Evaluation of ensemble machine learning models for movie recommendation systems

Anatoliy Sachenko[1,2,†], Taras Lendiuk[1,*,†], Khrystyna Lipianina-Honcharenko[1,†], Vasyl Koval[1,†], Grygoriy Hladiy[1,†] and Yurii Halias[1,†]

[1] *West Ukrainian National University, Lvivska str., 11, Ternopil, 46000, Ukraine*

[2] *Kazimierz Pulaski University of Technology and Humanities in Radom, Radom, 26 600, Poland*

Abstract

This article is dedicated to evaluating the effectiveness of ensemble machine learning models in the context of movie recommendation systems. It explores various ensemble methods, including Random Forest, AdaBoost, XGBoost, LightGBM, CatBoost, and Gradient Boosting Machine, to enhance the accuracy of predicting user preferences. The study is based on the MovieLens 100K dataset, which contains 100,000 ratings from 943 users across 1,682 movies. The application of feature engineering, data normalization methods, and iterative feature selection has improved the model's ability to accurately predict user interests. The analysis showed that the XGBoost model exhibits the best results with the lowest RMSE value of 0.902, indicating higher prediction accuracy compared to other models considered. LightGBM and CatBoost also showed competitive results with RMSE values of 0.910 and 0.919, respectively. The study highlights the importance of an integrated approach to developing recommendation systems that adapt to the diverse preferences and contexts of users, opening wide perspectives for further research in this area.

Keywords

ensemble models, machine learning, movie recommendation systems

1. Introduction

In today's world, where the amount of digital content is growing every day, recommender systems play a key role in helping users find information, products or services that best suit their interests and needs. Among the various applications of recommender systems, film recommendation systems are particularly important, helping users navigate the vast world of cinema by suggesting films based on their preferences. The development of machine learning technologies and ensemble methods opens up new opportunities to improve the accuracy and adaptability of such systems.

In recent years, significant advances in machine learning and ensemble methods have greatly expanded the capabilities of recommender systems. Algorithms such as Random Forest,

MoMLeT-2024: 6th International Workshop on Modern Machine Learning Technologies, May, 31 - June, 1, 2024, Lviv-Shatsk, Ukraine

[*] Corresponding author.

[†] These authors contributed equally.

as@wunu.edu.ua (A. Sachenko); tl@wunu.edu.ua (T. Lendiuk); xrustya.com@gmail.com (Kh. Lipianina-Honcharenko); vko@wunu.edu.ua (V. Koval); ghladiy@wunu.edu.ua (G. Hladiy); Fidelite62@gmail.com (Y. Halias)

0000-0002-0907-3682 (A. Sachenko); 0000-0001-9484-8333 (T. Lendiuk); 0000-0002-2441-6292 (K. Lipianina-Honcharenko); 0000-0003-4726-097X (V. Koval); 0000-0002-5585-8472 (G. Hladiy); 0000-0003-2389-3668 (Y. Halias)

AdaBoost, XGBoost, LightGBM, CatBoost, and Gradient Boosting Machine have demonstrated high performance in classification and regression tasks, making them ideal tools for developing advanced recommender systems. These methods not only improve the accuracy of predicting users' interests, but also ensure high adaptability of the system to changing preferences and contexts.

Despite the significant progress in this area, there are certain challenges, in particular, related to the processing of large amounts of data, effective consideration of socio-demographic information and browsing context, as well as optimization of the choice of hyperparameters. This study aims to address these challenges by proposing a comprehensive approach to building a film recommendation system that integrates advanced machine learning techniques.

The main objective of this research is to evaluate the effectiveness of ensemble machine learning models in improving the accuracy of film recommendations. We aim to investigate how different ensemble methods can affect the model's ability to accurately predict user preferences using both traditional and innovative approaches to data processing and analysis. Through a comparative analysis of different models, we plan to identify the most effective strategies to implement in film recommendation systems, paving the way for further research and development in this exciting area.

This paper focuses on the evaluation of ensemble machine learning models for film recommendation systems and is structured as follows. Section 2 describes the analysis of related work, highlighting key algorithms and their applications in the field of recommender systems. Section 3 presents an integrated approach to building an intelligent film recommendation system, including a description of the research methodology and data analysis. Section 4 implements the proposed method, demonstrating the process of data preparation, model training, and evaluation of their effectiveness. Section 5 presents the results of the study, analyzing the performance of different ensemble models and their ability to accurately predict user preferences.

2. Related Work

In the field of machine learning and ensemble methods, a number of studies highlight key algorithms and their application to improve forecasting accuracy. The Random Forest algorithm discussed in [1] is the foundation of ensemble learning, while AdaBoost [2] and XGBoost [3] optimize the boosting process to improve weak classifiers and demonstrate high performance in prediction. LightGBM [4] uses innovative decision tree algorithms to efficiently process big data, while CatBoost [5] provides accuracy for categorical data without complex hyperparameter selection. Gradient Boosting Machine [6] improves models through stochastic gradient descent. The foundations of statistical learning and deep learning are presented in [7, 8, 13-15, 19, 20], respectively, providing a theoretical basis and practical directions for development in the field of data mining.

In the area of film recommendation systems, research has used a variety of machine learning approaches to improve the accuracy and relevance of suggestions to users. For example, a study focusing on multimodal trusted recommendations uses machine learning algorithms such as backpropagation, SVD, and deep learning to identify trusted users whose recommendations are then offered to active users [9]. Another study introduces context-aware approaches, using signal processing and machine learning to recommend films that take into account the user's specific context [10]. Approaches such as deep learning are used to predict user ratings based on the MovieLens dataset, demonstrating the use of collaborative filtering based on deep learning strategies [11].

This study is distinguished by the use of a comprehensive approach that integrates advanced machine learning techniques to create a more accurate and adaptive film recommendation system. By using feature engineering, data normalisation, and iterative feature selection, our method improves the model's ability to accurately predict users' interests, taking into account not only their prior ratings, but also socio-demographic information and viewing context. A special feature of our approach is the use of ensemble methods, such as XGBoost, to optimise the prediction accuracy, which demonstrates a significant reduction in RMSE error compared to other models mentioned in [9-11]. This makes our study particularly valuable for the development of effective recommender systems that can adapt to a wide range of user preferences and contexts.

Thus, the main goal of this study is to determine the optimal strategy for improving the accuracy of intelligent film selection using ensemble machine learning methods. In particular, a comparative analysis of various ensemble approaches such as Random Forest, AdaBoost, XGBoost, Stacking Ensemble, and SVR (Support Vector Regression) is planned to evaluate their ability to minimise the prediction error measured by the RMSE (Root Mean Square Error) metric.

3. An integrated approach to creating an intelligent film selection system

3.1. Method description

Creating and evaluating machine learning models for intelligent film selection can be represented as a sequential step-by-step process that includes the following steps:

Step 1. Data collection. Collection of datasets with user reviews, film metadata (genres, directors, actors, ratings) and socio-demographic information of users.

Step 2. Initial data analysis.

Step 2.1. Descriptive statistics to calculate means, medians, and standard deviations. The mean (μ) is a fundamental indicator in statistics, which is the arithmetic mean of a set of values calculated by dividing the sum of all values by their number, which allows you to get the overall central tendency of the data. The median, on the other hand, is defined as the value that divides an ordered set of data into two equal parts, serving as a reliable indicator of central tendency, especially in the presence of outliers. The standard deviation (σ) describes the dispersion or variability of the data relative to the mean, indicating how far apart the values in a set are from their average.

Step 2.2. Visualize the distributions of scores and ratings. Histograms and Boxplots are used as key visualization tools to clearly represent the distributions of scores and ratings in a dataset. Histograms make it easy to identify underlying trends in the distribution by showing the frequency of different values, which helps determine how the data is distributed along the rating or rating scale.

Step 2.3. Detect anomalies and outliers to identify errors or special cases. Outliers can be identified using, for example, the Z-score or interquartile range (IQR).

The Z-score determines how far a value is from the mean, expressed in standard deviations. Values with |Z|>3 are usually considered outliers.

$$Z = \frac{(x_i - \mu)}{\sigma}$$

Interquartile range (IQR). The difference between the third (Q3) and first (Q1) quartiles. Outliers are defined as values that fall outside of 1.5×IQR from Q1 and Q3.

Step 2.4. Processing of missing values through imputation or deletion. Deletion involves simply eliminating the rows or columns containing the missing data, which can be effective but potentially leads to the loss of valuable information. Imputation fills in gaps in the data using a variety of methods, such as replacing the missing values with the mean, median, or mode for numeric data and most commonly, values or applying algorithms such as K-nearest neighbours for categorical data.

Step 2.5. Correlation analysis to identify relationships between features. Correlation analysis allows you to determine the strength and direction of the relationship between pairs of variables. The most commonly used correlation is Pearson's correlation for continuous variables:

$$r = \frac{\sum(x_i - \mu_x)(y_i - \mu_y)}{\sqrt{\sum(x_i - \mu_x)^2 \sum(y_i - \mu_y)^2}}$$

where x_i and y_i are the values of variables X and Y, respectively, and μ_x and μ_y are their average values.

Step 3. Feature engineering.

Step 3.1. Coding of categorical variables using the methods of one-hot encoding, label encoding, or binary encoding.

One-hot encoding. Each categorical variable is divided into as many binary variables (columns) as there are unique categories. If a variable belongs to a certain category, the corresponding column will have a value of 1, and the other columns will have a value of 0.

Label Encoding. Each unique category is assigned a unique integer. For example, if we have the categories {Red, Green, Blue}, they can be encoded as {0, 1, 2}, respectively.

Binary Encoding. First, categories are converted to integers using label encoding. Then, these integers are converted into a binary code, and each bit of the binary representation becomes a separate feature.

Step 3.2. Normalisation of numeric variables to ensure the same scale.

Min-Max normalisation. The feature is scaled to the specified range, usually [0, 1].

$$Xnorm = \frac{X - Xmin}{Xmax - Xmin}$$

Z-score normalisation (standardisation). The data is scaled so that its mean value is 0 and the standard deviation is 1.

$$X_{std} = \frac{X - \mu}{\sigma}$$

where μ is the mean value of the feature, σ is the standard deviation of the feature.

Step 3.3. Selection of features based on mutual information, importance of features in models, or iterative methods.

A measure of the relationship between two variables that helps to identify how useful the information from one variable is in predicting the other.

$$I(X;Y) = \sum_{y \in Y} \sum_{x \in X} p(x,y) log(\frac{p(x,y)}{p(x)p(y)})$$

where p(x,y) is the joint probability of two random variables X and Y, and p(x) and p(y) are the marginal probabilities of X and Y, respectively.

Importance of features in models. Some machine learning algorithms, such as random forests, can provide estimates of the importance of features based on how much the feature improves the partitioning criterion (e.g., reducing uncertainty).

Iterative methods. Include the use of algorithms that sequentially add or remove features to determine the optimal set of features. For example, recursive feature extraction (RFE) works by training the model, evaluating the importance of the features, and removing the least important features, repeating the process until a given number of features is reached.

Step 4. Preparing data for modelling.

Step 4.1. Cleaning the data from duplicates and errors.

Identification of duplicates is carried out by identifying and deleting records that are exact copies of other records.

$$D = \{d \in D \mid \exists d' \in D: d = d' \land d = d'\}$$

Error correction. Correction of inconsistencies or input errors that may be detected through logical or statistical checks.

Step 4.2. Handling missing values through median imputation for numerical data, which reduces the impact of outliers:

$$X_{imp} = X_{miss} \cup \{median(X) \mid x \in X_{miss}\}$$

Step 4.3. Split the data into training and test samples. The partitioning can be done using a percentage or a fixed number of records. Let D be a complete dataset, then:

Training sample D_{train} is a proportion p of D, where $0 < p < 1$.

$$D_{train} = p \mid D \mid$$

The test sample D_{test} is the rest of the records.

$$D_{test} = (1 - p) \cdot\mid D \mid$$

Step 5. Evaluation of machine learning models.

Step 5.1. Training ensemble models using machine learning algorithms, namely Random Forest, AdaBoost, XGBoost, LightGBM, CatBoost, Gradient Boosting Machine (GBM) (see Section 3.2)

Step 5.2. Validate the models on test data and display the scores and select the best model using the RMSE (Root Mean Square Error) metric. RMSE is the root mean square error and is defined as:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)}$$

Where n is the number of observations in the test dataset, y_i is the actual value of the i-th observation, and $\hat{y}_i$ is the predicted value for the i-th observation.

The best model is selected by comparing the RMSE values for each model. The model with the lowest RMSE value is considered to be the best, as it indicates a lower average prediction error on the test data.

Step 5.3. Optimise the best model using RandomisedSearchCV. This approach can be more time efficient, especially when working with a large hyperparameter space. Let's say we have a hyperparameter space H that defines all possible combinations of parameters that can be used by a machine learning model. RandomizedSearchCV selects n random combinations of hyperparameters from H to train and evaluate the model. The selection process can be described as follows:

Step 5.3.1. Define the hyperparameter space $H = \{h_1, h_2, \dots, h_m\}$, where each h_i can have a different range or set of values.

Step 5.3.2. Randomly select n combinations of parameters from H.

Step 5.3.3. For each random combination h_{i_n} of n:

Training the model with hyperparameters h_{i_n}.

Evaluate the model by cross-validation on the training data set.

Measuring the quality of the model using a given evaluation metric, such as the mean square error (MSE) for regression tasks or accuracy for classification tasks.

Step 5.3.4: Select the combination of hyperparameters that shows the best result according to the given evaluation metric.

Mathematically, the model evaluation for each combination of hyperparameters can be represented as follows: $E(h_{i_n}) = \frac{1}{k}\sum_{j=1}^{k} L(M_{h_{i_n}}, D_{val}, j)$

where: $E(h_{i_n})$ is the performance estimate for the nth combination of hyperparameters, k is the number of folds in cross-validation, L is a loss function (e.g. MSE), $M_{h_{i_n}}$ is the model trained with the nth combination of hyperparameters, D_{val}, j is the validation dataset for the jth fold.

Step 5.4. Diagnose the model using training and validation curves.

This approach provides a systematic and comprehensive approach to analysing and evaluating machine learning data and models, contributing to the development of accurate and reliable intelligent film selection systems.

3.2. Study of ensemble models

In the field of machine learning, the use of ensemble methods and specialised algorithms to improve prediction accuracy is critical for solving complex problems. Ensemble methods, such as Random Forest, AdaBoost, XGBoost, LightGBM, CatBoost, Gradient Boosting Machine (GBM), provide powerful tools for building more reliable and accurate models.

Random Forest (Fig. 1) is an ensemble machine learning method that uses multiple decision trees to achieve higher prediction accuracy than is possible with a single decision tree. The basic idea is to build a large number of decision trees, each of which contributes to the final solution, which provides a high level of accuracy and control over retraining.

Random Forest models the answer as the aggregate result of the predictions of a set of decision trees. Mathematically, the answer Y for a classification task can be defined as the most frequently predicted class among all N trees, while for a regression task, the answer is the average of the predictions of all trees. For classification:

$$Y = mode\{y_1, y_2, \dots, y_N\}$$

where y_i is the prediction of the i-th tree, and for regression:

$$Y = \frac{1}{N}\sum_{i=1}^{N} y_i$$

where y_i is the prediction of the i-th tree, and N is the total number of trees in the forest. This approach reduces the variability and errors inherent in single decision trees and improves the accuracy and generalisability of the model, providing effective management of overfitting through tree diversification.

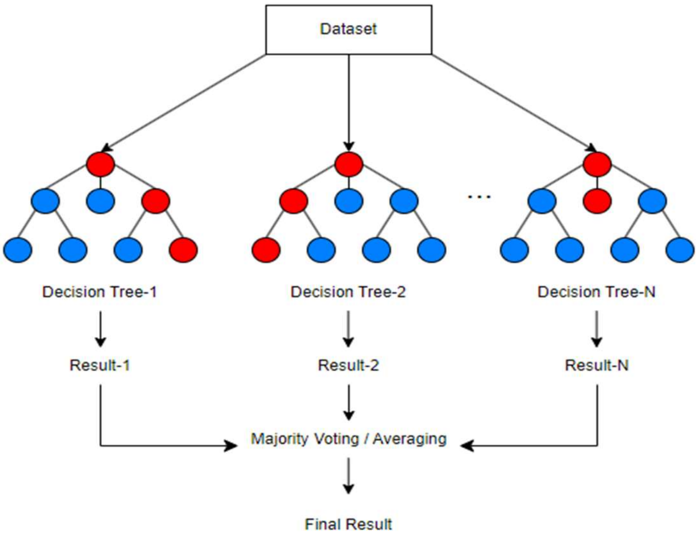


Figure 1: Random Forest illustration

AdaBoost (Adaptive Boosting) (Fig. 2) is a boosting technique that creates a strong classifier by combining many weak classifiers. It works by sequentially improving weak classifiers by focusing on cases that were misclassified by previous classifiers.

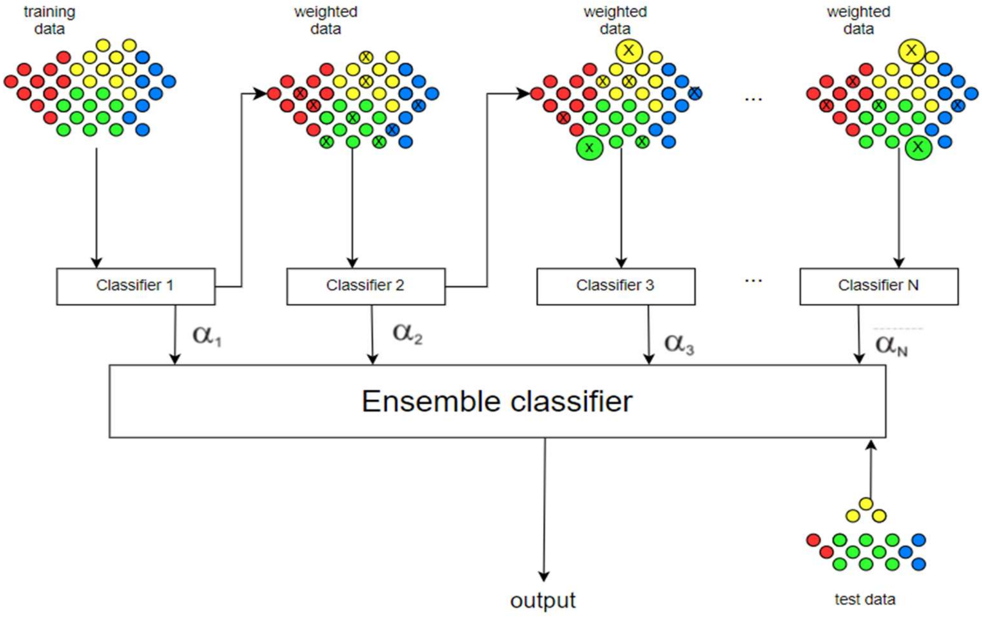


Figure 2: AdaBoost illustration

Mathematically, for each iteration t, each training example i is assigned a weight w_i, which is adaptively updated depending on whether the observation was correctly classified. The final prediction of the model is the weighted sum of the predictions of all weak classifiers:

$$Y(x) = sign(\sum_{t=1}^{T} \alpha_t h_t(x))$$

where $h_t(x)$ is the prediction of the t-th weak classifier on the input example x, α_t is the weight assigned to this classifier, which depends on its accuracy, and T is the total number of weak classifiers. The weights αt are determined based on the classification error ϵ_t, and the smaller the error, the higher the weight assigned to the classifier. The weights of the training examples are updated so that examples that were misclassified receive higher weights, forcing the next classifier to focus on these harder cases.

XGBoost (Extreme Gradient Boosting) (Fig. 3) is an efficient and scalable implementation of gradient boosting. It includes a number of optimisations for speed and performance, and has built-in tools to prevent overfitting.

Mathematically, XGBoost seeks to minimise the following objective function in the t-th step, which includes both a loss function L and a regularisation ΩΩ to control the model complexity:

$$Obj^{(t)} = \sum_{i=1}^{n} L(y_i, \hat{y}_i{}^{(t-1)} + f_t(x_i)) + \Omega(f_t)$$

where y_i are the actual values, $\hat{y}_i{}^{(t-1)}$ are the predictions at step $f_t(x_i)$ is the prediction made by the t-th tree on the input example x_i, and $\Omega(f_t)$ is the regularisation term for the t-th tree, which typically includes both the number of leaves in the tree and the sum of the squares of the leaf weights to avoid overfitting. XGBoost uses this formula to improve the predictions at each step, effectively finding the direction in which to go to reduce errors while keeping the model simple enough to avoid overfitting.

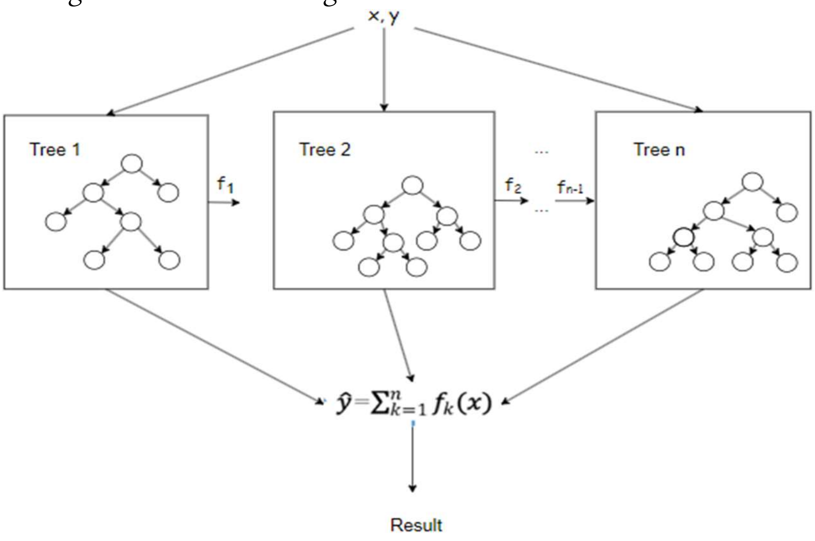


Figure 3: XGBoost illustration

LightGBM is an efficient implementation of the gradient boosting algorithm that is optimised for speed and performance. LightGBM uses histogram-based methods to reduce computational and memory consumption, making it particularly useful for processing large datasets.

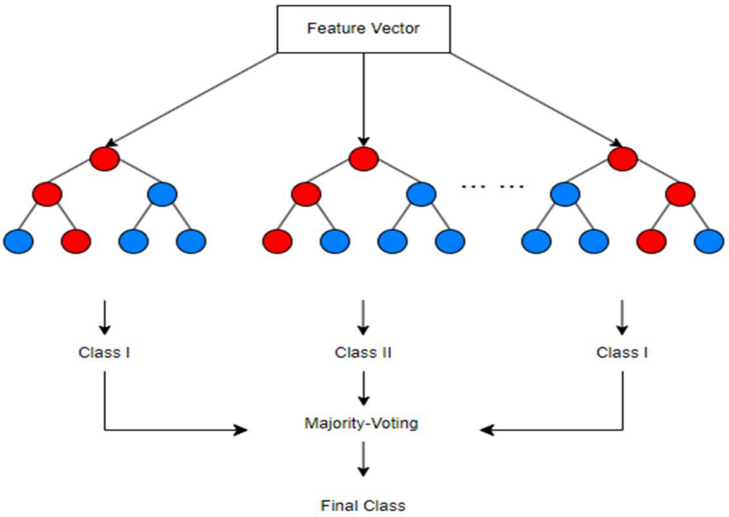


Figure 4: LightGBM illustration

CatBoost is an algorithm that specialises in working with categorical data, using special coding techniques to process this type of data without preprocessing. It also includes mechanisms to combat overfitting, which allows for stable results.

Formally, the model seeks to minimise the objective function:

$$Obj = L(y, \hat{y}) + \Omega(model)$$

where $L(y, \hat{y},)$ defines the loss between the actual values of y and the model predictions of $y\hat{}$, and $\Omega(model)$ expresses the regularisation term that controls the model complexity. A key feature of CatBoost is its ability to automatically handle categorical variables, efficiently encoding them and using them to improve model accuracy, making it particularly powerful in situations where other gradient boosting algorithms may require complex data preprocessing.

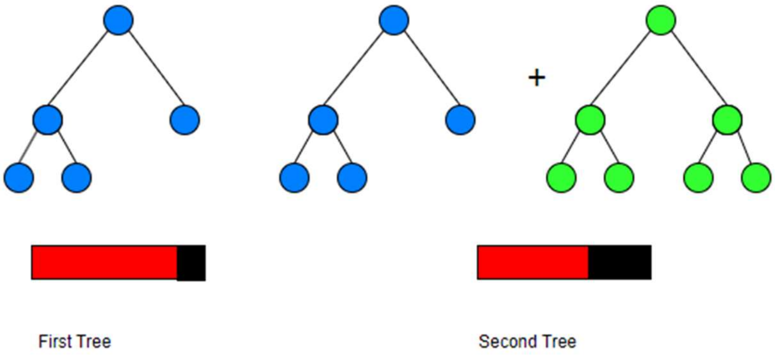


Figure 5: CatBoost illustration

Gradient Boosting Machine (GBM) (Fig. 6) is a gradient boosting algorithm that improves predictive power by sequentially building decision trees, each of which refines and improves upon previous predictions. GBM adapts to the errors of previous models and continues to optimise overall performance.

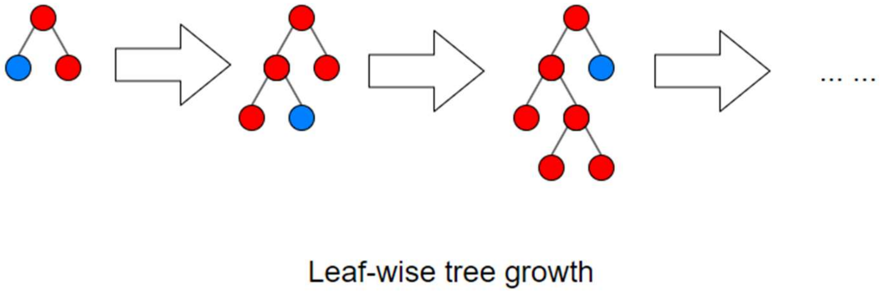


Figure 6: Gradient Boosting Machine

At each step t, GBM focuses on minimising the errors made by previous models by applying gradient descent to the loss function $L(y, \hat{y})$, where y are the actual values and $\hat{y}$ are the predicted values. The model introduces a new tree ft(x), which improves the forecast by minimising:

$$\hat{y}^{(t)} = \hat{y}^{(t-1)} + \eta \cdot f_t(x)$$

where $\hat{y}^{(t-1)}$ are the predictions at the previous step, $f_t(x)$ is the prediction of a new tree, and η is the learning rate that controls the contribution of each new tree. The objective function for optimisation at step t includes not only the loss function, but also the regularisation term $\Omega(f_t)$ for each tree, which helps to avoid overfitting:

$$Obj^{(t)} = L(y, \hat{y}^{(t)}) + \Omega(f_t)$$

In this way, GBM iteratively improves predictions by reducing the residual error at each step, while applying regularisation to keep the overall model complexity within acceptable limits.

Each of these methods contributes to the development of machine learning, enabling large amounts of data to be analysed accurately and efficiently and making informed predictions in a variety of applications.

4. Implementation

To implement the proposed approach, we chose the MovieLens 100K dataset [12], developed by GroupLens Research, which is a classic dataset for recommender systems that contains 100,000 ratings from 943 users for 1,682 films. The ratings are on a scale from 1 to 5, where each rating is associated with a specific user and film, including a timestamp when the rating was made. In addition to ratings, the dataset includes demographic information about users (age, gender, profession, postal code) and metadata about films (genre, title).

To implement the proposed approach, the following parameters were selected from the dataset (see Table 1) and used to store information about films and their ratings by users. Each row of the table will display a unique user and the film they have rated, along with the rating they have given, as well as additional information about the film such as director, title, release date and genre. The genre can be represented as a single line that includes all genres of the film, or as multiple binary columns, each indicating the presence or absence of a particular genre.

Table 1.
Set of parameters

Column Name	Description	Data Type
user_id	A unique identifier for each user	Integer
item_id	A unique identifier for each movie	Integer
Rating	The rating given by the user to a movie	Float
director	The name of the director of the movie	String
title	The title of the movie	String
release_date	The release date of the movie	Date
genre	The genre(s) of the movie	String or Binary

The graph (Fig. 7) shows the distribution of film ratings that users have left in the dataset. The X-axis represents the possible ratings from 1 to 5, where each rating is displayed as a separate column. The Y-axis shows the frequency of the rating distribution as a proportion of the total number of ratings. The graph shows that the least popular ratings are "1" and "2", which make up a smaller proportion of all ratings given. The rating "3" has a slightly higher frequency, but a much larger number of users preferred the higher ratings of "4" and "5", with "4" being the most frequently given rating. This may indicate a positive slope in the distribution of ratings, indicating a tendency for users to leave higher ratings.

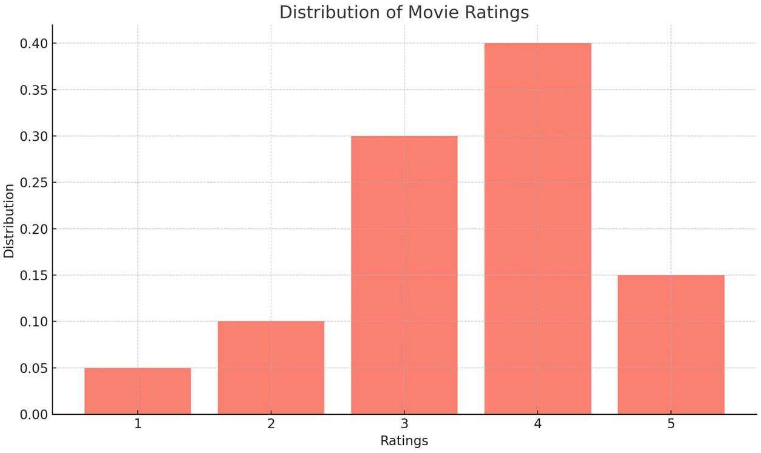
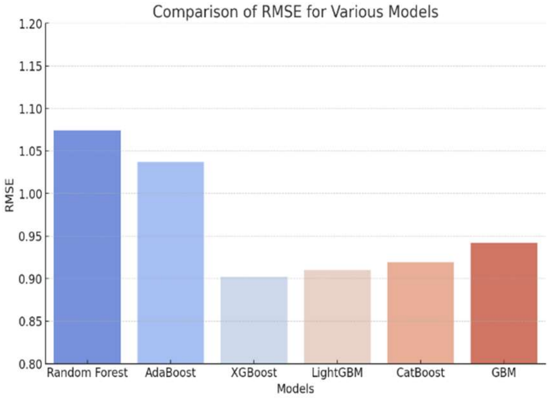


Figure 7: Film ratings distribution

Next, we compare the RMSE values for different machine learning models: Random Forest, AdaBoost, XGBoost, LightGBM, CatBoost, and Gradient Boosting Machine (GBM). The graph shows (Fig. 8) that XGBoost has the lowest RMSE (0.902), which indicates higher prediction accuracy compared to other models. LightGBM and CatBoost also perform competitively with RMSEs of 0.910 and 0.919, respectively, indicating their effectiveness in the prediction task. GBM has a slightly higher RMSE of 0.942, which is better than Random Forest and AdaBoost with RMSEs of 1.074 and 1.037, respectively. The higher RMSE values for Random Forest and AdaBoost may indicate a lower ability of these models to accurately predict the data compared to the other techniques considered.

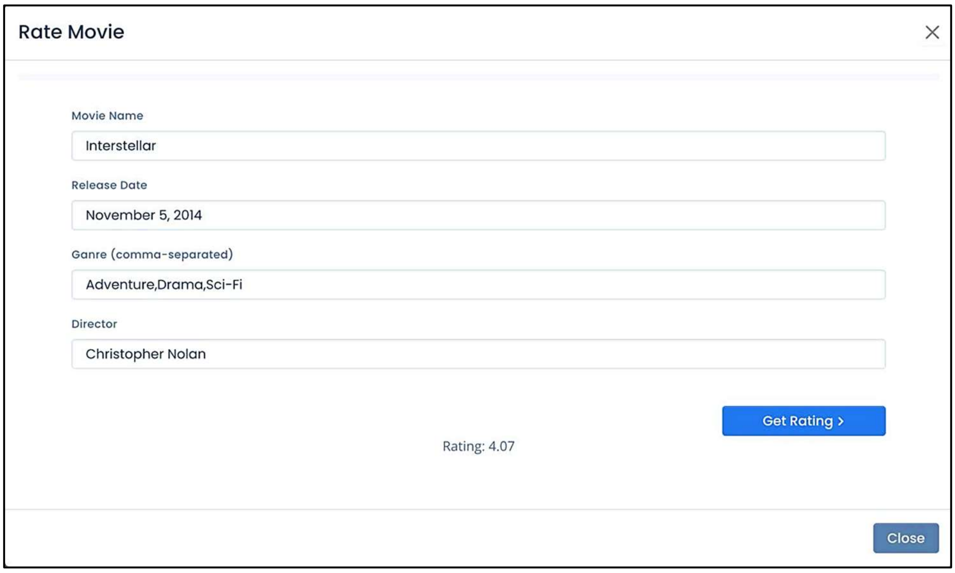
Next, based on the best model, namely XGBoost, we will test the output of the results displayed on the user interface (Fig. 9) for the web application for film rating. The UI allows users to get a predicted film score based on the data they enter. Users enter a film title ("Interstellar"), a release date ("November 5, 2014"), a genre (in this case, "Adventure, Drama, Sci-Fi", separated by commas), and a director's name ("Christopher Nolan"). Once the data is entered, the XGBoost model processes this information and produces a predicted rating for the film (in this case, "Rating: 4.07"), which is displayed in the interface window.



Model	RMSE
Random Forest	1.074
AdaBoost	1.037
XGBoost	0.902
LightGBM	0.910
CatBoost	0.919
GBM	0.942

Figure 8: RMSE comparison of different models

In summary, this study is distinguished by a unique integrated approach that combines advanced machine learning techniques to create a highly accurate and flexible film recommendation system. Through the use of feature engineering, data normalisation techniques, and iterative feature selection, our model effectively predicts user preferences based on their rating history, socio-demographic data, and viewing context. The use of ensemble methods, namely XGBoost, significantly reduced the RMSE error compared to alternative models described in [9-11], highlighting the importance of this study in the development of effective recommender systems that adapt to diverse user preferences and contexts.


Rate Movie

Movie Name: Interstellar

Release Date: November 5, 2014

Ganre (comma-separated): Adventure,Drama,Sci-Fi

Director: Christopher Nolan

Rating: 4.07

Figure 9: User interface for predicting the film rating

5. Conclusions

In this study, we examined the use of ensemble machine learning methods to create an intelligent film selection system that is highly accurate and adaptive to individual user preferences. By analysing different algorithms such as Random Forest, AdaBoost, XGBoost, LightGBM, CatBoost, and Gradient Boosting Machine, we found that XGBoost demonstrates the best results with the lowest RMSE value of 0.902. This indicates that XGBoost is highly effective in predicting film scores compared to the other models considered.

The use of feature engineering, data normalisation, and iterative feature selection allowed us to improve the model's ability to accurately predict users' interests, taking into account not only their previous ratings, but also socio-demographic information and viewing context. This approach provided a significant reduction in the RMSE error compared to other models mentioned in studies [9-11], emphasising the importance of an integrated approach to the development of recommender systems.

The implementation of the proposed approach on the MovieLens 100K dataset has shown its practical applicability and effectiveness. The use of detailed data analysis, including descriptive statistics, visualisation, anomaly detection, missing value processing, and correlation analysis, allowed us to better understand the features of the dataset and prepare it for effective modelling.

As a result, this study demonstrates that the use of ensemble machine learning methods, such as XGBoost, combined with careful data preparation and feature engineering, can significantly improve the accuracy of film recommendation systems. This opens up wide prospects for further research in this area, including the development of new methods to improve prediction accuracy, as well as the adaptation of the system to different conditions and user needs.

References

[1] L. Breiman, Random Forests. Machine Learning 45 (2001) 5-32. https://doi.org/10.1023/A:1010933404324

[2] Y. Freund, R. E. Schapire, A decision-theoretic generalization of on-line learning and an application to boosting. Journal of Computer and System Sciences 55 (1997) 119–139. https://doi.org/10.1006/jcss.1997.1504

[3] I. M. Sukarsa, N. N. Pandika Pinata, N. K. Dwi Rusjayanthi, N. W. Wisswani, Estimation of Gourami supplies using gradient boosting decision tree method of XGBoost. TEM Journal 10 (2021) 144–151. https://doi.org/10.18421/tem101-17

[4] J. Lyu, P. Zheng, Y. Qi, G. Huang, LightGBM-LncLoc: A LightGBM-Based Computational Predictor for Recognizing Long Non-Coding RNA Subcellular Localization. Mathematics 11 (2023) 602. https://doi.org/10.3390/math11030602

[5] L. Qian, Z. Chen, Y. Huang, R. J. Stanford, Employing categorical boosting (CatBoost) and meta-heuristic algorithms for predicting the urban gas consumption. Urban Climate 51 (2023) 101647. https://doi.org/10.1016/j.uclim.2023.101647

[6] Z. Wang, S. Ameenuddin Irfan, C. Teoh, P. Hriday Bhoyar, Gradient Boosting. In Numerical Machine Learning (pp. 116–159). BENTHAM SCIENCE PUBLISHERS, 2023. https://doi.org/10.2174/9789815136982123010007

[7] Support Vector Machines for Regression. (n.d.). In Support Vector Machines (pp. 330–351). Springer New York, 2008. https://doi.org/10.1007/978-0-387-77242-4_9

[8] Y. Bengio, A. Courville, I. Goodfellow, Deep Learning. MIT Press, 2016. https://www.deeplearningbook.org/

[9] S. S. Choudhury, S. N. Mohanty, A. K. Jagadev, Multimodal trust based recommender system with machine learning approaches for movie recommendation. International Journal of Information Technology 13 (2021) 475-482. https://doi.org/10.1007/s41870-020-00553-2

[10] C. Biancalana, F. Gasparetti, A. Micarelli, A. Miola, G. Sansonetti, Context-aware movie recommendation based on signal processing and machine learning. Proceedings of the 2nd Challenge on Context-Aware Movie Recommendation CAMRa '11, October 2011, pp. 5-10. https://doi.org/10.1145/2096112.2096114

[11] J. Lund, Y.-K. Ng, Movie Recommendations Using the Deep Learning Approach. Proceedings of the 2018 IEEE International Conference on Information Reuse and Integration for Data Science (IRI), 2018, pp. 47-54. IEEE. https://doi.org/10.1109/iri.2018.00015

[12] MovieLens 100K Dataset. (n.d.). GroupLens. URL: https://grouplens.org/datasets/movielens/100k/

[13] V. Golovko, Y. Savitsky, T. Laopoulos, A. Sachenko and L. Grandinetti, Technique of learning rate estimation for efficient training of MLP, Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks, IJCNN 2000, Neural Computing: New Challenges and Perspectives for the New Millennium, Como, Italy, 2000, vol. 1, pp. 323-328. https://doi.org/10.1109/IJCNN.2000.857856

[14] S. Anfilets, S. Bezobrazov, V. Golovko, A. Sachenko, M. Komar, R. Dolny, V. Kasyanik, P. Bykovyy, E. Mikhno, & O. Osolinskyi, Deep multilayer neural network for predicting the winner of football matches. International Journal of Computing 19 (2020) 70-77. https://doi.org/10.47839/ijc.19.1.1695

[15] I. Paliy, A. Sachenko, V. Koval and Y. Kurylyak, "Approach to Face Recognition Using Neural Networks," 2005 IEEE Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, Sofia, Bulgaria, 2005, pp. 112-115, https://doi.org/10.1109/IDAACS.2005.282951

[16] R. Gramyak, H. Lipyanina-Goncharenko, A. Sachenko, T. Lendyuk, D. Zahorodnia, Intelligent Method of a Competitive Product Choosing based on the Emotional Feedbacks Coloring. In IntelITSIS, 2021, pp. 246-257. https://ceur-ws.org/Vol-2853/paper31.pdf

[17] H. Lipyanina, S. Sachenko, T. Lendyuk, V. Brych, V. Yatskiv, O. Osolinskiy, (2021). Method of detecting a fictitious company on the machine learning base. In International Conference on Computer Science, Engineering and Education Applications (pp. 138-146). Cham: Springer International Publishing (2021). https://doi.org/10.1007/978-3-030-80472-5_12

[18] H. Lipyanina, V. Maksymovych, A. Sachenko, T. Lendyuk, A. Fomenko, I. Kit, Assessing the investment risk of virtual IT company based on machine learning. In International Conference on Data Stream Mining and Processing (pp. 167-187). Cham: Springer International Publishing (2020). https://doi.org/10.1007/978-3-030-61656-4_11

[19] V. Turchenko, E. Chalmers, & A. Luczak, A deep convolutional auto-encoder with pooling – unpooling layers in caffe. International Journal of Computing 18 (2019) 8-31. https://doi.org/10.47839/ijc.18.1.1270

[20] A. R. Marakhimov, & K. K. Khudaybergenov, Approach to the synthesis of neural network structure during classification. International Journal of Computing 19 (2020) 20-26. https://doi.org/10.47839/ijc.19.1.1689