**Programming Assignment-2**

## 2242-CSE-6363-002-MACHINE LEARNING

**Swarag Reddy Pingili – 1002158023**

**REPORT**

## INTRODUCTION

**Semantic Search:**

Semantic search is an information retrieval technique that aims to retrieve the closest data to a search query. In sematic search the entire data is converted in the form of vectors and stored in a vector database like elastic search.

**Steps involved in Sematic Search**:

* **Data Preprocessing**: The first step is to pre process the data. This includes filling missing values and combining all the columns that will be used for embedding.
* **Embedding the data**: The next step is to embed the combined column by using a sentence transformer model.
* **Storing in a database**: The next step is to create an index and store all the embedded data into that database. We can use any vector search database.
* **Query Processing**: The next step is to take input from a user and embedding the input by using the same model. After embedding, compare it with the database and find out the most similar data.
* **Return Results:** The final step is to return the closest data to the search query by comparing it with the vector database.

**Datasets Description:**

* **IMDB\_Top\_1000:** There are a total of 1000 instances.

The different attributes in the IMDB\_Top dataset are:

1. Poster\_Link: This attribute provides a link to the movie poster.
2. Series\_Title: This attribute gives the title of the movie. This is a string value.
3. Released\_year: This shows the year the movie was released in.
4. Certificate: Shows the certificate
5. Runtime: Gives the total runtime of the movie in minutes
6. IMDB\_Rating: This attribute gives the rating of the movie.
7. Meta\_score: This shows the score earned by the movie.
8. Director: Gives the name of the director of the movie.
9. Star1, Star2, Star3, Star4: These attributes give the names of the actors in the movie.
10. No\_of\_votes: Total number of votes for the move on imdb.
11. Gross: Total money earned by the movie.

**IMPLEMENTATION:**

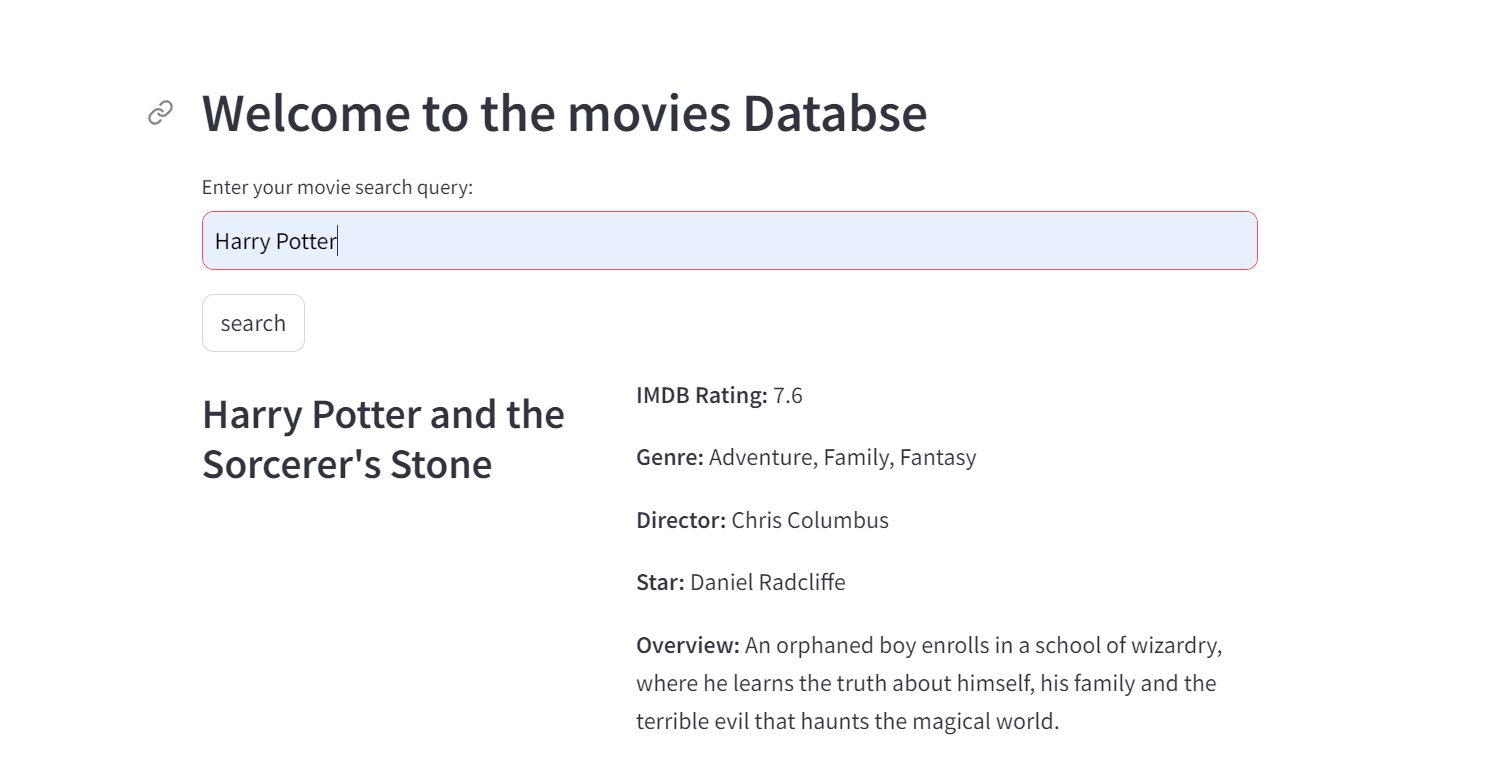
Different files used:

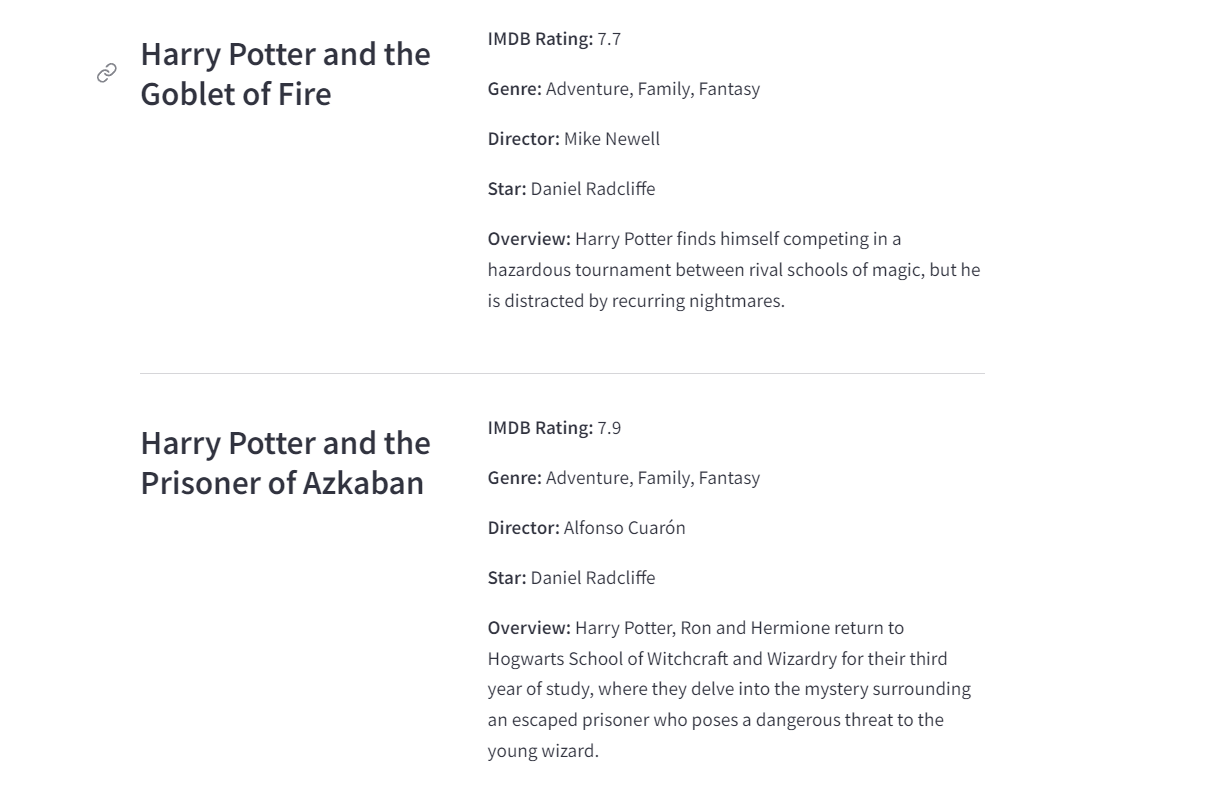
1. clean\_data.py: This file reads the csv file by using pandas and then cleans the dataset. This is done by filling in the missing values.
2. embed.ipynb: This file is used to initialize a model by using sentence transformer. All the required columns are combined and embedded into vectors using the model. Then these vectors are stored in a vector search database like faiss.

The index of faiss is converted into a pickle file for later usage.

1. faiss.pkl: This pickle file has the index which contains all the vector data. This is imported into the seach.py file to compare with a search query given by a user.
2. search.py: This file is a streamlit app. The app contains a search bar where user can enter a search query. This search query is embedded using the same model and converted into a vector. This vector is then compared with the database to find and return the closest results.

**Results:**





The above images show the final streamlit app. In the streamlit app there is a search bar. Using the search bar a user can enter any queries related to movies. The input is taken from the search bar and then compared to the vector database and the closest results are displayed. In the figure when Harry Potter is entered the movies closest to Harry Potter are displayed.

**REFRENCES USED:**

I Have used the below resources:

# Faiss Documentation:

# <https://faiss.ai/index.html>

# Sentence Transformers Documentation:

# <https://sbert.net/>

# Sematic Search Introduction:

# <https://www.elastic.co/what-is/semantic-search> Professor's GitHub repository:

# https://github.com/jesusglzbernal/semantic\_search