**PUNE INSTITUTE OF COMPUTER**

**TECHNOLOGY**



**Department of Computer Engineering**

(2021-2022)

**DSBDAL**

Movie recommendation model

**Group members:**

1. **31146**: Sanket Kulkarni
2. **31147**: Sourabh Kumbhar

**Guided by:** Prof. N. P. Jadhav

**Problem Statement:**

Develop a movie recommendation model using the scikit-learn library in python.

**Objectives:**

* To learn the working of scikit-learn library and all the related functions.
* To understand concepts such as Cosine similarity and Count-Vectorizer.
* To develop a model that judges similarities between entities on many factors. ● To analyze our model on the basis of techniques like similarity matrix.

**Theory:**

**AI Recommendation System:**

It is a model / engine that uses machine learning to predict the users’ choices and offer relevant suggestions to users. It filters and recommends the most suitable options to the users, hence aiding in the selection process of the user.

From a business standpoint, it promotes better customer engagement, thus resulting in higher sales. It also provides the customer with insights into similar products, thus increasing one’s field of vision while choosing a product.

**Scikit-learn:**

It is a free and open-source machine learning library for Python. It provides a variety of classification, regression and clustering algorithms including support-vector machines, random forests, gradient boosting, k-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

**Difflib:**

This module provides classes and functions for comparing sequences. It can be used for example, for comparing files, and can produce information about file differences in various formats, including HTML and context and unified diffs.

In this program, we use the get\_close\_matches() function that returns a list of matches that have a high accuracy to the search parameter.

**CountVectorizer:**

It converts a collection of text documents to a matrix of token counts. This implementation produces a sparse representation of the counts using scipy.sparse.csr\_matrix.

If you do not provide an a-priori dictionary and you do not use an analyzer that does some kind of feature selection then the number of features will be equal to the vocabulary size found by analyzing the data.

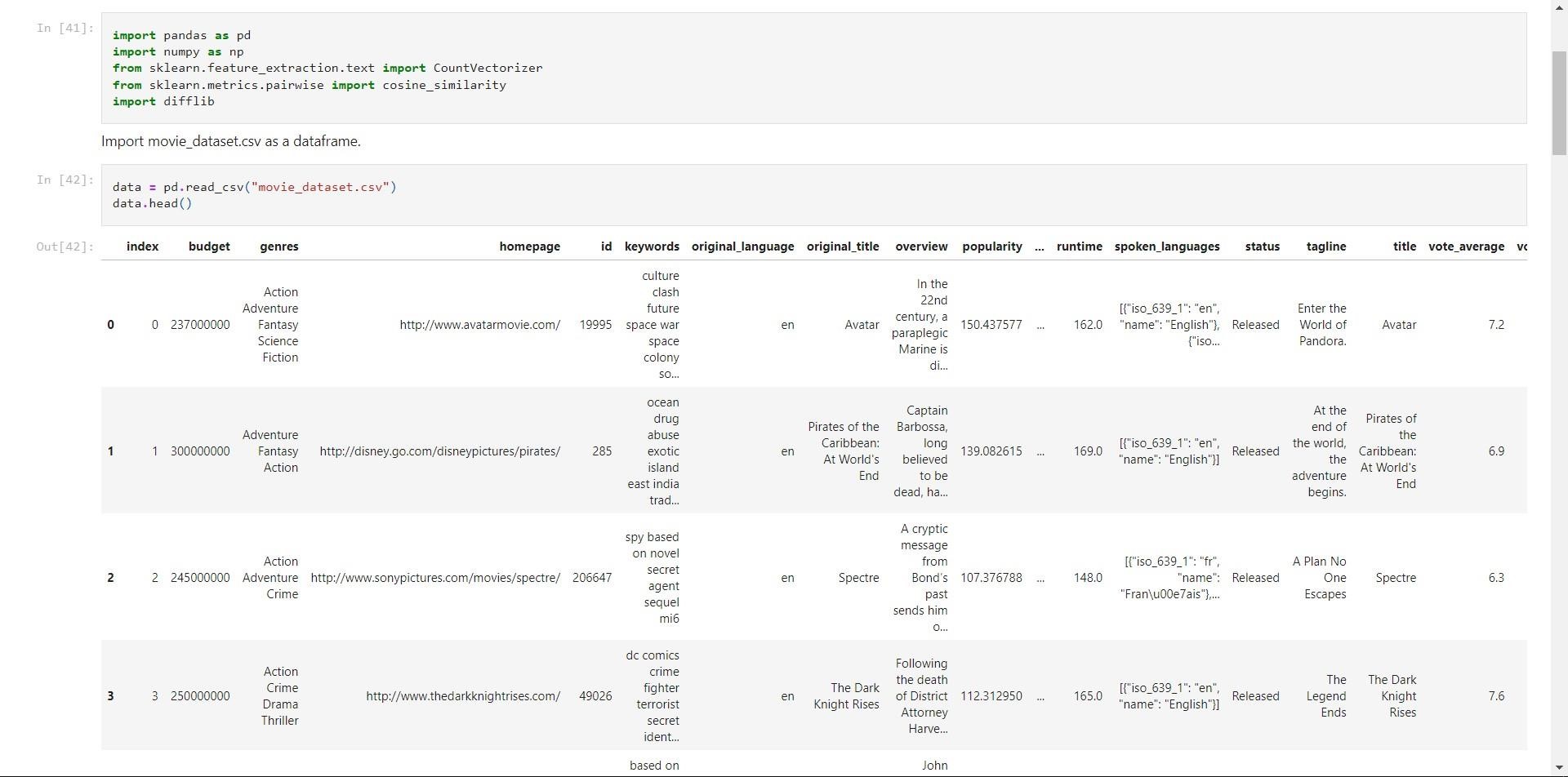
**Cosine\_Similarity:**

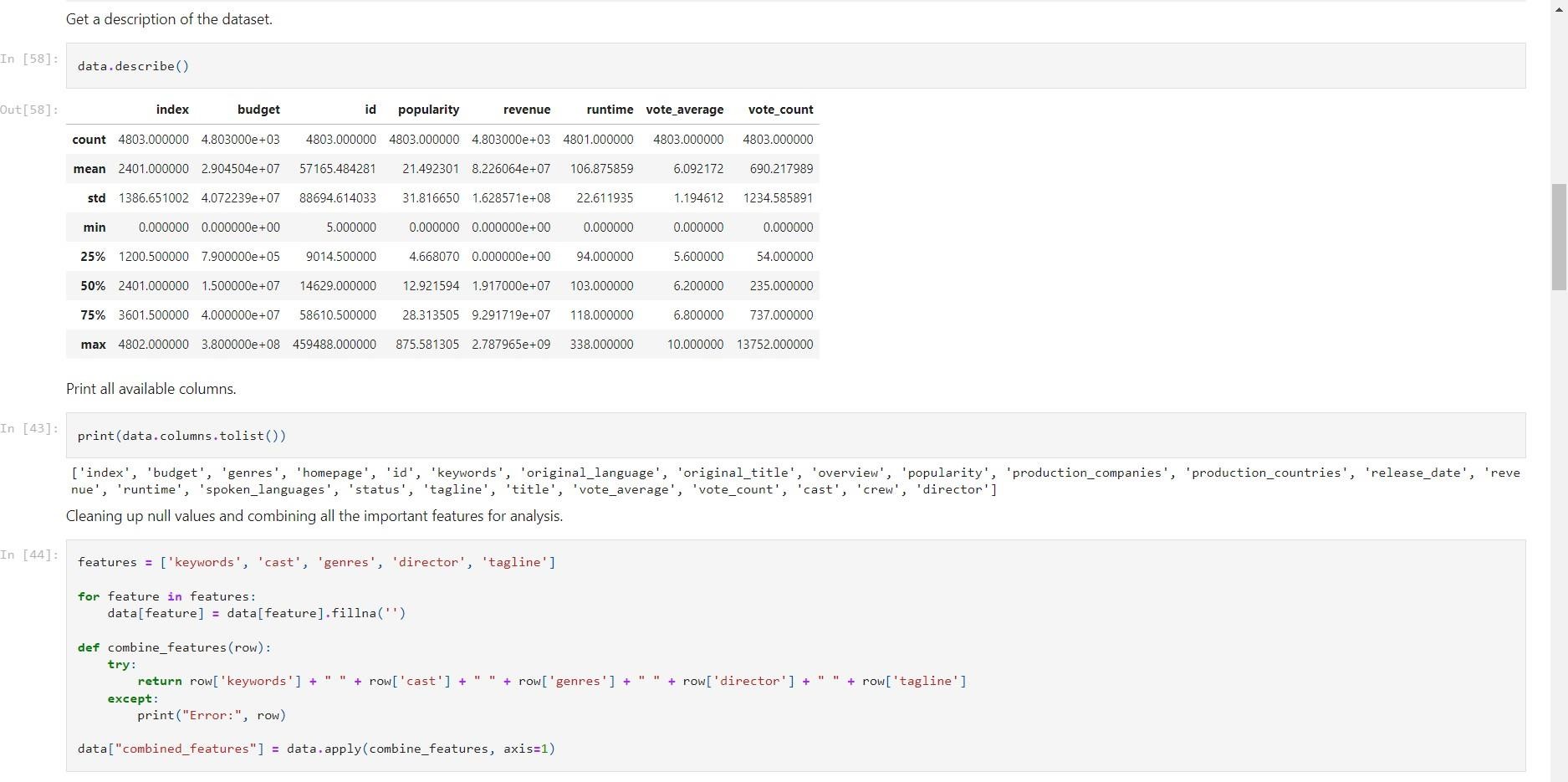
It computes cosine similarity between samples in X and Y. Cosine similarity, or the cosine kernel, computes similarity as the normalized dot product of X and Y:

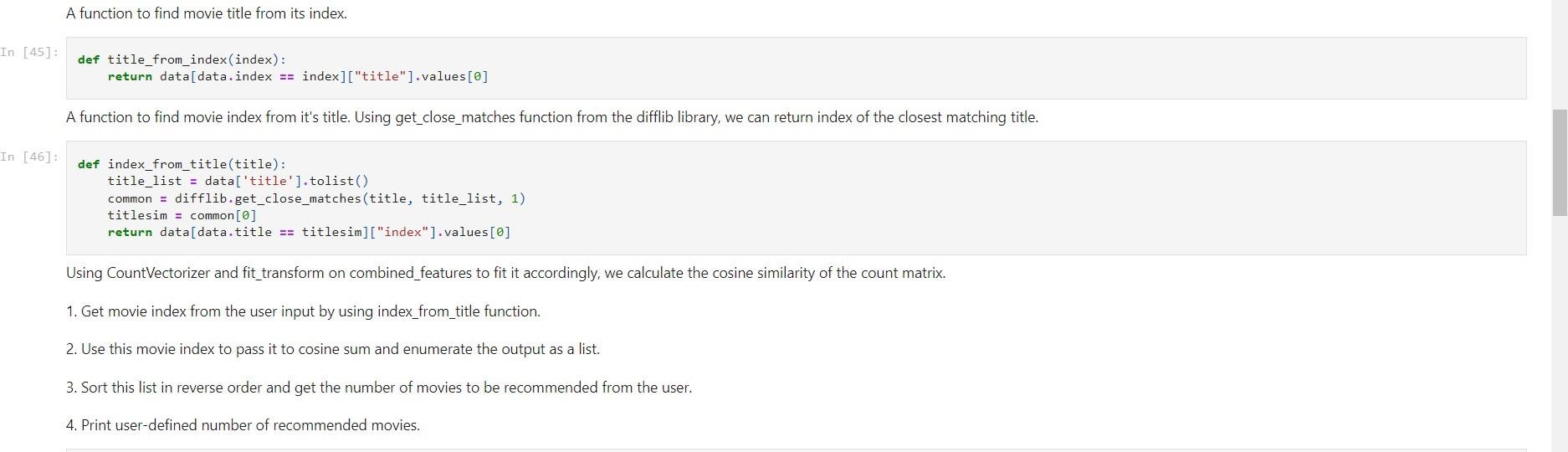
K(X, Y) = <X, Y> / (||X||\*||Y||)

On L2-normalized data, this function is equivalent to linear\_kernel.

**Implementation:**







**Conclusion:**

Hence we successfully implemented a movie recommendation system using scikit-learn library in Python.