UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
CÂMPUS DE DOIS VIZINHOS
CURSO DE ESPECIALIZAÇÃO EM CIÊNCIA DE DADOS

ESTEVAN AQUILES PAZZETTI

# USO DE APRENDIZADO DE MÁQUINA E MÉTODOS DE ANÁLISE DE DADOS PARA PREDIÇÃO DE DESEMPENHO EM BANCOS DE DADOS

DOIS VIZINHOS
2022

ESTEVAN AQUILES PAZZETTI

# USO DE APRENDIZADO DE MÁQUINA E MÉTODOS DE ANÁLISE DE DADOS PARA PREDIÇÃO DE DESEMPENHO EM BANCOS DE DADOS

Trabalho apresentado ao Curso de Especialização em Ciência de Dados da Universidade Tecnológica Federal do Paraná, como requisito para a obtenção do título de Especialista em Ciência de Dados.

Orientador: Prof. Dr. Rafael Gomes Mantovani
Coorientador: Prof. Dr. Francisco Carlos Monteiro Souza

DOIS VIZINHOS
2022

ESTEVAN AQUILES PAZZETTI

# USO DE APRENDIZADO DE MÁQUINA E MÉTODOS DE ANÁLISE DE DADOS PARA PREDIÇÃO DE DESEMPENHO EM BANCOS DE DADOS

Trabalho apresentado ao Curso de Especialização em Ciência de Dados da Universidade Tecnológica Federal do Paraná, como requisito para a obtenção do título de Especialista em Ciência de Dados.

Data de Aprovação: 05/janeiro/2022

Rafael Gomes Mantovani
Doutorado
Universidade Tecnológica Federal do Paraná - Câmpus Apucarana

Ives Renê Venturini Pola
Doutorado
Universidade Tecnológica Federal do Paraná - Câmpus Pato Branco

Rafael Alves Paes de Oliveira
Doutorado
Universidade Tecnológica Federal do Paraná - Câmpus Dois Vizinhos

DOIS VIZINHOS
2022

Dedico este trabalho a minha digníssima esposa Aline Malagi, pelo apoio e parceria de vida. Sempre dedicada para que alcancemos o nosso melhor.

## AGRADECIMENTOS

Primeiramente gostaria de agradecer aos meus orientadores Rafael Gomes Mantovani e Francisco Carlos Monteiro Souza, pelo apoio e dedicação fundamental na construção desse trabalho.

Aos meus familiares e amigos, pelo carinho, incentivo e compreensão em todas as jornadas que tracei até aqui.

Aos professores pela imensa contribuição e dedicação na transferência do conhecimento.

Agradecer a Universidade Tecnológica Federal do Paraná pela oportunidade de participar deste curso, sempre proporcionado experiências de alto nível no meu desenvolvimento pessoal e profissional.

# RESUMO

PAZZETTI, Estevan Aquiles. Uso de Aprendizado de Máquina e Métodos de Análise de Dados para Predição de Desempenho em Bancos de Dados. 2022. 35 f. – Curso de Especialização em Ciência de Dados, Universidade Tecnológica Federal do Paraná. Dois Vizinhos, 2022.

O avanço tecnológico das últimas décadas tem gerado um crescimento exponencial do volume de dados na sociedade contemporânea. Provenientes de diferentes fontes, os dados impulsionam a utilização de diferentes sistemas de armazenamento. Um problema conhecido e bastante relatado pela área de Tecnologia da Informação, é a constante queda de desempenho de bancos de dados. Este é considerado um grande problema que poderia ser corrigido e tornar os processos de processamento de dados mais eficientes. Neste sentido, ter mecanismos que detectam possíveis quedas de desempenho de um banco de dados, constituir-se-ia como uma forma de diminuir o tempo de análise e identificação destas quedas de desempenho do banco de dados pelo analista. Este trabalho tem como objetivo abordar o uso de algoritmos de aprendizagem de máquina para predição de problemas de performance em servidores de banco de dados relacional. Foram coletados e tabulados referentes ao desempenho de bancos de dados utilizando o sistema gerenciador IBM DB2 em sistema operacional Linux, e um estudo de caso conduzido, avaliando-se diferentes algoritmos de aprendizado de máquina. Os resultados foram promissores e indicaram ser possível identificar quedas de desempenho na operação de um servidor de banco de dados.

**Palavras-chave**: Monitoramento de banco de dados, Classificação, Aprendizado de Máquina.

## ABSTRACT

PAZZETTI, Estevan Aquiles. Use of Machine Learning and Data Analysis Methods to Predict Database Performance. 2022. 35 f. – Curso de Especialização em Ciência de Dados, Universidade Tecnológica Federal do Paraná. Dois Vizinhos, 2022.

Technological advances in recent decades have generated an exponential growth in the volume of data in contemporary society. Coming from different sources, data drives the use of different storage systems. A well-known problem, which is frequently reported by the Information Technology area, is the constant decline in database performance. This is considered a big problem that could be fixed and make data processing processes more efficient. In this sense, having mechanisms that detect possible drops in the performance of a database would be a way of reducing the analysis and identification time of these drops in the database's performance by the analyst. This work aims to approach the use of machine learning algorithms to predict performance problems in relational database servers, using the database manager IBM DB2 on operational system Linux. Database performance was collected and tabulated, and a case study was conducted, evaluating different machine learning algorithms. The results were promising and indicated that it is possible to identify performance drops in the operation of a database server.



**Keywords**: Database Monitoring, Classification, Machine Learning.

## LISTA DE FIGURAS

## LISTA DE TABELAS

## LISTA DE ABREVIATURAS E SIGLAS

| | |
|---|---|
| AM | Aprendizado de máquina |
| ML | Machine Learning |
| IA | Inteligência Artificial |
| SGBD | Sistema Gerenciador de banco de dados |
| BD | Banco de Dados |
| TCC | Trabalho de Conclusão de Curso |
| KNN | k-Nearest Neighbors Algorithm |
| SVM | Support Vector Machines |
| MLP | Multilayer Perceptron |
| CPU | Central Process Unit |
| GB | Gigabytes |
| MDI | Mean Decrease Impurity |

SUMÁRIO

# 1 INTRODUÇÃO

Os humanos são seres que conseguem aprender sobre problemas e tomar decisões com base em suas experiências prévias e tarefas já aprendidas. No entanto, ao longo da história da humanidade e com o crescente volume de dados gerado pelo mundo tecnológico em que vivemos, há uma crescente necessidade de processamentos e armazenamentos cada vez mais rápidos e ágeis. Desta forma, a realização destes processos tornou-se cada vez mais difícil de ser realizado por um ser humano. É neste contexto que tornou-se necessário o uso de ferramentas computacionais que melhorassem o processamento de dados, tornando assim possível a resolução de problemas práticos de forma mais rápida (BERTOZZO, 2019).

O avanço tecnológico das últimas décadas tem gerado um crescimento exponencial do volume de dados na sociedade contemporânea. Tal comportamento tem sido identificado em diferentes setores como: manufatura, serviços financeiros, educação, comércio, entre outros. Provenientes de diferentes fontes, os dados impulsionam a utilização de diferentes sistemas de armazenamento.

Na área da computação os problemas reais que demandam de processamentos de dados normalmente são resolvidos através da criação de softwares, que basicamente consistem na escrita de um código-fonte que implementa um algoritmo responsável por automatizar uma tarefa. Técnicas e algoritmos de Inteligência Artificial (IA), especificamente uma de suas áreas, o Aprendizado de Máquina (AM) (*Machine Learning*, do inglês), têm demonstrado alto grau de sucesso na solução de problemas que envolvem tarefas de reconhecimento de padrões. Como o nome diz, essas soluções são capazes de aprender a partir de dados obtidos e tomar decisões (BISHOP, 2007).

O AM tem por objetivo desenvolver técnicas e algoritmos capazes de de adquirir informação e gerar conhecimento automaticamente sobre esses dados. Um sistema baseado em AM tem capacidade de tomada de decisões com base no que aprendeu em experiências anteriores. De acordo com Mitchell (1997), as conclusões e conhecimentos do AM são geradas a partir da indução, considerada uma forma de inferência lógica que retira as conclusões genéricas, ou seja, algo diferente do que aprendeu previamente de um conjunto de dados. Quando um algoritmo de AM está "aprendendo" a partir de um conjunto de dados, ele está a procura de hipóteses, verdadeiras ou não, capazes de descrever as relações existentes entre os objetos e que melhor se ajustem a estes dados.

Um problema conhecido e bastante relatado pela área de Tecnologia da Informação, é a constante queda de desempenho de bancos de dados. Este é considerado um grande problema que poderia ser corrigido e tornar os processos de processamento de dados mais eficientes. Neste sentido, ter mecanismos que detectam possíveis quedas de desempenho de um banco de dados, constituir-se-ia como uma forma de diminuir o tempo de análise e identificação destas quedas de desempenho do banco de dados pelo analista. No entanto, se faz necessário

destacar que não existe apenas um algoritmo perfeito para todos os problemas existentes, e isso se aplica também na manipulação e gerenciamento de bases de um banco de dados. Cada algoritmo tem suas especificidades e, consequentemente, a forma que realiza a captação de conhecimento e de hierarquia na indução do aprendizado. Logo, é importante investigar as possibilidades para contornar esse problema.

## 1.1 Problema de Pesquisa

Explorar técnicas e algoritmos de AM para monitoramento de desempenho de bancos de dados, através da coleta, armazenamento e análise das características e métricas fornecidas pelo SGBD e servidor de banco de dados. Assim, deseja-se antecipar quedas de desempenho, com a expectativa de aumentar a satisfação de clientes e ganho de tempo de análise dos analistas.

Assim, será conduzido um estudo experimental do comportamento do banco de dados para detectar padrões de uso de processamento e taxas de transferência de memória. Serão aplicados algoritmos de AM supervisionados e soluções baseada em uma sequência de processos convencional (*pipeline*) na elaboração de uma solução prática para o problema.

## 1.2 Objetivos

Diante deste contexto, o objetivo do presente estudo é identificar os motivos que levam à quedas de desempenho em um servidor de banco de dados e evidenciar quais as características destas quedas. Esse estudo das características tem como finalidade entender melhor o problema e descobrir padrões que possam melhorar a agilidade no funcionamento dos bancos de dados. Os principais objetivos do trabalho são apresentados a seguir:

### 1.2.1 Objetivo Geral

Explorar técnicas e algoritmos de AM para predição de quedas de desempenho em um servidor de banco de dados.

### 1.2.2 Objetivos Específicos

- extrair e organizar características relativas à quedas de desempenho de um servidor de banco de dados;
- avaliar e explorar diferentes algoritmos de AM para predizer situações de queda de desempenho;
- analisar quais as principais características que indicam possíveis quedas.

## 1.3 Justificativa

Normalmente o gerenciamento de quedas de desempenho de um banco de dados envolve a coordenação de diferentes atividades e sistemas conectados. Automatizar este processo de gerenciamento pode otimizar o uso de recursos disponíveis e evitar possíveis gargalos nos recursos computacionais. Neste sentindo o AM pode ajudar na detecção de janelas de baixo desempenho dos processos do banco de dados.

Um algoritmo de análise de uso do processamento e de monitoramento da memória do banco de dados pode obter informações sobre o uso dos recursos computacionais do servidor em que o banco de dados encontra-se alocado. Desta forma, um algoritmo de AM pode ser treinado para identificar possíveis padrões, e realizar as separação dos recursos computacionais utilizados, como por exemplo, baixo, médio e alto uso de transferência memória e processamento.

## 1.4 Organização do Trabalho

Este trabalho está estruturado da seguinte forma: O Capítulo 2 apresenta a fundamentação teórica. Então, os métodos de coleta de dados e todas as etapas da elaboração da solução do problema de pesquisa são detalhados no Capítulo 3. Os resultados experimentais são discutidos no Capítulo 4. No Capítulo 5 são apresentadas as conclusões e as futuras direções de pesquisa.

# 2 REVISÃO DE LITERATURA

Neste capítulo são apresentados os aspectos conceituais para a construção desta pesquisa, com base em uma revisão bibliográfica na produção acadêmica sobre o tema. Aprendizado de máquina pode ser definida como uma tecnologia de computação, que une técnicas, um campo de estudo que permite aos computadores aprender comportamentos, padrões e tomar decisões sem a intervenção humana, utilizando principalmente algoritmos previamente escritos para essa tarefa. A indução é o recurso mais utilizado pelo cérebro humano (MONARD, 2003).

Por que usar Aprendizado de Máquina? Seu uso pode ser útil para automatizar tarefas que são realizados por humanos, tais como realizar recomendações, reconhecer de padrões ou até mesmo serviços de autoatendimento. O método tem a capacidade de executar tarefas de análise de dados, apresentando os resultados mais relevantes conforme os padrões e informações aprendidas pelo algoritmo. Partimos do entendimento de que o aprendizado de máquina utiliza técnicas de análise e fórmulas matemáticas para encontrar padrões de comportamento nos dados, gerando conhecimento a partir das informações coletadas e automatizando as tarefas que são executadas por humanos (ESCOVEDO; KOSHIYAMA, 2020).

Existem diversos métodos e algoritmos no uso de aprendizado de máquina, podendo existir várias maneiras de resolver problemas com as técnicas e algoritmos. Algoritmos supervisionados, utilizam algoritmos preditivos a partir de conjuntos já rotulados e conhecidos, os dados são divididos para sejam realizados os processos de treinamento e teste, apontando possíveis hipóteses nas classes do conjunto de dados. No aprendizado com algoritmos não supervisionado o aprendizado das informações apresenta apenas uma hipótese, nesse modelo busca identificar os padrões com base nas informações existentes, sem a necessidade do conhecimento prévio ou a classificação dos dados. O algoritmo aprende os padrões do conjunto apresentando a melhor hipótese encontrada como valor de saída.

## 2.1 Tipos de Aprendizado de Máquina

Atualmente o aprendizado de máquina é classificado em 3 principais paradigmas divididos entre supervisionado, não-supervisionado e aprendizagem por reforço, cada um dos métodos aborda técnicas especificas para realizar o treinamento dos dados e a forma que o problema pode ser aprendido.

### 2.1.1 Aprendizado Supervisionado

No aprendizado supervisionado os algoritmos recebem os atributos definidos, onde os valores das amostras como classe ou categoria são conhecidos, este tipo de abordagem trabalha com a classificação da informação com base nas amostras disponíveis e previamente rotuladas, para cada saída é atribuído um rótulo, que pode ser um valor numérico ou uma

classe. O algoritmo determina uma forma de prever qual o rótulo de saída com base em uma entrada informada (MITCHELL, 1997). Pode-se utilizar como exemplo uma base de dados de imagens de frutas, o algoritmo receberá as características de cada fruta disponível na base de dados e a partir disso, consegue realizar a classificação da fruta baseada nas características individuais de cada amostra.

### 2.1.2 Aprendizado Não Supervisionado

Neste paradigma o algoritmo de aprendizado irá treinar o modelo sem a rotulação ou conhecimento prévio dos dados, o principal objetivo nesse formato é fazer com que o algoritmo descubra os padrões dos valores analisados e apresente possíveis valores de solução que respondam o problema analisado. No caso do aprendizado não supervisionado o objetivo do algoritmo irá usar as características para descobrir as possíveis classes e criar uma separação das amostras conforme o padrão de características encontradas, criando assim as classes dentro do conjunto de dados.

### 2.1.3 Aprendizado por Reforço

Este formato de aprendizado tem como objetivo é de que o modelo seja treinado com base em recompensa ou penalização dependendo do resultado da ação executada, o processo do algoritmo tem como objetivo tentar encontrar uma solução para determinado problema com base na recompensa positiva (ACADEMY, 2021). Os algoritmos de aprendizagem por reforço exploram os os resultados buscando o maior que resolva o problema ou para que avance na solução, tendo a capacidade de explorar os valores já que o algoritmo não recebe informações sobre a solução proposta (BISHOP, 2007).

## 2.2 Algoritmos de Aprendizagem Supervisionada

O uso de algoritmos de aprendizado de máquina adequado é o ponto chave para o sucesso na resolução de determinado problema, a escolha do algoritmo adequado para cada situação traz diferenças relevantes na análise dos dados (MUELLER; MASSARON, 2019).

### 2.2.1 Naïve Bayes

O algoritmo Naïve Bayes utiliza métodos de aprendizado supervisionado com base na aplicação do teorema de Bayes, trabalhando com classificação de probabilidades a partir das características dado o valor da classe definida para a aprendizagem (LEARN, 2021e).

### 2.2.2 Árvore de Decisão

Árvores de Decisão é um método de aprendizado supervisionado, sendo utilizado na área de aprendizado de máquina para tarefas de classificação e regressão. O modelo opera com

base em decisões, basicamente tomando uma decisão a partir do valor da variável (verdadeiro ou falso, por exemplo), criando novos nós folhas havendo novas possibilidades de obter outros conjuntos de dados para a solução do problema (LEARN, 2021a).

### 2.2.3 `k-Nearest Neighbors`

O `k-Nearest Neighbors (KNN)` é algoritmo de aprendizado supervisionadao e não supervisionado, é um método de aprendizado baseado nas características dos vizinhos mais próximos, tem como objetivo identificar os valores que mais se aproximam das características de cada amostra, criando a classificação de objetos ou valores. O algoritmo calcula a distância entre os dados conforme medida de distância definida nos parâmetros para criar a classificação dos objetos (LEARN, 2021b)

### 2.2.4 Regressão Logística

O algoritmo de classificação de regressão logística trabalha com o conceito de probabilidade, tendo como objetivo traçar uma reta que possa separar as amostras, medir a curva de separação as classes definidas para o modelo e responder a qual classe cada valor pertence (LEARN, 2021c).

### 2.2.5 `Random Forest`

O algoritmo `Random Forest` pode ser utilizado para classificação e regressão, o objetivo desse algoritmo visa criar um conjunto de árvores de decisão, dividindo as amostras de dados conjuntos menores, esses conjuntos são gerados aleatoriamente. O resultado da classificação ou da regressão será gerado a partir da média individuais dos classificadores (LEARN, 2021f).

### 2.2.6 `Support Vector Machines`

`Support Vector Machines (SVM)` é um algoritmo de aprendizado supervisionado, seu método que pode ser utilizado para classificação, regressão e detecção de `outliers`. A operação desse algoritmo consiste na separação dos dados gerando um hiperplano, dessa forma o algoritmo cria a classificação das amostras separando as categorias dentro do melhor planos gerado, modelo busca sempre maximizar a distância entre as classes (LEARN, 2021g).

### 2.2.7 `Multilayer Perceptron`

O `Multilayer Perceptron (MLP)` é um algoritmo de aprendizado supervisionado, é um método de rede neural que utiliza uma camada de entrada dos valores das amostras para gerar uma camada de saída aprendizagem, gera um valor de resultado da classificação das amostras computando as médias das entradas em cada neurônio apresentando o resultado da classificação da amostra na saída do neurônio (LEARN, 2021d). O `MLP` é considerado um

algoritmo de rede neural, onde a composição de camadas de entrada recebem o sinal tomando uma decisão prévia para a saída do sinal, sendo assim, as multicamadas aprendem a correlação dos parâmetros entre entrada e saída, gerando o valor de peso na saída como resultado do treinamento.

O processo de aprendizado de máquina contempla várias etapas para que se tenha sucesso na sua aplicação, dentre os principais passos pode-se destacar o entendimento do negócio, entendimento dos dados que serão utilizados, a coleta e preparação dos dados, criação do modelo de aprendizado que será utilizado, a avaliação do modelo e a implantação dos modelos de aprendizado de máquina (HARRISON, 2019).

# 3 METODOLOGIA

Neste capítulo são apresentados os detalhes da metodologia experimental. A Figura 1 apresenta o fluxo de processos realizados, desde a obtenção dos dados até a geração dos resultados. Em linhas gerais, informações são coletadas de um data center. Os dados são então tabulados e um processo de análise exploratória é conduzido. Nesse processo também é feita a limpeza dos dados via pré-processamentos. Com a base pronta, uma classe (atributo preditivo) é criada, e algoritmos de classificação são alimentados com essa informação. O processo de treinamento dos modelos é iterativo, pois nem sempre o primeiro algoritmo avaliado irá gerar bons resultados. Porém, uma vez que resultados promissores são atingidos, os modelos são refinados, avaliados e testados em dados para gerar conhecimento sobre o domínio. Logo, cada etapa desse fluxograma descreve um processo de mineração de dados ou aprendizado de máquina. Nas próximas seções, essas processos serão mais detalhados e explicados.

## 3.1 Obtenção dos dados

A base de dados utilizada neste trabalho teve suas informações coletadas a partir de bases de testes operando em um ambiente de *datacenter*. As informações sensíveis foram ajustadas para descaracterizar qualquer identificação do banco banco de dados original. Os valores coletados são informações referentes a: indicadores de performance; capacidade de processamento do servidor, como quantidade de memória e núcleos de processador; informação de objetos ou pacotes de estatísticas inválidos; informações de datas de execução de manutenção e atualização; informação de versão do SGBD; informações de configuração de performance do banco de dados, como ajuste automático ou não do uso de memória pelo SGBD; configuração de *bufferpools* de memória; e parâmetros relacionados ao processamento de instruções SQL no uso do banco de dados.

Todas estas características presentes na base de dados foram coletadas em ambiente de testes de sistema ERP. As informações da base consistem em métricas de desempenho do sistema gerenciador de banco de dados(SGBD) IBM-DB2 acrescido de informações de performance do sistema operacional, que foram coletadas através da linguagem Python, exportadas para um arquivo no formato CSV.

## 3.2 Dataset

Para o desenvolvimento do trabalho, os dados da etapa anterior de coleta foram organizados em uma base dados (dataset). Esses dados apresentam informações do consumo de servidor e do sistema gerenciador de bando de dados DB2 da IBM. Dentre os principais parâmetros coletados estão:

- consumo de processadores;

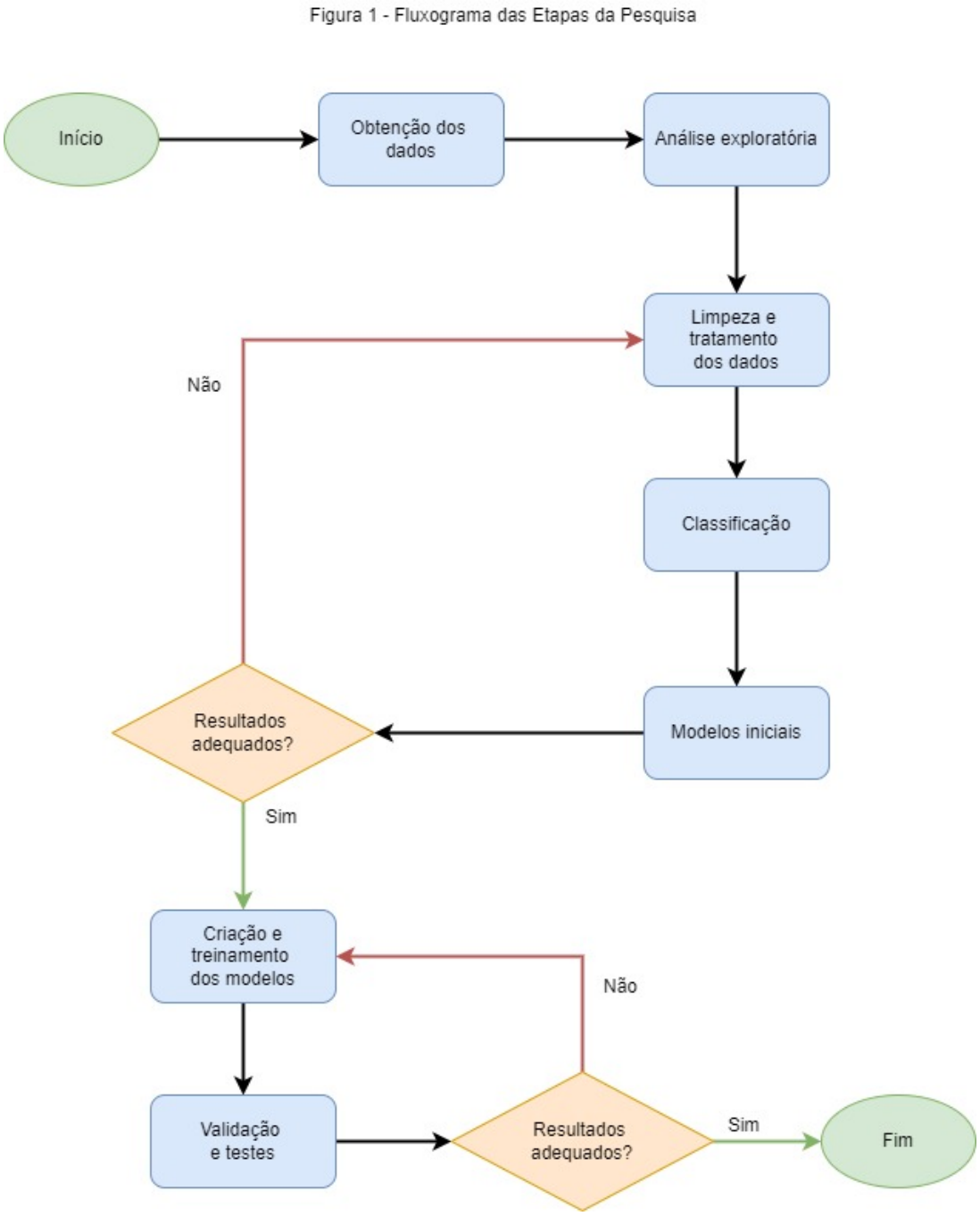



Figura 1 – Metodologia usada para realizar os experimentos descritos no trabalho.

Fonte: Autoria própria.

- uso de memória;
- quantidade de CPU e memória instalados no servidor;
- versão de instalação do SGBD;
- quantidade de objetos e pacotes inválidos no banco;
- etc.

A Tabela 1 descreve todas as informações coletadas e tabuladas para esse trabalho.

Para cada atributo é apresentado seu identificador, uma breve explicação do que representa, e o tipo de dado que ela contém (inteiro, real, categórico, data etc). Dentre as principais características podemos destacar os atributos de uso processador (USO_TOTAL_CPU), consumo de memória (MEMORIA_*) e *load short average* do Linux (CPU_LOAD_SHORT). Essa última em específico, é uma medida do próprio sistema operacional Linux que os mostra o consumo de processamento no intervalo de 1 minuto.

A coleta de dados realizada nos servidores de teste gerou um conjunto de dados contendo $5667$ linhas e $35$ colunas no total. Para realização dos experimentos algumas linhas e colunas foram removidas devido a pouca relevância que tem na análise do comportamento de um banco de dados. Atributos como GRUPO, SUBGRUPO, DATAHORACONSULTA, SPEED_CPU, OS_NAME, DB_NAME, PROD_FIXPACK_NUM, INSTALLED_PROD, OS_FULL_VERSION, foram removidos.

Além disso, o tratamento dos valores faltantes também foi realizado, linhas com uma grande quantidade de valores ausentes foram removidas. As demais passaram por um processo de preenchimento de dados (*data imputation* foi realizado substituindo os valores ausentes de acordo com o tipo de atributo de cada coluna. Após o pré-processamento e tratamento dos dados o conjunto total de dados resultou em 4735 linhas e $19$ colunas.

Considerando as características que permaneceram no conjunto de dados, tem grande relevância na avaliação de uma operação saudável em um banco de dados. Valores altos podem indicar problemas de performance no uso do SGBD, e assim sugerir que há sobrecarga na operação do hardware gerando filas no processamento das informações. As demais características presentes no conjunto de dados são informações de parametrização do SGBD, como configuração de memória, configuração de gerenciamento de memória (se está no modo automático ou não), informação de versão do SGBD, quantidade de objetos e pacotes inválidos (nesse atributo indica problemas caso a quantidade seja maior que zero). Há também o registro de datas de atualização dos objetos e data da última manutenção realizada na base de dados. Em casos de datas muito antigas, isso indica problemas de performance, já que as informações de estatísticas nas tabelas de dados ficam desatualizadas, aumentando o consumo de leitura em disco pela busca de dados no servidor.

## 3.3 Definição da Classe

Para indicar se há ou não queda de desempenho na execução de um sistema de banco de dados é necessário definir um atributo alvo, ou atributo preditivo. Essa coluna do dataset é a que será predita pelos algoritmos de aprendizagem supervisionada. Sendo assim, testes iniciais foram feitos com a escolha do atributo de "USO_TOTAL_CPU" com base no conhecimento geral de um administrador de banco de dados. O consumo alto de processador é uma característica comum em servidores banco de de dados com problemas de performance. A Figura 2 mostra um histograma com os valores de uso de CPU de todas as instâncias do dataset.

Seguindo o conhecimento do especialista da área, instâncias do dataset que apresen-

Tabela 1 – Descrição dos atributos que compõem a base de dados para detecção de queda de desempenho de banco de dados. Para cada atributo é mostrado seu id, descrição e qual tipo de dado representa.

| **Id** | **Descrição** | **Tipo** |
|---|---|---|
| CPU_LOAD_SHORT | Consumo médio cd CPU no último minuto | real |
| LOAD_MEDIO_CPU | Consumo médio cd CPU nos últimos 5 min. | real |
| CPU_LOAD_LONG | Consumo médio de CPU nos últimos 15 min. | real |
| USO_TOTAL_CPU | Consumo total de CPU (%) | inteiro |
| NR_CPU | Quantidade de núcleos de CPU no server | inteiro |
| MEMORIA_TOTAL | Qtde de memória do server | inteiro |
| MEMORIA_FREE | Qtde de memória livre no server | inteiro |
| DB_NAME | Cód de identificação do banco de dados | inteiro |
| QTD_CONEXAO | Qtde de conexões no momento da coleta | inteiro |
| BUILD_BANCO_DB2 | Cód da versão do SGBD | alfanumérico |
| QTD_ERROS_ATUALIZACAO | Qtde de erros/atualização de ERP | inteiro |
| ULTIMO_RUNSTATS | Informação de data da última atualização de estatísticas das tabelas do banco de dados | Data_Hora |
| QTD_OBJETOS_INVALIDOS | Qtde de objetos inválidos no BD | inteiro |
| DT_PACOTE_ANTIGO | Data da última atualização de pacotes SQl | inteiro |
| PACOTES_INVALIDOS | Qtde de pacotes SQL inválidos | inteiro |
| BUILD_INST_DB2 | Versão do SGBD em operação do banco de dados | alfanumérico |
| BUFFERPOLLS_AUTO | Indica configuração de buffers de memória automática | texto |
| MEMORIA_BD | Configuração de memória disponível no BD | inteiro |
| DFT_QUERYOPT | Indica grau de otimização em que o DB2 compila as instruções SQL | inteiro |
| LOGBUFSZ | Utilizado como cache de de fila do BD antes da gravação dos dados | inteiro |
| DB_MEM_THRESH | Parâmetro representa a porcentagem máxima de memória compartilhada do banco de dados (0 (ruim) – 100 (ótimo)) | inteiro |
| SELF_TUNING_MEM | Indica config automática dos buffers de memória no BD | texto |
| DATABASE_MEMORY | config de memória auto no BD | alfanumérico |
| DFT_DEGREE | Parâmetro de banco, determina se haverá paralelismo nos processos, com base no número de CPU e tipo da consulta (1 – não terá paralelismo e -1 significa que o otimizador do banco define o grau de paralelismo nas instruções, ou seja, -1 é o ideal) | inteiro |
| TAMANHO_DO_BANCO_GB | Tamanho da base de dados | real |
| TRIGGERS_AUDLOG | Qtde de triggers de auditoria de ERP | inteiro |
| INSTANCE_MEMORY | Config de qtde de memória de instâncias do SGBD | inteiro |

taram mais de 75% de uso da CPU disponível foram rotulados como passíveis à falhas na performance, enquanto os demais exemplos foram rotulados como normais. A figura 3 mostra a distribuição de classes obtidas, com $3602$ exemplos normais, e $1133$ amostras com com indício de performance.

Os primeiros experimentos realizados, e chamados aqui no trabalho de " Experimentos 1 e 2", levam em conta esse atributo preditivo, para essa análise foi utilizado a condição do consumo de CPU acima de $75\%$. É importante frisar que uma vez que o atributo seja definido como preditivo ele não é considerado como atributo descritivo do problema.

Depois de analisar alguns resultados preliminares, uma segunda abordagem foi elaborada. Além do uso de CPU, mais dois atributos foram considerados para definição do rótulo: uso de memória do sistema, e CPU_LOAD_SHORT. O primeiro tem uma interpretação simples, já que indica a porcentagem de Memória RAM usada pelo sistema. Valores altos indicam uma maior probabilidade de falhas, e valores baixos um desempenho dentro da normalidade.

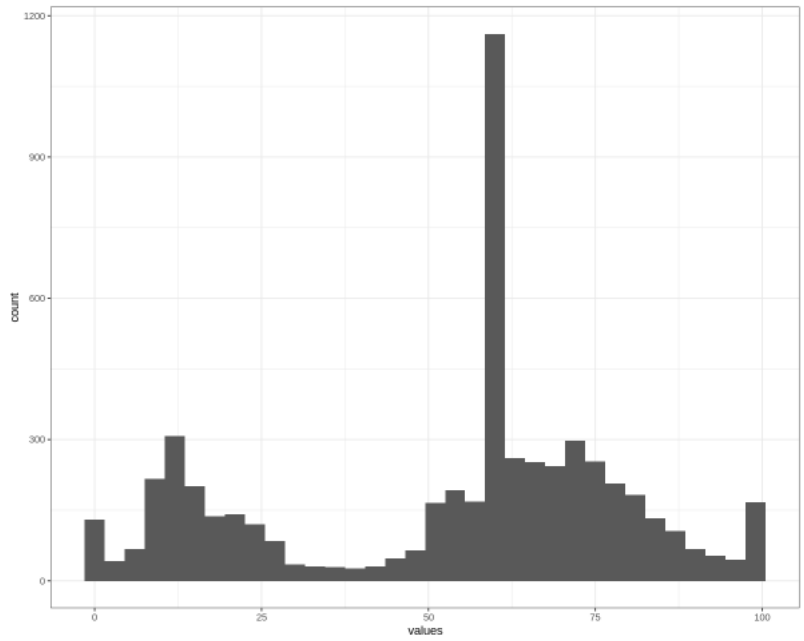



Figura 2 – Distribuição dos valores de uso de CPU (em porcentagem).

Fonte: Autoria própria.

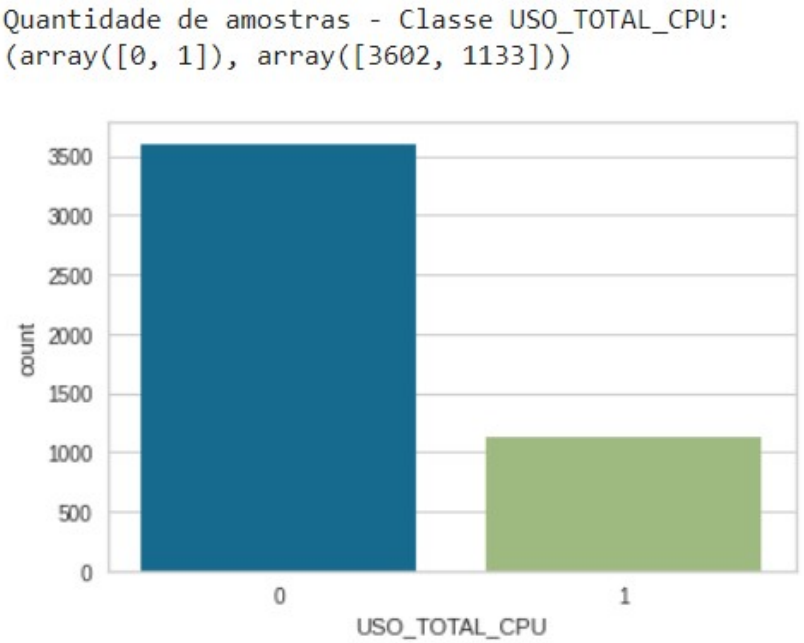



Figura 3 – Distribuição de classes considerando o atributo de uso total de CPU.

Fonte: Autoria própria.

A Figura 4 mostra o histograma de valores de uso de memória para todas as instâncias do dataset. Para gerar esse atributo foi feita uma simples operação levando em conta os valores de memória livre e memória total de cada instância. Valores acima de $95\%$ foram definidos

como passíveis à quedas de desempenho. Já a figura 5 mostra a distribuição dos valores de CPU_LOAD_SHORT. Valores acima de 5 foram definidos como passíveis de problemas de performance.

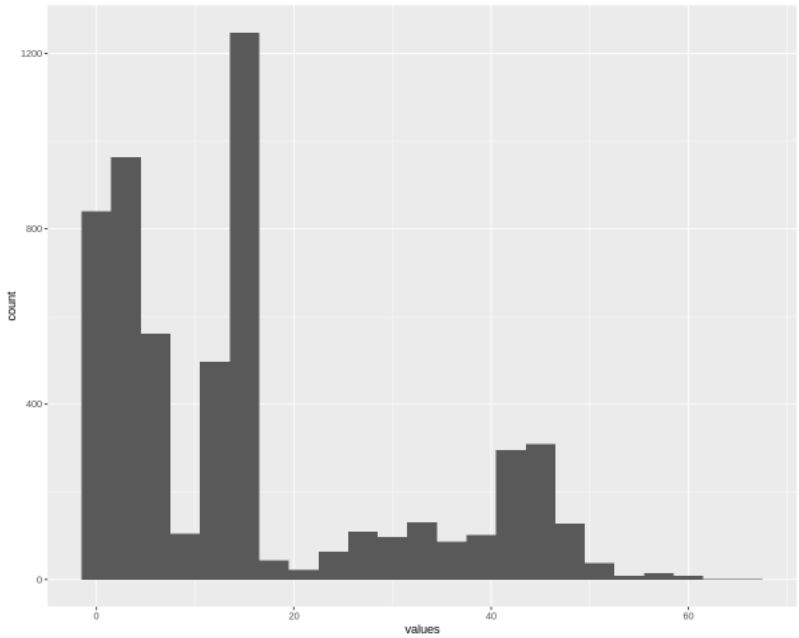



Figura 4 – Distribuição dos valores de uso de memória (em porcentagem).

Fonte: Autoria própria.

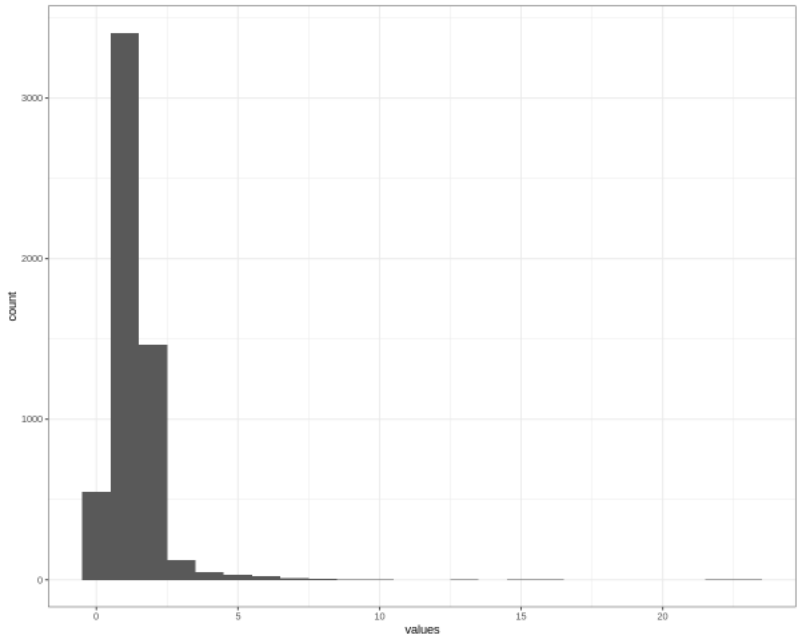



Figura 5 – Distribuição dos valores de CPU_LOAD_SHORT (em porcentagem).

Fonte: Autoria própria.

Os três atributos foram combinados para gerar uma nova classe. Essa nova base é usada no que chamamos de "Experimento 3", as seguintes condições foram definidas para esse experimento,CPU_LOAD_SHORT acima de $5$, consumo de memória acima de $95\%$ ou tamanho da base base de dados menor ou igual a $25$ GB. Se uma instância satisfaz ao menos uma das três condições acima descritas ela vai ser rotulada como passível à queda de desempenho. Assim, com a combinação destes três atributos, é possível verificar um pouco mais de robustez na identificação de possíveis quedas de performance devido ao alto consumo no processamento de hardware no geral. Conforme visto na Figure 6, houve um "balanceamento" das classes, equilibrando as amostras nas duas classes, sendo $2536$ amostras normais e $2199$ amostras indicando queda no desempenho.


Figura 6 – Distribuição de classe considerando o atributo composto criado (PERFORMANCE).

Fonte: Autoria própria.

## 3.4 Pré-processamento

As duas versões de dataset foram pré-processadas para lidar com eventuais ruídos presentes nas informações coletadas. As etapas de pré-processamento realizadas foram:

- remoção de atributos identificadores, que poderiam caracterizar clientes ou indivíduos. Além disso, alguns outros atributos foram removidos porque não acrescentavam nenhuma informação útil sobre o problema;
- preenchimento de valores ausentes: no primeiro experimento as instâncias com valores ausentes foram removidas. Porém, nesse processo quase 1000 instâncias eram removidas, e por isso o preenchimento de dados também foi realizado. As estratégias escolhidas

foram o preenchimento com a mediana da coluna, para atributos numéricos, e uma nova categoria para atributos categóricos;

- conversão para valores numéricos: depois de preenchidos, os atributos categóricos foram convertidos para valores numéricos, ou binários. Esse processo é necessário para a execução de vários dos algoritmos de AM implementados no scikit learn;
- normalização dos dados: como cada atributo (coluna) pode apresentar intervalos de valores diferentes e disformes, todos os atributos foram então reescalados dentro de um mesmo intervalo de valores. A opção foi transformar valores originais em novos valores dentro do intervalo [0,1].

## 3.5 Algoritmos para Classificação

Durante os experimentos foram usadas diferentes algoritmos seguindo diferentes viéses de aprendizado, ou seja, cada um aprende de uma forma diferente com os dados aos quais são alimentados. Nos experimentos os os seguintes algoritmos de classificação foram usados: Naï ve Bayes (NB), Árvore de Decisão (*Decision Tree*), *K-Nearest Neighbors* (KNN), Regressão Logística (RL), *Random Forest* (RF), *Support Vector Machines* (SVM) e o algoritmo de rede neural Multilayer Perceptron (MLP). Todos eles foram implementados usando o scikit learn e seus valores default de hiperparâmetros.

## 3.6 Reprodutibilidade dos experimentos

O dataset gerado possui classes levemente desbalanceadas, assim, a métrica de desempenho utilizada neste trabalho foi a acurácia balanceada por classes. Esta medida fornece uma alternativa à acurácia tradicional por não ser afetada pelo desbalanceamento e pode ser definida como sendo a acurácia média obtida em qualquer classe. Assim, temos uma melhor estimativa de desempenho dos modelos induzidos quando realizando a predição de ambas as classes (iminência de falha ou banco operando dentro da normalidade).

Durante os experimentos os dados foram divididos em conjuntos de treino e teste por meio da metodologia de validação cruzada com estratificação das classes. A validação cruzada permite reduzir a variância no desempenho real dos algoritmos, e a estratificação permite manter a mesma distribuição de classes nas partições amostradas. O número de partições foi definido como 10, e o processo foi repetido 5 vezes com inicialização aleatória dos exemplos contidos em cada partição. Assim, temos um total de 50 valores parciais de performance quando avaliamos o comportamento dos algoritmos.

Os experimentos relatados nesse trabalho foram realizados no ambiente virtual `Google Colaboratory`, com a linguagem `Python` em sua versão 3. As principais bibliotecas utilizadas no tratamento dos dados e criação dos modelos foram: `Pandas`; `Numpy` - para tratamento e conversão de dados; e `Scikit-learn` para implementação dos modelos de aprendizado de máquina. Foi feito uso também das bibliotecas: `matplotlib`, `seaborn` e `plotly` para criação

dos gráficos e visualização dos dados. Para fins de reprodutibilidade, todos os códigos usados para implementação do processo de avaliação e treinamento descrito neste capítulo pode ser encontrados nos links abaixo:

- Dataset:
  <https://docs.google.com/spreadsheets/d/1tXHWLf8K9SDaD_ayGMPkYiFKXvrBgBE0Rghftb86a1s
  edit?usp=sharing>
- Experimento: <https://colab.research.google.com/drive/1wHoKA5gR9nGuqdSrtzyPqs1AlEpW0cTT
  usp=sharing>

# 4 RESULTADOS

Neste capitulo são discutidos os resultados obtidos durante as experimentações realizadas. Tem-se como hipótese inicial de que um algoritmos de AM poderia prever e antecipar comportamentos de problema de performance no servidor SGBD, e assim, criar condições para prevenção de eventos de baixo desempenho.

## 4.1 Desempenho geral dos modelos induzidos

Para apuração dos resultados de acurácia o processo foi dividido em três etapas de experimentos. As execuções dos algoritmos foram divididas com o objetivo de testar diferentes cenários de predição com o mesmo dataset. Na primeira etapa, descrita como Experimentos 1, todas as amostras com dados ausentes foram removidas, e as remanescentes foram divididas entre treino e teste seguindo a metodologia de *holdout*, na proporção de 75% das amostras usadas para treino e 25% para a etapa de teste. Essa divisão dos dados foi feita pela função `train_test_split` da biblioteca `Scikit Learn`. A classe escolhida como objetivo de predição foi o consumo de CPU.

No segundo experimento (Experimento 2) a classe preditiva foi mantida (uso total de CPU). A alteração nesse cenário foi realizar imputação dos dados, preenchendo os valores ausentes via método da mediana. Dessa forma, aumentou-se a quantidade de amostras para uso nos algoritmos. No geral foi obtida uma pequena melhora nos percentuais de acerto para a maioria dos classificadores.

Para o terceiro experimento (Experimento 3), manteve-se a estratégia de preencher os dados (e ter mais dados para aprender), mas realizando a troca do atributo preditivo para a classe de atributos composta `PERFORMANCE`. Esse atributo combina a informação de três atributos descritivos: uso de cpu, load short cpu, e uso de memória. Com o novo atributo preditivo, obteve-se uma melhora significativa no desempenho dos algoritmos quando comparados aos resultados dos experimentos anteriores. Nesse experimento foram obtidos resultados de acerto acima de 96% de acurácia balanceada por classe para todos os algoritmos testados. Os resultados dos três cenários de experimentos são apresentados na Tabela 3.

Observou-se que para o primeiro teste com a exclusão das amostras com dados ausentes, os resultados obtidos variaram entre 61.08% para o classificador Naïve Bayes e 83.39% para o algoritmo de Árvore de Decisão. Para o segundo teste foi utilizado os dados preenchidos, com todas as amostras disponíveis para aplicação nos algoritmos. O objetivo era reavaliar a performance dos resultados, nessa etapa foram obtidos os valores mínimos de 60.88% no KNN e 83.27% para os algoritmos Árvore de Decisão e Random Forest. Na terceira etapa o teste foi realizado novamente com o conjunto inteiro dos dados, já que os resultados demonstraram mais eficiência nesse formato de dados, e o atributo PERFORMANCE como

Tabela 2 – Resultados dos experimentos em termos de acurácia balanceada por classe.

| | Exp. 1 | Exp. 2 | Exp. 3 |
|---|---|---|---|
| Naïve Bayes | 61.08 | 60.78 | 96.09 |
| Árvore de Decisão | 83.39 | 83.27 | 99.17 |
| KNN | 77.62 | 80.31 | 98.85 |
| Regressão Logística | 77.26 | 76.30 | 98.85 |
| Random Forest | 78.74 | 83.27 | 99.17 |
| SVM | 76.07 | 76.24 | 98.85 |
| MLP | 77.26 | 78.71 | 98.75 |

Fonte: Autoria própria

classe composta. Nesse teste foram obtidos valores de acurácia de 96.09% para o classificador Naïve Bayes até 99.17% para os classificadores Árvore de Decisão e Random Forest. No geral, podemos verificar que usar o atributo preditivo de forma simples baseado apenas no uso de CPU não demonstra uma eficiência confiável. Os resultados são então melhorados com o uso de mais características do conjunto de dados na composição da classe (atributo alvo), e consequentemente um aumento na confiança dos resultados da aplicação dos algoritmos.

## 4.2 Análise das predições dos modelos

Para a análise dos resultados, segui-se o modelo de relatório de métricas `Classification Report` do pacote `Scikit Learn`. Nas tabelas abaixo são apresentados os valores com maior eficiência obtidos no experimento 3, que faz uso da classe composta, e apresenta os resultados de todos os algoritmos experimentados. Os valores fornecidos por este relatório representam a capacidade do algoritmo de identificar os erros e acertos nas amostras do conjunto de dados. O retorno desses valores de resultado consistem em quatro maneiras de verificar as previsões, que são os valores `TN / verdadeiro negativo`: é uma amostra de caso negativo e que foi previsto como negativo, `TP / verdadeiro positivo`: uma amostra de caso positivo que foi prevista como positivo, `FN / falso negativo`: é uma amostra de caso positivo mas foi previsto como negativo, `FP / falso positivo`: é uma amostra de caso negativo que foi prevista como positiva. Essas métricas são representadas pela matriz de confusão na área de aprendizado de máquina, representada pela figura 7.

Para interpretação das informações presentes na figura, e leitura dos resultados apresentados para cada algoritmo, são utilizadas as métricas de `precision`, que representa o valor de quanto o algoritmo acertou corretamente os exemplos para cada amostra, na métrica `recall` são apresentados os valores da capacidade do classificador de encontrar as amostras positivas para cada classe e na métrica `f1-score` são apresentados os resultados da média harmônica dos valores de `precision` e `recall`, mostrando o valor de precisão geral para cada classe no conjunto de dados.

Conforme valores obtidos na tabela 3 verificamos que os valores de performance (f1-

Figura 7 – Matriz de Confusão.

| | | **Valor Predito** | |
|---|---|---|---|
| | | **Sim** | **Não** |
| **Real** | **Sim** | Verdadeiro Positivo (TP) | Falso Negativo (FN) |
| | **Não** | Falso Positivo (FP) | Verdadeiro Negativo (TN) |

Fonte: Acervo do autor.

Tabela 3 – Resultado geral dos classificadores. Classification Report.

| Algoritmo | métrica | classe 0 | classe 1 |
|---|---|---|---|
| Naive Bayes | precision | 0.68 | 1.00 |
| | recall | 1.00 | 0.46 |
| | f1-score | 0.81 | 0.63 |
| Arvore Decisão | precision | 0.98 | 1.00 |
| | recall | 1.0 | 0.98 |
| | f1-score | 0.99 | 0.99 |
| KNN | precision | 0.98 | 1.00 |
| | recall | 1.0 | 0.98 |
| | f1-score | 0.99 | 0.99 |
| Reg Logística | precision | 0.98 | 1.00 |
| | recall | 1.0 | 0.98 |
| | f1-score | 0.99 | 0.99 |
| Random Forest | precision | 0.99 | 1.00 |
| | recall | 1.0 | 0.98 |
| | f1-score | 0.99 | 0.99 |
| SVM | precision | 0.98 | 1.00 |
| | recall | 1.0 | 0.98 |
| | f1-score | 0.99 | 0.99 |
| MLP | precision | 0.98 | 1.00 |
| | recall | 1.0 | 0.98 |
| | f1-score | 0.99 | 0.99 |

Fonte: Autoria própria.

score) ficaram altos para quase todos os algoritmos, com exceção do classificador `Naive Bayes` onde a taxa acerto ficou em $72\%$. Embora resultados próximos de 1.00 sejam ideais, todos os algoritmos demonstrar tal comportamento, pode ser um indicativo de erro na modelagem.

Logo, foi preciso checar se não ocorreu uma superestimação dos valores gerados, fazendo com que o modelo decore o que deve ser feito para alcançar esse resultado, e não aprenda nada na prática.

Desse modo para que seja possível confrontar os valores de desempenho do `Classification_Report`, adicionamos mais uma métrica: a acurácia balanceada por classe (`balanced_accuracy_score`) também presente na biblioteca do `Scikit Learn`. Com essa medida é possível calcular a taxa de acerto balanceada para ambas as classes preditivas, esse método é útil para verificação de problemas de classificação binária e multi classe, obtendo o valor médio do resultado de todas as amostras presentes no conjunto de dados. O resultado do balanced_accuracy_score é verificado na tabela 4.

Tabela 4 – Resultado da métrica *Balance Accuracy Score (BAS)*.

| **Algoritmo** | **BAS** |
|---|---|
| Naïve Bayes | 0.729 |
| Árvore de Decisão | 0.991 |
| KNN | 0.987 |
| Regressão Logística | 0.987 |
| Random Forest | 0.991 |
| SVM | 0.987 |
| MLP | 0.987 |

Fonte: Autoria própria.

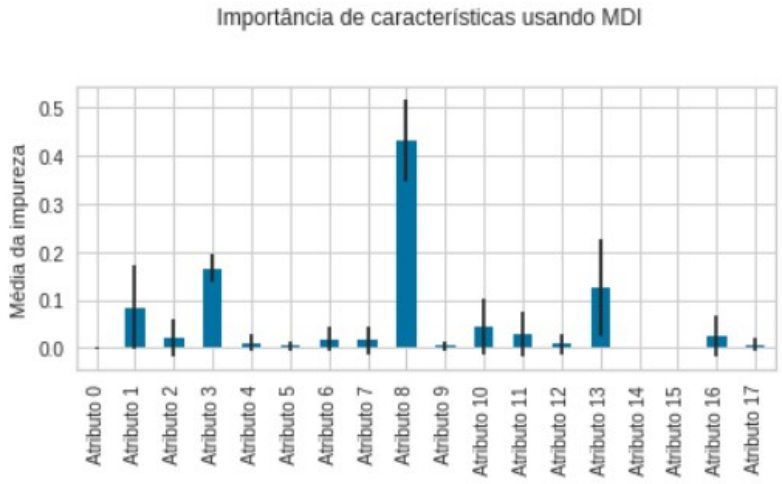
Com a análise pela métrica BAS evidenciou-se que os modelos de classificação experimentados são sim capazes de aprender com os dados do conjunto fornecido, e podem prever quedas de desempenho no servidor de banco de dados com base nos atributos escolhidos para verificação de performance. Esses resultados iniciais demonstram que todo o trabalho de tratamento dos dados, binarização e uso do método de escalonamento dos valores trouxeram resultados satisfatórios para o problema proposto. Claro que ainda carece de mais aprofundamentos e pesquisas, mas a hipótese inicial foi corroborada.

## 4.3 Análise das características mais descritivas

A fim de verificar a importância de cada uma das características no processo de aprendizagem dos algoritmos, tomou-se como base os valores da importância baseada na permutação do algoritmo Random Forest. Este procedimento se justifica pois o Random Forest pode indicar quais atributos são capazes de guiar o processo de decisão da classe.

A Figura 8 mostra os valores de importância dos atributos observados no dataset durante o Experimento 3. Na figura podemos verificar maior peso para as características de consumo de CPU load short, uso de memória do servidor e o tamanho da base de dados, calculado em gigabytes no conjunto de dados. Essas características são representadas pelo

Figura 8 – Importância de Características do dataset de acordo com o RF




Fonte: Autoria própria.

atributos 1, 3 e 8 respectivamente. O atributo ligado ao código 13 representa a informação de presença de triggers de auditoria no banco de dados, criadas para armazenar informações de operações do sistema para registro histórico de alterações nos dados. Apesar de apresentar uma alta relevância esse atributo não é considerado crítico em uma análise de performance, sendo assim, o seu valor não é incluído na classe preditiva.

# 5 CONCLUSÃO

Neste trabalho, buscou-se compreender o uso de algoritmos de Aprendizado de Máquina (AM), técnicas de análise e processamento de dados para verificar a possibilidade de identificação nos problemas de queda de performance em servidores de banco de dados. Com base na coleta de informações, estatísticas e métricas geradas pelo sistema operacional e Sistema Gerenciador de Banco de Dados (SGBD), foi possível extrair, tabelas e interpretar as principais características na identificação de problemas de desempenho. Com a aplicação dos algoritmos de AM buscou-se demonstrar um estudo de caso, por meio de experimentos iniciais, que é possível classificar erros ou quedas de desempenho no servidor, e com isso reduzir o risco de parada na operação do negócio. Os resultados obtidos com os algoritmos selecionados tiveram valores de acurácia balanceada por classes (BAS) acima de 92%. A análise e leitura dos resultados apresentados dos métodos `classification_report` e do `balanced_accuracy_score` foram de grande utilidade para o entendimento da interpretação dos resultado alcançados, facilitando o entendimento dos dados e eficiência na escolha dos atributos que integraram a classe preditiva mais assertiva.

A experiência obtida neste trabalho foi de extrema relevância para o meu desenvolvimento acadêmico e profissional, dados os inúmeros desafios propostos em todos os aspectos que o tema exige. Pude aprender mais desde a obtenção, preparação e tratamento dos dados com a linguagem de programação, até o aprofundamento dos estudos na área de aprendizado de máquina, para que fosse possível realizar todas as etapas e compreender os resultados obtidos. Assim, espera-se que o presente trabalha possa abrir novas perspectivas de investigação para o problema apresentado.

## 5.1 Limitações e Dificuldades

Abordando os aspectos de limitações, os principais empecilhos deste trabalho ocorreram na etapa de preparação dos dados, devido a necessidade de transformar os atributos no formato que fosse possível realizar a aplicação nos principais algoritmos de AM. Tal tarefa demandou grande parte do tempo dispendido na elaboração deste trabalho.

Outro aspecto que não foi abordado neste trabalho foi o teste massivo de possibilidades diferentes da configuração do atributo preditivo, testando os atributos mais relevantes de forma com que fosse possível simular outras possibilidades de configuração dessa classe. Além disso, uma característica importante que não é abordada neste trabalho foi a inclusão de características de performance do conjunto de `bufferpools` de memória do SGBD DB2. Esses valores são relevantes na verificação de performance de uma banco de dados, já que um baixo desempenho desses conjuntos de memória indicam alto consumo de leitura em disco rígido, que geralmente são muito mais lentos que o processamento de memória em um servidor.

Outro ponto que demandou esforço foi o estudo da linguagem de programação `python` escolhida para o desenvolvimento deste trabalho, sendo necessário o estudo de características e funções até então desconhecidas do discente. Assim como também a revisão e estudos no tema de aprendizado de máquina, que também consumiu uma boa parte do desenvolvimento da pesquisa.

## 5.2 Trabalhos Futuros

No desenvolvimento deste trabalho foram abordadas as possibilidades de utilizar algoritmos de AM para prever quedas de performance em bancos de dados DB2, com base análise das características do SGBD e sistema operacional. Há outras possibilidades de pesquisa relacionadas ao tema, sendo revisão dos atributos no conjunto de dados, adicionando novas características, inclusão de novos atributos preditivos ao modelo proposto. Realizar o teste com outros algoritmos de AM. Aumentar o período de coleta e número de bancos de dados envolvido para que sejam criadas mais amostras, aumentando a precisão de acerto dos algoritmos, visto que os resultados alcançados neste trabalho apresentam possibilidades de novas explorações, tanto na parte dos dados como nas técnicas e uso dos algoritmos.

# Referências


ACADEMY, D. S. **Capítulo 62 – O Que é Aprendizagem Por Reforço?** 2021. Disponível em: <https://www.deeplearningbook.com.br/o-que-e-aprendizagem-por-reforco/>. Citado na página 16.

BERTOZZO, R. J. Aplicação de machine learning em dataset de consultas médicas do sus. 2019. Citado na página 12.

BISHOP, C. M. **Pattern recognition and machine learning (information science and statistics)**. [S.l.]: Springer New York, NY, USA, 2007. Citado 2 vezes nas páginas 12 e 16.

ESCOVEDO, T.; KOSHIYAMA, A. **Introduco a Data Science**. [S.l.]: Casa do Código, 2020. v. 1. Citado na página 15.

HARRISON, M. **Machine Learning – Guia de Referência Rápida: Trabalhando com dados estruturados em Python**. Novatec Editora, 2019. ISBN 9788575228173. Disponível em: <https://books.google.com.br/books?id=i-7CDwAAQBAJ>. Citado na página 18.

LEARN, S. **Decision Tree**. 2021. Disponível em: <https://scikit-learn.org/stable/modules/tree.html#classification>. Citado na página 17.

LEARN, S. **KNN**. 2021. Disponível em: <https://scikit-learn.org/stable/modules/neighbors.html>. Citado na página 17.

LEARN, S. **Logistic Regression**. 2021. Disponível em: <https://scikit-learn.org/stable/modules/linear_model.html>. Citado na página 17.

LEARN, S. **MLP**. 2021. Disponível em: <https://scikit-learn.org/stable/modules/neural_networks_supervised.html>. Citado na página 17.

LEARN, S. **Naive Bayes**. 2021. Disponível em: <https://scikit-learn.org/stable/modules/naive_bayes.html>. Citado na página 16.

LEARN, S. **random Forest**. 2021. Disponível em: <https://scikit-learn.org/stable/modules/ensemble.html#forest>. Citado na página 17.

LEARN, S. **SVM**. 2021. Disponível em: <https://scikit-learn.org/stable/modules/svm.html>. Citado na página 17.

MITCHELL, T. **Machine Learning**. McGraw-Hill, 1997. (McGraw-Hill International Editions). ISBN 9780071154673. Disponível em: <https://books.google.com.br/books?id=EoYBngEACAAJ>. Citado 2 vezes nas páginas 12 e 16.

MONARD, J. A. B. M. C. **Sistemas Inteligentes**. [S.l.]: Ciencia Moderna, 2003. v. 1. Citado na página 15.

MUELLER, J.; MASSARON, L. **Aprendizado de Máquina Para Leigos**. Alta Books, 2019. (Para Leigos). ISBN 9788550815985. Disponível em: <https://books.google.com.br/books?id=ZmXTDwAAQBAJ>. Citado na página 16.