**Part 1: Data Exploration & Preprocessing**

**Task 1: Exploratory Data Analysis (EDA)**

To identify patterns, trends, and potential anomalies within the Online Retail dataset.

* Loaded the **Online Retail dataset** from an Excel file.
* Displayed sample data to understand the structure and contents

Key Insights from Visualizations:

*Top 10 Best-Selling Products*

* I identified the most popular products based on total sales

Top 10 Countries by Sales

* Highlighted geographical trends to identify key markets
* The UK typically dominates, while other countries might represent growth opportunities

Monthly Sales Trend

* Sales peaks around year-end suggest seasonal demand, likely related to holiday shopping

Sales by Hour of the Day

* Sales spikes during business hours indicate higher customer activity during work hours

Task 2: Data Cleaning & Feature Engineering

To prepare clean, structured data for further analysis and modeling

* **Missing Data:** Removed rows with missing CustomerID and Description.
* **Anomaly Removal:** Filtered out records with negative or zero Quantity and UnitPrice.
* **Text Standardization:** Cleaned product descriptions by removing special characters.
* **Feature Engineering:** Created a new column Totalsales to represent each transaction's monetary value.

Part 2: Machine Learning Model Development

Task 3: Model Selection & Training

To build and compare multiple machine learning models to predict repeat purchases based on customer and product interaction.

Target Variable Creation - A new binary feature **RepeatPurchase** was created, indicating whether a customer has purchased a product more than once. This was achieved using the duplicated() method with the condition keep=False to mark all repeated entries.

Feature Engineering - Input features included CustomerID and Description. Product descriptions were factorized to convert categorical text data into numerical format.

Train-Test Split - The dataset was split into **80% for training** and **20% for testing** using train\_test\_split()

Model Selection - Multiple models were tested with RandomizedSearchCV to identify the best-performing algorithm: Logistic Regression, Decision Tree, Random Forest, Gradient Boosting, Naive Bayes, K-Nearest Neighbors (KNN)

Final Model Selection- Based on cross-validation scores, **KNN** with **Manhattan distance** and **distance-based weighting** was selected as the final model.

**Best Parameters:** weights='distance', n\_neighbors=3, metric='manhattan'

**Model Comparison Results:**

| **Model** | **Train Score** | **Test Score** | **Best Parameters** |
| --- | --- | --- | --- |
| Gradient Boosting | 0.80 | 0.81 | n\_estimators=300, learning\_rate=0.7, max\_depth=10 |
| **K-Nearest Neighbors** | 0.79 | **0.82** | n\_neighbors=3, weights='distance', metric='manhattan' |
| Random Forest | 0.64 | 0.64 | n\_estimators=300, max\_depth=10 |
| Decision Tree | 0.63 | 0.63 | max\_depth=10, min\_samples\_split=10 |
| Logistic Regression | 0.55 | 0.55 | Default |
| Naive Bayes | 0.55 | 0.55 | Default |

KNN outperformed other models, demonstrating strong performance on both train and test datasets.

The Manhattan distance metric effectively captured the relationships between product descriptions and customers' purchasing patterns.

Task 4: Model Evaluation & Interpretation

A screenshot of a computer program

Description automatically generated

Precision: Out of the predictions for a given class, how many were correct?

Non-Repeat: 94% of the time when the model predicted "Non-Repeat," it was correct.

Repeat: 75% of the time when the model predicted "Repeat," it was correct.

Recall: Out of all the actual instances of a class, how many were correctly predicted?

Non-Repeat: 70% of actual "Non-Repeat" cases were detected correctly.

Repeat: 95% of actual "Repeat" cases were detected correctly.

F1-Score: The harmonic mean of precision and recall.

The Non-Repeat class has a lower F1 due to its lower recall, while Repeat has a higher F1 due to strong recall.

Support: The number of actual samples for each class.

Balanced classes: 40,501 (Non-Repeat) vs 39,076 (Repeat).

Accuracy (82%): The model correctly classified 82% of the total observations.

Cosine Similarity for Product Recommendation

To recommend products based on similarity in product descriptions.

**1**: Products are identical in customer purchasing patterns.

**0**: No similarity.

**-1**: Completely dissimilar patterns.

Converted into matrix and calculated the cosine similarity

For a given product, **sort** similarity scores **descendingly**

**Exclude** the product itself (first row) because it will always have **similarity = 1**.

Return the **top n similar products**

Part 3: Deployment & Business Impact Analysis

Task 5: Model Deployment

I developed a web application using **Streamlit** and successfully deployed it on **Streamlit Cloud**. This deployment makes the application accessible via a web interface, allowing users to interact with the model easily without installing any dependencies locally

Key steps involved in deployment:

1. **Streamlit Development:**  
   Built an intuitive and user-friendly interface using Streamlit for interactive data analysis, RFM segmentation, and product recommendations.
2. **Deployment on Streamlit Cloud:**  
   Deployed the app on Streamlit Cloud, enabling access from any device with internet connectivity.

Task 6: Business Insights & Recommendations

 **Product Sales Trends:**

* **Best-Selling Products**: Insights into top-performing items to optimize inventory.
* **Sales by Time**: Identification of peak sales hours to optimize staffing and marketing efforts.

 **Product Recommendation System:**

* Implemented a **cosine similarity-based recommendation engine** to suggest products based on past purchase behavior, enhancing customer engagement and cross-selling opportunities.

 **Predictive Insights:**

* The machine learning model predicts the likelihood of repeat purchases, helping businesses identify and target customers with a high probability of returning.

Task 7: RFM Analysis Implementation

**RFM (Recency, Frequency, Monetary) segmentation** was performed using the **Online Retail Dataset**. The dataset was loaded and cleaned before calculating the RFM metrics for each customer:

1. **Recency (R)**: Days since the customer's last purchase.
2. **Frequency (F)**: Number of unique transactions made by the customer.
3. **Monetary (M)**: Total amount spent by the customer.

**Steps in the Code:**

* The maximum date in the dataset is identified.
* Grouped data by CustomerID to calculate Recency, Frequency, and Monetary.
* Quartile-based scoring (qcut) was applied to assign scores from 1 (low) to 4 (high) for each RFM metric.
* Combined these scores to create a three-digit **RFM Score** (e.g., **444** for best customers).
* Customers were segmented into four categories based on their RFM scores:
  + **VIP**: High recency (R=4) → recently active and high-value customers.
  + **Frequent**: High frequency (F=4) → customers who purchase often.
  + **Lost**: High recency (R=1) → customers who haven’t purchased in a long time.
  + **Regular**: Customers who don’t fall into the above segments.