Advanced Recipe Recommendation System with Machine and Deep Learning

Personalized User Experience

Table of Contents

List of Figures

Acknowledgment

I want to express my sincere gratitude to all those who supported me throughout this project.

I am thankful to my project supervisor Professor. Karim Ouazzane, for his valuable guidance, constructive feedback, and constant encouragement at every stage of the project. His insights and expertise have been instrumental in shaping the direction of my work.

I am also deeply grateful to my professors and classmates for their continuous support, and for providing the resources and facilities necessary to carry out this research.

A special thanks goes to my friends and family, whose unwavering support and patience helped me stay focused and motivated throughout this process. I am especially grateful to my parents, who have always done their best to provide me with a good education and opportunities they did not have. I would not have achieved what I have so far without their endless support and dedication, and I will always be thankful for it.

I would also like to acknowledge the London Met for their assistance, and the UCI Machine Learning Repository for providing the data that made this project possible. I extend my appreciation to the users who contributed valuable feedback, helping me to refine the recommendation system.

Thank you all for your contributions, without which this project would not have been possible.

Abstract

This study develops a sophisticated recipe recommendation system designed to enhance user experience by offering highly personalized recipe suggestions. Leveraging advanced computational techniques, the project utilizes sentiment analysis and a combination of recommendation algorithms to address the challenge of recipe discovery amidst the vast array of options available online.

In the digital age, where online platforms overflow with recipe options, users often struggle to find dishes that align with their individual preferences and dietary requirements. This study addresses this challenge by developing an advanced recipe recommendation system designed to improve user experience through personalized suggestions. Utilizing a comprehensive dataset that includes user reviews and recipe ratings, our approach integrates cutting-edge computational techniques to deliver more relevant and engaging recipe recommendations.

Employing a blend of sentiment analysis, collaborative filtering, and content-based filtering methodologies, this project aims to uncover patterns and preferences that drive user satisfaction. Sentiment analysis categorizes user feedback into positive, neutral, or negative sentiments, providing critical insights into user preferences and improving the recommendation process. Meanwhile, collaborative and content-based filtering techniques work in tandem to offer suggestions tailored to individual tastes and dietary requirements.

The dataset used in this study encompasses both textual review data and numerical features such as star ratings, creating a comprehensive foundation for analysis. The integration of these diverse data sources enables the system to deliver accurate and relevant recipe suggestions, enhancing user satisfaction and engagement on food platforms. By synthesizing both numerical and textual data, the system refines its recommendations, making meal planning simpler and more enjoyable for users.

Our research not only enhances the accuracy of recipe suggestions but also informs the design of user-centric interfaces and APIs that integrate seamlessly with online platforms. The findings of this study highlight the efficacy of combining sentiment analysis with recommendation algorithms to improve recipe recommendations. By addressing the complexities of user preferences and dietary restrictions, the project aims to inform the development of more effective and personalized food recommendation systems. This research underscores the potential for advanced recommendation technologies to transform user experiences and drive engagement in the digital food landscape.

Keywords: Recipe Recommendation System, Sentiment Analysis, Feature Engineering, User Reviews, Recipe Ratings, Collaborative filtering, Content-based filtering, Natural Language Processing, Deep Learning, Machine learning Models, API Integration, Recurrent Neural

Network, Power BI, Personalization, User Experience, Data integration, Engagement, Dietary needs, Real-Time Recommendations, User Feedback, Recipe Discovery, Recommendation Accuracy.

Chapter 1: Introduction

1.1. Introduction:

In this fast-paced world of online food platforms, it's necessary to understand what users want and help them find recipes that suit their tastes. With so many recipes available, users can struggle to discover dishes that match their personal preferences and dietary needs. If a user is looking for gluten-free vegan recipes may find it challenging to sort through a vast array of options to find suitable dishes. This project tackles this problem by creating a personalized recipe recommendation system. The system is designed to improve the user experience and make recipe discovery easier. By using advanced techniques such as sentiment analysis to gauge user opinions and recommendation algorithms to tailor suggestions, the system provides customized recipe recommendations. For Example, users frequently review and rate vegan dishes positively, the system will prioritize similar recipes in their suggestions, ensuring that recommendations are closely aligned with their tastes and dietary requirements.

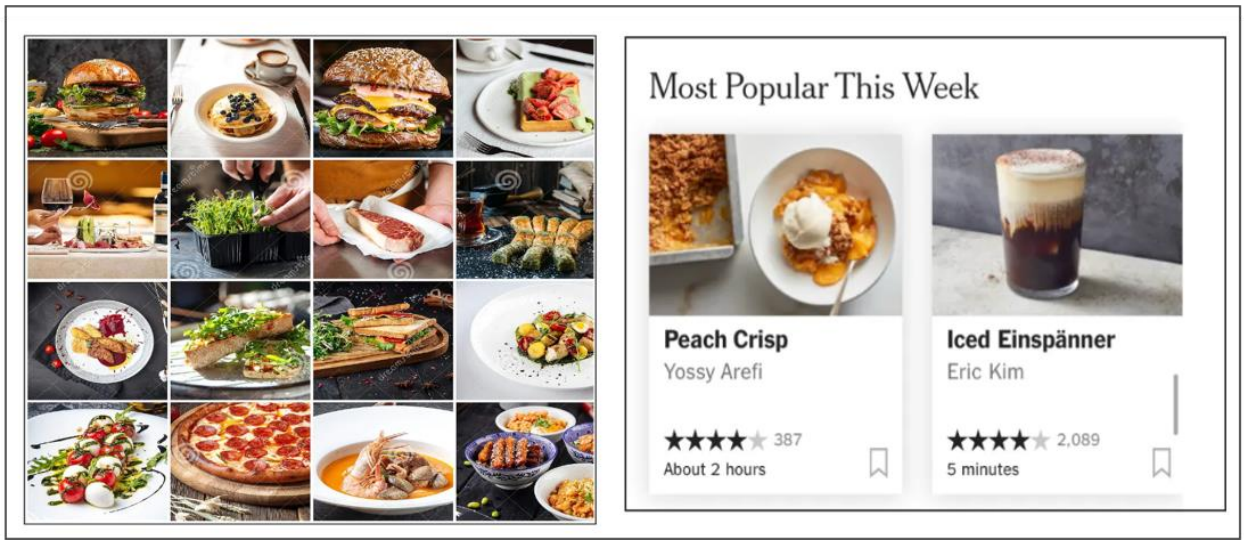


Figure 1: Recommended Recipes

Here the Sentiment analysis plays a major role in this system by evaluating user reviews and feedback to understand how users feel about various recipes. The user frequently leaves positive reviews for spicy dishes, the system can use this information to recommend other highly-rated spicy recipes. This helps users find recipes that they are more likely to enjoy based on their past interactions.

The recommendation algorithms are designed to suggest based on individual preferences and dietary needs. For example, if a user prefers vegetarian dishes or follows a gluten-free diet, the system will prioritize recipes that meet these criteria. This personalized approach ensures that users

receive recommendations that are not only relevant but also catered to their specific dietary requirements.

To further illustrate, imagine a user who is trying to find family-friendly dinner ideas. The system can analyze past user interactions, such as ratings and search history, to suggest popular recipes among families and receive high ratings for taste and ease of preparation. This not only makes recipe discovery more efficient but also enhances the overall satisfaction of users by providing them with recipes that suit their lifestyles and preferences.

This project is the development and implementation of a personalized recipe recommendation system. With the vast amount of available recipes online, users often find it challenging to discover dishes that align with their preferences and dietary needs. This system is designed to recommend recipes based on user preferences, sentiment from user reviews, and other factors such as star ratings. The system aims to enhance user experience and improve recipe discovery by leveraging sentiment analysis and recommendation algorithms.

This research intends to transform the way users discover recipes by combining sophisticated sentiment analysis with personalized recommendation algorithms. By doing so, it addresses the common challenge of finding recipes that fit individual tastes and dietary needs, ultimately improving the user experience on online food platforms.

1.2. Business Understanding:

In today's digital age, the food industry is rapidly expanding on online platforms. With the rise of food blogs, recipe-sharing websites, and mobile applications dedicated to cooking, users have access to an overwhelming number of recipes. However, despite this abundance of options, users often struggle to find dishes that align with their tastes, dietary restrictions, and lifestyle preferences. This creates a gap in the user experience, where many find themselves spending considerable time searching for relevant recipes that meet their specific needs.

People are constantly seeking quick and effective solutions to their daily challenges. One common challenge is deciding what to cook for meals. While many individuals enjoy cooking, others find it difficult to come up with new and exciting dishes regularly. This difficulty is especially pronounced given the vast array of recipes available online, which can be overwhelming and time-consuming to navigate.

This is where a recipe recommender system becomes highly valuable. Such a system addresses the problem of recipe discovery by providing personalized recommendations based on user preferences, dietary needs, and past behavior. By suggesting recipes that align with individual tastes and restrictions, the system makes meal planning simpler and more enjoyable.

From a business perspective, implementing a recipe recommendation system offers significant benefits. For online food platforms, a well-designed recommender system can enhance user experience by making it easier for users to find dishes that meet their specific needs. This personalized approach not only increases user satisfaction but also drives higher engagement and retention rates.

A recipe recommender system is a type of recommendation system that suggests recipes to users based on their preferences, history, and other factors. The system is typically designed to provide users with a personalized set of recipe recommendations that are tailored to their specific needs and tastes.

In the competitive landscape of online food platforms, understanding user preferences is essential for delivering a superior user experience. The ability to recommend recipes that align with users' tastes and dietary restrictions can significantly enhance engagement and satisfaction. As online food platforms continue to grow, offering personalized recommendations becomes a crucial differentiator.

To build a recipe recommender system, we need to collect data from users, including their search queries, recipe ratings, and other information. Then machine learning algorithm is used to analyze this to create a personalized recipe recommendation system.

This project is designed to address the challenge of recipe discovery by providing a solution that not only meets users' culinary preferences but also adheres to their dietary needs. By leveraging sentiment analysis and recommendation algorithms, the system aims to boost user satisfaction, increase user retention, and drive engagement with the platform. For businesses, this means not only enhancing the user experience but also potentially increasing customer loyalty and driving growth through improved recommendation accuracy and relevance.

1.3. Aims and Objectives:

The goal of this project is to build a robust recipe recommendation system that improves the user experience by offering highly personalized suggestions based on sentiment analysis of user reviews. By analyzing both the textual content of the reviews and the numerical features (such as star ratings), the system can better understand user preferences and offer more relevant recommendations.

Project Objectives:

- *Sentiment Analysis:* Develop a model capable of accurately classifying user reviews into positive, neutral, or negative categories. This sentimental information will serve as a key feature in personalizing recommendations.
- *Recommendation System Development:* Design and implement a system that uses sentiment analysis and recommendation algorithms to provide personalized recipe suggestions based on user preferences and needs.
- *Star Rating Prediction:* Use deep learning models to predict star ratings based on user feedback and reviews, improving the system's ability to recommend recipes that users are likely to enjoy.
- *Integration of Numerical and Textual Features:* Combine both numerical (e.g., user reputation, thumbs up/down) and textual (review content) data to enhance the accuracy of the recommendations.

* *Visual Insights Using Power BI:* Leverage data visualization tools to present actionable insights from user reviews, model performance, and recommendation outcomes, making the system's benefits accessible to business stakeholders.
* *Enhance User Experience:* Create a user-friendly interface that simplifies the process of discovering recipes and allows users to easily access personalized recommendations.
* *API Implementation for Real-Time Use:* Create a RESTful API that allows easy integration of the recommendation system into websites or apps, ensuring real-time, scalable functionality.
* *Evaluate System Performance:* Assess the effectiveness of the recommendation system through user feedback and performance metrics, and make necessary adjustments to optimize its accuracy and relevance

By achieving these objectives, the project will deliver a practical solution that food-related platforms can use to improve customer experience, engagement, and overall satisfaction. This system will also serve as a foundation for future enhancements, such as including more complex dietary preferences or integrating external data sources (seasonal ingredients or trending recipes).

Chapter 2: Literature Review

2.1. Evolution of Recommendation Systems

Recommendation systems have significantly evolved over the past few decades, becoming an integral part of various online platforms. Initially, these systems were simple and based on basic algorithms that relied on manually curated rules to suggest items to users. [1]. However, as technology advanced and the volume of data available for analysis grew exponentially, recommendation systems have become increasingly sophisticated.

A recommendation system is a type of machine-learning tool that uses data to help people find what they want from a vast number of choices. These systems learn about people's preferences, past decisions, and characteristics by analyzing their interactions, such as clicks, likes, and purchases. Because they can predict what consumers might like or need on a very personalized level, content and product providers widely use recommendation systems. They guide consumers to products or services that match their interests, whether it is books, videos, health classes, or clothing.

A recommendation system is an artificial intelligence (AI) algorithm, usually associated with machine learning, which leverages Big Data to suggest or recommend or suggest additional products to customers. These recommendations are based on various criteria, including past purchases, search history, demographic information, and other relevant factors. [2]. The ability to process and analyze large datasets in real time has enabled recommendation systems to deliver highly personalized and accurate suggestions, thereby enhancing user experience and engagement.

The early models of recommendation systems, such as content-based filtering, focused on matching user preferences with product attributes. Collaborative filtering, another widely used technique, shifted the focus to understanding the behavior of users with similar tastes and preferences. [3]. Over time, these methods have been combined to create hybrid systems that can offer more comprehensive and nuanced recommendations. Hybrid recommendation systems, which merge collaborative and content-based filtering, address these issues through weighted averages and feature combinations. In recipe recommendation systems, hybrid models are particularly beneficial due to their ability to manage diverse preferences and complex recipe attributes. Recent advancements have integrated nutritional information, dietary restrictions, and user reviews, enhancing user satisfaction and engagement in recipe recommendations.

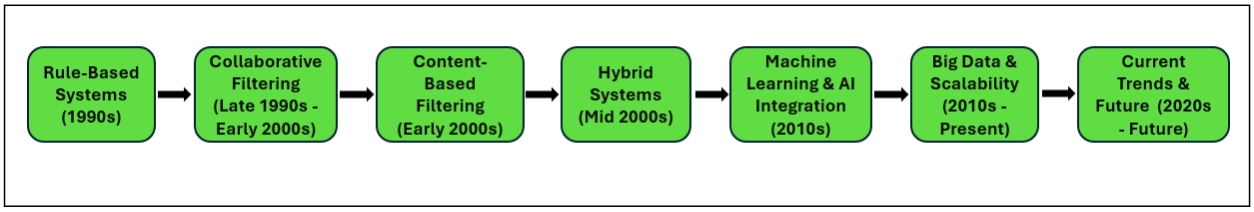


Figure 2: Recommendation Systems Evolution

Today, recommendation systems are powered by advanced machine learning techniques, including deep learning and neural networks, allowing them to process unstructured data like text, images, and even sentiments expressed in user reviews [4]. This evolution has been driven by the growing demand for personalized user experiences and the need to efficiently manage diverse and complex datasets.

2.2. Foundations of Predictive Analytics

The evolution of recommendation systems is a testament to the broader field of predictive analytics, which lies at the heart of these technologies. Predictive analytics involves the use of statistical techniques, machine learning algorithms, and data mining processes to analyze historical data and make predictions about future events [5]. In the context of recommendation systems, predictive analytics enables the anticipation of user preferences, driving personalized suggestions that enhance user engagement and satisfaction. This predictive capability is the cornerstone of modern recommendation systems, including those for recipes.

Predictive analytics, at its core, is the science of using historical data to make informed predictions about future outcomes. This discipline emerged from the intersection of statistics, data mining, and machine learning, all of which have evolved significantly over the years [6].

The journey begins with data. Data is the raw material, the lifeblood of predictive analytics. In the early days, data was scarce and often siloed within organizations. But as technology advanced, the ability to collect, store, and process data grew exponentially [7]. Suddenly, organizations were awash in data from transactions, customer interactions, social media, sensors, and more. The challenge shifted from gathering data to making sense of it.

Enter statistics is the first tool in the predictive analytics toolbox. Traditional statistical methods, such as regression analysis, provided a way to model relationships between variables and predict outcomes. In the early 20th century, insurance companies used actuarial science, a form of predictive analytics, to estimate life expectancy and set premiums. These methods were effective but limited in scope, relying heavily on assumptions about the data.

As the volume of data increased, the need for more sophisticated tools. Data mining emerged as the next phase in the evolution of predictive analytics. Unlike traditional statistics, data mining could oversee large datasets and uncover patterns that were not immediately obvious [8].Techniques such as clustering, classification, and association rule learning allowed analysts to discover hidden relationships within the data, leading to more accurate predictions.

But data mining was just the beginning. The real breakthrough came with the advent of machine learning. Unlike traditional statistical models, which require explicit instructions from the analyst, machine learning algorithms can learn from data. They can adapt, improve, and even uncover patterns that humans might miss. This capability transformed predictive analytics, enabling it to tackle complex problems like image recognition, natural language processing, and, of course, recommendation systems.

Nowadays, predictive analytics is at the heart of many industries, including healthcare, finance, marketing, and retail. Its ability to forecast outcomes and guide decision-making is invaluable, especially in the fast-paced digital landscape where user preferences and behaviors can change rapidly [9]. The integration of predictive analytics in recommendation systems has made it possible to provide users with highly personalized experiences, making the systems more accurate, relevant, and effective.

2.3. Role of Machine Learning in Recommendation Systems

Machine learning is central to the development of recommendation systems, enabling the creation of models that learn from data to predict user preferences [10]. These models fall into three primary learning paradigms they are supervised, unsupervised, and reinforcement learning. Supervised learning, extensively used in recommendation systems, trains models on labeled data, making it suitable for predicting user ratings based on past behavior. Algorithms like matrix factorization and neural collaborative filtering have significantly improved the accuracy of predictions by uncovering latent factors in user-item interactions [11].

Unsupervised learning uncovers hidden patterns in data without labeled examples, which is particularly useful for clustering similar users or items to refine recommendations. Techniques such as k-means clustering or hierarchical clustering help group users with similar tastes, enabling the system to suggest items that have been popular among similar users [12]. This approach is highly beneficial in scenarios where explicit user feedback is limited, allowing the system to make informed recommendations even with sparse data.

Although less common, reinforcement learning is gaining traction for its potential in adaptive recommendation systems. In this paradigm, models learn by interacting with users and adjusting recommendations based on feedback. This is particularly valuable in dynamic environments where user preferences change over time. Reinforcement learning algorithms can optimize long-term user engagement by continually updating the model based on the rewards (or feedback) received from users [13].

Machine learning has become the backbone of modern technology, which significantly enhanced the accuracy and effectiveness of recommendation systems, revolutionizing the way these systems operate by enabling them to learn from data and improve over time. Unlike traditional recommendation methods, which rely heavily on predefined rules and static algorithms, machine learning empowers recommendation systems to dynamically adapt to users' evolving preferences and behaviors.

At the core of machine learning in recommendation systems is the ability to model complex relationships within data. Collaborative filtering benefits from advanced matrix factorization techniques like Singular Value Decomposition (SVD) and neural collaborative filtering, which can uncover latent factors that influence user-item interactions [14]. This allows the system to make more accurate predictions even with sparse data.

Content-based filtering, which traditionally relied on matching user preferences with item attributes, has also been enhanced by machine learning. Algorithms such as decision trees, support vector machines (SVM), and more recently, deep learning models, can process and analyze vast amounts of unstructured data, including text, images, and videos. This capability is particularly valuable in domains like recipe recommendation, where user preferences are influenced by numerous factors, including ingredients, cooking methods, and nutritional information.

Deep learning, a subset of machine learning, has further pushed the boundaries of recommendation systems. Techniques like Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) enable the processing of complex data types such as images and sequential data, respectively [15]. For example, CNNs can analyze images of dishes to identify visual similarities, while RNNs can capture temporal patterns in user behavior, such as changes in dietary preferences over time. These models can also incorporate sentiment analysis, allowing the system to factor in user emotions expressed in reviews or feedback, leading to more personalized recommendations.

Another significant contribution of machine learning to recommendation systems is its ability to handle large-scale, real-time data. Techniques like online learning and reinforcement learning allow recommendation systems to update their models continuously as new data arrives. This is crucial for maintaining relevance in dynamic environments, where user preferences and market trends can shift rapidly. Reinforcement learning, in particular, is valuable for optimizing long-term user engagement by learning from user interactions and adjusting recommendations accordingly.

Moreover, machine learning enables the creation of hybrid recommendation systems, which combine multiple recommendation techniques to leverage their respective strengths [16]. For instance, a hybrid system might use collaborative filtering to identify similar users and content-based filtering to match users with items that align with their specific tastes. Machine learning models can intelligently weigh and blend these different approaches, resulting in a system that is more robust and capable of handling diverse user preferences.

2.4. Advancements in Sentiment Analysis

Sentiment analysis has emerged as a crucial tool for enhancing user experience, particularly on user-generated content, such as product reviews and social media feedback. The journey of sentiment analysis began with traditional methods that primarily depended on lexicon-based approaches [17]. These methods were relatively simple, using predefined dictionaries of positive and negative words to determine the sentiment of a piece of text. While they offered a straightforward and accessible means of gauging sentiment, they often struggled to capture the nuances of language, such as sarcasm, idioms, or cultural references. This limitation frequently leads to misinterpretations, where the actual true sentiment expressed by users behind a user's words is lost or misconstrued.

To address these limitations and recognize these challenges, the field began to shift towards more sophisticated, machine-learning-based techniques. These new methods brought a level of depth and accuracy that lexicon-based approaches could not achieve. Algorithms such as Naive Bayes, Support Vector Machines (SVM), and Logistic Regression became widely adopted for sentiment analysis, leveraging the power of data to improve understanding. By training models on labeled datasets, these techniques allowed the systems to learn from examples and better recognize and categorize sentiment patterns [18]. This marked a significant step forward in managing the variability and complexity inherent in natural language, making sentiment analysis more accurate dependable, and nuanced.

The advent of deep learning further revolutionized sentiment analysis, pushing the boundaries of what was possible. Models such as Long Short-Term Memory (LSTM) networks, Convolutional Neural Networks (CNNs), and transformers like BERT (Bidirectional Encoder Representations from Transformers) brought unprecedented capabilities to the field [19]. These models excel at capturing complex linguistic patterns and contextual dependencies within text, allowing for a much deeper understanding of sentiment. For example, LSTM networks, with their ability to process sequential data, proved particularly effective in analyzing longer reviews or sentences, where context is key to determining sentiment. Meanwhile, the CNN model originally designed for image processing, found a new application in identifying key phrases or words within text that significantly influence overall sentiment.

The introduction of transformers, particularly BERT, marked the latest advancement in sentiment analysis. BERT's use of attention mechanisms allows it to understand the context of each word concerning all other words in a sentence, providing a more holistic and accurate interpretation of sentiment [20]. This capability is especially useful in recommendation systems, where understanding the subtle differences in user feedback can lead to more personalized and precise suggestions.

Incorporating sentiment analysis into recommendation systems has significantly refined the way these systems build user profiles. By integrating the sentiments expressed in user reviews, recommendation systems can prioritize items that have been positively reviewed, while filtering out those with negative feedback. In our recipe recommendation system, sentiment analysis can be used to rank recipes not just by their popularity, but by the sentiment of the reviews they receive. This ensures that only recipes with high satisfaction levels are suggested to users, leading to a more tailored and satisfying user experience.

Beyond refining recommendations, advanced sentiment analysis techniques provide valuable insights into user preferences and pain points. By identifying common areas of dissatisfaction, businesses can better tailor their products and services to meet customer expectations, further enhancing the effectiveness of recommendation systems.

As we look to the future, the role of sentiment analysis in recommendation systems is likely to expand even further. With the integration of real-time feedback and more sophisticated natural language processing, these systems will become increasingly responsive and personalized. The ongoing advancements in sentiment analysis represent a significant step towards making

recommendation systems more user-centric, aligning suggestions more closely with the emotions and preferences of users.

2.5. Integration of Hybrid Recommendation Models

The integration of hybrid recommendation models represents a significant advancement in the field of recommendation systems, combining the strengths of different approaches to deliver more accurate and personalized suggestions [21]. The evolution of recommendation systems has seen the transition from simple methods to hybrid models that effectively address the limitations of individual techniques or approaches.

Early recommendation systems primarily relied on collaborative filtering and content-based filtering. Collaborative filtering, which predicts user preferences based on past behavior and the behavior of similar users, faced challenges such as the cold start problem, where new users or items lacked sufficient data for accurate predictions. Content-based filtering, which recommends items based on their attributes and the user's profile, was constrained by its reliance on detailed item descriptions and struggled with scalability issues [22].

To overcome these limitations, hybrid recommendation models were developed. These models combine collaborative filtering and content-based filtering to leverage the advantages of both approaches. By integrating collaborative filtering's ability to recognize patterns from user behavior with content-based filtering's detailed analysis of item attributes, hybrid models provide a more comprehensive recommendation system. This integration helps address issues like data sparsity and improves recommendation quality.

There are several strategies for integrating these approaches. Weighted hybrid models combine predictions from both collaborative and content-based filtering, using a weighted average to balance their contributions based on their effectiveness in different contexts [23]. Switching hybrid models dynamically selects the most suitable approach based on the specific situation or user behavior. For instance, a switching model might use collaborative filtering for users with extensive historical data and content-based filtering for new users with limited data. Feature combination models merge attributes from both methods into a unified model, enhancing the depth of the recommendations by incorporating both user behavior patterns and item characteristics.

In our recipe recommendation systems, hybrid models are particularly advantageous. They effectively handle diverse user preferences and the complex nature of recipes. By utilizing hybrid models, recipe recommendation systems can provide more accurate suggestions that align with users' tastes and reviews.

Recent advancements have further refined hybrid recommendation models by incorporating sentiment analysis. This integration allows recommendation systems to consider user feedback and opinions, adding a layer of personalization [24]. By analyzing the sentiment of user reviews, hybrid models can prioritize recipes with positive feedback and adjust recommendations based on user satisfaction levels. This integration enhances the relevance of recommendations and improves the overall user experience.

The combination of hybrid recommendation models with advanced techniques like deep learning and sentiment analysis represents cutting-edge recommendation system development. These models are increasingly able to provide highly personalized and contextually relevant suggestions, making them invaluable tools in various domains, including recipe recommendations. As technology continues to advance, hybrid models will evolve further, incorporating more data sources and analytical methods to deliver even more precise and engaging recommendations.

Chapter 3: Methodology

3.1. Software Environment

To ensure the success of this project, the first step was selecting the right software to meet all analysis requirements. Choosing the appropriate software is important because it impacts the project's success and future growth. For this project, Python was chosen due to its versatility and rich ecosystem of libraries. Google Colab (Google Colaboratory) was used as the primary environment. Google Colab is a cloud-based service with an easy-to-use interface, especially when running Python code in a web browser. Since it is a Jupyter Notebook-as-a-service version, users may quickly generate interactive notebooks using code, text, and visualizations. This cloud-based capability made it easier to manage data processing and model training without local resource constraints [25].

To facilitate the development and implementation of the recipe recommendation system, I utilized a combination of powerful tools and platforms. The entire project was developed in Python, primarily within the Google Colab environment, which acts as a flexible workspace for data processing and model training. Python was chosen for its extensive library support and versatility, allowing seamless integration of various data science libraries.

For developing the recipe recommendation system, we utilized a variety of tools and technologies to ensure a robust and efficient workflow. Python was the primary programming language due to its extensive libraries for machine learning, and natural language processing. Google Colab provided an ideal environment for collaborative coding and model development, which facilitated the handling of large datasets and complex models. Additionally, Power BI was employed for data visualization and reporting, enabling the creation of interactive dashboards that present the system's findings in an accessible manner. These tools combined to create a flexible and scalable environment, crucial for the iterative process of model development and refinement.

In Python, several libraries were essential for different tasks. Powerful libraries such as TensorFlow and Scikit-learn were employed for building and tuning the machine-learning models, while Pandas and NumPy were essential for data manipulation and preprocessing [26]. For visualizing data and model outcomes, Matplotlib and Seaborn were used. The combination of these

tools provided a robust software environment capable of handling the project's complexity and scale.

3.2. Data Analytics Lifecycle

The methodology for this project adhered to the Data Analytics Lifecycle, a structured approach that guided the entire process from inception to deployment. The lifecycle began with the Discovery phase, where the primary business goals were defined, and key stakeholders' expectations were gathered. This foundational step was followed by the Data Preparation phase, involving extensive data cleaning, transformation, and feature engineering to ensure the datasets were ready for analysis.

The Data Analytics Lifecycle, as a whole, ensures a thorough and systematic process, crucial for achieving accurate results and aligning the analysis with business objectives. It encompasses several stages, each playing a vital role in the overall success of the project by guiding the transformation of raw data into actionable insights.

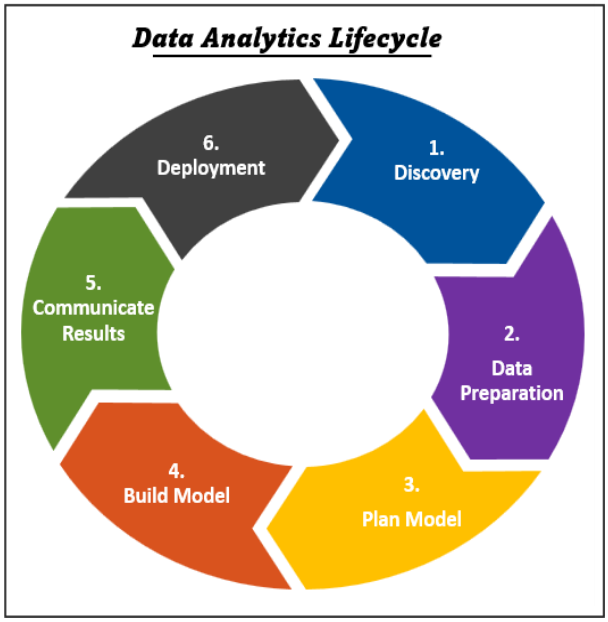


Figure 3: Data Analytics Lifecycle

1. **Discovery:**

The journey begins with the Discovery phase. In this initial stage, the primary business goals are defined, and the expectations of key stakeholders are gathered. This phase is critical as it lays the foundation for the entire analysis, setting clear objectives and understanding the problem or opportunity at hand. It also involves exploring available data sources to assess their relevance and feasibility for the project.

This phase involved understanding the project's objectives, defining the problem space, and identifying the key stakeholders. We conducted a thorough needs assessment to align the project goals with the stakeholders' expectations.

2. **Data Preparation:**

Data Preparation phase transforms raw data into a format suitable for analysis. In this stage, raw data is collected, cleaned, and transformed to make it suitable for analysis [27]. This involves several key tasks, including data collection from various sources such as databases and APIs, followed by data cleaning to address issues like missing values and duplicates. Data transformation is also performed to normalize and structure the data appropriately. These steps ensure that the data is reliable and ready for detailed analysis.

After acquiring the necessary data, we focused on cleaning and preprocessing it to ensure quality and consistency. This step included handling missing values, normalizing data formats, and encoding categorical variables.

3. **Plan Model:**

During the Model Planning phase, potential algorithms and analytical models are evaluated to match the project's specific requirements. This stage involves defining the models to be used and determining the appropriate evaluation metrics to measure their performance [28]. By carefully planning the models and metrics, this phase ensures that the chosen approaches align with the goals of the analysis.

In this phase, we selected appropriate modeling techniques based on the data characteristics and project requirements. We planned the integration of collaborative filtering, content-based filtering, and sentiment analysis into a hybrid recommendation system.

4. **Build Model:**

The model-building phase is where the selected models are developed, trained, and validated using the prepared data. Training involves applying algorithms to the data to create predictive models or uncover patterns. Model tuning is also conducted to optimize performance and accuracy. This phase is pivotal as it directly impacts the effectiveness of the resulting models.

Here, the actual implementation of the models took place. We developed and fine-tuned the machine learning models, incorporating techniques such as Singular Value Decomposition (SVD) for collaborative filtering, cosine similarity for content-based filtering, and Recurrent Neural Networks (RNNs) for sentiment analysis.

5. **Communicate Results:**

Once the models are built, the Communicating Results phase focuses on presenting the findings in a clear and actionable manner. This includes creating visualizations such as charts and graphs to represent the data and insights and preparing reports or summaries to explain the analysis and recommendations to stakeholders. Effective communication of results is essential for ensuring that the insights are understood and can be acted upon.

The results of the models were communicated through detailed reports and visualizations. Power BI played a significant role in this phase, allowing stakeholders to interact with the data and understand the implications of the model's predictions.

6. **Deployment/Apply Live:**

The final stages involve Deployment and Monitoring. During Deployment, the models and insights are integrated into business processes or systems, enabling real-time predictions or decisions. This phase also includes ensuring seamless integration with existing workflows. Post-deployment, the Monitoring phase is crucial for maintaining model performance [29]. It involves tracking the performance of models and updating them as needed based on new data or changing requirements. This ongoing evaluation ensures that the models remain accurate and relevant over time.

Finally, the recommendation system was deployed in a test environment, where its performance was monitored and evaluated. Continuous monitoring allowed for ongoing refinement and optimization, ensuring the system remained accurate and relevant.

Each stage of the Data Analytics Lifecycle plays a crucial role in ensuring that the analysis is comprehensive, dependable, and aligned with the goals of the project. By following this structured approach, organizations can derive valuable insights from their data and make informed decisions.

3.3. Data Collection and Preparation

The foundation of any robust machine learning model lies in the quality and relevance of the data utilized. For this project, the dataset was sourced from the UCI Machine Learning Repository [30], a well-known resource that provides diverse and high-quality datasets suitable for various machine learning applications. The chosen dataset was particularly relevant to the problem domain, offering a rich compilation of information necessary for developing an effective and useful recipe recommendation system.

The dataset encompassed a comprehensive range of variables, including user reviews and detailed recipe metadata. These features provided a multifaceted view of each recipe, capturing not only the factual components such as subjective user experiences and preferences through reviews. This diversity in data was important for building a recommendation system capable of understanding and predicting user tastes and delivering personalized suggestions.

Upon collection, an extensive Data Cleaning process was undertaken to ensure the integrity and reliability of the dataset. This process involved several steps:

- ***Handling Missing Values:*** Identifying and appropriately addressing any gaps in the data to prevent biases and inaccuracies in the model's predictions [31]. Techniques such as imputation were employed where suitable, replacing missing entries with meaningful estimates based on available data.
- ***Correcting Inconsistencies:*** Ensuring uniformity across the dataset by rectifying discrepancies in data entries. This included standardizing units of measurement, correcting typographical errors, and ensuring consistent formatting across all records.
- ***Standardizing Formats:*** Converting data into consistent and compatible formats to facilitate seamless processing and analysis. For example, text data was standardized in terms of case and punctuation, and numerical data was formatted uniformly.

Following data cleaning, the dataset underwent thorough Preprocessing to prepare it for effective model training and evaluation. Key preprocessing steps included:

- ***Normalization:*** Scaling numerical features to a common range to eliminate disparities caused by differing scales and units, thereby improving the convergence and performance of machine learning algorithms.
- ***Encoding Categorical Variables:*** Transforming categorical data, such as ingredient names and cuisine types, into numerical representations using techniques like one-hot encoding and label encoding [32]. This transformation was essential for algorithms that require numerical input.
- ***Dataset Splitting:*** Dividing the data into training and testing sets to enable the evaluation of model performance on unseen data. A typical split ratio was maintained to ensure that the model had sufficient data to learn from while also being rigorously evaluated for generalization capabilities.

Additionally, Feature Engineering played a key role in enhancing the dataset's informational value and improving model performance. This involved creating new features and extracting deeper insights from existing data:

- ***Sentiment Polarity of User Reviews:*** Using natural language processing techniques to analyze and quantify the sentiment expressed in user reviews. This provided valuable context regarding user satisfaction and preferences, allowing the recommendation system to factor in qualitative feedback effectively.
- ***Preprocessing:*** Preprocessing techniques included tokenization, lemmatization, and stop-word removal for textual data, as well as normalization and scaling for numerical features.
- ***Feature Engineering:*** Feature engineering played a significant role in enhancing the model's performance. We derived new features, such as user engagement scores and ingredient sentiment scores, which were integrated into the recommendation engine. Feature transformation techniques, such as Principal Component Analysis (PCA), were also employed to reduce dimensionality and improve model efficiency [33].

This comprehensive approach to data collection and preparation ensured that the machine learning models built upon this foundation were well-equipped to learn meaningful patterns and provide accurate, personalized recipe recommendations. Meticulous attention to data quality and preparation was instrumental in the subsequent phases of model development and deployment, ultimately contributing to the project's overall success.

3.4. Model Selection and Building

The selection of machine learning models was based on the specific needs of the recommendation system. We employed a combination of Collaborative Filtering using Singular Value Decomposition (SVD), Content-Based Filtering using cosine similarity, and Sentiment Analysis using Recurrent Neural Networks (RNNs).

3.4.1. Collaborative Filtering:

Collaborative Filtering (CF) is designed to predict user preferences by learning latent factors that describe users and items. SVD was selected for its

capability to decompose the user-item interaction matrix into latent factors, capturing the underlying preferences of users and the characteristics of items. Collaborative Filtering model, which relied on user-item interaction data to predict preferences [34].

Algorithm: The user-item interaction matrix R, where r_{ui} represents the interaction (rating) between user 'u' and item 'I', is decomposed as follows:

$$R \approx U \sum V^T$$

where:

- U (dimension m × k) is the matrix of user features,
- Σ (dimension k × k) is the diagonal matrix of singular values,
- V^T (dimension k × n) is the matrix of item features.

The predicted interaction $\hat{r}_{ui}$ for user 'u' and item 'i' is computed as:

$$\hat{r}_{ui} = U_u . \sum . V_i^T$$

This allows the system to recommend items that match the user's preferences.x

3.4.2. Content-Based Filtering: Content-based filtering recommends items similar to those the user has interacted with, based on item attributes like ingredients and cooking techniques. Cosine Similarity was chosen to measure the similarity between items based on their attributes. It recommends analyzing item characteristics and user profiles to recommend similar recipes [35].

Algorithm: Cosine Similarity between two items 'X' and 'Y', represented as vectors X and Y, is calculated using the following formula:

$$\text{Cosine Similarity (X, Y)} = \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2}\sqrt{\sum_{i=1}^{n} y_i^2}}$$

Cosine Similarity (X, Y) = $\frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2}\sqrt{\sum_{i=1}^{n} y_i^2}}$

Where:

- $\sum_{i=1}^{n} x_i y_i$ is the dot product of the vectors X and Y.
- $\sqrt{\sum_{i=1}^{n} x_i^2}$ and $\sqrt{\sum_{i=1}^{n} y_i^2}$ are the magnitudes (lengths) of the vectors.

This formula calculates the cosine of the angle between the two vectors x and y, giving a value between -1 and 1. A higher cosine similarity indicates greater similarity between the items.

For a user 'u', the system calculates the similarity score between items the user has previously interacted with and other available items. The items with the highest similarity scores are recommended to the user [36].

3.4.3. Sentiment Analysis: To incorporate user sentiment into the recommendation system, RNNs were employed to analyze review text and extract sentiment scores. RNNs are particularly effective for processing sequential data, making them suitable for this task [37].

Recurrent Neural Networks, known for their effectiveness in processing sequential data, were employed to analyze user reviews and extract sentiment scores. These scores were then integrated into the recommendation model, enhancing its ability to prioritize positively reviewed recipes.

Algorithm: For a sequence of words $x_1, x_2, \ldots\ldots, x_T$ the RNN updates its hidden layers (state) h_t at each time step t using the following equation:

$$h_t = \tan h(W_{xh}x_t + W_{hh}h_t + b_h$$

Where:

$$y_t = W_{hy}h_t + b_y$$

- W_{xh} is the weight matrix for the input,
- W_{hh} is the weight matrix for the hidden layer,
- b_h is the bias vector,
- $\tan h$ is the activation function.

The output sentiment score at each time step is given by:

Where W_{hy} and b_y are the output weight matrix and bias.

The sentiment scores generated by the RNNs were integrated into the recommendation system to prioritize positively reviewed recipes.

3.4.4. Hybrid Recommendation Model: The individual models (SVD for Collaborative Filtering, Cosine Similarity for Content-Based Filtering, and RNNs for Sentiment Analysis) were combined into a hybrid recommendation system. This integration used a weighted blending approach to balance the strengths of each model:

$$\text{Final Score(i)} = \alpha.\hat{r}_{ui} + \beta.ScoreCB(i) + \gamma.Sentiment\ score(\text{i})$$

Where $\alpha, \beta, and\ \gamma$ are weights assigned to each component.

Building the hybrid recommendation model involved integrating these individual models into a cohesive system. The hybrid model combined the strengths of each approach, using a weighted blending technique to deliver more accurate and personalized recommendations.

Model Training and Evaluation: Each model was rigorously trained and validated using cross-validation techniques to ensure their generalizability. Hyperparameter tuning was conducted to optimize model performance, and the results were evaluated using metrics such as Root Mean Square Error (RMSE) for predictive accuracy and precision/recall for classification tasks. The combination of these models into a hybrid system allowed for a more comprehensive and accurate recommendation engine.

Chapter 4: Analysis & Design

4.1. Data Exploration and Insights:

To start our project, we turn our attention to the "Recipe Reviews and User Feedback Dataset", which serves as the foundation for our research. Sourced from the UCI Machine Learning Repository, this dataset provides a rich foundation for sentiment analysis, user behavior analysis, and the development of recipe recommendation systems. It captures various facets of user interactions with recipes, offering invaluable insights into user preferences and behaviors.

This dataset serves as the cornerstone of this project, offering a wealth of information crucial for analyzing user interactions and sentiments within the culinary domain. In this section, we will explore the dataset in detail, focusing on understanding its structure, the nature of its variables, and any underlying patterns or trends.

a. Dataset Overview:

The dataset consists of 18,182 entries and 15 columns, each capturing different aspects of recipe reviews and user interactions. The columns have been renamed to provide clarity and consistency. Here is a summary of the updated column names along with their descriptions:

Variables	Description
recipe_id	An index column with unique integers for each entry.
recipe_number	An identifier for each recipe.
recipe_code	A unique code assigned to each recipe to ensure precise identification.
recipe_name	The name of the recipe.
review_id	A unique identifier for each review comment.
user_id	An identifier for the user who wrote the review.
username	The name of the user who wrote the review.
user_reputation	A score that reflects the reputation of the user, which may influence the credibility of their review.
created_at	A timestamp showing when the review was created.
reply_count	The number of replies to the review, providing insights into user engagement.
thumbs_up	The count of positive votes received for the review.
thumbs_down	The count of negative votes received for the review.
star_rating	The star rating given by the user, ranging from 1 to 5 stars, with a score of 0 indicating no rating.
best_score	A metric related to the highest score received or an overall evaluation of the review.
review_text	The content of the review comment.

Figure 4: Dataset Description

Key Features of the Dataset:

- *Recipe Details:* Each recipe is identified by its name and ranking on the top 100 recipes list, along with a unique recipe code. These features ensure precise identification and tracking of recipes across the dataset.
- *User Information:* The dataset includes extensive user data, such as user ID, username, and an internal reputation score. This information is instrumental in understanding user engagement and the credibility of reviews.
- *Review Comments:* Each review is uniquely tagged with a comment ID and accompanied by detailed metadata, including the creation timestamp, number of replies, and counts of up-votes and down-votes. This metadata is crucial for temporal analysis and understanding the response to each review.
- *Sentiment and Ratings:* Users express their sentiments through a star rating system ranging from 1 to 5 stars, with a score of 0 showing no rating. This rating system is important to apply for sentiment analysis and significantly influences the recommendation models developed in this project.

By exploring these features, we aim to uncover patterns and trends that will inform the development of the recommendation system. Analyzing the distribution of star ratings can reveal

overall user satisfaction while examining review metadata can provide insights into engagement levels and review quality.

b. Descriptive Statistics:

To gain an initial understanding of the dataset, we first computed descriptive statistics for key variables. This includes measures of central tendency such as the mean, median, and mode, as well as measures of dispersion like the standard deviation.

The dataset consists of 18,182 entries across 15 attributes, each providing different insights into recipe reviews and user interactions. The recipe_id column, serving as an index, contains 725 unique values, which indicates the number of distinct recipes reviewed in the dataset. The recipe_number and recipe_code columns further categorize these recipes, with 100 unique identifiers and codes, respectively. These codes span a wide range, from 386 to 191,775, and the standard deviation suggests significant variability in how these codes are assigned.

The recipe_name column, which is an object data type, also contains 100 unique values, consistent with the number of distinct recipes. The review_id, user_id, and username columns each contain a high number of unique values (18,182, 13,812, and 13,586, respectively), reflecting the diverse set of reviews and users in the dataset. This diversity is further emphasized by the user_reputation attribute, which has a mean of 2.16 but a large standard deviation of 10.01, indicating a few users with very high reputation scores amidst a majority with lower scores.

The created_at column records the timestamp of each review, showing a broad time range with a mean timestamp of approximately 1.62 billion seconds, equating to the dataset capturing reviews over several years. The variability in review interaction is evident in the reply_count, thumbs_up, and thumbs_down attributes, where the low mean values (0.01, 1.09, and 0.55, respectively) paired with their respective standard deviations indicate that while most reviews receive little engagement, a few garners significant attention.

Attribute	Description	Data Type	Unique Values	Total Entries	Min	Max	Mean	Std. Deviation	Missing Values
recipe_id	Index	int64	725	18,182	0	724	121.47	116.75	0
recipe_number	Recipe Number	int64	100	18,182	1	100	38.69	29.79	0
recipe_code	Recipe Code	int64	100	18,182	386	191,775	21,773.67	23,965.11	0
recipe_name	Name of the recipe	object	100	18,182	NaN	NaN	NaN	NaN	0
review_id	Comment ID	object	18,182	18,182	NaN	NaN	NaN	NaN	0
user_id	User ID	object	13,812	18,182	NaN	NaN	NaN	NaN	0
username	User Name	object	13,586	18,182	NaN	NaN	NaN	NaN	0
user_reputation	User Reputation	int64	22	18,182	0	520	2.16	10.01	0
created_at	Timestamp of Comment Creation	int64	2,695	18,182	1,613,035,336	1,665,756,035	1,623,710,486.55	5,468,696.71	0
reply_count	No. of Replies	int64	4	18,182	0	3	0.01	0.14	0
thumbs_up	No. of Thumbs Up	int64	62	18,182	0	106	1.09	4.20	0
thumbs_down	No. of Thumbs Down	int64	58	18,182	0	126	0.55	3.47	0
star_rating	Star Rating	int64	6	18,182	0	5	4.29	1.54	0
best_score	Best Score	int64	588	18,182	0	946	153.16	141.08	0
review_text	comment Text	object	17,731	18,180	NaN	NaN	NaN	NaN	2

Figure 5: Descriptive Statistics of Data

The star_rating attribute reveals that the average rating given by users is relatively high, at 4.29 stars, but with a standard deviation of 1.54, indicating varied opinions. The best_score attribute, which likely measures the perceived quality or popularity of the review, has a mean of 153.16, with a wide range of values, highlighting diverse levels of review impact. Lastly, the review_text attribute, with 17,731 unique entries, underscores the richness of qualitative feedback, though it is important to note that two entries are missing.

Mode for Nominal Attributes:

The mode represents the most frequently occurring value in a dataset. For nominal (categorical) attributes in this dataset, the mode provides insights into the most common values [38]. Below are the modes for each nominal attribute:

Attribute	Mode
recipe_name	Cheeseburger Soup
review_id	sp_aUSaElGf_100276_c_106707
user_id	u_1oKVZzipo1u8lcqQzDUcw4UBn9e
username	2124arizona
review_text	Delicious!

Figure 6: Mode for Categorical variables

The mode analysis of the nominal attributes in our dataset reveals interesting insights into user behavior and popular content. The most commonly reviewed recipe is "Cheeseburger Soup", indicating its popularity or frequent mention among users. The review with the ID sp_aUSaElGf_100276_c_106707 is the most frequently appearing, which could suggest either duplication or its high popularity. The user with ID u_1oKVZzipo1u8lcqQzDUcw4UBn9e stands out as the most active contributor, having the highest number of reviews in the dataset. Similarly, the username 2124arizona is the most frequently occurring, possibly indicating a prolific reviewer. Notably, the phrase "Delicious!" is the most common review comment, underscoring a generally positive sentiment associated with the recipes.

c. **Identifying Patterns and Trends:** During the exploration phase, several significant patterns and trends emerged:
   - *Rating Distribution:* A higher frequency of 4 and 5-star ratings was observed, which suggests overall satisfaction with the recipes. This pattern indicates that most users found the recipes to be of high quality.

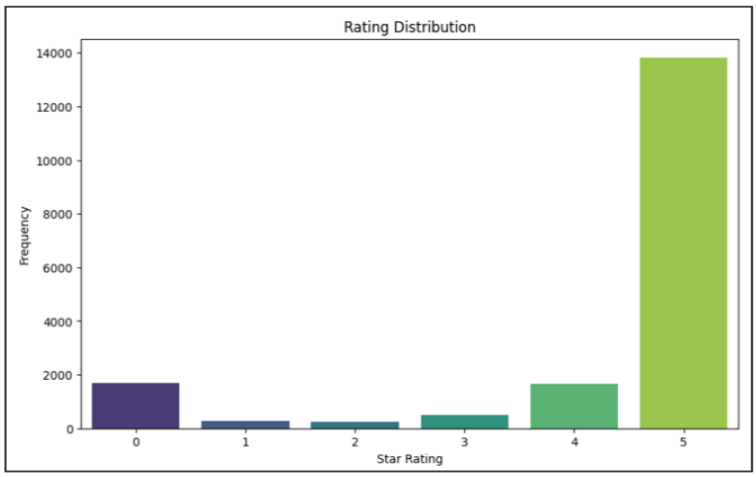


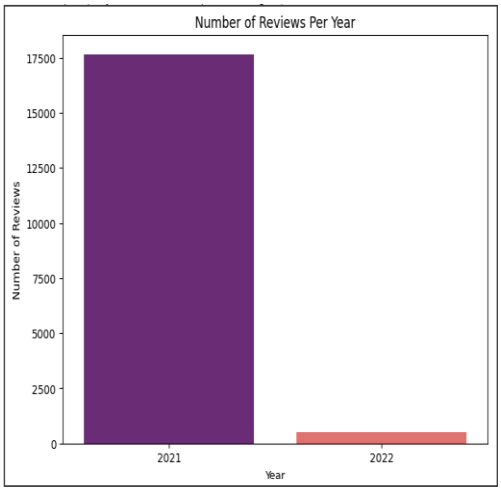
Figure 7: Star Rating Distribution

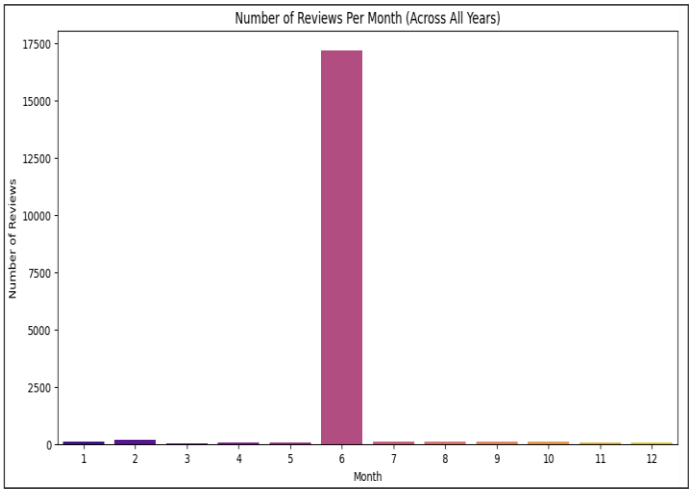
   - *User Engagement:* Reviews with higher reply counts and thumbs-up votes were often correlated with higher star ratings, indicating a positive relationship between user engagement and feedback. This suggests that more engaged users tend to leave more favorable reviews.

- *Temporal Patterns:* Analysis of review timestamps revealed peak periods of activity, potentially corresponding to specific events or seasonal trends. To accurately capture and analyze these patterns, the "created_at" column was converted into a Date Time object. This conversion allowed for a more precise exploration of temporal patterns, such as identifying particular months or seasons when user engagement with recipes spikes. Understanding these temporal trends can be instrumental in refining the recommendation system to account for seasonal recipe preferences, ensuring that recommendations are more relevant to users during different times of the year.

```
#Convert created_at to datetime objects
data['created_at'] = pd.to_datetime(data['created_at'], unit='s')
data
```

	recipe_id	recipe_number	recipe_code	recipe_name	review_id	user_id	username	user_reputation	created_at
0	0	1	14299	Creamy White Chili	sp_aUSaElGf_14299_c_2G3aneMRgRMZwXqlHmSdXSG1hEM	u_9lFLlhMa8QaG	Jeri326	1	2022-10-13 00:11:29
1	1	1	14299	Creamy White Chili	sp_aUSaElGf_14299_c_2FsPC83HtzCsQAtOxlbL6RcaPbY	u_Lu6p25tmE77j	Mark467	50	2022-10-09 01:08:07
2	2	1	14299	Creamy White Chili	sp_aUSaElGf_14299_c_2FPrSGyTv7PQkZq37j92r9mYGkP	u_s0LwgpZ8Jsqq	Barbara566	10	2022-09-28 22:35:57
3	3	1	14299	Creamy White Chili	sp_aUSaElGf_14299_c_2DzdSlgV9qNiuBaLoZ7JQaartoC	u_fqrybAdYjgjG	jeansch123	1	2022-08-29 15:43:28
4	4	1	14299	Creamy White Chili	sp_aUSaElGf_14299_c_2DtZJuRQYeTFwXBoZRfRhBPEXjI	u_XXWKwVhKZD69	camper77	10	2022-10-04 20:03:43
...	...	...	...	...	...	...	...	...	...
18177	114	100	82745	Mamaw Emily's Strawberry Cake	sp_aUSaElGf_82745_c_331352	u_1oKbIS4ULpTAACUTILo0QIO5mrN	WhittierCA	1	2021-06-03 10:59:37
18178	115	100	82745	Mamaw Emily's Strawberry Cake	sp_aUSaElGf_82745_c_204836	u_1oKY7CKLaFQpI3cBCVaxHXAEGbt	susieswan	1	2021-02-11 09:45:20





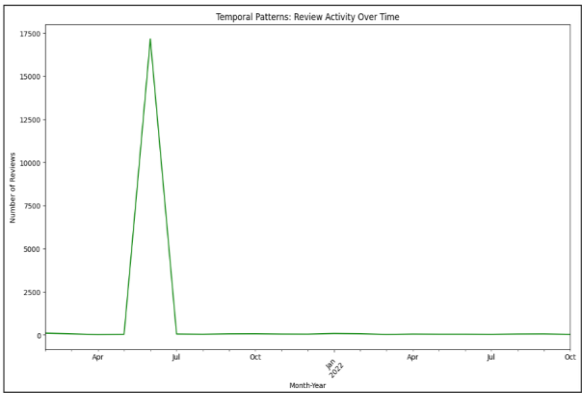


Figure 8: User Engagement reviews over time

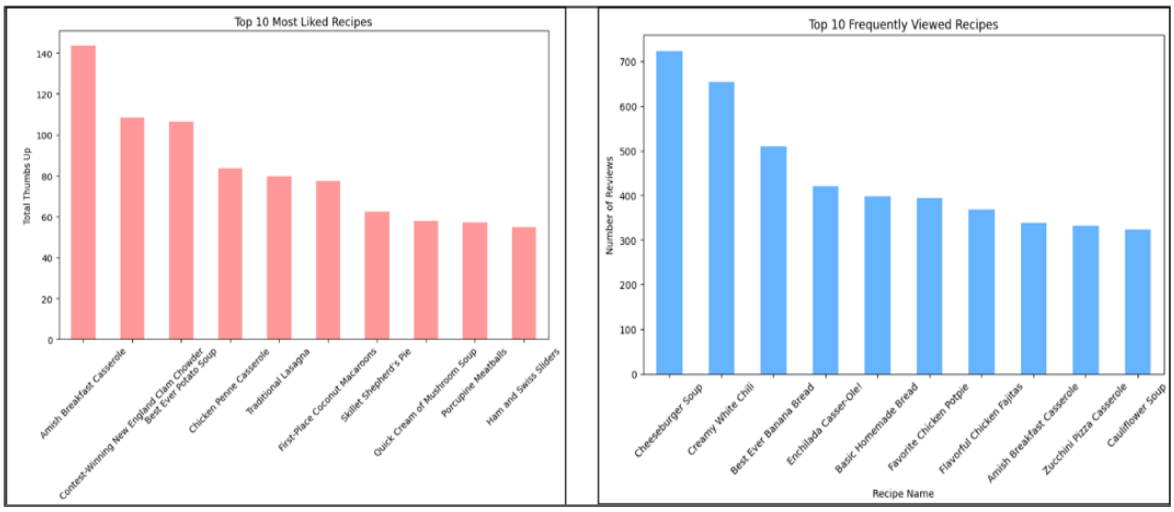


Figure 9: Most Liked and Viewed Recipes

d. <u>Data Visualization:</u>

To better understand the data and uncover deeper insights, various visualizations were employed:

- *<u>Heatmaps:</u>* Heatmaps were utilized to visualize correlations between numerical features, such as user reputation and thumbs-up counts. The heatmaps highlighted strong correlations, such as between user reputation and review quality, which are important for identifying influential factors in user feedback.
- *<u>Correlation Matrices:</u>* Correlation matrices were constructed to display the relationships between different variables, such as the correlation between the number of replies and star ratings. These matrices are crucial for identifying potential predictive features that could enhance the recommendation model.
- *<u>Histograms, Box Plots, and Scatter Plots</u>*: Histograms were used to illustrate the distribution of star ratings, revealing how users generally rated the recipes. Box plots highlighted outliers in user reputation scores, offering insights into the range and distribution of user reputation. Scatter plots were employed to examine the relationship between different engagement metrics and star ratings, providing a clearer picture of how various factors interact.

These visual tools were instrumental in elucidating the relationships between various attributes and identifying potential features for model development. By thoroughly exploring and visualizing the data, we uncovered patterns and trends that are important for the subsequent stages of building a sophisticated and effective recommendation system.

4.2. <u>Data Preprocessing and Feature Engineering:</u>

In this section, we outline the essential steps taken to prepare the raw data for effective machine-learning modeling. Data preprocessing and feature engineering are critical processes that ensure the dataset is clean, consistent, and well-structured, thereby optimizing the performance of the models [39]. The procedures described here include data cleaning, normalization, encoding of

categorical variables, and the careful engineering of features, all of which are fundamental to building robust and accurate predictive models.

In this phase, we focused on preparing the data for analysis and modeling by addressing inconsistencies, transforming variables, and generating new features that could enhance the performance of our recipe recommendation system.

a. <u>Data Cleaning</u>: Data cleaning is an important step in the preprocessing phase, addressing issues such as missing values, duplicates, and outliers that could otherwise compromise model accuracy. The key aspects of our data cleaning process include:

1. Handling Missing Values: In the data cleaning process, handling missing values was crucial to ensure the integrity and usability of the dataset. We employed two primary strategies:

- Dropping columns with a high percentage of missing values
- Imputing missing values based on the type of data.

- ***Dropping Missing Values****:* Deciding when to drop columns with missing data can vary based on the context and the dataset's specific needs. While there is no universally agreed-upon threshold, common guidelines include:
  - ***5-10%:*** In conservative approaches, columns with more than 5-10% missing data may be dropped, particularly if the feature is not essential or if there are sufficient other features available to compensate for the loss.
  - ***25%:*** For columns where missing data ranges between 20-30%, dropping may be considered if the feature's importance is moderate, and the data loss does not significantly impact the analysis.
  - ***50%:*** If a column has 50% or more missing data, it is often recommended to remove it unless the feature is critical and more complex imputation methods are warranted.
  - ***80-85%:*** Columns with missing values exceeding this range are typically dropped due to substantial data loss and the unreliability of any imputation attempts.
- ***Imputation:*** For columns that were retained despite having missing data, we employed different imputation strategies based on the data type:
  - ***Numerical Features****:* Missing numerical values were imputed using either the mean or median of the available data. The mean was used for normally distributed data, while the median was preferred for skewed distributions to prevent distortion by outliers.
  - ***Categorical Features****:* Missing categorical values were filled with the most frequent category (mode) or a placeholder such as “Unknown”. This approach ensured that no significant information was lost, and the categorical variables remained usable in the subsequent modeling phase.

In our specific case, only two missing values were present in the dataset. These were dropped, as their removal had no significant impact on the overall analysis, this was a pragmatic and efficient solution.

```
# Checking for missing values in 'text' column
missing_review_text_count = data['review_text'].isnull().sum()
print(f"Number of missing values in 'review_text' column: {missing_review_text_count}")

# Find, display rows with missing 'text' values
if missing_review_text_count > 0:
    missing_review_text_rows = data[data['review_text'].isnull()]
    print("Rows with missing 'review_text' values:")
    print(missing_review_text_rows)

    missing_review_text_indexes = missing_review_text_rows.index
    print("Indexes of rows with missing 'review_text' values:")
    print(missing_review_text_indexes)
```

```
Number of missing values in 'review_text' column: 2
Rows with missing 'review_text' values:
      recipe_id  recipe_number  recipe_code              recipe_name  \
1507        344              3         2832        Cheeseburger Soup
2722        299              6        21444  Favorite Chicken Potpie

                      review_id                         user_id  \
1507   sp_aUSaElGf_2832_c_260955  u_1oKZmfyycO3xVnQeffIKHuaKT2U
2722  sp_aUSaElGf_21444_c_260817  u_1oKXz42APTrXTFF6SGvR3YMBJGy

             username  user_reputation  created_at  reply_count  thumbs_up  \
1507  No.1FamilyBaker                1  1622717645            0          0
2722         TRGriggs                1  1622717652            0          0

      thumbs_down  star_rating  best_score review_text
1507            0            5         100         NaN
2722            0            5         100         NaN
Indexes of rows with missing 'review_text' values:
Index([1507, 2722], dtype='int64')
```

Figure 10: Checking Missing values

```
# Drop rows with missing 'text' values
data = data.dropna(subset=['review_text'])

# Verify that rows with missing 'text' values have been removed
missing_review_text_count_after = data['review_text'].isnull().sum()
print(f"Number of missing values in 'review_text' column after removal: {missing_review_text_count_after}")
data
```

```
Number of missing values in 'review_text' column after removal: 0
```

	recipe_id	recipe_number	recipe_code	recipe_name	review_id	user_id	username	user
0	0	1	14299	Creamy White Chili	sp_aUSaElGf_14299_c_2G3aneMRgRMZwXqlHmSdXSG1hEM	u_9iFLIhMa8QaG	Jeri326	
1	1	1	14299	Creamy White Chili	sp_aUSaElGf_14299_c_2FsPC83HtzCsQAtOxlbL6RcaPbY	u_Lu6p25tmE77j	Mark467	
2	2	1	14299	Creamy White Chili	sp_aUSaElGf_14299_c_2FPrSGyTv7PQkZq37j92r9mYGkP	u_s0LwgpZ8Jsqq	Barbara566	
3	3	1	14299	Creamy White Chili	sp_aUSaElGf_14299_c_2DzdSlgV9qNiuBaLoZ7JQaartoC	u_fqrybAdYjgjG	jeansch123	
4	4	1	14299	Creamy White Chili	sp_aUSaElGf_14299_c_2DtZJuRQYeTFwXBoZRfRhBPEXjI	u_XXWKwVhKZD69	camper77	

Figure 11: Removing Outliers

2. Removing Duplicates: To avoid bias in the analysis and model training process, duplicate entries were identified and removed. This step was essential to ensure that the model was trained on unique and valid instances, thereby improving the reliability and accuracy of the predictive outcomes.

3. Outlier Detection and Treatment: If we have any Outliers in numerical features, we can detect them using statistical methods such as the Z-score and Interquartile Range (IQR). Depending on the context, outliers were either capped to a maximum/minimum threshold to reduce their impact

or removed if deemed erroneous. Treating outliers in this way helped to mitigate the risk of skewed results, ensuring that the model was not unduly influenced by extreme values.

b. Data Transformation: Data transformation involves converting raw data into a format suitable for analysis [40]**.** To prepare the data for analysis, we made a few key changes to ensure it was in the right format for our machine-learning models.

1. Date and Time Conversion: First, we transformed the "created_at" column from just a timestamp into a Date Time object. This allowed us to pull out useful details like the year, month, day, and hour of each review, which helped us analyze patterns over time.

2. Encoding Categorical Variables: We worked on encoding categorical variables like "recipe_name" and "user_name". Since our models need numbers to work with, we converted these categories into numerical values. One-hot encoding was used for nominal categories, while label encoding was applied to ordinal categories where the order matters.

For categories without any order (like recipe names), we used one-hot encoding, turning each unique category into its own column. If categories have an order, we applied label encoding to assign numerical values based on their order.

For categorical variables with a natural order, such as sentiment labels, we used label encoding. This method assigns a unique integer to each category based on its order. For instance, sentiment labels like "Positive", "Neutral", and "Negative" might be encoded as 2, 1, and 0, respectively.

By applying these encoding techniques, we ensured that all categorical variables were in a numerical format suitable for our models.

- **Normalization and Standardization:** Lastly, we applied standardization to our numerical features. This step was crucial to ensure that all features were on a similar scale, making sure no single feature dominated the model. Standardization adjusted the data so that it had a mean of 0 and a standard deviation of 1, which is particularly important for models sensitive to the scale of the data.
- **Text Normalization**: The review texts were preprocessed to standardize the text data. This involved converting all text to lowercase, removing punctuation, and eliminating stopwords. These steps were necessary to prepare the text for subsequent sentiment analysis and feature extraction.

```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
data[['user_reputation', 'reply_count', 'thumbs_up', 'thumbs_down','star_rating', 'best_score']] = scaler.fit_transform(data[['user_reputation', 'reply_count', 'thumbs_up', 'thumbs_down','star_rating', 'best_score']])
data
```

review_id	user_id	username	user_reputation	created_at	reply_count	thumbs_up	thumbs_down	star_rating	best_score	review_text	year	month	month_year
_aUSaElGf_14299_c_2G3aneMRgRMZwXqIHmSdXSG1hEM	u_9iFLlhMa8QaG	Jeri326	-0.115801	2022-10-13 00:11:29	-0.106042	-0.259309	-0.158317	0.460429	2.649823	I tweaked it a little, removed onions because ...	2022	10	2022-10
sp_aUSaElGf_14299_c_2FsPC83HtzCsQAtOxIbL6RcaPbY	u_Lu6p25tmE77j	Mark467	4.776892	2022-10-09 01:08:07	-0.106042	1.406920	-0.158317	0.460429	4.046213	Bush used to have a white chili bean and it ma...	2022	10	2022-10
sp_aUSaElGf_14299_c_2FPrSGyTv7PQkZq37j92r9mYGkP	u_s0LwgpZ8Jsqq	Barbara566	0.782857	2022-09-28 22:35:57	-0.106042	0.454789	-0.158317	0.460429	3.946977	I have a very complicated white chicken chili ...	2022	9	2022-09
sp_aUSaElGf_14299_c_2DzdSIgV9qNiuBaLoZ7JQaartoC	u_fqrybAdYjgjG	jeansch123	-0.115801	2022-08-29 15:43:28	14.389054	0.216756	-0.158317	-2.776213	3.032590	In your introduction, you mentioned cream chee...	2022	8	2022-08
sp_aUSaElGf_14299_c_2DtZJuRQYeTFwXBoZRfRhBPEXjI	u_XXWKwVhKZD69	camper77	0.782857	2022-10-04 20:03:43	7.141506	1.406920	-0.158317	-2.776213	4.726687	Wonderfull I made this for a "Chili/Stew&#...	2022	10	2022-10
...	...	...	...	...	...	...	...	...	...	...	...	...	...
sp_aUSaElGf_82745_c_331352	u_1oKbIS4ULpTAACUTILo0QlO5mrN	WhittierCA	-0.115801	2021-06-03 10:59:37	-0.106042	-0.259309	-0.158317	0.460429	-0.376869	This Strawberry Cake has been a family favorit...	2021	6	2021-06
sp_aUSaElGf_82745_c_204836	u_1oKY7CKLaFQpI3cBCVaxHXAEGbt	susieswan	-0.115801	2021-02-11 09:45:20	-0.106042	-0.259309	-0.158317	0.460429	-0.376869	<p>I received endless compliments on this cake...	2021	2	2021-02

Figure 12: Encoding Nominal Variables

Normalization vs. Standardization:

When preparing data for machine learning models, it's important to scale features so that they contribute equally to the model [41]. This can be done through two common techniques: **Normalization** and **Standardization**.

- ***Normalization****:* This technique rescales the values of a feature to a specific range, typically between 0 and 1. This method is useful when you want to preserve the relative distances between data points while ensuring all features are on the same scale. Normalization is particularly important for algorithms like K-Nearest Neighbors and neural networks, where the magnitude of the data can significantly impact the model's performance.
- ***Standardization****:* Unlike normalization, standardization scales the features so that they have a mean of 0 and a standard deviation of 1. This method is useful when you want to center the data and ensure that all features have the same unit variance. Standardization is particularly important for algorithms like Support Vector Machines and Principal Component Analysis (PCA), which assume that the data is normally distributed.

In this project, standardization was chosen because it centers the data, making it more suitable for algorithms that rely on normally distributed data. The StandardScaler from sklearn. preprocessing was applied to the features user_reputation, reply_count, thumbs_up, thumbs_down, star_rating, and best_score, ensuring that they all had a mean of 0 and a standard deviation of 1.

To ensure that all features contributed equally to the model, numerical features were normalized or standardized. This was particularly important for algorithms sensitive to the scale of the data, such as SVM or K-nearest neighbors.

By applying standardization, we ensured that each feature contributed equally to the model, without being dominated by features with larger magnitudes. These transformations ensured our data was well-prepared and ready to be used effectively in our machine-learning models.

C. Feature Engineering:

Feature engineering is the process of transforming raw data into a set of features that better represent the underlying problem to predictive models, thereby improving their performance. This section discusses the techniques and strategies applied to create new features from existing data, aiming to boost the accuracy and robustness of our recipe recommendation system.

The goal of feature engineering is to extract more meaningful information from the raw dataset, enabling machine learning models to capture patterns more effectively. This process involves creating new variables, transforming existing ones, and combining features to reflect complex relationships within the data. We focused primarily on text-based feature engineering techniques for our recipe recommendation system, while also exploring potential techniques to further enhance model performance.

1. Text Features:

In our project, text features were of paramount importance as user reviews provided valuable insights into user preferences and sentiments. The following techniques were employed to engineer features from text data:

- **Sentiment Scores:** Using the TextBlob Python library for processing textual data, sentiment polarity scores were extracted from the processed_review_text column. Reviews were classified into 'Positive', 'Neutral', or 'Negative' and these categories were added as a new feature based on scores. This allowed us to quantify user sentiment and tailor recommendations, accordingly, providing insight into the emotional tone of the reviews.

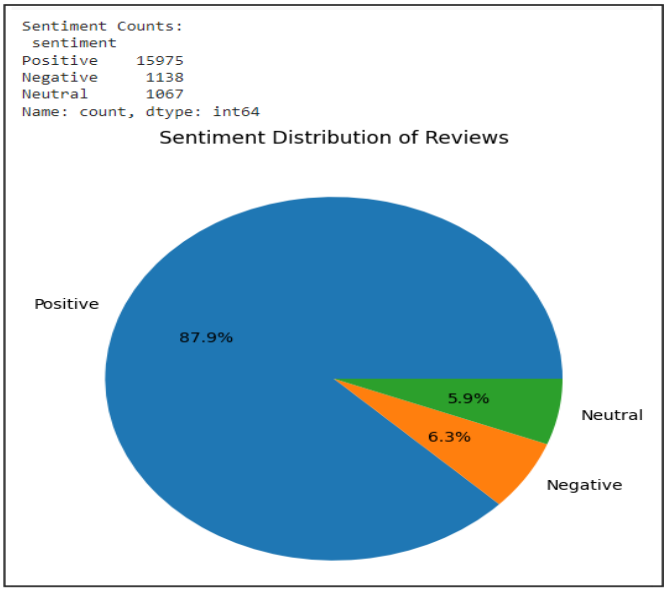


Figure 13: Sentiment Score Distribution

We performed sentiment analysis on the user reviews to extract sentiment scores, which were then categorized as positive, neutral, or negative. These scores provided a quantitative measure of user satisfaction and were crucial in tailoring the recommendations to match user preferences. Sentiment analysis helped us understand the emotional tone of the reviews, influencing how recipes were recommended based on the expressed sentiment. These scores were crucial in understanding user preferences and improving recommendation accuracy.

- **Word Counts and Keyword Presence:** To quantify the content of each review, we calculated the total word count. This feature helped gauge the verbosity of the review, which could indicate the depth of user engagement. Additionally, we tracked the presence of specific keywords related to user satisfaction (e.g., "spicy" or "easy to make" could provide context about the user's experience and preferences, which was useful in refining recommendations. These features helped in identifying common themes and preferences among users.
- **TF-IDF Representation:** We applied Term Frequency-Inverse Document Frequency (TF-IDF) to the review text. This technique converts text into numerical values, reflecting how important a word is to a particular document relative to the entire corpus. TF-IDF was particularly useful in understanding the relevance of certain words or phrases in predicting user satisfaction and played a key role in distinguishing between common and unique terms within the reviews.

2. Potential Feature Engineering Techniques:

To enhance the predictive power of our models, we engineered several new features that provide additional insights into user behavior, recipe characteristics, and temporal trends.

- **User Engagement Features:**
  - **Interaction Metrics:** Features such as the number of times a user interacted with a recipe (likes, shares, comments) could be engineered to understand user engagement levels. To measure user interaction levels, we can normalize the number of replies per review by the total number of reviews. This feature helps identify which reviews generated more discussions and engagement, suggesting their influence on user decision-making.
  - **Session-Based Features:** Analyzing user behavior over a session (e.g., time spent on a recipe page, sequence of viewed recipes) could offer insights into user intent and preferences. Thumbs-Up to Thumbs-Down Ratio, this feature can quantify the overall sentiment of user reviews. By dividing the number of thumbs-up by thumbs-down, we developed a ratio that indicates how positively or negatively a recipe was received by the community.
- **Temporal Features:**
  - **Review Timestamp Analysis:** Extracting features from the timestamp of reviews, such as the day of the week, month, or hour, could help in identifying patterns related to when users are most likely to leave positive or negative feedback. We can extract the day of the week from the timestamp of each review to capture patterns in user activity.

For instance, some users might be more active on weekends, which could influence their recipe choices.

- **Time Since Last Interaction:** Calculating the time elapsed since the user's last interaction with the platform could reveal trends in user activity and engagement.A binary feature indicating whether the review was posted on a weekend or a weekday. This feature helps in understanding how user behavior varies across different days.

- **Demographic and Geographic Features:**
  - **User Demographics:** If user demographics are available in the dataset, features such as user age, gender, or location could be engineered to personalize recommendations further based on demographic trends.
  - **Geographic Preferences:** Identifying regional preferences by analyzing the geographic data tied to users could uncover location-based trends in recipe popularity.
- **Advanced Text Features:** In addition to basic text features, advanced techniques such as N-grams and Topic Modeling will be used to capture deeper contextual information from user reviews. N-grams allow the model to recognize common phrases that carry specific sentiments or preferences, such as 'easy to make' or 'too spicy,' which might be overlooked by single-word analysis. While Topic Modeling, through methods of Latent Dirichlet Allocation (LDA), can uncover underlying themes in user reviews, categorizing them into topics such as 'healthy recipes' or 'comfort food.' These topics can then be used to fine-tune recommendations by aligning them more closely with user interests**.**

Feature engineering is an iterative process that often requires multiple rounds of experimentation and validation. If a data scientist discovers that additional data is needed, they may request further preprocessing from data engineers [42]. Once the features are finalized, they are deployed into the machine learning pipeline for model training and prediction. While we focused on text features for this project, the techniques, and potential features discussed highlight the importance of thoughtful feature selection and transformation in improving model performance. By effectively engineering features, we can ensure that our models capture the most relevant aspects of the data, leading to more accurate recommendations.

D. Data Splitting: Data splitting is the process of dividing data into multiple subsets, which is an important part of machine learning. In machine learning, data splitting is often used to prepare for training and testing machine learning models [43].The goal is to avoid overfitting, when a model fits its training data too well but doesn't fit additional data reliably. Data splitting can also help build robust models that generalize well to unseen data.

To ensure that our machine learning models generalize well to new, unseen data, we strategically split the dataset into training, validation, and test sets. This separation helps in evaluating the model's performance at different stages of development and minimizing the risk of overfitting. In machine learning, data splitting is typically done into three or four sets:

- **Training Set:** The majority of the data was allocated to the training set, which is used to teach the model the underlying patterns and relationships within the dataset. This phase is essential for enabling the model to learn and make accurate predictions. The training set

lays the groundwork for the model's ability to understand and capture the complexities of the data.

- **Validation Set:** A portion of the data was reserved as a validation set to tune hyperparameters and select the best model. This set provided an unbiased evaluation of the model's performance during the training phase. The validation set serves as an intermediary step, providing an unbiased evaluation of the model's performance during the training phase. By tuning hyperparameters and selecting the best model based on validation results, we ensure that the model is optimized before the final testing phase.
- **Test Set:** The final test set, kept entirely separate from the training and validation data, is used to assess the model's ability to generalize to new, unseen data. This final evaluation ensures that the performance metrics truly reflect the model's capacity to make accurate predictions in real-world scenarios.
- **Dev set:** A set of examples used to change learning process parameters.

When splitting data, it's important to consider the representativeness of the data. The extracted data should have enough population for every class of data and be unbiased. Biased data can lead to an inaccurate model. One approach to splitting data is random sampling, which is straightforward and protects the process from bias. However, it can be problematic if the response isn't evenly distributed across the outcome. Data should be split so that data sets can have high training data. Normally data might be split at an 80-20 or a 70-30 ratio of training vs. testing data. The exact ratio depends on the data, but a 70-20-10 ratio for training, dev, and test splits is optimal for small data sets [44].

To ensure robust model evaluation and prevent overfitting, the dataset was divided into training, validation, and test sets. The split was done separately for both the classification and regression tasks for our Recommendation system, The train_test_split function was used with an 80-20 split, ensuring that the model had sufficient data to learn from while maintaining a separate set of data for validation and testing purposes.

By meticulously splitting the data, along with careful preprocessing and feature engineering, we laid a solid foundation for building a robust and effective recommendation system. The next phase involves selecting and fine-tuning models that can leverage these engineered features to deliver personalized recipe recommendations to users.

4.3. Model Architecture and Design:

Model Selection: The primary objective of this project is to develop a recommendation system that can accurately predict both the sentiment of user reviews and the corresponding star ratings for recipes. After considering several machine-learning models, including traditional algorithms like logistic regression, Multinomial Naïve Bayes, and decision trees, it was decided to implement a neural network-based approach. Specifically, a Recurrent Neural Network (RNN) combined with a dense network was chosen due to its ability to capture sequential patterns in textual data and effectively process numerical features.

Reasons for Model Selection:

- ***RNN with LSTM:*** RNNs, particularly with Long Short-Term Memory (LSTM) units, are well-suited for processing sequential data such as text. LSTMs can capture long-term dependencies, making them ideal for sentiment analysis.
- **Dense Network:** For numerical features, a dense network is employed to model interactions between features, enhancing the overall prediction accuracy for both sentiment classification and star rating regression.
- **Multi-Task Learning**: The chosen architecture supports multi-task learning, where both sentiment classification and star rating prediction are addressed within a single framework. This approach enables the model to share information between tasks, leading to improved generalization.

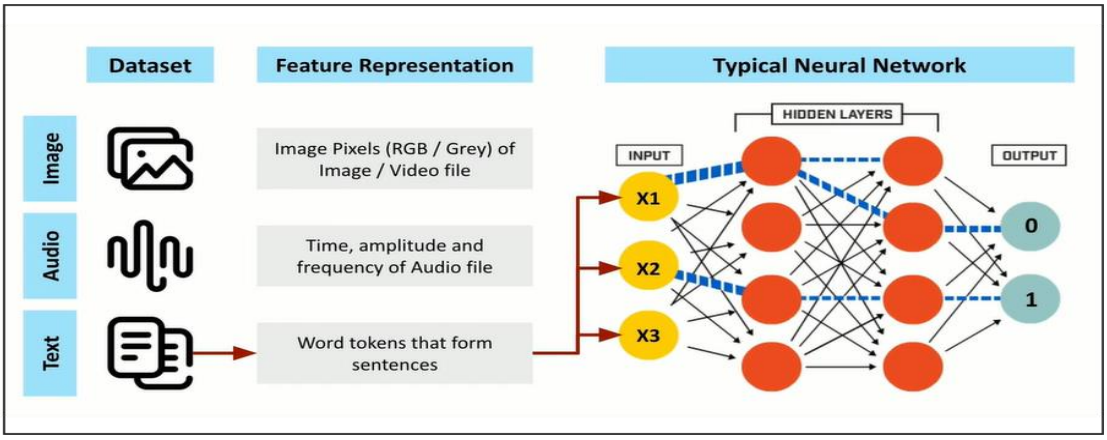


Figure 14: Standard Architecture for Neural Network

	Simple Neural Network	Convolutional Neural Network (CNN)	Long Short-Term Memory Network (LSTM)
Intuition	Input data fed forward to dense layer, only capable to learn simple patterns	Ability to extract spatial features from input data using self-learning filters	Capable of learning & remembering patterns across time
Structure (specific to this project)	Embedding layer >> Flatten >> Dense layer >> Output	Embedding layer >> Convolution + Pooling >> Dense layer >> Output	Embedding layer >> LSTM >> Dense layer >> Output
Speciality	Can handle incomplete knowledge, high fault tolerance	Accuracy in recognizing images	Memory and self-learning
Data Type	Fed on tabular and text data	Relies on image data	Trained with sequence data
Applications	Simple problem solving such as predictive analysis, prone to Overfitting	Image/Video Analytics, Speech Recognition and understanding natural language processing	Machine Translation, Language Modeling and Multilingual Language Processing

Figure 15: Difference Between NN models

Model Architectural Design:

The model architecture for the recipe recommendation system integrates both textual and numerical data to predict sentiment classifications and star ratings from user reviews. The model

is designed as a multi-task learning framework that combines a Recurrent Neural Network (RNN) for text processing with a fully connected neural network for numerical data processing [45].

The model is designed to handle two types of input data: textual data from user reviews and numerical data from user interactions and ratings. The key components and the flow of the architecture are described below:

1. Input Layers:

- ***Text Input Layer:*** The text data, specifically the processed review text, is first tokenized and converted into sequences of integers. These sequences are then padded to ensure uniform length across all samples. The model takes in this padded text sequence as input.
- ***Numerical Input Layer:*** Numerical features like user reputation, thumbs up, thumbs down, and best score are preprocessed and normalized. These features are fed into the model as a separate input.

2. **Text Processing**:

- ***Embedding Layer:*** Converts the input text(words) sequences into dense vector representations of fixed size (128 dimensions), which captures the semantic relationships between the meanings of the words.
- ***Bidirectional LSTM Layers:*** Two LSTM layers are stacked. The first LSTM layer (64 units) processes the word embeddings in both forward and backward directions, allowing the model to capture context from both sides of each word [46]. A second LSTM layer (32 units) further processes the output of the first layer, reducing the dimensionality and extracting higher-level features.

3. **Numerical Data Processing:**

- ***Normalization:*** The numerical features are standardized to have a mean of 0 and a standard deviation of 1, ensuring that the model treats each feature on the same scale.

4. **Feature Combination:**

- ***Concatenation Layer:*** The processed textual features from the LSTM layers are concatenated with the normalized numerical features. This combined representation encapsulates all available information, both textual and numerical, related to the user's review.
- ***Dense Layers:*** The concatenated features are passed through the fully connected dense layers (ReLU) activation function which applies non-linear transformations and learns complex interactions between textual and numerical data [47]. A dropout rate of 0.5 is applied to prevent overfitting. Dropout layers are used between dense layers to prevent overfitting by randomly setting a fraction of input units to zero during training.

5. **Output Layers:**

- ***Classification Output Layer:*** A SoftMax layer produces a probability distribution over three possible sentiment classes (positive, neutral, and negative). This layer is responsible for predicting the sentiment category of each review.

- ***Regression Output:*** A single linear output layer provides a prediction for the star rating, which is treated as a continuous value. This layer is trained to minimize the mean squared error value between the predicted and actual star ratings.

6. **Training Configuration:** The model is trained using a combination of loss functions appropriate for the two tasks:
   - ***Loss Functions:*** Sparse Categorical Cross entropy for sentiment classification.
   - Mean Squared Error (MSE) for star rating prediction.
   - ***Optimizer:*** The Adam optimizer is used with a learning rate of 0.0001. An Adam optimizer is used to minimize the loss functions, with early stopping implemented to halt training when performance on the validation set ceases to improve [48].
   - ***Metrics:*** Accuracy is used to evaluate the classification task, and mean squared error is used for the regression task.
7. ***Model Evaluation:*** The model's performance is evaluated using accuracy for the sentiment classification task and mean squared error (MSE) for the regression task. These metrics provide insight into how well the model is performing in predicting both the sentiment and the star rating from the reviews [49].

This architecture is designed to leverage both textual and numerical data effectively, providing robust predictions that can be used to personalize recipe recommendations based on user reviews [50].

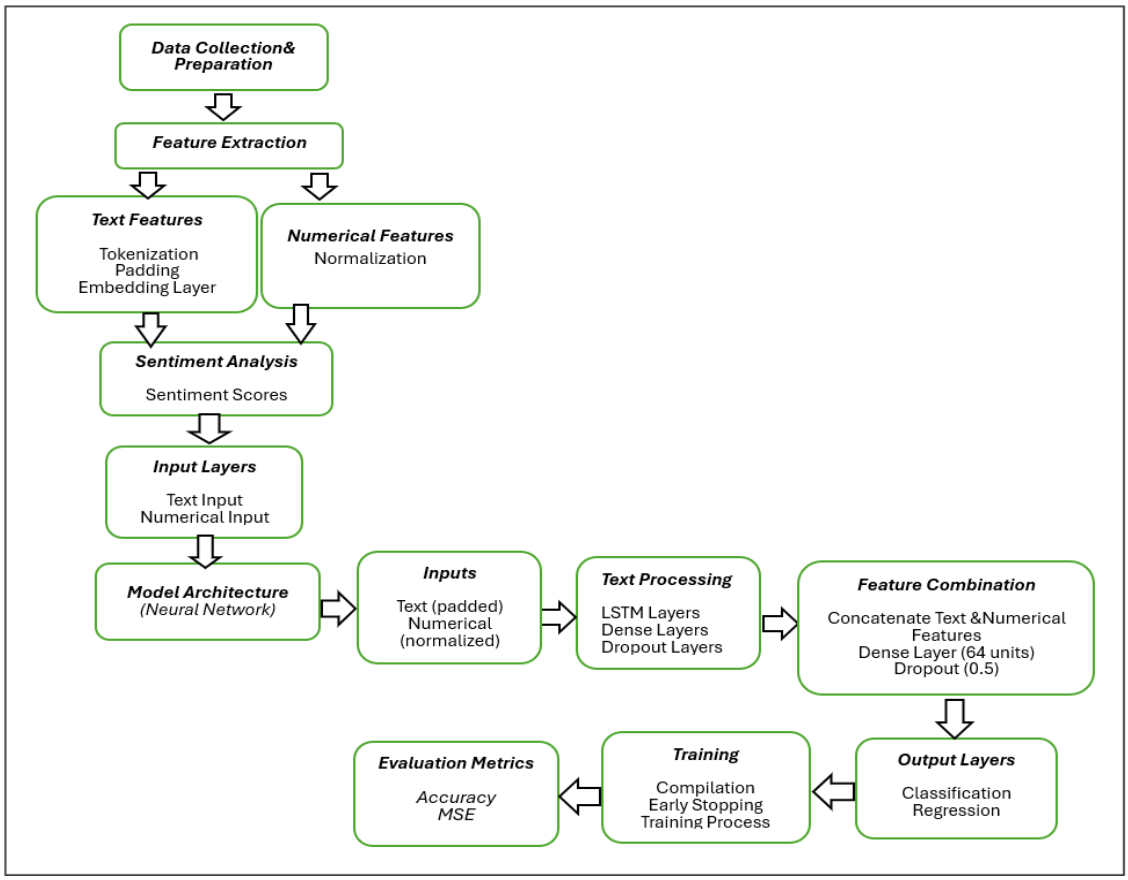


Figure 16: Designed RNN Model architecture

Design Justification:

The architecture is designed to address the specific needs of the recipe recommendation system:

- ***Sequential Data Processing:*** The use of LSTM layers allows the model to capture complex patterns in the text data, which is essential for accurately predicting sentiments based on user reviews.
- ***Feature Fusion:*** Combining textual and numerical data in a single model enables the system to make more informed predictions by leveraging both types of data. This approach improves the model's ability to predict star ratings while considering user sentiment.
- ***Multi-Task Learning*:** By jointly training the model on both sentiment classification and star rating prediction tasks allows the model to learn shared representations, which improves the overall performance and generalization capabilities.

Challenges and Considerations:

- ***High-Dimensional Text Data:*** The text input, after tokenization and padding, resulted in high-dimensional data. To address this, LSTM layers were used to reduce dimensionality and focus on relevant patterns to manage this complexity.
- ***Balancing Model Complexity:*** Ensuring that the model was complex enough to capture necessary patterns without overfitting posed a challenge. Dropout layers and early stopping techniques were used to prevent overfitting and optimize model performance.
- ***Data Imbalance:*** The sentiment classes were imbalanced, which could have led to biased predictions. We address this issue by applying SMOTE (Synthetic Minority Over-sampling Technique) to the text data, ensuring that the model was not biased towards the majority class, balanced representation across classes and it could better handle minority classes.

Schematic of the RNN Architecture: The RNN model used in the recipe recommendation system is designed to process sequential text data (e.g., user reviews) and numerical features. The model consists of an input layer, an LSTM recurrent layer, two dense (hidden) layers, and output layers responsible for sentiment classification and star rating prediction. The architecture of the Recurrent Neural Network (RNN) used in this project is composed of several key layers.

- **Input Handling:**
  - ***Text Input:*** Text data is processed through an embedding layer followed by bidirectional LSTM layers to capture sequential patterns.
  - ***Numerical Input:*** Numerical features undergo normalization to ensure consistent scaling.
- **Feature Combination:** The output from the LSTM layers is concatenated with the normalized numerical features, allowing the model to leverage both types of data.
- **Final Processing:** The concatenated features are passed through dense layers with dropout to prevent overfitting and to refine the combined feature representation.
- **Outputs:**
  - ***Sentiment Classification:*** A SoftMax activation function is applied for classifying sentiment into predefined categories.

* ***Star Rating Prediction:*** A linear activation function is used for predicting the star rating, providing a continuous output.

Temporal Data Processing: The RNN leverages the temporal structure of user reviews by processing sequences of words or phrases. Specifically, the LSTM layers analyze sequences of words within a review, where each sequence is treated as a temporal evolution of user sentiment. For instance, given 10 previous sequences (words or phrases), the LSTM layers predict the sentiment trajectory and its impact on the overall review score.

* Each review is tokenized and converted into a sequence of word embeddings, which serve as the intensity profiles (denoted as $h_z{-}10\Delta_z \ldots h_z{-}\Delta_z$) in this model.
* The intensity profile of each review is divided into bins, where each bin represents a word or phrase's embedding in the sequence.

LSTM Cell Functionality: The LSTM cells in the network are responsible for maintaining and updating hidden cell states as they process the sequence of word embeddings. This mechanism is critical for capturing long-term dependencies in the text data.

* ***Cell Input (x_i):*** The cell receives the input sequence (word embeddings) at each time step i (where i represents different points in the sequence).
* ***Hidden State (h_i) and Cell State (c_i):*** The LSTM cell updates the hidden state (h_i) and cell state (c_i) at each time step based on the input and previous states. These states are essential for predicting the next word in the sequence and, ultimately, the sentiment and star rating.

The operations within the LSTM cell, denoted by yellow rectangles for layer operations and orange circles for pointwise operations, manage the flow of information and the gradient during backpropagation, allowing the model to learn efficiently from the review data in below fig [51].

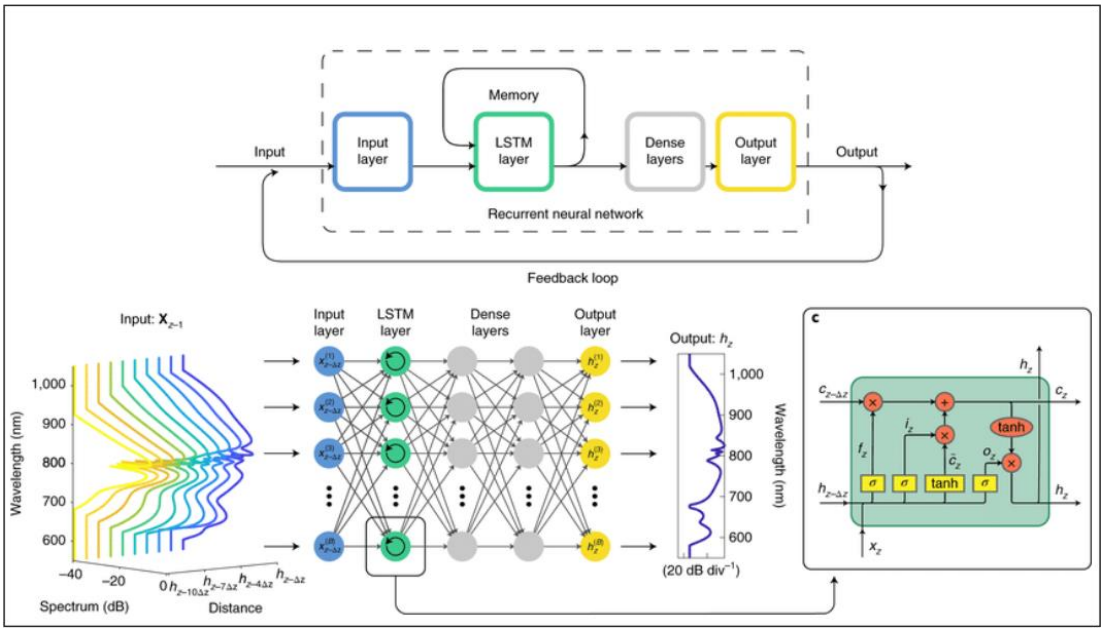


Figure 17: LSTM RNN Model Functionality

In our system, the RNN with LSTM layers processes the sequential text data (user reviews) to capture the underlying sentiment trajectory. This is particularly useful for understanding how the sentiment evolves within a review and how it influences the predicted star rating.

- ***Dense Layers:*** After processing through the LSTM layers, the output is passed through dense layers to further refine the feature representation before making predictions.
- ***Output Layers:*** Finally, the model outputs sentiment classifications via a SoftMax function and star rating predictions via a linear function**.**

This architecture is well-suited for handling the complex, sequential nature of user reviews and combining it with numerical features (user reputation, thumbs up/down) to provide more accurate and personalized recipe recommendations. It is designed to maximize the model's ability to predict both sentiment and star ratings from user reviews effectively.

Chapter 5: Implementation

5.1. Model Development and Training:

Introduction to Recommender Systems: Recommender systems are designed to provide personalized suggestions to users by analyzing their preferences and behavior. These systems have been implemented in various domains such as e-commerce, entertainment, and social media to enhance user engagement and satisfaction [52]. The two primary methods used in recommender systems are content-based filtering and collaborative filtering. Content-based filtering focuses on the attributes of items and user's preferences, while collaborative filtering leverages user interactions and preferences to make recommendations.

Now we are focusing on building and training the models that form the backbone of the recipe recommendation system. The development of the recipe recommendation system was approached through a multi-faceted process that required careful consideration of various techniques to ensure personalized and accurate recommendations. Each model was built to capture different aspects of user preferences and recipe characteristics, with these models ultimately being blended into a comprehensive system.

Collaborative Filtering (SVD):

The development process began with implementing Collaborative Filtering using Singular Value Decomposition (SVD). Collaborative filtering is a popular recommendation technique that leverages the preferences and feedback of multiple users to generate personalized suggestions. The core idea is that if users A and B have similar tastes, recommendations made to user A should also be relevant to user B, and vice versa [53]. This approach is particularly effective in capturing latent patterns and preferences that might not be immediately apparent. This method was chosen for its ability to capture latent factors that represent user preferences and item features by decomposing the user-item interaction matrix into three matrices. The model was trained on a dataset of user ratings, and fine-tuning of hyperparameters was carried out to enhance its predictive accuracy.

Implementation of Singular Value Decomposition (SVD):

In this project, Collaborative Filtering was implemented using Singular Value Decomposition (SVD). The SVD technique was selected for its ability to uncover hidden patterns in user preferences by decomposing the user-item interaction matrix into three distinct matrices: user factors, item factors, and a diagonal matrix of singular values [54]. This decomposition helps in capturing latent factors that influence a user's taste in recipes, such as their preference for certain ingredients or cuisines.

SVD Technique:

- **Matrix Decomposition:** The user-item interaction matrix is decomposed into three matrices:
    - **User Factors Matrix:** Captures user preferences.
    - **Item Factors Matrix:** Captures item characteristics.
    - **Diagonal Matrix of Singular Values:** Represents the strength of the latent factors.

- **Latent Factors:** These matrices collectively help in understanding how users' preferences align with item features, such as ingredients or cuisines.

Collaborative filtering is utilized to make recommendations by analyzing the behavior and preferences of similar users or items. This method's strengths and limitations, such as issues with cold-start or data sparsity, are considered when implementing the system.

Training the Model: The SVD model was trained using a rich dataset of user ratings, allowing it to learn from past interactions and predict future preferences. Hyperparameters were carefully tuned to strike the optimal balance between accuracy and generalization. This involved adjusting parameters such as the number of latent factors, learning rate, and regularization terms to enhance the model's performance.

- **Hyperparameter Tuning:** To enhance model performance, adjusting hyperparameters such as the number of latent factors, learning rate, and regularization terms to optimize the model's performance. This tuning process aimed to achieve a balance between accuracy and generalization
- **Learning from Past Interactions:** The model used historical user interactions to predict future preferences, leveraging past behavior to recommend relevant recipes.

Evaluation and Challenges:

After training the SVD model, its performance was evaluated using several metrics. These metrics included:

- **Precision and Recall:** These metrics assessed the relevance and completeness of the recommendations.
- **Root Mean Squared Error (RMSE):** This measure evaluated the accuracy of predicted ratings compared to actual user ratings.

The implementation of collaborative filtering using SVD captured latent factors influencing user preferences and delivered accurate and relevant recipe recommendations. The model's effectiveness was evident in the personalized suggestions it provided, though addressing the challenges of cold-start and data sparsity remains an area for future improvement. The recommendations generated for users, including those for **User ID u_14Jt3Onn7vE8**, were saved and reviewed, showcasing the practical application of the SVD-based approach in delivering tailored suggestions.

```
Duplicates after aggregation: Empty DataFrame
Columns: [recipe_id, recipe_number, recipe_code, recipe_name, review_id, user_id, username, user_reputation, created_at, reply_count
Index: []

[0 rows x 22 columns]
User-Item Matrix Shape: (12600, 721)
     User ID:  u_14Jt3Onn7vE8
Generating recommendations for User ID: u_14Jt3Onn7vE8
```

	recipe_id	recipe_name	star_rating	processed_review_text	sentiment	sentiment_score
13	13	Creamy White Chili	0.460429	made alongside traditional chili winter gather...	Positive	0.8519
48	48	Creamy White Chili	0.460429	made movie night friend got rave review though...	Positive	0.8312
95	95	Creamy White Chili	0.460429	loved perfect little spice	Positive	0.8225
106	106	Creamy White Chili	0.460429	husband made night crowd delicious add jalapen...	Positive	0.8555
663	9	Best Ever Banana Bread	0.460429	best banana bread ever used pecan instead waln...	Positive	0.8360
...	...	...	...	...	...	...
18046	95	Egg Roll Noodle Bowl	0.460429	keeper house used chicken changed noodle chine...	Positive	0.6249
18057	106	Egg Roll Noodle Bowl	0.460429	truly simple delicious recipe definitely keepe...	Positive	0.8910
18072	9	Mamaw Emily's Strawberry Cake	-1.481556	bit rich taste make think would light whipped ...	Positive	0.7269
18158	95	Mamaw Emily's Strawberry Cake	0.460429	wonderfully moist rich strawberry flavor husba...	Positive	0.9493
18169	106	Mamaw Emily's Strawberry Cake	-0.834228	heaven people act like new invention recipe ar...	Positive	0.8910

374 rows × 6 columns

Figure 18: Recommend Recipes for specific Customer

The evaluation results revealed that the SVD model was effective in identifying and recommending recipes that matched users' preferences. However, the analysis also highlighted some challenges, such as cold-start problems and data sparsity. Cold-start problems occur when there is insufficient data for new users or items, while data sparsity refers to a lack of user-item interactions. Despite these challenges, the SVD model demonstrated its capability to provide relevant and personalized recipe recommendations.

The analysis of these metrics revealed valuable insights into the model's performance, highlighting its strengths and areas for improvement. The SVD model demonstrated the capability to recommend recipes that aligned with users' tastes, though there were challenges related to cold-start problems and data sparsity, which are common in collaborative filtering approaches.

Content-Based Filtering (TF-IDF and Cosine Similarity):

In content-based filtering, recommendations are generated based on the features and attributes of the items and users. For instance, if a user's preference for sci-fi movies is known, similar movies are suggested by the system based on genre or actors. In our recipe recommendation, Content-based filtering is used to be employed to recommend recipes based on the features and attributes of the recipes and user preferences [55]. Here, if a user frequently engages with recipes that are high in protein, a content-based system might suggest other recipes with similar nutritional profiles or ingredients. The impact of content-based filtering is measured using metrics such as precision, recall, and F1-score, which assess how relevant and comprehensive the recommendations are. Additionally, metrics such as diversity, novelty, and uncertainty are utilized to determine how diverse, unexpected, and surprising the recommended recipes.

Implementation of Cosine Similarity and TF-IDF:

The implementation begins by preparing the data, where essential columns such as sentiment score, sentiment polarity, star rating, thumbs up, thumbs down, and user reputation are ensured to be present. Missing values are handled appropriately to maintain data integrity and a feature matrix is then created using these selected columns, which serves as the basis for computing cosine similarity between recipes. This similarity measure allows the system to identify and recommend recipes that are similar to a given recipe, excluding the selected recipe itself. This approach focused on understanding the content of the recipes themselves, with Term Frequency-Inverse Document Frequency (TF-IDF) being used to represent recipe descriptions. Cosine Similarity was then calculated between these descriptions, enabling the model to identify and recommend recipes similar to those the user had previously enjoyed.

Training the Model:

A new dimension was added to the recommendation system by the Content-Based Filtering model, which considered the textual content of recipes. This allowed the system to suggest dishes that matched the specific ingredients, cuisines, or cooking styles preferred by the user, making the recommendations more tailored and relevant.

A key feature of the system is its interactivity. Users can input a recipe index, and the system will generate a list of the top 20 most similar recipes based on the precomputed cosine similarity scores. This interactive element enables users to explore and discover new recipes that align closely with their preferences, making the recommendation process more engaging and personalized.

After the implementation of this model, its performance was thoroughly evaluated. The final results highlighted the model's ability to generate personalized recommendations based on the content analysis of recipe descriptions. Precision and recall metrics were pivotal in assessing how well the recommendations aligned with user preferences.

Evaluation:

To evaluate the effectiveness of these recommendations, precision, recall, and F1-score are calculated, comparing true labels (based on sentiment scores) with predicted labels. These metrics provide insight into how relevant and comprehensive the recommendations are.

After generating the recommendations, the system removed any duplicate recipe names to ensure that each recommendation was unique, and the results were saved in a CSV file. This CSV file can then be used in Power BI for visualization, enabling further analysis of the recommended recipes and their impact on user preferences. Additionally, the recommended recipes, along with their corresponding sentiment scores and processed review text, were displayed to the user, offering a comprehensive view of the suggestions. This implementation not only generates personalized recipe suggestions but also provides a framework for evaluating and refining the recommendation system based on user feedback and engagement.

In an interactive session, the system was tested by entering a specific recipe index (8 and 16). The system then generated a list of the top 20 most similar recipes based on cosine similarity.

The metrics for this specific session were as follows:

- Precision: 1.00
- Recall: 1.00
- F1 Score: 1.00

These scores indicate that the system performed perfectly in this case, successfully recommending recipes that were all relevant according to the sentiment analysis. The system provided the following recipe recommendations based on the input recipe index: 8 and 16.

```
Enter a recipe index to get recommendations: 8
Precision: 1.00, Recall: 1.00, F1 Score: 1.00
```

	recipe_id	recipe_name	processed_review_text	sentiment	sentiment_score	sentiment_polarity
17636	8	Brown Sugar Oatmeal Pancakes	made great next time drop sugar cup sweet like...	Positive	0.9451	0.256944
3223	37	Apple Pie	first foray baking apple pie ive done lot diff...	Positive	0.9867	0.291260
11820	41	Grilled Huli Huli Chicken	use sweet baby ray bbq sauce instead ketchup g...	Positive	0.8625	0.286667
13329	0	Pork Chops with Scalloped Potatoes	first time wrote review absolutely hate read r...	Positive	0.7845	0.166667
13500	14	Chicken and Dumplings	amazes think way always way grew southern geor...	Positive	0.7842	0.193750
10058	0	Shrimp Scampi	ive made recipe several time added little garl...	Positive	0.8490	0.104167
30	30	Creamy White Chili	absolutely delicious first time making chili l...	Positive	0.9565	0.363889
17966	15	Egg Roll Noodle Bowl	really enjoyed recipe read comment prior makin...	Positive	0.8929	0.350000
2821	3	Flavorful Chicken Fajitas	recipe keeper made chicken steak shrimp steak ...	Positive	0.9702	0.166667
14688	0	Blueberry French Toast	five star recipe used sara lee artesano brioch...	Positive	0.8040	0.270833
9274	8	Peanut Butter Chocolate Dessert	fantastic dessert actually freeze enjoy longer	Positive	0.7906	0.266667
7992	15	Macaroni Coleslaw	one family favorite salad make quite often sum...	Positive	0.8225	0.311111
5998	30	Fluffy Key Lime Pie	love easy flavorful summer recipe key lime pie...	Positive	0.9792	0.380303
12399	0	Cheeseburger Paradise Soup	omg hearty soup bomb easy cut recipe half smal...	Positive	0.9042	0.252083
4225	45	Traditional Lasagna	made recipe since appeared year ago absolutely...	Positive	0.8981	0.259524
18099	36	Mamaw Emily's Strawberry Cake	actually recipe year called strawberry party c...	Positive	0.9269	0.230952
5713	13	Rustic Italian Tortellini Soup	delicious simple soup cooked exactly written p...	Positive	0.9216	0.385417
17019	18	Tennessee Peach Pudding	found recipe week ago made three time already ...	Positive	0.8625	0.290909
12875	3	Smothered Chicken Breasts	really good made twice second time loosely fol...	Positive	0.7574	0.283333
1961	73	Amish Breakfast Casserole	delicious next time might try using sausage in...	Positive	0.9201	0.300000

Figure 19: Content-Based Model Recommendations

```
Enter a recipe index to get recommendations: 16
Precision: 1.00, Recall: 1.00, F1 Score: 1.00
```

	recipe_id	recipe_name	processed_review_text	sentiment	sentiment_score	sentiment_polarity
8789	59	Basic Banana Muffins	would star spice luckily realized putting cupc...	Positive	0.8591	0.511111
11018	25	Black Bean 'n' Pumpkin Chili	like idea adding pumpkin chili enjoyed recipe ...	Positive	0.8074	0.500000
8398	36	Winning Apple Crisp	first time making apple crisp thought good rea...	Positive	0.8625	0.510000
16585	102	First-Place Coconut Macaroons	delicious easy complaini followed receipt tee ...	Positive	0.8020	0.483333
11258	103	Simple Au Gratin Potatoes	really quick easy sauce make hour turn oven ge...	Positive	0.7951	0.488889
5275	98	Zucchini Cupcake	made gluten free sugar free still good	Positive	0.8591	0.500000
9830	76	Taco Lasagna	good recipe reduce amount fat used fat free re...	Positive	0.8591	0.500000
7118	62	Baked Spaghetti	love recipe put spaghetti sauce mine like much...	Positive	0.8625	0.500000
15157	82	Baked Tilapia	eliminated caper dont like forgot add butter m...	Positive	0.8552	0.544444
6365	171	Stuffed Pepper Soup	love recipe problem isnt recipe comment made s...	Positive	0.8360	0.483333
14170	127	Chicken Wild Rice Soup	really good although season make huge lot hope...	Positive	0.8655	0.550000
13913	121	Creamy Coleslaw	sometimes coleslaw sweet sometimes tart good b...	Positive	0.8126	0.525000
17369	43	Teriyaki Chicken Thighs	easy make love serve rice	Positive	0.7964	0.466667
15269	64	Chocolate-Strawberry Celebration Cake	love idea cake really beautiful generally dont...	Positive	0.8065	0.466667
4978	198	Basic Homemade Bread	baker trade im home like use simple recipe use...	Positive	0.8779	0.562500
974	320	Best Ever Banana Bread	like sweet banana bread excellent difference r...	Positive	0.9042	0.566667
13240	73	Slow-Cooker Lasagna	stated additional spice really make recipe exc...	Positive	0.8055	0.525000
10649	73	Garlic Beef Enchiladas	good take bit time dish prepare added cup ques...	Positive	0.9081	0.525000

Figure 20: Recommendations for the Given Recipe Index

The developed recipe recommendation system using cosine similarity not only provides personalized suggestions based on user preferences but also incorporates an interactive component that allows for real-time exploration of recommendations. The system's performance is evaluated using standard metrics, and the recommendations are saved for further analysis, ensuring that the system is both user-friendly and analytically robust.

The interactive nature of the system, combined with its high precision and recall, makes it a valuable tool for personalized recipe recommendations. Future work may involve refining the model further to handle more complex scenarios and user preferences.

The combined output from the Collaborative Filtering and Content-Based Filtering models was used to generate a list of recommended recipes. The results from both models were blended to create a more comprehensive recommendation system, leveraging the strengths of each approach. This ensured that the recommendations not only matched user preferences based on previous interactions but also introduced new items that were contextually similar to those the user had shown interest in.

With both the Collaborative Filtering and Content-Based Filtering models in place, the focus was shifted to generating a unified list of recommended recipes. By blending the results from both models, a more comprehensive recommendation system was created. This hybrid approach ensured that the recommendations not only reflected past user interactions but also introduced new recipes that were contextually similar to those already enjoyed.

Hybrid systems combine content-based and collaborative filtering to leverage the strengths and overcome the weaknesses of both methods. For example, a hybrid system might use content-based filtering to deal with the cold start problem, when there is not enough data about new users or items and use collaborative filtering to exploit the wisdom of the crowd and discover latent patterns [56]. To measure the impact of hybrid systems, we can use the same metrics as for content-based and collaborative filtering, but also consider metrics such as synergy, which measures how much the hybrid system improves over the individual methods.

Sentiment Classification Using Multinomial Naive Bayes:

For sentiment classification of recipe reviews, we employed the Multinomial Naive Bayes classifier. This model is particularly effective for text classification tasks where features are discrete, such as word counts or term frequencies. The MultinomialNB algorithm is suitable for this purpose due to its simplicity and efficiency in handling large datasets [57].

To classify sentiment in the recipe reviews, we utilized the Multinomial Naive Bayes classifier (MultinomialNB) model, which is well-suited for text classification tasks involving discrete features.

For preparing the data, we first transformed the review texts into numerical features using the TF-IDF (Term Frequency-Inverse Document Frequency) vectorization method. TF-IDF is effective in capturing the importance of each term within the document as well as across the corpus. Alongside this, we normalized additional numerical features, including user_reputation, thumbs_up, thumbs_down, star_rating, and best_score, to a range between 0 and 1 using MinMaxScaler. This

normalization ensures that each feature contributes equally to the model, preventing any single feature from dominating the results.

The MultinomialNB model was then trained on the combined dataset of text and numerical features. We optimized the hyperparameter alpha, which controls the smoothing of the Naive Bayes model. Through grid search, we determined that an alpha value of 0.1 yielded the best results, balancing bias and variance effectively.

Evaluation:

Cross-Validation Performance: To evaluate the model's performance, we used cross-validation.

- Cross-Validation Scores: [0.9375, 0.9375, 0.9375, 0.9378, 0.9381]
- Mean Cross-Validation Score: 0.9377 (approximately 93.77%)

This high mean score indicates that the model performs consistently well across different subsets of the data.

Classification Report: The classification report reveals that the model achieved an overall accuracy of 91%, indicating strong performance. Precision, recall, and F1 scores were computed for each sentiment class Negative, Neutral, and Positive. The model demonstrated high precision and recall for the Negative and Neutral classes, signifying that it accurately identifies and classifies these sentiments. For Positive reviews, although the recall was high, the precision was slightly lower, suggesting some misclassifications in this category.

- **Precision:** Measures the accuracy of positive predictions. High precision for all classes indicates that the model is effective at avoiding false positives.
- **Recall:** Measures the ability of the model to identify all relevant instances. High recall for each class shows the model's efficiency in capturing true positives.
- **F1-Score:** The harmonic mean of precision and recall. A high F1 score indicates a good balance between precision and recall.

Confusion Matrix:

```
Best Parameters: {'alpha': 0.1}
Best Score: 0.8983265675647869
Classification Report:
              precision    recall  f1-score   support

    Negative       0.88      0.91      0.90      4551
     Neutral       0.94      0.93      0.94      4647
    Positive       0.90      0.88      0.89      4699

    accuracy                           0.91     13897
   macro avg       0.91      0.91      0.91     13897
weighted avg       0.91      0.91      0.91     13897

Confusion Matrix:
[[4158  123  270]
 [ 130 4344  173]
 [ 421  162 4116]]
```

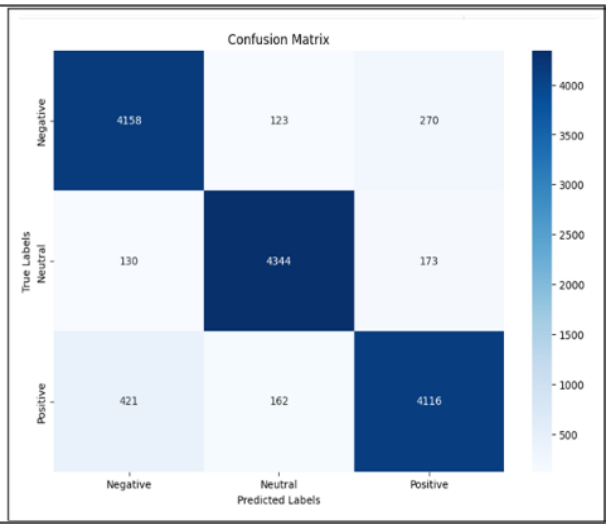


Figure 21: Classification Report &Confusion Matrix for Multinomial Naive Bayes

To evaluate the model's performance, we employed cross-validation. The cross-validation scores were consistently high, with values of approximately 0.9375 across different folds, resulting in a mean cross-validation score of 0.9377. This suggests that the model performs reliably across various subsets of the data.

The confusion matrix further illustrates the model's performance, showing a high number of correct classifications along the diagonal. However, some misclassifications are evident, particularly between the Negative and Positive classes. Despite these errors, the model shows a robust ability to classify sentiments correctly overall.

To further improve performance, future work could involve advanced text preprocessing techniques, exploring ensemble methods to combine different classifiers, and conducting additional validation to ensure model robustness. We further enhanced the sentiment classification by implementing BERT (Bidirectional Encoder Representations from Transformers), a state-of-the-art transformer-based model. BERT, developed by Google, represents a significant advancement in natural language processing due to its ability to understand context and semantics in a bidirectional manner. Unlike traditional models, BERT captures the nuanced meaning of words based on their surrounding words, which improves its performance on complex text classification tasks.

Deep Learning:

Deep Learning is a subset of machine learning that aims to mimic the workings of the human brain using artificial neural networks. These networks consist of layers of neurons that learn complex representations of data through multiple transformations. Common types of neural networks are:

- ***Artificial Neural Networks (ANNs):*** Basic networks where neurons are organized in layers, and information flows from input to output through hidden layers.
- ***Convolutional Neural Networks (CNNs):*** Specialized for processing grid-like data, particularly images.
- ***Recurrent Neural Networks (RNNs):*** Designed for sequential data, RNNs use connections that loop back to previous states, enabling the model to remember information from earlier inputs. They are widely used in tasks involving time series, text, and speech data.

The main advantage of deep learning is its ability to automatically learn relevant features from raw data, making it highly effective for complex tasks like sentiment analysis and recommendation systems [58].

Text Classification with RNN:

Recurrent Neural Networks (RNNs) are a widely used supervised deep learning approach, particularly effective in tasks involving sequential data, such as text classification and time series analysis. Other deep learning architectures like CNNs and ANNs serve different purposes, with each type of network designed for specific tasks. Deep learning, in general, mimics the function of the human brain, with each neural network structure representing a different cognitive process. By modeling these brain-like functions, deep learning enables machines to perform tasks such as image recognition, natural language processing, and decision-making with high accuracy.

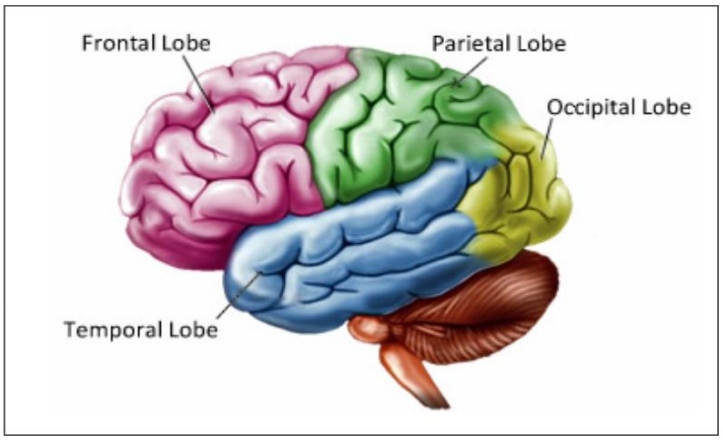


Figure 22: Analogy of Brain Function

In this analogy, each neural network structure loosely represents a specific function or region of the brain [59]:

- Artificial Neural Networks (ANNs) resemble the function of the brain's temporal lobe, which stores information for extended periods.
- Convolutional Neural Networks (CNNs) correspond to the occipital lobe, responsible for processing visual information.
- Recurrent Neural Networks (RNNs) mirror the brain's frontal lobe, handling short-term memory and sequential processing, making them ideal for working with sequential or time-dependent data like text.

For tasks such as sentiment analysis and star rating prediction in recipe reviews, RNNs specifically Long Short-Term Memory (LSTM) networks are highly effective. LSTMs retain and leverage previous information from sequences, allowing them to accurately classify sentiment and predict star ratings by analyzing both textual and numerical inputs.

Recurrent Neural Networks (RNNs) for Sentiment Analysis:

Recurrent Neural Networks (RNNs) are well-suited for processing sequence data such as text due to their ability to capture temporal dependencies between words or phrases. In sentiment analysis, RNNs can effectively comprehend the context within user reviews, making them an ideal choice for classifying sentiment in recipe reviews.

In this project, we implemented a comparative analysis of sentiment analysis methods. We evaluated both **Text Blob**, a lexicon-based approach, and **BERT,** a transformer-based model, to determine the most effective method for sentiment classification in our recipe review dataset. The goal was to select the approach that provided the highest accuracy and robustness for our recipe recommendation system.

Implementation of RNN Model Using LSTM for Sentiment Analysis and Star Rating Prediction

Before applying deep learning models, we first used Text Blob for sentiment analysis. Text Blob is a library built on top of NLTK and Pattern, providing a straightforward API for common natural language processing (NLP) tasks. Text Blob is a straightforward tool that uses a rule-based approach to classify sentiment based on polarity scores. This initial analysis allowed us to quickly categorize reviews and understand the distribution of sentiment in our dataset. The results indicate that the majority of reviews were classified as Positive, with fewer reviews categorized as Negative or Neutral.

Although Text Blob provided quick and reasonable results, its rule-based approach has limitations in handling complex and nuanced sentiments. To enhance our sentiment analysis, we implemented BERT (Bidirectional Encoder Representations from Transformers), a deep learning model known for its advanced contextual understanding.

Text Blob Sentiment Analysis:

Text Blob is a simple library for processing textual data, providing a straightforward way to perform sentiment analysis. It uses a rule-based approach to classify sentiment based on polarity scores. In our implementation, we defined a function get_sentiment that categorizes reviews into 'Positive', 'Neutral', or 'Negative' based on the polarity score.

While BERT offered advanced capabilities and potentially higher accuracy due to its deep learning architecture, Text Blob provided a highly satisfactory performance with less computational complexity. Text Blob's simplicity made it easier to implement and interpret, yielding effective results for our use case. Text Blob was found to be more practical and sufficient for our initial analysis. The decision to use Text Blob was based on its balance between performance and ease of use, which proved to be effective for our dataset of recipe reviews.

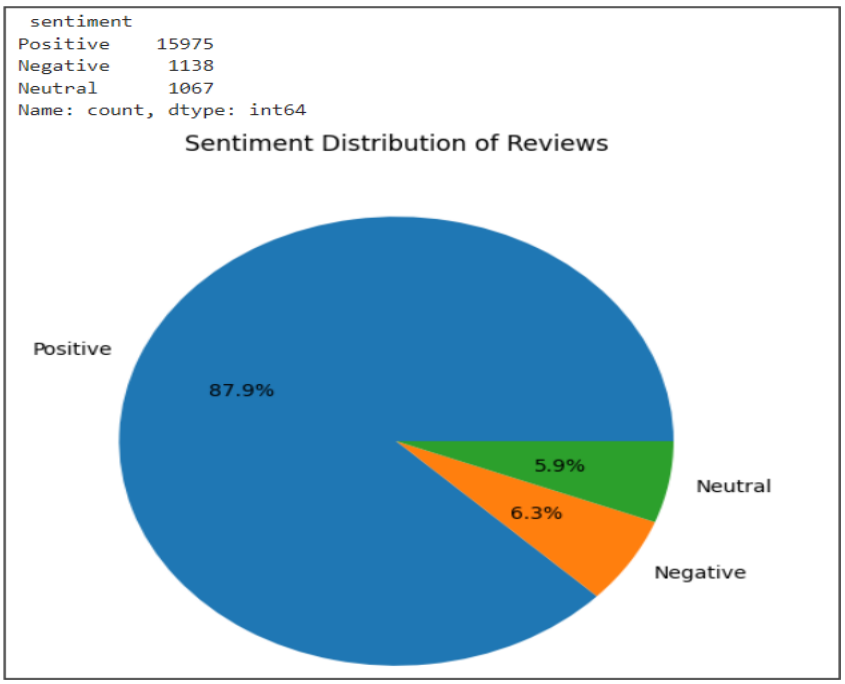


Figure 23: Sentiment Analysis Distribution

RNN Implementation in the Recipe Recommendation System:

To develop a Neural Network, we explore various machine learning models for analyzing sentiment and predicting star ratings from recipe reviews. We employ both deep learning techniques and classical machine learning methods to assess their performance and determine the most effective approach.

In this project, we implemented a Recurrent Neural Network (RNN), specifically a Bidirectional Long Short-Term Memory (LSTM) network, to analyze user reviews and predict both the sentiment of the review (classification) and the corresponding star rating (regression).

1. Model Architecture of RNN:

1.1. Text Preprocessing:

We started by preprocessing the dataset of recipe reviews, ensuring that all text data was properly formatted, and numerical features were selected. Text data was tokenized and padded for use with LSTM models, while numerical features included user reputation, thumbs up, thumbs down, star rating, and best score. The dataset of recipe reviews was preprocessed to prepare both textual and numerical data for the RNN model. Key steps are:

- ***Tokenization:*** Text data was tokenized, converting words into sequences of indices.
- ***Padding:*** Sequences were padded to ensure uniform length, necessary for input into LSTM models.
- ***Numerical Features:*** Additional features such as user reputation, thumbs-up/down, star ratings, and best score were selected and prepared for integration with the text-based input.
- ***Feature Scaling and Balancing:***
    - ***Scaling:*** Numerical features were scaled using StandardScaler to normalize their values, improving model performance by ensuring that all features contribute equally to the learning process.
    - ***Balancing:*** SMOTE (Synthetic Minority Over-sampling Technique) was applied to address the class imbalance in the sentiment classification task. By generating synthetic samples, SMOTE helps in achieving a balanced representation of different sentiment classes, which is critical for training a robust classification model.

1.2. LSTM Layers:

The RNN model for sentiment analysis consisted of an embedding layer followed by two Bidirectional LSTM layers. The Bidirectional LSTM layers help capture dependencies in both forward and backward directions in the text, enhancing the model's ability to understand sentiment. The final dense layer with a SoftMax activation function outputs the probability distribution over sentiment classes. The RNN model was designed to handle the sequential nature of text data:

- ***Embedding Layer:*** This layer mapped each word index to a dense vector representation, allowing the model to learn semantic relationships between words.

- ***Bidirectional LSTM Layers:*** Two Bidirectional LSTM layers were used to capture both forward and backward dependencies in the text. This allowed the model to understand the full context of user reviews, improving its ability to interpret sentiment.
- ***Final Dense Layer:*** The output from the LSTM layers was passed through a dense layer with a SoftMax activation function, which produced the probability distribution over sentiment classes (positive, neutral, negative).

The model was compiled using the **Adam optimizer** and **categorical cross-entropy** loss function. **Accuracy** was the key metric used to evaluate the performance of the sentiment classification task. To avoid overfitting, **early stopping** was implemented, which terminated training when no further improvement in validation loss was detected.

1.3. Output Layers: The model had two outputs:

- **Classification Output:** A softmax layer was used to predict the sentiment of the review (positive, neutral, or negative).
- **Regression Output:** A dense layer with a single unit was used to predict the star rating of the review.

1.4. Traditional Models for Comparison: In addition to the RNN model, classical machine learning models were implemented as baselines.

We also implemented traditional machine learning models including:

- Logistic Regression was used for sentiment classification.
- Support Vector Regression (SVR) and Random Forest Regressor were used for star rating prediction.

These models provided a performance benchmark against which the deep learning approach was compared.

2. Sentiment Prediction: We tested our RNN model with new recipe reviews to predict sentiment. The predictions showed that the model could effectively classify sentiments into positive, neutral, or negative categories, providing valuable insights into user feedback.

The RNN model was tested with new recipe reviews to evaluate its sentiment classification performance. Here are the new reviews we used for predictions:

Review 1: "This recipe was amazing! I loved it."

- Predicted Sentiment: Positive
- Predicted Sentiment Score: [0.87024]

The review uses enthusiastic language with terms like "amazing" and "loved," which strongly indicates a positive sentiment. The RNN model correctly classifies this review as positive, aligning with the sentiment expressed in the review.

The sentiment score of 0.87024 is positive, reflecting the positive sentiment of the review. This score is not only a classification but also indicates the strength of the sentiment, where higher values suggest stronger positive feelings.

Review 2: "The dish was too salty for my taste."

- Predicted Sentiment: Negative
- Predicted Sentiment Score: [-0.026190476]

The review indicates dissatisfaction due to excessive (it being too salty) saltiness, aligning with a negative sentiment. The model correctly identifies the review as negative, with a slightly negative score reflecting mild negativity. The model correctly identifies this review as negative, which is appropriate given the review's content. The issue described (excessive saltiness) is a negative aspect of the dish.

The sentiment score of -0.026190476, though close to zero, is slightly negative. This score reflects the minor negative sentiment expressed in the review. The low magnitude of the score suggests that the negativity is mild, which corresponds with the review's relatively restrained critique.

3. Performance Metrics and Results: The model was trained using two types of losses:

- Categorical Cross-Entropy for the sentiment classification task.
- Mean Squared Error (MSE) for the star rating regression task.

To prevent overfitting, dropout layers were used, and early stopping was applied during training. The model was evaluated on both accuracy (for sentiment classification) and MSE (for regression).

- **Accuracy** for the sentiment classification.
- **Mean Squared Error (MSE)** for the star rating regression

Performance Results:

Sentiment Classification:

The deep learning RNN model achieved a **classification accuracy of 90.98%** for sentiment prediction. This Accuracy is very good for a multi-class classification problem (positive, neutral, negative). A high accuracy means our model successfully understands the context of the reviews and categorizes sentiments correctly.

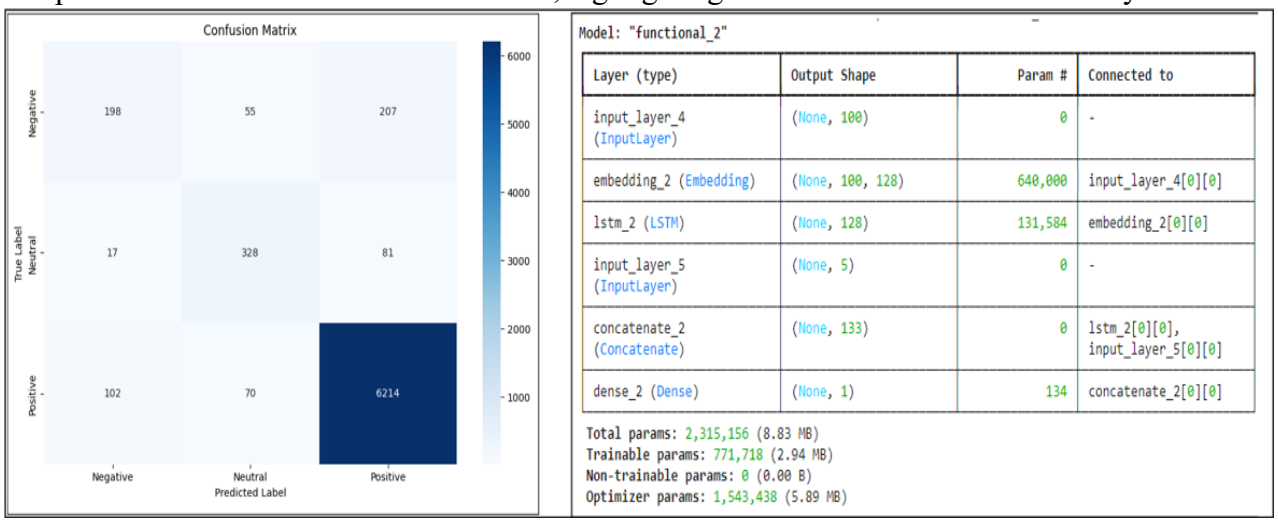
The performance comparison with Logistic Regression (**91.61%**) shows that the deep learning model performs similarly, though slightly lower. Given that Logistic Regression is often a strong baseline, it's common for deep learning models to outperform classical models when more data or features are involved.

Star Rating Regression:

For star rating prediction, the RNN achieved a Mean Squared Error (MSE) of **0.0254** for the test set is quite low, which indicates that our model can predict star ratings very accurately. Lower MSE implies fewer errors between predicted and actual ratings. The comparison with SVR (MSE 0.0498) and Random Forest Regressor (MSE 0.0424) shows that our RNN has a clear advantage in star rating prediction.

The model demonstrates its ability to classify sentiments based on the review text accurately. In this case, the review's negative aspect is correctly classified, showing that the model can effectively discern between positive, neutral, and negative sentiments. Sentiment scores provide a quantitative measure of the sentiment's strength. Positive scores indicate positive sentiment, while negative scores suggest negative sentiment. In this case, the negative score aligns with the review's critical tone, even if it's not very strong.

The RNN model's performance in classifying sentiments and generating meaningful sentiment scores demonstrates its effectiveness. It can interpret the sentiment context of reviews accurately and provide detailed sentiment assessments, highlighting its robustness in sentiment analysis tasks.



Model: "functional_2"

Layer (type)	Output Shape	Param #	Connected to
input_layer_4 (InputLayer)	(None, 100)	0	-
embedding_2 (Embedding)	(None, 100, 128)	640,000	input_layer_4[0][0]
lstm_2 (LSTM)	(None, 128)	131,584	embedding_2[0][0]
input_layer_5 (InputLayer)	(None, 5)	0	-
concatenate_2 (Concatenate)	(None, 133)	0	lstm_2[0][0], input_layer_5[0][0]
dense_2 (Dense)	(None, 1)	134	concatenate_2[0][0]

Total params: 2,315,156 (8.83 MB)
Trainable params: 771,718 (2.94 MB)
Non-trainable params: 0 (0.00 B)
Optimizer params: 1,543,438 (5.89 MB)

Figure 24: Confusion Matrix and Model Summary of RNN model

Classification Report:

	precision	recall	f1-score	support
Negative	0.63	0.38	0.47	341
Neutral	0.73	0.61	0.66	303
Positive	0.95	0.98	0.96	4810
accuracy			0.90	5454
macro avg	0.77	0.66	0.70	5454
weighted avg	0.91	0.92	0.92	5454

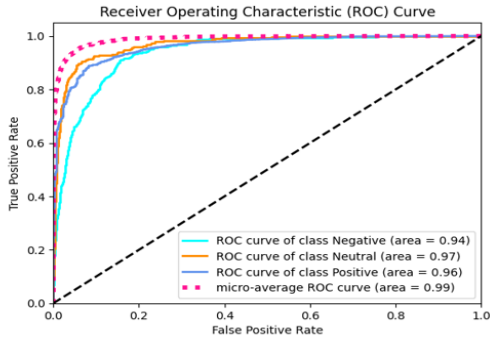


Figure 25: Classification Report and Roc Curve of RNN Model

4. Role Sentiment Analysis in the Recommendation System:

The RNN model has proven effective in classifying sentiments and predicting star ratings from user reviews. Its ability to accurately interpret sentiment and provide detailed sentiment scores highlights its robustness in sentiment analysis tasks. The comparative performance against traditional models underscores the effectiveness of the deep learning approach in this context.

Recurrent Neural Networks (RNNs) were employed to capture the sequential nature of text data in user reviews. The sentiment analysis output was integrated into the recommendation system as an additional feature, enhancing the personalization of recipe suggestions. By incorporating sentiment analysis, the recommendation engine could better tailor recommendations based on user feedback, considering both positive and negative sentiments expressed in the reviews.

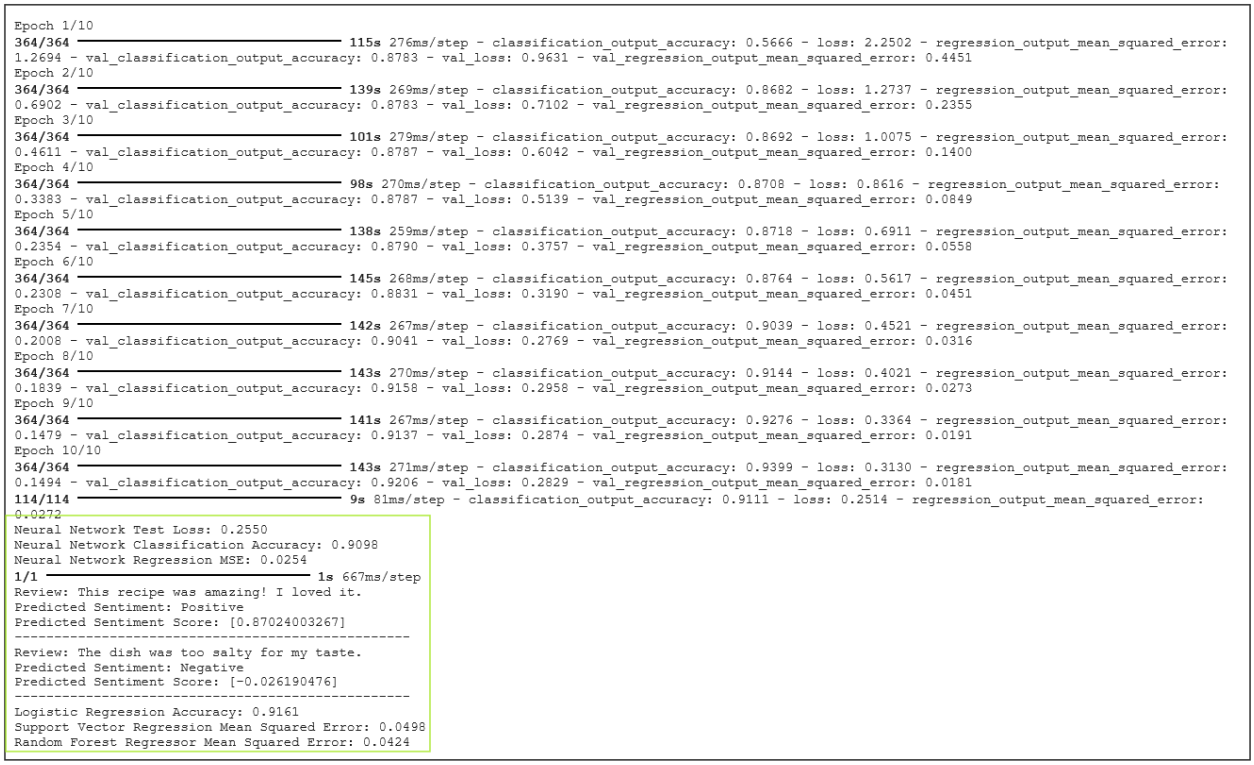


Figure 26: Predicted Output of New Reviews of RNN model

Sentiment Analysis was conducted using Recurrent Neural Networks (RNNs) to analyze user reviews. The RNN model was trained to understand the sentiment expressed in the text, providing an additional layer of personalization to the recommendations. By incorporating sentiment analysis, the system was able to better tailor recommendations based on the positivity or negativity of user feedback, further refining the relevance of suggested recipes.

5.2. Model Integration and System Deployment with Flask API:

The developed models of collaborative filtering, content-based filtering, and sentiment analysis were integrated into a unified recommendation system. This integration process involved combining the strengths of each model to provide a comprehensive recommendation mechanism.

- **Combining Individual Models:** The collaborative filtering, content-based filtering, and sentiment analysis models were combined to form a unified recommendation system. Each model contributes to the final recommendation, providing a more comprehensive and accurate suggestion list.
- **Weighting and Blending Techniques:** Different weighting and blending techniques were explored to optimize the performance of the hybrid model. By adjusting the contribution of each model, the system was fine-tuned to balance the strengths of collaborative filtering, content-based filtering, and sentiment analysis. This ensured that the final recommendations were well-rounded and aligned with user preferences.

To make the RNN model accessible for real-time predictions and integrate it into the recipe recommendation system, we implemented a Flask API [60]. Flask, a lightweight WSGI web application framework, was used to serve the model and handle HTTP requests for predictions.

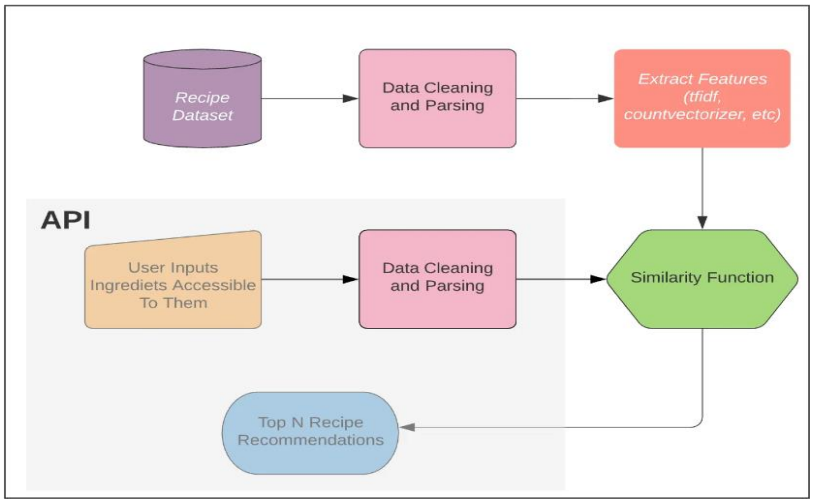


Figure 27: Flow Diagram of RESTful API

Flask API Implementation:

1. Setup and Configuration:

- We set up a Flask application to expose the trained RNN model via a RESTful API.
- The Flask app allows users to send requests to the model and receive predictions in response.

2. API Endpoint: The application is configured to run in development mode, which is suitable for testing and debugging purposes.

URL: http://127.0.0.1:5000

The individual models developed in the previous section were integrated into a hybrid recommendation system. The integration process involved:

```
Collecting flask-ngrok
  Downloading flask_ngrok-0.0.25-py3-none-any.whl (3.1 kB)
Requirement already satisfied: Flask>=0.8 in /usr/local/lib/python3.10/dist-packages (from flask-ngrok) (2.2.5)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-packages (from flask-ngrok) (2.31.0)
Requirement already satisfied: Werkzeug>=2.2.2 in /usr/local/lib/python3.10/dist-packages (from Flask>=0.8->flask-ngrok) (3.0.3)
Requirement already satisfied: Jinja2>=3.0 in /usr/local/lib/python3.10/dist-packages (from Flask>=0.8->flask-ngrok) (3.1.4)
Requirement already satisfied: itsdangerous>=2.0 in /usr/local/lib/python3.10/dist-packages (from Flask>=0.8->flask-ngrok) (2.2.0)
Requirement already satisfied: click>=8.0 in /usr/local/lib/python3.10/dist-packages (from Flask>=0.8->flask-ngrok) (8.1.7)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests->flask-ngrok) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests->flask-ngrok) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests->flask-ngrok) (2.0.7)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests->flask-ngrok) (2024.6.2)
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.10/dist-packages (from Jinja2>=3.0->Flask>=0.8->flask-ngrok) (2.1.5)
Installing collected packages: flask-ngrok
Successfully installed flask-ngrok-0.0.25
 * Serving Flask app '__main__'
 * Debug mode: off
INFO:werkzeug:WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
INFO:werkzeug:Press CTRL+C to quit
```

Figure 28: Output URL of Flask API

3. Running the Flask Application:

- The application is launched using the following above command
- The server runs locally on port 5000 and is accessible via the provided URL. It is important to note that this setup is intended for development and testing. For production use, a robust WSGI server (eGuni corn, uWSGI) should be employed.

4.Testing with Postman:

- **Prediction Endpoint:** Users can POST review data to the API, and the model will return predictions for sentiment classification and star rating regression.To test this we used Postman.
- **Request**: The recipe was fantastic, and the instructions were clear.
- **Response**:


```
{
"predicted_sentiment": "Positive",
"Predicted_sentiment_score": 0.875,
"predicted_star_rating": 4.5
}
```


Figure 29: JSON Output for Designed API in Postman

The integration of the Flask API with the recommendation system demonstrates a practical deployment of the model, providing real-time access to predictions and enabling interactive user experiences. The implementation highlights the model's readiness for deployment and its potential for real-world applications.

5.3. Performance Evaluation and Metrics Analysis:

To assess the effectiveness of the models and the hybrid system, several evaluation metrics were applied:

1. Metrics for Evaluating Model Performance: Key metrics such as precision, recall, and Root Mean Squared Error (RMSE) were used to evaluate the performance of each model. Precision and recall measured the accuracy and completeness of the recommendations, while RMSE provided insight into the accuracy of predicted ratings.

- **Precision**: Measures the accuracy of the recommendations by assessing the proportion of true positive recommendations among all positive recommendations.
- **Recall**: Evaluate the completeness of the recommendations by determining the proportion of true positive recommendations among all possible relevant recommendations.
- **Root Mean Squared Error (RMSE)**: Assesses the accuracy of the predicted ratings by calculating the square root of the average squared differences between predicted and actual ratings.
- **Categorical Cross-Entropy**: Used for the sentiment classification task to measure the performance of the model in predicting the correct sentiment class.
- **Mean Squared Error (MSE)**: Employed for the star rating regression task to quantify the accuracy of the predicted star ratings compared to actual ratings.

2. Comparison of Models Using Evaluation Metrics: The Performance of individual models was compared based on the evaluation metrics to determine their strengths and weaknesses. This comparison informed decisions on how to weigh and blend the models in the hybrid system, ensuring optimal performance.

- **Collaborative Filtering**: Evaluated using precision, recall, and RMSE to gauge its effectiveness in providing relevant recommendations based on user interactions and preferences.
- **Content-Based Filtering**: Assessed using similar metrics to determine its accuracy in recommending items based on item features and user preferences.
- **Sentiment Analysis**: The RNN model was evaluated using categorical cross-entropy for sentiment classification and MSE for star rating prediction. This helped in understanding the model's ability to accurately classify sentiments and predict ratings.

3. Performance Results:

Sentiment Classification:

- ***RNN Model:*** Achieved a classification accuracy of 90.98%. This indicates strong performance in understanding and categorizing the sentiment of user reviews into positive, neutral, or negative categories.

The RNN model's accuracy was compared to Logistic Regression (91.61%), showing similar performance. Although Logistic Regression performed slightly better, the deep

learning model's performance is competitive and demonstrates its capability in handling sentiment analysis.

Star Rating Regression:

- ***RNN Model:*** Obtained a Mean Squared Error (MSE) of 0.0254 for star rating prediction, indicating high accuracy in predicting star ratings with minimal error.

The RNN model outperformed traditional models, such as Support Vector Regression (SVR) with an MSE of 0.0498 and Random Forest Regressor with an MSE of 0.0424, showcasing its superior performance in star rating prediction.

5.4. Project Review and Evaluation of the Hybrid System's Effectiveness and Limitations:

The evaluation of the hybrid recommendation system highlighted several key findings they are:

- ***Effectiveness of Hybrid Approach:*** The integration of collaborative filtering, content-based filtering, and sentiment analysis models resulted in a recommendation system that effectively balanced multiple sources of information. This hybrid approach provided a more comprehensive and accurate recommendation experience compared to single-model systems.
- ***Model Performance:*** The RNN model's high accuracy in sentiment classification and star rating prediction underscored the advantages of deep learning approaches in handling complex tasks. The model's performance was superior to traditional methods, demonstrating its capability to capture nuanced user sentiments and predict ratings effectively.
- ***Impact on User Experience:*** The system's ability to deliver relevant recommendations based on a combination of user preferences, item features, and sentiment analysis enhanced the overall user experience. The integration of sentiment analysis allowed for a more nuanced understanding of user feedback, improving the relevance of recommendations.
- **Enhanced Personalization:** By combining collaborative filtering and content-based filtering, the system effectively personalized recommendations based on user interactions, preferences, and item features.
- ***Accurate Sentiment Analysis:*** The RNN model excelled in sentiment classification and star rating prediction, contributing significantly to the system's ability to understand user feedback and predict ratings with high accuracy.
- **Comprehensive Recommendations:** The hybrid approach ensured that recommendations were not only based on user preferences and item features but also aligned with sentiment analysis, offering a more nuanced and comprehensive recommendation mechanism.

Limitations:

- ***Complexity in Integration:*** Combining multiple models increased the complexity of the system, requiring careful balancing and tuning of weights and blending techniques to achieve optimal performance.
- ***Development Mode Limitations:*** The Flask API, while functional for development and testing, may face scalability and performance issues in a production environment. Transitioning to a production-grade server and optimizing API performance will be essential for real-world deployment.
- ***Scalability:*** Implementing robust scaling strategies and optimizing the hybrid system's performance will be crucial for handling larger datasets and increased user interactions.
- ***Model Fine-Tuning:*** Continuous refinement of the models, particularly in handling edge cases and improving prediction accuracy, will enhance the system's overall effectiveness.
- ***Exploring Different Classification Models:*** The study was limited to using only three types of classification models based on existing academic knowledge. Exploring a broader range of classification models could potentially uncover more effective techniques and improve the system's performance.
- ***Users Feedback Integration:*** Incorporating user feedback into the recommendation process can further personalize and improve the relevance of recommendations.

Chapter 6: Conclusion and Recommendations for Future Work

6.1. Conclusion:

This recipe recommendation system offers practical benefits for real-world applications, especially for improving personalized recommendations on food-related platforms. By using deep learning models like Recurrent Neural Networks (RNNs) to analyze user reviews, the system can better understand the opinions expressed and make more accurate suggestions based on user preferences. This helps businesses provide recommendations tailored to individual tastes, dietary preferences, and feedback, improving user satisfaction.

Power BI played an important role in this project by making the data easy to understand through visualizations. It helped display insights like how users feel about different recipes, what ratings they give, and how well the models performed. These visuals made it simple for both technical teams and business stakeholders to see the results and make decisions about how to improve the platform.

The development of the recipe recommendation system incorporating sentiment analysis and star rating prediction demonstrated the potential of deep learning models, specifically Recurrent Neural Networks (RNNs), in understanding user preferences and enhancing recommendation quality. By processing user reviews through an LSTM model and integrating numerical features, the system could classify sentiments and predict star ratings with high accuracy.

The RNN achieved a classification accuracy of 90.98% for sentiment analysis, showing its ability to capture the contextual meaning of reviews. Additionally, the Mean Squared Error (MSE) of 0.0254 for star rating prediction highlights its effectiveness in estimating user ratings. The classical machine learning models (Logistic Regression, SVR, and Random Forest Regressor) performed slightly lower, particularly in star rating predictions, where deep learning models have a distinct advantage due to their capacity to handle large volumes of sequential text data.

Moreover, the development of a RESTFUL API for this project allows the system to be easily integrated into websites or apps. This means that users can receive personalized recipe recommendations in real time after submitting reviews or interacting with the platform. The API also makes the system scalable, so it can be used in various applications like food delivery services or health apps. In summary, this project combines deep learning, easy-to-understand visuals, and practical tools like APIs to create a system that can be useful for both businesses and users in the food industry.

The system was designed with several key considerations in mind. It prioritizes user-centric design, ensuring that recommendations are closely aligned with user emotions and feedback. The system is adaptable to changing preferences, continuously learning from new interactions to keep recommendations relevant. From a business perspective, it enhances engagement and customer retention by offering personalized recommendations, providing a competitive advantage in the food industry. The automation of the recommendation process and the use of AI for sentiment analysis improve operational efficiency, reducing manual curation efforts. The scalability and

flexibility of the API allow for easy integration into various platforms and accommodate growing data volumes. Looking ahead, there is potential for future enhancements, such as incorporating features like dietary restrictions, cuisine preferences, or seasonal recipes, which could further personalize the system and broaden its applicability. Additionally, the system addresses ethical considerations by anonymizing user data, balancing the need for personalization with the responsibility to protect user privacy.

In this project, we successfully combine advanced deep learning techniques, user-friendly visualizations, and scalable tools like APIs, making it a powerful solution for both businesses and consumers in the food industry.

6.2. Recommendations for Future Work:

Throughout the development and evaluation of the recipe recommendation system in this MSc project, we successfully built and tested several predictive models to identify the most effective approach for personalized recommendations. Despite achieving promising results with the selected models, the analysis was constrained by the limited set of features available. To enhance the system's performance and accuracy, it is necessary to expand and refine the dataset used.

The analysis has demonstrated that only a subset of features such as user ratings and review sentiments significantly impacted the prediction outcomes. This finding underscores the need for additional measures that could potentially offer a deeper understanding of user preferences and improve the recommendation quality. For instance, integrating features related to dietary restrictions, user demographics, or trending food preferences could provide a more comprehensive understanding of user needs and enhance the system's recommendations.

Based on these observations, several areas for future development and research have been identified. These include incorporating new data sources, exploring advanced techniques, and addressing various aspects of user interaction and system performance. Below are detailed recommendations for future work that aim to build upon the existing foundation and further advance the recipe recommendation system.

* *Incorporation of Dietary Restrictions and Preferences:* Future enhancements could include integrating dietary restrictions and specific cuisine preferences. By considering individual dietary needs, such as vegan or gluten-free options, the system could offer even more personalized recommendations. This adjustment would cater to users' unique dietary requirements, increasing satisfaction and engagement by aligning suggestions with their personal needs.
* *Seasonal and Trending Recipes:* Integrating seasonal and trending recipes could make the recommendations timelier and more relevant. By accounting for factors such as seasonal ingredients and current food trends, the recommendation engine can offer suggestions that remain fresh and appealing throughout the year, enhancing the overall user experience.
* *Enhanced Sentiment Analysis with Fine-Tuned Models:* Improving sentiment analysis accuracy could be achieved by utilizing advanced natural language processing models, such as BERT or GPT. These models can capture more nuanced sentiments and contextual meanings in user reviews. Fine-tuning these models on datasets specific to recipes and

reviews could significantly enhance the system's ability to interpret user feedback more accurately.

* *Time Constraints and Computational Efficiency:* During this project, it was noted that training Recurrent Neural Networks (RNNs) and performing Synthetic Minority Over-sampling Technique (SMOTE) on the dataset was computationally intensive, often taking up to an hour. This time constraint highlights the need for further optimization of these processes. Future work should focus on improving computational efficiency, perhaps by exploring more efficient algorithms, leveraging hardware acceleration, or employing model optimization techniques to reduce training times and enhance overall system performance.
* *Integration of Recipe Nutritional Information:* Adding nutritional information to the recommendation system could help users make healthier choices. By analyzing the nutritional content of recipes, the system could recommend options that align with specific dietary goals or health preferences, providing an extra layer of personalization and assisting users in making informed dietary decisions.
* *Expansion to Multi-Language Support:* Expanding the system to support multiple languages would broaden its accessibility. Training models on diverse linguistic data could ensure accurate sentiment analysis and recommendations across different languages and cultural contexts, making the system more inclusive and catering to a wider audience.
* *Exploration of Hybrid Recommendation Techniques:* Implementing advanced hybrid recommendation techniques could refine the system's accuracy and effectiveness. Combining collaborative filtering with content-based filtering could leverage both user preferences and content attributes, significantly improving the quality of the recommendations.
* *Investigation of User Engagement Metrics:* Future research should explore additional user engagement metrics to evaluate the effectiveness of recommendations. Metrics such as click-through rates, time spent on recommended recipes, and user feedback can provide valuable insights into the system's performance. Analyzing these metrics will help identify areas for improvement and optimize the recommendation process.
* *Development of Real-Time Adaptation Algorithms:* Investigating real-time adaptation algorithms could enhance the system's responsiveness by dynamically adjusting recommendations based on immediate user interactions and feedback. This capability would enable the system to offer more relevant suggestions as user preferences evolve, improving the overall user experience.
* *Ethical Implications and Privacy Considerations:* Ongoing research into the ethical implications and privacy considerations of recommendation systems is crucial. Ensuring responsible handling of user data, maintaining transparency in data usage, and addressing potential biases in recommendations are essential for balancing personalization with the protection of user information.
* *User Experience and Interface Design:* Exploring improvements in user experience (UX) and interface design can enhance user interactions with the recommendation system. Conducting usability studies and gathering user feedback will lead to more intuitive and

engaging interfaces, ultimately improving user satisfaction and making the system more effective in meeting their needs.

6.3. Thesis Contribution:

This thesis makes several important contributions:

* *Combining Advanced Tools and Techniques:* This MSc project brings together various statistical methods, data mining techniques, and machine learning approaches to build an effective recipe recommendation system. By using advanced models like Recurrent Neural Networks (RNNs) along with traditional methods, the research creates a strong system for offering personalized recipe suggestions.
* *Innovative Use of Sentiment Analysis:* The incorporation of sentiment analysis using advanced natural language processing techniques, including Long Short-Term Memory (LSTM) networks, has significantly improved the system's ability to interpret user reviews. The use of sentiment analysis enables the system to better understand user opinions and preferences, leading to more tailored recommendations. The high classification accuracy achieved (90.98%) underscores the effectiveness of this approach in capturing the nuanced sentiments expressed in reviews.
* *Real-Time API Development:* The development of a RESTful API for the recommendation system is a key contribution. This API facilitates the seamless integration of the recommendation system into various applications, such as websites and mobile apps. It supports real-time recommendation updates based on user interactions and feedback, making the system adaptable to dynamic user preferences and enhancing its usability in practical scenarios.
* *Using Visuals to Understand Data:* The thesis uses charts and graphs to make data easier to understand. Tools like Power BI are used to turn complex information into clear visualizations, which help in spotting important patterns and making better decisions. These visuals make it easier for everyone to see and understand the results.
* *Improving User Engagement*: This work shows how data analysis can be used to keep users more engaged with the platform. By customizing recommendations based on what users like and their feedback, the system aims to make users happier and more likely to continue using the platform.
* *Identification of Computational Challenges:* In this project, we address the computational challenges encountered during model training, particularly with RNNs and the SMOTE algorithm. By documenting the time constraints and proposing future work on improving computational efficiency, the thesis contributes to the ongoing development of more scalable and efficient recommendation systems.

References

[1] D. W. M. Jie Lu, "Decision Support Systems," *Recommender system application developments: A survey,* vol. 74, no. Decision Support Systems, pp. 12-32, 2015.

[2] NVIDIA., "Recommendation System," nvidia., [Online]. Available: https://www.nvidia.com/en-us/glossary/recommendation-system/. [Accessed 16 July 2024].

[3] S. S. S. Mengqi Liao, "When E-Commerce Personalization Systems Show and Tell," *Investigating the Relative Persuasive Appeal of Content-Based versus Collaborative Filtering,* vol. 2, no. Journal of Advertising 51 , p. 267, 2021.

[4] M. &. N. M. &. V. M. Vivek, "Machine Learning Based Food Recipe Recommendation System," in *ResearchGate*, Mysore, 2023.

[5] M. L. G. Vaibhav Kumar, "Predictive Analytics: A Review of Trends and Techniques," vol. 182, no. International Journal of Computer Applications (0975 – 8887), pp. 31-37, 2018.

[6] S. E. F. Matthew A. Waller, "Data Science, Predictive Analytics, and Big Data," *A Revolution That Will Transform Supply Chain Design and Management,* vol. 34(2), no. Journal of Business logistics, pp. 77-84, 2013.

[7] B. Marr, "Why Data Is The Lifeblood Of Modern Organizations," Forbes, 2022. [Online]. Available: https://www.forbes.com/sites/bernardmarr/2022/06/29/why-data-is-the-lifeblood-of-modern-organizations/.

[8] L. Y. F. A. L. L. H. T. X. A. L. J. Wu WT, "Data mining in clinical big data," *the frequently used databases, steps, and methodological models,* vol. 8, no. National Library Of Medicine, 2021.

[9] D. Rondeau, "AI's Role in Enhancing Predictive Analytics," Rocketfarmstudios, 2024. [Online]. Available: https://www.rocketfarmstudios.com/blog/ais-role-in-enhancing-predictive-analytics/.

[10] Y. B. A. C. Goodfellow lan, Deep Learning, London: Cambridge,England, 2016.

[11] A. Ng, Machine Learning Yearning: Technical Strategy for AI Engineers, In the Era of Deep Learning, Self-publishing, 2018.

[12] P. D. Y. T. Kamal Taha, "Empirical and Experimental Perspectives on Big Data in Recommendation Systems," *Taha, Kamal, Paul D. Yoo, Chan Yeun, and Aya Taha. "Empirical and Experimental PeBig Data Mining and Analytics 7, no. 3 (2024): 964-1014.,* vol. 7, no. 3, pp. 964-1014, 2024.

[13] M. M. T. C. a. B. F. Afsar, "Reinforcement learning based recommender systems: A survey," *ACM Computing Survey,* vol. 55, no. 7, pp. 1-38, 2022.

[14] F. O. Isinkaye, "Matrix Factorization in Recommender Systems:Algorithms, Applications, and Peculiar Challenges," *IETE Journal of Research ,* vol. 69, no. 9, 2023.

[15] S. S. B. P. S. S. P. V. T. F. A. V. R. A. P. Sengupta, "A review of deep learning with special emphasis on architectures, applications and recent trends," *Knowledge-Based Systems,* vol. 194, no. 105596, 2020.

[16] A. S. N. Da'u, "Recommendation system based on deep learning methods," *Artificial Intelligence Review 53.4,* vol. 53, 2019.

[17] M. M. K. A. B.-H. Birjali, "A comprehensive survey on sentiment analysis: Approaches, challenges and trends," *Knowledge-Based Systems,* vol. 226, no. 107134, 2021.

[18] F. M. K. S. Hemmatian, "A survey on classification techniques for opinion mining and sentiment analysis," *Artificial intelligence review,* vol. 52, no. 3, pp. 1495-1545, 2019.

[19] T. a. A. A. Abdullah, "Deep learning in sentiment analysis: Recent architectures," *ACM Computing Surveys,* vol. 55, no. 8, pp. 1-37, 2022.

[20] Z. A. F. X. S. X. W. Gao, "Target-Dependent Sentiment Classification," *Ieee Access 7 ,* vol. 12, 2019.

[21] H. S. L. Y. P. A. C. Ko, "A Survey of Recommendation Systems: Recommendation Models, Techniques, and Application Fields," *Electronics 11.1,* vol. 11, no. 1, 2022.

[22] M. S. S. S. J. B. W. Liao, "User Trust in Recommendation Systems: A comparison of Content-Based, Collaborative and Demographic Filtering," *In Proceedings of the 2022 CHI conference on human factors in computing systems,* pp. 1-14, 2022.

[23] Y. M. L. M. A. A. Afoudi, "Hybrid recommendation system combined content-based filtering and collaborative prediction using artificial neural network," *Simulation Modelling Practice and Theory ,* vol. 113, no. 102375, 2021.

[24] A. A. Z. M. B. M. Huang Huang, "Sentiment Analysis in E-Commerce Platforms: A Review of Current Techniques and Future Directions," *IEEE Access 11,* vol. 11, 2023.

[25] S. R. a. S. S. S. Sukhdeve, "A Crash Course on Big Data, Machine Learning, and Data Analytics Services," *Google Cloud Platform for Data Science,* vol. SpringerLink, 2023.

[26] S. V. M. Raschka, Python machine learning: Machine learning and deep learning with Python, scikit-learn, and TensorFlow, Birmingham: Packt publishing ltd, 2018.

[27] L. S. W. K. K. L. Yu, "An integrated data preparation scheme for neural network data analysis," *IEEE Transactions on Knowledge and Data Engineering,* vol. 18, no. 2, pp. 217-230, 2005.

[28] GeeksForGeeks, "Life Cycle Phases of Data Analytics," geeksforgeeks.org, 2024. [Online]. Available: https://www.geeksforgeeks.org/life-cycle-phases-of-data-analytics/.

[29] G. Duta, "Mastering MLOps in 2024: A Comprehensive Guide to MLOps, Its Integration, and Qwak's Unified Platform," MLOps Platform, 2024. [Online]. Available: https://www.qwak.com/post/the-ultimate-guide-to-mlops-tools-in-2024#:~:text=A/B%20testing.-

,Model%20Monitoring:%20Ensuring%20Continuous%20Performance,patterns%20or%20the%20operational%20environment..

[30] A. M. S. &. C. J. Ali, "Recipe Reviews and User Feedback," UC Irvine Machine Learning Repository, 2023. [Online]. Available: https://doi.org/10.24432/C5FG95.

[31] r. devi, "How to handle Missing values?," Medium, 2020. [Online]. Available: https://medium.com/analytics-vidhya/how-to-handle-missing-values-cbd03fb79ef8.

[32] A. Bhandari, "Feature Scaling: Engineering, Normalization, and Standardization," analyticsvidhya.com, 2024. [Online]. Available: https://www.analyticsvidhya.com/blog/2020/04/feature-scaling-machine-learning-normalization-standardization/.

[33] McKinsey, "Data preprocessing vs. feature engineering," iguazio, [Online]. Available: https://www.iguazio.com/questions/data-preprocessing-vs-feature-engineering-whats-the-difference/. [Accessed 2024].

[34] D. M. D. E. J. A. K. Kluver, "Rating-Based Collaborative Filtering: Algorithms and Evaluation," *Social information access: Systems and technologies,* p. 344–390, 2018.

[35] E. K. Jacob Murel Ph.D., "What is content-based filtering?," IBM, 2024. [Online]. Available: https://www.ibm.com/topics/content-based-filtering.

[36] Google Developers, "Recommendation Systems Content-based filtering," developers.google.com, 2024. [Online]. Available: https://developers.google.com/machine-learning/recommendation/content-based/basics.

[37] M. M. Z. A. A. G. V. H. J. J. M. J. M. R. T. Jalali, "A Hybrid Hierarchical Mathematical Heuristic Solution of Sparse Algebraic Equations in Sentiment Analysis," *Information,* vol. 15, no. 9, 2024.

[38] Ubaid, "Mean, Mode and Median – Measures of Central Tendency," Analyticsvidhya, 2023. [Online]. Available: https://www.analyticsvidhya.com/blog/2021/04/3-central-tendency-measures-mean-mode-median/.

[39] H. Patel, "Feature Engineering Explained," *Brennan Whitfield ,* Apr 29, 2024.

[40] Edge Delta , "What is Data Transformation? Importance and Best Practices," *Data Transformation Overview,* 2024.

[41] A. Bhandari, "Feature Scaling: Engineering, Normalization, and Standardization," Analyticsvidhya, 2024. [Online]. Available: https://www.analyticsvidhya.com/blog/2020/04/feature-scaling-machine-learning-normalization-standardization/.

[42] Data Headhunters academy, "Advanced Feature Engineering: Techniques for Predictive Accuracy," Data Headhunters, 2024. [Online]. Available: https://dataheadhunters.com/academy/advanced-feature-engineering-techniques-for-predictive-accuracy/.

[43] A. S. Gillis, "data splitting," TechTarget, 2024. [Online]. Available: https://www.techtarget.com/searchenterpriseai/definition/data-splitting.

[44] O. K. Afshin Gholamy, "Why 70/30 or 80/20 Relation Between Training and Testing Sets: A Pedagogical Explanation," Technical Report: UTEP-CS-18-09, 2018.

[45] D. Mesquita, "Python AI: How to Build a Neural Network & Make Predictions," Realpython, 2021. [Online]. Available: https://realpython.com/python-ai-neural-network/.

[46] S. Poudel, "Recurrent Neural Network (RNN) Architecture Explained," Medium, 2023. [Online]. Available: https://medium.com/@poudelsushmita878/recurrent-neural-network-rnn-architecture-explained-1d69560541ef.

[47] ,. S. K. S. B. C. Shiv Ram Dubey, "Activation Functions in Deep Learning: A Comprehensive Survey and Benchmark," Elsevier, 2021.

[48] D. Wei, "Demystifying the Adam Optimizer in Machine Learning," Medium, 2024. [Online]. Available: https://medium.com/@weidagang/demystifying-the-adam-optimizer-in-machine-learning-4401d162cb9e.

[49] Tavishi, "Part-1 Evaluation metrics for Regression problems in Machine Learning," Medium, 2023. [Online]. Available: https://medium.com/@tavishi.1402/part-1-performance-metrics-in-machine-learning-27cd9264e398.

[50] X. G. e. al, "Hierarchical Attention Network for Visually-Aware Food Recommendation," *IEEE Transactions on Multimedia,* vol. 22, 2020.

[51] N. T. B. Lauri Salmela, Predicting ultrafast nonlinear dynamics in fibre optics with a recurrent neural network, ResearchGate, 2021.

[52] K. E. B. Ö. B. C. E. Jon Nicolas Bondevik, "A systematic review on food recommender systems,," *Expert Systems with Applications,* vol. 238, no. 122166, 2024.

[53] Vatsal, "Recommendation Systems Explained," towardsdatascience, 2021. [Online]. Available: https://towardsdatascience.com/recommendation-systems-explained-a42fc60591ed.

[54] S. M. M. I. A. Sankari, "Exploring Matrix Decomposition Methods for Recommender Systems," *JOURNAL OF SCIENTIFIC RESEARCH,* vol. 16, no. 3, 2024.

[55] S. V. B. T. H. a. S. B. Chhipa, "Recipe recommendation system using TF-IDF," *ITM web of conferences,* vol. 44, 2022.

[56] H. B. P. Krupa Patel, "A state-of-the-art survey on recommendation system and prospective extensions," *Computers and Electronics in Agriculture,* vol. 178, 2020.

[57] S. a. A. K. Kazhuparambil, "Cooking is all about people: Comment classification on cookery channels using Bert and classification models," *arXiv preprint,* 2020.

[58] M. S. Jim Holdsworth, "What is deep learning," *IBM,* 2024.

[59] nbia.ca, "Brain Structure And Function," Northern Brain Injury Association, 2024. [Online]. Available: https://www.nbia.ca/brain-structure-function/.

[60] J. Leitch, "Building a Recipe Recommendation API using Scikit-Learn, NLTK, Docker, Flask, and Heroku," Towards Data Science, 2020. [Online]. Available: https://jackmleitch.medium.com/list/recipe-recommendation-system-f9070406eb41.

Appendix:

The codes used in this Project Report is available on Google Colab:[https://colab.research.google.com/drive/1eqELyFjnSUurT55SY4RV2H5x1JLow2bK?usp=sharing].

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVR
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import accuracy_score, mean_squared_error
from tensorflow.keras.models import Model
from tensorflow.keras.layers import Input, Dense, Dropout, LSTM, Embedding, Bidirectional, concatenate
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.callbacks import EarlyStopping
from tensorflow.keras.optimizers import Adam
from sklearn.feature_extraction.text import TfidfVectorizer

# Load and prepare your dataset
# data = pd.read_csv('/content/data_with_sentiment.csv')

# Ensure that all text data is string type and replace NaN values
data['processed_review_text'] = data['processed_review_text'].astype(str).fillna('')

# Preprocessing
max_words = 10000  # Maximum number of words to consider in the tokenizer
max_len = 100  # Maximum length of sequences

# Tokenize and pad sequences
tokenizer = Tokenizer(num_words=max_words)
tokenizer.fit_on_texts(data['processed_review_text'])
X_text_seq = tokenizer.texts_to_sequences(data['processed_review_text'])
X_text_pad = pad_sequences(X_text_seq, maxlen=max_len)

# Select numerical features
X_numerical = data[['user_reputation', 'thumbs_up', 'thumbs_down', 'star_rating', 'best_score']].values

# Encode target variable (Sentiment)
```

Figure 30: Python Code for RNN Model

```python
# Train and evaluate Support Vector Regression (regression task)
svr = SVR(kernel='rbf')
svr.fit(X_train_reg, y_train_reg)
svr_predictions = svr.predict(X_test_reg)
svr_mse = mean_squared_error(y_test_reg, svr_predictions)
print(f"Support Vector Regression Mean Squared Error: {svr_mse:.4f}")

# Train and evaluate Random Forest Regressor (regression task)
rf_regressor = RandomForestRegressor(n_estimators=100, random_state=42)
rf_regressor.fit(X_train_reg, y_train_reg)
rf_predictions = rf_regressor.predict(X_test_reg)
rf_mse = mean_squared_error(y_test_reg, rf_predictions)
print(f"Random Forest Regressor Mean Squared Error: {rf_mse:.4f}")
```

```
Epoch 1/10
364/364 ━━━━━━━━━━━━━━━━━━━━ 115s 276ms/step - classification_output_accuracy: 0.5666 - loss: 2.2502 - regression_output_mea
Epoch 2/10
364/364 ━━━━━━━━━━━━━━━━━━━━ 139s 269ms/step - classification_output_accuracy: 0.8682 - loss: 1.2737 - regression_output_mea
Epoch 3/10
364/364 ━━━━━━━━━━━━━━━━━━━━ 101s 279ms/step - classification_output_accuracy: 0.8692 - loss: 1.0075 - regression_output_mea
Epoch 4/10
364/364 ━━━━━━━━━━━━━━━━━━━━ 98s 270ms/step - classification_output_accuracy: 0.8708 - loss: 0.8616 - regression_output_mean
Epoch 5/10
364/364 ━━━━━━━━━━━━━━━━━━━━ 138s 259ms/step - classification_output_accuracy: 0.8718 - loss: 0.6911 - regression_output_mea
Epoch 6/10
364/364 ━━━━━━━━━━━━━━━━━━━━ 145s 268ms/step - classification_output_accuracy: 0.8764 - loss: 0.5617 - regression_output_mea
Epoch 7/10
364/364 ━━━━━━━━━━━━━━━━━━━━ 142s 267ms/step - classification_output_accuracy: 0.9039 - loss: 0.4521 - regression_output_mea
Epoch 8/10
364/364 ━━━━━━━━━━━━━━━━━━━━ 143s 270ms/step - classification_output_accuracy: 0.9144 - loss: 0.4021 - regression_output_mea
Epoch 9/10
364/364 ━━━━━━━━━━━━━━━━━━━━ 141s 267ms/step - classification_output_accuracy: 0.9276 - loss: 0.3364 - regression_output_mea
Epoch 10/10
364/364 ━━━━━━━━━━━━━━━━━━━━ 143s 271ms/step - classification_output_accuracy: 0.9399 - loss: 0.3130 - regression_output_mea
114/114 ━━━━━━━━━━━━━━━━━━━━ 9s 81ms/step - classification_output_accuracy: 0.9111 - loss: 0.2514 - regression_output_mean_s
Neural Network Test Loss: 0.2550
Neural Network Classification Accuracy: 0.9098
Neural Network Regression MSE: 0.0254
```

Figure 31: Output for RNN model Prediction