МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
им. Н.Э. Баумана

Факультет «Информатика и системы управления»
Кафедра «Систем обработки информации и управления»

ОТЧЕТ

Лабораторная работа №__1__
по дисциплине «Методы машинного обучения в автоматизированных системах»

Тема: «Создание «истории данных» (Data Storytelling)»

ИСПОЛНИТЕЛЬ: ___Калюта Н.И.___
ФИО
группа ИУ5-22М _____
подпись
"23"___05___2024 г.

ПРЕПОДАВАТЕЛЬ: _____
ФИО
_____
подпись
"__"_____2024 г.

Москва - 2024

Задание:

- Выбрать набор данных (датасет). Вы можете найти список свободно распространяемых датасетов здесь. Для лабораторных работ не рекомендуется выбирать датасеты очень большого размера.
- Создать "историю о данных" в виде юпитер-ноутбука, с учетом следующих требований:
    1. История должна содержать не менее 5 шагов (где 5 - рекомендуемое количество шагов). Каждый шаг содержит график и его текстовую интерпретацию.
    2. На каждом шаге наряду с удачным итоговым графиком рекомендуется в юпитер-ноутбуке оставлять результаты предварительных "неудачных" графиков.
    3. Не рекомендуется повторять виды графиков, желательно создать 5 графиков различных видов.
    4. Выбор графиков должен быть обоснован использованием методологии data-to-viz. Рекомендуется учитывать типичные ошибки построения выбранного вида графика по методологии data-to-viz. Если методология Вами отвергается, то просьба обосновать Ваше решение по выбору графика.
    5. История должна содержать итоговые выводы. В реальных "историях о данных" именно эти выводы представляют собой основную ценность для предприятия.
- Сформировать отчет и разместить его в своем репозитории на github.

Описание датасета

Для выполнения работы был выран набор данных "52,000 Animation Movie Details". Этот набор данных содержит подробную информацию о 54 000 анимационных фильмах, включая такие характеристики, как название, среднее число голосов, подсчет голосов, дата выхода, выручка, время выполнения и многое другое. Всего в датасете 23 столбца. Я привел таблицу с названиями столбцов и их описанием. Так же у столбцов с численными значениями и с категориальными значениями я сделала пометки.

```
import pandas as pd

data = [
    ["id","Уникальный идентификатор фильма.", ""],
    ["title", "Название фильма.", ""],
    ["vote_average", "Средняя оценка фильма.", "num"],
    ["vote_count", "Количество голосов.", "num"],
    ["status","Статус фильма (например, выпущен, в производстве и т.д.).", "cat"],
    ["release_date", "Дата выхода фильма.", ""],
    ["revenue","Доход, полученный от фильма.", "num"],
    ["runtime","Продолжительность фильма в минутах.", "num"],
    ["adult","Указывает, подходит ли фильм для взрослых.", "cat"],
    ["backdrop_path","URL-адрес фонового изображения для фильма.", ""],
    ["budget","Бюджет, выделенный на фильм.", "num"],
    ["homepage","URL-адрес официального веб-сайта фильма", ""],
    ["imdb_id","Идентификатор фильма в IMDB.", ""],
    ["original_language","Язык оригинала фильма.", "cat"],
    ["original_title","Оригинальное название фильма.", ""],
    ["overview","Краткое содержание или обзор фильма.", ""],
    ["popularity","Рейтинг популярности фильма.", "num"],
    ["poster_path","URL изображения постера фильма.", ""],
    ["tagline","Слоган, связанный с фильмом.", ""],
    ["genres","Список жанров, связанных с фильмом.", "[cat]"],
    ["production_companies","Список продюсерских компаний, задействованных в фильме.", "[cat]"],
    ["production_countries","Список стран, в которых был снят фильм.", "[cat]"],
    ["spoken_languages","Список языков, на которых говорят в фильме.", "[cat]"]
    ]

df = pd.DataFrame(data)
df
```

	0	1	2
0	id	Уникальный идентификатор фильма.	
1	title	Название фильма.	
2	vote_average	Средняя оценка фильма.	num
3	vote_count	Количество голосов.	num
4	status	Статус фильма (например, выпущен, в производст...	cat
5	release_date	Дата выхода фильма.	
6	revenue	Доход, полученный от фильма.	num
7	runtime	Продолжительность фильма в минутах.	num
8	adult	Указывает, подходит ли фильм для взрослых.	cat
9	backdrop_path	URL-адрес фонового изображения для фильма.	
10	budget	Бюджет, выделенный на фильм.	num
11	homepage	URL-адрес официального веб-сайта фильма	
12	imdb_id	Идентификатор фильма в IMDB.	
13	original_language	Язык оригинала фильма.	cat
14	original_title	Оригинальное название фильма.	
15	overview	Краткое содержание или обзор фильма.	
16	popularity	Рейтинг популярности фильма.	num
17	poster_path	URL изображения постера фильма.	
18	tagline	Слоган, связанный с фильмом.	
19	genres	Список жанров, связанных с фильмом.	[cat]
20	production_companies	Список продюсерских компаний, задействованных ...	[cat]
21	production_countries	Список стран, в которых был снят фильм.	[cat]
22	spoken_languages	Список языков, на которых говорят в фильме.	[cat]

Загрузка данных

```
[ ] from google.colab import drive
    drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
import matplotlib.pyplot as plt
import seaborn as sns
df = pd.read_csv("/content/drive/MyDrive/Временные файлы/Animation_Movies.csv")
df
```

1	14160	Up	7.949	18857	Released	2009-05-28	735099082	96	False	/hGGC9gKo7CFE3fW07RA587e5kol.jpg	...	en	Up
2	12	Finding Nemo	7.824	18061	Released	2003-05-30	940335536	100	False	/h3uqFk7sZRJvLZDdLiFB9qwbL07.jpg	...	en	Finding Nemo
3	354912	Coco	8.222	17742	Released	2017-10-27	800526015	105	False	/askg3SMvhqEl4OL52YuvdtY40Yb.jpg	...	en	Coco
4	10681	WALL·E	8.078	17446	Released	2008-06-22	521311860	98	False	/fK5ssgvtI43z19FoWlgdlqgpLRE.jpg	...	en	WALL·E
...	...	...	...	...	...	...	...	...	...	...	...	...	...
51940	656677	Белозубка	0.000	0	Released	2018-12-20	0	0	False	NaN	...	ru	Белозубка
51941	657149	Shimajiro to Ururu no Heroland	0.000	0	Released	2019-03-15	0	60	False	/jQMHu7B7LOY3R2PNJA4bBahEewN.jpg	...	ja	映画しまじろうとうるるのヒーローランド
51942	656945	Robo Force: The Revenge of Nazgar	0.000	0	Released	1984-12-08	0	22	False	NaN	...	en	Robo Force: The Revenge of Nazgar
51943	656893	Beginning Responsibility: A Lunchroom Goes Ban...	0.000	0	Released	1978-01-01	0	12	False	NaN	...	en	Beginning Responsibility: A Lunchroom Goes Ban...
51944	656966	Natural Selection	0.000	0	Released	2019-08-20	0	10	False	NaN	...	bs	Prirodni odabir

```
# Колонки с пропусками
df_na = [c for c in df.columns if df[c].isnull().sum() > 0]
# Доля (процент) пропусков
[(c, df[c].isnull().mean()) for c in df_na]
```

```
[('title', 1.9251131003946483e-05),
 ('release_date', 0.04113966695543363),
 ('backdrop_path', 0.6951583405525075),
 ('homepage', 0.8411204158244296),
 ('imdb_id', 0.43109057657137356),
 ('original_title', 1.9251131003946483e-05),
 ('overview', 0.11702762537299066),
 ('poster_path', 0.26972759649629413),
 ('tagline', 0.9099432091635383),
 ('production_companies', 0.43405525074598134),
 ('production_countries', 0.23573009914332468),
 ('spoken_languages', 0.34896525170853787)]
```

Первые признаки, которые я увидел и их можно оценить - это чмсленный признак бюджет, выделенный на фильм и доход, выделенный на фильм. Сравним их, это поможет оценить. Поскольку мы имеем 2 числовых признака, более 20000 записей, 2 признака и они не сортированы, то в соответствии с методологией data-to-viz https://www.data-to-viz.com/#scatter выберем график jpontplot().

```
[ ] # use the function regplot to make a scatterplot
    sns.jointplot(x=df["revenue"], y=df["budget"], kind='scatter')
```

```
<seaborn.axisgrid.JointGrid at 0x7e91312ebc40>
```

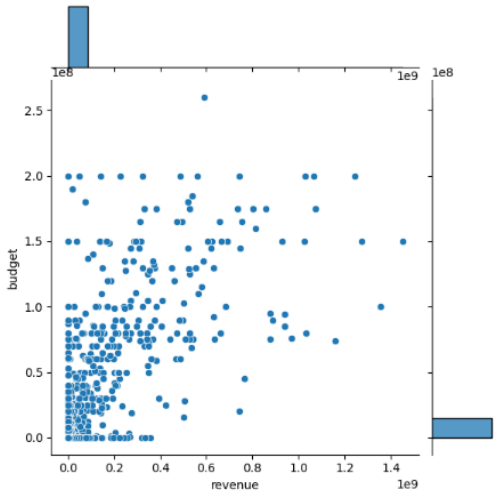


График подобран удачно. По нему можно увидеть, что разброс численных значений очеть велик. от 10^3 до 10^9.

Следующие два численных признака, которые интересно визуализировать это средняя оценка фильма и количество голосов.

```
sns.set_style("white")
sns.kdeplot(x=df.vote_average, y=df.vote_count, cmap="Reds", fill=True)
```

```
<Axes: xlabel='vote_average', ylabel='vote_count'>
```

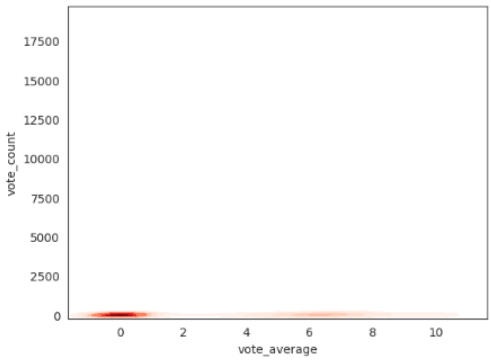
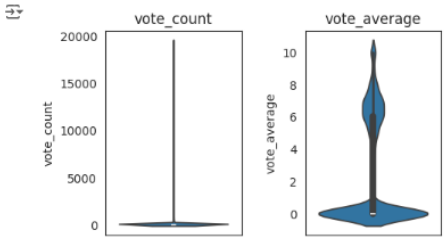


График оказался не очень информативным, тк большая часть сумм голосов не превышает 1000, но тк есть всего несколько значений с суммой голосов в тысячу раз больше, то график выглядит "сплюснутым". Выведем распределение этих параметров на двух отдельных графиках.

```
[ ]
plt.figure(figsize=(12, 8))

plt.subplot(3, 5, 1)
sns.violinplot(y=df["vote_count"])
plt.title("vote_count")
plt.subplot(3, 5, 2)
sns.violinplot(y=df["vote_average"])
plt.title("vote_average")

plt.tight_layout()
plt.show()
```



Мы видим, что у большинства фильмов датасета очень маленькое количество голосов.Не более ста примерно. А еще у многих фильмов средняя оценка близка или равна нулю. Предполагаю, что это означает,что пока никто еще не оценил эти фильмы.

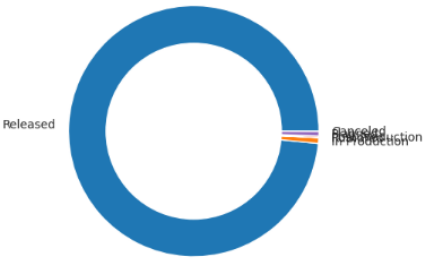
Перейдем к визуализации категориальных признаков. Рассмотрим возможные статусы фильмов, поскольку по прошлому графику мы сделали предположение о том, что у многих фильмов нет голосов, тоесть они еще не выпущены. В данном случае мы визуализируем один категориальный признак. В соответствии с методологией data-to-viz https://www.data-to-viz.com/#scatter выберем график donut.

```
[ ] df["status"].values
    status_unique = df["status"].unique ()
    d = dict.fromkeys(status_unique, 0)
    for i in df["status"].values:
      d[i] = d[i]+1
    num_unique = d.values()
```

```
[ ] plt.pie(num_unique, labels = status_unique)

    # add a circle at the center to transform it in a donut chart
    my_circle=plt.Circle( (0,0), 0.7, color='white')
    p=plt.gcf()
    p.gca().add_artist(my_circle)

    plt.show()
```



Предположени не подтвердилось. Практически васе фильмы выпущены.

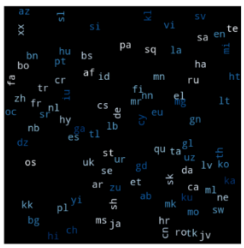
Визуализируем информацию об еще одном категориальном признаке - языках оригинала фильмов. Используем для этого словестную диаграмму.

```
[ ] lang_unique = df["original_language"].unique()
    str = ""
    for i in lang_unique:
      str  = str + " " + i
    str
```

```
' en ja fr he it es zh pl uk ko da ru de cs pt xx tr fi hu no fa dz sr th is sv nl la eu hi ta ar sk cn gl sh lv ms et el nb mo hr ur pa tl ro sl ca iu hy bg ga si id uz mi bn lt sq
```

```
from wordcloud import WordCloud
# Create the wordcloud object
wordcloud = WordCloud(width=480, height=480, margin=0, max_font_size=20, min_font_size=10, colormap="Blues").generate(str)

# Display the generated image:
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.margins(x=0, y=0)
plt.show()
```



```
[ ] plt.figure(figsize=(25, 5))  # Задаем размер фигуры в дюймах
    sns.countplot(df, x="original_language")
```

```
<Axes: xlabel='original_language', ylabel='count'>
```

Теерь мы знаем список всех возможных оригинальных языков фильмов и частоту их употребления.

Давайте отобразим еще один категориальныый признак - жанры, которым принадлежат фильмы. Отобразим самые популярные жанры с помощью диаграммы древовидной карты.

```
[ ] df["genres"].unique()
```

```
array(['Animation, Family, Adventure, Drama, Comedy',
       'Animation, Comedy, Family, Adventure', 'Animation, Family', ...,
       'Animation, Fantasy, TV Movie, Family',
       'Drama, Comedy, Adventure, Animation, Fantasy, TV Movie, Family',
       'Drama, Comedy, Documentary, History, Animation'], dtype=object)
```

```
[ ] import numpy as np
    df["genres"].unique()
    genres = df["genres"].values
    print(genres[1])
    allgenres = []
    for i in genres:
      str = i.split()
      allgenres = np.concatenate([allgenres, str])
    allgenres
```

```
Animation, Comedy, Family, Adventure
array(['Animation,', 'Family,', 'Adventure,', ..., 'Documentary,',
       'Family', 'Animation'], dtype='<U32')
```

```
allgenresdf = pd.DataFrame(data = allgenres)
allgenresdf
```

	0
0	Animation,
1	Family,
2	Adventure,
3	Drama,
4	Comedy
...	...
94508	Fiction
94509	Animation,
94510	Documentary,
94511	Family
94512	Animation

94513 rows × 1 columns

```
uniqueGenres = allgenresdf[0].unique()
d = dict.fromkeys(uniqueGenres, 0)
for i in genres:
  str = i.split()
  for i in str:
    d[i] = d[i] + 1
d
```

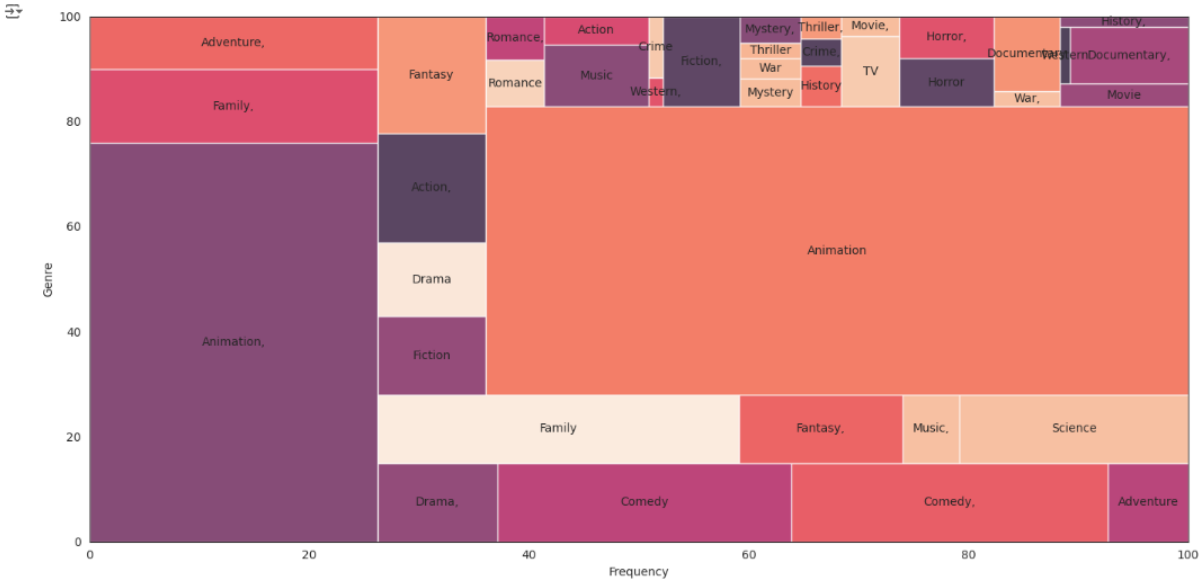
```
{'Animation,': 18801,
 'Family,': 3460,
 'Adventure,': 2512,
 'Drama,': 1547,
 'Comedy': 3788,
 'Comedy,': 4091,
 'Adventure': 1027,
 'Family': 4020,
 'Fantasy,': 1824,
 'Music,': 635,
 'Science': 2540,
 'Fiction': 1397,
 'Drama': 1302,
 'Action,': 1946,
 'Fantasy': 2072,
 'Animation': 33153,
 'Romance': 449,
 'Romance,': 417,
 'Music': 1062,
 'Action': 482,
 'Western,': 67,
 'Crime': 142,
 'Fiction,': 1143,
 'Mystery': 281,
 'War': 201,
 'Thriller': 156,
 'Mystery,': 262,
 'History': 272,
 'Crime,': 178,
 'Thriller,': 150,
 'TV': 679,
 'Movie,': 190,
 'Horror': 751,
 'Horror,': 642,
 'War,': 167,
 'Documentary': 807,
 'Movie': 489,
 'Western': 94,
 'Documentary,': 1090,
 'History,': 227}
```

```
pip install squarify
```

```
Collecting squarify
  Downloading squarify-0.4.3-py3-none-any.whl (4.3 kB)
Installing collected packages: squarify
Successfully installed squarify-0.4.3
```

```
[ ] import squarify

    plt.figure(figsize=(14, 7))
    squarify.plot(sizes=d.values(), label=d.keys(), alpha=.8 )
    plt.xlabel('Frequency')
    plt.ylabel('Genre')
    plt.tight_layout()
    plt.show()
```



Вывод

В ходе лабораторной работы я выполнил ряд задач по анализу и визуализации данных о фильмах. Я загрузил данные из CSV-файла в DataFrame, проанализировал пропущенные значения, построил различные графики и диаграммы для изучения взаимосвязей между показателями, распределения данных и частоты встречаемости различных атрибутов. В частности, я построил scatter plot, показывающий взаимосвязь между доходом и бюджетом фильмов, heatmap, отображающую распределение оценок и количества голосов, violin plot для визуализации распределения данных по столбцам "vote_count" и "vote_average", круговую диаграмму, показывающую распределение статусов фильмов, облако слов, визуализирующее частоту использования языков оригинала фильмов, гистограмму, отображающую количество фильмов по языкам оригинала, и диаграмму Treemap, показывающую частоту встречаемости различных жанров. Эти визуальные представления данных позволяют лучше понять структуру и особенности набора данных о фильмах.