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

Отчёт по рубежному контролю №1

По дисциплине:
«Технологии машинного обучения»

Выполнил:

Студент группы ИУ5-61Б _____ **Шевчук М.С**

(Подпись, дата) (Фамилия И.О.)

Проверил:

_____ **Гапанюк Ю. Е.**

(Подпись, дата) (Фамилия И.О.)

Москва, 2021

Задание

Для заданного набора данных произведите масштабирование данных (для одного признака) и преобразование категориальных признаков в количественные двумя способами (label encoding, one hot encoding) для одного признака. Какие методы Вы
использовали для решения задачи и почему?

Для студентов групп ИУ5-61Б, ИУ5Ц-81Б - для пары произвольных колонок данных построить график "Диаграмма рассеяния".
Набор данных:

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

РК ИУ5-61Б

Импорт библиотек

In [1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from pandas.plotting import scatter_matrix
from sklearn.impute import SimpleImputer
from sklearn.impute import MissingIndicator
import warnings
warnings.filterwarnings('ignore')
sns.set(style="ticks")
%matplotlib inline
```

In [2]:
```
data = pd.read_csv('states_all.csv')
```

In [3]:
```
data.head()
```

Out[3]:

	PRIMARY_KEY	STATE	YEAR	ENROLL	TOTAL_REVENUE	FEDERAL_REVENUE	S
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	

5 rows × 25 columns

In [4]:
```
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]:
```
data.isnull().sum()
# проверим есть ли пропущенные значения
```

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
```

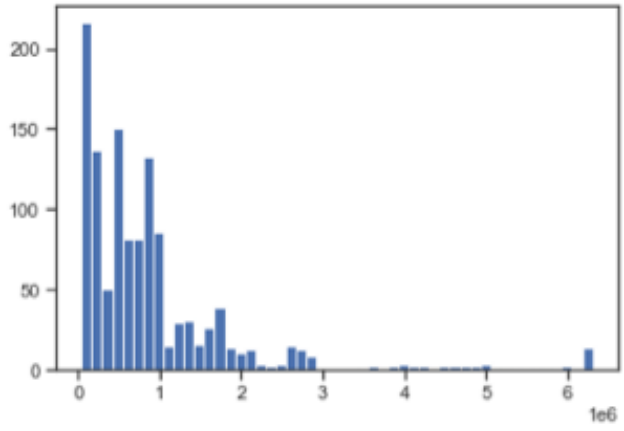
In [6]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1715 entries, 0 to 1714
Data columns (total 25 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   PRIMARY_KEY                     1715 non-null   object
 1   STATE                           1715 non-null   object
 2   YEAR                            1715 non-null   int64
 3   ENROLL                          1224 non-null   float64
 4   TOTAL_REVENUE                   1275 non-null   float64
 5   FEDERAL_REVENUE                 1275 non-null   float64
 6   STATE_REVENUE                   1275 non-null   float64
 7   LOCAL_REVENUE                   1275 non-null   float64
 8   TOTAL_EXPENDITURE               1275 non-null   float64
 9   INSTRUCTION_EXPENDITURE         1275 non-null   float64
 10  SUPPORT_SERVICES_EXPENDITURE    1275 non-null   float64
 11  OTHER_EXPENDITURE               1224 non-null   float64
 12  CAPITAL_OUTLAY_EXPENDITURE      1275 non-null   float64
 13  GRADES_PK_G                     1542 non-null   float64
 14  GRADES_KG_G                     1632 non-null   float64
 15  GRADES_4_G                      1632 non-null   float64
 16  GRADES_8_G                      1632 non-null   float64
 17  GRADES_12_G                     1632 non-null   float64
 18  GRADES_1_8_G                    1020 non-null   float64
 19  GRADES_9_12_G                   1071 non-null   float64
 20  GRADES_ALL_G                    1632 non-null   float64
 21  AVG_MATH_4_SCORE                565 non-null    float64
 22  AVG_MATH_8_SCORE                602 non-null    float64
 23  AVG_READING_4_SCORE             650 non-null    float64
 24  AVG_READING_8_SCORE             562 non-null    float64
dtypes: float64(22), int64(1), object(2)
memory usage: 335.1+ KB
```

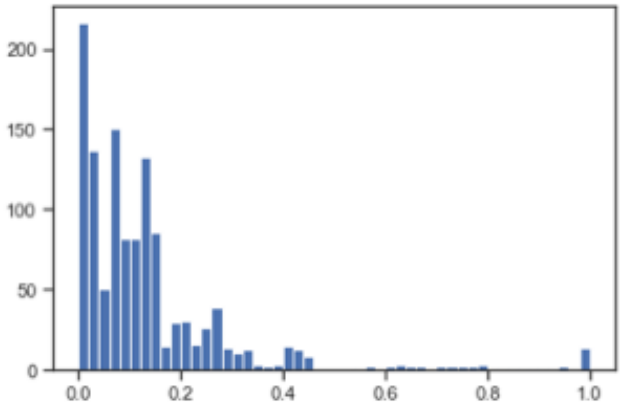
In [7]:
```
from sklearn.preprocessing import MinMaxScaler, StandardScaler, Normalizer
```

In [8]:
```
sc1 = MinMaxScaler()
sc1_data = sc1.fit_transform(data[['ENROLL']])
```

In [9]:
```
plt.hist(data['ENROLL'], 50)
plt.show()
```



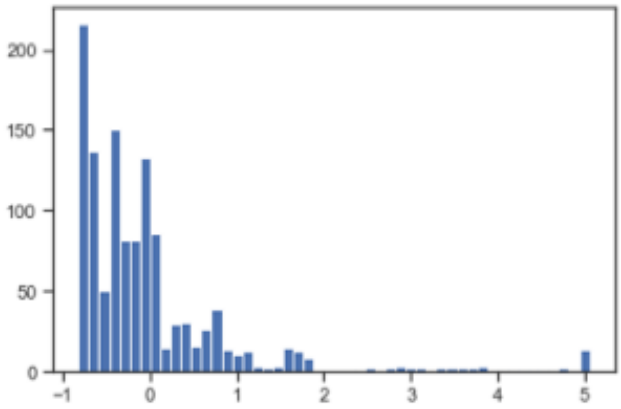
In [10]:
```
plt.hist(sc1_data, 50)
plt.show()
```



Масштабирование данных на основе Z-оценки - StandardScaler¶

In [11]:
```
sc2 = StandardScaler()
sc2_data = sc2.fit_transform(data[['ENROLL']])
```

In [12]:
```
plt.hist(sc2_data, 50)
plt.show()
```



In [13]:
```
cat_temp_data = data[['STATE']]
cat_temp_data.head()
```

Out[13]:

	STATE
0	ALABAMA
1	ALASKA
2	ARIZONA
3	ARKANSAS
4	CALIFORNIA

In [14]:
```
# Импьютация наиболее частыми значениями
imp2 = SimpleImputer(missing_values=np.nan, strategy='most_frequent')
data_imp2 = imp2.fit_transform(cat_temp_data)
data_imp2
```

Out[14]:
```
array([['ALABAMA'],
       ['ALASKA'],
       ['ARIZONA'],
       ...,
       ['WEST_VIRGINIA'],
       ['WISCONSIN'],
       ['WYOMING']], dtype=object)
```

In [15]:
```
cat_enc = pd.DataFrame({'STATE':data_imp2.T[0]})
cat_enc
```

Out[15]:

	STATE
0	ALABAMA
1	ALASKA
2	ARIZONA
3	ARKANSAS
4	CALIFORNIA
...	...
1710	VIRGINIA
1711	WASHINGTON
1712	WEST_VIRGINIA
1713	WISCONSIN
1714	WYOMING

1715 rows × 1 columns

In [16]:
```
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
```

Кодирование категорий целочисленными значениями - label encoding

In [17]:
```
le = LabelEncoder()
cat_enc_le = le.fit_transform(cat_enc['STATE'])
```

In [18]:
```
cat_enc['STATE'].unique()
```

Out[18]:
```
array(['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', 'DODEA', 'NATIONAL'],
      dtype=object)
```

In [19]:
```
np.unique(cat_enc_le)
```

Out[19]:
```
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
       17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33,
       34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50,
       51, 52])
```

In [20]:
```
le.inverse_transform([0, 1, 2, 3])
```

Out[20]:
```
array(['ALABAMA', 'ALASKA', 'ARIZONA', 'ARKANSAS'], dtype=object)
```

Кодирование категорий наборами бинарных значений - one-hot encoding

In [21]:
```
ohe = OneHotEncoder()
cat_enc_ohe = ohe.fit_transform(cat_enc[['STATE']])
```

In [22]:
```
cat_enc.shape
```

Out[22]:
```
(1715, 1)
```

In [23]:
```
cat_enc_ohe.shape
```

Out[23]:
```
(1715, 53)
```

In [24]:
```
cat_enc_ohe
```

Out[24]:
```
<1715x53 sparse matrix of type '<class 'numpy.float64'>'
        with 1715 stored elements in Compressed Sparse Row format>
```

In [25]:
```
cat_enc_ohe.todense()[0:10]
```

Out[25]:
```
matrix([[1., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
         0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
         0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
         0., 0., 0., 0., 0.],
        [0., 1., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
         0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
         0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
         0., 0., 0., 0., 0.],
        [0., 0., 1., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
         0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
         0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
         0., 0., 0., 0., 0.],
        [0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
         0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
         0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
         0., 0., 0., 0., 0.],
        [0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
         0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
         0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
         0., 0., 0., 0., 0.],
        [0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
         0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
         0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
         0., 0., 0., 0., 0.],
        [0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
         0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
         0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
         0., 0., 0., 0., 0.],
        [0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 0., 0.,
         0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
         0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
         0., 0., 0., 0., 0.],
        [0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 0.,
         0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
         0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
         0., 0., 0., 0., 0.],
        [0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0.,
         0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
         0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
         0., 0., 0., 0., 0.]])
```

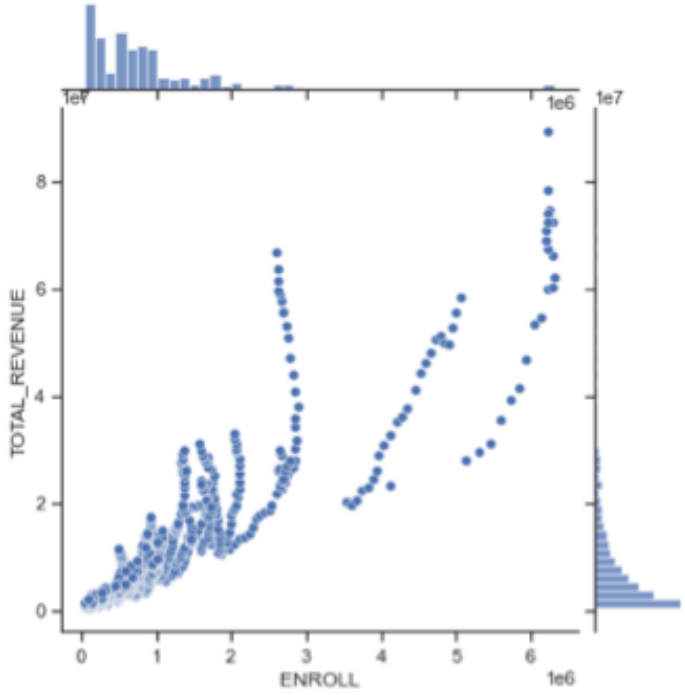
In [26]:
```
cat_enc.head(10)
```

Out[26]:

	STATE
0	ALABAMA
1	ALASKA
2	ARIZONA
3	ARKANSAS
4	CALIFORNIA
5	COLORADO
6	CONNECTICUT
7	DELAWARE
8	DISTRICT_OF_COLUMBIA
9	FLORIDA

In [27]:
```
# Увеличенные диаграммы рассеяния
sns.jointplot(x = "ENROLL", y = "TOTAL_REVENUE", kind="scatter", data = da
```

Out[27]:
```
<seaborn.axisgrid.JointGrid at 0x7f8ba7251b20>
```



In [28]:
```
fig, ax = plt.subplots(figsize=(10,10))
sns.scatterplot(ax=ax, x='ENROLL', y='TOTAL_REVENUE', data=data, hue='OTHE
```

Out[28]: <AxesSubplot:xlabel='ENROLL', ylabel='TOTAL_REVENUE'>