INSTITUTO FEDERAL DO ESPÍRITO SANTO

ENGENHARIA DE CONTROLE E AUTOMAÇÃO

MARIANA SPADETTO LEÃO

DETECÇÃO DE ANOMALIAS NO SISTEMA APS DE CAMINHÕES DE CARGA UTILIZANDO ONE-CLASS SVM

SERRA

2019

MARIANA SPADETTO LEÃO

DETECÇÃO DE ANOMALIAS NO SISTEMA APS DE CAMINHÕES DE CARGA UTILIZANDO ONE-CLASS SVM

Trabalho de Conclusão de Curso apresentado à Coordenadoria do Curso de Engenharia de Controle e Automação do Instituto Federal do Espírito Santo como requisito parcial para a obtenção do título de Bacharel em Engenharia de Controle e Automação.

Orientador: Prof. Dr. Daniel Cruz Cavalieri

SERRA

2019

AGRADECIMENTOS.

Agradeço aos meus pais, Eluzinete e Wilson, que tanto me apoiaram e nunca desistiram de mim. Agradeço ao meu irmão, André, que desde pequeno sempre esteve ao meu lado e mesmo nas horas mais difíceis me deu suporte e me ajudou. Obrigada por serem o meu porto seguro.

Agradeço aos meus melhores amigos e namorado, Aristides, Carolina e Thiago, que me acolheram e fizeram com que os momentos fossem mais leves e felizes. Obrigada pelos sorrisos e pelas lições de vida.

Agradeço aos meus professores por todos os ensinamentos e conhecimentos compartilhados, pois este momento só é real devido a vocês. Um obrigado especial ao meu querido orientador, Daniel Cavalieri, que acreditou em mim desde o começo e confiou no meu potencial.

Agradeço ao Ifes por ser um excelente instituto de aprendizado e um lugar que me proporcionou muitas amizades, conhecimento e crescimento. Muito obrigada!

“No meio de toda dificuldade, encontra-se a oportunidade.”

Albert Einstein

RESUMO

Controlar custos de manutenção é uma medida que exige constantes melhorias em indústrias e empresas. Uma forma interessante de se controlar a periodicidade da manutenção em um equipamento ou máquina é fazendo a detecção de pontos incomuns nestes processos antes mesmo que estes possam acontecer. Estes pontos incomuns são chamados *outliers* ou anomalias. A abordagem deste projeto refere-se a um algoritmo de detecção de *outliers* utilizando a técnica de aprendizado de máquina não supervisionado denominada *One-class* SVM. Os dados utilizados neste projeto são referentes a falhas operacionais de caminhões de carga da marca Scania, onde o algoritmo de detecção deve prever se a falha será em um componente do sistema APS (*outliers*) ou não (*inliers*), para fins de minimização dos custos de manutenção. Com a metodologia utilizada foi possível alcançar um custo de manutenção de $15.290, detectando ainda 99% dos outliers. Este valor representa uma economia de mais de $140.000 para a empresa. O projeto apresentará a metodologia de pré-processamento dos dados e de como foi desenvolvido o algoritmo, bem como outros resultados obtidos durante o desenvolvimento, sendo possível ainda encontrar 100% de acerto de *outliers* em um dos resultados apresentados.

Palavras-chave: Machine Learning. One-class SVM. Python. Clustering. Detecção de anomalias.

ABSTRACT

Controlling maintenance costs is a measure that requires constant improvements in industries and companies. An interesting way to control the periodicity of maintenance on an equipment or machine is by detecting unusual points in these processes even before they happen. These unusual points are known as outliers or anomalies. The approach of this project refers to an outlier detection algorithm using the unsupervised machine learning technique called One-class SVM. The data used in this project refer to operational failures of heavy trucks from the company Scania, where the detection algorithm must predict whether the failure will be in an APS system component (outliers) or not (inliers), with the purpose of minimization of maintenance costs. The methodology used in this project, it was possible to reach a maintenance cost of $ 15,290, being able to detect 99% of outliers. This represents an economy reduction of over $ 140,000 for the company. The project will present the data preprocessing methodology and how the algorithm was developed, as well as other results obtained during the development, being possible to find 100% of outliers in one of the presented results.

Keywords: Machine Learning. One-class SVM. Python. Clustering. Outlier detection.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS, SIGLAS

APS – *Air Pressure System*

IA – *Inteligência Artificial*

IDA – *Intelligent Data Analysis*

IoT – *Internet of Things*

FN – Falso negativo

FP – Falso positivo

LOF – *Local Outlier Factor*

NSERC - *Natural Sciences and Engineering Research Council of Canada*

RBF – *Radial Basis Function*

SMOTE – *Synthetic Minority Over-sampling Technique*

SVD – *Singular Value Decomposition*

SVDD – *Support Vector Data Description*

SVM – *Support Vector Machines*

TVP – Taxa de verdadeiro positivo

TFP – Taxa de falso positivo

VN – Verdadeiro negativo

VP – Verdadeiro positivo

SUMÁRIO

1 INTRODUÇÃO

O cenário atual da indústria está crescentemente se modernizando e implantando projetos de inovação para soluções de problemas. Com o advento da Indústria 4.0, os processos de produção tendem a se tornar cada vez mais eficientes, autônomos e customizáveis (SILVEIRA, 201-). Então, para capturar tudo que as tecnologias digitais podem oferecer em termos de aumento da confiabilidade e redução de custos, as empresas devem ampliar suas ambições de manutenção digital (BADBURY *et al*., 2018).

O termo Indústria 4.0 foi usado pela primeira vez na Feira de Hannover, Alemanha, em 2011, mas somente em 2013 foi publicado um trabalho sobre o desenvolvimento da Indústria 4.0. (SILVEIRA, 201-). De acordo com Silveira (201-), as empresas poderão criar redes inteligentes ao longo de toda a cadeia de valor que podem controlar os módulos da produção de forma autônoma, se máquinas, sistemas e ativos estiverem conectados. Ou seja, as fábricas inteligentes terão a capacidade e autonomia para agendar manutenções, prever falhas nos processos e se adaptar aos requisitos e mudanças não planejadas na produção.

Um dos efeitos resultantes da Indústria 4.0 faz com que a Inteligência Artificial (IA) tenha uma participação cada vez maior neste fenômeno (O POTENCIAL..., 201-). O site explica que a IA funciona por meio da integração de vários fatores, como a utilização de sensores, *Smart Data*, *Internet of Things* (IoT), *Cloud Computing* e outras tecnologias. O fato de a tecnologia ter a capacidade de se auto otimizar, sem a necessidade do intermédio de uma pessoa é uma das principais vantagens da IA, já que o sistema se alimenta de suas próprias experiências, identificando as práticas mais produtivas e tornando o processo mais eficaz (O POTENCIAL..., 201-). Este é o conceito de aprendizado de máquina.

O aprendizado de máquina pode ser utilizado em inúmeras funções, como detecção de fraude, análise de *streaming* de dados, manutenção preditiva e inclusive para detecção de anomalias (17 CASOS..., 2018), também conhecido como *outliers*, que é o escopo deste trabalho.

Neste contexto, para demonstrar uma das diversas aplicações de aprendizado de máquina em cenários reais, foi utilizado uma base de dados fornecido pela marca sueca Scania, conhecida por ser um fabricante de caminhões de carga, ônibus, entre outros. Os dados desta base mostram informações relacionadas a falhas operacionais de caminhões, os quais são divididos entre falhas no sistema APS (*Air Pressure Sistem*) do caminhão e outras falhas (DUA; GRAFF, 2019).

A base de dados traz a informação que permite distinguir os dados entre positivos e negativos, porém foi convencionado que a distinção será da seguinte forma:

- Classe majoritária – são todos os dados de falhas que não ocorreram em algum componente do sistema de APS.
- Classe minoritária – são todos os dados de falhas referentes a algum componente do sistema de APS;

Assim, este trabalho visa desenvolver um algoritmo com base em aprendizado de máquina não supervisionado para realizar a detecção das falhas referentes a componentes do sistema APS, as quais são definidas pela classe minoritária.

O modelo de otimização escolhido para ser utilizado no algoritmo foi o *One-Class Support Vector Machines* (referido neste texto como *One-Class SVM*). Este método é especificamente utilizado para a detecção de *outliers* (anomalias) e detecção de *novelties* (novidades), isso significa que o interesse e objetivo do algoritmo são captar anormalidades e observações incomuns.

1.1 JUSTIFICATIVA

O aprendizado de máquina pode ser utilizado em uma empresa para auxiliar a tomada de decisão quanto a um investimento ou para classificar o perfil de compra de um cliente, por exemplo, mas também pode ser uma ferramenta bastante inovadora solucionar problemas específicos de manutenção preditiva (BADBURY *et al*., 2018).

A aplicação apresentada neste trabalho aborda justamente como o aprendizado de máquina auxilia em previsões de manutenção e na otimização da qualidade do produto. O sistema APS do caminhão é um sistema de gerenciamento de ar comprimido que atua em sistemas essenciais do caminhão, como nos freios e engrenagens. É de grande importância que este sistema esteja sempre em boas condições e em dia com as manutenções, já que estes veículos devem lidar diariamente com centenas de quilômetros e estão suscetíveis a falhas enquanto estão sendo operados.

Portanto, a justificativa deste trabalho é otimizar a predição de falhas nos caminhões e minimizar gastos de manutenção. Juntamente com esta base de dados, foi disponibilizado também o custo de manutenção das falhas que ocorrem no sistema APS de caminhões de carga e também o custo de manutenção para uma falha não prevista. Os custos de manutenção são os seguintes:

- Custo 1 = $10. Este é o custo referente a uma vistoria desnecessária feita no caminhão em uma oficina.
- Custo 2 = $500. Este é o custo de uma manutenção não prevista, pois ocorrerá uma falha inesperada no caminhão.

A partir destes valores, é possível observar que o caminhão passar por uma vistoria e não apresentar nenhum defeito é menos custoso do que o caminhão não passar por uma vistoria e falhar repentinamente. Assim, neste trabalho será feita uma análise dos custos de manutenção dos caminhões de acordo com os resultados obtidos pelo modelo de predição.

1.2 OBJETIVO

De acordo com a contextualização apresentada nas seções anteriores, nesta seção serão apresentados os objetivos gerais e específicos do presente trabalho.

1.2.1 Objetivo geral

Desenvolver um algoritmo baseado em aprendizado de máquinas para a detecção de *outliers* no sistema APS de caminhões de carga da empresa Scania.

1.2.2 Objetivos específicos

Para se alcançar o objetivo geral, foram utilizadas as seguintes etapas intermediárias:

- Coletar dados sobre falhas no sistema de APS em caminhões de carga da Scania fornecidos para o desafio do IDA (*Intelligent Data Analysis*), disponibilizado no repositório da UCI (DUA; GRAFF, 2019);
- Realizar a mineração dos dados coletados a fim de caracterizar e reduzir os atributos, bem como otimizar o algoritmo;
- Analisar a performance do estimador *One-Class Support Vector Machine*;
- Avaliar os resultados do estimador calculando o custo de manutenção dos caminhões.

1.3 TRABALHOS CORRELATOS

Em 2016, foi publicado um desafio para um simpósio internacional organizado pelo IDA (*Intelligent Data Analysis*) sobre a utilização de técnicas de *machine learning* para a predição de anomalias (INDUSTRIAL..., 2016). Este desafio consistia no desenvolvimento e apresentação de mecanismos de predição utilizando aprendizado de máquina aplicados a casos reais. Para a realização deste desafio,

a fabricante de caminhões de carga, Scania, forneceu um vasto banco de dados com informações de falhas operacionais em caminhões.

Neste desafio foram avaliados dois quesitos:

- O modelo de previsão que apresenta a melhor performance, ou seja, o modelo que alcança o valor mais baixo de manutenção; e
- A abordagem mais inovadora do assunto.

Neste contexto, vários estudiosos e entusiastas no assunto desenvolveram diferentes soluções para esta aplicação. Assim, com a intenção de enriquecer este trabalho, nesta seção serão abordados alguns trabalhos desenvolvidos para este desafio, pois podem servir de complementação e comparação.

Cerqueira *et al.* (2016) decidiram utilizar as técnicas de *Gradient Boost* e *Random Forest*, podendo ou não serem aliados a "*Metafeature Engineering*". O "*Metafeature Engineering*" consiste em analisar os *outliers* do banco de dados de forma a avaliar o quão distante estes pontos estão do restante dos dados. Para fazer isso, os autores utilizaram três diferentes técnicas de detecção de *outliers*: o Boxplot Analysis, que descreve a distribuição dos dados por meio de alguns princípios básicos de estatística, como mediana; o LOF (*Local Outlier Factor*), que é um método para quantificar a periferia de uma observação (ou seja, a probabilidade desta observação ser um *outlier*) a partir de comparações com a vizinhança local; e por fim, o *Clustering-Based Outlier Ranking*, que separa as observações em grupos pela similaridade até que tenha apenas um grupo contendo todas as observações.

Para o pré-processamento dos dados, os autores decidiram fazer a remoção das amostras de dados faltosos nos atributos com mais de 80% de informações vazias (*NaN*). Também para a questão do desbalanceamento da base de dados, utilizaram *SMOTE* (*Synthetic Minority Over-sampling Technique*), a qual é uma técnica que realiza o "*over-sampling*", criando exemplos sintéticos na classe minoritária para uma melhor generalização do banco de dados. Os melhores resultados obtidos por estes autores foi a técnica de *Gradient Boost* aliado a *metafeature engineering*, gerando um custo médio de $4560.

Já os autores Costa e Nascimento (2016), utilizaram as técnicas de *Logistic Regression*, KNN (*K-Nearest Neighbors*), *Support Vector Machines* (SVM), *Decision Trees* e *Random Forest*. Para os dados faltosos, os autores utilizaram uma técnica denominada "*Multiple Imputation*" e "*Expectation Maximization*", que insere dados estimados para que o algoritmo não seja prejudicado com perda de informação. Os autores chegaram à conclusão que o classificador com os melhores resultados para este caso, realizando o devido pré-processamento no banco de dados, foi o *Random Forest*. Este classificador apresentou uma porcentagem de 3,74% de FP (falso positivo) e 3,7% de FN (falso negativo), onde o erro de classificação custaria um total de $40570.

De outra maneira, os autores Gondek *et al.* (2016) optaram por testar quatro tipos de classificadores: *Naive Bayes*, *Multilayer* Perceptron (MLP), *Support Vector Machines* (SVM) e *Random Forest*. Porém, após alguns testes com os classificadores mencionados, decidiram utilizar somente o classificador *Random Forest*, pois este produziria os melhores resultados. Para lidar com dados faltosos, os autores decidiram substituir os valores vazios pela mediana dos atributos. Os autores conseguiram reduzir a média de custos que estava em $9,83 por caminhão para $0,60, o que totaliza $36.000 em custo de manutenção.

No trabalho de Rafsunjani *et al*. (2019), os autores visaram apresentar o quanto os resultados são influenciados pela forma com que se lida com os dados faltosos. Para provar isso, os autores examinaram cinco formas diferentes de lidar com estes dados. São eles: *Expectation Maximization*, que faz a imputação dos dados por meio de uma predição destes valores; *Mean Imputation*, este faz a imputação dos dados faltosos de um atributo pela média deste atributo; Iterative SVD (*Singular Value Decomposition*), este método utiliza o SVD para a obtenção de um conjunto de padrões para expressões mutualmente ortogonais, que podem ser linearmente combinadas para estimar os valores faltosos da base de dados; *Soft Imputation*, o qual calcula o limiar suave SVD de uma matriz por iterações; MICE (*Multiple Imputation by Chained Equation*), o qual cada variável é imputada utilizando um modelo próprio.

Ainda no trabalho de Rafsunjani *et al*. (2019), também foi feita uma comparação empírica da performance de cinco diferentes classificadores: *Naive Bayes*, KNN,

SVM, *Random Forest* e *Gradient Boost*. Os autores chegaram à conclusão de que os resultados do classificador *Random Forest* aliado a todos os métodos de imputação foram consistentemente melhores do que os outros classificadores. O método de imputação por *Soft Imputation* obteve os piores resultados, enquanto o método de imputação MICE foi o mais impactante para a obtenção de bons resultados. Os autores não divulgaram os valores de manutenção para estes resultados.

Oznan *et al.* (2016) desenvolveram um método de imputação de dados baseado em KNN para lidar com os dados faltosos. Em seguida, os autores definiram uma função de custo de manutenção e reduziram essa função utilizando uma abordagem baseada em gradiente descendente estocástico (GDE), que consiste em encontrar distâncias e pesos para a classificação KNN. Para fazer a validação do modelo de predição dos autores, foram feitos testes com outros três classificadores tradicionais: SVM, *Random Forest* e *AdaBoost*. Foram feitos cinco testes com estes classificadores e por fim, calculado a média de custo de manutenção de cada um para comparação. A média de custo de manutenção do modelo proposto pelos autores se destacou entre os outros classificadores, tendo um valor de $10.792.

1.4 ESTRUTURA DO TRABALHO

No capítulo 1 foram apresentadas a contextualização, justificativa e também foram traçados o objetivo geral e o os objetivos específicos. O capítulo 2 consiste na fundamentação teórica necessária para o desenvolvimento e entendimento do trabalho, onde foram abordados os temas de aprendizado de máquina e sua vertente aqui estudada: as *One-class SVM*. No capítulo 3 é mostrado como o trabalho foi desenvolvido, onde explica-se toda a metodologia aplicada para obter-se os resultados. O capítulo 4 apresenta os resultados obtidos pelo método *One-class SVM* aplicado a detecção de *outliers* no sistema de APS de caminhões de carga, bem como a análise de custos de manutenção com a aplicação desta técnica. Por fim, o capítulo 5 apresenta a avaliação dos resultados e a conclusão deste trabalho.

2 FUNDAMENTAÇÃO TEÓRICA

2.1 *MACHINE LEARNING*

Quando se fala em computação e tecnologia, o aprendizado de máquina (do inglês *machine learning*) pode ser definido como uma área de pesquisa que visa "desenvolver programas computacionais capazes de melhorar sua performance em alguma tarefa por meio da experiência" (MITCHELL, 1997, p. 2). Em outra literatura, explica-se este mesmo conceito de uma maneira diferente:

> *Machine learning* é fazer com que computadores modifiquem ou adaptem suas ações (sejam essas ações fazer predições ou controlar robôs) de forma que essas ações sejam mais precisas, onde a acurácia é medida por quão bem as ações escolhidas refletem nas ações corretas. (MARSLAND, 2015, p. 4)

Faceli *et al.* (2011) afirmam que os algoritmos de aprendizado de máquina são organizados em diferentes critérios para lidar com a realização de tarefas. Os autores dividiram estes critérios em tarefas preditivas e tarefas descritivas. De acordo com os autores, em tarefas de previsão a meta é encontrar uma função (modelo) a partir dos dados de treinamento que possa ser utilizada para prever um rótulo ou valor que caracterize um novo exemplo. Já as tarefas de descrição, a meta é explorar ou descrever um conjunto de dados.

Na literatura de Faceli *et al.* (2011) é explicado que os algoritmos utilizados nas tarefas preditivas seguem o paradigma de aprendizado supervisionado, pois nestes são utilizados atributos de entrada e saída. Em contrapartida, as tarefas descritivas seguem o paradigma de aprendizado não supervisionado, pois não fazem a utilização do atributo de saída. A figura 1 mostra uma hierarquia de aprendizado de acordo com os tipos de tarefas de aprendizado.

Figura 1 - Hierarquia de aprendizado.

Fonte: Faceli et al. (2011)

No topo está o aprendizado indutivo, processo pelo qual são realizadas as generalizações a partir dos dados. Em seguida, tem-se os tipos de aprendizado supervisionado (preditivo) e não supervisionado (descritivo). As tarefas supervisionadas se distinguem pelo tipo dos rótulos dos dados: discreto, no caso de classificação; e contínuo, no caso de regressão. As tarefas descritivas são genericamente divididas em: agrupamento, em que os dados são agrupados de acordo com a sua similaridade; sumarização, cujo objetivo é encontrar uma descrição simples e compacta para um conjunto de dados; e associação, que consiste em encontrar padrões frequentes de associações entre os atributos de um conjunto de dados.

Para explicar sobre o aprendizado não supervisionado, os autores Castro *et al.* definem que agrupamento, do inglês *clustering*, é o nome dado ao processo de separar (particionar ou segmentar) um conjunto de objetos em grupos (*clusters*) de objetos similares. De acordo com os autores:

> Diferentemente da tarefa de classificação, o agrupamento de dados considera dados de entrada não rotulados, ou seja, o grupo (classe) ao qual cada dado de entrada (objeto) pertence não é conhecido *a priori*. (CASTRO; FERRARI, 2017, capítulo 1.2.1)

“Diversas outras nomenclaturas possuem significado similar, como taxonomia numérica, análise de clusters, reconhecimento não supervisionado de padrões e

análise tipológica” (CASTRO; FERRARI, 2017, capítulo 4.1). Neste contexto, o presente trabalho envolverá somente a técnica de aprendizado não supervisionada de agrupamento.

2.1.1 Aprendizado não supervisionado

Marsland (2015) define o aprendizado não supervisionado da seguinte forma:

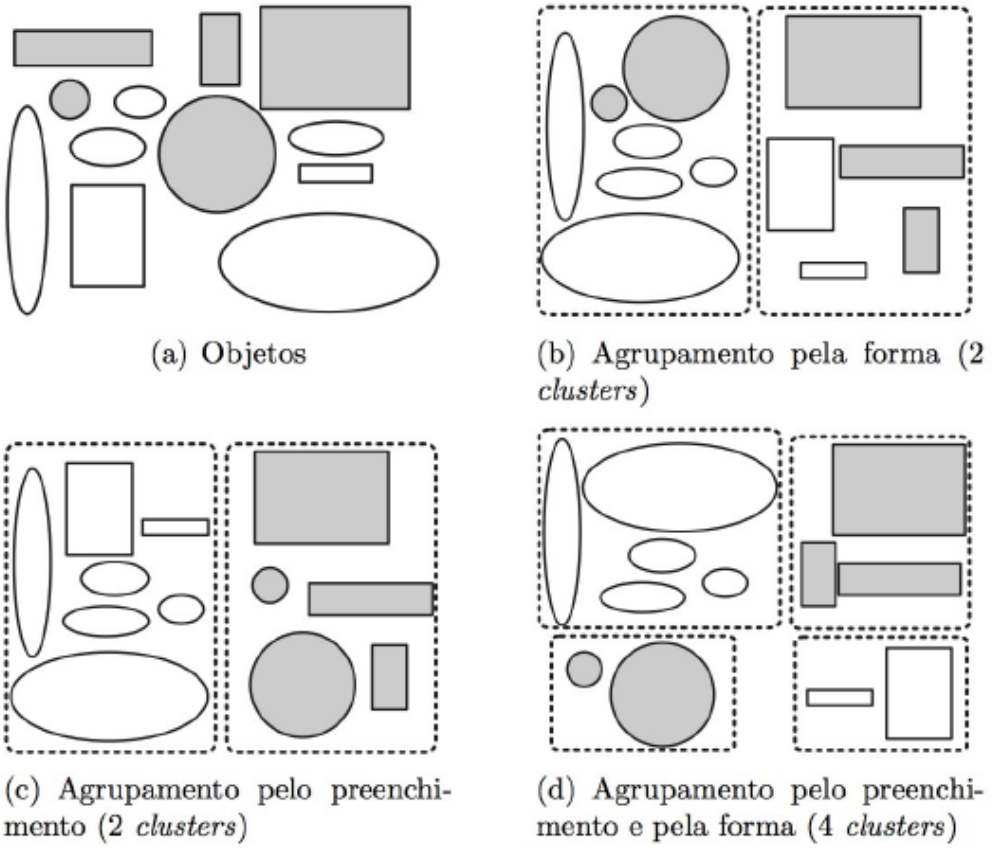
> As saídas corretas não são dadas, porém o algoritmo tenta identificar similaridades entre as entradas, para que as entradas que possuam algo em comum sejam então categorizadas. A abordagem estatística para aprendizado não supervisionado é conhecida como estimativa de densidade. (MARSLAND, 2015, p. 6)

Os autores Castro e Ferrari (2017) explicam que o processo de agrupamento (ou *clusterização*) é normalmente utilizado para identificar tais grupos e, portanto, cada grupo formado pode ser visto como uma classe de objetos. Como os rótulos das classes dos dados de treinamento não são conhecidos a priori, esse processo é denominado treinamento não supervisionado.

> Em um processo de agrupamento, os objetos são agrupados com o objetivo de maximizar a distância interclasse e minimizar a distância intraclasse, ou, dito de outra forma, maximizar a similaridade intraclasse e minimizar a similaridade interclasse. (CASTRO; FERRARI, 2017, capítulo 1.2.1)

Os autores Faceli et al. (2015) sintetizaram a ideia de agrupamento e clusters na figura 2:

Figura 2 - Demonstração de objetos (*clusters*) agrupados de diferentes maneiras.



Fonte: FACELI *et al*. (2015)

> O agrupamento de dados é uma técnica comum em análise de dados que é utilizada em diversas áreas, incluindo aprendizagem de máquina, mineração de dados, reconhecimento de padrões, análise de imagens e bioinformática. (CASTRO; FERRARI, 2017, capítulo 4.1)

Castro e Ferrari (2017) afirmam que intuitivamente, objetos que pertencem ao mesmo grupo são mais similares entre si do que a objetos que pertencem a outros grupos. Em outras palavras, “um *cluster* pode ser definido como uma coleção de objetos similares uns aos outros e dissimilares aos objetos pertencentes a outros *clusters*” (CASTRO; FERRARI, 2017, capítulo 1.2.1).

2.2 *SUPPORT VECTOR MACHINES*

De acordo com Huang *et al.* (2006), a técnica de *Support Vector Machines* (SVM) foi desenvolvida na ordem inversa das mais conhecidas redes neurais. Primeiramente, as SVMs partiram da teoria e depois evoluíram para a implementação e experimentos, enquanto as redes neurais seguiram um caminho mais heurístico, das aplicações e extensos experimentos para a teoria. Os autores também afirmam que as SVMs ficaram por muito tempo sem muita visibilidade, porém hoje tem-se melhores resultados que (ou até mesmo resultados comparáveis que) as redes neurais ou outros modelos estatísticos em solução de problemas.

De acordo com Awad e Khanna (2015) as SVMs constroem soluções em termos de um subconjunto de entrada de treinamento. São muito utilizadas para classificação, regressão, tarefas de detecção de novidade e também redução de atributos. Já Faceli *et al.* (2011) explicam que as SVMs são embasadas pela teoria de aprendizado estatístico, desenvolvida a partir de estudos iniciados com Vapnik (1995) e Chervonenkis (1971). Essa teoria estabelece uma série de princípios que devem ser seguidos na obtenção de classificadores com boa capacidade de generalização. Os autores argumentam que a teoria de aprendizado estatístico estabelece condições matemáticas que auxiliam na escolha de um classificador particular a partir de um conjunto de dados de treinamento. Essas condições levam em conta o desempenho do classificador no conjunto de treinamento e sua complexidade.

2.2.1 SVM lineares

“Classificadores que separam os dados por meio de um hiperplano são denominados lineares” (FACELI *et al.*, 2011, p. 126). Suponha que **X** seja um conjunto de treinamento com n objetos $x_i \in X$ e os seus respectivos rótulos sejam $y_i \in Y$, onde **X** constitui o espaço de entrada e $Y = \{-1, +1\}$ são as possíveis

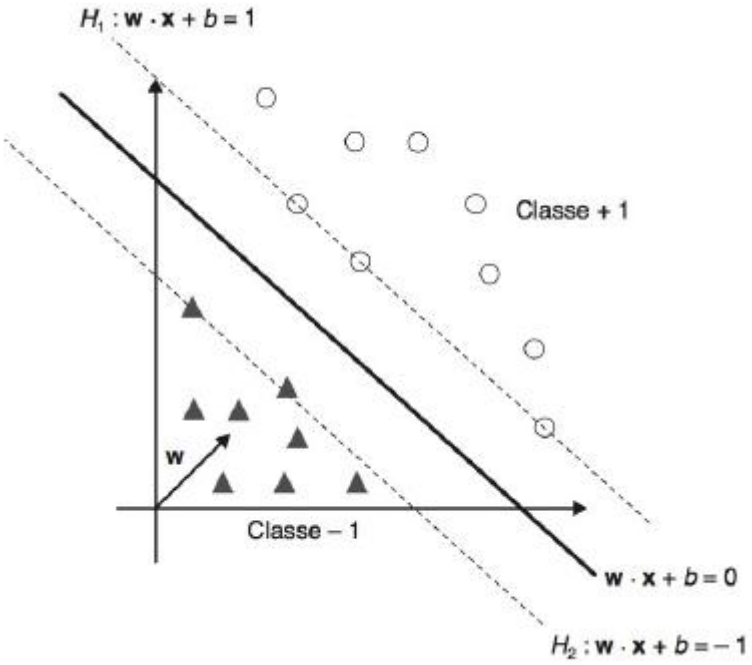
classes. Se for possível separar os objetos das classes $+1$ e -1 por um hiperplano, então **X** é linearmente separável, explicam os autores Faceli *et al.* (2011).

Neste sentido, a equação de um hiperplano, representada na equação 1, é o produto escalar entre os vetores w e x, onde $w \in X$ é o vetor normal do hiperplano descrito e $b \in R$.

$$h(x) = w \cdot x + b \quad (1)$$

De maneira resumida, esta equação pode ser usada para separar o espaço de entrada X em duas regiões: $w \cdot x + b > 0$ e $w \cdot x + b < 0$. Na figura 3 é possível visualizar a separação dois conjuntos distintos por um hiperplano linear.

Figura 3 - Ilustração da separação de classes por hiperplano.

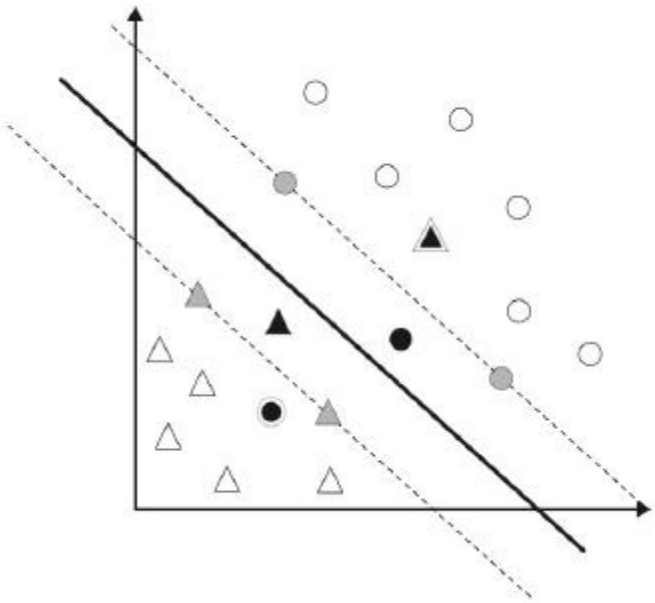


Fonte: Faceli et al. (2011)

A figura 3 caracteriza as SVM com margens rígidas, pois as restrições são impostas de maneira a assegurar que não haja dados de treinamento entre as margens de separação das classes (FACELI *et al*., 2011, p.126). Porém, em situações reais não é comum encontrar aplicações cujos dados sejam linearmente separáveis devido a diversos fatores, entre eles a presença de ruídos e *outliers*, explicam os autores.

Desta forma, também existem as SVM com margens suaves, ilustrada de maneira simplificada na figura 4.

Figura 4 - Ilustração de uma SVM com margens suaves.



Fonte: Faceli *et al.* (2011)

Resumidamente:

> As SVMs lineares são eficazes na classificação de conjuntos de dados linearmente separáveis ou que possuam uma distribuição aproximadamente linear, e a versão de margens suaves tolera a presença de alguns ruídos e *outliers*. (FACELI et al., 2011, p.130)

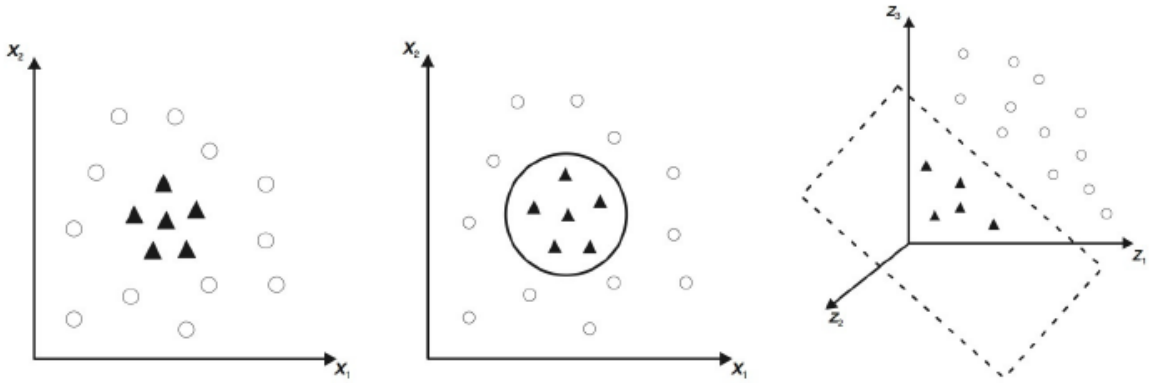
Porém, os casos em que não é possível separar os dados por meio de um hiperplano também existem e podem ser tratados por meio de SVM não lineares, como será explicado a seguir.

2.2.2 SVM não lineares

De acordo com Faceli *et al.* (2011), as SVMs não lineares lidam com problemas mapeando o conjunto de treinamento de seu espaço original para um espaço de

maior dimensão. Supondo que $\Phi : X \rightarrow I$ seja um mapeamento de um espaço de características (*feature space*), onde X é o espaço de entradas e I denota o espaço de características transformadas, a escolha apropriada de Φ faz com que o conjunto mapeado em I possa ser separado por uma SVM linear. Os autores afirmam que o uso deste procedimento é motivado pelo teorema de Cover. Dado um conjunto de dados não linear no espaço de entradas *X*, este teorema afirma que *X* pode ser transformado em um espaço de características I no qual, para que os objetos tenham alta probabilidade de ser linearmente separáveis, devem satisfazer a duas condições: a primeira é que a transformação seja não linear, enquanto a segunda é que a dimensão do espaço de características seja suficientemente alta. A figura 5 ilustra estes conceitos.

Figura 5 - Transformação realizada em um conjunto de dados não linear (a), onde a fronteira de separação é não linear (b), para o espaço de características (c), onde a fronteira de decisão pode ser representada por um hiperplano linear.



(a) Conjunto de dados (b) Fronteira para duas dimensões (c) Fronteira para três dimensões

Fonte: Faceli et al. (2011)

Na figura 5(a) tem-se um exemplo de conjunto de dados. Com o mapeamento representado na equação (2) , transforma-se os objetos de R^2 para R^3 (figura 5(b)), desta forma o conjunto não linear em R^2 torna-se linear em R^3, de acordo com a equação (3), como mostra a figura 5(c).

$$\Phi(x) = \Phi(x_1, x_2) = \left(x_1^2, \sqrt{2}x_1x_2, x_2^2\right) \quad (2)$$

$$h(x) = w \cdot \Phi(x) + b = w_1x_1^2 + w_2\sqrt{2}x_1x_2 + w_3x_2^2 + b = 0 \quad (3)$$

Logo, utilizando-se Φ é mapeado inicialmente os objetos para um espaço de maior dimensão e aplica-se a SVM linear de margens suaves sobre este espaço, pois esta permite lidar com ruídos e *outliers*. Ela encontra o hiperplano com maior margem de separação, garantindo assim uma boa generalização.

Faceli *et al.* (2011) também explicam que como I pode ter uma dimensão muito alta podendo até ser infinita, a computação de Φ pode ser extremamente custosa ou até mesmo inviável. Porém, a única informação necessária sobre o mapeamento é de como realizar o cálculo de produtos escalares entre objetos no espaço de características e isso é obtido a partir de funções denominadas *kernels*.

2.2.3 Função k*ernel*

De acordo com a literatura de Faceli et al. (2011), um *kernel K* é uma função que recebe dois pontos x_i e x_j no espaço de entradas e calcula o produto escalar desses objetos no espaço de características. Desta forma, tem-se:

$$K(x_i, x_j) = \Phi(x_i) \cdot \Phi(x_j) \tag{4}$$

Com o mapeamento apresentado na equação (2) e dois objetos x_i e x_j em R^2, por exemplo, tem-se:

$$K(x_i, x_j) = (x_i \cdot x_j)^2 \tag{5}$$

É comum empregar a função *kernel* sem conhecer o mapeamento Φ, que é gerado implicitamente. De acordo com os autores, a utilidade dos *kernels* está na simplicidade de seu cálculo e em sua capacidade de representar espaços abstratos.

Para garantir que o *kernel* represente mapeamentos nos quais seja possível o cálculo de produtos escalares, utilizam-se funções *kernel* que seguem as condições do teorema de Mercer. Para satisfazer este teorema, um *kernel* é caracterizado por dar origem a matrizes positivas semi-definidas **K**, em que cada elemento K_{ij} é definido por $K_{ij} = K(x_i, x_j)\ \forall\ i, j = 1, \dots, n$. Como se trata de uma SVM não linear,

escolheu-se para utilização deste trabalho o *kernel* de função de base radial (RBF, do inglês *Radial Basis Function*), que é regido pela equação (6).

$$K = exp\ exp\ \left(-\sigma\left|\left|x_i - x_j\right|\right|^2\right) \quad (6)$$

Onde σ representa o parâmetro do *kernel*, o qual deve ser definido pelo usuário. Alguns *kernels* mais utilizados na prática são os polinomiais, os RBF e os sigmoidais, porém, neste texto será abordado apenas o RBF.

2.3 ONE-CLASS SVM

Awad e Khanna (2015) afirmam que a formulação original das SVMs não contava com o desbalanceamento de classes durante a fase de aprendizado supervisionado, porém, algumas pesquisas propuseram uma modificação na formulação de SVMs para classificações com conjunto de dados desbalanceados. Kubat e Matwin (1997) recomendaram o balanceamento da base de dados fazendo a técnica de *undersampling* da classe majoritária ao invés de *oversampling* a classe minoritária. Já os autores Veropoulos *et al.* (1999) introduziram diferentes funções de perda para as classes positiva e negativa com a intenção de penalizar o erro de classificação dos dados da classe minoritária. Tax e Duin (1999) resolveram o problema de desbalanceamento usando a técnica de SVDD (*Support Vector Data Description*), que visa encontrar uma esfera que engloba a classe minoritária e a separa dos *outliers* o mais otimamente possível.

Em muitas aplicações reais e em base de dados não-sintetizados, os dados são desbalanceados. O desbalanceamento de classes apresenta um grande desafio para os algoritmos de classificação quando o risco de perda de informação da classe minoritária é maior do que da classe majoritária (AWAD; KHANNA, 2015, p. 35). Os autores declaram:

> Quando os dados pontuais da classe minoritária são mais importantes do que os da classe majoritária e o objetivo principal é a classificação correta destes pontos, os algoritmos padrões de aprendizado de máquina que

> giram em torno de otimização por acurácia não são ideais; isto resultará em hiperplanos que favorecem a classe majoritária e fazem uma generalização fraca. (AWAD; KHANNA, 2015, p. 35)

De acordo com Awad e Khanna (2015), os autores Tax e Duin (2004) avançaram com a técnica de *One-class SVM*, que tende a aprender apenas com a classe minoritária. O *One-class SVM* tem o objetivo de estimar a função de densidade de probabilidade, que fornece um valor positivo para elementos da classe minoritária e um valor negativo para qualquer outra coisa.

Por outro lado, Perdisci, Gu e Lee (2006) dizem que as técnicas de classificação *One-class* são particularmente úteis para problemas de aprendizagem com duas classes, os quais uma das classes, denominada classe alvo (*target*), é bem definida por amostragem, e a outra, denominada classe de anomalias (*outliers*), é demasiadamente pequena (baixa-amostragem). O objetivo da classificação *One-class* é construir uma superfície de decisão em torno de exemplos da classe alvo, afim de diferenciar entre objetos alvos e todos os outros possíveis objetos, ou seja, os *outliers*.

Perdisci, Gu e Lee (2006) afirmam que o problema de classificação *One-class* é formulado para encontrar o hiperplano que separa a fração desejada dos padrões de treinamento dos atributos no espaço. Este hiperplano nem sempre pode ser encontrado somente no espaço original dos atributos, então é utilizada uma função de mapeamento em um *kernel*, como descrito na seção 2.2.3.

A classe minoritária em alguns casos pode ser vista como um *outlier*, ou seja, uma anomalia. Castro e Ferrari (2017) definem anomalia como sendo um valor discrepante, ou seja, um valor que se localiza significativamente distante dos valores considerados normais. É importante notar que uma anomalia não necessariamente é um erro ou um ruído, ou seja, ela pode caracterizar um valor ou uma classe bem definida, porém de baixa ocorrência. Os autores também argumentam que a detecção de anomalias tem sido usada há séculos para detectar e, quando apropriado, executar alguma tomada de decisão sobre objetos anômalos da base de dados. No contexto deste trabalho, a classe minoritária será tratada como *outliers*.

3 METODOLOGIA

A partir dos conceitos e técnicas descritas nos capítulos anteriores, este capítulo apresentará a metodologia utilizada para o cumprimento dos objetivos propostos neste trabalho.

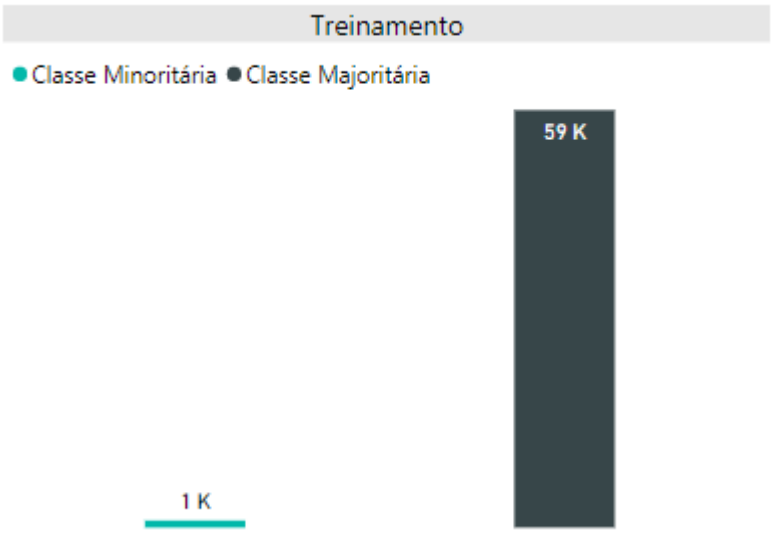
3.1 BASE DE DADOS

Para a construção do algoritmo foi utilizada uma base de dados fornecido pela marca Scania, mundialmente famosa pela fabricação de caminhões de carga. Este banco de dados foi disponibilizado em um repositório da UCI (DUA; GRAFF, 2019). Esta base de dados diz respeito a alguns tipos de falhas que podem ocorrer nos caminhões da Scania. A fabricante forneceu duas bases de dados separadamente: uma contendo 60 mil dados para o treinamento e validação da rede e outra contendo 16 mil dados para teste.

As bases de dados de treinamento e de teste possuem um total de 170 atributos. Além desses atributos, existe a informação de saída que é a coluna "*class*" (classe). As informações desta coluna estão divididas entre positivo e negativo, onde essa classificação diz respeito ao tipo da falha, ou seja, se é referente ao sistema APS ou não. A divisão da coluna "*class*" será referenciada neste trabalho como classe majoritária e classe minoritária, onde na classe majoritária estão contidas as falhas não referentes ao sistema APS e na classe minoritária estão contidas as falhas referentes a componentes do sistema APS.

A base de dados para treinamento possui um total de 59 mil amostras referentes à classe majoritária e 1 mil são referentes à classe minoritária. Já a base de dados de testes possui um total 15.625 amostras referentes à classe majoritária e apenas 375 referentes à classe minoritária. O gráfico da figura 6 fornece uma boa visualização de proporção das classes da base de treinamento.

Figura 6 - Distinção do banco de dados de treinamento entre "Classe Majoritária" e "Classe Minoritária". Os dados da classe majoritária são relativos ao sistema sem falhas e os dados da classe minoritária são relativos a uma falha do sistema APS do caminhão.



Fonte: Autora

Como mencionado na seção 1, o objetivo do algoritmo é a detecção das falhas no sistema APS no caminhão, ou seja, detectar todos os casos em que a variável "*class*" for um *outlier*.

3.2 *SOFTWARE* E COMPUTADOR

A linguagem escolhida para o desenvolvimento deste projeto foi o *Python*. Esta linguagem de alto nível pode ser utilizada para diversos fins, como construção de sistemas *Web*, aplicativos, sistemas *desktop*, mas também para análise de dados, inteligência artificial, *machine learning* (PYTHON..., 20-), os quais são o tema deste trabalho.

A interface utilizada foi o *Google Colaboratory*, que é um serviço de hospedagem em nuvem disponibilizado pela *Google* e pode ser acessado por um navegador. O *Google Colaboratory* é comumente utilizado com a finalidade de treinamento de redes de *machine learning* e pesquisa, pois é possível utilizar aceleração de GPU

gratuitamente. Este serviço é basicamente “um ambiente em *Jupyter* que não necessita de nenhum tipo de configuração” (FREQUENTLY..., 201-).

3.3 PRÉ-PROCESSAMENTO DOS DADOS

“Conhecer e preparar de forma adequada os dados para análise é uma etapa chamada de pré-processamento de dados” (CASTRO; FERRARI, 2017, capítulo 2.1). Castro e Ferrari (2017) também afirmam que dados mal ou não pré-processados podem inviabilizar uma análise ou invalidar um resultado.

As informações contidas em uma base de dados podem ser, por vezes, de alguma forma inconsistentes. Por exemplo, é comum um analista digitar um valor errado enquanto está preenchendo uma tabela, um sensor falhar durante uma medição e até uma pessoa mentir, por exemplo sua faixa salarial ou idade (CASTRO; FERRARI, 2017, capítulo 2.1).

De acordo com Faceli *et al.* (2011), as técnicas de pré-processamento de dados são úteis não apenas porque podem minimizar ou eliminar problemas existentes em um conjunto de dados, mas também porque podem tornar os dados mais adequados para sua utilização por um determinado algoritmo de aprendizado de máquina. Por esses motivos, esta é uma fase indispensável para o desenvolvimento deste trabalho.

Como já descrito, a base de dados utilizado para a execução deste trabalho consiste em dados coletados a partir do uso diário de caminhões de carga da Scania (INDUSTRIAL..., 2016). A problemática desta base de dados é focada no sistema de APS (*Air Pressure System*) dos caminhões, o qual fornece ar pressurizado para várias funções dos caminhões, como para os freios e engrenagens.

Esta seção foi dividida em subseções para o melhor entendimento do leitor sobre como foi feito o pré-processamento dos dados.

3.3.1 Tratamento de dados faltosos

Como já mencionado, a base de dados possui 60 mil objetos, porém é possível observar que nem todos os dados possuem informações em todos os atributos. A figura 7 mostra as primeiras linhas e primeiras colunas da base, onde observa-se que mesmo em uma pequena amostra do banco, é possível ver dados faltosos, destacados como “NaN” (do inglês, *not a number* que significa “não é um número”).

Figura 7 - Primeiras dez linhas e treze colunas do banco de dados de treinamento

	class	aa_000	ab_000	ac_000	ad_000	ae_000	af_000	ag_000	ag_001	ag_002	ag_003	ag_004	ag_005
0	0	76698	NaN	2130706438	280	0	0	0	0	0	0	37250	1432864
1	0	33058	NaN	0	NaN	0	0	0	0	0	0	18254	653294
2	0	41040	NaN	228	100	0	0	0	0	0	0	1648	370592
3	0	12	0	70	66	0	10	0	0	0	318	2212	3232
4	0	60874	NaN	1368	458	0	0	0	0	0	0	43752	1966618
5	0	38312	NaN	2130706432	218	0	0	0	0	0	0	9128	701702
6	0	14	0	6	NaN	0	0	0	0	0	0	1202	3766
7	0	102960	NaN	2130706432	116	0	0	0	0	0	0	2130	142462
8	0	78696	NaN	0	NaN	0	0	0	0	0	0	458	440704
9	1	153204	0	182	NaN	0	0	0	0	0	11804	684444	326536

Fonte: Autora

Para otimizar o resultado, fez-se uma análise de quatro possíveis tratamentos destes valores “NaN”:

- Realizar a remoção dos dados que continham atributos faltosos;
- Substituir os atributos faltosos pela mediana dos atributos;
- Substituir os atributos faltosos pela média dos atributos; ou
- Substituir os atributos faltosos pelos valores mais frequentes dos atributos.

Como esta base de dados apresentou muitos campos “NaN”, a opção de remover estes dados foi desconsiderada, pois muita informação seria perdida, tendo em vista que o conjunto de dados seria reduzido demasiadamente. Como será visto no

capítulo de resultados (Seção 4), os melhores resultados foram obtidos fazendo a substituição dos dados faltosos pelo valor médio de cada atributo.

A linguagem Python, na qual o algoritmo foi desenvolvido, permite um amplo leque de bibliotecas que podem ser usadas para facilitar a construção do código e obtenção dos resultados. Então, foram utilizadas as bibliotecas “pandas” e “numpy”, que lidam com operações e tratamentos de matrizes mais complicadas. A biblioteca “pandas”, que é uma abreviação para “*panel data*”, é uma poderosa biblioteca para manipulação de tabelas e análise de dados. A biblioteca “numpy” é um módulo utilizado para manipulação de dados numéricos. Estas duas bibliotecas são essenciais para o desenvolvimento deste projeto, pois elas permitem a interação com a biblioteca utilizada para o treinamento do estimador *One-class SVM*, o “scikit-learn”.

3.3.2 Padronização dos dados

Com a finalidade de otimizar os resultados, uma etapa indispensável neste trabalho é a padronização dos dados do conjunto de treinamento e teste. A padronização se faz necessária para evitar que um atributo predomine sobre o outro (FACELI *et al.*; 2011, p. 45). Ainda de acordo com Faceli *et al.* (2011), um outro motivo para esta etapa é que quando os limites superiores e inferiores dos atributos são muito diferentes, é, assim, pode ocorrer uma grande variação de valores dos dados.

De outra maneira, o objetivo principal da padronização é resolver as diferenças de unidades e escalas dos dados (CASTRO; FERRARI, 2017, capítulo 2.6.1). Os autores Castro e Ferrari (2017) afirmam que um tipo comum de problema das bases de dados brutas é a não uniformidade dos atributos e essas características afetam muitos dos algoritmos de mineração, logo, precisam ser devidamente tratadas antes de sua aplicação.

Para esta etapa, foi utilizada a biblioteca “*scikit-learn*”, que possui um método para o pré-processamento de dados. Com este método é possível chamar a função “*Standard Scaler*”, a qual faz a transformação dos dados a partir da normalização

de cada coluna (atributo), fazendo com que a média dos atributos seja 0 e o desvio padrão seja igual a 1.

3.3.3 *Principal Component Analysis* (PCA)

Castro e Ferrari (2017) articulam sobre a importância da redução de dimensionalidade:

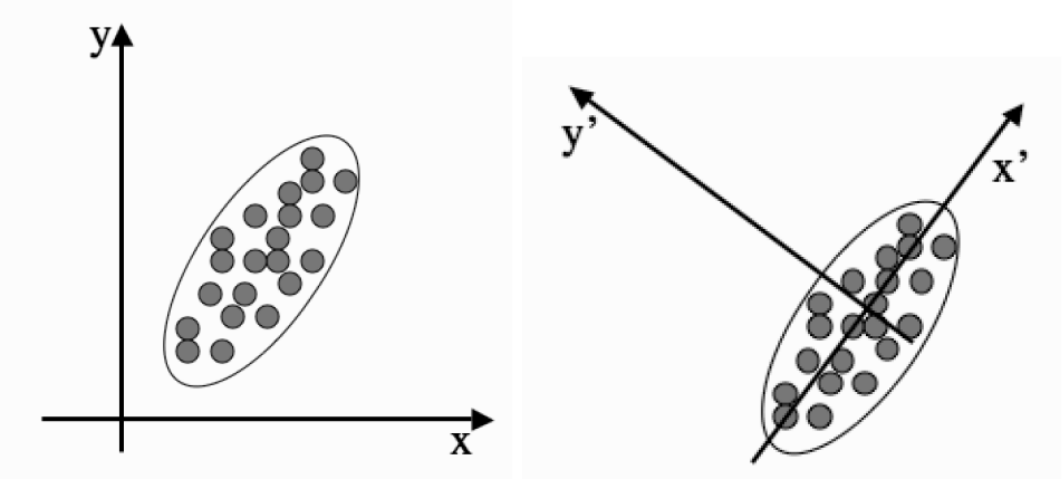
> [...]O aumento do número de objetos e da dimensão no espaço (número de atributos na base) pode fazer com que os dados disponíveis se tornem esparsos e as medidas matemáticas usadas na análise tornem-se numericamente instáveis. (CASTRO; FERRARI, 2017, capítulo 2.5).

Além disso, uma quantidade muito grande de objetos e atributos pode tornar o processamento de algoritmos de mineração muito complexos, assim como os modelos gerados, acrescentam os autores.

Segundo Faceli *et al.* (2011), uma das técnicas mais utilizadas para reduzir as dimensões por agregação é a Análise de Componentes Principais (PCA, do inglês *Principal Component Analysis*). Esta técnica correlaciona estatisticamente os atributos, reduzindo a dimensionalidade do conjunto de dados original pela eliminação de redundâncias.

De acordo com o autor Marsland (2015), a ideia do conceito do PCA é que encontrando um conjunto particular de eixos, torna-se claro de que algumas dimensões não são necessárias. Isso pode ser demonstrado na figura 8, que mostra duas versões do mesmo conjunto de dados.

Figura 8 - Dois diferentes conjuntos de eixos coordenados. O segundo consiste em uma rotação e translação do primeiro e foi encontrado utilizando PCA



Fonte: Marsland (2015)

No primeiro eixo da figura, os dados estão dispostos em uma elipse a um ângulo de 45° em relação aos eixos, enquanto no segundo eixo, os dados foram movidos de modo que os dados ao longo do eixo x fossem centrados na origem. O potencial de redução de dimensionalidade está no fato de que a dimensão y, no segundo caso, não apresenta muita variabilidade, então pode ser possível ignorá-lo e utilizar somente os valores do eixo x individualmente.

A biblioteca "*scikit-learn*" utilizada no algoritmo já fornece a função PCA. A implementação desta função permite o usuário escolher entre dar a razão da variância (entre 0 e 1) ou a quantidade de componentes. Neste caso, desejou-se configurar a variância em 0,63. Este valor pode parecer baixo, mas a partir de testes feitos diminuindo a variância do PCA, percebeu-se que os resultados do estimador melhoravam e o ponto ótimo foi no valor de 0,63.

3.4 TREINAMENTO E VALIDAÇÃO

O treinamento do estimador foi feito a partir da biblioteca “scikit-learn” por meio dos métodos “GridSearchCV” aliado ao “OneclassSVM”. Na seção a seguir haverá uma explicação de como foram utilizadas estas ferramentas.

3.4.1 GridSearchCV e *One-class* SVM

O “*GridSearchCV*” é uma função da biblioteca “*scikit-learn*” que é utilizada para a procura exaustiva de parâmetros específicos para um estimador (GRIDSEARCHCV, 201-), onde este estimador será neste caso o *One-class SVM*, implementada pelo método “*OneclassSVM*” da mesma biblioteca.

De acordo com a documentação do “*scikit-learn*”, esta função realiza uma procura exaustiva pela melhor combinação de parâmetros do classificador, ou seja:

> Os parâmetros do estimador utilizados para a aplicação deste método são otimizados através de procura por validação cruzada em uma grade de parâmetros (*grid-search cross-validation*). (GRIDSEARCHCV, 201-).

Para que o “*GridSearchCV*” faça a escolha do melhor estimador, é necessário inserir os parâmetros para o estimador. Os valores dos parâmetros inseridos foram:

- *Gamma* = [0.01, 0.1, 0.2, 1]
- *Nu* = [0.01, 0.1, 0.16, 0.5]
- *Kernel* = [‘rbf’]
- *Scoring* = ‘*precision*’

O parâmetro ‘*gamma*’ é um coeficiente que define o quanto um único treinamento tem influência sobre um exemplo (RBF..., 201-). Este parâmetro é referente ao *kernel* escolhido (‘rbf’) do *One-classSVM*. O parâmetro *kernel* especifica o tipo de transformação será realizada pelo algoritmo.

Foi escolhido o ‘rbf’, pois se trata de um problema que não pode ser linearmente separável (não linear).

O parâmetro ‘*nu*’ é um valor de delimitação superior da fração de erros de treinamento e uma delimitação inferior dos vetores de suporte (RBF..., 201-). Este parâmetro deve estar entre o intervalo (0,1). O parâmetro ‘*scoring*’ diz respeito à seleção do modelo e a utilização de ferramentas de avaliação do modelo (MODEL..., 201-). Neste caso, utilizou-se a precisão (*precision*) como métrica de avaliação do modelo.

O método “*OneClassSVM*” da biblioteca *scikit-learn* foi o estimador escolhido para solucionar a problemática deste trabalho. Ao realizar a predição com um banco de dados utilizando-se este método, ele retornará o valor -1 para *outliers*, ou seja, para a classe minoritária, e 1 para dados que pertencem à classe treinada (*inliers*), ou seja, classe majoritária.

Após muitas tentativas, o método “*GridSearchCV*” não apresentou bons resultados, então optou-se por não utilizar esta função. Ao invés disso, treinou-se o estimador *OneClassSVM* assumindo alguns valores e analisando as respostas.

Os parâmetros que geraram os melhores resultados foram:

- Gamma = 0.1
- Nu = 0.03
- Kernel = ‘rbf’

O treinamento do estimador com o método *OneClassSVM* foi feito com apenas os dados da classe majoritária, ou seja, os que não representam uma falha no sistema APS do caminhão. Como mencionado no capítulo 2, o *One-class SVM* faz o agrupamento dos dados de uma só classe (neste caso, a classe majoritária) e todas as outras informações que não se enquadram dentro desta classe são consideradas *outliers*. É importante destacar que o estimador escolhe como *inlier* (retorno 1) os dados que não são de falha no sistema APS, e *outlier* com retorno -1 os dados que são de falha no sistema APS.

4 ANÁLISE E DISCUSSÃO DOS RESULTADOS

Na seção 3.4.1, foi explicado que o método utilizado retorna os valores -1 para *outliers* e 1 para *inliers*. Desta forma, foi possível contabilizar a quantidade de dados que o estimador acusou como *inliers* (os dados que não representam uma falha no sistema APS) e a quantidade de *outliers* (aqueles que representam falha em componentes do sistema APS do caminhão). Com estas quantidades em mãos foi possível obter a matriz confusão dos resultados.

De acordo com Awad e Khanna (2015) a matriz confusão em aprendizado não supervisionado é chamada *matching matrix* e fornece uma representação visual da acurácia das classes reais *versus* previstas.

> Ao lidar com conjuntos de dados desequilibrados, a acurácia geral é uma medida tendenciosa da qualidade do classificador. Em vez disso, a matriz de confusão e as informações sobre verdadeiro positivo (VP) e falso positivo (FP) que ela contém são uma melhor indicação do desempenho do classificador. (AWAD; KHANNA, 2015, p. 35)

Assim como a matriz confusão, a *matching matrix* é representada da seguinte forma:

Tabela 1 - *Matching matrix* ou matriz de confusão para um problema binário

		Classe predita	
		+	**-**
Classe verdadeira	**+**	VP	FN
	-	FP	VN

Fonte: FACELI et al., 2011, p. 164

A classe positiva é a classe de maior importância para o estimador e geralmente é a classe minoritária, explicam Awad e Khanna (2015). Porém, neste trabalho a classe positiva é caracterizada pela classe majoritária, pois foi a classe utilizada para realizar o treinamento do estimador.

Para o entendimento da tabela 1, os autores Faceli *et al*. (2011) esclarecem os termos:

- Verdadeiro Positivo (VP) é o número de exemplos da classe positiva classificados corretamente;
- Verdadeiro Negativo (VN) é o número de exemplos da classe negativa classificados corretamente;
- Falso Positivo (FP) é o número de exemplos classificados na classe positiva, mas que na verdade pertencem a classe negativa;
- Falso Negativo (FN) é o número de exemplos classificados na classe negativa, mas que na verdade pertencem a classe positiva.

Os valores de VP representam então todos os dados de falhas que não são referentes ao sistema APS que o estimador previu corretamente, ou seja, serão os caminhões que não apresentarão falhas no APS. Os valores de VN são as falhas previstas corretamente pelo estimador que são referentes ao sistema APS, ou seja, serão todos os casos em que o caminhão terá uma falha referente ao sistema APS.

Já os valores de FP e FN são os dados em que o estimador previu as falhas de forma errada. Os FP serão todos os casos em que o estimador acusou como falha em outros sistemas do caminhão (que não era no APS) e errou. Isso traz uma consequência no custo de manutenção, pois este erro será referente ao Custo 2 ($500). No caso dos FN, representam-se então todas as falhas que o estimador previu como sendo no sistema APS, porém o fez de forma errada. Este erro terá também uma consequência no custo final de manutenção, pois este erro é referente ao Custo 1 ($10).

A precisão e a revocação (ou recall) são outros dois indicadores de desempenho do detector que também serão avaliados neste trabalho. Os autores Faceli *et al*. (2011) definiram a precisão como sendo a proporção de exemplos positivos classificados corretamente entre todos aqueles preditos como positivo pelo estimador. A equação (7) mostra como este indicador é calculado:

$$Precisão = \frac{VP}{VP+FP} \tag{7}$$

Faceli *et al*. (2011) elucidam que revocação também pode ser chamada de sensibilidade e definem como sendo a taxa de acerto na classe positiva. Também

é chamada de taxa de verdadeiros positivos (TVP). É calculado de acordo com a equação (8):

$$Sensibilidade = Revocação = TVP = \frac{VP}{VP+FN} \quad (8)$$

Outro fator que também será avaliado na próxima seção é o custo de manutenção dos caminhões de carga com a utilização do detector de *outliers* desenvolvido.

4.1 RESULTADOS DO *ONE-CLASS SVM*

O primeiro passo é avaliar a matriz de confusão gerada com o teste do *One-class SVM*. A Tabela 2 apresenta o resultado do teste realizado pela metodologia proposta considerando-se os parâmetros estabelecidos na seção 3.4.1 e considerando a substituição dos dados faltantes pela média dos valores das respectivas colunas onde há falta de dados.

Tabela 2 - Matriz confusão das predições do algoritmo

		Classe predita	
		+	**-**
Classe verdadeira	**+**	14146	1479
	-	1	374

Fonte: Autora

Interpretando a matriz confusão da Tabela 2, é possível observar que dentre os 16 mil dados reservados para testes, o algoritmo acertou, em meio de *outliers* e *inliers*, 14.520 dados, apresentando 90% de acerto da classe majoritária (*inliers*) e 99,7% de acerto na classe minoritária (*outliers*).

De acordo com as fórmulas apresentadas no início desta seção, a precisão do algoritmo é de 0,99, o que representa um resultado excelente, tendo em vista que o número de FP indicado pelo algoritmo é apenas 1. Avançando para outro indicador de qualidade do algoritmo, a sensibilidade do algoritmo foi de 0,90.

4.2 COMPARAÇÕES E CUSTOS DE MANUTENÇÃO

Os custos de manutenção fornecidos pela Scania são:

- Custo 1: $10, que representa uma vistoria desnecessária; e
- Custo 2: $500, que representa uma falha não prevista, resultando em uma manutenção corretiva.

O custo de manutenção total sem a utilização do algoritmo foi calculado da seguinte forma:

$$Total = Custo\ 1 * 16.000 = \$160.000 \quad (9)$$

Neste caso, presume-se que em todos os casos os caminhões passarão por uma vistoria. Por outro lado, o algoritmo faz uma seleção dos casos em que os caminhões precisarão passar por uma vistoria ou não. O custo de manutenção com a utilização do estimador *One-class SVM* foi calculado da seguinte forma:

$$Total_{OCSVM} = Custo\ 1 * FN + Custo\ 2 * FP \quad (10)$$

Assim, o custo total para o melhor caso, apresentado na tabela 2 foi de $15.290,00.

É possível observar que com a utilização do algoritmo desenvolvido neste trabalho, a empresa terá uma economia de $144.710. Esta economia mostra que o estimador *One-class SVM* é uma excelente alternativa para casos em que a detecção de anomalias é de suma importância.

Apesar do resultado mostrado acima ter sido o melhor obtido, para enriquecer este trabalho, também foi feito um teste com a inserção dos valores de mediana nos atributos com dados faltosos na etapa de pré-processamento (vide seção 3.3.1), ao invés da inserção da média. Os parâmetros inseridos no *One-class SVM* foram exatamente os mesmos que no caso acima (*nu*=0.03, *gamma*=0.1, pca=0.63). Os resultados também foram satisfatórios, como mostra a tabela 3.

Tabela 3 - Matriz confusão do estimador *One-class SVM* com inserção de medianas

		Classe predita	
		+	**-**
Classe verdadeira	**+**	14196	1429
	-	7	368

Fonte: Autora

Com a substituição dos dados faltosos pelas medianas dos atributos, o algoritmo acertou um total de 14.564 dados, dentre os 16.000, apresentando 88,7% de acerto da classe majoritária (*inliers*) e 98% de acerto da classe minoritária (*outliers*). Os indicadores de qualidade de precisão e sensibilidade também foram calculados e são, respectivamente, 0,99 e 0,9.

Neste caso, o custo de manutenção dos caminhões seria então $17.790, o que resulta em uma economia de $142.210 para a empresa.

Na expectativa de encontrar um estimador que retornasse 100% de acerto dos *outliers*, executou-se um outro teste com os seguintes parâmetros:

- Imputação do valor médio para dados faltosos;
- Variância do PCA = 0.73
- Parâmetros do *One-class SVM*: *Nu*=0.03 e *Gamma*=0.1

A tabela 4 mostra a matriz confusão do resultado do estimador para estes parâmetros.

Tabela 4 - Matriz confusão para um estimador com 100% de acerto de *outliers*

		Classe predita	
		+	**-**
Classe verdadeira	**+**	13906	1719
	-	0	375

Fonte: Autora

Neste caso, o algoritmo acertou um total de 100% dos dados da classe minoritária (*outliers*), porém acertou 86,9% dos dados da classe majoritária (*inliers*). O custo de manutenção para este estimador é de $17.190, resultando em uma economia de $142.810 para a empresa. A precisão e revocação para este caso foram respectivamente 1 e 0,88.

Para facilitar a visualização, foi feita uma comparação do melhor resultado deste trabalho com os citados no capítulo de trabalhos correlatos (Seção 1.3), que está descrita na tabela 5.

Tabela 5 - Tabela de comparação deste trabalho com outros

Autores	**Técnica**	**Custo de manutenção**	**Economia para a empresa**
Cerqueira *et al.* (2016)	*Gradient Boost*	$4.560	$155.440
Costa e Nascimento (2016)	*Random Forest*	$40.570	$119.430
Gondek *et al.* (2016)	*Random Forest*	$36.000	$124.000
Rafsunjani *et al*. (2019)	*Random Forest*	-	-
Oznan *et al*. (2016)	*GDE*	$10.792	$149.208
Este trabalho	***One-class SVM***	**$15.290,00**	**$144.710**

Fonte: Autora

Para calcular a coluna “Economia para a empresa”, foi utilizado o valor base de manutenção definido neste trabalho ($160.000), porém, não é possível saber se os autores utilizaram este valor como base.

Os autores Rafsunjani *et al.* (2019) não participaram do desafio do IDA e o foco do trabalho foi provar o quanto o método de imputação de dados era relevante para os resultados. Por isso, os autores não divulgaram os valores de manutenção.

É válido ressaltar que cada autor desenvolveu métodos de otimização e minimização de custos diferentes. Os autores Oznan *et al.* (2016) realizaram a técnica de imputação de dados por meio de KNN, porém, desenvolveram o modelo de classificação por uma função custo calculada por meio de GDE (Gradiente Descendente Estocástico). Os autores Cerqueira et al. (2016) receberam patrocínio de várias entidades (ECSEL *Joint Undertaking*, ERDF – *European Regional Development Fund*, FCT – Fundação para Ciência e Tecnologia Portuguesa) para desenvolver e realizar o trabalho. Costa e Nascimento (2016) também receberam financiamento parcial do projeto desenvolvido pelo NSERC Canada (NATURAL ..., 2019).

O estimador *One-class SVM* mostrou um diferencial muito importante dos outros trabalhos apresentados, pois em uma configuração específica foi capaz de detectar todas as falhas no sistema APS do caminhão, enquanto nenhum outro trabalho foi capaz de detectar 100% dos *outliers*. Isso é de extrema relevância, tendo em vista que o sistema APS é um sistema crítico do caminhão, o qual faz o gerenciamento da alimentação de ar comprimido para freios e engrenagens. É válido salientar que freios e engrenagens são funcionalidades essenciais, onde uma falha neste sistema pode colocar a segurança do motorista (e também outras pessoas) em risco.

5 CONCLUSÃO

O trabalho apresentou uma técnica não supervisionada para detecção de *outliers* em uma base de dados de falhas do sistema APS em caminhões da marca Scania. Neste sentido, os resultados obtidos através do estimador *One-class SVM* para o caso estudado, com a substituição dos dados faltantes pela média dos valores das respectivas colunas, se mostraram satisfatórios, alcançando um valor de custo de $15.290, ou seja, uma economia de $144.710 para a empresa. Além disso, em uma configuração específica, o algoritmo foi capaz de encontrar todos os *outliers*, porém, com um aumento do erro na detecção de *inliers,* o qual o custo de manutenção final foi $17.190.

A técnica de aprendizado de máquina não supervisionado que foi apresentada neste trabalho pode ser aplicada também a várias outras situações, como em detecção de fraudes em sistemas de distribuição de energia, detecção de fraudes em cartões de crédito, detecção de anomalias em equipamentos industriais, entre outros. Além disso, é importante destacar como trabalho futuro a utilização do algoritmo XGboost para comparação com a metodologia adota aqui.

REFERÊNCIAS

17 CASOS de uso de Machine Learning. 2018. Elaborada por Equipe DAS. Disponível em: <http://datascienceacademy.com.br/blog/17-casos-de-uso-de-machine-learning/>. Acesso em: 26 nov. 2019

AWAD, Mariette; KHANNA, Rahul. **Efficient Learning Machines: Theories, Concepts, and Applications for Engineers and System Designers**. Nova Iorque: Springer, 2015.

BADBURY, Steve; CARPIZO, Brian; GENTZEL, Matt; HORAH, Drew; THIBERT, Joël. **Confiabilidade viabilizada digitalmente: além da manutenção preditiva**. 2018. Disponível em: <https://www.mckinsey.com/business-functions/operations/our-insights/digitally-enabled-reliability-beyond-predictive-maintenance/pt-br>. Acesso em: 26 nov. 2019.

CASTRO, Leandro Nunes; FERRARI, Daniel Gomes. **Introdução à mineração de dados: conceitos básicos, algoritmos e aplicações**. 1 ed. São Paulo: Saraiva, 2016.

CERQUEIRA, Vítor; PINTO, Fábio; SÁ, Claudio; SOARES, Carlos. **Combining Boosting Trees with Metafeature Engineering for Predictive Maintenance.** 2016. Disponível em: <https://www.researchgate.net/publication/313067390_Combining_Boosted_Trees_with_Metafeature_Engineering_for_Predictive_Maintenance>. Acesso em: 01 nov. 2019.

FREQUENTLY Asked Questions. 201-. Disponível em: <https://research.google.com/colaboratory/faq.html>. Acesso em: 07 nov. 2019.

COSTA, Camila; NASCIMENTO, Mario. **IDA 2016 Industrial Challenge: Using Machine Learning for Predicting Failures**. 2016. Disponível em: <https://www.researchgate.net/publication/313065916_IDA_2016_Industrial_Challenge_Using_Machine_Learning_for_Predicting_Failures>. Acesso em: 01 nov. 2019.

DUA, Dheeru; GRAFF, Casey. **UCI Machine Learning Repository.** 2019. Disponível em: <http://archive.ics.uci.edu/ml>. Acesso em: 26 nov. 2019.

FACELI, Katti; LORENA, Ana Carolina; GAMA, João; CARVALHO, André Carlos P. L. F. **Inteligência Artificial: Uma Abordagem de Aprendizagem de Máquina**. Rio de Janeiro: LTC, 2011.

GONDEK, Christopher; HAFNER, Daniel; SAMPSON, Oliver R. **Prediction of Failures in the Air Pressure System of Scania Trucks using a Random Forest and Feature Engineering**. 2016. Disponível em: <https://www.researchgate.net/publication/309195602_Prediction_of_Failures_in_the_Air_Pressure_System_of_Scania_Trucks_Using_a_Random_Forest_and_Feature_Engineering>. Acesso em: 01 nov. 2019.

GridSearchCV. 201-. Disponível em: <https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html>. Acesso em: 18 nov. 2019.

HUANG, Te-Ming; KECMAN, Vojislav; KOPRIVA, Ivica. **Kernel Based Algorithms for Mining huge Data Sets Supervised, Semi-supervised and Unsupervised Learning.** Países Baixos: Springer, 2006.

INDUSTRIAL CHALLENGE. 2016. Disponível em: <https://ida2016.blogs.dsv.su.se/?page_id=1387>. Acesso em: 26 nov. 2019

KUBAT, Miroslav; MATWIN, Stan. **Addressing the Curse of Imbalanced Training Sets: One-Sided Selection**. 1999. Disponível em: < https://sci2s.ugr.es/keel/pdf/algorithm/congreso/kubat97addressing.pdf >. Acesso em: 26 nov. 2019.

MARSLAND, Stephen. **MACHINE LEARNING An Algorithmic Perspective**. 2 ed. Boca Raton: CRC Press Taylor & Francis Group, 2015.

MITCHELL, Tom M. **Machine Learning**. 1 ed. Nova Iorque: McGraw-hill, 1997.

Model evaluation: quantifying the quality of predictions. 201-. Disponível em: <https://scikit-learn.org/stable/modules/model_evaluation>. Acesso em: 19 nov. 2019.

NATURAL Sciences and Engineering Research Council of Canada. 2019. Disponível em: < https://www.nserc-crsng.gc.ca/index_eng.asp>. Acesso em: 27 nov. 2019.

PERDISCI, Roberto; GU, Guofei; LEE, Wenke. **Using an Ensemble of One-Class SVM Classifiers to Harden Payload-based Anomaly Detection Systems**. Sixth International Conference On Data Mining (icdm'06), [s.l.], p.1-11, dez. 2006. IEEE. http://dx.doi.org/10.1109/icdm.2006.165.

PYTHON para quem está começando. 20-. Disponível em: <https://python.org.br/introducao/>. Acesso em: 26 nov. 2019.

O POTENCIAL da Inteligência Artificial na indústria. 201-. Disponível em: <http://www.vdibrasil.com/o-potencial-da-inteligencia-artificial-na-industria/>. Acesso em: 26 nov. 2019.

OZNAN, Ezgi Can; RIABCHENKO, Ekaterina; KIRANYAZ, Serkan; GABBOUJ, Moncef. **An Optimized k-NN Approach for Classification on Imbalanced Datasets with Missing Data**. 2016. Disponível em: <https://www.researchgate.net/publication/313065896_An_Optimized_k-NN_Approach_for_Classification_on_Imbalanced_Datasets_with_Missing_Data>. Acesso em: 26 nov. 2019.

RAFSUNJANI, Siam; SAFA, Rifat Sultana; IMRAN, Abdullah Al; RAHIM, Shamsur; NANDI, Dip. **An Empirical Comparison of the Missing Value Imputation Techniques on APS Failure Prediction**. 2019. Disponível em: < https://www.researchgate.net/publication/330854331_An_Empirical_Comparison_of_Missing_Value_Imputation_Techniques_on_APS_Failure_Prediction>. Acesso em: 26 nov. 2019.

RBF SVM parameters. 201-. Disponível em: <https://scikit-learn.org/stable/auto_examples/svm/plot_rbf_parameters.html>. Acesso em: 19 nov. 2019.

SILVEIRA, Cristiano B. **O Que é Indústria 4.0 e Como Ela Vai Impactar o Mundo**. 201-. Disponível em: <https://www.citisystems.com.br/industria-4-0/>. Acesso em: 26 nov. 2019

TAX, David M. J.; DUIN, Robert P.W. **Support Vector Data Description**. 2004. Disponível em: <http://homepage.tudelft.nl/a9p19/papers/ML_SVDD_04.pdf>. Acesso em: 26 nov. 2019.

TAX, David M. J.; DUIN, Robert P.W. **Support Vector Domain Description**. 1999. Disponível em: <http://rduin.nl/papers/prl_99_svdd.pdf>. Acesso em: 26 nov. 2019.

VAPNIK, Vladimir. **The Nature of Statistical Learning Theory**. 2 ed. Nova Iorque: Springer, 1995.

VAPNIK, Vladimir; CHERVONENKIS, Alexey. **Theory of Probability and its Applications.** 1971.

VEROPOULOS, Konstantin; CAMPBELL, Ian C.G.; CRISTIANINI, Nello. **Controlling the Sensitivity of Support Vector Machines.** 1999. Disponível em: < https://pdfs.semanticscholar.org/44f9/51f88a82b19f36ccf46768f9c02da86c3ecf.pdf ?_ga=2.2360816.1767048078.1574822570-282204442.1574822570>. Acesso em: 26 nov. 2019