IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 11, November 2026

Spendwise: An Integrated Machine Learning Framework for Expense Categorization, Anomaly Detection, and Monthly Forecasting

Harshavardhan G[1], Deepak B T[2], Prawin Kumar C[3], Dr. Beenarani Manoj[4]

Department of Computer Science and Engineering, SRM Institute of Science and Technology, Vadapalani Campus, Chennai, India[1-4]

Abstract: Personal expense management involves organizing transactions, identifying unusual spending behavior, and estimating future expenditure. This paper presents SpendSmart, a web-based expense analytics prototype integrating three machine-learning capabilities: transaction categorization, anomaly detection, and monthly expense forecasting. The system employs term frequency–inverse document frequency (TF-IDF) with Logistic Regression for text-based categorization, Isolation Forest for anomaly detection, and Linear Regression for forecasting. A React and TypeScript frontend communicates with a Python FastAPI inference service through an Express.js backend. The system was evaluated using synthetic datasets containing 1,800 transactions, 1,500 categorization examples, and 600 anomaly-evaluation records. On the categorization test benchmark, the word-plus-character candidate achieved 93.59% accuracy and a six-category macro-F1 score of 96.54%; however, candidate rankings varied across validation and test data, so the production model was retained. Anomaly detection experiments revealed substantial false-positive challenges. In forecasting, a previous-complete-month baseline outperformed the production Linear Regression model on the evaluated validation and final-test periods. These findings emphasize benchmark design, uncertainty handling, chronological evaluation, and cautious model selection. Because all datasets are synthetic and the final forecasting test covers only one distinct month, the results do not establish real-world financial performance.

Keywords: Expense analytics; transaction categorization; anomaly detection; expense forecasting; machine learning; Isolation Forest; rolling-origin evaluation.

1. INTRODUCTION

Digital financial applications increasingly use automated data analysis to help users understand spending patterns. However, transaction descriptions can be inconsistent, unusual expenses may be difficult to identify using fixed rules, and future expenditure is inherently uncertain.

This paper investigates an integrated expense analytics prototype that combines three complementary machine-learning tasks within a single application. Unlike a system limited to displaying financial summaries, SpendSmart connects trained models to a web interface through an API-based architecture.

The research focuses on both implementation and evaluation. It examines whether lightweight machine-learning models can be integrated into a working application and how their observed performance changes when alternative models, thresholds, and evaluation strategies are considered.

The principal contributions are:

- An integrated web architecture connecting transaction categorization, anomaly detection, and expense forecasting.
- A transaction categorization evaluation comparing word-based and word-plus-character text representations.
- An anomaly detection investigation examining false positives and the effect of alternative thresholds and candidate features.
- A rolling-origin forecasting evaluation comparing Linear Regression with simple spending baselines.
- A documented model-selection process that retains production models when candidate evidence is insufficient or inconsistent.

The study is intended as an empirical prototype investigation, not as evidence of a validated commercial financial prediction system.

© IARJSET This work is licensed under a Creative Commons Attribution 4.0 International License

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 11, November 2026

2. REVIEW OF LITERATURE

Text classification commonly transforms documents into numerical representations before applying supervised learning. TF-IDF weighting emphasizes terms according to their distribution across a corpus and is widely used in text classification pipelines (Scikit-learn Developers, n.d.-a). SpendSmart applies this representation to transaction descriptions and uses Logistic Regression to predict expense categories.

Isolation Forest, introduced by Liu et al. (2008), identifies observations that are easier to isolate through randomized partitioning. It provides a computationally practical approach to anomaly detection. However, detecting statistical outliers is not equivalent to determining fraudulent intent. This distinction is particularly important in financial applications, where legitimate transactions may also be unusual.

Forecast evaluation requires respect for temporal ordering. Rolling-origin evaluation repeatedly generates forecasts using only information available at each forecast origin (Bergmeir et al., 2018). This approach is more appropriate for assessing temporal prediction than randomly mixing future and past observations. SpendSmart applies an expanded rolling-origin design and compares its forecasting model against simple baselines.
The present study brings these methods together in a single prototype and reports limitations observed during controlled synthetic-data experiments.

3. MATERIALS AND METHODS

3.1 System Architecture

SpendSmart follows a three-layer architecture.
A. Frontend layer: The user interface is implemented using React, TypeScript, and Vite. It provides the dashboard, model interaction controls, charts, navigation, and result presentation.
B. Backend layer: An Express.js service acts as a frontend-facing proxy. It validates requests, forwards approved requests to the Python service, enforces configured cross-origin restrictions, and handles upstream errors and timeouts.
C. Machine-learning layer: A Python FastAPI service exposes endpoints for model health, categorization, anomaly analysis, and forecasting.

The request flow is:

User interface → Express.js proxy → FastAPI service → model inference → response to the frontend.

The API interface includes /api/ml/health, /api/ml/category, /api/ml/anomaly, and /api/ml/forecast on the frontend-facing service.
The architecture separates user-interface concerns from model inference and allows the ML components to be tested independently of the frontend.

3.2 Dataset and Experimental Setup

3.2.1 Synthetic datasets

Three primary datasets were used, as summarized in Table 1.

Table 1. Synthetic datasets used in the study.

Dataset	Number of records	Purpose
Transaction dataset	1,800	Historical transaction and forecasting experiments
Categorization dataset	1,500	Supervised transaction classification
Anomaly evaluation dataset	600	Anomaly detection benchmarking

The transaction history covers 24 October 2025 through 23 October 2026. The datasets were synthetically generated and do not contain a representative sample of real customer financial behavior.
For categorization, a separate benchmark used 576 training examples, 84 validation examples, and 156 test examples. The benchmark contained six expense categories and an additional uncertainty label used for evaluation. Normalized transaction-description groups were separated to reduce overlap between splits.
For forecasting, the original evaluation included three monthly forecast origins. The expanded experiment used five origins per month, on days 5, 10, 15, 20, and 25. July and August provided ten validation origins, while September provided five final-test origins.

3.2.2 Evaluation metrics

Categorization was evaluated using accuracy, macro-F1, prediction coverage, and accuracy among accepted predictions. Macro-F1 helps characterize performance across classes rather than allowing larger classes to dominate the summary.
Anomaly detection was assessed using precision, recall, F1 score, false-positive rate, and confusion matrices.
Forecasting was evaluated using mean absolute error (MAE) and root mean squared error (RMSE):

$$\text{MAE} = 1/n \ \Sigma_{i=1}^{n} | y_i - \hat{y}_i | \quad (1)$$

$$\text{RMSE} = \sqrt{[\ 1/n \ \Sigma_{i=1}^{n} (y_i - \hat{y}_i)^2\ ]} \quad (2)$$

Here, y_i represents the observed monthly expenditure, $\hat{y}_i$ the predicted value, and n the number of evaluated observations. Lower MAE and RMSE indicate lower errors on the evaluated observations.

3.3 Machine-Learning Methods

3.3.1 Transaction categorization

Transaction descriptions are transformed into TF-IDF feature vectors. A Logistic Regression classifier then predicts the expense category.
The inference pipeline evaluates the prediction against a configured confidence threshold. If the prediction does not satisfy the threshold, the system can return an Uncertain label rather than forcing a category.
Two candidate representations were evaluated: word-based features and a combination of word- and character-level features. The candidate artifacts were isolated from the production model to avoid unintended changes during experimentation.

3.3.2 Anomaly detection

The production anomaly detector uses Isolation Forest (Scikit-learn Developers, n.d.-b). The algorithm identifies observations that are relatively easy to isolate in randomized tree partitions.
The evaluation also investigated alternative thresholds, a category-relative candidate, and an amount-only baseline. These comparisons were designed to determine whether candidate approaches improved the balance between detecting labelled anomalies and avoiding false positives.
The output is interpreted as a review signal. It does not establish that a transaction is fraudulent.

3.3.3 Expense forecasting

The production forecasting model uses Linear Regression (Scikit-learn Developers, n.d.-c) with time-dependent features derived from historical spending. These include prior-day, prior-week, and trailing-seven-day values.
Forecast generation uses recursive predictions for future daily values rather than actual future transactions. The inference pipeline excludes transactions occurring after the forecast origin before computing historical features.
The expanded experiment compared the saved Linear Regression model against a month-to-date run-rate baseline, a previous-complete-month baseline, a historical average of completed months, and a month-seasonality candidate.

4. RESULTS AND DISCUSSION

4.1 Categorization results

The held-out categorization benchmark produced the results shown in Table 2.

Table 2. Held-out categorization benchmark results.

Metric	Production	Word-only candidate	Word + character candidate
Accuracy	58.97%	89.10%	**93.59%**
Seven-label macro-F1	60.96%	87.66%	**90.54%**
Six-category macro-F1	67.23%	93.70%	**96.54%**
Coverage	69.23%	85.26%	**93.59%**
Accepted-prediction accuracy	78.70%	**97.74%**	95.89%

Note. The best value in each row is shown in bold.

The word-plus-character candidate achieved the highest accuracy, six-category macro-F1, and coverage on this test benchmark. However, the word-only candidate achieved higher accuracy among accepted predictions. Validation and held-out test rankings were not fully consistent.

Therefore, neither candidate was promoted to production. These metrics apply only to the synthetic benchmark and should not be interpreted as expected accuracy on real financial transactions.

4.2 Anomaly detection results

At its existing cutoff, the production model achieved 2.68% precision, 8.33% recall, and an F1 score of 0.041 on the reported synthetic test benchmark. Its false-positive rate was 26.27%.

A validation-selected production threshold increased recall to 70.83%, but precision remained 16.04% and the false-positive rate increased to 32.25%. The category-relative candidate achieved 58.33% recall and 13.08% precision, with a 33.70% false-positive rate.

The candidate also flagged 109 of 120 legitimate high-value transactions in the benchmark. These results demonstrate that increasing anomaly detection sensitivity can generate substantial review burden. No candidate was promoted.

4.3 Forecasting results

The expanded rolling-origin evaluation yielded the results shown in Table 3.

Table 3. Expanded rolling-origin forecasting results (₹).

Method	Validation MAE	Validation RMSE	September MAE	September RMSE
Saved Linear Regression	8,751.40	12,235.55	15,462.88	18,778.29
Month-to-date run-rate	22,634.02	29,790.34	11,559.51	13,932.42
Previous complete month	**6,625.49**	**7,362.83**	**8,153.93**	**8,153.93**
Historical average	11,726.59	11,989.53	16,460.42	16,460.42
Month-seasonality candidate	15,551.40	21,540.71	47,510.38	53,102.66

Note. MAE = mean absolute error; RMSE = root mean squared error. The lowest error in each column is shown in bold.

The previous-complete-month baseline achieved the lowest MAE and RMSE in both the validation and September final-test periods. The month-seasonality candidate performed poorly in the final test.

The September test includes five forecast origins, but all origins share the same underlying monthly actual total. They are therefore correlated observations, not five independent monthly test cases. The final test covers only one distinct month, limiting the strength of conclusions about performance across future months.

The production forecasting model was retained pending further evaluation on longer and more representative time series.

4.4 Integrated interpretation

The experiments reveal different challenges across the three ML tasks.

For transaction categorization, text representation and uncertainty handling affect performance. The candidate comparison suggests that character-level features may capture useful patterns in transaction descriptions, but inconsistent validation and test rankings caution against selecting a model from a single metric.

For anomaly detection, the main difficulty is the trade-off between sensitivity and false positives. A model that flags more unusual transactions can also produce a high volume of legitimate transactions requiring review. The synthetic benchmark further illustrates why unusual spending should not automatically be labelled fraudulent.

For forecasting, the previous-complete-month baseline outperformed the more complex production model in the reported evaluation. This result suggests that model complexity alone does not guarantee better predictions. However, because the evaluation covers limited synthetic history and only one distinct final-test month, further chronological testing is required before replacing the production model.

A notable implementation decision was to preserve the existing production artifacts while testing candidates independently. This separation made it possible to investigate improvements without treating experimental results as sufficient evidence for deployment.

5. LIMITATIONS AND THREATS TO VALIDITY

The study has several limitations.
First, all datasets are synthetic. Generated transaction descriptions and spending distributions may not reflect the diversity, noise, class imbalance, or temporal behavior of real financial records.
Second, the categorization benchmark is limited in size. Although candidate results are promising on the held-out split, they do not establish generalization to unseen real-world merchants.
Third, anomaly metrics depend on the synthetic definition of normal and unusual transactions. Real anomalies are contextual, and a legitimate transaction may differ substantially from historical behavior.
Fourth, the forecasting evaluation covers a limited time period. Multiple origins within the same month share the same monthly target, so the effective diversity of the final test is smaller than the number of origins suggests.
Finally, the application is a functional prototype rather than a complete financial platform. Persistent transaction storage, bank integration, and production-grade account management are not implemented. These limitations constrain both the practical scope and the claims that can be made from the results.

6. CONCLUSION AND FUTURE WORK

This paper presented SpendSmart, an integrated expense analytics prototype combining text-based transaction categorization, anomaly detection, and monthly expense forecasting within a React, Express.js, and FastAPI architecture. The experimental results showed that categorization candidates can substantially outperform the production model on a particular synthetic benchmark, while inconsistent validation and test rankings support a cautious model-selection strategy. Anomaly experiments highlighted the impact of false positives, and forecasting experiments showed that a previous-complete-month baseline outperformed the production Linear Regression model on the evaluated periods.
The results emphasize the importance of appropriate baselines, chronological evaluation, uncertainty handling, and transparent reporting of limitations. The findings do not establish real-world financial accuracy because the experiments use synthetic data and limited forecasting history.
Future work will focus on evaluating representative, appropriately anonymised and labelled transaction data; expanding chronological forecasting tests across multiple distinct months; improving anomaly detection; and assessing production alternatives through predefined acceptance criteria. Persistent storage and secure financial integrations may also be considered in a future system version.

7. SUMMARY AND IMPLICATIONS

SpendSmart demonstrates the integration of transaction categorization, anomaly detection, and expense forecasting within a web-based prototype. The experiments indicate that model evaluation must consider multiple metrics, uncertainty, false-positive burden, chronological validation, and simple baselines. The findings are limited to the synthetic datasets and evaluation periods described in this study; they should not be generalized to real financial behavior without further testing. These results imply that future development should prioritize representative labelled data, longer chronological evaluation across distinct months, and predefined criteria for promoting candidate models.

8. DECLARATIONS

8.1 Author contributions
Author contribution statements were not provided in the source manuscript. The authors must confirm and provide the individual contributions before submission.

8.2 Funding
Funding information was not provided in the source manuscript. The authors must confirm whether the study received any financial support.

8.3 Competing interests
A competing-interest statement was not provided in the source manuscript. The authors must confirm and disclose any relevant competing interests.

8.4 Use of AI-assisted tools
AI-assisted tools were used to support language editing and manuscript formatting. The authors are responsible for reviewing the manuscript and confirming the accuracy, integrity, and appropriateness of this disclosure before submission.

© IARJSET This work is licensed under a Creative Commons Attribution 4.0 International License

REFERENCES

[1]. Bergmeir, C., Hyndman, R. J., & Koo, B. (2018). A note on the validity of cross-validation for autoregressive time series prediction. *Computational Statistics & Data Analysis, 120*, 70–83. https://doi.org/10.1016/j.csda.2017.11.003

[2]. Liu, F. T., Ting, K. M., & Zhou, Z.-H. (2008). Isolation forest. In *Proceedings of the Eighth IEEE International Conference on Data Mining* (pp. 413–422). IEEE. https://doi.org/10.1109/ICDM.2008.17

[3]. Scikit-learn Developers. (n.d.-a). Feature extraction: TF-IDF term weighting. *Scikit-learn User Guide*. https://scikit-learn.org/stable/modules/feature_extraction.html

[4]. Scikit-learn Developers. (n.d.-b). IsolationForest. *Scikit-learn API Reference*. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.IsolationForest.html

[5]. Scikit-learn Developers. (n.d.-c). LinearRegression. *Scikit-learn API Reference*. https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html