University of West Florida

Capston Project In Data Science IDC-6940 | Professor Dr. Shusen Pu

Edwin Quijano

Tuesday, October 14, 2025

Intelligent Inventory Optimization with Deep Learning and Reinforcement Learning.

Introduction

Problem Definition

In today's dynamic global economy, accurate inventory forecasting remains one of the most complex and impactful challenges in supply chain management. Variability in consumer demand, fluctuating supplier lead times, and unpredictable disruptions can lead to costly stockouts or excess inventory. These inefficiencies directly affect operational costs, profit margins, and sustainability outcomes. The research problem addressed in this project is how to design intelligent forecasting and inventory optimization systems that accurately predict product demand and dynamically adjust reorder policies under uncertainty. From a data science perspective, this problem requires the integration of statistical modeling, time series forecasting, and machine learning-based decision-making under stochastic conditions.

Context and Background

Traditional forecasting methods, such as Exponential Smoothing and ARIMA models, have long been used to predict demand trends. However, their linear assumptions and limited adaptability often fail to capture nonlinear and temporal dependencies inherent in large-scale inventory systems. Recent advances in Deep Learning (DL) and Reinforcement Learning (RL) have introduced data-driven methods capable of learning complex temporal relationships and adaptive decision-making strategies.

For instance, Long Short-Term Memory (LSTM) networks have shown superior performance in capturing long-term dependencies in time series data, while Convolutional Neural Networks (CNNs) can extract spatial and sequential features useful for multivariate forecasting. In contrast, Reinforcement Learning — particularly Deep Q-Networks (DQN) and Proximal Policy Optimization (PPO) — enables systems to optimize inventory policies by continuously interacting with an environment and receiving feedback in the form of rewards or penalties.

This project builds upon recent research demonstrating the convergence of these fields. Pasupuleti et al. (2024) highlight that combining ML and RL methods improves supply chain agility and reduces waste through adaptive decision-making. Similarly, Sathish et al. (2024) show that Gradient Boosting and other machine learning algorithms outperform traditional forecasting models. Collectively, these studies establish a theoretical foundation for using hybrid models to enhance accuracy and efficiency in inventory management.

Objectives and Goals

The objectives of this research are threefold:

Forecasting Accuracy: Develop and evaluate deep learning models (e.g., LSTM, CNN-LSTM) to predict product-level demand across multiple time horizons.

Dynamic Inventory Optimization: Implement a reinforcement learning framework that learns optimal reorder policies through continuous interaction with simulated inventory environments.

Comparative Evaluation: Benchmark the performance of deep learning and RL-based methods against classical statistical models (ARIMA, Exponential Smoothing) using quantitative performance metrics such as MAE, RMSE, and cost efficiency.

The overarching goal is to create an intelligent inventory optimization system capable of adapting to changing demand patterns and improving decision-making accuracy at scale.

Summary of Approach

The study integrates Deep Learning and Reinforcement Learning methodologies using the Global Product Inventory Dataset 2025 from Kaggle. The dataset contains global-level records of product demand, inventory quantities, supplier information, and lead times. First, exploratory data analysis (EDA) and feature engineering will prepare time-dependent variables. Next, deep learning models will forecast demand, while reinforcement learning agents will optimize restocking actions within a simulated environment. Model performance will be evaluated using both statistical accuracy measures and business-oriented metrics such as cost savings and service levels.

Methods

Data Acquisition and Sources

The dataset used in this project is the Global Product Inventory Dataset 2025 (Keyushnisar, 2025) available on Kaggle. It contains over 10~~0~~,000 observations of product-level inventory and demand records across multiple global warehouses and suppliers. Key features include product

identifiers, stock levels, sales quantities, reorder amounts, lead times, and timestamps, enabling time-series analysis.

Prior to modeling, data preprocessing steps will include:

- Data Cleaning: Handling missing values and outliers through imputation and winsorization.
- Feature Engineering: Creating time-based variables such as rolling averages, lag features, and reorder point estimates.
- Normalization: Applying Min-Max scaling and one-hot encoding for categorical variables to optimize model training performance.

Data Source: https://www.kaggle.com/datasets/keyushnisar/global-product-inventory-dataset-2025

Mathematical and Statistical Models

1. Time Series Forecasting Models

The deep learning component employs LSTM and CNN-LSTM architectures to model temporal dependencies and nonlinear demand relationships.

Loss Functions: Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE).

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

Baseline comparisons will include ARIMA, Exponential Smoothing, and XGBoost models to establish a statistical benchmark.

2. Reinforcement Learning Model

The decision-making component uses a Deep Q-Network (DQN) to simulate inventory control under uncertainty.

State Space (S): Current stock level, predicted demand, lead time, holding cost, shortage cost.

Action Space (A): Discrete reorder quantities.

Reward Function (R):

$$R_t = -(C_h \times H_t + C_s \times S_t)$$

where C_h and C_s represent holding and shortage costs, respectively, and H_t S_t are the respective quantities at time t.

The DQN is trained using experience replay and an -greedy exploration policy to balance exploitation and exploration.

Experimental Design and Analytical Procedures

The analysis proceeds through the following steps:

1. Exploratory Data Analysis (EDA): Examine distributions, seasonality, and correlation among features.
2. Feature Engineering: Construct lagged and rolling features to enhance temporal signal representation.
3. Model Training and Validation:
    1. Split dataset into training (70%), validation (15%), and testing (15%) sets.
    2. Train and tune models using grid search and cross-validation.
4. Performance Evaluation:
    1. Forecasting models: MAE, RMSE, and MAPE.
    2. Reinforcement learning agent: total cost reduction, service level (fill rate), and stockout frequency.
5. Comparison and Integration: Combine best-performing models into a unified hybrid system for evaluation.

Software and Tools

The project is implemented primarily in Python and R, using the following tools:

- Python libraries: TensorFlow, Keras, Scikit-learn, XGBoost, Stable Baselines3
- Data analysis and visualization: pandas, NumPy, matplotlib, seaborn
- R environment: Bibliometrix for analytical reference management
- Development environment: Jupyter Notebook and Google Colab

Ethical Considerations

Although the dataset is publicly available and anonymized, ethical standards for data use and reproducibility are upheld. The project ensures:

- Transparency in data preprocessing and modeling decisions.
- Reproducibility of results through documented code and parameter logs.
- Proper citation of all data and research sources.

Time Series Forecasting

The primary objective of this step is to develop robust demand forecasts at the SKU level to support downstream inventory optimization. Accurate forecasting improves service levels, minimizes stockouts, and reduces unnecessary holding costs.

3.1 Data Preparation for Forecasting

Each SKU is converted into a standardized time series:

Time index: Daily or weekly frequency

Target variable: Units sold per day/week

Optional features:

Promotions/discounts

Day-of-week, month, or season indicators

Holiday markers

Lagged demand features (t–1, t–7, t–30, etc.)

Rolling window features (7-day moving average, rolling standard deviation)

Demand series are inspected for missing values and non-selling days. Missing demand is either imputed as zero

```
library(tidyverse)
```

```
Warning: package 'tidyverse' was built under R version 4.4.2
```

```
Warning: package 'ggplot2' was built under R version 4.4.3
```

```
Warning: package 'tibble' was built under R version 4.4.3
```

```
Warning: package 'purrr' was built under R version 4.4.3
```

```
Warning: package 'lubridate' was built under R version 4.4.2
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.4     v readr     2.1.5
v forcats   1.0.0     v stringr   1.5.1
v ggplot2   3.5.2     v tibble    3.3.0
v lubridate 1.9.4     v tidyr     1.3.1
v purrr     1.1.0
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becor
```

```r
library(lubridate)

# Replace with your actual file path / object
df_raw <- read_csv("products.csv")
```

```
Rows: 10000 Columns: 14
-- Column specification --------------------------------------------------------
Delimiter: ","
chr  (8): Product ID, Product Name, Product Category, Product Description, P...
dbl  (4): Price, Stock Quantity, Warranty Period, Product Ratings
date (2): Manufacturing Date, Expiration Date

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
df <- df_raw %>%
  rename(
    sku      = SKU,      # change if your column is named differently
    date     = `Manufacturing Date`,     # e.g., "order_date", "invoice_date"
    quantity = `Stock Quantity`  # e.g., "qty", "units_sold"
  ) %>%
  mutate(
    date = as.Date(date)
  ) %>%
  filter(!is.na(sku), !is.na(date))
```

Aggregate to daily demand per SKU

Sometimes there are multiple transactions per SKU per day. We want one row per SKU per day.

```
daily_demand <- df %>%
  group_by(sku, date) %>%
  summarise(
    demand = sum(quantity, na.rm = TRUE),
    .groups = "drop"
  )
```

Create a complete calendar (fill missing days with 0)

```
# Get full date range of the dataset
min_date <- min(daily_demand$date)
max_date <- max(daily_demand$date)

calendar <- tibble(date = seq.Date(from = min_date, to = max_date, by = "day"))

# For each SKU, join with the full calendar and replace NAs with 0
daily_complete <- daily_demand %>%
  group_by(sku) %>%
  right_join(calendar, by = "date") %>%
  arrange(sku, date) %>%
  mutate(
    demand = replace_na(demand, 0)
  ) %>%
  ungroup()
```

Add basic calendar features (for later modeling)

```
daily_prepared <- daily_complete %>%
  mutate(
    dow   = wday(date, label = TRUE, week_start = 1),  # day of week
    month = month(date, label = TRUE),
    year  = year(date),
    week  = isoweek(date)
  )
```

Extract one SKU as a time series example

```
# Pick one SKU to start with
example_sku <- "SKU_001"  # change to an actual SKU value in your data

ts_sku <- daily_prepared %>%
  filter(sku == '00ZTIS') %>%
  arrange(date)

# If you want a pure numeric time series object:
demand_ts <- ts(ts_sku$demand, frequency = 7)  # daily data with weekly seasonality
```

```
colnames(df_raw)
```

```
 [1] "Product ID"            "Product Name"          "Product Category"
 [4] "Product Description"   "Price"                 "Stock Quantity"
 [7] "Warranty Period"       "Product Dimensions"    "Manufacturing Date"
[10] "Expiration Date"       "SKU"                   "Product Tags"
[13] "Color/Size Variations" "Product Ratings"
```

```
df <- df_raw %>%
  rename(
    sku = ProductID,
    date = Date,
    quantity = Qty
  ) %>%
  mutate(date = as.Date(date))
```

References

Bastos, A. da S. T. (2024). Machine learning in digital retail: Demand forecasting for inventory management in a sportswear company (Order No. 31790125) [Master's thesis, ProQuest Dissertations & Theses Global]. https://www.proquest.com/dissertations-theses/machine-learning-digital-retail-demand/docview/3144033974/se-2

Keyushnisar, K. (2025). Global product inventory dataset 2025 [Data set]. Kaggle. https://www.kaggle.com/datasets/keyushnisar/global-product-inventory-dataset-2025

Pasupuleti, V., Thuraka, B., Kodete, C. S., & Malisetty, S. (2024). Enhancing supply chain agility and sustainability through machine learning: Optimization techniques for logistics and inventory management. Logistics, 8(3), 73. https://doi.org/10.3390/logistics8030073

Polo-Triana, S., Gutierrez, J. C., & Leon-Becerra, J. (2024). Integration of machine learning in the supply chain for decision making: A systematic literature review. Journal of Industrial Engineering and Management, 17(2), 344. https://doi.org/10.3926/jiem.6403

R, P., Kumar, S., Bhardwaj, S., Agrahari, N., Pandey, S., & Harakannanavar, S. S. (2023). E-commerce inventory management system using machine learning approach. In Proceedings of the 2023 International Conference on Data Science and Network Security (ICDSNS) (pp. 1–7). IEEE. https://doi.org/10.1109/icdsns58469.2023.10245500

Sathish, T., SaiKumar, D., Patil, S., Saravanan, R., Giri, J., & Aly, A. A. (2024). Exponential smoothing method against the gradient boosting machine learning algorithm-based model for materials forecasting to minimize inventory. AIP Advances, 14(6). https://doi.org/10.1063/5.0208491