Bhartiya Vidya Bhavan's

Sardar Patel Institute of Technology

(Autonomous Institute Affiliated to University of Mumbai)

Department of Computer Engineering

RETAIL SALES ANALYSIS AND FORECASTING

By

Harsh Sahu (2023300192)
Aarya Rajwade (2023300179)
Sarthak Rajhans (2023300177)
Tanish Rane (2023300184)

Guided by

Prof. Abhijeet Salunke

Course Project

Python for Data Science (S.Y.)

Abstract

This retail sales analysis project aims to identify patterns and trends in sales data to enhance decision-making and boost revenue. Key objectives include data preprocessing, exploratory data analysis (EDA), feature engineering, and building machine learning models to predict sales performance accurately. The process began with data cleaning to handle missing values and inconsistencies, followed by an in-depth EDA to understand data distribution and trends. Feature extraction and selection were then performed to optimize model input and reduce noise, ensuring relevant variables were utilized effectively.

Several machine learning models were trained and evaluated, with a focus on fine-tuning hyperparameters for improved accuracy. Additionally, Power BI was employed to visualize the insights and findings, making it easier to interpret the data from a business perspective. Despite testing various algorithms, the Extra Trees Regressor consistently outperformed others, achieving an R^2 score above 97.6%, signifying its accuracy and robustness in predicting sales outcomes. This analysis provides actionable insights that can be leveraged to drive strategic sales and marketing initiatives, ultimately contributing to increased profitability in the retail sector.

Introduction

1. **Problem Statement**
   In the highly competitive retail industry, accurately predicting sales is essential for optimizing inventory, improving resource allocation, and maximizing profitability. However, due to seasonal trends, varying customer preferences, and external factors, achieving consistent sales forecasts is challenging. This project addresses the need for a robust predictive model to analyze retail sales data effectively.

2. **Objective**
   The primary goal of this project is to develop an accurate model to predict retail sales, identify key influencing factors, and provide actionable insights to support strategic business decisions.

3. **Motivation**
   Sales forecasting is critical for retail businesses to maintain competitive advantage, minimize overstock/understock issues, and enhance customer satisfaction. By accurately predicting sales, companies can optimize operations and make data-driven decisions that improve overall efficiency and profitability.

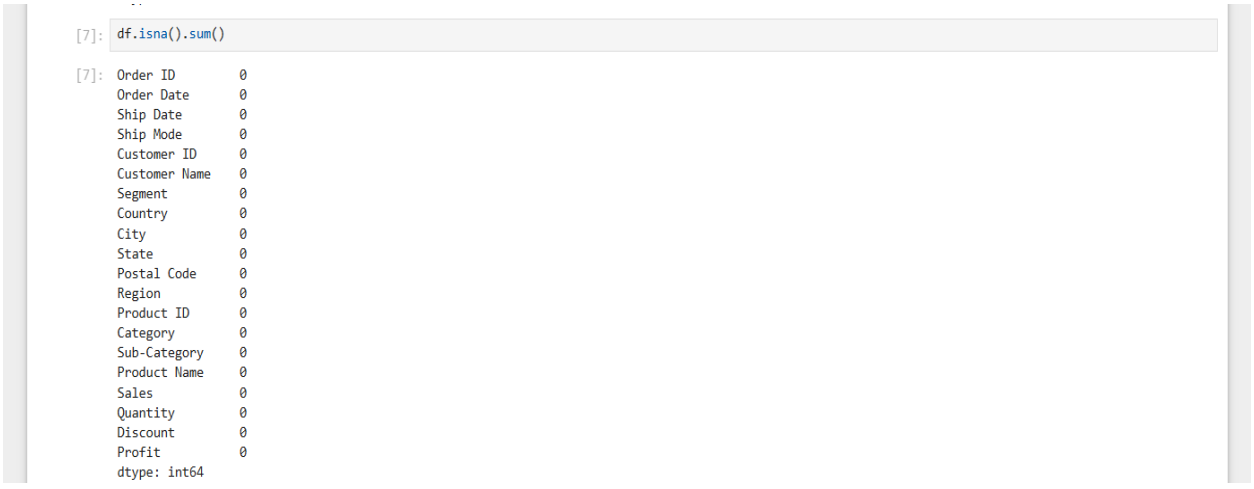
4. **Outline**
   - **Data Preprocessing:** Steps taken for data cleaning, handling missing values, and preparing data for analysis.
   - **Exploratory Data Analysis (EDA):** Analyzing data distribution, trends, and insights.
   - **Feature Engineering:** Extracting and selecting relevant features for model training.
   - **Model Training and Evaluation:** Training various ML models, tuning hyperparameters, and comparing performance.
   - **Visualization with Power BI:** Visual representations of key findings and insights.
   - **Conclusion:** Summary of results, insights, and potential improvements.

Dataset

- **Source**: This dataset appears to come from a retail or business context, likely related to sales data for a company that handles orders for various product categories.
- **Size**: The dataset contains 21 columns, with each row representing a unique order line item.
- **Type of Data**: The data includes categorical, numerical, and date features.
- **Features**:
    - **Order Details**: Includes Order ID, Order Date, Ship Date, and Ship Mode.
    - **Customer Details**: Includes Customer ID, Customer Name, Segment, and geographic information like City, State, Region, and Country.
    - **Product Information**: Contains Product ID, Category, Sub-Category, and Product Name.
    - **Sales Metrics**: Contains Sales, Quantity, Discount, and Profit.

Preprocessing

- **Handling Missing Data**: We performed an analysis of missing values in the dataset to ensure data completeness. Each column was carefully inspected for missing entries. The results of the analysis showed that no columns contained missing values, indicating that the dataset was complete and did not require imputation or removal of rows/columns.



```
[7]: df.isna().sum()

[7]: Order ID        0
     Order Date      0
     Ship Date       0
     Ship Mode       0
     Customer ID     0
     Customer Name   0
     Segment         0
     Country         0
     City            0
     State           0
     Postal Code     0
     Region          0
     Product ID      0
     Category        0
     Sub-Category    0
     Product Name    0
     Sales           0
     Quantity        0
     Discount        0
     Profit          0
     dtype: int64
```

- **Feature Engineering**: We created new features to enhance the dataset's predictive power. Key derived columns include profitability metrics such as Profit Margin and Discounted Profit, along with time-based features like Order Year, Order Month, and Order Weekday. Additionally, we calculated operational metrics such as Operating Expenses and Net Profit. Dates were split into granular components (year, month, day, and weekday) for both order and shipping dates, and the original date columns were removed for simplicity.

```
<class 'pandas.core.frame.DataFrame'>
Index: 9993 entries, 0 to 9993
Data columns (total 26 columns):
 #   Column               Non-Null Count  Dtype
---  ------               --------------  -----
 0   Ship Mode            9993 non-null   object
 1   Segment              9993 non-null   object
 2   City                 9993 non-null   object
 3   State                9993 non-null   object
 4   Postal Code          9993 non-null   int64
 5   Region               9993 non-null   object
 6   Category             9993 non-null   object
 7   Sub-Category         9993 non-null   object
 8   Product Name         9993 non-null   object
 9   Sales                9993 non-null   float64
 10  Quantity             9993 non-null   int64
 11  Discount             9993 non-null   float64
 12  Profit               9993 non-null   float64
 13  Profit Margin        9993 non-null   float64
 14  Discounted Profit    9993 non-null   float64
 15  Discount Percentage  9993 non-null   float64
 16  Operating Expenses   9993 non-null   float64
 17  Net Profit           9993 non-null   float64
 18  Order Year           9993 non-null   int32
 19  Order Month          9993 non-null   int32
 20  Order Day            9993 non-null   int32
 21  Order Weekday        9993 non-null   int32
 22  Ship Year            9993 non-null   int32
 23  Ship Month           9993 non-null   int32
 24  Ship Day             9993 non-null   int32
 25  Ship Weekday         9993 non-null   int32
dtypes: float64(8), int32(8), int64(2), object(8)
memory usage: 1.8+ MB
```

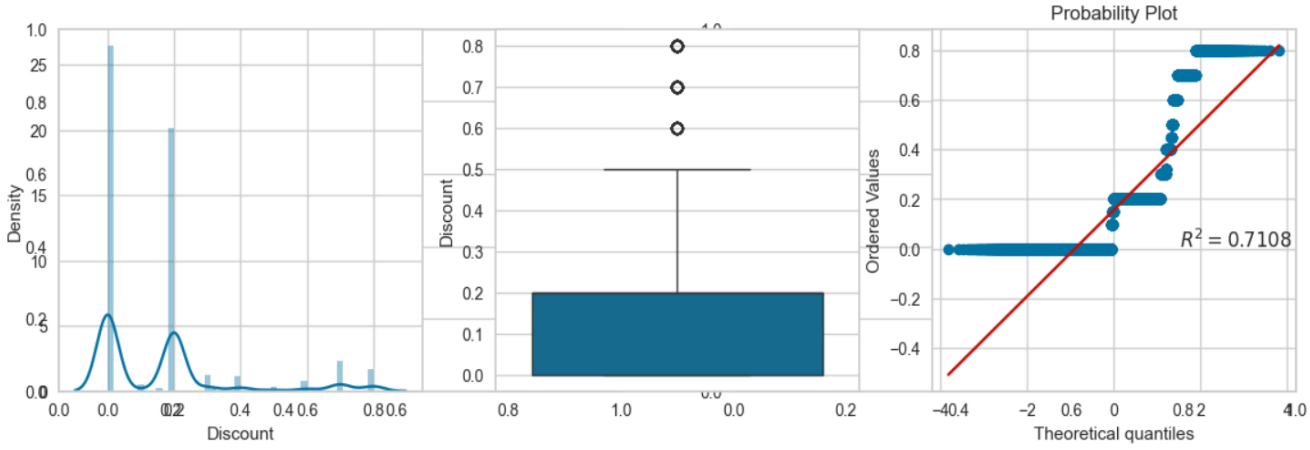
- **Feature Selection**: To improve model performance and reduce noise, we identified and removed features that were less relevant to our analysis. Columns such as Row ID, Customer Name, Product Name, and Product ID were dropped, as they were unique identifiers or text-based fields that did not contribute meaningfully to predictive modelling.

- **Normalization and Standardization**:
  We used **StandardScaler** to standardize the dataset, ensuring each feature had a mean of 0 and a standard deviation of 1. The scaler was fit on the training data and applied to both training and test sets for consistency.

Summary of Data Analysis

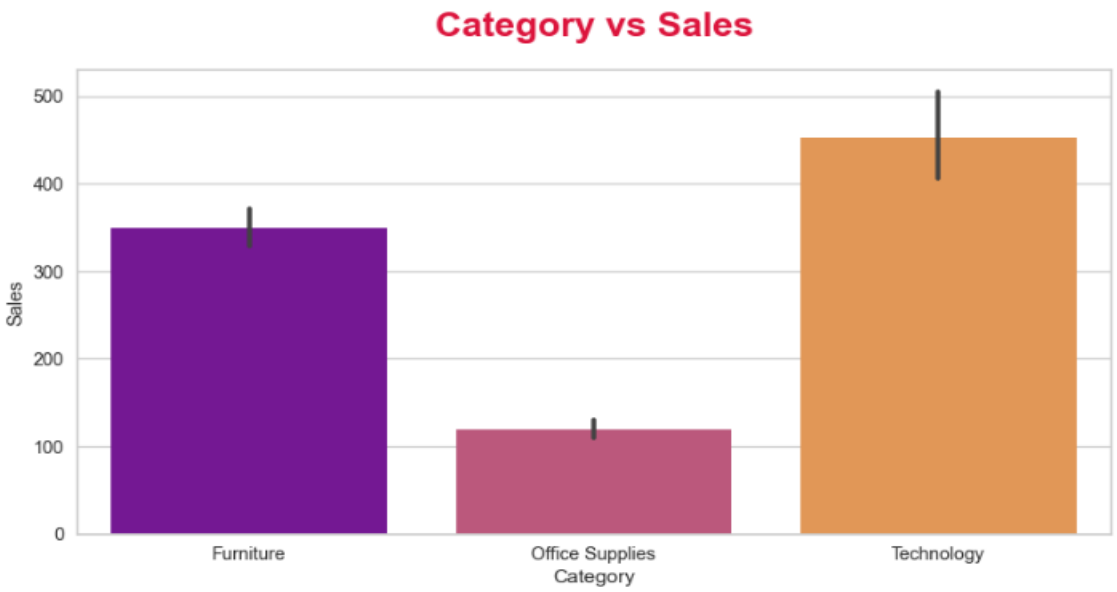
Univariate Analysis

1. Skewness and Kurtosis:
    - Numerical columns were assessed for skewness and kurtosis to evaluate their distribution.
    - Significant skewness was observed in features like Quantity, Discount, and Profit, indicating non-normal distributions.
    - Graphical representations such as histograms, boxplots, and Q-Q plots were used for visualization.
2. Category Distributions:
    - The majority of products belonged to Office Supplies, with Technology being the least represented category.
    - Pie charts illustrated categorical distributions, revealing that Western Region customers were the most dominant, followed by the Southern Region with the least.
3. Shipping Mode Analysis:
    - A significant proportion of orders utilized Standard Shipping, while First Class and Second Class modes were less frequent.



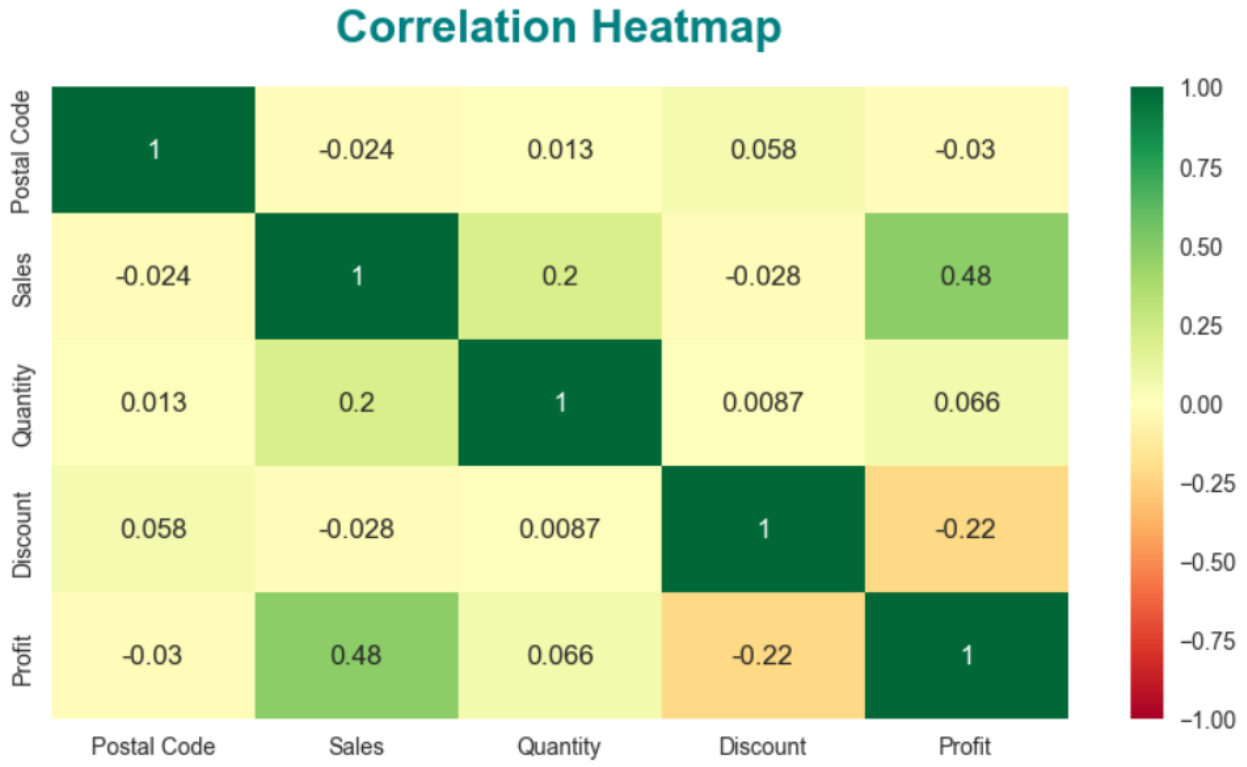
Bivariate Analysis

1. Profit Analysis:
    - The First Class shipping mode proved most profitable, while Standard Shipping generated the least profit.
    - Orders from the Home Office Segment achieved the highest profit compared to other customer segments.
2. Sales Analysis:
    - Highest sales were associated with Second Class and Same Day shipping modes.
    - The Technology Category generated maximum sales, while Office Supplies accounted for the least.
3. Discount Trends:
    - Discounts were highest in the Furniture Category and for First Class Shipping, whereas Second Class Shipping received the lowest discounts.
4. Region-Wise Insights:
    - The Western Region showed the highest profitability, while the Central Region recorded the lowest.
    - Discounting trends were highest in the Central Region.



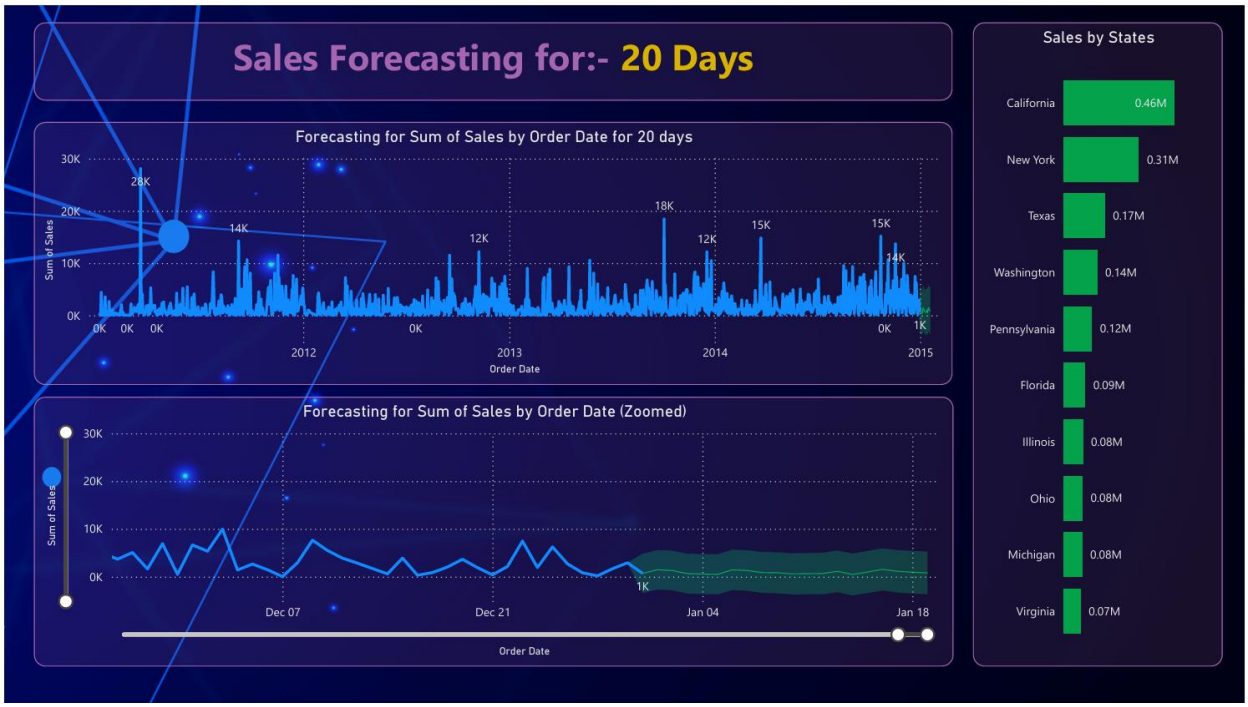
Multivariate Analysis

1. Correlation Heatmap:
    - Explored the relationships between features. Key insights include:
        - A moderate correlation between Sales and Profit.
        - Discounts showed a negative impact on Profit, indicating diminishing returns from excessive discounts.
2. Cluster Map and Pair Plots:
    - Clustering revealed patterns in numerical data. The Ship Mode category showcased significant variations across different metrics.
    - Pair plots highlighted outliers and trends in subsets of the dataset.
3. Scatter Plots:
    - Explored Sales vs. Profit across categories and regions.
    - Highest profits were observed in Technology, while Office Supplies struggled across regions.


Correlation Heatmap
Postal Code
Sales
Quantity
Discount
Profit
1
-0.024
0.013
0.058
-0.03
-0.024
1
0.2
-0.028
0.48
0.013
0.2
1
0.0087
0.066
0.058
-0.028
0.0087
1
-0.22
-0.03
0.48
0.066
-0.22
1
Postal Code
Sales
Quantity
Discount
Profit
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00


Sales vs Profit
Profit
8000
6000
4000
2000
0
−2000
−4000
−6000
0
5000
10000
15000
20000
Sales
Ship Mode
Second Class
Standard Class
First Class
Same Day

Power BI Dashboard Insights



1. Introduction

The Power BI dashboard analyzed sales, profits, and other key performance metrics across various dimensions, such as category, shipping modes, customer segments, and regions. It also includes a sales forecast for 20 days to predict future trends.

2. Key Insights

2.1 Sales and Profit by Category

- **Technology** and **Furniture** categories performed similarly with sales around **$0.17M each**, while **Office Supplies** slightly lagged with **$0.16M**.
- This indicates balanced sales across categories, but deeper profit analysis could highlight which category is more profitable.

2.2 Sales by Ship Mode

- **Standard Class** dominated with **$0.32M**, followed by **Second Class** at **$0.10M** and **First Class** at **$0.06M**.
- Most customers prefer affordable shipping options, reflecting possible cost sensitivity.

2.3 Monthly Sales and Profit Trends

- Sales and profit exhibited seasonal variations over three years (2012–2014). Peaks likely correspond to holiday seasons or promotional campaigns.
- The trend suggests identifying and capitalizing on peak periods for maximizing revenue.

2.4 Regional Sales Distribution

- **California** and **New York** were the top-performing states, generating **$0.46M** and **$0.31M** in sales, respectively.
- **Texas** and **Washington** followed, showing opportunities to expand marketing efforts in states with lower sales.

2.5 Segment Analysis

- The **Consumer** segment contributed to **50%** of total sales, followed by **Corporate** (**32%**) and **Home Office** (**18%**).
- This highlights the dominance of individual consumers and potential for growth in the Home Office segment.

2.6 Overall Metrics

- **Total Sales:** $501.24K,**Quantity Sold:** 8,780 units,**Average Delivery Time:** 4 days,**Profit:** $39.71K

2.7 Forecasting

- Sales forecasts for 20 days post-2014 show an increasing trend, with potential peaks around specific dates. It reflects positive growth momentum.

3. Strategic Recommendations

1. **Category-Level Optimization**: Focus on improving profit margins for Office Supplies while maintaining competitive pricing for Technology and Furniture.
2. **Shipping Strategy**: Enhance the customer experience for Standard Class, as it dominates shipping preferences.
3. **Geographical Focus**: Target underperforming states like Michigan and Virginia with tailored marketing campaigns.
4. **Segment Expansion**: Develop products and strategies for Home Office customers to capture a larger market share.
5. **Seasonal Promotions**: Align marketing efforts with peak sales periods to maximize profits.

4. Conclusion

The Power BI dashboard provides actionable insights into sales trends, customer preferences, and regional performance. Implementing the outlined strategies can drive improved sales and profitability while optimizing customer experience.

Methodology

We used a variety of machine learning models which captures all the methods of predicting the correct values.Following is the list of the machine learning models used along with their performance metrics

Machine Learning Models Used for Sales Prediction

1. **ExtraTreesRegressor**
   An ensemble method that reduces overfitting by averaging multiple decision trees, making it ideal for handling non-linear relationships in sales data.

2. **XGBRegressor**
   A gradient-boosting model known for its high performance and speed, making it suitable for complex and large sales datasets with noise and non-linear patterns.

3. **RandomForestRegressor**
   A robust ensemble method that reduces variance by averaging predictions from multiple trees, ideal for handling both numerical and categorical sales data.

4. **CatBoostRegressor**
   Specifically designed for categorical data, it efficiently prevents overfitting, making it perfect for sales forecasting with various categorical features.

5. **LGBMRegressor**
   A fast gradient-boosting model that excels in processing large datasets with low latency, ideal for real-time sales prediction.

6. **BaggingRegressor**
   An ensemble method that builds multiple models on random data subsets, improving accuracy and robustness, especially in fluctuating sales data.

7. **XGBRFRegressor**
   A hybrid model combining XGBoost and Random Forest principles, adding randomness to improve model diversity and generalization for sales prediction.

8. **DecisionTreeRegressor**
   A simple, interpretable model that splits data based on thresholds, useful for capturing straightforward relationships in sales data.

9. **HistGradientBoostingRegressor**
   A histogram-based boosting model that efficiently handles large datasets, making it an excellent choice for fast and accurate sales prediction.

10. **NGBRegressor**
    A probabilistic boosting model that outputs distributions instead of point estimates, offering a more flexible and nuanced approach to sales forecasting.

In total, we applied 31 models, but these are the most significant for our sales prediction task.

Model Implementation

Training and Testing:

For training and testing the models, we employed the **train-test split** technique to divide the dataset into training and testing subsets. The training subset was used to fit the models, while the testing subset was reserved for evaluating the performance of each model. This approach helps ensure that the model is evaluated on data it has not seen before, providing a better estimate of its generalization ability..

Hyperparameter Tuning:

To fine-tune the models and improve their performance, we used **RandomizedSearchCV** with a specified parameter grid for each model. This method helps identify the optimal hyperparameters by evaluating random combinations from the grid, reducing the search space and computational time compared to **GridSearchCV**.

Additionally, to ensure robust evaluation, we performed **cross-validation** for all models. Specifically, we used **5-fold cross-validation**, where the data is divided into five subsets. The model is trained on four of these subsets and tested on the remaining subset.

FEATURE SELECTION AND ENGINEERING

Initial Feature Selection

We began by analyzing the dataset and removing irrelevant or redundant features. Specifically, we dropped **Order Date** and **Ship Date** after extracting useful time-related features such as **Order Year**, **Order Month**, and **Ship Year**. Categorical columns were encoded using **one-hot encoding** and removed, allowing the dataset to focus solely on numerical features.

Skewness and Kurtosis Analysis

To understand the distribution of numerical features, we analyzed their **skewness** and **kurtosis**. Skewed features were addressed by applying transformations, including **logarithmic**, **square root**, and **power transformations**, to normalize the data and reduce the impact of skewness.

Correlation Analysis

We conducted a **correlation analysis** to identify highly correlated features and prevent multicollinearity. Using the **DropCorrelatedFeatures** method, we eliminated redundant features that could negatively affect the model's performance and accuracy.

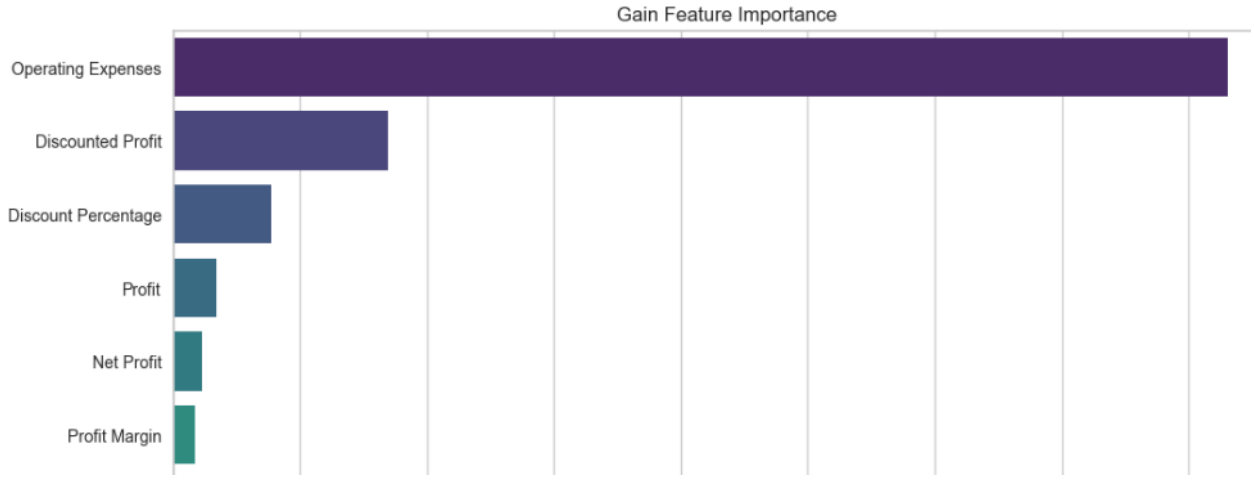
Statistical Feature Selection Techniques

Several statistical methods were applied to select the most significant features:

- **SelectKBest**: We used the **f_regression** method to identify the top 10 features based on their F-statistics, including **Profit**, **Discount Percentage**, **Postal Code**, **Discount**, **Order Day**, **Order Month**, **Quantity**, **Operating Expenses**, **Ship Day**, and **Order Weekday**.
- **SelectPercentile**: We applied **SelectPercentile** to select the top 20% of features based on their F-statistics.

- **SelectFromModel**: Using **Ridge regression** and **RandomForestRegressor**, we identified important features based on their importance scores and selected the most impactful ones.
- **Recursive Feature Elimination (RFE)**: RFE with **RandomForestRegressor** was employed to recursively eliminate less important features, keeping only the top 10 features.
- **Sequential Feature Selector (SFS)**: We used backward selection with **Ridge regression** to identify the optimal subset of features.

Gain Importance :



Final Selected Columns:

	Profit	Discount Percentage	Postal Code	Discount	Order Day	Order Month	Quantity	Operating Expenses	Ship Day	Order Weekday
5579	-15.2450	0.081994	90004.0	0.2	10.0	9.0	5.0	259.1650	15.0	2.0
2728	94.4937	0.063496	43615.0	0.4	23.0	12.0	7.0	535.4643	25.0	1.0
477	7.1820	0.208855	90008.0	0.2	13.0	7.0	6.0	88.5780	20.0	5.0
1329	16.3020	0.000000	94109.0	0.0	24.0	5.0	3.0	20.7480	28.0	4.0
4544	16.7040	0.134698	60610.0	0.2	12.0	7.0	2.0	131.7760	17.0	3.0

Experimental Setup

Tools Used

The following software, libraries, and tools were utilized to implement and evaluate the model:

- **Python**: Core programming language used for data preprocessing, model training, and evaluation.
- **Scikit-learn**: Employed for model implementation, data splitting, preprocessing, and metrics computation.
- **Matplotlib**: Used for visualizing the ROC curve, precision-recall curve, and other plots.
- **Pandas**: For data manipulation and analysis.
- **Numpy**: Used for numerical computations and efficient handling of data structures.

Hardware/Environment

The experiments were conducted in the following computational environment:

- **Processor**: Intel Core i5 (or equivalent).
- **RAM**: 8 GB.
- **Operating System**: Windows 10.
- **GPU**: Not used; computations were performed on the CPU.

Evaluation Metrics

The evaluation metrics used in this project include Mean Absolute Error (MAE) to measure average prediction errors, Mean Squared Error (MSE) and Root Mean Squared Error (RMSE) to assess error magnitude, with RMSE offering interpretability in target units. Mean Absolute Percentage Error (MAPE) evaluates prediction accuracy as a percentage, while R-squared (R^2) indicates the proportion of variance explained by the model. Additionally, training time was tracked to compare computational efficiency across models.

Results and Discussion

Performance Comparison:

	Model	Train R2	Test R2	Train MAE	Test MAE	Train RMSE	Test RMSE	Training Time
0	ExtraTreesRegressor	1.000000	0.974789	2.887998e-13	12.012504	8.852335e-13	82.314801	2.391563
1	XGBRegressor	0.999946	0.964563	2.128146e+00	17.091720	3.911030e+00	97.590917	0.245141
2	RandomForestRegressor	0.994343	0.961377	4.747117e+00	14.263193	3.987807e+01	101.883489	4.677731
3	CatBoostRegressor	0.999531	0.960660	5.065879e+00	17.901773	1.148560e+01	102.824565	4.294122
4	LGBMRegressor	0.963835	0.959196	1.290775e+01	18.645762	1.008298e+02	104.720411	0.341861
5	BaggingRegressor	0.991795	0.958994	5.802398e+00	15.596935	4.802730e+01	104.980137	0.484203
6	XGBRFRegressor	0.989208	0.956297	1.544078e+01	21.353970	5.508112e+01	108.377236	0.207962
7	DecisionTreeRegressor	1.000000	0.952145	1.527410e-16	16.188377	1.584261e-15	113.408094	0.123786
8	HistGradientBoostingRegressor	0.966856	0.950026	1.308215e+01	20.295129	9.652748e+01	115.892593	0.729304
9	NGBRegressor	0.995870	0.949756	1.144696e+01	19.877388	3.407371e+01	116.204847	18.580468
10	GradientBoostingRegressor	0.995637	0.949703	1.218050e+01	20.742806	3.502329e+01	116.265781	1.229091
11	PoissonRegressor	0.908815	0.922972	3.542472e+01	36.553124	1.601060e+02	143.881530	0.204451
12	MLPRegressor	0.631444	0.691982	9.873691e+01	99.251464	3.218836e+02	287.720174	6.420322
13	KNeighborsRegressor	0.713503	0.621294	6.604304e+01	81.891782	2.837965e+02	319.031130	0.024451
14	BayesianRidge	0.341383	0.377960	2.085742e+02	213.060720	4.302923e+02	408.875721	0.041935
15	ARDRegression	0.341182	0.377954	2.087011e+02	213.158281	4.303580e+02	408.877450	0.088074
16	Lasso	0.341397	0.377895	2.089565e+02	213.414098	4.302878e+02	408.896952	0.029473
17	Ridge	0.341397	0.377890	2.089645e+02	213.420708	4.302878e+02	408.898694	0.000000
18	LinearRegression	0.341397	0.377890	2.089646e+02	213.420727	4.302878e+02	408.898696	0.031267
19	SGDRegressor	0.340939	0.376512	2.065598e+02	211.178228	4.304372e+02	409.351260	0.040361
20	GammaRegressor	0.342780	0.362304	1.168747e+02	121.885857	4.298357e+02	413.988994	0.010511
21	ElasticNet	0.303980	0.335042	1.844454e+02	188.946697	4.423417e+02	422.745752	0.011743
22	TweedieRegressor	0.288041	0.316624	1.822220e+02	186.833894	4.473780e+02	428.560243	0.055073
23	AdaBoostRegressor	0.404968	0.311541	3.831546e+02	391.103376	4.089941e+02	430.151250	0.435519
24	TheilSenRegressor	0.231232	0.250506	1.515614e+02	157.522838	4.648842e+02	448.813636	3.836542
25	HuberRegressor	0.204014	0.221991	1.488244e+02	154.626681	4.730420e+02	457.271804	0.084596
26	PassiveAggressiveRegressor	0.164265	0.179346	1.492770e+02	154.590311	4.847092e+02	469.636776	0.015640
27	LinearSVR	0.164107	0.178452	1.478663e+02	153.552981	4.847550e+02	469.892403	0.048045
28	NuSVR	0.140934	0.148038	1.358590e+02	142.580677	4.914284e+02	478.511159	1.846104
29	SVR	0.127221	0.132530	1.330009e+02	139.800594	4.953352e+02	482.846805	2.504212
30	RANSACRegressor	0.085671	0.084803	1.729179e+02	179.472832	5.069886e+02	495.951616	0.299825

Error Analysis

Analysis of Model Underperformance:

1. Overfitting:

- **Model:** DecisionTreeRegressor
- **Symptoms:** Perfect performance on the training set (Train $R^2 = 1.000000$) with a significant drop on the test set (Test $R^2 = 0.950855$).
- **Explanation:** The DecisionTreeRegressor is likely overfitting because it can memorize specific patterns in the training data, but fails to generalize well to unseen data. Decision trees are prone to overfitting, especially when the tree depth is not controlled.
- **Potential Reasons for Overfitting:**
    - The tree might be too deep, capturing noise in the data rather than the underlying patterns.
    - Lack of pruning or regularization techniques.
- **Suggested Improvements:**
    - **Pruning the tree** by limiting the maximum depth or using parameters like max_depth, min_samples_split, and min_samples_leaf to control the complexity of the tree.
    - **Use cross-validation** to select the best hyperparameters to avoid fitting to noise.
    - **Ensemble methods** like Random Forest or Gradient Boosting can help mitigate overfitting by averaging predictions across many trees.

2. Underfitting:

- **Models:** PoissonRegressor, MLPRegressor, KNeighborsRegressor, SGDRegressor, Lasso, Ridge, ElasticNet, TweedieRegressor, TheilSenRegressor, HuberRegressor, PassiveAggressiveRegressor, LinearSVR, NuSVR, SVR, RANSACRegressor
- **Symptoms:** These models have low performance on both training and test sets, with low R^2 values for both (e.g., Train R^2 for **PoissonRegressor** = 0.908815, Test $R^2 = 0.922972$).
- **Explanation:** These models likely underfit the data, meaning they are too simple to capture the complexity of the relationships in the dataset. Models with high bias and low

variance are often underfitting. For instance, **Lasso**, **Ridge**, and **ElasticNet** are regularized linear models that tend to underfit when the dataset's true relationship is nonlinear.

- **Potential Reasons for Underfitting:**
    - **Insufficient model complexity**: Simple models like **LinearSVR**, **SVR**, and **Lasso** are unable to capture the complex patterns in the data, especially if the data exhibits non-linear relationships.
    - **Hyperparameter tuning**: For models like **MLPRegressor**, the architecture (e.g., number of layers or neurons) may not be optimized for the task.
    - **Too much regularization**: Over-regularization in models like **Lasso**, **Ridge**, or **ElasticNet** can lead to a model that is too constrained and fails to learn adequately.
- **Suggested Improvements:**
    - **Increase model complexity**: For example, use more complex models like **RandomForestRegressor** or **XGBRegressor** that can capture non-linearities in the data.
    - **Tune hyperparameters**: For models like **MLPRegressor**, try adjusting the number of layers, neurons, and learning rate.
    - **Reduce regularization**: For linear models like **Lasso**, **Ridge**, and **ElasticNet**, consider reducing the regularization strength to allow for more flexibility in fitting the data.
    - **Try non-linear models**: For models like **SVR** or **LinearSVR**, explore using **PolynomialSVR** or other kernel-based methods to better capture complex relationships.

Conclusion

This project aimed at analyzing Superstore data to understand patterns and predict Sales using machine learning models. We started with Exploratory Data Analysis (EDA), followed by feature selection techniques like SelectKBest, RFE, and XGBoost to identify key predictors. We then applied several machine learning models, tuning hyperparameters with RandomizedSearchCV. We also used Power BI to create an interactive dashboard to visualize trends and insights from the data.

Findings

The analysis revealed that Profit, Discount Percentage, and Quantity were the strongest predictors of Sales. The Gradient Boosting Regressor model performed best. The Power BI dashboard effectively showcased trends, product performance, and regional insights.

Limitations

The primary limitation was the size of the dataset, which could have restricted the performance of some models. Additionally, the dataset might contain bias, and the lack of real-time data updates limits the model's adaptability.

Future Work

In the future, we plan to develop a web-based application for real-time predictions, improving scalability. We also aim to explore larger datasets, test more advanced models, and refine feature engineering. Enhancements to the Power BI dashboard will focus on dynamic visualizations and real-time data updates.

Timesheet

- The breakage of 20 hours of work put by each individual team member.

Sarthak Rajhans (2023300177) contributed to the project by researching and inspecting data (1.5 hours), performing data preprocessing (2 hours), analyzing data distribution (1.5 hours), exploring categorical variables (2 hours), visualizing region and segment (2 hours), conducting ship mode analysis (2 hours), analyzing scatterplot relationships (2 hours), comparing categories (2 hours), creating a correlation heatmap (2 hours), performing pairplot analysis (2 hours), and finalizing and cleaning up the work (1.5 hours). Total - 20.5 hours

Harsh Sahu contributed significantly to the project, dedicating time to various critical tasks: researching the dataset (2 hours), removing outliers (2 hours), handling missing values (1.5 hours), feature selection (2 hours), analyzing feature importance (2 hours), encoding categorical variables (1.5 hours), scaling and normalizing data (2 hours), splitting the dataset into training and testing sets (1 hour), , analyzing overfitting and underfitting (2 hours), comparing evaluation metrics (1.5 hours), and documenting results and insights (1.5 hours). Total time contributed: 21 hours.

Aarya Rajwade (2023300179) contributed a quite important part to the project which was implementation of machine learning models and hyperparameter tuning.It included researching and shortlisting machine learning models which would be relevant to predict based on the selected dataset.He shortlisted around 20 30 models.Finding relevant models and implementing them itself made up for around 10 to 12 hours which included solving syntax errors , figuring out parameter and conceptual prompts , after that evaluation metrics and hyperparameter tuning took around 7-8 hours.

Tanish Rane (2023300184) contributed to the project by refining the introduction (1 hour), inspecting the Power BI dashboard for key metrics (2 hours), validating the sales forecast (1 hour), analyzing insights like category performance and shipping modes (2 hours), exploring regional and segment-wise trends (2 hours), drafting strategic recommendations (3 hours), reviewing theoretical principles for strategy support (1 hour), compiling and writing the report (2 hours), polishing dashboard visuals (1 hour), rehearsing the presentation (1 hour), and finalizing the project for submission (1 hour). Total – 20 hours.

References

1. Géron, A. (2019). *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*. O'Reilly Media.
2. Towards Data Science Blog. (n.d.). *How to Handle Missing Data in Machine Learning*. Retrieved from https://towardsdatascience.com
3. Scikit-learn. (n.d.). *Model Evaluation Documentation*. Retrieved from https://scikit-learn.org