Clustering

1. **Fundamentals**
    - Get ready to use the same dataset as for the previous class. Keep columns [budget, popularity, revenue, runtime, vote_average, vote_count] as numerical columns. Clean them by removing rows having undefined (nan) values. Remove 0 in budget , revenue, runtime, and vote_count.
    - Using sklearn library, applying KMeans algorithm on the dataframe with 3 clusters.
    - Compute the centroid (mean values for each feature), and the size for each cluster. A flexible way to proceed is to extract the clusters (fit_predict), add the resulting list as a new column (e.g., “cluster”) in a copy of the feature dataframe, then compute statistics by cluster in that dataframe, for instance with .groupby(“cluster”).agg([‘mean’, ‘count’]
    - Remember that k-mean is based on the notion of distance between points in the D-dimensional space. Thus is sensible to magnitudes. Standardize the feature with StandardScaler (fit_transform) and re-run K-means (be careful not to use the dataframe of the previous question that contains the previous clustering!)
    - To better observe the difference, we can plot (sns.scatterplot) the relation between a variable with large values (x=”budget”), a variable with low values (y=”vote_average”), with dot colors corresponding to clusters (hue=”cluster”). You should observe a clear difference between normalized and non-normalized version
    - If you had to give a manual label to those clusters, to describe the movies they contain, what would it be? (e.g.: “popular movies that people like”)
2. **Advanced**
    - On the class page, you can find a dataset from the website. Download it.
    - Apply a similar analysis on the dataset.