Movie Recommendation System

OVERVIEW

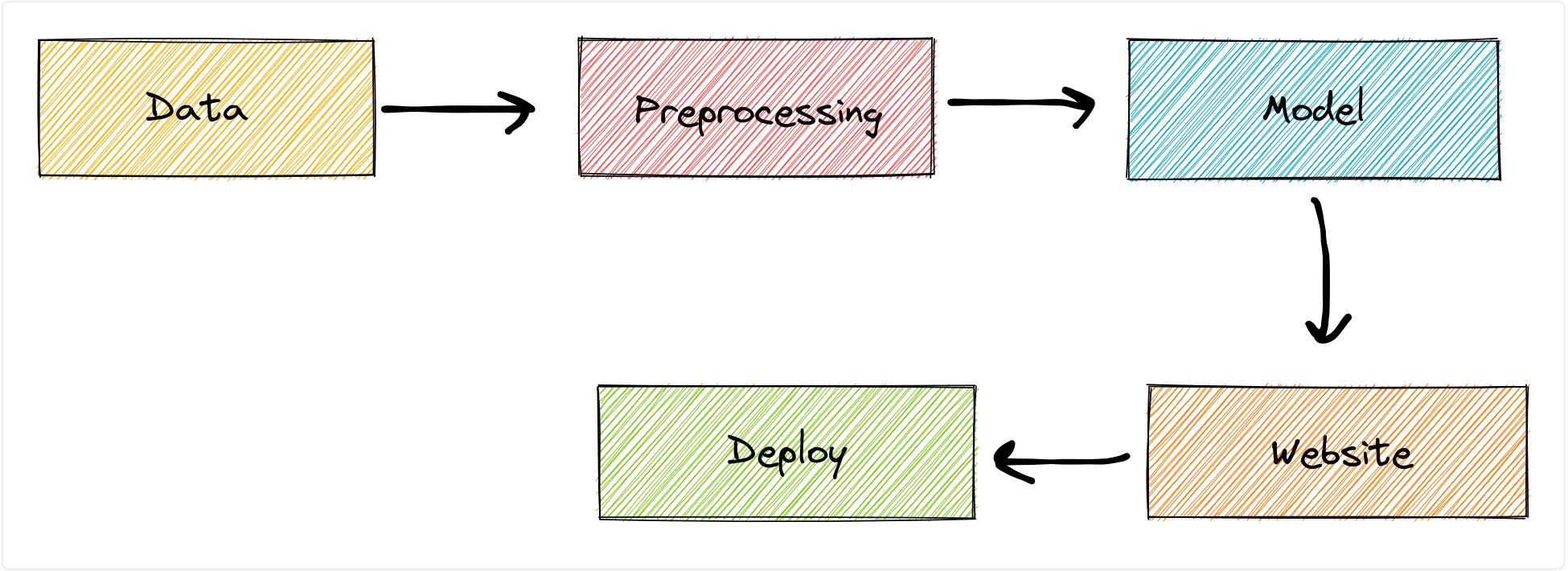
This project aims to demonstrate various movie recommendation algorithms used by major platforms like Netflix and Amazon Prime.

Métflix is a recommendation engine of movies from the given dataset.
For educational purposes, the project shall begin with the most basic algorithms or techniques used in a recommendation, and progressively build upon them to implement more advanced and sophisticated techniques.
Finally, the project will enable a user to get movie recommendations through content based filtering.

WORKING

The user interface is meant to be as self-descriptive as possible.

PROJECT FLOW



DATASETS

The datasets have been taken form Kaggle. There are two sets of data in the TMDB 5000 Movie Dataset, movies data and credits data.

https://www.kaggle.com/datasets/tmdb/tmdb-movie-metadata?select=tmdb_5000_movies.csv
https://www.kaggle.com/datasets/tmdb/tmdb-movie-metadata?select=tmdb_5000_credits.csv

EXPLORATORY DATA ANALYSIS

Before preprocessing we clean data and take out only the required and necessary data that can be worked upon on.

Firstly libraries such as NumPy, ast and Pandas are imported along with the above mentioned data.
Then both the datasets are merged on the basis of the title of the movie.
Then unnecessary columns which cannot be used as tags are dropped.

PREPROCESSING

Dataset is checked for null values and then the rows with null values are dropped.

Dataset is checked for duplicate values and then the duplicate entry is removed.

The dataset is simplified on into fewer columns(3).

Various further steps are performed to simplify the data.

TEXT VECTORIZATION

Converting tags to vectors using bag of words technique.

Class PorterStemmer is imported and a helper function is created to convert
string to list and then stemming each word.

CountVectorizer class imported as cv to covert a collection of text documents to a matrix of token counts.

Then tags are explicitly converted into NumPy array as cv returns sci-py sparks matric.

MODEL - COSINE SIMILARITY

Here we are dealing with high dimensions of data, so Euclidian distance fails, hence we use cosine similarity model here.

Similarity matrix is an array of arrays whose diagonal will always be one.

A list of tuples is generated and then a sorted list in reverse order on the basis of cosine distance.

A recommender function is created that will take input a movie and will return 5 similar movies to it.

```
def recommend(movie):
        #fetching the index
        movie_index = new_df[new_df['title'] == movie].index[0]
        distances = similarity[movie_index]
        movies_list = sorted(list(enumerate(distances)),reverse = True, key = lambda x:x[1])[1:6]

        for i in movies_list:
            print(new_df.iloc[i[0]].title)
```

Pickle module is imported. dump() function to store the object data to the file.

WEBSITE

The website is made using streamlit using pycharm as an IDE.

The code for this is given in https://github.com/jyotsnad246/Movie-Recommendation-Web-App/blob/0fc32ac99318b289dd869c5aa65cba93b56de35a/app.py .

DEPLOYMENT

The web application is deployed using Heroku

The link for the same is https://movie-recommendation-web-app.herokuapp.com/