Universidade de Brasília
Faculdade de Tecnologia

Estudo exploratório da aplicação de máquinas de vetores de suporte na predição do lead time de fabricação em indústria do ramo mecatrônico

Pedro Victor Galieta Tomaz

TRABALHO DE GRADUAÇÃO

ENGENHARIA DE CONTROLE E AUTOMAÇÃO

Brasilía

2022

Universidade de Brasília
Faculdade de Tecnologia

Estudo exploratório da aplicação de máquinas de vetores de suporte na predição do lead time de fabricação em indústria do ramo mecatrônico

Pedro Victor Galieta Tomaz

Trabalho de Graduação submetido como requisito parcial para obtenção do grau de Engenheiro de Controle e Automação.

Orientador: Prof. Dr. Carlos Humberto Llanos
Coorientador: Prof. Dr. Sanderson César Macedo Barbalho

Brasília
2022

Galieta Tomaz, Pedro Victor.
G156e Estudo exploratório da aplicação de máquinas de vetores de suporte na predição do lead time de fabricação em indústria do ramo mecatrônico / Pedro Victor Galieta Tomaz; orientador Carlos Humberto Llanos; coorientador Sanderson César Macedo Barbalho. -- Brasilía, 2022.
91 p.

Trabalho de Graduação em Engenharia de Controle e Automação -- Universidade de Brasília, 2022.

1. *Lead-time* de fabricação. 2. aprendizagem de máquina. 3. máquinas de vetores de suporte. 4. modelo preditivo. I. Llanos, Carlos Humberto, orient. II. Macedo Barbalho, Sanderson César, coorient. III. Título

Universidade de Brasília
Faculdade de Tecnologia

Estudo exploratório da aplicação de máquinas de vetores de suporte na predição do lead time de fabricação em indústria do ramo mecatrônico

Pedro Victor Galieta Tomaz

Trabalho de Graduação submetido como requisito parcial para obtenção do grau de Engenheiro de Controle e Automação.

Trabalho aprovado. Brasilía, 11 de maio de 2022:

Prof. Dr. Carlos Humberto Llanos, UnB/FT/ENM
Orientador

Prof. Dr. Sanderson César Macedo Barbalho, UnB/FT/EPR
Co-Orientador

Prof. Dr. Carlos Eduardo da Silva Santos, IFTO
Examinador externo

Prof. Dr. Eugênio Libório Feitosa Fortaleza, UnB/FT/ENM
Examinador interno

Brasilía
2022

Este trabalho é dedicado à minha família. Em especial à minha mãe, Patrícia, por quem me faltam palavras para expressar gratidão e admiração.

Agradecimentos

Primeiramente, gostaria de agradecer aos professores que, durante toda a graduação, transmitiram o conhecimento que tanto custaram para acumular. Em especial, ao professor Llanos e Sanderson que acompanharam de perto essa etapa tão importante da minha formação e aceitaram esse desafio comigo, demonstrando muita paciência e compreensão. Ao professor Carlos Eduardo eu agradeço pelo seu empenho e entusiasmo em ensinar e compartilhar, e por ter sido grande apoio durante esse processo.

Aos meus amigos, agradeço por sempre terem estado ao meu lado. Em alguns momentos, estudando madrugada adentro e, em outros, dividindo momentos de lazer que amenizam as dificuldades da graduação e deixam tantas memórias.

Dito isso, agradeço à minha família, que me deu suporte em todos os níveis imagináveis para que meu estudo fosse possível. Foram eles que acompanharam os momentos mais complicados dessa caminhada e não me deixaram desistir. Espero que um dia, de alguma forma, eu possa retribuir pelo menos uma parcela desse cuidado.

”It is our choices that show what we truly are, far more than our abilities”
(J.K. Rowling)

Resumo

A predição do *Lead-Time* de fabricação permite alocação estratégica de recursos materiais e humanos. Portanto, vem se tornando desafio comum às empresas em vista do cenário amplamente competitivo do mercado contemporâneo. O presente trabalho, avalia a viabilidade de modelar o *Lead-Time* em empresa de produção mecatrônica da área médica e espacial segundo base de dados de produção. Abordou-se o problema por meio de regressão via *Support Vector Machines* (SVM), utilizando a ferramenta NIOTS que implementa meta-heurísticas para ajuste dos hiper-parâmetros a partir de um problema de otimização multi-objetivo. Aos modelos encontrados, foram aplicadas métricas de avaliação e, com base nelas, foram propostas novas abordagens para a fase de pré-processamento de dados, buscando maximizar a capacidade de generalização. O resultado obtido com as SVM foi comparado a implementações de regressão linear múltipla e *random forest*. Ao fim do trabalho, atingiu-se uma marca de *Mean Average Percentage Error* (MAPE) correspondente a 29%, 35% e 17% para os três algoritmos respectivamente. Adicionalmente, propôs-se modelo flexível que permitisse ajuste de predições otimistas ou pessimistas implementado através de manipulação da função de perda ϵ-insensitiva das máquinas de vetores de suporte.

Palavras-chave: *Lead-time* de fabricação. aprendizagem de máquina. máquinas de vetores de suporte. modelo preditivo.

Abstract

Predicting Production Lead-Time is vital for allocation of both material and human resources, therefore becoming a challenge to companies worldwide due to the recent competitive Market. The proposed work evaluates viability on modelling Lead-Time for a mechatronic production company in medical and space equipment area using a manufacturing database. The problem was approached by regression based on Support Vector Machines (SVM) using the NIOTS tool, that implements two meta-heuristics to adjust hyper-parameters through a multi-objective optimization problem. Evaluation metrics were applied to each model and led to proposal of new methodologies in pre processing stage trying to maximize model accuracy. The SVM results were compared to alternative approaches implemented by multiple linear regression and random forests achieving Mean Average Percentage Error (MAPE) of 29%, 35% and 17% respectively. Moreover, the present work also proposed a flexible model capable of optimistic or pessimistic predictions based on a manipulation of support vector machines ϵ-insensitive loss function.

Keywords: Manufacturing lead-time. machine learning. Support Vector Machines. predictive model.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

APMT-MODE	Adaptive Parameter with Mutant Tournament Multi-Objective Differential Evolution . . . 18
CPU	Central Processing Unit . . . 74
KKT	Karush-Kuhn-Tucker . . . 26
MAPE	Mean Absolute Percentage Error . . . 19
MODE	Multi Objective Differential Evolution . . . 50
MSE	Mean Squared Error . . . 19
NIOTS	Nature inspired optimization tools for SVMs . . . 18
RBF	Radial Basis Function . . . 50
RNA	Redes Neurais Artificiais . . . 32
RSS	Residual Sum of Squares . . . 33
SV	Support Vectors . . . 32
SVM	Support Vector Machines . . . 18
SVR	Support Vector Regressors . . . 30
TSS	Total Sum of Squares . . . 33
VC	Vapnik Chervonenkis . . . 23

Sumário

1 Introdução

1.1 Contextualização

A Quarta Revolução Industrial, popularmente conhecida por Indústria 4.0, carrega o compromisso de viabilizar processos de manufatura flexíveis, otimizando qualidade e produtividade paralelamente à customização em massa. Para tal, a manufatura inteligente transforma recursos comuns em objetos inteligentes aptos a atuar e ajustar seu comportamento em um ambiente altamente integrado. Dessa forma, habilita-se o desenvolvimento de produtos cada vez mais individualizados e com menor *lead-time* (ZHONG et al., 2017).

Nesse contexto, a nova era de produção passa a ser baseada em princípios de instrumentação, modelagem de tomada de decisão, sistemas inteligentes, análise de dados e materiais avançados (LI et al., 2017). Portanto, o desenvolvimento das indústrias inteligentes é atrelado ao avanço tecnológico em áreas como integração homem-máquina, operações remotas, *Internet of Things* (IoT), *big data*, manufatura aditiva, computação em nuvem e sistemas ciber-físicos (KHAN et al., 2017).

Outrossim, sistemas de manufatura preditiva recebem um enfoque no cenário de Indústria 4.0, uma vez que desempenham funções de auto-previsão e auto-manutenção. Com base em técnicas de estatística, mineração de dados, modelagem e inteligência artificial, realizam previsões de estados futuros e estimam, antecipadamente, uma atitude ou decisão. Assim, diminuem-se medidas reativas tomadas pela empresa (NIKOLIC et al., 2017). Dentre as implementações fundamentadas pela inteligência artificial, aprendizagem de máquina vem apresentando resultados promissores nesse âmbito, a exemplo da manutenção preditiva (SUSTO et al., 2015).

Uma possível definição para *lead-time*, adotando a perspectiva do cliente, é o tempo de entrega para uma atividade ou produto (MOURTZIS et al., 2014). Complementarmente, Silva e Fernandes (SILVA, F.; FERNANDES, 2008) segmentam esse tempo em: *lead-time* de (a) projeto, (b) cadeia de suprimentos, (c) fabricação, (d) montagem, (e) distribuição.

Naturalmente, estimativas de *lead-time*, tempo de desenvolvimento e de fabricação se tornam vitais no contexto da Indústria 4.0, já que impactam diretamente gestão e custos de produção (BASHIR, 2008). Uma vez que o mercado converge à redução do ciclo de vida do produto, a predição do *lead-time* se relaciona com o possível sucesso dos prazos de introdução no mercado. Logo, uma correta predição é ferramenta relevante para a gestão de produto, conferindo vantagem competitiva à empresa. Contudo, em geral, o desenvolvimento de novos produtos é impactado por falhas de cronograma e estimativas sub-ótimas de *lead-time* (JUN; PARK; SUH, 2006).

1.2 Descrição do Problema

Estudos recentes abordaram implementações de métodos analíticos, empíricos e heurísticos para estimar o *lead-time* de produção, a exemplo do **raciocínio baseado em caso** (MOURTZIS et al., 2014), ***support vector machines*** (COS JUEZ et al., 2010) e **redes bayesianas** (MORI; MAHALEC, 2015). Com base em (MOURTZIS et al., 2014) e (COS JUEZ et al., 2010), é razoável concluir que abordagens envolvendo técnicas de inteligência artificial tendem a apresentar resultados satisfatórios.

Nesse cenário, o presente trabalho se propõe a investigar nova abordagem para modelagem do *lead-time*. O trabalho foi desenvolvido a partir dos dados de produção de empresa brasileira que atua com foco nas nas áreas médica, industrial, aeroespacial e de defesa, adotando os modelos de produção *Make to Order* (MTO) e *Engineering to Order* (ETO).

Em 2008, Barbalho e Torres, inseridos no contexto da referida empresa, empreenderam estudo para otimizar os fluxos de documentos e materiais focando no aumento da capabilidade dos processos (BARBALHO; TORRES, 2008). Com base em seguidas iterações da equipe, as gerências de engenharia e manufatura definiram o processo logístico sintetizado no fluxograma da Figura 1, cujas abreviações representam (BARBALHO; TORRES, 2008):

1. **UENG:** Unidade de engenheiros responsáveis pelo projeto, que acompanham, junto ao escritório, o andamento da fabricação por meio de reuniões semanais.
2. **UGD:** Unidade de Gerenciamento e Documentação, que estabelece interface entre os setores de engenharia e manufatura.
3. **UMONT:** Unidade de Montagem.
4. **G. IND:** Gerência Industrial.
5. **COMPRAS:** Departamento de compras.
6. **ALMOX.:** almoxarifado.
7. **FORNEC.:** fornecedor.
8. **DSGEO:** Departamento de Sistemas de Gestão.
9. **T. SUPER:** setor de tratamento superficial.
10. **FABRIC.:** setor de fabricação.

Com base no fluxograma da Figura 1, as gerências sistematizaram uma planilha de rastreamento para as peças produzidas, reduzindo métodos particulares de controle e grande

fluxo de informações via e-mail. Assim, passaram a ser registrados dados de engenharia, tais quais: código da peça, projeto ao qual pertence, data em que a UGD recebeu o desenho, tipo de fabricação, material, quantidade, data que o desenho será passado para a Gerência Industrial, etc. Após todo o *pipeline* descrito na Figura 1, a UGD insere na base de dados a data de recebimento da peça, gerando indicadores de desempenho de prazos. Ou seja, *lead-times* gerais e intermediários passaram a ser monitorados (BARBALHO; TORRES, 2008).

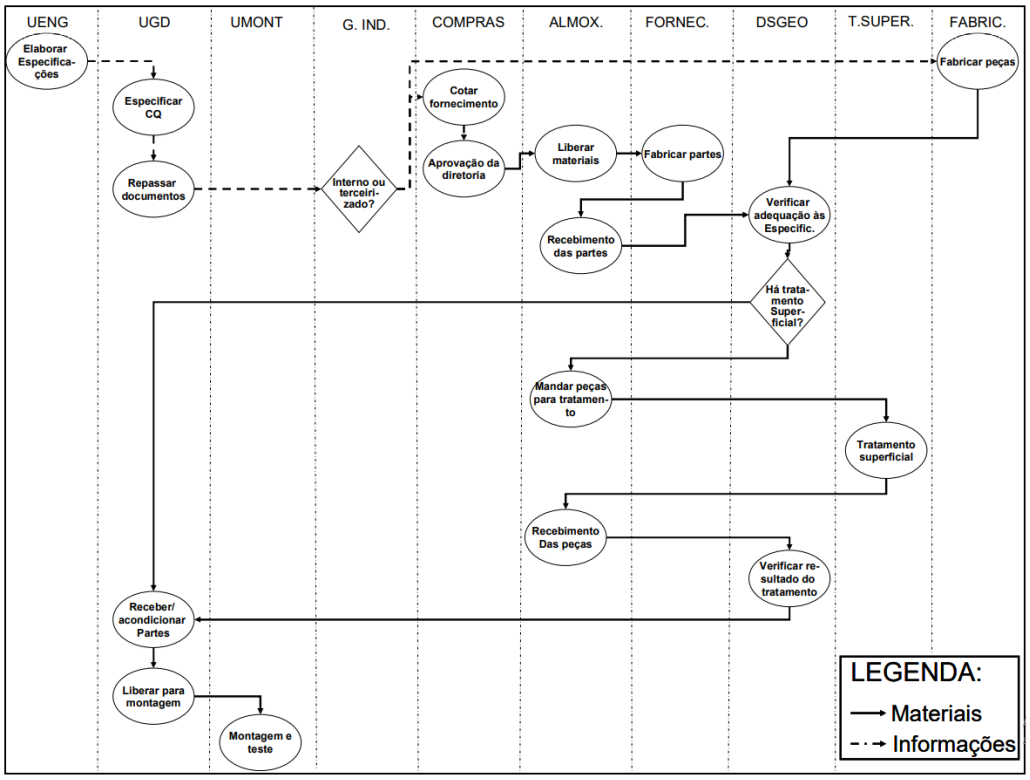


Figura 1 – Processo de confecção de protótipos (BARBALHO; TORRES, 2008)

Dessa forma, pode-se verificar a estrutura de uma base de dados própria para treinamento supervisionado: as informações de engenharia correspondem às *features* e o *lead-time* do componente corresponde ao valor-alvo da predição. A referência (ROSA, 2020) realiza trabalho utilizando um registro de produção dessa mesma empresa de 2006 a 2012 no formato descrito e enuncia, em forma de perguntas, os seguintes problemas de pesquisa:

1. Como os métodos de aprendizagem de máquina podem ser úteis na estimativa do *lead-time* de fabricação de componentes para protótipos em uma empresa de base tecnológica com foco no desenvolvimento de produtos mecatrônicos da área médica e espacial?

2. De que forma é possível utilizar as ordens de serviço de uma empresa com foco no desenvolvimento de produtos mecatrônicos para predizer o tempo de fabricação de componentes dessa empresa estudo de caso?

No entanto, optou-se por uma abordagem distinta daquela utilizada em (ROSA, 2020). Primeiramente, o foco desse trabalho foi a modelagem do problema a partir de regressão - em vez de classificação - por meio de Máquinas de Vetores de Suporte (SVM). Para o ajuste dos hiperparâmetros do modelo, processo inerente à aprendizagem de máquina, aplicou-se a ferramenta NIOTS (LLANOS; SILVA SANTOS; DOS SANTOS, 2021) que utilliza o método de Parâmetros Adaptativos com Torneio Multiobjetivo Evolução Diferencial (APMT-MODE) (SANTOS, 2019). Para estabelcer critério de comparação às SVM, algoritmos alternativos foram propostos, sendo eles regressão linear múltipla e *random forest*.

1.3 Hipóteses

Tendo problema e contexto definido, a **primeira hipótese** do presente trabalho investiga a aplicação de um método para ajuste ótimo dos hiperparâmetros na modelagem do *lead time* por meio de regressão. Para essa verificação, propõe-se a utilização da ferramenta NIOTS (LLANOS; SILVA SANTOS; DOS SANTOS, 2021), que implementa meta-heurísticas para ajuste dos hiperparâmetros da SVM por meio de um problema de otimização multi-objetivo.

Em seguida, considerando um conjunto de dados extremamente complexo a ser analisado, a **segunda hipótese** define investigação de metodologia para simplificar essa análise. De forma geral, esse processo deve ser realizado através da seleção de rótulos dentre as variáveis mais relevantes.

Além disso, considerou-se a necessidade de comparar os resultados obtidos no treinamento das máquinas de vetores de suporte. Portanto, propõe-se implementação de outros dois modelos de aprendizagem de máquina que tenham princípio matemático ou estatístico distinto das SVM e sejam, ao mesmo tempo, relevantes no contexto de análise de dados. Assim, a **terceira hipótese** avalia a eficácia da abordagem desse problema por SVMs.

Por fim, considera-se um cenário em que modelos de previsão de *lead time* possam ser empregados no meio corporativo para tomada de decisão. Dessa forma, a **quarta hipótese** avalia o desenvolvimento de modelo flexível que tenha margem de ajuste para predição pessimista ou otimista do *lead time*.

1.4 Objetivos

Os objetivos do trabalho se dividem entre **objetivo geral**, que sintetiza o foco do estudo empreendido, e **objetivos específicos**, etapas intermediárias que devem ser cumpridas para que se possa atingir o objetivo geral.

1.4.1 Objetivo geral

O objetivo geral é o estudo da aplicação de máquinas de vetores de suporte através da ferramenta NIOTS na predição do *lead time* de uma empresa de produção mecatrônica.

1.4.2 Objetivos específicos

Destacam-se os seguintes:

- familiarização com a base de dados, compreendendo as variáveis que a compõem e sua contextualização no âmbito industrial;
- familiarização com a ferramenta, abrangendo: formato dos dados de entrada, comandos necessários, formato e salvamento dos dados de saída;
- pré processamento e ajuste das variáveis para padrão compatível com o NIOTS;
- divisão do *dataset* a ser utilizado, considerando a priorização de determinadas variáveis;
- implementação da métrica de avaliação MAPE e de histograma, ambos para facilitar a visualização dos resultados por parte de gestores de produção;
- execução da ferramenta para cada um dos casos de testes propostos;
- implementação de algoritmos alternativos para comparação com as SVM;
- desenvolvimento de modelo flexível para tomada de decisão em meio corporativo;
- avaliação dos resultados obtidos.

1.5 Contribuições da proposta

O estudo tem como principal contribuição a definição de uma metodologia que permita aplicar a ferramenta NIOTS para modelar o *lead time* de produtos em uma empresa de produção mecatrônica.

Assim, destaca-se como primeira contribuição do trabalho a aplicação do NIOTS (LLANOS; SILVA SANTOS; DOS SANTOS, 2021) em contexto industrial. Os testes realizados com a ferramenta tratam de conjuntos de dados estudados no meio acadêmico, denominados *benchmarks*, e também de problemas de pesquisa no ambiente universitário do autor. Portanto, seria de extrema valia aplicar e validar seu funcionamento com circunstâncias e objetivos distintos.

Em seguida, uma etapa essencial para realização do trabalho e portanto, uma contribuição, deve ser a adequação dos dados tabulares para formato próprio à aprendizagem de máquina, incluindo codificação de rótulos.

Além disso, a ferramenta e as meta-heurísticas por ela empregadas são, seguramente, um grande avanço para esse tópico de estudo. Logo, contribuir para sua otimização, divulgação e reconhecimento tende a gerar cada vez mais frutos e resultados em trabalhos futuros.

Por fim, ressalta-se que o problema abordado nesse estudo, a predição de produção, é, indubitavelmente, uma das grandes dores do mercado hoje. Portanto, desenvolver trabalho que promova essa discussão e apresente alternativa de solução com recursos inovadores é, também, contribuição desse estudo.

1.6 Estrutura do texto

O texto foi dividido em cinco capítulos com esse incluso, que trata da **Introdução** do trabalho. Em seguida, o Capítulo 2 apresenta **Aspectos teóricos das ferramentas utilizadas**, isto é, trata também de questões introdutórias mas definindo arcabouço teórico e técnico dos recursos aplicados ao projeto. A exemplo, citam-se os algoritmos utilizados e as métricas de avaliação comumente aplicadas.

O Capítulo 3 descreve a **Metodologia**, isto é, todas as etapas que foram seguidas para a realização do trabalho, embasando e justificando as tomadas de decisão ao longo do processo. Ao fim dele, com os resultados disponíveis, o Capítulo 4 apresenta **Análise dos resultados** comparando os diferentes modelos obtidos para as diferentes divisões realizadas no *dataset*.

Por fim, o Capítulo 5 trata da **Conclusão** que apresenta breve apanhado da proposta, do metodologia desenvolvida e dos resultados obtidos. Com isso, realizam-se considerações a respeito do trabalho e do que pode ser realizado no futuro.

2 Aspectos teóricos das ferramentas utilizadas

2.1 Aprendizagem de máquina

De modo geral, aprendizagem de máquina explora o reconhecimento de padrões em um conjunto de amostras, possibilitando a identificação de um sistema sintetizada em um modelo. Dessa forma, podem ser obtidas soluções generalistas, ou seja, cujo resultado previsto pelo modelo aproxima satisfatoriamente valores reais para um determinado conjunto de entrada.

Esse processo geralmente se divide em duas fases. Primeiramente, o mapeamento entre entradas e saídas é realizado por meio de um algoritmo com etapas definidas e recebe o nome de **treinamento**. Em seguida, valores distintos de entrada são fornecidos ao modelo para avaliação de sua capacidade de generalização, se dando a fase de **teste**. Em geral, os procedimentos de treinamento e teste são realizados, respectivamente, com 60-90% e 10-40% do conjunto de dados disponível (SILVA, I. et al., 2017).

A forma como o algoritmo processa os dados determina como será o processo de aprendizagem, estabalecendo três principais métodos de treinamento: **supervisionado**, **não supervisionado** e por **reforço**.

O supervisionado é o mais antigo deles, tendo sido proposto em 1949 por Donald Hebb. Nesse caso, tanto entradas quanto saídas do sistema estão disponíveis para o treinamento. Isto é, ocorre um processo de indução em que as variáveis internas do modelo tentam se ajustar de forma a atingir os resultados esperados. Em termos mais simplistas, é como se houvesse um "professor"guiando o treinamento.

Contrariamente, o treinamento não supervisionado trata somente dos valores de entrada de um sistema. Nesse caso, identificam-se características comuns aos elementos que definem subconjuntos. Ou seja, as variáveis internas do modelo são ajustadas de forma a agrupar amostras de acordo com sua similaridade (*clusterização*). Em geral, esse treinamento requer que o projetista tenha conhecimento prévio do sistema.

Por fim, o treinamento por reforço, proposto em 1998 por Sutton e Barto, é considerado uma variação do supervisionado. No entanto, o processo de aprendizagem não ocorre com base nos padrões entre entrada e saída mas é construído a partir de experiências. Para tal, o modelo pode ser visto como um agente autônomo que explora o ambiente. Cada nova entrada que o leva mais próximo da saída esperada gera uma recompensa. Caso contrário, o modelo é penalizado e, com o passar das iterações, "aprende" quais decisões não devem ser

repetidas (SANTOS, 2019)(SILVA, I. et al., 2017).

2.2 Máquinas de Vetores de Suporte (SVM)

2.2.1 Introdução

Quando se trata de funções classificadores, (SANTOS, 2019) explica que sua eficiência pode ser avaliada através de dois fatores. Primeiramente, a **capacidade de generalização**, como dito anteriormente, indica a capacidade do modelo em classificar dados que não pertenciam ao conjunto de treinamento. Por outro lado, a **complexidade** diz respeito aos elementos necessários à composição da função, como os pontos de treinamento das SVM, os centróides das RBF ou ainda as árvores de uma *random forest*.

Idealmente, busca-se maximizar a capacidade de generalização e minimizar a complexidade. No entanto, é natural observar que esses objetivos são contraditórios, e assim, aplicações reais demandam que haja equilíbrio entre os dois fatores.

Nesse contexto, as Máquinas de Vetores de Suporte (SVM), são modelos gerados a partir de aprendizagem supervisionada com um conjunto de treinamento composto por $X = (x_i, y_i)$, $i = 1,2,...,N$ tal que $x_i \in \mathbb{R}^n$ e $y_i \in \{-1, 1\}$, sendo x_i o vetor de amostras, y_i os rótulos (classes) e N a cardinalidade de X.

Uma vez realizado o treinamento, a função $f(x, \alpha)$ denota uma máquina obtida a partir de X, em que α é o ajuste (peso) obtido durante o treinamento e $f(x_i, \alpha)$ a própria previsão do modelo para a entrada x_i. Ao realizar um procedimento de teste, ou seja, com vetores que não estão em X, é possível definir o erro cometido pelo modelo de acordo com a Equação 2.1 (SANTOS, 2019) na qual c é uma métrica para diferença entre $f(x_i, \alpha)$ e y_i, $R(\alpha)$ corresponde ao **risco esperado**.

$$R(\alpha) = \frac{1}{2} \int c(f(x, \alpha), y)\, dP(x, y) \tag{2.1}$$

No entanto, esse risco não pode ser calculado uma vez que a distribuição de probabilidade $P(x, y)$ é desconhecida. Portanto, calcula-se o **risco empírico** definido pela Equação 2.2.

$$R_{emp}(\alpha) = \frac{1}{2N} \sum_{i=1}^{N} c(f(x_i, \alpha), y_i) \tag{2.2}$$

Ao contrário do risco esperado, o risco empírico R_{emp} pode ser obtido a partir da técnica de *cross validation* aplicada sobre os dados de treinamento ou por um conjunto de amostras e rótulos que não pertença ao de treinamento, denomidado conjunto de validação (SANTOS, 2019).

Embora o risco esperado 2.1 não possa ser calculado, é possível definir limite superior para o risco com probabilidade $\eta - 1$ tal que $0 < \eta < 1$, estimado pela Equação 2.3. O termo h corresponde à dimensão de Vapnik Chervonenkis (VC), que indica a complexidade da SVM e, com isso, sua capacidade de classificação. Por fim, o segundo termo da expressão é o termo de capacidade (SANTOS, 2019).

$$R(\alpha) \leq R_{emp}(\alpha) + \sqrt{\frac{h(log(2N/h) + 1) - log(\eta/4)}{N}} \tag{2.3}$$

Naturalmente, busca-se minimizar o limite do risco esperado. Todavia, temos novamente uma situação contraditória uma vez que quanto menor o risco empírico, maior o termo de capacidade, como indicado na Figura 2, adaptada de (LORENA; CARVALHO, 2007).

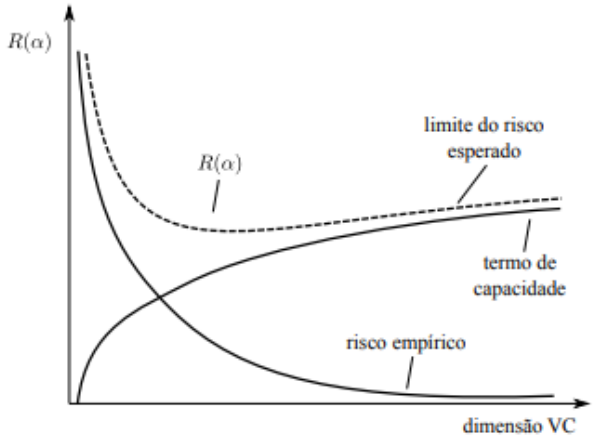


Figura 2 – Minimização do risco estrutural

Analisando a Figura 2, pode-se observar dois fenômenos comuns à aprendizagem de máquina: ***underfitting*** e ***overfitting***. Modelos que tenham risco empírico mínimo não necessariamente apresentarão boa capacidade de generalização, uma vez que estão sobreajustados aos dados (*overfitting*) e acabam incorporando ruídos na modelagem. Por outro lado, o subajuste dos modelos (*underfitting*) acontece quando o risco empírico é alto, indicando baixa aderência do modelo aos dados. Em sua implementação, as SVMs buscam minimizar simultaneamente risco empírico o termo de capacidade delimitando um hiperplano de margens largas, isto é, um plano separador que conta com distância máxima até as amostras mais próximas dos dados do conjunto de treinamento (SANTOS, 2019).

2.2.2 Hiperplanos

Para formalizar a definição de SVM ou máquinas de vetores suporte, é necessário tratar, primeiramente, de hiperplanos. Em um espaço de dimensão p, o hiperplano é definido como um subespaço de dimensão $p-1$. Isto é, em um espaço tri-dimensional o hiperplano é,

de fato, um plano, enquanto em espaço bi-dimensional é, simplesmente, uma reta (JAMES et al., 2013).

A definição matemática de um hiperplano, considerando espaço de p dimensões, é:

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_p X_p = 0 \tag{2.4}$$

Ou seja, o conjunto de pontos $X = (X_1, X_2, ..., X_p)$ para o qual a Equação 2.4 é válida constitui pontos do hiperplano (JAMES et al., 2013).

No entanto, podem existir pontos do espaço que não pertencem ao hiperplano e, em vez da Equação 2.4, satisfazem a Equação 2.5 ou 2.6:

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_p X_p > 0 \tag{2.5}$$

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_p X_p < 0 \tag{2.6}$$

Desse modo, pode-se observar que o hiperplano divide o espaço e é, portanto, chamado **hiperplano separador** (JAMES et al., 2013). Por isso, é natural que sejam empregados como fronteiras de classificação, denotando, espacialmente, a divisão entre duas classes.

2.2.3 SVM - Classificação

Quando aplicadas em contexto de classificação, as SVMs buscam definir um hiperplano que separe os dados de treinamento em duas classes distintas e assegure boa generalização. Para tal, aplicam o conceito de margens, hiperplanos paralelos ao hiperplano separador que definem a flexibilidade ou tolerância do modelo a erros. A fim de maximizar a generalização essas margens devem ser definidas para que a distância máxima entre elas ocorra com o mínimo de risco empírico (SANTOS, 2019).

Em seu trabalho, (CORTES; VAPNIK, 1995) propõe categorização das SVM de acordo com os dados a serem tratados. As SVM **linearmente separáveis** se dividem entre margens rígidas e margens suaves. Por outro lado, dados **não linearmente separáveis** sofrem transformação através de uma **função de kernel** de modo que sejam levados a um espaço de dimensão superior, passando a ser linearmente separáveis, e permitindo modelagem por margens suaves.

2.2.3.1 SVM - Linearmente separáveis

Em um espaço com dados linearmente separáveis nota-se que existem infinitos hiperplanos possíveis para cumprir essa função, dado que mínimos movimentos de translação e rotação definem diferentes hiperplanos, como é exemplificado na Figura 3 (JAMES et al.,

2013). Nesse contexto, faz-se necessário definir método para a seleção de um hiperplano ótimo.

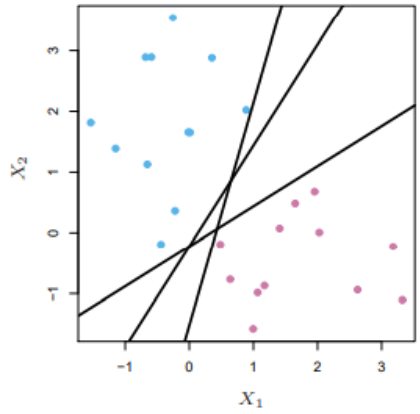


Figura 3 – Infinitos hiperplanos sobre um espaço (JAMES et al., 2013)

Calculando a distância perpendicular entre todas as amostras e um determinado hiperplano, a menor distância observada é denominada **margem** e os pontos que a delimitam são os **vetores de suporte**. O objetivo das SVM, nesse caso, é determinar quais são os vetores de suporte que implicam margens de largura máxima, que não admitem erro, e são portanto denominadas margens rígidas (SANTOS, 2019)(CORTES; VAPNIK, 1995).

Pontos que se encontram sobre as margens satisfazem a Equação 2.7, que sintetiza a restrição do problema de treinamento e assegura erro empírico nulo na classificação. A maximização da distância entre as margens (tratadas por H_1 e H_2), por sua vez, é dada pela Equação 2.8.

$$y_i(w \cdot x_i + b) - 1 \geq 0 \tag{2.7}$$

$$d(H_1, H_2) = \frac{2}{||w||} \tag{2.8}$$

Diante do exposto, as margens máximas são obtidas minimizando $||w||$ na Equação 2.8, de acordo com a restrição da Equação 2.7. Portanto, w e b estabelecem o problema de otimização primal da Equação 2.9, no qual N representa a cardinalidade do conjunto de treinamento.

$$\begin{aligned} &Min \quad \frac{1}{2}||w||^2 \\ &s.a. \\ &y_i(w \cdot x_i + b) - 1 \geq 0 \quad i = 1,2,...,N \end{aligned} \tag{2.9}$$

Analisando a Equação 2.9, verifica-se que o problema apresenta duas variáveis e N restrições que implicam em alta complexidade de solução. Para simplificar a etapa de treinamento, (GRIVA; NASH; SOFER, 2008) propõe a aplicação de multiplicadores de Lagrange

além do teorema de Karush-Kuhn-Tucker (KKT), transformando o problema primal em seu correspondente dual que, pelo Teorema da Dualidade Fraca, deve apresentar a mesma solução ótima (SANTOS, 2019). Assim, o problema de otimização descrito pela Equação 2.10 é conhecido como dual de Wolf, no qual α_i são os multiplicadores de Lagrange, ou seja, as variáveis de otimização, enquanto x e y são amostras e rótulos de X, respectivamente.

$$\begin{aligned} &Max \quad L_d(\alpha_i) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j (x_i \cdot x_j) \\ &s.a. \\ &\sum_{i=1}^{N} \alpha_i y_i = 0 \\ &\alpha_i \geq 0 \qquad i = 1,2,...,N \end{aligned} \tag{2.10}$$

Assim, uma vantagem notável das SVM é que o modelo dado pela Equação 2.10 é convexo e, portanto, admite solução ótima e única.

2.2.3.2 SVM - Não linearmente separáveis

Ao contrário do exposto na seção anterior, conjuntos de dados podem estar distribuídos de forma não linearmente separável. Ainda assim, existem casos que podem ser divididos por um hiperplano separador desde que o modelo apresente certa tolerância a erros. Esse modelo recebe o nome de **SVM de margens suaves**, com os erros administrados por um **parâmetro de regularização**.

O problema de otimização, portanto, envolve a Equação 2.9 acrescida da variável de folga ξ_i, que confere ao modelo flexibilidade a erros, e do parâmetro de regularização C, implicando em 2.11 (SANTOS, 2019).

$$\begin{aligned} &Min \quad \frac{1}{2}||w||^2 + C \sum_{i=1}^{N} \xi_i \\ &s.a. \\ &y_i(w \cdot x_i + b) + \xi_i - 1 \geq 0 \quad i = 1,2,...,N \end{aligned} \tag{2.11}$$

Aplicando esse método, os erros podem ser verificados de duas formas distintas, conforme ilustrado na Figura 4 (JAMES et al., 2013). A amostra 1 é uma classificação correta feita entre a margem e o hiperplano mas penalizada por ξ_1, caso análogo a 8 que é penalizado por ξ_8. Por outro lado, 11 é um exemplo de erro empírico, pois se encontra do lado oposto do hiperplano, que será penalizado por ξ_{11}. Esse fato se estende a 12 e ξ_{12}.

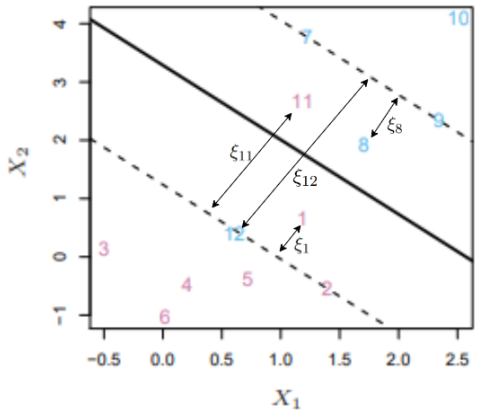


Figura 4 – Adaptação de amostras classificadas incorretamente (JAMES et al., 2013)

A determinação do parâmetro de regularização C cabe ao projetista responsável pela modelagem e implica no compromisso entre a complexidade e a capacidade de generalização de uma SVM. Em geral, altos valores de C (que pode tender ao infinito) aproximam o modelo das margens rígidas. Assim, se estabelece uma hipervalorização de dados específicos ao invés de gerais, o que pode aumentar a sensibilidade a ruído. Por outro lado, se C tender a zero, a SVM desenvolve tolerância a erros a tal ponto que não poderá classificar satisfatoriamente, afetando sua capacidade de generalização.

Por fim, o problema de otimização dual da SVM com margens suaves 2.12 pode ser obtido de maneira análoga ao caso anterior de margens rígidas como demonstrado em (SANTOS, 2019).

$$
\begin{aligned}
&Max \quad L_d(\alpha_i, \xi_i) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j (x_i \cdot x_j) \\
&s.a. \\
&\sum_{i=1}^{N} \alpha_i y_i = 0 \\
&\xi_i \geq 0 \qquad i = 1,2,...,N \\
&0 \leq \alpha_i \leq C \qquad i = 1,2,...,N
\end{aligned}
\tag{2.12}
$$

2.2.3.3 Funções de *kernel*

Ainda que as margens suaves apresentem alternativa à classificação de dados não linearmente separáveis, existem classes dispostas de tal maneira que não retornarão resultados satisfatórios com essa técnica, como na Figura 5 (CORTES; VAPNIK, 1995). Analisando o espaço $\mathcal{L}$, observa-se que não é possível delimitar hiperplanos separadores, tal como definido na Seção 2.2.2, que dividam o conjunto de dados. Nesse caso, uma nova abordagem é proposta por meio da aplicação de uma transformação não linear Φ que mapeia dados do **espaço original** $\mathcal{L}$ para o **espaço de características** $\mathcal{H}$ com dimensão superior. Uma

vez realizado esse procedimento, nota-se que passa a ser possível definir hiperplanos no conjunto de dados.

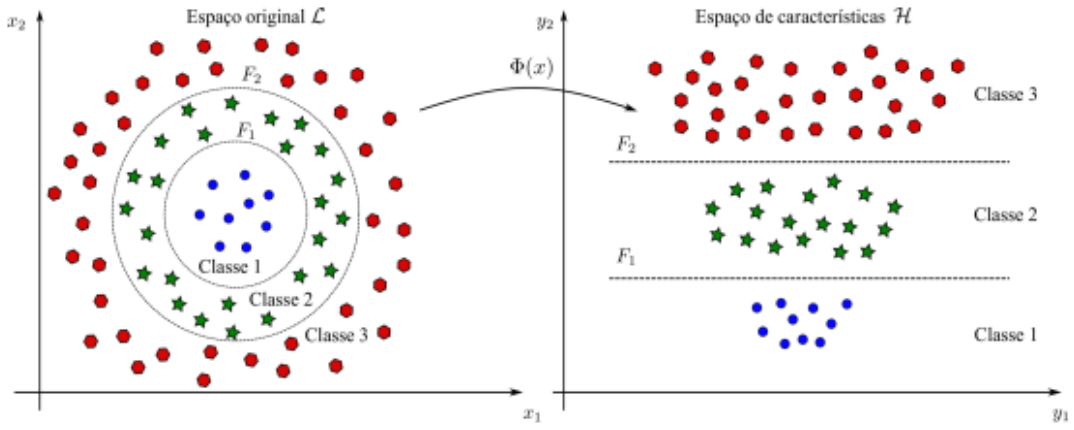


Figura 5 – Transformação não linear do espaço de dados

Nesse contexo, o **Teorema de Cover** prevê duas condições para que um conjunto de dados, independente da combinação de rótulos, possa ser separável em $\mathcal{H}$ de acordo com determinada probabilidade (SANTOS, 2019):

- A transformação $\Phi(x)$ seja não linear;
- A dimensão do espaço de característica seja suficientemente alta.

Tomando as condições do Teorema de Cover e propriedades de um **Espaço de Hilbert**, ou seja, espaço vetorial no qual é definido o produto interno, fundamenta-se teoria suficiente para definir transformação que mapeie dados do espaço de entrada para o espaço de características tendo dimensão adequada (THEODORIDIS; KOUTROUMBAS, 2009).

Portanto, $\mathcal{H}$ deve ser um espaço de Hilbert já que o modelo de margens suaves depende exclusivamente do produto interno $\Phi(x_i) \cdot \Phi(x_j)$, dando origem ao problema 2.13.

$$
\begin{aligned}
&Max \quad L_d(\alpha_i) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j (\Phi(x_i) \cdot \Phi(x_j)) \\
&s.a. \\
&\sum_{i=1}^{N} \alpha_i y_i = 0 \\
&\xi_i \geq 0 \qquad i = 1,2,...,N \\
&0 \leq \alpha_i \leq C \qquad i = 1,2,...,N
\end{aligned}
\tag{2.13}
$$

Embora esse método mantenha as características das SVM de margens suaves para o caso não linear tratado nessa seção, torna-se inviável determinar a lei de formação da função Φ e calcular o produto interno no espaço de características. Nesse contexto, a técnina do *kernel trick* permite que Φ seja definido implicitamente e o modelo tenha acesso aos produtos internos do espaço $\mathcal{H}$ (SANTOS, 2019).

Uma vez que o treinamento de uma SVM depende exclusivamente do produto interno $\Phi(x_i) \cdot \Phi(x_j)$, definir uma função K, denominada função núcleo, tal que $K(x_i, x_j) = \Phi(x_i) \cdot \Phi(x_j)$, implica em simplificação do cálculo realizado pelo algoritmo. Substituindo $K(x_i, x_j)$ no Problema 2.13 resulta 2.14.

$$
\begin{aligned}
&Max \quad L_d(\alpha_i) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j K(x_i, x_j) \\
&s.a. \\
&\sum_{i=1}^{N} \alpha_i y_i = 0 \\
&\xi_i \geq 0 \qquad i = 1,2,...,N \\
&0 \leq \alpha_i \leq C \qquad i = 1,2,...,N
\end{aligned}
\tag{2.14}
$$

No entanto, para que K represente o produto interno entre elementos do conjunto de treinamento, a função deve satisfazer o Teorema de Mercer. Isto é, tomando matriz $K = k_{ij}$ tal que $k_{ij} = K(k_i, k_j)$, denominada **matriz de Gram**, esta deve ser semidefinida positiva e simétrica (HA QUANG; NIYOGI; YAO, 1970).

Por consequência, o Problema 2.14 será convexo se, e somente se, a matriz K satisfaz o Teorema de Mercer. Caso contrário, o *kernel* proposto implicará em modelo com mínimos locais, elevando significativamente a dificuldade do treinamento (SANTOS, 2019). A Tabela 1 sintetiza as funções de kernel mais populares (BURGES, C. J., 1998).

Kernel	**Função** $K(x_i, x_j)$	**Parâmetros**
Polinomial	$(\beta(x_i, x_j) + \kappa)^d$	β, κ, d
Gaussiano (RBF)	$e^{\frac{-\|\|x_i - x_j\|\|^2}{\gamma}}$	γ
Sigmoidal	$tanh(\delta(x_i \cdot x_j) + \rho)$	δ, ρ

Tabela 1 – *Kernels* mais utilizados

2.2.4 SVM - Regressão

Além da classificação, já enunciada anteriormente, é possível aplicar o princípio das SVM também em problemas de regressão com *Support Vector Regressors* (SVR). Nesse

contexto, de forma simplificada, o hiperplano é definido como um aproximador de funções que se ajuste à curva dos dados.

As SVR derivam de um conjunto de treinamento $X = [(x1, y1), ...,(xN, yN)] \subset X \times \mathbb{R}$, em que $X = \mathbb{R}^d$ corresponde ao espaço da curva de entrada. A modelagem ocorre de maneira similar à classificação, desde que ajustada a função de penalização dos erros durante o treinamento. Tomando inicialmente o caso linear, a função $f : X \to \mathbb{R}$, com $w \in X$ e $b \in \mathbb{R}$, é definida 2.15 (SANTOS, 2019):

$$f(x) = \langle w, x \rangle + b \tag{2.15}$$

Com essa definição, pode-se estabelecer que as margens do aproximador serão na forma da Equação 2.16, na qual ϵ denota o máximo desvio aceitável (erro empírico).

$$\begin{aligned} H1 &: \ w \cdot x + b + \epsilon = 0 \\ H2 &: \ w \cdot x + b - \epsilon = 0 \end{aligned} \tag{2.16}$$

Logo, a distância entre as margens pode ser definida por 2.17:

$$d(H_1, H_2) = \frac{2\epsilon}{||w||} \tag{2.17}$$

Novamente, visando boa capacidade de generalização, busca-se que as margens sejam o mais largas possíveis, ou seja, busca-se maximização da Equação 2.17. Assim, o problema de otimização primal é tal qual 2.18, no qual N e y_i indicam a cardinalidade e os rótulos do conjunto de treinamento, respectivamente (SANTOS, 2019).

$$\begin{aligned} & Min \quad \frac{1}{2}||w||^2 \\ & s.a \\ & y_i - (w \cdot x_i + b) \geq \epsilon \qquad i = 1,2,...,N \\ & (w \cdot x_i + b) - y_i \geq \epsilon \qquad i = 1,2,...,N \end{aligned} \tag{2.18}$$

Da mesma forma como nos casos de classificação, o problema 2.18 pode estar sujeito a dados não linearmente separáveis. Portanto, erros devem ser ser tolerados diante de penalização por uma constante positiva implicando em margens suaves e no problema da Equação 2.19.

$$
\begin{aligned}
&Min \quad \frac{1}{2}||w||^2 + C\sum_{i=1}^{N}(\xi_i + \xi_i^*) \\
&s.a \\
&y_i - (w \cdot x_i + b) \geq \epsilon + \xi_i \qquad i = 1,2,...,N \\
&(w \cdot x_i + b) - y_i \geq \epsilon + \xi_i^* \qquad i = 1,2,...,N \\
&\xi_i, \quad \xi_i^* \geq 0
\end{aligned}
\tag{2.19}
$$

Analogamente à calssificação, o parâmetro de regularização C penaliza dados que se encontram além da região definida por ϵ, que é um resultados da função de perda (*loss function*). Entre diversas funções de perda, destacam-se a ϵ-*insentive*, a Laplaciana, a Gaussiana, a perda robusta de Huber, a polinomial e polinomial por partes (SMOLA; SCHÖLKOPF, 2004)(SMOLA; BURGES, C. et al., 1996). Nesse trabalho, adotou-se a ϵ-*insensitive*, que em geral apresenta os melhores resultados mantendo a esparsidade da SVM (SMOLA; SCHÖLKOPF, 2004) e também foi aplicada na ferramenta a ser utilizada (LLANOS; SILVA SANTOS; DOS SANTOS, 2021).

De forma geral, a função aproximadora é definida dentro de um **tubo insensível** que delimita uma zona de insensibilidade conforme indicado na Figura 6 (SANTOS, 2019). Portanto, a penalização dos dados ocorre de acordo a Equação 2.20.

$$
|\xi|_e = \begin{cases} 0, & |\xi| \leq \epsilon \\ |\xi| - \epsilon, & |\xi| > \epsilon \end{cases}
\tag{2.20}
$$

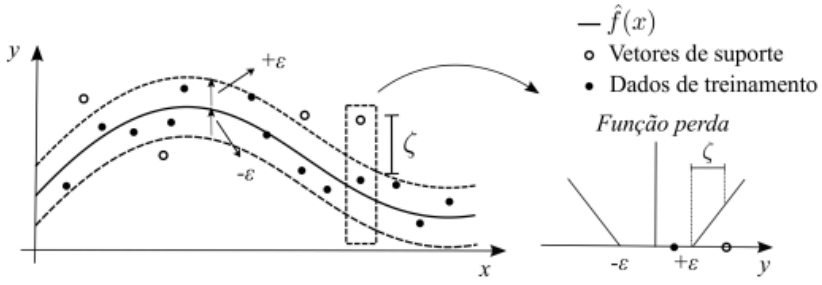


Figura 6 – Função de perda ϵ-*insensitive* (SANTOS, 2019)

É natural observar que a função de penalização afeta diretamente a de aproximação e estabelece-se um equilíbrio entre a capacidade de ajuste e complexidade. Isto é, quanto maior o valor de ϵ, maior será o tubo e consequentemente a tolerância a erros. Logo o modelo apresentará aproximação pouco ajustada à curva de entrada e baixo número de vetores de suporte (complexidade). Por outro lado, valores menores de ϵ tendem a hiper-ajustar o modelo à curva de entrada com alto número de vetores de suporte.

Por fim, buscando novamente simplificação da solução matemática, é possível definir o problema dual de 2.19 de acordo com a Equação 2.21 (SANTOS, 2019).

$$
\begin{aligned}
&Max \quad L(\alpha,\alpha^*) = -\frac{1}{2}\sum_{i,j=1}^{N}(x_i \cdot x_j)(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*) - \epsilon\sum_{i=1}^{N}(\alpha_i - \alpha_i^*) + \sum_{i=1}^{N} y_i(\alpha_i - \alpha_i^*) \\
&s.a \\
&\sum_{i=1}^{N}(\alpha_i - \alpha_i^*) = 0 \\
&\alpha_i, \alpha_i^* \in [0, C] \qquad i = 1,2,...,N
\end{aligned}
\tag{2.21}
$$

2.3 NIOTS

Diante do exposto na Seção 2.1, pode-se aprofundar a discussão sobre Máquinas de Vetores Suporte. As SVM, como outros algoritmos de *machine learning*, definem modelos para mapear processos da vida real por meio de parâmetros obtidos por um método de treinamento. Adotando maior rigor matemático, no caso de SVMs estes parâmetros são coeficientes de Lagrange, cujo cálculo é um problema convexo, isto é, um problema de minimização sem mínimos locais (SANTOS, 2019). Por conta disso, garante-se solução ótima para o modelo, o que representa vantagem sobre outros algoritmos como as Redes Neurais Artificiais (RNAs). Essas, por sua vez, têm problema de determinação dos parâmetros (pesos da rede) considerado não convexo, o que significa que a solução ótima não é garantida pela heurística de treinamento.

Por outro lado, quando parâmetros envolvidos em modelos de *machine learning* definem sua estrutura, são denominados hiperparâmetros. No caso das RNAs, estão atrelados ao número de camadas da rede, neurônios por camada, épocas de treinamento, entre outros. Nas SVMs, por sua vez, os parâmetros estão relacionados às funções *kernel* que compõem o modelo.

Nesse sentido, o NIOTS é uma ferramenta que implementa meta-heurísticas bio-inspiradas para seleção de hiperparâmetros de SVMs como um problema de otimização multi-objetivo (LLANOS; SILVA SANTOS; DOS SANTOS, 2021). Ao fim do processo, retorna até 40 modelos distintos, seus hiperparâmetros obtidos, o *Mean Square Error* (MSE) de treinamento e o número de vetores de suporte (SV) obtidos. Todos eles são igualmente ótimos entre si, estando no limite da Fronteira de Pareto (LLANOS; SILVA SANTOS; DOS SANTOS, 2021).

2.4 Regressão Linear

Para exposição da modelagem por **Regressão Linear Múltipla** é necessário o estabelecimento da teoria de **Regressão Linear** e outros conceitos básicos.

2.4.1 Regressão Linear Simples

A Regressão Linear é uma abordagem estatística dentre as mais simples e objetivas que permite estabelecer uma predição quantitativa Y com base em única variável X assumindo relação aproximadamente linear. Relação essa que pode ser sintetizada na forma da Equação 2.22.

$$Y \approx \beta_0 + \beta_1 X \tag{2.22}$$

Tomando a Equação 2.22, que define uma equação geral, β_0 e β_1 são constantes que representam os coeficientes linear e angular de um modelo linear (JAMES et al., 2013). No entanto, os valores ideais são desconhecidos.

Portanto, o processo de treinamento em Regressão Linear consiste em utilizar os dados de treinamento para estimar $\hat{\beta_0}$ e $\hat{\beta_1}$. Tomando $X = \{x_1, x_2, ..., x_n\}$ e $Y = \{y_1, y_2, ..., y_n\}$ com cardinalidade n, busca-se aproximação tal como a Equação 2.23.

$$y_i \approx \hat{\beta_0} + \hat{\beta_1} x_i \qquad i = 1, 2, ..., n \tag{2.23}$$

Em outras palavras, o modelo obtém $\hat{\beta_0}$ e $\hat{\beta_1}$ que definam a reta mais próxima possível de todos os n pares de X e Y. Matematicamente falando, "proximidade" pode ser medida de diferentes maneiras. Nesse caso, toma-se o critério de *least squares* ou mínimos quadrados (JAMES et al., 2013).

Com base no exposto, é possível expressar a i-ésima previsão e o i-ésimo resíduo do modelo de acordo com 2.24.

$$\begin{aligned} \hat{y_i} &= \hat{\beta_0} + \hat{\beta_1} x_i \\ e_i &= y_i - \hat{y_i} \end{aligned} \tag{2.24}$$

Assim, pode-se definir *residual sum of squares* (RSS), isto é, a soma dos quadrados dos resíduos, como $RSS = {e_1}^2 + {e_2}^2 + ... + {e_n}^2$. O critério dos mínimos quadrados, portanto, determina $\hat{\beta_0}$ e $\hat{\beta_1}$ de forma a minimizar o RSS, o que implica na Equação 2.25.

$$\begin{aligned} \hat{\beta}_1 &= \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2} \\ \hat{\beta}_0 &= \bar{y} - \hat{\beta}_1 \bar{x} \\ onde \quad \bar{y} &= \frac{1}{n}\sum_{i=1}^{n} y_i, \qquad \bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \end{aligned} \tag{2.25}$$

Em outras palavras, 2.25 determina as **estimativas dos coeficientes de mínimos quadrados** (JAMES et al., 2013).

Uma vez realizado o treinamento e gerado o modelo é natural que se determine quão bem ele se ajusta aos dados de entrada. No contexto da Regressão Linear, esse ajuste pode ser verificado através de métricas dentre as quais se destaca a estatística R^2.

Em suma, R^2, que independe da magnitude de Y, indica a proporção de variância que pode ser explicada no modelo de acordo com a Equação 2.26.

$$\begin{aligned} R^2 &= \frac{TSS - RSS}{TSS} = 1 - \frac{RSS}{TSS} \\ onde \quad TSS &= \sum_{i=1}^{n}(y_i - \bar{y})^2, \qquad RSS = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \end{aligned} \tag{2.26}$$

Analisando o resultado de 2.26, TSS corresponde ao *total sum of squares*, ou seja, soma total dos quadrados. De forma geral, indica a variância total de Y, e pode ser interpretado como a variabilidade inerente à resposta antes da regressão. Por outro lado, RSS é um indicador da variabilidade "não explicada", ou seja, que permanece após definição do modelo. Dessa forma, o termo $TSS - RSS$ mede quanta variabilidade na resposta é "explicada" pela regressão, e por fim, R^2 **representa a proporção de variabilidade em** Y **que pode ser explicada a partir de** X.

2.4.2 Regressão Linear Múltipla

Em aplicações práticas, grande parte dos casos contam com múltiplas variáveis preditoras (de entrada), o que inviabiliza a aplicação da Regressão Linear Simples. Uma alternativa possível consiste em treinar modelos individuais para cada variável, abordagem que se mostra pouco satisfatória. Primeiramente, não é definido como realizar uma única predição a partir das diferentes equações geradas. Além disso, é natural observar que cada um desses modelos ignora a variável referente aos demais no cálculo dos coeficientes, que são determinados de forma independente (JAMES et al., 2013).

Nesse contexto, a Regressão Linear Múltipla estende a estratégia anterior propondo um modelo capaz de ajustar múltiplos preditores atribuindo coeficiente angular a cada

um deles. Em geral, para um caso de p variáveis preditoras, o modelo da Regressão Linear Múltipla é tal como a Equação 2.27.

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_p X_p \tag{2.27}$$

X_j identifica o j-ésimo preditor e β_j relaciona quantitativamente a relação dessa variável com a resposta Y. Logo, β_j pode ser interpretado como **o efeito médio causado em Y pelo aumento de uma unidade de X_j enquanto todos os outros preditores são mantidos fixos** (JAMES et al., 2013).

Analogamente ao caso anterior, o treinamento consiste em estimar os coeficientes $\hat{\beta}_1, \hat{\beta}_2, ..., \hat{\beta}_p$. Consequentemente, previsões podem ser realizadas de acordo com a Equação 2.28.

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x_1 + \hat{\beta}_2 x_2 + ... + \hat{\beta}_p x_p \tag{2.28}$$

A obtenção desses parâmetros acontece pelo mesmo processo da regressão linear simples, ou seja, pelo método dos mínimos quadrados. Isto é, $\hat{\beta}_1, \hat{\beta}_2, ..., \hat{\beta}_p$ são determinados de forma a minimizar o RSS, como expressado na Equação 2.29 (JAMES et al., 2013).

$$\begin{aligned} RSS &= \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \\ &= \sum_{i=1}^{n} (y_i - \hat{\beta}_0 - \hat{\beta}_1 x_{i1} + \hat{\beta}_2 x_{i2} + ... + \hat{\beta}_p x_{ip})^2 \end{aligned} \tag{2.29}$$

Assim, considerando que o conjunto de dados empregado no trabalho conta com múltiplas variáveis preditoras, a regressão linear múltipla se mostra uma abordagem aplicável ao problema.

2.5 Floresta randômica (*Random forest*)

2.5.1 Árvores de decisão

Árvores de Decisão constituem algoritmo supervisionado capaz de estimar valores a partir da aplicação de **regras de divisão** às variáveis de entrada, modelo que pode ser interpretado como uma aproximação por partes. Dentre as vantagens de sua aplicação, destaca-se alta capacidade de visualização gráfica. Inclusive, por conta dessa característica e de um arcabouço matemático reduzido quando comparado aos demais modelos de aprendizagem de máquina, convém a utilização de figuras e exemplos na exposição dessa abordagem (JAMES et al., 2013).

Primeiramente, assume-se *dataset* fictício com variáveis $x \in \{X_1, X_2, ...X_p\}$. Uma regra de decisão é estabelecida na forma $x_i < t_k$, e dessa forma o espaço passa a ser dividido em dois conjuntos: $x_i < t_k$, representado à esquerda, e $x_i \geq t_k$, representado à direita. A Figura 7 exemplifica essa construção.

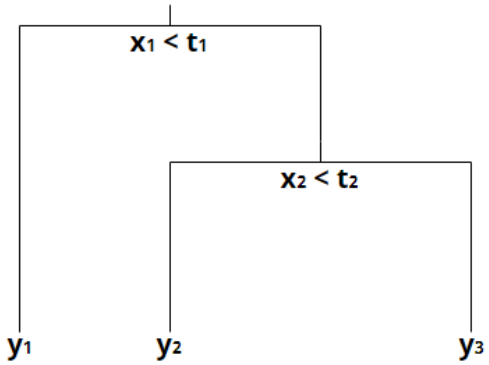


Figura 7 – Exemplo de ramificação em árvore de decisão

É natural, portanto, observar que a árvore da Figura 7 gera uma segmentação ou estratificação do conjunto de dados em três seções R_1, R_2 e R_3, representadas por: $R_1 = \{X|x_1 < t_1\}$, $R_2 = \{X|x_1 \geq t_1, x_2 < t_2\}$, $R_3 = \{X|x_1 \geq t_1, x_2 \geq t_2\}$ (JAMES et al., 2013). Esse resultado pode ser observado graficamente na Figura 8, na qual os retângulos correspondem às regiões e os círculos pretos às amostras.

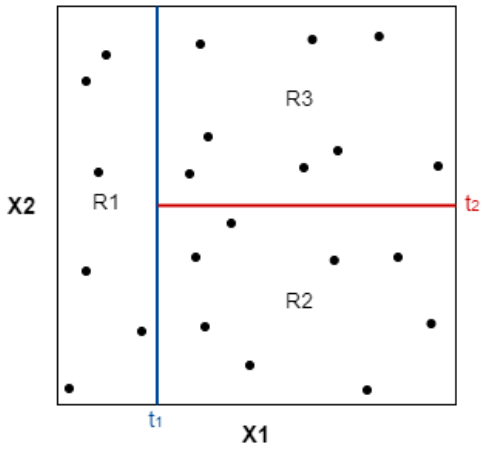


Figura 8 – Exemplo de espaço estratificado por árvore de decisão

Diante do exposto, a analogia com árvores se estende a toda a topologia de forma *top-down*, ou seja, de cima para baixo. Cada um dos pontos onde o espaço é dividido (regras) corresponde a um **nó**, enquanto divisões definidas a partir deles são denominadas **ramos**. Quando um determinado nó é o último de seu segmento, não sendo possível derivar qualquer ramo a partir dele, ele é chamado **folha** e delimita uma região R.

2.5.2 Árvores de Regressão

Com base nos conceitos definidos sobre árvores de decisão, os passos abaixo estabelecem o processo para que sejam aplicadas em regressão (JAMES et al., 2013):

1. Treinamento: Dividir o espaço preditor, isto é, o espaço de possíveis valores de $X_1, X_2, ..., X_p$ em J regiões $R_1, R_2, ..., R_J$.

2. Teste: Caso uma amostra do conjunto de testes se enquadre em uma região R_j, deve-se predizer valor que corresponda à média de todas as amostras de treinamento em R_j.

O desafio principal consiste em definir a estratégia para realização do passo 1. Em geral, busca-se delimitar $R_1, ..., R_J$ de forma a minimizar o RSS definido pela Equação 2.30, onde y_i e $\hat{y}_{R_j}$ correspondem, respectivamente, ao valor atribuído à i-ésima amostra e à média das amostras em R_j (JAMES et al., 2013).

$$\sum_{j=1}^{J} \sum_{i \in R_j} (y_i - \hat{y}_{R_j})^2 \tag{2.30}$$

No entanto, é computacionalmente inviável calcular todas as possíveis combinações do espaço em J regiões e, portanto, adota-se **divisão binária recursiva**, abordagem *top-down* e *greedy*. Primeiramente, *top-down* pois inicia no topo da árvore, quando ainda não há divisão, e sucessivamente divide o espaço gerando dois novos ramos a cada iteração. *Greedy*, que do inglês significa "ganancioso", se dá porque em cada etapa o algoritmo escolhe a melhor divisão naquele momento, sem considerar qualquer ramificação que possa levar a uma árvore melhor no futuro (JAMES et al., 2013).

Assim, pode-se sintetizar que o algoritmo percorre cada uma das variáveis $X_1, ..., X_p$ estabelecendo limiar s de forma que o espaço seja dividido em

$$R_1(j,s) = \{X | X_j < s\} \quad e \quad R_2(j,s) = \{X | X_j \geq s\} \tag{2.31}$$

e obtendo valores de j e s que minimizem a Equação 2.32.

$$\sum_{i : x_i \in R_1(j,s)} (y_i - \hat{y}_{R_1})^2 + \sum_{i : x_i \in R_2(j,s)} (y_i - \hat{y}_{R_2})^2 \tag{2.32}$$

Essa otimização de j e s para minimização de 2.32 pode ser computada relativamente rápido, mas depende de um critério de parada (JAMES et al., 2013). Em geral, pode-se repetir o processo até que uma região R_j não agrupe mais do que m amostras ou ainda até um número pré determinado de folhas.

2.5.3 *Random Forest*

As árvores de decisão apresentadas anteriormente tendem a sofrer com alta variância. Considerando um conjunto de dados hipotético, dividi-lo aleatoriamente em duas partes e aplicar o algoritmo de árvores de decisão em cada uma delas poderia retornar resultados amplamente distintos (JAMES et al., 2013). Uma alternativa, nesse caso, é a aplicação de ***bootstrap***.

De maneira simplificada, *bootstrap* consiste em estratégia para emular a obtenção de novos conjuntos de dados sem a necessidade de gerar quaisquer amostras artificialmente. Esse processo constrói um novo conjunto amostrando dados do *dataset* original de forma aleatória, conforme a Figura 9 (JAMES et al., 2013). Para um *dataset* Z com $n = 3$ dados, novos conjuntos $Z^{*1}, ..., Z^{*B}$ podem ser obtidos amostrando aleatoriamente n vezes de Z com reposição.

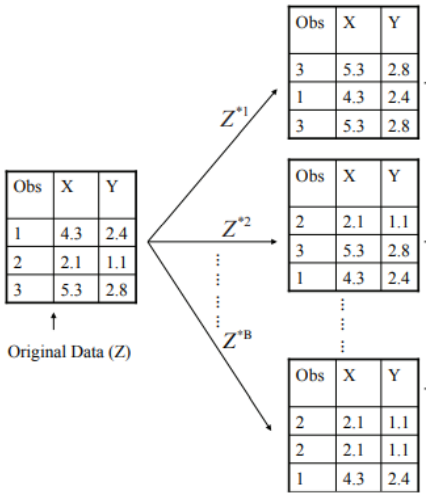


Figura 9 – Exemplo de *bootstrap* aplicado a conjunto de dados hipotético

Tendo em vista que B diferentes conjuntos $Z_1, ..., Z_B$ com variância σ^2 têm média $\bar{Z}$ com variância equivalente a σ^2/B (JAMES et al., 2013), esse artifício pode ser estendido a modelos estatísticos. Ou seja, com base em B *datasets* é possível treinar um modelo para cada e, ao fim, tomar a média. Quando esses B conjuntos são obtidos a partir de *bootstrap*, todo esse processo recebe o nome de ***bagging***, e encontra-se sintetizado na Equação 2.33 (JAMES et al., 2013).

$$\hat{f}_{bag}(x) = \frac{1}{B}\sum_{b=1}^{B}\hat{f}^{*b}(x) \tag{2.33}$$

Embora *bagging* proponha alternativa para diminuição da variância entre as árvores, ainda é possível verificar um efeito indesejado de **correlação**. Supondo que dentre as variáveis preditoras $X_1, ..., X_P$ exista uma que se mostre muito mais relevante do que as demais, é

lógico concluir que grande parte (se não todas) árvores tomarão essa variável como nó raiz, ou seja, a primeira regra para divisão. Dessa forma, ainda que os conjuntos de dados sejam distintos, os modelos tendem a ser similares. Por conta disso, as predições realizadas por diferentes modelos serão altamente correlacionadas (JAMES et al., 2013).

Nesse cenário, surgem as ***random forests*** ou **florestas randômicas**, cuja diferença central se dá durante as iterações que constroem a árvore. Quando uma divisão é considerada, ao invés de verificar qual a variável preditora mais relevante dentre $\{X_1, ..., X_P\}$, o algoritmo analisará apenas m variáveis candidatas escolhidas aleatoriamente do conjunto anterior. Esse subconjunto é redefinido a cada divisão e, em geral, $m \approx \sqrt{P}$ (JAMES et al., 2013).

Portanto, as florestas randômicas "descorrelacionam" as árvores geradas, uma vez que, em média, $(P - m)/P$ das divisões sequer considerarão a variável preditora mais relevante (JAMES et al., 2013).

2.6 Métricas de Avaliação

Algoritmos de *machine learning* que realizam operação de classificação têm métricas bem definidas e amplamente testadas. Por outro lado, problemas de regressão podem contar com múltiplas métricas que avaliam propriedades distintas do desempenho obtido, tornando complexa sua seleção (SOKOLOVA; LAPALME, 2009).

Dentre as métricas comumente utilizadas, nas quais $\hat{y}$, y e n correspondem, respectivamente, às previsões, rótulos do *dataset* e índice da amostra, destacam-se (SOKOLOVA; LAPALME, 2009)(BOTCHKAREV, 2018):

- Coeficiente de Determinação (CoD):

$$\frac{\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}{\sum_{i=1}^{n}(\hat{y}_i - media(\hat{y}))^2} \tag{2.34}$$

- Erro médio absoluto (MAE):

$$\frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{2.35}$$

- Erro médio quadrático (MSE):

$$\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{2.36}$$

- Raiz quadrática média (RMSE):

$$\sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} = \sqrt{MSE} \tag{2.37}$$

- Raiz quadrática média normalizada (NRMSE):

$$\frac{\sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}}{(y_{max} - y_{min})} = \frac{RMSE}{(y_{max} - y_{min})} \tag{2.38}$$

- Erro percentual médio (MPE):

$$\frac{100}{n}\sum_{i=1}^{n}\frac{(y_i - \hat{y}_i)}{y_i} \tag{2.39}$$

- Erro percentual médio absoluto (MAPE):

$$\frac{100}{n}\sum_{i=1}^{n}\frac{|y_i - \hat{y}_i|}{|y_i|} \tag{2.40}$$

Com base nas principais métricas descritas, é possível separá-las de acordo com a natureza de seus valores ser absoluta ou relativa. Para as métricas absolutas, tais quais MSE e MAE, verifica-se, matematicamente, que seu valor pode ser tão grande quanto for o erro entre as amostras. Com isso, o resultado obtido isoladamente é de pouca informação, uma vez que se trata dos erros somados iterativamente e não reflete diretamente a qualidade de um modelo. Em termos gerais, métricas absolutas não respondem se um modelo é bom por si só, mas tendo um conjunto deles, pode indicar qual o melhor.

Por outro lado, métricas relativas permitem análise individual, focada em único modelo. No caso do CoD, por exemplo, o resultado varia no intervalo 0 a 1, sendo 1 o máximo de exatidão. O MAPE, por sua vez, tem valor percentual, ou seja, quanto menor a porcentagem de erro, mais ajustado o modelo. Então não é necessário executar diversos terinamentos distintos para comparar o resultado dessas métricas, basta verificar quão bom é o valor retornado para o modelo desejado. Assim, conhecendo o problema, projetistas podem inclusive definir valor máximo de erro aceitável e trabalhar com essa margem.

2.7 Codificação *One Hot*

Para discorrer a respeito de Codificação *One Hot*, é necessário definir o que são variáveis categóricas. Em suma, contém rótulos descritivos em vez de valores numéricos. Podem ser tomados como exemplo:

- **Cores**: "vermelho", "azul", "verde", etc.
- **Andares**: "térreo", "primeiro", "segundo", etc.

Adicionalmente, o segundo caso, que descreve andares, pode ser categorizado variável ordinal, uma vez que existe relação de ordenação entre os rótulos.

Embora seja natural que variáveis categóricas constituam interface facilitadora para análise humana dos dados, esse fato não necessariamente se aplica a algoritmos computacionais. Ainda que existam aqueles insensíveis a esse formato, como as árvores de classificação, outros perdem eficiência ou sequer executam corretamente tendo rótulos como entrada.

Nesse contexto, duas principais abordagens podem ser adotadas para ajustar variáveis categóricas.**Codificação por inteiros** atribui valores numéricos inteiros a cada um dos rótulos, tal como:

- **Cores**: "vermelho" (1), "azul" (2), "verde" (3), etc.
- **Andares**: "térreo" (0), "primeiro" (1), "segundo" (2), etc.

Apesar de solucionar a questão, essa abordagem insere, implicitamente, relação ordinal entre todas as categorias. Durante o treinamento, algoritmos tendem a levar esse fator em consideração, fato nem sempre desejado. Além disso, a magnitude atribuída aos inteiros de cada rótulo podem interferir no aprendizado, estabelecendo relações de dominância ou relevância entre as categorias.

Alternativamente, **codificação *one hot*** transforma uma variável categórica de n categorias em n variáveis binárias que indicam, exclusivamente, pertencimento ou não pertencimento da amostra a esse conjunto, como ilustrado na Tabela 2. Dessa forma, os efeitos indesejados da **codificação por inteiros** são eliminados.

Amostra	**Cor**
1	Vermelho
2	Verde
3	Azul

Amostra	**Vermelho**	**Verde**	**Azul**
1	1	0	0
2	0	1	0
3	0	0	1

Tabela 2 – Variável categórica vs *One hot encoding*

Como pode ser observado na Tabela 2, após aplicação dessa codificação apenas um dos bits correspondente aos rótulos deve assumir valor 1. Por isso é dado o nome "*one-hot*", ou seja, somente um verdadeiro.

3 Metodologia

3.1 Pré-processamento

O trabalho descrito em (ROSA, 2020) utilizou a mesma base de dados, também sob orientação do Professor Sanderson Barbalho. Um dos produtos da pesquisa foi o pré-processamento do *dataset*, incluindo limpeza e tratamento dos dados. Tendo em vista os bons resultados obtidos, optou-se por utilizar o conjunto de dados pré-processado, habilitando mais tempo para as etapas a seguir.

Primeiramente, (ROSA, 2020) realizou estudo exploratório a respeito das variáveis mais relevantes para o *lead-time* (preditoras), incluindo diversas rodadas de discussão com gestores da empresa. Desse modo, uma boa quantidade de *features* se mostrou irrelevante e pôde ser removida do conjunto.

Em seguida, realizou-se a limpeza dos dados que apresentassem valores inválidos: valores numéricos negativos; variáveis numéricas contendo texto; variáveis categóricas contendo números; variáveis com dados faltantes. Adiconalmente, (ROSA, 2020) tratou datas, bem como erros de digitação em variáveis categóricas, diminuindo significativamente o número de classes por elas representadas.

Por fim, removeram-se *outliers*, ou seja, amostras tão discrepantes da média que levantam suspeitas sobre representarem um evento atípico ou mesmo um erro na coleta de dados. Dessa forma, a base de dados final obtida por (ROSA, 2020) e utilizada no presente trabalho tem forma apresentada na Tabela 3 e variáveis descritas na sequência.

Amostras	***Features***
11.355	13

Tabela 3 – Forma do *dataset* utilizado

1. **Conjunto (categórica)**: conjunto a que o projeto pertence na estrutura de produto;
2. **Quantidade (numérica)**: quantidade de peças solicitadas;
3. **Ordem de produção (categórica binária)**: definição se a produção será interna ou externa;
4. **Descrição do material (categórica)**: definição do material em que a peça deve ser produzida;

5. **CQ-Entrada (categórica binária)**: indicação se o produto deve passar por primeira etapa de controle de qualidade;
6. **Envio para TS (categórica binária)**: indicação se o produto deve passar por tratamento de superfície (têmpera, etc);
7. **CQ-Entrada1 (categórica binária)**: indicação se o produto deve passar por segunda etapa de controle de qualidade;
8. **Repasse GI-EAPD (numérica)**: total de tempo para repasse de informações da ordem de serviço entre o Escritório de Apoio à Pesquisa e Desenvolvimento (EAPD) para a Gerência Industrial (GI);
9. **Material liberado (categórica binária)**: indicação se o material para fabricação será liberado com atraso, *falso* para ordem externa;
10. **Tempo de espera (numérica)**: tempo de espera em dias por conta da fila de produção, *0* para ordem externa;
11. **LT produção (numérica)**: tempo de fabricação da peça;
12. **LT necessidade (numérica)**: tempo entre o repasse da ordem para o EAPD e a data que a peça necessita estar pronta;
13. **LT total (numérica)**: intervalo de tempo para finalização da ordem de serviço considerando todo o fluxo de materiais e documentos. **É a variável-alvo da predição do modelo.**

3.2 Codificação One Hot

Como descrito no Capítulo 1, variáveis categóricas podem impactar signifcativamente o aprendizado de um algoritmo computacional. Portanto, aplicou-se o método de codificação *one hot* nas seguintes variáveis: Conjunto; Descrição do Material; Material liberado.

O procedimento foi realizado de maneira automática, utilizando a biblioteca **pandas** (TEAM, 2020a) disponível para **Python**, mais especificamente, **pandas.get_dummies** (TEAM, 2020b).

3.3 Divisões da base de dados

Após aplicação de codificação *one hot*, o número de *features*/variáveis pode aumentar drasticamente. Por conta disso, verificou-se empiricamente que treinar um modelo com dimensões elevadas implicava em elevado custo computacional além de resultar em diversas

falhas na execução do software NIOTS. Para solução desses conflitos, novas estratégias de divisão dos dados foram propostas.

3.3.1 *Dataset* reduzido

Primeiramente, optou-se pela simplificação do problema. Logo, buscou-se definir um *dataset* que apresentasse validade em relação ao original, mas que fosse reduzido o suficiente para validar o funcionamento das ferramentas utilizadas. Limitou-se, portanto, o conjunto a 1000 registros amostrados aleatoriamente do cojunto original. A quantidade de *features*, por sua vez, foi resultado da codificação *one hot*, como descrito anteriormente. Essa divisão dos dados foi chamada ***Dataset* reduzido** e suas dimensões resultantes sintetizadas na Tabela 4.

Amostras	***Features***
1000	71

Tabela 4 – Forma do *dataset* reduzido

Tendo feito os ajustes necessários, restou ajustar os dados para aplicação de aprendizagem supervisionada. Essa estratégia conta com a saída do sistema não somente disponível, mas relacionada a seus respectivos dados de entrada. Com base nisso, os algoritmos buscam hipóteses e padrões que estabeleçam essa relação entrada-saída, como se houvesse uma entidade que, de fato, ensinasse ao modelo (SILVA, I. et al., 2017).

Nessa circunstância, o objetivo consiste em treinar e testar um modelo. Isto é, criar modelo com base em um conjunto de dados e, em seguida, verificar resultados obtidos com um conjunto distinto de entradas. Dessa forma, pode-se avaliar o aprendizado por meio de sua capacidade de generalização ou extrapolação.

Portanto, o *Dataset* reduzido foi dividido aleatoriamente em dois, gerando **conjunto de treino e de teste**. Nesse caso, utilizou-se a divisão apresentada na Tabela 5, obedecendo proporção de 70-30%.

Conjunto	**Amostras**	***Features***
Treino	700	71
Teste	300	71

Tabela 5 – Divisão do *dataset* reduzido

3.3.2 *Dataset* filtrado

Com base no *Dataset* reduzido foi possível verificar o funcionamento da ferramenta por meio de resultados ainda a serem discutidos. No entanto, como explicado, essa etapa consistiu apenas em uma simplificação inicial do problema. Em seguida, buscou-se nova

divisão otimizada da base de dados que pudesse aumentar a confiabilidade do modelo e melhorar os resultados. Para isso, formulou-se a seguinte hipótese:

> *Tomando uma determinada variável X, verifica-se que, dentre os rótulos que ela pode assumir, existem aqueles que são menos populosos, ou seja, têm baixo número de ocorrências. De modo geral, espera-se que essa baixa quantidade de amostras desempenhe papel de ruído, aumentando a complexidade do modelo sem contribuir para sua capacidade de generalização. Portanto, analisar um conjunto de dados derivado somente dos rótulos mais populosos poderia elevar o desempenho?*

Primeiramente, essa proposta foi apresentada a um Professor que já compôs a Equipe de Engenharia da empresa e forneceu os dados. O objetivo central nesse ponto consistiu em validar teoricamente a abordagem e, na sequência, discutir sobre as variáveis mais relevantes para sua aplicação. Como candidatas, foram selecionadas:

a) **Conjunto**: nesse caso, as peças avaliadas pertencem a um mesmo conjunto de montagem;

b) **Descrição do material**: nesse outro, as peças relacionadas foram fabricadas em um mesmo material.

Com base no *Dataset* original, realizou-se levantamento dos dez rótulos com maiores e menores números de ocorrência para as duas variáveis selecionadas. Assim, essa etapa permitiu avaliar a premissa da hipótese proposta. Ou seja, verificar se realmente haviam rótulos pouco populosos dentre os dados. Os números obtidos foram sintetizados nas Tabelas 6 e 7.

Conjuntos mais populosos		**Conjuntos menos populosos**	
MUX_FREE	660	RBNB_MC	3
MUX	614	RBNA_OA	3
ADS_10	582	AWFI_SPE2_EM	3
LASERA	548	RBS_OEB_OMB	3
GSE	543	RBNA_RA_FM1	3
MUX_ME	493	AWFI_OEB_OC_OA	2
MIV	474	RBNA	2
RBN_TO	401	RBNA_ELDA_FM2	2
WFI	374	RBNA_ELDD_FM2	2
LASERV	332	RBNA_CUA	1

Tabela 6 – Comparação entre rótulos da variável **Conjunto**

Materiais mais populosos		Materiais menos populosos	
AL_6351	3382	ACRÍLICO	3
AISI_304	980	SAE_340	3
AL_7075	973	AL_7000	3
AL_6061	950	AISI_420	3
AL_6351	692	LATAO_CZ_121M	3
LATAO	521	POLICARBONATO	2
POLIACETAL	329	POLIPROPILENO	2
KOVAR	328	PERFIL	2
AL_1100	307	COBRE	1
INOX	285	BRONZE	1

Tabela 7 – Comparação entre rótulos da variável **Descrição de Material**

Observando as tabelas, nota-se grande discrepância entre os maiores e menores números de ocorrências, o que validou a investigação da hipótese. Logo, restou a criação de *datasets* que contemplassem somente os casos mais recorrentes de cada variável.

Portanto, aplicou-se uma restrição, que foi chamada de **filtro**, ao *dataset* original. Em suma, a variável **Conjunto** somente poderia assumir os valores **MUX_FREE, MUX, ADS_10,** e **LASER_A** e, para cada um deles, criou-se um conjunto de dados. O mesmo foi realizado para a variável **Descrição do material** com os rótulos **AL_6351, AISI_304, AL_7075, AL_6061**, totalizando 8 ***datasets* filtrados**.

Cada um deles foi separado para as fases de treinamento e teste seguindo a regra de 70-30%, conforme indicado na Tabela 8. A ordem de representação da dimensão é **(amostras)x(*features*)**.

Filtro	Variável	Treino	Teste
ADS_10	Conjunto	407x32	175x32
LASER_A	Conjunto	384x34	164x34
MUX	Conjunto	430x49	184x49
MUX_FREE	Conjunto	462x37	198x37
AISI_304	Material	686x117	294x117
AL_6061	Material	665x56	285x56
AL_7075	Material	681x90	292x90
AL_6351	Material	2367x105	1015x105

Tabela 8 – Divisão dos *Datasets* filtrados

3.3.3 *Dataset* final

A partir dos modelos gerados com o ***dataset* filtrado**, verificou-se nítida superioridade na aplicação do filtro à variável **Conjunto**, com resultados que serão discutidos em maior profundidade adiante no texto. Embora essa divisão dos dados tenha sido importante

para avaliação da hipótese proposta, julgou-se impraticável prosseguir o trabalho apenas com ela. Primeiramente, modelos ajustados individualmente a somente um dos rótulos da variável desenvolveriam pouca generalização, por conta da homogeneidade dos dados. Ainda, de um ponto de vista prático, seria contraprodutivo trabalhar com múltiplos *datasets*.

Por isso, propôs-se a construção de um único *dataset* que combinasse dados de **MUX_FREE, MUX, ADS_10,** e **LASER_A**. Dessa forma, garantiu-se que todos esses rótulos contariam com quantidade suficiente de amostras, minimizando o ruído. Além disso, os dados seriam heterogêneos, isto é, não estariam restritos a um único valor da variável.

Assim, criou-se o ***Dataset* final**. Por ser finalmente a base de dados definitiva do trabalho, aplicou-se a separação entre conjuntos de treino, validação e teste seguindo regra 70-15-15%, como indicado na Tabela 9. O conjunto de validação permite avaliação do modelo durante sua fase de treinamento, otimizando o ajuste de hiperparâmetros. Sua inserção foi uma recomendação do próprio desenvolvedor do NIOTS (SANTOS, 2019), a principal ferramenta utilizada.

Conjunto	**Amostras**	***Features***
Treino	1683	62
Teste	361	62
Validação	360	62

Tabela 9 – Divisão do *Dataset* final

3.4 Métricas de avaliação

Tomando o erro entre a curva predita e a real, diferentes métricas matemáticas se propõem a avaliar os resultados de regressões, incluindo as de algoritmos de *machine learning* (BOTCHKAREV, 2018) (SPÜLER et al., 2015). Nesse trabalho, duas delas serão adotadas: MSE e MAPE.

Inicialmente, o *Mean Squared Error* (MSE) ou Erro Quadrático Médio toma o quadrado da diferença entre as curvas predita e real, ponto a ponto, e retorna a média desses valores, como na Equação 3.1.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{3.1}$$

Em geral, essa métrica é aplicada como função de perda para otimizações, como na própria ferramenta NIOTS (LLANOS; SILVA SANTOS; DOS SANTOS, 2021), que balanceia o *trade off* entre MSE e quantidade de vetores de suporte. Por isso, convém ainda mais utilizá-lo em fase de teste.

Embora seja amplamente utilizado, o MSE é sensível à magnitude dos dados em que é aplicado, pois como enunciado na Seção 2.3, é uma métrica absoluta. Isto é, caso $(y_i - \hat{y}_i) < 1$, ao elevar este termo ao quadrado, a componente de erro dessa amostra diminui. Por outro lado, para $(y_i - \hat{y}_i) > 1$, a potência implica o aumento da componente de erro da amostra. Em outras palavras, o MSE penaliza fortemente erros elevados de um modelo, principalmente os *outliers*.

Nesse cenário, complementa-se a análise utilizando o *Mean Absolute Percentage Error* (MAPE), descrito pela Equação 3.2.

$$MAPE = \frac{100}{n} \sum_{i=1}^{n} \frac{|y_i - \hat{y}_i|}{|y_i|} \tag{3.2}$$

Ao contrário do MSE, que utiliza diretamente a diferença entre as curvas, o MAPE toma a diferença percentual, atenuando a influência da magnitude mencionada anteriormente. Nesse caso, não existe penalização para erros maiores, sendo todos eles balanceados linearmente. Logo, constitui elemento válido para análise dos resultados da regressão.

Retomando o conteúdo da Seção 2.3, optou-se por aplicar uma métrica denomida *absoluta* e outra *relativa*. Assim, dispõe-se de recursos para empreender análise comparativa entre diferentes modelos mas também para seu desempenho individual. Além disso, de modo geral, o MAPE constitui métrica que facilita a visualização do resultado por parte de um gestor envolvida na fabricação das peças, público-alvo dos resultados desse trabalho.

Por fim, com o intuito de propiciar visualização dos resultados, propôs-se a geração de um **histograma** com a configuração da Figura 10: o eixo horizontal representa o erro em dias, sendo centrado em zero, ou seja, apresentando valores positivos e negativos; o eixo vertical denota a "população", isto é, quantas amostras se encontram na faixa de erro determinada pelo outro eixo. Diante disso, é possível estabelecer análise da distribuição de erro em torno de zero, indentificação dos piores casos, além de verificação da proporção entre erro negativo e positivo.

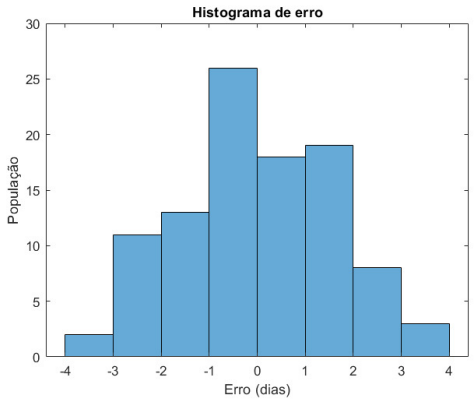


Figura 10 – Exemplo de histograma proposto

3.5 Obtenção dos modelos

O estudo exploratório proposto para esse trabalho consiste fundamentalmente na aplicação do NIOTS, ferramenta desenvolvida para ajuste ótimo de hiperparâmetros das SVM com base em meta-heurísticas bioinspiradas (LLANOS; SILVA SANTOS; DOS SANTOS, 2021). No entanto, a avaliação de um modelo se beneficia amplamente da comparação. Portanto, é importante que métodos ou algoritmos alternativos sejam aplicados ao mesmo conjuntos de dados a fim de gerar resultados que possam ser comparados.

Nesse contexto, julgou-se pertinente definir pelo menos outros dois algoritmos. De forma geral, essa seleção foi realizada com base em dois princípios. Primeiramente, se deu preferência àqueles algoritmos cujo princípio matemático (ou estatístico) fosse o mais distinto possível das SVM, ou seja, do ajuste de um hiperplano não linear. Além do mais, os candidatos também deveriam ser expressivamente relevantes na solução desse problema e no contexto de análise de dados em geral.

Em primeiro momento, buscou-se novamente orientação do Professor que atuou com gestão de projetos e produtos na indústria. De acordo com relatos da experiência dele, **Regressão Linear Múltipla** é uma técnica amplamente aplicada por profissionais no mercado quando um problema é composto por múltiplas variáveis. Somado a isso, esse algoritmo é baseado na definição de modelos lineares no espaço de dados e a otimização de seus parâmetros por mínimos quadrados, o que o qualifica como uma das abordagens alternativas para comparação.

Finalizando a seleção, optou-se por um método baseado em árvores de decisão. Sumariamente, estes aplicam um conjunto de regras que dividem o espaço de dados em regiões de acordo com suas características. Em geral, são reconhecidos como algoritmos populares e de simples aplicação. Provou-se um desafio encontrar evidências científicas dessa popularidade, mas como confirmação, consultou-se pesquisa realizada em 2019 pelo ***Kaggle***, uma das maiores plataformas e comunidades de aprendizagem de máquina e ciência de dados. Ao longo do processo, os participantes foram questionados sobre quais algoritmos utilizavam regularmente na solução de seus problemas, podendo assinalar mais de um. Ao todo, aproximadamente 15.000 usuários dentre profissionais, estudantes e entusiastas contribuíram e os resultados estão sintetizados na Figura 11. Assim, corroborou-se a decisão por ***random forest***.

3.5.1 SVM - NIOTS

Embora os hiperparâmetros das SVM sejam realizados de forma automática pelo NIOTS, cabe ao projetista uma etapa prévia de configuração. Dado ao curto período disponível para familiarização com a ferramenta, contou-se com apoio de seu desenvolvedor para configurar a aplicação conforme a Tabela 10.

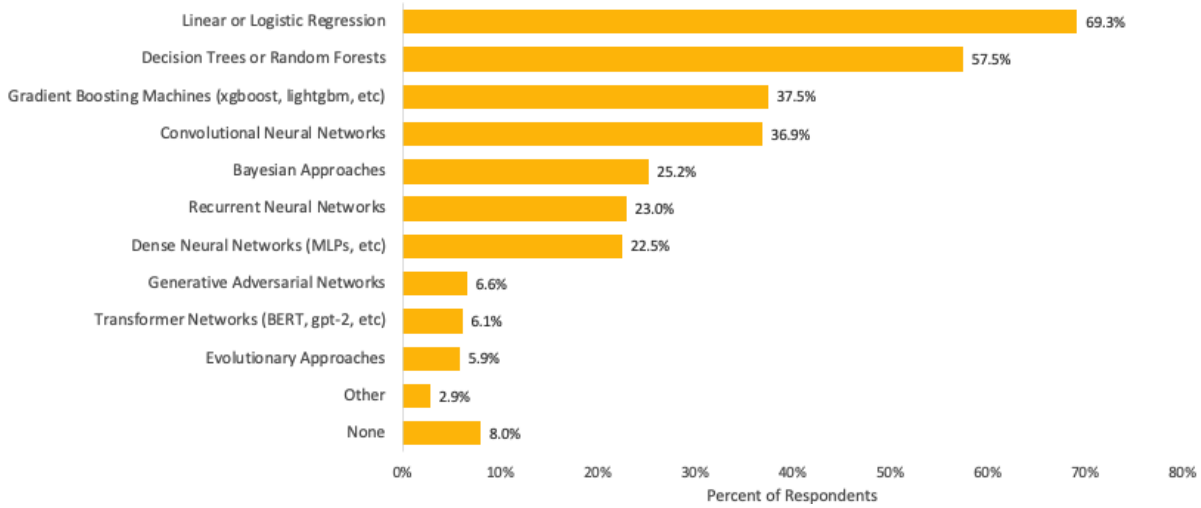


Figura 11 – Adaptação dos algoritmos mais populares na comunidade Kaggle

Configurações NIOTS (SVM)	
Máquina	
Modelo	SVR
Biblioteca	LibSVM
Meta-heurística	*Multi Objective Differential Evolution (MODE)*
Kernel	RBF
Gerais	
Iterações	150
Distribuição aleatória	Uniforme
Gamma (min/máx)	-10/10
Épsilon (máx)	2
MODE	
População	20
Fator de escala	0.7
Taxa de crossover	0.4

Tabela 10 – Parâmetros de configuração - NIOTS

Em seguida, o treinamento é executado. Para o mesmo conjunto de dados, podem ser obtidos até 40 modelos distintos mas que sejam equivalentes na solução do problema de otimização da meta-heurística adotada, ou seja, modelos igualmente ótimos de acordo com a Fronteira de Pareto (LLANOS; SILVA SANTOS; DOS SANTOS, 2021). O produto desse processo consiste em uma série de arquivos, na qual **Resumo** é gerado com as informações de todos os modelos obtidos. Esses modelos, por sua vez, contam com um arquivo individual que os descreve conforme a Tabela 11.

Arquivos retornados - NIOTS	
Resumo	
Configurações	Registro da configuração utilizada
C	Parâmetro de regularização
Gamma	Parâmetro do kernel RBF
Épsilon	Tolerância máxima
MSE	Erro Quadrático Médio
SV	Número de vetores de suporte
Modelo	
Hiperparâmetros	C, Gamma, Épsilon
Resultados	MSE, SV
SV ind/coef	Índice e coeficiente dos SV

Tabela 11 – Conteúdo dos arquivos gerados pelo NIOTS em fase de treinamento

Os arquivos **Modelo** da Tabela 11 são nomeados **SVM_1, SVM_2, ..., SVM_40** e ordenados da seguinte forma: SVM_1 e SVM_2 devem ser, respectivamente, os modelos com menor MSE e menor número de vetores de suporte em treinamento; os demais apresentam um equilíbrio entre esses dois valores.

Cabe ressaltar que o MSE e a quantidade de vetores de suporte são inversamente proporcionais. Estabelecendo um paralelo com a teoria de máquina de vetores de suporte, pode-se sintetizar que quanto menor o MSE, melhor é a capacidade de generalização. Por outro lado, quanto mais vetores de suporte mais complexo é o modelo. Portanto, é natural observar que resultados que apresentem baixo MSE (alta generalização) impliquem em alta complexidade (grande quantidade de SV), sendo a recíproca verdadeira.

No presente trabalho, para cada um dos *datasets*, será realizada análise dos seguintes modelos: **SVM_1**, que apresenta menor MSE e, portanto, tende a ter predições mais ajustadas aos dados de entrada; **SVM_2**, com menor número de vetores de suporte e, assim, o modelo menos complexo gerado; além deles, um modelo intermediário com valores equilibrados para ambos resultados. No entanto, não há um método matemático para definir esse equilíbrio do terceiro modelo. Em geral, eles foram selecionados por inspeção, buscando, dentro do possível, minimizar simultaneamente MSE e SV.

3.5.2 Regressão Linear Múltipla

Como exposto na Seção 3.5, algoritmos alternativos foram propostos para comparação com as SVMs ajustadas pelo NIOTS, dentre eles a **Regressão Linear Múltipla**. A geração desses modelos foi realizada através da biblioteca Scikit-Learn (PEDREGOSA et al., 2011), em sua versão implementada na linguagem Python.

Novamente, devem-se definir os parâmetros de configuração para execução do algoritmo. Para todos os *datasets* testados, as configurações do modelo foram tal como a Tabela

12.

Configurações - Regressão	
Parâmetros	
fit_intercept	True
normalize	False
copy_X	True
n_jobs	None
positive	False

Tabela 12 – Parâmetros da regressão - Scikit Learn

Primeiramente, a *flag* **fit intercept** atua sobre o ponto de intersecção da reta definida pelo modelo com o eixo das ordenadas, ou seja, o coeficiente linear. Defini-la como *False* força que esse coeficiente seja igual a zero, enquanto *True* habilita que seja calculado a partir da curva de melhor ajuste, como exemplificado na Figura 12.

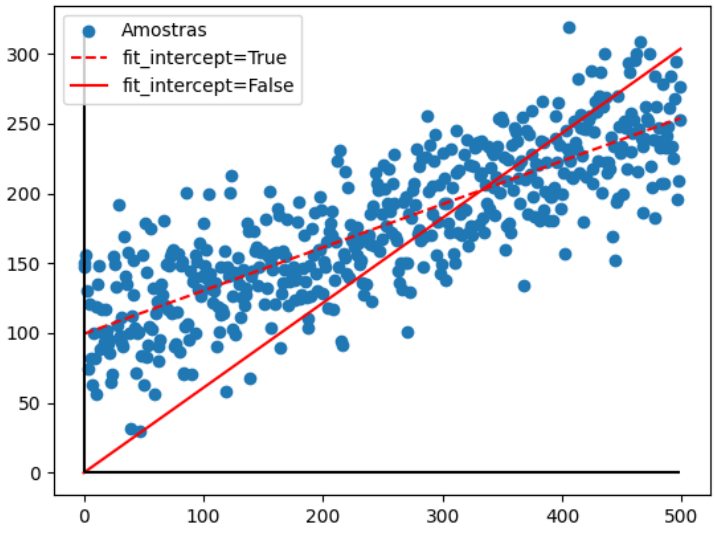


Figura 12 – Exemplificação da *flag* fit_intercept

Enquanto isso, **normalize** define se antes da regressão os dados devem passar por processo de normalização, ou seja, subtração da média e divisão pela norma L2. Por sua vez, **positive** controla se os coeficientes obtidos pelo modelos devem ser, necessariamente, positivos.

Por fim, **copy_X** e **n_jobs** não têm relação com a modelagem em si, apenas com o funcionamento da ferramenta. Definem, respectivamente, se há cópia do vetor de entrada e quantos serão os processos utilizados na execução (caso processamento paralelo seja aplicado).

3.5.3 Random Forest

O segundo algoritmo poposto para comparação foram as **Random Forest**. Novamente, optou-se por utilizar a implementação de regressor em Python da biblioteca Scikit-Learn. Dessa vez, o modelo foi configurado conforme a Tabela 13 para todos os *datasets* testados.

Configurações - Random Forest	
Parâmetros	
n_estimators	100
criterion	squared_error
max_depth	None
min_samples_split	2
min_samples_leaf	1
max_features	auto
bootstrap	True
n_jobs	None
random_state	0
max_samples	None

Tabela 13 – Parâmetros da *random forest* - Scikit Learn

Em comparação à regressão múltipla, *random forests* são algoritmos com maior margem de ajuste em seus parâmetros, que atuam nas seguintes configurações:

- **n_estimators** define o número de estimadores, ou seja, de árvores que serão utilizadas no modelo;
- **criterion**, por sua vez, estabelece a função utilizada para calcular a qualidade de uma divisão;
- **max_depth** é a profundidade máxima (vertical) que a árvore pode atingir;
- **min_samples_split** corresponde ao número mínimo de amostras necessárias para uma divisão;
- **min_samples_leaf** determina a quantidade mínima de amostras em uma folha;
- **bootstrap**, quando verdadeiro, cria novos conjuntos de dados como introduzido na Seção 2.5.3;
- **max_features** delimita quantas são as m variáveis analisadas em uma divisão dentre as p possíveis. Se *auto*, $m = p$;
- **max_samples** é a quantidade de amostragens que o *bootstrap* realiza no conjunto de dados. Quando igual a *None*, a dimensão do conjunto original é mantida;

- **n_jobs** corresponde novamente ao número de núcleos empregados caso se trate de processamento paralelo;
- **random_state** determina um valor usado como semente de aletoriedade tanto no *bootstrap* quanto na seleção de variáveis aleatórias em uma divisão.

3.5.4 Assimetria - SVM

Em geral, a aplicação de algoritmos de regressão tem como objetivo inferir a relação entre variáveis independentes e dependentes. De forma prática, busca-se ajustar um modelo ao conjunto de dados de entrada de forma a descrevê-lo da maneira mais similar possível, sendo essa similaridade determinada através de diversos métodos.

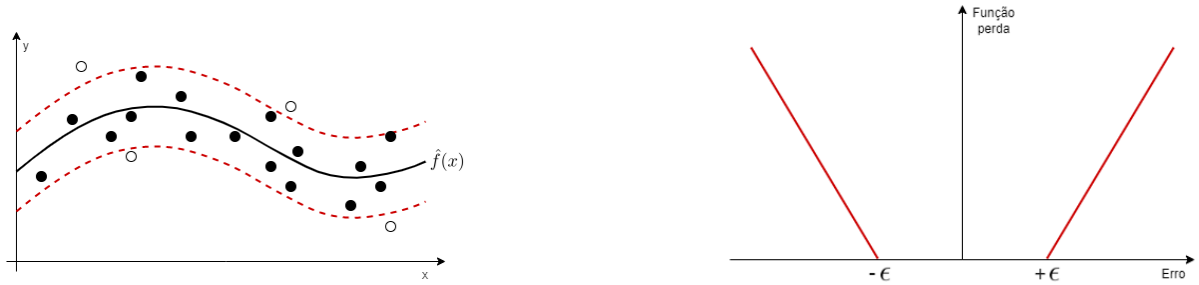
No entanto, no âmbito da Engenharia existem decisões que somente podem ser tomadas a partir de bom senso e experiência do profissional, sem que haja procedimento definido. Essas decisões são denominadas **não-estruturadas** (LAUDON, K. C.; LAUDON, J. P., 2013). Desde a formulação do plano de trabalho, não se considerou que os resultados desse projeto pudessem ser aplicados como solução automatizada em um contexto empresarial. Ao invés disso, os resultados de um modelo treinado deveriam ser avaliados como parte de um **Sistema inteligente de apoio à tomada de decisão**. (LAUDON, K. C.; LAUDON, J. P., 2013) descreve essas ferramentas como capazes de tornar o conhecimento mais disponível a profissionais que tomam decisões e mais integrado aos processos de negócio.

Com base nesse princípio, formulou-se uma nova hipótese:

> *Tendo em vista a definição de **Sistema inteligente de apoio à tomada de decisão** e sua relação com a gerência empresarial, caberia explorar a possibilidade de gerar um modelo flexível? Isto é, com ajuste que permitisse tanto predições pessimistas quanto otimistas.*

Nesse caso, encarregados de atividades relacionadas ao *lead time* poderiam ter à sua disposição um sistema capaz de realizar previsões com limites superior e inferior. Dessa forma, seriam estabelecidas condições de contorno confiáveis à tomada de decisão, que combinadas ao julgamento de um profissional tenderiam a gerar bons resultados.

Investigou-se, portanto, a viabilidade de implementação dessa hipótese por meio de SVMs, foco principal do trabalho. Primeiramente, para estabelecer o referencial teórico necessário, retoma-se o conteúdo da seção 2.2.4. Mais especificamente, a forma da função perda ϵ *insensitive* representada na Figura 13.



(a) Função estimadora (b) Função de perda

Figura 13 – ϵ *insensitive* simétrica

De forma geral, a função delimita um intervalo de insensibilidade no qual não há penalização para predições que atendam à Equação 3.3. Ou seja, sendo y o valor real do conjunto de dados e $\hat{y}$ o valor predito, caso o erro e seja inferior a ϵ o modelo não é penalizado. Observa-se que essa função é aplicada da mesma forma nos semiplanos positivo e negativo e, portanto, é dita **simétrica**.

$$\begin{aligned} e &= |\hat{y} - y|, \\ e &\leq |\epsilon| \end{aligned} \tag{3.3}$$

Com base no exposto, propôs-se implementação da hipótese através de função de perda **assimétrica**, ou seja, que estabeleça comportamento distinto entre os semiplanos. Para desenvolver essa implementação, primeiramente define-se $+\epsilon = 0$. Assim, a função de perda é deslocada tal qual indicado pela Figura 14, na qual a curva tracejada representa a função original. Naturalmente, não se verifica mais intervalo insensitivo no semiplano positivo, deixando então de haver tolerância. Portanto, todos os erros positivos passam a ser prontamente penalizados, o que, teoricamente, implica que passem a ser evitados pelo modelo.

(a) Função estimadora (b) Função de perda

Figura 14 – ϵ *insensitive* assimétrica negativa

Alternativamente, ao definir $-\epsilon = 0$, processo análogo se aplica ao semiplano negativo. A função perda sofre deslocamento tal como indicado na Figura 15, na qual novamente a curva tracejada representa a função original. Dessa vez, o intervalo insensitivo não deve mais

ser verificado no semiplano negativo. Com isso, erros negativos passam a sofrer penalização independente de seu valor e, teoricamente, o modelo deve passar a evitá-los.

(a) Função estimadora

(b) Função de perda

Figura 15 – ϵ *insensitive* assimétrica positiva

Assim, tomando as duas considerações acima, conclui-se que teoricamente é possível estabelecer abordagem que implemente a hipótese proposta através de máquinas de vetores de suporte regressoras (SVR).

No entanto, a ferramenta NIOTS utilizada não conta com tal funcionalidade. Por conta disso, essa alteração foi realizada manualmente acessando um nível abaixo na implementação do projeto, ou seja, a LibSVM (CHANG; LIN, 2011). Essencialmente, verificou-se em qual trecho de código estava definida a variável correspondente a ϵ e modificou-se seu valor. Dada à complexidade da biblioteca, seria inviável exibir tal alteração.

Por fim, de toda forma o NIOTS calcula o valor de ϵ. Como a alteração realizada foi em camada inferior, espera-se que esse processo, de alguma forma, cause perturbações no modelo final.

3.5.5 *Dataset benchmark* de solda

Naturalmente, a hipótese formulada na Seção 3.5.4 deve ser aplicada no conjunto de dados pertencente ao escopo desse projeto. No entanto, esses dados representam processos industriais que se mostram não lineares e complexos, chegando a 61 variávis como indicado na Tabela 9.

Nesse contexto, julgou-se pertinente contar com *dataset* alternativo e simplificado que permitisse avaliar a implementação da hipótese de maneira mais objetiva sem interferência da complexidade dos dados. Assim, selecionou-se estudo de caso da predição de geometria de cordões de solda apresentado em (ALVAREZ BESTARD et al., 2018) e também avaliado por (SANTOS, 2019). Em geral, modelos do processo de soldagem são aplicados na predição da penetração a largura do cordão de solda, a partir de fusão de sensor termográfico responsável pela monitoração da geometria do cordão e pelo controle dos parâmetros do processo.

Por fim, o *dataset* denominado ***benchmark* de solda** tem dimensões indicadas na Tabela 14.

Conjunto	Amostras	Features
Treino	426	8
Teste	94	8
Validação	93	8

Tabela 14 – Divisão do *Dataset benchmark* de solda

4 Resultados

4.1 Análise de dados - *Dataset* reduzido

De início, ressalta-se que em sua implementação, prévia ao presente trabalho, a ferramenta NIOTS aplica apenas o erro quadrático médio (MSE) durante o treinamento e, portanto, o erro absoluto médio percentual (MAPE) será analisado somente para a etapa de testes.

4.1.1 Treinamento

Assim, utilizou-se o NIOTS para treinamento dos 40 modelos-candidatos para o *dataset* reduzido, explicado na seção 3.3.1. Para análise, selecionaram-se os modelos **SVM1**, **SVM2** e **SVM18** com resultados de treinamento indicados na Tabela 15.

Treinamento - Dataset reduzido			
	SVM1	SVM2	SVM18
MSE	47	71	55
SV	315	75	158

Tabela 15 – Restultados de treino - Dataset reduzido

Com base na tabela, observa-se que **SVM1** conta com o menor valor de MSE e **SVM2** a menor quantidade de vetores de suporte (SV), resultado que vai de encontro ao exposto na seção 3.5.1 sobre o funcionamento da ferramenta. Além disso, verifica-se relação inversa entre MSE e SV, que respectivamente se relacionam à capacidade de generalização e à complexidade do modelo, conforme a teoria.

Dessa forma, é coerente concluir que **SVM1** corresponde ao melhor ajuste dos dados de entrada enquanto **SVM2** atende ao compromisso de ser o modelo menos complexo ao custo de uma redução na precisão. Por fim, **SVM18** se mostrou balanceado, ou seja, tanto seu MSE quanto SV apresentaram valores intermediários com relação aos modelos anteriores.

Em seguida, é possível verificar a qualidade do modelo com base em testes estatísticos. Essa verificação pode ser realizada a partir da auto-correlação dos resíduos e funções de correlação cruzada entre os mesmos resíduos e os valores de entrada (BILLINGS; VOON, 1986)(BILLINGS; JAMALUDDIN; CHEN, 1992). De forma geral, quando a dinâmica entre entrada e saída foi completamente capturada pelo modelo, os resíduos devem ser aleatórios e independetes (descorrelacionados) da entrada.

De acordo com (BILLINGS; VOON, 1986), para dados não lineares deve ser atendido o conjunto de testes descrito na Equação 4.1, na qual as duas primeiras funções verificam

correlações lineares e as três últimas correlações não lineares. Nesse contexto, δ representa a função delta de Kronecker e ϕ_{ab} corresponde à correlação cruzada normalizada entre os vetores **a** e **b** (BILLINGS, 2013). Além disso, y simboliza rótulos de treinamento, x as variáveis preditores e τ indica o deslocamento (atraso ou avanço) aplicado pela função de correlação cruzada. O resultado desses testes deve ser avaliado dentro de uma margem de confiança de 95%, representada graficamente.

$$\begin{cases} \phi_{\xi\xi}(\tau) = \delta(\tau) \\ \phi_{y\xi}(\tau) = 0, \quad \forall\tau \\ \phi_{\xi(\xi y)}(\tau) = 0, \quad \tau \geq 0 \\ \phi_{(y^2)'\xi}(\tau) = 0, \quad \forall\tau \\ \phi_{(y^2)'\xi^2}(\tau) = 0, \quad \forall\tau \end{cases} \longrightarrow \begin{array}{l} (y^2)' = y^2 - \bar{y}^2, \\ (\xi y) = \xi(x+1)y(x+1) \end{array} \tag{4.1}$$

A Figura 16 contempla o resultado gráfico da implementação da Equação 4.1 para o modelo SVM1. Observa-se que a auto-correlação (curva superior) é verificada somente na região de deslocamento zero, como previsto pela função delta de Kronecker (BILLINGS, 2013). As demais funções de correlação cruzada retornam valores próximos a zero dentro da margem de confiança estabelecida e representada em pontilhado.

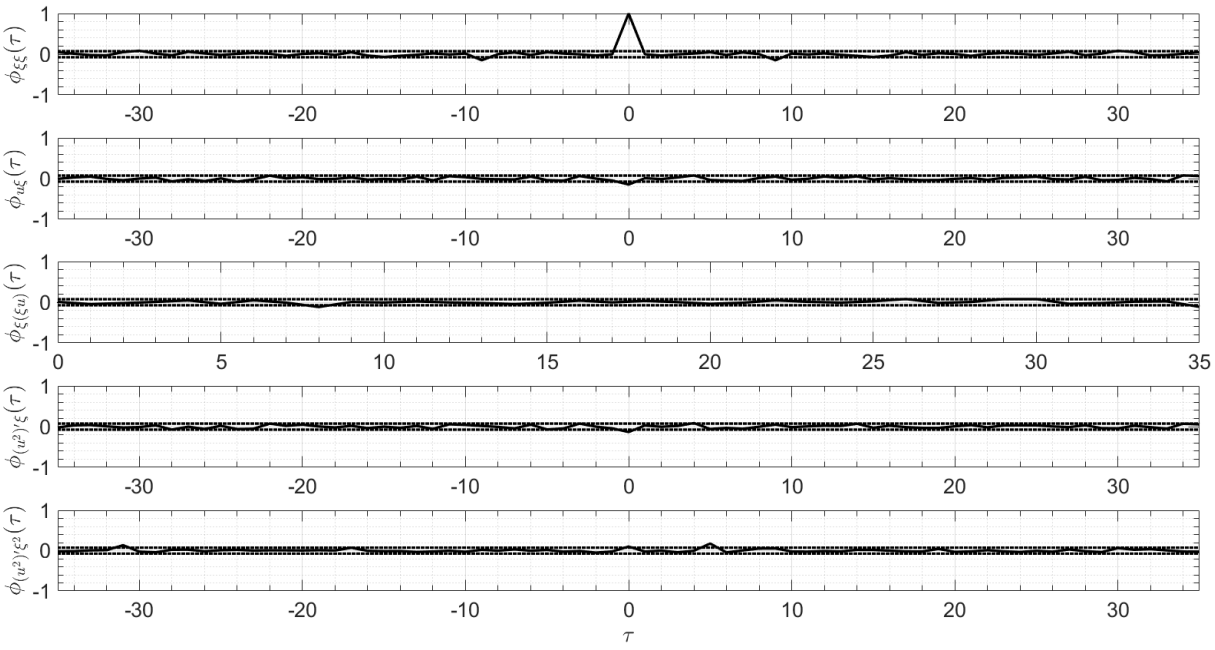


Figura 16 – Curvas do modelo SVM1 - Testes estatíticos

Resultados análogos foram verificados para os modelos SVM2 e SVM18, como indicado nas Figuras 17 e 18.

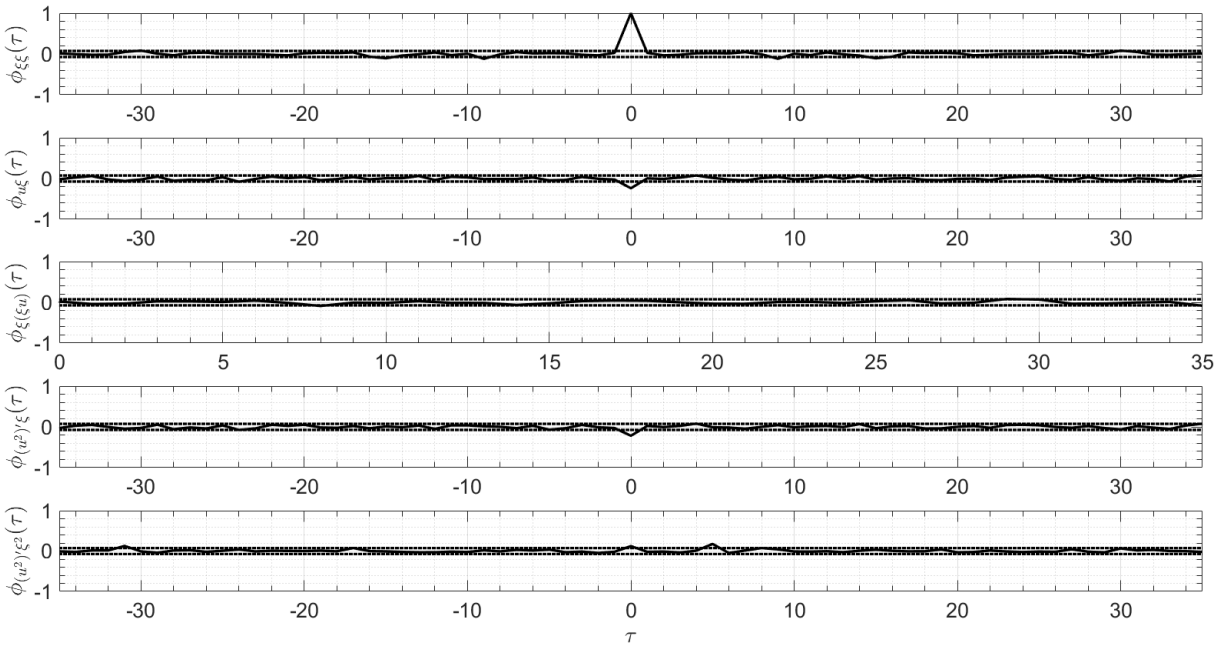


Figura 17 – Curvas do modelo SVM 2 - Testes estatíticos

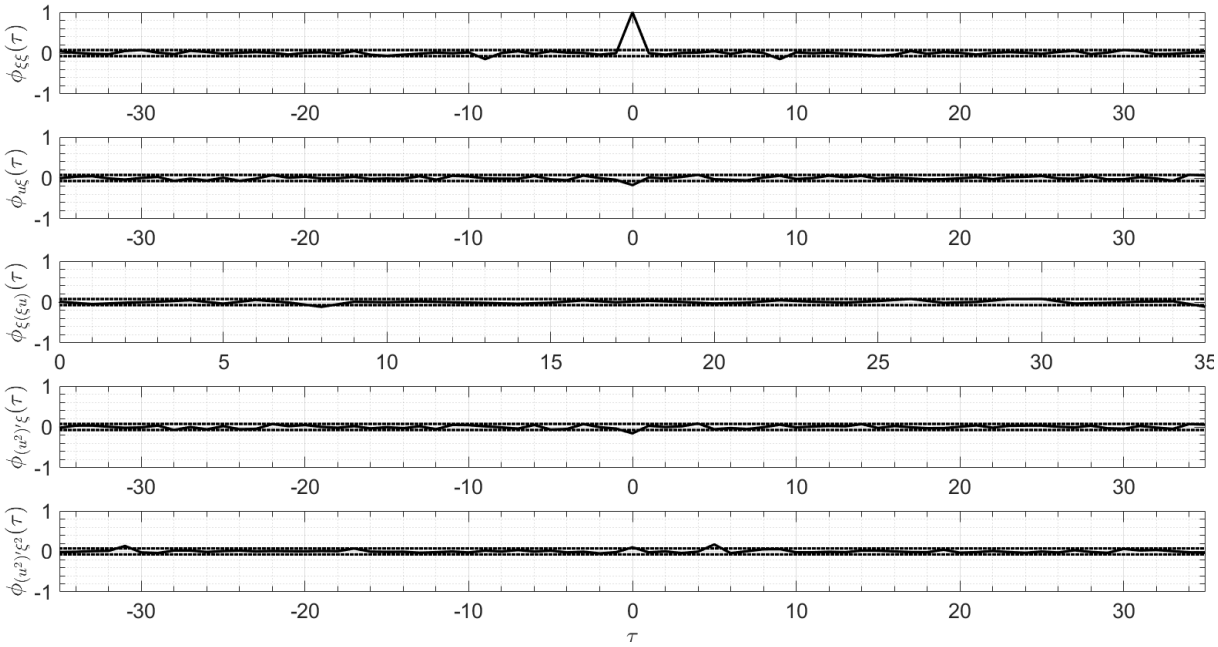


Figura 18 – Curvas do modelo SVM 18 - Testes estatíticos

Com base no exposto, foi possível verificar que os testes estatísticos atenderam às funções da Equação 4.1. De forma geral, os resultados apontam que os modelos indentificaram a informação disponível nos dados, ou seja, que o NIOTS pode ser aplicado de maneira satisfatória na modelagem desse problema.

4.1.2 Teste

Em seguida, tomando o conjunto de teste do *dataset* reduzido, executou-se a ferramenta em modo de predição com cada um dos modelos selecionados anteriormente. Dessa vez, obteve-se os valores resultantes de MSE e MAPE, além do histograma de erros e a representação gráfica das curvas real e predita sobrepostas. Por fim, os resultados foram sintetizados na Tabela 16.

Teste - Dataset reduzido			
	SVM1	SVM2	SVM18
MSE	75	106	85
MAPE	36	84	65

Tabela 16 – Restultados de teste - Dataset reduzido

Analisando a Tabela 16 para os resultados de teste, observa-se, para todos os modelos, MSE de teste superior ao de treino. Novamente, essa observação pode ser explicada com base na teoria, pois os conjuntos de treino e teste são distintos. Assim, a generalização resultante da primeira etapa pode não se estender perfeitamente à segunda. Consequentemente, em geral, são esperados mais erros em teste.

Embora seja uma métrica importante e extremamente popular, os valores retornados pelo MSE muitas vezes acabam sendo complexos para analisar de forma quantitativa, principalmente pelo fator quadrático que altera a unidade de medida. Antecipando essa questão, estabeleceu-se o cálculo do MAPE, como justificado na Seção 3.4. Este, por outro lado, retorna percentual de erro com relação ao valor real, tendo sido verificado o melhor resultado em **SVM1** com 36%, que corresponde à diferença média entre as curvas real (laranjada) e predita (azul) da Figura 19.

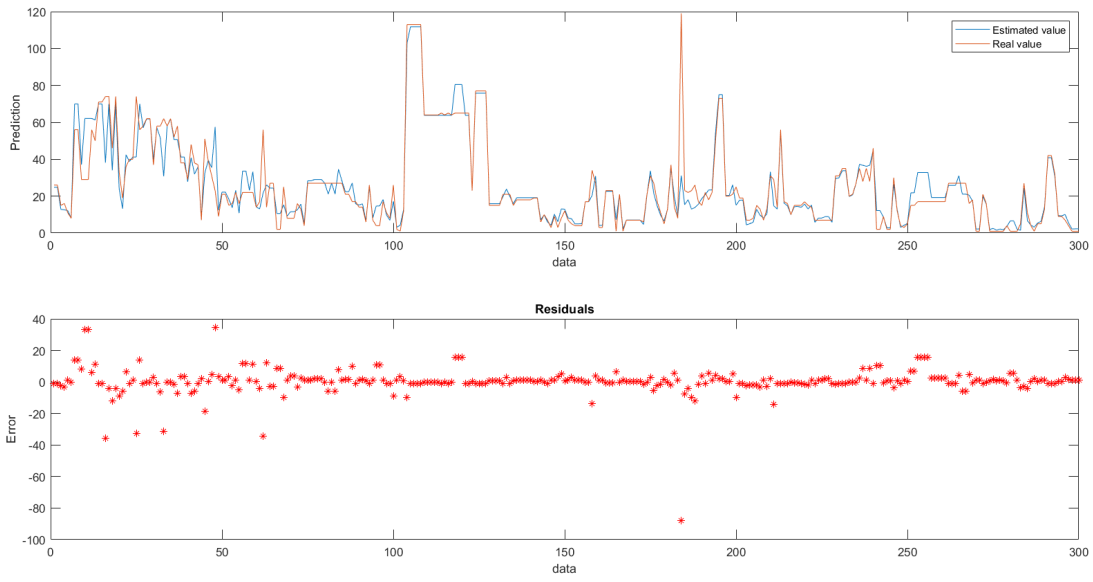


Figura 19 – Curvas do modelo SVM1 - Dataset reduzido

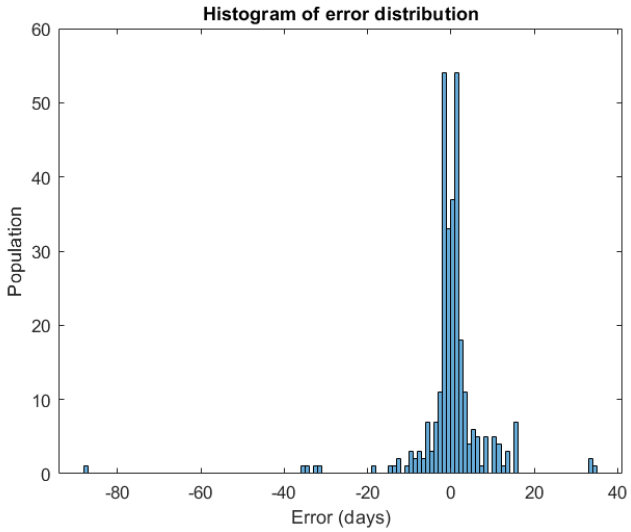


Figura 20 – Histograma do modelo SVM1 - Dataset reduzido

As Figuras 19 e 20 apresentam graficamente o resultado obtido. Na Figura 19 é possível observar que a curva azul, predita, se ajusta significativamente à curva real.

Complementarmente, a Figura 20 apresenta o histograma de erro para o teste. Como mencionado na Seção 2.5, espera-se que o maior número de amostras possíveis apresentem erro de estimação mínimo. Ou seja, que as barras do histograma se concentrem em torno de zero, fato verificado para esse modelo. Além disso, constata-se que os piores casos apresentam erro em torno de 40 e 80 dias, que são extremamente elevados. Em um contexto de aplicação fabril, pode-se concluir que seria possível estabalecer investigação dessas amostras a fim de identificar se houve imprevistos durante a fabricação, alguma incoerência nos registros ou mesmo se o modelo precisa de maior refinamento.

4.1.3 Análise comparativa

Com base no exposto, pode-se estabelecer uma comparação entre os três modelos avaliados levando em conta **capacidade de generalização** e **complexidade**. Para tal, deve-se definir como serão quantificados cada um desses parâmetros.

Primeiramente, generalização indica capacidade do modelo em se ajustar corretamente ao conjunto de dados. Portanto, pode ser definida de forma inversa ao MAPE do modelo, ou seja, quanto menor o erro, maior deve ser a generalização, definida como g do modelo M de acordo com a Equação 4.2.

$$g(M) = \frac{min(MAPE)}{MAPE(M)} \cdot 100 \tag{4.2}$$

Por outro lado, a complexidade de uma SVM é resultante da quantidade de vetores de suporte usados em sua construção. Portanto, é possível definir complexidade c relativa ao

modelo M na forma da Equação 4.3.

$$c(M) = SV(M) \tag{4.3}$$

Finalmente, deve-se estabelecer a comparação entre esses resultados. A fim de padronizar a escala, propôs-se normalização percentual relativa ao valor máximo dentre cada um dos parâmetros. A título de exemplo, será atribuído valor de 100% ao modelo correspondente à maior complexidade e os demais serão relacionados a ele. O resultado foi sintetizado na Tabela 17 e no gráfico de colunas da Figura 21.

Comparação percentual - Dataset reduzido			
	SVM1	SVM2	SVM18
Generalização	100	42	55
Complexidade	100	23	50

Tabela 17 – Comparação percentual - Dataset reduzido

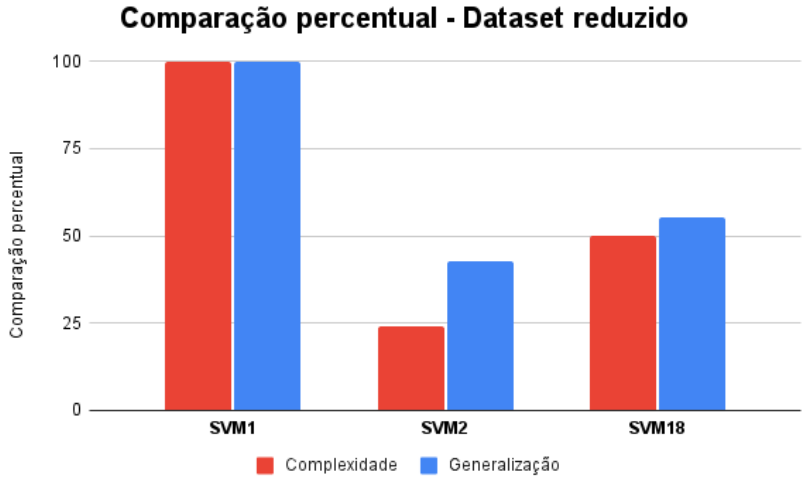


Figura 21 – Comparação percentual - Dataset reduzido

Como descrito anteriormente, **SVM1** é o modelo com melhor generalização ao custo de simultaneamente ser o mais complexo dentre os três. Contudo, o resultado mais expressivo da análise comparativa é a relação direta entre complexidade e generalização. Ou seja, futuras tentativas de diminuição da complexidade tendem a implicar redução da generalização.

4.2 Análise de dados - *Dataset* filtrado

Analogamente à seção anterior, a ferramenta NIOTS foi utilizada para modelar o *dataset* filtrado, composto por conjuntos de dados nos quais as variáveis **Conjunto** e **Descrição de Material** foram filtradas em suas quatro categorias mais populosas. Isto é, um total de oito conjuntos de dados e oito processos de modelagem realizados.

Novamente, 40 modelos-candidatos foram retornados em cada processo e, dentre eles, selecionaram-se para análise **SVM1**, **SVM2** e um terceiro que apresentasse resultados equilibrados. No entanto, esse último modelo foi distinto em cada um dos casos de teste. Para fins de simplificação, adotou-se generalização de forma que esses modelos serão referenciados por **SVM10**.

4.2.1 Treinamento

A Tabela 18 apresenta o resultado dos modelos obtidos para cada um dos filtros aplicados ao conjunto de treinamento. Observam-se resultados análogos ao *dataset* anterior. Isto é, em todos os filtros aplicados **SVM1** conta com o menor valor de MSE e **SVM2** a menor quantidade de vetores de suporte (SV), funcionamento já esperado da ferramenta, como descrito na seção 3.5.1.

Por conta disso, é possível concluir novamente que **SVM1** correspondeu ao ajuste mais fiel dos dados de entrada enquanto **SVM2** atendeu ao compromisso de apresentar a menor complexidado ao custo de aumento no erro médio. Por fim, **SVM10** se mostrou balanceado, ou seja, tanto os valores de MSE quanto SV se mostraram intermediários com relação aos modelos anteriores.

4.2.2 Teste

Na sequência, foram realizadas predições a partir do conjunto de teste e de cada um dos modelos treinados anteriormente (**SVM1**, **SVM2** e **SVM10**) para todos casos propostos. Mais uma vez, o produto dessa etapa consiste nos valores de MSE e MAPE, além do histograma de erros e as representações gráficas das curvas real e predita sobrepostas. Tais resultados foram sintetizados na Tabela 19.

Como explicado na Seção 3.4 e retomado em 4.1.2, o MAPE oferece vantagens em relação ao MSE e portanto foi o foco da análise. É importante ressaltar que esse erro pode superar o valor de 100%, uma vez que isso significa que a curva predita é duas vezes maior que a curva real, em média.

Com base nos valores de MAPE relacionados na Tabela 19, verifica-se que **SVM2** e **SVM10**, em geral, apresentam desempenho inferior a **SVM1**, sendo o filtro **Al_6061** a única exceção. Nos demais casos, o MAPE dos modelos **SVM10** e **SVM2** chega a 60% e 100%, respectivamente.

A fim de sintetizar essa observação, compilou-se a Tabela 20 com o valor médio de erro para as variáveis filtradas, ou seja, **Conjunto** e **Material**, destacando em negrito o menor valor de MAPE para cada uma delas. É natural observar que ambas pertencem a **SVM1** e, portanto, pode-se concluir que esse é o modelo com melhor capacidade de generalização.

Essa conclusão corrobora tanto o funcionamento da ferramenta, exposto na Seção 3.5.1 quanto a análise comparativa realizada para o *dataset* reduzido em 4.1.3.

Tabela 18 – Treinamento - *dataset* filtrado

Treinamento			
	SVM1	SVM2	SVM10
Conjunto ADS10			
MSE	28	70	35
SV	152	30	80
Conjunto LaserA			
MSE	30	59	34
SV	115	20	62
Conjunto MUX			
MSE	15	35	20
SV	202	21	53
Conjunto MUX_FREE			
MSE	22	46	27
SV	148	30	72
Material AISI_304			
MSE	82	112	91
SV	267	108	152
Material Al_6061			
MSE	135	145	141
SV	242	98	108
Material Al_7075			
MSE	231	281	245
SV	312	172	227
Material Al_6351			
MSE	96	117	101
SV	840	312	458

Tabela 19 – Teste - *dataset* filtrado

Teste			
	SVM1	SVM2	SVM10
Conjunto ADS10			
MSE	44	61	22
MAPE	27	108	50
Conjunto LaserA			
MSE	28	52	28
MAPE	43	117	63
Conjunto MUX			
MSE	32	47	29
MAPE	15	102	56
Conjunto MUX_FREE			
MSE	25	50	29
MAPE	33	94	56
Material AISI_304			
MSE	93	104	94
MAPE	32	79	60
Material Al_6061			
MSE	129	174	171
MAPE	49	50	40
Material Al_7075			
MSE	318	181	259
MAPE	48	48	50
Material Al_6351			
MSE	81	102	85
MAPE	41	95	62

Médias - Resultado de Teste						
	SVM1		SVM2		SVM10	
Variável	MSE	MAPE	MSE	MAPE	MSE	MAPE
Conjunto	32,25	**29,5**	49	99,25	27	56,25
Material	155,25	**42,5**	140,25	68	152,25	53

Tabela 20 – Médias de teste - *dataset* filtrado

Em seguida, verifica-se que a variável **Conjunto**, em média, apresenta MAPE inferior a **Material**, levando a uma conclusão preliminar de que é benéfico ao modelo filtrar a partir da primeira. Para corroborar esse ponto, compararam-se os resultados do *dataset* filtrado aos do *dataset* reduzido através das Tabelas 20 e 16. Tomando o modelo **SVM1**, é possível observar que aplicar filtro à variável **Conjunto** diminuiu o MAPE de teste obtido quando comparado aos dados sem filtro. Por outro lado, fazer o mesmo com **Material** aumentou esse erro. Portanto, conclui-se de fato que há melhora no modelo aplicando filtro à variável **Conjunto**.

Por fim, as Figuras 22 a 25 correspondem ao melhor e ao pior resultado de MAPE da Tabela 19, ou seja, modelo SVM1 para o filtro MUX e modelo SVM2 para o filtro LaserA, respectivamente.

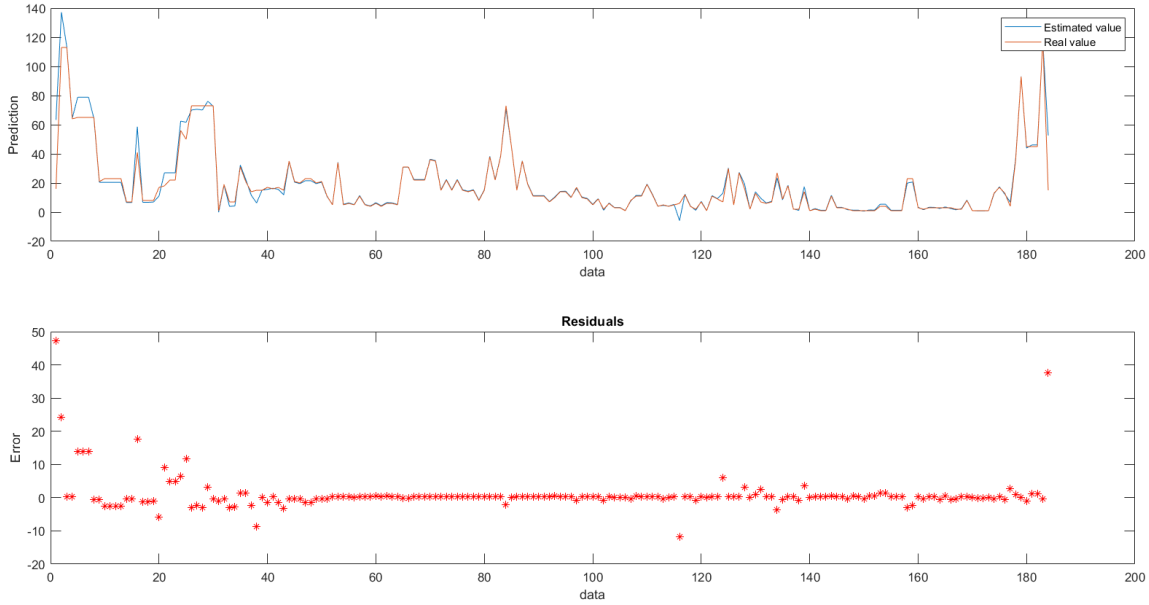


Figura 22 – Curva para o filtro MUX - modelo SVM1 (melhor caso)

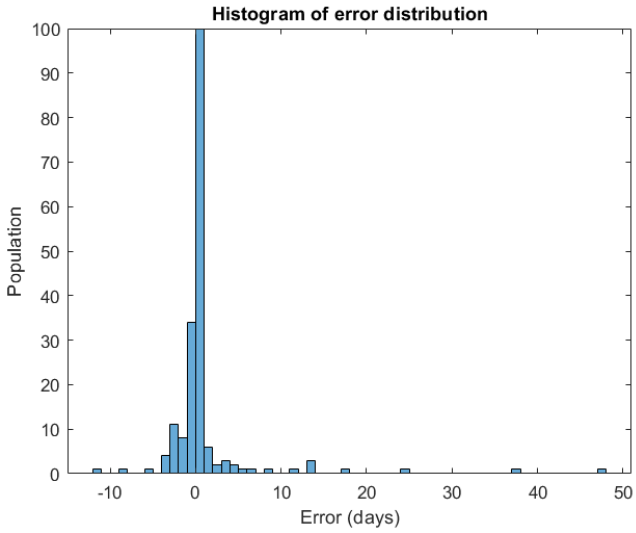


Figura 23 – Histograma para o filtro MUX - modelo SVM1 (melhor caso)

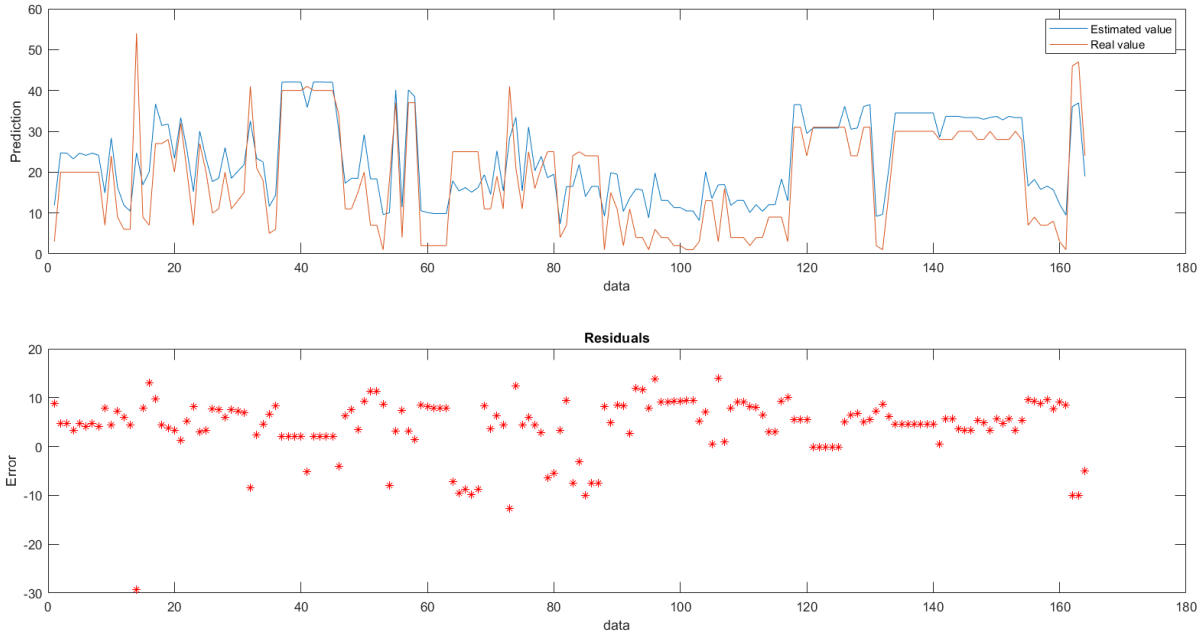


Figura 24 – Curva para o filtro LaserA - modelo SVM2 (pior caso)

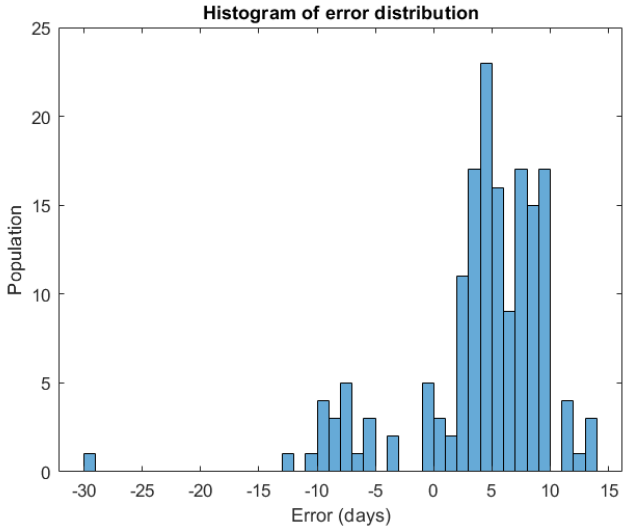


Figura 25 – Histograma para o filtro LaserA - modelo SVM2 (pior caso)

Nitidamente, a curva predita da Figura 22, referida como melhor caso, se mostra muito mais ajustada aos dados quando comparada à curva da Figura 24. De maneira complementar, o histograma da Figura 23 apresenta alta concentração em torno de erro zero, enquanto a Figura 25 tem barras muito mais dispersas. Dessa forma, conclui-se que os resultados gráficos refletem os numéricos, contribuindo efetivamente na análise. Portanto, considera-se que ao aplicar a metodologia proposta por esse trabalho em ambiente de produção industrial, ambos resultados gráficos podem ser utilizados de maneira qualitativa pelos responsáveis técnicos para avaliação do modelo.

4.3 Análise de dados - *Dataset* final

Como proposto na Seção 3.5, o estudo exploratório de máquinas de vetores de suporte, foco desse trabalho, seria realizado comparando seus resultados aos de algoritmos alternativos aplicados sobre o mesmo conjunto de dados. São eles **regressão linear múltipla** e ***random forests***.

As análises anteriores, discutidas nas Seções 4.1 e 4.2, estabeleceram critérios para seleção dos modelos e do *dataset* a serem utilizados. Primeiramente, verificou-se que na aplicação de SVMs com a ferramenta NIOTS, que retorna diversos resultados, **SVM1** tende, em média, a apresentar melhor capacidade de generalização. Portanto, esse foi o modelo selecionado para análises seguintes. Alternativamente, observou-se que a aplicação de filtros foi mais vantajosa quando realizada sobre a variável **Conjuntos**. Assim, utilizou-se esse procedimento para construção do *dataset* final, como exposto na Seção 3.3.3.

Consequentemente, os resultados apresentados a seguir foram obtidos com base na aplicação de máquinas de vetores de suporte (modelo **SVM1**), regressão linear múltipla e *random forest* sobre o *dataset* final.

4.3.1 SVM - NIOTS

Executou-se a ferramenta NIOTS conforme configuração da Tabela 10 tendo como entrada o *dataset* final, que por sua vez teve dimensões especificadas na Tabela 9. Assim, realizou-se o processo de treinamento, no qual foi retornado o modelo **SVM1**. Em seguida, a partir desse modelo, foram realizadas predições com base no conjunto de testes e os resultados obtidos sintetizados na Tabela 21.

Teste - Dataset final	
SVM	
MSE	22
MAPE	29

Tabela 21 – Resultados SVM - *Dataset* final

Primeiramente, comparando as Tabelas 21 e 20, observa-se proximidade expressiva entre os valores de MAPE do procedimento de teste para o *dataset* final (29%) e para o filtrado a partir de **Conjunto** (29,5%). De forma geral, tal resultado é coerente já que o *dataset* final foi construído como uma condensação do outro. Portanto, era esperado teoricamente que a generalização se estendesse entre eles.

Novamente, um dos produtos de teste do modelo é a representação gráfica presente na Figura 26. Observa-se que, em geral, a curva estimada (azul) se ajusta à real (laranja), o que pode ser analisado como uma perspectiva visual da generalização do modelo. No entanto, ainda existem - e devem existir - divergências entre essas curvas, entre as quais se

destaca a região inicial em torno da amostra 50. Levando em conta que os dados de teste não foram expostos ao modelo anteriormente, são dessas regiões que advém o erro observado.

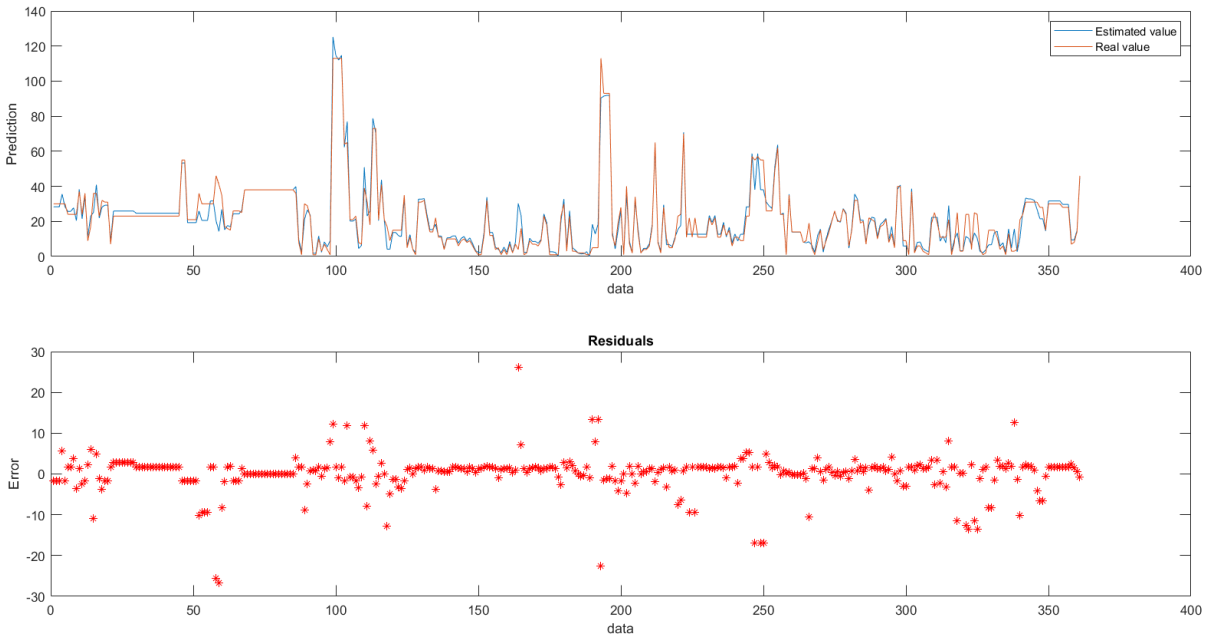


Figura 26 – Curvas de teste do *dataset* final - SVM

Além disso, também é gerado histograma de erro conforme apresentado na Seção 3.4 e exposto na Figura 27. Em suma, o eixo das abscissas define intervalos de erro em dias enquanto as ordenadas indicam o total de amostras - denomindado população - com erro de predição correspondente a cada intervalo. Nota-se que houve distribuição em torno de zero, observada pelas barras mais compridas do histograma agrupadas na região central. Esse fato indica que, em geral, os erros de predição de *lead time* são de poucos dias.

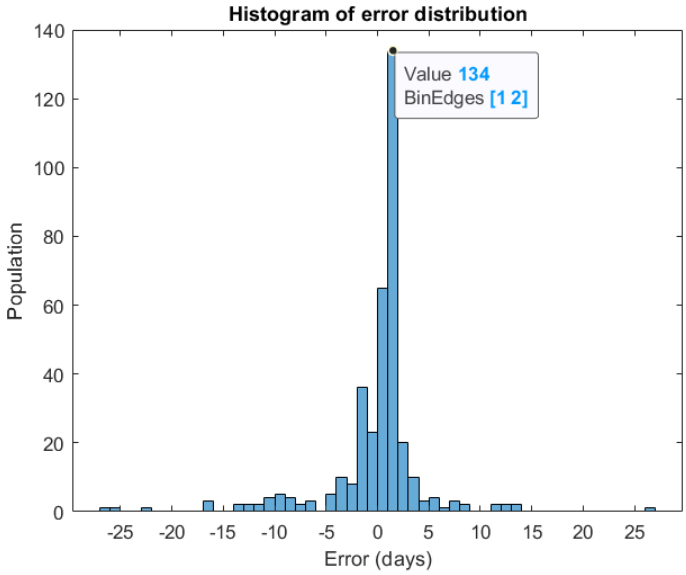


Figura 27 – Histograma de erro do *dataset* final - SVM

Para o caso analisado, destacou-se a maior barra. Ela indica que 134 amostras apresentam erro no intervalo [1,2], o que corresponde a 37% do total de 361 predições.

4.3.2 Regressão Linear Múltipla

Alternativamente, executou-se a função de regressão linear múltipla da biblioteca **Scikit-Learn** conforme configuração da Tabela 12 tendo como entrada o *dataset* final, com dimensões especificadas na Tabela 9. No entanto, esse algoritmo não utiliza validação semelhante à do NIOTS e os dados de validação e teste foram combinados em um único conjunto com dimensões (721x61).

A partir do modelo gerado, realizaram-se predições com base no conjunto de testes e os resultados obtidos sintetizados na Tabela 22.

Teste - Dataset final	
Regressão múltipla	
MSE	35
MAPE	35

Tabela 22 – Resultados regressão múltipla - *Dataset* final

Primeiramente, comparando as Tabelas 22 e 21, observa-se que modelando a partir de regressão os erros aumentam quando em comparação ao resultado da SVM. Analisando o MAPE, que foi adotado como prioritário nessa análise, verifica-se piora na predição de 6% em média.

Além disso, gerou-se representação gráfica dos resultados conforme a Figura 28. Em geral, a curva estimada (azul) se ajustou à real (laranja), o que pode ser analisado como uma perspectiva visual da generalização do modelo. No entanto, da mesma forma como nas SVM, ainda houve divergências entre as duas curvas, principalmente nas 200 primeiras amostras. Mais uma vez, são dessas regiões que advém o erro observado - como esperado - já que os dados de teste são desconhecidos ao modelo.

Por fim, gerou-se o histograma de erro da Figura 29 de acordo com o apresentado na Seção 3.4. Em suma, o eixo das abscissas define intervalos de erro em dias enquanto as ordenadas indicam o total de amostras - denomindado população - com erro de predição correspondente a cada intervalo. Novamente, verifica-se distribuição em torno de zero, observada pelas barras mais compridas do histograma agrupadas na região central, fato que indica que, em geral, os erros de predição de *lead time* são de poucos dias.

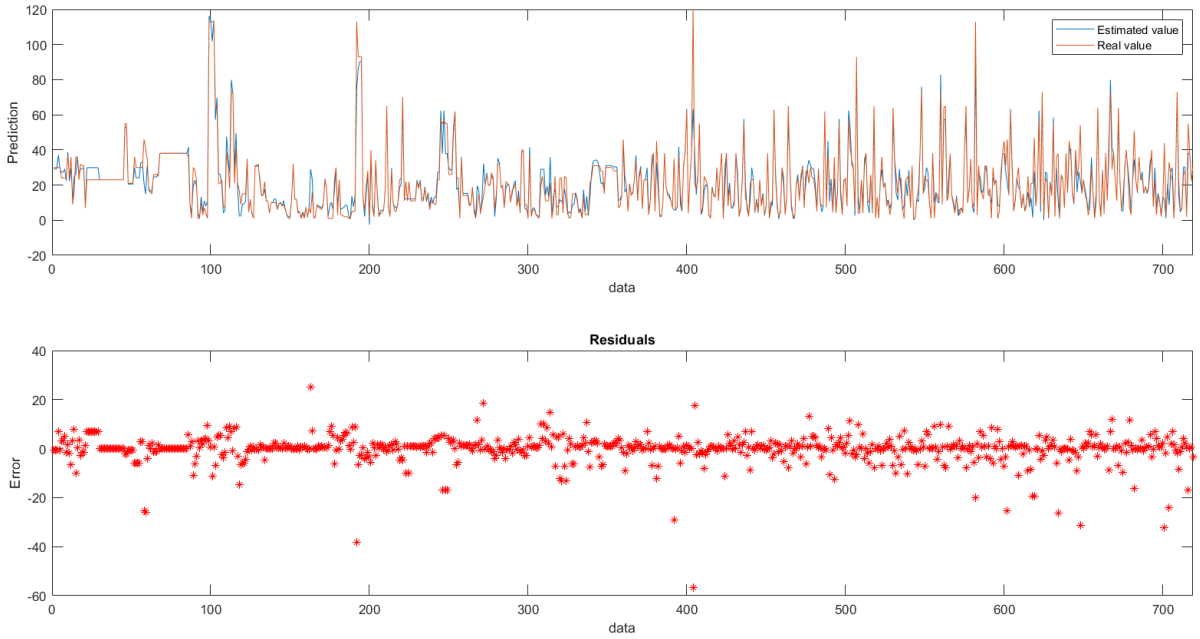


Figura 28 – Curvas de teste do *dataset* final - regressão múltipla

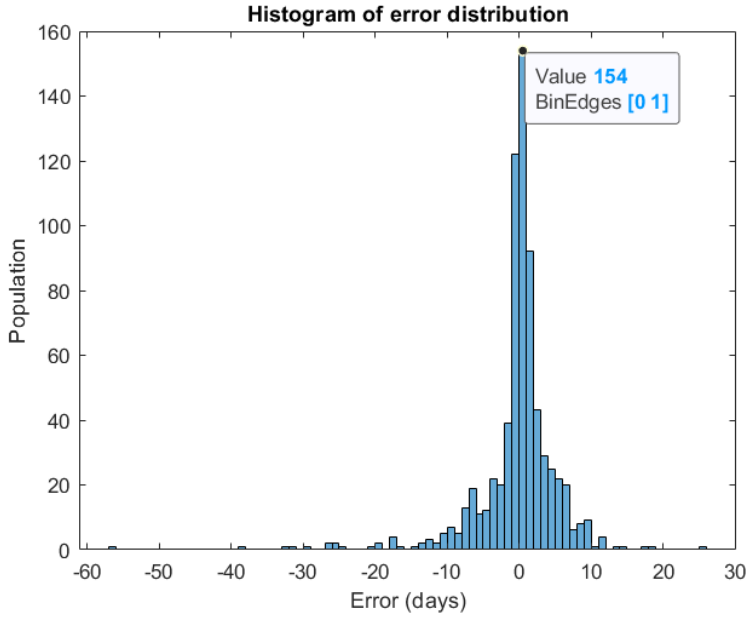


Figura 29 – Histograma de erro do *dataset* final - regressão múltipla

Destacando a maior barra, 154 amostras apresentam erro no intervalo [0,1], o que corresponde a 21% do total de 721 predições. Com base nisso e em comparação com a Figura 27, das SVM, nota-se que, apesar da distribuição central, o histograma obtido da regressão múltipla é mais populoso na região $[-10,10]$. Tal resultado visual corrobora os valores numéricos obtidos, já que o erro de fato aumentou.

4.3.3 *Random forest*

Finalizando a análise dos algoritmos propostos, executou-se a função de *random forest*, também da biblioteca **Scikit-Learn**, conforme configuração da Tabela 13 para o *dataset* final, com dimensões especificadas na Tabela 9. Analogamente à regressão linear, esse algoritmo não aplica validação da mesma forma que o NIOTS e os dados de validação e teste foram combinados em único conjunto com dimensões (721x61). Após treinamento do modelo, realizaram-se predições com base nesse conjunto de testes e os resultados obtidos foram sintetizados na Tabela 23.

Teste - Dataset final	
Random forest	
MSE	33
MAPE	17

Tabela 23 – Resultados *random forest - Dataset* final

Primeiramente, comparando as Tabelas 23, 22 e 21, verifica-se que a implementação de regressão por *random forest* apresenta o menor MAPE dentre os três algoritmos. Mais especificamente, o erro é 12% e 18% menor quando comparado às SVM e à regressão múltipla, respectivamente.

Além do mais, obteve-se representação gráfica dos resultados apresentada na Figura 30. Observou-se ajuste da curva estimada (azul) à real (laranja), o que pode ser analisado como uma perspectiva visual da generalização do modelo já indicada pelos erros da Tabela 23. Ainda assim, houve divergências entre as duas curvas, da mesma forma como nos algoritmos anteriores. Mais uma vez, são essas regiões que implicam o erro observado - e esperado - já que os dados de teste são desconhecidos ao modelo.

Finalmente, obteve-se histograma de erro conforme a Figura 31 de acordo com o apresentado na Seção 3.4. Em suma, o eixo das abscissas define intervalos de erro em dias enquanto as ordenadas indicam o total de amostras - denomindado população - com erro de predição correspondente a cada intervalo. Novamente, ocorreu distribuição em torno de zero, observada por conta das barras mais compridas se acumulatem na região central. De forma geral, isso indica que os erros de predição de *lead time* são de poucos dias.

Destacando a maior barra, 283 amostras apresentam erro no intervalo [0,1], correspondendo a 39% do total de 721 predições.

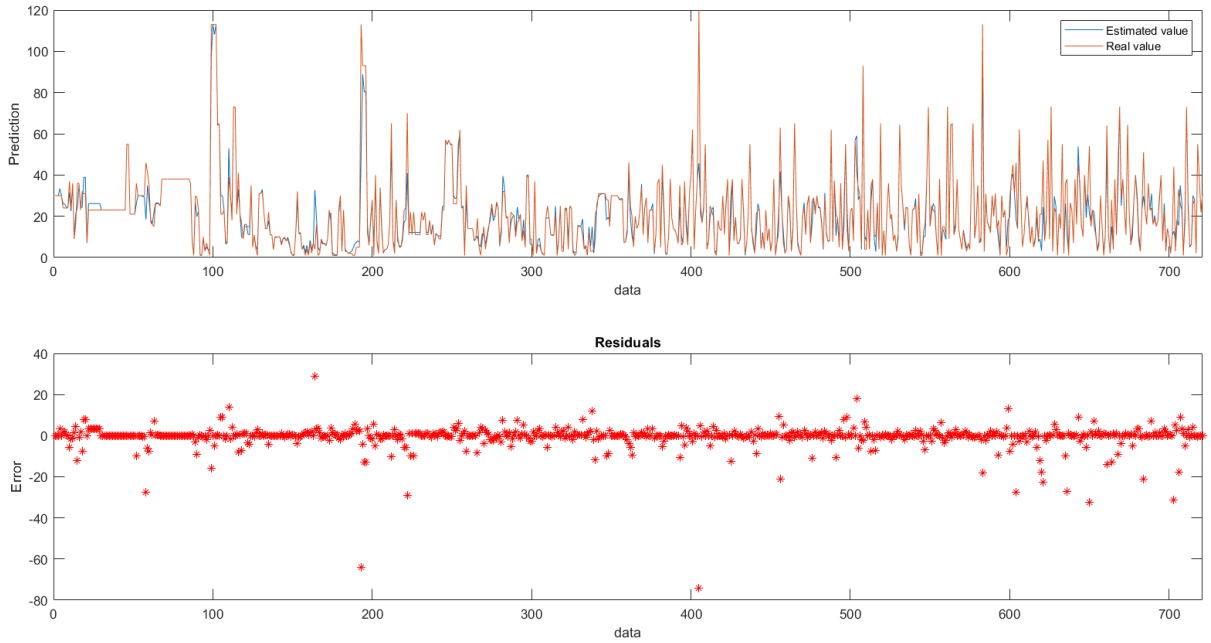


Figura 30 – Curvas de teste do *dataset* final -*random forest*

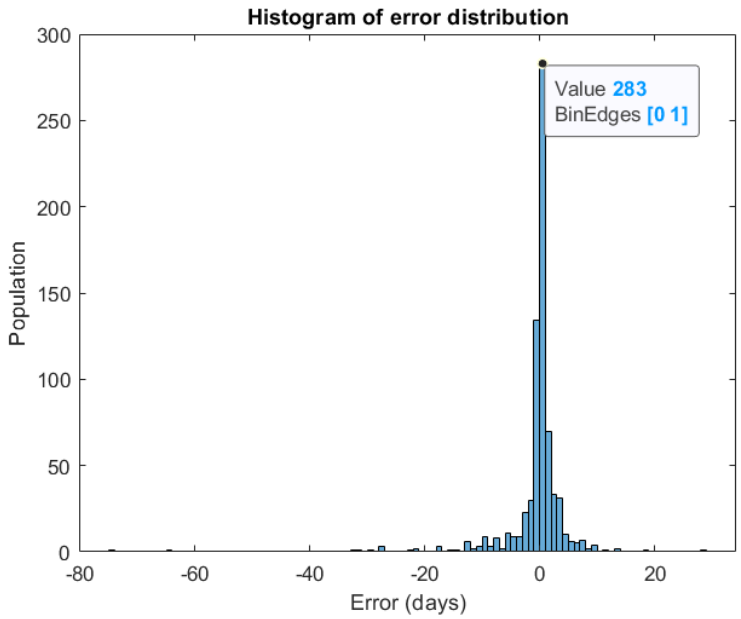


Figura 31 – Histograma de erro do *dataset* final - *random forest*

4.3.4 Análise comparativa

Como forma de avaliar os três algoritmos testados, propõe-se comparação entre suas **capacidades de generalização** e **complexidades**. Para tal, é necessário definir como quantificar cada um desses parâmetros de forma análoga à realizada na Seção 4.1.3.

Primeiramente, generalização indica capacidade do modelo em se ajustar corretamente ao conjunto de dados. Portanto, pode ser definida de forma inversa ao MAPE do modelo, ou seja, quanto menor o erro, maior deve ser a generalização, definida como g do

modelo M novamente de acordo com a Equação 4.2.

Por outro lado, na Seção 4.1.3 utilizou-se o número de SV das SVM como métrica de complexidade. Nessa análise, que trata de modelos distintos, não seria possível prosseguir dessa forma. Portanto, optou-se por verificar o tempo computacional (CPU) médio para treinamento dos modelos em 5 execuções, definindo a complexidade c relativa ao modelo M na forma da Equação 4.4.

$$c(M) = \frac{1}{5}\sum_{i=1}^{5} tempo(M) \tag{4.4}$$

Finalmente, deve-se estabelecer a comparação entre os resultados obtidos. Mais uma vez, buscando padronizar a escala, propôs-se normalização percentual relativa ao valor máximo dentre cada um dos parâmetros. A título de exemplo, será atribuído valor de 100% ao modelo correspondente à maior complexidade e os demais serão quantificados relativamente a ele.

No entanto, foi necessário considerar que o modelo obtido pela ferramenta NIOTS passa por extensas iterações de otimização no ajuste de seus hiperparâmetros. Como esse processo é contabilizado, eleva o tempo total observado. Portanto, o tempo de treinamento da ferramenta foi relacionado mas não será comparado aos demais.

O resultado foi sintetizado na Tabela 24 e no gráfico de colunas da Figura 32.

Comparação - Dataset final			
	SVM	Regressão múltipla	*Random forest*
MAPE (%)	29	35	17
Generalização (%)	**58**	**48**	**100**
Tempo (s)	16,925	1,945	3,558
Complexidade (%)	**-**	**54**	**100**

Tabela 24 – Comparação percentual - Dataset final

Com base na Tabela 24 e na Figura 32, observa-se que as *random forest* se mostraram, em média, modelos duas vezes mais complexos em comparação à regressão linear múltipla baseado na análise temporal de sua execução.

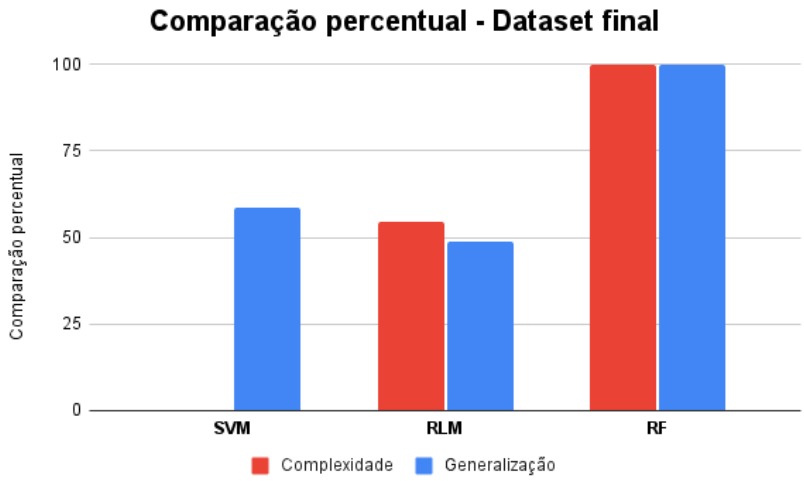


Figura 32 – Comparação percentual - Dataset final

Todavia, de forma geral, constatou-se que *random forest* supera os demais algoritmos no quesito generalização, pois apresenta MAPE expressivamente menor. Primeiramente, analisando essa implementação e a regressão linear múltipla, James et al. prevê que a comparação entre tais algoritmos depende do conjunto de dados a ser testado. Caso a relação entre as variáveis seja aproximadamente linear, é esperado que a regressão múltipla se sobressaia. Por outro lado, em caso de não linearidade e de relações complexas entre essas variáveis, métodos baseados em árvore tendem a entregar melhores resultados (JAMES et al., 2013).

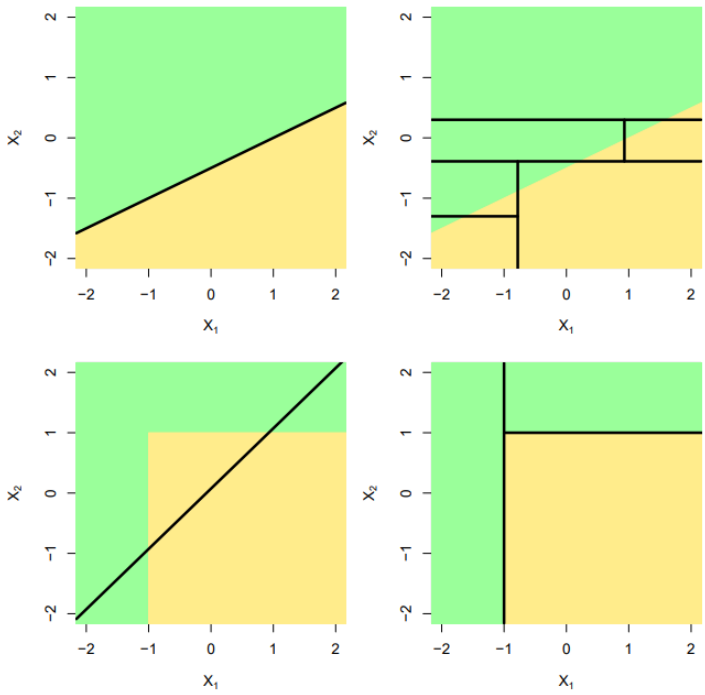


Figura 33 – Adaptação - Comparação entre modelos lineares e baseados em árvores

As imagens superiores da Figura 33 (JAMES et al., 2013), representam como modelos

lineares (à esquerda) e árvores (à direita) modelam um problema linear. Por outro lado, as imagens inferiores indicam como ambos modelos se ajustam a uma fronteira de decisão não linear, com vasta vasta vantagem para as árvores. Como o *dataset* utilizado se mostrou amplamente não linear, os resultados obtidos corroboram esse raciocínio, que pode ser estendido ao presente trabalho.

Em seguida, tratando de *random forest* e SVMs, artigos recentes encontraram resultados análogos. Han et al. aplicou ambos algoritmos para estimação através de regressão do índice de área foliar no dossel de macieiras, verificando melhor resultado com *random forest* (HAN et al., 2016). Por sua vez, Rodriguez-Galiano, V., et al. comparou a implementação de diversos modelos de aprendizagem de máquina na identificação de regiões de depósitos de outro no distrito minerador de Rodalquilar, na região sudeste da Espanha, observando resultados superiores para *random forest* em comparação às SVM (RODRIGUEZ-GALIANO et al., 2015).

4.4 Análise de dados - Assimetria

Na Seção 3.5.4, considerando a definição de Sistema inteligente de apoio à tomada de decisão, formulou-se hipótese acerca da geração de um modelo flexível. Isto é, que habilitasse aos gestores ajuste para predições pessimistas ou otimistas. Em seguida, definiu-se a base teórica para essa formulação com base na manipulação da função de perda ϵ insensitiva das máquinas de vetores de suporte, a qual se denominou "assimetria".

4.4.1 *Dataset* final

Após implementada, a hipótese foi verificada, em primeiro momento, no conjunto de dados definitivo desse trabalho, ou seja, *dataset* final.

4.4.1.1 Assimetria positiva

Primeiramente, implementou-se a assimetria positiva conforme detalhado na Seção 3.5.4, mais especificamente na Figura 15. Ou seja, espera-se que o modelo seja pessimista, prevendo valores maiores que os reais. A ferramenta NIOTS foi então executada conforme configuração da Tabela 10 tendo como entrada o *dataset* final, que por sua vez teve dimensões especificadas na Tabela 9. Após o processo de treinamento, o modelo **SVM1**, tomado como referência nas análises anteriores, foi retornado. Com base nele, foram realizadas predições a partir do conjunto de testes e os resultados obtidos sintetizados na Tabela 25.

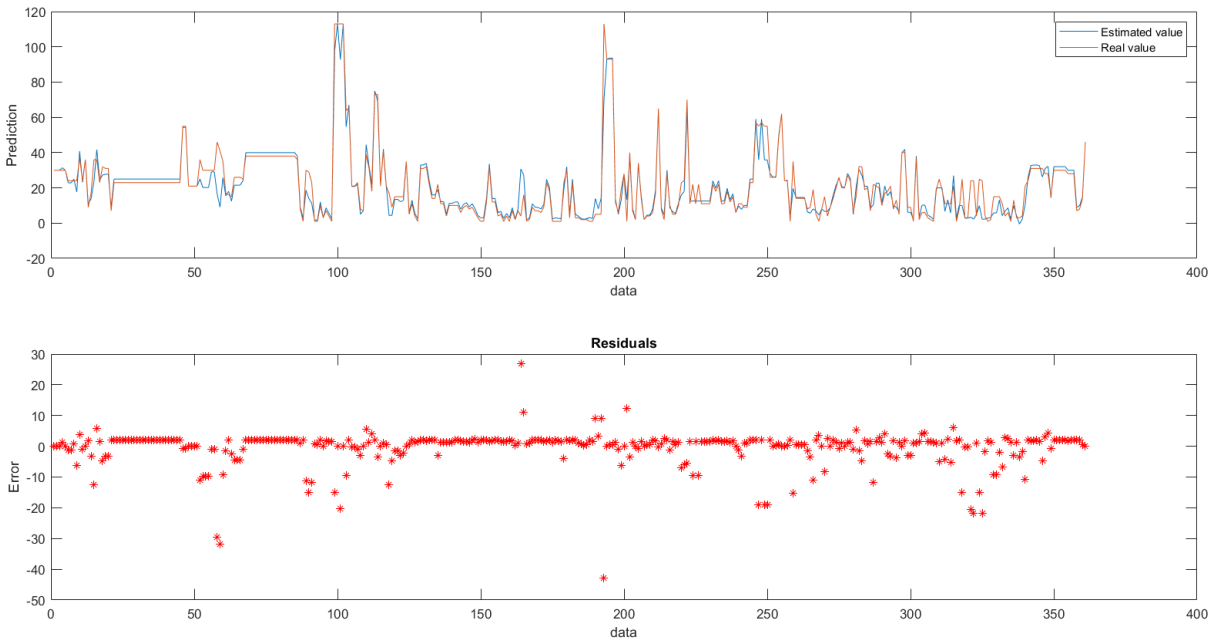


Figura 34 – Curvas de teste do *dataset* final - SVM com assimetria positiva

Teste - Dataset final	
SVM - Assimetria positiva	
MSE	34
MAPE	34

Tabela 25 – Resultados da SVM com assimetria positiva - *Dataset* final

Comparando as Tabelas 25 e 21, verifica-se que a implementação de assimetria positiva aumentou o MAPE de teste. De forma geral, variação no valor de erro era esperada uma vez que o modelo deve, propositalmente, se deslocar.

Novamente, um dos produtos desse teste foi a representação gráfica contida na Figura 34. Observa-se que a curva estimada (azul) tende a um deslocamento superior com relação à curva real (laranja). No entanto, visualmente, não se constitui um resultado expressivo.

Além disso, gerou-se também histograma de erro conforme apresentado na Seção 3.4 e exposto na Figura 35. Em suma, o eixo das abscissas define intervalos de erro em dias enquanto as ordenadas indicam o total de amostras - denomindado população - com erro de predição correspondente a cada intervalo. Esse resultado, ao contrário das curvas, é um indicativo de assimetria positiva pois observa-se que a maior parte das barras do histograma se distribuem à direita de zero, ou seja, indicam erros positivos. Para o caso analisado, destacou-se a maior delas, que indica 173 amostras no intervalo [1,2], o que corresponde a 48% do total de 361 predições.

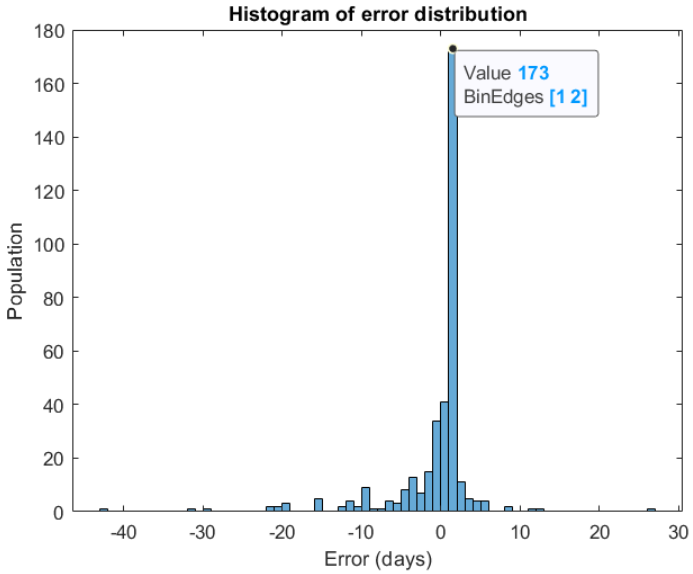


Figura 35 – Histograma de erro do *dataset* final - SVM com assimetria positiva

4.4.1.2 Assimetria negativa

De maneira análoga, o processo foi repetido para assimetria negativa, implementada conforme detalhado na Seção 3.5.4, mais detalhadamente na Figura 14. Ou seja, espera-se que o modelo seja otimista, prevendo valores menores que os reais. A ferramenta NIOTS foi então executada conforme configuração da Tabela 10 tendo como entrada o *dataset* final, que por sua vez teve dimensões especificadas na Tabela 9. Novamente, a partir do modelo **SVM1**, realizaram-se predições do conjunto de testes e os resultados obtidos foram sintetizados na Tabela 26.

Teste - Dataset final	
SVM - Assimetria negativa	
MSE	38
MAPE	28

Tabela 26 – Resultados da SVM com assimetria negativa - *Dataset* final

Comparando as Tabelas 26 e 21, constata-se que a implementação de assimetria negativa diminuiu 1% o MAPE de teste. De forma geral, variação no valor de erro era esperada uma vez que o modelo deve, propositalmente, se deslocar.

Mais uma vez, um dos produtos desse teste foi a representação gráfica contida na Figura 36. Observa-se que a curva estimada (azul) se encontra ajustada à curva real (laranja) e pouco deslocada dela. No entanto, quando comparamos esse resultado ao anterior, da Figura 34, nota-se o deslocamento no sentido inferior em diversos pontos a exemplo das regiões: anteriores à amostra 100; em torno de 150; em torno de 300; em torno de 350.

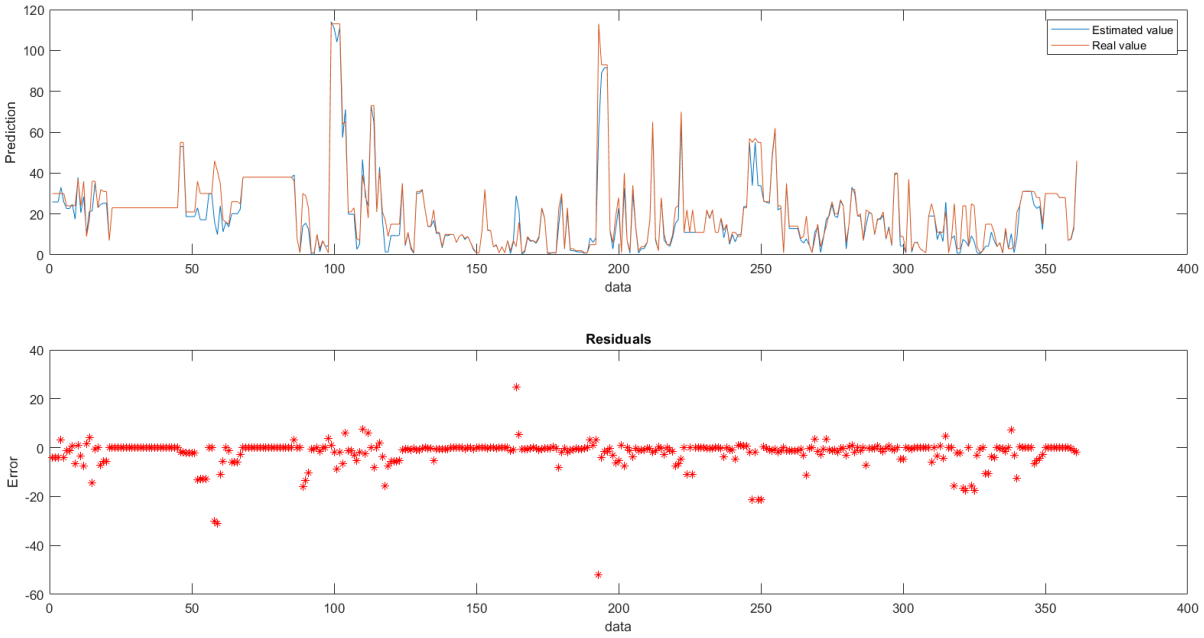


Figura 36 – Curvas de teste do *dataset* final - SVM com assimetria negativa

Por fim, também foi gerado o histograma de erro exposto na Figura 37. Esse resultado contaria novamente as curvas e indica assimetria negativa uma vez que a maior parte das barras do histograma se distribuem à esquerda de zero, ou seja, indicam erros negativos. Para o caso analisado, destacou-se a maior delas, que indica 148 amostras no intervalo [−1,0], o que corresponde a 41% do total de 361 predições.

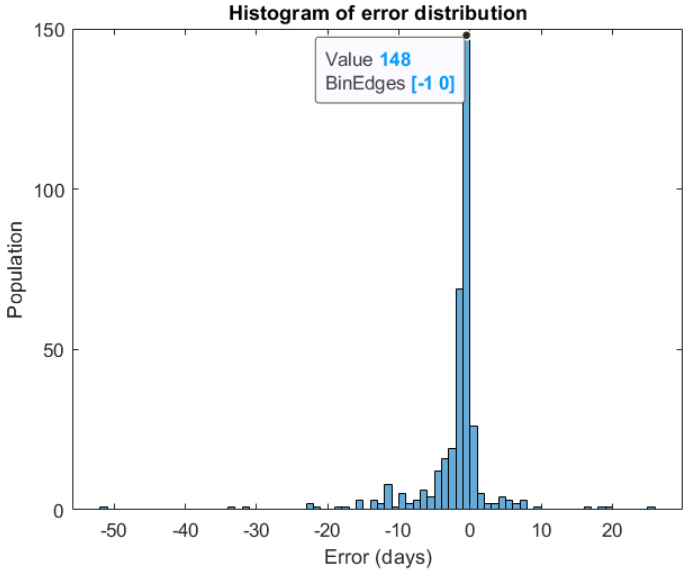


Figura 37 – Histograma de erro do *dataset* final - SVM com assimetria negativa

4.4.2 *Dataset benchmark* de solda

Os resultados obtidos com a aplicação de assimetria no *dataset* final foram sub-ótimos, pois apenas os histogramas permitem avaliar a implementação enquanto as curvas geradas

se mostraram pouco expressivas. No entanto, tais observações podem estar relacionadas à complexidade e não-linearidade dos dados utilizados, que dificultam a avaliação da proposta.

Anteriormente, essa possibilidade foi levantada na Seção 3.5.5 e, exatamente por isso, propôs-se aplicação do *dataset benchmark* de solda. De forma geral, esperava-se que utilizar um conjunto de dados mais simples favorecesse a visualização de resultados da implementação sem interferência de complexidade das entradas.

4.4.2.1 Implementação simétrica

Primeiramente, o treinamento foi realizado com SVM simétrica, ou seja, convencional, como parâmetro de comparação à posterior implementação assimétrica. O procedimento foi análogo ao da Seção 4.3.1, sendo a ferramenta NIOTS executada conforme configuração da Tabela 10 tendo como entrada o *dataset benchmark* de solda com dimensões especificadas na Tabela 14.

O modelo **SVM1**, tomado como referência nas análises anteriores, foi retornado após treinamento. A partir dele, realizaram-se predições do conjunto de testes e os resultados obtidos foram sintetizados na Tabela 27.

Teste - Dataset benchmark de solda	
SVM - Simétrica	
MSE	0,0004
MAPE	4

Tabela 27 – Resultados da SVM - *Dataset benchmark* de solda

Em seguida, o resultado foi representado graficamente na Figura 38. Observa-se ajuste expressivo entre a curva estimada (azul) e a curva real (laranja), o que corrobora o baixo MAPE obtido.

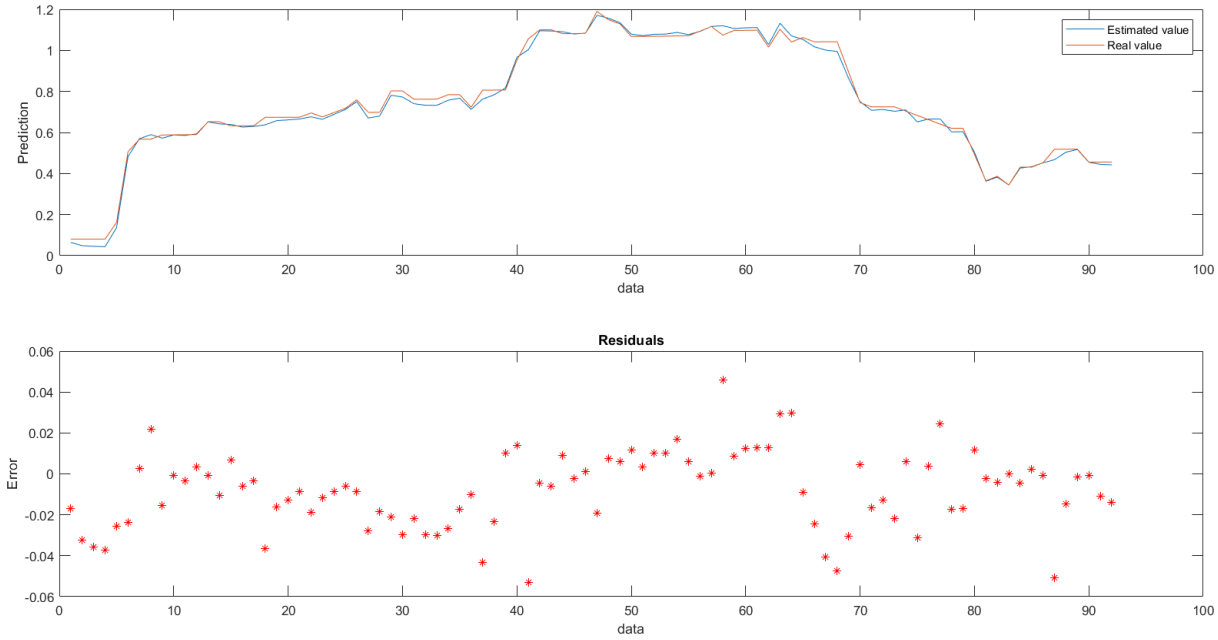


Figura 38 – Curvas de teste do *dataset benchmark* de solda - SVM simétrica

Da mesma forma, gerou-se o histograma de erro para esse caso de teste conforme a Figura 39. Com base nela, verificou-se que em geral o modelo gera tanto erros positivos quanto negativos, embora tenha tendência maior ao negativo.

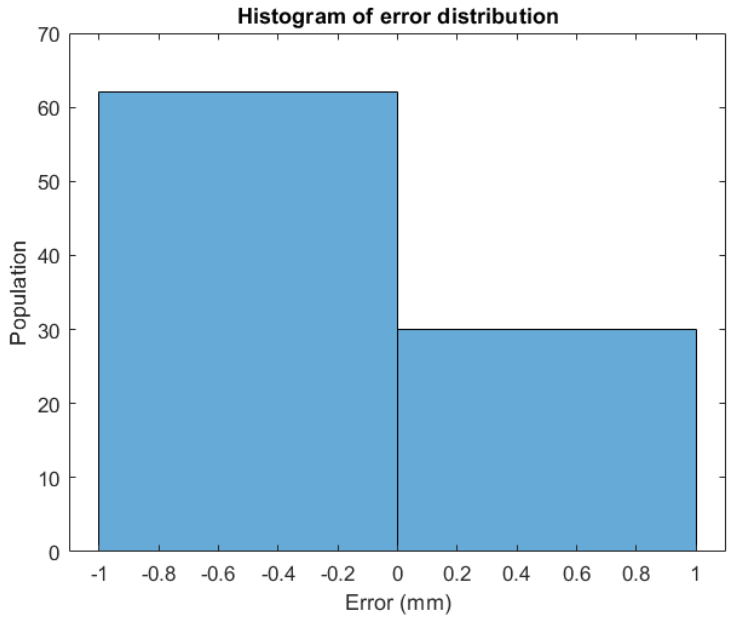


Figura 39 – Histograma de erro do *dataset benchmark* de solda - SVM simétrica

4.4.2.2 Assimetria positiva

Uma vez obtidos os resultados simétricos, prosseguiu-se à implementação da assimetria, iniciando pela positiva conforme detalhado na Seção 3.5.4, mais especificamente na Figura 15. Ou seja, espera-se que o modelo seja pessimista, prevendo valores maiores

que os reais. O processo de treinamento foi realizado com a ferramenta NIOTS conforme configuração da Tabela 10 tendo como entrada o *dataset benchmark* de solda com dimensões especificadas na Tabela 14.

Após obtenção do modelo SVM1, foram realizadas predições a partir do conjunto de testes e os resultados obtidos sintetizados na Tabela 28.

Teste - Dataset benchmark de solda	
SVM - Assimetria positiva	
MSE	0,0047
MAPE	14

Tabela 28 – Resultados da SVM com assimetria positiva - *Dataset benchmark* de solda

Com base nos testes, verificou-se que o MAPE aumentou 10% quando comparado ao valor da Tabela 27, ou seja, em comparação à SVM simétrica. Essa variação do erro pode ser justificada pela ocorrência do deslocamento proposto e, portanto, geraram-se os gráficos da Figura 40. Dessa vez, notou-se deslocamento expressivo da curva predita (azul) com relação à real (laranja), que corrobora o MAPE observado e indica que a implementação da assimetria ocorreu com sucesso.

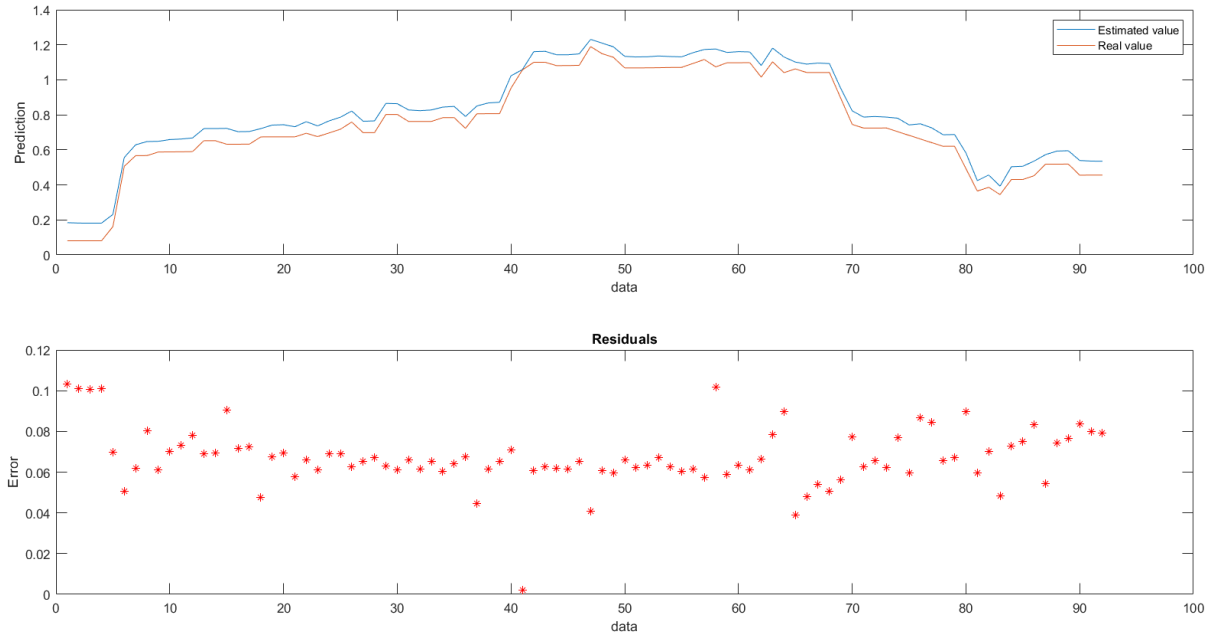


Figura 40 – Curvas de teste do *dataset benchmark* de solda - SVM assimétrica positiva

Em seguida, também foi gerado histograma de erros para esse caso de teste conforme a Figura 41. A distribuição observada reafirma a assimetria, uma vez que todos os erros do modelo se encontram no semiplano positivo, resultado completamente diferente do verificado na Figura 39.

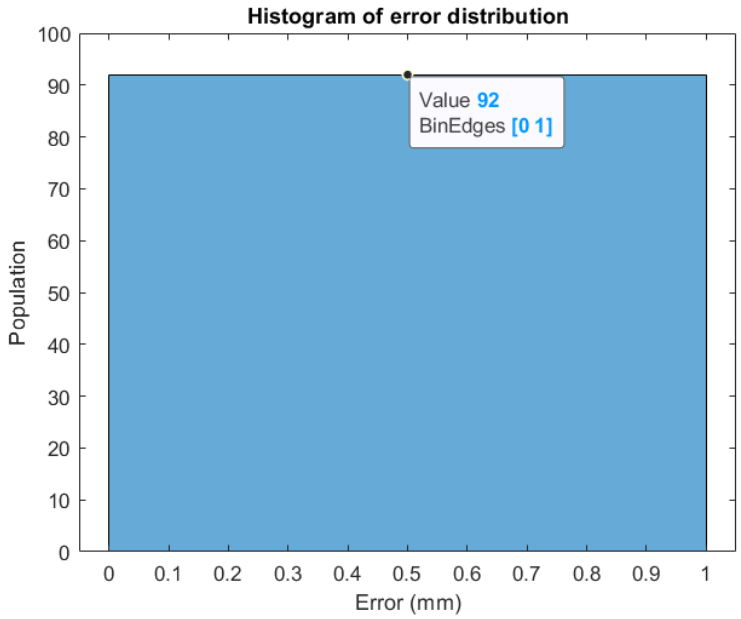


Figura 41 – Histograma de erro do *dataset benchmark* de solda - SVM assimétrica positiva

4.4.2.3 Assimetria negativa

De maneira análoga, o processo foi repetido para assimetria negativa, implementada conforme especificado na Seção 3.5.4, mais detalhadamente na Figura 14. Ou seja, espera-se que o modelo seja otimista, prevendo valores menores que os reais. A ferramenta NIOTS foi executada de acordo com configuração da Tabela 10 tendo o *dataset benchmark* de solda como entrada, com dimensões especificadas na Tabela 14.

A partir do modelo **SVM1** retornado, realizaram-se predições com o conjunto de testes e os resultados obtidos foram sintetizados na Tabela 29.

Teste - Dataset benchmark de solda	
SVM - Assimetria positiva	
MSE	0,0022
MAPE	8

Tabela 29 – Resultados da SVM com assimetria positiva - *Dataset benchmark* de solda

Constata-se que o MAPE aumentou 4% quando comparado ao valor da Tabela 27, ou seja, em comparação à SVM simétrica. Novamente, essa variação do erro pode ser justificada pela ocorrência do deslocamento esperado e, portanto, os gráficos da Figura 42 foram gerados. Ao contrário do caso anterior, o deslocamento da curva predita (azul) com relação à real (laranja) é no sentido negativo. Dessa forma, corrobora que haja alteração no valor de MAPE e, além disso, indica pleno funcionamento da implementação assimétrica.

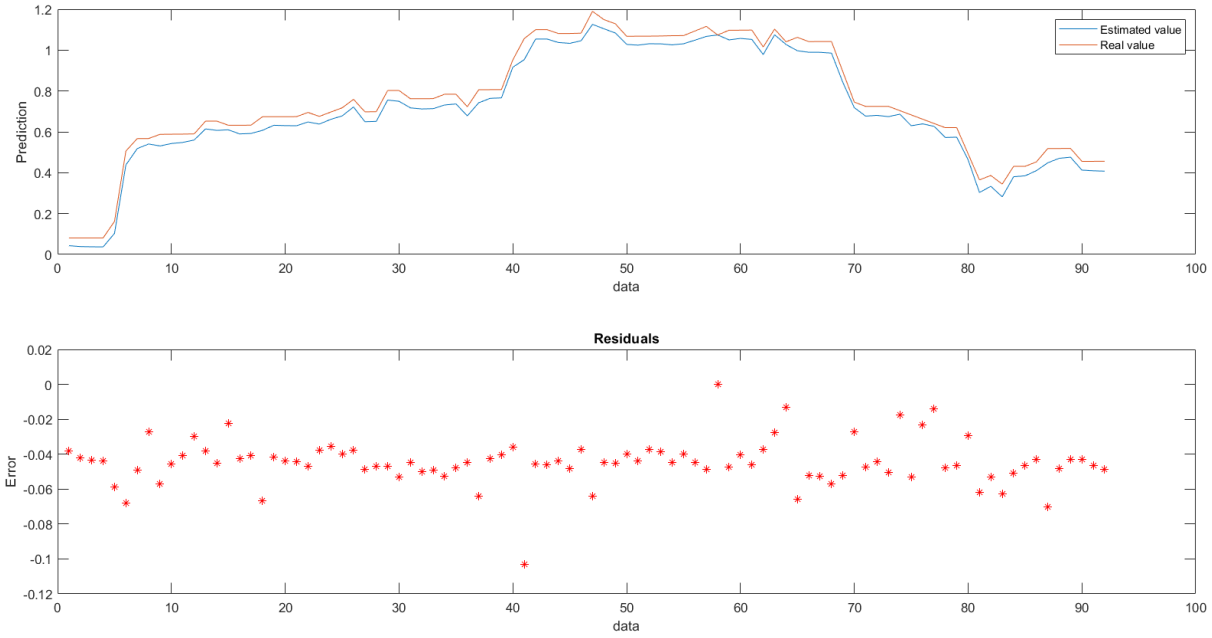


Figura 42 – Curvas de teste do *dataset benchmark* de solda - SVM assimétrica negativa

Da mesma forma, gerou-se histograma de erro para essa análise, que se encontra na Figura 43. As duas barras resultantes indicaram que a maioria das amostras apresentam erro negativo, visto que se encontram nesse semiplano,resultado que reforça o funcionamento da assimetria.

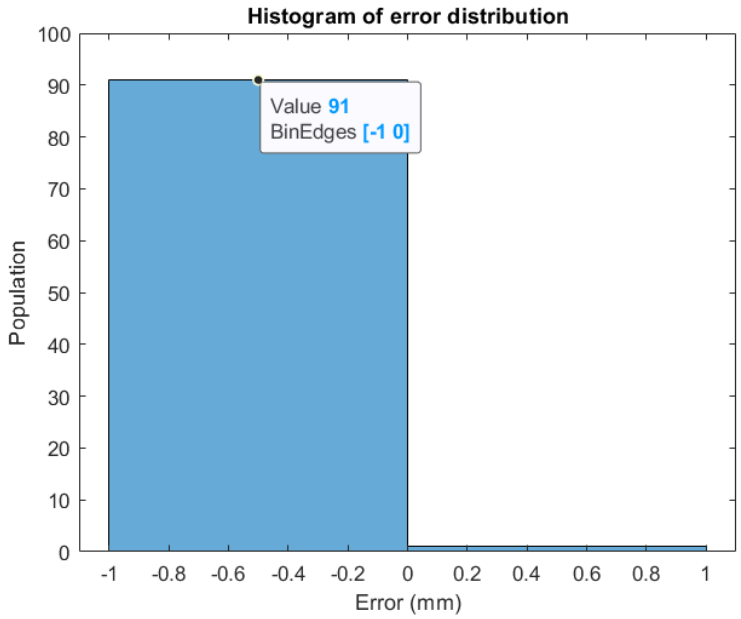


Figura 43 – Histograma de erro do *dataset benchmark* de solda - SVM assimétrica negativa

4.4.2.4 Análise comparativa

Por fim, após obter os resultados assimétricos, estes foram comparados através de representação gráfica na Figura 44. A curva laranja representa os valores previstos pelo modelo

simétrico exposto na Seção 4.4.2.1, ou seja, indica a previsão de um modelo convencional. Por outro lado, as curvas azuis tracejadas correspondem às previsões pessimistas e otimistas obtidas pelos modelos de assimetria positiva da seção 4.4.2.2 e assimetria negativa da seção 4.4.2.3, respectivamente.

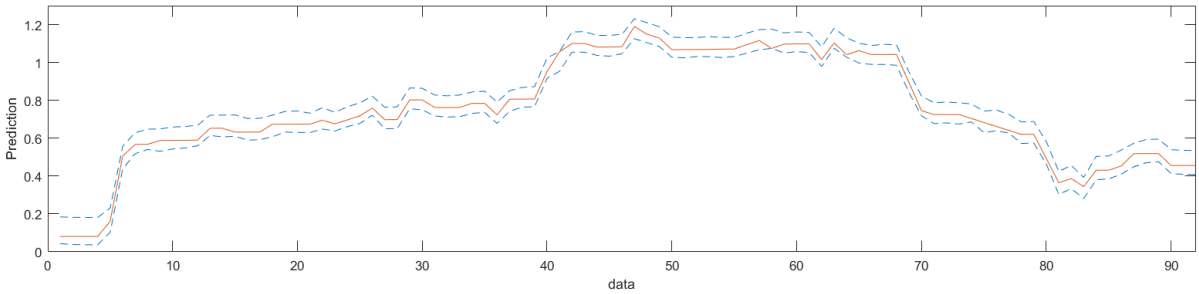


Figura 44 – Comparação *dataset benchmark* de solda - SVM assimétrica positiva e negativa

Dessa forma, é possível observar visualmente o deslocamento da curva estimada nos dois sentidos, delimitado pelos limites de cada assimetria. Portanto, valida-se a possibilidade de implementar modelo flexível com base em máquinas de vetors de suporte regressoras através da manipulação de sua função de perda ϵ insensitiva, como proposto na Seção 3.5.4.

5 Conclusão

O trabalho se propôs a aplicar *machine learning* na modelagem do *lead-time* de fabricação em empresa de produção mecatrônica na área médica e espacial, abordando o problema por meio de regressão utilizando *Support Vector Machines* (SVM).

De início, utilizando conjunto de dados disponibilizado por (ROSA, 2020), aplicou-se *one hot encoding* às variáveis, tornando-as compatíveis aos algoritmos utilizados. Em seguida, três propostas de divisão desses dados foram adotadas. A redução, primeiramente, utilizou um subconjunto do *dataset* original, que contava com muitas entradas. Reduzindo esse número, assegurou-se correto funcionamento da ferramenta e avaliação inicial dos dados. Em seguida, filtros foram aplicados a determinadas variáveis do problema selecionando os rótulos mais populosos e avaliando seu impacto na modelagem. Dessa forma, foi possível verificar que a variável **Conjunto** apresentava maior relevância e, com base nos resultados obtidos, definiu-se o terceiro e último conjunto de dados do trabalho.

Na sequência, utilizou-se a ferramenta NIOTS (LLANOS; SILVA SANTOS; DOS SANTOS, 2021) que implementa meta-heurísticas para ajuste dos hiperparâmetros de SVMs a partir de um problema de otimização multi-objetivo. A cada execução, 40 modelos-candidatos foram gerados em cada caso de teste proposto, apresentando variações em seus erros médios e na quantidade de vetores de suporte. Dentre esses 40, três modelos foram selecionados para realização de predições. Posteriormente, análises indicaram que o modelo **SVM1**, com menor MSE e maior quantidade de vetores de suporte em treinamento, apresentava os melhores resultados em teste.

Em fase de análise, duas métricas distintas foram propostas, MSE e MAPE. O primeiro deles retorna o erro médio quadrático entre as curvas e, portanto, é sensível à sua magnitude. O segundo, por outro lado, trata do erro médio percentual, estabelecendo análise relativa entre dados reais e preditos e tendo se mostrado eficaz no trabalho realizado. Além disso, para possibilitar visualização gráfica dos resultados, geraram-se imagens que, para cada caso de teste, apresentaram a sobreposição das curvas real e predita e um histograma da distribuição de erro.

Com base no exposto, foi estabelecida comparação entre os resultados obtidos pelas máquinas de vetores de suporte e outros dois algoritmos. Primeiramente, as SVM apresentaram MAPE de 29%, que se mostrou 6% melhor quando comparado à aplicação de **regressão linear múltipla** sobre o mesmo conjunto de dados. Por outro lado, ambas foram superadas pelas ***random forest*** que retornaram erro de 17%, provando ser o modelo mais eficiente na análise realizada. Tais constatações corroboram a teoria e o resultado de outros trabalhos recentes.

Finalmente, considerando a definição de **Sistema inteligente de apoio à tomada de decisão** (LAUDON, K. C.; LAUDON, J. P., 2013), propôs-se a definição de um modelo flexível. Isto é, que permita profissionais do meio corporativo ajustarem predições para um viés otimista ou pessimista. Essa abordagem foi possível apenas no algoritmo SVM, a partir da manipulação de sua função de perda ϵ insensitiva, e com base na análise das curvas e histogramas obtidos, obteve-se sucesso em sua implementação.

Dessa forma, é notável que a proposta tem aplicações em cenários de produção industrial e pode evoluir ao ponto de constituir ferramenta para tomada de decisões em equipes de Engenharia. Para viabilizar isso, no entanto, otimizações ainda são necessárias em trabalhos futuros.

Primeiramente, sugere-se uma nova etapa de pré processamento com perspectiva distinta da usada em (ROSA, 2020), que promova comparação entre os dois métodos e permita otimizar a extração de informação dos dados. Além disso, seria de extrema relevância ter acesso a novos conjuntos de dados da mesma empresa e verificar se os resultados obtidos se estendem a eles e podem, portanto, ser generalizados. Ainda nesse aspecto, sugere-se a possibilidade de testar novas estratégias de codificação dos dados que sejam diferentes da utilizada (*one-hot*). Destacam-se a codificação binária e de *hamming*.

Em seguida, destaca-se a seleção de novas métricas de avaliação. Embora MSE e MAPE tenham apresentado bons resultados, é possível que outras regras enriqueçam ainda mais a análise, permitindo aumentar os acertos ou até mesmo a confiabilidade do modelo. Assim, cabe a realização de estudo que compare novas métricas a partir dos resultados obtidos nesse trabalho.

Por fim, sugere-se também continuação da investigação dos modelos flexíveis que, no contexto das SVM, ainda podem ser refinados com a definição de método para um segundo ajuste que quantifique essa flexibilidade. Além do mais, é possível estudar a aplicação dessa hipótese a outro algoritmos.

Referências

ALVAREZ BESTARD, G.; SAMPAIO, R.; VARGAS, J.; ALFARO, S. Sensor Fusion to Estimate the Depth and Width of the Weld Bead in Real Time in GMAW Processes. **Sensors (Basel, Switzerland)**, v. 18, mar. 2018. DOI: 10.3390/s18040962. Citado na p. 56.

BARBALHO, S.; TORRES, L. Melhoria de indicadores de desempenho em desenvolvimento de produtos por meio do enfoque no aumento da capabilidade dos processos: o caso da fabricação de protótipos de novos produtos. In. Citado nas pp. 16, 17.

BASHIR, H. A. Modeling of development time for hydroelectric generators using factor and multiple regression analyses. **International Journal of Project Management**, v. 26, n. 4, 2008. ISSN 0263-7863. DOI: https://doi.org/10.1016/j.ijproman.2007.08.006. Citado na p. 15.

BILLINGS, S. A.; JAMALUDDIN, H. B.; CHEN, S. Properties of neural networks with applications to modelling non-linear dynamical systems. **International Journal of Control**, Taylor & Francis, v. 55, n. 1, p. 193–224, 1992. DOI: 10.1080/00207179208934232. Citado na p. 58.

BILLINGS, S. A.; VOON, W. S. F. Correlation based model validity tests for non-linear models. **International Journal of Control**, Taylor & Francis, v. 44, n. 1, p. 235–244, 1986. DOI: 10.1080/00207178608933593. Citado na p. 58.

BILLINGS, S. Nonlinear System Identification: NARMAX Methods in the Time, Frequency, and Spatio-Temporal Domains. **Nonlinear System Identification: NARMAX Methods in the Time, Frequency, and Spatio-Temporal Domains**, ago. 2013. DOI: 10.1002/9781118535561. Citado na p. 59.

BOTCHKAREV, A. Evaluating Performance of Regression Machine Learning Models Using Multiple Error Metrics in Azure Machine Learning Studio. **SSRN Electronic Journal**, jan. 2018. DOI: 10.2139/ssrn.3177507. Citado nas pp. 39, 47.

BURGES, C. J. A Tutorial on Support Vector Machines for Pattern Recognition. English. **Data Mining and Knowledge Discovery**, Kluwer Academic Publishers, v. 2, n. 2, p. 121–167, 1998. ISSN 1384-5810. DOI: 10.1023/A:1009715923555. Disponível em: <http://dx.doi.org/10.1023/A%5C%3A1009715923555>. Citado na p. 29.

CHANG, C.-C.; LIN, C.-J. LIBSVM: A library for support vector machines. **ACM transactions on intelligent systems and technology (TIST)**, Acm New York, NY, USA, v. 2, n. 3, p. 1–27, 2011. Citado na p. 56.

CORTES, C.; VAPNIK, V. Support Vector Networks. **Machine Learning**, v. 20, p. 273–297, 1995. Citado nas pp. 24, 25, 27.

COS JUEZ, F. de; GARCIA NIETO, P. J.; MARTÍNEZ, J.; CASTRO, J. Analysis of lead times of metallic components in the aerospace industry through a supported vector machine model. **Mathematical and Computer Modelling**, v. 52, p. 1177–1184, out. 2010. DOI: 10.1016/j.mcm.2010.03.017. Citado na p. 16.

GRIVA, I.; NASH, S. G.; SOFER, A. **Linear and Nonlinear Optimization (2. ed.).** SIAM, 2008. P. i–xxii, 1–742. ISBN 978-0-89871-661-0. Citado na p. 25.

HA QUANG, M.; NIYOGI, P.; YAO, Y. Mercer's Theorem, Feature Maps, and Smoothing. In: p. 154–168. ISBN 978-3-540-35294-5. DOI: 10.1007/11776420_14. Citado na p. 29.

HAN, Z.-y.; ZHU, X.-c.; FANG, X.-y.; WANG, Z.-y.; WANG, L.; ZHAO, G.-X.; JIANG, Y.-m. [Hyperspectral Estimation of Apple Tree Canopy LAI Based on SVM and RF Regression]. **Guang pu xue yu guang pu fen xi = Guang pu**, v. 36, n. 3, p. 800–805, mar. 2016. ISSN 1000-0593. Disponível em: <http://europepmc.org/abstract/MED/27400527>. Citado na p. 76.

JAMES, G.; WITTEN, D.; HASTIE, T.; TIBSHIRANI, R. **An Introduction to Statistical Learning with Applications in R**. 2. ed.: Apringer, 2013. Citado nas pp. 24–27, 33–39, 75.

JUN, H.-B.; PARK, J.-Y.; SUH, H.-W. Lead Time Estimation Method for Complex Product Development Process. **Concurrent Engineering: R&A**, v. 14, p. 313–328, dez. 2006. DOI: 10.1177/1063293X06073302. Citado na p. 15.

KHAN, M.; WU, X.; XU, X.; DOU, W. Big Data Challenges and Opportunities in the Hype of Industry 4.0. In. DOI: 10.1109/ICC.2017.7996801. Citado na p. 15.

LAUDON, K. C.; LAUDON, J. P. **Management Information Systems: Managing the Digital Firm**. 13. ed. Boston: Pearson, 2013. ISBN 978-0-273-78997-0. Citado nas pp. 54, 87.

LI, B.; HOU, B.-c.; YU, W.-t.; LU, X.-b.; YANG, C.-w. Applications of artificial intelligence in intelligent manufacturing: a review. **Frontiers of Information Technology & Electronic Engineering**, v. 18, p. 86–96, jan. 2017. DOI: 10.1631/FITEE.1601885. Citado na p. 15.

LLANOS, C. H.; SILVA SANTOS, C. E. da; DOS SANTOS, L. C. Nature Inspired Optimization Tools for SVMs - NIOTS. **MethodsX**, p. 101574, 2021. ISSN 2215-0161. DOI: https://doi.org/10.1016/j.mex.2021.101574. Disponível em: <https://www.sciencedirect.com/science/article/pii/S2215016121003642>. Citado nas pp. 18, 19, 31, 32, 47, 49, 50, 86.

LORENA, A. C.; CARVALHO, A. C. P. L. F. de. Uma Introdução às Support Vector Machines. **Revista de Informática Teórica e Aplicada**, v. 14, n. 2, p. 43–67, out. 2007. Citado na p. 23.

MORI, J.; MAHALEC, V. Planning and scheduling of steel plates production. Part I: Estimation of production times via hybrid Bayesian networks for large domain of discrete variables. **Computers & Chemical Engineering**, v. 79, mar. 2015. DOI: 10.1016/j.compchemeng.2015.02.005. Citado na p. 16.

MOURTZIS, D.; DOUKAS, M.; FRAGOU, K.; EFTHYMIOU, K.; MATZOROU, V. Knowledge-based Estimation of Manufacturing Lead Time for Complex Engineered-to-order Products. **Procedia CIRP**, v. 17, p. 499–504, 2014. Variety Management in Manufacturing. ISSN 2212-8271. DOI: https://doi.org/10.1016/j.procir.2014.01.087. Citado nas pp. 15, 16.

NIKOLIC, B.; IGNJATIC, J.; SUZIC, N.; STEVANOV B & RIKALOVIC, A. Predictive Manufacturing Systems in Industry 4.0: Trends, Benefits and Challenges. **28th DAAAM International Symposium**, 2017. DOI: DOI:10.2507/28th.daaam.proceedings.112. Citado na p. 15.

PEDREGOSA, F.; VAROQUAUX, G.; GRAMFORT, A.; MICHEL, V.; THIRION, B.; GRISEL, O.; BLONDEL, M.; PRETTENHOFER, P.; WEISS, R.; DUBOURG, V.; VANDERPLAS, J.; PASSOS, A.; COURNAPEAU, D.; BRUCHER, M.; PERROT, M.; DUCHESNAY, E. Scikit-learn: Machine Learning in Python. **Journal of Machine Learning Research**, v. 12, p. 2825–2830, 2011. Citado na p. 51.

RODRIGUEZ-GALIANO, V.; SANCHEZ-CASTILLO, M.; CHICA-OLMO, M.; CHICA-RIVAS, M. Machine learning predictive models for mineral prospectivity: An evaluation of neural networks, random forest, regression trees and support vector machines. **Ore Geology Reviews**, v. 71, p. 804–818, 2015. ISSN 0169-1368. DOI: https://doi.org/10.1016/j.oregeorev.2015.01.001. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0169136815000037>. Citado na p. 76.

ROSA, R. C. Aprendizagem de máquina aplicada à predição do tempo de fabricação de novos produtos : um estudo exploratório com foco no tipo de material utilizado em empresa de produção mecatrônica da área médica e espacial. In. Citado nas pp. 17, 18, 42, 86, 87.

SANTOS, C. E. d. S. Seleção de parâmetros de máquinas de vetores de suporte usando otimização multiobjetivo baseada em meta-heurísticas. In. Citado nas pp. 18, 22–32, 47, 56.

SILVA, F.; FERNANDES, F. Proposal of a production control system for shoemakers operating in a resources-to-order or make-to-order product market environment. **Gestão & Produção**, v. 15, p. 523–538, dez. 2008. Citado na p. 15.

SILVA, I.; SPATTI, D.; FLAUZINO, R.; BARTOCCI LIBONI, L.; REIS ALVES, S. **Artificial Neural Networks**. Jan. 2017. ISBN 978-3-319-43161-1. DOI: 10.1007/978-3-319-43162-8. Citado nas pp. 21, 22, 44.

SMOLA, A.; SCHÖLKOPF, B. A tutorial on support vector regression. **Statistics and Computing**, v. 14, p. 199–222, ago. 2004. DOI: 10.1023/B%3ASTCO.0000035301.49549.88. Citado na p. 31.

SMOLA, A.; BURGES, C.; DRUCKER, H.; GOLOWICH, S.; HEMMEN, L.; MÜLLER, K.-R.; SCHOLKOPF, B.; VAPNIK, V. Regression Estimation with Support Vector Learning Machines, nov. 1996. Citado na p. 31.

SOKOLOVA, M.; LAPALME, G. A systematic analysis of performance measures for classification tasks. **Information Processing & Management**, v. 45, p. 427–437, jul. 2009. DOI: 10.1016/j.ipm.2009.03.002. Citado na p. 39.

SPÜLER, M.; SARASOLA SANZ, A.; BIRBAUMER, N.; ROSENSTIEL, W.; RAMOS-MURGUIALDAY, A. Comparing Metrics to Evaluate Performance of Regression Methods for Decoding of Neural Signals. In: v. 2015. DOI: 10.1109/EMBC.2015.7318553. Citado na p. 47.

SUSTO, G. A.; SCHIRRU, A.; PAMPURI, S.; MCLOONE, S.; BEGHI, A. Machine Learning for Predictive Maintenance: A Multiple Classifier Approach. **Industrial Informatics, IEEE Transactions on**, v. 11, p. 812–820, jun. 2015. DOI: 10.1109/TII.2014.2349359. Citado na p. 15.

TEAM, T. pandas development. **pandas-dev/pandas: Pandas**. Zenodo, fev. 2020a. DOI: 10.5281/zenodo.3509134. Disponível em: <https://doi.org/10.5281/zenodo.3509134>. Citado na p. 43.

TEAM, T. pandas development. **pandas.get_dummies doc**. Fev. 2020b. https://pandas.pydata.org/docs/reference/api/pandas.get_dummies.html. Citado na p. 43.

THEODORIDIS, S.; KOUTROUMBAS, K. **Pattern Recognition, Fourth Edition**. Academic Press, 2009. Hardcover. ISBN 9781597492720. Citado na p. 28.

ZHONG, R. Y.; XU, X.; KLOTZ, E.; NEWMAN, S. T. Intelligent Manufacturing in the Context of Industry 4.0: A Review. **Engineering**, v. 3, n. 5, p. 616–630, 2017. ISSN 2095-8099. DOI: https://doi.org/10.1016/J.ENG.2017.05.015. Citado na p. 15.