Саргсян Тигран ИУ5-62Б

Рубежный контроль №1

20 Вариант

Задача №3

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?

Дополнение для ИУ5-62Б.

Для произвольной колонки данных построить гистограмму.

In [7]:

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

In [8]:

```
data=pd.read_csv("states_all.csv")
```

In [9]:

```
data.head()
```

Out[9]:

	PRIMARY_KEY	STATE	YEAR	ENROLL	TOTAL_REVENUE	FEDERAL_REVENUE	STATE_REVENUE	LOCAL_REVENUE	TOTAL_EXPENDITURI
0	1992_ALABAMA	ALABAMA	1992	NaN	2678885.0	304177.0	1659028.0	715680.0	2653798.(
1	1992_ALASKA	ALASKA	1992	NaN	1049591.0	106780.0	720711.0	222100.0	972488.(
2	1992_ARIZONA	ARIZONA	1992	NaN	3258079.0	297888.0	1369815.0	1590376.0	3401580.(
3	1992_ARKANSAS	ARKANSAS	1992	NaN	1711959.0	178571.0	958785.0	574603.0	1743022.(
4	1992_CALIFORNIA	CALIFORNIA	1992	NaN	26260025.0	2072470.0	16546514.0	7641041.0	27138832.(

5 rows × 25 columns

In [10]:

```
# типы колонок
data.dtypes
```

Out[10]:

```
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 [11]:

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

Out[11]:

```
(1715, 25)
```

Масштабирование

In [12]:

```
from sklearn.preprocessing import MinMaxScaler, StandardScaler
```

In [13]:

```
data.describe()
```

Out[13]:

	YEAR	ENROLL	TOTAL_REVENUE	FEDERAL_REVENUE	STATE_REVENUE	LOCAL_REVENUE	TOTAL_EXPENDITURE	INSTRUCTIO
count	1715.000000	1.224000e+03	1.275000e+03	1.275000e+03	1.275000e+03	1.275000e+03	1.275000e+03	
mean	2002.075219	9.175416e+05	9.102045e+06	7.677799e+05	4.223743e+06	4.110522e+06	9.206242e+06	
std	9.568621	1.066514e+06	1.175962e+07	1.146992e+06	5.549735e+06	5.489562e+06	1.199279e+07	
min	1986.000000	4.386600e+04	4.656500e+05	3.102000e+04	0.000000e+00	2.209300e+04	4.816650e+05	
25%	1994.000000	2.645145e+05	2.189504e+06	1.899575e+05	1.165776e+06	7.151210e+05	2.170404e+06	
50%	2002.000000	6.499335e+05	5.085826e+06	4.035480e+05	2.537754e+06	2.058996e+06	5.242672e+06	
75%	2010.000000	1.010532e+06	1.084516e+07	8.279320e+05	5.055548e+06	4.755293e+06	1.074420e+07	
max	2019.000000	6.307022e+06	8.921726e+07	9.990221e+06	5.090457e+07	3.610526e+07	8.532013e+07	

8 rows × 23 columns

Выберем для масштабирования параметр "LOCAL_REVENUE"

In [14]:

```
data['LOCAL_REVENUE']
```

Out[14]:

```
0        715680.0
1        222100.0
2       1590376.0
3        574603.0
4       7641041.0
          ...
1710          NaN
1711          NaN
1712          NaN
1713          NaN
1714          NaN
Name: LOCAL_REVENUE, Length: 1715, dtype: float64
```

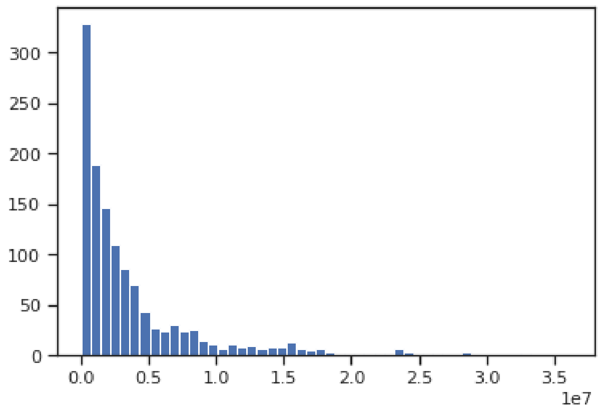
MinMax масштабирование

In [15]:

```
sc1 = MinMaxScaler()
sc1_data = sc1.fit_transform(data[['LOCAL_REVENUE']])
```

In [16]:

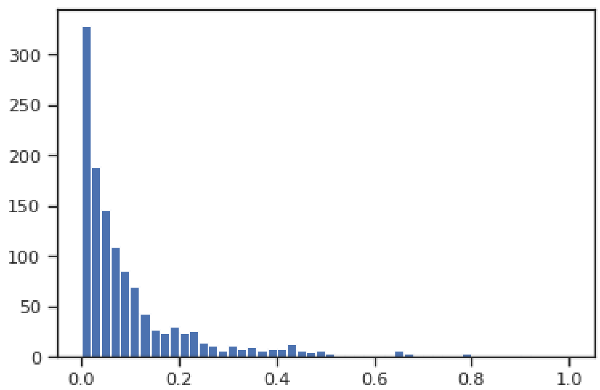
```
plt.hist(data['LOCAL_REVENUE'][:1275], 50)
plt.show()
```



Значения лежат в диапазоне от 0 до 2.5.

In [17]:

```
plt.hist(sc1_data[:1275], 50)
plt.show()
```



Масштабирование данных на основе Z-оценки (StandardScaler)

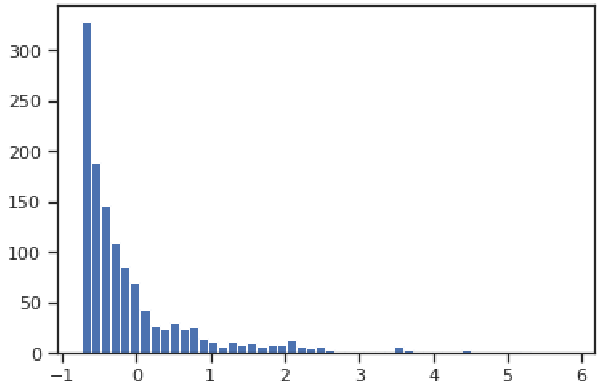
In [18]:

```
sc2 = StandardScaler()
sc2_data = sc2.fit_transform(data[['LOCAL_REVENUE']])
```

В этом случае большинство значений попадает в диапазон от -0,7 до 3.

In [19]:

```
plt.hist(sc2_data[:1275], 50)
plt.show()
```



Преобразование категориальных признаков в количественные

label encoding

In [20]:

```
types=data["STATE"]
types.value_counts()
```

Out[20]:

```
KANSAS                  33
ARIZONA                 33
OREGON                  33
COLORADO                33
DISTRICT_OF_COLUMBIA    33
ILLINOIS                33
NEBRASKA                33
WEST_VIRGINIA           33
VERMONT                 33
IOWA                    33
OKLAHOMA                33
WYOMING                 33
OHIO                    33
MASSACHUSETTS           33
WASHINGTON              33
VIRGINIA                33
MICHIGAN                33
HAWAII                  33
NEW_JERSEY              33
PENNSYLVANIA            33
NEVADA                  33
TENNESSEE               33
ALABAMA                 33
SOUTH_DAKOTA            33
NORTH_CAROLINA          33
MARYLAND                33
NORTH_DAKOTA            33
ARKANSAS                33
MINNESOTA               33
INDIANA                 33
CONNECTICUT             33
MONTANA                 33
SOUTH_CAROLINA          33
RHODE_ISLAND            33
NEW_YORK                33
DELAWARE                33
ALASKA                  33
UTAH                    33
CALIFORNIA              33
NEW_HAMPSHIRE           33
IDAHO                   33
NEW_MEXICO              33
LOUISIANA               33
MISSISSIPPI             33
WISCONSIN               33
KENTUCKY                33
MAINE                   33
GEORGIA                 33
MISSOURI                33
FLORIDA                 33
TEXAS                   33
DODEA                   16
NATIONAL                16
Name: STATE, dtype: int64
```

In [21]:

```
le=sklearn.preprocessing.LabelEncoder()
type_le=le.fit_transform(types)
print(np.unique(type_le))
le.inverse_transform(np.unique(type_le))
```

```
[ 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]
```

Out[21]:

```
array(['ALABAMA', 'ALASKA', 'ARIZONA', 'ARKANSAS', 'CALIFORNIA',
       'COLORADO', 'CONNECTICUT', 'DELAWARE', 'DISTRICT_OF_COLUMBIA',
       'DODEA', 'FLORIDA', 'GEORGIA', 'HAWAII', 'IDAHO', 'ILLINOIS',
       'INDIANA', 'IOWA', 'KANSAS', 'KENTUCKY', 'LOUISIANA', 'MAINE',
       'MARYLAND', 'MASSACHUSETTS', 'MICHIGAN', 'MINNESOTA',
       'MISSISSIPPI', 'MISSOURI', 'MONTANA', 'NATIONAL', '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'], dtype=object)
```

One hot encoding

In [22]:

```
type_s=pd.get_dummies(types)
type_s.head(25)
```

Out[22]:

	ALABAMA	ALASKA	ARIZONA	ARKANSAS	CALIFORNIA	COLORADO	CONNECTICUT	DELAWARE	DISTRICT_OF_COLUMBIA	DODEA	...	SOU
0	1	0	0	0	0	0	0	0	0	0	...	
1	0	1	0	0	0	0	0	0	0	0	...	
2	0	0	1	0	0	0	0	0	0	0	...	
3	0	0	0	1	0	0	0	0	0	0	...	
4	0	0	0	0	1	0	0	0	0	0	...	
5	0	0	0	0	0	1	0	0	0	0	...	
6	0	0	0	0	0	0	1	0	0	0	...	
7	0	0	0	0	0	0	0	1	0	0	...	
8	0	0	0	0	0	0	0	0	1	0	...	
9	0	0	0	0	0	0	0	0	0	0	...	
10	0	0	0	0	0	0	0	0	0	0	...	
11	0	0	0	0	0	0	0	0	0	0	...	
12	0	0	0	0	0	0	0	0	0	0	...	
13	0	0	0	0	0	0	0	0	0	0	...	
14	0	0	0	0	0	0	0	0	0	0	...	
15	0	0	0	0	0	0	0	0	0	0	...	
16	0	0	0	0	0	0	0	0	0	0	...	
17	0	0	0	0	0	0	0	0	0	0	...	
18	0	0	0	0	0	0	0	0	0	0	...	
19	0	0	0	0	0	0	0	0	0	0	...	
20	0	0	0	0	0	0	0	0	0	0	...	
21	0	0	0	0	0	0	0	0	0	0	...	
22	0	0	0	0	0	0	0	0	0	0	...	
23	0	0	0	0	0	0	0	0	0	0	...	
24	0	0	0	0	0	0	0	0	0	0	...	

25 rows × 53 columns

Гистограмма

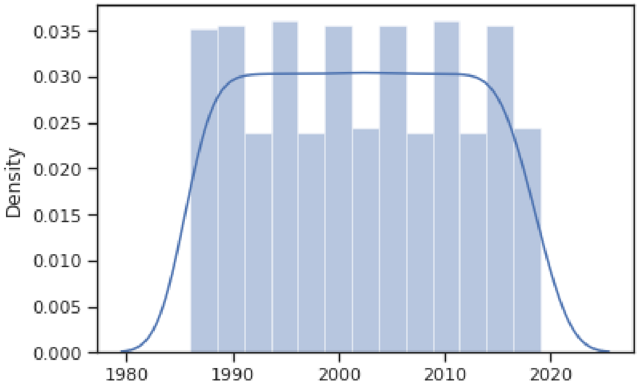
In [23]:

```
sns.distplot(x=data['YEAR'])
```

```
/opt/conda/lib/python3.8/site-packages/seaborn/distributions.py:2551: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for h istograms).
  warnings.warn(msg, FutureWarning)
```

Out[23]:

```
<AxesSubplot:ylabel='Density'>
```



In [ ]: