Table of Contents

# Log Data Mining & Drift Detection: Comprehensive Project Report

## 1. Introduction

Modern distributed systems generate massive volumes of logs across diverse components. Detecting subtle changes in system behavior—known as reliability drift—is critical for early warning of incidents and maintaining system health. This project explores unsupervised methods for mining, analyzing, and detecting drift in multiple real-world log datasets.

## 2. Problem Statement

* Massive log volumes make it challenging to detect subtle system behavior changes (“reliability drift”).
* Drift can manifest as:
  + Gradual performance degradation
  + Unexpected changes in component interactions
  + Variations in error rates/types
  + Shifts in resource utilization
  + Changes in response times or communication patterns
* If undetected, these can lead to instability, failures, and outages.
* Our analysis spans multiple log types for comprehensive health monitoring and early warning.

## 3. Datasets Overview

* **Sources:** HDFS, Apache, BGL, HealthApp, HPC, Linux, Mac logs.
* **Formats:** Highly structured (HDFS, BGL, Apache), semi-structured (HealthApp), syslog-style (Mac, Linux).
* **Key Point:** Each dataset reflects unique system behaviors and operational patterns.
* **Sample log entry formats and notable characteristics:**
  + **HDFS:** 081109 203615 148 INFO dfs.DataNode$PacketResponder: PacketResponder 1 for block blk\_38865049064139660 terminating
    - Highly structured, block-related events, mostly INFO.
  + **Apache:** [Thu Jun 09 06:07:04 2005] [notice] LDAP: Built with OpenLDAP LDAP SDK
    - Web server format, many ERRORs, clear cycles.
  + **HealthApp:** 20171223-22:15:29:606|Step\_LSC|30002312|onStandStepChanged 3579
    - Semi-structured, user/device context, mostly OTHER.
  + **BGL:** - 1117838570 2005.06.03 R02-M1-N0-C:J12-U11 2005-06-03-15.42.50.363779 ...
    - HPC cluster logs, node/job IDs, structured.
  + **HPC, Linux, Mac:**
    - Syslog style, system daemons, user actions, error/warning spikes.

## 4. Data Quality & Preparation

* **Output:**
  + HDFS: 0 empty entries found.
  + APACHE: 0 empty entries found.
  + BGL: 0 empty entries found.
  + HEALTHAPP: 0 empty entries found.
  + HPC: 0 empty entries found.
  + LINUX: 0 empty entries found.
  + MAC: 0 empty entries found.
* **Finding:** All datasets are high quality, with no missing or malformed entries.
* Special handling for different timestamp formats; some logs had default/malformed timestamps (e.g., 1900).
* Used robust encoding detection and whitespace handling.
* Thought process: Ensured clean, consistent data for downstream analysis.

## 5. Exploratory Data Analysis (EDA)

* **Message Type Distributions (example):**
  + HDFS: INFO 96%, ERROR 4%
  + Apache: ERROR 67%, INFO 24%, OTHER 8%
  + BGL: INFO 61%, ERROR 29%, CRITICAL 8%
  + HealthApp: OTHER 99%, ERROR 0.7%
  + Linux/Mac: Mostly OTHER, some ERROR/INFO
* **Entry Length & Word Count (example):**
  + HDFS: Mean 12.4 words, Std 3.7
  + Apache: Mean 12.4 words, Std 2.3
  + BGL: Mean 15.2 words, Std 7.5
* **Temporal Patterns:**
  + HDFS: Activity peaks at 21:00, 10:00, 7:00
  + Apache: Bursts of errors during certain days/hours
  + HealthApp: Usage peaks in morning/evening
* **Component Analysis:**
  + HDFS: Top = DataNode, PacketResponder, FSNamesystem
  + Apache: Top = error, notice, client IPs
  + BGL: Top = ciod, s, idoproxydb
* **Interesting Patterns & Outliers:**
  + Apache: Error spikes = possible attacks
  + HealthApp: Error spikes after updates
  + BGL/HPC: Nodes with high error rates
* (Add plots and sample log entries as needed)

## 6. Feature Engineering

* **Output:**
  + Features extracted: message\_type, component, word\_count, entry\_length, special\_char\_count, num\_value\_count, timestamp features.
  + Uniqueness ratio (example):
    - HDFS: 100% unique entries
    - Apache: 50% unique
    - BGL: 100% unique
    - HealthApp: 99.8% unique
    - Linux: 99.9% unique
    - Mac: 82% unique
  + Rare components (≤2 times):
    - HDFS: 2,201
    - Apache: 19,953
    - BGL: 178,301
    - HealthApp: 9
    - HPC: 1,801
    - Linux: 7,345
    - Mac: 6,142
* These features powered clustering, drift detection, and anomaly identification.
* Thought process: Chose features to capture both structure and semantics of logs.

## 7. Clustering & Visualization

* **Output:**
  + KMeans (k=5) cluster sizes (example, HDFS):
    - Cluster 0: 263, Cluster 1: 770, Cluster 2: 314, Cluster 3: 311, Cluster 4: 342
  + t-SNE plots: Showed well-separated clusters for all datasets.
  + Outliers: Small, isolated clusters or points.
* **Finding:**
  + Most logs group into a few main patterns; outliers often correspond to rare or anomalous events.
  + Subsampling (2,000–5,000 entries) was necessary for large datasets.
* **Interpretation:**
  + Clustering is effective for summarizing log diversity and identifying key operational patterns.
  + Outliers are valuable for drift and anomaly detection.
* (Add t-SNE plots and cluster size charts as needed)

## 8. Drift Detection

* **Output:**
  + KS and Chi-squared tests detected multiple drift windows (see time series plots).
  + Change point detection (ruptures) found significant shifts in error rate (e.g., at windows 50, 8125).
* **Finding:**
  + Drift windows often aligned with operational events, updates, or incidents.
  + Example: After drift, Apache logs showed repeated web server errors.
* **Interpretation:**
  + Drift detection provides early warning of system changes or incidents.
  + Visualizations make it easy to communicate findings.
* (Add time series plots and drift window tables as needed)

## 9. Example: Before & After Drift

* **Before drift:**
  + Jan 5 04:02:05 combo su(pam\_unix)[20782]: session opened for user root by (uid=0)
  + Jan 5 04:02:06 combo logrotate: ALERT exited abnormally with exit status 1
* **After drift:**
  + [Fri Jan 06 09:19:49 2006] [error] [client …] File does not exist: /var/www/html/phpmyadmin
  + [Fri Jan 06 11:07:11 2006] [error] [client …] File does not exist: /var/www/html/phpmyadmin
* **Interpretation:**
  + Clear shift from routine operations to repeated error messages, indicating a system change or incident.
* (Add sample log entries from before/after drift windows)

## 10. Challenges & Lessons Learned

* **Timestamp inconsistencies** (e.g., defaulting to 1900) complicated temporal analysis.
* **Large dataset size** required subsampling, which may miss rare events.
* **Diverse log formats** needed flexible, dataset-specific preprocessing.
* **Not all drift signals** corresponded to real incidents—some were due to data quality or routine changes.
* **Clustering limitations:** DBSCAN sometimes found only one cluster due to high similarity or parameter choice.

## 11. Conclusions & Recommendations

* Unsupervised log analysis revealed system behavior, dominant patterns, and drift.
* Outliers and rare events provided valuable early warning signals.
* **Recommendations:**
  + Improve timestamp normalization.
  + Automate drift monitoring and alerting.
  + Investigate drift windows for root causes.
  + Expand feature set for deeper insights.
  + Scale analysis for very large datasets.

## 12. References

* Chandola, V., Banerjee, A., & Kumar, V. (2009). “Anomaly Detection: A Survey.” *ACM Computing Surveys*, 41(3), 1–58.
* He, P., et al. (2017). “An Evaluation Study on Log Parsing and Its Use in Log Mining.” *DSN*.
* Ruptures: Change Point Detection in Python. <https://centre-borelli.github.io/ruptures-docs/>
* Scikit-learn, Pandas, Seaborn documentation.
* Loghub repository: <https://github.com/logpai/loghub>

(Add figures, plots, and sample log entries in the relevant sections as you finalize the document.)