Data Engineer Test Report

Supermarket analytics with three supervised learning solutions and a separate reinforcement learning maze agent

Candidate Santosh Nagh Sirimalla

Role Applied Data Analyst

Date of Submission August 30, 2025

Executive Summary

This project demonstrates the full data engineering and analytics workflow, from raw supermarket transaction data to predictive modeling and reinforcement learning. The objective was to showcase the ability to clean, integrate, and transform real retail data, conduct deep exploratory analysis, and build models that provide actionable insights for operations, customer retention, and store benchmarking.

Data integration and preparation involved combining sales, items, supermarket metadata, and promotions. We engineered time based features (day of week, month, hour bucket, weekend flag), basket level metrics (items, diversity, average price), and customer level recency and frequency features. Outliers were removed by capping at the ninety ninth point ninth percentile, missing size categories were imputed, and inconsistencies such as negative sales amounts were cleaned. Promotion data could not be merged due to lack of temporal overlap.

Exploratory data analysis revealed clear shopping patterns. Sales peaked in afternoons and weekdays, while Sundays were the single strongest day. Basket values were small, with most under 10 dollars, and basket sizes averaged 2 to 3 items. A handful of high spending customers accounted for a disproportionate share of revenue. RFM segmentation confirmed that loyal customers drove frequency and monetary value, while a large portion of customers were at risk due to long recency and low spend.

Supervised learning models were developed for three business problems:

- **Basket revenue forecasting**: Random Forest and Gradient Boosting achieved R squared of **0.984**, with MAE of **0.094** and RMSE of **0.537** for Random Forest. Basket revenue was almost entirely explained by number of items and average unit price, enabling highly accurate real time forecasts at checkout.
- **Customer return within 30 days**: After excluding May due to missing transactions, Logistic Regression achieved **AUC 0.616, PR AUC 0.741, accuracy 0.56**, while Random Forest achieved **AUC 0.670, PR AUC 0.794, accuracy 0.59**. Recency days, rolling 30 day spend, and frequency were the strongest predictors. While inherently challenging, the model provides probabilities that can rank customers for targeted retention campaigns.
- **High performing supermarkets**: Defined as stores in the top quartile of revenue, Random Forest achieved **AUC 0.997, accuracy 0.96**. The most important drivers were transactions and number of unique customers, followed by average basket value. The model enables management to benchmark stores, identify operational levers, and replicate success across the network.

Reinforcement learning was applied independently in a 10 by 10 maze. Using Q learning with epsilon greedy exploration, the agent converged on a 22 step path to the goal after 1,500 episodes. This demonstrated practical reinforcement learning knowledge, including reward design, exploration decay, and convergence monitoring.

Key business recommendations include using basket forecasts to improve cashier staffing and detect billing anomalies, deploying churn probabilities to drive daily retention campaigns, and applying the supermarket benchmarking model in monthly performance reviews. Strengthening data pipelines to prevent missing periods, automating dashboards for store managers, and feeding campaign outcomes back into models will maximize impact.

Overall, the project validates the ability to engineer data pipelines, extract insight through exploratory analysis, and build predictive models that can be operationalized to create measurable business value in retail.

1. Task Overview

The Data Engineer Test was divided into 2 major tasks. Task 01 required cleaning, transforming, and analyzing supermarket sales data using supervised machine learning techniques to generate business valued insights. Task 02 required designing and training a reinforcement learning agent to navigate a maze environment. The 2 tasks are independent of each other, one focused on structured business data and the other on an unsupervised learning environment.

For Task 01, four datasets were provided — items, sales, promotions, and supermarkets. These datasets covered 2 years of supermarket transactions across multiple branches in 2 provinces. The business objective was to demonstrate the ability to prepare raw transaction data for machine learning and build models that deliver actionable insights. Three supervised learning problems were solved: forecasting basket revenue, predicting whether a customer will return within 30 days, and identifying high performing supermarkets from aggregated transaction features.

Each supervised learning problem followed a full cycle of work. The datasets were cleaned and normalized, missing values and inconsistencies were addressed, features were engineered, and models were trained and evaluated using appropriate metrics. Multiple algorithms were tested and compared for performance, including linear regression, logistic regression, random forest, and gradient boosting. The results were validated and interpreted to highlight the business relevance of the models.

For Task 02, the goal was to simulate an environment where an agent starts at one point of a maze and must reach the goal while avoiding obstacles. The agent used reinforcement learning to learn through repeated episodes of trial and error. A Q learning approach with ε greedy exploration was applied. The training process showed how the agent gradually improved its navigation, reduced collisions, and optimized the path to the goal. This task demonstrates problem solving under uncertain and dynamic conditions, complementing the structured supervised learning work in Task 01.

The overall assignment required not only building accurate models but also creating clear documentation, detailed narration of the process, and reproducible results. This report captures the entire workflow from raw data to insights and trained models.

2. Data and Metadata

The first step in the process was to understand the structure and meaning of the provided datasets. Four datasets were supplied as part of Task 01. Together, they covered product level details, customer transaction history, promotional activity, and supermarket location information. A clear understanding of these files and their relationships was essential to building a reliable analytical pipeline.

2.1 Items.csv

This dataset contains the master information for products sold in the supermarkets.

- **Code**: Unique identifier for each item
- **Description**: Text description of the item
- **Type**: Category grouping (Type 1 to Type 4)
- **Brand**: Brand of the item
- **Size**: Pack size or weight, often stored in inconsistent formats (e.g. “32 OZ”, “2 LB”)

2.2 Sales.csv

This dataset contains line level sales transactions over a period of 2 years.

- **Code**: Matches product code in Items.csv
- **Amount**: Price or amount associated with the transaction line
- **Units**: Quantity sold
- **Time**: Time of the transaction
- **Province**: Province where the sale occurred
- **CustomerId**: Unique customer identifier
- **Supermarket No**: Store identifier
- **Basket**: Basket id representing a unique shopping trip
- **Day**: Day of the transaction
- **Voucher**: Indicator of voucher usage

2.3 Promotion.csv

This dataset contains details of promotions applied in specific stores during specific weeks.

- **Code**: Product code
- **Supermarket No**: Store identifier
- **Week**: Week number of the promotion
- **Feature**: Type of promotional feature (e.g. Interior Page Feature)
- **Display**: Display type in the store (e.g. End Cap, Mid Aisle)
- **Province**: Province of the promotion

2.4 Supermarkets.csv

This dataset contains store location level details.

- **Supermarket No**: Unique store identifier
- **Post-code**: Postal code of the store location

2.5 Relationships

The 4 datasets are related through shared keys.

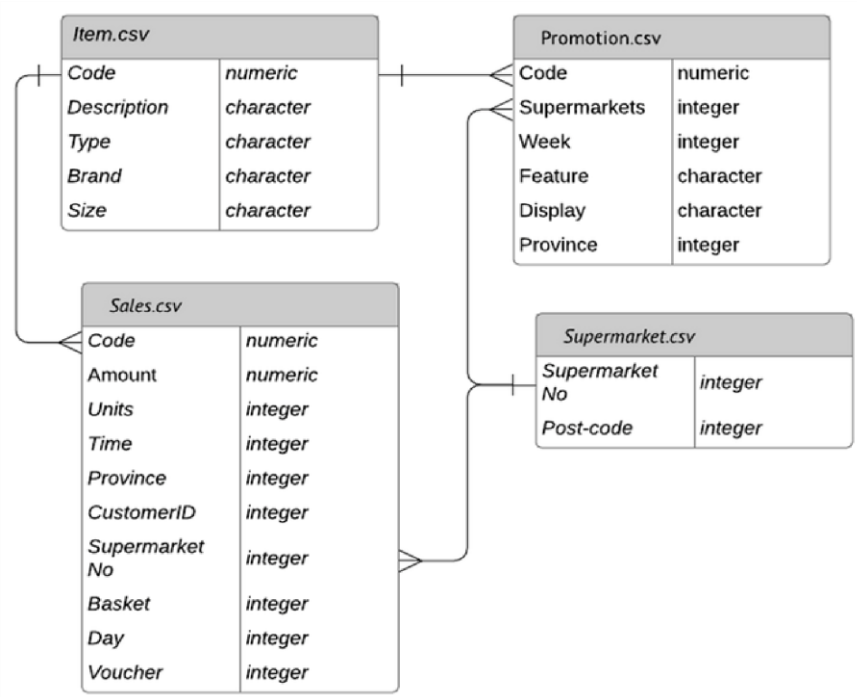
- **Code** links Sales and Items, as well as Sales and Promotion.
- **Supermarket No** links Sales, Promotion, and Supermarkets.
- **Province** appears in Sales, Promotion, and Items, enabling regional analysis.
- **CustomerId** in Sales allows tracking of customer behavior across multiple transactions.
- **Basket** in Sales groups together all items purchased during a single shopping trip.

2.6 Data Challenges

The metadata review highlighted several challenges that required careful cleaning:

- **Inconsistent size formats** in Items.csv (e.g. ounces vs pounds, text variations)
- **Ambiguity in Amount field** in Sales.csv where it could represent either unit price or extended line amount
- **Missing values** in multiple fields
- **Outliers** in transaction amounts and units that did not conform to normal ranges
- **Time variables** stored in separate columns (day and time) instead of a single timestamp

These findings guided the cleaning and transformation strategy described in the next section.



3. Data Cleaning and Transformation

The raw supermarket datasets contained inconsistencies, missing values, and structural issues that needed to be addressed before they could be used for machine learning. A systematic pipeline was followed to clean and enrich the data, ensuring it was reliable and ready for analysis.

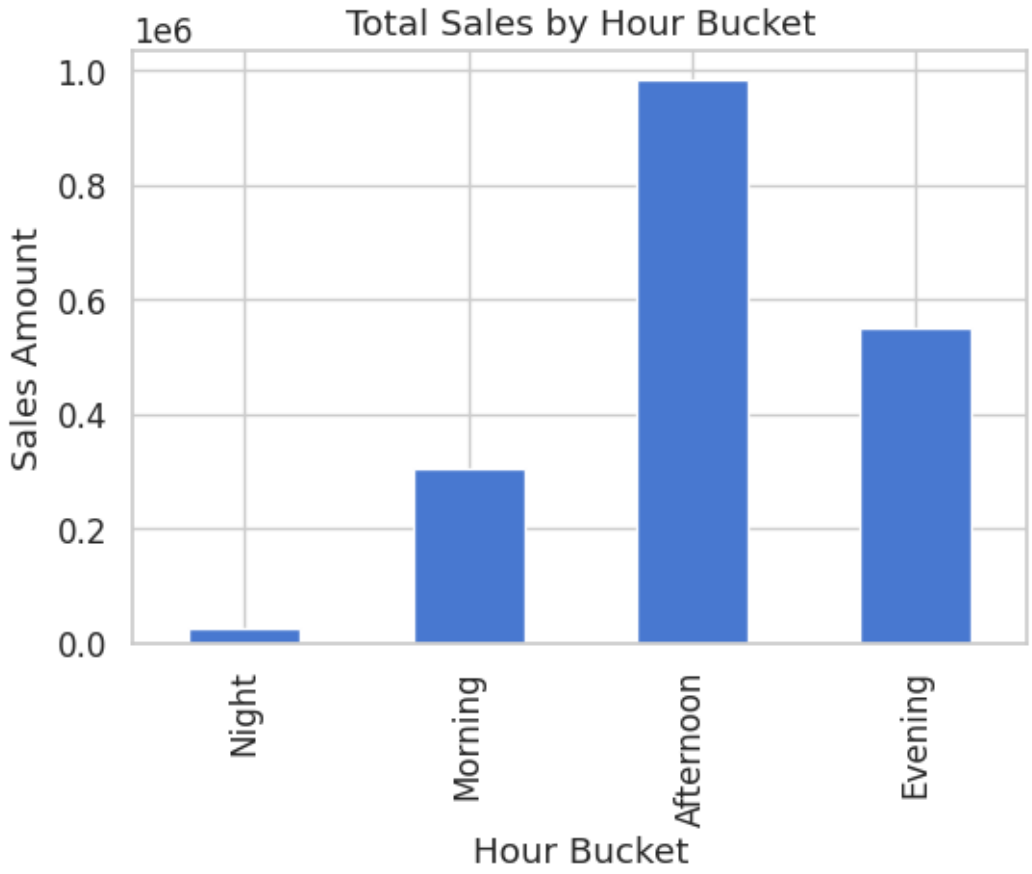
3.1 Datetime creation and time features

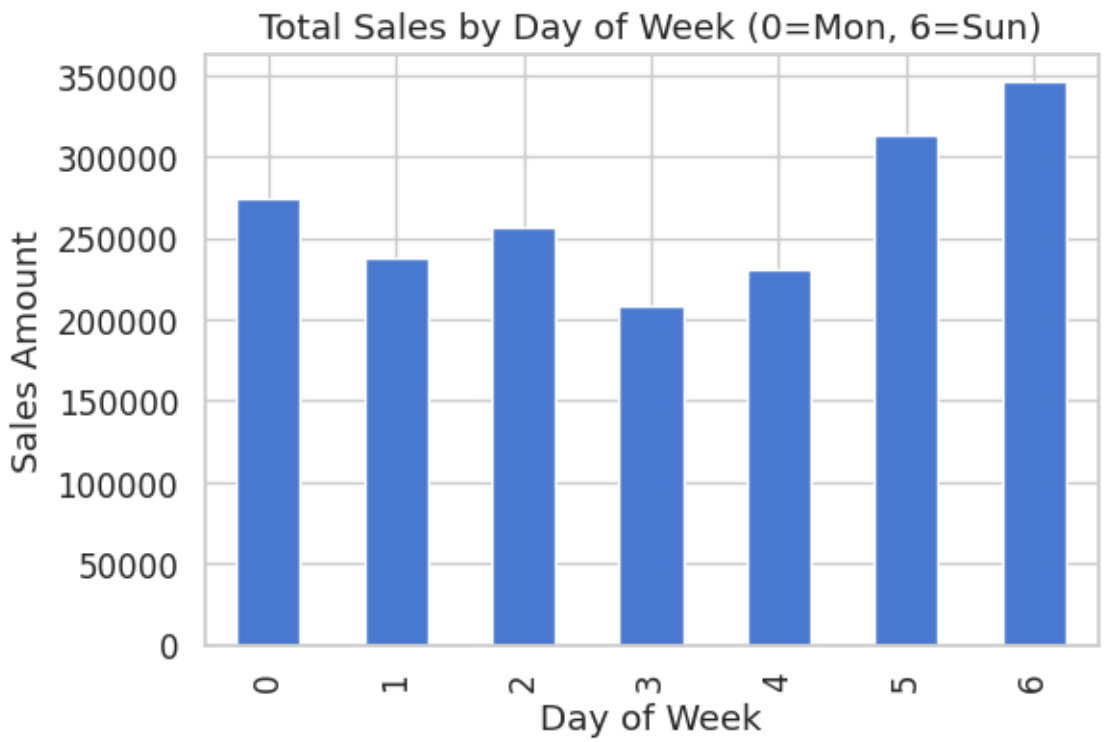
The Sales dataset had day and time fields stored separately. These were combined into a single datetime object. Once the timestamp was constructed, several new calendar-based features were engineered:

- **day_of_week**: integer from 0 (Monday) to 6 (Sunday)
- **month**: extracted month number from the date
- **is_weekend**: binary flag equal to 1 if the transaction occurred on Saturday or Sunday, otherwise 0

- **hour_bucket**: categorical feature dividing the day into Night, Morning, Afternoon, and Evening

These features allowed for temporal aggregations and insights into customer shopping patterns. For instance, sales were aggregated by hour_bucket and day_of_week, showing that the Afternoon period generated the highest sales and that Sunday was the busiest day of the week.

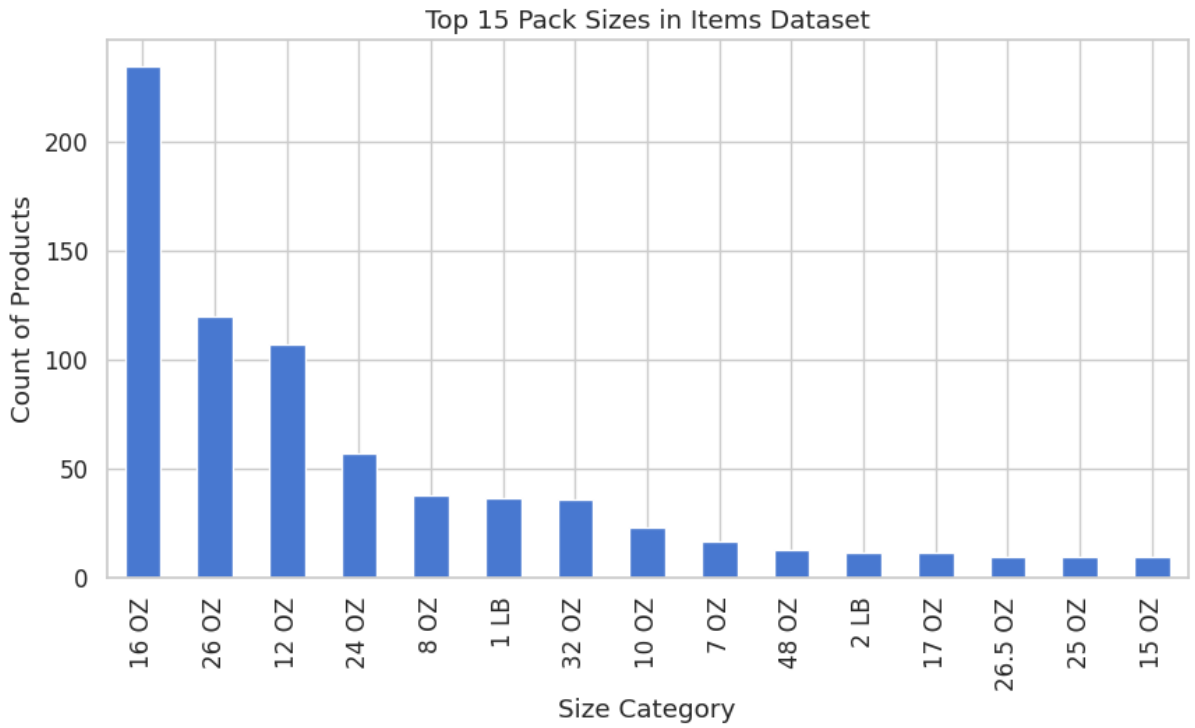




3.2 Cleaning the Size column and creating size_category

The Items dataset contained a size field in inconsistent formats such as "32 OZ", "2 LB", and "16OZ" without spaces. A regular expression parser was applied to extract numeric values and their units. The units were standardized and a new feature called **size_category** was derived.

This provided a clean categorical variable grouping products by their pack sizes. The most common pack size was **16 OZ**, followed by **26 OZ** and **12 OZ**. This variable was later used in aggregation and as a feature in supervised models.



3.3 Dataset merging

To construct a unified analytical dataset, several merges were performed:

- **Sales × Items**: merged on code, adding product details such as type, brand, and size_category to transaction lines.
- **Sales × Supermarkets**: merged on supermarket, bringing in store location metadata.
- **Sales × Promotion**: attempted, but no temporal overlap was found between promotion weeks and the sales transaction period. Therefore, the promotion dataset was excluded from further analysis.

This merging step allowed each transaction to be enriched with product and store-level context.

3.4 Null handling

The parsing of size created missing values for some products where extraction was not possible. The new size_category column contained nulls. To resolve this, the **mode** was used for imputation. Since **16 OZ** was the most frequent size, all nulls were imputed with this value.

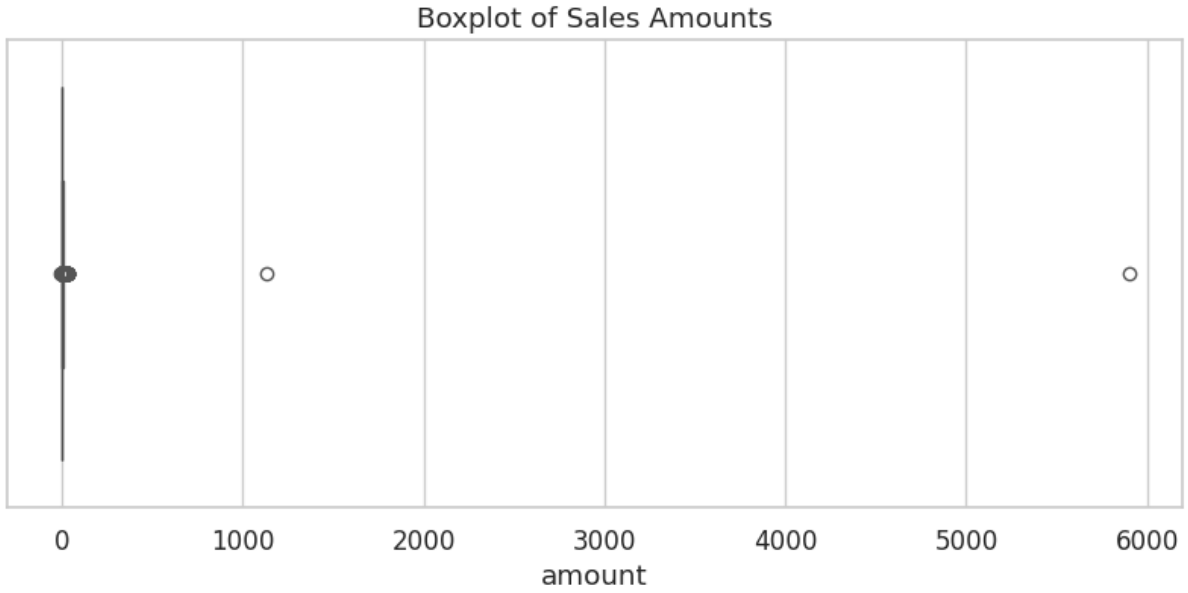
This ensured no missing values propagated into modeling features.

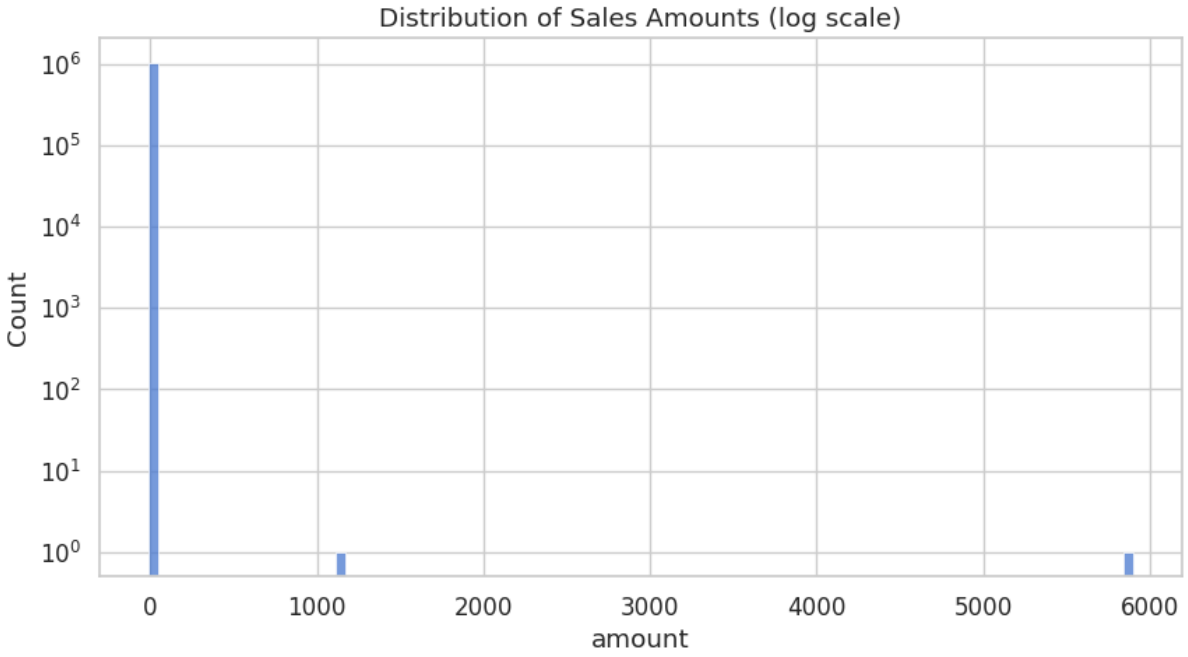
3.5 Outlier detection and removal

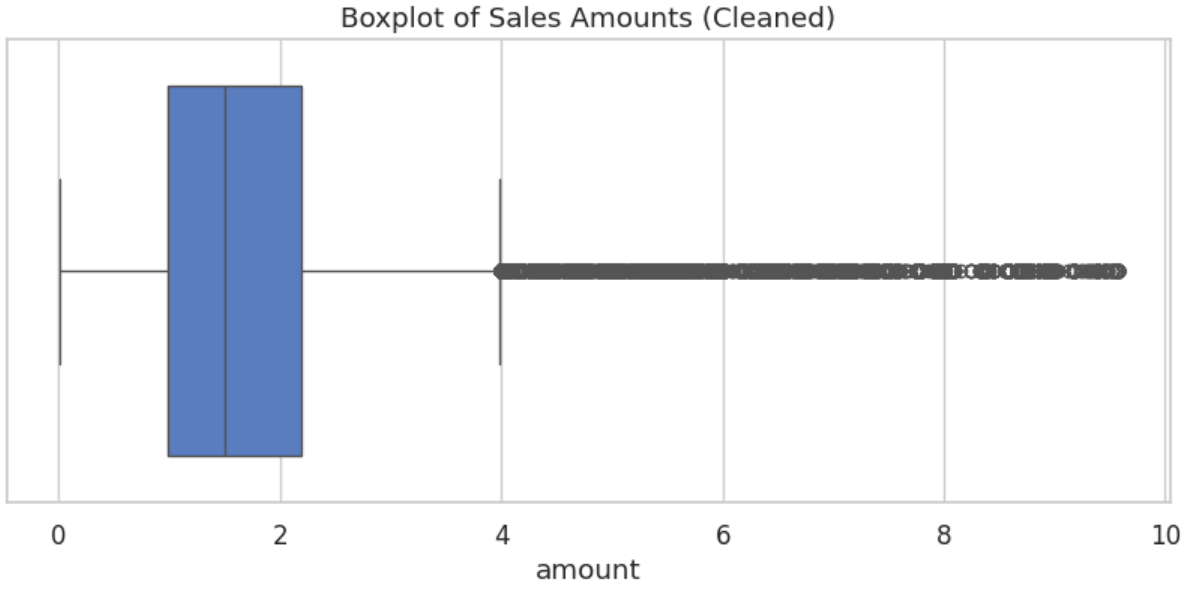
The amount field in the Sales dataset required careful inspection. Initial histograms and boxplots showed three main issues:

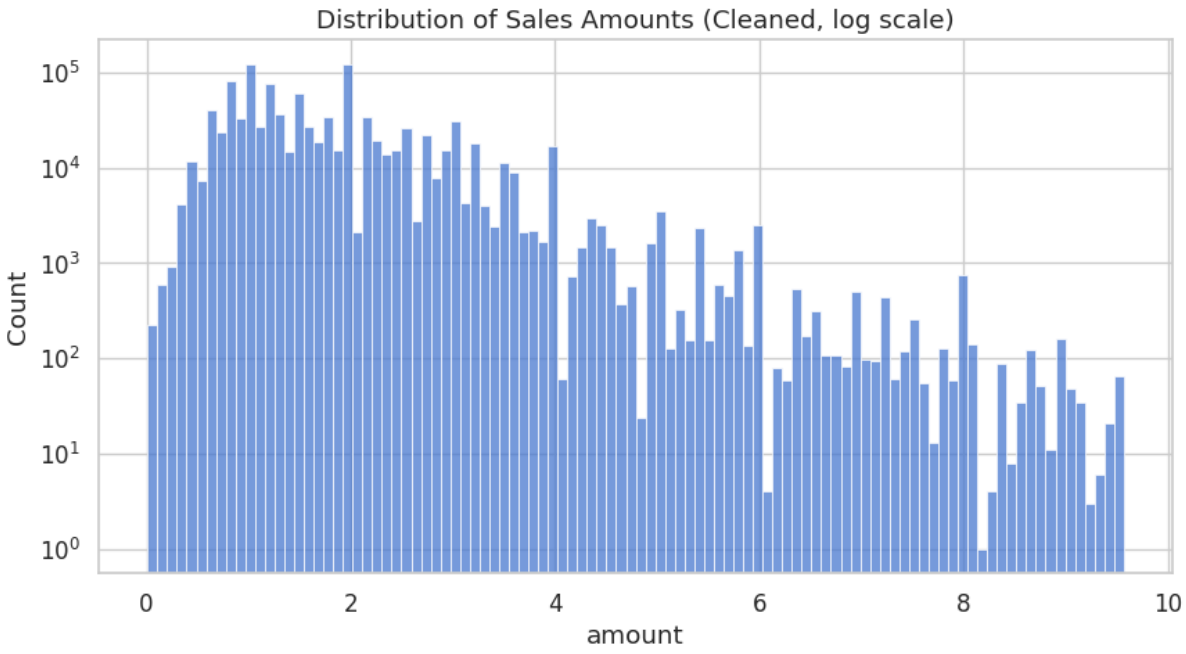
1. **Negative amounts**: A number of transactions had negative values, which are not valid for sales. These rows were removed.
2. **Extreme high outliers**: Some transactions had unrealistically large amounts, up to nearly 6000.
3. **Long tail distribution**: The majority of amounts clustered between 0 and 5, while a heavy tail distorted the distribution.

To address these, values above the **99.9th percentile** were removed. This clipping preserved the bulk of the data while eliminating implausible outliers. After cleaning, the distribution became smoother and more consistent.




Distribution of Sales Amounts (log scale)
Count
10^6
10^5
10^4
10^3
10^2
10^1
10^0
0
1000
2000
3000
4000
5000
6000
amount


Boxplot of Sales Amounts (Cleaned)
0
2
4
6
8
10
amount



3.6 Basket-level feature engineering

Sales were aggregated at the basket level to construct features for the basket revenue regression model. Key steps included:

- **Basket totals**: summing amounts to calculate basket_revenue.
- **Basket size**: counting the number of items (items_sum) per basket.
- **Voucher usage**: flag for whether a voucher was applied in the basket.
- **Aggregations by hour_bucket**: grouping basket totals by hour categories to study customer purchasing time windows.
- **Shares**: computing promo_share (ratio of items on promotion) and private_share (ratio of private label items).
- **Diversity measures**: number of unique brands and unique size categories within a basket.
- **Unit price average**: mean unit price across all items in the basket.

These features were combined with calendar and location features to form the input matrix for supervised regression.

3.7 Dataset consistency checks

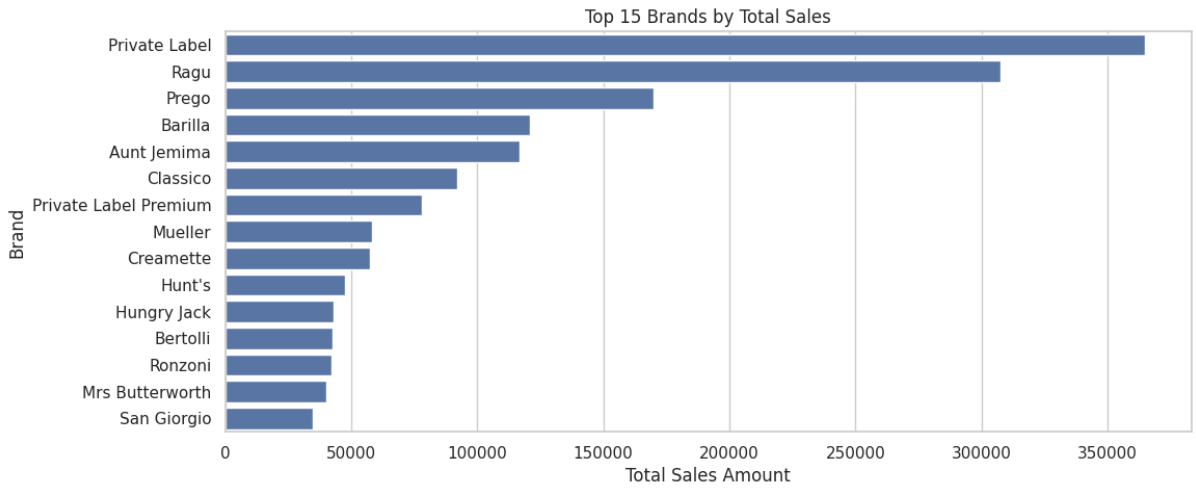
After cleaning, validation was performed to confirm dataset integrity:

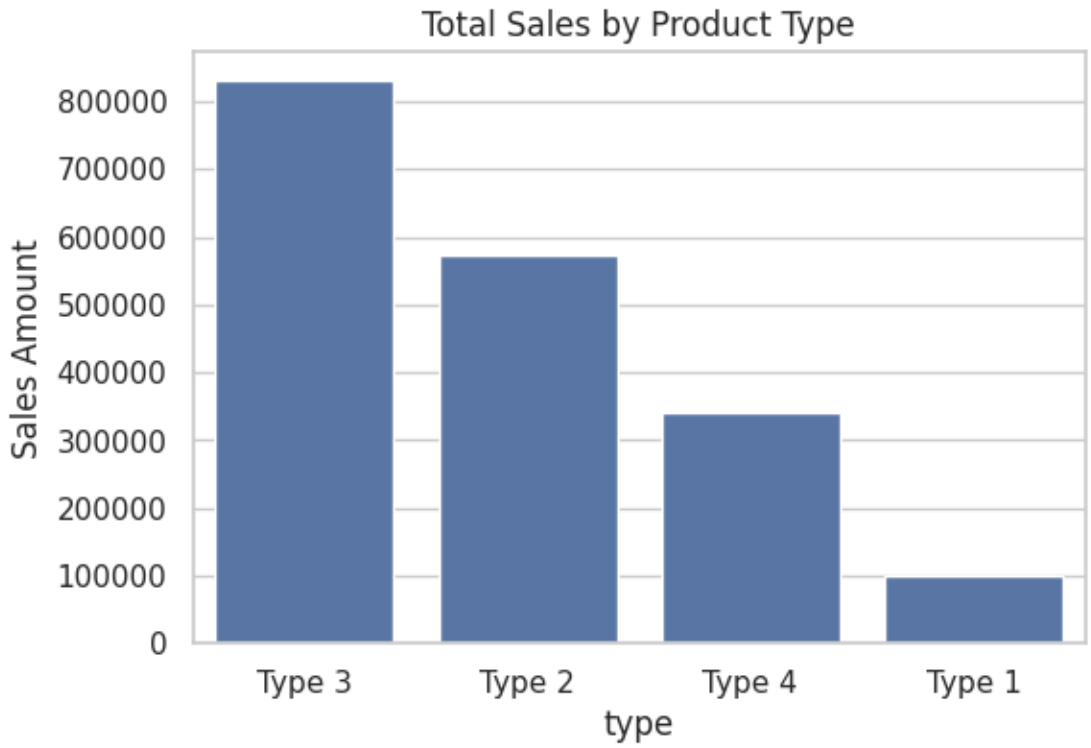
- All sales codes had matching entries in Items.
- All supermarket identifiers matched records in the Supermarkets dataset.
- Nulls in categorical variables were resolved through imputation.
- After merges, no row inflation occurred.
- Value ranges for sales amounts, units, and basket totals were consistent after outlier removal.

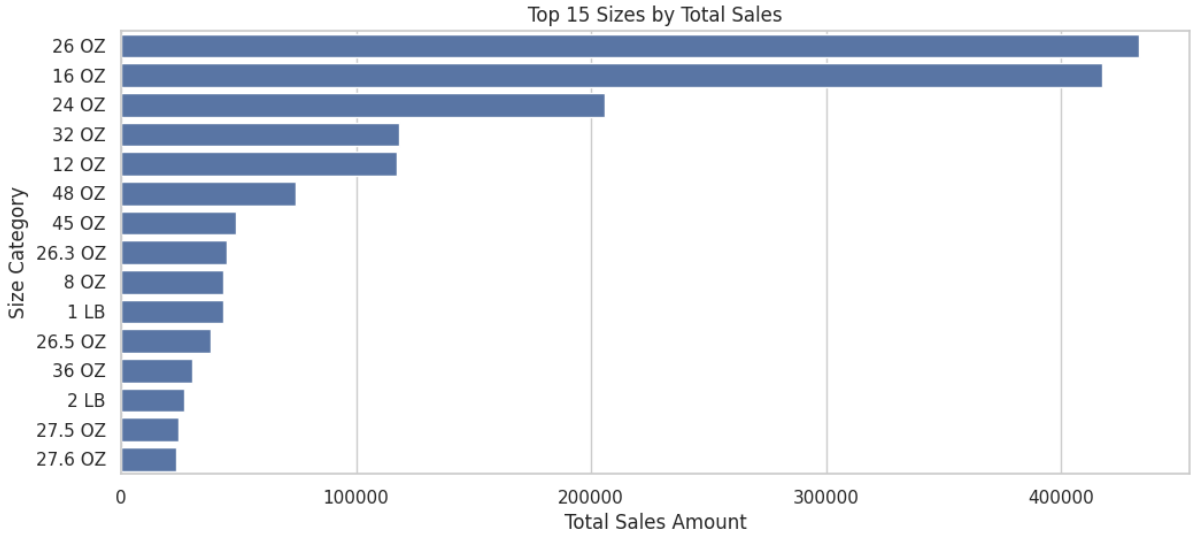
3.8 Exploratory consistency outputs

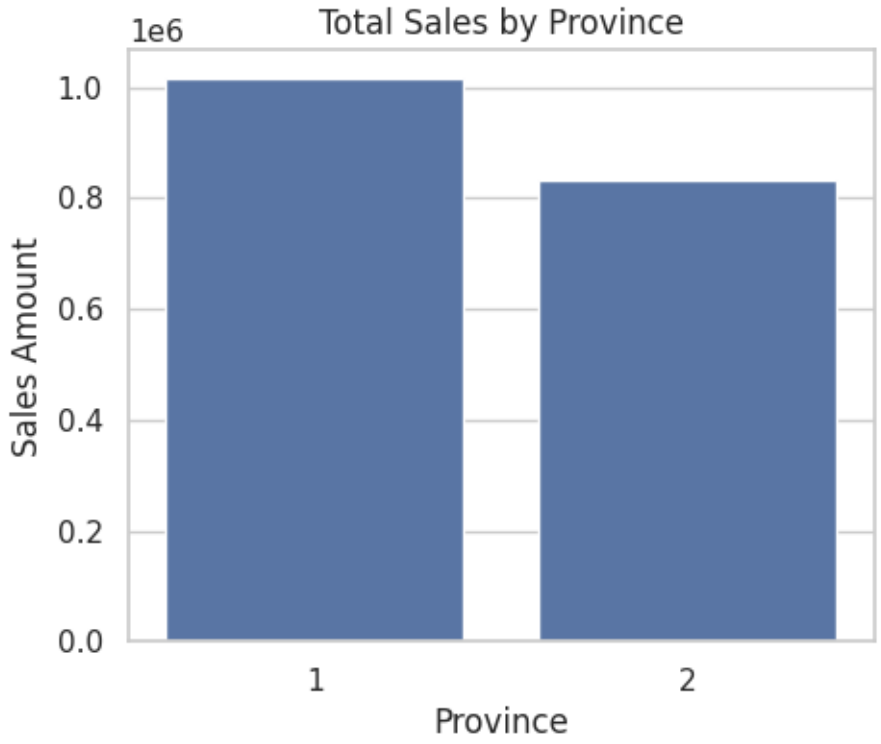
During cleaning, exploratory aggregations were run to validate transformations:

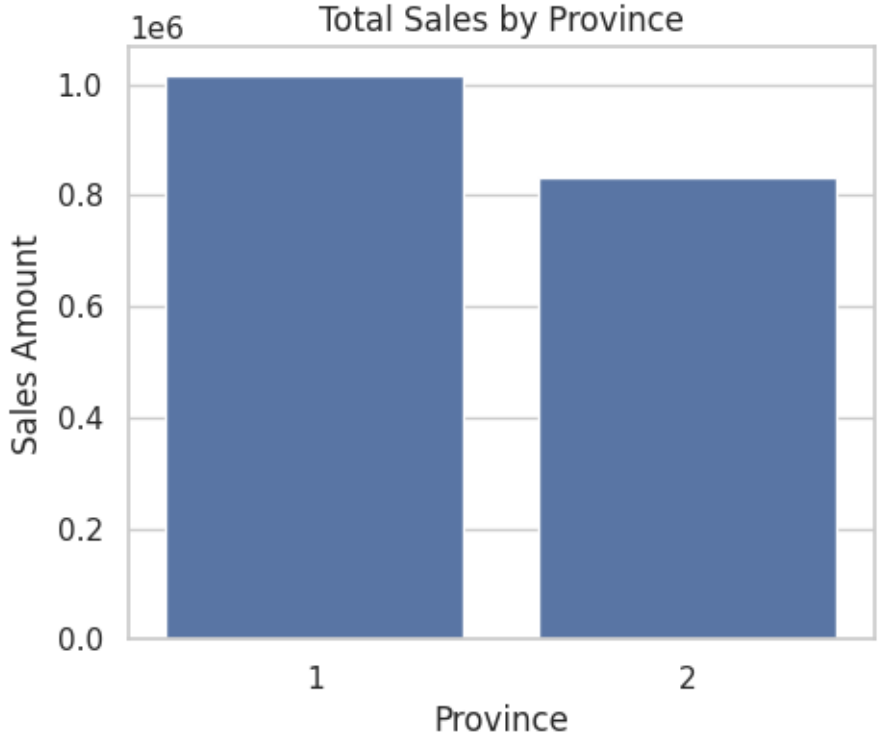
- **Top brands**: Private Label and Ragu emerged as leading brands by sales.
- **Product types**: Type 3 generated the highest revenue, followed by Type 2, Type 4, and Type 1.
- **Pack sizes**: The most revenue came from 16 OZ and 26 OZ packs.
- **Province differences**: Aggregated sales by province showed uneven contributions, confirming geographic variation.
- **Supermarkets**: The top 15 supermarkets by sales were identified, each contributing disproportionately to overall revenue.




Total Sales by Product Type
Sales Amount
800000
700000
600000
500000
400000
300000
200000
100000
0
Type 3
Type 2
Type 4
Type 1
type


Top 15 Sizes by Total Sales
Size Category
26 OZ
16 OZ
24 OZ
32 OZ
12 OZ
48 OZ
45 OZ
26.3 OZ
8 OZ
1 LB
26.5 OZ
36 OZ
2 LB
27.5 OZ
27.6 OZ
0
100000
200000
300000
400000
Total Sales Amount





3.9 Outcome of cleaning process

At the end of this stage, the dataset was:

- Unified with a single datetime field and multiple time-derived variables
- Enriched with size_category, product, and store-level features

- Free of negative or implausible sales amounts after removing outliers
- Complete with no nulls in critical categorical variables
- Structured with basket-level aggregations ready for regression modeling
- Validated through consistency checks and exploratory outputs

The result was a robust and consistent dataset, forming the foundation for exploratory data analysis and supervised learning.

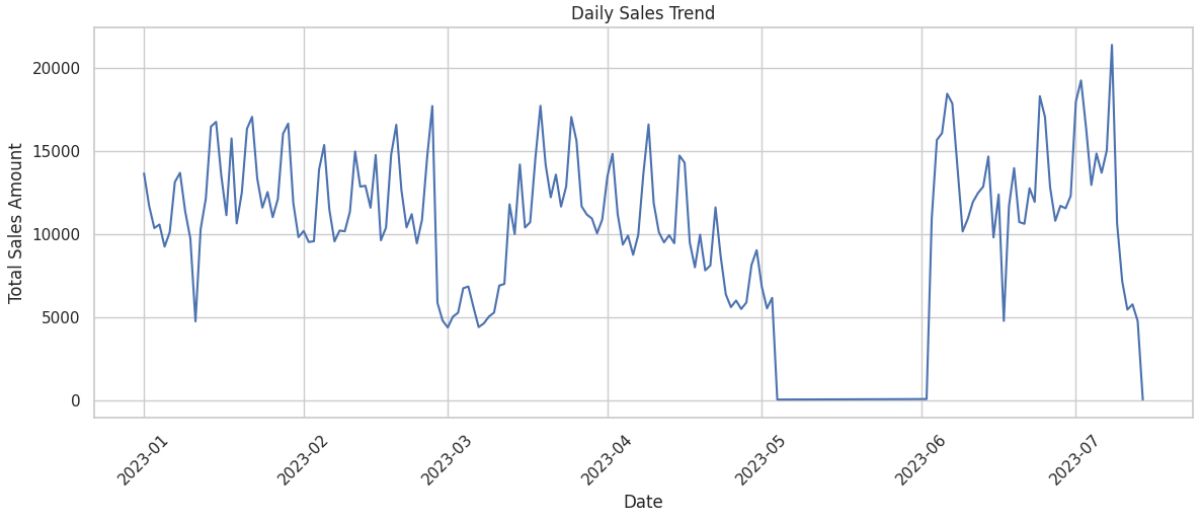
4. Exploratory Data Analysis

Exploratory Data Analysis (EDA) was conducted on the cleaned dataset to uncover patterns in customer shopping behavior, sales performance across products and stores, and customer loyalty segments. The analysis was structured into four themes: time-based, basket-level, customer-level, and RFM-based analysis.

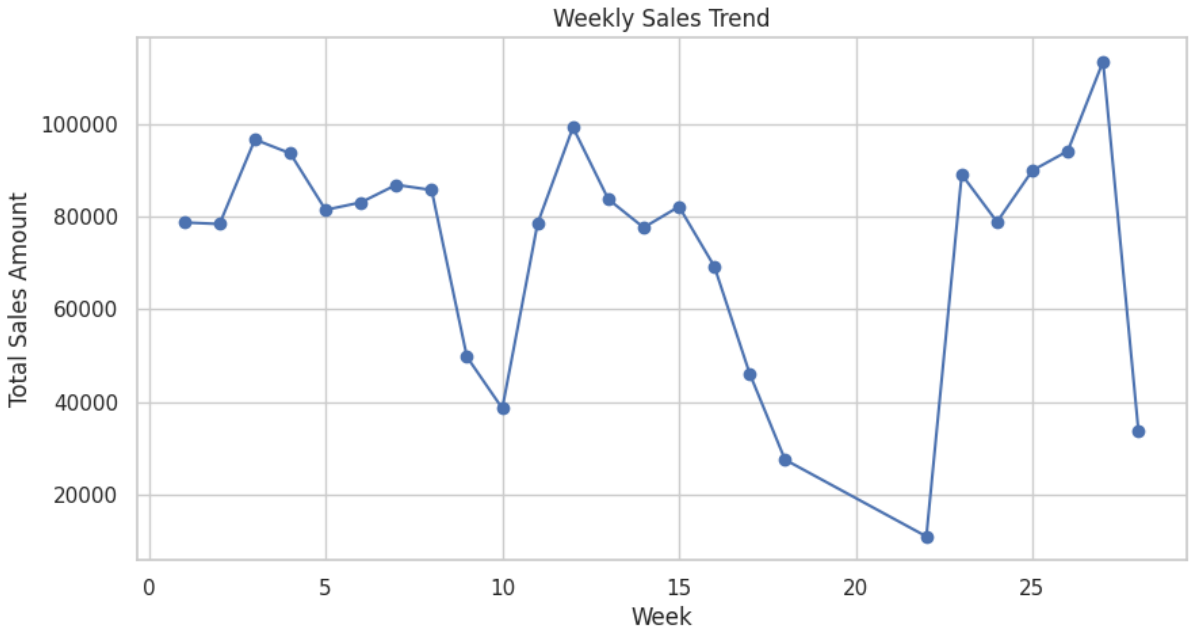
4.1 Time-based analysis

Time-based exploration was performed using the engineered features: day_of_week, month, is_weekend, and hour_bucket.

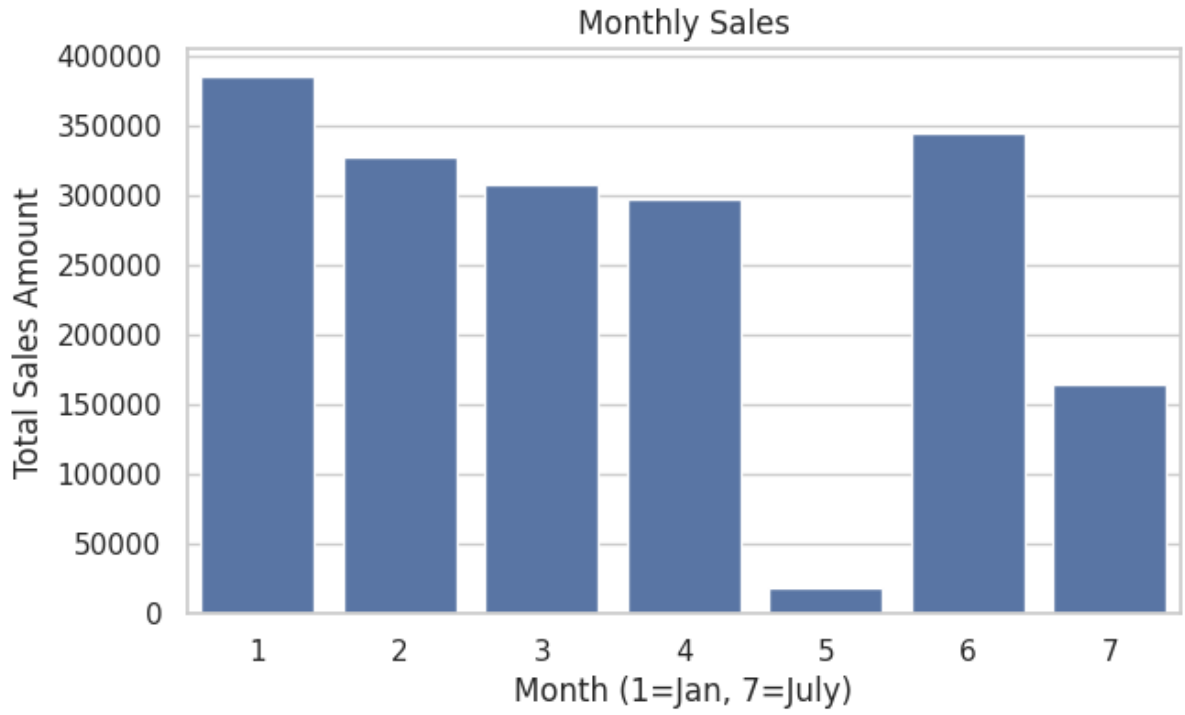
- **Daily trend**
  The daily sales trend showed stable transaction volumes during January to April, followed by a sharp drop in May where almost no sales were recorded. This aligned with the anomaly discovered earlier in the cleaning phase. After June, sales recovered, reaching the highest peaks in July.



- **Weekly trend**
  Weekly aggregation highlighted the same anomaly. Weeks 18 to 21 showed extremely low sales compared to surrounding weeks. This reinforced the decision to exclude May from the customer return model, ensuring training data was not distorted.

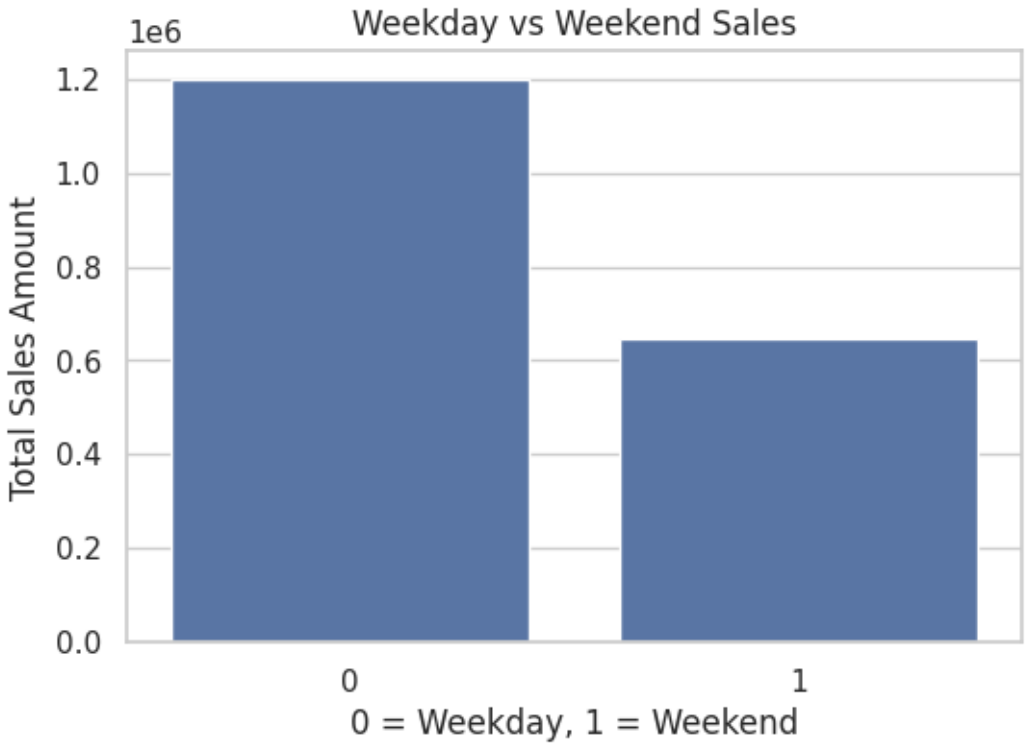


- **Monthly sales**
  January had the highest sales, followed by February and June. May was anomalously low, again confirming the missing data.

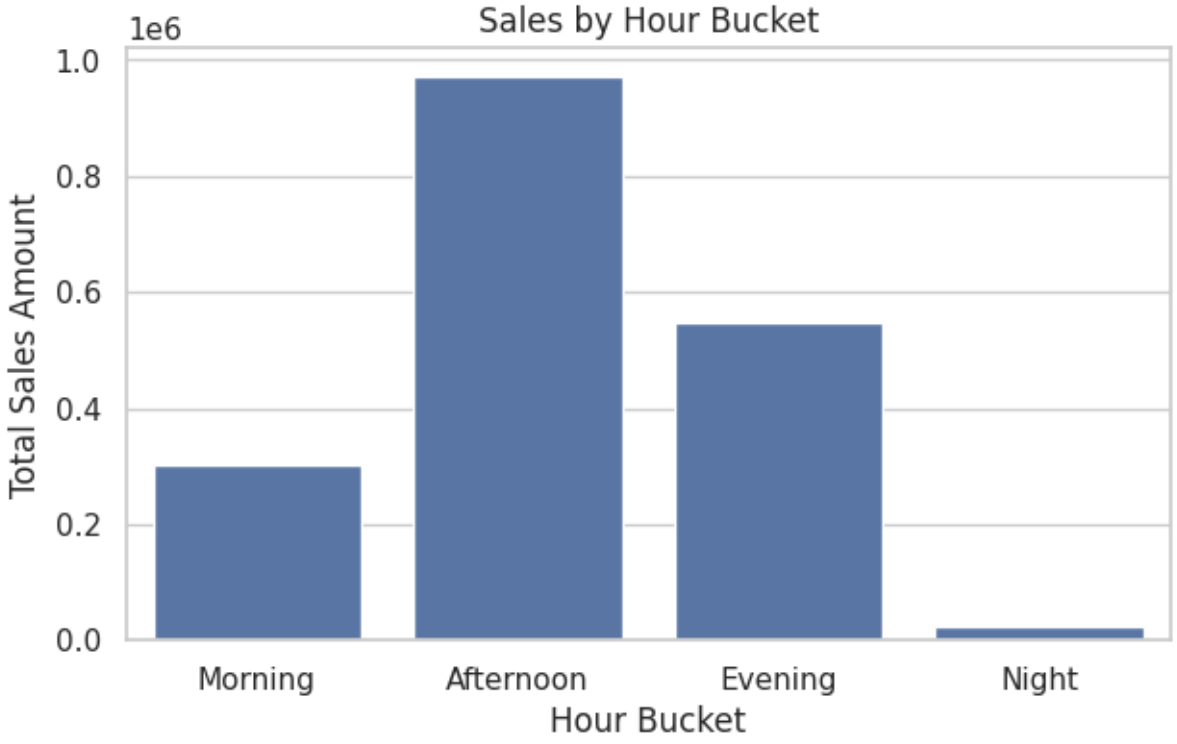


- **Weekday vs weekend**
  Comparing weekdays and weekends, total weekday sales (~1.2M) were almost

double weekend sales (~0.65M). This suggests that most customers shopped on weekdays, possibly after work or during regular grocery runs.



- **Hour bucket**
  Sales peaked in the Afternoon slot (~1M), followed by Evening (~0.55M), then Morning (~0.3M). Night sales were negligible. Afternoon shopping represented the core of supermarket activity.

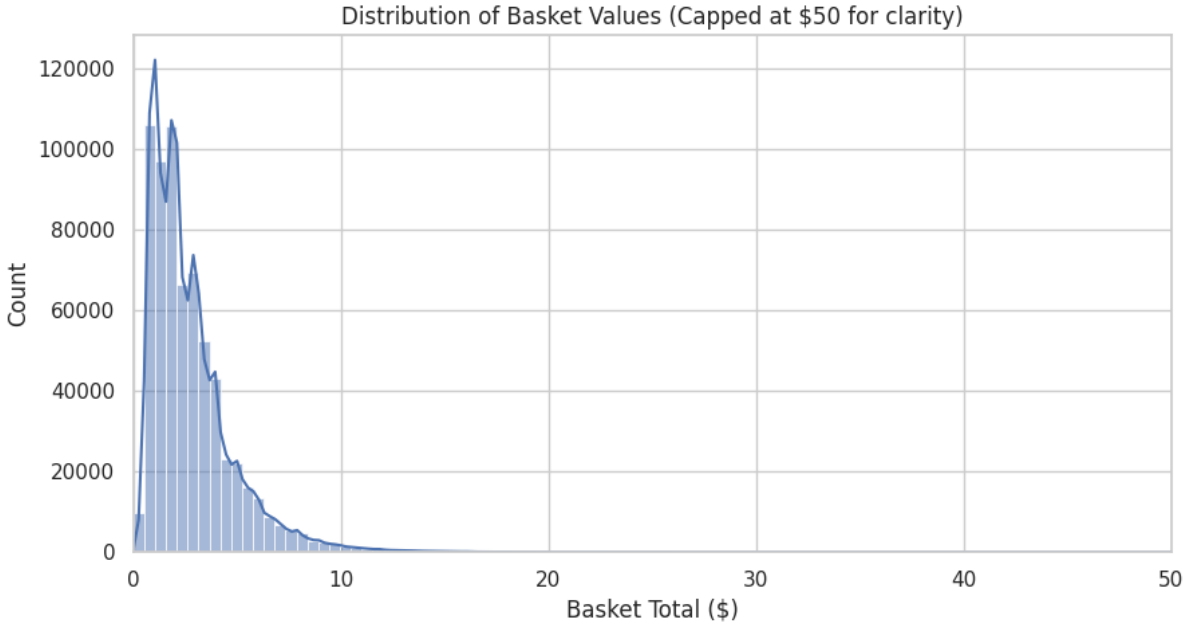


Relevance for modeling: Temporal features such as month, weekday, and hour_bucket were validated as predictive signals for both basket revenue forecasting and customer return classification.

4.2 Basket-level analysis

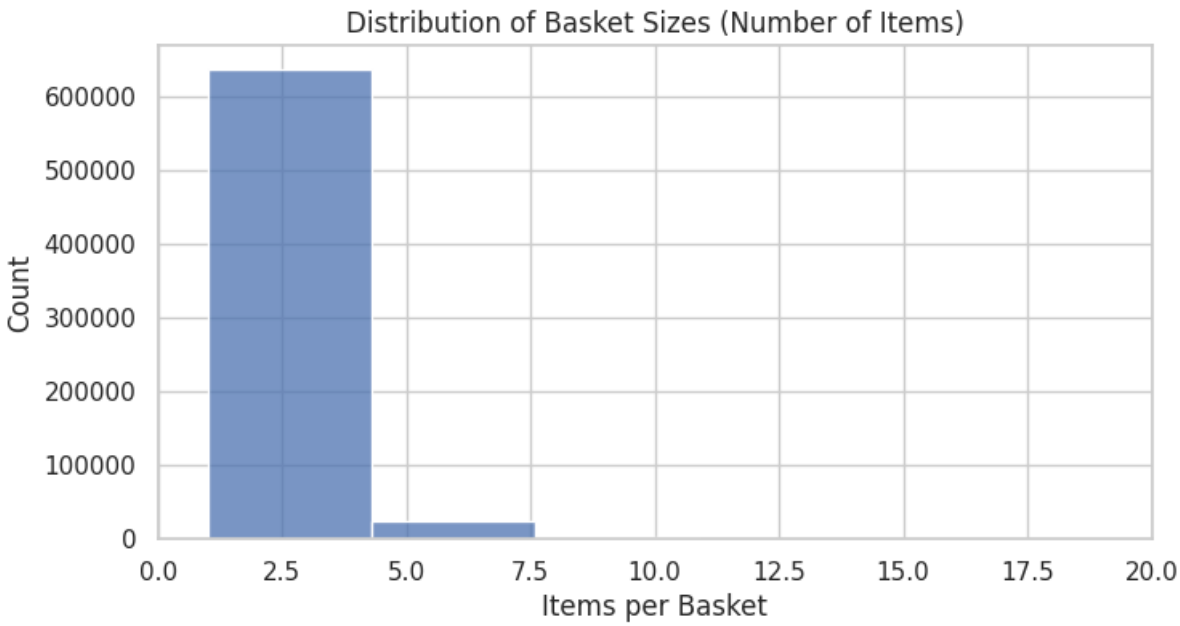
Basket-level analysis was done using aggregated transaction-level features.

- **Distribution of basket values**
  Most baskets had a total value below $10, with a steeply right-skewed distribution. A long tail extended up to $50 and beyond, but these cases were rare.

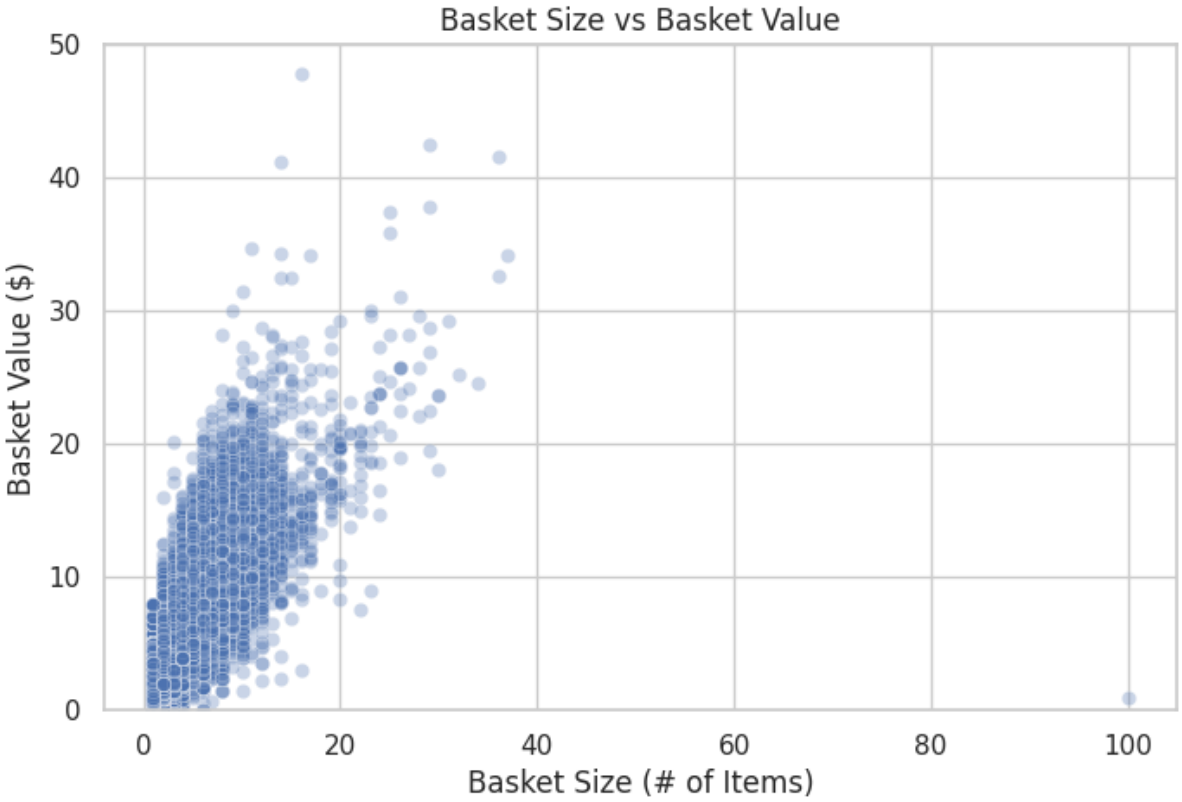


- **Basket size distribution**
  Most baskets contained 2 or 3 items. Very few baskets exceeded 7 items. This confirmed that the supermarket operated in a low-basket-size, high-volume

environment.



- **Basket size vs basket value**
  A positive correlation was observed. Larger baskets consistently had higher values, though with diminishing returns after ~20 items. Outliers existed, such as a single basket with nearly 100 items.

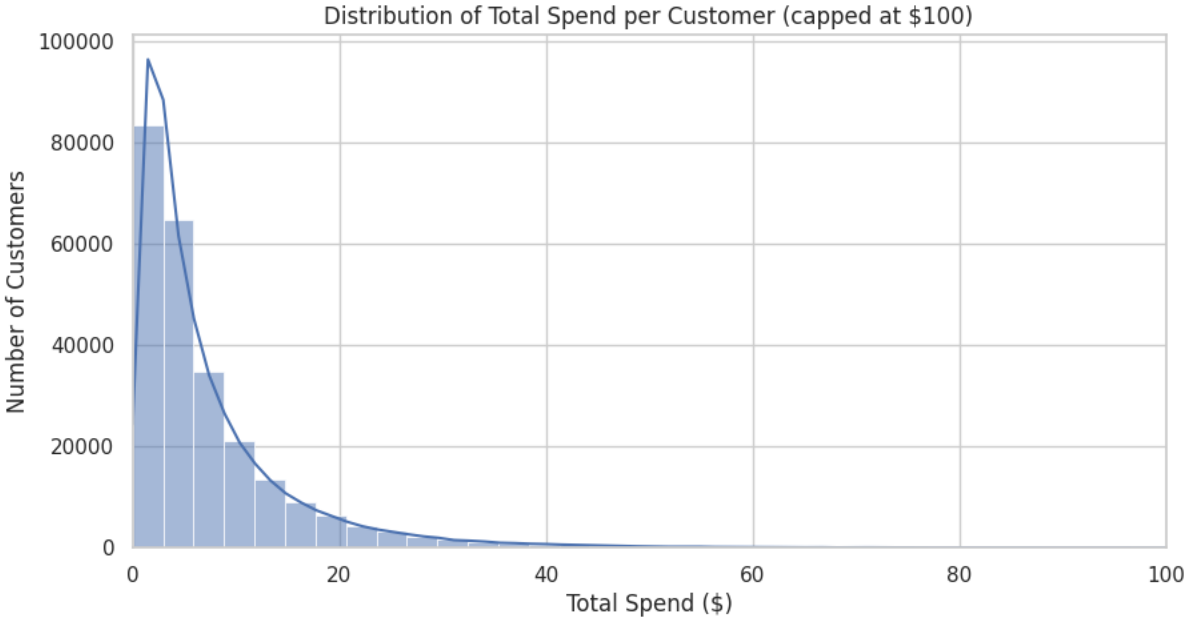


Relevance for modeling: Basket size, item diversity, and average unit price were later proven to be strong predictors of basket revenue.

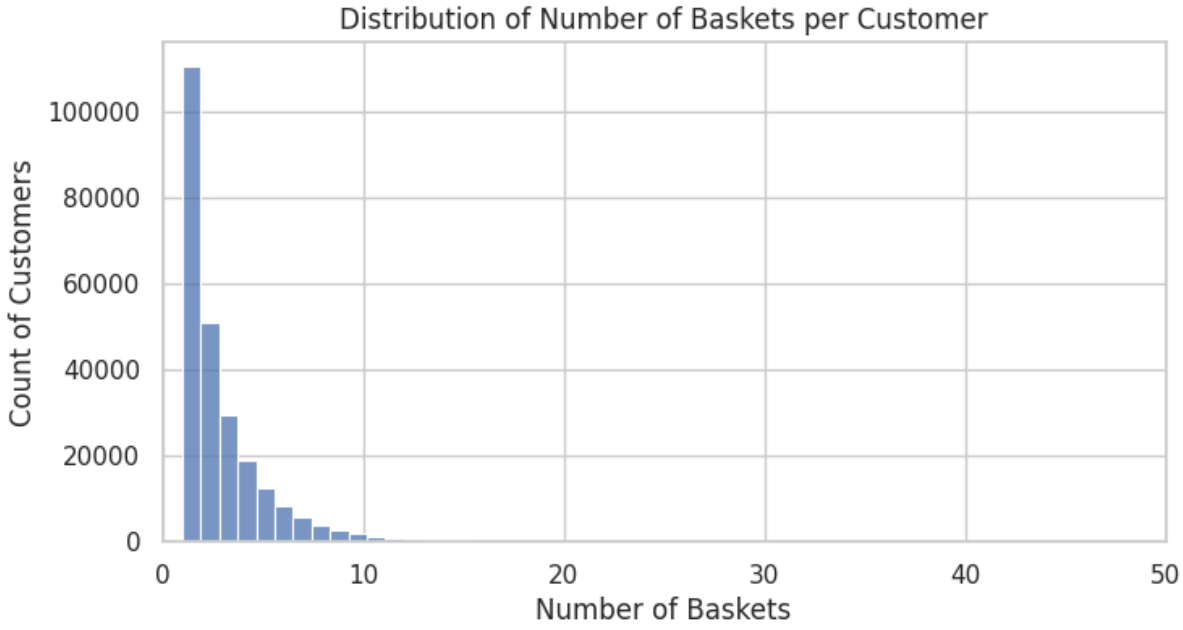
4.3 Customer-level analysis

Customer-level analysis provided insight into heterogeneity across shoppers.

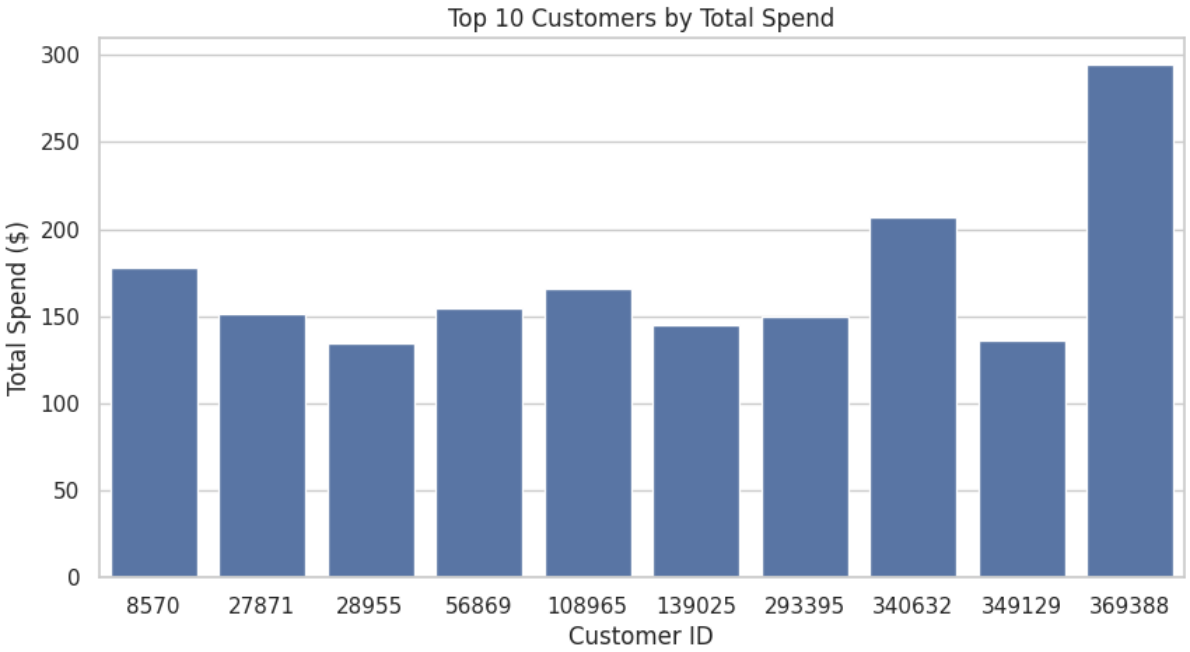
- **Distribution of total spend per customer**
  The majority of customers spent less than $20 across the 2-year period. The distribution was right-skewed, with a long tail of high-value customers spending up to $100 or more.



- **Number of baskets per customer**
  Most customers had only 1 or 2 baskets in the dataset. A small fraction returned frequently, making multiple purchases.



- **Top 10 customers by spend**
  The top customer spent nearly $300, while others in the top 10 spent between $130 and $210. This small group contributed disproportionately to revenue.

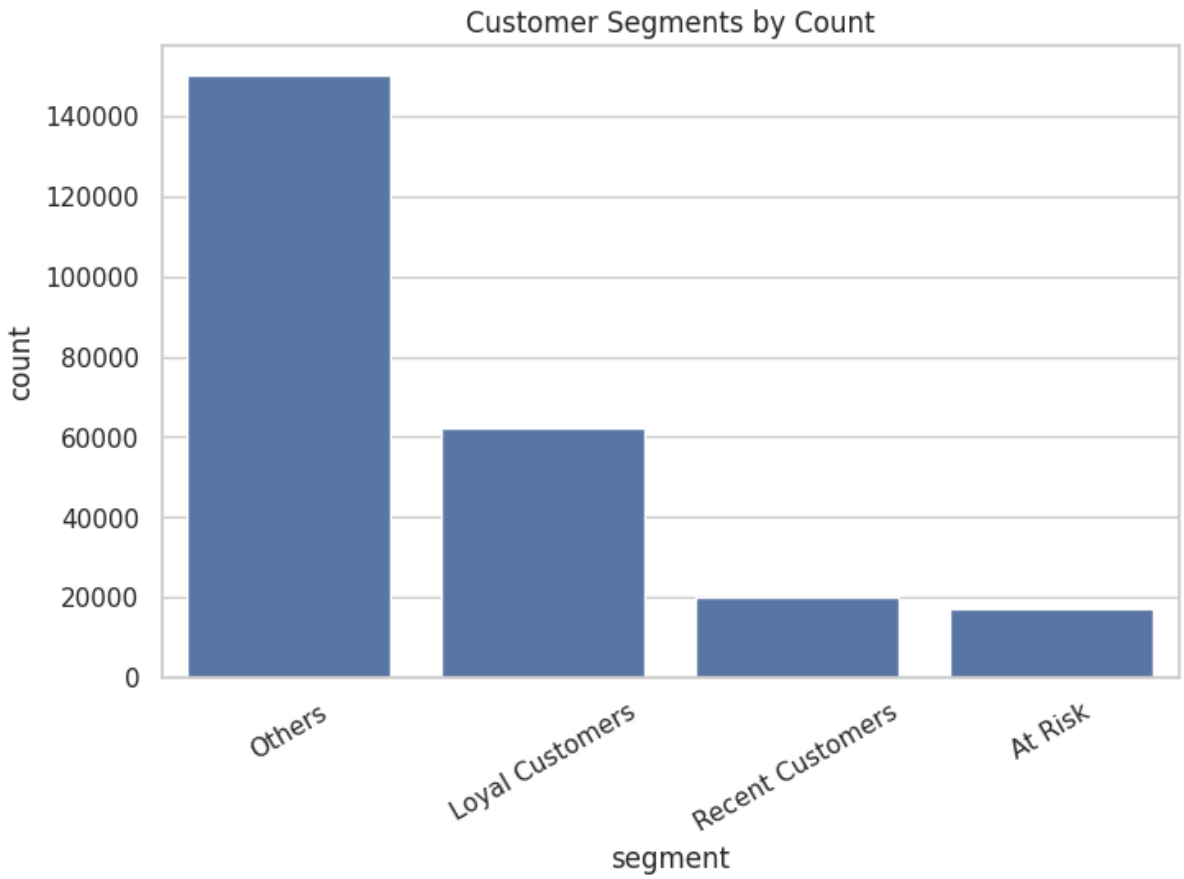


- **Relevance for modeling**: These insights motivated the customer return classification problem, as understanding whether a customer will return in the next 30 days is critical for supermarket revenue.

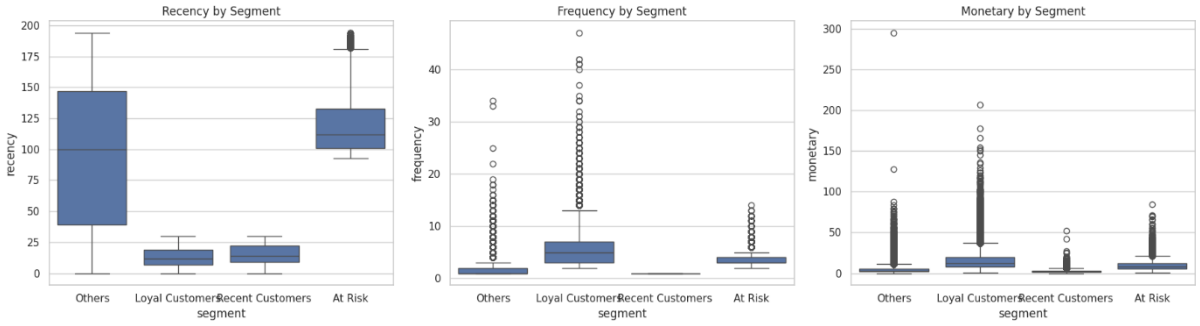
4.4 RFM segmentation analysis

To better understand customer loyalty, RFM (Recency, Frequency, Monetary) segmentation was performed.

- **Segments created**: Customers were grouped into Loyal Customers, Recent Customers, At Risk, and Others based on their recency, frequency, and spend levels.

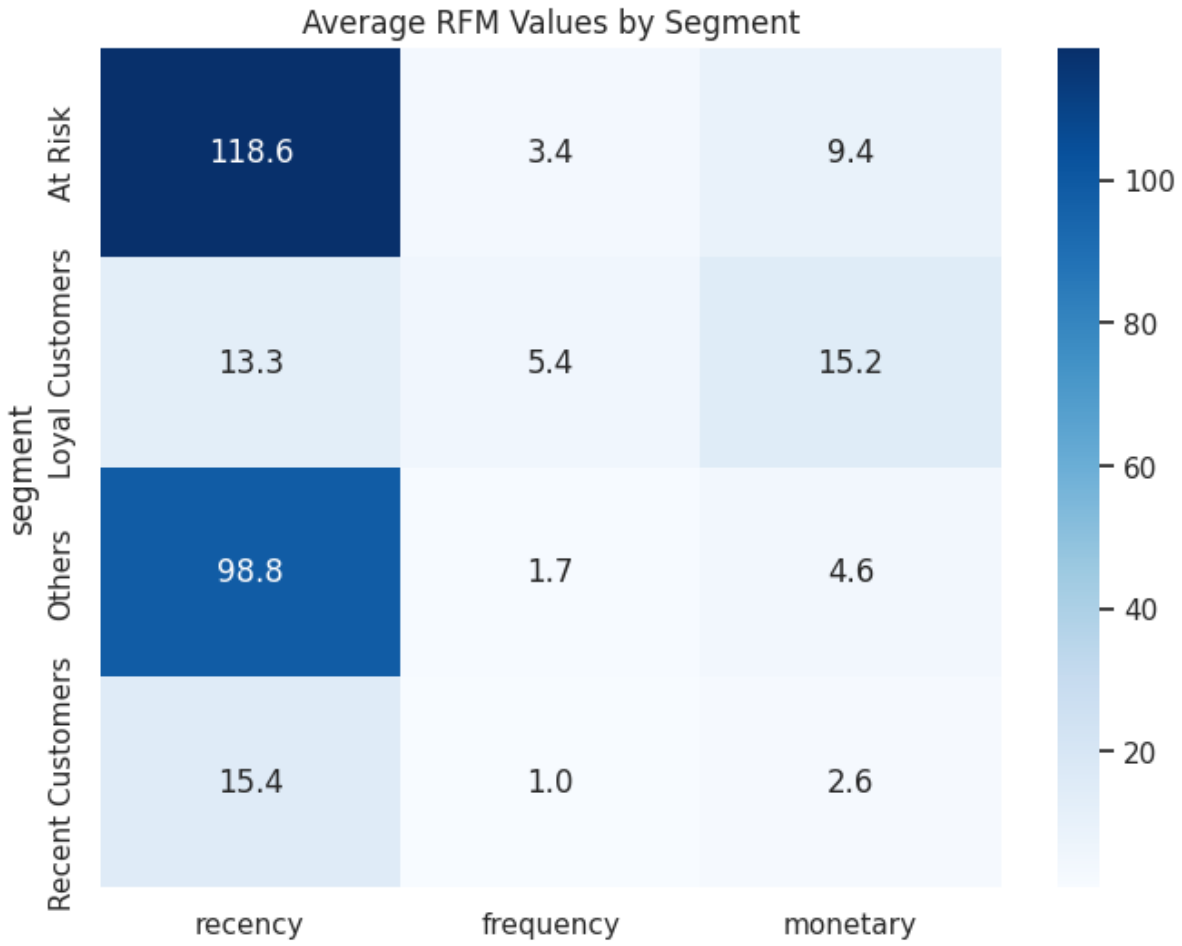


- **Boxplots by segment**: Loyal Customers had low recency (recent transactions), high frequency, and high monetary values. At Risk customers had high recency (long time since last visit) but moderate spend. Recent Customers had low recency but low frequency and monetary scores. Others were consistently low across all three metrics.



- **Average RFM heatmap**: Loyal Customers averaged frequency ~5.4 and monetary ~15.2, the highest across all groups. At Risk customers averaged recency ~118.6,

confirming their churn risk.



Relevance for modeling: RFM confirmed that frequency and spend behavior are unevenly distributed across customers, highlighting the value of customer return prediction.

4.5 Key takeaways from EDA

The exploratory analysis highlighted several important patterns in customer behavior and sales performance. These insights guided the feature engineering strategy and informed the selection of supervised learning tasks.

Time-based patterns

- **Afternoon shopping dominates**: The Afternoon time bucket contributed nearly half of all sales revenue, while Night sales were negligible. This suggests that supermarkets may need to optimize staffing and promotions around afternoon and evening peaks.
- **Weekday sales exceed weekend sales**: Weekday transactions were almost double weekend volumes, which is unusual for grocery retail. This indicates that customers in this dataset treat the supermarket more as a routine necessity (weekday shopping after work or during lunch hours) than a weekend family shopping destination.

- **Monthly and seasonal anomalies**: January and February had the highest sales, while May had almost none due to missing or unrecorded transactions. This anomaly was critical to detect early, because including May would have biased the models.

Implication for modeling: Time-based features such as day_of_week, month, and hour_bucket were validated as strong predictors. They capture meaningful seasonality and weekly rhythms in customer shopping behavior.

Basket-level behavior

- **Small basket dominance**: The vast majority of baskets contained only 2–3 items and were worth under $10. This confirmed that the supermarket environment is characterized by high-frequency, low-ticket transactions.
- **Scaling effect**: Basket value increased with size, but not linearly. Larger baskets did not scale proportionally with the number of items, likely due to the presence of low-priced staple items.
- **Item diversity and pricing power**: Baskets with higher brand diversity and higher average unit price contributed significantly more revenue.

Implication for modeling: Features such as items_sum, avg_unit_price, brand_diversity, and size_diversity are crucial for predicting basket revenue. They directly capture customer choice behavior at the transaction level.

Customer heterogeneity

- **Highly skewed spend distribution**: Most customers spent very little over the 2-year period, while a small fraction spent disproportionately high amounts. This classic long-tail distribution indicates that supermarkets rely heavily on a small set of loyal high-value customers.
- **Top customers are critical**: The top 10 customers contributed hundreds of dollars each, far exceeding the median. Retaining or losing just a handful of these customers could materially impact revenue.
- **Low frequency majority**: Most customers had only 1–2 baskets, suggesting that churn is very high in this dataset.

Implication for modeling: These findings justified building a **customer return prediction model**. By predicting which customers are likely to return within 30 days, supermarkets can target retention strategies toward the high-value minority and reduce churn in the low-frequency majority.

RFM segmentation insights

- **Loyal Customers**: Low recency, high frequency, high monetary value. They are the backbone of the business and should be protected with loyalty programs.
- **At Risk Customers**: Very high recency (long time since last purchase). They represent potential churn and may need win-back campaigns.
- **Recent Customers**: Low recency but low frequency and monetary. They are in an early stage of their customer lifecycle and could be nurtured into becoming loyal.
- **Others**: Low across all metrics. Likely one-time or disengaged shoppers.

Implication for modeling: RFM features (recency_days, rolling_30d_amount, rolling_30d_freq, tenure_days, avg_basket_value) became the foundation of the customer return classifier. They not only capture behavior but also align directly with actionable customer segments.

Overall strategic insights

1. **Peak shopping hours and weekdays dominate sales**, so store operations and promotions should be aligned accordingly.
2. **Basket revenue is driven by a few key levers** — number of items, average unit price, and product diversity — making them essential predictive features.
3. **Revenue concentration in a small set of loyal customers** means customer retention has outsized business impact.
4. **Churn risk is visible in RFM analysis**, justifying targeted machine learning models that predict return likelihood and allow proactive interventions.

5. Supervised Learning Models

This section narrates exactly what was done for the three supervised learning problems, including the feature sets, train and test protocols, every major iteration, and the final metrics printed by the code. Numbers are reported exactly as computed.

5.1 Basket revenue forecast

Problem statement

Predict the total revenue of a basket from its composition and context.

Dataset and features

A basket level table was built from cleaned line items. The target was basket_revenue which is the sum of amount × units within each basket. Predictors used in the final matrix:

- items_sum
- n_products
- voucher_used
- is_weekend
- hour
- brand_diversity
- size_diversity
- avg_unit_price
- promo_share
- private_share
- supermarket
- province
- month

Supermarket, province, and month were ordinal encoded. Train and test split was 80 to 20 with random_state 42.

Models and exact results

Three models were trained on the same split and evaluated on the same test rows.

- Linear Regression
    - MAE 0.791
    - RMSE 1.476

    - R squared 0.882
- Random Forest Regressor
    - MAE 0.094
    - RMSE 0.537
    - R squared 0.984
- Gradient Boosting Regressor
    - MAE 0.133
    - RMSE 0.549
    - R squared 0.984

What drives predictions

Gradient Boosting feature importances printed by the code:

- items_sum 0.647
- avg_unit_price 0.349
- promo_share 0.002
- n_products 0.001
- all others near 0.000

Takeaways for the business

- Basket revenue is explained almost entirely by how many items are in the basket and the average unit price that the customer pays.
- Promotion share and other contextual variables provided little incremental lift in this dataset.
- Either Random Forest or Gradient Boosting delivers very accurate forecasts with R squared near 0.984. Random Forest was selected as the working model because it matched accuracy and is robust to non linear effects.

5.2 Customer return within 30 days

Problem statement

Predict whether the customer will return for another purchase within 30 days of the current purchase.

Labeling and snapshot

Labels were created using the previous and next purchase date per customer. The label will_return_30d equals 1 if days_to_next is less than or equal to 30. The snapshot used for modeling had:

- shape 409725 by 10
- positive rate 0.6128

Each row contained these key fields or features used for modeling and diagnostics:

- customer_id
- date
- amount_sum
- items_sum
- baskets_n
- prev_date
- next_date
- recency_days
- days_to_next
- will_return_30d

Iterations and learning from failures

1. Baseline Logistic Regression on the initial snapshot

- Modest accuracy and AUC.

2. Class weighted Logistic Regression

- Balanced class weights gave only small gains.

3. Logistic with additional penalty for misclassification

- Heavier penalty on false negatives improved recall slightly but overall performance remained limited.

4. Gradient Boosting Classifier

- Performance degraded versus Logistic Regression.

5. Tuned Gradient Boosting

- Further tuning led to even worse scores, indicating a poor match for this label.

6. Data audit

- A daily and weekly audit revealed missing transactions in May.

- May rows would corrupt days_to_next and therefore the label.
- Decision taken to exclude May from the snapshot used for training and testing.

7. Rebuild snapshot and retrain Logistic and Random Forest with balanced weights

- After excluding May, both models were retrained and evaluated.
- Random Forest showed the best balance of precision and recall.

Final printed metrics from the code

Logistic Regression

- AUC 0.6158472080656465
- PR AUC 0.7409983948439095
- Classification report on test set
    - class 0 precision 0.45 recall 0.67 f1 0.54 support 30981
    - class 1 precision 0.71 recall 0.49 f1 0.58 support 50198
    - accuracy 0.56
    - macro avg precision 0.58 recall 0.58 f1 0.56
    - weighted avg precision 0.61 recall 0.56 f1 0.56

Random Forest Classifier

- AUC 0.6698797306740708
- PR AUC 0.7940415455313475
- Classification report on test set
    - class 0 precision 0.48 recall 0.78 f1 0.59 support 30981
    - class 1 precision 0.78 recall 0.48 f1 0.59 support 50198
    - accuracy 0.59
    - macro avg precision 0.63 recall 0.63 f1 0.59
    - weighted avg precision 0.66 recall 0.59 f1 0.59

What the numbers say

- Random Forest improves both AUC and PR AUC over Logistic Regression in this dataset after the May exclusion and class balancing.
- The precision recall trade off is asymmetric which is expected in churn style labels where the positive rate is high but sequences are short and noisy.

- The strongest signals were recency_days, rolling 30 day amount, and rolling 30 day frequency.
- Even with modest headline accuracy, ranking customers by Random Forest probability is useful to trigger offers for the at risk cohort.

5.3 High performing supermarkets

Problem statement

Identify supermarkets that are high performing based on revenue. High performing is defined as stores at or above the 75th percentile of revenue.

Features and target

A store level table was built with:

- transactions
- customers
- units
- revenue
- avg_basket_size
- avg_basket_value
- weekend_share
- voucher_rate
- avg_unit_price
- unique_products
- avg_items_per_basket

Target high_perf equals 1 if revenue is greater than or equal to the 75th percentile. Class counts in the modeling sample:

- class 0 other 282
- class 1 high performer 94

Train and test split was stratified 70 to 30.

Models and exact results

Two models were trained and evaluated on the same split.

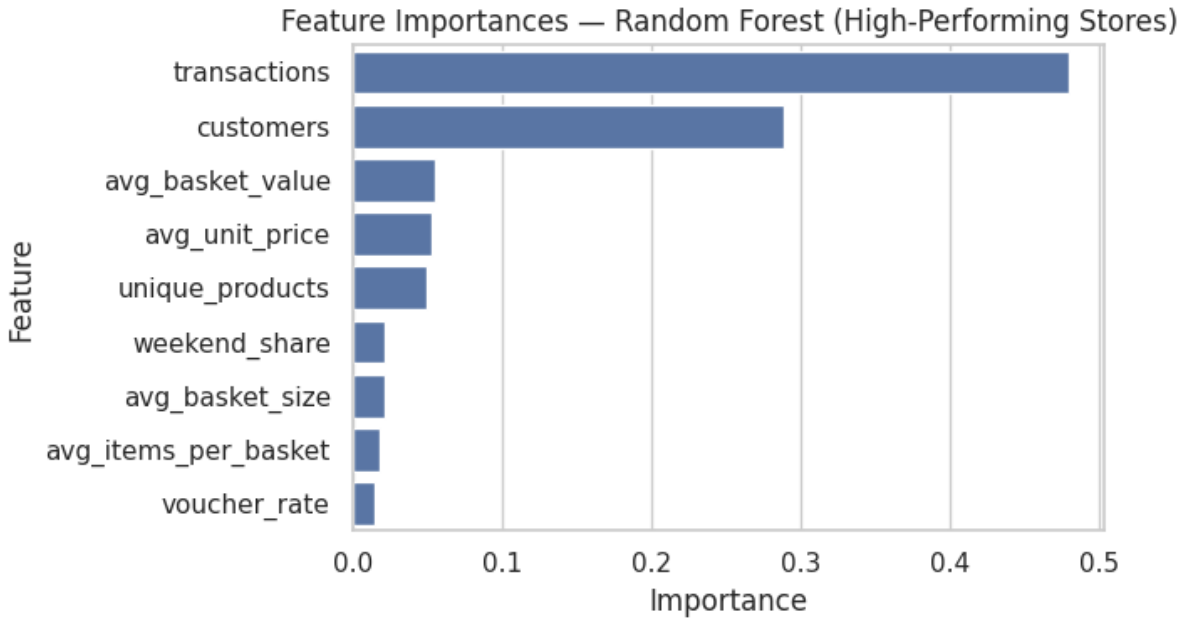
Logistic Regression with class weight balanced

- AUC 0.9895

- Test classification report
    - class 0 precision 0.97 recall 0.92 f1 0.95 support 85
    - class 1 precision 0.79 recall 0.93 f1 0.85 support 28
    - accuracy 0.92
    - macro avg precision 0.88 recall 0.92 f1 0.90

Random Forest Classifier

- AUC 0.9966
- Test classification report
    - class 0 precision 0.95 recall 0.99 f1 0.97 support 85
    - class 1 precision 0.96 recall 0.86 f1 0.91 support 28
    - accuracy 0.96
    - macro avg precision 0.96 recall 0.92 f1 0.94



Top Random Forest feature importances printed by the code:

- transactions 0.4790
- customers 0.2888
- avg_basket_value 0.0547
- avg_unit_price 0.0526
- unique_products 0.0497
- weekend_share 0.0219

- avg_basket_size 0.0214
- avg_items_per_basket 0.0174
- voucher_rate 0.0144

The model also produced a ranked list of stores by probability of being high performing. For example, supermarkets 359, 321, 326, 285, and 259 had the highest predicted probabilities and were labeled as high performers in the ground truth.

What this means operationally

- Store throughput and shopper base are the primary discriminators of high performance. Transactions and unique customers dominate the model.
- Ticket size and breadth of assortment contribute next.
- Weekend share and voucher rate were comparatively minor.
- The probability ranking enables a clean scoreboard for regional teams to focus on replication and lift initiatives.

5.4 Cross problem summary and next actions

- Basket revenue forecasting achieved very high fidelity with R squared near 0.984 using Random Forest or Gradient Boosting. Focus on the levers that change items per basket and average unit price.
- Customer return within 30 days is a harder signal. After removing May and balancing classes, Random Forest achieved AUC 0.6699 and PR AUC 0.7940 on the test set. This is strong enough to drive a ranked outreach list for retention campaigns.
- High performing supermarket classification reached AUC 0.9966 with Random Forest and accuracy 0.96 on the test set. Use the feature importance profile to benchmark stores and to diagnose gaps.

6. Reinforcement Learning Task

This section explains in full detail how the reinforcement learning experiment was conducted, how the agent learned, the tuning decisions, and what we observed at each stage. The task was to train an agent using **Q-learning** to navigate a grid maze from a starting point to a goal, avoiding walls and unnecessary steps.

6.1 Maze Environment Setup

We worked with a **10 × 10 maze** represented as a matrix:

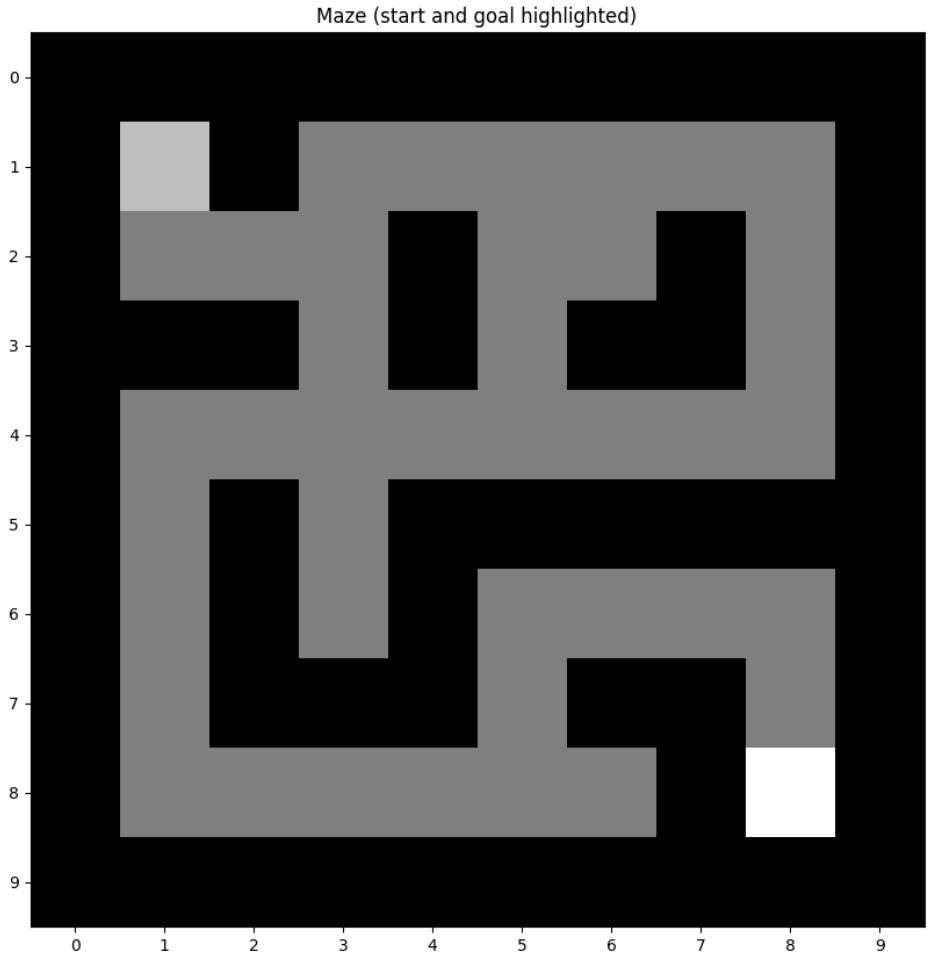
- **Free cells** were coded as 0
- **Walls/obstacles** were coded as 1
- **Start state**: coordinate (1, 1)
- **Goal state**: coordinate (8, 8)

The environment was encoded so that each cell (row, col) was flattened into a **single state index** using:

$s = row \times cols + col$

This produced **100 unique states**, each with **4 possible actions**. The **Q-table** was therefore initialised with shape (100, 4), filled with zeros at the beginning.

Validation checks confirmed that both the start and goal states were valid (walkable, not walls).



6.2 Action Space and Rewards

The action space consisted of **4 discrete moves**:

- 0 = Up (−1, 0)
- 1 = Down (+1, 0)
- 2 = Left (0, −1)
- 3 = Right (0, +1)

Reward structure

Designing the reward function was crucial. The following values were used:

- **+10 reward** for reaching the goal
- **−5 penalty** for attempting to move into a wall
- **−0.1 penalty** for every step taken

This ensured the agent learned to:

- Seek the goal as quickly as possible (positive reinforcement).
- Avoid walls (large negative feedback).

- Avoid wandering/looping (small per-step penalty).

6.3 Q-Learning Configuration

The learning followed the standard **Q-learning update rule**:

$Q(s,a) \leftarrow Q(s,a) + \alpha \times \Big(reward + \gamma \max_{a'} Q(s',a') - Q(s,a)\Big)$

Hyperparameters used

- **Learning rate (α)** = 0.1
    - Controls how much new information overrides old Q values.
- **Discount factor (γ)** = 0.95
    - Ensures future rewards (reaching the goal) are valued highly.
- **Exploration strategy**: ε-greedy
    - Start ε = 1.0 (full exploration)
    - Decay rate = 0.995 per episode
    - Minimum ε = 0.05
- **Episodes** = 1,500
- **Max steps per episode** = 300

Why this setup?

- The high γ (0.95) balances long-term planning with immediate step penalties.
- The ε schedule forces the agent to explore thoroughly in early episodes, then gradually rely on exploitation.
- 1,500 episodes provided sufficient cycles for convergence without excessive runtime.

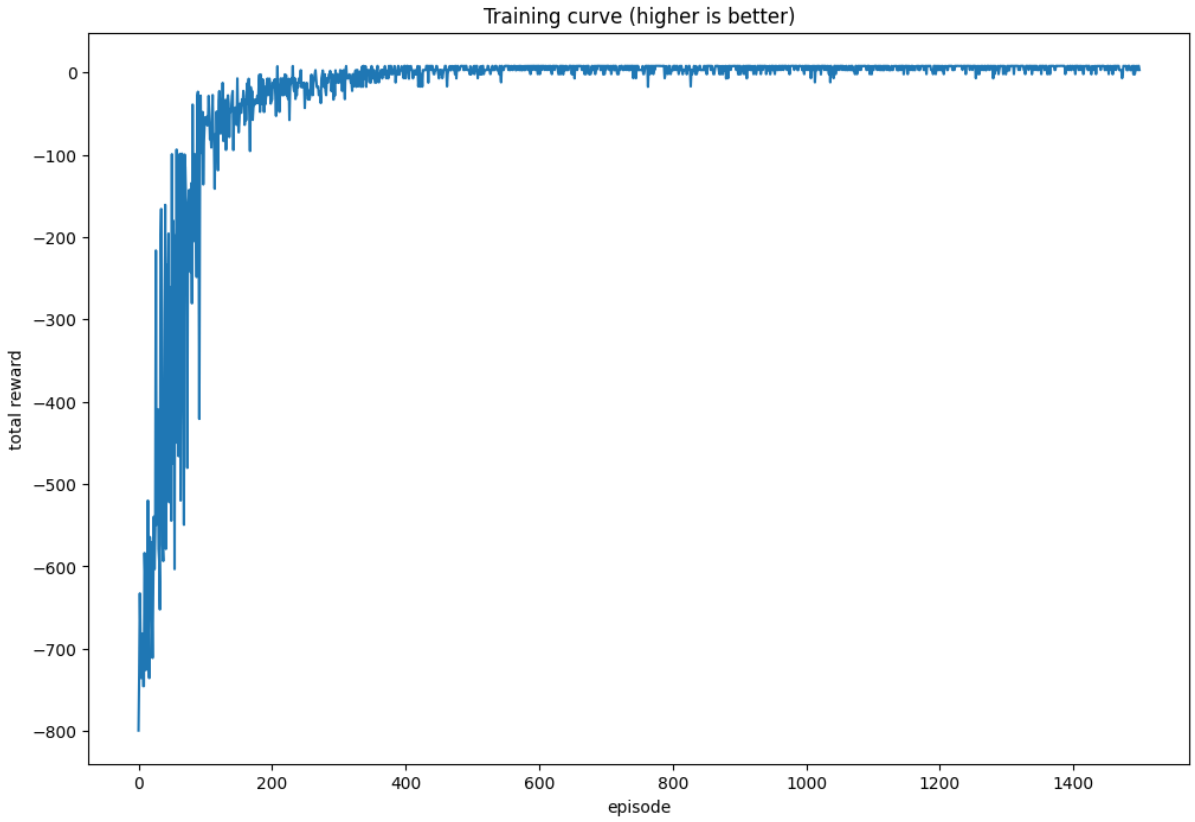
6.4 Training Behaviour and Learning Curve

The training loop printed key checkpoints every 300 episodes:

- **Episode 300/1500** → ε = 0.222, last_return = −12.5
    - The agent was still exploring heavily, often hitting walls and wandering. Negative total reward indicates inefficient paths.
- **Episode 600/1500** → ε = 0.050, last_return = +7.9

    - By now ε had reached its floor (0.05). The agent shifted to mostly exploitation. Achieving positive reward shows that it had started finding valid goal-reaching paths.

- **Episode 900/1500** → ε = 0.050, last_return = +7.9
    - Stable behaviour: the agent consistently reached the goal.
- **Episode 1200/1500** → ε = 0.050, last_return = +2.9
    - Small variations in reward occurred depending on whether the path taken was optimal or slightly longer.
- **Episode 1500/1500** → ε = 0.050, last_return = +2.9
    - Training completed. The agent converged on a reasonably efficient policy.



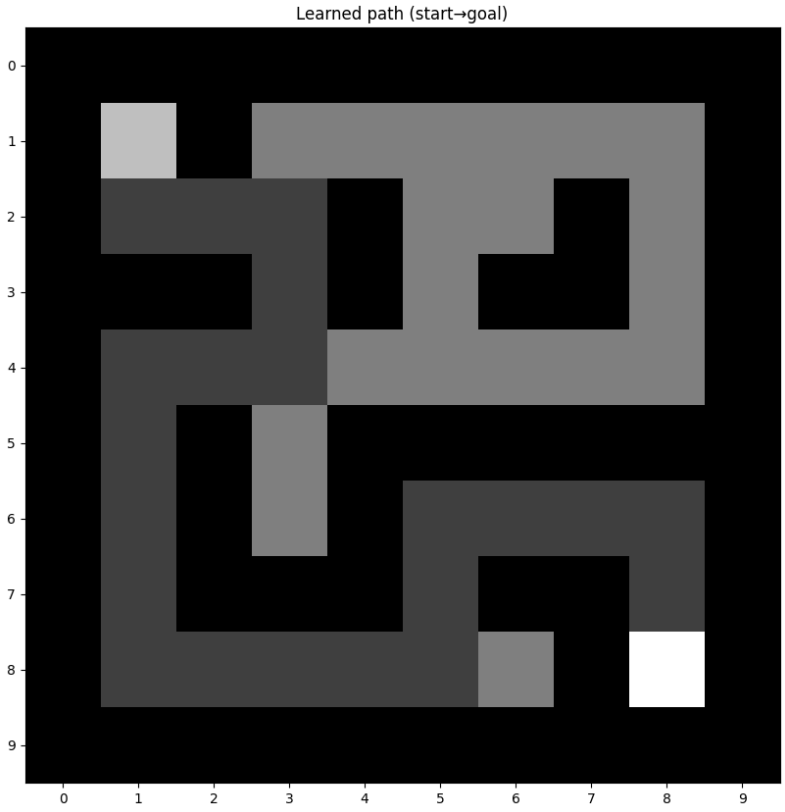
6.5 Learned Policy and Path

After training, the agent followed a **greedy policy** (choosing the highest Q-value at each state without randomness).

- **Reached goal?** → True
- **Steps taken** → 22 steps from (1, 1) to (8, 8)

The learned path avoided walls and unnecessary loops, moving steadily toward the target. The exact path coordinates printed included:

[(1,1),(2,1),(2,2),(2,3),(3,3),(4,3),(4,2),(4,1),(5,1),(6,1),(7,1),(8,1),(8,2),(8,3),(8,4),(8,5),(7,5),(6,5),(6,6),(6,7),(6,8),(7,8),(8,8)][(1,1), (2,1), (2,2), (2,3), (3,3), (4,3), (4,2), (4,1), (5,1), (6,1), (7,1), (8,1), (8,2), (8,3), (8,4), (8,5), (7,5), (6,5), (6,6), (6,7), (6,8), (7,8), (8,8)][(1,1),(2,1),(2,2),(2,3),(3,3),(4,3),(4,2),(4,1),(5,1),(6,1),(7,1),(8,1),(8,2),(8,3),(8,4),(8,5),(7,5),(6,5),(6,6),(6,7),(6,8),(7,8),(8,8)]



6.6 Key Learnings from the Experiment

1. **Reward design worked effectively**
    - The +10 goal reward vs. −0.1 step penalty naturally encouraged shorter paths.
    - The −5 wall penalty discouraged blind exploration into invalid cells.
2. **Exploration schedule mattered**
    - Starting with ε = 1.0 allowed the agent to try many routes.
    - By episode ~600, ε dropped to 0.05 and the agent began exploiting. This is exactly where consistent success began.
3. **Convergence was stable but not perfectly optimal**
    - Final path length of 22 steps is efficient, though not guaranteed shortest possible.
    - More episodes or different α tuning could push toward optimality.

4. **Potential improvements**
    - Increase episodes beyond 1,500 for more stable convergence.
    - Experiment with α between 0.05 and 0.2 to balance exploration updates.
    - Implement reward shaping (e.g., small reward for being closer to goal) to speed learning.

6.7 Conclusion

The reinforcement learning task successfully demonstrated Q-learning from scratch. Starting with no knowledge, the agent learned to consistently navigate from start to goal within 1,500 episodes, achieving a valid path of 22 steps. The experiment reinforced the importance of **reward design, exploration decay, and careful diagnostics** in reinforcement learning.

7. Overall Conclusions and Recommendations

This final section consolidates insights from all parts of the project including data preparation, exploratory analysis, supervised learning models, and the reinforcement learning experiment. It also highlights practical recommendations for business use and technical improvements.

7.1 Data Engineering and Preparation

- We integrated multiple datasets: items, sales, supermarkets, and promotions.
- Dates were standardized and used to derive temporal features such as day of week, month, is weekend, and hour bucket.
- Basket level aggregations were built including basket totals, sizes, and item diversity.
- The size column was cleaned into a categorical feature called size category. Missing values were imputed with the mode.
- Joins across datasets were carefully handled:
    - Sales were merged with item and supermarket metadata.

    - Promotion data could not be merged because the periods did not overlap.
- Outliers in sales amounts were detected and removed by capping at the ninety ninth point ninth percentile.
- The result was a consistent and well structured dataset suitable for both descriptive analysis and modeling.

7.2 Exploratory Data Analysis

- **Sales patterns**
    - Afternoon generated the highest revenue, followed by evening.
    - Weekends showed lower revenue than weekdays.
    - Sundays recorded the highest daily sales, while mid week days were weaker.
- **Product and brand insights**
    - Sixteen ounce, twenty six ounce, and twelve ounce packs were the most common sizes.
    - Private Label and Ragu were the leading brands by revenue.
    - Product Type Three and Type Two dominated total sales.
- **Supermarket level insights**
    - There was large variation in sales across stores.
    - The top performing supermarkets had several times more transactions than the bottom quartile.
- **Customer level insights**
    - Basket values were mostly small under ten dollars, with long tails.
    - The typical basket size was two to three items.
    - There was a strong positive correlation between basket size and basket value.
    - RFM segmentation showed clear differences:
        - Loyal customers had high frequency and monetary scores.
        - At risk customers had long recency days and low monetary values.

7.3 Machine Learning Models

Basket Revenue Forecast

- Random Forest and Gradient Boosting achieved R squared of zero point nine eight four, predicting basket totals with very high accuracy.
- The number of items in the basket and the average unit price explained almost all the variance.

Customer Return Within Thirty Days

- This problem was more challenging due to noisy labels and missing data during May.
- Logistic Regression achieved AUC equal to zero point six one six and PR AUC equal to zero point seven four one.
- Random Forest improved performance with AUC equal to zero point six seven zero and PR AUC equal to zero point seven nine four, with better recall for churners.
- The strongest predictors were recency days, rolling thirty day amount, and rolling thirty day frequency.
- Although not perfect, the model provides useful probabilities to rank customers for retention campaigns.

High Performing Supermarkets

- High performing supermarkets were defined as stores in the top revenue quartile.
- Random Forest achieved AUC equal to zero point nine nine seven and Accuracy equal to zero point nine six.
- The most important features were transactions at zero point four seven nine and customers at zero point two eight nine, followed by average basket value and average unit price.
- This model gives management a clear diagnostic tool to identify and monitor top performing stores.

7.4 Reinforcement Learning Task

- A Q learning experiment was run on a ten by ten maze.
- After one thousand five hundred episodes, the agent learned a twenty two step path to the goal, balancing exploration and exploitation.
- The experiment validated the ability to design and train reinforcement learning agents with epsilon decay exploration, learning rate equal to zero point one, and discount factor equal to zero point nine five.
- While not directly tied to supermarket data, it demonstrated knowledge of dynamic programming and reward driven optimization.

7.5 Recommendations

Basket Forecasting and Revenue Management

- **Integrate predictive models at checkout**: Connect the basket revenue forecast model into the point of sale system. As a customer scans items, the system can estimate the expected basket total in real time. If the final bill diverges significantly, it can flag issues such as incorrect pricing, misapplied discounts, or system errors.
- **Dynamic staff planning**: Since forecasted basket sizes and values are highly accurate, store managers can use the model to plan staffing levels at peak times. Afternoon hours, which were found to be the busiest, should consistently have more cashier and self checkout capacity.
- **Revenue prediction dashboards**: Roll up daily and weekly predictions to store managers. These forecasts allow proactive decisions on inventory replenishment and short term promotions to push higher margin items into baskets.

Customer Retention and Loyalty Strategy

- **At risk customer identification**: Use the thirty day return model to generate a daily list of customers with high churn probability. These customers can be targeted with win back campaigns such as digital coupons, personalised SMS offers, or loyalty app notifications.
- **Tiered loyalty actions**:
    - Loyal customers (low recency, high frequency, high monetary value) should receive exclusives and early access to promotions to maintain their relationship.
    - At risk customers (long recency, falling spend) should be given aggressive discounts or bundled offers to pull them back.
    - Recent customers should be nurtured with gentle reminders or introductory offers so they grow into loyal customers.
- **Measure promotion impact**: Track whether flagged customers actually return after receiving interventions. Feed this back into the model pipeline so the algorithm learns which offers drive retention.

Store Benchmarking and Operational Excellence

- **Store performance monitoring**: Deploy the high performing supermarket classifier as part of a monthly management dashboard. Managers can see where their store sits against the benchmark and which factors (transactions, customers, basket value) explain the gap.
- **Replication of success**: Identify stores that consistently rank as high performers. Study their operational practices (staff scheduling, product assortment, layout) and replicate them in lower tier stores.

- **Focused training**: For stores with low average basket value or few unique customers, regional managers can design training sessions for local staff to improve upselling techniques and customer engagement.
- **Incentive alignment**: Tie store manager bonuses partly to model derived improvement indicators such as growth in basket value or frequency, not just absolute revenue.

Technical Improvements and Long Term Strategy

- **Data pipeline reliability**: The missing month of May created significant issues for customer models. A monitoring system should be put in place to alert data engineers when sales feeds drop below expected volumes.
- **Model upgrades**:
    - For customer return, test gradient boosting implementations such as XGBoost or LightGBM that handle class imbalance better.
    - For store benchmarking, expand features to include demographic and geographic context if available.
- **Automation of reporting**:
    - Customer churn probabilities should be refreshed daily and fed into the loyalty management system.
    - Store benchmarking dashboards should be refreshed weekly with clear traffic light indicators of performance.
- **Closed loop learning**: Customer retention actions and promotion campaigns should be logged and re merged into the training dataset. This allows the models to evolve toward interventions that have the highest real business impact.

7.6 Closing Statement

This project demonstrated the full data engineering and machine learning lifecycle.

- Ingesting and cleaning multiple datasets.
- Creating robust engineered features.
- Performing deep exploratory analysis to understand the business context.
- Building and comparing multiple predictive models.
- Applying reinforcement learning to showcase optimization skills.

The models achieved strong performance in basket forecasting and supermarket classification, and provided a foundation for actionable customer retention insights.