Einleitung

Der vorliegende Report dient zur Beschreibung des Projekts im Rahmen der Data Exploration Vorlesung. Die Abgabe umfasst ein GitHub Repositpory mit dem erstellten Code und dem Report in Form eines Jupyter Notebooks. Ziel dieses Projekts ist es anhand des, im Folgenden beschriebenen Datensatzes, eine exlorative Datenanalyse zu betreiben und ein Machine Learning Modell zu entwickeln, das zuverlässige Ergebnisse liefert.

Insalliere Requirements

In [ ]:
```
! pip install -r requirements.txt
```

```
Requirement already satisfied: appnope==0.1.4 in ./venv/lib/python3.9/site
-packages (from -r requirements.txt (line 1)) (0.1.4)
Requirement already satisfied: asttokens==2.4.1 in ./venv/lib/python3.9/si
te-packages (from -r requirements.txt (line 2)) (2.4.1)
Requirement already satisfied: comm==0.2.1 in ./venv/lib/python3.9/site-pa
ckages (from -r requirements.txt (line 3)) (0.2.1)
Requirement already satisfied: contourpy==1.2.0 in ./venv/lib/python3.9/si
te-packages (from -r requirements.txt (line 4)) (1.2.0)
Requirement already satisfied: cycler==0.12.1 in ./venv/lib/python3.9/site
-packages (from -r requirements.txt (line 5)) (0.12.1)
Requirement already satisfied: debugpy==1.8.1 in ./venv/lib/python3.9/site
-packages (from -r requirements.txt (line 6)) (1.8.1)
Requirement already satisfied: decorator==5.1.1 in ./venv/lib/python3.9/si
te-packages (from -r requirements.txt (line 7)) (5.1.1)
Requirement already satisfied: exceptiongroup==1.2.0 in ./venv/lib/python
3.9/site-packages (from -r requirements.txt (line 8)) (1.2.0)
Requirement already satisfied: executing==2.0.1 in ./venv/lib/python3.9/si
te-packages (from -r requirements.txt (line 9)) (2.0.1)
Requirement already satisfied: fonttools==4.49.0 in ./venv/lib/python3.9/s
ite-packages (from -r requirements.txt (line 10)) (4.49.0)
Requirement already satisfied: importlib-metadata==7.0.1 in ./venv/lib/pyt
hon3.9/site-packages (from -r requirements.txt (line 11)) (7.0.1)
Requirement already satisfied: importlib-resources==6.1.1 in ./venv/lib/py
thon3.9/site-packages (from -r requirements.txt (line 12)) (6.1.1)
Requirement already satisfied: ipykernel==6.29.2 in ./venv/lib/python3.9/s
ite-packages (from -r requirements.txt (line 13)) (6.29.2)
Requirement already satisfied: ipython==8.18.1 in ./venv/lib/python3.9/sit
e-packages (from -r requirements.txt (line 14)) (8.18.1)
Requirement already satisfied: jedi==0.19.1 in ./venv/lib/python3.9/site-p
ackages (from -r requirements.txt (line 15)) (0.19.1)
Requirement already satisfied: joblib==1.3.2 in ./venv/lib/python3.9/site-
packages (from -r requirements.txt (line 16)) (1.3.2)
Requirement already satisfied: jupyter_client==8.6.0 in ./venv/lib/python
3.9/site-packages (from -r requirements.txt (line 17)) (8.6.0)
Requirement already satisfied: jupyter_core==5.7.1 in ./venv/lib/python3.
9/site-packages (from -r requirements.txt (line 18)) (5.7.1)
Requirement already satisfied: kiwisolver==1.4.5 in ./venv/lib/python3.9/s
ite-packages (from -r requirements.txt (line 19)) (1.4.5)
Requirement already satisfied: matplotlib==3.8.3 in ./venv/lib/python3.9/s
ite-packages (from -r requirements.txt (line 20)) (3.8.3)
Requirement already satisfied: matplotlib-inline==0.1.6 in ./venv/lib/pyth
on3.9/site-packages (from -r requirements.txt (line 21)) (0.1.6)
Requirement already satisfied: nest-asyncio==1.6.0 in ./venv/lib/python3.
9/site-packages (from -r requirements.txt (line 22)) (1.6.0)
Requirement already satisfied: numpy==1.26.4 in ./venv/lib/python3.9/site-
packages (from -r requirements.txt (line 23)) (1.26.4)
Requirement already satisfied: packaging==23.2 in ./venv/lib/python3.9/sit
e-packages (from -r requirements.txt (line 24)) (23.2)
Requirement already satisfied: pandas==2.2.0 in ./venv/lib/python3.9/site-
packages (from -r requirements.txt (line 25)) (2.2.0)
Requirement already satisfied: parso==0.8.3 in ./venv/lib/python3.9/site-p
ackages (from -r requirements.txt (line 26)) (0.8.3)
Requirement already satisfied: pexpect==4.9.0 in ./venv/lib/python3.9/site
-packages (from -r requirements.txt (line 27)) (4.9.0)
Requirement already satisfied: pillow==10.2.0 in ./venv/lib/python3.9/site
-packages (from -r requirements.txt (line 28)) (10.2.0)
Requirement already satisfied: platformdirs==4.2.0 in ./venv/lib/python3.
9/site-packages (from -r requirements.txt (line 29)) (4.2.0)
Requirement already satisfied: prompt-toolkit==3.0.43 in ./venv/lib/python
3.9/site-packages (from -r requirements.txt (line 30)) (3.0.43)
```

```
Requirement already satisfied: psutil==5.9.8 in ./venv/lib/python3.9/site-
packages (from -r requirements.txt (line 31)) (5.9.8)
Requirement already satisfied: ptyprocess==0.7.0 in ./venv/lib/python3.9/s
ite-packages (from -r requirements.txt (line 32)) (0.7.0)
Requirement already satisfied: pure-eval==0.2.2 in ./venv/lib/python3.9/si
te-packages (from -r requirements.txt (line 33)) (0.2.2)
Requirement already satisfied: Pygments==2.17.2 in ./venv/lib/python3.9/si
te-packages (from -r requirements.txt (line 34)) (2.17.2)
Requirement already satisfied: pyparsing==3.1.1 in ./venv/lib/python3.9/si
te-packages (from -r requirements.txt (line 35)) (3.1.1)
Requirement already satisfied: python-dateutil==2.8.2 in ./venv/lib/python
3.9/site-packages (from -r requirements.txt (line 36)) (2.8.2)
Requirement already satisfied: pytz==2024.1 in ./venv/lib/python3.9/site-p
ackages (from -r requirements.txt (line 37)) (2024.1)
Requirement already satisfied: pyzmq==25.1.2 in ./venv/lib/python3.9/site-
packages (from -r requirements.txt (line 38)) (25.1.2)
Requirement already satisfied: scikit-learn==1.4.1.post1 in ./venv/lib/pyt
hon3.9/site-packages (from -r requirements.txt (line 39)) (1.4.1.post1)
Requirement already satisfied: scipy==1.12.0 in ./venv/lib/python3.9/site-
packages (from -r requirements.txt (line 40)) (1.12.0)
Requirement already satisfied: seaborn==0.13.2 in ./venv/lib/python3.9/sit
e-packages (from -r requirements.txt (line 41)) (0.13.2)
Requirement already satisfied: six==1.16.0 in ./venv/lib/python3.9/site-pa
ckages (from -r requirements.txt (line 42)) (1.16.0)
Requirement already satisfied: stack-data==0.6.3 in ./venv/lib/python3.9/s
ite-packages (from -r requirements.txt (line 43)) (0.6.3)
Requirement already satisfied: threadpoolctl==3.3.0 in ./venv/lib/python3.
9/site-packages (from -r requirements.txt (line 44)) (3.3.0)
Requirement already satisfied: tornado==6.4 in ./venv/lib/python3.9/site-p
ackages (from -r requirements.txt (line 45)) (6.4)
Requirement already satisfied: traitlets==5.14.1 in ./venv/lib/python3.9/s
ite-packages (from -r requirements.txt (line 46)) (5.14.1)
Requirement already satisfied: typing_extensions==4.9.0 in ./venv/lib/pyth
on3.9/site-packages (from -r requirements.txt (line 47)) (4.9.0)
Requirement already satisfied: tzdata==2024.1 in ./venv/lib/python3.9/site
-packages (from -r requirements.txt (line 48)) (2024.1)
Requirement already satisfied: wcwidth==0.2.13 in ./venv/lib/python3.9/sit
e-packages (from -r requirements.txt (line 49)) (0.2.13)
Requirement already satisfied: zipp==3.17.0 in ./venv/lib/python3.9/site-p
ackages (from -r requirements.txt (line 50)) (3.17.0)
```

In [ ]:

```python
# importing all required libraries
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np

# machine learning libraries
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix, classificat
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import roc_auc_score
from sklearn.metrics import roc_curve
from sklearn.svm import SVC
from sklearn.model_selection import train_test_split

%matplotlib inline
```

Data Quality Check & Data Characterization

Die verwendeten Daten

Bei den verwendeten Daten handelt es sich um einen Kaggle Datensatz (https://www.kaggle.com/datasets/fedesoriano/heart-failure-prediction/data; letzter Abruf: 04.04.2024). Der Datensatz enthält Informationen von 918 Patienten und umfasst zwölf verschiedene Merkmale, darunter de- mografische Angaben wie Alter und Geschlecht, klinische Messungen wie Ruheblutdruck und maximale Herzfrequenz, sowie Informationen zu Symptomen wie Brustschmerzen und zuvor dia- gnostizierten Herzkrankheiten. Die Daten widerspiegeln auch medizinischen Tests wie Ruhe- Elektrokardiogrammen und Belastungsuntersuchungen.

In [ ]:
```
# defining path in which the data is stored
data = "data/heart.csv"
```

In [ ]:
```
# reading the data
df = pd.read_csv(data)
df
```

Out[ ]:

	Age	Sex	ChestPainType	RestingBP	Cholesterol	FastingBS	RestingECG	M
0	40	M	ATA	140	289	0	Normal	
1	49	F	NAP	160	180	0	Normal	
2	37	M	ATA	130	283	0	ST	
3	48	F	ASY	138	214	0	Normal	
4	54	M	NAP	150	195	0	Normal	
...	...	...	...	...	...	...	...	
913	45	M	TA	110	264	0	Normal	
914	68	M	ASY	144	193	1	Normal	
915	57	M	ASY	130	131	0	Normal	
916	57	F	ATA	130	236	0	LVH	
917	38	M	NAP	138	175	0	Normal	

918 rows × 12 columns

Beschreibung der Attribute:

- Age: Alter des Patienten [Jahre]

- Sex: Geschlecht des Patienten [M: Männlich, F: Weiblich]
- ChestPainType: Brustschmerztyp [TA: Typische Angina, ATA: Atypische Angina, NAP: Nicht-Anginaler Schmerz, ASY: Asymptomatisch]
- RestingBP: Ruheblutdruck [mm Hg]
- Cholesterol: Serumcholesterin [mm/dl]
- FastingBS: Nüchternblutzucker [1: Wenn Nüchternblutzucker > 120 mg/dl, 0: Ansonsten]
- RestingECG: Ruheelektrokardiogrammergebnisse [Normal: Normal, ST: Mit ST-T-Wellen-Abnormalitäten (T-Wellen-Inversionen und/oder ST-Hebungen oder -Senkungen von > 0,05 mV), LVH: Zeigt wahrscheinliche oder definitive linksventrikuläre Hypertrophie nach Estes-Kriterien]
- MaxHR: Maximale erreichte Herzfrequenz [Numerischer Wert zwischen 60 und 202]
- ExerciseAngina: Belastungsinduzierte Angina [J: Ja, N: Nein]
- Oldpeak: ST-Depression = ST [Numerischer Wert gemessen in Depression]
- ST_Slope: Die Steigung des Spitzen-Übungs-ST-Segments [Up: Aufsteigend, Flat: Flach, Down: Absteigend]
- HeartDisease: Ausgabeklasse [1: Herzkrankheit, 0: Normal]

In [ ]:
```
# using the pandas method "describe()"" to get a describtion of the datas
# ".T" transposes the dataframe (rows and columns are switched)
df.describe().T
```

Out[ ]:

	count	mean	std	min	25%	50%	75%	max
Age	918.0	53.510893	9.432617	28.0	47.00	54.0	60.0	77.0
RestingBP	918.0	132.396514	18.514154	0.0	120.00	130.0	140.0	200.0
Cholesterol	918.0	198.799564	109.384145	0.0	173.25	223.0	267.0	603.0
FastingBS	918.0	0.233115	0.423046	0.0	0.00	0.0	0.0	1.0
MaxHR	918.0	136.809368	25.460334	60.0	120.00	138.0	156.0	202.0
Oldpeak	918.0	0.887364	1.066570	-2.6	0.00	0.6	1.5	6.2
HeartDisease	918.0	0.553377	0.497414	0.0	0.00	1.0	1.0	1.0

Bereits nachdem man sich die Beschreibung des Datensatzes anschaut, kann man feststellen, dass die SPalten "Cholesterol" und "RestingBP" unerwartete minimal Werte aufweisen (ruhe Puls und Cholisterinspiegel können keine Werte von 0 annehmen).

In [ ]:
```
# count null values
null_values_count = (df['RestingBP'] == 0).sum()
print("Anzahl der Nullwerte in der Spalte 'RestingBP':", null_values_coun
```

```
Anzahl der Nullwerte in der Spalte 'RestingBP': 1
```

In [ ]:
```
# delete the only patient with the null value in RestingBP
df = df[df['RestingBP'] != 0]
```

Da es nur bei einem Patienten eine vermutliche Fehlmessung gab, wird dieser Patient aus dem Datensatz gelöscht.

In [ ]:
```
# count null values
null_values_count = (df['Cholesterol'] == 0).sum()
print("Anzahl der Nullwerte in der Spalte 'Cholesterol':", null_values_co
```

```
Anzahl der Nullwerte in der Spalte 'Cholesterol': 171
```

Leider weisen dennoch 171 Patienten bei Cholesterol den Wert 0 auf. Dies war bei der initialen explorativen Datenanalyse nicht auf den ersten Blick ersichtlich. Da das löschen von 171 Einträgen problematisch ist, wird in den fehldenen Stellen der durschnittliche Cholesterol Wert des Datensatzes eingesetzt. Somit sollen erheblichere Verfälschungen im Machine Learning Model im nachhinein vermieden werden. Ein Modell, das mit Daten von Patienten mit einem Cholesterol Wert von 0 trainiert ist, ist in der Realität nicht nützlich.

In [ ]:
```
# we don't wont the 0 values, when calculating the mean value
df_cleaned = df[df['Cholesterol'] != 0]

# calculate mean value
average_chol = round(df_cleaned['Cholesterol'].mean())

print("Durchschnittlicher Cholesterinspiegel nach Entfernen von Nullwerte
```

```
Durchschnittlicher Cholesterinspiegel nach Entfernen von Nullwerten (ohne
Nachkommastellen): 245
```

In [ ]:
```
# replace 0 values with the mean value
df.loc[df['Cholesterol'] == 0, 'Cholesterol'] = average_chol
```

In [ ]:
```
# check if the anomaly still exists
df["Cholesterol"].min()
```

Out[ ]:
```
85
```

In [ ]:
```
# checking for missung values in the dataframe
missing_values = df.isnull().sum()
missing_values
```

Out[ ]:
```
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 [ ]:
```
# checking for duplicated rows in the dataframe
duplicates = df.duplicated().sum()
```

```
duplicates
```

Out[ ]: 0

In [ ]:
```
# determining unique values of categorial columns in the dataframe
categorical_columns = ['Sex', 'ChestPainType', 'RestingECG', 'ExerciseAng
for col in categorical_columns:
    unique_values = df[col].unique()
    print(f"Eindeutige Werte für {col}:")
    print(unique_values)
```

```
Eindeutige Werte für Sex:
['M' 'F']
Eindeutige Werte für ChestPainType:
['ATA' 'NAP' 'ASY' 'TA']
Eindeutige Werte für RestingECG:
['Normal' 'ST' 'LVH']
Eindeutige Werte für ExerciseAngina:
['N' 'Y']
Eindeutige Werte für ST_Slope:
['Up' 'Flat' 'Down']
```

In [ ]:
```
# get dataframe info
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 917 entries, 0 to 917
Data columns (total 12 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   Age             917 non-null    int64  
 1   Sex             917 non-null    object 
 2   ChestPainType   917 non-null    object 
 3   RestingBP       917 non-null    int64  
 4   Cholesterol     917 non-null    int64  
 5   FastingBS       917 non-null    int64  
 6   RestingECG      917 non-null    object 
 7   MaxHR           917 non-null    int64  
 8   ExerciseAngina  917 non-null    object 
 9   Oldpeak         917 non-null    float64
 10  ST_Slope        917 non-null    object 
 11  HeartDisease    917 non-null    int64  
dtypes: float64(1), int64(6), object(5)
memory usage: 93.1+ KB
```

In [ ]:
```
# getting the highest values of each column (categorial columns may be ig
df.max()
```

Out[ ]:
```
Age                77
Sex                 M
ChestPainType      TA
RestingBP         200
Cholesterol       603
FastingBS           1
RestingECG         ST
MaxHR             202
ExerciseAngina      Y
Oldpeak           6.2
ST_Slope           Up
HeartDisease        1
dtype: object
```

In [ ]:
```
# same goes for this but for minimal values
df.min()
```

Out[ ]:
```
Age                28
Sex                 F
ChestPainType     ASY
RestingBP          80
Cholesterol        85
FastingBS           0
RestingECG        LVH
MaxHR              60
ExerciseAngina      N
Oldpeak          -2.6
ST_Slope         Down
HeartDisease        0
dtype: object
```

In [ ]:
```
# check how many unique elements the dataset contains in each column
df.nunique()
```

Out[ ]:
```
Age                50
Sex                 2
ChestPainType       4
RestingBP          66
Cholesterol       221
FastingBS           2
RestingECG          3
MaxHR             119
ExerciseAngina      2
Oldpeak            53
ST_Slope            3
HeartDisease        2
dtype: int64
```

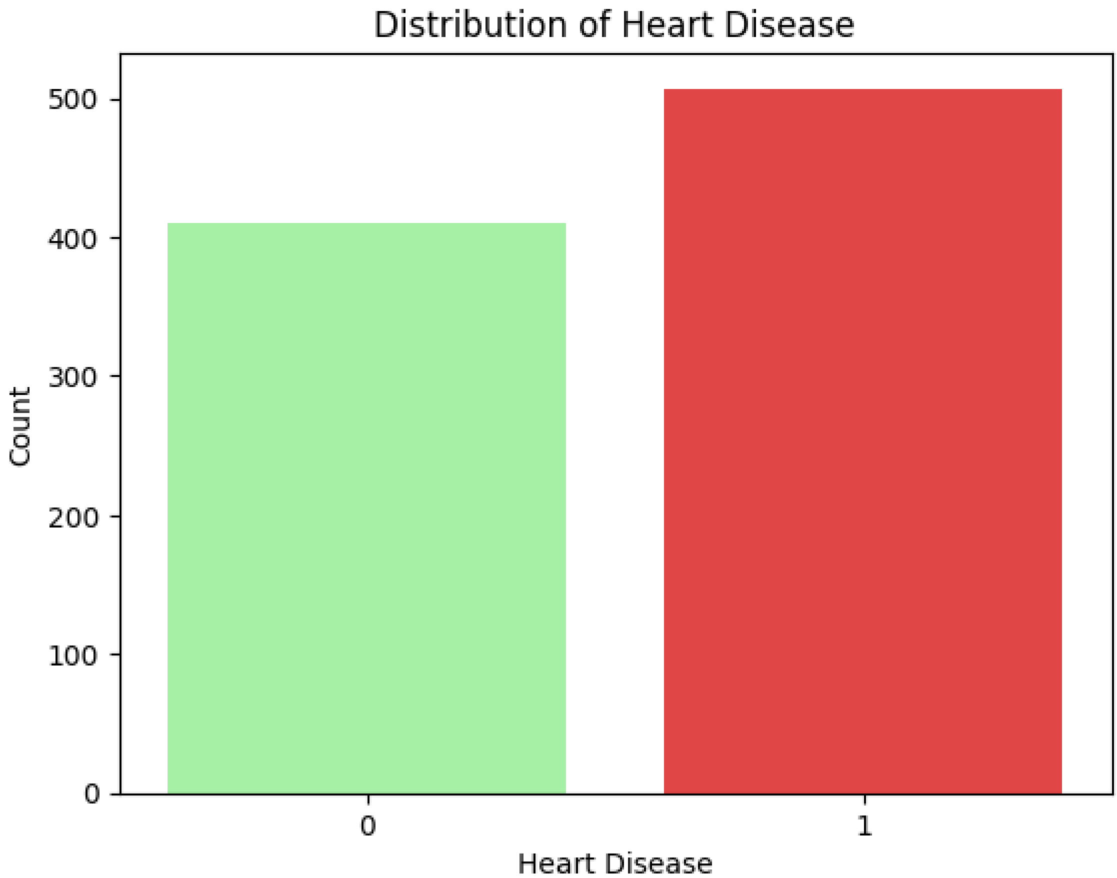
Die Analyse zur Datenqualität liefert auf den ersten Blick, bis auf die 2 Anomalien, kaum Mängel, da es keine fehlenden Einträge oder duplizierte Zeilen gibt. Auch die Spalten mit den kategorischen Werten liefern saubere und „aufgeräumte“ Werte. Der Datensatz ist im Allgemeinen sehr gut gepflegt.

Exploratory Data Analysis

Im folgenden werden die Daten analysiert und statistische Verteilungen und Merkmale, sowie Anhängigkeiten zwischen verschiedenen Attributen werden grafisch aufgezeigt.

In [ ]:
```
# visualize disease distribution in the dataset

colors_red_green = ["#9aff9a", "#ff3030"]

sns.countplot(x='HeartDisease', data=df, palette=colors_red_green)

plt.xlabel('Heart Disease')
plt.ylabel('Count')
plt.title('Distribution of Heart Disease')

plt.show()
```

```
/var/folders/3l/_xvv3581559_krvl1r82px5w0000gn/T/ipykernel_3402/182032164
9.py:5: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be remove
d in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for
the same effect.

  sns.countplot(x='HeartDisease', data=df, palette=colors_red_green)
```

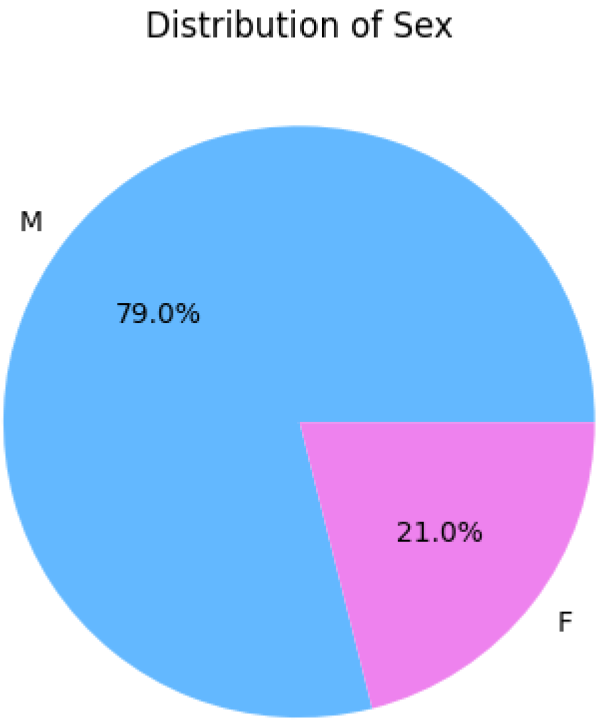


In [ ]:
```
heart_disease_distribution = df['HeartDisease'].value_counts()
heart_disease_distribution
```

Out[ ]:
```
HeartDisease
1    507
0    410
Name: count, dtype: int64
```

Die erste Visualisierung zeigt die Verteilung zwischen gesunden und kranken Patienten. Es ist eine leichte Inbalance der Werte vorhanden. Nach Absprache mit dem Dozenten kann diese Aufgrund ihrer leichten Ausprägung in diesem Fall ignoriert werden. Der Datensatz enthält 98 mehr betroffene als gesunde Patienten.

In [ ]:
```
# pie chart for sex distribution
distribution = df["Sex"].value_counts()
colors = ['#63b8ff', '#ee82ee']

plt.title("Distribution of Sex")

plt.pie(distribution, labels=distribution.index, colors=colors, autopct='

plt.show()
```

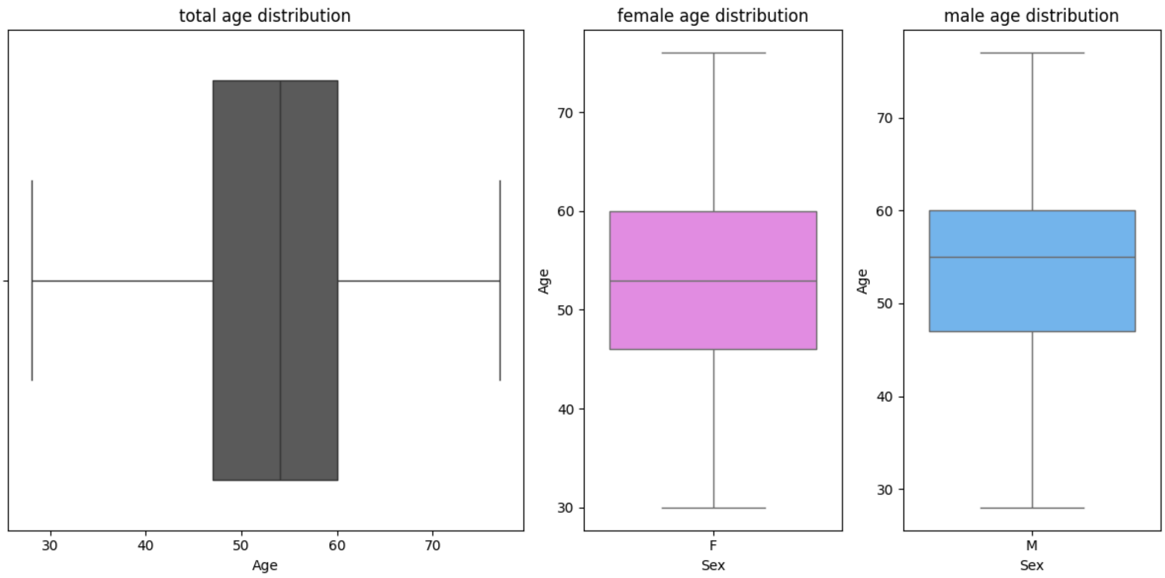


Dieses Kuchendiagramm zeigt die Verteilung der Geschlächter in den Daten. 79% der Patienten sind männlich und 21% sind weiblich.

In [ ]:
```
# create boxplots to display age distribution
fig = plt.figure(figsize=(12, 6))
gs = fig.add_gridspec(1, 3, width_ratios=[2, 1, 1])

# total age distribution
ax1 = fig.add_subplot(gs[0])
sns.boxplot(x=df["Age"], ax=ax1, color='#5c5c5c')
ax1.set_title('total age distribution')

# female age distribution
```

```
ax2 = fig.add_subplot(gs[1])
sns.boxplot(x='Sex', y='Age', data=df[df['Sex'] == 'F'], ax=ax2, palette=
ax2.set_title('female age distribution')

# male age distribution
ax3 = fig.add_subplot(gs[2])
sns.boxplot(x='Sex', y='Age', data=df[df['Sex'] == 'M'], ax=ax3, palette=
ax3.set_title('male age distribution')

plt.tight_layout()
plt.show()
```

```
/var/folders/3l/_xvv3581559_krvl1r82px5w0000gn/T/ipykernel_3402/293609210
3.py:12: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be remove
d in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for
the same effect.

  sns.boxplot(x='Sex', y='Age', data=df[df['Sex'] == 'F'], ax=ax2, palette
=['#ee82ee'])
/var/folders/3l/_xvv3581559_krvl1r82px5w0000gn/T/ipykernel_3402/293609210
3.py:17: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be remove
d in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for
the same effect.

  sns.boxplot(x='Sex', y='Age', data=df[df['Sex'] == 'M'], ax=ax3, palette
=['#63b8ff'])
```



Diese Boxplots zeigen die gesamte Altersverteilung sowie die Verteilung pro Geschlecht. Die Altersspanne liegt im Durschschnitt zwischen 48 und 60 Jahren.

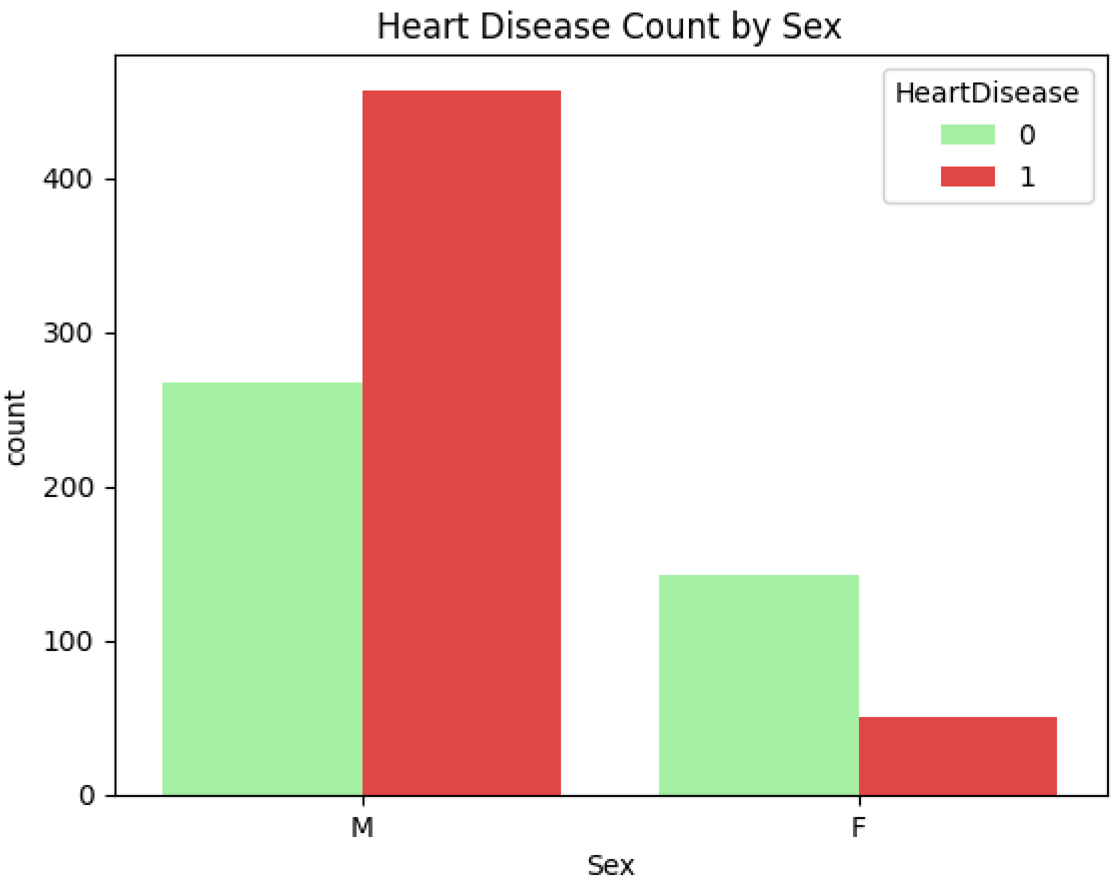
In [ ]:
```
# print value_counts of age to identify outliers
counts = df["Age"].value_counts()
print("Counts for Age:")
print(counts)
```

```
Counts for Age:
Age
54    51
58    42
55    40
56    38
57    38
52    36
51    35
59    35
62    35
53    33
60    32
48    31
61    31
63    30
50    25
46    24
41    24
43    24
64    22
65    21
49    21
47    19
44    19
42    18
45    18
38    16
67    15
39    15
66    13
69    13
40    13
35    11
37    11
68    10
34     7
74     7
70     7
36     6
71     5
32     5
72     4
29     3
75     3
33     2
77     2
76     2
31     2
30     1
28     1
73     1
Name: count, dtype: int64
```

Es gibt auch Patienten die "sehr" jung oder alt sind. Der jüngste Patient ist 28 und der älteste ist 77. Es kommen allerdings wenige Personen in diesem Datensatz vor, die an diese Altersgrenzen stoßen.

Im folgenden werden die verschiedenen kategorischen Attribute je nach Haufigkeit der Erkrankungen dargestellt.

In [ ]:
```
# create countplot to display HeartDisease distribution by sex
sns.countplot(x='Sex', hue='HeartDisease', data=df, palette=['#9aff9a', '

plt.xlabel("Sex")
plt.title("Heart Disease Count by Sex")

plt.show()
```

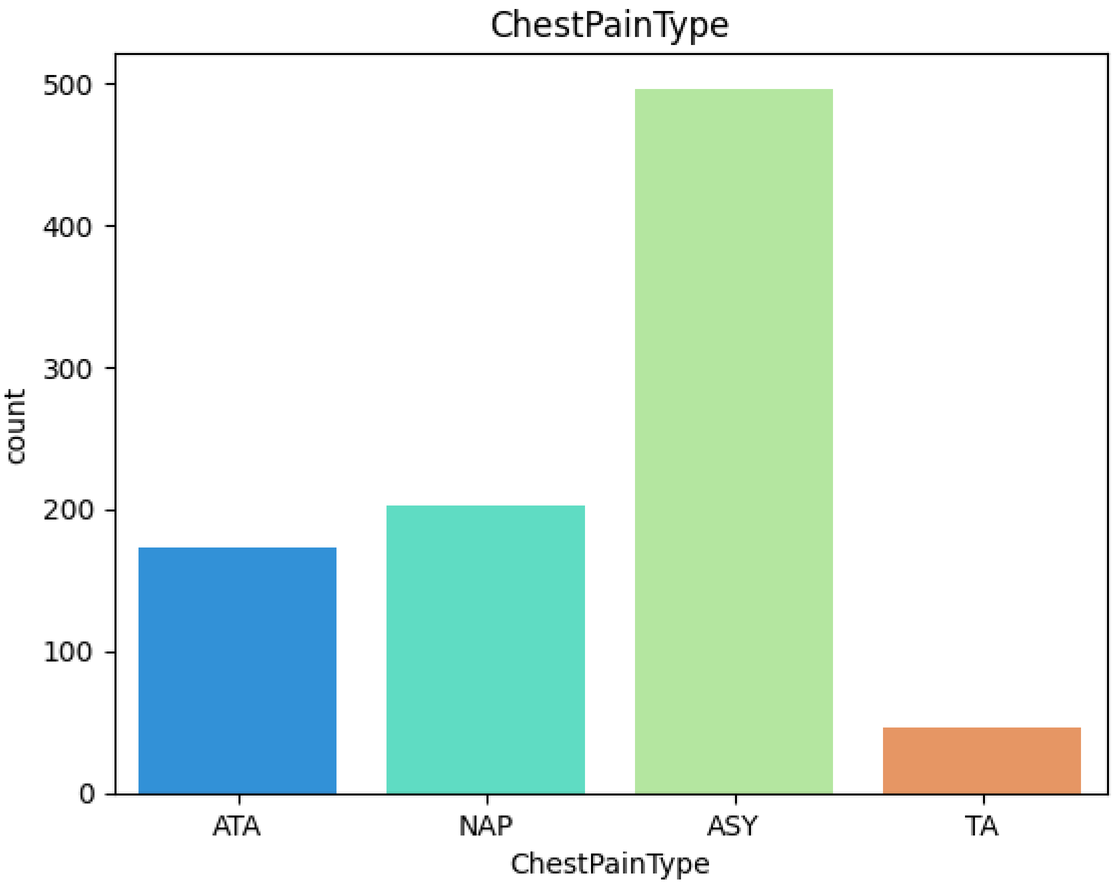


In diesem Datensatz gibt es innerhalt der männlichen Patientengruppe deutlich mehr Herzerkrankte, während es bei der weiblichen Gruppe weniger Betroffene gibt. Man beachte, dass der Datensatz mehr männliche Einträge enthält als weibliche.

In [ ]:
```
# countplot to display ChestPainType distribution
sns.countplot(x=df['ChestPainType'], palette="rainbow")
plt.title('ChestPainType')
```

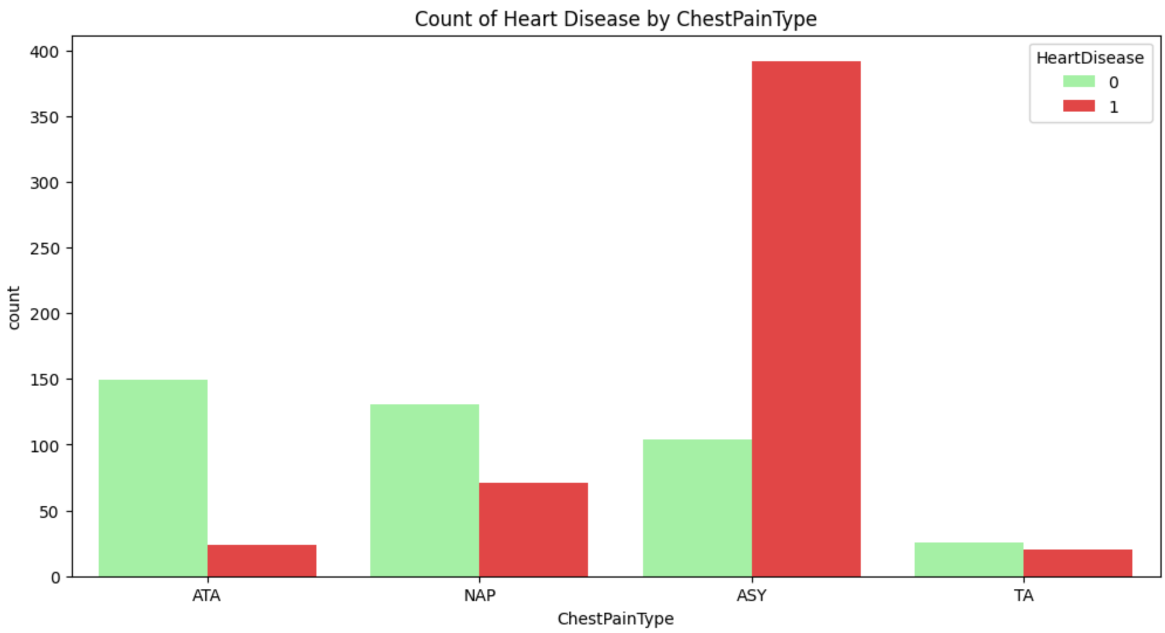
```
/var/folders/3l/_xvv3581559_krvl1r82px5w0000gn/T/ipykernel_3402/798950861.
py:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be remove
d in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for
the same effect.

  sns.countplot(x=df['ChestPainType'], palette="rainbow")
```

Out[ ]:
```
Text(0.5, 1.0, 'ChestPainType')
```



Die häufigste Ausprägung bei den Brustschmerzen sind die asymptomatischen Brustschmerzen. Die wenigsten Fälle beschreiben typical angina chest pain.

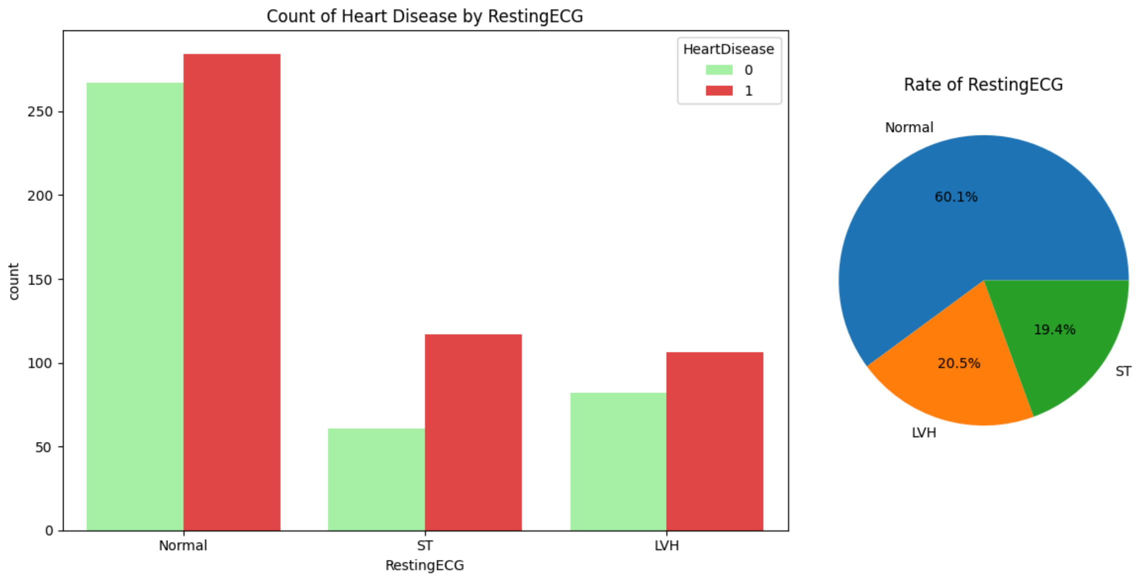
In [ ]:
```
# create a countplot showing the distribution of heart disease by ChestPa
plt.figure(figsize=(12, 6))
sns.countplot(x='ChestPainType', hue='HeartDisease', data=df, palette=col
plt.title('Count of Heart Disease by ChestPainType')

plt.show()
```



Interessanterweise zeigen Patienten mit asystomatischen Brustschmerzen am häufigsten eine Herzkrankheit auf. Bei der Gruppe TA gibt es in etwas gleich viele

Gesunde wie Erkrankte. In den anderen beiden Gruppen überwiegt die Anzahl der gesunden Patienten.

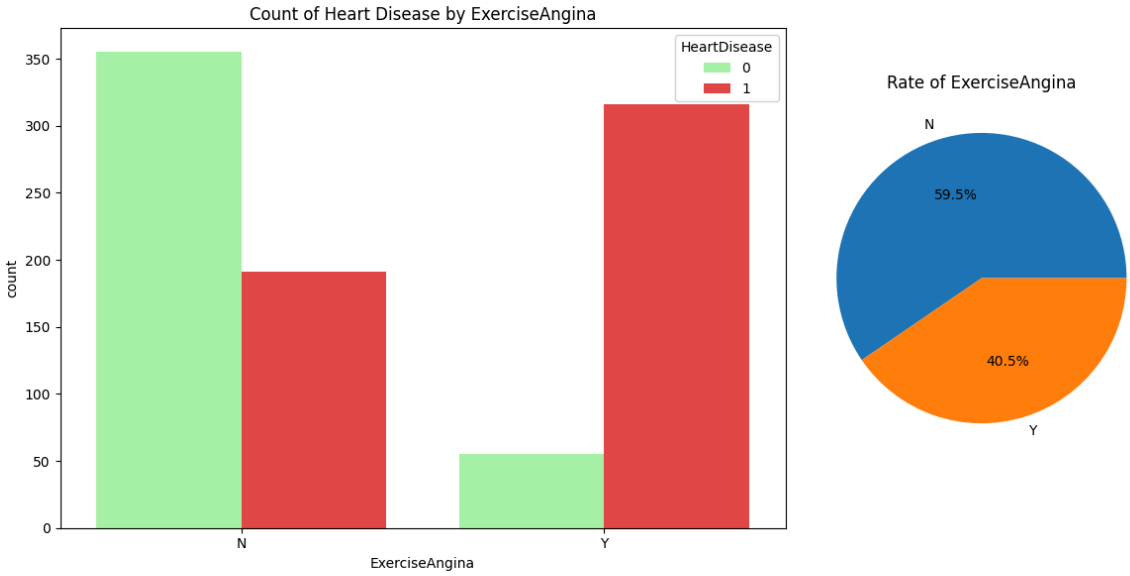
In [ ]:
```
# create a figure with a grid of 1 row and 2 columns, with the second col
fig = plt.figure(figsize=(12, 6))
gs = fig.add_gridspec(1, 2, width_ratios=[2, 1])

# subplot 1: Countplot showing the distribution of heart disease by Resti
ax1 = fig.add_subplot(gs[0])
sns.countplot(x='RestingECG', hue='HeartDisease', data=df, palette=colors
ax1.set_title('Count of Heart Disease by RestingECG')

# subplot 2: Pie chart illustrating the distribution of RestingECG values
ax2 = fig.add_subplot(gs[1])
types = df['RestingECG'].value_counts()
ax2.pie(types, labels=types.index, autopct='%1.1f%%')
ax2.set_title('Rate of RestingECG')

plt.tight_layout()
plt.show()
```



Die Ruheelektrokardiogrammergebnisse zeigen überwiegend normale Werte (60%). Die Gruppen LVH und ST (Beschreibung siehe oben) sind mit jeweils annähernd 20% seltener vertreten. Pro Gruppe gibt es allerdings stets mehr erkrankte als gesunde Patienten.

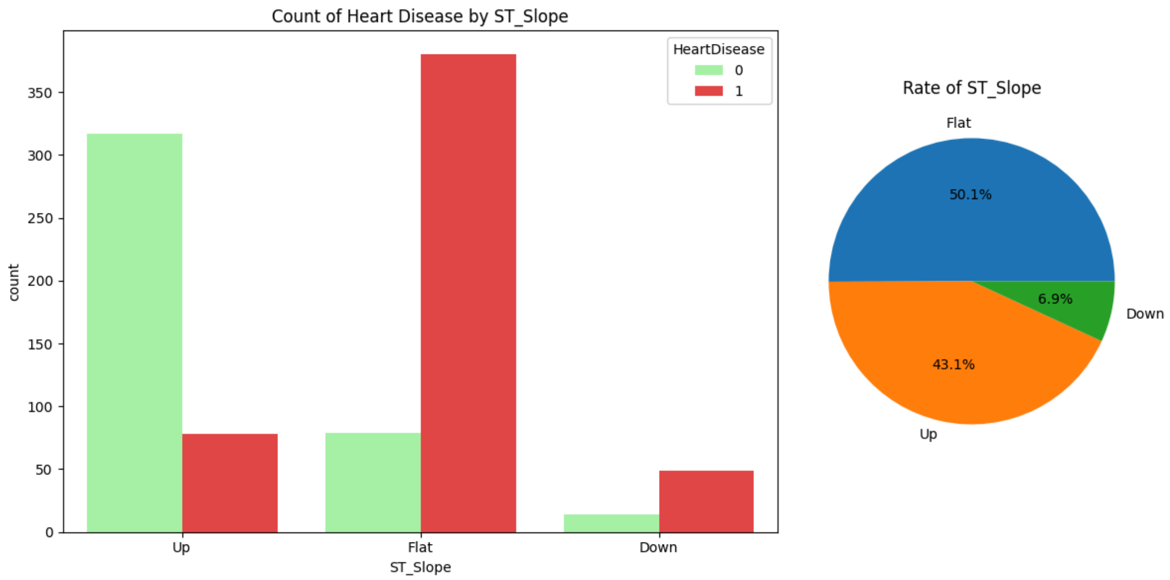
In [ ]:
```
# create a figure with a grid of 1 row and 2 columns, with the second col
fig = plt.figure(figsize=(12, 6))
gs = fig.add_gridspec(1, 2, width_ratios=[2, 1])

# subplot 1: Countplot showing the distribution of heart disease by Exerc
ax1 = fig.add_subplot(gs[0])
sns.countplot(x='ExerciseAngina', hue='HeartDisease', data=df, palette=co
ax1.set_title('Count of Heart Disease by ExerciseAngina')

# subplot 2: Pie chart illustrating the distribution of ExerciseAngina in
ax2 = fig.add_subplot(gs[1])
types = df['ExerciseAngina'].value_counts()
ax2.pie(types, labels=types.index, autopct='%1.1f%%')
ax2.set_title('Rate of ExerciseAngina')
```

```
plt.tight_layout()
plt.show()
```



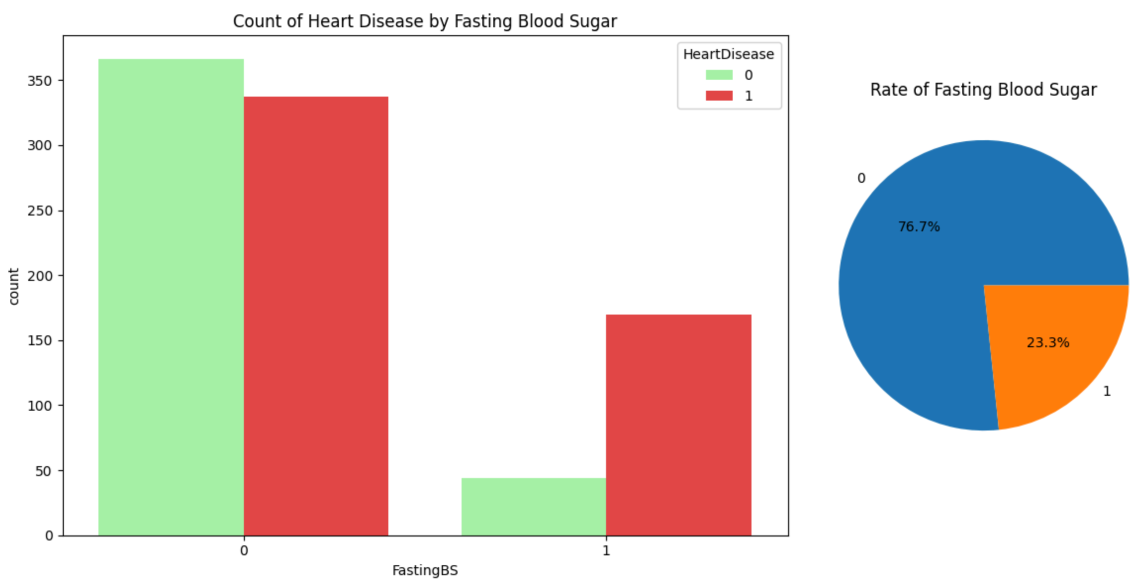
Die Mehrheit im Datensatz besitzt keine Belastungsinduzierte Brustschmerzen. Diejenigen Patienten die derartige Brustschmerzen aufweisen, haben jedoch signigfikant öfter eine Herzerkranung als die andere Gruppe.

In [ ]:
```
# create a figure with a grid of 1 row and 2 columns, with the second col
fig = plt.figure(figsize=(12, 6))
gs = fig.add_gridspec(1, 2, width_ratios=[2, 1])

# subplot 1: Countplot showing the distribution of heart disease by ST_Sl
ax1 = fig.add_subplot(gs[0])
sns.countplot(x='ST_Slope', hue='HeartDisease', data=df, palette=colors_r
ax1.set_title('Count of Heart Disease by ST_Slope')

# subplot 2: Pie chart illustrating the distribution of ST_Slope in the d
ax2 = fig.add_subplot(gs[1])
types = df['ST_Slope'].value_counts()
ax2.pie(types, labels=types.index, autopct='%1.1f%%')
ax2.set_title('Rate of ST_Slope')

plt.tight_layout()
plt.show()
```



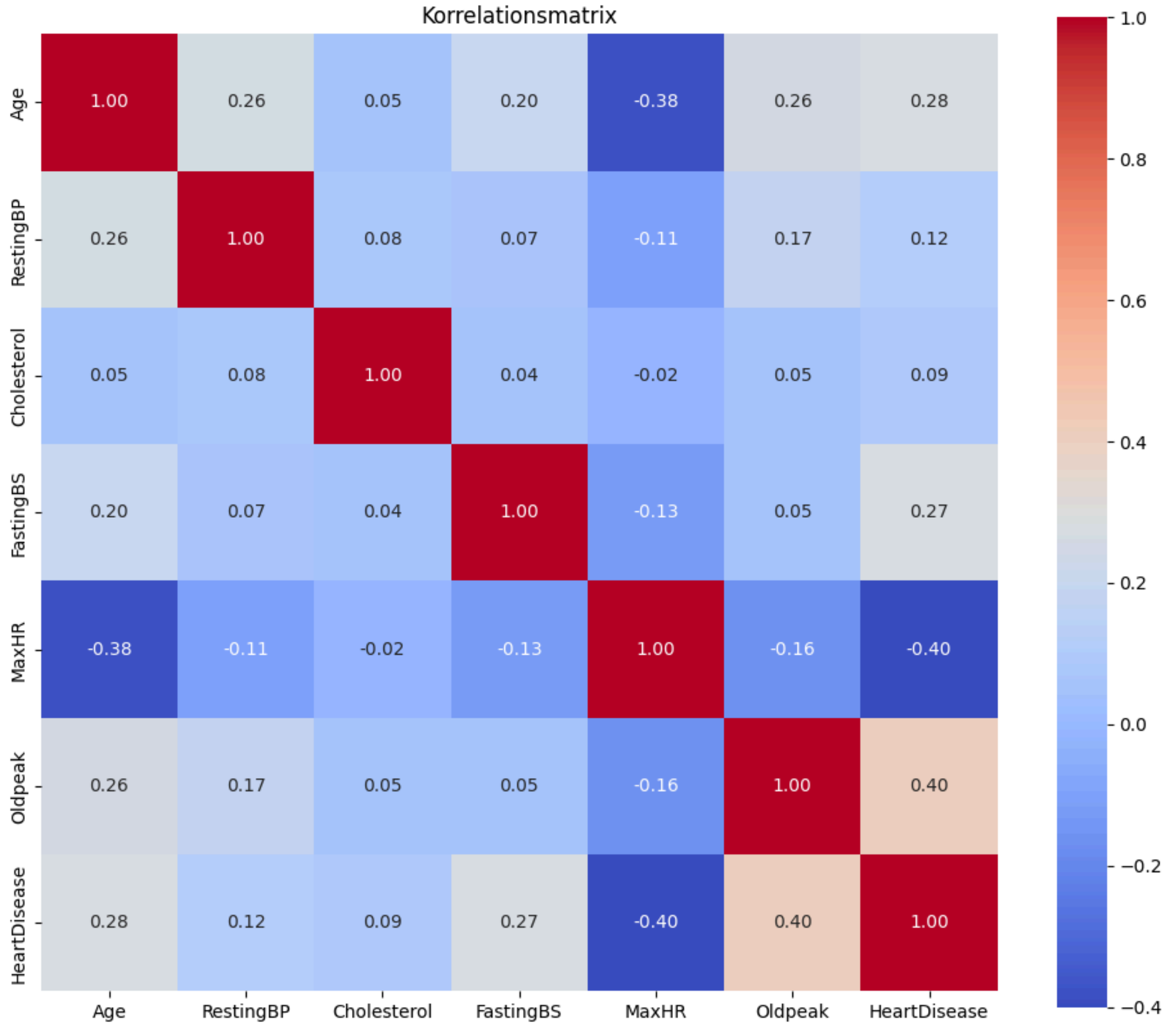
ST_Slope beschreibt die Steigung des peak exercise ST Segments. Wie man der Visualisierung entnehmen kann gibt es überwiegend flache und und steigende ST Segmente. Bis auf den steigenden Segmenten gibt es in jeder Gruppe deutlich mehr Herzerkrankte zu geben. Wie es scheint, sind Patienten mit einer steigenden Kurve wahrscheinlicher gesund.

In [ ]:
```
fig = plt.figure(figsize=(12, 6))
gs = fig.add_gridspec(1, 2, width_ratios=[2, 1])

# countplot for heart disease by fasting blood sugar
ax1 = fig.add_subplot(gs[0])
sns.countplot(x='FastingBS', hue='HeartDisease', data=df, palette=colors_
ax1.set_title('Count of Heart Disease by Fasting Blood Sugar')

# pie chart for the distribution of fasting blood sugar
ax2 = fig.add_subplot(gs[1])
types = df['FastingBS'].value_counts()
ax2.pie(types, labels=types.index, autopct='%1.1f%%')
ax2.set_title('Rate of Fasting Blood Sugar')

plt.tight_layout()
plt.show()
```



Das Attribut FastingBS beschreibt den nüchternen Blutzuckerspiegel eines Patienten, während Werte von 1 einen Blutzuckerspiegel von > 120 mg/dl kennzeichnen. Werte darunter sind mit 0 beschrieben. Der Großteil der Patienten fällt unter die Gruppe 0. In dieser Gruppe gibt es annähernd gleich viele Patienten mit sowie ohne Krankheit. In der Gruppe mit dem höheren Blutzuckerspiegel haben weitaus mehr Patienten eine Herzkrankheit.

In [ ]:
```
# create corelation matrix

correlations = df.corr(numeric_only=True)

plt.figure(figsize=(12, 10))
sns.heatmap(correlations, annot=True, cmap='coolwarm', fmt=".2f", square=
plt.title('Korrelationsmatrix')
plt.show()
```

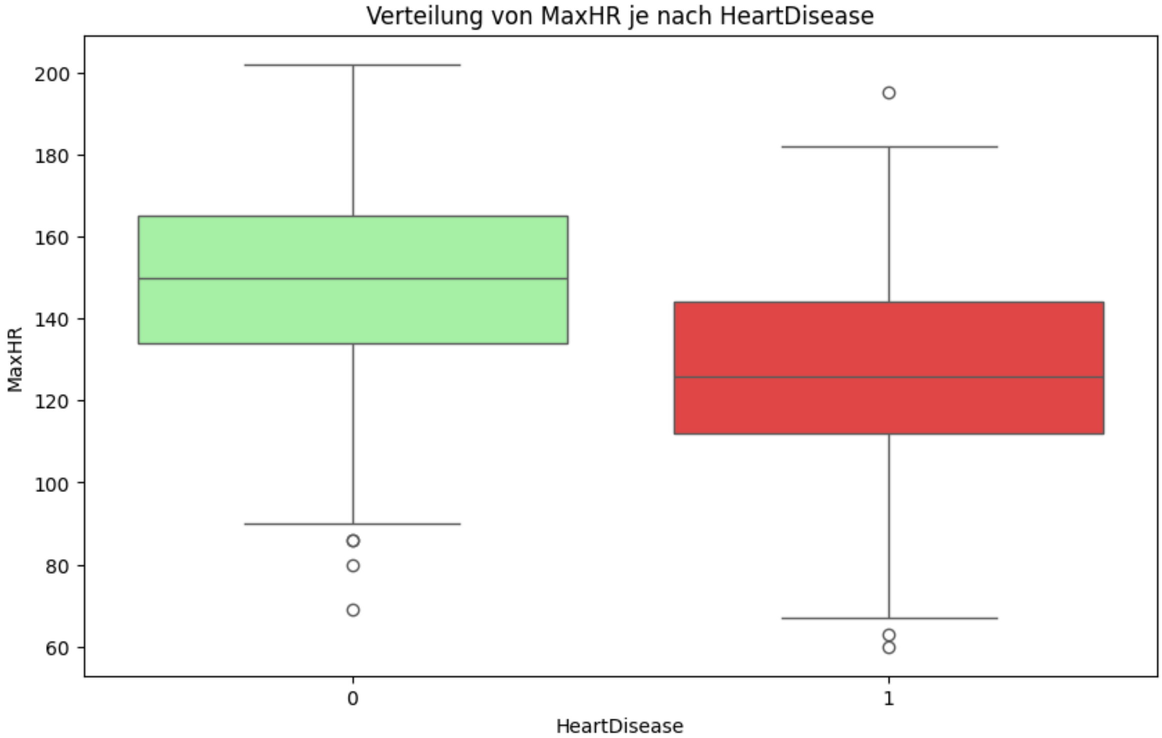


Um die Korrelationen zwischen den einzelnen Attributen zu ermittlen, wird diese Korrelationsmatrix erstellt. Die stärkste Korrelation weisen die Attribute Oldpeak und Heartdisease auf. Das lässt darauf schließen, dass sich je nach Gruppe innerhalb des Attributs Oldpeak eine genauere Aussage über den Gesundheitszustands eines Patienten fallen lässt. Weitere, jedoch schwächere Korrelationen (>= 0.20) herrschen zwischen den Attributen Age und Heartdisease, Age und Oldpeak, RestingPB und Age, FastingBS und Age, MaxHR und Cholesterol. Auffällig ist die negative Korrelation zwsichen MaxHR und Heartdisease. Der Wert -0,40 besagt, dass ein Patient mit

Herzerkrankung einen tendenziell niedrigeren Maximalen Puls hat. Dies erschien auf den ersten Blick merkwürdig, da die Annahme herrschte, Herzerkrankte menschen hätten einen höheren Puls.

In [ ]:
```
# create box plot
plt.figure(figsize=(10, 6))
sns.boxplot(x='HeartDisease', y='MaxHR', data=df, palette=colors_red_gree
plt.title('Verteilung von MaxHR je nach HeartDisease')
plt.xlabel('HeartDisease')
plt.ylabel('MaxHR')
plt.show()
```

```
/var/folders/3l/_xvv3581559_krvl1r82px5w0000gn/T/ipykernel_3402/995114334.
py:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be remove
d in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for
the same effect.

  sns.boxplot(x='HeartDisease', y='MaxHR', data=df, palette=colors_red_gre
en)
```



Doch dieser Boxplot bestätigt den Wert in der Korrelationsmatrix.

Outlier Detection

In den nächsten werden die Outlier im Datensatz analysiert.

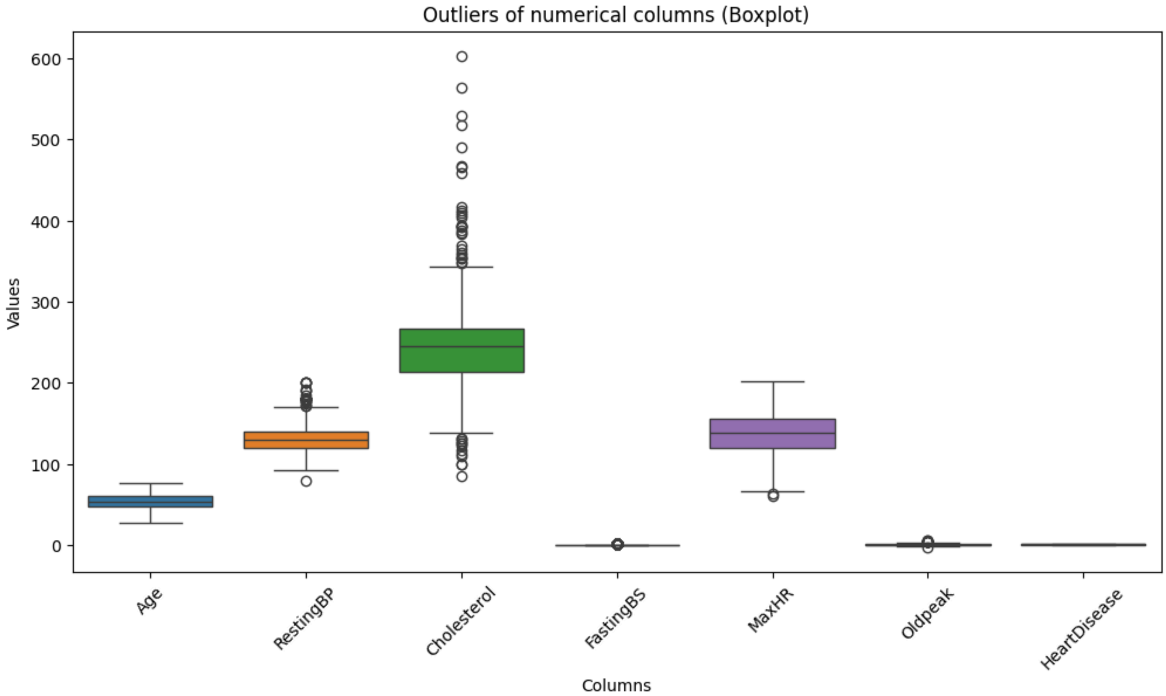
In [ ]:
```
# choose only numeric columns
numeric_cols = df.select_dtypes(include=[np.number]).columns

# funtion to define outliers
def detect_outliers(data):
    """
```

```
    Detect outliers in the given DataFrame.

    Parameters:
    - data (DataFrame): The DataFrame containing the data.

    Returns:
    - outliers (list): A list of indices corresponding to the outliers in
    """
    outliers = []
    for col in data.columns:
        q1 = data[col].quantile(0.25)
        q3 = data[col].quantile(0.75)
        iqr = q3 - q1
        lower_bound = q1 - 1.5 * iqr
        upper_bound = q3 + 1.5 * iqr
        outlier_indices = data[(data[col] < lower_bound) | (data[col] > u
        outliers.extend(outlier_indices)
    return outliers


# find outliers
outliers_indices = detect_outliers(df[numeric_cols])

# remove duplicated indices
outliers_indices = list(set(outliers_indices))

print("Indices of outliers:", outliers_indices)

print("Outlier rows:")
print(df.iloc[outliers_indices])
```

```
Indices of outliers: [515, 516, 518, 521, 522, 531, 532, 536, 537, 538, 2
8, 541, 30, 544, 546, 547, 36, 549, 550, 38, 553, 554, 556, 557, 559, 563,
52, 564, 58, 571, 826, 573, 574, 575, 577, 579, 580, 69, 582, 68, 584, 58
5, 76, 589, 78, 592, 593, 594, 595, 84, 86, 599, 604, 605, 606, 607, 97, 9
8, 610, 612, 613, 102, 103, 616, 105, 108, 109, 621, 624, 112, 117, 120, 6
32, 123, 639, 128, 132, 644, 650, 658, 659, 660, 149, 666, 667, 155, 160,
673, 672, 675, 165, 166, 679, 682, 686, 182, 185, 187, 189, 190, 702, 701,
718, 208, 210, 725, 728, 732, 734, 224, 738, 227, 744, 238, 752, 241, 242,
759, 247, 250, 256, 771, 774, 263, 775, 780, 782, 784, 785, 274, 275, 278,
790, 791, 793, 795, 284, 796, 799, 802, 803, 294, 295, 296, 297, 298, 299,
300, 809, 302, 303, 304, 305, 306, 308, 309, 820, 311, 312, 313, 314, 315,
316, 317, 318, 319, 320, 321, 322, 323, 324, 325, 838, 327, 328, 329, 330,
331, 842, 333, 334, 335, 843, 337, 338, 339, 340, 341, 342, 855, 343, 344,
850, 347, 349, 350, 869, 871, 872, 365, 880, 370, 372, 887, 888, 377, 378,
900, 389, 901, 390, 393, 394, 395, 396, 397, 398, 399, 400, 401, 402, 915,
403, 404, 405, 406, 407, 408, 409, 411, 410, 412, 413, 414, 415, 420, 422,
423, 424, 425, 430, 437, 441, 442, 443, 444, 448, 454, 457, 458, 460, 914,
469, 472, 473, 908, 475, 476, 477, 478, 480, 481, 482, 485, 486, 911, 491,
496, 498, 500, 503, 504, 505, 508, 511]
Outlier rows:
     Age Sex ChestPainType  RestingBP  Cholesterol  FastingBS RestingECG  \
516   68   M           NAP        150          195          1     Normal   
517   65   M           ASY        150          235          0     Normal   
519   63   M           ASY         96          305          0         ST   
522   50   M           ASY        144          349          0        LVH   
523   59   M           ASY        124          160          0     Normal   
..   ...  ..           ...        ...          ...        ...        ...   
504   62   M           ASY        158          210          1     Normal   
505   55   M           NAP        136          245          1         ST   
506   75   M           ASY        136          225          0     Normal   
509   58   M           ASY        110          198          0     Normal   
512   35   M           NAP        123          161          0         ST   

     MaxHR ExerciseAngina  Oldpeak ST_Slope  HeartDisease  
516    132              N      0.0     Flat             1  
517    120              Y      1.5     Flat             1  
519    121              Y      1.0       Up             1  
522    120              Y      1.0       Up             1  
523    117              Y      1.0     Flat             1  
..     ...            ...      ...      ...           ...  
504    112              Y      3.0     Down             1  
505    131              Y      1.2     Flat             1  
506    112              Y      3.0     Flat             1  
509    110              N      0.0     Flat             1  
512    153              N     -0.1       Up             0  

[275 rows x 12 columns]
```

In [ ]:

```
# box plot to display outliers
plt.figure(figsize=(12, 6))
sns.boxplot(data=df[numeric_cols])
plt.xticks(rotation=45)
plt.title('Outliers of numerical columns (Boxplot)')
plt.xlabel('Columns')
plt.ylabel('Values')
plt.show()
```



Die Outlier Detection liefert bis gute Ergebnisse. Dadurch, dass die Anomalien bei der Analyse der Datenqulität behoben worden, gibt es keine weiteren erheblichen Einschränkungen in den Daten. Während Serum Cholesterol Werte von > 600 äußert gefährlich erscheinen, sind diese in der Realität dennoch möglich.

Machine Learning

Der vorliegende Datensatz liefert ein binäres Klassifikationsproblem. Um einen ersten Ansatz für die Auswahl eines endgültigen Classifiers zu ermitteln, wurden im Folgenden 3 verschiedene Classifier getestet.

Diese wären:

- Randomforest
- Logistic Regression
- Support Vector Machine

Die Auswahl dieser drei Classifie beruht auf ihrer Effektivität bei binären Klassifikationsproblemen. Jeder Algorithmus bietet spezifische Vorzüge und kann unterschiedliche Aspekte des Problems abdecken.

Metriken

Für die Bewertung des Models werden insbesondere die folgenden Metriken verwendet:

- Recall
- F1-Score
- ROC-AUC-Score

Der F1-Score ist ein harmonisches Mittelmaß zwischen Präzision und Recall und ermöglicht eine ausgewogene Bewertung von False Positives und False Negatives.

Der Recall bewertet die Fähigkeit des Modells, positive Instanzen korrekt zu identifizieren, was besonders wichtig ist, um sicherzustellen, dass keine relevanten Fälle übersehen werden.

Der ROC-AUC-Score bewertet die Fähigkeit des Modells, zwischen den Klassen zu unterscheiden, indem er die Fläche unter der ROC-Kurve misst, wobei ein höherer Wert auf eine bessere Leistung hinweist.

Diese Metriken in Kombination bieten eine umfassende Bewertung des Modells für das binäre Klassifikationsproblem.

Feature Engineering

Da die Kategorischen Attribute nicht vom Classifier erkannt werden, wird hier ein One-Hot-Encoding angewandt. One-Hot-Encoding ist eine Methode zur Umwandlung von kategorischen Variablen in ein binäres Format, das von maschinellen Lernalgorithmen besser verstanden werden kann.

In [ ]:
```
# featrue engineering
df_encoded = pd.get_dummies(df, columns=["Sex", "ChestPainType", "Resting
```

Im nächsten Schritt werden die target und feature Variablen festgelegt.

In [ ]:
```
# preparation for train/test split
target = df_encoded["HeartDisease"]
features = df_encoded.drop("HeartDisease", axis=1)
```

Train-/Testsplit

Um einen Bias im Machine Learning Model zu vermeiden, splittet man den Datensatz auf in Trainings- und Testdaten. Der Trainingsdatensatz wird verwendet, um das tatsächliche Modell zu erstellen, das der Algorithmus verwenden wird, wenn er neuen Daten ausgesetzt ist.

Das Testset ist der letzte Datensatz, der verwendet wird. Die Genauigkeit bei der Vorhersage des Testsets entspricht der Genauigkeit des ML-Algorithmus.

Für den train/test Split wird ein Verhältnis von 80/20 gewählt.

In [ ]:
```
# train/test split (80%/20%)
features_train, features_test, target_train, target_test = train_test_spl
```

Classifier

Die folgende Methoden werden verwendet, um die drei gewählten Classifier zur fitten, die Scores anzuzeigen und jeweils die Confusion Matrix auszuegebn.

In [ ]:
```
def model(classifier):
    """
    Train the classifier on the training data and evaluate its performanc

    Parameters:
    - classifier: The classifier model to be trained and evaluated.

    Returns:
    None
    """
    classifier.fit(features_train, target_train)
    prediction = classifier.predict(features_test)
    print("Accuracy: {:.2%}".format(accuracy_score(target_test, predictio
    print("ROC_AUC Score: {:.2%}".format(roc_auc_score(target_test, predi


def model_evaluation(classifier):
    """
    Evaluate the classifier using various performance metrics and visuali

    Parameters:
    - classifier: The trained classifier model.

    Returns:
    None
    """
    # disply confusion Matrix
    cm = confusion_matrix(target_test, classifier.predict(features_test))
    names = ['True Neg', 'False Pos', 'False Neg', 'True Pos']
    counts = [value for value in cm.flatten()]
    percentages = ['{:.2%}'.format(value) for value in cm.flatten() / np.
    labels = [f'{v1}\n{v2}\n{v3}' for v1, v2, v3 in zip(names, counts, pe
    labels = np.asarray(labels).reshape(2, 2)
    sns.heatmap(cm, annot=labels, cmap=colors, fmt='')

    # show classification Report
    print(classification_report(target_test, classifier.predict(features_
```

ML: Random Forrest

Der Random Forest Classifier ist ein Algorithmus für die Klassifizierung, der auf der Kombination mehrerer Entscheidungsbäume basiert. Er eignet sich gut für die Vorhersage von Herzkrankheiten aufgrund seiner Fähigkeit, mit verschiedenen Datentypen umzugehen und robuste Ergebnisse zu liefern. Er heißt "Random" Forest, da beim Algorithmus zwei zufällige Prozesse ablaufen. Zum einen das Bootstrapping zum anderen die Feature Auswahl beim erstellen der Entscheidungsbäume. Der Algorithmus baut also eine Vielzahl an Bäumen, die auf zufälligen Daten des Datensatzes basieren.

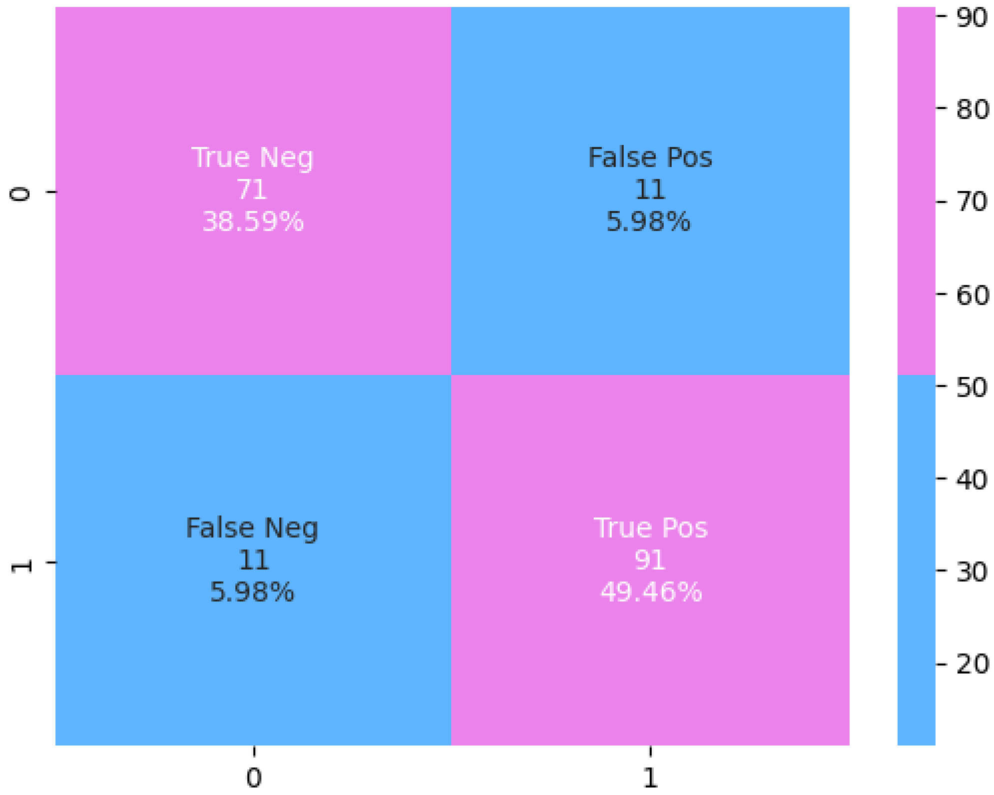
In [ ]:
```
# defining RFC
forest = RandomForestClassifier()
```

```
# get scores
model(forest)
```

```
Accuracy: 88.04%
ROC_AUC Score: 87.90%
```

In [ ]:
```
# get evalutation
model_evaluation(forest)
```

```
              precision    recall  f1-score   support

           0       0.87      0.87      0.87        82
           1       0.89      0.89      0.89       102

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



ML: Logistic Regression

Die logistische Regression ist ein Algorithmus zur Klassifizierung, der die Wahrscheinlichkeit für das Eintreten eines Ereignisses basierend auf einer oder mehreren unabhängigen Variablen schätzt. Dabei nutzt sie die logistische Funktion, um die Vorhersage zwischen 0 und 1 zu skalieren. Sie eignet sich gut für binäre Klassifizierungsaufgaben wie die Vorhersage von Herzkrankheiten.

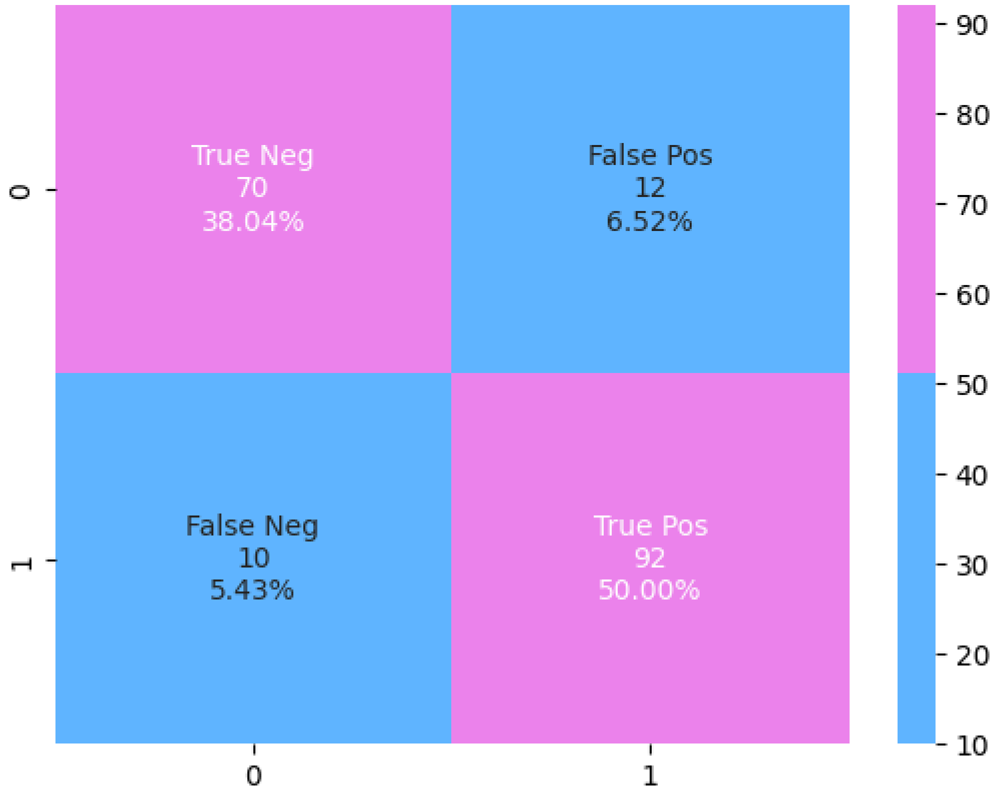
In [ ]:
```
# defining LRC
classifier_lr = LogisticRegression(max_iter=10000)
```

```
# get scores
model(classifier_lr)
```

```
Accuracy: 88.04%
ROC_AUC Score: 87.78%
```

In [ ]:
```
# get scores
model_evaluation(classifier_lr)
```

```
              precision    recall  f1-score   support

           0       0.88      0.85      0.86        82
           1       0.88      0.90      0.89       102

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



ML: Support Vector Machine

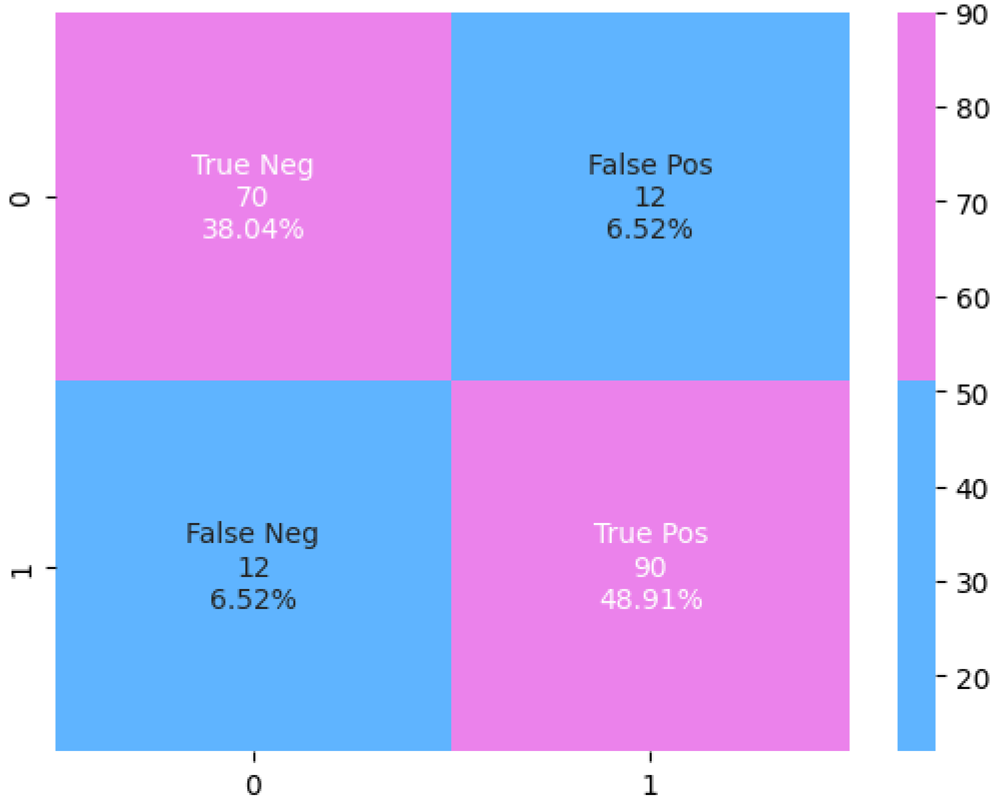
Die Support Vector Machine ist ebenfalls ein Klassifizierungsalgorithmus, der darauf abzielt, eine Trennung zwischen den verschiedenen Klassen zu finden, indem er die beste Entscheidungsgrenze (Hyperplane) zwischen den Datenpunkten sucht. Sie funktioniert, indem sie den Abstand zwischen den Datenpunkten maximiert und gleichzeitig eine minimale Fehlerrate aufweist. SVM eignet sich gut für datengetriebene Anwendungen mit komplexen Entscheidungsgrenzen und kann auch mit nicht-linearen Daten umgehen, indem sie den sogenannten Kernel-Trick anwendet. In Bezug auf Herzkrankheiten eignet sich die SVM, wenn die Daten gut separierbar sind und klare Entscheidungsgrenzen zwischen den Klassen existieren.

In [ ]:
```
# defining SVM
svc = SVC(kernel = 'linear', C = 0.1)

# get scores
model(svc)
```

```
Accuracy: 86.96%
ROC_AUC Score: 86.80%
```

In [ ]:
```
# get evaluation
model_evaluation(svc)
```

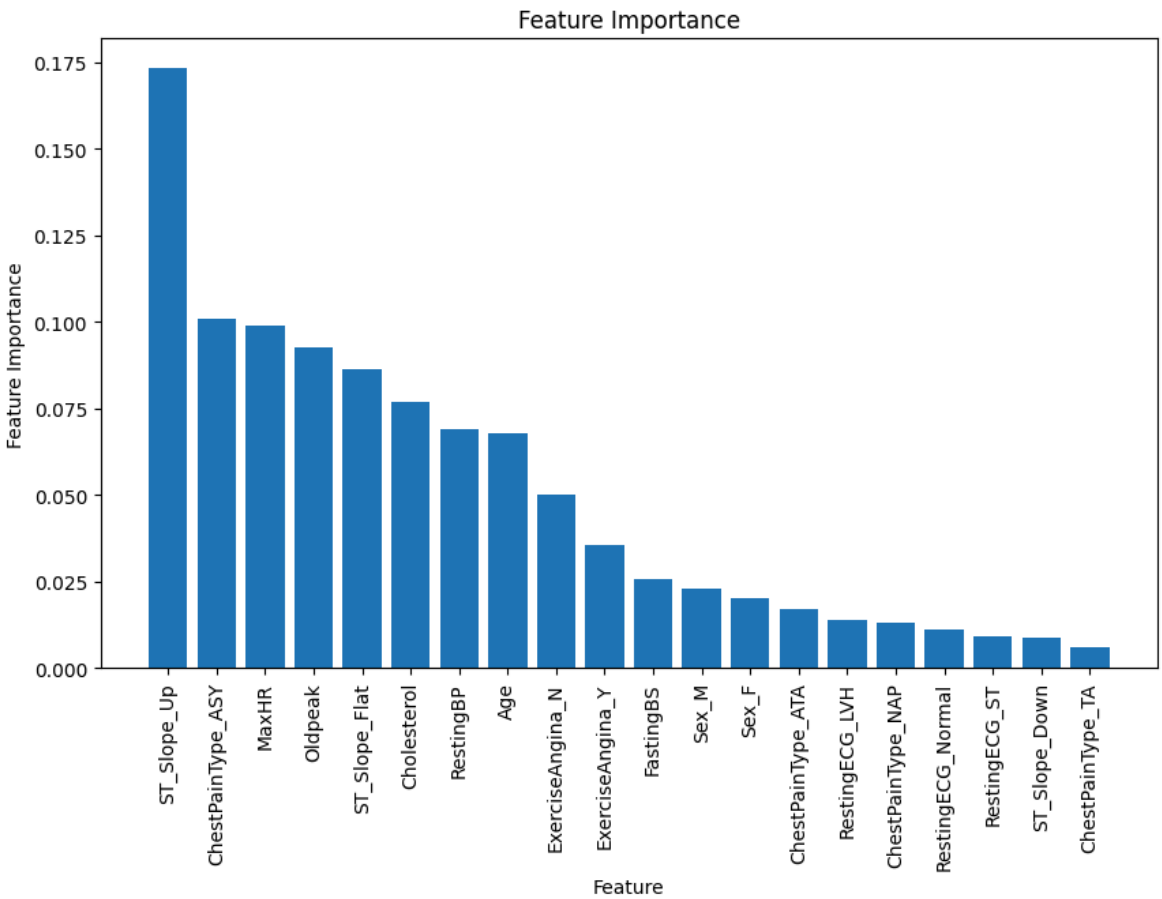
```
              precision    recall  f1-score   support

           0       0.85      0.85      0.85        82
           1       0.88      0.88      0.88       102

    accuracy                           0.87       184
   macro avg       0.87      0.87      0.87       184
weighted avg       0.87      0.87      0.87       184
```



Feature Importance für den RFC

In [ ]:
```
# plot for feature importances
importances = forest.feature_importances_
indices = np.argsort(importances)[::-1]
feature_names = features_train.columns

plt.figure(figsize=(10, 6))
plt.title("Feature Importance")
plt.bar(range(features_train.shape[1]), importances[indices], align="cent
plt.xticks(range(features_train.shape[1]), feature_names[indices], rotati
plt.xlabel("Feature")
```

```
plt.ylabel("Feature Importance")
plt.show()
```



Gridsearch Prameter Tuning

Das Parameter-Tuning wird nur für den Random Forest Classifier (RFC) durchgeführt. Der RFC ist bekannt für seine Vielseitigkeit und Robustheit, insbesondere bei binären Klassifikationsproblemen wie im vorliegenden Fall. Durch das Feintuning seiner Hyperparameter kann die Vorhersagegenauigkeit weiter optimiert und potenzielles Overfitting reduziert werden. Dies ermöglicht eine präzisere Identifizierung von Herzkrankheiten, was in medizinischen Anwendungen von entscheidender Bedeutung ist.

Das Parameter-Tuning wird mithilfe von Grid Search durchgeführt, einem Ansatz zur systematischen Suche nach den besten Hyperparameter-Kombinationen für ein Machine Learning-Modell. Grid Search durchläuft vordefinierte Kombinationen von Hyperparametern und bewertet die Leistung des Modells anhand einer bestimmten Metrik für jede Kombination. In unserem Fall optimieren wir den Receiver Operating Characteristic Area Under Curve (ROC AUC) Score. Der ROC AUC Score ist eine Metrik, die die Fähigkeit eines Modells bewertet, zwischen den Klassen zu unterscheiden und die Trade-offs zwischen True Positive Rate und False Positive Rate darstellt. Für das binäre Klassifikationsproblem mit Herzkrankheiten ist es wichtig, dass unser Modell eine hohe Unterscheidungskraft zwischen kranken und gesunden Patienten aufweist, weshalb wir den ROC AUC Score optimieren.

In [ ]:
```
# defining the grid search parameters
param_grid = {
    'n_estimators': [100, 200, 300],
    'max_depth': [None, 10, 20, 30],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4]
}
```

In [ ]:
```
# defining the hyperparameter space to identify the optimal combination o
grid_search = GridSearchCV(estimator=forest,
                           param_grid=param_grid,
                           scoring= "roc_auc",
                           refit="roc_auc",
                           cv=5,
                           n_jobs=-1,
                           verbose=4)
```

In [ ]:
```
# searching for the best combination
grid_search.fit(features_train, target_train)
```

```
Fitting 5 folds for each of 108 candidates, totalling 540 fits
[CV 1/5] END max_depth=None, min_samples_leaf=1, min_samples_split=2, n_es
timators=100;, score=0.913 total time=   0.1s
[CV 4/5] END max_depth=None, min_samples_leaf=1, min_samples_split=2, n_es
timators=100;, score=0.914 total time=   0.1s
[CV 3/5] END max_depth=None, min_samples_leaf=1, min_samples_split=2, n_es
timators=100;, score=0.942 total time=   0.1s
[CV 2/5] END max_depth=None, min_samples_leaf=1, min_samples_split=2, n_es
timators=100;, score=0.970 total time=   0.2s
[CV 5/5] END max_depth=None, min_samples_leaf=1, min_samples_split=2, n_es
timators=100;, score=0.872 total time=   0.2s
[CV 3/5] END max_depth=None, min_samples_leaf=1, min_samples_split=2, n_es
timators=200;, score=0.943 total time=   0.3s
[CV 1/5] END max_depth=None, min_samples_leaf=1, min_samples_split=2, n_es
timators=200;, score=0.919 total time=   0.3s
[CV 2/5] END max_depth=None, min_samples_leaf=1, min_samples_split=2, n_es
timators=200;, score=0.970 total time=   0.3s
[CV 4/5] END max_depth=None, min_samples_leaf=1, min_samples_split=2, n_es
timators=200;, score=0.911 total time=   0.3s
[CV 5/5] END max_depth=None, min_samples_leaf=1, min_samples_split=2, n_es
timators=200;, score=0.868 total time=   0.4s
[CV 1/5] END max_depth=None, min_samples_leaf=1, min_samples_split=5, n_es
timators=100;, score=0.922 total time=   0.2s
[CV 2/5] END max_depth=None, min_samples_leaf=1, min_samples_split=2, n_es
timators=300;, score=0.978 total time=   0.5s
[CV 2/5] END max_depth=None, min_samples_leaf=1, min_samples_split=5, n_es
timators=100;, score=0.974 total time=   0.2s
[CV 4/5] END max_depth=None, min_samples_leaf=1, min_samples_split=5, n_es
timators=100;, score=0.915 total time=   0.1s
[CV 3/5] END max_depth=None, min_samples_leaf=1, min_samples_split=2, n_es
timators=300;, score=0.944 total time=   0.5s
[CV 1/5] END max_depth=None, min_samples_leaf=1, min_samples_split=2, n_es
timators=300;, score=0.922 total time=   0.6s
[CV 5/5] END max_depth=None, min_samples_leaf=1, min_samples_split=5, n_es
timators=100;, score=0.871 total time=   0.1s
[CV 5/5] END max_depth=None, min_samples_leaf=1, min_samples_split=2, n_es
timators=300;, score=0.871 total time=   0.4s
[CV 1/5] END max_depth=None, min_samples_leaf=1, min_samples_split=5, n_es
timators=200;, score=0.922 total time=   0.3s
[CV 3/5] END max_depth=None, min_samples_leaf=1, min_samples_split=5, n_es
timators=100;, score=0.946 total time=   0.2s
[CV 4/5] END max_depth=None, min_samples_leaf=1, min_samples_split=2, n_es
timators=300;, score=0.915 total time=   0.5s
[CV 3/5] END max_depth=None, min_samples_leaf=1, min_samples_split=5, n_es
timators=200;, score=0.947 total time=   0.3s
[CV 1/5] END max_depth=None, min_samples_leaf=1, min_samples_split=10, n_e
stimators=100;, score=0.916 total time=   0.1s
[CV 5/5] END max_depth=None, min_samples_leaf=1, min_samples_split=5, n_es
timators=200;, score=0.865 total time=   0.3s
[CV 3/5] END max_depth=None, min_samples_leaf=1, min_samples_split=10, n_e
stimators=100;, score=0.941 total time=   0.1s
[CV 5/5] END max_depth=None, min_samples_leaf=1, min_samples_split=10, n_e
stimators=100;, score=0.876 total time=   0.1s
[CV 2/5] END max_depth=None, min_samples_leaf=1, min_samples_split=10, n_e
stimators=100;, score=0.978 total time=   0.1s
[CV 2/5] END max_depth=None, min_samples_leaf=1, min_samples_split=5, n_es
timators=200;, score=0.973 total time=   0.3s
[CV 4/5] END max_depth=None, min_samples_leaf=1, min_samples_split=10, n_e
stimators=100;, score=0.916 total time=   0.1s
[CV 2/5] END max_depth=None, min_samples_leaf=1, min_samples_split=5, n_es
```

```
timators=300;, score=0.977 total time=   0.4s
[CV 4/5] END max_depth=None, min_samples_leaf=1, min_samples_split=5, n_es
timators=300;, score=0.915 total time=   0.4s
[CV 4/5] END max_depth=None, min_samples_leaf=1, min_samples_split=5, n_es
timators=200;, score=0.919 total time=   0.3s
[CV 1/5] END max_depth=None, min_samples_leaf=1, min_samples_split=10, n_e
stimators=200;, score=0.922 total time=   0.3s
[CV 1/5] END max_depth=None, min_samples_leaf=1, min_samples_split=5, n_es
timators=300;, score=0.919 total time=   0.4s
[CV 4/5] END max_depth=None, min_samples_leaf=1, min_samples_split=10, n_e
stimators=200;, score=0.909 total time=   0.3s
[CV 2/5] END max_depth=None, min_samples_leaf=1, min_samples_split=10, n_e
stimators=200;, score=0.977 total time=   0.4s
[CV 2/5] END max_depth=None, min_samples_leaf=2, min_samples_split=2, n_es
timators=100;, score=0.977 total time=   0.1s
[CV 1/5] END max_depth=None, min_samples_leaf=1, min_samples_split=10, n_e
stimators=300;, score=0.924 total time=   0.4s
[CV 3/5] END max_depth=None, min_samples_leaf=1, min_samples_split=5, n_es
timators=300;, score=0.944 total time=   0.4s
[CV 5/5] END max_depth=None, min_samples_leaf=1, min_samples_split=5, n_es
timators=300;, score=0.873 total time=   0.4s
[CV 3/5] END max_depth=None, min_samples_leaf=2, min_samples_split=2, n_es
timators=100;, score=0.947 total time=   0.1s
[CV 3/5] END max_depth=None, min_samples_leaf=1, min_samples_split=10, n_e
stimators=300;, score=0.941 total time=   0.4s
[CV 5/5] END max_depth=None, min_samples_leaf=1, min_samples_split=10, n_e
stimators=200;, score=0.869 total time=   0.2s
[CV 5/5] END max_depth=None, min_samples_leaf=1, min_samples_split=10, n_e
stimators=300;, score=0.873 total time=   0.4s
[CV 4/5] END max_depth=None, min_samples_leaf=2, min_samples_split=2, n_es
timators=100;, score=0.918 total time=   0.1s
[CV 3/5] END max_depth=None, min_samples_leaf=1, min_samples_split=10, n_e
stimators=200;, score=0.942 total time=   0.2s
[CV 1/5] END max_depth=None, min_samples_leaf=2, min_samples_split=2, n_es
timators=100;, score=0.927 total time=   0.1s
[CV 5/5] END max_depth=None, min_samples_leaf=2, min_samples_split=2, n_es
timators=100;, score=0.869 total time=   0.1s
[CV 1/5] END max_depth=None, min_samples_leaf=2, min_samples_split=2, n_es
timators=200;, score=0.923 total time=   0.3s
[CV 3/5] END max_depth=None, min_samples_leaf=2, min_samples_split=2, n_es
timators=200;, score=0.944 total time=   0.2s
[CV 2/5] END max_depth=None, min_samples_leaf=1, min_samples_split=10, n_e
stimators=300;, score=0.977 total time=   0.4s
[CV 5/5] END max_depth=None, min_samples_leaf=2, min_samples_split=2, n_es
timators=200;, score=0.872 total time=   0.2s
[CV 1/5] END max_depth=None, min_samples_leaf=2, min_samples_split=5, n_es
timators=100;, score=0.922 total time=   0.1s
[CV 4/5] END max_depth=None, min_samples_leaf=1, min_samples_split=10, n_e
stimators=300;, score=0.913 total time=   0.4s
[CV 3/5] END max_depth=None, min_samples_leaf=2, min_samples_split=5, n_es
timators=100;, score=0.949 total time=   0.1s
[CV 2/5] END max_depth=None, min_samples_leaf=2, min_samples_split=2, n_es
timators=300;, score=0.977 total time=   0.4s
[CV 2/5] END max_depth=None, min_samples_leaf=2, min_samples_split=2, n_es
timators=200;, score=0.980 total time=   0.2s
[CV 2/5] END max_depth=None, min_samples_leaf=2, min_samples_split=5, n_es
timators=100;, score=0.974 total time=   0.1s
[CV 5/5] END max_depth=None, min_samples_leaf=2, min_samples_split=5, n_es
timators=100;, score=0.871 total time=   0.1s
[CV 4/5] END max_depth=None, min_samples_leaf=2, min_samples_split=2, n_es
```

```
timators=200;, score=0.913 total time=   0.3s
[CV 4/5] END max_depth=None, min_samples_leaf=2, min_samples_split=2, n_es
timators=300;, score=0.912 total time=   0.4s
[CV 4/5] END max_depth=None, min_samples_leaf=2, min_samples_split=5, n_es
timators=100;, score=0.914 total time=   0.1s
[CV 1/5] END max_depth=None, min_samples_leaf=2, min_samples_split=2, n_es
timators=300;, score=0.920 total time=   0.4s
[CV 2/5] END max_depth=None, min_samples_leaf=2, min_samples_split=5, n_es
timators=200;, score=0.976 total time=   0.3s
[CV 4/5] END max_depth=None, min_samples_leaf=2, min_samples_split=5, n_es
timators=200;, score=0.915 total time=   0.3s
[CV 1/5] END max_depth=None, min_samples_leaf=2, min_samples_split=5, n_es
timators=200;, score=0.926 total time=   0.3s
[CV 3/5] END max_depth=None, min_samples_leaf=2, min_samples_split=2, n_es
timators=300;, score=0.943 total time=   0.4s
[CV 1/5] END max_depth=None, min_samples_leaf=2, min_samples_split=5, n_es
timators=300;, score=0.923 total time=   0.4s
[CV 2/5] END max_depth=None, min_samples_leaf=2, min_samples_split=10, n_e
stimators=100;, score=0.974 total time=   0.1s
[CV 5/5] END max_depth=None, min_samples_leaf=2, min_samples_split=2, n_es
timators=300;, score=0.872 total time=   0.4s
[CV 3/5] END max_depth=None, min_samples_leaf=2, min_samples_split=5, n_es
timators=300;, score=0.945 total time=   0.4s
[CV 4/5] END max_depth=None, min_samples_leaf=2, min_samples_split=10, n_e
stimators=100;, score=0.913 total time=   0.1s
[CV 5/5] END max_depth=None, min_samples_leaf=2, min_samples_split=5, n_es
timators=200;, score=0.873 total time=   0.2s
[CV 3/5] END max_depth=None, min_samples_leaf=2, min_samples_split=5, n_es
timators=200;, score=0.945 total time=   0.3s
[CV 5/5] END max_depth=None, min_samples_leaf=2, min_samples_split=5, n_es
timators=300;, score=0.871 total time=   0.4s
[CV 3/5] END max_depth=None, min_samples_leaf=2, min_samples_split=10, n_e
stimators=100;, score=0.942 total time=   0.1s
[CV 5/5] END max_depth=None, min_samples_leaf=2, min_samples_split=10, n_e
stimators=100;, score=0.870 total time=   0.1s
[CV 1/5] END max_depth=None, min_samples_leaf=2, min_samples_split=10, n_e
stimators=100;, score=0.930 total time=   0.1s
[CV 1/5] END max_depth=None, min_samples_leaf=2, min_samples_split=10, n_e
stimators=200;, score=0.926 total time=   0.3s
[CV 5/5] END max_depth=None, min_samples_leaf=2, min_samples_split=10, n_e
stimators=200;, score=0.871 total time=   0.2s
[CV 3/5] END max_depth=None, min_samples_leaf=2, min_samples_split=10, n_e
stimators=200;, score=0.943 total time=   0.3s
[CV 4/5] END max_depth=None, min_samples_leaf=2, min_samples_split=5, n_es
timators=300;, score=0.912 total time=   0.4s
[CV 1/5] END max_depth=None, min_samples_leaf=4, min_samples_split=2, n_es
timators=100;, score=0.924 total time=   0.1s
[CV 2/5] END max_depth=None, min_samples_leaf=2, min_samples_split=5, n_es
timators=300;, score=0.976 total time=   0.4s
[CV 2/5] END max_depth=None, min_samples_leaf=2, min_samples_split=10, n_e
stimators=200;, score=0.975 total time=   0.2s
[CV 2/5] END max_depth=None, min_samples_leaf=4, min_samples_split=2, n_es
timators=100;, score=0.980 total time=   0.1s
[CV 2/5] END max_depth=None, min_samples_leaf=2, min_samples_split=10, n_e
stimators=300;, score=0.977 total time=   0.4s
[CV 5/5] END max_depth=None, min_samples_leaf=4, min_samples_split=2, n_es
timators=100;, score=0.870 total time=   0.1s
[CV 3/5] END max_depth=None, min_samples_leaf=4, min_samples_split=2, n_es
timators=100;, score=0.945 total time=   0.1s
[CV 4/5] END max_depth=None, min_samples_leaf=2, min_samples_split=10, n_e
```

```
stimators=300;, score=0.910 total time=   0.4s
[CV 4/5] END max_depth=None, min_samples_leaf=2, min_samples_split=10, n_e
stimators=200;, score=0.909 total time=   0.3s
[CV 1/5] END max_depth=None, min_samples_leaf=2, min_samples_split=10, n_e
stimators=300;, score=0.924 total time=   0.4s
[CV 4/5] END max_depth=None, min_samples_leaf=4, min_samples_split=2, n_es
timators=100;, score=0.912 total time=   0.1s
[CV 2/5] END max_depth=None, min_samples_leaf=4, min_samples_split=2, n_es
timators=200;, score=0.981 total time=   0.3s
[CV 4/5] END max_depth=None, min_samples_leaf=4, min_samples_split=2, n_es
timators=200;, score=0.911 total time=   0.3s
[CV 1/5] END max_depth=None, min_samples_leaf=4, min_samples_split=2, n_es
timators=200;, score=0.926 total time=   0.3s
[CV 3/5] END max_depth=None, min_samples_leaf=2, min_samples_split=10, n_e
stimators=300;, score=0.945 total time=   0.4s
[CV 2/5] END max_depth=None, min_samples_leaf=4, min_samples_split=5, n_es
timators=100;, score=0.981 total time=   0.1s
[CV 5/5] END max_depth=None, min_samples_leaf=2, min_samples_split=10, n_e
stimators=300;, score=0.872 total time=   0.4s
[CV 1/5] END max_depth=None, min_samples_leaf=4, min_samples_split=2, n_es
timators=300;, score=0.925 total time=   0.4s
[CV 3/5] END max_depth=None, min_samples_leaf=4, min_samples_split=2, n_es
timators=200;, score=0.941 total time=   0.3s
[CV 5/5] END max_depth=None, min_samples_leaf=4, min_samples_split=2, n_es
timators=300;, score=0.871 total time=   0.3s
[CV 5/5] END max_depth=None, min_samples_leaf=4, min_samples_split=2, n_es
timators=200;, score=0.872 total time=   0.3s
[CV 4/5] END max_depth=None, min_samples_leaf=4, min_samples_split=5, n_es
timators=100;, score=0.906 total time=   0.1s
[CV 3/5] END max_depth=None, min_samples_leaf=4, min_samples_split=5, n_es
timators=100;, score=0.943 total time=   0.1s
[CV 3/5] END max_depth=None, min_samples_leaf=4, min_samples_split=2, n_es
timators=300;, score=0.946 total time=   0.4s
[CV 1/5] END max_depth=None, min_samples_leaf=4, min_samples_split=5, n_es
timators=100;, score=0.925 total time=   0.1s
[CV 5/5] END max_depth=None, min_samples_leaf=4, min_samples_split=5, n_es
timators=100;, score=0.870 total time=   0.1s
[CV 1/5] END max_depth=None, min_samples_leaf=4, min_samples_split=5, n_es
timators=200;, score=0.926 total time=   0.2s
[CV 3/5] END max_depth=None, min_samples_leaf=4, min_samples_split=5, n_es
timators=200;, score=0.941 total time=   0.2s
[CV 5/5] END max_depth=None, min_samples_leaf=4, min_samples_split=5, n_es
timators=200;, score=0.875 total time=   0.2s
[CV 1/5] END max_depth=None, min_samples_leaf=4, min_samples_split=10, n_e
stimators=100;, score=0.928 total time=   0.1s
[CV 2/5] END max_depth=None, min_samples_leaf=4, min_samples_split=2, n_es
timators=300;, score=0.979 total time=   0.4s
[CV 2/5] END max_depth=None, min_samples_leaf=4, min_samples_split=10, n_e
stimators=100;, score=0.979 total time=   0.1s
[CV 2/5] END max_depth=None, min_samples_leaf=4, min_samples_split=5, n_es
timators=300;, score=0.979 total time=   0.4s
[CV 4/5] END max_depth=None, min_samples_leaf=4, min_samples_split=2, n_es
timators=300;, score=0.909 total time=   0.4s
[CV 3/5] END max_depth=None, min_samples_leaf=4, min_samples_split=10, n_e
stimators=100;, score=0.943 total time=   0.1s[CV 4/5] END max_depth=None,
min_samples_leaf=4, min_samples_split=5, n_estimators=200;, score=0.909 to
tal time=   0.2s

[CV 2/5] END max_depth=None, min_samples_leaf=4, min_samples_split=5, n_es
timators=200;, score=0.979 total time=   0.3s
```

```
[CV 5/5] END max_depth=None, min_samples_leaf=4, min_samples_split=10, n_e
stimators=100;, score=0.871 total time=   0.1s
[CV 4/5] END max_depth=None, min_samples_leaf=4, min_samples_split=5, n_es
timators=300;, score=0.908 total time=   0.4s
[CV 4/5] END max_depth=None, min_samples_leaf=4, min_samples_split=10, n_e
stimators=100;, score=0.915 total time=   0.1s
[CV 1/5] END max_depth=None, min_samples_leaf=4, min_samples_split=5, n_es
timators=300;, score=0.925 total time=   0.4s
[CV 4/5] END max_depth=None, min_samples_leaf=4, min_samples_split=10, n_e
stimators=200;, score=0.908 total time=   0.2s
[CV 1/5] END max_depth=None, min_samples_leaf=4, min_samples_split=10, n_e
stimators=200;, score=0.922 total time=   0.2s
[CV 3/5] END max_depth=None, min_samples_leaf=4, min_samples_split=5, n_es
timators=300;, score=0.944 total time=   0.3s
[CV 2/5] END max_depth=None, min_samples_leaf=4, min_samples_split=10, n_e
stimators=200;, score=0.979 total time=   0.3s
[CV 1/5] END max_depth=None, min_samples_leaf=4, min_samples_split=10, n_e
stimators=300;, score=0.929 total time=   0.3s
[CV 2/5] END max_depth=10, min_samples_leaf=1, min_samples_split=2, n_esti
mators=100;, score=0.971 total time=   0.2s
[CV 4/5] END max_depth=10, min_samples_leaf=1, min_samples_split=2, n_esti
mators=100;, score=0.916 total time=   0.1s
[CV 5/5] END max_depth=None, min_samples_leaf=4, min_samples_split=10, n_e
stimators=200;, score=0.873 total time=   0.2s
[CV 5/5] END max_depth=None, min_samples_leaf=4, min_samples_split=5, n_es
timators=300;, score=0.874 total time=   0.4s
[CV 3/5] END max_depth=None, min_samples_leaf=4, min_samples_split=10, n_e
stimators=300;, score=0.946 total time=   0.4s
[CV 3/5] END max_depth=10, min_samples_leaf=1, min_samples_split=2, n_esti
mators=100;, score=0.947 total time=   0.1s
[CV 5/5] END max_depth=None, min_samples_leaf=4, min_samples_split=10, n_e
stimators=300;, score=0.873 total time=   0.4s
[CV 3/5] END max_depth=None, min_samples_leaf=4, min_samples_split=10, n_e
stimators=200;, score=0.942 total time=   0.3s
[CV 5/5] END max_depth=10, min_samples_leaf=1, min_samples_split=2, n_esti
mators=100;, score=0.869 total time=   0.2s
[CV 2/5] END max_depth=None, min_samples_leaf=4, min_samples_split=10, n_e
stimators=300;, score=0.978 total time=   0.3s
[CV 1/5] END max_depth=10, min_samples_leaf=1, min_samples_split=2, n_esti
mators=200;, score=0.919 total time=   0.3s
[CV 1/5] END max_depth=10, min_samples_leaf=1, min_samples_split=2, n_esti
mators=100;, score=0.921 total time=   0.2s
[CV 3/5] END max_depth=10, min_samples_leaf=1, min_samples_split=2, n_esti
mators=200;, score=0.944 total time=   0.3s
[CV 5/5] END max_depth=10, min_samples_leaf=1, min_samples_split=2, n_esti
mators=200;, score=0.870 total time=   0.3s
[CV 1/5] END max_depth=10, min_samples_leaf=1, min_samples_split=5, n_esti
mators=100;, score=0.925 total time=   0.1s
[CV 3/5] END max_depth=10, min_samples_leaf=1, min_samples_split=5, n_esti
mators=100;, score=0.943 total time=   0.1s
[CV 4/5] END max_depth=None, min_samples_leaf=4, min_samples_split=10, n_e
stimators=300;, score=0.910 total time=   0.4s
[CV 2/5] END max_depth=10, min_samples_leaf=1, min_samples_split=5, n_esti
mators=100;, score=0.975 total time=   0.1s
[CV 2/5] END max_depth=10, min_samples_leaf=1, min_samples_split=2, n_esti
mators=200;, score=0.975 total time=   0.3s
[CV 4/5] END max_depth=10, min_samples_leaf=1, min_samples_split=5, n_esti
mators=100;, score=0.914 total time=   0.1s
[CV 4/5] END max_depth=10, min_samples_leaf=1, min_samples_split=2, n_esti
mators=200;, score=0.915 total time=   0.3s
```

```
[CV 4/5] END max_depth=10, min_samples_leaf=1, min_samples_split=2, n_esti
mators=300;, score=0.916 total time=   0.4s
[CV 2/5] END max_depth=10, min_samples_leaf=1, min_samples_split=2, n_esti
mators=300;, score=0.976 total time=   0.4s
[CV 5/5] END max_depth=10, min_samples_leaf=1, min_samples_split=5, n_esti
mators=100;, score=0.864 total time=   0.1s
[CV 2/5] END max_depth=10, min_samples_leaf=1, min_samples_split=5, n_esti
mators=200;, score=0.973 total time=   0.3s
[CV 1/5] END max_depth=10, min_samples_leaf=1, min_samples_split=2, n_esti
mators=300;, score=0.923 total time=   0.4s
[CV 4/5] END max_depth=10, min_samples_leaf=1, min_samples_split=5, n_esti
mators=200;, score=0.916 total time=   0.3s
[CV 1/5] END max_depth=10, min_samples_leaf=1, min_samples_split=5, n_esti
mators=200;, score=0.923 total time=   0.3s
[CV 5/5] END max_depth=10, min_samples_leaf=1, min_samples_split=2, n_esti
mators=300;, score=0.874 total time=   0.4s
[CV 1/5] END max_depth=10, min_samples_leaf=1, min_samples_split=5, n_esti
mators=300;, score=0.924 total time=   0.5s
[CV 3/5] END max_depth=10, min_samples_leaf=1, min_samples_split=2, n_esti
mators=300;, score=0.944 total time=   0.4s
[CV 3/5] END max_depth=10, min_samples_leaf=1, min_samples_split=5, n_esti
mators=300;, score=0.944 total time=   0.5s
[CV 3/5] END max_depth=10, min_samples_leaf=1, min_samples_split=5, n_esti
mators=200;, score=0.942 total time=   0.3s
[CV 2/5] END max_depth=10, min_samples_leaf=1, min_samples_split=10, n_est
imators=100;, score=0.976 total time=   0.1s
[CV 5/5] END max_depth=10, min_samples_leaf=1, min_samples_split=5, n_esti
mators=200;, score=0.875 total time=   0.3s
[CV 4/5] END max_depth=10, min_samples_leaf=1, min_samples_split=10, n_est
imators=100;, score=0.916 total time=   0.2s
[CV 5/5] END max_depth=10, min_samples_leaf=1, min_samples_split=5, n_esti
mators=300;, score=0.870 total time=   0.4s
[CV 3/5] END max_depth=10, min_samples_leaf=1, min_samples_split=10, n_est
imators=100;, score=0.943 total time=   0.1s
[CV 1/5] END max_depth=10, min_samples_leaf=1, min_samples_split=10, n_est
imators=200;, score=0.923 total time=   0.3s
[CV 5/5] END max_depth=10, min_samples_leaf=1, min_samples_split=10, n_est
imators=100;, score=0.875 total time=   0.1s
[CV 5/5] END max_depth=10, min_samples_leaf=1, min_samples_split=10, n_est
imators=200;, score=0.876 total time=   0.2s
[CV 3/5] END max_depth=10, min_samples_leaf=1, min_samples_split=10, n_est
imators=200;, score=0.944 total time=   0.3s
[CV 1/5] END max_depth=10, min_samples_leaf=1, min_samples_split=10, n_est
imators=100;, score=0.922 total time=   0.2s
[CV 2/5] END max_depth=10, min_samples_leaf=1, min_samples_split=5, n_esti
mators=300;, score=0.975 total time=   0.4s
[CV 4/5] END max_depth=10, min_samples_leaf=1, min_samples_split=5, n_esti
mators=300;, score=0.912 total time=   0.4s
[CV 3/5] END max_depth=10, min_samples_leaf=2, min_samples_split=2, n_esti
mators=100;, score=0.951 total time=   0.1s
[CV 1/5] END max_depth=10, min_samples_leaf=2, min_samples_split=2, n_esti
mators=100;, score=0.920 total time=   0.2s
[CV 4/5] END max_depth=10, min_samples_leaf=1, min_samples_split=10, n_est
imators=200;, score=0.911 total time=   0.2s
[CV 5/5] END max_depth=10, min_samples_leaf=2, min_samples_split=2, n_esti
mators=100;, score=0.870 total time=   0.1s
[CV 2/5] END max_depth=10, min_samples_leaf=1, min_samples_split=10, n_est
imators=200;, score=0.977 total time=   0.3s
[CV 2/5] END max_depth=10, min_samples_leaf=1, min_samples_split=10, n_est
imators=300;, score=0.979 total time=   0.4s
```

```
[CV 4/5] END max_depth=10, min_samples_leaf=2, min_samples_split=2, n_esti
mators=100;, score=0.916 total time=   0.1s
[CV 2/5] END max_depth=10, min_samples_leaf=2, min_samples_split=2, n_esti
mators=100;, score=0.975 total time=   0.1s
[CV 4/5] END max_depth=10, min_samples_leaf=1, min_samples_split=10, n_est
imators=300;, score=0.913 total time=   0.4s
[CV 1/5] END max_depth=10, min_samples_leaf=1, min_samples_split=10, n_est
imators=300;, score=0.923 total time=   0.4s
[CV 2/5] END max_depth=10, min_samples_leaf=2, min_samples_split=2, n_esti
mators=200;, score=0.976 total time=   0.2s
[CV 1/5] END max_depth=10, min_samples_leaf=2, min_samples_split=2, n_esti
mators=200;, score=0.921 total time=   0.2s
[CV 4/5] END max_depth=10, min_samples_leaf=2, min_samples_split=2, n_esti
mators=200;, score=0.916 total time=   0.3s
[CV 2/5] END max_depth=10, min_samples_leaf=2, min_samples_split=5, n_esti
mators=100;, score=0.979 total time=   0.1s
[CV 3/5] END max_depth=10, min_samples_leaf=2, min_samples_split=2, n_esti
mators=300;, score=0.944 total time=   0.4s
[CV 3/5] END max_depth=10, min_samples_leaf=1, min_samples_split=10, n_est
imators=300;, score=0.943 total time=   0.4s
[CV 3/5] END max_depth=10, min_samples_leaf=2, min_samples_split=2, n_esti
mators=200;, score=0.944 total time=   0.3s
[CV 5/5] END max_depth=10, min_samples_leaf=1, min_samples_split=10, n_est
imators=300;, score=0.870 total time=   0.4s
[CV 1/5] END max_depth=10, min_samples_leaf=2, min_samples_split=2, n_esti
mators=300;, score=0.926 total time=   0.4s
[CV 3/5] END max_depth=10, min_samples_leaf=2, min_samples_split=5, n_esti
mators=100;, score=0.943 total time=   0.1s
[CV 5/5] END max_depth=10, min_samples_leaf=2, min_samples_split=2, n_esti
mators=200;, score=0.868 total time=   0.2s
[CV 5/5] END max_depth=10, min_samples_leaf=2, min_samples_split=2, n_esti
mators=300;, score=0.869 total time=   0.4s
[CV 4/5] END max_depth=10, min_samples_leaf=2, min_samples_split=5, n_esti
mators=100;, score=0.910 total time=   0.1s
[CV 1/5] END max_depth=10, min_samples_leaf=2, min_samples_split=5, n_esti
mators=100;, score=0.924 total time=   0.2s
[CV 5/5] END max_depth=10, min_samples_leaf=2, min_samples_split=5, n_esti
mators=100;, score=0.875 total time=   0.1s
[CV 1/5] END max_depth=10, min_samples_leaf=2, min_samples_split=5, n_esti
mators=200;, score=0.925 total time=   0.3s
[CV 5/5] END max_depth=10, min_samples_leaf=2, min_samples_split=5, n_esti
mators=200;, score=0.870 total time=   0.3s
[CV 3/5] END max_depth=10, min_samples_leaf=2, min_samples_split=5, n_esti
mators=200;, score=0.946 total time=   0.3s
[CV 4/5] END max_depth=10, min_samples_leaf=2, min_samples_split=2, n_esti
mators=300;, score=0.914 total time=   0.4s
[CV 1/5] END max_depth=10, min_samples_leaf=2, min_samples_split=10, n_est
imators=100;, score=0.925 total time=   0.1s
[CV 2/5] END max_depth=10, min_samples_leaf=2, min_samples_split=2, n_esti
mators=300;, score=0.975 total time=   0.5s
[CV 2/5] END max_depth=10, min_samples_leaf=2, min_samples_split=5, n_esti
mators=300;, score=0.975 total time=   0.5s
[CV 5/5] END max_depth=10, min_samples_leaf=2, min_samples_split=10, n_est
imators=100;, score=0.872 total time=   0.1s
[CV 2/5] END max_depth=10, min_samples_leaf=2, min_samples_split=10, n_est
imators=100;, score=0.979 total time=   0.2s
[CV 3/5] END max_depth=10, min_samples_leaf=2, min_samples_split=10, n_est
imators=100;, score=0.945 total time=   0.2s
[CV 2/5] END max_depth=10, min_samples_leaf=2, min_samples_split=5, n_esti
mators=200;, score=0.976 total time=   0.3s
```

```
[CV 4/5] END max_depth=10, min_samples_leaf=2, min_samples_split=5, n_esti
mators=300;, score=0.914 total time=   0.4s
[CV 4/5] END max_depth=10, min_samples_leaf=2, min_samples_split=5, n_esti
mators=200;, score=0.908 total time=   0.3s
[CV 4/5] END max_depth=10, min_samples_leaf=2, min_samples_split=10, n_est
imators=100;, score=0.911 total time=   0.1s
[CV 1/5] END max_depth=10, min_samples_leaf=2, min_samples_split=5, n_esti
mators=300;, score=0.924 total time=   0.4s
[CV 2/5] END max_depth=10, min_samples_leaf=2, min_samples_split=10, n_est
imators=200;, score=0.976 total time=   0.3s
[CV 1/5] END max_depth=10, min_samples_leaf=2, min_samples_split=10, n_est
imators=200;, score=0.926 total time=   0.3s[CV 4/5] END max_depth=10, min
_samples_leaf=2, min_samples_split=10, n_estimators=200;, score=0.911 tota
l time=   0.3s

[CV 3/5] END max_depth=10, min_samples_leaf=2, min_samples_split=5, n_esti
mators=300;, score=0.948 total time=   0.4s
[CV 2/5] END max_depth=10, min_samples_leaf=4, min_samples_split=2, n_esti
mators=100;, score=0.979 total time=   0.1s
[CV 5/5] END max_depth=10, min_samples_leaf=2, min_samples_split=5, n_esti
mators=300;, score=0.872 total time=   0.5s
[CV 1/5] END max_depth=10, min_samples_leaf=2, min_samples_split=10, n_est
imators=300;, score=0.923 total time=   0.4s
[CV 4/5] END max_depth=10, min_samples_leaf=4, min_samples_split=2, n_esti
mators=100;, score=0.911 total time=   0.2s
[CV 3/5] END max_depth=10, min_samples_leaf=2, min_samples_split=10, n_est
imators=300;, score=0.944 total time=   0.5s
[CV 5/5] END max_depth=10, min_samples_leaf=2, min_samples_split=10, n_est
imators=300;, score=0.871 total time=   0.4s
[CV 3/5] END max_depth=10, min_samples_leaf=4, min_samples_split=2, n_esti
mators=100;, score=0.943 total time=   0.1s
[CV 5/5] END max_depth=10, min_samples_leaf=2, min_samples_split=10, n_est
imators=200;, score=0.868 total time=   0.3s
[CV 3/5] END max_depth=10, min_samples_leaf=2, min_samples_split=10, n_est
imators=200;, score=0.942 total time=   0.3s
[CV 5/5] END max_depth=10, min_samples_leaf=4, min_samples_split=2, n_esti
mators=100;, score=0.872 total time=   0.1s
[CV 1/5] END max_depth=10, min_samples_leaf=4, min_samples_split=2, n_esti
mators=100;, score=0.925 total time=   0.1s
[CV 1/5] END max_depth=10, min_samples_leaf=4, min_samples_split=2, n_esti
mators=200;, score=0.920 total time=   0.3s
[CV 3/5] END max_depth=10, min_samples_leaf=4, min_samples_split=2, n_esti
mators=200;, score=0.944 total time=   0.3s
[CV 1/5] END max_depth=10, min_samples_leaf=4, min_samples_split=5, n_esti
mators=100;, score=0.928 total time=   0.2s
[CV 5/5] END max_depth=10, min_samples_leaf=4, min_samples_split=2, n_esti
mators=200;, score=0.872 total time=   0.3s
[CV 2/5] END max_depth=10, min_samples_leaf=2, min_samples_split=10, n_est
imators=300;, score=0.978 total time=   0.4s
[CV 4/5] END max_depth=10, min_samples_leaf=2, min_samples_split=10, n_est
imators=300;, score=0.909 total time=   0.4s
[CV 2/5] END max_depth=10, min_samples_leaf=4, min_samples_split=5, n_esti
mators=100;, score=0.979 total time=   0.1s
[CV 2/5] END max_depth=10, min_samples_leaf=4, min_samples_split=2, n_esti
mators=300;, score=0.977 total time=   0.4s
[CV 3/5] END max_depth=10, min_samples_leaf=4, min_samples_split=5, n_esti
mators=100;, score=0.941 total time=   0.1s
[CV 4/5] END max_depth=10, min_samples_leaf=4, min_samples_split=2, n_esti
mators=200;, score=0.908 total time=   0.2s
[CV 4/5] END max_depth=10, min_samples_leaf=4, min_samples_split=2, n_esti
```

```
mators=300;, score=0.912 total time=   0.4s
[CV 2/5] END max_depth=10, min_samples_leaf=4, min_samples_split=2, n_esti
mators=200;, score=0.979 total time=   0.3s
[CV 5/5] END max_depth=10, min_samples_leaf=4, min_samples_split=5, n_esti
mators=100;, score=0.872 total time=   0.1s
[CV 4/5] END max_depth=10, min_samples_leaf=4, min_samples_split=5, n_esti
mators=100;, score=0.911 total time=   0.2s
[CV 2/5] END max_depth=10, min_samples_leaf=4, min_samples_split=5, n_esti
mators=200;, score=0.981 total time=   0.2s
[CV 1/5] END max_depth=10, min_samples_leaf=4, min_samples_split=2, n_esti
mators=300;, score=0.925 total time=   0.4s
[CV 4/5] END max_depth=10, min_samples_leaf=4, min_samples_split=5, n_esti
mators=200;, score=0.911 total time=   0.3s
[CV 1/5] END max_depth=10, min_samples_leaf=4, min_samples_split=5, n_esti
mators=200;, score=0.926 total time=   0.3s
[CV 3/5] END max_depth=10, min_samples_leaf=4, min_samples_split=2, n_esti
mators=300;, score=0.943 total time=   0.4s
[CV 1/5] END max_depth=10, min_samples_leaf=4, min_samples_split=5, n_esti
mators=300;, score=0.926 total time=   0.3s
[CV 5/5] END max_depth=10, min_samples_leaf=4, min_samples_split=2, n_esti
mators=300;, score=0.868 total time=   0.4s
[CV 2/5] END max_depth=10, min_samples_leaf=4, min_samples_split=10, n_est
imators=100;, score=0.980 total time=   0.1s
[CV 3/5] END max_depth=10, min_samples_leaf=4, min_samples_split=5, n_esti
mators=200;, score=0.942 total time=   0.2s
[CV 4/5] END max_depth=10, min_samples_leaf=4, min_samples_split=10, n_est
imators=100;, score=0.916 total time=   0.1s
[CV 3/5] END max_depth=10, min_samples_leaf=4, min_samples_split=5, n_esti
mators=300;, score=0.945 total time=   0.4s
[CV 3/5] END max_depth=10, min_samples_leaf=4, min_samples_split=10, n_est
imators=100;, score=0.948 total time=   0.1s
[CV 5/5] END max_depth=10, min_samples_leaf=4, min_samples_split=5, n_esti
mators=200;, score=0.872 total time=   0.3s
[CV 5/5] END max_depth=10, min_samples_leaf=4, min_samples_split=5, n_esti
mators=300;, score=0.873 total time=   0.4s
[CV 5/5] END max_depth=10, min_samples_leaf=4, min_samples_split=10, n_est
imators=100;, score=0.873 total time=   0.2s
[CV 1/5] END max_depth=10, min_samples_leaf=4, min_samples_split=10, n_est
imators=200;, score=0.925 total time=   0.3s
[CV 1/5] END max_depth=10, min_samples_leaf=4, min_samples_split=10, n_est
imators=100;, score=0.923 total time=   0.1s
[CV 3/5] END max_depth=10, min_samples_leaf=4, min_samples_split=10, n_est
imators=200;, score=0.943 total time=   0.3s
[CV 2/5] END max_depth=10, min_samples_leaf=4, min_samples_split=5, n_esti
mators=300;, score=0.979 total time=   0.4s
[CV 5/5] END max_depth=10, min_samples_leaf=4, min_samples_split=10, n_est
imators=200;, score=0.872 total time=   0.2s
[CV 1/5] END max_depth=20, min_samples_leaf=1, min_samples_split=2, n_esti
mators=100;, score=0.921 total time=   0.1s
[CV 3/5] END max_depth=20, min_samples_leaf=1, min_samples_split=2, n_esti
mators=100;, score=0.941 total time=   0.1s
[CV 2/5] END max_depth=10, min_samples_leaf=4, min_samples_split=10, n_est
imators=200;, score=0.976 total time=   0.3s
[CV 4/5] END max_depth=10, min_samples_leaf=4, min_samples_split=5, n_esti
mators=300;, score=0.909 total time=   0.4s
[CV 2/5] END max_depth=10, min_samples_leaf=4, min_samples_split=10, n_est
imators=300;, score=0.980 total time=   0.4s
[CV 4/5] END max_depth=10, min_samples_leaf=4, min_samples_split=10, n_est
imators=200;, score=0.908 total time=   0.2s
[CV 4/5] END max_depth=10, min_samples_leaf=4, min_samples_split=10, n_est
```

```
imators=300;, score=0.910 total time=   0.4s
[CV 4/5] END max_depth=20, min_samples_leaf=1, min_samples_split=2, n_esti
mators=100;, score=0.915 total time=   0.1s
[CV 2/5] END max_depth=20, min_samples_leaf=1, min_samples_split=2, n_esti
mators=100;, score=0.973 total time=   0.2s
[CV 5/5] END max_depth=20, min_samples_leaf=1, min_samples_split=2, n_esti
mators=100;, score=0.863 total time=   0.1s
[CV 1/5] END max_depth=10, min_samples_leaf=4, min_samples_split=10, n_est
imators=300;, score=0.926 total time=   0.4s
[CV 2/5] END max_depth=20, min_samples_leaf=1, min_samples_split=2, n_esti
mators=200;, score=0.969 total time=   0.2s
[CV 4/5] END max_depth=20, min_samples_leaf=1, min_samples_split=2, n_esti
mators=200;, score=0.914 total time=   0.3s
[CV 1/5] END max_depth=20, min_samples_leaf=1, min_samples_split=2, n_esti
mators=200;, score=0.922 total time=   0.2s
[CV 3/5] END max_depth=10, min_samples_leaf=4, min_samples_split=10, n_est
imators=300;, score=0.946 total time=   0.4s
[CV 5/5] END max_depth=10, min_samples_leaf=4, min_samples_split=10, n_est
imators=300;, score=0.870 total time=   0.4s
[CV 3/5] END max_depth=20, min_samples_leaf=1, min_samples_split=2, n_esti
mators=300;, score=0.943 total time=   0.4s
[CV 1/5] END max_depth=20, min_samples_leaf=1, min_samples_split=2, n_esti
mators=300;, score=0.921 total time=   0.4s
[CV 2/5] END max_depth=20, min_samples_leaf=1, min_samples_split=5, n_esti
mators=100;, score=0.978 total time=   0.1s
[CV 3/5] END max_depth=20, min_samples_leaf=1, min_samples_split=2, n_esti
mators=200;, score=0.938 total time=   0.2s
[CV 4/5] END max_depth=20, min_samples_leaf=1, min_samples_split=5, n_esti
mators=100;, score=0.910 total time=   0.1s
[CV 5/5] END max_depth=20, min_samples_leaf=1, min_samples_split=2, n_esti
mators=200;, score=0.870 total time=   0.2s
[CV 5/5] END max_depth=20, min_samples_leaf=1, min_samples_split=2, n_esti
mators=300;, score=0.874 total time=   0.4s
[CV 3/5] END max_depth=20, min_samples_leaf=1, min_samples_split=5, n_esti
mators=100;, score=0.945 total time=   0.1s
[CV 5/5] END max_depth=20, min_samples_leaf=1, min_samples_split=5, n_esti
mators=100;, score=0.867 total time=   0.1s
[CV 1/5] END max_depth=20, min_samples_leaf=1, min_samples_split=5, n_esti
mators=200;, score=0.921 total time=   0.2s
[CV 1/5] END max_depth=20, min_samples_leaf=1, min_samples_split=5, n_esti
mators=100;, score=0.918 total time=   0.1s
[CV 3/5] END max_depth=20, min_samples_leaf=1, min_samples_split=5, n_esti
mators=200;, score=0.942 total time=   0.3s
[CV 5/5] END max_depth=20, min_samples_leaf=1, min_samples_split=5, n_esti
mators=200;, score=0.872 total time=   0.2s
[CV 1/5] END max_depth=20, min_samples_leaf=1, min_samples_split=10, n_est
imators=100;, score=0.923 total time=   0.1s
[CV 4/5] END max_depth=20, min_samples_leaf=1, min_samples_split=2, n_esti
mators=300;, score=0.917 total time=   0.4s
[CV 2/5] END max_depth=20, min_samples_leaf=1, min_samples_split=2, n_esti
mators=300;, score=0.971 total time=   0.4s
[CV 2/5] END max_depth=20, min_samples_leaf=1, min_samples_split=5, n_esti
mators=200;, score=0.973 total time=   0.3s
[CV 2/5] END max_depth=20, min_samples_leaf=1, min_samples_split=10, n_est
imators=100;, score=0.974 total time=   0.1s
[CV 2/5] END max_depth=20, min_samples_leaf=1, min_samples_split=5, n_esti
mators=300;, score=0.975 total time=   0.4s
[CV 3/5] END max_depth=20, min_samples_leaf=1, min_samples_split=10, n_est
imators=100;, score=0.939 total time=   0.1s
[CV 5/5] END max_depth=20, min_samples_leaf=1, min_samples_split=10, n_est
```

```
imators=100;, score=0.865 total time=   0.1s
[CV 4/5] END max_depth=20, min_samples_leaf=1, min_samples_split=5, n_esti
mators=200;, score=0.917 total time=   0.2s
[CV 4/5] END max_depth=20, min_samples_leaf=1, min_samples_split=5, n_esti
mators=300;, score=0.913 total time=   0.4s
[CV 4/5] END max_depth=20, min_samples_leaf=1, min_samples_split=10, n_est
imators=100;, score=0.914 total time=   0.1s
[CV 1/5] END max_depth=20, min_samples_leaf=1, min_samples_split=5, n_esti
mators=300;, score=0.923 total time=   0.4s
[CV 2/5] END max_depth=20, min_samples_leaf=1, min_samples_split=10, n_est
imators=200;, score=0.980 total time=   0.3s
[CV 4/5] END max_depth=20, min_samples_leaf=1, min_samples_split=10, n_est
imators=200;, score=0.911 total time=   0.2s
[CV 1/5] END max_depth=20, min_samples_leaf=1, min_samples_split=10, n_est
imators=200;, score=0.924 total time=   0.2s
[CV 5/5] END max_depth=20, min_samples_leaf=1, min_samples_split=5, n_esti
mators=300;, score=0.870 total time=   0.3s
[CV 1/5] END max_depth=20, min_samples_leaf=1, min_samples_split=10, n_est
imators=300;, score=0.924 total time=   0.4s
[CV 2/5] END max_depth=20, min_samples_leaf=2, min_samples_split=2, n_esti
mators=100;, score=0.979 total time=   0.1s
[CV 3/5] END max_depth=20, min_samples_leaf=1, min_samples_split=5, n_esti
mators=300;, score=0.944 total time=   0.4s
[CV 3/5] END max_depth=20, min_samples_leaf=1, min_samples_split=10, n_est
imators=200;, score=0.941 total time=   0.2s
[CV 5/5] END max_depth=20, min_samples_leaf=1, min_samples_split=10, n_est
imators=200;, score=0.872 total time=   0.2s
[CV 3/5] END max_depth=20, min_samples_leaf=1, min_samples_split=10, n_est
imators=300;, score=0.945 total time=   0.4s
[CV 3/5] END max_depth=20, min_samples_leaf=2, min_samples_split=2, n_esti
mators=100;, score=0.943 total time=   0.1s
[CV 4/5] END max_depth=20, min_samples_leaf=2, min_samples_split=2, n_esti
mators=100;, score=0.910 total time=   0.1s
[CV 5/5] END max_depth=20, min_samples_leaf=1, min_samples_split=10, n_est
imators=300;, score=0.873 total time=   0.4s
[CV 5/5] END max_depth=20, min_samples_leaf=2, min_samples_split=2, n_esti
mators=100;, score=0.872 total time=   0.1s
[CV 1/5] END max_depth=20, min_samples_leaf=2, min_samples_split=2, n_esti
mators=200;, score=0.925 total time=   0.2s
[CV 1/5] END max_depth=20, min_samples_leaf=2, min_samples_split=2, n_esti
mators=100;, score=0.925 total time=   0.1s
[CV 3/5] END max_depth=20, min_samples_leaf=2, min_samples_split=2, n_esti
mators=200;, score=0.945 total time=   0.3s
[CV 5/5] END max_depth=20, min_samples_leaf=2, min_samples_split=2, n_esti
mators=200;, score=0.870 total time=   0.2s
[CV 2/5] END max_depth=20, min_samples_leaf=1, min_samples_split=10, n_est
imators=300;, score=0.977 total time=   0.4s
[CV 1/5] END max_depth=20, min_samples_leaf=2, min_samples_split=5, n_esti
mators=100;, score=0.927 total time=   0.1s
[CV 4/5] END max_depth=20, min_samples_leaf=1, min_samples_split=10, n_est
imators=300;, score=0.911 total time=   0.3s
[CV 2/5] END max_depth=20, min_samples_leaf=2, min_samples_split=2, n_esti
mators=300;, score=0.977 total time=   0.4s
[CV 3/5] END max_depth=20, min_samples_leaf=2, min_samples_split=5, n_esti
mators=100;, score=0.942 total time=   0.1s
[CV 2/5] END max_depth=20, min_samples_leaf=2, min_samples_split=2, n_esti
mators=200;, score=0.977 total time=   0.2s
[CV 2/5] END max_depth=20, min_samples_leaf=2, min_samples_split=5, n_esti
mators=100;, score=0.972 total time=   0.1s
[CV 4/5] END max_depth=20, min_samples_leaf=2, min_samples_split=2, n_esti
```

```
mators=200;, score=0.912 total time=   0.2s
[CV 5/5] END max_depth=20, min_samples_leaf=2, min_samples_split=5, n_esti
mators=100;, score=0.867 total time=   0.1s
[CV 4/5] END max_depth=20, min_samples_leaf=2, min_samples_split=5, n_esti
mators=100;, score=0.911 total time=   0.1s
[CV 4/5] END max_depth=20, min_samples_leaf=2, min_samples_split=2, n_esti
mators=300;, score=0.914 total time=   0.4s
[CV 1/5] END max_depth=20, min_samples_leaf=2, min_samples_split=2, n_esti
mators=300;, score=0.924 total time=   0.3s
[CV 2/5] END max_depth=20, min_samples_leaf=2, min_samples_split=5, n_esti
mators=200;, score=0.975 total time=   0.3s
[CV 4/5] END max_depth=20, min_samples_leaf=2, min_samples_split=5, n_esti
mators=200;, score=0.915 total time=   0.3s
[CV 1/5] END max_depth=20, min_samples_leaf=2, min_samples_split=5, n_esti
mators=200;, score=0.926 total time=   0.3s
[CV 3/5] END max_depth=20, min_samples_leaf=2, min_samples_split=2, n_esti
mators=300;, score=0.943 total time=   0.4s
[CV 1/5] END max_depth=20, min_samples_leaf=2, min_samples_split=5, n_esti
mators=300;, score=0.919 total time=   0.4s
[CV 5/5] END max_depth=20, min_samples_leaf=2, min_samples_split=2, n_esti
mators=300;, score=0.869 total time=   0.4s
[CV 3/5] END max_depth=20, min_samples_leaf=2, min_samples_split=5, n_esti
mators=200;, score=0.943 total time=   0.2s
[CV 2/5] END max_depth=20, min_samples_leaf=2, min_samples_split=10, n_est
imators=100;, score=0.971 total time=   0.2s
[CV 3/5] END max_depth=20, min_samples_leaf=2, min_samples_split=5, n_esti
mators=300;, score=0.942 total time=   0.4s
[CV 4/5] END max_depth=20, min_samples_leaf=2, min_samples_split=10, n_est
imators=100;, score=0.912 total time=   0.2s
[CV 5/5] END max_depth=20, min_samples_leaf=2, min_samples_split=5, n_esti
mators=200;, score=0.870 total time=   0.3s
[CV 5/5] END max_depth=20, min_samples_leaf=2, min_samples_split=5, n_esti
mators=300;, score=0.870 total time=   0.4s
[CV 5/5] END max_depth=20, min_samples_leaf=2, min_samples_split=10, n_est
imators=100;, score=0.870 total time=   0.1s
[CV 3/5] END max_depth=20, min_samples_leaf=2, min_samples_split=10, n_est
imators=100;, score=0.941 total time=   0.1s
[CV 1/5] END max_depth=20, min_samples_leaf=2, min_samples_split=10, n_est
imators=100;, score=0.920 total time=   0.1s
[CV 1/5] END max_depth=20, min_samples_leaf=2, min_samples_split=10, n_est
imators=200;, score=0.924 total time=   0.2s
[CV 3/5] END max_depth=20, min_samples_leaf=2, min_samples_split=10, n_est
imators=200;, score=0.944 total time=   0.2s
[CV 5/5] END max_depth=20, min_samples_leaf=2, min_samples_split=10, n_est
imators=200;, score=0.870 total time=   0.3s
[CV 1/5] END max_depth=20, min_samples_leaf=4, min_samples_split=2, n_esti
mators=100;, score=0.929 total time=   0.1s
[CV 2/5] END max_depth=20, min_samples_leaf=2, min_samples_split=5, n_esti
mators=300;, score=0.976 total time=   0.4s
[CV 4/5] END max_depth=20, min_samples_leaf=2, min_samples_split=5, n_esti
mators=300;, score=0.915 total time=   0.4s
[CV 4/5] END max_depth=20, min_samples_leaf=2, min_samples_split=10, n_est
imators=200;, score=0.910 total time=   0.2s
[CV 2/5] END max_depth=20, min_samples_leaf=2, min_samples_split=10, n_est
imators=200;, score=0.977 total time=   0.3s
[CV 4/5] END max_depth=20, min_samples_leaf=2, min_samples_split=10, n_est
imators=300;, score=0.912 total time=   0.3s
[CV 2/5] END max_depth=20, min_samples_leaf=2, min_samples_split=10, n_est
imators=300;, score=0.977 total time=   0.4s
[CV 3/5] END max_depth=20, min_samples_leaf=4, min_samples_split=2, n_esti
```

```
mators=100;, score=0.942 total time=   0.1s
[CV 2/5] END max_depth=20, min_samples_leaf=4, min_samples_split=2, n_esti
mators=100;, score=0.977 total time=   0.2s
[CV 5/5] END max_depth=20, min_samples_leaf=4, min_samples_split=2, n_esti
mators=100;, score=0.872 total time=   0.1s
[CV 4/5] END max_depth=20, min_samples_leaf=4, min_samples_split=2, n_esti
mators=100;, score=0.912 total time=   0.1s
[CV 1/5] END max_depth=20, min_samples_leaf=2, min_samples_split=10, n_est
imators=300;, score=0.924 total time=   0.3s
[CV 2/5] END max_depth=20, min_samples_leaf=4, min_samples_split=2, n_esti
mators=200;, score=0.978 total time=   0.2s
[CV 4/5] END max_depth=20, min_samples_leaf=4, min_samples_split=2, n_esti
mators=200;, score=0.911 total time=   0.3s
[CV 1/5] END max_depth=20, min_samples_leaf=4, min_samples_split=2, n_esti
mators=200;, score=0.925 total time=   0.3s
[CV 5/5] END max_depth=20, min_samples_leaf=2, min_samples_split=10, n_est
imators=300;, score=0.872 total time=   0.3s
[CV 3/5] END max_depth=20, min_samples_leaf=2, min_samples_split=10, n_est
imators=300;, score=0.945 total time=   0.4s
[CV 2/5] END max_depth=20, min_samples_leaf=4, min_samples_split=5, n_esti
mators=100;, score=0.983 total time=   0.1s
[CV 3/5] END max_depth=20, min_samples_leaf=4, min_samples_split=2, n_esti
mators=200;, score=0.943 total time=   0.3s
[CV 1/5] END max_depth=20, min_samples_leaf=4, min_samples_split=2, n_esti
mators=300;, score=0.926 total time=   0.4s
[CV 3/5] END max_depth=20, min_samples_leaf=4, min_samples_split=2, n_esti
mators=300;, score=0.945 total time=   0.3s
[CV 4/5] END max_depth=20, min_samples_leaf=4, min_samples_split=5, n_esti
mators=100;, score=0.907 total time=   0.1s
[CV 5/5] END max_depth=20, min_samples_leaf=4, min_samples_split=2, n_esti
mators=300;, score=0.875 total time=   0.3s
[CV 5/5] END max_depth=20, min_samples_leaf=4, min_samples_split=2, n_esti
mators=200;, score=0.869 total time=   0.2s
[CV 3/5] END max_depth=20, min_samples_leaf=4, min_samples_split=5, n_esti
mators=100;, score=0.941 total time=   0.2s
[CV 1/5] END max_depth=20, min_samples_leaf=4, min_samples_split=5, n_esti
mators=200;, score=0.925 total time=   0.2s
[CV 5/5] END max_depth=20, min_samples_leaf=4, min_samples_split=5, n_esti
mators=100;, score=0.868 total time=   0.1s
[CV 1/5] END max_depth=20, min_samples_leaf=4, min_samples_split=5, n_esti
mators=100;, score=0.925 total time=   0.1s
[CV 3/5] END max_depth=20, min_samples_leaf=4, min_samples_split=5, n_esti
mators=200;, score=0.944 total time=   0.2s
[CV 5/5] END max_depth=20, min_samples_leaf=4, min_samples_split=5, n_esti
mators=200;, score=0.873 total time=   0.2s
[CV 1/5] END max_depth=20, min_samples_leaf=4, min_samples_split=10, n_est
imators=100;, score=0.926 total time=   0.1s
[CV 2/5] END max_depth=20, min_samples_leaf=4, min_samples_split=2, n_esti
mators=300;, score=0.979 total time=   0.4s
[CV 4/5] END max_depth=20, min_samples_leaf=4, min_samples_split=2, n_esti
mators=300;, score=0.909 total time=   0.4s
[CV 2/5] END max_depth=20, min_samples_leaf=4, min_samples_split=5, n_esti
mators=200;, score=0.981 total time=   0.2s
[CV 2/5] END max_depth=20, min_samples_leaf=4, min_samples_split=10, n_est
imators=100;, score=0.980 total time=   0.1s
[CV 4/5] END max_depth=20, min_samples_leaf=4, min_samples_split=5, n_esti
mators=200;, score=0.911 total time=   0.2s
[CV 3/5] END max_depth=20, min_samples_leaf=4, min_samples_split=10, n_est
imators=100;, score=0.941 total time=   0.1s
[CV 2/5] END max_depth=20, min_samples_leaf=4, min_samples_split=5, n_esti
```

```
mators=300;, score=0.979 total time=   0.3s
[CV 5/5] END max_depth=20, min_samples_leaf=4, min_samples_split=10, n_est
imators=100;, score=0.874 total time=   0.1s
[CV 4/5] END max_depth=20, min_samples_leaf=4, min_samples_split=5, n_esti
mators=300;, score=0.909 total time=   0.3s
[CV 1/5] END max_depth=20, min_samples_leaf=4, min_samples_split=5, n_esti
mators=300;, score=0.921 total time=   0.3s
[CV 4/5] END max_depth=20, min_samples_leaf=4, min_samples_split=10, n_est
imators=100;, score=0.908 total time=   0.1s
[CV 2/5] END max_depth=20, min_samples_leaf=4, min_samples_split=10, n_est
imators=200;, score=0.978 total time=   0.2s
[CV 4/5] END max_depth=20, min_samples_leaf=4, min_samples_split=10, n_est
imators=200;, score=0.911 total time=   0.2s
[CV 1/5] END max_depth=20, min_samples_leaf=4, min_samples_split=10, n_est
imators=200;, score=0.926 total time=   0.3s
[CV 1/5] END max_depth=20, min_samples_leaf=4, min_samples_split=10, n_est
imators=300;, score=0.924 total time=   0.4s
[CV 5/5] END max_depth=20, min_samples_leaf=4, min_samples_split=5, n_esti
mators=300;, score=0.873 total time=   0.3s
[CV 3/5] END max_depth=20, min_samples_leaf=4, min_samples_split=10, n_est
imators=200;, score=0.943 total time=   0.2s
[CV 3/5] END max_depth=20, min_samples_leaf=4, min_samples_split=5, n_esti
mators=300;, score=0.945 total time=   0.4s
[CV 2/5] END max_depth=30, min_samples_leaf=1, min_samples_split=2, n_esti
mators=100;, score=0.972 total time=   0.2s
[CV 5/5] END max_depth=20, min_samples_leaf=4, min_samples_split=10, n_est
imators=200;, score=0.870 total time=   0.3s
[CV 3/5] END max_depth=20, min_samples_leaf=4, min_samples_split=10, n_est
imators=300;, score=0.946 total time=   0.4s
[CV 4/5] END max_depth=30, min_samples_leaf=1, min_samples_split=2, n_esti
mators=100;, score=0.916 total time=   0.2s
[CV 5/5] END max_depth=20, min_samples_leaf=4, min_samples_split=10, n_est
imators=300;, score=0.873 total time=   0.4s
[CV 3/5] END max_depth=30, min_samples_leaf=1, min_samples_split=2, n_esti
mators=100;, score=0.945 total time=   0.1s
[CV 1/5] END max_depth=30, min_samples_leaf=1, min_samples_split=2, n_esti
mators=200;, score=0.919 total time=   0.3s
[CV 5/5] END max_depth=30, min_samples_leaf=1, min_samples_split=2, n_esti
mators=100;, score=0.876 total time=   0.1s
[CV 3/5] END max_depth=30, min_samples_leaf=1, min_samples_split=2, n_esti
mators=200;, score=0.940 total time=   0.3s
[CV 1/5] END max_depth=30, min_samples_leaf=1, min_samples_split=2, n_esti
mators=100;, score=0.914 total time=   0.1s
[CV 2/5] END max_depth=20, min_samples_leaf=4, min_samples_split=10, n_est
imators=300;, score=0.979 total time=   0.4s
[CV 5/5] END max_depth=30, min_samples_leaf=1, min_samples_split=2, n_esti
mators=200;, score=0.873 total time=   0.3s
[CV 1/5] END max_depth=30, min_samples_leaf=1, min_samples_split=5, n_esti
mators=100;, score=0.921 total time=   0.1s
[CV 4/5] END max_depth=20, min_samples_leaf=4, min_samples_split=10, n_est
imators=300;, score=0.909 total time=   0.3s
[CV 3/5] END max_depth=30, min_samples_leaf=1, min_samples_split=5, n_esti
mators=100;, score=0.940 total time=   0.1s
[CV 2/5] END max_depth=30, min_samples_leaf=1, min_samples_split=2, n_esti
mators=200;, score=0.972 total time=   0.2s
[CV 4/5] END max_depth=30, min_samples_leaf=1, min_samples_split=2, n_esti
mators=200;, score=0.914 total time=   0.2s
[CV 2/5] END max_depth=30, min_samples_leaf=1, min_samples_split=5, n_esti
mators=100;, score=0.977 total time=   0.1s
[CV 5/5] END max_depth=30, min_samples_leaf=1, min_samples_split=5, n_esti
```

```
mators=100;, score=0.875 total time=   0.1s
[CV 4/5] END max_depth=30, min_samples_leaf=1, min_samples_split=5, n_esti
mators=100;, score=0.914 total time=   0.1s
[CV 2/5] END max_depth=30, min_samples_leaf=1, min_samples_split=2, n_esti
mators=300;, score=0.976 total time=   0.4s
[CV 4/5] END max_depth=30, min_samples_leaf=1, min_samples_split=2, n_esti
mators=300;, score=0.915 total time=   0.4s
[CV 1/5] END max_depth=30, min_samples_leaf=1, min_samples_split=2, n_esti
mators=300;, score=0.919 total time=   0.4s
[CV 2/5] END max_depth=30, min_samples_leaf=1, min_samples_split=5, n_esti
mators=200;, score=0.976 total time=   0.2s
[CV 4/5] END max_depth=30, min_samples_leaf=1, min_samples_split=5, n_esti
mators=200;, score=0.907 total time=   0.2s
[CV 1/5] END max_depth=30, min_samples_leaf=1, min_samples_split=5, n_esti
mators=200;, score=0.927 total time=   0.3s
[CV 1/5] END max_depth=30, min_samples_leaf=1, min_samples_split=5, n_esti
mators=300;, score=0.918 total time=   0.4s
[CV 3/5] END max_depth=30, min_samples_leaf=1, min_samples_split=5, n_esti
mators=300;, score=0.946 total time=   0.4s
[CV 2/5] END max_depth=30, min_samples_leaf=1, min_samples_split=10, n_est
imators=100;, score=0.973 total time=   0.1s
[CV 3/5] END max_depth=30, min_samples_leaf=1, min_samples_split=2, n_esti
mators=300;, score=0.942 total time=   0.4s
[CV 3/5] END max_depth=30, min_samples_leaf=1, min_samples_split=5, n_esti
mators=200;, score=0.946 total time=   0.2s
[CV 5/5] END max_depth=30, min_samples_leaf=1, min_samples_split=5, n_esti
mators=200;, score=0.870 total time=   0.3s
[CV 5/5] END max_depth=30, min_samples_leaf=1, min_samples_split=5, n_esti
mators=300;, score=0.872 total time=   0.3s
[CV 3/5] END max_depth=30, min_samples_leaf=1, min_samples_split=10, n_est
imators=100;, score=0.940 total time=   0.1s
[CV 5/5] END max_depth=30, min_samples_leaf=1, min_samples_split=2, n_esti
mators=300;, score=0.876 total time=   0.4s
[CV 4/5] END max_depth=30, min_samples_leaf=1, min_samples_split=10, n_est
imators=100;, score=0.910 total time=   0.1s
[CV 1/5] END max_depth=30, min_samples_leaf=1, min_samples_split=10, n_est
imators=100;, score=0.928 total time=   0.1s
[CV 1/5] END max_depth=30, min_samples_leaf=1, min_samples_split=10, n_est
imators=200;, score=0.924 total time=   0.3s
[CV 3/5] END max_depth=30, min_samples_leaf=1, min_samples_split=10, n_est
imators=200;, score=0.944 total time=   0.3s
[CV 5/5] END max_depth=30, min_samples_leaf=1, min_samples_split=10, n_est
imators=100;, score=0.877 total time=   0.1s
[CV 2/5] END max_depth=30, min_samples_leaf=1, min_samples_split=5, n_esti
mators=300;, score=0.971 total time=   0.4s
[CV 5/5] END max_depth=30, min_samples_leaf=1, min_samples_split=10, n_est
imators=200;, score=0.872 total time=   0.2s
[CV 4/5] END max_depth=30, min_samples_leaf=1, min_samples_split=5, n_esti
mators=300;, score=0.911 total time=   0.4s
[CV 1/5] END max_depth=30, min_samples_leaf=2, min_samples_split=2, n_esti
mators=100;, score=0.923 total time=   0.1s
[CV 3/5] END max_depth=30, min_samples_leaf=2, min_samples_split=2, n_esti
mators=100;, score=0.949 total time=   0.1s
[CV 2/5] END max_depth=30, min_samples_leaf=1, min_samples_split=10, n_est
imators=300;, score=0.979 total time=   0.4s
[CV 5/5] END max_depth=30, min_samples_leaf=2, min_samples_split=2, n_esti
mators=100;, score=0.872 total time=   0.1s
[CV 4/5] END max_depth=30, min_samples_leaf=1, min_samples_split=10, n_est
imators=300;, score=0.911 total time=   0.3s
[CV 2/5] END max_depth=30, min_samples_leaf=1, min_samples_split=10, n_est
```

```
imators=200;, score=0.977 total time=   0.3s
[CV 4/5] END max_depth=30, min_samples_leaf=1, min_samples_split=10, n_est
imators=200;, score=0.914 total time=   0.2s
[CV 2/5] END max_depth=30, min_samples_leaf=2, min_samples_split=2, n_esti
mators=100;, score=0.977 total time=   0.1s
[CV 4/5] END max_depth=30, min_samples_leaf=2, min_samples_split=2, n_esti
mators=100;, score=0.911 total time=   0.1s
[CV 1/5] END max_depth=30, min_samples_leaf=1, min_samples_split=10, n_est
imators=300;, score=0.925 total time=   0.4s
[CV 1/5] END max_depth=30, min_samples_leaf=2, min_samples_split=2, n_esti
mators=200;, score=0.926 total time=   0.3s
[CV 2/5] END max_depth=30, min_samples_leaf=2, min_samples_split=2, n_esti
mators=200;, score=0.976 total time=   0.3s
[CV 4/5] END max_depth=30, min_samples_leaf=2, min_samples_split=2, n_esti
mators=200;, score=0.912 total time=   0.3s
[CV 3/5] END max_depth=30, min_samples_leaf=1, min_samples_split=10, n_est
imators=300;, score=0.940 total time=   0.4s
[CV 5/5] END max_depth=30, min_samples_leaf=1, min_samples_split=10, n_est
imators=300;, score=0.870 total time=   0.4s
[CV 2/5] END max_depth=30, min_samples_leaf=2, min_samples_split=5, n_esti
mators=100;, score=0.975 total time=   0.2s
[CV 1/5] END max_depth=30, min_samples_leaf=2, min_samples_split=2, n_esti
mators=300;, score=0.923 total time=   0.4s
[CV 3/5] END max_depth=30, min_samples_leaf=2, min_samples_split=2, n_esti
mators=300;, score=0.944 total time=   0.4s
[CV 4/5] END max_depth=30, min_samples_leaf=2, min_samples_split=5, n_esti
mators=100;, score=0.911 total time=   0.1s
[CV 5/5] END max_depth=30, min_samples_leaf=2, min_samples_split=2, n_esti
mators=200;, score=0.868 total time=   0.2s
[CV 3/5] END max_depth=30, min_samples_leaf=2, min_samples_split=2, n_esti
mators=200;, score=0.947 total time=   0.3s
[CV 3/5] END max_depth=30, min_samples_leaf=2, min_samples_split=5, n_esti
mators=100;, score=0.946 total time=   0.2s
[CV 5/5] END max_depth=30, min_samples_leaf=2, min_samples_split=5, n_esti
mators=100;, score=0.881 total time=   0.1s
[CV 5/5] END max_depth=30, min_samples_leaf=2, min_samples_split=2, n_esti
mators=300;, score=0.871 total time=   0.4s
[CV 1/5] END max_depth=30, min_samples_leaf=2, min_samples_split=5, n_esti
mators=200;, score=0.926 total time=   0.3s
[CV 1/5] END max_depth=30, min_samples_leaf=2, min_samples_split=5, n_esti
mators=100;, score=0.923 total time=   0.1s
[CV 4/5] END max_depth=30, min_samples_leaf=2, min_samples_split=2, n_esti
mators=300;, score=0.912 total time=   0.4s
[CV 3/5] END max_depth=30, min_samples_leaf=2, min_samples_split=5, n_esti
mators=200;, score=0.943 total time=   0.3s
[CV 2/5] END max_depth=30, min_samples_leaf=2, min_samples_split=2, n_esti
mators=300;, score=0.975 total time=   0.4s
[CV 5/5] END max_depth=30, min_samples_leaf=2, min_samples_split=5, n_esti
mators=200;, score=0.871 total time=   0.3s
[CV 1/5] END max_depth=30, min_samples_leaf=2, min_samples_split=10, n_est
imators=100;, score=0.919 total time=   0.1s
[CV 2/5] END max_depth=30, min_samples_leaf=2, min_samples_split=5, n_esti
mators=200;, score=0.976 total time=   0.2s
[CV 2/5] END max_depth=30, min_samples_leaf=2, min_samples_split=5, n_esti
mators=300;, score=0.980 total time=   0.4s
[CV 3/5] END max_depth=30, min_samples_leaf=2, min_samples_split=10, n_est
imators=100;, score=0.945 total time=   0.1s
[CV 2/5] END max_depth=30, min_samples_leaf=2, min_samples_split=10, n_est
imators=100;, score=0.978 total time=   0.1s
[CV 5/5] END max_depth=30, min_samples_leaf=2, min_samples_split=10, n_est
```

```
imators=100;, score=0.870 total time=   0.2s
[CV 4/5] END max_depth=30, min_samples_leaf=2, min_samples_split=5, n_esti
mators=300;, score=0.913 total time=   0.4s
[CV 4/5] END max_depth=30, min_samples_leaf=2, min_samples_split=5, n_esti
mators=200;, score=0.907 total time=   0.3s
[CV 4/5] END max_depth=30, min_samples_leaf=2, min_samples_split=10, n_est
imators=100;, score=0.915 total time=   0.2s
[CV 2/5] END max_depth=30, min_samples_leaf=2, min_samples_split=10, n_est
imators=200;, score=0.980 total time=   0.3s
[CV 1/5] END max_depth=30, min_samples_leaf=2, min_samples_split=5, n_esti
mators=300;, score=0.924 total time=   0.4s
[CV 1/5] END max_depth=30, min_samples_leaf=2, min_samples_split=10, n_est
imators=200;, score=0.927 total time=   0.3s
[CV 4/5] END max_depth=30, min_samples_leaf=2, min_samples_split=10, n_est
imators=200;, score=0.911 total time=   0.3s
[CV 3/5] END max_depth=30, min_samples_leaf=2, min_samples_split=5, n_esti
mators=300;, score=0.942 total time=   0.4s
[CV 5/5] END max_depth=30, min_samples_leaf=2, min_samples_split=5, n_esti
mators=300;, score=0.873 total time=   0.4s
[CV 2/5] END max_depth=30, min_samples_leaf=4, min_samples_split=2, n_esti
mators=100;, score=0.981 total time=   0.1s
[CV 3/5] END max_depth=30, min_samples_leaf=2, min_samples_split=10, n_est
imators=200;, score=0.943 total time=   0.2s
[CV 1/5] END max_depth=30, min_samples_leaf=2, min_samples_split=10, n_est
imators=300;, score=0.927 total time=   0.4s
[CV 4/5] END max_depth=30, min_samples_leaf=4, min_samples_split=2, n_esti
mators=100;, score=0.909 total time=   0.1s
[CV 3/5] END max_depth=30, min_samples_leaf=2, min_samples_split=10, n_est
imators=300;, score=0.946 total time=   0.4s
[CV 3/5] END max_depth=30, min_samples_leaf=4, min_samples_split=2, n_esti
mators=100;, score=0.948 total time=   0.1s
[CV 5/5] END max_depth=30, min_samples_leaf=2, min_samples_split=10, n_est
imators=200;, score=0.876 total time=   0.2s
[CV 1/5] END max_depth=30, min_samples_leaf=4, min_samples_split=2, n_esti
mators=200;, score=0.925 total time=   0.2s
[CV 5/5] END max_depth=30, min_samples_leaf=4, min_samples_split=2, n_esti
mators=100;, score=0.865 total time=   0.1s
[CV 5/5] END max_depth=30, min_samples_leaf=2, min_samples_split=10, n_est
imators=300;, score=0.876 total time=   0.4s
[CV 3/5] END max_depth=30, min_samples_leaf=4, min_samples_split=2, n_esti
mators=200;, score=0.942 total time=   0.3s
[CV 1/5] END max_depth=30, min_samples_leaf=4, min_samples_split=2, n_esti
mators=100;, score=0.923 total time=   0.1s
[CV 2/5] END max_depth=30, min_samples_leaf=4, min_samples_split=2, n_esti
mators=200;, score=0.979 total time=   0.2s
[CV 5/5] END max_depth=30, min_samples_leaf=4, min_samples_split=2, n_esti
mators=200;, score=0.869 total time=   0.3s
[CV 2/5] END max_depth=30, min_samples_leaf=2, min_samples_split=10, n_est
imators=300;, score=0.979 total time=   0.4s
[CV 4/5] END max_depth=30, min_samples_leaf=2, min_samples_split=10, n_est
imators=300;, score=0.915 total time=   0.4s
[CV 1/5] END max_depth=30, min_samples_leaf=4, min_samples_split=5, n_esti
mators=100;, score=0.927 total time=   0.1s
[CV 3/5] END max_depth=30, min_samples_leaf=4, min_samples_split=5, n_esti
mators=100;, score=0.945 total time=   0.1s
[CV 2/5] END max_depth=30, min_samples_leaf=4, min_samples_split=2, n_esti
mators=300;, score=0.978 total time=   0.4s
[CV 2/5] END max_depth=30, min_samples_leaf=4, min_samples_split=5, n_esti
mators=100;, score=0.978 total time=   0.1s
[CV 4/5] END max_depth=30, min_samples_leaf=4, min_samples_split=2, n_esti
```

```
mators=300;, score=0.906 total time=   0.3s
[CV 5/5] END max_depth=30, min_samples_leaf=4, min_samples_split=5, n_esti
mators=100;, score=0.872 total time=   0.2s
[CV 4/5] END max_depth=30, min_samples_leaf=4, min_samples_split=2, n_esti
mators=200;, score=0.907 total time=   0.3s
[CV 4/5] END max_depth=30, min_samples_leaf=4, min_samples_split=5, n_esti
mators=100;, score=0.914 total time=   0.1s
[CV 2/5] END max_depth=30, min_samples_leaf=4, min_samples_split=5, n_esti
mators=200;, score=0.980 total time=   0.3s
[CV 1/5] END max_depth=30, min_samples_leaf=4, min_samples_split=2, n_esti
mators=300;, score=0.926 total time=   0.4s
[CV 1/5] END max_depth=30, min_samples_leaf=4, min_samples_split=5, n_esti
mators=200;, score=0.921 total time=   0.3s
[CV 4/5] END max_depth=30, min_samples_leaf=4, min_samples_split=5, n_esti
mators=200;, score=0.911 total time=   0.3s
[CV 3/5] END max_depth=30, min_samples_leaf=4, min_samples_split=2, n_esti
mators=300;, score=0.942 total time=   0.4s
[CV 5/5] END max_depth=30, min_samples_leaf=4, min_samples_split=2, n_esti
mators=300;, score=0.873 total time=   0.4s
[CV 1/5] END max_depth=30, min_samples_leaf=4, min_samples_split=5, n_esti
mators=300;, score=0.925 total time=   0.3s
[CV 2/5] END max_depth=30, min_samples_leaf=4, min_samples_split=10, n_est
imators=100;, score=0.978 total time=   0.1s
[CV 3/5] END max_depth=30, min_samples_leaf=4, min_samples_split=5, n_esti
mators=200;, score=0.945 total time=   0.2s
[CV 3/5] END max_depth=30, min_samples_leaf=4, min_samples_split=5, n_esti
mators=300;, score=0.944 total time=   0.4s
[CV 4/5] END max_depth=30, min_samples_leaf=4, min_samples_split=10, n_est
imators=100;, score=0.908 total time=   0.1s
[CV 5/5] END max_depth=30, min_samples_leaf=4, min_samples_split=10, n_est
imators=100;, score=0.869 total time=   0.1s
[CV 5/5] END max_depth=30, min_samples_leaf=4, min_samples_split=5, n_esti
mators=200;, score=0.872 total time=   0.2s
[CV 3/5] END max_depth=30, min_samples_leaf=4, min_samples_split=10, n_est
imators=100;, score=0.945 total time=   0.1s
[CV 5/5] END max_depth=30, min_samples_leaf=4, min_samples_split=5, n_esti
mators=300;, score=0.871 total time=   0.3s
[CV 1/5] END max_depth=30, min_samples_leaf=4, min_samples_split=10, n_est
imators=200;, score=0.926 total time=   0.2s
[CV 1/5] END max_depth=30, min_samples_leaf=4, min_samples_split=10, n_est
imators=100;, score=0.925 total time=   0.1s
[CV 2/5] END max_depth=30, min_samples_leaf=4, min_samples_split=10, n_est
imators=200;, score=0.979 total time=   0.2s
[CV 3/5] END max_depth=30, min_samples_leaf=4, min_samples_split=10, n_est
imators=200;, score=0.943 total time=   0.2s
[CV 4/5] END max_depth=30, min_samples_leaf=4, min_samples_split=10, n_est
imators=200;, score=0.914 total time=   0.2s
[CV 4/5] END max_depth=30, min_samples_leaf=4, min_samples_split=5, n_esti
mators=300;, score=0.910 total time=   0.3s
[CV 5/5] END max_depth=30, min_samples_leaf=4, min_samples_split=10, n_est
imators=200;, score=0.870 total time=   0.2s
[CV 2/5] END max_depth=30, min_samples_leaf=4, min_samples_split=5, n_esti
mators=300;, score=0.979 total time=   0.4s
[CV 1/5] END max_depth=30, min_samples_leaf=4, min_samples_split=10, n_est
imators=300;, score=0.924 total time=   0.3s
[CV 2/5] END max_depth=30, min_samples_leaf=4, min_samples_split=10, n_est
imators=300;, score=0.980 total time=   0.3s
[CV 3/5] END max_depth=30, min_samples_leaf=4, min_samples_split=10, n_est
imators=300;, score=0.945 total time=   0.3s
[CV 4/5] END max_depth=30, min_samples_leaf=4, min_samples_split=10, n_est
```

```
imators=300;, score=0.909 total time=   0.3s
[CV 5/5] END max_depth=30, min_samples_leaf=4, min_samples_split=10, n_est
imators=300;, score=0.874 total time=   0.3s
```

Out[ ]:

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

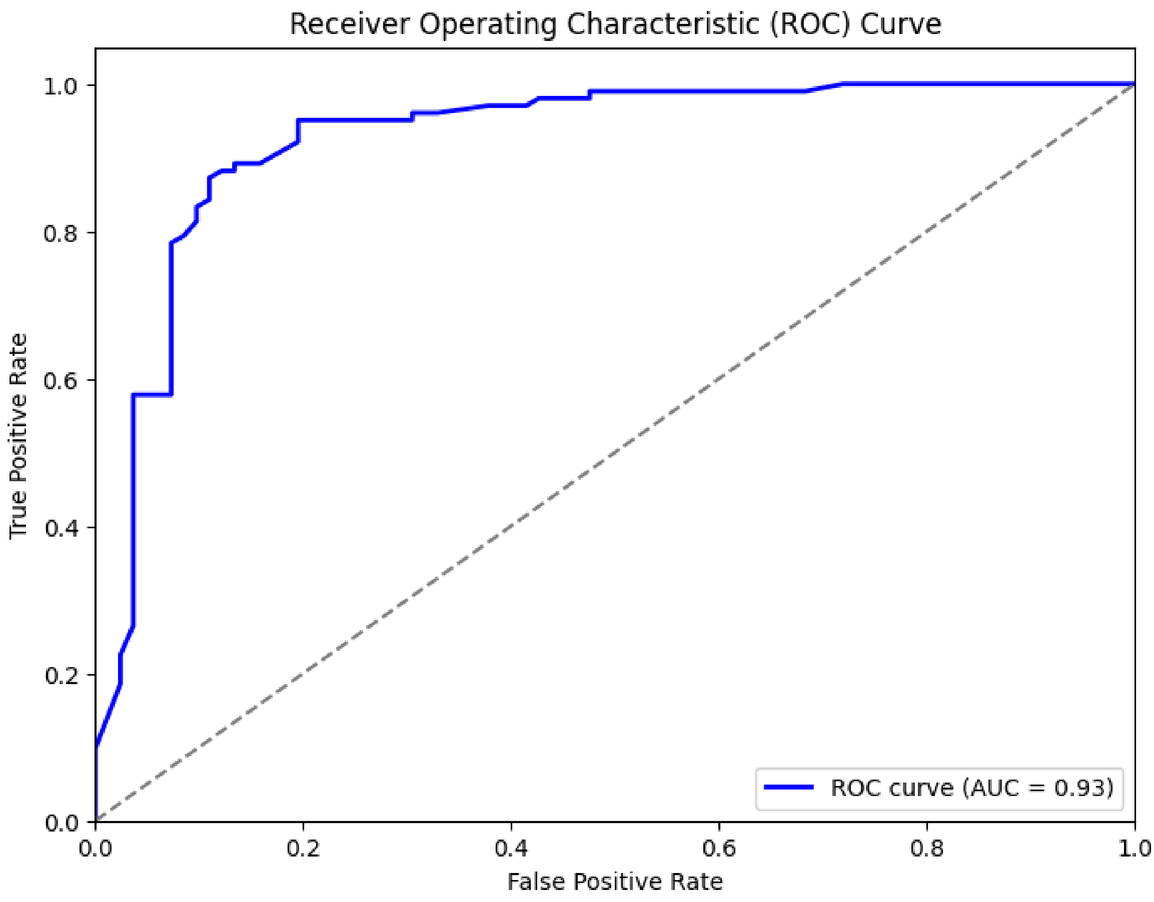
In [ ]:

```
# get best combination and result
print("Beste Hyperparameter-Kombinationen: ", grid_search.best_params_)
print("Beste Performance: ", grid_search.best_score_)
```

```
Beste Hyperparameter-Kombinationen:  {'max_depth': 30, 'min_samples_leaf':
2, 'min_samples_split': 10, 'n_estimators': 300}
Beste Performance:  0.9283950617283951
```

In [ ]:

```
target_pred_proba = forest.predict_proba(features_test)[:, 1]

# compute ROC curve and ROC-AUC score
fpr, tpr, thresholds = roc_curve(target_test, target_pred_proba)
roc_auc = roc_auc_score(target_test, target_pred_proba)

# plot ROC curve
plt.figure(figsize=(8, 6))
plt.plot(fpr, tpr, color='blue', lw=2, label='ROC curve (AUC = %0.2f)' %
plt.plot([0, 1], [0, 1], color='gray', linestyle='--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC) Curve')
plt.legend(loc="lower right")
plt.show()
```



Evaluation und Ergebnisdarstellung

Basierend auf der Evaluation der drei Klassifikationsmodelle – Random Forest Classifier, Logistische Regression und Support Vector Machine – erzielte der RFC vor dem Parameter-Tuning die beste Leistung mit einer Genauigkeit von 88.59% und einem ROC AUC Score von 88.27% (Diese Werte können bei erneutem ausführen des Notebooks abweichen, da der Train-Test split einen randomstate von 42 besitzt). Der precision, recall und f1-score für beide Klassen (Herzkrankheit und Normal) zeigen eine ausgeglichene Leistung des Modells. Nach dem Parameter-Tuning wurden die Hyperparameter des RFC optimiert, wodurch eine verbesserte Leistung mit einer ROC AUC Score von 92.78% erzielt wurde. Dies unterstreicht die Wirksamkeit des gewählten Ansatzes und die Fähigkeit des Modells, zwischen Herzkrankheit und Normalzustand zu unterscheiden.

Vorhersage-Demo

In [ ]:
```
selected_data_point = features_train.iloc[0:1, :]
selected_target = target_train.iloc[0]

# make the prediction for the chosen data point
prediction = forest.predict(selected_data_point)

# disply the chosen data point, the true class and the predicted class
print("Ausgewählter Datenpunkt:")
```

```
print(selected_data_point)
print("\nWahre Klasse des ausgewählten Datenpunkts:", selected_target)
print("\nVorhersage für den ausgewählten Datenpunkt:", prediction)
```

```
Ausgewählter Datenpunkt:
     Age  RestingBP  Cholesterol  FastingBS  MaxHR  Oldpeak  Sex_F  Sex_M  
\
161   49        128          212          0     96      0.0  False   True   

     ChestPainType_ASY  ChestPainType_ATA  ChestPainType_NAP  \
161               True              False              False   

     ChestPainType_TA  RestingECG_LVH  RestingECG_Normal  RestingECG_ST  \
161             False           False               True          False   

     ExerciseAngina_N  ExerciseAngina_Y  ST_Slope_Down  ST_Slope_Flat  \
161             False              True          False           True   

     ST_Slope_Up  
161        False  

Wahre Klasse des ausgewählten Datenpunkts: 1

Vorhersage für den ausgewählten Datenpunkt: [1]
```

Wie man anhand des Beispiels sieht, erkennt das Model den gegeben Datenpunkt korrekt als Herzerkrankung an.