# Обработка пропусков в данных, кодирование категориальных признаков, масштабирование данных

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

Используем данные из соревнования Top 100 popular movies from 2003 to 2022 (iMDB) <https://www.kaggle.com/datasets/georgescutelnicu/top-100-popular-movies-from-2003-to-2022-imdb>

data = pd.read\_csv('data/movies.csv', sep=";")

# размер набора данных  
data.shape

(2000, 11)

# типы колонок  
data.dtypes

Title object  
Rating float64  
Year int64  
Month object  
Certificate object  
Runtime float64  
Directors object  
Stars object  
Genre object  
Filming\_location object  
Country\_of\_origin object  
dtype: object

# проверка на пропущенные значения  
data.isnull().sum()

Title 0  
Rating 1  
Year 0  
Month 0  
Certificate 34  
Runtime 2  
Directors 0  
Stars 0  
Genre 0  
Filming\_location 0  
Country\_of\_origin 0  
dtype: int64

# Первые 5 строк датасета  
data.head()

Title Rating Year Month Certificate \  
0 Avatar: The Way of Water 7.8 2022 December PG-13   
1 Guillermo del Toro's Pinocchio 7.6 2022 December PG   
2 Bullet Train 7.3 2022 August R   
3 The Banshees of Inisherin 7.8 2022 November R   
4 M3gan 6.4 2022 December PG-13   
  
 Runtime Directors \  
0 192.0 James Cameron   
1 117.0 Guillermo del Toro, Mark Gustafson   
2 127.0 David Leitch   
3 114.0 Martin McDonagh   
4 102.0 Gerard Johnstone   
  
 Stars \  
0 Sam Worthington, Zoe Saldana, Sigourney Weaver...   
1 Ewan McGregor, David Bradley, Gregory Mann, Bu...   
2 Brad Pitt, Joey King, Aaron Taylor Johnson, Br...   
3 Colin Farrell, Brendan Gleeson, Kerry Condon, ...   
4 Jenna Davis, Amie Donald, Allison Williams, Vi...   
  
 Genre Filming\_location \  
0 Action, Adventure, Fantasy New Zealand   
1 Animation, Drama, Family USA   
2 Action, Comedy, Thriller Japan   
3 Comedy, Drama Ireland   
4 Horror, Sci-Fi, Thriller New Zealand   
  
 Country\_of\_origin   
0 United States   
1 United States, Mexico, France   
2 Japan, United States   
3 Ireland, United Kingdom, United States   
4 United States

total\_count = data.shape[0]  
print('Всего строк: {}'.format(total\_count))

Всего строк: 2000

## Обработка пропусков в данных

### Обработка пропусков в числовых данных

# Удаление колонок, содержащих пустые значения  
data\_new\_1 = data.dropna(axis=1, how='any')  
(data.shape, data\_new\_1.shape)

((2000, 11), (2000, 8))

# Удаление строк, содержащих пустые значения  
data\_new\_2 = data.dropna(axis=0, how='any')  
(data.shape, data\_new\_2.shape)

((2000, 11), (1965, 11))

# Заполнение всех пропущенных значений нулями  
# В данном случае это некорректно, так как нулями заполняются в том числе категориальные колонки  
data\_new\_3 = data.fillna(0)  
data\_new\_3.head()

Title Rating Year Month Certificate \  
0 Avatar: The Way of Water 7.8 2022 December PG-13   
1 Guillermo del Toro's Pinocchio 7.6 2022 December PG   
2 Bullet Train 7.3 2022 August R   
3 The Banshees of Inisherin 7.8 2022 November R   
4 M3gan 6.4 2022 December PG-13   
  
 Runtime Directors \  
0 192.0 James Cameron   
1 117.0 Guillermo del Toro, Mark Gustafson   
2 127.0 David Leitch   
3 114.0 Martin McDonagh   
4 102.0 Gerard Johnstone   
  
 Stars \  
0 Sam Worthington, Zoe Saldana, Sigourney Weaver...   
1 Ewan McGregor, David Bradley, Gregory Mann, Bu...   
2 Brad Pitt, Joey King, Aaron Taylor Johnson, Br...   
3 Colin Farrell, Brendan Gleeson, Kerry Condon, ...   
4 Jenna Davis, Amie Donald, Allison Williams, Vi...   
  
 Genre Filming\_location \  
0 Action, Adventure, Fantasy New Zealand   
1 Animation, Drama, Family USA   
2 Action, Comedy, Thriller Japan   
3 Comedy, Drama Ireland   
4 Horror, Sci-Fi, Thriller New Zealand   
  
 Country\_of\_origin   
0 United States   
1 United States, Mexico, France   
2 Japan, United States   
3 Ireland, United Kingdom, United States   
4 United States

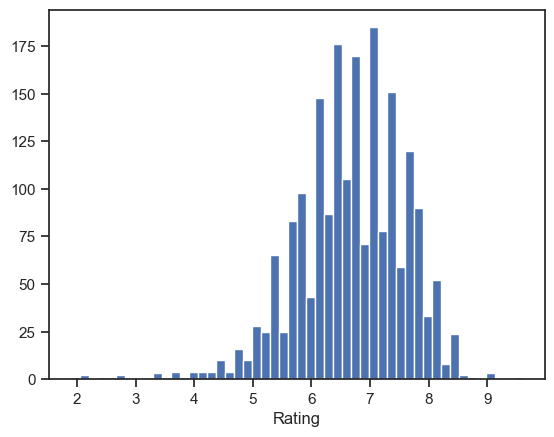
# Выберем числовые колонки с пропущенными значениями  
# Цикл по колонкам датасета  
num\_cols = []  
for col in data.columns:  
 # Количество пустых значений   
 temp\_null\_count = data[data[col].isnull()].shape[0]  
 dt = str(data[col].dtype)  
 if temp\_null\_count>0 and (dt=='float64' or dt=='int64'):  
 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))

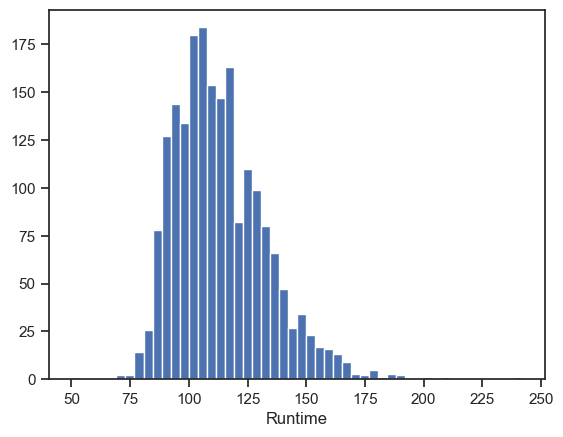
Колонка Rating. Тип данных float64. Количество пустых значений 1, 0.05%.  
Колонка Runtime. Тип данных float64. Количество пустых значений 2, 0.1%.

# Фильтр по колонкам с пропущенными значениями  
data\_num = data[num\_cols]  
data\_num

Rating Runtime  
0 7.8 192.0  
1 7.6 117.0  
2 7.3 127.0  
3 7.8 114.0  
4 6.4 102.0  
... ... ...  
1995 7.1 114.0  
1996 5.5 117.0  
1997 5.8 98.0  
1998 7.1 105.0  
1999 5.8 79.0  
  
[2000 rows x 2 columns]

# Гистограмма по признакам  
for col in data\_num:  
 plt.hist(data[col], 50)  
 plt.xlabel(col)  
 plt.show()





Будем использовать встроенные средства импьютации библиотеки scikit-learn - <https://scikit-learn.org/stable/modules/impute.html>

Пропуски данных в колонке Rating заполним на медианное значение, в колонке Runtime - на наиболее повторяющееся

data\_num\_Rating = data\_num[['Rating']]  
data\_num\_Rating.head()

Rating  
0 7.8  
1 7.6  
2 7.3  
3 7.8  
4 6.4

from sklearn.impute import SimpleImputer  
from sklearn.impute import MissingIndicator

# Фильтр для проверки заполнения пустых значений  
indicator = MissingIndicator()  
mask\_missing\_Rating\_only = indicator.fit\_transform(data\_num\_Rating)  
mask\_missing\_Rating\_only

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

imp\_num = SimpleImputer(strategy='median')  
data\_num\_imp\_Rating = imp\_num.fit\_transform(data\_num\_Rating)  
data\_num\_imp\_Rating[mask\_missing\_Rating\_only]

array([6.7])

np.unique(data\_num\_imp\_Rating)

array([1.9, 2.1, 2.2, 2.4, 2.6, 2.8, 3.3, 3.4, 3.5, 3.6, 3.7, 3.8, 4. ,  
 4.1, 4.2, 4.3, 4.4, 4.5, 4.6, 4.7, 4.8, 4.9, 5. , 5.1, 5.2, 5.3,  
 5.4, 5.5, 5.6, 5.7, 5.8, 5.9, 6. , 6.1, 6.2, 6.3, 6.4, 6.5, 6.6,  
 6.7, 6.8, 6.9, 7. , 7.1, 7.2, 7.3, 7.4, 7.5, 7.6, 7.7, 7.8, 7.9,  
 8. , 8.1, 8.2, 8.3, 8.4, 8.5, 8.6, 8.8, 9. , 9.6])

data\_num\_Runtime = data\_num[['Runtime']]  
data\_num\_Runtime.head()

Runtime  
0 192.0  
1 117.0  
2 127.0  
3 114.0  
4 102.0

from sklearn.impute import SimpleImputer  
from sklearn.impute import MissingIndicator

indicator = MissingIndicator()  
mask\_missing\_Runtime\_only = indicator.fit\_transform(data\_num\_Runtime)  
mask\_missing\_Runtime\_only

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

imp\_num = SimpleImputer(strategy='most\_frequent')  
data\_num\_imp\_Runtime = imp\_num.fit\_transform(data\_num\_Runtime)  
data\_num\_imp\_Runtime[mask\_missing\_Runtime\_only]

array([104., 104.])

np.unique(data\_num\_imp\_Runtime)

array([ 50., 70., 71., 75., 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., 172.,  
 175., 176., 178., 179., 180., 181., 187., 188., 191., 192., 201.,  
 209., 242.])

### Обработка пропусков в категориальных данных

# 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 Certificate. Тип данных object. Количество пустых значений 34, 1.7%.

cat\_temp\_data = data[['Certificate']]  
cat\_temp\_data.head()

Certificate  
0 PG-13  
1 PG  
2 R  
3 R  
4 PG-13

cat\_temp\_data['Certificate'].unique()

array(['PG-13', 'PG', 'R', 'TV-14', 'TV-MA', 'TV-PG', 'TV-Y7',  
 'Not Rated', nan, 'NC-17', 'TV-G', 'Unrated', 'G'], dtype=object)

cat\_temp\_data[cat\_temp\_data['Certificate'].isnull()].shape

(34, 1)

# Импьютация константой  
imp = SimpleImputer(missing\_values=np.nan, strategy='constant', fill\_value='NA')  
data\_imp = imp.fit\_transform(cat\_temp\_data)  
data\_imp

array([['PG-13'],  
 ['PG'],  
 ['R'],  
 ...,  
 ['R'],  
 ['R'],  
 ['R']], dtype=object)

np.unique(data\_imp)

array(['G', 'NA', 'NC-17', 'Not Rated', 'PG', 'PG-13', 'R', 'TV-14',  
 'TV-G', 'TV-MA', 'TV-PG', 'TV-Y7', 'Unrated'], dtype=object)

data\_imp[data\_imp=='NA'].size

34

### Преобразование категориальных признаков в числовые

Используем OrdinalEncoder, который ориентирован на применение к матрице объект-признак, то есть для кодирования матрицы нецелевых признаков.

from sklearn.preprocessing import OrdinalEncoder

data\_oe = data[['Title', 'Month', 'Certificate', 'Directors', 'Stars', 'Genre', 'Filming\_location', 'Country\_of\_origin']]  
data\_oe.head()

Title Month Certificate \  
0 Avatar: The Way of Water December PG-13   
1 Guillermo del Toro's Pinocchio December PG   
2 Bullet Train August R   
3 The Banshees of Inisherin November R   
4 M3gan December PG-13   
  
 Directors \  
0 James Cameron   
1 Guillermo del Toro, Mark Gustafson   
2 David Leitch   
3 Martin McDonagh   
4 Gerard Johnstone   
  
 Stars \  
0 Sam Worthington, Zoe Saldana, Sigourney Weaver...   
1 Ewan McGregor, David Bradley, Gregory Mann, Bu...   
2 Brad Pitt, Joey King, Aaron Taylor Johnson, Br...   
3 Colin Farrell, Brendan Gleeson, Kerry Condon, ...   
4 Jenna Davis, Amie Donald, Allison Williams, Vi...   
  
 Genre Filming\_location \  
0 Action, Adventure, Fantasy New Zealand   
1 Animation, Drama, Family USA   
2 Action, Comedy, Thriller Japan   
3 Comedy, Drama Ireland   
4 Horror, Sci-Fi, Thriller New Zealand   
  
 Country\_of\_origin   
0 United States   
1 United States, Mexico, France   
2 Japan, United States   
3 Ireland, United Kingdom, United States   
4 United States

data\_oe['Title'].unique()

array(['Avatar: The Way of Water', "Guillermo del Toro's Pinocchio",  
 'Bullet Train', ..., 'Gothika', 'Ong-Bak: The Thai Warrior',  
 'Open Water'], dtype=object)

imp2 = SimpleImputer(missing\_values=np.nan, strategy='constant', fill\_value='NA')  
data\_oe\_filled = imp2.fit\_transform(data\_oe)  
data\_oe\_filled

array([['Avatar: The Way of Water', 'December', 'PG-13', ...,  
 'Action, Adventure, Fantasy', 'New Zealand', 'United States'],  
 ["Guillermo del Toro's Pinocchio", 'December', 'PG', ...,  
 'Animation, Drama, Family', 'USA',  
 'United States, Mexico, France'],  
 ['Bullet Train', 'August', 'R', ..., 'Action, Comedy, Thriller',  
 'Japan', 'Japan, United States'],  
 ...,  
 ['Gothika', 'November', 'R', ..., 'Horror, Mystery, Thriller',  
 'Canada', 'United States, France, Canada, Spain'],  
 ['Ong-Bak: The Thai Warrior', 'February', 'R', ...,  
 'Action, Crime, Thriller', 'Thailand',  
 'Thailand, France, Hong Kong'],  
 ['Open Water', 'August', 'R', ..., 'Adventure, Drama, Horror',  
 'Bahamas', 'United States']], dtype=object)

oe = OrdinalEncoder()  
cat\_enc\_oe = oe.fit\_transform(data\_oe\_filled)  
cat\_enc\_oe

array([[ 151., 4., 5., ..., 7., 56., 245.],  
 [ 597., 4., 4., ..., 98., 87., 355.],  
 [ 249., 3., 6., ..., 25., 41., 121.],  
 ...,  
 [ 580., 11., 6., ..., 233., 11., 293.],  
 [1034., 5., 6., ..., 33., 83., 172.],  
 [1039., 3., 6., ..., 74., 4., 245.]])

# Уникальные значения 1 признака  
np.unique(cat\_enc\_oe[:, 0])

array([0.000e+00, 1.000e+00, 2.000e+00, ..., 1.986e+03, 1.987e+03,  
 1.988e+03])

# Уникальные значения 2 признака  
np.unique(cat\_enc\_oe[:, 1])

array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10., 11., 12.,  
 13.])

# Уникальные значения 3 признака  
np.unique(cat\_enc\_oe[:, 2])

array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10., 11., 12.])

# Наименования категорий в соответствии с порядковыми номерами  
oe.categories\_

[array(["'71", '10 Cloverfield Lane', '10,000 BC', ..., 'Zootopia',  
 'tick, tick... BOOM!', 'Æon Flux'], dtype=object),  
 array(['2008', '2014', 'April', 'August', 'December', 'February',  
 'January', 'July', 'June', 'March', 'May', 'November', 'October',  
 'September'], dtype=object),  
 array(['G', 'NA', 'NC-17', 'Not Rated', 'PG', 'PG-13', 'R', 'TV-14',  
 'TV-G', 'TV-MA', 'TV-PG', 'TV-Y7', 'Unrated'], dtype=object),  
 array(['Aamir Khan, Amole Gupte', 'Aaron Blaise, Robert Walker',  
 'Aaron Hann, Mario Miscione', ..., 'Zach Cregger', 'Zack Snyder',  
 'Zak Hilditch'], dtype=object),  
 array(['A J Cook, Ali Larter, Tony Todd, Michael Landes',  
 'Aamir Khan, Madhavan, Mona Singh, Sharman Joshi',  
 'Aaron Eckhart, Hilary Swank, Delroy Lindo, Bruce Greenwood', ...,  
 'Zoey Deutch, Kendrick Sampson, Ray Nicholson, Shay Mitchell',  
 'Zoey Deutch, Lucy Fry, Danila Kozlovskiy, Gabriel Byrne',  
 'Zooey Deschanel, Joseph Gordon Levitt, Geoffrey Arend, Chlo Grace Moretz'],  
 dtype=object),  
 array(['Action', 'Action, Adventure', 'Action, Adventure, Biography',  
 'Action, Adventure, Comedy', 'Action, Adventure, Crime',  
 'Action, Adventure, Drama', 'Action, Adventure, Family',  
 'Action, Adventure, Fantasy', 'Action, Adventure, Horror',  
 'Action, Adventure, Mystery', 'Action, Adventure, Romance',  
 'Action, Adventure, Sci-Fi', 'Action, Adventure, Thriller',  
 'Action, Adventure, Western', 'Action, Biography, Crime',  
 'Action, Biography, Drama', 'Action, Comedy',  
 'Action, Comedy, Crime', 'Action, Comedy, Drama',  
 'Action, Comedy, Family', 'Action, Comedy, Fantasy',  
 'Action, Comedy, Horror', 'Action, Comedy, Mystery',  
 'Action, Comedy, Romance', 'Action, Comedy, Sci-Fi',  
 'Action, Comedy, Thriller', 'Action, Comedy, War', 'Action, Crime',  
 'Action, Crime, Drama', 'Action, Crime, Fantasy',  
 'Action, Crime, Horror', 'Action, Crime, Mystery',  
 'Action, Crime, Sci-Fi', 'Action, Crime, Thriller',  
 'Action, Drama', 'Action, Drama, Family', 'Action, Drama, Fantasy',  
 'Action, Drama, History', 'Action, Drama, Horror',  
 'Action, Drama, Mystery', 'Action, Drama, Romance',  
 'Action, Drama, Sci-Fi', 'Action, Drama, Sport',  
 'Action, Drama, Thriller', 'Action, Drama, War',  
 'Action, Drama, Western', 'Action, Fantasy, Horror',  
 'Action, Fantasy, Sci-Fi', 'Action, Fantasy, Thriller',  
 'Action, Horror', 'Action, Horror, Mystery',  
 'Action, Horror, Sci-Fi', 'Action, Horror, Thriller',  
 'Action, Horror, War', 'Action, Mystery, Sci-Fi',  
 'Action, Mystery, Thriller', 'Action, Sci-Fi',  
 'Action, Sci-Fi, Thriller', 'Action, Thriller',  
 'Adventure, Biography, Crime', 'Adventure, Biography, Drama',  
 'Adventure, Comedy', 'Adventure, Comedy, Crime',  
 'Adventure, Comedy, Drama', 'Adventure, Comedy, Family',  
 'Adventure, Comedy, Fantasy', 'Adventure, Comedy, Music',  
 'Adventure, Comedy, Mystery', 'Adventure, Comedy, Romance',  
 'Adventure, Comedy, Sci-Fi', 'Adventure, Drama',  
 'Adventure, Drama, Family', 'Adventure, Drama, Fantasy',  
 'Adventure, Drama, History', 'Adventure, Drama, Horror',  
 'Adventure, Drama, Mystery', 'Adventure, Drama, Romance',  
 'Adventure, Drama, Sci-Fi', 'Adventure, Drama, Thriller',  
 'Adventure, Drama, War', 'Adventure, Family, Fantasy',  
 'Adventure, Fantasy', 'Adventure, Fantasy, Horror',  
 'Adventure, Horror', 'Adventure, Horror, Mystery',  
 'Adventure, Horror, Sci-Fi', 'Adventure, Horror, Thriller',  
 'Adventure, Mystery, Sci-Fi', 'Adventure, Sci-Fi, Thriller',  
 'Adventure, Thriller, Western', 'Animation, Action, Adventure',  
 'Animation, Action, Comedy', 'Animation, Adventure, Comedy',  
 'Animation, Adventure, Drama', 'Animation, Adventure, Family',  
 'Animation, Comedy, Drama', 'Animation, Comedy, Family',  
 'Animation, Drama', 'Animation, Drama, Family',  
 'Animation, Drama, Fantasy', 'Animation, Fantasy, Horror',  
 'Biography, Comedy, Crime', 'Biography, Comedy, Drama',  
 'Biography, Comedy, Music', 'Biography, Crime, Drama',  
 'Biography, Drama', 'Biography, Drama, Family',  
 'Biography, Drama, History', 'Biography, Drama, Music',  
 'Biography, Drama, Musical', 'Biography, Drama, Romance',  
 'Biography, Drama, Sport', 'Biography, Drama, Thriller',  
 'Biography, Drama, War', 'Comedy', 'Comedy, Crime',  
 'Comedy, Crime, Drama', 'Comedy, Crime, Family',  
 'Comedy, Crime, Mystery', 'Comedy, Crime, Romance',  
 'Comedy, Crime, Sport', 'Comedy, Crime, Thriller',  
 'Comedy, Crime, Western', 'Comedy, Drama', 'Comedy, Drama, Family',  
 'Comedy, Drama, Fantasy', 'Comedy, Drama, History',  
 'Comedy, Drama, Horror', 'Comedy, Drama, Music',  
 'Comedy, Drama, Musical', 'Comedy, Drama, Mystery',  
 'Comedy, Drama, Romance', 'Comedy, Drama, Sci-Fi',  
 'Comedy, Drama, Sport', 'Comedy, Drama, Thriller',  
 'Comedy, Drama, War', 'Comedy, Family', 'Comedy, Family, Fantasy',  
 'Comedy, Family, Musical', 'Comedy, Family, Romance',  
 'Comedy, Family, Sport', 'Comedy, Fantasy',  
 'Comedy, Fantasy, Horror', 'Comedy, Fantasy, Music',  
 'Comedy, Fantasy, Mystery', 'Comedy, Fantasy, Romance',  
 'Comedy, Horror', 'Comedy, Horror, Musical',  
 'Comedy, Horror, Mystery', 'Comedy, Horror, Romance',  
 'Comedy, Horror, Sci-Fi', 'Comedy, Horror, Thriller',  
 'Comedy, Music', 'Comedy, Music, Romance', 'Comedy, Musical',  
 'Comedy, Musical, Romance', 'Comedy, Mystery', 'Comedy, Romance',  
 'Comedy, Romance, Sci-Fi', 'Comedy, Romance, Sport',  
 'Comedy, Sci-Fi', 'Comedy, Sport', 'Comedy, Western',  
 'Crime, Drama', 'Crime, Drama, Fantasy', 'Crime, Drama, History',  
 'Crime, Drama, Horror', 'Crime, Drama, Music',  
 'Crime, Drama, Musical', 'Crime, Drama, Mystery',  
 'Crime, Drama, Romance', 'Crime, Drama, Sport',  
 'Crime, Drama, Thriller', 'Crime, Horror',  
 'Crime, Horror, Mystery', 'Crime, Horror, Thriller',  
 'Crime, Horror, Western', 'Crime, Mystery, Thriller',  
 'Crime, Thriller', 'Drama', 'Drama, Family',  
 'Drama, Family, Fantasy', 'Drama, Fantasy',  
 'Drama, Fantasy, History', 'Drama, Fantasy, Horror',  
 'Drama, Fantasy, Mystery', 'Drama, Fantasy, Romance',  
 'Drama, Fantasy, Sci-Fi', 'Drama, Fantasy, Thriller',  
 'Drama, Fantasy, War', 'Drama, History', 'Drama, History, Music',  
 'Drama, History, Romance', 'Drama, History, Thriller',  
 'Drama, History, War', 'Drama, Horror', 'Drama, Horror, Music',  
 'Drama, Horror, Musical', 'Drama, Horror, Mystery',  
 'Drama, Horror, Romance', 'Drama, Horror, Sci-Fi',  
 'Drama, Horror, Thriller', 'Drama, Horror, Western',  
 'Drama, Music', 'Drama, Music, Musical', 'Drama, Music, Romance',  
 'Drama, Music, Thriller', 'Drama, Musical, Romance',  
 'Drama, Mystery', 'Drama, Mystery, Romance',  
 'Drama, Mystery, Sci-Fi', 'Drama, Mystery, Thriller',  
 'Drama, Romance', 'Drama, Romance, Sci-Fi',  
 'Drama, Romance, Sport', 'Drama, Romance, Thriller',  
 'Drama, Romance, War', 'Drama, Romance, Western', 'Drama, Sci-Fi',  
 'Drama, Sci-Fi, Thriller', 'Drama, Sport', 'Drama, Thriller',  
 'Drama, Thriller, War', 'Drama, War', 'Drama, Western',  
 'Family, Fantasy, Musical', 'Fantasy, Horror, Mystery',  
 'Fantasy, Horror, Thriller', 'Fantasy, Mystery, Sci-Fi', 'Horror',  
 'Horror, Musical, Sci-Fi', 'Horror, Mystery',  
 'Horror, Mystery, Sci-Fi', 'Horror, Mystery, Thriller',  
 'Horror, Romance, Thriller', 'Horror, Sci-Fi',  
 'Horror, Sci-Fi, Thriller', 'Horror, Thriller', 'Mystery',  
 'Mystery, Thriller', 'Romance, Drama, Family',  
 'Romance, Sci-Fi, Thriller', 'Sci-Fi, Thriller', 'Thriller'],  
 dtype=object),  
 array(['Arctic Ocean', 'Argentina', 'Australia', 'Austria', 'Bahamas',  
 'Bangladesh', 'Belgium', 'Bolivia', 'Brazil', 'Bulgaria',  
 'California USA', 'Canada', 'Chile', 'China', 'Colombia',  
 'Costa Rica', 'Croatia', 'Czech Republic', 'Denmark',  
 'Dominican Republic', 'Ecuador', 'Egypt', 'Estonia', 'Fiji',  
 'Finland', 'France', 'French Polynesia', 'Gabon', 'Georgia',  
 'Germany', 'Greece', 'Guam', 'Guatemala', 'Hungary', 'Iceland',  
 'India', 'Indonesia', 'Iran', 'Ireland', 'Israel', 'Italy',  
 'Japan', 'Jordan', 'Kenya', 'Lebanon', 'Lithuania', 'Luxembourg',  
 'Malaysia', 'Malta', 'Mexico', 'Michigan USA', 'Monaco', 'Morocco',  
 'Namibia', 'Nepal', 'Netherlands', 'New Zealand', 'Norge',  
 'Northern Ireland', 'Norway', 'Official Facebook', 'Ontario',  
 'Panama', 'Peru', 'Poland', 'Polynesia', 'Puerto Rico', 'Romania',  
 'Russia', 'Rwanda', 'Saudi Arabia', 'Serbia', 'Singapore',  
 'Slovakia', 'Slovenia', 'South Africa', 'South Korea', 'Space',  
 'Spain', 'St Vincent and the Grenadines', 'Sweden', 'Switzerland',  
 'Taiwan', 'Thailand', 'The Netherlands', 'Turkey', 'UK', 'USA',  
 'Uganda', 'Ukraine', 'United Arab Emirates', 'United Kingdom',  
 'Unknown', 'Uruguay', 'Vatican City', 'Venezuela', 'Vietnam'],  
 dtype=object),  
 array(['Argentina, Spain', 'Argentina, Spain, France, United Kingdom',  
 'Australia', 'Australia, Canada', 'Australia, France',  
 'Australia, United Kingdom, United States, India',  
 'Australia, United States',  
 'Australia, United States, United Kingdom',  
 'Australia, United States, United Kingdom, China', 'Austria',  
 'Austria, Luxembourg, Germany, France', 'Belgium',  
 'Belgium, France',  
 'Belgium, Germany, Canada, France, United Kingdom, Luxembourg, United States',  
 'Canada', 'Canada, France', 'Canada, France, United States',  
 'Canada, Germany', 'Canada, Hungary, United States',  
 'Canada, Spain, France', 'Canada, Spain, Mexico',  
 'Canada, United Arab Emirates, United States',  
 'Canada, United Kingdom', 'Canada, United States',  
 'Canada, United States, Australia',  
 'Canada, United States, United Kingdom',  
 'Chile, Israel, United States',  
 'China, Germany, United States, Austria, Italy, Canada',  
 'China, Hong Kong', 'China, United States',  
 'China, United States, Canada, Japan',  
 'Czech Republic, United Kingdom, Germany, Canada, United States',  
 'Czech Republic, United Kingdom, Germany, United States',  
 'Denmark, France, Sweden, Germany, Belgium, Tunisia',  
 'Denmark, France, United States, Sweden, Belgium',  
 'Denmark, Germany, Belgium, United Kingdom, France, Sweden',  
 'Denmark, Germany, Belgium, United Kingdom, France, Sweden, United States',  
 'Denmark, Germany, France, Sweden, Italy, Poland',  
 'Denmark, Netherlands, Sweden, Germany, United Kingdom, France, Finland, Norway, Italy',  
 'Denmark, Sweden', 'Denmark, Sweden, Czech Republic, Germany',  
 'Denmark, Sweden, Finland', 'Denmark, Sweden, France, Germany',  
 'Denmark, Sweden, Netherlands', 'Finland, Norway, France, Sweden',  
 'France', 'France, Belgium', 'France, Belgium, Netherlands',  
 'France, Belgium, Spain', 'France, Belgium, United States',  
 'France, Canada', 'France, Canada, United Kingdom',  
 'France, Canada, United States',  
 'France, China, Belgium, Germany, United Arab Emirates, United States, Canada',  
 'France, Germany, Bulgaria, United States',  
 'France, Germany, Italy, Canada, Japan',  
 'France, Germany, Taiwan, Canada, United States, United Kingdom',  
 'France, Germany, United Kingdom',  
 'France, Germany, United Kingdom, Belgium, Romania, Japan',  
 'France, Germany, United States', 'France, Japan, United States',  
 'France, Poland, United States', 'France, Spain, United States',  
 'France, United Kingdom', 'France, United Kingdom, Czech Republic',  
 'France, United Kingdom, Germany, Poland, United States',  
 'France, United Kingdom, United States', 'France, United States',  
 'France, United States, Canada, Russia',  
 'France, United States, Turkey, United Kingdom',  
 'France, United States, United Kingdom', 'Germany',  
 'Germany, Austria, Italy', 'Germany, Canada, United States',  
 'Germany, France',  
 'Germany, France, Spain, United States, Belgium',  
 'Germany, India, United States',  
 'Germany, Luxembourg, United States', 'Germany, United Kingdom',  
 'Germany, United Kingdom, Canada, Sweden, United States',  
 'Germany, United Kingdom, United States',  
 'Germany, United Kingdom, United States, Canada, France',  
 'Germany, United Kingdom, United States, Italy',  
 'Germany, United States', 'Germany, United States, Australia',  
 'Germany, United States, Canada',  
 'Germany, United States, Japan, China, South Africa, Canada',  
 'Germany, United States, Mexico, Canada',  
 'Germany, United States, United Kingdom',  
 'Germany, United States, Uruguay, Paraguay', 'Greece',  
 'Greece, Germany', 'Hong Kong, China',  
 'Hong Kong, China, France, Italy, Germany',  
 'Hong Kong, China, United States',  
 'Hong Kong, Taiwan, United States', 'Hong Kong, United States',  
 'India', 'India, Sweden', 'India, United States', 'Indonesia',  
 'Indonesia, France, United States', 'Iran, France, Australia',  
 'Ireland', 'Ireland, Belgium, Denmark', 'Ireland, Canada',  
 'Ireland, Canada, United Kingdom, United States',  
 'Ireland, United Kingdom',  
 'Ireland, United Kingdom, Greece, France, Netherlands, United States',  
 'Ireland, United Kingdom, United States', 'Ireland, United States',  
 'Italy', 'Italy, France', 'Italy, France, United Kingdom',  
 'Italy, France, United Kingdom, Switzerland',  
 'Italy, France, United States',  
 'Italy, France, United States, Brazil',  
 'Italy, United Kingdom, France, Luxembourg',  
 'Italy, United States', 'Japan', 'Japan, Canada, United States',  
 'Japan, United States', 'Japan, United States, France',  
 'Jordan, Saudi Arabia, Sweden',  
 'Lebanon, France, Cyprus, Qatar, United Kingdom', 'Mexico',  
 'Mexico, Spain', 'Mexico, United States', 'Netherlands',  
 'Netherlands, Belgium',  
 'Netherlands, France, Germany, Belgium, Sweden, United Kingdom, United States',  
 'Netherlands, Germany, United Kingdom, Belgium',  
 'Netherlands, United Kingdom, Finland, United States',  
 'New Zealand', 'New Zealand, Japan, United States',  
 'New Zealand, United Kingdom, Canada, Australia',  
 'New Zealand, United States',  
 'New Zealand, United States, Czech Republic', 'Norway',  
 'Norway, France, Sweden, Denmark', 'Norway, Sweden',  
 'Norway, Sweden, Denmark, Finland, France, United Kingdom',  
 'Poland', 'Russia', 'Russia, United States, China', 'Serbia',  
 'South Africa, United States, Japan',  
 'South Africa, United States, New Zealand, Canada', 'South Korea',  
 'South Korea, Czech Republic', 'South Korea, Japan',  
 'South Korea, United States', 'Spain', 'Spain, Colombia',  
 'Spain, France, United Kingdom, United States', 'Spain, Mexico',  
 'Spain, Thailand, United States',  
 'Spain, United Kingdom, France, United States',  
 'Spain, United Kingdom, United States', 'Spain, United States',  
 'Spain, United States, France', 'Sweden',  
 'Sweden, Denmark, Germany, Norway',  
 'Sweden, France, Norway, Denmark, Italy',  
 'Sweden, France, United Kingdom, Germany, Turkey, Greece, United States, Denmark, Switzerland, Mexico',  
 'Sweden, Germany, France, Denmark', 'Sweden, Netherlands, France',  
 'Sweden, United States', 'Switzerland, Canada, United States',  
 'Switzerland, United Kingdom, United States, Mexico', 'Taiwan',  
 'Taiwan, United States, Hong Kong, China',  
 'Thailand, France, Hong Kong', 'Turkey',  
 'United Arab Emirates, United States',  
 'United Arab Emirates, United States, France', 'United Kingdom',  
 'United Kingdom, Australia',  
 'United Kingdom, Australia, France, United States',  
 'United Kingdom, Australia, United States',  
 'United Kingdom, Belgium, United States', 'United Kingdom, Canada',  
 'United Kingdom, Canada, Ireland',  
 'United Kingdom, Canada, United States',  
 'United Kingdom, Canada, United States, Australia',  
 'United Kingdom, Canada, United States, Sweden',  
 'United Kingdom, China',  
 'United Kingdom, China, France, United States',  
 'United Kingdom, China, Japan, United States',  
 'United Kingdom, China, Japan, United States, Mexico',  
 'United Kingdom, China, United States',  
 'United Kingdom, Czech Republic, Germany, United States',  
 'United Kingdom, Czech Republic, United States, Germany, Bahamas',  
 'United Kingdom, Denmark', 'United Kingdom, France',  
 'United Kingdom, France, Canada',  
 'United Kingdom, France, Canada, United States',  
 'United Kingdom, France, Germany, Spain, United States',  
 'United Kingdom, France, Germany, United States',  
 'United Kingdom, France, Italy, Germany, Canada, United States',  
 'United Kingdom, France, United States',  
 'United Kingdom, France, United States, Belgium',  
 'United Kingdom, France, United States, Canada',  
 'United Kingdom, France, United States, China',  
 'United Kingdom, France, United States, China, Canada',  
 'United Kingdom, Germany',  
 'United Kingdom, Germany, Austria, United States',  
 'United Kingdom, Germany, Japan, United States',  
 'United Kingdom, Germany, Kenya, France, United States, Switzerland',  
 'United Kingdom, Germany, Spain, Morocco, United States, Italy, France',  
 'United Kingdom, Germany, Spain, United States',  
 'United Kingdom, Germany, Sweden, Belgium, United States',  
 'United Kingdom, Germany, United States',  
 'United Kingdom, Germany, United States, Chile',  
 'United Kingdom, Hungary, United States, Australia',  
 'United Kingdom, Ireland, United States',  
 'United Kingdom, Italy, France, United States',  
 'United Kingdom, Japan', 'United Kingdom, Luxembourg',  
 'United Kingdom, Luxembourg, France, Belgium, United States',  
 'United Kingdom, Malta, United States', 'United Kingdom, Mexico',  
 'United Kingdom, Mexico, Hong Kong, United States',  
 'United Kingdom, Netherlands, France, United States',  
 'United Kingdom, Norway, China, Canada, United States, France',  
 'United Kingdom, South Africa, India',  
 'United Kingdom, South Africa, Italy, United States',  
 'United Kingdom, Spain, United States',  
 'United Kingdom, Switzerland', 'United Kingdom, United States',  
 'United Kingdom, United States, Australia',  
 'United Kingdom, United States, Canada',  
 'United Kingdom, United States, Canada, Australia, Japan, China',  
 'United Kingdom, United States, China',  
 'United Kingdom, United States, France',  
 'United Kingdom, United States, France, Italy',  
 'United Kingdom, United States, Germany, Denmark, Belgium, Japan',  
 'United Kingdom, United States, Iceland',  
 'United Kingdom, United States, India, Spain',  
 'United Kingdom, United States, Japan',  
 'United Kingdom, United States, Japan, China, Croatia',  
 'United Kingdom, United States, Luxembourg',  
 'United Kingdom, United States, South Africa, Germany',  
 'United Kingdom, United States, Sweden, China',  
 'United Kingdom, United States, Turkey', 'United States',  
 'United States, Argentina', 'United States, Australia',  
 'United States, Australia, China',  
 'United States, Australia, China, Hong Kong',  
 'United States, Australia, France, Canada, Germany, Brazil, United Kingdom',  
 'United States, Australia, Germany, Japan',  
 'United States, Australia, Spain',  
 'United States, Australia, United Kingdom',  
 'United States, Australia, United Kingdom, Canada',  
 'United States, Belgium, Denmark, France', 'United States, Brazil',  
 'United States, Brazil, Canada', 'United States, Bulgaria',  
 'United States, Bulgaria, Germany, India',  
 'United States, Bulgaria, Spain, Germany', 'United States, Canada',  
 'United States, Canada, Bulgaria', 'United States, Canada, China',  
 'United States, Canada, France',  
 'United States, Canada, France, Germany',  
 'United States, Canada, Germany',  
 'United States, Canada, Germany, France',  
 'United States, Canada, Hong Kong',  
 'United States, Canada, Ireland', 'United States, Canada, Japan',  
 'United States, Canada, Japan, France',  
 'United States, Canada, Luxembourg',  
 'United States, Canada, New Zealand, Hungary',  
 'United States, Canada, United Kingdom', 'United States, China',  
 'United States, China, Brazil', 'United States, China, Canada',  
 'United States, China, France, Norway, United Kingdom',  
 'United States, China, Germany', 'United States, China, Hong Kong',  
 'United States, China, Hong Kong, Australia, Canada',  
 'United States, China, Japan',  
 'United States, China, Japan, Canada, United Arab Emirates',  
 'United States, China, Malta',  
 'United States, China, Spain, Japan, Germany',  
 'United States, China, United Kingdom',  
 'United States, Czech Republic',  
 'United States, Czech Republic, Romania, United Kingdom',  
 'United States, Czech Republic, Spain',  
 'United States, Czech Republic, United Kingdom',  
 'United States, Denmark, Australia', 'United States, France',  
 'United States, France, Canada, Spain',  
 'United States, France, Hong Kong',  
 'United States, France, Italy, United Kingdom',  
 'United States, France, Italy, United Kingdom, Germany',  
 'United States, France, Japan', 'United States, France, Morocco',  
 'United States, France, United Kingdom, Austria, Germany, Italy',  
 'United States, France, United Kingdom, Hong Kong, Taiwan, Malta, Spain, Canada',  
 'United States, Germany', 'United States, Germany, Australia',  
 'United States, Germany, Brazil, Italy',  
 'United States, Germany, Canada, France',  
 'United States, Germany, Canada, United Kingdom',  
 'United States, Germany, China',  
 'United States, Germany, China, Canada',  
 'United States, Germany, China, Italy',  
 'United States, Germany, Czech Republic, United Kingdom',  
 'United States, Germany, France',  
 'United States, Germany, France, Spain',  
 'United States, Germany, France, United Kingdom',  
 'United States, Germany, Hong Kong, Singapore, China',  
 'United States, Germany, Hungary',  
 'United States, Germany, Italy, Spain, France, Japan, United Kingdom',  
 'United States, Germany, Japan',  
 'United States, Germany, Japan, United Kingdom, Hong Kong',  
 'United States, Germany, Russia, India',  
 'United States, Germany, Sweden, Hungary',  
 'United States, Germany, United Kingdom',  
 'United States, Germany, United Kingdom, Australia',  
 'United States, Germany, United Kingdom, Canada',  
 'United States, Germany, United Kingdom, Italy',  
 'United States, Germany, United Kingdom, Netherlands',  
 'United States, Greece', 'United States, Hong Kong',  
 'United States, Hong Kong, Bulgaria, Netherlands, Canada, United Kingdom, France',  
 'United States, Hong Kong, Canada',  
 'United States, Hong Kong, China',  
 'United States, Hong Kong, France, Bulgaria, Spain, Sweden',  
 'United States, Hong Kong, Japan, Canada, New Zealand',  
 'United States, Hungary', 'United States, Iceland, Canada',  
 'United States, India',  
 'United States, India, Hong Kong, China, Canada',  
 'United States, Ireland', 'United States, Ireland, United Kingdom',  
 'United States, Italy', 'United States, Italy, Canada',  
 'United States, Italy, Hong Kong, Canada',  
 'United States, Italy, Romania, United Kingdom',  
 'United States, Italy, Switzerland, Thailand',  
 'United States, Japan', 'United States, Japan, Canada',  
 'United States, Japan, China', 'United States, Japan, Colombia',  
 'United States, Japan, Mexico',  
 'United States, Japan, South Korea, Philippines, United Arab Emirates, Canada',  
 'United States, Japan, Spain, United Kingdom',  
 'United States, Malta, France, United Kingdom',  
 'United States, Malta, Morocco',  
 'United States, Malta, United Kingdom', 'United States, Mexico',  
 'United States, Mexico, Canada', 'United States, Mexico, France',  
 'United States, Mexico, Hong Kong',  
 'United States, Mexico, Hungary, Canada',  
 'United States, Netherlands', 'United States, New Zealand',  
 'United States, New Zealand, Canada',  
 'United States, New Zealand, Germany',  
 'United States, New Zealand, Japan',  
 'United States, New Zealand, Japan, China',  
 'United States, New Zealand, United Kingdom, France, Australia',  
 'United States, Norway', 'United States, Poland',  
 'United States, Poland, Slovenia, Czech Republic, United Kingdom',  
 'United States, Romania', 'United States, Russia',  
 'United States, South Africa',  
 'United States, South Africa, Canada',  
 'United States, South Africa, France',  
 'United States, South Africa, India', 'United States, Spain',  
 'United States, Spain, France, Canada, United Kingdom',  
 'United States, Spain, Hungary', 'United States, Sweden',  
 'United States, Sweden, Norway', 'United States, Switzerland',  
 'United States, Taiwan, United Kingdom, Canada, India, France',  
 'United States, United Arab Emirates',  
 'United States, United Arab Emirates, Poland, India',  
 'United States, United Kingdom',  
 'United States, United Kingdom, Australia',  
 'United States, United Kingdom, Belgium, Canada',  
 'United States, United Kingdom, Bulgaria',  
 'United States, United Kingdom, Bulgaria, Canada, Portugal, France',  
 'United States, United Kingdom, Canada',  
 'United States, United Kingdom, Canada, Japan',  
 'United States, United Kingdom, Canada, Spain',  
 'United States, United Kingdom, China',  
 'United States, United Kingdom, France',  
 'United States, United Kingdom, Germany',  
 'United States, United Kingdom, Germany, Hungary',  
 'United States, United Kingdom, Germany, Netherlands, France, Italy',  
 'United States, United Kingdom, Hungary',  
 'United States, United Kingdom, Hungary, Jordan',  
 'United States, United Kingdom, India',  
 'United States, United Kingdom, Israel, Greece',  
 'United States, United Kingdom, Japan',  
 'United States, United Kingdom, Japan, Hungary',  
 'United States, United Kingdom, Luxembourg',  
 'United States, United Kingdom, Malta',  
 'United States, United Kingdom, New Zealand',  
 'United States, United Kingdom, Taiwan, Japan, Mexico'],  
 dtype=object)]

# Обратное преобразование  
oe.inverse\_transform(cat\_enc\_oe)

array([['Avatar: The Way of Water', 'December', 'PG-13', ...,  
 'Action, Adventure, Fantasy', 'New Zealand', 'United States'],  
 ["Guillermo del Toro's Pinocchio", 'December', 'PG', ...,  
 'Animation, Drama, Family', 'USA',  
 'United States, Mexico, France'],  
 ['Bullet Train', 'August', 'R', ..., 'Action, Comedy, Thriller',  
 'Japan', 'Japan, United States'],  
 ...,  
 ['Gothika', 'November', 'R', ..., 'Horror, Mystery, Thriller',  
 'Canada', 'United States, France, Canada, Spain'],  
 ['Ong-Bak: The Thai Warrior', 'February', 'R', ...,  
 'Action, Crime, Thriller', 'Thailand',  
 'Thailand, France, Hong Kong'],  
 ['Open Water', 'August', 'R', ..., 'Adventure, Drama, Horror',  
 'Bahamas', 'United States']], dtype=object)

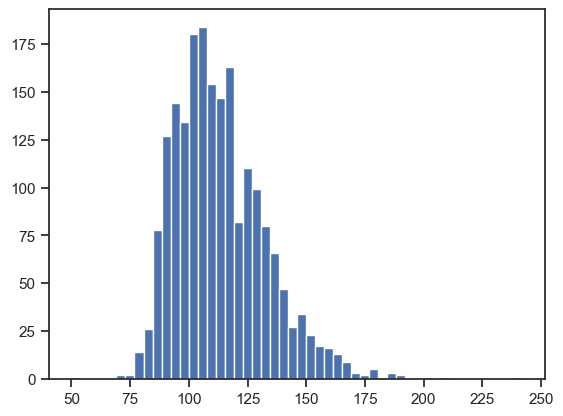
## Масштабирование данных

from sklearn.preprocessing import MinMaxScaler, StandardScaler

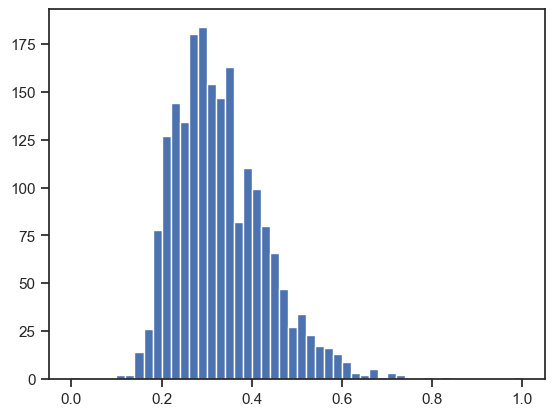
### MinMax масштабирование

sc1 = MinMaxScaler()  
sc1\_data = sc1.fit\_transform(data[['Runtime']])

plt.hist(data['Runtime'], 50)  
plt.show()



plt.hist(sc1\_data, 50)  
plt.show()



### Масштабирование данных на основе Z-оценки - StandardScaler

sc2 = StandardScaler()  
sc2\_data = sc2.fit\_transform(data[['Runtime']])

plt.hist(sc2\_data, 50)  
plt.show()

