Московский государственный технический университет им. Н.Э. Баумана

Факультет «Информатика и системы управления»

Кафедра «Системы обработки информации и управления»

Курс «Технологии машинного обучения»

Рубежный контроль № 2
«Методы построения моделей машинного обучения»

Выполнил:

студент группы ИУ5-65Б

Герасименко А.В.

Подпись и дата:

Проверил:

преподаватель каф. ИУ5

Гапанюк Ю.Е.

Подпись и дата:

г. Москва, 2021 г.

Рубежный контроль №2

Методы построения моделей машинного обучения

Задание

Для заданного набора данных постройте модели классификации или регрессии (в зависимости от конкретной задачи, рассматриваемой в наборе данных). Для построения моделей используйте методы 1 и 2. Оцените качество моделей на основе подходящих метрик качества (не менее двух метрик). Какие метрики качества Вы использовали и почему? Какие выводы Вы можете сделать о качестве построенных моделей? Для построения моделей необходимо выполнить требуемую предобработку данных: заполнение пропусков, кодирование категориальных признаков и так далее. При решении задач можно выбирать любое подмножество признаков из приведенного набора данных. Для сокращения времени построения моделей можно использовать фрагмент набора данных (например, первые 200-500 строк).

- Номер варианта: 4
- Метод №1: Метод опорных векторов
- Метод №2: Градиентный бустинг
- https://www.kaggle.com/carlolepelaars/toy-dataset

Импорт библиотек

In [1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_absolute_error, r2_score
from sklearn.svm import SVR
from sklearn.ensemble import GradientBoostingRegressor
import matplotlib.pyplot as plt
```

Подготовка датасета

In [2]:
```
data = pd.read_csv('toy.csv', sep = ';')
del data['Unnamed: 6'] #Удаляем, так как это пустой столбец
del data['City'] #Удаляем, так как этот столбец содержит все значения равные "Dallas"
data.head()
```

Out[2]:

	Number	Gender	Age	Income	Illness
0	1	Male	41	40367.0	No
1	2	Male	54	45084.0	No
2	3	Male	42	52483.0	No
3	4	Male	40	40941.0	No
4	5	Male	46	50289.0	No

In [3]:
```
data.dtypes
```

Out[3]:
```
Number       int64
Gender      object
Age          int64
Income     float64
Illness     object
dtype: object
```

In [4]:
```
data['Gender'].value_counts()
```

Out[4]:
```
Male      145
Female    119
Name: Gender, dtype: int64
```

In [5]:
```
data['Illness'].value_counts()
```

Out[5]:
```
No     240
Yes     24
Name: Illness, dtype: int64
```

Кодирование категориальных признаков

In [6]:
```
data['Gender_1']=data.Gender.replace({'Female':0,'Male':1})
data.drop('Gender', axis = 1, inplace = True)
data['Illness_1']=data.Illness.replace({'No':0,'Yes':1})
data.drop('Illness', axis = 1, inplace = True)
data.head()
```

Out[6]:

	Number	Age	Income	Gender_1	Illness_1
0	1	41	40367.0	1	0
1	2	54	45084.0	1	0
2	3	42	52483.0	1	0
3	4	40	40941.0	1	0
4	5	46	50289.0	1	0

In [7]:
```
data.dtypes
```

Out[7]:
```
Number          int64
Age             int64
Income        float64
Gender_1        int64
Illness_1       int64
dtype: object
```

Проверим, есть ли пропущенные значения

In [8]:
```
data.isnull().sum()
```

Out[8]:
```
Number       0
Age          0
Income       0
Gender_1     0
Illness_1    0
dtype: int64
```

In [9]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 264 entries, 0 to 263
Data columns (total 5 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   Number      264 non-null    int64  
 1   Age         264 non-null    int64  
 2   Income      264 non-null    float64
 3   Gender_1    264 non-null    int64  
 4   Illness_1   264 non-null    int64  
dtypes: float64(1), int64(4)
memory usage: 10.4 KB
```

Построим корреляционную матрицу

In [10]:
```
fig, ax = plt.subplots(figsize=(10,5))
sns.heatmap(data.corr(method='pearson'), ax=ax, cmap = 'Purples', annot=True, fmt='.3f')
```

Out[10]:
```
<AxesSubplot:>
```

Разделим выборку на обучающую и тестовую

Разделим данные на целевой столбец и признаки

In [11]:
```
X = data.drop("Gender_1",axis=1)
Y = data["Gender_1"]
```

In [12]:
```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.25, random_state = 1)
X_train.shape, X_test.shape, Y_train.shape, Y_test.shape
```

Out[12]:
```
((198, 4), (66, 4), (198,), (66,))
```

Метод опорных векторов

In [13]:
```
def print_metrics(X_train, Y_train, X_test, Y_test, clf):
    clf.fit(X_train, Y_train)
    target = clf.predict(X_test)
    print(f'Средняя абсолютная ошибка: {mean_absolute_error(Y_test, target)}')
    print(f'Коэффициент детерминации: {r2_score(Y_test, target)}')
```

In [14]:
```
print_metrics(X_train, Y_train, X_test, Y_test, SVR())
```

```
Средняя абсолютная ошибка: 0.3632601576687834
Коэффициент детерминации: 0.15286739529929083
```

Метод градиентного бустинга

In [15]:
```
def print_metrics(X_train, Y_train, X_test, Y_test, clf):
    clf.fit(X_train, Y_train)
    target = clf.predict(X_test)
    print(f'Средняя абсолютная ошибка: {mean_absolute_error(Y_test, target)}')
    print(f'Коэффициент детерминации: {r2_score(Y_test, target)}')
```

In [16]:
```
print_metrics(X_train, Y_train, X_test, Y_test, GradientBoostingRegressor(random_state=0))
```

```
Средняя абсолютная ошибка: 0.3991059157733054
Коэффициент детерминации: 0.06933835489495876
```

Выводы:

В РК были использованы метрики:

Mean absolute error - средняя абсолютная ошибка

R2 (коэффициент детерминации) позволяет оценить общее качество модели, чем R2 ближе к 1, тем модель лучше.

Метод градиентного бустинга и метод опорных векторов показали себя примерно одинакого по отношению к данной модели. В выборке слабая связность датасета. Поэтому метрика R2 мала у метода опорных векторов и у метода градинтного бустинга.