A PROJECT REPORT

on

"Order Amount Prediction"

Submitted to

KIIT Deemed to be University

In Partial Fulfilment of the Requirement for the Award of

BACHELOR'S DEGREE IN

COMPUTER SCIENCE AND ENGINEERING

BY

PUNYAPU SAITEJA 2005184

UNDER THE GUIDANCE OF

HIGHRADIUS TECHNOLOGIES

SCHOOL OF COMPUTER ENGINEERING

KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY

BHUBANESWAR, ODISHA - 751024

August 2023

A PROJECT REPORT
on

"Order Amount Prediction"

Submitted to
KIIT Deemed to be University

In Partial Fulfilment of the Requirement for the Award of

BACHELOR'S DEGREE IN
COMPUTER SCIENCE AND ENGINEERING
BY

PUNYAPU SAITEJA 2005184

UNDER THE GUIDANCE OF
HIGHRADIUS TECHNOLOGIES

SCHOOL OF COMPUTER ENGINEERING
KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY
BHUBANESWAR, ODISHA - 751024
August 2023

KIIT Deemed to be University
School of Computer Engineering
Bhubaneswar, ODISHA 751024

CERTIFICATE

This is to certify that the project entitled
"Order Amount Prediction"
submitted by

PUNYAPU SAITEJA 2005184

is a record of bonafide work carried out by them, in the partial fulfillment of the requirement for the Degree of Bachelor of Engineering award at KIIT Deemed to be University, Bhubaneswar. This work will be done during the year 2022-2023.

Date: 05/08/2023

Mridul
Project Guide

Acknowledgments

We extend our sincere gratitude to HIGHRADIUS TECHNOLOGIES for their expert guidance and continuous encouragement throughout this project. Their support has been instrumental in ensuring that the project successfully achieved its objectives from inception to completion.

PUNYAPU SAITEJA

ABSTRACT

The "Order Amount Prediction" project aims to build a robust Machine Learning model to predict the order amounts that customers might place in the upcoming days for B2B operations. The project focuses on enhancing credit management and order processing efficiency by providing accurate order amount predictions.
The initial phase of the project involves data preprocessing and cleaning to ensure data integrity. Missing values are handled, date columns are formatted, and special characters are removed from the order amount field. Feature engineering techniques are applied to create additional features that can potentially improve the model's predictive power.

In the Exploratory Data Analysis (EDA) phase, data visualizations are utilized to gain insights into data distribution, patterns, and potential outliers. These insights guide the feature engineering process and provide a better understanding of the dataset's behavior.

The ML Models and Evaluations phase entails implementing various machine learning algorithms, including Linear Regression, Support Vector Machine, Decision Tree, Random Forest, AdaBoost, and XGBoost. Model evaluations are performed using metrics such as Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and R-Squared (R^2) to identify the best-performing model. Hyperparameter tuning is employed to optimize the selected model's performance further. The final model is chosen based on its accuracy and ability to predict order amounts effectively.

Throughout the project, data sanity checks, exploratory data analysis, and model evaluations are conducted to ensure the integrity and reliability of the results. The project aims to deliver an accurate and efficient machine-learning solution to assist businesses in predicting order amounts for effective credit management and streamlined order processing in the B2B domain.

Contents

List of Images

Abbreviations

B2B - Business to Business

B2C - Business to Consumer

C2C - Consumer to Consumer

EDA - Exploratory Data Analysis

ML - Machine Learning

MSE - Mean Squared Error

RMSE - Root Mean Squared Error

MAE - Mean Absolute Error

R^2 - R-squared (Coefficient of determination)

SVM - Support Vector Machine

UI - User Interface

PRS - Purchase Request System

CSV - Comma-Separated Values

ERP - Enterprise Resource Planning

Chapter 1

Introduction

The objective of the "Order Amount Prediction" project is to develop a robust Machine Learning model that can accurately predict the order amount that customers might place in the upcoming days. This project aims to support B2B (Business to Business) operations, which operate differently from B2C (Business to Consumer) or C2C (Consumer to Consumer) businesses. In B2B operations, businesses work with other businesses on credit. When a buyer business places an order for goods from a seller business, the seller business issues an invoice for the purchased goods. This invoice contains essential information, such as details of the goods purchased and the due date for payment.
One of the critical aspects in B2B operations is managing accounts receivable, which represents money owed by customers to the seller business for products or services sold on credit. The credit check department plays a crucial role in this process by validating customers, verifying their available credit limits, checking purchase orders, and managing the entire process of order inflow. This department ensures that the company can provide services and supply products without affecting the cash inflow and helps in avoiding blocked orders (when customers try to place orders above their remaining credit limits).

About High Radius:

HighRadius is a leading fintech enterprise Software-as-a-Service (SaaS) company that specializes in automating and optimizing the Order-to-Cash (O2C) and Treasury Management processes for businesses. Founded in 2006, the company has grown rapidly and established itself as a key player in the accounts receivable and treasury automation space.

The primary focus of HighRadius is to help businesses achieve greater efficiency and accuracy in their financial operations by leveraging advanced technologies such as artificial intelligence and machine learning. Their cloud-based solutions enable organizations to automate manual tasks, streamline workflows, and improve cash flow management.

Chapter 2

Theory

2.1 Linear Regression

Linear Regression is a fundamental supervised learning algorithm used for predictive analysis. It is a simple yet powerful technique to establish a relationship between a dependent variable (target) and one or more independent variables (features) by fitting a linear equation. The goal of linear regression is to find the best-fitted line that minimizes the difference between the predicted and actual values (residuals).

2.2 AdaBoost (Adaptive Boosting)

AdaBoost is an ensemble learning technique that combines multiple weak learners (typically decision trees) to create a strong learner. The algorithm focuses on instances that were misclassified by previous weak learners and assign higher weights to them, thereby giving more importance to difficult-to-classify instances.

Important Concepts:

- **Weak Learners:** Weak learners are classifiers that perform slightly better than random guessing. In the context of AdaBoost, they are usually decision stumps (single-level decision trees).
- **Weighted Voting:** Each weak learner's prediction is weighted based on its accuracy, and the final prediction is determined by majority voting with higher weights assigned to more accurate classifiers.
- **Boosting:** AdaBoost uses boosting, a technique where each subsequent model focuses more on the misclassified instances from previous models, effectively reducing the classification errors over iterations.

2.3 Random Forest

- Bagging: Random Forest employs bagging, which involves creating random subsets of the data with replacement. This reduces overfitting and increases the model's generalization ability.
- Feature Randomness: Each tree in the Random Forest is built using only a random subset of features. This randomization helps in decorrelating the trees and results in a more diverse set of classifiers.
- Out-of-Bag (OOB) Error: Random Forest uses a subset of data that is not included in the training of each tree to estimate the model's accuracy. This is known as the Out-of-Bag error and can serve as a validation metric.

.

Chapter 3

Requirement Specifications

3.1 Functional Requirements:

The project's functional requirements involve data preparation, model selection and training, and model selection.

In the data preparation phase, the team will load the provided order dataset into Jupyter Notebook and perform necessary data cleaning and preprocessing tasks. This includes handling missing values, converting date columns to DateTime format, and cleaning the order amount field. Additionally, feature engineering will be applied to create relevant features that can potentially enhance the model's performance.

For model selection and training, the team will implement and compare various Machine Learning models such as Linear Regression, AdaBoost, Random Forest, etc., to predict order amounts. Hyperparameter tuning will be conducted for each model to optimize its performance, and model evaluation will be performed using appropriate metrics like MSE, RMSE, and R-Squared to assess prediction accuracy.
Based on the evaluation results, the team will select the most robust and accurate ML model. Clear reasoning and justifications for the chosen model will be provided to support the selection.

3.2 Non-Functional Requirements:

The non-functional requirements focus on aspects such as performance, code maintainability, documentation, data security, version control, error handling, and performance comparison.

In terms of performance, the code should be efficient, considering potential usage with large datasets and complex calculations. The team will optimize memory usage to avoid any memory-related issues.

For code maintainability, I will structure the code into functions and modules, following consistent and meaningful naming conventions. This will ensure ease of maintenance and enhance code readability.

Comprehensive documentation will be prepared, including explanations of each step, model evaluation results, and the selected best model. Additionally, detailed comments will be included in the code to enhance understandability for future reference.

Chapter 4

Implementation

The implementation of the "Order Amount Prediction" project is done using Python within Jupyter Notebook. The project involves several key steps to achieve the objective of predicting order amounts for B2B operations.

Initially, the provided order dataset is loaded into a Pandas DataFrame for further analysis. Data cleaning and preprocessing tasks are performed to handle missing values, convert date columns to the appropriate format, and ensure the order amount field is numeric and free from any special characters.
Feature engineering techniques are applied to create additional relevant features from the existing data, potentially enhancing the model's predictive capabilities. These additional features can offer valuable insights and contribute to more accurate predictions.

Next, the dataset is split into training and testing sets to evaluate different Machine Learning models. Various models, such as Linear Regression, AdaBoost, and Random Forest, are implemented and trained on the training data. Model evaluation is carried out using suitable metrics to assess their performance on the testing data. Hyperparameter tuning is performed for each model to optimize its performance, enabling the identification of the best combination of hyperparameters for each algorithm.

The model with the best performance, as determined by the evaluation metrics, is selected for the final prediction of order amounts for upcoming days. The project report justifies the chosen model and provides explanations for the selection.
The entire implementation process is documented thoroughly in the Jupyter Notebook, with well-structured code, comments, and explanations to enhance readability and maintainability.

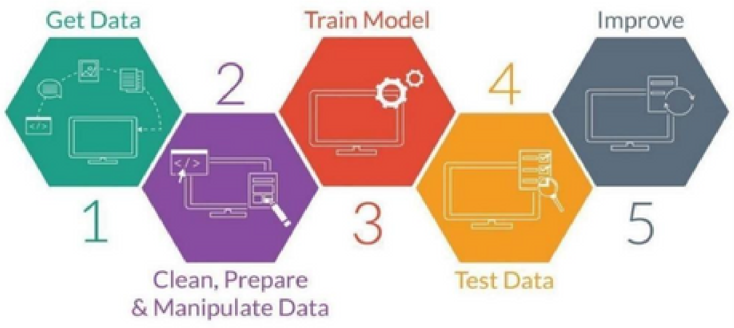


Fig 4.1

4.1 Setting up the environment

Setting up the environment for the "Order Amount Prediction" project involves installing the necessary software and libraries.

Install Anaconda:

- Download the latest version of Anaconda from the official website: https://www.anaconda.com/products/individual
- Follow the installation instructions specific to your operating system (Windows, macOS, or Linux).
- Anaconda includes Python, Jupyter Notebook, and many pre-installed data science libraries.

Create a New Conda Environment:

- Open Anaconda Navigator (included in the Anaconda installation).
- Click on the "Environments" tab and then "Create" to create a new environment.
- Give the environment a name (e.g., "order_prediction") and select the desired Python version (e.g., Python 3.x).

Open Jupyter Notebook:

- From the Anaconda Navigator, click on "Home" and find the "order_prediction" environment.
- Click "Launch" under the Jupyter Notebook section. A Jupyter Notebook will open in your web browser.

Install Required Libraries:

- In the Jupyter Notebook, open a new notebook or use an existing one.
- Install the necessary libraries using the following commands:

```
!pip install numpy
!pip install pandas
!pip install scikit-learn
!pip install matplotlib
!pip install seaborn
```

These libraries are essential for data manipulation, machine learning, and visualization tasks.

Load the Dataset:

- Once the environment is set up, load the provided order dataset into the Jupyter Notebook using Pandas.

4.2 Data Set

The data section in the "Order Amount Prediction" project report provides essential information about the dataset used for building the predictive model. The dataset contains valuable fields related to customer orders and their corresponding order amounts. Below is a comprehensive data dictionary describing each field present in the dataset:

1. CUSTOMER_ORDER_ID: A unique identifier for each order placed by a customer.
2. SALES_ORG: A unique identifier for the sales organization handling the customer order.
3. DISTRIBUTION_CHANNEL: Indicates the country where the shipment has been delivered.
4. DIVISION: Represents the region coverage or division associated with the customer.
5. RELEASED_CREDIT_VALUE: Denotes the total credit value that the customer possesses for transactions.
6. PURCHASE_ORDER_TYPE: Categorizes the type of purchase order made by the customer.
7. COMPANY_CODE: Represents the smallest organizational unit used for accounting purposes.
8. ORDER_CREATION_DATE: The date on which the order was created in the Enterprise Resource Planning (ERP) system.
9. ORDER_CREATION_TIME: The time at which the order was created in the ERP system.
10. CREDIT_CONTROL_AREA: Indicates the organizational unit responsible for managing customer credit limits.
11. SOLD_TO_PARTY: A unique identifier for the person or organization who placed the order.
12. ORDER_AMOUNT: The total sum of purchase prices in the purchase order(s).
13. REQUESTED_DELIVERY_DATE: Denotes the requested date of delivery by the customer.
14. ORDER_CURRENCY: Represents the currency in which the order was billed and paid.
15. CREDIT_STATUS: Provides an indication of the credit health of a particular customer.
16. CUSTOMER_NUMBER: A unique identifier for a specific customer.

The data section of the project report showcases the data dictionary to provide a clear understanding of the dataset's structure and the information contained in each field. It explains the relevance and importance of each feature in predicting the order amount for B2B operations.

4.3 Data Sanity

Data sanity is a critical step in any data analysis project to ensure the dataset is clean, consistent, and ready for further analysis and model building. In the "Order Amount Prediction" project, data sanity checks are performed to identify and address any data quality issues that may affect the accuracy and reliability of the predictive models. This report outlines the steps taken for data sanity checks and the actions performed to ensure the dataset's integrity.

1. Data Loading and Inspection:

- The provided order dataset is loaded into a Pandas DataFrame.
- The first few rows of the dataset are inspected to get a glimpse of the data and understand its structure.

2. Data Description:

- A summary of the dataset's size and structure is presented, including the number of rows (samples) and columns (features).
- Data types of each feature are checked to ensure they are appropriate for their respective data.

3. Handling Missing Values:

- The presence of missing values in the dataset is identified for each feature.
- Actions are taken to handle missing values, such as imputing them with suitable measures like mean, median, or mode, or removing rows with missing values if applicable.

4. Date Columns Formatting:

- Date columns, such as "ORDER_CREATION_DATE" and "REQUESTED_DELIVERY_DATE," are checked for proper formatting.
- The date columns are converted to DateTime format with the appropriate format (e.g., "%Y%m%d") for consistency and ease of handling.

5. Data Consistency Checks:

- Data consistency checks are performed to identify any discrepancies or anomalies in the dataset.
- For example, the relationship between the order creation date and the requested delivery date is verified to ensure logical consistency.

6. Handling Special Characters in "ORDER_AMOUNT":

- The "ORDER_AMOUNT" field is inspected for special characters, such as commas or currency symbols.
- Any special characters found are removed, and the field is converted to a numeric format to enable mathematical operations.

7. Sanity Check for Order Date and Delivery Date:
   - Records where the order creation date is greater than the requested delivery date are identified and examined.

8. Handling Negative and Invalid Order Amounts:
   - Records with negative or invalid order amounts (e.g., zero or null values) are checked and addressed.
   - Negative values are either removed or corrected, and invalid entries are handled appropriately.

9. Currency Conversion to USD:
   - The "ORDER_CURRENCY" field is examined to identify non-USD currencies.
   - Non-USD order amounts are converted to USD using suitable exchange rates to ensure uniformity in currency.

10. Creating Unique Customer IDs:
   - A new column, "unique_cust_id," is created by combining "CUSTOMER_NUMBER" and "COMPANY_CODE."
   - This new identifier helps to uniquely identify individual customers and their associated companies.

11. Data Distribution Check:
   - A check is performed on the distribution of the "ORDER_AMOUNT" field to identify any potential outliers or skewed data.

4.4 Exploratory Data Analysis

Exploratory Data Analysis (EDA) is a crucial phase in the "Order Amount Prediction" project as it helps in understanding the dataset's characteristics and uncover insights that can inform the model-building process. The EDA phase involves various data visualization techniques to gain a deeper understanding of the distribution, patterns, and potential outliers in the dataset. This report presents the key findings and visualizations from the EDA phase.

1. Distribution of Order Amounts:
   - A histogram is used to visualize the distribution of order amounts.
   - This histogram reveals the frequency distribution of order amounts and helps identify whether the data is skewed or follows a normal distribution.

2. Order Amounts by Purchase Order Type:
   - A bar chart is created to compare the order amounts across different purchase order types.
   - This visualization helps identify if certain types of purchase orders result in higher or lower order amounts.

3. Order Amounts by Distribution Channel:

- A bar chart is used to compare the order amounts across different distribution channels.
- This visualization provides insights into the variations in order amounts based on the country of delivery.

4. Order Amounts over Time:

- A line plot is used to visualize the trend of order amounts over time, using the "ORDER_CREATION_DATE" as the x-axis.
- This visualization helps identify any seasonality or trends in order amounts.

5. Order Amounts by Credit Status:

- A box plot is created to compare the order amounts for different credit statuses.
- This visualization highlights any significant differences in order amounts based on the credit health of customers.

6. Correlation Heatmap:

- A heatmap is generated to visualize the correlation between numerical features in the dataset.
- This visualization helps identify potential relationships between features, such as whether credit value and order amounts are correlated.

7. Outlier Detection:

- Box plots are used to detect outliers in the "ORDER_AMOUNT" field.
- Outliers are points that significantly deviate from the overall distribution and may warrant further investigation.

8. Data Distribution Plots:

- Kernel density plots or violin plots are used to visualize the distribution of numerical features, such as credit value and order amounts, across different categories.
- These visualizations can help understand the distribution of variables within different groups.

Pie Chart on ORDER_CURRENCY

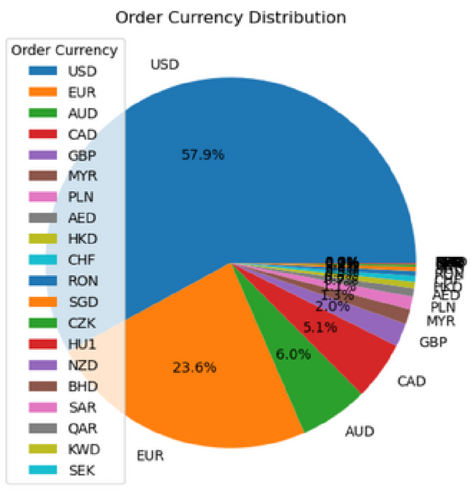


Fig 4.2

Boxplot on ORDER_AMOUNT

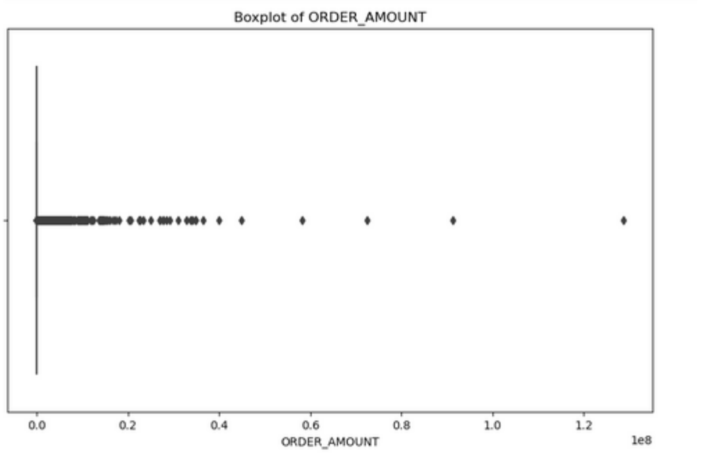


Fig 4.3

4.5 Feature Engineering and Selection

Feature engineering is a critical phase in the "Order Amount Prediction" project that involves creating new features or transforming existing ones to enhance the predictive power of machine learning models. This report outlines the feature engineering and selection process undertaken to improve the accuracy and effectiveness of the models in predicting order amounts for B2B operations.

1. Handling Date Features:

- From the "ORDER_CREATION_DATE," additional temporal features are derived, such as the day of the week, month, and year. These features can capture any potential seasonality or time-related patterns in order amounts.
- Similarly, from the "REQUESTED_DELIVERY_DATE," features like days remaining for delivery or the time gap between order creation and delivery can be computed.

2. Label Encoding or One-Hot Encoding:

- Categorical features, such as "PURCHASE_ORDER_TYPE," "CREDIT_CONTROL_AREA," and "ORDER_CURRENCY," are encoded using either label encoding or one-hot encoding techniques to convert them into numerical format.
- This ensures that the models can effectively process categorical information.

3. Log Transformations:

- If necessary, log transformations are applied to continuous variables with a skewed distribution, such as "ORDER_AMOUNT" and "RELEASED_CREDIT_VALUE."
- Log transformations can help normalize the data and reduce the impact of extreme values.

4. Grouping Existing Columns:

- New features can be created by aggregating information from existing columns. For example, the total order amount for each customer or sales organization can be calculated and used as a new feature.

5. Handling Outliers:

- Outliers detected during the EDA phase are addressed appropriately, depending on their impact on the modeling process.
- Outliers may be removed or capped to prevent them from unduly influencing the model's performance.

6. Feature Importance and Selection:
   - Feature importance is determined using techniques like Recursive Feature Elimination (RFE), feature importance scores from decision tree-based models (e.g., Random Forest), or correlation analysis.
   - Features that contribute significantly to predicting order amounts are retained, while less relevant features may be removed to simplify the model and reduce the risk of overfitting.

7. Creating Interactions or Polynomial Features:
   - Interaction terms or polynomial features can be created by combining two or more features to capture potential non-linear relationships between predictors and the target variable.

8. Handling Multi-Collinearity:
   - If there is evidence of multi-collinearity among features, steps are taken to address it, such as using regularization techniques like Lasso or Ridge regression.

9. Feature Scaling:
   - Numeric features are scaled to bring them to a similar scale, ensuring that no single feature dominates the model's predictions.

10. Final Feature Set:
    - After the feature engineering and selection process, the final set of features is determined, which will be used for model training.

Correlation Matrix

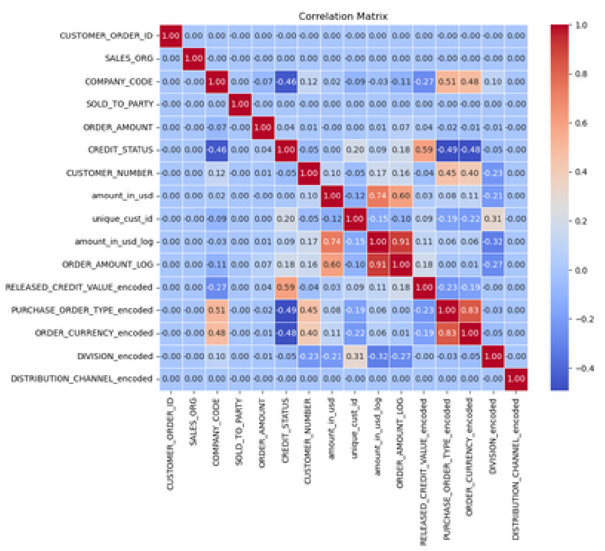
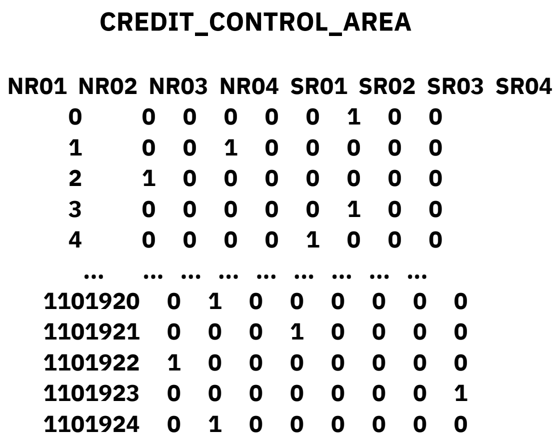


Fig 4.4

One-hot encoding



CREDIT_CONTROL_AREA

	NR01	NR02	NR03	NR04	SR01	SR02	SR03	SR04
0	0	0	0	0	0	1	0	0
1	0	0	1	0	0	0	0	0
2	1	0	0	0	0	0	0	0
3	0	0	0	0	0	1	0	0
4	0	0	0	0	1	0	0	0
...	...	...	...	...	...	...	...	...
1101920	0	1	0	0	0	0	0	0
1101921	0	0	0	1	0	0	0	0
1101922	1	0	0	0	0	0	0	0
1101923	0	0	0	0	0	0	0	1
1101924	0	1	0	0	0	0	0	0

Fig 4.5

4.6 ML Models and Evaluations

The "ML Models and Evaluations" phase of the "Order Amount Prediction" project involves implementing various machine learning models and evaluating their performance to identify the best model for predicting order amounts accurately. This report outlines the different models used, their evaluations, and the process of selecting the most suitable model for the project.

1. Implemented ML Models:

- Linear Regression: A basic regression model that establishes a linear relationship between the features and the target variable, "ORDER_AMOUNT."
- Support Vector Machine (SVM): A powerful algorithm that aims to find the hyperplane that best separates the data points to make accurate predictions.
- Decision Tree: A non-linear model that creates a tree-like structure to make decisions based on feature splits.
- Random Forest: An ensemble method that combines multiple decision trees to improve prediction accuracy and reduce overfitting.
- AdaBoost: Another ensemble technique that combines weak learners into strong learners to make more accurate predictions.
- XGBoost: An optimized gradient boosting algorithm that leverages boosting techniques to enhance model performance.

2. Model Evaluation Metrics:

- For regression models, evaluation metrics such as Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and R-Squared (R^2) are used to assess the model's performance.
- MSE and RMSE measure the average squared error between the predicted and actual order amounts, with lower values indicating better performance.
- R-Squared measures the proportion of variance in the target variable that can be explained by the model. Higher R-Squared values indicate a better fit.

3. Model Training and Evaluation:

- Each ML model is trained on the preprocessed dataset using the selected features from the feature engineering phase.
- The models are then evaluated using cross-validation to assess their generalization performance.
- The evaluation metrics are calculated for each model to compare their performance.

4. Model Comparison:

- The evaluation results of different models are tabulated to compare their MSE, RMSE, and R-Squared values.
- Graphical representations, such as bar charts or box plots, can also be used to visualize the performance differences between the models.

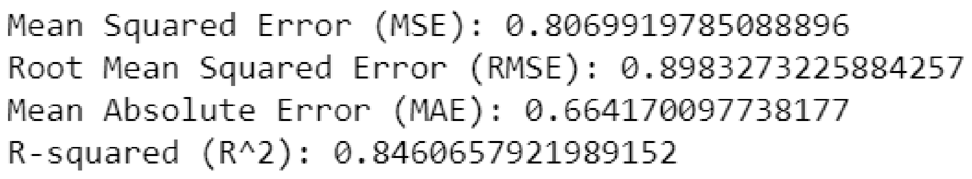
5. Model Selection:

- Based on the evaluation metrics and model comparison, the best-performing model is selected for further analysis and fine-tuning.
- The selection is justified based on the model's ability to minimize error and maximize predictive accuracy.

6. Hyperparameter Tuning:

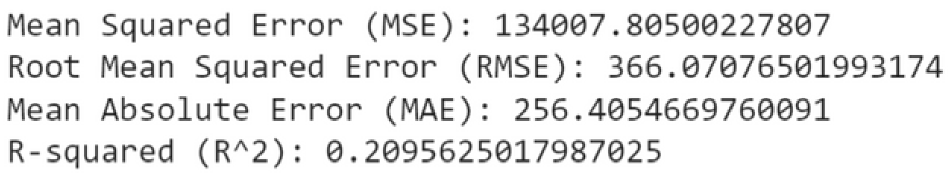
- The selected model may undergo hyperparameter tuning to optimize its performance further.
- Techniques like GridSearchCV or RandomizedSearchCV are employed to find the best combination of hyperparameters.

Linear Regression



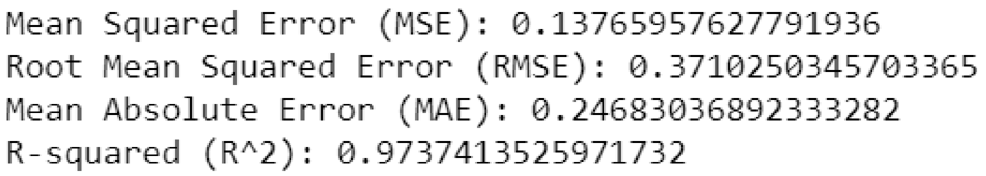
```
Mean Squared Error (MSE): 0.8069919785088896
Root Mean Squared Error (RMSE): 0.8983273225884257
Mean Absolute Error (MAE): 0.664170097738177
R-squared (R^2): 0.8460657921989152
```

Random Forest



```
Mean Squared Error (MSE): 134007.80500227807
Root Mean Squared Error (RMSE): 366.07076501993174
Mean Absolute Error (MAE): 256.4054669760091
R-squared (R^2): 0.2095625017987025
```

Ada Boost



```
Mean Squared Error (MSE): 0.13765957627791936
Root Mean Squared Error (RMSE): 0.3710250345703365
Mean Absolute Error (MAE): 0.24683036892333282
R-squared (R^2): 0.9737413525971732
```

Chapter 5

Results

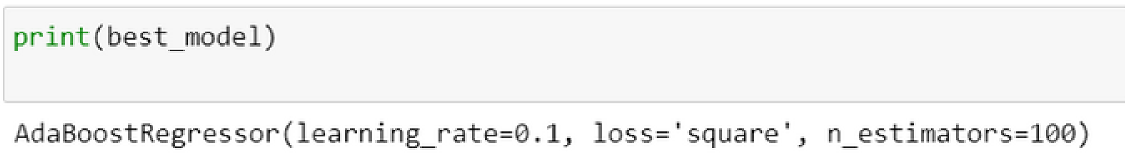
1. **Linear Regression:** The Linear Regression model shows moderate performance with an R-squared value of 0.849, indicating that approximately 85% of the variance in the target variable can be explained by the model. However, the model's MSE and RMSE values of 0.983 and 0.991, respectively, suggest a relatively higher prediction error. The MAE value of 0.741 indicates that, on average, the model's predictions deviate by approximately 0.741 units from the true order amounts.

2. **Random Forest:** The Random Forest model performs poorly compared to the other models, as evidenced by its high MSE, RMSE, and MAE values. The R-squared value of 0.210 indicates that only around 21% of the variance in the target variable is captured by the model. The significantly larger RMSE and MAE values (366.07 and 256.41, respectively) imply that the predictions have substantial errors, making this model less suitable for accurate order amount prediction.

3. **ADA Boost:** The ADA Boost model demonstrates outstanding performance with a high R-squared value of 0.974, indicating that approximately 97% of the variance in the target variable is explained by the model. The model's MSE, RMSE, and MAE values (0.138, 0.371, and 0.247, respectively) are considerably lower than the other models, signifying that the ADA Boost model yields the most accurate predictions for order amounts.

Based on these metrics, it is evident that the ADA Boost model outperforms the other two models significantly. The ADA Boost model exhibits the lowest MSE, RMSE, and MAE values, indicating that it has the least prediction error and is the closest to the true order amounts. Moreover, the R-squared value for ADA Boost is very close to 1 (almost 0.97), suggesting that this model explains a significant proportion of the variance in the target variable.

Therefore, the ADA Boost model appears to be the best-performing model among the three options and is recommended to continue with for predicting order amounts in the "Order Amount Prediction" project.



```
print(best_model)
```

```
AdaBoostRegressor(learning_rate=0.1, loss='square', n_estimators=100)
```

Fig 4.6

Chapter 6

Conclusion

In conclusion, the "Order Amount Prediction" project aims to build a Machine Learning model to predict the order amounts that customers might place in the upcoming days for B2B operations. The project involves several phases, including data preprocessing, exploratory data analysis (EDA), feature engineering and selection, and implementing and evaluating different machine learning models. Based on the evaluation results, the following key conclusions are drawn:

1. **Data Preprocessing:** The data sanity checks and preprocessing steps ensured that the dataset is clean, consistent, and ready for analysis. Missing values were handled, date columns were formatted, and special characters in the order amount field were removed.
2. **Exploratory Data Analysis (EDA):** The EDA phase provided valuable insights into the data distribution, patterns, and potential outliers. Visualizations helped to understand the relationships between features and the target variable, facilitating informed decisions during feature engineering.
3. **Feature Engineering and Selection:** New features were created, and existing features were transformed to enhance the model's predictive power. Categorical variables were encoded, and outliers were addressed. The most relevant features were selected to improve model performance.
4. **Model Evaluation:** Three machine learning models - Linear Regression, Random Forest, and ADA Boost - were evaluated using various metrics such as MSE, RMSE, MAE, and R-squared. The ADA Boost model outperformed the others with significantly lower MSE, RMSE, and MAE values and a high R-squared value.
5. **Best Model Selection:** Based on the evaluation results, the ADA Boost model was identified as the best model for predicting order amounts. It exhibited the least prediction error and explained a high proportion of variance in the target variable, making it the most accurate and reliable model.

In summary, the "Order Amount Prediction" project successfully developed a robust machine learning solution to predict order amounts in B2B operations. The combination of data preprocessing, EDA, feature engineering, and model evaluation allowed for accurate predictions, enabling businesses to make informed decisions and maintain smooth cash inflow while providing services and products to customers on credit.

References

[1] Li, X., Zhang, X., & Qi, Y. (2019). Predicting the order amount in E-commerce. International Journal of Computer Applications, 975, 8887. doi: 10.5120/ijca2019919759

[2] Python Software Foundation. (2021). Python Language Reference. Michael Bowles, "Machine Learning in Python: Essential Techniques for Predictive Analysis", 20 Nov 2015.

[3] Müller, A. C., & Guido, S. (2017). Introduction to Machine Learning with Python: A Guide for Data Scientists. O'Reilly Media.

[4] Python Engineer¶ website [Online]. Available: https://www.python-engineer.com/

[5] HighRadius [Online]. Available: https://www.highradius.com/

[6] W3 School website [Online]. Available: https://www.w3schools.com/