Movie Recommendation System*

Weronika Wołowczyk[1,*,†], Ewa Szymik[1,†]

Faculty of Applied Mathematics, Silesian University of Technology, Kaszubska 23, 44100 Gliwice, POLAND

Abstract

The goal of the project is to deliver personalized movie suggestions based on user preferences by analyzing and processing a dataset of movies. The project's primary stages are data cleaning (as some unnecessary dataset columns were removed), exploratory data analysis (several visualizations of dataset characteristics were presented), and creating a recommendation system based on a soft-set theory.

The core of the project is a recommendation system that makes movie suggestions based on user input. Users are asked to state their preferences concerning actors, genres, and keywords. Then, a soft set-based classification method is applied to score and rank the films depending on these preferences. The system calculates a total score for each movie based on its attributes, ultimately providing the top five propositions.

There are also introduced methods for recommending 5 most similar movies to a given title and predicting movie ratings based on their features using k-nearest neighbor (knn) algorithm. In the first method, the algorithm searches for most similar movies based on their attributes and in the second it predicts a movie's rating by analyzing the votes of the k films with the most similar features.

Overall, the project presents the application of algorithmic techniques and machine learning methods such as soft-sets, to provide personalized suggestions, and k nearest neighbours algorithm to analyse data and predict data attributes.

Keywords

movie recommendation, softset, data analysis, personalized recommendations, data preprocessing, KNN algorithm, vote predictions

1. Introduction

This increase in the number of films being produced each year creates a difficulty for the viewers, to chose a movie that suites the most to their liking. With all of the streaming services, users have a rich library of content available to them, and picking and choosing what to watch may get more and more complicated. That is why there is a need for recommendation systems: to provide personalized content and, as a result, to improve the user experience by recommending only the movies that might spark an interest and match the viewers' individual preferences.

Collaborative filtering, content-based recommendation and hybrid methods are the typical techniques that are employed by many existing recommendation systems. Collaborative filtering depends on preferences of other like-minded users whereas content-based filtering recommends items to the user based on descriptions of items. Hybrid methods combine both approaches to leverage their strengths and mitigate their weaknesses.

*IVUS2024: Information Society and University Studies 2024, May 17, Kaunas, Lithuania

[1,*] Corresponding author

[†] These author contributed equally.

ww308053@student.polsl.pl (W. Wołowczyk); es308045@student.polsl.pl (E. Szymik)

This paper explores the use of soft set theory as an approach to movie recommendation systems. Soft sets, introduced by Molodtsov in 1999, are mathematical models used for reasoning under conditions of uncertainty and vagueness. Unlike traditional sets, where an element either belongs to the set or does not, soft sets allow for partial membership, with elements having degrees of belonging. This degree is typically represented by a value between 0 and 1, indicating how strongly an element is associated with the set. Soft sets are particularly useful in fields such as decision-making and artificial intelligence, where uncertainty and vagueness are common. In the context of movie selection, they provide a flexible classification method that can accommodate the varied nature of user preferences.

The KNN (k-nearest neighbors) algorithm is a simple, non-parametric method for classification tasks in machine learning. It operates on the principle of proximity: it consists of finding k closest objects in feature space to the element currently being tested . Therefore, regarding feature similarity, they are called neighbours. Neighbors are derived from a set of objects used to train the algorithm. The resulting class is the one in which there is the highest number of neighbours. Most often, the distance between the elements is calculated using Euclidean or Manhattan metric.

The KNN classifier is used, firstly, to recommend 5 most similar movies to the one provided by a user, and secondly, to predict movie ratings based on the *vote_average* attribute of similar movies. The steps of the algorithm are normalizing the data, splitting the data into training set and test set, fitting the model, and evaluating the accuracy of the predictions to determine the effectiveness of the model.

Overall, main point of this project is the development of a personalized recommendation system that leverages user-defined preferences for genres, actors, and keywords. By applying soft set theory, the system calculates a total score for each movie based on its alignment with user preferences and displays the five films with the highest mark, resulting in customized movie suggestions. Second point is the recommendation of 5 most similar movies. Users are asked to input a title and then the system finds similar movies feature-wise. The prediction of movie ratings based on their features is the third point. The system finds similar movies in the training set and based on their *vote_average* attribute predicts the rating of movies from test set. Both points use K-nearest neighbours algorithm.

2. Methodology

Soft set methodology offers a flexible approach for handling uncertainties and making decisions based on multiple parameters. Its simplicity and adaptability make it a powerful tool for various applications, including recommendation systems.

2.1. SoP Set

A soft set (F, E) over a universal set U is a pair where F is a mapping given by $F: E \rightarrow P(U)$. Here, E is a set of parameters, and $P(U)$ denotes the power set of U. For each parameter $e \in E$, $F(e)$ is a subset of U.

2.2. Mathematical Model

- Step 1: Define the Universal Set U
  Let U represent a universal set containing elements that need to be analyzed and categorized. In a movie recommendation system, U is a set of films:

$$U = \{f_1, f_2, f_3, \ldots, f_n\}$$

- Step 2: Define the Set of Parameters E
  Parameters E define the attributes relevant to the elements in U. These parameters could be movie genres:

$$E = \{\text{action, drama, comedy, adventure, } \ldots\}$$

- Step 3: Define the Mapping F
  The mapping F associates each parameter $e \in E$ with a subset of U. For instance, if the parameter is "adventure," F(adventure) might include films classified as adventure films:

$$F(\text{adventure}) = \{f_1, f_2, f_3\}$$

$$F(\text{drama}) = \{f_2, f_4\}$$

2.3. Constructing the SoP Set

- Step 4: Construct the Soft Set (F, E)
  The soft set is constructed by pairing each parameter with its corresponding subset:

$$(F, E) = \{(\text{adventure}, \{f_1, f_2, f_3\}), (\text{drama}, \{f_2, f_4\}), \ldots\}$$

2.4. Decision-Making Using SoP Sets

- Step 5: Represent Soft Set in a Binary Table
  The soft set can be represented in a binary table for easier analysis. Each row corresponds to an element in U, and each column corresponds to a parameter in E. An entry is 1 if the element is associated with the parameter, and 0 otherwise.

U	adventure	drama	comedy
f_1	1	0	0
f_2	1	1	0
f_3	1	0	1
...	...	...	...

- Step 6: Calculate Selection Values
  Assign weights to each parameter to reflect their importance. Multiply the binary values by these weights and sum them up for each element in U. This gives a selection value indicating the relevance of each element based on the given parameters.
- Step 7: Determine the Best Choice
  The elements with the highest selection values are considered the best choices based on the parameters. This can be used for recommendations.

2.5. Computational Example

- Class U:

$$U = \{f_1, f_2, f_3, \ldots, f_n\}$$

where f_i are films.

- Set of parameters E defining movie genres:

E = {action, drama, crime, adventure, science-fiction, thriller, fantasy, western, animation, . . .}

- Set of considered parameters A:

$$A = \{\text{adventure, fantasy, animation}\}$$

- There are 6 films in the class U:

$$U = \{f_1, f_2, f_3, f_4, f_5, f_6\}$$

$$E = \{e_1, e_2, e_3\}$$

- Assumption: F:

$$F(e_1) = \{f_1, f_2, f_3, f_6\}$$
$$F(e_2) = \{f_1, f_4, f_6\}$$
$$F(e_3) = \{f_1, f_3, f_6\}$$

- Soft set F:

(F, E) = {(adventure = $\{f_2, f_6\}$), (fantasy = $\{f_1, f_4, f_6\}$), (animation = $\{f_3, f_6\}$)}

- c_i - selected value of object $f_i \in U$

$$c_i = \sum_j f_{ij}$$

- $d_{ij} = w_j \times f_{ij}$ - input data of the weighted table , $w_j \in (0, 1]$

$$c_i = \sum_j d_{ij}$$

U	adventure, $u_1 = 0.8$	fantasy, $u_1 = 0.3$	animation, $u_1 = 0.9$	Selection Value
f_1	1	1	1	$a_1 = 2$
f_2	1	0	0	$a_2 = 0.8$
f_3	1	0	1	$a_3 = 1.7$
f_4	0	1	0	$a_4 = 0.3$
f_5	0	0	0	$a_5 = 0$
f_6	1	1	1	$a_6 = 2$

Table 1
Binary table with weights assigned to parameters, for adventure = 0.8; for fantasy = 0.3; for animation = 0.9

- In the table, it is evident that the films most corresponding to the selection parameters are f_1 and f_6.
- The same calculations are performed for actors and keywords.

Visualization of the recommendation system:

```
YOUR GENRES PREFERENCES
Genres list:
['Action', 'Adventure', 'Science Fiction', 'Thriller', 'Fantasy', 'Crime', 'Western', 'Drama', 'Family', 'Animation', 'Comedy', 'Mystery', 'Romance', 'W
ar', 'History', 'Music', 'Horror', 'Documentary', 'TV Movie', '', 'Foreign']
Select a genre from the list or write NEXT:  Crime
Enter value:  0.5
Select a genre from the list or write NEXT:  Action
Enter value:  0.4
Select a genre from the list or write NEXT:  NEXT
RATE ACTORS
Actors list:
['Chris Pratt', 'Bryce Dallas Howard', 'Irrfan Khan', "Vincent D'Onofrio", 'Nick Robinson', 'Tom Hardy', 'Charlize Theron', 'Hugh Keays-Byrne', 'Nichola
s Hoult', 'Josh Helman', 'Shailene Woodley', 'Theo James', 'Kate Winslet', 'Ansel Elgort', 'Miles Teller', 'Harrison Ford', 'Mark Hamill', 'Carrie Fishe
r', 'Adam Driver', 'Daisy Ridley', 'Vin Diesel', 'Paul Walker', 'Jason Statham', 'Michelle Rodriguez', 'Dwayne Johnson', 'Leonardo DiCaprio', 'Will Poul
ter', 'Domhnall Gleeson', 'Paul Anderson', 'Arnold Schwarzenegger']
Select an actor from the list or write NEXT:  Chris Pratt
Enter value:  0.7
Select an actor from the list or write NEXT:  Harrison Ford
Enter value:  0.6
Select an actor from the list or write NEXT:  NEXT
WHAT ELSE DO YOU PREFER?
Keywords list:
['monster', 'dna', 'tyrannosaurus rex', 'velociraptor', 'island', 'future', 'chase', 'post-apocalyptic', 'dystopia', 'australia', 'based on novel', 'rev
olution', 'sequel', 'dystopic future', 'android', 'spaceship', 'jedi', 'space opera', '3d', 'car race', 'speed', 'revenge', 'suspense', 'car', 'father-s
on relationship', 'rape', 'mountains', 'winter', 'saving the world', 'artificial intelligence']
Write keyword or NEXT:  dna
Enter value:  0.3
Write keyword or NEXT:  car race
Enter value:  0.2
Write keyword or NEXT:  NEXT
```

id	original_title	genres	cast	keywords	vote_average	keywords_values	total_score
5503	The Fugitive	[Adventure, Action, Thriller, Crime, Mystery]	[Harrison Ford, Tommy Lee Jones, Sela Ward, Ju...	[chicago, showdown, undercover, surgeon, death...	7.0	0.0	1.5
10675	Frantic	[Thriller, Action, Crime, Drama, Mystery]	[Harrison Ford, Emmanuelle Seigner, Betty Buck...	[wife husband relationship, married couple, wi...	6.4	0.0	1.5
9869	Patriot Games	[Drama, Action, Thriller, Crime]	[Harrison Ford, Anne Archer, Patrick Bergin, T...	[assassination, assassin, repayment, ira, jack...	6.0	0.0	1.5
135397	Jurassic World	[Action, Adventure, Science Fiction, Thriller]	[Chris Pratt, Bryce Dallas Howard, Irrfan Khan...	[monster, dna, tyrannosaurus rex, velociraptor...	6.5	0.3	1.4
118340	Guardians of the Galaxy	[Action, Science Fiction, Adventure]	[Chris Pratt, Zoe Saldana, Dave Bautista, Vin ...	[marvel comic, spaceship, space, scene during ...	7.9	0.0	1.1

Figure 1: Input of example preferences and matching results

2.6. K-nearest neighbours

The KNN algorithm is a simple classifier that consist of finding k elements in a given dataset that are most similar to the test element. It follows the steps:

1. **Data Collection:** Gathering training data, which will be used to build the model. Each data point is represented by a set of features and its corresponding class to be predicted. In this project, data points are movies from the database, class to be predicted is the vote_average value of the movie.
2. **Determining the Value of Parameter K:** The parameter K specifies how many nearest neighbors will be considered during the classification of a new data point. Choosing an appropriate value for K can significantly impact the effectiveness of the model. In the movie recommendation system, k takes values from 2 to 9.
3. **Calculating Distances:** For a new data point whose class is to be predicted, distances to all points in the training set are calculated. This determines the similarity between points. In the project, the Manhattan metric is used:

$$d(a,b) = \sum_{i=1}^{m} |a_i - b_i|$$

4. **Selecting K Nearest Neighbors:** The next step is to select K training points that have the closest distances to the point currently tested.
5. **Classifying the Point:** After selecting the K nearest neighbors, the point is classified. The method for this is a majority vote, where the class of the new data point is determined by the dominant class among the K nearest neighbors.
6. **Determining the accuracy** The final step is to assess the performance of the KNN model. This is be done by splitting the data into a training and testing set, and then comparing the predicted classes with the actual classes in the testing set.
7. To avoid the dominance of features with larger values, feature normalization is applied before using KNN.

$$x_{\text{norm}} = \frac{x - x_{\text{min}}}{x_{\text{max}} - x_{\text{min}}}$$

3. Experiments

This chapter focuses on the experiments conducted to develop a machine learning model for movie recommendations. By testing various algorithms, we aim to enhance the accuracy and effectiveness of our recommendation system. Our goal is to bet- ter understand the key factors that contribute to successful movie recommendations, ultimately improving the user experience.

3.1. Database description

The dataset utilized in this study was sourced from Kaggle.com, a widely recognized open-access platform renowned for its vast collection of publicly available datasets. Title of database is "TMDb Movies Dataset". There are 10856 records in total, which contain 21 columns.

```
<class 'pandas.core.frame.DataFrame'>
Index: 10866 entries, 135397 to 22293
Data columns (total 20 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   imdb_id               10856 non-null  object
 1   popularity            10866 non-null  float64
 2   budget                10866 non-null  int64
 3   revenue               10866 non-null  int64
 4   original_title        10866 non-null  object
 5   cast                  10790 non-null  object
 6   homepage              2936 non-null   object
 7   director              10822 non-null  object
 8   tagline               8042 non-null   object
 9   keywords              9366 non-null   object
 10  overview              10305 non-null  object
 11  runtime               10866 non-null  int64
 12  genres                10843 non-null  object
 13  production_companies  9836 non-null   object
 14  release_date          10866 non-null  object
 15  vote_count            10866 non-null  int64
 16  vote_average          10866 non-null  float64
 17  release_year          10866 non-null  int64
 18  budget_adj            10866 non-null  float64
 19  revenue_adj           10866 non-null  float64
dtypes: float64(4), int64(5), object(11)
memory usage: 1.7+ MB
```

Figure 2: Database

3.2. Evaluation Metric

Model will be measured by using metric accuracy. Accuracy is the most popular metric and it shows how often a classification of an ML model is correct overall.

$$Accuracy = \frac{TP + TN}{TP + TN + FN + FP} \tag{1}$$

Where *TP* (True Positives) represent instances that were accurately identified as positive, *TN*(True Negatives) represent instances that were accurately identified as

negative, FN (False Negatives) are instances where positive cases were incorrectly identified as negative, and FP (False Positives) are instances where negative cases were incorrectly identified as positive.

3.3. Model analysis

```
Accuracy for k=2: 0.33839429491603407
Accuracy for k=3: 0.3551874856222682
Accuracy for k=4: 0.39751552795031053
Accuracy for k=5: 0.42259029215550953
Accuracy for k=6: 0.42374051069703245
Accuracy for k=7: 0.432712215320911
Accuracy for k=8: 0.4357027835288705
Accuracy for k=9: 0.4428341384863124
```

Figure 3: Comparing accuracy for different k using Standard normalization

```
Accuracy for k=2: 0.32367149758454106
Accuracy for k=3: 0.35150678628939497
Accuracy for k=4: 0.38808373590982287
Accuracy for k=5: 0.4074074074074074
Accuracy for k=6: 0.4140786749482402
Accuracy for k=7: 0.422360248447205
Accuracy for k=8: 0.4313319530710835
Accuracy for k=9: 0.43823326432022086
```

Figure 4: Comparing accuracy for different k using Min-max normalization

The figures above depict the accuracy of our KNN model across different k values using standard and min-max normalization techniques. Our target variable, the rounded vote average, poses a challenge due to its unpredictability.

Higher k values generally lead to improved model performance, indicating more stable predictions as more neighbors are considered. Additionally, standard normalization slightly outperforms min-max normalization, when applied to features like runtime and release year.

In summary, our experiments highlight the effectiveness of higher k values and standard normalization in enhancing the predictive performance of our movie recommendation system. These findings emphasize the importance of careful normalization and k value selection in predictive modeling tasks.

4. Conclusion

This paper presents the design of a personalized movie recommendation system using soft set theory and the k-nearest neighbors (KNN) algorithm. The main goal is to build a system that recommends top five movies for a given user according to their preferences. Additional functionalities include suggesting similar movies to a given title, and predicting movie ratings based on data features.

Soft Set Theory provides a powerful tool for dealing with the uncertainty and vagueness associated with user' preferences. Representing user preferences as soft sets allows calculating a total score for each movie and making personalized recommendations that align with users' individual preferences. This shows the flexibility and efficiency of soft sets in decision making processes.

The use of K nearest neighbours algorithm further expands the project. The KNN classifier identifies movies similar to a user-specified title and predicts movie ratings based on ratings of records closest in the feature space. The effectiveness of those predictions was checked. The accuracy oscillates for different values of k, increasing as k increases and reaching the highest value of 44% when k equals to 9 (the accuracy for higher values of k was not checked).

Experimental results validate the system's effectiveness in generating personalized movie recommendations as recommended movies are, in fact, aligning with provided preferences. The accuracy of KNN classifier reached only 44% because higher values of k were not tested and it is hard to predict movie ratings based solely on their features.

Soft set theory and KNN have shown to be a potent combination for creating recommendation system that can process diverse user inputs and provide personalized movie suggestions.

References

[1] D. Molodtsov, "Soft set theory — First results," Computers & Mathematics with Applications, vol. 37, no. 4-5, pp. 19-31, 1999.

[2] F. Ricci, L. Rokach, and B. Shapira, "Introduction to Recommender Systems Handbook," in Recommender Systems Handbook, Springer, Boston, MA, 2011, pp. 1-35.

[3] P. Lops, M. De Gemmis, and G. Semeraro, "Content-based Recommender Systems: State of the Art and Trends," in Recommender Systems Handbook, Springer, Boston, MA, 2011, pp. 73-105.

[4] TMDb, The Movie Database, available at: https://www.kaggle.com/datasets/juzershakir/tmdb-movies-dataset/data