClassifier
      ▸ RandomForestClassifier
```

```
print("Melhores hiperparâmetros:", grid_search.best_params_)
```

```
Melhores hiperparâmetros: {'max_depth': 20, 'n_estimators': 150}
```

```
y_pred = grid_search.predict(X_test)
precisao = precision_score(y_test, y_pred)
recall = recall_score(y_test, y_pred)
f1 = f1_score(y_test, y_pred)
print(f'Precisão: {precisao}, Recall: {recall}, F1-score: {f1}')
```

```
Precisão: 0.9285714285714286, Recall: 0.52, F1-score: 0.6666666666666666
```

```
from sklearn.model_selection import cross_val_score
```

```
pontuacoes = cross_val_score(modelo, X, y, cv=5)
```

```
print("Pontuações de validação cruzada:", pontuacoes)
```

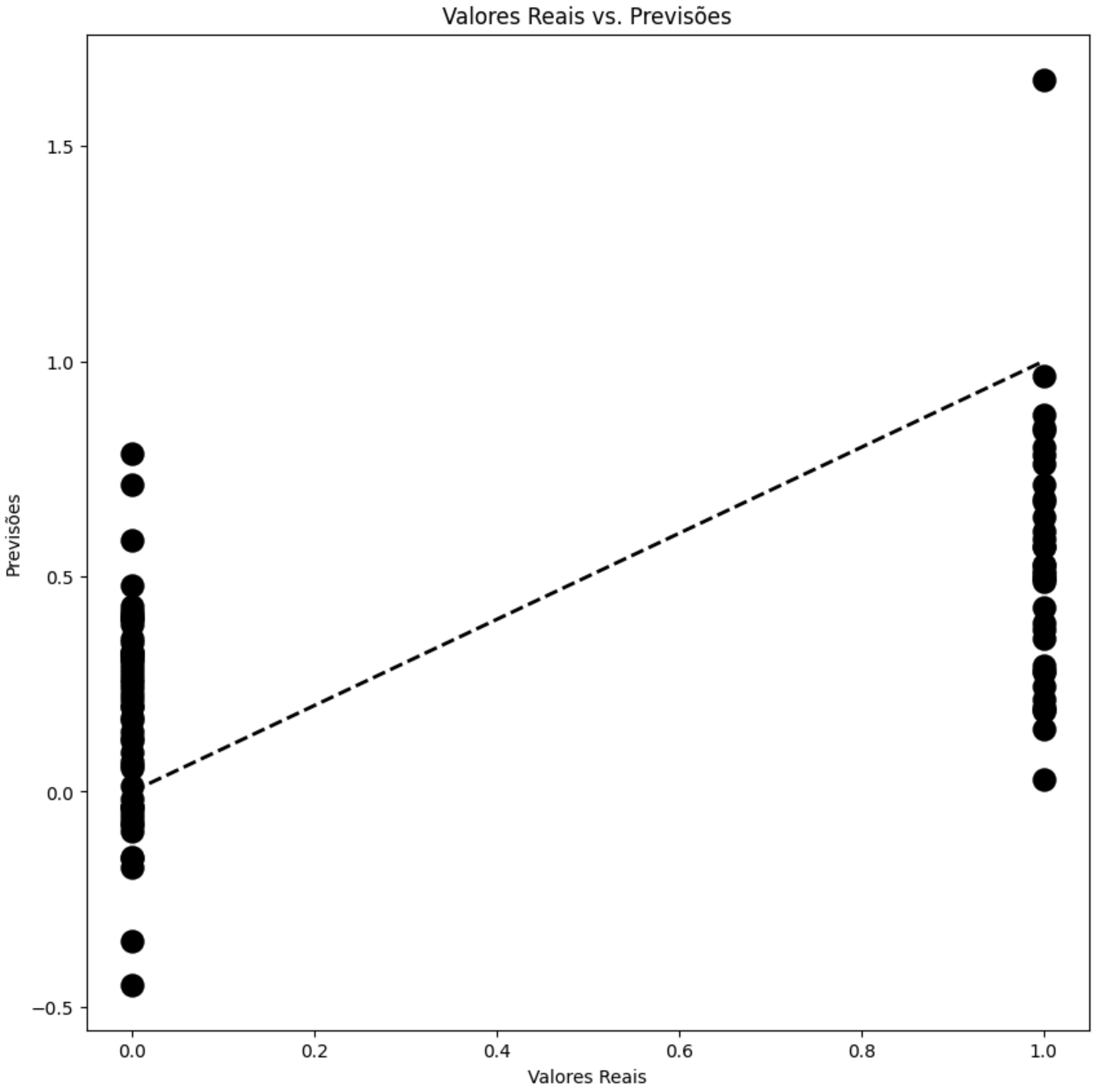
```
Pontuações de validação cruzada: [0.41666667 0.81666667 0.88333333 0.7        0.71186441]
```

```
print("Média das pontuações:", pontuacoes.mean())
print("Desvio padrão das pontuações:", pontuacoes.std())
```

```
Média das pontuações: 0.7057062146892654
Desvio padrão das pontuações: 0.15967339147010895
```

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
import matplotlib.pyplot as plt

X_train, X_val, y_train, y_val = train_test_split(X, y, test_size=0.3, random_state=42)

model = LinearRegression()

model.fit(X_train, y_train)

predictions = model.predict(X_val)

mse = mean_squared_error(y_val, predictions)
r2 = r2_score(y_val, predictions)

print("Erro Quadrático Médio (MSE):", mse)
print("R-quadrado (R²):", r2)

plt.figure(figsize=(10, 10,))
plt.scatter(y_val, predictions, color='black', alpha=1, marker='o', s=150)
plt.plot([y_val.min(), y_val.max()], [y_val.min(), y_val.max()], 'k--', lw=2)
plt.xlabel('Valores Reais')
plt.ylabel('Previsões')
plt.title('Valores Reais vs. Previsões')
plt.show()
```

```
Erro Quadrático Médio (MSE): 0.18028241159969233
R-quadrado (R²): 0.2553352708018828
```


Valores Reais vs. Previsões
Previsões
Valores Reais