Министерство науки и высшего образования Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего образования «Московский государственный технический университет имени Н.Э. Баумана

(национальный исследовательский университет)»

(МГТУ им. Н.Э. Баумана)

Факультет «Информатика и системы управления»

Кафедра ИУ5 «Системы обработки информации и управления»

Отчёт по РК1

«Технологии машинного обучения»

Вариант 9

Выполнила:

студентка группы ИУ5-63Б

Пересыпкина М.А.

Преподаватель:

Гапанюк Ю. Е.

2023 г.

Задание:

Задача №2.

Для заданного набора данных проведите обработку пропусков в данных для одного категориального и одного количественного признака. Какие способы обработки пропусков в данных для категориальных и количественных признаков Вы использовали? Какие признаки Вы будете использовать для дальнейшего построения моделей машинного обучения и почему?

Для студентов групп ИУ5-63Б, ИУ5Ц-83Б - для произвольной колонки данных построить график "Ящик с усами (boxplot)".

https://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_iris.html#sklearn.datasets.load_iris

Решение:

Загружаем датасет, подключаем необходимые библиотеки и проводим первичный анализ данных:

Ввод [4]:
```
import sklearn
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
import math as mth
from sklearn.datasets import load_iris
plt.rcParams.update({'figure.max_open_warning': 0})
```

Ввод [5]:
```
data = load_iris()
```

Ввод [6]:
```
data.feature_names
```

Out[6]:
```
['sepal length (cm)',
 'sepal width (cm)',
 'petal length (cm)',
 'petal width (cm)']
```

Ввод [7]:
```
data = pd.DataFrame(data=data['data'],columns=data['feature_names'])
```

Ввод [8]:
```
data.head()
```

Out[8]:

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2

Ввод [9]:
```
# проверим есть ли пропущенные значения
data.isnull().sum()
```

Out[9]:
```
sepal length (cm)    0
sepal width (cm)     0
petal length (cm)    0
petal width (cm)     0
dtype: int64
```

Датасет не подходит под задачу, поэтому возьмем другой датасет

Ввод [13]:
```
data = pd.read_csv(r'movies_dataset.csv', sep=",")
```

Ввод [14]:
```
# типы колонок
data.dtypes
```

Out[14]:
```
Unnamed: 0           int64
IMDb-rating        float64
appropriate_for     object
director            object
downloads           object
id                   int64
industry            object
language            object
posted_date         object
release_date        object
run_time            object
storyline           object
title               object
views               object
writer              object
dtype: object
```

Ввод [15]:
```
# проверим есть ли пропущенные значения
data.isnull().sum()
```

Out[15]:
```
Unnamed: 0            0
IMDb-rating         841
appropriate_for    9476
director           1938
downloads             1
id                    0
industry              1
language            542
posted_date           1
release_date          1
run_time           1768
storyline          1701
title                 1
views                 1
writer             2192
dtype: int64
```

Ввод [16]:
```
# Первые 5 строк датасета
data.head()
```

Out[16]:

	Unnamed: 0	IMDb-rating	appropriate_for	director	downloads	id	industry	language	poste
0	0	4.8	R	John Swab	304	372092	Hollywood / English	English	
1	1	6.4	TV-PG	Paul Ziller	73	372091	Hollywood / English	English	
2	2	5.2	R	Ben Wheatley	1,427	343381	Hollywood / English	English,Hindi	20 Ap
3	3	8.1	NaN	Venky Atluri	1,549	372090	Tollywood	Hindi	
4	4	4.6	NaN	Shaji Kailas	657	372089	Tollywood	Hindi	

Ввод [19]:
```
total_count = data.shape[0]
print('Всего строк: {}'.format(total_count))
```

```
Всего строк: 20548
```

Обработка пропусков в числовых данных:

Ввод [20]:
```
# 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 IMDb-rating. Тип данных float64. Количество пустых значений 841, 4.09%.
```

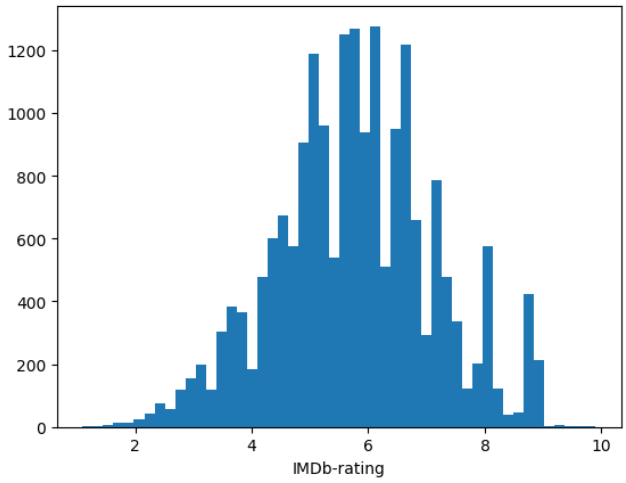
Ввод [21]:
```
# Фильтр по колонкам с пропущенными значениями
data_num = data[num_cols]
data_num
```

Out[21]:

	IMDb-rating
0	4.8
1	6.4
2	5.2
3	8.1
4	4.6
...	...
20543	NaN
20544	7.7
20545	8.0
20546	NaN
20547	NaN

20548 rows × 1 columns

Ввод [22]:
```
# Гистограмма по признакам
for col in data_num:
    plt.hist(data[col], 50)
    plt.xlabel(col)
    plt.show()
```



Ввод [24]:
```
data_num_IMDb = data_num[['IMDb-rating']]
data_num_IMDb.head()
```

Out[24]:

	IMDb-rating
0	4.8
1	6.4
2	5.2
3	8.1
4	4.6

Ввод [25]:
```
from sklearn.impute import SimpleImputer
from sklearn.impute import MissingIndicator
```

Ввод [26]:
```
# Фильтр для проверки заполнения пустых значений
indicator = MissingIndicator()
mask_missing_values_only = indicator.fit_transform(data_num_IMDb)
mask_missing_values_only
```

Out[26]:
```
array([[False],
       [False],
       [False],
       ...,
       [False],
       [ True],
       [ True]])
```

Ввод [27]:
```
strategies=['mean', 'median', 'most_frequent']
```

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

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

Out[29]:
```
('mean',
 array([5.7621505, 5.7621505, 5.7621505, 5.7621505, 5.7621505, 5.7621505,
        5.7621505, 5.7621505, 5.7621505, 5.7621505, 5.7621505,
```

Ввод [30]:
```
strategies[1], test_num_impute(strategies[1])
```

Out[30]:
```
('median',
 array([5.7, 5.7, 5.7, 5.7, 5.7, 5.7, 5.7, 5.7, 5.7, 5.7, 5.7, 5.7, 5.7,
```

Ввод [31]:
```
strategies[2], test_num_impute(strategies[2])
```

Out[31]:
```
('most_frequent',
 array([6.6, 6.6, 6.6, 6.6, 6.6, 6.6, 6.6, 6.6, 6.6, 6.6, 6.6, 6.6, 6.6,
```

Ввод [32]:
```
# Более сложная функция, которая позволяет задавать колонку и вид импьютации
def test_num_impute_col(dataset, column, strategy_param):
    temp_data = dataset[[column]]

    indicator = MissingIndicator()
    mask_missing_values_only = indicator.fit_transform(temp_data)

    imp_num = SimpleImputer(strategy=strategy_param)
    data_num_imp = imp_num.fit_transform(temp_data)

    filled_data = data_num_imp[mask_missing_values_only]

    return column, strategy_param, filled_data.size, filled_data[0], filled_data[filled_data.size-1]
```

Ввод [42]:
```
test_num_impute_col(data, 'IMDb-rating', strategies[0])
```

Out[42]:
```
('IMDb-rating', 'mean', 841, 5.762150504896736, 5.762150504896736)
```

Ввод [43]:
```
test_num_impute_col(data, 'IMDb-rating', strategies[1])
```

Out[43]:
```
('IMDb-rating', 'median', 841, 5.7, 5.7)
```

Ввод [45]:
```
test_num_impute_col(data, 'IMDb-rating', strategies[2])
```

Out[45]:
```
('IMDb-rating', 'most_frequent', 841, 6.6, 6.6)
```

Ввод [46]:
```
data['IMDb-rating'] = data['IMDb-rating'].fillna(data['IMDb-rating'].median())
```

Заполнили пропуски медианой. Так же можно использовать заполнение наиболее часто встречающимися значениями и заполнение средним значением.

Обработка пропусков в категориальных данных:

Ввод [33]:
```
# 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 appropriate_for. Тип данных object. Количество пустых значений 9476, 46.12%.
Колонка director. Тип данных object. Количество пустых значений 1938, 9.43%.
Колонка downloads. Тип данных object. Количество пустых значений 1, 0.0%.
Колонка industry. Тип данных object. Количество пустых значений 1, 0.0%.
Колонка language. Тип данных object. Количество пустых значений 542, 2.64%.
Колонка posted_date. Тип данных object. Количество пустых значений 1, 0.0%.
Колонка release_date. Тип данных object. Количество пустых значений 1, 0.0%.
Колонка run_time. Тип данных object. Количество пустых значений 1768, 8.6%.
Колонка storyline. Тип данных object. Количество пустых значений 1701, 8.28%.
Колонка title. Тип данных object. Количество пустых значений 1, 0.0%.
Колонка views. Тип данных object. Количество пустых значений 1, 0.0%.
Колонка writer. Тип данных object. Количество пустых значений 2192, 10.67%.
```

Ввод [34]:
```
cat_temp_data = data[['appropriate_for']]
cat_temp_data.head()
```

Out[34]:

	appropriate_for
0	R
1	TV-PG
2	R
3	NaN
4	NaN

Ввод [35]:
```
cat_temp_data['appropriate_for'].unique()
```

Out[35]:
```
array(['R', 'TV-PG', nan, 'PG-13', 'Unrated', 'Not Rated', 'TV-MA',
       'TV-14', 'TV-G', 'PG', 'TV-Y7', 'G', 'NC-17', 'TV-Y', 'Approved',
       'TV-Y7-FV', 'MA-17', 'TV-13', 'Drama', 'Drama, Romance', 'Passed',
       '18+'], dtype=object)
```

Для обработки пропусков в категориальных данных будем использовать импьютацию константой, так как не можем определить значения для этой колонки. Так же для обработки пропусков в категориальных данных можно использовать заполнение наиболее частыми значениями.

Ввод [38]:
```
# Импьютация константой
imp3 = SimpleImputer(missing_values=np.nan, strategy='constant', fill_value='NA')
data_imp3 = imp3.fit_transform(cat_temp_data)
data_imp3
```

Out[38]:
```
array([['R'],
       ['TV-PG'],
       ['R'],
       ...,
       ['NA'],
       ['NA'],
       ['NA']], dtype=object)
```

Ввод [39]:
```
np.unique(data_imp3)
```

Out[39]:
```
array(['18+', 'Approved', 'Drama', 'Drama, Romance', 'G', 'MA-17', 'NA',
       'NC-17', 'Not Rated', 'PG', 'PG-13', 'Passed', 'R', 'TV-13',
       'TV-14', 'TV-G', 'TV-MA', 'TV-PG', 'TV-Y', 'TV-Y7', 'TV-Y7-FV',
       'Unrated'], dtype=object)
```

Ввод [40]:
```
data_imp3[data_imp3=='NA'].size
```

Out[40]:
```
9476
```

Построение графика "Ящик с усами (boxplot)":

Ввод [49]:
```
sns.boxplot(x=data["IMDb-rating"])
```

Out[49]:
```
<Axes: xlabel='IMDb-rating'>
```