Modelagem IA - BTAlert

MODELO CRISP

FATEC

BureauTech

SUMÁRIO

1. Compreensão do Negócio

Identificar os especialistas na organização: Allan Vital, Henrique Campos e Kosh Narek.

Levantar e esboçar as necessidades e expectativas: A empresa parceira possui uma plataforma que 9 a cada 10 brasileiros acessam por dia, o que demanda grande disponibilidade, para garantir a eficiência das suas aplicações a empresa parceira deseja realizar a mitigação do risco de indisponibilidade dos sistemas, através de um modelo de IA, que irá prever possíveis falhas na aplicação e emitirá alertas, com isso, haverá melhora na experiência de seus usuários, garantindo a manutenção do número de usuários em sua plataforma, influenciando em suas regras de negócio, e mantendo sua rentabilidade.

Inicialmente serão coletados os dados da aplicação teste, o monitoramento será realizado através da ferramenta Prometheus, estes registros serão salvos no banco de dados do próprio Prometheus, eles serão utilizados para treinar o modelo de IA, do modelo supervisionado.

O modelo de IA será responsável por fazer a predição de possíveis indisponibilidades do sistema e emitir alertas.

Para uma boa volumetria de dados, será utilizado a ferramenta Locust para gerar o estresse da aplicação teste.

Levantamento dos hardwares e softwares: Prometheus, Grafana, Python, Spring boot, Vue, PostgreSQL, Docker Container e AlertManager.

2. Compreensão dos Dados

Conhecer e entender os dados disponíveis:

- Métricas de requisições como load, status, conexões, entre outras.
- Recursos de containers tais como CPU, memória, inbound e outbound.
- Recursos de banco de dados, como load de escrita, de leitura, entre outros.

Avaliação da qualidade dos dados disponíveis: Com base nas ferramentas utilizadas para extração das métricas, os dados já possuem uma qualidade compatível para as nossas necessidades, portanto não exigem tratamento adicional.

Verificar se a volumetria dos dados atende ao negócio: As métricas extraídas atendem as expectativas de identificação e previsibilidade de indisponibilidade da aplicação.

Saúde do Sistema é o principal indicador de disponíbilidade da nossa aplicação. Esta métrica é uma porcentagem que diz sobre a capacidade da API de processar e responder às requisições recebidas.

3. Preparação dos Dados

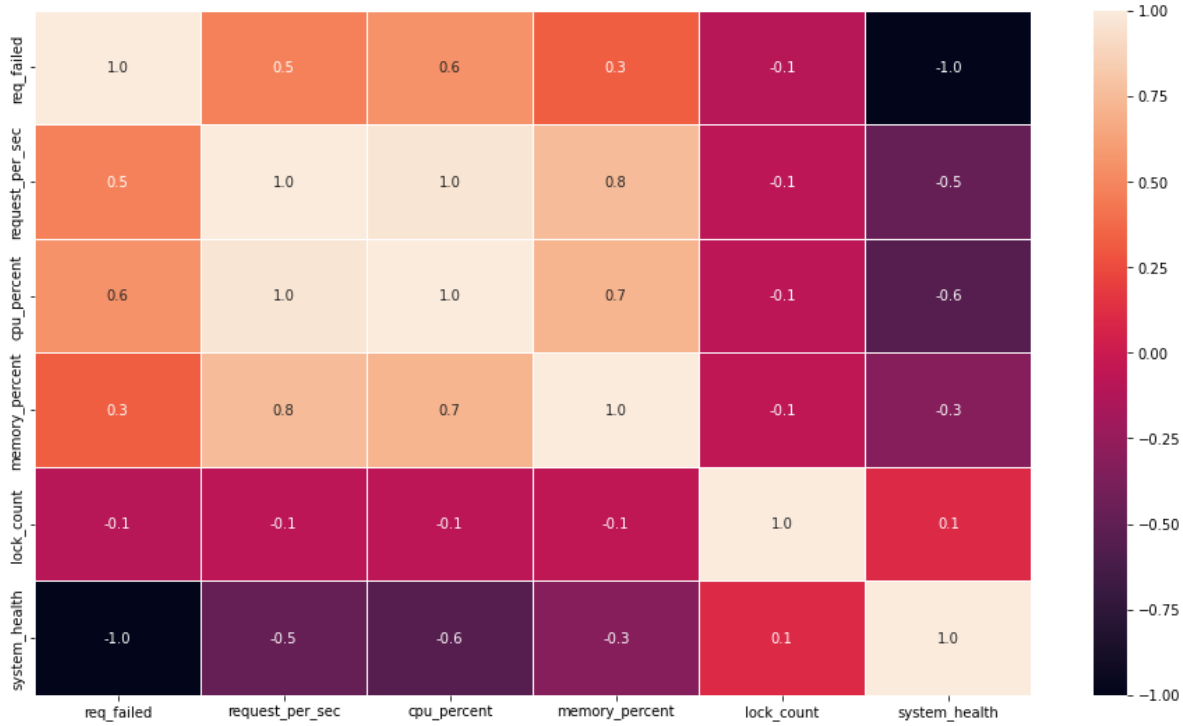
Seleção dos dados para análise:

- **container_cpu_load_average_10s:** Value of container cpu load average over the last 10 seconds.
- **container_memory_failcnt:** Number of memory usage hits limits
- **container_fs_io_current:** Number of I/Os currently in progress
- **nginx_vts_server_request_duration_seconds_count:** The total number of observations for: The histogram of request processing time
- **nginx_vts_server_request_seconds:** The average of request processing times in seconds
- **nginx_vts_server_requests_total:** The requests counter
- **nginx_vts_main_connections:** Nginx connections
- **pg_archiver_last_failed_time:** Time of the last failed archival operation
- **pg_db_blk_read_time:** Time spent reading data file blocks by backends in this database, in ms
- **pg_db_blk_write_time:** Time spent writing data file blocks by backends in this database, in ms

4. Modelagem dos Dados

column	type	range	example	description
timestamp	int64	[0,+Inf]	1652266207	timestamp when metrics were collected
req_failed	float64	[0,1]	0.0559	percentage of requests that failed
request_per_sec	float64	[0,+Inf]	20.93	requests per second sent to the server
cpu_percent	float64	[0,1]	0.842445	cpu percentage spent
memory_percent	float64	[0,1]	0.863525	memory percentage spent
lock_count	int64	[0,+Inf]	2	number of locks in the database
system_health	float64	[0,1]	0.9286	server's ability to respond to requests correctly

5. Avaliação



A matriz de correlação diz qual a influência que uma métrica tem em cima de outra. Correlações positivas nos dizem que quando uma aumenta, a outra tende a aumentar também, e a negativa diz que quando uma diminui, a outra aumenta. Com isso, verificamos que a CPU e a memória RAM possuem uma boa correlação com a saúde do sistema, e por isso foram escolhidas. Já as requisições por segundo estão diretamente ligadas com a CPU e a memória RAM, e seria redundante, logo, por este motivo não foi escolhida. E por fim, a contagem de lock do banco tem uma correlação muito fraca (mais próximo de 0, mais fraco), e foi deixada de fora do modelo da IA por conta dessa fraca correlação.

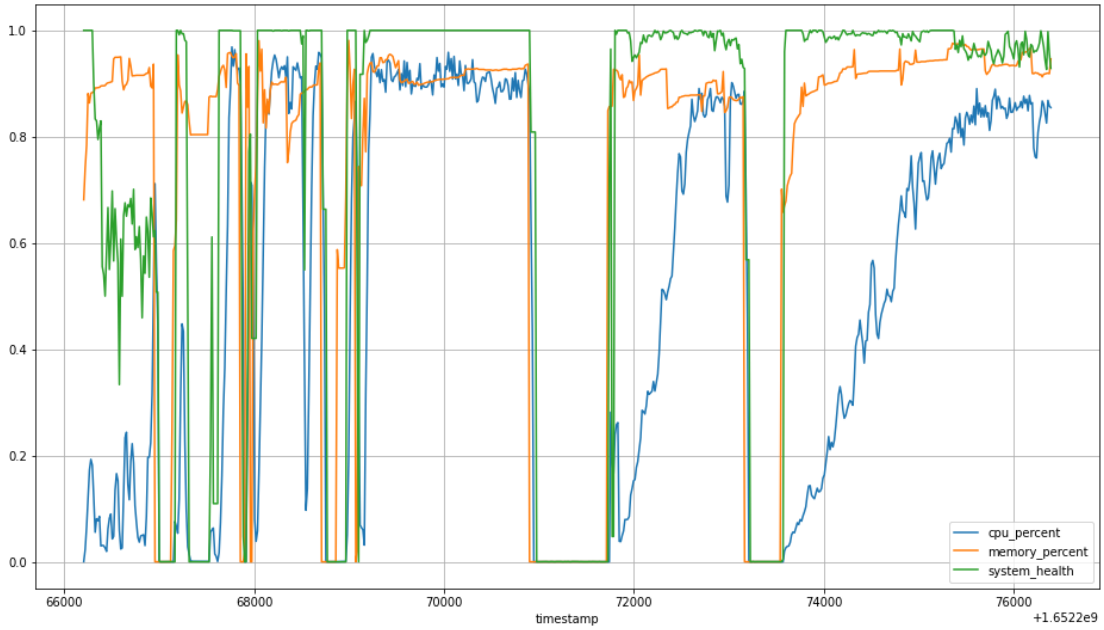
Para o presente trabalho, estamos utilizando apenas os últimos 185 dados, pois há grandes lacunas nas quais o sistema estava indisponível por muito tempo, no qual não foi reiniciado, e isso atrapalharia tanto a correlação, quanto o modelo de IA, por isso a necessidade de filtrar esses dados.

O algoritmo utilizado na solução é a LSTM, que é uma arquitetura de rede neural recorrente (RNN) que “lembra” valores em intervalos arbitrários. A LSTM é bem adequada para classificar, processar e prever séries temporais com intervalos de tempo de duração desconhecida. A insensibilidade relativa ao comprimento do gap dá uma vantagem à LSTM em relação a RNNs tradicionais (também chamadas “vanilla”), Modelos Ocultos de Markov (MOM) e outros métodos de aprendizado de sequências.

A estrutura de uma RNN é muito semelhante ao Modelo Oculto de Markov. No entanto, a principal diferença é como os parâmetros são calculados e construídos. Uma das vantagens da LSTM é a insensibilidade ao comprimento do gap. RNN e MOM dependem do estado oculto antes da emissão /

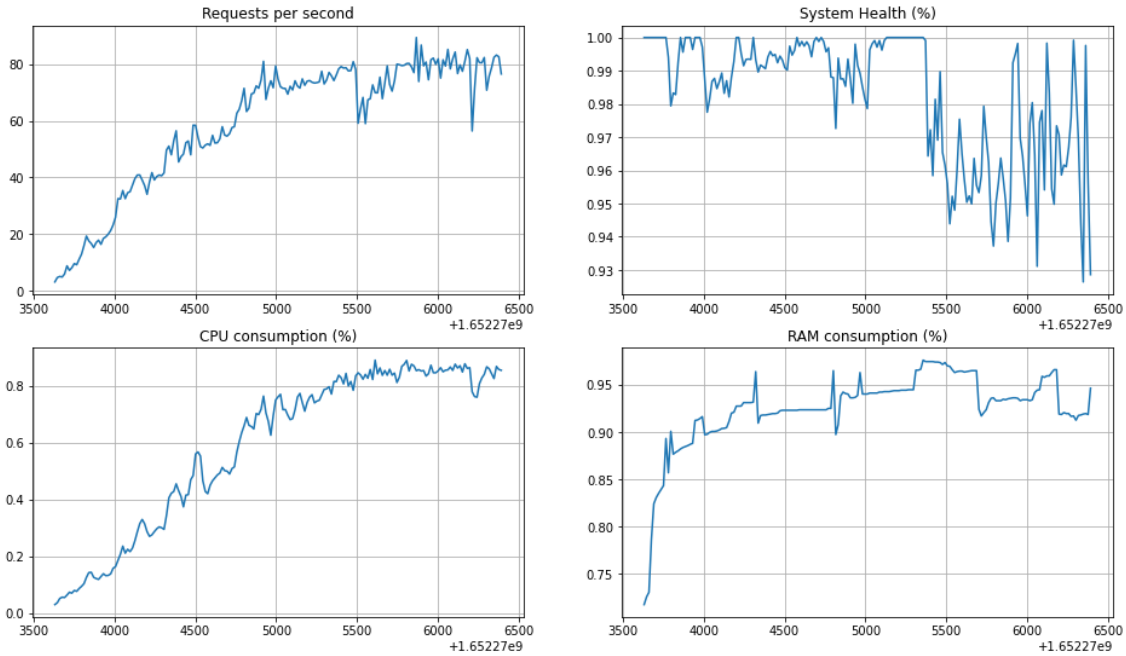
sequência. Se quisermos prever a sequência após 1.000 intervalos em vez de 10, o modelo esqueceu o ponto de partida até então. Mas um modelo LSTM é capaz de “lembrar” por conta de sua estrutura de células, o diferencial da arquitetura LSTM.

6. Desenvolvimento (implantação)



O gráfico acima mostra o uso de CPU, memória RAM e a saúde do sistema em porcentagem.

É possível perceber que conforme o uso de memória RAM e CPU aumentam, a saúde do sistema começa a cair, ficando instável. E caso a memória esteja em 0%, a saúde do sistema e o uso de CPU também ficam em 0%, pois é quando o sistema está totalmente indisponível. Notamos que a exaustão do sistema é em torno de 80%+ de uso dos recursos, e caso o sistema fique muito tempo nessa faixa de exaustão, o sistema tende a ficar indisponível após certo período na exaustão.



Os gráficos acima mostram a quantidade de requisições por segundo e a saúde do sistema, consumo de CPU e de memória RAM em porcentagem.

É possível perceber também que, conforme o uso dos recursos aumentam, a saúde do sistema fica instável, uma vez que está chegando na sua faixa de exaustão, que é por volta de 80% do uso dos recursos.

Para visualização da implantação completa deste algoritmo na solução, basta acessar este link:
https://colab.research.google.com/drive/1TjYjSaaQG8uydJfeuYG2cL9_lA4f2tr-?usp=sharing