GUIDEWIRE DEVTrails University Hackathon

KUBERNETES FAILURE PREDICTION & ANOMALY DETECTION

Team Name: GDSquared

OVERVIEW & MOTIVATION

- Background:
  - Kubernetes clusters power critical applications but are prone to failures
  - Failures can include pod/node issues, resource exhaustion, network problems, and service disruptions
- Motivation:
  - Reduce downtime, optimize resource usage, and enhance reliability
  - Leverage machine learning to predict failures and detect anomalies

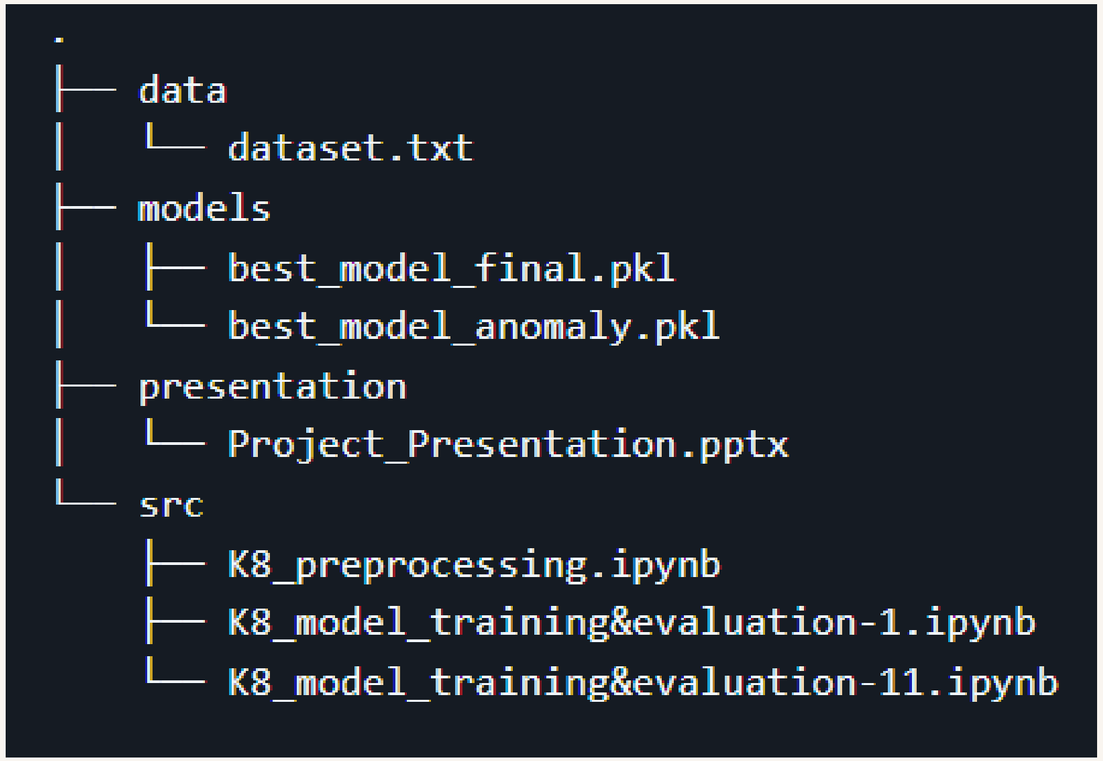
OBJECTIVES

- Build a model to predict failures (failed events)
- Detect anomalies in resource usage and logs
- Create a reproducible data pipeline and modeling framework

DATA & FILE STRUCTURE

- Dataset:
  - Derived from Google Cluster Dataset 2019.
  - Includes metrics such as CPU usage, memory usage, event logs, etc.

- Repository Structure:
  - data: Contains dataset links
  - models: Contains saved failure and anomaly prediction models
  - presentation: Contains this presentation file
  - src: Contains three notebooks:
    - Preprocessing & feature engineering
    - Training various ML models
    - Advanced anomaly detection and prediction



DATA PREPROCESSING

- Time Conversion:
  - Converting timestamps to datetime objects
  - Extracting hour, day, weekend indicators

- Feature Engineering:
  - Deriving metrics (memory_utilization, memory_pressure, cache_ratio)
  - Extracting CPU distribution statistics (mean, std, max, p95)

- Anomaly Detection:
  - Using Isolation Forest to add an anomaly flag

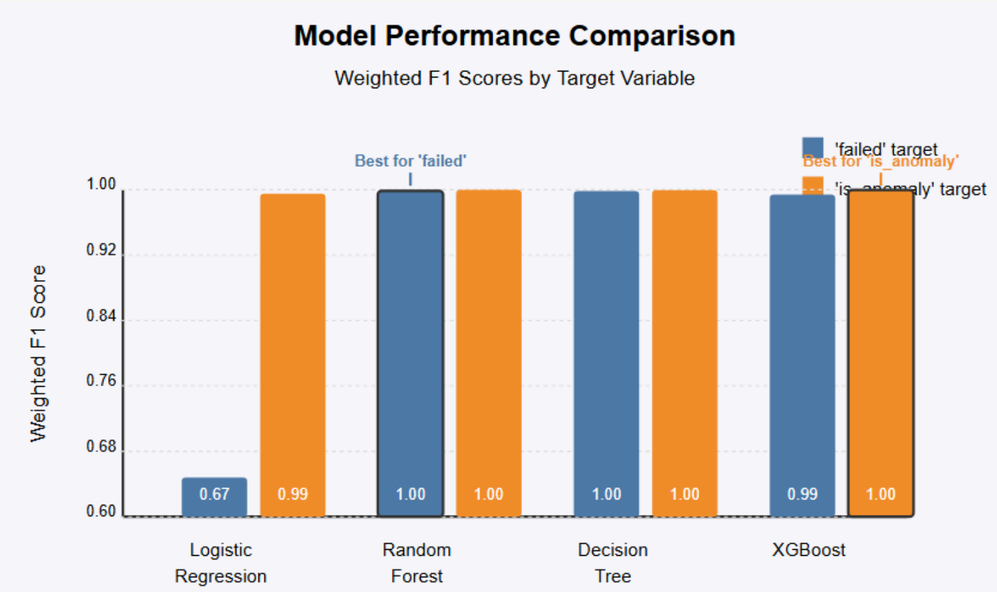
MODEL TRAINING OVERVIEW

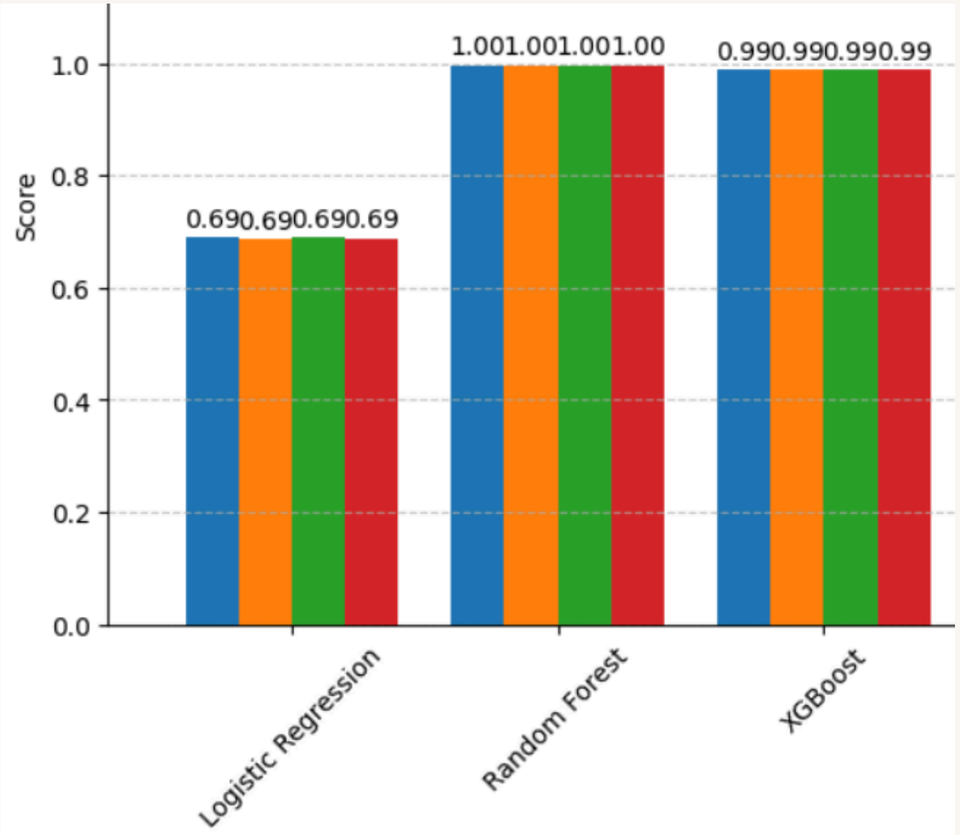
- Static Models:
  - Logistic Regression, Random Forest, Decision Tree, XGBoost
  - Looping over targets (failed and is_anomaly)

- Evaluation:
  - Confusion matrix and classification reports
  - Weighted F1 scores to select best models

- Saving Best Models:
  - Models are saved as pickle files for later use

EXPERIMENTATION & RESULTS





DISCUSSION

- Key Observations:
  - "For predicting 'failed', Random Forest achieved the best performance with a F1-score of 0.9979. For predicting 'is_anomaly', XGBoost model yielded the highest F1-score of 0.9995."

- Challenges:
  - Data imbalance and feature scaling issues
  - Threshold tuning for different failure types

- Learnings:
  - Importance of robust preprocessing and feature engineering
  - Benefits of using multiple models and automated model selection

CONCLUSION

- Summary:
  - Developed a complete ML pipeline for predicting failures and detecting anomalies
  - Successfully trained multiple models and selected the best-performing ones

- Impact:
  - Potential to reduce downtime and improve resource allocation in Kubernetes environments