Data visualisation lab 5

Made by: Paulius Lapienis

In [ ]:
```
from labs.definitions import DATA_DIR
import pandas as pd

DATA_PATH = DATA_DIR / "anime_filtered.csv"
df = pd.read_csv(DATA_PATH)
df
```

Out[ ]:

	anime_id	title	title_english	title_japanese	title_synonyms	ima
0	11013	Inu x Boku SS	Inu X Boku Secret Service	妖狐×僕SS	Youko x Boku SS	https://myanimelidena.com/images/ar
1	2104	Seto no Hanayome	My Bride is a Mermaid	瀬戸の花嫁	The Inland Sea Bride	https://myanimelidena.com/images/ar
2	5262	Shugo Chara!! Doki	Shugo Chara!! Doki	しゅごキャラ！！どきっ	Shugo Chara Ninenme, Shugo Chara! Second Year	https://myanimelidena.com/images/ar
3	721	Princess Tutu	Princess Tutu	プリンセスチュチュ	NaN	https://myanimelidena.com/images/ar
4	12365	Bakuman. 3rd Season	Bakuman.	バクマン。	Bakuman Season 3	https://myanimelidena.com/images/ar
...	...	...	...	...	...	
14469	26089	Gutchonpa Omoshiro Hanashi	NaN	グッチョンパおもしろ話	NaN	https://myanimelidena.com/images/ar
14470	21525	Geba Geba Shou Time!	NaN	ゲバゲバ笑タイム!	NaN	https://myanimelidena.com/images/ar
14471	37897	Godzilla: Hoshi wo Kuu Mono	NaN	GODZILLA -星を喰う者-	Godzilla Part 3, Godzilla: Eater of Stars	https://myanimelidena.com/images/ar
14472	34193	Nippon Mukashibanashi: Sannen Netarou	NaN	日本昔ばなし 三ねん寝太郎	NaN	https://myanimelidena.com/images/ar
14473	37908	Senjou no Valkyria Special	NaN	戦場のヴァルキュリア Valkyria Chronicles	Senjou no Valkyria Fake Movie Promo	https://myanimelidena.com/images/ar

14474 rows × 31 columns

In [ ]:
```
df.columns
```

Out[ ]:
```
Index(['anime_id', 'title', 'title_english', 'title_japanese',
       'title_synonyms', 'image_url', 'type', 'source', 'episodes', 'statu
s',
       'airing', 'aired_string', 'aired', 'duration', 'rating', 'score',
       'scored_by', 'rank', 'popularity', 'members', 'favorites', 'backgrou
nd',
       'premiered', 'broadcast', 'related', 'producer', 'licensor', 'studi
o',
       'genre', 'opening_theme', 'ending_theme'],
      dtype='object')
```

In [ ]:
```
df.describe()
```

Out[ ]:

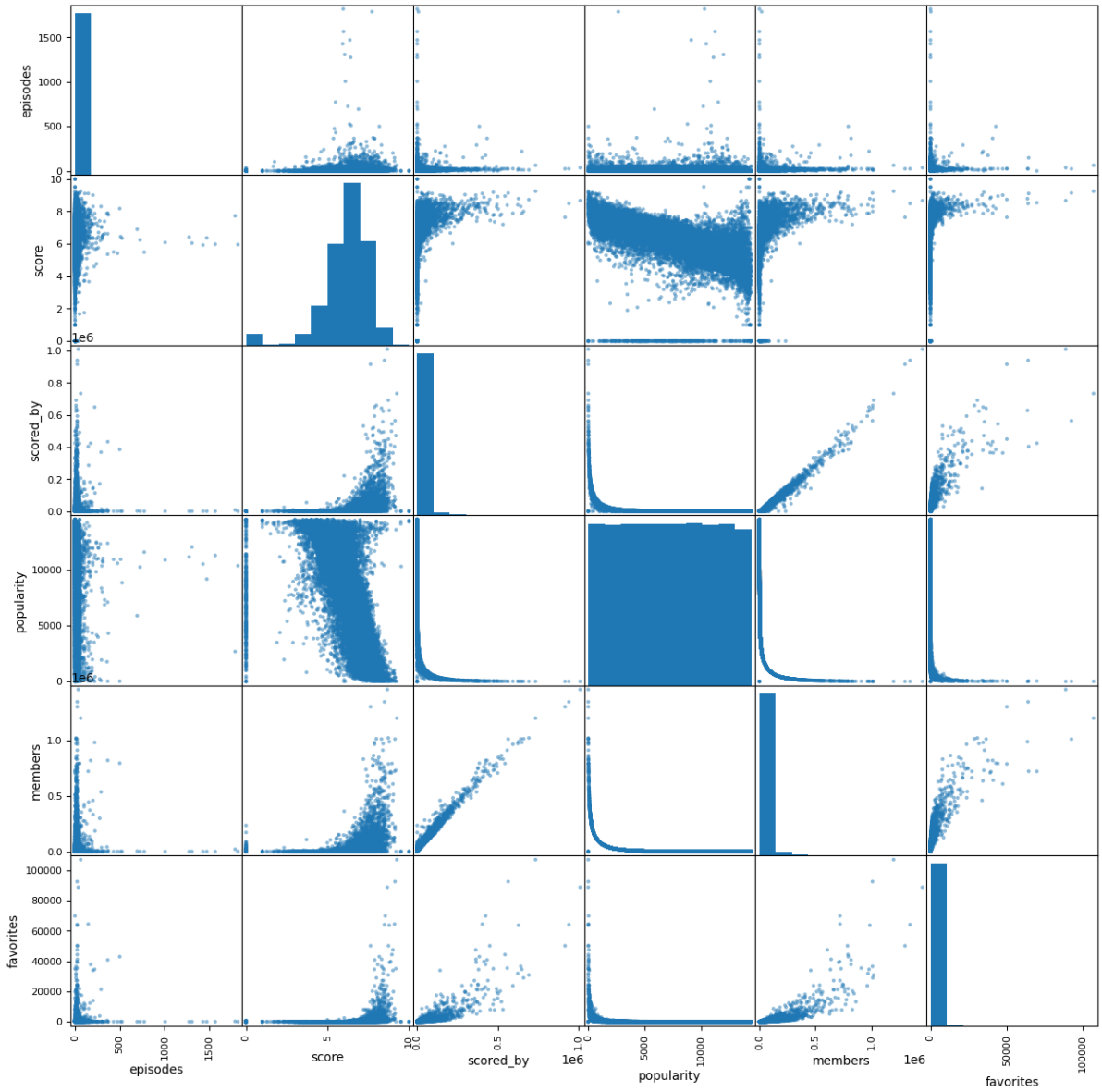
	anime_id	episodes	score	scored_by	rank	popularity	
count	14474.000000	14474.000000	14474.000000	1.447400e+04	12901.000000	14474.000000	1.4
mean	17371.948183	11.310971	6.144179	1.146319e+04	6439.625068	7220.277256	2.2
std	13163.266015	43.449161	1.460617	4.311072e+04	3719.462602	4168.959000	7.4
min	1.000000	0.000000	0.000000	0.000000e+00	0.000000	0.000000	0.0
25%	4387.500000	1.000000	5.550000	4.600000e+01	3218.000000	3613.250000	2.4
50%	15128.000000	1.000000	6.370000	5.010000e+02	6442.000000	7225.500000	1.6
75%	31142.000000	12.000000	7.060000	3.947250e+03	9664.000000	10826.750000	1.0
max	37916.000000	1818.000000	10.000000	1.009477e+06	12919.000000	14487.000000	1.4

Matrix of scatterplots is chosen as the first multidimensional data direct visualization method.

In [ ]:
```
import numpy as np
from sklearn import preprocessing
from function_pipes import pipe
from functools import partial


df.select_dtypes(include=np.number).drop(columns=["anime_id", "rank"])
pd.plotting.scatter_matrix(df.select_dtypes(include=np.number).drop(columns=
```

Out[ ]:
```
array([[<Axes: xlabel='episodes', ylabel='episodes'>,
        <Axes: xlabel='score', ylabel='episodes'>,
        <Axes: xlabel='scored_by', ylabel='episodes'>,
        <Axes: xlabel='popularity', ylabel='episodes'>,
        <Axes: xlabel='members', ylabel='episodes'>,
        <Axes: xlabel='favorites', ylabel='episodes'>],
       [<Axes: xlabel='episodes', ylabel='score'>,
        <Axes: xlabel='score', ylabel='score'>,
        <Axes: xlabel='scored_by', ylabel='score'>,
        <Axes: xlabel='popularity', ylabel='score'>,
        <Axes: xlabel='members', ylabel='score'>,
        <Axes: xlabel='favorites', ylabel='score'>],
       [<Axes: xlabel='episodes', ylabel='scored_by'>,
        <Axes: xlabel='score', ylabel='scored_by'>,
        <Axes: xlabel='scored_by', ylabel='scored_by'>,
        <Axes: xlabel='popularity', ylabel='scored_by'>,
        <Axes: xlabel='members', ylabel='scored_by'>,
        <Axes: xlabel='favorites', ylabel='scored_by'>],
       [<Axes: xlabel='episodes', ylabel='popularity'>,
        <Axes: xlabel='score', ylabel='popularity'>,
        <Axes: xlabel='scored_by', ylabel='popularity'>,
        <Axes: xlabel='popularity', ylabel='popularity'>,
        <Axes: xlabel='members', ylabel='popularity'>,
        <Axes: xlabel='favorites', ylabel='popularity'>],
       [<Axes: xlabel='episodes', ylabel='members'>,
        <Axes: xlabel='score', ylabel='members'>,
        <Axes: xlabel='scored_by', ylabel='members'>,
        <Axes: xlabel='popularity', ylabel='members'>,
        <Axes: xlabel='members', ylabel='members'>,
        <Axes: xlabel='favorites', ylabel='members'>],
       [<Axes: xlabel='episodes', ylabel='favorites'>,
        <Axes: xlabel='score', ylabel='favorites'>,
        <Axes: xlabel='scored_by', ylabel='favorites'>,
        <Axes: xlabel='popularity', ylabel='favorites'>,
        <Axes: xlabel='members', ylabel='favorites'>,
        <Axes: xlabel='favorites', ylabel='favorites'>]], dtype=object)
```



For the second method I chose a graph.

In [ ]:
```
from pyvis import network as net
import networkx as nx

columns = ["genre", "source"]
data = (
    df[columns]
    .dropna()
    .assign(genre=lambda df: df.genre.str.split(", "))
    .explode("genre")
)
data.isnull().sum()

g = nx.Graph()
G = net.Network(notebook=True, cdn_resources="remote")
for column in columns:
    for node in data[column].unique():
        g.add_node(node + column, title=node, group=column)
```

```
G.from_nx(g)
edges = []
for _, row in data.iterrows():
    # edges.append((row.studio, row.genre))
    edges.append((row.genre + "genre", row.source + "source"))

edges = list(set(edges))
pos = nx.circular_layout(g, scale=500)
for node in G.get_nodes():
    G.get_node(node)["x"] = pos[node][0]
    G.get_node(node)["y"] = -pos[node][
        1
    ]  # the minus is needed here to respect networkx y-axis convention
    G.get_node(node)["physics"] = False
    G.get_node(node)["label"] = str(
        g.nodes[node]["title"]
    )  # set the node label as a string so that it can be displayed
    G.get_node(node)["group"] = g.nodes[node]["group"]
G.add_edges(edges)

G.toggle_physics(False)
G.show("nx.html")
```

nx.html

Out[ ]: