Séries temporais

IA, mas não LLM

Uma introdução de machine learning na observabilidade de séries temporais

“Observabilidade é a capacidade de **inferir** os **estados internos** de um **sistema complexo** a partir de suas **saídas externas**.”

“Observabilidade é o processo em que um humano possa fazer perguntas **SIGNIFICATIVAS**, receba respostas **ÚTEIS** e possa tomar **AÇÕES** com informações obtidas.” Weakly, Hazel + Cantrill, Bryan

Igor Estevan Jasinski

Engenherio de Observabilidade Sicredi

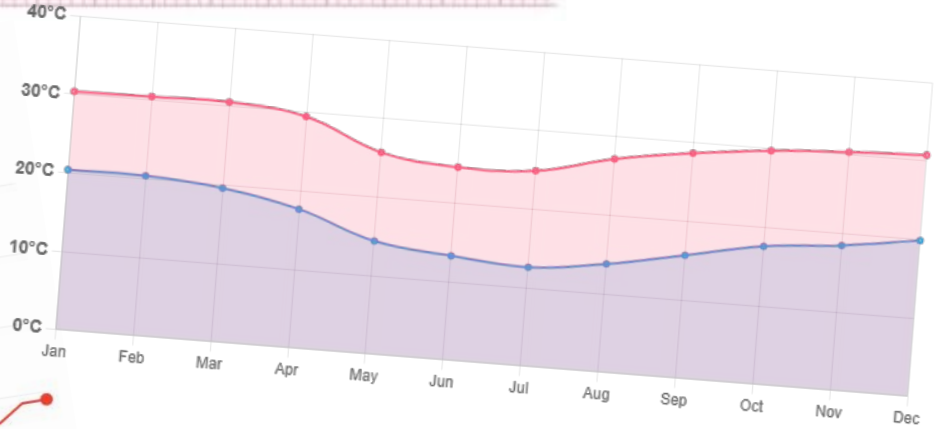
SME - CNCF de observabilidade e OpenTelemetry CLOUD NATIVE COMPUTING FOUNDATION

Pai de 5 gatos

Séries temporais

Séries temporais referem-se a conjuntos de **dados** que são coletados, registrados ou observados ao longo do tempo e gravados em intervalos **regulares**.


5,4439
↓2,03%
-0,1130 1 mês
12 de ago., 11:02:24 UTC · Exoneração de responsabilidade
1 dia
5 dias
1 mês
6 meses
YTD
1 ano
5 anos
MÁX.
5,60
5,55
5,50
5,45
5,40
13 de jul.
20 de jul.
27 de jul.
3 de ago.
10 de ago.


40°C
30°C
20°C
10°C
0°C
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec

Séries temporais na observabilidade

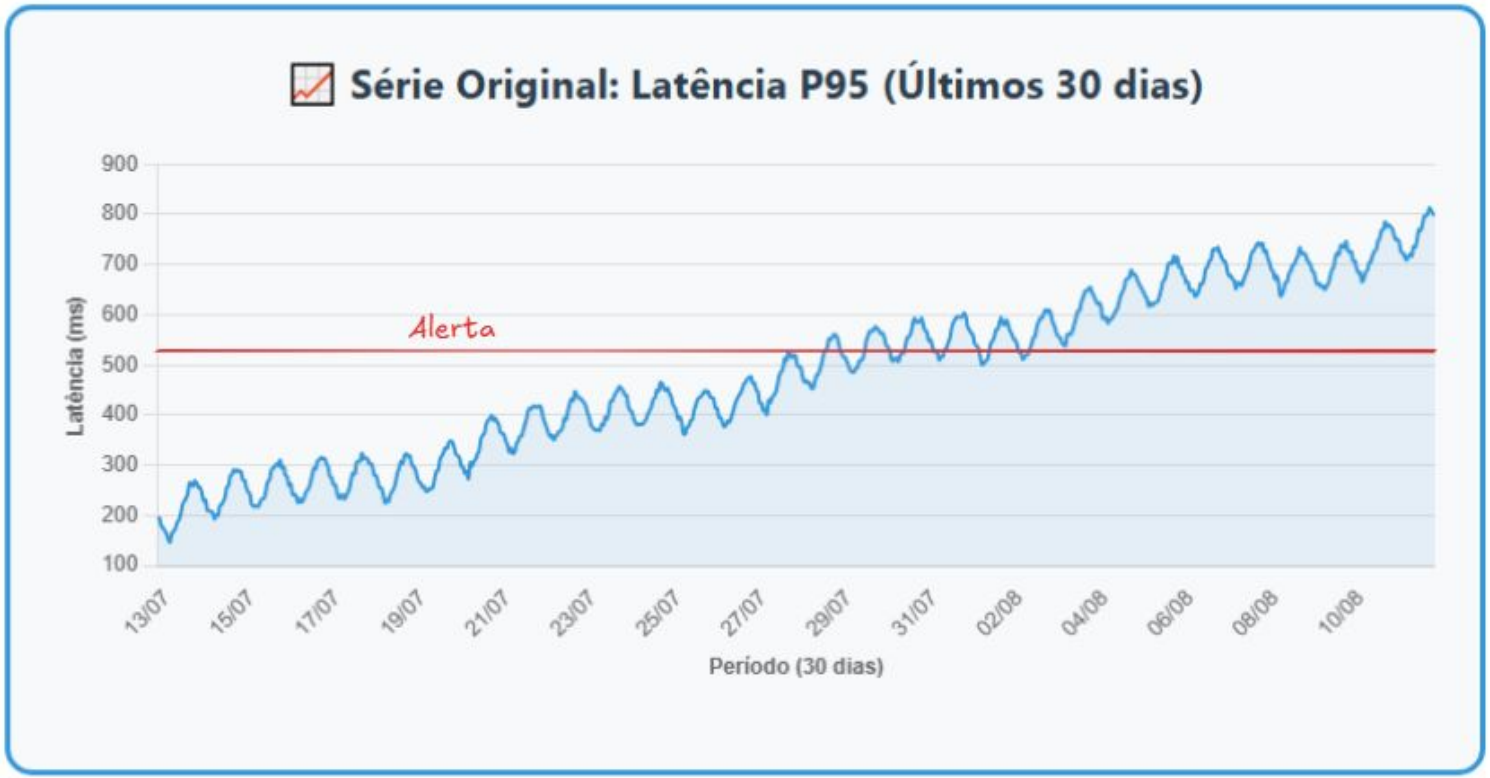








Séries temporais na Observabilidade



Introdução a Machine Learning para observabilidade

Aprendizado de Máquina (Machine Learning, ML) é um ramo da Inteligência Artificial (IA) que permite que sistemas computacionais aprendam com **dados**, identifiquem **padrões**, **anomalias** e a criar **predições**.

Introdução machine learning

- Input dos dados (Dataset)
- Seleção de algoritmo
- Treino do algoritmo
- Desenvolvimento do modelo
- Inferência do modelo

Machine learning na observabilidade

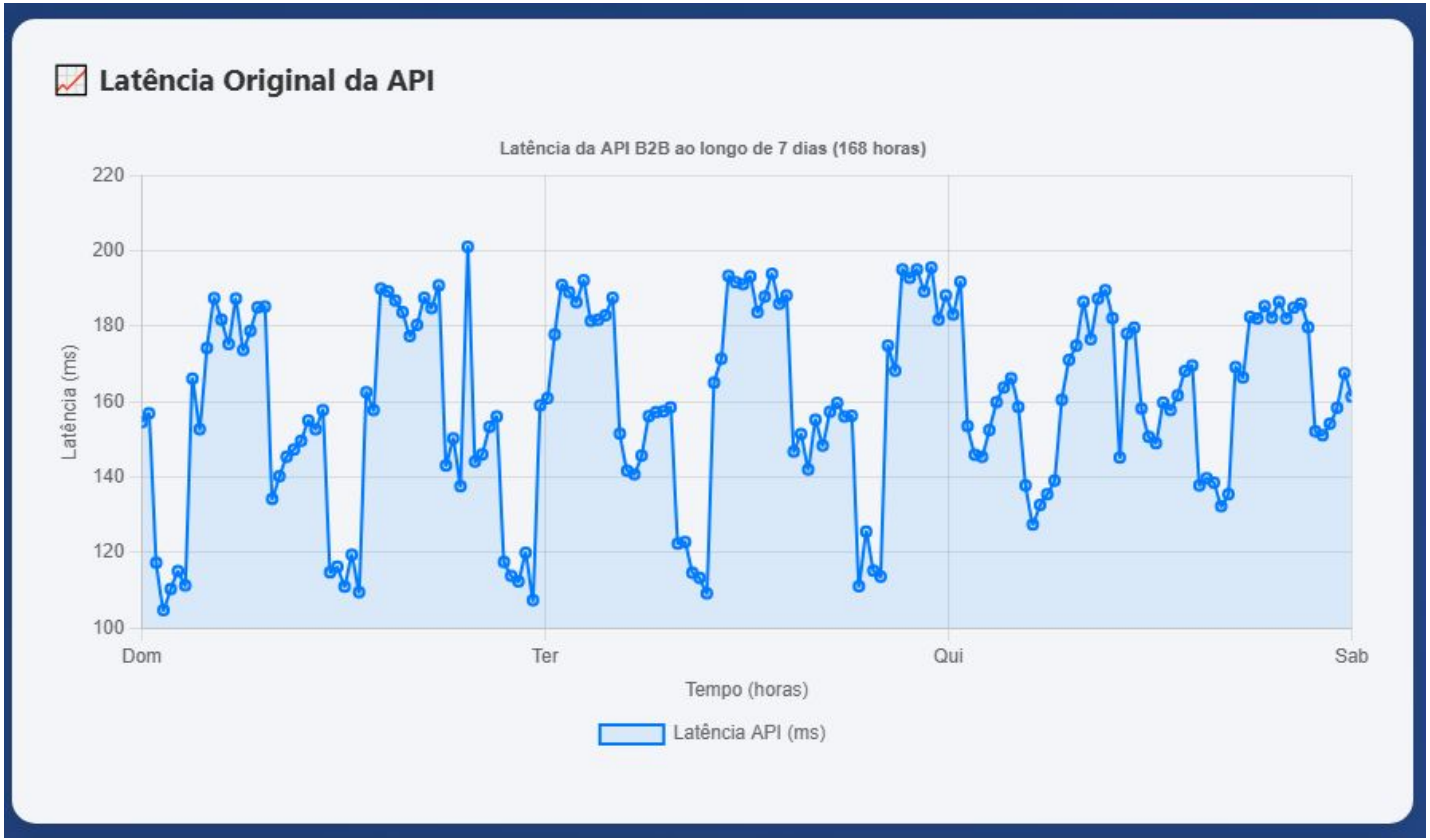
Entrada → **Treino** → **Inferência** → **Saída**

Timestamp(ds)	Valor(y)
2024-10-30 22:30:00	273
2024-10-30 22:35:00	301
2024-10-30 22:40:00	320
2024-10-30 22:45:00	337
2024-10-30 22:50:00	402
2024-10-30 22:55:00	340

- Decomposição
- Forecast
- Detecção de anomalia

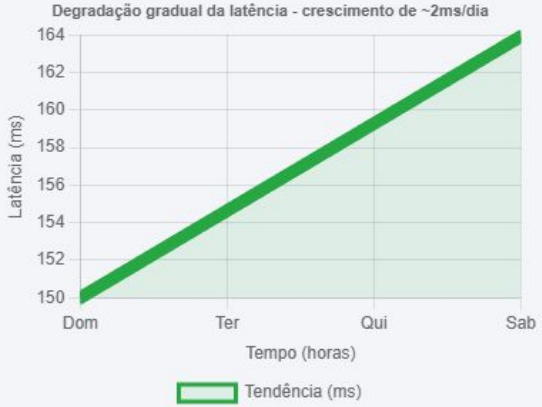
Decomposição

Decomposição



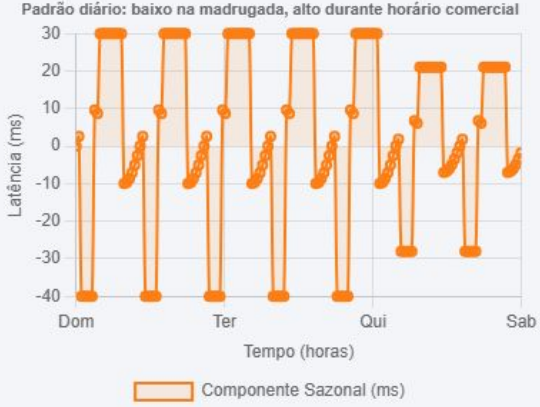
Decomposição

Tendência



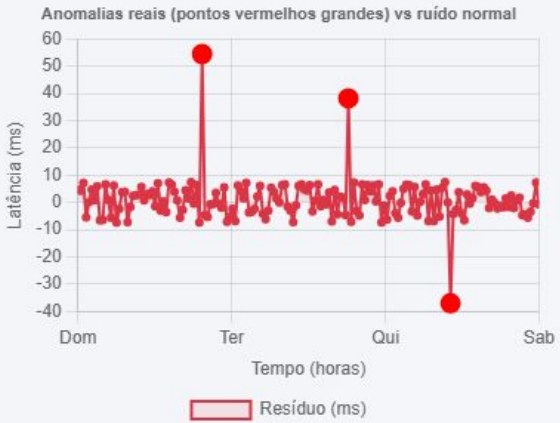
Tendência: Movimento de longo prazo da série temporal. Representa a direção geral (crescimento, declínio ou estabilidade) da métrica ao longo do tempo, ignorando flutuações de curto prazo.

Sazonalidade



Sazonalidade: Padrões regulares e previsíveis que se repetem em intervalos fixos (hora, dia, semana, mês). Reflete comportamento natural e esperado dos usuários e sistemas.

Resíduo



Resíduo: Variações que não são explicadas nem por tendência nem por sazonalidade. Representa o componente aleatório onde encontramos anomalias verdadeiras.

Algoritmos de decomposição

STL (Seasonal-Trend decomposition using LoESS)

Paradigma: Non-parametric, LOESS-based decomposition
Fundamento: Aplicação iterativa de suavização LOESS para extração robusta de componentes temporais com handling automático de outliers.
Aplicação Ótima: Séries com sazonalidade única e estável, alta tolerância a outliers, infraestrutura geral
Complexidade Computacional: O(n log n)

Robusto a Outliers

$$Y(t) = T(t) + S(t) + R(t)$$

Iterative LOESS smoothing com robustness weights

X13 ARIMA-SEATS

Paradigma: Model-based, statistically optimal decomposition
Fundamento: Combinação de métodos X-11 com modelagem ARIMA para ajuste sazonal estatisticamente ótimo.
Aplicação Ótima: Séries econômicas/financeiras, análises oficiais, máxima precisão estatística
Complexidade Computacional: $O(n^2)$ devido à estimação ARIMA

Precisão Estatística

$$ARIMA(p,d,q)(P,D,Q)_s \text{ modeling} + \text{X-11 filters}$$

Model-based seasonal adjustment

TBATS (Trigonometric, Box-Cox, ARMA, Trend, Seasonal)

Paradigma: State-space model com componentes trigonométricas
Fundamento: Modelagem de sazonalidade complexa através de funções trigonométricas com state-space representation para handling de periodicidades não-inteiras.
Aplicação Ótima: Sazonalidade não-inteira, múltiplas periodicidades, séries de alta frequência
Complexidade Computacional: $O(n^3)$ para estimação de parâmetros

Flexibilidade Máxima

$$Y(t) = \ell_t + b_t + \sum s_{j,t} + \varepsilon_t$$

State-space com componentes trigonométricas

Algoritmos de decomposição

```
from statsmodels.tsa.seasonal import STL

decomp = STL(data, period=m).fit()
```

Entrada do treino: Dados históricos

Saída: Dados de tendência, sazonalidade e resíduo.

```
from statsmodels.tsa.x13 import x13_arima_analysis

result = x13_arima_analysis(data)
components = {
    'trend': result.trend,
    'seasonal': result.seasonal,
    'cycle': result.cycle,
    'irregular': result.irregular
}
```

Decomposição na observabilidade

Análise de Tendência (Trend Component)

Detecção precoce de drift sistêmico e degradação gradual

Permite detecção precoce de degradação gradual de performance, crescimento insustentável de recursos, ou mudanças estruturais no comportamento do sistema. Isolamento da componente de tendência revela patterns de longo prazo invisíveis em dados brutos, possibilitando intervenção proativa 6-12h antes de impacto crítico.

Modelagem de Sazonalidade (Seasonal Component)

Thresholds adaptativos baseados em padrões cíclicos

Fundamental para implementação de thresholds adaptativos que se ajustam dinamicamente aos padrões esperados, reduzindo drasticamente falsos positivos. Extração de padrões sazonais permite baselines que variam conforme contexto temporal, eliminando 85% dos alertas desnecessários durante variações normais.

Análise de Resíduos (Residual Component)

Isolamento de variações não explicadas para detecção pura

O resíduo isola variações não explicadas pelos padrões conhecidos, permitindo detecção de anomalias através de métodos estatísticos puros. Componente residual oferece signal-to-noise ratio 10x superior, pois variações previsíveis foram removidas, concentrando análise apenas em eventos genuinamente anômalos.

Decomposição na observabilidade

```
Tendência

deriv(api_latency_trend[2h]) > 5

Sazonalidade

abs(api_latency - api_latency_seasonal) > api_latency_seasonal * 0.3

Resíduo

abs(api_latency_residual) > 2 * stddev_over_time(api_latency_residual[24h])
```

Forecast

Forecast


Dados Históricos
Previsão (ŷ)
Limite Superior (95%)
Limite Inferior (95%)
Response Time (ms)
500
450
400
350
300
250
200
150
100
50
19:40
22:40
01:40
04:40
07:40
10:40
13:40
16:40
19:40
22:40
01:40
04:40
07:40
10:40
13:40
16:40
19:40
22:40
01:40
04:40
07:40
10:40
13:40
16:40
19:40
Tempo

Algoritmos de Forecast

📈 ARIMA (AutoRegressive Integrated Moving Average)

Melhor para: Séries estacionárias com padrões lineares
Vantagens: Rápido, interpretável, boa para tendências
Desvantagens: Dificuldade com sazonalidade complexa

$$ARIMA(p,d,q): (1-\varphi_1 L-\ldots-\varphi_p L^p)(1-L)^d X_t = (1+\theta_1 L+\ldots+\theta_e L^q)\varepsilon_e$$

🔮 Prophet (Facebook)

Melhor para: Dados com forte sazonalidade e holidays
Vantagens: Robustos a outliers, missing data
Desvantagens: Maior complexidade computacional

$$y(t) = g(t) + s(t) + h(t) + \varepsilon_t$$
trend + seasonality + holidays + noise

LSTM (Long Short-Term Memory)

Melhor para: Padrões complexos e não-lineares
Vantagens: Captura dependências de longo prazo
Desvantagens: Requer mais dados e recursos

$$h_t = LSTM(x_t, h_{t-1})$$
Hidden state com memory gates

Algoritmos de Forecast

```
my_model = Prophet(interval_width=0.95)

my_model.fit(df)

future_dates = my_model.make_future_dataframe(periods=horizon, freq='MS')
future_dates.head()
```

```
model = Sequential()
model.add(LSTM(units=64, input_shape=(sequence_length, 1)))
model.add(Dense(units=1))
model.compile(optimizer='adam', loss='mean_squared_error')

model.fit(X_train, y_train, epochs=10, batch_size=32)

predictions = model.predict(X_test)
```

Entrada do treino: Dados históricos

Saída da inferência do modelo: horizonte do forecast (**yhat**, **yhat_upper**, **yhat_lower**)

Forecast na observabilidade

🔮 Observabilidade Proativa

Antecipe problemas antes que afetem usuários. Identifique tendências de degradação com até 24-48h de antecedência.

📈 Planejamento de Capacidade

Preveja necessidades de recursos baseado em padrões históricos e sazonalidade, otimizando custos de infraestrutura.

🎯 Alertas Inteligentes

Reduza falsos positivos usando intervalos de confiança dinâmicos ao invés de thresholds estáticos.

🔧 Manutenção Preditiva

Agende manutenções baseado em previsões de degradação de performance.

Forecast na observabilidade

```
Alert com Intervalos de Confiança

cpu_usage_percent > cpu_forecast_upper_95 or cpu_usage_percent < cpu_forecast_lower_95

Requests acima do predito

requests_current > requests_forecast_yhat * 3

Vendas baixas vs predição
sales_current < sales_forecast_yhat * 0.8

Anomaly Score
kafka_lag - kafka_lag_yhat / kafka_lag_yhat_upper - kafka_lag_yhat_lower
```

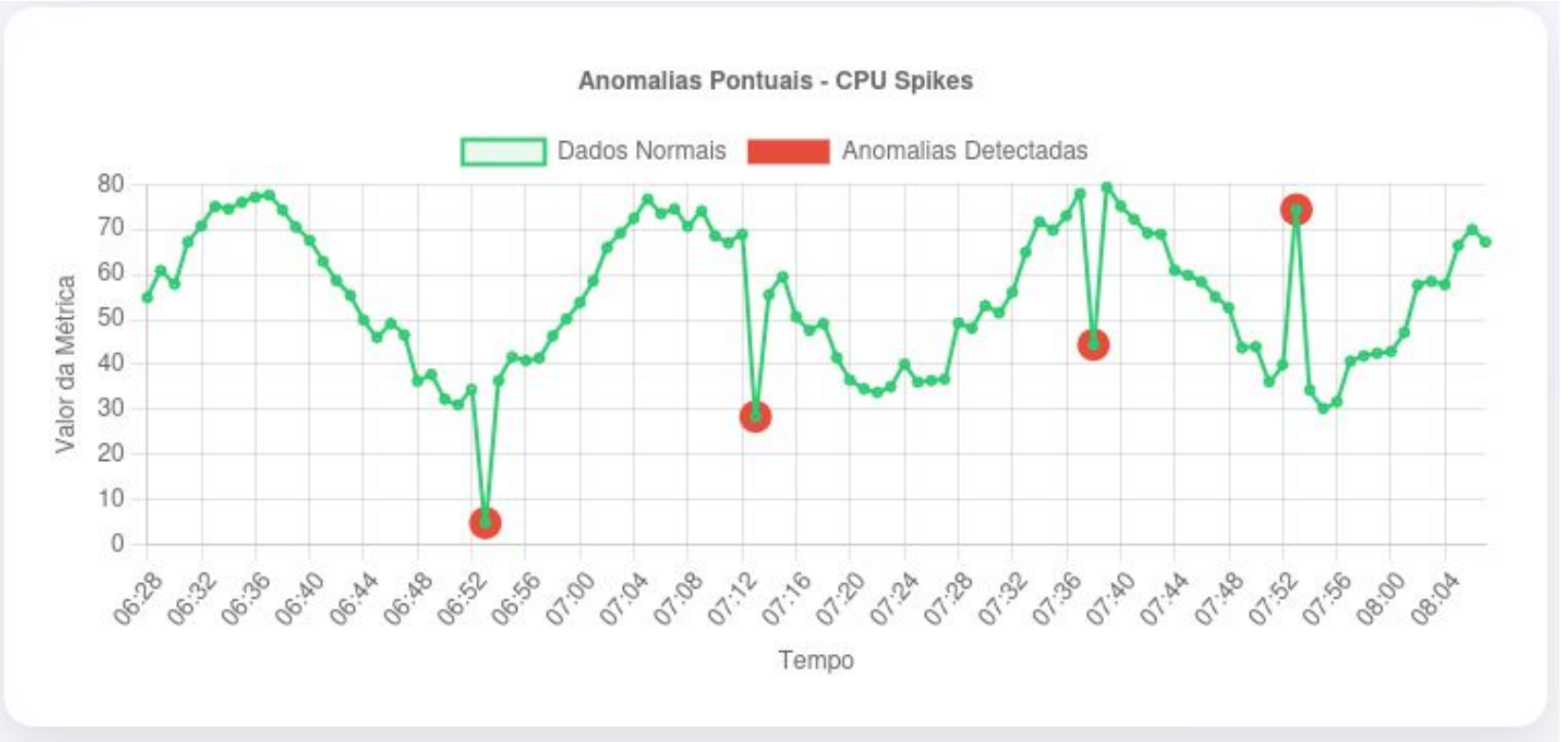
Detecção de anomalia

Detecção de anomalias - Pontuais

Anomalias Pontuais (Point Anomalies)

Definição: Observações individuais que apresentam desvio estatísticamente significativo em relação à distribuição esperada.
Características: Eventos isolados, tipicamente representando falhas instantâneas ou spikes de carga.
Exemplos: Spike abrupto de utilização de CPU, erro HTTP 500 isolado, latência de request anômala.
Métodos de Detecção: Z-score, Interquartile Range (IQR), Isolation Forest, One-Class SVM.

Detecção de anomalias - Pontuais

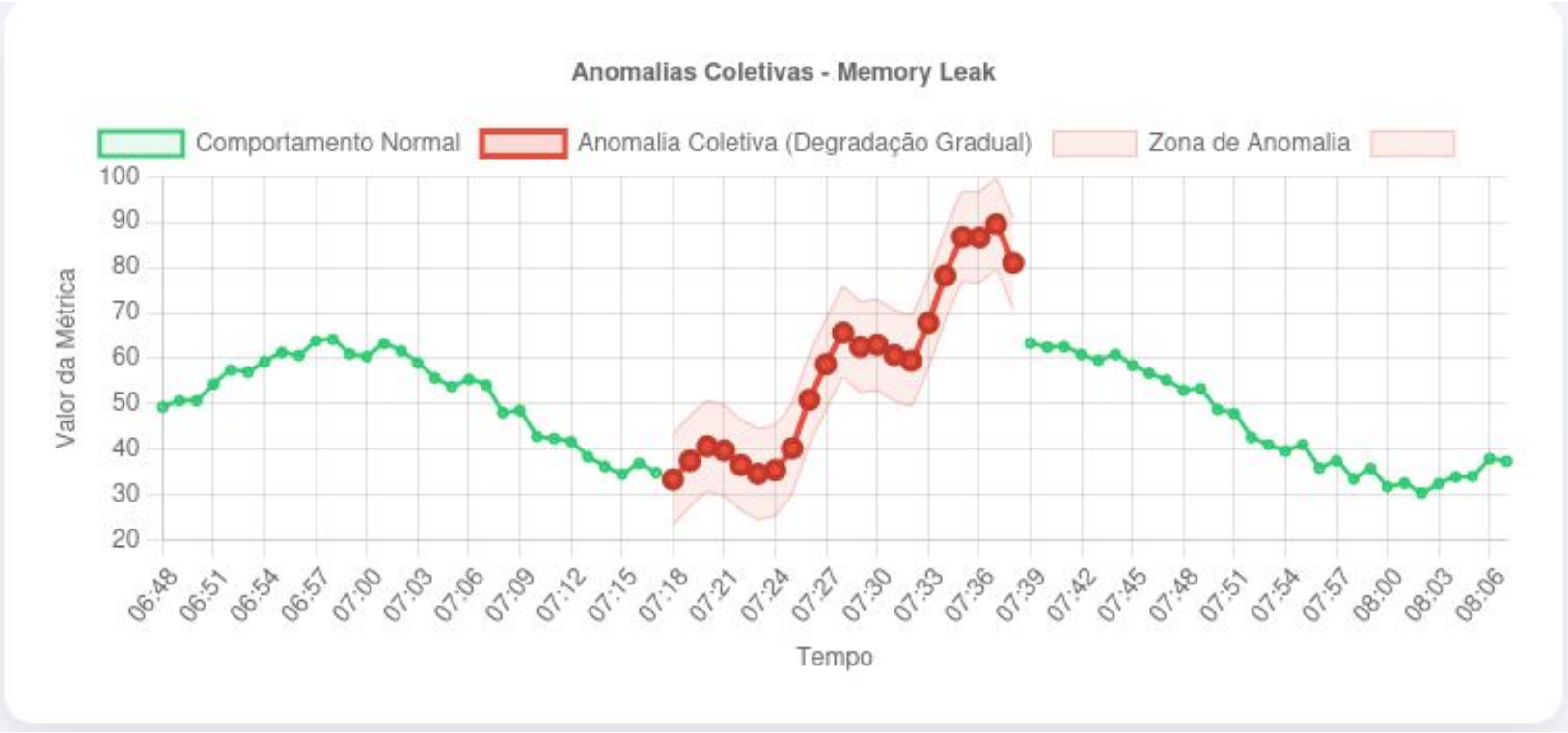


Detecção de anomalias - Coletivas

Anomalias Coletivas (Collective Anomalies)

Definição: Sequências temporais onde observações individuais podem ser normais, mas o padrão coletivo apresenta comportamento anômalo.
Características: Cada ponto individual encontra-se dentro de parâmetros normais, mas a sequência temporal revela degradação ou instabilidade sistêmica.
Exemplos: Degradação gradual de performance, memory leak progressivo, oscilações de frequência anômala.
Métodos de Detecção: LSTM Autoencoders, Sequential Pattern Mining, Change Point Detection, Hidden Markov Models.

Detecção de anomalias - Coletivas

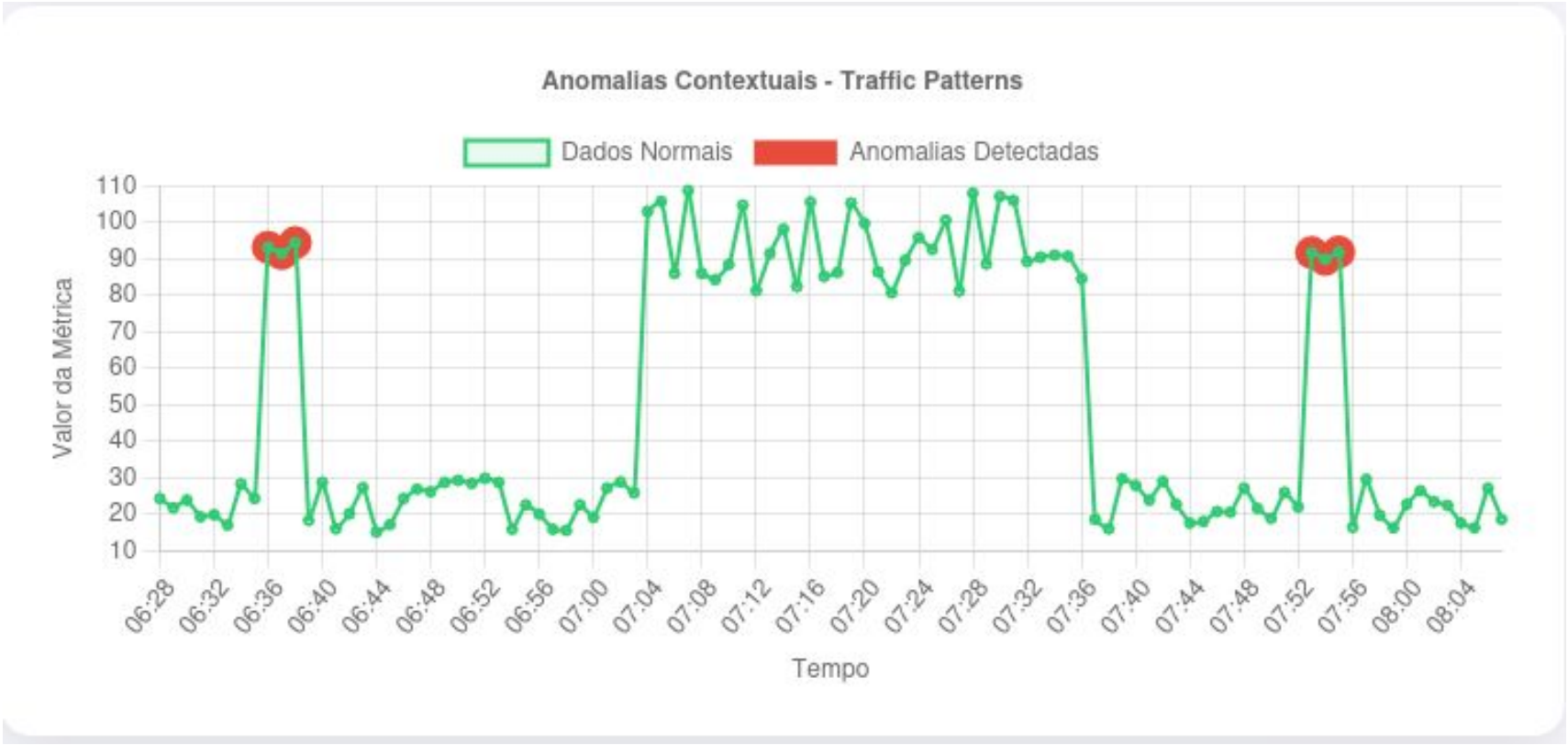


Detecção de anomalias - Contextuais

Anomalias Contextuais (Contextual Anomalies)

Definição: Observações que são anômalas apenas quando consideradas em contexto específico de features temporais, geográficas ou categóricas.
Características: Valores numericamente normais que se tornam anômalos devido ao contexto temporal, espacial ou comportamental.
Exemplos: Tráfego elevado durante período de baixa atividade, CPU alta em horário de manutenção programada.
Métodos de Detecção: Conditional Anomaly Detection, Time-Series Decomposition, Feature-based Classification.

Detecção de anomalias - Contextuais



Algoritmos de Detecção de anomalias

Z Statistical Methods

Paradigma: Métodos Estatísticos Clássicos
Fundamento: Detecção de desvios estatísticamente significativos assumindo distribuição normal.
Aplicação Ótima: Séries univariadas, implementação de baixa latência, baseline systems
Complexidade Computacional: O(n)

Latência Mínima

$$Z\text{-score} = (x - \mu) / \sigma$$

Anomalia quando $|Z| >$ threshold (tipicamente 2.5 ou 3)

LF Local Outlier Factor

Paradigma: Density-based Outlier Detection
Fundamento: Comparação de densidade local de pontos com densidade de vizinhança k-nearest.
Aplicação Ótima: Detecção de outliers em regiões de diferentes densidades
Complexidade Computacional: $O(n^2)$

Precisão Contextual

$$LOF(x) = \sum(lrd(o)/lrd(x)) / |N(x)|$$

Local Reachability Density comparativa

IF Isolation Forest

Paradigma: Unsupervised, Tree-based Ensemble
Fundamento: Anomalias requerem menor número de partições para isolamento em estruturas hierárquicas.
Aplicação Ótima: Datasets multidimensionais, detecção em tempo real, alta escalabilidade
Complexidade Computacional: O(n log n)

Alta Performance

$$s(x,n) = 2^{(-E(h(x))/c(n))}$$

onde $E(h(x))$ representa a profundidade média de isolamento

Algoritmos de Detecção de anomalias

```
IsolationForest
model = IsolationForest(contamination=0.05)

model.fit(data)
#retorna -1 para anomalias e 1 para normais
data['Anomaly'] = model.predict(data)
```

Entrada do treino: Dados históricos
Entrada da inferência: Valor atual
Saída da inferência: anomaly_score [1 ou -1]

Detecção de anomalias na observabilidade

🎯 Contexto Inteligente

Considera sazonalidade e correlações
100 req/s às 3AM é anômalo, mas normal às 14h. ML entende contexto.

📊 Redução Drástica de Ruído

Muito menos alertas desnecessários
Algoritmos aprendem padrões normais, alertando apenas em verdadeiras anomalias.

📈 Redução Significativa de MTTR

Menos tempo perdido com falsos positivos
Equipes focam em problemas reais ao invés de investigar alarmes falsos.

🤖 Escalabilidade Automática

Monitora milhares de métricas
Impossível configurar thresholds manualmente para todas as métricas.

Detecção de anomalias na observabilidade

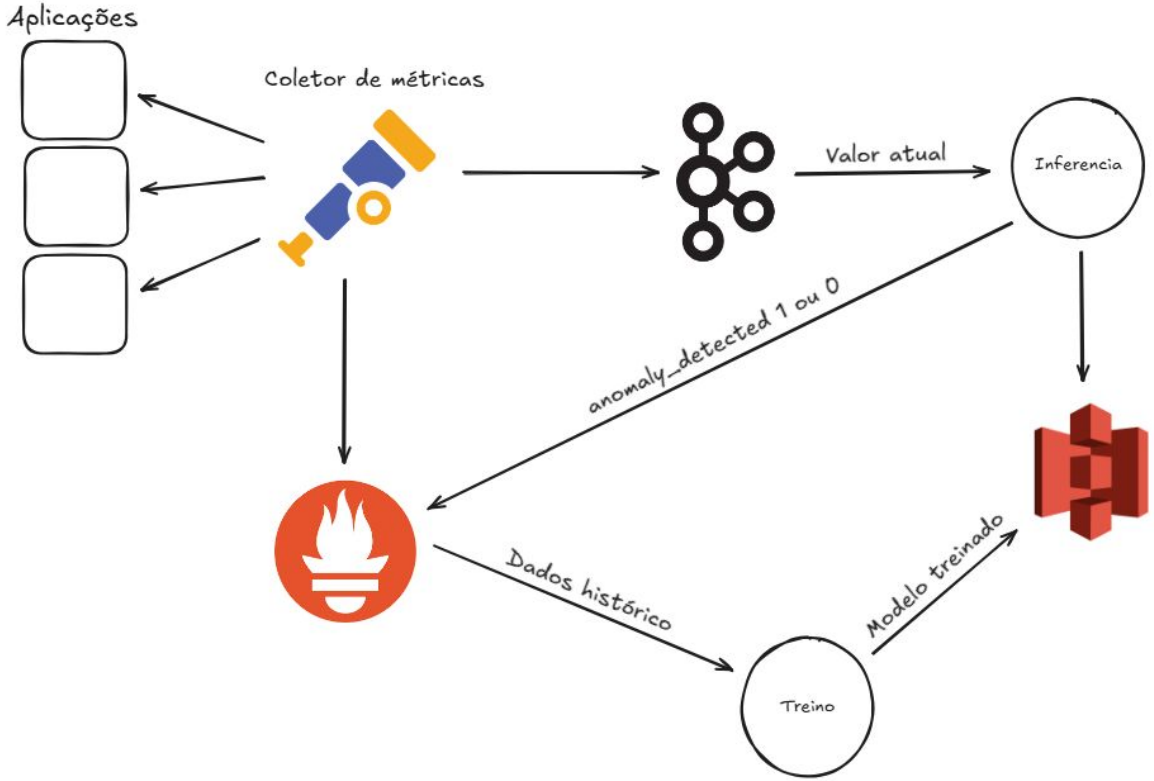
```
Zscore

(rate(http_requests_total[1m]) - avg_over_time(rate(http_requests_total[1m])[15m:]))
  / stddev_over_time(rate(http_requests_total[1m])[15m:])> 3

Isolation anomaly score gauge

anomaly_score_metric = 1
```

Exemplo detecção de anomalia



Conclusão


Latência
100
75
50
25
0
Tendencia?
Yhat
Anomalias?
Sazonalidade?
yhat_lower/yhat_upper
11:45
12:00
12:15
12:30
12:45
T-2
T-1
T
T+1
T+2

Obrigado