Experiment No. 7
Implementation of a recommendation system using Hybrid approach
Date of Performance:
Date of Submission:
Marks:
Sign:

Aim: Implementation of a recommendation system using Hybrid approach

Objective: Able to implement and understand the aspects of Hybrid Recommendation System.

Theory:

Hybrid recommendation systems bring the best of both worlds together. Hybrid mechanisms to predict recommendation, as the name implies, combine two or more recommendation techniques. The disadvantages of a recommendation methodology can be resolved using hybrid technology, and the benefits of various recommendation techniques can be summed up. The hybrid recommendation methodology to develop a search model for selecting the best cloud service provider. It recommended service providers close to the organization to leverage the enterprise location to boost bandwidth and eliminate latency difficulties.

In addition, this work proposes a search model, recommendation system and evaluation standard for customers based on their needs and location. According to the implementation results, cloud servers must be located close to the enterprise in order to improve bandwidth and reduce latency. This framework tracks the enterprise's location, and the recommended function recommends the closest cloud server to the enterprise. Two models are used to classify the essential factors used for the searching function: the basic and feature models. These models can make it easier for enterprises to find what they need in the system. In, the author discusses how recommendation systems play an important role in providing adequate information to users and filtering data. The various recommendation techniques, such as Collaborative Filtering, demographic-based, and knowledge-based, all have drawbacks. While opting for recommendation systems, these techniques fail to provide effective recommendations. As a result, it is necessary to optimize these techniques by identifying more distinct features. This is possible by integrating the advantages of various techniques in a hybrid manner. The author proposed an effective hybrid technique for book recommendations that uses ontology for user profiling to improve system efficiency. The results show that combination of Collaborative Filtering, knowledge-based techniques, and demographic attributes into a Hybrid technique produces the best recommendations.

A hybrid recommendation the framework uses a combination of content-based recommendation system(CBRS) and collaborative filtering recommendation systems (CFRS) to achieve precisely performance by reducing the drawbacks of the conventional recommendation techniques.

Implementation:

```
import pandas as pd
import numpy as np
credits = pd.read_csv ('./tmdb_5000_credits.csv')
movies_df = pd.read_csv
('./tmdb_5000_movies.csv')
print (f'Credits : {credits.shape}')
print (f'Movie Dataset : {movies_df.shape}')
credits_column_renamed = credits.rename
(index=str, columns = {'movie_id' : 'id'})
movies_df_merged = movies_df.merge
(credits_column_renamed, on='id')
movies_df_merged.head ()
movies_df_merged.columns
```

```
movies_cleaned_df = movies_df_merged.drop
(columns = ['homepage', 'title_x', 'title_y', 'status',
'production_countries'])
movies_cleaned_df.columns
movies_cleaned_df.info ()
v = movies_cleaned_df.vote_count
m = movies_cleaned_df.vote_count.quantile (0.7)
R = movies_cleaned_df.vote_average
C = movies_cleaned_df.vote_average.mean ()
movies_cleaned_df['weighted_average'] = ((R*v) +
(C*m))/(v + m)
movies_sorted_ranking =
movies_cleaned_df.sort_values
('weighted_average', ascending=False)
movies_sorted_ranking[['original_title',
'vote_count', 'vote_average', 'weighted_average',
'popularity']].head (10)
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler ()
movie_scaled_df = scaler.fit_transform
(movies_cleaned_df[['weighted_average',
'popularity']])
movie_normalized_df = pd.DataFrame
(movie_scaled_df, columns= ['weighted_average',
'popularity'])
movies_cleaned_df[['normalized_weighted_averag
e', 'normalized_popularity']] =
movie_normalized_df
movies_cleaned_df['score'] =
movies_cleaned_df['normalized_weighted_average'
] * 0.5 +
movies_cleaned_df.normalized_popularity * 0.5
movies_scored_df =
movies_cleaned_df.sort_values (['score'],
ascending=False)
movies_scored_df[['original_title',
'normalized_weighted_average',
'normalized_popularity', 'score']].head (10)
movies_cleaned_df.head(1)['overview']
from sklearn.feature_extraction.text import
TfidfVectorizer
tfv = TfidfVectorizer (
   min_df = 3,
   max_features = None,
   strip_accents = 'unicode',
   analyzer = 'word',
   token_pattern = r'\w{1,}',
   ngram_range = (1, 3),
   stop_words = 'english'
)
movies_cleaned_df['overview'] =
movies_cleaned_df['overview'].fillna ('')
tfv_matrix = tfv.fit_transform
(movies_cleaned_df['overview'])
from sklearn.metrics.pairwise import
sigmoid_kernel
sig = sigmoid_kernel (tfv_matrix, tfv_matrix)
indices = pd.Series (movies_cleaned_df.index,
index=movies_cleaned_df['original_title']).drop_du
plicates()
def give_recommendations (title, sig = sig):
   idx = indices[title]
   sig_scores = list (enumerate (sig[idx]))
   sig_scores = sorted (sig_scores, key=lambda x :
x[1], reverse=True)
   sig_scores = sig_scores[1:11]
   movie_indices = [i[0] for i in sig_scores]
   return
movies_cleaned_df['original_title'].iloc[movie_indi
ces]
give_recommendations ('The Dark Knight')
```

Output:

```
3                         The Dark Knight Rises
299                              Batman Forever
428                              Batman Returns
3854    Batman: The Dark Knight Returns, Part 2
1359                                     Batman
2507                                  Slow Burn
1181                                        JFK
119                               Batman Begins
205          Sherlock Holmes: A Game of Shadows
879                         Law Abiding Citizen
Name: original_title, dtype: object
```

Conclusion:

Hybrid recommendation systems combine collaborative filtering and content-based methods to provide diverse and accurate recommendations. By leveraging both user-item interactions and item features, these systems offer personalized suggestions, overcoming limitations of individual approaches. This hybridization enhances recommendation quality, effectively addressing the cold-start problem and improving user satisfaction. By integrating multiple techniques, hybrid systems deliver robust and versatile recommendation solutions for various domains.