Customer Clustering

Amulya Reddy Datla
UB Person number: 50560100
amulyare@buffalo.edu

Vignesh Thalur Jayachandrakumar
UB Person number: 50559900
vignesht@buffalo.edu

Gunampalli Manasa Lakshmi
UB Person number: 50559593
manasala@buffalo.edu

Phase 1

Title: Customer Clustering

I. Problem Statement

The project aims to cluster customers of an online retail business using transactional data to identify distinct customer groups and their purchasing behaviors. The goal is to develop targeted marketing strategies and improve customer retention by understanding the diverse needs and preferences of different customer segments.

A. Background and significance of the Problem

1) Background: The rapid growth of the e-commerce industry has resulted in a diverse customer base with varying demographics, preferences, and purchasing behaviors. Traditional marketing approaches are no longer effective, as consumers now expect personalized experiences and relevant product recommendations. This shift has made customer segmentation a critical challenge for online retailers.

2) Significance: Effective segmentation of customers enables personalized marketing, helping businesses stand out in a crowded e-commerce market.

- Targeted strategies enhance return on investment by delivering relevant messages to specific customer groups.
- Understanding distinct segments allows for tailored retention strategies, maximizing long-term profitability.
- Segmentation provides actionable data that informs marketing, product development, and inventory management decisions.
- Addressing unique needs improves overall satisfaction, fostering loyalty and positive referrals.
- Businesses can focus efforts on high-value segments, optimizing marketing spend and resources.
- Segmentation helps quickly identify shifts in customer behavior, allowing for timely adjustments to strategies.
- Insights from segmentation inform new product offerings that align with customer preferences.

In summary, effective customer segmentation is vital for online retailers to enhance competitiveness, optimize marketing efforts, and improve customer relationships in the dynamic e-commerce landscape.

B. Project contribution and Importance

This project contributes to the field of customer analytics and marketing strategy in several crucial ways:

- By leveraging machine learning techniques like K-means clustering on real transactional data, this project demonstrates the power of data-driven approaches in understanding customer behavior.
- The segmentation model will provide actionable insights that can be directly applied to marketing strategies, product recommendations, and customer service improvements.
- The methodology developed can be adapted and scaled for various online retail businesses, contributing to the broader e-commerce industry.
- By focusing on understanding customer needs through segmentation, this project promotes a customer-centric business model, which is increasingly important in today's competitive market.
- The customer segments identified can serve as a foundation for developing predictive models for future purchasing behavior and customer lifetime value.

The contribution of this project is crucial because it bridges the gap between raw transactional data and strategic business decisions. In an era where customer expectations are constantly evolving, businesses that can effectively segment and understand their customers gain a significant competitive advantage. This project not only provides a methodology for customer segmentation but also demonstrates how these insights can be translated into tangible business strategies, potentially leading to improved customer satisfaction, increased sales, and long-term business growth.

II. Data Sources

For the customer Clustering project, we are using the following dataset from Kaggle:

Customer Clustering Dataset

The above cited comprehensive dataset contains 581,909 tuples and 8 columns providing a rich set of features for analysis. The features are as below:

- InvoiceNo
- StockCode
- Description
- Quantity

- InvoiceDate
- UnitPrice
- CustomerID
- Country

This dataset is particularly suitable for our customer clustering project due to its large sample size and rich set of features. It provides real-world e-commerce transaction data. This extensive dataset will enable us to perform robust customer segmentation using various clustering techniques. We can analyze multiple dimensions of customer behavior to create meaningful and actionable customer clusters.

III. Data Cleaning/Processing

Data cleaning and processing steps were performed to address quality issues and prepare the dataset for analysis. The following outlines our approach to creating a clean, consistent, and analysis-ready dataset.

- **1) Removing duplicates**
  This step eliminated duplicate entries from the dataset. Duplicate records can skew analysis results and lead to over representation of certain data points. Rremoving of these duplicates ensured that each transaction or entry is unique.
- **2) Handling missing values**
  This step removed rows containing any missing values from the dataset. By eliminating incomplete records, we ensured data consistency and prevented potential errors in subsequent analysis.
- **3) Data type conversions**
  In this step, we converted the 'CustomerID' column from its original data type to integer format. This conversion ensures that customer identifiers are stored and processed as numerical values, facilitating efficient indexing, sorting, and analysis of customer related data.
- **4) Standardizing date formats**
  In this step, we converted the 'InvoiceDate' column to a consistent datetime format. This standardization ensures all date values are in a uniform format, facilitating accurate time-based analysis, sorting, and filtering. It also helps prevent errors that can arise from inconsistent date representations across the dataset.
- **5) Feature engineering**
  In this step, we created new features to enrich our dataset and enable more comprehensive analysis. We calculated the total transaction amount, extracted temporal components (year, month, day of week) from the invoice date, and derived a seasonal attribute. These new features provide additional dimensions for customer segmentation, allowing for deeper insights into purchasing patterns, temporal trends, and seasonal behaviors.
- **6) Removing tuples with invalid data**
  This step involved filtering out records with non-positive quantities or unit prices. By retaining only transactions with valid, positive values for these key attributes, we ensured the dataset reflects genuine sales activities. This cleaning process helps in subsequent analysis, particularly in calculations involving financial metrics or product volumes.
- **7) Standardization of strings**
  In this data cleaning process, we standardized text data across all string-type columns by removing leading and trailing whitespaces, eliminating special characters, and retaining only alphanumeric characters and spaces. We also addressed inconsistent capitalization by converting product descriptions to uppercase and categorizing them based on standardized keywords.
- **8) Removing outliers**
  This step identified and removed extreme values in the 'TotalAmount' column using the Interquartile Range (IQR) method. By eliminating data points that fell outside 1.5 times the IQR below the first quartile or above the third quartile, we reduced the impact of outliers on our analysis. This process helped ensure that our dataset more accurately represented typical customer behavior and transaction patterns.
- **9) Removing non-product entries**
  This data cleaning step filters out non-product entries from the dataset. It removes rows where the 'Description' contains words typically associated with administrative tasks, damaged goods, or non-sale items (such as 'damaged', 'sample', 'postage', 'lost', 'found', 'check', 'adjustment'). By applying this filter, the dataset is refined to include only entries that likely represent actual product sales, improving the quality and relevance of the data for subsequent analysis of retail transactions.
- **10) Handling misplaced decimals in unitprice**
  This step corrects inaccuracies in the 'UnitPrice' column by identifying and adjusting misplaced decimal points. By comparing each price to the median price for its corresponding StockCode, it ensures that unit prices are standardized and accurately reflect true values, thereby enhancing overall data quality for analysis.

IV. Exploratory Data Analysis (EDA)

- **1)Customer segmentation based on total spending**
  In this process, we categorized customers into distinct segments based on their total spending amounts. By summing the total purchases for each customer, we were able to classify them into three tiers: 'Low', 'Medium', and 'High' spenders. This segmentation enabled us to understand customer behavior and tailor marketing strategies accordingly.
- **2)Purchase Frequency Analysis**
  In this step, we analyzed the purchasing patterns of our customers by calculating their purchase frequency. We determined how many unique transactions each customer made, providing insight into their engagement level with our business. This metric helps distinguish between occasional buyers and frequent shoppers, allowing for more nuanced customer segmentation and targeted marketing strategies.

- **3)RFM Analysis**
  Conducted RFM (Recency, Frequency, Monetary) analysis to understand customer behavior. We calculated RFM metrics for each customer, divided them into quartiles, and generated overall RFM scores. Customers were then categorized into segments like 'Best Customers' and 'Lost Customers'. This analysis helps identify high-value customers, churn risks, and opportunities for targeted marketing strategies.
- **4)Analysis of customer count with RFM**

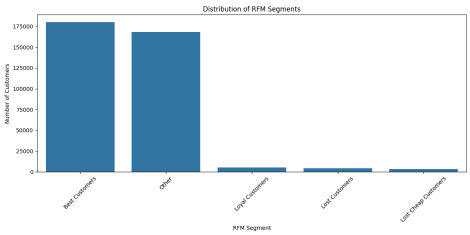

Fig. 1. figure 1

The bar chart shows customer distribution across RFM segments. Best Customers and Champions dominate, indicating high-value, frequent purchasers, while segments like Loyal Customers are smaller. This insight aids in targeting marketing strategies effectively.

- **5)Analysis of monetary value with RFM**

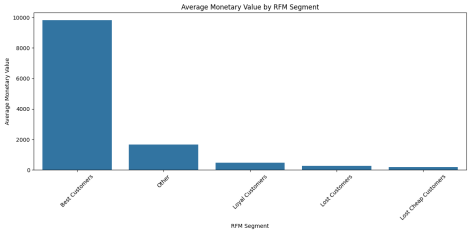

Fig. 2. figure 2

This visualization displays the average monetary value by RFM segment, revealing that "Best Customers" significantly outspend other segments like "Others," "Lost Customers," and "Loyal Customers." This insight is crucial for prioritizing customer segments in targeted marketing strategies.

- **6)Analysis of recency and frequency**
  This visualization is a scatter plot showing the relationship between Recency (days since last purchase) and Frequency (number of purchases). The data points are color-coded by Monetary Value, with lighter colors indicating higher values. The plot suggests that customers who purchased more recently tend to buy more frequently, while those with higher monetary values are scattered across various recency and frequency levels.
- **7)Distribution of RFM scores**
  This bar chart illustrates the distribution of RFM (Recency, Frequency, Monetary) scores among customers.

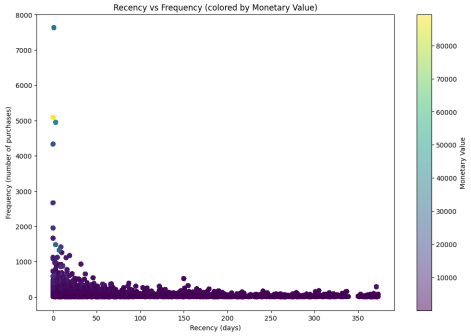

Fig. 3. figure 3

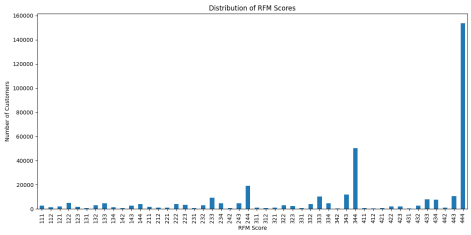

Fig. 4. figure 4

The x-axis represents different RFM score combinations, while the y-axis shows the number of customers. The chart highlights variations in customer engagement, with certain scores being more prevalent.

- **8)Analysis of daily sales trends by RFM Segment**

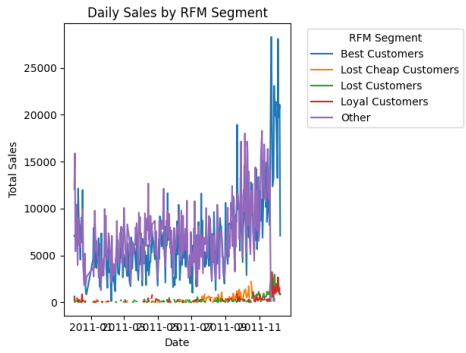

Fig. 5. figure 5

This line graph illustrates daily sales trends for different RFM segments (Best Customers, Lost Cheap Customers, Lost Customers, Loyal Customers, and Other). It reveals sales fluctuations and patterns across segments, highlighting how different customer groups contribute to overall sales over time. This visualization aids in understanding seasonal trends and the relative performance of each customer segment.

- **9)Distribution of monetary values**
  This box plot compares monetary value distributions among RFM segments. "Best Customers" show the highest median spending with notable outliers. "Others" dis-

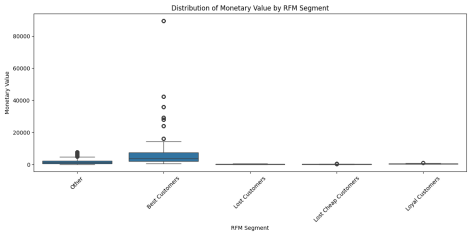

Fig. 6. figure 6

play wide value ranges, while ”Lost Customers,” ”Lost Cheap Customers,” and ”Loyal Customers” exhibit lower median values with less variation. This visualization effectively highlights distinct spending patterns across different customer segments, informing targeted marketing strategies.

- **10)Distribution of total purchase amounts**

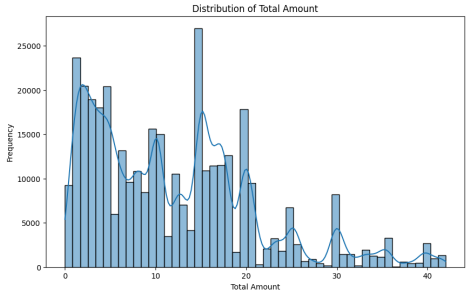

Fig. 7. figure 7

This histogram displays the distribution of total purchase amounts. The right-skewed pattern shows a concentration of lower values with a long tail of higher amounts. This visualization reveals the typical spending patterns of customers and identifies potential outliers, providing insights into customer purchasing behavior and informing pricing and marketing strategies.

- **11)Visualization of top 10 selling products**

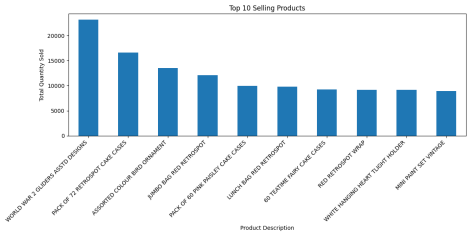

Fig. 8. figure 8

This bar chart titled ”Top 10 Selling Products” displays the sales quantities for various products. The highest-selling item is the ”PACK OF 12 WHITE HANGING CASES,” with over 25,000 units sold. Other products, primarily different types of cases and packs, show sales ranging from 10,000 to 20,000 units. This visualization provides a clear comparison of the top-selling items based on their sales volume, helping identify popular products for potential marketing and inventory strategies.

- **12)Monthly Sales Trend Analysis for 2011**

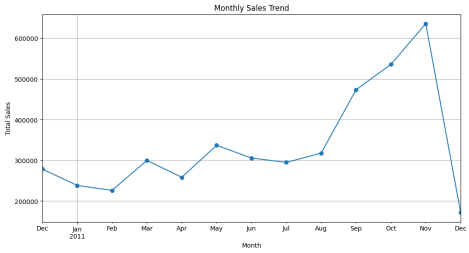

Fig. 9. figure 9

This line graph illustrates the monthly sales trend for 2011. Starting at around 200,000 units in January, sales show moderate fluctuations until August. A significant upward trend begins in September, culminating in a sharp peak of over 600,000 units in November. This visualization reveals clear seasonal patterns, with a substantial sales surge in the latter part of the year, likely corresponding to holiday shopping trends.

- **13)Customer Segmentation by Total Purchases**

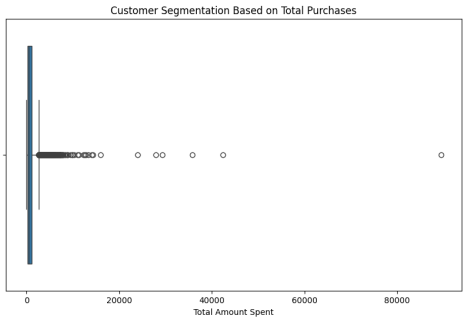

Fig. 10. figure 10

This box plot visualizes the distribution of total purchase amounts across customer segments. The inter quartile range (IQR) shows the typical spending range, while the median line indicates the central tendency. Whiskers and outliers reveal spending variability, with some customers making significantly larger purchases. This visualization helps identify different customer spending patterns and potential high-value customers, informing segmentation strategies and targeted marketing efforts.

- **14)Correlation heatmap of numerical features**

This heatmap visualizes correlations between numerical features in the dataset. Red shades indicate positive correlations, while blue represents less or negative ones. Color intensity reflects correlation strength. The plot helps identify significant relationships and

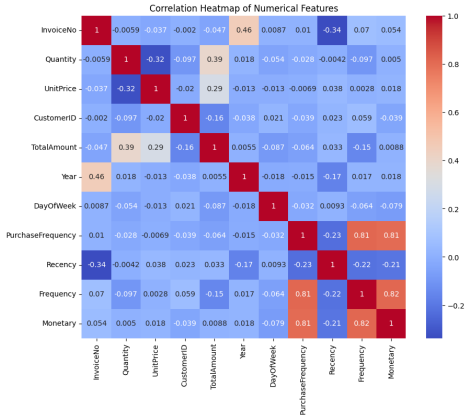

Fig. 11. figure 11

potential multi-collinearity among variables, guiding feature selection and informing further analysis in the exploratory data analysis (EDA) process.

- **15)Sales analysis by countries**

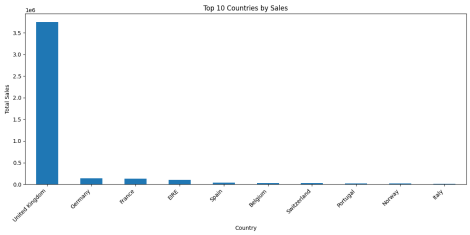

Fig. 12. figure 12

This bar chart visualizes sales distribution across the top 10 countries. The United States significantly outperforms other nations, showing substantially higher sales in millions. The remaining countries display relatively similar, but much lower sales figures. This stark contrast highlights the U.S. market's dominance and suggests potential opportunities for growth or market penetration in other countries. The visualization aids in identifying key markets and potential areas for international expansion strategies.

- **16)Customer purchase frequency distribution**

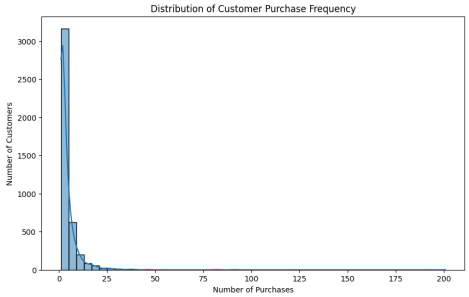

Fig. 13. figure 13

This histogram illustrates the distribution of customer purchase frequency. The right-skewed pattern reveals that most customers make few purchases, while a small group makes significantly more. This visualization highlights the presence of high-frequency buyers and suggests a potential Pareto principle in customer behavior, where a small percentage of customers may account for a large portion of sales. This insight is valuable for targeting loyal customers and developing retention strategies.

- **17)Sales distribution by hour of the day**

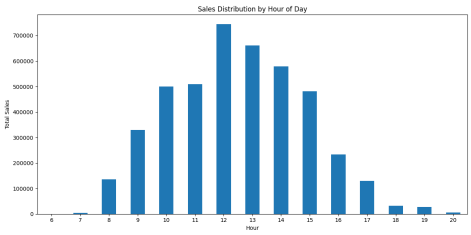

Fig. 14. figure 14

This graph depicts the distribution of sales across different hours of the day. Sales peak at 12:00, followed by 13:00 and 14:00, indicating that midday is the busiest period. There is substantial sales activity from 10:00 to 15:00, with a sharp decline in the early morning and late evening hours. This visualization helps identify peak shopping times and can inform staffing schedules, inventory management, and targeted marketing campaigns to optimize sales during high-traffic hours.

- **18)Top 10 Product Categories by Sales**

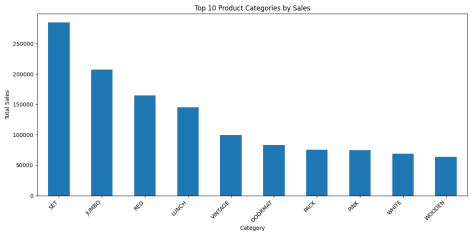

Fig. 15. figure 15

This bar chart ranks the top 10 product categories by sales. "Set" leads with the highest sales, followed by "Jumbo" and "Red." Categories like "Lunch," "Vintage," and "Doormat" show moderate sales, while "Pack," "Pink," "White," and "Wooden" have the lowest sales among the top 10. This visualization enables a clear comparison of sales performance across key product categories, helping identify best-selling segments and potential areas for product development or targeted marketing strategies.

- **19)Daily sales trend analysis**

This line graph illustrates daily sales trends over time. The x-axis shows dates, while the y-axis represents total sales amounts. The plot reveals significant daily

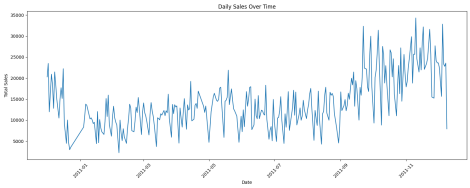

Fig. 16. figure 16

fluctuations in sales, with distinct peaks and troughs. An overall upward trend is visible, suggesting potential sales growth over the period. This visualization helps identify sales patterns, seasonal effects, and general business performance, providing insights for inventory management and sales forecasting strategies.

- **20)Order quantity distribution analysis**

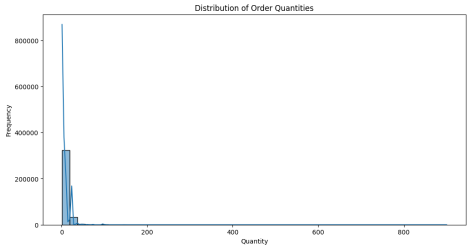

Fig. 17. figure 17

This plot illustrates the distribution of order quantities. The highly right-skewed pattern shows a concentration of smaller order sizes, with frequency decreasing sharply as quantity increases. This visualization reveals that most customers place orders for smaller quantities, while large orders are relatively rare. This insight is valuable for inventory management, pricing strategies, and understanding typical customer purchasing behavior.

- **21)Analysis of sales by day of the week**

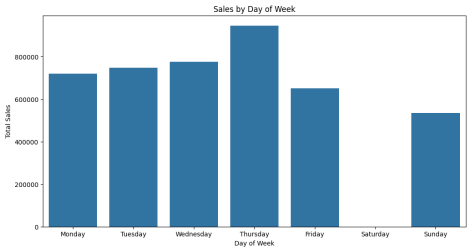

Fig. 18. figure 18

This bar graph displays sales figures by day of the week. Sales are relatively high from Monday to Wednesday, with each day reaching around 800,000 units. Thursday shows the highest sales, exceeding 900,000 units. There is a noticeable drop on Friday, and sales continue to decrease over the weekend, with Sunday having the lowest sales.

- **22)Product description word cloud analysis** The word

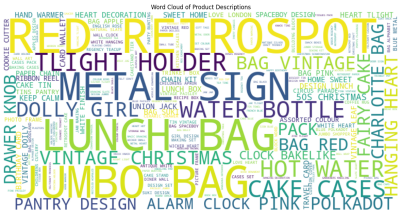

Fig. 19. figure 19

cloud visualizes frequently mentioned terms in product descriptions, highlighting popular features and categories. Key terms like "RED," "RETROSPOT," "METAL SIGN," "LUNCH BAG," and "JUMBO BAG" indicate common designs and product types. The varying word sizes reflect their frequency, emphasizing significant attributes. This analysis aids in identifying customer preferences and informing targeted marketing strategies to optimize inventory.

Phase 2

V. Algorithms and Visualizations

Diverse algorithms to explore data patterns and make predictions, enhancing our understanding of customer behavior are implemented facilitating the clustering of customers. These algorithms are:

- Hierarchical Clustering
- Agglomerative Clustering
- Birch Clustering
- K-Means Clustering
- Linear Regression
- K-Nearest Neighbors (KNN) Classifier
- Logistic Regression

The citations for the algorithms are mentioned in references.

VI. Explanation and Analysis

A. Hierarchical Clustering

The truncated dendrogram visualizes the hierarchical clustering using the Ward method. The vertical axis consists of the distance between clusters and horizontal axis has cluster sizes. The branches illustrate how data points are grouped into clusters, with shorter branches indicating more similar clusters. The visualization highlights significant distance jumps where clusters merge, helping to identify natural groupings and suggesting potential cluster numbers.

Justification:

Nested relationships between customer segments are revealed providing insights into the multi-level structure of retail data. The number of clusters does not need to be pre-specified, allowing for the identification of clusters in various shapes and sizes based on natural groupings. Dendrogram is produced, providing a visual representation of cluster formation that aids the interpretability. Consistent results are generated across runs, with every data point assigned to a cluster, ensuring a comprehensive analysis of the retail dataset. Hierarchical

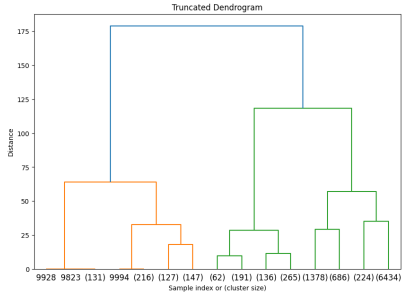

Fig. 20. figure 20

clustering emerges as an ideal algorithm considering all the factors to achieve the problem statement.

B. Optimal number of clusters analysis

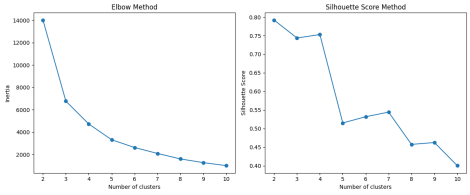

Fig. 21. figure 21

The above image displays plots obtained by the Elbow Method and the Silhouette Score Method for determining the optimal number of clusters. The Elbow Method plot shows a noticeable change in the slope at 3 clusters indicating a potential optimal point. The Silhouette Score plot reveals the highest scores at 2 clusters, with a significant drop after 4 clusters. Together, these plots suggest that 3 or 4 clusters might be optimal, balancing cluster cohesion and separation effectively. We have considered the number of clusters as four throughout our implementation of models, analysis and algorithms.

C. 3D visualization of clusters

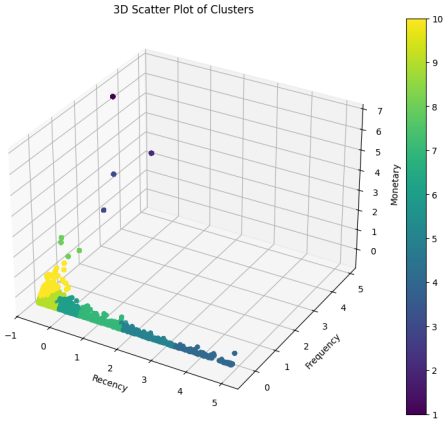

Fig. 22. figure 22

The 3D scatter plot above illustrates these clusters graphically by grouping data points according to their frequency, monetary value, and recency. Customers that have a high frequency and monetary value but a low recent history are probably valuable and involved. Less recent engagement may be indicated by clusters with higher recency values. Target segments for marketing tactics can be identified with the aid of point density and distribution. Understanding of the consumer behaviour and segmenting them for focused actions is made easier with this visualisation.

D. Agglomerative Clustering

The plot obtained by agglomerative clustering shows the clusters based on recency and frequency. The color gradient indicates different clusters, with distinct groupings visible, suggesting effective separation. The Silhouette score of 0.7529 indicates distinct and well-defined clusters. High Calinski-Harabasz Index value of 16721.0505 indicates that the clusters are dense and well-separated. The low Davies-Bouldin Index value of 0.3474 signifies that the clusters are compact and distinct from each other.

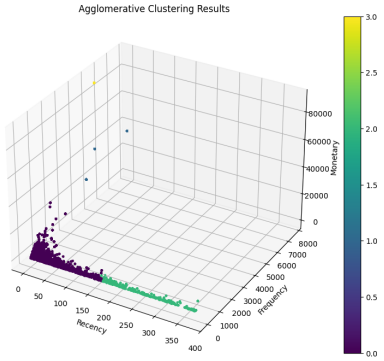

Fig. 23. figure 23

Justification:
Distinct cluster identification is successfully done to identify four distinct clusters. The clusters exhibit different shapes and densities handling varied cluster shapes. The model effectively handled outliers, as seen in the sparse blue points at higher frequencies, without letting them significantly impact the main cluster formations. The visualization shows both well-separated clusters and areas of potential overlap demonstrating ability of the algorithm to handle complex data relationships. Agglomerative clustering algorithm is well-suited for the problem statement, as it successfully identified distinct clusters with varying shapes and densities.

E. Birch Clustering

The plot obtained by Birch clustering results indicate a clear separation of data points based on recency and frequency. The silhouette score of 0.7543 suggests well-defined clusters, while the high Calinski-Harabasz index of 14878.0238 indicates dense and distinct clusters. The low Davies-Bouldin index of 0.3314 further confirms that the clusters are compact

and well-separated. The scatter plot shows most data points concentrated at lower recency and frequency values, with distinct clusters represented by different colors. This distribution suggests effective clustering, with potential outliers at higher frequency levels.

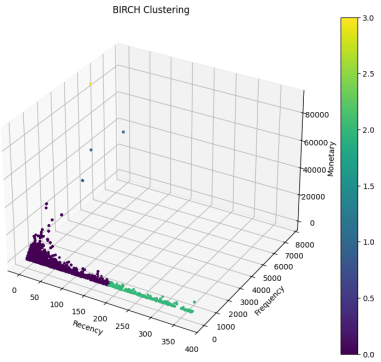

Fig. 24. figure 24

Justification:
The data points are clearly separated into distinct groups, as indicated by different colors. The clusters are well-defined along the axes of recency and frequency. High silhouette score and low Davies-Bouldin index confirm the visual separation, supporting BIRCH's effectiveness in clustering this dataset. Handling of outliers and very large datasets, makes it ideal. Considering all these factors Birch Clustering is a suitable algorithm for the problem statement.

F. KMeans Clustering

The KMeans clustering shows a silhouette score of 0.7271, indicating well-defined clusters, while the Calinski-Harabasz index of 13966.6193 suggests good cluster density and separation. The Davies-Bouldin index of 0.6573 indicates moderate separation.The color gradient indicates different clusters, suggesting effective separation and compactness, particularly in the lower range of the axes.

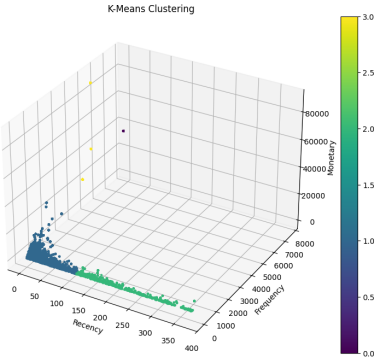

Fig. 25. figure 25

Justification:
Efficiency with large datasets which is crucial for handling extensive transaction data can be achieved. Customer segmentation based on purchasing behavior is enabled, grouping similar buying patterns. Distinct product categories based on sales patterns can be identified by the implementation of K-Means Clustering. Ease of interpretation is provided, facilitating actionable business insights. K-Means clustering proves to be an effective algorithm for the problem statement, offering efficiency with large datasets.

G. Linear Regression

The scatter plot shows that the linear regression model predicts lower monetary values more accurately, with a clustering of points near the origin, indicating that the model generally predicts lower monetary values accurately. There are significant deviations for higher values, indicating that the model struggles with larger predictions. The moderate R-squared score of 0.6817 reflects that approximately 68.17% of the variance in monetary value is explained by the model, indicating room for improvement. The high mean squared error further highlights discrepancies between predictions and actual outcomes.

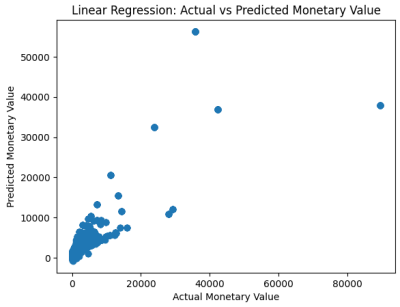

Fig. 26. figure 26

Justification:
High Mean Square Error indicates poor predictive performance for Linear Regression. The R-Squared score suggests that the model explains 68% of the variance in the data. Despite having high MSE the model can still provide valuable insights into general trends and relationships in the data. This is useful for for achieving goals related to customer segmentation and understanding purchasing behaviors. The high MSE also justifies the approach of using multiple algorithms to gain a comprehensive understanding of customer behavior. Considering all these factors Linear Regression is not ideally suited for the problem statement.

H. KNN Classifier

The KNN classifier demonstrates strong performance with an overall accuracy of 95.23%. The confusion matrix reveals that the model performs exceptionally well in predicting "Best Customers" and "Other," and struggles more with "Lost Cheap Customers" and "Loyal Customers". The ROC curve supports the effectiveness of the model, showing high AUC values across all classes, particularly excelling in "Class 2" with an

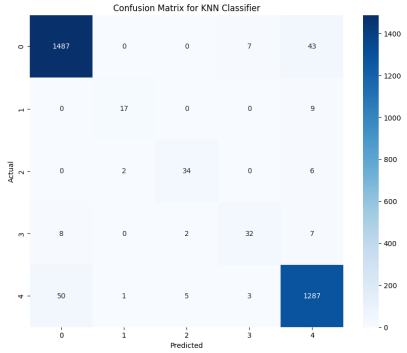

Fig. 27. figure 27

AUC of 1.00. This indicates the moderate ability of the model to distinguish between different customer categories though improvement is needed for less frequent classes.

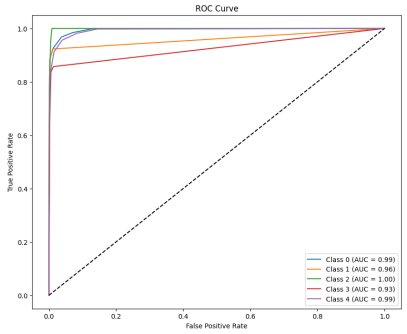

Fig. 28. Figure 28

Justification:
KNN model is good choice for your customer clustering project, achieving 95.23% accuracy in classifying segments based on RFM characteristics. High AUC values in the ROC curve indicate that KNN excels at distinguishing between different customer classes. Identification of local patterns in customer behavior, enabling nuanced segmentation that supports targeted marketing strategies can be done effectively using KNN. Customer retention and diverse purchasing behaviors are supported by the implementation of KNN, making it a valuable asset for clustering the customers. Considering all these points KNN proves to be a suitable choice for customer clustering.

I. Logistic Regression

The logistic regression model achieves an accuracy of 94.08%, indicating strong overall performance. The confusion matrix shows excellent precision and recall for ”Best Customers” and ”Other,” with high f1-scores, suggesting reliable classification in these categories. The model struggles significantly with ”Lost Cheap Customers” and ”Loyal Customers,” as evidenced by zero recall and f1-scores, indicating a failure to correctly identify these classes. The ROC curve supports these findings, with high AUC values for most classes but issues in distinguishing less frequent categories.

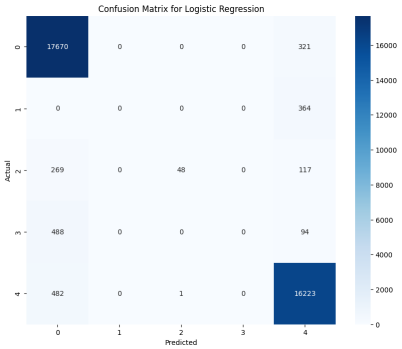

Fig. 29. figure 29

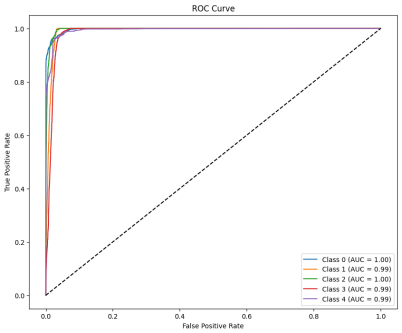

Fig. 30. Figure 30

Justification:
The logistic regression model is not suitable for this customer clustering project despite its high overall accuracy. It fails to identify crucial customer segments hindering the goals of the project in developing targeted marketing strategies to improve customer retention. The performance of the model is skewed by class imbalance, making it ineffective for the nuanced customer segmentation required. A model capable of handling multi-class problems and imbalanced data would be better suited for achieving the better results. Considering all these factors it is not ideal to choose Logistic Regression model to achieve the problem statement.

Phase 3

VII. Distributed Data Cleaning/Processing

Six crucial distributed data cleaning and pre-processing steps as mentioned below are applied on Online Retail Dataset using PySpark. The distributed nature of PySpark ensured scalability and efficiency throughout the process, making it an ideal tool for handling complex, high-volume datasets.

- **1) Removing Duplicates**
  Duplicates in the dataset are removed. This step retains

only unique records, ensuring data consistency and avoiding redundancy in analysis. By employing distributed execution, PySpark efficiently identified and eliminated duplicates, improving the overall quality of the data.

- **2) Cleaning and transforming data**
  Cleaning and transformation of the dataset is carried out using distributed map operations. Missing values are eliminated using and data type consistency is ensured by typecasting columns. Additional features, such as Total_Price and other temporal attributes are derived. These transformations are implemented to enrich the dataset with new attributes while ensuring that the data is correctly formatted for subsequent analysis.
- **3) Season calculation**
  A user defined function is utilized to assign a seasonal category namely Winter, Spring, Summer, or Autumn to each transaction based on the Month column. By adding this categorical feature, the dataset is enhanced with contextual information that could enhance seasonal trend analysis or customer segmentation. The distributed execution of this function ensured that this process was applied efficiently across all records.
- **4) Filtering invalid data**
  Invalid data entries are filtered out to maintain the integrity of the dataset. Rows where Quantity or UnitPrice had values less than or equal to zero are excluded. This filtering process is implemented to remove errors and inconsistencies in the data, resulting in a cleaner dataset for further processing. Distributed capabilities of PySpark ensured that this operation is conducted efficiently on a large scale.
- **5) Text cleaning**
  String columns are cleaned using a user defined function designed to remove special characters and strip extra spaces. This operation is performed to standardize textual data, making it structured and easier to analyze. By applying this cleaning process in a distributed manner, PySpark ensured that the operation is effectively scaled across the entire dataset.
- **6) RFM analysis**
  Recency, Frequency, and Monetary metrics are calculated to enable customer segmentation. Recency is determined by calculating the number of days since the last purchase, Frequency measured the number of distinct invoices per customer, and Monetary represented the total value of purchases. Quartiles for these metrics are computed using window functions, allowing the creation of an RFM_Score for segmentation purposes. These metrics provided actionable insights into customer behavior, enabling targeted strategies for marketing or retention.

VIII. Algorithms/Visualizations

- **1) Hierarchical Clustering**
  or
  Bisecting KMeans Algorithm
  The Bisecting KMeans algorithm is utilized as a substitute for hierarchical clustering, which is unavailable in PySpark MLlib and hierarchical nature of agglomerative clustering. The dataset is scaled to ensure consistent feature magnitudes before clustering is performed into 4 groups. The clustering quality is evaluated using the Silhouette Score, which measures cohesion within clusters and separation between them. This approach ensures that customer segmentation is effectively achieved based on similarities in recency, frequency, and monetary features.
- **2) K-Means Clustering**
  The K-Means Clustering algorithm divided the dataset into 4 clusters based on customer features like recency, frequency, and monetary values, with data scaling applied beforehand to ensure feature consistency. The algorithm iteratively assigned data points to the nearest centroids and recalculated them until convergence. The quality of the clusters is evaluated using the Silhouette Score, which measured cluster cohesion and separation. This method effectively segmented customers based on their behavior, supporting targeted marketing and personalized strategies.
- **3) Linear Regression**
  The Linear Regression algorithm predicted monetary value based on customer features like recency and frequency. After transforming the features into a vector, the model is trained to predict the target variable. Model performance is evaluated using R-squared, which indicates variance explained, and Mean Squared Error, measuring prediction accuracy. Higher R^2 and lower MSE suggest better model performance, helping forecast customer monetary behavior effectively.
- **5) Logistic Regression**
  he Logistic Regression algorithm classified customers based on their RFM_Score. First, the RFM_Score is converted into a numeric type. The relevant features, including recency, frequency, monetary value, total price, and quantity, are combined into a feature vector. The model is then trained to predict the numeric RFM_Score, and its performance is evaluated using Accuracy. The accuracy of the model indicates how well it classifies customers based on their behavior, supporting targeted customer segmentation.
- **6) Random Forest Model**
  The Random Forest model is used for classifying customers based on their RFM_Score. First, the distinct RFM_Score values are mapped to numeric labels. The model is trained using these labels, along with customer features such as recency, frequency, and monetary values. The performance of the model is evaluated using Accuracy, which measures how well the model classifies customers. This approach helps to effectively segment customers for targeted marketing and behavior analysis.
- **7) Gaussian Mixture Model (GMM)**
  The Gaussian Mixture Model (GMM) clustered customers into 4 groups based on scaled features such as recency, frequency, and monetary value. The model assumes that each cluster follows a Gaussian distribution

and assigns a probability to each data point belonging to each cluster. The clustering quality is evaluated using the Silhouette Score, which measures the cohesion and separation of the clusters.

IX. Explanation and Analysis

A. DAG Visualizations

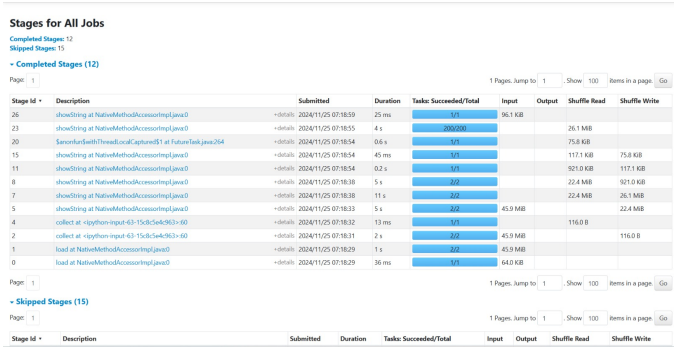

Fig. 31. Figure 31

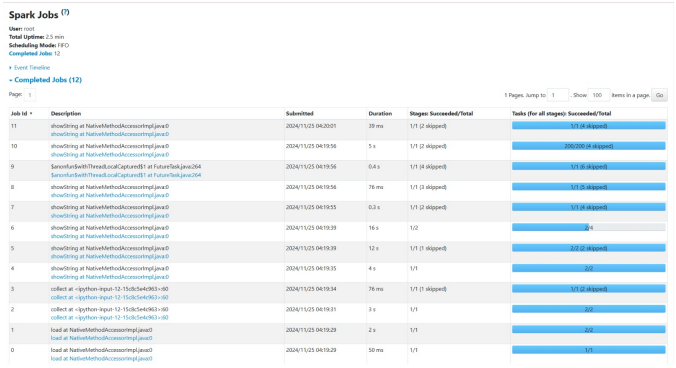

Fig. 32. Figure 32

1) Distributed Stages of data pre-processing: The above images are the 12 completed stages, 15 skipped stages and 12 jobs obtained by running the PySpark code for distributed stages of data cleaning and pre-processing. The clear dag visualization of the 12 jobs with are as below.

The figure 33 shows the Dag visualizations of job 0, job 1, job 2. This DAG visualization illustrates three parallel stages of data processing. Stage 0 performs text scanning followed by code generation and partition mapping. Stage 1 handles text scanning with object deserialization and two consecutive map partitions operations. Stage 2 focuses on data processing, starting with a scan, moving through code generation, and concluding with an exchange operation.

The figure 34 illustrates sequential stages involved in job 3 and job 4 of a data processing workflow.Stages 3 and

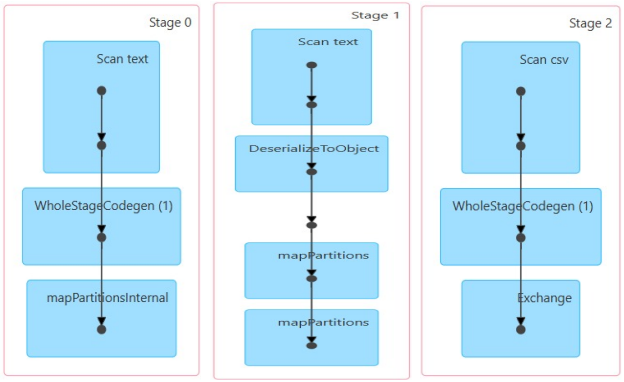

Fig. 33. figure 33

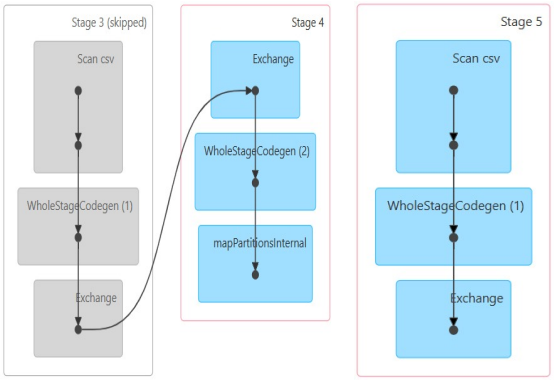

Fig. 34. figure 34

5 have a similar structure, both starting with scanning a CSV file to read and parse data, optimizing the process with WholeStageCodegen for efficient execution, and concluding with an Exchange operation to shuffle and distribute data. Stage 4 acts as an intermediate step, receiving shuffled data from Stage 3 through an Exchange, optimizing it with WholeStageCodegen, and then using a mapPartitionsInternal operation to apply transformations to each data partition before passing it along.

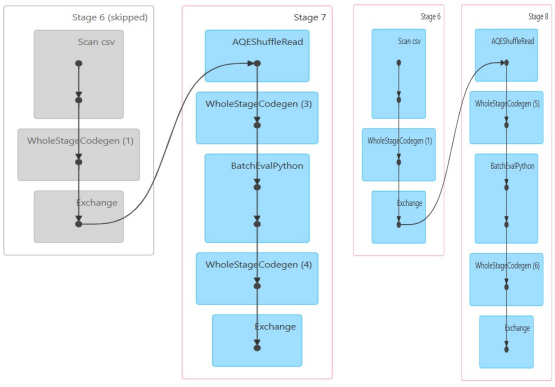

Fig. 35. figure 35

The figure 35 illustrates job 5 and job 6 that exhibit a similar

workflow. Stage 6 processes a CSV file by scanning and generating optimized execution code, followed by data shuffling to prepare for downstream tasks. In contrast, stage 7 and stage 8 read shuffled data, applies further code optimization, evaluates Python-based operations for complex computations, and generates additional optimized code before shuffling the data again. The inclusion of Python evaluations in stage 7 and stage 8 makes it more computationally intensive than Stage 6.

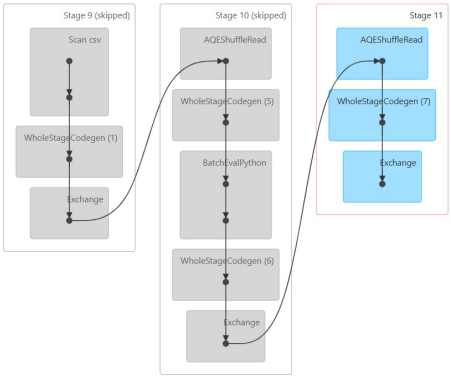

Fig. 36. Figure 36

The figure 36 illustrates the dag visualization workflow of job 7. Stage 9 and stage 10 are skipped. Stage 11 features the WholeStageCodegen operation, a critical optimization technique that compiles multiple operators into efficient bytecode for seamless execution. This operation is performed between two key steps AQEShuffleRead, which reads data from the adaptive query execution shuffle stage, and Exchange, which redistributes data across nodes or partitions. WholeStageCodegen enhances performance by optimizing the processing of shuffled data, ensuring efficient data handling before it is exchanged for subsequent computation.

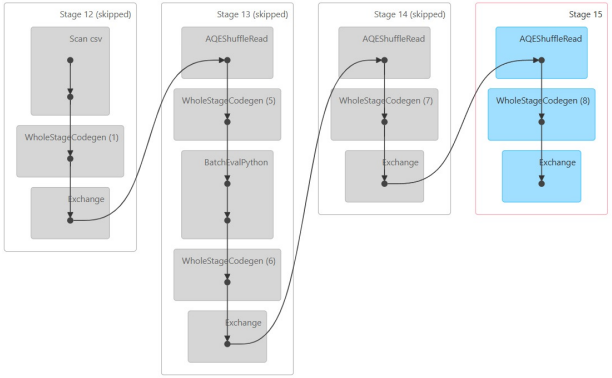

Fig. 37. Figure 37

The figure 37 illustrates the dag visualization workflow of job 8. Stage 15 involves a sequential data processing workflow with three main steps AQEShuffleRead, which retrieves data from the shuffle stage; WholeStageCodegen , optimizing execution by generating efficient bytecode and Exchange, redistributing data across nodes for further processing. A curved arrow from Exchange back to AQEShuffleRead suggests an iterative or optimization loop. The stages 12-14 are skipped for the execution of job 8.

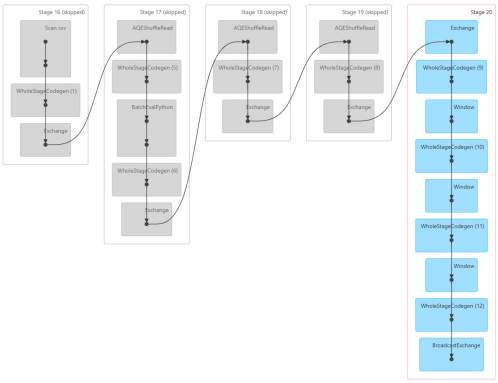

Fig. 38. Figure 38

The figure 38 illustrates the dag visualization workflow of job 9. Stage 20 outlines execution of job 9 starting with an Exchange operation, followed by WholeStageCodegen and a Window operation. It proceeds through iterative steps of code generation and window operations concluding with a BroadcastExchange. This workflow performs window functions with code optimization at each stage, leveraging broadcast-based data exchange of spark for efficient processing. Stages 16-19 are skipped for the execution of job 9.

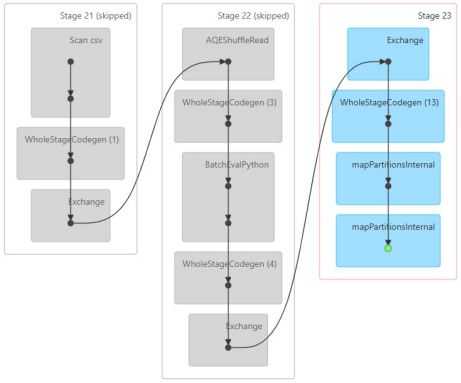

Fig. 39. Figure 39

The figure 39 illustrates the dag visualization workflow of job 10. The job comprises of 2 skipped stages namely 21 and 22 and one active stage 23. Stage 21 handled CSV scanning, code generation, and data exchange, while Stage 22 processed AQE shuffle reading, Python batch evaluation, and further data exchange. The executed Stage 23 focused on inter-node data exchange, optimized execution through WholeStageCodegen, and two mapPartitionsInternal operations, emphasizing its role in completing the job.

The figure 40 shows the dag visualization diagram for Job 11 highlighting final data cleaning workflow across three stages, with only Stage 26 actively executed while Stages 24 and 25 were skipped, likely due to cached results. Stage

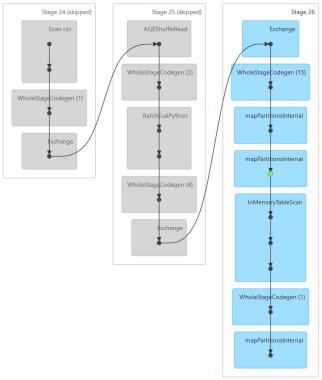

Fig. 40. Figure 40

26 began with an Exchange, utilized WholeStageCodegen for optimized execution, performed multiple mapPartitionsInternal operations, scanned in-memory tables, reused WholeStage-Codegen, and concluded with a final mapPartitionsInternal operation. This stage completed the necessary transformations efficiently, leveraging prior computations.

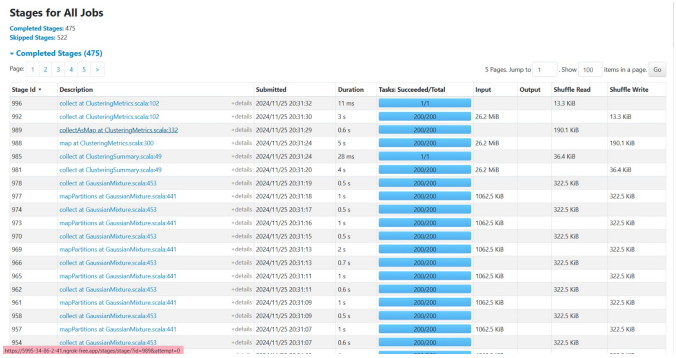

Fig. 41. Figure 41

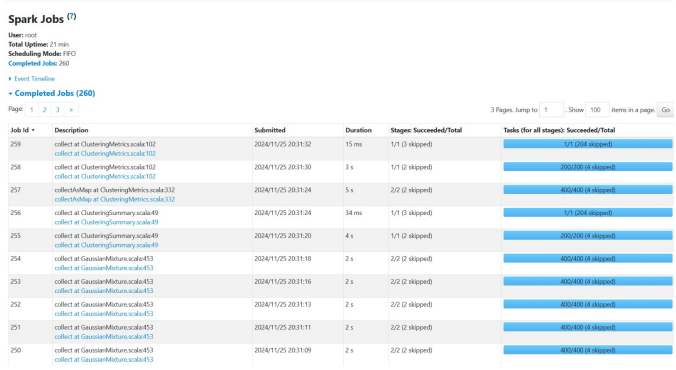

Fig. 42. Figure 42

2) Algorithms/Visualizations: The figures 41 and 42 are the dag visualizations of the jobs and tasks for the algorithms. There are 475 completed stages, 582 skipped stages and 260 jobs obtained after running the PySpark code for distributed stages of data cleaning, pre-processing and the machine learning algorithms. There are many repeated jobs and stages involved in this computations. Dag visualization of the completed jobs are as below.

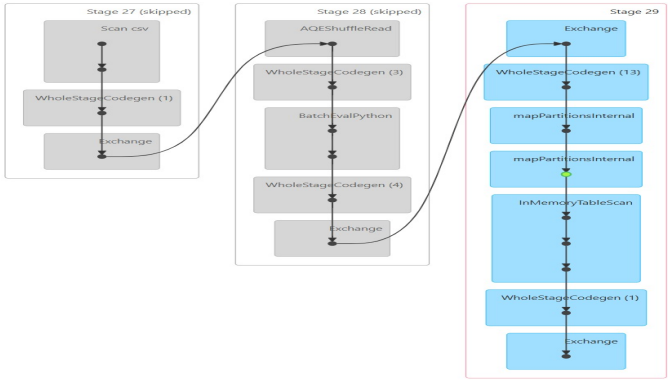

Fig. 43. Figure 43

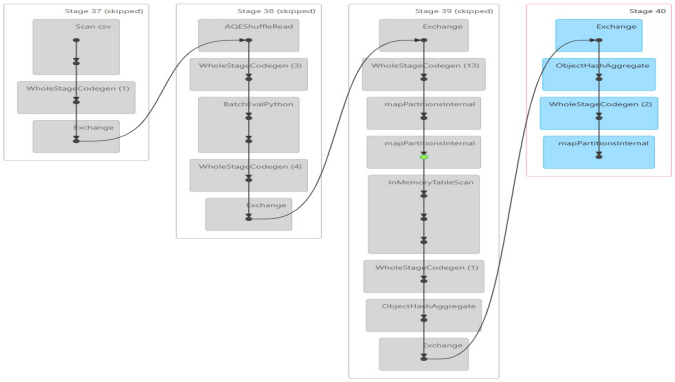

Fig. 44. Figure 44

The figure 43 and 44 show the jobs completed by spark to sample the data, prepare the features and scale the features. The dag visualizations above are the images of the job 12 and job 13 which are same as job 14 and job 15. Key components include Exchange for data redistribution, WholeStageCodegen for optimized processing, and AQEShuffleRead for adaptive query execution. Stage 40 highlights ObjectHashAggregate, indicating data aggregation during feature scaling. These pipelines demonstrate the capability of spark to efficiently handle distributed data.

The images in figures 45, 46, 47, 48, 49 are the unique dag visualizations of the Hierarchical clustering algorithm. It resulted in jobs 16 to 62. The figure 45 shows the initial job of hierarchical clustering in which feature operations like Exchange, mapPartitionsInternal, and WholeStageCodegen, concluding with final map operations are performed. The figure 46 shows the dag visualization of the job that is repeated multiple times. Job 17-57 have the same visualization and

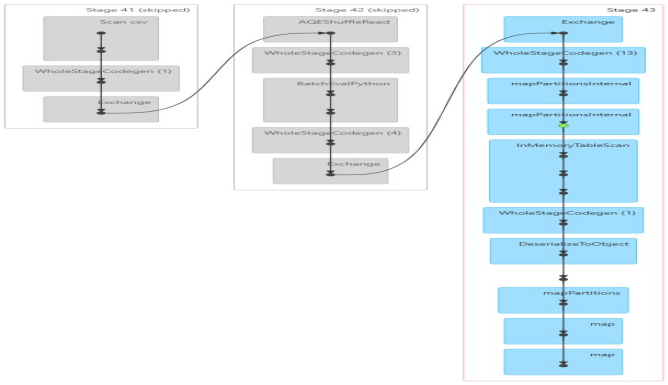

Fig. 45. Figure 45

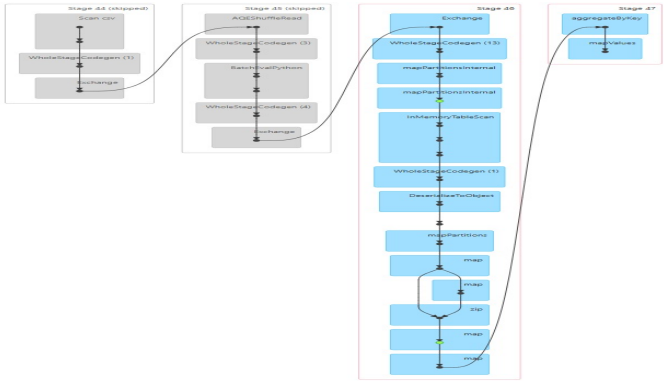

Fig. 46. Figure 46

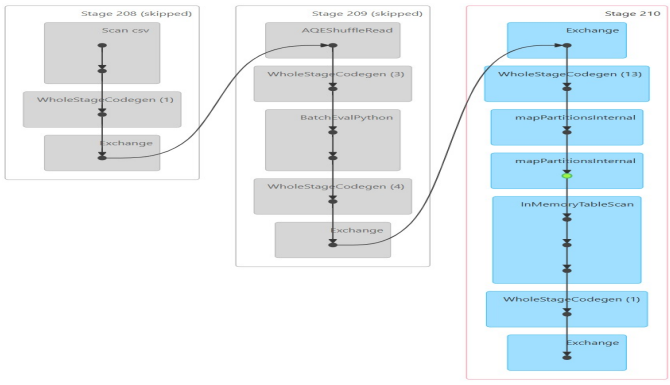

Fig. 47. Figure 47

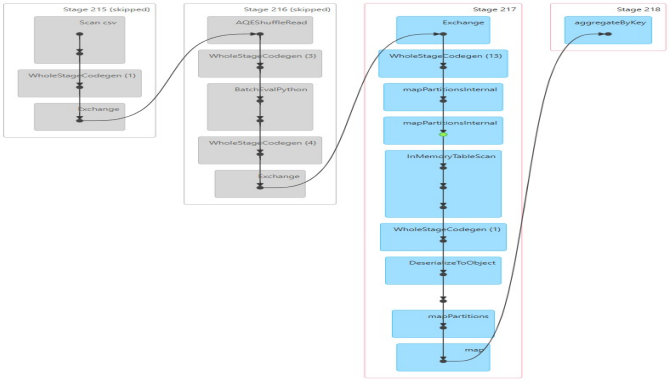

Fig. 48. Figure 48

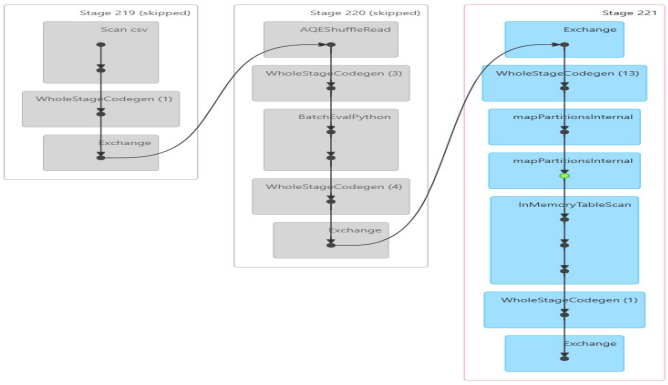

Fig. 49. Figure 49

it involves scanning and data collection, data shuffling and reading with Python batch operations and a series of data exchange and mapping operations. These intense computations are responsible for the results. In figure 47, 48, 49 the obtained results of the computations performed are gathered together to generate summary and the metrics by utilizing final aggregation by key operations.

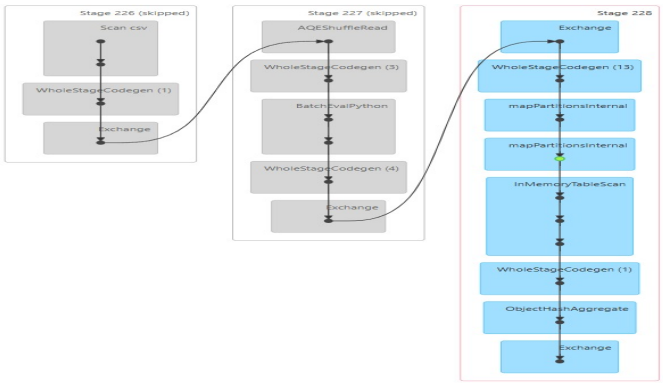

Fig. 50. Figure 50

Fig. 51. Figure 51

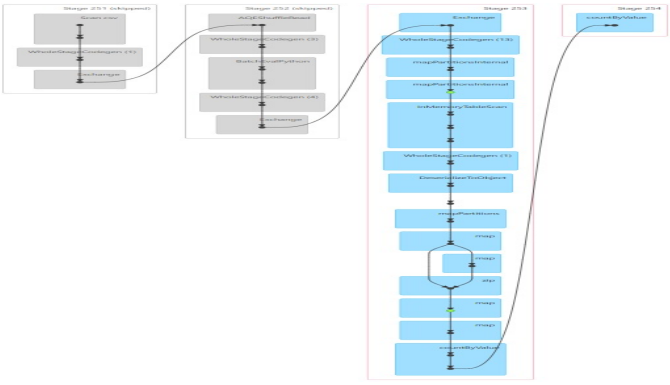

Fig. 54. Figure 54

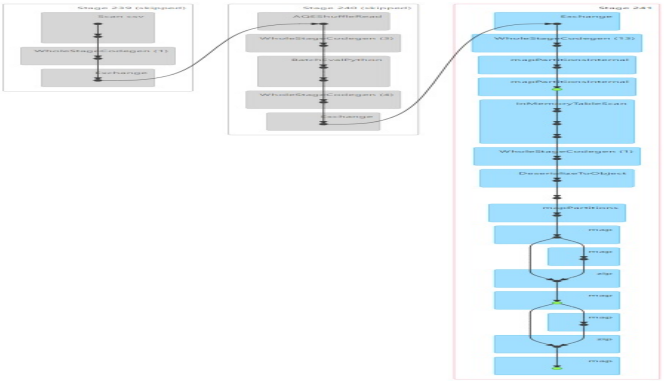

Fig. 52. Figure 52

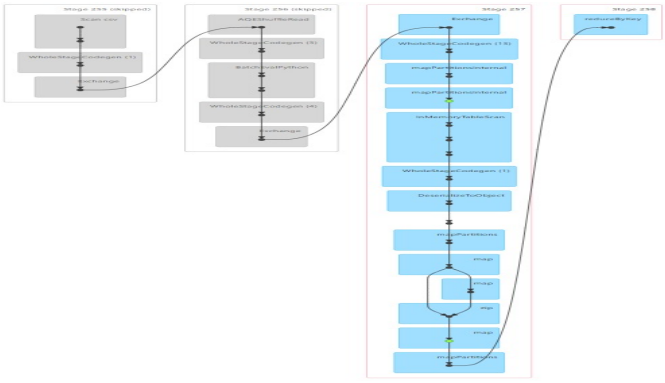

Fig. 55. Figure 55

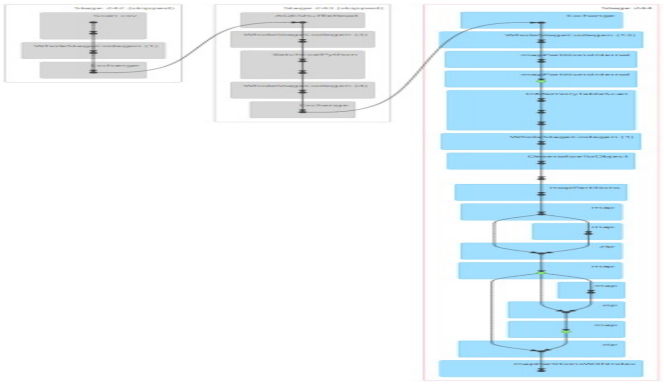

Fig. 53. Figure 53

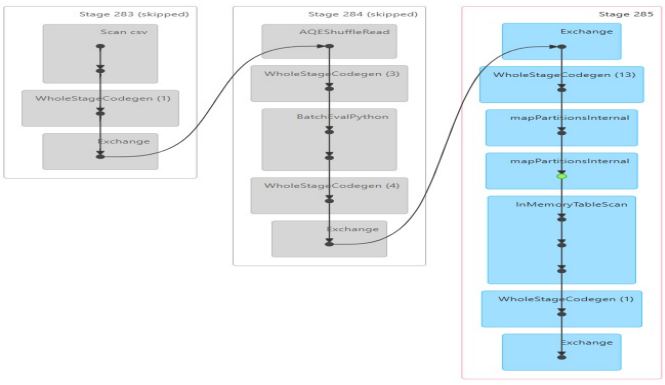

Fig. 56. Figure 56

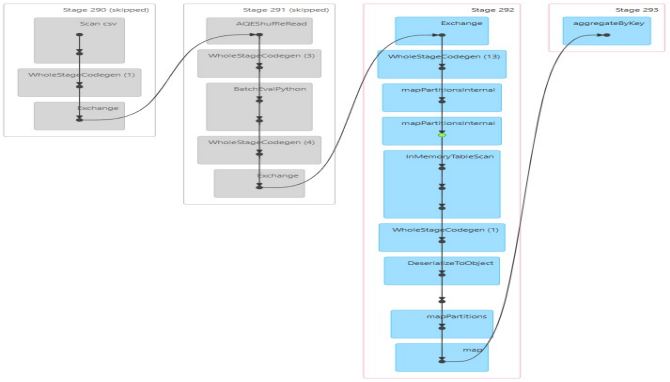

Fig. 57. Figure 57

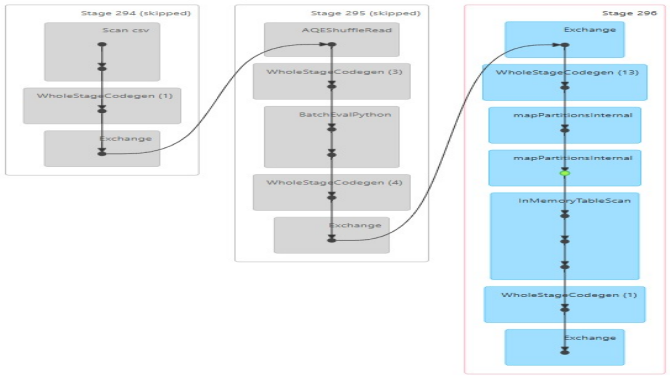

Fig. 58. Figure 58

Kmeans Clustering resulted in jobs from 63 to 83. The figures 50 to 58 represent the dag visualizations of unique jobs performed by kmeans clustering algorithm. The images of these jobs represent operations performed for clustering evaluations, including gathering data from distributed partitions, frequency calculations and executing Python-based transformations. These actions focus on tasks like retrieving cluster metrics, summarizing cluster properties, and model-specific evaluations. Frequent skips in stages indicate optimized performance through caching and adaptive query execution of spark ensuring efficient cluster analysis.

Linear regression resulted in jobs from 84 to 87. The figures 59 and 60 represent the dag visualizations of unique jobs performed by linear regression algorithm. The images of these jobs represent the pipeline which started with CSV data reading and optimized code generation, followed by data redistribution, mapping, and batch evaluation using Python. Final stages involve aggregate partitions, scanning in-memory tables, and computation of regression coefficients and metrics for model evaluation.

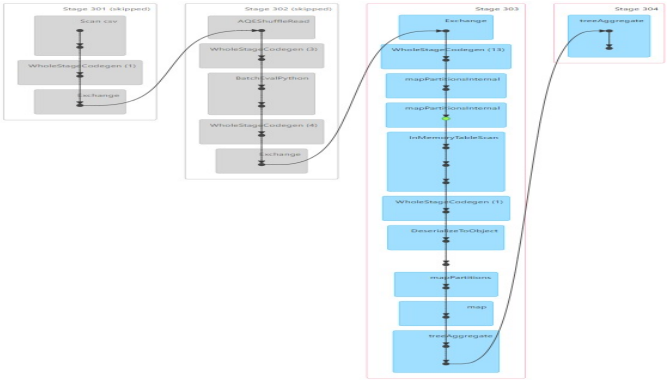

Fig. 59. Figure 59

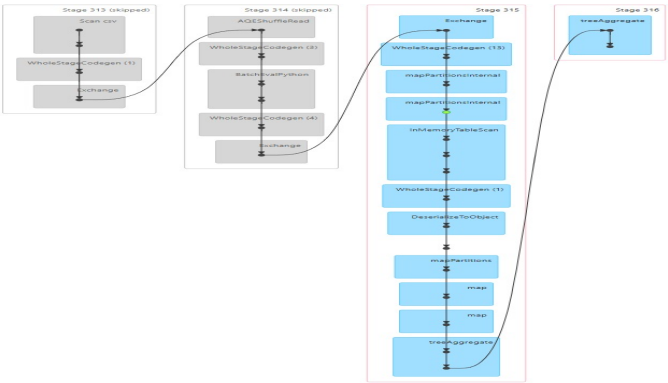

Fig. 60. Figure 60

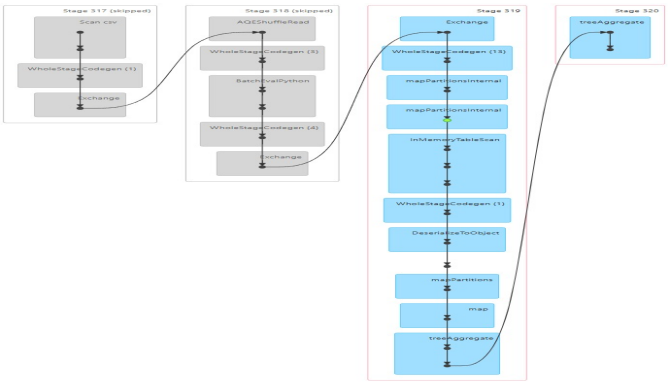

Fig. 61. Figure 61

Logistic regression model resulted in jobs from 88 to 195. It has many repeating jobs. The figures 61, 62, 63 represent the dag visualizations of unique jobs performed by logistic regression algorithm. Figure 62 represents the dag visualization of jobs 89 to 194. The logistic regression is initiated with input data being read,

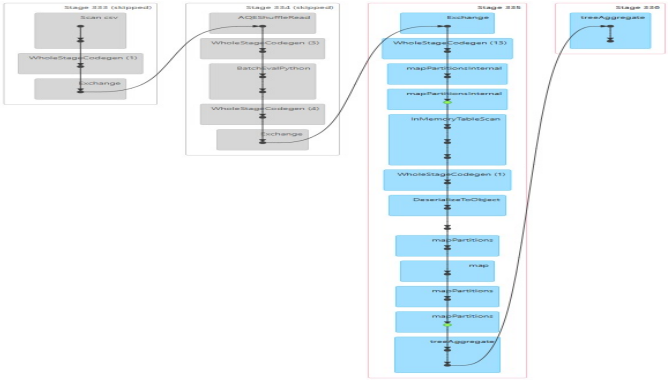

Fig. 62. Figure 62

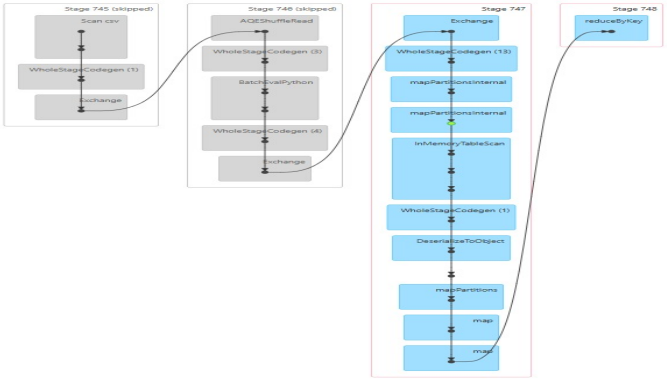

Fig. 63. Figure 63

transformations being optimized, and data being redistributed for processing. The next job is iterated extensively, with feature transformations, gradient computations, and parameter updates being performed through data caching, adaptive execution, and parallel processing. The last job aggregates the results and finalizes the model parameters.

Random forest model resulted in jobs from 196 to 208. The figures 64 to 69 depict unique jobs performed by the Random Forest model. The job begins with a CSV scan, followed by data transformations such as adaptive query execution, data redistribution, and feature processing. Key steps include sampling data, computing metadata, evaluating feature splits, and growing decision trees in parallel. The final stages aggregate results from individual trees to build the Random Forest model, with predictions and metrics computed for evaluation. This iterative, parallelized approach ensures efficient processing and robust model generation. The last jobs involves summarizing the metrics obtained during the computations.

Gaussian mixture model resulted in jobs from 209 to 259. The figures 70, 71, 72, 73, 74 show the unique jobs involved in

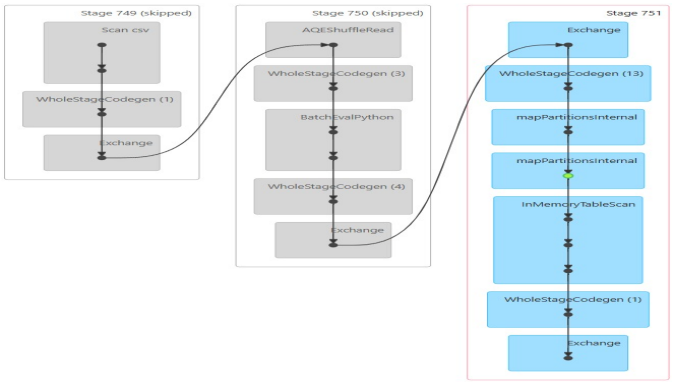

Fig. 64. Figure 64

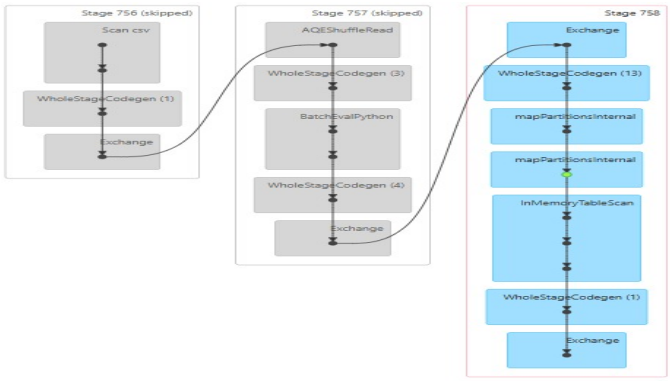

Fig. 65. Figure 65

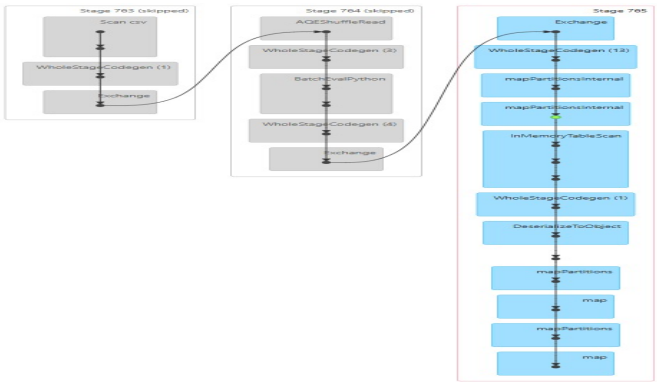

Fig. 66. Figure 66

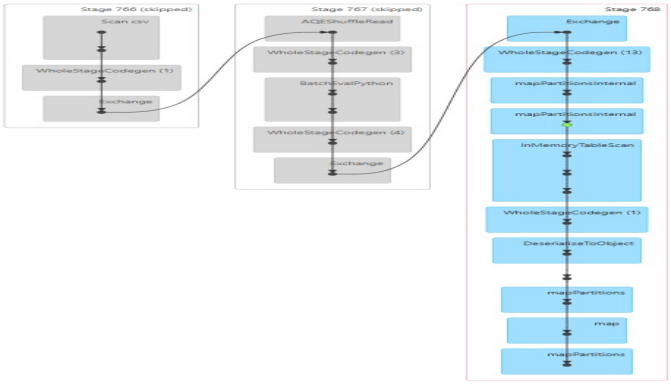

Fig. 67. Figure 67

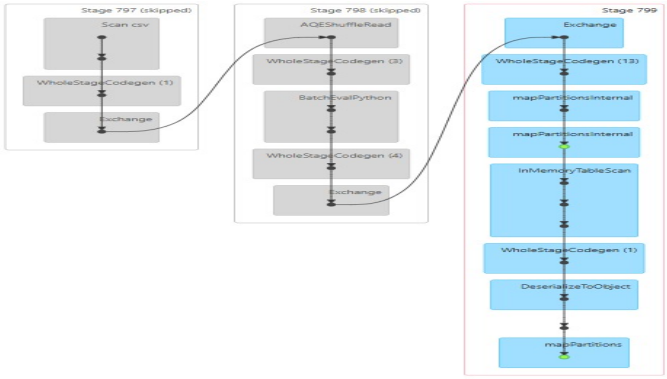

Fig. 70. Figure 70

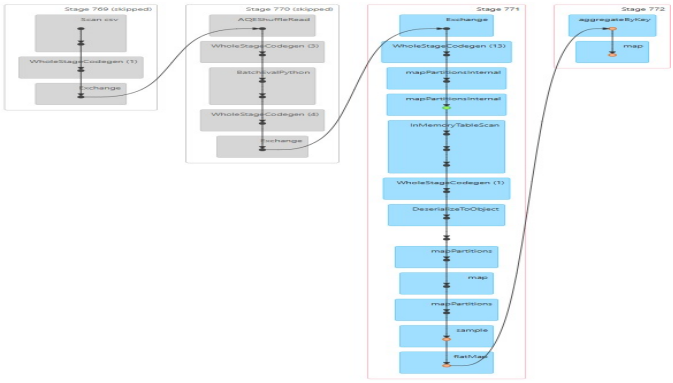

Fig. 68. Figure 68

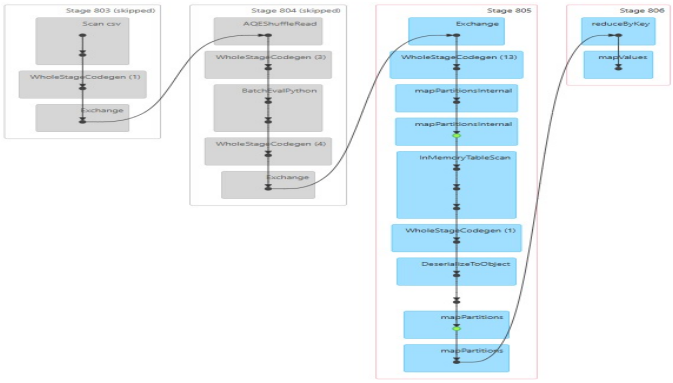

Fig. 71. Figure 71

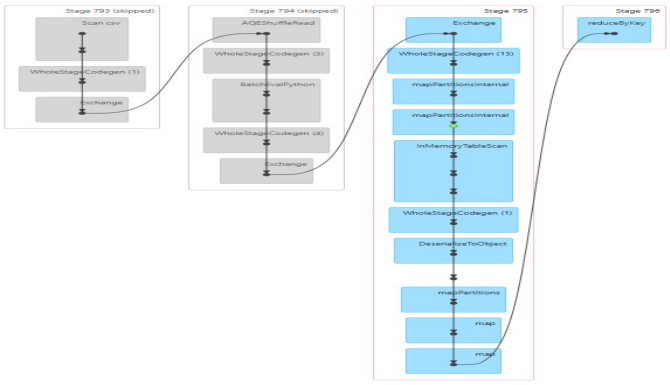

Fig. 69. Figure 69

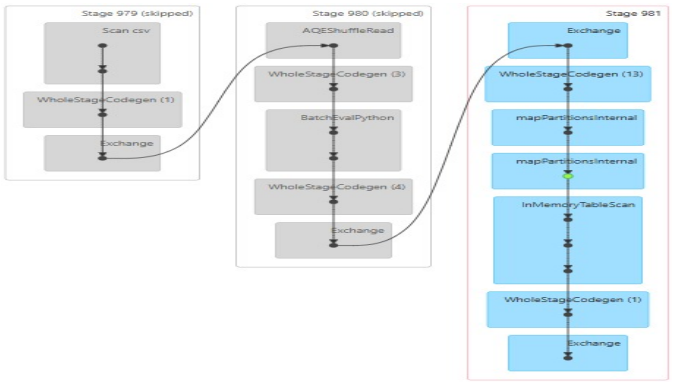

Fig. 72. Figure 72

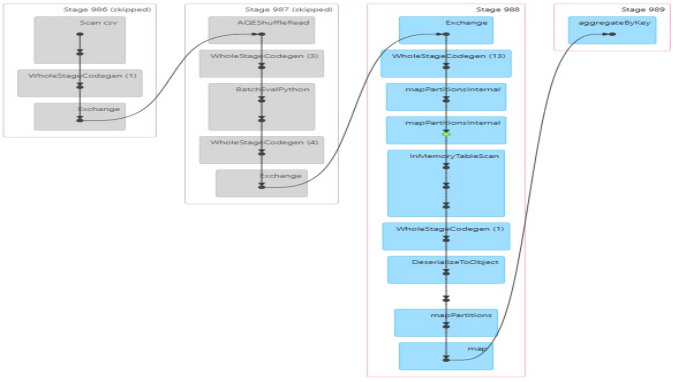

Fig. 73. Figure 73

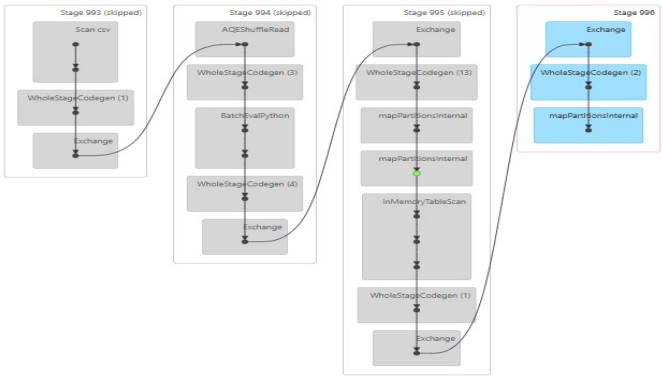

Fig. 74. Figure 74

the Gaussian mixture model. These images illustrate a series of operations performed during a Gaussian Mixture Model clustering process in PySpark. The operations include repeated data collection and sampling. The repetitive nature of collect and takeSample operations highlights iterative refinements, vital for convergence in clustering algorithms like GMM. Evaluation metrics are summerized in the final job.

B. Comparitive Analysis

TABLE I
COMPARATIVE ANALYSIS OF NON-DISTRIBUTED AND DISTRIBUTED MODELS

Model	Non-Distributed	Distributed
Hierarchical (Silhouette)	0.7529	N/A
BIRCH (Silhouette)	0.7543	N/A
Bisecting KMeans (Silhouette)	N/A	0.8470
KMeans (Silhouette)	0.7271	0.8614
Linear Regression (R2)	0.6817	0.3686
Linear Regression (MSE)	47.4M	620.7M
KNN Classifier (Accuracy)	0.9523	N/A
Logistic Regression (Accuracy)	0.9408	0.7582
Random Forest (Accuracy)	N/A	0.8541

Distributed processing of PySpark significantly enhances the scalability and performance of machine learning models, particularly with large datasets. Few algorithms are not possible to be executed using pyspark so a reasonable alternative algorithm is executed. Execution times for distributed models, like BisectingKMeans and KMeans, are higher due to task management overhead, as they show improvements in clustering quality (e.g., higher Silhouette Scores) compared to non-distributed methods. However, models like Logistic Regression and Linear Regression experience a drop in accuracy when distributed, indicating a trade-off between scalability and precision. Overall, PySpark is highly effective for large-scale data processing, offering parallelism and scalability, but comes with increased execution time and potential accuracy trade-offs.

TABLE II
EXECUTION TIME COMPARISON OF NON-DISTRIBUTED AND DISTRIBUTED MODELS

Model	Non-Distributed (seconds)	Distributed (seconds)
Hierarchical Clustering	12.39	112.6334
Elbow Method	16.97	N/A
Agglomerative Clustering	7.92	N/A
BIRCH Clustering	2.80	N/A
KMeans Clustering	3.45	43.5863
Linear Regression	0.03	10.3524
KNN Classifier	1.00	N/A
Logistic Regression	12.72	210.2223
Random Forest Model	N/A	65.3989
Gaussian Mixture Model	N/A	106.1357

C. Effectiveness and Advantages of PySpark

PySpark enhanced the effectiveness of machine learning algorithms by leveraging its distributed computing framework, making it ideal for handling large datasets. For clustering, BisectingKMeans serves as an efficient alternative to hierarchical clustering, while K-Means benefits from parallel computation for faster convergence. In linear and logistic regression models scale well, handling large data efficiently with regularization options. Random Forest implementation in PySpark supports the creation of multiple decision trees in parallel, improving accuracy and speed. Additionally, the Gaussian Mixture Model (GMM) scales well with distributed architecture of PySpark allowing for efficient probabilistic clustering, making PySpark a powerful tool for big data machine learning.

X. BONUS TASK

The above images are the screenshots of the product developed using the best performing model from phase 2. Streamlit app is developed by selecting the Kmeans clustering model to identify which cluster a customer belongs to based on the recency, frequency, monetory input values given by the user. The figure 75 shows the screenshot when a dataset is uploaded. The images in figure 76 show the input fields and the output obtained and visualized graphically for better representation.
The product link for the customer segmentation is as below.

Customer segmentation product

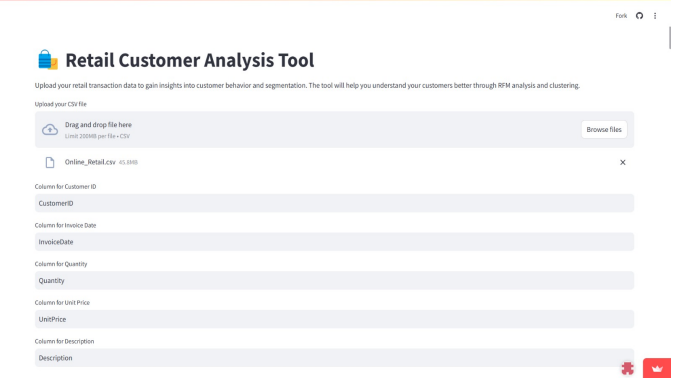

Fig. 75. Figure 75

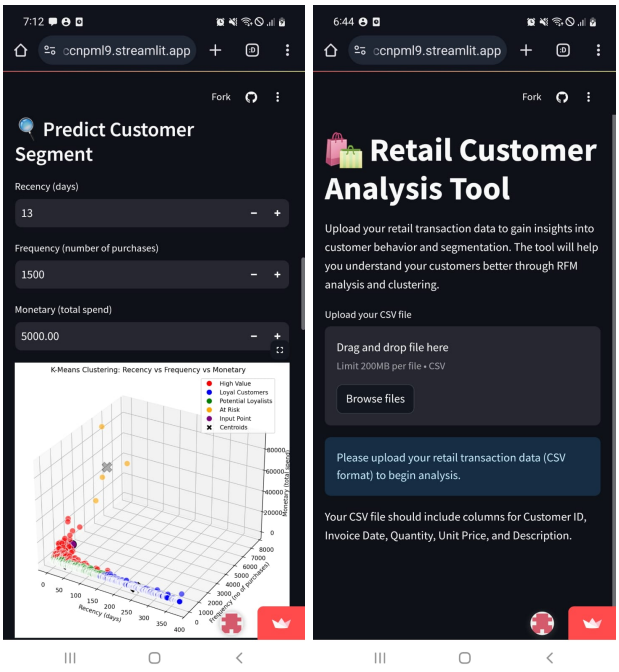

Fig. 76. figure 76

REFERENCES

[1] S. Patel, S. Sihmar and A. Jatain, "A study of hierarchical clustering algorithms," 2015 2nd International Conference on Computing for Sustainable Global Development (INDIACom), New Delhi, India, 2015, pp. 537-541. keywords: Clustering algorithms;Algorithm design and analysis;Heuristic algorithms;Partitioning algorithms;Couplings;Complexity theory;Rocks;Agglomerative;Dendrogram;Hierarchical clusterin,

[2] R. J. Gil-Garcia, J. M. Badia-Contelles and A. Pons-Porrata, "A General Framework for Agglomerative Hierarchical Clustering Algorithms," 18th International Conference on Pattern Recognition (ICPR'06), Hong Kong, China, 2006, pp. 569-572, doi: 10.1109/ICPR.2006.69. keywords: Clustering algorithms;Pattern recognition;Data mining;Data visualization;Taxonomy,

[3] A. D. Fontanini and J. Abreu, "A Data-Driven BIRCH Clustering Method for Extracting Typical Load Profiles for Big Data," 2018 IEEE Power Energy Society General Meeting (PESGM), Portland, OR, USA, 2018, pp. 1-5, doi: 10.1109/PESGM.2018.8586542. keywords: Shape;Clustering algorithms;Measurement;Time series analysis;Smart meters;Cost function;Clustering;BIRCH;smart meter data;typical load shapes;load profile.,

[4] https://seaborn.pydata.org/tutorial.html

[5] https://scikit-learn.org/stable/

[6] https://plotly.com/python/

[7] C. O'Neill and R. Schutt. Doing Data Science., O'Reilly. 2013.

[8] J. VanderPlas, Python Data Science Handbook., O'Reilly. 2016