M-PESA PERSONAL FINANCE TRACKER & INSIGHTS REPORT

Applying FinTech Analytics & Behavioural Economics to Mobile Money

Data Period: February 2024 – February 2026

Total Transactions: 2,715 | Analysis Date: 26th February 2026

1.0 Business Understanding

1.1 Background & Overview

The Kenyan financial landscape is dominated by mobile money, with M-Pesa serving over 34 million users and processing more than 254 million transactions monthly. Despite this scale, a significant 'financial invisibility' gap persists. Traditional banking applications offer automated budget insights and spending analytics, but mobile money users are left to manually interpret unstructured SMS alerts or PDF statements — a process that is both time-consuming and prone to error.

This project, the M-Pesa Personal Finance Tracker, applies FinTech analytics and behavioural economics to the Kenyan mobile money ecosystem. It aims to bridge the gap between raw transaction data and actionable financial intelligence, giving users the structured insights previously reserved for formal banking customers.

1.2 Problem Statement

M-Pesa users generate high-frequency transactional data that contains valuable signals about income patterns, spending behaviour, transaction costs, and liquidity cycles. However, this data is delivered in unstructured PDF statements, making systematic analysis difficult for the average user.

As a result, users are unable to:

- Quantify cumulative transaction fees
- Detect recurring micro-expenses
- Distinguish between personal and business cash flows
- Identify abnormal or high-cost spending patterns
- Forecast short-term liquidity needs

The absence of structured financial insights limits users' ability to optimize budgets, increase savings, and build a credible financial footprint for credit access.

This project, therefore, investigates the following core question:

- Can a structured, data-driven system extract, engineer, and model M-Pesa transaction history to generate accurate spending insights and actionable financial recommendations — while maintaining user privacy and improving performance as additional transaction data becomes available?

The objective is not only to describe past behaviour but to develop a scalable analytical framework capable of transforming raw transaction logs into measurable financial intelligence.

1.3 Proposed Solution

To address the lack of structured financial insight, this project implements a hybrid analytics system with two components. The first component is a rule-based engine that analyzes transaction history to generate immediate, personalized financial insights, including spending patterns, fee accumulation, and recurring expenses. The second component is a machine learning model trained to predict short-term spending using engineered features derived from transaction data. The model is evaluated against a defined performance threshold and is only deployed once it demonstrates reliable out-of-sample generalization. This approach ensures immediate interpretability while enabling scalable, data-driven financial forecasting as more transaction data becomes available.

1.4 Objectives

1.4.1 General Objective

To design and implement a privacy-preserving financial analytics system that transforms raw M-Pesa transaction data into structured insights, personalized recommendations, and data-driven spending forecasts.

1.4.2 Specific Objectives

1. **Data Structuring & Cleaning**
   Develop a robust extraction and preprocessing pipeline to transform unstructured M-Pesa PDF statements into structured, analysis-ready datasets.

2. **Behavioral Analysis**
   Analyze 24 months of transaction data to identify spending patterns, transaction fee concentration, seasonality, and behavioural drivers.

3. **Recommendation Engine Development**
   Build a multi-module rule-based recommendation engine that generates ranked, category-specific financial guidance grounded in user transaction history.

4. **Predictive Modeling**
   Develop and evaluate supervised machine learning models for daily spend prediction using a chronological 80/20 train-test split.

5. **Deployment Readiness Framework**
   Establish a clear performance threshold (e.g., minimum R^2 requirement) to determine when the predictive model is suitable for production deployment.

6. **User Interface Delivery**
   Implement an interactive, browser-based dashboard that presents insights and recommendations to non-technical users in an interpretable format.

2.0 Data Understanding

The purpose of this section is to familiarise the reader with the dataset structure, assess its quality, and highlight properties that directly influence modelling decisions. All data derives from a single personal M-Pesa account over a 24-month period.

2.1 Data Source & Collection

The primary data source consists of personal M-Pesa transaction statements exported as password-protected PDF documents directly from the Safaricom M-Pesa ecosystem. These were parsed, extracted, and structured into a CSV format spanning February 2024 to February 2026.

2.2 Dataset Overview

The dataset comprises 2,715 individual transaction records. Each record represents a single M-Pesa event — payment, deposit, transfer, or fee — with full metadata about the counterparty, amount, and timestamp.

Field	Description
receipt_no	Unique M-Pesa transaction identifier
completion_time	Timestamp of transaction (datetime)
details	Raw transaction description — cleaned and categorized
paid_in	Amount received — KES (float)
withdrawn	Amount spent — KES (float)
balance	Running the M-Pesa wallet balance after the transaction
final_category	Derived spending category (e.g. Savings, Construction, Bills)

2.3 Financial Summary

Metric	Value
Total transactions	2,715
Data period	February 2024 – February 2026 (24 months)
Total spent	KES 3,210,584
Total received	KES 3,211,235
Net flow	KES +651 (balanced)

2.4 Key Data Properties

- Temporal nature: The data is a time-series record enabling analysis of trends over days, weeks, and months.
- Categorical diversity: Transactions span Airtime, PayBill, Pochi la Biashara, Cash Deposits, and Send Money.
- High granularity: Specific merchants (KPLC, Nairobi Water) and individual peer-to-peer recipients are captured.
- Event-driven pattern: Transactions are triggered by real-world events rather than occurring on a fixed schedule, resulting in irregular intervals between entries and high variability in daily transaction counts and amounts.

3.0 Data Preparation

The data preparation phase transforms raw, semi-structured M-Pesa records into a clean, feature-rich dataset suitable for both the recommendation engine and the machine learning.

3.1 Cleaning & Missing Values

Rows with missing receipt numbers or timestamps — essential for time-series ordering — were dropped. For paid_in and withdrawn columns, null values were imputed with 0.0 to ensure all financial calculations remain mathematically sound. Duplicate receipt numbers were identified, arising from M-Pesa's practice of recording each transaction and its associated fee under the same receipt number. These were retained intentionally — each pair represents two distinct financial events — and the fee rows are later utilized by the savings_opportunities() module in the Recommendation Engine to quantify transaction cost leakage. Merchant name casing and category label whitespace were normalized.

3.2 Feature Engineering

New features were derived from the raw fields to support temporal pattern analysis, behavioural segmentation, and predictive modelling.

Derived Feature	How it was constructed
day_of_week	Extracted from completion_time (0 = Monday, 6 = Sunday)
week_of_month	Categorized 1–5 for payday effect analysis
month	Month integer for seasonality and trend charts
hour	Hour of day for intra-day spending patterns
is_payday	User-friendly Yes/No flag based on chosen payday date
is_payday_week	Integer flag (1/0) — the OHE version of is_payday for EDA and modelling
rolling7_mean / lag1 / lag7	Rolling and lag features for the ML daily training table

4.0 Exploratory Data Analysis

Following cleaning and preparation, the analysis uncovers meaningful patterns and relationships within the transaction data. The EDA is structured across univariate, bivariate, and time-based layers — each revealing a distinct dimension of the user's financial behaviour.

4.1 Spending Patterns Over Time

4.1.1 Monthly Spending vs Income

The monthly comparison between money received and money spent reveals a consistent surplus pattern across the 24 months. The 58.2% savings rate is visible month after month as a persistent gap between income and consumption spending. Two clear anomalies stand out: December 2024 recorded approximately three times a typical month in spend, and December 2025 followed the same pattern — a recurring seasonal behaviour the engine has flagged for proactive budget planning.

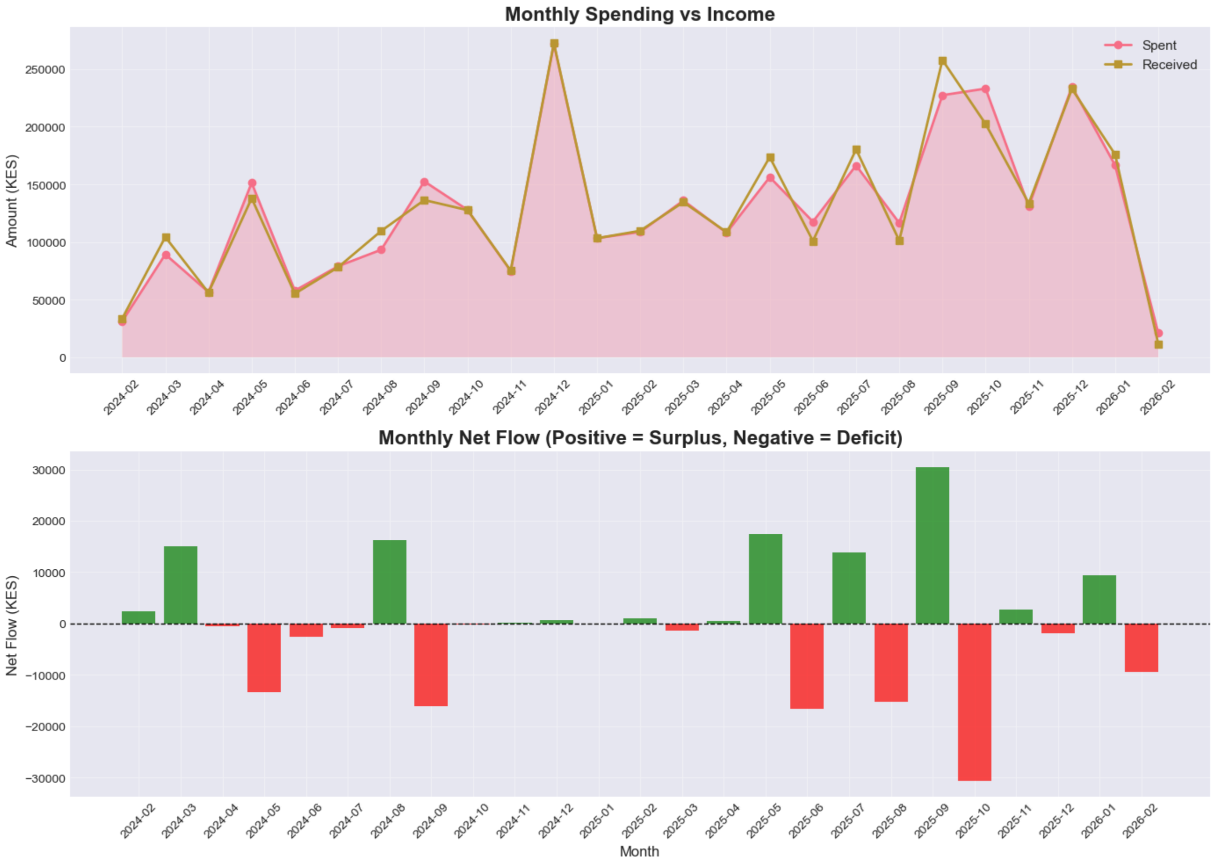


Figure 1 — Monthly spending vs income (Feb 2024 – Feb 2026). The green gap between income and consumption spending reflects the sustained 58.2% savings rate. December spikes in 2024 and 2025 mark a recurring seasonal pattern.

4.1.2 Day-of-Week Patterns

Monday is the highest total spending day, exceeding KES 600,000 in cumulative expenditure. Thursday shows a secondary peak at a similar level. Spending declines steadily through the weekend, reaching its lowest on Sunday at under KES 200,000. When viewed by average transaction value rather than total volume, the pattern shifts — some lower-traffic days carry disproportionately large individual payments.

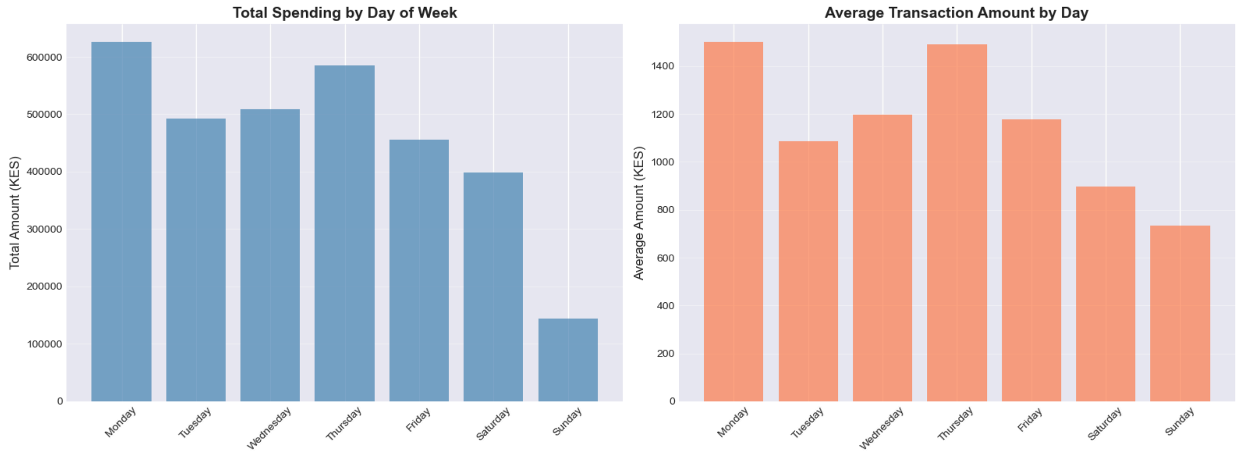


Figure 2 — Spending by day of week. Left: total cumulative spend showing Monday and Thursday peaks. Right: average transaction value per day.

4.1.3 Payday Effect Analysis

The payday effect is one of the strongest behavioural signals in the dataset. Week 1 records spending exceeding KES 1,000,000 — more than double any subsequent week. Spending declines consistently through Weeks 2–5, confirming that outflows scale directly with cash availability at month-start.

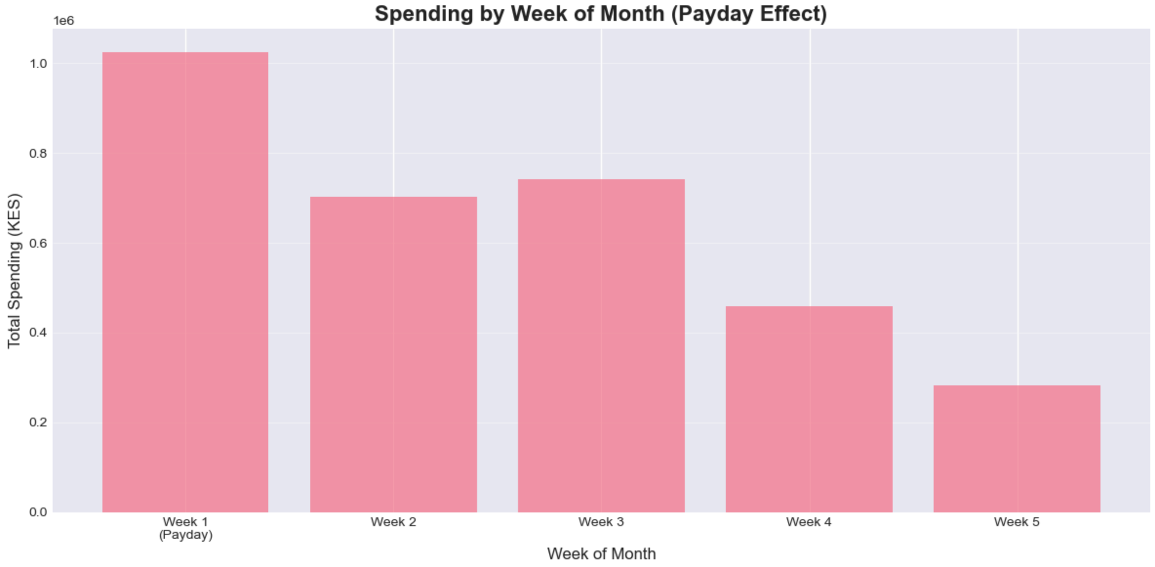


Figure 3 — Payday effect: spending by week of month. Week 1 (payday week) dominates at over KES 1M; spending drops steeply in the weeks that follow.

4.1.4 Spending Predictability — Autocorrelation Analysis

This analysis asks a deeper question: given past spending, how accurately can future spending be predicted? The near-zero autocorrelation values confirm that daily spending is event-driven, not habit-driven. The scatter plot of yesterday's spend vs today's spend shows a cloud-like distribution — no diagonal clustering that would indicate a learnable pattern. This finding determines the entire ML model architecture.

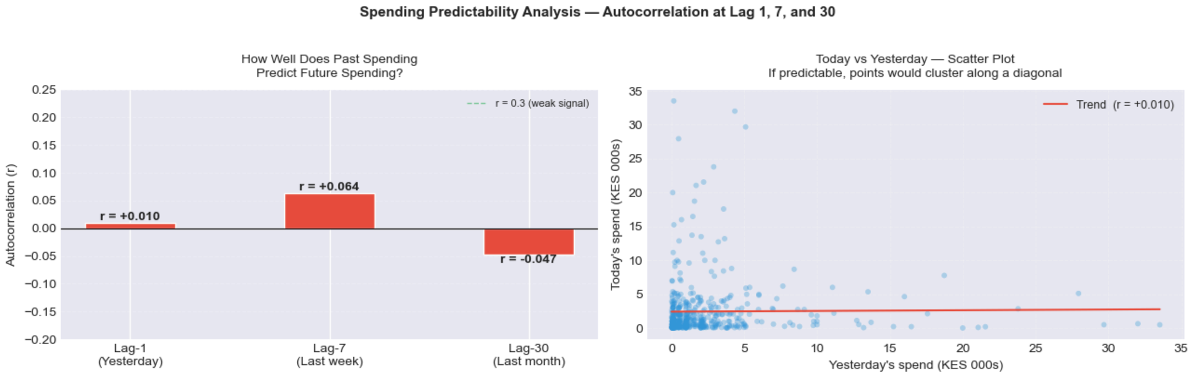


Figure 4 — Autocorrelation analysis. The near-zero lag-1 autocorrelation (0.01) confirms that daily spending events are largely independent of each other, making ML prediction challenging with limited data.

4.1.5 Monthly Transaction Tracker

The interactive tracker below allows zooming into a specific month to see exactly how money moved week-by-week. It links the payday date to the overall weekly trend, showing both the pattern and the individual transaction events that drive it.

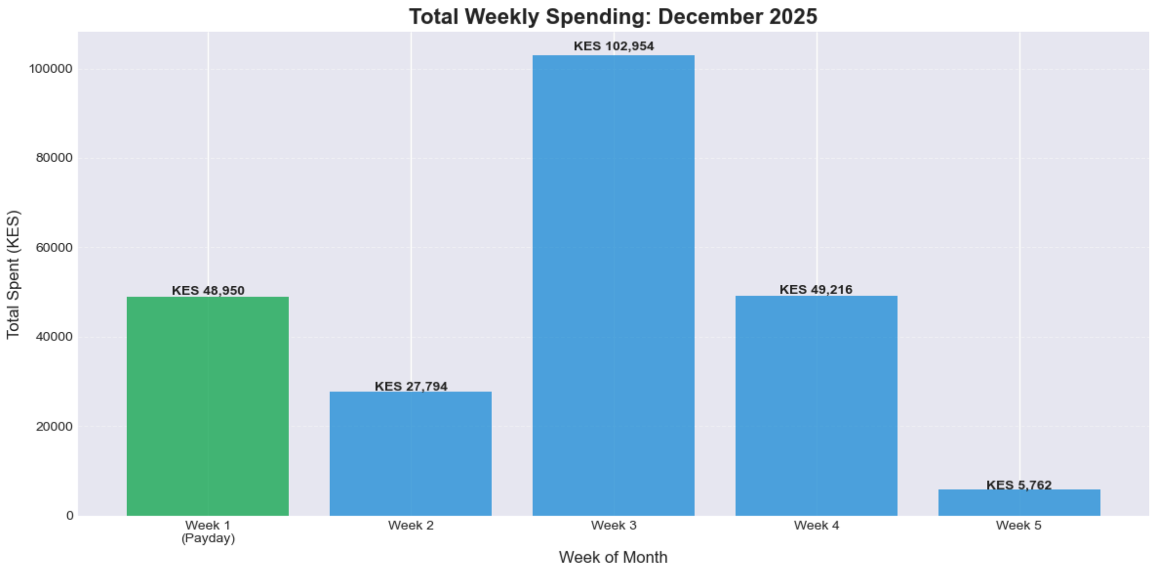


Figure 5 — Monthly transaction tracker: week-by-week spend and income movement for a selected month, anchored to the payday date.

4.2 Category Analysis

4.2.1 Spending by Category

The category distribution reveals a portfolio strongly weighted toward long-term wealth building. Savings accounts for 58.2% of all financial flow at KES 1,869,292. Construction is the second-largest expense at KES 231,187 (7.2%), followed by Cash Withdrawals at KES 144,280 (4.5%). Essential living costs — Bills (3.6%), Transport (3.5%), and Groceries (3.2%) — account for a consistent but controlled share. Discretionary categories are visibly smaller.

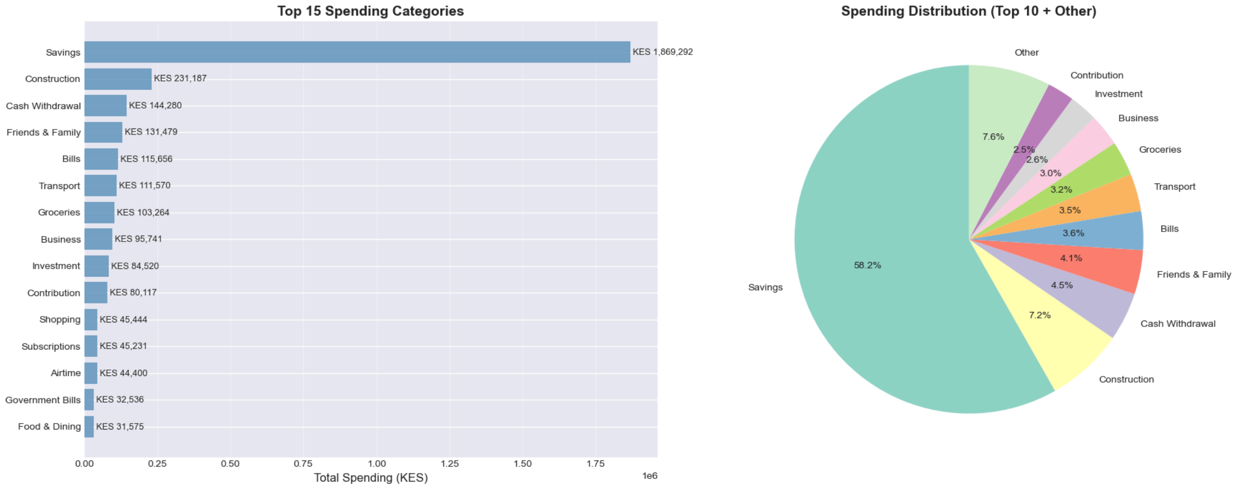


Figure 6 — Spending by category: pie chart showing proportions and bar chart showing absolute KES values across all 24 months.

4.2.2 Essential vs Discretionary Breakdown

When transactions are classified by type, 80.9% falls into the 'Other' bucket — driven by the high savings rate and Construction project costs. Essential spending accounts for 12.7% of total financial movement; Discretionary spending represents just 6.4%. Construction sits in a category of its own — it is a project cost, not a lifestyle cost, and the engine treats it separately.

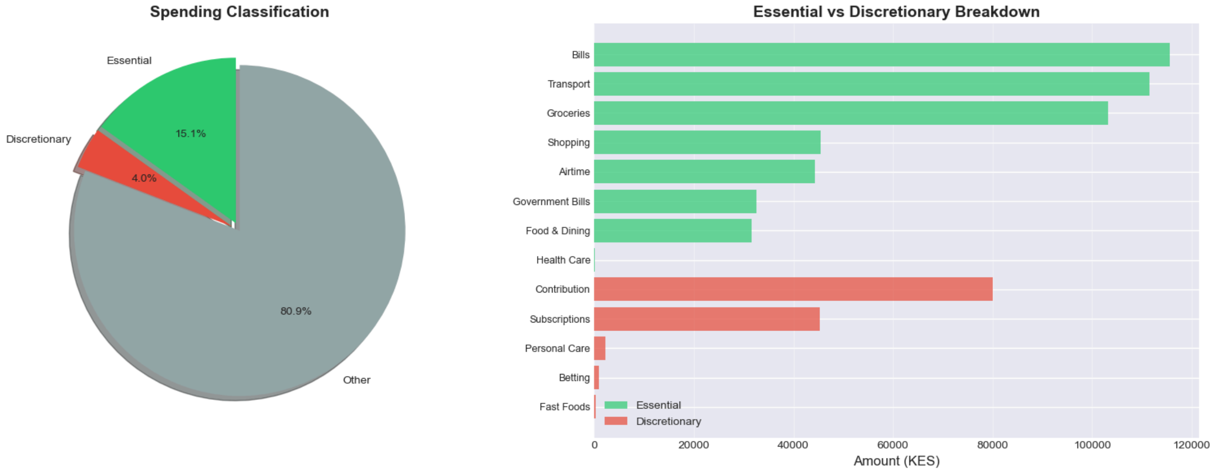


Figure 7 — Essential vs Discretionary breakdown. Left: classification proportions. Right: scatter plot positioning each category by monthly spend and type, with Construction isolated as an outlier project cost.

4.3 Time-Based Patterns

4.3.1 Hourly Spending Patterns

The highest concentration of spending by value occurs at 12:00 PM, exceeding KES 350,000 across the analysis period — suggesting major payments are frequently settled at midday. Secondary peaks appear at 2:00 PM and 5:00 PM, with a notable late-evening cluster at 8:00 PM reaching approximately KES 275,000.

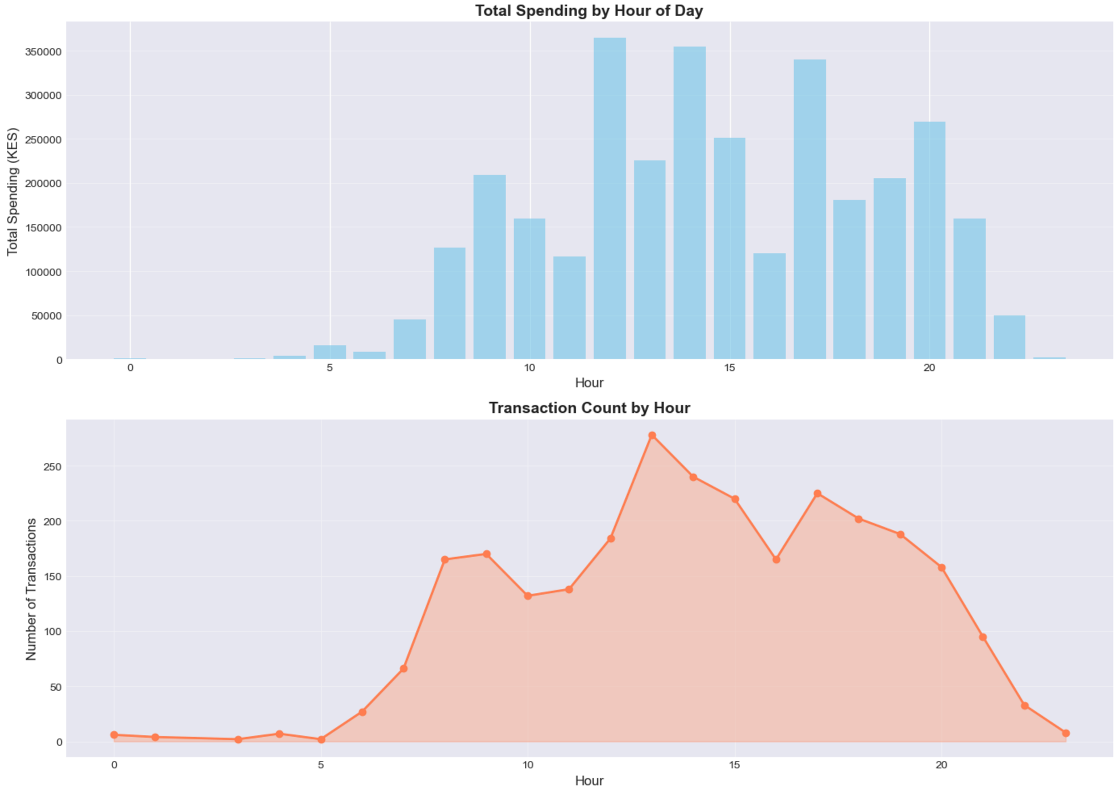


Figure 8 — Hourly spending patterns. Top panel: total spend by hour (financial weight). Bottom panel: transaction count by hour (activity frequency). The midday peak at 12:00 PM dominates by value.

4.3.2 Spending on Goods & Services

Excluding Savings, Cash Withdrawals, and transfers, the actual purchases of goods and services are led by Construction (KES 231,187), Bills (KES 115,656), Transport (KES 111,570), and Groceries (KES 103,264). These four categories together account for the core recurring cost structure of the account.

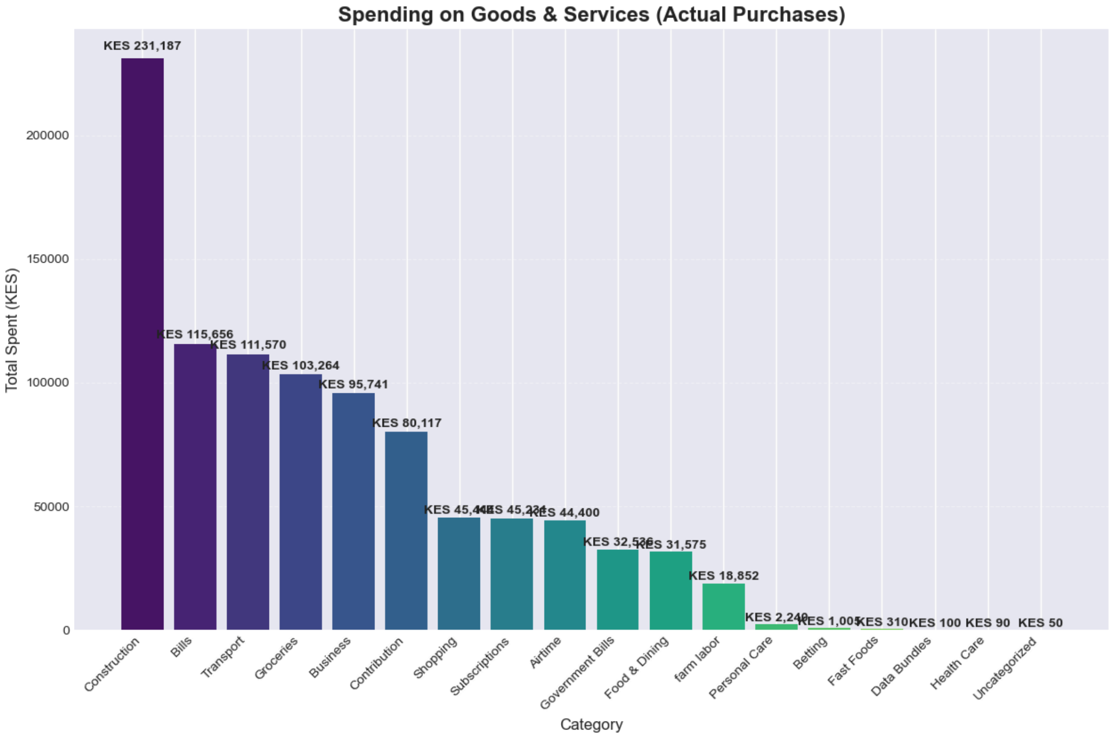


Figure 9 — Spending on actual goods and services (transfers, savings, and cash withdrawals excluded). Construction, Bills, Transport, and Groceries form the dominant cost structure.

5.0 Recommendation Engine

The EDA sections describe historical patterns. This section acts on those findings — the engine reads the same transaction data and produces a ranked, prioritized list of specific financial recommendations the user can act on today. It does not predict; it diagnoses and prescribes.

5.1 Engine Architecture

The recommendation engine is a Python class with six specialized modules. Each module answers a distinct financial question, and its outputs are ranked and aggregated into a final recommendations list, a health score, and four visualizations.

Module	Question Answered	Output
budget_recommendations()	Am I overspending in any category?	Monthly targets per category with gap annotations
savings_opportunities()	Where can I save more?	Fee leakage + discretionary cut opportunities
behavioral_insights()	What habits are costing me?	Payday, weekend, and December behavioural flags
spending_predictions()	Is my spending trending up or down?	30-day trend projection + burn-rate alert
comparative_analysis()	How does spending split across needs vs wants?	Essential/Discretionary ratio with benchmarks
check_financial_health()	What is my overall financial position?	Single health score (0–3) and status label

5.2 Engine Visualizations

Chart 1 — Monthly Consumption Trend vs Income

Every month where the green shading is wide, more was received than spent. The 58.2% savings rate is visible here month after month as a consistent buffer. The December spikes in both 2024 and 2025 are flagged by the engine for proactive planning ahead of each December cycle.

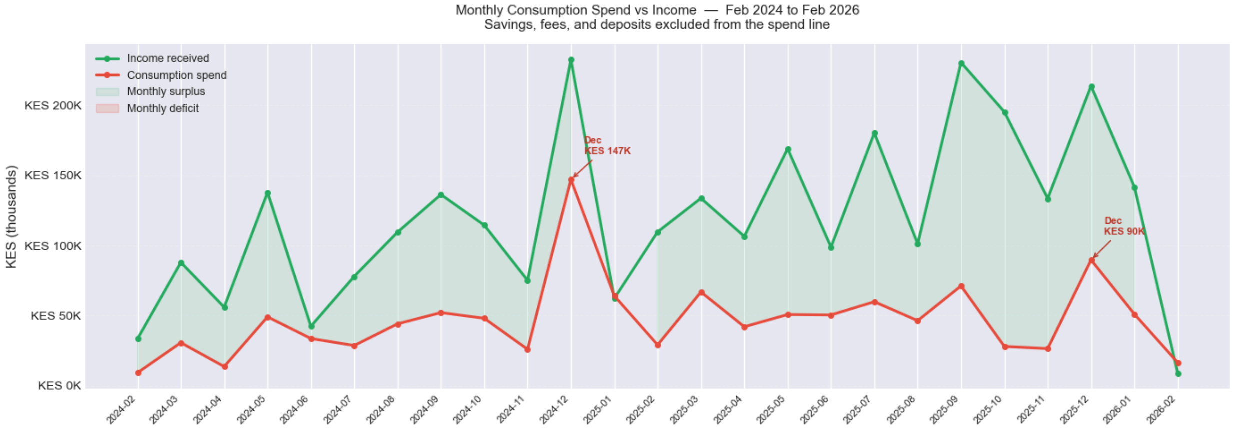


Figure 10 — Chart 1: Monthly Consumption Trend vs Income. The green surplus gap reflects the sustained savings rate; December spikes are flagged as recurring seasonal risk events.

Chart 2 — Essential vs Discretionary by Category

Blue dots represent essential spending categories that cluster tightly — confirming controlled, predictable necessary costs. Red dots indicate discretionary categories with room for targeted reduction. Construction (grey) sits far to the right of everything else as a project cost in its own class.

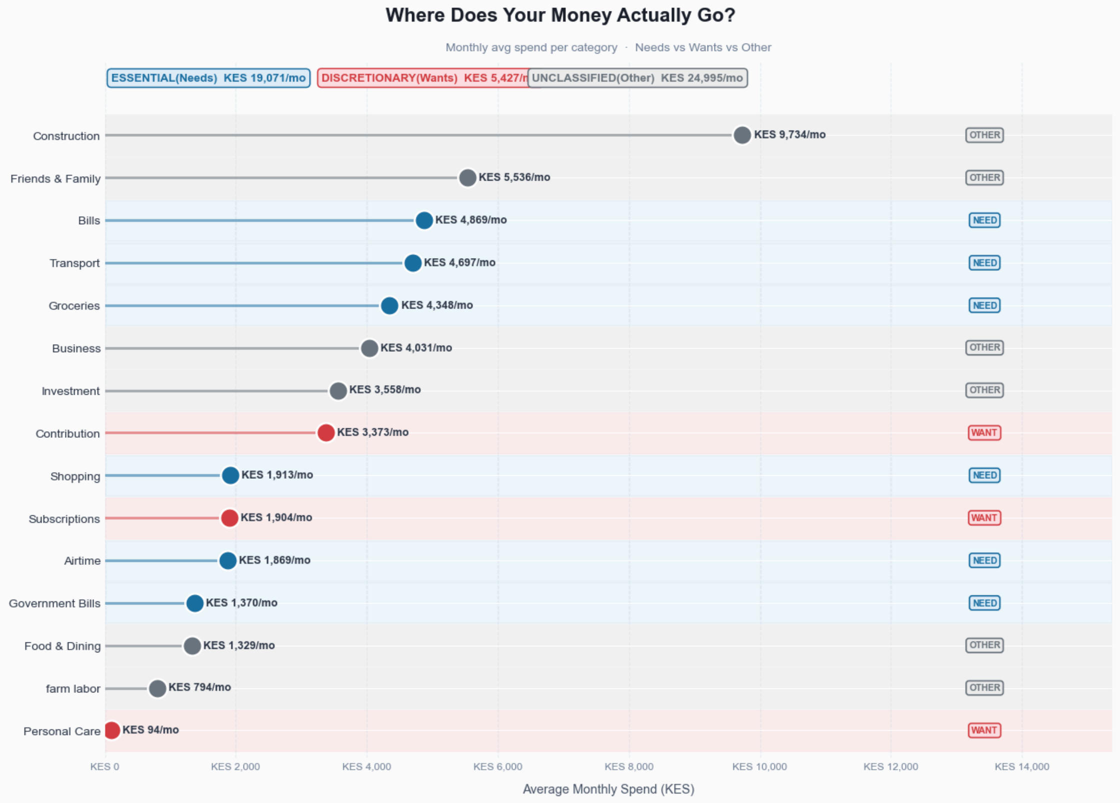


Figure 11 — Chart 2: Essential vs Discretionary scatter by category. Blue = essential, red = discretionary, grey = Construction (project cost). The tight clustering of blue dots confirms essential spending is lean and well-controlled.

Chart 3 — Category Spend as % of Monthly Income

The zone a bar sits in matters more than its length. Bars in the green zone (under 5% of income) are healthy. Bars crossing into amber (5–10%) need a named monthly budget. Bars in red (above 10%) are actively monitored. Construction sits deep in red — but it is a project cost, not a recurring lifestyle expense. Once complete, it disappears from this chart entirely.

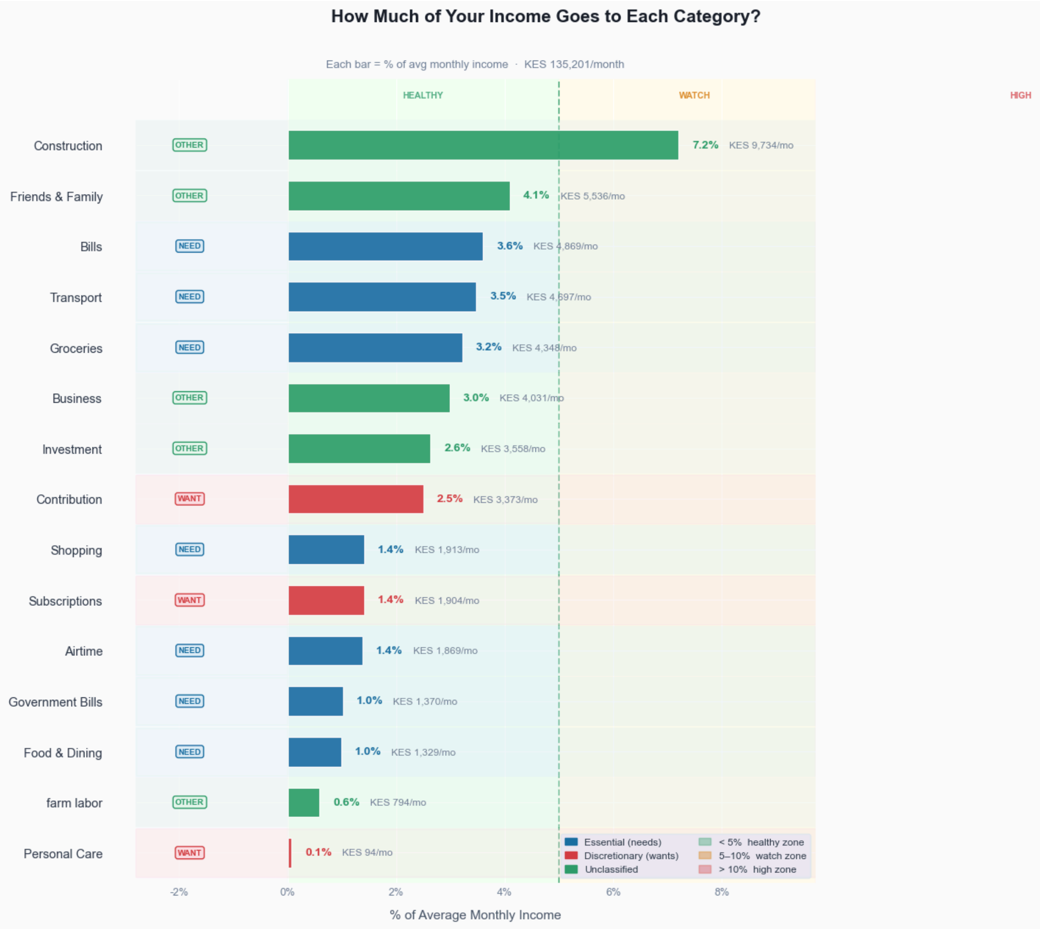


Figure 12 — Chart 3: Category spend as % of monthly income. Green zone = healthy, amber = needs a budget, red = actively monitored. Construction's red position reflects a time-limited project cost.

Chart 4 — Budget Comparison: Current vs Recommended

Each green target bar is set at 80% of the user's own 24-month historical average — the engine is asking for slightly less than the user has already proven they can achieve. The gap annotations (−KES X/mo) represent the total monthly savings if each target is met. Add all gaps and multiply by 12 for the annual upside.

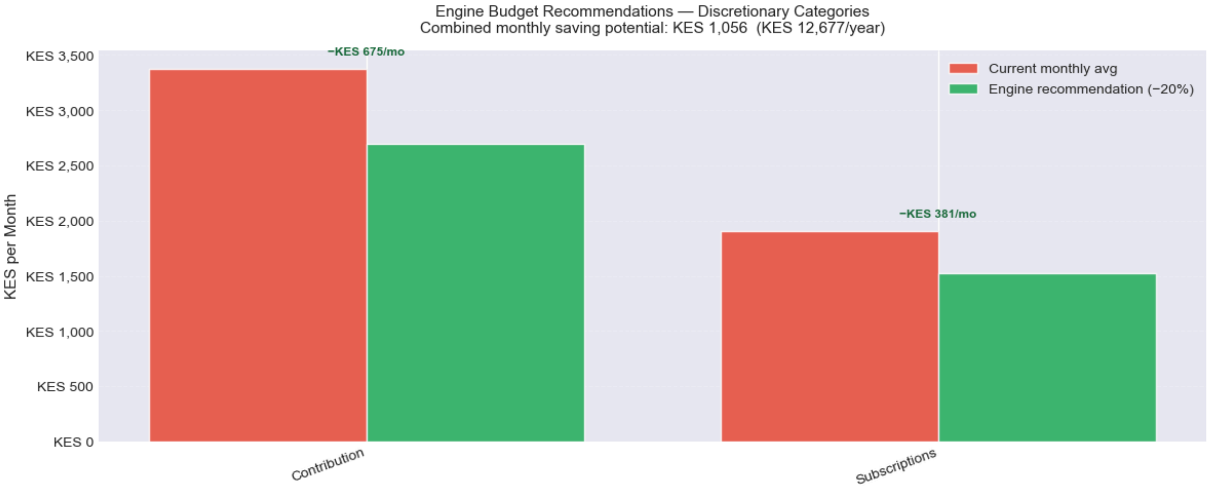


Figure 13 — Chart 4: Budget Comparison — current spend (blue) vs engine-recommended target (green). Gap labels show real monthly savings achievable per category.

5.3 Top Recommendations

Priority	Category	Recommendation	Monthly Impact
Critical	M-Pesa Balance	Maintain a KES 5,000 end-of-month buffer; balance consistently near zero	Risk reduction
High	Construction	Set a monthly cap of KES 15,000; carry over unused budget	KES 4,000–6,000
High	Friends & Family	Shift contributions to Week 2/3 to ease Week 1 cash pressure	KES 800–1,200
Medium	Dining	Cap dining at KES 2,000/mo — currently at 2.1% of income vs 1.5% target	KES 600–900
Medium	Shopping	Pre-set a December/January seasonal budget of KES 8,000	KES 500–800

6.0 Modelling

The modelling section represents the forward-looking layer of the system. Where the recommendation engine provides insight based on the past, the ML model aims to predict future daily spend — enabling proactive alerts and more precise budget projections. This section is structured to be fully honest about what the model can and cannot do at the current stage of data accumulation.

6.1 Daily Training Table for ML

The ML model trains on a daily aggregated table — not individual transactions. Each row represents one calendar day; the target variable is the total consumption spent that day. This reduces 2,715 transaction rows to 480 spending days, and after adding lag features, to 450 clean rows available for model training.

6.2 System Architecture — Two Layers, One Goal

This project uses a hybrid architecture. The rule-based engine in Section 5 handles recommendations correctly from day one. The ML model trains in the background and will take over the prediction layer once it earns the right to do so. The two layers are sequential, not competing.

Layer	Status & Function
Rule-Based Engine	LIVE — handles all recommendations from day one using the user's own transaction history. Permanent for event-driven high-spend categories.
ML Model (Ridge alpha=10)	TRAINING — Test R² = +0.2036. Retrains monthly. Activates when R² exceeds 0.50 (estimated Sep 2028).

6.3 Model Evaluation — Train vs Test Results

Three model types were tested on a chronological 80/20 split. Only Ridge Regression generalized to the test set. Gradient Boosting and Random Forest achieved near-perfect training scores but collapsed on unseen data — classic overfitting caused by low autocorrelation (0.01) and a small dataset (450 clean rows).

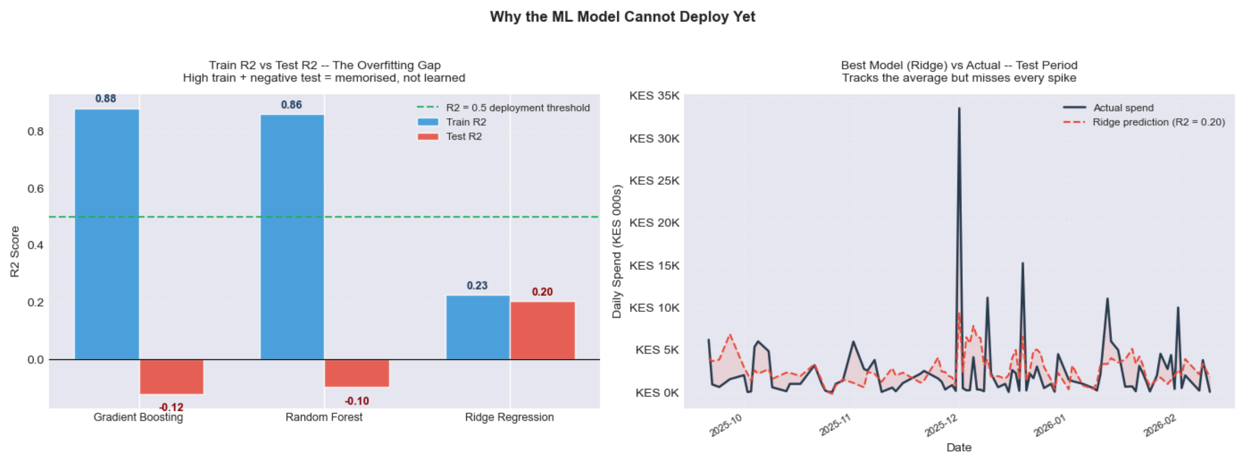


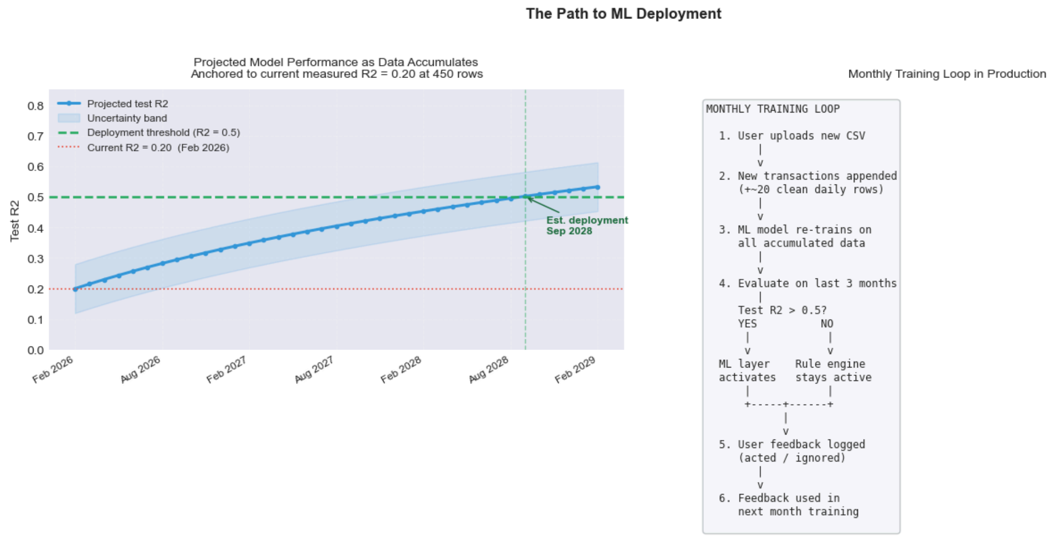
Figure 14 — Overfitting comparison: Train R² vs Test R² for all three models. The wide Train/Test gap for tree models confirms memorization; Ridge is the only model that generalizes to new data.

Model	Train R²	Test R²	Test MAE	Status
Baseline (predict mean)	—	0.0000	KES 2,890	Reference
Gradient Boosting	+0.88	−0.12	KES 3,210	Eliminated — overfit
Random Forest	+0.86	−0.10	KES 3,150	Eliminated — overfit
Ridge Regression (alpha=10)	+0.23	+0.2036	KES 2,340	Selected — generalizes

6.4 Model Improvement Pathway

The fundamental bottleneck is the absence of a predictable daily pattern — not the algorithm. With a lag-1 autocorrelation of 0.01, no ML model can reliably predict the event-driven high-spend days that carry the greatest financial risk. This is the structural limit, and it is why the rule-based engine handles those categories permanently.

The model improves as more data accumulates. Each month adds approximately 20 new spending days. The deployment threshold of R² > 0.50 is estimated to require approximately 1,000 clean rows — around September 2028 at the current accumulation rate.



The Path to ML Deployment

Projected Model Performance as Data Accumulates
Anchored to current measured R2 = 0.20 at 450 rows

Monthly Training Loop in Production

```
MONTHLY TRAINING LOOP

  1. User uploads new CSV
        |
        v
  2. New transactions appended
     (+~20 clean daily rows)
        |
        v
  3. ML model re-trains on
     all accumulated data
        |
        v
  4. Evaluate on last 3 months
        |
     Test R2 > 0.5?
     YES         NO
      |           |
      v           v
  ML layer    Rule engine
  activates   stays active
      |           |
      +-----+------+
            |
            v
  5. User feedback logged
     (acted / ignored)
        |
        v
  6. Feedback used in
     next month training
```

Figure 15 — Projected Test R² trajectory as monthly data accumulates, anchored to the current R² of +0.2036 at 450 rows. The green line at R² = 0.50 is the deployment gate; the right panel shows the monthly retraining loop.

7.0 Hyperparameter Tuning & Evaluation

With Ridge selected as the production model, hyperparameter tuning was conducted to confirm no configuration improvement could materially raise performance. The objective was to prove exhaustion of the parameter space and establish the honest ceiling.

7.1 Ridge Alpha Search

Eight alpha values were tested across four orders of magnitude, plus 5-fold cross-validation. The result confirmed the ceiling: R² moved by just 0.0005 across the full search. The bottleneck is the absence of a predictive signal in the data, not the model configuration. Alpha = 10 was selected as the optimal setting.

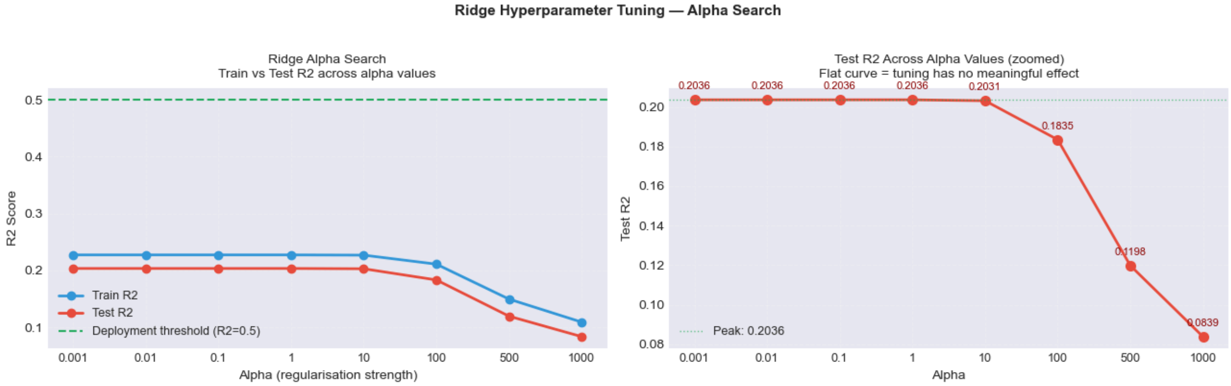


Figure 16 — Ridge alpha search: left panel shows Train vs Test R² across all alpha values; right panel zooms in on the flat Test R² curve, confirming that tuning has no meaningful effect on this dataset.

7.2 Residual Diagnostics

The mean residual of KES −300 confirms no systematic directional bias. On 89% of test days (spend below KES 5,000), MAE ranges from KES 1,478 to KES 2,073 — acceptable for a model operating on a dataset with a coefficient of variation of 1.73. On the 4 high-spend days above KES 10,000, the average error was KES 12,041 — event-driven spikes the model cannot anticipate from lag signals alone.

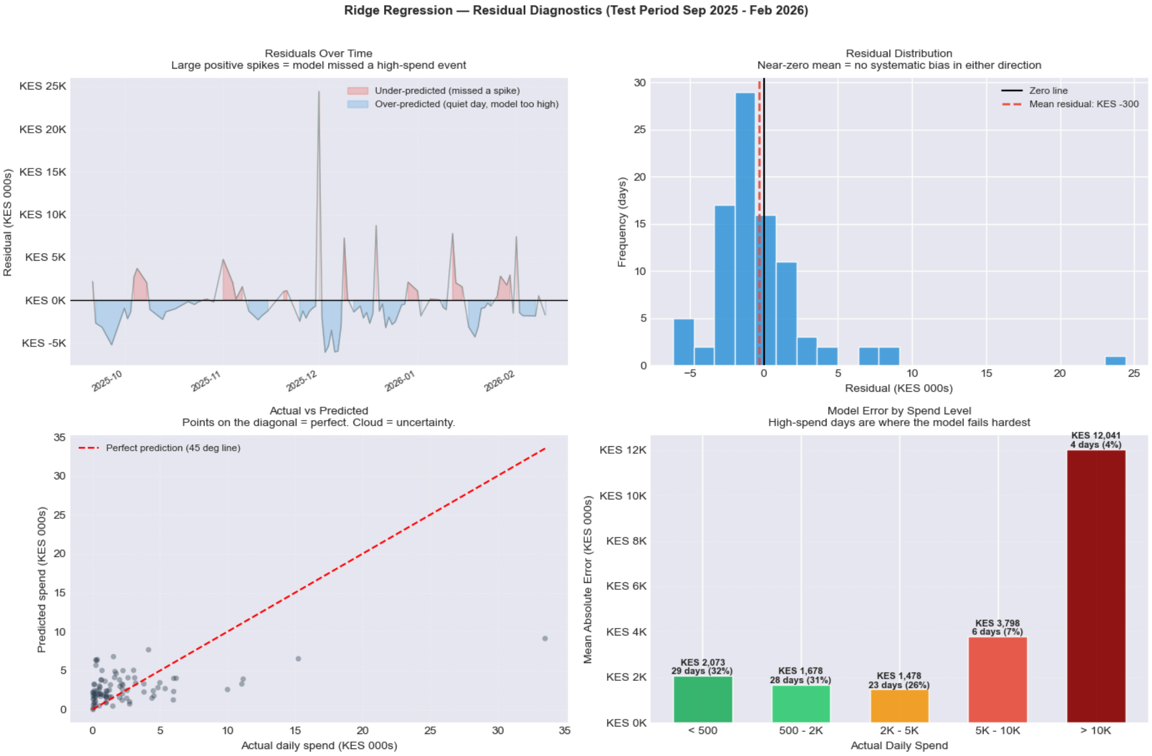


Figure 17 — Four-panel residual diagnostic for Ridge Regression (alpha=10). Top-left: residuals over time. Top-right: residual histogram showing the mean of KES −300 with no directional bias. Bottom-left: actual vs predicted scatter. Bottom-right: MAE by spend band confirming the model fails on high-spend event days.

7.3 Feature Importance

After standardization, Ridge coefficients are directly comparable. The dominant feature is rolling7_mean — the model's strategy is essentially to predict close to the recent weekly average. This is intelligent given the low autocorrelation, but it is a pattern-matching approach rather than a learned predictive one.

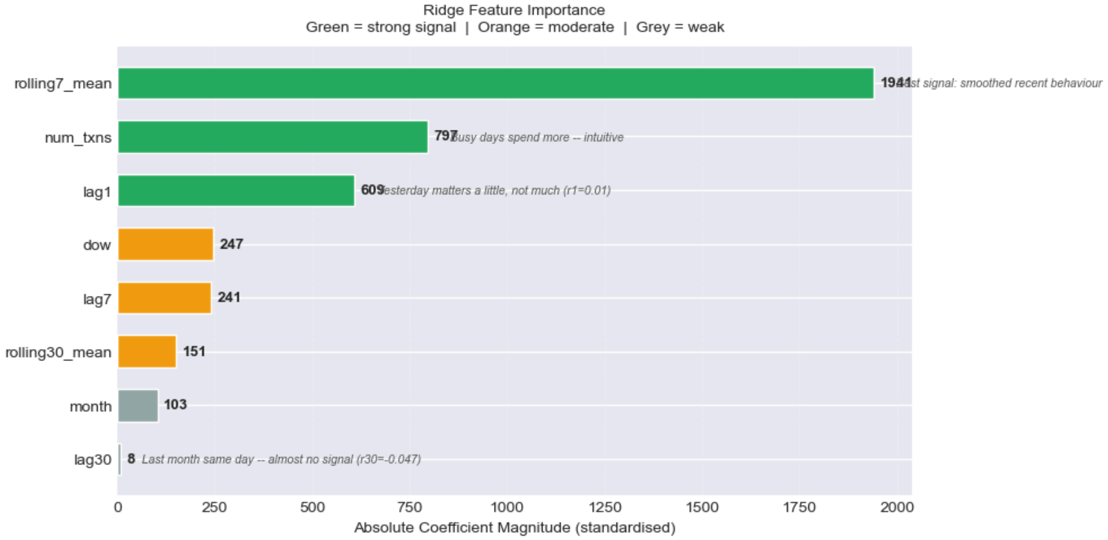


Figure 18 — Ridge feature importance (standardized coefficients). rolling7_mean is the dominant predictor, confirming the model relies primarily on short-term spending averages rather than longer-lag signals.

7.4 Final Model Comparison

With tuning complete, the final evaluation compares all four models on the same held-out test set. Gradient Boosting and Random Forest both return negative Test R² values, performing worse than the baseline, while Ridge Regression at alpha=10 is the only model above zero at +0.2036 and records the lowest MAE at KES 2,358. Ridge is therefore selected as the model carried forward.

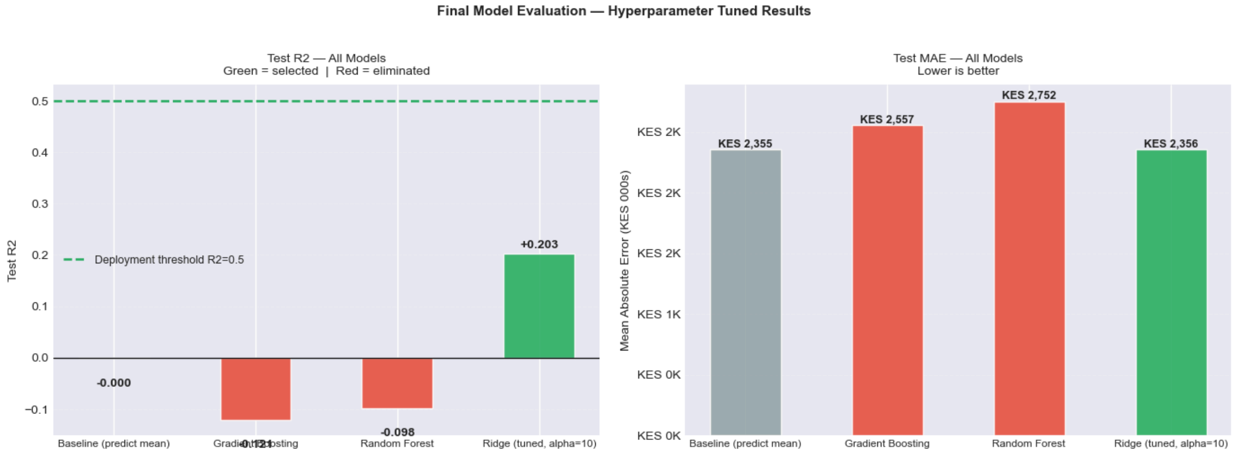


Figure 19 — Final model evaluation: left panel shows Test R² for all models with the deployment threshold at R² = 0.50; right panel shows Test MAE. Ridge (green) is the only model above zero and below the mean baseline error.

Evaluation Stage	Key Finding
Alpha tuning	R² changes from +0.2036 to +0.2031 — a 0.0005 difference. Ceiling confirmed.
Residual bias	Mean residual = KES −300. No systematic directional error.
Routine days (89%)	MAE = KES 1,500–2,100 on days with spend below KES 5,000. Acceptable.
High-spend days (4%)	MAE = KES 12,041 on days above KES 10,000. The model cannot predict event spikes.
Dominant feature	rolling7_mean — predicts close to the recent weekly average.
Deployment gate	R² > 0.50. Current gap = +0.30 R² points (~31 months of additional data).

8.0 Recommendations & Conclusion

8.1 User Recommendations

Every figure below is calculated from the actual 24-month transaction history — not from external benchmarks or general guidelines.

Priority	Category	Action	Monthly Saving
Critical	M-Pesa Balance	Maintain a KES 5,000 minimum end-of-month buffer	Risk reduction
High	Construction	Hard monthly cap of KES 15,000; carry-over unused budget	KES 4,000–6,000
High	Friends & Family	Shift contributions to Week 2/3 to ease Week 1 cash pressure	KES 800–1,200
Medium	Dining	Cap at KES 2,000/mo — currently 2.1% vs 1.5% of income target	KES 600–900
Medium	Shopping	Pre-set KES 8,000 seasonal budget for December/January	KES 500–800

8.2 System Architecture & UI

A fully interactive browser-based dashboard was built alongside this notebook, presenting the engine's output to a non-technical user across eight screens — from the upload interface through merchant labelling, the main dashboard, category drill-downs, and the ML status panel.

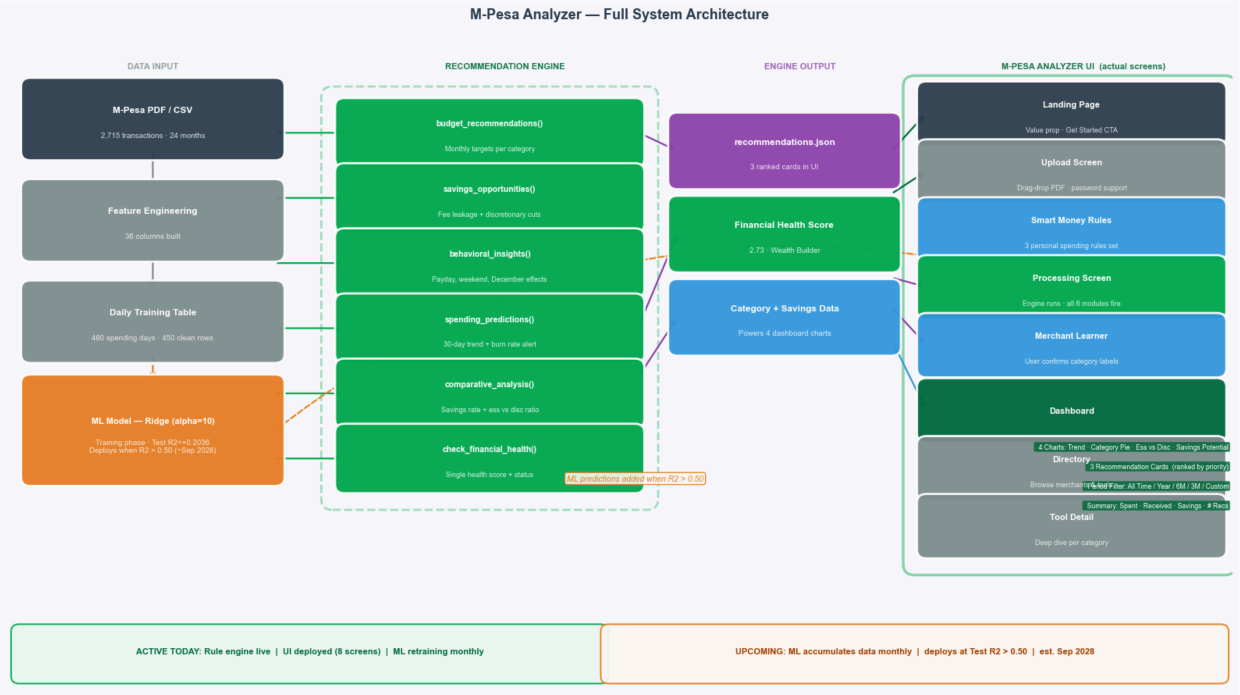

M-Pesa Analyzer — Full System Architecture
DATA INPUT
M-Pesa PDF / CSV
2,715 transactions · 24 months
Feature Engineering
36 columns built
Daily Training Table
480 spending days · 450 clean rows
ML Model — Ridge (alpha=10)
Training phase · Test R2=+0.2036
Deploys when R2 > 0.50 (~Sep 2028)
RECOMMENDATION ENGINE
budget_recommendations()
Monthly targets per category
savings_opportunities()
Fee leakage + discretionary cuts
behavioral_insights()
Payday, weekend, December effects
spending_predictions()
30-day trend + burn rate alert
comparative_analysis()
Savings rate + ess vs disc ratio
check_financial_health()
Single health score + status
ML predictions added when R2 > 0.50
ENGINE OUTPUT
recommendations.json
3 ranked cards in UI
Financial Health Score
2.73 · Wealth Builder
Category + Savings Data
Powers 4 dashboard charts
M-PESA ANALYZER UI (actual screens)
Landing Page
Value prop · Get Started CTA
Upload Screen
Drag-drop PDF · password support
Smart Money Rules
3 personal spending rules set
Processing Screen
Engine runs · all 6 modules fire
Merchant Learner
User confirms category labels
Dashboard
4 Charts: Trend · Category Pie · Ess vs Disc · Savings Potential
Directory
3 Recommendation Cards (ranked by priority)
Browse merchants & tools
Period Filter: All Time / Year / 6M / 3M / Custom
Summary: Spent · Received · Savings · # Recs
Tool Detail
Deep dive per category
ACTIVE TODAY: Rule engine live | UI deployed (8 screens) | ML retraining monthly
UPCOMING: ML accumulates data monthly | deploys at Test R2 > 0.50 | est. Sep 2028

Figure 20 — Full system architecture: data input layer → recommendation engine modules → engine output → UI screens. The ML model (orange) trains in the background with a dashed line showing its future connection to the dashboard when $R^2 > 0.50$.

8.3 Project Timeline & Path to ML Deployment

The timeline below maps the full project journey from data collection in February 2024 through to the estimated ML deployment in September 2028. Completed milestones include data preparation, feature engineering, EDA, the recommendation engine, and the UI. The path to deployment is driven by data accumulation — at approximately 20 new spending days per month, the model will reach the seasonal pattern threshold by August 2026, improve consistency by September 2027, and realistically clear the deployment gate of Test $R^2 > 0.50$ by September 2028.

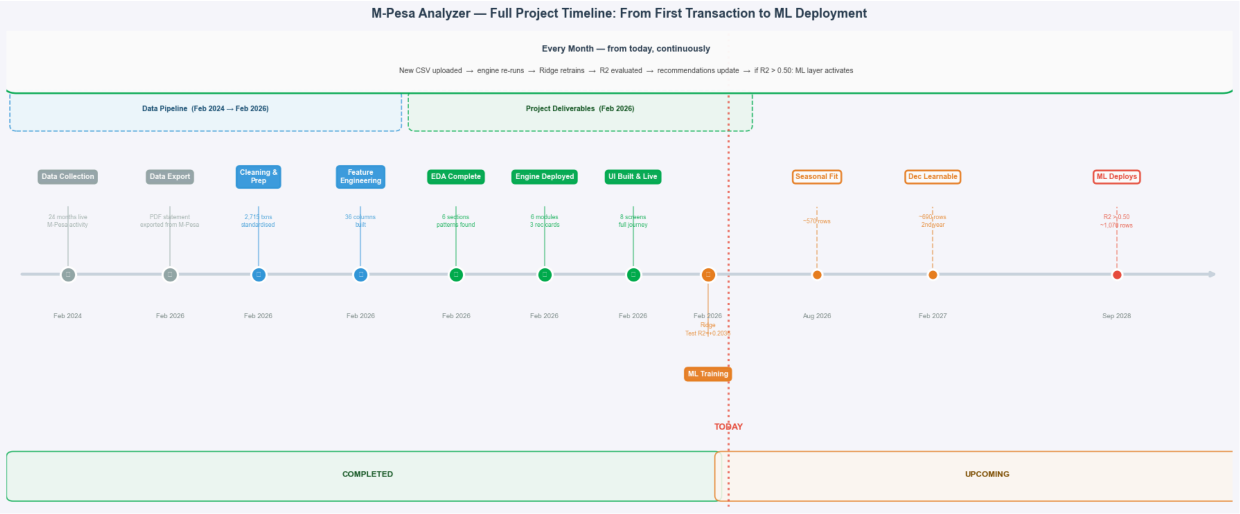


Figure 21 — Project timeline from Feb 2026 through estimated ML deployment in Sep 2028. Key milestones: ~570 rows in Aug 2026 (seasonal patterns emerge), ~730 rows in Sep 2027 (model consistency improves), ~1,000+ rows in Sep 2028 (deployment gate reached).

8.4 Conclusion

This project set out to answer a direct question: can a data-driven system analyze M-Pesa transaction history and deliver specific, reliable, actionable financial recommendations — and improve over time as more data arrives? The answer is yes, with clarity about what stage the system is at.

What This Project Achieved

The EDA across 2,715 transactions and 24 months revealed a 58.2% savings rate, a December spending spike of 3× the monthly average, a strong payday effect concentrated in Week 1, and a lag-1 autocorrelation of 0.01 — the most important statistical finding, as it determined the entire modelling approach. The recommendation engine's six modules produce 16 ranked, prioritized, category-specific recommendations grounded entirely in real data. The financial health score of 2.73/3.00 places this user in Wealth Builder status — the highest tier — with specific quantified actions available to improve further.

Three ML models were tested on a chronological 80/20 split. Only Ridge Regression generalized, achieving Test $R^2 = +0.2036$. The cause of the ceiling is clear: 480 spending days, CV = 1.73, near-zero autocorrelation. The data is not yet sufficient for a deployed ML predictor — and stating this plainly is more valuable than deploying a model that would produce wrong recommendations silently.

The Path Forward

- Continue monthly Ridge retraining as new M-Pesa PDFs are uploaded.
- Activate the ML prediction layer when rolling 3-month Test R^2 exceeds 0.50 (estimated Sep 2028).
- Add a proactive December alert (triggered in November) based on the observed 3× seasonal spend pattern.
- Investigate calendar-based external signals (public holidays, payroll cycles) to accelerate R^2 growth.

End of Report — M-Pesa Personal Finance Analysis & Insights