Estudo de caso Sobre Otimização de Máquinas Industriais uma Abordagem Comparativa dos Modelos de Árvore de Decisão, Rede Neural e Regressão Logística

Bruno Henrique Parente de Carvalho[1], **João Victor Cardoso Palheta**[2], **José de Sousa Ribeiro Filho**[3]

[1]Instituto Federal de Educação, Ciência e Tecnologia do Estado do Pará

brunoparente22@gmail.com, victorpalheta2@gmail.com,

jose.ribeiro@ifpa.edu.br

Resumo. *O artigo aborda a otimização de máquinas industriais por meio da comparação de três modelos de aprendizado de máquina: árvore de decisão, rede neural e regressão logística. O estudo se fundamenta na análise de um dataset denominado "Otimization of Machine Downtime", disponível no Kaggle, que contém 2500 instâncias e 16 atributos, sendo 3 categóricos e 13 numéricos. O objetivo principal é identificar falhas durante o funcionamento das máquinas e fornecer insights para os gestores sobre prevenções corretivas. A pesquisa segue uma metodologia de pré-processamento de dados, que inclui tratamento de valores ausentes, conversão de tipos de dados, detecção e remoção de outliers, além de codificação de variáveis categóricas. Após a limpeza e preparação dos dados, os modelos são treinados e testados, utilizando 60% dos dados para treino e 40% para teste. Os resultados mostram que a árvore de decisão teve o melhor desempenho, com uma acurácia de 97,3%, seguida pela rede neural com 96,8% e pela regressão logística com 84,9%. As análises estatísticas indicaram variabilidades nas pressões e temperaturas das máquinas, ressaltando a importância da monitoração das condições operacionais. Os insights obtidos são cruciais para a implementação de modelos preditivos para problemas de classificação binária.*

1. Introdução

A tecnologia voltada às indústrias surgiu no século XIX na Inglaterra com as máquinas à vapor voltadas ao setor têxtil, mais adiante surgiram as máquinas que se utilizavam de eletricidade, veículos, moinhos e maquinário agrícola [Marson 2017]. Todas essas tecnologias são complexas e envolvem partes móveis, aquecimento e rotação, além de estarem geralmente expostas ao clima, impactos e situações adversas que perduram até hoje, fazendo com que ocorram falhas, desligamentos e mau funcionamento das mesmas.

Diante disso, o presente trabalho se propõe a utilizar um *dataset*, coleção de dados estruturados, por meio da linguagem Python de programação, acerca da otimização de máquinas industriais em um problema de classificação, buscando identificar se houve ou não falha durante o período de funcionamento, pois com os dados coletados é possível identificar o atributo que pode vir a causar futuras falhas, além de auxiliar os gestores técnicos de indústrias com a indicação de prevenções corretivas e outras ações.

Por fim, o objetivo desta pesquisa é realizar uma análise comparativa simplificada entre três modelos de aprendizado de máquina de diferentes níveis de complexidade, um baseado em árvore de decisão, outro em rede neural e o último em regressão logística, aplicados ao contexto de máquinas industriais, visando avaliar e comparar o desempenho dos modelos com base em medidas clássicas de performance com a finalidade de identificar qual o melhor modelo para predição de falhas do maquinário.

2. Metodologia

2.1. Dataset

O *dataset* utilizado nessa pesquisa chama-se *Otimization of Machine Downtime* e pode ser encontrado no site Kaggle; famoso por ser um repositório de *datasets*, *notebooks* e organizar competições sobre ciência de dados e aprendizagem de máquina.

Detalhando nossa coleção de dados vemos que lidamos com um problema estatístico conhecido como 'Problema de Classificação' que busca enquadrar cada máquina análisada em uma das duas classes 'fins' existentes, falha ou sem falha. Para isso, buscamos utilizar três modelos de aprendizagem de máquina: Árvore de decisão, Redes Neurais e Regressão Logística.

Estruturalmente o *dataset* é composto por 2500 instâncias e 16 atributos, dos quais 3 são categóricos e 13 são numéricos contínuos. O atributo que exprime a coluna "objetivo" do dataset é o atributo "Downtime", o qual estabelece se houve falha ou não.

2.2. Pré-processamento de Dados

Ao utilizar um *dataset* é comum que se encontrem dados desorganizados, duplicados, faltantes e de uma maneira geral, poluídos. Diante disso, é necessário realizar uma série de tratamentos nos dados, como visto abaixo, para que a análise seja o mais precisa possível.

2.2.1. Tratamento de valores ausentes

Em dados extraídos de cenários como o apresentado nesta pesquisa − cenários instáveis − é comum que os aparelhos registradores falhem ou que algum dado seja perdido durante as muitas etapas de transformação. Desta forma, quando visualizamos nossa coleção de dados, teremos algumas lacunas com informações faltantes que tratamos utilizando uma função da biblioteca 'pandas' chamada 'fillna', responsável por localizar essas células sem dados e preenchê-las com o parâmetro especificado, neste caso -1, pois como os dados são de medidas (e não podem existir medidas negativas) serão facilmente identificáveis. Isso é positívo pois conseguimos completar os dados para executar os demais passos, porém desta maneira criamos um problema abordado na seção 2.2.3.

2.2.2. Conversão de tipos de dados

Os arquivos que carregam os *datasets* geralmente são 'Comma Separated Values' (Valores Separados por Virgulas − CSVs) nos quais todos os dados estão em formato de texto, porém ao utilizá-los para realizar análises de dados e treinamentos de modelos

necessitamos que cada dado seja representado com o seu respectivo tipo. Tendo isso em vista, utilizamos a função 'convert_dtypes' presente na biblioteca 'pandas' responsável por identificar automaticamente o tipo de dado mais adequado para cada atributo do *dataset*.

2.2.3. Detecção e remoção de outliers

"Outliers" ou pontos fora da curva representam alguns dados muito discrepantes dos demais e considerados fora do padrão, podendo interferir em algumas métricas sensíveis como os desvios, variância e média, porém eles ainda assim podem representar valores reais e corretos a depender da natureza dos dados e são calculados utilizando a seguinte fórmula [Kwak and Kim 2017]:
Para os outliers superiores:

$$\mathrm{Q}_3 + 1,5 \cdot \mathrm{A}_Q$$

Para os outliers inferiores:

$$\mathrm{Q}_1 - 1,5 \cdot \mathrm{A}_Q$$

Onde:

Q_1 = Quartil 1.

Q_3 = Quartil 3.

A_Q = Amplitude inter-quartis.

Como citado na seção 2.2.1, ao tratar os dados ausentes como -1 acabamos criando diversos outliers em alguns atributos, por isso ao realizar algumas ações, como gerar gráficos, devemos desconsiderá-los para fins de fidelidade aos dados originais.

2.2.4. Codificação de variáveis categóricas

Durante a fase de tratamento de dados é comum que os dados categóricos sejam transformados em rótulos numéricos, os quais são melhor utilizados pelas máquinas, oferecendo uma melhora de desempenho e aprimorando os modelos. Utilizamos a técnica conhecida como *Label Encoding*. Essa técnica busca transformar categorias textuais em códigos, ou seja, ao invés de usarmos nomes para classificar as máquinas em estudo, usaremos números inteiros positivos.

2.2.5. Transformação de dados

As transformações de dados são bem vindas quando um atributo além de exprimir valor numérico, também contém significado social, como datas, as quais nesta pesquisa foram transformadas de valores textuais para o seu correspondente numérico conhecido como *Timestamp*.

Outra transformação importante dos dados é normalizalos, ou seja, deixá-los entre 0 e 1 através de uma função de "Scaler" chamada "MinMaxScaler".

2.2.6. Remoção de atributos irrelevantes ou redundantes

Quando estamos estudando conjuntos de dados é comum calcular a correlação entre os atributos. Isso consiste em calcular a afinidade entre as variáveis e o quanto elas estão relacionadas. Quando duas variáveis atingem o índice 1 podemos inferir que seus dados demonstram a mesma informação, gerando redundância desnecessária, como é o caso dos atributos: "Machine_ID" e "Assembly_Line_No". Diante disso optamos por remover a primeira coluna como visto nas figuras 1 e 2, já que a informação de localização das máquinas não se perde e ganhamos desempenho no modelo.

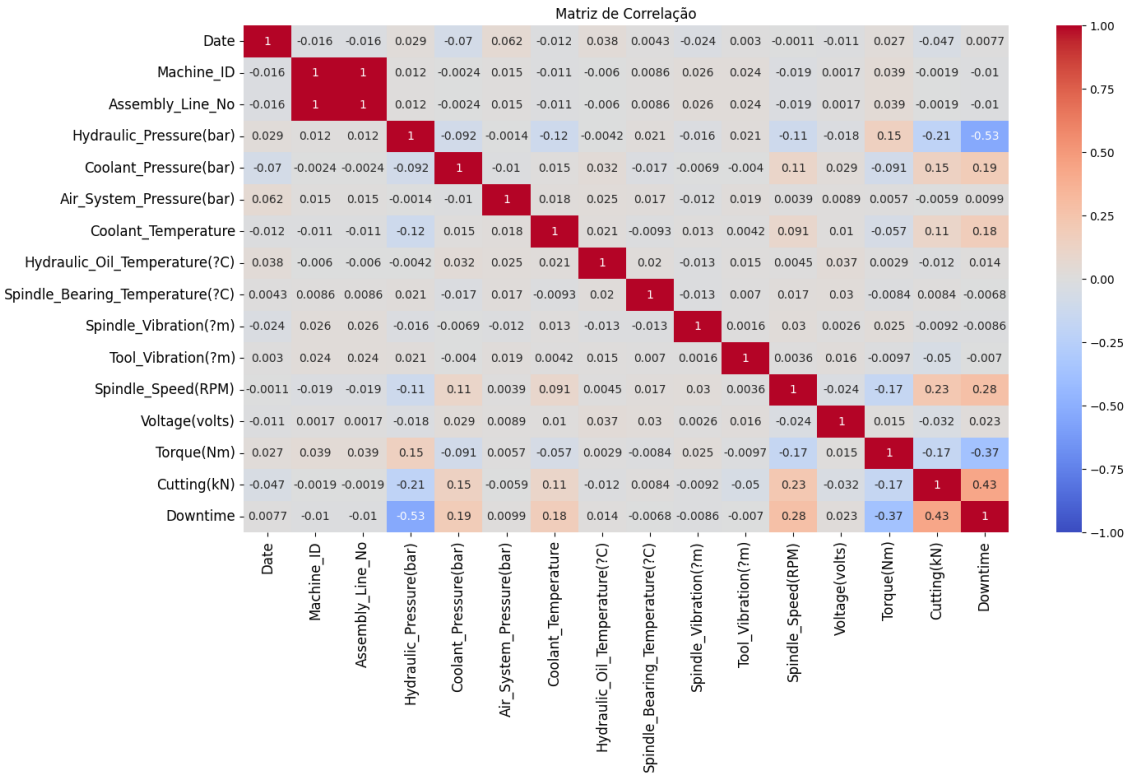

Figura 1. Matriz de correlação antes da exclusão do atributo Machine_ID

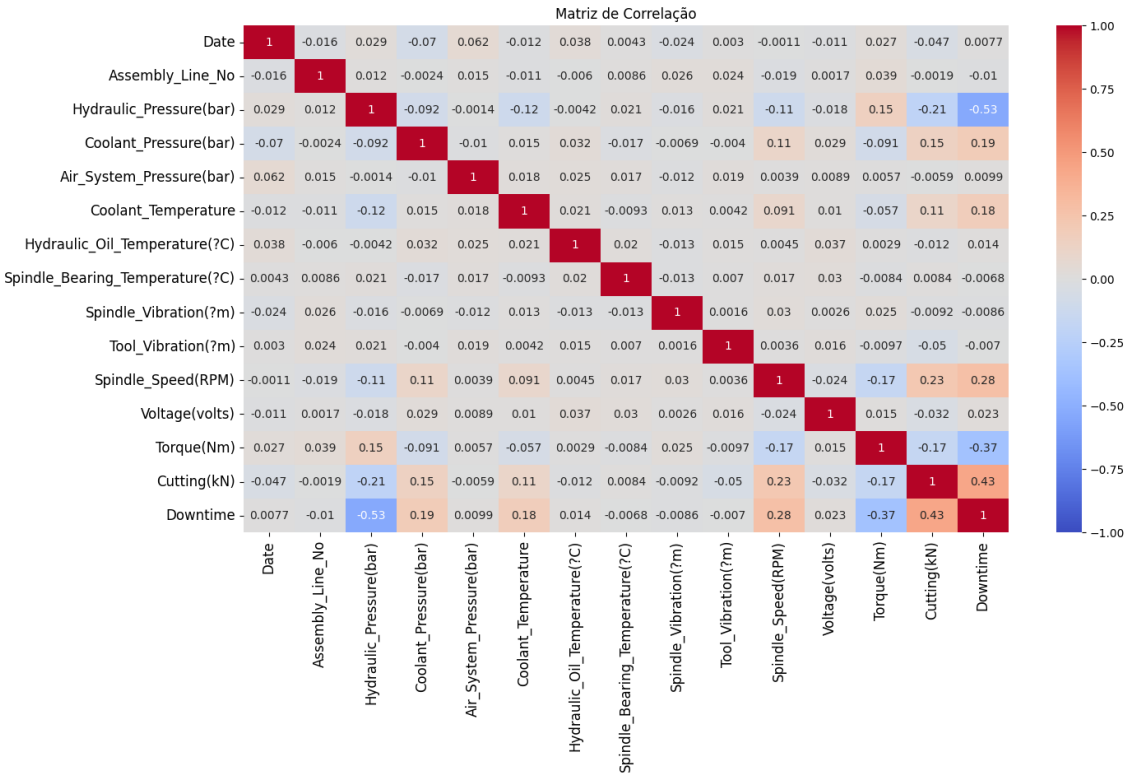

Figura 2. Matriz de correlação após a exclusão do atributo Machine_ID

2.3. Análise Exploratória

Depois de processar todos os atributos do dataset podemos começar a explorá-los para extrair informações, para isso usamos algumas estimativas de localidade, variabilidade e gráficos, os quais serão detalhados das seções a seguir.

2.3.1. Estimativas de localidade

As estimativas de localidade traduzem a distribuição de dados, a concentração, e as tendências centrais [Guedes et al. 2005]. Nesta pesquisa foram utilizadas as seguintes estimativas:

1. Média: Utilizada para descobrir o valor médio dos dados (valor representativo), o ponto de equilíbrio, além de ser utilizada para o cálculo de outros parâmetros;
2. Moda: Utilizada para descobrir a instância com maior frequência dentre todos os dados;
3. Mediana: Utilizada para identificar o elemento central da distribuição e dividir o conjunto em duas partes iguais, uma superior e outra inferior ao valor central;
4. Separatrizes (percentis e quartis): Medidas utilizadas para separar posicionalmente o conjunto de dados em intervalos iguais, são utilizadas para calcular outro parâmetros e para nos descrever como os dados estão organizados espacialmente.

2.3.2. Estimativas de variabilidade

As estimativas de variabilidade buscam, de maneira geral, auxiliar as medidas de localidade e indicam a proximidade dos dados ou seu espalhamento e as utilizadas foram:

1. Variância: É uma medida que indica o quão distante da média está cada atributo;
2. Desvio Padrão: Medida que indica o quão dispersos os dados estão, ou seja, o quão uniforme são os dados.

Essas medidas foram utilizadas, em grande parte, para entender a natureza dos dados.

2.3.3. Gráficos

Os gráficos utilizados durante a pesquisa foram diversos, dentre eles: boxplots para ter uma visão geral do conjunto de dados; histogramas para visualizar a distribuição de um determinado atributo, matrizes de correlação para verificar o grau de afinidade das variáveis, árvores de decisão (visão gráfica) e matrizes de confusão para verificar o grau de acertabilidade dos modelos construídos.

2.4. Modelagem

Esta seção se dedica a explicar o funcionamento dos algorítmos utilizados e suas implementações, além de detalhar o método de treinamento e as bibliotecas utilizadas.

Um algorítmo computacional consiste, via de regra, em receber um dado, processá-lo e gerar uma saída. Isso se aplica aos algorítmos de árvore de decisão, rede neural e regressão logística utilizados nesta pesquisa.

2.4.1. Árvore de decisão

Uma árvore de decisão constitui uma estrutura hierárquica que descreve um conjunto de caminhos possíveis para a tomada de decisão, bem como os resultados ou consequências associadas a cada um desses caminhos. Essa estrutura acaba afunilando os dados por meio de perguntas que geram conhecimento através da entropia, fazendo com que na folha (terminal responsável por classificar um rótulo) tenhamos uma resposta.

A entropia, por sua vez, nos ajuda a determinar a "Quantidade de informação" que ganharemos ao passar por um nó da árvore. Essa medida funciona de forma que se os dados estiverem bem distribuídos entre as classes a quantidade de incerteza é muito grande e a entropia é alta, já se os dados estiverem concentrados em uma classe a quantidade de incerteza e a entropia são baixas, tornando mais fácil para o modelo classificar um determinado dado. As árvores se aproveitam dessa propriedade para subdividir os dados em partições cada vez menores até ter um grau de certeza suficiente para indicar uma resposta.

O algorítmo utilizado para a implementação de nossa árvore foi o ID3, que consiste na seguinte sequência de passos:

- Identificar o atributo com menor entropia (Maior partição de dados) para ser a raíz da árvore;

- Utilizar os atributos seguintes para repetir o processo e assim, particionar cada vez mais os dados;
- Ao chegar ao fim da lista de atributos, criar nós folhas para prever o rótulo mais comum.

2.4.2. Rede Neural

Redes Neurais artificiais são algorítmos preditivos bio-inspirados, ou seja, são baseados no modo de operação do cérebro (redes neurais naturais). O cérebro humano pode ser entendido como um conjunto de neurônios conectados que recebem sinais, processam e então passam o resultado desse processamento para outros neurônios. Redes neurais artificiais imitaram esse funcionamento por meio da estrutura conhecida como "perceptron". Esse modelo é conhecido por ser "blackbox" ou caixa preta, isso significa que a forma como o problema está sendo resolvido não é 100% transparente.

A estrutura perceptron é uma rede neural artificial que aproxima-se de 1 neurônio e quando conectadas, criam redes cada vez maiores e capazes de resolver problemas mais complexos. Ao conectar essas redes de perceptrons criamos"layers" (camadas). Quando essas camadas são criadas é necessário também que seja estabelecida uma ordem de processamento, sempre da camada de "input" para a camada de "output". Esse método é conhecido como "feed-forward", onde os neurônios repassam as informações através das camadas até atingir o final do processamento.

Para treinar esse algoritmo é utilizada uma técnica chamada de "backpropagation", a qual consiste em realizar uma calibragem automática da rede neural com base em gradiente descendente, o que faz com que o modelo seja ajustado dinamicamente e assim, aprenda mais.

A forma mais comum de implementar uma rede neural é utilizando listas de layers com conjuntos de dados para treinar o modelo utilizando backpropagation e ajustando os pesos a partir disso.

2.4.3. Regressão logística

A regressão logística é um modelo estatístico, cujo objetivo é estimar a probabilidade de uma observação pertencer a uma classe específica. Em vez de prever um valor contínuo, ela mapeia uma combinação linear das variáveis independentes para um intervalo entre 0 e 1 utilizando a função logística (ou sigmoide). Essa característica a torna especialmente adequada para tarefas onde a saída representa uma probabilidade ou uma chance de sucesso.

Esse algorítmo é desenvolvido aplicando a função sigmoide aos dados e decidindo a classe com base em sua proximidade com os valores 0 e 1.

2.4.4. Bibliotecas

Todos os algoritmos citados nas seções 2.4.1, 2.4.2 e 2.4.3 foram desenvolvidos utilizando *Python*, de maneira direta e indireta, utilizando as seguintes bibliotecas:

- Numpy [Harris et al. 2020];
- Pandas [McKinney et al. 2011];
- Seaborn [Waskom 2021];
- Matplotlib [Hunter 2007];
- Sklearn [Pedregosa et al. 2011].

2.4.5. Treino e teste

Quando estamos lidando com modelos de aprendizagem de máquina precisamos utilizar nossos dados para treinar o modelo, porém parte desse conjunto precisa funcionar como teste, ou seja, uma espécie de "Prova" para entender como está a assertividade do modelo. Essa divisão é conhecida como "Split de treino e teste", o qual consiste em 60% dos dados serem destinados ao treino e 40% destinado ao teste.

2.5. Métricas de Avaliação

As métricas de avaliação são métodos para determinar o quanto um determinado modelo está acertando ou errando e assim avaliá-lo [Strauss et al. 2022]. Durante essa pesquisa foram utilizadas 3 métricas para avaliar os modelos: Acurácia, Matriz de confusão e precisão.

- Acurácia: É a razão entre a quantidade de previsões corretas e a quantidade total de previsões.
- Matriz de confusão: Uma matriz 2x2 utilizada para reconhecer os falsos positivos, falsos negativos, verdadeiros positivos e verdadeiros negativos.
- Precisão: Comparação entre a quantidade de verdadeiros positivos e a quantidade de previsões positivas que o modelo realizou.

2.6. Reprodutibilidade

Esta pesquisa possui um repositório no github onde pode ser encontrado tanto o *dataset* utilizado quanto o notebook do google colab utilizado para realizar as etapas de carregamento do conjunto de dados, tratamento e análise.

- Link

3. Resultados e Discussão

Os principais resultados desta pesquisa serão divididos em seções, pois cada parte do tratamento dos dados pode gerar insights valiosos. Portanto, as seções a seguir apresentam os resultados, não só dos modelos, mas também da análise dos dados.

3.1. Análise Estatística

O *Dataset* possui 2500 instâncias de resgistros de várias informações acerca de máquinas industriais.

O dataset apresenta atributos com diferentes níveis de variabilidade e padrões. A **pressão hidráulica** (*Hydraulic_Pressure*) mostra alta variabilidade e presença de outliers, indicando flutuações significativas. A **pressão do líquido refrigerante** (*Coolant_Pressure*) e a **pressão do sistema de ar** (*Air_System_Pressure*) possuem baixa variabilidade, mas com alguns outliers, mostrando relativa constância dentro de limites aceitáveis.

As **temperaturas** (*Coolant_Temperature*, *Hydraulic_Oil_Temperature* e *Spindle_Bearing_Temperature*) apresentam alta dispersão e alguns outliers, evidenciando condições operacionais variáveis. Já as **vibrações do eixo e da ferramenta** (*Spindle_Vibration* e *Tool_Vibration*) variam em uniformidade: o eixo tem menor dispersão, enquanto a ferramenta apresenta alta variabilidade.

Por fim, o **torque** (*Torque*) e a **voltagem** (*Voltage*) possuem alta variância e outliers, sugerindo instabilidade nas medições, enquanto o **corte** (*Cutting*) apresenta baixa variância, indicando boa consistência operacional. Esses insights são valiosos em momentos de decidir sobre análises corretivas nestas maáquinas e podem evitar acidentes.

3.2. Análise dos Outliers

Como citado na seção 2.2.3 alguns outliers representam dados importantes, como é o caso dos outliers presentes no atributo "Voltage", Figura 3, onde os outliers para a voltagem nos mostra que em alguns momentos as maquinas operam com uma carga muito acima da média, reforçando a necessidade de uma vistoria nas instalações elétricas das fábricas ou uma manutenção preventiva.

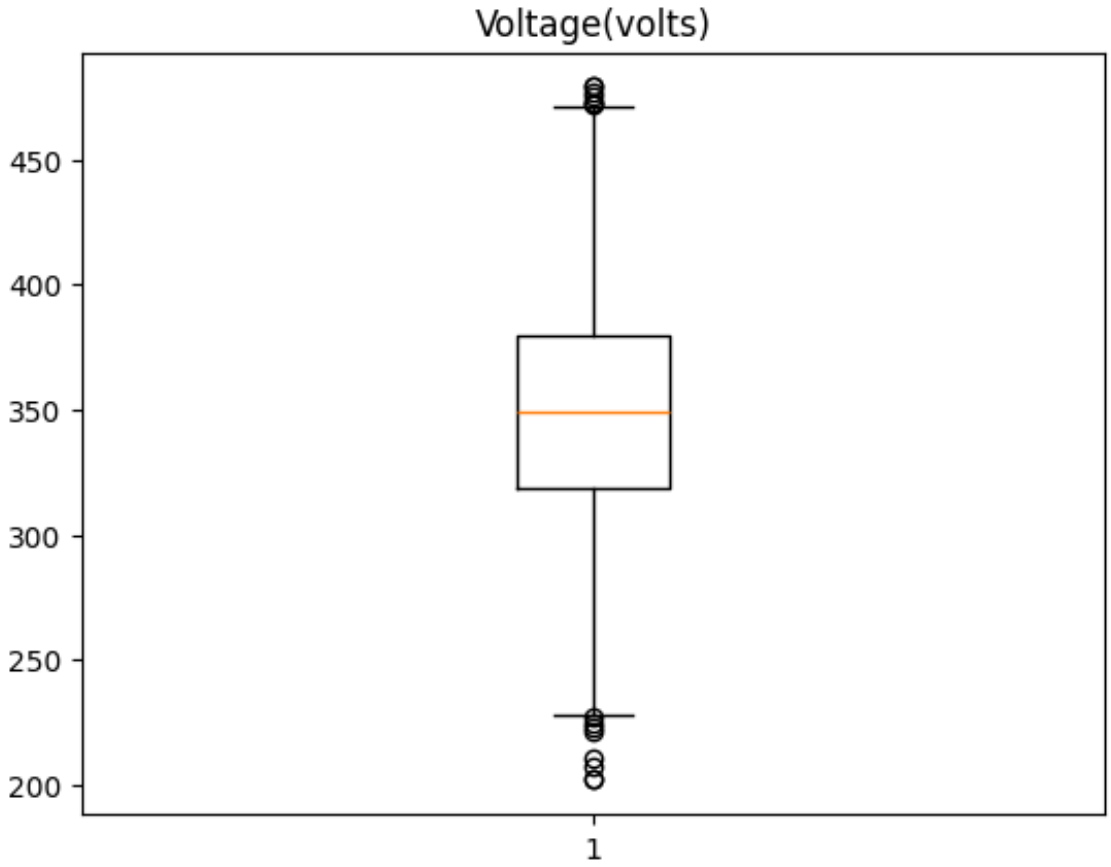

Figura 3. Box-Plot demonstrando os outliers

3.3. Análise da Árvore de Decisão

A árvore de decisão, diferentemente dos outros dois modelos, possui uma visualização gráfica presente na imagem 4, a qual nos mostra como o modelo conver-

giu para uma resposta. Podemos observar que o atributo mais significativo para separar os dados é o “Hydraulic_Pressure”.

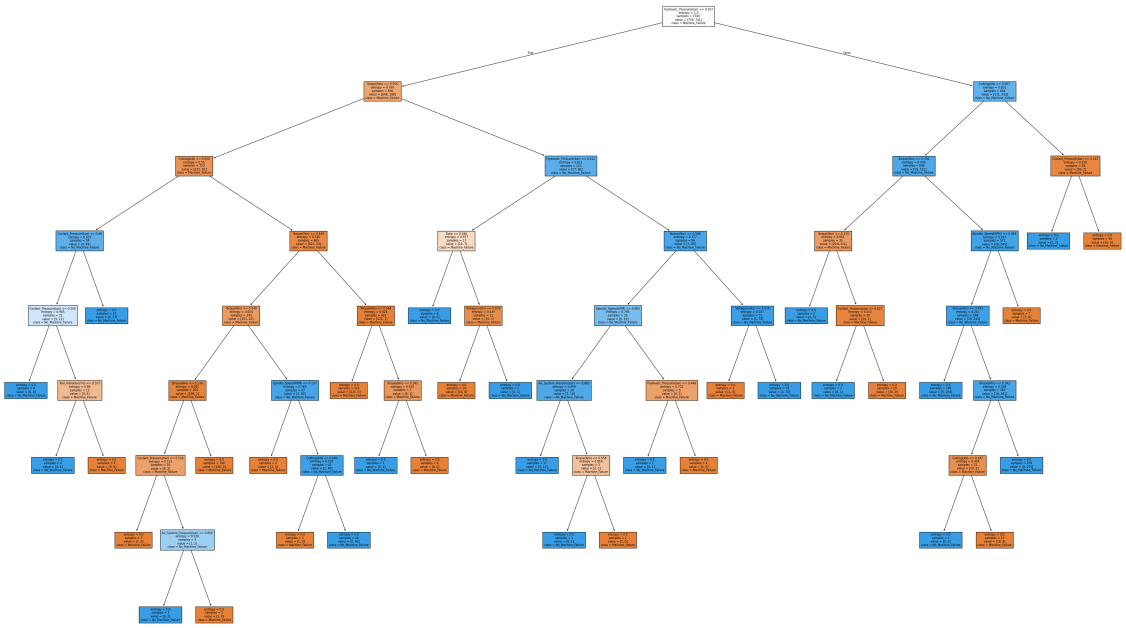

Figura 4. Árvore de decisão gerada durante a pesquisa

3.4. Comparação entre modelos

Dentre os 3 modelos apresentados; Árvore de decisão, Redes Neurais e Regressão logística; o que obteve o melhor desempenho foi o primeiro, como demonstrado na figura 5, com uma acurácia de 0.973, seguido pelo modelo de redes neurais, com 0.968 e por fim o modelo de regressão logística com 0.849. Isso se dá, provavelmente, devido à natureza dos dados e a forma como a árvore separou os atributos por classes, demonstrando que um algorítmo simples, quando bem executado em dados estruturados, pode ser mais eficaz do que algorítmos mais complexos. Porém a diferença entre os dois primeiros modelos foi de apenas 0.005 miléssimos, então para obtermos uma análise mais certeira, devemos olhar as matrizes de confusão e a precisão dos modelos.

Avaliando a métrica de precisão presente na figura 6, vemos que o modelo baseado em árvore ainda sai na frente, com 0.980, 1 ponto decimal acima do modelo de rede neural. o que nos mostra que o modelo de árvore realmente acertou mais previsões que os outros dois modelos.

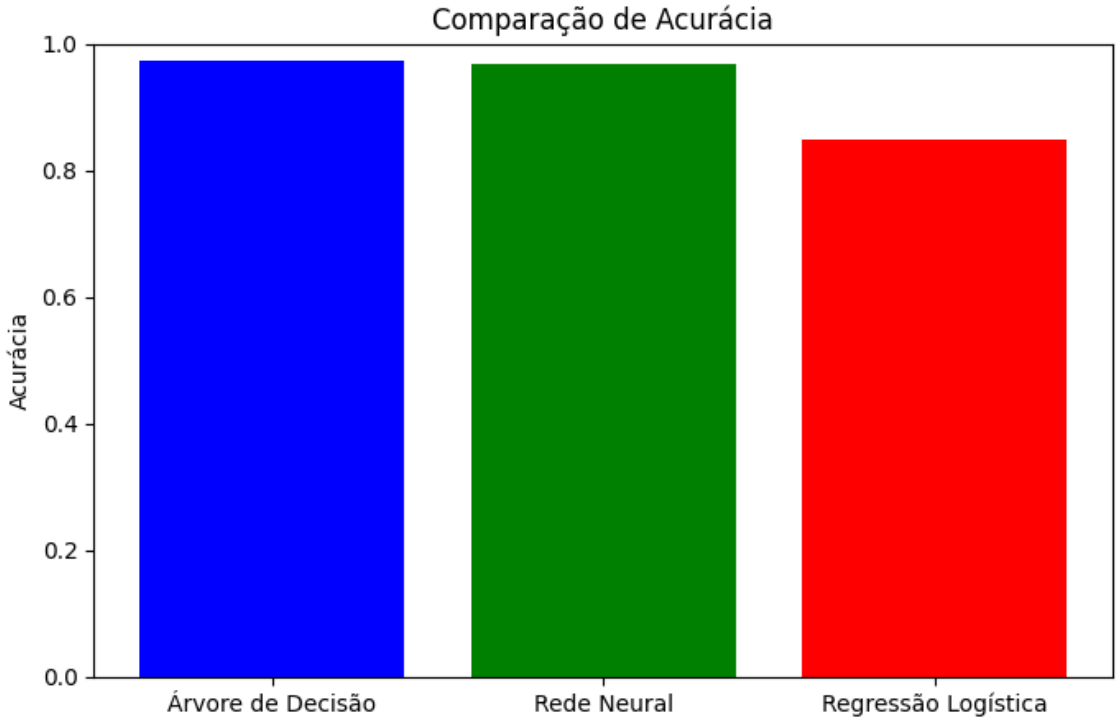

Figura 5. Comparação entre as acurácias

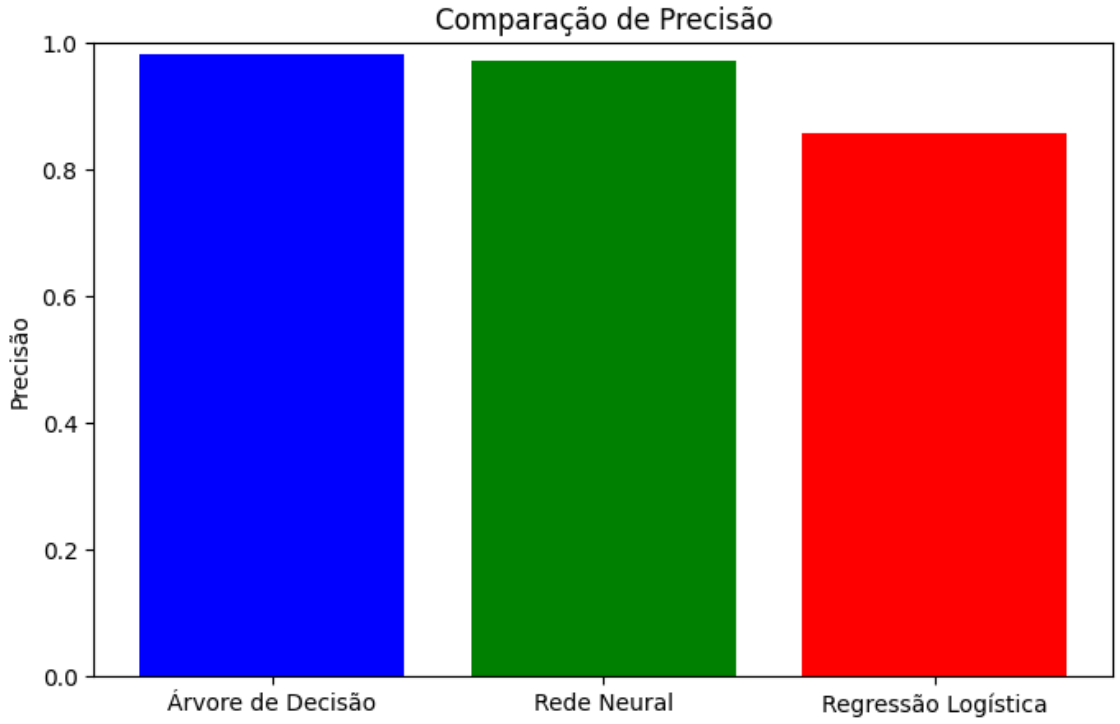

Figura 6. Comparação entre as precisões

Por fim, corroborando com a presente análise, vemos que as matrizes de confusão presentes na figura 7, nos confirmam que o modelo de arvores de decisão “ganha”dos

outros e em especial do modelo de rede neural pela previsão de 5 falsos negativos a menos. Assim, chegamos a conclusão que, para esse *dataset*, o melhor modelo é o baseado em árvore.

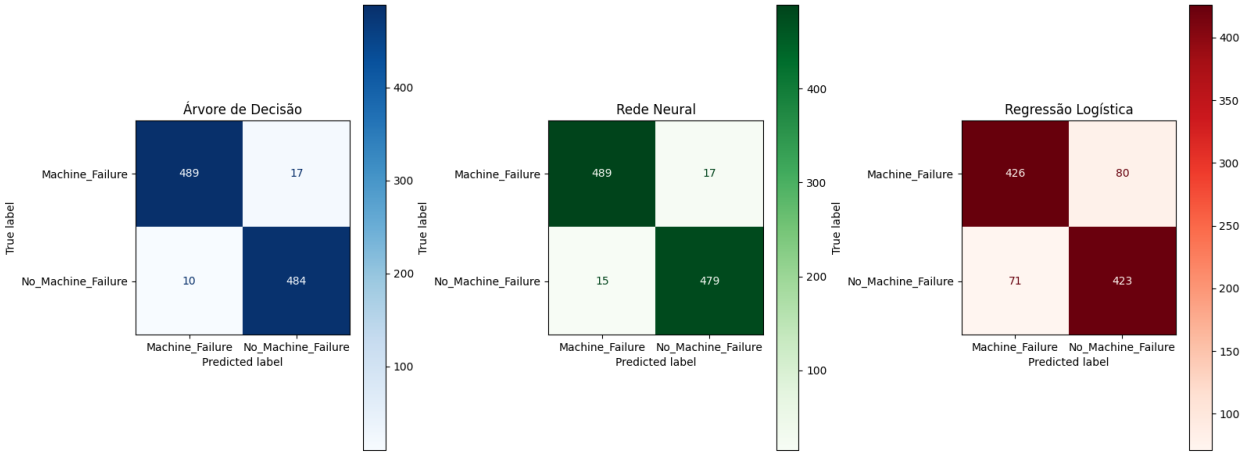

Figura 7. Comparação entre as matrizes de confusão

4. Conclusão

Ao fim desta pesquisa, concluímos que todos os passos envolvidos na obtenção dos dados, tratamento e estruturação são importantes para constituir conjuntos que possam ser bem analisados e compreendidos pelos modelos de aprendizagem de máquina.

Destacamos, também, que o modelo mais adequado para esta classificação binária foi o modelo de árvore de decisão com uma acurácia de 97,3%, seguida pela rede neural com 96,8% e pela regressão logística com 84,9%.

Dentre as principais limitações envolvidas estão o tempo de seção disponibilizado no google colab, que após aproximadamente 3 horas desconecta o ambiente de execução, fazendo com que qualquer trabalho não salvo seja perdido, levando à limitações quanto à algorítmos mais complexos como o grid search, que foi usado (de maneira minimizada) para encontrar os melhores hiperpâmetros para as redes neurais.

Mediante isso, esperamos, para trabalhos futuros, incluir nesta análise comparativa, mais modelos de aprendizagem como as redes neurais de aprendizagem profunda (deep learning) e random forest, além da construção de um ambiente em máquina local para utilização de algorítmos mais complexos.

Referências

Guedes, T. A., Martins, A. B. T., Acorsi, C. R. L., and Janeiro, V. (2005). Estatística descritiva. *Projeto de ensino aprender fazendo estatística*, pages 1–49.

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., del Río, J. F., Wiebe, M., Peterson, P., Gérard-Marchant, P., Sheppard, K., Reddy, T., Weckesser, W., Abbasi, H., Gohlke, C., and Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, 585(7825):357–362.

Hunter, J. D. (2007). Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95.

Kwak, S. K. and Kim, J. H. (2017). Statistical data preparation: management of missing values and outliers. *Korean journal of anesthesiology*, 70(4):407–411.

Marson, M. D. (2017). *Origens e evolução da indústria de máquinas e equipamentos em São Paulo 1870-1960*. PhD thesis, Universidade de São Paulo.

McKinney, W. et al. (2011). pandas: a foundational python library for data analysis and statistics. *Python for high performance and scientific computing*, 14(9):1–9.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Strauss, E., Júnior, M. V. B., and Ferreira, W. L. L. (2022). A importância de utilizar métricas adequadas de avaliação de performance em modelos preditivos de machine learning. *Projectus*, 7(2):52–62.

Waskom, M. L. (2021). seaborn: statistical data visualization. *Journal of Open Source Software*, 6(60):3021.