Ministry of Science and Higher Education of the Republic of Kazakhstan
SDU University

UDC 004.021

Serek Azamat Galymzhanuly

Development of supervisorship system with tracking progress and the use of artificial intelligence

8D06102 - Computer Science

Dissertation submitted in fulfillment of the requirements
for the degree Doctor of Philosophy (Ph.D.)

Scientific Advisor
Assist. Professor, Ph.D.
Meirambek Zhaparov

Foreign scientific advisor
Assoc. Professor, Ph.D.
Seong-Moo (Sam) Yoo

Republic of Kazakhstan
Kaskelen, 2024

Contents

Normative references

This thesis uses references to the following standards:

- “Instructions for the preparation of a dissertation and author’s abstract” Ministry of education and science of the Republic of Kazakhstan, 377-3 Zh.
- GOST 7.32-2001. Report on research work. Structure and design rules.
- GOST 7.1-2003. Bibliographic record. Bibliographic description. General requirements and compilation rules.
- GOST 7.32-2017. System of standards of information, librarianship, and publishing. Research report. Structure and design rule

List of abbreviations

- CNN - convolutional neural network
- AI - artificial intelligence
- ML - machine learning
- SIFT - Scale-invariant feature transform
- SURF - Speeded Up Robust Features
- DoG - difference-of-Gaussian
- POS - part-of-speech
- NLP - natural language processing
- NER - Named Entity Recognition
- SGD - stochastic gradient descent
- AR - autoregressive
- ARIMA - Autoregressive Integrated Moving Average
- SARIMA - Seasonal Autoregressive Integrated Moving Average
- VAR - Vector Autoregression
- BSTS - Bayesian Structural Time Series
- GARCH - Generalized Autoregressive Conditional Heteroscedasticity
- MLE - maximum likelihood estimation
- MSE - mean squared error
- RMSE - root mean squared error

Abstract

The Supervisorship service has been developed with the feature to match students to supervisors based on psychological perceptions through multidimensional analysis of matching algorithms and the feature to track students' progress.
This study explores the utilization of four distinct algorithms for the purpose of student-supervisor matching. A comprehensive evaluation of these algorithms is conducted, encompassing a variety of metrics including preference satisfaction, workload balance, time and space complexities, minimum and maximum workload, and compatibility scores which this work introduced.
Algorithms such as collaborative filtering, K-Means, One-to-Many Gale-Shapley, and genetic algorithms were used. Among the used algorithms, it was decided to denote one-to-many Gale Shapley as the most optimal one, because according to experimental results, it achieved 0.74 preference satisfaction, the best balance of workload at 0.5, the lowest maximum workload of 6, the highest compatibility score of 0.46, and the lowest time complexity.
Matching was done based on a dataset of perceptions of students and supervisors regarding their workflow. The matching algorithm was integrated into the service.
Based on data from dataset 4, where students were paired with supervisors manually, yielding an average preference satisfaction score of 0.48. In comparison, in experiment 4, where the one-to-many Gale-Shapley algorithm was employed for matching, the average preference satisfaction score significantly increased to 0.74. This signifies a 35% improvement in satisfaction rates.

Абстракт

Супервайзерлік қызмет студенттерді жетекшілерге психологиялық түсініктерге негізделген алгоритмдердің көмегімен сәйкес келтіру мүмкіндігі бар сервиске айналды. Сервис студенттердің үлгерімін бақылау функционалымен қосылды.

Бұл зерттеу тәжірибелерді жеке алгоритмдерді қолдану жолымен студенттерді жетекшілерге сәйкес келтіру мақсатында негізгіп талдау етеді. Осы алгоритмдерді қолдану жайлы жүйелік бағалау қамтылып, олардың арасында таңдаушының құрылымын, жүктеме балансын, уақыт және орын кеңістігін, мүмкіндік жоғары және төмен жүктемесін, сондай-ақ үйлесімдік көрсеткіштерді қамтылады.

Коллаборативтік фильтрлеу, K-орташасы, Бірден көпке Гейла-Шепли, және генетикалық алгоритмдер сияқты алгоритмдерді қолданылды. Қолданылатын алгоритмдердің ішінде Гейла-Шеплидің бірден көпке алгоритмін ең оптимальды деп таңдау қарар берілді, себебі эксперименттік нәтижелерге сәйкес ол 0,74 деңгейдегі құрылыммен жүктеменің ең жақсысын, 0,5 деңгейдегі ең жақсы жүктеме балансын, 6 деңгейдегі ең төменгі максималды жүктемені, 0,46 деңгейдегі ең жоғарғы үйлесімдік көрсеткішін, және ең төменгі уақыт қиыншылықты. Сапарымен, оқушылар мен супервизорлардың орындауы туралы мәліметтер жиынтығы бойынша салыстыру үшін жасалды. Салыстыру алгоритмі қызметке біріктірілді.

Студенттер жетекшілермен қолмен жұптастырылған 4-деректер жинағының деректеріне негізделіп, 0,48 орташа артықшылықты қанағаттандыру ұпайын берді. Салыстыру үшін 4-тәжірибеде Гейл-Шапли алгоритмі сәйкестендіру үшін қолданылған кезде, артықшылықты қанағаттандырудың орташа баллы 0,74-ке дейін айтарлықтай өсті. Бұл қанағаттану деңгейінің 35% жақсарғанын білдіреді.

Абстракт

Сервис супервайзерства был разработан с возможностью подбора студентов к научным руководителям на основе психологических восприятий, основанных на многомерном анализе алгоритмов сопоставлений, а также с функцией отслеживания прогресса студентов.

В данном исследовании исследуется использование четырех различных алгоритмов, которые сопоставляют студентов и руководителей. Проводится комплексная оценка этих алгоритмов, охватывающая различные метрики, включая удовлетворение предпочтениям, баланс нагрузки, временную и пространственную сложность, минимальную и максимальную нагрузку, а также показатель совместимости.

Были использованы алгоритмы, такие как коллаборативная фильтрация, K-средних, один-ко-многим Гейла-Шепли и генетический алгоритм. Среди используемых алгоритмов было решено выделить алгоритм один-ко-многим Гейла-Шепли как наиболее оптимальный, потому что, согласно результатам экспериментов, оно обеспечило удовлетворение предпочтениям на уровне 0,74, наилучший баланс нагрузки на уровне 0,5, минимальную максимальную нагрузку на уровне 6, наивысший показатель совместимости на уровне 0,46 и наименьшую временную сложность.

Сопоставление было выполнено на основе набора данных о восприятиях студентов и руководителей относительно их рабочего процесса. Алгоритм сопоставления был интегрирован в сервис.

На основе данных из набора данных 4, где студенты были объединены в пары с руководителями вручную, что дало средний балл удовлетворенности предпочтениями 0.48. Для сравнения, в эксперименте 4, где для сопоставления использовался алгоритм Гейла-Шепли «один ко многим», средний балл удовлетворенности предпочтений значительно увеличился до 0.74. Это означает улучшение уровня удовлетворенности на 35%.

1.Introduction

Currently, a lot of universities manually match students to supervisors and do not have technological infrastructure to automate this process and improve their workflow by allowing supervisors to track progresses of their students automatically. In this work, there was built a supervisorship system that includes a feature to track students' progress effectively with the function of automatic matching based on student- supervisor perceptions of their workflows. Matching algorithm was selected based on multidimensional analysis of a set of matching algorithms which can be used to match students to supervisors such as collaborative filtering, K-Means, genetic algorithm, Gale- Shapley.

1.1 Problem statement

Address the issue of lack of consideration of various metrics in implementation of matching students to supervisors and the lack of supervisorship systems that include matching and tracking features.

1.2 Relevance

Currently, Kazakhstan does not have services that can facilitate tracking progress of implementation of students' thesis with feature to match them properly based on psychological perceptions of student-supervisor relationship. Increase of number of students and modern conditions require optimization and automatization of the above-mentioned processes to facilitate efficient student-supervisor workflows to enhance research output of universities. There exists systems to match students to supervisors, but no multidimensional comparative study between them has been done and none of the work considered psychological perceptions during the process. Also, no supervisorship service exists that integrate efficient tracking with feature to match students to supervisors

1.3 Aim and objectives

Aim:

- Create supervisorship system that has a feature to track students' progress and effectively matches students to supervisors based on to be conducted multidimensional comparison of matching algorithms.

Objectives:

1. Collect data from students and supervisors regarding their psychological perceptions
2. Define metrics for comparing matching algorithms
3. Match students to supervisors by the use of different matching algorithms
4. Implement web-based system that incorporates matching and tracking features
5. Compare matched results according to defined metrics such as balance of workload, preference satisfaction, etc.

1.4 Research questions

1. Which algorithms to use to match students to supervisors considering metrics of success?
2. Which algorithm demonstrates highest preference satisfaction score for matching students to supervisors?
3. What supervisorship service should entail?
4. What algorithm has the best balance of workload for matching students to supervisors?
5. How recommendation system algorithms can be used in matching of students to supervisors?
6. How to apply K-Means efficiently in the matching of students to supervisors?

1.5 Scientific novelty

The novelty of the dissertation is outlined according to the following results:

1. There was constructed a supervisorship system that can efficiently track the progress of students with an automated method to match students to supervisors by taking into account the psychological perceptions of students and supervisors regarding their workflow.
2. New metrics were introduced to compare algorithms of matching students to supervisors such as
    a. Workload Balance. It is calculated as the ratio of the minimum workload to the maximum workload among supervisors which is useful to prevent overload and underload among supervisors.
    b. Preference Satisfaction The preference satisfaction for each matched student-supervisor pair is calculated as the absolute difference between the student's preference and supervisor’s preference, where preference is summation of 8 criteria. It is useful to consider this metric to leave students and supervisors satisfied with their match.

c. Maximum workload - maximum number of students allotted to one supervisor, which is useful to see the most overloaded supervisor for managing workload.
d. Minimum workload - minimum number of students allotted to one supervisor, which is useful to see the most underloaded supervisor for managing workload
e. Compatibility score is a count of pairs where students prefer other supervisors, normalizing it against total pairs to quantify compatibility. It is useful to see how many success pairs are there.

3. Multidimensional analysis between several algorithms such as Gale-Shapley, genetic algorithm, collaborative filtering, and K-Means was conducted
4. There was found matching algorithm to match students to supevisors that has the most optimum values in terms of introduced metrics.

1.6 Publications

4 journal articles:

1. A. Serek and M.Zhaparov, “Optimizing preference satisfaction with genetic algorithm in matching students to supervisors”, Applied Mathematics and Information Sciences (AMIS), Volume 18, No. 01, PP:133-138 (**Percentile: 52**), ISSN 1935-0090 (print), ISSN 2325-0399 (online). (2024).
2. A. Serek and M.Zhaparov, “Algorithm Comparison for Student-Supervisor Matching in Supervisorship System Development: K-Means vs. One-to-Many Gale-Shapley”, Information Sciences Letters (ISL), Vol. 12, No. 12 (2023), PP:2417-2425, ISSN 2090-9551 (Print), ISSN 2090-956X (Online). (**Percentile: 38**). (2023).
3. A.Talasbek, A.Serek, M.Zhaparov, S.Yoo, Y.Kim and G.Jeong, “Personality classification experiment by applying k-means clustering”, International Journal of Emerging Technologies in Learning (iJET), Vol. 15 No.16, PP 162-177, ISSN:1868-8799, E-ISSN:1863-0383 (**Scopus: Q2, Percentile: 66**). (2020).
4. A.Serek, A.Talasbek, M.Zhaparov, S.Yoo, Y. K. Kim and M.W.Jin, “Best Practices in Running IT Hackathons Based on Paragon University Dataset”, International Journal of Emerging Technologies in Learning (iJET), Vol.15 No.19, PP 231-238, ISSN:1868-8799, E-ISSN:1863-0383 (**Scopus: Q2, Percentile: 66**). (2020).

9 IEEE conference proceedings indexed by Scopus:

1. A. Serek, K. Orynbekova, A. Talasbek, D. Kariboz, G. Saimassay and A. Bogdanchikov, "Recommendation System for Human Resource

Management by the Use of Apache Spark Cluster," 2023 17th International Conference on Electronics Computer and Computation (ICECCO), Kaskelen, Kazakhstan, 2023, pp. 1-4, doi: 10.1109/ICECCO58239.2023.10147129.
2. A. Serek, G. Saimassay, M. Zhaparov, C. Nguyen Giang, V. Truong Hoang and Z. Zhalgassova, "Analysis of Self-esteem on Students' Performance in Online Programming Competition," 2023 17th International Conference on Electronics Computer and Computation (ICECCO), Kaskelen, Kazakhstan, 2023, pp. 1-4, doi: 10.1109/ICECCO58239.2023.10147137.
3. A. Serek, A. Akhmetov, N. Ismagulov, B. Rysbek, B. Rysbek and S. Alim, "Application of k-means in the perception of supervisors from students' side," 2021 16th International Conference on Electronics Computer and Computation (ICECCO), 2021, pp. 1-3, doi: 10.1109/ICECCO53203.2021.9663859.
4. A. Serek, A. Issabek, A. Akhmetov and A. Sattarbek, "Part-of-speech tagging of Kazakh text via LSTM network with a bidirectional modifier," 2021 16th International Conference on Electronics Computer and Computation (ICECCO), 2021, pp. 1-4, doi: 10.1109/ICECCO53203.2021.9663794.
5. A. Serek, A. Bazarkulova, A. Chazhabayev and A. Akhmetov, "Analysis of supervisors and students in the context of diploma defense," 2021 16th International Conference on Electronics Computer and Computation (ICECCO), 2021, pp. 1-4, doi: 10.1109/ICECCO53203.2021.9663776.
6. A. Talasbek, A. Serek, M. Zhaparov, S. -M. Yoo, Y. -K. Kim and G. -H. Jeong, "Personality Classification by Applying k-Means Clustering," 2020 International Conference on Artificial Intelligence in Information and Communication (ICAIIC), 2020, pp. 421-426, doi: 10.1109/ICAIIC48513.2020.9065244.
7. A. Serek, M. Zhaparov and S. -M. Yoo, "Analysis of Data to Improve System of an Educational Organization," 2018 14th International Conference on Electronics Computer and Computation (ICECCO), Kaskelen, Kazakhstan, 2018, pp. 206-212, doi: 10.1109/ICECCO.2018.8634709.
8. A. Serek, A. Issabek and A. Bogdanchikov, "Distributed sentiment analysis of an agglutinative language via Spark by applying machine learning methods," 2019 15th International Conference on Electronics, Computer and Computation (ICECCO), Abuja, Nigeria, 2019, pp. 1-4, doi: 10.1109/ICECCO48375.2019.9043264.
9. Y. Amirgaliyev, S. Shamiluulu and A. Serek, "Analysis of Chronic Kidney Disease Dataset by Applying Machine Learning Methods," 2018 IEEE 12th International Conference on Application of Information and Communication

Technologies (AICT), Almaty, Kazakhstan, 2018, pp. 1-4, doi: 10.1109/ICAICT.2018.8747140.

1.7 Expected results

1. It is expected to collect essential data regarding the student-supervisor perceptions of their workflows.
2. New metrics that are suitable to compare matching of students to supervisors are expected to be defined.
3. It is expected to apply proper algorithms to match students to supervisors.
4. Sophisticated and useful web service with feature to track progress of students and feature to match students to supervisors is expected to be implemented.
5. All used algorithms for matching students to supervisors are expected to be compared according to various criteria.

1.8 Thesis outline

The thesis is structured in the following way.
Chapter 1 covers introduction part where there are outlined general characteristics of research, research problem, relevance of the research, aim, objectives, scientific novelty, publications and expected results.
Chapter 2 covers a literature review that gives a thorough analysis of the pertinent literature and defines the knowledge gap that this work addresses. Specifically, there is described in details various research papers related to matching students to supervisors. Also, association analysis and time series prediction related literature is described. It ends by comparison with other research work where there is an outlined difference of the current work with the related work. Based on literature review, there is described knowledge gap that this work addresses.
Chapter 3 provides a detailed overview and analysis of applicable algorithms to match students to supervisors. It covers a variety of matching algorithms, including content- based filtering, collaborative filtering, user-item matching and stable matching. Also, in details there are described k-means algorithms and its various improvements.
Chapter 4 explains the experiments conducted where for each experiment there is described what dataset has been used and explanation of experiment with corresponding experimental results. It outlines all taken approaches to match students to supervisors and offers a thorough analysis and discussion of the outcomes by comparing them in terms of the defined metrics.

Chapter 5 shows the implemented supervisorship web service. The system's characteristics and functions are presented together with a thorough examination of its possible advantages, drawbacks, and implementation considerations.
The thesis's key conclusions are outlined in the last chapter, which also evaluates whether the thesis's goals and objectives were met.

2. Literature review

2.1 Matching students to supervisors

Scientific supervisors are considered to be staff of universities which are assigned to manage the workflow of their students in the implementation of thesis or dissertation. It is widely known that the relationship between a supervisor and a student is very important for a graduation program to be successful. If the relationship between them is established in a bad way, then it usually results in complete failure of the graduation [1]. Right communication between supervisors and students plays a vital role in the successful implementation of graduation work [2].
The relationship of scientific supervisors with students is important for several reasons:

- Mentorship and guidance: Scientific supervisors serve as mentors to their students, providing guidance and support throughout the research process. They help students develop research skills, provide feedback on their work, and offer advice on career development.
- Professional networking: Scientific supervisors often have established networks within their field of study, which can be beneficial to students seeking opportunities to collaborate with other researchers or find employment after graduation.
- Research opportunities: Scientific supervisors may have ongoing research projects or collaborations that students can participate in, providing valuable research experience and exposure to the latest developments in their field.
- Personal growth and development: Scientific supervisors can help students develop important personal skills, such as critical thinking, problem-solving, and communication. They can also provide emotional support and encouragement throughout the research process.
- Success in academia: The relationship between scientific supervisors and students is an important factor in determining the success of the student in academia. Strong mentorship and guidance can lead to the successful

completion of research projects, publications, and presentations, which are important milestones in an academic career.

Overall, the relationship between scientific supervisors and students is critical for the success of both parties. It provides opportunities for personal and professional growth, networking, and academic success, while also fostering a sense of community and collaboration within the field of study. When a supervisor shares knowledge with a student, then a student gets motivated to persevere in the implementation of their graduation work.

This study [3] investigated the effect of the student-supervisor relationship on final grades, student satisfaction, and perceived supervisor's contribution to the learning of the student. Based on the experimental studies, it was shown that supervisors should be highly affiliated, and the level of control should be carefully balanced.

This study [4] discovered that the presence of abusive supervision has a detrimental impact on psychological capital. Moreover, it was identified that the constructive influence of team member support on the connection between abusive supervision and psychological capital is mediated by the quality of the relationship between supervisors and students. In essence, the positive moderating effect of team member support on the association between abusive supervision and psychological capital is channeled through the intermediary variable of supervisor-student exchange.

One of the most important steps in the process of giving graduate students research projects is matching them with supervisors. The success of research projects can be increased, and a nice experience for both parties can be ensured by carefully matching students with supervisors. Creating effective tools and algorithms for matching students with supervisors has garnered more attention in recent years.

Ismail et al. investigated the problem of matching final-year students with supervisors. Given that the students of their university were not aware of what kind of faculty lecturers they had and what their interests were, they might have been restricting their selection pool by considering only those lecturers they knew. The authors built a recommendation system that matches final year project students with potential supervisors based on students' and supervisors' interests. To implement the system, firstly, the data was collected from students outlining their rate of interest in the project titles of the supervisors' previous projects. Overall, 51 data records were collected. Then the data was passed to the recommendation engine, which by the use of the Euclidean distance function calculated similarities and outputted potential supervisors to students [5].

Kawagoe T and Matsubae investigated the issue of student-supervisor matching. Given that each supervisor had a minimal quota in a Japanese university, and students select supervisors themselves, it could introduce some problems regarding conflicts of interest between students. To resolve this issue, the authors used a

mechanism on the basis of deferred acceptance (DA) and constructed a student-supervisor matching engine. To accomplish the matching, firstly students were involved in interviews with supervisors to construct their order of preferences. After this process, supervisors received the list of students who ranked supervisors, then supervisors ranked students themselves to construct a priority order [6].

Jing Ren et al. conducted a review of the basic models of the matching theory and algorithms in explicit matching. Different kinds of issues throughout the process of implicit matching were analyzed and reviewed by the authors. Various applications of matching algorithms were thoroughly described. Open issues and future directions were outlined at the end of the research. Matching algorithms are considered to be valuable in various fields considering the increase in popularity of science and emerging technologies. The authors paid attention to situations where constructing the order of preferences for each data point where big data is concerned might be too time-consuming which leads to the lack of resources to conduct the experiment. To overcome this issue, they researched various algorithms to predict the order of preferences. They also described a very important concept of matching algorithms which is stability. The results of matching algorithm are stable if no one from any pair wants to leave the matching and form a new pair [7].

2.2 Association analysis

Association analysis is a technique used in data mining and machine learning to find relationships between variables in large datasets. It is often used in market basket analysis, where the objective is to identify the goods that are frequently purchased together, and in healthcare to identify variables that affect the onset of diseases. A search of the related literature revealed that association analysis has been extensively studied, from theoretical bases to practical applications. Some of the key deductions from the literature are as follows:

The core of association analysis is the concept of frequent itemsets, or collections of items that frequently appear together in a dataset. The Apriori approach is widely used to identify frequent itemsets by continually deleting infrequent itemsets until only frequent ones are left. Association analysis can include additional types of associations, such as sequences (where items appear in a specific order) and trees (where items have hierarchical relationships).

In association analysis, dealing with high-dimensional data is a substantial challenge because the number of variables far outnumbers the available observations. As useful tools for dealing with this difficulty, strategies such as feature selection and dimensionality reduction come into play.

Association analysis is used in a variety of disciplines, including market basket analysis, web mining, healthcare, and social network analysis. Recent research has focused on its possible applications in developing domains such as cybersecurity and environmental monitoring. Ongoing research is aimed at improving the efficiency and scalability of association analysis methods. Simultaneously, attempts are being made to develop fresh techniques capable of managing complex data types and to integrate domain knowledge in order to enhance the value of association analysis methodologies.

Generally speaking, association analysis is a powerful technique for finding links in huge datasets, with applications in numerous fields. As a result of ongoing study, the field is anticipated to continue to develop and become accustomed to increasingly more complex and diverse datasets.

The Apriori method, a quick and scalable approach for mining frequent itemsets and association rules, is introduced in this study. The method is described by the authors, who also offer experimental findings demonstrating the system's efficacy on various datasets [8].

In order to mine frequent itemsets, this work introduces the FP-growth algorithm as an alternative to the Apriori approach. The technique is often more effective than Apriori since it uses a tree structure to describe the dataset and does not require candidate generation [9].

This study extends association analysis to the arena of data streams, where the aim is to process data quickly and constantly. The authors offer a framework for mining frequent itemsets in data streams at various time granularities, and they evaluate its effectiveness across a range of real-world datasets [10].

Furthermore, this study introduces a novel approach that combines classification and association rule mining. This hybrid methodology not only elucidates variable interactions, but also forecasts the values of a target variable. The authors demonstrate the effectiveness of this technique by conducting extensive testing on a range of benchmark datasets [11].

This paper offers a thorough overview of association rules mining in huge data, including both more established approaches like Apriori and FP-growth as well as more current ones like parallel and distributed mining. The writers also go through the difficulties and possibilities in the area and suggest numerous lines of future investigation [12].

These publications show the numerous uses of association analysis as well as the current research to increase its effectiveness and scalability.

The Apriori algorithm functions in the manner described below. A well-known approach for frequent itemset mining in transactional databases is the Apriori algorithm. It finds frequent item sets of progressively larger sizes until no more

frequent item sets can be discovered. Here is a step-by-step breakdown of the algorithm's operation.

The first phase entails scanning the transactional database to determine the frequency of occurrence of each item, which serves as the foundation for constructing the set of frequent 1-itemsets.

The procedure then moves on to the production of candidate 2-itemsets in the second step. This is accomplished by utilizing the frequent 1-itemsets discovered in the first phase and generating all possible pairings of items. These pairings are checked against the transactional database to see how frequently they occur, and any potential 2-itemset that falls below the user-specified minimum support criterion is removed.

In the third step, the frequent 2-itemsets from the previous phase are used to generate candidate 3-itemsets. This entails generating all possible triples of items from the frequently occurring 2-itemsets, which are then cross-checked against the transactional database to establish their frequency. Any candidate 3-itemset that does not match the user-specified minimum support criterion is deleted.

The iterative process of producing and pruning potential itemsets continues until no more frequent itemsets can be found. The process usually ends when it is unable to find any more frequent k-itemsets, where 'k' represents the size of the itemsets generated in the previous iteration.

Following the identification of all frequent itemsets, the next step is to use this information to create association rules. These rules are written in the manner "if X then Y," where X and Y represent sets of things. The efficacy of an association rule is measured using metrics such as support and confidence, which are generated from the frequency counts of the rule's constituent itemsets.

The Apriori algorithm has some limitations, such as its high computational complexity for large datasets and the fact that it can only find itemsets that contain at least one frequent item. However, it is still widely used as a benchmark algorithm and as a basis for more advanced frequent itemset mining techniques.

2.3 Time series prediction with linear regression

In a number of disciplines, including engineering, economics, finance, and medical, time series prediction is crucial. To anticipate the future values of time series data, numerous prediction models have been created. Among them, the most often used technique for time series prediction is linear regression. We will talk about the usage of linear regression for time series prediction and its applications in a variety of domains in this literature study.

A statistical approach called linear regression is used to evaluate the relationship between a dependent variable and one or more independent variables. Linear regression is used in time series prediction to anticipate future values of a dependent variable based on its previous values as well as those of one or more independent variables. Linear regression models' predictive strength stems from their capacity to extrapolate future values of time series data by fitting a linear equation to the existing dataset. This linear equation looks like this:

$$Y = b0 + b1X1 + b2X2 + ... + bnXn \quad (2.1)$$

Where Y represents the dependent variable and X1, X2, ..., Xn are the independent variables, the coefficients b0, b1, b2, ..., bn play a crucial role. These coefficients are determined through a least squares method, a statistical technique that minimizes the sum of squared errors between observed values and their corresponding predicted values.
In the context of time series analysis, the dependent variable is often the time series data, with time acting as the independent variable. Time series prediction using linear regression entails fitting a linear equation to the time series data and leveraging this equation to forecast future values of the time series.
The essential steps for employing linear regression in time series prediction include the following.

Data Preparation.
Before applying linear regression to time series data, it is imperative to prepare the dataset by partitioning it into two distinct sets: a training set and a testing set. The testing set serves as the benchmark for evaluating the model's performance, while the training set is utilized to train and fit the linear regression model. Maintaining a clear demarcation between testing and training data is crucial in the context of time series linear regression to ensure an unbiased assessment of the model's effectiveness on unseen data.

- Rolling Window Method: The rolling window method is a technique that involves partitioning time series data into a fixed training set and a sliding testing set. In this approach, the testing set typically comprises one observation at a time, resulting in a small window of data. To illustrate, consider a scenario where we have time series data representing stock prices spanning from January 2018 to December 2022. For the purpose of illustration, an 80% training set can be defined from January 2018 to December 2020, and a corresponding 20% testing set from January 2021 to December 2022. The rolling window technique is then applied to the testing set, dividing it into smaller windows—let's say, one month each. This

process enables the model to be evaluated sequentially on different segments of the time series, offering insights into its performance across various temporal contexts.

- The Fixed Window Approach divides the time series data into training and testing sets with predetermined sizes. The first three years of the data would be included in the training set, and the latter two years of the data would be included in the testing set. As an example, we might split the data into two sets. This approach is useful when we want to train the model with as much data as possible but have limited data.
- Shuffle Split Approach: Using a random mixing technique, this method splits the time series data into training and testing sets. This strategy is useful when we have a large amount of data and want to ensure that the training and testing sets contain a representative sample of the data. For example, it is possible to randomly shuffle the monthly stock price time series data and then separate it into 80% training and 20% testing data.

The primary goal of employing these methods is to evaluate the performance of the linear regression model using simulated or hypothetical data. By partitioning the data into distinct training and testing sets, these methods allow for the assessment of the model's ability to generalize to new, unseen data. This separation is essential to guard against overfitting, a scenario where the model becomes overly tailored to the idiosyncrasies of the training data and may not perform well on new observations.

In summary, the objective is twofold: first, to gauge how effectively the model can generalize to previously unseen data, and second, to mitigate the risk of overfitting by ensuring a rigorous evaluation on a separate testing set. These practices contribute to building robust and reliable linear regression models.

Model Selection.

Choosing the suitable linear regression model is the next step. This entails selecting the linear equation's form and the quantity of predictor variables to include to the model. The autoregressive (AR) model, which uses the time series' lag values as predictors, is the most widely used linear equation in time series analysis.

Various models can be employed for time series linear regression, contingent upon the characteristics of the data and the specific issue at hand. The following are some instances of time series linear regression models.

In the realm of time series analysis and forecasting, the Autoregressive Integrated Moving Average (ARIMA) model stands out as a widely employed method. This model assumes that the time series data needs to be stationary, meaning that its statistical properties remain constant over time. The ARIMA model expresses the

time series as a linear combination of its historical prediction errors and past values.

The model's components—namely, the moving average (MA), autoregressive (AR), and differencing (I) components—are determined by three parameters denoted by the letters p, d, and q. These parameters encapsulate the essential characteristics of the ARIMA model:

- p: Represents the order of the autoregressive component, indicating the number of lagged observations included in the model.
- d: Signifies the order of differencing, representing the number of times the time series data is differenced to achieve stationarity.
- q: Denotes the order of the moving average component, specifying the number of lagged forecast errors integrated into the model.

These parameters play a crucial role in tailoring the ARIMA model to the specific characteristics of the time series data, allowing for effective modeling and forecasting.

An expansion of the ARIMA model, the Seasonal Autoregressive Integrated Moving Average (SARIMA) model was created especially to handle seasonal trends in time series data. To take into consideration the model's autoregressive elements, seasonal variations, and moving average impacts, SARIMA adds more parameters.

When working with several connected time series, the multivariate Vector Auto regression (VAR) Model is utilized. According to this approach, every time series in the system may be represented as a linear combination of all the other time series' past values as well as its own. The model contains parameters for cross-lagged terms that take into account interactions between the various time series as well as autoregressive components unique to each time series.

A flexible time series model that permits the addition of outside effects and nonlinear trends is the Structural Time Series Bayesian Model (BSTS). The posterior distribution of the parameters of this model is estimated by use of the Markov Chain Monte Carlo (MCMC) approach, which has its roots in Bayesian statistics.

Generalized Autoregressive Conditional Heteroscedasticity (GARCH) Model: For simulating the volatility of financial time series data, the GARCH model is frequently utilized. The model makes the assumption that the time series data' variance is a function of both its historical values and historical forecasting mistakes. The variance equation, the moving average term, and the autoregressive term each have parameters in the model.

In a linear regression framework, these models can be helpful for analyzing and projecting time series data, and they can reveal underlying patterns and trends in the data.

Model Fitting.

The third phase involves applying an appropriate technique, like maximum likelihood estimation or least squares, to fit the linear regression model to the training set of data. This entails calculating the linear equation's coefficients, which show how the independent and dependent variables are related to one another.

For time series data, there are multiple techniques for estimating the parameters of a linear regression model. Least squares and maximum likelihood estimation are two popular techniques.

Least Squares: The least squares method is, in fact, a fundamental strategy for estimating the parameters of a linear regression model. The sum of squared discrepancies between the projected values generated from the independent variables and the observed values of the dependent variable is minimized using this method. Similarly, when fitting an autoregressive model to time series data, a similar approach is used.

When dealing with time series data, the least squares method is used to find the parameters of the autoregressive model by minimizing the sum of squared errors between the observed and predicted values of the time series. This optimization method ensures that the autoregressive model is as near to the observed data as possible, allowing

Consider a time series of the monthly sales data for a corporation. Our goal is to anticipate future sales using an autoregressive model based on prior months' sales. We could use the least squares method to get the values of the autoregressive coefficients.

One possible form of the model is:

$$Sales(t) = c + a1Sales(t - 1) + a2Sales(t - 2) + ... + ap * Sales(t - p) + e(t) \quad (2.2)$$

where Sales(t) is the value of the time series at time t, c is a constant term, a1 through ap are the autoregressive coefficients, p is the order of the model, and e(t) is the error term. We would then use least squares to estimate the values of the coefficients a1 through ap.

Maximum Likelihood Estimation: A method for estimating a statistical model's parameters is called maximum likelihood estimation, or MLE. It works by determining the values that maximize the likelihood of the observed data. In the context of linear regression for time series data, this means specifying a probability distribution for the error component and figuring out the values of the autoregressive coefficients that maximize the likelihood of the observed data given the defined distribution.

As an example, consider a time series of the daily temperature values for a particular location. We would like to use an autoregressive model to predict future temperatures based on the temperatures recorded in the preceding days. The values

of the autoregressive coefficients that maximize the likelihood of the observed data under this assumption could be estimated using MLE under the assumption that the error term has a normal distribution. One possible form of the model is:

$$T(t) = c + a1 * T(t-1) + a2 * T(t-2) + ... + ap * T(t-p) + e(t) \quad (2.3)$$

where Temperature(t) is the value of the time series at time t, c is a constant term, a1 through ap are the autoregressive coefficients, p is the order of the model, and e(t) is the error term, assumed to be normally distributed with mean 0 and constant variance. We would then use MLE to estimate the values of the coefficients a1 through ap.

Bayesian Methods: Bayesian techniques are another way to determine a linear regression model's parameters for time series data. Using these methods, a prior distribution is applied to the parameters, and the Bayes theorem is used to update it in light of the observed data.

In conclusion, least squares estimation, maximum likelihood estimation, and Bayesian approaches are available for estimating the parameters of a linear regression model for time series data. The specific circumstances at hand and the assumptions that make sense in light of the information at hand determine which strategy is best. Every technique has benefits and drawbacks of its own.

Model validation.

The performance of the time series linear regression model is evaluated using the testing data in the fourth stage of model evaluation. This entails comparing the time series' expected and actual values. Several essential performance indicators are produced to characterize the model's performance, including mean squared error (MSE), root mean square error (RMSE), and mean absolute error (MAE).

The MSE is determined as the average of the squared discrepancies between the expected and actual values. It gives a simple measure of the average squared difference between expected and observed values.

The square root of the mean squared error yields the root mean square error (RMSE). This measure is frequently preferred since it uses the same units as the original data, making it easier to read. The root mean square error (RMSE) measures the typical magnitude of errors between expected and observed values.

Another often used statistic is the mean absolute error (MAE). It represents the average absolute difference between expected and actual values, providing a measure of error magnitude without the influence of squared terms.

These performance measures contribute to a full evaluation of the time series linear regression model, providing insights into its forecasting accuracy and efficacy.

There is used the Mean Absolute Error (MAE), which quantifies this difference in an absolute, non-squared manner, to measure the mean absolute difference between expected and actual values.
Larger errors are given more weight in MSE and RMSE, whereas MAE handles all errors equally. The situation at hand and the required balance between precision and interpretability will determine the best metric to use.
Let's look at an example where we want to predict a company's daily stock price based on its stock price over the previous p days. We can use linear regression for time series to assess the link between the historical and current stock prices. We can use MSE, RMSE, and MAE to evaluate the model's performance once it has been fitted. Due to the fact that the MSE is expressed in square units, a score of 100 suggests that the projected stock price is frequently off by 10. An RMSE of 10 means that the expected stock price is usually 10 dollars off because it is expressed in the same units as the original data. MAE of 8 means that on average, the predicted stock price is off by 8 dollars, regardless of whether the actual stock price is high or low.

Model Refinement.
The final step is to refine the model if necessary by adjusting the model parameters or selecting a different form of the linear equation. This may involve repeating the previous steps until the desired level of prediction accuracy is achieved.
The process of improving a model's performance in linear regression for time series involves either altering the model's parameters or selecting a selection of variables that explains the data the best. The residuals—the differences between the actual and expected values of the dependent variable—are examined in order to achieve this. One way to improve a model is by stepwise regression, which involves adding or removing variables one at a time according to their statistical significance. Enhancing the predictive performance of a model can be achieved through various techniques, and two common strategies are forward selection and backward elimination. In forward selection, variables are added to the model one at a time, evaluating their impact on model performance. Conversely, in backward elimination, variables are systematically removed from the model to refine its structure.
Regularization is another effective approach for model improvement. This involves adding a penalty term to the cost function, discouraging overfitting and promoting a more generalized model. Techniques such as Lasso regression and Ridge regression are commonly employed for regularization.
After implementing these model enhancement techniques, it is crucial to assess the effectiveness of the upgraded models. Evaluation metrics such as mean squared error (MSE), root mean square error (RMSE), and mean absolute error (MAE) can

be utilized to measure the performance of the models and gauge their accuracy in predicting outcomes. These metrics provide valuable insights into the model's ability to generalize and make accurate predictions on new, unseen data.
Take the time series linear regression model below, for instance:

$$Y(t) = \beta 0 + \beta 1 X1(t) + \beta 2 X2(t) + \epsilon(t) \quad (2.4)$$

where Y (t) is the dependent variable, X1(t) and X2(t) are the independent variables,
β0, β1, and β2 are the model parameters, and ϵ(t) is the error term.
Let's imagine we want to select the most important variables in order to enhance the model. We can use backward elimination to exclude variables that are not statistically significant.
Include both variables for a complete model fit. If the p-values for each variable are more than a preset level of significance, such as 0.05, you should eliminate the variable with the highest p-value.
Using the remaining variables, fit the simplified model. Steps 2 and 3 should be repeated until all variables have p-values less than the significance level. Metrics like as mean squared error, root mean square error, and mean absolute error can be used to assess the model's performance after refinement.
Regularization can be used to reduce overfitting and improve the model even further. Ridge regression is a widely used regularization approach that adds a penalty term to the cost function equal to the sum of the squared values of the model coefficients multiplied by a regularization parameter, lambda. The regularization parameter regulates the magnitude of the cost function penalty and can be changed to improve model performance.
To determine the ideal value for the regularization parameter, we can apply cross-validation, which divides the data into training and validation sets. The model is then trained on the training set and assessed on the validation set. This process is repeated for several values of the regularization parameter, and the regularization parameter value that yields the lowest validation error is selected.
To put it simply, model refining is an iterative process that increases the accuracy of the model by changing its parameters while choosing a subset of variables that best explains the data. Metrics like mean absolute error, root mean square error, and mean squared error can be used to evaluate the performance of the upgraded models. Regularization reduces overfitting, and cross-validation helps determine the ideal regularization parameter value.
Applications of Linear Regression for Time Series Prediction.
Many fields have employed linear regression for time series prediction. For instance, linear regression has been used in finance to forecast stock prices by

taking historical prices and economic data into account. In a 2019 study, Jeong et al. employed linear regression to forecast Samsung Electronics' stock price based on historical stock prices as well as the US dollar, gold, and crude oil prices [13].
Based on patient data, linear regression has been used in medicine to forecast how diseases would progress. Based on patient characteristics such age, gender, and cognitive scores, Hsieh et al. (2019) used linear regression to predict the course of Alzheimer's disease [14].
In engineering, linear regression has been used to forecast machine performance based on maintenance history and historical performance. Based on historical performance and maintenance data, a study by Guo et al. (2018) employed linear regression to estimate the remaining usable life of aircraft engines [15].
In conclusion, because of its efficiency and ease of use, linear regression is a widely used technique for time series prediction. The ability to forecast future values of time series data based on historical values and the values of one or more independent variables has been applied in a variety of industries. For time series data with complicated patterns or nonlinear relationships, linear regression might not be the best option.
In such cases, more advanced methods such as neural networks or support vector regression may be more appropriate.

2.4 Comparison with other research work

Tables 2.1 shows comparison with other research work.

Table 2.1: Comparison with other research work

Paper	Theme	Methodology	Key Findings
[16]	Identification of supervisory relationships and differences in perception	Usage of semi-structured interviews for data collection. 23 supervisors from 16 institutions were invited, and 14 actual interviews with supervisors.	Found important differences between the perception of supervisors in their role and actual students' needs.

[17]	Tracking postgraduate supervision	Focus group with students and faculty staff, comprehensive literature review in the process of defining important questions	Importance of setting short-term and long-term goals and their tracking to increase the chances of a successful thesis defense.
[18]	Multi-agent framework for research supervision	The analysis of the most efficient and standard research activities, as well as the development of a research supervision management system based on multi-agent systems.	An agent-based architecture is proposed to handle the supervision process in the research setting.
[5]	Recommender engine for student-supervisor matching	Usage of the Euclidean distance algorithm by calculating similarity based on the results of a questionnaire given to students and supervisors.	A simple but feasible way to make recommendations for students in the process of selecting their supervisors, thereby reducing the chances of mistakes in the allocation process.
[19]	Multi-objective evolutionary approach	Collecting data from students and supervisors regarding their preferences for research topics and applying a genetic algorithm to make a match in	Presenting a genetic algorithm that can match students to supervisors.

		this regard	
[20]	Thesis supervisor recommendation with content retrieval	Using information retrieval to analyze students' proposals and recommending supervisors based on that regard.	Grouping supervisors based on their research work. The average accuracy was 53.09%. Considering only the top 10 words in supervisors' profiles can be effective in information retrieval.
[21]	System for student- supervisor matching with reinforcement learning	Reinforcement learning is used to create a model of students' interests, in which students alter terms from faculty members' papers to explain their interests.	The authors tackled the issue of cases when there is very little information about potential supervisors or when students struggle in formulating exact queries regarding their research interests.
[6]	Matching with minimal quota	Usage of two-sided matching and consideration of type-specific minimal and maximum quotas. In total, there were 254 students and 67 supervisors.	The system eliminates justified envy across students of the same "type".
This study	Development of a supervisorship system with a tracking process and matching students to supervisors based on	Collecting data regarding the perceptions of student-supervisor relationships from each side, constructing preferences based on this regard, and then	The selection of an algorithm should align with specific project requirements. The Genetic Algorithm excels in preference satisfaction (0.91), while One-to-Many Gale Shapley offers efficient execution with low time and space complexity and a balance of workload

	multidimensional analysis of matching algorithms and usage of a dataset related to the perceptions of students and supervisors	implementing matching by the use of Gale-Shapley, genetic algorithm, collaborative filtering, k-means. Integrating the matching algorithm into a service that also has a tracking feature.	(0.5). Collaborative Filtering, despite relatively high compatibility scores of 0.46, may introduce workload imbalances and higher computational costs.

2.5 Knowledge gap

Based on the literature review, it can be observed that none of the found work performed multidimensional analysis of matching students to supervisors specific to the context of university settings in terms of psychological perceptions of students and supervisors regarding their workflows and none of the work made a supervisorship system that integrates matching feature with tracking progress of students to facilitate diploma preparation and improve research outcomes.

3. Analysis of applicable algorithms to match students to supervisors

3.1 Matching algorithms

Pairing two objects together in a way that is appropriate and generally accepted is the task of matching. The matching algorithm should take into account these restrictions and rules to construct a matching in a way that results in stability of matched pairings. There may be a collection of rules and preferences that surround the context of the elements that participate in the matching process.

In many areas of life, from market activities to daily operations, matching is crucial. It can be used to pair products in a store with prospective customers, match a group of entities while taking into account the fact that they belong to distinct

sets and have varied tastes, and more [5]—[7], [22], [23]. The abstract types of matching tasks are the following.

One-to-one matching.

There is a lot of work dedicated to one-to-one matching [24]—[27]. Basically, one-to-one matching states that each element in one set is paired with exactly one element in the other in a process known as one-to-one matching. In other words, it is a matching when the components of the two sets have a singular, exclusive correspondence.

For example, consider two sets A and B, where A = 1, 2, 3 and B = a, b, c. A one-to-one matching between these two sets might be:

1 to a
2 to b
3 to c

Accordingly, every element in set A is paired with exactly one element in set B, and vice versa. There are two requirements that must be met for a match to be one-to-one: first, no two elements from set A can match the same element from set B, and second, no two elements from set B can match the same element from set A.

Numerous disciplines, such as mathematics, computer science, economics, and biology, use one-to-one matching. It can be used, for instance, to pair tutors with students, employees with tasks, or patients with therapies in clinical studies.

Concrete examples of one-to-one matching:

- A school wishes to pair up each student with a teacher for individualized mentorship sessions. The stable marriage method, also known as the Gale-Shapley algorithm, can help the school resolve this matching issue. The algorithm selects the student that each teacher prefers the most after iteratively enabling each student to propose to their preferred teacher. The process is repeated until every student is paired with a teacher, and the results are stable, meaning that neither the students nor the teachers would choose their present pairing over any other.
- A corporation seeks to place qualified job candidates in positions that match their abilities and qualifications. Algorithm: This matching issue can be resolved using the Hungarian algorithm, also referred to as the Kuhn-Munkres algorithm.
- A matrix that depicts the preferences or compatibility between candidates and positions is used by the algorithm. By repeatedly looking for the maximum-weight match in the bipartite graph the matrix defines, it determines the best assignment. The final pairing of candidates to roles represents the best possible assignment of candidates.

- A hospital wishes to match each patient with a qualified doctor based on their health status and the doctor's area of expertise. For this matching problem, the greatest cardinality matching algorithm can be used. The matching problem is transformed into a graph by the algorithm, where each patient and doctor is represented by a node, and the edges signify how well-matched they are. The hospital can allocate each patient to a doctor in a one-to-one manner and improve the overall quality of care by locating the highest cardinality matching in the graph.

Many-to-one matching.
A sort of matching called many-to-one matching allows for the pairing of several elements from one collection with just one from another [28]—[30]. In other words, it is a matching in which the components of the two sets do not always correlate in a singular, exclusive way.
For example, consider two sets A and B, where A = 1, 2, 3, 4 and B = a, b. A many-to-one matching between these two sets might be:

- 1 to a
- 2 to b
- 3 to b
- 4 to a

This shows that two elements from set A, specifically 2 and 3, are matched with the same element in set B (specifically, b). It is important to note that only a many-to-one matching may satisfy the first criteria of a one-to-one matching, which specifies that no two elements from set A may be matched with the same element from set B.
Numerous disciplines, including economics, game theory, and social choice theory, use many-to-one matching. It can be used, for example, to replicate situations where multiple agents may be assigned to a single task or resource, such as when a team of employees is assigned to a project or a class of students is assigned to a certain institution. Many-to-one matching is also widely used in market design to match buyers and sellers in many sorts of marketplaces.
Concrete examples of many-to-one matching

- Matching jobs to Available Workers: In this scenario, a corporation must distribute jobs to workers who can manage a variety of duties. The maximal bipartite matching algorithm or the Hungarian algorithm can be applied in this case. The matching issue is transformed by these algorithms into a bipartite graph, where one set of nodes corresponds to the tasks, another set to the employees, and the edges denote the degree of compatibility between the two sets of nodes. The business can distribute jobs to workers in a

one-to-many manner, improving overall efficiency and workload distribution, by locating the maximum matching in the bipartite graph.

- A marketplace platform must connect suppliers with client expectations in a scenario where each supplier can satisfy a number of demands. For this matching problem, the greedy method can be used. The algorithm assigns the best supplier to each demand iteratively depending on variables like price, proximity to the demand, or customer reviews. Until all criteria are met and each supplier can match numerous demands, the procedure is repeated.
- An online publishing platform seeks to pair together publishers and subscribers for the delivery of material, however each publisher may have several subscribers. Stable marriage or the postponed acceptance algorithms can be applied in this situation. With the use of these algorithms, publishers can suggest their content to preferred subscribers, who then pick the publisher they like the best. The process keeps going until the matches are steady and profitable for both publishers, and each has reached the target number of subscribers.
- Employees must be assigned to project teams, and each employee may be a member of more than one team, in the following scenario. Approach: This matching problem can be solved using the maximum flow approach. The project teams and employees are represented as nodes in the algorithm's flow network representation of the assignment, while the capacity limitations and preferences are represented as edges. The organization can assign workers to project teams in a one-to-many manner, maximizing the usage of skills and resources, by determining the maximum flow through the network.

Many-to-many matching.

Many-to-many matching is a type of matching where multiple elements in one set can be paired with multiple elements in another set. In other words, it is a matching in which there is not necessarily a unique, exclusive correspondence between the elements of the two sets.

For example, consider two sets A and B, where A = 1, 2, 3 and B = a, b, c. A many-to-many matching between these two sets might be:

- 1 to a,b
- 2 to b,c
- 3 to b
- 3 to a,c

This means that multiple elements in set A are paired with multiple elements in set B, and vice versa. In this example, element 1 in set A is paired with both elements

A and B in set B, and element b in set B is paired with both elements 1, 2, and 3 in set A.

It is important to note that a many-to-many matching can satisfy neither of the two conditions of a one-to-one matching, which are that no two elements from set A can be matched with the same element from set B, and no two elements from set B can be matched with the same element from set A.

Numerous disciplines, including computer science, economics, and social choice theory, use many-to-many matching. It can be used, for instance, to model scenarios in which numerous agents can be matched with various tasks or resources, such assigning a group of employees to various projects or a group of students to various educational institutions. Many-to-many matching is also extensively employed in network analysis, where it is used to model relationships between nodes in a network. A few illustrations of many-to-many matching are as follows:

- Job Applicants and Job Positions: There are many job applicants and many job positions in a recruitment process. An applicant for a job may submit more than one application, and there may be more than one applicant for each vacancy.
- In a learning environment, students may enroll in multiple courses, and each course may have a number of students. This enables students to select courses based on their academic needs and areas of interest.
- Projects and Freelancers: Freelancers or independent contractors can work on a variety of projects, and each project may involve a number of different freelancers. This is typical in fields like software development or artistic creation.
- Mentors and Mentees: In a mentoring program, mentors may be paired with more than one mentee, and each mentee may have more than one mentor. As a result, mentees can receive a wide spectrum of instruction and support.
- Applications for Dating: Many-to-many matching is common on dating services. Multiple people can communicate or indicate interest in one another, creating a variety of connections and prospective pairings.
- Airlines and flight routes: Airlines run a variety of flight routes, and different airlines may fly on a given route. To get to their destinations, passengers can select from a variety of airlines and travel options.

These are merely a few of instances to demonstrate the idea of many-to-many matching. Each time, a number of entities from one group are matched with a number of entities from another set, providing for adaptability and a variety of connections between the entities.

Retrieval matching.

Information retrieval systems, like search engines, use retrieval matching as a type of matching. It involves matching user queries with pertinent documents or other resources in a database based on how closely the two match.
Retrieval matching aims to return the documents that are most pertinent to a particular user query. Techniques including keyword matching, semantic analysis, and machine learning algorithms are used to do this.
The simplest type of retrieval matching is keyword matching, in which documents are matched based on the existence of particular terms in both the query and the document. This method is useful for straightforward searches with a small number of keywords, but it may be less efficient for more intricate questions or queries with several alternative interpretations.
On the other hand, semantic analysis examines the meaning of the query and the document to ascertain how similar they are. By considering the context and aim of the query, this method can assist in retrieving documents that are more pertinent to the user's search. Techniques including NLP, topic modeling, and machine learning algorithms can be used to perform semantic analysis.
Large amounts of data can be analyzed to find patterns, and then predictions can be made based on those patterns using machine learning algorithms. Machine learning techniques can be used in retrieval matching to examine the connections between queries and documents and to foretell which documents are most likely to be pertinent for a specific query.
In general, retrieval matching is an essential part of information retrieval systems because it ensures that users can easily and rapidly access the resources or documents that are most pertinent to their needs.
The practice of matching a query or search request with pertinent data or documents from a database is known as retrieval matching. Here are some illustrations of retrieval matching across several domains:

- Web search: A search engine like Google uses retrieval matching algorithms to match your query with pertinent web pages in its index when you enter a search term. The search engine determines the best matches for your query by taking into account variables like keyword relevancy, page authority, and user behavior.
- Retrieval matching is a technique used in information retrieval systems, such as document databases or digital libraries, to match user searches with pertinent documents. To get the most pertinent documents, the system analyzes the query keywords or phrases with the document's text and metadata.
- Retrieval matching algorithms are used by music streaming sites to suggest tracks or playlists to users. In order to match user choices with appropriate

songs, artists, or playlists in the platform's database, the system analyzes user listening habits, preferences, and comparable user behavior.

- Image Search: To match a user's query image with visually related images in their database, image search engines use retrieval matching techniques. To locate matched photos or visually relevant content, the system examines visual elements including color, texture, and shape.
- Retrieval matching is essential to the operation of question-answering systems. When a user submits a query, the system searches a knowledge base or database of documents to find pertinent responses. To determine the most relevant response, the matching method takes into account both the semantic meaning of the question and the content of the potential answers.

These are but a few of instances of retrieval matching in various applications. The basic idea is to successfully satisfy user information needs by matching their queries with pertinent data or content. Depending on the domain and the features of the data being matched, the specific algorithms and approaches used may change.

User-item matching in recommendation systems.

User-item matching is a type of matching used in recommendation systems to match users with relevant items they may be interested in. There is a lot of applications of user-item matching [31]—[40]. Recommendation systems use a variety of techniques to match users with items, including collaborative filtering, content-based filtering, and hybrid approaches.

Collaborative filtering is a popular technique for user-item matching that is based on an analysis of user behavior such as past purchases, ratings, and browsing history. This method seeks to find user similarities and recommend things based on the interests of similar users. Collaborative filtering is widely classified into two types:

- User-Based Collaborative Filtering: this method suggests products to a target user based on the preferences of similar users. Through behavioral and preference analysis, similar users are identified, and the target user is subsequently recommended highly rated things by those similar users.
- Item-Based Collaborative Filtering: Items are recommended to a user based on their similarity in this method. Analyzing the features and characteristics of similar items identifies them. Items that are similar to those highly rated by the user or that share common characteristics are recommended to the target user.

In addition to collaborative filtering, content-based filtering is a way for matching user-items. This technique suggests goods based on a user's previous choices by assessing the objects' qualities and attributes. To produce individualized recommendations, content-based filtering can analyze the components of a recipe or the genre of a movie.

In recommendation systems, hybrid techniques combine collaborative filtering with content-based filtering. These hybrid models strive to deliver more accurate and personalized suggestions to users by exploiting a combination of user behavior and item features, solving some of the constraints inherent in individual recommendation systems.

Overall, user-item matching is an important component of recommendation systems, as it helps to ensure that users are provided with personalized recommendations that are relevant to their preferences and interests. By leveraging user behavior and item attributes, recommendation systems can provide more accurate and useful recommendations to users, which can ultimately lead to increased user engagement and satisfaction.

Regarding user-item matching, recommendation systems are widely used to solve the problem.

The recommendation system is a software and important software application in the research area. The system was invented so that users could cope with the information load. This is accomplished by a content filtering mechanism that is based on preferences and interests [41]. It was also found that a good effective system is financially beneficial. It turned out that if the recommendation is good enough for sale, then the consumer buys the product 35 percent more often [42]. This is a class of information filtering system that predicts a "preference" or "rating" that is given to a user [43]. Therefore, we can say that the recommendation system is also useful for sellers. Thanks to the recommendation system, the sales increase. Recommendation systems are an important area of research and the first paper containing information on filtration appeared in the 1990s. All fields as an industry, scientific work on the development of new approaches over the past decades worked to improve the quality of recommendation systems [43].

These are some examples of the most popular products using the recommendation systems:

- Amazon, which is one of the most popular e-commerce sites, uses its own recommendation system for users. When selecting an item for purchase, the system recommends other items based on the buyer’s choice. Amazon patented this system and called item-to-item collaborative filtering [43].
- YouTube also uses its own system for recommended videos, which is based on maintaining user privacy and ensuring the control of personal user data issued by the back-end system for download [43].
- LinkedIn has a strong relationship with the professional world, which is focused on business, among social networks. LinkedIn generates recommendations for persons with whom the user may be acquainted, projects that may be of interest to the user, or groups and companies that may be of interest to you. Reid Hoffman, Allen Blue, Konstantin Guericke,

Eric Ly, and Jean-Luc Vaillant founded it in 2002, and it now has over 310 million users. LinkedIn employs Apache Hadoop, a computational framework that has recently emerged as one of the most popular systems for distributed storage and parallel processing of huge data [43][44][45][46].

Recommendation systems can be classified into three categories.

Content-based filtering technique.

Content-based filtering is a recommendation system technique that suggests items to users based on their preferences and interests. The recommendation system analyzes the features of products that the user has liked or interacted with and recommends items with comparable features in this technique.

The content-based filtering technique relies on the idea that users who like certain items are likely to enjoy other items with similar characteristics. To implement this technique, the recommendation system first analyzes the features of the items in the dataset. These features can include attributes such as genre, artist, director, rating, and so on, depending on the type of items being recommended.

After identifying the features, the recommendation system uses them to create a user profile based on the items the user has engaged with or evaluated. The system then suggests new items with features similar to those in the user's profile.

Consider a movie recommendation system that employs content-based filtering. If a user likes multiple action movies with high ratings, the recommendation system will suggest other action movies with comparable characteristics such as high-intensity action sequences, compelling plot lines, and high production qualities.

When there is limited user interaction data or when the purpose is to propose things based on specific traits or features, content-based filtering can be especially effective. However, one limitation of this technique is that it may lead to a narrow set of recommendations, as it only recommends items with similar features to those the user has interacted with in the past.

Overall, content-based filtering is a powerful technique used in recommendation systems to recommend items to users based on their preferences and interests by analyzing the features of items in the dataset.

Collaborative filtering technique.

Collaborative filtering is a recommendation system technique that recommends products to consumers based on the preferences and behavior of other users who share similar interests. The recommendation system in this technique analyzes the interactions of many users with things and discovers patterns and similarities in their behavior to propose items to a specific user.

User-based collaborative filtering.

In this method, the recommendation system finds people that have the target user’s likes and behaviors and suggests products that these users have positively

connected with. For instance, if a user has rated and loved multiple films, the system will find other users who have also like those films and will suggest further films that they would like. The user-based collaborative filtering procedure frequently includes the following steps.

Data Collection: the initial step in building a recommendation system involves collecting information about user preferences or habits. This information can be acquired through the utilization of both explicit and implicit feedback mechanisms. Explicit feedback includes user-provided reviews, ratings, or explicit declarations of preference, while implicit feedback encompasses more subtle indicators such as browsing history, purchase records, or other user interactions.

To organize and represent this data effectively, a matrix is commonly employed. In this matrix, each row corresponds to a user, and each column corresponds to an object (e.g., items, products, services). The interactions or preferences of users for specific items are then represented by the values in the matrix. For instance, a cell at the intersection of a user's row and an item's column may contain a rating, a purchase history indicator, or another form of feedback reflecting the user's preference for that particular item.

This matrix-based representation forms the foundation for collaborative filtering techniques, where patterns and similarities among users or items are analyzed to generate personalized recommendations. The information encapsulated in this matrix is pivotal for training recommendation models and enhancing the accuracy of the system's suggestions.

The following stage is to determine how similar different users are based on their preferences. Cosine similarity and Pearson correlation coefficient are two examples of similarity measurements. These measures assess how closely two users' preference vectors resemble one another. For instance, two individuals will have a greater similarity score if they evaluated a number of products similarly.

The following stage is to determine how similar different users are based on their preferences. Cosine similarity and Pearson correlation coefficient are two examples of similarity measurements. These measures assess how closely two users' preference vectors resemble one another. For instance, two individuals will have a greater similarity score if they evaluated a number of products similarly.

Prediction Generation: The system forecasts the target user's choice for items they have not yet interacted with after choosing the closest neighbors. By combining the preferences of those things' closest neighbors, this is accomplished. The similarity scores between the target user and the neighbors can be utilized as weights in various aggregation approaches, such as weighted averaging. A list of recommendations can then be created by ranking the projected preferences.

A list of recommendations is then produced by the system by choosing the top-ranked items from the projected preferences. System specifications or user preferences may be used to determine how many recommendations to offer.
Collaborative filtering with user input has many benefits. When there is enough user data available, it can make precise recommendations and is straightforward to implement and understand. By drawing on the preferences of other users who have similar preferences, it can help address the "cold start" issue, when new users have little or no data. When there are many users or objects, user-based collaborative filtering may have scalability problems since it can become computationally expensive to generate predictions and perform similarity computations. Furthermore, it might not fully reflect user preferences, such as enduring interests or changing preferences, and it might be vulnerable to data sparsity problems when some users have a very low number of recorded interactions or preferences.

Item-based collaborative filtering.
In user-based collaborative filtering, the recommendation system explores the connections between items and suggests products that are similar to those that the target user has positively interacted with. For example, if a user has enjoyed and given high ratings to several action films, the algorithm will recommend additional action films with comparable themes, stories, and genres. This approach relies on identifying users with similar preferences and recommending items that have appealed to users with analogous tastes.
On the other hand, item-based collaborative filtering takes a different approach. Instead of emphasizing similarities between users, this method focuses on determining the similarity between items based on user preferences. The underlying assumption is that users who have previously expressed a preference for similar items will likely continue to do so in the future. Consequently, if a user has shown a liking for specific movies or products, item-based collaborative filtering recommends additional items that share similarities with those the user has positively engaged with.
In the development of personalized recommendation systems, both user-based and item-based collaborative filtering play important roles. They exploit trends and similarities in user interactions to make tailored recommendations, resulting in a more fulfilling and relevant user experience.
The item-based collaborative filtering method typically consists of the following steps: Data Gathering: First, similar to user-based collaborative filtering, data on user preferences or behaviors is gathered. This information can be gathered through the use of both explicit and implicit feedback, such as user ratings and reviews, browsing or purchase histories. The data is described using a matrix, where each item is represented by each column and a user by each row. The user's

interaction with or preference for each item is represented by the values in the matrix.

Calculating Similarity: based on user choices, the next stage is to determine how similar two items are. Cosine similarity and Pearson correlation coefficient are two examples of similarity measurements. These measurements assess how closely two items' preference vectors resemble one another. For instance, a pair of items will have a greater similarity score if people have given them comparable ratings.

The system chooses a selection of the nearest neighbors for each item after calculating the similarity scores between items. The items with the highest similarity scores to the target item are usually the closest neighbors.

Prediction Generation: the system forecasts the target user's choice for items they have not yet interacted with after choosing the closest neighbors. By combining the preferences of the user's closest neighbors with whom they have engaged, this is accomplished. It is possible to utilize a variety of aggregation approaches, such as weighted averaging, where the weights are the similarity scores between the target item and its neighbors. A list of recommendations can then be created by ranking the projected preferences.

A list of recommendations is then produced by the system by choosing the top-ranked items from the projected preferences. System specifications or user preferences may be used to determine how many recommendations to offer.

Collaborative filtering with items offers many benefits. As the similarity computations are focused on things rather than users, it can handle the scaling problems that user-based collaborative filtering may encounter. When items' features are more stable than users' preferences, it can also capture item similarities more accurately than user-based collaborative filtering. Item-based collaborative filtering is also appropriate when the quantity of items is relatively low in comparison to the quantity of users. Item-based collaborative filtering does, however, have some drawbacks. When dealing with new items that have scant or no interaction data, it sometimes struggles with the "cold start" problem. Certain user preferences, which are unique to each user and aren't always connected to item similarity, might not be able to be captured. When certain things have very few interactions or preferences recorded, data sparsity can potentially provide problems.

Item-based collaborative filtering is an effective method that may deliver precise recommend based on user preferences and item similarity. To enhance the overall effectiveness of recommender systems, it is frequently used in conjunction with other recommendation strategies.

Because it may recommend items based on the behavior of many users, not just the target user, collaborative filtering is a powerful tool in recommendation systems. This means that even if the target user has only had a few interactions with the

system, the recommendation engine can still make effective recommendations based on the behavior of other users with similar interests.

The cold start problem, which arises when there is insufficient user interaction data to find patterns and similarities in behavior, is one restriction of collaborative filtering. To deliver more accurate and meaningful recommendations in such instances, hybrid approaches combining collaborative filtering and content-based filtering techniques can be deployed.

The term "cold start problem" describes a problem that occurs across many industries, but is especially prevalent in the fields of machine learning and recommender systems. It happens when a system comes across a brand-new human, object, or circumstance for which there is insufficient historical data to reliably forecast outcomes or offer helpful advice. The term "cold start" describes a system that begins "cold" with no past knowledge of the new entity.

There are several forms of cold start issues, including:

A platform or service that needs personalization or recommendation capabilities has a user cold start. The algorithm finds it difficult to offer individualized recommendations because it lacks any previous information about the user’s preferences or behavior.

Item cold start: In this scenario, a brand-new product is added to a platform or catalog with little to no previous data accessible. As a result, the algorithm has trouble appropriately ranking or recommending the new item to users.

In recommendation systems that largely rely on contextual data, such as time, location, or user context, the context cold start issue might occur. The system may struggle to give pertinent recommendations based on the present circumstance when there is inadequate context data.

Addressing the cold start problem requires various strategies:

- Content-based recommendations: The algorithm can offer suggestions based on similarities to other goods by examining the traits or attributes of the objects themselves. For situations where an item needs to start cold, this method is useful.
- Utilizing data from user-item interactions, collaborative filtering approaches can find comparable individuals or objects and provide recommendations. However, this strategy is less successful in a cold start scenario where there is little to no interaction data available.
- Hybrid strategies: Reducing the cold start issue may be accomplished by combining several techniques, such as content-based and collaborative filtering. The system can generate more accurate recommendations by including user data, item qualities, and other information.

- Active learning and exploration: The system can proactively ask users for input or preferences rather than passively waiting for user interactions to collect data and lessen the cold start issue.
- Popular recommendations: Until it has enough information about users' interests, the system can initially rely on popular or trending items to produce recommendations.

It's vital to remember that the specific strategy for tackling the cold start problem may change depending on the domain, data that is accessible and technical specifications.

A common and efficient method used in recommendation systems across numerous disciplines is collaborative filtering. To provide precise and individualized item recommendations to a particular user, it makes use of the behavior and tastes of a large number of users. Collaborative filtering's fundamental tenet is that individuals are more likely to hold similar preferences in the future if they have previously displayed similar behavior or interests.

In collaborative filtering, the system collects and analyzes past user-item interaction data, such ratings, reviews, or purchase history. To find patterns and relationships between people and items, start with this information. Using content-based and collaborative filtering approaches while making accuracy a forward-looking factor is the hybrid technique [43]. In recommendation systems, a hybrid technique combines two or more distinct recommendation strategies to produce recommendations that are more varied and accurate. Overcoming the shortcomings
of individual recommendation techniques and enhancing the recommendation system's overall performance are the objectives of employing a hybrid approach.

There are several ways in which hybrid techniques can be implemented in recommendation systems:

- Weighted hybrid approach: In this approach, the recommendation system assigns weights to different recommendation techniques based on their performance and combines their outputs to provide a final recommendation. For example, if content- based filtering performs well for certain users, while collaborative filtering performs better for others, the recommendation system could assign different weights to each technique based on the user's interaction history and combine their outputs to provide a final recommendation.
- Switching hybrid approach: In this approach, the recommendation system selects the most appropriate recommendation technique based on the user's interaction history and behavior. For example, if a user has interacted with the system frequently and has provided many ratings, the recommendation

system could switch to collaborative filtering, which is better suited to identifying patterns and similarities in user behavior.

- The feature combination hybrid approach combines the properties of several recommendation systems to produce more accurate and diverse recommendations. A hybrid approach, for example, might combine content-based filtering and collaborative filtering by identifying trends in user preferences using item attributes and user behavior.

Hybrid techniques can provide significant benefits over individual recommendation techniques, such as increased accuracy, diversity, and coverage. By combining multiple techniques, hybrid approaches can overcome the limitations of individual techniques and provide more personalized and relevant recommendations to users.

Overall, hybrid techniques are an important tool in recommendation systems, allowing for more accurate and diverse recommendations that can better reflect the preferences and interests of individual users.

Figure 3.1 shows visually the types of recommendation systems.

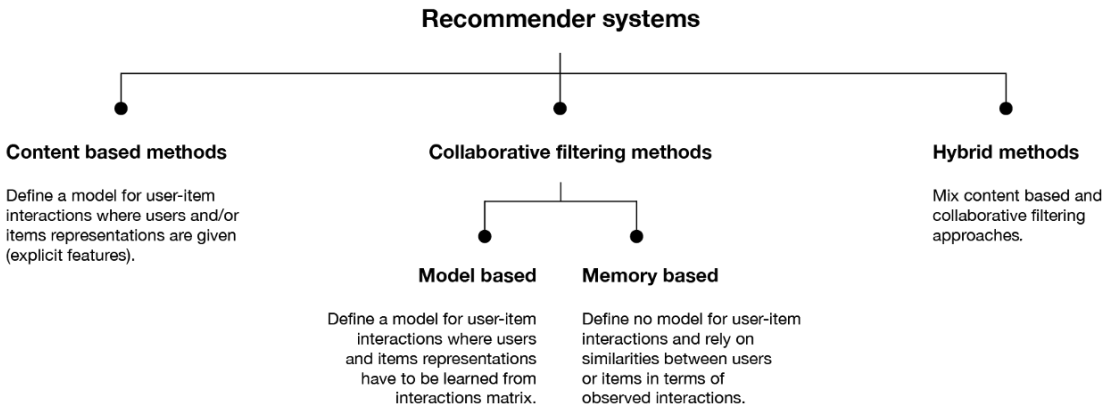

Figure 3.1: Types of recommendation systems [47]

Content-based filtering: This method uses attributes or features of the items themselves to recommend items to users. For example, in a movie recommendation system, the system would recommend movies that have similar attributes to the movies that the user has previously liked or watched. This method is suitable for situations where there is a clear understanding of the features that define an item, and when the user's preferences are well-defined.

Collaborative filtering: To suggest items to users, this approach uses information on interactions between users and items. Based on how users engage with products, the algorithm compares people to find patterns and suggests products that other

users who are similar to them have found appealing. The system would suggest a movie that User A hasn't watched but that User B has loved, for instance, if their taste in movies is similar. When consumers' preferences are not clearly defined and there is a lack of consensus over the characteristics that identify an item, collaborative filtering might be helpful.

Hybrid approaches: These techniques combine collaborative and content-based filtering to give consumers tailored suggestions. Hybrid systems can overcome the drawbacks of individual approaches and give users better recommendations by combining the advantages of both. A hybrid recommendation system might, for instance, employ collaborative filtering to suggest goods that users with similar tastes have already appreciated and content-based filtering to suggest items that are comparable to those the user has liked in the past. When there is an abundance of data accessible and a need for tailored advice, hybrid systems can be helpful.

Figure 3.2 shows the general structure of recommendation system, where there are abstractly defined users, items, and user-item interaction matrix.

Users	User-item interactions matrix	Items
suscribers	rating given by a user to a movie (integer)	movies
readers	time spent by a reader on an article (float)	articles
buyers	product clicked or not when suggested (boolean)	products
...		

Figure 3.2: The general structure of recommendation system [47]

The user-item interaction matrix can be generated by users when they rate objects, or parts of the cells of the matrix can be anticipated using machine learning techniques and algorithms. The changes in the creation of the user-item interaction matrix are depicted in Figure 3.3.

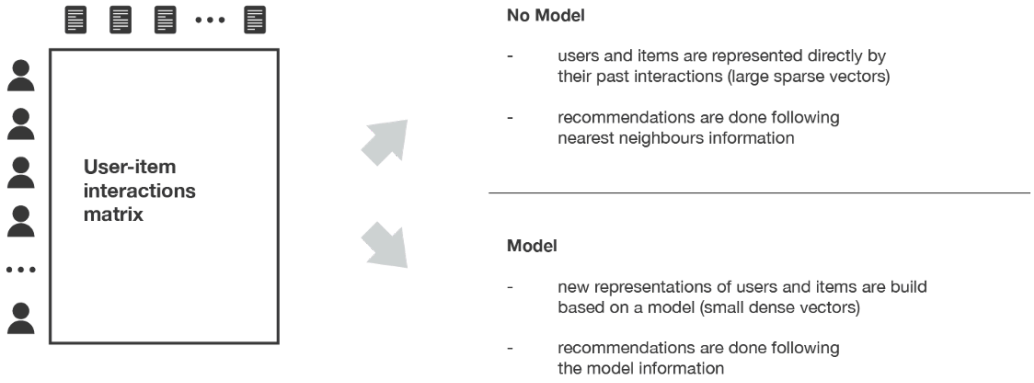

Figure 3.3: Different methods to construct user-item interaction matrix [47]

A user-item interaction matrix is a table that depicts how people and objects interact in a recommendation system. The interactions between users and items are depicted in the table's cells, with each row representing a user and each column representing an item.
To construct a user-item interaction matrix, you will need to follow these steps:

- Identify the users and items: The first step is to identify the users and items in your dataset. For example, if you are building a movie recommendation system, the users could be the viewers, and the items could be the movies.
- Collect user-item interactions: The next step is to collect the user-item interactions. For example, if you are building a movie recommendation system, you would need to collect data on which movies each user has watched, rated, or liked.
- Create the matrix: Once you have collected the user-item interactions, you can create the matrix. The matrix should have the users as rows and the items as columns. The cells in the matrix should represent the interactions between the users and items. For example, if a user has watched or rated a movie, the cell in the corresponding row and column would have a value to represent that interaction.
- Handle missing values: In some cases, there may be missing values in the matrix, which represent situations where the user has not interacted with a particular item. In such cases, you can either set the value to 0 or use an imputation technique to fill in the missing values.

Overall, constructing a user-item interaction matrix is an essential step in building a recommendation system as it provides the foundation for analyzing user-item interactions and making personalized recommendations.

As previously mentioned, recommendation systems provide the user with personalized advice on issues that may interest them. Systems already help people effectively manage content overload and reduce the complexity of finding the right information. A recommendation algorithm requires three components: 1) a database that stores the characteristics of existing elements, 2) profiles that model user interests, 3) recommendation algorithms that contain personalized suggestions for each user. The first strategy created for the recommendation system was content-based filtering, consisting of suggestions of items similar to those that the user previously liked. But, despite its accuracy, this technique is limited due to the similarity indicators used. Metrics are based on rigid syntax approaches, which can only find similarities between elements that have common attributes. Therefore, approaches that are considered traditional and content-based lead to excessive concretization of sentences, including only those elements that have a very strong resemblance to those known to the user [45]. For instance, if a user has accessed a web page associated with the mobile world, "RAM," content-based filtering will recommend to the user elements associated with the world of electronics. Figure 3.4 shows the role of user-item interaction matrix in the recommendation of the most popular items among the k - nearest neighbors.

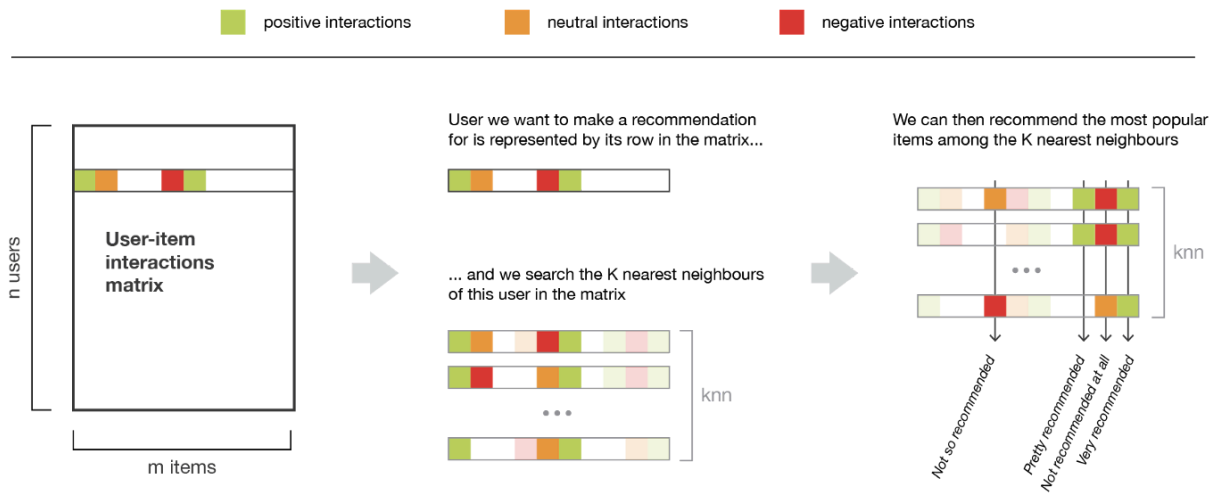

Figure 3.4: the role of user-item interaction matrix in the recommendation of the most popular items among the k - nearest neighbors [47]

Figure 3.5 shows the process of starting from identifying the preferred item of user to the search and recommendation of k-nearest items.

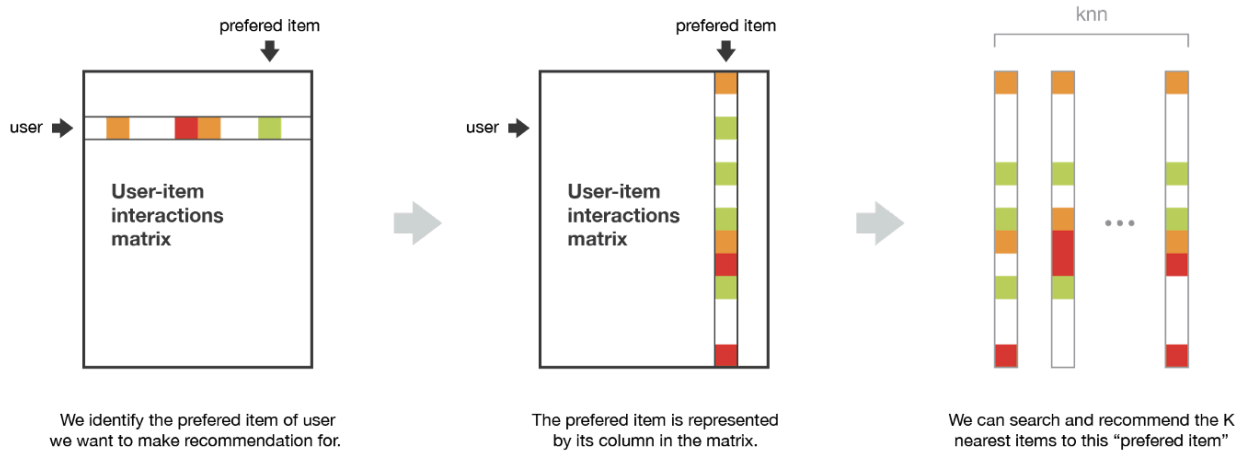

Figure 3.5: the process of starting from identifying the preferred item of user to the search and recommendation of k - nearest items [47].

Many methods have specialized options for classifying learners whose purpose is to teach a function that will predict which class a particular document belongs to. Other algorithms will consider the recommendations as a regressive problem in order to teach the function predicting the numerical value, the rating of the document. There are two subtasks when designing a content-based filtering system. The first is to search for a view of documents, elements. The second is to create a profile that will allow invisible documents, elements to be recommended [43]. A content-based user profile is created using a weighted element vector. The weights weigh the importance of each function and element to the user. Importance can be calculated from individually evaluated content vectors using different skills. Content based filtering includes following steps:

- Reduce number of part attributes for recommendation
- Compare feature attributes with active user settings

Hybrid technique - based on using content-based and collaborative filtering techniques by establishing correctness as a forward aspect [48].

An important advantage of this type of filtering is that content-based filtering gives the user independence through the exclusive ratings used when creating a new profile as an active user. The system also provides transparency to the user by explaining how the recommendation system works. Also, the system adequately recommends items not yet reviewed by the user. It will be very profitable for new users. However, creating attributes for elements is itself a difficult task in a specific area. The system also suffers from excessive specialization due to the fact that it represents products of the same type. Sometimes it is difficult to get feedback from

users to understand whether the recommendation is correct or not, since users in particular do not rank elements.
Collaborative filtering is the most used method for recommending items to users. The system offers recommendations based on similar users or similar user elements.
This type of filtering includes two methods:

- based on the model
- based on memory

The first type of method defines a model in order to explain the interests of users and predict position estimates. The second method first determines similarities between users, and then selects the most similar users as user neighbors in order to make recommendations. The memory-based method provides a significant recommendation. However, computational time increases rapidly with the number of users and elements. In some situations, it is difficult to take action in real time. One of the main problems of collaborative filtering is the time complexity of forming squint groups. Usually, time complexity would be O(k^2) with using k nearest-neighbors, where k is the number of items. The main goal of this method is to calculate pairwise likeness for every user and finding similar users. The alternative method for forming groups is the k-means algorithm. Figure 3.6 depicts how collaborative information and content information can be used to form a model which will serve to recommend essential items to users.

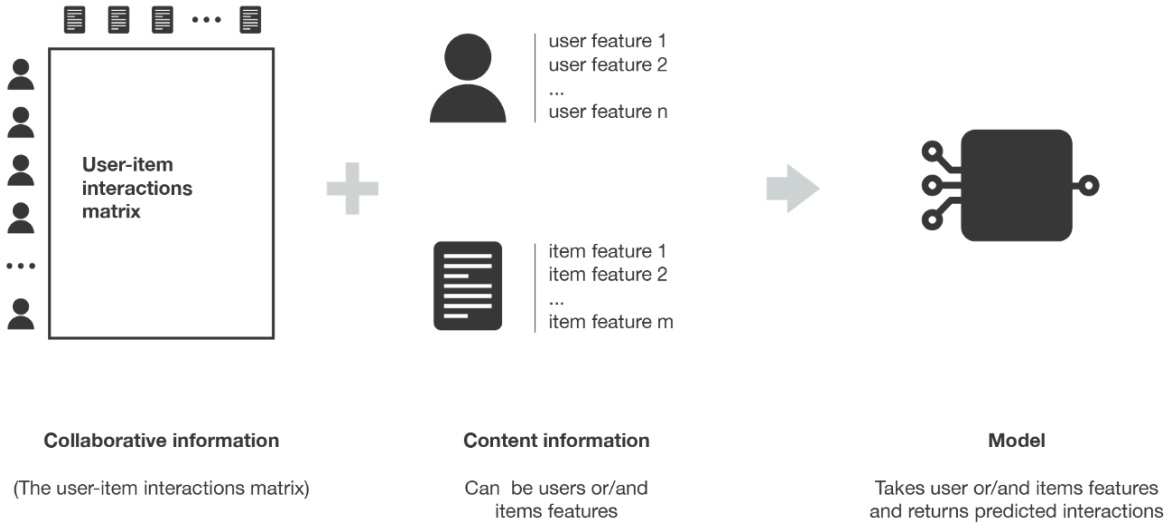

Figure 3.6: Collaborative information and content information combined can form a model [47]

Figure 3.7 shows the difference between user-user and item-item models.

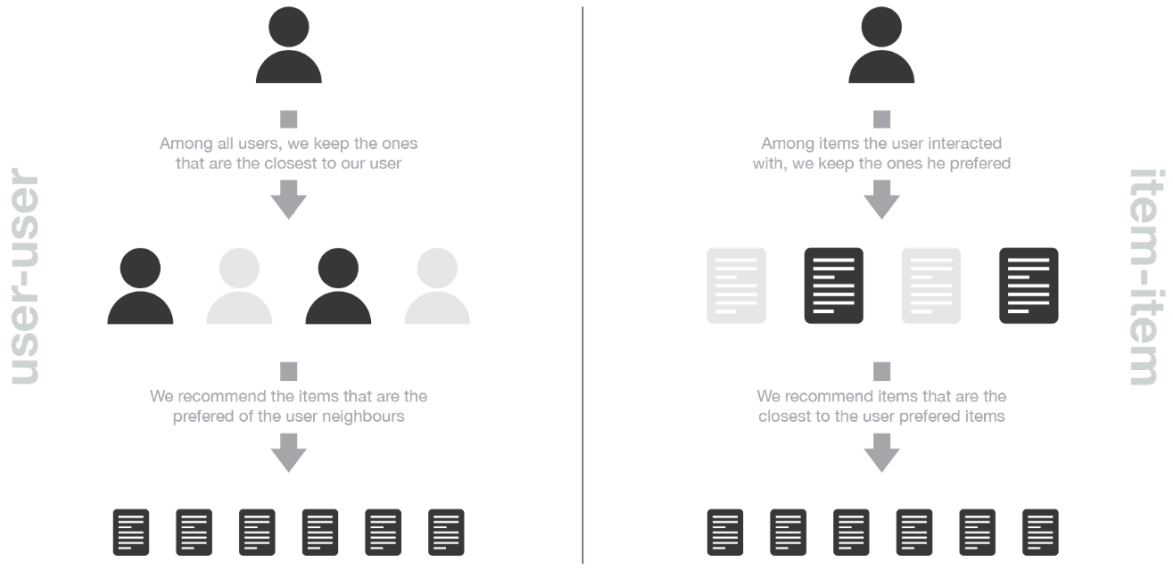

Figure 3.7: the difference between user-user and item-item models [47]

Figure 3.8 describes the process of user-item interaction matrix's formation. It can be seen that the user-item interaction matrix is summation of the reconstructed interactions matrix and reconstruction error matrix.

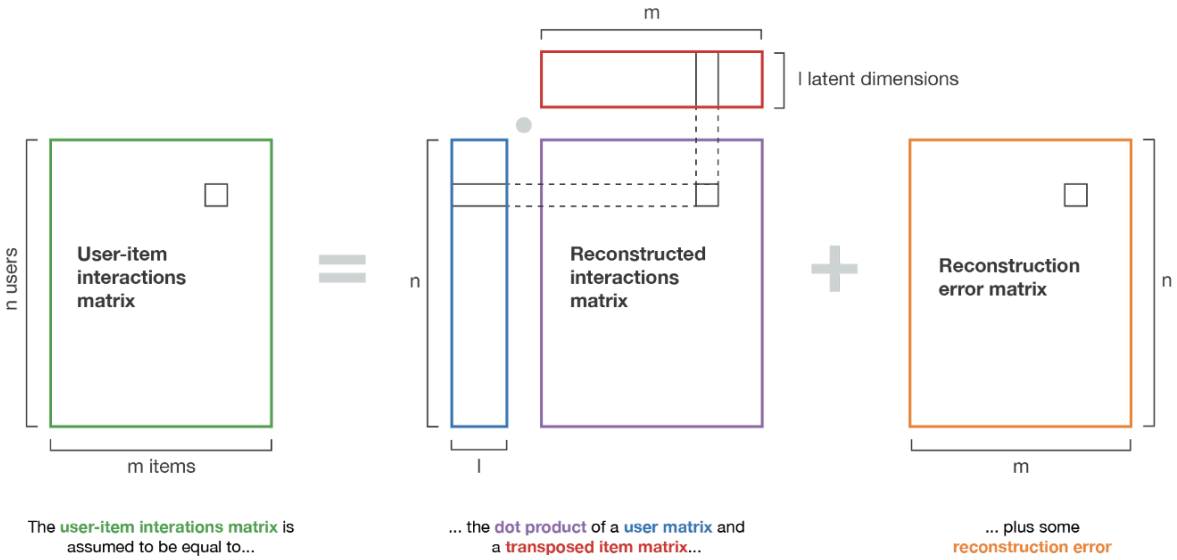

Figure 3.8: the process of user-item interaction matrix's formation [47]

Entity-relation matching.

Entity-relation matching is a type of matching used in information extraction and natural language processing (NLP) to discover connections between items in a text or document. It has a lot of applications [49]—[61]. Entities can be any interesting entity in a particular environment, including people, places, organizations, or other objects. Identifying and classifying the relationships between these entities in a text, including determining which entities are connected to one another and the type of relationship between them, is known as entity-relation matching.

As an illustration, think about the phrase "John works for Apple." The relationship between the two words "John" and "Apple" in this statement is that John works for Apple. Knowing that "John" is an employee and "Apple" is an employer, and that their relationship is that John works for Apple, is known as entity-relation matching.

NLP methods like dependency parsing and named entity recognition are frequently used to match entities. Identification of named entities in a text entails categorizing them into predetermined groups, such as person, group, or location. In dependency parsing, the grammatical relationships between words and entities are determined by examining the sentence's structure.

Once entities have been identified and categorized, entity-relation matching involves analyzing the grammatical structure of a sentence to identify the relationships between entities. This can involve using techniques such as dependency parsing to identify the subject and object of a sentence and the nature of the relationship between them.

The process of Named Entity Recognition typically involves the following steps:

- The input text is separated into discrete tokens or words, which serve as the fundamental building blocks of analysis.
- Each token is given a part-of-speech tag, such as a noun, verb, adjective, etc., known as a part-of-speech (POS) tag. This process aids in determining the grammatical function of each word in a sentence.
- The NER system recognizes and categorizes tokens or sequences of tokens that correspond to identified entities in this stage. To identify the entities, it uses a variety of linguistic patterns, guidelines, and machine learning methods.
- Entity Classification: After being detected, entities are classified into several types or predetermined categories. For instance, in the context of a technology business, a named entity such as "Apple" can be categorized as an organization.

Named entity recognition can be carried out in a variety of ways:

Rule-Based Approach: Named items are identified using established patterns or rules in rule-based NER. These rules, which describe patterns like capitalization, context, surrounding words, or regular expressions, are often hand-crafted by professionals.

Machine Learning Approach: In this method, entities are manually labeled in training data before being fed into a machine learning model for analysis. In order to anticipate entities in unseen text, the model learns patterns and features from the labeled data. Conditional Random Fields (CRF), Hidden Markov Models (HMM), and deep learning-based models like Recurrent Neural Networks (RNN) and Transformer-based architectures are examples of popular machine learning techniques used for NER.

NER is used in many different contexts and domains:

- Extraction of Information: NER is essential in the process of separating structured information from unstructured text. For instance, extracting identified items from a news story, such as the names of persons, companies, or places, might aid in summarizing or organizing the data.
- Answering questions: NER aids in locating pertinent entities in a query and pulling the relevant response from a corpus or knowledge base.
- Named Entity Linking: To enable more thorough and context-sensitive information retrieval, NER can be used in conjunction with Named Entity Linking (NEL) to link detected named entities with their corresponding entities in a knowledge base or database.
- Sentiment analysis: By identifying named items, tasks involving sentiment analysis can be given extra context. This is because feelings toward particular entities can be examined independently.

With the availability of massive labeled datasets, cutting-edge models, and pre-trained language representations like BERT, GPT, and ELMO, NER systems have made considerable strides. These advancements have increased NER's performance and accuracy in a variety of languages and domains, making it an important tool in information extraction processes and NLP pipelines.

Overall, entity relation matching is an important component of NLP and information extraction, as it helps to identify the relationships between entities in a text, which can be useful for tasks such as text summarization, question answering, and knowledge graph construction.

Image matching.

Image matching is a process of comparing and analyzing two or more images to determine their similarity or differences. This process is used in a wide range of applications, such as computer vision, image retrieval, image recognition, object tracking, etc. [62]—[75].

Image matching typically involves two main steps: feature extraction and feature matching. Feature extraction involves identifying and extracting relevant features from images, such as edges, corners, and textures, that can be used to represent the image. Feature matching involves comparing the extracted features between images to determine their similarity or differences.

There are different techniques used for image matching.

Template matching.

In this technique, a template image is matched against a target image to determine whether the template is present in the target image. This technique is useful for object recognition and detection in images.

Scale-invariant feature transform (SIFT).

This technique involves extracting features from images that are invariant to scale and rotation, making it useful for object recognition and image retrieval. David Lowe created the Scale-Invariant Feature Transform (SIFT) computer vision technique in 1999. In applications like image identification, object detection, and picture stitching, it is frequently utilized for feature extraction and matching. The main idea underlying SIFT is to locate and describe distinct, strong, and resilient features in an image that are resistant to affine, rotational, and scale-related transformations. The SIFT algorithm entails the following steps:

Scale-space extrema detection: SIFT builds a scale-space model of the input image first. This is accomplished by applying a variety of Gaussian filters on the image at various scales. The Gaussian filter's standard deviation is gradually increased to form the scale space. To highlight the areas with noticeable intensity variations, the Difference of Gaussians (DoG) is calculated at each scale by subtracting adjacent scales. Next, suitable feature locations in the DoG are determined by identifying the extrema (local maxima and minima).

Keypoint localization: SIFT eliminates unstable keypoints that are incorrectly localized or insufficiently specified in order to guarantee proper keypoint localization. This is achieved by removing keypoints with poor contrast or keypoints near borders and comparing each extrema with its surrounding pixels in the scale space.

SIFT gives each keypoint an orientation in order to make the algorithm resistant to picture rotation. Each keypoint computes a local picture gradient, and the nearby neighborhood creates a histogram of gradient orientations. The keypoint's orientation is chosen to be that which dominates the histogram.

Calculation of the keypoint descriptor: SIFT creates a reliable and condensed representation known as the keypoint descriptor to describe the keypoint's local appearance. The local image gradients and their orientations within a region around the keypoint are used to construct the keypoint descriptor. The descriptor is resistant to changes in light, viewpoint, and partial occlusion since it captures information about the intensity gradients, orientations, and spatial connections.

Keypoint matching: After the descriptors for the keypoints in various photos have been calculated, SIFT uses feature matching to determine the relationships between the keypoints in various images. Typically, distance metrics like Euclidean distance or cosine similarity between the descriptors are used for this. Potential matches are those keypoints that have the closest matches.

Due to its capacity to cope with changes in scale, rotation, and affine transformations, SIFT has demonstrated to be a potent technique for feature extraction and matching. It has been extensively employed in a variety of computer vision applications, such as image stitching, 3D reconstruction, object recognition, and image retrieval. Convolutional neural networks (CNNs), an alternate feature extraction technique, have grown in prominence in recent years due to the introduction of deep learning methodologies.

Speeded Up Robust Features (SURF).

His technique is similar to SIFT but is faster and more robust to changes in scale and rotation. Herbert Bay et al. invented the computer vision technique known as SURF, or Speeded-Up Robust Features, in 2006. SURF is frequently used in image recognition, object detection, and image stitching applications for feature extraction and matching. By using integral images and estimating the generation of image gradients and descriptors, it seeks to offer a reliable and effective alternative to the SIFT technique.

The SURF algorithm entails the following crucial steps:

- Scale-space extrema detection: SURF creates a scale-space representation of the input image, much like SIFT does. It employs a difference-of-Gaussian (DoG) method, which computes the difference between Gaussian-blurred images at various sizes. The image is convolved using an increasing number of Gaussian filters to produce the DoG pyramid. Each pixel in the scale space is compared to its 26 neighbors to identify the extremes in the DoG

pyramid.

- Localization of keypoints: SURF uses a Fast Hessian-based approach to localize keypoints. For each pixel in the scale space, the Hessian matrix (second- order derivatives) must be calculated. Potential keypoints are found using the Hessian matrix's determinant. Then, by comparing the response values with the nearby keypoints, non-maximum suppression is carried out to eliminate unstable keypoints.
- To achieve rotational invariance, SURF gives each keypoint a specific orientation. Haar wavelet responses are used to assess the area surrounding each keypoint because they can be quickly computed from integral images. Based on the sum of the Haar wavelet responses within various orientation bins, the dominant orientation is identified.
- Calculation of the keypoint descriptor: The local intensity distribution surrounding each keypoint is used to compute the keypoint descriptor in SURF. SURF effectively calculates the gradient magnitude and orientation in the horizontal and vertical directions using the idea of Haar wavelet responses and integral pictures. The responses of the Haar wavelet features within a specific area surrounding the keypoint are concatenated to create the keypoint descriptor.
- Keypoint matching: After the descriptors for the keypoints in various pictures have been generated, SURF uses feature matching to determine correspondences. When comparing the descriptors of two keypoints, SURF uses a distance metric like the Euclidean distance or the sum of absolute differences (L1 norm). Potential correspondences are those keypoints that have the closest matches.

SURF is superior to SIFT in a number of ways, including computational effectiveness brought about by the use of integral pictures, quicker feature extraction, and scale and affine transformation resistance. It has been used successfully for a number of computer vision tasks, including image stitching, object detection, and image registration. However, similar to SIFT, SURF has recently been partially supplanted by deep learning-based techniques.

Convolutional Neural Networks (CNNs).

In order to recognize patterns and characteristics in images, a neural network must first be trained. CNNs are employed in a variety of tasks, such as picture segmentation, object detection, and classification.

Convolutional Neural Networks are a particular kind of deep learning algorithm made for processing and analyzing structured grid-like data, like audio spectrograms, movies, and image data. CNNs have shown extraordinary

performance in applications including image classification, object identification, and picture segmentation, revolutionizing a variety of fields, particularly computer vision.

The structure of the visual cortex in the human brain, which consists of neurons that respond to particular areas of the visual field, served as an inspiration for CNNs. Similar to this, CNNs are made up of convolutional layers, which are interconnected layers of artificial neurons that can automatically recognize and extract useful information from input data.

Here is a summary of the main features and procedures in a standard CNN:

- Convolutional Layers: These layers serve as the foundation of CNNs and carry out the essential convolutional function. Convolution is the process of applying a number of tiny filters, commonly referred to as kernels, to the input data, multiplying the filter weights by the local receptive field element by element, and adding the results to create a feature map. At various sizes, the filters record spatial patterns and features.
- ReLU (Rectified Linear Unit) is a common activation function that is used element- by-element after convolution to provide nonlinearity to the network. ReLU activation maintains positive values constant while setting negative values to zero to aid CNNs in modeling complex relationships in the data.
- Pooling Layers: The feature maps acquired from the convolutional layers are downsampled using pooling layers. The feature map is separated into non-overlapping sections using the popular max pooling technique, and only the maximum value in each zone is kept. By decreasing the spatial dimensions of the feature maps through pooling, the network becomes more resilient to changes in the input while also requiring less processing power.
- Fully Connected Layers: After multiple convolutional and pooling layers, the feature maps that are created are vectorized and sent through fully connected layers, which are similar to traditional neural networks. These layers establish a connection between every neuron in one layer and every neuron in the one above. By combining features from several input regions, they develop higher-level representations and perform tasks such as regression or classification.
- Loss Function and Optimization: CNNs are usually taught using supervised learning, where they pick up new information from instances that have been labeled. The difference between the predicted output and the true label is measured by the loss function. The weights of the network are modified using gradient- based optimization algorithms, such as stochastic gradient

descent (SGD) or its derivatives, in a manner that minimizes loss.

The main approach for calculating the gradients of the loss function with respect to the network weights is called backpropagation. The optimization technique then makes use of these gradients to change the weights during training. Up until the network converges to a state where the loss is minimized, the process is repeated iteratively.

The advantage of Convolutional Neural Networks (CNNs) is that they can automatically learn hierarchical feature representations from the incoming data. Deeper layers of the network learn high-level notions like forms and object pieces while early layers collect low-level data like edges and textures. CNNs can model complicated patterns and perform better in tasks like object identification and picture classification due to their hierarchical feature extraction capabilities.

CNNs have developed into a potent tool for a variety of applications beyond computer vision, including natural language processing, speech recognition, and recommendation systems, thanks to developments like transfer learning, where pre-trained CNN models are used as a starting point and fine-tuned on specific tasks.

Due to their capacity to extract useful characteristics from organized grid-like data, convolutional neural networks (CNNs) have been used in a broad variety of industries. The following are some significant applications of CNNs:

- Classification of photos: CNNs are excellent at reliably classifying photos into predetermined categories. Applications span from basic image identification (such as classifying objects, animals, or settings) to more specialized fields like medicalimaging, where CNNs can help with disease diagnosis through picture analysis.
- CNNs are frequently used for object identification tasks, where the objective is to locate and identify items within an image. CNN-based object detection models are able to offer bounding box coordinates and class labels for several items in an image, opening up possibilities for use in autonomous vehicles, security cameras, and facial recognition software.
- CNNs are able to segment images into several areas or objects by giving a name to each pixel. This is helpful in applications that demand a thorough grasp of object boundaries, such as medical imaging for segmenting malignancies, satellite images analysis, and scene interpretation.
- CNNs can be used for video analysis tasks like action recognition, where the objective is to recognize and categorize various actions or activities taking place in a video sequence. CNN-based models have applications in video surveillance, video recommend systems because they can learn

spatiotemporal patterns and offer insights into video content.

- Medical Imaging: CNN analysis of medical imaging has significantly advanced. They can help with illness diagnosis, the detection of anomalies in X-ray, CT, and MRI pictures, and the prediction of patient outcomes based on imaging data.
- CNNs have demonstrated promise in the detection of cancer, the identification of certain diseases, and the support of radiologists' clinical judgment.
- Natural Language Processing (NLP): CNNs have been used for NLP jobs even though their main use case is computer vision. CNNs can process word or character sequences in text classification, sentiment analysis, and text summarization to extract significant features and generate predictions.
- CNNs are essential for autonomous cars to be able to detect and comprehend their surroundings. They can interpret sensor data from cameras or LiDAR scans to identify traffic signs, detect objects, and help with lane recognition, making autonomous driving systems safer and more dependable.
- Robotics: CNNs are used in robotics for a variety of tasks, such as grasping and object recognition. Robots can recognize and locate items in their surroundings using CNN-based models, enabling more sophisticated manipulation and navigation capabilities.
- In the field of biometrics, CNNs are employed in processes including face, fingerprint, and iris recognition. They are able to extract distinguishing characteristics from biometric data and compare them to reference templates, enabling efficient and safe identification and authentication methods.

These are just a few instances illustrating the numerous uses for CNNs in various fields. CNNs are an effective tool for a variety of tasks in computer vision, image analysis, and beyond due to their flexibility, robustness, and capacity to extract meaningful features from structured grid-like data.

Overall, image matching is a crucial component of computer vision and image processing, as it enables the comparison and analysis of images for various applications, such as object recognition, image retrieval, and tracking.

Stable matching.

We can consider elements to be matched as agents. In case if agents provide themselves preference lists, then the problem of matching can be classified according to the following types:

- one-to-one matching [24]—[27]
- one-to-many matching

- many-to-many matching

Below, there are outlined descriptions of each type:

- In one-to-one matching, each element of the first set can be matched only with one element from the other set, and vice versa.
- In one-to-many matching, each element of the first set can be matched more than one element from the other set.
- In one-to-many matching, each element of the first set can be matched more than one element from the other set, and vice versa.

Deferred acceptance algorithm.

The input data for this matching algorithm is a set for each type of agent and data structure that holds their preferences. Stability always holds in this algorithm. Stability in the context of matching algorithms means that the following conditions hold:

There is no element in the first set that prefers some other element in the second set over the element to which the element from the first set is already matched

There is no element in the second set that prefers some other element in the first set over the element to which the element from the second set is already matched

Gale and Shapley come up with an algorithm to match elements of two sets of equal size together holding the conditions of stability. Assume that there are 2 sets: A and B. Then the algorithm works in the following way:

- Every round, each element of set A makes an offer to be matched with the most preferred element in the set B
- Each element of set B evaluates the given offer comparing it with its current candidate in case it has one. In case if the element in set B is not matched or it prefers the offer more than the current matched element from set A, then it accepts the offer. In another case, this element rejects the offer.
- The algorithm iterates repetitively until the condition of stability will not hold. The first stable matching algorithm was proposed by Gale and Shapley in 1962 [76].

The algorithm was designed to solve the problem of assigning students to colleges based on their preferences. Since then, many variations of the algorithm have been proposed to solve different matching problems.

Stable matching algorithms are used to solve the problem of matching two sets of elements. The elements can be people, companies, or any other objects that need to

be matched based on their preferences. The algorithm takes as input the preferences of the elements and produces a matching that is stable. A matching is stable if there are no two elements that prefer each other to their assigned partner.

The Gale-Shapley algorithm is the most well-known stable matching algorithm [76]. Each element makes a proposal to their most desired spouse at the beginning of the algorithm. The most appealing proposal of all the ones received is then chosen by each spouse. If a partner rejects an element, the other partner is proposed to until the elements are matched. When all of the elements match, the algorithm is finished.

In situations where two groups of entities—often referred to as "men" and "women" for historical reasons—need to be matched in accordance with their preferences, the Gale- Shapley method is frequently utilized. It can, however, be applied to any collection of things having preferences.

Here is a description of the Gale-Shapley algorithm in general terms:

- Initialization: Each entity suggests the alternative from the other set that it finds most appealing. For instance, each guy makes a proposal to the woman who is at the top of his list, and each woman initially maintains track of the offers she has received.
- Acceptance and rejection: After reviewing the proposals, each woman chooses to reject all but one of the potential partners. The rejected guys then revise their offers by advancing them to their next ideal candidate who has not yet rejected them.
- Iteration: Up until a stable match is made, steps 2 and 3 are repeated repeatedly. Men who have been rejected by women make proposals to their next most favored possibilities while women evaluate their proposals and approve or reject potential suitors at each iteration.
- When no new proposals are presented, the algorithm comes to an end. Now that a stable pairing has been established, each entity is paired with a partner that they can both agree on.

The central tenet of the Gale-Shapley algorithm is that women can accept or reject suggestions made by males based on their preferences. The algorithm assures that each woman receives her most desired spouse among those who propose to her, given their relative positions, by allowing women to have this authority.

The Gale-Shapley algorithm’s ability to ensure the creation of a stable matching is among its most crucial features. If no two entities would both choose the other over their current partners, a pairing is regarded as stable. In other words, there aren’t any "unstable" partnerships in which one party would have a reason to break

off from the other and form a new pair.

Additionally, the Gale-Shapley algorithm has the advantageous quality of being "man-optimal." According to the proposing entities' (in this case, the men's) perspective, the final pairing represents the greatest possible match they could make based on their preferences and the responses of the other entities.

Overall, the Gale-Shapley algorithm offers a methodical and effective strategy for resolving the stable matching problem. It has been used in a number of fields, such as college admissions, job placement, and matchmaking services. It is a widely used algorithm in the field of matching theory because of its fairness, simplicity, and guarantee of stability.

Applications for stable matching algorithms can be found in many industries. These algorithms are employed in economics to match medical residents with hospitals, students with institutions, and employees with businesses. Stable matching algorithms are employed in online dating and job marketplace matchmaking systems in computer science. These algorithms are used in social sciences to research labor marketplaces and marital patterns.

In numerous disciplines, stable matching algorithms have been employed for a long time to address matching issues. The most well-known algorithm is the Gale-Shapley one, which has been used to solve numerous problems in the real world. To create reliable matching algorithms that are more precise and efficient, more study is required.

In real-world circumstances where matching or pairing entities based on preferences are necessary, stable matching algorithms have several uses. The following are some significant applications:

- Job placement: Processes for placing employees, such resident matching programs for medical students (like the National Resident Matching Program in the US), and use stable matching algorithms. According to their compatibility and preferences, the algorithms assist in matching medical students with hospitals. In other areas, matching job seekers with potential employers also involves algorithms.
- Stable matching algorithms can be used in college applications to match individuals with institutions or universities based on their choices and entrance requirements. This helps to minimize disappointment and keep the admissions process stable by ensuring that both students and educational institutions receive an equitable distribution.
- Stable matching algorithms can be used to pair people according to their preferences, such as lifestyle, sleeping patterns, and interests, when

assigning roommates in shared housing situations or dorms. This promotes harmony and lessens hostility between housemates.

- Stable matching algorithms are used in the context of organ transplantation to match available organs with patients in need of transplants. The algorithms calculate the best distribution of organs, optimizing the overall success probability of transplants, taking into account elements including medical compatibility, urgency, and patient preferences.
- School Choice Programs: To match students with their selected schools, school choice programs use stable matching algorithms. In order to provide an equitable and effective distribution of students to schools, these programs take into account both student preferences and admissions regulations, avoiding instances in which kids are left without any viable school options.
- Online dating platforms: To match people based on their likes, hobbies, and compatibility, online dating platforms can utilize stable matching algorithms. The algorithms seek to produce matches that are mutually satisfactory and raise the likelihood that a long-term relationship will succeed by taking into account the interests of both persons.
- Ride-sharing Services: The matching of drivers and passengers in ride-sharing services can be optimized using stable matching algorithms. The algorithms try to make effective and satisfying matches by taking into account the location, distance, and preferences of both drivers and passengers, resulting in a more pleasant ride- sharing experience.

3.2 K-means algorithm

K-means is a flexible and simple unsupervised classification technique, in other words it is a type of cluster algorithm [46] on the basis of dividing. This algorithm was suggested by J.B.MacQueen in 1967 [48] and can be engaged in binarization [77]. Binarization in particular is used to recognize handwritten text. This method allows you to more accurately archive recognized text for a printed document, and more accurate and fast binarization, as well as segmentation methods, allows you to obtain high accuracy. In these cases, the k-means algorithm is used [78] [79]. The main intention of this algorithm is to divide the data set into different clusters so that the objects in the cluster are similar to each other, and the objects in different clusters differ according to predetermined criteria [80]. And commonly used in pattern acceptance, mining of data and image recognition [46]. Also, there are some problems which can be or solved by using k-means algorithm in different types (with evaluation, modified version, improved version and so on):

- Internet of acoustic environment characteristics

- Clustering analysis
- Discriminative subspace clustering
- Global binarization methods
- Image Recognition
- Geyser’s Eruptions Segmentation [77][80][81][82] and so on.

The main elements on which the algorithm is based are index, square-error and error criterion. The fundamental k-means algorithm served as the foundation for numerous further techniques. The medoid, which is the object nearest to the center, and the median were substituted for the elements in the basic k-means in k-medoids and k-median [80]. These algorithmic forms are variations of the k-means algorithm [83]. Many research used a cosine metric, often known as spherical k-means, in place of Euclidean distance to handle text data problems [80]. Three categories can be used to categorize algorithms that might be classified as k-means algorithms: weighted matrix; vector; various features have varying degrees of bias in actual requests.

However, because there are no weight algorithms of type k-means that treat all characteristics equally in the process of decreasing cluster dispersions, several types of feature selection and weighting approaches have been proposed in numerous clustering procedures [80]. Assume we have a set of data that needs to be divided into K separate clusters. The k-means clustering algorithm, which is based on iterative techniques, can be used to tackle such a problem. The algorithm is divided into two major phases. We start by randomly initializing k cluster centroids. The centroid is the cluster's center point. The first step is to allocate each data point to the closest cluster by computing the distance between data points and centroids.

Second step: recalculate centroids of those clusters. And those steps we repeat until we reach some converged point. At the beginning it was mentioned “nearest” to the cluster. This implies some sort of distance measure. Usually the algorithm used Euclidean distance, but the algorithm will work slowly for very large datasets. Performance of your implementation will depend on the distance measure you choose. Let’s look at the example with using k-means in image. In image processing, the technique divides picture pixels into a specific number of clusters, with each cluster represented by a centroid [84]. Figure 3.9 depicts structures before and after k-means execution. As can be seen, in the first portion, all points are dispersed randomly with no clusters, however after clusterisation, data points are distributed in three clusters.

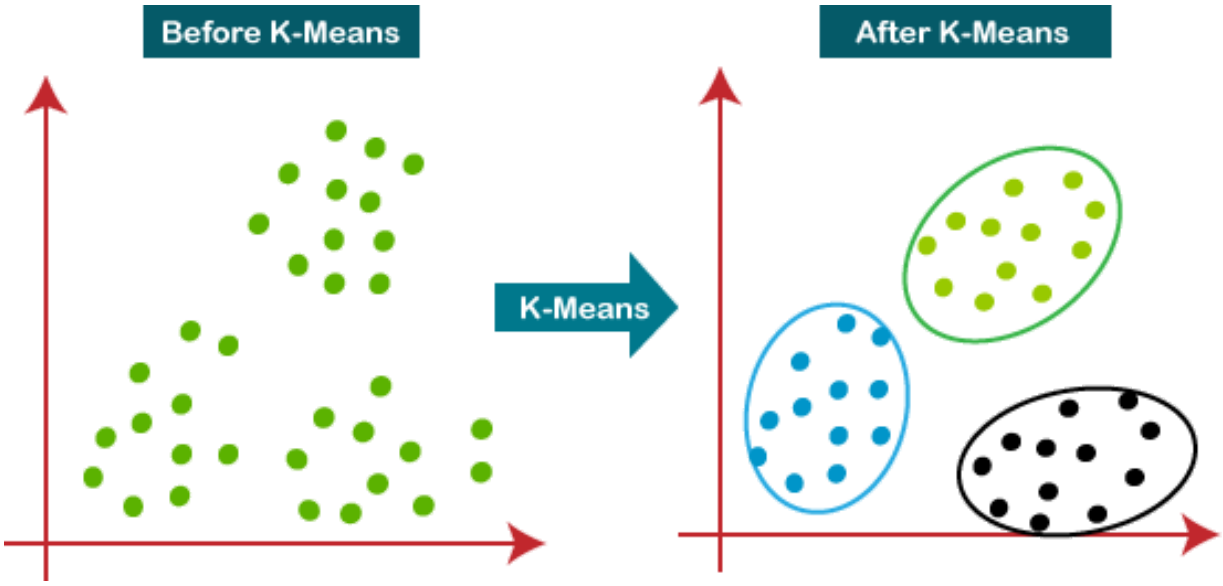

Figure 3.9: Process flow of K-Means [85]

Figure 3.10 illustrates the first step of the execution of k-means algorithm. Initially the data points are just set or accepted as input data.

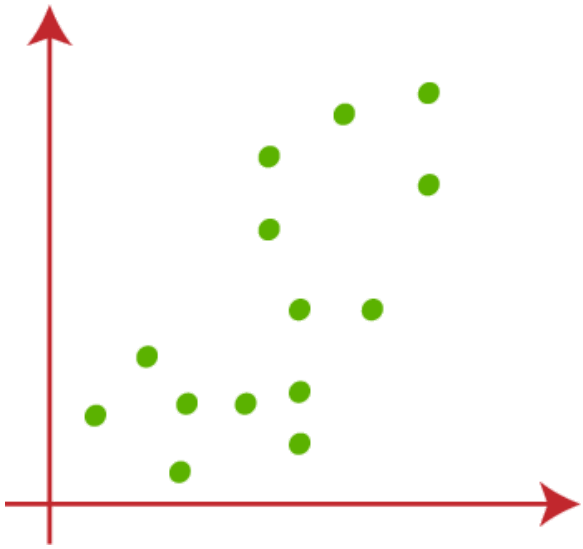

Figure 3.10: Process flow of K-Means [85]

Figure 3.11 shows the second step of the execution of k-means. In this case, k

centroids are to be found which will be used for further calculations and clusterization at the end.

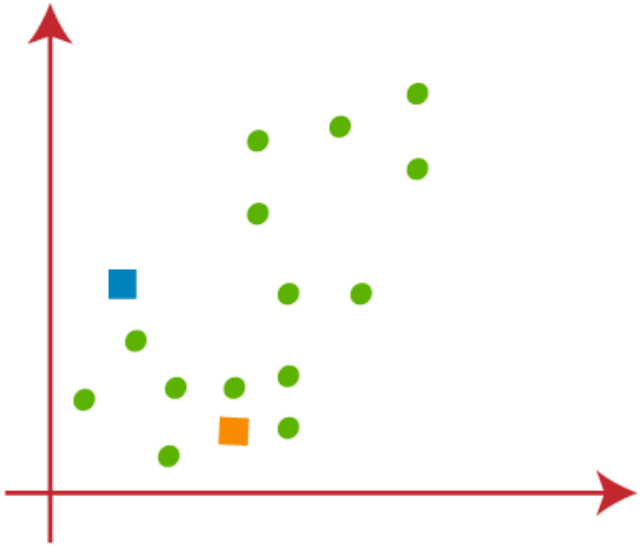

Figure 3.11: Process flow of K-Means [85]

Figure 3.12 illustrates the next step of the execution of k-means algorithm. Here we can observe that two centroids have been set. Our next step involves identifying the nearest k-point or centroid for each data point in the scatter plot. This can be achieved through mathematical calculations that determine the distance between two points, a concept we have previously studied. Once the nearest centroid is identified, we will draw a median between the two centroids. An example of this step is illustrated in the image.

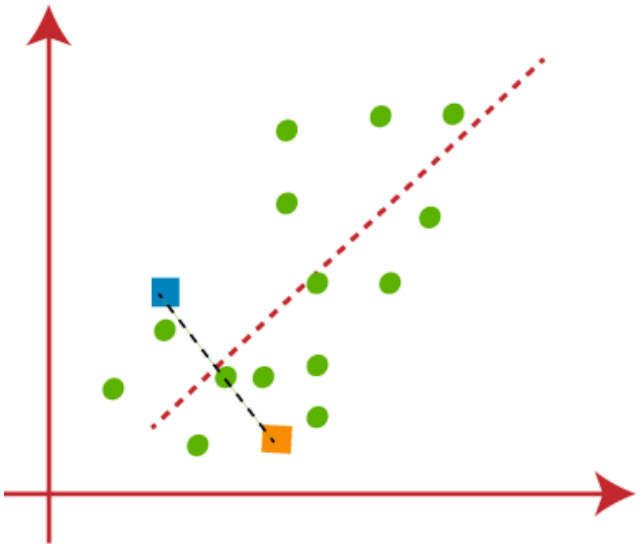

Figure 3.12: Process flow of K-Means [85]

Figure 3.13 illustrates the next step of the execution of k-means algorithm. Our next step involves assigning each data point in the scatter plot to the nearest k-point or centroid. To do this, we will use mathematical techniques that we have learned to determine the distance between two points. Once we have identified the closest centroid for each point, we will draw a line connecting the two centroids. Refer to the image for reference.

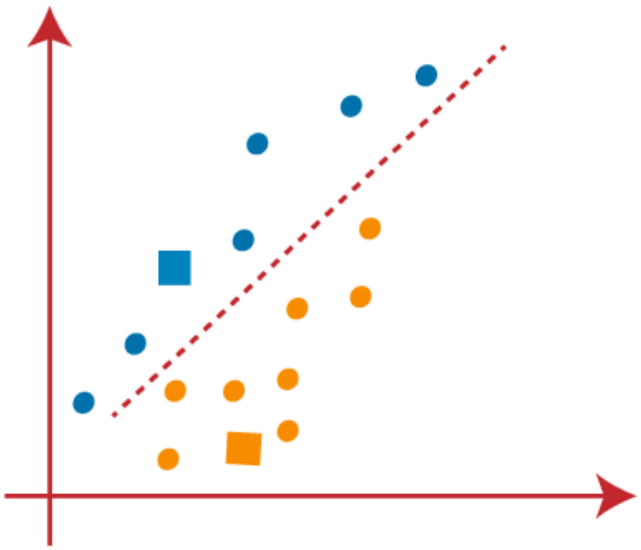

Figure 3.13: Process flow of K-Means [85]

Figure 3.14 depicts the following step in the k-means algorithm execution. Because our goal is to find the cluster that is closest to each data point, we will repeat the procedure by choosing a new centroid. To find the new centroids, we will compute the centers of gravity for the old centroids and utilize this information to find the new centroids, as shown.

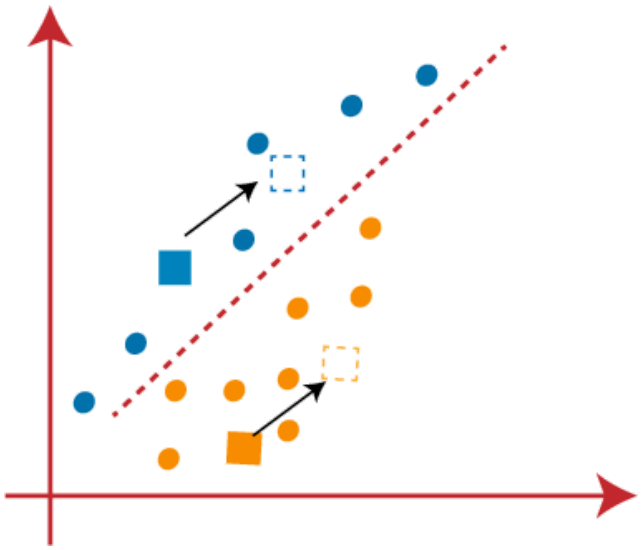

Figure 3.14: Process flow of K-Means [85]

Afterward, we will reallocate every data point to the updated centroid. To achieve this, we will iterate the same procedure of determining a median line.

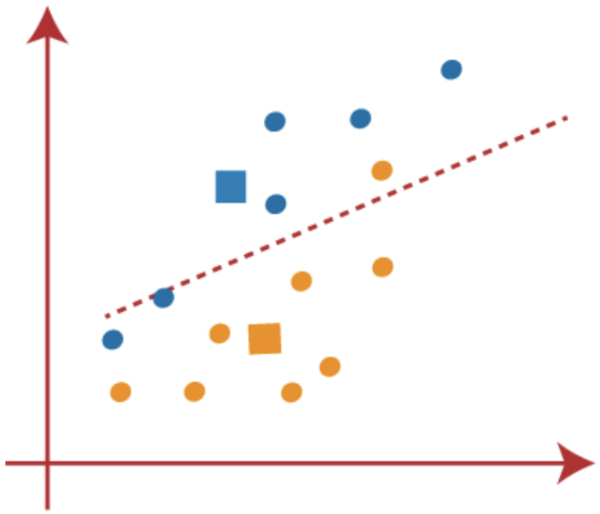

Figure 3.15: Process flow of K-Means [85]

A yellow point is located on the left side of the line, whereas two blue points are located on the right side of the line, as seen in Figure 3.15. As a result, as shown in Figure 3.16, these three points will be assigned to new centroids.

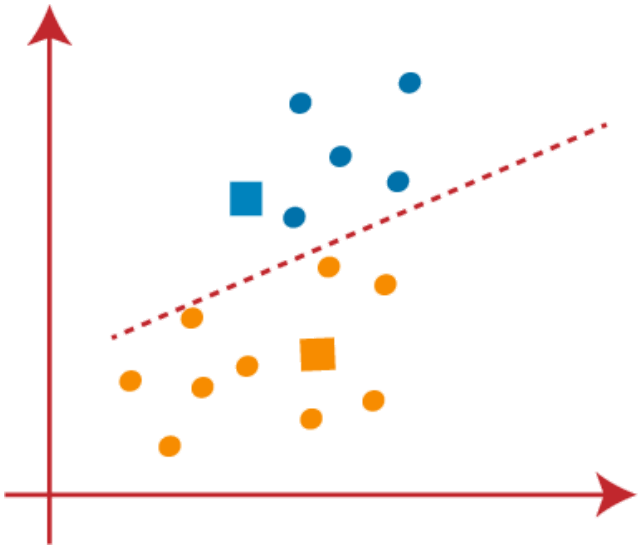

Figure 3.16: Process flow of K-Means [85]

The process will be repeated to locate the center of gravity of centroids, resulting in new centroids depicted in Figure 3.17.

Figure 3.17: Process flow of K-Means [85]

After obtaining the new centroids, we will draw a new median line and reassign the data points accordingly, resulting in an update as can be seen in Figure 3.18.

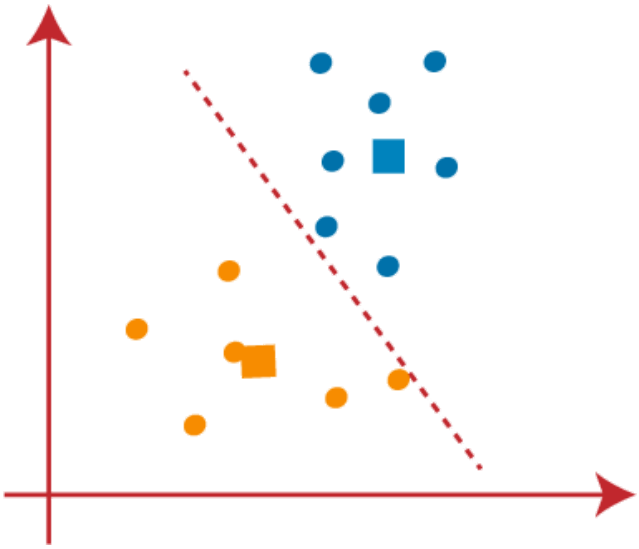

Figure 3.18: Process flow of K-Means [85]

Looking at Figure 3.19, we can observe that there are no disparate data points present on either side of the line, indicating that our model has been established. So there are no distinct or different data points located on one side or the other of the line, which suggests that our model has been successfully established.

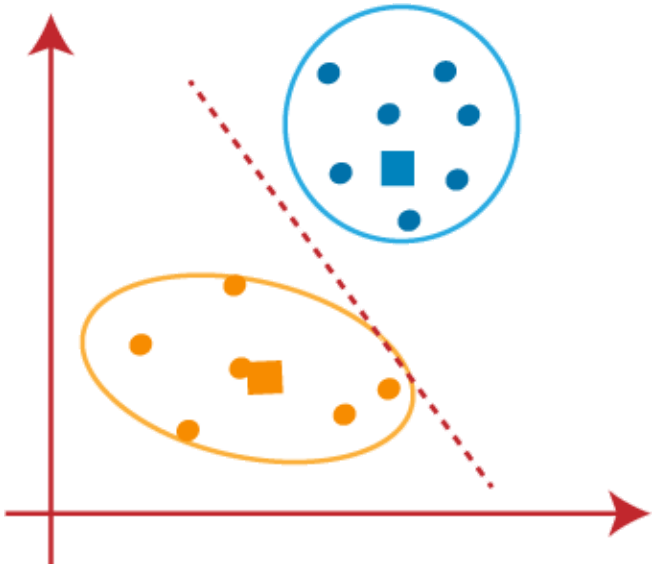

Figure 3.19: Process flow of K-Means [85]

Now that our model is prepared, we can eliminate the supposed centroids, and the two clusters that remain will appear as depicted in the Figure 3.20.

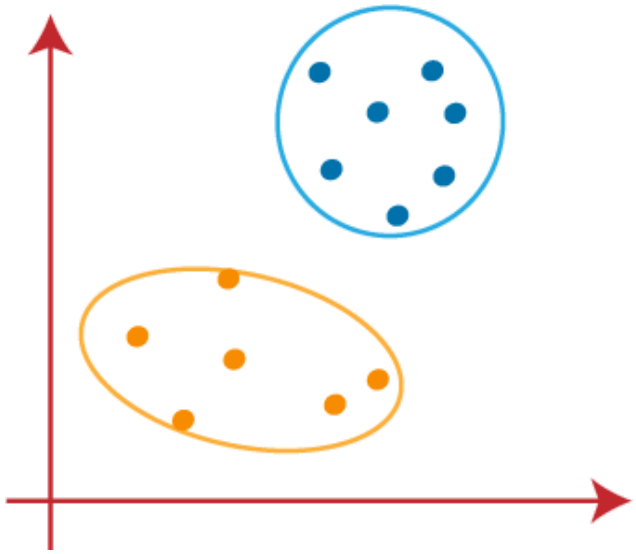

Figure 3.20: Process flow of K-Means [85]

Improved K-means algorithm.

To establish the initial focal points in clustering, a hybrid approach can be employed by combining the algorithms of the greatest minimum distance and traditional k-means. This combined approach aims to leverage the strengths of both methods and address the limitations inherent in each.

The traditional k-means algorithm is known for its simplicity and effectiveness in partitioning data into clusters based on the minimization of intra-cluster variance. However, its performance can be sensitive to the initial selection of cluster centroids.

The algorithm of the greatest minimum distance, on the other hand, focuses on determining the optimal distance between data points, aiding in the identification of suitable cluster centers.

The combined strategy tries to improve the determination of initial focal points by integrating these two algorithms, thereby improving the clustering process. During the selection process, the goal is to alleviate the obstacles associated with a high number of prospective focal areas.

This improved algorithm, which is the result of combining the greatest minimum distance and traditional k-means, is intended to provide a more robust and efficient solution to clustering problems, contributing to better overall cluster quality and addressing issues related to focal point selection [86].

Gray K-means algorithm.

There are numerous K-means clustering techniques available nowadays. For example, the Gray K-means method [87] is based on gray relational analysis. We have phrases like the balanced closeness degree of the i-th comparison sequence in gray relational analysis. The main distinction between Gray K-means and K-means clustering algorithms is that distance in Gray K-means is calculated using a balanced closeness degree. We should define two terms: gray relational degree as denoted by and entropy relational degree as denoted by E. For example, we have a gray relational factor set called X, and the first member of X is used as a reference sequence. It is used as a comparison sequence [88].

By using a balanced closeness degree for calculating distance Gray K-means algorithm takes both the closeness of the points and the global indiscriminate [89].

Weighted K-means algorithm.

Weighted K-means algorithm with the main idea that data points with the bigger weight can attract centroids to them. Distinction with the original one is that you calculate centroid based on weights of data points. According to the article of Joshua Zhexue Huang to divide data into k clusters then minimize the objective function [90]:

Let's add to the function P the new variable W, which will be weights for variables. To improve the algorithm so that noisy environments and large numbers of different sample sized clusters don't affect the result, we can replace Euclidean distance with the exponential distance [91]. Consequently k-means is a variety algorithm with different types and methods of implementation. Also, we can infer that the k-means algorithm is easy to implement and every cluster is declared by centroids or focal points [92-93]. The improved algorithm enhances the efficiency and convergence rate of the cluster focal point. In terms of cluster accuracy and stability, the improved algorithm is better than the basic one. Benefits are noticeable for large scale problems [94].

4. Supervisorship System Experiments

This chapter presents experiments related to the supervisorship system. It includes four experiments focused on matching students to supervisors using various metrics, such as workload balance, preference satisfaction, compatibility score, and more. There was a conducted multidimensional comparison of these algorithms and discussions. The algorithms used in these experiments are One-to-Many Gale-Shapley, K-Means, Collaborative Filtering, and Genetic Algorithm.
Figure 4.1 shows the general outline of steps undertaken in the methodology process. It depicts the usage and connnections of datasets and algorithms in the implementation of objectives of the work.
General approach for the conduction of the above-mentioned experiments to match students to supervisors is the following:

- Used 8 criteria of student-supervisor relationship where each value is from 1 to 5 from Dataset 1 for 4 algorithms and Dataset 2 (1403 records) for collaborative filtering algorithm;
- Summed up all criteria's which denote the perception value of student/supervisor (Q);
- Standardized Q value by scaling to unit variance;
- Executed 4 algorithms (one-to-many Gale Shapley, K-Means, Collaborative Filtering, genetic algorithm) and measured results in terms of metrics: preference satisfaction, time complexity, space complexity, balance of workload, minimum workload, maximum workload, compatibility score.

In addition to the above-mentioned experiments, there was a performed additional experiment, including an association analysis of student-supervisor perceptions and a time series linear regression-based model that predicts student productivity based on past data after using the built supervisorship service.

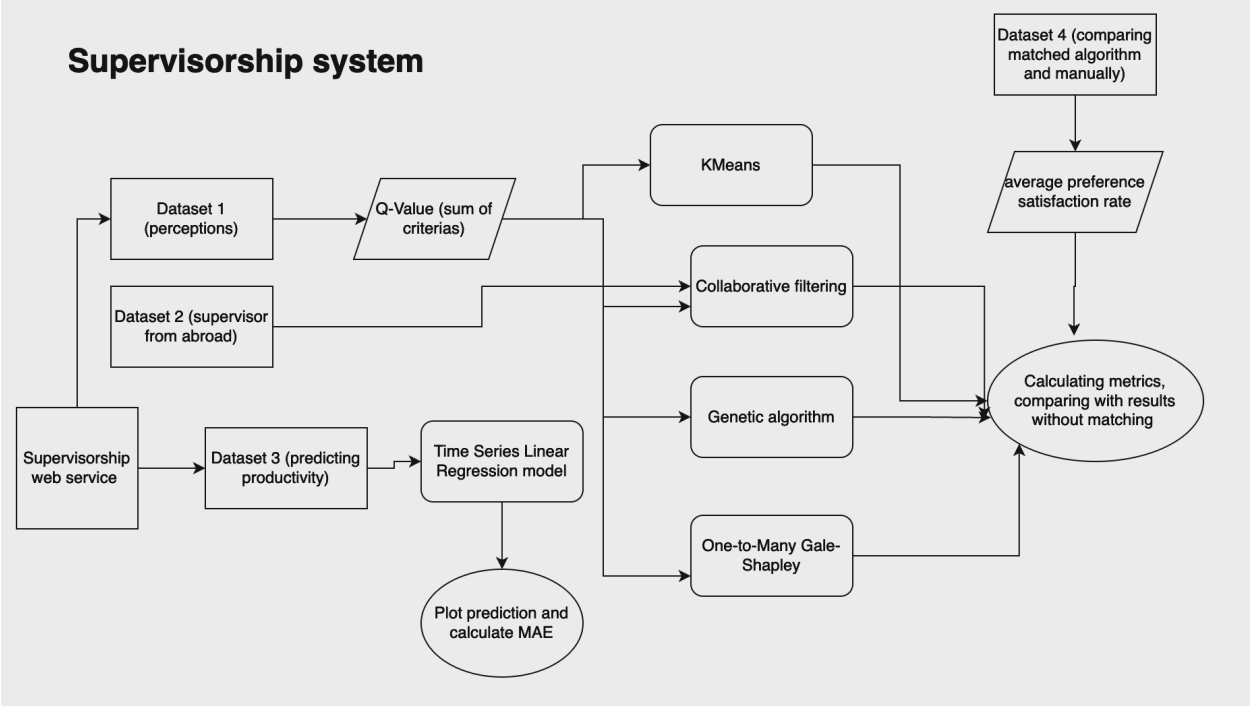

Figure 4.1: Supervisorship system

4.1 Datasets

4.1.1 Dataset 1: Perceptions of Student-Supervisor Relationship

This dataset focuses on exploring perceptions of a proper student-supervisor relationship. The data was collected through a questionnaire consisting of both general and specific questions related to the person and their perceptions of the student- supervisor relationship. The general questions include whether the person is a student or supervisor and the university.

The particular questions are given as statements and pertain to opinions about what constitutes a good student-supervisor relationship. On a scale of 1 to 5, respondents were asked to indicate how much they agreed or disagreed with these statements. The statements outline the supervisor's duties, which include choosing a research topic, choosing a suitable theoretical framework and methodology, creating a suitable program and schedule for the student's research and study, upholding a strictly professional relationship, insisting on frequent meetings, making sure the student is working consistently and on task, demanding to see all drafts of the work, and, if needed, helping with the thesis writing and making sure the presentation is faultless. It is thought of these variables as discrete quantitative variables. The questions for defining perceptions were used from [95]. However,

this perceptions do not imply that student will not follow individual plan signed by scientific supervisor and the general rules will not be violated.
Overall, 130 records were collected, where 110 are from undergraduate students and 20 records are from supervisors. 55
The matching based on perceptions, were done on matching students to supervisors sharing a set of common research areas.

4.1.2 Dataset 2: For recommendation system

An open-source dataset [96] was used for this dataset. It consists of 1403 records, where each record corresponds to a supervisor and their associated characteristics such as university, faculty, panel, and expertise.
Table 4.1 shows a list of universities from which data was collected.

Table 4.1: List of Universities in Dataset 2

University	Count
Aix-Marseille Universit´e	305
Sapienza Universit`a di Roma	258
Universidad Aut´onoma de Madrid	180
University of Glasgow	167
Universit´e libre de Bruxelles	167
National and Kapodistrian University of Athens	129
Stockholms universitet	97
Eberhard Karls Universita¨t Tu¨bingen	74
University of Bucharest	26

4.1.3 Dataset 3: To Predict Productivity of Students

Real-time data gathered from students over 5 weeks was used to predict the productivity of students according to their last progress by using a time series model with linear regression. Each value represented productivity for a day, where productivity is 100 if the student spent the specified number of hours on a task or more. The data is represented as shown in Table 4.2.

Week 1	**Week 2**	**Week 3**	**Week 4**	**Week 5**
100	75	100	50	100
0	50	25	50	75
100	25	100	40	50

0	75	100	40	50
0	25	100	40	50
0	50	100	40	50
0	25	25	40	50
50	25	50	40	50
60	25	40	40	50
40	25	40	40	50
100	25	100	40	50

Table 4.2: Productivity Data

4.1.4 Dataset 4: Comparing preference satisfaction with students without the matching algorithm

To compare preference satisfaction of students which were matched by the use of the algorithm, and those without, there was collected data from students which were matched without algorithms.

The questions consisted of the following ones:

1. To what extent do you believe you and your supervisor share the same opinion regarding the responsibility for selecting a research topic? (1 to 5)
2. How much responsibility do you think you and your supervisor should have in deciding the most appropriate theoretical framework and/or methodology for your research? (1 to 5)
3. On a scale of 1 to 5, how responsible do you feel you and your supervisor are for developing an appropriate program and timetable for your research and study?
4. In your opinion, to what extent do you and your supervisor view the student-supervisor relationship as purely professional, with no room for personal relationships to develop? (1 to 5)
5. Rate the importance, on a scale of 1 to 5, of having regular scheduled meetings between you and your supervisor.
6. How crucial do you think it is to regularly check that you are working consistently and staying on task, on a scale from 1 to 5?
7. On a scale of 1 to 5, how much do you agree that there should be meetings to review all drafts of your work to ensure you are on the right track?

8. To what extent do you believe regular meetings are necessary to assist in the writing of your thesis and ensure that the presentation is flawless? (1 to 5)

4.2 One-to-Many Gale-Shapley approach

4.2.1 Explanation and Used Dataset

For this experiment, there was utilized Dataset 1, which is related to perceptions of student- supervisor workflows. The algorithm used is an implementation of the "One-to-Many Gale-Shapley" algorithm, a variant of the classic Gale-Shapley algorithm used to solve the stable marriage problem. In this context, it focuses on matching students to supervisors based on their preferences. The algorithm works as follows:

- Initialization: Two dictionaries are used: student_to_supervisors to store current assignments and supervisor_to_students to store assigned students. Additionally, a list called free_students contains all students initially.
- Main Loop: The algorithm enters a while loop that continues while there are free students. Inside the loop, a student is selected from the free_students list, usually the first one. The student's preference list, student_pref_list, is obtained from the students_preferences dictionary.
- Processing Student Preferences: The algorithm iterates through the student's preference list, which contains pairs of (supervisor, student_pref_score).
- Checking Supervisor Capacity: For each supervisor in the student's preference list, the algorithm checks whether the supervisor can accept more students. This is determined by comparing the number of students currently assigned to the supervisor with a predetermined limit, often based on the ratio of students and supervisors.
- Assignment Decision: If the supervisor has room for the student, the algorithm proceeds to check whether the supervisor prefers the current student over any of their current students. It iterates through the supervisor's current students, comparing their preferences using the preferences of supervisors dictionary. If the supervisor prefers the new student (prefers_new_student is True) and there is a current student to replace (current_student is not none), it removes the current student from the supervisor's list and makes the current student free again.
- Final Assignment: Finally, the new student is added to the supervisor's list, signifying their assignment to that supervisor. The assignment is updated in the student_to_supervisors dictionary. The loop continues until there are no more free students.

- Return: The function returns the student_to_supervisors dictionary, which contains the final assignments of students to supervisors.

This algorithm effectively matches students to supervisors based on their preferences while ensuring that supervisors do not exceed their capacity, resulting in a stable assignment. It adapts the Gale-Shapley algorithm to solve this one-to-many matching problem.

4.2.2 Results

Table 4.3 shows the experimental results related to the application of the Gale-Shapley algorithm in matching students to supervisors. According to experimental results, the obtained values indicate relatively good results, which include relatively high preference satisfaction (0.74) and relatively good time and space complexities.

Metric	Value
Preference Satisfaction	0.74
Balance of Workload	0.5
Time Complexity	O(num_students × num_supervisors)
Space Complexity	O(num_students + num_supervisors)
Maximum Workload	6
Minimum Workload	3
Compatibility Score	0.46

Table 4.3: Summary of Metrics of One-to-Many Gale Shapley

4.3 Collaborative Filtering Approach

4.3.1 Explanation and Used Dataset

For these experiments, we used Dataset 1 related to perceptions of student-supervisor workflows and Dataset 2.

Explanation of Experiment on Dataset 1

- Initialization:
    - User-item matrix: Each row corresponds to a student, and each column corresponds to a supervisor. The values in the matrix indicate the strength of preference.
    - Similarity matrix: The rows correspond to students, and the columns correspond to supervisors. The values in this matrix represent how similar each student is to each supervisor.

- For each student, the algorithm calculates a set of similarity scores by taking the dot product between the similarity matrix and the student's preferences. These scores represent how well each supervisor matches the student's preferences.
- The supervisor with the highest similarity score is selected as the best match for the student.

Explanation of Experiment on Dataset 2

- Loads a dataset of supervisors' expertise from a CSV file.
- Utilizes TF-IDF vectorization to convert text expertise into numerical vectors.
- Defines student interests and calculates cosine similarities between student interests and supervisor expertise.
- Specifies student and supervisor preferences.
- Generates recommendations for students based on cosine similarity and calculates Preference Satisfaction (PS) for each recommendation.
- Based on student preferences, this approach produced results for a given student.

4.3.2 Results

Table 4.4 shows the experimental results related to the application of collaborative filtering in matching students to supervisors. According to the experimental results, the obtained values indicate poor balance of workload but relatively good preference satisfaction.

Metric	**Value**
Preference Satisfaction	0.8
Balance of Workload	0
Time Complexity	$O(\text{num\_students}^2 \cdot \text{num\_supervisors})$
Space Complexity	$O(\text{num\_students} \times \text{num\_supervisors})$
Maximum Workload	61
Minimum Workload	0
Compatibility Score	0.24

Table 4.4: Summary of Metrics of Collaborative Filtering

Experiment on Dataset 2.

Considering that Dataset 2 has data only related to supervisors, it was used only as an additional experiment in the context of the algorithm of collaborative filtering.

Below there is an example of list of recommended supervisors for a student given his preferences.

- PROTIERE C
- Salvatore D'Amore
- Sandrine Detandt
- Luisa Santos Pastor
- Noret, Joel

4.4 Genetic algorithm approach

4.4.1 Explanation and Used Dataset

For this experiment, we used Dataset 1 related to perceptions of student-supervisor workflows. This approach uses a genetic algorithm to optimize student-supervisor assignments. It iteratively improves assignments over generations by selecting the best solutions, creating new assignments through genetic operations, and selecting the best assignment based on fitness. The final assignment is applied to student data.

4.4.2 Results

Table 4.5 shows the experimental results related to the application of the genetic algorithmic approach in matching students to supervisors. According to the experimental results, the obtained values indicate high preference satisfaction but also complex time and space complexities, and a low compatibility score.

Metric	**Value**
Preference Satisfaction	0.91
Balance of Workload	0.16
Time Complexity	O(num_generations · population_size · num_students)
Space Complexity	O(population_size · num_students)
Maximum Workload	12
Minimum Workload	2
Compatibility Score	0.04

Table 4.5: Summary of Metrics of Genetic Algorithm Approach

4.4 K-Means approach

4.4.1 Explanation and Used Dataset

For this experiment, we utilized Dataset 1, which is related to perceptions of student- supervisor workflows. The K-Means clustering approach involves the following actions:

- Separation of Data: Based on a specific column value, the dataset is divided into two groups: students and supervisors.
- Cluster Identification: The algorithm determines the number of clusters (groups) into which it wishes to classify supervisors and students.
- Student K-Means Clustering: Students are grouped using the K-Means clustering technique, based on a specific property, the "Q" values.
- Clustering via K-Means for Supervisors: K-Means clustering is employed to group supervisors based on the same attribute, the 'Q' values, as it does for students.
- Enhancing Data with Cluster Information: Based on the findings of the clustering, supervisors and students are allocated to particular clusters.
- Matching Students to Supervisors: Students are assigned to supervisors who belong to the same cluster, likely with the goal of making assignments based on similar characteristics.

4.4.2 Results

Table 4.6 displays the experimental results related to the application of K-Means in matching students to supervisors. According to the experimental results, the obtained values indicate relatively mediocre results. However, the time and space complexity is relatively better than applications of other algorithms.

Metric	**Value**
Preference Satisfaction	0.34
Balance of Workload	0.2
Time Complexity	O(num_students × num_supervisors)
Space Complexity	O(num_students + num_supervisors)
Maximum Workload	15
Minimum Workload	3
Compatibility Score	0.23

Table 4.6: Summary of Metrics in K-Means

4.5 Predicting productivity of students by supervisors

4.5.1 Explanation and used dataset

There was used dataset 3 to predict productivity of students based on their last progress. The supervisorship web service was tested on a 5-week thesis implementation by a student working with his supervisor. For 5 weeks, the student's production was [100,75,100,50,100]. Productivity was calculated to be 100 per week if a student completed at least 5 hours of work from the task list each week.

4.5.2 Results

Figure 4.2 depicts the display of a time series linear regression model that was used to predict student productivity in thesis implementation. MAE is a measure of the average absolute differences between expected and actual values. An MAE of 18.0 indicates that the model's predictions differ from the actual data by 18 percentage points on average. The lower the MAE, the more closely the model's predictions match the actual results. MAE was determined to be 18.

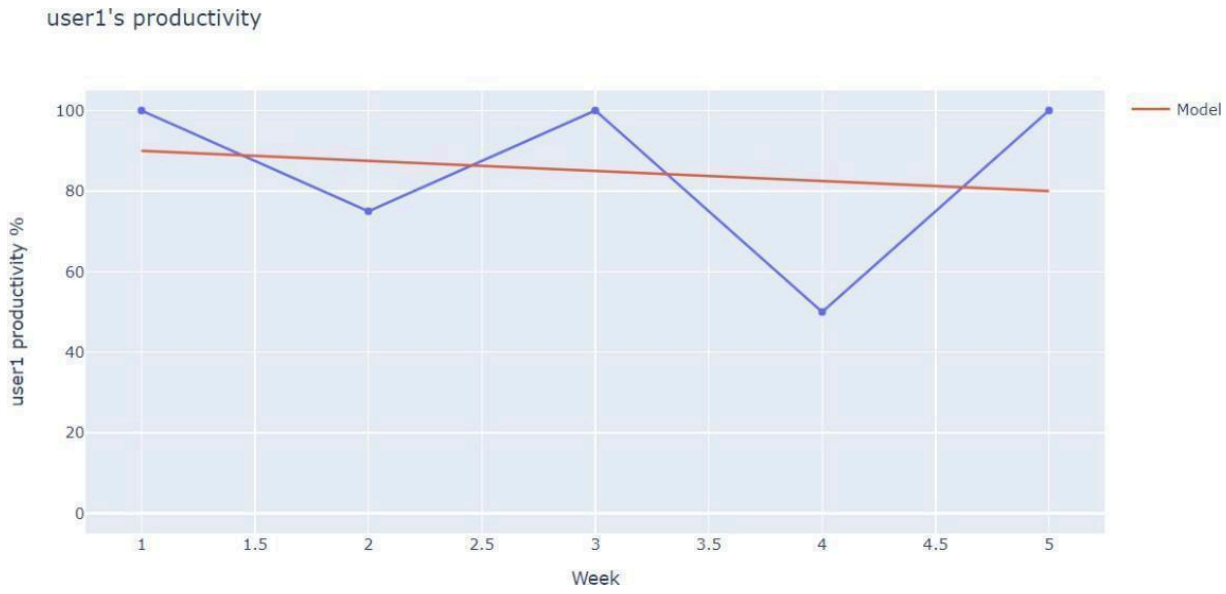

Figure 4.2: Predicting productivity of students

4.6 Association analysis

4.6.1 Explanation and used dataset

Dataset.

For this experiment, there was used dataset 1 related to perceptions of student-supervisor workflows.

Explanation.

Association analysis is a data mining technique that identifies patterns or relationships between items in a dataset. It helps to find interesting associations or correlations between variables in a large set of data. Association analysis is based on the idea that if two or more items occur together frequently, then they may be related in some way.

Association analysis has numerous applications, including market basket analysis, consumer behavior analysis, medical diagnostics, and recommendation systems [8], [97]—[105].

Association analysis can be used to explore the relationship between numerous elements that affect the student-supervisor interaction in the context of student-supervisor perceptions. It can be used, for example, to determine whether there is a link between a supervisor's obligation to choose a research topic and the student's opinion of the supervisor's involvement in designing an acceptable program and timetable of research and study for the student. It is possible to find places where the student-supervisor relationship can be enhanced by detecting such relationships.

Association analysis can also be used to identify frequent itemsets and association rules that can help to understand the preferences and behavior of students and supervisors. This information can be used to develop better programs for students and to improve the quality of supervision provided by supervisors. In general, association analysis can provide insights into the underlying relationships between different factors and can help to develop strategies for improving the student-supervisor relationship.

To make association analysis, BigML service has been used. BigML.com is a cloud- based platform that offers machine learning and predictive modeling tools to businesses and individuals without requiring specialized technical

skills. The platform allows users to easily upload and preprocess data, create and compare models using a variety of algorithms, and generate predictions and insights in real time. The platform also offers advanced features such as anomaly detection, cluster analysis, and time series forecasting. Overall, BigML.com enables users to make data-driven decisions and gain insights from their data in a simple and user-friendly manner [106].

The Apriori algorithm is a popular algorithm used for generating frequent itemsets and mining association rules in data mining. It uses a "bottom-up" approach, where frequent subsets of items are identified, and then the algorithm works its way up to larger sets of items, eventually identifying all frequent itemsets.

The algorithm works in two main steps:

- Finding Frequent Itemsets: The method first searches the database for items that appear more frequently than a predetermined minimum support level in order to find frequent itemsets. These frequently found items are then used to generate candidate itemsets of size two, which consist of pairs of items. Next, to identify frequently occurring itemsets of size 2, these candidate pairs are cross-referenced with the database. Iteratively increasing the size of the itemsets, this process continues until further frequent itemset generation is impossible.
- Generate association rules: Once the frequent itemsets have been identified, the algorithm generates association rules by examining each frequent itemset and identifying all possible non-empty subsets. The confidence of each rule is then calculated, and only those with a confidence above a specified minimum confidence threshold are output as association rules.

Factors to be used in this association analysis:

- A = How much do you agree that the supervisor should choose the research topic?
- B = How much do you agree that the supervisor(s) determine the best theoretical framework and/or methodology?
- C = How much do you agree that the supervisor(s) should create a suitable research and study schedule and program for the student?
- D = To what extent do you believe that connections between students and supervisors should only be professional in nature and shouldn't get personal?

- E = How much do you agree that the student and/or supervisor(s) should have regular meetings?
- F = How much do you agree that the supervisor(s) should make sure the student is consistently and on task on a frequent basis?
- G = How much do you agree that in order to make sure the student is on the right track, the supervisor(s) should insist on reviewing all revisions of the work?
- H = To what extent do you believe that the supervisor(s) should make sure the presentation is excellent and should help with thesis writing if needed?

4.6.2 Results

Table 4.7 presents the results of association rule mining between supervisor behaviors. The table includes five columns: Antecedent, Consequent, Antecedent Coverage, Support, and Confidence. Antecedent and Consequent refer to the supervisor behaviors that are being analyzed. Antecedent Coverage refers to the number of transactions in which the Antecedent occurs. Support refers to the number of transactions in which both Antecedent and Consequent occur. Confidence refers to the percentage of transactions in which the consequent occurs when the Antecedent also occurs.

Antecedent and consequent are terms used to describe the two parts of an association rule. The antecedent represents the condition or attribute that is used to predict the consequent, which is the outcome or consequence of the antecedent. For example, in a supermarket, a customer who purchases bread (antecedent) is likely to purchase butter (consequent). Understanding the antecedent and consequent can help businesses identify complementary products and improve their marketing strategies.

Another significant statistic in association analysis is antecedent coverage, which measures the proportion of transactions that contain the antecedent of a rule. A high antecedent coverage shows that the antecedent is a popular item in the dataset, implying that the rule is more credible. A poor antecedent coverage, on the other hand, may signal that the rule is insignificant or that further inquiry is required.

Support is a metric that measures how frequently an itemset or rule appears in a dataset. It is calculated by dividing the total number of transactions in the dataset by the number of transactions that contain both the antecedent and consequent of a rule.

A high support value indicates that the rule is frequent, and the itemset is popular in the dataset. Therefore, it can be used for identifying important items and improving recommendation systems.

Confidence is a metric that shows the strength of the relationship between an antecedent and its consequence. It is calculated by dividing the number of transactions containing both the antecedent and consequent of a rule by the number of transactions containing only the antecedent. A high confidence value indicates a strong association between the antecedent and consequent, which can be used to make predictions or generate recommendations.

Table 4.7 shows results that provide useful insights into the relationships between supervisor behaviors and can be used to inform management practices. The findings suggest that certain behaviors may co-occur and that supervisors who engage in one behavior are likely to engage in others. This can be useful for identifying areas for improvement in supervisor training and development. Additionally, the associations found between the behaviors can inform the design of interventions that target multiple behaviors simultaneously.

The results suggest that there are strong associations between the supervisor behaviors. For example, when the supervisor engages in behavior G > 4 (which could represent a high level of delegation), it is highly likely that the supervisor also engages in behavior F > 4 (which could represent a high level of feedback and communication). This is shown by the high support (23) and confidence (0.82143) for the association rule G > 4 => F > 4 and F > 4 => G > 4.

Similar strong associations were found between other behaviors, such as G > 4 and E > 4, and F > 4 and E > 4. There were also weaker associations between the behaviors, such as 2 < F <= 3 and 2 < E <= 3.

Table 4.7: Association rules between supervisor behaviors

Antecedent	**Consequent**	**Antecedent Coverage**	**Support**	**Confidence**
G > 4	F > 4	28	23	0.82
F > 4	G > 4	28	23	0.82
G > 4	E > 4	28	22	0.78
E> 4	G > 4	30	22	0.73
F > 4	E > 4	28	21	0.75
E > 4	F > 4	30	21	0.7

2 < F <= 3	2 < E <= 3	44	29	0.65
2 < E <= 3	2 < F <= 3	46	29	0.63

4.7 Comparison of Experiments on Matching

To evaluate the algorithms and discuss why the "One-to-Many Gale Shapley" algorithm was chosen as the best, there was assessment of them based on several criteria, including preference satisfaction, workload balance, time complexity, space complexity, maximum workload, minimum workload, and compatibility score.

Preference Satisfaction:

- Collaborative Filtering: 0.8
- Genetic Algorithm: 0.91
- K-means: 0.34
- One-to-Many Gale Shapley: 0.74

The "Genetic Algorithm" achieved the highest preference satisfaction score, indicating its effectiveness in satisfying the preferences of students and supervisors.

Workload Balance:

- Collaborative Filtering: 0
- Genetic Algorithm: 0.16
- K-means: 0.2
- One-to-Many Gale Shapley: 0.5

The "One-to-Many Gale Shapley" algorithm scored highest in workload balance, distributing the workload more evenly among supervisors compared to other algorithms.

Time Complexity:

- Collaborative Filtering: $O(num\_students2 \cdot num\_supervisors)$
- Genetic Algorithm: $O(num\_generations \cdot population\_size \cdot num\_students)$
- K-means: $O(num\_students \cdot num\_supervisors)$
- One-to-Many Gale Shapley: $O(num\_students \cdot num\_supervisors)$

The "Collaborative Filtering" algorithm exhibited the highest time complexity. In contrast, both the "K-means" and "One-to-Many Gale Shapley" algorithms showed lower time complexity, making them more efficient in terms of computation time.

Space Complexity:

- Collaborative Filtering: $O(num\_students \cdot num\_supervisors)$
- Genetic Algorithm: $O(population\_size \cdot num\_students)$
- K-means: $O(num\_students + num\_supervisors)$
- One-to-Many Gale Shapley: $O(num\_students + num\_supervisors)$

The "Collaborative Filtering" algorithm had the highest space complexity. The "K-means" and "One-to-Many Gale Shapley" algorithms displayed lower space complexity, indicating more efficient memory usage.

Maximum Workload:

- Collaborative Filtering: 61
- Genetic Algorithm: 12
- K-means: 15
- One-to-Many Gale Shapley: 6

The "Collaborative Filtering" algorithm resulted in the highest maximum workload, potentially overloading some supervisors. In contrast, the "One-to-Many Gale Shapley" algorithm had the lowest maximum workload, indicating a more balanced distribution of students among supervisors.

Minimum Workload:

- Collaborative Filtering: 0
- Genetic Algorithm: 2
- K-means: 3
- One-to-Many Gale Shapley: 3

The "Collaborative Filtering" algorithm had the lowest minimum workload, potentially leading to underutilization of some supervisors. The "Genetic Algorithm," "K-means," and "One-to-Many Gale Shapley" algorithms exhibited higher minimum workloads, suggesting a more equitable distribution of students.

Compatibility Score:

- Collaborative Filtering: 0.24
- Genetic Algorithm: 0.04
- K-means: 0.23
- One-to-Many Gale Shapley: 0.46

The "One-to-Many Gale Shapley" algorithm achieved the highest compatibility score, indicating more compatible student-supervisor assignments compared to other algorithms.

Based on this evaluation, the "One-to-Many Gale Shapley" algorithm emerges as the best choice. It offers high preference satisfaction, workload balance, and compatibility score. It also performs well in terms of time and space complexity and achieves the lowest maximum workload, which is crucial in maintaining a balanced workload for supervisors.

Based on dataset 4, which encompassed students that were matched to supervisors manually without the algorithm, the average preference satisfaction score was 0.48. Comparing with the experiment 4, where there was done matching by the use of one-to-many Gale-Shapley, the average preference satisfaction score was 0.74. The improvement rate is 35%.

4.8 Discussion

Based on experimental results, it can be shown that One-To-Many Gale-Shapley has in average the most optimum values with preference satisfaction 0.74, balance of workload 0.5, maximum workload 6, minimum workload 3. Considering a set of metrics showed a comprehensive approach in evaluation of algorithms in matching students to supervisors Comparing with existing work in the context of matching students to supervisors which is outlined in the literature review, this study put focus on defining similarity of students and supervisors related to psychological compatibility rather than only preferences for research areas. Also this work considered new metrics as compatibility score and preference satisfaction in the context of psychological compatibility. Adding this up into supervisorship web service with function to track progresses of students, comparing with, this service is more complex with transforming matching process and tracking process into one coherent service.

The main limitation of the conducted experiments is small size of dataset 1. While every effort has been made to ensure the representativeness of the sample, the limited number of observations may restrict the generalizability of the findings to a broader population. It is essential to recognize that a larger

dataset could offer a more comprehensive understanding of the intricacies of student-supervisor dynamics and workflow perceptions. The study's focus on the perceptions of students and supervisors within a specific context may impact the generalizability of the results to diverse educational or professional settings. The findings are grounded in the unique characteristics and nuances of the studied population, and caution should be exercised when applying these conclusions to dissimilar environments.

Despite the modest size of the dataset, consisting of 130 records with inputs from 110 undergraduate students and 20 supervisors, several factors contribute to its sufficiency for the development of robust algorithms aimed at matching students to supervisors effectively. The matching of students to supervisors is a task with a relatively focused scope. The dataset is tailored to this specific objective, ensuring that the features and parameters relevant to the matching process are adequately represented. In scenarios where the task is narrowly defined, a smaller dataset can still yield meaningful results. Despite the smaller overall size, the inclusion of diverse perspectives from both students and supervisors enhances the dataset's richness. Variability in responses, experiences, and preferences allows algorithms to learn patterns that can be more generalizable across a broader spectrum of student-supervisor relationships.

5. Supervisorship web service

The service allows students to put their tasks regarding thesis implementation and supervisors to check their progress and predict their performance. It can be quite helpful for both sides of the dissertation writing process: students and supervisors. For students, it is quite convenient to see a set of tasks to accomplish, change their priority and manage for time-management purposes, whereas for supervisors it can be beneficial to see overall progress and statistics of the students to track their progress, look up on any difficulties they currently have to share their opinion and advice to accelerate and improve the process of writing thesis. The clever integrated algorithm used by this web site expertly streamlines the process of linking students with qualified supervisors. This platform provides an ideal pairing of students and supervisors based on a variety of pertinent factors and considerations through the seamless integration of advanced computational techniques and intelligent algorithms. This web service greatly streamlines and improves the efficiency and efficacy of the matching process by utilizing cutting-edge technology. As a result, better outcomes and improved collaboration between students and their designated supervisors are the end results.

Supervisorship web service has a set of functions related to tracking progress of students and matching students to supervisors. Figure 5.1 shows a function that plays a crucial part in finding the best matches between students and supervisors, utilizing the supplied dataset to support thorough and well-informed decision-making.

The task of this function is to process and analyze the dataset, which normally contains essential data on students and supervisors, using powerful sophisticated algorithms and computational approaches. These algorithms are used by the web service to examine the dataset in a methodical manner while looking for possible alignments and synergies between students and supervisors. The goal of this comprehensive analysis is to find the most appropriate matches that have the best chances of encouraging fruitful and cooperative collaborations. It takes into account perceptions of students and supervisors about their work relationship.

The function employs advanced matching algorithms that iteratively improve and optimize the selection process in order to operate in an iterative fashion. The ultimate objective is to increase the possibility of positive interactions, allowing students to gain from the knowledge and experience of supervisors.

The web service relies on this feature to streamline what would otherwise be a difficult and time-consuming process of discovering suitable matches for students and supervisors. Significant time and effort are saved, and both students and supervisors experience greater overall success as a result.

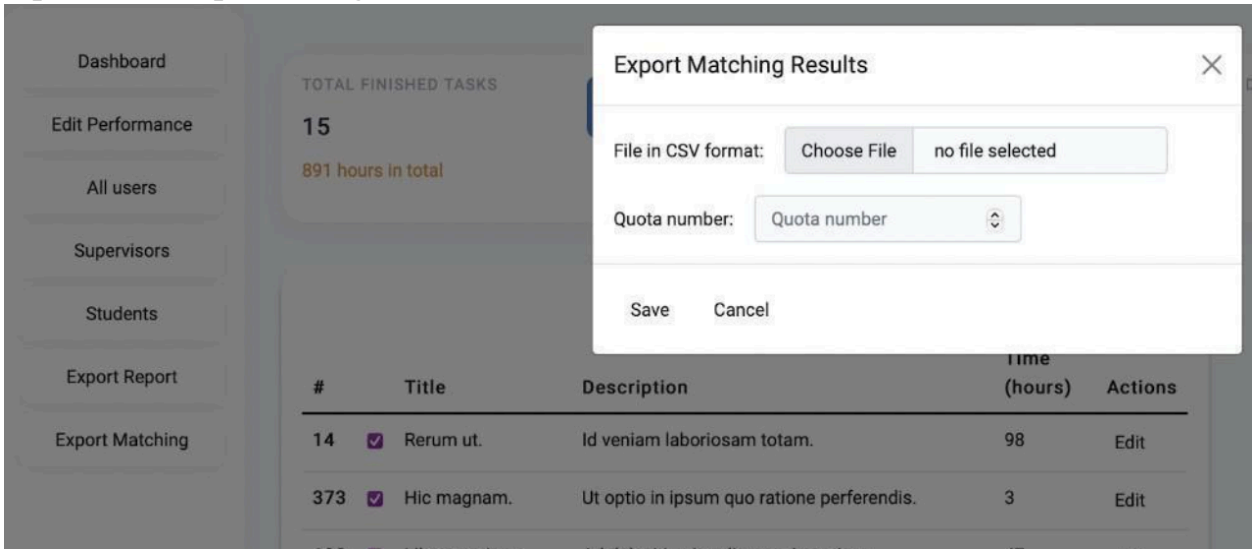

Figure 5.1: Exporting matching results based on the file

Figure 5.2 shows the process of creating a task by a student. Student enters the title of the task, its description and approximate amount of time needed to complete a task. After the student enters information related to the mentioned questions, the task appears in the task list. It can be quite convenient for a user to see the list of

tasks in the way as it is outlined in the Figure considering that user experience was a focus in the process of creating user interface in the service.

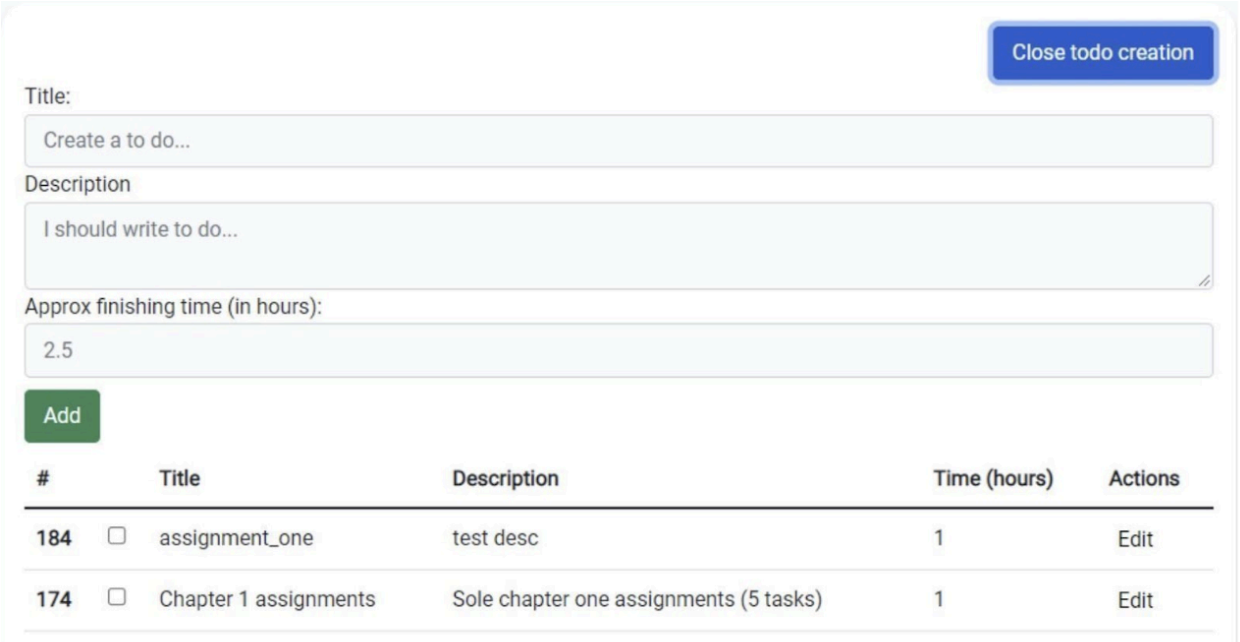

Figure 5.2: the process of creating a task by a student

Figure 5.3 shows the process of marking tasks as completed. After a student finishes his task, he checks whether everything is done according to requirements, and if that's the case, then this student will mark his task as completed. As a result, the task will disappear from the list of tasks that the student sees. All completions of the tasks have impact on measuring the productivity of a student, and time series model which is included in this project will use it in the own accord to predict productivity of students.

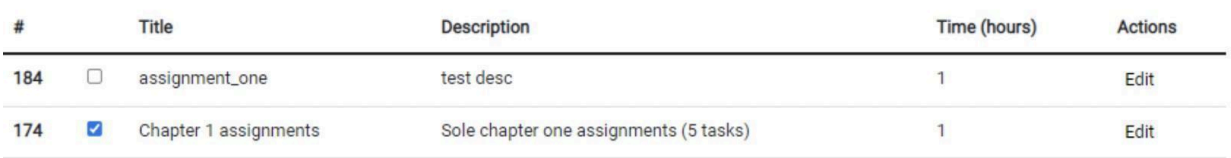

Figure 5.3: the process of marking tasks as completed

Figure 5.4 shows the main page of the service. On the main page, the administrator can see a set of tasks. He can also edit performance measures and see a set of users in the system, their roles. Also, by navigating over the menu, he can navigate to a

page which shows all supervisors and to a page that shows all students in the system.

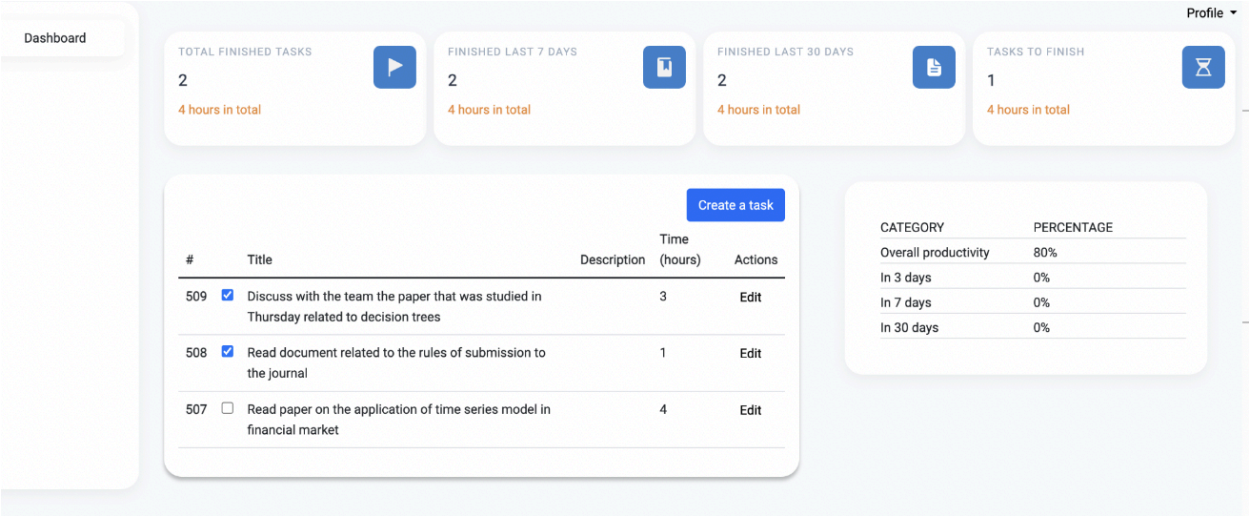

Figure 5.4: the main page of the service.

Figure 5.5 shows the section of managing users of the service. For each user, there can be seen his first name, last name, email, role and corresponding action can be applied to them considering that you are an admin. The actions that there can be applied to users are changing their role from a student to a supervisor or vice versa, and deletion of a user. Deletion results in the situation that user will not be found in the list and essential request will be sent to a server which will execute essential database management systems query to remove the record from the database.

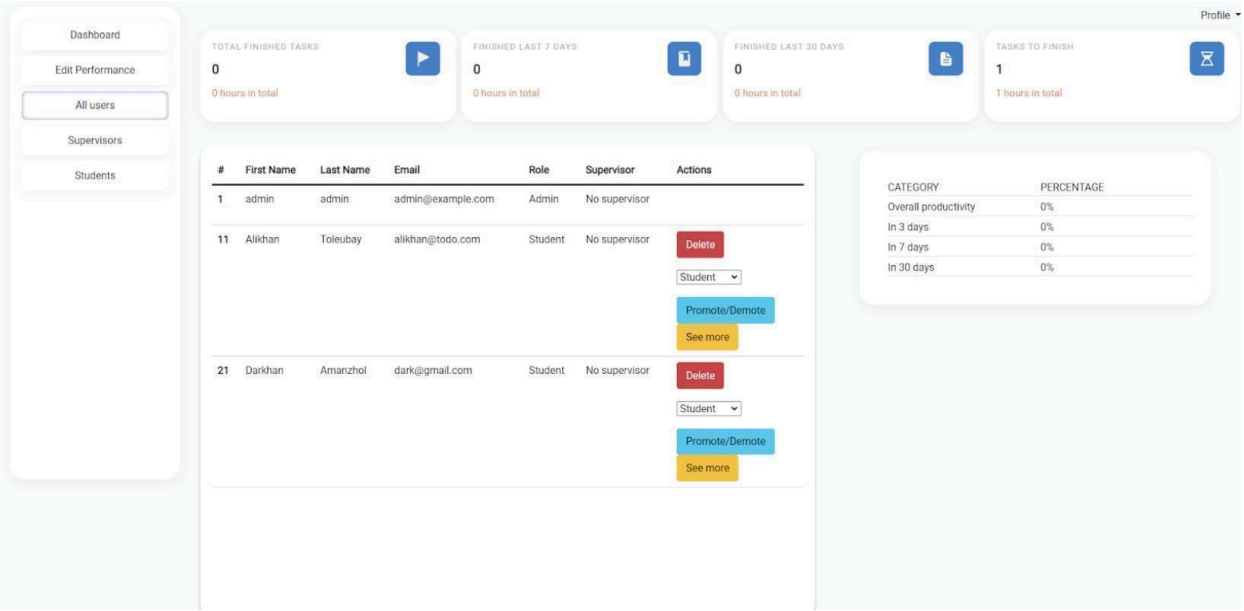

Figure 5.5: section of managing users of the service

Figure 5.6 shows the page of supervisors of the service. On this page, supervisors can manage their students, they can see their work and track their progress. Through the application and practice on this service, supervisors can use the obtained information in the formation or reformation of strategies directed at the improvement of processes regarding student-supervisor relationships and working towards the successfulness of the thesis work.

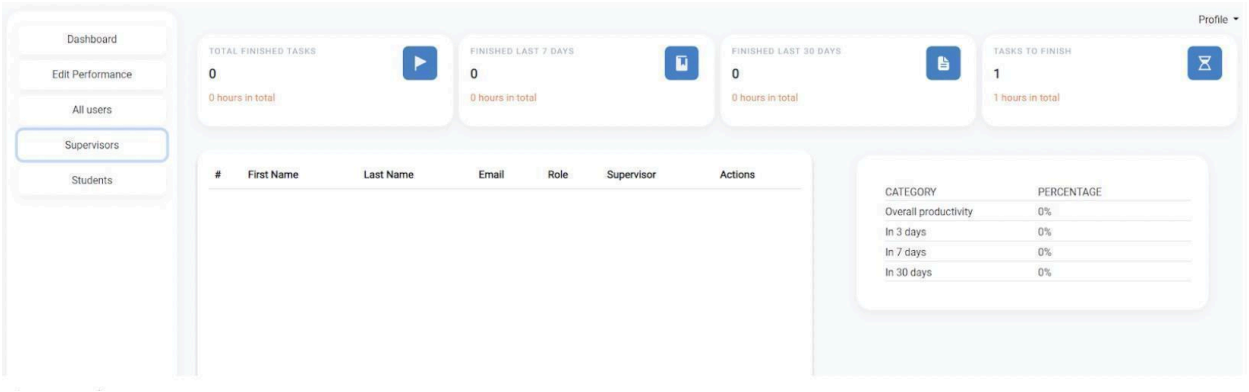

Figure 5.6: Page of supervisors

Figure 5.7 shows the page of students of the service.

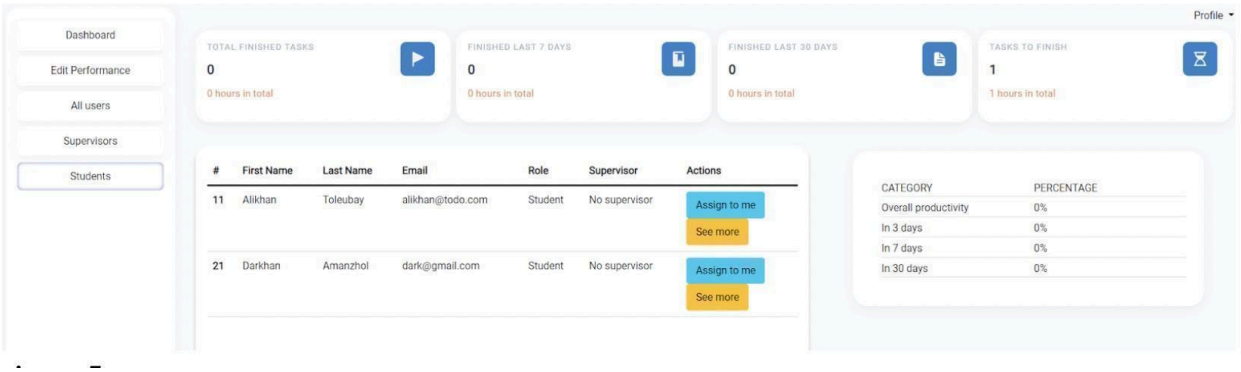

Figure 5.7: Page of students

By the use of method to predict student's progress there can be adjusted strategies to increase probability of successful implementation of thesis's. For instance, if it is predicted that productivity is going to be low for a student, then his/her supervisor can arrange one-to-one meetings in advance to define the issues and find solutions to increase productivity of a student. Then, throughout

the time, the supervisor can see progress of student and forecasting for the future weeks.

This web service's incorporation of an algorithmic matching system is of utmost importance for a number of compelling reasons.

First off, academic activities' quality and performance are greatly influenced by the process of pairing students with supervisors. By allowing students to obtain the appropriate advice, support, and expertise from supervisors who have the necessary knowledge and experience in their respective professions, a well-aligned pairing increases the likelihood of successful and beneficial collaborations. As a result, this enables students to have a more meaningful and enriching learning experience, supporting their intellectual development and general academic advancement.

Second, an integrated algorithmic matching system makes the distribution of supervisors efficient and equitable. This web service can evaluate and analyze a massive quantity of data, taking into account elements like research interests, competence, availability, and compatibility, by using computational techniques. This makes it possible to evaluate and compare many characteristics in-depth, ensuring a fair and impartial matching procedure that increases the likelihood of finding the best supervisor for each student. This technique helps to reduce potential mismatches and improves overall satisfaction for both students and supervisors by removing subjective biases and human limitations.

Additionally, the web service's integration of such a system offers scalability and flexibility. Manual matching becomes more time-consuming and error-prone as the number of students and supervisors rises. Large datasets may be processed accurately and efficiently using an algorithmic method, making it easier to scale the service to handle an increasing user base. The algorithm can also be improved and modified in response to user input and changing needs, ensuring ongoing improvement and flexibility to meet new demands.

The use of an algorithmic matching mechanism also encourages accountability and transparency. The matching decisions can be explained and justified in detail thanks to the ability to audit and document the underlying mathematical process. Through continued review and refinement of the matching algorithm, this transparency not only fosters user trust but also ensures the matching algorithm's effectiveness and fairness throughout time.

In summary, this web service's incorporation of an algorithmic matching system is a key component in streamlining the process of matching students with supervisors. It raises the standard of academic cooperation, makes the distribution of supervisors efficient and equitable, allows for scalability and adaptability, and encourages transparency and accountability.

6. Conclusion

Supervisorship service has been implemented which has a feature to match students to supervisors based on psychological perceptions of students and supervisors regarding their workflows and a feature of tracking progress of students.

There was collected data to measure student-supervisor perceptions regarding their workflows based on what there was conducted experiments to match students to supervisors. New metrics specifically to compare matching students to supervisors were defined such as preference satisfaction, balance of workload, minimum workload, maximum workload, compatibility score.

4 different algorithms were applied for matching students to supervisors such as One- to-Many Gale-Shapley, K-Means, collaborative filtering, and genetic algorithm. They were compared according to the defined metrics. Based on experimental results related to matching students to supervisors the following points were made:

1. One-to-Many Gale Shapley offers efficient execution with low time and space complexity and balance of workload 0.5. Also, it has relatively good values comparing with majority of used algorithms which indicate that the usage of One-to-Many Gale-Shapley is the most optimal approach to match students to supervisors.
2. The Genetic Algorithm excels in preference satisfaction 0.91, however it has very poor workload balance and compatibility and can be not suitable for matching students to supervisors in practice.
3. Collaborative Filtering despite having high preference satisfaction has very poor balance of workload.
4. K-means obtained relatively not optimal results by getting 0.34 preference satisfaction and 0.2 balance of work.

Derived from dataset 4, which involved manually matching students with supervisors without utilizing the algorithm, the average satisfaction score for preferences was 0.48. Contrasting this with experiment 4, where matching was performed using the one-to-many Gale-Shapley algorithm, the average satisfaction score for preferences was 0.74. This represents a 35% increase in the satisfaction rate.

References

1. D. E., “A growing phobia”, Nature, 2017.
2. H. C. Yarwood-Ross L, “As others see us: What phd students say about supervi- sors”, Nurse Researcher, pp. 38–43, 2014.
3. R. A. de Kleijn, M. T. Mainhard, P. C. Meijer, A. Pilot, and M. Brekelmans, “Master’s thesis supervision: Relations between perceptions of the supervisor– student relationship, final grade, perceived supervisor contribution to learning and student satisfaction”, Studies in Higher Education, vol. 37, no. 8, pp. 925– 939, 2012.
4. Z. Liao and Y. Liu, “Abusive supervision and psychological capital: A mediated moderation model of team member support and supervisor-student exchange”, Frontiers of Business Research in China, vol. 9, no. 4, p. 576, 2015.
5. M. H. Ismail, T. R. Razak, M. A. Hashim, and A. F. Ibrahim, “A simple recommender engine for matching final-year project student with supervisor”, arXiv preprint arXiv:1908.03475, 2019.
6. T. Kawagoe and T. Matsubae, “Matching with minimal quota: Case study of a student-supervisor assignment in a japanese university”, Available at SSRN 3429626, 2020.
7. J. Ren, F. Xia, X. Chen, et al., “Matching algorithms: Fundamentals, applications and challenges”, IEEE Transactions on Emerging Topics in Computational Intelligence, vol. 5, no. 3, pp. 332–350, 2021.
8. R. Agrawal and R. Srikant, “Fast algorithms for mining association rules in large databases”, in Proceedings of the 20th International Conference on Very Large Data Bases, ACM, 1994, pp. 487–499.
9. J. Han, J. Pei, and Y. Yin, “Mining frequent patterns without candidate gener- ation”, in Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, 2000, pp. 1–12.
10. J. Wang, J. Han, and X. Li, “Mining frequent patterns in data streams at multi- ple time granularities”, IEEE Transactions on Knowledge and Data Engineering, vol. 19, no. 8, pp. 1088–1102, 2007.
11. B. Liu, W. Hsu, and Y. Ma, “Integrating classification and association rule min- ing”, in Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining (KDD), ACM, 1998, pp. 80–86.
12. J. Lu, D. Chen, and J. Wang, “A survey of association rules mining in big data”, Knowledge-Based Systems, vol. 89, pp. 42–58, 2016.
13. S. H. Jeong, J. Y. Kim, and K. H. Song, “A study on the stock price prediction using linear regression analysis: A case of samsung electronics”,

Journal of Open Innovation: Technology, Market, and Complexity, vol. 5, no. 1, p. 5, 2019.
14. C.-Y. Hsieh, C.-H. Lin, H.-L. Lin, and Y.-C. Chiu, “Predicting alzheimer’s dis- ease progression using a linear regression model”, Journal of Alzheimer’s Disease, vol. 68, no. 3, pp. 1047–1056, 2019.
15. Y. Guo, X. Wang, H. Wang, and Y. Chen, “Prediction of remaining useful life of aircraft engines based on linear regression analysis”, Aerospace Science and Technology, vol. 80, pp. 405–413, 2018.
16. M. L. Orellana, A. Darder, A. P´erez, and J. Salinas, “Improving doctoral success by matching phd students with supervisors”, International Journal of Doctoral Studies, vol. 11, p. 87, 2016.
17. H. Edwards, T. Aspland, J. O’Leary, Y. Ryan, G. Southey, and P. Timms, Tracking postgraduate supervision. Queensland University of Technology, 1995.
18. O. Jassim, M. Mahmoud, and M. Ahmad, “A multi-agent framework for research supervision management”, in Distributed Computing and Artificial Intelligence, 12th International Conference, Springer, 2015.
19. V. Sanchez-Anguix, R. Chalumuri, and V. Julian, “A multi-objective evolutionary proposal for matching students to supervisors”, in Distributed Computing and Artificial Intelligence, 15th International Conference 15, Springer, 2019, pp. 94– 102.
20. M. C. Wijanto, R. Rachmadiany, and O. Karnalim, “Thesis supervisor recommen- dation with representative content and information retrieval”, Journal of Infor- mation Systems Engineering and Business Intelligence, vol. 6, no. 2, pp. 143–150, 2020.
21. Y. Gao, K. Ilves, and D. G lowacka, “Officehours: A system for student super- visor matching through reinforcement learning”, in Proceedings of the 20th In- ternational Conference on Intelligent User Interfaces Companion, 2015, pp. 29– 32.
22. L. Wang, Y. Li, J. Huang, and S. Lazebnik, “Learning two-branch neural net- works for image-text matching tasks”, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 41, no. 2, pp. 394–407, 2018.
23. M. Akbarpour, S. Li, and S. O. Gharan, “Thickness and information in dynamic matching markets”, Journal of Political Economy, vol. 128, no. 3, pp. 783–815, 2020.
24. A. E. Roth and M. Sotomayor, “Two-sided matching”, Handbook of game theory with economic applications, vol. 1, pp. 485–541, 1992.
25. G. S. Becker, “A theory of marriage: Part i”, Journal of Political economy, vol. 81, no. 4, pp. 813–846, 1973.

26. T. C. Bergstrom and M. Bagnoli, “Courtship as a waiting game”, Journal of political economy, vol. 101, no. 1, pp. 185–202, 1993.
27. D. Gale and L. S. Shapley, “College admissions and the stability of marriage”, The American Mathematical Monthly, vol. 120, no. 5, pp. 386–391, 2013.
28. Y.-K. Che, J. Kim, and F. Kojima, “Stable matching in large economies”, Econo- metrica, vol. 87, no. 1, pp. 65–110, 2019.
29. A. Ismaili, N. Hamada, Y. Zhang, T. Suzuki, and M. Yokoo, “Weighted match- ing markets with budget constraints”, Journal of Artificial Intelligence Research, vol. 65, pp. 393–421, 2019.
30. A. S. Kelso Jr and V. P. Crawford, “Job matching, coalition formation, and gross substitutes”, Econometrica: Journal of the Econometric Society, pp. 1483–1504, 1982.
31. X. Su and T. M. Khoshgoftaar, “A survey of collaborative filtering techniques”, Advances in artificial intelligence, vol. 2009, 2009.
32. T. Tran, K. Lee, Y. Liao, and D. Lee, “Regularizing matrix factorization with user and item embeddings for recommendation”, in Proceedings of the 27th ACM in- ternational conference on information and knowledge management, 2018, pp. 687– 696.
33. S. Sedhain, A. K. Menon, S. Sanner, and L. Xie, “Autorec: Autoencoders meet collaborative filtering”, in Proceedings of the 24th international conference on World Wide Web, 2015, pp. 111–112.
34. Y. Wu, C. DuBois, A. X. Zheng, and M. Ester, “Collaborative denoising auto- encoders for top-n recommender systems”, in Proceedings of the ninth ACM in- ternational conference on web search and data mining, 2016, pp. 153–162.
35. Y. Koren, S. Rendle, and R. Bell, “Advances in collaborative filtering”, Recom- mender systems handbook, pp. 91–142, 2022.
36. H.-J. Xue, X. Dai, J. Zhang, S. Huang, and J. Chen, “Deep matrix factorization models for recommender systems.”, in IJCAI, Melbourne, Australia, vol. 17, 2017, pp. 3203–3209.
37. J. Chen, H. Zhang, X. He, L. Nie, W. Liu, and T.-S. Chua, “Attentive collaborative filtering: Multimedia recommendation with item-and component-level attention”, in Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval, 2017, pp. 335–344.
38. X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, “Neural collaborative filtering”, in Proceedings of the 26th international conference on world wide web, 2017, pp. 173–182.

39. Y. Tay, L. Anh Tuan, and S. C. Hui, “Latent relational metric learning via memory-based attention for collaborative ranking”, in Proceedings of the 2018 world wide web conference, 2018, pp. 729–739.
40. J. Xiao, H. Ye, X. He, H. Zhang, F. Wu, and T.-S. Chua, “Attentional factorization machines: Learning the weight of feature interactions via attention networks”, arXiv preprint arXiv:1708.04617, 2017.
41. H.-C. Lu, F. Hwang, and Y.-H. Huang, “Parallel and distributed architecture of genetic algorithm on apache hadoop and spark”, Applied Soft Computing, vol. 95, p. 106 497, 2020.
42. Y. Blanco-Ferna´ndez, M. Lo´pez-Nores, A. Gil-Solla, M. Ramos-Cabrer, and J. J. Pazos-Arias, “Exploring synergies between content-based filtering and spreading activation techniques in knowledge-based recommender systems”, Information Sci- ences, vol. 181, no. 21, pp. 4823–4846, 2011.
43. M. J. Pazzani, “A framework for collaborative, content-based and demographic filtering”, Artificial intelligence review, vol. 13, no. 5, pp. 393–408, 1999.
44. P. B. Thorat, R. Goudar, and S. Barve, “Survey on collaborative filtering, content- based filtering and hybrid recommendation system”, International Journal of Computer Applications, vol. 110, no. 4, pp. 31–36, 2015.
45. J. Basilico and T. Hofmann, “Unifying collaborative and content-based filtering”, in Proceedings of the twenty-first international conference on Machine learning, 2004, p. 9.
46. Y. Li and H. Wu, “A clustering method based on k-means algorithm”, Physics Procedia, vol. 25, pp. 1104–1109, 2012.
47. B. Rocca, Introduction to recommender systems, https://towardsdatascience.com/introduction-to-recommender-systems-6c66cf15ada.
48. I. Mavridis and H. Karatza, “Performance evaluation of cloud-based log file anal- ysis with apache hadoop and apache spark”, Journal of Systems and Software, vol. 125, pp. 133–151, 2017.
49. J. Liu, J. Ren, W. Zheng, L. Chi, I. Lee, and F. Xia, “Web of scholars: A scholar knowledge graph”, in Proceedings of the 43rd International ACM SIGIR Con- ference on Research and Development in Information Retrieval, 2020, pp. 2153– 2156.
50. J. Liu, J. Tian, X. Kong, I. Lee, and F. Xia, “Two decades of information systems: A bibliometric review”, Scientometrics, vol. 118, no. 2, pp. 617–643, 2019.
51. I. Sutskever, J. Tenenbaum, and R. R. Salakhutdinov, “Modelling relational data using bayesian clustered tensor factorization”, Advances in neural information processing systems, vol. 22, 2009.

52. M. Nickel, V. Tresp, and H.-P. Kriegel, “A three-way model for collective learning on multi-relational data”, in Icml, 2011.
53. A. Garc´ıa-Dur´an, A. Bordes, and N. Usunier, “Effective blending of two and three- way interactions for modeling multi-relational data”, in Joint European Confer- ence on Machine Learning and Knowledge Discovery in Databases, Springer, 2014, pp. 434–449.
54. H. Liu, Y. Wu, and Y. Yang, “Analogical inference for multi-relational embed- dings”, in International conference on machine learning, PMLR, 2017, pp. 2168– 2178.
55. A. Bordes, X. Glorot, J. Weston, and Y. Bengio, “A semantic matching energy function for learning with multi-relational data”, Machine Learning, vol. 94, no. 2, pp. 233–259, 2014.
56. R. Socher, D. Chen, C. D. Manning, and A. Ng, “Reasoning with neural tensor networks for knowledge base completion”, Advances in neural information pro- cessing systems, vol. 26, 2013.
57. X. Dong, E. Gabrilovich, G. Heitz, et al., “Knowledge vault: A web-scale approach to probabilistic knowledge fusion”, in Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, 2014, pp. 601– 610.
58. Q. Liu, H. Jiang, A. Evdokimov, et al., “Probabilistic reasoning via deep learning: Neural association models”, arXiv preprint arXiv:1603.07704, 2016.
59. A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, “Trans- lating embeddings for modeling multi-relational data”, Advances in neural infor- mation processing systems, vol. 26, 2013.
60. Z. Wang, J. Zhang, J. Feng, and Z. Chen, “Knowledge graph embedding by trans- lating on hyperplanes”, in Proceedings of the AAAI conference on artificial intel- ligence, vol. 28, 2014.
61. Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu, “Learning entity and relation embeddings for knowledge graph completion”, in Twenty-ninth AAAI conference on artificial intelligence, 2015.
62. M. D. Levine, D. A. O’Handley, and G. M. Yagi, “Computer determination of depth maps”, Computer Graphics and Image Processing, vol. 2, no. 2, pp. 131– 150, 1973.
63. A. Gruen, “Adaptive least squares correlation: A powerful image matching tech- nique”, South African Journal of Photogrammetry, Remote Sensing and Cartog- raphy, vol. 14, no. 3, pp. 175–187, 1985.
64. P. Wu, W. Li, and W. Song, “Fast, accurate normalized cross-correlation image matching”, Journal of Intelligent & Fuzzy Systems, vol. 37, no. 4, pp. 4431–4436, 2019.

65. R. Ishiyama, T. Takahashi, K. Makino, and Y. Kudo, “Fast image matching based on fourier-mellin phase correlation for tag-less identification of mass-produced parts”, in 2018 IEEE Global Conference on Signal and Information Processing (GlobalSIP), IEEE, 2018, pp. 380–384.
66. Q. Yang and Q. Wei, “An image matching algorithm based on mutual information for small dimensionality target”, in 2018 14th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD), IEEE, 2018, pp. 1166–1171.
67. C. Harris, M. Stephens, et al., “A combined corner and edge detector”, in Alvey vision conference, Citeseer, vol. 15, 1988, pp. 10–5244.
68. D. G. Lowe, “Distinctive image features from scale-invariant keypoints”, Interna- tional journal of computer vision, vol. 60, no. 2, pp. 91–110, 2004.
69. H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, “Speeded-up robust features (surf)”, Computer vision and image understanding, vol. 110, no. 3, pp. 346–359, 2008.
70. E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, “Orb: An efficient alternative to sift or surf”, in 2011 International conference on computer vision, Ieee, 2011, pp. 2564–2571.
71. U. M. Babri, M. Tanvir, and K. Khurshid, “Feature based correspondence: A com- parative study on image matching algorithms”, International Journal of Advanced Computer Science and Applications, vol. 7, no. 3, 2016.
72. E. Karami, S. Prasad, and M. Shehata, “Image matching using sift, surf, brief and orb: Performance comparison for distorted images”, arXiv preprint arXiv:1710.02726, 2017.
73. Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, “Deepface: Closing the gap to human-level performance in face verification”, in Proceedings of the IEEE confer- ence on computer vision and pattern recognition, 2014, pp. 1701–1708.
74. F. Schroff, D. Kalenichenko, and J. Philbin, “Facenet: A unified embedding for face recognition and clustering”, in Proceedings of the IEEE conference on com- puter vision and pattern recognition, 2015, pp. 815–823.
75. K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recog- nition”, in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.
76. D. Gale and L. S. Shapley, “College admissions and the stability of marriage”, The American Mathematical Monthly, vol. 69, no. 1, pp. 9–15, 1962.
77. M. Soua, R. Kachouri, and M. Akil, “A new hybrid binarization method based on kmeans”, in 2014 6th International Symposium on

Communications, Control and Signal Processing (ISCCSP), IEEE, 2014, pp. 118–123.
78. M. Soua, R. Kachouri, and M. Akil, “Improved hybrid binarization based on kmeans for heterogeneous document processing”, in 2015 9th International Sym- posium on Image and Signal Processing and Analysis (ISPA), IEEE, 2015, pp. 210– 215.
79. H. Vinod and S. Niranjan, “Binarization and segmentation of kannada hand-written document images”, in 2018 Second International Conference on Green Computing and Internet of Things (ICGCIoT), IEEE, 2018, pp. 488–493.
80. X. Huang, Y. Ye, H. Guo, Y. Cai, H. Zhang, and Y. Li, “Dskmeans: A new kmeans-type approach to discriminative subspace clustering”, Knowledge-Based Systems, vol. 70, pp. 293–300, 2014.
81. X. Ji and F. Lu, “K-means clustering analysis and evaluation for internet of acous- tic environment characteristics”, Cognitive Systems Research, vol. 52, pp. 603– 609, 2018.
82. I. Dabbura, “K-means clustering: Algorithm, applications, evaluation methods, and drawbacks”, Towards Data Science, 2018.
83. P. Arora, S. Varshney, et al., “Analysis of k-means and k-medoids algorithm for big data”, Procedia Computer Science, vol. 78, pp. 507–512, 2016.
84. S. Jain, A. Grover, P. S. Thakur, and S. K. Choudhary, “Trends, problems and solutions of recommender system”, in International Conference on Computing, Communication & Automation, IEEE, 2015, pp. 955–958.
85. K-Means Clustering Algorithm, https://www.javatpoint.com/k-means-clustering-algorithm-in-machine-learning.
86. R. V. Bullough Jr and R. J. Draper, “Making sense of a failed triad: Mentors, uni- versity supervisors, and positioning theory”, Journal of teacher education, vol. 55, no. 5, pp. 407–420, 2004.
87. A. K. Sharma, B. Bajpai, R. Adhvaryu, S. D. Pankajkumar, P. P. Gordhanbhai, and A. Kumar, “An efficient approach of product recommendation system using nlp technique”, Materials Today: Proceedings, 2021.
88. J. Z. Huang, M. K. Ng, H. Rong, and Z. Li, “Automated variable weighting in k-means type clustering”, IEEE transactions on pattern analysis and machine intelligence, vol. 27, no. 5, pp. 657–668, 2005.
89. W. Min and Y. Siqing, “Improved k-means clustering based on genetic algorithm”, in 2010 International Conference on Computer Application and System Modeling (ICCASM 2010), IEEE, vol. 6, 2010, pp. V6–636.

90. W.-L. Hung, M.-S. Yang, and C.-M. Hwang, “Exponential-distance weighted k- means algorithm with spatial constraints for color image segmentation”, in 2011 International Conference on Multimedia and Signal Processing, IEEE, vol. 1, 2011, pp. 131–135.
91. Q. Qiu, Q. Zhang, and K. Guo, “Grey kmeans algorithm and its application to the analysis of regional competitive ability”, in 2014 IEEE 7th Joint Interna- tional Information Technology and Artificial Intelligence Conference, IEEE, 2014, pp. 249–253.
92. J. Xu and H. Liu, “Web user clustering analysis based on kmeans algorithm”, in 2010 International Conference on Information, Networking and Automation (ICINA), 2010.
93. A. Nowak-Brzezin´ska and C. Horyn´, “Outliers in rules-the comparision of lof, cof and kmeans algorithms.”, Procedia Computer Science, vol. 176, pp. 1420–1429, 2020.
94. C. Bouras and V. Tsogkas, “Clustering user preferences using w-kmeans”, in 2011 Seventh International Conference on Signal Image Technology & Internet-Based Systems, IEEE, 2011, pp. 75–82.
95. “Supervisor & research student expectation questionnaire”. [Accessed: 23-Apr- 2022]. (), [Online]. Available: https://www.notredame.edu.au/__data/assets/ pdf_file/0019/13078/Supervisor-Research-Student-Expectation-Questionnair e. pdf.
96. “Msca supervisors dataset landing page”. Accessed: October 2, 2023. (), [Online]. Available: https://civis.opendatasoft.com/explore/dataset/msca-supervisors/.
97. H. Chen, R. H. Chiang, and V. C. Storey, “Business intelligence and analytics: From big data to big impact”, MIS quarterly, vol. 36, no. 4, pp. 1165–1188, 2012.
98. N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “Smote: Syn- thetic minority over-sampling technique”, Journal of artificial intelligence research, vol. 16, pp. 321–357, 2002.
99. I. Anagnostopoulos, C. Anagnostopoulos, and A. V. Vasilakos, “Recommender systems: An introduction”, in Recommender Systems Handbook, Springer, 2014, pp. 1–34.
100. X. Wu and V. Kumar, The top ten algorithms in data mining. CRC press, 2009.
101. B. Liu and K. Zhang, “A survey of association mining in evolutionary databases”, Knowledge and Information Systems, vol. 32, no. 2, pp. 227–258, 2012.

102. M. Kantardzic, Data mining: concepts, models, methods, and algorithms. John Wiley Sons, 2011.
103. P.-N. Tan, M. Steinbach, and V. Kumar, Introduction to data mining. Pearson Education, 2019.
104. L. Zhang, X. Luo, and J. Yang, “Association rule mining: A survey”, IEEE Trans- actions on Systems, Man, and Cybernetics: Systems, vol. 44, no. 6, pp. 793–808, 2014.
105. J. Han and M. Kamber, Data mining: concepts and techniques. Elsevier, 2006.
106. BigML Inc., Bigml - machine learning made easy, accessed 2023. [Online]. Avail- able: https://bigml.com/