INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SANTA CATARINA - CAMPUS FLORIANÓPOLIS

DEPARTAMENTO ACADÊMICO DE METAL-MECÂNICA

BACHARELADO EM ENGENHARIA MECATRÔNICA

PEDRO VON HERTWIG BATISTA

SISTEMA DE ANÁLISE DE DADOS DE VIBRAÇÃO COM APRENDIZADO DE MÁQUINA PARA MANUTENÇÃO PREDITIVA DE MÁQUINAS ELÉTRICAS

FLORIANÓPOLIS, 2019

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SANTA CATARINA - CAMPUS FLORIANÓPOLIS

DEPARTAMENTO ACADÊMICO DE METAL-MECÂNICA

BACHARELADO EM ENGENHARIA MECATRÔNICA

PEDRO VON HERTWIG BATISTA

SISTEMA DE ANÁLISE DE DADOS DE VIBRAÇÃO COM APRENDIZADO DE MÁQUINA PARA MANUTENÇÃO PREDITIVA DE MÁQUINAS ELÉTRICAS

Trabalho de Conclusão de Curso submetido ao Instituto Federal de Educação, Ciência e Tecnologia de Santa Catarina como parte dos requisitos para obtenção do título de Bacharel em Engenharia Mecatrônica.

Orientador: Prof. Dr. Roberto Alexandre Dias

FLORIANÓPOLIS, DEZEMBRO DE 2019

Ficha de identificação da obra elaborada pelo autor.

Batista, Pedro
SISTEMA DE ANÁLISE DE DADOS DE VIBRAÇÃO COM APRENDIZADO DE MÁQUINA PARA MANUTENÇÃO PREDITIVA DE MÁQUINAS ELÉTRICAS / Pedro Batista ; orientação de Roberto Dias.
– Florianópolis, SC, 2019.
47 p.

Trabalho de Conclusão de Curso (TCC) – Instituto Federal de Santa Catarina, Câmpus Florianópolis. Bacharelado em Engenharia Mecatrônica. Departamento Acadêmico de Metal Mecânica.
Inclui Referências.

1. Aprendizado de máquina. 2. Manutenção preditiva. 3. Análise de vibração. I. Dias, Roberto. II. Instituto Federal de Santa Catarina. Departamento Acadêmico de Metal Mecânica. III. Título.

MINISTÉRIO DA EDUCAÇÃO
SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SANTA CATARINA
CAMPUS FLORIANÓPOLIS

INSTITUTO FEDERAL
SANTA CATARINA

ATA DA BANCA FINAL DO TRABALHO DE CONCLUSÃO DO CURSO DE ENGENHARIA MECATRÔNICA DO INSTITUTO FEDERAL DE SANTA CATARINA – Nº 0050

Aos 12 dias do mês de dezembro, de 2019, às 10:00h, o estudante **Pedro von Hertwig Batista** apresentou o seu Trabalho de Conclusão de Curso para julgamento à Banca Examinadora constituída pelos seguintes integrantes: Prof Roberto Alexandre Dias (orientador/presidente da banca/IFSC), Prof Mario de Noronha Neto (IFSC) e Prof Valdir Noll (IFSC). A sessão pública de defesa foi aberta pelo Presidente da Banca, que apresentou a Banca Examinadora e deu continuidade aos trabalhos, fazendo uma breve referência ao TCC que tem como título ***Análise de Vibrações no Domínio do Tempo em Máquinas Rotativas Empregando Técnica de Aprendizado de Máquina***. Na sequência, o estudante teve até 30 minutos para a exposição de seu trabalho, e cada integrante da Banca Examinadora fez a arguição após a apresentação do mesmo. Finalmente, foi aberto um espaço aos presentes para eventuais perguntas ou comentários sobre o trabalho apresentado. Ouvidas as explicações do estudante, a Banca Examinadora, reunida em caráter sigiloso, para proceder à avaliação final, deliberou pelo conceito 10. Foi dada ciência ao estudante que a versão final do trabalho deverá ser entregue até o dia **15/02/2020**, com as devidas alterações sugeridas pela banca. Nada mais havendo a tratar, a sessão foi encerrada às 10 h 50 min, dela sendo lavrada a presente ata, que, uma vez aprovada, foi assinada por todos os membros da Banca Examinadora e pelo estudante.

Prof. Orientador / Presidente: Roberto Alexandre Dias

Prof. Avaliador 1: Mario de Noronha Neto

Prof. Avaliador 2: Valdir Noll

Pedro von Hertwig

Acadêmico: Pedro von Hertwig Batista

Agradecimentos

À minha irmã, Dani, por estar sempre presente, sempre ao meu lado, e sempre alerta.

À minha namorada, Stefani, por me aguentar e por ajudar a tornar a jornada mais doce.

Ao Rodolfo Schiavi, à Victória Zanetti, à Taciele Hemckmeier e a todas as outras amizades que construí ao longo do curso e que tanto me ensinaram, tecnicamente e humanamente.

À minha família por me apoiar nas minhas decisões e me dar o suporte necessário para alcançar o que almejei.

Ao meu orientador, Roberto Alexandre Dias, e a todos os professores da Engenharia Mecatrônica do IFSC, que sempre foram fantásticos e demonstraram uma preocupação enorme com a qualidade do ensino que cada aluno recebe.

Resumo

A condição de uma máquina afeta a qualidade e eficiência do seu trabalho, e deixar um problema chegar em estado crítico resulta em consequências negativas, podendo causar a perda do equipamento e paradas extensas em uma fábrica. A manutenção de máquinas, portanto, é uma preocupação que passou a existir junto com a criação da indústria. O presente trabalho mostra o desenvolvimento de uma plataforma que se aproveita dos avanços recentes em tecnologia de sensoriamento e aprendizado de máquina para auxiliar no processo de manutenção preditiva, identificando problemas antes que falhas graves possam ocorrer. O sistema desenvolvido para tanto conta com um sistema supervisório e uma plataforma RESTful, que recebe dados de vibração de alta frequência, os armazena e analisa o funcionamento de uma máquina para classificar seu comportamento como normal ou anômalo, gerando alertas em caso de necessidade. Os resultados alcançados mostram que é apropriado o uso de aprendizado de máquina para monitoramento de máquinas, já que algoritmos bem estruturados conseguem detectar possíveis problemas antes que eles se tornem aparentes a humanos.

Palavras-chave: Aprendizado de máquina, Manutenção preditiva, Análise de vibração.

Abstract

A machine's condition affects the quality and efficiency of its byproduct and allowing a problem to get to a critical stage results in negative consequences, potentially causing the loss of equipment and long production breaks at a factory. Machine maintenance is a concern that has come into existence along with the creation of industry. This paper shows de development of a platform that utilizes recent advances in technology for sensors and machine learning to help in the predictive maintenance process, identifying problems before serious damage can occur. To this end, the developed system contains a supervisory system and RESTful platform, that receives high frequency vibration data, stores them and analyses a machine's condition to classify its behavior as normal or anomalous, generating alerts if they're needed. The results attained show that the use of machine learning is appropriate for machine monitoring, because well structured algorithms can detect possible problems before they are apparent to humans.

Keywords: Machine learning, Predictive maintenance, Vibration analysis.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Em um meio de produção automatizado, a qualidade do produto depende, entre outros fatores, do desempenho do equipamento que o fábrica. A deterioração das condições do equipamento leva a desvios no processo de produção e queda de qualidade. Apenas a manutenção adequada pode garantir que o processo não perderá qualidade devido a desvios provocados pelo equipamento. (MARCORIN e LIMA, 2003)

Além da queda em qualidade do produto, a redução da produtividade em função de paradas não programadas é um malefício óbvio da falta de manutenção adequada. Menos óbvia é a queda de produtividade mesmo não havendo parada do equipamento. Essa condição leva a empresa a buscar a origem da deficiência em fatores como ferramental, materiais, e operadores, elevando os custos operacionais de forma mensurável, e tendo efeitos de custos não mensuráveis, como o desgaste da imagem da empresa. (MARCORIN e LIMA, 2003)

A manutenção de máquinas é, portanto, uma prática essencial na indústria. Sem manutenção as máquinas estão sujeitas a falha em seu funcionamento, o que incorre custos elevados devido à reposição de mais peças e ao tempo improdutivo perdido enquanto um reparo é realizado.

A manutenção preventiva, que ocorre em intervalos fixos de tempo de acordo com uma recomendação do fabricante, é uma das opções de manutenção. Ela envolve a troca de peças antes do momento de falha, a fim de reduzir a taxa de falha do equipamento. Essa prática, entretanto, não é a de maior eficiência porque a intervenção em equipamento saudável gera custos que podem ser evitados.

A manutenção preditiva, em contraste, parte do princípio da análise de condições de funcionamento da máquina para determinar a necessidade ou não-necessidade de intervenção. Assim, os custos com manutenção ficam restritos ao equipamento que de fato apresenta a possibilidade de falha iminente.

A análise de vibração para diagnóstico e avaliação de condição tem uma longa história de aplicação a equipamento mecânico e de energia. Muitos tipos de defeitos

aumentam o nível de vibração de máquinas, ou mudam seu comportamento de alguma forma. (LEBOLD, MCCLINTIC, *et al.*, 2000)

Nesse contexto, é chamada de “monitoração subjetiva” aquela realizada manualmente pelo pessoal de manutenção utilizando-se de seus sentidos – quando um mecânico toca uma caixa de mancal, por exemplo, pode perceber sua vibração. Assim, um mecânico mais experiente poderá fornecer maior precisão no diagnóstico de uma máquina. (OTANI e MACHADO, 2008)

Tais técnicas de monitoramento, porém, têm inúmeras deficiências: o diagnóstico é subjetivo, depende de pessoal com experiência e não fornece análise contínua.

Nos anos recentes, os sistemas de monitoramento de condições de máquinas elétricas vêm se tornando cada vez mais eficientes e sofisticados. Para monitoramento autônomo, a vibração deve ser convertida em um sinal elétrico. Alguns dos tipos de hardware utilizáveis para tal são acelerômetros piezoelétricos e MEMS (*Micro Electro Mechanical Systems*). (PANDIYAN, UMAPATHY, *et al.*, 2006)

A crescente automação dos processos de detecção têm tornado sistemas automatizados de diagnóstico e prognóstico uma ferramenta valiosa para pessoal de manutenção, e pode até substituir humanos. (HENG, ZHANG, *et al.*, 2009)

Nesse contexto, o aprendizado de máquina é uma ferramenta para auxílio no processo de tomada de decisão que vem ganhando popularidade por sua facilidade de adaptação a cenários não familiares e capacidade de resolver problemas difíceis de serem resolvidos através de modelagem matemática simples. Abordagens de aprendizado de máquina já foram utilizadas com sucesso para identificação de falhas em máquinas rotativas. (PAUDYAL, ATIQUE e YANG, 2019)

O presente trabalho almeja explorar o campo da utilização de aprendizado de máquina para aplicação em manutenção preditiva.

2 DEFINIÇÃO DE PROBLEMA

Máquinas elétricas rotativas são extensamente usadas em indústrias como as de processamento, óleo e gás. Essas indústrias precisam que as máquinas operem continuamente a um nível ótimo. O desempenho de tais máquinas é dependente principalmente da condição de seus componentes como rolamentos, transmissões, bombas, compressores, motores e geradores. (PAUDYAL, ATIQUE e YANG, 2019)

Na indústria, a aplicação da manutenção preventiva se dá através de experiência ou de recomendações do fornecedor do equipamento, e é baseada numa abordagem científica estatística. Na maioria dos casos, é realizada em intervalos de tempo regulares. (AHMAD e KAMARUDDIN, 2012)

Entretanto, planos de manutenção preventiva copiados do manual do fabricante em geral não são aplicáveis porque: (LABIB, 2004)

- Cada máquina trabalha sob condições e ambiente diferentes, e, portanto, necessita de planos diferentes.
- Projetistas de máquinas muitas vezes não têm a mesma experiência em relação às falhas das máquinas, e modos de prevenção, que os que as operam e mantêm.
- Fornecedores de máquinas podem ter uma agenda oculta de maximização de troca de peças através de planos que recomendem manutenções mais frequentes do que o necessário.

A manutenção preditiva, diferentemente da preventiva, caracteriza-se pela medição e análise de variáveis da máquina que possam prognosticar uma eventual falha. Com isso, a equipe de manutenção pode se programar para a intervenção e aquisição de peças (custo da manutenção), reduzindo gastos com estoque e evitando paradas desnecessárias. (MARCORIN e LIMA, 2003)

Para manutenção preditiva baseada em condição, são três os elementos chaves: (i) a coleta e armazenamento de informação, (ii) o condicionamento e extração de atributos de aprendizagem para dados adquiridos, e (iii) o processo de decisão para

recomendação de ações de manutenção através do diagnóstico ou prognóstico de falhas. (HENG, ZHANG, *et al.*, 2009).

O presente trabalho visa desenvolver um algoritmo no que se refere aos dois últimos pontos acima, adotando o ponto de vista de um sistema que recebe os dados de vibração de uma fonte qualquer, preferencialmente em tempo real. A partir desses dados, o algoritmo deverá predizer a iminência de uma falha.

Segundo (HENG, ZHANG, *et al.*, 2009), métodos existentes de predição de falha podem ser agrupados em três categorias principais:

- Método de confiabilidade tradicional (predição baseada em eventos passados – manutenção preventiva)
- Método prognóstico (predição baseada em monitoramento de condição)
- Método integrado (predição baseada em eventos passados e em monitoramento de condição)

A primeira e última categoria, que envolvem a distribuição estatística de eventos passados, não são sempre possíveis porque os dados de eventos passados só são úteis para outros equipamentos em condições idênticas de trabalho (HENG, ZHANG, *et al.*, 2009). Máquinas na indústria frequentemente operam sob condições únicas e não têm tais dados históricos disponíveis.

Para o método prognóstico baseado em monitoramento de condição, as predições se dividem, ainda, em três subcategorias (CUBILLO, PERINPANAYAGAM e ESPERON-MIGUEZ, 2016):

- Modelo físico
- Modelo orientado a dados
- Modelos híbridos, que unem as duas categorias anteriores (CUBILLO, PERINPANAYAGAM e ESPERON-MIGUEZ, 2016)

A predição baseada em condição com modelo físico (e, por extensão, também a com modelo híbrido) implicam na construção de um modelo matemático abrangente que descreva as relações físicas do sistema e seus modos de falha. Isso traz algumas

vantagens ao processo; notavelmente, como modelos físicos não dependem da comparação a dados de treinamento, são reduzidos os erros associados a extrapolação de dados e o modelo é imediatamente efetivo. Também há a possibilidade de simular cenários para os quais a obtenção de dados seria rara ou difícil, e deles determinar o funcionamento da máquina em condições adversas (STRINGER, SHETH e ALLAIRE, 2012).

Entretanto, esses modelos apresentam pouca escalabilidade e a adoção de modelagem física é cara no que toca ao fato de que cada máquina monitorada requer um modelo detalhado de seu funcionamento normal e de seus modos de falha. O modelo é afetado de modo complexo pela condição de funcionamento a que a máquina em questão é submetida e pela interação entre seus componentes internos.

Em contraste a isso, um modelo orientado a dados tem menor necessidade de conhecimento específico do domínio em que a máquina está inserida, apresenta maior simplicidade, escalabilidade e custo de desenvolvimento reduzido. Outra vantagem é o fato de que uma vez treinado, seu uso é computacionalmente mais eficiente do que o de modelos físicos (JIA, HUANG, *et al.*, 2018).

Sendo crítica a necessidade de geração de alertas para minimização de custos dentro da indústria, o cerne do problema consiste na detecção de comportamento anormal que venha a indicar um princípio de falha no equipamento. A detecção de pontos atípicos é um tema comum no aprendizado de máquina, sendo central em muitas aplicações relacionadas a segurança; uma degradação do funcionamento típico pode resultar em falhas por exemplo em turbinas de avião, e um ponto anômalo em uma imagem de satélite pode indicar a presença de uma mina terrestre. (HODGE e AUSTIN, 2004)

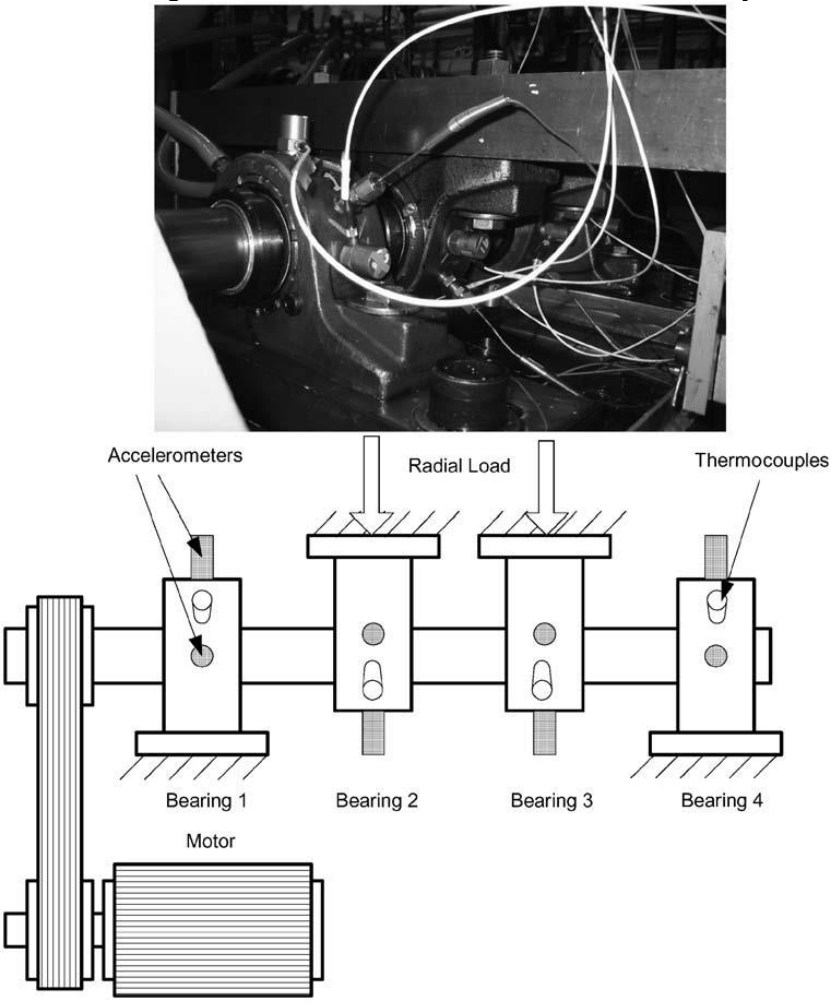
2.1 Conjunto de dados NASA

O conjunto de dados a ser utilizado para avaliação do algoritmo é o conjunto desenvolvido gerado para o artigo *“Wavelet Filter-based Weak Signature Detection Method and its Application on Roller Bearing Prognostics”* (QIU, LEE, *et al.*, 2006) e disponibilizado publicamente no repositório de dados abertos de prognóstico da NASA.

Foram gravados três conjuntos de dados referentes a experimentos de testes até a falha de rolamentos. Cada conjunto de dados consiste em arquivos individuais de dados de vibração captados em janelas de 1 segundo. Cada arquivo de janela foi gravado a um intervalo de 10 minutos de distância do outro.

Foram utilizados 4 rolamentos Rexnord ZA-2115 distribuídos ao longo de um eixo rotativo. A rotação do eixo foi de 2000 RPM aplicados por um motor AC via correia. Uma carga radial de 6000 libras foi aplicada no eixo, em contato com dois dos rolamentos, conforme mostrado na Figura 1.

Figura 1 - Bancada de testes de dados de vibração



Fonte: (QIU, LEE, *et al.*, 2006)

Os acelerômetros *PCB 353B33 High Sensitivity Quartz ICP* foram posicionados próximos a cada um dos rolamentos. No primeiro dos três conjuntos de dados, oito acelerômetros foram usados, dois para cada rolamento; determinou-se que os dados resultantes de dois acelerômetros em cada rolamento são redundantes, e então nos dois conjuntos seguintes, somente um acelerômetro está presente em cada rolamento.

Os experimentos em cada um dos conjuntos de dados foram parados após a falha de um ou mais dos rolamentos. A Tabela 1 mostra os resultados de cada um dos testes.

Tabela 1 - Resultados dos experimentos

Conjunto	Duração do experimento	Falha(s) apresentada(s)
1	827,6 horas	Defeito no anel interior do rolamento 3 e esfera no rolamento 4
2	163,8 horas	Defeito no anel exterior do rolamento 1
3	753,6 horas	Defeito no anel exterior do rolamento 3

3 PROPOSTA DE SOLUÇÃO

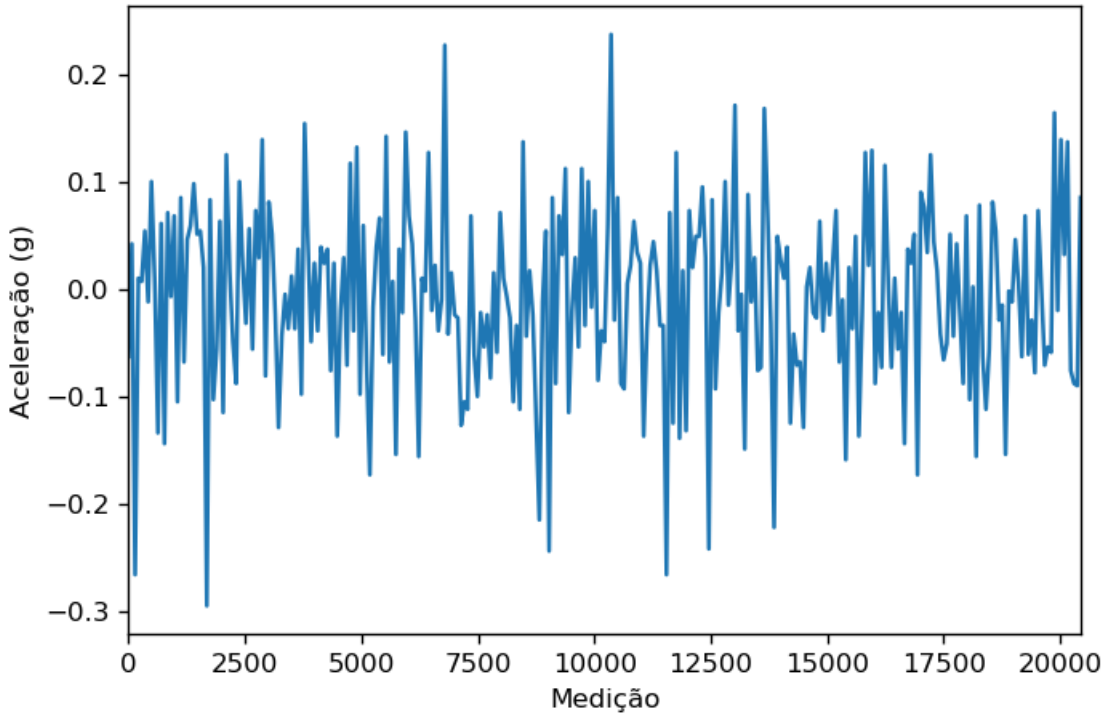
Tendo em vista as vantagens da manutenção preditiva baseada em análise de vibração e tendo como prioridade a flexibilidade do algoritmo para que seja aplicável a diversas situações na indústria, o presente trabalho desenvolverá um algoritmo e plataforma voltados à análise de vibração para manutenção baseada em condição orientada a dados.

Uma plataforma deverá receber dados contínuos de uma dada máquina elétrica a fim de treinar seu modelo de monitoramento de condição.

O cerne do trabalho encontra-se na detecção de anomalias e alerta de desvio do comportamento normal do funcionamento de uma máquina com padrão de vibrações conhecido.

Para o desenvolvimento e teste de confiabilidade dos algoritmos desenvolvidos, esse trabalho se utilizará como referência o conjunto de dados referente a um teste de bancada disponibilizado pela NASA (QIU, LEE, *et al.*, 2006). No teste, rolamentos foram submetidos a uma carga radial até que chegassem a um ponto de falha. Um exemplo dos dados brutos contidos no experimento pode ser visualizado na Figura 2.

Figura 2 – Visualização referente a um segundo de dados brutos



Fonte: Autor

O gráfico mostra os dados de aceleração gravados por um dos sensores, para um intervalo de 1 segundo. O eixo X indica o número da medição, feita a uma taxa de aquisição de 20 kHz. O conjunto de dados é composto por janelas como essa, de 1 segundo cada, gravadas em intervalos de 10 minutos.

3.1 Classificadores temporais e características agregadas

Um braço da detecção de pontos fora da curva é a detecção temporal de anomalias, que se preocupa com dados ao longo do tempo. Nesse contexto, ainda há várias subdivisões de metodologias que podem ser utilizadas: séries temporais ou multidimensionais, análise ponto-a-ponto ou janelada, séries contínuas ou discretas, e disponibilidade ou não-disponibilidade de dados anômalos prévios. (GUPTA, GAO, *et al.*, 2014)

Assim, a solução apresentada explorará as metodologias de que utilizam a série de pontos em ordem cronológica (série temporal) que vem dos sinais de vibração de uma máquina elétrica monitorada.

Apesar de que, em princípio, a informação necessária para observação da condição de uma máquina está contida nos dados brutos de vibração captados, na prática a comparação direta entre um sinal saudável e um sinal que indica avaria não é totalmente efetiva. Isso se dá porque a alta variação e comportamento errático de dados brutos torna difícil a comparação entre assinaturas do sinal. (LEBOLD, MCCLINTIC, *et al.*, 2000)

Faz-se interessante também, então, que a solução desenvolvida se utilize da extração de características mais indicativas a partir dos dados brutos captados de sensores. Na solução proposta optou-se por utilizar as seguintes propriedades:

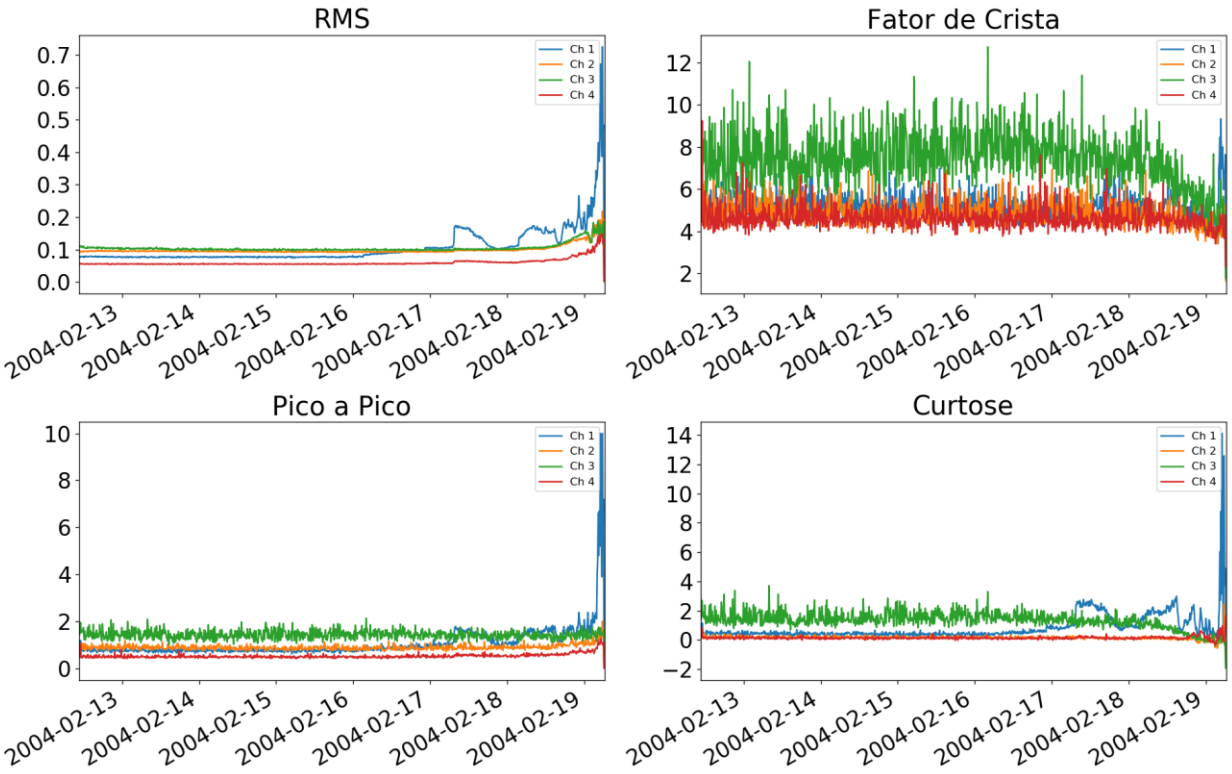
- **RMS** – *root mean square,* é uma característica que mede a energia em uma vibração de assinatura. Pode ser bastante efetiva na detecção de desbalanceamento em sistemas rotativos. (LEBOLD, MCCLINTIC, *et al.*, 2000)
- **Curtose** – é definida como o quarto momento da distribuição e mede quão lisa ou pontuda é a distribuição, em relação a uma distribuição normal. É indicativa de grandes picos no conjunto de dados. (LEBOLD, MCCLINTIC, *et al.*, 2000)
- **Pico a pico** – diferença entre maior e menor valores registrados. Dá uma ideia da amplitude de sinal de vibração.
- **Fator de crista** – definida como sendo o maior pico da janela dividido pelo valor RMS, um alto fator de crista indica valores de amplitude muito acima da média presentes no conjunto de dados.

Essas características são escalares obtidas através da aplicação de uma operação matemática sobre diversos pontos de uma série temporal. Assim, podem ser mais diretamente comparadas entre janelas do mesmo tamanho em instantes diferentes.

Na Figura 3 podemos ver o resultado do processamento das características aplicadas ao conjunto de dados fornecido pela NASA. Nesse caso, foram computadas em janelas de um segundo com frequência de aquisição de 20 kHz e medições feitas a cada 10 minutos, as janelas disponibilizadas pelo conjunto de dados.

A legenda “Ch 1 .. 4” se refere aos canais de medição (acelerômetros referentes a cada um dos 4 rolamentos monitorados).

Figura 3 - Escalares sob janelas



Fonte: Autor

No experimento mostrado na Figura 3 (um de três realizados sob as mesmas condições), o rolamento referente ao canal 1 (legenda “Ch 1”) veio a falha com defeito no anel exterior do rolamento.

É possível observar mais claramente a alteração do valor RMS do sinal de vibração no rolamento em questão. Um aumento em RMS está ligado a uma energia maior dispersada por meio da vibração, e pode ser um indicador efetivo na detecção de desbalanceamentos de sistemas rotatórios. (LEBOLD, MCCLINTIC, *et al.*, 2000)

3.2 Classificadores agrupadores

A solução a ser desenvolvida, portanto, pode ser beneficiada por abordagens diferentes, cada uma com vantagens e desvantagens.

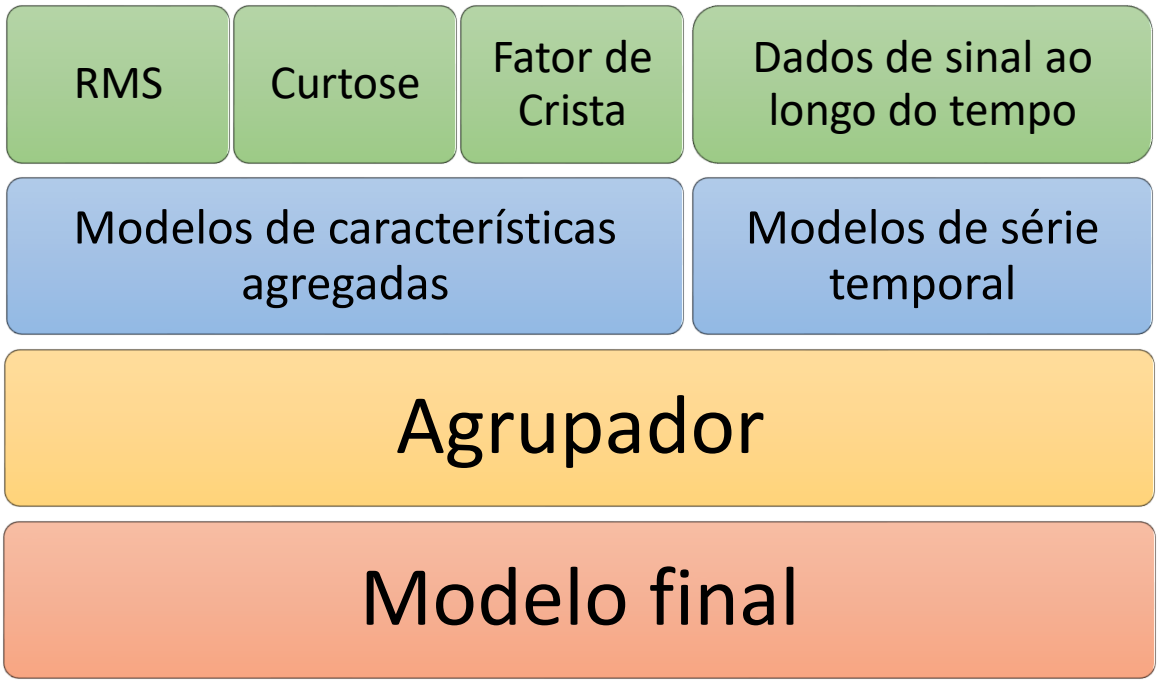
Felizmente, um trabalho desenvolvido no campo do aprendizado de máquina não obrigatoriamente fica restrito a um ou outro método; no caso do presente trabalho, a análise temporal ou análise de características escalares agregadas ao longo do tempo.

Agrupadores (no inglês, *ensembles*) são conjuntos de classificadores que almejam aumentar a confiabilidade de um sistema como um todo através da união de diversos modelos.

Um agrupador, também chamado de classificador múltiplo ou comitê, é um conjunto de classificadores individuais cujas predições são combinadas para criar uma nova predição. Agrupadores têm se mostrado um jeito eficiente de melhorar a precisão de um problema complexo, transformando-o em vários subproblemas. (KRAWCZYK, MINKU, *et al.*, 2017)

A Figura 4 demonstra a hierarquia de elementos presentes em um possível sistema agrupador para manutenção baseada em vibração.

Figura 4 - Cascata de classificador por agrupamento



Fonte: Autor

No primeiro nível, encontram-se os dados brutos e características extraídas de janelas de tempo de sinais. Esses dados e características por si só não predizem a condição de um sistema monitorado. Para tanto, é necessário que sejam consumidos por modelos de aprendizado de máquina (segundo nível). Os modelos são treinados com observações sabidamente não-anômalas e depois, dada uma nova observação, a classifica como normal ou anômala.

O agrupador, por sua vez, recebe as classificações de diversos modelos sobre a mesma observação e disso determina a classificação final. É importante assegurar que o processo de combinação do agrupador seja robusto, porque uma sintetização não apropriada pode ser prejudicial à performance do modelo. (ZHAO, NASRULLAH, *et al.*, 2019)

3.3 Tecnologias específicas utilizadas

O presente trabalho procura utilizar tecnologia existente na forma de bibliotecas *open-source* como ponto de partida para construção dos algoritmos. Pela familiaridade do autor com a linguagem de programação Python, e o fato de que excelentes recursos estão disponíveis e sendo ativamente desenvolvidos na área de aprendizado de máquina, essa foi a principal tecnologia escolhida, da qual as outras seguem.

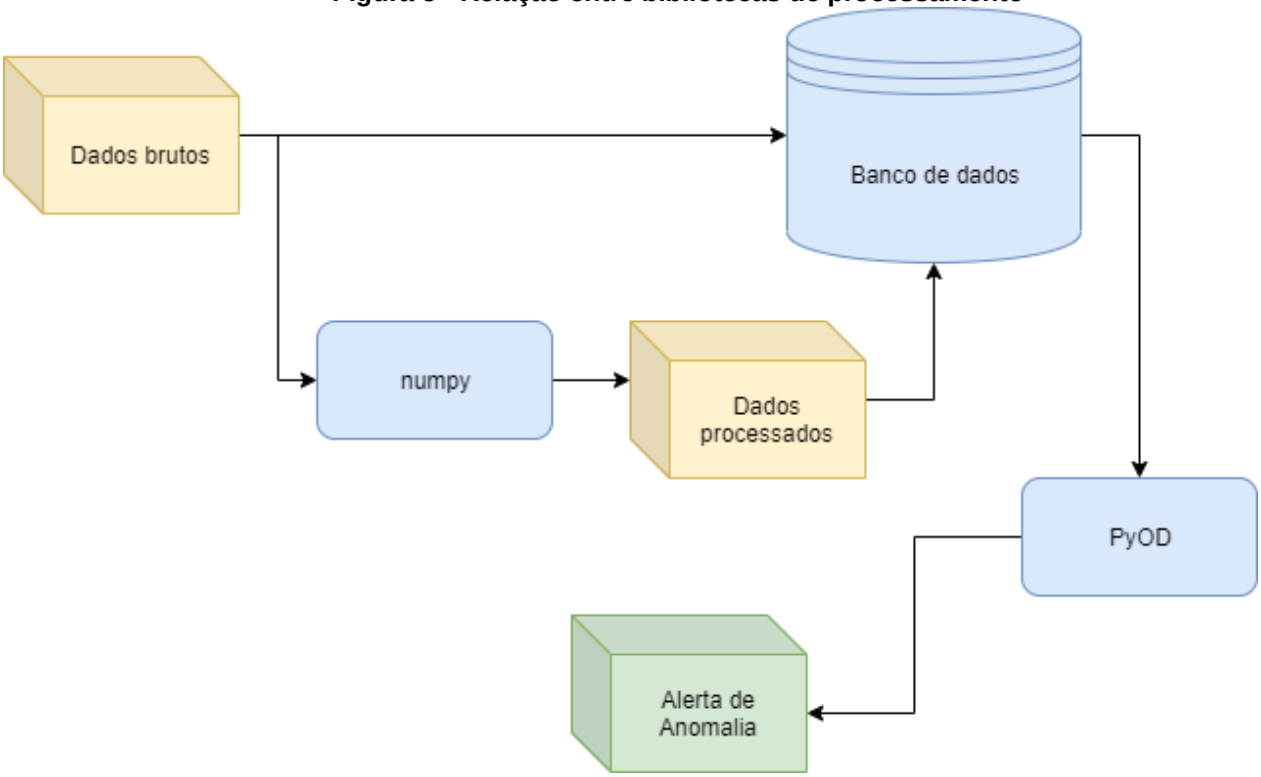
Para operações rápidas com matrizes e gerenciamento de dados tabulares, foram utilizadas as bibliotecas `numpy` e `pandas`. O Python, por ser uma linguagem interpretada, apresenta certas ineficiências que o `numpy` em especial consegue contornar.

Esse contorno é feito principalmente através da chama de funções em C, quando possível, uso de estruturas de dado específicas e sem *overhead* associado a objetos de linguagens interpretadas, e utilização de vetorização de operações para realizar múltiplas instruções de processamento simultaneamente. Tudo isso é disponibilizado ao usuário através de uma API Python em que as partes complexas são abstraídas, de modo a tornar o código escrito com a biblioteca simples e compreensível.

O `numpy` foi portanto utilizado para o cálculo de valores escalares extraídos dos dados brutos de aceleração recebidos.

Para o treinamento dos modelos de aprendizado de máquina, optou-se por utilizar o `PyOD`, ou biblioteca *Python Outlier Detection.* Ela conta com uma variedade de modelos de aprendizado de máquina específicos para detecção de anomalias. (ZHAO, NASRULLAH e LI, 2019)

Figura 5 - Relação entre bibliotecas de processamento



Fonte – Autor

A Figura 5 mostra a relação entre as bibliotecas de processamento de dados. O `numpy` é usado para processar os dados brutos, e esses novos dados são armazenados junto aos originais. O `PyOD`, então, usa desses dados para criar modelos de aprendizado de máquina e gerar alertas de pontos anômalos.

3.4 Plataforma RESTful

Para maior utilidade em uma situação prática, foi desenvolvido não só o algoritmo (classificador agrupador) para detecção de anomalias, mas também uma plataforma RESTful que possa receber tais dados em tempo real, armazená-los, treinar os modelos classificadores, gerar alertas e visualizações da condição de máquinas monitoradas.

Para tanto, foram escolhidas tecnologias de código aberto que facilitem a criação do sistema.

3.4.1 Orquestração de sistemas

O crescimento da globalização de serviços em rede tem aumentado a demanda por aplicações que sejam flexíveis e escaláveis; quer dizer, que tenham a possibilidade de se adaptar, em questão de consumo de recursos de hardware, de acordo com a demanda de momento a momento.

Tal demanda comandou o desenvolvimento de paradigmas que atendessem a esses requisitos. Um deles é a *containerização* de processos. Um *container* é um “balde” de software que compõe todas as dependências necessárias para se executar um programa de software independentemente. (DOCKER)

Além de aumentar a elasticidade de um serviço, o uso de *containers* permite que a infraestrutura seja tratada como código, dando maiores garantias de estabilidade e permitindo facilidades como maior testabilidade e fácil versionamento e documentação.

Um dos serviços de containerização disponíveis é a solução *open*-source Docker. Essa foi a ferramenta escolhida, junto com o orquestrador de containers docker-compose, para serem as ferramentas para especificação da infraestrutura da plataforma, por sua simplicidade e grande quantidade de recursos e documentação disponível. O seguinte trecho é um exemplo do arquivo de configuração d*ocker-compose.yml* que especifica dois dos processos utilizados pela plataforma:

```
version: '3.5'
services:
  timescaledb:
    container_name: tcc_timescaledb
    image: timescale/timescaledb
    environment:
      POSTGRES_USER: postgres
      POSTGRES_PASSWORD: ...
      POSTGRES_DB: postgres
    ports:
      - 5432:5432
    volumes:
      - pgdata:/var/lib/postgresql/data
```

```
    restart: always
  grafana:
    container_name: tcc_grafana
    image: grafana/grafana:6.3.5
    ports:
      - 6001:3000

volumes:
  pgdata: {}
```

Pode se observar que duas das imagens utilizadas são a do TimescaleDB, serviço de banco de dados relacional, e Grafana, plataforma de visualização de dados. São configurados também o mapeamento de portas entre os serviços e o sistema operacional, configurações de variáveis de ambiente que definem as credenciais do banco de dados, e volumes de armazenamento de dados que podem ou não ser compartilhados entre containers diferentes.

3.4.2 Armazenamento de dados

O armazenamento de séries temporais é um braço da tecnologia de banco de dados em geral. Apesar de ser possível guardar tais dados em bancos de dados tradicionais, existem projetos feitos especificamente para esse propósito.

Para a plataforma RESTful desenvolvida, optou-se por usar o TimescaleDB, projeto que estende o PostgreSQL, banco de dados relacional tradicional *open-source* amplamente usado. Isso traz vantagens como o uso de ferramentas projetadas para uso com PostgreSQL, como bibliotecas ORM (*Object Relational Mapping*) que servem como um nível de abstração sobre a camada de banco de dados, tornando o código mais semântico do que o uso de SQL. Por exemplo, definimos uma estação de monitoramento do seguinte modo:

```
class Station(BaseModel):
    name = CharField(primary_key=True, unique=True)
    description = CharField()
    is_training = BooleanField()
```

E em seguida, criar uma estação:

```
new_station = Station.create(name='lathe_1',
                             description='Lathe #AB123',
                             is_training=True)
```

Ou buscar todas as estações que estejam em modo de aquisição de dados para treinamento:

```
stations_in_training = Station.get(Station.is_training == True)
```

O uso do TimescaleDB é vantajoso nessa situação quando comparado ao PostgreSQL puro, porque permite executar *queries* orientadas a séries temporais com maior performance e eficiência. Em sistemas de maior escala, onde particionamento de bancos de dados entre diferentes máquinas é necessário, o TimescaleDB torna o particionamento por tempo o comportamento padrão. Assim, como é comum para a maioria das aplicações de geração e análise de dados em tempo real, o acesso a dados recentes torna-se mais rápido.

3.4.3 Geração de visualizações

Para a geração de visualização dos dados ao longo do tempo, optou-se por utilizar o Grafana, solução de código aberto que permite a fácil criação de *dashboards* interativos com diversos tipos de gráficos configuráveis.

O Grafana possui integração com diversos serviços de monitoramento, e pode buscar dados diretamente de um banco de dados, como o TimescaleDB. Isso o torna uma solução adequada à necessidade da solução.

3.4.4 Framework de servidor

A linguagem Python, escolhida para o projeto, tem como uma de suas utilizações mais populares o desenvolvimento de serviços *back-end*[1] RESTful, como o proposto por esse trabalho. *Frameworks* populares incluem Django, sistema completo com uma ORM

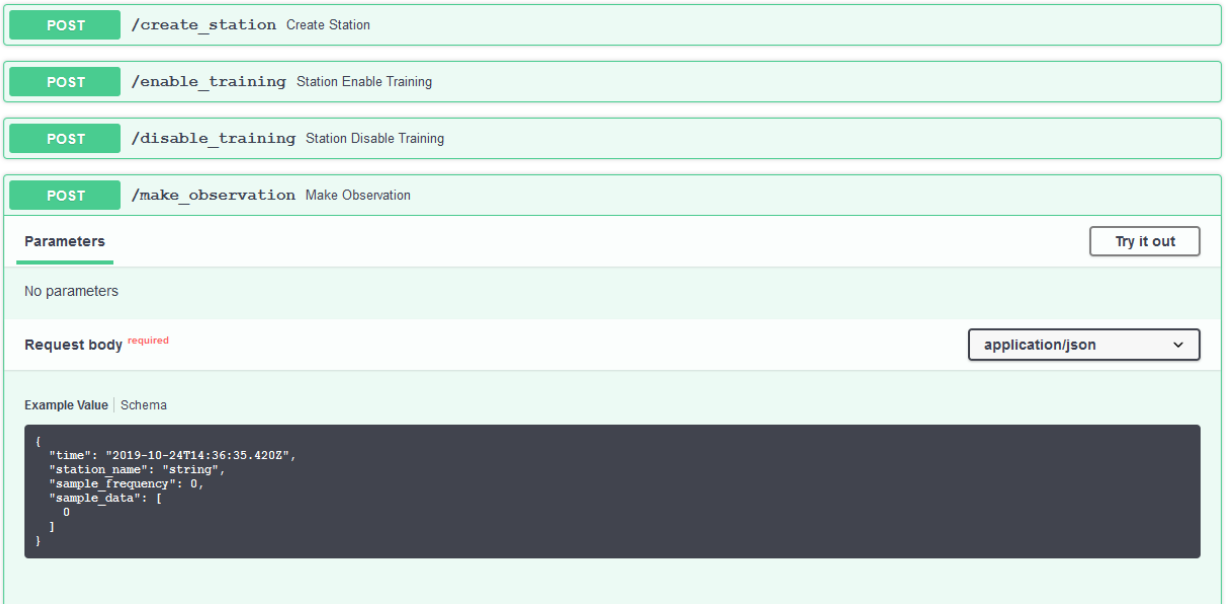
[1] *Back-end*: software que realiza o processamento e armazenamento de dados, mas por si só não tem interface com a qual o usuário possa interagir.

e diversas facilidades incluídas, e Flask, uma *micro-framework* que facilita o desenvolvimento sendo extremamente leve e concisa. (HERMAN, 2019)

A tecnologia selecionada para esse projeto é uma ainda emergente em popularidade, mas inspirada no sistema de *micro-frameworks*: o FastAPI. As vantagens da tecnologia escolhida incluem o uso do paradigma assíncrono recentemente introduzido na linguagem Python. O paradigma assíncrono permite a um programa (nesse caso, o servidor RESTful) que se utilize de concorrência para não blocar a execução do programa durante operações de E/S, sem o custo computacional e de complexidade da utilização de *threads* ou múltiplos processos.

Além do uso de assincronia, o FastAPI possui validação automática de dados de entrada e saída da API, e a geração automática de documentação conforme a especificação *OpenAPI*.

Figura 6 - Exemplo de documentação gerada



Fonte – Autor

A Figura 6 mostra um exemplo da documentação on-line automaticamente gerada e disponibilizada pelo servidor. São mostradas as rotas HTTP/S e métodos utilizados (para os mostrados na figura, todos POST). Ao clicar em uma das rotas, é mostrado um

exemplo de requisição válida e (não mostrado na figura) um exemplo de saída, além dos possíveis erros gerados e porque ocorrem.

3.5 Lógica de implementação

A plataforma desenvolvida trabalha com o conceito de estações de monitoramento e observações. Uma observação é atrelada a uma estação, e consiste em um vetor de dados adquiridos em alta frequência.

A fim de poupar banda, espaço de armazenamento e, se for o caso, bateria do dispositivo integrado ao acelerômetro, o monitoramento não é feito de forma contínua, e sim através da captação de dados de alta frequência em janelas curtas de tempo (por exemplo, janelas de 1 segundo a cada 10 minutos).

Os dados são então transmitidos à plataforma via API RESTful.

3.5.1 Criação de estações de monitoramento

Antes que a plataforma consiga receber os dados, é necessário que seja criada uma estação de monitoramento, equivalente a um sensor, para que os dados de vibração recebidos sejam ligados a um sensor específico.

Para propósitos de identificação, a estrutura de dados correspondente a uma estação foi equipada com os seguintes parâmetros:

- Nome
- Descrição
- Treinamento em progresso

O campo “Nome” tem como objetivo armazenar uma chave primária da estação em questão. O campo “Descrição” pode abrigar qualquer outro dado que ajude a identificar a estação, como seu posicionamento dentro de uma fábrica. O campo “Treinamento em progresso” é uma *flag* (assume somente os valores verdadeiro/falso) e indica se os dados coletados dessa estação devem ser tratados como dados de treinamento, ou se o treinamento já foi realizado e alertas devem ser gerados.

Na plataforma desenvolvida, a criação de uma estação de monitoramento se dá através de uma requisição POST à rota `/create_station`.

3.5.2 Aquisição e armazenamento de dados

O envio de uma janela de dados de alta frequência é feito através de uma requisição POST à rota `/make_observation`.

Os parâmetros obrigatórios são o nome da estação de monitoramento à qual a observação pertence, a frequência de aquisição em Hertz, e os dados de alta frequência como um vetor de números de ponto flutuante.

Opcional é o parâmetro de horário da observação. Se omitido, usa-se o horário atual do servidor.

Quando recebe a requisição, o servidor valida os dados recebidos (verificando se a estação é válida e se os tipos de dados recebidos coincidem com os esperados), calcula os valores pico a pico, RMS, curtose e fator de crista referentes à janela de dados, e os insere, junto com os dados brutos originais, no banco de dados.

Esse pré-cálculo é feito para que o sistema não tenha que recalcular todos os parâmetros para gerar uma visualização por exemplo, e consome consideravelmente menos espaço do que os dados brutos, sendo um custo comparativamente pequeno.

3.5.3 Treinamento de modelos

O treinamento dos modelos de aprendizado de máquina para detecção de anomalias em novos dados ocorre quando uma estação deixa de estar em modo de treinamento em progresso, e passa a estar em modo de monitoramento e geração de alertas.

Quando recebe o comando de desabilitar o modo de treinamento, o sistema usa os dados de treinamento recebidos até então para treinar o modelo classificador agrupador. O modelo escolhido para tanto foi o *Locally Selective Combination in Parallel Outlier Ensembles* (LSCP) (ZHAO, NASRULLAH, *et al.*, 2019).

Sendo um classificador agrupador, o LSCP recebe inicialmente vários sub-modelos de detecção de anomalias não treinados. Ele então realiza o treinamento individual de cada um deles.

Para tanto, ele seleciona combinações aleatórias de parâmetros de treinamento de modo a ter sub-modelos diversificados; por exemplo, para uma instância de um sub-modelo pode fornecer apenas dados de RMS e pico a pico, para outra pode fornecer RMS e curtose, e para outra pode fornecer todos os parâmetros disponíveis.

O resultado é que diferentes possíveis correlações são analisadas, e os sub-modelos contidos são diversificados a ponto de terem precisão maior quando usados em conjunto.

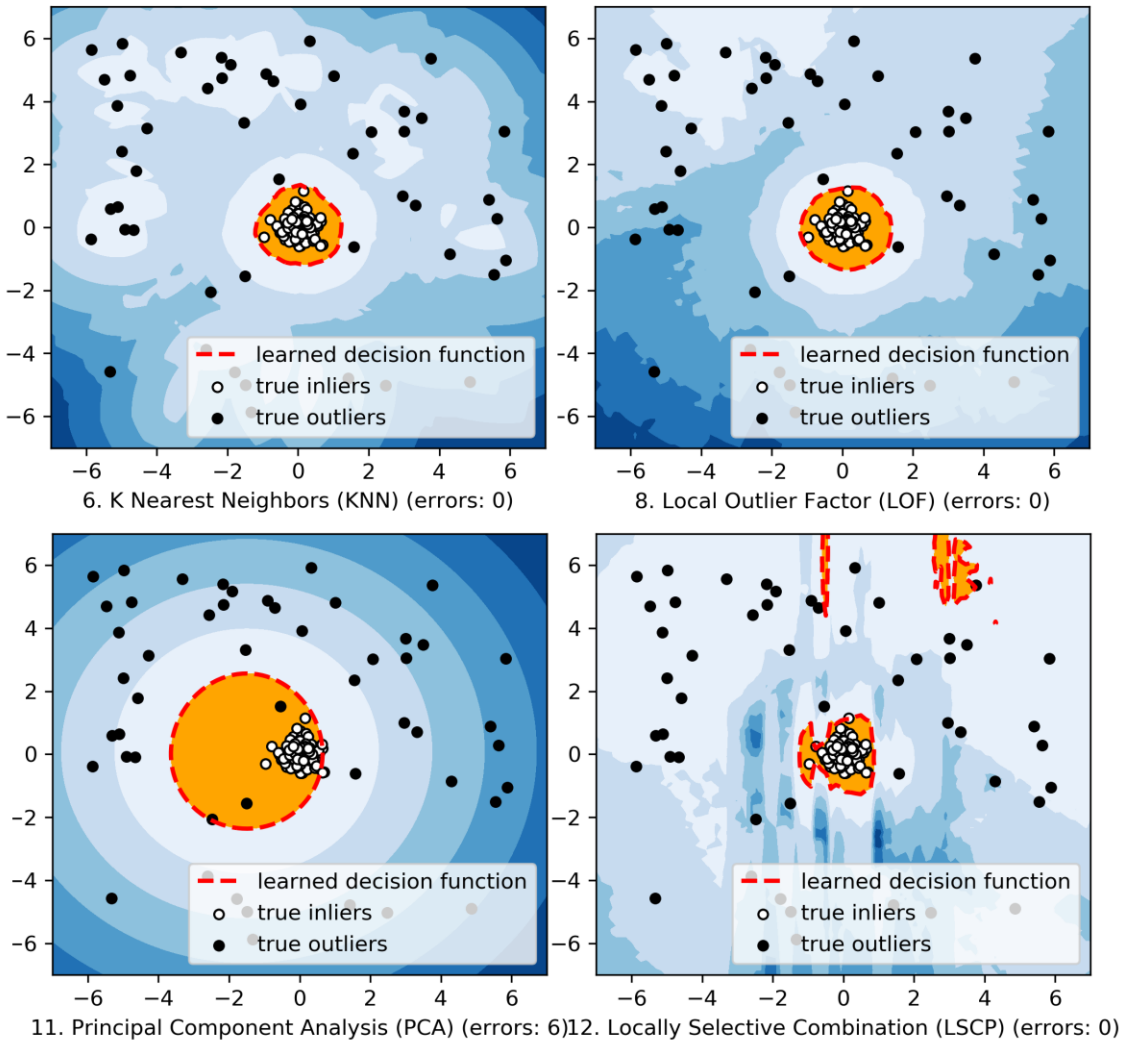
Os tipos de sub-modelos escolhidos para utilização do LSCP foram os seguintes:

- kNN – Detecção de ponto anômalo baseado na distância de um ponto a seu vizinho mais próximo de nível *k* (RAMASWAMY, RASTOGI e SHIM, 2000)
- LOF – Detecção de ponto anômalo baseado no princípio de *local outlier factors*, analisando a densidade de pontos e determinando a probabilidade de um dado ponto não pertencer ao conjunto padrão (BREUNIG, KRIEGEL, *et al.*, 2000)
- PCA – *Principal Component Analysis*, modelo baseado na diferença entre os valores de características de maior peso nos dados. (SHYU, CHEN, *et al.*, 2003)

Para que se haja uma boa utilização de todas as características (de modo que nenhuma seja descartada) mesmo com a seleção randomizada do algoritmo LSCP, cada tipo de sub-modelo gera 5 instâncias. Assim, temos 5 detectores kNN, 5 LOF e 5 PCA, totalizando 15 instâncias gerenciadas pelo agrupador LSCP.

A Figura 7 mostra o processo de decisão para um *dataset* não relacionado ao presente trabalho, e ajuda a criar uma intuição sobre o funcionamento dos algoritmos escolhidos quando aplicado a duas dimensões.

Figura 7 - Visualização de modelos treinados



Fonte – (ZHAO, NASRULLAH e LI, 2019), modificado

Na figura, observa-se em laranja a área classificada como não-anômala, e nas áreas em azul, o nível de certeza de que os pontos mostrados não se enquadram na área de normalidade. Um azul mais claro representa certeza menor. Pontos brancos são *"inliers"*, ou pontos dentro da curva, e pontos pretos são *"outliers"*, observações anômalas.

É interessante observar que as áreas compreendidas pelos modelos simples - KNN, LOF e PCA – são de forma simples e não captam muita complexidade. Já a

classificação do LSCP se dá de forma mais complexa, e não compreende somente uma área aproximada a um círculo, mesmo que as outras áreas nesse caso sejam predominantemente ruidosas. Da maior complexidade geométrica das áreas de certeza, podemos intuir que esse classificador é mais propício a captar nuances presentes em conjuntos de dados.

3.5.4 Armazenamento de modelos

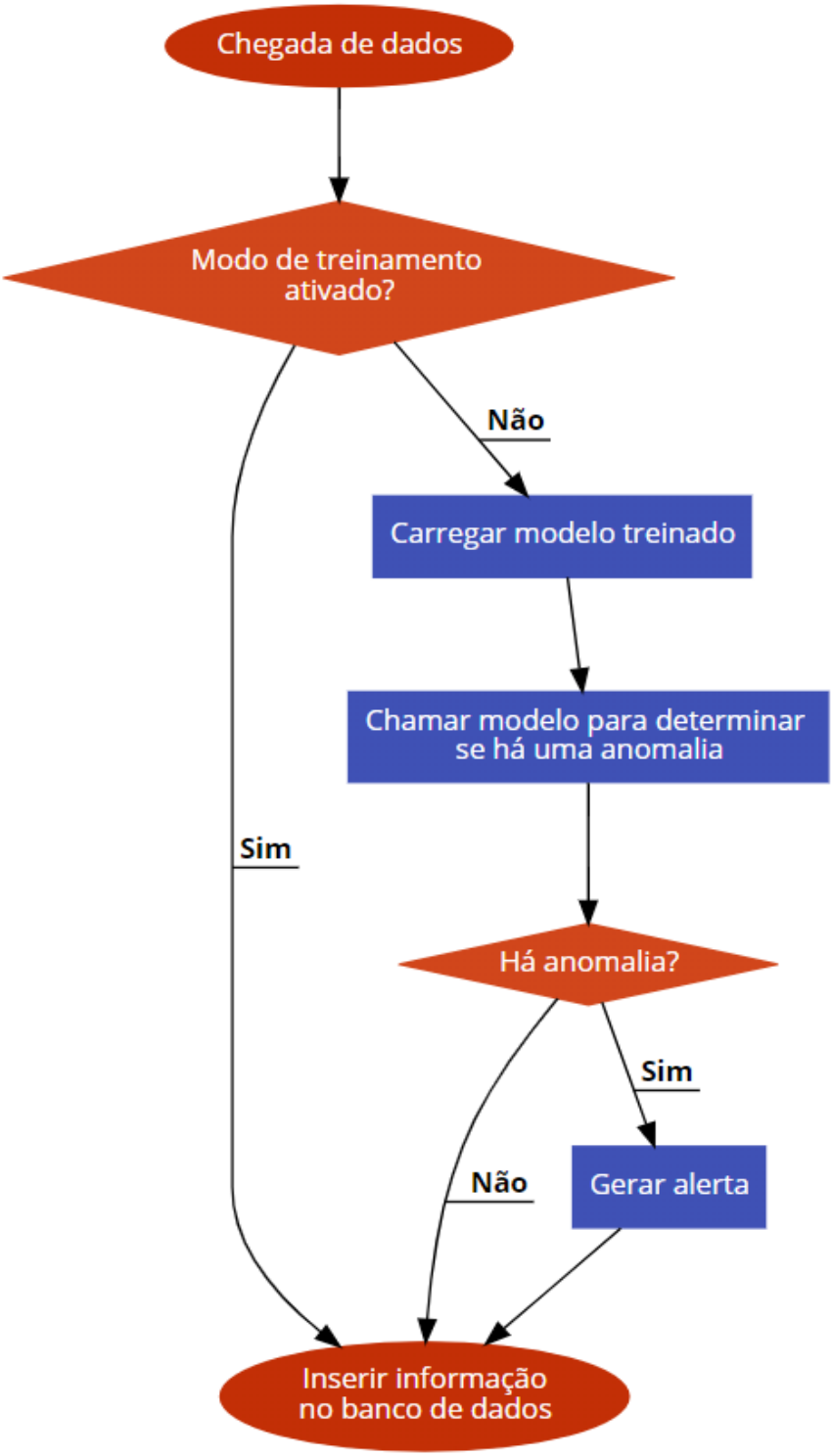
Tendo um modelo treinado com base nos dados de vibração, outro desafio que surge é o armazenamento dos modelos, que dependem de estado interno do objeto Python gerado. Não é direta a serialização desse objeto para que possa ser armazenado em disco em formato JSON, por exemplo.

Para que o modelo possa ser armazenado e carregado do banco de dados quando for necessária fazer a análise de um novo ponto de dados, portanto, é necessário um passo adicional. O passo consiste na utilização da biblioteca `dill` para a geração de um conjunto de bytes não legível a um humano que pode por sua vez ser armazenado no banco de dados. Para carregar o modelo a fim de utilizá-lo, carrega-se o conjunto de bytes do banco de dados e novamente utiliza-se a biblioteca `dill` para de-serializar o objeto modelo `PyOD` e usar suas funções de predição. (MCKERNS, STRAND, *et al.*, 2011)

3.5.5 Geração de alertas

Quando uma nova observação é feita através do endpoint RESTful de `/make_observation` e entra no sistema, o processo descrito pela Figura 8 é iniciado:

Figura 8 - Fluxograma de entrada de dados



Fonte - Autor

Independente do que aconteça, os dados são registrados no banco de dados. Adicionalmente, se o modo de treinamento para a estação em questão estiver desabilitado e se for detectada uma anomalia entre os dados vistos, então é gerado um alerta.

O alerta gerado poderá ser entregue ao administrador do sistema por diversos meios, como por e-mail ou SMS.

O retorno da requisição HTTP inclui informação sobre a geração de alerta: assim, uma máquina que além do monitoramento tenha integrado a si um sistema de desligamento automático pode ser configurada a parar de operar se uma anomalia for detectada.

3.5.6 Emulação de carga de trabalho

Para que a plataforma pudesse ser testada com os dados de treinamento, foi também desenvolvido um módulo de emulação de carga de trabalho. O módulo é desvinculado do sistema original a fim de simular um sistema separado, como seria um dispositivo real emitindo dados de um acelerômetro.

O módulo de emulação de carga de trabalho cria as estações de análise relevantes, e em seguida envia os dados referentes ao *dataset* disponibilizado pela NASA e gerado por (QIU, LEE, *et al.*, 2006) um a um, a fim de imitar um sistema real.

A plataforma oferece duas opções quanto à gravação do tempo de inserção: se o parâmetro de tempo for incluído, então esse é o adotado. Se for omitido, então a plataforma considera o tempo de inserção como sendo o instante em que a mensagem chega. Como o espaçamento entre amostras do *dataset* disponibilizado pela NASA não foi totalmente homogêneo, optou-se por manter o *timestamp* original de cada uma das medições. Assim, podem ser observados nos gráficos as datas originais dos experimentos, ocorridos em 2004.

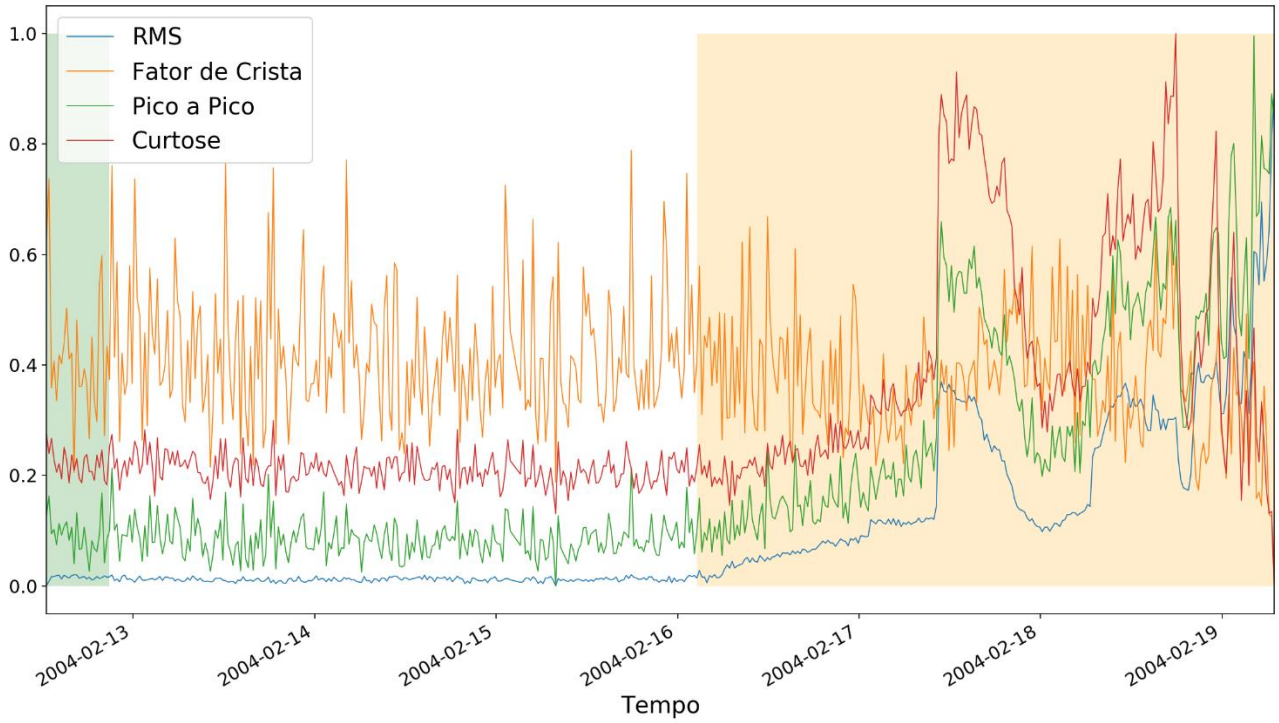
O teste final do sistema consiste na utilização do módulo de emulação de carga de trabalho para realizar a execução de todo o ciclo de monitoramento, desde a criação de estação até o reconhecimento de uma falha.

4 RESULTADOS E DISCUSSÃO

4.1 Do algoritmo

A Figura 9 mostra os resultados da aplicação do algoritmo no conjunto de dados de (QIU, LEE, *et al.*, 2006) para o canal número 1 do teste 2.

Figura 9 - Dados brutos, treinamento e predição



Fonte – Autor

No gráfico, são mostrados os valores calculados de intensidade RMS, fator de crista, pico a pico e curtose ao longo do experimento. A área colorida em verde representa o trecho usado para treinamento do modelo, e a área colorida em laranja representa os pontos para os quais um funcionamento anormal foi detectado e um alerta foi gerado. A área em branco, portanto, representa os pontos em que o funcionamento foi analisado e considerado normal.

Para mais fácil visualização das grandezas de diferentes magnitudes, os valores mostrados foram normalizados à faixa de 0 a 1, onde 1 é a intensidade mais alta atingida durante o experimento.

Para a janela de dados em questão, a falha ocorrida foi um defeito no anel exterior do rolamento. Pode-se observar um aumento gradual dos valores RMS, pico a pico e curtose no período entre os dias 16/02 e 17/02. Aproximadamente na metade do dia 17/02, há um grande pico nas grandezas medidas.

Na Figura 9, observa-se que o modelo identificou a anomalia imediatamente antes que os valores medidos começassem a subir gradualmente, e aproximadamente 36 horas antes que houvesse o grande pico em vibração da máquina.

Para os dados referentes a esse experimento (canal 1 do teste 2 do *dataset*), não foram observados falsos positivos gerados pelo modelo (disparos de alerta sem que o funcionamento fosse de fato anômalo).

O tempo de treinamento utilizado foi de 51 observações, compreendendo um intervalo de 8.3 horas. Como o padrão para o *dataset*, as observações compreendem intervalos de 1 segundo cada a 20 kHz, espaçadas 10 minutos entre si.

4.2 Do sistema RESTful

O produto do desenvolvimento foi uma plataforma RESTful que disponibiliza uma API para a inserção e consulta dos dados.

Devido às escolhas de tecnologia e ao fato de que as dependências associadas em questão de biblioteca e de serviços de banco de dados e visualização estão especificadas em arquivos de texto análogos ao de código, o sistema pode ser reproduzido sem grandes dificuldades em qualquer ambiente.

A performance do sistema foi razoável, tendo atingido tempos de execução satisfatórios para as operações realizadas. A Tabela 2 mostra os tempos medidos.

Tabela 2 - Tempo de execução de operações

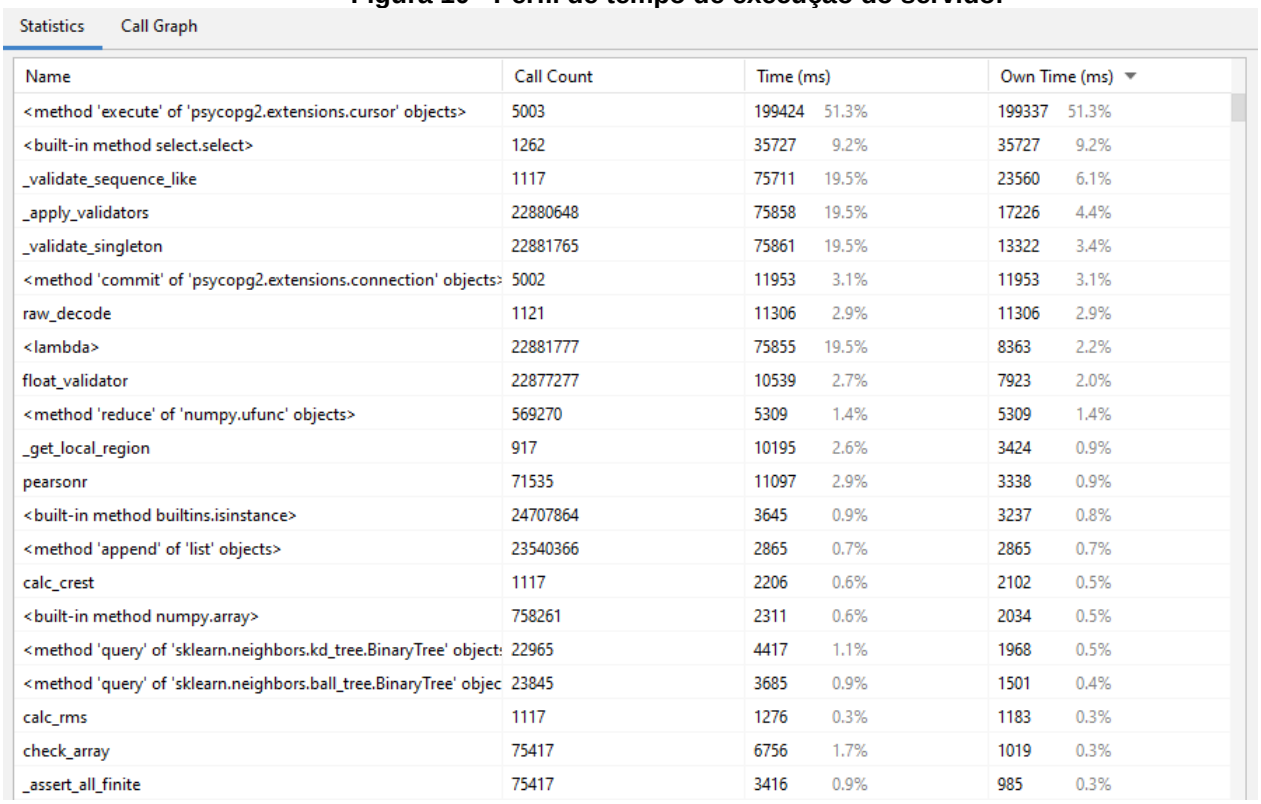
Operação realizada	Tempo por operação	Operações por segundo
Inserção de janela de alta frequência (em treinamento)	0,172 s	5,8
Inserção de janela de alta frequência (em análise)	0,328 s	3,0
Treinamento de modelo	0,179s	5,6

Os testes em questão foram realizados enviando o mais rápido possível as janelas referentes ao *dataset* disponibilizado pela NASA, referentes a janelas de 1 segundo com taxa de amostragem de 20 kHz. Cada arquivo em texto referente a uma janela tem aproximadamente 541 kB.

Podemos observar que o tempo de inserção é mais rápido durante o modo de treinamento do que durante o modo de análise. Isso é o esperado, porque durante o treinamento o único processamento feito é o cálculo de valores RMS, curtose, pico a pico e fator de crista, além da inserção dos valores no banco de dados. Fora do treinamento, quando a análise em si é feita, também é necessário carregar do banco de dados o modelo para que se tenha a predição, aumentando o tempo total necessário.

Uma análise do tempo de execução durante o teste de inserção revelou que a maior parte do tempo (51,3%) é consumida por latência de acesso ao banco de dados, mesmo localizado na mesma máquina e, portanto, mesma rede local que o processo do servidor. A Figura 10 mostra o perfil de tempo de execução de código do sistema.

Figura 10 - Perfil de tempo de execução do servidor



Statistics | Call Graph

Name	Call Count	Time (ms)	Own Time (ms)
<method 'execute' of 'psycopg2.extensions.cursor' objects>	5003	199424 51.3%	199337 51.3%
<built-in method select.select>	1262	35727 9.2%	35727 9.2%
_validate_sequence_like	1117	75711 19.5%	23560 6.1%
_apply_validators	22880648	75858 19.5%	17226 4.4%
_validate_singleton	22881765	75861 19.5%	13322 3.4%
<method 'commit' of 'psycopg2.extensions.connection' objects>	5002	11953 3.1%	11953 3.1%
raw_decode	1121	11306 2.9%	11306 2.9%
<lambda>	22881777	75855 19.5%	8363 2.2%
float_validator	22877277	10539 2.7%	7923 2.0%
<method 'reduce' of 'numpy.ufunc' objects>	569270	5309 1.4%	5309 1.4%
_get_local_region	917	10195 2.6%	3424 0.9%
pearsonr	71535	11097 2.9%	3338 0.9%
<built-in method builtins.isinstance>	24707864	3645 0.9%	3237 0.8%
<method 'append' of 'list' objects>	23540366	2865 0.7%	2865 0.7%
calc_crest	1117	2206 0.6%	2102 0.5%
<built-in method numpy.array>	758261	2311 0.6%	2034 0.5%
<method 'query' of 'sklearn.neighbors.kd_tree.BinaryTree' object	22965	4417 1.1%	1968 0.5%
<method 'query' of 'sklearn.neighbors.ball_tree.BinaryTree' objec	23845	3685 0.9%	1501 0.4%
calc_rms	1117	1276 0.3%	1183 0.3%
check_array	75417	6756 1.7%	1019 0.3%
_assert_all_finite	75417	3416 0.9%	985 0.3%

Fonte – Autor

Na figura, obtida com o auxílio da ferramenta de *profiling* de código da IDE PyCharm, pode se observar o tempo de execução das funções compõem o programa. Cada linha representa uma função chamada, e contém o número de vezes que ela foi chamada e o tempo de execução total daquela função. Em primeiro lugar com 51,3% do tempo de execução está a função de execução do cursor do banco de dados, indicando o maior gargalo.

Funções como o cálculo do valor RMS `calc_rms` e cálculo do fator de crista `calc_crest` aparecem com tempos de execução menores em comparação, com 0.3% e 0.6%, respectivamente.

Disso, toma-se que o maior benefício em questão de otimização de código ou de alocação de recursos seria sentido quando direcionado às interações com o banco de dados.

Ainda assim, dois fatos diminuem a importância do tempo de processamento necessário.

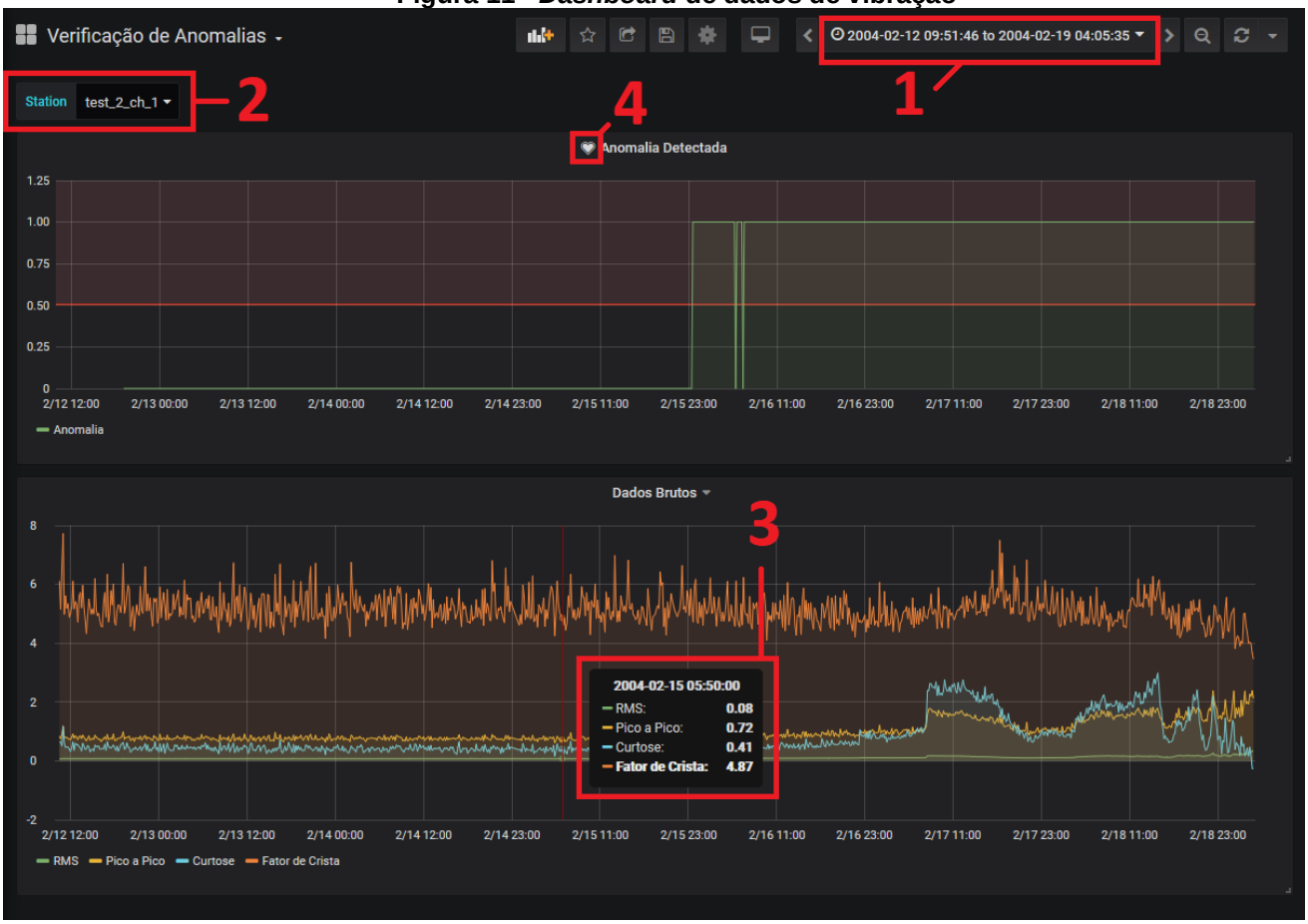
O primeiro é que o sistema atua com janelas pontuais; o teste foi realizado sob condições de inserção mais rápida possível, mas na realidade seriam efetuadas em intervalos de 10 minutos ou maiores, tornando o tempo de 0,328s menos significante.

O segundo é que por causa da arquitetura reproduzível e containerizada do sistema, seria possível ter escalabilidade horizontal; quer dizer, várias máquinas poderiam ao mesmo tempo executar o serviço, e a carga poderia ser distribuída entre elas para diminuir o tempo de resposta geral em uma situação de grande fluxo de dados.

4.3 Da visualização através de supervisório com Grafana

Desenvolveu-se o supervisório através da integração do banco de dados com o Grafana, solução aberta de visualização. O dashboard com informações captadas pelo sistema pode ser visto na Figura 11.

Figura 11 - *Dashboard* de dados de vibração



Fonte – Autor

Na figura, podem ser observados os dados brutos e o status atual, de anomalia detectada ou não, para o momento atual e alguns dias prévios.

A faixa de visualização pode ser facilmente configurada para mostrar mais detalhes ou uma visão de mais longo prazo, através do seletor de tempo indicado pelo marcador 1. Há também a possibilidade de definir um intervalo fixo, para que sejam mostrados, por exemplo, os últimos dois dias, com atualização automática da visualização.

O marcador 2 indica o seletor de estação de monitoramento. Cada estação pode representar por exemplo uma máquina, então é por meio desse controle que se define qual sensor se deseja observar.

O marcador 3 mostra o detalhe de dados brutos que se obtém ao posicionar o mouse por cima de um determinado ponto no gráfico. Na pequena janela, são mostrados os dados reais para cada grandeza mostrada, e o tempo exato daquele ponto.

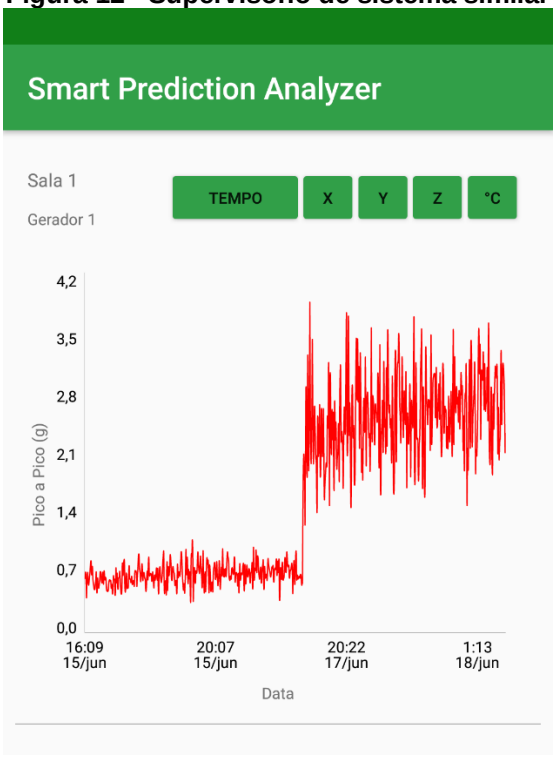
O marcador 4 mostra o indicador em forma de coração de que aquele gráfico está conectado ao sistema de geração de alertas. Ao ultrapassar a linha vermelha por um período configurável, é enviado um alerta pelos meios configurados.

4.4 Comparação com sistema similar

Um sistema comparável foi recentemente desenvolvido na literatura e serviu como *benchmark* para o presente trabalho, apesar de com focos diferentes.

Sistema De Manutenção Preditiva Utilizando Redes BLE E Wi-Fi com Aprendizado De Máquina (SILVEIRA, 2019) foi um trabalho desenvolvido por um aluno também do Instituto Federal de Santa Catarina como trabalho de conclusão de curso do curso de Engenharia de Telecomunicações.

Figura 12 - Supervisório de sistema similar



Fonte – SILVEIRA, 2019

Tendo tido escopo mais amplo, que compreende também a conectividade de sensores, o desenvolvimento de um aplicativo supervisório e a execução de testes em bancada, o trabalho foi menos a fundo no desenvolvimento de um algoritmo de aprendizado de máquina e plataforma escalável, que foram o foco do presente trabalho.

Não foi possível fazer uma comparação direta de resultados devido à falta de um conjunto de dados de alta frequência disponível (apesar de que valores RMS e pico a pico estão disponíveis minuto a minuto no repositório online *Github* do autor).

Ainda assim, é possível fazer uma comparação qualitativa da metodologia de aplicação de aprendizado de máquina aos dados.

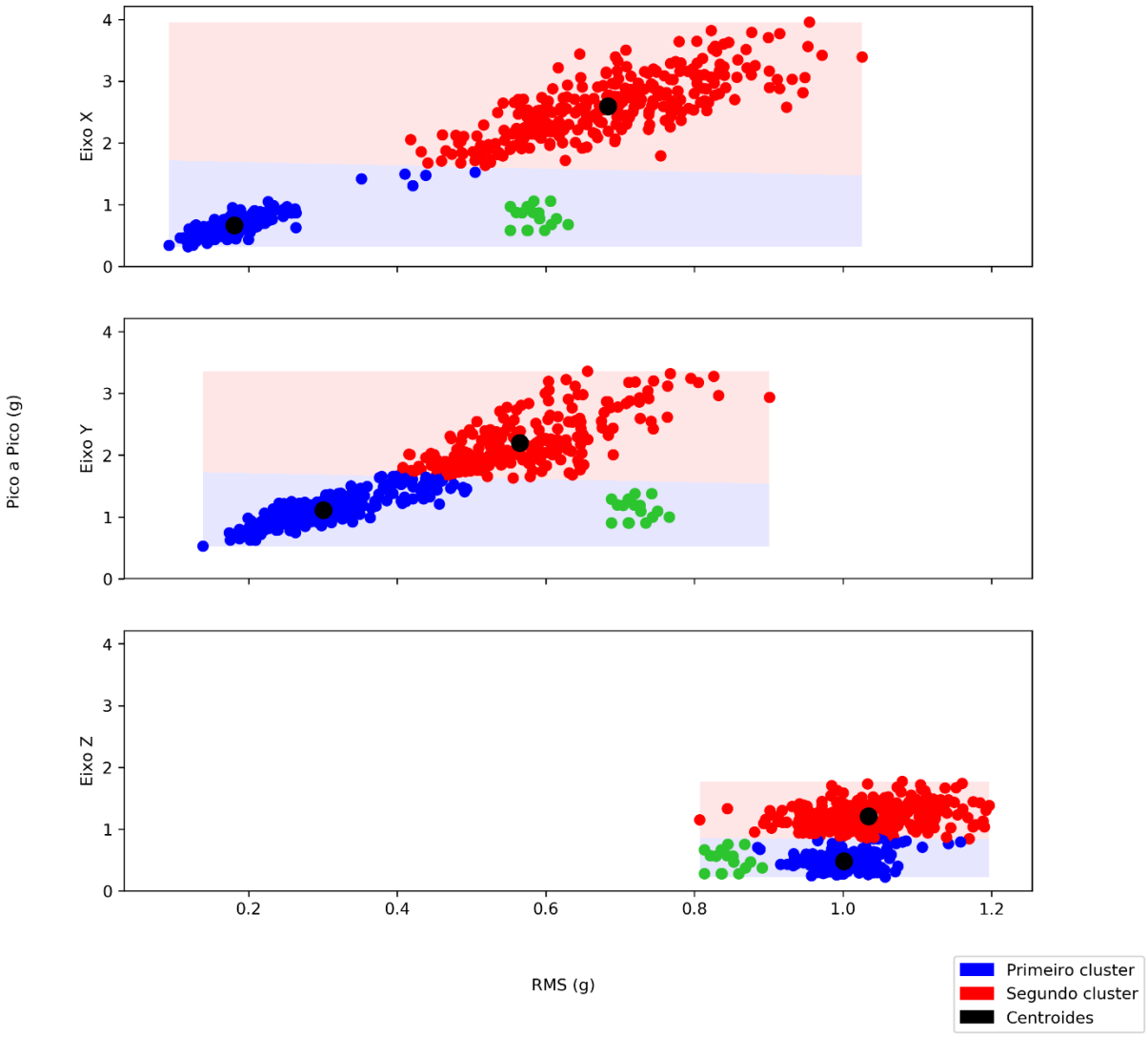
No trabalho supracitado, a *clusterização* dos dados, e portanto classificação entre funcionamento normal ou anômalo, é feita com base em um conjunto de dados conhecidamente falho, e referente ao mesmo modo de falha que está sendo testado (há um conjunto de treinamento para a falha de desbalanceamento induzido no motor, e a falha de teste é o mesmo desbalanceamento). O algoritmo pode ser considerado, portanto, semi-supervisionado, no sentido de que se sabe o que se pode esperar como falha.

Essa abordagem tem dois principais problemas.

Um deles é que os dados de vibração para uma máquina que já apresente defeitos são difíceis de se conseguir num contexto industrial, porque não é interessante que se deixe uma máquina sabidamente com problemas em funcionamento para coleção de informação.

O outro é que o sistema não está pronto para lidar com modos de falha novos, resultante de um defeito diferente e que não tenham sido vistos antes. Um exemplo pode ser encontrado na Figura 13.

Figura 13 - Visualização de falha de classificação



Fonte – SILVEIRA, 2019 (modificado)

Na Figura 13, pode ser observado o processo de decisão do algoritmo de clusterização semi-supervisionado. Em azul são mostrados os pontos considerados normais, e em vermelho os anômalos. As áreas azul e vermelha representam a decisão feita para um dado ponto que se encontre nela.

Em verde, em modificação feita pelo autor do presente trabalho, observam-se pontos hipotéticos que são visivelmente diferenciados do agrupamento de pontos normais, mas que seriam erroneamente classificados como normais.

O algoritmo desenvolvido no presente trabalho, por ser não-supervisionado, não é afetado pelo viés de esperar que a falha se encaixe em algum padrão previamente conhecido. Desvios do funcionamento normal são todos detectados e considerados problemas.

De modo geral, pode se fazer uma comparação qualitativa entre os sistemas desenvolvidos. Tal comparação pode ser vista na Tabela 3.

Tabela 3 - Comparação qualitativa entre sistemas

Critério	SILVEIRA, 2019	Presente trabalho
Sistema supervisório	✓	✓
Alerta de funcionamento anômalo	✓	✓
Necessidade de sensor de alta frequência	×	✓
Detecção de falhas inéditas	×	✓

5 CONCLUSÃO

Ao longo do trabalho, foram apresentados e desenvolvidos os passos necessários à criação de um sistema de predição de falhas com tecnologias *open-source*. Desde o algoritmo até a especificação da infraestrutura utilizada, foram encontradas soluções abertas e que proporcionaram um resultado que aos olhos do autor foi satisfatório.

Os resultados obtidos pelo algoritmo foram bons no que toca a terem conseguido prever falhas em estágio em que não são óbvias a um ser humano, e a plataforma foi desenvolvida de modo tal que o sistema poderia ser adaptado sem dificuldades a um número grande de sensores/máquinas. Pela natureza aberta das bibliotecas e ferramentas utilizadas, não há impedimento monetário a qualquer pessoa que se inspire a criar um sistema similar.

Trabalhos futuros que venham abordar um tema similar podem desenvolver, como sugestão, personalizações dos algoritmos utilizados a nível mais baixo; talvez com bibliotecas como tensorflow ou pytorch, que apresentam ferramentas de aprendizado de máquina por redes neurais, diferente dos métodos estatísticos utilizados pelo presente trabalho. Ao custo da simplicidade do modelo, podem ser captadas maiores nuances no processo de detecção de anomalias.

Seria interessante também que aplicassem o modelo a um caso de uso real, validando assim a utilidade do conceito de predição de falhas com aprendizado de máquina na indústria. O conjunto de dados utilizado para teste do algoritmo, apesar de são em sua metodologia, não reflete exatamente as condições de trabalho de uma máquina do mundo real, que pode ter padrões diferentes de utilização ou sinais mais sutis de falha.

Quanto à plataforma, para que se aproximasse melhor de um produto pronto para uso, seria importante criar uma interface de usuário que conectasse a API do serviço *back*-end e o supervisório Grafana integrado de modo mais intuitivo ao usuário.

6 BIBLIOGRAFIA

AHMAD, R.; KAMARUDDIN, S. An overview of time-based and condition-based maintenancein industrial application. **Computers & Industrial Engineering**, Nibong Tebal, n. 63, p. 135-149, 2012.

BREUNIG, M. M. et al. **LOF:** Identifying Density-Based Local Outliers. ACM SIGMOD International Conference on Management of Data. Dallas: ACM New York. 2000. p. 93-104.

CUBILLO, A.; PERINPANAYAGAM, S.; ESPERON-MIGUEZ, M. A review of physics-based models in prognostics: Application to gears and bearings of rotating machinery. **Advances in Mechanical Engineering**, v. 8, n. 8, p. 1-21, 2016.

DOCKER, S. What is a Container? **Official Docker Website**. Disponivel em: <https://www.docker.com/resources/what-container>. Acesso em: 18 Novembro 2019.

GUPTA, M. et al. Outlier Detection for Temporal Data: A Survey. **IEEE Transactions on knowledge and data engineering**, Hyderabad, v. 26, n. 9, p. 2250-2267, 2014.

HENG, A. et al. Rotating machinery prognostics: State of the art, challenges and opportunities. **Mechanical Systems and Signal Processing**, Brisbane, n. 23, p. 724-739, 2009.

HERMAN, M. Django vs. Flask in 2019: Which Framework to Choose. **testdriven.io**, 2019. Disponivel em: <https://testdriven.io/blog/django-vs-flask/>. Acesso em: 14 Novembro 2019.

HODGE, V. J.; AUSTIN, J. A survey of outlier detection methodologies. **Artificial Intelligence Review**, York, v. 2, n. 22, p. 85-126, 2004.

JIA, X. et al. **Review of PHM Data Competitions from 2008 to 2017:** Methodologies and Analytics. Annual Conference of the Prognostics and Health Management Society. Cincinnati: [s.n.]. 2018. p. 1-10.

KRAWCZYK, B. et al. Ensemble learning for data stream analysis: A survey. **Information Fusion**, Richmond, n. 37, p. 132-156, 2017.

LABIB, A. W. A decision analysis model for maintenance policy selection using a CMMS. **Journal of Quality in Maintenance Engineering**, Manchester, v. 10, n. 3, p. 191-202, 2004.

LEBOLD, M. et al. **Review of Vibration Analysis Methods for Gearbox Diagnostics and Prognostics**. 54th Meeting of the Society for Machinery Failure Prevention Technology. Virginia Beach: [s.n.]. 2000. p. 623-634.

MARCORIN, W. R.; LIMA, C. R. C. Análise dos Custos de Manutenção e de Não-manutenção de Equipamentos Produtivos. **Revista de Ciência & Tecnologica**, Santa Bárbara d'Oeste, v. 11, n. 22, p. 35-42, 2003.

MCKERNS, M. M. et al. **Building a framework for predictive science**. Python in Science Conference. [S.l.]: Python in Science. 2011.

OTANI, M.; MACHADO, W. V. A proposta de desenvolmento de gestão da manutençãoindustrial na busca da excelência ou classe mundial. **Revista Gestão Industrial**, Ponta Grossa , v. 4, n. 2, p. 1-16, 2008.

PANDIYAN, J. et al. **Design of Industrial Vibration Transmitter Using MEMS Accelerometer**. International MEMS Conference. Gujarat: Institute of Physics Publishing. 2006. p. 442–447.

PAUDYAL, S.; ATIQUE, M. S. A.; YANG, C. X. **Local Maximum Acceleration Based Rotating Machinery Fault Classification Using KNN**. IEEE EIT 2019. Grand Forks: [s.n.]. 2019.

QIU, H. et al. Wavelet filter-based weak signature detection method and its application on rolling element bearing prognostics. **Journal of Sound and Vibration**, Cincinatti, n. 289, p. 1066-1090, 2006. Disponivel em: <https://ti.arc.nasa.gov/tech/dash/groups/pcoe/prognostic-data-repository/>. Acesso em: 2019.

RAMASWAMY, S.; RASTOGI, R.; SHIM, K. **Efficient algorithms for mining outliers from large data sets**. ACM SIGMOD international conference on Management of data. Dallas: ACM New York. 2000. p. 427-438.

SHYU, M.-L. et al. **A Novel Anomaly Detection Scheme Based on Principal Component Classifier**. IEEE Foundations and New Directions of Data Mining Workshop. [S.l.]: [s.n.]. 2003. p. 172-179.

SILVEIRA, V. M. D. Sistema De Manutenção Preditiva Utilizando Redes BLE E Wi-Fi com Aprendizado De Máquina, São José, 2019.

STRINGER, D. B.; SHETH, P. N.; ALLAIRE, P. E. Physics-based modeling strategies for diagnostic and prognostic application in aerospace systems. **Journal of Intelligent Manufacturing**, Cleveland, n. 23, p. 155-162, 2012.

ZHAO, Y. et al. **LSCP:** Locally Selective Combination in Parallel Outlier Ensembles. SIAM International Conference on Data Mining. Calgary: Society for Industrial and Applied Mathematics. 2019.

ZHAO, Y.; NASRULLAH, Z.; LI, Z. PyOD: A Python Toolbox for Scalable Outlier Detection. **Journal of Machine Learning Research**, v. 20, n. 96, p. 1-7, 2019.