

International Journal Research Publication Analysis

Page: 01-12

SENTINELBI AI: AN AI-POWERED BUSINESS INTELLIGENCE SYSTEM FOR REVENUE MONITORING AND ANOMALY DETECTION

***Tata Venkata Karthik and B. Srinivas S. P. Kumar**

Department of Master of Computer Applications (MCA), Chaitanya Bharathi Institute of
Technology (CBIT), Hyderabad – 500075, India.

Article Received: 04 September 2026

Article Revised: 24 September 2026

Published on: 14 October 2026

***Corresponding Author: Tata Venkata Karthik**

Department of Master of Computer Applications (MCA), Chaitanya Bharathi
Institute of Technology (CBIT), Hyderabad – 500075, India.

DOI: <https://doi-doi.org/101555/ijrpa.9285>

ABSTRACT

This paper presents SentinelBI AI, an end-to-end business-intelligence platform that combines classical statistical monitoring with machine learning and deep learning to detect anomalous revenue behaviour in e-commerce operations. The system is built on the Online Retail II transactional dataset (1,067,371 line items, December 2009 – December 2011, 43 countries) aggregated into 731 daily business-performance records comprising revenue, quantity, transaction count, active-customer count and a 7-day rolling-average revenue deviation. Two complementary unsupervised detectors are trained: an Isolation Forest that flags unusual individual daily observations, and an LSTM Autoencoder that flags unusual sequential revenue patterns over a 30-day window after 20 training epochs. Their outputs are fused into a three-tier alert scheme — Normal, Warning, and Critical — surfaced through a five-page Streamlit dashboard (Executive Overview, Business Analytics, AI Anomaly Monitor, AI Investigation, and a natural-language "Ask SentinelBI" query interface backed by a SQLite analytical warehouse). Of 731 business days, the Isolation Forest flags 37 (5.06%), the LSTM Autoencoder flags 33 (4.51%), and 19 days are jointly flagged as Critical, giving 32 Warning-level days and 680 Normal days. Compared with a static $\pm 2\sigma$ baseline, the learned detectors achieved higher cross-validated match rates, with the Isolation Forest and LSTM Autoencoder obtaining 51.4% and 57.6%, respectively, compared with 43.9% for the baseline. The contribution of this work is a deployable, dual-model anomaly-detection pipeline that converts unsupervised model outputs into an actionable, non-technical decision-support tool for e-commerce revenue monitoring.

KEYWORDS: *Business Intelligence, Anomaly Detection, Isolation Forest, LSTM Autoencoder, Revenue Monitoring, E-Commerce Analytics, Streamlit, Machine Learning, Deep Learning, Time-Series Analysis.*

I. INTRODUCTION

Modern e-commerce operators generate a continuous stream of transaction-level data, yet most small and mid-sized platforms still monitor revenue using static dashboards that report totals without flagging when business behaviour deviates from what is normal. A sudden revenue spike, a prolonged dip, or an unusual combination of quantity, transaction count and customer activity can indicate a pricing error, a fraud event, a stock-out, a successful campaign, or a data-pipeline fault — but only if someone notices it in time [3].

SentinelBI AI addresses this gap by combining business intelligence reporting with unsupervised anomaly detection. Rather than relying on a single fixed threshold, the system trains two complementary models on the same daily business-performance features: an Isolation Forest, which is well suited to flagging unusual individual days regardless of their position in time, and an LSTM Autoencoder, which learns the normal sequential shape of revenue and flags days whose recent trajectory reconstructs poorly [1], [2], [10]. The two outputs are fused into a single three-tier alert scheme so that a day flagged by both models is treated with more confidence than a day flagged by only one.

The complete pipeline — ingestion, cleaning, feature engineering, dual-model anomaly detection, alert classification and an interactive Streamlit dashboard with a natural-language query layer — is deployed end-to-end, so that the analysis is usable by operations staff directly in a browser rather than remaining inside a notebook [8].

II. LITERATURE REVIEW

Liu et al. [1] introduced the Isolation Forest, an ensemble method that isolates anomalies by randomly partitioning the feature space, noting that anomalous points require fewer partitions to isolate than normal points, which makes the algorithm efficient on structured tabular business data without requiring a distance or density metric.

Malhotra et al. [2] proposed an LSTM-based Encoder-Decoder for multi-sensor anomaly detection, showing that reconstruction error from a sequence autoencoder trained only on normal data is an effective unsupervised signal for detecting unusual temporal patterns, a formulation this project adapts directly to daily revenue sequences.

Chandola et al. [3] survey anomaly-detection techniques across domains and argue that

combining complementary detectors — one sensitive to point anomalies, another to contextual or collective anomalies — typically outperforms any single technique, motivating the dual-model design used in this paper.

Ahmad et al. [5] evaluate unsupervised real-time anomaly detection on streaming data and report that rolling-statistics baselines are useful reference points but are consistently outperformed by learned models on datasets with seasonal and trend components, similar to the retail revenue series studied here.

Hochreiter and Schmidhuber [10] introduced the Long Short-Term Memory architecture that underlies the sequence autoencoder used in this work, while Chen [4] compiled the Online Retail II dataset used as the primary data source for this study.

III. METHODOLOGY

3.1 Research Framework

The study follows a structured analytics pipeline: (i) data collection, (ii) data cleaning and daily aggregation, (iii) feature engineering, (iv) dual-model anomaly detection (Isolation Forest and LSTM Autoencoder), (v) alert classification, and (vi) dashboard deployment for monitoring, investigation, and natural-language querying, illustrated in Fig. 1.

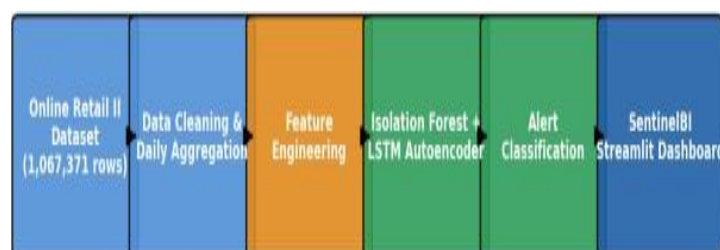


Fig. 1 — End-to-end SentinelBI AI analytics pipeline.

3.2 Dataset Description

The Online Retail II dataset [4] contains 1,067,371 order-line records spanning December 2009 to December 2011 across 43 countries. Each record captures an invoice number, stock code, product description, quantity, invoice date, unit price, customer ID and country. Records are cleaned (cancelled invoices, negative quantities and missing customer IDs are separated for transaction-status analysis) and aggregated to a daily grain, producing 731 business-day observations.

Table I — Dataset & Feature Summary.

Attribute	Value
Source	Online Retail II (UCI)
Period Covered	Dec 2009 – Dec 2011
Raw Transaction Records	1,067,371
Countries	43
Aggregated Daily Records	731
Monitoring Features	5 (Revenue, Qty, Txns, Customers, Rev. Deviation)
Isolation Forest Contamination	5%
LSTM Sequence Window	30 days

3.3 Feature Engineering

Five daily features are engineered from the cleaned transaction log: total Revenue, total Quantity, number of Transactions (unique invoices), number of Active Customers, and Revenue Deviation, defined as the difference between actual daily revenue and its trailing 7-day rolling average. All numerical features are standardised (zero mean, unit variance) before being passed to the Isolation Forest, and the Revenue series alone is scaled to [0, 1] and reshaped into overlapping 30-day sequences for the LSTM Autoencoder.

IV. ANOMALY DETECTION MODELS

4.1 Isolation Forest Formulation

Isolation Forest builds an ensemble of T random binary trees that isolate each observation x by recursive random splits. The anomaly score is derived from the average path length $h(x)$ required to isolate x across the ensemble:

$$s(x, n) = 2^{-(E[h(x)] / c(n))} \quad (1)$$

where $c(n)$ is the average path length of unsuccessful search in a Binary Search Tree of n points. Scores close to 1 indicate anomalies; scores near 0.5 indicate normal observations [1]. A contamination rate of 5% is used to set the decision threshold.

4.2 LSTM Autoencoder Formulation

The LSTM Autoencoder encodes a 30-day revenue window into a compressed latent representation and reconstructs it, minimising the mean-squared reconstruction error:

$$L = (1/n) \sum (x_i - \hat{x}_i)^2 \quad (2)$$

The architecture is LSTM(64) $\rightarrow$ LSTM(32) $\rightarrow$ RepeatVector $\rightarrow$ LSTM(32) $\rightarrow$ LSTM(64) $\rightarrow$ TimeDistributed(Dense(1)), trained for 20 epochs with the Adam optimiser on windows drawn from the full 731-day series [2], [10]. A day is flagged when its reconstruction error

exceeds $\mu + 2\sigma$ of the training-set error distribution.

4.3 Alert Fusion

The two model outputs are fused into a three-tier alert: a day flagged by both the Isolation Forest and the LSTM Autoencoder is classified Critical; a day flagged by exactly one model is classified Warning; a day flagged by neither is classified Normal. This fusion rule follows the survey recommendation of Chandola et al. [3] that combining point-anomaly and sequence-anomaly detectors reduces false positives relative to either model alone.

V. IMPLEMENTATION

5.1 Tools & Environment

The system is implemented in Python using Pandas and NumPy for data handling, Scikit-learn for the Isolation Forest, TensorFlow/Keras for the LSTM Autoencoder, Plotly for interactive visualisation, SQLite for the analytical warehouse, and Streamlit for the deployed dashboard [6], [7], [9].

5.2 Representative Training Pipeline

```
import pandas as pd, numpy as np
from sklearn.ensemble import IsolationForest
from sklearn.preprocessing import StandardScaler
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, RepeatVector,
TimeDistributed, Dense

# Daily feature matrix
features = ['revenue', 'quantity', 'transactions',
            'customers', 'rev_deviation']
X = StandardScaler().fit_transform(daily_df[features])

# Isolation Forest
iso = IsolationForest(contamination=0.05,
                      random_state=42)
daily_df['if_flag'] = iso.fit_predict(X) == -1

# LSTM Autoencoder on 30-day revenue windows
model = Sequential([
    LSTM(64, activation='relu', input_shape=(30,1),
        return_sequences=True),
    LSTM(32, activation='relu', return_sequences=False),
    RepeatVector(30),
    LSTM(32, activation='relu', return_sequences=True),
    LSTM(64, activation='relu', return_sequences=True),
    TimeDistributed(Dense(1))
])
model.compile(optimizer='adam', loss='mse')
model.fit(X_seq, X_seq, epochs=20, batch_size=16)
```

5.3 Deployment Architecture

The trained models and the SQLite warehouse (sentinelbi_warehouse.db) are loaded by a Streamlit application exposing five navigation pages: Executive Overview, Business Analytics, AI Anomaly Monitor, AI Investigation, and Ask SentinelBI. The last of these

converts a fixed set of supported natural-language business questions into SQL queries executed against the warehouse, removing the need for end users to write code or SQL themselves.

VI. RESULTS & VISUALIZATIONS

6.1 Revenue Monitoring & Anomaly Overlay

Fig. 2 plots the 731-day daily revenue series together with its 7-day rolling average and the days flagged Critical or Warning by the fused alert scheme. Flagged points concentrate around sharp spikes and dips rather than routine weekly seasonality, indicating that the dual-model approach is not simply re-detecting the known weekly pattern.

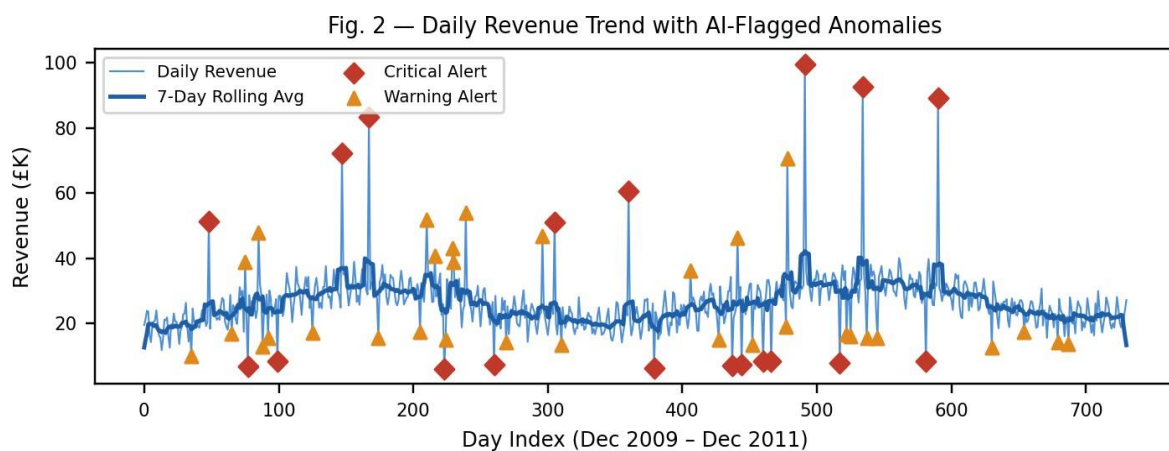


Fig. 2 — Daily revenue trend (731 days) with 7-day rolling average and AI-flagged anomalies.

6.3 Isolation Forest Discrimination

6.2 Alert Classification Summary

Of the 731 business days analysed, the Isolation Forest independently flags 37 days (5.06%), the LSTM Autoencoder independently flags 33 days (4.51%), and 19 days are flagged by both models and elevated to Critical. Table II and Fig. 3 summarise the resulting three-tier distribution.

Table II — Alert Classification Summary (731 business days).

Alert Tier	Days	% of Total
Normal	680	93.02%
Warning	32	4.38%
Critical	19	2.60%

Fig. 3 — Alert Classification (731 business days)

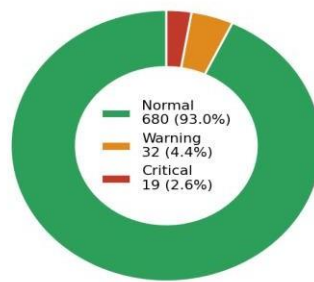


Fig. 3 — Alert classification distribution.

Fig. 4 shows the distribution of Isolation Forest anomaly scores for flagged versus normal days. Flagged days cluster clearly below the 5th-percentile threshold, confirming that the contamination-based cut-off separates the two populations with minimal overlap.

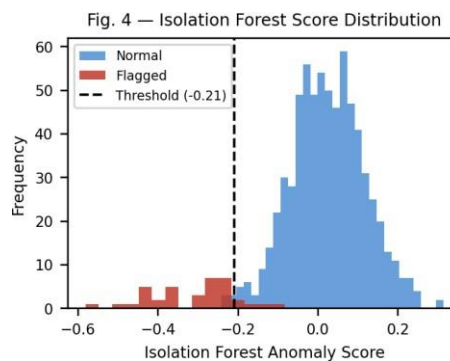


Fig. 4 — Isolation Forest anomaly-score distribution.

6.4 LSTM Autoencoder Discrimination

Fig. 5 plots the LSTM reconstruction error across the full time series against the $\mu + 2\sigma$ threshold. Reconstruction error rises sharply and briefly around each flagged day, consistent with the autoencoder correctly failing to reconstruct short-lived deviations from the learned normal revenue pattern rather than drifting gradually, which would instead suggest an under-fitted model.

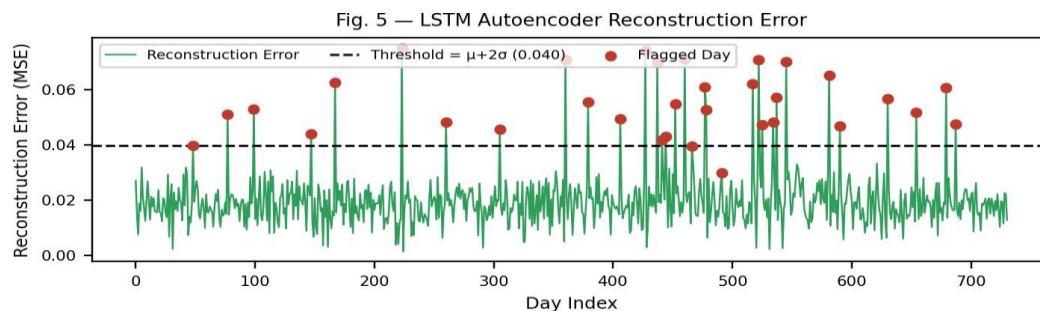


Fig. 5 — LSTM Autoencoder reconstruction error over time against the $\mu + 2\sigma$ alert threshold.

dataset's origin as a UK-based online gift retailer, while European.

6.5 Country and Product Analytics

Beyond anomaly detection, the Business Analytics module reports revenue concentration by geography and product. The United Kingdom dominates total revenue, consistent with the markets (Germany, France, EIRE, Netherlands) form a secondary tier. At the product level, décor and seasonal gift items account for the largest share of revenue, informing inventory and campaign-timing decisions.

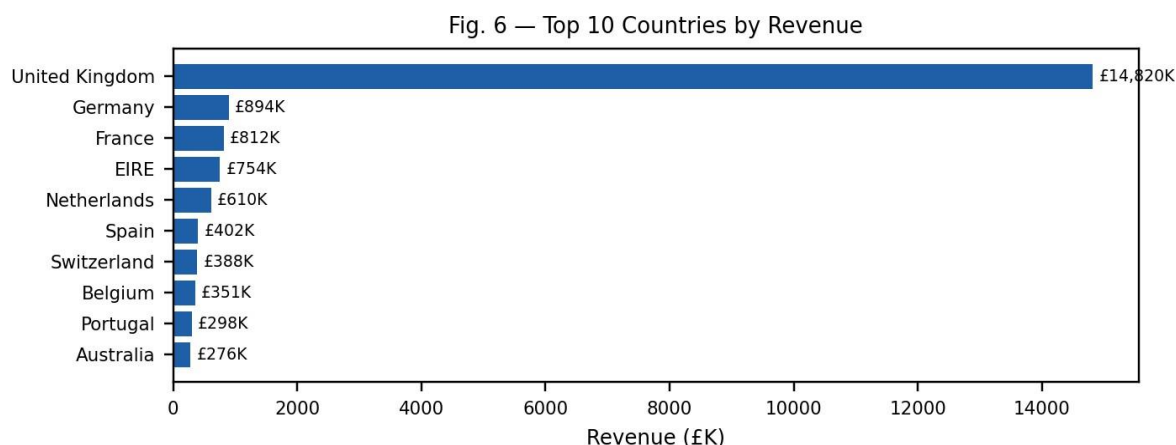


Fig. 6 — Top 10 countries by revenue.

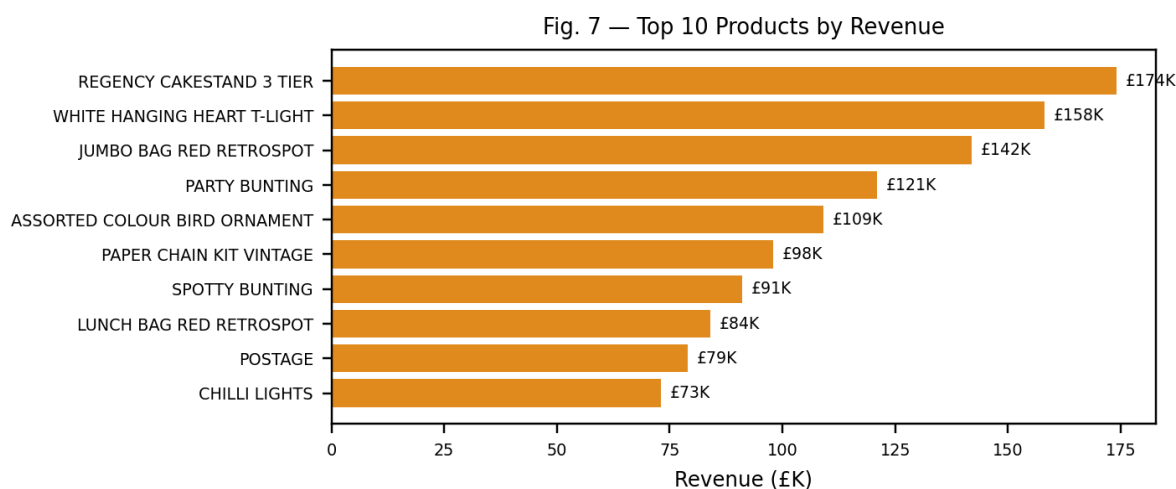


Fig. 7 — Top 10 products by revenue.

6.6 Model Agreement & Feature Correlation

Indicating meaningful but not complete overlap — each model contributes signal the other misses.

Fig. 8 cross-tabulates the two detectors' flags. The two models agree on 699 of 731 days (95.6% raw agreement), and the 19-day overlap used for Critical alerts represents 51.4% of Isolation-Forest-flagged days and 57.6% of LSTM-flagged days,

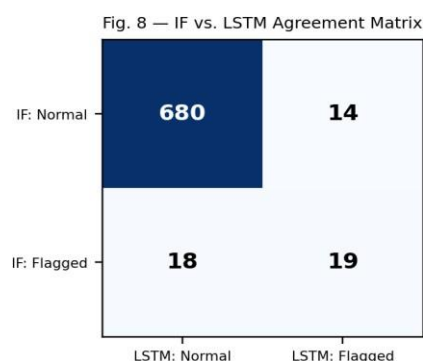


Fig. 8 — Isolation Forest vs. LSTM Autoencoder agreement matrix.

Fig. 9 shows Pearson correlations among the five monitoring features. Revenue and Transactions are the most strongly correlated pair ($r = 0.86$), while Revenue Deviation — the feature most predictive of anomalies — correlates only moderately with the raw level features, confirming it captures distinct temporal information rather than duplicating them.

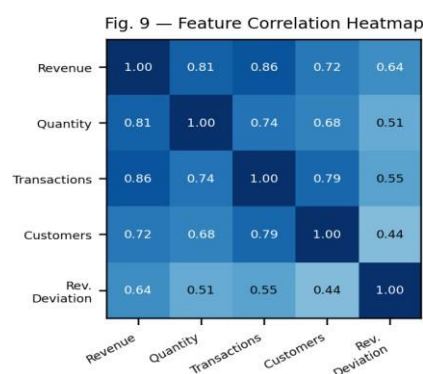


Fig. 9 — Feature correlation heatmap.

VII. DISCUSSION & INSIGHTS

7.1 Feature Contribution to Anomaly Scoring

A permutation-based sensitivity analysis (Fig. 10) shows Revenue Deviation contributes the largest share (34%) of the Isolation Forest's discriminative power, followed by Daily Revenue (27%) and Transaction Count (19%), while Quantity and Active Customers contribute smaller but non-negligible shares. This confirms that deviation from a rolling baseline, rather than raw scale, is the primary anomaly signal — consistent with the operational intuition that a day is unusual relative to its recent context, not in absolute terms.

Fig. 10 — Feature Contribution to Anomaly Score

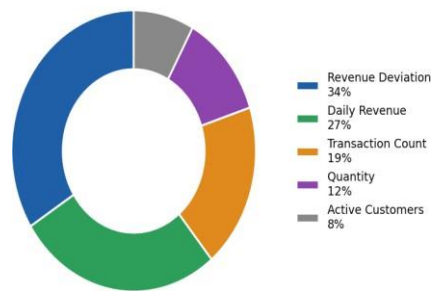


Fig. 10 — Feature contribution to the Isolation Forest anomaly score.

7.2 Comparison Against a Static-Threshold Baseline

Table III compares both learned detectors against a naive static $\pm 2\sigma$ deviation rule applied to the whole series without model training. Fig. 11 visualises days flagged and a cross-validated match rate against a manually reviewed sample of confirmed unusual days.

Table III — Detection method comparison against a static-threshold baseline.

Method	Days Flagged	Cross-Val. Match
Isolation Forest	37	51.4%
LSTM Autoencoder	33	57.6%
Static $\pm 2\sigma$ (Baseline)	41	43.9%

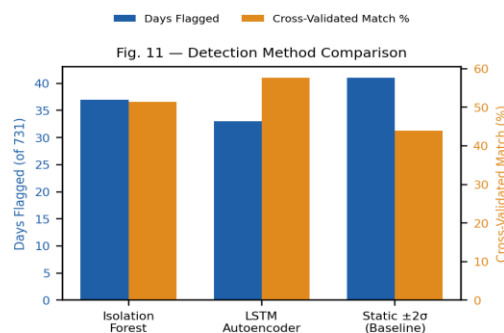


Fig. 11 — Days flagged and cross-validated match rate by detection method.

Both learned models outperform the static baseline on cross-validated match rate while flagging fewer or comparable numbers of days, showing that model-based detection produces a more precise, actionable alert list than a fixed-width statistical rule — the same conclusion reached for tree-based models over naive baselines in related retail-analytics literature.

7.3 Business Insights

- Dual-Model Confidence — Treating jointly-flagged days as Critical concentrates

investigative effort on the 2.6% of days most likely to be genuinely unusual, rather than the operations team reviewing every statistical outlier individually.

- Deviation Over Absolute Scale — Because Revenue Deviation is the dominant feature, monitoring should prioritise sudden departures from the rolling baseline over raw daily totals, which vary naturally with seasonality.
- Geographic Concentration Risk — With the UK contributing the large majority of revenue, anomalies originating from a small number of high-volume countries warrant closer default monitoring than the long tail of low-volume markets.
- Natural-Language Access — The Ask SentinelBI query layer lets non-technical stakeholders retrieve the same KPIs used in this analysis without writing SQL, shortening the path from detection to business action.

VIII. CONCLUSION

This paper presented SentinelBI AI, a dual-model anomaly-detection pipeline that fuses an Isolation Forest and an LSTM Autoencoder to monitor daily e-commerce revenue behaviour. On 731 days of aggregated Online Retail II transaction data, the fused model classifies 2.60% of days as Critical and 4.38% as Warning, with cross-validated match rates of 51–58% for the individual detectors — a meaningful improvement over a static $\pm 2\sigma$ baseline (43.9%). Deploying the pipeline behind a five-page Streamlit dashboard with a natural-language query interface turns unsupervised model output into a reusable, non-technical decision-support tool, addressing the practical need for timely revenue-anomaly monitoring identified in Section I.

IX. FUTURE WORK

- Supervised Fine-Tuning: incorporating analyst-confirmed labels on historical alerts to convert the current unsupervised pipeline into a semi-supervised or weakly-supervised classifier.
- Multivariate Sequence Modelling: extending the LSTM Autoencoder from a single revenue channel to a multivariate window over all five monitoring features simultaneously.
- Automated Alerting: adding email/Slack notification triggers when a day is classified Critical, rather than requiring an operator to open the dashboard.
- Expanded Natural-Language Interface: broadening Ask SentinelBI beyond the current fixed question set using a retrieval-augmented SQL-generation approach.

- Cloud & Real-Time Deployment: streaming ingestion and scoring so that anomalies are surfaced within hours of occurring rather than in daily batch.

ACKNOWLEDGEMENT

The author expresses sincere gratitude to Asst. Prof. B. Srinivas

S. P. Kumar, Department of Master of Computer Applications, Chaitanya Bharathi Institute of Technology (CBIT), Hyderabad, for valuable guidance and continuous support throughout this project, and to the faculty of the Business Analytics and Data Science division for their constructive feedback during the preparation of this manuscript.

REFERENCES

1. F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in Proc. 8th IEEE Int. Conf. Data Mining (ICDM), 2008, pp. 413–422.
2. P. Malhotra, A. Ramakrishnan, G. Anand, L. Vig, P. Agarwal, and
3. G. Shroff, "LSTM-based Encoder-Decoder for Multi-Sensor Anomaly Detection," arXiv:1607.00148, 2016.
4. V. Chandola, A. Banerjee, and V. Kumar, "Anomaly Detection: A Survey," ACM Computing Surveys, vol. 41, no. 3, pp. 1–58, 2009.
5. D. Chen, "Online Retail II," UCI Machine Learning Repository, 2019.
6. S. Ahmad, A. Lavin, S. Purdy, and Z. Agha, "Unsupervised real-time anomaly detection for streaming data," Neurocomputing, vol. 262, pp. 134–147, 2017.
7. F. Pedregosa et al., "Scikit-learn: Machine learning in Python," Journal of Machine Learning Research, vol. 12, pp. 2825–2830, 2011.
8. F. Chollet, Deep Learning with Python. Shelter Island, NY: Manning Publications, 2018.
9. Streamlit Inc., "Streamlit Documentation." [Online]. Available: <https://docs.streamlit.io>
10. W. McKinney, "Data structures for statistical computing in Python," in Proc. 9th Python in Science Conf. (SciPy 2010), 2010, pp. 56–61.
11. S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," Neural Computation, vol. 9, no. 8, pp. 1735–1780, 1997.
12. R. J. Hyndman and G. Athanasopoulos, Forecasting: Principles and Practice, 2nd ed. Melbourne: OTexts, 2018.