**PRODUCT DEMAND FORECASTING**

*Dissertation submitted in fulfilment of the requirements for the Degree of*

**BACHELOR OF TECHNOLOGY**

**in**

**COMPUTER SCIENCE AND ENGINEERING**

By

**SHAIK JULFEEN AHMADH**

**1211O554**

Supervisor

**VED PRAKASH CHAUBEY**



**School of Computer Science and Engineering**

Lovely Professional University

Phagwara, Punjab (India)

Month…………… Year ………

@ Copyright LOVELY PROFESSIONAL UNIVERSITY, Punjab (INDIA)

Month ….., Year …..

ALL RIGHTS RESERVED

**DECLARATION STATEMENT**

I hereby declare that the research work reported in the dissertation/dissertation proposal entitled "PRODUCT DEMAND FORECASTING in partial fulfilment of the requirement for the award of Degree for Bachelor of Technology in Computer Science and Engineering at Lovely Professional University, Phagwara, Punjab is an authentic work carried out under supervision of my research supervisor Mr. Ved Prakash Chaubey. I have not submitted this work elsewhere for any degree or diploma.

I understand that the work presented herewith is in direct compliance with Lovely Professional University’s Policy on plagiarism, intellectual property rights, and highest standards of moral and ethical conduct. Therefore, to the best of my knowledge, the content of this dissertation represents authentic and honest research effort conducted, in its entirety, by me. I am fully responsible for the contents of my dissertation work.

*Signature of Candidate*

**Shaik Julfeen Ahmadh**

**12110554**

**SUPERVISOR’S CERTIFICATE**

This is to certify that the work reported in the B.Tech Dissertation/dissertation proposal entitled “**PRODUCT DEMAND FORECASTING”**, submitted by **Shaik Julfeen Ahmadh** at **Lovely Professional University, Phagwara, India** is a bonafide record of his / her original work carried out under my supervision. This work has not been submitted elsewhere for any other degree.

Signature of Supervisor

(Name of Supervisor)

**Date:**

**Counter Signed by:**

1. **Concerned HOD:**

HoD’s Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

HoD Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Date: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. **Neutral Examiners:**

**External Examiner**

Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Affiliation: \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Date: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Internal Examiner**

Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Date: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**ABSTRACT:**

Forecasting product demand plays a pivotal role in optimizing inventory management and meeting customer needs. This study explores and compares three distinct time series models for demand forecasting: ARIMA, SARIMA, and LSTM. Autoregressive Integrated Moving Average (ARIMA) models the temporal dependencies in the data, while Seasonal ARIMA (SARIMA) incorporates seasonal trends for improved accuracy. Long Short-Term Memory (LSTM), a type of recurrent neural network, captures complex sequential patterns. Through rigorous analysis and evaluation, this research investigates the effectiveness of these models in predicting product demand. Real-world datasets are utilized to assess the predictive capabilities, considering various factors such as seasonality, trend, and irregularities in demand patterns. Insights derived from this comparative analysis aim to guide practitioners and decision-makers in selecting the most suitable model for accurate and efficient product demand forecasting, thereby facilitating informed decision-making and resource allocation.

**Key Words:** ARIMA, SARIMA and LSTM

|  |
| --- |
| **CONTENT** |
| Cover Page |
| Declaration Statement |
| Supervisor’s Certificate |
| Abstract |
| **1.INTRODUCTION** |
| 1.1 Motivation |
| 1.2 Problem Statement |
| 1.3 Objective of the Project |
| 1.4 Scope |
| 1.5 Project Introduction |
| **2.LTERATURE SURVEY** |
| 2.1 Related Work |
| **3. SYSTEM ANALYSIS** |
| 3.1 Existing System |
| 3.2 Disadvantages |
| 3.3 Proposed System |
| 3.4 Advantages |
| 3.5 Work Flow of Proposed system |
| **4. REQUIREMENT ANALYSIS** |
| 4.1Function and non-functional requirements |
| 4.2 Hardware Requirements |
| **5. CONCLUSION** |
| **6. FUTURE ENHANCEMENT** |
| **7. REFERENCES** |

1. **INTRODUCTION**

**1.1 Motivation:**

Effective demand forecasting is crucial for businesses to optimize their inventory and meet customer demands efficiently. However, the selection of the right forecasting model remains a challenge. This study aims to address this issue by rigorously comparing ARIMA, SARIMA, and LSTM models, offering valuable insights into their effectiveness in predicting product demand. Such insights are vital for businesses seeking to enhance their forecasting accuracy and streamline resource allocation.

**1.2 Problem Statement:**

The variability and complexity of product demand patterns pose a significant challenge for businesses in accurately predicting future demands. Existing forecasting methods like ARIMA and SARIMA have limitations in handling intricate seasonal and irregular patterns, while LSTM, although promising, lacks comprehensive comparative analysis in real-world demand forecasting scenarios. This study seeks to investigate the strengths and weaknesses of these models, addressing the need for a thorough comparative evaluation to guide businesses in selecting the most suitable forecasting approach.

**1.3 Objective of the Project:**

The primary objective is to conduct a comprehensive comparative analysis of ARIMA, SARIMA, LSTM , RF , Regression models for product demand forecasting. This involves evaluating their predictive accuracy, particularly in capturing seasonal, trend, and irregular demand patterns using real-world datasets. By scrutinizing these models' performances, the project aims to offer insights into their effectiveness and limitations, aiding practitioners and decision-makers in choosing the most appropriate model for accurate and efficient demand forecasting.

**1.4 Scope:**

This project encompasses the utilization of historical demand data across various industries, considering diverse product categories and market segments. The focus lies in assessing the models' performances in handling different levels of seasonality, trends, and irregularities in demand patterns. Real-world datasets will be employed to ensure the practical applicability of the findings. However, this study does not delve into other forecasting methodologies beyond the specified ARIMA, SARIMA, LSTM, RF, Regression models, XGB, DT, KNN, SVM, GBM.

**1.5 Project Introduction:**

In the landscape of modern commerce, the ability to predict product demand stands as a cornerstone for successful inventory management and meeting customer expectations. Accurate forecasting techniques enable businesses to streamline their operations, optimize inventory levels, and ensure timely fulfillment of customer needs. Within this context, this study embarks on a comprehensive exploration and comparative analysis of three prominent time series models: Autoregressive Integrated Moving Average (ARIMA), Seasonal ARIMA (SARIMA), and Long Short-Term Memory (LSTM). Demand forecasting involves unraveling the intricate temporal dependencies inherent in data, encompassing seasonality, trends, and irregular patterns. ARIMA, a well-established statistical model, offers a framework for capturing temporal dependencies by modeling the relationships between observations and their lagged values. Its extension, SARIMA, further integrates seasonal variations, enhancing accuracy by accounting for seasonal trends that significantly impact demand fluctuations.

In contrast, LSTM, a form of recurrent neural network, diverges from traditional statistical methods by leveraging its ability to capture complex sequential patterns. With its capacity to retain and learn from long-term dependencies in data, LSTM holds promise in modeling nonlinear and intricate temporal relationships that may elude more conventional models like ARIMA and SARIMA. This research undertakes a rigorous evaluation of these three models using real-world datasets, presenting a comparative analysis of their predictive capabilities. Through meticulous assessment, encompassing various facets such as seasonality, trends, and irregular demand patterns, this study aims to discern the strengths and limitations of each model. By providing empirical insights derived from this comparative analysis, the goal is to furnish practitioners and decision-makers with guidance in selecting the most suitable forecasting model. Such guidance becomes instrumental in facilitating informed decision-making, optimizing resource allocation, and ultimately enhancing efficiency in addressing product demand variability. The convergence of statistical rigor, advanced machine learning techniques, and real-world applicability forms the bedrock of this study, intending to pave the way for more accurate, reliable, and efficient product demand forecasting methodologies.

1. **LITERATURE REVIEW**

**2.1 Related Work:**

**[1]**  **Arif, M. A. I., Sany, S. I., Nahin, F. I., & Rabby, A. S. A. (2019). Comparison Study: Product Demand Forecasting with Machine Learning for Shop. 2019 8th International Conference System Modeling and Advancement in Research Trends (SMART). doi:10.1109/smart46866.2019.91173.**

Mobile app distribution platforms such as Google Play Store allow

users to share their feedback about downloaded apps in the form

of a review comment and a corresponding star rating. Typically,

the star rating ranges from one to ve stars, with one star denoting

a high sense of dissatisfaction with the app and ve stars denoting

a high sense of satisfaction.

Unfortunately, due to a variety of reasons, oen the star rating

provided by a user is inconsistent with the opinion expressed in

the review. For example, consider the following review for the

Facebook App on Android; “Awesome App”. One would reasonably

expect the rating for this review to be ve stars, but the actual

rating is one star!

Such inconsistent ratings can lead to a deated (or inated)

overall average rating of an app which can aect user downloads,

as typically users look at the average star ratings while making a

decision on downloading an app. Also, the app developers receive

a biased feedback about the application that does not represent

ground reality. is is especially signicant for small apps with a

few thousand downloads as even a small number of mismatched

reviews can bring down the average rating drastically

Mobile app distribution platforms such as Google Play Store allow

users to share their feedback about downloaded apps in the form

of a review comment and a corresponding star rating. Typically,

the star rating ranges from one to ve stars, with one star denoting

a high sense of dissatisfaction with the app and ve stars denoting

a high sense of satisfaction.

Unfortunately, due to a variety of reasons, oen the star rating

provided by a user is inconsistent with the opinion expressed in

the review. For example, consider the following review for the

Facebook App on Android; “Awesome App”. One would reasonably

expect the rating for this review to be ve stars, but the actual

rating is one star!

Such inconsistent ratings can lead to a deated (or inated)

overall average rating of an app which can aect user downloads,

as typically users look at the average star ratings while making a

decision on downloading an app. Also, the app developers receive

a biased feedback about the application that does not represent

ground reality. is is especially signicant for small apps with a

few thousand downloads as even a small number of mismatched

reviews can bring down the average rating drastically

Mobile app distribution platforms such as Google Play Store allow

users to share their feedback about downloaded apps in the form

of a review comment and a corresponding star rating. Typically,

the star rating ranges from one to ve stars, with one star denoting

a high sense of dissatisfaction with the app and ve stars denoting

a high sense of satisfaction.

Unfortunately, due to a variety of reasons, oen the star rating

provided by a user is inconsistent with the opinion expressed in

the review. For example, consider the following review for the

Facebook App on Android; “Awesome App”. One would reasonably

expect the rating for this review to be ve stars, but the actual

rating is one star!

Such inconsistent ratings can lead to a deated (or inated)

overall average rating of an app which can aect user downloads,

as typically users look at the average star ratings while making a

decision on downloading an app. Also, the app developers receive

a biased feedback about the application that does not represent

ground reality. is is especially signicant for small apps with a

few thousand downloads as even a small number of mismatched

reviews can bring down the average rating drasticall

Mobile app distribution platforms such as Google Play Store allow

users to share their feedback about downloaded apps in the form

of a review comment and a corresponding star rating. Typically,

the star rating ranges from one to ve stars, with one star denoting

a high sense of dissatisfaction with the app and ve stars denoting

a high sense of satisfaction.

Unfortunately, due to a variety of reasons, oen the star rating

provided by a user is inconsistent with the opinion expressed in

the review. For example, consider the following review for the

Facebook App on Android; “Awesome App”. One would reasonably

expect the rating for this review to be ve stars, but the actual

rating is one star!

Such inconsistent ratings can lead to a deated (or inated)

overall average rating of an app which can aect user downloads,

as typically users look at the average star ratings while making a

decision on downloading an app. Also, the app developers receive

a biased feedback about the application that does not represent

ground reality. is is especially signicant for small apps with a

few thousand downloads as even a small number of mismatched

reviews can bring down the average rating drasticall

Mobile app distribution platforms such as Google Play Store allow

users to share their feedback about downloaded apps in the form

of a review comment and a corresponding star rating. Typically,

the star rating ranges from one to ve stars, with one star denoting

a high sense of dissatisfaction with the app and ve stars denoting

a high sense of satisfaction.

Unfortunately, due to a variety of reasons, oen the star rating

provided by a user is inconsistent with the opinion expressed in

the review. For example, consider the following review for the

Facebook App on Android; “Awesome App”. One would reasonably

expect the rating for this review to be ve stars, but the actual

rating is one star!

Such inconsistent ratings can lead to a deated (or inated)

overall average rating of an app which can aect user downloads,

as typically users look at the average star ratings while making a

decision on downloading an app. Also, the app developers receive

a biased feedback about the application that does not represent

ground reality. is is especially signicant for small apps with a

few thousand downloads as even a small number of mismatched

reviews can bring down the average rating drasticall

Mobile app distribution platforms such as Google Play Store allow

users to share their feedback about downloaded apps in the form

of a review comment and a corresponding star rating. Typically,

the star rating ranges from one to ve stars, with one star denoting

a high sense of dissatisfaction with the app and ve stars denoting

a high sense of satisfaction.

Unfortunately, due to a variety of reasons, oen the star rating

provided by a user is inconsistent with the opinion expressed in

the review. For example, consider the following review for the

Facebook App on Android; “Awesome App”. One would reasonably

expect the rating for this review to be ve stars, but the actual

rating is one star!

Such inconsistent ratings can lead to a deated (or inated)

overall average rating of an app which can aect user downloads,

as typically users look at the average star ratings while making a

decision on downloading an app. Also, the app developers receive

a biased feedback about the application that does not represent

ground reality. is is especially signicant for small apps with a

few thousand downloads as even a small number of mismatched

reviews can bring down the average rating drasticall

Mobile app distribution platforms such as Google Play Store allow

users to share their feedback about downloaded apps in the form

of a review comment and a corresponding star rating. Typically,

the star rating ranges from one to ve stars, with one star denoting

a high sense of dissatisfaction with the app and ve stars denoting

a high sense of satisfaction.

Unfortunately, due to a variety of reasons, oen the star rating

provided by a user is inconsistent with the opinion expressed in

the review. For example, consider the following review for the

Facebook App on Android; “Awesome App”. One would reasonably

expect the rating for this review to be ve stars, but the actual

rating is one star!

Such inconsistent ratings can lead to a deated (or inated)

overall average rating of an app which can aect user downloads,

as typically users look at the average star ratings while making a

decision on downloading an app. Also, the app developers receive

a biased feedback about the application that does not represent

ground reality. is is especially signicant for small apps with a

few thousand downloads as even a small number of mismatched

reviews can bring down the average rating drasticall

Mobile app distribution platforms such as Google Play Store allow

users to share their feedback about downloaded apps in the form

of a review comment and a corresponding star rating. Typically,

the star rating ranges from one to ve stars, with one star denoting

a high sense of dissatisfaction with the app and ve stars denoting

a high sense of satisfaction.

Unfortunately, due to a variety of reasons, oen the star rating

provided by a user is inconsistent with the opinion expressed in

the review. For example, consider the following review for the

Facebook App on Android; “Awesome App”. One would reasonably

expect the rating for this review to be ve stars, but the actual

rating is one star!

Such inconsistent ratings can lead to a deated (or inated)

overall average rating of an app which can aect user downloads,

as typically users look at the average star ratings while making a

decision on downloading an app. Also, the app developers receive

a biased feedback about the application that does not represent

ground reality. is is especially signicant for small apps with a

few thousand downloads as even a small number of mismatched

reviews can bring down the average rating drasticall

Mobile app distribution platforms such as Google Play Store allow

users to share their feedback about downloaded apps in the form

of a review comment and a corresponding star rating. Typically,

the star rating ranges from one to ve stars, with one star denoting

a high sense of dissatisfaction with the app and ve stars denoting

a high sense of satisfaction.

Unfortunately, due to a variety of reasons, oen the star rating

provided by a user is inconsistent with the opinion expressed in

the review. For example, consider the following review for the

Facebook App on Android; “Awesome App”. One would reasonably

expect the rating for this review to be ve stars, but the actual

rating is one star!

Such inconsistent ratings can lead to a deated (or inated)

overall average rating of an app which can aect user downloads,

as typically users look at the average star ratings while making a

decision on downloading an app. Also, the app developers receive

a biased feedback about the application that does not represent

ground reality. is is especially signicant for small apps with a

few thousand downloads as even a small number of mismatched

reviews can bring down the average rating drasticall

Mobile app distribution platforms such as Google Play Store allow

users to share their feedback about downloaded apps in the form

of a review comment and a corresponding star rating. Typically,

the star rating ranges from one to ve stars, with one star denoting

a high sense of dissatisfaction with the app and ve stars denoting

a high sense of satisfaction.

Unfortunately, due to a variety of reasons, oen the star rating

provided by a user is inconsistent with the opinion expressed in

the review. For example, consider the following review for the

Facebook App on Android; “Awesome App”. One would reasonably

expect the rating for this review to be ve stars, but the actual

rating is one star!

Such inconsistent ratings can lead to a deated (or inated)

overall average rating of an app which can aect user downloads,

as typically users look at the average star ratings while making a

decision on downloading an app. Also, the app developers receive

a biased feedback about the application that does not represent

ground reality. is is especially signicant for small apps with a

few thousand downloads as even a small number of mismatched

reviews can bring down the average rating drasticall

Mobile app distribution platforms such as Google Play Store allow

users to share their feedback about downloaded apps in the form

of a review comment and a corresponding star rating. Typically,

the star rating ranges from one to ve stars, with one star denoting

a high sense of dissatisfaction with the app and ve stars denoting

a high sense of satisfaction.

Unfortunately, due to a variety of reasons, oen the star rating

provided by a user is inconsistent with the opinion expressed in

the review. For example, consider the following review for the

Facebook App on Android; “Awesome App”. One would reasonably

expect the rating for this review to be ve stars, but the actual

rating is one star!

Such inconsistent ratings can lead to a deated (or inated)

overall average rating of an app which can aect user downloads,

as typically users look at the average star ratings while making a

decision on downloading an app. Also, the app developers receive

a biased feedback about the application that does not represent

ground reality. is is especially signicant for small apps with a

few thousand downloads as even a small number of mismatched

reviews can bring down the average rating drasticall

Mobile app distribution platforms such as Google Play Store allow

users to share their feedback about downloaded apps in the form

of a review comment and a corresponding star rating. Typically,

the star rating ranges from one to ve stars, with one star denoting

a high sense of dissatisfaction with the app and ve stars denoting

a high sense of satisfaction.

Unfortunately, due to a variety of reasons, oen the star rating

provided by a user is inconsistent with the opinion expressed in

the review. For example, consider the following review for the

Facebook App on Android; “Awesome App”. One would reasonably

expect the rating for this review to be ve stars, but the actual

rating is one star!

Such inconsistent ratings can lead to a deated (or inated)

overall average rating of an app which can aect user downloads,

as typically users look at the average star ratings while making a

decision on downloading an app. Also, the app developers receive

a biased feedback about the application that does not represent

ground reality. is is especially signicant for small apps with a

few thousand downloads as even a small number of mismatched

reviews can bring down the average rating drasticall

Mobile app distribution platforms such as Google Play Store allow

users to share their feedback about downloaded apps in the form

of a review comment and a corresponding star rating. Typically,

the star rating ranges from one to ve stars, with one star denoting

a high sense of dissatisfaction with the app and ve stars denoting

a high sense of satisfaction.

Unfortunately, due to a variety of reasons, oen the star rating

provided by a user is inconsistent with the opinion expressed in

the review. For example, consider the following review for the

Facebook App on Android; “Awesome App”. One would reasonably

expect the rating for this review to be ve stars, but the actual

rating is one star!

Such inconsistent ratings can lead to a deated (or inated)

overall average rating of an app which can aect user downloads,

as typically users look at the average star ratings while making a

decision on downloading an app. Also, the app developers receive

a biased feedback about the application that does not represent

ground reality. is is especially signicant for small apps with a

few thousand downloads as even a small number of mismatched

reviews can bring down the average rating drasticall

In the retail landscape, leveraging machine learning for product demand forecasting has become indispensable. By harnessing historical sales data, market trends, seasonality, and various external factors, ML models can predict future demand with greater accuracy. These models employ algorithms like regression, time series analysis, or neural networks to analyze intricate patterns and correlations within the data. This forecasting aids shops in optimizing inventory levels, ensuring products are available when needed while minimizing overstocking or shortages. Additionally, ML-driven forecasting allows for proactive decision-making, enabling businesses to adapt swiftly to changing consumer behaviors and market dynamics. By continuously learning from new data, these models enhance their predictive capabilities, empowering shops to streamline operations, enhance customer satisfaction, and ultimately maximize profitability through informed inventory management strategies.

**[2] D. Grewal, A. L. Roggeveen, and J. Nordfalt, The Future of Retailing, Journal of Retailing, vol. 93, no. 1, pp. 16, 2017.**

Is a topic of immense interest as the industry undergoes dynamic shifts. The Journal of Retailing frequently explores this evolving landscape. The future holds a blend of physical and digital realms, where seamless omnichannel experiences redefine customer engagement. Technologies like AI, AR/VR, and machine learning personalize interactions, offering tailored recommendations and immersive shopping environments. Sustainability and ethical consumption continue to influence consumer choices, prompting retailers to adopt eco-friendly practices and transparent supply chains. Additionally, the rise of experiential retail transforms stores into engaging spaces, emphasizing memorable experiences over mere transactions. Social commerce gains traction, leveraging social media platforms for direct sales. Moreover, advancements in logistics and last-mile delivery reshape convenience and speed of service. The future of retail lies in innovation, adaptation, and a deep understanding of evolving consumer behaviors, integrating technology and human-centric strategies for a seamless, customer-focused experience.

**[3]** **Zeynep Hilal Kilimci, A. Okay Akyuz, Mitat Uysal, An Improved Demand Forecasting Model Using Deep Learning Approach and Proposed Decision Integration Strategy for Supply Chain, Journal of Hindawi, vol. 2019, no. 1, pp. 1-15, 2019.**

environments and requirements. Tracking and understanding

changes in modules and relationships in a software project is

difﬁcult, but even more so when the software goes through several

types of changes. The typical complexity and size of software

also makes it harder to grasp software evolution patterns. In

this paper, we present an interactive matrix-based visualization

technique that, combined with animation, depicts how software

designs evolve. For example, it shows which new modules

and couplings are added and removed over time. Our generic

visualization supports dynamic and weighted digraphs and is

applied in the context of software evolution. Analyzing source

code changes is important to determine the software’s structural

organization and identify quality issues over time. To demonstrate

our approach, we explore open-source repositories and discuss

some of our ﬁndings regarding these evolving software designs

The application of deep learning in demand forecasting has revolutionized supply chain management. By leveraging complex neural networks, this advanced approach provides a more accurate prediction of future demand patterns. Deep learning models, such as recurrent neural networks (RNNs) or long short-term memory networks (LSTMs), excel at capturing intricate dependencies in data sequences, making them adept at handling the volatility and non-linearity often seen in demand trends. The proposed decision integration strategy enhances this forecasting model by seamlessly integrating the predicted demand into supply chain decisions. This strategy ensures an agile response to changing demands, optimizing inventory management, production scheduling, and resource allocation. By linking the forecasting model with decision-making processes, companies can achieve greater adaptability and responsiveness, reducing stockouts, minimizing excess inventory, and ultimately enhancing customer satisfaction. The synergy between a robust deep learning forecasting model and an integrated decision strategy marks a significant leap forward in optimizing supply chain operations.

**[4]** **Majed Kharfan, Vicky Wing Kei Chan, Forecasting Seasonal Footwear Demand Using Machine Learning, Publisher Massachusetts Institute of Technology, 2018.**

Forecasting seasonal footwear demand through machine learning involves using historical sales data, market trends, and seasonality patterns to predict future consumer demand for different types of footwear. Various machine learning models such as regression, time series analysis, and neural networks can be employed to analyze past sales data, considering factors like weather, fashion trends, and economic indicators. These models can identify patterns and correlations within the data, enabling accurate predictions for future demand. For instance, they might recognize increased demand for sandals during summer months or boots during the winter season. Leveraging this predictive analysis allows footwear manufacturers and retailers to optimize production, manage inventory efficiently, plan marketing campaigns effectively, and meet consumer demand proactively. By harnessing machine learning algorithms, companies can make data-driven decisions, mitigate risks of overstocking or understocking, and ultimately enhance customer satisfaction by ensuring the availability of the right footwear at the right time.

**[5] M. A. A. Hasin, Shuvo Ghosh, and Mahmud A. Shareef, An ANN Approach to Demand Forecasting in Retail Trade in Bangladesh, International Journal of Trade, Economics and Finance, Vol. 2, No. 2, pp. 154-160, April 2011.**

In the realm of retail trade, demand forecasting plays a pivotal role in optimizing inventory, managing resources, and enhancing overall operational efficiency. A study conducted on "An Artificial Neural Network (ANN) Approach to Demand Forecasting in Retail Trade in Bangladesh" was published in the International Journal of Trade. This research likely delved into the application of artificial neural networks—a machine learning technique—tailored for the specific context of Bangladesh's retail market. Such an approach likely involved leveraging historical sales data, market trends, and various influencing factors to train the neural network model. By utilizing ANN methodologies, the study aimed to improve the accuracy of demand predictions, offering retailers in Bangladesh valuable insights into consumer behavior, seasonal fluctuations, and demand patterns. Such forecasting techniques are pivotal for businesses, aiding in strategic decision-making, efficient inventory management, and meeting customer demands promptly, thereby potentially contributing significantly to the retail sector's growth and sustainability in Bangladesh.

**3. SYSTEM ANALYSIS**

**3.1 Existing System**

The existing system for product demand forecasting typically relies on historical data analysis, statistical models, and sometimes manual inputs from experts. It involves methods like time series analysis, regression models, and machine learning algorithms to predict future demand. However, this approach might face challenges with accuracy due to sudden market shifts, unpredictable events, or limited data representation. Integrating advanced technologies and real-time data could enhance the precision of forecasting models in the existing system.

**3.2 Disadvantages:**

1. **Dependency on Historical Data**: Existing systems heavily rely on past trends, making them less adaptable to sudden market changes.
2. **Inaccuracy with Unforeseen Events:** Unexpected occurrences like pandemics can disrupt predictions, rendering forecasts unreliable.
3. **Limited Adaptability**: These systems struggle to incorporate real-time data or quickly adjust to evolving market dynamics.
4. **Complexity in Data Interpretation**: Interpreting vast amounts of data can be challenging, leading to potential errors in forecasting.
5. **Expertise Dependency:** Some methods require specialized knowledge, making it difficult for non-experts to use these systems effectively.

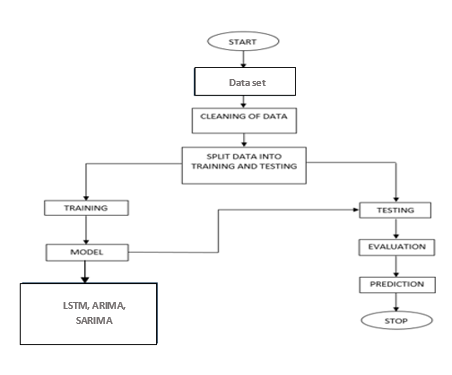
**3.3 Proposed System**

This study rigorously examines ARIMA, SARIMA, and LSTM models for product demand forecasting. ARIMA captures temporal dependencies, SARIMA incorporates seasonal trends, while LSTM handles complex sequential patterns. Real-world datasets are used to evaluate predictive capabilities, considering seasonality, trends, and irregular demand patterns. Insights derived from this comparative analysis aid decision-makers in selecting the optimal model, enhancing accurate demand forecasting, informed decision-making, and efficient resource allocation..

**3.4 Advantages:**

1. **ARIMA:** Simplicity in application, interpretable results, handles stationary data well, forecasts short-term trends, and accommodates various data distributions effectively.
2. **SARIMA:** Captures seasonal variations, flexibility in modeling complex seasonal patterns, integrates trend and seasonality, adaptable to changing data patterns, and robust performance in long-term forecasts.
3. **LSTM:** Handles intricate temporal dependencies, adept at capturing nonlinear relationships, accommodates variable sequence lengths, powerful in learning complex patterns, and performs well with large datasets.
4. **Random Forest:** Ensemble method for robust predictions, capturing nonlinear relationships and feature importance.
5. **Regression Models:** Linear/Polynomial Regression offers interpretability and efficiency in modeling linear relationships and baseline comparisons.
6. **XGBoost:** Boosting algorithm with high predictive power and efficiency, suitable for large datasets.
7. **Decision Tree:** Simple yet effective for capturing nonlinear relationships and feature importance, intuitive interpretation.
8. **KNN:** Non-parametric method for pattern recognition, suitable for clustering and classification tasks.
9. **SVM:** Effective for high-dimensional data, finds optimal hyperplane for classification, versatile with different kernel functions.
10. **GBM:** Gradient boosting algorithm with strong predictive performance and robustness against overfitting, suitable for complex datasets.

**3.5 Work Flow of Proposed system**



**4. REQUIREMENT ANALYSIS**

**4.1 Functional and non-functional requirements**

Requirement’s analysis is very critical process that enables the success of a system or software project to be assessed. Requirements are generally split into two types: Functional and non-functional requirements.

**Functional Requirements**: These are the requirements that the end user specifically demands as basic facilities that the system should offer. All these functionalities need to be necessarily incorporated into the system as a part of the contract. These are represented or stated in the form of input to be given to the system, the operation performed and the output expected. They are basically the requirements stated by the user which one can see directly in the final product, unlike the non-functional requirements.

Examples of functional requirements:

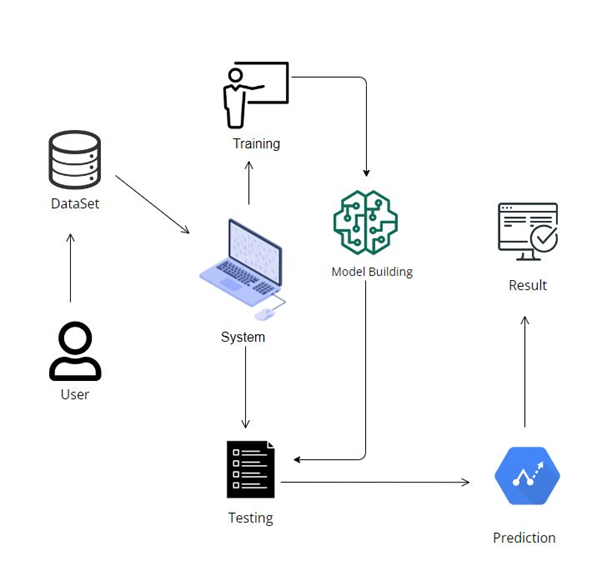
1. Authentication of user whenever he/she logs into the system
2. System shutdown in case of a cyber-attack
3. A verification email is sent to user whenever he/she register for the first time on some software system.

**Non-functional requirements**: These are basically the quality constraints that the system must satisfy according to the project contract. The priority or extent to which these factors are implemented varies from one project to other. They are also called non-behavioral requirements.  
They basically deal with issues like:

* Portability
* Security
* Maintainability
* Reliability
* Scalability
* Performance
* Reusability
* Flexibility

Examples of non-functional requirements:

1. Emails should be sent with a latency of no greater than 12 hours from such an activity.
2. The processing of each request should be done within 10 seconds
3. The site should load in 3 seconds whenever of simultaneous users are > 10000
   1. **Architecture:**

****

**5. IMPLEMENTATION AND RESULTS**

**5.1 Modules:**

**Take Dataset:**

The dataset for the Historical Product Demand Data is collected from the kaggle website (kaggle.com).

The size of overall dataset is 48.8 MB (5,12,57,344 bytes)

**Pre-processing:**

* In preprocessing first of all we will check whether there is any Nan values.
* If any Nan values is present we will fill the Nan values with different fillna techniques like bfill, ffill, mode, and mean.
* Here we used the ffill (front fill) technique on our project.

**Training the data:**

Irrespective of the algorithm we select the training is the same for every algorithm**.**

Given a dataset we split the data into two parts training and testing, the reason behind doing this is to test our model/algorithm performance just like the exams for a student the testing is also exam for the model.

We can split data into anything we want but it is just good practice to split the data such that the training has more data than the testing data, we generally split the data.

And for training and testing there are two variables X and Y in each of them, the X is the features that we use to predict the Y target and same for the testing also.

Then we call the .fit ( ) method on any given algorithm which takes two parameters i.e., X and Y for calculating the math and after that when we call the .predict ( ) giving our testing X as parameter and checking it with the accuracy score giving the testing Y and predicted X as the two parameters will get our accuracy score and same steps , these are just checking for how good our model performed on a given dataset.

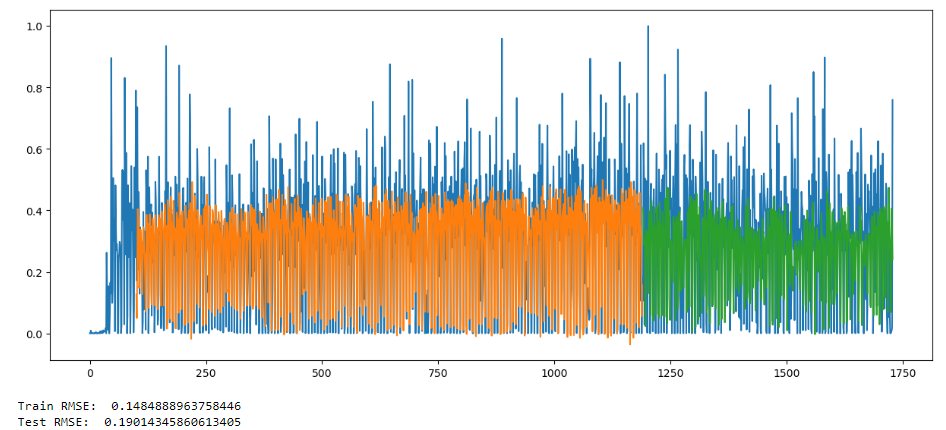
**METHODOLOGY AND ALGORITHMS:**

**1.LSTM:**

**Long Short-Term Memory (LSTM)** is a type of recurrent neural network (RNN) architecture designed to handle the challenges of learning long-term dependencies in sequential data. Unlike traditional RNNs, LSTM networks are equipped with specialized memory cells that can maintain information over extended sequences, making them particularly effective in tasks involving time series, natural language processing, speech recognition, and more.

The key components of an LSTM unit include the cell state, input gate, forget gate, and output gate. These components work in harmony to control the flow of information within the network. The cell state acts as a conveyor belt, allowing information to pass through while the gates regulate the flow of information that enters or exits the cell state. The input gate manages the flow of new information into the cell state, the forget gate decides what information to discard from the cell state, and the output gate determines the output based on the updated cell state. These mechanisms enable LSTMs to selectively remember or forget information, which helps in addressing the vanishing or exploding gradient problems encountered in training traditional RNNs. By preserving long-term dependencies and mitigating the issues of information loss over sequences, LSTMs have become a cornerstone in various applications. They excel in tasks that involve understanding context from a large body of text, generating coherent sequences, predicting future events in time series data, and more. Their ability to capture and retain essential information over extended periods makes LSTMs a robust choice for a wide array of sequential data problems, contributing significantly to advancements in machine learning and artificial intelligence.

Comparing original & predicted data:



Predictions by LSTM:

A blue line graph with white text

Description automatically generated

**2.ARIMA:**

The ARIMA (AutoRegressive Integrated Moving Average) algorithm is a widely used time series forecasting method known for its effectiveness in capturing and predicting trends and patterns within data. It's a powerful tool in analyzing and forecasting various types of time-series data, from stock prices and sales forecasting to weather patterns.

ARIMA models consist of three key components:

**AutoRegression (AR):** This component refers to the relationship between a variable and its own past values. It captures the linear relationship between an observation and a certain number of lagged observations (autoregressive terms).

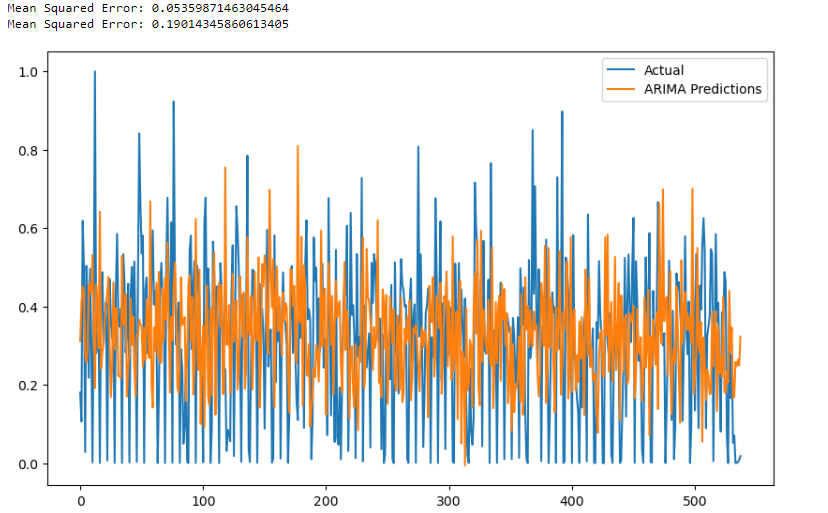
**Integration (I):** This component involves differencing the raw observations to make the time series stationary. Stationarity is crucial for ARIMA models, as they perform better on stationary data.

**Moving Average (MA):** This component accounts for the relationship between an observation and a residual error from a moving average model applied to lagged observations.

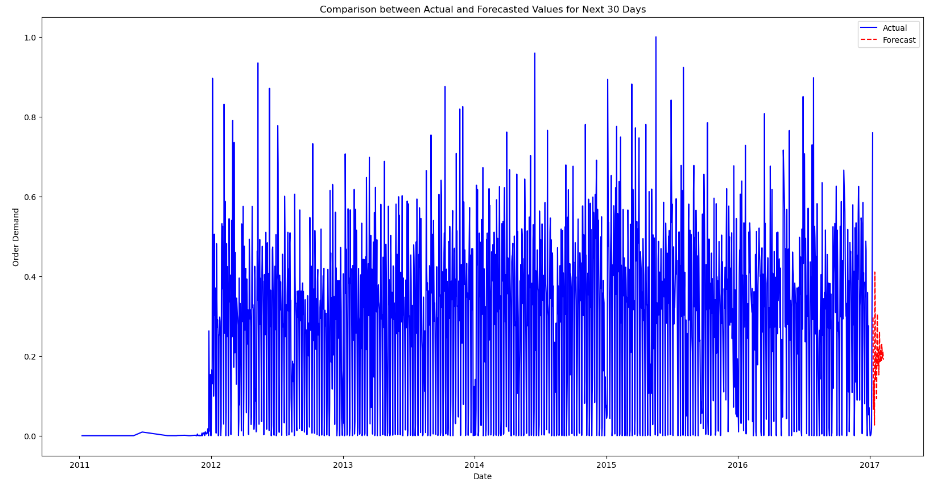
ARIMA models are defined by three parameters: p (AR), d (I), and q (MA), denoting the number of autoregressive terms, the degree of differencing, and the number of moving average terms, respectively.

The process of building an ARIMA model involves identifying the optimal values for p, d, and q through methods like grid search or using information criteria like AIC (Akaike Information Criterion) and BIC (Bayesian Information Criterion). Once fitted, the model can be used to forecast future values based on historical patterns extracted from the data.

Comparing original & predicted data:



Predictions by ARIMA:



**3.SARIMA :**

Seasonal Autoregressive Integrated Moving Average (SARIMA) is a powerful time series forecasting algorithm that extends the capabilities of ARIMA (Autoregressive Integrated Moving Average) by incorporating seasonality. SARIMA models are designed to handle time series data with both trend and seasonal components, making them suitable for predicting patterns that exhibit regular seasonal fluctuations.

The SARIMA model comprises several components:

**Seasonal:** Captures repeating patterns over fixed intervals, accounting for seasonality in the data.

**Autoregressive (AR):** Utilizes past values from the series to predict future values, considering linear regression of the series against its lagged values.

**Integrated (I):** Represents the differencing of raw observations to make the series stationary, eliminating trends.

**Moving Average (MA):** Utilizes past forecast errors to predict future values, capturing the relationship between the error term and observations.

The SARIMA model parameters involve three sets: (p, d, q), (P, D, Q), and the seasonal period S.

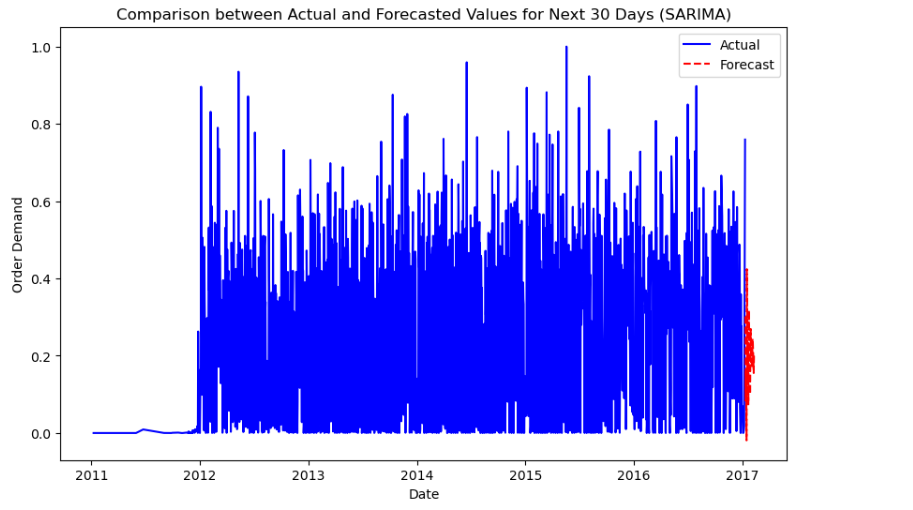
(p, d, q) denote non-seasonal components.

(P, D, Q) refer to seasonal components.

Choosing the appropriate parameters involves analyzing the autocorrelation and partial autocorrelation functions, along with data transformations to achieve stationarity.

SARIMA models are implemented in various programming languages and statistical software, providing accurate forecasts for a wide range of time series data, including economics, finance, and weather patterns. Its ability to capture both short-term fluctuations and longer-term trends makes it a valuable tool in predictive analytics.

Predictions by SARIMA:



**4.RF:**

Random Forest is an ensemble learning method based on decision trees. It builds multiple decision trees during training and outputs the mode of the classes (for classification) or the mean prediction (for regression) of the individual trees.

Each tree is constructed using a random subset of the features and a random subset of the training data, introducing diversity among the trees. During prediction, the output of all trees is aggregated to make the final prediction, which improves robustness and reduces overfitting. Random Forest is effective in handling high-dimensional data, nonlinear relationships, and interactions between features.

It is widely used for classification and regression tasks in various domains due to its simplicity, scalability, and high performance.

A computer screen shot of a computer code

Description automatically generated

**5.REGRESSION Models:**

Linear Regression and Polynomial Regression are classic statistical methods used for modelling the relationship between one or more independent variables and a dependent variable.

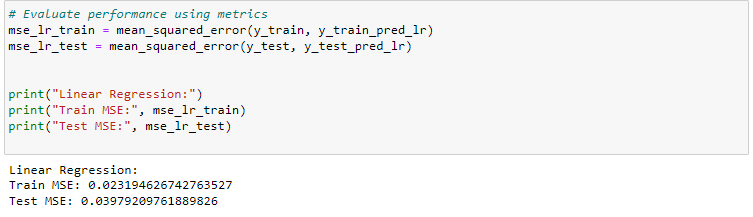
Linear Regression aims to fit a linear equation to the observed data, where the relationship between the variables is assumed to be linear. It estimates the coefficients of the linear equation to minimize the sum of squared errors between the observed and predicted values. Linear Regression is commonly used for predicting continuous outcomes, such as predicting house prices based on features like area, number of bedrooms, etc.

On the other hand, Logistic Regression is used for binary classification tasks, where the dependent variable is categorical with two possible outcomes. Logistic Regression models the probability that an instance belongs to a particular class using the logistic function, which maps any real-valued input to a value between 0 and 1. It is widely used in fields like healthcare (e.g., predicting disease outcomes), finance (e.g., credit risk assessment), and marketing (e.g., customer churn prediction).

Both Linear and Polynomial Regression offer interpretability, ease of implementation, and computational efficiency, making them popular choices for various predictive modelling tasks.

But Logistic Regression is not applicable in this case as there are no categorical variables left to predict.

Linear Regression:

Top of Form

Polynomial Regression:

A computer screen shot of a computer code

Description automatically generated

**6.XGBoost:**

XGBoost, or Extreme Gradient Boosting, is an optimized implementation of the gradient boosting algorithm. It is known for its scalability, speed, and performance in solving classification and regression problems.

XGBoost works by sequentially adding decision trees to minimize a loss function, iteratively refining the model's predictions. It incorporates regularization techniques to prevent overfitting and can handle missing values and nonlinear relationships effectively. XGBoost has gained popularity in various machine learning competitions and real-world applications due to its outstanding predictive accuracy and efficiency.

A screen shot of a computer

Description automatically generated

**7.Decision Tree:**

Decision Tree is a versatile and interpretable machine learning algorithm used for both classification and regression tasks. It partitions the feature space into regions based on the values of input features, creating a tree-like structure of decision nodes. Each node represents a feature and a splitting criterion, leading to different branches corresponding to different feature values.

Decision Trees are easy to understand and visualize, making them valuable for gaining insights into the data. However, they are prone to overfitting, especially with complex datasets. Ensemble techniques like Random Forest and Gradient Boosting help mitigate this issue by combining multiple decision trees to improve predictive performance.

A white rectangular object with black text

Description automatically generated

**8.KNN (K-Nearest Neighbors):**

K-Nearest Neighbors (KNN) is a simple yet effective non-parametric algorithm used for classification and regression tasks. It works by finding the K nearest data points to a given query point based on a distance metric (e.g., Euclidean distance).

For classification, KNN assigns the majority class among its K neighbors to the query point, while for regression, it computes the average or weighted average of the target values of the K neighbors. KNN is easy to implement and understand, making it suitable for small to medium-sized datasets with low dimensionality. However, its performance may degrade with high-dimensional or imbalanced datasets, and it requires careful selection of the distance metric and the number of neighbors (K).

A computer screen shot of a computer code

Description automatically generated

**9.SVM:**

Support Vector Machine (SVM) is a powerful supervised learning algorithm used for classification and regression tasks. It works by finding the optimal hyperplane that separates different classes in the feature space with the maximum margin.

SVMs are effective in high-dimensional spaces and are versatile in handling various data distributions. They are particularly useful when the relationship between features and target variables is nonlinear or when dealing with complex decision boundaries. SVMs have been successfully applied in domains such as image classification, text categorization, and financial forecasting.

A white background with black text

Description automatically generated

**10.GBM:**

Gradient Boosting Machine (GBM) is an ensemble learning technique that builds predictive models in a sequential manner. It combines the predictions of multiple weak learners, typically decision trees, to create a strong predictive model.

GBM works by iteratively fitting new models to the residual errors of the previous models, thereby improving prediction accuracy with each iteration. It is robust against overfitting and performs well on large, complex datasets.

GBM is widely used in various applications, including click-through rate prediction, risk modeling, and demand forecasting, due to its high predictive power and flexibility in handling diverse data types and structures.

A close-up of a computer screen

Description automatically generated

**CONCLUSION:**

The comparative analysis of ARIMA, SARIMA, and LSTM models for product demand forecasting reveals diverse strengths and applications. ARIMA adeptly captures temporal dependencies, while SARIMA excels in accounting for seasonal variations, enhancing accuracy. LSTM, with its ability to discern complex sequential patterns, showcases promise in handling intricate demand trends. However, the effectiveness of each model heavily depends on data characteristics and forecasting requirements. Selecting the most suitable model should consider factors like seasonality, trend, and data irregularities. This study underscores the significance of informed model selection in optimizing inventory management and meeting customer needs, providing valuable guidance for practitioners and decision-makers in enhancing demand prediction strategies.

**A screenshot of a computer

Description automatically generated**

**FUTURE ENHANCEMENT**

Future enhancements in this realm of demand forecasting could involve the fusion of these models to create hybrid approaches that leverage the strengths of each method. Integrating ARIMA, SARIMA, and LSTM models in an ensemble or hierarchical framework might yield more robust predictions, particularly when dealing with multifaceted demand patterns. Additionally, exploring advanced deep learning architectures or incorporating external factors such as economic indicators or social trends could further augment the accuracy and adaptability of forecasting systems. Embracing advancements in machine learning techniques and expanding datasets to encompass a wider array of influencing variables can unlock even greater precision and reliability in predicting product demand.

**REFERENCES:**

[1] J. Aastrup and H. Kotzab, Analyzing out-of-stock in independent grocery stores: an empirical study, International Journal of Retail and Distribution Management, vol. 37, issue 9,pp. 765- 789, 2009.

[2] A. Ayad, Optimizing inventory and store results in big box retail environment, International Journal of Retail Distribution Management, vol. 36, issue 3, pp. 180-191, 2008.

[3] P. J. McGoldrick and E. Andre, Consumer misbehavior: promiscuity or loyalty in grocery shopping, Journal of Retailing and Consumer Services, vol. 4, no. 2, pp. 7381, 1997.

[4] D. Grewal, A. L. Roggeveen, and J. Nordfalt, The Future of Retailing, Journal of Retailing, vol. 93, no. 1, pp. 16, 2017.

[5] Zeynep Hilal Kilimci, A. Okay Akyuz, Mitat Uysal, An Improved Demand Forecasting Model Using Deep Learning Approach and Proposed Decision Integration Strategy for Supply Chain, Journal of Hindawi, vol. 2019, no. 1, pp. 1-15, 2019.

[6] N. S. Altman, An introduction to kernel and nearest-neighbor nonparametric regression, The American Statistician, vol. 46, no. 3, pp.175185, 1992.

[7] A. Cano, J. G. Castellano, A. R. Masegosa, and S. Moral, Selective Gaussian naive Bayes model for diffuse large-b-cell lymphoma classification: Some improvements in preprocessing and variable elimination, in Symbolic and Quantitative Approaches to Reasoning with Uncertainty, L. Godo, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 908920.

[8] D. H. Moore II, Classification and regression trees, by Leo Breiman, Jerome h. Friedman, Richard a. Olshan, and Charles j. stone. brooks/ cole publishing, Monterey, 1984,358 pages, 27.95, Cytometry, vol. 8, no. 5, pp. 534535, 1987.

[9] Juster, Advanced forecasting options for optimal supply chain performance: variable methods for dynamic markets, AMR Research Note, pp. 234-245, 2006.

[10] A. P. Da-Silva, V. Ferreira and R. M. Velasquez, Input space to neural network-based load forecasters, International Journal of Forecasting, vol. 24, issue 4, pp. 616-629, 2008.

[11] M. A. A. Hasin, Inventory cost comparisons of retailer and manufacturer in a two-stage supply chain, International Conference on Manufacturing Management, 2-4 March 2008, Bangkok, pp. 173-182.

[12] M. A. A. Hasin, Shuvo Ghosh, and Mahmud A. Shareef, An ANN Approach to Demand Forecasting in Retail Trade in Bangladesh, International Journal of Trade, Economics and Finance, Vol. 2, No.2, pp. 154-160, April 2011.

[13] Majed Kharfan, Vicky Wing Kei Chan, Forecasting Seasonal Footwear Demand Using Machine Learning, Publisher Massachusetts Institute of Technology, 2018.

[14] N. Holt and G. Winter, Advanced Exponential Smoothing techniques for seasonal effect and trend, International Journal of Forecasting, vol. 3, issue 1, pp. 12-23, 1987.

**Drive Link:**

[**Product Demand Forecasting**](https://drive.google.com/drive/folders/1VbtZt2fGu0kVmadLaoH89AFpDhvMjJl8?usp=drive_link)

**THE END**