Рубежный контроль №1

Киселев Владислав, ИУ5-62Б,

Вариант 12, Задание 2.

Задание №2.

Для заданного набора данных проведите обработку пропусков в данных для одного категориального и одного количественного признака. Какие способы обработки пропусков в данных для категориальных и количественных признаков Вы использовали? Какие признаки Вы будете использовать для дальнейшего построения моделей машинного обучения и почему?

Технологии разведочного анализа и обработки данных.

Этот набор данных предназначен для объединения нескольких аспектов данных об образовании США.

In [1]:

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.impute import SimpleImputer
from sklearn.impute import MissingIndicator
%matplotlib inline
sns.set(style="ticks")
```

In [2]:

```
data = pd.read_csv('states_all.csv', sep=",")
```

In [3]:

```
# размер набора данных
data.shape
```

Out[3]:

```
(1715, 25)
```

In [4]:

```
# типы колонок
data.dtypes
```

Out[4]:

```
PRIMARY_KEY                    object
STATE                          object
YEAR                            int64
ENROLL                        float64
TOTAL_REVENUE                 float64
FEDERAL_REVENUE               float64
STATE_REVENUE                 float64
LOCAL_REVENUE                 float64
TOTAL_EXPENDITURE             float64
INSTRUCTION_EXPENDITURE       float64
SUPPORT_SERVICES_EXPENDITURE  float64
OTHER_EXPENDITURE             float64
CAPITAL_OUTLAY_EXPENDITURE    float64
GRADES_PK_G                   float64
```

```
GRADES_KG_G                     float64
GRADES_4_G                      float64
GRADES_8_G                      float64
GRADES_12_G                     float64
GRADES_1_8_G                    float64
GRADES_9_12_G                   float64
GRADES_ALL_G                    float64
AVG_MATH_4_SCORE                float64
AVG_MATH_8_SCORE                float64
AVG_READING_4_SCORE             float64
AVG_READING_8_SCORE             float64
dtype: object
```

In [5]:

```
# проверим есть ли пропущенные значения
data.isnull().sum()
```

Out[5]:

```
PRIMARY_KEY                       0
STATE                             0
YEAR                              0
ENROLL                          491
TOTAL_REVENUE                   440
FEDERAL_REVENUE                 440
STATE_REVENUE                   440
LOCAL_REVENUE                   440
TOTAL_EXPENDITURE               440
INSTRUCTION_EXPENDITURE         440
SUPPORT_SERVICES_EXPENDITURE    440
OTHER_EXPENDITURE               491
CAPITAL_OUTLAY_EXPENDITURE      440
GRADES_PK_G                     173
GRADES_KG_G                      83
GRADES_4_G                       83
GRADES_8_G                       83
GRADES_12_G                      83
GRADES_1_8_G                    695
GRADES_9_12_G                   644
GRADES_ALL_G                     83
AVG_MATH_4_SCORE                1150
AVG_MATH_8_SCORE                1113
AVG_READING_4_SCORE             1065
AVG_READING_8_SCORE             1153
dtype: int64
```

In [6]:

```
# Первые 10 строк датасета
data.head(10)
```

Out[6]:

	PRIMARY_KEY	STATE	YEAR	ENROLL	TOTAL_REVENUE	FEDERAL_REVENUE	STATE_
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	1
5	1992_COLORADO	COLORADO	1992	NaN	3185173.0	163253.0	
6	1992_CONNECTICUT	CONNECTICUT	1992	NaN	3834302.0	143542.0	
7	1992_DELAWARE	DELAWARE	1992	NaN	645233.0	45945.0	
8	1992_DISTRICT_OF_COLUMBIA	DISTRICT_OF_COLUMBIA	1992	NaN	709480.0	64749.0	
9	1992_FLORIDA	FLORIDA	1992	NaN	11506299.0	788420.0	

10 rows × 25 columns

In [7]:

```
# Основные статистические характеристки набора данных
data.describe()
```

Out[7]:

	YEAR	ENROLL	TOTAL_REVENUE	FEDERAL_REVENUE	STATE_REVENUE	LOCAL_REVENUE	TOTAL_EXPE
count	1715.000000	1.224000e+03	1.275000e+03	1.275000e+03	1.275000e+03	1.275000e+03	1.27
mean	2002.075219	9.175416e+05	9.102045e+06	7.677799e+05	4.223743e+06	4.110522e+06	9.20
std	9.568621	1.066514e+06	1.175962e+07	1.146992e+06	5.549735e+06	5.489562e+06	1.19
min	1986.000000	4.386600e+04	4.656500e+05	3.102000e+04	0.000000e+00	2.209300e+04	4.81
25%	1994.000000	2.645145e+05	2.189504e+06	1.899575e+05	1.165776e+06	7.151210e+05	2.17
50%	2002.000000	6.499335e+05	5.085826e+06	4.035480e+05	2.537754e+06	2.058996e+06	5.24
75%	2010.000000	1.010532e+06	1.084516e+07	8.279320e+05	5.055548e+06	4.755293e+06	1.07
max	2019.000000	6.307022e+06	8.921726e+07	9.990221e+06	5.090457e+07	3.610526e+07	8.53

8 rows × 23 columns

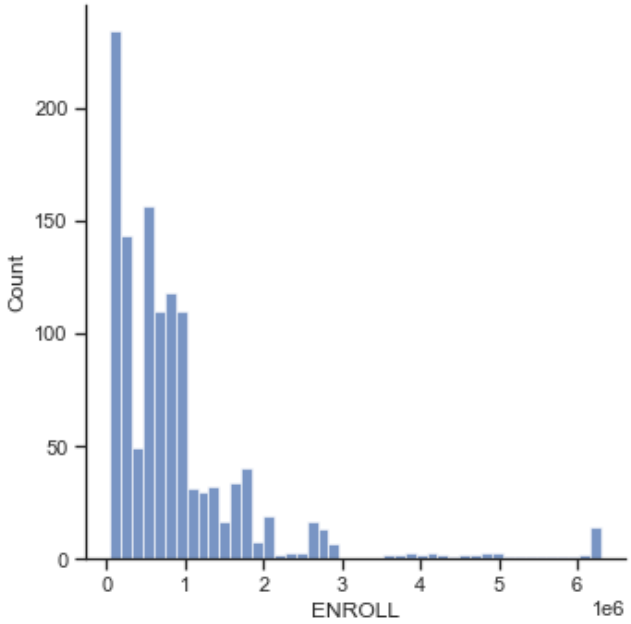
Построим гистограмму для столбца **ENROLL**

In [8]:

```
sns.displot(x=data['ENROLL'])
```

Out[8]:

```
<seaborn.axisgrid.FacetGrid at 0x1cb76ba8af0>
```



Обработка пропусков в числовых данных

Простая стратегия - удаление

In [9]:

```
# Удаление строк, имеющих пустые значения в колонке TOTAL_REVENUE
```

```
data1 = data.dropna(axis=0, subset=['TOTAL_REVENUE'])
(data.shape, data1.shape)
```

Out[9]:

```
((1715, 25), (1275, 25))
```

In [10]:

```
# проверим есть ли пропущенные значения
data1.isnull().sum()
```

Out[10]:

```
PRIMARY_KEY                         0
STATE                               0
YEAR                                0
ENROLL                             51
TOTAL_REVENUE                       0
FEDERAL_REVENUE                     0
STATE_REVENUE                       0
LOCAL_REVENUE                       0
TOTAL_EXPENDITURE                   0
INSTRUCTION_EXPENDITURE             0
SUPPORT_SERVICES_EXPENDITURE        0
OTHER_EXPENDITURE                  51
CAPITAL_OUTLAY_EXPENDITURE          0
GRADES_PK_G                        13
GRADES_KG_G                         0
GRADES_4_G                          0
GRADES_8_G                          0
GRADES_12_G                         0
GRADES_1_8_G                      255
GRADES_9_12_G                     255
GRADES_ALL_G                        0
AVG_MATH_4_SCORE                  835
AVG_MATH_8_SCORE                  836
AVG_READING_4_SCORE               752
AVG_READING_8_SCORE               839
dtype: int64
```

"Внедрение значений" - импьютация **(imputation)**

In [11]:

```
total_count = data.shape[0]
print('Всего строк: {}'.format(total_count))
```

```
Всего строк: 1715
```

In [12]:

```
# Выберем числовые колонки с пропущенными значениями
# Цикл по колонкам датасета
num_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 and (dt=='float64' or dt=='int64'):
        num_cols.append(col)
        temp_perc = round((temp_null_count / total_count) * 100.0, 2)
        print('Колонка {}. Тип данных {}. Количество пустых значений {}, {}%.'.format(co
l, dt, temp_null_count, temp_perc))
```

```
Колонка ENROLL. Тип данных float64. Количество пустых значений 491, 28.63%.
Колонка TOTAL_REVENUE. Тип данных float64. Количество пустых значений 440, 25.66%.
Колонка FEDERAL_REVENUE. Тип данных float64. Количество пустых значений 440, 25.66%.
Колонка STATE_REVENUE. Тип данных float64. Количество пустых значений 440, 25.66%.
Колонка LOCAL_REVENUE. Тип данных float64. Количество пустых значений 440, 25.66%.
Колонка TOTAL_EXPENDITURE. Тип данных float64. Количество пустых значений 440, 25.66%.
Колонка INSTRUCTION_EXPENDITURE. Тип данных float64. Количество пустых значений 440, 25.6
```

Колонка INSTRUCTION_EXPENDITURE. Тип данных float64. Количество пустых значений 440, 25.6
6%.
Колонка SUPPORT_SERVICES_EXPENDITURE. Тип данных float64. Количество пустых значений 440,
25.66%.
Колонка OTHER_EXPENDITURE. Тип данных float64. Количество пустых значений 491, 28.63%.
Колонка CAPITAL_OUTLAY_EXPENDITURE. Тип данных float64. Количество пустых значений 440, 2
5.66%.
Колонка GRADES_PK_G. Тип данных float64. Количество пустых значений 173, 10.09%.
Колонка GRADES_KG_G. Тип данных float64. Количество пустых значений 83, 4.84%.
Колонка GRADES_4_G. Тип данных float64. Количество пустых значений 83, 4.84%.
Колонка GRADES_8_G. Тип данных float64. Количество пустых значений 83, 4.84%.
Колонка GRADES_12_G. Тип данных float64. Количество пустых значений 83, 4.84%.
Колонка GRADES_1_8_G. Тип данных float64. Количество пустых значений 695, 40.52%.
Колонка GRADES_9_12_G. Тип данных float64. Количество пустых значений 644, 37.55%.
Колонка GRADES_ALL_G. Тип данных float64. Количество пустых значений 83, 4.84%.
Колонка AVG_MATH_4_SCORE. Тип данных float64. Количество пустых значений 1150, 67.06%.
Колонка AVG_MATH_8_SCORE. Тип данных float64. Количество пустых значений 1113, 64.9%.
Колонка AVG_READING_4_SCORE. Тип данных float64. Количество пустых значений 1065, 62.1%.
Колонка AVG_READING_8_SCORE. Тип данных float64. Количество пустых значений 1153, 67.23%.

In [13]:

```
# Фильтр по колонкам с пропущенными значениями
data_num = data[num_cols]
data_num
```

Out[13]:

	ENROLL	TOTAL_REVENUE	FEDERAL_REVENUE	STATE_REVENUE	LOCAL_REVENUE	TOTAL_EXPENDITURE	INSTRUC
0	NaN	2678885.0	304177.0	1659028.0	715680.0	2653798.0	
1	NaN	1049591.0	106780.0	720711.0	222100.0	972488.0	
2	NaN	3258079.0	297888.0	1369815.0	1590376.0	3401580.0	
3	NaN	1711959.0	178571.0	958785.0	574603.0	1743022.0	
4	NaN	26260025.0	2072470.0	16546514.0	7641041.0	27138832.0	
...	...	...	...	...	...	...	
1710	NaN	NaN	NaN	NaN	NaN	NaN	
1711	NaN	NaN	NaN	NaN	NaN	NaN	
1712	NaN	NaN	NaN	NaN	NaN	NaN	
1713	NaN	NaN	NaN	NaN	NaN	NaN	
1714	NaN	NaN	NaN	NaN	NaN	NaN	

1715 rows × 22 columns

In [14]:

```
data_num_GRADES_4_G = data_num[['GRADES_4_G']]
data_num_GRADES_4_G.head()
```

Out[14]:

	GRADES_4_G
0	57948.0
1	9748.0
2	55433.0
3	34632.0
4	418418.0

In [15]:

```
# Фильтр для проверки заполнения пустых значений
```

```
indicator = MissingIndicator()
mask_missing_values_only = indicator.fit_transform(data_num_GRADES_4_G)
mask_missing_values_only
```

Out[15]:

```
array([[False],
       [False],
       [False],
       ...,
       [ True],
       [ True],
       [ True]])
```

Попробуем заполнить пропущенные значения в колонке **GRADES_4_G** значениями, вычисленными по среднему арифметическому, медиане и моде.

In [16]:

```
strategies=['mean', 'median', 'most_frequent']
```

In [17]:

```
def test_num_impute_col(dataset, column, strategy_param):
    temp_data = dataset[[column]]

    indicator = MissingIndicator()
    mask_missing_values_only = indicator.fit_transform(temp_data)

    imp_num = SimpleImputer(strategy=strategy_param)
    data_num_imp = imp_num.fit_transform(temp_data)

    filled_data = data_num_imp[mask_missing_values_only]

    return column, strategy_param, filled_data.size, filled_data[0], filled_data[filled_
data.size-1]
```

In [18]:

```
data[['GRADES_4_G']].describe()
```

Out[18]:

	GRADES_4_G
count	1632.000000
mean	69166.507966
std	79957.990356
min	4577.000000
25%	18971.500000
50%	49087.000000
75%	78504.250000
max	493415.000000

In [19]:

```
test_num_impute_col(data, 'GRADES_4_G', strategies[0])
```

Out[19]:

```
('GRADES_4_G', 'mean', 83, 69166.50796568628, 69166.50796568628)
```

In [20]:

```
test_num_impute_col(data, 'GRADES_4_G', strategies[1])
```

Out[20]:

```
('GRADES_4_G', 'median', 83, 49087.0, 49087.0)
```

In [21]:

```
test_num_impute_col(data, 'GRADES_4_G', strategies[2])
```

Out[21]:

```
('GRADES_4_G', 'most_frequent', 83, 36435.0, 36435.0)
```

In [22]:

```
data['GRADES_4_G'] = data['GRADES_4_G'].fillna(round(data[['GRADES_4_G']].median()[0], 3
))
```

In [23]:

```
data.isnull().sum()
```

Out[23]:

```
PRIMARY_KEY                       0
STATE                             0
YEAR                              0
ENROLL                          491
TOTAL_REVENUE                   440
FEDERAL_REVENUE                 440
STATE_REVENUE                   440
LOCAL_REVENUE                   440
TOTAL_EXPENDITURE               440
INSTRUCTION_EXPENDITURE         440
SUPPORT_SERVICES_EXPENDITURE    440
OTHER_EXPENDITURE               491
CAPITAL_OUTLAY_EXPENDITURE      440
GRADES_PK_G                     173
GRADES_KG_G                      83
GRADES_4_G                        0
GRADES_8_G                       83
GRADES_12_G                      83
GRADES_1_8_G                    695
GRADES_9_12_G                   644
GRADES_ALL_G                     83
AVG_MATH_4_SCORE                1150
AVG_MATH_8_SCORE                1113
AVG_READING_4_SCORE             1065
AVG_READING_8_SCORE             1153
dtype: int64
```

Обработка пропусков в категориальных данных

In [24]:

```
# проверим есть ли пропущенные значения
data.isnull().sum()
```

Out[24]:

```
PRIMARY_KEY                       0
STATE                             0
YEAR                              0
ENROLL                          491
TOTAL_REVENUE                   440
FEDERAL_REVENUE                 440
STATE_REVENUE                   440
LOCAL_REVENUE                   440
TOTAL_EXPENDITURE               440
INSTRUCTION_EXPENDITURE         440
SUPPORT_SERVICES_EXPENDITURE    440
OTHER_EXPENDITURE               491
CAPITAL_OUTLAY_EXPENDITURE      440
GRADES_PK_G                     173
```

```
GRADES_KG_G                      83
GRADES_4_G                        0
GRADES_8_G                       83
GRADES_12_G                      83
GRADES_1_8_G                    695
GRADES_9_12_G                   644
GRADES_ALL_G                     83
AVG_MATH_4_SCORE               1150
AVG_MATH_8_SCORE               1113
AVG_READING_4_SCORE            1065
AVG_READING_8_SCORE            1153
dtype: int64
```

In [25]:

```
# типы колонок
data.dtypes
```

Out[25]:

```
PRIMARY_KEY                     object
STATE                           object
YEAR                             int64
ENROLL                         float64
TOTAL_REVENUE                  float64
FEDERAL_REVENUE                float64
STATE_REVENUE                  float64
LOCAL_REVENUE                  float64
TOTAL_EXPENDITURE              float64
INSTRUCTION_EXPENDITURE        float64
SUPPORT_SERVICES_EXPENDITURE   float64
OTHER_EXPENDITURE              float64
CAPITAL_OUTLAY_EXPENDITURE     float64
GRADES_PK_G                    float64
GRADES_KG_G                    float64
GRADES_4_G                     float64
GRADES_8_G                     float64
GRADES_12_G                    float64
GRADES_1_8_G                   float64
GRADES_9_12_G                  float64
GRADES_ALL_G                   float64
AVG_MATH_4_SCORE               float64
AVG_MATH_8_SCORE               float64
AVG_READING_4_SCORE            float64
AVG_READING_8_SCORE            float64
dtype: object
```

Категориальные признаки в этом **dataset** не имеют пропусков**.**

Выводы

In [26]:

```
# Выберем числовые колонки с пропущенными значениями
# Цикл по колонкам датасета
num_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 and (dt=='float64' or dt=='int64'):
        num_cols.append(col)
        temp_perc = round((temp_null_count / total_count) * 100.0, 2)
        print('Колонка {}. Тип данных {}. Количество пустых значений {}, {}%.'.format(co
l, dt, temp_null_count, temp_perc))
```

```
Колонка ENROLL. Тип данных float64. Количество пустых значений 491, 28.63%.
Колонка TOTAL_REVENUE. Тип данных float64. Количество пустых значений 440, 25.66%.
Колонка FEDERAL_REVENUE. Тип данных float64. Количество пустых значений 440, 25.66%.
Колонка STATE_REVENUE. Тип данных float64. Количество пустых значений 440, 25.66%.
```

```
Колонка LOCAL_REVENUE. Тип данных float64. Количество пустых значений 440, 25.66%.
Колонка TOTAL_EXPENDITURE. Тип данных float64. Количество пустых значений 440, 25.66%.
Колонка INSTRUCTION_EXPENDITURE. Тип данных float64. Количество пустых значений 440, 25.6
6%.
Колонка SUPPORT_SERVICES_EXPENDITURE. Тип данных float64. Количество пустых значений 440,
25.66%.
Колонка OTHER_EXPENDITURE. Тип данных float64. Количество пустых значений 491, 28.63%.
Колонка CAPITAL_OUTLAY_EXPENDITURE. Тип данных float64. Количество пустых значений 440, 2
5.66%.
Колонка GRADES_PK_G. Тип данных float64. Количество пустых значений 173, 10.09%.
Колонка GRADES_KG_G. Тип данных float64. Количество пустых значений 83, 4.84%.
Колонка GRADES_8_G. Тип данных float64. Количество пустых значений 83, 4.84%.
Колонка GRADES_12_G. Тип данных float64. Количество пустых значений 83, 4.84%.
Колонка GRADES_1_8_G. Тип данных float64. Количество пустых значений 695, 40.52%.
Колонка GRADES_9_12_G. Тип данных float64. Количество пустых значений 644, 37.55%.
Колонка GRADES_ALL_G. Тип данных float64. Количество пустых значений 83, 4.84%.
Колонка AVG_MATH_4_SCORE. Тип данных float64. Количество пустых значений 1150, 67.06%.
Колонка AVG_MATH_8_SCORE. Тип данных float64. Количество пустых значений 1113, 64.9%.
Колонка AVG_READING_4_SCORE. Тип данных float64. Количество пустых значений 1065, 62.1%.
Колонка AVG_READING_8_SCORE. Тип данных float64. Количество пустых значений 1153, 67.23%.
```

Исходя из количества пропусков в определенный колонках можно сделать следующие выводы:

- Колонки, содержащие менее **5%,** пропусков выбираем для построения модели.
- Колонки, содержащие менее **30%** пропусков также выбираем для построения модели.
- Колонки **GRADES_1_8_G (40.52%), GRADES_9_12_G (37.55%)** не выбираем для построения модели, в случае отсутствия необходимости в этих колонках.
- Колонки **AVG_MATH_4_SCORE (67.06%), AVG_MATH_8_SCORE (64.9%), AVG_READING_4_SCORE (62.1%)** не выбираем для построения модели в любом случае.

В данной работе для обрабоки пропусков данных мы использовали внедрение значений или импутацию данных в количественном признаке **GRADES_4_G,** использовав медиану **(**мода и среднее значение также были проверены как возможные**).** Также была проведена простая стратегия удаления строк, содержащий пустые значения, из-за одинаковых значений **(440)** в большинстве признаках, поэтому были удалены все строки, в которых признак **TOTAL_REVENUE** имел пропуски, таким образом получилась хорошая модель из **1275** строк из **1715** начальных с минимальным количеством пропусков **(**данные ниже**).**

In [27]:

```
# Выберем числовые колонки с пропущенными значениями
# Цикл по колонкам датасета
num_cols = []
for col in data1.columns:
    # Количество пустых значений
    temp_null_count = data1[data1[col].isnull()].shape[0]
    dt = str(data1[col].dtype)
    if temp_null_count>0 and (dt=='float64' or dt=='int64'):
        num_cols.append(col)
        temp_perc = round((temp_null_count / total_count) * 100.0, 2)
        print('Колонка {}. Тип данных {}. Количество пустых значений {}, {}%.'.format(co
l, dt, temp_null_count, temp_perc))
```

```
Колонка ENROLL. Тип данных float64. Количество пустых значений 51, 2.97%.
Колонка OTHER_EXPENDITURE. Тип данных float64. Количество пустых значений 51, 2.97%.
Колонка GRADES_PK_G. Тип данных float64. Количество пустых значений 13, 0.76%.
Колонка GRADES_1_8_G. Тип данных float64. Количество пустых значений 255, 14.87%.
Колонка GRADES_9_12_G. Тип данных float64. Количество пустых значений 255, 14.87%.
Колонка AVG_MATH_4_SCORE. Тип данных float64. Количество пустых значений 835, 48.69%.
Колонка AVG_MATH_8_SCORE. Тип данных float64. Количество пустых значений 836, 48.75%.
Колонка AVG_READING_4_SCORE. Тип данных float64. Количество пустых значений 752, 43.85%.
Колонка AVG_READING_8_SCORE. Тип данных float64. Количество пустых значений 839, 48.92%.
```

В этой модели также колонки **AVG_MATH_4_SCORE (48.69%), AVG_MATH_8_SCORE (48.75%), AVG_READING_4_SCORE (43.85%)** с соответсвующим количеством пропусков первыми подходят к удаления, для построения соответсвующей модели машинного обучения.