Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ _____Информатика и системы управления_____

КАФЕДРА _____Системы обработки информации и управления_____

РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

К НАУЧНО-ИССЛЕДОВАТЕЛЬСКОЙ РАБОТЕ ПО ОБРАБОТКЕ И АНАЛИЗУ ДАННЫХ

НА ТЕМУ:

___Предсказание сердечных заболеваний с_____
___использованием моделей AdaBoost и XGBoost____

Студент __ИУ5-32М_____ _____ _____**О.К. Румянцев**__
(Группа) (Подпись, дата) (И.О.Фамилия)

Руководитель _____ __**Ю.Е. Гапанюк**__ ___
(Подпись, дата) (И.О.Фамилия)

Консультант _____ _____
(Подпись, дата) (И.О.Фамилия)

2021 г.

Введение

Сердечно-сосудистые заболевания (ССЗ) являются основной причиной смерти во всём мире, забирая около 18 миллионов жизней каждый год, что составляет 31 процент всех смертей в мире. Четыре из пяти смертей, вызванных ССЗ, вызваны сердечными приступами и инсультами, и треть этих смертей являются преждевременными для людей моложе 70 лет. Отказ сердца является частым следствием ССЗ и в представленном наборе данных содержится одиннадцать признаков, которые могут быть использованы для предсказания возможности сердечных заболеваний.

Люди с ССЗ или с высоким риском заболевания нуждаются в ранней диагностике и лечении, в чём может помочь модель машинного обучения.

In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

Чтение датасета

In [2]:
```
df = pd.read_csv("../input/heart-failure-prediction/heart.csv")
df.head()
```

Out[2]:

	Age	Sex	ChestPainType	RestingBP	Cholesterol	FastingBS	RestingECG	MaxHR	Exercis
0	40	M	ATA	140	289	0	Normal	172	
1	49	F	NAP	160	180	0	Normal	156	
2	37	M	ATA	130	283	0	ST	98	
3	48	F	ASY	138	214	0	Normal	108	
4	54	M	NAP	150	195	0	Normal	122	

In [3]:
```
df.shape
```

Out[3]:
```
(918, 12)
```

Пропуски в данных

Проверим пропуски в данных (значения null)

In [4]:
```
df.isnull().sum()
```

Out[4]:
```
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
```

Пропусков в данных нету

Посмотрим уникальные значения в каждой категории

In [5]:
```
print("Sex:",df['Sex'].unique())
```

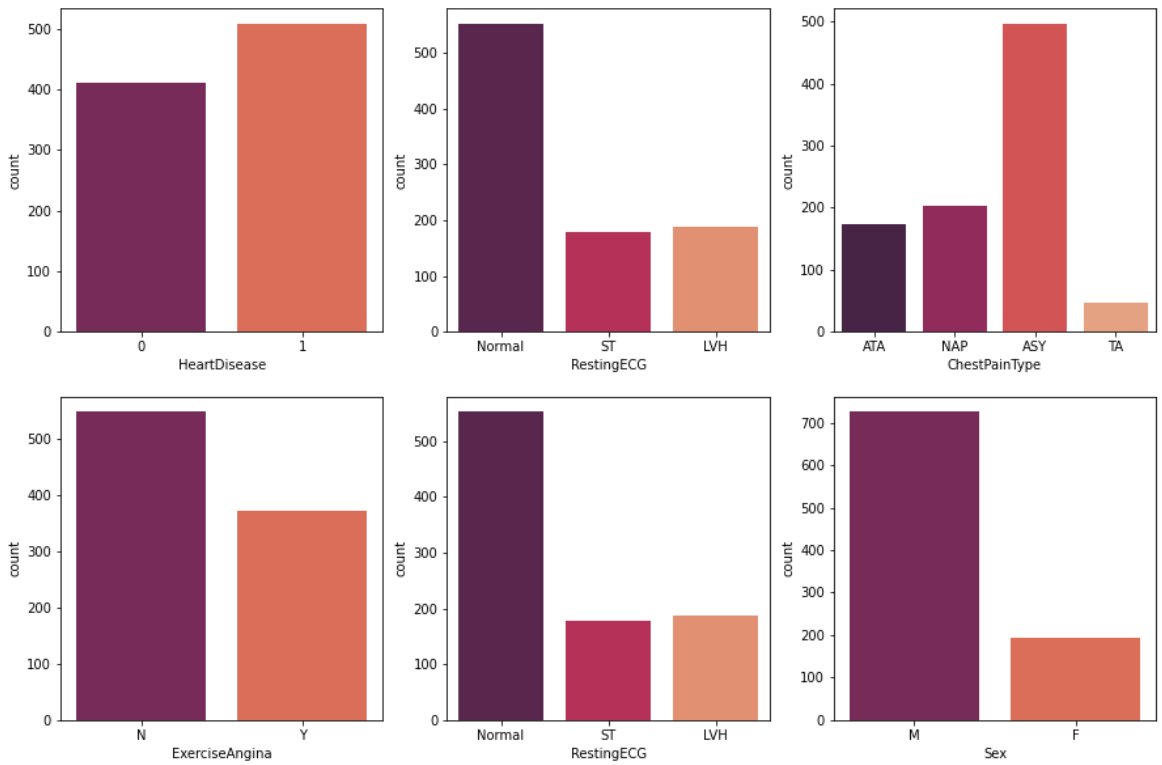
```
print("RestingECG:",df['RestingECG'].unique())
print("ChestPainType:",df['ChestPainType'].unique())
print("ExerciseAngina:",df['ExerciseAngina'].unique())
print("ST_Slope:",df['ST_Slope'].unique())
```

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

Распределение переменных

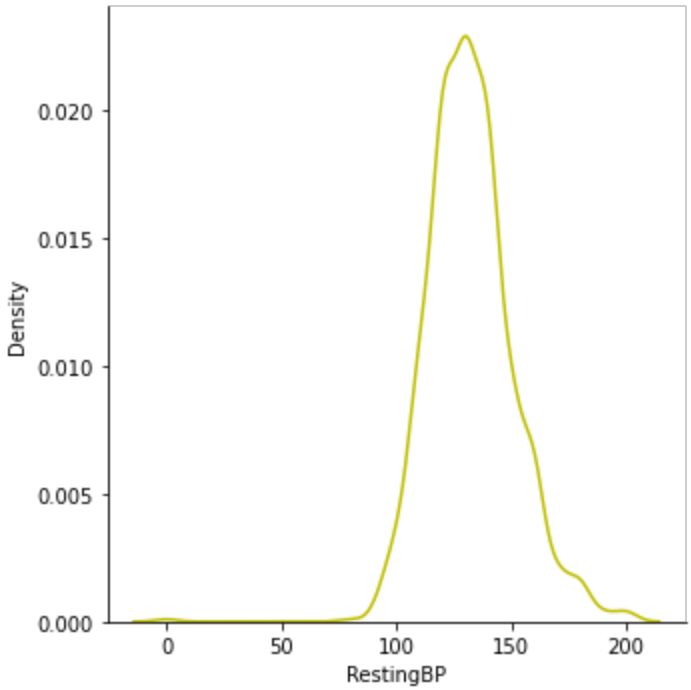
Рассмотрим распределение категориальных переменных с использованием библиотеки Seaborn

In [6]:
```
f, axes = plt.subplots(2, 3, figsize=(15, 10))

sns.countplot(x = df['HeartDisease'], data = df, palette='rocket', ax=
sns.countplot(x = df['RestingECG'], data = df, palette='rocket', ax=ax
sns.countplot(x = df['ChestPainType'], data = df, palette='rocket', ax

sns.countplot(x = df['ExerciseAngina'], data = df, palette='rocket', a
sns.countplot(x = df['RestingECG'], data = df, palette='rocket', ax=ax
sns.countplot(x = df['Sex'], data = df, palette='rocket', ax=axes[1,2]
plt.show()
```



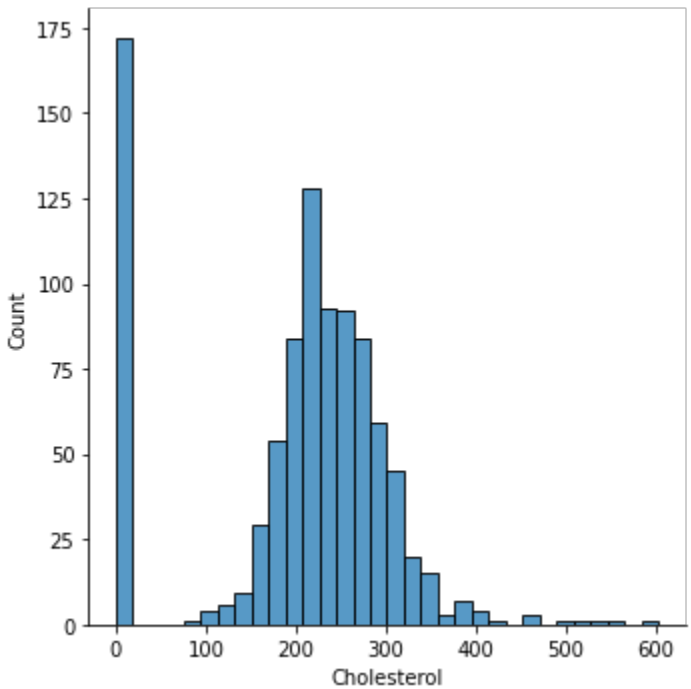
In [7]:
```
plt.figure(figsize = (15, 10))
sns.displot(df['RestingBP'], color = 'y', kind='kde')

plt.show()
```

```
<Figure size 1080x720 with 0 Axes>
```



In [8]:
```
plt.figure(figsize = (20, 10))
sns.displot(df['Cholesterol'])

plt.show()
```

```
<Figure size 1440x720 with 0 Axes>
```



Подготовка датасета к обучению модели

Кодирование категориальных признаков

Для обучения модели закодируем категориальные признаки с помощью LabelEncoder

In [9]:
```
from sklearn.preprocessing import LabelEncoder

le=LabelEncoder()

df['Sex']=le.fit_transform(df['Sex'])
df['RestingECG']=le.fit_transform(df['RestingECG'])
df['ChestPainType']=le.fit_transform(df['ChestPainType'])
df['ExerciseAngina']=le.fit_transform(df['ExerciseAngina'])
df['ST_Slope']=le.fit_transform(df['ST_Slope'])

df.head()
```

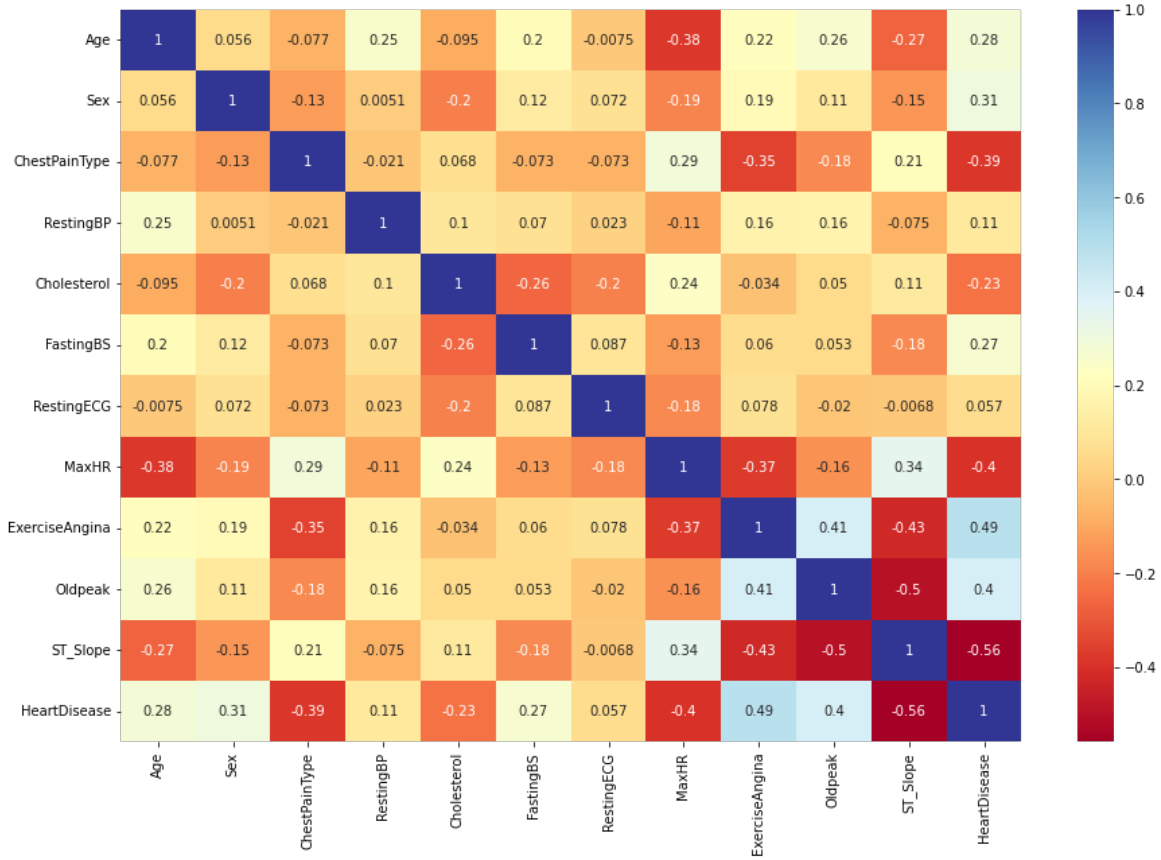
Out[9]:

	Age	Sex	ChestPainType	RestingBP	Cholesterol	FastingBS	RestingECG	MaxHR	Exercis
0	40	1	1	140	289	0	1	172	
1	49	0	2	160	180	0	1	156	
2	37	1	1	130	283	0	2	98	
3	48	0	0	138	214	0	1	108	
4	54	1	2	150	195	0	1	122	

Кореляционная матрица

In [10]:
```
plt.figure(figsize=(15,10))
sns.heatmap(df.corr(), annot=True, cmap='RdYlBu')
```

Out[10]: <AxesSubplot:>



Удалим целевую переменную для обучения

In [11]:
```
X = df.drop('HeartDisease', axis=1)
y = df['HeartDisease']
```

Разделим выборку на train и test

In [12]:
```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.

print("X_train: ", X_train.shape)
print("X_test: ", X_test.shape)
print("y_train: ", y_train.shape)
print("y_test: ", y_test.shape)
```

```
X_train:  (550, 11)
X_test:  (368, 11)
y_train:  (550,)
y_test:  (368,)
```

Классификатор AdaBoost

In [13]:
```
from sklearn.ensemble import AdaBoostClassifier

abc = AdaBoostClassifier(n_estimators=500, learning_rate=0.01, random_
model = abc.fit(X_train, y_train)
```

Предсказание результатов

In [14]:
```
y_pred_adaboost = model.predict(X_test)
print(y_pred_adaboost)
```

```
[1 1 1 0 1 0 1 1 1 0 1 0 1 0 1 0 0 1 0 1 1 1 1 0 0 0 0 1 0 1 1 1 0 1
1 0 0
 0 1 0 1 1 0 0 1 0 0 1 1 1 0 1 0 0 0 1 0 1 1 1 0 1 1 1 1 0 0 1 1 0 1
1 0 0
 1 1 0 0 0 0 0 1 1 1 0 1 1 1 1 0 1 0 0 1 1 1 1 0 1 0 1 0 0 1 1 1 1 1
1 0 1
 0 1 1 1 1 1 0 0 0 0 1 0 1 0 0 0 1 1 0 0 1 1 1 0 1 1 0 0 0 0 1 1 1 1
1 0 0
 0 0 0 1 0 1 1 1 1 0 1 1 1 0 1 1 1 0 1 1 0 0 1 1 1 0 1 0 1 1 1 1 1 1
0 1 0
 1 0 1 0 1 0 0 1 1 1 0 1 0 1 1 0 1 1 0 0 0 0 0 1 1 1 0 1 1 0 0 1 1 0
0 0 1
 1 1 1 1 1 1 0 1 0 1 0 1 0 0 1 0 1 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 1 1
0 0 1
 1 1 1 0 0 1 0 1 0 1 1 0 0 1 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 1 0 0 1 1
1 0 0
 0 0 1 1 1 1 1 1 1 1 1 0 0 0 1 1 1 1 1 1 1 1 1 0 0 0 0 1 1 0 0 0 1 0 1
1 0 0
 0 1 1 0 1 0 0 1 1 0 1 1 1 1 0 0 0 1 0 0 0 1 0 1 1 0 0 0 1 0 0 1 0 1
1]
```

Accuracy модели

In [15]:
```
from sklearn.metrics import accuracy_score

print("AdaBoost Classifier Model Accuracy:", accuracy_score(y_test, y_
```

```
AdaBoost Classifier Model Accuracy: 0.8478260869565217
```

Классификатор XGBoost

In [16]:
```
from sklearn.ensemble import GradientBoostingClassifier

gbc = GradientBoostingClassifier(n_estimators=500, learning_rate=0.01,
model = gbc.fit(X_train, y_train)
```

Предсказание результатов

In [17]:
```
y_pred_xgboost = model.predict(X_test)
print(y_pred_xgboost)
```

```
[1 1 1 0 1 0 1 1 1 0 1 0 1 0 1 0 0 1 0 0 1 1 1 0 0 0 0 1 0 1 1 1 0 1
1 0 0
 0 1 0 1 1 0 0 1 0 0 1 1 1 0 1 1 0 0 1 0 1 1 1 0 1 1 1 1 0 0 1 1 0 1
1 0 0
 1 1 0 0 0 0 1 1 1 1 0 1 1 1 1 0 1 0 0 1 1 1 1 0 1 0 1 0 0 1 1 1 1 1
1 0 1
 0 1 1 1 1 1 0 0 0 0 1 0 1 0 0 0 1 1 0 0 1 1 1 0 1 1 1 1 0 1 1 1 1 1
1 0 0
 0 0 0 1 0 1 1 1 1 0 1 1 1 0 1 1 1 0 1 1 0 0 1 1 1 0 1 0 1 1 0 1 1 1
0 1 0
 1 0 1 1 1 0 0 1 1 1 0 1 0 1 1 1 1 1 1 0 0 0 0 1 1 1 0 1 1 0 0 1 1 0
0 0 1
 1 1 1 1 1 1 1 1 0 1 0 1 0 0 1 0 1 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 1 1
0 1 0
 1 1 1 0 0 1 0 1 0 0 1 0 0 1 1 0 0 0 1 1 1 1 1 1 1 0 1 1 1 1 0 0 1 1
1 1 0
 0 0 1 1 1 1 1 1 1 1 1 0 0 0 1 1 1 1 1 1 1 1 1 0 0 0 0 1 0 0 0 0 1 0 1
1 0 0
 0 1 1 0 0 0 0 1 1 0 1 1 1 1 0 0 0 1 0 0 0 1 1 1 1 0 0 0 1 0 0 0 1 1
1]
```

Accuracy модели

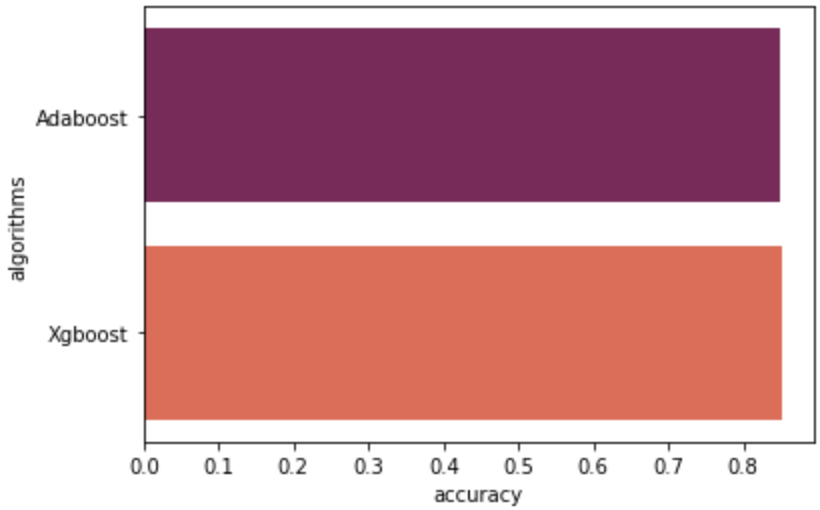
In [18]:
```
from sklearn.metrics import accuracy_score

print("XGBoost Classifier Model Accuracy:", accuracy_score(y_test, y_p
```

```
XGBoost Classifier Model Accuracy: 0.8505434782608695
```

Сравнение моделей Adaboost и XGboost

In [19]:
```
acc = pd.DataFrame({
    "algorithms": ['Adaboost', 'Xgboost'],
    "accuracy": [accuracy_score(y_test, y_pred_adaboost),accuracy_scor
})
sns.barplot(x='accuracy', y='algorithms', data=acc, palette='rocket')
```

Out[19]: <AxesSubplot:xlabel='accuracy', ylabel='algorithms'>



Вывод

На данном датасете немного лучше отработала модель Xgboost

In [ ]:

Вывод

В рамках курсовой работы был проанализирован датасет сердечных заболеваний, проведён анализ признаков, влияющих на возможность сердечно-сосудистых заболеваний. Проведена очистка и подготовка данных, а так же обучена модель, которая, на основе анализов человека может предсказать риск отказа сердца

Источники

1. Документация библиотеки seaborn https://seaborn.pydata.org/
2. Документация библиотеки pandas https://pandas.pydata.org/
3. Датасет Heart Failure Prediction https://www.kaggle.com/fedesoriano/heart-failure-prediction
4. Гапанюк Ю.Е. Методы машинного обучения – конспект лекций. https://github.com/ugapanyuk/ml_course_2021/wiki/COURSE_MMO
5. Документация библиотеки XGBoost https://xgboost.readthedocs.io/en/latest/