Ministry of Science and Higher Education of the Republic of Kazakhstan

Suleyman Demirel University

Makhul Maulen

Development of Recommendation System for Online Library

THESIS

Presented in Partial Fulfilment for the

Master of Technical Sciences Degree in Computer Science

(degree code: 7M06102)

Department of Computer Science

Faculty of Engineering and Natural Sciences

Supervisor: **Dauren Ayazbayev**

Kaskelen 2023

Suleyman Demirel University
Faculty of Engineering and Natural Sciences
Department of Computer Science

Dean of Faculty

Associate Professor

PhD Azamat Zhamanov

06 06 2023

Topic of the thesis:

Development of Recommendation system for online library

Thesis submitted as part of the requirements for the award of the MSc in "7M06102 - Computer Science" SDU, 2021-2023

Head of Department _____ Assistant Professor, PhD Mukash Zh.

Academic Supervisor _____ Dauren Ayazbayev

Master student _____ Maulen Makhul

Kaskelen 2023

Declaration

I confirm that this is my own work and the use of all material from other sources has been properly and fully acknowledged.

Makhul Maulen

2023

Acknowledgements

I would like to express my gratitude to my supervisor for guiding me and giving me materials to study the topic in depth. I also want to thank my friends Zhanbolat, Rustem and Nurken for helping with the experiments.

Abstract

This dissertation is devoted to the development of a recommendation system for an online library. The aim of the research is to create an efficient and personalized recommendation system that takes into account user preferences, comments in Kazakh language and likes to provide up-to-date book recommendations. The paper considers various methods of data collection, including the use of a telegram bot to generate comments in the Kazakh language and the collection of information from familiar users. Created a dataset containing 330 comments in Kazakh for model training and was divided into positive and negative comments using sentiment analysis methods. Various classification models were used for sentiment analysis, including Logistic Regression, Random Forest, Naive Bayes, and Support vector machine. The Support vector machine model achieved the highest accuracy of 95%, outperforming other models.

In addition, the analysis of comments using histogram showed that positive comments usually contain more words than negative comments, which indicates more detailed and informative reviews. The identification of influential words and phrases provided insight into what aspects of books are valued by users. The developed recommendation system was integrated into the website of the online library. Two new users were created, who were given the opportunity to choose their preferred genres and languages. The system used the positive comments and likes associated with each book to generate personalized recommendations. This approach allows users to quickly find books they are interested in, which have already been popular and received positive feedback from other readers.

This research has practical implications for developers of online libraries and other platforms where personalized recommendations are required. The results and con-

clusions of this work can be used in the further development and improvement of recommendation systems, which will lead to an improvement in the quality of user service and an increase in their satisfaction.

Аңдатпа

Бұл диссертация онлайн кітапхана үшін ұсыныс беру жүйесін әзірлеуге арналған. Зерттеудің мақсаты - жаңартылған кітап ұсыныстарын қамтамасыз ету үшін пайдаланушы пікірлерін және қалауларын ескеретін тиімді және жекелендірілген ұсыныстар жүйесін құру. Жұмыста мәліметтерді жинаудың әртүрлі әдістері, соның ішінде қазақ тілінде пікірлер жинау үшін телеграмм ботын пайдалану және таныс қолданушылардан ақпарат жинау қарастырылған. Модельдік оқыту үшін қазақ тіліндегі 330 пікірден тұратын деректер жинағы жасалды, ол көңіл-күйді талдау әдістерін қолдану арқылы оң және теріс пікірлерге бөлінді. Көңіл-күйді талдау үшін әртүрлі жіктеу үлгілері пайдаланылды, соның ішінде Логистикалық регрессия (Logistic Regression), Кездейсоқ орман (Random Forest), Найв Байес (Naive Bayes) және Тірек векторлық машиналар (Support vector machine). Тірек векторлық машина (Support vector machine) моделі басқа модельдерден асып түсетін 95% ең жоғары дәлдікке қол жеткізді.

Сонымен қатар, гистограмманы пайдалана отырып, пікірлерді талдау оң пікірлерде әдетте теріс пікірлерге қарағанда көбірек сөздер бар екенін көрсетті, бұл егжей-тегжейлі және мазмұнды шолуларды көрсетеді. Әсер ететін сөздер мен сөз тіркестерін анықтау оқырмандардың кітаптың қандай аспектілерін бағалайтынын түсінуге мүмкіндік берді. Әзірленген кеңес беру жүйесі онлайн кітапхана веб-сайтына біріктірілген. Екі жаңа пайдаланушы құрылды, оларға қалаған жанрлар мен тілдерді таңдау мүмкіндігі берілді. Жүйе жеке ұсыныстар жасау үшін әрбір кітапқа қатысты оң пікірлер мен ұнатуларды пайдаланды. Бұл тәсіл пайдаланушыларға өздерін қызықтырған, бұрыннан танымал болған және басқа оқырмандардан оң пікірлер алған кітаптарды

жылдам табуға мүмкіндік береді.

Бұл зерттеудің жекелендірілген ұсыныстар қажет онлайн кітапханалар мен басқа платформаларды әзірлеушілер үшін практикалық салдары бар. Осы жұмыстың нәтижелері мен қорытындылары ұсынымдар жүйесін одан әрі дамыту мен жетілдіруде пайдаланылуы мүмкін, бұл пайдаланушыларға қызмет көрсету сапасының артуына және олардың қанағаттанушылығының артуына әкеледі.

Аннотация

Данная диссертация посвящена разработке рекомендательной системы для онлайн-библиотеки. Цель исследования — создать эффективную и персонализированную систему рекомендаций, которая учитывает предпочтения пользователей, комментарии на казахском языке и предпочтения, чтобы предоставлять актуальные рекомендации по книгам. В работе рассмотрены различные методы сбора данных, в том числе использование телеграм-бота для генерации комментариев на казахском языке и сбор информации от знакомых пользователей. Создан набор данных, содержащий 330 комментариев на казахском языке для обучения модели, который был разделен на положительные и отрицательные комментарии с помощью методов анализа настроений. Для анализа настроений использовались различные модели классификации, в том числе Логистическая регрессия, Случайный лес, Наивный Байесовский метод и Метод опорных векторов. Модель машины опорных векторов достигла наивысшей точности 95%, превзойдя другие модели.

Кроме того, анализ комментариев с помощью гистограммы показал, что положительные комментарии обычно содержат больше слов, чем отрицательные, что свидетельствует о более подробных и информативных отзывах. Выявление влиятельных слов и фраз позволило понять, какие аспекты книг ценят пользователи. Разработанная рекомендательная система интегрирована в сайт онлайн-библиотеки. Были созданы два новых пользователя, которым была предоставлена возможность выбирать предпочитаемые жанры и языки. Система использовала положительные комментарии и лайки, связанные с каждой книгой, для создания персонализированных рекомендаций. Такой подход позволяет пользователям быстро находить интересующие их книги,

которые уже пользовались популярностью и получили положительные отзывы других читателей.

Это исследование имеет практическое значение для разработчиков онлайн-библиотек и других платформ, где требуются персональные рекомендации. Результаты и выводы данной работы могут быть использованы при дальнейшей разработке и совершенствовании рекомендательных систем, что приведет к повышению качества обслуживания пользователей и повышению их удовлетворенности.

Abbreviations

SVM Support Vector Machine
mySQL Structured Query Language
TF-IDF Term Frequency-Inverse Document Frequency

Table of Contents

Chapter 1

Introduction

To understand what a recommendation system is, we need to understand a huge amount of information, so we will start with the following questions: what problems does the recommendation system solve and how it is applied.

1.1 Recommendation in real life

Every week somebody buy products in one place and he or she is a good buyer, so the seller knows if he advises him or her on some quality products, they will buy them, although they were originally going to buy strictly according to the list [1]. Sometimes the seller did not advise to buy products whose quality was not up to par, and people trusted his opinion. This is an example of a recommendation. He only recommends products he owns, but when it comes to other things like books, movies, or music he can't recommend. For example, somebody buy movie discs in a store and every time buy it remembers the genre of the movie and asks if he or she liked the movie. He collects which films customers liked or disliked, based on this, he advises others. This is also an example of a recommendation [2].
A recommendation system is not only an intricate algorithm. It is also based on understanding data and users. Data scientists have long argued about what is more important: having a super smart algorithm or a large amount of data. Everywhere there are difficulties. The above examples are intended to show that a user may not see the difference between an advertisement and a recommendation.

But the real difference is in purpose: a recommendation is given based on: the user's tastes (if the user is active enough), the tastes of other users, and what that person most often searches for. Every day the volume of information, especially on the Internet, is growing very fast. There is an extensive range of information and products such as movies, books, articles, music and so on [3]. The amount of information is growing and finding the information you need is becoming more difficult. Recommendation systems are designed to solve such problems.

1.2 What is recommendation system

According to a user's choices, interests, and behavior, recommendation systems are information systems that offer individualized recommendations. They are widely employed in many different industries, including social networks, streaming services, online commerce, and libraries. The basic objective of recommendation systems is to present the user with the most relevant and captivating items, including products, music, movies, and books. Collaborative filtering, content filtering, hybrid methods, and machine learning are just a few of the different approaches and techniques that can be used to base recommendations. Based on its understanding of the user, the content, and user-content interactions, the recommendation system chooses and presents the user with relevant content. Works are founded on data from personal profiles and other sources. Finding a good book to read before bed or finding a movie to watch on the weekend would be difficult without a recommendation system [3]. Without a recommendation system, we would spend a lot of time looking for our preferred things. Thus, many websites today benefit from recommendation systems to promote and sell their products. The examination of user preferences and how they compare to those of other users forms the basis of collaborative filtering. The system creates profiles for people and entities, then searches for comparable users or entities to present recommendations based on the interests of the comparable users [3].
Analyzing the traits and content of objects is the basis of content-based filtering. Based on similarities to previously rated objects, the system examines object qualities like genre, author, or keywords and presents the user with objects that suit their tastes [3].

Hybrid techniques integrate many strategies to raise the standard of recommendations. To obtain more precise and comprehensive recommendations, they might make use of both collaborative filtering and content filtering [3].
The ability of recommendation systems to uncover intricate patterns and relationships from vast volumes of data in order to more precisely forecast user preferences is a key component of machine learning.
Software solutions called recommendation systems employ data and algorithms to offer consumers individualized recommendations. The goal of recommendation systems is to assist users in sifting through a sea of information to identify the most pertinent and captivating material. They are based on analysis of user data, including past preferences, purchase history, ratings, and reviews, as well as content data, including product descriptions, categories, or attributes [1].
Many industries, including e-commerce, streaming services, social networks, news portals, and others, employ recommendation systems extensively. They contribute to a better user experience, greater user satisfaction and loyalty, higher revenues, more successful marketing initiatives, and better overall business performance.
However, there are a number of difficulties and issues related to the creation and use of recommendation systems. The first step is to effectively gather and retain user and content data, which necessitates taking data privacy and security into account. Second, recommendation engines must be flexible enough to adjust to shifting user preferences and interests as well as contextual elements like time, place, and mood. The capacity to give users justification and explanations for why certain information is being recommended is another crucial factor. This is known as the explainability of recommendations.

1.3 Familiarization with the project

Recommendation systems are crucial in today's information culture as users must search and choose the appropriate information while access to vast volumes of data is getting easier and more common. The online library is one place where recommendations are very crucial. By offering users recommendations that are tailored to their likes and interests, a good recommendation system for these libraries can significantly enhance the user experience.

The purpose of this dissertation is to create a like/dislike rating system and Kazakh-language comments-based recommendation system for an online library. Users of the library will have the option to select their preferred reading language and genre when registering. To make recommendations that are pertinent and assist users in finding new books and authors they may enjoy, the system will make use of these preferences as well as user comments that have been parsed. The potential to enhance user experience by offering individualized recommendations is one of the main advantages of creating such a recommendation system. Because of the overwhelming amount of material available, traditional techniques of finding books in an online library may be constrained, and consumers may find it challenging to identify what truly interests them. This space will be reduced and the user will receive more accurate recommendations that match their tastes and preferences if a recommendation system is based on user preferences and comment analysis.

A study of the usage of machine learning models for assessing user comments and ratings will be done as part of this dissertation, along with a review of the current approaches and techniques used by recommendation systems. To find the best model for making precise and individualized suggestions, a number of models, including Support Vector Machine, Logistic Regression, Naive Bayes, and Random Forest, will be constructed and tested. The creation and deployment of a website using the Django framework and the use of the MySQL database for data storage will also play a significant role in the project. This will offer a useful and trustworthy platform that can process user, book, and comment information as well as their interaction with one another quickly and efficiently.

Numerous machine learning and data analysis methods will be employed in the creation of this system. The algorithm will be able to distinguish between positive and negative reviews by analyzing remarks to identify their emotion. The system will be able to better understand user tastes and interests by creating a profile of each user's preferences using user comments. Additionally, the system will use the data on preferences to gauge the popularity of books and present users with those that have received favorable ratings and reviews from other readers.

Created a database of books with details on the author, genre, description, and reviews in order to carry out this project. Following that, users will be prompted

to choose a language and genre during registration, on the basis of which the initial user profile will be created. The system will then utilize sentiment analysis techniques to examine user comments and determine whether they are good or unfavorable. This will enable the system to consider user preferences and present them with books that suit those choices. To estimate the popularity of books among users, the system will also consider data on likes and dislikes. Books with a high number of likes and favorable reviews will be highlighted, whereas books with low numbers of likes, dislikes, and unfavorable reviews won't be included. User preferences, comment analysis, and like/dislike data will all be combined to create the final suggestions. Users will receive pertinent and individualized recommendations as a result of the system's ongoing updating and adaptation to their evolving preferences and inclinations. As a result, there are several benefits to creating a recommendation system for an online library based on comments and like/dislike systems. By making personalized recommendations and assisting users in finding new authors and publications that appeal to their interests, it will enhance the user experience. Additionally, the recommendation algorithm helps to boost user engagement with the online library. Giving users accurate and pertinent recommendations makes it more likely that they will stay on the platform longer to look up new books and writers. More page views, downloads, and eventually greater user engagement and satisfaction result from this.

There is a lot of room for data gathering and analysis in the construction of such a recommendation system. The system will gather data on reader preferences, comments, and responses to books, which will be analyzed and important data about reader preferences, popular genres, literary trends, etc., will be extracted. This data can be utilized to construct marketing and promotion plans as well as to enhance the book catalog and schedule events. Overall, creating an online library recommendation system based on reviews and like/dislike systems is a crucial step in giving readers a tailored and enjoyable reading experience. Users will be able to quickly identify books that suit their likes and learn about new literary works thanks to this method. Additionally, it can aid in boosting user activity, gathering beneficial data, and develop the library platform.

Additionally, it presents options for enhancing the caliber of recommendations and customizing the system to the tastes of users who speak Kazakh.

The processing of the Kazakh text in the context of sentiment analysis is given particular consideration. To accurately assess the tone in texts written in Kazakh, special algorithms and models are being created due to the language's distinctive structure and peculiarities. As a result, recommendations can be made that are more pertinent to users' tastes and take into consideration their particular culture and language.

In addition to improving personalization of recommendations, the use of Kazakh-language comments in the online library's recommendation system provides opportunities for analyzing and taking into account the cultural, social, and linguistic aspects of users' preferences, to quote Maulen. This enhances user pleasure and engagement in the process of reading and utilizing the online library by enabling a distinctive and personalized user experience.

The primary techniques and algorithms utilized to create a recommendation system will be discussed in the dissertation's subsequent chapters. The findings of research using actual data will be given, and the system's efficiency in making individualized recommendations will be examined. This system's development and implementation will also be examined for any potential drawbacks and issues.

1.4 Significance of the study

1. *User Experience Improvement:* Users frequently struggle to identify the appropriate books because of the exponential growth of online libraries and the volume of content available [3]. The project intends to give personalized recommendations that match users' interests and enhance their overall browsing and reading experience by creating a recommendation system that takes into consideration user preferences such as language and genre selections. The creation of a recommendation system based on sentiment analysis and comments in Kazakh is also crucial for enhancing the user experience on the web platform. Users will be more satisfied and engaged with the platform if it can make accurate, individualized recommendations that take into consideration their emotional responses.

2. *Enhancing User Engagement:* The recommendation system goes beyond

conventional approaches by taking into account user input and interactions by including comments and like/dislike mechanisms. Users can share their ideas and participate in the suggestion process, making for a more dynamic and interactive user experience. As a result, the study hopes to promote greater user happiness and engagement with the online library platform.

3. *Improving the quality of recommendations:* The process of making recommendations is made much more complex by using sentiment analysis. Research can efficiently ascertain the positive or bad features of a book by assessing the sentiment of user comments, assisting in the creation of more accurate and pertinent suggestions. Users will only see offers that fit their interests and feelings thanks to this sentiment-based approach, which also improves the quality of suggestions overall. The quality of recommendations will increase with the use of text sentiment analysis. The ability of conventional recommendation systems to make precise and pertinent recommendations may be constrained. Research will make it possible to more precisely identify the emotional content of user comments and utilize this knowledge to provide recommendations that are more intriguing and relevant.

4. *Application of modern technologies:* The creation of a recommendation system based on the Django programming language, the MySQL database, and text sentiment analysis makes it possible to use cutting-edge tools for data analysis and machine learning. The advancement of text data analysis techniques and their use in recommendation systems will be aided by research.

5. *Potential for commercial applicability:* There is a lot of potential for commercial use in the creation of an effective recommendation system. Online commercial platforms will be able to raise the quality and relevance of their suggestions with the successful installation of a Django and MySQL-based system, which can result in higher sales, user retention, and increased competition.

6. *Comparison of Models and Algorithms:* The development of recommenda-

tion systems and various models and algorithms for sentiment analysis are among the research topics. This is crucial because it makes it possible to identify the best strategies and select the best model for the best outcomes. The growth of research in the areas of sentiment analysis and recommendation systems is also aided by comparisons of models and algorithms, which may be helpful to other academics and professionals working in this area.

7. *Innovative approach:* An inventive method for creating a recommendation system based on comments made in Kazakh and text sentiment analysis is shown in the research. Making more contextual and individualized recommendations is possible by using information from user comments and accounting for their emotional responses. Such a strategy may make a substantial impact on the field of recommendation systems and catch the interest of academics and industry professionals.

8. *Applicability in various fields:* The findings of the study on the creation of a recommendation system based on text sentiment analysis offer a broad range of possible applications in numerous fields. For online shops, video streaming, social networks, and other platforms where giving consumers useful and engaging content is a top priority, the capacity to customize and enhance the quality of recommendations can be valuable.

In general, research is quite important and can help with practical things like improving user experience, developing recommendation systems, and using cutting-edge technologies for text data analysis [4]. The findings and recommendations of the study can serve as the foundation for additional investigation and advancement in this field.

1.5 Aim and objectives of research

The purpose of this dissertation is to create a user-commented, like/dislike-based recommendation system for an online library. The major objective is to offer customers individualized recommendations based on their preferred reading language and genre.

Research objectives:

The following tasks will be completed in order to reach this objective:

- *Study of existing recommendation methods and algorithms:* The most acceptable and efficient ways for creating a recommendation system will be determined by reviewing the currently used recommendation methods and algorithms in the context of online libraries.
- *Data collection and analysis:* The online library will collect and analyze information on user preferences, comments, and likes and dislikes. To build recommendation models and gauge the tone of comments, this data will be used.
- *Development of recommendation algorithms:* The development and implementation of recommendation algorithms based on the evaluation of user comments in the Kazakh language and the like/dislike system. The user choices they set upon registration, such as language and reading genre, will be taken into account by the algorithms [5].
- *Algorithm Evaluation and Comparison:* The effectiveness and accuracy of various recommendation algorithms will be measured and compared. This will establish the most efficient algorithm for the online library's personalized recommendation system.
- *Development and integration of the system into the online library:* A recommendation system will be created and effectively integrated into the current online library based on the findings. Users will benefit from more individualized recommendations as a result of this study based on their language and reading genre preferences.
- *System Testing and Evaluation:* Real user data will be used to test the created recommendation system and assess its effectiveness, precision, and recommendation quality. We'll gather user reviews and ratings to make the system even better.
- *Integration of the system into the web platform:* To ensure practical application and accessibility for users, the recommendation system will be integrated into the already-existing web platform of the online library. The

appropriate configurations will be made, and the current infrastructure will be integrated.

- *User Experience Evaluation:* The recommendations given will be the subject of a user experience study to gather opinions and rankings from users. This will make it easier to assess user satisfaction and find areas where the system may be improved.
- *Analysis of the results and conclusions:* The analysis and interpretation of the study's findings will be done in light of the findings and user comments. On the effectiveness and practical usability of the created recommendation system for the online library, conclusions will be reached.

The overall goal of the project is to create and implement a user-commented, like/dislike-based recommendation system for an online library. The study of existing techniques, gathering and analyzing data, developing algorithms, testing, integrating into a web platform, and assessing the system are some of the goals of the research. The research's findings will be helpful for enhancing the quality and personalization of recommendations in the online library.
The tasks for developing a recommendation system for an online library were discussed above, but just the primary activities were included. There are little things that can assist in resolving these key issues or, more generally, in improving the suggestion process.

Chapter 2

Related work

The use of sentiment analysis techniques and machine learning models in recommendation systems for online libraries has been the subject of extensive research. This section will discuss some of the most important pieces of research on the subject.

2.1 Recommendation system

This article [6] provides a summary of the state-of-the-art in the subject of recommender systems and examines potential upgrades and additions to current approaches. The authors begin the article by providing a summary of the key ideas and responsibilities involved with recommender systems, including information gathering and analysis, modeling of user preferences, and suggestion creation. Additionally, they outline the primary categories of recommender systems, including collaborative filtering, content filtering, and hybrid methods. The authors then give a brief summary of the current approaches and algorithms in each of these fields. They give examples of how each method can be used on actual systems, explain how it functions, and discuss its benefits and drawbacks. The writers also go through numerous metrics and techniques for measuring the quality of recommendations. Then, Adomavicius and Tuzhilin think about potential upgrades and additions to the techniques used by current recommender systems. They examine issues like data scarcity, cold starts, concealed information, and context, and they

provide solutions. The use of extra information sources, such as social networks and contextual data, is also covered by the authors as a way to raise the standard of recommendations. The authors conclude the review and suggest future lines of inquiry in the area of recommender systems at the conclusion of the article.

In this work [7], the authors introduce a brand-new method for collaborative filtering that is based on an examination of how closely related objects (or products) are in the recommendation engine. In the first section of this paper, the writers discuss the problem of recommendations in the context of an expanding body of knowledge and a range of user preferences. They clarify that, especially when there are many users and sparse data, the traditional collaborative filtering strategy based on user similarity analysis has significant drawbacks. The concept and guiding principles of the authors' object-based collaborative filtering approach are then presented. They suggest generating recommendations based on the resemblance between objects, which is determined using past user preferences. They compare objects based on how frequently they have received ratings from the same people rather than simply comparing users. They outline the steps involved in determining item similarity, choosing the most comparable objects, and producing user recommendations. The authors compare their strategy with existing collaborative filtering techniques while also conducting trials on actual data. The findings demonstrate that the object-based method is effective and accurate at recommending objects to the user.

This article [8] provides an overview of these systems as well as some experimental results. In order to enhance the quality of recommendations, hybrid recommender systems combine several methodologies and strategies. The article's introduction gives a summary of the primary issues and difficulties with recommender systems, including cold start, data scarcity, and the relevance issue. He shows that hybrid recommender systems can get over these problems by fusing several strategies and methods to produce recommendations that are more precise and diversified. Burke continues by giving a general overview of the several strategies utilized in hybrid recommender systems, such as collaborative filtering, content filtering, domain knowledge, and context. He explains each strategy's operation and provides instances of its application in actual systems. The author

also does an experiment to measure how well hybrid recommender systems work. It outlines the experimental process, including dataset selection, measures for gauging data quality, and comparisons with alternative approaches. According to experimental findings, hybrid systems are able to produce recommendations that are much better than single-device approaches. Burke summarizes the experiments and review at the end of the essay and talks about the benefits and drawbacks of hybrid recommender systems. He also identifies areas where study is still needed, such as ways to better combine approaches and make recommendations that are tailored to the needs of certain users.

The authors of this article [9] introduce a deep learning-based platform for book recommendations. In the opening section of this paper, the writers discuss the problem of book suggestions in light of the expanding volume of books and the variety of user interests. They describe the drawbacks of conventional recommendation techniques like collaborative filtering and content filtering and propose the use of deep learning to produce recommendations that are more precise. The architecture and parts of the authors' book suggestion platform are then explained. They describe how they extracted elements from book metadata, including title, author, genre, description, and ratings, using neural networks, and then used those features to produce suggestions based on user interests. They detail the data collection and preprocessing steps they took, as well as the trials they carried out to gauge the efficacy and performance of their book recommendation algorithm. According to experimental findings, the deep learning technology can provide users with individualized book recommendations and has a high degree of accuracy. It offers a variety of recommendations while also taking into account user preferences and interests. The authors provide a summary of their findings and explore potential directions for the platform for book recommendations at the conclusion of the article.

In order to increase the precision and transparency of recommendations, this article [10] offers an explanation-based recommender system. The suggested system examines customer preferences and product attributes using machine learning techniques and algorithms. Based on details about customer preferences, product features, and purchasing context, it creates justifications for why a specific

product is suggested. Users can better grasp recommendations and make knowledgeable purchasing selections thanks to these explanations. The system's design and functioning, including data collecting and processing, user preference analysis, explanation generation, and suggestion giving, are described in the article. The incorporation of explanations in a recommender system, according to experimental findings, considerably raises user satisfaction and enables them to make more educated selections while making purchases from an online retailer. The post is useful for e-shop developers who want to raise the standard of recommendations and give their clients a better shopping experience.

2.2 Sentiment analysis

This article [11] proposes a text review emotional categorization model based on the topical and tonal qualities. The use of natural language processing and recommender systems is highly demanded for the sentiment analysis of textual reviews. The author summarizes the findings of the study of text review themes and emotional tone. It offers a paradigm that considers and evaluates both the text's theme and tone. The model blocks text reviews on emotional categories using machine learning techniques like support vector machines (SVM). In the essay, the selection of characteristics for analysis and the start of the model generation process are discussed. The author tests several feedback datasets and assesses how well the models perform on on the metrics used, such as accuracy, recall, and F-score. The study's findings demonstrate that the model, which accounts for topical traits and sentiments, performs particularly well when it comes to text review emotional filtering. The essay touches on significant facets of themes and sentiment while presenting intriguing research findings and a novel method for the emotional filtering of textual evaluations. In the area of textual emotion analysis and the development of recommender systems, where the emotional component plays a significant role in personalized manifestations, it can be helpful for observation and involvement.

In this article [12], machine learning-based text sentiment analysis techniques are compared. Several well-known machine learning-based text sentiment anal-

ysis techniques are presented and contrasted by the authors, including logistic regression, support vector machine (SVM), naive Bayes, and random forest. The authors evaluate the benefits and drawbacks of each technique and make judgments regarding its efficacy in determining the tone of the text. The study's findings demonstrate that, depending on the text's kind and the task's context, several text sentiment analysis techniques have advantages and disadvantages. For instance, Naive Bayes may be better suited for some particular text kinds whereas Support Vector Machine (SVM) works well in the overall case. The paper offers insightful comparisons of various machine learning-based approaches to text sentiment analysis. It can be helpful for text sentiment analysis researchers and developers as well as applications in recommender systems, where precise sentiment analysis is a crucial component for producing pertinent and individualized recommendations.

The following article [13] represents a significant contribution to the field of sentiment analysis. Liu gives a thorough explanation of the approaches and procedures employed in this book to analyze and draw out sentiment and emotion from textual data. The author also takes into account related activities like sentiment classification, emotion extraction, and figuring out a text's subjectivity. Liu continues by introducing several sentiment analysis techniques, such as lexical methods, machine learning, rule-based models, emotion analysis techniques, and others. He outlines the fundamental ideas behind each strategy and offers illustrations of how each can be used in practical situations. In addition to discussing the future directions for study and development in the field of sentiment analysis, Liu presents remedies and methods to these issues. For scholars, students, and practitioners interested in sentiment analysis, "Sentiment Analysis: Mining Opinions, Sentiments, and Emotions" is a helpful reference. The book offers a thorough overview of the key approaches, methodologies, and difficulties in this field and can be used as a resource for information and direction for creating and implementing sentiment analysis models and algorithms in a variety of contexts.

One of the most often referenced papers in the area of opinion and sentiment analysis is the paper [14]. The main approaches and procedures for assessing opinions and sentiments are reviewed in this article. The authors discuss the

essential tasks, such as figuring out the sentiment of the text, extracting opinions, and classifying sentiment, before providing an overview of the vocabulary used in the analysis of opinions and feelings. The key elements of these strategies are described in depth, including the use of dictionaries, statistical techniques, machine learning algorithms, and semantic text processing. The authors also go over the issues and difficulties that arise when analyzing opinions and sentiments, including ambiguity, context dependence, and the handling of unstructured data. They also suggest possible research areas and offer viewpoints on how the field can progress in the future. For scholars and practitioners interested in the analysis of opinions and sentiments, the paper by Pang and Lee is a useful resource. It offers a summary of the key techniques and strategies and poses vital queries to think about when creating solutions in this field.

The article [15] discusses a study that employs genre-based recommender systems for streaming services along with hybrid deep learning models for sentiment analysis. The writers start the piece by discussing the issue of suggestions in the context of streaming services, where users have access to a lot of content. To ensure user pleasure, they underline how critical accuracy and customization of recommendations are. The authors then suggest a hybrid strategy that uses deep learning models to evaluate genres and sentiment within the subject matter. While genres help to account for user preferences for particular sorts of material, sentiment analysis aims to understand users' emotional views regarding information. The hybrid deep learning models employed in the study are thoroughly described by the authors. They describe data processing, employing embeddings to show material, and building models to forecast sentiment and genres. The authors then run experiments on actual streaming service data to gauge the viability of their strategy. They compare the outcomes with other approaches and utilize a variety of criteria, such as accuracy and coverage, to assess the quality of the recommendations. According to experimental findings, streaming services' recommendations can be made more accurate by including sentiment and genre data into hybrid deep learning models. More relevant and individualized recommendations are given to users.

Studies on related work in sentiment analysis and recommendation systems

have indicated that the majority of studies concentrate on the usage of rating data. It's vital to remember, nevertheless, that not enough research has been done employing Kazakh-language remarks in the context of the Kazakh language. The lack of resources and data in Kazakh is one of the difficulties scholars encounter. Sentiment analysis and recommendation systems created for other languages do not always take into consideration the particular and distinctive characteristics of the Kazakh language.
In addition, some publications investigate the use of sentiment analysis and recommendation systems in other languages, which may be helpful in creating systems for the Kazakh language. The major techniques and methodologies for sentiment analysis based on text data are covered in the paper "Sentiment Analysis and Opinion Mining" (Pang, Bo, and Lillian Lee, 2008).

However, greater focus should be placed on gathering and annotating Kazakh-language comments and data in order to advance research in the fields of recommendation systems and sentiment analysis in the Kazakh language. This will enable the development of more precise and effective models that take into consideration the unique characteristics of the Kazakh language, as well as enhance sentiment analysis and suggestions for Kazakh-speaking users.

Chapter 3

Method and Materials

3.1 Creating a website with Django

Python-based Django is an open source web application framework. A framework is a collection of elements that makes it simple and quick to create websites. Django offers nearly everything developers need natively and adheres to a "all-inclusive" ethos. Everything we require is included in a single "product," which means that everything is flawlessly compatible, adheres to a single set of design principles, and has thorough, current documentation. Django offers a framework that is intended to "do the right thing" to automatically protect websites, which helps developers avoid many typical security problems. Django, for instance, avoids typical mistakes like saving immediately and offers a secure mechanism to manage user accounts and passwords. It also maintains session information in secure cookies (cookies only hold keys; the actual data is kept in a database). instead than using a password hash. The name of the website that will be developed and created in this post is Local Library Platform. Create a library first, where people can only see the books that are currently accessible. Exploring operations that are practically everywhere is possible in this approach. reading from databases and presenting the data. [16].
The website of the neighborhood library is the project's name because I want to provide at least the bare minimum of information. As a result, information about books, book copies, authors, likes and dislikes, comments, and other crucial

details will be stored. Will not, however, keep additional data that the library might find valuable, nor will it build a sizable infrastructure to handle numerous library sites or other "large library" features. This will make for an easy testing location [16].

This section outlines the steps involved in building a website using Django and its primary models, views, and templates. (see figure 3.1)

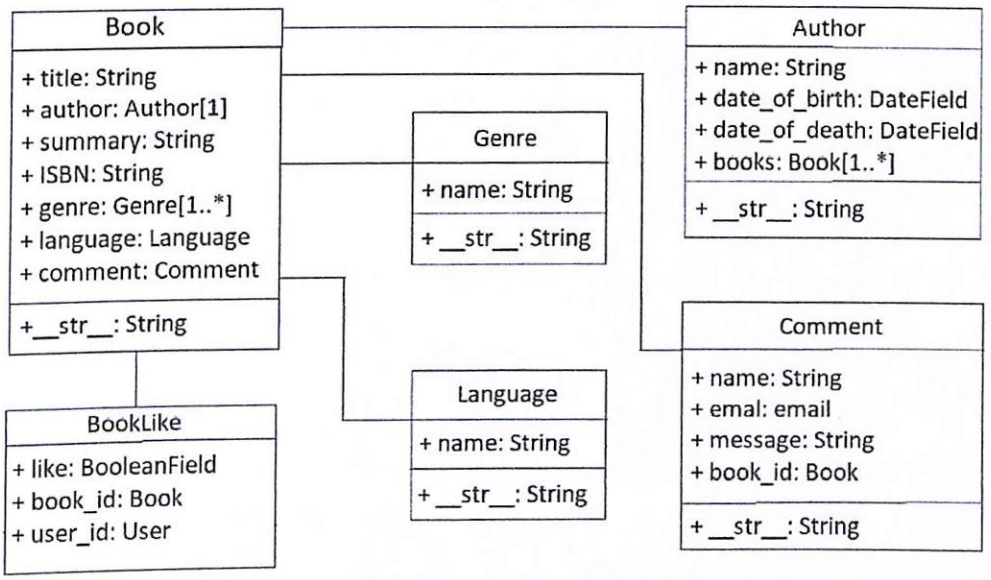

Figure 3.1: An UML diagram of the website

1. Models:
   Website uses the following data models:
   Model "Author": Describes information about the author of the book, such as first name, last name, and biography.
   Genre Model: Contains information about different book genres such as fantasy, romance, detective story and so on in the Kazakh language.
   Book Title Model: Contains information about books, including title, author, genre, and language.
   Model "Language": Contains information about available languages of books.
   Model "Comment": Allows users to leave comments on books.
   Cart Model: Allows users to add books to their shopping cart for later reading.
   Like/Dislike Model: Allows users to express their position on books by liking or disliking.

Book Search Model: Allows users to search for books by title.

2. Performances:
   The web application uses the following views:
   Book List View: Shows a list of books that are available along with basic details like title, author, and genre.
   Book Details View: Details about the chosen book, including its author, genre, and reader comments, are displayed.
   Cart View: shows the items in the user's shopping basket and gives them the option to add or remove books.
   Like/Dislike View: Lets users rate books they like or dislike in order to express their choices.
   Book Search View: Handles user queries and shows book title search results.

3. Templates:
   Web pages are rendered using Django templates, which specify the page's layout and structure. Use the following template files on your website:
   Template "base.html": All other templates inherit their basic functionality from this template. includes standard design components such the page header, menu, and footer.
   Template "book_list.html": Lists the available books along with some basic information about each. includes options for language and genre-specific sorting and filtering of books.
   Template "book_details.html": Displays comprehensive details on the chosen book, including a description, the author's bio, the book's genre, and user reviews. enables readers to review and comment on the book.
   Template "cart.html": Shows the books that are currently in the user's basket, along with the number of books that have been added. allows you to adjust the quantity of books.
   Template "like_dislike.html": Lets users rate books they like or dislike in order to express their choices. shows the current percentage of each book's likes and dislikes.
   Template "search.html": Displays a search form allowing users to enter a book's title. displays any search results.

4. Methods and libraries:
   The following techniques and libraries are employed to implement the functionality of the website:
   Django module: Provides pre-built tools and capabilities for creating online applications, including database interaction, form processing, and URL routing.
   SQL database: Used to hold information on writers, genres, books, comments, and other system entities.
   Sentiment Analysis Python Libraries: May do sentiment analysis on user comments using a variety of libraries, such scikit-learn or VaderSentiment. Scikit-Learn library will be used in study.
   Django template tags and filters: Permit controlling the logic and formatting of data in templates, including how dates and times are shown and how text is formatted, among other things.
   In order to implement the functionality of a recommendation system for an online library, websites can be built utilizing Django and its components, including models, views, and templates.

5. Development of functionality:
   The functionality of the recommendation system will be enhanced when the Django-based website is developed. Several options include:
   Recommendations based on user preferences: Use user preference data, such as likes and dislikes, to recommend books to users based on their interests.
   Improving the Sentiment Analysis Algorithm: Investigate alternative sentiment analysis techniques and models to increase the precision and dependability of recognizing favorable and unfavorable user comments.
   Integration of social features: Users will eventually be able to remark on and score the comments of other users as well as share their own suggestions and opinions with other users.
   Performance Improvement: To guarantee that the website is quick and responsive, optimize database queries, use caching, and other optimization strategies.

All of these enhancements will be directed toward raising user satisfaction, en-

hancing recommendation system efficiency, and enhancing recommendation quality.

Therefore, the creation of the website using Django and its continuous enhancement with the functionality of the recommendation system will allow customers of the online library an effective and customized selection of books.

3.2 Database

One of the most well-liked relational databases for storing and managing structured data is the MySQL database. The MySQL database is crucial to the thesis because it stores data about the online library, including information about authors, genres, books, commentaries, and other system components.

Database structure:

A collection of tables that are arranged logically like an online library make up the MySQL database. Each table represents a particular data type and has columns that describe its structure and properties. These are the database's contained tables: (see figure 3.2)

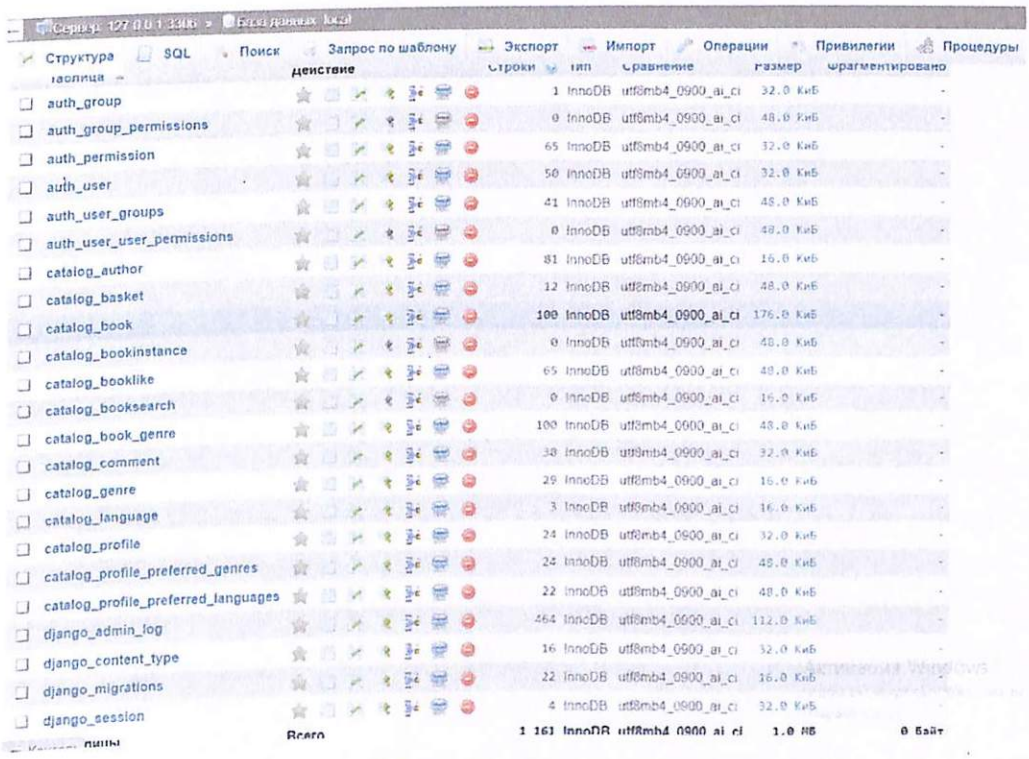

Figure 3.2: List of tables in the database

The database contains 23 tables in total. Let's go over the primary tables that the recommendation will use.
Table "catalog_author" (Authors): Contains details about the writers, including their ID, first and last names, birth dates, and other characteristics.
Table "catalog_book_genre" (Genres): contains data on several book subgenres, including detective, romance, and fantasy.
Table "catalog_book" (Books): contains information on each book, such as its special identification number, title, description, author, genre, and other relevant characteristics.
Table "catalog_language" (languages): contains information on languages. Because the user chooses the language of the book he prefers to read when registering, the language interacts with the profile.
"catalog_profile" table (User profile): The user table and this table are connected. includes user information, such desired genre and language.
There are also tables "catalog_profile_preferred_genres" and "catalog_profile_prefer they are linked to the table "catalog_profile".
Table "catalog_comment" (Comments): Database that keeps track of user comments made on a specific book. Contains the time, the comment's content, and the ID of the user or book it relates to.
Table "catalog_booklike" (likes/dislikes): includes information on reader preferences for novels. includes user, book, and preference state (like, dislike) information.
Table "catalog_booksearch" (Search): Contains information such as the book titles that searchers have looked up.

Database operations:

In order to interface with the database, MySQL offers a strong set of operations. Will employ the following operations in the context of the dissertation:

- *Creating tables:* The required database tables were created by specifying their structure (columns and data types) and connections (primary and foreign keys).
- *Data Insertion:* Tables can have additional records added via data insertion

techniques. Then, to add information to the database, insert information about authors, book genres, comments, and user preferences into the appropriate tables.

- *Data Update:* When information in an online library changes, such as when new books are added or users post comments, data update operations are performed to update the database to reflect the changes.
- *Data Retrieval:* Data retrieval operations enable the selection of data from a database based on predetermined criteria. For instance, we can retrieve all books in a particular category or comments exclusively for a specific book. The retrieved data can be utilized for statistical analysis, performing operations relevant to the functionality of the recommendation system, and displaying recommendations to users.

Django integration with MySQL database:
Working with a MySQL database is made simple by Django. Utilize Django's ORM (Object-Relational Mapping) to build data models that correspond to the layout of the database's tables. Django will automatically construct the necessary database tables and use the models to interact with the data.
Django's ORM queries can be used to carry out database operations such as creating, inserting, updating, and retrieving data. This makes working with the database easier and guarantees the data's security and integrity.
To ensure the effective operation of the website and the recommendation system, optimization is a crucial component of working with the MySQL database. The usage of indexes on frequently used columns, selecting the appropriate data types for information storage, separating data into distinct tables to increase query efficiency, and other optimization strategies are all examples of optimization.

3.3 Data Collection

The practice of gathering information for use in research or other objectives is known as data collection. In addition to information about user preferences and actions, it also covers the collecting of numerous sorts of data, including words, numbers, photos, audio, and video recordings. The development of a recommen-

dation system for an online library depends heavily on data collection, which is a crucial step in research. It involves gathering, setting up, and getting ready the data needed for modeling and analysis. The process of gathering data for research is crucial, and there are many different ways to do so [3]. Data was gathered for the dissertation from a variety of sources.

Utilizing bots to gather comments in Kazakh was one of the data collection techniques used. The bots generated comments, which were then subjected to sentiment analysis techniques for examination and evaluation. This method made it possible to gather a significant amount of textual information in the Kazakh language, which can be used to gauge readers' positive and negative responses to books.

Used information gathered from various libraries and bookstores as well. Information on books, authors, genres, and other characteristics that can be used to create recommendations are included in this. This information offers a wealth of details on the books that are available on the market and user preferences.

Making sure data is accurate and of high quality is one component of data collection. We must ensure that the data is accurately gathered, current, and presented in a way that makes it simple to examine. The ethical considerations of data gathering, such as getting user consent and protecting privacy, must also be taken into account [17].

Benefits of data collection:

Rich and Diverse Dataset: A diversified dataset is produced via data collection techniques like utilizing bots to generate comments and gathering information from physical sources. As a result of this diversity, the recommendation system is more dependable and accurate because it can take into account a wider range of user interests and viewpoints.

Real World Relevance: The final data set reflects actual world scenarios and user behavior because it was compiled from reliable sources like bookstores and libraries. By doing so, the recommendation system is guaranteed to reflect the actual tastes and interests of users, enhancing its usefulness.

Personalization: The creation of personalized suggestions is made possible by the consent-based collecting of user personal information such names and email

addresses. Individual user preferences enable the system to recommend the most pertinent publications, enhancing user engagement and enjoyment.
Scalability: Large amounts of information can be handled by scaling up data collection techniques. In the case of an online library, where a sizable amount of data must be processed and analyzed in order to produce useful recommendations, this is particularly crucial.

Disadvantages of data collection:
Effort and cost: Data collection can be labor-intensive and expensive in terms of both time and money. For instance, gathering information from physical sources like bookstores and libraries necessitates traveling to several places, contacting with employees, and planning the information collection. It can take a lot of money and time to do this.
Data quality: There are dangers related to the quality of the data when it is collected utilizing various techniques. For instance, when utilizing bots to create comments in Kazakh, false or distorted data may be produced, which could impede the effectiveness and precision of the recommendations [17].
Restricted access to data: There might be restrictions on access to data in some circumstances, particularly when using data from outside sources. This may reduce the volume and diversity of data that can be gathered and reduce the study's overall efficacy.

Let's now discuss the precise data collection procedure employed for the dissertation. used a number of techniques, such as bots, to produce Kazakh-language comments and gather data from bookstores and libraries. With the use of these techniques, information on books, authors, genres, and user preferences as well as roughly 330 comments in the Kazakh language were gathered.
This method of data gathering makes it possible to gather a wide range of current data. The preferences and viewpoints of the Kazakh audience are reflected in comments, which is crucial for creating advice that is suitable for this particular cultural setting. The gathering of information from physical sources enhances the information found online and offers a more comprehensive picture of the book market. Respect for privacy and ethical standards is also a part of the data collection process. The use of consents and collecting consents from users ensures

that they are in agreement with the study's use of their personal data. This emphasizes dedication to privacy protection and assuring moral data processing.
It is crucial to remember that data collecting must adhere to all applicable legal and ethical requirements. We have taken precautions to ensure that data gathering procedures respect users' privacy and uphold the authorship rights of books and other publications. To guarantee their anonymity and the safety of personal information, it is also crucial to apply the proper data depersonalization and anonymization techniques.
The gathered information—comments, details on books, authors, genres, and user preferences—is a useful tool for creating a recommendation engine. This information enables analysis and understanding of the preferences and tastes of users, allowing for the recommendation of the most appropriate books for them. The recommendation system we are developing can be used in online libraries to give customers individualized book recommendations, which is another practical application of the dissertation. This can enhance the reading experience, make it simpler to choose and find books, and advance reading culture in general.
In conclusion, gathering data for a thesis is a crucial stage that creates a strong foundation for the creation of a recommendation system. The variety, relevance, and customisation required to produce high-quality suggestions are ensured by the data collection techniques used. In addition, I consider moral and legal considerations, which ensure that consumers' privacy and their data are protected.

3.4 Data analysis

Data analysis is a crucial component of research since it offers insightful knowledge for comprehending and analyzing the gathered data. To extract meaningful results and reach wise conclusions, it requires the processing, visualization, and statistical interpretation of data [4]. Data analysis is crucial for comprehending user comments, gauging their attitude, and assessing preferences in the context of the thesis. used the scikit-learn library, which offers strong capabilities for machine learning and data analysis.
One of the methods used is *TfidfVectorizer()* and *CountVectorizer()*. *TfidfVectorizer()* enables the use of the term-frequency-inverse document frequency approach

to transform text data to numeric vectors. By considering the significance of terms in a document and their relative impact on the overall collection of documents, this aids in building prediction models. *CountVectorizer()*, nevertheless, transforms text data into vectors based on a count of the frequency at which each term occurs. This makes it possible to assess the distribution of words and pinpoint the most prevalent or uncommon keywords.

In order to compare the distribution of terms in favorable and negative evaluations, a histogram of comments was also produced. This makes the distinctions in how specific words or phrases are used more clear and can shed more light on the sentiment behind comments.

Analyzed the ratio of favorable to unfavorable reviews as well. This makes it possible to gauge the general level of customer satisfaction and identify the most common remark categories.

A dissertation's data analysis is crucial to developing the recommendation system for an online library. It aids in determining user preferences, comprehending their reviews and remarks, and producing customised recommendations in light of this data. Data analysis enables intelligent inferences and data analysis optimization. Thesis also heavily relies on the optimization of data analysis. used a variety of strategies and procedures to enhance the analysis's efficiency and precision. Data preparation is one of these methods. To normalize text data, preprocessing techniques such data denoising, stopword removal, lemmatization, and stemming are used. This makes it possible to enhance the prediction's accuracy and get rid of extraneous data that can influence the outcome. The choice and fine-tuning of machine learning algorithms is a further component of optimization. implemented techniques like Naive Bayes, Support Vector Machine (SVM), Logistic Regression, or Random Forest. And tried out several algorithms and adjusted their settings to get the best outcomes.

The effectiveness and optimization of databases should also be taken into account. MySQL was used to store the data, and tables, indexes, and queries were optimized to ensure quick data retrieval and updating. can be used to increase query speed and overall database performance through the use of techniques like indexing, data partitioning, and caching. Thesis plays a significant role in the creation of a recommendation system for an online library. Understanding user

preferences, identifying their ratings and comments, and basing suggestions intelligently on this data are all made possible by data analysis.

The quality of models can be evaluated, and experiments can be run, as part of data analysis for theses in order to enhance the recommendation system. The following factors need to be taken into account:
Model quality assessment: Used a variety of criteria to assess the performance of the models, including accuracy, recall, precision, and F1 score. Finding the best algorithms and methodologies will be possible by comparing metrics across many models.
Experiments and Improvement of Models: To find the ideal model configuration, researchers tested with a range of methodologies, parameters, and algorithms. This could entail adjusting model hyperparameters, employing ensemble techniques, or adding further characteristics to enhance predictions.
Model Validation: It's crucial to test models on a different data set (the validation set) from the one used for model training. This makes it possible to assess how well the model generalizes and predicts using new data.
Further directions of research: It is possible to offer further study directions after the data has been analyzed and a recommendation system has been developed.

The analysis of the data in the thesis seeks to give website users more precise and individualized recommendations. This enhances user pleasure, user experience, and the effectiveness of the recommendation system.
Create a chart showing the ratio of positive to negative comments after first separating the positive and negative comments. Start pre-processing after that. Data preparation for sentiment analysis comes after data analysis. (see figure 3.3)

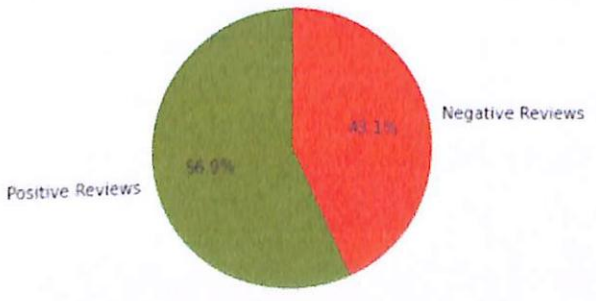

Figure 3.3: Chart of the ratio of positive and negative reviews

3.5 Pre-processing

In text data analysis, particularly sentiment analysis, pre-processing is a crucial step. It covers several operations designed to prepare text data for subsequent analysis by cleaning and converting it. Several methods were employed in the algorithm to preprocess the input, including removing punctuation, changing the text's case, and utilizing the *CountVectorizer()* and *TfidfVectorizer()* to turn the text into a vector form. (see pseudocode Listing 3.1)

Listing 3.1: Pseudocode for preprocessing Comments

```
Set an empty list called processed_data

For each comment in the comments list:
    Remove punctuation marks from the comment
    Convert the comment to lowercase
    Add the processed comment to the processed_data list
```

Removing punctuation: Code uses the regular expression "`re.sub(r'[^\w\s]', '', comment)`" to edit comments to remove any punctuation. This enables the removal of superfluous symbols that have no semantic meaning and may skew the analysis's findings.
Lowercase: Code utilizes *comment.lower()* to lowercase every word in order to preserve uniformity and eliminate case differences. This enables the same-character but different-case terms to be regarded as synonymous.
Vector Representation Conversion: Use the *CountVectorizer()* and *TfidfVectorizer()* methods to convert text data to vector representation. The *CountVectorizer()* converts each comment into a vector containing the frequency counts of each word in the comment. *TfidfVectorizer()* transforms comments using the TF-IDF (Term Frequency-Inverse Document Frequency) approach, It considers the word's relevance within the context of the full remark corpus as well as its frequency within the comment.
Feature scaling: The StandardScaler is used by code to apply feature standardization and normalize the values of feature vectors. This equalizes the features and lessens the influence of outliers on the analysis's findings.

The effectiveness and precision of sentiment analysis are enhanced by the use of various pre-processing techniques. They assist in bringing the data to an easy-to-analyze view by removing extraneous information and reducing noise [18]. These are crucial actions to take in order to succeed in the process of analyzing the sentiment of comments.

3.6 Sentiment analysis

Sentiment analysis is a text analysis technique used to identify the emotional undertone, tone, or mood that the text is expressing. It is employed to ascertain whether text data expressed a positive or negative emotional value. Numerous fields, such as social media, marketing, product evaluations, political analysis, and others, might benefit from sentiment analysis. The capability of sentiment analysis to automatically process massive amounts of textual data and extract emotional information is one of its main advantages [19]. Companies and organizations can use this to assess user attitudes and opinions from a variety of sources, including social networks, forums, blogs, etc. Determine public perception of a good, service, or event using sentiment analysis to make better business decisions. Sentiment analysis does, however, have several restrictions and drawbacks. The vagueness and difficulty in reading emotions in the text is one of the primary issues. It might be challenging to determine the exact tone, particularly when there is sarcasm, irony, ambiguity, or poor grammar. Additionally, sentiment analysis can be sensitive to context and culture, thus outcomes may differ across languages and cultures.

In order to assess the tone of user comments in an online library, sentiment analysis is crucial in the context of dissertations. This makes it possible to categorize comments as positive or negative automatically, which is helpful for producing recommendations and directing users to the most pertinent books.

However, it's also critical to focus on the moral implications of sentiment analysis, such as safeguarding user data privacy and exercising caution when using the analysis's findings to inform judgments [19]. These factors need to be considered and investigated in the dissertation. Overall, research on sentiment analysis offers applications for enhancing an online library's recommendation system. Modern

machine learning techniques and technologies are used to assess comment sentiment and categorize it into positive and negative sentiment. Additionally, the potential for furthering the development of sentiment analysis as well as the significance of include new settings and ethical considerations in dissertations are discussed. We'll also look into the phrases that have the biggest impact on model predictions.

The initial stage in training sentiment analysis models is to create a list with data that indicates the emotional tone of remarks, with positive comments denoted by (+) and negative comments denoted by (-). The labeled dataset, which consists of this list, serves as the foundation for developing and testing machine learning models. The labeled data set consists of a number of text comments, each of which has a distinct emotional tone (either positive or negative). With the help of this dataset, sentiment analysis algorithms are trained to "understand" and "learn" to detect the emotional undertone of text. Give the model clear cues as to the emotional tone of each comment as you mark up the data. This aids the model's ability to connect specific textual elements with particular emotional categories. For instance, the model would typically credit a comment's positive emotional tone if it recognizes that it contains positive words, phrases, or expressions.

Because they give the model context for how to discern between positive and negative texts, labeled datasets are an important tool for training sentiment analysis models. But it's crucial to keep in mind that the model's output might be significantly impacted by the quality of the data markup. To maintain the quality and robustness of the trained model, it is crucial to label data consistently and objectively. You can utilize a labeled dataset you've created to develop sentiment analysis models. This information will be used by machine learning models, such as classifiers or neural networks, to extract textual properties and create models that can automatically categorize fresh texts according to their emotional tone.

It's crucial to keep in mind that using and producing a labeled dataset is just one step in the training of sentiment analysis models [20]. When developing sentiment analysis models, there are additional factors to take into account in addition to the labeled information.

Data pretreatment, which was discussed in Data analysis, is one of the crucial elements. The texts must be pre-processed in order to standardize their format

and get rid of any potential noise or extraneous information before training the models. Punctuation, stop words, and word normalization (such as making all words lower case) may all be part of this process. The quality of models is increased and noise elements that can skew analytical results are reduced by data pretreatment. Additionally, picking the right model or algorithm for training is crucial. The sentiment analysis problem can be solved using a variety of machine learning techniques, such as linear models, decision trees, naive Bayes classifiers, support vector machines, and neural networks. Each has advantages and disadvantages of its own, and the best model to choose will rely on the particular needs and qualities of the data. The emotional sentiment of comments was examined using machine learning techniques and classification models in the dissertation Sentiment Analysis of Comments in an Online Library. Additionally noted areas for improvement, such as accounting for context and the semantic relationship between words. The size and caliber of the labeled dataset should also be taken into account. The model can learn and adapt to new texts more effectively the more data we have. To help the model learn to distinguish between many facets of emotional tone, it is crucial to make sure that the data is diverse. Models that lack variation in data may be unreliable or unstable.

To determine the model's performance and quality, evaluation and testing are also important in addition to training the model [20]. Utilized criteria like accuracy, recall, F1-score, and precision to achieve this. The model can be evaluated to see if it can accurately categorize texts based on their emotional tone and to compare various models or techniques.

Finally, it's critical to remember that the sentiment analysis model will need to be updated and maintained over time [21]. A sentiment analysis model must be able to adjust to new linguistic features, expressions, and circumstances because language and emotion are both continually changing. This can entail retraining the model with fresh datasets and routinely updating labeled data. It is also important to note some of the difficulties and restrictions associated with developing sentiment analysis models. First of all, data markup can take a while, especially when dealing with a lot of text. To carefully label the data and produce a quality labeled dataset, it takes some time and work. Second, subjectivity is a potential issue with sentiment analysis. Different readers may interpret the text's emotional

tone differently, and some comments may contain a variety of tones or phrases that are difficult to categorize clearly. Due to this, sentiment analysis is uncertain to some extent. Additionally, generalization issues with sentiment analysis models can arise. It's possible that the model will perform well with the data it was trained on but poorly with brand-new, ambiguous data. The overfitting problem necessitates meticulous model control and regularization to avoid overfitting and guarantee its generalizability.

Also keep in mind that depending on the language and culture used, the sentiment analysis results could vary. Languages and cultures may use different concepts and idioms to describe different emotional states. Because of this, it's crucial to consider these linguistic and cultural variances while creating sentiment analysis models.

In conclusion, using labeled datasets to train sentiment analysis models is a crucial step in creating effective and precise algorithms. However, it is crucial to take into account data preprocessing, choosing an appropriate model, evaluating the model, as well as subjectivity, generalization, and linguistic and cultural dependencies difficulties.

The examination of a text's emotional tone and the determination of users' thoughts and sentiments are both possible using the strong technique of sentiment analysis. The following section will concentrate on data analysis and the preparation of data for training models.

In order to enhance data analysis and investigate comments made in Kazakh, a histogram was made to compare the word usage frequencies of positive and negative remarks. The column on the histogram will be greater if the remark has more words. A visual depiction of the word distribution of comments is provided by the word count histogram, which can help discern between positive and negative comments. The average word count of each comment in each category can be ascertained by studying the histogram. This made it easy to compare comment lengths and spot any patterns or distinctions in emotion. It can also be determined which comment categories tend to contain longer or more detailed views by comparing the word counts of positive and negative comments. For instance, if we discover that the average word count of favorable comments is larger, this may

reflect a more in-depth description of a pleasant user experience. The context and content of the comments should be taken into account when evaluating the word count histogram data, among other things. Word count is merely one factor in assessing sentiment; other aspects of the text, such as expressions, phrases, and contextual factors, should also be taken into account. The comment word count histogram is a further tool for data analysis in the sentiment analysis thesis, in general. This method makes it feasible to evaluate the length of comments in several categories graphically and identify any discrepancies in the text's tone and content.

Let's examine how the comment word count histogram is applied in a particular scenario as one of the data analysis techniques used in dissertations. The word count histogram offers details on the word count distribution in comments and enables visual evaluation of which comments have larger or shorter contents. This is a handy tool for investigating textual data and spotting potential patterns or peculiarities in a group of comments. By examining the histogram, it is possible to ascertain the typical word count of positive and negative comments and investigate variations in the distribution of comment length between various categories. Positive comments, for instance, may contain more words on average, indicating that they are longer and more in-depth assessments. Negative remarks, on the other hand, might be shorter and use fewer words. The analysis of the word count histogram can also be used to spot anomalies or outliers in the data. When analyzing the results, it is important to take into account features or flaws in the data, which may be indicated by comments with very few or very many words. Note that the word count histogram does not consider the content or context of comments; it merely offers generic information about the length of comments. It assists in conducting a primary analysis of the data and acts as a supplemental tool for a broad understanding of the word distribution in the comments. The length of comments in the positive and negative categories were compared using a comment word count histogram in the thesis. This made it possible to spot variations in the distribution of comment lengths and determine which categories might have longer statements. (see figure 3.4)

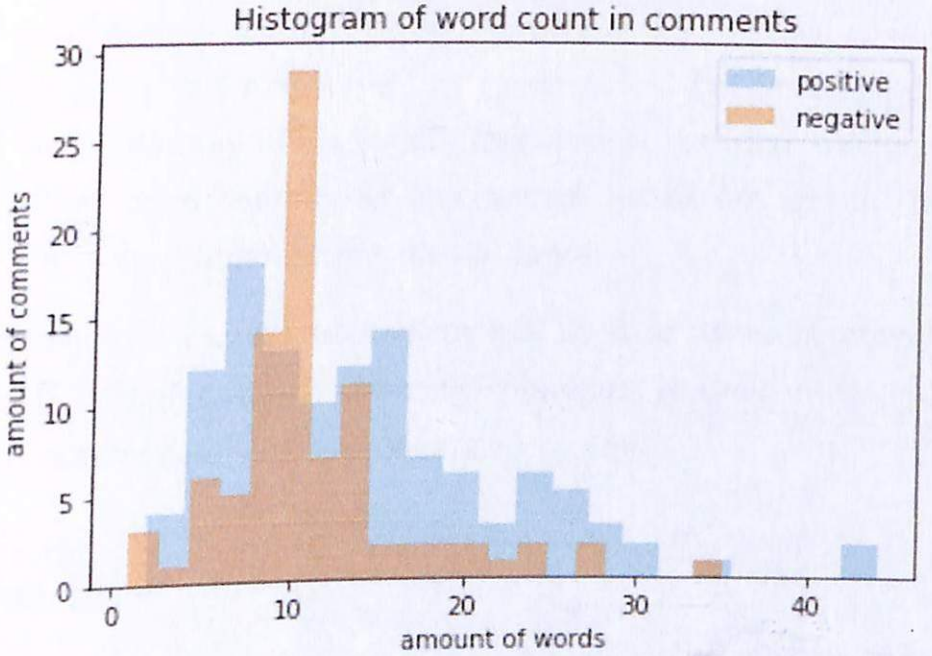

Figure 3.4: Histogram of word count in comments

Comparing the word count of favorable and negative comments is an essential component of study. This makes it possible to assess whether there are variations in sentence length across various categories and whether these variations are statistically significant. Positive comments typically contain more words than negative remarks, as shown by the histogram.

The following analytical procedures were carried out after analyzing the word count histogram:

Determining the Average Word Count: The average number of words in both favorable and negative comments was calculated. This will make it possible to ascertain how long or short comments tend to be for each category.

Comparison of distributions: The distribution of the number of words in favorable and negative comments can be visually compared using the histogram. In particular word count ranges, we can search for variations in the distributional shapes, the presence of peaks, or outliers.

Interpreting the results: It is possible to determine the typical length of comments in each category by analyzing the histogram. For instance, the histogram revealed that positive reviews had higher average word counts and a wider range

than negative reviews, which may mean that people who give favorable reviews tend to express their opinions in greater detail.
Identification of special cases: The histogram can also be used to find abnormalities or unique cases that might need more attention. For instance, data problems or unique comments may be to blame if we discover outliers with extremely high or extremely low word counts. As you can see in the histogram, the negative is represented by a very huge outlier on the graph.

The comment word count histogram was used to assess comment length and differentiate between good and negative comments in the context of the dissertation. The sentiment analysis was enhanced by this.

3.7 Models

The dataset was then divided into training and test sets after analysis and preprocessing. This made it possible to assess the effectiveness of models using several sets of data while avoiding overfitting. A model needs to be trained and tested before it can be added to a website as a recommendation system. Several machine learning models are available for application [5]. Train and contrast models like support vector machines, naive bayes, and logistic regression [22].The most accurate model is applied to make recommendations. Let's examine each model individually.

Logistic regression is a statistical method for simulating how a group of independent variables relate to a binary dependent variable. To determine the likelihood that an observation belongs to one of two classes is its goal.
The principal operations in logistic regression are as follows:
Linear Combination:
The creation of a linear combination of input variables is the first step in logistic regression:
`z = b0 + b1x1 + b2x2 + ... + bn*xn`
where b0, b1, b2, ..., bn - model coefficients (weights), x1, x2, ..., xn - values of independent variables.
Application of the logistic function:

The logistic function (sigmoid) then receives the input from the linear combination and transforms the value into an interval (0, 1):
$p = \sigma(z) = 1 \ / \ (1 + e\hat{}(-z))$
where $\sigma(z)$ is the logistic function, and p is the likelihood that a class member, respectively.
Class Probability Estimation: The logistic function is used, and the resulting p value is regarded as the likelihood of being in class 1 (spam). The observation is categorized as "spam" if p >= 0.5 and as "not spam" otherwise.
Model training:
By identifying such b0, b1, b2,..., bn values that minimize the loss function (often using the maximum likelihood approach or gradient descent), the model coefficients are determined. The objective is for the model, given the available independent factors, to predict the dependent variable as accurately as feasible.

A popular linear classifier for binary classification issues including spam detection, churn prediction, health diagnostics, and others is logistic regression.

Random Forest is a machine learning approach that addresses classification and regression issues by using a group of decision trees. To obtain more reliable and precise forecasts, it aggregates the predictions of different trees.
The basic steps of the Random Forest algorithm are as follows:
Bootstrap fetch: The bootstrap approach is used by Random Forest to create a large number of random subsamples from the training dataset. Each subsample is the same size as the main dataset and has a random overlap. To train each tree, various datasets are thus created.
Building trees: A decision tree is constructed for each dataset from the first stage. Each node is divided when constructing a tree according to specific criteria, such as information gain (Information Gain) or Gini index (Gini Index). Trees are constructed up to a predetermined depth or until a predetermined stop condition is satisfied.
Forecasting: Based on the forecasts of all the forest's trees, a class or value is projected for each new observation. The majority voting principle is used for classification problems, and the class that receives the most votes is taken as the final forecast. The predictions of all trees are averaged for regression issues.

Importance of variables: The relevance of variables is another piece of information that Random Forest offers. The degree to which a variable enhances class separation or lowers the total error across all trees is used to determine how effective it is. This makes it possible to assess the contribution of each variable to the model's predictions.
Handling imbalanced data: When one class dominates another in the data, Random Forest does a good job at handling this problem. This is because each tree's bootstrap sampling and voting processes automatically balance data as they proceed.
Stability: Data noise and outliers are not a problem for Random Forest. Outliers in individual trees can be smoothed out or made up for by other trees because the predictions are based on an ensemble of trees.
Scalability: Large datasets can be effectively used with Random Forest. Due to the independence of tree creation, it is easily parallelisable. As a result, the technique is scalable and can handle massive data sets.
Model parameters: There are a number of tuning options available for Random Forest. The amount of trees in the forest, the depth of the trees, and the quantity of factors taken into account for each split are a few examples.

The versatile machine learning tool Random Forest can be used for a variety of tasks, including classification, regression, image processing, and anomaly detection. It is a popular option in machine learning because it brings together the ease of use and adaptability of decision trees with the strength of ensemble models.

Naive Bayes is a straightforward but efficient classification method built on the idea of Bayesian inference and the feature independence assumption (hence the name "naive"). It is frequently used for applications including sentiment analysis, spam filtering, and text classification.
The Naive Bayes algorithm's primary steps are as follows:
Assumption of Independence: Naive Bayes makes the assumption that each feature has an independent impact on a class. We refer to this as conditional independence. Naive Bayes provides accurate findings in practice, notwithstanding the possibility that this assumption is untrue for actual data.
Model building: By computing the probability of each class and the probabilities

of each feature for each class, Naive Bayes creates a model from the training data. A count is utilized for categorical features while a distribution model (such as the normal distribution) is used for numerical features.
Class Probability Calculation: After creating the model, Naive Bayes determines the likelihood that an observation belongs to each class using Bayes' theorem. Based on the premise of feature independence, the conditional probability for each class for a new observation is computed.
Decision-making: When predicting a new observation, the Naive Bayes classifier selects the class with the highest probability. The observation will be labeled as "spam" for instance, if the likelihood that it belongs to the class "spam" is higher than the class "not spam".

The following benefits of Naive Bayes, a quick and easy classification algorithm, include strong performance on large datasets, effectiveness in training and prediction, and relative immunity to missing data and noise. However, in some circumstances, particularly when the traits are strongly interdependent, its independence assumption can limit accuracy. Additionally, because Naive Bayes examines each feature separately, it may have problems managing uncommon or missing combinations of features in the training dataset.
Naive Bayes is still a popular option for text categorization and other jobs despite its drawbacks, particularly when there are a lot of features. It is appealing for use in practical applications because of its ease of use, speed, and capacity to process massive amounts of data. Additionally, compared to other algorithms, Naive Bayes typically needs less training data to develop the model, which might be advantageous when there is a shortage of training data.

Support Vector Machine (SVM) is a potent machine learning method that handles classification and regression issues. In a multidimensional space, SVM creates a hyperplane or group of hyperplanes that effectively separates data from various classes.
Following are the Support Vector Machine algorithm's fundamental tenets:
Maximum separability: Finding the best hyperplane to maximally divide the two data classes is the goal of SVM. An N-dimensional space has a (N-1)-dimensional subspace called a hyperplane, where N is the quantity of features. To maximize

the "classification gap," also known as the distance (gap) between two classes, SVM looks for such a hyperplane.
Support vectors: The data points closest to the line separating the classes are called the support vectors, and SVM uses these to build a hyperplane. The hyperplane can be found and additional observations can be categorized using support vectors.
Building a separating hyperplane: SVM solves an optimization problem and creates a hyperplane that divides classes. It optimizes a "loss function" while accounting for the classification gap and the consequences of crossing the separation border.
Extension to non-linear cases: By utilizing a "kernel" to convert the initial features into a higher dimension, SVM can process non-linear data. As a result, non-linear class separation boundaries can be built by SVM in a new feature space.
Making decisions: The SVM is used to categorize fresh observations after the model has been constructed. The class to which the new observation belongs is determined, together with which side of the hyperplane it is on.

A few significant traits and advantages of SVM include:
Good generalizing ability: SVM is a powerful classification algorithm because it can create class separation borders that generalize to new observations.
Outlier tolerance: Support vectors are used in SVM to specify the class border. Since the support vectors are closest to the split boundary as a result, the technique is resilient against outliers in the data.
Efficient in High-Dimensional Spaces: SVM effectively manages the "curse of dimensionality" and can operate in feature-rich environments.
Relatively few options: SVM offers a number of settings that can be customized for optimum performance, including kernel type and regularization options. SVM operates well by default, though, even without changing the settings.
Ability to handle both binary and multi-class classification problems: SVM can be used to solve problems involving binary classification (dividing a problem into two classes) and multi-class classification (dividing a problem into more than two classes).

Described and introduced each model separately, however for model training will use the pre-made Scikit-learn library and its methods.

Scikit-learn (also known as sklearn) is a library for the Python programming language that supports machine learning. For resolving machine learning issues, it offers a variety of tools and methods, such as classification, regression, clustering, dimensionality reduction, data preprocessing, and others [23].
The scikit-learn library's primary characteristics and elements are listed below:
Machine learning models: Scikit-learn provides implementations of numerous machine learning models, including closest neighbor algorithms, support vector machines, decision trees, random forest, and gradient boosting model ensembles.
Data preprocessing tools: Data preprocessing methods from Scikit-learn are available, such as feature scaling, categorical feature encoding, missing value filling, dimensionality reduction, and others.
Choice of models and parameters: Cross-validation techniques, parameter grid searches, and model performance estimation are just a few of the tools that Scikit-learn provides for choosing the optimal model and fine-tuning its parameters.
Model Performance Evaluation: Scikit-learn offers measures including accuracy, recall, F1-measure, standard error, and more for assessing the performance of models.
Text data processing: Scikit-learn offers a variety of methods for processing text data, such as text vectorization, feature extraction from texts, text data conversion to numeric vectors, and others.
Expandability: An extensible package, Scikit-learn makes it simple to incorporate unique techniques and algorithms [23].
Integration with other libraries: Other well-liked Python data analysis packages like NumPy and Pandas are integrated with Scikit-learn. As a result, handling data arrays, operating on matrices, and vectors, as well as preprocessing and analyzing data, are made simple.
Convenient user interface: Scikit-learn's user interface is straightforward and easy to use, making it convenient for machine learning newcomers to utilize the package. It is simple to put up models and study data because many methods and classes have simple and understandable parameters.

Training on big data: With Scikit-learn, you can train models on big datasets using a variety of techniques, including stochastic gradient descent and incremental learning. This makes it possible to process and analyze big data effectively.
Extending functionality with plugins: Scikit-learn encourages the creation and incorporation of plugins that increase the library's capability. New data processing techniques, machine learning models, and data analysis tools can all be added using plugins.
Active community and documentation: Scikit-learn has a vibrant developer and user community that supports the library's continued development by exchanging ideas, offering solutions, and sharing experiences. It is simple to learn how to utilize the library's features because of its thorough documentation and usage examples.

One of the most well-known and often used machine learning libraries in the Python community is scikit-learn. It provides a wide range of machine learning tasks with an effective, usable, and reliable collection of tools and algorithms. This library is utilized during the research process for preprocessing, data training, and prediction.

Used a variety of machine learning models, as well as related tools and libraries, in the dissertation. Here is a more thorough explanation of each model and the tools employed.

Logistic Regression:
Library: sklearn.linear_model
Model: LogisticRegression
Used tools:
CountVectorizer from *sklearn.feature_extraction.text:* Converts text data into a matrix of counts.
StandardScaler from *sklearn.preprocessing:* Scales characteristics to improve model performance.

Random Forest:
Library: *sklearn.ensemble*
Model: RandomForestClassifier

Used tools:
TfidfVectorizer from *sklearn.feature_extraction.text:* Converts text data to numeric features using TF-IDF.
StandardScaler from *sklearn.preprocessing:* Scales characteristics to improve model performance.

Naive Bayes:
Library: *sklearn.naive_bayes*
Model: *MultinomialNB* Used tools:
CountVectorizer from *sklearn.feature_extraction.text:* Converts text data into a matrix of counts.
Normalizer from *sklearn.preprocessing:* data is Data is normalized for improved model performance.

SVC (Support Vector Machine):
Library: sklearn.svm
Model: SVC
Used tools:
TfidfVectorizer from *sklearn.feature_extraction.text:* Converts text data to numeric features using TF-IDF.
StandardScaler from *sklearn.preprocessing:* Scales characteristics to improve model performance.

TfidfVectorizer() is one of the text-vectorization techniques employed in problems involving natural language processing. The "Term Frequency-Inverse Document Frequency" (TF-IDF) approach, which enables text data to be represented as numeric vectors, is the foundation of this system.

TF-IDF Weights:
TfidfVectorizer() determines the weights for each term in text data according to the frequency with which it appears in a certain document (Term Frequency) and the inverse frequency of that document (Inverse Document Frequency) [23]. In order to emphasize crucial words and phrases, terms that frequently exist in this text but infrequently in other publications will be given a larger weight.

Converting text to numeric vectors:

TfidfVectorizer() creates a sparse matrix of numerical vectors from text data, where each vector is a representation of a distinct document. The TF-IDF weight of each element in the vector corresponds to a phrase in this document. As a result, the text can be used in machine learning algorithms that exclusively process numerical input.

Text Preprocessing:
TfidfVectorizer() enables text pre-processing prior to vectorization as well. To clean up and normalize text, it can conduct a number of operations, including lemmatization (reducing words to their simplest forms), stopword elimination, and tokenization (splitting text into individual words or tokens).

Parameter flexibility:
TfidfVectorizer() offers a variety of solutions that can be tailored to the unique requirements and properties of text data. The minimum and maximum term frequency, term length requirements, n-gram usage, and other criteria are a few of these parameters.

TfidfVectorizer() is a potent text vectorization technique that is frequently employed in tasks including classification, grouping, and information extraction. It provides for the consideration of terms' uniqueness in documents as well as their significance for a specific study. As a result, text classification is more accurate and machine learning models are of higher quality.

CountVectorizer() is one of the text-vectorization techniques employed in problems involving natural language processing. Based on the frequency count of each term (word) in the text, it turns text data into numerical vectors.
Features of CountVectorizer:

Term Frequency Count:
CountVectorizer() the number of times each term appears in the text. Based on the unique words in the text, it compiles a dictionary of terms and records data on how frequently each phrase appears in each document. As a result, the text is transformed into a numeric matrix, where each row represents a document and each column represents a phrase.

Converting text to numeric vectors:
CountVectorizer() creates a sparse matrix of numerical vectors from text data. The number of times the related term appears in the provided document is represented by each element of the vector. Therefore, these number vectors can be used by machine learning models for both training and prediction.

Text Preprocessing:
CountVectorizer() allows for the execution of different text preparation operations to enhance vectorization quality. To clean up and normalize text, it can execute operations such as tokenization (splitting text into individual words or tokens), deleting stop words (often occurring but meaningless terms), changing words to lowercase, and other operations.

Parameter flexibility:
The *CountVectorizer()* gives me a range of alternatives that I may tailor to my individual needs and the properties of text data. The minimum and maximum term frequency, term length requirements, n-gram usage, and other criteria are a few of these parameters.

CountVectorizer() is a straightforward and effective text vectorization method that's frequently used to solve classification, clustering, and information extraction issues. It enables the representation of text data as numerical vectors while accounting for the frequency of occurrence of textual phrases.

Utilised the following resources and tools as well:
numpy: A library for handling data arrays.
pandas: Work with data in table format using a library.
sklearn.model_selection.train_test_split: Instrument for separating data into test and training sets.
sklearn.pipeline.Pipeline: A device for building pipelines, models, and chains of data manipulations.
sklearn.preprocessing.StandardScaler: Tool for feature scaling
sklearn.preprocessing.Normalizer: Tool for normalizing data.

In experiment, split the data into train and test sets using *train_test_split* from *sklearn.model_selection*. This made it possible to train the models on

the training set and test their effectiveness on the test set. Pipeline from was used to build pipelines *sklearn.pipeline* that combined data transformations (eg CountVectorizer and TfidfVectorizer) with models (eg LogisticRegression, RandomForestClassifier, MultinomialNB, SVC). Pipelines made it possible to train models and convert data systematically. Several models, such as *LogisticRegression* and *RandomForestClassifier*, used text data preprocessing with *CountVectorizer* or *TfidfVectorizer*. *CountVectorizer* transforms text into a counters matrix, and *TfidfVectorizer* text is converted to numeric characteristics, given TF-IDF (term frequency-inverse document frequency). Also for some models, such as *SVC* and *MultinomialNB*, used *StandardScaler* strategies for data scaling to improve model performance.

Additionally, to cope with data in array and table formats, the *numpy* and *pandas* libraries were employed. These libraries offer strong tools for processing and analyzing data.
The following steps made up the experiment's general structure:

1. Loading data.
2. Creating training and test sets from the data.
3. construction of pipelines for every model.
4. Using pipelines, train models on a training set.
5. Calculating a model's test set performance estimate.
6. Analyze the performance of each model by comparing the outcomes.
7. Discussion of the results and conclusions.

In order to experiment with several machine learning models and assess how well they performed when applied to a text categorization problem, we employed a variety of tools and libraries from scikit-learn, numpy, and pandas.

3.8 Evaluation

The outcomes of assessing the effectiveness of machine learning models for the text categorization problem are presented in this section. Several metrics were employed to evaluate the results, such as *accuracy_score, precision_score, recall_score* and *f1_score* from the *sklearn.metrics* library.
Let's examine the metrics stated in more detail:

- accuracy_score: This statistic counts the proportion of samples that were properly identified out of all samples in order to determine the model's accuracy. When the classes in the data are balanced and the model's overall quality is crucial, it may be useful.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.8.1}$$

- precision_score: This statistic assesses how well the model predicts a positive class. It is calculated as the proportion of positively categorized samples to the total of positively and falsely classified samples. It is helpful in situations when reducing false positive predictions is crucial.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{3.8.2}$$

- recall_score: This score assesses how well the model can identify positive samples. The ratio of correctly identified positive samples to the total of correctly classified positive and false negative samples is used to compute it. It is helpful in situations where it is crucial to reduce the number of unfounded negative forecasts.

$$\text{Recall} = \frac{TP}{TP + FN} \tag{3.8.3}$$

- f1_score: This measure represents the harmonic mean of the model's accuracy and recall. It is a fair metric that considers the completeness and correctness of the model. It is helpful in situations with unbalanced classes

when it is crucial to take both false positive and false negative predictions into account.

$$\text{F1-score} = \frac{2 \cdot (\text{Precision} \cdot \text{Recall})}{\text{Precision} + \text{Recall}} \tag{3.8.4}$$

These metrics were calculated using the corresponding functions from *sklearn.metri* such as *accuracy_score, precision_score, recall_score, f1_score.* Then, after passing anticipated labels and ground truth as parameters to these functions, metric values were returned.
These measures were used to compare the effectiveness of several models and determine which model is more effective at classifying text.

There are various stages in evaluation:

- Description of the dataset: A total of 330 comments were used in the experiment, split into a training set and a test set. Each comment was marked with a "+" or "-" to denote its positivity or negativity. Additionally, the models were assessed using 100 comments.

- Setting up an experiment: Pipelines were created using a variety of machine learning models, including Logistic Regression, Random Forest, Naive Bayes, and SVM, to assess the performance of the models. The data transformations included in each pipeline, like text vectorization utilizing the *CountVectorizer()* or *TfidfVectorizer()*, and data preprocessing using the *StandardScaler()* or Normalizer for some models.

- Evaluation results: The following indicators were employed to assess how well the models performed: *accuracy_score, precision_score, recall_score,* and *f1_score.* These metrics were determined for each model using test data, and the findings were recorded in a table for comparison. In-depth descriptions of the findings will be provided for each of the experiments.

Chapter 4

Experiments

4.1 Comparison of models

The experiment starts with a model comparison. About 230 Kazakh-language comments were utilized for training, and 100 remarks were used for the test. The results of the test were also shown in the new comments as + and -, positive results (+) and negative results (-). The models were trained using the prepared data at the start of the experiment. The dataset was then divided into training and test sets after analysis and preprocessing. This made it possible to assess the effectiveness of models using several sets of data while avoiding overfitting. Next, we chose a few well-known machine learning models that were appropriate for our task: Naive Bayes, Support Vector Machine, Logistic Regression, and Random Forest. During the experiment, the models were trained using the proper learning methods on the training set.

The first time, the models were trained immediately without the aid of any training aids. As an illustration, early training without *TfidfVectorizer()* and *CountVectorizer()*. The text was not tokenized, the words that had no special value were not removed, and the text was not turned into a vector of integers. After training, test them with fresh comments, and then use the *sklearn.metrics* library method to determine their accuracy. Naive Bayes had an accuracy of 85%, which was very low, but I tried the test again after employing the crucial and essential procedures. The section on models and scientific learning went into great

depth on how each model was prepared for training separately using the relevant library methods.

Used each technique for all models and abandoned the techniques that produced the best outcomes. In the experiment, various machine learning models were compared in order to assess their efficacy and select the model that would best address the categorization issue in the context of the thesis. The Random Forest, Logistic Regression, Naive Bayes, and SVM models were trained and tested as part of the experiment.

Training was done on the training set using the correct settings, starting with the Random Forest model. Following training, the model was put to the test on a test sample, and the following findings were found: accuracy was 83%, precision was 81% percent, recall was 95% percent, and the F1-score value was 86%. These results demonstrate the model's successful classification of comments.

After that, the model for logistic regression was trained. The accuracy, precision, recall, and F1-score values were as follows after the model had been trained and tested: 88%, 85%, 96%, and 90%. These metrics show how well the Logistic Regression model performs in terms of classification accuracy and recall.

Next, the Naive Bayes model was examined. Accuracy, precision, recall, and the F1-score value of the model were found to be 88%, 90%, 88% and 87%, respectively, after training and testing. In comparison to earlier models, the Naive Bayes model displayed good accuracy and accuracy, but slight loss in recall.

The SVM model was then trained. Following model testing, the following results were discovered: accuracy of 95%, precision of 90%, recall of 96%, and the F1-score value of 95%. The SVM model performed well, with good accuracy, recall, and F1-measure values. (see table Table 4.1)

Table 4.1: Comparing the accuracy of various machine learning algorithms

Algorithm	Accuracy	Precision	Recall	F1-Score
SVM	95%	92%	96%	95%
Random Forest	83%	81%	95%	86%
Logistic Regression	88%	85%	96%	90%
Naive Bayes	88%	90%	88%	87%

According to the results, it can be said that all four models did a decent job

of sorting comments into positive and negative categories. The SVM model, however, demonstrated the highest precision, recall, and F1-score values, making it the most appropriate for handling the classification issue in the context of a thesis. Because of the SVM model's great accuracy and completeness, it can accurately categorize a vast number of comments and identify both positive and negative ones. The F1-score, which measures precision and recall harmonically, has a high value, supporting the model's overall effectiveness. (see figure 4.1)

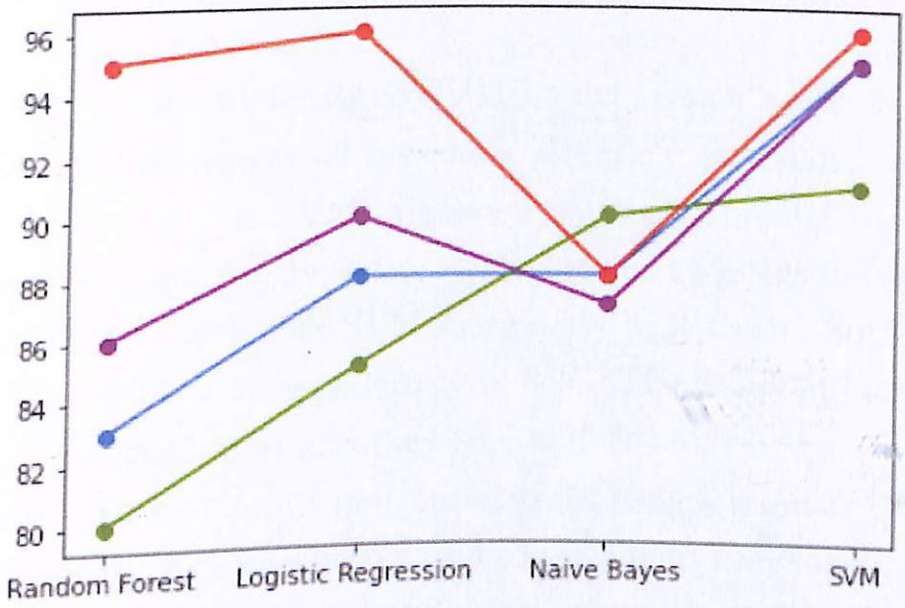

Figure 4.1: The comparison graph of classification models

Accuracy, Precision, Recall, and F1-Score are the four different metrics used to compare classification model performance. It displays the findings of an experiment in which four different models—Random Forest, Logistic Regression, Naive Bayes, and SVM—were trained and put to the test. The graph has a Y-axis that shows the values of the metrics and an X-axis that lists the names of the models. The graph has a different line for each metric, and the dots on the lines correspond to the values for each model's metric. To set one statistic out from the others, colors are employed. Accuracy is represented by blue, precision by green, recall by red, and F1-Score by purple. It is simple to distinguish which column belongs to which metric thanks to the variation in color. The graph illustrates how each model yields unique outcomes for various measures. For instance, Random Forest offers somewhat lower F1-Score, but great accuracy (accuracy) and good recall (recall). However, Logistic Regression has higher F1-Score and accuracy,

but lower F1-Score and recall. Naive Bayes has a high accuracy (accuracy), a high precision (accuracy of prediction), but a low recall (completeness). Finally, SVM achieves the top results in accuracy, precision, recall, and F1-Score, performing well across all criteria.

The graph is a crucial tool for evaluating the effectiveness of various models and showing the outcomes of an experiment. It enables comparison and evaluation of each model's performance against several indicators, allowing the researcher and readers to quickly determine which model is best for a given classification assignment.

The findings demonstrated that the SVM model, which achieved an accuracy of 93%, performed best across all measures (accuracy, precision, recall, and F1-score). This is so that the SVM, a more complicated model that can handle non-linear class borders, can be used. Naive Bayes and logistic regression also produced good results, although SVM marginally beat them. Both Naive Bayes and Logistic Regression have an accuracy of 88%. The Random Forest model, on the other hand, had the lowest accuracy 83% and this may be because it struggles to perform well with text input and necessitates a high number of trees. However, the experiment was continued in order to evaluate the models. I sought the opinions of two people. When comparing the outcomes of the models with the opinions of humans, what category are test comments labeled as good or negative? Following a comparison of the models among themselves, a comparison of each model's output with people was made. since the model presents the outcomes in the manner in which they were learned, for instance, if positive then (+) and if negative then (-). There were some noteworthy interesting nuances discovered when contrasting the model results with human assessments.People were given remarks as part of the experiment that were either favorable or negative according to the models. The aim for the participants was to rate each comment and indicate whether they thought it was favorable or negative.

People found it challenging to pinpoint the precise emotional content of many comments while analyzing them and to decide between positive and negative options. In these situations, the comments frequently fell into the category of neutral. Even though such comments are uncommon, they are intriguing because they can include emotional nuances that are challenging to convey with a straight-

forward binary classifier.

This finding paves the way for additional study and development. Consider including impartial comments in the model training data and investigate how this would effect the models' performance and accuracy. More of these comments might be gathered to broaden the emotional categories that the models can use and improve their ability to precisely categorize various emotional nuances.

Additionally, it may be inferred from the results of comparing the models with people's ratings that there is some degree of agreement between the models and the ratings. It's possible that some models are more reliable and compatible with observations than others. This can be used as a foundation for additional model refinement and for choosing the best model for particular categorization issues.

The effectiveness of categorization models in reflecting the emotional content of comments and interacting with human perception is generally improved by comparing the results of models with human ratings. This enables a more thorough analysis of the models' performance and identification of both their advantages and disadvantages. The ability to gauge the degree of agreement between model findings and estimations made by individuals is one of the key benefits of doing so. The capacity of the models to accurately mimic how individuals perceive emotions is demonstrated by how well they accord with observers' judgements. Developers and academics who want to develop models that most faithfully represent human preferences and judgments may find this information to be helpful. On the other side, a comparison to assessments from the general public can also highlight variations and disparities between model predictions and actual perceptions. This might be a sign of the challenges and constraints models face when it comes to effectively categorizing emotions and capturing the intricacies of text's meaning. Such differences may be the focus of future study and model development.

Additionally, the gathering of estimations from individuals enables the research to take into account human intuition and expertise. Human assessments can be based on a richness of knowledge and comprehension of the text, adding value to the evaluation of models. In order to construct models for classifying emotional content that are more accurate and effective, machine learning and human intelligence can be combined through comparison with human assessments.

The extension of the model comparison with individual participants' scores may

be considered in future research. For instance, new studies might be conducted with a wider range of participants and more evaluation comments.
For instance, Naive Bayes and Logistic Regression produced good findings, but many comments did not agree with the results when they were compared to people's opinions. Additionally, I believed that since the aim of logistic regression is to forecast the likelihood that an observation belongs to one of two classes, it would be better suited to the job of the thesis. and only two classes. However, as evidenced by the outcomes, the Support vector machine model performed well with text data. also contrasted his findings with those of those who nearly all matched.

4.2 Identifying words with the greatest impact on prediction

The research included an investigation of the phrases that had the greatest impact on the classifier's prediction. For this, the *get_feature_ names_ out* function was used, which allows to get a list of all words from the model dictionary. 20 terms were found to have the biggest influence on prediction based on this list. The 20 words with the most influence were derived in order to better understand which lexical components are most crucial for categorizing comments. In addition to understanding which parts of the text have the most influence on the text's emotional content, this can help uncover major themes and concepts linked to both positive and negative comments.
Further investigation was done to identify the top 10 words that are most effective at predicting favorable and negative responses. This makes it possible to concentrate on the phrases that best describe each emotional group.
First, the Support vector machine model was used to determine the most powerful words. The code to examine the word weights in a support vector machine (SVM) model and output the words with the greatest influence on the model's predictions can be found here.

In the pseudocode (see pseudocode Listing A.1) gets a list of all features (words) using the *get_feature_ names_ out()* method from the vectorizer object and stores

it in the *feature_names* variable. Then, using the coef_ attribute of the SVM svc model, get the weights of each feature (word) and convert them into an array (array) using *toarray()*. The *ravel()* method is then used to convert the two-dimensional array to a one-dimensional array. Next, a *weights_dict* is created, where the keys are the words from feature_names and the values are the corresponding weights from *svm_weights*. This dictionary enables you to determine a word's weight based on its name. Using the sorted function and a for loop, the 20 most influential words and their weights from the *weights_dict* are displayed under the header "20 most influence words for the prediction:". The words are listed in decreasing weight order.

Similar to this, other models also created algorithms and highlighted their most important words.

Two empty lists are then made: *negative_words* and *positive_words*. Next, iterates over the words in the *weights_dict*, and if the weight of a word is greater than 0, then this word is added to the *positive_words* list, otherwise - to the *negative_words* list. So, categorize words into good and negative categories based on their importance. The 10 most influential positive words and their weights from the *weights_dict* are presented along with the caption "The 10 Most Influential Negative Words:" using the sorted function and a for loop. The words are listed in decreasing weight order. The 10 most influential negative words and their weights from the *weights_dict* are presented along with the title "The 10 Most Influence Positive Words:" using the sorted function and a for loop. The words are listed in ascending weight order. (see pseudocode Listing A.2)
The appendix contains the output of each model. A table and graph are used to display the results. (see results of research Figure A.1, Figure A.2, Figure A.3 Table A.1)

The examination of the most significant words reveals important details about the lexical traits that are crucial in determining the emotional tone of remarks. This will be helpful for the development of new methods for the study of emotions in texts as well as for further refining categorization models. This allows for a greater grasp of how textual information relates to emotional content thanks to the findings about the most influential words and their impact on prediction. This

could be beneficial for future study and method development for understanding emotions in text.

4.3 Adding a recommendation system to the website

After all the data adjustments, this model was posted to a nearby website for testing. The trained SVM was then added to the Django project in order to be used for book suggestions.

1. Created a new Python module for the Django project, *recommendation.py.*
2. In the *recommendation.py* file, put the commands used to preprocess the data and train the SVM model.
3. A new function that will take user input and return book suggestions has been added.
4. saved the *recommendation.py* file and sent it on to the Django project's *views.py* file.
5. Imported the *get_book_recommendations()* function from the recommendation module
6. created a view that would deal with the user's request and use the SVM model to obtain book recommendations.
7. In *urls.py* file, include a path in the suggested view.
8. Created a *recommendation.html* template that will show both the results of the recommendations and the user input form.

In the recommendation.py file, defined the get_book_recommendations(user) function that will use in view to get book recommendations. Define a *get_book_recommendations(user)* a function that accepts an input object. then get the user's information to determine their favourite languages and genres. After that, use the filter and annotate approach to annotate the amount of likes while filtering the books depending on these preferences, comments, and the presence

of likes. Then arrange the books according to the quantity of "likes"and "positive comments"in descending order.

After that, registered new users on website, for each of them indicating their preferences regarding the genre and language of books. The first user chose the genre "Жеке даму"(self-development) and the language "Russian"as preferences. This indicates that the user prefers to read Russian-language literature on the topic of self-development. The second user, on the other hand, chose the genre "Психология"(psychology) and the language "Russian". He values psychology-related books, and he also likes to read in Russian.

Users can utilize the relevant functions and models in the Django project to acquire book recommendations that fit their preferred genre and language using information about each user's preferences. Each user will be able to benefit from tailored book recommendations and take pleasure in reading books that suit their interests and tastes thanks to this.

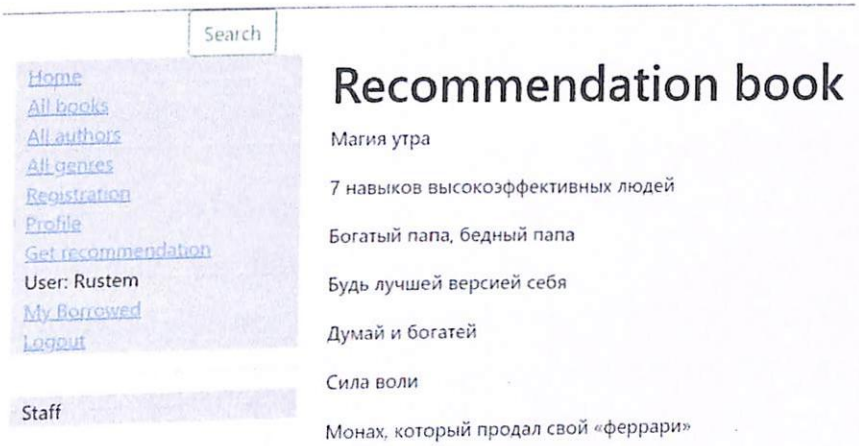

Figure 4.2: Book recommendations for the first user based on his preferences

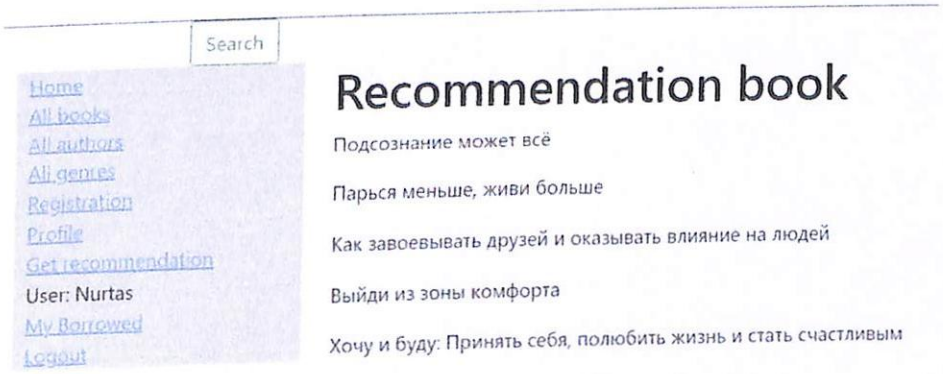

Figure 4.3: Book recommendations for the second user based on his preferences

As you can see from the figure 4.2 for the user who chose the genre "Жеке даму"(self-development), the website provided 7 recommendations from a total list of 15 books in this genre. The first book recommended was "Магия утра" (The Magic of the Morning). Considering that it has 4 positive comments and 4 likes, this book was chosen as the first suggestion. 2-3 people liked and commented on the remaining recommendations. Some of them also received criticism and disapproval, which may have lowered their ranking on the list of suggestions.
As a consequence, the website offered tailored suggestions for a user interested in self-development, taking into consideration the quantity of favorable remarks and likes for each book. This enables the user to choose the appropriate books for their reading needs and self-development by providing recommendations based on the popularity and favorable reviews of the books.

For the user who chose the genre "Психология"(Psychology) in figure 4.3, the website provided 5 recommendations from a total list of 11 books in this genre. The first book recommended was "Подсознание может все" (Mind power). The first suggestion was made for this book because it has received 4 positive comments and 5 likes. Additionally, the remaining suggestions received 2 to 3 likes and comments. Some of them also received criticism and disapproval, which may have lowered their ranking on the list of suggestions.
As a result, the website offered tailored suggestions for a user with an interest in psychology, taking into consideration the quantity of favorable remarks and likes for each book. This enables the user to choose the appropriate books to study psychology and satisfy their reading demands by providing recommendations based on the popularity and favorable ratings of the books.

Chapter 5

Discussion of results

The four machine learning models compared in the study were SVM, Logistic Regression, Random Forest, and Naive Bayes. The objective of the study was to determine the best practical model for classifying comments into positive and negative groups.

Following training and testing, the SVM model outperformed the other models in terms of classification predictions for comments. It had excellent recall and accuracy, proving the model's capacity to correctly categorize both positive and negative comments. It is imperative to keep in mind that the results could vary based on the precise activity and the data used.

For a more in-depth examination of the comments, a histogram was made that shows the word count in each comment. This allowed for the measurement of comment length and word count distribution. Future work with data and model optimization may benefit from the analysis of this kind of data.

Furthermore, the importance of words for model predictions was looked into. made a list of the words and phrases that had the biggest influence on model predictions. This makes it simpler to understand whether particular characteristics or words are important in discriminating between positive and unfavorable responses. In the future, this information might be useful for the models and their interpretation.

Overall, the research's conclusions corroborate the SVM model's capacity to distinguish between positive and negative comments. It is critical to remember that

the model to apply and the pertinent qualities to identify depend on the specific task at hand and the data being used. The results of this study can be applied to user evaluation analysis and the development of more precise and potent recommendation systems.

Some keywords were discovered during the investigation of word importance for model predictions that significantly affect whether comments are categorized as positive or negative. In negative comments, the words "тым"(too much), "ұнамады"(did not like), "болжамды"(predictable), "өлшемді"(size) and "нашар"(bad) were identified, which are found in many negative comments, and all the models that were tested are highlighted as important.These lines perfectly express the comments' displeasure and critical tone. Their existence reflects dissatisfied users and negative sentiments. It's noteworthy to see how these terms were all emphasized uniformly across all models, highlighting their potency and significance in identifying key statements. On the other hand, the words "керемет"(amazing), "жақсы" (good) and "ұнады"(liked) were identified in positive comments, which all examined models emphasized as crucial for predicting favorable remarks. These words convey pleasant emotions and user enjoyment. Their inclusion in the comments confirms the good tone of the evaluation and affects how supportive comments are perceived. Analyzing such keywords aids comprehension of which specific terms are essential in the classification of comments. This information may be used to enhance the model further and gain a better grasp of the important factors influencing user input.

In addition, it is important to note the presence of the word "емес"(not), which can significantly change the context of the sentence and affect the classification of comments. For example, if we consider the phrase "Жақсы кітап"(good book), it will be considered as a positive comment. However, if we add the word "емес the phrase "Жақсы кітап емес"(not a good book) will be perceived as a negative comment. Also vice versa "Жаман кітап"is a negative comment, but if we add the word "емес"for example "Жаман кітап емес"(not a bad book) then this is already a positive comment. The word "емес"has a strong negative connotation and can change the meaning of the entire sentence. Its existence denotes hostility toward the thing or thing in question. This word may be a crucial indicator of

unfavorable comments in the context of comment analysis. Given this aspect, it is crucial to account for contextual changes brought on by the term "емес"while developing a model for comment classification. It is important to pay attention to how it affects the model's predictions and to the likelihood that, if any, the context and tone of the comments may change.

To determine the effect of the word "емес"on model predictions, it is possible to use a method based on the creation of distinctive features or the modification of data.
One approach is to incorporate a special feature that will indicate the presence of the word "емес"in the comment. To do this, used the *CountVectorizer()* or *TfidfVectorizer()* method with settings to count individual words, including "емес". Then add this feature to the feature matrix before training the model. (see figure 5.1)

```
vectorizer = TfidfVectorizer(ngram_range=(1, 1), token_pattern=r'\b\w+\b|емес')
```

Figure 5.1: Adding a word to the method that it is taken into account in the prediction.

In this algorithm, *TfidfVectorizer()* will take into account individual words, including "емес". The feature matrix X will then get a feature indicating the presence of the term "емес". This attribute is taken into consideration during the model's training and prediction processes.
This conclusion emphasizes how important it is to look at word combinations and contextual interactions in addition to single words in order to classify comments more properly and understand the emotional undertone of users' perspectives.

The word "дегенмен"(although), which is also a part of the Kazakh language, might alter the sentence's context. For instance, in lengthy comments that span two or more phrases, the user may condemn the book in the first sentence while praising it in the second. After the first sentence's criticism, the word "дегенмен"in the second sentence might modify the context of the comment and draw a contrast between its negative and positive aspects. This is so because the word "дегенмен"has the opposite connotation and is frequently used to introduce information or arguments that are in opposition.

In example, when a user in the first sentence criticizes and describes the negatives of the book, and then begins the second sentence with "дегенмен"and proceeds to describe the good points, this can create a contrast effect and change the overall perception of the comment. The word "дегенмен"emphasizes that despite the criticisms in the first sentence, the second sentence will bring out the positive aspects of the book This use of the word "дегенмен"helps to balance the statement and demonstrates that the commentary is not entirely negative, but contains both positive and negative opinions about the book. This makes it possible to present a more thorough and impartial point of view on the subject at hand.
Understanding how words like "дегенмен"are used in context will help you better grasp how to read comments and how different word choices can change how statements are meant to be taken emotionally.

Thus, research related to the words "емес"and "дегенмен"will enhance the recommendation system by increasing its sensitivity to user demands and preferences. This will enhance the suggested recommendations' quality and relevancy, resulting in a more positive user experience.

The examination of user preferences and the results of the recommendation experiment can be used to note the following:
Genre "Жеке даму"(Self-development):
Of the total list of 15 books in the genre of "Жеке даму 7 books were recommended for the first user. The first recommended book was "Магия утра"(Miracle morning) with 4 positive comments and 5 likes. 2-3 people liked and commented on the remaining recommendations. A few of the suggested works received criticism and disapproval.

Genre "Психология"(Psychology):
Out of a total list of 11 books in the "Психология"genre, 5 books were recommended for the second user. The first book recommended was "Подсознание может все"(Mind power) with 4 positive comments and 5 likes. The rest of the recommendatior also had 2-3 likes and comments.
An study of these statistics shows that the recommendation system takes into account the quantity of positive comments and likes for each book. Users can now get personalized recommendations based on their preferences and interests.

To provide clients with personalized recommendations, the recommendation system was successfully developed, tested, and implemented into the website. In order to evaluate the system's functionality and its ability to take into account positive comments and the quantity of likes, we created two new users and gave them the option to choose their preferred genre and language. Using the selections made at user registration, the system provided recommendations.The algorithm utilized by the system looked at every book that was available in the selected genre and language while also considering how many positive reviews and likes each book had. The machine ranked the suggestions based on this data, showing customers the books with the best ratings and likes at the top. New users might then get recommendations that are personalized for them and take into account the positive reviews and preferences of prior users. This will enable users to rapidly find intriguing publications that have garnered interest and positive reader evaluations in the past.

By employing comments and likes as system recommendation components, it is able to take into account both user preferences and the consensus among readers on the book. This improves the quality of recommendations and makes it easier for consumers to select books they will like based on popularity and positive ratings.

Chapter 6

Conclusions and future work

6.1 Conclusions

A recommendation system for an online library was created as part of this project using machine learning and data analysis techniques. The study's goal was to develop a useful recommendation system that could offer consumers tailored book recommendations based on their interests and an analysis of comments and likes. Data about books, users, and comments in the Kazakh language were gathered during the study. Several techniques were employed for this, including data from libraries and well-known people, a Telegram bot that produced remarks in Kazakh, and data from libraries. Users selected reading preferences for genre and language upon registration, allowing recommendations to be generated with their preferences in mind.

Using Sentiment Analysis techniques, the obtained data were pre-processed and split into positive and negative remarks. The sentiment of the comments was then predicted using a variety of classification models, including Logistic Regression, Random Forest, Naive Bayes, and Support Vector Machine. The Support Vector Machine model has the highest classification accuracy result (95%) when comparing the outcomes of the models, proving its efficacy in foretelling the sentiment of comments made in Kazakh.

A histogram was created to count the words in the comments as part of the data analysis. Positive comments contain more words than negative ones, according

to the histogram. Along with ten words for both positive and negative remarks, the top 20 words that had the biggest impact were highlighted. For each model, particular words were compared. The peculiarities of these terms' influence on prediction could be shown by comparing them across several models [24].
The research also found that there are words in the Kazakh language that can change the context of a sentence, such as "емес"and "дегенмен". This was taken into account when designing the recommendation system to prevent misrepresentation of comment values. Added these words to *TfidfVectorizer()* and *CountVectorizer()* methods so that these words are taken into account when predicting.
A comparison with the feedback of actual users was conducted to assess the validity of the model results. The comparison revealed a match between user reviews and the SVM model's predictions, demonstrating the efficiency of the recommendation system.
The research shown that the Support Vector Machine model and the created online library recommendation system are extremely accurate and can successfully propose books to the user while taking into consideration their preferences and ratings.
As a consequence, the Support Vector Machine model and user preference data (genre and language) were used to include the recommendation system into the created website. Two new users were created and assigned reading genre and language preferences, 7 books were recommended for the "Жеке даму" (Self-development) genre and 5 books for the "Психология"(Psychology) genre. Books with the highest number of positive comments and likes were the first in the recommendations. Taking into account the quantity of favorable comments and likes, the website successfully recommended books, proving the value of the built recommendation system.

The following outcomes of this investigation were attained:

1. The user's preferences for reading genre and language have been taken into account in the development of a recommendation system for an online library.

2. A telegram bot, data from libraries, and information from well-known users were only a few of the approaches used to gather information about books,

users, and comments in the Kazakh language.

3. To examine comments and identify their sentiment, Sentiment Analysis techniques and classification models (Logistic Regression, Random Forest, Naive Bayes, and Support Vector Machine) were used.

4. The recommendation system will use the most accurate Support Vector Machine model, which has a 95% accuracy rate.

5. When data was analyzed using a histogram, disparities between the quantity of terms used in positive and negative comments were found.

6. The most potent phrases that have a noticeable impact on model predictions are included.

7. It was also made clear that there are terms in Kazakh that can alter a sentence's context.

8. Created and put into use a website with a recommendation engine that successfully gives consumers individualized recommendations.

The research of user comments and likes served as the foundation for this dissertation's research and development of an efficient recommendation system for an online library. The results show that using machine learning and data analysis techniques enables the delivery of pertinent and customized recommendations that take into account user preferences. The created technology has the potential for real-world applications and can greatly enhance online library users' reading experiences.

6.2 Future work

However, more research and creation of the recommendation system might cover the following areas:

1. Increasing the study's participant base and data set will enable more accurate analysis and the formulation of recommendations.

2. Enhancing machine learning models with more complex algorithms and

methods, including deep learning or neural networks, to increase the precision of predictions.

3. To accommodate a larger range of user preferences, the capability of the recommendation system should be expanded to include other criteria like the book author, user ratings, and other metrics.

4. Investigating the use of additional text analysis techniques, such as topic modeling or sentiment analysis, to better comprehend the content of books and user reviews.

5. To offer a more practical and intuitive user experience, user interface changes and system optimizations have been made.

6. Testing the effectiveness of the recommendation system by additional study and tests in various scenarios and languages.

The proposed recommendation system is a potential tool for online libraries that can considerably enhance user experience and satisfy user needs, as can be inferred from the findings and recommendations. This system's future study and development could result in even more precise and pertinent recommendations that take into account individuals' unique preferences and raise their level of happiness with reading books.
Thus, the dissertation research on "Development of recommendation system for online library" contributes significantly to both the creation of recommendation systems and the study of user behavior in online libraries. The findings and recommendations of this study may be helpful to researchers, developers, and practitioners tackling related issues.

In conclusion, the creation of an online library recommendation system based on sentiment analysis and the SVM model is a significant step toward the creation of intelligent systems and the tailoring of user experiences. This approach can be used in other fields as well, not just the library industry, where it is necessary to make pertinent recommendations based on the evaluations of users and a like/dislike system.

Bibliography

[1] Zanker; Alexander Felfernig; Gerhard Friedrich Dietmar, Jannach; Markus. Recommender systems: An introduction. 2010.

[2] Rokach; Bracha Shapira; Paul B Kantor Francesco, Ricci; Lior. Recommender systems handbook. LLC, 233 Spring Street, New York, NY 10013, USA, pages 1–845, 11 2011.

[3] Falk Kim. Practical recommender systems. Journal of Ohio State University, pages 1–448, 01 2019.

[4] Bilbro; Tony Ojeda Benjamin, Bengfort; Rebecca. Applied text analysis with python: Enabling language-aware data products with machine learning. 2018.

[5] Suthaharan Shan. Machine learning models and algorithms for big data classification: Thinking with examples for effective learning. 1st ed. 2016, October 2015.

[6] Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. IEEE transactions on knowledge and data engineering, 17(6): 734–749, 2005.

[7] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms. In Proceedings of the 10th international conference on World Wide Web, pages 285–295, 2001.

[8] Robin Burke. Hybrid recommender systems: Survey and experiments. User modeling and user-adapted interaction, 12:331–370, 2002.

[9] Dhanashri Wadikar, Nandani Kumari, Ranjana Bhat, and Vaishali Shirodkar. Book recommendation platform using deep learning. International Research Journal of Engineering and Technology, 7(6):6764–6770, 2020.

[10] Bogdan Walek and Petr Fajmon. A recommender system for recommending suitable products in e-shop using explanations. In 2022 3rd International Conference on Artificial Intelligence, Robotics and Control (AIRC), pages 16–20. IEEE, 2022.

[11] Wang Tao. Emotional classification model of review text based on topic and sentiment characteristics. In 2020 2nd International Conference on Economic Management and Model Engineering (ICEMME), pages 1057–1061. IEEE, 2020.

[12] Xueying Zhang and Xianghan Zheng. Comparison of text sentiment analysis based on machine learning. In 2016 15th international symposium on parallel and distributed computing (ISPDC), pages 230–233. IEEE, 2016.

[13] Bing Liu. Sentiment analysis: mining sentiments, opinions, and emotions. Cambridge, UK: Cambridge University, 2015.

[14] Bo Pang, Lillian Lee, et al. Opinion mining and sentiment analysis. Foundations and Trends® in information retrieval, 2(1–2):1–135, 2008.

[15] Cach N Dang, María N Moreno-García, and Fernando De la Prieta. Using hybrid deep learning models of sentiment analysis and item genres in recommender systems for streaming services. Electronics, 10(20):2459, 2021.

[16] Django tutorial: The local library website. URL https://developer.mozilla.org/en-US/docs/Learn/Server-side/Django/Tutorial_local_library_website.

[17] George A Morgan and Robert J Harmon. Data collection techniques. Journal-American Academy Of Child And Adolescent Psychiatry, 40(8):973–976, 2001.

[18] J. Damerau Nitin, Indurkhya; Fred. Handbook of natural language process-

ing. Chapman Hall/CRC Machine Learning Pattern Recognition Series, 2010.

[19] Huttner Steven. Sentiment analysis: A practitioner's guide. 2019.

[20] Hassan; Hoda Korashy Walaa, Medhat; Ahmed. Sentiment analysis algorithms and applications: A survey. Ain Shams Engineering Journal, 2014.

[21] Yelena Mejova. Sentiment analysis: An overview. University of Iowa, Computer Science Department, 2009.

[22] Jasmina Dj Novaković, Alempije Veljović, Siniša S Ilić, Željko Papić, and Milica Tomović. Evaluation of classification models in machine learning. Theory and Applications of Mathematics & Computer Science, 7(1):39, 2017.

[23] Scikit-learn machine learning in python. URL https://scikit-learn.org/stable/.

[24] Maulen Makhul. Comparison of different classification models for sentiment analysis. Demirel University Bulletin: Natural and Technical Sciences, (2): 5–13, June 2023.

Appendix A

Appendix

Listing A.1: The 20 most influence words for the prediction

```
Set an empty list called feature_names
Set an empty list called svm_weights
Set an empty dictionary called weights_dict

Set feature_names to list of feature names obtained from
   vectorizer.get_feature_names_out()
Set svm_weights to the flattened array obtained from svc.
   coef_.toarray()
Set weights_dict to a dictionary created by zipping
   feature_names and svm_weights

Print "The 20 most influential words for the prediction:"

Set sorted_words to the list of words in weights_dict,
   sorted by their values in descending order, taking only
   the first 20 words

For each word in sorted_words:
    Print the word and its corresponding weight from
      weights_dict
```

Listing A.2: The 10 most influential Positive words for the prediction

```
Set an empty list called negative_words
Set an empty list called positive_words

For each word in weights_dict:
    If weights_dict[word] is greater than 0, then:
        Append word to positive_words list
    Else:
        Append word to negative_words list

Print "The 10 most influential Positive words for
  prediction"

Set sorted_positive_words to the list of words in
  positive_words, sorted by their values in ascending
  order

For each word in sorted_positive_words, taking only the
  first 10 words:
    Print the word and its corresponding weight from
      weights_dict
```

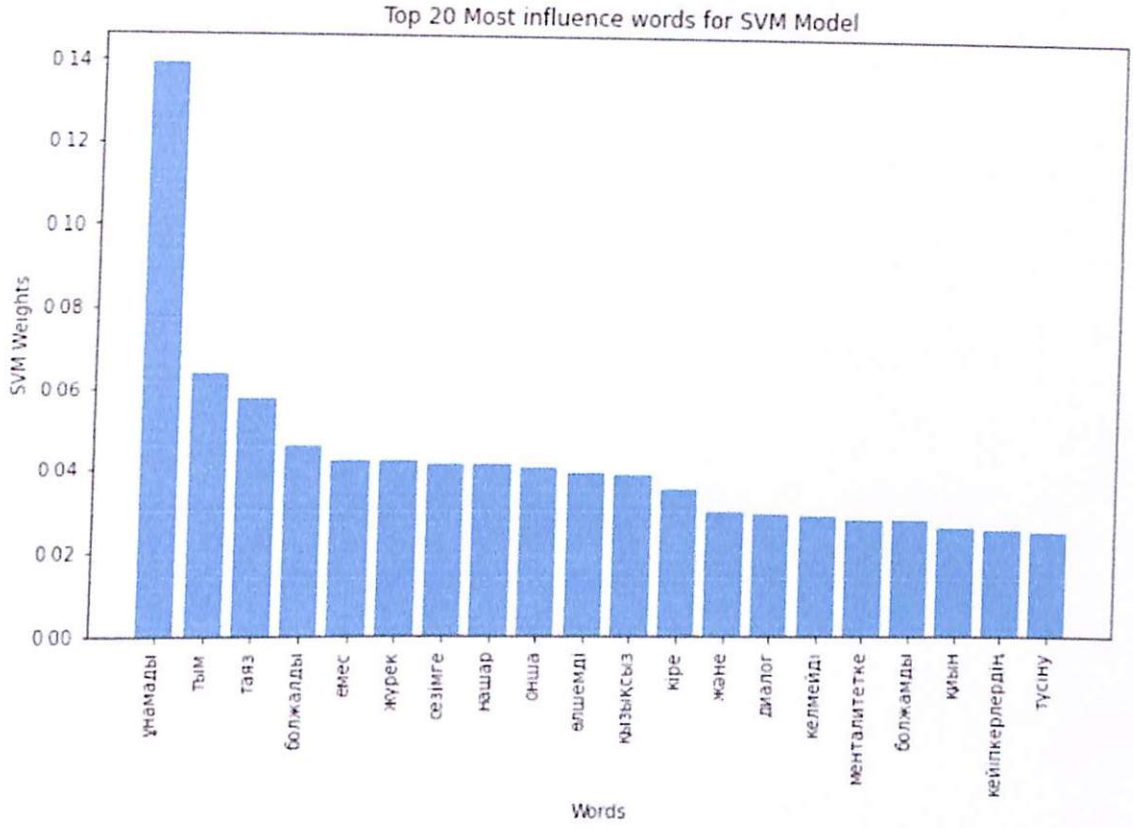

Figure A.1: Top 20 Most influence words for the prediction of the SVM Model

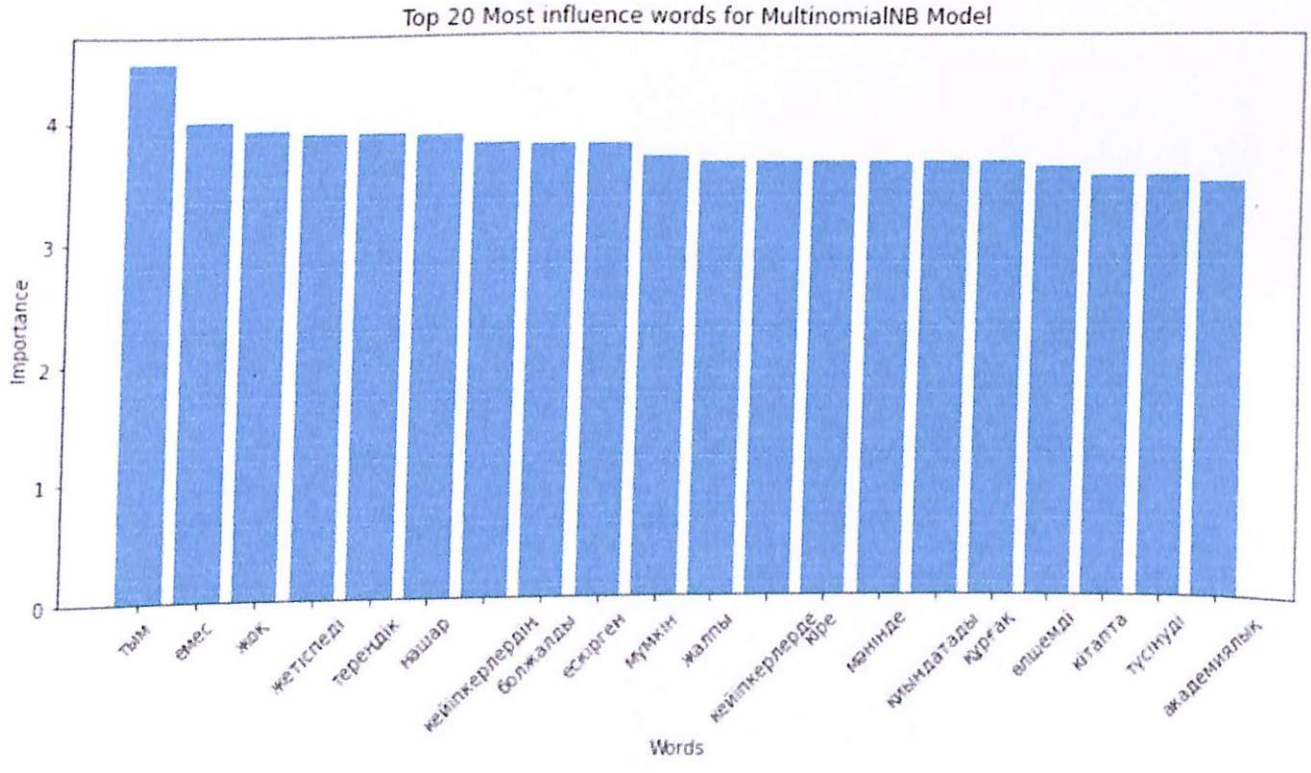

Figure A.2: Top 20 Most influence words for the prediction of the MultinomialNB Model

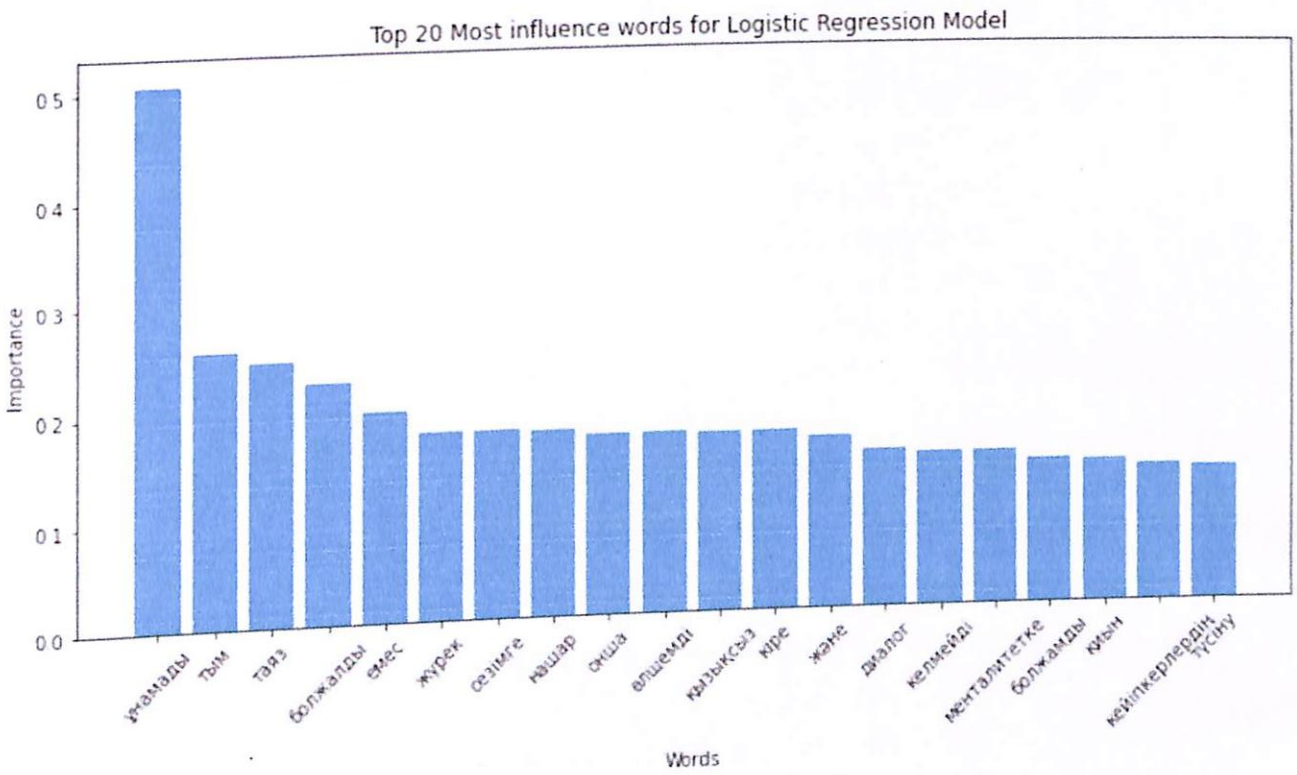

Figure A.3: Top 20 Most influence words for the prediction of the Logistic Regression Model

Table A.1: Research results table most influential words for prediction

Models	Positive Words
SVM	керемет (awesome) 0.09107522535870648 тың 0.060717484742525446 жақсы (good) 0.058282666431600696 ұнады (liked) 0.04786075054444368 мені (me) 0.04146780101256813 триллерлік (thriller) 0.039181946680994256 ұлы (great) 0.03601023963397543 ғажайып (amazing) 0.03229746319213919 сүйкімді (lovely) 0.03221715071411773 қайғылы (sad) 0.032202413436524455
Logistic Regression	керемет (awesome) 0.5050926123848026 мені (me) 0.25923679396779313 жақсы (good) 0.22678988389950344 тың 0.20804214070902619 триллерлік (thriller) 0.17482103607435304 әсер (effect) 0.16752868491675885 сүйкімді (nice) 0.16685636152892308 кітап (book) 0.15844974583307192 ұлы (great) 0.15733187075195731 кітапты (book) 0.1553806010803329
Models	Negative Words
SVM	ұнамады (didn't like) -0.13429591878408734 келешегі (the future) -0.061783627391424255 таяз (shallow) -0.05812047538614953 қызықсыз (boring) -0.04536975780441565 емес (not) -0.04378601863906304 болжалды (estimated) -0.0423126255776497 онша -0.038141456521928616 тым (too) -0.038035722380839854 кіре (input) -0.03747320253921703 өлшемді (size) -0.0359262022706327
Logistic Regression	Ұнамады (didn't like) -0.2913357129391812 тым (too) -0.23923679396779313 таяз (shallow) -0.22678988389950344 болжалды (estimated) -0.2250930820069513 емес (not) -0.19779787576739197 жүрек (heart) -0.17701996409640824 сезімге (to sense) -0.17701996409640824 нашар (bad) -0.1745529791762287 онша (so) -0.17029593483891384 өлшемді (size) -0.17010131735642617