Markov Model for Improved Outage Candidate Generation in Supervised Outage Prediction Models

Aditya Mate
Microsoft

Youjiang Wu
Microsoft

Joe Hu
Microsoft

Udaivir Yadav
Microsoft

Yingnong Dang
Microsoft

1 INTRODUCTION

Large-scale cloud platforms support millions of users and mission-critical services, making timely and accurate detection of service outages a central requirement for reliability management and incident response [2]. Early identification of outages enables faster mitigation, reduces customer impact, and underpins many automated and human-in-the-loop operational workflows in modern AIOps systems [1].

To address this challenge, cloud platforms often employ supervised learning–based Outage Prediction Models (OPMs) that classify events as outage or non-outage using service-level indicators and other features. These models typically produce a continuous probability signal indicating the likelihood of an ongoing or imminent service outage. In practice, outage declaration is often driven by heuristic rules or fixed thresholds applied to this signal, such as triggering when the probability or an aggregate statistic over a sliding window exceeds a predefined value. While simple to operationalize, such approaches exhibit systematic failure modes: transient probability spikes can lead to false positives, while sustained periods of moderately elevated probability may never cross thresholds, resulting in missed outages (false negatives).

To address these limitations, we recast outage declaration as a sequential decision problem over the trajectory of supervised model outputs. Our approach introduces a lightweight Markovian abstraction that captures probability magnitude together with temporal persistence, enabling data-driven outage decisions that go beyond pointwise thresholding.

2 PROBLEM FORMULATION

Let ${p_t}_{t=1}^{T}$, where $p_t \in [0, 1]$, denote the time series of outage probabilities produced by a supervised OPM. From historical data, we observe corresponding binary outage labels ${y_t}_{t=1}^{T}$, where $y_t \in \{0, 1\}$ indicates the presence or absence of an outage at time t. The goal is to determine a declaration policy π that maps this probability sequence to a binary outage decision $d_t \in 0, 1$ at each time step.

3 METHOD

We frame outage declaration as a state-based decision problem over the time series of outage probabilities produced by the supervised OPM. We define a Markov model by constructing a discrete state representation that encodes the current probability value together with temporal context, including recent probability history, short-term trends (rising, falling, or stable), and magnitude of change over multiple lookback windows (e.g., 30 and 60 minutes). These elements are combined into a state tuple, yielding a finite state space that captures both probability magnitude and persistence.

The probability sequence is mapped to a corresponding state trajectory $\{s_t\}$, where each state is paired with the available binary outage observation $y_t \in \{0, 1\}$.Using historical data, we estimate the emission probability of outage for each state, identifying states that are predictive of outages. At inference time, outage declaration is performed via a learned decision rule over states: when the system transitions into outage-indicative states, an outage is declared if the emission probability is above a learned threshold value.

4 INITIAL RESULTS

We compare the proposed Markov-based decision model against a baseline that directly uses the raw OPM output on two production services: *Azure OpenAI Service* and *Azure Databricks*. For Azure OpenAI, precision improves from 66.7% to 72.7%, while recall improves by 2× (from 7.5% to 15.1%), resulting in an increase in F1 score from 0.136 to 0.250; this corresponds to higher true positives (8 vs. 4), fewer false positives (3 vs. 2), and reduced false negatives (45 vs. 49). For Azure Databricks, precision increases from 50.0% to 66.7%, with a slight drop in recall (from 6.7% to 4.4%). driving a reduction in false positives (1 vs. 3). Overall, these results suggest that explicitly modeling temporal persistence shows promise in improving decision performance.

REFERENCES

[1] Shilin He, Jieming Zhu, Pinjia He, and Michael R. Lyu. Experience report: System log analysis for anomaly detection. *IEEE International Symposium on Software Reliability Engineering (ISSRE)*, 2016. Demonstrates the operational importance of ML-based anomaly and failure detection in production systems.

[2] Dongmei Zhang et al. An aiops framework for incident management. *Microsoft Technical Report*, 2019. Describes the role of machine learning in modern cloud incident management pipelines.