4/23/2025

MLG382 CYO Project

Waldo Blom (578068)

Erin David Cullen (600531)

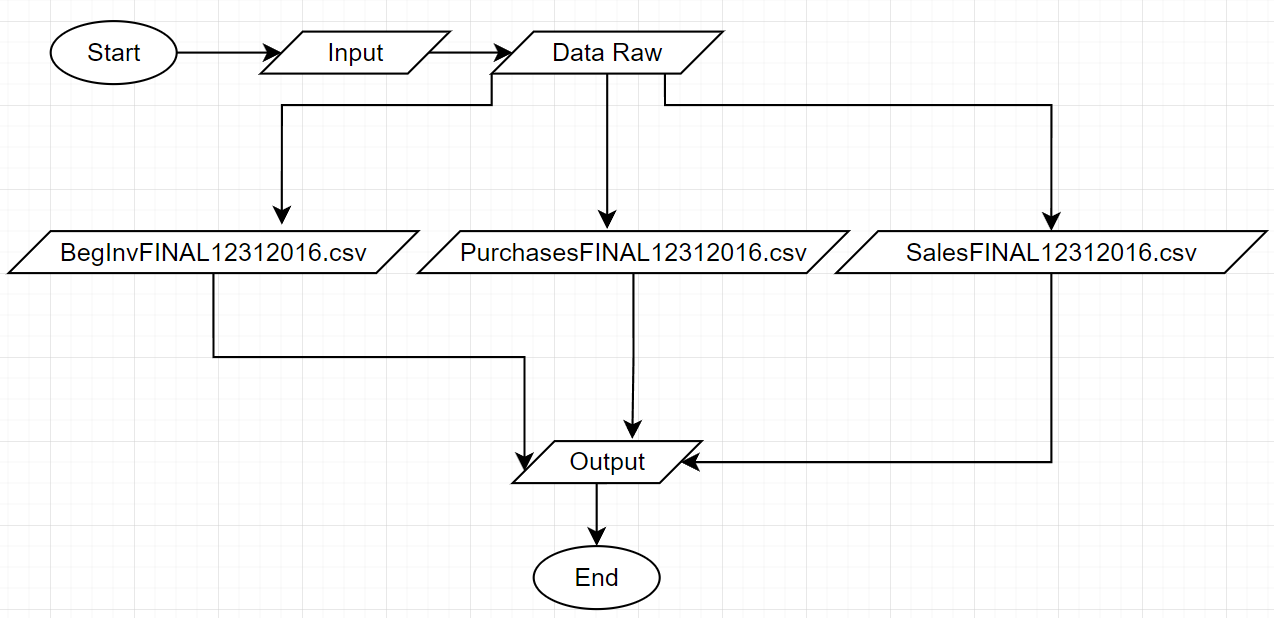
Brandon Alfonso Lemmer (578062)

Tristan James Ball (601541)

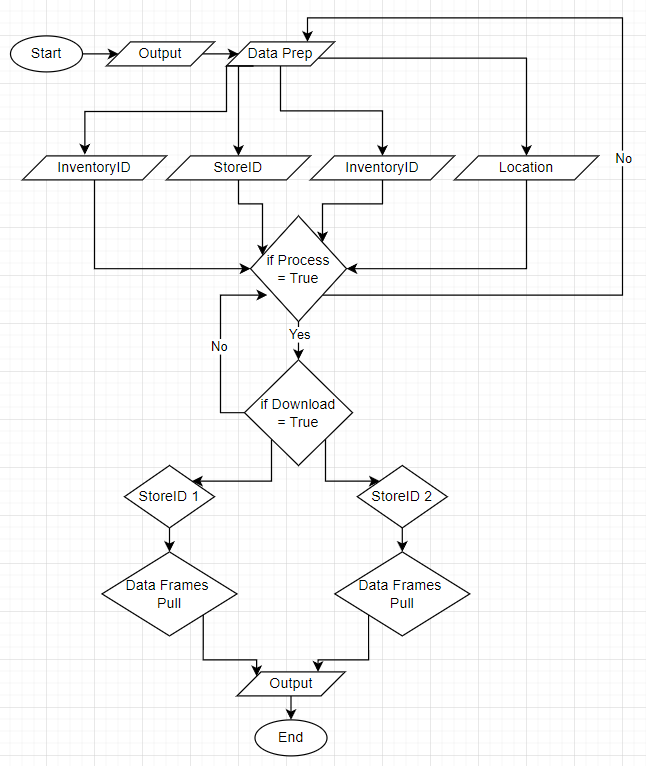
# Contents:

|  |  |
| --- | --- |
| Chapter: | Page: |
| Flow Charts | 3-4 |
| Problem statement | 5 |
| Data Understanding | 5-11 |
| Data load and cleaning | 11 |
| Web application development process | 12 |
| Lead time model | 12 |
| Sales model | 13-14 |
| Link to GitHub repository | 14 |
| Link to Dash App | 15 |
| Key Findings | 15 |
| Challenges faced | 15 |
| Lessons Learned | 15 |

**Flow chart to visualize our data flow prosses:**



**Flow chart to visualize our data preparation prosses:**



**Flow chart to visualize our data processing prosses through our models:**

A diagram of a software development

AI-generated content may be incorrect.

**Flow chart to visualize our Web app process:**

A diagram of a software process

AI-generated content may be incorrect.

# Problem statement

This is data is based on a dataset from Kaggle that has been modified. Link to Kaggle dataset: <https://www.kaggle.com/datasets/bhanupratapbiswas/inventory-analysis-case-study>

**Optimizing Inventory Management for Tops Liquor Stores in South Africa:**

Tops, a leading liquor retail chain in South Africa, operates across multiple store locations and carries a wide variety of alcoholic beverages. The brand seeks to optimize its inventory management system to enhance operational efficiency, reduce carrying costs, and ensure product availability across all its stores.

Currently, inventory data is scattered across various sources including purchase records, sales data, opening stock levels, and individual store information. However, the absence of a centralized, predictive system often results in **stockouts**, **overstocking** and inefficient **reorder processes** negatively impacting sales and increasing operational costs.

The goal of this project is to develop a data-driven inventory management solution for Tops that:

* Accurately forecasts a specific product demand (sales) at a specific store based on historical sales with the help of a XG boost machine learning model applied.
* Accurately forecasts lead times of product orders, so they know when to order new products.
* Optimizes stock levels to avoid both overstocking and stockouts.

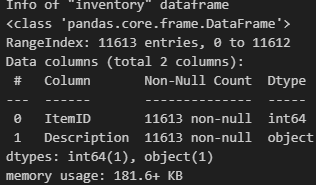
Using available datasets on inventory, purchases, sales, and store locations, this system will help Tops make smarter inventory decisions and better align stock levels with customer demand.

# Data Understanding

**Data prep and load:**

EDA analysis:

* Missing value analysis



From this we can see there is no data missing.

Check for duplicates:

* We only check for duplicate products as that is the only field that should not contain any duplicates. It is possible that the same purchase or sale could take place on the same day, so we do not check for any duplicates.

A black screen with white text

AI-generated content may be incorrect.

We can see the index is empty so there are no duplicates.

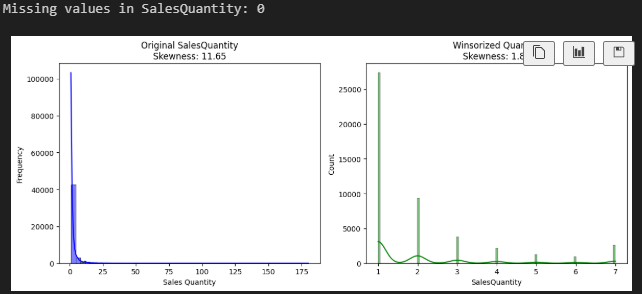
Describe data:

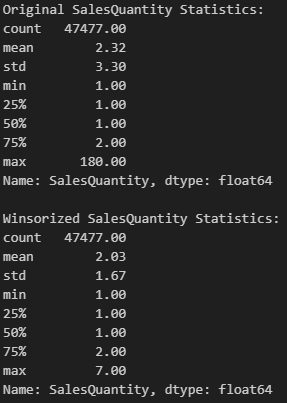
**A screenshot of a black screen

AI-generated content may be incorrect.**

We can gain valuable insights form the table like count, mean and standard deviation.

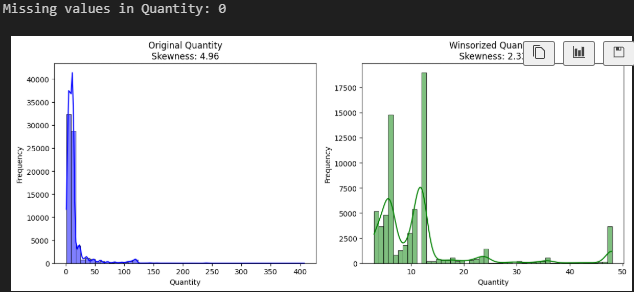
**Checking for missing values and comparing original data against winsorized data for Sales Quantity:**

****

****

We can see no data is missing and all values were successfully replaced with acceptable values.

**Checking for missing values and comparing original data against winsorized data for Quantity:**

****

**A screenshot of a computer

AI-generated content may be incorrect.**

We can see no data is missing and all values were successfully replaced with acceptable values.

**Sales analysis:**

A graph of blue bars

AI-generated content may be incorrect.

Here we can see Item 8111 sells the best.

**Purchases analysis:**

A graph of purple bars

AI-generated content may be incorrect.

Here we can see that the month Jully had the most sales.

**Lead time analysis**

A screenshot of a computer screen

AI-generated content may be incorrect.

Here we can see that we will wait on average 8 days for placing a new stock order till receiving it.

**Final data prep:**

Load the Prepped Data into:

* lead\_time\_data.csv
* sales\_forecast\_data.csv

Both files need to go through some final processing before they are ready to train models

2 functions to read, both are notarized

* create\_lead\_time\_data
* create\_sales\_forecast\_data

Lead\_time\_data.csv:

Column Name     | Description

* PODate          | Date the item was ordered (purchases\_df) (yyyy-mm-dd)
* ReceivingDate   | Date the item was received (purchases\_df) (yyyy-mm-dd)
* LeadTimeDays    | Target = ReceivingDate - PODate (in days)
* ItemID          | Item identifier (purchases\_df)
* Description     | Item description (inventory\_df)
* StoreID         | Store identifier (purchases\_df)
* Location        | Store location (stores\_df)
* Quantity        | Quantity ordered (purchases\_df)
* Week            | Week number of the PODate (optional feature)
* Month           | Month of PODate (optional feature)
* Day             | Day of week PODate was placed (optional feature)

Sales\_forecast.csv:

Column Name     | Description

* SalesDate       | Date of sale (daily or weekly aggregated)
* ItemID          | Item identifier
* StoreID         | Store identifier
* SalesQuantity   | Target: quantity sold
* Lag\_1           | Sales one day/week before
* Lag\_7           | Sales one week before
* RollingAvg\_7    | 7-day rolling average
* Month           | Month of sale
* DayOfWeek       | Day of week of sale
* StockOnHand     | (optional) On-hand inventory if available
* ReceivedQty     | Quantity received in last x days (optional)

**A screenshot of a computer

AI-generated content may be incorrect.**

All data has been saved to the correct file, collum and row.

# Data load and cleaning

**Load and clean data:**

Creates 5 clean tables and exports them to Data/Processed:

Inventory.csv:

* ItemID - Gives the ItemID
* Description - Gives the Item Description

OpeningStock.csv:

* StoreID - StoreID
* onHand - How Much Stock is on hand
* startDate - 2015-12-31 stock taken for the start of 2016
* ItemId - Gives the ItemID

Purchases.csv

* StoreID – StoreID
* PODate - Purchase Order Date
* ReceivingDate - Order Revieved Date
* Quantity - Quantity Recieved in Order
* ItemId - Gives the ItemID

Sales.csv

* StoreID - StoreID
* SalesQuantity - Quantity of item sold
* SalesDate - Date that the sale took place
* ItemId - Gives the ItemID

Stores.csv

* StoreID - StoreID
* Location - Location of the store

# Web application development process

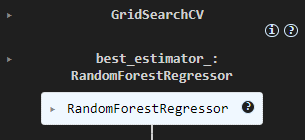
Build a dash app in repo:

* In the dash app, set up server = app.server.
* deploy repo to GitHub.
* Start a new project that runs on python in render.com.
* Link to GitHub repo.
* In Start Command set gunicorn to WebApp.app:server.

# Models

## Lead time model

Train Random Forest Model:



We have now successfully trained our model.

Evaluate Model:



Mean Absolute Error (MAE):

* Value: 1.0788
* This model performed well as the predictions are, on average, around 1 day off actual lead time.

Root Mean Squared Error (RMSE):

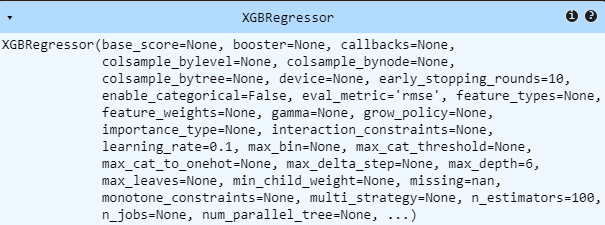
* Value: 1.3865
* This metric makes up for what MAE lacks in that RMSE heavily penalizes larger errors and outliers. Its value indicates that are not many large errors as it is close to the value of MAE.

R^2:

* Value: 0.6130/~61%
* This model explains around 61% of the variance in the target. This is a solid value considering the difficulty of predicting lead times.

## Sales model

Train XG Boost Model:



Here we can see all the parameters we used to train the XG boost model.

Predictions & Evaluation:



1. Mean Absolute Error (MAE: 0.9914)

* Meaning: Predictions deviate by approximately 0.99 units of Sales Quantity on average.
* Implication: Good performance on typical days (6-12 units) but may miss larger spikes (near 48).

2. Root Mean Square Error (RMSE: 1.3993)

* Meaning: Higher than MAE, indicating some larger prediction errors.
* Implication: Model struggles with outliers/high-sales events (30-48 range).

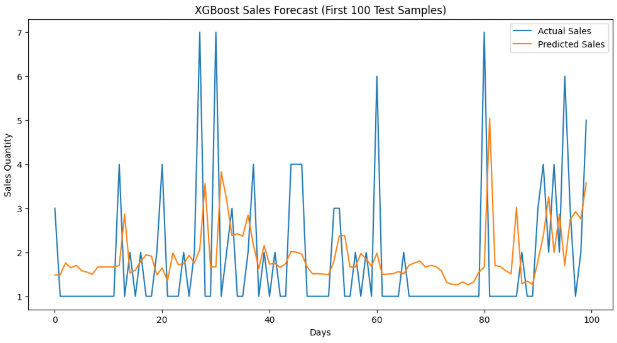
3. R² Score (0.2323)

* Meaning: The model explains approximately 23.23% of the variance in sales.
* Implication: Missing critical explanatory variables for high-sales days.

Strengths & Limitations

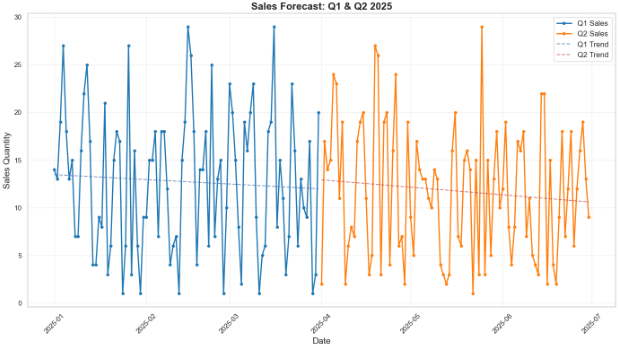
* Model performs well for average conditions.
* Struggles with extreme values, which inflate RMSE due to squared error weighting
* Model likely underpredicts high-sales events in the 30-48 range

Plot Predictions vs Actual (First 100 Days):



Here we can see the accuracy of our model. It predicts accurately within 1-3 units of stock.

Example of how the model would work:



Here we predict the next 6 months of sales.

# Link to GitHub repository

[BeardedSeal77/MLG382\_CYO\_Project](https://github.com/BeardedSeal77/MLG382_CYO_Project)

# Link to Dash App

[CYO Project on Render](https://mlg382-cyo-project-mbr5.onrender.com/)

# Key Findings

WebApp is the folder that the web server runs in (where all the website files are).

# Challenges faced

Acquiring the data was a challenge, understanding how the data would lead to predictions, dash app deployment and how to set up the dash app.

# Lessons Learned

How to deploy dash app.