Dominos Predictive Purchase System

1. Problem Statement

1.1 Business Objective

Dominos aims to optimize its ingredient ordering process by accurately forecasting future sales and creating efficient purchase orders. By leveraging predictive analytics, the business can maintain optimal inventory levels, reduce waste, and improve supply chain operations.

1.2 Business Use Cases

- **Inventory Management:** Ensure stock levels are sufficient to meet demand while avoiding overstocking.
- **Cost Reduction:** Minimize costs associated with expired or excess inventory.
- **Sales Forecasting:** Predict sales trends to inform business strategies and promotions.
- **Supply Chain Optimization:** Streamline ordering processes to prevent disruptions and align with sales trends.

---

2. Approach

2.1 Data Preprocessing and Exploration:

1. **Data Cleaning:**
    - Removed missing or inconsistent data entries.
    - Handled outliers and formatted the data appropriately.
2. **Exploratory Data Analysis (EDA):**
    - Analyzed sales trends, seasonality, and patterns in historical sales data.
    - Visualized data to identify significant features and relationships.

2.2 Sales Prediction:

1. **Feature Engineering:**
    - Created relevant features such as day of the week, month, promotional periods, and holiday effects.
2. **Model Selection:**
    - Evaluated various time-series forecasting models (ARIMA, SARIMA, Prophet, LSTM, Regression).

3. **Model Training:**
    - Trained predictive models using historical sales data.
4. **Model Evaluation:**
    - Used Mean Absolute Percentage Error (MAPE) to evaluate model performance.

2.3 Purchase Order Generation:

1. **Sales Forecasting:**
    - Predicted pizza sales for the next week using the trained model.
2. **Ingredient Calculation:**
    - Calculated required ingredient quantities based on predicted sales and the ingredient dataset.
3. **Purchase Order Creation:**
    - Generated a detailed purchase order listing ingredient quantities needed for the forecasted period.

---

3. Data Preprocessing

3.1 Handling Missing Values in Sales Dataset

3.1.1 `pizza_name_id`

1. Identified rows with missing `pizza_name_id`.
    - Printed rows with null values to understand the context.
    - Displayed details of rows (e.g., `pizza_id`, `pizza_size`, `pizza_name`) to assist in filling missing data.
2. Checked for matching combinations of `pizza_name` and `pizza_size` with existing `pizza_name_id` values.
3. Filled null values by mapping the appropriate `pizza_name_id` to the respective rows.
    - Unmatched rows were logged for further review.

3.1.2 `total_price`

1. Identified rows with missing `total_price` values.
2. Filled missing values by calculating `quantity * unit_price`.

3.1.3 `pizza_category`

1. Identified rows with missing `pizza_category` values.
2. Created a mapping of `pizza_name` to `pizza_category` using non-null data.
3. Filled null values by applying the mapping and reviewed updates for accuracy.

3.1.4 `pizza_ingredients`

1. Created a mapping of `pizza_name` to `pizza_ingredients` based on existing data.
2. Filled null values using this mapping.

3.1.5 `pizza_name`

1. Identified rows with missing `pizza_name`.
2. Used `pizza_name_id` to find the most common `pizza_name` for the corresponding ID.
3. Filled null values with the most frequent `pizza_name` and validated changes.

3.1.6 Results:

- All missing values in the sales dataset were successfully addressed.
- Verified with a final check to ensure zero null values remained.

3.2 Handling Missing Values in Ingredients Dataset

1. Filled missing `Items_Qty_In_Grams` values using the mean quantity grouped by `pizza_name_id`.
- Verified that no missing values remained.

3.3 Duplicate Analysis

3.3.1 Sales Dataset

1. Checked for duplicate rows.
- Found no duplicates:
  Duplicated rows in Sales_dataset:
- Empty DataFrame
- Columns: [pizza_id, order_id, pizza_name_id, quantity, order_date, order_time, unit_price, total_price, pizza_size, pizza_category, pizza_ingredients, pizza_name]
- Index: []
2. Number of duplicated rows: 0

3.3.2 Ingredients Dataset

1. Checked for duplicate rows.
- Found no duplicates:
  Duplicated rows in Ingredients_dataset:
- Empty DataFrame
- Columns: [pizza_name_id, pizza_name, pizza_ingredients, Items_Qty_In_Grams]
- Index: []

2. Number of duplicated rows: 0

3.4 Merging and Feature Engineering

3.4.1 Merging Sales and Ingredients DataFrames

- The `Sales_dataset` and `Ingredients_dataset` were merged on the `pizza_name_id` column using a left join.
- Duplicate columns such as `pizza_ingredients_y` and `pizza_name_y` were removed, and key columns were renamed for clarity:
  - `pizza_name_x` → `pizza_name`
  - `pizza_ingredients_x` → `pizza_ingredients`
- Duplicate rows were identified and removed, resulting in 187,371 unique entries.
- Summary statistics and data types were reviewed to ensure consistency:
  - Key metrics like `quantity`, `unit_price`, `total_price`, and `Items_Qty_In_Grams` were validated.

3.4.2 Feature Engineering

- **Date-Time Features:**
  - Converted `order_date` into a datetime format to ensure compatibility for time-based analysis.
  - Extracted new temporal features:
    - `day_of_week`: Numerical representation of the day (e.g., Monday = 0).
    - `month`: Month number and corresponding name.
    - `week`: ISO week number of the year.
    - `year`: Extracted year from `order_date`.
- **Day Type (Weekday vs Weekend):**
  - Identified weekends (Saturday and Sunday) using `day_of_week`.
  - Created a `day_type` column categorizing entries as "Weekday" or "Weekend".
- **Time-Based Features:**
  - Extracted `hour` from `order_time` and categorized it into:
    - `working_hours`: "Working Hours" (9 AM to 6 PM) vs "Non-Working Hours".
    - `time_period`: Segmented into "Morning", "Afternoon", "Evening", "Night", and "Midnight" based on the hour.
- **Mapping Month Names:**
  - Replaced numeric month values with corresponding names (e.g., 1 → "January").

3.4.3 Results:

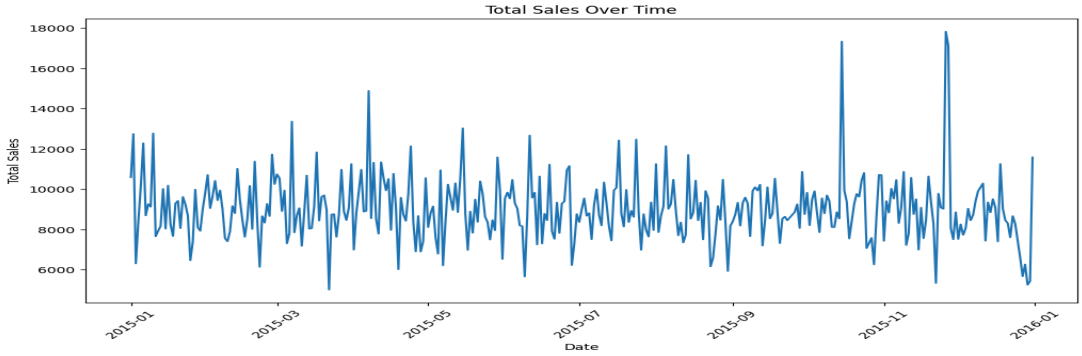
- Successfully enriched the dataset with detailed temporal and categorical features, enabling advanced analysis and model development.
- Ensured all data types and feature names were consistent and descriptive for seamless integration into downstream processes.

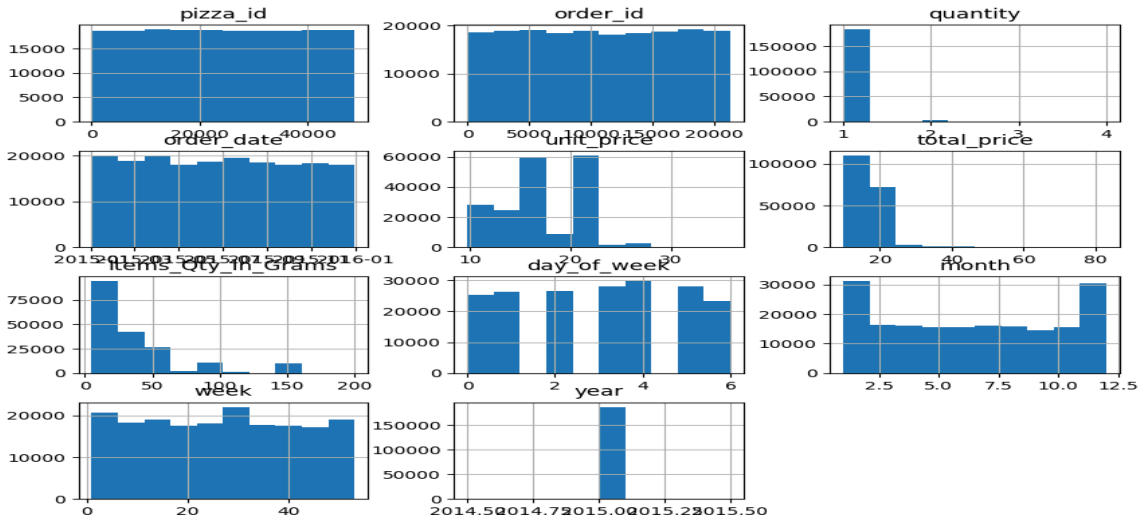
---

4. Exploratory Data Analysis (EDA)

4.1 Visualizing Distributions

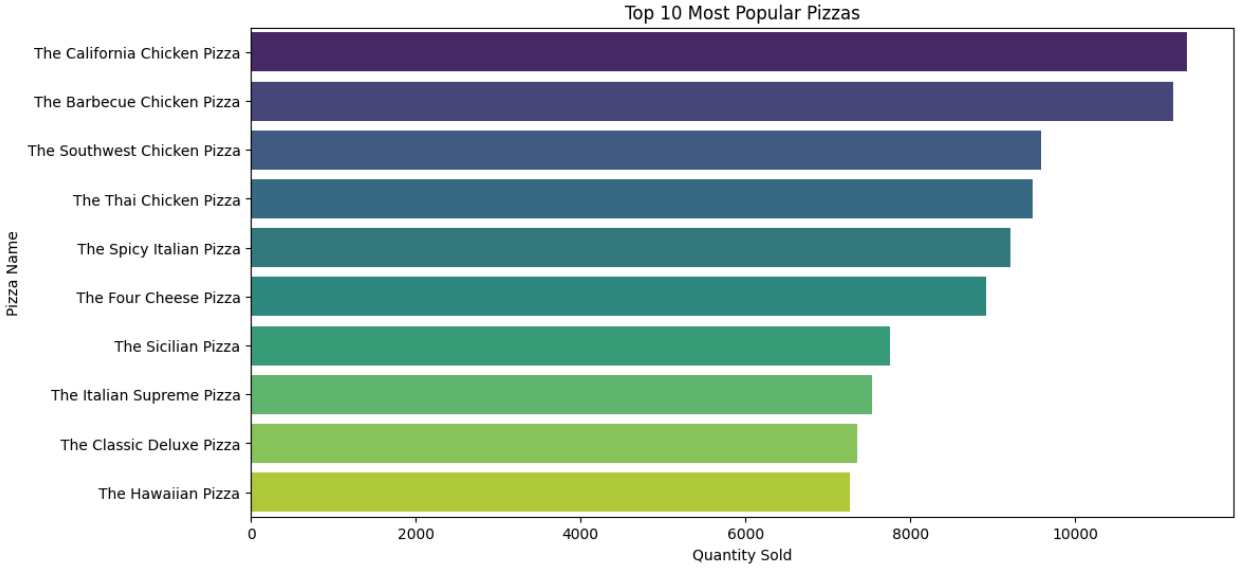
- **Analyzing Sales Trends Over Time:**
    - Aggregated daily sales data to observe overall sales patterns.
    - Generated a time-series line plot of `total_price` over `order_date` to identify trends and seasonality.
    - Histogram plots were created to understand the distribution of numerical variables like `quantity`, `unit_price`, and `total_price`.





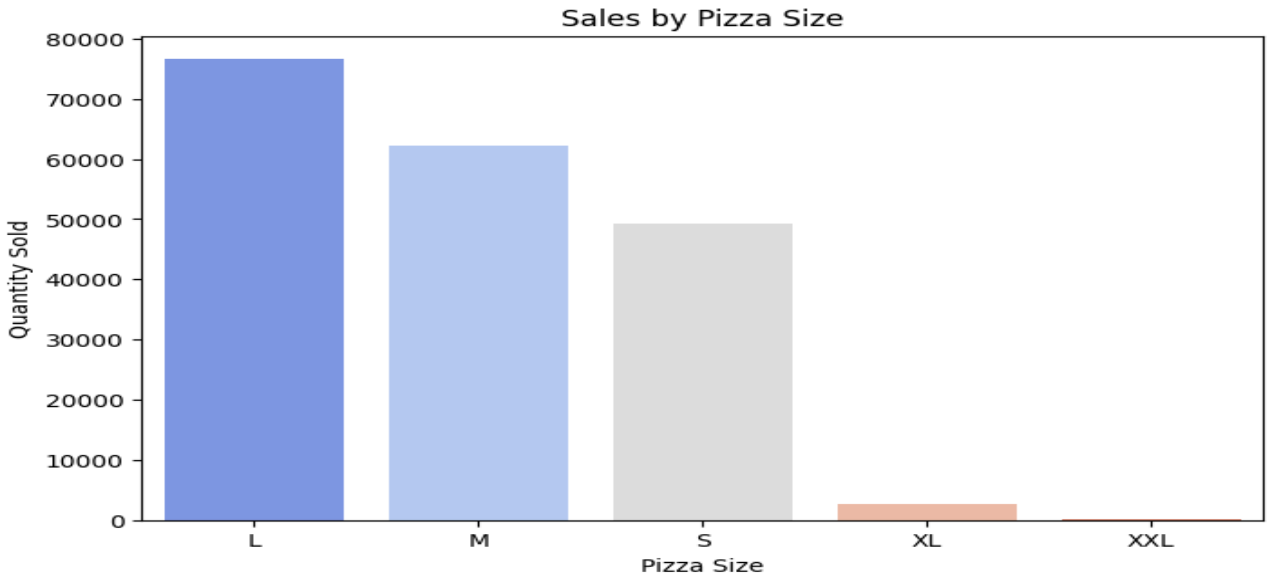
4.2 Top-Selling Pizzas

- Aggregated `quantity` sold by `pizza_name` to identify the most popular pizzas.
- Sorted the results and plotted the top 10 pizzas using a horizontal bar chart.
- Observed that specific pizza types dominate overall sales.

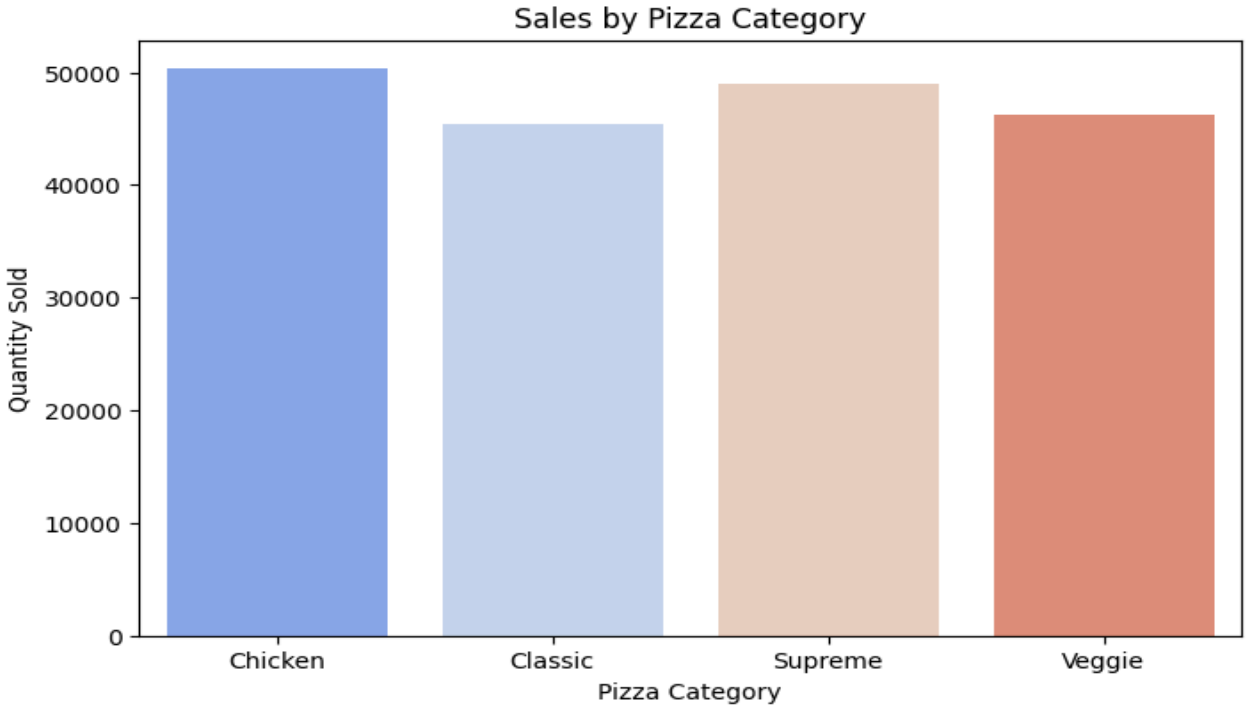


4.3 Sales by Pizza Size and Category

- **Pizza Size:**
  - Grouped data by `pizza_size` and calculated total `quantity` sold.
  - Created a bar chart to visualize sales across different pizza sizes.

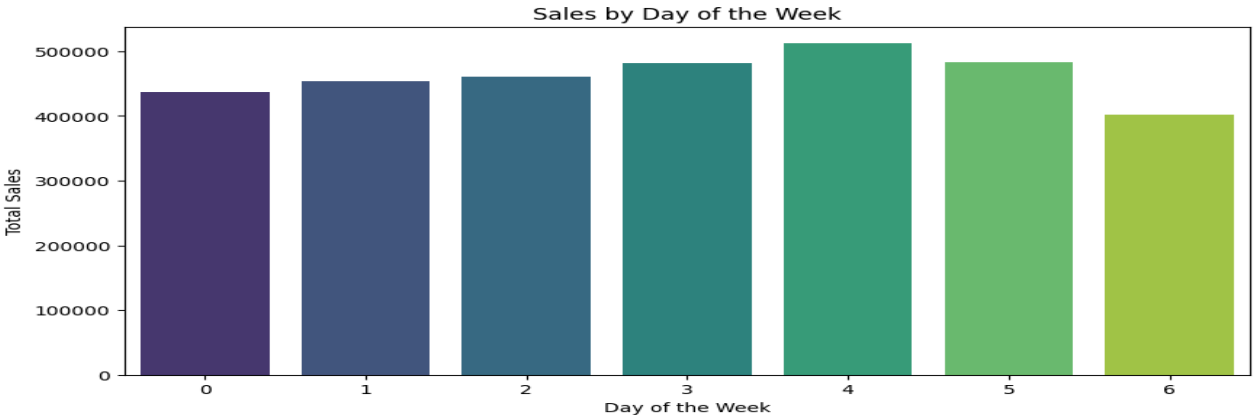


- **Pizza Category:**
    - Aggregated `quantity` sold by `pizza_category`.
    - Bar charts revealed differences in sales between categories.

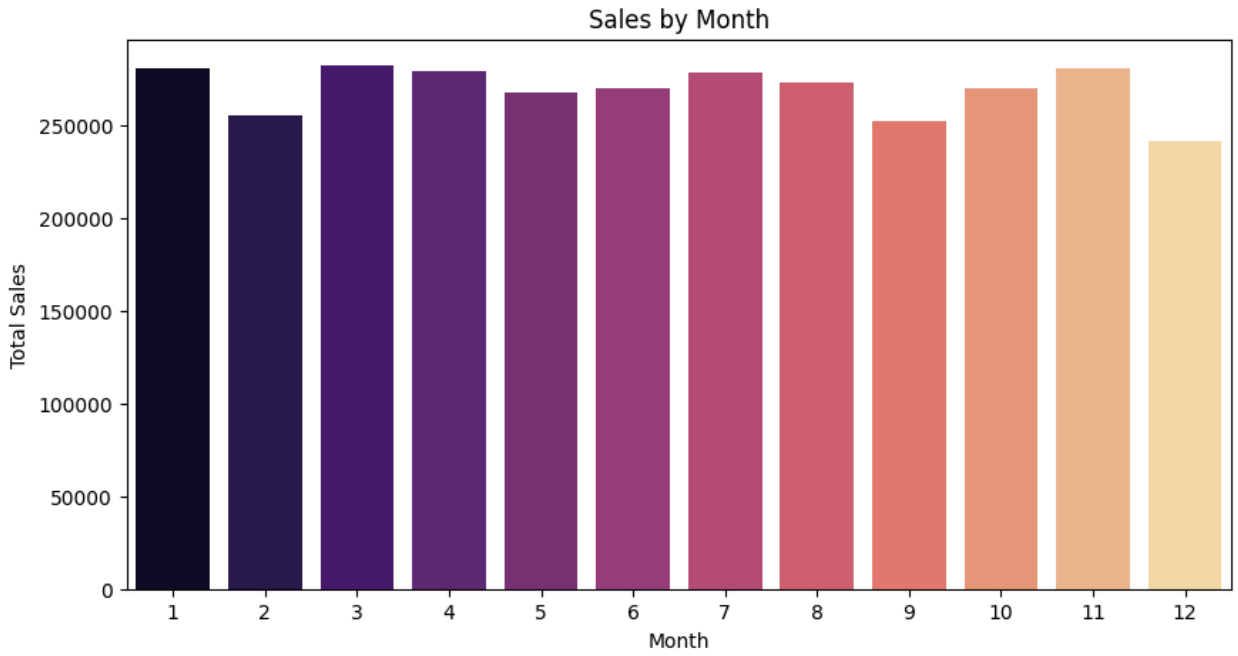


4.4 Seasonal Sales Trends and Patterns

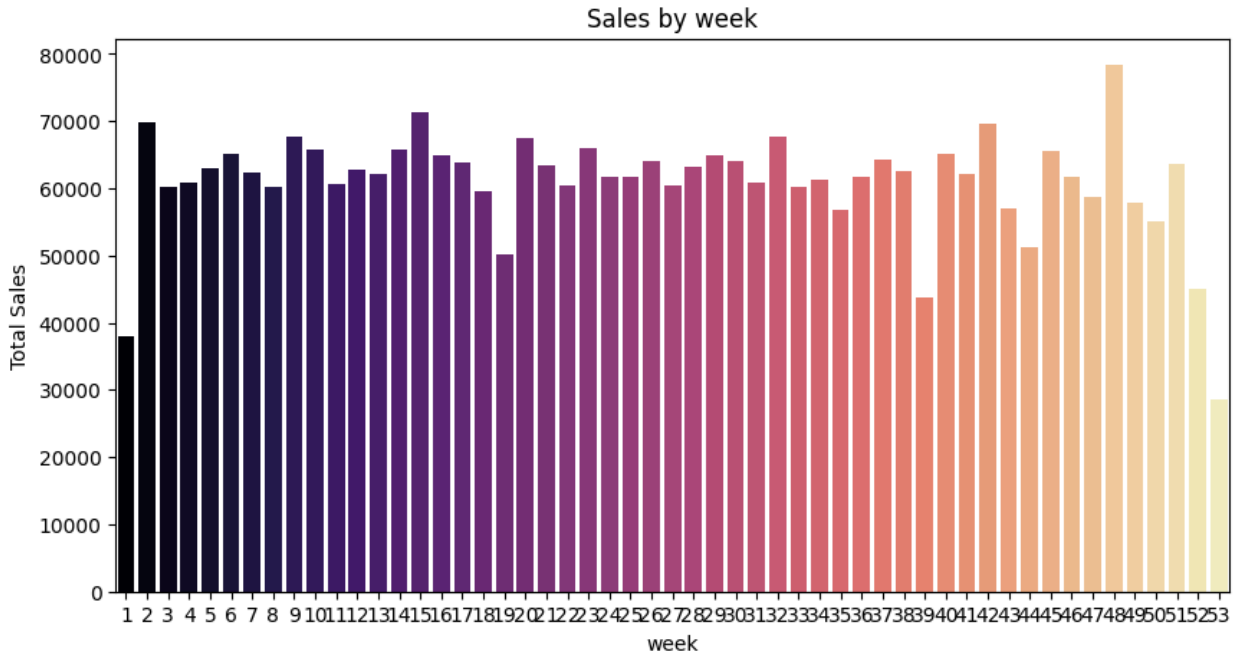
- **Day of the Week Analysis:**
    - Aggregated total sales (`total_price`) by `day_of_week`.
    - Visualized weekly patterns using bar plots, indicating higher sales on weekends.



- **Monthly Analysis:**
  - Grouped data by `month` to analyze seasonal sales patterns.
  - Bar plots revealed peak sales during specific months, indicating potential holiday effects.

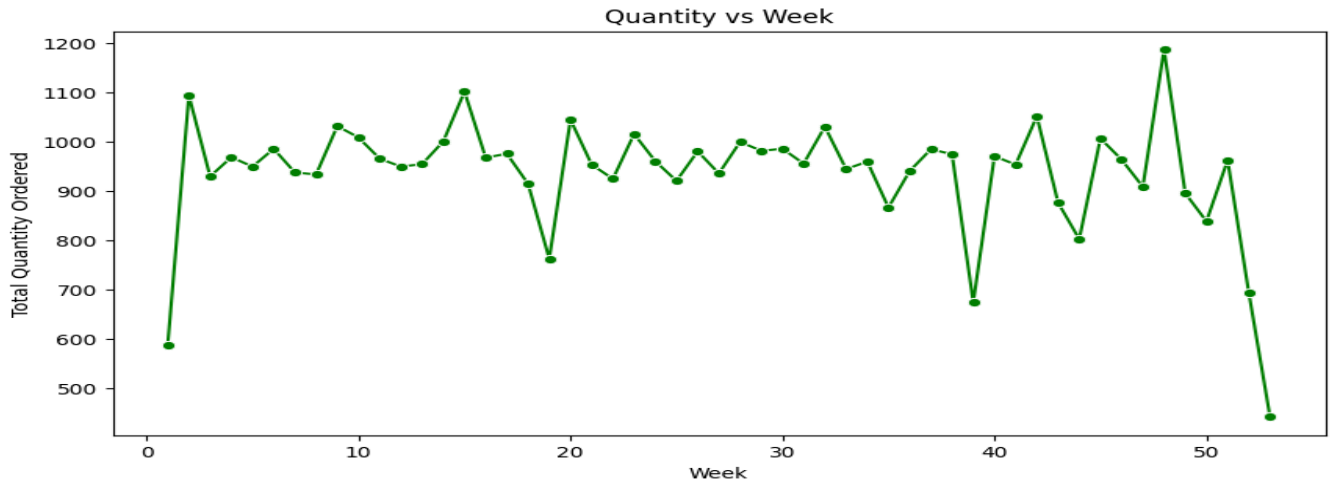


- **Weekly Trends:**
  - Examined weekly sales data to analyze trends over the year.

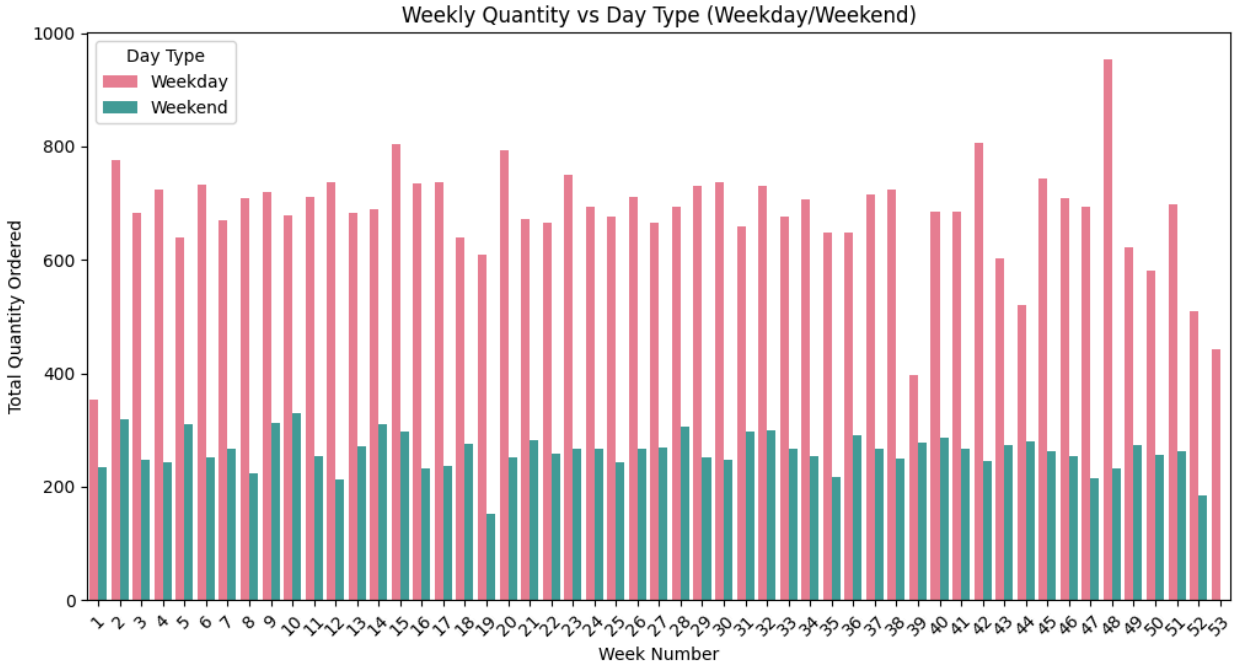


4.5 Quantity vs Time-Based Metrics

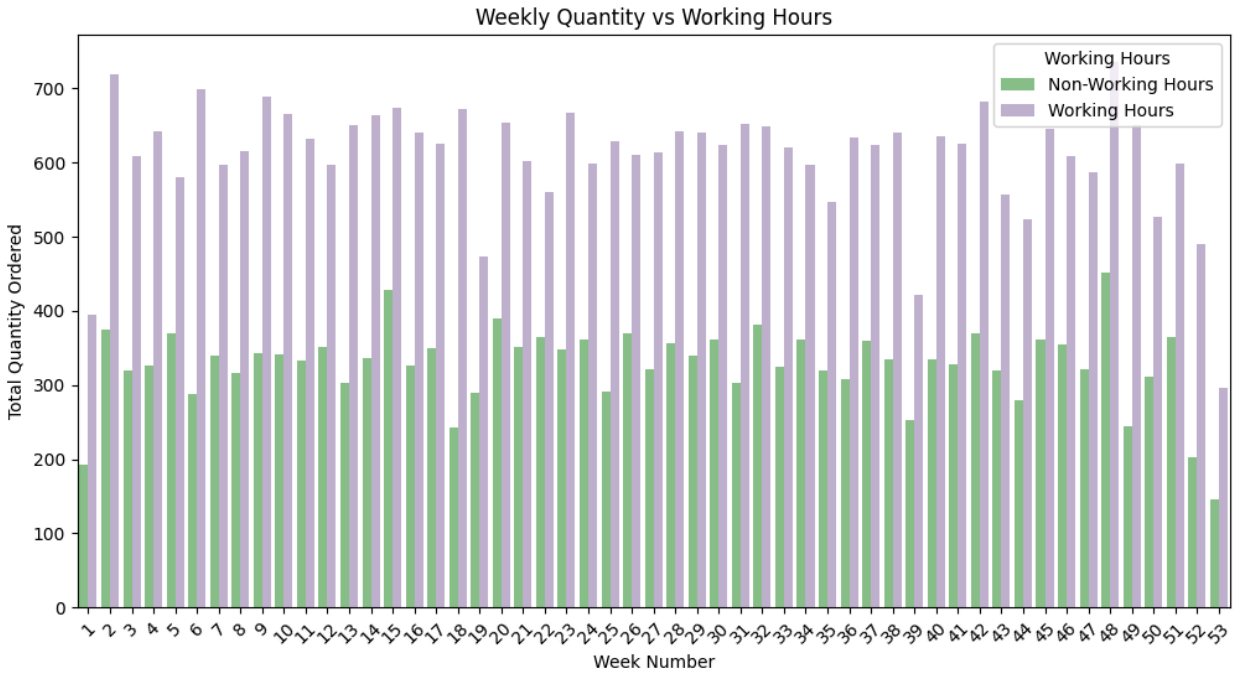
- **Quantity vs Week:**
    - Line plots highlighted changes in `quantity` ordered per week.



- **Quantity by Day Type (Weekday vs Weekend):**
    - Compared sales volumes on weekdays versus weekends using grouped bar plots.

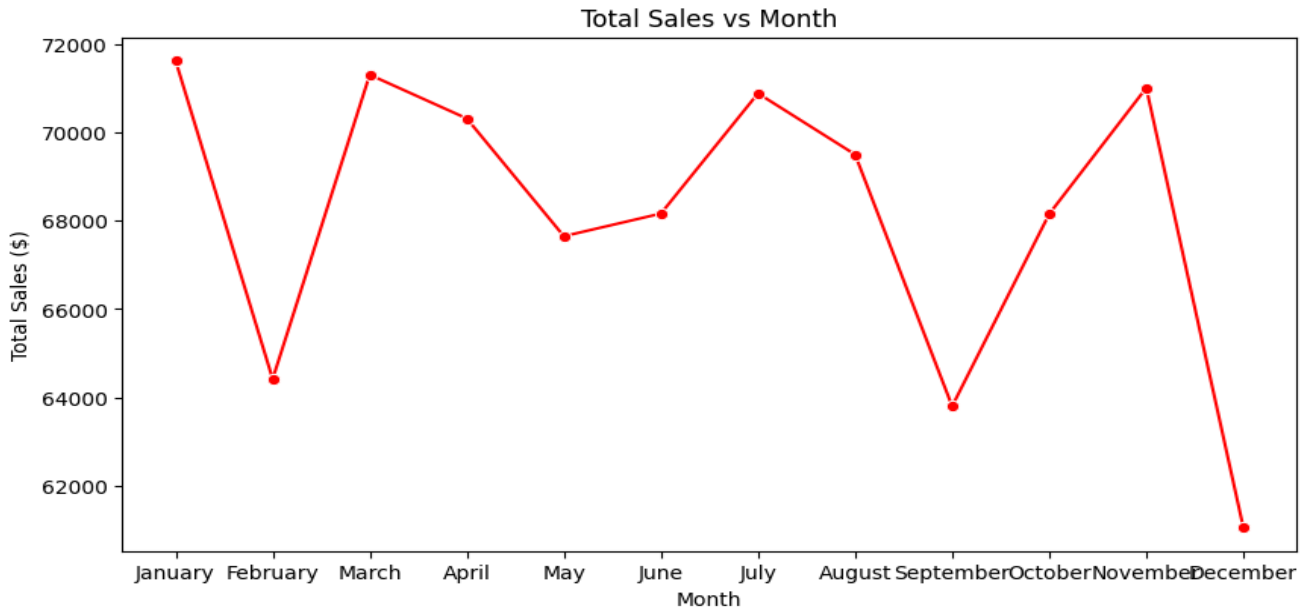


- **Quantity by Working Hours:**
    - Investigated how `quantity` sold varies across different working hours.

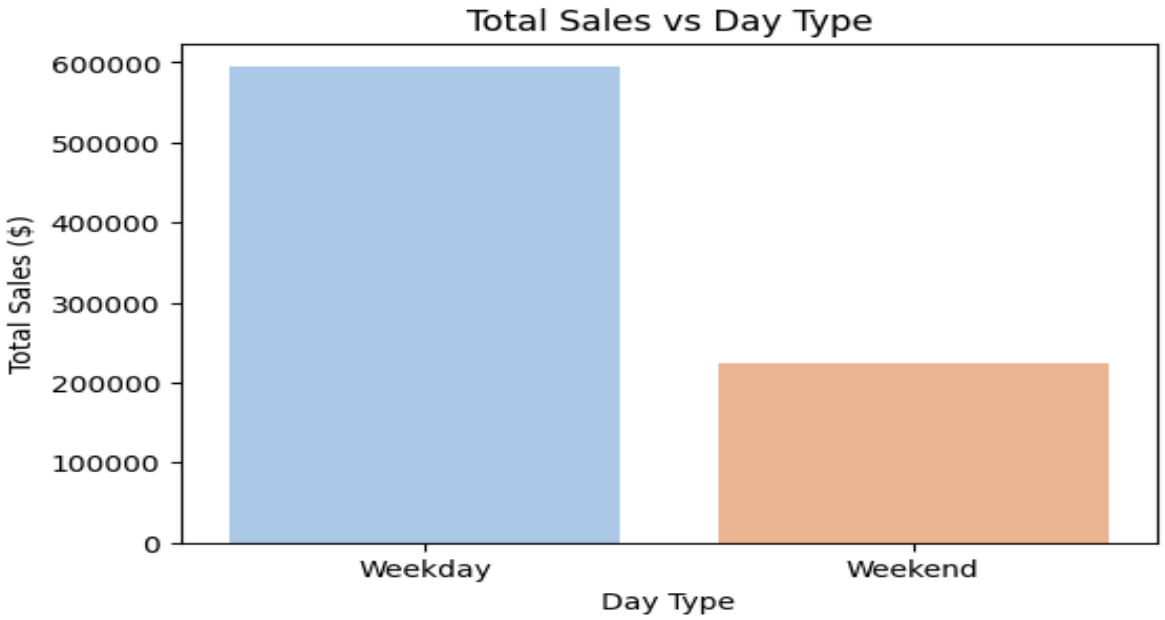


4.6 Total Price vs Time-Based Metrics

- **Total Price by Month:**
    - Analyzed monthly sales (`total_price`) to understand revenue trends using line plots.

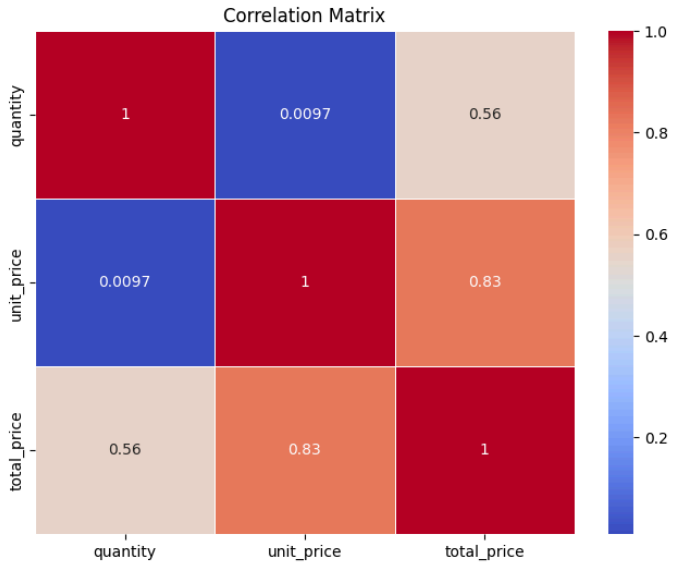


- **Total Price by Day Type:**
    - Bar charts compared revenue between weekdays and weekends, showing significant differences.



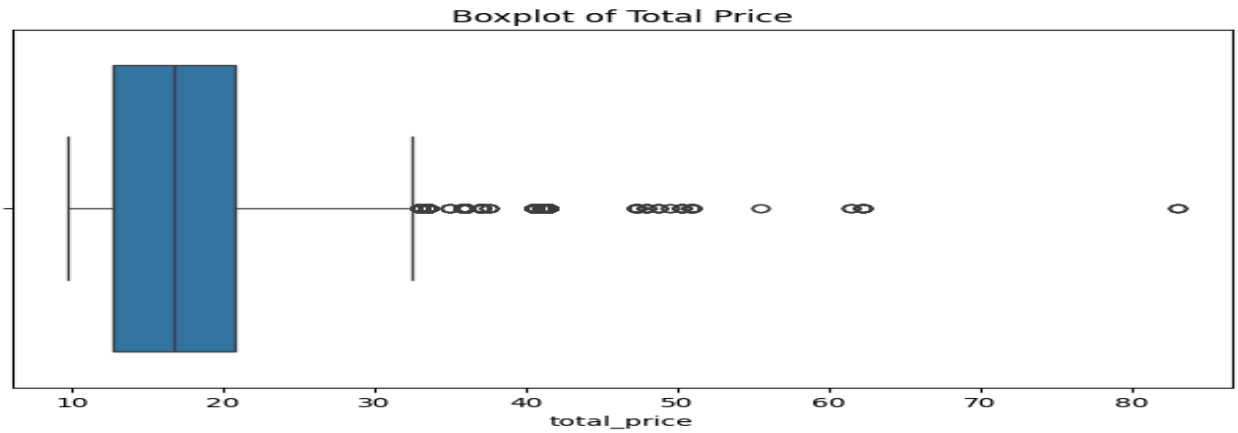
4.7 Correlation Analysis of Sales Metrics

- Computed the correlation matrix for numerical variables (`quantity`, `unit_price`, `total_price`).
- Visualized correlations using a heatmap, highlighting positive relationships between `quantity` and `total_price`.



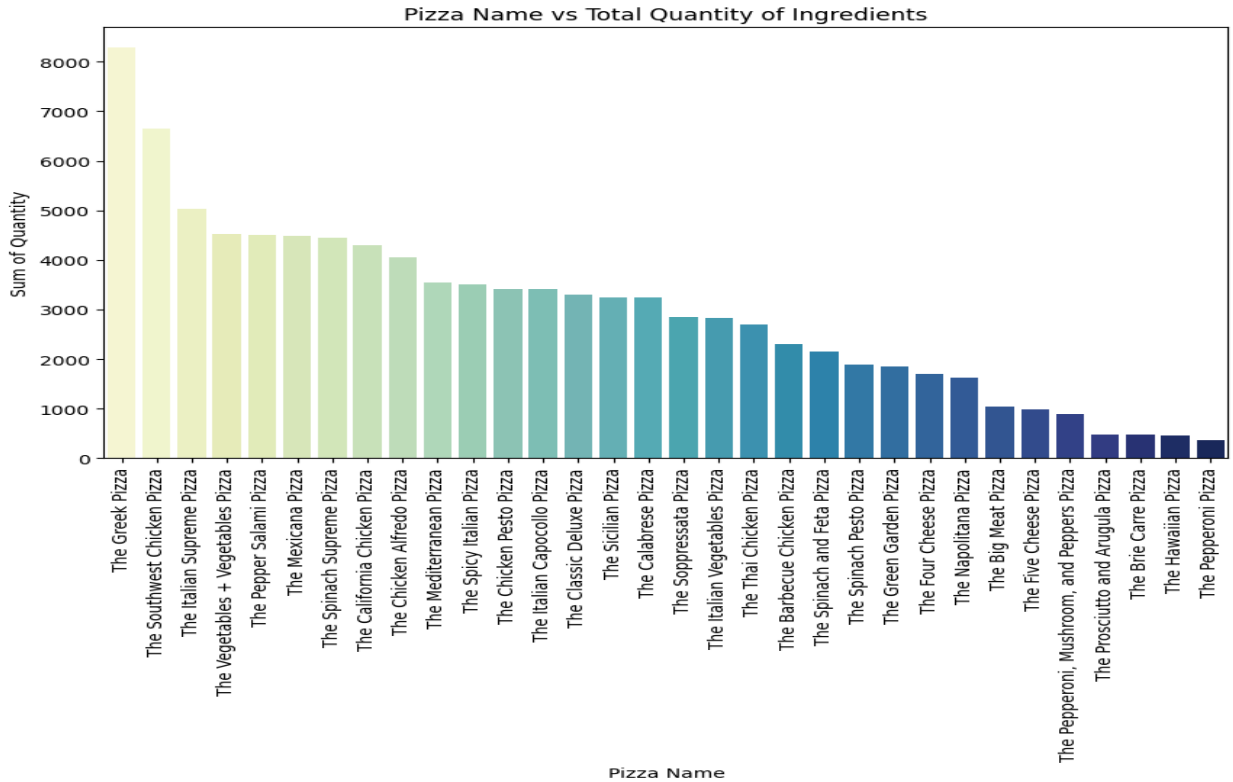
4.8 Outlier Detection

- Used boxplots to detect outliers in `total_price` and other metrics.
- Identified potential anomalies requiring further investigation.

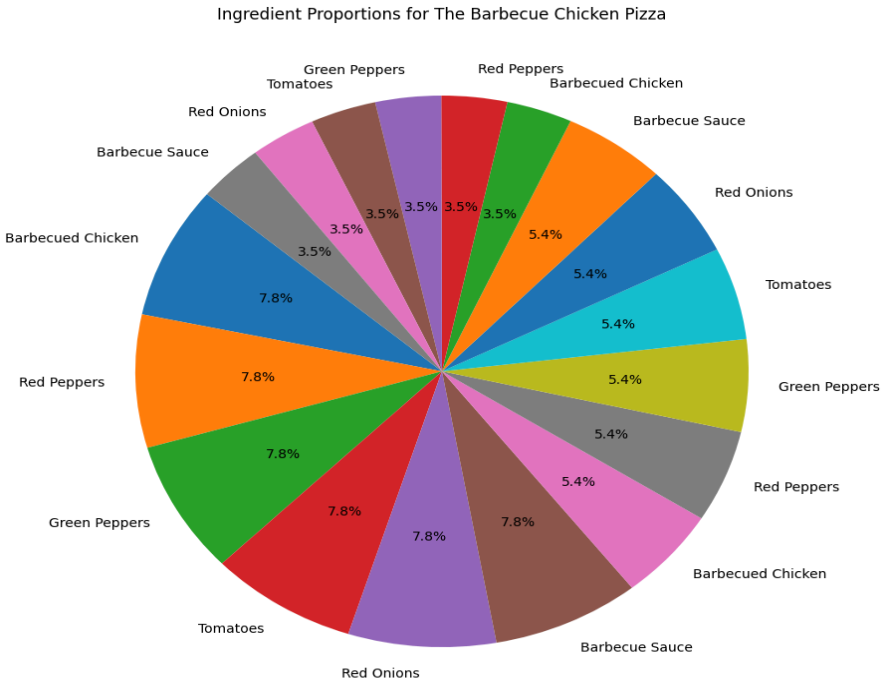


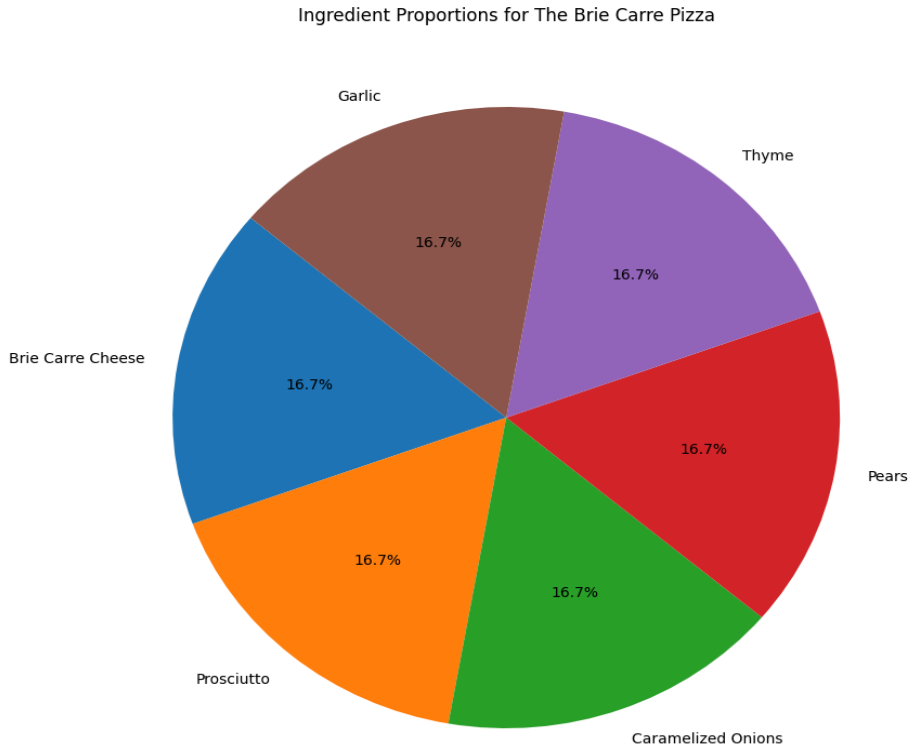
4.9 Ingredient Analysis

- **Pizza Name vs Total Ingredients Quantity:**
  - Aggregated `Items_Qty_In_Grams` for each `pizza_name`.
  - Bar charts displayed the total quantity of ingredients required for each pizza type.



- **Ingredient Proportions per Pizza:**
    - Created pie charts to show the proportional distribution of ingredients for individual pizza types, revealing key insights into ingredient utilization.





4.10 Results:

- Visualizations and metrics provided insights into:
    - Best-selling pizzas and their sizes/categories.
    - Seasonal and weekly patterns in sales.
    - Correlation between sales variables and revenue trends.
    - Ingredient usage and proportional analysis for inventory optimization.

---

5. Predictive Modeling

5.1 Models Implemented

5.1.1. ARIMA Model

- **Best Parameters**: (3, 1, 5)
- **MAPE Score**: 0.1979
- **Implementation Details**:
    - Used iterative parameter tuning across p_values(0-6), d_values(0-2), q_values(0-6)
    - Train-test split: 80-20
    - Forecast horizon: 11 weeks

5.1.2. SARIMA Model

- **Best Parameters**: (1, 0, 1)x(2, 0, 0, 7)
- **MAPE Score**: 0.1507
- **Implementation Details**:
    - Incorporated weekly seasonality (s=7)
    - Parameter ranges:
        - p, q: 0-3
        - d: 0-2
        - P, Q: 0-3
        - D: 0-2

5.1.3. Prophet Model

- **MAPE Score**: 0.2155
- **Implementation Details**:
    - Weekly frequency forecasting
    - No additional regressors or holidays included
    -

5.1.4. Linear Regression

- **MAPE Score**: 0.1906
- **Feature Engineering**:
    - week_of_year
    - day_of_week
    - month
    - year
- **Implementation**: Simple linear regression with temporal features

5.1.5. XGBoost Regression

- **MAPE Score**: 0.3110
- **Model Parameters**:
    - n_estimators: 100
    - learning_rate: 0.1
    - max_depth: 5
- **Features**: Same as Linear Regression

5.1.6. LSTM Neural Network

- **MAPE Score**: 0.2379
- **Architecture**:
    - Input shape: (time_steps=3, features=1)
    - LSTM layer: 50 units with ReLU activation
    - Dense output layer: 1 unit
- **Training Parameters**:
    - Optimizer: Adam
    - Loss: MSE
    - Epochs: 50
    - Batch size: 32

---

6. Forecasting Results

6.1 Time Series Model Selection and Validation

6.1.1 Model Evaluation

- **Selected Model**: SARIMA(1,0,1)x(2,0,0,7)
- **Performance Metric**: MAPE (Mean Absolute Percentage Error) of 0.1507
- **Model Components**:

    - Non-seasonal: AR(1), no differencing, MA(1)
    - Seasonal: SAR(2), no differencing, period=7

6.1.2 Model Parameters and Statistics

- **AIC**: 570.977
- **BIC**: 579.165
- **Log Likelihood**: -280.489
- **Sample Period**: 12-29-2014 to 12-28-2015
- **Number of Observations**: 53 weeks

6.2 Next Week's Sales Forecast (2016-01-04 to 2016-01-10)

6.2.1 Overall Pizza Sales

- **Total Forecasted Units**: 3,566 pizzas
- **Forecast Period**: Week of January 4th, 2016

6.2.2 Top Performing Pizza Types

1. **Spicy Italian Large**: 110 pizzas
2. **Thai Chicken Large**: 109 pizzas
3. **Southwest Chicken Large**: 99 pizzas
4. **Pepperoni & Salami Large**: 69 pizzas
5. **Southwest Chicken Medium**: 57 pizzas

6.2.3 Pizza Size Distribution

- **Large Size**: Generally higher sales predictions
- **Medium Size**: Moderate sales volumes
- **Small Size**: Lower but consistent sales predictions

6.3 Ingredient Requirements Forecast

6.3.1 High-Volume Ingredients

1. **Chicken**: 80,950 grams
2. **Red Onions**: 65,860 grams
3. **Capocollo**: 59,250 grams
4. **Tomatoes**: 47,510 grams
5. **Bacon**: 27,510 grams

6.3.2 Medium-Volume Ingredients

1. **Mushrooms**: 24,680 grams
2. **Spinach**: 24,285 grams
3. **Pepperoni**: 23,790 grams
4. **Garlic**: 23,650 grams
5. **Corn**: 23,000 grams

6.4 Forecasting Methodology

6.4.1 Data Preparation

1. **Time Series Conversion**
    - Converted order dates to datetime format
    - Aggregated sales data into weekly periods
    - Created separate time series for each pizza type
2. **Model Training Process**
    - Used 80% of data for training
    - Applied parameter tuning through grid search
    - Validated results on remaining 20% test data

6.4.2 Forecasting Approach

1. **Overall Sales Forecast**
    - Applied SARIMA model to total weekly sales
    - Generated one-week ahead predictions
2. **Pizza-Type Specific Forecasts**
    - Individual SARIMA models for each pizza type
    - Implemented fallback mechanisms using 4-week moving averages
    - Rounded predictions to practical whole numbers
3. **Ingredient Calculation**
    - Mapped pizza forecasts to ingredient requirements
    - Calculated total ingredient quantities needed
    - Generated detailed ingredient-wise requirements

6.5 Key Findings and Recommendations

6.5.1 Sales Patterns

- Higher sales volumes predicted for large-size pizzas
- Specialty pizzas (Spicy Italian, Thai Chicken) show strong demand
- Consistent baseline demand for traditional varieties

6.5.2 Inventory Planning

- **Primary Focus**: Ensure adequate stock of high-volume ingredients
- **Special Attention**: Perishable ingredients like chicken and vegetables
- **Safety Stock**: Recommended for critical ingredients based on forecast

6.5.3 Operational Implications

- Kitchen staffing needs based on predicted volumes
- Preparation requirements for high-demand pizza types
- Storage capacity planning for predicted ingredient volumes

6.5.4 Model Performance

- Strong statistical fit (as indicated by AIC, BIC values)
- Reliable short-term forecasting capability
- Effective handling of weekly seasonality patterns

---

7. Evaluation and Recommendations

7.1 Performance Comparison

Model	MAPE	Rank	Performance Category
SARIMA	0.1507	1	Best Performer
ARIMA	0.1704	2	Strong Performer
Linear Regression	0.1906	3	Good Performer
Prophet	0.2155	4	Moderate Performer
LSTM	0.2379	5	Moderate Performer
XGBoost	0.3110	6	Needs Improvement

7.2 Key Findings

1. **Best Model**: SARIMA demonstrated superior performance with the lowest MAPE of 0.1507, likely due to its ability to capture both seasonal and trend components.
2. **Traditional vs Modern Approaches**:

    - Traditional statistical models (SARIMA, ARIMA) outperformed modern machine learning approaches
    - Simple Linear Regression performed surprisingly well compared to more complex models
3. **Model Complexity vs Performance**:
    - More complex models (LSTM, XGBoost) didn't necessarily yield better results
    - The simple Linear Regression model outperformed both LSTM and XGBoost
4. **Prediction Patterns**:
    - Most models captured the general trend
    - SARIMA showed better handling of seasonal patterns
    - Prophet demonstrated stable but less dynamic predictions

7.3 Recommendations

1. **Primary Model Selection**: Deploy SARIMA as the primary forecasting model due to its superior performance and ability to capture seasonal patterns.
2. **Ensemble Approach**: Consider combining predictions from SARIMA, ARIMA, and Linear Regression for potentially more robust forecasts.
3. **Model Improvements**:
    - Fine-tune XGBoost hyperparameters to improve performance
    - Explore feature engineering for machine learning models
    - Consider adding external variables (weather, events, promotions) to improve predictions
4. **Monitoring and Updates**: Implement regular model retraining and performance monitoring to maintain forecast accuracy.

---

8. Conclusion

This project successfully demonstrated the application of various statistical and machine learning models for sales forecasting. SARIMA emerged as the most effective model, providing accurate predictions for weekly pizza sales. The insights and models developed here can aid in optimizing inventory management and improving business decisions.

8.1 Deliverables:

1. Cleaned and preprocessed datasets.
2. Exploratory Data Analysis report.
3. Predictive models with code and evaluation metrics.
4. Detailed purchase order for the next week.
5. Github repository.
6. Comprehensive Project Report