TITLE: **M-Pesa Spending Predictor using Machine Learning**

Objective:

To develop a web-based machine learning application that analyzes M-Pesa transaction statements and predicts future spending based on user input.

Project Description:

The M-Pesa Spending Predictor is a machine learning-powered web application built with Flask and XGBoost. It allows users to input details of their M-Pesa transactions and get predictions of their expected spending. The system is trained using historical transaction data and uses a combination of preprocessing, encoding, scaling, and regression modeling techniques.

For code review open GitHub account: mpesa prediction

Tools & Technologies:

1. Python (Data processing, Model training)
2. Flask (API backend)
3. HTML/JavaScript (Frontend)
4. XGBoost (Regression model)
5. scikit-learn (Preprocessing, Evaluation)
6. pandas/NumPy (Data manipulation)
7. pickle (Model serialization)
8. LabelEncoder / StandardScaler
9. CORS / flask_restful

Data Used:

Original M-pesa statement (pdf format) is cleaned and anonymized from ['Receipt No', 'Com pletion Time', 'Details', 'Transaction Status',
'Paid in', 'Withdraw\rn', 'Balance', 'Unnamed: 0']

[10]:

	Receipt No	Completion Time	Details	Transaction Status	Paid in	Withdraw\rn	Balance	Unnamed: 0
0	TC17XS3U67	2025-03-01 15:55:18	Customer Transfer to 254718***450 -\rGideon Ki...	COMPLETED	0.00	80.00	27.00	NaN
1	TC14XHM64C	2025-03-01 14:49:55	M-Shwari Deposit	COMPLETED	0.00	1,000.00	107.00	NaN
2	TC11XF54B3	2025-03-01 14:34:40	Funds received from 0706***880 - sheila\rmercy...	COMPLETED	1,100.00	0.00	1,107.00	NaN
3	TC13WSE6N5	2025-03-01 12:11:53	Pay Bill Online to 4076749 - TINGG Acc.\r25410...	COMPLETED	0.00	10.00	7.00	NaN
4	TBS1THWSOL	2025-02-28 17:50:41	Customer Transfer to 254716***061 -\rNAOMI NYA...	COMPLETED	0.00	10.00	17.00	NaN
...	...	...	...	...	...	...	...	...

To a
Cleaned M-Pesa transaction CSV with the following features:

1. Receipt
2. transaction_Day (number of transactions in a day)
3. Year
4. Month
5. Date

6. Weekday
7. Hour
8. Minute
9. Seconds
10. Transaction_type
11. Transaction_party
12. Transaction_amount
13. paid_in_or_Withdraw
14. Balance

[29]:

Receipt	transaction_Day	Year	Month	Date	Weekday	Hour	Minute	Seconds	Transaction_type	Transaction_party	Transaction_amount	paid_i
'XS3U67	49	2025	3	1	5	15	55	18	SEND MONEY	GIDEON KIPAMET KAIYIAN	0.0	
XHM64C	49	2025	3	1	5	14	49	55	M-SHWARI DEPOSIT FROM M-PESA	M-SHWARI	0.0	
1XF54B3	49	2025	3	1	5	14	34	40	RECEIVED FUNDS	SHEILA MERCY NDUSYA	1100.0	
WSE6N5	49	2025	3	1	5	12	11	53	PAY BILL	TINGG	0.0	
THWSOL	48	2025	2	28	4	17	50	41	SEND MONEY	NAOMI NYAMBURA MBURU	0.0	
...	...	...	...	...	...	...	...	...	...	...	...	

Feature Engineering:

1. Extracted relevant time features from the date and time columns.
2. Filtered and encoded categorical variables using LabelEncoder.
3. Special handling of infrequent Transaction_party values.
4. Applied feature scaling using StandardScaler.

Model Training & Evaluation:

1. Used XGBRegressor with hyperparameter tuning via GridSearchCV.

Evaluated model using:

1. Root Mean Squared Error (RMSE) 40.19
2. Mean Absolute Error (MAE) 24.71
3. R2 Score 0.47

Flask API:

/Prediction - Accepts POST requests with transaction details and returns predicted spending.
/Data - Returns historical transaction data in JSON format.

Frontend:

1. HTML form with fields for Transaction Amount, Type, Party, and Paid/Withdrawn.
2. JavaScript to submit data via Fetch API and render results.

Model Saving:

1. Serialized trained model, scaler, and encoder into .pkl files for use in the API.

Challenges & Solutions:

1. Handled unseen categorical values by mapping them to constant value eg balance 1000.
2. Addressed scaling issues by fitting the scaler only on training data.
3. Improved model performance via feature selection and parameter tuning.