Работу выполнил Сорокин Андрей, студент группы ИУ5-64Б

Вариант 19, задание 3, датасет 3.

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 произвольной колонки данных построить график "Скрипичная диаграмма (violin plot)"

Ввод [1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
sns.set(style="ticks")
```

Ввод [2]:
```
# Будем анализировать данные только на обучающей выборке
data = pd.read_csv('marvel-wikia-data.csv', sep = ',')
```

Ввод [3]:
```
data.shape
```

Out[3]:
```
(16376, 13)
```

Ввод [4]:
```
total_count = data.shape[0]
print('Всего строк: {}'.format(total_count))
```

```
Всего строк: 16376
```

Ввод [5]:
```
# типы колонок
data.dtypes
```

Out[5]:
```
page_id              int64
name                object
urlslug             object
ID                  object
ALIGN               object
EYE                 object
HAIR                object
SEX                 object
GSM                 object
ALIVE               object
APPEARANCES        float64
FIRST APPEARANCE    object
Year               float64
dtype: object
```

Ввод [6]:
```
# Первые 5 строк датасета
data.head()
```

Out[6]:

ge_id	name	urlslug	ID	ALIGN	EYE	HAIR	SEX	GSM	ALIVE	APPEARANCES	FIRST APPEARANCE	Year
1678	Spider-Man (Peter Parker)	\Spider-Man_(Peter_Parker)	Secret Identity	Good Characters	Hazel Eyes	Brown Hair	Male Characters	NaN	Living Characters	4043.0	Aug-62	1962.0
7139	Captain America (Steven	\Captain_America_(Steven_Rogers)	Public Identity	Good Characters	Blue Eyes	White Hair	Male Characters	NaN	Living Characters	3360.0	Mar-41	1941.0

4786	Wolverine (James \"Logan\" Howlett)	\Wolverine_(James_%22Logan%22_Howlett)	Public Identity	Neutral Characters	Blue Eyes	Black Hair	Male Characters	NaN	Living Characters	3061.0	Oct-74	1974.0
1868	Iron Man (Anthony \"Tony\" Stark)	\Iron_Man_(Anthony_%22Tony%22_Stark)	Public Identity	Good Characters	Blue Eyes	Black Hair	Male Characters	NaN	Living Characters	2961.0	Mar-63	1963.0
2460	Thor (Thor Odinson)	\Thor_(Thor_Odinson)	No Dual Identity	Good Characters	Blue Eyes	Blond Hair	Male Characters	NaN	Living Characters	2258.0	Nov-50	1950.0

Ввод [7]:
```
# 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(col, dt, temp_null_count, temp_perc))
```

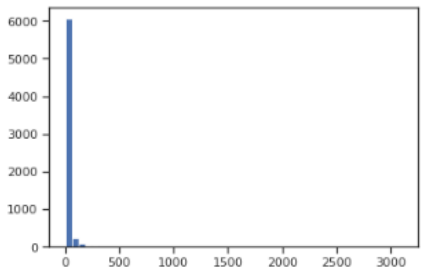
```
Колонка APPEARANCES. Тип данных float64. Количество пустых значений 1096, 6.69%.
Колонка Year. Тип данных float64. Количество пустых значений 815, 4.98%.
```

Выполним масштабирование колонки "Appearances" - кол-во появлений персонажей. Будем использовать метод MinMax

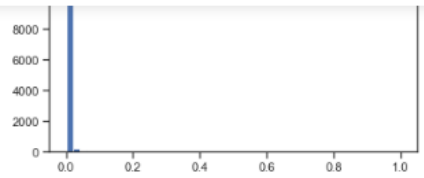
Ввод [8]:
```
from sklearn.preprocessing import MinMaxScaler, StandardScaler, Normalizer
```

Ввод [9]:
```
sc1 = MinMaxScaler()
sc1_data = sc1.fit_transform(data[['APPEARANCES']])
```

Ввод [12]:
```
plt.hist(data['APPEARANCES'], 50)
plt.show()
```



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



Выполним преобразование категориальных признаков в количественные

Ввод [11]:
```
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
```

Ввод [12]:
```
# Выберем категориальные колонки с пропущенными значениями
# Цикл по колонкам датасета
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 and (dt=='object'):
        cat_cols.append(col)
        temp_perc = round((temp_null_count / total_count) * 100.0, 2)
        print('Колонка {}. Тип данных {}. Количество пустых значений {}, {}%.'.format(col, dt, temp_null_count, temp_perc))
```

```
Колонка ID. Тип данных object. Количество пустых значений 3770, 23.02%.
Колонка ALIGN. Тип данных object. Количество пустых значений 2812, 17.17%.
Колонка EYE. Тип данных object. Количество пустых значений 9767, 59.64%.
Колонка HAIR. Тип данных object. Количество пустых значений 4264, 26.04%.
Колонка SEX. Тип данных object. Количество пустых значений 854, 5.21%.
Колонка GSM. Тип данных object. Количество пустых значений 16286, 99.45%.
Колонка ALIVE. Тип данных object. Количество пустых значений 3, 0.02%.
Колонка FIRST APPEARANCE. Тип данных object. Количество пустых значений 815, 4.98%.
```

Для преобразования возьмем колонку First APPEARANCE. Для этой колонки лучше всего использовать Label Encoding, так как это колонки отражают даты, то есть в них есть порядок.

Ввод [13]:
```
#Для начала заполним пропущенные данные
cat_temp_data = data[['FIRST APPEARANCE']]
cat_temp_data.head()
```

Out[13]:

	FIRST APPEARANCE
0	Aug-62
1	Mar-41
2	Oct-74
3	Mar-63
4	Nov-50

Ввод [14]:
```
cat_temp_data['FIRST APPEARANCE'].unique()
```

```
'Mar-72', 'Sep-67', 'Nov-80', 'Nov-63', 'May-71', 'Aug-89',
'Sep-73', 'Dec-83', 'Sep-00', 'Oct-66', 'Sep-03', 'Aug-60',
'Apr-49', 'Jul-85', 'Oct-71', 'Jun-87', 'Nov-04', 'Oct-69',
'Mar-82', 'Feb-64', 'Mar-51', 'Nov-94', 'Jun-75', 'Oct-83',
'Oct-84', 'Sep-62', 'Mar-79', 'Nov-84', 'Jan-75', 'Jan-08',
'May-88', 'Jan-64', 'Jul-73', 'Oct-06', 'Aug-03', 'Jul-68',
'Mar-84', 'Jun-64', 'Nov-66', 'Feb-72', 'Nov-01', 'Feb-69',
```

Ввод [15]:
```
from sklearn.impute import SimpleImputer
from sklearn.impute import MissingIndicator
```

Ввод [16]:
```
# Импьютация наиболее частыми значениями
imp2 = SimpleImputer(missing_values=np.nan, strategy='most_frequent')
data_imp2 = imp2.fit_transform(cat_temp_data)
data_imp2
```

Out[16]:
```
array([['Aug-62'],
       ['Mar-41'],
       ['Oct-74'],
       ...,
       ['Jun-92'],
       ['Jun-92'],
       ['Jun-92']], dtype=object)
```

Ввод [17]:
```
cat_enc = pd.DataFrame({'c1':data_imp2.T[0]})
cat_enc
```

Out[17]:

	c1
0	Aug-62
1	Mar-41
2	Oct-74
3	Mar-63
4	Nov-50
...	...
16371	Jun-92
16372	Jun-92
16373	Jun-92
16374	Jun-92
16375	Jun-92

16376 rows × 1 columns

Ввод [18]:
```
cat_enc['c1'].unique()
```

Out[18]:
```
array(['Aug-62', 'Mar-41', 'Oct-74', 'Mar-63', 'Nov-50', 'Nov-61',
       'May-62', 'Sep-63', 'Jun-92', 'May-75', 'Sep-64', 'Apr-64',
       'Jul-63', 'Jun-65', 'Jan-62', 'Mar-64', 'May-63', 'Jun-63',
       'Oct-68', 'Jan-80', 'Mar-68', 'Feb-80', 'Feb-74', 'Jun-72',
       'Oct-70', 'Aug-67', 'Oct-65', 'Jul-62', 'Dec-76', 'Mar-66',
       'Jul-64', 'Oct-64', 'Nov-82', 'Jul-90', 'May-74', 'Jan-86',
       'Nov-64', 'Mar-69', 'Nov-44', 'Oct-39', 'Aug-49', 'Oct-76',
       'Jan-79', 'Dec-65', 'Sep-69', 'Jul-67', 'Oct-62', 'Feb-91',
       'Dec-67', 'Nov-68', 'Feb-77', 'Aug-72', 'May-89', 'Aug-65',
       'Aug-75', 'Mar-65', 'Aug-77', 'Aug-41', 'Jan-67', 'Sep-40',
       'Apr-63', 'Sep-76', 'Apr-78', 'Jun-71', 'Dec-70', 'May-84',
       'Nov-86', 'Nov-72', 'Dec-75', 'Jul-79', 'Jul-65', 'Apr-05',
       'Jul-78', 'Jan-73', 'Dec-45', 'Jun-84', 'May-85', 'Sep-86',
       'Dec-64', 'Dec-68', 'Nov-85', 'Apr-79', 'May-90', 'Mar-92',
       'Feb-73', 'Dec-73', 'Nov-62', 'Sep-88', 'Jul-75', 'Jan-78',
       'Oct-48', 'Aug-64', 'Jan-81', 'Mar-76', 'Aug-84', 'Nov-65',
       'Feb-75', 'Apr-81', 'Feb-04', 'Oct-75', 'Jul-71', 'Aug-79',
       'Feb-83', 'Mar-72', 'Sep-67', 'Nov-80', 'Nov-63', 'May-71',
       'Aug-83', 'Sep-73', 'Dec-83', 'Sep-00', 'Oct-66', 'Sep-03',
```

Ввод [19]:
```
le = LabelEncoder()
cat_enc_le = le.fit_transform(cat_enc['c1'])
```

Ввод [20]:
```
le.classes_
```

```
'Aug-65', 'Aug-66', 'Aug-67', 'Aug-68', 'Aug-69', 'Aug-70',
'Aug-71', 'Aug-72', 'Aug-73', 'Aug-74', 'Aug-75', 'Aug-76',
'Aug-77', 'Aug-78', 'Aug-79', 'Aug-80', 'Aug-81', 'Aug-82',
'Aug-83', 'Aug-84', 'Aug-85', 'Aug-86', 'Aug-87', 'Aug-88',
'Aug-89', 'Aug-90', 'Aug-91', 'Aug-92', 'Aug-93', 'Aug-94',
'Aug-95', 'Aug-96', 'Aug-97', 'Aug-98', 'Aug-99', 'Dec-00',
'Dec-01', 'Dec-02', 'Dec-03', 'Dec-04', 'Dec-05', 'Dec-06',
'Dec-07', 'Dec-08', 'Dec-09', 'Dec-10', 'Dec-11', 'Dec-12',
'Dec-39', 'Dec-40', 'Dec-41', 'Dec-42', 'Dec-43', 'Dec-44',
'Dec-45', 'Dec-46', 'Dec-47', 'Dec-48', 'Dec-50', 'Dec-51',
'Dec-52', 'Dec-53', 'Dec-54', 'Dec-56', 'Dec-60', 'Dec-61',
'Dec-62', 'Dec-63', 'Dec-64', 'Dec-65', 'Dec-66', 'Dec-67',
'Dec-68', 'Dec-69', 'Dec-70', 'Dec-71', 'Dec-72', 'Dec-73',
'Dec-74', 'Dec-75', 'Dec-76', 'Dec-77', 'Dec-78', 'Dec-79',
'Dec-80', 'Dec-81', 'Dec-82', 'Dec-83', 'Dec-84', 'Dec-85',
'Dec-86', 'Dec-87', 'Dec-88', 'Dec-89', 'Dec-90', 'Dec-91',
'Dec-92', 'Dec-93', 'Dec-94', 'Dec-95', 'Dec-96', 'Dec-97',
'Dec-98', 'Dec-99', 'Feb-00', 'Feb-01', 'Feb-02', 'Feb-03',
'Feb-04', 'Feb-05', 'Feb-06', 'Feb-07', 'Feb-08', 'Feb-09',
'Feb-10', 'Feb-11', 'Feb-12', 'Feb-13', 'Feb-40', 'Feb-41',
```

Ввод [21]:
```
np.unique(cat_enc_le)
```

Out[21]:
```
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,  53,  54,  55,  56,  57,  58,  59,  60,  61,  62,  63,  64,
        65,  66,  67,  68,  69,  70,  71,  72,  73,  74,  75,  76,  77,
        78,  79,  80,  81,  82,  83,  84,  85,  86,  87,  88,  89,  90,
        91,  92,  93,  94,  95,  96,  97,  98,  99, 100, 101, 102, 103,
       104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116,
       117, 118, 119, 120, 121, 122, 123, 124, 125, 126, 127, 128, 129,
       130, 131, 132, 133, 134, 135, 136, 137, 138, 139, 140, 141, 142,
       143, 144, 145, 146, 147, 148, 149, 150, 151, 152, 153, 154, 155,
       156, 157, 158, 159, 160, 161, 162, 163, 164, 165, 166, 167, 168,
       169, 170, 171, 172, 173, 174, 175, 176, 177, 178, 179, 180, 181,
       182, 183, 184, 185, 186, 187, 188, 189, 190, 191, 192, 193, 194,
       195, 196, 197, 198, 199, 200, 201, 202, 203, 204, 205, 206, 207,
       208, 209, 210, 211, 212, 213, 214, 215, 216, 217, 218, 219, 220,
       221, 222, 223, 224, 225, 226, 227, 228, 229, 230, 231, 232, 233,
       234, 235, 236, 237, 238, 239, 240, 241, 242, 243, 244, 245, 246,
       247, 248, 249, 250, 251, 252, 253, 254, 255, 256, 257, 258, 259,
       260, 261, 262, 263, 264, 265, 266, 267, 268, 269, 270, 271, 272,
       273, 274, 275, 276, 277, 278, 279, 280, 281, 282, 283, 284, 285,
       286, 287, 288, 289, 290, 291, 292, 293, 294, 295, 296, 297, 298,
       299, 300, 301, 302, 303, 304, 305, 306, 307, 308, 309, 310, 311,
       312, 313, 314, 315, 316, 317, 318, 319, 320, 321, 322, 323, 324,
       325, 326, 327, 328, 329, 330, 331, 332, 333, 334, 335, 336, 337,
       338, 339, 340, 341, 342, 343, 344, 345, 346, 347, 348, 349, 350,
       351, 352, 353, 354, 355, 356, 357, 358, 359, 360, 361, 362, 363,
       364, 365, 366, 367, 368, 369, 370, 371, 372, 373, 374, 375, 376,
       377, 378, 379, 380, 381, 382, 383, 384, 385, 386, 387, 388, 389,
       390, 391, 392, 393, 394, 395, 396, 397, 398, 399, 400, 401, 402,
       403, 404, 405, 406, 407, 408, 409, 410, 411, 412, 413, 414, 415,
       416, 417, 418, 419, 420, 421, 422, 423, 424, 425, 426, 427, 428,
       429, 430, 431, 432, 433, 434, 435, 436, 437, 438, 439, 440, 441,
       442, 443, 444, 445, 446, 447, 448, 449, 450, 451, 452, 453, 454,
       455, 456, 457, 458, 459, 460, 461, 462, 463, 464, 465, 466, 467,
       468, 469, 470, 471, 472, 473, 474, 475, 476, 477, 478, 479, 480
```

Ввод [22]:
```
#Для начала заполним пропущенные данные
cat_temp_data2 = data[['SEX']]
cat_temp_data2.head()
```

Out[22]:

	SEX
0	Male Characters
1	Male Characters
2	Male Characters
3	Male Characters
4	Male Characters

Ввод [23]:
```
cat_temp_data2['SEX'].unique()
```

Out[23]:
```
array(['Male Characters', 'Female Characters', 'Genderfluid Characters',
       'Agender Characters', nan], dtype=object)
```

Ввод [24]:
```
# Импьютация наиболее частыми значениями
imp4 = SimpleImputer(missing_values=np.nan, strategy='most_frequent')
data_imp4 = imp4.fit_transform(cat_temp_data2)
data_imp4
```

Out[24]:
```
array([['Male Characters'],
       ['Male Characters'],
       ['Male Characters'],
       ...,
       ['Male Characters'],
       ['Male Characters'],
       ['Male Characters']], dtype=object)
```

Ввод [25]:
```
cat_enc4 = pd.DataFrame({'c2':data_imp4.T[0]})
cat_enc4
```

Out[25]:

	c2
0	Male Characters
1	Male Characters
2	Male Characters
3	Male Characters
4	Male Characters
...	...
16371	Male Characters
16372	Male Characters
16373	Male Characters
16374	Male Characters
16375	Male Characters

16376 rows × 1 columns

Ввод [26]:
```
pd.get_dummies(cat_enc4).head()
```

Out[26]:

	c2_Agender Characters	c2_Female Characters	c2_Genderfluid Characters	c2_Male Characters
0	0	0	0	1

Ввод [27]:
```
ohe = OneHotEncoder()
cat_enc_ohe = ohe.fit_transform(cat_enc4[['c2']])
```

Ввод [28]:
```
cat_enc_ohe
```

Out[28]:
```
<16376x4 sparse matrix of type '<class 'numpy.float64'>'
        with 16376 stored elements in Compressed Sparse Row format>
```

Ввод [29]:
```
cat_enc_ohe.todense()[0:10]
```

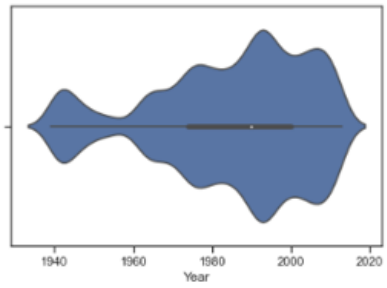
Out[29]:
```
matrix([[0., 0., 0., 1.],
        [0., 0., 0., 1.],
        [0., 0., 0., 1.],
        [0., 0., 0., 1.],
        [0., 0., 0., 1.],
        [0., 0., 0., 1.],
        [0., 0., 0., 1.],
        [0., 0., 0., 1.],
        [0., 0., 0., 1.],
        [0., 0., 0., 1.]])
```

Построение скрипичной диаграммы

Ввод [34]:
```
import seaborn as sns
```

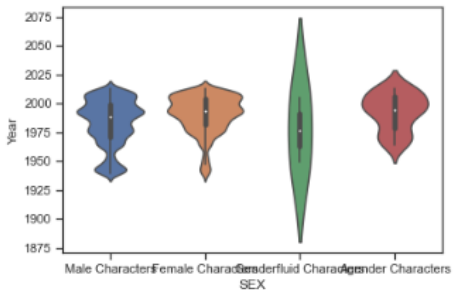
Ввод [35]:
```
sns.violinplot(x=data['Year'])
```

Out[35]:
```
<AxesSubplot:xlabel='Year'>
```



Ввод [37]:
```
# Распределение параметра YEAR сгруппированные по SEX.
sns.violinplot(x='SEX', y='Year', data=data)
```

Out[37]:
```
<AxesSubplot:xlabel='SEX', ylabel='Year'>
```



Ввод [ ]: