Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 235 (2024) 347–360

Procedia Computer Science

www.elsevier.com/locate/procedia

International Conference on Machine Learning and Data Engineering (ICMLDE 2023)

Enhanced Recommender Systems with the Removal of Fake User Profiles

Jagjeet Suryawanshi[a,*], Saifulla Md.Abdul[a], Rajendra Prasad Lal[a], Amarajyothi Aramanda[a,b,*], Nazrul Hoque[c], Nooraini Yusoff[d]

[a]*SCIS, University of Hyderabad, Hyderabad, 500046, Telangana, India*
[b]*IDRBT, Masab Tank, Hyderabad, 500057, Telangana, India*
[c]*Department of Computer Science, Manipur University, Imphal, 795003, Manipur, India*
[d]*Universiti Malaysia Kelantan, Kota Bharu, 16100, Kelantan, Malaysia*

Abstract

The rapid expansion of information resources on the internet has made it increasingly challenging for users to get valuable and relevant information. Recommendation systems (RSs) play a vital role in addressing this issue by providing personalized suggestions based on user preferences. However, the open nature of RSs allows for the injection of fake profiles by malicious users, leading to biased ratings and manipulation of the overall system results. These fake profiles have been intended to promote or degrade specific items, which creates bias in the system and compromises the user experience. The motivation behind these fake profiles is to promote or degrade a particular item, which affects the system and makes it more biased toward certain items. This research paper aims to detect and eliminate such fake profiles in RSs, ensuring users receive genuine and reliable results. The proposed approach utilizes a voting ensemble (VE) method, combining the strengths of multiple classification algorithms including Quadratic Discriminant Analysis (QDA), K-Nearest Neighbor (KNN), and Naive Bayes classifier (NBC). By utilizing these classifiers, the system enhances its robustness against malicious activities. The proposed VE method effectively identifies the fake profiles with a minimum number of parameters, and it shows high detection accuracy on Movielens datasets. It has been demonstrated that the proposed method gives 99.4%, 99.5%, 99.4% and 99.6% accuracy using NBC, QDA, KNN, and VE, respectively and it outperforms all other competing methods in terms of accuracy, precision, recall and MCC.

Peer-review under responsibility of the scientific committee of the International Conference on Machine Learning and Data Engineering

Keywords: Recommendation Systems; Fake Profiles; Malicious Users; Voting Ensemble Method; Shilling Attacks; Classification algorithms; KNN; Naive Bayes Classifier;

* Corresponding author. Tel.: +919340699366; +919885267996.
E-mail address: jagjeet2907@gmail.com; amarajyothi.aramanda@gmail.com

1877-0509
Peer-review under responsibility of the scientific committee of the International Conference on Machine Learning and Data Engineering
10.1016/j.procs.2024.04.035

1. Introduction

In today's information-driven world, users often face difficulties in finding the desired results among the huge amount of available information. Recommender systems (RSs) have emerged as invaluable tools to assist users by offering personalized recommendations based on their preferences and past behavior. These systems find applications in diverse domains, including suggesting articles, books, and movies, and facilitating e-commerce experiences. There are three different approaches in RSs to suggest the items, viz., content-based filtering (CBF) [1], collaborative filtering (CF) [2], and hybrid filtering [3]. The CBF system analyzes the characteristics and attributes of items to generate personalized recommendations. It provides recommendations based on the user's previous interactions with similar items by taking into consideration the content of the items themselves, such as genre, keywords, or metadata.

On the other hand, the CF technique focuses on finding similarities between users to make recommendations [4]. The CF identifies users with similar preferences and behaviors and suggests items based on the experiences of those similar users. However, the openness of CF-based systems leaves room for malicious users to exploit them by injecting fake ratings. This manipulation of recommendation results leads to biased outcomes, adversely degrading the user experience.

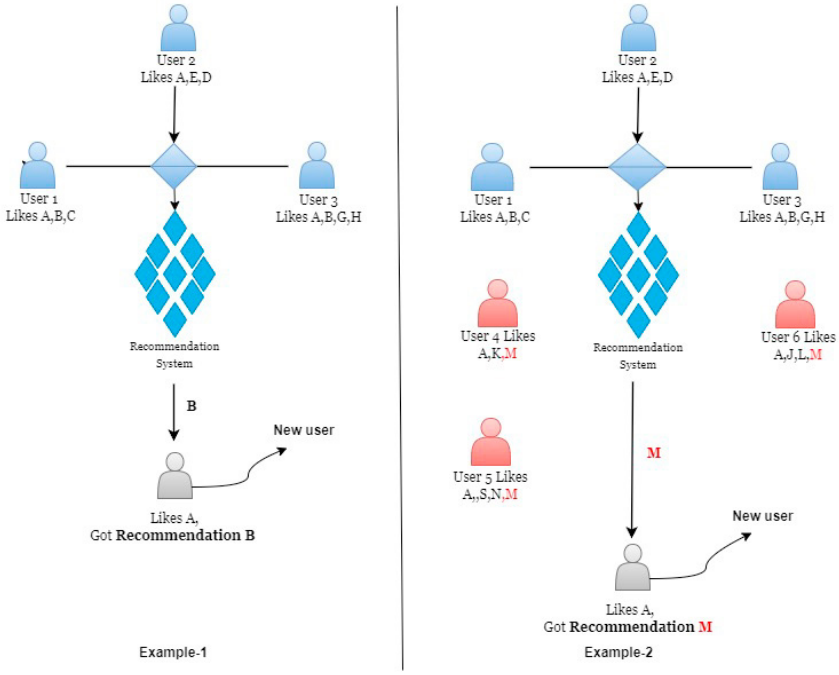

Fig. 1. Example 1 and 2 demonstrate how collaborative recommendation systems work with genuine and malicious users.

Fig. 1 provides an illustration of how the CF system works with normal and malicious users. In this scenario, there are three genuine users: *User 1*, who likes items *A*, *B*, and *C*; *User 2*, who likes items *A*, *E*, and *D*; and *User 3*, who shows interest in items *A*, *B*, *G* and *H*. Now, a new user who expresses interest in item *A* arrives. According to this user's preferences, the RSs suggest item *B*, which aligns with their interests.

However, the presence of malicious users in the CF system can significantly impact the recommendation process, as depicted in Fig. 1. In this case, along with the three genuine users, three malicious users have also joined the system *User 4*, who likes items *A*, *K*, and *M* *User 5*, who likes items *A*, *S* , *N* and *M* and *User 6*, who is interested in items *A*, *J*, *L* and *M*. When a new user enters the system with a preference for item *A*, the RSs will suggest item *M* instead. Since it is a false recommendation, it's also likely that the user won't enjoy the suggested item, which degrades the user experience.

The primary purpose for such behaviour is to promote or degrade a specific item, creating imbalances in the RSs and influencing its overall behavior. This practice of introducing false profiles into the system is commonly known

as shilling attacks (SA), with the profiles themselves referred to as fake profiles or shilling profiles. The detection and processing of these fake profiles are of the highest importance to ensure that users receive authentic and reliable recommendations.

Therefore, in this research, our primary objective is to tackle the problem of fake profiles in RSs by proposing a robust solution using a *voting ensemble* (VE) method. This approach leverages the collective strength of multiple classification algorithms, including naive bayes classifier (NBC), k-nearest neighbors (KNN), and quadratic discriminant analysis (QDA) [5]. Together, these classifiers improve the system's ability to resist malicious activity and effectively identify and eliminate fake profiles. The NBC assumes that features are conditionally independent and learns the probability distribution associated with genuine and fake profiles. It uses this distribution to calculate the probability that a profile belongs to a particular class, making it easy to classify accurately. KNN, a non-parametric classifier, assigns class labels based on the proximity of profiles in the feature space. It uses a voting mechanism to determine the class of a profile by locating its K-nearest neighbors. This assists in the detection of fake profiles by taking into account local patterns and groups of profiles. On the contrary, QDA assumes a quadratic decision boundary between classes. QDA analyzes the characteristics of real and fake profiles separately and calculates the probability of a new profile belonging to each category.

Our proposed approach combines the predictions of these classifiers using a VE method, capitalizing on their individual strengths to make final decisions. By comparing the performance of the ensemble model against various existing detection techniques, it has been demonstrated that the proposed approach outperforms all other methods, showing its effectiveness in detecting and eliminating fake profiles in RSs.

The rest of this paper is organized as follows. Section 2 briefly describes the related work carried out in the field of RSs, SA, and the methods used for identifying SA. Then, Section 3 presents a detailed explanation of the proposed approach and its framework, including its principles and how it works. Further, Section 4 comprises the experimental setup, the dataset used, detection methods for SA, and the obtained results. Finally, Section 5 concludes with a concise summary of the main findings and contributions while outlining future research directions.

2. Related Work

In the realm of detecting and eliminating fake profiles in RSs, several studies have been conducted. An overview of the relevant studies done in this area is provided in this section.

2.1. Recommender Systems

RSs have been extensively studied and developed to improve the user experience by providing personalized recommendations. Several approaches have been explored to develop effective RSs based on user preferences and past behavior.

One of the widely used approaches in RSs is CF [4]. To produce precise recommendations, Sarwar et al. proposed a neighborhood-based collaborative filtering algorithm that computes item-item similarity to generate accurate recommendations [2]. This approach has been widely adopted and forms the foundation for many RSs. Conversely, matrix factorization techniques capture latent factors to model user-item interactions [6]. Techniques such as Singular Value Decomposition (SVD) and Probabilistic Matrix Factorization (PMF) have proven successful in improving recommendation accuracy. Factorization machines have been proposed as a versatile tool for RSs [7]. These machines model pairwise interactions between user and item features, allowing them to incorporate various types of information and ultimately enhance the performance of recommendations.

The CBF approach analyzes item attributes and user profiles to generate recommendations. Pazzani and Billsus provided insights into CBF methods, including TF-IDF weighting, cosine similarity, and term-based profiles [1]. These techniques utilize item characteristics to recommend relevant items to users. The application of deep learning in RSs has gained attention, with Zhang et al. discussing the use of neural network architectures like Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) to capture intricate patterns and representations, resulting in accurate recommendations [8].

Recently, there has been an emphasis on hybrid RSs that combine multiple approaches. Burke investigated the integration of CF, CBF, and knowledge-based methods, demonstrating the benefits of hybrid systems [3]. By leverag-

ing the strengths of different techniques, hybrid systems offer improved recommendation performance. These studies lay the foundation for understanding the existing methodologies and techniques used in RSs. However, despite the significant progress in recommendation algorithms, the presence of malicious activities, such as SA, poses a challenge to the reliability and trustworthiness of RSs. Therefore, the next section will focus on addressing these challenges and propose a robust solution to detect and eliminate fake profiles in RSs, ensuring users receive authentic and reliable recommendations.

2.2. Types of Shilling attacks

SA poses a significant threat to the integrity and effectiveness of RSs. In these attacks, malicious users insert attack profiles into the system, consisting of biased ratings associated with fake user identities. The objective of such attacks is to manipulate recommendation results by promoting or demoting specific target items [9, 10]. The profiles involved in such attacks are known as shilling profiles.The standard profile format of attacker [11] consists of set of filler items, selected items, target and unrated items as shown in Fig. 2.

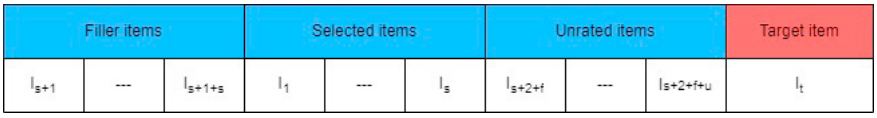

Fig. 2. The standard attacker profile format

SA encompass a variety of strategies and characteristics [9]. Several well-known attack models have been identified.

Table 1. Attack Strategies and Ratings

Attack Strategy	Selected Items	Target Items	Filler Items
Bandwagon Attack	Popular items	R_{max}	System mean (normal distribution)
Random Attack	None	R_{min} (nuke) or R_{max} (push)	System mean (normal distribution)
Reverse-Bandwagon	Least popular	R_{min}	System mean (normal distribution)
Hate/Love Attack	None	R_{min} (Hate), R_{max} (Love)	R_{max} (normal distribution)
Average Attack	None	R_{min} or R_{max}	Item mean (normal distribution)

Table 1 summarizes different SA strategies used in RSs [12]. It presents the types of attacks, the items targeted, and the ratings assigned. Additionally, it provides details on the ratings given to target and filler items, including Rmax (maximum), Rmin (minimum), and normal distributions according to the system mean or item mean ratings.

2.3. Existing Detection Approaches

In their research paper, Zhang et al. propose UD-HMM, an unsupervised method for detecting SA in RSs [10]. The UD-HMM method combines a hidden markov model with hierarchical clustering to identify suspicious user profiles. By modeling sequential user behavior using the HMM, the method captures patterns indicative of SA. Hierarchical clustering is employed to group similar profiles based on behavior patterns, aiding in distinguishing them from genuine profiles.

Another innovative approach is presented in [13], where the authors propose a novel technique using graph analysis for identifying these attacks. The authors introduce a graph model that represents the relationships between users and items in the RSs. By analyzing the connectivity patterns and graph properties, such as the clustering coefficient and degree centrality, the proposed approach can identify suspicious users involved in shilling attacks.

Panagiotakis et al. [14] introduce both unsupervised and supervised methods for detecting hurriedly created profiles in RSs. The unsupervised approach utilizes clustering techniques, specifically the K-means algorithm, to group profiles based on their characteristics. This clustering analysis helps identify clusters that exhibit similar patterns, which could indicate the presence of SA. On the other hand, the supervised method involves training a classifier using carefully extracted features from user profiles to differentiate between genuine and hurriedly created profiles.

Bilge et al. [15] propose a unique approach for detecting SA in RSs. The method is specifically designed to detect specific types of attacks, such as bandwagon, segment, and average attacks. The approach makes use of a binary decision tres (DTs) leaf node to group the attack profiles together using a splitting k-means clustering algorithm.

A two-phase detection method for SA [16]. Previous research has focused on individual profile differences without considering group characteristics in attack profiles. The Borderline-SMOTE approach is utilised in the first stage. A preliminary detection result is provided at this stage. In the second stage, the target items within the list of probable attack profiles are analyzed.

Cai and Zhang [17] propose a detection method that analyzes user rating behavior in RSs to identify suspicious activities associated with SA. By examining various characteristics, such as the distribution of ratings, rating consistency, and rating velocity, the proposed method aims to distinguish genuine user behavior from malicious activities.

Zhang et al. [18] propose an approach that utilizes Principal Component Analysis (PCA) and data complexity analysis to identify suspicious activities associated with SA.

2.4. Ensemble Learning Methods

Various models combined using the effective machine learning technique known as ensemble learning to produce a more robust and precise predictive model. Ensemble approaches seek to increase generalization and robustness by utilizing the diversity and combined intelligence of several models.

This method operates on the principle that aggregating the predictions of multiple models can yield better results than relying on a single model. Some of the widely used ensemble learning methods are, viz., i) *bagging*, ii) *boosting*, iii) *stacking*, and iv) *voting ensemble*. These methods are explained in the following.

i) ***Bagging***: By using bootstrap sampling, bagging involves breaking up the initial training data into numerous subsets [19]. Each subset is used to train a separate base model. The results of the individual model's predictions are averaged or voted on to generate the final projection.

ii) ***Boosting***: It seeks to enhance the effectiveness of weak base models by sequentially training them in a way that emphasizes the incorrectly categorized instances [20]. Each new model builds on the instances that the prior models had struggles with, creating an ensemble model that is powerful and reliable.

iii) ***Stacking***: It involves training a meta-model on the outputs of several base models to integrate their predictions [21]. The meta-model learns to create the final prediction using the input features from the underneath models. In recent years, stacking has gained more attention.

iv) ***Voting ensemble***: It takes a majority vote for classification or combines for regression to incorporate the results from multiple models' predictions [22]. There are two ways these VE models can be used such as *majority VE* and *soft VE*.

Majority VE: The *majority VE* [22] is a type of ensemble learning model where multiple base models are trained independently on the same dataset. Every base model provides predictions on unseen data, and the ensemble's final prediction is determined by a majority vote.

Soft VE: In soft VE, where results are decided by averaging or taking a weighted average of the anticipated probability from all models, with each model's prediction being treated as a vote [22]. When the predictions from the models are probabilities or continuous values, soft voting is frequently used. For the purpose of identifying SA in RSs, Zhang and Chen [23] present an ensemble approach. The technique makes use of the item's ordered sequences to identify suspicious user behaviour. The authors present an ensemble framework that integrates many basic classifiers that have each been trained on various facets of ordered item sequences [23]. The frequency of item occurrences, the temporal patterns of item selections, and the similarity across item sequences are a few examples of these aspects. It makes use of the diversity of the base classifiers to identify various aspects of SA in ordered item sequences.

3. Proposed Method

In this section, we proposed an effective method to detect and eliminate fake profiles in RSs, addressing the problem of biased ratings and manipulation of recommendation results caused by fake profiles injected by malicious users.

Our proposed approach utilizes a *VE* method that combines multiple classification algorithms, including QDA, NBC, and KNN. By leveraging these classifiers, the system enhances its robustness against malicious activities and improves its accuracy in identifying and eliminating fake profiles.

The proposed framework, depicted in Fig. 3, incorporates a soft VE method to combine the predictions from the classification models. This ensemble method takes into account the confidence levels of the predictions and assigns appropriate weights to each prediction. By considering the collective opinion of the models, the ensemble method makes a final decision on whether a profile is classified as fake or genuine.

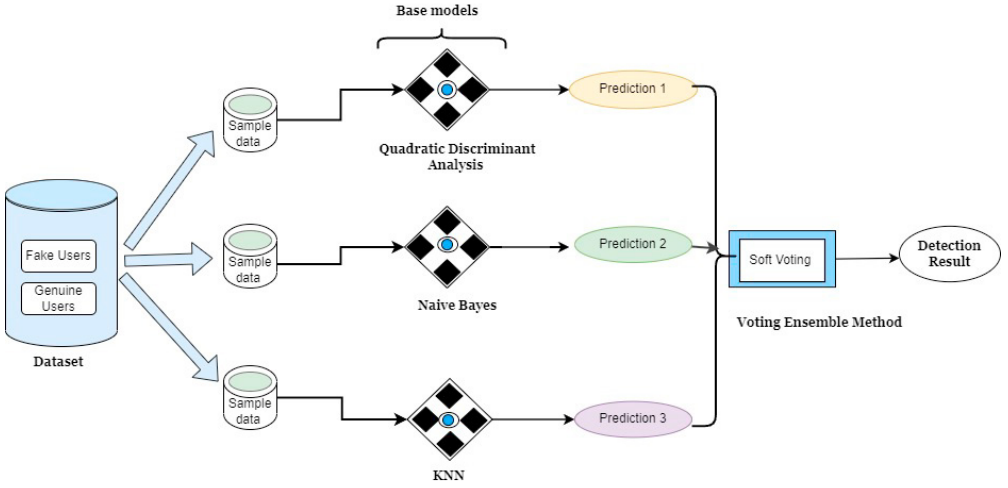

Fig. 3. Proposed Framework

Algorithm 1, presented in this research paper, aims to address the issue of fake profiles within RSs. The purpose of the algorithm is to detect and eliminate these malicious profiles, ensuring the delivery of reliable and genuine recommendations to users. The algorithm takes three inputs: B, N, and parameters, and produces two output lists: F and G, representing fake and genuine profiles, respectively.

The execution of the algorithm begins by constructing user and item sets from the user-item interaction data, which is represented by matrix B. These sets serve as the basis for subsequent operations. Additionally, two lists, G and F, are initialized to store the detected genuine and fake profiles, respectively. Moving forward, the algorithm proceeds with iterating over each user u in the user set U. For each u, it extracts user-specific features by analyzing their interactions with items from matrix B. Furthermore, a vote count is initialized for each item present in the item set I. To classify the user, u's, interaction with each item, i, the algorithm employs N classification algorithms. Each algorithm independently evaluates the nature of the interaction, determining whether it is genuine or not. In case a classification algorithm identifies a genuine interaction, the vote count for the corresponding i is incremented.

Upon completing the classification process for all items, the algorithm sorts the items in descending order based on their respective vote counts. The item(s) with the highest vote count is selected as the preferred item(s) for the u. Subsequently, the algorithm calculates the ratio between the highest vote count and the total number of votes received. By comparing this ratio to a predefined threshold, the algorithm determines the authenticity of the user profile. If the ratio exceeds the threshold, the u is considered a genuine profile and is added to the list G. Conversely, if the ratio falls below the threshold, the u is classified as a fake profile and added to the list F.

Finally, as the execution of the algorithm concludes, it returns the lists G and F, representing the detected genuine and fake profiles, respectively.

Algorithm 1 Fake Profile Detection and Elimination in Recommendation Systems

Require: $B, N, Parameters$
Ensure: F, G
Construct U and I from B
Initialize G , F
for each user $u \in U$ **do**
    Extract user features from user-item interactions in B for u
    Initialize a vote_count for each item in I
    for each item $i \in I$ **do**
        for $n = 1$ to N **do**
            Apply classification algorithm n
            if result is genuine **then**
                Increment the vote count for i
            end if
        end for
    end for
    Sort items in decreasing order
    Select the items with the highest vote count
    Compute vote ratio
    if ratio exceeds a predefined threshold **then**
        Add u to G
    else
        Add u to F
    end if
end for
Return G and F

4. Experiment Results

In this section, we provide experimental configuration, experiment results, and performance analysis.

4.1. Experimental configuration

For the experimentation configuration, two datasets were utilized: MovieLens 100k and MovieLens-1M. These datasets were selected due to their high density, ensuring a sufficient number of ratings for the analysis. The MovieLens-100k dataset contains 100,000 ratings provided by 1,004 users for 1,700 movies, with ratings ranging from 1 to 5. In contrast, the MovieLens-1M dataset comprises 1 million ratings from 6,000 users for 4,000 movies. Additionally, each rating in both datasets is associated with a timestamp, indicating when it was given by the user.

To simulate the presence of fake profiles, we implemented various SA models, including random, bandwagon, average, and reverse-bandwagon attacks. These attacks were conducted using different filler sizes, specifically 30, 60, 90, 120, and 150, and attack sizes represented as percentages 3, 6, 9, 12, and 15. Our experimental setup closely follows the methodology outlined in a previous study [24], and we implemented the techniques using the Python programming language.

For evaluation, stratified cross-validation was employed to ensure a proportional representation of each class within the sample data. The evaluation process utilized 5-fold stratified cross-validation, dividing the data into 5 folds. Each fold served as a separate testing set, while the remaining 4 folds were used for training the models. Considering the combinations of 5 attack sizes, 4 different attacks, and 2 datasets, a total of 40 files were generated to represent unique configurations for evaluation.

The experiment was designed in this manner to allow for a complete evaluation of the performance of the suggested techniques. By employing stratified cross-validation, we were able to validate the approach's effectiveness across a range of attack and filler sizes, carefully evaluating how well the technique performs and how effectively it can handle various types of attacks.

4.2. Detection systems

The methods used in this research work for the detection of fake user profiles in the RSs include:

4.2.1. Proposed detection method

- **VE**: The VE is a popular ensemble method that uses the collective wisdom of multiple classifiers to make predictions. A group of classifiers is trained independently in this method, and the final prediction is formed by aggregating the classifier's predictions or conclusions. It can be implemented in two ways majority VE (hard voting) and soft VE.

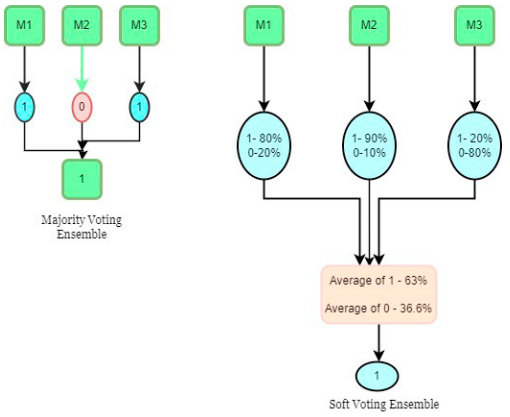

Fig. 4. Types of VE method

Each classifier contributes a discrete vote in majority voting, whereas the classifiers provide continuous probabilities in soft voting represented in Fig. 4. Because of the difference in voting systems, the soft VE can detect deeper patterns in the data and make more informed predictions.
After evaluating the performance of the detection methods, KNN, QDA, and NBC emerged as top performers in detecting fake profiles. A **soft voting ensemble** method was used to leverage their individual strengths. Each algorithm independently classified profiles and assigned probabilities to each class. The final classification was determined by aggregating these probabilities using weighted voting, where the weights were based on the confidence of each algorithm's prediction. The soft VE method demonstrated exceptional efficacy in detecting and eliminating fake profiles in RS, outperforming existing techniques.

- **KNN:** It is an unconventional algorithm that allocates class labels based on its nearest neighbor's majority class [25]. To detect fake profiles, KNN compares the profiles in the feature space and classifies them based on the predominant class among their nearest neighbors. This approach leverages local patterns and groups of profiles to effectively identify fake profiles.
- **NBC:** Naive Bayes is a probabilistic classifier that calculates the likelihood of a profile belonging to the genuine or fake class based on probabilities derived from the training data [26]. It assumes that the features are conditionally independent, simplifying the calculation of probabilities. The profile is then classified based on the higher likelihood.
- **QDA:** QDA eases the assumption of equal covariance matrices for different classes [5], allowing for a more flexible decision boundary. It analyzes the statistical properties of genuine and fake profiles separately and calculates the probability of a new profile belonging to each class. The category with the greatest probability is used by QDA to determine the label.

4.2.2. Existing detection methods

- Logistic regression (LR): It is a linear classification algorithm [27] that works by analyzing the features associated with each profile and calculating the probability of a profile being a fake profile. It achieves this by taking

the weighted sum of the profile's features and applying a logistic function to transform the probability into a binary classification.

- SVM: SVM examines the characteristics and patterns of genuine and fake profiles to identify the optimal hyperplane that separates the two classes [28]. It constructs a decision boundary by maximizing the margin between the classes, effectively distinguishing between genuine and fake profiles.
- Linear discriminant analysis (LDA): analyzes the statistical properties of genuine and fake profiles [29]. It finds linear combinations of features that maximize the separation between the classes. By projecting profiles onto this linear subspace, LDA effectively distinguishes between genuine and fake profiles.
- XgBoost (Xgb), AdaBoost (Adab), and Catboost (Catb): These boosting ensemble methods combine the predictions of multiple weak classifiers to make final decisions [30]. Xgb sequentially builds DTs and corrects the errors of previous models, while Adab adjusts the weights of weak classifiers based on their performance. Catb, specifically designed for handling categorical features, is effective in the context of fake profile detection.
- Decision tree (DTs) is a tree-based algorithm that analyzes profile features by splitting the data based on different features. It constructs a tree model with nodes representing features, branches representing decisions, and leaf nodes indicating class labels. By examining feature patterns, DTs identify potential indicators of fake profiles.
- Random forest (RF) is an ensemble method that combines multiple DTs for prediction [23]. It selects random subsets of features and data samples to build each tree, reducing overfitting. By aggregating the predictions of these trees, RF identifies patterns in the data, enabling effective detection of fake profiles.

4.3. Performance analysis

To evaluate the effectiveness of our classification-based models, we have used key metrics: Matthews Correlation Coefficient (MCC), accuracy, precision (Pre), recall (Rec), and F1-score.

MCC: The MCC provides a more complete understanding of the model's performance by taking into consideration false negatives (FN), true negatives (TN), false positives (FP), and true positives (TP). This enlarged set of evaluation measures makes sure that our models are thoroughly evaluated across a variety of dimensions, which eventually enhances the clarity of our findings. The MCC is calculated using the formula given in Eq. (1).

$$\text{MCC} = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{1}$$

Table 2. Evaluation of MCC in detecting Random Attack

	Algorithms											
	LR	NBC	SVM	LDA	QDA	Catb	Adab	Xgb	DT	RF	KNN	VE
3%	0.99	**0.427**	0.0	0.0	**0.49**	0.096	0.095	0.096	0.096	0.096	**0.61**	**0.47**
6%	0.242	**0.762**	0.0	0.063	**0.889**	0.188	0.188	0.188	0.188	0.188	**0.804**	**0.891**
9%	0.200	**0.839**	0.061	0.374	**0.843**	0.273	0.272	0.272	0.272	0.273	**0.883**	**0.937**
12%	0.343	**0.905**	0.271	0.511	**0.903**	0.349	0.349	0.348	0.349	0.349	**0.946**	**0.963**
15%	0.46	**0.939**	0.484	0.706	**0.935**	0.413	0.412	0.413	0.413	0.413	**0.952**	**0.970**

The MCC comparison among the algorithms, as shown in Table 2, clearly shows the VE method's superior performance, particularly in scenarios with larger attack sizes. For instance, algorithms like LR that showed good efficiency with smaller attack sizes saw performance decline as attack sizes increased. In contrast, the proposed method showcased unwavering superior performance across various attack sizes.

F1-score: The F1 score is defined in Eq. (2). The comparison depicted in Fig. 5 and Fig. 6 showcase the performance of the proposed method along with other algorithms, including LDA, LR, and Xgb, under varying attack sizes. Ini-

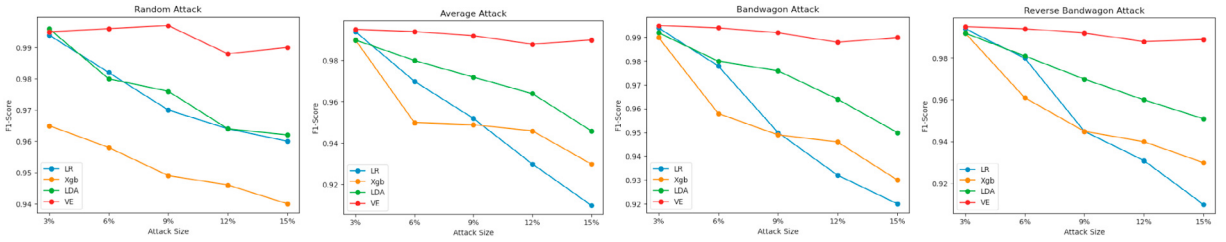

Fig. 5. F1-Score comparison on MovieLens-100k dataset

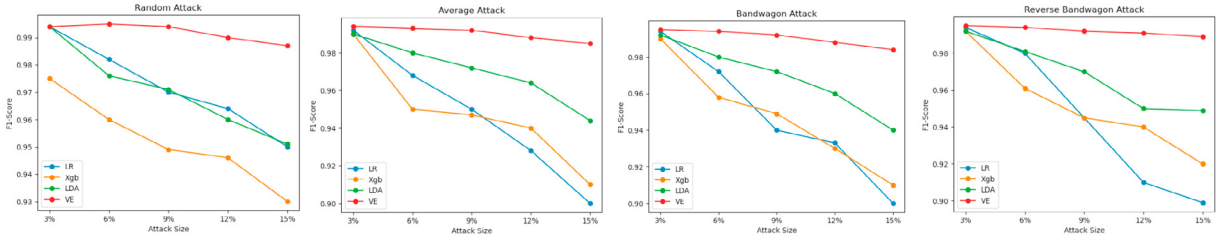

Fig. 6. F1-Score comparison on MovieLens-1M dataset

tially, when the attack size is small, these algorithms, including the proposed method, demonstrate relatively good performance. However, as the attack size increases, there is a sudden and noticeable decrease in the performance of the other algorithms, such as LDA, LR, and Xgb shown in Fig. 5 and Fig. 6.

$$\text{F1-score} = \frac{2 * \text{Pre} * \text{Rec}}{\text{Pre} + \text{Rec}} \tag{2}$$

Now, we present the experiment results and its analysis using precision, recall and accuracy metrics.
Precision: It determines how well the model performed in making accurate predictions. Precision indicates the model's capability to minimize FP and make accurate positive predictions and is defined in Eq. (3).

$$\text{Pre} = \frac{\text{TP}}{\text{FP} + \text{TP}} \tag{3}$$

Recall: Often referred to as sensitivity, evaluates the ability of the model to identify every positive instance. It provides insights into the model's capacity to minimize FN and accurately capture positive instances and is defined in Eq. (4).

$$\text{Rec} = \frac{\text{TP}}{\text{FN} + \text{TP}} \tag{4}$$

Accuracy: It relates to how close or accurate a measurement, prediction, or result is to the actual or expected value. The accuracy is defined in Eq. (5).

$$\text{Acc} = \frac{\text{Correct Predictions}}{\text{Total Predictions}} \tag{5}$$

These metrics offer valuable insights into the performance of our models. By considering these metrics together, we can assess the effectiveness of our classification-based models accurately.

During the performance analysis, it was observed that when the attack size was small, most of the algorithms exhibited satisfactory performance in detecting fake profiles. However, as the attack size increased, a notable decline in the performance of several models was observed. These models started misclassifying fake profiles, compromising the effectiveness of the detection process.

In contrast, the proposed techniques maintained their accuracy and consistency even with the increase in attack size. It outperformed the other models that initially showed promising results but faltered as the attack size grew.

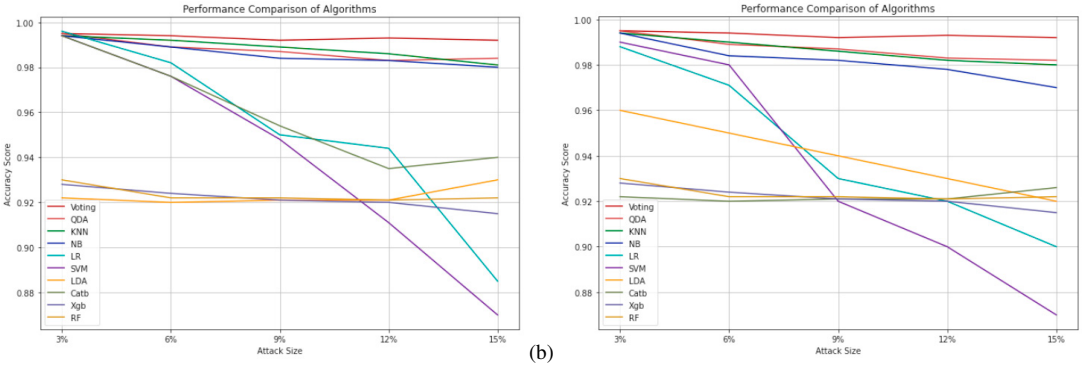

Fig. 7. Performance Comparison of methods on (a) MovieLens-100k and (b) MovieLens-1M using accuracy score

The accuracy comparison shown in Fig. 7 clearly highlighted the advantage of utilizing the *VE*, as it consistently achieved high accuracy across varying attack sizes. To evaluate the performance of the detection methods under different attack sizes, precision and recall values were measured and recorded for each attack type: random attack, bandwagon attack, average attack, and reverse-bandwagon attack.

The following tables present the precision, recall, and accuracy values for each attack type using MovieLens-100k dataset.

Table 3. Compare the effectiveness of the model's precision, recall, and accuracy in detecting Random Attack

attac		**3%**			**6%**			**9%**			**12%**			**15%**	
Algo	Pre	Rec	Acc	Pre	Rec	Acc	Pre	Rec	Acc	Pre	Rec	Acc	Pre	Rec	Acc
LR	0.996	0.999	0.996	0.978	0.99	0.982	0.964	0.98	0.95	0.962	0.97	0.944	0.926	0.942	0.885
NBC	**0.995**	**0.999**	**0.994**	**0.989**	**0.99**	**0.989**	**0.985**	**0.99**	**0.98**	**0.986**	**0.99**	**0.985**	**0.984**	**0.982**	**0.983**
SVM	0.994	0.998	0.994	0.976	0.99	0.976	0.948	0.99	0.948	0.971	0.99	0.911	0.876	0.997	0.87
LDA	0.994	0.999	0.994	0.976	0.99	0.976	0.959	0.99	0.954	0.943	0.98	0.935	0.948	0.983	0.94
QDA	**0.995**	**0.999**	**0.995**	**0.989**	**0.99**	**0.989**	**0.985**	**0.99**	**0.986**	**0.981**	**0.99**	**0.983**	**0.984**	**0.982**	**0.985**
Catb	0.998	0.922	0.922	0.997	0.92	0.921	0.996	0.92	0.922	0.989	0.92	0.921	0.995	0.922	0.93
Adab	0.999	0.922	0.922	0.997	0.92	0.922	0.994	0.92	0.921	0.988	0.92	0.922	0.983	0.922	0.918
Xgb	0.999	0.922	0.92	0.997	0.92	0.921	0.994	0.92	0.92	0.989	0.92	0.921	0.984	0.922	0.918
DT	0.998	0.922	0.922	0.997	0.99	0.922	0.994	0.92	0.920	0.986	0.92	0.920	0.983	0.922	0.919
RF	0.999	0.922	0.922	0.997	0.99	0.922	0.99	0.92	0.922	0.989	0.92	0.920	0.984	0.922	0.919
KNN	**0.996**	**0.999**	**0.994**	**0.992**	**0.99**	**0.992**	**0.992**	**0.99**	**0.989**	**0.99**	**0.98**	**0.986**	**0.985**	**0.984**	**0.981**
VE	**0.995**	**0.999**	**0.995**	**0.996**	**0.99**	**0.994**	**0.994**	**0.99**	**0.992**	**0.992**	**0.99**	**0.993**	**0.992**	**0.99**	**0.992**

Table 3 represent the precision, recall, and accuracy values for different attack sizes under the random attack scenario. These values provide insights into the performance of the detection methods in accurately classifying fake

profiles when subjected to *random attacks*. The precision value represents the proportion of correctly classified fake profiles, while the recall value indicates the ability to identify all actual positive instances. Additionally, the accuracy value reflects the overall correctness of the detection methods' predictions.

Table 4. Compare the effectiveness of the model's precision, recall, and accuracy in detecting Bandwagon Attack

attac		**3%**			**6%**			**9%**			**12%**			**15%**	
Algo	Pre	Rec	Acc	Pre	Rec	Acc	Pre	Rec	Acc	Pre	Rec	Acc	Pre	Rec	Acc
LR	0.989	0.99	0.989	0.932	0.979	0.914	0.93	0.94	0.94	0.92	0.93	0.936	0.92	0.895	0.90
NBC	**0.997**	**0.99**	**0.997**	**0.997**	**0.997**	**0.995**	**0.989**	**0.99**	**0.991**	**0.98**	**0.99**	**0.985**	**0.98**	**0.98**	**0.981**
SVM	0.982	0.99	0.982	0.932	0.999	0.932	0.93	0.94	0.94	0.922	0.93	0.91	0.92	0.885	0.875
LDA	0.982	0.99	0.982	0.95	0.993	0.946	0.93	0.94	0.983	0.91	0.93	0.931	0.92	0.895	0.941
QDA	**0.998**	**0.99**	**0.997**	**0.995**	**0.997**	**0.992**	**0.989**	**0.99**	**0.991**	**0.984**	**0.98**	**0.988**	**0.98**	**98**	**0.982**
Catb	0.999	0.92	0.922	0.999	0.922	0.926	0.93	0.94	0.922	0.922	0.919	0.920	0.91	0.895	0.929
Adab	0.999	0.92	0.923	0.999	0.922	0.927	0.93	0.94	0.922	0.91	0.92	0.92	0.91	0.885	0.919
Xgb	0.999	0.92	0.923	0.99	0.922	0.927	0.93	0.94	0.922	0.91	0.922	0.92	0.91	0.89	0.919
DT	0.998	0.92	0.922	0.999	0.922	0.927	0.93	0.94	0.922	0.91	0.922	0.92	0.97	0.922	0.919
RF	0.998	0.93	0.923	0.999	0.922	0.927	0.93	0.94	0.922	0.91	0.922	0.92	0.97	0.93	0.92
KNN	**0.993**	**0.99**	**0.992**	**0.993**	**0.994**	**0.988**	**0.989**	**0.99**	**0.985**	**0.98**	**0.979**	**0.982**	**0.98**	**0.98**	**0.981**
VE	**0.997**	**0.99**	**0.997**	**0.997**	**0.996**	**0.995**	**0.989**	**0.99**	**0.994**	**0.99**	**0.992**	**0.992**	**0.99**	**0.991**	**0.991**

Table 4 indicates the precision, recall, and accuracy values for different attack sizes under the bandwagon attack scenario. These evaluation metrics provide insights into the detection methods' ability to identify fake profiles in the presence of bandwagon attacks.

Table 5. Compare the effectiveness of the model's precision, recall and accuracy in detecting Average Attack

attac		**3%**			**6%**			**9%**			**12%**			**15%**	
Algo	Pre	Rec	Acc	Pre	Rec	Acc	Pre	Rec	Acc	Pre	Rec	Acc	Pre	Rec	Acc
LR	0.997	0.996	0.997	0.981	0.99	0.974	0.962	0.97	0.943	0.95	0.972	0.936	0.969	0.964	0.885
NBC	**0.995**	**0.999**	**0.995**	**0.993**	**0.99**	**0.993**	**0.989**	**0.99**	**0.988**	**0.98**	**0.991**	**0.99**	**0.984**	**0.99**	**0.984**
SVM	0.993	0.99	0.993	0.976	0.99	0.976	0.948	0.99	0.948	0.91	0.999	0.91	0.875	0.999	0.87
LDA	0.993	0.999	0.993	0.976	0.99	0.976	0.959	0.99	0.954	0.93	0.989	0.931	0.953	0.973	0.94
QDA	**0.996**	**0.999**	**0.996**	**0.992**	**0.99**	**0.992**	**0.98**	**0.99**	**0.988**	**0.98**	**0.989**	**0.984**	**0.989**	**0.99**	**0.985**
Catb	0.999	0.922	0.922	0.997	0.92	0.921	0.994	0.92	0.922	0.98	0.922	0.921	0.996	0.993	0.93
Adab	0.999	0.922	0.922	0.997	0.92	0.922	0.994	0.92	0.921	0.98	0.922	0.922	0.984	0.922	0.918
Xgb	0.999	0.921	0.92	0.996	0.92	0.921	0.994	0.92	0.92	0.99	0.922	0.921	0.986	0.922	0.918
DT	0.999	0.922	0.921	0.997	0.92	0.921	0.994	0.92	0.920	0.98	0.922	0.920	0.984	0.921	0.919
RF	0.999	0.922	0.922	0.997	0.92	0.922	0.994	0.92	0.922	0.98	0.922	0.920	0.984	0.922	0.919
KNN	**0.996**	**0.999**	**0.996**	**0.993**	**0.99**	**0.992**	**0.991**	**0.99**	**0.989**	**0.99**	**0.99**	**0.988**	**0.988**	**0.989**	**0.98**
VE	**0.995**	**0.999**	**0.995**	**0.996**	**0.99**	**0.996**	**0.996**	**0.99**	**0.994**	**0.99**	**0.996**	**0.993**	**0.994**	**0.992**	**0.993**

Table 6. Compare the effectiveness of the model's precision, recall, and accuracy in detecting Reverse-bandwagon Attack

attac		**3%**			**6%**			**9%**			**12%**			**15%**	
Algo	Pre	Rec	Acc	Pre	Rec	Acc	Pre	Rec	Acc	Pre	Rec	Acc	Pre	Rec	Acc
LR	0.989	0.99	0.972	0.932	0.97	0.924	0.93	0.93	0.94	0.91	0.92	0.932	0.92	0.882	0.89
NBC	**0.996**	**0.99**	**0.997**	**0.992**	**0.99**	**0.992**	**0.98**	**0.99**	**0.99**	**0.97**	**0.98**	**0.982**	**0.98**	**0.972**	**0.98**
SVM	0.978	0.99	0.982	0.932	0.99	0.95	0.92	0.93	0.94	0.92	0.92	0.911	0.92	0.875	0.87
LDA	0.982	0.99	0.982	0.942	0.989	0.944	0.93	0.93	0.979	0.91	0.92	0.931	0.91	0.895	0.94
QDA	**0.998**	**0.99**	**0.995**	**0.993**	**0.99**	**0.99**	**0.98**	**0.99**	**0.99**	**0.97**	**0.98**	**0.985**	**0.97**	**0.981**	**0.981**
Catb	0.999	0.92	0.922	0.996	0.92	0.926	0.92	0.93	0.922	0.92	0.92	0.922	0.90	0.885	0.92
Adab	0.999	0.92	0.923	0.995	0.93	0.926	0.93	0.94	0.922	0.91	0.92	0.92	0.91	0.875	0.919
Xgb	0.999	0.92	0.923	0.991	0.92	0.927	0.93	0.93	0.922	0.90	0.93	0.92	0.91	0.889	0.919
DT	0.998	0.92	0.922	0.995	0.92	0.927	0.93	0.94	0.922	0.92	0.92	0.92	0.96	0.912	0.919
RF	0.998	0.93	0.923	0.996	0.92	0.927	0.93	0.94	0.922	0.91	0.93	0.92	0.97	0.929	0.922
KNN	**0.993**	**0.99**	**0.99**	**0.993**	**0.99**	**0.984**	**0.98**	**0.99**	**0.982**	**0.98**	**0.97**	**0.98**	**0.98**	**0.978**	**0.979**
VE	**0.997**	**0.99**	**0.995**	**0.957**	**0.99**	**0.994**	**0.99**	**0.99**	**0.994**	**0.99**	**0.99**	**0.992**	**0.99**	**0.991**	**0.991**

Similarly, Table 5 and 6 represent the precision, recall, and accuracy values for different attack sizes under the *average attack* and *reverse-bandwagon attack*, respectively.

Similarly, we have also computed precision, recall, and accuracy using the MovieLens-1M dataset for different attacks, and the results obtained are almost similar to the MovieLens-100k dataset. Due to space constraints, we presented only MovieLens-100k dataset results.

Based on the evaluation results presented in the tables, it is evident that the classification algorithms KNN, NBC, and QDA consistently demonstrate superior performance in detecting fake profiles across different attack scenarios and sizes. These algorithms exhibit high precision, recall, and accuracy values, indicating their effectiveness in accurately detecting fake profiles. To utilize the strengths of these individual algorithms, they were combined to form the VE method. This ensemble approach maximizes the collective decision-making power of KNN, NBC, and QDA to make final predictions.

The proposed method consistently outperforms other detection techniques in terms of precision, recall, and accuracy. This means that it is highly effective in identifying fake profiles from RSs. It provides an efficient and reliable approach to tackle the challenges posed by fake profiles, thereby enhancing the overall integrity and credibility of RSs.

5. Conclusion

The purpose of this research was to address the issue of fake profiles in RSs, which can lead to biased ratings and system manipulation. To overcome this challenge, a novel approach utilizing a VE method was proposed. The approach combined the strengths of multiple classification algorithms, including QDA, NBC, and KNN, to enhance the system's robustness against malicious activities. In comprehensive evaluations against two real-time datasets and comparative assessments against existing detection techniques, this strategy consistently outperformed other methods, demonstrating superior performance in detecting and eliminating fake profiles in RSs. By addressing the issue of fake profiles, the proposed approach ensures that users receive genuine and reliable recommendations, thereby enhancing their overall experience. We conducted experiments on MovieLens datasets. We evaluated our approach using accuracy, precision, recall, and MCC metrics, and the results show that the proposed VE method using NBC, QDA, and KNN as base models performed better than the existing approaches. This research significantly contributes to the field of RSs by providing an effective solution to mitigate the impact of malicious activities and promote fairness in RSs.

As a future work, we are planning to develop an effective hybrid Deep Neural Network (DNN) model incorporating Long Short-Term Memory (LSTM) for identification of fake profiles. By continuously improving the detection and elimination of fake profiles, RSs can continue to provide personalized and unbiased recommendations in the face of evolving malicious activities.

Statements and Declarations

Competing Interests: The authors confirm that they have no known financial or interpersonal conflicts that could have been expected to have an impact on the research presented in this paper.

Authors contribution statement:

- Jagjeet Suryawanshi: Contributor.
- Saifulla Md.Abdul: Supervisor 1.
- Rajendra Prasad Lal: Supervisor 2.
- Amarajyothi Aramanda: Contributor.
- Dr.Nazrul Hoque : Contributor.
- Nooraini Yusoff : Contributor.

Ethical and informed consent for data used: All data/datasets used in this paper are publicly available. Corresponding links and references are provided in the reference section.

Data availability and access: Not applicable

Acknowledgment None. No funding to declare.

References

[1] Lops P, De Gemmis M, Semeraro G. (2011) "Content-based recommender systems: State of the art and trends." *Recommender systems handbook*: 73–105.
[2] Sarwar B, Karypis G, Konstan J, Riedl J. (2001) "Item-based collaborative filtering recommendation algorithms", in *Proc. of the 10th Int. Conf. on World Wide Web*: 285–295.
[3] Burke R. (2002) "Hybrid recommender systems: Survey and experiments." *User Modeling and User-adapted Interaction* **12**: 331–370.
[4] Aramanda A, Abdul SM, Vedala R. (2020) "A comparison analysis of collaborative filtering techniques for recommender systems", in *Proc. of ICCCE*: 87–95.
[5] Tharwat A. (2016) "Linear vs. quadratic discriminant analysis classifier: A tutorial." *International Journal of Applied Pattern Recognition* **3**(2): 145–180.
[6] Koren Y, Bell R, Volinsky C. (2009) "Matrix factorization techniques for recommender systems." *Computer* **42** (8): 30–37.
[7] Rendle S. (2012) "Factorization machines with libfm." *ACM Transactions on Intelligent Systems and Technology* **3** (3): 1–22.
[8] Zhang S, Yao L, Sun A, Tay Y. (2019) "Deep learning based recommender system: A survey and new perspectives." *ACM Computing Surveys* **52** (1): 1–38.
[9] Si M, Li Q. (2020) "Shilling attacks against collaborative recommender systems: A review." *Artificial Intelligence Review* **53**: 291–319.
[10] Zhang F, Zhang Z, Zhang P, Wang S. (2018) "UD-HMM: An unsupervised method for shilling attack detection based on hidden Markov model and hierarchical clustering." *Knowledge-Based Systems* **148**: 146–166.
[11] Yang F, Gao M, Yu J, Song Y, Wang X. (2018) "Detection of shilling attack based on bayesian model and user embedding", in *Proc. of Int. Conf. on Tools with Artificial Intelligence*: 639–646.
[12] Gunes I, Kaleli C, Bilge A, Polat H. (2014) "Shilling attacks against recommender systems: A comprehensive survey." *Artificial Intelligence Review* **42** (4).
[13] Zhang, Zhuo, Kulkarni SR. (2013) "Graph-based detection of shilling attacks in recommender systems", in *Proc. of Int. Workshop on Machine Learning for Signal Processing*: 1–6.
[14] Panagiotakis C, Papadakis H, Fragopoulou P. (2020) "Unsupervised and supervised methods for the detection of hurriedly created profiles in recommender systems." *International Journal of Machine Learning and Cybernetics* **11**.
[15] Bilge A, Ozdemir Z, Polat H. (2014) "A novel shilling attack detection method." *Procedia Computer Science* **31**:165–174.
[16] Zhou W, Wen J, Xiong Q, Gao M, Zeng J. (2016) "SVM-TIA a shilling attack detection method based on SVM and target item analysis in recommender systems." *Neurocomputing* **210**: 197–205.
[17] Cai H, Zhang F. (2019) "Detecting shilling attacks in recommender systems based on analysis of user rating behavior." *Knowledge-Based Systems* **177**: 22–43.
[18] Zhang F, Deng ZJ, He ZM, Lin XC, Sun LL. (2018) "Detection of shilling attack in collaborative filtering recommender system by pca and data complexity", in *Proc. of Int. Conf. on Machine Learning and Cybernetics*: 673–678.
[19] Breiman L. (1996) "Bagging predictors." *Machine learning* **24**:123–140.
[20] Schwenk H, Bengio Y. (2000) "Boosting neural networks." *Neural Computation* **12** (8):1869–1887.
[21] Rincy TN, Gupta R. (2020) "Ensemble learning techniques and its efficiency in machine learning: A survey", in *Proc. of Int. Conf. on Data, Engineering and Applications*: 1–6.
[22] Dietterich TG. (2000) "Ensemble methods in machine learning", in *Proc. of Int. Workshop on Multiple Classifier Systems*: 1–15.
[23] Zhang F, Chen H. (2016) "An ensemble method for detecting shilling attacks based on ordered item sequences." *Security and Communication Networks* **9** (7): 680–696.
[24] Panagiotakis C, Papadakis H, Fragopoulou P. (2018) "Detection of hurriedly created abnormal profiles in recommender systems", in *Proc. of Int. Conf. on Intelligent Systems*: 499–506.
[25] Guo G, Wang H, Bell D, Bi Y, Greer K. (2003) "KNN model-based approach in classification", in *Proc. of Int. Conf. On The Move to Meaningful Internet Systems*: 986–996.
[26] Rish I, et al. (2001) "An empirical study of the Naive Bayes classifier", *Proc. of Workshop on Empirical Methods in Artificial Intelligence*: 41–46.
[27] Zheng S, Jiang T, Baras JS. (2011) "A robust collaborative filtering algorithm using ordered logistic regression", in *Proc. of Int. Conf. on Communications*: 1–6.
[28] Karthikeyan P, Selvi ST, Neeraja G, Deepika R, Vincent A, Abinaya V. (2017) "Prevention of shilling attack in recommender systems using discrete wavelet transform and support vector machine", in *Proc. of Int. Conf. on Advanced Computing*: 99-104.
[29] Balakrishnama S, Ganapathiraju A. (1998) "Linear discriminant analysis-a brief tutorial." *Institute for Signal and Information Processing* **18**: 1–8.
[30] Schapire RE. (2003) "The boosting approach to machine learning: An overview." *Nonlinear estimation and classification*: 149–171.