UNIVERSIDADE DO VALE DO RIO DOS SINOS - UNISINOS
UNIDADE ACADÊMICA DE GRADUAÇÃO
CURSO DE BACHARELADO EM ENGENHARIA ELÉTRICA

RENATA SANTOS GRAFF

SISTEMA INDUSTRIAL DE INTERNET DAS COISAS PARA PREDIÇÃO DE ANOMALIAS APLICANDO TÉCNICAS DE APRENDIZADO DE MÁQUINA

São Leopoldo
2019

RENATA SANTOS GRAFF

SISTEMA INDUSTRIAL DE INTERNET DAS COISAS PARA PREDIÇÃO DE ANOMALIAS APLICANDO TÉCNICAS DE APRENDIZADO DE MÁQUINA

Trabalho de Conclusão de Curso apresentado como requisito parcial para obtenção do título de Bacharel em Engenharia Elétrica, pelo Curso de Engenharia Elétrica da Universidade do Vale do Rio dos Sinos – UNISINOS

Orientador: Prof. Dr. Rodrigo Marques de Figueiredo

São Leopoldo

2019

RESUMO

A evolução decorrente da Indústria 4.0 trouxe conceitos tecnológicos que cada vez mais vêm sendo aplicados as empresas de manufatura. Dentre estas tecnologias estão o aprimoramento de sensores, de sistemas embarcados e de computação de borda, além da conexão cada vez mais evidente entre as áreas de Tecnologia da Informação e Operacional. Estas áreas acabam fazendo uso de sistemas de Inteligência Artificial para tornar as práticas de manutenção mais assertivas com predições de falhas, ampliando a avaliação e o diagnóstico de danos em máquinas. Estas práticas são denominadas de sistemas de manutenção inteligente, que provem da capacidade de monitorar as condições físicas, auxiliar na tomada de decisões para efetuar ações de manutenção além de fornecer os diagnósticos das anomalias do equipamento, porém muitos produtos disponíveis no mercado possuem um preço alto, o que impossibilita que empresas de pequeno e médio porte façam uso desse tipo de sistema. Com isto, este trabalho aborda o desenvolvimento de um sistema de baixo custo para a manutenção preditiva de máquinas utilizando *hardware* de computação de borda e plataforma em nuvem para o desenvolvimento do Aprendizado de Máquina, além de prover o resultado da inteligência em uma interface para com o usuário possibilitando a compreensão dos dados para à tomada das devidas ações ao sistema. Espera-se com este estudo auxiliar de forma inteligente e com o mínimo de investimento possível, a manutenção preditiva de sistemas ou equipamentos na indústria, abrangendo o emprego dessas tecnologias de Internet das Coisas Industriais nas empresas de pequeno e médio porte, permitindo o aumento da vida útil e a confiabilidade de seus equipamentos. O desenvolvimento feito neste trabalho capacita, dotando de inteligência o sistema de classificar os dados lidos entre quatro *clusters,* denominando o status do mesmo entre desligado, operação normal, com atenção e anormal, possibilitando a correção futura de algum problema que possa ocorrer.

Palavras-chave: Inteligência Artificial. Aprendizado de Máquina. Internet das Coisas Industriais. Predição de Falhas. Manutenção Preditiva.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE SIGLAS

ADC	*Analog Digital Converter*
API	*Application Programming Interface*
ARIMA	*Autoregressive Integrated Moving Average*
ASIC	*Application Specific Integrated Circuits*
ASSP	*Application Specific Standard Parts*
BI	*Business Intelligence*
CART	*Classification And Regression Trees*
CLP	Computador Lógico Programável
CNC	Comando Numérico Computadorizado
CSV	*Comma-Separated-Values*
DAC	*Digital Analog Converter*
DSP	*Digital Signal Processors*
ELM	*Extreme Learning Machines*
HTTP	*Hypertext Transfer Protocol*
IA	Inteligência Artificial
IIoT	*Industrial Internet of Things*
IoT	*Internet of Things*
M2M	*Machine-to-machine*
MGE	Medidor de Grandezas Elétricas
MQTT	*Message Queuing Telemetry Transport*
PaaS	*Platform as a Service*
PCA	*Principal Component Analysis*
RFID	*Radio Frequency Identification*
SVM	*Support Vector Machine*
TCP	*Transmission Control Protocol*
TI	Tecnologia de Informação
TO	Tecnologia Operacional

SUMÁRIO

1 INTRODUÇÃO

Nos últimos anos, as atividades para a manutenção de máquinas começaram a ter uma importância de âmbito crítico nas empresas de manufatura, especialmente devido ao crescimento da complexidade dos equipamentos para atender todas as demandas de produção. Devido a isto, muitas empresas focam na disciplina de gerenciamento de manutenção como forma de elencar seu valor comercial. No mercado atual, a situação das empresas é procurar cada vez mais por alternativas capazes de tornar seus processos optimizados e compatíveis com as tecnologias disponíveis, garantindo a competitividade no mercado para com outras empresas e atingindo demandas maiores, entregando seus produtos de forma mais ágil e com custo de produção reduzido.

Considerando os avanços e a convergência entre as áreas de Tecnologia de Informação (TI) e Tecnologia Operacional (TO) englobadas na Indústria 4.0, as tão tradicionais atividades de manutenção estão se adaptando através da mudança de paradigma, mantendo suas raízes em conceitos já admitidos de manutenção e evoluindo para uma manutenção inteligente. A manutenção inteligente é proveniente do progresso da computação cognitiva, onde as estratégias de manutenção preditiva podem ser aplicadas através de sistemas que empregam Inteligência Artificial (IA), os quais fornecem recursos de monitoramento e prognósticos em tempo real. Como consequência, as indústrias desejam melhorar a disponibilidade de seus equipamentos enquanto diminuem o tempo de inatividade e aumentam a produtividade por meio desta nova era, assim, um investimento em manutenção de máquinas passa a ser muito bem visto para empresas de grande porte, onde os custos com manutenção permutam de 15% a 40% dos custos totais da produção. (CYRINO, 2018).

Porém, este é um cenário não atingido por pequenas empresas. Segundo a FINEP (2019), atualmente apenas 1,6% da indústria brasileira tem tecnologia de Indústria 4.0 e este valor existe devido às grandes empresas. Inserido neste contexto, o presente trabalho tem como objetivo principal desenvolver uma solução de baixo custo para a manutenção preditiva de máquinas e equipamentos da indústria, com o intuito de aumentar a assertividade na previsão de falhas do sistema em que estiver inserido.

Sabendo que um dos fatores de riscos para qualquer empresa é a quebra de maquinário, este estudo traz uma abordagem capaz de eliminar este tipo de risco. Aplicando conceitos de sistemas embarcados conectados à sensores de máquina, computação de borda para tratamento e envio dos dados para algoritmos em nuvem de Aprendizado de Máquina fornecidos por plataformas de código aberto, o sistema proposto é capaz de aprender quais os padrões

operacionais esperados e, a partir disto, aumentar a assertividade da manutenção do equipamento no momento em que percebe uma atividade anormal.

Para alcançar este objetivo, fez-se necessário definir o sistema no qual este estudo realiza suas análises e experimentos; definir o *hardware* que faz a coleta, o tratamento e a transmissão dos dados à plataforma de computação em nuvem; implementar, testar e validar o modelo de Aprendizado de Máquina; e por fim, realizar a comunicação entre a plataforma e a interface com o usuário final.

Este trabalho está organizado em seis capítulos. A fundamentação teórica é abordada no Capítulo 2, onde o leitor encontrará os principais conceitos para sistemas embarcados, suas diferentes formas de processamento, comunicação e quais suas características quando aplicado à indústria, além dos principais conceitos de Aprendizado de Máquina e de computação de borda. O Capítulo 3 apresenta os trabalhos relacionados a este, contribuindo para a pesquisa do estado da arte. O Capítulo 4 traz a metodologia proposta para este estudo de caso, exemplificando o sistema proposto, os materiais, e quais serão as abordagens de análise e testes dos dados e do modelo. O Capítulo 5 apresenta a análise dos resultados obtidos ao emular a metodologia proposta em um laboratório da universidade, como foi escolhido o conjunto de dados utilizado, além de como foi feito o treinamento e teste do Aprendizado de Máquina, assim como o resultado final com a interface para o usuário. Por fim, o Capítulo 6 aborda as conclusões obtidas com a realização deste trabalho e as melhorias e próximos estudos que partem do mesmo.

2 FUNDAMENTAÇÃO TEÓRICA

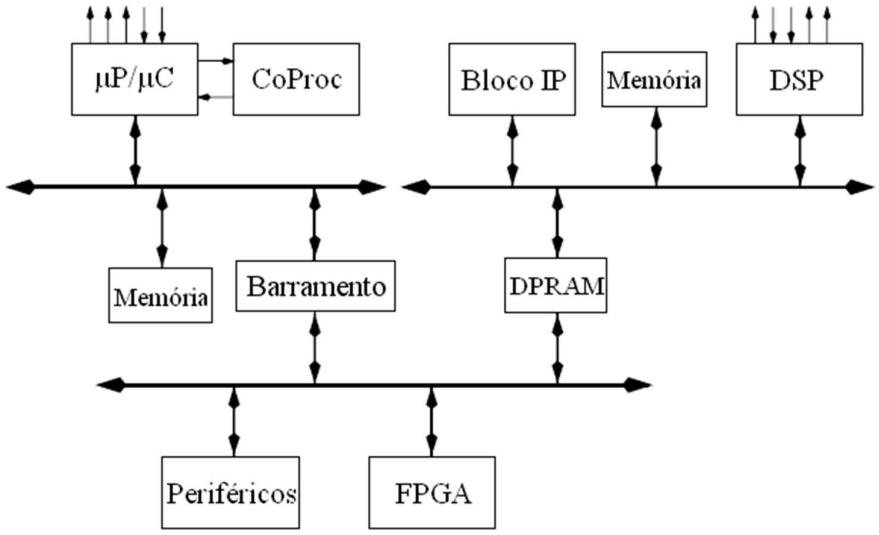
Este capítulo aborda os conceitos básicos das tecnologias e técnicas necessárias para a compreensão do objetivo proposto por este trabalho. A seguir serão apresentadas as definições para sistemas embarcados e como estes são aplicados à indústria, as técnicas de Aprendizado de Máquina e Inteligência Artificial, além de ser abordado o conceito de computação de borda e como este influencia nas trocas de informações das máquinas inseridas no contexto de IIoT (*Industrial Internet of Things* – do inglês Internet das Coisas Industriais).

2.1 SISTEMAS EMBARCADOS

Sistemas embarcados são compostos por *hardware*, *software* e em alguns casos, de partes mecânicas. Estes dispositivos realizam funções e compartilham propriedades como recursos limitados, dentre eles memória, peso, espaço e energia, além de requisições em tempo real, múltiplas tarefas, alta complexidade e compartilhamento de recursos. Ou seja, sistemas embarcados consistem em diversos componentes desenvolvidos para interagir entre si e com o ambiente em que estão inseridos. (BORGES; RODRIGUES, 2011). Realizando um contraste com sistemas funcionais, que são sistemas específicos para funções que já possuem valores para entradas e saídas, um sistema embarcado é definido a partir de suas propriedades desejadas, sendo estas definidas através dos comportamentos que o sistema deve possuir. (KAPITONOVA *et al.*, 2005).

Lavagno e Passerone (2005) explicam que estes sistemas são informalmente definidos como uma coleção de partes programáveis cercadas por ASIC (*Application Specific Integrated Circuits* - do inglês Circuitos Integrados de Aplicações Específicas) e por outros ASSP (*Application Specific Standard Parts* - do inglês Circuitos Integrados de Aplicações Padrões) que interagem constantemente com o ambiente através de sensores e atuadores. Esta coleção de partes pode ser fisicamente um conjunto de *chips* dispostos em uma placa ou um conjunto de módulos de circuitos integrados. O *software* é utilizado para características específicas e para manter a flexibilidade, enquanto o *hardware* é desenvolvido para aumentar o desempenho e reduzir o consumo de energia. Um exemplo de uma arquitetura para um sistema embarcado pode ser visto na Figura 1.

Figura 1 – Arquitetura de Sistema Embarcado



Fonte: Lavagno e Passerone (2005).

Na arquitetura, os principais componentes programáveis são os microprocessadores e DSPs (*Digital Signal Processors* - do inglês Processadores de Sinal Digital), os quais implementam a parte do *software* do sistema embarcado. Os componentes reconfiguráveis exibem as características intermediárias entre *hardware* e processadores dedicados, como área, custo, desempenho e energia. Já os componentes de *hardware* customizáveis, por sua vez, implementam blocos específicos da aplicação e dos periféricos. Todos os componentes são conectados através de barramentos padrões e redes dedicadas, e os dados são armazenados em memórias. Geralmente, vários subsistemas menores são ligados em rede para o controle que se deseja obter. (LAVAGNO; PASSERONE, 2005).

Haugen *et al.* (2005) listam as principais características dos sistemas embarcados. Dentre elas, na maioria das vezes, o *software* é executado em uma plataforma de *hardware* dedicada a uma tarefa específica, como já explicado anteriormente. Naturalmente, o *hardware* impõe limites ao que o *software* poderá executar. Além disto, a maioria dos sistemas embarcados pode ser considerada reativa, ou seja, o sistema recebe uma mensagem e é suposto deste retornar uma resposta, o que pode envolver alguma alteração de estado no controle do *hardware*. Tal *software* é elaborado de maneira a ter resposta em tempo real, mas os requisitos de desempenho são mais frequentemente estatísticos do que absolutos. Estes sistemas geralmente são necessários para lidar com muitos pedidos independentes ao mesmo tempo.

Lavagno e Passerone (2005) ainda concluem que os sistemas embarcados aumentam exponencialmente o desempenho e a funcionalidade através de um custo cada vez menor, sendo

isto possível através das tecnologias dos circuitos integrados e da manufatura, admitindo uma produção em escala e o desenvolvimento de dispositivos mais complexos, além de novas metodologias de *design*, o que permite a eficiência e a inteligência destes sistemas.

2.1.1 Processamento

Todo sistema embarcado faz uso de dispositivos processadores em seu *hardware*, obtendo a função através da programação do *software*. Pode-se dizer que são dispositivos utilizados para alterar dados e/ou tomar decisões sobre as ações que devem ser executadas. Eles podem ser diferenciados entre processadores de *software*, os quais são programados via *software* e entregam flexibilidade para mudanças como em funcionalidade e comportamento; e os processadores de *hardware*, desenvolvidos especificamente para executar uma determinada função, entregando velocidade de processamento, mas sem a elevada possibilidade de alterações, necessitando serem substituídos quando defasados. (BARROS, 2015).

No Quadro 1, serão apresentados alguns tipos de processadores com suas principais características elencadas, dos quais podem ou não estarem empregados em sistemas embarcados.

Quadro 1 – Tipos de Processadores

Tipo de processador	Característica
Processador de propósito geral	Utilizados em computadores, não são geralmente empregados na implementação de sistemas embarcados. Mesmo apresentando vantagens em desempenho por tornar o sistema flexível, acabam aumentando o custo do projeto dos sistemas embarcados.
Processadores embarcados	Baseados em processadores de propósito geral, são capazes de incorporar diversos dispositivos que facilitam e reduzem o custo do projeto de sistemas embarcados.
Microcontroladores	Processadores de *software* baseados em processadores embarcados, desenvolvidos para incorporar diversas funções em um único *chip*. Apresentam poder de processamento menor, além de possuírem um conjunto de instruções adaptados para a sua utilização, como manipulação de bit ou acesso a pinos específicos.

	Podem agrupar diversos dispositivos em um, como conversores ADC ou DAC, temporizadores, contadores, interfaces seriais, geradores de *clock*, entre outros.
ASIC	Desenvolvidos especialmente para uma determinada função, seja de controle, processamento de sinais, comunicação etc. Apresentam melhor performance e são mais flexíveis, visto que suas alterações são realizadas via *software*.

Fonte: Barros (2005).

Para Pereira e Carro (2006), na manufatura e nos processos industrias é necessário ter a disposição altos índices de configurabilidade física, visando acomodar mudanças frequentes devido à variedade e volume de produtos produzidos, assim como a introdução de novas tecnologias de fabricação. Em razão disso, é exigido uma alta taxa de resposta para se recuperar de falhas na máquina e no processo, obtendo perdas mínimas na produção. Por isso é importante evidenciar que o processador de um sistema embarcado deve entregar este tipo de resposta no menor tempo de processamento possível, para que o sistema não se torne um gargalo por falta de agilidade. Porém, essa alta taxa de resposta não deve ser provada apenas pela definição do processador que o sistema embarcado possuir, ante o exposto, aborda-se a seguir as principais formas de comunicação deste dispositivo com os demais, com o objetivo de garantir a solução mais apropriada para o perfeito funcionamento do sistema.

2.1.2 Comunicação

Mahalik (2003) e Neumann (2007) abordam que, para que os sistemas embarcados possam interagir e sincronizar com tempos mínimos entre si, os mesmos necessitam de protocolos de comunicação industriais em tempo real, a fim de que ocorra a comunicação entre estes sistemas que, por muitas vezes, estão distribuídos pela fábrica, tendo seus componentes afastados fisicamente. Para isto, os mesmos necessitam da existência destes protocolos de comunicação. Rodrigues *et al.* (2013) listam algumas tecnologias para que esta comunicação seja efetuada:

- RFID: Manipulação de dados através da frequência de rádio;
- *WiFi*: Troca de dados sem fio, através de ondas de rádio, em uma rede de computador ou internet;
- *Bluetooth*: Troca de dados sem fio entre dispositivos próximos um do outro, realizada através de radiofrequência;

- RS232/485: Envio de dados sequenciais em um canal de comunicação ou barramento, sendo enviado um bit de cada vez;
- I2C: Envio e recebimento de dados simultâneos através da hierarquia mestre/escravo, possuindo baixa velocidade.

A comunicação com o ambiente em que os dispositivos estão inseridos é essencial para os sistemas embarcados, principalmente aqueles que tem sua empregabilidade na indústria, assunto que será abordado na próxima seção.

2.1.3 Aplicações na indústria

Sistemas embarcados industriais são utilizados para supervisionar operações específicas em aplicações mecânicas e de sistemas elétricos. Agem como um sistema operacional programável que executa tarefas especificas como acionamento de motores, controle de velocidades, ajuste de temperaturas, entre outros. Estes sistemas são desenvolvidos para auxiliar no controle de eficiência energética, aumento de desempenho e controle de processos dos ambientes em que estão inseridos. (MONKEY, 2018).

Na automação industrial, estes sistemas são utilizados em dois casos: para o controle de máquinas e para o monitoramento de máquinas. No controle de máquinas, o sistema serve para executar tarefas dentro dos equipamentos, como manter níveis de fluído em uma máquina CNC (Comando Numérico Computadorizado) e controlar a velocidade de linhas de montagem, sendo possível integrar o sistema embarcado com os controles já existentes da própria máquina, reaproveitando suas funcionalidades. Para o monitoramento de máquinas, os sistemas embarcados possibilitam a revisão constante de condições da máquina em tempo real, como temperatura, potência, pressão, vibração etc. Estes dados podem ser enviados aos sistemas de manufatura existentes e reutilizados para gerar análises de desempenho, fornecendo informações para relatórios, notificações ou painéis informativos, além de auxiliarem para o aumento da produtividade, otimização de recursos e exclusão da necessidade de haver diversos tipos de aplicações de sistemas incorporados. (MONKEY, 2018). Os mesmos dados ainda podem alimentar algoritmos de Aprendizado de Máquina, possibilitando a previsão e análise dos mesmos de forma inteligente através destas tecnologias atuais.

2.2 APRENDIZADO DE MÁQUINA

Aprendizado de Máquina é uma área da Inteligência Artificial que tem por objetivo desenvolver algoritmos e técnicas computacionais que permitem que o computador seja capaz de aprender, em outras palavras, faz com que o computador consiga adquirir conhecimento de forma automática a partir de exemplos históricos, e assim, aperfeiçoa seu desempenho em determinada tarefa. (GOLDSCHMIDT, 2010).

Segundo Coppin (2010), o aprendizado está diretamente ligado com a inteligência, uma vez que o processo de aprender torna capaz o conhecimento de novas formas no tratamento dos dados, como desenvolvimento motor e a habilidade cognitiva através de instruções ou prática, a organização deste novo conhecimento e as descobertas e teorias tidas em cima dos novos fatos através da observação e experimentação realizada. Desde que iniciou-se a construção de computadores são realizadas pesquisas para implantar algumas dessas capacidades às máquinas, sendo o maior desafio da Inteligência Artificial a resolução de um destes problemas.

Mitchell (1997), exemplifica o Aprendizado de Máquina como um aprendizado baseado na experiência da máquina, através da tarefa que a mesma executa e dos problemas que eventualmente podem acontecer. Logo, este aprendizado fica responsável por encontrar uma maneira de resolver o problema. Esta área da Inteligência Artificial tem como objetivo desenvolver técnicas computacionais que permitam a predição e o aprendizado de determinados comportamentos e padrões a partir de experiências acumuladas em decorrência de problemas anteriormente encontrados.

Existem diversos métodos de Aprendizado de Máquina, entre eles, o aprendizado por hábito. Este método tem como característica do programa aprender por experiência de acordo com o que foi informado anteriormente, porém, este programa armazena apenas os dados que podem vir a ser classificados, caso haja algum outro valor que ele não consiga classificar, o método falhará. Há também o método de aprendizado por conceito, o qual analisa todas as hipóteses e retorna qual está correta. Neste método existe uma subdivisão, sendo esta uma hipótese mais geral, o que significa que não existe especificamente uma única possibilidade correta, e sim que o programa encontrará aquela que mais se aproxima do que é considerado correto, originando o problema de que nem sempre o usuário deseja receber a hipótese correta, e sim a mais comum. (COPPIN, 2010).

Segundo Goldschmidt (2010), muitos dos exemplos utilizados pelos algoritmos de aprendizado são representados por meio de um conjunto de característica, ou seja, por atributos que procuram descrever o problema em questão e que podem variar dependendo do contexto

da aplicação. Para cada tipo de situação ou problema a ser solucionado, os atributos contêm valores que descrevem tal situação. Por isso, a escolha destes atributos para descreverem os exemplos de uma aplicação possuem grande influência na qualidade do conhecimento adquirido pelos algoritmos de aprendizado.

Goldschmidt (2010) também traz que a forma com que o espaço de busca por um modelo de conhecimento deve ser percorrido é denominado um paradigma de aprendizado. Os principais paradigmas de aprendizado em que os algoritmos se baseiam são apresentados a seguir.

- Simbólico: Compreende a construção da representação de conceitos a partir da análise de exemplos. O modelo de conhecimento a ser construído pode ser representado por meio de expressões lógicas, árvores, regras, redes semânticas, dentre outras.
- Estatístico: Métodos, em geral paramétricos, que são utilizados para encontrar boas aproximações do modelo de conhecimento que esteja sendo induzido.
- Baseado em exemplos: Envolve a buscas de casos existentes similares ao novo exemplo a ser analisado para deduzir a saída do sistema, sendo ideal a utilização de casos anteriores representativos.
- Conexionista: Utiliza modelos matemáticos simplificados que, em sua maioria, são inspirados no modelo biológico do sistema nervoso para tentar abstrair mapeamentos de novos exemplos nas saídas desejadas ou agrupamentos de exemplos similares.
- Evolucionário: Baseia-se nos modelos biológicos da evolução natural e da reprodução genética para evoluir soluções que competem entre si, a fim de abstrair um modelo que solucione o problema em questão.

Segundo Russel e Norvig (2013), a aprendizagem é classificada conforme as suas variáveis de entrada e de saída, obtendo a capacidade de aprender uma função que preveja a saída conveniente para os novos valores de entrada. Existem três tipos de aprendizagem que caracterizam diferentes formas de respostas: aprendizagem supervisionada, aprendizagem por reforço e a aprendizagem não supervisionada. A aprendizagem supervisionada precisa encontrar uma função que se aproxime da função verdadeira, ou seja, tanto o valor de entrada como o de saída são conhecidos através dos dados. Esse modelo de aprendizagem busca por esta nova função através de um espaço de hipóteses possíveis que atendam em um bom desempenho. Quando a saída de valores for caracterizada por um conjunto de valores finitos, o

problema de aprendizagem será chamado de um problema de classificação. Quando o valor da saída for um número, será um problema de regressão. (RUSSEL; NORVIG, 2013).

Na aprendizagem por reforço também se reconhece as entradas e saídas, porém as saídas devem ser informadas se estiverem certas ou erradas, sendo responsabilidade do agente decidir em quais ações realizadas anteriormente obtive-se o maior reforço e quais delas foram responsáveis por isto. (RUSSEL; NORVIG, 2013). Por fim, Russel e Norvig (2013) explicam que a aprendizagem não supervisionada não possui um valor conhecido na saída da inteligência, por isto, a tarefa mais comum para esta aprendizagem é o agrupamento dos dados (também conhecido como *clustering* de dados), onde são encontrados grupos para os tipos de entradas obtidos. Para Wu (2012) uma análise realizada através da técnica de *clustering* possibilita encontrar automaticamente os grupos de objetos conceitualmente mais significativos para a aprendizagem, ajudando na análise dos dados, assim como na descrição dos mesmos e também em como utilizar estas informações que antes estavam ocultas dentro destes grupos. Um dos diversos métodos de agrupamento de dados é o método *K-Means*, utilizado para calcular a média da distância entre os pontos do mesmo *cluster*. (ARTHUR; VASSILVITSKII, 2007).

Com isto, muitos destes algoritmos de Inteligência Artificial, como Aprendizado de Máquina e técnicas de *clustering* necessitam de uma capacidade de processamento alto e de servidores de ponta, por isso, acabam muitas vezes atuando na computação em nuvem e necessitando da computação de borda para processar o mais próximo possível os dados brutos do sensor, conforme será explicado posteriormente.

2.3 COMPUTAÇÃO DE BORDA

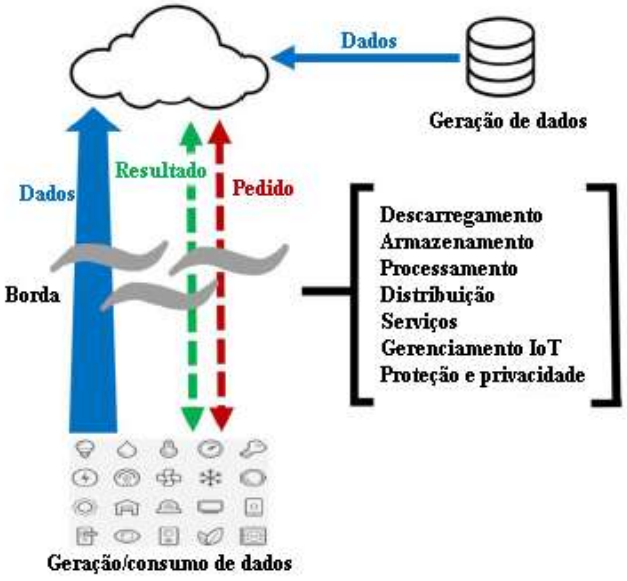
A computação de borda é caracterizada por processar os dados de dispositivos IoT (*Internet of Things* – do inglês Internet das Coisas) na borda da rede onde os dados são produzidos, possibilitando uma maior eficiência quando comparado ao envio de todos os dados diretamente para a computação feita em nuvem. Isto acontece devido a capacidade dos aplicativos IoT de possuirem um tempo de resposta muito curto, envolverem dados privados ou produzirem uma vasta quantidade de dados, o que representa uma carga pesada para as redes de comunicação. (SHI *et al.*, 2016).

Para Shi *et al.* (2016), colocar em nuvem todas as tarefas é uma maneira eficiente para realizar o processamento de dados. Porém, ao comparar a velocidade com que estes dados são gerados para serem enviados ao processamento com a largura de banda necessária, chega-se a um impasse, pois a crescente quantidade de dados gerados na borda faz com que a velocidade

de transporte dos mesmos se torne um gargalo para o paradigma da computação baseada em nuvem. Caso todos os dados sejam necessários e precisem ser enviados, o tempo de resposta será muito longo, sem elencar a falta de confiabilidade na rede, já que ela passará a receber uma grande quantidade de dados. Com isto, realizar o refinamento destes dados na borda traz um tempo de resposta mais curto, um processamento mais eficiente e uma menor pressão na rede.

O termo "borda" é definido neste contexto como qualquer recurso de computação e rede ao longo do caminho entre as fontes de dados, neste caso os dispositivos IoT e os servidores em nuvem, aproximando os recursos entre si. (DOLUI; DATTA, 2017). A Figura 2 exemplifica o fluxo dos dados.

Figura 2 – Fluxos de Computação Bidirecional



Fonte: SHI *et al.* (2016).

Presente no modelo da computação de borda, os dispositivos não são tratados apenas como consumidores de dados já que também acabam atuando como geradores de dados. Com isto, os dispositivos passam a não apenas solicitar algum serviço ou conteúdo provenientes da nuvem, como também executam as suas tarefas através da computação em nuvem. A borda tem a capacidade de executar o descarregamento dos dados, o armazenamento em cache e o processamento, bem como distribuir o serviço de solicitações e entregas da nuvem ao usuário, além de tomar decisões e executar ações. (SHI *et al.*, 2016). Segundo Dolui e Datta (2017), torna-se essencial que a borda seja projetada para atender aos requisitos de maneira eficiente, além de entregar confiabilidade, segurança e proteção com privacidade.

A ciência da computação de borda é efetivada na prática devido a separação por níveis da automação industrial. Djiev (2003) aborda os mesmos como sendo nível de campo, nível de

controle e nível de informação. O nível de campo é o mais baixo da hierarquia, sendo aquele que se encontra mais próximo do processo produtivo, tendo os dispositivos atuadores e os sensores e transmitindo dados entre o produto manufaturado e o processo; o nível de controle é aquele em que o fluxo de informação consiste em carregar os programas, os parâmetros e os próprios dados; o nível de informação é o topo da hierarquia, onde as informações dos demais níveis são monitoradas e servem para realizar o gerenciamento de todo o sistema de automação.

Mařík e Pechoucek (2001) acercam-se ainda que as aplicações industriais que empregam tecnologias baseadas em agentes para a manufatura também são divididas em níveis parecidos com os de sistemas de automação, sendo eles o controle de fabricação em tempo real, o gerenciamento de produção, e as empresas virtuais que integram a fabricação, as redes para vendas, fornecedores e canais de distribuição. Assim, este tipo de tecnologia estabelece uma sintonia entre os maiores níveis de gestão e os menores, resultando em ganhos de eficiência em praticamente todas as áreas do processo de manufatura.

A computação de borda está entre um dos principais elementos da IIoT, pois torna viável mesclar os dados industriais e transformá-los em itens de fácil utilização em outros serviços, como para a realimentação da própria manufatura, tornando possível a integração entre todas as etapas, desde a cadeia de suprimentos até a chegada do produto no cliente. (RESENDE, 2019). Com isto, será possível conferir na próxima seção como este conceito de computação de borda está sendo aplicado em estudos de casos, assim como os demais tópicos elencados neste capítulo.

3 ESTADO DA ARTE

Neste capítulo se faz presentes os trabalhos correlatos com a proposta deste trabalho, relacionando conceitos de Indústria 4.0, IIoT e Aprendizado de Máquina com diferentes tipos de aplicações de metodologias e ferramentas. A seguir estão apresentados, tendo ao final do capítulo um resumo dos mesmos destacando sua relação direta com a proposta do presente trabalho.

Proposto por Sezer *et al.* (2018), o trabalho trata dos conceitos, materiais e métodos para desenvolver uma arquitetura de Indústria 4.0 aplicada à manutenção preditiva, objetivando obter baixos custos de implantação deste tipo de tecnologia para pequenas empresas. No estudo, realizou-se a captura de dados de temperatura e vibração de uma máquina CNC, compartilhando estes dados com a nuvem, na qual eram aplicadas técnicas de particionamento recursivo e árvore de decisão para prever a rejeição de itens produzidos pela máquina que obtinham baixa qualidade. Para isto, utilizou-se o computador *single board* Raspberry Pi 3, responsável pela aquisição dos dados, o processamento e a conexão com a rede de internet para a transmissão dos mesmos. Junto a isto, para a estrutura de banco de dados disponível na nuvem, utilizou-se a plataforma *Dropbox*, aplicativo que pode ser instalado e configurado diretamente no processador. Concluindo o método, utilizou-se a ferramenta estatística de programação R para criar o modelo de Aprendizado de Máquina. Constatou-se que os principais resultados obtidos através deste estudo foram a estabilidade e a alta durabilidade física da arquitetura de baixo custo montada para os experimentos, além de ter a capacidade de prever através da integração das técnicas de Aprendizado de Máquina, que em média 81% das peças usinadas excederiam o limite de qualidade exigido.

Seguindo no mesmo ramo da Indústria 4.0, Kanawaday e Sane (2017) descrevem um estudo de caso aplicado à uma máquina de cortes em que se prevê a possibilidade de falhas e defeitos de qualidade através de modelos supervisionados de Aprendizado de Máquina. Combinando a comunicação M2M (*Machine-to-Machine* – do inglês Máquina à Máquina), a coleta de dados realizada através de um CLP (Computador Lógico Programável) que controla a máquina e a análise de dados baseada em BI (*Business Intelligence* – do inglês Inteligência Empresarial), se torna possível providenciar prognósticos para máquinas da indústria, alavancando a produção e prevendo as possíveis falhas de produtos. A comunicação M2M é realizada através de uma interface RS485 que envia o dado para um adaptador, o qual converte o mesmo em TCP (*Transmission Control Protocol* – do inglês Protocolo de Controle de Transmissão) para que seja recebido pelo computador industrial, sendo este último responsável

por fazer a conexão com a internet e, por consequência, a disponibilização destes dados na nuvem através de protocolo MQTT (*Message Queuing Telemetry Transport* – do inglês Transporte de Sistema de Mensagens). Com os dados salvos na nuvem, aplicou-se o modelo ARIMA (*Autoregressive Integrated Moving Average*) para a análise preditiva. Resultou-se que dentre os quatro tipos de modelos supervisionados utilizados (*Naive Bayes, Support Vector Machine, Classification And Regression Trees* e *Deep Neural Network*), a rede de *Deep Neural* tornou-se a mais efetiva na modelagem dos dados, contudo, as ocorrências de ciclos de má qualidade foram baixas, comparadas aos ciclos de boa qualidade da produção. Assim, o modelo necessita estar em constante aprendizado para continuar aprimorando suas previsões.

Abordando o tema de IIoT, Strauss *et al.* (2018) tratam de uma solução de baixo custo para implantação da manutenção preditiva através do *retrofitting* de máquina combinado com sensores baratos, arquitetura IIoT e Aprendizado de Máquina, aplicando o estudo de caso em uma ponte rolante. Como método, a aplicação utilizada para coleta de dados dos sensores, preparação destes dados e comunicação com a nuvem de IIoT, foi o produto da *Microsoft*, *Azure IoT Device SDK*. Obteve-se os dados através da interface feita entre os sensores disponibilizados na máquina com o CLP. Os mesmos foram preparados através da biblioteca *Pandas Python* em uma máquina virtual com o ambiente *Jupyter Notebook*. Na parte de Aprendizado de Máquina, o estudo faz a análise de diferentes algoritmos supervisionados, não supervisionados e semi-supervisionados, e a comparação entre eles é baseada na sua performance individual. Dentre estas comparações, constatou-se que para modelos de Aprendizado de Máquina focados em classificação de falhas, os modelos supervisionados se tornam excelentes, assim como para a detecção de anomalias. Para realizar previsões em tempo real do estado da máquina, os modelos semi-supervisionados ofereceram a melhor performance. Além disso, o artigo traz o desafio em incorporar as soluções de manutenção preditiva nas condições já existentes de um *framework*. A arquitetura apresentada é de importante relevância, visto que a mesma apresenta um conceito da concentração de dados em um IoT *Hub* dedicado a todos os sensores utilizados no estudo.

No âmbito do Aprendizado de Máquina, Amruthnath e Gupta (2018) realizaram o estudo de aprendizado não supervisionado aplicado a dados de vibrações obtidos de um exaustor, possibilitando prever a detecção antecipada de falhas. Toda a pesquisa foi performada na ferramenta estatística de programação R, na qual definiu-se a hipótese de dois tipos de classificação dos dados, sendo separados por dados saudáveis e dados não saudáveis. Após as diversas comparações realizadas entre os métodos de aprendizado utilizado, concluiu-se que os

métodos de aprendizado não supervisionados trouxeram maiores resultados de previsão, comparados com metodologias antigas de análise de dados.

Outra pesquisa deste ramo é o estudo realizado por Butte *et al.* (2018), no qual se discute diversos tipos de algoritmos de Aprendizado de Máquina baseados nas estratégias de manutenção preditiva, propondo-se um estudo mais aprofundado em redes de *Deep Learning*. Aplicada a produção de semicondutores, a metodologia é realizada através da coleta dos dados obtidos dos equipamentos observados até que estes necessitassem a sua substituição, antes de começar a degradar a performance da máquina para não influenciar no treinamento da inteligência. Os dados foram obtidos através de 21 sensores atrelados as ferramentas, além de parâmetros de 3 processos de produção e dados de componentes críticos. Em valores, 80% dos dados foram utilizados para treinar os modelos enquanto 20% foram usados para testar e validar o modelo da performance. Assim, foi possível garantir que a estratégia adotada sinalizasse o tempo de falha antes de chegar a 25 peças produzidas, obtendo a previsão de manutenção necessária. O autor comenta que para ter uma abordagem eficaz, é necessário fazer uso de diversos tipos de dados provenientes de múltiplas fontes, como sensores, dados de processo e de manutenções passadas. Também é possível verificar que os métodos foram aplicados por sensores, podendo expandir o nível de detecção de falha.

Trazendo uma visão de como a coleta de dados é importante para o Aprendizado de Máquina, Cline *et al.* (2017) abordam um caso de estudo em que há a aplicação de diversas técnicas de aprendizado através da base de dados, de aproximadamente 19 anos, de inspeções realizadas em conectores do serviço de ativos de um fabricante de óleo e gás, com o intuito de fornecer periodicamente aos seus clientes quais os potenciais de falhas para estes tipos de conectores. As análises destes dados foram feitas na plataforma *ThingWorx*, onde a mesma pode entregar diversos tipos de análises em uma plataforma totalmente *online*. O melhor resultado encontrado para o estudo foi entregue pela rede neural com três camadas de neurônios, a qual observou todas as variáveis e constatou que 46% das peças sofreriam algum tipo de falha. Além disso, um dos pontos importantes a ser ressaltado do artigo é a necessidade de considerar que, ao aplicar em um ambiente dinâmico, é preciso verificar todos os tipos de ativos que se podem vir a ter durante todos os dias em uma máquina e/ou produção.

Por fim, Yan (2016) apresenta a técnica ELM (*Extreme Learning Machines*), usada para classificação de multi classes e de regressão. A técnica aplicada por Yan serve para o monitoramento das condições das máquinas industriais, avaliando de forma mais precisa e confiável a detecção de anomalias nos combustores de turbina a gás. Através da utilização de 27 sensores instalados em cada câmara de combustão, separou-se os dados entre eventos

considerados normais e anormais. Com isto, aplicou-se a tecnologia de *Deep Learning* para aprender as características dos sensores utilizados para realizar as medições, antes de passar para a parte de classificação *one-class* dos dados. Todos os experimentos foram performados em Matlab. Comprovou-se então que, para o caso estudado, a técnica ELM apresenta melhor performance do que a antiga técnica SMV (*Support Vector Machine)*, além de ser mais eficiente, visto que o objetivo principal seria monitorar a saúde da máquina, uma vez que a mesma opera em condições normais na maior parte do tempo e os eventos de falhas ocorridos acontecem em raros momentos.

O Quadro 2 apresenta de forma simplificada os estudos descritos anteriormente, além de listar as ferramentas abordadas em cada um e a sua relação com o estudo proposto por este trabalho.

Quadro 2 – Artigos da Pesquisa

Autor e Ano	**Título**	**Tema**	**Ferramenta**	**Relação com o Estudo**
SEZER, E.; ROMERO, D.; GUEDEA, F.; MACCHI, M.; EMMANOUILIDIS, C. 2018	*An Industry 4.0-enabled Low Cost Predictive Maintenance Approach for SMEs: A Use Case Applied to a CNC Turning Centre*	Indústria 4.0	Particionamento recursivo; Árvore de decisão.	Computação de borda
KANAWADAY, A.; SANE, A. 2017	*Machine Learning for Predictive Maintenance of Industrial Machines using IoT Sensor Data*	Internet Industrial das Coisas (IIoT)	*AutoRegressive Integrated Moving Average* (*ARIMA*)	Comunicação M2M; Redes neurais.
STRAUSS, P.; SCHMITZ, M.; WÖSTMANN, R.; DEUSE, J. 2018	*Enabling of Predictive Maintenance in the Brownfield through Low-Cost Sensors, an IIoT-Architecture and Machine Learning*	Internet Industrial das Coisas (IIoT)	Árvores de Decisão; Floresta Aleatória; SVM; Regressão logística; *K-NN*.	*IoT Hub* dedicado; *Microsoft Azure IoT Device SDK*; Pandas Python.
AMRUTHNATH, N.; GRUPTA, T. 2018	*A Research Study on Unsupervised Machine Learning Algorithms for Early Fault Detection in Predictive Maintenance*	Aprendizado de Máquina	*PCA* T^2; *Clustering* hierárquico; *K-Means*; *Fuzzy C-Means*; *Clustering* modelo básico.	Ferramenta estatística de programação R

BUTTE, S.; PRASHANTH, A. R.; PATIL, S. 2018	*Machine Learning Based Predictive Maintenance Strategy: A Super Learning Approach With Deep Neural Networks*	Aprendizado de Máquina	*Deep Learning*; Floresta aleatória; Regressão logística; Aumento de Gradiente.	*Deep Learning*
CLINE, B.; NICULESCU, R. S.; HUFFMAN, D.; DECKEL, B. 2017	*Predictive Maintenance Applications for Machine Learning*	Aprendizado de Máquina	Regressão Linear; Regressão Lógica; Redes Neurais; Árvores de Decisão; Floresta Aleatória; Aumento de Gradiente.	Plataforma *ThingWorx*
YAN, W. 2016	*One-Class Extreme Learning Machines for Gas Turbine Combustor Anomaly Detection*	Aprendizado de Máquina	*ELMs – Extreme Learning Machines*	Matlab; ELM

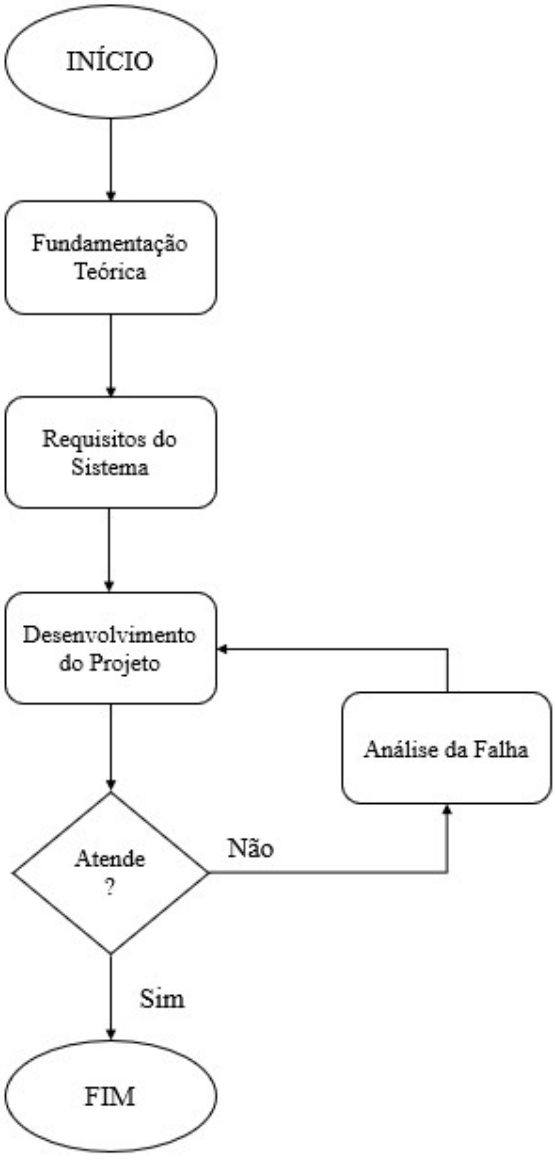
Fonte: Elaborado pela autora.

A partir destes trabalhos correlatos estudados, percebe-se a elevação de estudos de casos aplicados à manutenção preditiva que empregam tecnologias emergentes no mercado para encontrar a melhor solução dos diversos problemas encontrados. Observa-se que é tangível realizar um estudo com tecnologia de baixo custo, com placas para aquisição de dados como o Raspberry Pi 3, utilizando plataformas *open source,* como a plataforma da *Microsoft* para desenvolvimento de projeto de IoT*,* e métodos que proporcionam ótimos resultados, como redes neurais e *Deep Learning*, aplicando os conceitos de computação de borda e Aprendizado de Máquina integrado à estas soluções.

4 METODOLOGIA

Este capítulo descreve a metodologia para realizar o presente trabalho. Nele estarão contidos o fluxo do projeto expondo a relação da parte prática a partir dos assuntos abordados no Capítulo 2, a proposta do sistema a ser implantado, o fluxo da análise dos dados e o fluxo de testes que serão aplicados ao estudo de caso.

Figura 3 – Fluxos do Projeto



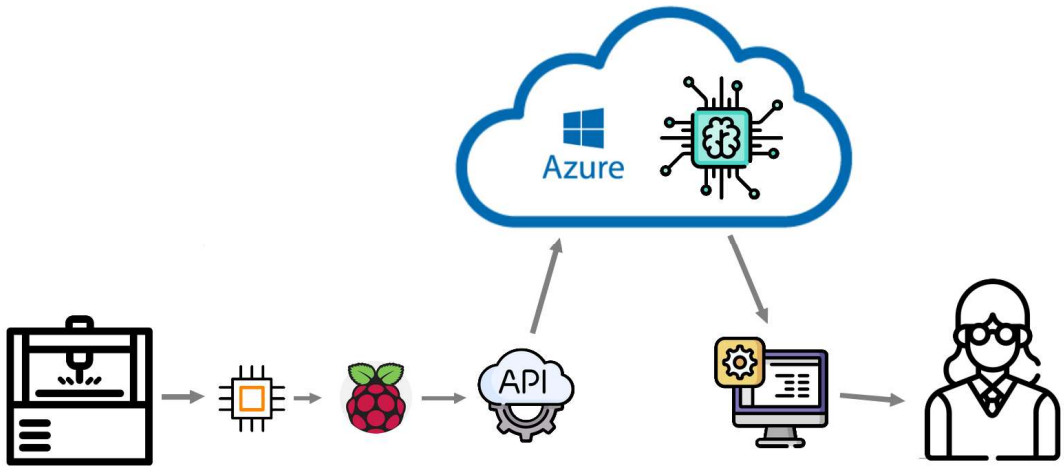
Fonte: Elaborada pela autora.

Apresenta-se na Figura 3 o fluxo de projeto proposto para este estudo. Após todo o embasamento obtido através da pesquisa realizada no Capítulo 2 e as contribuições obtidas do Capítulo 3, definem-se os requisitos necessário para o sistema, como *hardware* e plataforma de

computação em nuvem a ser utilizados. Logo, realiza-se o desenvolvimento do projeto em si, contemplando todos os materiais elencados pelos requisitos e, se o mesmo tem resultado satisfatório, encerra-se o fluxo do projeto, se não, analisa-se o que ocorreu e retorna-se para o desenvolvimento do projeto corrigindo os erros encontrados até que os resultados obtidos sejam satisfatórios.

Inserido no desenvolvimento do projeto, estão presentes as partes de comunicação com a máquina através de um sistema embarcado, a computação de borda, o Aprendizado de Máquina em nuvem e a apresentação dos dados em uma interface intuitiva para o usuário, conforme exemplificado na Figura 4.

Figura 4 – Sistema Proposto



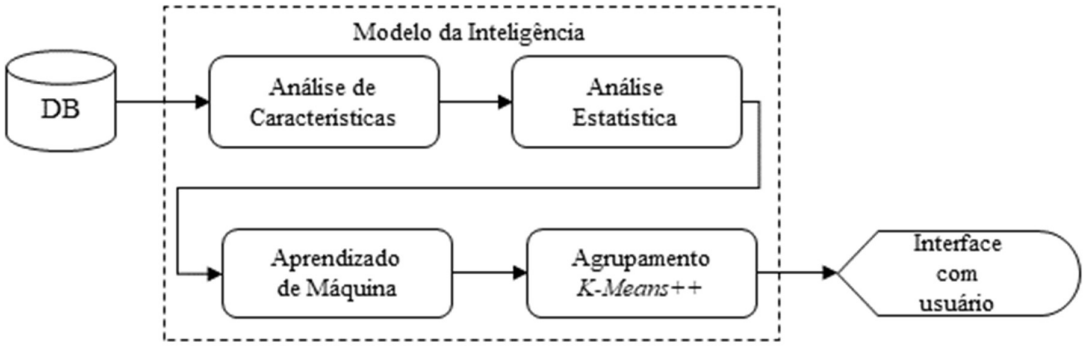
Fonte: Elaborada pela autora.

O sistema proposto apresenta como são aplicados os materiais no desenvolvimento. Para realizar a comunicação com a máquina, pressupõe-se que nela já estejam instalados os instrumentos e sensores necessários para realizar a leitura dos parâmetros que irão auxiliar na identificação da causa da anomalia. Os dados devem ser recebidos pelo controlador da máquina e após realiza-se a comunicação com o Raspberry Pi, sendo este o responsável por realizar a computação de borda e refinar os dados mais relevantes para serem enviados a nuvem por meio de uma API (*Application Programming Interface* – do inglês Interface de Programação de Aplicativos). Os dados estão alimentando a plataforma *Azure Machine Learning Studio* da *Microsoft* através de um *web server*, sendo a plataforma responsável por realizar toda a parte de Aprendizado de Máquina em que o algoritmo irá aprender com os dados de entrada e retornar as possíveis anomalias que podem ocorrer no futuro. Os resultados do algoritmo da inteligência

serão enviados a uma interface para apresentar de forma simples e intuitiva os dados para o usuário final.

A plataforma *Azure Machine Learning Studio* corresponde por um serviço PaaS (*Platform as a Service* – do inglês Plataforma como Serviço) em que a própria plataforma está à disposição do usuário por meio de uma estrutura computacional externa e possibilita o acesso a ela por meio de uma conexão com a internet. Logo, toda a parte de processamento e análise de dados estará sendo executada em uma infraestrutura de rede à parte da estrutura física do sistema proposto. A Figura 5 proporciona uma maior compreensão de como os dados serão analisados na plataforma.

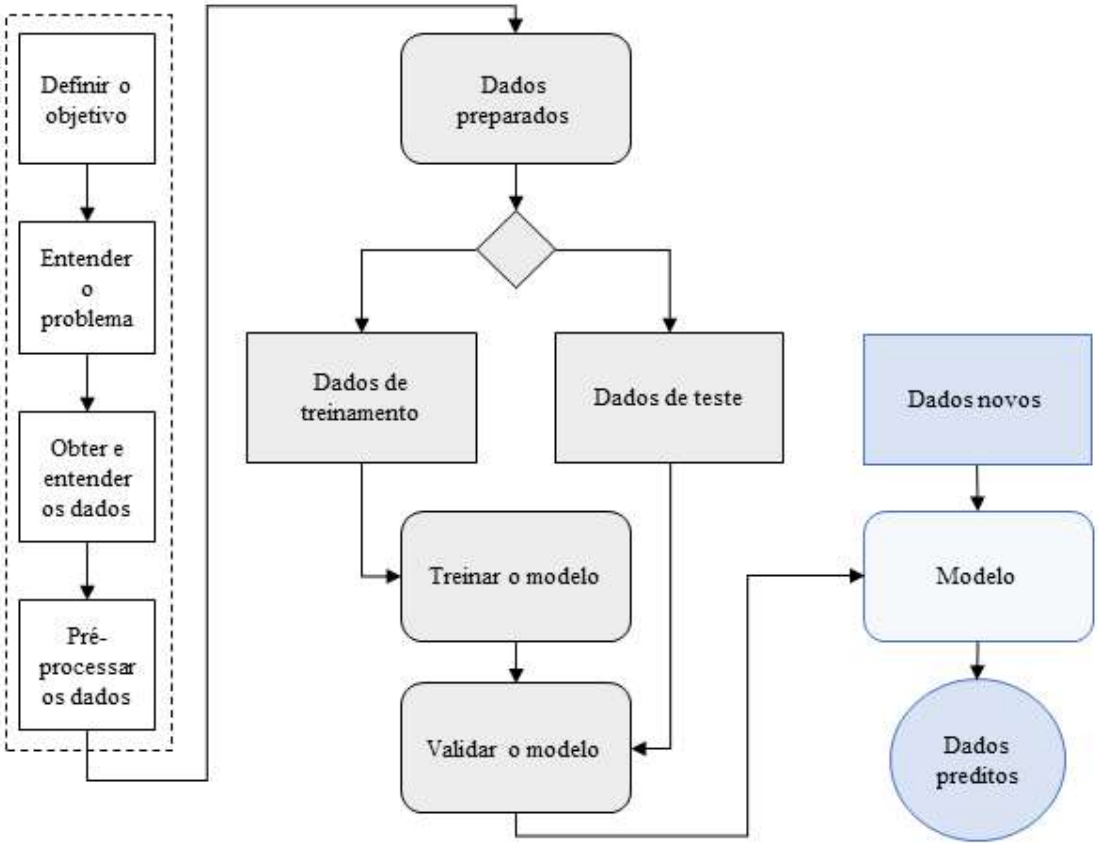
Figura 5 – Análise de Dados



Fonte: Elaborada pela autora.

Os dados devem ser analisados quanto as suas características com o objetivo de eliminar os ruídos, inconsistências, problemas estruturais e até mesmo analisar valores que podem estar fora do comum, como possíveis valores de extremo mínimo e máximo. Após este filtro, os mesmos devem passar por uma análise estatística aplicando as devidas funções por meio de *scripts* de auxílio na atividade de Aprendizado de Máquina. O processo de aprendizado é aquele que aplica estas técnicas estatísticas à grandes quantidades de dados e identifica os padrões que especificam a resposta correta a um determinado problema. O aprendizado será não supervisionado, visto que o mesmo realiza o treinamento do modelo através do cálculo de médias entre os *clusters* por meio da técnica de *K-Means++,* sendo capaz de definir o grupo para cada tipo de valores das variáveis de interesse dos novos dados. Este treinamento consiste em fazer com que o algoritmo encontre padrões e, ao final, será possível predizer os resultados através da entrega ao usuário. A Figura 6 traz o fluxo de testes deste modelo de inteligência em uma forma mais detalhada.

Figura 6 – Fluxo de Testes



Fonte: Elaborada pela autora.

A preparação dos dados parte do momento em que o objetivo de utilizar um modelo inteligente é definido, com isto, será necessário entender o problema evidenciado pela máquina, coletar os dados e analisá-los através de um pré-processamento. Logo, os dados estarão preparados para serem separados entre aqueles que serão os dados treinados e aqueles que servirão para testes após o treinamento do modelo e validarão o mesmo. Geralmente, os dados são divididos entre 70% para treino e 30% para testes, podendo haver alterações nestes valores conforme o tipo de problema. Por fim, o modelo estará criado, permitindo a entrada de novos dados e a partir disto, realizará a predição desejada.

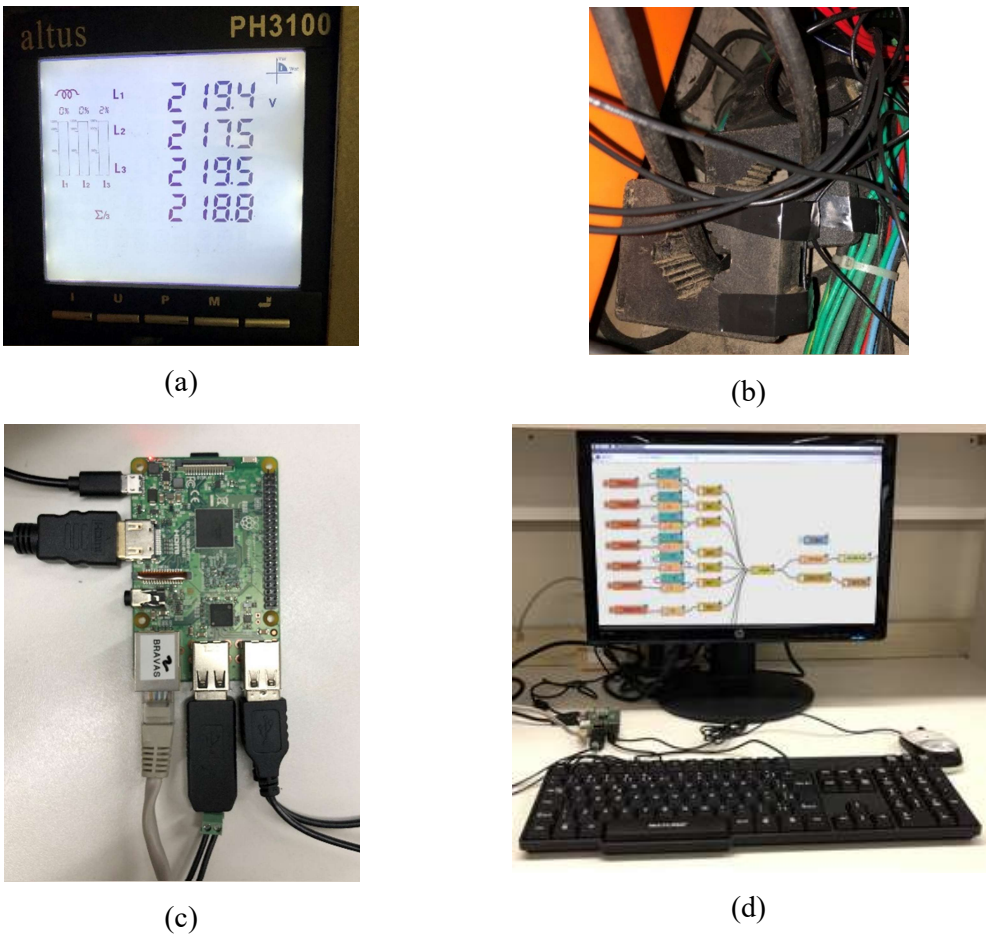
Importante ressaltar a engenharia de recursos necessária para realizar a seleção das variáveis com maior grau de relevância para atingir o objetivo do modelo. Sempre será necessário realizar uma amostragem significativa para que se possa testar e comprovar a real obtenção de resultados concisos gerados por ele. Esta metodologia está aplicada conforme descrito na próxima seção, onde estarão estratificadas as ações fundamentais para a efetivação deste estudo.

5 ANÁLISE DE RESULTADOS

A partir da metodologia proposta, estão presentes neste capítulo o estudo de caso realizado, exemplificando cada etapa do sistema proposto, assim como a realização da coleta de dados, a descrição da técnica de *K-Means* aplicada à Aprendizagem de Máquina e o treinamento e teste do aprendizado através da plataforma *Azure Machine Learning Studio*. Também está composto o resultado final do estudo, apresentando os testes realizados e como os resultados estão expostos ao usuário final através da interface integrada.

O sistema IIoT deste estudo de caso é caracterizado por medir as unidades de tensão e corrente elétrica do laboratório de máquinas elétricas da Unisinos. Emulou-se neste sistema o estudo proposto visando que, com as adaptações necessárias de instrumentos, o mesmo pode ser aplicado à uma máquina da indústria sem grandes dificuldades, possibilitando o mesmo tipo de análise para diversos equipamentos. Abaixo estão dispostas as imagens provenientes do experimento, assim como uma breve explicação sobre cada item.

Figura 7 – Componentes do Sistema IIoT



(a) (b)

(c) (d)

Fonte: Elaborada pela autora.

Na Figura 7a exibe-se o MGE (Medidor de Grandezas Elétricas) PH3100 instalado ao circuito trifásico principal do painel geral de distribuição de cargas do laboratório de máquinas elétricas. Neste medidor, é possível realizar a conexão direta dos cabos das três tensões e, com a instalação de transformadores de corrente através dos cabos das fases, também realizar a conexão dos cabos para leitura das correntes elétricas de cada fase (Figura 7b). O medidor possui alimentação 220 V em corrente alternada e uma interface de comunicação serial RS485 com protocolo de comunicação MODBUS-RTU, que possibilita a obtenção dos valores lidos. (ALTUS, 2015).

A Figura 7c traz a *single board* Raspberry Pi 3. Nela conectou-se o adaptador USB para serial RS485, tendo a outra extremidade conectada ao MGE, além dos periféricos de monitor, *mouse*, teclado e alimentação 5 V em corrente contínua através de fonte chaveada. Na *board,* habilitou-se a entrada USB0 como comunicação serial para realizar a troca de dados com o medidor.

Obtendo as configurações de *hardware* prontas, iniciou-se a parte de programação do *software* para obter os dados do medidor, mostrar os dados na interface do usuário, salvar os dados em um arquivo CSV (*Comma-Separated-Values* – do inglês Valores Separados por Vírgulas) e enviar os mesmos para a plataforma *Azure*®. Para isto, fez-se uso da plataforma de desenvolvimento *Node-Red* (Figura 7d)*,* a qual é uma ferramenta de desenvolvimento baseada em fluxo propiciando a uma programação visual capaz de conectar dispositivos de *hardware*, APIs e serviços *online* como parte da IoT. O fluxo desenvolvido para este estudo pode ser verificado no Apêndice A deste trabalho.

Para a coleta de dados, definiu-se que seriam coletados do MGE os valores das três tensões de fase, as três correntes elétricas de cada fase e a frequência elétrica do sistema. Para detectar o período em que ocorreu a anomalia no sistema, foi necessário adicionar ao arquivo CSV o dia da semana e o horário que foram adicionados pela plataforma *Node-Red*. O período de coleta dos dados foi de meio segundo a um segundo, conforme às atividades desenvolvidas no laboratório. Na Figura 8 pode-se verificar um pequeno trecho do conjunto de dados obtido.

Figura 8 – Exemplo do Conjunto de Dados

Tensão L1 (V)	Tensão L2 (V)	Tensão L3 (V)	Corrente I1 (A)	Corrente I2 (A)	Corrente I3 (A)	Frequência (Hz)	Dia da Semana	Hora
222,7	220,6	222,9	0,7	0,9	3,2	60	Qua	16:24:44
222,7	220,7	222,9	0,7	0,9	3,2	60	Qua	16:24:45
222,5	220,5	222,8	0,7	0,9	3,2	60	Qua	16:24:46
222,8	220,7	222,9	0,7	0,9	3,2	60	Qua	16:24:47
222,8	220,8	222,9	0,7	0,9	3,2	60	Qua	16:24:48
222,8	220,8	222,9	0,7	0,9	3,2	60	Qua	16:24:49
222,6	220,6	223	0,7	0,9	3,2	60	Qua	16:24:50

Fonte: Elaborada pela autora.

5.1 ESCOLHA DO CONJUNTO DE DADOS

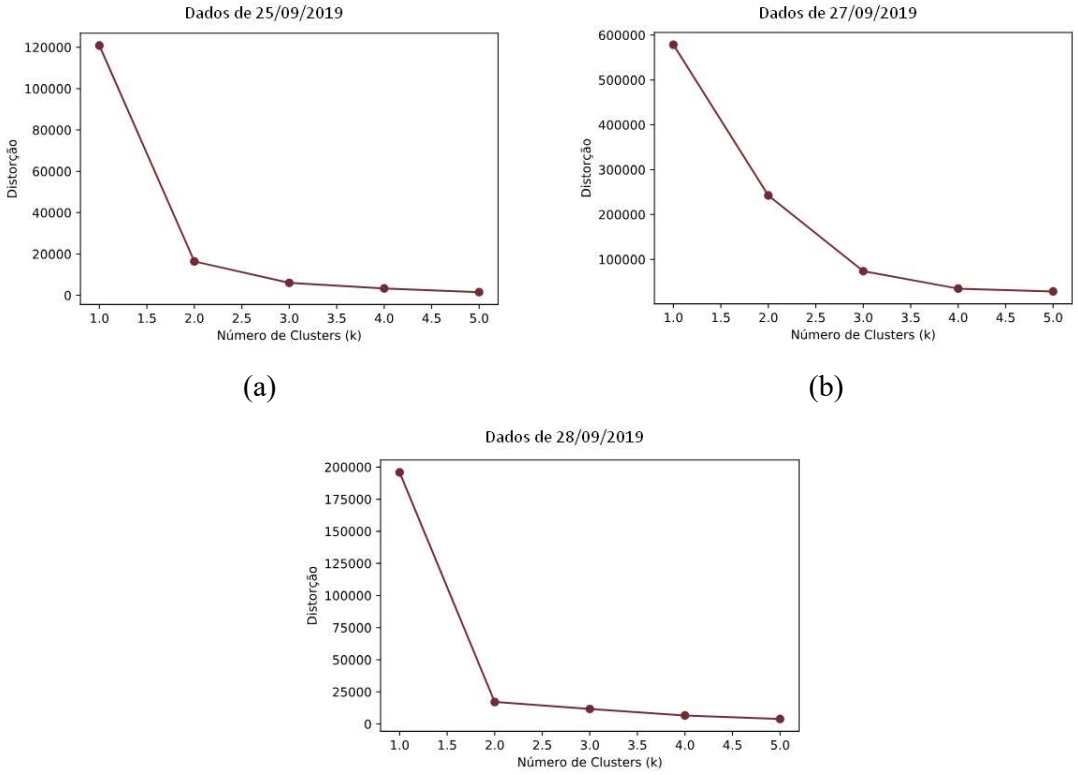
Iniciou-se o processo de treinamento da aprendizagem de máquina a partir da escolha do conjunto de dados em que se apresenta a melhor distribuição dos mesmos e aquele que mais coincide-se com a metodologia proposta no Capítulo 4. Para isto, foram separados três conjuntos de dados de períodos de atividades diferentes realizadas no laboratório. Considerou-se apenas o período do dia em que o laboratório permaneceu aberto aos alunos, ou seja, das 8 horas da manhã às 23 horas da noite.

Realizada a escolha dos conjuntos de dados a serem testados, dividiu-se cada conjunto entre 70% dos dados para treino e 30% para testes posteriores, visando também que os três conjuntos testados obtivessem aproximadamente a mesma quantidade de dados para melhor comparação entre os resultados dos testes.

A primeira parte de testes realizou-se através de um *script* em Python, para verificar a quantidade mais assertiva de número de centroides (*clusters*). Optou-se por treinar o modelo a partir dos valores das correntes elétricas, sendo estes os dados que apresentaram resposta mais rápida de variação de estado conforme o tipo de atividade exercida no laboratório. Como o gráfico dos centroides é plotado em duas dimensões, necessitou-se realizar a redução de três colunas de dados para duas dimensões através do método de PCA (*Principal Component Analysis* – do inglês Análise de Componentes Principais). Esta técnica consiste em uma análise multivariada que analisa as inter-relações entre um grande número de variáveis e retorna em termos de componentes, ou seja, a informação de várias variáveis é condensada em um conjunto menor de variáveis estatísticas com a menor perda possível de informações. (OLIVEIRA, 2019).

Após atenuada a dimensão dos dados, aplicou-se o método *elbow* para encontrar o número real de centroides a serem divididos os dados. Este método consiste em rodar o algoritmo em *loop* múltiplas vezes, com um número crescente de opções de *clusters* e, em seguida, plotar a relação de distorção em função do número de centroides. O valor a ser definido é aquele anterior a linearização da curva. (SARKAR, 2019). Na imagem que segue, pode-se verificar a resposta deste método para cada conjunto de dados.

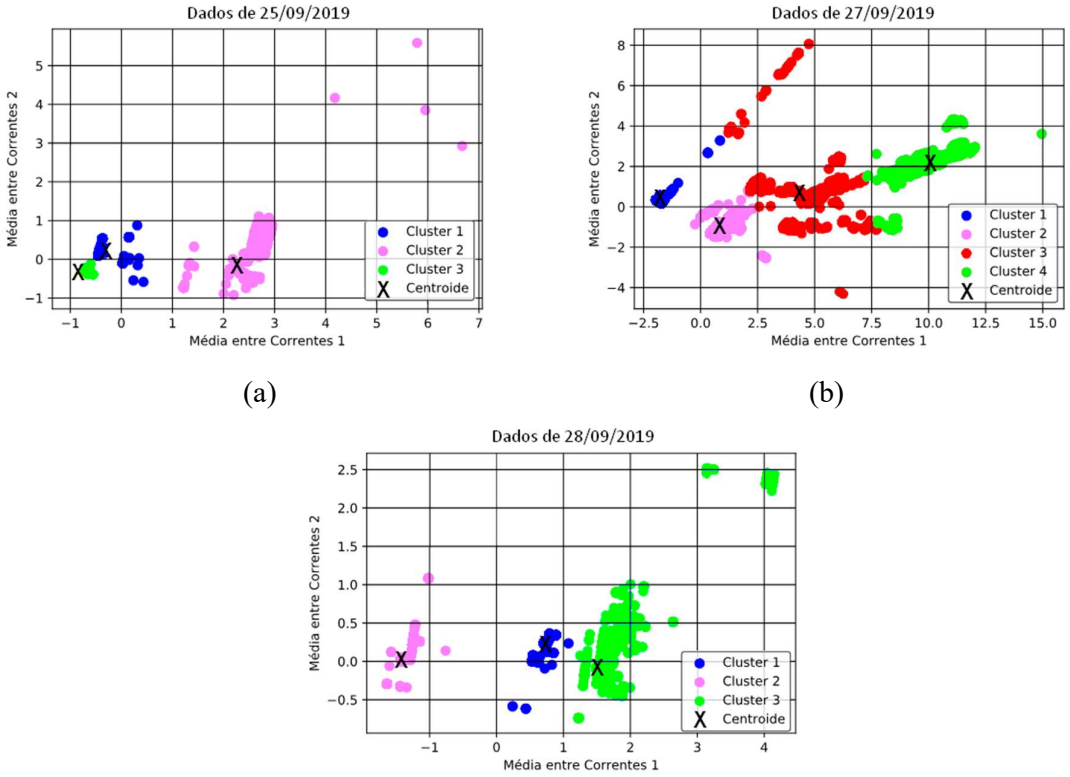
Figura 9 – Respostas do método *elbow*



Fonte: Elaborada pela autora.

Conforme a Figura 9a e 9c, é possível observar que o primeiro e o terceiro conjunto de dados devem ser treinados com um número de 3 *clusters* para o agrupamento dos dados. Já o resultado da Figura 9b apresenta que este conjunto de dados deve ser treinado para até 4 números de *clusters*. Assim, obtendo os valores de número de centroides para treinamento, em seguida, aplica-se o treinamento da técnica de *K-Means++* para verificar a distribuição dos dados entre os centroides. Em Python, a função de agrupamento *K-Means* é acessada através da biblioteca *sklearn,* própria para modelos de Aprendizado de Máquina. Como explicado no Capítulo 2, optou-se por utilizar desta técnica visto que o valor de saída da inteligência é desconhecido, por isto o agrupamento dos dados com características semelhantes baseadas no valor da média entre os dados e os centroides será o método mais indicado para o problema proposto na metodologia deste trabalho. O resultado de cada treino pode ser visto na Figura 10, bem como um resumo da maneira em que os dados foram distribuídos no Quadro 3.

Figura 10 – Gráficos de dispersão entre médias das correntes



Fonte: Elaborada pela autora.

Através da Figura 10, torna-se possível observar como os dados foram agrupados entre os centroides no treinamento. Na Figura 10a ressalta-se que a distância entre o *cluster* 1 e o *cluster* 3 se faz muito pequena, acarretando em erros que possam ocorrer ao realizar a classificação de dados novos. Já o agrupamento dos dados nas Figuras 10b e 10c podem ser qualificados como ótimos para uma nova classificação de dados. Quantificando os dados por *clusters,* elaborou-se o Quadro 3 que auxilia no entendimento de como os dados foram distribuídos.

Quadro 3 – Distribuição de dados por *cluster*

Conjunto de Dados	**Nº de *Clusters***	**Quant. de Linhas**	***Cluster* 1**	***Cluster* 2**	***Cluster* 3**	***Cluster* 4**
Dados 25/09/2019	3	93463	50416	16702	26345	-
Dados 27/09/2019	4	93020	47773	33709	10261	1277
Dados 28/09/2019	3	93252	9499	45666	38037	-

Fonte: Elaborada pela autora.

Com isto, analisando-se o Quadro 3 juntamente com a Figura 10, percebe-se que os dados de 27/09/2019 exemplificam melhor os momentos em que acontecem anomalias no sistema, ao categorizar dados mais precisos no *cluster* 4. Sendo assim, este será o conjunto de

dados adotado para o treinamento e teste da inteligência na plataforma *Azure*®. O *script* utilizado para esta parte do estudo pode ser analisado através do Apêndice B presente neste trabalho.

5.2 TREINAMENTO DA APRENDIZAGEM DE MÁQUINA

Após realizada a escolha do conjunto de dados a ser treinado, partiu-se para a aplicação na plataforma *Azure*®. A plataforma está disponível em ambiente de nuvem oferecendo gerenciamento completo para compilar, implantar e compartilhar soluções de análise preditiva com extrema facilidade. A plataforma ainda permite fazer o uso do modelo criado a partir de um serviço *web*, ou seja, ser acessado em tempo real. (MICROSOFT, 2019).

Iniciou-se o experimento realizando o *upload* do conjunto de dados do dia 27/09/2019 no formato de arquivo CSV. Após, selecionou-se as colunas de dados desejados, retirando as colunas de dias da semana e de horários. Assim como no *script* em Python, os dados foram divididos entre treino (70%) e teste (30%). Seguindo com o experimento, selecionou-se o método de *K-Means Clustering*, definindo a quantidade de centroides em 4 e a inicialização com a técnica de *K-Means*++. Após, associou-se os dados de treino com o método *K-Means* através do modelo de treino de agrupamento da plataforma. Neste modelo, é necessário selecionar as colunas que estão relacionadas, no caso, as colunas com os valores das correntes elétricas. Este componente limita as colunas e com isto, reduz o tamanho do conjunto de dados que será feito o treinamento.

Posteriormente, o modelo estará treinado e poderá ter dados associados a ele. Designando os dados de teste através do modo para atribuir dados a *clusters* da plataforma, pode-se verificar como os mesmos foram atribuídos e qual o resultado das médias para cada um dos quatro centroides. Uma prévia deste resultado pode ser vista no Quadro 4 que segue abaixo.

Quadro 4 – Resultado das médias entre *clusters* para cada linha de dados

Atribuições (*clusters*)	**Distância para *Cluster* 0**	**Distância para *Cluster* 1**	**Distância para *Cluster* 2**	**Distância para *Cluster* 3**
2	3,05E+14	6,37E+14	0,00E+00	1,24E+14
1	3,27E+14	2,84E+13	4,85E+14	8,47E+14
1	4,10E+14	8,53E+14	6,70E+14	5,89E+14
1	3,62E+14	5,71E+14	5,95E+14	6,45E+14
0	9,33E+14	3,89E+14	3,07E+14	9,93E+14
0	6,04E+14	4,40E+14	3,10E+14	1,04E+14
0	1,43E+14	3,84E+14	3,17E+14	9,88E+14

2	3,05E+14	6,37E+14	0,00E+00	1,24E+14
0	3,54E+14	3,59E+14	3,31E+14	9,65E+14
0	9,33E+14	3,89E+14	3,07E+14	9,93E+14
2	3,05E+14	6,37E+14	0,00E+00	1,24E+14
1	4,03E+14	5,83E+14	6,61E+14	5,96E+14
2	3,05E+14	6,37E+14	0,00E+00	1,24E+14
0	9,33E+14	3,89E+14	3,07E+14	9,93E+14
1	6,44E+14	2,88E+14	8,47E+14	4,91E+14
2	3,05E+14	6,37E+14	0,00E+00	1,24E+14
2	3,05E+14	6,37E+14	0,00E+00	1,24E+14

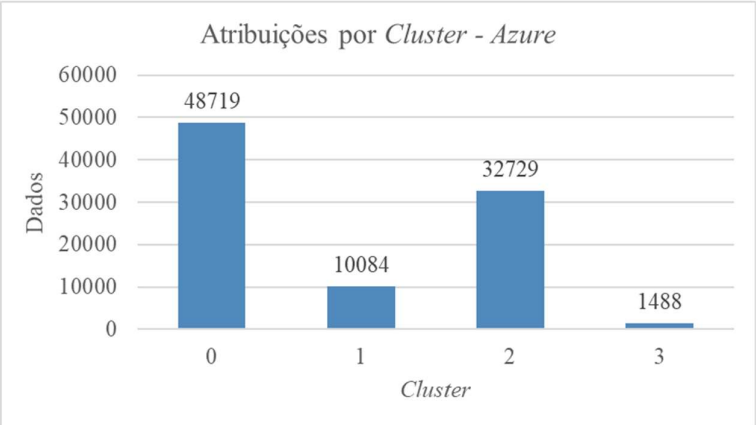
Fonte: Elaborada pela autora.

No Quadro 4 verifica-se os resultados dos cálculos das médias entre os centroides obtidos para cada linha de dados das correntes elétricas e em qual *cluster* o dado acabou sendo categorizado. Com isto, possibilitou-se seguir com os testes do Aprendizado de Máquina.

5.3 TESTE DA APRENDIZAGEM DE MÁQUINA

Dando sequência ao experimento, iniciou-se o processo de teste da aprendizagem através da plataforma *Azure*®. Conforme exemplificado no Quadro 4, um teste prévio já havia sido feito ao designar os dados separados para testes aos *clusters* do treinamento. Ao converter a resposta das atribuições dos *clusters* do modelo de treino para um gráfico em barras, pode-se verificar como os dados foram distribuídos entre os centroides, conforme indicado na Figura 11 a seguir.

Figura 11 – Gráfico de distribuição dos dados entre *clusters* – Plataforma *Azure*®



Fonte: Elaborada pela autora.

Na Figura 11, pode-se perceber que das 93020 linhas de dados de treinamento, aproximadamente 52% ficaram classificados no *cluster* zero, 11% deles no *cluster* um, 35% no *cluster* dois e os 1,6% restantes no *cluster* três. Estratificando-os em uma pequena tabela, pode-

se observar quais os valores de corrente elétrica foram considerados para cada centroide, conforme Quadro 5.

Quadro 5 – Dados atribuídos aos *clusters*

Tensão L1 (V)	**Tensão L2 (V)**	**Tensão L3 (V)**	**Corrente I1 (A)**	**Corrente I2 (A)**	**Corrente I3 (A)**	**Frequência (Hz)**	**Dia da Semana**	**Horário**	***Cluster***
224,2	222,4	224,3	0,5	0,5	3,3	60	Qui	17:43:08	0
223,5	220,9	223,2	8,1	9,6	11,9	60	Qui	17:43:09	3
221,6	220,6	222	0,4	0,5	3,2	60	Qui	17:52:53	0
222,3	221,2	221,4	3,6	0,5	3,2	60	Qui	17:52:54	1
220,1	218,6	220,3	6,5	7,3	9,6	60	Qui	18:13:53	3
220,6	219,3	220,8	0,4	4,1	6,6	60	Qui	18:13:54	1
220,2	219,7	220,8	3,6	3,6	5,7	60	Qui	18:14:32	1
220,7	220,1	221,3	0,1	0,5	3,2	60	Qui	18:14:33	0
221,5	220,6	222,7	0	0	0,2	60	Qui	22:23:44	0
221,4	221,5	222,4	0	0	0	60	Qui	22:23:44	2

Fonte: Elaborada pela autora.

Analisando a maneira como os dados foram classificados aos centroides, pode-se rotular o tipo da atividade através destes valores. Com isto, o *cluster* 2 será considerado como sistema desligado, visto que os valores de corrente atribuídos a ele são de 0 A. O *cluster* 0 está atribuído a operação normal do laboratório onde a faixa das três correntes deve se manter em aproximadamente 3 A, já o *cluster* 1 será categorizado como uma operação no limite da capacidade, necessitando de atenção à atividade realizada, onde a corrente ultrapassa os 3 A e chega a aproximadamente 7 A e, por fim, o *cluster* 3 como aquele em que os valores caracterizam a anomalia propriamente dita, com correntes que ultrapassam o limite de 7 A, predizendo que o sistema necessita de maior atenção pois futuramente alguma falha possa vir a ocorrer. O fluxo deste experimento na plataforma pode ser melhor observado no Apêndice C deste trabalho. Após esta etapa estar concluída, avançou-se para a finalização do estudo, apresentando estes dados ao usuário por meio de uma interface.

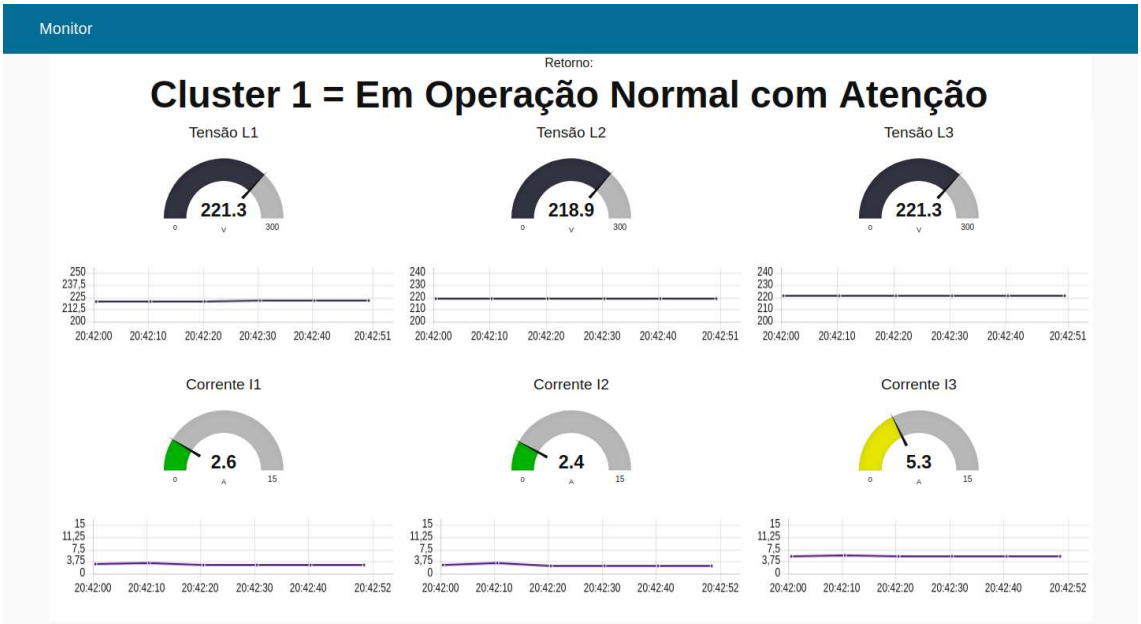
5.4 INTERFACE COM DADOS EM TEMPO REAL

Conforme proposto no escopo deste estudo, o retorno dos valores calculados pela inteligência se dará por meio de uma interface para com o usuário do sistema. Para isto, necessitou-se criar na plataforma *Azure*® um experimento preditivo que fosse capaz de receber e enviar os valores diretamente para um *web server.* A própria plataforma permite a criação deste servidor *online* onde a comunicação com os *hardwares* pode ser feita por meio de uma API.

A API é uma forma ágil e segura de integrar sistemas quando se necessita do intercâmbio de informações entre diferentes plataformas e linguagens de programação. As estruturas das mensagens de solicitação e de resposta são feitas por meio de protocolo HTTP (*Hypertext Transfer Protocol* – do inglês Protocolo de Transferência de Hipertexto), onde as mensagens são fornecidas em formato JSON para ambas as aplicações. (FERNANDES, 2018). O Apêndice D representa o corpo da mensagem a ser enviada e recebida pelas plataformas *Azure*® e *Node-Red*. Logo, por meio das especificações da API fornecida pela própria plataforma *Azure*®, possibilitou-se enviar o conjunto de dados fornecidos pela plataforma *Node-Red* e após, também receber a resposta do Aprendizado de Máquina.

Com o retorno da classificação dos dados por *cluster* e com os valores medidos pelo MGE, elaborou-se a interface para o usuário também através da própria plataforma *Node-Red*, a qual possibilita acrescentar indicadores de valores e gráficos através de um *dashboard*, concentrando todas as informações na mesma tela. O resultado desta implementação pode ser observado na Figura 12.

Figura 12 – Interface com o usuário



Fonte: Elaborada pela autora.

Através da Figura 12 percebe-se que a primeira informação fornecida ao usuário será o status da operação do sistema. Abaixo estão representados os valores das tensões das três fases fornecidas pelo medidor, seguindo também pelos valores das três correntes elétricas. Também é disposto ao usuário o gráfico desde valores com relação ao horário da leitura. Além disso, os indicadores de corrente elétrica possuem cores que correspondem aos limites dos valores lidos

e esta legenda pode ser verificada conforme Quadro 6, assim como a legenda que acompanha a resposta da inteligência apresentada no topo da tela.

Quadro 6 – Legenda dos indicadores da interface

	***Cluster* 0**	***Cluster* 1**	***Cluster* 2**	***Cluster* 3**
Mensagem	Em Operação Normal	Em Operação Normal com Atenção	Desligado	Anormal: Investigar Possível Falha Futura
Corrente (A)	**Verde**	**Amarelo**	**-**	**Vermelho**
	0,1 - 3,0	3,1 - 7,0		7,1 - 15,0

Fonte: Elaborada pela autora.

Com os valores e indicações definidos pelo Quadro 6 e através dos testes realizados no laboratório, constatou-se que o sistema como um todo é funcional e atende a metodologia proposta neste estudo. Assim, por se tratar de um sistema emulado em um ambiente diferente de uma máquina, é possível garantir que ao ser aplicado à mesma, o comportamento obtido será condizente aos testes aqui realizados.

6 CONCLUSÃO

O presente trabalho valida que a tecnologia elencada pelo estudo é viável para realizar as ações necessárias visando à aplicação do mesmo na prática, trazendo consigo a melhoria dos processos de interação entre o usuário e a máquina e ampliando as possibilidades de crescimento e competitividade da empresa aliado ao uso de materiais de baixo custo. Coleta de dados produtivos, previsões de falha e auxílio na antecipação de manutenções preventivas baseadas nas informações obtidas são alguns dos pontos que podem ser considerados de grande valor para o aumento do desempenho dos equipamentos industriais.

A consulta dos dados em tempo real somado ao uso de inteligências para tratamento e manipulação das informações geradas no ambiente industrial, é o grande diferencial na atualidade com a presença cada vez mais forte da tecnologia no chão de fábrica. Desta forma, este trabalho buscou incorporar este paradigma de uma maneira simples e eficaz, trazendo os benefícios necessários para os usuários e empresas que não possuem condições de investir em grandes sistemas e aplicações que estão disponíveis no mercado.

O sistema proposto através do estudo feito, se torna uma valiosa ferramenta para auxiliar na transformação de dados em informações, visando o Aprendizado de Máquina e o auxílio à tomada de decisão, solucionando de forma inteligente a manutenção preditiva e possibilitará a inserção da manufatura de uma empresa diretamente no contexto de Indústria 4.0. Com isto, o principal resultado obtido foi integrar algumas das tecnologias disponíveis de baixo custo em um sistema capaz de entregar um resultado claro e específico ao usuário, permitindo a tomada de decisões de forma mais eficaz. Obtendo os dados classificados entre os quatro *clusters* a partir da técnica de *K-Means*, o sistema se torna capaz de informar ao usuário quando o sistema está desligado, em operação normal, em operação normal no limite de operação e quando apresenta anormalidade, evidenciando que uma falha pode vir a acontecer no futuro se o sistema permanecer neste estado por mais tempo.

O estudo realizado neste trabalho abre precedentes para análises mais aprofundadas de Inteligência Artificial, assim como melhorias que poderiam ser obtidas com a implantação do mesmo em alguma máquina. Com isto, ficam como sugestões de trabalhos futuros os seguintes tópicos:

- Aplicar o sistema desenvolvido em uma máquina observando as devidas alterações de *hardware* para leitura de diferentes sensores ou de dados provenientes de um CLP, aprimorando o treinamento da Aprendizagem de Máquina para outras intemperes do sistema além da corrente elétrica;

- Realizar um aprimoramento do *software* da Inteligência Artificial aplicando técnicas de rede neural para aprender com as experiências providas dos *clusters* e predizer quando, em um tempo preciso, haverá a falha;
- Aprimorar o *software* da Inteligência Artificial para tomar decisões, gerar ações autônomas, além de análises completas e detalhadas do processo e da produtividade, gerando valor para o aumento do desempenho dos equipamentos industriais;
- Realizar a implantação deste trabalho em alguma empresa, afim de obter o retorno sobre a funcionalidade prática da aplicação, além dos custos envolvidos para implantação e verificar quais os resultados que o sistema poderá trazer a empresa.

REFERÊNCIAS

ALTUS. **Manual de Utilização Série Phase**. Revisão E. São Leopoldo, 2015.

AMRUTHNATH, Nagdev; GRUPTA, Tarun. **A Research Study on Unsupervised Machine Learning Algorithms for Early Fault Detection in Predictive Maintenance**. 5th International Conference on Industrial Engineering and Applications, 2018.

ARTHUR, David; VASSILVITSKII, Sergei. **K-Means++: The advantages of careful seeding.** SODA, 2007.

BARROS, Edna; CAVALCANTE, Sérgio. **Introdução aos sistemas embarcados.** Recife: Universidade Federal de Pernambuco, 2005.

BORGES, Rodrigo W.; RODRIGUES, Evandro L. L. **Embedded System Desing: An Overview of Brazilian Development.** 9th IEEE International Symposium on Parallel and Distributed Processing with Applications Workshops, 2011.

BUTTE, Sujata; PRASHANTH, A. R.; PATIL, Sainath. **Machine Learning Based Predictive Maintenance Strategy: A Super Learning Approach With Deep Neural Networks**. IEEE Workshop on Microelectronics and Electron Devices (WMED), 2018.

CLINE, Brad; NICULESCU, Radu S.; HUFFMAN, Duane; DECKEL, Bob. **Predictive Maintenance Applications for Machine Learning**. Annual Reliability and Maintainability Symposium (RAMS), 2017.

COPPIN, Ben. **Inteligência artificial.** Rio de Janeiro: LTC, 2010.

CYRINO, Luis. Desperdícios e Redução de Custos de Manutenção. 20 mai 2018. Disponível em: https://www.manutencaoemfoco.com.br/desperdicios-e-reducao-custos-manutencao/. Acesso em: 11 jun. 2019.

DJIEV, S. **Industrial Networks for Communication and Control**. Reading for Elements for Industrial Automation, Technical University, Sofia, Bulgaria, 2003.

DOLUI, Koustabh; DATA, Soumya K. **Comparison of Edge Computing Implementations: Fog Computing, Cloudlet and Mobile Edge Computing**. IEEE Global Internet of Things Summit (GIoTS), 2017.

FERNANDES, André. O que é API? Entenda de uma maneira simples. 01 mar 2018. Disponível em: https://vertigo.com.br/o-que-e-api-entenda-de-uma-maneira-simples/. Acesso em: 31 out. 2019.

GOLDSCHMIDT, Ronaldo R. **Inteligência computacional**. Rio de Janeiro: IST-Rio, 2010.

HAUGEN, Oystein; MOLLER-PEDERSEN, Birger; WEIGERT, Thomas. Introduction to UML and the Modeling of Embedded Systems. In ZURAWSKI, Richard. **Embedded Systems Handbook.** CRC press, 2005. p. 268-300.

KANAWADAY, Ameeth; SANE, Aditya. **Machine Learning for Predictive Maintenance of Industrial Machines using IoT Sensor Data**. 8th IEEE International Conference on Software Engineering and Service Science (ICSESS), 2017.

KAPITONOVA, J.V; LETICHEVSKY, A. A; VOLKOV, V. A. System Validation. In ZURAWSKI, Richard. **Embedded Systems Handbook.** CRC press, 2005. p. 168-224.

LAVAGNO, Luciano; PASSERONE, Claudio. Design of Embedded Systems. In ZURAWSKI, Richard. **Embedded Systems Handbook.** CRC press, 2005. p. 85-108.

MAHALIK, Nitaigour P. **Fieldbus technology: industrial network standards for real-time distributed control.** 1. ed. Sambalpur: Burla, 2003.

MAŘÍK, Vladimir; FLETCHER, Martyn; PECHOUCEK, Michal. **Holons and agents: Recent development and mutual impacts.** In Multi-Agent Systems and Applications II, 2001.

MICROSOFT. Azure Machine Learning Studio. Disponível em: https://azure.microsoft.com/pt-br/services/machine-learning-studio/. Acesso em: 29 out. 2019.

MITCHELL, Thomas. **Machine learning**. 1. ed. Mcgraw-Hill, 1997.

MONKEY, Stephen. Using Embedded Systems in Industrial Automation Applications. 20 set 2018. Disponível em: https://www.digitroniklabs.com/blog/using-embedded-systems-in-industrial-automation-applications/. Acesso em: 08 mai. 2019.

MORAES, Lucas. Finep lança linha de crédito para financiar tecnologia 4.0 em pequenas e medias empresas. **Jornal do Commercio**, Recife, 11 jun. 2019. Disponível em: https://jconline.ne10.uol.com.br/canal/economia/nacional/noticia/2019/06/11/finep-lanca-linda-de-credito-para-financiar-tecnologia-40-em-pequenas-e-medias-empresas-380827.php. Acesso em: 11 jun. 2019.

NEUMANN, Peter. **Industrial communication: What is going on?** Barleben: Institut f. Automation und Kommunikation Magdeburg, 2005.

OLIVEIRA, Bruno. Análise de componentes principais. 11 out 2019. Disponível em: https:// https://operdata.com.br/blog/analise-de-componentes-principais/. Acesso em: 29 out. 2019.

PEREIRA, Eduardo Carlos; CARRO, Luigi. **Distributed Real-Time Embedded Systems: Recent Advances, Future Trends and their Impact on Manufacturing Plant Control.** Porto Alegre: Universidade Federal do Rio Grande do Sul, 2006.

RESENDE, Ubiratan. Ganhos e demandas da Indústria 4.0 com a computação de borda. 01 mar 2019. Disponível em: https://docmanagement.com.br/03/01/2019/ganhos-e-demandas-da-industria-4-0-com-a-computacao-de-borda/. Acesso em: 11 mai. 2019.

RODRIGUES, Evando; PEDÓ, Ricardo; TEDESCO, Leonel Pablo. **Sistemas Embarcados e sua Aplicação na Indústria.** Santa Cruz do Sul: Universidade de Santa Cruz do Sul, 2013.

RUSSEL, Stuart; NORVIG, Peter. **Inteligência Artificial**. Rio de Janeiro: Elsevier, 2013.

SARKAR, Tirthajyoti. Clustering metrics better than the elbow-method. 06 set 2019. Disponível em: https://towardsdatascience.com/clustering-metrics-better-than-the-elbow-method-6926e1f723a6. Acesso em: 29 out. 2019.

SEZER, Erim; ROMERO, David; GUEDEA, Federico; MACCHI, Marco; EMMANOUILIDIS, Christos. **An Industry 4.0-enabled Low Cost Predictive Maintenance Approach for SMEs: A Use Case Applied to a CNC Turning Center**. IEEE International Conference on Engineering, Technology and Innovation, 2018.

SHI, Weisong; CAO, Jie; ZHANG, Quan; LI, Youhuizi; XU, Lanyu. **Edge Computing: Vision and Challenges**. IEEE Internet of Things Journal, 2016.

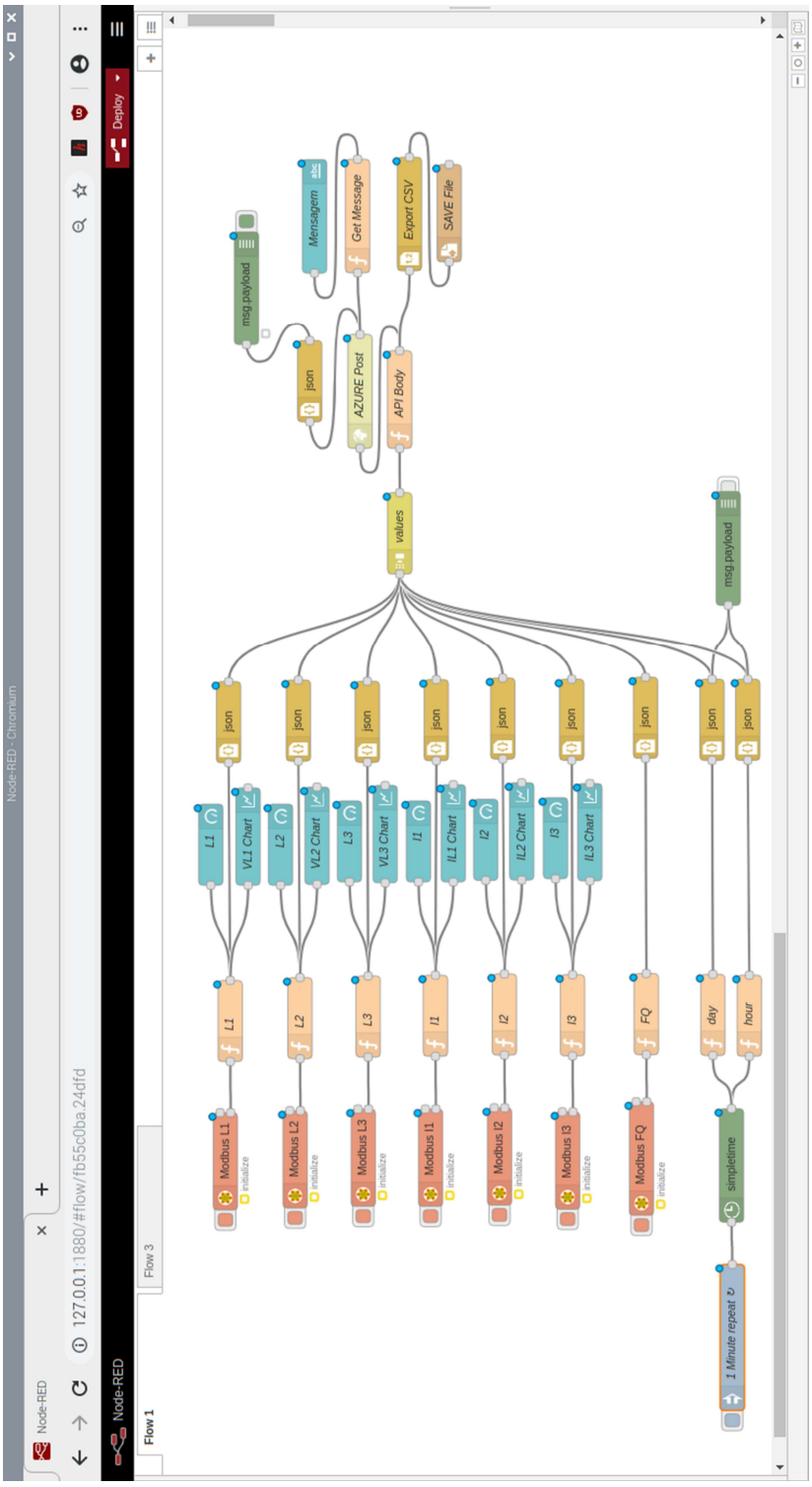
STRAUSS, Patrick; SCHMITZ, Markus; WÖSTMANN, René; DEUSE, Jochen. **Enabling of Predictive Maintenance in the Brownfield through Low-Cost Sensors, an IIoT-Architecture and Machine Learning**. IEEE International Conference on Big Data (Big Data), 2018.

WU, Junjie. **Advances in K-means Clustering: a Data Mining Thinking.** Springer-Verlag Berlin Heidelberg, 2012.

YAN, Weizhong. **One-Class Extreme Learning Machines for Gas Turbine Combustor Anomaly Detection**. International Joint Conference on Neural Networks (IJCNN), 2016.

APÊNDICE A – FLUXO *NODE-RED*

Imagem do fluxo desenvolvido na plataforma *Node-Red* para coleta, envio dos dados e recebimento da resposta de inteligência.



Fonte: Elaborada pela autora.

APÊNDICE B – *SCRIPT* EM PYTHON

O *script* programado em linguagem Python para escolher do conjunto de dados a ser empregado a plataforma *Azure® Machine Learning Studio*.

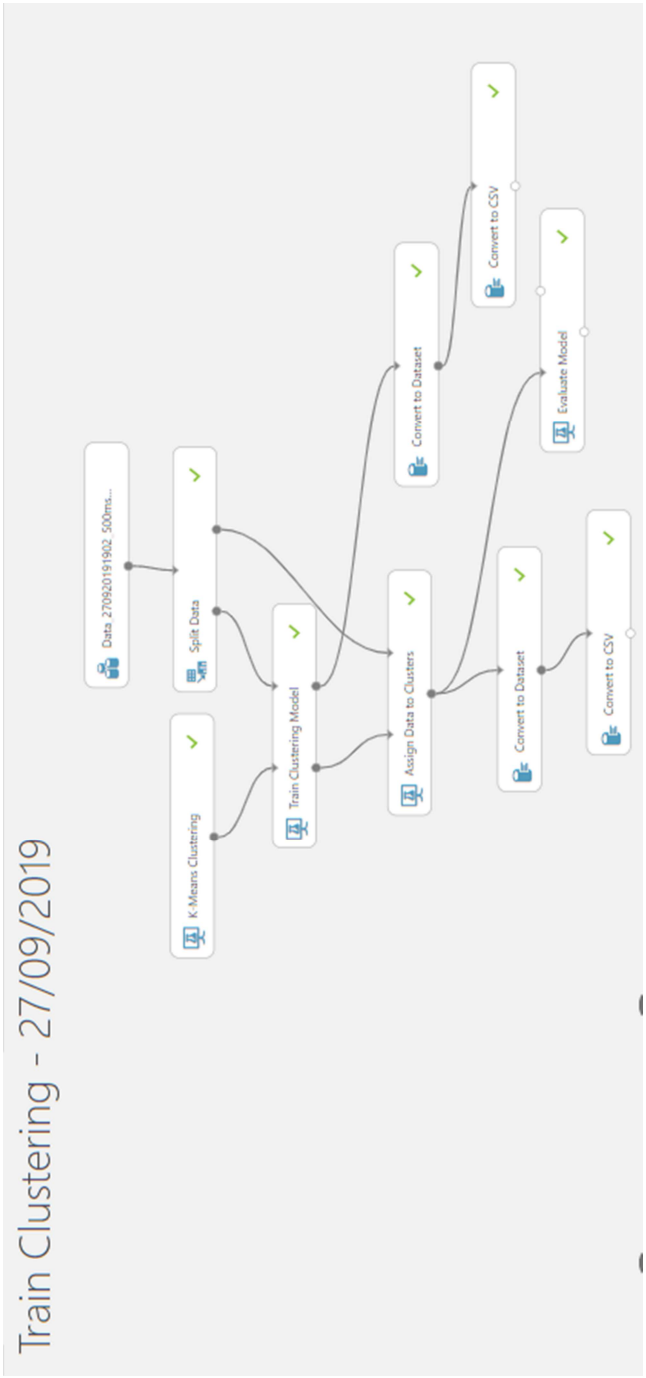
```
import pandas as pd
import numpy as np
import collections
from sklearn.cluster import KMeans
from sklearn.decomposition import PCA
from sklearn.metrics import silhouette_samples
import matplotlib.pyplot as plt
from matplotlib import cm

df_train = pd.read_csv("dataset_3.csv", header=None, names=['a','b','c'])

print(df_train)

# reduce to 2 dimensions
pca = PCA(n_components = 2, random_state=1)
X_pca = pca.fit_transform(df_train)

print('Explained Variance Ratio : ' + str(pca.explained_variance_ratio_.cumsum()[1
]))

distortions = []
K_to_try = range(1, 6)

for i in K_to_try:
    model = KMeans(
            n_clusters=i,
            init='k-means++',
            # n_init=10,
            # max_iter=300,
            # n_jobs=-1,
            random_state=1)
    model.fit(X_pca)
    distortions.append(model.inertia_)

plt.plot(K_to_try, distortions, marker='o')
plt.xlabel('Número de Clusters (k)')
plt.ylabel('Distorção')
plt.show()

model = KMeans(
    n_clusters=4,
    init='k-means++',
    # n_init=10,
```

```
    # max_iter=300,
    # n_jobs=-1,
    random_state=1)

model = model.fit(X_pca)

y = model.predict(X_pca)

plt.scatter(X_pca[y == 0, 0], X_pca[y == 0, 1], s = 50, c = 'yellow', label = 'Clu
ster 1')
plt.scatter(X_pca[y == 1, 0], X_pca[y == 1, 1], s = 50, c = 'green', label = 'Clus
ter 2')
plt.scatter(X_pca[y == 2, 0], X_pca[y == 2, 1], s = 50, c = 'red', label = 'Cluste
r 3')
plt.scatter(X_pca[y == 3, 0], X_pca[y == 3, 1], s = 50, c = 'grey', label = 'Clust
er 4')
plt.scatter(model.cluster_centers_[:, 0], model.cluster_centers_[:, 1], s = 100, c
 = 'blue', label = 'Centroide')
plt.title('Clusters')
plt.xlabel('Média entre Correntes 1')
plt.ylabel('Média entre Correntes 2')
plt.legend()
plt.grid()
plt.show()

print('K Means Result : ')
print(collections.Counter(y))
```

Fonte: Elaborado pela autora.

APÊNDICE C – FLUXO *AZURE MACHINE LEARNING STUDIO*

Imagem do fluxo de Aprendizado de Máquina desenvolvido na plataforma *Azure Machine Learning Studio*.



Fonte: Elaborado pela autora.

APÊNDICE D – CORPO DA API

Abaixo está representado o corpo das mensagens de envio e retorno dos valores entre as plataformas por meio da API criada.

```
REQUEST:
{
  "Inputs": {
    "input1": {
      "ColumnNames": [
        "Col1",
        "Col2",
        "Col3",
        "Col4",
        "Col5",
        "Col6",
        "Col7",
        "Col8",
        "Col9"
      ],
      "Values": [
        [
          "0",
          "0",
          "0",
          "0",
          "0",
          "0",
          "0",
          "value",
          ""
        ],
        [
          "0",
          "0",
          "0",
          "0",
          "0",
          "0",
          "0",
          "value",
          ""
        ]
      ]
    }
  },
  "GlobalParameters": {}
}

RESPONSE:
{
  "Results": {
    "output1": {
      "type": "DataTable",
      "value": {
        "ColumnNames": [
          "Col1",
```

```
            "Col2",
            "Col3",
            "Col4",
            "Col5",
            "Col6",
            "Col7",
            "Assignments",
            "DistancesToClusterCenter no.0",
            "DistancesToClusterCenter no.1",
            "DistancesToClusterCenter no.2",
            "DistancesToClusterCenter no.3"
          ],
          "ColumnTypes": [
            "Numeric",
            "Numeric",
            "Numeric",
            "Numeric",
            "Numeric",
            "Numeric",
            "Numeric",
            "Categorical",
            "Numeric",
            "Numeric",
            "Numeric",
            "Numeric"
          ],
          "Values": [
            [
              "0",
              "0",
              "0",
              "0",
              "0",
              "0",
              "0",
              "0",
              "0",
              "0",
              "0",
              "0"
            ],
            [
              "0",
              "0",
              "0",
              "0",
              "0",
              "0",
              "0",
              "0",
              "0",
              "0",
              "0",
              "0"
            ]
          ]
        }
      }
    }
  }
```


Fonte: Elaborada pela autora.