

---

Title: "Retail AI Analytics & Recruitment Assistant — Whitepaper"

Author: "Sri Charan Konidina"

Date: 2025-11-18

---

## # Retail AI Analytics & Recruitment Assistant

### \*\*Abstract\*\*

This whitepaper describes the design, implementation, and evaluation of two applied AI systems: (1) a retail sales anomaly detection proof-of-concept (PoC) and streaming simulation, and (2) an AI recruitment assistant that automates resume scoring, comparison, shortlisting, and analytics dashboards. The goal is applied impact: provide lightweight, reproducible systems that businesses can use for fraud detection in retail and data-driven hiring assistance. We present methodology, datasets, implementation details, quantitative results, a polished Streamlit UI, and guidance for deployment.

---

### ## 1. Introduction

Organizations increasingly rely on data and AI to detect operational anomalies and streamline hiring. Small and medium retailers need low-cost methods to detect rare unusual transactions (refund fraud, data entry mistakes) while HR teams need objective, repeatable screening tools for high-volume hiring. This project demonstrates two practical systems:

- **Retail Anomaly Detection PoC**: A minimal pipeline using Isolation Forests to flag unusual daily sales and a streaming simulation to demonstrate near-real-time detection.
- **AI Recruitment Assistant**: A suite of Streamlit apps to score resumes against job descriptions (GPT-based ranking), compare candidates, shortlist by threshold, provide AI reasoning and exportable reports, and offer recruitment analytics.

This whitepaper consolidates the engineering steps, provides reproducible code links, and presents evaluation and deployment suggestions.

---

### ## 2. Related Work (short)

- Anomaly detection in retail and finance: unsupervised techniques such as Isolation Forest, Local Outlier Factor (LOF), and autoencoders have been used to detect fraud and errors.
  - Automated resume screening: recent systems use NLP embeddings, supervised ranking, or LLMs to score and summarize candidate fit.
  - Our contribution lies in integrating simple, interpretable ML techniques and LLM-based reasoning into an end-to-end, documented pipeline that is reproducible and lightweight.
- 

### ## 3. Data Sources

#### ### 3.1 Retail dataset (synthetic + Kaggle)

For experiments we used:

- A public supermarket sales dataset (Kaggle) for exploratory analysis and feature understanding.
- A small synthetic time series (25 days) for PoC and demo reproducibility.

Dataset fields used: `day`, `sales` (daily total), and categorical attributes for simulated scenarios.

### ### 3.2 Resume dataset

- Small collection of synthetic/resumed sample texts (plain text), created to test the recruitment assistant. Each resume text contains skill tokens (e.g., Python, SQL, NLP) to evaluate scoring and keyword extraction.
- 

## ## 4. Methods & Implementation

### ### 4.1 Retail anomaly detection

- \*\*Model\*\*: `IsolationForest` from scikit-learn. Contamination set conservatively (e.g., 0.05–0.1).
- \*\*Pipeline\*\*: Load CSV → basic cleaning → fit model on `sales` → predict anomalies (`-1` labels) → visualize.
- \*\*Visualization\*\*: matplotlib / Plotly interactive plots embedded into notebooks and Streamlit.

\*\*PoC steps\*\*:

- Minimal notebook (`retail\_anomaly\_demo.ipynb`) with sample data.
- Streamlit dashboard demonstrating anomaly flags and upload capability.

### ### 4.2 Recruitment assistant (NLP + LLM)

- \*\*Skill extraction & scoring\*\*: spaCy for lightweight tokenization and keyword checks; scoring via rule-based presence matching and later LLM scoring (OpenAI) to provide nuanced fit (score with reasoning).
- \*\*Comparison & heatmap\*\*: Multiple resumes compared with pre-defined skill lists; heatmap visualization with seaborn.
- \*\*Shortlisting\*\*: Threshold filtering (user adjustable). Scores kept both as integer numeric for logic and as `XX/100` strings for display.
- \*\*Explainability\*\*: LLM provides 2–3 sentence rationale per resume and overall recruiter summary.

\*\*Deployment\*\*:

- Streamlit apps for interactive UIs. Files include:

- `resume\_analyzer\_enhanced.py`
- `resume\_comparison\_dashboard.py`
- `resume\_summary\_generator.py`
- `ai\_shortlisting\_assistant.py`
- `ai\_recruitment\_dashboard.py` (analytics)

## ## 5. Experiments & Results

### ### 5.1 Retail PoC result (example)

- Running the IsolationForest demo on a 25-day synthetic series flagged 3 anomalous days (e.g., extremely high sales days and a mid-range spike). Visual inspection indicates correct anomaly localization for demonstrative scenarios.

\_Example output (sample):\_

Detected anomalies:

day sales anomaly

14 15 300 -1

18 19 1005 -1

24 25 500 -1 (Full notebook and screenshots archived in repo.)

### ### 5.2 Recruitment assistant demonstration

- For 3 sample resumes and a job description for "Data Analyst", the LLM scoring produced interpretable scores and reasoning. The system ranked candidates, showed heatmap of skill coverage, and created a shortlisting CSV and an AI Hiring Summary.

\_Example ranked output:\_

- Resume 1 — 92/100 — strong alignment with Python, SQL, Tableau
  - Resume 3 — 78/100 — Power BI and analyst background
  - Resume 2 — 45/100 — non-aligned software engineering profile
- 

## ## 6. Discussion

### ### 6.1 Strengths

- **Reproducibility**: All code and notebooks are public on GitHub.
- **Practicality**: Uses minimal dependencies; can run locally on a laptop.
- **Explainability**: LLM output paired with rule-based checks provides both scores and human-readable reasoning.

### ### 6.2 Limitations

- **LLM costs & quota**: OpenAI usage incurs token costs and may hit quotas — we include ability to swap in smaller open-source LLMs or offline scoring rules.
- **Dataset scale**: Current experiments use small/synthetic examples. For production, larger labeled datasets and human-in-the-loop validation are needed.

- **Bias & fairness**: Automated ranking must be audited in real deployments for bias; treat outputs as decision-support, not final decisions.
- 

## ## 7. Reproducibility & How to Run

### ### 7.1. Clone repo:

- ``bash
- git clone <https://github.com/konidinasricharan/retail-anomaly-poc.git>
- cd retail-anomaly-poc

### ### 7.2. Install dependencies (recommended to use a virtual env):

- python -m venv venv
- venv\Scripts\activate
- pip install -r requirements.txt
- # if no requirements file, install: streamlit pandas scikit-learn matplotlib seaborn spacy openai python-dotenv fpdf

### ### 7.3. Run notebooks or Streamlit apps:

#### # Notebook

- python -m notebook
- # Or run Streamlit shortlisting assistant
- streamlit run ai\_shortlisting\_assistant.py

### 7.4. Create evidence:

- Save screenshots of app UI, charts, and summary.
  - Export shortlisted\_candidates.csv and ai\_hiring\_summary.txt.
  - Add to your Evidence Google Drive folder.
- 

## ## 8. Next Steps & Future Work

- Expand datasets and evaluate on real retail transaction logs and real-world resumes.
  - Implement privacy-preserving resume parsing and anonymization.
  - Integrate controlled A/B testing with recruiters for real feedback.
  - Replace closed LLMs with fine-tuned local models for lower cost and privacy.
- 

## ## 9. Conclusion

This project demonstrates applied AI techniques for two common business needs: anomaly detection in retail and automated recruitment assistance. The systems are engineered to be reproducible, explainable, and accessible, turning prototypes into deployable tools that can be documented and presented

---

### ### 10. References & Resources

- Liu, F. et al. (2019). Isolation Forest — Anomaly detection. scikit-learn docs.
  - spaCy: Industrial-strength NLP in Python. <https://spacy.io>
  - OpenAI API documentation: <https://platform.openai.com/docs>
  - Streamlit documentation: <https://docs.streamlit.io>
- 

### ### 11. Appendix

- Links to GitHub repo: <https://github.com/konidinasricharan/retail-anomaly-poc>
- Notebooks and Streamlit apps (list of filenames) included in the repo.