# Project:

## **NETFLIX Movie Recommender System**

## **Using Machine Learning:**

My project is on **Recommendation Systems**.  
For example, if I search for "Samsung" on Amazon, Amazon will recommend the latest Samsung products.

There are two types of recommendation systems:

1. **Context-based**
2. **Collaborative-based**

My project focuses on **context-based recommendation systems**.

url=https://youtu.be/kuC38ZCcbZI?si=qZ9sZXom3KE22FIF

**📌 Step-by-Step Explanation**

**🔹 Line [16]**

python

CopyEdit

from sklearn.feature\_extraction.text import CountVectorizer

* **Purpose**: Import the CountVectorizer class.
* **Why**: It's used to convert a collection of text documents to a **matrix of token counts** (also called Bag-of-Words model).

**🔹 Line [17]**

python

CopyEdit

cv = CountVectorizer(max\_features=10000, stop\_words='english')

* **What it does**: Creates a CountVectorizer object named cv.
* **Parameters**:
  + max\_features=10000: Only the top 10,000 most frequent words will be kept.
  + stop\_words='english': Common English words like “the”, “is”, “in” are ignored.

**🔹 Line [18]**

python

CopyEdit

cv

* **What it does**: Just outputs the configuration of the CountVectorizer object.
* **Output**:

scss

CopyEdit

CountVectorizer(max\_features=10000, stop\_words='english')

**🔹 Line [20]**

python

CopyEdit

vector = cv.fit\_transform(new\_data['tags'].values.astype('U')).toarray()

* **Purpose**: Converts the text data in the tags column of the new\_data DataFrame into a **numerical matrix**.
* **Explanation**:
  + new\_data['tags']: Takes the 'tags' column from the dataset.
  + .values: Converts it to a NumPy array.
  + .astype('U'): Converts all entries to Unicode string (useful for text).
  + cv.fit\_transform(...):
    - **fit**: Learns the vocabulary (i.e., builds a dictionary of words).
    - **transform**: Transforms the text into a matrix of token counts.
  + .toarray(): Converts the result into a regular NumPy array.

**🔹 Line [21]**

python

CopyEdit

vector.shape

* **What it shows**: The shape (size) of the resulting matrix.
* **Output**:

scss

CopyEdit

(10000, 10000)

* **Meaning**:
  + There are **10,000 rows** → 10,000 text documents (probably rows in the tags column).
  + There are **10,000 columns** → Each column is a word (token) in the top 10,000 vocabulary.

**📌 Summary**

| **Step** | **Action** | **Explanation** |
| --- | --- | --- |
| 1 | Import CountVectorizer | To convert text to numbers |
| 2 | Create cv object | Limit to 10,000 most frequent words; ignore English stop words |
| 3 | Use .fit\_transform() | Learn vocabulary and transform tags into number form |
| 4 | Use .shape | Shows that the matrix has 10,000 rows and 10,000 word columns |

**🎬 Netflix Movie Recommender System – Case Study and Line-by-Line Explanation**

**🔹 Step 1: Import Libraries**

python

CopyEdit

import pandas as pd

from sklearn.feature\_extraction.text import CountVectorizer

from sklearn.metrics.pairwise import cosine\_similarity

import pickle

import ipywidgets as widgets

from IPython.display import display

* **pandas**: Handles data in table format (CSV file).
* **CountVectorizer**: Converts movie descriptions into numbers.
* **cosine\_similarity**: Finds how similar two movies are.
* **pickle**: Saves and loads Python objects to/from files.
* **widgets**: Creates interactive input boxes and buttons.
* **display**: Shows widgets in notebook or Colab.

**🔹 Step 2: Load Dataset**

python

CopyEdit

try:

movies = pd.read\_csv('/content/sample\_data/top10K-TMDB-movies.csv')

except FileNotFoundError:

print("Dataset not found...")

exit()

* Tries to read the CSV file.
* If not found, shows an error and exits.

**🔹 Step 3: Select Useful Columns**

python

CopyEdit

movies = movies[['id', 'title', 'genre', 'overview']]

movies['tags'] = movies['overview'].fillna('') + ' ' + movies['genre'].fillna('')

* Keeps only the id, title, genre, and overview columns.
* Combines overview + genre into a new column called tags to describe each movie in text form.

**🔹 Step 4: Prepare Data**

python

CopyEdit

new\_data = movies.drop(columns=['genre', 'overview'])

* Keeps only id, title, and tags in the final dataset.

**🔹 Step 5: Convert Text to Numbers**

python

CopyEdit

cv = CountVectorizer(max\_features=10000, stop\_words='english')

vector = cv.fit\_transform(new\_data['tags'].values.astype('U')).toarray()

* CountVectorizer: Turns each movie’s tags into a numerical vector.
* Removes common English words (like "the", "is").
* Each movie becomes a list of word counts.

**🔹 Step 6: Calculate Similarity**

python

CopyEdit

similarity = cosine\_similarity(vector)

* Compares all movies to each other.
* Stores how similar each pair of movies is (value between 0 and 1).

**🔹 Step 7: Save Data for Later Use**

python

CopyEdit

pickle.dump(new\_data, open('movies\_list.pkl', 'wb'))

pickle.dump(similarity, open('similarity.pkl', 'wb'))

* Saves new\_data and similarity to .pkl files (used later for fast loading).

**🔹 Step 8: Define Recommendation Function**

python

CopyEdit

def recommand(movie\_title):

...

* Checks if the input movie exists.
* Finds its index.
* Looks at its similarity with all other movies.
* Shows top 5 similar movies (excluding itself).

**🔹 Step 9: Create UI Widgets (Input & Button)**

python

CopyEdit

movie\_input = widgets.Text(...)

recommend\_button = widgets.Button(...)

output = widgets.Output()

* Input box for user to type movie name.
* Button to trigger recommendation.
* Output area to show results.

**🔹 Step 10: Button Click Event**

python

CopyEdit

def on\_button\_clicked(b):

...

* When the button is clicked:
  + Clears old output.
  + Calls recommand() with the user’s movie title.

**🔹 Step 11: Display the Widgets**

python

CopyEdit

display(movie\_input, recommend\_button, output)

* Shows the input box, button, and output area in the notebook/Colab.

Website link:

https://snazzy-elf-529231.netlify.app/