A MINERAÇÃO DE DADOS COMO FERRAMENTA PARA AVALIAÇÃO DE DADOS RELACIONADOS À PANDEMIA DO COVID-19

Lucas Ferreira Hubie (UEPG) E-mail: lucas.ferreira.hubie@gmail.com
Mateus Bueno Gonçalves (UEPG) E-mail: bueno.030702@gmail.com
Arion de Campos Junior (UEPG) E-mail: arion@uepg.br

Departamento de Informática - Universidade Estadual de Ponta Grossa (UEPG)
Ponta Grossa - PR - Brasil

Resumo: O objetivo dessa pesquisa é apresentar os conceitos de mineração de dados aplicados na área da saúde, em especial aos dados gerados por consequência da pandemia do COVID-19. Para metodologia do trabalho utilizamos o software WEKA para testar os algoritmos em bases de dados sobre o COVID-19, obter os resultados em termos de indicadores de desempenho e posteriormente analisar os resultados utilizando testes estatísticos. Os algoritmos escolhidos para serem testados na pesquisa foram o J48, REPTree, RandomForest, JRip e OneR. Como conclusão do trabalho podemos destacar o bom desempenho da maioria dos algoritmos com exceção do algoritmo OnerR, que obteve piores resultados em comparação aos demais algoritmos utilizados no trabalho.

Abstract: The aim of this research is to present the concepts of data mining applied in the health area, especially to the COVID-19 pandemic database. For the methodology of the work, we applied the WEKA software to test the algorithms in databases about COVID-19 to get the results in terms of performance indicators. After we analyzed the results by using statistical tests. The chosen algorithms to be tested in this research were J48, REPTree, RandomForest, JRip and OneR. As conclusion of the work, we can highlight the good performance of most algorithms with the exception of the OnerR algorithm, which had worse results compared to the other algorithms applied in this work.

Palavras-chave: Mineração de dados; COVID-19; Algoritmos de Classificação;

1. Introdução

No final do ano de 2019, começaram a surgir os primeiros casos de COVID-19 que é uma doença respiratória causada pelo coronavírus da síndrome respiratória, os coronavírus são uma grande família de vírus comuns em muitas espécies diferentes de animais, incluindo camelos, gado, gatos e morcegos. Raramente, os coronavírus que infectam animais podem infectar pessoas, como exemplo do MERS-CoV e SARS-CoV. No entanto, houve a transmissão de um novo coronavírus (SARS-CoV-2), o qual foi identificado em Wuhan, na China, com o primeiro caso confirmado em 12 de dezembro de 2019 [1]. Esta doença afeta o sistema respiratório do infectado, com sintomas variáveis entre indivíduos. Idosos e pessoas com comorbidades como problemas cardiovasculares e respiratórios, tendem a apresentar sintomas mais graves. [2].

Devido a sua alta capacidade de transmissão, logo alastrou-se por todas as regiões do planeta, tornando-se um problema global. A Organização Mundial da Saúde (OMS) declarou medidas sanitárias como distanciamento social e uso de máscaras para prevenir a infecção. A OMS listou os principais sintomas causados: febre, tosse seca e cansaço, havendo outros sintomas menos comuns como dores de cabeça, dores na garganta, perda de olfato e paladar, erupções na pele, entre outros [3].

À medida que o vírus foi se espalhando rapidamente, os hospitais e as entidades de saúde foram ficando com escassez de recursos e de profissionais da área, já que a demanda subiu consideravelmente e, apesar dos esforços para conter a disseminação do vírus, ele continuava a se propagar. Necessitando de dados sobre a propagação e sobre a

doença para poder gerar um estudo sobre o comportamento do vírus, as instituições de saúde passaram a informatizar cada vez mais o sistema com dados de prontuários de pacientes, algo que já havia se iniciado alguns anos atrás [4] e se intensificou durante a pandemia. Logo, começaram a ser disponibilizadas bases de dados para estudos, mas como as informações são armazenadas em grande quantidade e geralmente não seguem um padrão específico, foram se mostrando necessárias tecnologias capazes de processar esses dados e trazer informações relevantes no auxílio de tomada de decisão.

Diante de toda essa problemática, tornou-se necessária a utilização de ferramentas para facilitar o diagnóstico inicial dos pacientes, como por exemplo, reconhecimento de imagens para realizar análises de raios-x de pulmões de pacientes e auxiliar no diagnóstico da doença [5]. Logo mostrou-se viável um sistema de previsão automático que pudesse auxiliar a identificar a doença de COVID-19 em uma pessoa. Com isso, técnicas de mineração de dados (MD) e aprendizagem de máquina se mostram eficazes para montar um modelo de classificação preditivo. As técnicas de mineração de dados podem ser definidas descendendo de 3 linhagens: a estatística clássica, a inteligência artificial (IA) e o *machine learning* [6]. Além dessas três linhagens, a MD tem inúmeras ramificações, sendo algumas mais conhecidas como redes neurais, indução de regras, árvores de decisão, análise de séries temporais e visualização.

Neste contexto serão aplicadas técnicas de mineração de dados focando principalmente na etapa de extração de informação, para apresentação dos dados obtidos, tendo um processo de avaliação das principais técnicas de MD para classificação e predição que visam a montagem de um modelo para rotular registros analisados.

O objetivo deste trabalho é utilizar a mineração de dados através do *software* de código aberto Weka[7], para obter informações relevantes sobre a eficácia dos algoritmos de mineração de dados, J48, REPTree, Random-Forest, JRip e OneR aplicados em bases de dados relacionadas ao COVID-19 e demonstrar quais informações podemos utilizar sobre pacientes com COVID-19 ou sem COVID-19, sendo utilizadas técnicas de comparações estatísticas sobre o desempenho.

2. REVISÃO BIBLIOGRÁFICA E TRABALHOS RELACIONADOS

2.1. Mineração de Dados1. Introdução

No final do ano de 2019, começaram a surgir os primeiros casos de COVID-19 que é uma doença respiratória causada pelo coronavírus da síndrome respiratória, os coronavírus são uma grande família de vírus comuns em muitas espécies diferentes de animais, incluindo camelos, gado, gatos e morcegos. Raramente, os coronavírus que infectam animais podem infectar pessoas, como exemplo do MERS-CoV e SARS-CoV. No entanto, houve a transmissão de um novo coronavírus (SARS-CoV-2), o qual foi identificado em Wuhan, na China, com o primeiro caso confirmado em 12 de dezembro de 2019 [1]. Esta doença afeta o sistema respiratório do infectado, com sintomas variáveis entre indivíduos. Idosos e pessoas com comorbidades como problemas cardiovasculares e respiratórios, tendem a apresentar sintomas mais graves. [2].

Devido a sua alta capacidade de transmissão, logo alastrou-se por todas as regiões do planeta, tornando-se um problema global. A Organização Mundial da Saúde (OMS) declarou medidas sanitárias como distanciamento social e uso de máscaras para prevenir a infecção. A OMS listou os principais sintomas causados: febre, tosse seca e cansaço, havendo outros sintomas menos comuns como dores de cabeça, dores na garganta, perda de olfato e paladar, erupções na pele, entre outros [3].

À medida que o vírus foi se espalhando rapidamente, os hospitais e as entidades de saúde foram ficando com escassez de recursos e de profissionais da área, já que a demanda subiu consideravelmente e, apesar dos esforços para conter a disseminação do vírus, ele continuava a se propagar. Necessitando de dados sobre a propagação e sobre a doença para poder gerar um estudo sobre o comportamento do vírus, as instituições de saúde passaram a informatizar cada vez mais o sistema com dados de prontuários de pacientes, algo que já havia se iniciado alguns anos atrás [4] e se intensificou durante a pandemia. Logo, começaram a ser disponibilizadas bases de dados para estudos, mas como as informações são armazenadas em grande quantidade e geralmente não seguem um padrão específico, foram se mostrando necessárias tecnologias capazes de processar esses dados e trazer informações relevantes no auxílio de tomada de decisão.

Diante de toda essa problemática, tornou-se necessária a utilização de ferramentas para facilitar o diagnóstico inicial dos pacientes, como por exemplo, reconhecimento de imagens para realizar análises de raios-x de pulmões de pacientes e auxiliar no diagnóstico da doença [5]. Logo mostrou-se viável um sistema de previsão automático que pudesse auxiliar a identificar a doença de COVID-19 em uma pessoa. Com isso, técnicas de mineração de dados (MD) e aprendizagem de máquina se mostram eficazes para montar um modelo de classificação preditivo. As técnicas de mineração de dados podem ser definidas descendendo de 3 linhagens: a estatística clássica, a inteligência artificial (IA) e o *machine learning* [6]. Além dessas três linhagens, a MD tem inúmeras ramificações, sendo algumas mais conhecidas como redes neurais, indução de regras, árvores de decisão, análise de séries temporais e visualização.

Neste contexto serão aplicadas técnicas de mineração de dados focando principalmente na etapa de extração de informação, para apresentação dos dados obtidos, tendo um processo de avaliação das principais técnicas de MD para classificação e predição que visam a montagem de um modelo para rotular registros analisados.

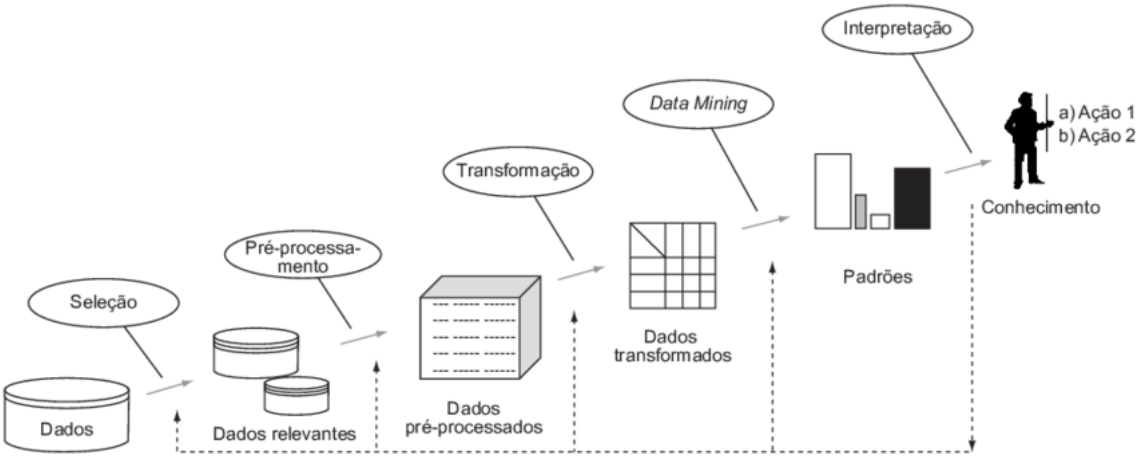
O objetivo deste trabalho é utilizar a mineração de dados através do *software* de código aberto Weka[7], para obter informações relevantes sobre a eficácia dos algoritmos de mineração de dados, J48, REPTree, Random-Forest, JRip e OneR aplicados em bases de dados relacionadas ao COVID-19 e demonstrar quais informações podemos utilizar

Existe uma grande quantidade de dados gerados por entidades de saúde, muitas vezes não se utilizam de formas corretas para obtenção de informações relevantes, o que acaba gerando perda de tempo e também de recursos [4]. Isso é muito prejudicial, pois em meio ao ambiente caótico da pandemia, tempo e recursos são imprescindíveis. O correto é recorrer aos meios seguros e confiáveis para se obter informações relevantes das bases de dados. Um dos métodos que se mostra eficaz para isto é a mineração de dados. A mineração de dados (MD) é uma tecnologia que pode trazer dados relevantes de uma grande quantidade de informações [4].

A mineração de dados é uma das etapas de um processo chamado *Knowledge Discovery in Databases* (KDD), ou seja Descoberta de Conhecimento em Bases de Dados, que consiste em identificar nos dados padrões válidos, novos padrões, potencialmente úteis e que sejam compreensíveis, contando com conceitos de base de dados, métodos estatísticos, ferramentas de visualização e técnicas de inteligência artificial [8]. O KDD se divide nas seguintes etapas, sendo elas: seleção, pré-

processamento, transformação, mineração de dados e avaliação/interpretação.

Figura 1. Processo KDD



Fonte: (Fayyad et al., 1996).

Nas primeiras fases do processo, o objetivo trata da escolha da base de dados, a eliminação dos dados imperfeitos, nulos e incorretos, seguindo para transformação dos dados, a qual irá depender do objetivo buscado e do algoritmo que vai ser utilizado. A melhoria na qualidade dos dados a serem processados é importante pois refletirá diretamente na eficácia final da mineração. Depois destas etapas se segue para a mineração dos dados, a etapa mais importante do processo, na qual será utilizado algum algoritmo com determinada técnica que irá representar um grupo de dados.

Dentro da mineração há diferentes técnicas e algoritmos que possuem diferentes finalidades. Entre as técnicas de MD mais utilizadas se destacam as Árvores de Decisão, Regras de Classificação e Redes Neurais, sendo que cada uma possui suas características próprias e se mostra mais eficiente dependendo do objetivo. Com a base de dados relacionada à doença da COVID-19,pode-se aplicar as referidas técnicas para saber quais delas são as mais eficazes. E para definição dos algoritmos utilizados deve-se primeiro definir quais os objetivos do estudo, podendo ter dois tipos de padrões, sendo eles preditivos e descritivos [8]. Na predição utiliza-se dados estatísticos para prever probabilidades futuras, já na descrição objetiva-se o entendimento dos dados existentes e geração de conhecimento.

As Árvores de Decisão são representadas como uma árvore usando nós e galhos, onde cada nó equivale a um atributo do banco de dados e uma decisão sobre a variável contida nele, a qual determina quais serão os nós seguintes (nós filhos), podendo ser aplicada a grandes bases de dados e se adaptando a diversos tipos de dados [8], sendo seu objetivo formar um modelo de predição para os novos dados a serem inseridos.

Na classificação, os registros são distribuídos em categorias, utilizando-se de uma função que realiza a associação dos registros com características em comum, verificando os atributos similares entre os mesmos [9].Nesta pesquisa serão utilizados algoritmos baseados em técnicas de árvores de decisão e regras de classificação e serão apresentados nas subseções a seguir.

2.1.1 Algoritmo J48

Considerado um dos mais clássicos algoritmos de árvore de decisão, trabalha com atributos discretos e contínuos e também permite a utilização de atributos desconhecidos. É um algoritmo que utiliza o método divisão e conquista para aumentar a capacidade de predição das árvores de decisão.

2.1.2 Algoritmo REPTree

O algoritmo desenvolve uma árvore de decisão usando a variância de informação e usando as podas de erro reduzido. A poda é um recurso utilizado para interromper a expansão da árvore, evitando que o modelo fique desnecessariamente complexo.

Segundo o próprio *software* Weka, o algoritmo é definido como uma classe para construção de uma árvore que considera atributos K escolhidos aleatoriamente em cada nó.

2.1.3 Algoritmo Random Forest

Pode ser dividido em 4 passos, sendo o primeiro, a seleção aleatória de alguns atributos, o segundo a seleção do atributo mais adequado para a posição de nó raiz, o terceiro a geração dos nós filhos e o quarto passo repete os passos acima até que se atinja a quantidade de árvores desejada.

Assim que gerado o modelo, as previsões são feitas por votações, cada subárvore toma uma decisão a partir dos dados apresentados e a decisão mais votada é a resposta apresentada pelo algoritmo.

2.1.4 Algoritmo JRip

Pode ser dividido em duas fases, a primeira gera um conjunto de regras para a comparação e a segunda otimiza o conjunto de regras iniciais para diminuir erros e tornar o processo mais seletivo. Esses passos são repetidos diversas vezes durante o processo.

O algoritmo também implementa uma ordenação baseada na técnica de dividir para conquistar, onde o número de exemplos são elencados para aprendizagem, fazendo esse esquema para todo exemplo em sua base de regras. O processo é repetido até que as chances de erro sejam as menores possíveis de serem detectadas pelo

sistema. A regra com menor incidência de erro é escolhida para a classificação, auxiliando na determinação da classe minoritária.

2.1.5 Algoritmo OneR

De "*One Rule*", é um algoritmo de classificação simples e objetivo, que produz uma regra para cada atributo e seleciona a regra com menor erro total como sua regra única. O algoritmo é um classificador simples e utiliza um método de classificação de custo reduzido e com uma alta acurácia.

2.2 Trabalhos Relacionados

A mineração de dados relacionados à saúde já é bem vasto e agora temos um aumento muito maior com temas relacionados ao COVID-19.

Como primeiro trabalho relacionado, o artigo Aplicação de Mineração de Dados em Informações Oriundas de Prontuários de Paciente, trata sobre um estudo referente ao uso da mineração de dados na descoberta de conhecimento vindas das informações de prontuários de pacientes, verificando a eficácia da técnica para tal [4]. Este trabalho se seguiu por etapas de levantamentos bibliográficos, sobre mineração de dados, sobre o formato dos prontuários e apresentando algumas ferramentas de código aberto para MD.

O total de registros foi superior a 6 milhões de registros, e uma taxa de porcentagem de aproveitamento dos dados é exibida após a análise da ferramenta. 52,18% dos conjuntos de dados fornecidos pelo portal vieram da busca pelo termo "saúde" e 70,35% de todos os conjuntos de dados do portal podem ser importados de forma nativa pela ferramenta [4]. No fim, a conclusão foi que a MD funciona bem sobre dados estruturados e que pode ser usada como meio de descoberta sobre conhecimento, vindo de informações providas por sistemas de saúde.

Outro trabalho que se relaciona diretamente com a nossa pesquisa é o *Predictive Data Mining Models for Novel Coronavirus Infected Patients Recovery* (Modelos de mineração de dados preditivos para recuperação de pacientes infectados com o novo coronavírus (COVID-19)), [10]. O objetivo deste trabalho é desenvolver modelos de mineração de dados para a previsão de recuperação de pacientes infectados pelo novo coronavírus (COVID-19). O método de trabalho foi utilizar algoritmos como DT (*decision tree*) árvore de decisão, *naive bayes*, *random forest*, regressão logística, SVM (*support vector machine*) e KNN (*k nearest neighbor*) com a linguagem de programação *Python* para desenvolver os modelos. A metodologia empregada foi dividida em duas partes, a primeira é descrita como coleção e descrição do conjunto de dados, na qual a base de dados escolhida para o trabalho foi analisada e seus atributos descritos. A segunda parte é chamada de preparação do conjunto de dados, e é a etapa em que os dados são discriminados e parâmetros foram definidos. O próximo passo foi desenvolver os modelos utilizando *Python*, e aplicar os algoritmos para posteriormente serem analisados. A avaliação do desempenho foi definida pela precisão, que determina a porcentagem das instâncias do conjunto de dados classificados corretamente para o modelo desenvolvido pelo algoritmo *predictive DM*. O modelo que teve um melhor desempenho foi o modelo DT (*decision tree*), mais conhecido como árvore de decisão, apresentando uma eficiência de quase 100 % , uma leve vantagem em relação aos outros algoritmos.

Na dissertação Classificação de agregados de rochas ígneas quanto à sua Alteração por meio de processamento digital de imagens [11], são apresentados conceitos de utilização de Processamento Digital de Imagens (PDI) e aprendizado de máquina para classificação dos agregados e comparar os resultados com os métodos utilizados pela engenharia civil. A metodologia utilizada neste trabalho buscou investigar imagens de agregados e encontrar alternativas mais baratas e eficientes para classificação de materiais que possam ser empregados em obras de pavimentação. Utilizando de algumas técnicas de PDI como análises de textura e histograma, matriz de co-ocorrência, *local binary patterns*, *local binary patterns uniform* e entropia de *shannon*. Além de outras abordagens como *data augmentation* que é uma abordagem que consiste em aumentar a base de dados através de técnicas computacionais, com o objetivo de melhorar os resultados obtidos [12]. A mineração de dados nesta pesquisa está atrelada ao uso de aprendizado de máquina supervisionado, utilizando técnicas de classificação em *machine learning*. Técnicas como KNN, Árvore de decisão (*Decision Tree* (DT)), *Random Forest*, *Naive Bayes* e *Multi-layer Perceptron*. Para as medidas de avaliação dessas classificações foram utilizadas as métricas: matriz de confusão, precisão, revocação ou *recall* e *score* f1. Para as análises estatísticas dos resultados foram empregados dois testes, o de *Friedman* e o

teste de *Nemenyi*.

Outro trabalho relacionado ao assunto da nossa pesquisa é a Análise comparativa de algoritmos de árvore de decisão do sistema Weka para classificação do uso e cobertura da terra [13], que também está relacionado ao tema de processamento de imagens como a pesquisa citada anteriormente. Porém, neste trabalho o foco é a utilização de algoritmos de árvore de decisão, utilizando o *software* Weka. Assim como na nossa pesquisa, eles utilizam de uma ferramenta para auxiliar nas análises comparativas entre os algoritmos escolhidos. Na metodologia deste trabalho utilizou-se o Weka para testar os algoritmos e obter os resultados para avaliação. Para a avaliação foram considerados o tamanho total da árvore de decisão, o número total de folhas, o tempo gasto para geração da árvore de decisão, o número de *pixels* corretamente classificados, o número de *pixels* incorretamente classificados e o valor do índice *Kappa*, obtido pela matriz de confusão [13].

Em adição aos trabalhos relacionados temos Avaliação de Desempenho do Algoritmo JRip na Classificação do Diagnóstico de Doenças Cardíacas [14], sendo que o objetivo foi encontrar o melhor resultado para o algoritmo de JRip no *software* do weka, sendo utilizado uma base de dados relacionada a doenças cardíacas. Para realizar as simulações foram utilizadas as técnicas de *Cross-validation*, *Percentage split* e *Use training set* que são disponibilizados pelo Weka. Para forma de avaliação dos resultados foi considerada a matriz confusão e o índice kappa, após as avaliações concluiu-se que o algoritmo JRip é muito eficiente na obtenção de dados, em um base com conclusão simples entre duas opções, sendo a deste caso o fato de ter ou não doença cardíaca.

3. METODOLOGIA

A metodologia do trabalho aplicada no desenvolvimento da pesquisa foi dividida em três etapas: revisão da bibliografia, execução dos algoritmos no *software* Weka e avaliação dos resultados.

3.1 Revisão da Bibliografia

Como apresentado no tópico anterior, a metodologia da pesquisa iniciou-se pela revisão da bibliografia, onde se pesquisou por artigos no portal de periódicos CAPES[15], e foram estudados os conceitos gerais da mineração de dados, as bases de dados relacionados à pandemia do COVID-19 e os trabalhos relacionados a pesquisa. Outro ponto da revisão é como a mineração de dados tem sido utilizada nessas bases para trazer resultados relevantes nas pesquisas envolvendo a área da saúde.

Na revisão bibliográfica, notou-se a importância e a quantidade de dados relacionados à saúde, concluindo-se que o uso de mineração de dados aplicados a esses dados é um tema muito importante que merece ser investigado.

3.2 Execução dos algoritmos no *software* WEKA

Esta etapa está dividida em duas partes: fonte de dados e execução dos algoritmos. Serão apresentadas a fonte de dados utilizadas, e o ambiente utilizado para execução dos algoritmos.

3.2.1 Fonte de dados

A fonte de dados escolhida para análise de dados é proveniente da plataforma *Kaggle* com nome de "Sintomas e presença de COVID" [16], sendo uma base de dados com informações sobre os possíveis sintomas de quem foi infectado pela doença, e um campo se existe a presença de COVID-19. O conjunto de dados foi montado com os recursos fornecidos pela Organização Mundial da Saúde (OMS) sobre os possíveis sintomas com o objetivo de estudo para verificar a presença de COVID-19.

Quadro 1: Campos da Base de Dados

Problema respiratório
Febre
Tosse seca
Dor de garganta
Nariz escorrendo
Asma
Doença Pulmonar Crônica
Dor de cabeça
Doença cardíaca
Diabetes
Hipertensão
Fadiga
Gastrointestinal
Viagens ao exterior
Contato com Paciente COVID
Participou de Grande Encontro de Pessoas
Locais expostos públicos visitados
Família trabalhando em locais públicos expostos
Usando máscaras
Sanitização do Mercado
COVID-19

Os campos apresentados no quadro 1 são dados binários e compreendem os atributos da base de dados. O último campo é a classe da base escolhida e os outros são atributos.

3.2.2 Execução dos algoritmos

Para a execução dos algoritmos utilizou-se o *software open source* WEKA na versão 3.8.5. O *software* foi desenvolvido pela *Waikato University*, Universidade de *Waikato* da Nova Zelândia e emitido sob a GNU *General Public License*.

Weka é uma coleção de algoritmos de aprendizado de máquina para tarefas de mineração de dados e contém ferramentas para preparação de dados, classificação, regressão, *clustering*, mineração de regras de associação e visualização dos dados [7].

Para esta pesquisa buscou-se uma base de dados a ser estudada e entendida. Onde fosse possível analisar os dados e o que esses dados podem proporcionar para serem trabalhados. Nesta etapa verificou-se os atributos da base de dados escolhida, buscando

entender melhor seus relacionamentos e qual a sua relevância perante a aplicação dos algoritmos e quais resultados poderiam ser obtidos. Os algoritmos escolhidos são J48, *REPTree*, *RandomForest*, JRip, OneR.

Além dos algoritmos de mineração de dados, o Weka disponibiliza algoritmos de fragmentação como o *Cross-validation (Folds)* e *Percentage split*. O algoritmo *Cross-validation* realiza um laço de repetição onde os *folds* são os números de pares e subconjuntos treinamento-teste fornecidos na entrada. Na *Percentage split* é criado um subconjunto de treinamento com i % do tamanho da base de dados, sendo *i* a porcentagem dada.

Para a representação dos resultados encontrados pelos algoritmos as seguintes informações serão apresentadas: algoritmo escolhido, algoritmo de fragmentação, taxa de acerto, estatística *Kappa* e matriz de confusão.

O algoritmo escolhido nada mais é que o algoritmo utilizado dentre as opções escolhidas para a pesquisa. O algoritmo de fragmentação foi o *Cross-validation* ou *Percentage split*. Taxa de acerto é a taxa geral de quanto o algoritmo conseguiu classificações corretas. Estatística *Kappa* é uma métrica que avalia o nível de concordância de uma tarefa de classificação na qual uma métrica irá considerar somente as concordâncias entre os classificadores, o que indica a coesão dos dados [17]. A matriz de confusão apresenta em números a quantidade de classificações corretas e incorretas.

Na matriz de confusão algumas métricas são disponibilizadas, como acurácia, sensibilidade, especificidade, eficiência, valor preditivo positivo, valor preditivo negativo e coeficiente de correlação de *Matthews* - coeficiente (PHI).

A acurácia é a proporção de predições corretas sem levar em consideração o que é positivo ou negativo. Sensibilidade é a proporção de verdadeiros positivos. Especificidade é a proporção de verdadeiros negativos. Eficiência é a média aritmética da Sensibilidade e Especificidade. Valor preditivo positivo é a proporção de verdadeiros positivos em relação a todas as predições positivas. Valor preditivo negativo é a proporção de verdadeiros negativos em relação a todas as predições negativas. E o coeficiente de correlação de *Matthews* é uma medida de qualidade de duas classificações binárias que pode ser usada mesmo se as classes possuem tamanhos bastante diferentes [18].

Para os cálculos das tabelas de métricas da matriz de confusão, foram utilizadas as seguintes equações:

Acurácia = total de acertos / total de dados do conjunto

Acurácia = (VP + VN) / (P + N)

Sensibilidade = acertos positivos / total de positivos

Sensibilidade = VP / (VP + FN)

Especificidade = acertos negativos / total de negativos

Especificidade = VN / (VN + FP)

Eficiência = (sensibilidade + especificidade) / 2

Valor preditivo positivo = acertos positivos / total de predições positivas

Valor preditivo positivo = VP / (VP + FP)

Valor preditivo negativo = acertos negativos / total de predições negativas

Valor preditivo negativo = VN / (VN + FN)

Correlação de *Matthews* = (VP * VN - FP * FN) / ((VP + FP) * (VP + FN) * (VN + FP) * (VN + FN))

Na Figura 2, há a demonstração visual de uma matriz de confusão que foi seguido como modelo para resolver as equações da matriz de confusão dos algoritmos utilizados na pesquisa.

Os campos destacados na matriz da Figura 2 são respectivamente:

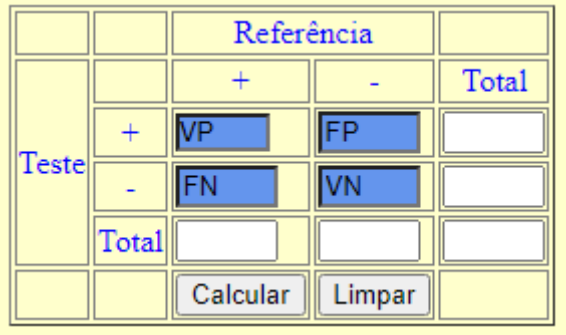
VP = Verdadeiro positivo

FP = Falso positivo

VN = Verdadeiro negativo

FN = Falso negativo

Figura 2. Referência para matriz de confusão



Representação gráfica para da tabela para cálculo das métricas da matriz de confusão

Fonte: http://www.cpdm.ufpr.br/testediaginostico.html

3. Avaliação dos algoritmos

Nesta etapa foram feitas as análises de testes comparativos para testar o desempenho dos algoritmos empregados. Para a análise, os resultados foram submetidos a testes estatísticos. Tais testes servem para afirmar que as diferenças observadas nos resultados não são casuais.

Os testes estatísticos são utilizados na forma de testes de hipóteses. Uma hipótese estatística é uma suposição que pode ser verdadeira ou falsa [19]. Exemplificando, uma hipótese pode afirmar que o algoritmo J48 é equivalente ao algoritmo JRip em termos de acurácia ou eficiência. Este tipo de teste é dependente de uma probabilidade, chamada *p-value*, que serve para embasar uma hipótese. Utiliza-se um nível de significância que é analisado e, então, é verificado se a hipótese de nulidade é rejeitada. Consequentemente uma hipótese alternativa é aceita, pois os resultados são diferentes.

Nessa pesquisa foi utilizado o teste não paramétrico conhecido como o teste de *Friedman*, que é utilizado para avaliar as diferenças entre várias amostras relacionadas. A hipótese nula (H0) para o teste de *Friedman* é de que não existem diferenças entre as amostras. Se a probabilidade *p-value* for menor que um nível de significância determinado, a hipótese nula é rejeitada e a hipótese alternativa (H1) é aceita [19]. Assim, pode-se concluir que há diferença entre os resultados. Porém, a informação de quais

grupos são diferentes não está disponível. Então, adota-se o procedimento de múltiplas comparações que permite determinar quais pares de grupos são diferentes entre si.

Para a aplicação do teste de *Friedman* utilizou-se a linguagem R, com auxílio do RStudio, um *software* livre de ambiente integrado de desenvolvimento para R. R é uma linguagem de programação para gráficos e cálculos estatísticos. Os gráficos gerados pela ferramenta foram do tipo diagrama de caixa ou boxplot e são importantes para visualizar os resultados obtidos. O diagrama identifica onde estão localizados os valores mais prováveis, a mediana e os valores extremos (atípicos, discrepantes ou *outliers*). Na execução do teste, foi utilizado uma média de 5 execuções para cada algoritmo escolhido, média obtida na execução da etapa anterior.

4. ANÁLISE DOS ALGORITMOS

Para análise decidiu-se utilizar alguns algoritmos para exemplificar as técnicas descritas na metodologia da pesquisa. Os algoritmos utilizados na análise são o J48, JRip, *REPTree*, *RandomForest* e *OneR*. Tal escolha justifica-se pela adoção de uma técnica baseada em árvores de decisão e outra, baseada em regras proposicionais.

Conforme citado na seção 3.2.1, a base utilizada tem alguns campos sobre sintomas, possível contato com paciente com COVID-19, entre outros. A classe utilizada para a avaliação é se está classificado como sim ou não para COVID-19.

Para cada algoritmo testado serão apresentadas duas tabelas, uma utilizando o algoritmo de fragmentação *Cross-validation* e outra tabela com o algoritmo de *Percentage split*.

No *Cross-Validation* os exemplos são divididos em n partições mutuamente exclusivas (*folds*) de tamanho aproximadamente igual. Os exemplos nos (n-1) *folds* são usados para treinamento e a hipótese induzida é testada no *fold* remanescente. O processo é repetido n vezes, cada vez considerando um *fold* diferente para teste.

À medida que o número de folds aumenta, esta sobreposição pode evitar que os testes estatísticos obtenham uma boa estimativa da quantidade da variação que seria observada se cada conjunto de treinamento fosse independente dos demais [20].

No *Percentage split* cria-se um subconjunto de treinamento com n% do tamanho da base de dados fornecida, sendo n a porcentagem dada. Então, basicamente trata-se de uma redução do tamanho da base de dados em uma porcentagem do seu total, visando a melhoria das estatísticas. Desses dois subconjuntos o primeiro fica responsável pelo treinamento e o segundo pelo teste, se for definido um *Percentage split* de 60%, então esses 60% serão treinados e os 40% serão utilizados para teste.

A sequência de apresentação da tabela será a seguinte: algoritmo escolhido, algoritmo de fragmentação, taxa de acerto, estatística *kappa* e matriz confusão.

Nas tabelas de 1 a 4, o algoritmo J48 foi dividido em duas execuções diferentes, uma com *Cross-validation* e outra com *Percentage split*. Na tabela 1 os valores encontrados executando o algoritmo J48 com *Cross-validation* utilizando 10 *folds*, obtendo uma taxa de acerto de 98.1% e índice *kappa* de 0.9422. E na tabela 2 as métricas da matriz confusão, explicado na seção 3.2, obtendo os resultados a serem avaliados na discussão dos resultados.

Na tabela 3 os valores encontrados utilizando o algoritmo J48 com *Percentage split*, nesse caso foi utilizado 40%, obtendo 97.6% e índice *kappa* de 0.9236. E na tabela

4 estão as métricas obtidas com os cálculos utilizando os valores da matriz confusão.

Tabela 1. Algoritmo J48 com Cross-validation

Algoritmo	Cross-validation (Folds)	Taxa de acerto	Estatística kappa	Matriz confusão
J48	10	98.1%	0.9422	a b 4320 63 a: Sim 36 1015 b: Não

Tabela 2. Métricas da matriz de confusão J48 com Cross-validation

Métricas	Resultados
Sensibilidade	99.2%
Especificidade	94.2%
Acurácia	98.2%
Correlação de Matthews	0.94
Eficiência	96.7%
Valor preditivo positivo	98.6%
Valor preditivo negativo	96.6%

Tabela 3. Algoritmo J48 com Percentage split

Algoritmo	Percentage split	Taxa de acerto	Estatística kappa	Matriz confusão
J48	40%	97.6%	0.9236	a b 2621 37 a: Sim 38 564 b: Não

Tabela 4. Métricas da matriz de confusão J48 com Percentage split

Métricas	Resultados
Sensibilidade	98.6%
Especificidade	93.8%
Acurácia	97.7%
Correlação de Matthews	0.92
Eficiência	96.2%
Valor preditivo positivo	98.6%
Valor preditivo negativo	93.7%

Nas tabelas de 5 a 8, o algoritmo JRip foi dividido em duas execuções diferentes, uma com *Cross-validation* e outra com *Percentage split*. Na tabela 5 os valores encontrados executando o algoritmo JRip com *Cross-validation* também utilizando 10 *folds*, obtendo uma taxa de acerto de 97.7% e estatística kappa de 0.9298 que se aproxima muito de 1, considerado ideal. E na tabela 6 as métricas da matriz confusão, explicado na seção 3.2, obtendo os resultados a serem avaliados na discussão dos resultados.

Na tabela 7 os valores encontrados utilizando o algoritmo JRip com *Percentage split*, nesse caso foi utilizado 40%, obtendo uma taxa de de acerto de 97.6% e índice *kappa* de 0.9206. E na tabela 8 estão as métricas obtidas com os cálculos utilizando os valores da matriz confusão.

Tabela 5. Algoritmo JRip com Cross-validation

Algoritmo	Cross-validation (Folds)	Taxa de acerto	Estatística kappa	Matriz confusão
JRip	10	97.7%	0.9298	a b 4299 84 a: Sim 37 1014 b: Não

Tabela 6. Métricas da matriz de confusão JRip com Cross-validation

Métricas	Resultados
Sensibilidade	99.1%
Especificidade	92.3%
Acurácia	97.8%
Correlação de Matthews	0.93
Eficiência	95.7%
Valor preditivo positivo	98.1%
Valor preditivo negativo	96.5%

Tabela 7. Algoritmo JRip com Percentage split

Algoritmo	Percentage split	Taxa de acerto	Estatística kappa	Matriz confusão
JRip	40%	97.6%	0.9206	a b 2618 40 a: Sim 38 564 b: Não

Tabela 8. Métricas da matriz de confusão JRip com Percentage split

Métricas	Resultados
Sensibilidade	98.6%
Especificidade	93.4%
Acurácia	97.6%
Correlação de Matthews	0.92
Eficiência	96.0%
Valor preditivo positivo	98.5%
Valor preditivo negativo	93.7%

Nas tabelas de 9 a 12, o algoritmo *REPTree* foi dividido em duas execuções diferentes, uma com *Cross-validation* e outra com *Percentage split*. Na tabela 9 temos os valores encontrados executando o algoritmo *REPTree* com *Cross-validation* também utilizando 10 *folds*, obtendo uma taxa de acerto de 97.6% e estatística *kappa* de 0.9232. Na tabela 10 as métricas da matriz confusão, explicado na seção 3.2, obtendo os resultados a serem avaliados na discussão dos resultados.

Na tabela 11 os valores encontrados utilizando o algoritmo *REPTree* com *Percentage split*, nesse caso foi utilizado 40%, obtendo uma taxa de acerto de 97.02% e índice *kappa* de 0.9225. E na tabela 12 estão as métricas obtidas com os cálculos utilizando os valores da matriz confusão.

Tabela 9. Algoritmo REPTree com Cross-validation

Algoritmo	Cross-validation (Folds)	Taxa de acerto	Estatística kappa	Matriz confusão
REPTree	10	97.6%	0.9232	a b 4320 63 a: Sim 67 984 b: Não

Tabela 10. Métricas da matriz de confusão REPTree com Cross-validation

Métricas	Resultados
Sensibilidade	98.5%
Especificidade	94%
Acurácia	97.6%
Correlação de Matthews	0.92
Eficiência	96.2%
Valor preditivo positivo	98.6%
Valor preditivo negativo	93.6%

Tabela 11. Algoritmo REPTree com Percentage split

Algoritmo	Percentage split	Taxa de acerto	Estatística kappa	Matriz confusão
REPTree	40%	97.02%	0.9025	a b 2599 59 a: Sim 38 564 b: Não

Tabela 12. Métricas da matriz de confusão REPTree com Percentage split

Métricas	Resultados
Sensibilidade	98.6%
Especificidade	90.5%
Acurácia	97%
Correlação de Matthews	0.90
Eficiência	94.5%
Valor preditivo positivo	97.8%
Valor preditivo negativo	93.7%

Nas tabelas de 13 a 16, o algoritmo *RandomForest* foi dividido em duas execuções diferentes, uma com *Cross-validation* e outra com *Percentage split*. Na tabela 13 os valores encontrados executando o algoritmo *RandomForest* com *Cross-validation* também utilizando 10 *folds*, obtendo uma taxa de acerto de 98.1% e estatística *kappa* de 0.9406. E na tabela 14 as métricas da matriz confusão, explicado na seção 3.2, obtendo os resultados a serem avaliados na discussão dos resultados.

Na tabela 15 os valores encontrados utilizando o algoritmo *RandomForest* com *Percentage split*, nesse caso foi utilizado 40%, obtendo uma taxa de de acerto de 98.02% e índice *kappa* de 0.9374. E na tabela 16 estão as métricas obtidas com os cálculos utilizando os valores da matriz confusão.

Tabela 13. Algoritmo RandomForest com Cross-validation

Algoritmo	Cross-validation (Folds)	Taxa de acerto	Estatística kappa	Matriz confusão
RandomForest	10	98.1%	0.9406	a b 4315 68 a: Sim 34 1017 b: Não

Tabela 14. Métricas da matriz de confusão RandomForest com Cross-validation

Métricas	Resultados
Sensibilidade	99.2%
Especificidade	93.7%
Acurácia	98.1%
Correlação de Matthews	0.94
Eficiência	96.5%
Valor preditivo positivo	98.4%
Valor preditivo negativo	96.8%

Tabela 15. Algoritmo RandomForest com Percentage split

Algoritmo	Percentage split	Taxa de acerto	Estatística kappa	Matriz confusão
RandomForest	40%	98.09%	0.9374	a b 2620 38 a: Sim 24 578 b: Não

Tabela 16. Métricas da matriz de confusão RandomForest com Percentage split

Métricas	Resultados
Sensibilidade	99.1%
Especificidade	93.8%
Acurácia	98.1%
Correlação de Matthews	0.94
Eficiência	96.5%
Valor preditivo positivo	98.6%
Valor preditivo negativo	96%

Nas tabelas de 17 a 20, o algoritmo *OneR* foi dividido em duas execuções diferentes, uma com *Cross-validation* e outra com *Percentage split*. Na tabela 17 temos os valores encontrados executando o algoritmo *OneR* com *Cross-validation* também utilizando 10 *folds*, obtendo uma taxa de acerto de 82.8% e estatística *kappa* de 0.4638. E na tabela 18 as métricas da matriz confusão, explicado na seção 3.2, obtendo os resultados a serem avaliados na discussão dos resultados.

Na tabela 19 os valores encontrados utilizando o algoritmo *OneR* com *Percentage split*, nesse caso foi utilizado 40%, obtendo uma taxa de de acerto de 82.8% e índice *kappa* de 0.4473. E na tabela 20 estão as métricas obtidas com os cálculos utilizando os valores da matriz confusão

Tabela 17. Algoritmo OneR com Cross-validation

Algoritmo	Cross-validation (Folds)	Taxa de acerto	Estatística kappa	Matriz confusão
OneR	10	82.8%	0.4638	a b 3878 505 a: Sim 429 622 b: Não

Tabela 18. Métricas da matriz de confusão OneR com Cross-validation

Métricas	Resultados
Sensibilidade	99.2%
Especificidade	93.8%
Acurácia	98.1%
Correlação de Matthews	0.94
Eficiência	96.5%
Valor preditivo positivo	98.5%
Valor preditivo negativo	96.8%

Tabela 19. Algoritmo OneR com Percentage split

Algoritmo	Percentage split	Taxa de acerto	Estatística kappa	Matriz confusão
OneR	40%	82.8%	0.4473	a b 2353 305 a: Sim 255 347 b: Não

Tabela 20. Métricas da matriz de confusão OneR com Percentage split

Métricas	Resultados
Sensibilidade	90.2%
Especificidade	53.2%
Acurácia	82.8%
Correlação de Matthews	0.45
Eficiência	71.7%
Valor preditivo positivo	88.5%
Valor preditivo negativo	57.6%

Na seção 5 os resultados são sintetizados e serão analisados em função de testes estatísticos.

5. DISCUSSÃO DOS RESULTADOS

A tabela 21 apresenta um resumo dos resultados obtidos pelos algoritmos utilizados na pesquisa. Tais resultados compreendem a média de cinco execuções para cada cenário e testes estatísticos foram conduzidos para comparar o desempenho dos algoritmos.

Os indicadores utilizados até então estão resumidos nesta tabela. Os algoritmos utilizados são representados da seguinte forma: J48 com *Cross-validation* (J48CV), J48 com *Percentage split* (J48PS), JRip com *Cross-validation* (JRipCV) e JRip com *Percentage split* (JRipPS), *REPTree* com *Cross-validation* (*REPTreeCV*), *REPTree* com *Percentage split* (*REPTreePS*), *RandomForest* com *Cross-validation* (*RandomForestCV*), *RandomForest* com *Percentage split* (*RandomForestPS*), *OneR* com *Cross-validation* (*OneRCV*), *OneR* com *Percentage split* (*OneRPS*).

Considerando os testes realizados, para todos os indicadores foram observadas diferenças estatísticas. Assim, múltiplas comparações foram realizadas. Tais comparações, incluindo o valor de cada *p-value* obtido estão sintetizados no apêndice A, tabela 24.

Os indicadores de desempenho foram apresentados na subseção 3.2.2. No apêndice B encontram-se os gráficos gerados pela ferramenta, do tipo diagrama de caixa

ou *boxplot*. Tais diagramas são úteis para visualizar os resultados obtidos. O diagrama identifica onde estão localizados os valores mais prováveis, a mediana e os valores extremos (atípicos, discrepantes ou *outliers*).

Tabela 21. Métricas de desempenho (resumo)

Algoritmo	Kappa	Acurácia	Sensibilidade	Especificidade	Eficiência	Valor pred. +	Valor pred. -	PHI
J48CV	0.9422	98.2%	**99.2%**	**94.2%**	**96.7%**	**98.6%**	**96.6%**	**0.94**
J48PS	0.9266	97.7%	98.6%	**93.8%**	96.2%	**98.6%**	93.7%	0.92
JRipCV	0.9298	97.8%	**99.1%**	92.3%	95.7%	98.1%	96.5%	0.93
JRipPS	0.9206	97.6%	98.6%	93.4%	96.0%	98.5%	93.7%	0.92
REPTreeCV	0.9232	97.6%	98.5%	94.0%	96.2%	98.6%	93.6%	0.92
REPTreePS	0.9025	97.0%	98.6%	90.5%	94.5%	97.8%	93.7%	0.90
RandomForestCV	0.9411	**98.1%**	**99.2%**	93.8%	**96.5%**	98.5%	**96.8%**	**0.94**
RandomForestPS	0.9374	**98.1%**	**99.1%**	93.8%	96.5%	98.6%	**96.0%**	**0.94**
OneRCV	0.4638	82.8%	99.2%	93.8%	96.5%	98.5%	96.8%	0.94
OneRPS	0.4473	82.8%	90.2%	53.2%	71.7%	88.5%	57.6%	0.45

A discussão dos resultados será realizada considerando cada indicador de desempenho, individualmente.

Após a realização dos testes estatísticos e das comparações par-a-par dos resultados dos algoritmos avaliados, os valores apresentados na tabela 21 foram destacados em negrito, quando o algoritmo foi superior em algum indicador de desempenho.

Por exemplo, em termos de acurácia, embora não haja um algoritmo que tenha se destacado em relação a todos os outros, observa-se que o algoritmo *RandomForestCV* e *RandomForestPS* foram superiores aos algoritmos *OneRCV* e *OneRPS*.

Também observa-se que os algoritmos *OneRCV* e *OneRPS* foram inferiores a quase todos os outros algoritmos. Por outro lado, considerando as outras estratégias, todas apresentaram bom desempenho nos indicadores utilizados.

Na tabela 22 foram reunidas as informações dos registros da base de dados cujos foram diagnosticados com COVID-19.

Tabela 22. Quantidade de Valores dos Registros de COVID

Nome do campo	Quantidade com Valor "Sim"	Quantidade com Valor "Não"	% Valor "Sim"	% Valor "Não"
Problema respiratório	3369	1014	76.86%	23.13%
Febre	3757	626	85.71%	14.28%
Tosse seca	3878	505	88.47%	11.52%
Dor de garganta	3669	714	83.70%	16.29%
Coriza	2375	2008	54.18%	45.81%
Asma	2124	2259	48.45%	51.54%
Doença Pulmonar Crônica	2008	2375	45.81%	54.18%
Dor de cabeça	2177	2206	49.66%	50.33%
Doença cardíaca	2064	2319	47.09%	52.90%
Diabetes	2131	2252	48.61%	51.38%
Hipertensão	2258	2125	51.51%	48.48%
Fadiga	2228	2155	50.83%	49.16%
Gastrointestinal	2054	2329	46.86%	53.13%
Viagem ao exterior	2451	1932	55.92%	44.07%
Contato com Paciente COVID	2582	1801	58.90%	41.09%
Participou de Grande Encontro	2442	1941	55.71%	44.28%
Locais Públicos Expostos Visitados	2403	1980	54.82%	45.17%
Família trabalhando em locais públicos expostos	1994	2389	45.49%	54.50%
Usando máscaras	0	4383	0%	100%
Sanitização do Mercado	0	4383	0%	100%

Como observa-se na tabela 22, a base possui dois campos cujos os valores foram mais impactantes para a análise dos algoritmos, que são os campos de utilização de máscaras, e o de sanitização de mercado, onde esses campos nos dizem que 100% dos infectados com COVID não utilizaram-se desses dois recursos.

Pode-se observar também que as pessoas que tiveram a doença, tiveram uma porcentagem maior de sintomas considerados mais comuns para a doença, como por exemplo, 88.47% das pessoas que tiveram o COVID-19, tiveram tosse seca. Outro sintoma com porcentagem alta em pessoas avaliadas como sim para COVID-19, é problema respiratório, com 76.86% das pessoas que tiveram a doença, apresentaram esse sintoma.

6. CONSIDERAÇÕES FINAIS

Com este trabalho foi possível compreender um pouco sobre as etapas e os conceitos da mineração de dados, destacando-se a utilização dos algoritmos de classificação e a forma como cada um se comporta melhor. Também observou-se que existe uma quantidade relevante de bases de dados em relação ao COVID-19.

Como verificado no decorrer desta pesquisa, um mesmo algoritmo pode ter um bom desempenho para algum indicador é ruim para outro. Isto justifica um aprofundamento de estudos sobre o tema em estudo e uma investigação mais ampla. Com a análise de desempenho dos algoritmos pode-se notar que um algoritmo em específico se destacou negativamente, o *OneR*. O *OneR* é um algoritmo que tem uma indução que busca encontrar uma única regra que melhor representa o *dataset*, utilizando apenas uma característica. Para determinar a melhor regra são computadas as ocorrências dos valores mais frequentes da classe alvo para cada valor de um atributo. Em relação a base escolhida para a pesquisa, a estratégia do *OneR* acaba sendo inadequada e tendo um desempenho pior que os algoritmos.

Ademais, o uso de base de dados associadas à pandemia do COVID-19 mostra-se uma relevante linha a ser pesquisada, investigando o desempenho dos principais algoritmos de classificação e outros tipos de algoritmos disponíveis na área de mineração de dados. Para trabalhos futuros, novos algoritmos podem ser explorados, tanto quanto outras técnicas de mineração de dados, outros indicadores e também outras bases de dados relevantes para o cenário.

Referências

[1] A. M. de Macedo Júnior, "Covid-19: calamidade pública," 2020. [Online]. Available: http://doi.org/10.6008/CBPC2674-6484.2020.001.0001

[2] X. A. I. J.-H. J. J.-G. H. ACharlyn Nayve Villavicencio, Julio Jerison Escudero Macrohon, "Covid-19 prediction applying supervised machine learning algorithms with comparative analysis using weka," 2021. [Online]. Available: https://doi.org/10. 3390/a1407020

[3] IBGE, "O ibge apoiando o combate `a covid-19." [Online]. Available: https: //covid19.ibge.gov.br/pnad-covid/

[4] R. C. d. Carvalho, "Aplicação de mineração de dados em informações oriundas de prontuários de paciente," Informação em Pauta, vol. 3, no. especial, pp. 161–181, nov. 2018. [Online]. Available: http://www.periodicos.ufc.br/informacaoempauta/ article/view/39723

[5] E. Santos, "Uma aplicação móvel para classificação de imagens utilizando deep learning: Um estudo de caso sobre doenças pulmonares," 2021.

[6] M. Brito, "Aspectos teóricos da mineração de dados e aplicação das regras de classificação para apoiar o comércio," 2012.

[7] U. of Waikato, "Machine learning software in java," 1993. [Online]. Available: https://www.cs.waikato.ac.nz/ml/weka/index.html

[8] H. d. F. M. Noemi Dreyer Galvão, "Técnica de mineração da- dos: uma revisão̊ ao da literatura," 2009. [Online]. Available: https: //doi.org/10.1590/S0103-21002009000500014

[9] P. U. M U Fayyad, Shapiro Piatetsky, "Advances in knowledge discovery and data mi- ning," 1996.

[10] S. S. U. S. I. A. L. J. Muhammad, Md. Milon Islam, "Predictive data mining models for novel coronavirus (covid-19) infected patients' recovery," 2020.

[11] R. K. Kohler, "Classificação de agregados de rochas ígneas quanto à sua alteração por meio de processamento digital de imagens," 2021.

[12] G. N. L Taylor, "Improving deep learning with generic data augmentation. in: Ieee. 2018 ieee symposium series on computational intelligence (ssci)," 2018.

[13] T. M. K. V. C. F. G. L Y Sato, Y E Shimabukuro, "Análise comparativa de algoritmos de árvore de decisão do sistema weka para classificação do uso e cobertura da terra," 2013.

[14] I. B. d. N. S. L. D. d. B. L. R. M. d. M. Ingrid Rafaella dos Santos Melo, Natasha Seleidy Ramos de Medeiros, "Avaliação do desempenho do algoritmo jrip na classificação do diagnóstico de doenças cardíacas," 2019.

[15] "Periódicos capes." [Online]. Available: https://www-periodicos-capes-gov-br.ezl. periodicos.capes.gov.br/index.php?

[16] H. Hari, "Symptoms and covid presence," 2020.

[17] F. Clésio, "Análise de concordância - métrica de kappa," 2012.

[Online]. Available: https://mineracaodedados.wordpress.com/2012/09/22/

analise-de-concordancia-metrica-de-kappa/

[18] ——, "Matriz de confusão - métricas de avaliação de modelos de

classificação/predição," 2014. [Online]. Available: https://mineracaodedados.

wordpress.com/tag/matriz-de-confusao/

[19] J. Demsar, "Statistical comparisons of classifiers over multiple data set," 2006.

[20] J. A. Baranauskas, "Métodos de amostragem e avaliação de algoritmos." [Online]. Availa-

ble: https://dcm.ffclrp.usp.br/~augusto/teaching/ami/AM-I-Metodos-Amostragem.

pdf

A. Apêndice

A tabela 23 mostra os resultados da execução do teste Friedman realizando comparações par a par, entre os indicadores dos algoritmos de 5 execuções diferentes feito na linguagem R utilizando o RStudio.

Algoritmos	Sensibilidade	Acurácia	Especificidade	Eficiência	Valor Pred. +	Valor Pred. -	PHI(matthews)
p-value	1.648e-05	0.0000944	0.0004394	4.477e-05	0.0003544	1.511e-05	6.377e-06
J48CV ~ J48PS	Falso	Falso	Falso	Falso	Falso	Falso	Falso
J48CV ~ JRipCV	Falso	Falso	Falso	Falso	Falso	Falso	Falso
J48CV ~ JRipPS	Falso	Falso	Falso	Falso	Falso	Falso	Falso
J48CV ~ REPTreeCV	Falso	Falso	Falso	Falso	Falso	Falso	Falso
J48CV ~ REPTreePS	Falso	Falso	Falso	Falso	Falso	Falso	Falso
J48CV ~ RandomForestCV	Falso	Falso	Falso	Falso	Falso	Falso	Falso
J48CV~ RandomForestPS	Falso	Falso	Falso	Falso	Falso	Falso	Falso
J48CV ~ OneRCV	Verdadeiro	Falso	Verdadeiro	Verdadeiro	Verdadeiro	Falso	Verdadeiro
J48CV ~ OneRPS	Verdadeiro	Falso	Verdadeiro	Verdadeiro	Falso	Verdadeiro	Verdadeiro
J48PS ~ JRipCV	Falso	Falso	Falso	Falso	Falso	Falso	Falso
J48PS ~ JRipPS	Falso	Falso	Falso	Falso	Falso	Falso	Falso
J48PS ~ REPTreeCV	Falso	Falso	Falso	Falso	Falso	Falso	Falso
J48PS ~ REPTreePS	Falso	Falso	Falso	Falso	Falso	Falso	Falso
J48PS ~ RandomForestCV	Falso	Falso	Falso	Falso	Falso	Falso	Falso
J48PS~ RandomForestPS	Falso	Falso	Falso	Falso	Falso	Falso	Falso
J48PS ~ OneRCV	Falso	Falso	Falso	Falso	Verdadeiro	Falso	Falso
J48PS ~ OneRPS	Falso	Falso	Verdadeiro	Falso	Verdadeiro	Falso	Falso
JRipCV ~ JRipPS	Falso	Falso	Falso	Falso	Falso	Falso	Falso
JRipCV ~ REPTreeCV	Falso	Falso	Falso	Falso	Falso	Falso	Falso
JRipCV ~ REPTreePS	Falso	Falso	Falso	Falso	Falso	Falso	Falso
JRipCV ~ RandomForestCV	Falso	Falso	Falso	Falso	Falso	Falso	Falso
JRipCV ~ RandomForestPS	Falso	Falso	Falso	Falso	Falso	Falso	Falso
JRipCV ~ OneRCV	Verdadeiro	Falso	Falso	Falso	Falso	Falso	Falso
JRipCV ~ OneRPS	Falso	Falso	Falso	Falso	Falso	Verdadeiro	Falso
JRipPS ~ REPTreeCV	Falso	Falso	Falso	Falso	Falso	Falso	Falso
JRipPS ~ REPTreePS	Falso	Falso	Falso	Falso	Falso	Falso	Falso
JRipPS ~ RandomForestCV	Falso	Falso	Falso	Falso	Falso	Falso	Falso
JRipPS ~ RandomForestPS	Falso	Falso	Falso	Falso	Falso	Falso	Falso
JRipPS ~OneRCV	Falso	Falso	Falso	Falso	Falso	Falso	Falso
JRipPS ~OneRPS	Falso	Falso	Falso	Falso	Falso	Falso	Falso
REPTreeCV ~REPTreePS	Falso	Falso	Falso	Falso	Falso	Falso	Falso
REPTreeCV ~RandomForestCV	Falso	Falso	Falso	Falso	Falso	Falso	Falso
REPTreeCV ~RandomForestPS	Falso	Falso	Falso	Falso	Falso	Falso	Falso
REPTreeCV ~OneRCV	Falso	Falso	Falso	Falso	Falso	Falso	Falso
REPTreeCV ~OneRPS	Falso	Falso	Falso	Falso	Falso	Falso	Falso
REPTreePS ~RandomForestCV	Falso	Falso	Falso	Falso	Falso	Falso	Falso
REPTreePS ~RandomForestPS	Falso	Falso	Falso	Falso	Falso	Falso	Falso
REPTreePS ~OneRCV	Falso	Falso	Falso	Falso	Falso	Falso	Falso
REPTreePS ~OneRPS	Falso	Falso	Falso	Falso	Falso	Falso	Falso
RandomForestCV ~RandomForestPS	Falso	Falso	Falso	Falso	Falso	Falso	Falso
RandomForestCV ~OneRCV	Verdadeiro	Verdadeiro	Falso	Verdadeiro	Falso	Verdadeiro	Verdadeiro
RandomForestCV ~OnePS	Verdadeiro	Verdadeiro	Falso	Verdadeiro	Falso	Verdadeiro	Verdadeiro
RandomForestPS ~OneRCV	Verdadeiro	Verdadeiro	Falso	Falso	Falso	Verdadeiro	Verdadeiro
RandomForestPS ~OneRPS	Verdadeiro	Verdadeiro	Falso	Falso	Falso	Verdadeiro	Verdadeiro
OneRCV ~OneRPS	Falso	Falso	Falso	Falso	Falso	Falso	Falso

B. Apêndice

Os gráficos de boxplot apresentados nas figuras de 3 a 9, trazem o desempenho dos algoritmos em uma escala de 0 á 100 \% para cada indicador e uma indicação de 0 a 10, que refere-se a cada algoritmo avaliado. Sendo 1, o algoritmo J48CV, 2 J48PS, 3 JRipCV, 4 JRipPS, 5 *REPTreeCV*, 6 *REPTreePS*, 7 *RandomForestCV*, 8 *RandomForestPS*, 9 *OneRCV*, 10 *OneRPS*.

Figura 3. Plot Acurácia

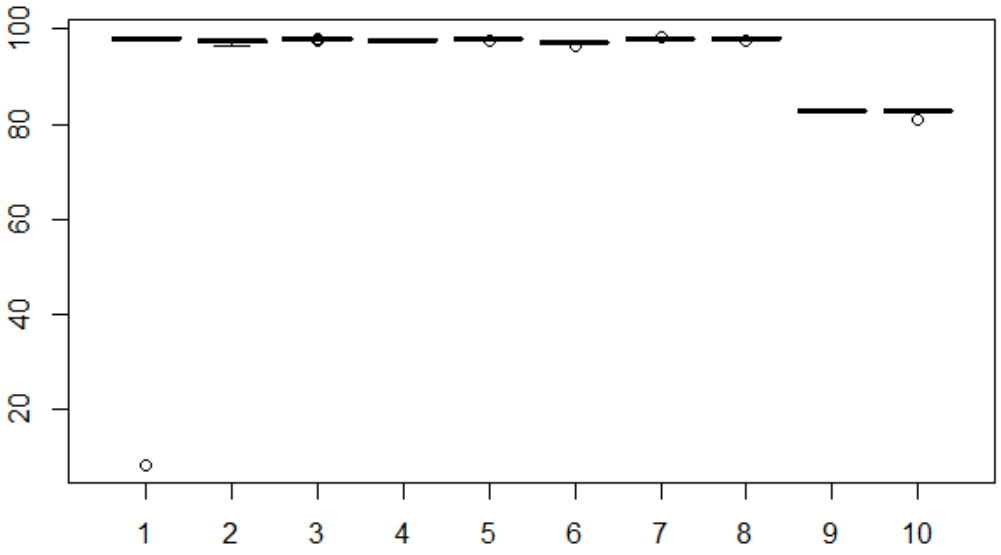


Figura 4. Plot Eficiência

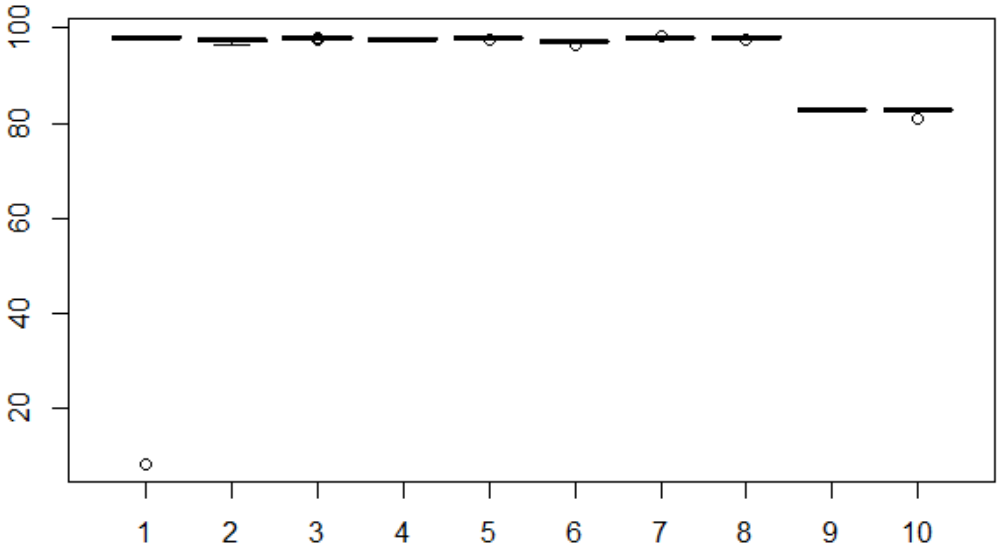


Figura 5. Plot Especificidade

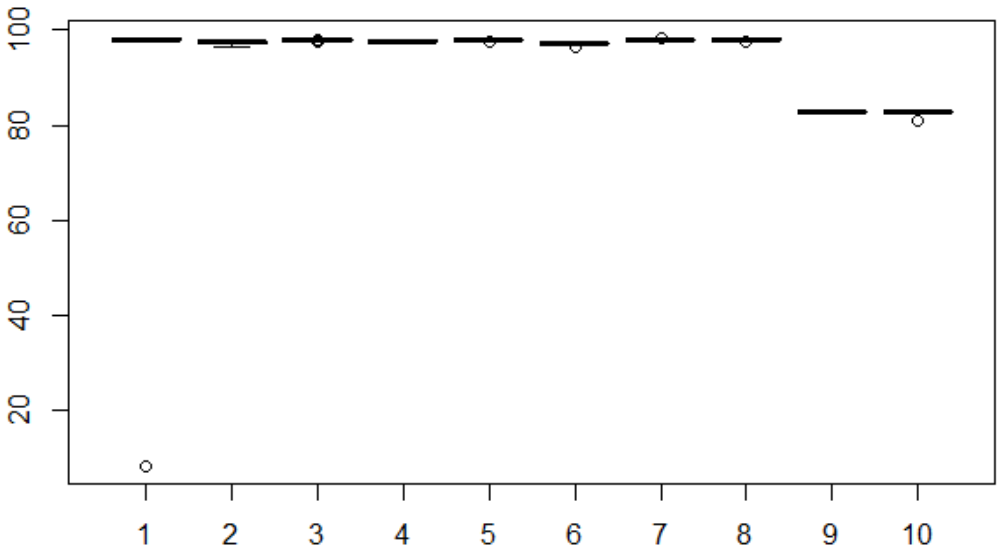


Figura 6. Plot PHI

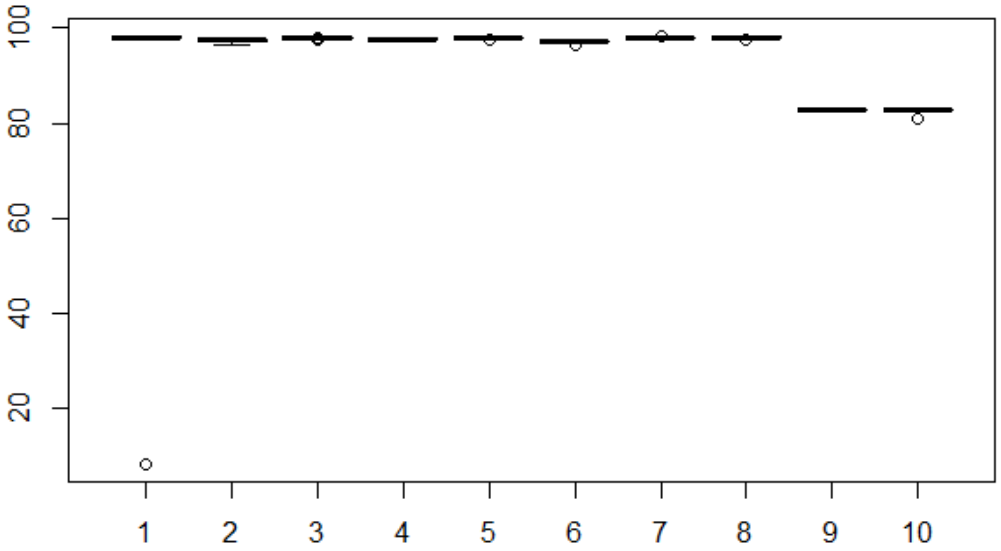


Figura 7. Plot Preditivo Negativo

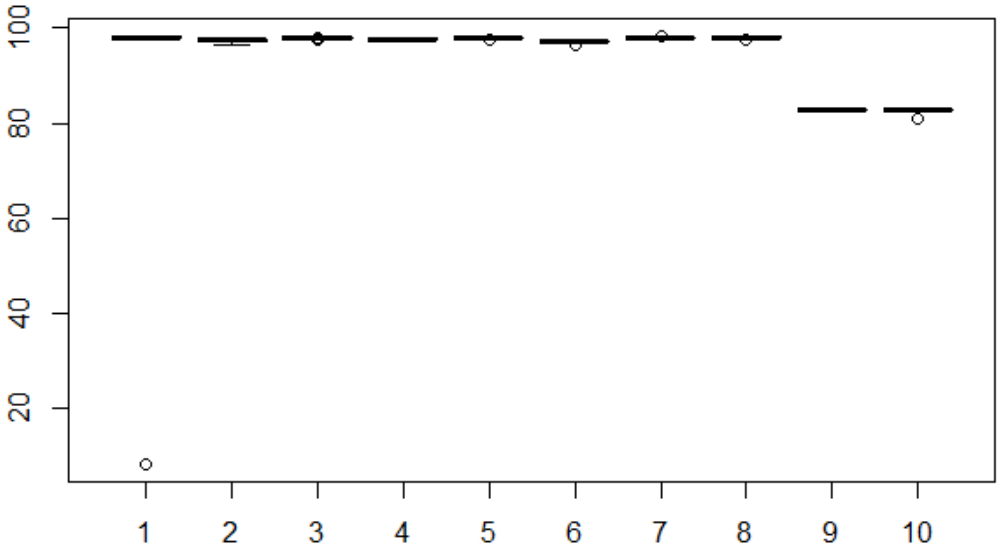


Figura 8. Plot Preditivo Positivo

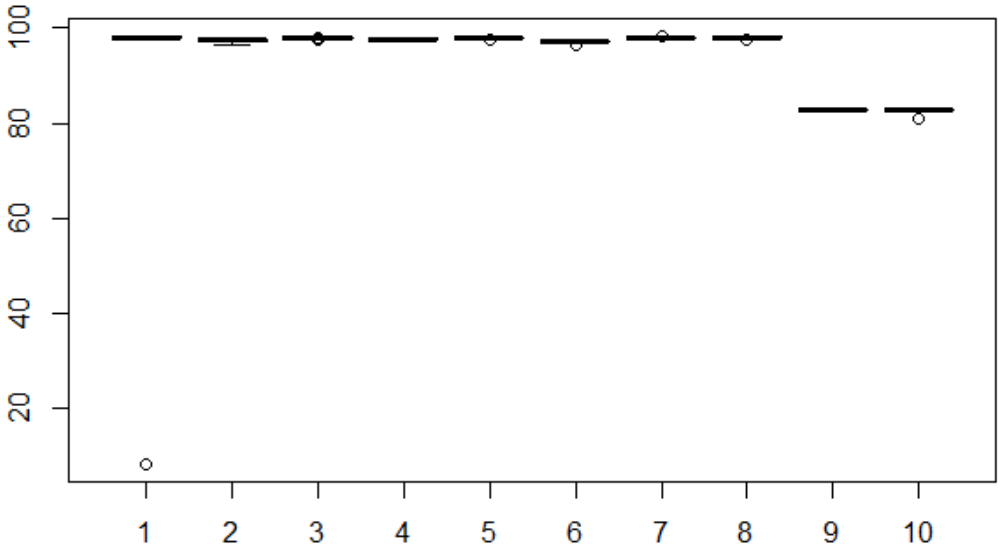


Figura 9. Plot Sensibilidade