# Introduction

## Team members

- Yvonne Gisser
- Dusan Resavac
- Roland Bauer

## General information

- First we created a single sweetviz-report for each data-file → 6 in total
- Using this reports we got first insights and a very good overview about the data available
- We then used the interesting insights and put them into this notebook
- additionally we submitted the link to this notebook and all sweetviz-reports on Moodle
- Link to published deepnote report: https://deepnote.com/@fh-dqda/SOEPython-b58ee753-d2eb-4fbf-a7b3-c86711836b6c

# Libraries

```
import pandas as pd
import matplotlib.pyplot as plt
```

# Read Files

```
ratings = pd.read_csv("ratings.csv")
movies = pd.read_csv("movies.csv")
tags = pd.read_csv("tags.csv")
links = pd.read_csv("links.csv")
genome_scores = pd.read_csv("genome-scores.csv")
genome_tags = pd.read_csv("genome-tags.csv")
```

# Movies

## Overview

Every movie has: **movieId**, **title** and **genres**.

```
movies.head(3)
```

| | movieId int64 | title object | genres object | |
|---|---|---|---|---|
| 0 | 1 | Toy Story (1995) | Adventure\|Animation\|Children\|Com... | |
| 1 | 2 | Jumanji (1995) | Adventure\|Children\|Fantasy | |
| 2 | 3 | Grumpier Old Men (1995) | Comedy\|Romance | |

## Size

In total we have 62423 movies.

```
movies.shape
```

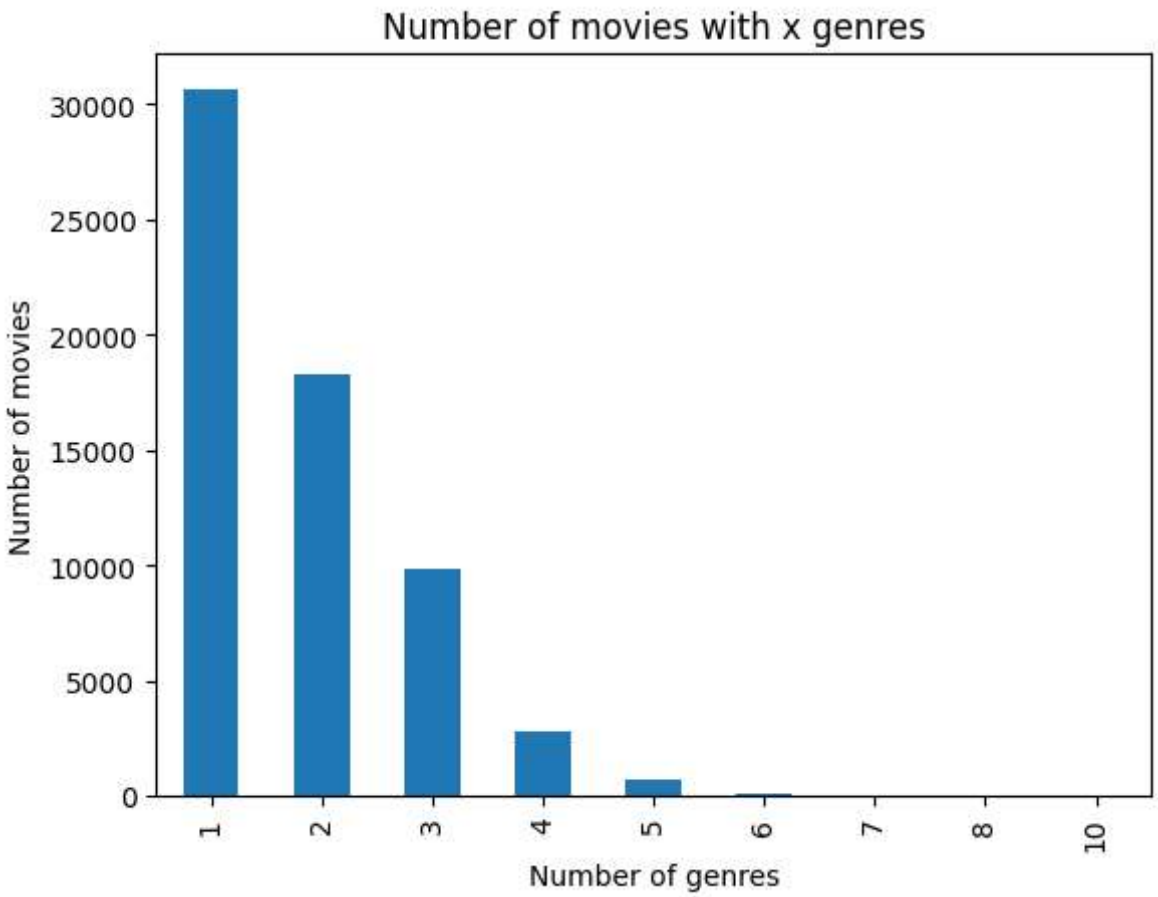
```
(62423, 4)
```

## Movie genres

To see how many genres movies normally have, we show a barchart with the number of genres per movie.

```
movies['genre_count'] = movies['genres'].apply(lambda x: len(x.split('|')))
movies['genre_count'].value_counts().sort_index().plot(kind='bar')
plt.title('Number of movies with x genres')
plt.xlabel('Number of genres')
plt.ylabel('Number of movies')
```

```
Text(0, 0.5, 'Number of movies')
```




# Ratings

## Overview

An entry in the ratings file has: **userId**, **movieId**, **rating** (ranging from 0.5 to 5.0) and timest**a**mp

```
ratings.head(3)
```

| | userId int64 | movieId int64 | rating float64 | timestamp int64 | |
|---|---|---|---|---|---|
| 0 | 1 | 296 | 5.0 | 1147880044 | |
| 1 | 1 | 306 | 3.5 | 1147868817 | |
| 2 | 1 | 307 | 5.0 | 1147868828 | |

## Size

In total we have 25.000.095 ratings

```
ratings.shape
```

```
(25000095, 4)
```

## Distribution of ratings

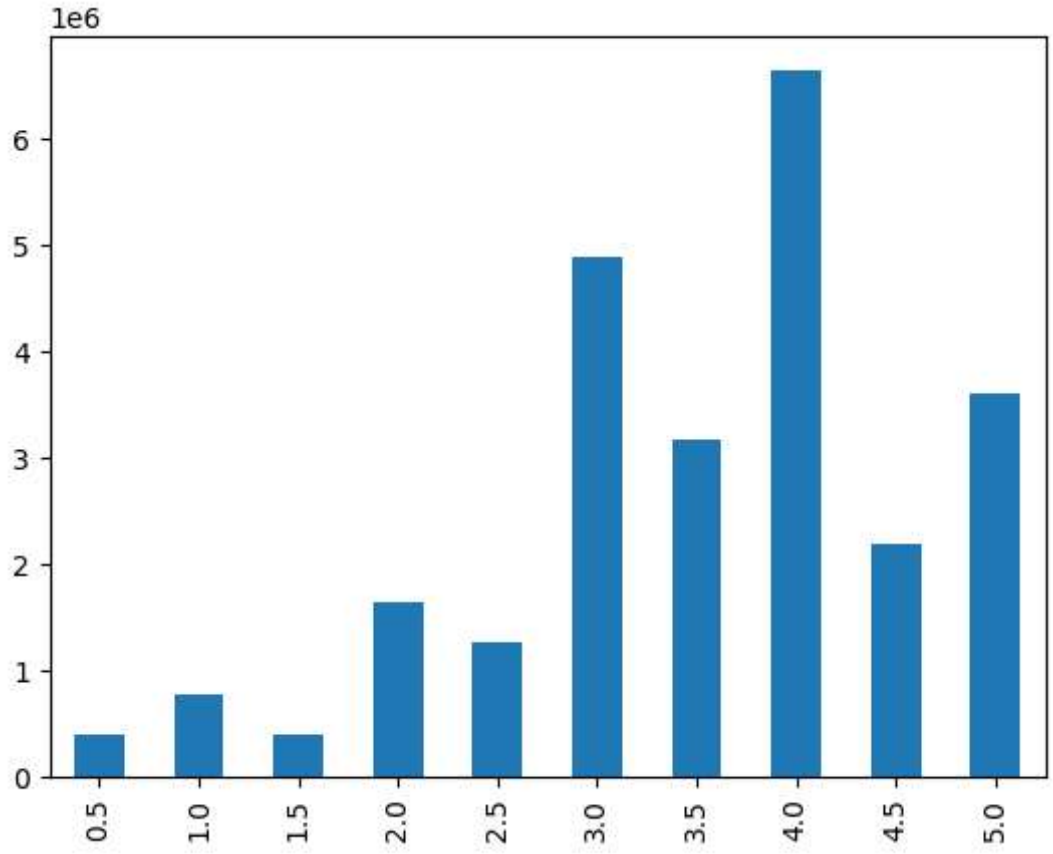
As a first overview we plot the distribution of ratings.

We can see that most ratings are a 4, followed by 3 and 5.

In general we see that "**half-ratings**" are less likely to occur/be given than "**full-point-ratings**".

```
ratings['rating'].value_counts().sort_index().plot(kind='bar')
```

```
<AxesSubplot: >
```




# Tags

## Overview

An entry in tags has **userId**, **movieId**, **tag** and **timestamp**

```
tags.head(3)
```

| | userId int64 | movieId int64 | tag object | timestamp int64 | |
|---|---|---|---|---|---|
| 0 | 3 | 260 | classic | 1439472355 | |
| 1 | 3 | 260 | sci-fi | 1439472256 | |
| 2 | 4 | 1732 | dark comedy | 1573943598 | |

## Size

```
print(f'Size of tags data: {str(tags.shape)}, with {len(tags["tag"].unique())} different t
```

```
Size of tags data: (1093360, 4), with 73051 different tags
```

## Frequency - most and least frequent tags

In the table below we can see top and low 10 used tags. Most used tag was **"sci-fi"**, while **"human alien"** was one of the tags which was used only once.

```
# list tags with the highest frequency
print(tags['tag'].value_counts().head(10))
print(tags['tag'].value_counts().tail(10))
```

```
sci-fi                8330
atmospheric           6516
action                5907
comedy                5702
surreal               5326
based on a book       5079
twist ending          4820
funny                 4738
visually appealing    4526
dystopia              4257
Name: tag, dtype: int64
search for child                          1
Writer: Erik Skjoldbjærg                  1
les requins zèbres ont bien un aileron    1
iranium                                   1
elders                                    1
instageek                                 1
Supernatural horror                       1
dads                                      1
```

```
screenwroter:Peter Hedges                1
human alien                              1
Name: tag, dtype: int64
```

# Genome Tags

Tag Genome (genome-scores.csv and genome-tags.csv) ----------

This data set includes a current copy of the Tag Genome.

[genome-paper]: http://files.grouplens.org/papers/tag_genome.pdf

The tag genome is a data structure that contains tag relevance scores for movies. The structure is a dense matrix: each movie in the genome has a value for *every* tag in the genome.

As described in [this article][genome-paper], the tag genome encodes how strongly movies exhibit particular properties represented by tags (atmospheric, thought-provoking, realistic, etc.). The tag genome was computed using a machine learning algorithm on user-contributed content including tags, ratings, and textual reviews.

```
genome_tags.head(3)
```

| | tagId int64 | tag object |
|---|---|---|
| 0 | 1 | 007 |
| 1 | 2 | 007 (series) |
| 2 | 3 | 18th century |

In total we have 1128 different genome tags listed.

```
len(genome_tags)
```

```
1128
```

In the **"genome_scores.csv"**-File are 13816 movies listed. For each **movie** we have a **relevance score** for each **genom tag**:

```
genome_scores.head(3)
```

| | movieId int64 | tagId int64 | relevance float64 |
|---|---|---|---|

| | | | | |
|---|---|---|---|---|
| 0 | 1 | 1 | 0.0287499999999999 | |
| 1 | 1 | 2 | 0.0237499999999999 | |
| 2 | 1 | 3 | 0.0625 | |

## Count Unique MovieIds with Genome Tags

```
genome_scores['movieId'].nunique()
```

13816

## Highest weighted genome tags

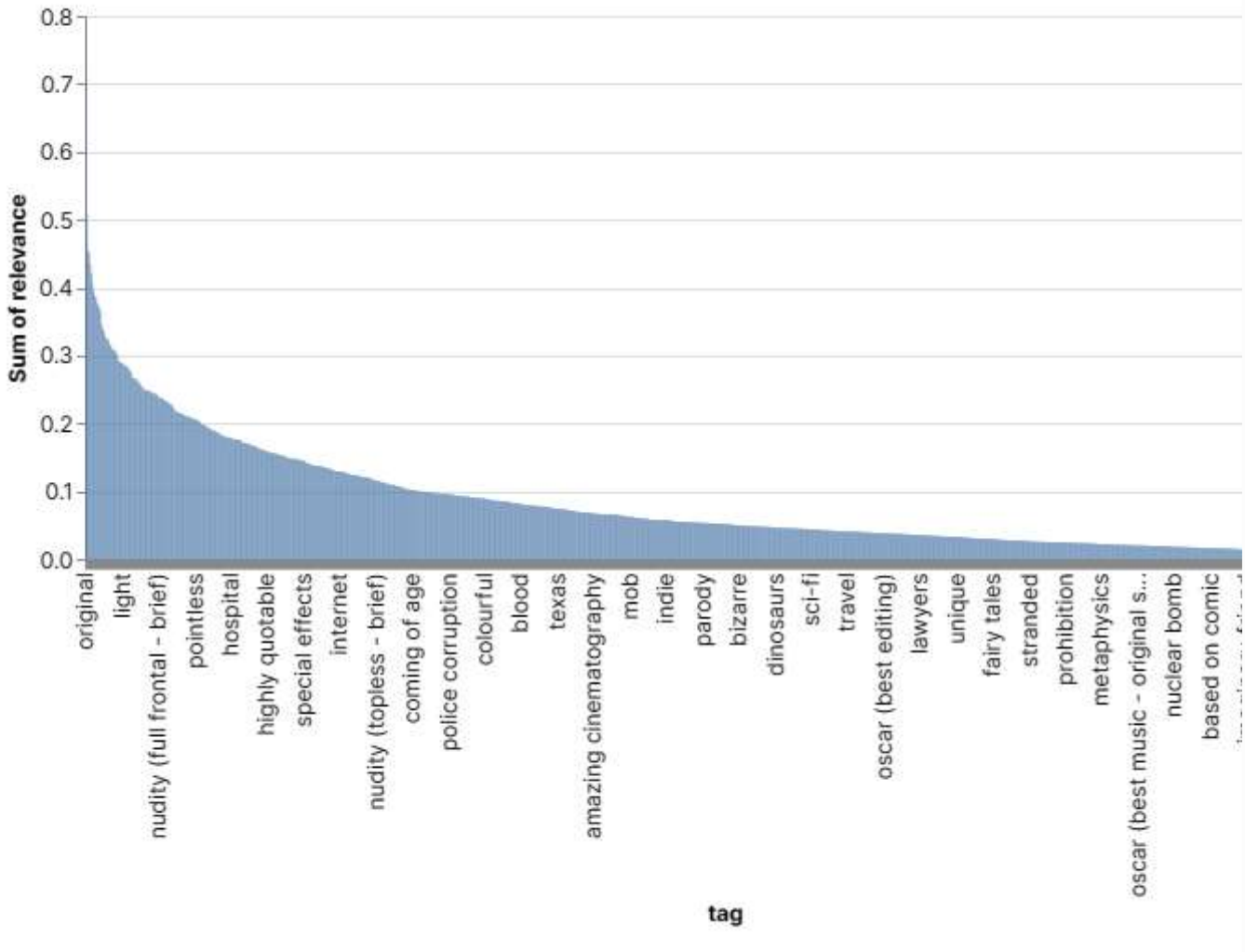
```
# group by tagId and calculate mean relevance score
scores_mean = genome_scores.groupby('tagId')['relevance'].median().sort_values(ascending=F

# join with genome_tags to get tag names
scores_mean = pd.merge(scores_mean, genome_tags, how='left', on='tagId')

scores_mean.sort_values(by='relevance', ascending=False)
```

| | tagId int64<br>1 - 1128 | relevance float64<br>0.0022499999999... | tag object<br>original ..... 0.1%<br>mentor ..... 0.1%<br>1126 others ..... 99.8% | |
|---|---|---|---|---|
| 0 | 742 | 0.7215 | original | |
| 1 | 646 | 0.508 | mentor | |
| 2 | 188 | 0.4529999999999999 | catastrophe | |
| 3 | 468 | 0.45175 | great ending | |
| 4 | 867 | 0.4337499999999999 | runaway | |
| 5 | 302 | 0.421 | dialogue | |
| 6 | 972 | 0.40475 | storytelling | |
| 7 | 452 | 0.39475 | good soundtrack | |
| 8 | 464 | 0.38775 | great | |
| 9 | 1070 | 0.3854999999999999 | vengeance | |

The following chart shows the mean relevance of each genome tag over all movies:




# Links

movieId is an identifier for movies used by <https://movielens.org>. E.g., the movie Toy Story has the link <https://movielens.org/movies/1>.

imdbId is an identifier for movies used by <http://www.imdb.com>. E.g., the movie Toy Story has the link <http://www.imdb.com/title/tt0114709/>.

tmdbId is an identifier for movies used by <https://www.themoviedb.org>. E.g., the movie Toy Story has the link <https://www.themoviedb.org/movie/862>.

```
links.head(5)
```

| | movieId int64 | imdbId int64 | tmdbId float64 |
|---|---|---|---|
| 0 | 1 | 114709 | 862.0 |
| 1 | 2 | 113497 | 8844.0 |

| | | | | |
|---|---|---|---|---|
| 2 | 3 | 113228 | 15602.0 | |
| 3 | 4 | 114885 | 31357.0 | |
| 4 | 5 | 113041 | 11862.0 | |