**Importing libraries**

|  |  |
| --- | --- |
| import pandas as pd | library for data analysis |
| import plotly.express as px | library for nice interactive plots |
| import seaborn as sns | library for nice static plots |

**Handy pandas settings**

|  |  |
| --- | --- |
| pd.options.display.max\_columns = 100 | show more columns of dataframe in Jupyter |

**Reading and saving data**

|  |  |
| --- | --- |
| df = pd.read\_csv('most\_voted\_titles.csv', header=0, sep=',') | read csv, specifying it has headers in row 0 |
| df.to\_csv('new\_movies\_selection.csv', header=True, index=False) | save to csv, include header, don't include index |

**Inspecting data**

|  |  |
| --- | --- |
| df.head() | check first 5 rows |
| df.tail(3) | check last 3 rows |
| df.info() | compact summary of your dataframe |
| df.shape | nr of rows, nr of columns |
| df.columns | all column names |
| df.index | all row labels |
| df.isnull().sum() | count all null values in a dataframe |
| df.describe(include='all') | summary statistics of all columns |

**Basic manipulations**

|  |  |
| --- | --- |
| df.drop(columns=['primaryTitle']) | drop particular columns or rows |
| df['new\_metascore'] = df['metascore'] / 10. | example of creating new column |
| df = df.rename(columns={'startYear': 'start\_year'}) | rename columns |
| df.sort\_values(by='originalTitle', ascending=False) | sort rows on 1 column |
| df.sort\_values(by=['startYear', 'runtimeMinutes'], ascending=[False, True]) | sort rows on multiple columns |

**Making selections**

|  |  |
| --- | --- |
| df['startYear'] | select 1 column = pandas series |
| df.startYear | select 1 column (same as above) |
| df[['tconst', 'averageRating', 'startYear']] | select multiple columns using a list |
| df[df['averageRating'] > 9.0] | 1 condition to select rows |
| df[(df['titleType'] == 'movie') & (df['averageRating'] > 9.0)] | multiple conditions to select rows |
| df.query("titleType == 'movie' and averageRating > 9") | mulitiple conditions to select rows |
| df[df['genre1'].isin(['Crime', 'Drama'])] | conditions based on multiple values |
| df[df['originalTitle'].str.contains('godfather', case=False)] | get all rows where column has a certain text in it |

**Data wrangling**

|  |  |
| --- | --- |
| df.fillna('special value') | fill all null values with a special value |
| df.drop\_duplicates(keep='first') | drop duplicate rows |

**Plotting data**

|  |  |
| --- | --- |
| px.scatter(  title='runtime vs average rating',  data\_frame=df.query('runtimeMinutes < 400'),  x='runtimeMinutes',  y='averageRating',  color='titleType',  hover\_data=['primaryTitle'],  height=500,) | plotly interactive scatterplot |
| sns.scatterplot(data=df, x='runtimeMinutes', y='averageRating',  hue='titleType',) | seaborn static scatterplot |

**Aggregating and summarizing data**

|  |  |  |
| --- | --- | --- |
| df['endYear'].value\_counts(dropna=False, normalize=True) | count values of 1 column | |
| df.groupby(['startYear'], dropna=False, as\_index=False)[['averageRating']].mean() | | groupby column and calculate mean |

**Joining /merging dataframes**

|  |  |
| --- | --- |
| df\_movies.merge(df\_actors, on='title', how='left') | joining 2 tables just like in SQL |

**Jupyter Notebooks**

|  |  |
| --- | --- |
| Shift + Enter | execute cell |
| Escape en daarna `a` of `b` | insert new cell above of below |
| Escape en daarna `dd` | delete cell |
| Shift + Tab (inside a function) | see all arguments of the function |
| %ls | magic command: list all files of current dir |