**Project Report**

**Movie Recommendation System Using TF-IDF and Cosine Similarity**

**1. Project Title:**

**Content-Based Movie Recommendation System**

**2. Objective:**

**To develop a movie recommendation engine that suggests movies similar to a given title based on the content (overview and genre) using natural language processing (NLP) techniques. The goal is to enhance user experience by providing personalized recommendations.**

**3. Tools and Technologies Used:**

|  |  |
| --- | --- |
| **Technology** | **Purpose** |
| **Python** | **Programming language** |
| **Pandas** | **Data manipulation and preprocessing** |
| **Scikit-learn** | **TF-IDF vectorization, cosine similarity computation** |
| **SciPy** | **Efficient sparse matrix handling** |
| **Jupyter Notebook / IDE** | **Code development and testing** |

**4. Dataset Description:**

**- Dataset Name: IMDb Top 1000 Movies  
- Source: Kaggle / IMDb  
- Format: CSV  
- Number of Records: 1000  
- Key Columns Used:  
 • Series\_Title – Movie Title  
 • Overview – Short description of the movie  
 • Genre – Genre(s) of the movie  
 • IMDB\_Rating – IMDb rating score**

**5. Methodology:**

**5.1 Data Preprocessing**

**- Loaded the CSV file into a Pandas DataFrame.  
- Handled missing values in the Overview and Genre columns by replacing them with empty strings.  
- Created a new feature Combined by merging Overview and Genre for richer textual information.**

**5.2 Feature Extraction Using TF-IDF**

**- Applied TF-IDF Vectorizer to the Combined text field.  
- Removed common English stopwords.  
- Converted the text into a TF-IDF matrix where each row represents a movie and each column represents a term.**

**5.3 Similarity Calculation**

**- Calculated cosine similarity between all movie pairs using the TF-IDF matrix.  
- Stored the result as a sparse matrix for memory efficiency.**

**5.4 Recommendation Logic**

**- Created a mapping of movie titles to their corresponding index in the dataset.  
- For a given movie title:  
 • Retrieved its index.  
 • Fetched similarity scores with all other movies.  
 • Sorted and selected the top N most similar movies (excluding the movie itself).  
- Returned the recommended movies along with their Series\_Title, Genre, and IMDB\_Rating.**

**6. Implementation Code (Summary):**

***# TF-IDF Vectorization  
tfidf = TfidfVectorizer(stop\_words='english')  
tfidf\_matrix = tfidf.fit\_transform(data['Combined'])  
  
# Cosine Similarity Matrix  
cosine\_sim = cosine\_similarity(tfidf\_matrix, tfidf\_matrix)  
cosine\_sim = csr\_matrix(cosine\_sim)  
  
# Recommendation Function  
def get\_recommendations(title, num\_recommendations=10):  
 idx = indices.get(title)  
 sim\_scores = list(enumerate(cosine\_sim[idx].toarray().flatten()))  
 sim\_scores = sorted(sim\_scores, key=lambda x: x[1], reverse=True)  
 sim\_scores = sim\_scores[1:num\_recommendations+1]  
 movie\_indices = [i[0] for i in sim\_scores]  
 return data.iloc[movie\_indices][['Series\_Title', 'Genre', 'IMDB\_Rating']]***

**7. Sample Output:**

**For the input "12 Angry Men", the system returns:  
  
| Recommended Movie | Genre | IMDB Rating |  
|--------------------------|------------------------------|-------------|  
| To Kill a Mockingbird | Crime, Drama | 8.2 |  
| The Verdict | Drama | 7.7 |  
| A Few Good Men | Drama, Thriller | 7.7 |  
| Schindler’s List | Biography, Drama, History | 8.9 |  
| 12 Years a Slave | Biography, Drama, History | 8.1 |  
| The Green Mile | Crime, Drama, Fantasy | 8.6 |  
| Dead Poets Society | Drama | 8.1 |  
| The Godfather | Crime, Drama | 9.2 |  
| The Shawshank Redemption | Drama | 9.3 |  
| The Intouchables | Biography, Comedy, Drama | 8.5 |**

**8. Conclusion:**

**This project demonstrates the application of natural language processing and machine learning in building an intelligent, content-based movie recommendation system. By analyzing movie overviews and genres using TF-IDF and cosine similarity, the system effectively recommends relevant titles, enhancing user discovery and engagement.**

**- Fast and efficient for content-based filtering.  
- Works well without requiring user interaction or ratings.  
- Easy to scale for small to medium datasets.**