LEANDRO PENATTI AGOSTINI

PREDIÇÃO DA RIGIDEZ À FLEXÃO DO PAPELCARTÃO, EMPREGANDO TÉCNICAS DE DATA MINING E MACHINE LEARNING

Monografia apresentada ao Programa de Educação Continuada da Escola Politécnica da Universidade de São Paulo, para obtenção do título de Especialista, pelo Programa de Pós-Graduação em Engenharia de Dados e Big Data.

SÃO PAULO

2024

LEANDRO PENATTI AGOSTINI

PREDIÇÃO DA RIGIDEZ À FLEXÃO DO PAPELCARTÃO, EMPREGANDO TÉCNICAS DE DATA MINING E MACHINE LEARNING

Monografia apresentada ao Programa de Educação Continuada da Escola Politécnica da Universidade de São Paulo, para obtenção do título de Especialista, pelo Programa de Pós-Graduação em Engenharia de Dados e Big Data.

Área de concentração: Tecnologia da Informação – Engenharia/ Tecnologia/ Gestão

Orientador: Luiz Sérgio de Souza

SÃO PAULO

2024

Agostini, Leandro
PREDIÇÃO DA RIGIDEZ À FLEXÃO DO PAPELCARTÃO, EMPREGANDO TÉCNICAS DE DATA MINING E MACHINE LEARNING / L. Agostini – São Paulo, 2024.
65 p.

Monografia (Especialização em Engenharia de Dados e Big Data) – Escola Politécnica da Universidade de São Paulo. PECE – Programa de Educação Continuada em Engenharia.

1. predição da rigidez à flexão do papelcartão I. Universidade de São Paulo. Escola Politécnica. PECE – Programa de Educação Continuada em Engenharia II. t.

AGRADECIMENTOS

Agradeço, primeiramente, à minha esposa e aos meus dois filhos, por todo o apoio, compreensão, incentivo durante essa jornada e por sempre acreditarem no meu potencial. Aos meus pais, pela educação e valores que me proporcionaram. Expresso também minha gratidão aos professores do curso de Engenharia de Dados e Big Data, por todo o conhecimento compartilhado, e aos meus colegas de turma, pela troca de experiências e colaboração ao longo dessa caminhada.

CURSO ENGENHARIA DE DADOS E BIG DATA

Coord.: Profa. Dra. Solange Nice Alves de Souza
Vice-Coord.: Prof. Dr. Pedro Luiz Pizzigatti Corrêa

Perspectivas profissionais alcançadas com o curso:

Com a conclusão da especialização em Engenharia de Dados e Big Data, minhas perspectivas profissionais se expandiram significativamente. O curso me proporcionou o conhecimento necessário para trabalhar com grandes volumes de dados e aplicar técnicas avançadas de aprendizado de máquina e mineração de dados. Agora, estou capacitado para atuar em projetos que demandam a análise e extração de insights estratégicos a partir de dados, além de desenvolver soluções eficientes para otimizar processos empresariais. Esse aprendizado fortalece minha capacidade de contribuir para a transformação digital nas empresas, agregando valor por meio de decisões baseadas em dados.

RESUMO

O assunto desenvolvido neste trabalho acadêmico está relacionado à Indústria e Produção de Papelcartão. Uma das principais características deste produto é sua rigidez à flexão. A produção de papelcartão é contínua e impossibilita a realização da medição de rigidez durante o processo produtivo, já que para se conhecer esta resultante, é necessário a coleta de amostras físicas e medição em laboratório com equipamento específico. O uso de dados históricos de todos os parâmetros de máquina registrados, atrelado à identificação de comportamentos e padrões pode predizer os valores de rigidez sem a necessidade de coletar amostras físicas de papelcartão, o que possibilitaria melhor controle da qualidade do produto, redução de desperdícios, economia de custos com matéria prima e flexibilização de atividades tanto da área de operações quanto da área de controle da qualidade.

Foram avaliados diferentes algoritmos de aprendizado de máquina para prever a rigidez do papelcartão em tempo real. Os resultados mostram que o modelo *Redes Neurais* obteve o melhor desempenho com um MSE de 0,0350, MAE de 0,1038 e r^2 de 0,9668 e os modelos *Random Forest* e *KNN* obtiveram excelente precisão. O *Random Forest*, por exemplo, apresentou um MSE de 1,6976, MAE de 0,5669 e r^2 de 0,9807, enquanto o *KNN* obteve um MSE de 1,7161, MAE de 0,5858 e r^2 de 0,9805. Esses valores refletem a capacidade dos modelos de capturar com precisão a rigidez do papelcartão com base em dados de processo, revelando sua potencial aplicação para controle de qualidade em ambientes produtivos contínuos.

A aplicação do algoritmo utilizando Redes Neurais, permite ação preventiva dos operadores antes que os parâmetros de qualidade do produto ultrapassem seus limites de especificação, além de viabilizar a identificação e remoção de materiais não conformes, garantindo que apenas produtos dentro dos padrões sejam enviados aos clientes.

Palavras-chave: Aprendizado de Máquina, Internet das Coisas, Papelcartão, Produção de Papel, Rigidez, Predição, Análise Preditiva, Simulação, Big Data, Análise Estatística, Mineração de Dados, Inteligência Artificial.

ABSTRACT

The subject developed in this academic work is related to the Paperboard Production Industry. One of the main characteristics of this product is its flexural rigidity. Paperboard production is continuous and makes it impossible to measure rigidity during the production process, as obtaining this result requires the collection of physical samples and measurement in a laboratory with specific equipment. The use of historical data for all registered machine parameters, coupled with the identification of behaviors and patterns, can predict rigidity values without the need to collect physical paperboard samples. This would enable better product quality control, waste reduction, cost savings on raw materials, and greater flexibility in both operations and quality control areas.

Different machine learning algorithms were evaluated to predict the stiffness of paperboard in real-time. The results show that the Neural Networks model achieved the best performance, with an MSE of 0.0350, MAE of 0.1038, and r^2 of 0.9668. The Random Forest and KNN models also demonstrated excellent accuracy. For example, Random Forest achieved an MSE of 1.6976, MAE of 0.5669, and r^2 of 0.9807, while KNN obtained an MSE of 1.7161, MAE of 0.5858, and r^2 of 0.9805. These values reflect the models' ability to accurately capture the stiffness of paperboard based on process data, highlighting their potential application for quality control in continuous production environments.

The application of the algorithm using Neural Networks enables preventive actions by operators before quality deviates from specifications, while also allowing the identification and removal of non-conforming materials, ensuring that only products meeting standards are delivered to customers.

Keywords: Machine Learning, Internet of Things, Paperboard, Paper Manufacturing, Bending Stiffness, Prediction, Predictive Analysis, Simulation, Big Data, Statistical Analysis, Data Mining, Artificial Intelligence.

LISTA DE FIGURAS E TABELAS

LISTA DE SIGLAS E ABREVIATURAS

ABNT – Associação Brasileira de Normas Técnicas

DSR – Design Science Research

ETL – Extração, Transformação e Carregamento de Dados

GBM – Gradient Boosting Machine

KDD – Knowledge Discovery in Databases

KNN – K-Nearest Neighbors

L&W – Lorentzen & Wettre (Equipamento de medição da Rigidez)

MAE – Erro Médio Absoluto

MSE – Erro Quadrático Médio

PCA – Principal Component Analisys

R2 – Coeficiente de Determinação

RBF – Radial Basis Function

SAP – Software para gestão de processos de negócios

SQL – Structured Query Language

SR – Schopper-Riegler

SVM – Support Vector Machine

SVR – Support Vector Regression

TAG – Etiqueta, Marcação

SUMÁRIO

1 INTRODUÇÃO

A indústria de papel é fundamental no mercado global e tem impacto significativo em diversas áreas econômicas, ambientais e sociais e segue em crescimento devido à demanda constante por produtos de papel em suas várias aplicações, como embalagens, papel de imprimir e escrever e produtos de higiene. De acordo com o Relatório Anual do IBÁ (Indústria Brasileira de Árvores) 2023, o Brasil, junto à Estados Unidos, Canadá e China representam uma parcela substancial da produção global.

Esta indústria está dividida em dois segmentos principais: o de papel para impressão e escrita, e o de papel para embalagens, onde o papelcartão é um dos produtos mais destacados e será tema deste trabalho acadêmico.

O papelcartão é um tipo de papel mais espesso e resistente, utilizado principalmente na fabricação de embalagens, utilizadas em diversos setores com destaque para o alimentício e farmacêutico. Suas propriedades de resistência, durabilidade, capacidade de impressão de alta qualidade e espaço para informações essenciais dos produtos, fazem dele uma ótima escolha para embalagens que precisam ser tanto funcionais quanto atraentes para o consumidor final.

Uma das principais características do papelcartão é sua resistência à flexão ou rigidez, característica esta muito importante no processo de formação da embalagem final, conhecida também pela nomenclatura “cartucho” na indústria gráfica, sendo este atributo fundamental para garantir a integridade estrutural, a estabilidade e a funcionalidade de embalagens, caixas e outros produtos fabricados com papelcartão.

Em primeiro lugar, a rigidez à flexão é essencial durante o processo de fabricação, especialmente nas etapas de corte, dobra e conformação do cartucho. Materiais que apresentam boa rigidez à flexão são mais fáceis de manusear nas linhas de produção, resultando em eficiência operacional e menor incidência de falhas.

A produção do papelcartão é contínua e a medição direta da rigidez à flexão acontece com a retirada de amostras físicas de tamanho pré-determinado nos sentidos longitudinal e transversal da folha de papel para ensaio em laboratório com

uso de um equipamento que mede através de uma força conhecida aplicada à esta amostra a sua rigidez resultante, medida em mN.m (mili Newton x metro).

A utilização de um scanner na máquina de papel, que realiza leituras de gramatura, espessura e umidade em tempo real, possibilita conhecer de maneira on-line, algumas variáveis correlatas à rigidez à flexão.

O scanner, ao realizar uma "varredura" contínua, gera registros a cada segundo das variáveis acima citadas. Isso não apenas fornece informações em tempo real para ajustes imediatos, mas também cria um histórico detalhado que pode ser analisado posteriormente para otimizações contínuas e aprimoramentos no processo produtivo. A leitura contínua de gramatura, espessura e umidade contribui para o controle de qualidade e a prevenção de desvios que poderiam comprometer a rigidez à flexão do papelcartão, além de outras inúmeras variáveis de processo.

O processo de medição de rigidez no laboratório de qualidade complementa essa abordagem. Mesmo que a medição não seja realizada continuamente, a retirada de amostras físicas após cada rolo finalizado proporciona uma representação do produto ao longo do tempo.

Este é um processo em que o analista de qualidade demanda aproximadamente 5 minutos de seu tempo a cada 30 minutos, que é o tempo aproximado de fabricação de um rolo jumbo de 6 à 7 toneladas na máquina de papel do caso estudado. A informação obtida é crucial para validar a conformidade do papelcartão com os padrões estabelecidos, assegurando que o produto final atende às exigências de rigidez à flexão, porém, variações de outros parâmetros de máquina e processo podem interferir nos resultados de rigidez e este não ter sido identificado nas amostras medidas.

1.1 Motivação

Para a comunidade acadêmica, a exploração e uso de diferentes modelos de predição oferece uma oportunidade de desenvolver novos conhecimentos sobre a mecânica e o comportamento dos materiais celulósicos. Para o setor industrial, resolver esses desafios pode resultar em produtos de papelcartão mais eficientes,

econômicos e ecologicamente sustentáveis. A solução dessas questões práticas pode melhorar a competitividade das empresas e contribuir para a evolução das embalagens sustentáveis, com impacto em vários setores, incluindo alimentos, cosméticos e produtos farmacêuticos.

1.2 Objetivo

Avaliar diferentes modelos de predição que, baseados em dados de processo e medições de qualidade, possam determinar a rigidez do papelcartão em tempo real. A abordagem proposta visa eliminar a dependência de amostras físicas, que frequentemente não refletem de forma precisa a produção contínua.

E para alcançar este objetivo principal, tem-se como objetivos específicos:

- Avaliar diferentes algoritmos para determinar quais possuem maior precisão e confiabilidade na predição da rigidez do papelcartão.
- Realizar experimentos com dados históricos de processo e de medições de qualidade, treinando os modelos com diferentes subconjuntos de dados, a fim de identificar aqueles que melhor capturam a relação entre as variáveis de processo e a rigidez do papelcartão.
- Implementar um processo de validação cruzada, com avaliação dos modelos em cenários de produção real, garantindo que os resultados obtidos em condições controladas sejam replicáveis e estáveis em ambiente de produção contínua.

1.3 Justificativa

Para garantir a predição precisa e confiável da rigidez do papelcartão, é essencial avaliar uma variedade de algoritmos que capturem comportamentos das variáveis de processo, identificando os mais adequados em termos de precisão e estabilidade. A experimentação com dados históricos de produção permite explorar a variabilidade do processo, treinando modelos com subconjuntos de dados que melhor representem a relação entre variáveis e rigidez, e resultando em um modelo adaptado às condições reais de produção. Ao avaliar diferentes modelos, busca-se maximizar

a acurácia e robustez das previsões, aproveitando os pontos fortes de cada abordagem, o que contribui para predições mais confiáveis e consistentes. Para assegurar que o modelo se mantenha replicável e estável em produção contínua, a implementação de validação cruzada é fundamental, garantindo previsões robustas e mitigando o risco de overfitting, além de possibilitar a integração do modelo ao sistema de produção com confiança, eliminando a necessidade de amostras físicas para monitoramento de qualidade.

1.4 Metodologia

A abordagem metodológica aplicada à esta pesquisa é a Design Science Research (DSR) que traz uma forma de produzir conhecimento relevante. Ela é utilizada em várias áreas, como sistemas de informação, gestão organizacional e ciência da informação e tem como principais características a orientação para solução de problemas específicos, envolve a pesquisa na resolução de situações-problema em que as ciências tradicionais não são suficientes e busca de soluções que melhorem sistemas existentes.

Explicitar os objetivos, explicar como os modelos podem ser testados e descrever os mecanismos que gerarão os resultados a serem controlados ou acompanhados vão garantir a validade dos resultados gerados pela DSR.

Realiza-se no decorrer deste trabalho a avaliação de algoritmos de aprendizado de máquina, selecionando um conjunto inicial de modelos a serem testados. Em seguida, realizar-se-á a experimentação e ajuste dos modelos com dados históricos de processo, utilizando técnicas de validação cruzada para encontrar os melhores parâmetros.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo tem como objetivo apresentar e discutir os principais conceitos, teorias e abordagens que embasam o tema investigado, situando-o no contexto acadêmico e científico relevante.

2.1 Processo de fabricação de papelcartão

O processo de fabricação do papelcartão pode ser dividido em vários “blocos”, em que cada um realiza uma transformação na matéria-prima fibrosa. Para um entendimento macro e inicial sobre este processo, dividiremos em seis grandes blocos: preparação de massa, caixa de entrada e formação, prensagem, secagem, aplicação e enroladeira.

2.1.1 Preparação de massa

Esta é a etapa inicial do processo de fabricação de papelcartão com uma sequência de operações que promovem alterações na estrutura das fibras celulósicas, adequando-as às necessidades do papel a ser produzido.

A matéria prima fibrosa é submetida a tratamentos mecânicos e à adição de produtos químicos (aditivos) necessários à fabricação do papel como mostrado na Figura 1.

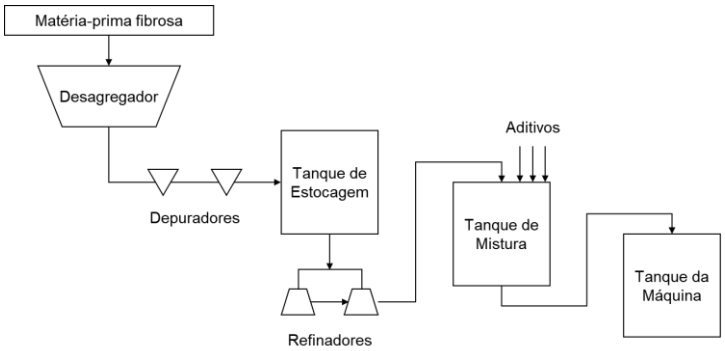

Figura 1. Diagrama de blocos da preparação de massa

Fonte: Reproduzido de ROBUSTI, Célio; VIANA, Eder Francisco; FERREIRA JÚNIOR, Fernando; GOMES, Ilduardo; TOGNETTA, Laudo; SANTOS, Osni dos; DRAGONI, Paulo. *Celulose e Papel*. São Paulo: SENAI-SP, 2014, p.37.

Durante a desagregação das fibras, devem ser rigorosamente controladas duas variáveis importantes desta etapa do processo, que são a consistência e o pH (potencial hidrogeniônico).

A consistência, expressa em porcentagem (%) e representada pela Equação 1, é utilizada para designar a quantidade de pasta celulósica seca e outros materiais sólidos existentes na massa em processo. Existem equipamentos (sensores) que realizam a medição da consistência em tempo real, além de medições periódicas em laboratório através de amostras físicas coletadas no processo.

Equação 1

$$Consistência\ (\%)\ \frac{ms}{mi} = \text{ x } 100$$

Onde:

m_s é a massa da amostra seca em estufa, expressa em g;

m_i é a massa amostra inicial, expressa em g.

O pH é a medida de acidez ou alcalinidade de uma amostra. A escala de pH compreende a faixa de 0 a 14, sendo o valor 7 considerado neutro. Este parâmetro afeta uma série de etapas na preparação da massa, como a desagregação, a refinação, a drenagem e a retenção de material fibroso e não fibroso, que farão parte da composição da folha durante sua formação. Portanto é importante ter um bom controle do pH durante todo o processo de fabricação. A medição do pH é realizada *in line* com aparelhos eletrônicos chamado medidor de pH ou “pH-metro”. Amostras também são coletadas em diversas etapas do processo para medição do pH em laboratório.

Depois de desagregada, a massa passa pela depuração, que é a operação que tem por objetivo retirar a maior quantidade possível de contaminantes com a menor perda de fibras celulósicas e de aditivos que comporão a receita de fabricação do papel. Portanto, a qualidade do papel depende fortemente do grau de limpeza da massa e uma consistência mínima de 3% e máxima de 6%.

Se a pasta celulósica, devidamente desagregada, alimentar a máquina de papel sem passar pela etapa de refinação, o produto final obtido apresentará alguma deficiência, portanto, posteriormente ao processo de depuração, esta precisa ser

refinada. A refinação consiste em um tratamento mecânico que modifica a estrutura das fibras e a fibrilação proveniente deste processo aumenta a área superficial da fibra com o consequente aumento das forças de ligação entre as fibras que vão compor a folha de papel.

Além de alguns parâmetros importantes nesta etapa do processo como consistência, temperatura, vazão e pressão, temos como principal característica a intensidade de refinação e os métodos mais comuns para determiná-la são drenabilidade (SR), potência de refino (kW), energia útil de refino (kWh/t) e nível de vácuo na máquina de papel.

No que tange a drenabilidade, é bastante comum determinar a intensidade de refinação por meio da resistência à drenagem de uma suspensão fibrosa com 2 g/L em um equipamento denominado Schopper-Riegler (SR).

No tanque de mistura, depois do processo de refinação, alguns aditivos são acrescentados à massa e os mais comuns são cargas minerais, amido e agentes de colagem interna para impermeabilização parcial do papel, alvejantes ópticos e agentes de retenção para reter na folha formada a maior quantidade possível das fibras celulósicas e aditivos que compõem a folha de papel. O cálculo da retenção é mostrado na Equação 2.

Equação 2

$$Retenção\ (\%) = \frac{\text{consistência da caixa de entrada } (\%) - consistência\ da\ água\ branca\ (\%) \times 100}{consistência\ da\ caixa\ de\ entrada\ (\%)}$$

2.1.2 Caixa de entrada e formação

A caixa de entrada, representada pela Figura 2, é o primeiro componente da máquina de papel. Sua função é receber a massa da etapa anterior e transportá-la para a mesa formadora em um fluxo constante, com a velocidade do jato uniforme em função da velocidade na direção da máquina. Este sistema não somente distribui a massa por igual ao longo da largura da máquina, como também trabalha com a velocidade e o ângulo corretos.

O lábio da caixa de entrada, também conhecido por régua, permite através de sua abertura, que o fluxo da suspensão fibrosa (jato de massa) saia da caixa de entrada e flua para a tela da máquina de papel, portanto, o controle da abertura do lábio é essencial para dar início à boa formação da folha de papel.

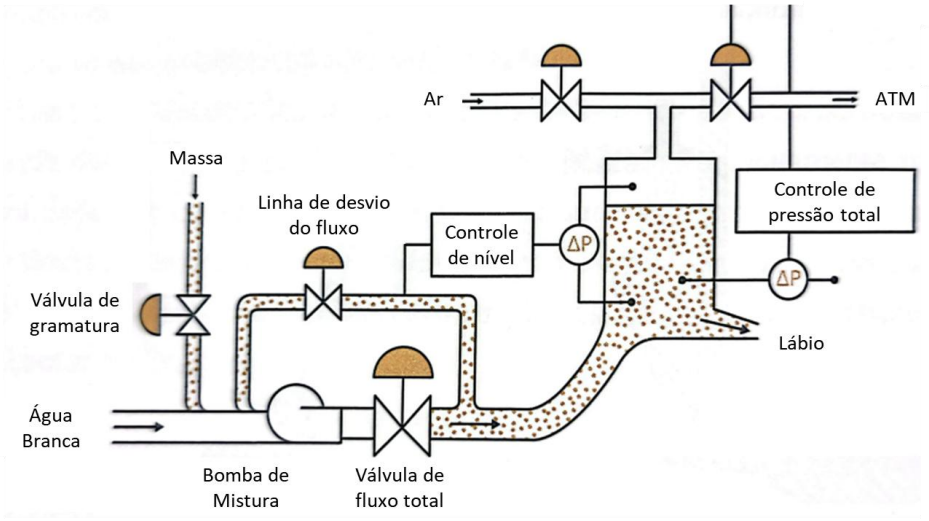

Figura 2. Caixa de entrada pressurizada

Fonte: Reproduzido de ROBUSTI, Célio; VIANA, Eder Francisco; FERREIRA JÚNIOR, Fernando; GOMES, Ilduardo; TOGNETTA, Laudo; SANTOS, Osni dos; DRAGONI, Paulo. *Celulose e Papel*. São Paulo: SENAI-SP, 2014, p.154.

Para calcular a abertura dos lábios, utiliza-se a seguinte fórmula indicada na Equação 3:

Equação 3

$$b = \frac{\mathrm{Q}}{\mathrm{Largura} \times \mathrm{V}}$$

Onde:

b é a abertura dos lábios;

Largura é a largura da máquina (m);

Q é a vazão (m^3/s);

V é a velocidade da máquina (m/min)

2.1.3 Prensagem

Logo após a formação da folha, a etapa seguinte é a prensagem que tem como principal função, retirar parte da água da estrutura capilar das fibras através da

compressão da folha, mostrado na Figura 3. Maior remoção de água determina melhor eficiência da seção.

Enquanto a mesa plana pode atingir desaguamento aproximado de 26% de teor seco, partindo de aproximadamente 1% de consistência na caixa de entrada, com as prensas é possível atingir de 46% a 51% de teor seco.

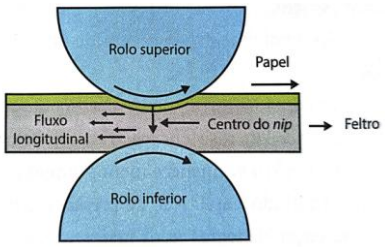

Figura 3. Prensa plana

Fonte: Reproduzido de ROBUSTI, Célio; VIANA, Eder Francisco; FERREIRA JÚNIOR, Fernando; GOMES, Ilduardo; TOGNETTA, Laudo; SANTOS, Osni dos; DRAGONI, Paulo. *Celulose e Papel*. São Paulo: SENAI-SP, 2014, p.197.

Nesta etapa, os principais parâmetros medidos pelo equipamento em tempo real são pressão e vácuo da 1ª, 2ª e 3ª prensa da máquina.

2.1.4 Secagem

Secagem é o processo na qual a água é removida da folha por evaporação e para isso faz-se o uso de cilindros secadores rotativos, representado pela Figura 4, aquecidos por vapor para fornecer calor e controle da folha em operação contínua, separados em grupos secadores com vários cilindros em cada grupo.

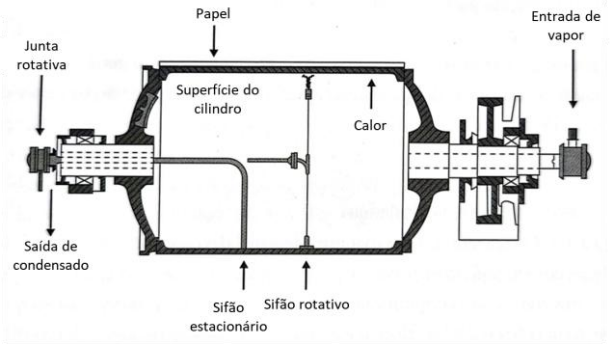

Figura 4. Cilindro secador

Fonte: Reproduzido de ROBUSTI, Célio; VIANA, Eder Francisco; FERREIRA JÚNIOR, Fernando; GOMES, Ilduardo; TOGNETTA, Laudo; SANTOS, Osni dos; DRAGONI, Paulo. *Celulose e Papel*. São Paulo: SENAI-SP, 2014, p.154.

No primeiro cilindro secador, a folha está ainda com um teor de umidade próximo de 50% e sai do último cilindro secador com teor de umidade próximo de 6%. O processo de secagem tem como principais parâmetros a vazão de vapor, abertura de válvulas, temperatura e retorno de condensado para a caldeira de vapor.

2.1.5 Aplicação de Tinta

Antes de ser aplicada, a tinta é preparada em um misturador, incluindo a adição de pigmentos, aditivos e solventes para ajuste de cor, viscosidade e brilho. Depois de preparada, o processo de aplicação da tinta no papelcartão consiste essencialmente em um rolo aplicador que retira tinta de uma calha, transferindo-a para o papel. O excesso de tinta é retirado por uma lâmina de aço de alta precisão, como pode ser observado na Figura 5.

Após a aplicação, o papelcartão passa por unidades de secagem. Essas unidades usam sistemas de ar quente ou infravermelho, que aceleram a evaporação dos solventes e ajudam na fixação da tinta na superfície.

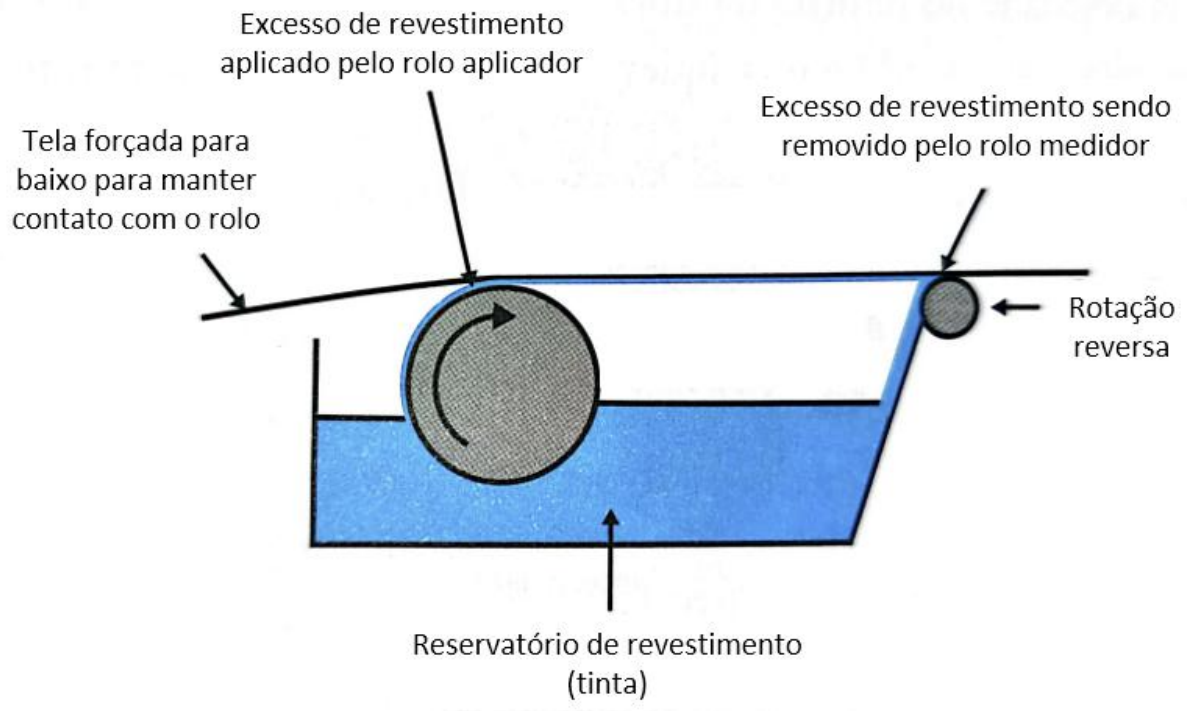

Figura 5. Cilindro aplicador

Fonte: Reproduzido de ROBUSTI, Célio; VIANA, Eder Francisco; FERREIRA JÚNIOR, Fernando; GOMES, Ilduardo; TOGNETTA, Laudo; SANTOS, Osni dos; DRAGONI, Paulo. *Celulose e Papel*. São Paulo: SENAI-SP, 2014, p.273.

2.1.6 Enroladeira

A seção de enrolamento, representada pela Figura 6, é o último elemento da máquina de papel. É constituída de eixos acionados ou cilindros suporte. Até a entrada da folha na seção de enrolamento, o processo é contínuo. A função da enroladeira é transformar a folha em unidades finitas e independentes, que permitirão o processamento e a utilização do papel.

Considerando a velocidade de produção da máquina constante, com o crescimento do rolo de papel, as rotações do eixo devem decrescer, a fim de manter constante a velocidade periférica do rolo de papel.

Para que o enrolamento não seja interrompido, quando um rolo é finalizado, corta-se o papel e passa-se a enrolá-lo em um outro eixo, também conhecido como estanga, o que permite que o processo seja initerrupto. Este é o momento em que se retira amostras fisicas do rolo que acaba de ser produzido para medições em laboratório.

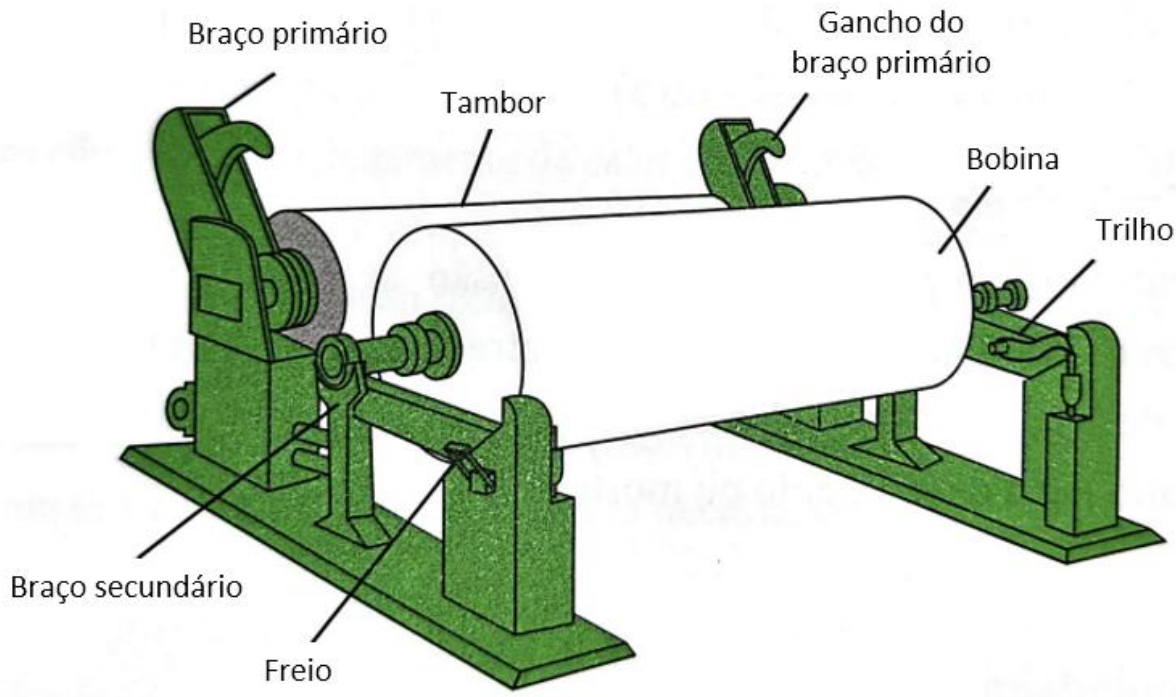

Figura 6. Seções da enroladeira

Fonte: Reproduzido de ROBUSTI, Célio; VIANA, Eder Francisco; FERREIRA JÚNIOR, Fernando; GOMES, Ilduardo; TOGNETTA, Laudo; SANTOS, Osni dos; DRAGONI, Paulo. *Celulose e Papel*. São Paulo: SENAI-SP, 2014, p.284.

2.2 Processo de medição na indústria de papelcartão

Para executar qualquer ensaio, é necessário obter amostras representativas do material a ser testado e o procedimento para amostragem de papelcartão está na NBR NM-ISSO 186:2006, Norma da Associação Brasileira de Normas Técnicas (ABNT).

Entre os diversos testes físicos realizados com amostras de papelcartão para verificação de todas suas características, a fundamentação teórica para este trabalho se restringirá ao tema do trabalho que é a rigidez à flexão do papelcartão.

Como a produção do papelcartão é contínua e não é possível realizar a medição da rigidez à flexão através de sensores enquanto o papel é fabricado, faz-se necessário a retirada de amostras físicas para medição da rigidez em laboratório. Esta medição é realizada nos dois sentidos do papel, longitudinal e transversal.

2.2.1 Determinação da rigidez à flexão (*Bending Stiffness*)

Rigidez à flexão é a força necessária para fletir um corpo de prova preso em uma de suas extremidades, até formar um ângulo de flexão de 15º. Esse procedimento visa determinar a força necessária para fletir um corpo de prova com comprimento de flexão de 50mm, preso em uma de suas extremidades, formando um ângulo especificado.

Podem ser utilizados alguns equipamentos como o Taber, Bekk ou L&W, desde que atendam à algumas especificações:

- possua ângulo de flexão de 15º +/- 0,3º ou 7,5º +/- 0,3º;
- possa prender um corpo de prova com largura de 38,0mm +/- 0,2 mm;
- velocidade de flexão constante e ângulo de flexão de 15º alcançado na faixa de tempo de 3s a 20s;
- escala de leitura com exatidão de +/- 2%;
- dispositivo para cortar os corpos de prova.

2.2.2 Procedimentos

- cortar cinco corpos de prova (mínimo), com largura de 38,0mm +/- 0,2mm e comprimento não inferior a 70mm, para cada direção do papel.
- colocar e alinhar o corpo de prova na garra do equipamento de forma que o comprimento livre seja de 57mm +/- 3mm e acionar o botão de início do teste.

2.2.3 Resultados de laboratório

Após alcançar os 15º de flexão da amostra, o equipamento mostra em seu display o resutado de resistência à flexão para a amostra medida. Um exemplo pode ser visualizado na Figura 7 abaixo. O mesmo processo é executado para as 10 amostras de cada rolo jumbo de papelcartão produzido e os valores são anotados e inseridos em sistema, no caso, o SAP. O sistema calcula a média aritmética das leituras para cada direção ensaiada e expressa a resistência à flexão em mN.m, com três algarismos significativos.

Figura 7. Aparelho L&W

Fonte: ABB. Imagem disponível em: https://new.abb.com /news/pt-br/detail/99192/abb-lanca-mais-nova-geracao-de-lw-bending-tester-para-testes-faceis-rapidos-e-confiaveis-de-papel. Acesso em: 06 nov. 2024.

2.3 Escolha dos modelos de predição

A escolha e validação de modelos é um passo essencial no processo de desenvolvimento de modelos de predição, garantindo que o modelo escolhido tenha bom desempenho e generalize bem para novos dados.

Os dados são normalmente separados em três conjuntos, um para treinamento que vai treinar o modelo, outro para validação, utilizado para ajustar hiperparâmetros e por fim o conjunto de dados de teste, completamente isolado para verificar o desempenho final do modelo em dados não vistos.

Existem mais de 40 técnicas de modelagem que podem ser utilizadas para prever um evento e alguns indicadores são utilizados para medição de desempenho do modelo. É comum utilizar:

- MSE (Erro Quadrático Médio): Dá mais peso a erros maiores, elevando ao quadrado a diferença entre valor real e predito, representado pela Equação 4:

Equação 4

$$MSE = \frac{1}{n_0}\sum_{i=1}^{n}(\mathrm{y}_i - \hat{\mathrm{y}}_i)^2$$

onde y_i é o valor real, $\hat{y}_i$ é o valor previsto, e n é o número de observações

Quanto menor o MSE, melhor é o ajuste do modelo. O valor tende a amplificar erros grandes, pois os erros são elevados ao quadrado. É útil principalmente para penalizar erros grandes, mas pode ser mais difícil de interpretar diretamente em termos práticos, já que está na unidade elevada ao quadrado dos valores reais.

- MAE (Erro Médio Absoluto): Mede a média das diferenças absolutas entre os valores preditos e os valores reais sendo representado pela Equação 5:

Equação 5

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|\mathrm{y}_i - \hat{\mathrm{y}}_i|$$

O MAE fornece a magnitude média dos erros de forma direta, pois não eleva os erros ao quadrado. Ele é menos sensível a erros grandes que o MSE, sendo útil para

entender o erro médio de cada previsão. Quanto menor o MAE, melhor é o ajuste do modelo.

- R^2 (Coeficiente de Determinação): Mede quão bem o modelo captura a variabilidade dos dados, mostrado na fórmula a seguir na Equação 6:

Equação 6

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

onde $\bar{y}$ é a média dos valores reais.

O valor do R^2 varia entre 0 e 1, onde 1 indica que o modelo explica toda a variação dos dados, enquanto valores próximos de 0 indicam que o modelo não explica quase nada. Um R^2 alto indica que o modelo se ajusta bem aos dados, mas um valor muito alto pode ser um sinal de sobre ajuste, dependendo do contexto.

- Acurácia (para classificação): Mede a porcentagem de previsões corretas em relação ao total. A fórmula é mostrada na Equação 7 abaixo:

Equação 7

$$Acurácia = \frac{Nº\ de\ Previsões\ Corretas}{Nº\ Total\ de\ Previsões}$$

Um valor de acurácia próximo de 1 (ou 100%) indica que o modelo classifica corretamente a maioria dos casos. Contudo, em conjuntos de dados desbalanceados, a acurácia pode ser enganosa, pois ela não considera a distribuição das classes (por exemplo, um modelo que classifica todas as amostras como a classe majoritária ainda pode ter alta acurácia).

Após a validação, o modelo deve ser testado em um ambiente de produção, onde recebe dados em tempo real. Isso ajuda a garantir que o modelo funciona adequadamente sob as condições reais de operação. O acompanhamento de desempenho contínuo é essencial para identificar possíveis problemas de degradação do modelo.

Por fim, realizar-se-á uma comparação entre os modelos para avaliar se a complexidade adicional de modelos mais avançados é justificada pelo ganho de desempenho.

De maneira geral, modelos de predição podem ser utilizados para classificação ou regressão. Os modelos comumente aplicados à classificação de dados, como o Naive Beyes, não serão considerados pois não há dados categóricos entre os parâmetros citados como principais no processo de fabricação do papelcartão.

Além da aplicação da Regressão Linear para n variáveis, serão analisadas as principais técnicas de modelagem para previsão de variáveis numéricas como Redes Neurais, SVM (Support Vector Machine), K-NN (Nearest Neighbors), Random Forest e a utilização do PCA (Principal Component Analisys).

2.3.1 Regressão Linear

A regressão linear é um método estatístico usado para modelar a relação entre uma variável dependente (ou alvo) e uma ou mais variáveis independentes (ou preditoras). Seu objetivo é prever o valor da variável dependente com base nos valores das variáveis independentes. A Equação 8 mostra a regressão linear simples (com uma variável independente) descrita como:

Equação 8

$$Y = b0 + b1X + \epsilon$$

Onde:

Y é a variável dependente que queremos prever.

X é a variável independente.

b_0 é o intercepto (o valor de Y quando X=0).

b_1 é o coeficiente de inclinação

ϵ é o erro, representando a diferença entre o valor observado e o valor previsto.

O modelo de regressão linear ajusta uma linha reta (no caso da regressão linear simples) ou um hiperplano (na regressão linear múltipla) que minimiza o erro entre os valores previstos e os valores reais. Para isso, utiliza o método dos mínimos quadrados, que minimiza a soma dos quadrados dos erros.

Uma vez que os coeficientes b_0 e b_1 são ajustados durante o treinamento, eles são usados para prever novos valores de Y com base nos novos valores de X.

Utilizando múltiplas variáveis, o modelo inclui mais termos b_2X_2,b_3X_3, etc., para modelar as influências de múltiplos fatores sobre a variável de interesse.

Esse modelo é especialmente útil quando a relação entre as variáveis é aproximadamente linear e quando se busca interpretar o impacto de cada variável preditora.

2.3.2 Árvore de Decisão

Uma árvore de decisão é um modelo preditivo que utiliza uma estrutura de árvore para tomar decisões baseadas nos valores de variáveis de entrada. Ela é amplamente usada tanto para problemas de regressão (previsão de variáveis contínuas) quanto de classificação (previsão de categorias). A árvore de decisão é composta por:

- Nós internos: Representam uma variável de decisão (ou preditora) e uma condição de divisão (split). Cada nó faz uma pergunta baseada no valor de uma variável.
- Ramos: São os caminhos que conectam os nós e representam o resultado de uma condição. Cada ramo leva a outro nó ou a um nó folha.
- Nós folha: São as saídas finais que fornecem a previsão.

O algoritmo de construção da árvore divide iterativamente o conjunto de dados em subconjuntos com base nas variáveis preditoras. O objetivo é escolher divisões que melhor separam os dados para a previsão da variável alvo.

Para problemas de regressão, a função de custo usada é geralmente o erro quadrático médio (MSE), e a árvore tentará minimizar esse erro a cada divisão.

A árvore cresce dividindo repetidamente os dados até que uma condição de parada seja atingida, como um número mínimo de amostras por nó, ou até que as divisões não melhorem significativamente a previsão.

Uma vez construída, a árvore é usada para fazer previsões ao passar novas amostras pelos nós, seguindo as condições de divisão até chegar a um nó folha, que dá o valor de previsão sendo este uma média dos valores das amostras nesse nó.

A principal vantagem das árvores de decisão é a interpretabilidade, já que é fácil visualizar o processo de tomada de decisão. Além disso, elas podem capturar interações complexas entre variáveis sem exigir muita preparação dos dados.

2.3.3 SVM – Máquinas de Vetores de Suporte

As Máquinas de Vetores de Suporte (SVM) são um poderoso algoritmo de aprendizado supervisionado, utilizado tanto para classificação quanto para regressão. Seu principal objetivo é encontrar um hiperplano que melhor separa os dados em diferentes categorias (para classificação) ou ajusta uma função que faça previsões precisas (para regressão).

O SVM busca encontrar uma função que mantém a maior parte dos dados dentro de um intervalo de erro aceitável, ajustando uma linha que se aproxima dos dados.

Os vetores de suporte são os pontos de dados que estão mais próximos do hiperplano e determinam sua posição e orientação. Apenas esses pontos influenciam diretamente o modelo, ignorando os dados que estão longe da fronteira de decisão.

O SVM maximiza a margem entre os dados mais próximos de classes opostas para garantir que as previsões futuras sejam feitas com maior confiança. Para regressão, a margem é um intervalo ao redor da linha de previsão que define o quão longe os pontos de dados podem estar da função de previsão sem serem penalizados.

Quando os dados não podem ser separados linearmente, o SVM usa funções de kernel para transformar os dados em um espaço de dimensões mais altas, onde um hiperplano pode ser encontrado. Os kernels mais comuns são o linear, polinomial e o RBF (Radial Basis Function).

No caso de regressão, o modelo é chamado de Support Vector Regression (SVR). Em vez de encontrar um hiperplano que separa as classes, o SVR encontra uma linha de regressão e tenta ajustar os dados dentro de uma margem de erro aceitável (denotada por um parâmetro ϵ) tendo como objetivo minimizar os desvios fora da margem de erro, penalizando os pontos de dados que estão fora desse intervalo.

O SVM é eficaz em problemas complexos e de alta dimensionalidade, com forte capacidade de generalização para novos dados.

2.3.4 KNN – K–Nearest Neighbors

O K-Nearest Neighbors (KNN) é um algoritmo de aprendizado supervisionado simples, usado tanto para classificação quanto para regressão. O princípio central do KNN é prever o valor de uma variável com base nos valores dos exemplos mais próximos no conjunto de dados.

O KNN não cria um modelo explícito durante o treinamento. Ao invés disso, armazena o conjunto de dados de treinamento. Quando uma previsão é solicitada, o KNN compara a nova entrada com os dados de treinamento para encontrar os exemplos mais próximos em termos de distância. A métrica de distância mais comum usada é a distância euclidiana (para variáveis contínuas), mas outras métricas, como distância de Manhattan ou Minkowski, também podem ser usadas.

O parâmetro K define o número de vizinhos mais próximos que serão considerados para fazer a previsão. Para encontrar esses vizinhos, o KNN calcula a distância entre o ponto de teste e todos os pontos de treinamento, selecionando os K pontos mais próximos.

Para regressão, o KNN faz a previsão tomando a média (ou, às vezes, a mediana) dos valores das variáveis alvo dos K vizinhos mais próximos. Isso significa que, se você estiver prevendo uma variável contínua, o valor previsto será a média dos valores dos vizinhos mais próximos.

Como o KNN usa distâncias para determinar os vizinhos mais próximos, é comum que as variáveis sejam normalizadas ou padronizadas para garantir que variáveis com escalas diferentes não influenciem desproporcionalmente as distâncias calculadas.

A escolha de K é crítica para o desempenho do KNN. Se K for muito pequeno, o modelo pode ser sensível ao ruído (overfitting). Se K for muito grande, o modelo pode

ser excessivamente suavizado (underfitting), não capturando bem a variabilidade dos dados.

O KNN é fácil de entender e aplicar, mas sua performance depende de uma boa escolha de K e da normalização dos dados.

2.3.5 Redes Neurais (Neural Networks)

Uma rede neural é um modelo de aprendizado de máquina inspirado no funcionamento do cérebro humano. É usada tanto para classificação quanto para regressão e pode modelar relações complexas entre variáveis. As redes neurais são compostas por várias camadas de neurônios artificiais que aprendem padrões a partir dos dados.

Um neurônio artificial recebe entradas (valores das variáveis preditoras), multiplica essas entradas por pesos, soma esses valores, adiciona um viés (bias) e passa o resultado por uma função de ativação, mostrado na Equação 9. A saída do neurônio, Equação 10 abaixo, é então passada para os neurônios da próxima camada. A fórmula geral de um neurônio é:

Equação 9

$$z = \omega 1x1 + \omega 2x2 + \cdots + \omega nxn + b$$

Equação 10

$$saída = f(z)$$

Onde:

f(z) é a função de ativação;

w_i são os pesos;

x_i são as entradas;

b é o viés.

As redes neurais consistem em camada de entrada, onde os dados de entrada são alimentados, camadas ocultas onde os neurônios intermediários processam as entradas (quanto mais camadas ocultas, mais "profunda" é a rede) e a camada de saída onde a previsão é gerada.

Cada neurônio de uma camada está conectado a todos os neurônios da próxima camada.

Para introduzir não-linearidade no modelo, uma função de ativação é aplicada à soma ponderada das entradas em cada neurônio. Funções comuns incluem:

- ReLU (Rectified Linear Unit): $f(z)=\max(0,z)$, que é rápida e eficaz em redes profundas.
- Sigmoide: Transforma o valor de saída em um intervalo entre 0 e 1, útil para classificação binária.
- Tanh: Transforma o valor de saída em um intervalo entre -1 e 1.

Durante o treinamento, a rede neural ajusta os pesos e vieses para minimizar o erro nas previsões. Isso é feito através de um processo chamado backpropagation (retropropagação), onde a rede faz uma previsão. O erro entre a previsão e o valor real é calculado usando uma função de custo (como erro quadrático médio para regressão). O erro é então "propagado para trás" através da rede, e os pesos são atualizados usando um otimizador, como o gradiente descendente.

Após o treinamento, a rede neural pode ser usada para prever novos dados. No caso de regressão, a saída será um valor contínuo.

As redes neurais são poderosas e podem modelar relações complexas, mas exigem grandes volumes de dados e podem ser mais difíceis de interpretar em comparação com modelos mais simples.

2.3.6 PCA – Análise de Componentes Principais

A Análise de Componentes Principais (PCA) é uma técnica de redução de dimensionalidade usada para simplificar conjuntos de dados, ao transformar várias variáveis correlacionadas em um conjunto menor de variáveis independentes, chamadas componentes principais. O PCA é frequentemente utilizado antes de aplicar modelos de aprendizado de máquina, especialmente quando há muitas variáveis preditoras.

O objetivo do PCA é transformar um conjunto de variáveis originais (que podem estar correlacionadas) em um conjunto menor de componentes principais que retêm a maior parte da variabilidade dos dados.

Em vez de usar todas as variáveis preditoras originais, o PCA cria novas variáveis (componentes principais) que são combinações lineares das variáveis originais. Os componentes principais são novas variáveis geradas a partir das variáveis originais, onde o primeiro componente principal (PC1) explica a maior parte da variação nos dados, o segundo componente principal (PC2) explica a segunda maior parte da variação, sendo ortogonal (independente) em relação ao PC1 e isso continua para os demais componentes, com cada um explicando menores variações que o anterior.

O número de componentes principais gerados é igual ao número de variáveis originais, mas geralmente apenas os primeiros poucos componentes são necessários para capturar a maior parte da variação nos dados.

O PCA identifica os autovalores e autovetores da matriz de covariância dos dados. Os autovalores representam a quantidade de variância explicada por cada componente, enquanto os autovetores representam as direções dos componentes principais.

O PCA transforma os dados projetando-os nos autovetores, criando os componentes principais. Esses novos componentes podem ser usados como variáveis preditoras em um modelo de aprendizado de máquina.

Antes de aplicar o PCA, as variáveis geralmente são normalizadas ou padronizadas (transformadas para terem média 0 e desvio padrão 1) para garantir que todas tenham a mesma escala. Caso contrário, variáveis com maior magnitude influenciariam mais no cálculo dos componentes.

Após aplicar o PCA, os primeiros componentes principais podem ser usados como variáveis preditoras em um modelo de regressão ou classificação. Isso ajuda a reduzir o número de variáveis e a evitar problemas como multicolinearidade (quando variáveis preditoras são altamente correlacionadas), além de tornar o modelo mais eficiente e fácil de interpretar.

O PCA é útil quando há muitas variáveis, e ajuda a simplificar o modelo sem perder muita informação dos dados.

2.3.7 GBM – Máquinas de Aprendizado por Gradiente

GBM (Gradient Boosting Machine), ou Máquina de Aprendizado por Gradiente, é uma técnica de aprendizado supervisionado que combina múltiplos modelos fracos (geralmente árvores de decisão simples) para formar um modelo forte, com o objetivo de melhorar a precisão de previsões. É amplamente utilizado em tarefas de regressão e classificação. O objetivo é combinar esses modelos simples de forma sequencial para criar um modelo final mais preciso.

O boosting é a ideia central do GBM. Ao invés de treinar todas as árvores de uma vez, as árvores são treinadas sequencialmente. Cada nova árvore tenta corrigir os erros cometidos pelas árvores anteriores, concentrando-se nos exemplos mais difíceis de prever. Cada nova árvore tenta minimizar o erro residual (a diferença entre a previsão do modelo anterior e o valor real).

O gradiente é utilizado para medir o quanto o modelo precisa ser ajustado. A cada nova iteração, a árvore é ajustada para reduzir o erro residual (resíduo) das previsões anteriores. No GBM, o processo de ajuste é feito minimizando a função de perda (erro) através do gradiente descendente. Para regressão, a função de perda comumente utilizada é o erro quadrático médio.

Inicialmente, o GBM começa com uma previsão simples (como a média dos valores de saída, no caso de regressão). Em cada iteração, a próxima árvore de decisão é treinada para prever os resíduos (erros) do modelo anterior. O modelo final é atualizado somando o novo modelo ajustado com um fator de aprendizado (learning rate), que controla a contribuição de cada árvore. Isso continua por várias iterações, até que o erro seja minimizado ou o número máximo de iterações seja atingido.

Alguns parâmetros importantes são o número de árvores, ou seja, quantas árvores de decisão serão treinadas sequencialmente, o learning rate que é a taxa de aprendizado, que controla o peso de cada nova árvore na atualização do modelo, sendo que taxas de aprendizado menores resultam em um modelo mais preciso, mas

requerem mais iterações e a profundidade da árvore, onde quanto mais profundas as árvores, mais complexas elas são. Árvores muito profundas podem levar ao overfitting.

Uma vez treinado, o modelo final faz previsões combinando todas as árvores de decisão geradas. Para problemas de regressão, a saída será um valor contínuo, resultante da soma ponderada das previsões de cada árvore.

2.3.8 Random Forest

Random Forest é um algoritmo de aprendizado supervisionado que combina várias árvores de decisão para criar um modelo mais robusto e preciso. Ele é utilizado tanto para classificação quanto para regressão, sendo conhecido por sua alta precisão e resistência ao overfitting.

O Random Forest constrói várias árvores de decisão de forma independente. Cada árvore é treinada com um subconjunto aleatório dos dados e das variáveis, e a previsão final é obtida a partir da combinação das previsões dessas árvores. A previsão final é a média das previsões de todas as árvores.

Para treinar cada árvore, o Random Forest usa um método chamado bootstrap, onde uma amostra aleatória com repetição (subconjunto dos dados) é selecionada a partir do conjunto de treinamento. Assim, cada árvore é treinada em um conjunto ligeiramente diferente de dados, o que contribui para a diversidade entre as árvores.

Em cada nó de uma árvore de decisão, o Random Forest seleciona aleatoriamente um subconjunto de variáveis em vez de usar todas as variáveis disponíveis. Isso garante que as árvores sejam menos correlacionadas entre si, aumentando a robustez do modelo.

Ao combinar múltiplas árvores (que individualmente poderiam sofrer overfitting), o Random Forest melhora a generalização. Ou seja, ele consegue capturar os padrões gerais dos dados sem se ajustar excessivamente a ruídos ou outliers.

Ele também calcula a importância das variáveis, avaliando o impacto de cada variável na precisão do modelo. Isso é útil para identificar quais variáveis são mais relevantes para o modelo de previsão.

Três parâmetros principais são considerados neste modelo, sendo a quantidade de árvores (n_estimators) que o modelo irá treinar (mais árvores podem aumentar a precisão, mas também aumentam o tempo de computação), o número de variáveis selecionadas em cada nó (max_features), que controla quantas variáveis são consideradas para dividir os nós das árvores e a profundidade das árvores (max_depth) limitando a profundidade de cada árvore para prevenir o overfitting.

3 DESENVOLVIMENTO

O desenvolvimento se inicia com o levantamento de dados históricos de sensores diversos da máquina de papel, que são historiados e gravados em arquivos Excel gerados diariamente pelo sistema DeltaV Emerson (sistema de automação) e dados de qualidade de ensaios e medições realizadas em laboratório que são registrados em sistema SAP e bases de dados SQL, representado na Figura 8.

Os dados são historiados em um servidor local, denominado Wedge Server, nome da ferramenta de análise de dados utilizada pela Produção, Engenharia de Processos, Qualidade e Manutenção Operacional.

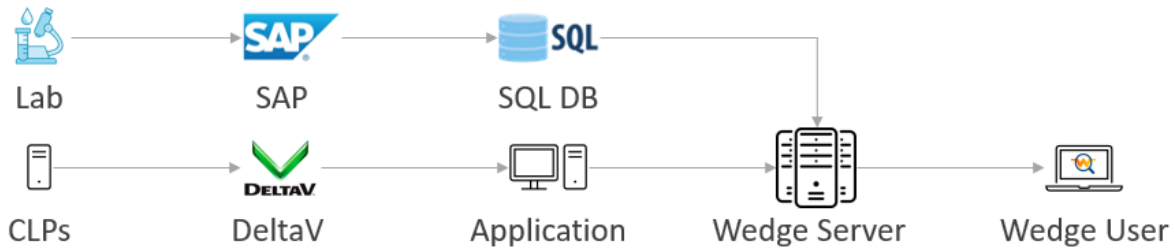

Figura 8. Arquitetura implementada

Fonte: Autoria própria

Dois processos podem ser aplicados, sendo ETL (Extração, Transformação e Carregamento de Dados) ou KDD (Knowledge Discovery in Databases).

Para a obtenção de um melhor resultado com a comparação de diferentes algoritmos, será utilizado o processo KDD que é um ciclo de descoberta de conhecimento em bases de dados, geralmente usado em análise de dados e aprendizado de máquina.

Ele envolve várias etapas que vão desde a seleção de dados até a descoberta de padrões e o uso do conhecimento extraído para tomada de decisão. A relação do KDD com ETL, consolidada na Tabela 1, está na preparação e manipulação dos dados para gerar insights.

Fase do KDD	Fase do ETL	Descrição
Seleção	Extração	Coleta de dados relevantes de várias fontes
Pré-processamento	Transformação	Limpeza e tratamento dos dados brutos
Transformação	Transformação	Criação de novas variáveis ou modificações
Mineração de Dados	-	Aplicação de algoritmos para identificar padrões
Interpretação	Carregamento	Armazenamento de dados processados e insights

Tabela 1 – Comparação das fases do Processo KDD e ETL

3.1 Seleção de Dados

A seleção de dados [Fayyad et al. (1996)], é a primeira etapa e envolve a escolha dos conjuntos de dados apropriados para a análise. A seleção de dados depende do conhecimento especializado em relação ao domínio em questão. Especialistas no campo desempenham um papel crucial na definição dos critérios de seleção e na identificação dos dados mais relevantes.

- Exploração Inicial: Para entender a estrutura e características dos dados. Isso ajuda na definição de critérios de seleção apropriados.
- Iteração: A seleção de dados não é necessariamente uma etapa única e linear do processo de KDD. Pode envolver iterações para refinar os critérios de seleção à medida que se ganha maior compreensão dos dados e dos resultados esperados

Duas bases de dados foram utilizadas para a modelagem, sendo a primeira os dados dos sensores de máquina com 387 variáveis (colunas) e registros a cada minuto do período entre 01/06/2024 e 15/09/2024, ou seja, 152.640 linhas que representam 106 dias de produção. A segunda fonte de dados é uma base dos registros de medição dos parâmetros de qualidade com 93 variáveis e mesmo período considerado.

Levando em consideração que boa parte destas variáveis não tem relação direta com as variáveis alvo rigidez transversal e rigidez longitudinal, uma seleção prévia foi realizada calculando-se a correlação entre estas e as demais variáveis possíveis.

Selecionando apenas correlações maiores que 70%, reduziu-se a quantidade de variáveis de 480 para 28, sendo que as principais podem ser observadas na Figura 9.

CORRELAÇÃO	RIGIDEZ LONGITUDINAL MEDIA	RIGIDEZ TRANSVERSAL MEDIA	DELAMINACAO MEDIA	PRESSAO CX ENT MIOLO	PRESSAO CX ENT VERSO	PRESSAO CX ENT CAPA	CONSIST MIOLO CALC	TEMP VAPOR MAQ	GRAMATURA SCANNER	UMIDADE SCANNER	ESPESSURA SCANNER	VELOCIDADE
RIGIDEZ LONGITUDINAL	100,0%	98,7%	27,9%	-88,5%	-88,2%	-88,5%	-7,4%	8,7%	94,6%	50,3%	93,7%	-90,2%
RIGIDEZ TRANSVERSAL	98,7%	100,0%	27,8%	-87,2%	-86,3%	-87,4%	-7,2%	7,6%	95,2%	53,4%	94,4%	-88,6%

Figura 9. Melhores correlações

Outro tratamento prévio nos dados foi a eliminação de registros em períodos em que a máquina de papel estava inoperante, reduzindo a base de 152.640 linhas (cada linha representa um minuto) para 137.714 linhas, resultando em uma base de dados que pode ser observada na Figura 10 (início e fim).

1	DATA_HORA	RIGIDEZ LONGITUDINAL MEDIA	RIGIDEZ TRANSVERSAL MEDIA	DELAMINACAO MEDIA	PRESSAO CX ENT MIOLO	PRESSAO CX ENT VERSO	PRESSAO CX ENT CAPA	CONSIST MIOLO CALC	TEMP VAPOR MAQ	GRAMATURA SCANNER	UMIDADE SCANNER	ESPESSURA SCANNER	VELOCIDADE
2	01/06/24 00:01	11,76	5,96	14,67	2125,92	1980,40	2146,80	5,48	157,26	230,70	7,20	393,90	431,88
3	01/06/24 00:02	11,76	5,96	14,67	2122,64	1993,66	2157,86	5,53	157,25	230,10	7,10	393,70	431,81
4	01/06/24 00:03	11,76	5,96	14,67	2125,12	1984,23	2157,40	5,57	157,30	228,60	7,00	391,70	431,92
137712	...	...	...	...	...	...	...	...	...	...	...	...	...
137713	15/09/2024 23:56	19,51	10,49	14,95	1857,16	1693,76	1868,53	5,01	5,41	268,90	7,40	478,00	396,46
137714	15/09/2024 23:57	19,51	10,49	14,95	1859,11	1693,15	1879,60	4,93	5,39	267,80	7,30	478,50	396,44
137715	15/09/2024 23:58	19,51	10,49	14,95	1855,22	1694,74	1877,78	4,84	5,37	268,40	7,40	478,40	396,71
137716	15/09/2024 23:59	19,51	10,49	14,95	1856,85	1694,37	1873,03	4,81	5,37	268,90	7,40	476,80	396,88

Figura 10. Base de dados após seleção de variáveis

3.1.1 Análise Descritiva

Para realização da análise descritiva dos dados, várias ferramentas podem ser utilizadas como Minitab, Excel e o Jamovi. A opção escolhida foi utilizar o recurso de Análise de Dados do Excel em conjunto com o Jamovi por se tratar de um software estatístico aberto para desktop e nuvem.

Aplicando uma fórmula que mede a correlação entre as variáveis dependentes (RIGIDEZ LONGITUDINAL e RIGIDEZ TRANSVERSAL) e as covariáveis (demais 26 variáveis selecionadas), selecionou-se apenas as que tem mais de 85% de correlação direta ou inversa, reduzindo as variáveis do estudo para 6.

As 6 variáveis são RIGIDEZ LONGITUDINAL MÉDIA, RIGIDEZ TRANSVERSAL MÉDIA, PRESSÃO DA CAIXA DE ENTRADA DO MIOLO, VERSO E CAPA, GRAMATURA SCANNER, ESPESSURA SCANNER E VELOCIDADE, com suas correlações indicadas na matriz da Figura 11.

	RIGIDEZ LONGITUDINAL MEDIA	RIGIDEZ TRANSVERSAL MEDIA	PRESSAO CX ENT MIOLO	PRESSAO CX ENT VERSO	PRESSAO CX ENT CAPA	GRAMATURA SCANNER	ESPESSURA SCANNER	VELOCIDADE
RIGIDEZ LONGITUDINAL MEDIA	—							
RIGIDEZ TRANSVERSAL MEDIA	0,99	—						
PRESSAO CX ENT MIOLO	-0,89	-0,87	—					
PRESSAO CX ENT VERSO	-0,88	-0,86	0,99	—				
PRESSAO CX ENT CAPA	-0,89	-0,87	0,99	0,99	—			
GRAMATURA SCANNER	0,95	0,95	-0,91	-0,91	-0,91	—		
ESPESSURA SCANNER	0,94	0,94	-0,87	-0,86	-0,87	0,97	—	
VELOCIDADE	-0,90	-0,89	0,99	0,99	0,99	-0,92	-0,87	—

Figura 11. Matriz de correlações

	RIGIDEZ LONGITUDINAL MEDIA	RIGIDEZ TRANSVERSAL MEDIA	PRESSAO CX ENT MIOLO	PRESSAO CX ENT VERSO	PRESSAO CX ENT CAPA	GRAMATURA SCANNER	ESPESSURA SCANNER	VELOCIDADE
N	137714	137714	137714	137714	137714	137714	137714	137714
Média	17.7	8.39	1994	1844	2031	246	438	408
Erro-padrão da média	0.0251	0.0114	1.29	1.23	1.34	0.107	0.221	0.154
Mediana	15.0	7.45	2153	2006	2207	241	433	431
Desvio-padrão	9.32	4.22	480	458	497	39.5	82.0	57.2
Mínimo	6.33	2.85	61.2	552	439	34.5	0.00	224
Máximo	57.1	25.2	2641	2435	2608	393	777	487

Figura 12. Estatística descritiva

Após análise descritiva dos dados realizada e mostrada na Figura 12 acima, temos a seguinte tabela de dados com as 5 primeiras linhas representadas na Figura 13:

Data_Hora	RIGIDEZ LONGITUDINAL MEDIA	RIGIDEZ TRANSVERSAL MEDIA	PRESSAO CX ENT MIOLO	PRESSAO CX ENT VERSO	PRESSAO CX ENT CAPA	GRAMATURA SCANNER	ESPESSURA SCANNER	VELOCIDADE
01/06/24 00:01	11,76	5,96	2125,92	1980,40	2146,80	230,70	393,90	431,88
01/06/24 00:02	11,76	5,96	2122,64	1993,66	2157,86	230,10	393,70	431,81
01/06/24 00:03	11,76	5,96	2125,12	1984,23	2157,40	228,60	391,70	431,92
01/06/24 00:04	11,76	5,96	2126,00	1986,78	2167,53	228,80	391,70	431,90
01/06/24 00:05	11,76	5,96	2130,02	1989,69	2162,74	229,50	392,10	431,83

Figura 13. Base de Dados com variáveis da Análise descritiva

Antes de carregar os dados para o Colab Research, ferramenta utilizada para a implementação do código que avaliará os resultados para as diferentes modelagens propostas, carrega-se algumas bibliotecas que serão utilizadas no decorrer do experimento.

```
import os
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
```

```
from sklearn import preprocessing
from sklearn.ensemble import RandomForestRegressor
```

O carregamento da base de dados do arquivo “Dados_PY.csv” com dados delimitados por tabulação é mostrado na Figura 14.

```
df = pd.read_csv('/content/Dados_PY.csv', delimiter = '\t', encoding = 'latin-1')
df.head()
```

	ï»¿TIME	GRAM PADRAO	RIGIDEZ LONGITUDINAL MEDIA	RIGIDEZ TRANSVERSAL MEDIA	PRESSAO CX ENT MIOLO	PRESSAO CX ENT VERSO	PRESSAO CX ENT CAPA	GRAMATURA SCANNER	ESPESSURA SCANNER	VELOCIDADE
0	01/06/24 00:00	225,00	11,76	5,96	2125,92	1980,40	2146,80	230,70	393,90	431,88
1	01/06/24 00:01	225,00	11,76	5,96	2125,92	1980,40	2146,80	230,70	393,90	431,88
2	01/06/24 00:02	225,00	11,76	5,96	2122,64	1993,66	2157,86	230,10	393,70	431,81
3	01/06/24 00:03	225,00	11,76	5,96	2125,12	1984,23	2157,40	228,60	391,70	431,92
4	01/06/24 00:04	225,00	11,76	5,96	2126,00	1986,78	2167,53	228,80	391,70	431,90

Figura 14. Carregamento da base de dados

Para uma verificação prévia do “data type” das variáveis, aplica-se o comando df.dtypes e o resultado é mostrado na Figura 15.

ï»¿TIME	object
GRAM PADRAO	object
RIGIDEZ LONGITUDINAL MEDIA	object
RIGIDEZ TRANSVERSAL MEDIA	object
PRESSAO CX ENT MIOLO	object
PRESSAO CX ENT VERSO	object
PRESSAO CX ENT CAPA	object
GRAMATURA SCANNER	object
ESPESSURA SCANNER	object
VELOCIDADE	object

Figura 15. Data Type base raw

3.2 Pré-processamento de Dados

O pré-processamento é uma das etapas mais importantes do KDD, pois aqui os dados são preparados e limpos. Essa fase inclui o tratamento de dados ausentes, inconsistentes ou ruidosos.

Removendo colunas sem nome e valores NaN:

```
df = df.loc[:, ~df.columns.str.contains('^Unnamed')]
df_s = df.dropna()
```

Verificação de contagem, valores únicos, primeiro valor e frequência dos dados são mostrados na Figura 16, após aplicação do comando df_s.describe().

	ï»¿TIME	GRAM PADRAO	RIGIDEZ LONGITUDINAL MEDIA	RIGIDEZ TRANSVERSAL MEDIA	PRESSAO CX ENT MIOLO	PRESSAO CX ENT VERSO	PRESSAO CX ENT CAPA	GRAMATURA SCANNER	ESPESSURA SCANNER	VELOCIDADE
count	137715	137715	137715	137715	137715	137715	137715	137715	137715	137715
unique	137715	21	2164	1397	74460	72628	78133	5587	8546	16005
top	01/06/24 00:00	215,00	11,53	5,51	2423,91	2239,96	1878,57	218,50	315,30	457,53
freq	1	24503	370	630	23	21	135	634	233	782

Figura 16. Descrição dos dados

3.3 Transformação

Após a limpeza, a transformação de dados no KDD refere-se à criação de novas características ou atributos a partir dos dados brutos. Isso pode incluir cálculos, agregações ou combinações de variáveis para facilitar a análise posterior.

No processo de ETL, a transformação pode envolver operações como agregações, cálculo de novos campos derivados, mapeamento de categorias ou reestruturação de dados.

Realizando algumas transformações como a substituição de caracteres vírgula por ponto e transformação do data type para “float”:

```
df["GRAM PADRAO"] = df['GRAM PADRAO'].str.replace(',','.')
df["GRAM PADRAO"] = df['GRAM PADRAO'].astype(float)
df["RIGIDEZ LONGITUDINAL MEDIA"] = df['RIGIDEZ LONGITUDINAL
MEDIA'].str.replace(',','.')
df["RIGIDEZ LONGITUDINAL MEDIA"] = df['RIGIDEZ LONGITUDINAL
MEDIA'].astype(float)
df["RIGIDEZ TRANSVERSAL MEDIA"] = df['RIGIDEZ TRANSVERSAL
MEDIA'].str.replace(',','.')
df["RIGIDEZ TRANSVERSAL MEDIA"] = df['RIGIDEZ TRANSVERSAL
MEDIA'].astype(float)
df["PRESSAO CX ENT MIOLO"] = df['PRESSAO CX ENT MIOLO'].str.replace(',','.')
df["PRESSAO CX ENT MIOLO"] = df['PRESSAO CX ENT MIOLO'].astype(float)
df["PRESSAO CX ENT VERSO"] = df['PRESSAO CX ENT VERSO'].str.replace(',','.')
df["PRESSAO CX ENT VERSO"] = df['PRESSAO CX ENT VERSO'].astype(float)
df["PRESSAO CX ENT CAPA"] = df['PRESSAO CX ENT CAPA'].str.replace(',','.')
df["PRESSAO CX ENT CAPA"] = df['PRESSAO CX ENT CAPA'].astype(float)
df["GRAMATURA SCANNER"] = df['GRAMATURA SCANNER'].str.replace(',','.')
df["GRAMATURA SCANNER"] = df['GRAMATURA SCANNER'].astype(float)
df["ESPESSURA SCANNER"] = df['ESPESSURA SCANNER'].str.replace(',','.')
df["ESPESSURA SCANNER"] = df['ESPESSURA SCANNER'].astype(float)
df["VELOCIDADE"] = df['VELOCIDADE'].str.replace(',','.')
df["VELOCIDADE"] = df['VELOCIDADE'].astype(float)
```

Novamente o df.dtypes é aplicado para verificação do data type da base transformada e o resultado é mostrado na Figura 17.

ï»¿TIME	object
GRAM PADRAO	float64
RIGIDEZ LONGITUDINAL MEDIA	float64
RIGIDEZ TRANSVERSAL MEDIA	float64
PRESSAO CX ENT MIOLO	float64
PRESSAO CX ENT VERSO	float64
PRESSAO CX ENT CAPA	float64
GRAMATURA SCANNER	float64
ESPESSURA SCANNER	float64
VELOCIDADE	float64

Figura 17. Data Type base transformada

Nova verificação de contagem, valores únicos, primeiro valor e frequência dos dados utilizando o df_s.describe(), resulta nos dados da Figura 18.

	GRAM PADRAO	RIGIDEZ LONGITUDINAL MEDIA	RIGIDEZ TRANSVERSAL MEDIA	PRESSAO CX ENT MIOLO	PRESSAO CX ENT VERSO	PRESSAO CX ENT CAPA	GRAMATURA SCANNER	ESPESSURA SCANNER	VELOCIDADE
count	137715.000000	137715.000000	137715.000000	137715.000000	137715.000000	137715.000000	137715.000000	137715.000000	137715.000000
mean	243.044636	17.738392	8.388461	1994.028578	1843.742318	2031.075325	246.094712	437.772029	407.631530
std	38.845111	9.319274	4.223417	480.075246	457.535185	497.169896	39.535008	82.017286	57.175763
min	190.000000	6.330000	2.850000	61.200000	551.550000	439.470000	34.540000	0.000000	224.170000
25%	215.000000	11.340000	5.440000	1647.325000	1539.410000	1683.375000	217.700000	380.385000	372.720000
50%	240.000000	15.050000	7.450000	2152.550000	2005.690000	2207.000000	240.900000	433.200000	430.670000
75%	265.000000	21.670000	10.210000	2413.630000	2238.330000	2475.570000	268.855000	491.300000	456.920000
max	350.000000	57.120000	25.180000	2641.360000	2434.800000	2608.360000	392.700000	776.900000	487.010000

Figura 18. Carregamento da base de dados

3.4 Mineração de Dados (*Data Mining*)

Aqui é onde o processo de descoberta de padrões realmente acontece. Mineração de dados envolve aplicar técnicas de aprendizado de máquina ou estatísticas para identificar padrões ou tendências ocultas.

Embora essa etapa não seja exatamente parte do processo ETL, ela depende fortemente da qualidade dos dados que foram extraídos e transformados corretamente. No KDD, essa etapa pode usar os dados limpos e transformados para rodar algoritmos de classificação, clusterização, regressão etc.

3.4.1 Regressão Linear com 3 variáveis

O algoritmo de Regressão Linear modela a relação entre uma variável dependente e uma ou mais variáveis independentes ajustando uma linha (ou

hiperplano) que minimiza a diferença entre os valores observados e previstos, assumindo uma relação linear entre elas.

Para iniciar a implementação de uma regressão linear com 3 variáveis, realiza-se a seleção destas variáveis dependentes (X) e a variável alvo (Y):

```
X3 = df_s[['GRAMATURA SCANNER','ESPESSURA
SCANNER','VELOCIDADE']].values
Y = df_s['RIGIDEZ LONGITUDINAL MEDIA'].values.reshape(-1, 1)
```

Divide-se a base de dados em duas partes, sendo treino e teste com 30% dos dados para as variáveis dependentes (X) e variável alvo (Y):

```
X_train, X_test, y_train, y_test = train_test_split(X3, Y, test_size = 0.3, random_state
= 101)
```

Modelagem com a criação do objeto para a classe e a execução da Regressão Linear:

```
linear_regressor = LinearRegression()
linear_regressor.fit(X_train, y_train)
```

Previsões para a base de teste:

```
predictions = linear_regressor.predict(X_test)
```

Avaliando o modelo:

```
print('mean_squared_error : ', mean_squared_error(y_test, predictions))
print('mean_absolute_error : ', mean_absolute_error(y_test, predictions))
print('r2 : ', r2_score(y_test, predictions))
```

```
mean_squared_error :  7.6486
mean_absolute_error :  1.8907
r2 :  0.9130
```

Coeficientes:

```
coefs = linear_regressor.coef_[0]
coefs
array([ 0.08170143,  0.04400415, -0.03991939])
```

```
intercept = linear_regressor.intercept_[0]
intercept
```

```
-5.367204267277689
```

Equação da Reta:

```
f"y = {intercept} + {coefs[0]} * b1 + {coefs[1]} * b2 + {coefs[2]} * b3 "
```

y = -5.367204267277689 + 0.08170143075754319 * b1 + 0.044004150450736454 * b2 + -0.03991939143707864 * b3

Plotando o Gráfico (Figura 19):

```
y_test.shape
```

(41315, 1)

```
sct_x = y_test.reshape(1,41315)[0]
sct_y = predictions.reshape(1,41315)[0]

plt.figure(figsize = (12,8))
plt.xlabel('Medição')
plt.ylabel('Previsão')
plt.title('Medição x Previsão - Regressão 3 Variáveis')
plt.grid(True)
plt.scatter(sct_x, sct_y)
plt.plot(sct_x, sct_x, color = 'red', label = "identidade")
plt.legend(loc = 'upper center')
```

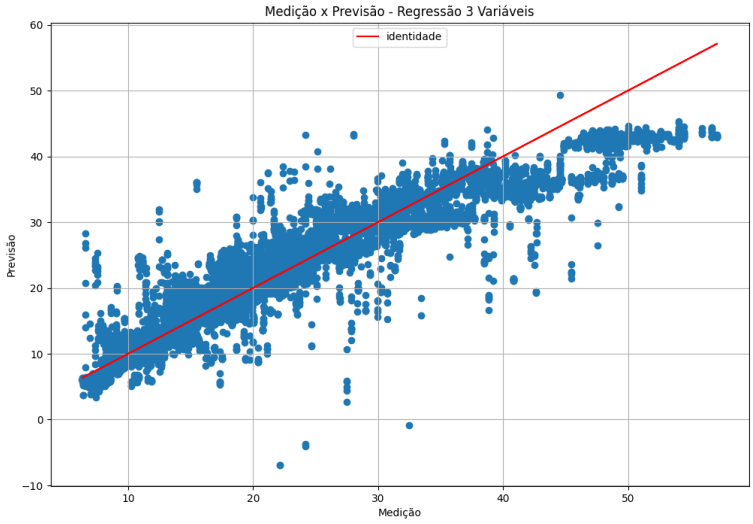

Figura 19. Medição vs. Previsão para Regressão Linear com 3 variáveis

```
print(f"y = {intercept} + {coefs[0]} * GRAMATURA SCANNER + {coefs[1]} * ESPESSURA SCANNER + {coefs[2]} * VELOCIDADE")
```

y = -5.367204267277689 + 0.08170143075754319 * GRAMATURA SCANNER + 0.044004150450736454 * ESPESSURA SCANNER + -0.03991939143707864 * VELOCIDADE

3.4.2 Regressão Linear com 6 variáveis

Uma outra maneira de realizar limpeza nos dados e conversão do data type:

```
for    col in ['GRAM PADRAO', 'RIGIDEZ LONGITUDINAL MEDIA', 'RIGIDEZ
       TRANSVERSAL MEDIA', 'PRESSAO CX ENT MIOLO', 'PRESSAO CX ENT
       VERSO', 'PRESSAO CX ENT CAPA', 'GRAMATURA SCANNER',
       'ESPESSURA SCANNER', 'VELOCIDADE']:
       df[col] = df[col].str.replace(',', '.', regex = False).astype(float)

df = df.loc[:, ~df.columns.str.contains('^Unnamed')]
df_s = df.dropna()
```

Seleção de variáveis:

```
X6 = df_s[['PRESSAO CX ENT MIOLO', 'PRESSAO CX ENT VERSO', 'PRESSAO
CX ENT CAPA', 'GRAMATURA SCANNER','ESPESSURA
SCANNER','VELOCIDADE']].values
Y = df_s['RIGIDEZ LONGITUDINAL MEDIA'].values.reshape(-1, 1)
```

Dividindo base de treinamento e de teste:

```
X_train, X_test, y_train, y_test = train_test_split(X6, Y, test_size = 0.3, random_state
= 101)
```

Criando o modelo:

```
linear_regressor = LinearRegression()
linear_regressor.fit(X_train, y_train)
```

Previsões para a base de teste:

```
predictions = linear_regressor.predict(X_test)
```

Avaliando o modelo:

```
print('mean_squared_error: ', mean_squared_error(y_test, predictions))
print('mean_absolute_error: ', mean_absolute_error(y_test, predictions))
print('r2 : ', r2_score(y_test, predictions))
```

```
mean_squared_error :  6.2482
mean_absolute_error :  1.6567
r2 :  0.9290
```

Coeficientes:

```
coefs = linear_regressor.coef_[0]
```

```
intercept = linear_regressor.intercept_[0]
```

Equação da reta:

```
print(f"y = {intercept} + {coefs[0]} * PRESSAO CX ENT MIOLO + {coefs[1]} * PRESSAO CX ENT VERSO + {coefs[2]} * PRESSAO CX ENT CAPA + {coefs[3]} * GRAMATURA SCANNER + {coefs[4]} * ESPESSURA SCANNER + {coefs[5]} * VELOCIDADE")
```

```
y = 22.105861398608358 + 0.0052209256160076906 * PRESSAO CX ENT MIOLO + -0.0027177422169695313 * PRESSAO CX ENT VERSO + 0.018294238688696188 * PRESSAO CX ENT CAPA + 0.07435119918212932 * GRAMATURA SCANNER + 0.05137885250369484 * ESPESSURA SCANNER + -0.21520044583452272 * VELOCIDADE
```

Plotando o gráfico (Figura 20):

```
sct_x = y_test.reshape(1, -1)[0]
sct_y = predictions.reshape(1, -1)[0]

plt.figure(figsize=(12, 8))
plt.xlabel('Medição')
plt.ylabel('Previsão')
plt.title('Medição x Previsão - Regressão 6 Variáveis')
plt.grid(True)
plt.scatter(sct_x, sct_y)
plt.plot(sct_x, sct_x, color='red', label="identidade")
plt.legend(loc='upper center')
plt.show()
```

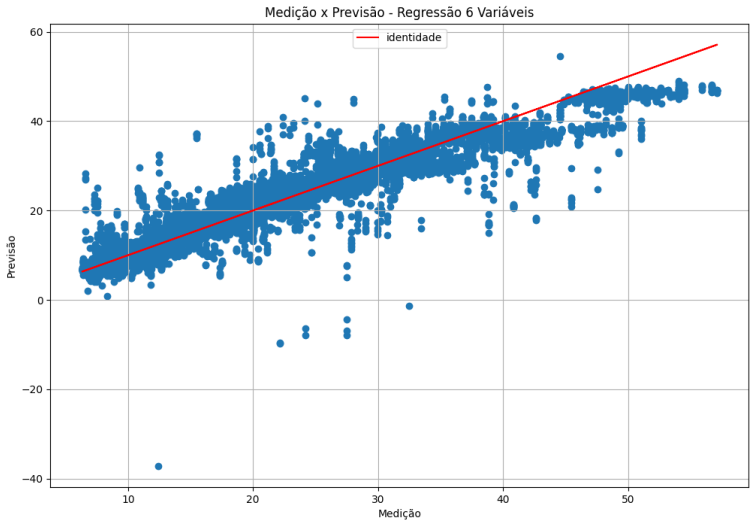

Figura 20. Medição vs. Previsão para Regressão Linear com 6 variáveis

3.4.3 Random Forest

O algoritmo Random Forest combina previsões de múltiplas Árvores de Decisão independentes, construídas a partir de diferentes subconjuntos dos dados e características, para obter um modelo robusto e preciso, reduzindo o risco de overfitting. Para a aplicação do Random Forest, seguindo com o código já desenvolvido para a Regressão linear, é necessário importar a biblioteca “RandomForestRegressor”:

```
from sklearn.ensemble import RandomForestRegressor
```

Inicializando e treinando o modelo Random Forest:

```
rf_regressor = RandomForestRegressor(random_state=101)
rf_regressor.fit(X_train, y_train.ravel())
```

Realizando as predições:

```
rf_predictions = rf_regressor.predict(X_test)
```

Avaliando o modelo:

```
print('Random Forest - mean_squared_error : ', mean_squared_error(y_test,
rf_predictions))
print('Random Forest - mean_absolute_error : ', mean_absolute_error(y_test,
rf_predictions))
print('Random Forest - r2 : ', r2_score(y_test, rf_predictions))
```

```
Random Forest - mean_squared_error :  1.6976
Random Forest - mean_absolute_error :  0.5669
Random Forest - r2 :  0.9807
```

Plotando o gráfico (Figura 21):

```
sct_x = y_test.reshape(1, -1)[0]
sct_y = rf_predictions

plt.figure(figsize=(12, 8))
plt.xlabel('Medição')
plt.ylabel('Previsão')
plt.title('Medição x Previsão - Random Forest')
plt.grid(True)
plt.scatter(sct_x, sct_y)
plt.plot(sct_x, sct_x, color='red', label="identidade")
```

```
plt.legend(loc='upper center')
plt.show()
```

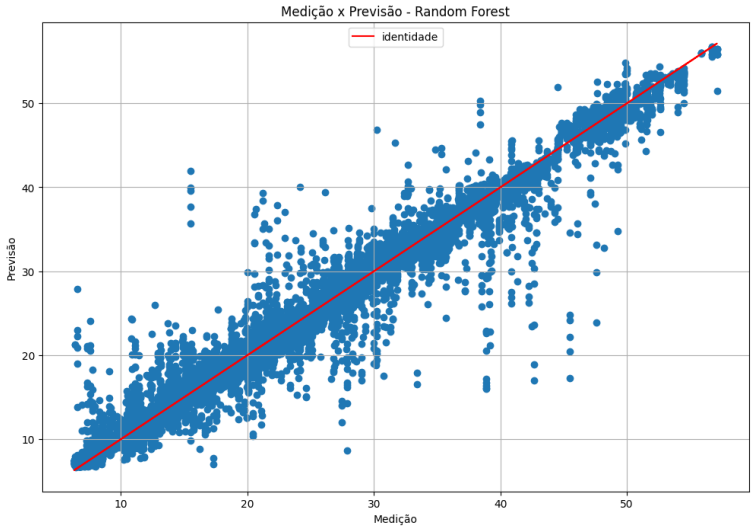

Figura 21. Medição vs. Previsão para Random Forest

3.4.4 KNN

O algoritmo KNN (K-Nearest Neighbors) classifica um novo dado com base nas classes dos “k” pontos mais próximos no espaço de características, assumindo que dados semelhantes estarão próximos uns dos outros. Importando a biblioteca:

```
from sklearn.neighbors import KNeighborsRegressor
```

Assumindo que X6 e Y já estão definidos:

```
X6 = df_s[['PRESSAO CX ENT MIOLO', 'PRESSAO CX ENT VERSO', 'PRESSAO
CX ENT CAPA', 'GRAMATURA SCANNER','ESPESSURA
SCANNER','VELOCIDADE']].values
Y = df_s['RIGIDEZ LONGITUDINAL MEDIA'].values.reshape(-1, 1)
```

Separando os dados de treino e teste e definição de parâmetros do KNN:

```
X_train, X_test, y_train, y_test = train_test_split(X6, Y, test_size=0.3,
random_state=101)

n_neighbors = 5 #@param {type:"slider", min:1, max:20, step:1}
weights = 'uniform' #@param ["uniform", "distance"]
algorithm = 'auto' #@param ["auto", "ball_tree", "kd_tree", "brute"]
leaf_size = 30 #@param {type:"slider", min:10, max:100, step:10}
p = 2 #@param {type:"slider", min:1, max:3, step:1}
metric = 'minkowski' #@param ["minkowski", "euclidean", "manhattan"]
```

Inicializando e treinando o modelo KNN com os parâmetros definidos:

```
knn_regressor = KNeighborsRegressor(n_neighbors=n_neighbors, weights=weights,
algorithm=algorithm, leaf_size=leaf_size, p=p, metric=metric)
knn_regressor.fit(X_train, y_train.ravel())
knn_predictions = knn_regressor.predict(X_test)
```

Avaliando o modelo:

```
print('KNN - mean_squared_error : ', mean_squared_error(y_test, knn_predictions))
print('KNN - mean_absolute_error : ', mean_absolute_error(y_test, knn_predictions))
print('KNN - r2 : ', r2_score(y_test, knn_predictions))
```

```
KNN - mean_squared_error :  1.7161
KNN - mean_absolute_error :  0.5858
KNN - r2 :  0.9805
```

Plotando o gráfico (Figura 22):

```
plt.figure(figsize=(12, 8))
plt.xlabel('Medição')
plt.ylabel('Previsão')
plt.title('Medição x Previsão - KNN')
plt.grid(True)
plt.scatter(y_test, knn_predictions)
plt.plot(y_test, y_test, color='red', label="identidade")
plt.legend(loc='upper center')
plt.show()
```

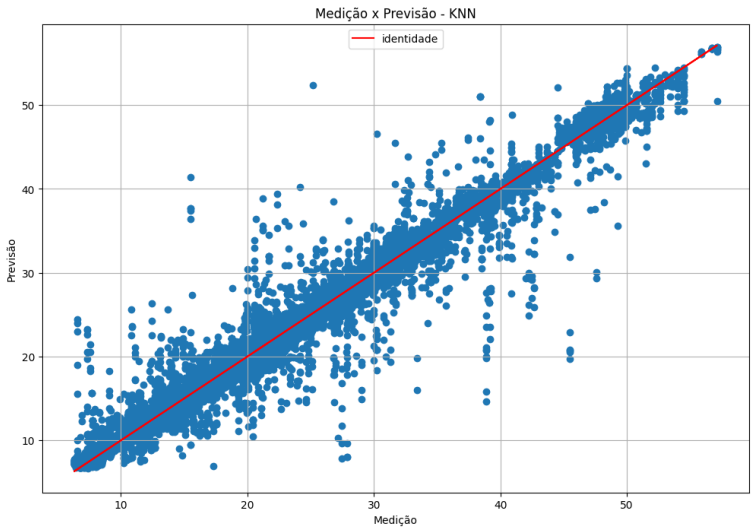

Figura 22. Medição vs. Previsão para KNN

3.4.5 Redes Neurais

Inspirado no funcionamento do cérebro humano, composto por múltiplas camadas de neurônios artificiais que processam e aprendem padrões complexos nos dados, ajustando suas conexões para realizar previsões ou classificações, o modelo Redes Neurais foi implementado, utilizando os mesmos dados e desenvolvimento até este ponto com a importação das seguintes bibliotecas:

```
import tensorflow as tf
from tensorflow import keras
from sklearn.preprocessing import StandardScaler
```

Assumindo que X6 e Y já estão definidos:

```
X3 = df_s[['PRESSAO CX ENT MIOLO', 'PRESSAO CX ENT VERSO', 'PRESSAO CX ENT CAPA', 'GRAMATURA SCANNER','ESPESSURA SCANNER','VELOCIDADE']].values
Y = df_s['RIGIDEZ LONGITUDINAL MEDIA'].values
```

Dimensionamento dos dados:

```
scaler = StandardScaler()
X3_scaled = scaler.fit_transform(X3)
Y_scaled = scaler.fit_transform(Y.reshape(-1, 1))
```

Separação dos dados em treino e teste:

```
X_train, X_test, y_train, y_test = train_test_split(X3_scaled, Y_scaled, test_size=0.3, random_state=101)
```

Definição do modelo de Redes Neurais:

```
model = keras.Sequential([keras.layers.Dense(64, activation='relu', input_shape=(X_train.shape[1],)), keras.layers.Dense(32, activation='relu'), keras.layers.Dense(1)
])
```

Compilar o modelo:

```
model.compile(optimizer='adam', loss='mse', metrics=['mae'])
```

Treinando o modelo:

```
history = model.fit(X_train, y_train, epochs=100, batch_size=32, validation_split=0.2, verbose=1)
```

Avaliando o modelo:

```
loss, mae = model.evaluate(X_test, y_test, verbose=0)
```

```
print(f"Mean Absolute Error: {mae}")
```

```
Mean Absolute Error: 0.0990
```

Realizando as predições:

```
nn_predictions = model.predict(X_test)
```

Transformação inversa das previsões e valores reais para a escala original:

```
nn_predictions = scaler.inverse_transform(nn_predictions)
y_test = scaler.inverse_transform(y_test)
```

Calculando o R^2:

```
r2 = r2_score(y_test, nn_predictions)
print(f"R-squared: {r2}")
```

```
R-squared: 0.9668
```

Plotar valores de perda de treinamento e validação (Figura 23):

```
plt.figure(figsize=(12, 8))
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('Model loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['Train', 'Validation'], loc='upper left')
plt.show()
```

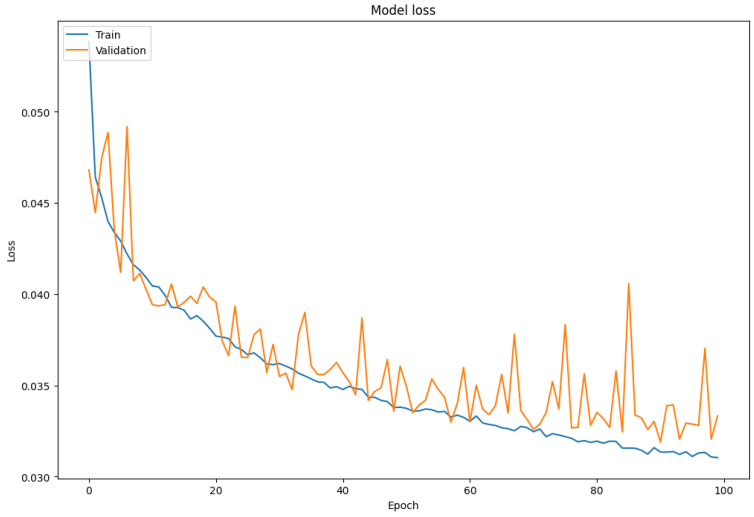

Figura 23. Valores de perda de treinamento e validação

Plotar valores reais versus valores previstos (Figura 24):

```
plt.figure(figsize=(12, 8))
plt.scatter(y_test, nn_predictions)
```

```
plt.xlabel("Actual Values")
plt.ylabel("Predicted Values")
plt.title("Actual vs. Predicted Values (Neural Network)")
plt.plot(y_test, y_test, color='red', label="identity")
plt.legend(loc='upper center')
plt.show()
```

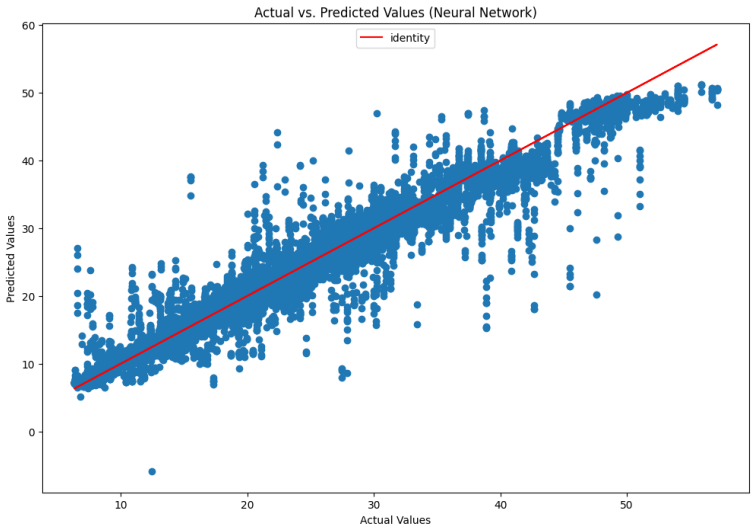

Figura 24. Medição vs. Previsão para Redes Neurais

3.4.6 SVM (Support Vector Machine)

O algoritmo SVM (Support Vector Machine) busca encontrar a melhor linha (ou hiperplano) que separa os dados em diferentes classes, maximizando a margem entre os pontos mais próximos de cada classe. Para regressão utiliza-se o SRV.

Importando a biblioteca relacionada:

```
from sklearn.svm import SVR
```

Assumindo que X6 e Y já estão definidos:

```
X6 = df_s[['PRESSAO CX ENT MIOLO', 'PRESSAO CX ENT VERSO', 'PRESSAO CX ENT CAPA', 'GRAMATURA SCANNER','ESPESSURA SCANNER','VELOCIDADE']].values
Y = df_s['RIGIDEZ LONGITUDINAL MEDIA'].values
```

Separando os dados:

```
X_train, X_test, y_train, y_test = train_test_split(X3, Y, test_size=0.3, random_state=101)
```

Inicializando e treinando o modelo:

```
svr_model = SVR(kernel='rbf', C=100, gamma=0.1, epsilon=.1)
```

```
svr_model.fit(X_train, y_train)
```

Realizando as predições:

```
svr_predictions = svr_model.predict(X_test)
```

Avaliando o modelo:

```
print('SVM - mean_squared_error : ', mean_squared_error(y_test, svr_predictions))
print('SVM - mean_absolute_error : ', mean_absolute_error(y_test, svr_predictions))
print('SVM - r2 : ', r2_score(y_test, svr_predictions))
```

```
SVM - mean_squared_error :  25.5277
SVM - mean_absolute_error :  3.2392
SVM - r2 :  0.7095
```

Plotando o gráfico (Figura 25):

```
plt.figure(figsize=(12, 8))
plt.xlabel('Medição')
plt.ylabel('Previsão')
plt.title('Medição x Previsão (SVM)')
plt.grid(True)
plt.scatter(y_test, svr_predictions)
plt.plot(y_test, y_test, color='red', label="identidade")
plt.legend(loc='upper center')
plt.show()
```

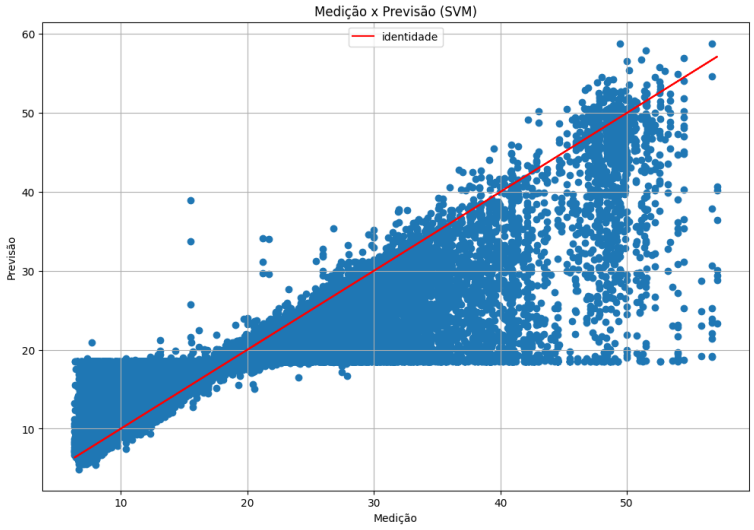

Figura 25. Medição vs. Previsão para SVM

3.4.7 Árvore de Decisão

Uma Árvore de Decisão é um único modelo de árvore que realiza previsões dividindo os dados em nós e folhas com base em regras de decisão. Cada nó

representa uma condição sobre um atributo do conjunto de dados, enquanto as folhas representam as previsões finais.

Importando a biblioteca:

```
from sklearn.tree import DecisionTreeRegressor
```

Assumindo que X6 e Y já estão definidos:

```
X6 = df_s[['PRESSAO CX ENT MIOLO', 'PRESSAO CX ENT VERSO', 'PRESSAO CX ENT CAPA', 'GRAMATURA SCANNER','ESPESSURA SCANNER','VELOCIDADE']].values
Y = df_s['RIGIDEZ LONGITUDINAL MEDIA'].values.reshape(-1, 1)
```

Separando os dados:

```
X_train, X_test, y_train, y_test = train_test_split(X6, Y, test_size = 0.3, random_state = 101)
```

Inicializando e treinando o modelo:

```
dt_regressor = DecisionTreeRegressor(random_state=101)
dt_regressor.fit(X_train, y_train)
```

Realizando as predições:

```
dt_predictions = dt_regressor.predict(X_test)
```

Avaliando o modelo:

```
print('Decision Tree - mean_squared_error : ', mean_squared_error(y_test, dt_predictions))
print('Decision Tree - mean_absolute_error : ', mean_absolute_error(y_test, dt_predictions))
print('Decision Tree - r2 : ', r2_score(y_test, dt_predictions))
```

```
Decision Tree - mean_squared_error :  3.1979
Decision Tree - mean_absolute_error :  0.6680
Decision Tree - r2 :  0.9636
```

Plotando o gráfico (Figura 26):

```
sct_x = y_test.reshape(1, -1)[0]
sct_y = dt_predictions

plt.figure(figsize=(12, 8))
plt.xlabel('Medição')
plt.ylabel('Previsão')
plt.title('Medição x Previsão - Decision Tree')
plt.grid(True)
plt.scatter(sct_x, sct_y)
```

```
plt.plot(sct_x, sct_x, color='red', label="identidade")
plt.legend(loc='upper center')
plt.show()
```

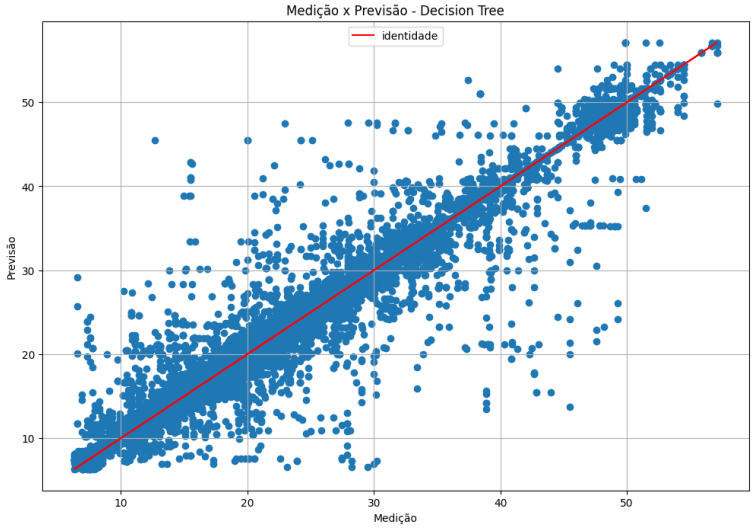

Figura 26. Medição vs. Previsão para Árvore de Decisão

3.4.8 GBM (Gradient Boosting Machine)

Importando a biblioteca:

```
from sklearn.ensemble import GradientBoostingRegressor
```

Assumindo que X6 e Y já estão definidos:

```
X6 = df_s[['PRESSAO CX ENT MIOLO', 'PRESSAO CX ENT VERSO',
'PRESSAO CX ENT CAPA', 'GRAMATURA SCANNER','ESPESSURA
SCANNER','VELOCIDADE']].values
Y = df_s['RIGIDEZ LONGITUDINAL MEDIA'].values.reshape(-1, 1)
```

Separando os dados:

```
X_train, X_test, y_train, y_test = train_test_split(X6, Y, test_size=0.3,
random_state=101)
```

Inicializando e treinando o modelo:

```
gbm_regressor = GradientBoostingRegressor(random_state=101)
gbm_regressor.fit(X_train, y_train.ravel())
```

Realizando as predições:

```
gbm_predictions = gbm_regressor.predict(X_test)
```

Avaliando o modelo:

```
print('GBM - mean_squared_error : ', mean_squared_error(y_test,
gbm_predictions))
print('GBM - mean_absolute_error : ', mean_absolute_error(y_test,
gbm_predictions))
print('GBM - r2 : ', r2_score(y_test, gbm_predictions))

GBM - mean_squared_error :  3.5044
GBM - mean_absolute_error :  1.0700
GBM - r2 :  0.9601
```

Plotando o gráfico (Figura 27):

```
sct_x = y_test.reshape(1, -1)[0]
sct_y = gbm_predictions

plt.figure(figsize=(12, 8))
plt.xlabel('Medição')
plt.ylabel('Previsão')
plt.title('Medição x Previsão - Gradient Boosting Machine')
plt.grid(True)
plt.scatter(sct_x, sct_y)
plt.plot(sct_x, sct_x, color='red', label="identidade")
plt.legend(loc='upper center')
plt.show()
```

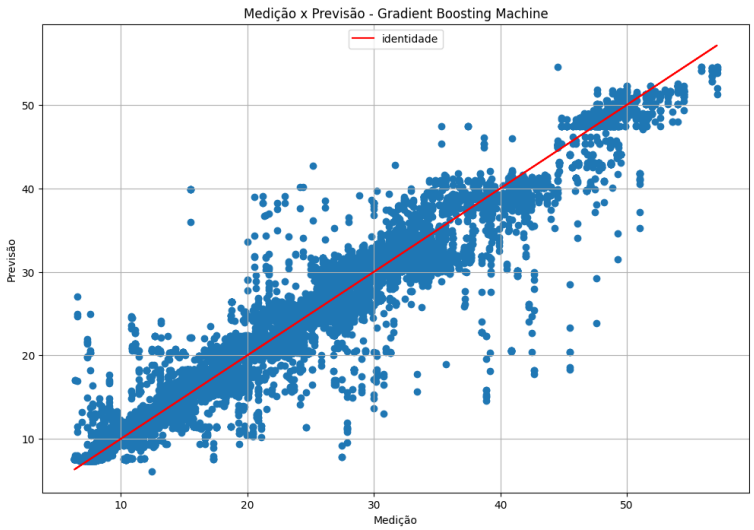

Figura 27. Medição vs. Previsão para Gradient Boosting Machine

3.5 Validação Cruzada

Nesta etapa realiza-se testes em múltiplos subconjuntos de dados, proporcionando uma estimativa mais confiável para seu desempenho geral e será aplicada para as modelagens Redes Neurais e Random Forest.

Existem dois tipos de validação cruzada. O K-Fold Cross-Validation divide os dados em k subconjuntos (folds) e em cada iteração, um fold é usado para teste e os restantes para treino, sendo adequado para dados aleatórios e não sequenciais, enquanto o Time Series Cross-Validation (ou Forward Chaining) é ideal para dados sequenciais ou temporais, dividindo os dados em conjuntos crescentes de treino e validação, mantendo a ordem temporal. Por este motivo, será implementada validação cruzada utilizando o Time Series Cross-Validation com o código abaixo.

```
import pandas as pd
import numpy as np
from sklearn.model_selection import TimeSeriesSplit
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
import tensorflow as tf
from tensorflow import keras
from sklearn.preprocessing import StandardScaler
```

Definindo as variáveis (x) e a variável alvo (y):

```
X = df_s[['PRESSAO CX ENT MIOLO', 'PRESSAO CX ENT VERSO', 'PRESSAO CX ENT CAPA', 'GRAMATURA SCANNER','ESPESSURA SCANNER','VELOCIDADE' ]].values
y = df_s['RIGIDEZ LONGITUDINAL MEDIA'].values
```

Aplicando a Validação Cruzada de Série Temporal

```
tscv = TimeSeriesSplit(n_splits=5)
```

Incluindo listas para armazenagem das métricas avaliadas:

```
rf_mse_scores = []
rf_mae_scores = []
rf_r2_scores = []

nn_mse_scores = []
nn_mae_scores = []
nn_r2_scores = []


for train_index, test_index in tscv.split(X):
    X_train, X_test = X[train_index], X[test_index]
    y_train, y_test = y[train_index], y[test_index]
        Random Forest

    rf_model = RandomForestRegressor(random_state=101)
    rf_model.fit(X_train, y_train)
```

```
    rf_predictions = rf_model.predict(X_test)
    rf_mse_scores.append(mean_squared_error(y_test, rf_predictions))
    rf_mae_scores.append(mean_absolute_error(y_test, rf_predictions))
    rf_r2_scores.append(r2_score(y_test, rf_predictions))

        Redes Neurais

    scaler = StandardScaler()
    X_train_scaled = scaler.fit_transform(X_train)
    X_test_scaled = scaler.transform(X_test)
    y_train_scaled = scaler.fit_transform(y_train.reshape(-1, 1))
    y_test_scaled = scaler.transform(y_test.reshape(-1, 1))


    nn_model = keras.Sequential([
        keras.layers.Dense(64, activation='relu', input_shape=(X_train.shape[1],)),
        keras.layers.Dense(32, activation='relu'),
        keras.layers.Dense(1)])
    nn_model.compile(optimizer='adam', loss='mse', metrics=['mae'])
    nn_model.fit(X_train_scaled, y_train_scaled, epochs=50, batch_size=32,
verbose=0)

    nn_predictions_scaled = nn_model.predict(X_test_scaled)
    nn_predictions = scaler.inverse_transform(nn_predictions_scaled)
    nn_mse_scores.append(mean_squared_error(y_test, nn_predictions))
    nn_mae_scores.append(mean_absolute_error(y_test, nn_predictions))
    nn_r2_scores.append(r2_score(y_test, nn_predictions))

        Plotando os resultados:

print("Random Forest:")
print("MSE:", np.mean(rf_mse_scores))
print("MAE:", np.mean(rf_mae_scores))
print("R^2:", np.mean(rf_r2_scores))

print("\nNeural Network:")
print("MSE:", np.mean(nn_mse_scores))
print("MAE:", np.mean(nn_mae_scores))
print("R^2:", np.mean(nn_r2_scores))
```

Random Forest:
MSE: 8.6366
MAE: 1.8555
R^2: 0.8791

Neural Network:
MSE: 7.0329
MAE: 1.6547
R^2: 0.9016

4 RESULTADOS E DISCUSSÃO

Após a mineração de dados, a interpretação dos resultados é essencial. A avaliação é feita para garantir que os padrões encontrados sejam válidos e úteis para os objetivos da empresa.

No contexto do KDD, a fase de interpretação pode ser vista como a finalização do pipeline de dados, assim como a fase de carregamento do ETL, onde os dados são armazenados em um banco de dados para visualização, relatórios ou modelos analíticos.

Na Figura 28, observa-se a consolidação dos resultados obtidos em uma tabela para melhor comparação entre os modelos.

Comparativo Modelos	MSE	MAE	r^2
Regressão Linear - 3 Variáveis	7,6486	1,8907	0,9130
Regressão Linear - 6 Variáveis	6,2482	1,6567	0,9290
Random Forest	1,6976	0,5669	0,9807
KNN (K=5)	1,7161	0,5858	0,9805
Redes Neurais	0,035	0,1038	0,9668
SVM	25,5277	3,2392	0,7095
Árvore de Decisão	3,199	0,668	0,9636
GBM	3,5044	1,0700	0,9601
PCA - Regressão Linear	15,9887	2,8509	0,8181
PCA - Random Forest	5,1657	1,2695	0,9412
PCA - KNN	4,9517	1,2499	0,9437

Figura 28. Tabela Comparativa dos Modelos

Com base na tabela de resultados fornecida, observamos que os modelos de Random Forest, KNN (com K=5) e Redes Neurais se destacam pela alta precisão, apresentando valores de r^2 acima de 0,96 e erros (MSE e MAE) relativamente baixos. Em particular, o modelo de Redes Neurais obteve um desempenho excepcional com o menor MSE (0,035) e MAE (0,1038), sugerindo uma excelente capacidade de predição com alta acurácia para a variável alvo. Esses resultados indicam que esse modelo captura bem a relação entre as variáveis de processo e a rigidez do papelcartão, o que pode ser valioso para prever a rigidez em tempo real.

Por outro lado, o modelo SVM apresentou o menor desempenho, com um MSE e MAE elevados e um r^2 significativamente inferior (0,7095), sugerindo que esse método pode não ser adequado para este conjunto de dados. A inclusão de técnicas de redução de dimensionalidade (PCA) melhorou a performance de alguns modelos, mas de maneira geral, as melhores performances foram obtidas com modelos não-linearizados sem PCA. Esses resultados apontam para a viabilidade do uso de modelos de aprendizado supervisionado, especialmente o Random Forest e as Redes Neurais, para a predição eficiente e precisa da rigidez do papelcartão, possibilitando um controle mais dinâmico e contínuo do processo produtivo.

Um período dentro do intervalo de dados foi escolhido para apresentar em um gráfico (Figura 29), o comportamento da variável alvo estimada em relação à RIGIDEZ LONGITUDINAL MÉDIA medida em laboratório. O intervalo do período entre 05:21h e 19:36h do dia 08 de Junho de 2024 mostra oscilações na Rigidez (Linha Roxa – “Yest Long”) que não são capturadas pelas medições de Rigidez do Laboratório de Qualidade (Linha Azul Claro – “RIGIDEZ LONGITUDINAL MEDIA”), inclusive com valores abaixo do Limite Inferior de Especificação de Produto (“L Inf” representado pela Linha Vermelha Tracejada).

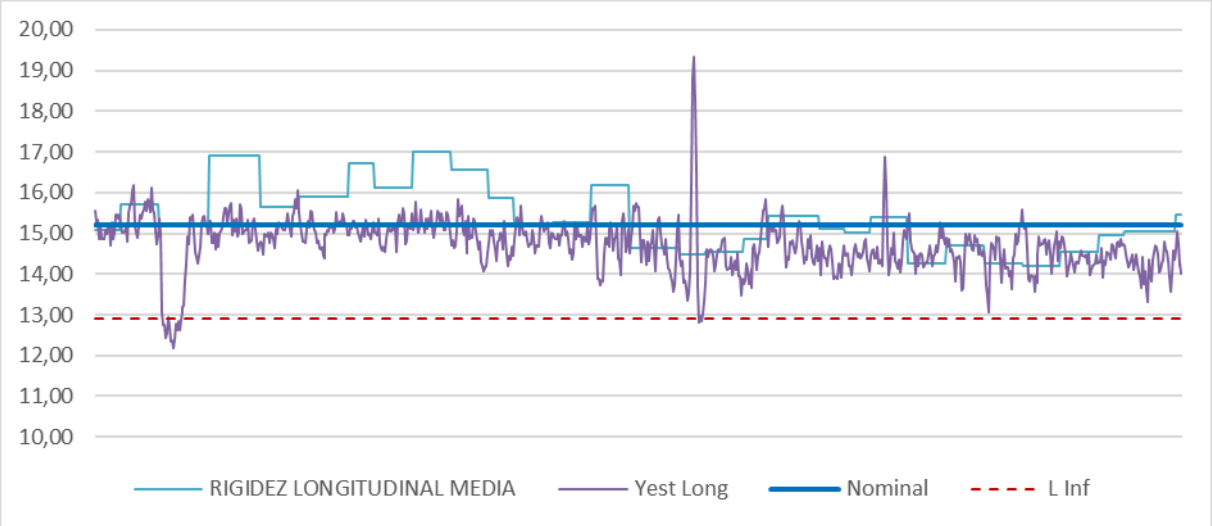

Figura 29. Rigidez Longitudinal Estimada vs. Rigidez Medida

Um comportamento similar pode ser observado no intervalo entre 15:13h e 23:39h do dia 16/08/2024 mostrado na Figura 30 abaixo, porém com um Yest (Rigidez Estimada) bem próximo do limite mínimo de especificação, o que possibilita alertas

aos colaboradores da operação para correção de parâmetros que levem a Rigidez mais próxima ao valor nominal para esta categoria de produto.

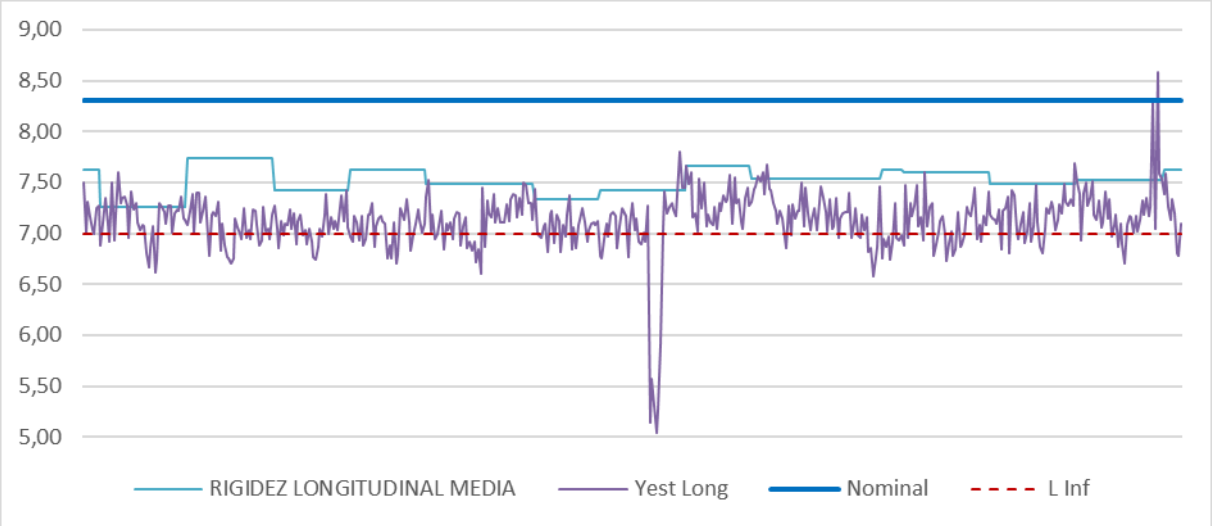

Figura 30. Rigidez Longitudinal Estimada vs. Rigidez Medida

A aplicação da validação cruzada mostrou como resultados do Random Forest um MSE 8.6366, MAE 1.8555 e r^2 0.8791 e para Redes Neurais um MSE 7.0329, MAE 1.6547 e r^2 0.9016.

Para comparar estes resultados e definir qual é o melhor modelo para predição da Rigidez do Papelcartão, calcula-se a média do desvio padrão das métricas obtidas de cada split, escrito no código seguinte:

```
print("\nRandom Forest Standard Deviations:")
print("MSE:", np.std(rf_mse_scores))
print("MAE:", np.std(rf_mae_scores))
print("R^2:", np.std(rf_r2_scores))

print("\nNeural Network Standard Deviations:")
print("MSE:", np.std(nn_mse_scores))
print("MAE:", np.std(nn_mae_scores))
print("R^2:", np.std(nn_r2_scores))

rf_avg_r2 = np.mean(rf_r2_scores)
rf_std_r2 = np.std(rf_r2_scores)

nn_avg_r2 = np.mean(nn_r2_scores)
nn_std_r2 = np.std(nn_r2_scores)

print("\nModel Comparison:")
print("Random Forest - Average R^2:", rf_avg_r2, ", Standard Deviation:", rf_std_r2)
```

```
print("Neural Network - Average R^2:", nn_avg_r2, ", Standard Deviation:",
nn_std_r2)
```

```
Random Forest Standard Deviations:
MSE: 3.3640
MAE: 0.3421
R^2: 0.0659

Neural Network Standard Deviations:
MSE: 2.5472
MAE: 0.2345
R^2: 0.0507

Model Comparison:
Random Forest - Average R^2: 0.8791, Standard Deviation: 0.0659
Neural Network - Average R^2: 0.9016, Standard Deviation: 0.0507
```

E por fim, a decisão de qual é o melhor modelo, baseado na maior média de r^2 e menor desvio padrão:

```
if rf_avg_r2 > nn_avg_r2 and rf_std_r2 < nn_std_r2:
    print("\nRandom Forest apresentou melhor performance.")
elif nn_avg_r2 > rf_avg_r2 and nn_std_r2 < rf_std_r2:
    print("\nRedes Neurais apresentou melhor performance.")
else:
    print("\nSelecao do modelo ambigua.")
```

```
Redes Neurais apresentou melhor performance.
```

5 CONCLUSÃO

O conhecimento adquirido nas disciplinas “Análise e Mineração de Dados” e “Análises Preditivas” foi essencial para o desenvolvimento deste trabalho que com a aplicação de princípios e métodos de sistemas de previsão integrados à análise de dados em tempo real e aprendizado de máquina, proporcionará melhoria significativa no produto e no processo alvo.

Este trabalho investigou a predição da rigidez longitudinal média do papelcartão utilizando diversas técnicas de aprendizado de máquina. Foram avaliados modelos de Regressão Linear, Random Forest, KNN, Redes Neurais, Support Vector Regression (SVR), Árvore de Decisão e Gradient Boosting Machine, com e sem aplicação de PCA, e com diferentes métricas de avaliação (MSE, MAE, R^2). Os resultados demonstraram que o desempenho dos modelos variou, sendo que o Random Forest e as Redes Neurais apresentaram os melhores resultados de R^2, e a validação cruzada por séries temporais confirmou a robustez dos modelos selecionados.

A comparação final, com base na média do R^2 e desvio padrão dos resultados obtidos via validação cruzada, permite concluir que o modelo mais robusto para a predição de valores de rigidez do papelcartão, levando em consideração tanto a acurácia quanto a estabilidade da predição é o modelo Redes Neurais.

Este algorítimo aplicado ao ambiente fabril, possibilita rápida reação dos operadores de máquina frente a problemas de qualidade iminentes, antes que atinjam o limite mínimo de especificação da rigidez do papelcartão por ter uma visualização em tempo real, sem ter que aguardar resultado de análise em laboratório.

Mesmo que um problema aconteça, como visto nas figuras 29 e 30, é possível fazer marcações na bobina de papelcartão e retirar este material abaixo do limite mínimo de especificação dos processos subsequentes, garantindo que o cliente receba apenas produtos com rigidez conforme especificação técnica.

6 REFERÊNCIAS BIBLIOGRÁFICAS

ROBUSTI, Célio; VIANA, Eder Francisco; FERREIRA JÚNIOR, Fernando; GOMES, Ilduardo; TOGNETTA, Laudo; SANTOS, Osni dos; DRAGONI, Paulo. *Celulose e Papel*. São Paulo: SENAI-SP, 2014.

FACELI, Katti; LORENA, Ana Carolina; GAMA, João; CARVALHO, André C. P. L. F. de. Inteligência Artificial: uma abordagem de aprendizado de máquina. 1. ed. Rio de Janeiro: LTC, 2011.

GUTMAN, Per-Olof; NILSSON, Bengt. Modelling and prediction of bending stiffness for paper board manufacturing. 1997. Disponível em: https://www.sciencedirect.com/science/article/pii/S095915249700036X. Acesso em: 21/11/2023.

HU, Xiaolei; ZHANG, Xue. A Novel Method for Product Quality Control Based on Production IoT Data. 2022. Disponível em: https://ieeexplore.ieee.org/document/10080209. Acesso em: 21/11/2023.

PETTERSON, Jens; GUTMAN, Per-Olof; BOHLIN, Torsten; NILSSON, Bengt. A Grey Box Bending Stiffness Model for Paper Board Manufacturing. 1997. Disponível em: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=627590. Acesso em: 21/11/2023.

ZHANGL, Yingfeng; LIU, Sichao; SIL, Shubin; YANG, Haidong. Production system performance prediction model based on manufacturing big data. 2015. Disponível em: https://ieeexplore.ieee.org/document/7116048. Acesso em: 21/11/2023.

DU, WenBo; ZHU, Zhixiang; WANG, Chuang; YUE, Zhifeng. The Real-time Big Data Processing Method Based on LSTM for the Intelligent Workshop Production Process. 2020. Disponível em: https://ieeexplore.ieee.org/document/9101345. Acesso em: 21/11/2023.

CUI, Yiming. Intelligent Optimization Prediction and Application Based on Statistical Analysis and Data Mining. 2023. Disponível em: https://ieeexplore.ieee.org/document/10213133. Acesso em: 21/11/2023.

GÖRTZ, Morgan; GUSTAV, Kettil; MÅLQVIST, Axel; FREDLUND, Mats; WESTER, Kenneth; EDEVIK, Fredrik. Network model for predicting structural properties of paper. 2022. Disponível em: https://www.degruyter.com/document/doi/10.1515/npprj-2021-0079/html. Acesso em: 21/11/2023.

VIEIRA, Osvaldo. Prediction of paperboard properties with virtual on-line solutions. 2003. Disponível em: https://imisrise.tappi.org/TAPPI/Products/03/JUL/03JULOE02.aspx. Acesso em: 21/11/2023.

ŽAPČEVIĆ, Seid; BUTALA, Peter. Adaptive process control based on a self-learning mechanism in autonomous manufacturing systems. 2013. Disponível em: https://link.springer.com/article/10.1007/s00170-012-4453-0. Acesso em: 21/11/2023.

CHOUDHARY, A. K.; HARDING, J. A.; TIWARI, M. K. Data mining in manufacturing: a review based on the kind of knowledge. Disponível em: https://link.springer.com/article/10.1007/s10845-008-0145-x. Acesso em: 21/11/2023.