UNIVERSIDADE ESTADUAL DO MARANHÃO-UEMA
CENTRO DE CIÊNCIAS TECNOLÓGICAS - CCT
CURSO DE PÓS-GRRADUAÇÃO EM ENGENHARIA DA COMPUTAÇÃO E SISTEMAS
SISTEMAS COMPUTACIONAIS APLICADOS A ENGENHARIA AEROESPACIAL

Jacyeude de Morais Passos Araújo Segundo

DESENVOLVIMENTO DE UM IDENTIFICADOR DE FALHAS PARA MOTOR TRIFÁSICO DE INDUÇÃO BASEADO EM MÁQUINAS DE VETORES DE SUPORTE IMPLEMENTADO EM CLOUD

Campus Paulo VI - UEMA
São Luís, MA – Brasil
2019

Jacyeude de Morais Passos Araújo Segundo

Dissertação apresentada à Pró-Reitoria de Pós-Graduação da Universidade Estadual do Maranhão, como parte dos requisitos para obtenção do título de Mestre em Engenharia da Computação do Curso de Mestrado Profissional em Engenharia da Computação e Sistemas.

Aprovado em 11/10/2019

Banca Examinadora:

Dr. Mauro Sergio Silva Pinto
Universidade Estadual do Maranhão - UEMA
Orientador

Dr. Leonardo Ramos Rodrigues
Instituto de Aeronáutica e Espaço - IAE
Coorientador

Dr. Luis Carlos Costa Fonseca
Universidade Estadual do Maranhão - UEMA
Examinador externo

Dr. Shigeaki Leite de Lima
Universidade Federal do Maranhão - UFMA
Examinador externo à instituição

UEMA - Campus Paulo VI
São Luis, MA – Brasil – 2019

Segundo, Jacyeude de Morais Passos Araújo.

Desenvolvimento de um identificador de falhas para motor trifásico de indução baseado em máquinas de vetores de suporte implementado em cloud / Jacyeude de Morais Passos Segundo. – São Luís, 2019.

105 f

Dissertação (Mestrado) – Curso de Engenharia da Computação e Sistemas, Universidade Estadual do Maranhão, 2019.

Orientador: Prof. Dr. Mauro Sergio Silva Pinto.

1.Computação na nuvem. 2.Motor trifásico de indução. 3.Máquinas de vetores de suporte. 4.Aprendizado de máquina. Desenvolvimento de um identificador de falhas para motor trifásico de indução baseado em máquinas de vetores de suporte implementado em cloud.

CDU: 621.313.333

CESSÃO DE DIREITOS

NOME DO AUTOR: Jacyeude de Morais Passos Araújo Segundo

TÍTULO DO TRABALHO: Desenvolvimento de um identificador de falhas para motor trifásico de indução baseado em máquinas de vetores de suporte implementado em cloud.

TIPO DO TRABALHO/ANO: Dissertação / 2019.

Jacyeude de Morais Passos Araújo Segundo

Avenida João Pessoa, nº 25. Condomínio Arco Verde Bl 05 Ap 207

CEP: 65040-470, São Luis - MA

Dedicatória

Dedico esta obra a meu pai Jacyeude Araújo e minha mãe Teresinha Jansen, pelo exemplo de vida, educação, bom senso e garra na busca de diferentes perspectivas. Dedico também à toda minha família, namorada e amigos, pelo apoio oferecido em todos os momentos.

Agradecimentos

À minha família, em especial aos meus pais pelo amor e incentivo incondicional a superar as dificuldades.

À Universidade Estadual do Maranhão e a instituição de fomento à pesquisa FAPEMA, por ofertarem infraestrutura e subsídios necessários para o desenvolvimento do programa de pós-graduação.

Aos docentes do programa de pós-graduação, com quem desenvolvi a base para uma nova jornada.

Aos meus orientadores, Prof. Mauro e Prof. Leonardo Ramos, com quem compartilhei conhecimentos e obtive suporte, críticas e sugestões fundamentais. Aos professores membros da comissão examinadora.

A todos os companheiros do PECS que fizeram parte desta formação.

Ao engenheiro e nobre amigo Carlos Vinicius Coimbra, com quem obtive suporte técnico e vivenciei diferentes momentos norteadores para o desenvolvimento deste trabalho.

À minha namorada Jéssica pelo incentivo e companheirismo.

E a todos que contribuíram direta ou indiretamente para a realização da minha pesquisa.

"In the midst of chaos, there is also opportunity"
Sun Tzu

RESUMO

Ao passo que a Indústria 4.0 avança, conjuntos de ações de automação e controle vem sendo implementados, dentro deste contexto o sensoriamento de motores de indução trifásicos torna-se remoto e conectado à internet. A partir disso, um fluxo de dados é continuamente analisado através de computação na nuvem em centros de processamento de dados. A manutenção preventiva pode então utilizar esse grande volume de dados para aumentar sua capacidade de detecção de falhas em relação aos métodos clássicos de classificação. Este trabalho propõe o desenvolvimento de um identificador de falhas externas no motor trifásico de indução W22 IR3, com base em dados de análises vibracionais e de correntes elétricas. Estes dados serão gerados utilizando um sistema para aquisição de dados que consiste em um acelerômetro MEMS (*Microelectromechanical Systems*) controlado por um SoC (*System on chip*) e um transformador de corrente não invasivo SCT-013. A análise dos dados foi realizada na IBM Cloud através de Watson Studio e SPSS Modeler para aplicação de um modelo estatístico MVS que foi treinado e testado usando diferentes funções kernel, utilizando a base de dados gerados pelo sistema embarcado de aquisição de dados.

Palavras-chave: Computação na nuvem, motor trifásico de indução, máquinas de vetores de suporte, aprendizado de máquina, IBM Cloud e SPSS Modeler.

ABSTRACT

As the Industry 4.0 advances, the induction motor sensoring becomes remote and connected to the Internet. With this, a data flow can be continuously analyzed through cloud computing in data centers. The predictive maintenance can then use this big data to increase your ability do detect faults when compared to the classical multi-signal approach. The purpose of this work is to develop a failure identifier in induction motor W22 IR3, based on current and vibration analysis. The data from these currents and vibrations are generated utilizing an embedded system for data acquisition consisting in a MEMS (Microelectromechanical Systems) controlled by a SoC (System on a Chip) and a current transformer. The data analysis will be implemented in IBM Cloud through Watson Studio and SPSS Modeler to apply an MVS statistical model that will be trained and tested using different kernel functions, using the database generated by the embedded acquisition system.

Key Words: Cloud computing, motor induction trifasic, support vector machine, machine Learn, IBM Cloud e SPSS Modeler.

Índice De Ilustrações

Índice De Tabelas

Lista de siglas e abreviações

AM	Aprendizado de Máquina
CRISP-DM	*Cross Industry Standard Process for Data Mining*
CSO	*Cloud Object Storage*
GPIO	*General-purpose input/output*
IaaS	Infraestrutura como Serviço
IA	Inteligência Artificial
IDA	Algoritmo de Dispersão de Informações
IDE	*Integrated Development Environment*
i.i.d.	Independente e identicamente distribuída
KDD	*Knowledge discovery in database*
MVS	Máquinas de Vetores de Suporte
MEMS	*Microelectromechanical Systems*
NIST	Instituto Nacional de Padrões e Tecnologia
PaaS	Plataforma como serviço
RNA	Rede neural artificial
RBF	Radial basis function
SPI	Protocolo Serial Síncrono
SaaS	*Software* como Serviço
TAE	Teoria de Aprendizado Estatístico
UUID	*Universally unique identifier*
VC	Dimensão Vapnik chevronik.

Lista de Símbolos

n	Velocidade do rotor
n_s	Velocidade do campo girante do estator
n_e	Velocidade de escorregamento
C	Parâmetro de Margem
f	Frequência em hertz
$\hat{f}$	Classificador particular
F	Conjunto de classificadores
p	Número de pólos
s	Escorregamento
T_p	Conjugado de partida
T_{min}	Conjugado mínimo
T_{max}	Conjugado máximo
T_{nom}	Conjugado nominal
T_{vazio}	Conjugado para o motor operando sem carga
γ	Gama
ε	Precisão de Regressão
T	Conjunto de treinamento
x_i , x_j, y_i	Amostra de dado
F_i	Subconjuntos
$\|w\|$	Magnitude de w
$\|x\|$	Magnitude de x
$R_p(f)$	Erro de margem de uma função
$R_{emp}(f)$	Risco Empírico
L	Multiplicadores de Lagrange
ξi	Erro no conjunto de treinamento
SV	Vetores de Suporte
ϕ	Função kernel
X	Espaço Amostral
β	Espaço de características

δ , k	Parâmetros de kernel
w	Vetor normal ao hiperplano descrito
x	Vetor descrito pela amostra
P	Distribuição de probabilidade
c , b	Valor constante
σ , ρ	Margem do classificador
θ	Espaço probabilístico
sgn	Função sinal
H_1 e H_2	Hiperplanos

Sumário

OBJETIVOS

Objetivos Gerais

Este trabalho visa desenvolver um sistema para o reconhecimento de padrão de falhas por meio de aquisição de dados de condições de falhas externas e condições normais de um motor trifásico de indução WEG W22-IR3 em laboratório.

Objetivos Específicos

- Desenvolvimento de um sistema de aquisição de dados de dados de vibração e correntes elétricas, de diferentes condições de funcionamento do motor de indução trifásico.
- Obter dados para alimentar análises preditivas do condicionamento em diferentes situações do motor trifásico de indução.
- Utilizar a infraestrutura disponível gratuitamente da Cloud para aplicação de MVS e investigar a capacidade dos métodos para modelagem de previsão.
- Avaliar o desempenho do classificador utilizando diferentes funções kernels e fazer uma discussão do seu impacto na acurácia dos resultados.
- Contribuir com o desenvolvimento, estudo e aplicação de sistemas de investigação de caraterísticas e mineração de dados.

JUSTIFICATIVA

O desenvolvimento de sistemas embarcados vem beneficiando o controle de manutenção na atualidade, de modo a aumentar a disponibilidade de operação das máquinas e controlar melhor as intervenções de manutenção. Porém, há de se notar que bons hardwares de sistemas disponíveis no mercado possuem um elevado custo de implementação, não sendo viáveis para uma difusão em larga escala. Este estudo propõe uma alternativa prática e sistemática para análise de falhas externas em motor trifásico de indução usando um sistema de captação de dados composto por um *System on Chip* – ESP8266, sensor acelerômetro MEMS e sensor de corrente SCT-013, para gerar o banco de dados necessários para as implementações em IBM Cloud® para utilizar a ferramenta de mineração de dados IBM SPSS Modeler, para aplicar os conhecimentos de MVS.

1 ESTADO DA ARTE

O motor trifásico de indução é um dos principais componentes utilizados na indústria. Isso ocorre devido a alta performance, baixo custo e confiabilidade para gerar energia mecânica a partir da energia elétrica [1]. Tais motores são utilizados em várias áreas da indústria devido a flexibilidade de aplicação, sendo encontrado desde aplicações domésticas até aplicações de alta potência e prioridade na indústria. Entretanto, apesar da alta confiabilidade, o motor trifásico de indução pode ser exposto a diferentes condições de falhas e tais falhas podem levar o motor a um colapso e consequentemente a uma parada não planejada na produção.

A partir disso, é importante que a manutenção esteja continuamente monitorando o motor para que falhas sejam detectadas ainda em estados iniciais, evitando um colapso. Essa detecção de falhas com antecedência permite que as ações corretivas sejam executadas em tempo hábil.

A primeira solução encontrada por engenheiros de manutenção para o monitoramento de motores de indução foi a utilização de relés eletromecânicos que protegiam o motor contra falhas [1]. Entretanto, essa solução se mostrou não eficiente devido a características do relé como lentidão na operação, significativo consumo de energia e necessitam de manutenção periódica devido às partes mecânicas envolvidas.

A manutenção preditiva, cujo conceito se traduz no acompanhamento periódico de equipamentos ou máquinas, por meio de dados coletados através de monitoração ou inspeções. A análise tem como premissa delinear intervenções nas máquinas, focando em indicadores dos próprios equipamentos [54].

Almeida e Santos [55], destacam que com manutenção preditiva, é possível controlar com maior eficácia a necessidade de serviços de manutenção do equipamento, aumenta-se o tempo de disponibilidade dos equipamentos, reduzindo paradas de emergência, aumentando o aproveitamento da vida útil e a confiabilidade do desempenho. As técnicas mais utilizadas para manutenção preditiva são: análise de vibração, ultrassom, termografia, análise de espectros de tensão e corrente elétrica entre outras técnicas de análise não destrutivas.

Com o surgimento da tecnologia dos semicondutores a manutenção preditiva obteve uma melhoria na capacidade de proteger o motor trifásico de indução. A partir desse momento, os relés eletromecânicos foram substituídos por relés de estado sólido que são mais eficientes nos quesitos citados anteriormente e ainda menor custo de fabricação e maior confiabilidade. O

desenvolvimento da tecnologia de microprocessadores no final dos anos 1970 permitiu que fossem utilizados junto aos relés para proteção de motores de indução [2] e a partir desse momento a lógica para a proteção passou a ser implementada através de softwares.

Os recentes desenvolvimentos nos softwares baseados em sistemas inteligentes fizeram com que engenheiros passassem a utiliza-los no diagnóstico de falhas de componentes de sistemas elétricos de potência como o motor de indução. [3]. Os sistemas baseados em técnicas de inteligência artificial podem substituir um humano, ao prover as informações necessárias sobre a performance de determinado sistema. Algumas dessas técnicas são Redes Neurais Artificiais, Lógica Fuzzy e Máquina de Vetores de Suporte (*support vector machine*).

A aplicação da metodologia de MVS vem sendo mais utilizadas para compor sistemas de detecção de falha devido à boa capacidade de generalização e alta taxa de acerto.

Através de uma pesquisa apresentada sobre a detecção de falhas em máquinas, Widodo e Yang [5], utilizando Máquinas de Vetores de Suporte, concluíram que MVS é a técnica mais promissora para diagnóstico de falha e que necessita de trabalhos amplos para a aplicação da ferramenta para o monitoramento da condição e chegar ao diagnóstico de falhas em motores de indução. Brun e Ernst, também encontraram bons resultados em algumas aplicações de MVS.

A maior eficiência na fase de treino obtida com o modelo MVS em relação ao modelo de RNA foi encontrada por Avci [6]. A fase de treino consiste na correlação entre medidas obtidas na aquisição de dados e as correspondentes falhas, e em termos práticos, dependendo da variação da severidade de falha, se não provido de suficientes padrões de falha o modelo poderá não correlacionar de maneira certa um dado de aquisição à uma falha especifica gerando um falso diagnostico. MVS é regido por teorias de aprendizado estatístico onde o modelo tenta desenvolver um sistema de classificação confiável que seja capaz de treinar em menos tempo.

Existem estudos propostos que obtiveram bons resultados com o objetivo de fazer o diagnóstico de um motor trifásico de indução. Podem ser citados trabalhos como [4], em que uma Rede Neural Artificial foi treinada com sinais de corrente e de voltagem para detectar falhas em motores de indução onde somente falhas elétricas foram analisadas.

Esta dissertação propõe uma aplicação prática de MVS à identificação de falhas mecânicas e elétricas externas a um motor trifásico de indução implementada na infraestrutura de algoritmos na nuvem. A metodologia proposta busca captar as falhas através de um acelerômetro para medir vibrações e um sensor de corrente não invasivo para captar a corrente no estator. MVS será

implementado na IBM Cloud® através da utilização da ferramenta de mineração de dados SPSS Modeler, que executará a parte de processamento dos sinais obtidos.

1.1 O Motor Trifásico de Indução

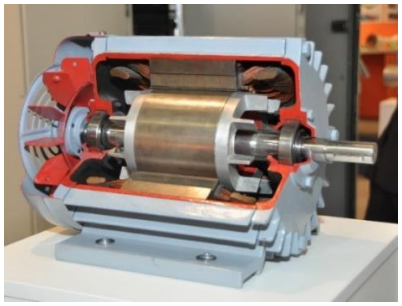
A minimização dos problemas em máquinas elétricas requer compreensão dos padrões de projeto, construção e instalação das máquinas. O funcionamento depende da correta instalação e da implementação das rotinas operacionais apropriadas, além de procedimentos de manutenção acertados, para evitar paradas inadequadas ou desnecessárias [7].

A base de qualquer método de detecção confiável passa pela compreensão do comportamento elétrico e eletromagnético da máquina sujeita às condições diversas. Os mais frequentes tipos de problemas que podem ser esperados em um motor trifásico de indução são relacionados à alimentação do motor, estrutura mecânica, enrolamentos estatóricos e rotóricos. Todo problema modifica as variáveis elétricas, magnéticas e mecânicas do motor [9], desta forma, torna-se fundamental o conhecimento sobre a estrutura e características de funcionamento sobre máquinas de indução.

1.1.1 Princípios de Funcionamento e Características Construtivas

A Figura 1 ilustra a vista em corte de um motor trifásico de indução, para mostrar os componentes principais. Os princípios de funcionamento das máquinas de indução estão firmados no conhecimento sobre o estator que é a parte fixa e o rotor é a parte móvel.

Figura 1 - Vista em corte de um motor trifásico de indução.

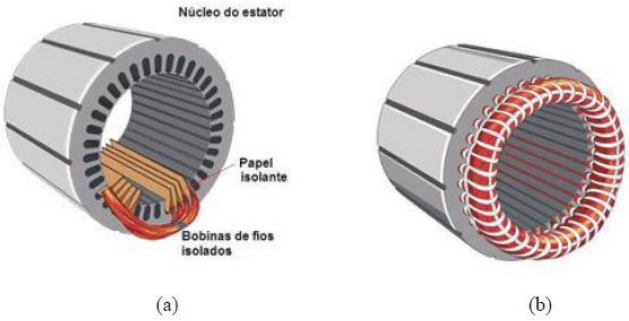


Fonte: [53].

O núcleo do estator é construído com chapas de material magnético e recebe o enrolamento de campo, cujas espiras são colocadas em ranhuras (Figura 2).

A maneira como esse enrolamento é construído determina o número de polos do motor, entre outras características operacionais. As pontas (terminais) são estendidas até uma caixa de terminais, onde pode ser realizada a conexão com a rede elétrica de alimentação. [10]

Figura 2 - Enrolamento de campo de um motor de indução: (a) execução dos enrolamentos; (b) núcleo.

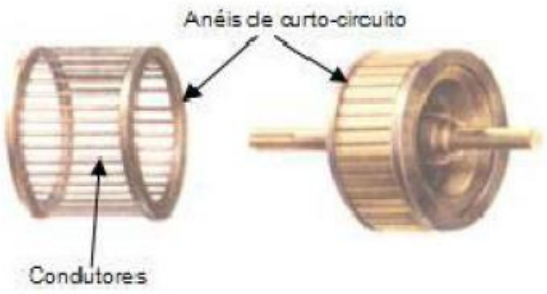


Fonte: [10].

Quanto ao rotor do motor de indução, pode ser de dois tipos:

O rotor em gaiola de esquilo ou rotor em curto: Os condutores são constituídos por barras de cobre ou alumínio colocadas entre as ranhuras.

Figura 3 - Estrutura de rotor tipo gaiola de esquilo.

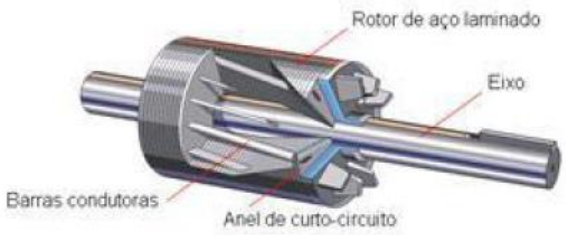


Fonte: [10].

Nas duas extremidades das barras existem dois anéis curto-circuitando todas as barras. Esta estrutura é semelhante a uma gaiola de esquilo. Este tipo de rotor é mais empregado, devido a características como baixo custo e manutenção.

Já o rotor bobinado ou rotor de anéis, tem custo superior devido a características como complexidade de fabricação (Figura 4).

Figura 4 - Estrutura do rotor bobinado.



Fonte: [10].

O fluxo magnético girante aparece no estator devido às correntes alternadas circulantes nas bobinas do estator. Este fluxo magnético do estator se desloca em relação ao rotor, cortando as barras do rotor induzindo tensões (Lei de Faraday e Lenz) que farão circular correntes também alternadas no rotor. Como as correntes do rotor têm polaridades contrárias do estator, por definição, cria-se também no rotor um campo magnético girante que será atraído e arrastado pelo campo girante do estator [10]. De modo que é desenvolvido um conjugado mecânico no rotor levando o mesmo a girar.

A velocidade do rotor (n) é sempre menor que a velocidade do campo girante do estator (n_s), também chamada velocidade síncrona. Se o rotor fosse levado até a velocidade síncrona ($n = n_s$), não haveria mais velocidade relativa entre os campos girantes do estator e do rotor e consequentemente a tensão induzida cessaria, não haveria mais corrente no rotor, o conjugado mecânico diminuiria e o rotor automaticamente perderia velocidade ($n<n_s$) [10].

A diferença entre a velocidade síncrona e a velocidade do rotor é chamada de velocidade de escorregamento (n_e):

$$n_e = n_s - n \tag{1.1}$$

Em que n_s velocidade síncrona e n velocidade do rotor

Assim, o escorregamento s é definido por:

$$s(\%) = \left(\frac{n_s - n}{n_s}\right) \cdot 100 \text{ , sendo } n = (1 - s) \cdot n_s[RPM] \tag{1.2}$$

O valor de s para motores de indução de gaiola é de 2% a 5% de acordo com [10]. A velocidade síncrona (n_s) é dado por:

$$n_s = \frac{120 \cdot f}{p} [RPM] \quad (1.3)$$

Em que:

f = frequência em hertz

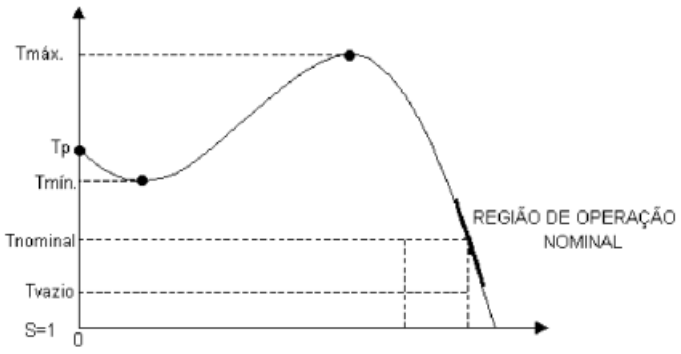
p = número de pólos.

Podemos variar a velocidade n_s e, consequentemente, n variando-se o número de pólos p (alterando-se construtivamente as bobinas do estator).

À medida que é aumentada a carga no eixo do motor, a velocidade diminui até um ponto onde o conjugado desenvolvido é máximo.

A Figura 5 ilustra a relação de desempenho através do gráfico. Para tanto, qualquer acréscimo de carga além do ponto T_{max} faz com a velocidade caia bruscamente, podendo em algumas situações travar o rotor. [10]

Figura 5 - Gráfico de conjugado x velocidade.



Fonte: [10].

Em que:

T_p - Conjugado de partida: é o conjugado com o motor travado.

T_{min} - Conjugado mínimo.

Tmáx - Conjugado máximo.

T_{nom}- Conjugado nominal, normalmente T_{nom} ocorre com s entre 2% e 5%.

T_{vazio} - Conjugado para o motor operando sem carga.

As características técnicas do motor utilizado neste trabalho podem ser revisadas no apêndice 3.

1.2 Falhas comuns em Motores de Indução

As condições diversas as quais está sujeito o motor de indução, o conduz a diferentes maneiras de falhas que se propagam no motor, sendo classificadas como internas ou externas. Neste trabalho serão abordadas causas de falhas externas, que podem se classificar da seguinte forma [10]:

1. Desbalanceamento no eixo
2. Fase única
3. Tensão de alimentação desequilibrada
4. Rotor bloqueado
5. Reversão de fase
6. Falta de aterramento
7. Desnível na base

Segundo Bloch [57], "falha" pode ser definida como qualquer mudança em um componente da máquina, que a deixa incapacitada para desempenhar a sua função de modo satisfatório, levando a parte ou o componente à condição de inconfiável ou inseguro, para continuar em uso.

Independentemente da classe em que seja classificada a falha, Nandi [58], ratifica que, seja de forma isolada ou em combinação de duas ou mais, as falhas podem proporcionar, entre outros, os seguintes sintomas: tensões e correntes de linha desbalanceadas; aumento de vibrações; queda do torque médio; aumento das perdas e redução do rendimento; aquecimento excessivo.

A utilização de sistemas de proteção e monitoramento de máquinas vem sendo amplamente difundida, objetivando monitorar e até mesmo desconectar o motor em caso de falha. Porém o avanço das aplicações de monitoramento e diagnóstico de falha vem se modificando de acordo com a evolução das aplicações.

1.3 Contexto atual de detecção de falhas no motor trifásico de indução

Muitas aplicações vêm sendo implementadas para testes e validações de novas variações em metodologias de sistemas de monitoramento da condição *online* e *offline.* O contexto atual que exige dos ativos industriais alta disponibilidade, de modo a necessitar de um controle

preciso de mapeamento de paradas para manutenção. Podendo ainda avaliar condições de falhas incipientes e evitar colapsos onerosos à produção e a estrutura do motor.

Com o propósito de detectar e diagnosticar as falhas nos motores de indução, muitos métodos foram desenvolvidos até então, envolvendo diferentes campos da ciência e tecnologia [13]. Destacamos aqui, os seguintes:

1.Monitoramento por vibração.
2.Monitoramento por emissão acústica.
3.Monitoramento do campo magnético.
5.Monitoramento por temperatura.
6.Monitoramento de flutuação de velocidade.
7.Monitoramento da assinatura elétrica.
8.Análise química.

De acordo com Isermann [14], o diagnóstico de falhas é baseado na observação analítica e heurística dos sintomas e no conhecimento heurístico do processo, os quais podem ser extraídos a partir da instrumentação das máquinas ou a partir da modelagem do próprio processo. Neste contexto, técnicas baseadas em Inteligência Artificial se mostram promissoras no campo do diagnóstico de falhas, entre as quais destacam-se: MVS (Máquinas de vetores de suporte*)*, métodos de elementos finitos, redes neurais artificiais, lógica fuzzy entre outros. Conforme visto no Estado da arte, existem diversos trabalhos que contemplam essas técnicas, seja de forma isolada ou através de combinações entre duas ou mais técnicas.

Atualmente a tendência de desenvolvimento de aplicações que fazem parte da *web* e estão conectadas a servidores que disponibilizam informações e conectividade em tempo real vem ganhando espaço de forma exponencial. A utilização de métodos estatísticos diversos aliados aos conhecimentos de AM (Aprendizado de máquina- *Machine Learn*) fazem a união necessária para melhorar o contexto de detecção e diagnóstico de falhas. Com base nisso, este trabalho utiliza conhecimentos de AM para solucionar questões reais de mineração de dados.

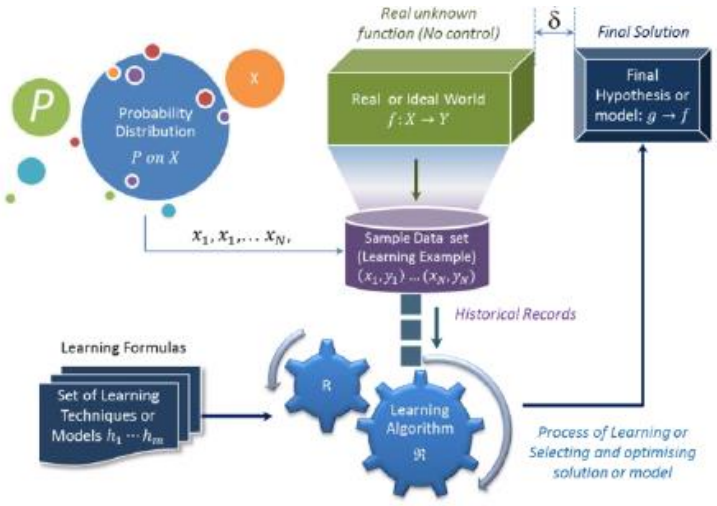
2 CONCEITOS DA COMPUTAÇÃO

2.1 Aprendizado de Máquina

A essência do Aprendizado de Máquina (AM) é um processo automático de reconhecimento de padrões. O principal objetivo do aprendizado de máquina é criar sistemas que possam executar ou exceder a competência no nível humano ao lidar com muitas tarefas ou problemas complexos. O aprendizado de máquina faz parte da Inteligência Artificial (IA). Durante o início da era de pesquisa da IA, o objetivo era construir robôs e simular atividades humanas. Posteriormente, a aplicação da IA foi generalizada para resolver problemas gerais por uma máquina. A solução popular era alimentar um computador com algoritmos (ou uma sequência de instruções) para transformar os dados de entrada em respostas. [15]

No entanto, ainda segundo [15] para muitos problemas, não é tão simples encontrar facilmente algoritmos adequados, por exemplo, o reconhecimento da caligrafia humana. Não é simples transformar a entrada da letra de escrita manual na saída da letra reconhecida padrão. Uma alternativa é aprender com os dados de diferentes caligrafias. Isso significa que, com uma tentativa, há um grande erro, mas agregando-se muitas tentativas, o erro será reduzido para um nível ou convergência aceitável. A Figura 6 ilustra um exemplo típico de processo de aprendizado de máquina.

Figura 6 - Processo de Aprendizado de Máquina.



Fonte: [15].

Desde o final dos anos 90 o volume de dados se tornou cada vez maior. Uma questão lógica é como lidar com esses grandes volumes de dados e como encontrar padrões úteis ou significativos a partir de um volume crescente de dados. Isso leva à “descoberta de conhecimento no banco de dados” (ou KDD- *knowledge discovery in database*), que também é chamada de mineração de dados. Em [15] [16] foi abordado a investigação em diferentes bancos de dados, afim de descobrir o conhecimento para auxiliar tomadas de decisão.

Para descobrir padrões significativos de um conjunto massivo de dados, a estatística é a ferramenta vital para agregar valor à amostragem, modelagem, análise, interpretação e apresentação de dados, assim como [17] indicou que a mineração de dados tem uma conexão inerente às estatísticas. Isso leva à convergência do sistema de mineração de dados e do sistema especialista difuso sob o grande guarda-chuva do aprendizado de máquina. Do ponto de vista da evolução do aprendizado de máquina, a teoria estatística ou a modelagem de probabilidade mudaram a disciplina de IA de sistemas especialistas baseados em regras ou aprendizado de esquema na gravação, para uma metodologia orientada a dados, que é resolver o problema de incerteza com probabilidade dos parâmetros de um modelo. Nessa perspectiva, as estatísticas foram incorporadas ao AM.

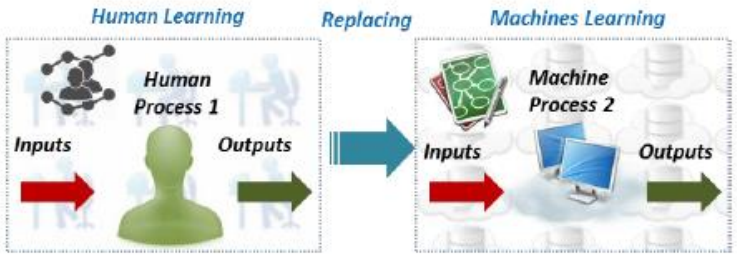
Desta forma, tem havido muitas definições funcionais de AM. Diferentes autores enfatizariam diferentes aspectos do aprendizado de máquina, como processo, aplicativo e utilidade. Por exemplo, a definição de Arthur Samuel [18], enfatizou o "aprendizado automático" de AM. Tom Mitchell [19] descreveu todos os componentes do processo de AM. Kevin [20] e Noan [21], enfatizaram a função do reconhecimento de padrões, sendo que Noan argumentou que AM poderia transformar pensamentos abstratos em operação física. No resumo de várias definições, encontram-se alguns dos ingredientes essenciais e comuns dessas definições de AM:

- Treinar a máquina para aprender automaticamente e melhorar os resultados à medida que obtém mais dados.
- Descobrir ou reconhecer padrões com dados de entrada.
- Executar predições através de entradas desconhecidas.
- A máquina irá adquirir conhecimento diretamente dos dados e resolverá problemas propostos.

De acordo com esses elementos, é possível descobrir que, fundamentalmente, o AM é uma consequência da interseção entre ciência da computação e estatística, visando aprender automaticamente a reconhecer padrões complexos e a tomar decisões inteligentes com base em

conjuntos de dados existentes. O objetivo final do AM é construir sistemas com nível de competência humana, na execução de tarefas complexas (Figura 7).

Figura 7 - Interpretação homem/máquina.



Fonte:[15].

O aprendizado de máquina permite que os modelos treinem em conjuntos de dados antes de serem implantados. Alguns modelos de aprendizado de máquina são online e contínuos. Esse processo iterativo de modelos online leva a uma melhoria nos tipos de associações feitas entre elementos de dados. Devido à complexidade e tamanho, esses padrões e associações podem ser facilmente ignorados pela observação humana. Depois que um modelo é treinado, ele pode ser usado em tempo real para aprender com os dados. As melhorias na precisão são resultado do processo de treinamento e automação que fazem parte do aprendizado de máquina [23].

Técnicas de aprendizado de máquina são necessárias para melhorar a precisão dos modelos preditivos. Dependendo da natureza do problema que está sendo resolvido, existem diferentes abordagens baseadas no tipo e no volume dos dados.

2.1.1 Aprendizado supervisionado

O aprendizado supervisionado começa com um conjunto estabelecido de dados e um certo entendimento de como esses dados são classificados. O aprendizado supervisionado tem como objetivo encontrar padrões nos dados que podem ser aplicados a um processo de análise. Esses dados rotularam recursos que definem o significado dos dados. Por exemplo, é possível criar um aplicativo de aprendizado de máquina que distinga entre milhões de animais, com base em imagens e descrições escritas [23].

2.1.2 Aprendizagem não supervisionada

O aprendizado não supervisionado é usado quando o problema exige uma enorme quantidade de dados não rotulados. Por exemplo, aplicativos de mídia social, como *Twitter*, *Instagram*, o qual todos têm grandes quantidades de dados não rotulados. Compreender o significado por trás desses dados requer algoritmos que classificam os dados com base nos padrões ou *clusters* encontrados. O aprendizado não supervisionado conduz um processo iterativo, analisando dados sem intervenção humana. Sendo usado com a tecnologia de detecção de *spam* de *e-mail* [23].

2.1.3 Aprendizagem por reforço

O aprendizado por reforço é um modelo de aprendizado comportamental. O algoritmo recebe *feedback* da análise de dados, guiando o usuário para o melhor resultado. O aprendizado por reforço difere de outros tipos de aprendizado supervisionado, porque o sistema não é treinado com o conjunto de dados de amostra. Em vez disso, o sistema aprende por tentativa e erro. Portanto, uma sequência de decisões bem-sucedidas resultará no reforço do processo em que estiver aplicado [23].

2.1.4 Aprendizagem profunda (*Deep Learn*)

O aprendizado profundo é um método específico de AM que incorpora redes neurais em camadas sucessivas para aprender com os dados de maneira iterativa. O aprendizado profundo é especialmente útil quando se está tentando aprender padrões a partir de dados não estruturados. As redes neurais complexas de aprendizado profundo são projetadas para imitar como o cérebro humano funciona, para que os computadores possam ser treinados para lidar com abstrações e problemas mal definidos.

2.2 Computação na Nuvem

O termo nuvem é usado como um termo genérico/metafórico frente a um conjunto virtualizado de *hardware* e recursos, tornando-se uma abstração para a infraestrutura complexa que

que possui. Uma boa definição de computação em nuvem vem do Instituto Nacional de Padrões e Tecnologia (NIST). A definição do NIST essencialmente diz que:

"A computação em nuvem é um modelo para permitir acesso conveniente e sob demanda da rede a um conjunto compartilhado de recursos de computação configuráveis que podem ser rapidamente provisionados e lançado com esforço mínimo de gerenciamento ou interação com o provedor de serviços."

Exemplos de recursos de computação na nuvem incluem:

- Redes
- Servidores
- Armazenamento
- Aplicações
- Serviços

A computação na nuvem como modelo de implantação está substituindo uma abordagem mais antiga, na qual cada aplicativo com o qual um usuário interage tem serviços próprios personalizados, rede, dados armazenamento e poder de computação [24].

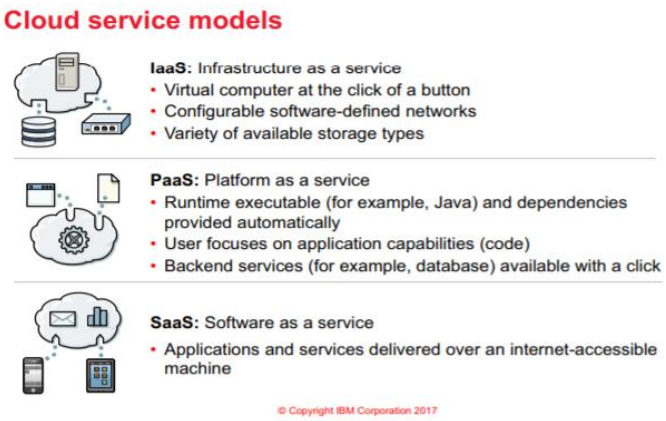
A capacidade de reutilizar e adaptar novamente o *hardware* rapidamente, além de hospedar vários aplicativos e sistemas em um único conjunto de *hardware* de maneira isolada, são algumas das principais características que impulsionam a adoção da computação em nuvem [23][24].

Uma contribuição importante para o crescimento da computação em nuvem acontece através da necessidade atual da velocidade da demanda de entrega de serviços, que devem ser entregues rapidamente. Os desenvolvedores são pressionados a colocar seu produto no mercado assim que possível. Necessitando de *feedbacks* rapidamente e, em seguida, repetem a ideia de tornar o produto melhor e mais rápido.[24]

A nuvem torna os recursos de hardware prontamente disponíveis e rápidos de configurar, desta forma reduzindo o tempo necessário para que seja mostrada uma versão funcional de produtos. Além disso, permite a reutilização dos mesmos recursos para vários projetos sucessivos, o que é mais econômico. Um outro fator que contribui para o crescimento da computação em nuvem é que os desenvolvedores podem usar diferentes linguagens de programação.

Na Figura 8 é ilustrada modelos de serviço oferecidos na nuvem:

Figura 8 - Modelos de serviços na nuvem.



Fonte: Adaptado de [24].

- Infraestrutura como Serviço (IaaS), um conjunto de ativos físicos, como servidores, dispositivos de rede e discos de armazenamento, são oferecidos como dedicados e particulares acessível aos consumidores. Os serviços neste modelo suportam a infraestrutura de aplicativos.

- Plataforma como serviço (PaaS) é um modelo de serviço em nuvem no qual a estrutura de aplicativos e o tempo de execução é uma entidade virtualizada, de autoatendimento. O objetivo do PaaS é permitir o desenvolvedor ou equipe para se concentrar nas funções, código e dados de negócios do aplicativo, em vez de se preocupar com infraestrutura.

- Software como Serviço (SaaS) oferece aplicativos sob demanda aos usuários através do Internet, em oposição aos aplicativos de desktop. Exemplos de aplicativos SaaS incluem Salesforce.com, Google Apps entre outros.

Existem vários provedores de serviços de computação em nuvem disponível na indústria. Algumas das empresas líderes estão listadas abaixo, conforme [26]:

- IBM Cloud - lançado em 2011.
- Amazon Web Services - lançado em 2006.
- Microsoft Azure - lançado em 2010.
- Google Cloud Platform - lançado em 2008.
- Alibaba Cloud - lançado em 2009.

A escolha da IBM Cloud® deu-se a partir da vasta documentação disponibilizada para utilização dos serviços além de acesso rápido à utilização das ferramentas de mineração de dados, através de Watson Studio e SPSS Modeler. Em [23][24], a documentação é disponibilizada com textos em português, sendo desta forma atrativa à utilização.

2.3 IBM Cloud®

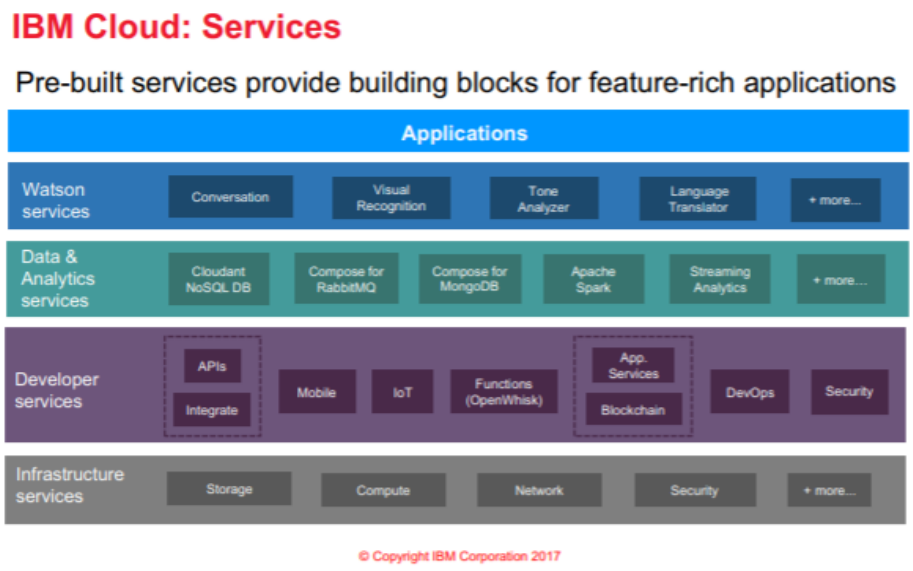
A IBM Cloud® é uma plataforma de computação em nuvem aberta que combina a plataforma como serviço (PaaS) com infraestrutura como serviço (IaaS) e inclui um catálogo de diversos serviços, que podem ser usados para criar e implantar rapidamente aplicativos ou infraestruturas diversas [23].

Na forma PaaS, fornece aos desenvolvedores acesso ao *software* IBM® para integração, segurança e outras funções-chave. Como IaaS, ele permite que os desenvolvedores controlem detalhadamente a infraestrutura na qual os aplicativos são implantados. Os desenvolvedores podem implantar servidores de alto desempenho, servidores virtuais e contêineres, utilizando a infraestrutura e sistemas de *hardware* nos locais dos data centers da IBM Cloud® em todo o mundo [23] [24].

2.4 IBM Cloud Services

Através da Figura 9 é possível observar a gama de divisões dentre aplicações em IBM Cloud®:

Figura 9 - Serviços IBM Cloud.



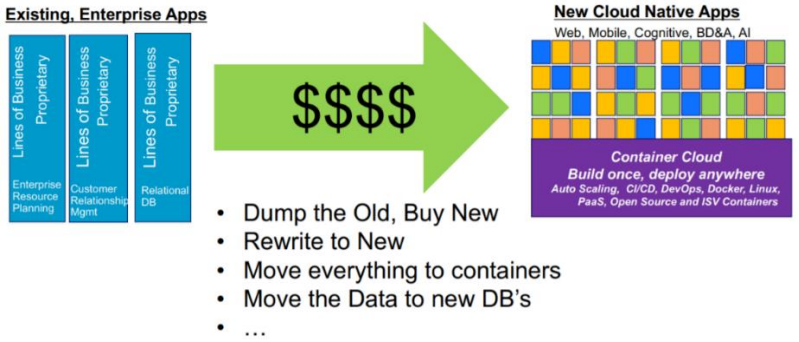
Fonte: Adaptado de [24].

A IBM Cloud® fornece uma ampla gama de serviços pré-criados (da IBM e de terceiros) que pode ser usado frente a diferentes necessidades de tratamento estatístico.

Além disso, pode-se citar a criação, gerenciamento, e execução de APIS, uso da infraestrutura de *back-end* para criar e testar aplicações. Comunicando-se com dispositivos, sensores e *gateways* conectados caso sejam implementados. Várias opções de serviços pré-desenvolvidos são oferecidos, como Blockchain. Message Hub, WebSphere Application Server, Business Rules, Watson Studio entre outros serviços na nuvem totalizando mais de 190 serviços disponibilizados em IBM Cloud®.[24]

Recomendações comuns para transformação atual das aplicações, na Figura 10.

Figura 10 - Contexto da atualidade envolvendo aplicações.



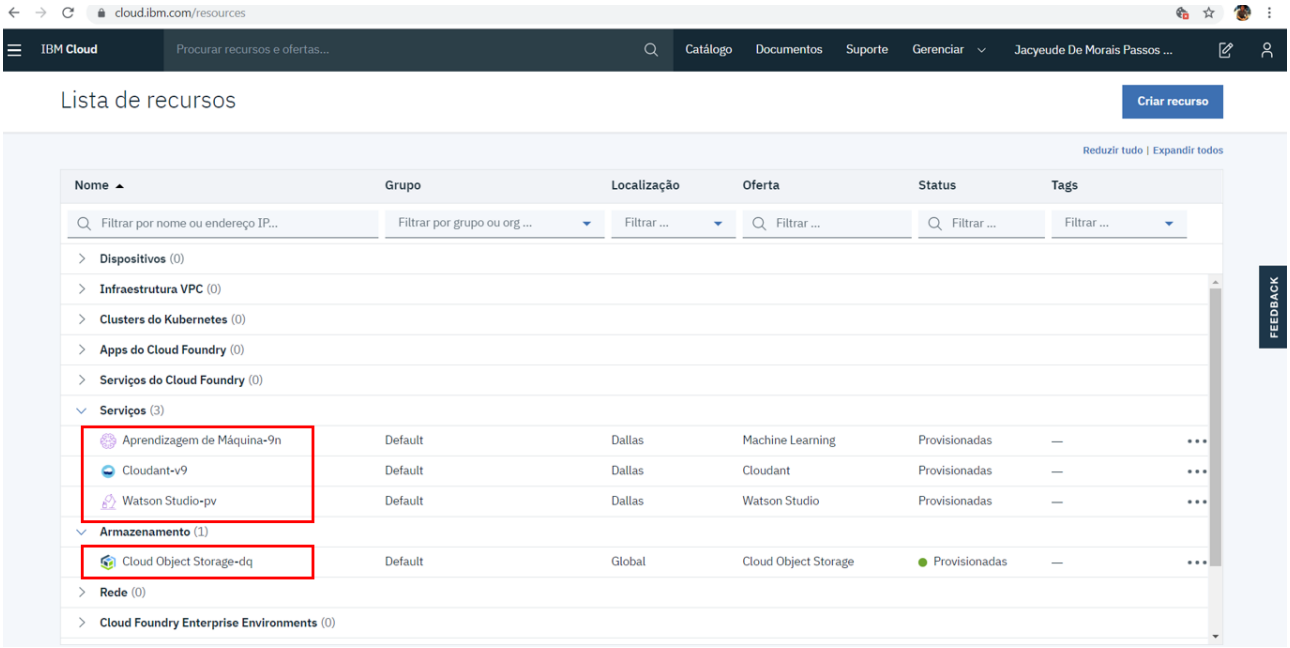
Fonte: Adaptado de IBM Cloud Private Docs.

A Figura 10 ilustra bem o que vem acontecendo com todas as aplicações que buscam inovar e fazer parte dos avanços em quesitos de desenvolvimento junto a investimento na migração para a computação na nuvem, como já mostrado anteriormente.

As documentações disponibilizadas pela IBM Cloud® fornecem informações base sobre as aplicações, sendo de fundamental importância ter conhecimento prévio para a seleção correta das ferramentas a serem provisionadas.

Dentre a vasta biblioteca de desenvolvimento oferecida pela IBM Cloud®, para o desenvolvimento dessa dissertação foi necessário provisionar os serviços de Watson Studio, Cloud Object Storage e Watson Machine Learn, descritos a seguir, com o objetivo de dar subsídios para a utilização de MVS dentro do fluxo de IBM SPSS Modeler. Figura 11

Figura 11 - Lista de recursos provisionados.



Fonte: O próprio autor.

2.4.1 Cloud Object Storage

O Cloud Object Storage (CSO) é um conceito moderno de tecnologia de armazenamento e uma progressão lógica do armazenamento de blocos e arquivos. O armazenamento de objetos é redondo desde o final dos anos 90, mas ganhou aceitação e sucesso no mercado nos últimos 10 anos. O IBM COS usa um Algoritmo de Dispersão de Informações (IDA) para dividir os arquivos em partes que são distribuídas aos nós de armazenamento [24]. Anteriormente o fornecimento de segurança de acesso exigia uma combinação de tecnologias, esquemas de segurança complexos e envolvimento humano significativo no gerenciamento dessas áreas.

O Cloud Storage Object (CSO) foi criado para superar vários problemas, dentre os quais pode-se citar limitações estruturais e de segurança em vários níveis dos dados.

Os conceitos de armazenamento de dados do objeto incluem as três seguintes construções principais, de acordo com [24]:

- Dados são compreendidos como: formato de texto, binários, planilhas, multimídia ou qualquer outro conteúdo gerado por humanos ou máquinas.

- Metadados: são os dados sobre os dados. Inclui alguns atributos predefinidos, como hora e tamanho do upload. O CSO permite que os usuários incluam metadados personalizados contendo informação em pares de chave e valor. Um aspecto exclusivo da manipulação de metadados nos sistemas de CSO é que os metadados são armazenados com o objeto.
- Um identificador universalmente exclusivo (*universally unique identifier*-UUID): esse ID é atribuído a todos os objetos em um sistema CSO. O UUID permite diferenciar objetos entre si para encontrar os dados sem precisar saber a unidade física exata, matriz de onde estão os dados.

A integridade, escalabilidade, gerenciamento, disponibilidade e segurança são essenciais para os dados, porém as soluções de armazenamento tradicionais têm dificuldade em lidar com a escala na qual os dados estão crescendo atualmente. [24]

Sendo desta forma o Cloud Storage Object oferecido pela IBM Cloud® que irá armazenar os dados deste trabalho, tornando-os disponíveis para o fluxo das análises subsequentes dentro da infraestrutura da nuvem.

2.4.2 Watson Studio e Watson Machine Learn

O Watson Studio, fornece o ambiente e as ferramentas para análises estatísticas em conjunto de dados. Oferecendo funções de visualização, limpeza, experimentos que aplicam IA, modelagem dados, podendo ainda criar e treinar modelos de aprendizado de máquina [23].

De acordo com [27] O Watson representa um avanço impressionante no design e análise de sistemas. Ele executa a tecnologia DeepQA(arquitetura probabilística paralela maciça baseada em evidências) da IBM®, um novo tipo de capacidade analítica que pode executar tarefas simultâneas em segundos para fornecer respostas precisas a diferentes questões. Ativado pela tecnologia dos processadores IBM POWER7, o Watson é um exemplo das cargas de trabalho de análise complexa que estão se tornando cada vez mais comuns e essenciais para modelos de negócios no ambiente atual de grande fluxo de dados.

O Watson se aproveita do desempenho de processamento paralelo maciço dos processadores POWER7 para executar as milhares de tarefas da DeepQA simultaneamente em núcleos de processadores individuais. Cada um dos 90 servidores IBM Power7 em cluster do Watson

conta com 32 núcleos POWER7, executados a 3.55 GHz. Executando o sistema operacional Linux®, os servidores são abrigados em 10 racks, juntamente com os nós e *hubs* de comunicação associados. O sistema tem um total combinado de 16 terabytes de memória e pode operar a mais de 80 teraflops (trilhões de operações por segundo).[27]

A capacidade do Watson para entender e processar rapidamente informações para encontrar respostas precisas para perguntas complexas, guarda um potencial enorme para transformar a forma na qual os computadores podem ajudar as pessoas a realizar tarefas em diferentes âmbitos de negócios.

Os colaboradores têm acesso às entradas de dados em *Data Assets*, podendo carrega-los automaticamente, tendo disposição posterior de uma importante ferramenta na aplicação, Data Refinery, onde podem ser criados fluxos de tratamentos dos dados incialmente dispostos, podendo retirar espaços vazios, ou realizar visualizações gráficas dos dados de entrada. Tendo opções de criação, treinamento e testes de modelos de aprendizado de máquina e aprendizado profundo.

Algumas ferramentas exigem complementos, havendo necessidade de fazer provisionamento das ferramentas: Cloud Object Stourage e Watson Machine Learn, para dar os subsídios necessários para o upload de dados e análises estatísticas para este trabalho. A seleção de ferramentas complementares desta dissertação se deu através de considerações sobre tipos de dados, tipos de tarefas e quanto de automação poderia ser implementado através das ferramentas disponíveis, dentre os quais pode-se citar: Jupyter, RStudio IDE ou SPSS Modeler.

Estas ferramentais diferenciam-se por serem melhor aplicadas a contextos de dados diferentes como: dados tabulares ou relacionais, dados textuais, dados de imagem.

A Tabela 1, disposta pela documentação de IBM Cloud® mostra as tarefas que são executadas pelas ferramentas:

Tabela 1 - Ferramentas e atribuições em IBM Cloud®.

Ferramentas para dados tabulares ou relacionais por tarefa:

Ferramenta	Tipo de ferramenta	Preparar dados	Analisar dados	Construir modelos
Editor de notebooks Jupyter	Editor de código	√	√	√
RStudio	Editor de código	√	√	√
Refinaria de Dados	Tela gráfica	√	√	
Editor de fluxo de fluxos	Tela gráfica	√	√	
Editor de painel	Tela gráfica		√	
SPSS Modeler	Tela gráfica	√	√	√
Modelador Spark MLlib	Tela gráfica	√		√
Construtor de Modelo de Otimização de Decisão	Editor de tela e código gráfico	√		√
AutoAI	Construtor automático	√		√

Fonte: Adaptado de [23].

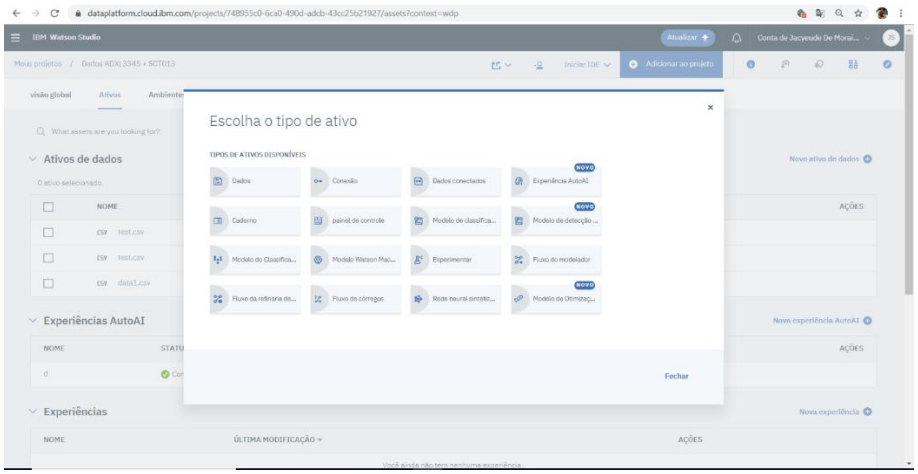
O ambiente de Watson Studio conta com ferramentas gráficas para projetar, treinar, implantar e gerenciar modelos com alguns dos serviços do Watson Aprendizagem de Máquina:

- As experiências da *AutoAI* processam automaticamente os dados, selecionam o melhor estimador para os dados e, em seguida, geram candidatos a modelos para que possam ser revisados e comparados.
- O modelador SPSS apresenta uma visualização gráfica do seu modelo enquanto é construído um combinando nós que representam objetos ou ações, realizando implementações de algoritmos para mineração de dados.
- O construtor de modelo *Decision Optimization* orienta na construção e resolução de modelos prescritivos. O modelador de rede neural apresenta uma visualização gráfica do seu modelo enquanto é construído um combinando nós de rede neural).

De acordo com [23], através de Watson Machine Learn, construído em uma plataforma escalável de código aberto baseada nos componentes Kubernetes e Docker, é possível construir modelos analíticos e redes neurais, treinadas com dados próprios, que podem ser implementadas para uso em aplicativos. Com isso a seleção do provisionamento de Watson Machine Learn é de fundamental para este trabalho.

A Figura 12 ilustra os tipos de ativos que podem ser implementados em Watson Studio.

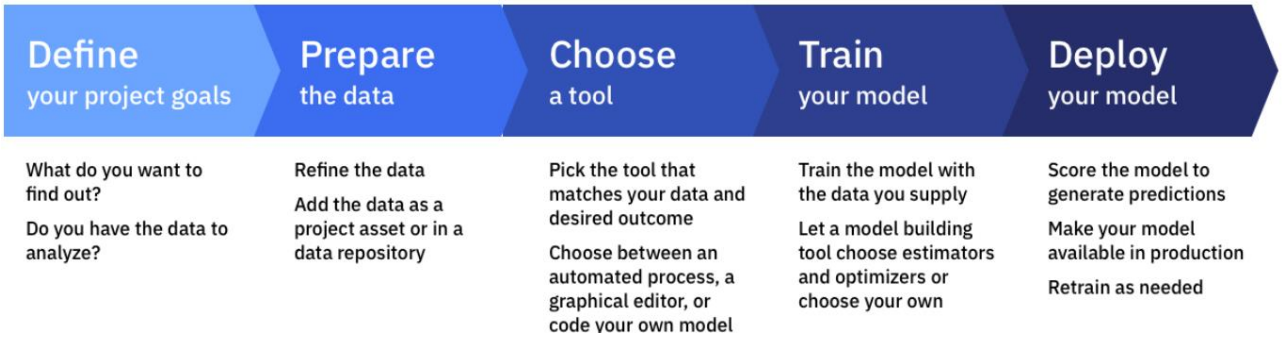
Figura 12 - Ativos para desenvolvimento de aplicações em Watson Studio.



Fonte: O próprio autor.

Abaixo na Figura 13, é ilustrado o fluxo de que ilustra o desenvolvimento de uma aplicação a ser desenvolvida em Watson Studio:

Figura 13 - Desenvolvimento de aplicação.

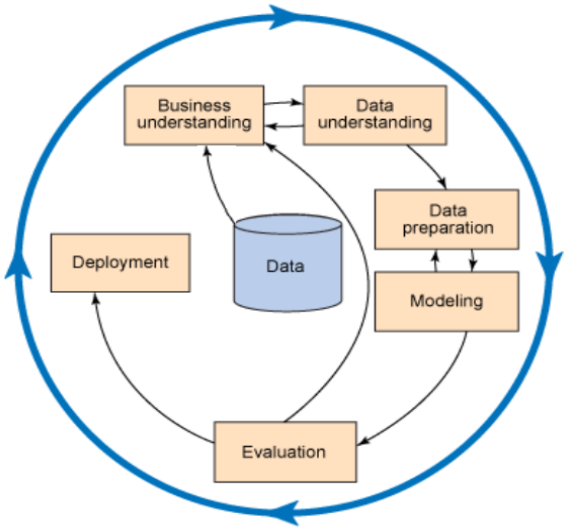


Fonte: Adaptado de [23].

2.4.3 IBM SPSS Modeler

O IBM SPSS Modeler, de acordo com [23], é um conjunto de ferramentas de mineração de dados que permite desenvolver modelos preditivos usando conhecimentos estatísticos. Projetado com base no modelo CRISP-DM. Este é descrito em termos de um modelo de processo hierárquico, consistindo em conjuntos de tarefas descritas em níveis de abstração, Figura 14.

Figura 14 - Ilustração CRISP data mining em seis etapas e como elas estão relacionadas.



Fonte:Adaptado de [25].

É necessário que seja observado o contexto da mineração de dados em tópicos científicos, em que se tornou importante e útil na maioria dos campos com largas *databases*, sendo hábil para extrair conhecimento de uma massa de dados brutos armazenados [25].

A mineração de dados está encontrando os padrões anteriormente “ocultos” em diferentes dados na forma semiautomática [25] descrendo as informações através de métodos e modelos, como modelos analíticos e de classificação, além de apresentar resultados usando diferentes ferramentas disponíveis.

Seu conceito base fundamenta-se na capacidade de sistematizar processos, podendo obter quesitos relevantes a informações e conhecimento a partir de bancos de dados. É indispensável associar a mineração de dados a eficiência e vantagem competitiva, características indispensáveis para o mercado atual de aplicações em diferentes contextos estatísticos, para buscar informações inerentes a cada processo, necessitando de abstrações particulares para eventos diferentes, respeitando um fluxo de ações a ser tomado.[25]

É necessário diferenciar conceitos de dados, informações e conhecimento dentro dos contextos de desenvolvimento. Um dado isolado não possui significado relevante e não conduz a nenhuma compreensão, representa algo que não tem sentido a princípio. Portanto, não tem valor algum para embasar conclusões, muito menos respaldar decisões. A informação é a ordenação e organização dos dados de forma a transmitir significado e compreensão dentro de um determinado contexto, seria o conjunto ou consolidação dos dados de forma a fundamentar para obtenção

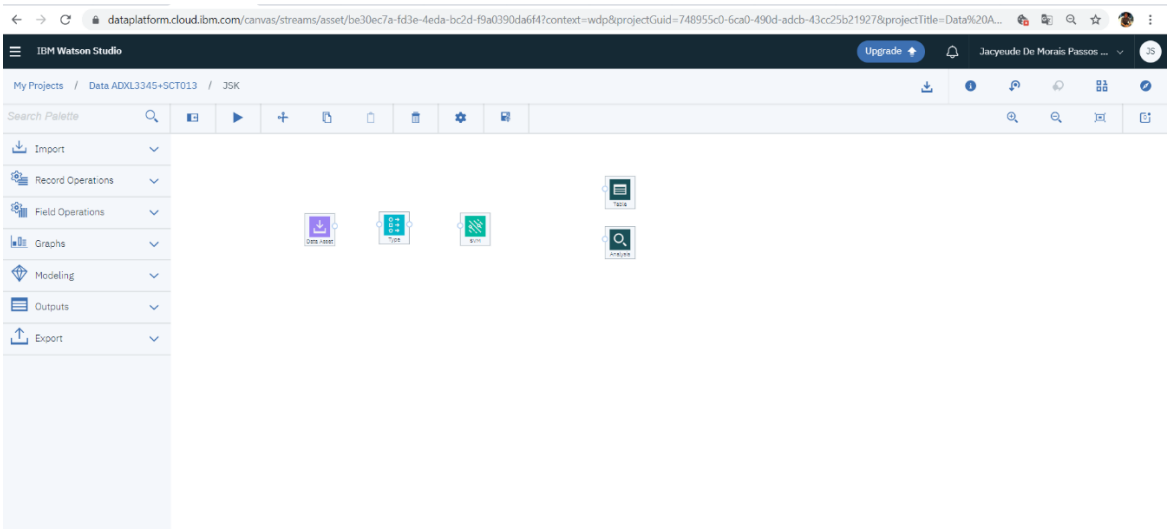
conhecimento. Enquanto conhecimento vai além de informações, pois com ele é possível extrair significados em termos de uma aplicação.[27]

O SPSS Modeler oferece uma variedade de métodos de modelagem retirados do AM, inteligência artificial, além de outros métodos estatísticos [23].

Ele utiliza nós para auxiliar a explorar dados. Vários nós na área de trabalho de sua interface representam diferentes objetos e ações. A aba na parte lateral esquerda da janela do IBM SPSS Modeler contém nas abas principais, conjuntos de nós usados na criação de fluxos, como ilustra a Figura 15.

Existem vários tipos de nós, organizados em campos específicos da aba principal que serão descritos. Conectando os nós, geram-se fluxos que, quando executados, permitem visualizar relacionamentos entre os dados e aplicações. Os fluxos são como *scripts*, sendo possível salvá-los e reutilizá-los com diferentes arquivos de dados.

Figura 15 - Interface SPSS Modeler.



Fonte: O próprio autor.

Com o IBM SPSS Modeler, é possível construir modelos de AM com facilidade de arrastar e soltar. Inicialmente carregando os dados, executando etapas do pré-processamento sendo possível transformá-los, antes de aplicar algoritmos e avaliar o desempenho do modelo preditivo criado.

Através da aba *Import* é possível inserir o banco de dados a que se deseja analisar, ou até mesmo cria-lo para seguir o fluxo de análises. Os conteúdos sobre IBM SPSS Modeler foram obtidos nas documentações de [23] e [29].

Os nós de *Record Operations* são usados para fazer alterações nos dados, compreendendo a etapa de pré-processamento. Essas operações são importantes durante as fases de mineração de dados para compreensão e preparação de dados, pois permitem sejam adaptados às necessidades devidas.

Os modelos de classificação usam os valores de um ou mais campos de entrada para prever o valor de um ou mais campos de saída. Alguns exemplos dessas técnicas são: Árvores de Decisão (algoritmos C&R Tree, QUEST, CHAID e C5.0), Regressão (Algoritmos Lineares, Linear Generalizado e Regressão de Cox), Redes Neurais, Máquinas de Vetores de Suporte- MSV e Redes Bayesianas [23].

Os modelos de classificação, ajudam desenvolvedores a prever um resultado conhecido, como se um cliente irá comprar ou sair ou se uma transação ou encaixar os resultados em um padrão conhecido. As técnicas de modelagem incluem aprendizado de máquina, indução de regras, identificação de subgrupos, métodos estatísticos e geração de modelos múltiplos [28].

A interface exclusiva do IBM SPSS Modeler permite minerar dados, trabalhando com diagramas de fluxos. No nível mais básico, é possível criar um fluxo de dados usando as seguintes etapas:

- Adição de nós à tela de fluxo.
- Conexão dos nós para formar um fluxo.
- Especificação das opções dos nós.
- Execução do fluxo.

Ao contrário dos métodos estatísticos mais tradicionais, não é estritamente necessário precisar saber o que está procurando quando se inicia uma avaliação de característica entre bancos de dados. Sendo possível explorar os dados, ajustando-se a diferentes modelos e investigar diferentes relacionamentos, até que sejam encontradas informações úteis [23].

2.5 Medidas de Precisão

Levando em consideração que as sistematizações em qualquer *software* de predição precisam considerar algum marco como referência para avaliar as condições das saídas geradas frente às entradas e às comparações realizadas, tem-se que aplicar alguns conceitos para a interpretação das saídas geradas a partir da execução da etapa de processamento desta dissertação.

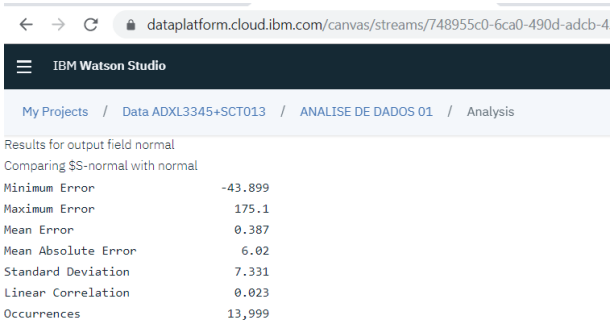
Desta forma, é necessário avaliar o preenchimento dos valores dos parâmetros da função kernel aplicada ao algoritmo de modelagem MVS, pois a seleção do tipo da função e a modificação dos parâmetros do kernel γ, C e ε tem influência direta sobre a execução da aplicação de MVS e as formulações dos resultados. Conduzindo a buscas para aplicação de parâmetros "ótimos" para obtenção de resultados satisfatórios, como em casos como [31] e [32] em que houve aplicação de métodos heurísticos de determinação dos parâmetros de kernel, estes parâmetros serão melhor discutidos no tópico 3.5.

O nó *Partition* (Partição) gera um campo de partição, que divide os dados em subconjuntos separados para os estágios de treinamento, teste e validação da construção do modelo. Dando espaço para testes do modelo em parte dos dados utilizados, sendo fundamental para verificação das características do modelo gerado.

Assim como os conhecimentos anteriormente citados, outra característica que deve ser compreendida para avaliar os resultados, é conhecimento do nó *Analysis* (na aba *Output*), que é de fundamental importância para a compreensão das avaliações da aplicação, no caso deste trabalho, de MVS sobre um banco de dados.

Para avaliar os resultados do nó *Analysis* a Figura 16 ilustra um exemplo de resultados obtidos.

Figura 16 - Saídas obtidas em Analysis.



Fonte: O próprio autor.

O Modeler Applications [29] e [23], fornecem as informações acerca de como avaliar os resultados de Analysis, que contém uma seção para cada campo de saída para o qual existe um campo de previsão correspondente criado por um modelo gerado.

A seguir as descrições das subseções de Analysis:

• **Erro mínimo:** Compreendido como a diferença mínima entre os valores observados e os previstos.

• **Erro máximo:** Compreendido como a diferença máximos entre os valores observados e os previstos.

• **Erro médio:** Mostra a média de erros em todos os registros. Indica se há uma tendência mais forte de superestimar do que subestimar o modelo. À medida que se percebe que este valor tem valor crescente, pode-se caracterizar o modelo com características de sobre ajuste.

• **Erro absoluto médio:** Mostra a média dos valores absolutos dos erros em todos os registros.

• **Desvio Padrão:** Mostra o desvio padrão dos erros.

• **Correlação linear:** Mostra a correlação linear entre os valores previstos e reais. Essa estatística varia entre -1,0 a 1,0. Valores próximos a 1,0 indicam uma forte associação positiva, de modo que valores previstos são associados a valores reais e baixos valores previstos, associados a baixos valores reais. Valores próximos a -1,0 indicam uma forte associação negativa, de modo que altos valores previstos são associados a baixos valores reais. Valores próximos a 0,0 indicam uma associação fraca, de modo que os valores previstos são um tanto quanto independentes dos valores reais.

Para avaliar as descobertas sobre resultados encontrados através do Analysis, deve-se compreender as saídas, como na figura 16, tomada como exemplo de interpretação, em que os valores encontrados da correlação linear é possível identificar uma baixa relação encontrada entre os valores de predição e os valores reais, devido à correlação linear estar próximo dos 50% do total que se pode obter, trazendo uma avaliação de baixa acurácia do modelo gerado. De acordo com a literatura [31] para bons valores entre a predição e os valores reais comparados, o valor de correlação linear deve ser superior a 80%, em [32] chegou-se a 98% de acurácia do modelo gerado. Para MVS, os valores dos erros representam as próprias medidas do plano de separação das classes, com valores de erro máximo e mínimo, erro médio, erro absoluto e desvio padrão.

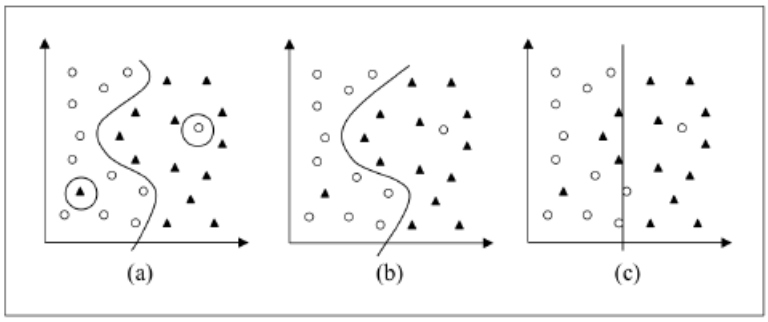
3 MÁQUINAS DE VETORES DE SUPORTE

De acordo com [33] as técnicas de AM empregam um princípio de inferência denominado indução, no qual obtém conclusões genéricas a partir de um conjunto particular de exemplos. O aprendizado indutivo pode ser dividido em dois tipos principais: supervisionado e não-supervisionado.

Seja f um classificador e F o conjunto de todos os classificadores que um determinado algoritmo de AM pode gerar. Esse algoritmo, durante o processo de aprendizado, utiliza um conjunto de treinamento T, composto de n pares $(x_i; y_i)$, para gerar um classificador particular $\hat{f} \in F$.

Considere, por exemplo, o conjunto de treinamento ilustrado na Figura 17. O objetivo do processo de aprendizado é encontrar um classificador que separe os dados das classes "círculo" e "triângulo". As funções ou hipóteses consideradas são ilustradas na figura, também denominadas fronteiras de decisão, traçadas entre as classes. [34]

Figura 17 - Conjunto de treinamento binário em três hipóteses diferentes.



Fonte: [33].

Na Figura 17a, tem-se uma hipótese que classifica corretamente todos os exemplos do conjunto de treinamento, incluindo dois possíveis ruídos. Por ser muito específica para o conjunto de treinamento, essa função apresenta elevada suscetibilidade a cometer erros quando confrontada com novos dados. Esse caso representa a ocorrência de um super ajustamento(*overfiting*) do modelo frente aos dados de treinamento [33][34].

Um outro classificador poderia desconsiderar pontos pertencentes a classes opostas que estejam muito próximos entre si como ilustra a figura 17c. A nova hipótese considerada, porém, comete muitos erros, mesmo para casos que podem ser considerados simples. Desta forma há a

ocorrência de um sub-ajustamento (*underfiting*), pois o classificador não é capaz de se ajustar mesmo aos exemplos de treinamento.[42]

A Teoria de Aprendizado Estatístico (TAE) estabelece condições matemáticas que auxiliam na escolha de um classificador particular $\hat{f}$ a partir de um conjunto de dados de treinamento. Essas condições levam em conta o desempenho do classificador no conjunto de treinamento e a sua complexidade, com o objetivo de obter um bom desempenho também para novos dados do mesmo domínio. [33][34]

3.1 Considerações sobre a Escolha do Classificador

Na aplicação da TAE, é assumido inicialmente que os dados do domínio em que o aprendizado está ocorrendo são gerados de forma independente e identicamente distribuída (i.i.d.) de acordo com uma distribuição de probabilidade $P(x, y)$ que descreve a relação entre os dados e os rótulos [36, 34]. O erro (também denominado risco) esperado de um classificador f para dados de teste pode então ser quantificado pela expressão 3.1 [35]. O risco esperado mede então a capacidade de generalização de f [37]. Na expressão (3.1), $c(f(x, y)$ é uma função de custo relacionando à previsão $f(x)$ quando a saída desejada é y.

$$R(f) = \int c\, f\big((x), y\big)\, dP(x, y) \tag{3.1}$$

Infelizmente, não é possível minimizar o risco esperado apresentado na expressão 3.1 diretamente, uma vez que em geral a distribuição de probabilidade $P(x, y)$ é desconhecida [35]. Tem-se unicamente a informação dos dados de treinamento, também amostrados de $P(x, y)$. Normalmente utiliza-se o princípio da indução para inferir uma função $\hat{f}$ que minimize o erro sobre esses dados e espera-se que esse procedimento leve também a um menor erro sobre os dados de teste [34]. O risco empírico de f , fornecido pela expressão (3.2) mede o desempenho do classificador nos dados de treinamento, por meio da taxa de classificações incorretas obtidas em T [33][35].

$$R_{emp}(f) = \frac{1}{n}\sum_{i=1}^{n} c((f x_i), y_i) \tag{3.2}$$

Esse processo de indução com base nos dados de treinamento conhecidos constitui o princípio de minimização do risco empírico [34]. Assintoticamente, com $n \to \infty$, é possível

estabelecer condições para o algoritmo de aprendizado que garantam a obtenção de classificadores cujos valores de risco empírico convergem para o risco esperado [35]. Para conjuntos de dados menores, porém, em que não é possível determinar esse tipo de garantia.

A noção expressa nesses argumentos é a de que, permitindo que $\hat{f}$ seja escolhida a partir de um conjunto de funções amplo F, é sempre possível encontrar uma f com pequeno risco empírico. Porém, nesse caso os exemplos de treinamento podem se tornar pouco informativos para a tarefa de aprendizado, pois o classificador induzido pode se super ajustada a eles. É necessário então restringir a classe de funções da qual $\hat{f}$ é extraída. Existem diversas abordagens para tal. A TAE lida com essa questão considerando a complexidade (também referenciada por capacidade) da classe de funções que o algoritmo de aprendizado é capaz de obter [34]. Nessa direção, a TAE provê diversos limites no risco esperado de uma função de classificação, os quais podem ser empregados na escolha do classificador.

A TAE lida com essa questão considerando a complexidade (também referenciada por capacidade) da classe de funções que o algoritmo de aprendizado é capaz de obter [42][33]. Nessa direção, a TAE provê diversos limites no risco esperado de uma função de classificação, os quais podem ser empregados na escolha do classificador. A próxima seção relaciona alguns dos principais limites sobre os quais as MVSs se baseiam.

3.1.1 Limite no risco esperado

Um limite importante fornecido pela TAE relaciona o risco esperado de uma função ao risco empírico e a um termo de capacidade. Esse limite, é garantido com probabilidade$1 - \theta$, é descrita em [36]

$$R(f) \leq R_{emp}(f) + \frac{\sqrt{h\left(ln\left(2n/h\right) + 1\right) - l_n\left(\frac{\theta}{4}\right)}}{n} \tag{3.3}$$

Nessa expressão, h denota a dimensão Vapnik-Chervonenkis (VC) [38] da classe de funções F à qual f pertence, n representa a quantidade de exemplos no conjunto de treinamento T e a parcela de raiz na soma é referenciada como termo de capacidade.

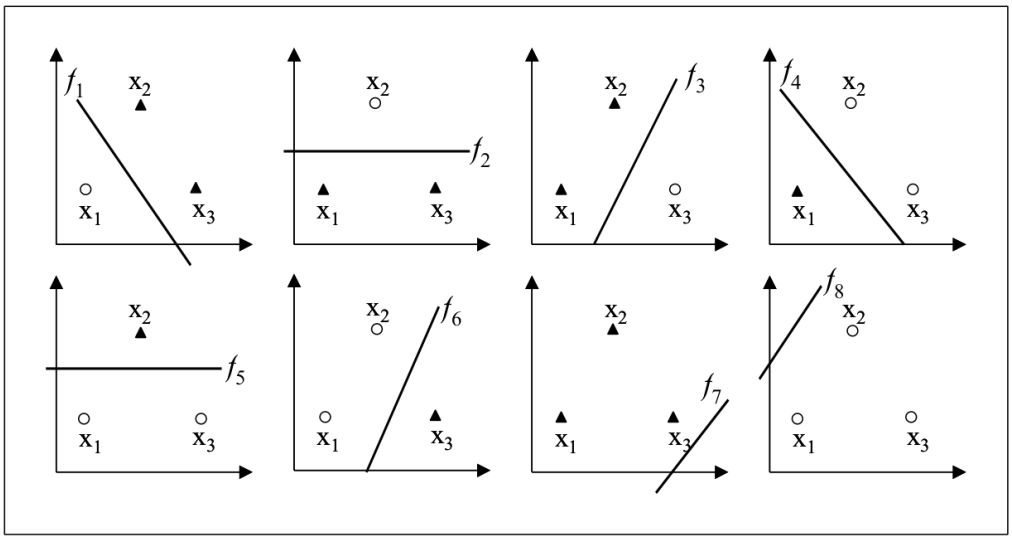
A dimensão VC h mede a capacidade do conjunto de funções F [36]. Quanto maior o seu valor, mais complexas são as funções de classificação que podem ser induzidas a partir de F.

Dado um problema de classificação binário, essa dimensão é definida como o número máximo de exemplos que podem ser particionados em duas classes pelas funções contidas em F, para todas as possíveis combinações binárias desses dados.

Para ilustrar esse conceito, considere os três dados ilustrados na Figura 18 [34].

Pode-se verificar que, para qualquer conformação arbitrária dos rótulos “círculo” e “triângulo” que esses dados possam assumir, é possível determinar retas capazes de separá-los.

Figura 18 - Separação de dados por meio de retas.

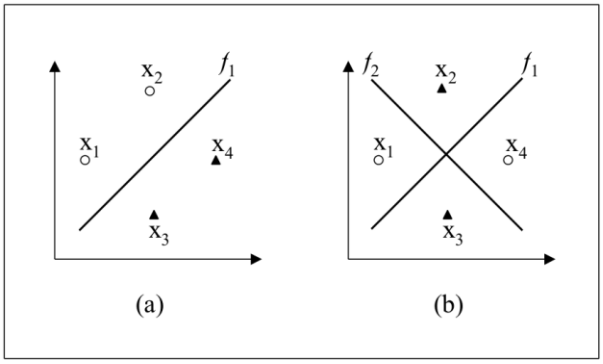


Fonte: Adaptado de [34].

Porém, para os quatro pontos ilustrados na Figura 19, existem rótulos para os dados que podem ser separados por uma reta (Figura 19a), mas também é possível definir rótulos tal que uma só reta seja incapaz de realizar a separação em classes (Figura 19b) [33]. Para uma divisão binária arbitrária desses quatro pontos, deve-se então recorrer a funções de complexidade superior à das retas. Essa observação se aplica a quaisquer quatro pontos no espaço bidimensional.

A contribuição principal da expressão (3.3) está em afirmar a importância de se controlar a capacidade do conjunto de funções F do qual o classificador é extraído. Interpretando-a em termos práticos, tem-se que o risco esperado pode ser minimizado pela escolha adequada. Com esses objetivos, definiu-se um princípio de indução denominado minimização do risco estrutural [33].

Figura 19 - Separação de quatro dados por meio de retas.



Fonte: Adaptado de [33].

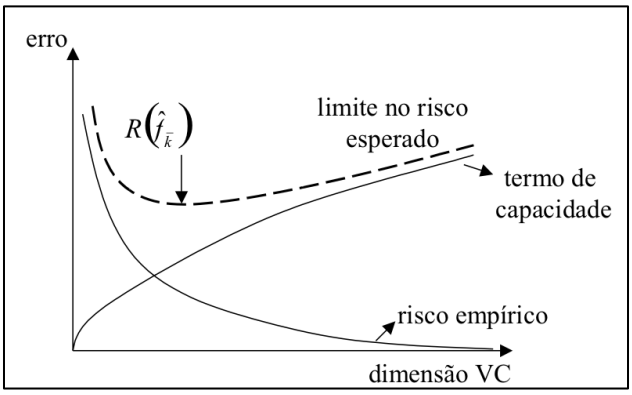
Como no limite apresentado o termo de capacidade diz respeito à classe de funções F e o risco empírico refere-se a um classificador particular f, para minimizar ambas as parcelas divide-se inicialmente F em subconjuntos de funções com dimensão VC crescente [33]. É comum referir-se a esse processo como introduzir uma estrutura em F, sendo os subconjuntos definidos também denominados estruturas [34]. Minimiza-se então o limite sobre as estruturas introduzidas.

Considerando subconjuntos F_i da seguinte forma: $F_0 \subset F_i \subset \cdots . \subset F_q \subset F$. Como cada F_i é maior com o crescimento do índice i, a capacidade do conjunto de funções que ele representa também é maior à medida que i cresce, ou seja, h0 < h1 < ... < hq < h Para um subconjunto particular F_k, seja $\hat{f}_k$o classificador com o menor risco empírico.

A medida que k cresce, o risco empírico de $\hat{f}_k$ diminui, uma vez que a complexidade do conjunto de classificadores é maior. Porém, o termo de capacidade aumenta com k. Como resultado, deve haver um valor ótimo $\bar{k}$ em que se obtém uma soma mínima do risco empírico e do termo de capacidade, minimizando assim o limite sobre o risco esperado. A escolha da função $\hat{f}_{\bar{k}}$constitui o princípio da minimização do risco estrutural. Os conceitos discutidos são ilustrados na Figura 20.

Embora o limite representado na expressão (3.3) tenha sido útil na definição do procedimento de minimização do risco estrutural, na prática surgem alguns problemas. Em primeiro lugar, computar a dimensão VC de uma classe de funções em que não é uma tarefa trivial. Soma-se a isso o fato de que o valor de h poder ser desconhecido ou infinito [35].

Figura 20 - Princípio de minimização do risco estrutural.



Fonte: Adaptado de [33].

Para funções de decisão lineares do tipo $f(x) = w \cdot x$, entretando, existem resultados alternativos que relacionam o risco esperado ao conceito de margem [34]. A margem de um exemplo tem relação com sua distância à fronteira de decisão induzida, sendo uma medida da confiança da previsão do classificador. Para um problema binário, em que $y_1 \epsilon \{1-, +1\}$, dada uma função f e um exemplo x_i, a margem $\sigma(f(x_i)y_i)$ com que esse dado é classificado por f pode ser calculada pela expressão (3.4). Logo, um valor negativo de $\sigma\ (x_i, y_i)$ denota uma classificação incorreta. [34]

$$\sigma(f(x_i)y_i) = y_i f(x_i) \tag{3.4}$$

Para obter a margem geométrica de um dado x_i em relação a uma fronteira linear $f(x) = w \cdot x + b$, a qual mede efetivamente a distância de x_i à fronteira de decisão, divide o termo à direita da expressão (3.4) pela magnitude de w, ou seja, por $\|w\|$ [34]. Para exemplos incorretamente classificados, o valor obtido equivale à distância com sinal negativo. Para realizar uma diferenciação, a margem da expressão (3.4) será referenciada como margem de confiança.

A partir do conceito introduzido, é possível definir o erro marginal de uma função$f\left(R_p(f)\right)$ sobre um conjunto de treinamento. Esse erro fornece a proporção de exemplos de treinamento cuja margem de confiança é inferior a uma determinada constante $\rho > 0$, expressão (3.5) [33].

$$R_\rho(f) = \frac{1}{n}\sum_{i=1}^{n} I(y_i f(xi) < \rho) \tag{3.5}$$

Na expressão 3.5, $I(q) = 1$ se q é verdadeiro e $I(q) = 0$ se q é falso.

Existe uma constante c tal que, com probabilidade $1 - \theta\epsilon$ [0,1], para todo $\rho > 0$ e F correspondendo à classe de funções lineares $f(x) = w \cdot x$ com $\|x\| \leq R$ e $\|x\| \leq 1$, o seguinte limite se aplica [33]:

$$R(f) \leq R_p(f) + \sqrt{\frac{c}{n}\left(\frac{R^2}{\rho^2}\right) log^2\left(\frac{n}{\rho}\right) + log\left(\frac{1}{\theta}\right)} \tag{3.6}$$

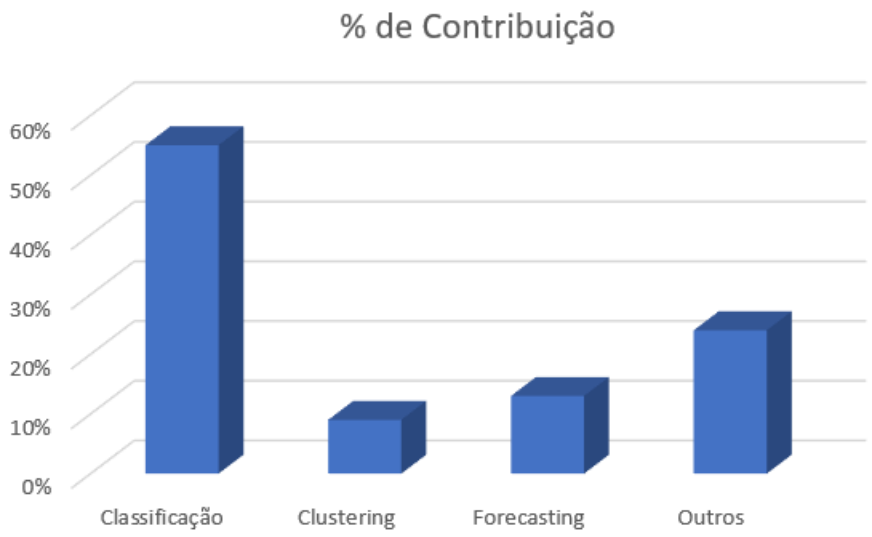
Como na expressão (3.3), tem-se na expressão (3.6) novamente o erro esperado limitado pela soma de uma medida de erro no conjunto de treinamento, neste caso o erro marginal, a um termo de capacidade. A interpretação do presente limite é de que uma maior margem ρ implica em um menor termo de capacidade. Entretanto, a maximização da margem pode levar a um aumento na taxa de erro marginal, pois torna-se mais difícil obedecer à restrição de todos os dados de treinamento estarem distantes de uma margem maior em relação ao hiperplano separador. Um baixo valor de ρ, em contrapartida, leva a um erro marginal menor, porém aumenta o termo de capacidade. Deve-se então buscar um compromisso entre a maximização da margem e a obtenção de um erro marginal baixo.[33][34]

Como conclusão, tem-se que, na geração de um classificador linear, deve-se buscar um hiperplano que tenha margem ρ elevada e cometa poucos erros marginais, minimizando assim o erro sobre os dados de teste e de treinamento, respectivamente. Esse hiperplano é denominado ótimo [34].

Existem diversos outros limites reportados na literatura, assim como outros tipos de medida de complexidade de uma classe de funções [35]. Um exemplo é a dimensão *fatshattering*, que caracteriza o poder de um conjunto de funções em separar os dados com uma margem ρ. Os limites apresentados anteriormente, embora possam ser considerados simplificados, provêm uma base teórica suficiente à compreensão de MVS.

Conforme ilustrado na Figura 21, a máquina de vetores de suporte foi aplicada em quase todos os domínios, incluindo problemas de classificação, previsão, análise de imagens, reconhecimento de padrões, extração de regras e otimização. A maioria das aplicações que foram analisadas, cerca de 54%, estão na área de classificação. Os trabalhos nas áreas de agrupamento e previsão representam 9% e 13%, respectivamente, e outros representam 24% dos trabalhos revisados.

Figura 21 - Aplicação de MVS.



Fonte: Adaptado de [42].

3.2 MVS Linear

As MVS surgiram pelo emprego direto dos resultados fornecidos pela TAE. Nesta seção é apresentado o uso de MVS na obtenção de fronteiras lineares para a separação de dados pertencentes a duas classes. A primeira formulação, mais simples, lida com problemas linearmente separáveis, definidos adiante [33]. Essa formulação foi posteriormente estendida para definir fronteiras lineares sobre conjuntos de dados mais gerais [38].

3.2.1 MVS margem rígida

As MVS lineares com margens rígidas definem fronteiras lineares a partir de dados linearmente separáveis. Seja T um conjunto de treinamento com n dados $x_i \in X$ e os respectivos rótulos $y_i \in Y$, em que X constitui o espaço dos dados e $Y = \{-1, +1\}$. T é linearmente separável se é possível separar os dados das classes $+1$ e -1 por um hiperplano [34].

Classificadores que separam os dados por meio de um hiperplano são denominados lineares [33]. A representação de um hiperplano é apresentada na expressão (3.7), em que $w \cdot x$ é o

produto escalar entre os vetores w e x, $w \in$ X é o vetor normal ao hiperplano descrito e $\frac{b}{\|w\|}$ corresponde à distância do hiperplano em relação à origem, com $b \in R$.

$$f(x) = wx + b = 0 \tag{3.7}$$

Essa expressão divide o espaço dos dados X em duas regiões: $w \cdot x + b > 0$ e $w \cdot x + b < 0$. Uma função sinal $g(x) = sng(f(x))$ pode então ser empregada na obtenção das classificações, conforme ilustrado na expressão 3.8 [33].

$$g(x) = sgn(f(x)) = \begin{Bmatrix} +1 \; se \; w \cdot x + b > 0 \\ -1 \; se \; w \cdot x + b < 0 \end{Bmatrix} \tag{3.8}$$

A partir de $f(x)$, é possível obter um número infinito de hiperplanos equivalentes, pela multiplicação de w e b por uma mesma constante [38]. Define-se o hiperplano canônico em relação ao conjunto T como aquele em que w e b são escalados de forma que os exemplos mais próximos ao hiperplano: $w \cdot x + b = 0$ satisfaçam a expressão (3.9) [34].

$$|w \cdot x_i + b| = 1 \tag{3.9}$$

Essa forma implica na expressão (3.10), resumida na expressão (3.11).

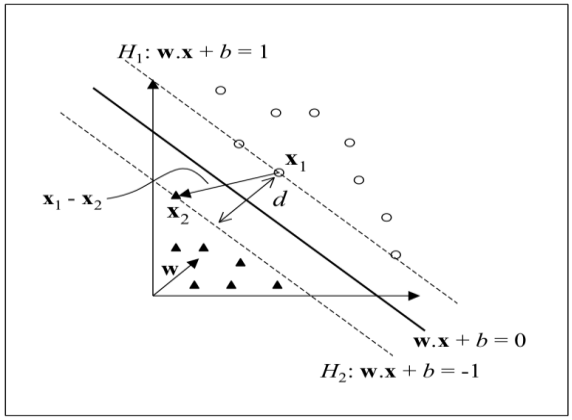
$$\begin{cases} w \cdot x_i + b \geq +1 \;\; se \; yi = +1 \\ w \cdot x_i + b \leq -1 \;\; se \; yi = -1 \end{cases} \tag{3.10}$$

$$y_i(w \cdot x_i + b) - 1 \geq 0, \qquad \forall (x_i, y_i) \in T \tag{3.11}$$

Seja x_1 um ponto no hiperplano $H_1: w \cdot x + b = +1$ e x_2 um ponto no hiperplano $H_2: w \cdot x + b = -1$, conforme ilustrado na Figura 22. Projetando $x_1 - x_2$ na direção de w, perpendicular ao hiperplano separador $w \cdot x + b = 0$, é possível obter a distância entre os hiperplanos H_1 e H_2 [33][38]. Essa projeção é apresentada na expressão (3.12).

$$(x_1 - x_2)\left(\frac{w}{\|w\|} \cdot \frac{(x_1 - x_2)}{\|x_1 - x_2\|}\right) \tag{3.12}$$

Figura 22 - Cálculo da distância d entre os hiperplanos H1 e H2.



Fonte: Adaptado de [15].

Tem-se que $w \cdot x_1 + b = +1$ e $w \cdot x_2 + b = -1$, representam os espações entre os hiperplanos. De onde torna-se matematicamente conveniente entre essas equações, chegar a $w \cdot (x_1 - x_2) = 2$. E substituindo esse resultado na expressão (3.12), tem-se:

$$\frac{2}{\|w\|} \cdot \frac{(x_1 - x_2)}{\|x_1 - x_2\|} \tag{3.13}$$

Essa é a distância d, ilustrada na Figura 23, entre os hiperplanos H_1 e H_2, paralelos ao hiperplano separador. Como w e b foram escalados de forma a não haver exemplos entre H_1 e H_2, $\frac{2}{\|w\|}$ é a distância mínima entre o hiperplano separador e os dados de treinamento. Essa distância é definida como a margem geométrica do classificador linear [33].

A partir das considerações anteriores, verifica-se que a maximização da margem de separação dos dados em relação a $w \cdot x + b = 0$ pode ser obtida pela minimização de$\|w\|$ [5][34]. Dessa forma, recorre-se ao seguinte problema de otimização:

$$\text{Minimizar } \frac{1}{2}\|w\|^2 \tag{3.14}$$

Com as restrições: $y_i(w \cdot x_i + b) - 1 \geq 0, \ \forall(1, \dots n$

As restrições são impostas de maneira a assegurar que não haja dados de treinamento entre as margens de separação das classes. Por esse motivo, a MVS obtida possui também a nomenclatura de MVS com margens rígidas.

O problema de otimização obtido é quadrático, cuja solução possui uma ampla e estabelecida teoria matemática [34]. Como a função objetivo sendo minimizada é convexa e os pontos que satisfazem as restrições formam um conjunto convexo, esse problema possui um único mínimo global [37]. Problemas desse tipo podem ser solucionados com a introdução de uma função Lagrangiana, que engloba as restrições à função objetivo, associadas a parâmetros denominados multiplicadores de Lagrange, expressão (3.15).

$$L(w, b, \alpha) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{n} \alpha_i \left(y_i(wx_i + b) - 1\right) \tag{3.15}$$

A função Lagrangiana deve ser minimizada, o que implica em maximizar as variáveis α_i e minimizar w e b [33]. Tem-se então um ponto de sela, no qual:

$$\frac{\partial L}{\partial n} = O \quad e \quad \frac{\partial L}{\partial w} = 0 \tag{3.16}$$

A resolução dessas equações leva aos resultados representados nas expressões (3.17) e (3.18).

$$\sum_{1=1}^{n} \alpha_i \cdot y_i = 0 \tag{3.17}$$

$$w = \sum_{1=1}^{n} \alpha_i \cdot y_i \, x_i \tag{3.18}$$

Substituindo as expressões (3.17), (3.18) em (3.15), obtém-se o seguinte problema de otimização:

$$Maximizar\ \alpha \sum_{i=1}^{M} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{M} \alpha_i\, \alpha_j y_i y_j (x_i x_{j)} \tag{3.19}$$

Com as restrições: $\begin{cases} \alpha_i \geq 0, \quad \forall i = 1, \dots, n \\ \sum_{i=1} \alpha_1 \cdot y_i = 0 \end{cases}$

Essa formulação é denominada forma dual, enquanto o problema original é referenciado como forma primal. A forma dual possui os atrativos de apresentar restrições mais simples e permitir a representação do problema de otimização em termos de produtos internos entre dados, o que será útil na posterior não-linearização das MVSs (Seção 3.3). É interessante observar também que o problema dual é formulado utilizando apenas os dados de treinamento e os rótulos.

Seja α^* a solução do problema dual e w^* e b^* as soluções da forma primal. Obtido o valor de α^*, w^* pode ser determinado pela expressão (3.18). O parâmetro b^* é definido por α^* e por condições de Kühn-Tucker, provenientes da teoria de otimização com restrições e que devem ser satisfeitas no ponto ótimo. Para o problema dual formulado é descrito da seguinte forma [33]:

$$\alpha^*(y_i(w^* \cdot x_i + b^*) - 1) = 0, \qquad \forall i = 1, \dots n \tag{3.20}$$

Observa-se nessa expressão que α_i^* pode ser diferente de 0 somente para os dados que se encontram sobre os hiperplanos H_1 e H_2. Estes são os exemplos que se situam mais próximos ao hiperplano separador, de acordo com [36], exatamente sobre as margens. Para outros casos, a condição apresentada na expressão (3.20) é obedecida apenas com α_i^* = 0. Esses pontos não participam então do cálculo de w^* (expressão 3.19). Os dados que possuem $\alpha_i^* > 0$ são denominados vetores de suporte (*Support Vectors*) e podem ser considerados os dados mais informativos do conjunto de treinamento, pois somente eles participam na determinação da expressão do hiperplano separado (3.23).

O valor de b^* é calculado a partir dos SVs e das condições representadas na expressão (3.21). Computa-se a média apresentada na expressão 3.21 sobre todos x_j tal que $\alpha_j^* > 0$, ou seja, todos os SVs. Nessa expressão, n_{SV} denota o número de SVs. [34]

$$b^* = \frac{1}{n_{SV}} \sum_{x_j \in SV} \frac{1}{y_j} - w^* \cdot x_j \tag{3.21}$$

Substituindo w^* na expressão 3.18, é obtido:

$$b^* = \frac{1}{n_{VS}} \sum_{x_j \in SV} (\frac{1}{y_j} - \sum_{x_i \in SV} \alpha_i^* \, y_i (x_i \cdot x_j)) \tag{3.22}$$

Como resultado final, tem-se o classificador $g(x)$ apresentado na expressão (3.23), em que sgn representa a função sinal, w^* é fornecido pela expressão (3.18) e b^* pela expressão (3.22).

$$g(x) = sgn(f(x)) = sng\left(\sum_{x_i \in SV} y_i \alpha_i^* x_i \cdot x + b^* \right) \tag{3.23}$$

Esta função linear representa o hiperplano que separa os dados com maior margem, considerado aquele com melhor capacidade de generalização de acordo com a TAE. Essa característica difere as MVSs lineares de margens rígidas Redes Neurais Perceptron, em que o hiperplano obtido na separação dos dados pode não corresponder ao de maior margem de separação [33].

3.2.2 MVS margem suave

Em situações reais, é difícil encontrar aplicações cujos dados sejam linearmente separáveis. Isso se deve a diversos fatores, entre eles a presença de ruídos e outliers nos dados ou à própria natureza do problema, que pode ser não linear. Nesta seção as MVSs lineares de margens rígidas são estendidas para lidar com conjuntos de treinamento mais gerais. Para realizar essa tarefa, permite-se que alguns dados possam violar a restrição da expressão (3.14). Isso é feito com a introdução de variáveis de folga ξi, para todo $i = 1 \ldots n$[37]. Essas variáveis relaxam as restrições impostas ao problema de otimização primal, que se tornam [38]:

$$y_i(wx_i + b) \geq 1 - \xi\text{i}, \qquad \xi\text{i} \geq 0, \;\; \forall\text{i} = 1, \ldots n \tag{3.24}$$

A aplicação desse procedimento suaviza as margens do classificador linear, permitindo que alguns dados permaneçam entre os hiperplanos e também a ocorrência de alguns erros de classificação. Por esse motivo, as MVSs obtidas neste caso também podem ser referenciadas como MVSs com margens suaves.

Um erro no conjunto de treinamento é indicado por um valor de ξi maior que 1. Logo, a soma dos ξi representa um limite no número de erros de treinamento [36]. Para levar em consideração esse termo, minimizando assim o erro sobre os dados de treinamento, a função objetivo da expressão (3.14) é reformulada como:

$$Minimizar\ \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi i \tag{3.25}$$

A constante C é um termo de regularização que impõe um peso à minimização dos erros no conjunto de treinamento em relação à minimização da complexidade do modelo [37]. A presença do termo $\sum_{i=1}^{n}\xi i$ no problema de otimização também pode ser vista como uma minimização de erros marginais, pois um valor de $\xi i\ \epsilon$ [0,1] indica um dado entre as margens.

Novamente o problema de otimização gerado é quadrático, com as restrições lineares apresentadas na expressão (3.24). A sua solução envolve passos matemáticos semelhantes aos apresentados anteriormente, com a introdução de uma função Lagrangiana e tornando as derivadas parciais nulas. Tem-se como resultado o seguinte problema dual:

$$\text{Maximizar } \sum_{i=1}^{n}\alpha i - \frac{1}{2}\sum_{i,j=1}^{n}\alpha_i\alpha_j y_1 \cdot y_j(x_i x_j) \tag{3.26}$$

$$\text{Com as restrições } \begin{cases} 0 \le \alpha_i x \le C\ , \forall \text{i} = 1 \dots n \\ \sum_{i=1}^{n}\alpha_i y_i = 0 \end{cases}$$

Pode-se observar que essa formulação é igual à apresentada para as SVMs de margens rígidas, a não ser pela restrição nos α_i, que agora são limitados pelo valor de C.

Seja α^* a solução do problema dual, enquanto w^*, b^*e ξ^* denotam as soluções da forma primal. O vetor w^* continua sendo determinado pela expressão (3.15). As variáveis ξi^* podem ser calculadas pela expressão (3.28), descrita em [33].

$$\xi i^* = max\ \{0,1 - y_{\mathrm{i}} \sum_{j=1}^{n} y_{\mathrm{j}} \alpha_j^* x_j\ x_i + \ \mathrm{b}^*\} \tag{3.27}$$

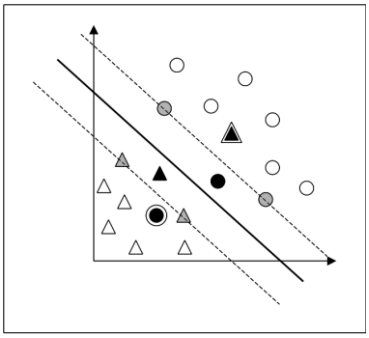
A variável b^* provém novamente de α^* e de condições de Karush-Kuhn-Tucker, que neste caso são [33]:

$$\alpha_i^* (y_i (w^* \cdot x_i + b^*) - 1 + \xi i^*) = 0 \tag{3.28}$$

$$(C - \alpha_i^*) \xi i^* = 0 \tag{3.29}$$

Como nas MVSs de margens rígidas, os pontos x_i para os quais $\alpha_i^* > 0$ são denominados vetores de suporte (SVs), sendo os dados que participam da formação do hiperplano separador. Porém, neste caso, pode-se distinguir tipos distintos de SVs [33]. Se $\alpha_i^* < \mathrm{C}$, pela expressão (3.29), $\xi i^* = 0$ e então, da expressão (3.28), estes SVs encontram-se sobre as margens e também são denominados livres. Os SVs para os quais $\alpha_i^* = \mathrm{C}$ podem representar três casos [33]: erros, se $\xi i^* > 1$; pontos corretamente classificados, porém entre as margens, se $0 < \xi i^* \leq 1$; ou pontos sobre as margens, se $\xi i^* = 0$. O último caso ocorre raramente e os SVs anteriores são denominados limitados. Na Figura 23 são ilustrados os possíveis tipos de SVs. Pontos na cor cinza representam SVs livres. SVs limitados são ilustrados em preto. Pontos pretos com bordas extras correspondem a SVs limitados que são erros de treinamento. Todos os outros dados, em branco, são corretamente classificados e encontram-se fora das margens, possuindo $\xi i^* = 0$ e $\alpha_i^* = 0$.

Figura 23 -Tipos de SVs: livres (cor cinza) e limitados (cor preta).



Fonte: Adaptado de [33].

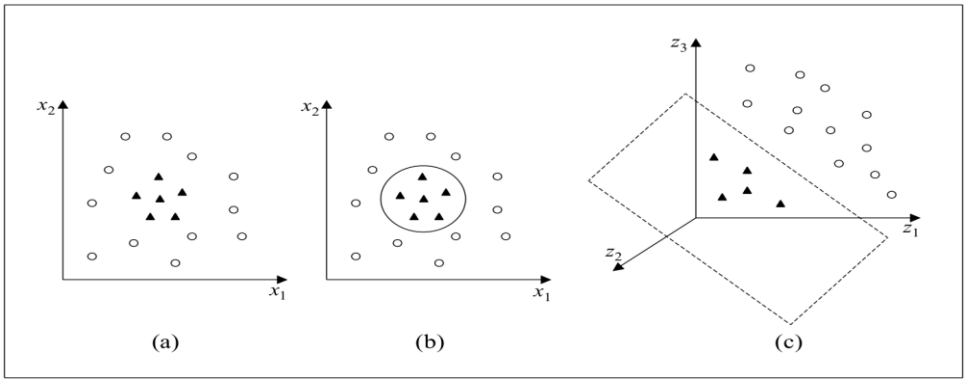
Para calcular b^*, computa-se a média da expressão (3.22) sobre todos SVs x_j entre as margens, ou seja, com $\alpha_i^* < C$ [34].

Tem-se como resultado final a mesma função de classificação representada na expressão (3.23), porém neste caso as variáveis α_i^* são determinadas pela solução da expressão (3.26).

3.3 MVS Não Linear

As MVSs lineares são eficazes na classificação de conjuntos de dados linearmente separáveis ou que possuam uma distribuição aproximadamente linear, sendo que a versão de margens suaves tolera a presença de alguns ruídos e *outliers*. Porém, há muitos casos em que não é possível dividir satisfatoriamente os dados de treinamento por um hiperplano. Um exemplo é apresentado na Figura 24a, em que o uso de uma fronteira curva seria mais adequado na separação das classes.[33][35]

Figura 24 -(a) Amostra não linear. (b) Fronteira não linear no espaço. (c) Espaço tridimensional para análise.



Fonte: Adaptado de [33][35].

As MVSs lidam com problemas não lineares mapeando o conjunto de treinamento de seu espaço original, referenciado como de entradas, para um novo espaço de maior dimensão, denominado espaço de características (*feature space*) [34]. Seja $\phi: X \rightarrow \beta$ um mapeamento, em que X é o espaço de entradas e β denota o espaço de características. A escolha apropriada de ϕ faz com que o conjunto de treinamento mapeado em β possa ser separado por uma MVS linear.

O uso desse procedimento é motivado pelo teorema de Cover [33]. Dado um conjunto de dados não linear no espaço de entradas X, esse teorema afirma que X pode ser transformado em um espaço de características β no qual com alta probabilidade os dados são linearmente separáveis. Para isso duas condições devem ser satisfeitas. A primeira é que a transformação seja não linear, enquanto a segunda é que a dimensão do espaço de características seja suficientemente alta.

Para ilustrar esses conceitos, considere o conjunto de dados apresentado na figura 24a [35]. Transformando os dados de R para R^2 com o mapeamento representado na expressão (3.30), o conjunto de dados não linear em R^2 torna-se linearmente separável em R (Figura 24c).

É possível então encontrar um hiperplano capaz de separar esses dados, descrito na expressão (3.31). Pode-se verificar que a função apresentada, embora linear em R^2 (Figura 24c), corresponde a uma fronteira não linear em R (Figura 24b).

$$\phi(x) = \phi(x_1, x_2) = \left(x_1^2, \sqrt{2x_1}x_2, x_2^2\right) \tag{3.30}$$

$$f(x) = w \cdot \phi(x) + b = w_1 \cdot x_1^2 + w_2\sqrt{2}x_1x_2 + w_3x_2^2 + b = 0 \tag{3.31}$$

Logo, mapeia-se inicialmente os dados para um espaço de maior dimensão utilizando ϕ e aplica-se a MVS linear sobre este espaço. Essa encontra o hiperplano com maior margem de separação, garantindo assim uma boa generalização. Utiliza-se a versão de MVS linear com margens suaves, que permite lidar com ruídos e *outliers* presentes nos dados. Para realizar o mapeamento, aplica-se ϕ aos exemplos presentes no problema de otimização representado na expressão (3.26), conforme a expressão (3.32):

$$\text{Maximizar } \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{n} \alpha_i\alpha_j y_\text{i} \cdot y_j(\phi(x_i)\ \ \phi(x_j)) \tag{3.32}$$

Sob as restrições da expressão (3.26). De forma semelhante, o classificador extraído se torna:

$$g(x) = sgn(f(x)) = sgn\left(\sum_{x_i \in sv} \alpha_1^* y_i \phi(x_i) \cdot \phi(x) \ + b^*\right) \tag{3.33}$$

Como β pode ter dimensão muito alta (até mesmo infinita), a computação de ϕ pode ser extremamente custosa ou inviável. Porém, percebe-se pelas expressão (3.32) e (3.33) que a única informação necessária sobre o mapeamento é de como realizar o cálculo de produtos escalares entre os dados no espaço de características, pois tem-se sempre $\phi(x_i) \cdot \phi x_j$, para dois dados x_i e x_j , em conjunto. Isso é obtido com o uso de funções denominadas Kernels.

Um Kernel K é uma função que recebe dois pontos x_i e x_j do espaço de entradas e computa o produto escalar desses dados no espaço de características [33]. Tem-se então:

$$K(x_i, x_j) = \ \phi(x_i) \cdot \phi x_j \tag{3.34}$$

Para o mapeamento apresentado na expressão (3.30) e dois dados e $x_i = (x_{1i}, x_{2i})$ $x_j = (x_{1j}, x_{2j})$ em R^2, por exemplo, o Kernel é dado por [39]:

$$K(x_i, x_j) = (x_{1i}, \sqrt{2}x_{1i}x_{2i}, x_{2i}^2) \cdot \ (x_{1j}, \sqrt{2}x_{1j}x_{2j}, x_{2j}^2) = (x_i \cdot x_j)^2 \tag{3.35}$$

É comum empregar a função Kernel sem conhecer o mapeamento ϕ, que é gerado implicitamente. A utilidade dos Kernels está, portanto, na simplicidade de seu cálculo e em sua capacidade de representar espaços abstratos.

Para garantir a convexidade do problema de otimização formulado na expressão (3.26) e também que o Kernel represente mapeamentos nos quais seja possível o cálculo de produtos escalares conforme a expressão (3.35), utiliza-se funções Kernel que seguem as condições estabelecidas pelo teorema de Mercer [33]. De forma simplificada, um Kernel que satisfaz as condições de Mercer é caracterizado por dar origem a matrizes positivas semi-definidas K, em que cada elemento é definido por $K_{ij} = K(x_i \cdot x_j)$, para todo $i, j = 1 \ldots n$ [38].

Alguns dos Kernels mais utilizados na prática são os Polinomiais, os Gaussianos ou RBF (*Radial-Basis Function*) e os Sigmoidais, listados na Tabela 2. Cada um deles apresenta parâmetros que devem ser determinados pelo usuário, indicados também na tabela. O Kernel Sigmoidal, em particular, satisfaz as condições de Mercer apenas para alguns valores de δ e k. Os Kernels Polinomiais com $d = 1$ também são denominados lineares.[31]

Tabela 2 - Funções kernel mais comuns.

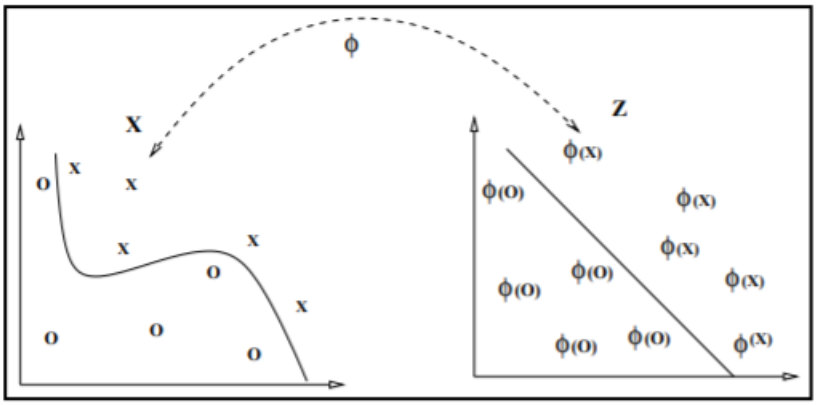
Tipo de Kernel	**Função** $K(\mathbf{x}_i, \mathbf{x}_j)$	**Parâmetros**
Polinomial	$(\delta(\mathbf{x}_i \cdot \mathbf{x}_j) + \kappa)^d$	δ, κ e d
Gaussiano	$\exp\left(-\sigma \|\mathbf{x}_i - \mathbf{x}_j\|^2\right)$	σ
Sigmoidal	$\tanh(\delta(\mathbf{x}_i \cdot \mathbf{x}_j) + \kappa)$	δ e κ

Fonte: Adaptado de [8][33].

Apesar de o MVS apresentar bom poder de generalização [31], seu desempenho depende da seleção de parâmetros na função de kernel do classificador. A escolha de parâmetros inadequados pode resultar em decréscimo na acurácia dos resultados. Atualmente não existe um método universal para guiar a seleção de parâmetros do kernel.

As representações Kernel trabalham com a projeção dos dados em um espaço de características com alta dimensão para permitir a classificação em espaços não-linearmente separáveis. Trata-se, em primeira instância, de uma estratégia de pré-processamento que envolve mudar a representação de dados da seguinte forma, Figura 25:

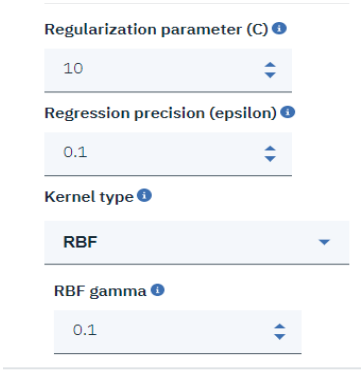
Figura 25 - Mapeamento de um espaço de entrada via função kernel.



Fonte: Adaptado de [42].

Dentro da aplicação de MVS no ambiente de IBM SPSS Modeler realizada neste trabalho, as modificações dos parâmetros ocorrem dentro de opções especializadas de MVS node, ainda na etapa de pré-processamento, como mostrado na Figura 26, permitindo ajustar o processo de treinamento, após a seleção da ferramenta. Necessitando sair do modo *simple* e selecionar o modo *expert*, para variar valores de parâmetros ou modificar a função kernel usada na aplicação.

Figura 26 - Parâmetros do kernel rbf.



Fonte: O próprio autor.

A partir dos conceitos obtidos em [40], é de fácil identificação o local de modificação dos padrões dos kernel aplicados na modelagem de MVS. Sendo possível executar modificações diversas entre o tipo de função do kernel usada para a transformação, sendo, portanto, aconselhável experimentar as várias opções de kernel e verificação de validações de acordo com a definição dos respectivos parâmetros.

Serão feitas as seguintes definições desses parâmetros com base em [40][23]:

Parâmetro de Margem (C): Controla o compromisso entre maximizar a margem e minimizar o termo do erro de treinamento. O valor normalmente deve estar entre 1 e 10, inclusive; o padrão é 10. O aumento do valor aprimora a precisão da classificação (ou reduz o erro de regressão) para os dados de treinamento, mas isso também pode levar ao ajuste excessivo.

Precisão de Regressão ($\boldsymbol{\varepsilon}$): Usado apenas se o nível de medição do campo de destino for contínuo. Faz com que erros sejam aceitos, desde que sejam menores que o valor especificado no preenchimento de $\boldsymbol{\varepsilon}$. Aumentar o valor pode resultar em modelagem mais rápida, mas à custa de perca de precisão.

RBF Gama(γ): Ativado apenas se o tipo de kernel estiver definido como RBF. O valor normalmente deve estar entre 3 / k e 6 / k, onde k é o número de campos de entrada. Por exemplo, se houver 12 campos de entrada, vale a pena tentar valores entre 0,25 e 0,5. Aumentar o valor melhora a precisão da classificação (ou reduz o erro de regressão) para os dados de treinamento, mas isso também pode levar ao ajuste excessivo (overfiting).

O modelo gerado através da simulação usando MVS cria um ou mais campos extras dependendo da aplicação. A seguir a Tabela 3 para a interpretação do novo campo criado após a simulação de MVS dentro do SPSS Modeler.

Tabela 3 - Descrição da classe do modelo.

Novo nome do campo	Descrição
$ S-Class	Valor para a classe prevista pelo modelo.
$SP-Class	propensão da classe SP para esta previsão (a probabilidade dessa previsão ser verdadeira, um valor de 0,0 a 1,0).

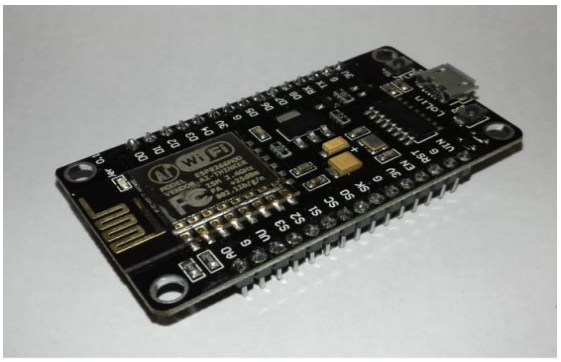
Fonte: O próprio autor.

4 MATERIAIS

4.1 System On Chip ESP 8266

O ESP8266, Figura 27, é um microcontrolador produzido pela empresa Espressif Systems. Este microcontrolador conta com 16 portas GPIO, comunicação I 2C, UART, SPI, um CPU de 32-bit com a arquitetura Xtensa, *clock* de 80 MHz podendo chegar a 160 MHz[43].

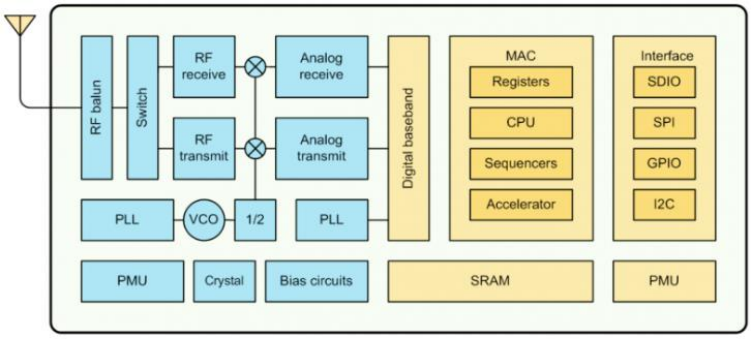
Figura 27 - ESP 8266.



Fonte: O próprio autor.

É um conjunto de alto desempenho, projetado para espaços pequenos com restrição de consumo de energia para plataformas móveis. Ele fornece a capacidade de incorporar Wi-Fi dentro de outros sistemas, podendo funcionar como aplicativo independente, com menor custo e com um mínimo de espaço. O diagrama de blocos do ESP8266 é ilustrado na Figura 28.

Figura 28 - Diagrama de Blocos ESP8266.



Fonte: Espressif systems.

O microcontrolador pode utiliza uma comunicação como I2C, SPI ou UART. A parte de RF do microcontrolador é formada pelos seguintes principais blocos: receptor, transmissor, gerador de *clock* de alta precisão, reguladores e gerenciador de energia.[43]

Para contornar os problemas gerados pelas condições adversas do canal de comunicação, são integrados no ESP8266 filtros de controle automático de ganho. Todos os componentes utilizados pelo gerador de *clock* estão integrados no microcontrolador.

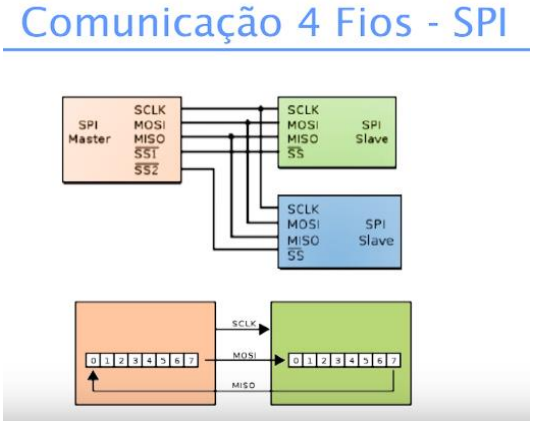
O principal motivo da escolha deste microcontrolador foi sua capacidade de processamento e forma de comunicação através do protocolo SPI.

4.1.1Comunicação SPI

É um protocolo de dados serial síncrono usado por microcontroladores para se comunicar com um ou mais dispositivos periféricos em alta velocidade em curtas distâncias. Também pode ser usado para comunicação entre dois microcontroladores. Com uma conexão SPI, sempre há um dispositivo mestre (um microcontrolador) que controla os dispositivos periféricos. A taxa de transmissão máxima é superior à do sistema de comunicação I2C.

A comunicação SPI possui algumas características básicas. Primeiramente os sinais de comunicação possuem uma direção fixa e definida, Figura 29. Isso significa que sempre existem dois transistores que definem o estado de um pin (*Push-Pull*). Esse recurso é uma das grandes diferenças entre outras comunicações seriais como I2C e *OneWire*, que possui o mesmo barramento de dados para os sinais de entrada e saída através do esquema de abertura [44].

Figura 29 - Comunicação SPI.



Fonte: Adaptado de [44].

Normalmente, existem três linhas comuns a todos os dispositivos:

- MISO (*Master In Slave Out*) - A linha *Slave* para enviar dados ao mestre, MOSI (*Master Out Slave In*) - A linha *Master* para o envio de dados aos periféricos,
- SCK (*Serial Clock*) - O pulso de *clock* sincroniza a transmissão de dados gerada pelo mestre.
- SS (*Slave Select*) - o pino em cada dispositivo que o mestre pode usar para ativar e desativar dispositivos específicos.

Além do *clock* e sinal de seleção esse tipo de comunicação possui potencial de fazer comunicação simultânea: basicamente à medida que os bits estão saindo do mestre(out), eles podem fazer o caminho inverso ao mesmo tempo. O mais comum é um trânsito de uma via somente. [44]. A Tabela 4 descreve o *mode spi* e como se comporta o clock:

Tabela 4 - Clock SPI.

Modo	CPOL	CPHA	Clock repouso	Borda p/ gravar
0	0	0	0	Subida
1	0	1	0	Descida
2	1	0	1	Descida
3	1	1	1	Subida

Fonte: Adaptado [44].

No qual CPOLé a polaridade do *clock* e CPHA a Fase do *clock*.

Desta forma se no repouso do *clock* está *low/high*, é possível identificar se o dado está disponível. A mudança do *clock* significa que se tem um dado para processar.

Micro *wire*, trata somente da comunicação em MODE 0 em *half duplex*, enquanto somente um dado transita com dados.

Quando o pino de *Slave Select* está baixo, ele se comunica com o mestre. Quando está alto, ignora o mestre. Isso permite que seja possível vários dispositivos SPI compartilhando as mesmas linhas MISO, MOSI e CLK.

Para escrever uma rotina de código para um novo dispositivo SPI, é necessário observar os seguintes quesitos:

- Conhecimento sobre qual é a velocidade máxima do SPI que o dispositivo pode usar. Isso é controlado pelo primeiro parâmetro em *SPISettings*. Se for usado um chip classificado em 15 MHz, usa-se 15000000.

- A entrada de dados é controlada pelo parâmetro *SPISettings*, MSBFIRST ou LSBFIRST. A maioria dos chips SPI usa o primeiro pedido de dados do MSB.

O padrão SPI é flexível e cada dispositivo o implementa de maneira um pouco diferente. Isso significa que se deve prestar atenção especial as amostras de dados do dispositivo ao desenvolver uma rotina.

4.2 Acelerômetro MEMS Adxl 345

Basicamente, todos os tipos de acelerômetros traduzem o sinal externo de aceleração em um deslocamento correspondente de sua massa móvel, também conhecida como massa inercial ou de prova. Este deslocamento pode ser detectado através de diferentes esquemas de medição, sendo que os mais comuns são: capacitivo, piezoelétrico, piezo resistivo, ressonante e óptico [45].

As principais especificações que devem ser consideradas na seleção de um acelerômetro são discutidas em seguida.

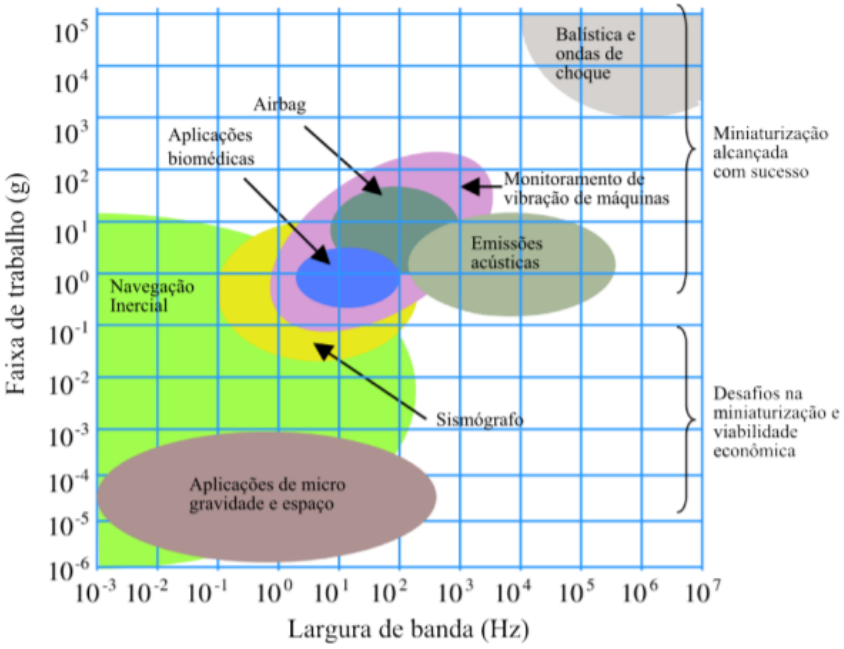
Sensibilidade: fator de escala de um sensor, medida em termos de mudança na saída para uma alteração no sinal de entrada. É uma referência à habilidade do sensor em detectar movimento e é normalmente especificada em mV/g.

- Largura de Banda ("*Bandwidth*"): é a faixa de frequências para a qual o sensor é aplicável. É normalmente especificado em Hertz (Hz), sendo tipicamente limitada a 1/5 da primeira frequência de ressonância [46]
- Estabilidade: define o quão constante é o sinal de saída em condições de entrada constantes, sendo a alteração na saída chamada de deriva ("*drift*"). Acelerômetros para uso em aplicações de alto desempenho devem apresentar alta estabilidade e por isso são muito mais caros.
- Resolução: menor nível de aceleração detectável pelo sensor e é limitada pelo nível de ruído do sensor. É normalmente especificada em mili-g (mg) ou micro-g (µg) [47].
- Alcance Dinâmico ("*Dynamic Range*"): é a faixa de valores de acelerações que podem ser medidas pelo sensor. O limite inferior é determinado pela resolução do dispositivo e o superior pela sua saturação, que é o ponto a partir do qual o sensor perde a linearidade na sua resposta;
- Confiabilidade: descreve a probabilidade de o dispositivo desempenhar adequadamente as funções, durante um período de tempo especificado e dentro de condições operacionais pré-estabelecidas. [47]

- Custo: tem importância menor para dispositivos de alto desempenho, no entanto, é talvez a característica mais importante em aplicações de consumo.

A Figura 30, ilustra as características do acelerômetro para diferentes aplicações:

Figura 30 - Faixa de trabalho x Largura de banda para diferentes aplicações de acelerômetros.



Fonte: Adaptado de [47].

As origens do uso da tecnologia do sistema micro-eletromecânico (MEMS) é datada de abril de 1954, quando um artigo de Smith (1954), foi publicado na Physical Review. Quando é descrita nas referências bibliográficas pela primeira vez certos efeitos sensíveis ao estresse no silício e no germânio, denominados piezo resistência. Em meados da década de 1950, os pesquisadores começaram a investigar se as mesmas tecnologias que produziram o transistor, que posteriormente revolucionou a incipiente indústria eletrônica, poderia ser aplicada aos sensores. [47]

Mostrando que os sensores eletromecânicos volumosos anteriormente bastante utilizados podem ser substituídos por dispositivos pequenos e robustos da mesma maneira que o transistor substituiu a válvula termiônica. O interesse na tecnologia de sensores de silício cresceu dramaticamente e, no final dos anos 60, vários pioneiros americanos comercializavam os primeiros sensores de pressão de silício.[47].

Os Acelerômetros do tipo MEMS (Micro-ElectroMechanical Systems) detiveram grande impacto comercial com aplicação nos mais diversos campos. Sua criação foi motivada além de tudo pelo avanço da indústria para atender a diferentes tecnologias. Os acelerômetros comerciais são baseados na medição das componentes cartesianas do vetor de aceleração gravitacional, comumente encontrados no mercado oferecendo boa resposta à aceleração dinâmica resultante de movimento, baixa demanda de energia e baixa tensão de excitação. [47]

O acelerômetro piezoelétrico é o mais recomendado para aplicações em Manutenção Preditiva [49], pois a resposta de frequência estende-se até dezenas de kHz, deslocamento e velocidade podem ser obtidos a partir da integração elétrica da aceleração e, medição de transientes é melhor relatada por aceleração do que deslocamento ou velocidade. São também comercializados os do tipo *Integrated Circuit Piezoelectric*, ICP, que possuem incorporado o circuito integrado de condicionamento do sinal, Figura 31. Em seguida, o sinal é enviado para o filtro *antialiasing* incorporado no CAD. Entretanto, o uso de acelerômetros é limitado devido ao relativo alto custo deste transdutor.[49][47]

Figura 31 - Acelerômetro ICP e uma unidade de condicionamento de sinal.

Fonte: Adaptado de [47].

Neste trabalho foi analisado o comportamento do acelerômetro MEMS ADXL345 fabricado pela empresa AnalogDevice, devido a facilidade de informações fundamentais obtidas através do datasheet, custo e operabilidade através da comunicação SPI.

ADXL 345

O ADXL345, ilustrado na Figura 32, é um MEMS versátil de 3 eixos, saída digital e facilmente configurado para faixas de trabalho diferentes. Faixas de medição e largura de banda selecionáveis além de detecção de movimento integrada e configurável o tornam adequado para a detecção aceleração em diversas aplicações. Ampla faixa de temperatura (-40 ° C a + 85 ° C) permitir o uso do acelerômetro mesmo em ambientes agressivos. [50]

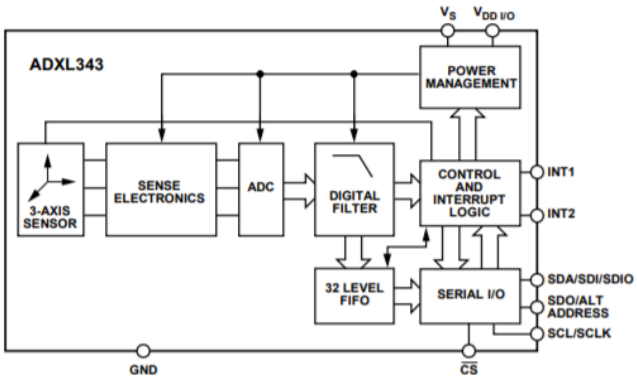
Figura 32 - Acelerômetro ADXL345.

Fonte: O próprio autor.

O ADXL345 mede a aceleração com alta resolução (13 bits) medição até ± 16 g. Os dados da saída digital são formatados de16 bits complementam e são acessíveis através de um SPI ou I2C. Ele pode medir a aceleração estática da gravidade em aplicações com sensor de inclinação, bem como a aceleração dinâmica resultante do movimento ou choque. [50]

O ADXL343 possui uma faixa de medição selecionável de ± 2 g, ± 4 g, ± 8 g, ou ± 16 g. Ele mede a aceleração dinâmica resultante de movimento ou choque e aceleração estática, como a gravidade, que permite que o dispositivo seja usado como um sensor de inclinação. Na figura 33 tem-se o diagrama de blocos do sensor.

Figura 33 - Diagrama de blocos ADXL345.

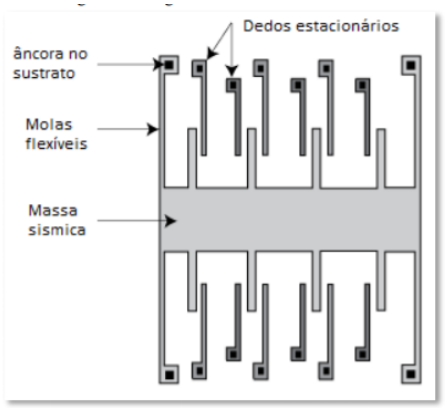


Fonte: Adaptado de [50].

O sensor é uma estrutura micro-usinada em superfície de polissilício construído em cima de uma pastilha de silicone. Molas de polissilício suspendem a estrutura sobre a superfície da bolacha e fornecer uma resistência contra forças devido à aceleração aplicada.

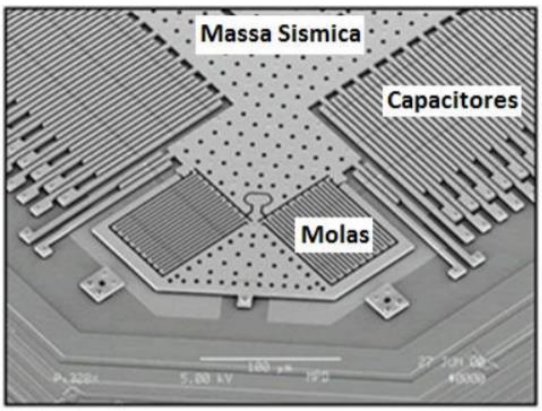
Agindo como um transdutor o qual permite transformar energia mecânica em energia elétrica, sendo sua fonte de informação a aceleração do sistema. É formado por três estruturas fundamentais, a massa sísmica, uma região de molas e as estruturas ou dedos capacitivos os dedos capacitivos se encontram nos dois lados da massa sísmica, Figuras 34 e 35, e tem a mesma capacitância na posição de equilíbrio mecânico. Ao se aplicar uma aceleração na massa sísmica que conectada fisicamente com as placas dos condensadores, altera a distância ou a superfície entre os dedos capacitivos, alterando a capacitância do condensador. Essa capacitância é proporcional à aceleração aplicada no sistema [51].

Figura 34 - Massas sísmicas e molas do MEMS.



Fonte: Adaptado de [48].

Figura 35 - Imagem microscópica do acelerômetro MEMS.



Fonte: Adaptado de [48].

A deflexão da estrutura é medida usando capacitores diferenciais que consistem em placas fixas independentes e placas fixadas a massa em movimento. A aceleração desvia a massa de prova e desequilibra o capacitor diferencial, resultando em uma saída do sensor cuja amplitude é proporcional à aceleração. Demodulação sensível à fase é usado para determinar a magnitude e a polaridade da aceleração.[48]

O acelerômetro capacitivo MEMS é governado pela seguinte expressão:

$$V_{sense} = \frac{2C_s}{2C_s + C_p + C_{gs} + C_{gd}} \cdot \frac{V_m}{{\omega_n}^2 d} \cdot a$$

Onde V_{sense} é a tensão de saída, C_s é capacitância nos dedos capacitivos, C_p é a capacitância parasita, C_{gs} e C_{gd} são duas capacitâncias do transistor MOS, V_m é amplitude do sinal modulado, d é a distância entre os dedos capacitivos do sensor, ω_n é a frequência de ressonância mecânica do transdutor e a é aceleração a ser medida.

Um acelerômetro típico MEMS tem frequência de ressonância 6 kHz, 1,5 μm na separação dos dedos capacitivos e uma sensibilidade de capacitância de 0,4 fF/g. Em um sinal de modulação de um 1 V de amplitude, a sensibilidade da tensão global é de cerca de 1 mV/g [52].

No caso da Manutenção Preditiva, as vibrações das máquinas rotativas geram sinais estacionários, que normalmente resultam da superposição de sinais senoidais com amplitudes e frequências diferentes. As características do acelerômetro que tem relação com esses dados são:

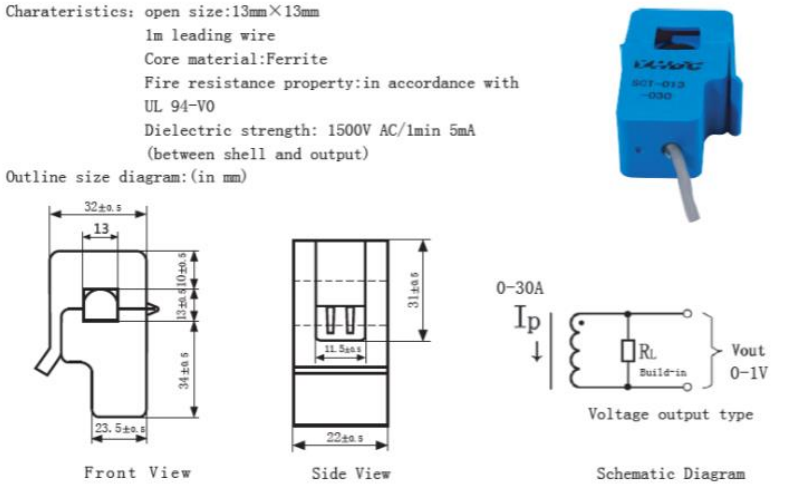
- Faixa de frequência (*Bandwith*) de medição.
- O limite de amplitude (*Acceleration Range*) que especifica o valor máximo de aceleração que pode ser medido com precisão, em “g”.

4.3 Sct-013

Sensor de corrente SCT-013-000 é uma ótima opção onde pode-se verificar correntes de até 100A e que não seja invasivo. Muito usado em projetos de automação residencial como medidores de corrente elétrica, proteção de motores AC/DC, iluminação e dentre outros.

Abaixo nas Figuras 36 e 37 tem-se as principais caraterísticas técnicas e funcionais.

Figura 36 - OVERVIEW sct013.

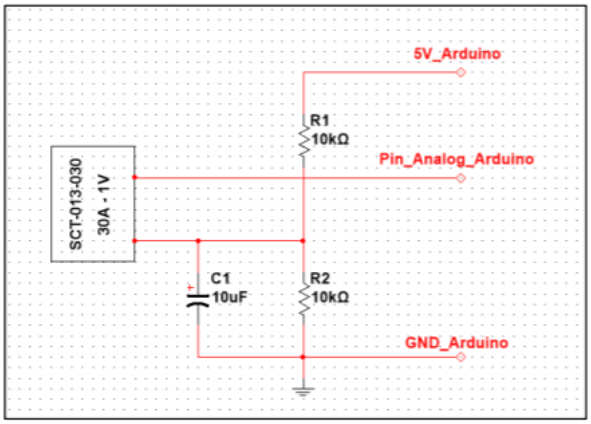


Typical table of technical parameters:

input current	output voltage	non-linearity	build-in sampling resistance (RL)
0-30A	0-1V	±1%	62Ω
turn ratio	resistance grade	work temperature	dielectric strength(between shell and output)
1800:1	Grade B	-25℃~+70℃	1500V AC/1min 5mA

Fonte: *Datasheet sct013.*

Figura 37 - Esquema de montagem sct013.



Fonte: *Datasheet sct013.*

5 METODOLOGIA

Os capítulos anteriores fundamentaram as aplicações que foram desenvolvidas nesta dissertação, trazendo os conceitos e o estado da arte que posiciona o trabalho frente aos processos e métodos mais modernos e eficientes., trazendo forte tendência para migração de aplicações para desenvolvimentos na nuvem. A busca constante por aplicações com alto nível de escalabilidade, que possam desenvolver as capacidades de acordo com a crescente demanda do volume de dados, independente da aplicação.

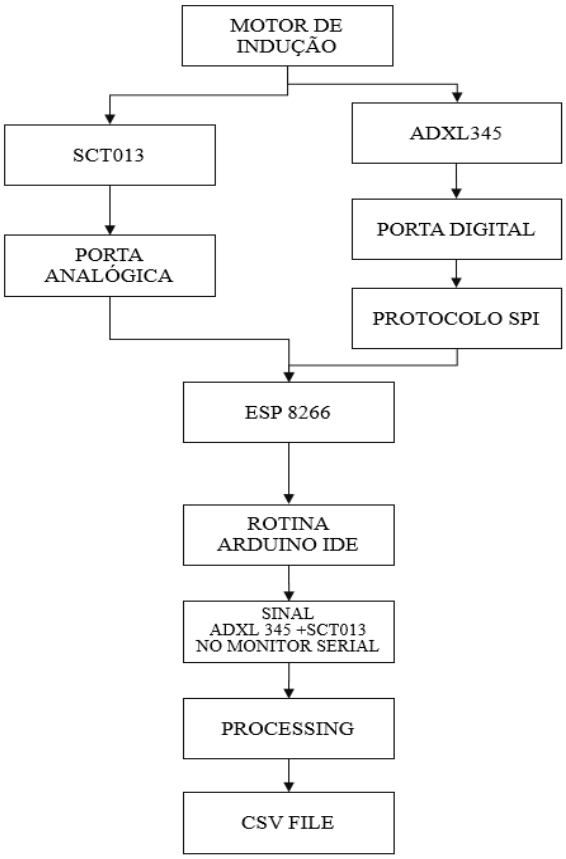
O contexto da evolução das técnicas de classificação com base em conceitos de AM traz a essência do desenvolvimento deste trabalho, onde se objetiva classificar diferentes condições do motor de indução através de ferramentas provisionadas em IBM Cloud®, para aplicação de MVS, o que permite que possam ser adquiridos os subsídios necessários para a configuração de um sistema de classificação com boa capacidade de generalização para encontrar as devidas classes avaliadas nos testes.

O capitulo é subdividido entre sistema de aquisição e sistema de implementação e discussão dos resultados obtidos para que melhor possam ser compreendidos as principais etapas do desenvolvimento deste trabalho.

5.1 Sistema de Aquisição

A metodologia deste trabalho consiste no treinamento de um modelo estatístico com base em dados de um motor de indução capturados por um sistema de aquisição embarcado, descrito pelo fluxo da Figura 38. Os dados obtidos foram os de aceleração nos eixos x, y e z e os valores de corrente na porta analógica do ESP8266.

Figura 38 - Fluxo do sistema de aquisição.



Fonte: O próprio autor.

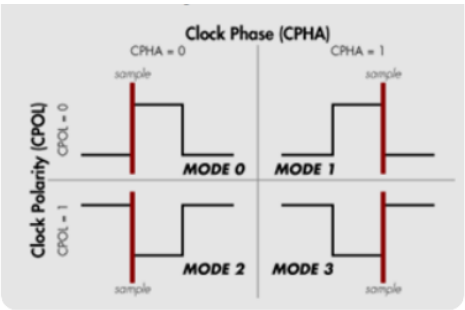
Os dados do acelerômetro ADXL345 foram obtidos através de uma comunicação serial via protocolo SPI. No caso deste trabalho, foi selecionada a frequência, de 3200hz, atendendo recomendações do fabricante do ADXL 345, que indica que esta faixa de frequência deve ser utilizada através do protocolo SPI.

Para prosseguir a configuração da comunicação entre o controlador e o sensor definiu-se o modo, sugerido pelo fabricante no *datasheet* do sensor. Esse modo define qual estado, dentre os

4 possíveis, que são obtidos a partir da combinação dos valores possíveis de polaridade do *clock*(CPOL), *high* ou *low*, e os valores possíveis da fase do clock (CPHA), *high* ou *low*.

A Figura 39, representa os 4 possíveis estados para esses parâmetros e o correspondente modo SPI.

Figura 39 – Descrição de modos SPI.



Fonte: Adaptado de [50].

O Mode 3 foi escolhido pela análise do diagrama de timing do ADXL345.

O diagrama ilustra que o estado *idle* do clock está no nível *high*, a borda de transmissão é a do nível *high* para nível *low* e a borda válida SDO é a borda de nível l*ow* para nível *high*.

A partir daí, deve-se configurar a ordem dos bits a serem enviados. A interface SPI pode enviar dados com o bit mais significativo primeiro ou o bit menos significativo primeiro, de acordo com o *datasheet* do ADXL345 deve-se escolher a configuração MSB (*most significant bit first*). A frequência de transferência de dados é então escolhida de forma a permitir o maior número de bit enviados na unidade de tempo, portanto a frequência de 5 KHz foi escolhida.

A seguir é informado o valor e o endereço do registro do qual se configura o range de serviço do acelerômetro e no caso deste trabalho foi o valor 0x01 no endereço 0x31, o modo de medição do acelerômetro ao escrever 0x08 no endereço 0x2D e o valor 0x0F no endereço 0x2C para configurar a taxa de Baud à 3200 Hz.

A leitura dos valores de aceleração começa no registro correspondente ao DATAX0 ou registro 0x32. O ADXL345 retorna valores de aceleração em 10 bits, sendo armazenados como bytes ou 8 bits. Portanto para se obter o valor completo de aceleração são combinados dois bytes para cada eixo. No caso deste trabalho, o valor de x é armazenado nas variáveis value1 e value2 e assim sucessivamente para o resto dos eixos como ilustra o fragmento de código na Figura 40.

Figura 40 - Fragmento do código implementado para aquisição de sinal do ADXL345

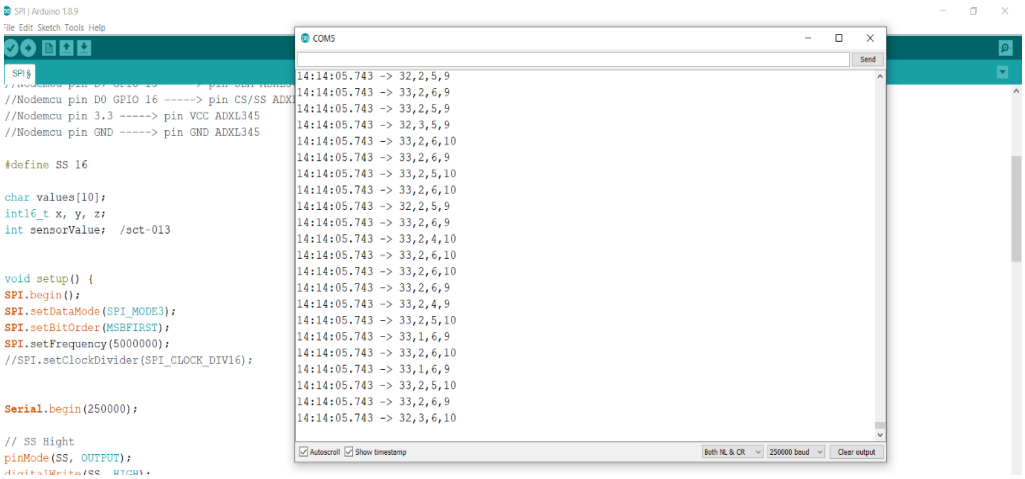
```
// DATAX0
readRegister(DATAX0, 6, values);

// 2Byte
x = ((int16_t)values[1] << 8) | (int16_t)values[0];
y = ((int16_t)values[3] << 8) | (int16_t)values[2];
z = ((int16_t)values[5] << 8) | (int16_t)values[4];
```

Fonte: O próprio autor.

O desenvolvimento do sistema objetivando a aquisição do sinal de corrente elétrica e vibração se deu através de um código escrito na Arduino IDE para o ESP8266 o qual faz a leitura dos registros do ADXL345 e a leitura dos valores obtidos na porta analógica na qual o SCT-013 é conectado e os envia para uma porta serial do computador. A partir desta etapa um código escrito no software Processing lê a porta em que o ESP8266 se encontra conectado e constrói um arquivo CSV, as rotinas implementas no arduíno IDE e Processing constam nos apêndices 1 e 2.

Após a concretização do sistema de monitoramento de vibração e correntes elétricas, foram realizados os primeiros testes do funcionamento do sistema. Segue a foto da visualização dos resultados obtidos, na Figura 41 , da captação de sinais dos sensores ADXL345+SCT-13 controlados do ESP 8266, tendo a visualização dos resultados no Serial da IDE(*Integrated Development Environment*) do Arduino.

Figura 41 - Monitor Serial do softaware Arduino IDE.



Fonte: O próprio autor.

Com a ferramenta de medição de vibração e corrente operando, tornou-se necessário elencar as condições em que seriam executados os testes em um sistema físico. Uma revisão

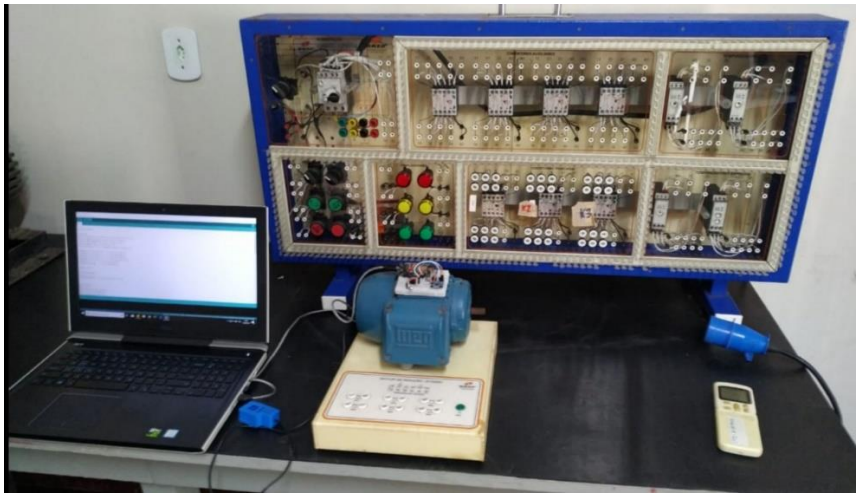
bibliográfica foi realizada para verificação dos principais tipos de falhas que podem ser avaliados, como verificado no capítulo 01, objetivando, fundamentar a seleção de condições de avaliação do motor para este estudo.

Desta foram selecionadas as seguintes condições do motor de indução trifásico, ajustadas as condições de realização dos testes em laboratório:

- Normal.
- Desbalanceamento do rotor.
- Alimentação de Tensão por Duas Fases.
- Desnível na base do motor.

As Figuras 42 e 43 mostram as instalações de onde ocorreram os testes.

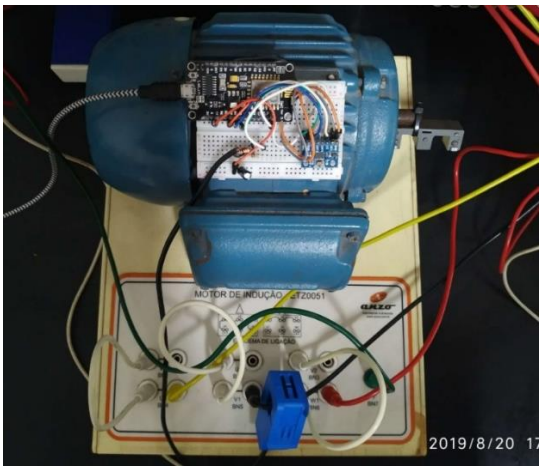
Figura 42 - Instalações do laboratório.



Fonte: O próprio autor.

Painel de instalação elétrica para ligação de partida direta no motor W22 IR3(Descrições do motor no anexo 4).

Figura 43 - Vista superior do sistema embarcado sobre o motor trifásico de indução W22-IR3.



Fonte: O próprio autor.

O ADXL345 fixado sobre a parte superior do motor, enquanto o SCT 013 fora conectado a um fluxo de fase de corrente de entrada do motor.

Para armazenar os dados em um arquivo no formato CSV, foi implementado o software *Processing*, que forneceu subsídios necessários para a indexação do fluxo dos dados a um novo arquivo, enquanto fosse requerida essa ação, em períodos de tempo determinados. Estes dados foram organizados em colunas em uma planilha sendo que para cada teste descrito foi gerado um arquivo.

A disposição das leituras do sistema de monitoramento de vibração e corrente gravou os dados de acordo com a Figura 44.

Figura 44 - Dados obtidos e organizados no arquivo CSV.

	A	B	C	D	E	F	G
1	-9	16	10	0			
2	2	-16	24	0			
3	0	-11	39	0			
4	-10	22	12	0			
5	9	-42	42	0			
6	-8	16	17	0			
7	-4	9	12	0			
8	5	-34	44	0			
9	-4	7	27	0			
10	2	-15	27	0			
11	6	-37	42	0			
12	-7	17	15	0			
13	6	-32	39	0			
14	1	-19	42	0			
15	-2	10	14	0			
16	5	-25	34	0			
17	-6	3	28	0			
18	-7	12	13	43			
19	-5						
20	8	-32	41	104			
21	-7	13	17	150			
22	9	-32	39	209			
23	-8	10	18	291			

Fonte: O próprio autor.

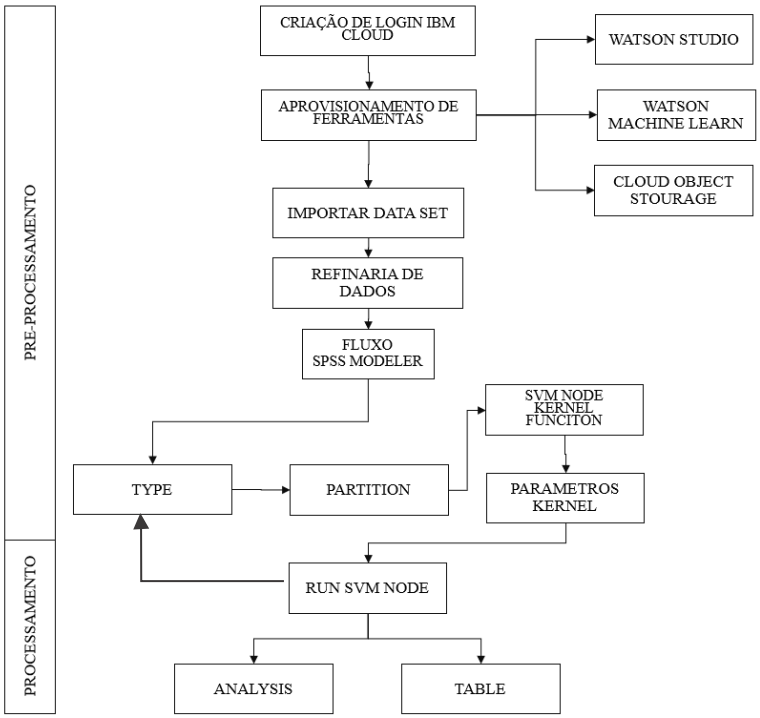
As aquisições dos dados foram realizadas, salvas em formato, sendo as três primeiras colunas, representação de dados obtidos de x, y e z respectivamente, e a quarta coluna representa os

dados de fluxo da corrente elétrica obtidos da leitura da porta analógica do ESP8266 conectado ao sensor SCT013. E desta forma foi concebido o material fundamental para dar sequência à etapa de implementações em computação na nuvem.

5.2 Sistema de Implementação

Após ter êxito no armazenamento de dados dos testes efetuados no motor de indução em diferentes condições (normal, alimentação de tensão por duas fases, desbalanceamento do rotor e desnível na base). Seguiu-se o fluxo para etapa de implementação (Figura 45), iniciando a utilização da infraestrutura fornecida por IBM Cloud: Watson Studio, SPSS Modeler e Watson Machine Learning. A implementação da ferramenta estatística MSV objetiva através dos resultados obter a correlação entre os dados e avaliar a acurácia das análises frente ao modelo preditivo gerado, possibilitando a identificação de características dos dados inerentes a cada teste físico realizado. Na figura o fluxo de implementação.

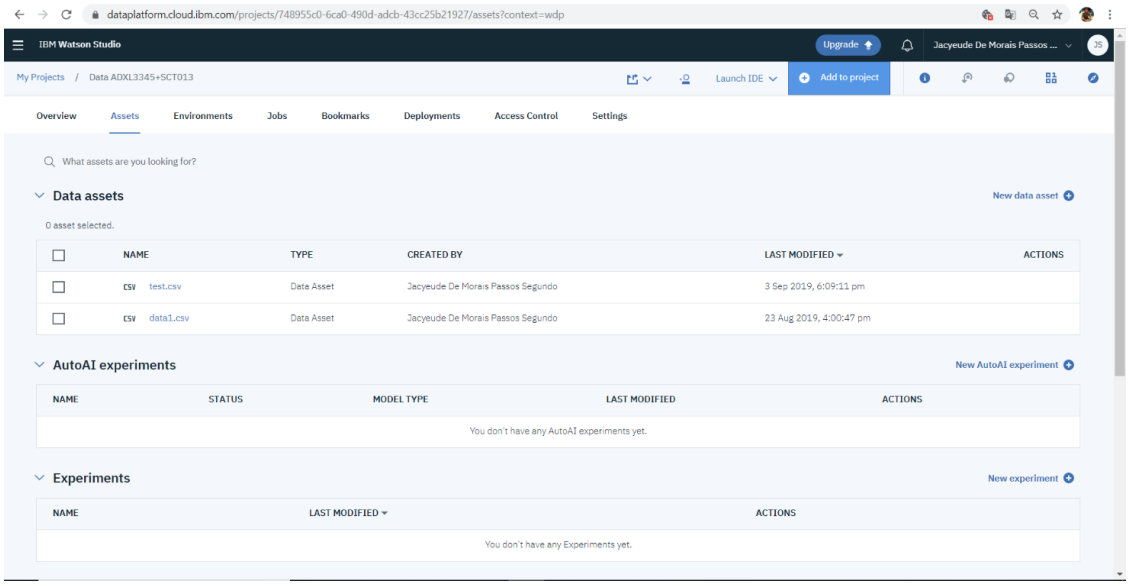
Figura 45 - Fluxograma do sistema implementado para utilizar o SPSS Modeler.



Fonte: O próprio autor.

A Figura 46 mostra a interface de Watson Studio, que disponibiliza uma aba de criação de um novo projeto e *gaps* de adição de novos *assets types* onde se encontra o *modeler flow*, que dá acesso à SPSS Modeler, a ferramenta de mineração de dados selecionada para implementar algoritmos preditivos que auxiliam na identificação de características entre os dados dos testes realizados no motor.

Figura 46 - Interface inicial de Watson Studio.



Fonte: O próprio autor.

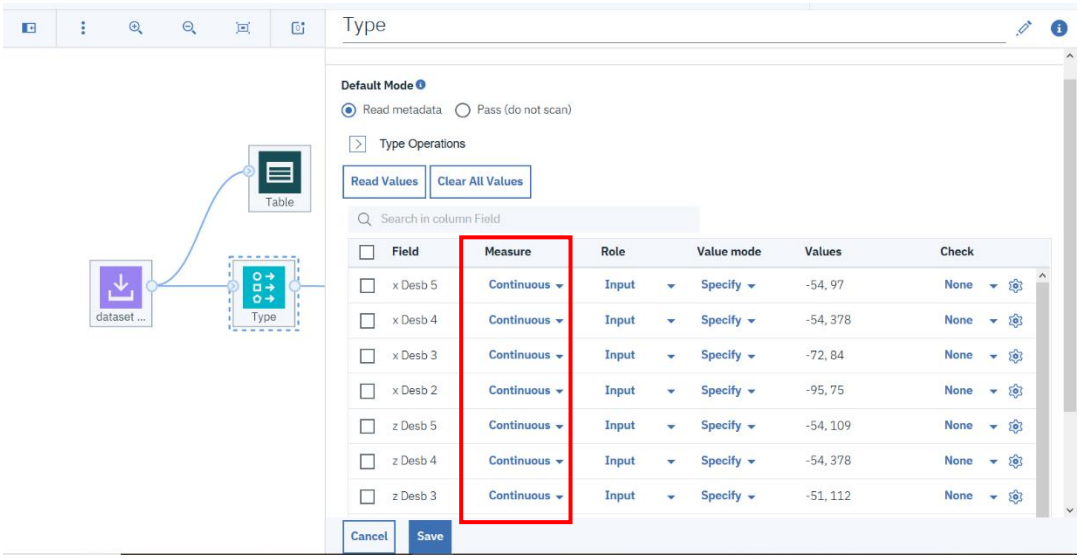
A Figura 46, mostra a interface de Watson Studio, de onde dar-se seguimento às etapas, sendo necessário que se tenha conhecimento de quais diretrizes devem ser tomadas de acordo com as proposições de estudo, pois cada estudo de caso requer ajustes diferentes em etapas anteriores ao processamento dos dados. Para este trabalho, a parte subsequente ao upload de dados é verificar, através de etapas de preparação de dados, para deixá-los em condição ideal de aplicação do algoritmo de MVS, esta etapa é compreendida como pré-processamento.

5.2.1 Pré-processamento

É importante ressaltar que o conjunto de dados do estudo de caso precisou passar por etapas de pré-processamento utilizando a refinaria de dados (retirada de espaços vazios, identificação e seleção da quantidade de dados do arquivo farão parte das análises), bem como foi utilizado o nó

type do SPSS Modeler para transformar o conjunto de dados, organizando a leitura das colunas e descrevendo as colunas em valores contínuos, Figura 47. A etapa de preparação é muito relevante pois interpretação dos valores das leituras dos dados organizados em colunas deve ser compreendida como valores numéricos, para este caso em especifico, para esta aplicação em MVS, faz-se a seleção em *Measure* do set *Continuous*, para a interpretação dos valores das colunas. Desenvolvendo uma organização prévia à aplicação que será realizada.

Figura 47 - Configuração em *type*.

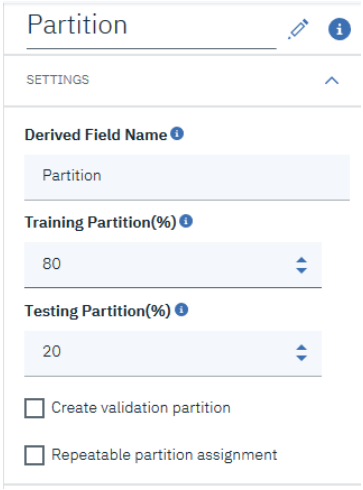


Fonte: O próprio autor.

Ainda dentro das configurações de *type*, é necessário na opção *role* selecionar o alvo da predição (*field target*), selecionando a coluna que será confrontada com campos específicos selecionados como entradas (*field inputs*), que serão utilizados na simulação.

O nó *partition* divide o banco de dados em duas partes, configurando a simulação em etapas de treino e teste. Utilizando 80 % do banco de dados para treino e 20% para testar o modelo, viabilizando uma verificação da capacidade de generalização do classificador MVS através do teste dos dados com o modelo gerado a partir do treinamento. A Figura 48 ilustra a divisão do banco de dados em *partition*.

Figura 48 - Configuração de particionamento dos dados.



Fonte: O próprio autor.

5.2.2 Execução do método

Nesta seção, serão descritos os resultados dos experimentos que tem por finalidade validar a metodologia descrita na seção anterior. Para esta finalidade gravação de sinais dos sensores em arquivo CSV e upload na infraestrutura da IBM Cloud® em Watson Studio, torna-se necessário desenvolver um fluxo de processamento dentro de SPSS Modeler.

Nesta etapa, ocorre a mineração de dados propriamente dita, momento em que se aplica um algoritmo para extração de conhecimento. No caso deste trabalho foram utilizados algoritmos de classificação, dentro de atividades preditivas, selecionando na aba de modelagem, o MVS. A implementação do algoritmo necessita de ajustes nos dados de entrada, como seleção de inputs, target além da seleção do kernel após selecionar o nó MVS na aba de *Modeling* (Figura 50).

A confiabilidade da análise dados é garantida por uma precisão de acurácia dos testes realizados após o treinamento do modelo preditivo. A utilização das funções kernel deve ser avaliada, tornando-se indispensável para visualização das tendências de classificação das condições a partir da utilização das funções kernel disponibilizadas. Os resultados obtidos através do nó *Alanylsis* indicam as características do modelo gerado, que podem ser verificadas, principalmente através da análise do índice de correlação linear que indica proximidade ou distanciamento do modelo em relação aos dados selecionados como alvo.

Os data sets, em formato CSV utilizados, foram estruturados de forma a organizar as etapas de verificação sobre cada uma das condições ensaiadas no motor de indução. Foram utilizados arquivos contendo informações de dois ensaios distintos, gerando um arquivo para analisar cada condição. Os arquivos foram organizados dessa forma para que possam ser utilizados dados que representem momentos de condições diferentes, usados como amostras de treinamento e teste para MVS, objetivando verificar a capacidade de identificação das classes através da aplicação da metodologia proposta.

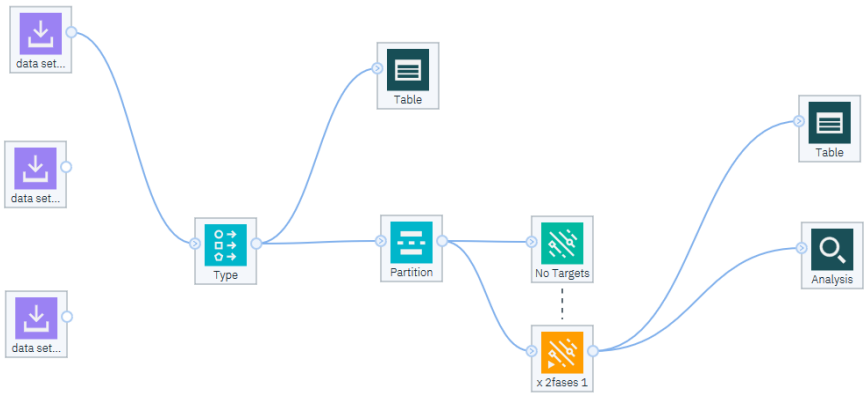
Os arquivos usados contendo a base de dados das simulações estão descritos abaixo:

Data set 1 :Condições Normais
Data set 2: Condições de Desbalanceamentos
Data set 3: Condições de ligação de duas fases
Data set 4 :Condições de Desnível na base

5.3 Modelos Preditivos Gerados

A Figura 49 contém um fluxo da estruturação no SPSS Modeler para a aplicação de MVS realizadas para o base de treinamento e teste dos dados, gerando um novo nó(em amarelo na figura 49), que é o modelo da predição gerado através das funções de decisão de MVS, a verificação desse resultado gera uma coluna $*filename*,(Figura 61), que deve ser investigada com a finalidade de extrair conhecimento das informações dispostas.

Figura 49 - Fluxo desenvolvido para aplicação e avaliação de resultados de MVS.



Fonte: O próprio autor.

Os bancos de dados abaixo descritos contêm os valores de x y z do acelerômetro e os dados de corrente das condições ensaiadas no sistema físico, de modo que cada arquivo contém dados

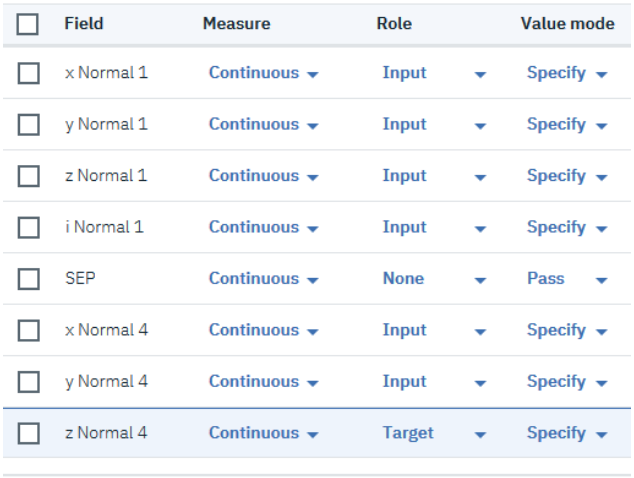
de dois ensaios de condições distintas para serem estudados individualmente, cada arquivo de *dataset* possui 10000 linhas de dados.

Usando as amostras de cada *dataset* para gerar o modelo e uma amostra separada para testá-lo, sendo possível obter uma boa indicação de quão bem o modelo será generalizado para conjuntos de dados que são semelhantes aos dados usados para as simulações.

5.3.1 *Dataset* 1 – Condição Normal

Para avaliar se MVS é capaz de reconhecer uma característica da condição de funcionamento normal do motor de indução foram selecionadas como alvo(*target*) a coluna "z Normal 1" e outras sete colunas de representação da condição do funcionamento normal foram selecionadas como entradas(*inputs)*, para a execução de MVS(Figura 50).

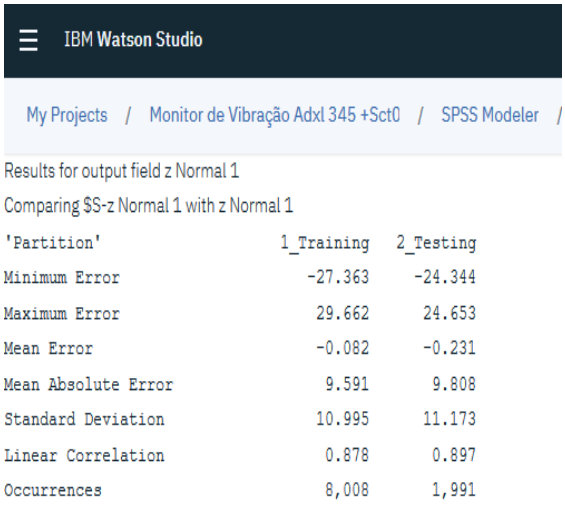
Figura 50 - Selecionando o alvo da predição.



Fonte: O próprio autor.

Após rodar a aplicação de MVS, é possível notar que foi criada uma nova coluna "$-z Normal4", e através de *Analysis*, foram obtidos os resultados de treino com 80% dos dados, e com 20% utilizado para testar o modelo, visando avaliar as características encontradas ainda na fase de treino e validar com o teste do modelo. Para avaliar a performance de MVS foram realizadas experimentações com os 4 tipos de funções kernel, os resultados seguem:

Figura 52 - Resultados kernel rbf.



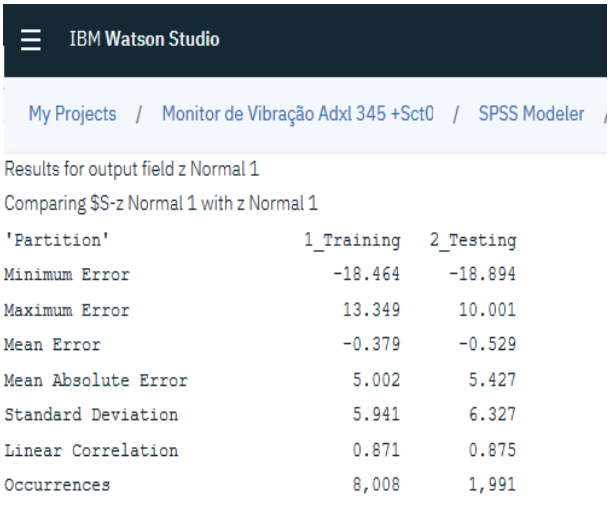
Results for output field z Normal 1

Comparing $S-z Normal 1 with z Normal 1

'Partition'	1_Training	2_Testing
Minimum Error	-27.363	-24.344
Maximum Error	29.662	24.653
Mean Error	-0.082	-0.231
Mean Absolute Error	9.591	9.808
Standard Deviation	10.995	11.173
Linear Correlation	0.878	0.897
Occurrences	8,008	1,991

Fonte: O próprio autor.

Figura 51 - Resultados kernel polynomial.



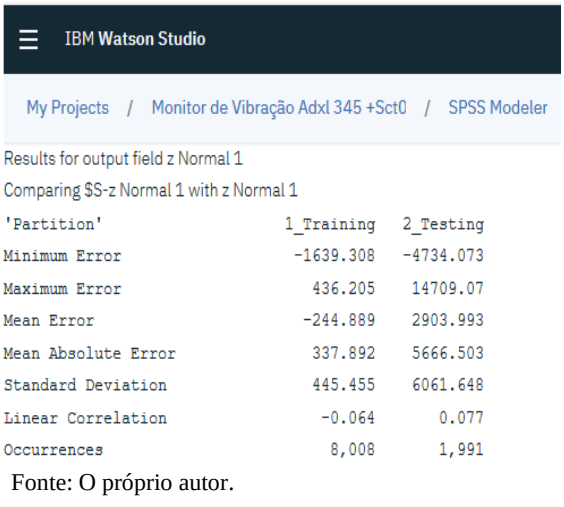
Results for output field z Normal 1

Comparing $S-z Normal 1 with z Normal 1

'Partition'	1_Training	2_Testing
Minimum Error	-18.464	-18.894
Maximum Error	13.349	10.001
Mean Error	-0.379	-0.529
Mean Absolute Error	5.002	5.427
Standard Deviation	5.941	6.327
Linear Correlation	0.871	0.875
Occurrences	8,008	1,991

Fonte: O próprio autor.

Figura 54 - Resultados kernel sigmoid.



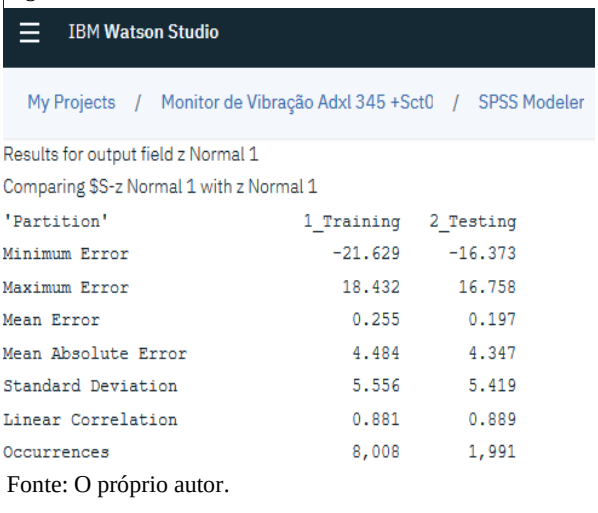
Results for output field z Normal 1

Comparing $S-z Normal 1 with z Normal 1

'Partition'	1_Training	2_Testing
Minimum Error	-1639.308	-4734.073
Maximum Error	436.205	14709.07
Mean Error	-244.889	2903.993
Mean Absolute Error	337.892	5666.503
Standard Deviation	445.455	6061.648
Linear Correlation	-0.064	0.077
Occurrences	8,008	1,991

Fonte: O próprio autor.

Figura 53 - Resultados kernel linear.



Results for output field z Normal 1

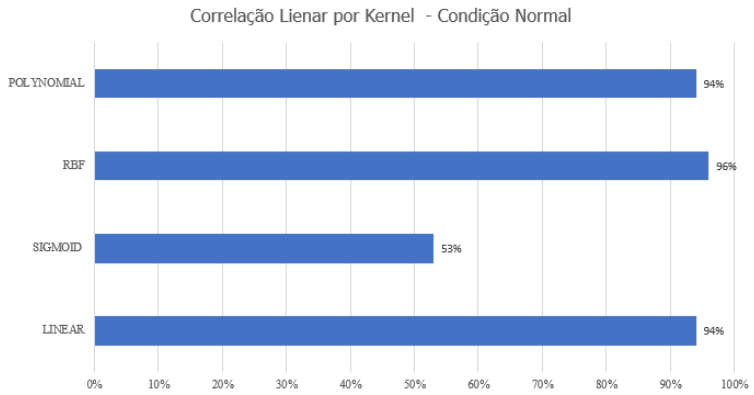
Comparing $S-z Normal 1 with z Normal 1

'Partition'	1_Training	2_Testing
Minimum Error	-21.629	-16.373
Maximum Error	18.432	16.758
Mean Error	0.255	0.197
Mean Absolute Error	4.484	4.347
Standard Deviation	5.556	5.419
Linear Correlation	0.881	0.889
Occurrences	8,008	1,991

Fonte: O próprio autor.

Avaliando que as aplicações de MVS foram feitas utilizando os parâmetros γ, ε e C em todos os kernels em *default*, observando que o ajuste desses parâmetros pode resultar em diferentes resultados para cada simulação, mostrou que o kernel polynomial foi o que apresentou o melhor desempenho em classificar a situação indicada, de acordo com o valor obtido no índice de correlação linear. A seguir um gráfico de desempenho das simulações, na Figura 55.

Figura 55 - Correlação linear por kernel - condição normal.



Fonte: O próprio autor.

Os resultados obtidos de *Analysis*, mostram que a correlação linear entre os valores dos dados das colunas geradas como modelos e os alvos selecionados alcançou valor máximo de 0,996, de correlação linear, utilizando o kernel, sendo que este valor varia entre -1 e 1, mostrando neste caso excelente índice de correlação de entre resultados previstos e os alvos(*targets*) indicados. Em termos de correlação linear foi obtido nesse caso 96% de correlação, utilizando o kernel rbf, notando que polynomial e linear também obtiveram valores de boa acurácia.

Sendo desta forma assertivo a confirmação de que o modelo gerado, caracteriza a situação a que o motor o motor foi ensaio, encontrando o padrão da condição de normalidade do motor de indução.

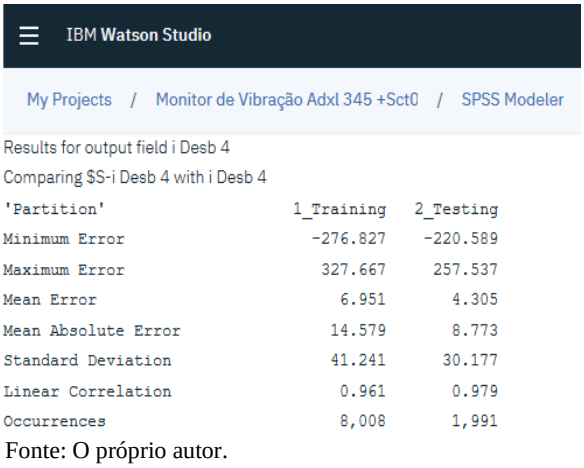
5.3.2 *Dataset* 2 – Condição de Desbalanceamento

Para avaliar o desempenho de MVS sobre o banco de dados de desbalanceamento, foi selecionado como alvo(*target*) a coluna “i Desb 4”, caracterizando o fluxo da corrente alternada em uma das fases, e outras sete colunas de representação de desbalanceamento foram selecionadas como *input* para serem comparadas com *target*, através de *Analaysis*, obtendo os seguintes resultados:

Figura 57 - Resultados kernel rbf.

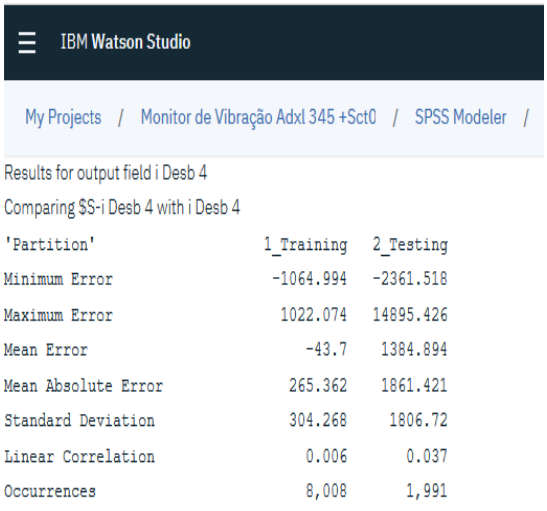
Fonte: O próprio autor.

Figura 56 - Resultados kernel polynomial.



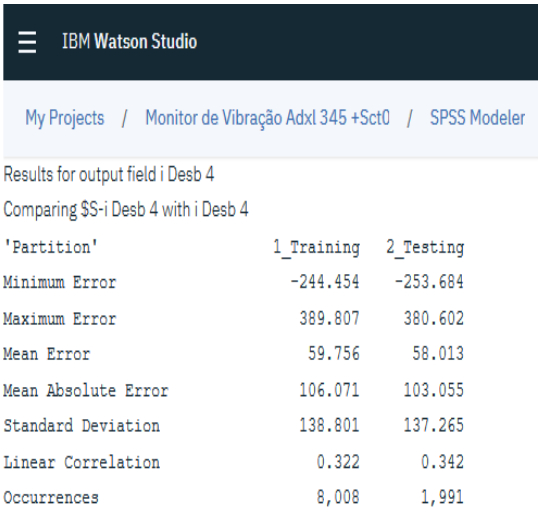
Fonte: O próprio autor.

Figura 58 - Resultado kernel sigmoid, condição de desbalanceamento.



Fonte: O próprio autor.

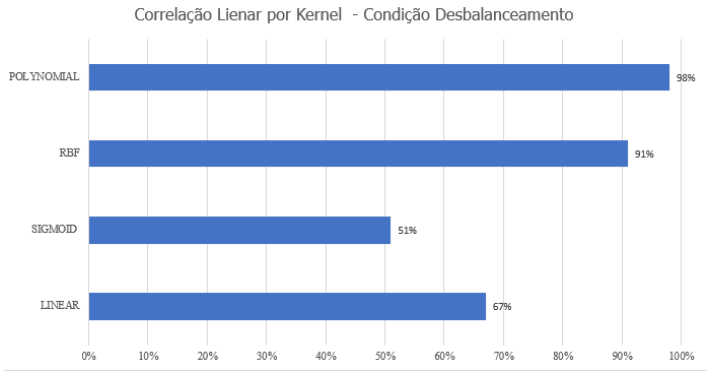
Figura 59 - Resultado kernel linear, condição de desbalanceamento.



Fonte: O próprio autor.

Os resultados visualizados em *Analysis*, retirados das simulações com cada função kernel, apresentam o desempenho do classificador MVS nos campos de treino e teste do modelo gerado para a classificação da condição especificada do motor de indução. Mostrando as tendências de correlação entre resultados previstos e o alvo(*target*) indicado na Figura 60. Tem-se nesse caso como desempenho mais eficiente de classificação utilizando o kernel polynomial, com 0,979. As descrições de valores dos erros representam os limites do dimensionamento na separação das classes.

Figura 60 - Correlação desbalanceamento verificar polynomial.

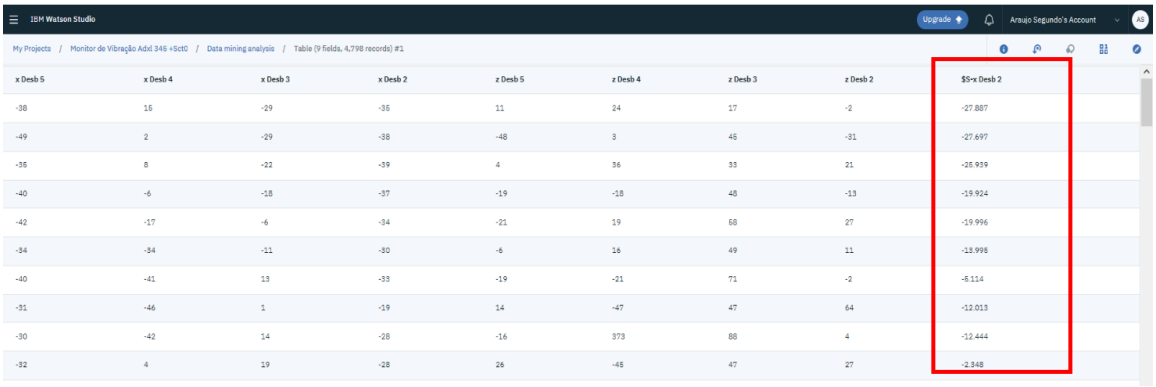


Fonte. O próprio autor.

Sendo satisfatório a acurácia de 98% entre os valores do modelo gerado “$ i Desb4” comparado com “i Desb4”, encontrada na utilização do kernel polynomial, seguido de rbf com 91%. A acurácia estima a capacidade de o classificador reconhecer corretamente uma amostra, caracterizando a situação a que o motor o motor foi ensaio, encontrando o padrão da condição de desbalanceamento.

Através da utilização do no *table*, é possível visualizar os valores gerados que compõe o modelo preditivo, na coluna “$ i Desb4”, Figura 61.

Figura 61 - Amostra de valores do modelo gerado.



Fonte: O próprio autor.

Satisfazendo um dos objetivos deste trabalho, classificando os valores de entrada para possa gerar um modelo preditivo de uma característica através dos dados de treinamento fornecidos, e testando o modelo com 20% dos valores do banco de dados fornecido. Sendo possível ratificar a eficácia de MVS para encontrar padrões.

5.3.3 *Dataset* 3 - Condições de alimentação por duas fases

Para esse *dataset* foram selecionadas as colunas “y 2fases 5” como alvo e outras sete colunas como inputs da condição de alimentação por somente duas fases para as simulações de MVS. A seguir os resultados obtidos através de *Analysis.*

Figura 63 - Resultados kernel rbf.

Results for output field z 2fases 1

Comparing $S-z 2fases 1 with z 2fases 1

'Partition'	1_Training	2_Testing
Minimum Error	-31.367	-31.423
Maximum Error	42.405	42.605
Mean Error	0.569	0.305
Mean Absolute Error	9.38	9.419
Standard Deviation	11.726	11.756
Linear Correlation	0.702	0.834
Occurrences	8,008	1,991

Fonte: O próprio autor.

Figura 62 - Resultados kernel polynomial.

Results for output field z 2fases 1

Comparing $S-z 2fases 1 with z 2fases 1

'Partition'	1_Training	2_Testing
Minimum Error	-33.162	-26.144
Maximum Error	36.559	41.185
Mean Error	-0.147	-0.24
Mean Absolute Error	7.261	7.1
Standard Deviation	8.707	8.195
Linear Correlation	0.72	0.771
Occurrences	8,008	1,991

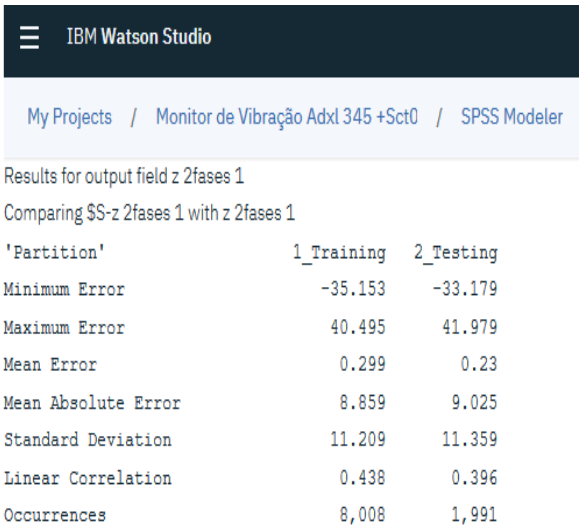
Fonte: O próprio autor.

Figura 65 - Resultados do kernel sgmoid, Condição de duas fases.

'Partition'	1_Training	2_Testing
Minimum Error	-35.153	-33.179
Maximum Error	40.495	41.979
Mean Error	0.299	0.23
Mean Absolute Error	8.859	9.025
Standard Deviation	11.209	11.359
Linear Correlation	0.438	0.396
Occurrences	8,008	1,991

Fonte: O próprio autor.

Figura 64 - Resultados do kernel linear, condição de duas fases.

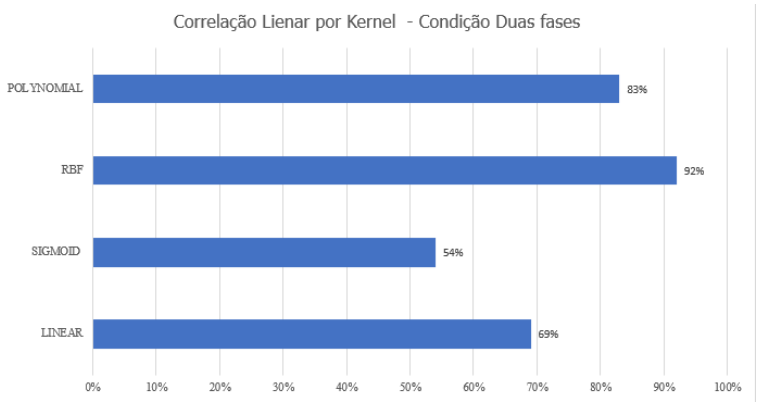


'Partition'	1_Training	2_Testing
Minimum Error	-35.153	-33.179
Maximum Error	40.495	41.979
Mean Error	0.299	0.23
Mean Absolute Error	8.859	9.025
Standard Deviation	11.209	11.359
Linear Correlation	0.438	0.396
Occurrences	8,008	1,991

Fonte: O próprio autor.

Os resultados obtidos de *Analysis*, mostraram que o kernel polynomial apresentou melhor performance de treino, onde foi obtido 0.953, mostrando a tendência de correlação entre resultados previstos e o alvo(*target*) indicado. As descrições de valores dos erros representam os limites do dimensionamento do hiperplano de margem máxima de separação das classes. A seguir na Figura 66, as representações das correlações obtidas para os kernels utilizados.

Figura 66 - Correlação linear por kernel - condição duas fases.



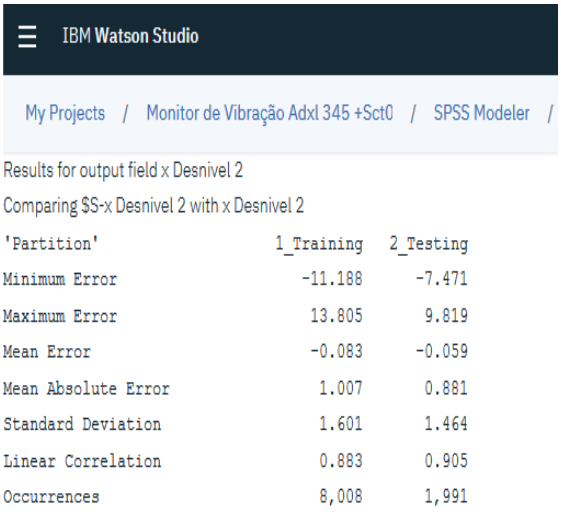
Fonte: O próprio autor.

Portanto, a capacidade de o classificador reconhecer corretamente a amostra selecionada é satisfatória, sendo rbf o kernel que obteve melhor performance com 0.83 de correlação entre dados previstos e indicados. Definindo a localização da característica do teste físico de funcionamento do motor com alimentação de somente duas fases.

5.3.4 *Dataset* 4 - Condições de Desnível na base

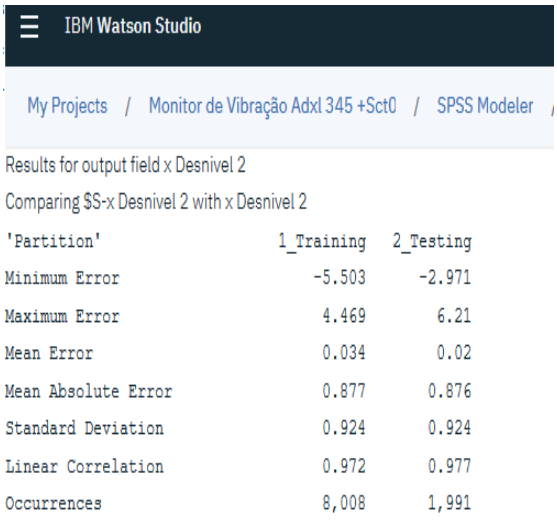
Para avaliar o reconhecimento de uma classe das condições de desnível ensaiadas no motor de indução foram selecionadas como alvo(*target*) a coluna “x Desnível 2” e outras sete colunas de representação da condição de desnível foram selecionadas como entradas(*inputs)*, para a execução de MVS. Através de *Analaysis*, foram obtidos os seguintes resultados. (Figuras 67,68,69 e 70)

Figura 68 - - Resultados do kernel Rbf, condição Desnível.



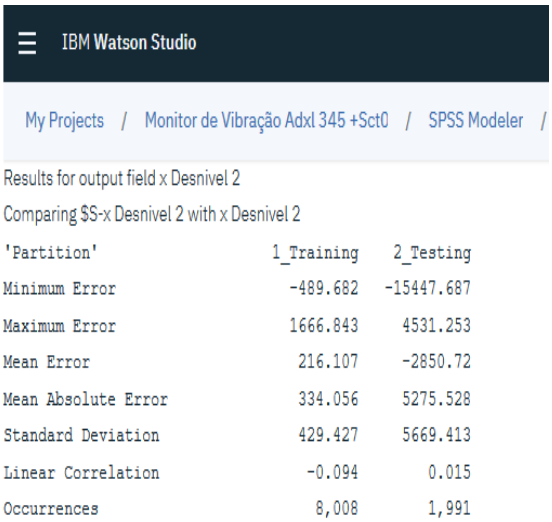
Fonte: O próprio autor.

Figura 67 - Resultados do kernel Polynomial, condição Desnível.



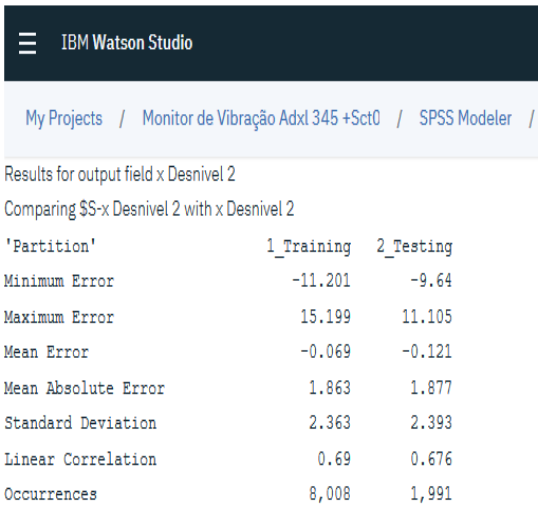
Fonte: O próprio autor.

Figura 69 - Resultados do kernel Sigmoid, condição Desnível.



Fonte: O próprio autor.

Figura 70 - Resultados do kernel Linear, condição Desnível.

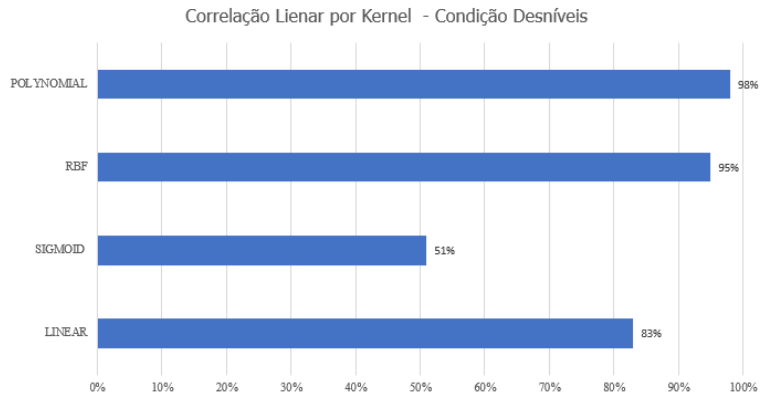


Fonte: O próprio autor.

Os resultados obtidos de *Analysis*, mostram que foi obtida correlação linear mais eficiente obtida foi de 0,977, mostrando a tendência de correlação positiva entre resultados previstos

e o alvo(*target*). Indicando que foi obtido, neste caso, 98% de correlação, no exemplo em que é usado o kernel polynomial, junto a rbf que atingiu 95% em termos de correlação. Segue o desempenho na Figura 71.

Figura 71 - Gráfico de desempenho em termos de correlação linear, condição Desníveis.



Fonte: o próprio autor.

Notando-se que a aplicação de MVS conseguiu classificar com alta correlação entre os valores reais e os valores previsto para a característica em questão, sendo novamente assertivo em classificar a classe que representa uma condição do motor de indução, neste caso a de desnível na base.

5.4 Discussão dos resultados obtidos no estudo sistema de monitoramento de vibração e correntes elétricas aplicado em IBM® Cloud

Na sessão anterior foram apresentados os resultados obtidos para análises de desempenho de MVS para auxiliar na decisão de qual conjunto de funções executou de maneira mais assertiva a criação de modelos com base em características de diferentes classes em diferentes bancos de dados simulados.

Nota-se que a oferta da escolha das funções de kernel condiciona os dados a diferentes performances de processamento. Podendo ser possível que sejam necessários vários testes empíricos para que se possa obter a melhor performance das funções, onde a característica de generalização do modelo deve ser mantida, para que desta forma MVS classifique modelos entre as classes avaliadas.

Neste estudo de caso, foram utilizados os valores dos parâmetros de kernel em modo *default*, de acordo com dados da Tabela 5:

Tabela 5 - Valores dos parâmetros dos kernels aplicados.

	RBF	POLYNOMIAL	SIGMOID	LINEAR
γ	0.1	0.5	0.5	-
ε	1	0.1	1	0.1
C	10	1	0.1	1

Fonte: O próprio autor.

Desta forma foi possível observar o desempenho de classificação proposto por MSV conseguiu com alto nível de correlação obter modelos preditivos acerca das características selecionadas, mostrando que o kernel polynomial obteve desempenho superior, como mostrado na Tabela 6.

Tabela 6 - Avaliação da classificação executada nos testes de MVS.

Condição	Maior índice de Correlação Linear	Kernel
Normal	96%	Rbf
Desbalanceamento	98%	Polynomial
Duas fases	92%	Rbf
Desnível na base	98%	Polynomial

Fonte: O próprio autor.

A utilização do método de classificação MVS, provou ser bastante robusto e eficiente mediante os resultados das classificações obtidas, satisfazendo aos objetivos deste trabalho.

Notando-se determinada sensibilidade das diferentes acurácias através de ajustes dos kernels MVS para gerar como saída um modelo de predição. Sendo útil esta contribuição que avaliou com elevados índices de relação entre as condições avaliadas. Todos os ensaios tiveram os dados particionados, objetivando o teste, para avaliação da qualidade do modelo desenvolvido. Nas Figuras 52, 57,63 e 68, representam um ensaio de cada condição respectivamente. é possível notar que os valores de treino e teste praticamente não sofreram alteração, validando as informações de treino. De certo que para este estudo de caso os kernels rbf e polymonial atenderam as necessidades de classificação entre classes, de acordo com os testes realizados e descritos nos tópicos anteriores.

6 CONCLUSÃO E TRABALHOS FUTUROS

6.1 Conclusão

Nessa dissertação foi proposta uma metodologia para classificar quatro condições diferentes de funcionamento de um motor trifásico de indução, através do método de classificação MVS disponível no SPSS Modeler, onde foram gerados diferentes modelos que apresentaram bons valores de acurácia para a determinação de cada condição do motor de indução.

A aplicação da metodologia de classificação de MVS mostrou capacidade de aprendizado a partir de um conjunto de dados de treinamento, onde a metodologia busca por uma hipótese(alvo), em um espaço de possíveis hipóteses, sendo capaz de descrever as relações entre os objetos e gerar um modelo que melhor se ajuste aos dados de treinamento fornecidos. Os dados das simulações foram particionados em 80% do conteúdo para treinamento e 20% para testar o modelo gerado.

Os resultados obtidos nas simulações no SPSS Modeler mostraram que a seleção distinta dos tipos e parâmetros das funções *kernel* aplicadas aos exemplos, produzem diferentes valores de acurácia dos modelos gerados, sendo rbf e polynomial as funções *kernel* que apresentaram desempenho mais elevado para a classificação entre os *kernels* testados, chegando a 98% de acurácia na classificação da classe normal do motor de indução. Esta metodologia pode ser bastante útil em situações onde o pesquisador não conhece a natureza dos dados a serem classificados, sendo obrigado a empregar métodos empíricos para configurar os o classificador MVS.

O método de MVS foi aplicado com os parâmetros das funções *kernel* em modo *default,* como mostrado na Tabela 5, trazendo resultados de classificação satisfatórios. Em que em termos de correlação linear chegou-se a 96% na condição normal, 98% em desbalanceamento,92%, em duas fases e 98% em desníveis. Mostrando que a capacidade de generalização do classificador foi garantida, sendo assertivo a identificação dos padrões de cada condição ensaiada e simulada.

Em suma, destaca-se que o desenvolvimento experimental de técnicas de inteligência artificial, podem ser combinadas com as tradicionais técnicas de manutenção, podendo ser a saída mais viável para uma manutenção industrial eficiente e eficaz no contexto do desenvolvimento indústria 4.0.

6.2 Trabalhos Futuros

A partir desse ponto de avaliação e confirmação que a ferramenta aplicada pode traz acurácia significativa em classificar as condições do motor trifásico de indução, torna-se viável estruturar uma proposta de aplicação de avaliação da condição do motor em tempo real, trazendo gatilhos que possam identificar, diagnosticar e disparar um alarme ou desarme do motor, desta forma evitando situações de colapsos que poderiam causar mais danos, tanto inerentes a disponibilidade da máquina quanto à produtividade de um sistema que dependa do motor trifásico de indução.

Parte dessa implementação tem fundamento com os seguintes tópicos:

- Aprimorar o sistema embarcado usando um sensor SIM800l, permitindo conexão móvel.
- Comunicação do sistema embarcado via protocolo HTTP REQUEST.
- Realizar análise através de MVS em tempo real na nuvem.
- Emissão de aviso da condição avaliada.

7 REFERÊNCIAS

[1] Elmore, W .A . (2004). **PROTECTIVE RELAYING THEORY AND APPLICATIONS**. New York, NY: Marcel Dekker, Inc.

[2] IEEE Operation Center (1997). **Advancement in Microprocessor Based Protection and Communication**, IEEE Tutorial Course, Piscataway, NJ.

[3] Kezunovic, M. (1997). **A SURVEY OF NEURAL NETWORK APPLICATION TO PROTECTIVE RELAYING AND FAULT ANALYSIS**. Engineering Intelligent Systems, vol. 5, No. 4,.

[4] Kolla, S., & Altman, S. **Artificial neural network based fault identification scheme implementation for a three-phase induction motor**. (2007).

[5] Widodo, A., & B-S.Yang. (2007). Support vector machine in machine condition and fault diagnosis. Mechanical System and Signal Processing,.

[6] Avci, E., **Selecting of the optimal feature subset and kernel parameters in digital modulation classification by using hybrid genetic algorithm–support vector machines: HGASVM, Expert Systems with Applications**, March 2009.

[7] CLARK, Paul E.; McSHANE, Ian E. e WAKELEY, Keith. **The Diagnosis and Solution of Electrical Machine Problems – Part I: Design Influences**. IEEE Transactions on Industry Applications. 1987.

[8] BONNETT, Austin H.; SOUKUP, George C. **Cause and Analysis of Stator and Rotor Failures in Three-Phase Squirrel-Cage Induction Motors**. IEEE Transactions on Industry Applications, 1992.

[9] Rosemary Antonia Lopes Faraco. **Detecção de Faltas Elétricas em Motores de Indução Utilizando Redes Neurais**. Pontifícia Universidade Católica de Minas Gerais **2000**

[10] Lorena Teixeira Marques. **Análise Computacional Do Motor De Indução Trifásico: Regime Transitório E Permanente**. UNIVERSIDADE DE SÃO PAULO,São Carlos 2009

[11] F. Imbault and K. Lebart. **A stochastic optimization approach for parameter tuning of support vector machines. In Proceedings of the 17th International Conference on Pattern Recognition**.

[12]Sudha, M., & Anbalagan, P. (2009). **A Protection Scheme for Three-Phase Induction Motor from Incipient Faults Using Embedded Controller**. Asian Journal of Scientific Research.

[13] Hélio Henrique Cunha Pinheiro - **Sistema para Detecção e Diagnóstico de Falhas em Motores Elétricos de InduçãoUtilizando Lógica Fuzzy**. Natal (RN), UFRN2011

[14] ISERMANN, **Rolf,Fault-Diagnosis Systems: An Introduction from Fault Detection to Fault Tolerance**. Berlin, 2006.

[15] Rajkumar Buyya e Kotagiri Ramamohanarao ,**Big Data Analytics = Machine Learning + Cloud Computing** , University of Melbourne January 2016

[16] Daniel Larose and Chantal Larose, **Discovering Knowledge in Data: An Introduction to Data Mining**, John Wiley & Sons, 2014.

[17] Jiawei Han, Micheline Kamber and Jian Pei, **Data Mining Concepts and Techniques 3rd Edition**, Elsevier Inc., 2012

[18] Arthur Samuel, **Some Studies in Machine Learning Using the Game of Checkers**, IBM Journal of Research and Development July 1959

[19] Tom M. Mitchell, **Machine Learning**, McGraw-Hill Science, 1997

[20] Kevin P. Murphy, **Machine Learning, A Probabilistic Perspective**, Kevin P. Murphy, The MIT Press, 2012.

[21] Christopher M. Bishop, **Pattern Recognition and Machine Learning**, Springer, 2006.

[22] Noam Nisan and Shimon Schocken, **The Element of Computing Systems Building a Modern Computer from First Principles**, MIT press 2005

[23] **IBM KNOWLEDGE CENTER** https://www.ibm.com/support/knowledgecenter/en

[24] REDBOOK IBM, **Cloud Object Storage as a Service IBM Cloud Object Storage from Theory to Practice**. International Technical Support Organization, Copyright International Business Machines Corporation 2017.

[25] Han, J. and Kamber, M.(2006). **Data Mining: Concepts and Techniques**. Morgan Kaufmann Publishers

[26] Gartner's Magic Quadrant, https://www.bmc.com/blogs/gartner-magicquadrant- cloud-iaas/

[27] IBM Systems and Technology, **Watson – Um Sistema Projetado para Respostas O futuro do design de sistemas otimizados para carga de trabalho**. 2011

[28] Peter Harrington, **Machine learning in Action**, Manning Publications, 2012.

[29] IBM SPSS Modeler 18.0 **Applications Guide**. IBM

[30] IBM SPSS Modeler 17 **Algorithms Guide**

[31] DIEGO BONESSO **Estimação dos Parâmetros do Kernel em um Classificador SVM na Classificação de Imagens Hiperespectrais em uma Abordagem Multiclasse**. UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

[32] SAMADZADEGAN, F.; HASANI, H.; SHENK, T. **Simultaneous feature selection and SVM parameter determination in classification of hyperspectral imagery using Ant Colony Optimization**., 2012.

[33] **Uma Introdução às Support Vector Machines**. Ana Carolina Lorena 1 André C. P. L. F. de Carvalho. 2007

[34] A. J. Smola and B. Schölkopf. **Learning with Kernels**. The MIT Press, Cambridge, MA, 2002.

[35] [28] K. R. Müller, S. Mika, G. Rätsch, K. Tsuda, and B. Schölkopf. **An introduction to kernel-based learning algorithms**. IEEE Transactions on Neural Networks, Março 2001.

[36] C. J. C. Burges. **A tutorial on support vector machines for pattern recognition**. Knowledge Discovery and Data Mining, 1998.

[37] A. Passerini. **Kernel Methods, multiclass classification and applications to computational molecular biology**. PhD thesis, Università Degli Studi di Firenze, 2004.

[38] V. Vapnik, **Statistical Learning Theory**, John Wiley and Sons, (1998).

[39] M. A. Hearst, B. Schölkopf, S. Dumais, E. Osuna, and J. Platt. **Trends and controversies - support vector machines**. IEEE Intelligent Systems, 1998.

[40] *IBM SPSS Modeler 18.1 Modeling Nodes*

[41] ESP8266EX Datasheet , Espressif

[42] Vinícius Augusto Diniz Silva. **Detecção de falhas em motores elétricos através das máquinas de vetores de suporte**. UNIVERSIDADE ESTADUAL DE CAMPINAS, Campinas, 2012.

[43] **Espressif.ESP8266EX Datasheet**. 2017. Disponível em: http://espressif.com/sites/default/files/documentation/0a-esp8266ex_datasheet_en.pdf>.

[44] Piyu Dhaker, **Introduction to SPI Interface**. Analog Decive

[45] YADI, N.; AYAZI, F.; NAJAFI, K. **Micromachined inertial sensors.Proceedings of theIEEE**, August 1998.

[46] HARRIS, C. M.; CREDE,E.**Shock and Vibration Handbook**. New York: McGraw-Hill,1976.

[47] André da Costa Teves. **Otimização de Acelerômetros MEMS Eletroestáticos de Alto Desempenho.** Escola Politécnicada Universidade de São Paulo 2013

[48] Roberto Carlos Díaz González. **Desenvolvimento de umprotótipo analisador de vibração de baixo custo parauso em manutenção preditiva**. Universidade Federal de Santa Catarina. Florianópolis2014

[49] MARTINS, R. **Um método para detectar falhas incipientes em máquinas rotativas baseado em análise de vibrações e lógica Fuzzy**. Tese apresentada ao Programa de Pós-graduação em Engenharia Metalúrgica de Minas e dos Materiais, UFRGS, Porto Alegre. 2000.

[50] **DATASHEET ANALOG DEVICE ADXL 345**, disponível em < https://www.analog.com/media/en/technical-documentation/data-sheets/ADXL345.pdf >

[51] TEZ, S.; AKIN, T. **Fabrication of a sandwich type three axis capacitive MEMS accelerometer.Sensors**, 2013 IEEE

[52] JIANGFENG WU. **A low-noise low-offset capacitive sensing amplifier for a 50-µg/√Hz monolithic CMOS MEMS accelerometer.Solid-State Circuits**, IEEE Journal, May 2004.

[53] https://www.indiamart.com/proddetail/induction-motor-4448114297.html

[54] O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee. **Choosing multiple parameters for support vector machines. Machine Learning** 2002.

[55] ALMEIDA, Marcio Tadeu. **Manutenção preditiva: benefícios e lucratividade**, 2008

[56] SANTOS, Marcos dos; Marcio Araujo Medeiros; Angélica Rodrigues. **Manutenção Preditiva: contribuindo para a melhoria dos processos e para a redução dos custos de operação**. Instituto Militar de Engenharia – IME

[57] BLOCH, Heinz P.; GEITNER, Fred K., **Machinery Failure Analisys and Thoubleshooting**. 3ª Ed. Volume 2, Gulf Professional Publishing, Houston, 1999.

[58]NANDI, Subhasis; TOLIYAT, Hamid A.; LI, Xiaodong, **Condition Monitoring and Fault Diagnosis of Electrical Motors – A Review**. IEEE Transactions on Energy Conversion, 2005.

8 APÊNDICES

Apêndice 1

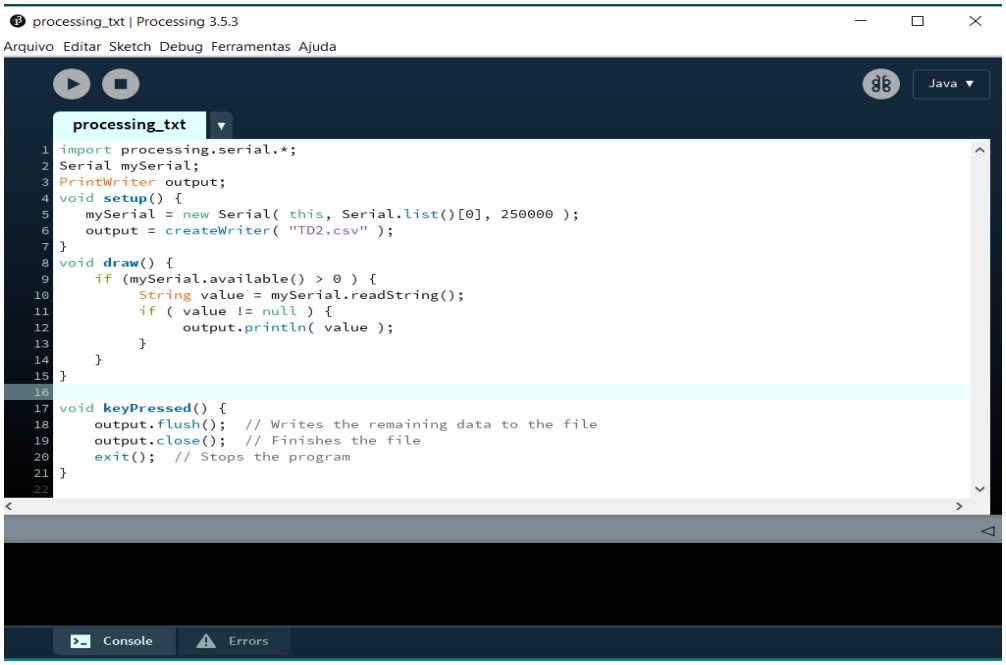
PROCESSING

Processing é uma biblioteca gráfica de código aberto e um ambiente de desenvolvimento integrado (IDE) criado para comunidades de artes eletrônicas, novas mídias e design visual, com o objetivo de ensinar a não programadores os fundamentos da programação de computadores em um contexto visual.

O processamento usa a linguagem Java , com simplificações adicionais, como classes adicionais e funções e operações matemáticas aliadas. Além disso, ele também possui uma interface gráfica do usuário para simplificar o estágio de compilação e execução.

A linguagem Processing e o IDE foram os precursores de outros projetos, incluindo Arduino , Wiring e p5.js

Além disso, a ideia da comunicação entre Processing e Arduino é basicamente assim: o Processing pode ser utilizado de duas formas, fazendo o Arduino executar determinada tarefa ou atuando como uma representação gráfica e visual daquilo que está acontecendo no seu protótipo em protoboard.



Rotina implementada no processing

Apêndice 2

Rotina implementada no arduíno IDE através da biblioteca SPI:

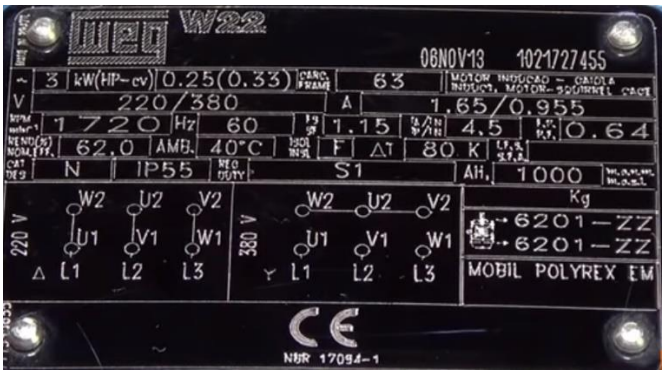
```
#include <SPI.h>

//ADXL345
//SXT 013 saida do fio RED + serie resistor -----> A0 e saida do white + capacitor 100 uf -----> ground

#define BW_RATE 0x2C //Data rate and power mode control
#define POWER_CTL 0x2D //Power Control Register
#define DATA_FORMAT 0x31 //Data format control
#define DATAX0 0x32 //X-Axis Data 0
int analogInPin = A0;  // ESP8266 Analog Pin ADC0 = A0

//Nodemcu pin D5 GPIO 14 -----> pin SCL ADXL345
//Nodemcu pin D6 GPIO 12 -----> pin SDO ADXL345
//Nodemcu pin D7 GPIO 13 -----> pin SDA ADXL345
//Nodemcu pin D0 GPIO 16 -----> pin CS/SS ADXL345
//Nodemcu pin 3.3 -----> pin VCC ADXL345
//Nodemcu pin GND -----> pin GND ADXL345

#define SS 16

char values[10];
int16_t x, y, z;
int sensorValue;  // value read from the pot

void setup() {
  SPI.begin();
  SPI.setDataMode(SPI_MODE3);
  SPI.setBitOrder(LSBFIRST);
  SPI.setFrequency(5000000);
  //SPI.setClockDivider(SPI_CLOCK_DIV16);

  Serial.begin(250000);

  // SS Hight
  pinMode(SS, OUTPUT);
  digitalWrite(SS, HIGH);

  // ADXL345
  writeRegister(DATA_FORMAT, 0x03); // ±16g 10bit
  writeRegister(POWER_CTL, 0x08); //
  writeRegister(BW_RATE, 0x0F); //}
```

```
void loop() {

  // DATAX0
  readRegister(DATAX0, 6, values);

  // 2Byte
  x = ((int16_t)values[1] << 8) | (int16_t)values[0];
  y = ((int16_t)values[3] << 8) | (int16_t)values[2];
  z = ((int16_t)values[5] << 8) | (int16_t)values[4];
  //
  Serial.print(x);
  Serial.print(",");
  Serial.print(y);
  Serial.print(",");
  Serial.print(z);
  // read the analog in value
  sensorValue = analogRead(analogInPin);
  Serial.print(",");
  Serial.println(sensorValue);}

void writeRegister(char registerAddress, char value) {
  // SPI
  digitalWrite(SS, LOW);
  //
  SPI.transfer(registerAddress);
  //
  SPI.transfer(value);
  // SPI
  digitalWrite(SS, HIGH);}

void readRegister(char registerAddress, int16_t numBytes, char * values) {
  //
  char address = 0x80 | registerAddress;
  //
  if (numBytes > 1)address = address | 0x40;
  // SPI
  digitalWrite(SS, LOW);
  //
  SPI.transfer(address);
  //
  for (int16_t i = 0; i < numBytes; i++) {
    values[i] = SPI.transfer(0x00);
  }
  // SPI CS HIGH
  digitalWrite(SS, HIGH);
}
```

Apêndice 3

Motor Trifásico de Indução

O modelo da WEG W22 IR3, foi o motor utilizado em laboratório, com partida direta, para fornecer diferentes condições do motor de indução, as descrições seguem na placa do motor e na descrição do fabricante:



weg		No.:
		Data: 02-OUT-2019

FOLHA DE DADOS
Motor trifásico de indução - Rotor de gaiola

Cliente	:	
Linha do produto	:	W22 IR3 Premium

Carcaça	:	71
Potência	:	0,25 HP
Freqüência	:	60 Hz
Polos	:	4
Rotação nominal	:	1720 rpm
Escorregamento	:	4,44 %
Tensão nominal	:	220/380 V
Corrente nominal	:	1,01/0,585 A
Corrente de partida	:	4,75/2,75 A
Ip/In	:	4,7
Corrente a vazio	:	0,650/0,376 A
Conjugado nominal	:	1,000 Nm
Conjugado de partida	:	230 %
Conjugado máximo	:	270 %
Categoria	:	---
Classe de isolação	:	F
Elevação de temperatura	:	80 K
Tempo de rotor bloqueado	:	55 s (quente)
Fator de serviço	:	1,25
Regime de serviço	:	S1
Temperatura ambiente	:	-20°C - +40°C
Altitude	:	1000 m
Proteção	:	IP55
Massa aproximada	:	7 kg
Momento de inércia	:	0,00049 kgm²
Nível de ruído	:	47 dB(A)

	Dianteiro	Traseiro
Rolamento	6202 ZZ	6202 ZZ
Intervalo de lubrificação	---	---
Quantidade de graxa	---	---

Carga	Fator potência	Rendimento (%)
100%	0,67	69,5
75%	0,58	67,0
50%	0,47	61,0