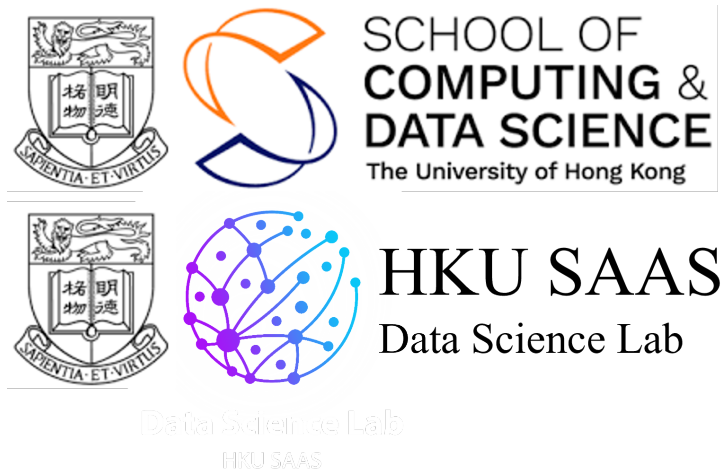




# RetailLLM: A Context-Aware LLM Agent for Personalized Retail Support

Luo Dongyu, Fan Zhenyi, Hui Lok Samuel, Zhu Yecheng  
Group 4

**Supervisor:** Dr. LAU, Adela Sau Mui



# Contents

<b>1</b>	<b>Introduction</b>	<b>5</b>
<b>2</b>	<b>Related Work</b>	<b>9</b>
2.1	Optimization of LLM Agents . . . . .	9
2.2	Advances in Retail Data Analysis . . . . .	10
2.3	Applications of LLMs in Retail . . . . .	10
<b>3</b>	<b>Methodology</b>	<b>11</b>
3.1	RetailLLM Architecture . . . . .	11
3.1.1	Auxiliary Classes . . . . .	12
3.1.2	Communication with modules . . . . .	13
3.2	Kernel LLM Module: Deployment and Fine-tuning . . . . .	14
3.2.1	Deployment of LLM Kernel through VLLM . . . . .	15
3.2.2	Client Access to Kernel LLM . . . . .	15
3.2.3	Fine-tuning of Llama 3.2 Models: 1B-Instruct and 3B-Instruct . . . . .	16
3.3	RetailLLM Analysis . . . . .	17
3.3.1	Correlation Analysis . . . . .	18
3.3.2	Regression Analysis . . . . .	20
3.3.3	Cluster Analysis . . . . .	24
3.3.4	Time Series Analysis . . . . .	26
3.3.5	Survival Analysis . . . . .	28
3.3.6	Summary of Analysis . . . . .	30
3.4	Database . . . . .	30

3.4.1	Framework Selection and Strategic Rationale . . . . .	31
3.4.2	Data Ecosystem and Structural Design . . . . .	32
3.4.3	Advanced Query Request and Search Architecture . . . . .	32
3.4.4	Technical Implementation . . . . .	33
3.5	Web Scraper . . . . .	33
3.5.1	Introduction and Motivation . . . . .	33
3.5.2	System Architecture and Advantages . . . . .	34
3.5.3	Analytical Techniques . . . . .	34
3.6	Web UI . . . . .	35
3.6.1	Components . . . . .	35
3.6.2	Architecture . . . . .	35
3.6.3	Example from calling the API . . . . .	37
<b>4</b>	<b>Result</b>	<b>39</b>
4.1	Comparison of System Capabilities . . . . .	39
4.2	Kernel LLM deployment and fine-tuning results . . . . .	40
4.2.1	Inference speed: VLLM vs Transformers . . . . .	40
4.2.2	Model's accuracy Testing . . . . .	40
4.3	Evaluation of Web Scraping Configurations . . . . .	41
4.4	Visualization Result . . . . .	42
<b>5</b>	<b>Discussion</b>	<b>43</b>
5.1	Key Contributions and Strengths . . . . .	44
5.2	Limitations . . . . .	45

<b>6</b>	<b>Conclusions and Future Work</b>	<b>45</b>
<b>A</b>	<b>Appendix Overview</b>	<b>52</b>
<b>B</b>	<b>LLM Deployment and Access Details</b>	<b>52</b>
B.1	VLLM Command for Deployment . . . . .	52
B.2	Client-side Python code . . . . .	53
<b>C</b>	<b>Prompts</b>	<b>54</b>
C.1	Synthetic Data Generation . . . . .	54
C.2	RetaiLLM Prompts . . . . .	55
C.2.1	Determine Get Data Methods . . . . .	55
C.2.2	Prompts for Extract Data from the User Input . . . . .	56
C.2.3	Prompts for Handle Previous Conversation Data . . . . .	57
C.2.4	Prompts for Webscraping . . . . .	58
C.2.5	Prompts for DB query . . . . .	58
C.2.6	Prompts for Insights Generation . . . . .	59
C.2.7	Prompts for Data analysis . . . . .	59
<b>D</b>	<b>DBSCAN Clustering Pseudocode</b>	<b>63</b>
<b>E</b>	<b>Transformer Model Architecture</b>	<b>64</b>
E.1	Overview . . . . .	64
E.2	Components . . . . .	64
E.3	Training Process . . . . .	64
E.4	Advantages . . . . .	65

E.5	Applications in This Study . . . . .	65
E.5.1	Transformer-Based Regression . . . . .	65
E.5.2	Transformer-Based Time Series Model . . . . .	65
<b>F</b>	<b>GitHub Repository for RetailLM</b>	<b>66</b>
<b>G</b>	<b>Additional RetailLM Conversation Examples</b>	<b>66</b>

# Abstract

In the context of rapid global economic development, the retail industry plays a critical role in connecting producers and consumers, and is increasingly transitioning towards digitalization and intelligence. However, existing Large Language Models (LLMs) have yet to fully meet the specific needs and complexities of the retail sector. This paper introduces RetailLLM, a pioneering LLM designed and optimized specifically for the retail industry. RetailLLM integrates retail-specific data and business requirements to offer advanced natural language processing and data analysis capabilities tailored to the unique challenges of the retail domain. The model supports diverse data inputs, including proprietary retail databases, web-scraped data, and user-provided information, ensuring flexible and accurate analysis. RetailLLM features robust multi-turn dialogue capabilities, enabling dynamic information retrieval and integration across multiple interactions. Additionally, it autonomously utilizes relevant databases and knowledge bases to enhance decision-making and operational optimization. The model is equipped with sophisticated analytical methods, including correlation analysis, clustering, regression, time series forecasting, and survival analysis, alongside deep learning-based techniques like transformers. Furthermore, RetailLLM adapts its data analysis approach based on the context of user inputs and provides insights through both textual descriptions and visualizations. A key feature is its web scraping and sentiment analysis capabilities, which enable retailers to assess market sentiment and make informed investment decisions based on real-time online data. RetailLLM sets a new benchmark for the application of LLMs in the retail sector, facilitating more efficient and intelligent business operations. Code is available at: <https://github.com/IILKA/RetailLLM/>.

## 1 Introduction

In the context of rapid global economic development, the retail industry serves as a crucial bridge connecting producers and consumers (Har, Rashid, Te Chuan, Sen, & Xia, 2022; Pantano, 2014), consistently occupying a central position in economic activities. With the diversification of consumer demands and the intensification of market competition, retail enterprises face unprecedented challenges and opportunities (Vaja, 2015). Traditional retail models are transitioning towards digitalization and intelligence, leveraging big data analytics and artificial intelligence technologies to more accurately discern market trends, optimize supply chain management, and enhance customer experience, thereby maintaining competitive advantages in a fiercely competitive market (Koh, Kim, & Kim, 2006;

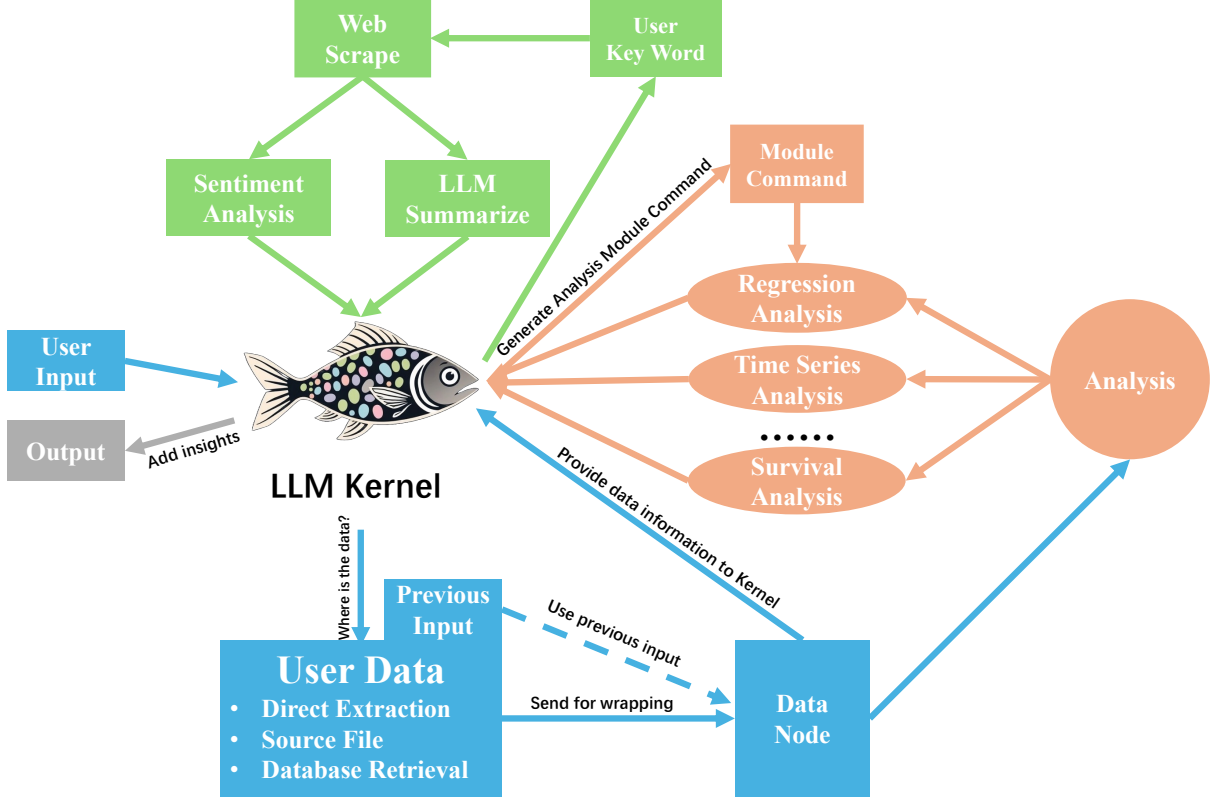


Figure 1: This flowchart represents a structured pipeline for RetaiLLM. Step 1, the system determine the method required to get data by the user’s input. Step 2, the system get data according to the determined get data approach, and send the data to Data Node, where all data in the conversation are managed at. Step 3, the data node conclude the information of the required data, which can be used to help the kernel LLM to determine which data analysis approach should be used. Finally, analysis module takes in the data from data node, and analyses the data according to the analysis command. It’s noteworthy that in step 1, the system can decide whether user needs information from internet. If information form the internet is required, the system will call webscrape module, which scrape the internet by the key words generated according to the user input and then pass for sentiment analysis and summarization.

Sansone & Colamatteo, 2017; Tokatli & Boyaci, 1998). The retail sector encompasses a wide range of data types, including sales data, inventory information, customer behavior data, and marketing data (Hameli, 2018; Yu & Ramanathan, 2008). The effective analysis and utilization of these data are key factors driving innovation and development in the retail industry (Chandramana, 2017; Roorderkerk, DeHoratius, & Musalem, 2022). Concurrently, recent advancements in deep learning technology have led to the widespread application of Large Language Models (LLMs) (Chang et al., 2024; W. X. Zhao et al., 2023). Models such as GPT-4 (Achiam et al., 2023) have demonstrated exceptional capabilities in natural language understanding and generation through training on vast amounts of textual data. These models excel not only in tasks like machine translation, intelligent question answering, and text generation but also exhibit strong adaptability

across various domains (Hadi et al., 2023; Naveed et al., 2023). However, despite the superior performance of existing LLMs in general-purpose applications, their effectiveness in specific industry applications remains limited (Y. Wang et al., 2023; L. Wu et al., 2024; Zhu et al., 2023). In particular, within the retail sector, current models struggle to fully meet the industry’s specific data requirements and business scenarios.

Retail industry data is characterized by high complexity and diversity, encompassing structured data such as sales records and inventory information, as well as unstructured data like customer feedback and social media comments (Bharadwaj, Ballings, Naik, Moore, & Arat, 2022; Sachs et al., 2015). Moreover, retail operations often require real-time responses and multi-turn interactions, posing higher demands on the model’s dynamic learning and contextual understanding capabilities (Seetharaman, Niranjan, Tandon, & Saravanan, 2016; Zvarikova, Michalikova, & Rowland, 2022). While general-purpose LLMs possess robust language processing abilities, they fall short in handling retail-specific data, understanding industry terminology, and supporting complex business processes (Y. Li, Liu, & Yu, 2025; Parida, Ramalingam, & Perumalsamy, 2024; Venkatachalam, Sreeram, & Soundarapandiyan, 2024). Therefore, developing a Large Language Model specifically tailored to the needs of the retail industry not only better serves the specific requirements of retail enterprises but also promotes the deeper integration of LLMs in industry applications.

Against this backdrop, this study introduces **RetaiLLM**, a Large Language Model designed and optimized specifically for the retail industry. RetaiLLM deeply integrates retail-specific data and business requirements to deliver efficient and precise natural language processing and data analysis capabilities. This integration provides intelligent support for decision-making and operational optimization within retail enterprises. Our contributions are highlighted as follows:

1. **The Most Comprehensive LLM Agent Specialized for Retail:** RetaiLLM is the pioneering LLM developed exclusively for the retail sector. Unlike existing general-purpose language models, RetaiLLM undergoes extensive customization in both its architecture and training data, fully addressing the unique characteristics and requirements of the retail industry. The training process incorporates a vast array of retail-related datasets, including sales records, inventory management data, and customer interaction logs, thereby enhancing its performance in handling retail-specific tasks.
2. **Diverse Data Input Capabilities:** RetaiLLM supports multiple types of data inputs, such as proprietary retail databases, web-scraped data, user-provided information from conversations, and direct user inputs. This multimodal data pro-



cessing capability ensures greater flexibility in practical applications, enabling the integration of data from various sources to deliver more comprehensive and accurate analysis results.

3. **Advanced Multi-Turn Dialogue Functionality:** RetailLLM features robust multi-turn dialogue capabilities, allowing for the dynamic extraction and analysis of key information from previous user interactions. Our model can autonomously identify and retrieve relevant data from prior conversations, integrating it seamlessly with new user inputs to perform comprehensive analyses. This functionality enhances the naturalness and coherence of interactions, enabling the model to provide more personalized and precise recommendations by effectively combining historical and real-time data.
4. **Autonomous Utilization of Databases and Knowledge Bases:** RetailLLM can intelligently decide when to leverage pre-built databases and knowledge bases based on the context of user interactions. This autonomous decision-making ensures that the model utilizes the most relevant and reliable data sources to support its responses and analyses.
5. **Comprehensive Multi-Dimensional Data Analysis:** RetailLLM is equipped with a wide range of data analysis methods, including correlation analysis, various clustering techniques, regression analysis, multiple time series forecasting models, and survival analysis. Additionally, some analytical processes incorporate deep learning methods such as transformers, providing robust and sophisticated insights for retail enterprises.
6. **Adaptive Data Analysis Selection and Visualization:** Based on the type of user input data and the context of the conversation, RetailLLM can autonomously select the most appropriate data analysis methods. It then presents the results through both textual descriptions and visualizations, offering users clear and actionable insights. This capability surpasses even the most advanced models like ChatGPT-o1 in terms of specialized analytical functionality.
7. **Web Scraping and Sentiment Analysis:** RetailLLM includes dedicated web scraping functionalities that allow it to gather data from online sources relevant to the retail industry. The scraped data undergoes sentiment analysis to evaluate public opinions and trends, assisting retailers in making informed investment decisions. This capability enables the model to assess market sentiment and predict potential investment opportunities based on real-time online data.

## 2 Related Work

### 2.1 Optimization of LLM Agents

The quest to build more capable large language model (LLM) agents in recent years grown from an interesting possibility into a focal point of serious investigation. Early work, such as (Brown, Mann, Ryder, et al., 2020), demonstrated that fine-tuning large pre-trained models on task-specific data could markedly improve their performance. Yet, these early successes also highlighted a recurring snag: scaling such methods often comes with heavy computational costs and the need for substantial labeled data, placing clear constraints on where and how these models can be deployed (Strubell, Ganesh, & McCallum, 2019).

With these difficulties in mind, a number of researchers have begun gravitating toward approaches that value adaptability and efficiency over brute force. For example, (J. Zhao, Liu, & Li, 2022) experimented with iterative prompt-tuning strategies that aim to refine the model’s instructions on the fly, thereby reducing the engineering overhead and data demands. Along similar lines, (X. Li, Wang, & Chen, 2023) proposed parameter-efficient adapters that allow models to pivot to new tasks without requiring a full architectural overhaul. Though these methods remain a work in progress, they gesture toward a future where LLMs can more gracefully adapt to the shifting needs of specialized domains, including retail (Tay, Dehghani, Bahri, et al., 2022).

Taken together, these various lines of inquiry hint that the field is moving toward more flexible architectures. In the specific context of retail, where customer behavior and product inventories can shift with the season—or even day-to-day—this notion of dynamic adaptation matters a great deal. Building upon prior insights into multi-agent setups (L. Wang, Zhang, & Zhou, 2023; J. Wu, Yang, & Zhang, 2021) and the tension between specialization and generalization (Zhou & Guo, 2022), we sketch out a direction that treats LLM agent teams not as static constructs, but as evolving ensembles that adjust to new tasks or data conditions. This approach, which we detail in our framework, is less about enforcing a one-size-fits-all pipeline and more about enabling LLMs to collaborate, learn from each other’s strengths, and become more responsive to the demands of retail analytics.

## 2.2 Advances in Retail Data Analysis

Over the last decade, retail analytics has leaned heavily on proven methods—such as clustering and regression—to make sense of sales patterns, inventory levels, and customer segments (Aggarwal, 2015; Kohavi, Rothleder, & Simoudis, 2002). These techniques, while foundational, were developed in a landscape where retail environments were simpler and updates came at a gentler pace. As today’s retailers juggle complex supply chains, omnichannel sales platforms, and highly variant consumer preferences, these classic tools can strain under the weight of high-dimensional and rapidly changing data (Fan, Chen, & Lin, 2006).

The rise of machine learning offered a lifeline. Deep learning-based forecasting models, for instance, have helped improve inventory predictions and adjust to fluctuating demand, as shown by (Chen, Li, & Wu, 2019). More recently, graph-based algorithms have helped model interconnections between products, customers, and markets, leading to subtler insights and more targeted recommendations (Gao et al., 2023). Yet for all these gains, there is a clear gap: most existing techniques focus heavily on structured inputs, leaving a wealth of unstructured retail data—customer reviews, product descriptions, and even external commentary from social media—largely untapped.

That’s where LLMs can add real value. As (Liu, Zhang, & Wang, 2023) suggests, blending LLMs with conventional analytical pipelines can yield more nuanced perspectives on market trends and consumer thinking. Imagine combining a model that excels at parsing natural language text with a traditional forecasting tool that understands raw sales numbers. By marrying these strengths, retail analytics could advance beyond identifying “what happened” toward understanding “why,” capturing the sentiment behind reviews or the subtle seasonal references in product descriptions. This line of thinking informs our approach, which aims to weave LLM-driven insights into existing retail frameworks so that the end result is richer, more context-aware decision support.

## 2.3 Applications of LLMs in Retail

Beyond these initial use cases, LLMs are gradually reshaping broader aspects of the retail experience. Consider the example from (Soni, 2023), where an LLM-powered chatbot interacts directly with consumers, helping them discover products more efficiently and, by extension, driving up satisfaction metrics. On the operational side, models described by (Holstein & Jack, 2024) categorize and interpret diverse streams of customer feedback. This allows retailers to spot brewing issues before they become full-blown problems, tailor

their marketing messages more precisely, and even customize product descriptions so that they resonate with particular target audiences.

However, achieving these benefits on an ongoing basis requires fine-tuned approaches that respect the intricacies of retail: an environment that demands real-time adaptability, scalability, and context-aware reasoning. As we show in our proposed architecture, harnessing LLMs for retail isn't merely a matter of applying off-the-shelf solutions. Instead, it involves thoughtfully merging LLMs with task-specific optimizations designed to handle the unique linguistic, operational, and performance constraints of the retail sector.

## 3 Methodology

### 3.1 RetailLLM Architecture

The flowchart in Figure 1 illustrates the core architecture and workflow of the RetailLLM system. It highlights the sequential processes involved, and these processes are managed by a "RetailLLM" class, which is the main class of this project. This class interacts with Kernel LLM module, Data Analysis module, Data Base module, Webscraper module, and WebUI module.

This section focuses on how the "RetailLLM" class leverages two auxiliary classes, Data Node and Prompt Tree, to effectively manage five submodules mentioned above. The Data Node class acts as the central repository for all data exchanged during system's conversation with the user, and it plays a critical role in enabling multi-turn conversations and data reuse. It ensures that the user does not need to repeatedly input the same data as the system can autonomously determine which previously provided information is relevant for the current query. On the other hand, the Prompt Tree class efficiently manages all the prompts required for interactions with the user. It dynamically structures and organizes prompts based on the context of the conversation, allowing the system to seamlessly adapt to user needs while optimizing interactions with the Kernel LLM.

While this section introduces the high-level orchestration of these components by the "RetailLLM" class, the subsequent sections will delve into the technical details and experimental results of the five submodules.

### 3.1.1 Auxiliary Classes

The RetailLLM system integrates two key auxiliary classes, Prompt Tree and Data Node, to streamline the workflow and enhance functionality. This section introduces these two classes, providing an overview of their design and implementation.

**Prompt Tree** The Prompt Tree class acts as a flexible template for organizing and managing prompts in a hierarchical structure. We designed this class to dynamically generate prompts tailored to specific user input or tasks, enabling seamless and context-aware communication with the Kernel LLM.

The Prompt Tree is initialized with a global "Background" variable, which establishes the system's identity and scope as a specialized AI assistant focused on retail analysis. The Prompt Tree is divided into several subtrees, each serving a distinct purpose. For instance, the "GetDataSubTree" is responsible for classifying user tasks and determining appropriate data acquisition methods, while the "GetAnalysisSubTree" generates prompts to guide the LLM in selecting suitable data analysis techniques. These subtrees are further subdivided into nodes, which contain task-specific instructions, options (if applicable), and constraints to ensure precise and efficient interactions.

The hierarchical structure of the Prompt Tree offers flexibility and scalability, enabling the system to adapt dynamically to diverse retail-specific tasks. The Prompt Tree ensures that the LLM receives well-structured and pertinent input, optimizing the quality and accuracy of its outputs.

**Data Node** The Data Node class serves as a critical component of the RetailLLM system, which enables data information extraction, data storage, and data information updating. This design allows the Data Node to maintain a shared state across multiple modules and interactions, facilitating information integration and data reuse.

Data Node manages a dictionary of "DataContainer" objects, each representing a dataset with associated attributes such as descriptions, data types, and unique identifiers. When new data is added to the system, the Data Node automatically updates its internal structures, and ensure that the data requested or newly provided by the user is always active. This mechanism allows us to design a set of prompts (shown in Appendix C.2.3) to enable multi-turn conversation. When new data is added, a description is generated according to the information extracted by the DataContainer and the user input. And when previous data is required for analysis, each descriptions stored in the DataNode is

integrated into the prompt through the Prompt Tree class. Thus, the kernel LLM knows all the information about the previous used data to make accurate decision. The selected data is tracked by activated variable. We say a data is activated if it's requested for analysis in current process.

### 3.1.2 Communication with modules

RetailLLM serves as the orchestrator, facilitating interactions with various specialized modules to accomplish different tasks. The details about communication are shown in Figure 2.

**Kernel LLM** Kernel LLM handles natural language understanding and reasoning tasks, processing user input wrapped in prompts generated by the prompt tree. Prompt samples can be found in Appendix C.2. It also helps to conclude the results generated by the webscraper and data analysis node according to the user's request.

**Data Analysis Module** Data Analysis module performs statistical and analytical tasks, such as regression or time series forecasting, on the data provided by the user or requested by the user that are currently activated in Data Node. The analysis method is dynamically determined by the Kernel LLM using prompt-based classification. The prompt used for classification integrates the information from the data (the attributes' description, name, types, shape, and head of the table) and the user's inputs, utilizes all the information that can be used for analysis task classification. The prompt is shown in Appendix ??.

**Database Module** Database module stores data previously collected by our team. It serves as an additional data source in decision making. Each table stored in database has a "description" attribute, which briefly introduces the content of this table, and significantly facilitates the kernel LLM's decision. We designed a prompt (shown in Appendix C.2.5) based on this attributes that asks the kernel LLM to determine which table should be chosen for process. The chosen table is then passed to Data Node, where a new container is created for storing this table and activated for further analysis.

**Webscraper Module** Webscraper module takes in keywords, and performs sentiment analysis on the scraped data, and return the results along with summarized data. Re-

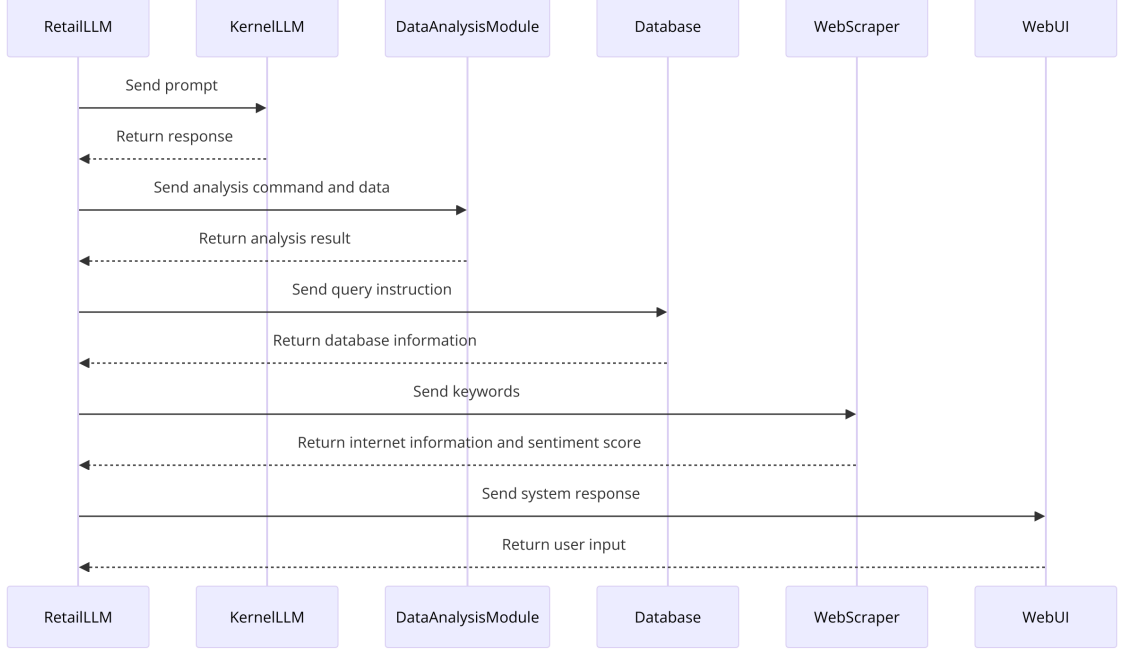


Figure 2: This is sequence diagram depicting the interactions between RetaiLLM with five other components.

tailLM class asks kernel LLM to generate keywords for webscraping using prompts shown in Appendix C.2.4.

**WebUI Module** WebUI Module acts as the user interface, presenting responses and analysis to the user. It passes user input to the RetaiLLM class, initiating the processing pipeline. And the RetaiLLM return the responses generated by the system.

**System Response** For each processing pipeline, RetaiLLM utilized certain submodules according to the user’s input. And for each submodule’s utilization, an analytical results is generated. We want the final system response to be diverse, dynamic and tailored to each question. Thus, we designed a prompt (shown in Appendix C.2.6 integrate the results to generate some insights.

### 3.2 Kernel LLM Module: Deployment and Fine-tuning

In this project, we utilized various large language models (LLMs) as kernel components to facilitate fine-tuning and prompt engineering. This section will introduce the details of LLM deployment and fine-tuning process in our framework.

Ensuring fast inference from the deployed LLMs is crucial to enable a seamless and com-

portable user experience in RetailLLM. Our work focuses primarily on the prompt engineering of large-scale models, such as Qwen-72B-Instruct (Qwen Team, 2024; Yang et al., 2024), while we also investigate fine-tuning strategies for smaller LMs such as LLaMA-1B-Instruct and LLaMA-3B-Instruct (Grattafiori et al., 2024). Although large-scale models exhibit superior performance, the exploration of smaller models is also meaningful. Lightweight models can run locally on edge devices such as smartphones and personal computers, which offers greater flexibility, lower operating costs, and enhanced privacy. These advantages are especially important for handling sensitive queries that involve business secrets. The following subsections will explore the details of deployment and fine-tuning.

### 3.2.1 Deployment of LLM Kernel through VLLM

To make the Qwen-72B-Instruct model work quickly and smoothly, we used VLLM (Shen, Ma, & et al., 2023), which is widely used in LLM industry. VLLM uses a technique called pipelining and smart memory management to accelerate model inference. On the server side, VLLM waits for requests on a specific port, allowing other programs to easily interact with the model using a familiar API. This setup makes it possible for handling multiple requests at once and makes it faster for model inference. After downloading a model from huggingface (Wolf et al., 2020), we can deploy this model through VLLM using cli-command shown in Appendix B.1. This approach significantly accelerates the model loading and model inference, whose results are shown in Section 4.2.1

### 3.2.2 Client Access to Kernel LLM

To make it easy to communicate with the vllm server, we use the Python OpenAI library (OpenAI, 2023). Treating the LLM as a separate server has several significant benefits:

- **Less data to send:** We only need to transmit text, not images or other files. This is possible because most of the computations can be performed on the user’s device.
- **More flexibility:** Switching the kernel LLM is easy with this approach.

We can use the VLLM setup described in 3.2.1 or integrate with other existing APIs. A sample Python code demonstrating how to achieve this is provided in Appendix B.2. This process is concluded in Figure 3



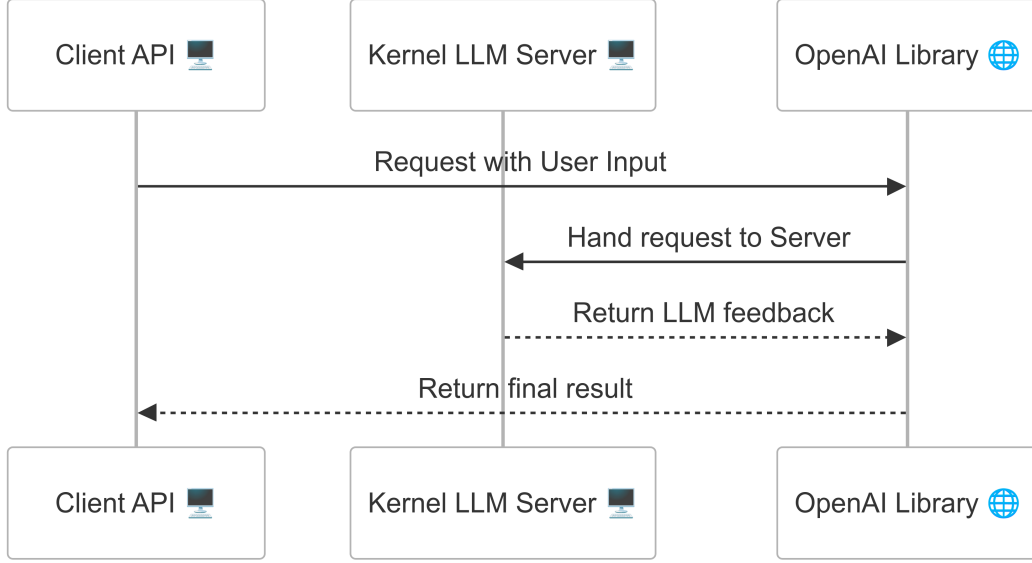


Figure 3: Client and Kernel LLM Server communication diagram: Two nodes Client and Kernel LLM Server communicate through OpenAI Library (OpenAI, 2023)

### 3.2.3 Fine-tuning of Llama 3.2 Models: 1B-Instruct and 3B-Instruct

We fine-tune Llama 3.2 1B-Instruct and Llama 3.2 3B-Instruct (Grattafiori et al., 2024) models. A synthetic dataset of approximately 800 dialogue samples is created using DeepSeek API (Authors, n.d.). This API gives accurate feedback, and is cheap and fast. We first request DeepSeek with prompts shown in Appendix C.1, which result in a raw dataset. We then sift the data manually, and add 50 more data points for difficult cases. The generation process is shown in Figure 4. This dataset is split into a training set of size 600 and a testing set of size 200. The fine-tuning process is executed within the Python Transformers (Wolf et al., 2020) framework, which provides robust environment for model training. The models are trained for 120 epochs. A comprehensive analysis of the experimental results will be presented in the Section 4.2.2.

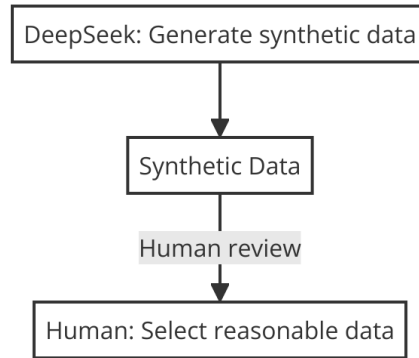


Figure 4: Pipeline for synthetic data generation

### 3.3 RetailLLM Analysis

In retail data analysis, extracting meaningful insights from vast and varied datasets is crucial for strategic decision-making. This section explores five primary analytical methodologies, each with multiple sub-methods designed to uncover different facets of retail dynamics. The integration of these methods provides a comprehensive understanding of the factors influencing retail performance and customer behavior.

#### 1. Correlation Analysis

- **Pearson Correlation Coefficient** (Cohen et al., 2009): Measures linear relationships between variables.
- **Spearman Correlation Coefficient** (Ali Abd Al-Hameed, 2022): Assesses monotonic relationships, suitable for non-parametric data.
- **Kendall Correlation Coefficient** (Stepanov, 2015): Provides a rank-based correlation, useful for small datasets.

#### 2. Regression Analysis

- **Ridge Regression** (Marquardt & Snee, 1975): Regularized regression method that prevents overfitting by adding a penalty to the coefficients.
- **Lasso Regression** (Ranstam & Cook, 2018): Regularization technique that shrinks some coefficients to zero, useful for feature selection.
- **Support Vector Machine (SVM) Regression** (Suthaharan & Suthaharan, 2016): Robust regression method that works well for high-dimensional data.
- **Random Forest Regression** (Rigatti, 2017): Ensemble method that uses multiple decision trees for predictions, handling non-linearity and interaction effects effectively.
- **Transformer-Based Regression** (Vaswani, 2017): Leverages transformer architectures to capture complex patterns in data.

#### 3. Cluster Analysis

- **KMeans Clustering** (Kodinariya, Makwana, et al., 2013): Groups data points into clusters based on their similarity, commonly used for customer segmentation.
- **DBSCAN Clustering** (Khan, Rehman, Aziz, Fong, & Sarasvady, 2014): Density-based clustering algorithm that groups closely packed points and marks outliers.

- **Hierarchical Clustering** (Murtagh & Contreras, 2012): Builds a tree of clusters, useful for understanding data hierarchies.

#### 4. Time Series Analysis

- **ARIMA** (Contreras, Espinola, Nogales, & Conejo, 2003): Autoregressive Integrated Moving Average model used for forecasting time series data.
- **Transformer-Based Time Series Model** (Vaswani, 2017): A transformer-based model for capturing long-term dependencies in time series data.

#### 5. Survival Analysis

- **Kaplan-Meier Estimator** (Goel, Khanna, & Kishore, 2010): Estimation method for survival functions, commonly used in customer retention analysis.
- **Cox Proportional Hazards Model** (Bender, Augustin, & Blettner, 2005): A regression model for investigating the effect of variables on survival times.

### 3.3.1 Correlation Analysis

Understanding the interrelationships between various retail metrics is essential for optimizing operations and strategies. This study employs three correlation coefficient methods—Pearson (Cohen et al., 2009), Spearman (Ali Abd Al-Hameed, 2022), and Kendall (Abdi, 2007)—to quantify the associations between different variables, each offering unique perspectives on the data.

**Pearson Correlation Coefficient** The Pearson correlation coefficient ( $r$ ) quantifies the linear relationship between two continuous variables, assuming normal distribution (Cohen et al., 2009). Its primary focus is on identifying the strength and direction of linear associations. A key advantage of Pearson’s method is its sensitivity to linear trends, making it ideal for detecting direct proportionality or inverse relationships in normally distributed data (Sedgwick, 2012).

$$r = \frac{\sum(X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum(X_i - \bar{X})^2 \sum(Y_i - \bar{Y})^2}},$$

where  $X_i$  and  $Y_i$  represent the observed values of variables  $X$  and  $Y$ , and  $\bar{X}$  and  $\bar{Y}$  are the respective means. The value of  $r$  ranges from  $-1$  to  $1$ , with values closer to  $1$  or  $-1$  indicating a stronger linear relationship.



RetailLLM

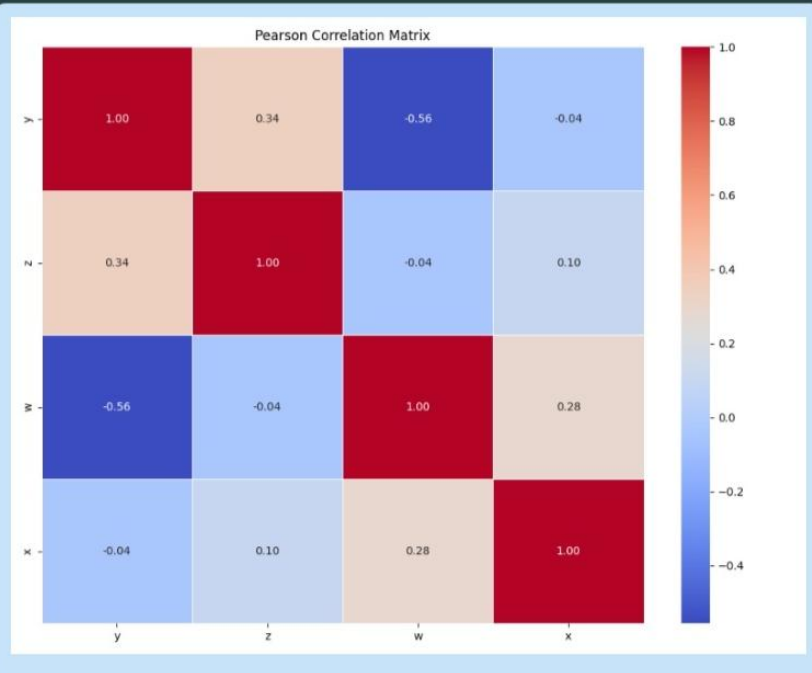
Hello, I am RetailLLM, a prompt tuned general-purpose retail assistant. I can help you analysis data, provide business insights, and search information.

You have uploaded a CSV file successfully!

Help me perform a correlation test on, sales, promote, discount Trends, and Advertising

Highest Pearson correlation is between y and w with a coefficient of -0.56. A correlation heatmap visualization plot has been generated.

The correlation test shows a strong negative relationship between sales and advertising.



Type message here

Figure 5: Demonstration of Pearson Correlation Coefficient

**Spearman Correlation Coefficient** Unlike Pearson’s method, the Spearman correlation coefficient ( $\rho$ ) assesses the monotonic relationship between variables based on their ranked values. This non-parametric approach is advantageous for handling non-linear relationships and ordinal data, providing flexibility when data do not meet the assumptions of normality required by Pearson’s coefficient (Ali Abd Al-Hameed, 2022; Hauke & Kossowski, 2011).

$$\rho = 1 - \frac{6 \sum d_i^2}{n(n^2 - 1)},$$

where  $d_i$  represents the rank differences between variable pairs, and  $n$  is the number of observations. The value of  $\rho$  also ranges from  $-1$  to  $1$ , indicating the strength and direction of the monotonic relationship.

**Kendall Correlation Coefficient** The Kendall correlation coefficient ( $\tau$ ) evaluates the concordance between pairs of observations, focusing on the consistency of rankings. Its strength lies in its robustness to outliers and its suitability for smaller sample sizes (Stepanov, 2015). Kendall’s method is particularly effective in scenarios where the data exhibit tied ranks or when the relationship between variables is not strictly linear (Abdi, 2007).

$$\tau = \frac{n_c - n_d}{\frac{1}{2}n(n - 1)},$$

where  $n_c$  is the number of concordant pairs,  $n_d$  is the number of discordant pairs, and  $n$  is the total number of observations.

### 3.3.2 Regression Analysis

Predicting retail sales and understanding the factors influencing them are critical for strategic planning. This study employs various regression techniques—Ridge Regression (Marquardt & Snee, 1975), Lasso Regression (Ranstam & Cook, 2018), Support Vector Machine (SVM) Regression (Suthaharan & Suthaharan, 2016), Random Forest Regression (Rigatti, 2017), and Transformer-Based (Vaswani, 2017) Regression—to model and forecast sales figures.

**Ridge Regression** Ridge Regression (Marquardt & Snee, 1975) addresses multicollinearity among predictors by introducing an  $\ell_2$  regularization term. This method stabilizes coefficient estimates, enhancing the model’s predictive accuracy when predictors are highly correlated.

$$\min_{\beta} \left\{ \sum_{i=1}^n (y_i - \mathbf{x}_i^T \beta)^2 + \alpha \sum_{j=1}^p \beta_j^2 \right\},$$

where  $\alpha$  is the regularization parameter controlling the magnitude of the coefficients.

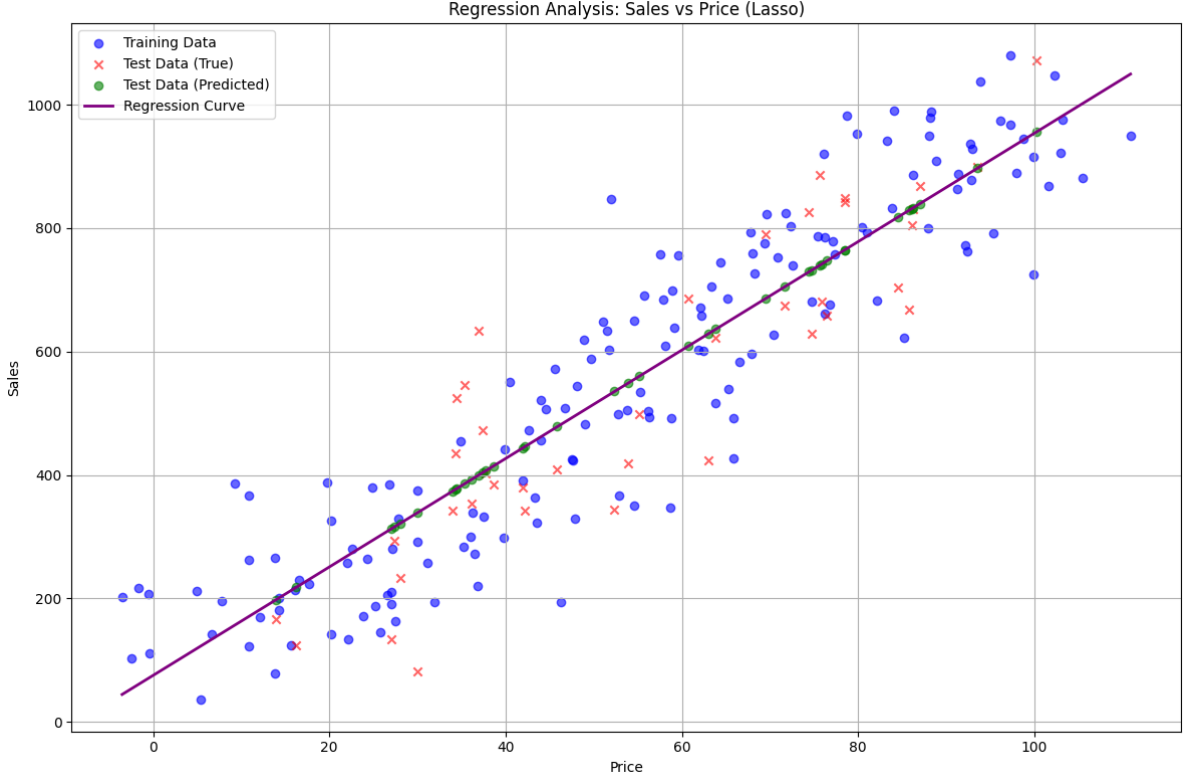


Figure 6: Demonstration of Lasso Regression

**Lasso Regression** Lasso Regression (Ranstan & Cook, 2018) incorporates an  $\ell_1$  penalty, promoting sparsity in the coefficient estimates. This dual capability of regularization and variable selection simplifies the model by eliminating insignificant predictors, thereby enhancing interpretability without compromising performance.

$$\min_{\beta} \left\{ \sum_{i=1}^n (y_i - \mathbf{x}_i^T \beta)^2 + \alpha \sum_{j=1}^p |\beta_j| \right\},$$

**Support Vector Machine (SVM) Regression** SVM Regression (Suthaharan & Suthaharan, 2016) captures non-linear relationships through kernel functions, enabling the modeling of complex patterns in sales data. Its robustness to outliers and flexibility in handling high-dimensional data make it a valuable tool for intricate retail forecasting tasks.

$$\min_{w,b} \left\{ \frac{1}{2} \|w\|^2 \right\}$$

subject to

$$|y_i - (w^T \mathbf{x}_i + b)| \leq \epsilon, \quad \forall i.$$

**Random Forest Regression** Random Forest Regression (Rigatti, 2017) leverages an ensemble of decision trees to model non-linear interactions and feature importance. Its ability to handle large datasets and provide insights into predictor significance makes it a powerful method for understanding the drivers of retail sales.

$$\hat{y} = \frac{1}{T} \sum_{t=1}^T \hat{y}_t,$$

, where  $T$  is the number of trees in the forest, and  $\hat{y}_t$  is the prediction from the  $t$ -th tree.

**Transformer-Based Regression** For advanced modeling of feature dependencies and interactions in sales data, a Transformer-based (Vaswani, 2017) regression model is employed. Unlike traditional regression models that assume a fixed relationship between predictors and the target variable, the Transformer leverages self-attention mechanisms to dynamically weigh the importance of different features. Specifically, the model processes input features  $\mathbf{X} \in \mathbb{R}^{n \times d}$ , where  $n$  is the number of samples and  $d$  is the number of features, and maps them to predictions  $\hat{\mathbf{y}} \in \mathbb{R}^n$ .

The self-attention mechanism computes attention scores  $A$  as:

$$A = \text{softmax} \left( \frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}} \right),$$

where  $\mathbf{Q} = \mathbf{X}\mathbf{W}^Q$ ,  $\mathbf{K} = \mathbf{X}\mathbf{W}^K$ , and  $\mathbf{V} = \mathbf{X}\mathbf{W}^V$  are the query, key, and value matrices derived from input  $\mathbf{X}$  through learned weight matrices  $\mathbf{W}^Q$ ,  $\mathbf{W}^K$ , and  $\mathbf{W}^V$ . The output of the attention layer is then:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = A\mathbf{V}.$$

Multiple attention heads allow the model to focus on different representation subspaces:

$$\text{MultiHead}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Concat}(\text{head}_1, \dots, \text{head}_h)\mathbf{W}^O,$$

where each  $\text{head}_i = \text{Attention}(\mathbf{Q}\mathbf{W}_i^Q, \mathbf{K}\mathbf{W}_i^K, \mathbf{V}\mathbf{W}_i^V)$ , and  $\mathbf{W}^O$  is the output weight matrix.

The Transformer model leverages these mechanisms to learn intricate feature interactions, enabling accurate regression predictions. For a comprehensive overview of the Transformer architecture, refer to Appendix E.5.1.



RetailLLM

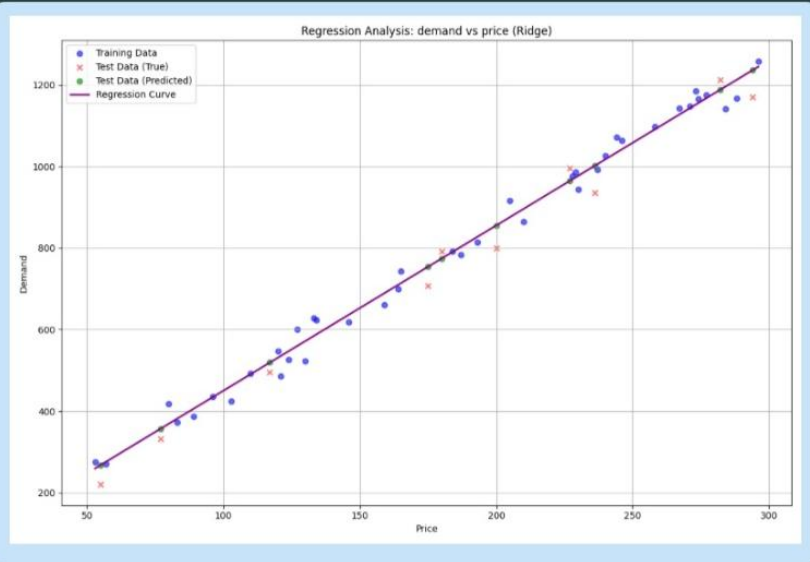
Hello, I am RetailLLM, a prompt tuned general-purpose retail assistant. I can help you analysis data, provide business insights, and search information.

help me perform linear ridge regression on price and demand

[53, 274, 240, 120, 277, 146, 83, 159, 80, 284, 296, 246, 96, 117, 237, 133, 210, 236, 127, 77, 228, 267, 130, 57, 205, 180, 294, 258, 230, 103, 200, 124, 55, 273, 271, 164, 288, 229, 187, 175, 193, 121, 89, 165, 110, 227, 244, 134, 282, 184]  
[275, 1165, 1027, 548, 1175, 618, 372, 660, 418, 1141, 1257, 1063, 436, 496, 993, 628, 865, 936, 600, 332, 976, 1143, 523, 271, 917, 792, 1171, 1097, 943, 425, 799, 527, 221, 1184, 1147, 700, 1167, 986, 784, 707, 814, 486, 387, 743, 493, 995, 1071, 624, 1212, 791]

Ridge Model - Intercept: 44.8649, Coefficients: {'price': 4.052686915817462} A regression visualization plot has been generated.

The Ridge Model shows a positive relationship between price and demand.



Type message here



Figure 7: Demonstration of Ridge Regression



### 3.3.3 Cluster Analysis

Segmenting customers into distinct groups allows retailers to tailor strategies to specific demographics and behaviors. This study employs three clustering techniques—KMeans (Sinaga & Yang, 2020), DBSCAN (Khan et al., 2014), and Hierarchical Clustering (Murtagh & Contreras, 2012)—to identify and visualize customer segments based on standardized feature sets.

**KMeans Clustering** KMeans (Sinaga & Yang, 2020) clustering partitions the dataset into a predefined number of clusters by minimizing intra-cluster variance. Its centroid-based approach is computationally efficient and effective for large datasets, enabling retailers to categorize customers into homogeneous groups for targeted marketing (Kodinariya et al., 2013).

$$J = \sum_{k=1}^K \sum_{i \in C_k} \|x_i - \mu_k\|^2,$$

where  $K$  is the number of clusters,  $C_k$  represents the set of data points assigned to cluster  $k$ ,  $x_i$  is the  $i$ -th data point in cluster  $k$ , and  $\mu_k$  is the centroid of cluster  $k$ .

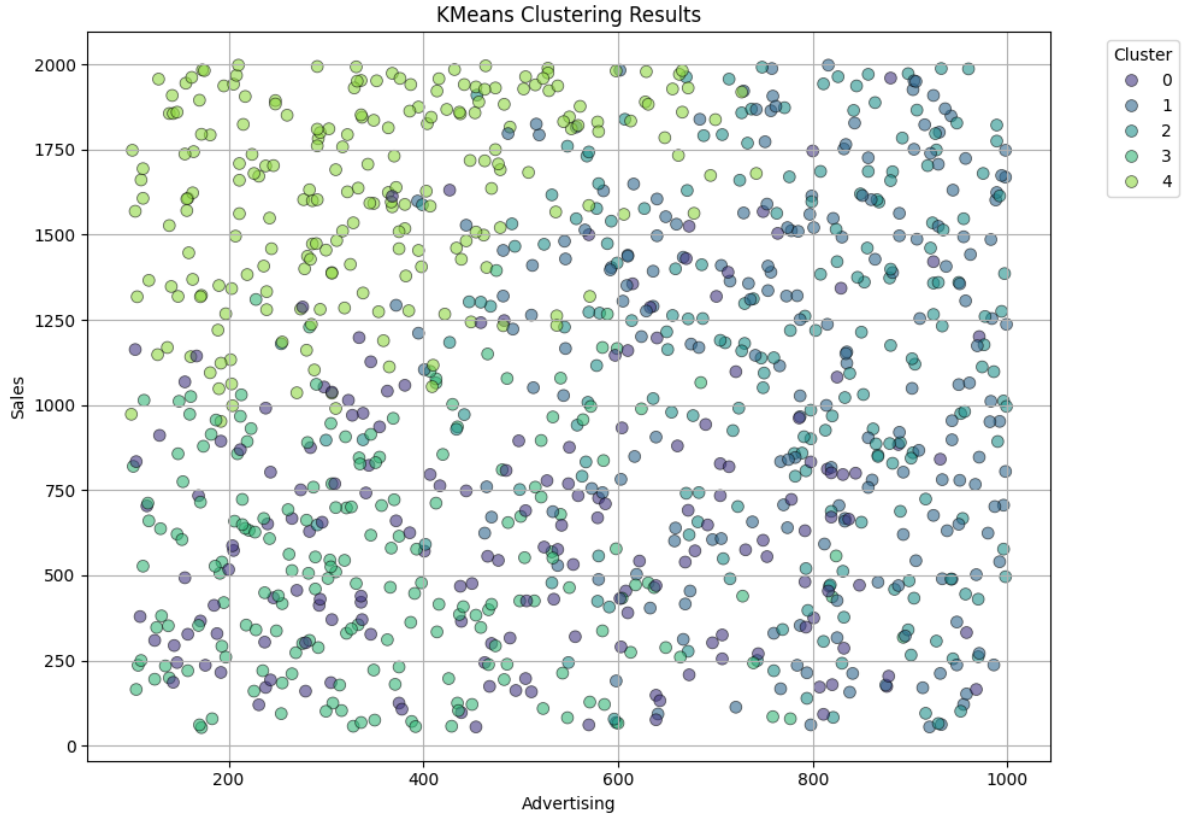


Figure 8: Demonstration of KMeans Clustering

**DBSCAN Clustering** DBSCAN (Khan et al., 2014) (Density-Based Spatial Clustering of Applications with Noise) identifies clusters based on data density, effectively handling noise and discovering clusters of arbitrary shapes. This method is advantageous for datasets with varying densities, allowing the detection of natural groupings without specifying the number of clusters in advance (Schubert, Sander, Ester, Kriegel, & Xu, 2017). The algorithm’s parameters,  $\epsilon$  (radius) and  $\text{min\_samples}$  (minimum number of points), are crucial in defining the density criteria for cluster formation. For a detailed pseudocode of the DBSCAN algorithm, refer to Algorithm 1 in Appendix D.

$$\text{Core}(p) = \{q \mid \|p - q\| \leq \epsilon\}, \quad \text{if } |\text{Core}(p)| \geq \text{min\_samples},$$

where  $p$  and  $q$  are points in the dataset, and  $\|\cdot\|$  represents the Euclidean distance between points. Points  $p$  and  $q$  are part of the same cluster if  $p$  is a core point and  $q$  is within its  $\epsilon$ -radius.

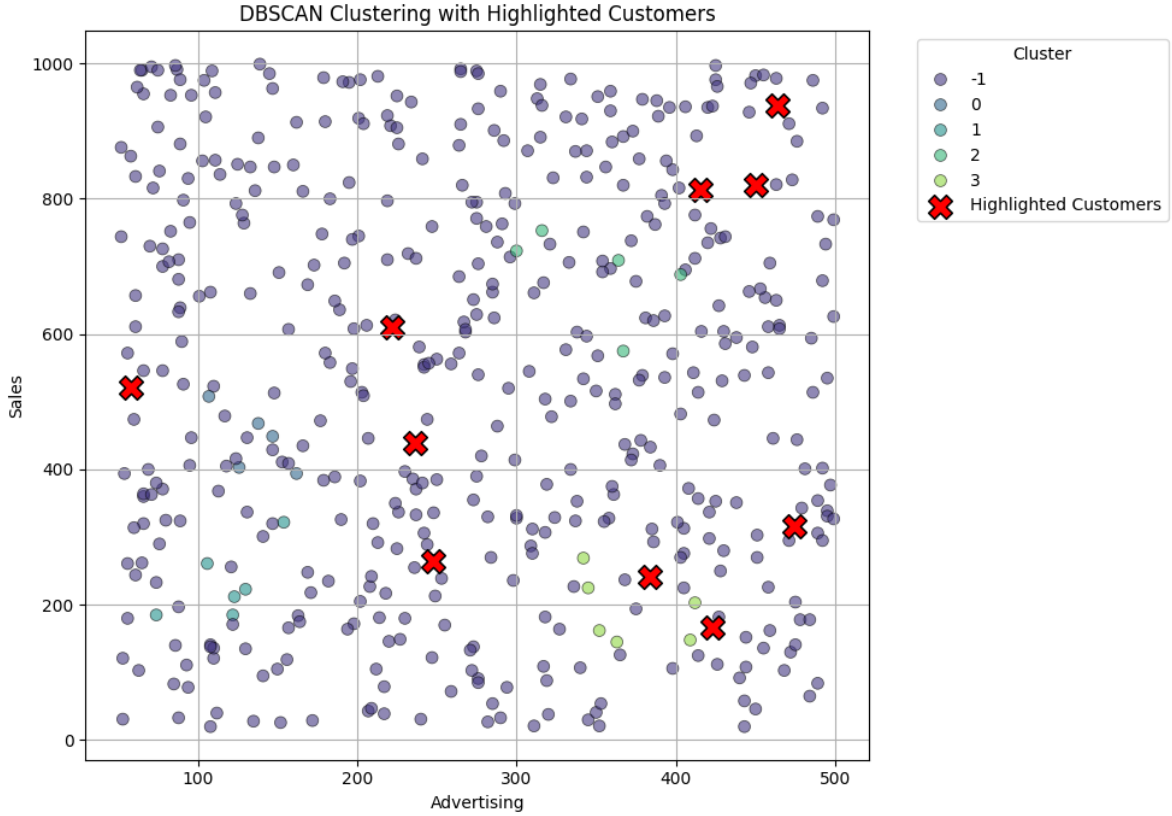


Figure 9: Demonstration of DBSCAN Clustering

**Hierarchical Clustering** Hierarchical clustering (Murtagh & Contreras, 2012) builds a tree-like structure of nested clusters, offering a multilevel perspective on customer segmentation. The agglomerative approach, commonly used in this study, merges clusters based on linkage criteria such as Ward’s method, which minimizes within-cluster vari-

ance (Nielsen & Nielsen, 2016).

$$\min \sum_{k=1}^K \sum_{i \in C_k} \|x_i - \mu_k\|^2,$$

where  $C_k$  represents cluster  $k$ ,  $x_i$  is a data point in cluster  $C_k$ , and  $\mu_k$  is the centroid of cluster  $C_k$ .

### 3.3.4 Time Series Analysis

Accurate forecasting of retail sales is essential for inventory management, budgeting, and strategic planning. This study employs the Autoregressive Integrated Moving Average (ARIMA) model (Shumway, Stoffer, Shumway, & Stoffer, 2017) and a Transformer-based (Vaswani, 2017) time series model to predict future sales based on historical data.

**ARIMA** The ARIMA model (Shumway et al., 2017) is applied to datasets exhibiting clear seasonal patterns over a moderate number of cycles. By decomposing the time series into trend and seasonal components, ARIMA captures both the underlying growth and periodic fluctuations, enabling reliable short to medium-term forecasts (Contreras et al., 2003).

$$Y_t = \alpha + \phi_1 Y_{t-1} + \phi_2 Y_{t-2} + \cdots + \phi_p Y_{t-p} + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \cdots + \theta_q \epsilon_{t-q} + \epsilon_t,$$

where:

- $\phi_i$  are the autoregressive coefficients,
- $\theta_j$  are the moving average coefficients,
- $\epsilon_t$  is the error term,
- $d$  denotes the order of differencing to achieve stationarity.

The forecasting is achieved by projecting the trend forward and reintroducing the seasonal pattern:

$$\hat{Y}_{t+h} = \hat{Trend}_{t+h} + Seasonality_{t+h}$$

where  $h$  represents the forecast horizon.



*RetailLM*

Hello, I am RetailLLM, a prompt tuned general-purpose retail assistant. I can help you analysis data, provide business insights, and search information.

predict the next sale value, sales:

[197, 114, 156, 115, 168, 141, 180, 108, 107, 184, 105, 151, 139, 155, 129, 144, 126, 162, 149, 111, 176, 163, 164, 169, 134, 186, 123, 110, 195, 177]

Selected Forecasting Method: Transformer. Forecasted values:

[2020-01-31: 166.89]

[2020-02-01: 152.93]

[2020-02-02: 124.79]

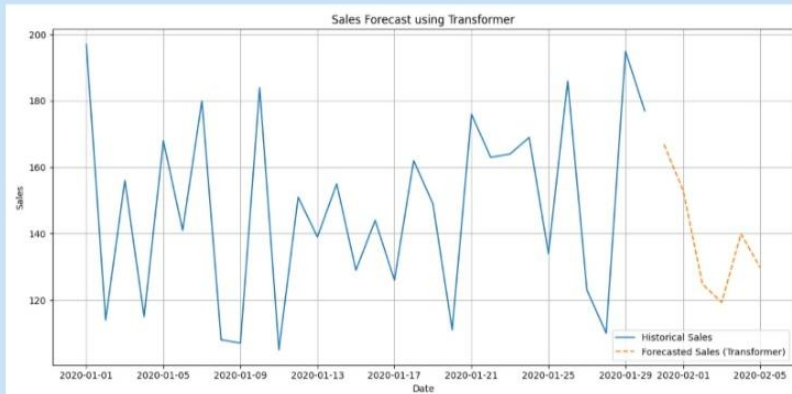
[2020-02-03: 119.29]

[2020-02-04: 139.93]

[2020-02-05: 129.73]

A time series forecast visualization plot has been generated.

The next predicted sale value is 178.



Type message here



Figure 10: Demonstration of Transformer-Based Time Series Analysis

**Transformer-Based Time Series Model** For datasets with extensive historical data and complex temporal dependencies, a Transformer-based (Vaswani, 2017) time series model is utilized. This model is specifically tailored to handle sequential data by capturing long-range temporal dependencies through its self-attention mechanisms. Let  $\mathbf{X}_t = [Y_{t-n+1}, Y_{t-n+2}, \dots, Y_t]$  represent the input sequence of past  $n$  observations used to predict the future value  $Y_{t+1}$ .

The Transformer processes the input sequence through multiple encoder layers, each consisting of multi-head self-attention and feedforward networks. The attention scores are computed as:

$$A = \text{softmax} \left( \frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}} \right),$$

where  $\mathbf{Q} = \mathbf{X}_t \mathbf{W}^Q$ ,  $\mathbf{K} = \mathbf{X}_t \mathbf{W}^K$ , and  $\mathbf{V} = \mathbf{X}_t \mathbf{W}^V$ . The output of the attention layer is then:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \mathbf{A}\mathbf{V}.$$

Multiple attention heads allow the model to attend to different parts of the input sequence simultaneously:

$$\text{MultiHead}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Concat}(\text{head}_1, \dots, \text{head}_h) \mathbf{W}^O,$$

The final prediction  $\hat{Y}_{t+1}$  is obtained by passing the transformed representation through a linear decoder:

$$\hat{Y}_{t+1} = \mathbf{W}^D \mathbf{H},$$

where  $\mathbf{H}$  is the output from the Transformer encoder layers, and  $\mathbf{W}^D$  is the decoder weight matrix. This architecture allows the model to effectively capture and utilize temporal patterns and dependencies inherent in the sales data. For a comprehensive overview of the Transformer architecture, refer to Appendix E.5.2.

### 3.3.5 Survival Analysis

Understanding customer retention is vital for sustaining revenue and fostering brand loyalty. This study employs Survival Analysis techniques—the Kaplan-Meier Estimator and the Cox Proportional Hazards Model—to evaluate customer churn and retention patterns.

**Kaplan-Meier Estimator** The Kaplan-Meier Estimator (Goel et al., 2010) is utilized to estimate the survival function, representing the probability of customers remaining ac-

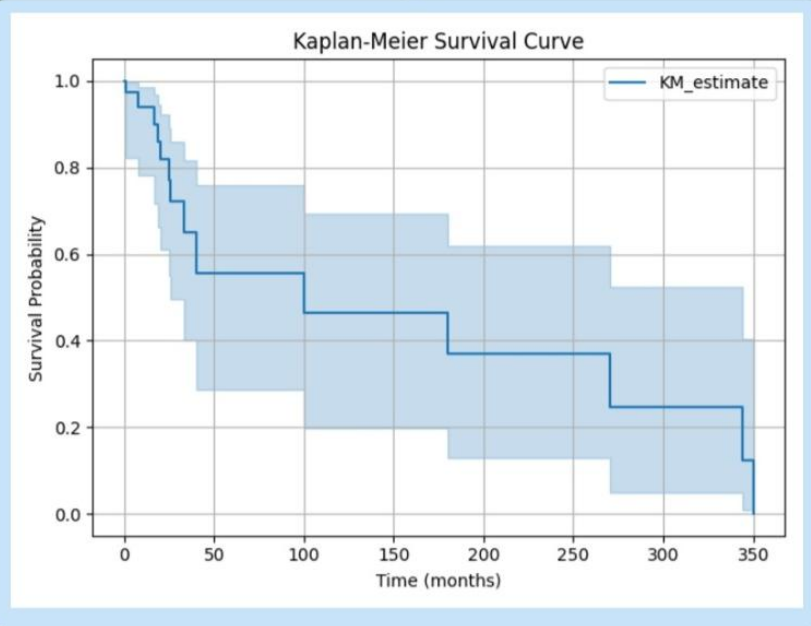


RetailLLM

Hello, I am RetailLLM, a prompt tuned general-purpose retail assistant. I can help you analysis data, provide business insights, and search information.

You have uploaded a CSV file successfully!  
Conducting Survival Analysis on  
Product

Median survival time: 100.00 months.  
The median survival is 100 months.



Type message here



Figure 11: Demonstration of Kaplan-Meier Estimator

tive over time. This non-parametric method effectively handles censored data, providing insights into median survival times and overall retention trends (Etikan, Abubakar, & Alkassim, 2017).

$$S(t) = \prod_{t_i \leq t} \left(1 - \frac{d_i}{n_i}\right),$$

where  $t_i$  is the event time,  $d_i$  is the number of events (e.g., churn) at time  $t_i$ , and  $n_i$  is the number of individuals at risk just prior to  $t_i$ .

**Cox Proportional Hazards Model** The Cox Proportional Hazards Model (Bender et al., 2005) assesses the impact of various covariates—such as age, income, and purchase frequency—on customer survival times. By estimating hazard ratios, this semi-parametric model identifies significant predictors of churn, enabling targeted retention strategies (Lin & Wei, 1989).

$$h(t|X) = h_0(t) \exp(\beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p),$$

where  $h_0(t)$  is the baseline hazard function,  $X_1, X_2, \dots, X_p$  are the covariates, and  $\beta_1, \beta_2, \dots, \beta_p$  are the coefficients to be estimated.

### 3.3.6 Summary of Analysis

This Analysis section presents a multifaceted approach to understanding and forecasting retail dynamics through Correlation Analysis, Cluster Analysis, Regression Analysis, Time Series Analysis, and Survival Analysis. By leveraging a diverse set of statistical and machine learning methodologies, the study uncovers intricate relationships, segments customers effectively, predicts sales with high accuracy, and evaluates customer retention comprehensively. The integration of these analytical techniques provides a robust foundation for informed decision-making and strategic planning in the retail sector.

## 3.4 Database

RetailLLM leverages MongoDB to efficiently store and manage a vast amount of information necessary for providing accurate and contextually relevant responses to user queries. When a user sends a message, the chatbot first processes the input to identify the context and key elements. This processed information is then used to query the MongoDB database for pertinent data. MongoDB, as a flexible, document-oriented database, contains a variety of entities, including user profiles, historical interactions, predefined responses, and domain-specific knowledge (Pribadi, Firdausi, & Arhandi, 2023). Table 1

presents the contents of the datasets.

Table 1: Content of Datasets

Dataset	Explanation
Principles of Finance	Covers the fundamental concepts and theories of finance, including time value of money, risk and return, and financial markets.
Principles of Entrepreneurship	Provides insights into the basics of entrepreneurship, including idea generation, business planning, and startup management.
How to Set Up Business	Includes step-by-step guides and resources on starting a new business, including legal requirements, business registration, and initial setup.
Business Insight	Offers analyses and case studies on various business strategies, market trends, and industry insights.
Markets, Investments, and Financial Management	Covers topics related to financial markets, investment strategies, portfolio management, and corporate finance.
Basics of Finance	Provides an introduction to finance, covering essential topics such as budgeting, saving, and basic financial instruments.
What is Finance	Explains the concept of finance, its importance, and its various branches, including personal, corporate, and public finance.
How to Own a Business	Includes information on acquiring, managing, and running a business, with practical tips and best practices.
Entrepreneurship Notes	Contains detailed notes on entrepreneurship, including lectures, seminars, and workshop summaries.
Overview of Financial Management	Provides a comprehensive overview of financial management principles, including financial planning, analysis, and control.

### 3.4.1 Framework Selection and Strategic Rationale

In the dynamic landscape of modern retail analytics, choosing MongoDB as the core database management system represents a strategic and forward-looking technological decision. Its NoSQL, document-oriented architecture offers unparalleled flexibility and scalability, surpassing the limitations of traditional relational databases. This makes MongoDB an ideal choice for our RetailLLM platform. Figure 12 illustrates the workings of MongoDB.

The decision to implement MongoDB is driven by its exceptional ability to manage complex, dynamic data structures typical in retail environments. Unlike rigid schema-based databases, MongoDB’s document model enables seamless representation of intricate business data, allowing for rapid iteration and adaptation to shifting analytical needs. This architectural flexibility is especially crucial in retail analytics, where data complexity and



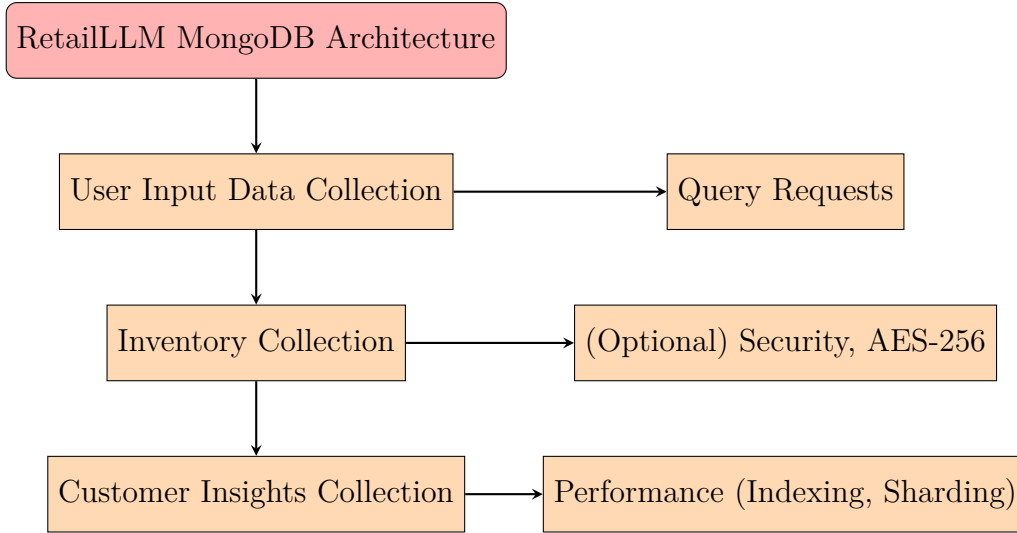


Figure 12: Flowchart showing the how MongoDB works

variability are persistent challenges.

This approach harnesses the strengths of both MongoDB and the LLM, with MongoDB offering scalable and reliable storage, while the LLM provides advanced natural language understanding and generation. Together, they enable the chatbot to deliver high-quality, context-aware responses that significantly enhance the user experience (Bratić, Šapina, Jurečić, & Gršić, 2024).

### 3.4.2 Data Ecosystem and Structural Design

Our MongoDB architecture adopts a multi-collection approach that captures the multidimensional nature of retail data. Each collection is tailored to a specific domain of business intelligence, enabling granular and contextually rich data representation.

### 3.4.3 Advanced Query Request and Search Architecture

The query design goes beyond traditional database querying methods, offering a sophisticated, flexible, and secure approach to data retrieval and analysis.

The search mechanism integrates advanced filtering, projection, and aggregation features, allowing users to extract precise insights with minimal computational overhead. By supporting complex JSON-based query payloads, the system offers unparalleled flexibility for data exploration, supporting everything from high-level strategic analysis to detailed, transaction-level investigations.

### 3.4.4 Technical Implementation

**Cloud-Native and Scalability Considerations** Our database architecture is cloud-native, designed to scale massively to meet the dynamic demands of modern retail environments. MongoDB’s distributed architecture supports horizontal scaling, ensuring the platform can handle exponential growth in both data volume and computational complexity. The database can be deployed on cloud servers or locally, providing flexibility in infrastructure deployment.

This design supports a variety of cloud environments, from public cloud platforms like Azure to hybrid cloud and local configurations, consistently maintaining high performance and reliability.

**JSON-like Document Storage** MongoDB’s native JSON-like document storage (BSON) seamlessly integrates with Python’s data processing ecosystem, making it an ideal fit for our machine learning-driven approach. Its horizontal scaling ensures that our platform can handle rapid growth in both data and computational demands, which is critical for enterprise-level retail intelligence solutions.

**Security Considerations** Security is a core aspect of our database architecture. We go beyond basic authentication, implementing multiple layers of protection, including role-based access control, JWT-based authentication, and advanced encryption strategies. AES-256 encryption for sensitive data ensures that our platform meets the highest enterprise security standards.

**Storing Messages for Diagnostics** Both user messages and chatbot responses are stored in MongoDB for diagnostic purposes, enabling continuous performance improvement and issue resolution. These stored interactions provide valuable data for analyzing and refining the chatbot’s capabilities, ensuring its ongoing effectiveness.

## 3.5 Web Scraper

### 3.5.1 Introduction and Motivation

The custom web scraper is designed to provide the chatbot with up-to-date market insights, trends, and expert opinions not included in its base training data. The aim is to

enhance recommendation quality and relevance through a dynamic knowledge pipeline that integrates web scraping, sentiment analysis, and summarization.

### 3.5.2 System Architecture and Advantages

The proposed web scraping solution is architected for flexibility, reliability, and efficient data extraction. Compared to open-source alternatives, it provides:

1. **Flexible Search:** Keyword-based scraping without strict URL dependencies, multi-search engine support, and robust content validation and filtering.
2. **Advanced Processing:** Integration with LangChain for precise extraction, Selenium-based fallback with headless browsing, and minimum word count thresholds to ensure content quality.
3. **Performance Optimizations:** Concurrent execution via ThreadPoolExecutor, asynchronous processing, and thread-safe progress tracking.
4. **Robust Error Handling:** Multi-attempt retry mechanisms, comprehensive logging with timing metrics, and multiple fallback strategies.

### 3.5.3 Analytical Techniques

The web scraper's output is further enhanced through sentiment analysis and intelligent summarization:

**Sentiment Analysis:** The sentiment analysis employs a dual approach for more nuanced profiling:

- **Balanced Methodologies:** TextBlob is used for polarity and subjectivity scoring, while VADER provides compound sentiment scores.
- **Detailed Insights:** This combination ensures a richer and more detailed sentiment distribution, enhancing the accuracy of the analysis.

**DeepSeek Integration:** The DeepSeek model is employed for cost-effective, high-quality summarization. Benefits include:

- **Cost-effectiveness:** High token allowance at a competitive price point.
- **Technical Control:** Temperature adjustments for precise summarization and fully open-source availability.

**Final Decision:** The integrated solution stands out as the most practical and effective approach. It captures the clarity and detail of the highest-quality option without incurring untenable runtime costs, and it avoids the pitfalls of methods that are either too slow, too low-quality, or insufficiently balanced in sentiment. The data-driven trade-off analysis thus supports the integrated solution as the optimal choice for future scraping and summarization tasks.

## 3.6 Web UI

The frontend of the project is a web application developed using React.js. It features an interactive chat interface that enables users to engage with a retail-focused chatbot. The user interface is illustrated in Fig. 13.

### 3.6.1 Components

The application utilizes several key React packages, including the core `react` and `react-dom` libraries, along with specialized components from `chat-ui-kit-react` to build the chat interface. For enhanced functionality, `react-markdown` is employed to render markdown content, and `vite` is used as the build tool.

### 3.6.2 Architecture

The application's architecture is designed around key components that manage different aspects of the user interface. The main `App` component controls the chat logic, while specialized components like `MainContainer`, `ChatContainer`, and `MessageList` collaborate to deliver a seamless chat experience. The interface includes features such as message display, typing indicators, and a message input field.

A key functionality is the file upload feature, which allows users to upload and process CSV files directly within the chat interface. Message handling is managed through dedicated functions like `handleSend` and `processMessageToAPI`, which facilitate com-

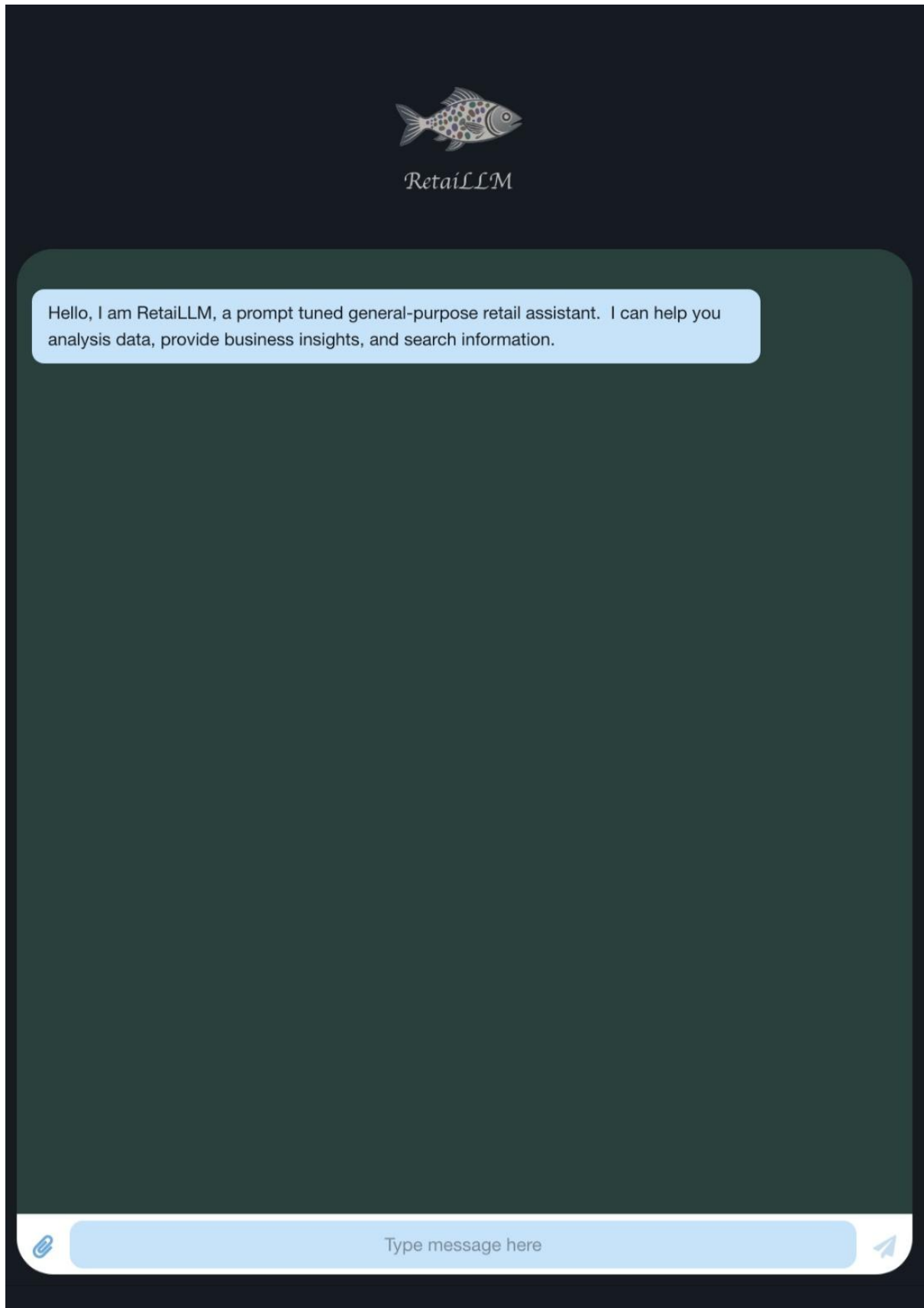


Figure 13: User Interface

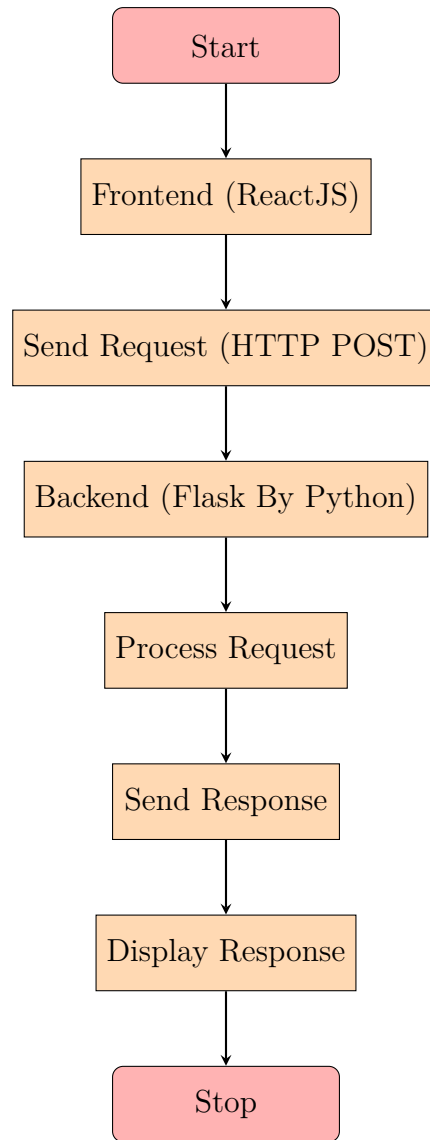


Figure 14: Flowchart of request and response processing with Flask backend and ReactJS frontend

munication with the backend server. Figure 14 illustrates the flow of request and response processing between the Flask backend and the ReactJS frontend.

Table 2: Frontend and Backend Communication Methods

Operation	HTTP Method	Description
Data Communication	POST	Communicate with backend by JSON
	GET	Fetch images from backend

### 3.6.3 Example from calling the API

**Call the API by HTTP POST** The HTTP POST request is made to the URL `http://localhost:5005/chat` using the `curl` command-line tool. The request method

---

## listing 1 HTTP POST REQUEST

---

```
[lable={code:http}]
curl -X POST http://localhost/chat \
-H "Content-Type: application/json" \
-d '{"user_input": "Help me find some information about selling shoes
in US"}'
```

---

## listing 2 JSON Response

---

```
[lable={code:http}]
"response": "Hello! I've analyzed the information you provided about Abbot
's Shoes and the broader context of selling shoes in the US. Here's a
summary tailored to your request:

Selling Shoes in the US: Insights from Abbot's Shoes\n\n**Platform
Overview:**\nAbbot's Shoes is a marketplace designed for buying and
selling pre-owned quality footwear. This platform is particularly
useful for those looking to sell shoes in the US, as it supports
transactions in various currencies.

**Professional Services:** Includes professional cleaning and photography
, enhancing the appeal of your products.\n-

**Set Up Product Pages:** Ensure high-quality photos and compelling
descriptions to attract buyers.\n3.
**Select Apps and Sales Channels:** Utilize Shopify\u2019s App Store for
essential tools and sales channels like Facebook and Instagram.\n4.
**Choose Payment Gateway:** Set up secure payment options through Shopify
\u2019s checkout system.\n5.
**Market Your Store:** Implement effective marketing strategies,
including social media campaigns and SEO tools.\n\n**Resources and
Support:**\n-

"sentiment": "overall sentiment related to this topic: {
'compound_score': 0.997, 'polarity': 0.196, 'subjectivity': 0.46,
'sentiment_category': 'very_positive',
'sentiment_distribution': {'positive': 0.157, 'neutral': 0.839, 'negative
': 0.004}}",

"urls":
URL: https://abbotsshoes.com/pages/sell-to-us?srsltid, https://www.
shopify.com/sell/shoes
}
```

is specified as POST with the `-X POST` option. It includes a header (`-H "Content-Type: application/json"`) to indicate that the content type is JSON. The request body contains a JSON object with a single key-value pair, where the key is `user input` and the value is the query string *"Help me find some information about selling shoes in the US."* This process is outlined in Listing 1.

In response, the server returns a JSON object that provides a detailed summary of selling shoes in the US, specifically from Abbot's Shoes. This summary includes key insights about the platform, which is a marketplace for pre-owned quality footwear. It highlights features such as a wide range of sizes, popular brands, and various selling options, including direct sales, trade-ins, and consignment. Additionally, the response emphasizes the professional services offered by Abbot's Shoes, such as shoe cleaning and photography, as well as its international reach.

The response also includes sentiment analysis regarding the topic of selling shoes in the US. The sentiment analysis provides a compound score reflecting a very positive overall sentiment, along with detailed breakdowns of polarity, subjectivity, and sentiment distribution across positive, neutral, and negative categories. This is detailed in Listing 2.

Lastly, the response includes relevant URLs for further information, including links to Abbot's Shoes' selling page and Shopify's selling shoes page. These resources offer additional support on setting up an online store, marketing strategies, and payment gateway options for selling shoes online, as detailed in Appendix B.

## 4 Result

### 4.1 Comparison of System Capabilities

Table 3 provides a comparison of the capabilities of various retail assistant systems, including OpenAI WebUI, DeepSeek WebUI, QwenV2.5 WebUI, ChatGML WebUI, and our RetailLLM. The features compared include contextual dialogue, result visualization, numerical analysis, web scraping, database knowledge bases, and sentiment score analysis.

The comparison demonstrates that RetailLLM provides comprehensive support for all listed features, making it the most capable retail assistant system currently available.



Table 3: Comparison of Capabilities Across Various Systems (OpenAI: OpenAI WebUI, DeepSeek: DeepSeek WebUI, QwenV2.5: QwenV2.5 WebUI, ChatGML: ChatGML WebUI, RetaiLLM: RetaiLLM System)

Feature	OpenAI	DeepSeek	QwenV2.5	ChatGML	RetaiLLM
Contextual Dialogue	✓	✓	✓	✓	✓
Result Visualization	✗	✗	✗	✗	✓
Numerical Analysis	✓	✗	✗	✓	✓
Web Scrape	✓	✗	✗	✓	✓
DB Knowledge Base	✗	✗	✗	✗	✓
Sentiment Score Analysis	✗	✗	✗	✗	✓

## 4.2 Kernel LLM deployment and fine-tuning results

### 4.2.1 Inference speed: VLLM vs Transformers

We compare the model loading time (i.e., the time required to load models onto GPUs) and the model inference speed (measured in tokens per second) between VLLM framework and the traditional Transformers library. The results shown in Table 4 demonstrate that the VLLM framework significantly accelerates both the model loading time and inference speed, thereby validating the effectiveness of our implementation.

Table 4: Framework Performance Comparison: VLLM vs Transformers Framework (MLT: Model Loading Time, IT: Inference Time).

Framework	MLT (s)	IT (tokens/s)
VLLM	54.02	13.32
Transformers	110.45	6.38

### 4.2.2 Model’s accuracy Testing

We also compare the model’s accuracy on testing set. The accuracy here refers to the model’s ability to correctly determine the method by which the user needs to obtain data. The results shown in Table 5 that models’ accuracy is improved significantly after fine-tuning. However, due to inherent limitations in model capacity, the smaller language model (small LM) still falls short of achieving the performance level of the larger language model (LLM).

Table 5: Model Accuracy Comparison Before and After Fine-tuning (BFT: Before Fine-tuning, AFT: After Fine-tuning). Accuracy is calculated as  $\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}}$ . A dash (-) indicates that the model was not fine-tuned or the metric was unavailable.

Model	Accuracy (BFT)	Accuracy (AFT)
QwenV2.5-72B-instruct	0.91	-
Llama3.2-1B	0.36	0.44
Llama3.2-3B	0.285	0.46

### 4.3 Evaluation of Web Scraping Configurations

We evaluated multiple web scraping configurations and the content scrapped against four key criteria:

1. **Content Structure and Coverage:** Ensuring extracted material is comprehensive and well-organized.
2. **Information Balance:** Combining high-level summaries with detailed insights.
3. **Readability and Formatting:** Maintaining clear, readable text to aid downstream analysis.
4. **Sentiment Equilibrium:** Achieving a well-balanced sentiment profile, avoiding overly positive or negative bias.

Table 6: Comparison of Web Scraping Methods, where **the best results are highlighted in bold** and the second-best results are underlined for clarity.

Method	Speed (s)	Quality	Compound Score	Balanced Sent. Dist.
Google Chrome WebDriver	303.99	<b>Excellent</b>	<b>0.999</b>	<b>0.837</b>
LangChain (No Summary)	<b>19.02</b>	Poor	<b>1.000</b>	0.781
LangChain (With Summary)	55.37	Good	0.996	0.742
Integrated Solution	<u>82.19</u>	<b>Excellent</b>	<b>0.999</b>	<u>0.816</u>

**Analysis:** The Google Chrome WebDriver approach delivers outstanding quality and a positive, balanced sentiment distribution, but its excessive runtime (over 300 seconds) makes it impractical for routine operations. In contrast, the LangChain (No Summary) configuration is exceptionally fast but yields poor content quality, limiting its utility despite strong sentiment scores. Incorporating summarization (LangChain With Summary) improves quality at a moderate speed, but the sentiment balance dips slightly, indicating that while the content is more digestible than the no-summary variant, the overall distribution could still be more even.

The integrated solution effectively balances these trade-offs. It achieves:

- **Good Performance (82.19s):** Significantly faster than the slowest method while still providing reasonable turnaround times.
- **High-Quality Output:** Comparable to the best-performing method in terms of data organization and readability.
- **Robust Sentiment Balance:** A near-ideal compound score (0.999) and a strong sentiment distribution (0.816), indicating that the scraped content neither skews too heavily positive nor lacks positivity altogether.

## 4.4 Visualization Result

In retail data analysis, visualization is not only a means of presenting results but also a crucial tool for uncovering underlying patterns in data and enhancing decision-making quality. This section focuses on showcasing and interpreting the visualization results of various analytical methods applied to retail data, including correlation analysis, regression analysis, clustering analysis, time series analysis, and survival analysis. These visualizations provide an intuitive understanding of the relationships between variables, the performance of predictive models, the distribution of customer segments, as well as trends in future sales and changes in customer retention probabilities. This insight serves as a valuable reference for retailers in strategy formulation and business optimization. Additionally, Appendix G presents further visualization results.

**Correlation Analysis Results Visualization** To elucidate the correlations, heatmaps are generated for each correlation coefficient method, providing a visual representation of the strength and direction of associations between variables. These visualizations aid in swiftly identifying key relationships, such as the impact of advertising expenditure on sales or the influence of customer satisfaction on repurchase rates. By leveraging these insights, retailers can make informed decisions to enhance marketing strategies and customer retention efforts. Fig. 5 illustrates the Pearson Correlation Coefficient analysis in RetailLLM.

**Regression Analysis Results Visualization** Various visualization techniques, including scatter plots of actual versus predicted sales and feature importance charts, are utilized to interpret model performance. These visual tools aid in assessing the accuracy

and reliability of the regression models, guiding data-driven decision-making in retail operations. Fig. 6 and Fig. 7 illustrate the Lasso and Ridge Regression analysis in RetaiLLM respectively.

**Clustering Analysis Results Visualization** Scatter plots with color-coded clusters are generated for each clustering algorithm, illustrating the distribution and separation of customer segments. These visualizations enable retailers to discern distinct customer profiles, facilitating personalized marketing and inventory management strategies. Fig. 8 and Fig. 9 illustrate the KMeans and DBSCAN Clustering analysis in RetaiLLM respectively.

**Time Series Analysis Results Visualization** Forecasting outcomes from both ARIMA and Transformer-based time series models are visualized through line plots juxtaposing historical sales with predicted values. These visualizations facilitate the comparison of model performances and the validation of forecasting accuracy, supporting strategic planning and operational adjustments in retail management. Fig. 10 illustrate the Transformer-Based Time Series analysis in RetaiLLM.

**Survival Analysis Results Visualization** Survival curves generated from both the Kaplan-Meier Estimator and the Cox Proportional Hazards Model offer intuitive visualizations of customer retention dynamics. These plots assist stakeholders in understanding retention probabilities and the influence of key factors on customer longevity. Fig. 11 illustrate the survival analysis based on the Kaplan-Meier Estimator in RetaiLLM.

## 5 Discussion

The RetaiLLM system, presented as a context-aware LLM agent for general purpose retail assistance, demonstrates significant potential in addressing the challenges of dynamic data acquisition, management, and analysis in retail applications. This section discusses the strengths of RetaiLLM and its limitations.

## 5.1 Key Contributions and Strengths

**Modular Architecture** The RetailLLM system features a modular structure where the main class interacts seamlessly with the submodules, which are Kernel LLM, Data Analysis module, Database module, Webscraper module, and WebUI. This modular design ensures flexibility, scalability, and efficient task execution, enabling the system to handle diverse retail-specific workflows.

**Dynamic Prompt Construction** Through Prompt Tree, RetailLLM efficiently manages context-aware prompts for classifying user requests, determining data acquisition methods, and guiding analytical tasks. The hierarchical structure of the Prompt Tree allows the system to dynamically adapt prompts for different tasks, improving LLM outputs' accuracy and flexibility.

**Data Node Design** One of the critical innovative design is the Data Node, which enables multi-turn communication and data reuse. In practical retail workflows, users often engage in follow-up questions or request analyses based on previously provided datasets. The Data Node autonomously tracks and manages active datasets, reducing the need for redundant inputs. By maintaining the contextual continuity using Data Node, the system can be aware of the user's previous input and ensures smoother user experience.

**Comprehensive Webscraper Module** RetailLLM also integrates a comprehensive webscraper module that enables acquisition of up-to-date internet information. This module performs summarization and sentiment analysis on the scraped content, ensuring the retrieved data is not only actionable but also enriched with insights. Such capabilities are particularly valuable for tasks like market research, competitor analysis, and customer sentiment tracking, where understanding and summarizing external data are critical.

**Database Support** RetailLLM includes a robust Database module that allows querying structured data from previously collected data. The system can dynamically select the most relevant data based on the user input and metadata descriptions. After release, we can still update this knowledge base, which will make RetailLLM smarter and smarter.

**Advanced Analytical Module** We formulate an analytical toolkits, known as the previously Data Analysis module. This module has nearly covered all the numerical analysis tasks required by retail, which fulfills the command of general-purpose retail assistance.

## 5.2 Limitations

Despite its strengths, RetaiLLM faces several limitations. Firstly, the Data Node does a good job of managing active datasets, but it has no session persistence. Without the ability to save and reload conversation states, the system cannot support long-term interactions across sessions, which limits its applicability in retail workflows requiring continuity over time.

Additionally, the webscraper module depends on the quality of external data sources. Inconsistent, incomplete, or noisy data may impact the system’s ability to generate reliable insights. Similarly, the Database module assumes that the data is well-structured, which may not always reflect real-world retail data environments.

Finally, the reliance on large-scale language models (LLMs) introduces significant computational costs, particularly for tasks requiring real-time or high-volume interactions. The system’s current dependency on a large kernel LLM (e.g., Qwen2.5-72B) makes deployment resource-intensive, limiting its scalability for widespread use in retail settings.

## 6 Conclusions and Future Work

**Conclusions** In an era of rapid global economic growth, the retail industry serves as a crucial link between producers and consumers, increasingly adopting digital and intelligent solutions to enhance efficiency and customer engagement. This paper presents **RetaiLLM**, the first Large Language Model (LLM) specifically designed for the retail sector. Unlike general-purpose models, RetaiLLM is tailored with retail-specific data and a customized architecture to address the unique challenges of the industry. It integrates diverse data sources, including sales records, inventory data, and customer interactions, to provide accurate and flexible analysis. RetaiLLM’s robust multi-turn dialogue capabilities and autonomous utilization of relevant databases enhance decision-making and operational optimization. Equipped with advanced analytical methods and real-time web scraping with sentiment analysis, RetaiLLM delivers sophisticated insights and actionable recommendations, setting a new standard for LLM applications in retail.

**Future Work** Future developments will focus on enhancing RetaiLLM’s compliance, expanding its functionalities, addressing current limitations, and improving user interactions. Key areas include:

**1. Web Scraping Enhancements:**

- Implement a comprehensive Terms of Service compliance framework.
- Advance content processing with machine learning classification and multi-language support.
- Improve data quality handling by incorporating data cleaning and validation techniques to manage inconsistent or noisy data sources.

**2. LLM Improvements:**

- Expand RetaiLLM’s modules to address more specific retail tasks.
- Incorporate additional datasets for fine-tuning.
- Develop knowledge graphs to strengthen reasoning capabilities.
- Optimize model architecture to reduce computational costs, enabling more efficient deployment in resource-constrained environments.

**3. Advanced Analysis:**

- Introduce region-specific analytical methods.
- Integrate more sophisticated predictive and prescriptive analytics techniques.

**4. Database Optimization:**

- Explore alternative database systems for greater scalability and flexibility.
- Enhance data security measures to protect sensitive retail information.
- Develop capabilities to handle unstructured or semi-structured data, reflecting real-world retail data environments.

**5. User Interface Upgrades:**

- Enhance user interactions with clickable visualizations that display detailed outputs.
- Develop interactive dashboards for customizable data views.
- Implement session persistence mechanisms to support long-term interactions across user sessions.
- Integrate user feedback mechanisms to continuously refine the system.

## 6. General Enhancements:

- Ensure continuous optimization aligned with emerging retail trends.
- Integrate RetailLLM with other enterprise systems like CRM and ERP to create a unified operational ecosystem.

These initiatives aim to solidify RetailLLM’s role as an essential tool for the retail industry, addressing current limitations and driving ongoing innovation to enable retailers to fully leverage intelligent data analysis and natural language processing for superior business outcomes.

## References

- Abdi, H. (2007). The kendall rank correlation coefficient. *Encyclopedia of measurement and statistics*, 2, 508–510.
- Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., ... others (2023). Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.
- Aggarwal, C. C. (2015). Data mining for retail and marketing applications. *Springer Book Series on Data Mining*.
- Ali Abd Al-Hameed, K. (2022). Spearman’s correlation coefficient in statistical analysis. *International Journal of Nonlinear Analysis and Applications*, 13(1), 3249–3255.
- Authors, D. (n.d.).
- Bender, R., Augustin, T., & Blettner, M. (2005). Generating survival times to simulate cox proportional hazards models. *Statistics in medicine*, 24(11), 1713–1723.
- Bharadwaj, N., Ballings, M., Naik, P. A., Moore, M., & Arat, M. M. (2022). A new livestream retail analytics framework to assess the sales impact of emotional displays. *Journal of Marketing*, 86(1), 27–47.
- Bratić, D., Šapina, M., Jurečić, D., & Gršić, J. (2024). Centralized database access: Transformer framework and llm/chatbot integration-based hybrid model. *Applied System Innovation*, 7(1), 17. doi: 10.3390/asi7010017
- Brown, T. B., Mann, B., Ryder, N., et al. (2020). Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33, 1877–1901.
- Chandramana, S. B. (2017). Retail analytics: driving success in retail industry with business analytics.
- Chang, Y., Wang, X., Wang, J., Wu, Y., Yang, L., Zhu, K., ... others (2024). A survey on evaluation of large language models. *ACM Transactions on Intelligent Systems and Technology*, 15(3), 1–45.



- Chen, X., Li, X., & Wu, W. (2019). Deep learning for demand forecasting in retail: A survey and new results. *IEEE Transactions on Neural Networks and Learning Systems*, 30(6), 2005–2016.
- Cohen, I., Huang, Y., Chen, J., Benesty, J., Benesty, J., Chen, J., ... Cohen, I. (2009). Pearson correlation coefficient. *Noise reduction in speech processing*, 1–4.
- Contreras, J., Espinola, R., Nogales, F. J., & Conejo, A. J. (2003). Arima models to predict next-day electricity prices. *IEEE transactions on power systems*, 18(3), 1014–1020.
- Etikan, I., Abubakar, S., & Alkassim, R. (2017). The kaplan-meier estimate in survival analysis. *Biom Biostat Int J*, 5(2), 00128.
- Fan, R.-E., Chen, P.-H., & Lin, C.-J. (2006). Handling high-dimensional data: Svm challenges. *Machine Learning Journal*, 28, 124–139.
- Gao, C., Zheng, Y., Li, N., Li, Y., Qin, Y., Piao, J., ... Li, Y. (2023). A survey of graph neural networks for recommender systems: Challenges, methods, and directions. *ACM Transactions on Recommender Systems*, 1(1), 1–51. (Published: 03 March 2023) doi: 10.1145/3568022
- Goel, M. K., Khanna, P., & Kishore, J. (2010). Understanding survival analysis: Kaplan-meier estimate. *International journal of Ayurveda research*, 1(4), 274.
- Grattafiori, A., et al. (2024). The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*. Retrieved from <https://arxiv.org/abs/2407.21783>
- Hadi, M. U., Qureshi, R., Shah, A., Irfan, M., Zafar, A., Shaikh, M. B., ... others (2023). A survey on large language models: Applications, challenges, limitations, and practical usage. *Authorea Preprints*.
- Hameli, K. (2018). A literature review of retailing sector and business retailing types. *ILIRIA International Review*, 8(1), 67–87.
- Har, L. L., Rashid, U. K., Te Chuan, L., Sen, S. C., & Xia, L. Y. (2022). Revolution of retail industry: from perspective of retail 1.0 to 4.0. *Procedia Computer Science*, 200, 1615–1625.
- Hauke, J., & Kossowski, T. (2011). Comparison of values of pearson’s and spearman’s correlation coefficients on the same sets of data. *Quaestiones geographicae*, 30(2), 87–93.
- Holstein, A., & Jack, W. (2024). Harnessing llms for enhanced sentiment analysis: Unpacking e-commerce customer feedback. *Unpublished Research*. Retrieved from [https://www.researchgate.net/publication/383860951\\_Harnessing\\_LLMs\\_for\\_Enhanced\\_Sentiment\\_Analysis\\_Unpacking\\_E-commerce\\_Customer\\_Feedback](https://www.researchgate.net/publication/383860951_Harnessing_LLMs_for_Enhanced_Sentiment_Analysis_Unpacking_E-commerce_Customer_Feedback) (Accessed: 2024-12-14) doi: 10.13140/RG.2.2.12599.53921
- Khan, K., Rehman, S. U., Aziz, K., Fong, S., & Sarasvady, S. (2014). Dbscan: Past, present and future. In *The fifth international conference on the applications of digital information and web technologies (icadiwt 2014)* (pp. 232–238).

- Kodinariya, T. M., Makwana, P. R., et al. (2013). Review on determining number of cluster in k-means clustering. *International Journal*, 1(6), 90–95.
- Koh, C. E., Kim, H. J., & Kim, E. Y. (2006). The impact of rfid in retail industry: issues and critical success factors. *Journal of Shopping Center Research*, 13(1), 101–117.
- Kohavi, R., Rothleder, N., & Simoudis, E. (2002). Mining e-commerce data: The good, the bad, and the ugly. *Data Mining and Knowledge Discovery*, 4, 445–460.
- Li, X., Wang, Y., & Chen, Z. (2023). Parameter-efficient fine-tuning with adapters: A comprehensive study. In *Proceedings of the 61st annual meeting of the association for computational linguistics* (pp. 345–357).
- Li, Y., Liu, Y., & Yu, M. (2025). Consumer segmentation with large language models. *Journal of Retailing and Consumer Services*, 82, 104078.
- Lin, D. Y., & Wei, L.-J. (1989). The robust inference for the cox proportional hazards model. *Journal of the American statistical Association*, 84(408), 1074–1078.
- Liu, X., Zhang, Y., & Wang, H. (2023). Combining structured and unstructured data for retail intelligence. *Expert Systems with Applications*, 205.
- Marquardt, D. W., & Snee, R. D. (1975). Ridge regression in practice. *The American Statistician*, 29(1), 3–20.
- Murtagh, F., & Contreras, P. (2012). Algorithms for hierarchical clustering: an overview. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 2(1), 86–97.
- Naveed, H., Khan, A. U., Qiu, S., Saqib, M., Anwar, S., Usman, M., ... Mian, A. (2023). A comprehensive overview of large language models. *arXiv preprint arXiv:2307.06435*.
- Nielsen, F., & Nielsen, F. (2016). Hierarchical clustering. *Introduction to HPC with MPI for Data Science*, 195–211.
- OpenAI. (2023). *Openai api*. Retrieved from <https://platform.openai.com/> (Accessed: 2024-12-13)
- Pantano, E. (2014). Innovation drivers in retail industry. *International Journal of Information Management*, 34(3), 344–350.
- Parida, P. R., Ramalingam, S., & Perumalsamy, J. (2024). Large language models in retail crm systems: A technical evaluation of improving customer support, engagement, and sales strategies. *Journal of Artificial Intelligence Research and Applications*, 4(2), 85–130.
- Pribadi, F. A., Firdausi, A. T., & Arhandi, P. P. (2023). Finite state machine in chatbot line e-commerce design using nodejs mongodb and express. *Jurnal Informatika Polinema*, 10(1). doi: 10.33795/jip.v10i1.1501
- Qwen Team. (2024, September). *Qwen2.5: A party of foundation models*. Retrieved from <https://qwenlm.github.io/blog/qwen2.5/>
- Ranstam, J., & Cook, J. A. (2018). Lasso regression. *Journal of British Surgery*, 105(10),

1348–1348.

- Rigatti, S. J. (2017). Random forest. *Journal of Insurance Medicine*, 47(1), 31–39.
- Rooderkerk, R. P., DeHoratius, N., & Musalem, A. (2022). The past, present, and future of retail analytics: Insights from a survey of academic research and interviews with practitioners. *Production and Operations Management*, 31(10), 3727–3748.
- Sachs, A.-L., et al. (2015). Retail analytics. *Lecture notes in economics and mathematical systems*, 680.
- Sansone, M., & Colamatteo, A. (2017). Trends and dynamics in retail industry: focus on relational proximity. *International Business Research*, 10(2), 169.
- Schubert, E., Sander, J., Ester, M., Kriegel, H. P., & Xu, X. (2017). Dbscan revisited, revisited: why and how you should (still) use dbscan. *ACM Transactions on Database Systems (TODS)*, 42(3), 1–21.
- Sedgwick, P. (2012). Pearson’s correlation coefficient. *Bmj*, 345.
- Seetharaman, A., Niranjana, I., Tandon, V., & Saravanan, A. (2016). Impact of big data on the retail industry. *Journal of Corporate Ownership & Control*, 14(1), 506–518.
- Shen, J., Ma, J., & et al. (2023). Vllm: Easy and fast llm serving with pipelined execution. *arXiv preprint arXiv:2307.11330*.
- Shumway, R. H., Stoffer, D. S., Shumway, R. H., & Stoffer, D. S. (2017). Arima models. *Time series analysis and its applications: with R examples*, 75–163.
- Sinaga, K. P., & Yang, M.-S. (2020). Unsupervised k-means clustering algorithm. *IEEE access*, 8, 80716–80727.
- Soni, V. (2023). Large language models for enhancing customer lifecycle management. *Journal of Empirical Social Science Studies*, 7(1), 67–89. Retrieved from <https://publications.dlpress.org/index.php/jesss/article/view/58> (Accessed: 2024-12-14)
- Stepanov, A. (2015). On the kendall correlation coefficient. *arXiv preprint arXiv:1507.01427*.
- Strubell, E., Ganesh, A., & McCallum, A. (2019). Energy and policy considerations for deep learning in nlp. *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 3645–3650.
- Suthaharan, S., & Suthaharan, S. (2016). Support vector machine. *Machine learning models and algorithms for big data classification: thinking with examples for effective learning*, 207–235.
- Tay, Y., Dehghani, M., Bahri, D., et al. (2022). Efficient transformers: A survey. *ACM Computing Surveys*, 55(6), 1–28.
- Tokatli, N., & Boyaci, Y. (1998). The changing retail industry and retail landscapes: The case of post-1980 turkey. *Cities*, 15(5), 345–359.
- Vaja, M. B. R. (2015). Retail management. *International Journal of Research and Analytics Reviews*, 2(1), 22–28.

- Vaswani, A. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*.
- Venkatachalam, D., Sreeram, J., & Soundarapandiyan, R. (2024). Large language models in retail: Best practices for training, personalization, and real-time customer interaction in e-commerce platforms. *Journal of Artificial Intelligence Research and Applications*, 4(1), 539–592.
- Wang, L., Zhang, W., & Zhou, J. (2023). Role-based multi-agent collaboration for large-scale language tasks. *Journal of Artificial Intelligence Research*, 76, 421–442.
- Wang, Y., Zhong, W., Li, L., Mi, F., Zeng, X., Huang, W., ... Liu, Q. (2023). Aligning large language models with human: A survey. *arXiv preprint arXiv:2307.12966*.
- Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., ... others (2020). *Transformers: State-of-the-art natural language processing*. Retrieved from <https://huggingface.co> (Accessed: 2024-12-13)
- Wu, J., Yang, Y., & Zhang, R. (2021). Recursively stacked transformers for complex task collaboration. *Transactions on Artificial Intelligence*, 45, 567–590.
- Wu, L., Zheng, Z., Qiu, Z., Wang, H., Gu, H., Shen, T., ... others (2024). A survey on large language models for recommendation. *World Wide Web*, 27(5), 60.
- Yang, A., Yang, B., Hui, B., Zheng, B., Yu, B., Zhou, C., ... Fan, Z. (2024). Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.
- Yu, W., & Ramanathan, R. (2008). An assessment of operational efficiencies in the uk retail sector. *International Journal of Retail & Distribution Management*, 36(11), 861–882.
- Zhao, J., Liu, W., & Li, H. (2022). Prompt-tuning for efficient adaptation of pre-trained language models. *arXiv preprint arXiv:2203.08191*.
- Zhao, W. X., Zhou, K., Li, J., Tang, T., Wang, X., Hou, Y., ... others (2023). A survey of large language models. *arXiv preprint arXiv:2303.18223*.
- Zhou, W., & Guo, L. (2022). Specialization and adaptability in llm-based agent teams. *Artificial Intelligence Frontiers*, 14, 123–139.
- Zhu, Y., Yuan, H., Wang, S., Liu, J., Liu, W., Deng, C., ... Wen, J.-R. (2023). Large language models for information retrieval: A survey. *arXiv preprint arXiv:2308.07107*.
- Zvarikova, K., Michalikova, K. F., & Rowland, M. (2022). Retail data measurement tools, cognitive artificial intelligence algorithms, and metaverse live shopping analytics in immersive hyper-connected virtual spaces. *Linguistic and Philosophical Investigations*, 21, 9–24.

## A Appendix Overview

This appendix provides comprehensive technical support and supplementary information for the RetailLLM project, covering deployment details, prompt engineering strategies, data processing methods, model architecture and training processes, code examples, and practical application cases. Appendix B details the deployment and access methods for the LLM, including instructions for launching the model with VLLM and client-side connection examples. Appendix C delves into prompt engineering tailored for retail scenarios, including synthetic data generation, user input parsing, multi-turn conversation handling, external data retrieval, database query construction, and guidance for business insights and data analysis. Appendix D presents pseudocode for the DBSCAN clustering algorithm, offering technical reference for data segmentation and clustering tasks. Appendix E explains the Transformer model architecture, training process, core components, and advantages, along with specific applications in this study, such as Transformer-based regression and time series forecasting. Appendix F provides a description of the GitHub code resources. Finally, Appendix G showcases additional conversation examples that demonstrate RetailLLM’s performance in handling more complex interactive scenarios. Together, these sections enable readers to gain deeper insights into and effectively apply RetailLLM’s features and potential in retail contexts.

## B LLM Deployment and Access Details

### B.1 VLLM Command for Deployment

---

**listing 3** Command to Serve Qwen2.5-72B-Instruct Model

---

```
vllm serve Qwen/Qwen2.5-72B-Instruct \
--tensor-parallel-size 8 \
--host 127.0.0.1 \
--port 6006 \
--served-model-name Qwen2.5-72B-Instruct \
--api-key xxx \
--gpu-memory-utilization 0.9 \
--max_model_len 2000 \
--enforce-eager \
--max-num-batched-tokens 8000
```

---

## B.2 Client-side Python code

---

**listing 4** Client-side Python code

---

```
class QwenVllm:
    def __init__(self,
        api_key,
        base_url,
        model_id = "Qwen2.5-72B-Instruct"
    ):
        self.model_id = model_id
        self.api_key = api_key
        self.base_url = base_url
        self.client = OpenAI(
            api_key = self.api_key,
            base_url = self.base_url
        )
    def inference(self, message, max_length=512):
        chat_completion = self.client.chat.completions.create(
            messages=message,
            model=self.model_id,
            max_tokens=max_length
        )
        return chat_completion.choices[0].message.content
```

---

## C Prompts

### C.1 Synthetic Data Generation

---

**listing 5** Generate synthetic data for classifying how to get data

---

Help generate some sythetic data according to following instructions:  
Output in json format: {userinput: , method: }.  
Include some user provided data, like "help me do linear regression x=[1,2,3,4,5], y = [1,2,3,4,5]. Condition: you are a specialized AI assistant focused on retail analysis. Your primary objective is to assist users by providing insights and solutions tailored to their retail-related needs, guided by the system instructions. Based on the user's request, identify the most suitable approach to obtain data and return only one tag from the following list:

Tags:

- USER\_INPUT: The data can be directly extracted from the user's input or the user has specified a given data source enclosed in <data></data>.
- PREVIOUS\_CONVO\_DATA: It seems that the data required is already available from the previous conversation.
- WEB\_SCRAPPE: The user's requests need additional web scraping to retrieve external textual data, if the user does not state clearly that we should perform numerical analysis, choose this tag.
- DB\_QUERY: The user;s requeests need additional query to a database to retrieve structured data.
- NO\_DATA: No data is required, only provide a response based on the user's input.
- MENU: The user is asking how the assistant can help them.

User Request: {user\_input}

The "method" should only include single, most appropriate tag from the list above (["USER\_INPUT", "PREVIOUS\_CONVO\_DATA", "WEB\_SCRAPPE", "DB\_QUERY", "NO\_DATA", MENU]).

---

## C.2 RetaiLLM Prompts

### C.2.1 Determine Get Data Methods

---

**listing 6** prompts for how to get data

---

```
"ClassifyingTasks": {
  "prompt": f"""
    Based on the user's request, identify the most suitable approach
    to obtain data and return only one tag from the following list:

    Tags:
    - USER_INPUT: The data can be directly extracted from the user's
    input or the user has specified a given data source enclosed in <data
    ></data>.
    - PREVIOUS_CONVO_DATA: It seems that the data required is already
    available from the previous conversation.
    - WEB_SCRAPING: The user's requests need additional web scraping to
    retrieve external textual data, if the user does not state clearly
    that we should perform numerical analysis, choose this tag.
    - DB_QUERY: The user's requests need additional query to a
    database to retrieve structured data.
    - NO_DATA: No data is required, only provide a response based on
    the user's input.
    - MENU: The user is asking how the assistant can help them.

    User Request: {user_input}

    Output only the single, most appropriate tag from the list above
    (["USER_INPUT", "PREVIOUS_CONVO_DATA", "WEB_SCRAPING", "DB_QUERY", "
    NO_DATA", "MENU"]).
    """,

  "options": ["USER_INPUT", "PREVIOUS_CONVO_DATA", "WEB_SCRAPING", "
  DB_QUERY", "NO_DATA", "MENU"],
  "max_length": 10,
  "model": "Qwen/Qwen2.5-72B-Instruct"
}
```

---



## C.2.2 Prompts for Extract Data from the User Input

---

**listing 7** Prompts for extract data from the user input

---

```
"Extract_from_user_input": {
  "prompt": f"""
    The data can be directly extracted from the user's input text.
    Your task is to extract attributes and their corresponding
    values** from the given user input and return them in the following
    JSON format**:
      {"attribute1": ["value1", "value2", ...], "attribute2": ["value1",
    "value2", ...], ...}
    **Please follow these guidelines**:
    - **Attribute Keys**: Use the exact attribute names as they
    appear in the user input.
    - **Values**: Collect all values for each attribute into a list,
    preserving their original order.
    - **Data Types**: If a value is numeric (integer or float),
    represent it as a number; otherwise, keep it as a string.
    - **Consistency**: Ensure all lists are of the same length. If an
    attribute is missing a value, use 'null' (without quotes) to
    represent it.
    - **Output Format**: Do not include any additional text,
    explanations, or formatting. Return only the JSON object.
    **Example**:
    User Input:
    I want to analysis the information of some customers.
    Name: Alice, Bob, Charlie
    Age: 30, 25, 35
    City: New York, Los Angeles, Chicago
    Expected Output: {"Name": ["Alice", "Bob", "Charlie"], "Age":
    [30, 25, 35], "City": ["New York", "Los Angeles", "Chicago"]}
    Now, based on the following user input, only output the extracted
    json object:
    **User Input**:
    {user_input}
    """,
  "max_length": 512, "options": None
  "model": "Qwen/Qwen2.5-72B-Instruct"
}
```

---

### C.2.3 Prompts for Handle Previous Conversation Data

---

**listing 8** Prompts for handle previous conversation data

---

```
"PREVIOUS_CONVO_DATA": {
  "choose_from_previous": {
    "prompt": f"""
      The required data is available from the previous conversation
      .
      Select the most appropriate data based on the user input
      provided below.
      The data options, including their names and descriptions, are
      as follows:
      {datalist}
      User input: {user_input}
      Return only the name of the most appropriate data from this
      list: {[data["name"] for data in datalist]}.
      """,
    "options": [data["name"] for data in datalist],
    "max_length": 10,
    "model": "Qwen/Qwen2.5-72B-Instruct"
  },
  "extract_from_previous": {
    "prompt": f"""
      Generate a description of the possible data provided by the
      user based on the user input.
      User input: {user_input}
      Data information:
      {data_info}
      return in the format: {"data_name": name of data, "
      description": description of the data}.
      """,
    "options": None,
    "max_length": 64,
    "model": "Qwen/Qwen2.5-72B-Instruct"
  }
}
```

---

## C.2.4 Prompts for Webscraping

---

**listing 9** Prompts for Webscraping

---

```
"WEB_SCRAP":{
  "generate_web_scrape_keys":{
    "prompt": f"""
      Based on the user's request, extract the most relevant
      keywords for a web scraping task to retrieve external textual data.
      The keywords should be concise and directly related to the
      user's input.
      Now process the following request:
      User Request: {user_input}
      Provide the keywords as a concise list of strings.
      """,
    "options": None, "max_length": 50,
    "model": "Qwen/Qwen2.5-72B-Instruct"
  }
}
```

---

## C.2.5 Prompts for DB query

---

**listing 10** Prompts for DB query

---

```
"DB_QUERY":{
  "choose_table":{
    "prompt": f"""
      Based on the user's request, choose the most relevant tables
      based on the descriptions of tables.
      The table options, including their names and descriptions, are as
      follows:
      {datalist}
      Now process the following request:
      User input: {user_input}
      Return only the name of the most appropriate data from this list:
      {[data["name"] for data in datalist]}. """,
    "options": [data["name"] for data in datalist], "max_length": 50,
    "model": "Qwen/Qwen2.5-72B-Instruct"
  }
}
```

---

## C.2.6 Prompts for Insights Generation

---

**listing 11** Prompts for insights generation

---

```
"response":{
  "prompt": f"""
    provide some insights based on the user's request and the data
    summary provided.
    Summary: {summary}
    And the user is asking about: {user_input}
    Your response should be no more than {response_return_length} words.
  """ ,
  "options": None,
  "max_length": response_return_length + 100,
  "model": "Qwen/Qwen2.5-72B-Instruct"
}
```

## C.2.7 Prompts for Data analysis

Below, we present prompts for data analysis.

```
analysis_tags_list = [
    "Time_series_forecast",
    "Correlation_Spearman",
    "Correlation_Pearson",
    "Correlation_Kendall",
    "Regression_Ridge",
    "Regression_Lasso",
    "Regression_RandomForest",
    "Clustering_KMeans",
    "Clustering_DBSCAN",
    "Clustering_Hierarchical",
    "Kaplan_Meier",
    "Cox_Proportional_Hazards"
]

"ClassifyDataTools":{
  "prompt": f"""
    ###Instructions:
    1. Review the user's request and the data information provided.
    2. Identify the predictor (independent) variable(s) and the
    target (dependent) variable(s) in the data.
    3. Select the most relevant analysis method by choosing the
    appropriate tag from the list below.
```

4. Provide only the method tag, the predictor variable(s), and the target variable(s) in the format:

```
{{"method": "method_tag", "predictor": ["predictor1", "predictor2", ...], "target": ["target1", "target2", ...]}}
```

###Method Tags:

- Time\_series\_forecast: Time Series Analysis model, you should output this tag if you think users requires time series analysis.
- Correlation\_Spearman: Correlation Analysis using Spearman's rank correlation coefficient.
- Correlation\_Pearson: Correlation Analysis using Pearson's correlation coefficient.
- Correlation\_Kendall: Correlation Analysis using Kendall's tau coefficient.
- Regression\_Ridge: Regression Analysis using Ridge Regression.
- Regression\_Lasso: Regression Analysis using Lasso Regression.
- Regression\_RandomForest: Regression Analysis using Random Forest.
- Clustering\_KMeans: Clustering Analysis using K-Means Clustering .
- Clustering\_DBSCAN: Clustering Analysis using DBSCAN Clustering.
- Clustering\_Hierarchical: Clustering Analysis using Hierarchical Clustering.
- Kaplan\_Meier: Survival Analysis using Kaplan-Meier method.
- Cox\_Proportional\_Hazards: Survival Analysis using Cox Proportional Hazards model.

### Examples:

#### Example 1:

**\*\*Input:\*\***

User Request: "Forecast the next 6 months of sales based on historical data."

Data Summary: {{"

```
  "description": "sales and date",
  "attribute names": ["Sales", "Date"],
  "dtypes": [timestamp, int64],
  "shape": (36, 2),
  "data_head": {{"
    "Sales": [500, 600, 550, 650, 780],
```

```

        "Date": ["2021-01-01", "2021-01-02", "2021-01-03",
"2021-01-04", "2021-01-05"]
    }}
}}

**Output:**
{"method": "TimeSeries_ARIMA", "predictor": ["Sales"], "target":
["Sales"]}

#### Example 2:

**Input:**
User Request: "Analyze the relationship between advertising spend
and sales."
Data Summary: {{
    "description": "sales and advertising data",
    "attribute names": ["Sales", "Advertising"],
    "dtypes": [int64, int64],
    "shape": (128, 2),
    "data_head": {{
        "Sales": [500, 600, 550, 650, 780],
        "Advertising": [100, 120, 110, 130, 150]
    }}
}}

**Output:**
{"method": "Correlation_Spearman", "predictor": ["Advertising"],
"target": ["Sales"]}

#### Example 3:

**Input:**
User Request: "Segment customers based on their annual spending
and purchase frequency."
Data Summary: {{
    "description": "customer segmentation data",
    "attribute names": ["CustomerID", "Spending", "Frequency"],
    "dtypes": [int64, int64, int64],
    "shape": (200, 3),
    "data_head": {{
        "CustomerID": [1, 2, 3, 4, 5],

```

```

        "Spending": [500, 600, 550, 650, 780],
        "Frequency": [2, 3, 2, 4, 5]
    }}
}}

**Output:**
{"method": "Clustering_KMeans", "predictor": ["Spending", "
Frequency"], "target": ["CustomerID"]}}

### Task:

User Request: {user_input}

Data Summary: {data_info}

**Provide**
- method tag: one of the tags from the list {analysis_tags_list}
- predictor variables: list of predictor variables from {
data_info["attribute names"]}
- target variables: list of target variables from {data_info["
attribute names"]}
"",
"options": analysis_tags_list,
"max_length": 50,
"model": "Qwen/Qwen2.5-72B-Instruct",
}
}

```

## D DBSCAN Clustering Pseudocode

---

**Algorithm 1** DBSCAN Algorithm

---

```
1: Input:
2:    $D$ : Dataset
3:    $\epsilon$ : radius for neighborhood search
4:   min_samples: minimum points required to form a cluster
5: Initialize
6:   visited  $\leftarrow$  False for all points in  $D$ 
7:   labels  $\leftarrow$  None for all points in  $D$ 
8: Iterate over each point
9: for each point  $p \in D$  do
10:   if  $p$  is not visited then
11:     Mark  $p$  as visited
12:      $N_p \leftarrow \text{RangeQuery}(p, \epsilon)$ 
13:     if  $|N_p| \geq \text{min\_samples}$  then
14:       Create a new cluster and add  $p$  to it
15:       ExpandCluster( $p, N_p$ )
16:     else
17:       Label  $p$  as noise
18:     end if
19:   end if
20: end for
21: Cluster Expansion
22: function EXPANDCLUSTER( $p, N_p$ )
23:   for each point  $q \in N_p$  do
24:     if  $q$  is not visited then
25:       Mark  $q$  as visited
26:        $N_q \leftarrow \text{RangeQuery}(q, \epsilon)$ 
27:       if  $|N_q| \geq \text{min\_samples}$  then
28:          $N_p \leftarrow N_p \cup N_q$ 
29:       end if
30:     end if
31:     if  $q$  is not yet in any cluster then
32:       Add  $q$  to the current cluster
33:     end if
34:   end for
35: end function
```

---



## E Transformer Model Architecture

The Transformer model, pivotal in both Regression Analysis and Time Series Analysis, is detailed herein to avoid repetitive descriptions in the main sections.

### E.1 Overview

The Transformer architecture is renowned for its efficacy in handling sequential data through self-attention mechanisms, enabling the modeling of long-range dependencies without relying on recurrent structures. Its parallelizable design facilitates efficient training on large datasets.

### E.2 Components

- **Input Linear Layer:** Transforms input features into a higher-dimensional embedding space, enhancing feature representation capabilities.
- **Positional Encoding:** Injects information about the position of each element in the sequence, preserving the order of data points essential for temporal analysis.
- **Transformer Encoder Layers:** Consist of multi-head self-attention mechanisms and position-wise feedforward neural networks. These layers capture complex patterns and interactions within the data.
- **Decoder Layer (if applicable):** Maps the encoded representations to the desired output space, such as forecasted sales values.

### E.3 Training Process

The Transformer model is trained by minimizing the Mean Squared Error (MSE) loss between predicted and actual values. Key hyperparameters, including the number of encoder layers, number of attention heads, learning rate, and batch size, are optimized through cross-validation to enhance model performance. Additionally, techniques such as early stopping are employed to prevent overfitting, ensuring the model generalizes well to unseen data.

## E.4 Advantages

- **Long-Range Dependency Modeling:** Capable of capturing dependencies across extensive sequences, making it suitable for complex temporal patterns.
- **Parallelization:** Enables efficient computation and faster training times compared to recurrent models.
- **Scalability:** Easily scales to large datasets, accommodating the growing volume of retail data.

## E.5 Applications in This Study

### E.5.1 Transformer-Based Regression

In Regression Analysis, the Transformer-based regression model leverages self-attention mechanisms to capture intricate feature interactions and dependencies that traditional regression models might overlook. By processing the entire feature set simultaneously, the model can identify complex nonlinear relationships between predictors and the target variable, enhancing prediction accuracy.

### E.5.2 Transformer-Based Time Series Model

In Time Series Analysis, the Transformer-based model is specifically adapted to handle sequential data by incorporating temporal embeddings and possibly utilizing encoder-decoder architectures tailored for forecasting tasks. This adaptation allows the model to effectively capture temporal dependencies and patterns over extended periods, making it highly effective for predicting future sales based on historical trends.

## F    **GitHub Repository for RetaiLLM**

The RetaiLLM project’s source code and related resources are available on GitHub for further exploration and implementation. The repository includes deployment scripts, prompt engineering examples, data preprocessing methods, and model architecture details. Additionally, the GitHub repository contains a comprehensive README and well-commented code, providing clear guidance for users to replicate the RetaiLLM setup, test its functionalities, and customize it to suit specific retail scenarios. Visit the repository at: <https://github.com/IILKA/RetaiLLM/>.

## G    **Additional RetaiLLM Conversation Examples**

To further demonstrate the capabilities and applications of RetaiLLM, we present additional conversation examples generated during our study. These examples showcase the versatility and robustness of RetaiLLM in handling diverse scenarios related to retail.



RetailLLM

Hello, I am RetailLLM, a prompt tuned general-purpose retail assistant. I can help you analysis data, provide business insights, and search information.

Explain how to make a good retail business

Creating a successful retail business requires a combination of strategic planning, customer-centric approaches, and operational efficiency. Here are key steps to build a strong retail business:

### 1. **Market Research and Niche Identification**

- **Understand Your Market:** Conduct thorough market research to identify trends, customer preferences, and competitive landscapes.
- **Define Your Niche:** Focus on a specific niche or target audience to differentiate your business and offer specialized products or services.

### 2. **Business Plan Development**

- **Clear Vision and Goals:** Outline your business vision, mission, and long-term goals.
- **Financial Projections:** Create detailed financial forecasts, including startup costs, revenue projections, and break-even analysis.
- **Operational Plan:** Define your operational structure, including inventory management, staffing, and logistics.

### 3. **Branding and Identity**

- **Unique Value Proposition (UVP):** Clearly articulate what makes your business unique and why customers should choose you over competitors.
- **Branding Elements:** Develop a strong brand identity through a compelling logo, consistent messaging, and a cohesive visual style.

### 4. **Location and Store Layout**

- **Strategic Location:** Choose a location that is accessible and attracts your target audience. Consider factors like foot traffic, visibility, and competition.
- **Store Layout:** Design an appealing and functional store layout that enhances the shopping experience and encourages customer engagement.



Type message here



Figure 15: Demonstration of Basic Conversation



RetaiLLM

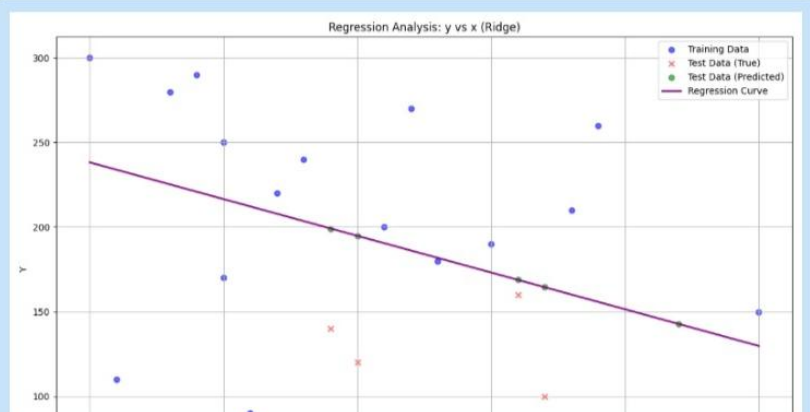
Hello, I am RetaiLLM, a prompt tuned general-purpose retail assistant. I can help you analysis data, provide business insights, and search information.

help me perform linear ridge regression

```
y = [  
120, 190, 250, 70, 300,  
150, 220, 180, 100, 280,  
270, 140, 60, 210, 90,  
290, 160, 200, 50, 260,  
110, 240, 80, 170  
]  
  
x = [  
15, 20, 10, 25, 5,  
30, 12, 18, 22, 8,  
17, 14, 19, 23, 11,  
9, 21, 16, 27, 24,  
6, 13, 28, 10  
]
```

Ridge Model - Intercept: 259.9485, Coefficients: {'x': -4.338883447600392} A regression visualization plot has been generated.

The Ridge regression model suggests a negative relationship between x and y.



Type message here

Figure 16: Demonstration of Regression Analysis



RetailLLM

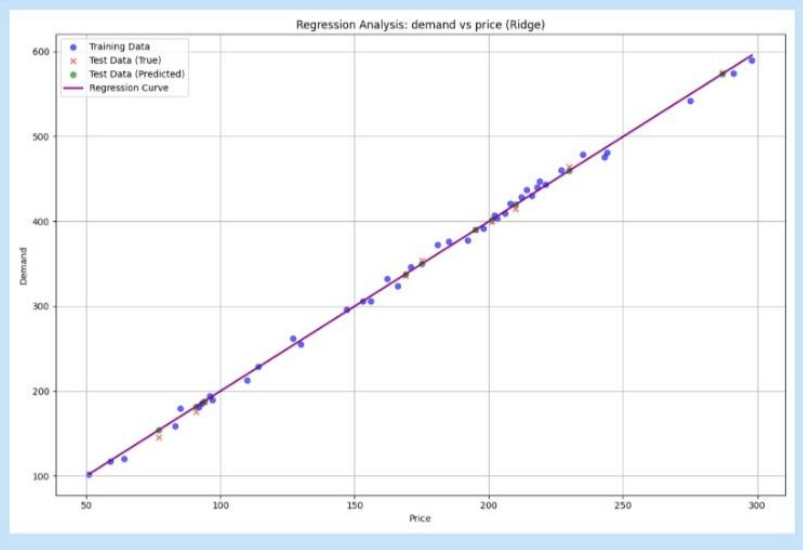
Hello, I am RetailLLM, a prompt tuned general-purpose retail assistant. I can help you analysis data, provide business insights, and search information.

help me perform linear ridge regression on price and demand

[147, 206, 244, 208, 291, 153, 235, 85, 92, 298, 275, 110, 218, 201, 221, 59, 64, 94, 166, 91, 202, 51, 83, 96, 130, 210, 175, 181, 214, 216, 77, 93, 169, 171, 114, 162, 212, 192, 227, 287, 185, 97, 198, 243, 203, 230, 156, 219, 195, 127]  
[296, 409, 481, 421, 574, 306, 479, 180, 181, 590, 542, 213, 440, 399, 443, 117, 120, 187, 324, 176, 407, 102, 159, 194, 255, 415, 354, 372, 437, 430, 146, 186, 336, 346, 229, 332, 429, 378, 460, 575, 376, 190, 392, 476, 403, 464, 306, 447, 390, 262]

Ridge Model - Intercept: 0.2449, Coefficients: {'price': 1.9979841495045068} A regression visualization plot has been generated.

The Ridge regression shows a positive relationship between price and demand.



Type message here



Figure 17: Demonstration of Regression Analysis



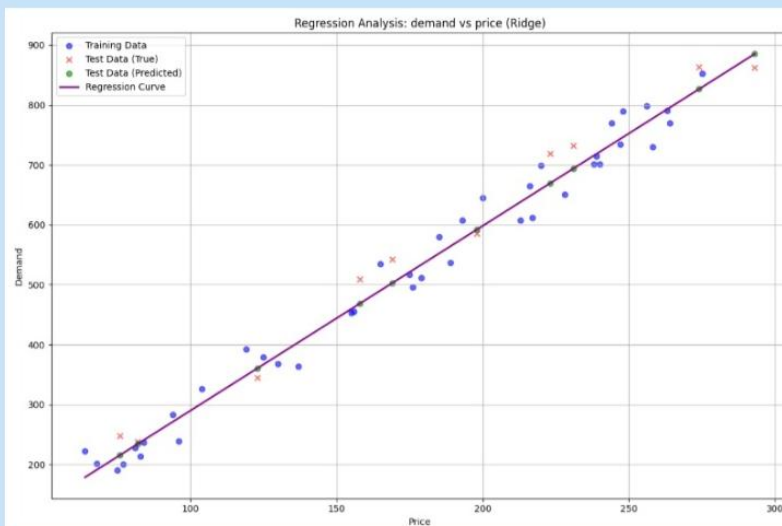
RetailLLM

Hello, I am RetailLLM, a prompt tuned general-purpose retail assistant. I can help you analysis data, provide business insights, and search information.

You have uploaded a CSV file successfully!  
help me perform linear ridge regression on price and demand

Ridge Model - Intercept: -18.8292, Coefficients: {'price': 3.0856029254431685} A regression visualization plot has been generated.

The Ridge regression shows a positive relationship between price and demand.



Type message here



Figure 18: Demonstration of Regression Analysis

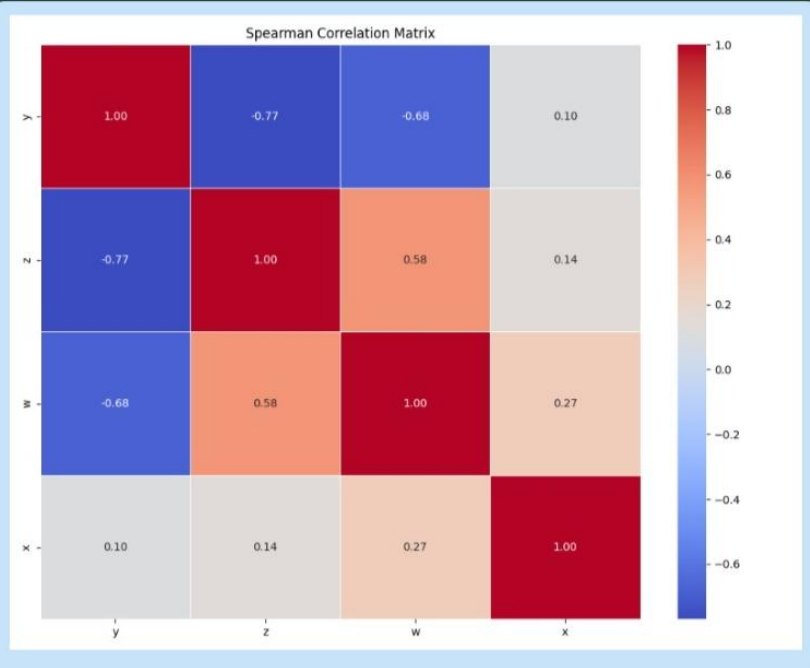


RetailLLM

Hello, I am RetailLLM, a prompt tuned general-purpose retail assistant. I can help you analysis data, provide business insights, and search information.

Help me perform a correlation test on, sales, discount, Trends, and Advertising.x: [70, 99, 81, 16, 18, 51, 13, 20, 61, 82]y: [51, 74, 61, 65, 92, 67, 63, 64, 81, 86]z: [148, 95, 134, 117, 86, 124, 132, 96, 90, 127]w: [49, 39, 48, 10, 19, 37, 40, 23, 35, 17]

Highest Spearman correlation is between y and z with a coefficient of -0.77. A correlation heatmap visualization plot has been generated.  
The correlation test shows a strong negative relationship between sales and discounts, with a coefficient of -0.77.



Type message here



Figure 19: Demonstration of Correlation Analysis





RetaiLLM

Hello, I am RetaiLLM, a prompt tuned general-purpose retail assistant. I can help you analysis data, provide business insights, and search information.

predict the next sale value, sales:[177, 106, 150, 67, 52, 166, 165, 54, 151, 76, 21, 110, 133, 60, 47, 174, 80, 46, 190, 114, 102, 147, 98, 57, 78, 135, 184, 109, 155, 28]

Selected Forecasting Method: Transformer. Forecasted values:

[2020-01-31: 48.91]

[2020-02-01: 70.66]

[2020-02-02: 106.70]

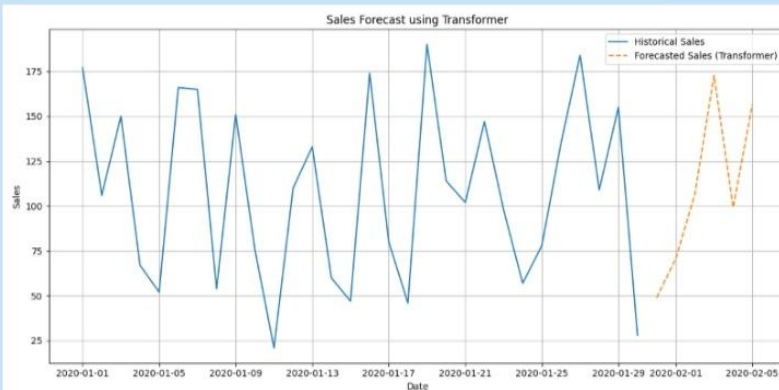
[2020-02-03: 172.65]

[2020-02-04: 99.22]

[2020-02-05: 156.50]

A time series forecast visualization plot has been generated.

The next predicted sale value is **\*\*162.35\*\***.



Type message here



Figure 20: Demonstration of Time Series Analysis



RetailLLM

Hello, I am RetailLLM, a prompt tuned general-purpose retail assistant. I can help you analysis data, provide business insights, and search information.

You have uploaded a CSV file successfully!  
predict the next sale value

Selected Forecasting Method: Transformer. Forecasted values:

[2020-02-10: 105.59]

[2020-02-11: 125.51]

[2020-02-12: 140.05]

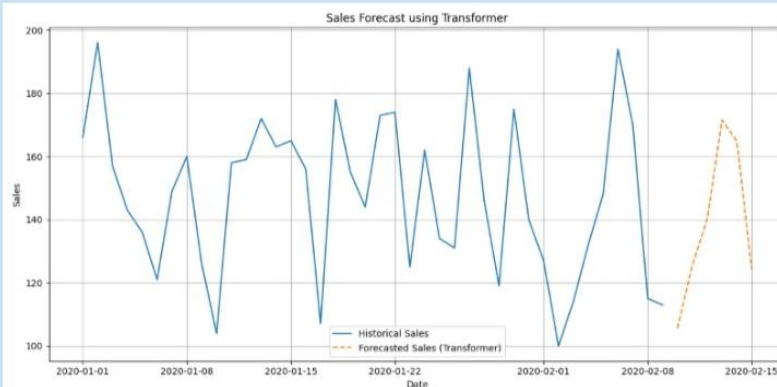
[2020-02-13: 171.51]

[2020-02-14: 164.81]

[2020-02-15: 124.24]

A time series forecast visualization plot has been generated.

The next sale value is predicted to be approximately 160.



Type message here



Figure 21: Demonstration of Time Series Analysis



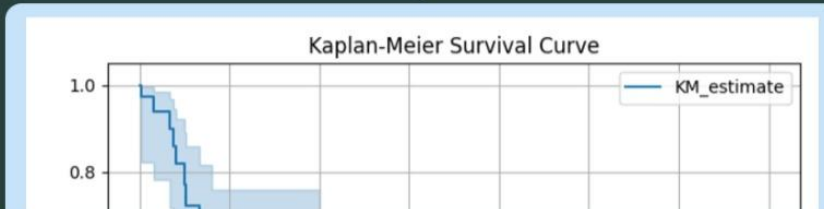
RetailLLM

Hello, I am RetailLLM, a prompt tuned general-purpose retail assistant. I can help you analysis data, provide business insights, and search information.

Conducting Survival Analysis on Product 3

```
 durations = [  
 8, 14, 270, 6, 19,  
 23, 11, 17, 9, 25,  
 15, 4, 20, 12, 22,  
 7, 100, 24, 180, 13,  
 5, 211, 26, 3, 29,  
 2, 28, 1, 30, 31,  
 32, 33, 344, 350, 36,  
 37, 40  
 ]  
 event_observed = [  
 1, 0, 1, 0, 1,  
 0, 0, 1, 0, 1,  
 0, 0, 1, 0, 0,  
 0, 1, 0, 1, 0,  
 0, 0, 1, 0, 0,  
 0, 0, 1, 0, 0,  
 0, 1, 1, 1, 0,  
 0, 1  
 ]
```

Median survival time: 100.00 months.  
Median survival time: 25.00 months.



Type message here



Figure 22: Demonstration of Survival Analysis



Median survival time: 100.00 months.  
Median survival time: 25.00 months.



75