Московский государственный технический университет им. Н.Э. Баумана

Факультет «Информатика и системы управления»

Кафедра «Системы обработки информации и управления»

Курс «Технологии машинного обучения»

Отчет по лабораторной работе № 2
«Обработка пропусков в данных, кодирование категориальных признаков, масштабирование данных»

Выполнил:

студент группы ИУ5-65Б

Герасименко А.В.

Подпись и дата:

Проверил:

преподаватель каф. ИУ5

Гапанюк Ю.Е.

Подпись и дата:

г. Москва, 2021 г.

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

Загрузка и первичный анализ данных

In [2]:
```
# Будем использовать выборку по вселенной Marvel
data = pd.read_csv('TMO LR2 Marvel.csv', sep=",")
```

In [3]:
```
# размер набора данных
data.shape
```

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

In [4]:
```
# типы колонок
data.dtypes
```

Out[4]:
```
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
```

In [5]:
```
# проверим есть ли пропущенные значения
data.isnull().sum()
```

Out[5]:
```
page_id                0
name                   0
urlslug                0
ID                  3770
ALIGN               2812
EYE                 9767
HAIR                4264
SEX                  854
GSM                16286
ALIVE                  3
APPEARANCES         1096
FIRST APPEARANCE     815
Year                 815
dtype: int64
```

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

Out[6]:

	page_id	name	urlslug	ID	ALIGN	EYE	HAIR
0	1678	Spider-Man (Peter	\/Spider-Man_(Peter_Parker)	Secret Identity	Good Characters	Hazel Eyes	Brown Hair

	page_id	name	urlslug	ID	ALIGN	EYE	HAIR
		Parker)					
1	7139	Captain America (Steven Rogers)	\/Captain_America_(Steven_Rogers)	Public Identity	Good Characters	Blue Eyes	White Hair
2	64786	Wolverine (James \"Logan\" Howlett)	\/Wolverine_(James_%22Logan%22_Howlett)	Public Identity	Neutral Characters	Blue Eyes	Black Hair
3	1868	Iron Man (Anthony \"Tony\" Stark)	\/Iron_Man_(Anthony_%22Tony%22_Stark)	Public Identity	Good Characters	Blue Eyes	Black Hair
		Thor		No			

In [7]:
```
total_count = data.shape[0]
print('Всего строк: {}'.format(total_count))
```

```
Всего строк: 16376
```

Обработка пропусков данных

In [8]:
```
# Удаление колонок, содержащих пустые значения
data_new_1 = data.dropna(axis=1, how='any')
(data.shape, data_new_1.shape)
```

Out[8]: ((16376, 13), (16376, 3))

Слишком много столбцов подверглось удалению

In [9]:
```
# Удаление строк, содержащих пустые значения
data_new_2 = data.dropna(axis=0, how='any')
(data.shape, data_new_2.shape)
```

Out[9]: ((16376, 13), (58, 13))

Слишком много строк подверглось удалению

Удаление столбцов с большим процентом пропусков данных (>10%)

In [10]:
```
# 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' or dt=='object'):
        num_cols.append(col)
        temp_perc = round((temp_null_count / total_count) * 100.0, 2)
        print('Колонка {}. Тип данных {}. Количество пустых значений {}, {}
```

```
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 APPEARANCES. Тип данных float64. Количество пустых значений 1096,
6.69%.
Колонка FIRST APPEARANCE. Тип данных object. Количество пустых значений 81
5, 4.98%.
Колонка Year. Тип данных float64. Количество пустых значений 815, 4.98%.
```

In [11]:
```
# Удаление колонок, содержащих пустые значения
data_new = pd.read_csv('TMO LR2 Marvel.csv', sep=",")
del data_new['ID']
del data_new['ALIGN']
del data_new['EYE']
del data_new['HAIR']
del data_new['GSM']
(data.shape, data_new.shape)
```

Out[11]:
```
((16376, 13), (16376, 8))
```

In [12]:
```
data_new.head()
```

Out[12]:

	page_id	name	urlslug	SEX	ALIVE	APPEARAN
0	1678	Spider-Man (Peter Parker)	\/Spider-Man_(Peter_Parker)	Male Characters	Living Characters	404
1	7139	Captain America (Steven Rogers)	\/Captain_America_(Steven_Rogers)	Male Characters	Living Characters	336
2	64786	Wolverine (James \"Logan\" Howlett)	\/Wolverine_(James_%22Logan%22_Howlett)	Male Characters	Living Characters	306
3	1868	Iron Man (Anthony \"Tony\" Stark)	\/Iron_Man_(Anthony_%22Tony%22_Stark)	Male Characters	Living Characters	296
4	2460	Thor (Thor Odinson)	\/Thor_(Thor_Odinson)	Male Characters	Living Characters	225

In [13]:
```
# Проверим оставшиеся колонки
num_cols = []
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)
    if temp_null_count>0 and (dt=='float64' or dt=='int64' or dt=='object'):
        num_cols.append(col)
        temp_perc = round((temp_null_count / total_count) * 100.0, 2)
        print('Колонка {}. Тип данных {}. Количество пустых значений {}, {}
```

```
Колонка SEX. Тип данных object. Количество пустых значений 854, 5.21%.
Колонка ALIVE. Тип данных object. Количество пустых значений 3, 0.02%.
Колонка APPEARANCES. Тип данных float64. Количество пустых значений 1096,
```

```
6.69%.
Колонка FIRST APPEARANCE. Тип данных object. Количество пустых значений 81
5, 4.98%.
Колонка Year. Тип данных float64. Количество пустых значений 815, 4.98%.
```

In [14]:
```
# Проверим оставшиеся колонки
num_cols = []
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)
    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 {}, {}
```

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

Выберем только числовые столбцы в оставшейся выборке

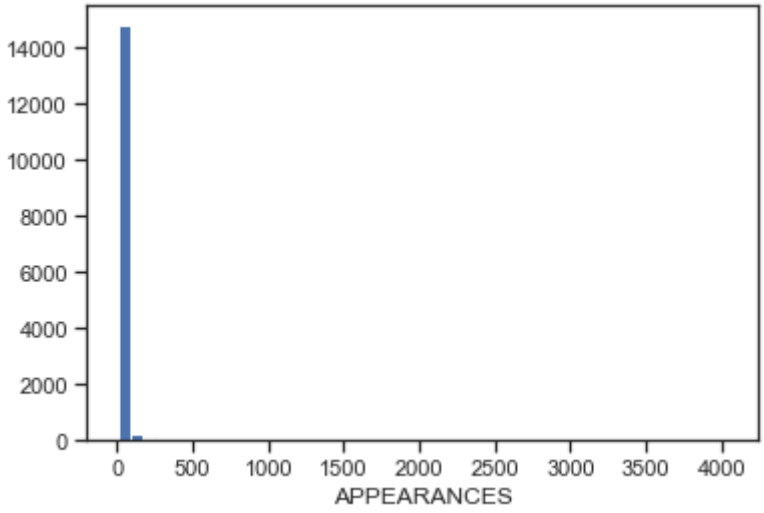
In [15]:
```
# Фильтр по колонкам с пропущенными значениями
data_num = data_new[num_cols]
data_num
```

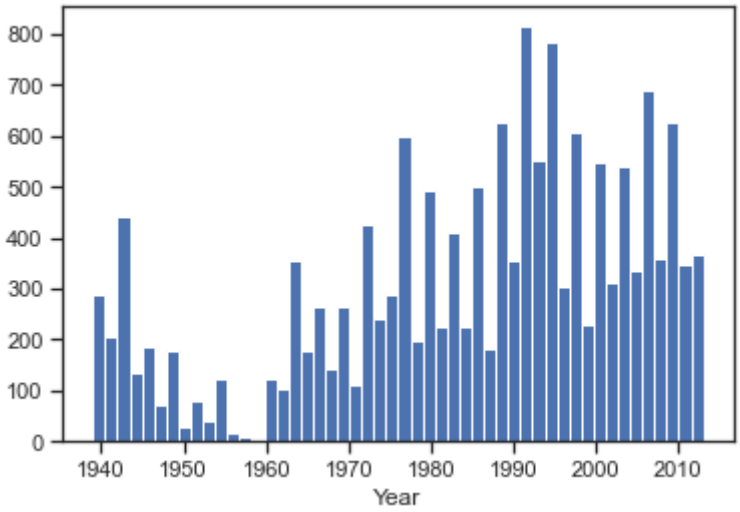
Out[15]:

	APPEARANCES	Year
0	4043.0	1962.0
1	3360.0	1941.0
2	3061.0	1974.0
3	2961.0	1963.0
4	2258.0	1950.0
...	...	...
16371	NaN	NaN
16372	NaN	NaN
16373	NaN	NaN
16374	NaN	NaN
16375	NaN	NaN

16376 rows × 2 columns

In [16]:
```
# Гистограмма по признакам
for col in data_num:
    plt.hist(data_new[col], 50)
    plt.xlabel(col)
    plt.show()
```





Проанализируем каким способом заполнить пропуски данных в столбце APPEARANCES

In [17]:
```
data_num_Appearances = data_num[['APPEARANCES']]
data_num_Appearances.head()
```

Out[17]:

	APPEARANCES
0	4043.0
1	3360.0
2	3061.0
3	2961.0
4	2258.0

In [18]:
```
from sklearn.impute import SimpleImputer
from sklearn.impute import MissingIndicator
```

In [19]:
```
# Фильтр для проверки заполнения пустых значений
indicator = MissingIndicator()
mask_missing_values_only = indicator.fit_transform(data_num_Appearances)
mask_missing_values_only
```

Out[19]:
```
array([[False],
```

```
       [False],
       [False],
       ...,
       [ True],
       [ True],
```

In [20]:
```
strategies=['mean', 'median', 'most_frequent']
```

In [21]:
```
def test_num_impute(strategy_param):
    imp_num = SimpleImputer(strategy=strategy_param)
    data_num_imp = imp_num.fit_transform(data_num_Appearances)
    return data_num_imp[mask_missing_values_only]
```

In [22]:
```
strategies[0], test_num_impute(strategies[0])
```

Out[22]:
```
('mean',
 array([17.03337696, 17.03337696, 17.03337696, ..., 17.03337696,
        17.03337696, 17.03337696]))
```

In [23]:
```
strategies[1], test_num_impute(strategies[1])
```

Out[23]:
```
('median', array([3., 3., 3., ..., 3., 3., 3.]))
```

In [24]:
```
strategies[2], test_num_impute(strategies[2])
```

Out[24]:
```
('most_frequent', array([1., 1., 1., ..., 1., 1., 1.]))
```

Пропуски из столбца Appearances заполним самым часто встречающимся значением ('most_frequent')

In [25]:
```
data_new['APPEARANCES'] = data_new['APPEARANCES'].fillna(value = 1)
data_new
```

Out[25]:

	page_id	name	urlslug	SEX	ALIVE	APPE
0	1678	Spider-Man (Peter Parker)	\/Spider-Man_(Peter_Parker)	Male Characters	Living Characters	
1	7139	Captain America (Steven Rogers)	\/Captain_America_(Steven_Rogers)	Male Characters	Living Characters	
2	64786	Wolverine (James \"Logan\" Howlett)	\/Wolverine_(James_%22Logan%22_Howlett)	Male Characters	Living Characters	
3	1868	Iron Man (Anthony \"Tony\" Stark)	\/Iron_Man_(Anthony_%22Tony%22_Stark)	Male Characters	Living Characters	
4	2460	Thor (Thor Odinson)	\/Thor_(Thor_Odinson)	Male Characters	Living Characters	
...	...	...	...	...	...	
16371	657508	Ru'ach (Earth-616)	\/Ru%27ach_(Earth-616)	Male Characters	Living Characters	

	page_id	name	urlslug	SEX	ALIVE	APPE
16372	665474	Thane (Thanos' son) (Earth-616)	\/Thane_(Thanos%27_son)_(Earth-616)	Male Characters	Living Characters	
16373	695217	Tinkerer (Skrull) (Earth-616)	\/Tinkerer_(Skrull)_(Earth-616)	Male Characters	Living Characters	
16374	708811	TK421 (Spiderling) (Earth-616)	\/TK421_(Spiderling)_(Earth-616)	Male Characters	Living Characters	
		Yologarch			Living	

Проанализируем каким способом заполнить пропуски данных в столбце Year

In [26]:
```
data_num_Year = data_num[['Year']]
data_num_Year.head()
```

Out[26]:

	Year
0	1962.0
1	1941.0
2	1974.0
3	1963.0
4	1950.0

In [27]:
```
# Фильтр для проверки заполнения пустых значений
indicator = MissingIndicator()
mask_missing_values_only = indicator.fit_transform(data_num_Year)
mask_missing_values_only
```

Out[27]:
```
array([[False],
       [False],
       [False],
       ...,
       [ True],
       [ True],
       [ True]])
```

In [28]:
```
def test_num_impute(strategy_param):
    imp_num = SimpleImputer(strategy=strategy_param)
    data_num_imp = imp_num.fit_transform(data_num_Year)
    return data_num_imp[mask_missing_values_only]
```

In [29]:
```
strategies[0], test_num_impute(strategies[0])
```

Out[29]:
```
('mean',
 array([1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
        1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
        1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
        1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
        1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
        1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
```

```
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
```

```
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
```

```
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258, 1984.95180258,
       1984.95180258, 1984.95180258, 1984.95180258]))
```

In [30]:

```
strategies[1], test_num_impute(strategies[1])
```

Out[30]:

```
('median',
 array([1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
        1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
        1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
        1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
        1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
        1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
        1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
        1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
        1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
```

```
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
```

```
        1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
        1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
        1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
        1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
        1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
        1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
        1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
        1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
        1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
        1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
        1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990., 1990.,
        1990., 1990., 1990., 1990., 1990.]))
```

In [31]:

```
strategies[2], test_num_impute(strategies[2])
```

Out[31]:

```
('most_frequent',
 array([1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
        1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
```

```
       1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
       1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
       1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
       1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
       1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
       1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
       1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
       1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
       1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
       1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
       1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
       1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
       1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
       1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
       1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
       1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
       1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
       1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
       1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
       1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
       1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
       1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
       1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
       1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
       1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
       1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
       1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
       1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
       1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
       1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
       1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
       1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
       1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
       1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
       1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
       1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
       1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993., 1993.,
       1993., 1993., 1993., 1993., 1993.]))
```

Пропуски из столбца Year заполним медианным значением ('median'), так как на гистограмме есть большые перепады значений

In [32]:
```
data_new['Year'] = data_new['Year'].fillna(value = 1990)
data_new
```

Out[32]:

	page_id	name	urlslug	SEX	ALIVE	APPE
0	1678	Spider-Man (Peter Parker)	\/Spider-Man_(Peter_Parker)	Male Characters	Living Characters	
1	7139	Captain America (Steven Rogers)	\/Captain_America_(Steven_Rogers)	Male Characters	Living Characters	
2	64786	Wolverine (James \"Logan\" Howlett)	\/Wolverine_(James_%22Logan%22_Howlett)	Male Characters	Living Characters	
3	1868	Iron Man (Anthony \"Tony\" Stark)	\/Iron_Man_(Anthony_%22Tony%22_Stark)	Male Characters	Living Characters	

	page_id	name	urlslug	SEX	ALIVE	APPE
4	2460	Thor (Thor Odinson)	\/Thor_(Thor_Odinson)	Male Characters	Living Characters	
...	...	...	...	...	...	
16371	657508	Ru'ach (Earth-616)	\/Ru%27ach_(Earth-616)	Male Characters	Living Characters	
16372	665474	Thane (Thanos' son) (Earth-616)	\/Thane_(Thanos%27_son)_(Earth-616)	Male Characters	Living Characters	
16373	695217	Tinkerer (Skrull) (Earth-616)	\/Tinkerer_(Skrull)_(Earth-616)	Male Characters	Living Characters	
16374	708811	TK421 (Spiderling) (Earth-616)	\/TK421_(Spiderling)_(Earth-616)	Male Characters	Living Characters	

Проверим остались ли пропуски в числовых данных

In [33]:
```
num_cols = []
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)
    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 {}, {}
```

Видим, что пропусков нет

Выберем категориальные признаки

In [34]:
```
num_cols = []
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)
    if temp_null_count>0 and (dt=='object'):
        num_cols.append(col)
        temp_perc = round((temp_null_count / total_count) * 100.0, 2)
        print('Колонка {}. Тип данных {}. Количество пустых значений {}, {}
```

```
Колонка SEX. Тип данных object. Количество пустых значений 854, 5.21%.
Колонка ALIVE. Тип данных object. Количество пустых значений 3, 0.02%.
Колонка FIRST APPEARANCE. Тип данных object. Количество пустых значений 81
5, 4.98%.
```

Найдём самые часто встречающиеся значения в каждой из этих колонок и заполним пропуски.

Заполним пропуски в категориальных столбцах

In [35]:
```
cat_temp_data = data[['SEX']]
cat_temp_data.head()
```

Out[35]:

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

In [36]:
```
cat_temp_data['SEX'].unique()
```

Out[36]:
```
array(['Male Characters', 'Female Characters', 'Genderfluid Characters',
       'Agender Characters', nan], dtype=object)
```

In [37]:
```
cat_temp_data[cat_temp_data['SEX'].isnull()].shape
```

Out[37]:
```
(854, 1)
```

In [38]:
```
# Импьютация наиболее частыми значениями
imp1 = SimpleImputer(missing_values=np.nan, strategy='most_frequent')
data_imp1 = imp1.fit_transform(cat_temp_data)
data_imp1
```

Out[38]:
```
array([['Male Characters'],
       ['Male Characters'],
       ['Male Characters'],
       ...,
       ['Male Characters'],
       ['Male Characters'],
       ['Male Characters']], dtype=object)
```

In [39]:
```
data_new['SEX'] = data_new['SEX'].fillna(value = 'Male Characters')
data_new
```

Out[39]:

	page_id	name	urlslug	SEX	ALIVE	APPE
0	1678	Spider-Man (Peter Parker)	\/Spider-Man_(Peter_Parker)	Male Characters	Living Characters	
1	7139	Captain America (Steven Rogers)	\/Captain_America_(Steven_Rogers)	Male Characters	Living Characters	
2	64786	Wolverine (James \"Logan\" Howlett)	\/Wolverine_(James_%22Logan%22_Howlett)	Male Characters	Living Characters	
3	1868	Iron Man (Anthony \"Tony\" Stark)	\/Iron_Man_(Anthony_%22Tony%22_Stark)	Male Characters	Living Characters	
4	2460	Thor (Thor Odinson)	\/Thor_(Thor_Odinson)	Male Characters	Living Characters	

	page_id	name	urlslug	SEX	ALIVE	APPE
...	...	...	...	...	...	
16371	657508	Ru'ach (Earth-616)	\/Ru%27ach_(Earth-616)	Male Characters	Living Characters	
16372	665474	Thane (Thanos' son) (Earth-616)	\/Thane_(Thanos%27_son)_(Earth-616)	Male Characters	Living Characters	
16373	695217	Tinkerer (Skrull) (Earth-616)	\/Tinkerer_(Skrull)_(Earth-616)	Male Characters	Living Characters	
16374	708811	TK421 (Spiderling) (Earth-616)	\/TK421_(Spiderling)_(Earth-616)	Male Characters	Living Characters	
16375	673702	Yologarch (Earth-616)	\/Yologarch_(Earth-616)	Male Characters	Living Characters	

In [40]:
```
cat_temp_data = data[['ALIVE']]
cat_temp_data.head()
```

Out[40]:

	ALIVE
0	Living Characters
1	Living Characters
2	Living Characters
3	Living Characters
4	Living Characters

In [41]:
```
cat_temp_data['ALIVE'].unique()
```

Out[41]:
```
array(['Living Characters', 'Deceased Characters', nan], dtype=object)
```

In [42]:
```
cat_temp_data[cat_temp_data['ALIVE'].isnull()].shape
```

Out[42]:
```
(3, 1)
```

In [43]:
```
# Импьютация наиболее частыми значениями
imp2 = SimpleImputer(missing_values=np.nan, strategy='most_frequent')
data_imp2 = imp2.fit_transform(cat_temp_data)
data_imp2
```

Out[43]:
```
array([['Living Characters'],
       ['Living Characters'],
       ['Living Characters'],
       ...,
       ['Living Characters'],
       ['Living Characters'],
       ['Living Characters']], dtype=object)
```

In [44]:
```
data_new['ALIVE'] = data_new['ALIVE'].fillna(value = 'Living Characters')
data_new
```

Out[44]:

	page_id	name	urlslug	SEX	ALIVE	APPE
0	1678	Spider-Man (Peter Parker)	\/Spider-Man_(Peter_Parker)	Male Characters	Living Characters	
1	7139	Captain America (Steven Rogers)	\/Captain_America_(Steven_Rogers)	Male Characters	Living Characters	
2	64786	Wolverine (James \"Logan\" Howlett)	\/Wolverine_(James_%22Logan%22_Howlett)	Male Characters	Living Characters	
3	1868	Iron Man (Anthony \"Tony\" Stark)	\/Iron_Man_(Anthony_%22Tony%22_Stark)	Male Characters	Living Characters	
4	2460	Thor (Thor Odinson)	\/Thor_(Thor_Odinson)	Male Characters	Living Characters	
...	...	...	...	...	...	
16371	657508	Ru'ach (Earth-616)	\/Ru%27ach_(Earth-616)	Male Characters	Living Characters	
16372	665474	Thane (Thanos' son) (Earth-616)	\/Thane_(Thanos%27_son)_(Earth-616)	Male Characters	Living Characters	
16373	695217	Tinkerer (Skrull) (Earth-616)	\/Tinkerer_(Skrull)_(Earth-616)	Male Characters	Living Characters	
16374	708811	TK421 (Spiderling) (Earth-616)	\/TK421_(Spiderling)_(Earth-616)	Male Characters	Living Characters	
16375	673702	Yologarch (Earth-616)	\/Yologarch_(Earth-616)	Male Characters	Living Characters	

In [45]:
```
cat_temp_data = data[['FIRST APPEARANCE']]
cat_temp_data.head()
```

Out[45]:

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

In [46]:
```
cat_temp_data[cat_temp_data['FIRST APPEARANCE'].isnull()].shape
```

Out[46]:
```
(815, 1)
```

In [47]:
```
# Импьютация наиболее частыми значениями
imp3 = SimpleImputer(missing_values=np.nan, strategy='most_frequent')
data_imp3 = imp3.fit_transform(cat_temp_data)
data_imp3
```

Out[47]:
```
array([['Aug-62'],
       ['Mar-41'],
       ['Oct-74'],
       ...,
       ['Jun-92'],
       ['Jun-92'],
       ['Jun-92']], dtype=object)
```

In [48]:
```
data_new['FIRST APPEARANCE'] = data_new['FIRST APPEARANCE'].fillna(value =
data_new
```

Out[48]:

	page_id	name	urlslug	SEX	ALIVE	APPE
0	1678	Spider-Man (Peter Parker)	\/Spider-Man_(Peter_Parker)	Male Characters	Living Characters	
1	7139	Captain America (Steven Rogers)	\/Captain_America_(Steven_Rogers)	Male Characters	Living Characters	
2	64786	Wolverine (James \"Logan\" Howlett)	\/Wolverine_(James_%22Logan%22_Howlett)	Male Characters	Living Characters	
3	1868	Iron Man (Anthony \"Tony\" Stark)	\/Iron_Man_(Anthony_%22Tony%22_Stark)	Male Characters	Living Characters	
4	2460	Thor (Thor Odinson)	\/Thor_(Thor_Odinson)	Male Characters	Living Characters	
...	...	...	...	...	...	
16371	657508	Ru'ach (Earth-616)	\/Ru%27ach_(Earth-616)	Male Characters	Living Characters	
16372	665474	Thane (Thanos' son) (Earth-616)	\/Thane_(Thanos%27_son)_(Earth-616)	Male Characters	Living Characters	
16373	695217	Tinkerer (Skrull) (Earth-616)	\/Tinkerer_(Skrull)_(Earth-616)	Male Characters	Living Characters	
16374	708811	TK421 (Spiderling) (Earth-616)	\/TK421_(Spiderling)_(Earth-616)	Male Characters	Living Characters	
16375	673702	Yologarch (Earth-616)	\/Yologarch_(Earth-616)	Male Characters	Living Characters	

16376 rows × 8 columns

Проверим есть ли пропущенные значения

In [49]:
```
data_new.isnull().sum()
```

Out[49]:
```
page_id             0
name                0
urlslug             0
SEX                 0
ALIVE               0
APPEARANCES         0
FIRST APPEARANCE    0
Year                0
dtype: int64
```

Выполним кодирование категориальных признаков целочисленными значениями - label encoding

In [50]:
```
cat_enc = pd.DataFrame({'c1':data_imp3.T[0]})
cat_enc
```

Out[50]:

	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

In [51]:
```
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
```

In [52]:
```
le = LabelEncoder()
cat_enc_le = le.fit_transform(cat_enc['c1'])
```

In [53]:
```
cat_enc['c1'].unique()
```

Out[53]:
```
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',
'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', 'Nov-74', 'Apr-80', 'Feb-02', 'Feb-76', 'Nov-77',
'Dec-89', 'Mar-91', 'Jan-68', 'Apr-75', 'May-80', 'Nov-87',
'Feb-78', 'Oct-77', 'Jun-86', 'Jan-70', 'Jun-85', 'Nov-90',
'Dec-63', 'Jul-66', 'Jun-73', 'Jun-76', 'Oct-85', 'Jan-05',
'Sep-87', 'Jun-61', 'Nov-83', 'Jan-63', 'Feb-66', 'Oct-93',
'Oct-04', 'Jul-77', 'Apr-83', 'Mar-67', 'Jan-04', 'Aug-66',
'Dec-53', 'Nov-43', 'Feb-95', 'Dec-40', 'May-47', 'Feb-65',
'Dec-94', 'May-66', 'Jun-77', 'Mar-86', 'Aug-40', 'Jan-03',
'Nov-76', 'Jan-40', 'Jul-76', 'Aug-81', 'Oct-89', 'Dec-02',
'Jun-49', 'Apr-74', 'May-86', 'Jul-02', 'Apr-84', 'Jan-76',
'Sep-94', 'Aug-05', 'Sep-68', 'Oct-72', 'Apr-67', 'Oct-56',
'Apr-72', 'May-83', 'Mar-88', 'Jul-01', 'Oct-03', 'Aug-02',
'Oct-90', 'Apr-77', 'Jan-88', 'Jun-80', 'Jul-84', 'Sep-85',
'Oct-94', 'Sep-05', 'Aug-69', 'Feb-81', 'Mar-48', 'Jul-03',
'Mar-75', 'Nov-10', 'Mar-05', 'Jan-06', 'Nov-03', 'Aug-63',
'Sep-66', 'Sep-46', 'Sep-65', 'Jul-74', 'Nov-75', 'Apr-76',
'Dec-78', 'Jun-95', 'Sep-98', 'Aug-04', 'Feb-09', 'Jan-97',
'Sep-04', 'May-67', 'Dec-48', 'Aug-90', 'Aug-86', 'Nov-60',
'May-68', 'Dec-79', 'Apr-85', 'Oct-92', 'Feb-67', 'Feb-84',
'Aug-91', 'Jul-80', 'Oct-80', 'Mar-85', 'May-96', 'Oct-73',
'Aug-76', 'Jul-04', 'Mar-06', 'Oct-86', 'Aug-45', 'Nov-81',
'Oct-63', 'May-65', 'Feb-86', 'Jan-92', 'Apr-97', 'Jun-11',
'Sep-75', 'Apr-66', 'Feb-79', 'Dec-86', 'Jun-97', 'Oct-01',
'Jun-69', 'Apr-70', 'Aug-82', 'Oct-87', 'Jun-04', 'Nov-40',
'Aug-54', 'Mar-89', 'Jun-91', 'Aug-10', 'Sep-11', 'Dec-50',
'Jun-05', 'Jan-12', 'Oct-61', 'Aug-68', 'Sep-83', 'Jul-93',
'Nov-98', 'Oct-09', 'Jul-72', 'Oct-79', 'Sep-82', 'Sep-84',
'Sep-06', 'Mar-54', 'Apr-69', 'Nov-69', 'Jan-74', 'Jun-79',
'Sep-80', 'Mar-10', 'Feb-92', 'Dec-99', 'May-09', 'May-55',
'Dec-72', 'Oct-81', 'Jul-91', 'May-05', 'Dec-80', 'Oct-82',
'Feb-71', 'Feb-06', 'Jun-88', 'Jun-00', 'Dec-08', 'Sep-54',
'Feb-63', 'Sep-74', 'Nov-97', 'Sep-02', 'Nov-78', 'Aug-80',
'Apr-06', 'Aug-70', 'Sep-79', 'Mar-73', 'Sep-10', 'Jan-11',
'Mar-44', 'Jan-69', 'Jun-90', 'Sep-91', 'Apr-10', 'Nov-67',
'May-69', 'Jun-07', 'Aug-74', 'Aug-88', 'Nov-89', 'Dec-10',
'Jan-13', 'Oct-41', 'Jun-66', 'Jan-93', 'Dec-95', 'Jan-95',
'Dec-54', 'Sep-77', 'Aug-85', 'Aug-93', 'May-00', 'Jan-09',
'Aug-48', 'May-54', 'May-93', 'Mar-93', 'Feb-97', 'Aug-01',
'Jul-09', 'Jun-41', 'Nov-51', 'May-76', 'Mar-77', 'Jun-89',
'Oct-91', 'Jun-93', 'May-98', 'Oct-10', 'May-72', 'Aug-73',
'Dec-74', 'Jan-96', 'Nov-96', 'Apr-00', 'May-07', 'Jun-08',
'Feb-60', 'May-79', 'Dec-01', 'Jul-08', 'May-64', 'Jan-66',
'Jan-87', 'Aug-98', 'Dec-11', 'Dec-62', 'Sep-70', 'Oct-98',
'Feb-03', 'Nov-08', 'Jul-87', 'Feb-87', 'Oct-95', 'Mar-43',
'Nov-71', 'Jan-72', 'Feb-82', 'Jul-86', 'Sep-95', 'Mar-03',
'Dec-04', 'Aug-71', 'Apr-73', 'Jun-74', 'Dec-77', 'Sep-89',
'Feb-90', 'Apr-90', 'Dec-91', 'Jul-92', 'Mar-70', 'Jan-77',
'Jan-82', 'Jul-94', 'Sep-01', 'Jun-03', 'Aug-78', 'Jan-85',
'Feb-89', 'Aug-92', 'Dec-92', 'Nov-92', 'Mar-97', 'Apr-02',
'Feb-13', 'Aug-89', 'Nov-05', 'Jan-07', 'Apr-13', 'Nov-52',
'Jan-83', 'Oct-02', 'May-06', 'Jul-53', 'Oct-60', 'Dec-90',
'May-92', 'Mar-99', 'Feb-01', 'Feb-70', 'Jan-71', 'Jul-89',
```

```
       'May-02', 'Jun-83', 'Mar-83', 'Dec-87', 'Mar-87', 'Nov-91',
       'Mar-94', 'Mar-95', 'Oct-07', 'Mar-07', 'Jun-09', 'Feb-43',
       'Jan-52', 'Oct-54', 'Sep-92', 'Mar-09', 'Nov-09', 'May-56',
       'Mar-74', 'May-77', 'Mar-80', 'Nov-88', 'May-91', 'Feb-93',
       'Mar-01', 'Oct-05', 'Sep-08', 'Apr-08', 'Apr-68', 'Dec-84',
       'Jun-01', 'Feb-08', 'Feb-40', 'Sep-78', 'Nov-79', 'Apr-82',
       'Apr-93', 'Aug-96', 'Jun-96', 'Jul-97', 'Mar-12', 'Mar-81',
       'May-04', 'May-11', 'Aug-12', 'Mar-62', 'May-82', 'Dec-85',
       'Dec-88', 'Aug-94', 'Jul-96', 'Aug-07', 'Mar-08', 'Oct-08',
       'Feb-42', 'Oct-43', 'May-60', 'Jan-65', 'May-78', 'Apr-87',
       'Oct-88', 'Jan-89', 'Apr-89', 'Dec-93', 'May-13', 'Jul-98',
       'Aug-11', 'Jul-70', 'Dec-82', 'Jan-91', 'Dec-00', 'Dec-12',
       'Apr-58', 'Jan-84', 'Sep-93', 'Aug-97', 'Aug-08', 'Sep-09',
       'Apr-41', 'Sep-81', 'Jul-83', 'Apr-91', 'Apr-92', 'Apr-96',
       'Jun-99', 'Mar-04', 'Jul-10', 'Aug-53', 'Jun-68', 'Dec-69',
       'May-73', 'May-81', 'Apr-88', 'Sep-90', 'Apr-99', 'Jan-01',
       'Feb-68', 'Mar-90', 'Jan-94', 'Nov-95', 'Dec-98', 'Jun-02',
       'Feb-61', 'Jun-82', 'Apr-86', 'May-87', 'Feb-88', 'May-95',
       'Sep-96', 'Jan-02', 'Mar-13', 'Jul-81', 'Jul-82', 'Jan-90',
       'Jun-94', 'Jan-99', 'Jun-06', 'Apr-07', 'Jul-11', 'Oct-11',
       'Apr-40', 'Apr-42', 'Jul-43', 'Oct-52', 'Sep-72', 'Feb-85',
       'Mar-96', 'Jul-06', 'Feb-07', 'May-08', 'Mar-11', 'Mar-40',
       'Jan-51', 'Jul-69', 'Oct-97', 'Mar-00', 'Apr-03', 'Dec-06',
       'Oct-12', 'Dec-43', 'Dec-71', 'Nov-73', 'May-94', 'Jul-00',
       'Aug-00', 'Sep-41', 'Mar-55', 'Jun-81', 'Nov-93', 'Feb-98',
       'Aug-99', 'Sep-07', 'Jan-10', 'Jun-10', 'May-51', 'Feb-51',
       'Feb-94', 'Jul-95', 'Aug-95', 'Feb-96', 'Feb-99', 'Aug-06',
       'Apr-09', 'May-10', 'Jun-42', 'Dec-81', 'Apr-98', 'Jun-98',
       'Jul-99', 'Nov-00', 'Feb-05', 'Dec-41', 'Dec-42', 'Sep-43',
       'Apr-54', 'Feb-55', 'Apr-65', 'May-70', 'Sep-97', 'May-99',
       'Sep-99', 'Jan-00', 'Dec-03', 'Mar-45', 'Apr-48', 'Mar-53',
       'Sep-55', 'Mar-78', 'Jul-88', 'Apr-94', 'Nov-07', 'Apr-12',
       'Nov-49', 'Apr-53', 'Nov-55', 'May-61', 'Jul-05', 'Nov-06',
       'Apr-11', 'Nov-11', 'Jun-12', 'Feb-12', 'Apr-43', 'Jun-47',
       'Mar-61', 'Apr-61', 'Mar-71', 'Oct-78', 'Oct-96', 'Mar-98',
       'Oct-99', 'Mar-02', 'Dec-09', 'Nov-41', 'Mar-47', 'May-53',
       'Jun-60', 'Sep-61', 'Jun-78', 'Aug-87', 'Nov-99', 'Feb-00',
       'May-03', 'Dec-05', 'Feb-11', 'Jun-40', 'May-42', 'Jan-42',
       'Nov-46', 'Jan-49', 'Jul-54', 'Jan-61', 'Jun-70', 'Dec-96',
       'Dec-97', 'Oct-00', 'Apr-04', 'Aug-09', 'Feb-10', 'Oct-40',
       'Dec-46', 'Mar-49', 'Jun-51', 'Feb-53', 'Sep-56', 'May-01',
       'Nov-02', 'Sep-12', 'Jul-12', 'Jul-42', 'May-48', 'Jun-50',
       'Jul-56', 'Dec-56', 'Dec-60', 'Jun-67', 'Oct-67', 'Nov-70',
       'Jul-07', 'Nov-12', 'Oct-49', 'Jul-50', 'Jan-53', 'Apr-60',
       'Aug-61', 'Feb-62', 'Apr-95', 'May-97', 'Jan-98', 'Apr-01',
       'May-12', 'Dec-39', 'Mar-42', 'Dec-44', 'Jan-47', 'Jun-53',
       'Nov-54', 'Jun-55', 'Mar-60', 'Jan-60', 'Apr-62', 'Dec-07',
       'Jun-13', 'May-40', 'May-41', 'Jul-51', 'Jul-57', 'Feb-59',
       'Sep-60', 'Dec-61', 'Jul-61', 'Jul-40', 'Feb-41', 'Aug-42',
       'Jan-43', 'Jul-44', 'Jun-45', 'Jul-45', 'Nov-48', 'Feb-52',
       'Dec-52', 'Aug-52', 'Apr-52', 'Jun-54', 'Feb-54', 'Jan-54',
       'Apr-55', 'Jun-57', 'Jun-58', 'Nov-59', 'Jul-59', 'Jul-60',
       'Dec-66', 'Sep-71', 'Apr-71', 'Jul-41', 'Jan-41', 'Sep-42',
       'Nov-42', 'Oct-42', 'Jun-43', 'May-43', 'Aug-43', 'Jun-44',
       'Sep-44', 'Oct-44', 'May-44', 'Apr-44', 'Feb-44', 'Jan-44',
       'Aug-44', 'Sep-45', 'Apr-45', 'Oct-45', 'Jan-45', 'Nov-45',
       'Feb-45', 'Feb-46', 'Mar-46', 'Jan-46', 'Aug-46', 'Jul-46',
       'May-46', 'Jun-46', 'Apr-46', 'Oct-47', 'Jul-47', 'Sep-47',
       'Dec-47', 'Jul-48', 'Feb-48', 'Jun-48', 'Sep-48', 'Jan-48',
       'Feb-49', 'Jul-49', 'May-49', 'Sep-49', 'May-50', 'Mar-50',
       'Feb-50', 'Apr-50', 'Sep-51', 'Apr-51', 'Aug-51', 'Dec-51',
       'Oct-51', 'Jul-52', 'Nov-53', 'Oct-55', 'Jan-55', 'Apr-57',
       'Feb-57', 'Sep-59', 'Jun-62', 'Mar-56'], dtype=object)
```

In [54]:
```
np.unique(cat_enc_le)
```

Out[54]:
```
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,
481, 482, 483, 484, 485, 486, 487, 488, 489, 490, 491, 492, 493,
494, 495, 496, 497, 498, 499, 500, 501, 502, 503, 504, 505, 506,
507, 508, 509, 510, 511, 512, 513, 514, 515, 516, 517, 518, 519,
520, 521, 522, 523, 524, 525, 526, 527, 528, 529, 530, 531, 532,
533, 534, 535, 536, 537, 538, 539, 540, 541, 542, 543, 544, 545,
546, 547, 548, 549, 550, 551, 552, 553, 554, 555, 556, 557, 558,
559, 560, 561, 562, 563, 564, 565, 566, 567, 568, 569, 570, 571,
572, 573, 574, 575, 576, 577, 578, 579, 580, 581, 582, 583, 584,
585, 586, 587, 588, 589, 590, 591, 592, 593, 594, 595, 596, 597,
598, 599, 600, 601, 602, 603, 604, 605, 606, 607, 608, 609, 610,
611, 612, 613, 614, 615, 616, 617, 618, 619, 620, 621, 622, 623,
624, 625, 626, 627, 628, 629, 630, 631, 632, 633, 634, 635, 636,
637, 638, 639, 640, 641, 642, 643, 644, 645, 646, 647, 648, 649,
650, 651, 652, 653, 654, 655, 656, 657, 658, 659, 660, 661, 662,
663, 664, 665, 666, 667, 668, 669, 670, 671, 672, 673, 674, 675,
676, 677, 678, 679, 680, 681, 682, 683, 684, 685, 686, 687, 688,
689, 690, 691, 692, 693, 694, 695, 696, 697, 698, 699, 700, 701,
702, 703, 704, 705, 706, 707, 708, 709, 710, 711, 712, 713, 714,
715, 716, 717, 718, 719, 720, 721, 722, 723, 724, 725, 726, 727,
728, 729, 730, 731, 732, 733, 734, 735, 736, 737, 738, 739, 740,
741, 742, 743, 744, 745, 746, 747, 748, 749, 750, 751, 752, 753,
754, 755, 756, 757, 758, 759, 760, 761, 762, 763, 764, 765, 766,
767, 768, 769, 770, 771, 772, 773, 774, 775, 776, 777, 778, 779,
780, 781, 782, 783, 784, 785, 786, 787, 788, 789, 790, 791, 792,
793, 794, 795, 796, 797, 798, 799, 800, 801, 802, 803, 804, 805,
806, 807, 808, 809, 810, 811, 812, 813, 814, 815, 816, 817, 818,
```

In [55]:
```
cat_enc_le
```

Out[55]:
```
array([ 99, 503, 738, ..., 480, 480, 480])
```

In [56]:
```
data_new['FIRST APPEARANCE'] = cat_enc_le
data_new
```

Out[56]:

	page_id	name	urlslug	SEX	ALIVE	APPE
0	1678	Spider-Man (Peter Parker)	\/Spider-Man_(Peter_Parker)	Male Characters	Living Characters	
1	7139	Captain America (Steven Rogers)	\/Captain_America_(Steven_Rogers)	Male Characters	Living Characters	
2	64786	Wolverine (James \"Logan\" Howlett)	\/Wolverine_(James_%22Logan%22_Howlett)	Male Characters	Living Characters	
3	1868	Iron Man (Anthony \"Tony\" Stark)	\/Iron_Man_(Anthony_%22Tony%22_Stark)	Male Characters	Living Characters	
4	2460	Thor (Thor Odinson)	\/Thor_(Thor_Odinson)	Male Characters	Living Characters	
...	...	...	...	...	...	
16371	657508	Ru'ach (Earth-616)	\/Ru%27ach_(Earth-616)	Male Characters	Living Characters	
16372	665474	Thane (Thanos' son) (Earth-616)	\/Thane_(Thanos%27_son)_(Earth-616)	Male Characters	Living Characters	
16373	695217	Tinkerer (Skrull) (Earth-616)	\/Tinkerer_(Skrull)_(Earth-616)	Male Characters	Living Characters	
16374	708811	TK421 (Spiderling) (Earth-616)	\/TK421_(Spiderling)_(Earth-616)	Male Characters	Living Characters	
16375	673702	Yologarch (Earth-616)	\/Yologarch_(Earth-616)	Male Characters	Living Characters	

16376 rows × 8 columns

In [57]:
```
cat_enc = pd.DataFrame({'c1':data_imp1.T[0]})
cat_enc
```

Out[57]:

	c1
0	Male Characters
1	Male Characters
2	Male Characters
3	Male Characters
4	Male Characters
...	...
16371	Male Characters

	c1
16372	Male Characters
16373	Male Characters
16374	Male Characters
16375	Male Characters

In [58]:
```
le = LabelEncoder()
cat_enc_le = le.fit_transform(cat_enc['c1'])
```

In [59]:
```
cat_enc['c1'].unique()
```

Out[59]:
```
array(['Male Characters', 'Female Characters', 'Genderfluid Characters',
       'Agender Characters'], dtype=object)
```

In [60]:
```
np.unique(cat_enc_le)
```

Out[60]:
```
array([0, 1, 2, 3])
```

In [61]:
```
data_new['SEX'] = cat_enc_le
data_new
```

Out[61]:

	page_id	name	urlslug	SEX	ALIVE	APPEARAN
0	1678	Spider-Man (Peter Parker)	\/Spider-Man_(Peter_Parker)	3	Living Characters	40
1	7139	Captain America (Steven Rogers)	\/Captain_America_(Steven_Rogers)	3	Living Characters	33
2	64786	Wolverine (James \"Logan\" Howlett)	\/Wolverine_(James_%22Logan%22_Howlett)	3	Living Characters	30
3	1868	Iron Man (Anthony \"Tony\" Stark)	\/Iron_Man_(Anthony_%22Tony%22_Stark)	3	Living Characters	29
4	2460	Thor (Thor Odinson)	\/Thor_(Thor_Odinson)	3	Living Characters	22
...	...	...	...	...	...	
16371	657508	Ru'ach (Earth-616)	\/Ru%27ach_(Earth-616)	3	Living Characters	
16372	665474	Thane (Thanos' son) (Earth-616)	\/Thane_(Thanos%27_son)_(Earth-616)	3	Living Characters	
16373	695217	Tinkerer (Skrull) (Earth-616)	\/Tinkerer_(Skrull)_(Earth-616)	3	Living Characters	

	page_id	name	urlslug	SEX	ALIVE	APPEARAN
16374	708811	TK421 (Spiderling) (Earth-616)	\/TK421_(Spiderling)_(Earth-616)	3	Living Characters	
16375	673702	Yologarch	\/Yologarch_(Earth-616)	3	Living	

In [62]:
```
cat_enc = pd.DataFrame({'c1':data_imp2.T[0]})
cat_enc
```

Out[62]:

	c1
0	Living Characters
1	Living Characters
2	Living Characters
3	Living Characters
4	Living Characters
...	...
16371	Living Characters
16372	Living Characters
16373	Living Characters
16374	Living Characters
16375	Living Characters

16376 rows × 1 columns

In [63]:
```
le = LabelEncoder()
cat_enc_le = le.fit_transform(cat_enc['c1'])
```

In [64]:
```
cat_enc['c1'].unique()
```

Out[64]:
```
array(['Living Characters', 'Deceased Characters'], dtype=object)
```

In [65]:
```
np.unique(cat_enc_le)
```

Out[65]:
```
array([0, 1])
```

In [66]:
```
data_new['ALIVE'] = cat_enc_le
data_new
```

Out[66]:

	page_id	name	urlslug	SEX	ALIVE	APPEARANCES
0	1678	Spider-Man (Peter Parker)	\/Spider-Man_(Peter_Parker)	3	1	4043.0
1	7139	Captain America (Steven Rogers)	\/Captain_America_(Steven_Rogers)	3	1	3360.0

	page_id	name	urlslug	SEX	ALIVE	APPEARANCES
2	64786	Wolverine (James \"Logan\" Howlett)	\/Wolverine_(James_%22Logan%22_Howlett)	3	1	3061.0
3	1868	Iron Man (Anthony \"Tony\" Stark)	\/Iron_Man_(Anthony_%22Tony%22_Stark)	3	1	2961.0
4	2460	Thor (Thor Odinson)	\/Thor_(Thor_Odinson)	3	1	2258.0
...	...	...	...	...	...	...
16371	657508	Ru'ach (Earth-616)	\/Ru%27ach_(Earth-616)	3	1	1.0
16372	665474	Thane (Thanos' son) (Earth-616)	\/Thane_(Thanos%27_son)_(Earth-616)	3	1	1.0
16373	695217	Tinkerer (Skrull) (Earth-616)	\/Tinkerer_(Skrull)_(Earth-616)	3	1	1.0
16374	708811	TK421 (Spiderling) (Earth-616)	\/TK421_(Spiderling)_(Earth-616)	3	1	1.0
16375	673702	Yologarch (Earth-616)	\/Yologarch_(Earth-616)	3	1	1.0

In [67]:
```
cat_temp_data = data[['urlslug']]
cat_temp_data.head()
```

Out[67]:

	urlslug
0	\/Spider-Man_(Peter_Parker)
1	\/Captain_America_(Steven_Rogers)
2	\/Wolverine_(James_%22Logan%22_Howlett)
3	\/Iron_Man_(Anthony_%22Tony%22_Stark)
4	\/Thor_(Thor_Odinson)

In [68]:
```
# Импьютация наиболее частыми значениями
imp4 = SimpleImputer(missing_values=np.nan, strategy='most_frequent')
data_imp4 = imp4.fit_transform(cat_temp_data)
data_imp4
```

Out[68]:
```
array([['\\/Spider-Man_(Peter_Parker)'],
       ['\\/Captain_America_(Steven_Rogers)'],
       ['\\/Wolverine_(James_%22Logan%22_Howlett)'],
       ...,
       ['\\/Tinkerer_(Skrull)_(Earth-616)'],
       ['\\/TK421_(Spiderling)_(Earth-616)'],
       ['\\/Yologarch_(Earth-616)']], dtype=object)
```

In [69]:
```
cat_enc = pd.DataFrame({'c1':data_imp4.T[0]})
cat_enc
```

Out[69]:

	c1
0	\/Spider-Man_(Peter_Parker)
1	\/Captain_America_(Steven_Rogers)
2	\/Wolverine_(James_%22Logan%22_Howlett)
3	\/Iron_Man_(Anthony_%22Tony%22_Stark)
4	\/Thor_(Thor_Odinson)
...	...
16371	\/Ru%27ach_(Earth-616)
16372	\/Thane_(Thanos%27_son)_(Earth-616)
16373	\/Tinkerer_(Skrull)_(Earth-616)
16374	\/TK421_(Spiderling)_(Earth-616)
16375	\/Yologarch_(Earth-616)

16376 rows × 1 columns

In [70]:
```
le = LabelEncoder()
cat_enc_le = le.fit_transform(cat_enc['c1'])
```

In [71]:
```
cat_enc['c1'].unique()
```

Out[71]:
```
array(['\\/Spider-Man_(Peter_Parker)',
       '\\/Captain_America_(Steven_Rogers)',
       '\\/Wolverine_(James_%22Logan%22_Howlett)', ...,
       '\\/Tinkerer_(Skrull)_(Earth-616)',
       '\\/TK421_(Spiderling)_(Earth-616)', '\\/Yologarch_(Earth-616)'],
      dtype=object)
```

In [72]:
```
np.unique(cat_enc_le)
```

Out[72]:
```
array([    0,     1,     2, ..., 16373, 16374, 16375])
```

In [73]:
```
data_new['urlslug'] = cat_enc_le
data_new
```

Out[73]:

	page_id	name	urlslug	SEX	ALIVE	APPEARANCES	FIRST APPEARANCE	Year
0	1678	Spider-Man (Peter Parker)	13953	3	1	4043.0	99	1962.0
1	7139	Captain America (Steven Rogers)	2330	3	1	3360.0	503	1941.0
2	64786	Wolverine (James \"Logan\" Howlett)	15999	3	1	3061.0	738	1974.0

	page_id	name	urlslug	SEX	ALIVE	APPEARANCES	FIRST APPEARANCE	Year
3	1868	Iron Man (Anthony \"Tony\" Stark)	6774	3	1	2961.0	521	1963.0
4	2460	Thor (Thor Odinson)	14709	3	1	2258.0	649	1950.0
...	...	...	...	...	...	...	...	...
16371	657508	Ru'ach (Earth-616)	12831	3	1	1.0	480	1990.0
16372	665474	Thane (Thanos' son) (Earth-616)	14585	3	1	1.0	480	1990.0
16373	695217	Tinkerer (Skrull) (Earth-616)	14816	3	1	1.0	480	1990.0
16374	708811	TK421 (Spiderling) (Earth-616)	14335	3	1	1.0	480	1990.0

In [74]:
```
cat_temp_data = data[['name']]
cat_temp_data.head()
```

Out[74]:

	name
0	Spider-Man (Peter Parker)
1	Captain America (Steven Rogers)
2	Wolverine (James \"Logan\" Howlett)
3	Iron Man (Anthony \"Tony\" Stark)
4	Thor (Thor Odinson)

In [75]:
```
# Импьютация наиболее частыми значениями
imp4 = SimpleImputer(missing_values=np.nan, strategy='most_frequent')
data_imp4 = imp4.fit_transform(cat_temp_data)
data_imp4
```

Out[75]:
```
array([['Spider-Man (Peter Parker)'],
       ['Captain America (Steven Rogers)'],
       ['Wolverine (James \\"Logan\\" Howlett)'],
       ...,
       ['Tinkerer (Skrull) (Earth-616)'],
       ['TK421 (Spiderling) (Earth-616)'],
       ['Yologarch (Earth-616)']], dtype=object)
```

In [76]:
```
cat_enc = pd.DataFrame({'c1':data_imp4.T[0]})
cat_enc
```

Out[76]:

	c1
0	Spider-Man (Peter Parker)
1	Captain America (Steven Rogers)
2	Wolverine (James \"Logan\" Howlett)
3	Iron Man (Anthony \"Tony\" Stark)
4	Thor (Thor Odinson)

	c1
...	...
16371	Ru'ach (Earth-616)
16372	Thane (Thanos' son) (Earth-616)
16373	Tinkerer (Skrull) (Earth-616)
16374	TK421 (Spiderling) (Earth-616)
16375	Yologarch (Earth-616)

In [77]:
```
le = LabelEncoder()
cat_enc_le = le.fit_transform(cat_enc['c1'])
```

In [78]:
```
cat_enc['c1'].unique()
```

Out[78]:
```
array(['Spider-Man (Peter Parker)', 'Captain America (Steven Rogers)',
       'Wolverine (James \\"Logan\\" Howlett)', ...,
       'Tinkerer (Skrull) (Earth-616)', 'TK421 (Spiderling) (Earth-616)',
       'Yologarch (Earth-616)'], dtype=object)
```

In [79]:
```
np.unique(cat_enc_le)
```

Out[79]:
```
array([    0,     1,     2, ..., 16373, 16374, 16375])
```

In [80]:
```
data_new['name'] = cat_enc_le
data_new
```

Out[80]:

	page_id	name	urlslug	SEX	ALIVE	APPEARANCES	FIRST APPEARANCE	Year
0	1678	13954	13953	3	1	4043.0	99	1962.0
1	7139	2327	2330	3	1	3360.0	503	1941.0
2	64786	15996	15999	3	1	3061.0	738	1974.0
3	1868	6771	6774	3	1	2961.0	521	1963.0
4	2460	14709	14709	3	1	2258.0	649	1950.0
...	...	...	...	...	...	...	...	...
16371	657508	12829	12831	3	1	1.0	480	1990.0
16372	665474	14585	14585	3	1	1.0	480	1990.0
16373	695217	14816	14816	3	1	1.0	480	1990.0
16374	708811	14335	14335	3	1	1.0	480	1990.0
16375	673702	16147	16150	3	1	1.0	480	1990.0

16376 rows × 8 columns

Все столбцы закодированы

Перейдём к масштабированию параметров

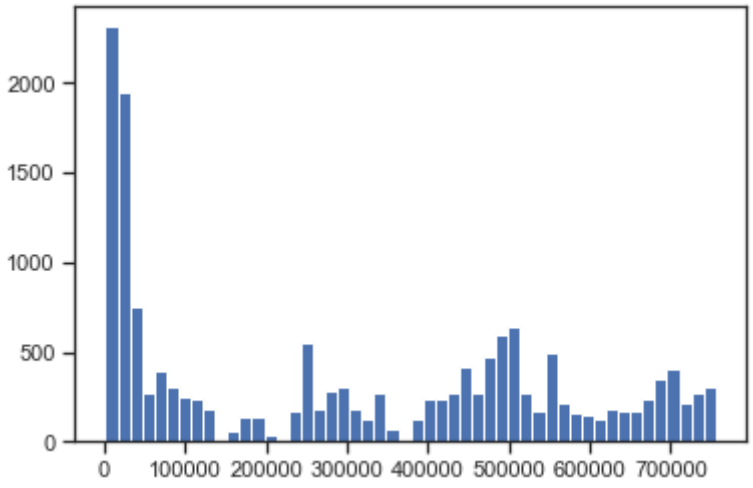
In [81]:
```
from sklearn.preprocessing import MinMaxScaler, StandardScaler, Normalizer
```

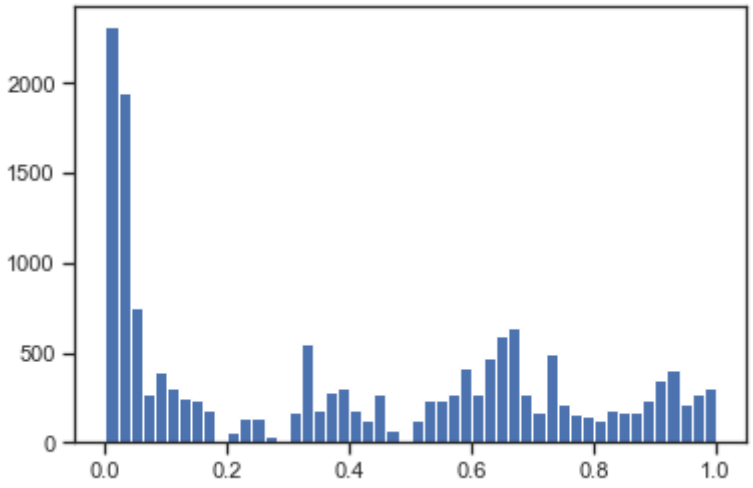
In [82]:
```
sc1 = MinMaxScaler()
sc1_data = sc1.fit_transform(data_new[['page_id']])

plt.hist(data_new['page_id'], 50)
plt.show()

plt.hist(sc1_data, 50)
plt.show()

data_new['page_id'] = sc1_data
data_new
```





Out[82]:

	page_id	name	urlslug	SEX	ALIVE	APPEARANCES	FIRST APPEARANCE	Year
0	0.000866	13954	13953	3	1	4043.0	99	1962.0
1	0.008106	2327	2330	3	1	3360.0	503	1941.0
2	0.084535	15996	15999	3	1	3061.0	738	1974.0
3	0.001118	6771	6774	3	1	2961.0	521	1963.0
4	0.001903	14709	14709	3	1	2258.0	649	1950.0
...	...	...	...	...	...	...	...	...
16371	0.870375	12829	12831	3	1	1.0	480	1990.0
16372	0.880937	14585	14585	3	1	1.0	480	1990.0

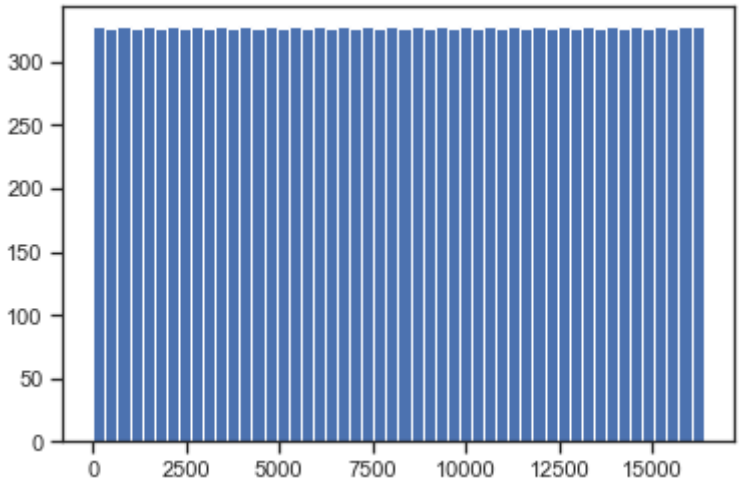
	page_id	name	urlslug	SEX	ALIVE	APPEARANCES	FIRST APPEARANCE	Year
16373	0.920370	14816	14816	3	1	1.0	480	1990.0
16374	0.938393	14335	14335	3	1	1.0	480	1990.0
16375	0.891845	16147	16150	3	1	1.0	480	1990.0

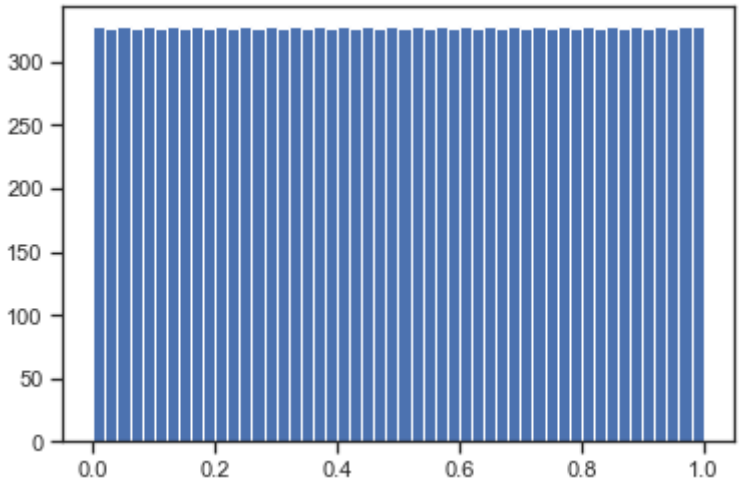
In [83]:
```
sc1 = MinMaxScaler()
sc1_data = sc1.fit_transform(data_new[['name']])

plt.hist(data_new['name'], 50)
plt.show()

plt.hist(sc1_data, 50)
plt.show()

data_new['name'] = sc1_data
data_new
```





Out[83]:

	page_id	name	urlslug	SEX	ALIVE	APPEARANCES	FIRST APPEARANCE	Year
0	0.000866	0.852153	13953	3	1	4043.0	99	1962.0
1	0.008106	0.142107	2330	3	1	3360.0	503	1941.0
2	0.084535	0.976855	15999	3	1	3061.0	738	1974.0
3	0.001118	0.413496	6774	3	1	2961.0	521	1963.0
4	0.001903	0.898260	14709	3	1	2258.0	649	1950.0
...	...	...	...	...	...	...	...	...

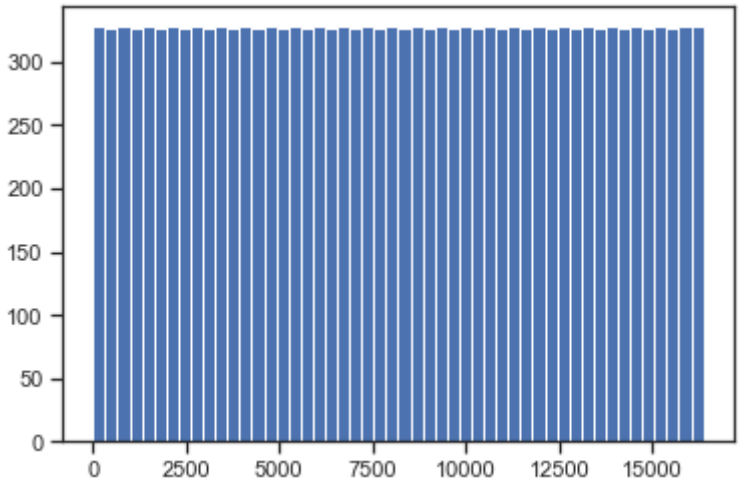
	page_id	name	urlslug	SEX	ALIVE	APPEARANCES	FIRST APPEARANCE	Year
16371	0.870375	0.783450	12831	3	1	1.0	480	1990.0
16372	0.880937	0.890687	14585	3	1	1.0	480	1990.0
16373	0.920370	0.904794	14816	3	1	1.0	480	1990.0
16374	0.938393	0.875420	14335	3	1	1.0	480	1990.0
16375	0.891845	0.986076	16150	3	1	1.0	480	1990.0

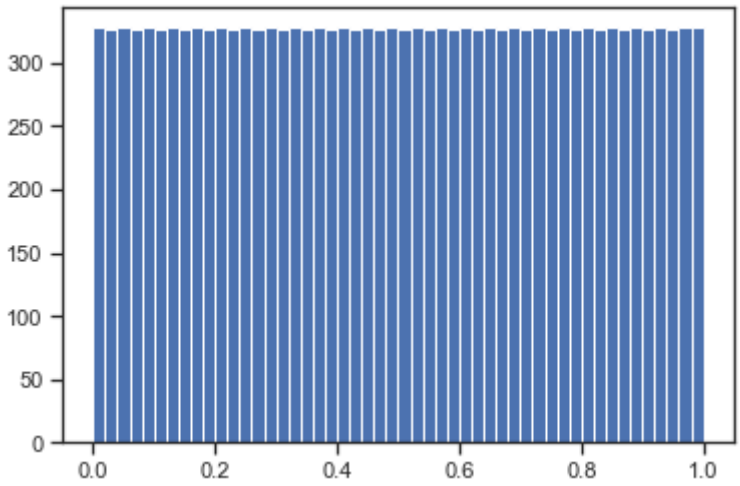
In [84]:
```
sc1 = MinMaxScaler()
sc1_data = sc1.fit_transform(data_new[['urlslug']])

plt.hist(data_new['urlslug'], 50)
plt.show()

plt.hist(sc1_data, 50)
plt.show()

data_new['urlslug'] = sc1_data
data_new
```





Out[84]:

	page_id	name	urlslug	SEX	ALIVE	APPEARANCES	FIRST APPEARANCE	Year
0	0.000866	0.852153	0.852092	3	1	4043.0	99	1962.0
1	0.008106	0.142107	0.142290	3	1	3360.0	503	1941.0
2	0.084535	0.976855	0.977038	3	1	3061.0	738	1974.0

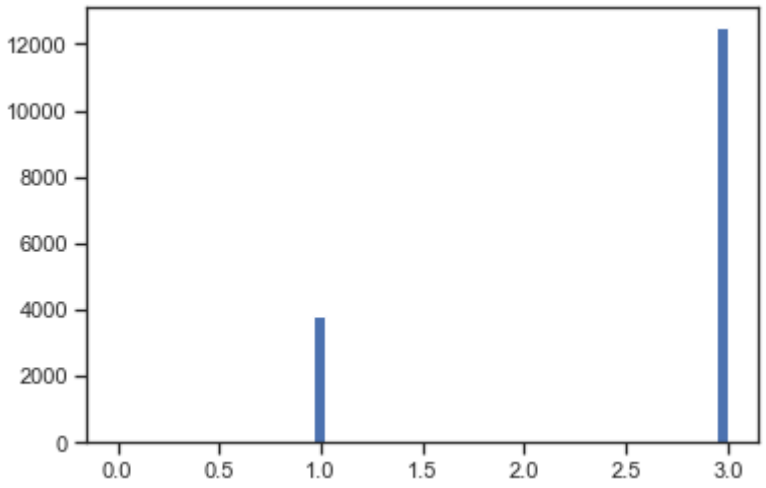
	page_id	name	urlslug	SEX	ALIVE	APPEARANCES	FIRST APPEARANCE	Year
3	0.001118	0.413496	0.413679	3	1	2961.0	521	1963.0
4	0.001903	0.898260	0.898260	3	1	2258.0	649	1950.0
...	...	...	...	...	...	...	...	...
16371	0.870375	0.783450	0.783573	3	1	1.0	480	1990.0
16372	0.880937	0.890687	0.890687	3	1	1.0	480	1990.0
16373	0.920370	0.904794	0.904794	3	1	1.0	480	1990.0
16374	0.938393	0.875420	0.875420	3	1	1.0	480	1990.0
16375	0.891845	0.986076	0.986260	3	1	1.0	480	1990.0

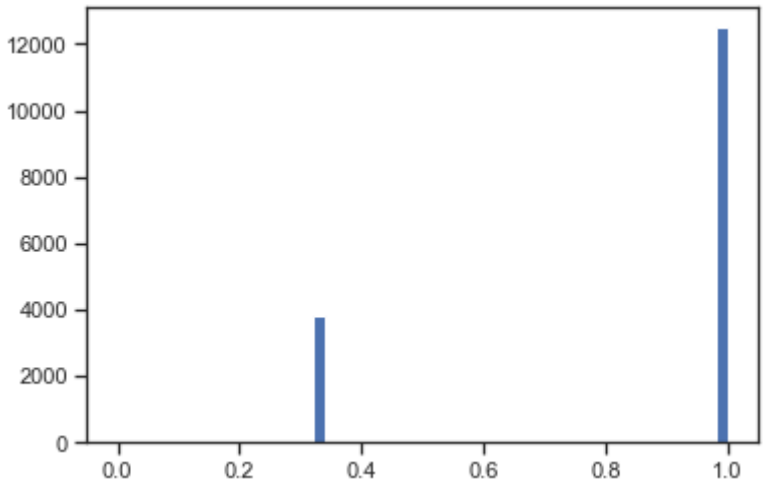
In [85]:
```
sc1 = MinMaxScaler()
sc1_data = sc1.fit_transform(data_new[['SEX']])

plt.hist(data_new['SEX'], 50)
plt.show()

plt.hist(sc1_data, 50)
plt.show()

data_new['SEX'] = sc1_data
data_new
```





Out[85]:

	page_id	name	urlslug	SEX	ALIVE	APPEARANCES	FIRST APPEARANCE	Year
0	0.000866	0.852153	0.852092	1.0	1	4043.0	99	1962.0

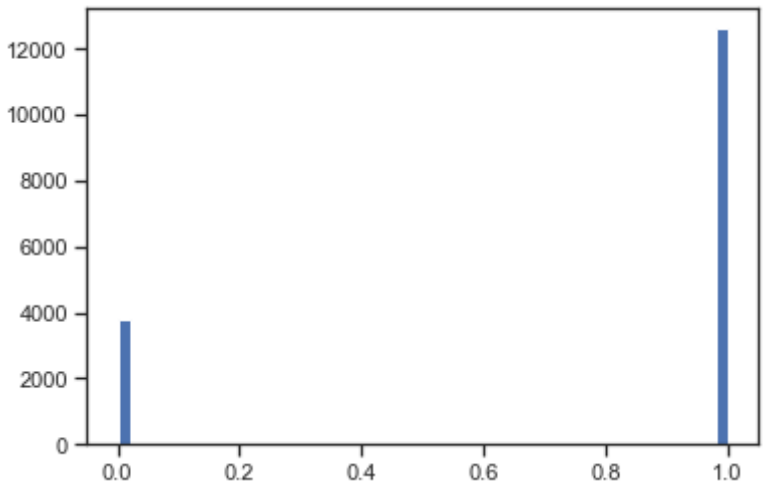
	page_id	name	urlslug	SEX	ALIVE	APPEARANCES	FIRST APPEARANCE	Year
1	0.008106	0.142107	0.142290	1.0	1	3360.0	503	1941.0
2	0.084535	0.976855	0.977038	1.0	1	3061.0	738	1974.0
3	0.001118	0.413496	0.413679	1.0	1	2961.0	521	1963.0
4	0.001903	0.898260	0.898260	1.0	1	2258.0	649	1950.0
...	...	...	...	...	...	...	...	...
16371	0.870375	0.783450	0.783573	1.0	1	1.0	480	1990.0
16372	0.880937	0.890687	0.890687	1.0	1	1.0	480	1990.0
16373	0.920370	0.904794	0.904794	1.0	1	1.0	480	1990.0
16374	0.938393	0.875420	0.875420	1.0	1	1.0	480	1990.0
16375	0.891845	0.986076	0.986260	1.0	1	1.0	480	1990.0

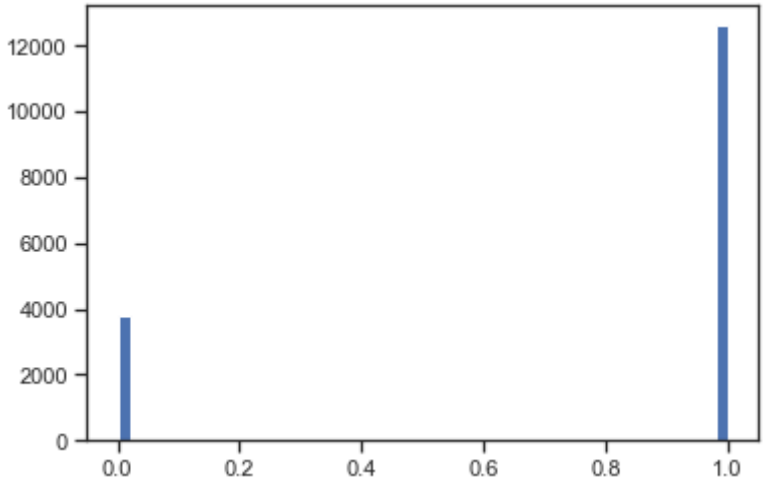
In [86]:
```
sc1 = MinMaxScaler()
sc1_data = sc1.fit_transform(data_new[['ALIVE']])

plt.hist(data_new['ALIVE'], 50)
plt.show()

plt.hist(sc1_data, 50)
plt.show()

data_new['ALIVE'] = sc1_data
data_new
```





Out[86]:

	page_id	name	urlslug	SEX	ALIVE	APPEARANCES	FIRST APPEARANCE	Year
0	0.000866	0.852153	0.852092	1.0	1.0	4043.0	99	1962.0
1	0.008106	0.142107	0.142290	1.0	1.0	3360.0	503	1941.0
2	0.084535	0.976855	0.977038	1.0	1.0	3061.0	738	1974.0
3	0.001118	0.413496	0.413679	1.0	1.0	2961.0	521	1963.0

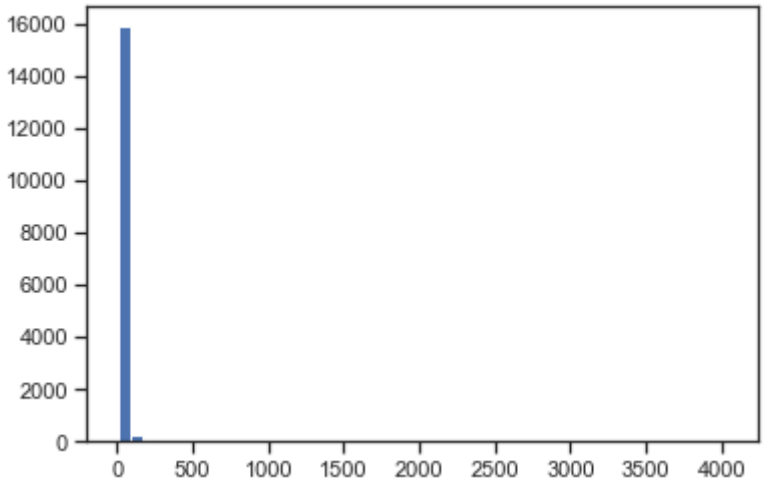
In [87]:

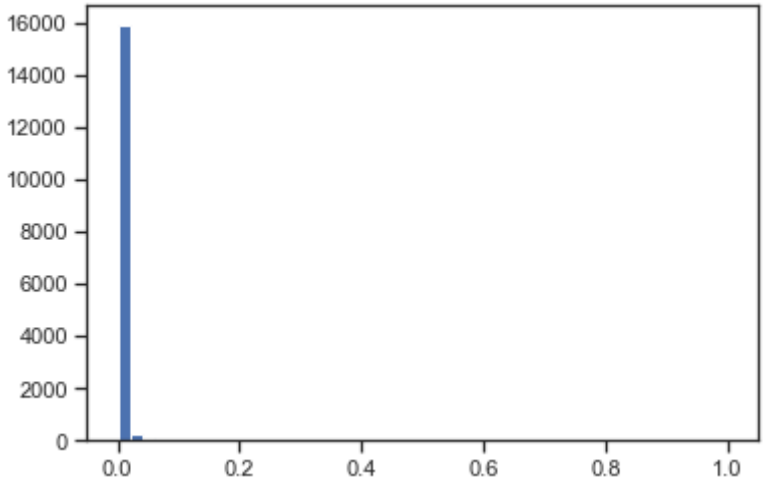
```
sc1 = MinMaxScaler()
sc1_data = sc1.fit_transform(data_new[['APPEARANCES']])

plt.hist(data_new['APPEARANCES'], 50)
plt.show()

plt.hist(sc1_data, 50)
plt.show()

data_new['APPEARANCES'] = sc1_data
data_new
```





Out[87]:

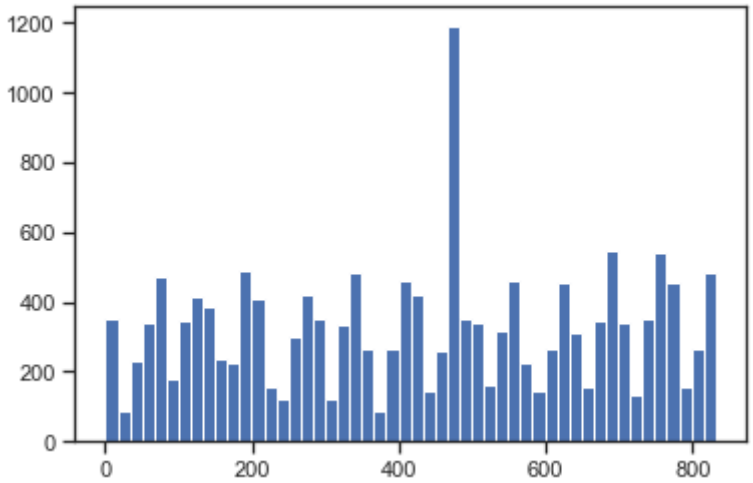
	page_id	name	urlslug	SEX	ALIVE	APPEARANCES	FIRST APPEARANCE	Year
0	0.000866	0.852153	0.852092	1.0	1.0	1.000000	99	1962.0
1	0.008106	0.142107	0.142290	1.0	1.0	0.831024	503	1941.0
2	0.084535	0.976855	0.977038	1.0	1.0	0.757051	738	1974.0
3	0.001118	0.413496	0.413679	1.0	1.0	0.732311	521	1963.0
4	0.001903	0.898260	0.898260	1.0	1.0	0.558387	649	1950.0
...	...	...	...	...	...	...	...	...
16371	0.870375	0.783450	0.783572	1.0	1.0	0.000000	480	1990.0

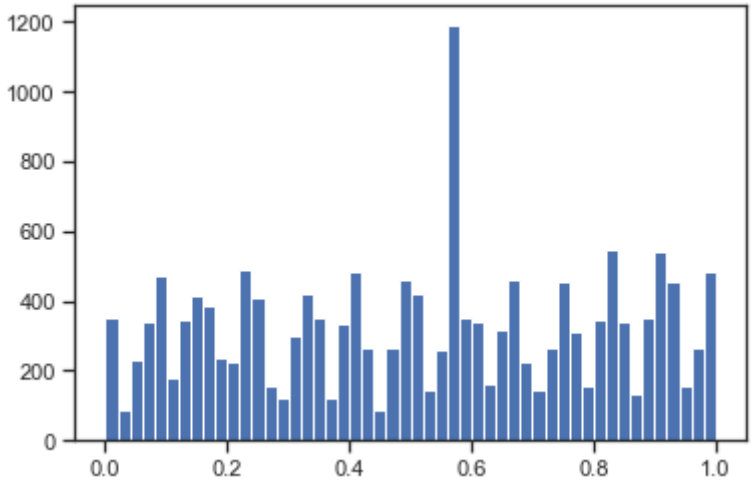
In [88]:
```
sc1 = MinMaxScaler()
sc1_data = sc1.fit_transform(data_new[['FIRST APPEARANCE']])

plt.hist(data_new['FIRST APPEARANCE'], 50)
plt.show()

plt.hist(sc1_data, 50)
plt.show()

data_new['FIRST APPEARANCE'] = sc1_data
data_new
```





Out[88]:

	page_id	name	urlslug	SEX	ALIVE	APPEARANCES	FIRST APPEARANCE	Year
0	0.000866	0.852153	0.852092	1.0	1.0	1.000000	0.119134	1962.0
1	0.008106	0.142107	0.142290	1.0	1.0	0.831024	0.605295	1941.0
2	0.084535	0.976855	0.977038	1.0	1.0	0.757051	0.888087	1974.0
3	0.001118	0.413496	0.413679	1.0	1.0	0.732311	0.626955	1963.0
4	0.001903	0.898260	0.898260	1.0	1.0	0.558387	0.780987	1950.0
...	...	...	...	...	...	...	...	...
16371	0.870375	0.783450	0.783573	1.0	1.0	0.000000	0.577617	1990.0
16372	0.880937	0.890687	0.890687	1.0	1.0	0.000000	0.577617	1990.0
16373	0.920370	0.904794	0.904794	1.0	1.0	0.000000	0.577617	1990.0

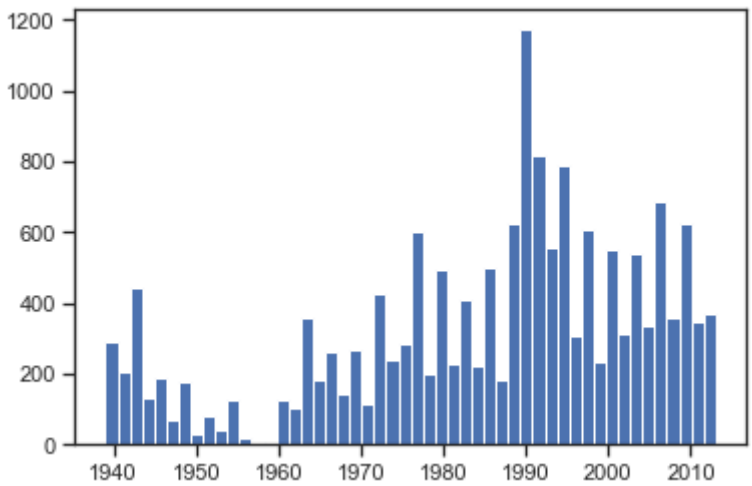
In [89]:

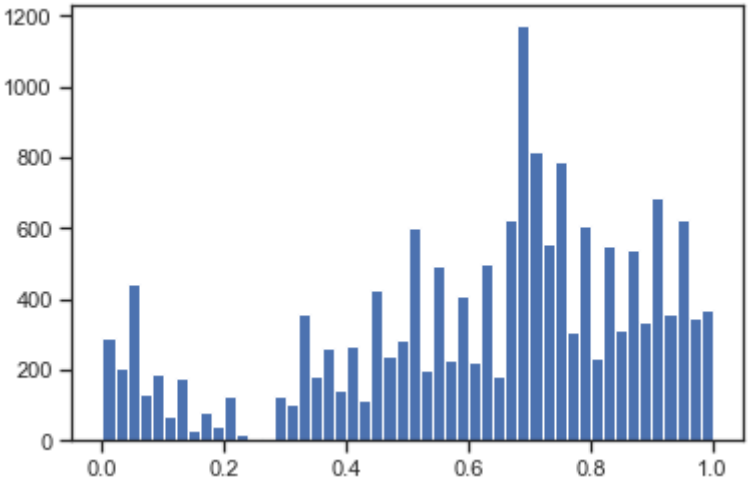
```
sc1 = MinMaxScaler()
sc1_data = sc1.fit_transform(data_new[['Year']])

plt.hist(data_new['Year'], 50)
plt.show()

plt.hist(sc1_data, 50)
plt.show()

data_new['Year'] = sc1_data
data_new
```





Out[89]:

	page_id	name	urlslug	SEX	ALIVE	APPEARANCES	FIRST APPEARANCE	Year
0	0.000866	0.852153	0.852092	1.0	1.0	1.000000	0.119134	0.310811
1	0.008106	0.142107	0.142290	1.0	1.0	0.831024	0.605295	0.027027
2	0.084535	0.976855	0.977038	1.0	1.0	0.757051	0.888087	0.472973
3	0.001118	0.413496	0.413679	1.0	1.0	0.732311	0.626955	0.324324
4	0.001903	0.898260	0.898260	1.0	1.0	0.558387	0.780987	0.148649
...	...	...	...	...	...	...	...	...
16371	0.870375	0.783450	0.783573	1.0	1.0	0.000000	0.577617	0.689189
16372	0.880937	0.890687	0.890687	1.0	1.0	0.000000	0.577617	0.689189
16373	0.920370	0.904794	0.904794	1.0	1.0	0.000000	0.577617	0.689189
16374	0.938393	0.875420	0.875420	1.0	1.0	0.000000	0.577617	0.689189
16375	0.891845	0.986076	0.986260	1.0	1.0	0.000000	0.577617	0.689189