The objective of this project is to predict the Customer Lifetime Value (LTV) based on historical transaction data. By analysing customer purchase behaviour, the model aims to assist in customer segmentation and enable more effective, targeted marketing strategies. This predictive approach allows businesses to focus their resources on high-value customers and personalize retention efforts.

The task was implemented using Python within a Jupyter Notebook environment. Core libraries used include pandas for data manipulation, scikit-learn for model evaluation and splitting, XGBoost for training a high-performance regression model, and joblib for model serialization. The final outputs consist of a trained model saved as a .ipynb file, a CSV file with predicted LTV values, and relevant visualizations for interpretation.

The dataset used for this analysis was downloaded from Kaggle and stores as a file named “transactions.csv.” Key columns within the dataset include customer\_id, transaction\_date, and amount. These fields contain the essential information required to track customer purchasing patterns and compute features for modelling.

In the data preprocessing phase, the transaction\_date column was converted to Python’s datetime format to facilitate time-based computations. All column names were standardized by converting them to lowercase and replacing spaces with underscores. The dataset was clean with no missing values in the key columns, so no imputation was required.

Feature engineering focused on generating RFM-style metrics for each customer. Specifically, Recency was calculated as the number of days since the customer's last transaction. Frequency represented the total number of transactions made by a customer. AOV (Average Order Value) was derived by taking the mean of transaction amounts, while Total Spent was computed as the sum of all transaction amounts. To estimate LTV, a simple formula was used: LTV = Frequency × AOV × 3, assuming a three-month prediction horizon.

The model was developed using the XGBoost Regressor, a robust and efficient tree-based ensemble method suitable for regression tasks. The model was configured with n\_estimators=100, learning\_rate=0.1, and max\_depth=4, and the data was split into training and test sets using an 80/20 ratio with train\_test\_split from scikit-learn.

Model evaluation was conducted using two regression metrics: Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE). These metrics provided insight into the average and squared deviations between actual and predicted LTV values. Example results included an MAE of approximately 215.32 and an RMSE of around 289.74; however, these figures may vary depending on the actual data used.