Московский государственный технический университет им. Н.Э. Баумана
Кафедра «Системы обработки информации и управления»

Домашнее задание №1
по дисциплине
«Методы машинного обучения»

Выполнил:
студент группы ИУ5-23М
Дин Но

Москва — 2021 г.

1. Задание

Домашнее задание по дисциплине направлено на решение комплексной задачи машинного обучения с учителем. Домашнее задание включает выполнение следующих шагов:

1.Поиск и выбор набора данных для построения модели машинного обучения. На основе выбранного набора данных строится модель машинного обучения для решения или задачи классификации, или задачи регрессии.

2.Для выбранного датасета (датасетов) на основе материалов лекций, в целях улучшения выборки, решить следующие задачи (если это необходимо в данном датасете):

у странение пропусков в данных;
кодирование категориальных признаков;
нормализацию числовых признаков;
масштабирование признаков;
обработку выбросов для числовых признаков;
обработку нестандартных признаков (которые не является числовым или категориальным);
отбор признаков, наиболее подходящих для построения модели;
устранение дисбаланса классов в случае решения задачи классификации на дисбалансированной выборке.

3.Обучить модель и оценить метрики качества для двух выборок :
исходная выборка, которая содержит только минимальную предобработку данных, необходимую для построения модели (например, кодирование категориальных признаков).
улучшенная выборка, полученная в результате полной предобработки данных в пункте 2.

4.Построить модель с использованием произвольной библиотеки AutoML.

5.Сравнить метрики для трех полученных моделей.

2. Ход выполнения работы
2.1. Выбор набора данных

Heart Attack Analysis & Prediction Dataset

A dataset for heart attack classification

Набор данных для классификации сердечных приступов

2.2. Проведение разведочного анализа данных

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

df = pd.read_csv("heart.csv")

print("The shape of the dataset is : ", df.shape)
df.head()

dict = {}
for i in list(df.columns):
    dict[i] = df[i].value_counts().shape[0]

pd.DataFrame(dict,index=["unique count"]).transpose()
```

	age	sex	cp	trtbps	chol	fbs	restecg	thalachh	exng	oldpeak	slp	caa	thall	output
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1

```
The shape of the dataset is :  (303, 14)
```

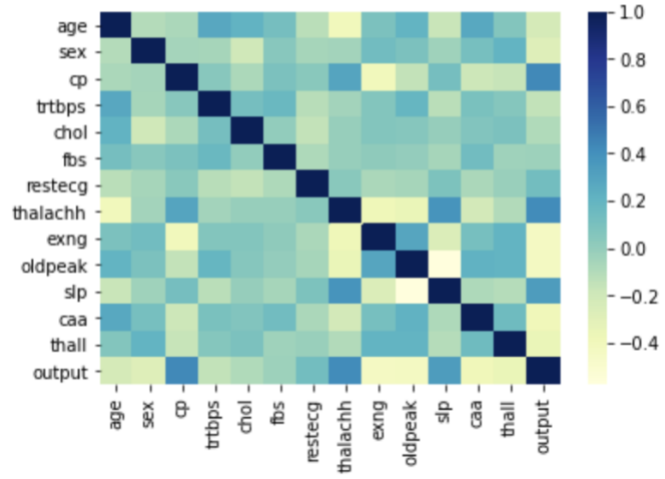
	unique count
age	41
sex	2
cp	4
trtbps	49
chol	152
fbs	2
restecg	3
thalachh	91
exng	2
oldpeak	40
slp	3
caa	5
thall	4
output	2

```
cat_cols = ['sex','exng','caa','cp','fbs','restecg','slp','thall']
con_cols = ["age","trtbps","chol","thalachh","oldpeak"]
target_col = ["output"]
print("The categorial cols are : ", cat_cols)
print("The continuous cols are : ", con_cols)
print("The target variable is :  ", target_col)
df[con_cols].describe().transpose()
```

```
The categorial cols are :  ['sex', 'exng', 'caa', 'cp', 'fbs', 'restecg', 'slp', 'thall']
The continuous cols are :  ['age', 'trtbps', 'chol', 'thalachh', 'oldpeak']
The target variable is :   ['output']
```

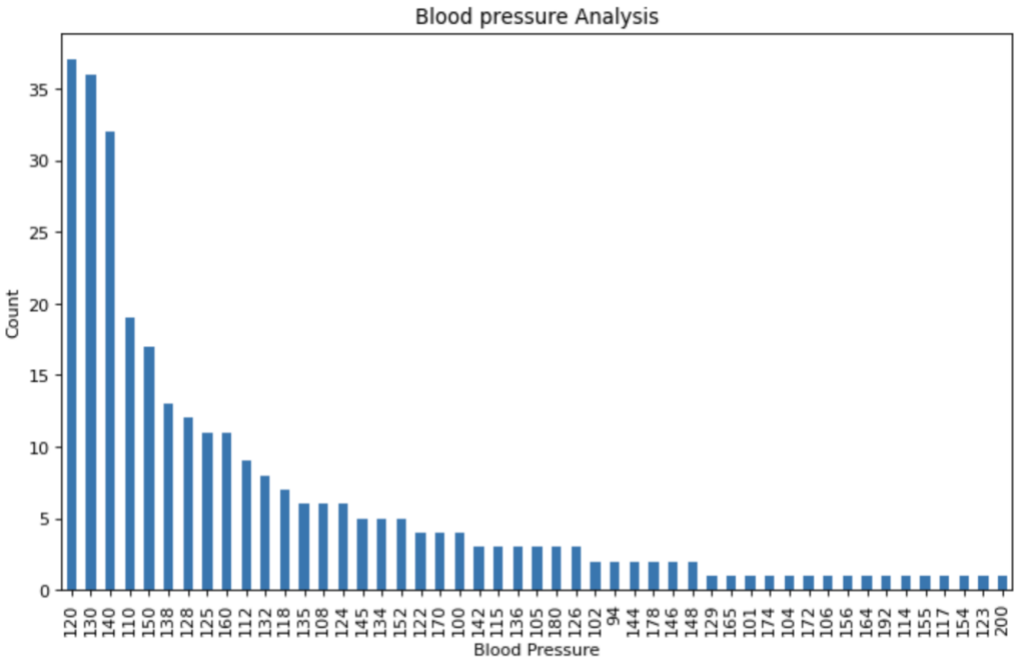
	count	mean	std	min	25%	50%	75%	max
age	303.0	54.366337	9.082101	29.0	47.5	55.0	61.0	77.0
trtbps	303.0	131.623762	17.538143	94.0	120.0	130.0	140.0	200.0
chol	303.0	246.264026	51.830751	126.0	211.0	240.0	274.5	564.0
thalachh	303.0	149.646865	22.905161	71.0	133.5	153.0	166.0	202.0
oldpeak	303.0	1.039604	1.161075	0.0	0.0	0.8	1.6	6.2

```
sns.heatmap(df.corr(),cmap='YlGnBu')
plt.show()
```



У кого бы ни была боль в груди 2-го типа, возможна высокая вероятность сердечного приступа.

```
from matplotlib.pyplot import figure

figure(figsize=(10, 6), dpi=80)
df['trtbps'].value_counts().plot(kind='bar')
plt.xlabel('Blood Pressure')
plt.ylabel('Count')
plt.title('Blood pressure Analysis')
plt.show()
```



2.3 исходная выборка, которая содержит только минимальную предобработку данных, необходимую для построения модели

```
from sklearn.preprocessing import StandardScaler

df1 = df
df1 = pd.get_dummies(df1, columns = cat_cols, drop_first = True)

X = df1.drop(['output'],axis=1)
y = df1[['output']]

scaler = StandardScaler()

X[con_cols] = scaler.fit_transform(X[con_cols])
X.head()
```

	age	trtbps	chol	thalachh	oldpeak	sex_1	exng_1	caa_1	caa_2	caa_3	...	cp_2	cp_3	fbs_1	restecg_1	restecg_2	slp_1	slp_2	thall_1
0	0.952197	0.763956	-0.256334	0.015443	1.087338	1	0	0	0	0	...	0	1	1	0	0	0	0	1
1	-1.915313	-0.092738	0.072199	1.633471	2.122573	1	0	0	0	0	...	1	0	0	1	0	0	0	0
2	-1.474158	-0.092738	-0.816773	0.977514	0.310912	0	0	0	0	0	...	0	0	0	0	0	0	1	0
3	0.180175	-0.663867	-0.198357	1.239897	-0.206705	1	0	0	0	0	...	0	0	0	1	0	0	1	0
4	0.290464	-0.663867	2.082050	0.583939	-0.379244	0	1	0	0	0	...	0	0	0	1	0	0	1	0

5 rows × 22 columns

2.3 улучшенная выборка, полученная в результате полной предобработки данных в пункте 2.

2.3.1 Проверьте, является ли это нулевым значением

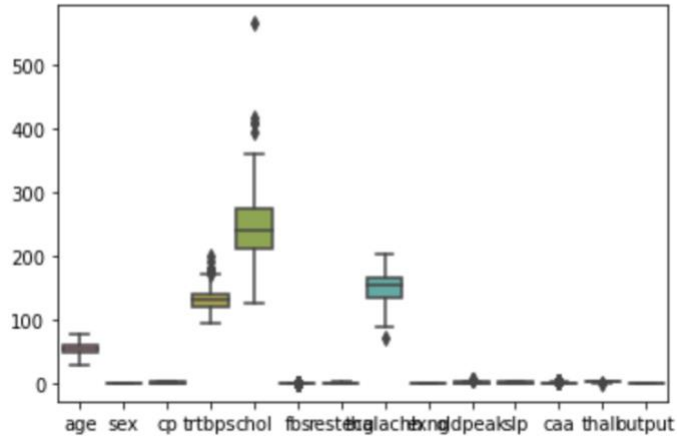
```
df.isnull().sum()
```

```
age         0
sex         0
cp          0
trtbps      0
chol        0
fbs         0
restecg     0
thalachh    0
exng        0
oldpeak     0
slp         0
caa         0
thall       0
output      0
dtype: int64
```

2.3.2 Нарисуйте прямоугольную диаграмму, чтобы проверить, есть ли выбросы.

```
import seaborn as sns
import matplotlib.pyplot as plt
sns.boxplot(data=df)
```

```
<AxesSubplot:>
```

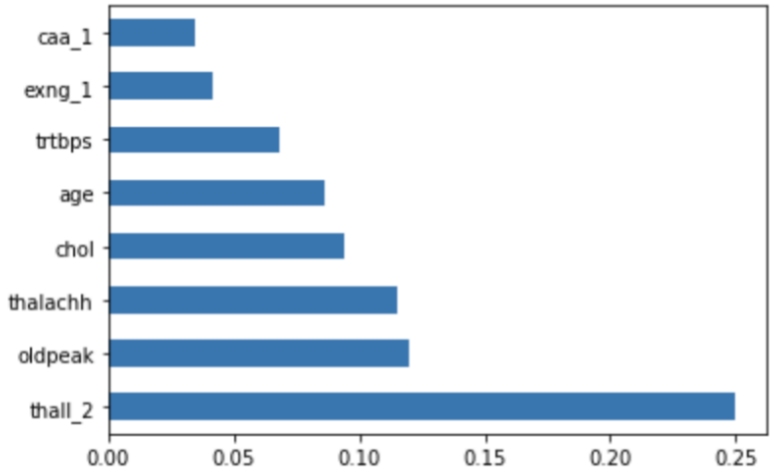


В этом наборе данных собраны все данные конкретных значений.Выбросы для наборов данных, скорее всего, существуют в реальности.Так что выбросы не обрабатываются.

2.3.3 С помощью метода случайного леса выберите 8 индикаторов с наибольшим весом..

```
from sklearn.ensemble import RandomForestRegressor

model = RandomForestRegressor ()
model.fit(X,y)
print(model.feature_importances_)

feat_importances = pd.Series(model.feature_importances_, index=X.columns)
feat_importances.nlargest(8).plot(kind='barh')
plt.show()
```

```
[8.64657310e-02 6.85138693e-02 9.42579198e-02 1.15088083e-01
 1.19909587e-01 2.79030970e-02 4.16307241e-02 3.47872934e-02
 1.61070390e-02 1.34504234e-02 2.11639466e-04 8.25298227e-03
 2.87637565e-02 2.93355346e-02 5.10211270e-03 9.51792123e-03
 5.89172328e-04 9.56742414e-03 1.42644305e-02 3.14641967e-03
 2.49491931e-01 2.36429089e-02]
```



2.4 Построить модель с использованием произвольной библиотеки AutoML.(TPOT)

```
from tpot import TPOTClassifier
from sklearn.model_selection import train_test_split

X = df.drop(['output'],axis=1)
y = df[['output']]
X_train, X_test, y_train, y_test = train_test_split(X,y, test_size = 0.2, random_state = 42)

tpot = TPOTClassifier(generations=5,population_size=20,verbosity=2)
tpot.fit(X_train,y_train)
print(tpot.score(X_test,y_test))
```

2.4 Сравнить метрики для трех полученных моделей.

2.4.1исходная выборка, которая содержит только минимальную предобработку данных, необходимую для построения модели (результат)

```
from sklearn import svm
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score

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

clf = svm.SVC(kernel='linear', C=1, random_state=42).fit(X_train,y_train)

y_pred = clf.predict(X_test)

print("The test accuracy score of SVM is ", accuracy_score(y_test, y_pred))
```

```
The test accuracy score of SVM is  0.8688524590163934
```

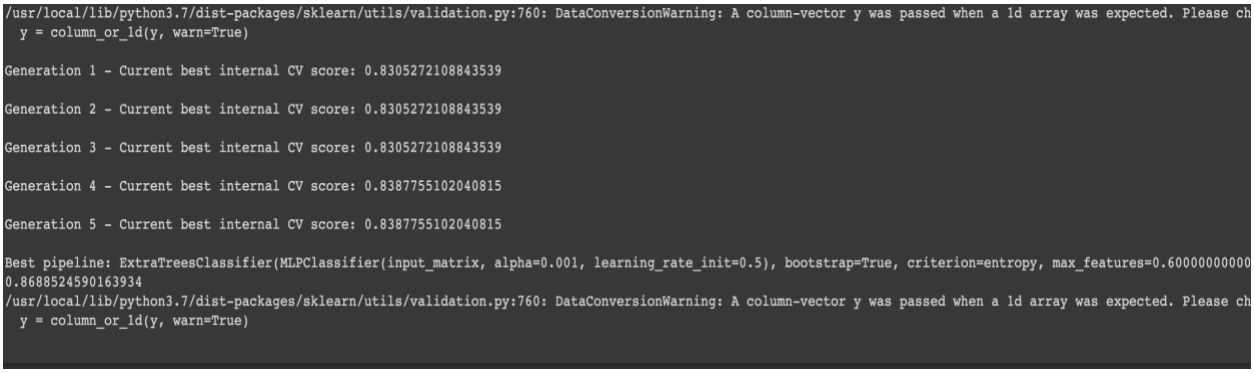
2.4.2 лучшенная выборка, полученная в результате полной предобработки данных в пункте 2 (результат)

```
X_1=df1[['chol','age','thalachh','oldpeak','thall_2','sex_1','exng_1','trtbps']]
Y_1=df1[['output']]

scaler = StandardScaler()

X_11 = scaler.fit_transform(X_1)

X_train, X_test, y_train, y_test = train_test_split(X_11,Y_1, test_size = 0.2, random_state = 42)

clf = svm.SVC(kernel='linear', C=1, random_state=42).fit(X_train,y_train)

y_pred = clf.predict(X_test)

print("The test accuracy score of SVM is ", accuracy_score(y_test, y_pred))
```

```
The test accuracy score of SVM is  0.8524590163934426
```

Почему точность падает после обработки данных? После метода AUTOML я обнаружил, что самая высокая точность составляет 86,88%, такая же точность, как и при отсутствии обработки данных. Когда это возможно, поскольку количество данных и индикаторов в этом наборе данных невелико, все данные более важны.

2.4.3 AutoML(результат)



Список литературы

[1] Гапанюк Ю. Е. Домашнее задание по дисциплине «Методы машинного обучения»[Электронный ресурс] // GitHub. — 2019. — Режим доступа: https://github.com/ugapanyuk/ml_course/wiki/MMO_DZ (дата обращения: 06.05.2019).

[2] You are my Sunshine [Electronic resource] // Space Apps Challenge. — 2017. Access mode: https://2017.spaceappschallenge.org/challenges/earth-and-us/ you-are-my-sunshine/details (online; accessed: 22.02.2019).

[3] dronio. Solar Radiation Prediction [Electronic resource] // Kaggle. — 2017. — Access mode: https://www.kaggle.com/dronio/SolarEnergy (online; accessed: 18.02.2019).

[4] Team The IPython Development. IPython 7.3.0 Documentation [Electronic resource] //Read the Docs. — 2019. — Access mode: https://ipython.readthedocs.io/en/stable/ (online; accessed: 20.02.2019).

[5] Waskom M. seaborn 0.9.0 documentation [Electronic resource] // PyData. — 2018. Access mode: https://seaborn.pydata.org/ (online; accessed: 20.02.2019).

[6] pandas 0.24.1 documentation [Electronic resource] // PyData. — 2019. — Access mode:http://pandas.pydata.org/pandas-docs/stable/ (online; accessed: 20.02.2019).

[7] Chrétien M. Convert datetime.time to seconds [Electronic resource] // Stack Overflow. 2017. — Access mode: https://stackoverflow.com/a/44823381 (online; accessed:20.02.2019).