



Hybrid Recommendation System

Raghul Krishna Rajasekaran¹, Alexander I. Iliev^{1,2}(✉)

¹ SRH University Berlin, Charlottenburg, Germany,
3105481@stud.srh-campus-berlin.de

² Institute of Mathematics and Informatics
Bulgarian Academy of Sciences, Sofia, Bulgaria
8 Acad. Georgi Bonchev Str., Bulgaria, 1113, Sofia
ailiev@berkeley.edu

Abstract. In this paper we aim to explore and implement hybrid recommendation system using collaborative and content based recommendation system that can be applied to repairs marketplace. We developed multiple variants of proposed hybrid recommendation system architecture and compared their performance with a metric called hit rate. Furthermore, we discussed about data preparation and mathematical explanation of the machine learning models we used for hybrid recommendation system. To achieve this we used different python frameworks used for natural language processing and recommendation systems.

Keywords Hybrid Recommendation System • Word2Vec • TF-IDF • SVD • KNN.

Introduction

During the recent times we are experiencing a phenomenon called information overload. This exponential growth of digital information in the recent advancements and digitalization has created the phenomenon information overload. Recommendation system has been around from late 90s. It is software tools and methods that provides suggestions to people for items they are looking for. In early days these recommendation systems are heuristic based on certain conditions or based on user input. Today we have extremely advanced recommendation system with the help of machine learning algorithms which analyses users behaviour and suggest the items that they are looking for. There many different types of recommendation system most used ones are content based recommendation system, collaborative recommendation system and hybrid recommendation system. In this paper we proposed a hybrid recommendation for repairs marketplace, developed different variants of hybrid recommendation system and compared their performance using a metric called hit rate. For development of hybrid recommendation system we used python libraries NLTK and SCIKIT SURPRISE. NLTK called as natural language processing toolkit is most used library for natural language processing tasks. In Scikit Surprise, SURPRISE stands for Simple python recommendation system Engine. It is used for building and analysing recommender systems.

Literature Review

The growing relevance of the Internet as a medium for electronic and business transactions has fuelled the advancement of recommender system technology. The simplicity with which

the Web allows consumers to submit feedback about their likes and dislikes is a key catalyst in this regard.

In the paper about the Content-based recommendation system, the authors (Zisopoulos, Charilaos & Karagiannidis, Savvas) [1] gave a brief overview of the main phenomenon and they have discussed various types of content-based recommendations systems. In the paper [2] Schafer, Ben & J, Ben (2007) gives a detailed overview of Collaborative Filtering Recommender Systems. Lianhuan Li, Zheng Zhang (2021) [3] in "Hybrid Algorithm Based on Content and Collaborative Filtering in Recommendation System Optimization and Simulation", Scientific Programming. The authors Lianhuan Li, Zheng and Zhang, Shaoda Zhang show how can we leverage both content-based filtering and collaborative filtering to create a most effective hybrid recommendation system. Vahidi Farashah, M., Etebarian, A., Azmi, R. *et al.* A hybrid recommender system based on link prediction for movie baskets analysis. *J Big Data* 8, 32 (2021). The authors (Vahidi Farashah, M., Etebarian, A., Azmi, R) [4] proposed a link prediction-based method to overcome problems in other methods like lack of accuracy and high error of recommendation the proposed method includes clustering users using DB scan and classification of the new user using Deep neural network, collaborative filtering to extract most similar users and then high rated movies are suggested for them, running an improved friend link algorithm to connect similar users in the link. The result shows the proposed model has an average of 8% less Mean Squared error than the traditional Machine learning algorithms. He, Ruining & Kang, Wang-Cheng & McAuley, Julian. (2017). Discuss Translation-based Recommendations which gave a brief context about the concept of this phenomenon. The authors Ruining & Kang, Wang-Cheng & McAuley, Julian [5] proposed a model called transfer using sequential models for neighbourhood-based collaborative filtering method. Personalized recommendation system K- nearest neighbour algorithm optimization, In this paper [5], the author discussed the KNN based user-based collaborative filtering approach also discussed the merits and demerits of the algorithm. Block-based Singular Value Decomposition approach to matrix factorization for recommender systems, In this paper [6], the authors proposed an approach called block matrix factorization with singular value decomposition which increases the scalability and performance of the singular value decomposition method. In the paper [7] An Improved Hybrid Recommender System by Combining Predictions, the authors (Chikhaoui, Belkacem & Chiazaro, Mauricio & Wang, Shengrui) proposed a hybrid recommender system, with a combination of content-based, collaborative, and demographic systems also showed how the demographic system helps solve the cold start problem. In the paper [8], Building a switching Hybrid Recommender System using machine learning classifiers and collaborative filtering author proposed a hybrid recommender system that combines machine learning classification with collaborative filtering which shows scalability and increased performance.

Problem Statement

In this paper we used repairs and management marketplace data where the dataset consist of various details about the repairs or maintenance. The aim of the paper is we try to recommend the contractors for every new repairs comes in. Unlike typical marketplaces like Netflix we cannot recommend same items to millions of user in repairs marketplace once the repair has been completed it cannot be recommended to other users hence it brings in the cold start problem every time a new item comes in. from this we derive the problem statement how to build a recommendation system for short-lived items.

Methodology

This research aims to build a hybrid recommendation system for short lived items. To achieve the aim of this paper, the first process is to research the background of various topics, so the literature study will be conducted. We used the data of a repairs marketplace which has the various details for the repairs or maintenance. The data will be later cleaned and pre-processed for building machine learning models appropriately. After data pre-processing, we used the pre-processed data to build a hybrid recommendations system which is a combination of a content-based recommendation system and a collaborative recommendation system. We developed different hybrid recommendation systems and compared their performance using a metric called hit rate.

Dataset

There are two datasets used for this paper, the First dataset contains a unique identifier of a Job along with features of the job and the second dataset contains the user interaction with these jobs which contains job id and user id along with the interaction status of the work order. In the first dataset, we used 9 features. They are work order service type, work order category, work order complexity, emergency tag type, organization name, latitude, longitude, work order job type, and description of the work order.

The dataset used in this paper has the following features:

- categories: Categories show the types of jobs example Handyman, Roofer, etc.
- service: Service refers to the subcategories of services like general maintenance, door easing, etc.
- Work order complexity: Work order complexity shows whether the Job is a standard or complex job.
- Emergency tag type: The emergency tag type shows the tags that the client added to the Job.
- Organization group: Organization group shows the organization name of the landlord or property manager who posted the job.
- Work order job type Work order job shows whether the job is an emergency job that needs immediate attention or other types of jobs.
- Work order description: The work order message has the description of the job that's written by landlords or property managers.
- Work order location: work order location has the latitude and longitude of the job.
- Work order id: Work order id has the job's unique identifier
- User id: User id shows the contactor's unique identifier who worked on the job.

Recommendation System

TFIDF

TFIDF is short for term frequency-inverse document frequency which is a weighting algorithm used in text mining and topic modeling in NLP which shows how important a word is for a document. The importance of word increases when the number of times a word appears in the document increases also it decreases when the frequency of the word in the whole corpus increases. TF IDF includes two parts named Term Frequency and Inverse Document Frequency. TF is representing the frequency of a word that appears in a document whereas IDF is responsible for the frequency of a word that appears in the whole corpus.

Hence words that are common like articles will have lower weightage. TF-IDF of a word is a product of the term frequency of a word and inverse document frequency of a word:

$$TF - IDF = TF \times IDF$$

Equation 1: TF-IDF

Term Frequency

Term frequency shows how important a word is in a specified document or how frequently a word appears in a document. As the document length is not constant among the dataset there is the possibility that a word would appear more in long documents. hence for normalization, the term frequency is divided by document length. Term Frequency of a word can be expressed as $TF(w_{ij})$, $N(w_i, d_j)$ represents the number of times word W_i occurs in document d_j and $N(d_j)$ represents the total number of words in the document:

$$TF(w_{ij}) = \frac{N(w_i, d_j)}{N(d_j)}$$

Equation 2: Term Frequency

Inverse Document Frequency:

Inverse document frequency shows how important a word is in the whole corpus. $IDF(w_{ij})$ represents the inverse document frequency, $N(D)$ represents the total amount of document, $N(w_{ij}, D)$ represents how many times a word occurs in the whole corpus. The higher the number of words W_{ij} appears in all documents will have less importance. Inverse document frequency is used to boost the importance of words that are unique to a document with the hope that it has more information:

$$IDF(w_{ij}) = \log \frac{N(D)}{N(w_{ij}, D)}$$

Equation 3: Inverse Document Frequency

Word2vec

Word2vec is one of the methods to create word embeddings using two later neural networks. The input of word2vec is a large corpus of text and the output will be the words vector representation. The represented vectors are selected by the cosine similarity function which shows the semantic relationship between words. If the cosine similarity is closer to 1 between two words means they are very similar to each other and if they are close to 0 they are not similar words. The words with more cosine similarity will be close to each other in the vector space.

Word2vec has two variants one using CBOW (continuous bag of words) and the other using the Skip gram model. CBOW and Skip gram learn the vectors which will be the representation of the words.

Continuous Bag of Words (cbow)

CBOW tried to predict the missing word given a window of words or word sequence. CBOW given the surrounding words predicts the probability of a missing word. The words undergo hot encoding as a first step. One hot encoding is a vector representation of text or categorical data where all the features of the vector are 0 except the feature it is representing which will be 1. After one hot encoding CBOW takes words that are in the given window. For example, given a window of 5 CBOW will try to predict the middle word and the surrounding words are given as input. This window moves through the entire corpus and the process is repeated. Once the process is completed the vector representation of words is given by word2vec. CBOW tries to maximize the function:

$$\frac{1}{i} \sum_{i=1}^i \log p(w_i | \sum_{-k \leq j \leq k, j \neq 0} w_{i+j})$$

Equation 4: CBOW

Where $w_1 w_2 \dots w_i$ represents the sequence of words, k is a window of words at the right and left of the target word.

Continuous Skip Gram Model

Skip gram model works in the same way as CBOW but does it in a reverse way. Skip gram given the target word tries to predict the surrounding words. It trains a neural network to learn the weights of the hidden layer. These weights are indeed the word vectors that we are trying to predict. Skip gram tries to maximize the function:

$$\frac{1}{i} \sum_{i=1}^i \sum_{-k \leq j \leq k, j \neq 0} \log p(w_{i+j} | w_t)$$

Equation 5: Skip Gram

Training

Both the variants are trained using backpropagation and stochastic gradient descent to tune the weights in the neural network and reduce the loss function. By training the models using these methods we learn the weights which become the vector representation of the words.

Problems With Training Word2vec

Let's assume we have 400 components and a vocabulary of 5000 words. We have two-layered neural networks hidden layer and an output layer. Both layers will have a matrix of $300 \times 5000 = 1.5$ million weights each. Training a matrix of huge size will be very slow also to get a better accurate model there needs to be a huge amount of data to achieve that.

While training word2vec we use backpropagation to update the values in the layers with every backpropagation there will be only sparse changes. Calculation of probability using softmax will be very expensive as it involves the summation of all the words in the corpus. To overcome these issues, we use a negative sampling method to overcome these shortcomings.

KNN

K-nearest neighbors is a supervised machine learning algorithm used for regression and classification problems. it is a simple way to classify data. The main intuition of KNN is that similar things exist at close distances. it is a non-parametric machine learning technique (which means it does not make any assumptions or parameters of frequency distribution). KNN calculated similarity based on different distance functions. Euclidean distance is the most common distance function used. KNN is also called a lazy algorithm as it does not involve any training steps. All the data points are used during prediction hence making the prediction process expensive. Since KNN is nonparametric the model structure is determined from the training data. As a typical supervised machine learning algorithm, KNN expects labelled input data to a learning model's function. KNN first computes a distance value between the predicted item to all the items it's been trained on. Finally, it picks the k closest data and gives the values or labels of the closest neighbors. The probability of assigning input a can be represented as:

$$P(y = j | A = a) = \frac{1}{K} \sum_{i \in X} I(y^{(i)} = j)$$

Equation 6: KNN

Distance Functions

Minkowski Distance

Minkowski distance is a distance metric intended for real-values vector spaces. Minkowski distance can be calculated only in a normed vector space where the vectors have a positive length. The formula for Minkowski distance can be represented as:

$$\left(\sum_{i=1}^n |a_i - b_i|^p \right)^{\frac{1}{p}}$$

Equation 7: Minkowski Distance

Euclidean Distance

Euclidean distance is the most used distance function where it is a straight-line distance between two points. The formula for Euclidean distance can be represented as:

$$\left(\sum_{i=1}^n |a_i - b_i|^2 \right)^{\frac{1}{2}}$$

Equation 8: Euclidean Distance

Manhattan Distance

Manhattan distance is also known as taxicab or city block distance. it is calculated by the sum of the absolute differences of the cartesian coordinates of the given two points:

$$\sum_{i=1}^n |a_i - b_i|$$

Equation 9: Manhattan Distance

Cosine Distance

Cosine distance is used to calculate the similarity between two vectors using cosine angle. It is calculated by measuring the cosine of the angle between two. The formula of cosine distance can be represented as:

$$\cos \theta = \frac{\vec{a} \cdot \vec{b}}{\|\vec{a}\| \cdot \|\vec{b}\|}$$

Equation 10: Cosine Distance

Singular Value Decomposition:

Singular value decomposition is a way to break down a matrix into consistent parts. It is a variant of eigen decomposition where the eigen decomposition is applied to the square matrix and singular value decomposition can be applied to non-square matrices. SVD breaks down the matrices into three different matrices. The three matrices that have been decomposed by SVD are two unitary matrices and a diagonal matrix. SVD can be represented as:

$$X = U\Sigma V^T$$

Equation 11: SVD

Where U is a real or complex unitary matrix, Σ is the rectangular diagonal matrix and V is a real or complex unitary matrix.

Recommendation System Architecture

In this paper, we combined the power of both content-based and collaborative filtering to produce recommendations. In figure 1, you can see the flow diagram of how the recommendation system work. The input of the recommendation system is preprocessed features combined into a single document where the content-based recommendation system first produces n similar jobs, Next output produced by content-based recommendation is used by collaborative filtering to produce recommended contractors for every new job that comes in the platform. Content-based recommendation system created by using jobs descriptive features and collaborative filtering recommendation system is created by using contractor's engagement data in the platform. We compared different models in this paper, for the content-based recommendation system we used TF-IDF and Word2vec, and for the collaborative filtering recommendation system we used K nearest neighbors and Singular value decomposition algorithms.

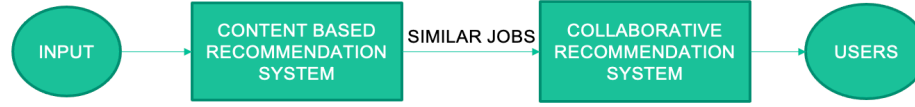


Fig. 1. Recommendation System Architecture.

Content-based Recommendation System

To Build a content-based recommender we use the descriptive features of the item. The descriptive feature of the item is combined into a single document. We used the models TFIDF and word2vec to convert text to vectors. Once all the documents are converted next step, we use cosine similarity to get the most similar documents with the new documents. In our case, we combine all the features of the job into a single document and get the most similar jobs using cosine similarity in a content-based system. With this approach, we can overcome the cold start problem for short-lived items.

Collaborative Filtering Recommendation System

To Build a collaborative filtering recommender we used engagement data of contractors in the marketplace. We later converted their engagement of contractors with jobs into rating data as if the contractor has engaged and got selected for a job, he was given a rating of 2, if the contractor has engaged with the job and was not selected, he was given a rating of 1.

Evaluation

Hit rate

The hit rate is a metric to evaluate recommendation systems. The hit rate is calculated by several hits produced by the recommendation system and the total size of test data. Hits in hit rate are the comparison between the test data and prediction data for top n recommendations it is considered as a hit whenever there predicted data is in the test data.

Results

We developed 4 different hybrid recommendation systems and compared them using hit rate. The 4 different combinations are created by content-based recommenders and collaborative recommenders. Content-based recommenders are built using TFIDF and Word2Vec. Collaborative filtering recommenders are built using KNN and SVD. Figure 2 and table 1 show the hit rates of all the hybrid recommendation systems. Figure 3 and table 2 show the hit rate changes over different epochs of Word2Vec.

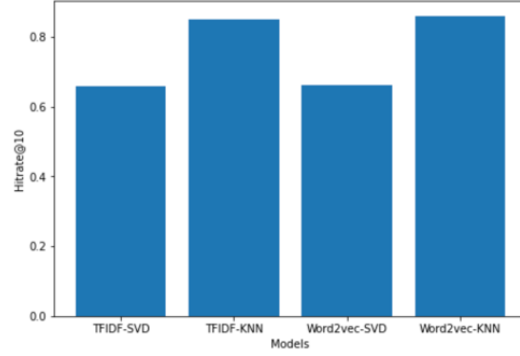


Fig. 2. Hit rates of all hybrid recommendation systems.

Table 1. Hit rates of all hybrid recommendation systems.

Models	TFIDF-KNN	TFIDF-SVD	Word2Vec-KNN	Word2vec-SVD
Hit rate@10	0.85	0.64	0.86	0.66

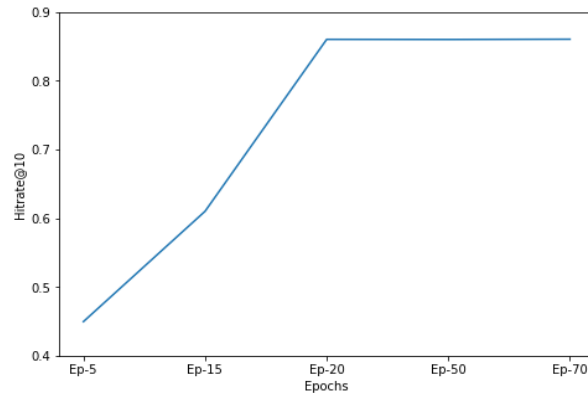


Fig. 3. Hit rate changes over different epochs

Table 2: Hitrate@10 over Epochs for Word2vec-KNN

Word2Vec Epochs	Hitrate@10
5	0.449544
15	0.610092
20	0.860280
50	0.860108
70	0.860501

Conclusion

Recommendation systems are becoming more and more popular due to information overload. There are many variants of recommendation systems available for various use cases. We attempt to find a new approach for a hybrid recommendation system for short-lived items. For the problem we mentioned at the beginning, we proposed a new approach of a hybrid recommendation system for short-lived items that leverages the power of both contents-based and collaborative recommendation methods. This Paper introduces a hybrid recommendation system for a repairs marketplace. Repairs marketplace is different from popular marketplaces, which suffers a cold start problem for a every new item that needs recommendations hence we proposed a new approach of hybrid recommendation system and developed four different types of hybrid recommendation system using machine learning algorithms.

At the end of the paper, we used a metric called hit rate to evaluate our recommendation system. From the comparison, we can see that the Hybrid recommendation system with word2vec and K nearest neighbours performed better than the other approaches.

Future Work

The detailed conclusion is precisely discussed in the above subsection. As with every research work, this paper also has areas that could be scaled for future research scopes. Apart from that, with our research work, the aim is accomplished. Below the possible future research works are listed.

- Use deep learning algorithms: for further improvements in the model, we can introduce deep learning algorithms in both content-based and collaborative filtering
- Collect and used click data: in this work we used only user's engagement data with the items additionally we can introduce user click which will be more effective than the engagement data
- Introduce user dislike data: in the future, we can collect more user data and add a user dislike item list. We will input the dislike item list into the recommender system and generate negative ratings. With this additional data, recommendations will be improved.

References

1. Zisopoulos, Charilaos & Karagiannidis, Savvas & Demirtsoglou, Georgios & Antaris, Stefanos. (2008). Content-Based Recommendation Systems.
2. Schafer, Ben & J, Ben & Frankowski, Dan & Dan, & Herlocker, & Jon, & Shilad, & Sen, Shilad. (2007). Collaborative Filtering Recommender Systems.
3. Lianhuan Li, Zheng Zhang, Shaoda Zhang (2020) "Hybrid Algorithm Based on Content and Collaborative Filtering in Recommendation System Optimization and Simulation", Scientific Programming, vol. 2021, Article ID 7427409, 11 pages, 2021. <https://doi.org/10.1155/2021/7427409>

4. Vahidi Farashah, M., Etebarian, A., Azmi, R. *et al.* (2021). A hybrid recommender system based on link prediction for movie baskets analysis. *J Big Data* 8, 32 (2021). <https://doi.org/10.1186/s40537-021-00422-0>
5. He, Ruining & Kang, Wang-Cheng & McAuley, Julian. (2017). Translation-based Recommendation. 10.1145/3109859.3109882.
6. Bhavana, P., Kumar, V., & Padmanabhan, V. (2019). Block-based Singular Value Decomposition approach to matrix factorization for recommender systems. ArXiv, abs/1907.07410.
7. Chikhaoui, Belkacem & Chiazzaro, Mauricio & Wang, Shengrui. (2011). An Improved Hybrid Recommender System by Combining Predictions. 644-649. 10.1109/WAINA.2011.12.
8. Ghazanfar, Mustansar ali & Prugel-Bennett, A.. (2010). Building Switching Hybrid Recommender System Using Machine Learning Classifiers and Collaborative Filtering. *IAENG International Journal of Computer Science*. 37.