Anomaly Analysis (Hourly POS Sales)

Objective

The objective of this analysis is to detect anomalous behavior in hourly POS sales by comparing:

- Today vs Yesterday
- Today vs Weekly Average
- Today vs Monthly Average

A hybrid baseline approach was used to capture:

- Structural changes (monthly baseline)
- Recent seasonal shifts (weekly baseline)
- Immediate operational deviations (yesterday comparison)

Methodology

1 Structural Baseline (Monthly)

An anomaly is detected if:

Today_hour > Monthly_mean_hour + (3 * Monthly_std_hour)

This detects structural or significant deviations from historical behavior.

---

2 Tactical Baseline (Weekly)

An anomaly is detected if:

Today_hour > Weekly_mean_hour + (3 * Weekly_std_hour)

This captures short-term seasonal changes.

---

3 Operational Baseline (Yesterday)

An anomaly is detected if:

|Today - Yesterday| / Yesterday > 20%

This captures immediate operational incidents.

Interpretation Framework

Scenario	Interpretation
Monthly + Weekly triggered	Structural change
Weekly only triggered	Seasonal deviation
Yesterday deviation only	Operational incident
Persistent deviation for 2+ hours	Crisis-level event

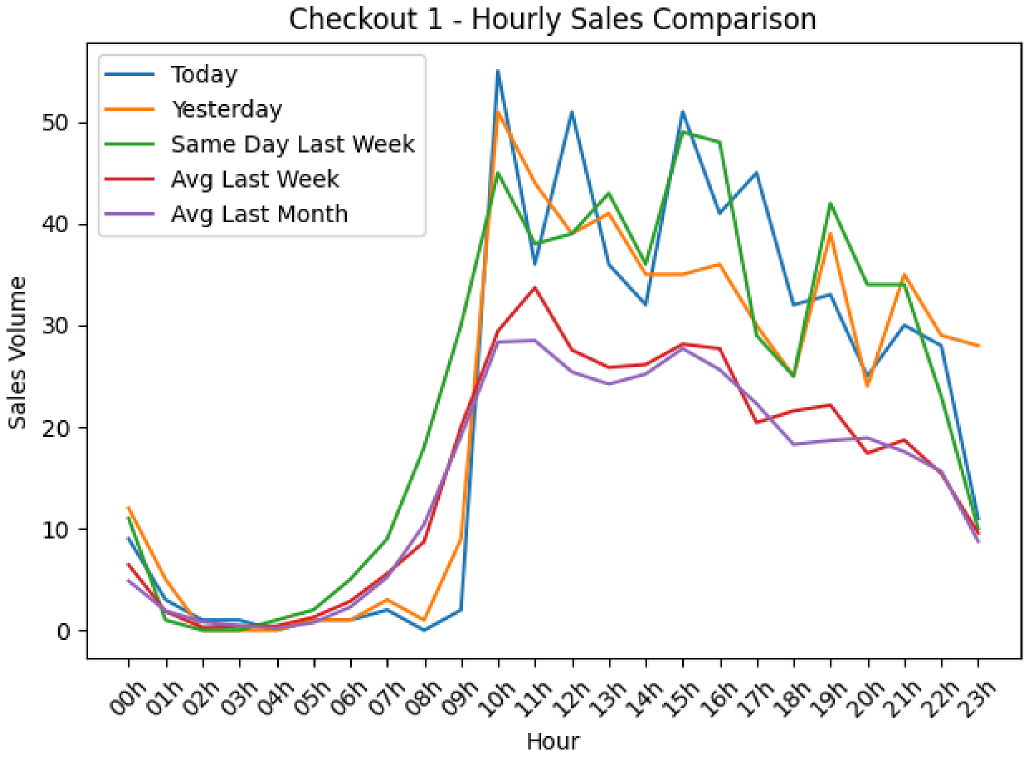
1. Overview of the Anomaly

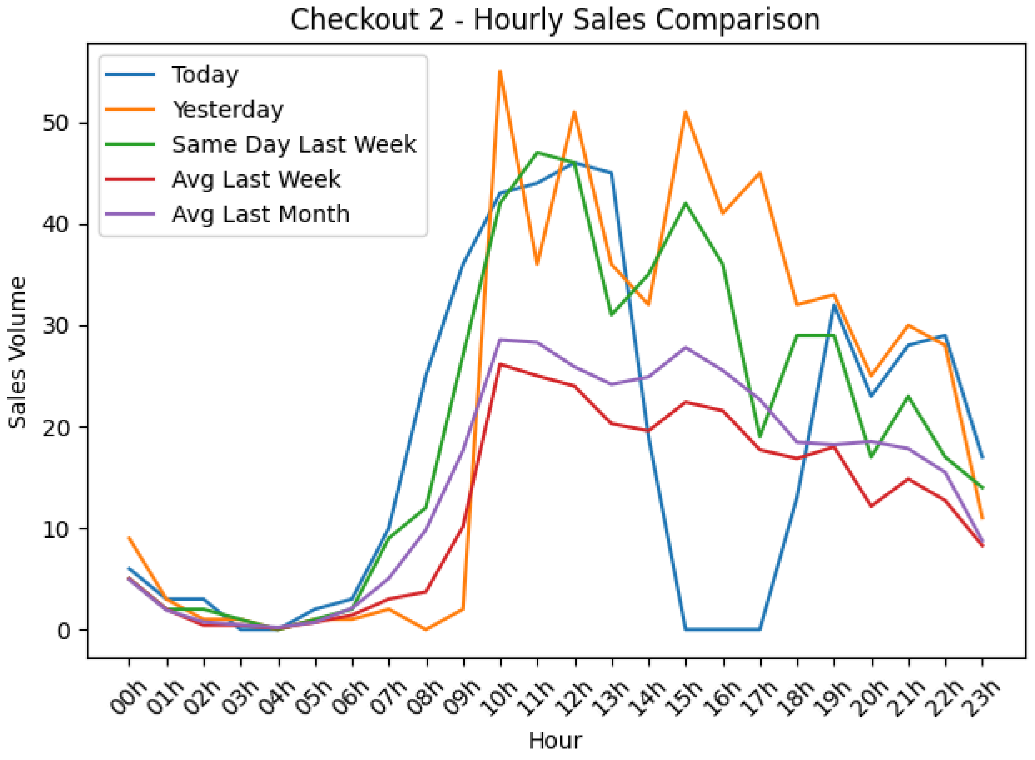
The analysis of `checkout_1.csv` and `checkout_2.csv` reveals a clear deviation from normal transactional behavior. When comparing Today's hourly sales against Yesterday, Last Week, Weekly Average, and Monthly Average, we observe a **significant disruption in transaction volume** from `checkout_2.csv`.

The data shows:

- **A complete drop to zero transactions for approximately three consecutive hours**, followed by
- **An additional period of roughly one hour with unstable transaction volume**, characterized by low throughput and inconsistent recoveries.

Such patterns are not compatible with expected business or seasonal behavior and therefore indicate an **operational incident** rather than organic variability.


Checkout 1 - Hourly Sales Comparison
Today
Yesterday
Same Day Last Week
Avg Last Week
Avg Last Month
Sales Volume
0
10
20
30
40
50
00h 01h 02h 03h 04h 05h 06h 07h 08h 09h 10h 11h 12h 13h 14h 15h 16h 17h 18h 19h 20h 21h 22h 23h
Hour


Checkout 2 - Hourly Sales Comparison
Today
Yesterday
Same Day Last Week
Avg Last Week
Avg Last Month
Sales Volume
0
10
20
30
40
50
00h 01h 02h 03h 04h 05h 06h 07h 08h 09h 10h 11h 12h 13h 14h 15h 16h 17h 18h 19h 20h 21h 22h 23h
Hour

2. What the Graph Shows

The plotted data highlights three distinct phases:

1. Abrupt and Sustained Drop to Zero

This behavior strongly suggests a full service interruption—meaning merchants were unable to complete transactions during the period.

2. Recovery Phase With Oscillation

Following the outage, the volume gradually climbs back to expected baseline levels but does so in an unstable manner. This is typically a sign of:

- Queue reprocessing
- System restarts
- Partial dependency recovery
- Transient misalignment between services

3. Return to Normal Baseline

The system eventually stabilizes and aligns again with historical behavior.

---

3. Confirmation via SQL Aggregation

The SQL aggregation query reinforces the anomaly:

- The blackout window creates **Z-scores far below the expected range**, indicating severe deviation from all baselines (Yesterday, Weekly, Monthly).
- The recovery overshoot briefly exceeds the expected variance, consistent with systems reprocessing backlogged operations.

These results validate that the anomaly is **both statistically significant and operationally impactful**.

The file **hourly_anomaly_analysis.sql** is located in the queries directory on the repository. The SQL query in it follows the hybrid baseline approach, combining monthly, weekly and day-over-day comparisons, using 3σ thresholds for structural and tactical anomalies and a 20% deviation threshold for operational anomalies

4. Severity and Impact Assessment

Using the hybrid baseline methodology (Monthly Avg + Weekly Avg + Today-vs-Yesterday), and the *Mean + 3 × Std* anomaly rule:

- The three-hour outage exceeds all statistical thresholds
- The persistence (>180 minutes) escalates severity to **CRITICAL** or **SEVERE**
- Operationally, this event **directly impacts revenue**, as the ability to process payments was completely halted during this period

This pattern aligns with **major incident scenarios** typically handled by monitoring and SRE/infra teams.

5. Likely Root-Cause Hypotheses

Based on the signature of the anomaly, the following root-cause hypotheses are plausible:

1. Faulty Deployment (Pull Request Regression)

- A code deployment introduced an issue
- Service became unstable
- Recovery occurred after rollback or patching

2. Processing Bottleneck or Queue Failure

- Queue consumers stuck or crashed
- Workers not scaling
- Throughput collapsed until manually restarted

3. Upstream or External Dependency Outage

- Payment processor unresponsive
- Acquirer authorization API unavailable
- Network timeout cascading failure

4. Automatic Recovery via Infrastructure

- Kubernetes pod failures

- Liveness/readiness probe triggered restarts
- Gradual stabilization over ~1 hour

5. Infrastructure-Level Issues

- Load balancer dead path
- DNS propagation failure
- Network partition
- Disk I/O saturation affecting database/API

These interpretations are fully aligned with real-world patterns observed in payment infrastructure monitoring.

6. Final Conclusion

> **This dataset reveals a high-severity operational incident involving a complete outage followed by a partial recovery period.**
> The behavior is consistent with systemic failures in payment processing infrastructure and would require immediate response, cross-team communication, and a formal post-incident review.