Московский государственный технический университет им. Н.Э. Баумана

Факультет «Информатика и системы управления»
Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Технологии машинного обучения»

Отчет по Рубежному контролю №2

Вариант №15

Выполнил:
студент группы ИУ5-63
Миронова Александра
Подпись и дата:
25.05.22

Проверил:
Юрий Евгеньевич Гапанюк
Подпись и дата:

Москва, 2022 г.

Задание:

Для заданного набора данных построить модели классификации или регрессии (в зависимости от конкретной задачи, рассматриваемой в наборе данных).

Для построения моделей использовать методы 1 и 2. Для построения моделей необходимо выполнить требуемую предобработку данных: заполнение пропусков, кодирование категориальных признаков, и т.д.

Оценить качество моделей на основе подходящих метрик качества (не менее двух метрик). Объяснить выбор метрик качества.

Сделать выводы о качестве построенных моделей?

Группа	Метод №1	Метод №2
ИУ5-63Б, ИУ5Ц-83Б	Дерево решений	Случайный лес

Результат:

RK2.md

Загрузка и первичный анализ данных

Для выполнения задания был предложен датасет с данными об образовании в США .

https://www.kaggle.com/noriuk/us-education-datasets-unification-project (файл states_all_extended.csv)

```
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.preprocessing import OrdinalEncoder
import numpy as np
import pandas as pd
import seaborn as sns
import math
import matplotlib.pyplot as plt
import matplotlib.ticker as ticker
%matplotlib inline
sns.set(style="ticks")


data = pd.read_csv('dataset/states_all_extended.csv', sep=",")
data.shape
```

```
(1715, 266)
```

```
data.dtypes
```

```
PRIMARY_KEY             object
STATE                   object
YEAR                     int64
ENROLL                 float64
TOTAL_REVENUE          float64
                         ...
G08_AM_A_MATHEMATICS   float64
G08_HP_A_READING       float64
G08_HP_A_MATHEMATICS   float64
G08_TR_A_READING       float64
```

```
G08_TR_A_MATHEMATICS    float64
Length: 266, dtype: object
```

```
data
```

<style scoped> .dataframe tbody tr th:only-of-type { vertical-align: middle; }

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

</style>

	PRIMARY_KEY	STATE	YEAR	ENROLL	TOTAL_REVENUE	FEDERAL_REVENUE
0	1992_ALABAMA	ALABAMA	1992	NaN	2678885.0	304177.0
1	1992_ALASKA	ALASKA	1992	NaN	1049591.0	106780.0
2	1992_ARIZONA	ARIZONA	1992	NaN	3258079.0	297888.0
3	1992_ARKANSAS	ARKANSAS	1992	NaN	1711959.0	178571.0
4	1992_CALIFORNIA	CALIFORNIA	1992	NaN	26260025.0	2072470.0
...	...	...	...	...	...	...
1710	2019_VIRGINIA	VIRGINIA	2019	NaN	NaN	NaN
1711	2019_WASHINGTON	WASHINGTON	2019	NaN	NaN	NaN
1712	2019_WEST_VIRGINIA	WEST_VIRGINIA	2019	NaN	NaN	NaN
1713	2019_WISCONSIN	WISCONSIN	2019	NaN	NaN	NaN
1714	2019_WYOMING	WYOMING	2019	NaN	NaN	NaN

1715 rows × 266 columns

Колонки датасета

- PRIMARY_KEY - Комбинация года и названия штата
- YEAR
- STATE - Название штата
- ENROLL - Количество учащихся в штате

- TOTAL REVENUE - Общий доход
- FEDERAL_REVENUE - Федеральный доход
- STATE_REVENUE - Доход штата
- LOCAL_REVENUE - Доход местного управления
- TOTAL_EXPENDITURE - Общий расход
- INSTRUCTION_EXPENDITURE - Расход на образование
- SUPPORT_SERVICES_EXPENDITURE - Расход на вспомогательные услуги
- CAPITAL_OUTLAY_EXPENDITURE - Капитальный расход
- OTHER_EXPENDITURE - Прочие расходы

- A_A_A - Общее количество учащихся в штате
- G01_A_A - G12_A_A - Количество учащихся в 1 - 12 классе
- G01-G08_A_A - Количество учащихся с 1 по 8 класс
- G09-G12_A_A - Количество учащихся с 9 по 12 класс
- G06_AS_M - Число учащихся 6 класса мужского пола, этническая принадлежность которых "азиатская"
- G08_AS_A_READING - Средний балл чтения учащихся 8 класса, этническая принадлежность которых "азиатская"
- PK_A_A - Количество учащихся в ясельной группе детского сада
- KG_A_A - Количество учащихся в старшей группе детского сада

Проверим есть ли пропущенные значения

```
# Цикл по колонкам датасета с записью колонок с более 76% пустых значений
print('{:30} {:10} {}'.format('Колонка', 'Тип данных', 'Количество пустых значений'))
cat_cols = []
for col in data.columns:
    # Количество пустых значений
    temp_null_count = data[data[col].isnull()].shape[0]
    dt = str(data[col].dtype)
    if temp_null_count > 0:
        temp_perc = round((temp_null_count / data.shape[0]) * 100.0, 2)
        if temp_perc > 76:
            cat_cols.append(col)
        print('{:30} {:10} {} {:10}%'.format(col, dt, temp_null_count, temp_perc))
```

```
Колонка                        Тип данных Количество пустых значений
ENROLL                         float64    491      28.63%
TOTAL_REVENUE                  float64    440      25.66%
FEDERAL_REVENUE                float64    440      25.66%
STATE_REVENUE                  float64    440      25.66%
LOCAL_REVENUE                  float64    440      25.66%
TOTAL_EXPENDITURE              float64    440      25.66%
INSTRUCTION_EXPENDITURE        float64    440      25.66%
SUPPORT_SERVICES_EXPENDITURE   float64    440      25.66%
```

```
OTHER_EXPENDITURE              float64    491      28.63%
CAPITAL_OUTLAY_EXPENDITURE     float64    440      25.66%
A_A_A                          float64    83       4.84%
G01_A_A                        float64    83       4.84%
G02_A_A                        float64    83       4.84%
G03_A_A                        float64    83       4.84%
G04_A_A                        float64    83       4.84%
G05_A_A                        float64    83       4.84%
G06_A_A                        float64    83       4.84%
G07_A_A                        float64    83       4.84%
G08_A_A                        float64    83       4.84%
G09_A_A                        float64    83       4.84%
G10_A_A                        float64    83       4.84%
G11_A_A                        float64    83       4.84%
G12_A_A                        float64    83       4.84%
KG_A_A                         float64    83       4.84%
PK_A_A                         float64    173      10.09%
G01-G08_A_A                    float64    695      40.52%
G09-G12_A_A                    float64    644      37.55%
G01_AM_F                       float64    1308      76.27%
G01_AM_M                       float64    1307      76.21%
G01_AS_F                       float64    1307      76.21%
G01_AS_M                       float64    1307      76.21%
G01_BL_F                       float64    1307      76.21%
G01_BL_M                       float64    1307      76.21%
G01_HI_F                       float64    1308      76.27%
G01_HI_M                       float64    1307      76.21%
G01_HP_F                       float64    1351      78.78%
G01_HP_M                       float64    1352      78.83%
G01_TR_F                       float64    1344      78.37%
G01_TR_M                       float64    1344      78.37%
G01_WH_F                       float64    1307      76.21%
G01_WH_M                       float64    1307      76.21%
G02_AM_F                       float64    1309      76.33%
G02_AM_M                       float64    1308      76.27%
G02_AS_F                       float64    1307      76.21%
G02_AS_M                       float64    1307      76.21%
G02_BL_F                       float64    1307      76.21%
G02_BL_M                       float64    1307      76.21%
G02_HI_F                       float64    1307      76.21%
G02_HI_M                       float64    1307      76.21%
G02_HP_F                       float64    1351      78.78%
G02_HP_M                       float64    1351      78.78%
G02_TR_F                       float64    1344      78.37%
G02_TR_M                       float64    1344      78.37%
G02_WH_F                       float64    1307      76.21%
G02_WH_M                       float64    1307      76.21%
G03_AM_F                       float64    1308      76.27%
G03_AM_M                       float64    1309      76.33%
G03_AS_F                       float64    1307      76.21%
G03_AS_M                       float64    1307      76.21%
G03_BL_F                       float64    1307      76.21%
G03_BL_M                       float64    1307      76.21%
```

```
G03_HI_F                float64    1307      76.21%
G03_HI_M                float64    1307      76.21%
G03_HP_F                float64    1352      78.83%
G03_HP_M                float64    1349      78.66%
G03_TR_F                float64    1344      78.37%
G03_TR_M                float64    1344      78.37%
G03_WH_F                float64    1307      76.21%
G03_WH_M                float64    1307      76.21%
G04_AM_F                float64    1308      76.27%
G04_AM_M                float64    1308      76.27%
G04_AS_F                float64    1307      76.21%
G04_AS_M                float64    1307      76.21%
G04_BL_F                float64    1307      76.21%
G04_BL_M                float64    1307      76.21%
G04_HI_F                float64    1307      76.21%
G04_HI_M                float64    1307      76.21%
G04_HP_F                float64    1351      78.78%
G04_HP_M                float64    1351      78.78%
G04_TR_F                float64    1344      78.37%
G04_TR_M                float64    1344      78.37%
G04_WH_F                float64    1307      76.21%
G04_WH_M                float64    1307      76.21%
G05_AM_F                float64    1308      76.27%
G05_AM_M                float64    1308      76.27%
G05_AS_F                float64    1307      76.21%
G05_AS_M                float64    1307      76.21%
G05_BL_F                float64    1307      76.21%
G05_BL_M                float64    1307      76.21%
G05_HI_F                float64    1307      76.21%
G05_HI_M                float64    1307      76.21%
G05_HP_F                float64    1352      78.83%
G05_HP_M                float64    1349      78.66%
G05_TR_F                float64    1344      78.37%
G05_TR_M                float64    1344      78.37%
G05_WH_F                float64    1307      76.21%
G05_WH_M                float64    1307      76.21%
G06_AM_F                float64    1307      76.21%
G06_AM_M                float64    1308      76.27%
G06_AS_F                float64    1307      76.21%
G06_AS_M                float64    1307      76.21%
G06_BL_F                float64    1307      76.21%
G06_BL_M                float64    1307      76.21%
G06_HI_F                float64    1307      76.21%
G06_HI_M                float64    1307      76.21%
G06_HP_F                float64    1351      78.78%
G06_HP_M                float64    1349      78.66%
G06_TR_F                float64    1344      78.37%
G06_TR_M                float64    1344      78.37%
G06_WH_F                float64    1307      76.21%
G06_WH_M                float64    1307      76.21%
G07_AM_F                float64    1307      76.21%
G07_AM_M                float64    1308      76.27%
G07_AS_F                float64    1307      76.21%
```

G07_AS_M	float64	1307	76.21%
G07_BL_F	float64	1307	76.21%
G07_BL_M	float64	1307	76.21%
G07_HI_F	float64	1307	76.21%
G07_HI_M	float64	1307	76.21%
G07_HP_F	float64	1350	78.72%
G07_HP_M	float64	1352	78.83%
G07_TR_F	float64	1344	78.37%
G07_TR_M	float64	1344	78.37%
G07_WH_F	float64	1307	76.21%
G07_WH_M	float64	1307	76.21%
G08_AM_F	float64	1308	76.27%
G08_AM_M	float64	1307	76.21%
G08_AS_F	float64	1307	76.21%
G08_AS_M	float64	1307	76.21%
G08_BL_F	float64	1307	76.21%
G08_BL_M	float64	1307	76.21%
G08_HI_F	float64	1307	76.21%
G08_HI_M	float64	1307	76.21%
G08_HP_F	float64	1350	78.72%
G08_HP_M	float64	1349	78.66%
G08_TR_F	float64	1344	78.37%
G08_TR_M	float64	1344	78.37%
G08_WH_F	float64	1307	76.21%
G08_WH_M	float64	1307	76.21%
G09_AM_F	float64	1307	76.21%
G09_AM_M	float64	1308	76.27%
G09_AS_F	float64	1307	76.21%
G09_AS_M	float64	1307	76.21%
G09_BL_F	float64	1307	76.21%
G09_BL_M	float64	1307	76.21%
G09_HI_F	float64	1307	76.21%
G09_HI_M	float64	1307	76.21%
G09_HP_F	float64	1352	78.83%
G09_HP_M	float64	1351	78.78%
G09_TR_F	float64	1344	78.37%
G09_TR_M	float64	1344	78.37%
G09_WH_F	float64	1307	76.21%
G09_WH_M	float64	1307	76.21%
G10_AM_F	float64	1307	76.21%
G10_AM_M	float64	1308	76.27%
G10_AS_F	float64	1307	76.21%
G10_AS_M	float64	1307	76.21%
G10_BL_F	float64	1307	76.21%
G10_BL_M	float64	1307	76.21%
G10_HI_F	float64	1307	76.21%
G10_HI_M	float64	1307	76.21%
G10_HP_F	float64	1352	78.83%
G10_HP_M	float64	1351	78.78%
G10_TR_F	float64	1344	78.37%
G10_TR_M	float64	1344	78.37%
G10_WH_F	float64	1307	76.21%
G10_WH_M	float64	1307	76.21%

```
G11_AM_F                float64    1307      76.21%
G11_AM_M                float64    1307      76.21%
G11_AS_F                float64    1307      76.21%
G11_AS_M                float64    1307      76.21%
G11_BL_F                float64    1307      76.21%
G11_BL_M                float64    1307      76.21%
G11_HI_F                float64    1308      76.27%
G11_HI_M                float64    1307      76.21%
G11_HP_F                float64    1352      78.83%
G11_HP_M                float64    1354      78.95%
G11_TR_F                float64    1344      78.37%
G11_TR_M                float64    1344      78.37%
G11_WH_F                float64    1307      76.21%
G11_WH_M                float64    1307      76.21%
G12_AM_F                float64    1309      76.33%
G12_AM_M                float64    1310      76.38%
G12_AS_F                float64    1307      76.21%
G12_AS_M                float64    1307      76.21%
G12_BL_F                float64    1307      76.21%
G12_BL_M                float64    1307      76.21%
G12_HI_F                float64    1307      76.21%
G12_HI_M                float64    1307      76.21%
G12_HP_F                float64    1352      78.83%
G12_HP_M                float64    1352      78.83%
G12_TR_F                float64    1344      78.37%
G12_TR_M                float64    1344      78.37%
G12_WH_F                float64    1307      76.21%
G12_WH_M                float64    1307      76.21%
KG_AM_F                 float64    1307      76.21%
KG_AM_M                 float64    1308      76.27%
KG_AS_F                 float64    1307      76.21%
KG_AS_M                 float64    1307      76.21%
KG_BL_F                 float64    1307      76.21%
KG_BL_M                 float64    1307      76.21%
KG_HI_F                 float64    1308      76.27%
KG_HI_M                 float64    1307      76.21%
KG_HP_F                 float64    1349      78.66%
KG_HP_M                 float64    1350      78.72%
KG_TR_F                 float64    1344      78.37%
KG_TR_M                 float64    1344      78.37%
KG_WH_F                 float64    1307      76.21%
KG_WH_M                 float64    1307      76.21%
PK_AM_F                 float64    1332      77.67%
PK_AM_M                 float64    1321      77.03%
PK_AS_F                 float64    1321      77.03%
PK_AS_M                 float64    1323      77.14%
PK_BL_F                 float64    1321      77.03%
PK_BL_M                 float64    1321      77.03%
PK_HI_F                 float64    1321      77.03%
PK_HI_M                 float64    1321      77.03%
PK_HP_F                 float64    1387      80.87%
PK_HP_M                 float64    1384       80.7%
PK_TR_F                 float64    1357      79.13%
```

```
PK_TR_M                 float64    1357      79.13%
PK_WH_F                 float64    1321      77.03%
PK_WH_M                 float64    1321      77.03%
G04_A_A_READING         float64    1065       62.1%
G04_A_A_MATHEMATICS     float64    1150      67.06%
G04_A_M_READING         float64    1065       62.1%
G04_A_M_MATHEMATICS     float64    1150      67.06%
G04_A_F_READING         float64    1065       62.1%
G04_A_F_MATHEMATICS     float64    1150      67.06%
G04_WH_A_READING        float64    1450      84.55%
G04_WH_A_MATHEMATICS    float64    1450      84.55%
G04_BL_A_READING        float64    1489      86.82%
G04_BL_A_MATHEMATICS    float64    1486      86.65%
G04_HI_A_READING        float64    1465      85.42%
G04_HI_A_MATHEMATICS    float64    1465      85.42%
G04_AS_A_READING        float64    1551      90.44%
G04_AS_A_MATHEMATICS    float64    1547       90.2%
G04_AM_A_READING        float64    1651      96.27%
G04_AM_A_MATHEMATICS    float64    1652      96.33%
G04_HP_A_READING        float64    1699      99.07%
G04_HP_A_MATHEMATICS    float64    1700      99.13%
G04_TR_A_READING        float64    1532      89.33%
G04_TR_A_MATHEMATICS    float64    1532      89.33%
G08_A_A_READING         float64    1153      67.23%
G08_A_A_MATHEMATICS     float64    1113       64.9%
G08_A_M_READING         float64    1153      67.23%
G08_A_M_MATHEMATICS     float64    1113       64.9%
G08_A_F_READING         float64    1153      67.23%
G08_A_F_MATHEMATICS     float64    1113       64.9%
G08_WH_A_READING        float64    1450      84.55%
G08_WH_A_MATHEMATICS    float64    1450      84.55%
G08_BL_A_READING        float64    1493      87.06%
G08_BL_A_MATHEMATICS    float64    1494      87.11%
G08_HI_A_READING        float64    1469      85.66%
G08_HI_A_MATHEMATICS    float64    1467      85.54%
G08_AS_A_READING        float64    1562      91.08%
G08_AS_A_MATHEMATICS    float64    1558      90.85%
G08_AM_A_READING        float64    1654      96.44%
G08_AM_A_MATHEMATICS    float64    1655       96.5%
G08_HP_A_READING        float64    1701      99.18%
G08_HP_A_MATHEMATICS    float64    1702      99.24%
G08_TR_A_READING        float64    1574      91.78%
G08_TR_A_MATHEMATICS    float64    1570      91.55%
```

Имеется достаточное количество пропусков. Переходим к их обработке.

Обработка пропусков

Все столбцы с данными о количестве учеников в классах и группах детского сада с учетом рассовой принадлежности имеют количество пропусков более 76%. Они не являются важными для обучения моделей и решения задачи регрессии. Поэтому избавимся от них. Список таких колонок хранится в массиве cat_cols.

```
data_reshape1 = data.drop(axis=1, columns=cat_cols).copy()
```

Посмотрим какие столбцы с пропущенными значениями остались

```
print('{:30} {:10} {}'.format('Колонка', 'Тип данных', 'Количество пустых значений'))
cat_cols2 = []
for col in data_reshape1.columns:
    # Количество пустых значений
    temp_null_count = data_reshape1[data_reshape1[col].isnull()].shape[0]
    dt = str(data_reshape1[col].dtype)
    if temp_null_count > 0:
        temp_perc = round((temp_null_count / data_reshape1.shape[0]) * 100.0, 2)
        cat_cols2.append(col)
        print('{:30} {:10} {} {:10}%'.format(col, dt, temp_null_count, temp_perc))
```

```
Колонка                        Тип данных Количество пустых значений
ENROLL                         float64    491      28.63%
TOTAL_REVENUE                  float64    440      25.66%
FEDERAL_REVENUE                float64    440      25.66%
STATE_REVENUE                  float64    440      25.66%
LOCAL_REVENUE                  float64    440      25.66%
TOTAL_EXPENDITURE              float64    440      25.66%
INSTRUCTION_EXPENDITURE        float64    440      25.66%
SUPPORT_SERVICES_EXPENDITURE   float64    440      25.66%
OTHER_EXPENDITURE              float64    491      28.63%
CAPITAL_OUTLAY_EXPENDITURE     float64    440      25.66%
A_A_A                          float64    83       4.84%
G01_A_A                        float64    83       4.84%
G02_A_A                        float64    83       4.84%
G03_A_A                        float64    83       4.84%
G04_A_A                        float64    83       4.84%
G05_A_A                        float64    83       4.84%
G06_A_A                        float64    83       4.84%
G07_A_A                        float64    83       4.84%
G08_A_A                        float64    83       4.84%
G09_A_A                        float64    83       4.84%
G10_A_A                        float64    83       4.84%
G11_A_A                        float64    83       4.84%
G12_A_A                        float64    83       4.84%
KG_A_A                         float64    83       4.84%
PK_A_A                         float64    173      10.09%
G01-G08_A_A                    float64    695      40.52%
G09-G12_A_A                    float64    644      37.55%
G04_A_A_READING                float64    1065       62.1%
```

```
G04_A_A_MATHEMATICS           float64    1150      67.06%
G04_A_M_READING               float64    1065       62.1%
G04_A_M_MATHEMATICS           float64    1150      67.06%
G04_A_F_READING               float64    1065       62.1%
G04_A_F_MATHEMATICS           float64    1150      67.06%
G08_A_A_READING               float64    1153      67.23%
G08_A_A_MATHEMATICS           float64    1113       64.9%
G08_A_M_READING               float64    1153      67.23%
G08_A_M_MATHEMATICS           float64    1113       64.9%
G08_A_F_READING               float64    1153      67.23%
G08_A_F_MATHEMATICS           float64    1113       64.9%
```

На данном этапе количество столбцов заметно уменьшилось ив основном остались наиболее полные и информативные. На основании данных датасета мы можем сормулировать задачу регрессии, которая заключается в обучении модели для выявления зависимости оценок по предметам учеников 4 класа от общих затрат на образование.

Следовательно, целевыми признаками будут являться столбцы с оценками(баллами) по чтению и математике 4 класса.

В этих столбцах много пропусков, но их удалять нельзя, поэтому удалим строки с пропущенными в этих столбцах значениями.

```
data_new = data_reshape1.dropna(axis=0, subset=cat_cols2).copy()
print(data_new.shape)
```

```
(355, 42)
```

```
data_new.drop(['A_A_A','G01_A_A','G02_A_A','G03_A_A','G05_A_A','G06_A_A','G07_A_A','G08_A_A','G09_A_A','G10_A
```

```
data_new.drop(['G01-G08_A_A','G09-G12_A_A','G04_A_M_READING','G04_A_M_MATHEMATICS','G04_A_F_READING','G04_A_F
```

```
data_new.drop(['G08_A_A_READING','G08_A_A_MATHEMATICS','G08_A_M_READING','G08_A_M_MATHEMATICS','G08_A_F_READI
```

Удалим столбец PRIMARY_KEY - тк в нем нет необходимости.

```
data_new.drop(['PRIMARY_KEY'], inplace=True, axis=1)
```

```
print('{:30} {:10} {}'.format('Колонка', 'Тип данных', 'Количество пустых значений'))
for col in data_new.columns:
    # Количество пустых значений
    temp_null_count = data_new[data_new[col].isnull()].shape[0]
    dt = str(data_new[col].dtype)
    temp_perc = round((temp_null_count / data_reshape1.shape[0]) * 100.0, 2)
    print('{:30} {:10} {} {:10}%'.format(col, dt, temp_null_count, temp_perc))
```

```
Колонка                        Тип данных Количество пустых значений
STATE                          object     0        0.0%
YEAR                           int64      0        0.0%
ENROLL                         float64    0        0.0%
TOTAL_REVENUE                  float64    0        0.0%
FEDERAL_REVENUE                float64    0        0.0%
STATE_REVENUE                  float64    0        0.0%
LOCAL_REVENUE                  float64    0        0.0%
TOTAL_EXPENDITURE              float64    0        0.0%
INSTRUCTION_EXPENDITURE        float64    0        0.0%
SUPPORT_SERVICES_EXPENDITURE   float64    0        0.0%
OTHER_EXPENDITURE              float64    0        0.0%
CAPITAL_OUTLAY_EXPENDITURE     float64    0        0.0%
G04_A_A                        float64    0        0.0%
G04_A_A_READING                float64    0        0.0%
G04_A_A_MATHEMATICS            float64    0        0.0%
```

```
data_new
```

<style scoped> .dataframe tbody tr th:only-of-type { vertical-align: middle; }

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

</style>

	STATE	YEAR	ENROLL	TOTAL_REVENUE	FEDERAL_REVENUE	STATE_REVENUE
561	ALABAMA	2003	727900.0	5196054.0	567704.0	2966981.0
562	ALASKA	2003	133303.0	1425948.0	259423.0	813371.0
563	ARIZONA	2003	875111.0	6529894.0	740579.0	2912629.0
564	ARKANSAS	2003	450158.0	3241275.0	379947.0	2394336.0

	STATE	YEAR	ENROLL	TOTAL_REVENUE	FEDERAL_REVENUE	STATE_REVENUE
565	CALIFORNIA	2003	6226552.0	59815855.0	5795655.0	33617766.0
...	...	...	...	...	...	...
1219	VIRGINIA	2015	1279867.0	15857524.0	1012205.0	6240349.0
1220	WASHINGTON	2015	1072359.0	13709442.0	1036422.0	8293812.0
1221	WEST_VIRGINIA	2015	279565.0	3478401.0	362959.0	1979466.0
1222	WISCONSIN	2015	861813.0	11637376.0	814385.0	5869265.0
1223	WYOMING	2015	93867.0	1962874.0	120290.0	1116917.0

355 rows × 15 columns

Теперь пропусков в данных нет.

Таким образом, получен фрагмент датасета размером в 355 строк и 15 колонок, допустимый по условию задачи. В датасете имеется один категориальный признак - столбец STATE. Закодируем его.

```
data_oe = data_new[['STATE']]
```

```
for col in range(0,100):
    print(data_new.iloc[col]['STATE'])
```

```
ALABAMA
ALASKA
ARIZONA
ARKANSAS
CALIFORNIA
COLORADO
CONNECTICUT
DELAWARE
DISTRICT_OF_COLUMBIA
FLORIDA
GEORGIA
HAWAII
IDAHO
ILLINOIS
INDIANA
IOWA
KANSAS
KENTUCKY
LOUISIANA
MAINE
```

```
MARYLAND
MASSACHUSETTS
MICHIGAN
MINNESOTA
MISSISSIPPI
MISSOURI
MONTANA
NEBRASKA
NEVADA
NEW_HAMPSHIRE
NEW_JERSEY
NEW_MEXICO
NEW_YORK
NORTH_CAROLINA
NORTH_DAKOTA
OHIO
OKLAHOMA
OREGON
PENNSYLVANIA
RHODE_ISLAND
SOUTH_CAROLINA
SOUTH_DAKOTA
TENNESSEE
TEXAS
UTAH
VERMONT
VIRGINIA
WASHINGTON
WEST_VIRGINIA
WISCONSIN
WYOMING
ALABAMA
ALASKA
ARIZONA
ARKANSAS
CALIFORNIA
COLORADO
CONNECTICUT
DELAWARE
DISTRICT_OF_COLUMBIA
FLORIDA
GEORGIA
HAWAII
IDAHO
ILLINOIS
INDIANA
IOWA
KANSAS
KENTUCKY
LOUISIANA
MAINE
MARYLAND
MASSACHUSETTS
```

```
MICHIGAN
MINNESOTA
MISSISSIPPI
MISSOURI
MONTANA
NEBRASKA
NEVADA
NEW_HAMPSHIRE
NEW_JERSEY
NEW_MEXICO
NEW_YORK
NORTH_CAROLINA
NORTH_DAKOTA
OHIO
OKLAHOMA
OREGON
PENNSYLVANIA
RHODE_ISLAND
SOUTH_CAROLINA
SOUTH_DAKOTA
TENNESSEE
TEXAS
UTAH
VERMONT
VIRGINIA
WASHINGTON
WEST_VIRGINIA
```

```
oe = OrdinalEncoder()
data_new['STATE'] = oe.fit_transform(data_new)
data_new
```

<style scoped> .dataframe tbody tr th:only-of-type { vertical-align: middle; }

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

</style>

	STATE	YEAR	ENROLL	TOTAL_REVENUE	FEDERAL_REVENUE	STATE_REVENUE	LOCAL_
561	0.0	2003	727900.0	5196054.0	567704.0	2966981.0	166136
562	1.0	2003	133303.0	1425948.0	259423.0	813371.0	353154.

	STATE	YEAR	ENROLL	TOTAL_REVENUE	FEDERAL_REVENUE	STATE_REVENUE	LOCAL_
563	2.0	2003	875111.0	6529894.0	740579.0	2912629.0	287668(
564	3.0	2003	450158.0	3241275.0	379947.0	2394336.0	466992.
565	4.0	2003	6226552.0	59815855.0	5795655.0	33617766.0	204024:
...	...	...	...	...	...	...	...
1219	46.0	2015	1279867.0	15857524.0	1012205.0	6240349.0	860497(
1220	47.0	2015	1072359.0	13709442.0	1036422.0	8293812.0	437920(
1221	48.0	2015	279565.0	3478401.0	362959.0	1979466.0	113597(
1222	49.0	2015	861813.0	11637376.0	814385.0	5869265.0	495372(
1223	50.0	2015	93867.0	1962874.0	120290.0	1116917.0	725667.

355 rows × 15 columns

Введем столбец со средним баллом учеников 4 класса за оба предмета: математику и чтение

```
data_new['G04_TEST'] = data_new.apply (lambda row: (row['G04_A_A_READING']+row['G04_A_A_MATHEMATICS'])/2, axi
```

```
data_new.tail()
```

<style scoped> .dataframe tbody tr th:only-of-type { vertical-align: middle; }

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

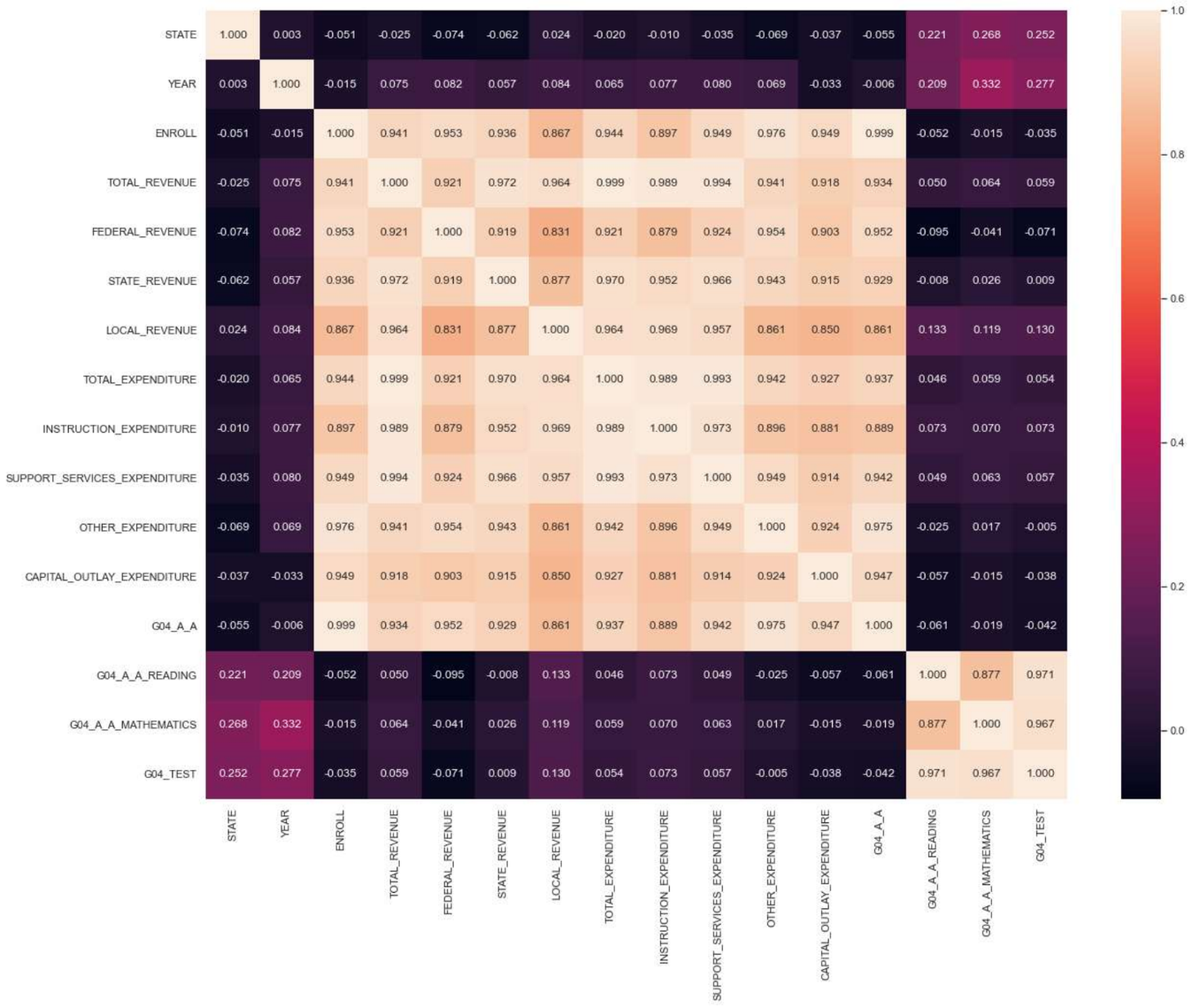
</style>

	STATE	YEAR	ENROLL	TOTAL_REVENUE	FEDERAL_REVENUE	STATE_REVENUE	LOCAL_
1219	46.0	2015	1279867.0	15857524.0	1012205.0	6240349.0	860497(
1220	47.0	2015	1072359.0	13709442.0	1036422.0	8293812.0	437920(
1221	48.0	2015	279565.0	3478401.0	362959.0	1979466.0	113597(

	STATE	YEAR	ENROLL	TOTAL_REVENUE	FEDERAL_REVENUE	STATE_REVENUE	LOCAL_
1222	49.0	2015	861813.0	11637376.0	814385.0	5869265.0	495372(
1223	50.0	2015	93867.0	1962874.0	120290.0	1116917.0	725667.

Корелляционная матрица

```
with sns.axes_style("white"):
    f, ax = plt.subplots(figsize=(20, 15))
    ax = sns.heatmap(data_new.corr(), annot=True, fmt='.3f')
```



Разделение выборки на обучающую и тестовую

```
y=np.array(data_new["G04_TEST"])
X=np.array(data_new.drop(["G04_TEST"], axis=1))
X, y
```

```
(array([[0.00000e+00, 2.00300e+03, 7.27900e+05, ..., 5.73230e+04,
         2.07000e+02, 2.23000e+02],
        [1.00000e+00, 2.00300e+03, 1.33303e+05, ..., 1.01150e+04,
         2.12000e+02, 2.33000e+02],
        [2.00000e+00, 2.00300e+03, 8.75111e+05, ..., 7.62070e+04,
         2.09000e+02, 2.29000e+02],
        ...,
        [4.80000e+01, 2.01500e+03, 2.79565e+05, ..., 1.98140e+04,
         2.16000e+02, 2.35000e+02],
        [4.90000e+01, 2.01500e+03, 8.61813e+05, ..., 6.09990e+04,
         2.23000e+02, 2.43000e+02],
        [5.00000e+01, 2.01500e+03, 9.38670e+04, ..., 7.55100e+03,
         2.28000e+02, 2.47000e+02]]),
 array([215. , 222.5, 219. , 221.5, 216.5, 229.5, 234.5, 230. , 196.5,
        226. , 222. , 217.5, 226.5, 224.5, 229. , 230.5, 231. , 224. ,
        215.5, 231. , 226. , 235. , 227.5, 232.5, 214. , 228.5, 229.5,
        228.5, 217.5, 235.5, 232. , 213. , 229. , 231.5, 230. , 230. ,
        221.5, 227. , 227.5, 223. , 225.5, 229.5, 220. , 226. , 227. ,
        234. , 231. , 229.5, 225. , 229. , 231.5, 216.5, 223.5, 218.5,
        226.5, 218.5, 231.5, 234. , 233. , 201. , 229. , 224. , 220. ,
        232. , 224.5, 229. , 230.5, 233. , 225.5, 219.5, 233. , 229. ,
        239. , 228. , 235.5, 215.5, 228. , 233. , 229.5, 218.5, 236.5,
        233.5, 215.5, 230.5, 229. , 234. , 232.5, 224. , 227.5, 232. ,
        224.5, 225.5, 232. , 223. , 230.5, 230. , 235.5, 233. , 232.5,
        223. , 231. , 233. , 222.5, 225.5, 221. , 227.5, 219.5, 232. ,
        235. , 233.5, 205.5, 233. , 227. , 223.5, 232. , 228. , 233.5,
        234. , 236.5, 228.5, 218.5, 234. , 232.5, 244. , 229. , 236. ,
        218. , 230. , 235.5, 230.5, 221.5, 239. , 240. , 220. , 233.5,
        230. , 235.5, 235.5, 227. , 225.5, 235. , 227.5, 225.5, 232. ,
        224.5, 231. , 230. , 237. , 235.5, 233.5, 225.5, 233.5, 234.5,
        222. , 224. , 220. , 227. , 221. , 234.5, 237. , 232.5, 210.5,
        234. , 227. , 223.5, 231. , 228.5, 233. , 232. , 234.5, 232.5,
        218. , 234. , 235. , 243. , 227. , 236. , 219. , 232.5, 234.5,
        231. , 223. , 240. , 238. , 219. , 232.5, 231.5, 235.5, 234.5,
        227. , 228. , 234. , 231. , 226. , 232. , 224.5, 229.5, 229.5,
        238.5, 235. , 231.5, 224. , 232. , 232.5, 225.5, 222. , 223.5,
        227.5, 222.5, 233.5, 234.5, 232.5, 211.5, 232.5, 229.5, 226.5,
        230.5, 229. , 232.5, 232. , 235. , 233. , 220.5, 233. , 239. ,
        245. , 227.5, 235.5, 219.5, 230. , 234.5, 231.5, 225. , 241. ,
        239.5, 220.5, 230. , 233. , 235.5, 234. , 226. , 226.5, 236.5,
        232. , 226. , 230.5, 224. , 229.5, 231.5, 237. , 235.5, 232. ,
        224.5, 233. , 234. , 226. , 222.5, 226.5, 229.5, 223.5, 237. ,
        236.5, 234.5, 217.5, 234.5, 231. , 229. , 230. , 229. , 237. ,
        235. , 234.5, 232.5, 220.5, 235.5, 238.5, 242.5, 227. , 240. ,
```

```
        220. , 231. , 233.5, 233. , 225. , 242.5, 238. , 219.5, 232. ,
        233.5, 235. , 235. , 228. , 229.5, 235. , 232. , 225.5, 229.5,
        230. , 229.5, 233. , 238. , 237.5, 235.5, 226. , 233. , 236.5,
        224. , 224.5, 226.5, 226.5, 233. , 234.5, 231.5, 221.5, 235. ,
        229. , 226.5, 230.5, 229.5, 237.5, 233.5, 231. , 235. , 225. ,
        233. , 231. , 243. , 226. , 236.5, 224. , 231. , 233. , 235.5,
        224. , 240.5, 237. , 219. , 230. , 235. , 235. , 234.5, 231. ,
        235. , 231.5, 227.5, 230. , 230. , 231. , 234.5, 236.5, 238. ,
        235.5, 225.5, 233. , 237.5]))
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=1,shuffle=True)
```

Обучение моделей

```
# Дерево решений
DTR = DecisionTreeRegressor(random_state=1)
DTR.fit(X_train, y_train)

DTR_mse = mean_squared_error(y_test, DTR.predict(X_test), squared = True)
DTR_r2 = r2_score(y_test, DTR.predict(X_test))
```

```
# Случайный лес
RF = RandomForestRegressor(random_state=1)
RF.fit(X_train, y_train)

RF_mse = mean_squared_error(y_test, RF.predict(X_test), squared = True)
RF_r2 = r2_score(y_test, RF.predict(X_test))
```

Оценка моделей

Для оценки моделей будем использовать метрики:

Mean squared error - средняя квадратичная ошибка -

```
выбрана для вычисления погрешности прогноза
```

Метрика $R2$ или коэффициент детерминации -

```
выбрана для оценки степени соответствия и меры того, насколько хорошо  выборки могут быть предсказаны
моделью через долю объясненной дисперсии.
```

Все метрики выбраны для оценки задачи регрессии.

```
# Вывод метрик
print('{:20} {:>20} {:>20}'.format('','Дерево решений', 'Случайный лес'))
print('{:20} {:20} {:21}'.format('mean_squared_error', DTR_mse, RF_mse))
print('{:20} {:20} {:20}'.format('r2_score', DTR_r2, RF_r2))
```

```
                          Дерево решений        Случайный лес
mean_squared_error      1.560747663551402   0.45064275700934486
r2_score               0.9687356201174011   0.9909728736582422
```

Выводы

По результатам оценок обеих метрик можно сделать вывод о том, что модель 'Случайный лес' обучена лучше.