UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO TECNOLÓGICO
DEPARTAMENTO DE ENGENHARIA DE PRODUÇÃO E SISTEMAS
CURSO DE ENGENHARIA DE PRODUÇÃO CIVIL

Lucca Magri Zaghi

Modelo de previsão de preços de imóveis na cidade de Florianópolis/SC a partir de técnicas de Machine Learning.

Florianópolis
2023

Lucca Magri Zaghi

Modelo de previsão de preços de imóveis na cidade de Florianópolis/SC a partir de técnicas de Machine Learning.

Trabalho de Conclusão do Curso de Graduação em Engenharia de Produção Civil do Centro Tecnológico da Universidade Federal de Santa Catarina como requisito para a obtenção do título de Bacharel em Engenharia Civil com Habilitação em Engenharia de Produção
Orientador: Prof. Dr. Mauricio Uriona Maldonado

Florianópolis
2023

Zaghi, Lucca Magri
Modelo de previsão de preços de imóveis na cidade de Florianópolis/SC a partir de técnicas de Machine Learning. / Lucca Magri Zaghi ; orientador, Mauricio Uriona Maldonado, 2023.
84 p.

Trabalho de Conclusão de Curso (graduação) - Universidade Federal de Santa Catarina, Centro Tecnológico, Graduação em Engenharia de Produção Civil, Florianópolis, 2023.

Inclui referências.

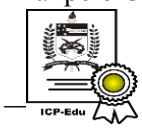
1. Engenharia de Produção Civil. 2. XGBoost. 3. Random Forest. 4. Regressão Lasso. 5. Web scraping. I. Maldonado, Mauricio Uriona . II. Universidade Federal de Santa Catarina. Graduação em Engenharia de Produção Civil. III. Título.

Lucca Magri Zaghi

Modelo de previsão de preços de imóveis na cidade de Florianópolis/SC a partir de técnicas de Machine Learning.

Florianópolis, 26 de Junho de 2023.

Este Trabalho Conclusão de Curso foi julgado adequado para obtenção do Título de Bacharel em Engenharia Civil com Habilitação em Engenharia de Produção e aprovado em sua forma final pelo Curso Engenharia de Produção Civil



Prof. Mauricio Uriona Maldonado, Dr.
Orientador
Universidade Federal de Santa Catarina

_____

Prof. Lynceo Falavigna Braghirolli, Dr.
Avaliador
Universidade Federal de Santa Catarina

_____

Cosme Polese
Avaliador
Universidade Federal de Santa Catarina

AGRADECIMENTOS

Primeiramente, gostaria de expressar minha profunda gratidão aos meus amados pais por todo o inestimável apoio que me concederam não apenas durante o meu ciclo universitário, mas ao longo de toda a minha vida. A imensa dedicação e amor incondicional foram pilares essenciais para o meu crescimento como profissional e como pessoa.

Quero estender meus agradecimentos especiais aos meus colegas de universidade, em particular à Yasmin Izzo, André Pegoraro e Bruno Patrício, pela amizade sincera e companheirismo extraordinário que sempre tivemos. Nossa jornada juntos não apenas enriqueceu minha experiência acadêmica, mas também forjou laços duradouros que carregarei comigo para sempre.

Além disso, gostaria de expressar uma profunda gratidão aos meus estimados professores, cuja sabedoria e orientação foram cruciais para minha jornada de formação como engenheiro. Seus comprometimentos com o ensino e seu compartilhamento de conhecimento foram verdadeiramente fundamentais para o meu crescimento acadêmico.

RESUMO

Este trabalho tem como objetivo propor um modelo de previsão de preços de imóveis na cidade de Florianópolis/SC. Para isto, foram extraídos dados de um portal imobiliário através da aplicação de extração de dados *web scraping*. Após a limpeza, organização, tratamento e classificação dos dados, foram utilizados três modelos preditivos de *Machine Learning* sendo eles: *Random Forest*, Regressão Lasso e *XGBoost*. Como parâmetros do modelo foram utilizados os seguintes atributos: número de quartos, banheiros, vagas, área total do imóvel (m²), localização (bairro), tipologia do imóvel e a presença ou não de condomínio. Para tanto, foram realizados procedimentos de validação cruzada e otimização de hiperparâmetros, objetivando a melhoria da precisão e redução de erros dos modelos. O melhor modelo foi o *Random Forest* com um RMSE de 5.16 e+05 e R2 de 0,818. Este modelo pode servir para corretores, investidores e demais usuários que desejam realizar a precificação de imóveis em Florianópolis/SC.

Palavras-chave: XGBoost; Random Forest; Regressão Lasso; Web scraping; Setor Imobiliário.

ABSTRACT

The aim of this study is to propose a real estate price prediction model in the city of Florianopolis, SC. For this purpose, data was extracted from a real estate portal using web scraping techniques. After data cleaning, organization, preprocessing, and classification, three predictive machine learning models were employed: Random Forest, Lasso Regression, and XGBoost. The model parameters used were the following attributes: number of bedrooms, bathrooms, parking spaces, total area of the property (m2), location (neighborhood), property typology, and the presence or absence of a condominium. Cross-validation and hyperparameter optimization procedures were conducted to improve the accuracy and reduce errors of the models. The best-performing model was Random Forest, with an RMSE of 5.16e+05 and R2 of 0.818. This model can be useful for real estate agents, investors, and other users interested in property pricing in Florianopolis, SC.

Keywords: XGBoost; Random Forest; LASSO Regression; Web scraping; Real Estate.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

BCB	Banco Central Brasileiro
XGBoost	eXtreme Gradient Boosting
IBGE	Instituto Brasileiro de Geografia e Estatística
ML	Machine Learning
RMSE	Root Mean Squared Error
MAPE	Mean Absolute Percentage Error
RSR	Relative Strength Ratio
RF	Random Forest
FIPE	Fundação Instituto de Pesquisas Econômicas
CBIC	Câmara Brasileira da Indústria da Construção
SELIC	Sistema Especial de Liquidação e de Custódia
SFH	Sistema Financeiro de Habitação
MCMV	Minha Casa Minha Vida

SUMÁRIO

1. INTRODUÇÃO

A aplicação de modelos de *Machine Learning* para prever preços de imóveis tem se tornado uma abordagem cada vez mais relevante no mercado imobiliário em escala global (BALDOMINOS, 2018). Com aproximadamente 72 milhões de domicílios no Brasil (IBGE, 2020), é fundamental encontrar maneiras eficientes e precisas de precificar os imóveis, considerando as particularidades de cada região.

Ao utilizar modelos de *Machine Learning*, é possível explorar os padrões e relacionamentos existentes entre as características dos imóveis e seus respectivos preços (BALDOMINOS, 2018). Dessa forma, esses modelos podem ser treinados para realizar previsões precisas dos preços de imóveis com base em características específicas, auxiliando no processo de precificação e fornecendo informações valiosas para agentes do mercado imobiliário.

Apesar do incremento no acesso de informações e dados ter beneficiado o setor imobiliário, a precificação de imóveis no Brasil ainda é amplamente realizada por meio de pesquisas de mercado manuais, que consomem tempo e recursos significativos.

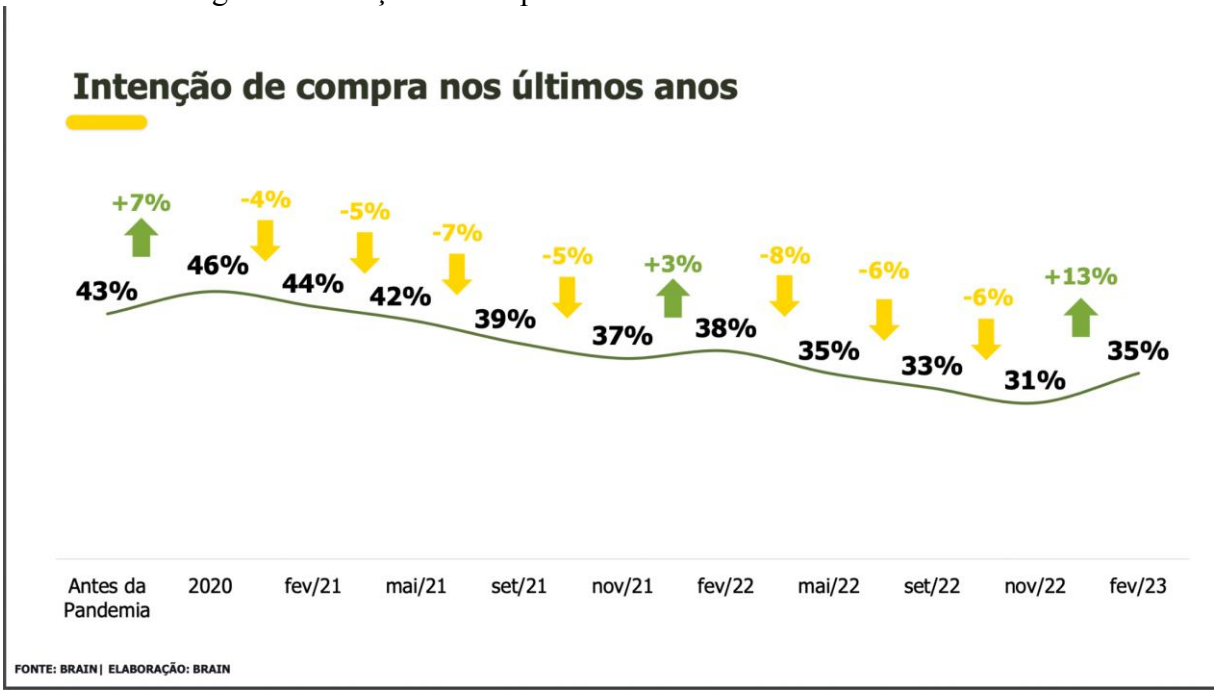
1.1 PROBLEMA DE PESQUISA

A constante mudança no mundo dos negócios sempre foi algo presente, mas a velocidade em que esse fluxo ocorre vem se tornando um dos maiores desafios para as mais diversas áreas se adaptarem (TASKAN et al., 2022). Dessa forma, torna-se cada vez mais essencial utilizar adereços, ferramentas e modelos que auxiliem a rápida tomada de decisão para organizações e indivíduos (TASKAN et al., 2022).

Fato esse que também ocorre dentro do mercado imobiliário que, pela definição de Matos e Bartkiw (2011), é composto por diversos atores como corretores, imobiliárias, empreiteiras, construtoras, proprietários e outros agregadores. Por se tratar de um setor com ampla participação econômica e que está diretamente relacionado ao cotidiano da civilização, o mesmo também acaba sofrendo com as constantes variações. Cabe ressaltar que esse é um dos mercados mais importantes para a economia nacional, sendo responsável por gerar diversos empregos e movimentar a economia, representando 10,2% do PIB em 2020 (BLUEPRINT, 2022). Ainda, segundo dados divulgados pela Câmara Brasileira da Indústria da Construção (CBIC, 2023), para o programa Minha Casa Minha Vida, pertencente ao Governo Brasileiro, no primeiro trimestre de 2023, 35% dos brasileiros apresentam intenção

de compra para imóveis, um crescimento de 13% frente ao trimestre anterior, o que ilustra a ampla gama de intenção de compra existente para o setor (Figura 1).

Figura 1. Intenção de compra de imóveis Minha Casa Minha Vida



Fonte: Indicadores Imobiliários Nacionais - CBIC (2023).

Dentro desse cenário de constante evolução e volatilidade, uma das maiores dores existentes nesse mercado diz respeito à precificação de imóveis, que vem sendo estudada pela academia desde a década de 1980 (DIN et al., 2001). De acordo com Santos (1997), o processo de decisão e formação de preços de serviços e bens leva em conta a coleta, ordenação e avaliação de diversos fatores complexos e interdependentes. Esses fatores podem ser trabalhados de maneira a formalizar um modelo de gestão que possa auxiliar na tomada de decisão e estabelecer um preço (SANTOS, 1997). No entanto, não existe uma formalização única que seja atualmente utilizada para a precificação de imóveis, podendo variar de acordo com os métodos adotados, exigindo uma grande quantidade de tempo de pesquisa e trabalho para alcançar um valor final eficaz. Essa dificuldade na formalização de um método que seja ao mesmo tempo eficiente e eficaz para precificar um imóvel pode ser explicada através de dois fatores principais: inacessibilidade de dados e particularidades de imóveis.

A inacessibilidade dos dados está diretamente relacionada à disponibilidade limitada e à descentralização das informações, o que restringe o acesso a esses dados por parte dos usuários que desejam utilizá-los. Alguns órgãos nacionais como o IBGE e o BCB divulgam

dados relacionados a esse mercado, mas ambos representam a macroeconomia, o que os tornam difíceis de serem utilizados para a precificação regional. Propondo-se a acompanhar as variações existentes no setor imobiliário em 2010, o Instituto FIPE se uniu à empresa ZAP Imóveis para concretizar o Índice FIPEZAP, que fornece informações sobre as variações de preços de maneira nacional e regional (FIPEZAP, 2023), sendo hoje um dos índices mais utilizados para acompanhamento de métricas referentes a valores de imóveis no Brasil. No entanto, embora o índice forneça dados com maior precisão regional, ele não oferece informações específicas sobre imóveis individuais. Isso leva muitos usuários que desejam precificar um imóvel a recorrer a portais imobiliários, onde podem encontrar dados mais detalhados e relevantes para suas análises. É nessa conjuntura que a particularidade de imóveis passa a ter valor também na dificuldade de precificar.

Existem diversas características que podem influenciar no preço de um imóvel no mercado. Variáveis como estado de conservação, localização e suas características (segurança e proximidade a locais de interesse público), facilidade de acesso, posição solar e infraestrutura de condomínio são alguns exemplos de fatores que influenciam não só no preço de um imóvel, mas também na decisão de compra de um potencial comprador (DIN et al., 2001). São características como essas que trazem complexidade para modelos únicos de precificação, sendo necessário compreender também a realidade em que cada imóvel se encontra e como ele se porta perante a região e os compradores existentes.

Assim, a dificuldade de acesso aos dados e as particularidades dos imóveis tornam a modelagem de preços um processo desafiador. Pode-se evidenciar que a existência de um único modelo de precificação talvez não seja suficiente para resolver essa questão, já que existem diversas variáveis capazes de influenciar nos valores imobiliários e cada qual com o seu respectivo peso diante de cenários específicos por região. No entanto, a construção de um modelo baseado em dados reais pode ser uma abordagem viável para auxiliar os usuários na tomada de decisão sobre o preço de um imóvel, economizando tempo e recursos para os usuários que necessitam de uma solução como essa.

É com esse contexto que os algoritmos de ML passam a ter espaço nos estudos direcionados à modelagem de preços de imóveis, sendo os mesmos capazes de criar modelos que representem um problema específico que deseja ser solucionado (NAQA et al., 2015), possuindo diversas aplicabilidades no setor de imóveis. Diversos estudos com o objetivo de precificar imóveis já foram feitos pela academia, como é o caso do estudo de Chou (2021) que utiliza diferentes combinações de algoritmos de Machine Learning para identificar os preços de imóveis na cidade de Taipei em Taiwan, obtendo resultados satisfatórios diante das

métricas de avaliação utilizados. Outro exemplo de aplicação pode ser encontrado no estudo dirigido por Borde (2017) na cidade de Mumbai, na Índia, conseguindo diversos indicadores positivos para os seus modelos preditivos de preço que se baseiam em um banco de dados públicos de mais de 10.000 imóveis listados à venda.

Diante dos estudos abordados e da necessidade de uma ferramenta que acelere o processo preditivo para valores imobiliários, as técnicas de Machine Learning têm conquistado cada vez mais relevância, transformando todo o processo de obtenção, tratamento e modelagem de dados na área de análise (WICKHAM, 2023).

1.2 OBJETIVOS

No presente tópico serão demonstrados os objetivos do trabalho desenvolvido.

1.2.1 Objetivo Geral

Propor um modelo precificador de imóveis na cidade de Florianópolis/SC com base em algoritmos de Machine Learning.

1.2.2 Objetivos Específicos

No contexto do objetivo geral do trabalho, os objetivos específicos são dados por:

- Extrair dados do setor imobiliário a partir de técnicas de *web scraping;*
- Tratar e organizar a base de dados;
- Treinar modelos de Machine Learning;
- Escolher o algoritmo com melhor desempenho.

1.3 JUSTIFICATIVA

O presente trabalho se justifica pela necessidade do mercado de ter um modelo de precificação de imóveis rápido e confiável. Atualmente, corretores e investidores imobiliários dedicam um grande número de horas em pesquisas manuais para determinar o valor de um imóvel, com base em dados públicos de portais imobiliários. Propor um modelo que possa beneficiar ambas as partes é uma das principais justificativas para a realização deste estudo.

Além disso, é importante considerar a perspectiva de outros atores do mercado imobiliário, que poderão utilizar as informações provenientes deste estudo para tomar decisões relacionadas ao mercado. Isso inclui a avaliação se os valores dos imóveis estão dentro das expectativas, bem como outras análises relevantes. Portanto, a importância do estudo vai além dos corretores e investidores diretos, abrangendo também outros profissionais e interessados no setor imobiliário.

1.4 LIMITAÇÕES

O presente trabalho limita-se a prever preços de imóveis para a cidade de Florianópolis/SC através de dados públicos extraídos de portais imobiliários em Abril/2023. A previsão foi realizada em uma base de 10.800 imóveis, tomando-se de base os fatores e dados existentes para cada um, limitando-se a informações disponibilizadas dentro da plataforma. Apesar da grande quantidade de dados, seria possível ainda realizar uma nova coleta de dados visando ampliar a gama de observações de imóveis para obter previsões mais ajustadas e que representem a maior parte do conjunto de imóveis de Florianópolis/SC.

É importante destacar que o modelo proposto baseia-se principalmente em valores de imóveis listados para venda, e não em valores transacionais reais. Dessa forma, os valores são ajustados de acordo com o que é comumente praticado no mercado, considerando as expectativas dos vendedores. De qualquer forma, acesso aos dados finais de venda são geralmente difíceis de serem coletados pois ficam espalhados em Cartórios Públicos sem nenhum tipo de centralização dessas informações.

1.5 ESTRUTURA DO TRABALHO

O primeiro capítulo do presente trabalho tem por objetivo introduzir brevemente a contextualização do objeto de estudo, bem como os problemas existentes. Também são abordados os objetivos e as justificativas para a execução do mesmo.

No segundo capítulo de fundamentação teórica, são abordados os principais conceitos necessários para o desenvolvimento e estudo do presente trabalho. Esse capítulo é construído com base em diversos referenciais teóricos do Mercado Imobiliário, *Machine Learning* e Análise de dados.

O terceiro capítulo aborda os procedimentos metodológicos adotados para o desenvolvimento do trabalho e cita as principais ferramentas utilizadas para alcançar os principais objetivos. Neste capítulo, as etapas de "Importação", "Organização de dados", "Transformação de dados", "Visualização de dados", "Modelagem" e "Análise de resultado" são descritas detalhadamente para embasar todos os resultados obtidos.

O quarto capítulo é destinado à apresentação de resultados obtidos através das metodologias e das ferramentas apresentadas nos capítulos anteriores. Aqui são apresentados resultados referentes à "Importação", "Análise Exploratória", "Validação do banco de dados" e "Comparação dos resultados".

No quinto e último capítulo, são descritas as conclusões referentes ao trabalho, bem como aos objetivos traçados inicialmente, além de fornecer sugestões de trabalhos futuros que possam complementar o estudo.

2 FUNDAMENTAÇÃO TEÓRICA

O capítulo de fundamentação teórica tem por objetivo explicar os conceitos principais utilizados durante o processo de pesquisa do presente trabalho. Logo em seguida serão abordados os tópicos referentes ao setor imobiliário, tratamento de dados e variáveis, modelos de Machine Learning e métricas de erro.

2.1 MERCADO IMOBILIÁRIO

O mercado imobiliário está diretamente relacionado ao cotidiano de grande parte dos indivíduos, já que o produto negociado nesse mercado é de extrema importância para a maioria das pessoas, seja o próprio imóvel em si ou os serviços relacionados, como locação, corretagem e administração de imóveis e condomínios (KREMER, 2008).

Segundo Matos (2012) os agentes envolvidos dentro deste setor são diversos e podem ser exemplificados como: corretoras de imóveis autônomas, imobiliárias, o profissional corretor, proprietário, empreiteiras de mão de obra, empresas da construção civil e empresas prestadoras de serviços, etc. Tais agentes possuem papel fundamental no que tange a comercialização de imóveis, sendo responsáveis por suprir a demanda existente de moradias no cenário nacional.

Além dos agentes privados mencionados anteriormente, o governo desempenha um papel importante no mercado imobiliário, atuando tanto de forma econômica quanto legislativa (KREMER, 2008). Nesse contexto, o governo é responsável por estabelecer políticas econômicas que direcionam e orientam diretamente as diretrizes do setor habitacional, especialmente no Brasil, onde uma parcela significativa da população não possui acesso adequado à moradia. Dessa forma, a intervenção governamental se torna necessária para garantir esse direito fundamental.

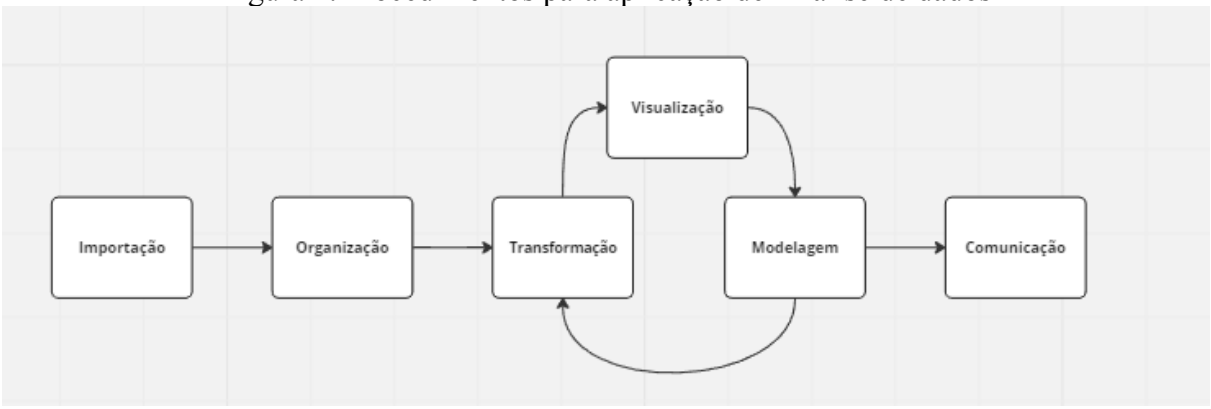
Dentre os agentes econômicos existentes no mercado imobiliário destaca-se também o Banco Central, responsável por definir a taxa básica de juros nacional, a SELIC, sendo responsável também pela fiscalização das atividades realizadas pelo SFH e a Caixa Econômica Federal (CEF), banco público que auxilia na execução de ações econômicas habitacionais, em especial crédito para programas de financiamento de obras, como o caso do programa MCMV (KREMER, 2008).

2.2 ANÁLISE DE DADOS

Nos últimos anos, o avanço tecnológico permitiu um maior volume de dados e informações, o que acabou desencadeando uma nova oportunidade para o desenvolvimento de métodos e estudos que pudessem traduzir esse enorme conjunto de dados em informações relevantes para tomadas de decisão (SOUZA et al., 2008). Dado o contexto, surge a análise de dados, que é a ciência responsável por transformar dados brutos em informações sistematizadas que possam fornecer conhecimento sobre um conjunto de dados específico (WICKHAM, 2023).

Na definição de Silvestre (2007) o pilar de sustentação da análise está diretamente relacionado ao conjunto de dados e os métodos aplicados no mesmo para gerar conclusões. Isso significa que cada banco de dados deve ser tratado de forma particular, a depender do objetivo final definido como resultado. Diante dessa observação, Wickham (2023) propõe um modelo não linear (Figura 2) que possa ser aplicado nas mais diversas fontes de dados existentes, sendo capaz de gerar resultados consistentes e confiáveis.

Figura 2. Procedimentos para aplicação de Análise de dados



Fonte: Autor (2023).

Tal processo de acordo com Wickham (2023) é dividido em seis macros etapas: importação, organização, transformação, visualização, modelagem e comunicação. Cada qual desempenha um papel diferente dentro da análise de dados e a definição de cada uma delas pode ser descrita a seguir:

I. Importação: Consiste em coletar dados de fontes diversas e importar os mesmos que estão armazenados em arquivos ou demais fontes para dentro de um ambiente capaz de interpretá-los;
II. Organização: Armazenar os dados de maneira sistemática, onde cada coluna será compreendida como uma variável assim como cada linha conterá uma variável das

observações dentro do estudo. Essa é uma das etapas mais importantes de uma análise, já que a mesma garante que os subsequentes passos sejam executados de maneira objetiva;

III. Transformação: Início do processo cíclico de entendimento dos dados que visa aperfeiçoar a análise. Nesta etapa, novas variáveis podem ser criadas com o objetivo de fornecer apoio às demais variáveis originais do conjunto de dados. Além disso, o conjunto pode ser filtrado ou reordenado de acordo com análise desejada;

IV. Visualização: Demonstração das variáveis através de ilustrações gráficas e tabelas informativas que condensam as informações e facilitam a visualização do conjunto de dados, também chamado de análise exploratória;

V. Modelagem; No último passo do entendimento dos dados são aplicados diferentes modelos que visam retornar informações preditivas, relações existentes entre variáveis ou simulações. Se necessário for, o analista pode retornar à etapa de transformação para realizar alterações e aplicar novos modelos dentro do conjunto. Esse processo será executado *n* vezes, conforme a necessidade de cada estudo;

VI. Comunicação: Etapa destinada a demonstrar os resultados obtidos através da geração de relatórios e gráficos que ilustram a avaliação das hipóteses levantadas no início do estudo.

Esse modelo, como já abordado anteriormente, não é linear porque as etapas de “Transformação”, “Modelagem” e “Visualização” são iterativas. Depois de executadas essas etapas é preciso verificar se os resultados preliminares obtidos são satisfatórios frente ao objetivo traçado no início do projeto. Caso o resultado não seja positivo, há uma nova rodada de iterações com as três etapas, transformando novamente os dados e consequentemente realizando novas visualizações e modelagem com os mesmos. Dificilmente esse processo ocorre de maneira única, sendo necessário refazê-lo repetidas vezes para obter boas conclusões.

As fontes de origem de dados na grande parte dos casos não fornece uma estrutura pronta para que seja possível a realização de estudos e análises de dados, demandando um processo de organização e transformação da base. De acordo com Wickham (2023), o processo que engloba a organização e transformação de dados faz parte do conjunto de ações de alteração dos dados, onde busca-se no primeiro momento interpretar as variáveis existentes, organizando-as em classes e formatos adequados para a posterior criação, se

necessária, de variáveis e colunas que descrevem outros comportamentos existentes para cada uma das observações do conjunto.

A grande quantidade de observações existentes em um conjunto de dados pode dificultar o entendimento do comportamento do mesmo sem a utilização de visualizações que condensam e sintetizam essas informações. A análise descritiva de dados, que por Wickham (2023) é abordada como a etapa de visualização, consiste em utilizar métodos estatísticos para descrever de maneira objetiva as características mais importantes presentes nas observações do conjunto (Reis, E.A., Reis I.A., 2002).

Essas características podem ser ilustradas através de gráficos (dispersão, histogramas, barras), tabelas ou índices que representem valores como porcentagem ou razões entre variáveis. Ainda de acordo com Reis, E.A., Reis I.A. (2002) as análises descritivas têm por objetivo a identificação de valores que potencialmente representam *outliers*, pontuando de maneira completa o comportamento das variáveis nas observações do conjunto.

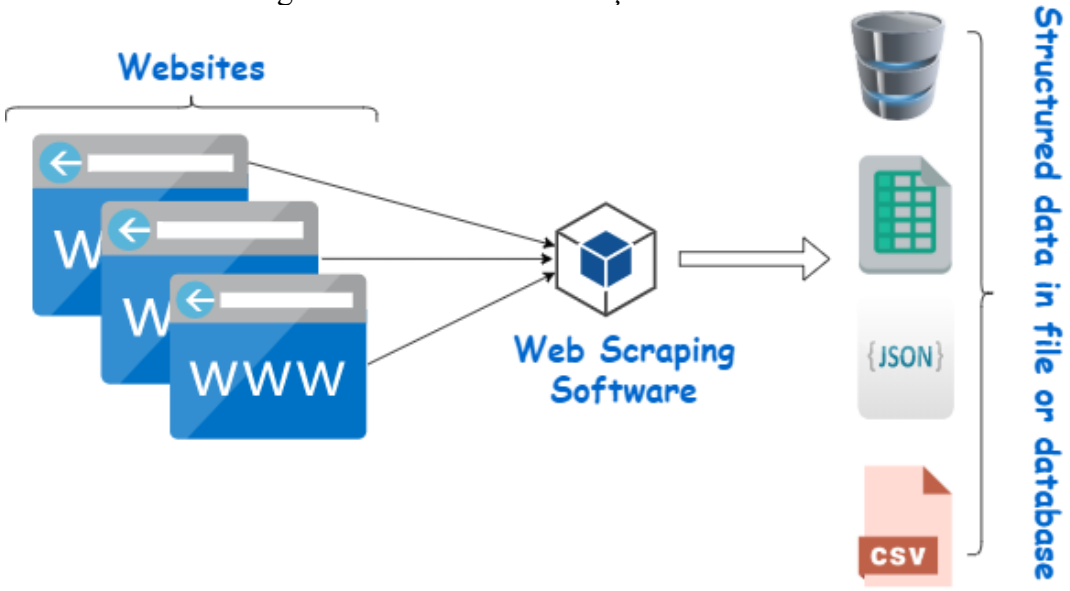
2.3 WEB SCRAPING

Silvestre (2007) descreve em seus estudos que os resultados obtidos para uma análise de dados partem do pressuposto de que há a necessidade da materialização dos dados para tal, ou seja, os dados precisam ser coletados de alguma forma. Dentro desse contexto, floresce a necessidade da coleta e parametrização de um conjunto de dados para ser analisado, onde o mesmo pode ser adquirido através do *web scraping.*

O *Web Scraping* segundo a definição de Krotov e Tennyson (2018) é o uso de ferramentas tecnológicas para a extração e organização automática de dados da *Web* com o objetivo de realizar análises posteriores desses dados. A utilização da tecnologia de *Web* Scraping apresenta uma ampla variedade de cenários de aplicação prática. De acordo com Zhou (2016) essa tecnologia possibilita o acompanhamento ou monitoramento de preços, a coleta de avaliações de produtos, a obtenção de listagens imobiliárias, entre muitas outras possibilidades.

A parte prática da extração de dados pode ser realizada através de duas formas: manualmente por um usuário ou automaticamente por um *bot* (algoritmo), utilizando-se dos *links* de páginas *web* que permitem a obtenção de diversas informações (OMARI, 2016 apud GALDINO, 2020). A extração por *bots* é capaz de interpretar dados e documentos em HTML, a linguagem utilizada em *webpages*, retornando resultados que podem ser salvos diretamente em arquivos CSV, JSON ou XML, conforme exemplificado na Figura 3.

Figura 3. Processos de extração de dados da web



Fonte: Nepomuceno (2016).

Essa automação da extração é comumente operada através de linguagens de programação e pacotes específicos, como o caso do *Beautifulsoup* que está disponível como uma biblioteca da linguagem Python (NAIR, 2014) ou o Rvest, um dos pacotes mais utilizados dentro da linguagem de programação R (SOETEWEY, 2023). No entanto, apesar de ser associado ao desenvolvimento de algoritmos, existem ferramentas gratuitas disponíveis na *internet* que visam entregar um resultado muito próximo aos de algoritmos citados anteriormente, operando através de interfaces amigáveis sem a necessidade profunda sobre programação (INDEED EDITORIAL TEAM, 2023).

2.4 MACHINE LEARNING

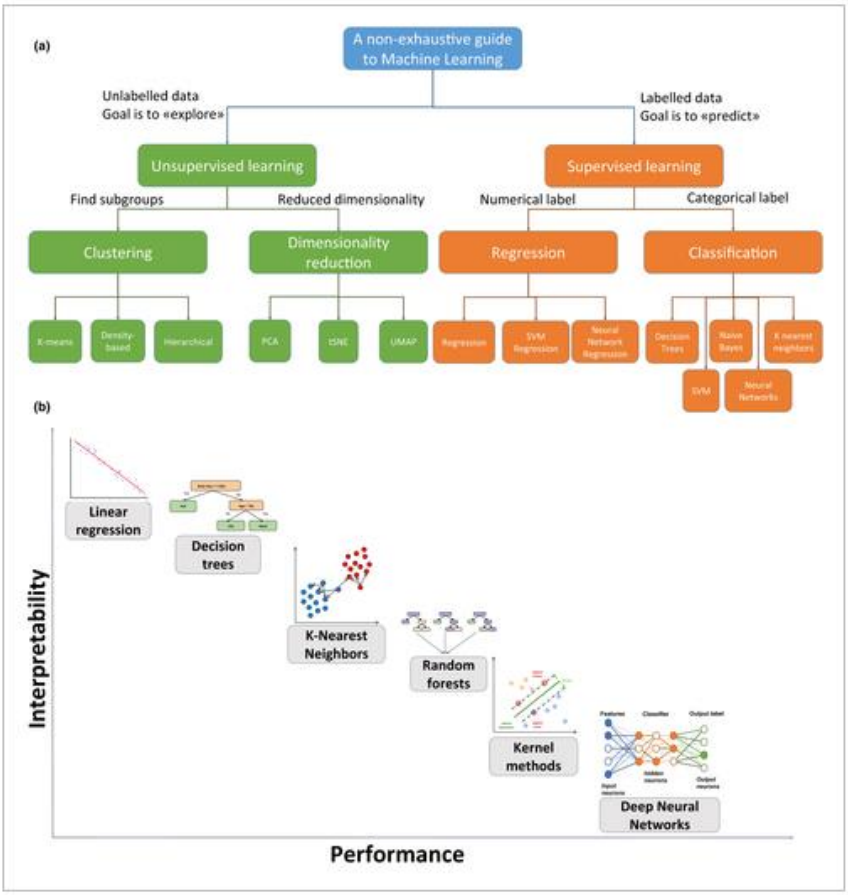
O *Machine Learning* (ML), ou aprendizado de máquina, é uma área em constante evolução que se baseia em algoritmos computacionais. Esses algoritmos são projetados para reproduzir a inteligência humana, modelando e aprendendo a partir do ambiente em que estão inseridos, desempenhando um papel fundamental para o tratamento de grandes quantidades de dados, sendo aplicados em diversos campos do conhecimento (NAQA, MURPHY, 2015). O objetivo da utilização desses algoritmos é gerar um resultado que possa representar um modelo e realizar previsões para um conjunto de dados.

Dentro dos estudos de ML existem dois grandes conjuntos em que podemos separar os algoritmos: supervisionados e não supervisionados. Os modelos que se utilizam da metodologia

supervisionada são caracterizados por objetivarem a previsão ou classificação a partir de um conjunto de dados já estabelecido para a realização de treinos e testes (JIANG et al., 2020), enquanto modelos não supervisionados visam encontrar relações entre os dados sem a existência prévia de um resultado já previsto, sendo úteis para tarefas descritivas (JIANG et al., 2020).

No contexto do mercado imobiliário a utilização de algoritmos de aprendizado de máquina se fazem cada vez mais presente, como no caso do estudo de Lorenz et al. (2022) que utiliza o algoritmo *eXtreme Gradient Boosting* (XGBoost) para realizar a previsão de aluguéis residenciais na cidade de *Frankfurt am Main*, sendo um dos pioneiros dentro desse cenário. O XGBoost é um exemplo de um modelo de ML que se enquadra nas técnicas supervisionadas, as quais são capazes de gerar dois *outputs* diferentes, a depender da particularidade de cada conjunto de dados e modelo. Dentro dos casos onde os resultados gerados são variáveis discretas, o modelo é caracterizado como um problema de classificação, enquanto resultados de variáveis contínuas se enquadram dentro de problemas de regressão (ZHOU, 2016). A figura 4 exemplifica de maneira visual como os modelos de ML são divididos e como os resultados são esperados para cada um deles.

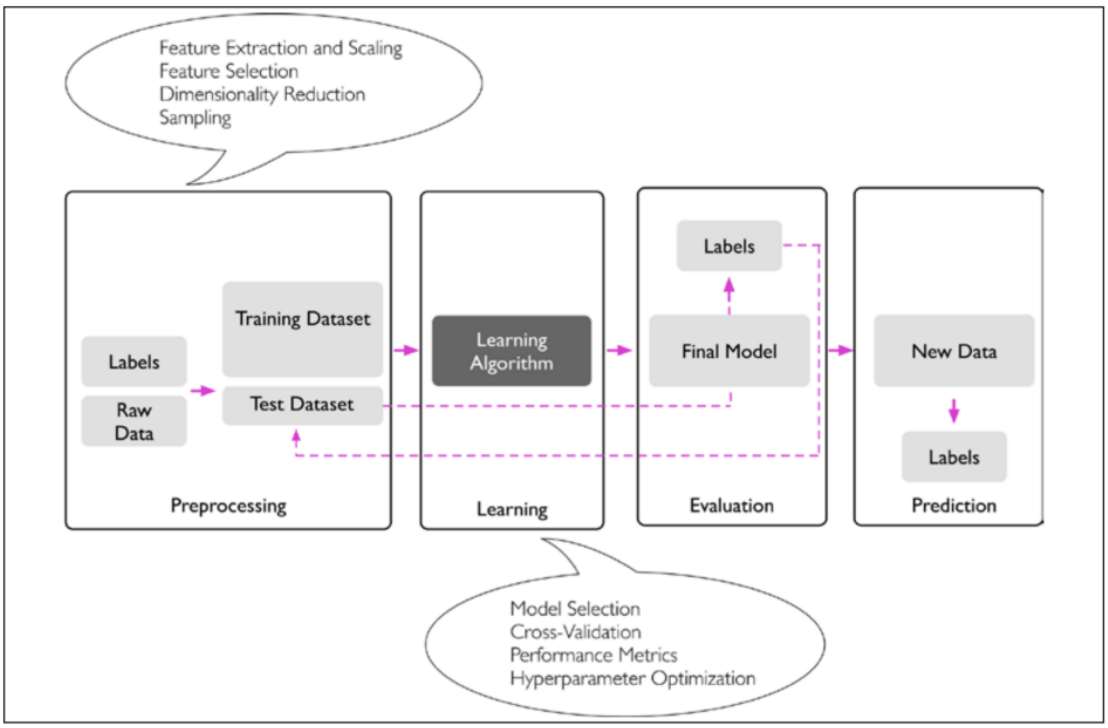
Figura 4. Modelos de Machine Learning



Fonte: BADILLO et al. (2020).

Assim como no tópico 2.1, a aplicação dos modelos de ML também segue uma sequência de etapas. Raschka et al. (2019) sugere um organograma (Figura 5) de passos para a aplicação de algoritmos de ML nos casos onde se deseja prever valores a partir a partir de um conjunto de dados já rotulados, caso que acontece para todos os modelos utilizados dentro do presente trabalho. As macro etapas presentes no *workflow* sugerido são dadas por: pré-processamento, aprendizado, avaliação e previsão.

Figura 5. Procedimentos para aplicação de Machine Learning



Fonte: Raschka et al. (2019).

A primeira macro etapa de pré-processamento é contemplada pelo modelo de Wickham (2023) que define todos os passos de tratamento e organização prévia de dados para a aplicação de modelos de ML. Nessa etapa mais especificamente são executados os processos de *feature engineering*, seleção de variáveis, redução das dimensões de variáveis categóricas e divisão do banco de dados entre treino, teste e validação, sendo o objetivo dessa etapa preparar o conjunto de dados para que os modelos de ML possam ser aplicados de maneira adequada. Na etapa seguinte parte-se para a parte prática do aprendizado de máquina, onde os algoritmos escolhidos serão submetidos ao treinamento e teste dentro dos conjuntos escolhidos. Nesse momento, a utilização de validação cruzada e otimização de hiperparâmetros são sugeridas por Raschka et al. (2019) para que os resultados obtidos possam ter a melhor performance frente às métricas de avaliação.

A subsequente etapa consiste em avaliar os modelos através das métricas de comparação entre os mesmos e interpretar se as previsões podem representar com assertividade o objetivo final do estudo. Wickham (2023) e Raschka et al. (2019) seguem o mesmo caminho e definem um modelo

iterativo, onde as atividades anteriores podem ser refeitas para melhorar a precisão e capacidade do modelo, fato esse que ocorre durante a etapa de avaliação das métricas que são os indicativos de desempenho dos modelos, cabendo ao usuário interpretar o que foi construído e executar atividades anteriores para corrigir ou melhorar o resultado obtido. Por fim, a etapa de previsão é onde é feita a escolha do melhor modelo de acordo com as métricas utilizadas e assim, geradas novas previsões para valores futuros ou para valores fictícios.

2.1.1 Feature Engineering

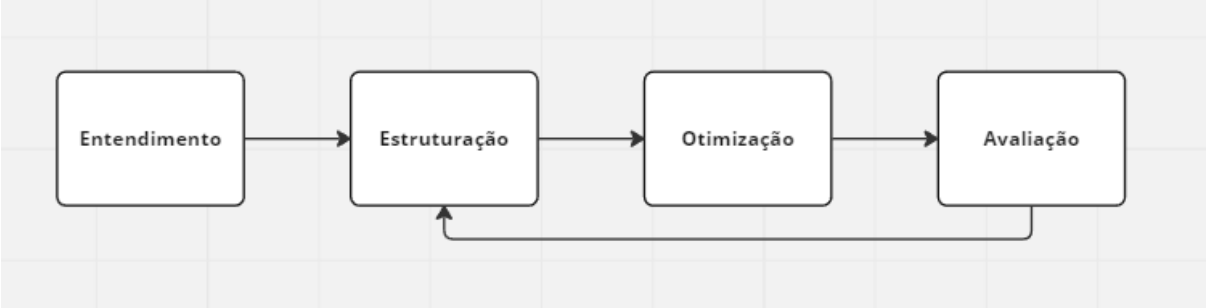
Anterior ao processo de *feature engineering* é preciso compreender qual a definição de uma *feature.* Dentro dos estudos e aplicações de aprendizado de máquina, uma *feature* é interpretada como um atributo (categórico) ou variável (não categórico) que fornece informações para as observações do conjunto de dados, sendo tratada como uma característica (DONG, 2018). De maneira sintética Ozdemir (2022) define o *feature engineering* como a capacidade de realizar a manipulação e transformação de dados em formatos que sejam corretamente interpretados por modelos de aprendizado de máquina de maneira otimizada.

A aplicação da *feature engineering* possui diferentes abordagens dependendo da perspectiva de cada necessidade de estudo. Na ótica de um analista de dados, esse processo também é definido como a capacidade de reduzir os recursos necessários para a execução dos algoritmos de aprendizagem de máquina supervisionados (OZDEMIR, 2022). Ainda segundo Ozdemir (2022), esse processo tende a ser trabalhoso se não for executado de maneira sistemática, o que sugere a criação de um processo para a execução dessa tarefa que pode ser dividida em quatro etapas: entendimento, estruturação, otimização e avaliação.

I. Entendimento: identificação de quais dados são importantes e quais estão disponíveis para aplicação dos modelos;
II. Estruturação: conversão de características para formatos adequados que possam ser interpretadas pelos modelos de aprendizado de máquina;
III. Otimização: etapa mais recorrente do trabalho; criação, modificação e extração dentro características utilizadas;
IV. Avaliação: avaliação do resultado das etapas anteriores e teste das características com diferentes modelos, utilizando diferentes combinações para averiguar a importância de cada uma. Se a avaliação for validada, finaliza-se o processo, caso contrário, volta para a etapa de Estruturação ou Otimização.

A figura 6 ilustra o processo de aplicação da *feature engineering* em um trabalho de análise de dados.

Figura 6. Procedimentos para aplicação de Feature Engineering



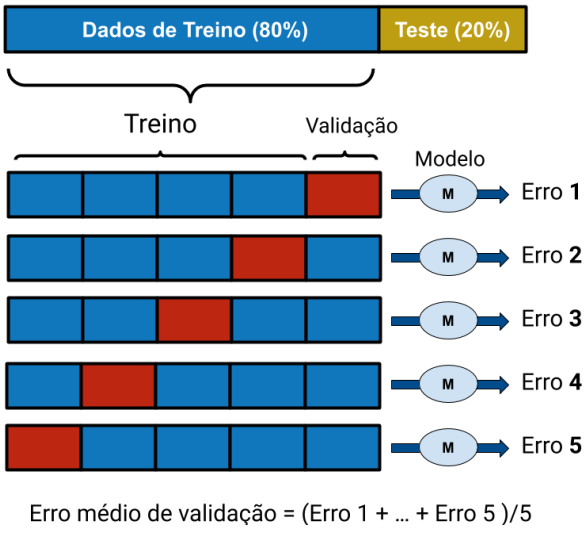
Fonte: Autor (2023).

2.4.1 Validação Cruzada

Zhang (1993) define que a ideia principal da validação cruzada consiste em realizar a separação dos dados em duas partes distintas (treino e teste), empregando uma das partes separadas dos dados para gerar uma regra de previsão e com os demais dados comparar a qualidade dos resultados previstos. A utilização dessa técnica de reamostragem permite estimar os verdadeiros erros existentes em modelos e aumentar a precisão dos mesmos. Ela pode ser executada através de diversos métodos distintos que podem ser: *k-fold* (aleatório ou não), *Leave-one-out* ou *Jackknife* (BERRAR, 2023); onde todas são derivadas da família de testes Monte Carlo, sendo a técnica *k-fold* a forma mais empregada nesse contexto.

A validação *k-fold,* também chamada de validação de dobras, divide os dados de treino aleatoriamente em *k* amostras de tamanhos próximos, chamadas de dobras, sendo cada uma utilizada de maneira única como parâmetro de avaliação do modelo. Como o conjunto de dados tem um tamanho finito, essa abordagem aumenta a interação existente dentro do mesmo conjunto, reduzindo as chances da ocorrência de *overfitting* do modelo (BERRAR, 2023). A Figura 7 ilustra o processo de utilização de validação cruzada com o número de dobras $k = 5$.

Figura 7. Exemplificação da aplicação de Validação Cruzada em um conjunto de dados



Fonte: Scaccia (2022).

Portanto, a validação cruzada retorna o resultado médio dos *k folds* treinados e o erro real do algoritmo passam a ser a média dos erros de cada treinamento realizado nas dobras, aumentando o espectro de testes, sem necessariamente aumentar o tamanho da amostra.

2.4.2 Otimização de Hiperparâmetros

Os hiperparâmetros são as variáveis dos modelos de *machine learning* que comandam todo o processo aplicado no conjunto de dados para a computação das previsões. Esses parâmetros externos podem ou não ser definidos anteriormente à execução dos modelos, a depender da escolha do usuário (PROBST et al., 2019). Por desempenharem um papel fundamental no desenvolvimento dos nos modelos, deve-se levar em consideração que a escolha dos hiperparâmetros influencia nos resultados finais obtidos com as previsões geradas.

A escolha desses hiperparâmetros é diferente para cada modelo de Machine Learning e pode ser feita manualmente com base em testes previamente executados na literatura, através dos valores padrões existentes em pacotes de software, por meio de tentativa e erro (PROBST et al., 2019) ou através do processo de otimização, visando obter melhores resultados para as previsões (PINA et al., 2019).

A escolha dos hiperparâmetros é de suma importância para os resultados dos modelos e pode ser realizada basicamente de duas maneiras distintas: manualmente ou automaticamente (PINA et al., 2019). A seleção de hiperparâmetros de forma manual

demanda tempo e experiência prévia dos usuários para que sejam assertivos o suficiente para gerar resultados satisfatórios para o modelo, o que torna esse um exaustivo trabalho. Todavia, a pesquisa automática poupa tempo do usuário e é eficaz na entrega de um conjunto de hiperparâmetros capazes de gerar bons resultados relativos às métricas avaliadas (WU et al., 2019).

Na busca por hiperparâmetros otimizados, existem várias abordagens desenvolvidas pela literatura, sendo uma das mais empregadas a busca de grade. A busca de grade segue uma abordagem exaustiva, testando todas as combinações possíveis de valores de hiperparâmetros dentro de um intervalo pré-definido pelo usuário em um conjunto de validação cruzada (WU et al., 2019). A utilização desse método permite explorar uma ampla variedade de configurações e encontrar aquela que resulta no melhor desempenho do modelo de acordo com as métricas avaliativas. No entanto, é importante ressaltar que a busca de grade exige grande capacidade computacional, uma vez que o mesmo testa múltiplas combinações diferentes para o mesmo modelo, o que pode gerar uma sobrecarga e lentidão na busca dos hiperparâmetros.

2.4.3 2.4.4 Algoritmos Escolhidos

2.4.3.1 Regressão Lasso

Proposto inicialmente por Tibshirani (1996), a técnica LASSO, *Least Absolute Selection and Shrinkage Operator* é um modelo utilizado dentro dos algoritmos de aprendizado de máquina e se destaca por apresentar bons resultados dependendo das suas configurações se comparados a outros métodos de regressão (RANSTAM et al., 2018). De acordo com Fonti (2017) as principais funções do modelo LASSO estão associadas a seleção de variáveis e regularização do modelo.

O LASSO estabelece uma restrição na soma dos valores absolutos dos parâmetros escolhidos do modelo, garantindo que essa soma seja inferior a um valor fixo, conhecido como limite superior, o qual pode ser definido pelo usuário ou através da validação cruzada. Para atingir esse valor inferior, o método adota um procedimento de redução, também chamado de regularização (TIBISHIRANI, 1996), que penaliza os coeficientes das variáveis de regressão, reduzindo alguns deles a zero. Durante o processo de seleção de recursos, apenas as variáveis que mantêm coeficientes não nulos após a etapa de encolhimento são escolhidas para integrar o modelo (FONTI, 2017) .

Como base, esse modelo utiliza dois hiperparâmetros distintos:

I. Penalty: quantidade de regularização utilizada. Quanto maior o valor aplicado, maior a penalização e consequentemente, mais coeficientes podem ser considerados nulos para o modelo.
II. Mixture: proporção da penalidade LASSO aplicada. Por definição, o valor pode ir de 0 a 1.

2.4.3.2 Random Forest

O Random Forest é um dos modelos mais empregados de ML na atualidade por ser capaz de atender a uma ampla variedade de problemas de previsão, além de possuir poucos hiperparâmetros existentes para serem ajustados (BIAU et al., 2016). Outro fator relevante é que esse algoritmo pode ser utilizado em ambas as oportunidades de estudo onde se pretende obter modelos de regressão ou modelos de classificação (BIAU et al., 2016).

Breiman (2001) que foi o idealizador desse modelo define que Random Forests são compostas por uma conjuntura de árvores de decisão, em que cada árvore depende dos valores de um vetor aleatório amostrado de forma independente, onde dentro de todo o conjunto da floresta a mesma distribuição é aplicada. A combinação dessas árvores de decisão é capaz de gerar um único resultado, o que se comparado à utilização de uma única árvore de decisão, apresenta desempenho superior devido à redução da variância do modelo (JÚNIOR, 2018). Outro ponto de extrema relevância para esse modelo é a não ocorrência de *overfitting*, uma vez que as múltiplas árvores de decisão possuem tamanhos diferentes, que quando combinados, trazem um modelo que é capaz de generalizar os resultados (BREIMAN, 2001).

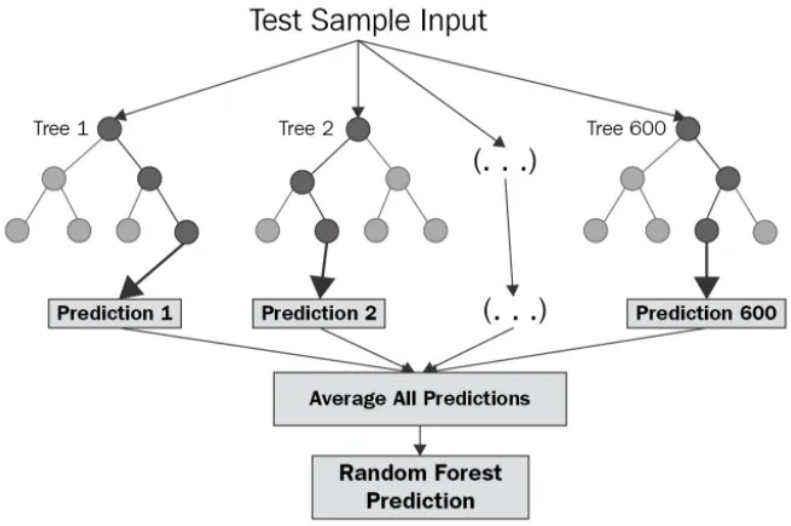
Para a aplicação prática, Biau (2016) pontua os três hiperparâmetros que regem o modelo de Random Forest:

I. *mtry*: quantidade de variáveis que serão selecionadas de forma aleatória para cada partição de árvore. Para casos de regressão, o valor inicialmente indicado para esse hiperparâmetro é o número total de variáveis do modelo dividido por três.
II. *Nodesize*: define o tamanho mínimo de nó de cada árvore. Por definição, o valor cinco é escolhido para regressões.

III. α_n Conhecida como *Bootstrapping*, esse hiperparâmetro tem a capacidade de trazer amostras únicas para cada árvore, aumentando a variação de dados dentro da floresta. Por definição, recomenda-se que seja usado o valor total de observações do conjunto numérico de treinamento.

Na Figura 8 pode-se entender o comportamento das múltiplas árvores de decisão que fornecem resultados únicos para um modelo de regressão utilizando-se do algoritmo RF. Com os resultados obtidos é feita então a média entre eles, resultando em um *output* mais provável para o modelo.

Figura 8. *Random Forest*

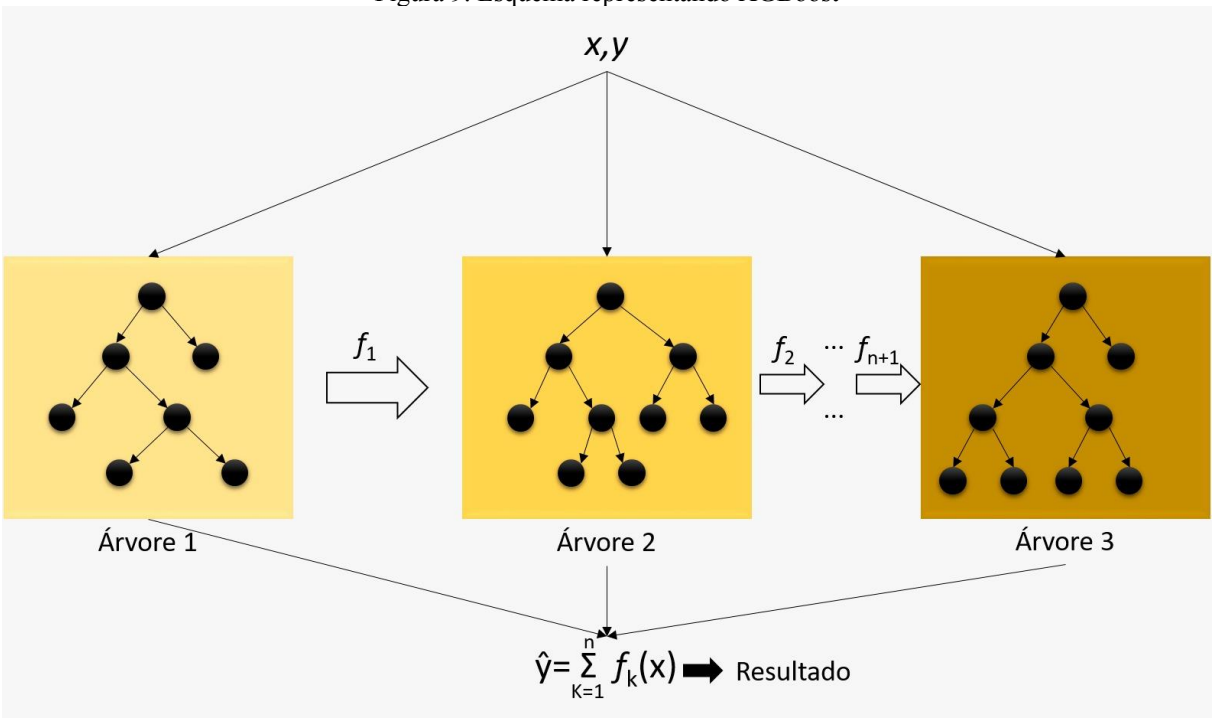


Fonte: Chirag Goyal (2021).

2.4.3.3 XGBoost

O *eXtreme Gradient Boosting* (*XGboost*), é amplamente conhecido por sua flexibilidade e usabilidade para a resolução de múltiplos problemas de regressão ou classificação, principalmente para conjuntos numéricos tabulares (ROLLI, 2020). Ele emprega o método de *gradient boosting*, que consiste em combinar várias árvores de decisão de baixa precisão para criar um modelo mais robusto, que seguindo um fluxo de construção sequencial (Figura 9), aprende com os erros do modelo anterior e realiza ajustes para minimizá-los (CHEN, 2016).

Figura 9. Esquema representando XGBoost



Fonte: Dados ao cubo (2023).

Os hiperparâmetros utilizados pelo XGBoost para gerar as previsões são dados por:

I. *mtry*: define a quantidade de variáveis que serão selecionadas de forma aleatória para cada partição de árvore. O número máximo possível é a quantidade de variáveis existentes no modelo.

II. trees: número de árvores aleatórias utilizadas no modelo.

III. *min_n*: determina o tamanho mínimo do nó durante o desenvolvimento da árvore. Quando a quantidade de observações em um único nó for menor do que o tamanho definido previamente pela variável, o iteração é interrompida.

IV. *tree_depth*: define a profundidade máxima das árvores de decisão utilizadas no modelo. Maior o valor utilizado, maior a probabilidade de *overfitting*.

V. *learn_rate*: conhecida como a taxa de aprendizagem, essa variável regula o peso de cada árvores para o modelo final. Valores menores aumentam a velocidade de treino e deixam o modelo com menores chances de overfitting. Por definição, os valores devem estar entre o intervalo de 0 a 1.

VI. *loss_reduction*: define a redução mínima aceita para a continuidade da divisão de nós da árvore. Em casos em que não for atingido o valor, não são feitas novas divisões de nós.

VII. *sample_size*: utilizado para selecionar a proporção de observações aleatórias que serão utilizadas como amostragem no treinamento do modelo.

VIII. *stop_iter*: Delimita o número máximo de iterações que vão ocorrer dentro do modelo, antes de ser interrompido. O valor selecionado nesse hiperparâmetro é o critério utilizado para a parada do modelo.

Nos últimos anos, o modelo tem sido empregado para modelar diversos casos de estudo do setor imobiliário. No estudo dirigido por Li (2021), o *XGBoost* foi empregado na cidade de Shenzhen para relacionar variáveis de geolocalização e os preços de imóveis para identificar a importância e influência de cada variável. Da mesma maneira, Zhao (2022) utiliza o *XGBoost* e o compararam diretamente com a utilização da regressão linear em um estudo voltado para realizar a predição de preços de imóveis em Pequim.

2.5 MÉTRICAS DE AVALIAÇÃO

As métricas de avaliação são utilizadas principalmente para averiguar a assertividade de modelos preditivos. Elas são capazes de retornar de maneira objetiva em um único valor o desempenho entre as previsões e os valores reais, sendo utilizadas como um parâmetro de comparação para os resultados obtidos com diferentes algoritmos aplicados em um mesmo conjunto de dados. A combinação dessas métricas em um sistema de avaliação é capaz de gerar conclusões mais consistentes a respeito da usabilidade dos modelos.

Neste capítulo serão apresentadas as quatro métricas utilizadas para parametrizar e escolher o melhor modelo preditivo, sendo elas: RMSE, RSR, R2 e MAPE.

2.5.1 RMSE

O *Root Mean Square Error* (RMSE) é o erro quadrático médio entre as observações reais e a previsão das mesmas em todo o conjunto de dados. A definição matemática do RMSE é dada por:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} (\gamma_i - p_i)^2}$$

onde:

n = número de observações do modelo
γ_i = valor real da observação i
p_i = valor previsto pelo modelo para a observação

É importante salientar que de acordo com Chai e Draxler (2014) a utilização dessa métrica é mais apropriada quando a distribuição dos erros do modelo for Gaussiana, evitando tendências de erro provocadas por *outliers* que acabam agregando na somatória de erros. Essa medida de erro é capaz de retornar apenas valores positivos (0, +∞), onde 0 é o valor ótimo para a métrica.

2.5.2 RSR

O *Root Mean Square Error Standard Deviation Ratio* (RSR) é uma métrica comumente utilizada dentro de estudos de avaliação hidrológica, mas que tem grande utilidade para avaliação de diversos modelos de aprendizado de máquina (GOLMOHAMMADI; PRASHER; MADANI; RUDRA, 2014).

A métrica de RSR pode ser calculada através da divisão do RMSE pelo desvio padrão da amostra utilizada para a realização das previsões. A Equação 2 representa matematicamente a formalização dessa métrica:

$$RSR = \frac{RMSE}{\sigma} = \frac{\sqrt{\frac{1}{n}\sum_{i=1}^{n}(\gamma_i - p_i)^2}}{\sqrt{\frac{1}{n}(\gamma_i - \underline{\gamma})}}$$

onde:

n = número de observações do modelo
γ_i = valor real da observação
$\underline{\gamma}$ = média dos valores das observações do conjunto de dados
p_i = valor previsto pelo modelo para a observação i

Ao parametrizar o RMSE pelo desvio padrão, o RSR é capaz de retornar apenas valores positivos (0, +∞), sendo 0 o valor ótimo para a métrica. Valores <1 indicam que que o RMSE possui maior precisão se comparado ao desvio padrão que retorna sobre a variabilidade dos dados, menor o valor de retorno, mais preciso tende a ser o modelo.

2.5.3 R²

Em paralelo a utilização de outras métricas, o coeficiente de determinação (R^2) realiza a avaliação dos pontos obtidos em um regressão, sendo interpretado como o quadrado da correlação amostral (MARTINS, 2018). A sua definição pode ser dada pela Equação 3.

$$R^2 = 1 - \frac{\sum_{i=1}^{n} (\hat{y}_i - p_i)^2}{\sum_{i=1}^{n} (\hat{y}_i - \underline{\gamma})^2}$$

onde:

$\hat{y}_i =$ valor do conjunto de treino para a observação i
$\underline{\gamma} =$ média dos valores do conjunto treino
$p_i =$ valor previsto pelo modelo para a observação i

2.5.4 MAPE

O *Mean Absolute Percentage Error* (MAPE), ou erro percentual absoluto médio, é usado para calcular a distância entre as observações e as previsões através da análise percentual. Essa medida é comumente usada em modelos onde é mais importante avaliar a variação relativa do que as variações absolutas em valores do mesmo (CHICCO; WARRENS; JURMAN, 2021). Apesar de ser de fácil intuição, essa métrica deve ser utilizada com certa cautela para modelos onde erros relativos maiores podem ser esperados.

$$MAPE = \frac{1}{n} \sum_{i=1}^{n} \frac{|\gamma_i - p_i|}{\gamma_i}$$

onde:

γ_i = valor real da observação i
p_i = valor previsto pelo modelo para a observação i
n = número de observações do modelo

O MAPE é capaz de retornar apenas valores positivos (0, +∞), onde 0 é o valor ótimo para a métrica.

3. METODOLOGIA

Este capítulo aborda o enquadramento da pesquisa, material e métodos utilizados para o desenvolvimento do trabalho e os procedimentos metodológicos, bem como os seus detalhamentos.

3.1 MATERIAL E MÉTODOS

O presente estudo teve como fonte de dados a plataforma de anúncios imobiliários Vivareal e o relatório de Março/2023 do Índice FIPEZAP+ de venda residencial. A partir de ambas as fontes foi possível extrair informações a respeito de imóveis na cidade de Florianópolis-SC.

Existem diversas formas de se conduzir um estudo de *machine learning*, mas uma das mais comuns e eficientes é através de linguagens de programação, já que as mesmas permitem a organização e modelagem dos dados que serão utilizados pelo algoritmo. Para realizar a análise estatística dos dados coletados de ambas as fontes citadas anteriormente foi aplicada a linguagem de programação R, sendo a mesma uma linguagem do tipo *open source,* utilizada principalmente para o desenvolvimento de trabalhos computacionais estatísticos e gráficos (VERZANI, 2011).

Além da linguagem, também foi utilizado o ambiente de desenvolvimento virtual *RStudio* para trazer praticidade e maior facilidade na visualização conforme o desenvolvimento do trabalho. O *Rstudio* é amplamente empregado dada a grande conveniência na utilização e instalação de pacotes, documentos e demais objetos necessários para a execução do estudo (VERZANI, 2011).

Outro meio que traz praticidade para a execução do projeto é o emprego de pacotes que reduzem a quantidade de código necessário. Diversos pacotes já vêm pré-instalados dentro da própria ferramenta (HAIR JUNIOR, 2021), mas para a realização do presente estudo outros pacotes foram necessários para realizar a execução de operações matemáticas, organização de dados e modelagem de algoritmos de *machine learning*. O Quadro 1 lista os principais pacotes utilizados durante a construção do trabalho além de breves descrições que contextualizam o uso dos mesmos e as suas respectivas referências bibliográficas.

Quadro 1. Principais pacotes utilizados na construção do trabalho

Pacote	**Descrição**	**Referência Bibliográfica**
skimr	Realiza o resumo de métricas mais importantes para um conjunto de dados	Waring E, Quinn M, McNamara A, Arino de la Rubia E, Zhu H, Ellis S (2022). _skimr: Compact and Flexible Summaries of Data_. R package version 2.1.5, <https://CRAN.R-project.org/package=skimr>.
parsnip	Interface para treinamento e teste de modelos de Machine Learning	Kuhn M, Vaughan D (2023). _parsnip: A Common API to Modeling and Analysis Functions_. R package version 1.1.0, <https://CRAN.R-project.org/package=parsnip>.
dplyr	Filtragem, seleção e reorganização do conjunto de dados	Wickham H, François R, Henry L, Müller K, Vaughan D (2023). _dplyr: A Grammar of Data Manipulation_. R package version 1.1.2, <https://CRAN.R-project.org/package=dplyr>.
tidyr	Utilizado para tratar valores ausentes e manipular colunas e seus formatos	Wickham H, Vaughan D, Girlich M (2023). _tidyr: Tidy Messy Data_. R package version 1.3.0, <https://CRAN.R-project.org/package=tidyr>.
dials	Especificação de hiperparâmetros e definição de grades de busca	Kuhn M, Frick H (2023). _dials: Tools for Creating Tuning Parameter Values_. R package version 1.2.0, <https://CRAN.R-project.org/package=dials>.
forcats	Utilizado para transformação de variáveis categóricas em fatores	Wickham H (2023). _forcats: Tools for Working with Categorical Variables (Factors)_. R package version 1.0.0, <https://CRAN.R-project.org/package=forcats>.
ggplot2	Realização de representações gráficas diversas	H. Wickham. ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag New York, 2016.
cowplot	Combinação e personalização de representações gráficas para diferentes variáveis e estilos	Wilke C (2020). _cowplot: Streamlined Plot Theme and Plot Annotations for 'ggplot2'_. R package version 1.1.1, <https://CRAN.R-project.org/package=cowplot>.
ranger	Implementação do algoritmo Random Forest dentro de modelos de Machine Learning	Marvin N. Wright, Andreas Ziegler (2017). ranger: A Fast Implementation of Random Forests for High Dimensional Data in C++ and R. Journal of Statistical Software, 77(1), 1-17. doi:10.18637/jss.v077.i01
xgboost	Implementação do algoritmo XGBoost dentro de modelos de Machine Learning	Chen T, He T, Benesty M, Khotilovich V, Tang Y, Cho H, Chen K, Mitchell R, Cano I, Zhou T, Li M, Xie J, Lin M, Geng Y, Li Y, Yuan J (2023). _xgboost: Extreme Gradient Boosting_. R package version 1.7.5.1, <https://CRAN.R-project.org/package=xgboost>.
glmnet	Implementação da regressão Lasso dentro de modelos de Machine Learning	Friedman J, Tibshirani R, Hastie T (2010). “Regularization Paths for Generalized Linear Models via Coordinate Descent.” _Journal of Statistical Software_, *33*(1), 1-22. doi:10.18637/jss.v033.i01 <https://doi.org/10.18637/jss.v033.i01>.
rsample	Ferramenta para criação de amostragens e criação de	Frick H, Chow F, Kuhn M, Mahoney M, Silge J, Wickham H (2022). _rsample: General Resampling

	conjunto de dados de validação (validação cruzada)	Infrastructure_. R package version 1.1.1, <https://CRAN.R-project.org/package=rsample>.
recipes	Preparação das etapas para serem aplicadas no conjunto de dados que permitem a análise	Kuhn M, Wickham H, Hvitfeldt E (2023). _recipes: Preprocessing and Feature Engineering Steps for Modeling_. R package version 1.0.6, <https://CRAN.R-project.org/package=recipes>.
tune	Habilita a tunagem e coleta dos dados de algoritmos submetidos a otimização de hiperparâmetros	Kuhn M (2023). _tune: Tidy Tuning Tools_. R package version 1.1.1, <https://CRAN.R-project.org/package=tune>.

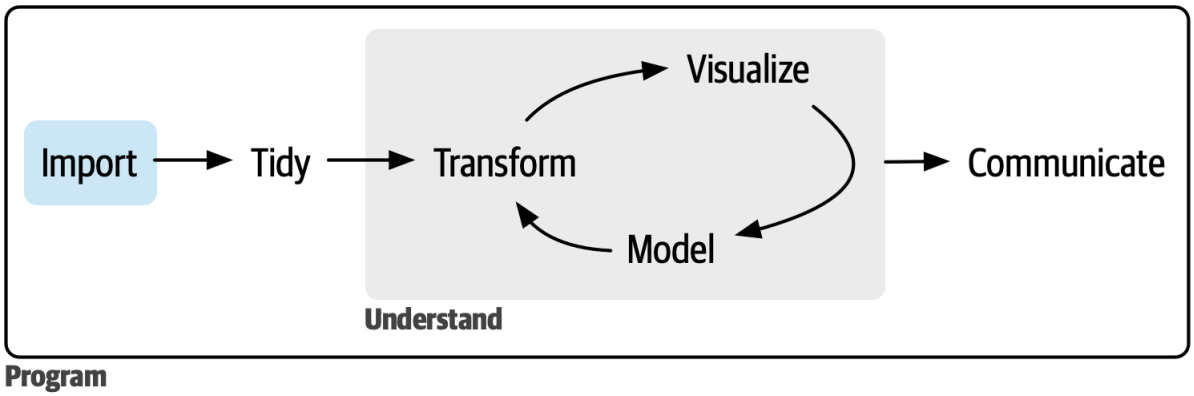
Fonte: Autor (2023).

3.2 PROCEDIMENTOS METODOLÓGICOS

O procedimento adotado foi baseado em Wickham (2023) e é composto por seis etapas distintas, conforme ilustrado na Figura 10: importação, organização de dados, transformação de dados, visualização de dados, modelagem e análise de resultados.

Deve-se pontuar também que para o presente estudo, bem como demonstrado pela ilustração (Figura 10), o processo de transformação, visualização e modelagem que compõe a etapa de entendimento dos dados é iterativo, o que levou a vários ciclos de implementação.

Figura 10. Processos para análise de dados



Fonte: Wickham (2023).

3.2.1 Importação

Na etapa de importação de dados foi realizada a coleta dos dados para a análise do cenário de vendas do mercado imobiliário da cidade de Florianópolis de acordo com a disponibilidade de informações existentes no portal imobiliário (referentes ao mês de Abril de 2023). Durante o processo de coleta nenhuma ordenação foi aplicada no site, a fim de evitar qualquer viés na seleção dos imóveis com base em faixas de preços específicas. Essa abordagem foi adotada para garantir a obtenção de um conjunto de dados mais representativo, abrangendo uma maior dispersão e variabilidade nos preços dos imóveis. No entanto, tal ação resultou em uma não homogeneidade da base de dados, o que foi explorado, analisado e devidamente tratado no decorrer do estudo.

Para coletar e importar dados reais do mercado imobiliário de Florianópolis a ferramenta *Data Miner* foi aplicada, uma vez que a mesma oferece uma interface amigável e requer um conhecimento básico de linguagem HTML para a extração das informações. Tais informações foram devidamente processadas e armazenadas em um banco de dados CSV para facilitar o acesso e a manipulação dos dados que posteriormente foram empregadas dentro do ambiente virtual do *RStudio*.

3.2.2 Organização de dados

A etapa de organização foi concentrada na estruturação dos dados de forma adequada. Essa etapa foi subdividida em dois subtópicos dada a ampla necessidade de diversos tratamentos e filtragens iniciais do conjunto de dados. Dentro dessa etapa foram englobados o tratamento de variáveis quantitativas e a remoção de duplicatas.

3.2.2.1 Tratamento de variáveis quantitativas

Ao analisar inicialmente o *dataframe* foi possível constatar que em algumas observações de colunas que deveriam ser quantitativas (numéricas) havia também caracteres não numéricos que impediam a utilização da observação por não serem reconhecidas como um número, mas sim como uma palavra *(character)* pelo programa.

Aplicando as bibliotecas *tidir* e *dplyr*, o primeiro passo para organizar os dados se consistiu em remover caracteres não numéricos de todas as observações das variáveis “Área”,

“Quartos”, “Banheiros”, “Vagas”, “Preço” e “Condomínio” que seriam utilizadas diretamente como valores numéricos dentro dos modelos, restando apenas números ou espaços vazios dentro das observações. Feita a remoção desses caracteres, todas as colunas citadas anteriormente foram convertidas para variáveis quantitativas discretas com a função *as.numeric()*.

Logo após a realização do tratamento foi possível evidenciar que diversas observações ficaram com valores muito acima do esperado para algumas colunas. Esse problema foi causado devido a existência de anúncios onde o imóvel anunciado era, na verdade, um anúncio de diversos imóveis e os seus valores eram demonstrados dentro de um intervalo de valores (Figura 11).

Figura 11. Intervalos existentes nas variáveis do portal imobiliário

Fonte: Autor (2023).

Os altos valores ocorreram porque após a remoção de caracteres não numéricos, os valores dessas variáveis que estavam dentro dos intervalos do anúncio e eram representados como “Quartos 2-3” acabavam retornando um valor de variável de 23, por exemplo. Tal fato ficou evidente ao ordenar qualquer uma das colunas numéricas de maneira decrescente.

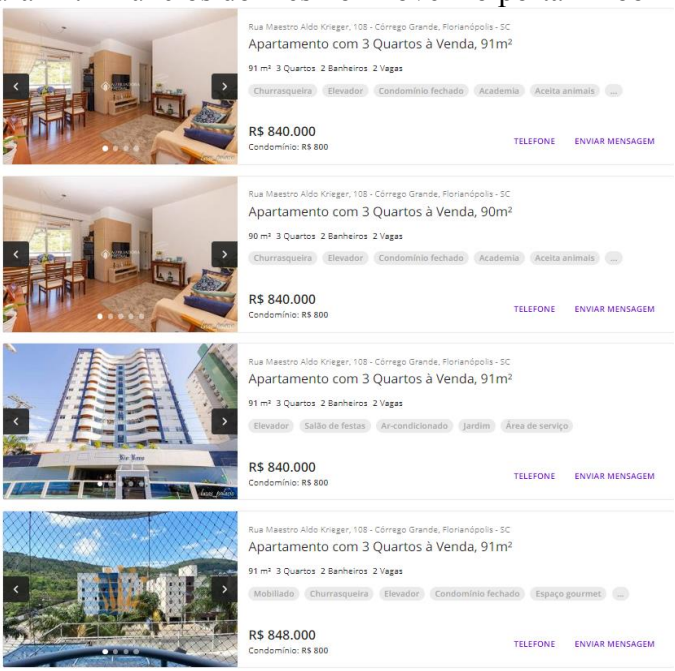
A solução adotada para esse caso foi a utilização de dois filtros. O primeiro realizou a remoção de imóveis com preços superiores a R$10.000.000,00, impedindo que a concatenação de valores pudesse atrapalhar o modelo. Enquanto o segundo filtro retirou imóveis que contenham mais de 11 banheiros, quartos ou vagas, já que o menor número possível para essa representação nº atributo - nº atributo após a remoção de caracteres não numéricos era 11.

3.2.2.2 Remoção de duplicados

A remoção de dados duplicados foi uma etapa crucial no estudo, uma vez que a presença de informações repetidas poderia introduzir algum viés e distorcer os resultados do trabalho. É comum no mercado imobiliário que um mesmo imóvel seja anunciado por diferentes corretores e imobiliárias em plataformas online, o que acaba resultando na ocorrência de múltiplos anúncios do mesmo imóvel dentro da mesma plataforma.

A Figura 12 ilustra essa situação, demonstrando que diferentes anúncios para o mesmo imóvel podem ser encontrados em momentos distintos durante o processo de extração de informações. Essa recorrência de dados duplicados foi considerada como um fator que poderia comprometer a confiabilidade dos dados e, consequentemente, os testes dos modelos nas análises posteriores.

Figura 12. Anúncios do mesmo imóvel no portal imobiliário



Fonte: Autor (2023).

Visando aumentar a integridade e confiabilidade dos dados e evitar distorções, os registros duplicados foram identificados e removidos do conjunto de dados. Para realizar a remoção eficaz de dados duplicados, foi utilizada a função *distinct()* da biblioteca *dplyr* onde a mesma realizou a varredura no banco de dados identificando imóveis que possuíam o mesmo valor para as variáveis "Preço", "Endereço" e "Descrição". A partir dessa análise, apenas uma observação de cada conjunto de dados duplicados foi preservada, eliminando as demais.

Essa ação acarreta em uma redução considerável no tamanho do banco de dados, uma vez que registros duplicados foram removidos. No entanto, essa abordagem reduziu as chances de ocorrência de duplicatas no modelo, permitindo uma análise mais precisa e confiável, evitando interpretações incorretas dos padrões e relações presentes nos dados.

3.2.3 Transformação de dados

Realizada a organização dos dados, foi iniciada a etapa de transformação dos mesmos. Nessa etapa, foram realizadas operações matemáticas e manipulações textuais que permitiram a criação de novas variáveis. Assim como na etapa anterior, a execução dessa atividade foi dividida em três subtópicos para facilitar o entendimento e interpretação da manipulação do conjunto de dados: criação de novas variáveis, tratamento de variáveis qualitativas e segmentação dos dados.

3.2.3.1 Feature Engineering

Dado o contexto do preço de um imóvel e do mercado como um todo, no presente modelo também buscou-se adicionar informações que pudessem agregar de maneira mais consiste o modelo, como a localização geográfica do mesmo ou o tipo de imóvel que está sendo analisado, sendo essa ação chamada de *feature engineering*. Objetivando-se agregar tais elementos, as colunas "Endereço" e "Descrição" foram manipuladas para que novas informações pudessem ser geradas a partir das mesmas.

Primeiramente adicionou-se a coluna Bairro, onde através da biblioteca de bairros utilizada pelo portal imobiliário e das funções *case when()* e *grepl()* foi possível identificar através da coluna "Endereço" em qual bairro de Florianópolis o imóvel se encontrava. Em caso de correspondência positiva, a coluna era preenchida com o nome do bairro identificado e em caso de não correspondência, a observação era descartada da base de dados.

Em seguida, utilizando-se de informações da coluna "Descrição", seguindo o mesmo raciocínio utilizado na criação da coluna "Bairro", foram identificados os tipos de imóveis possíveis com base na biblioteca utilizada pelo Vivareal. Assim, foi gerada a coluna "Tipo_Imovel", que indica o tipo de imóvel, como apartamento, casa, chácara, entre outros. Com a criação de duas colunas, tornou-se possível saber, por exemplo, se um determinado imóvel é um apartamento localizado no bairro João Paulo. O quadro 2 ilustra os resultados

descritos anteriormente. É possível verificar que dentro do conjunto de escrita o modelo é capaz de identificar em qual categoria se enquadra o imóvel.

Quadro 2. Exemplificação de fatores existentes na variável “Tipo Imóvel”

Criação da coluna "Tipo_Imovel"	
Descrição	Tipo_Imovel
Apartamento com 3 Quartos à Venda, 90m²	Apartamento
Casa com 3 Quartos à Venda, 180m²	Casa
Sala/Conjunto à Venda, 20m²	Sala/Conjunto
Apartamento com Quarto à Venda, 39m²	Apartamento
Edifício residencial com 2 Quartos à Venda, 519m²	Edifício Residencial

Fonte: Autor (2023).

Por fim, foi criada uma variável adicional para o modelo, chamada "Cond_fac", relacionada aos valores de condomínio dos imóveis. Considerando que nem todos os imóveis possuem esse valor (como casas, terrenos, etc.), foi estabelecida uma condição binária. Quando há valor de condomínio, é atribuído o valor 1; quando não há, é atribuído o valor 0. Dessa forma, descarta-se a magnitude do valor do condomínio, mas considera-se a presença do mesmo dentro dos modelos.

Além da adição de novas colunas para o banco de dados, também foram calculadas as métricas de preço de metro quadrado para cada observação que dividiram os valores da coluna “Preço” pelos valores da coluna “Área”.

3.2.3.2 Tratamento de variáveis qualitativas

Com o advento de novas variáveis criadas, foi preciso adequar-se para a utilização dentro dos modelos. A coluna "Tipo_Imovel" pode assumir até 16 valores únicos, enquanto o Bairro possui 64 valores únicos, todos eles interpretados como texto pelo programa. É importante ressaltar que o grande número de categorias distintas nessas variáveis categóricas dificulta a análise completa do modelo, uma vez que o conjunto de dados analisado não representa a totalidade dos imóveis anunciados em Florianópolis devido a inacessibilidade total a população, sendo necessária a realização de uma ação que pudesse ser capaz de

simplificar esses dados no modelo, bem como transformá-los em variáveis que pudessem ser interpretadas adequadamente.

Para traduzir as informações presentes nas colunas "Tipo_Imovel" e "Bairro" em variáveis categóricas, utilizou-se a função *as.factor()*. Essa função converte os valores das colunas em fatores, que são uma representação de variáveis categóricas no R. Portanto, ao utilizar a função *as.factor()*, está-se preparando as colunas "Tipo_Imovel" e "Bairro" para serem usadas em modelos de ML, permitindo que os algoritmos processem essas variáveis de maneira adequada e realizem previsões com base nas categorias definidas. Isso facilita a interpretação e o tratamento dessas variáveis ao longo do processo de modelagem e análise dos dados.

Como descrito anteriormente, havia também a necessidade de reduzir a complexidade e a quantidade de categorias nas colunas "Tipo_Imovel" e "Bairro", devido a baixa quantidade de dados. Visando solucionar essa questão, foram identificados os níveis (valores únicos) que representavam estatisticamente menos de 2% do total na coluna "Tipo_Imovel" e 1% na coluna "Bairro", sendo os mesmos agrupados em um único nível denominado "Outros", enquanto os níveis mais comuns foram mantidos para análise como variáveis únicas.

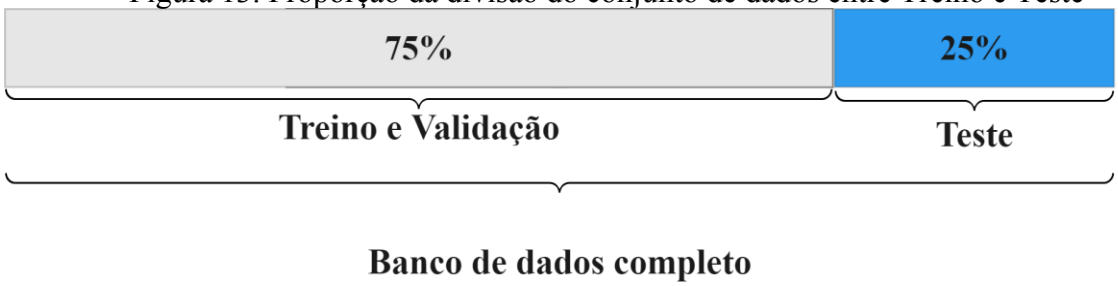
3.2.3.3 Segmentação de dados

No âmbito do desenvolvimento de modelos de ML é uma prática comum dividir o banco de dados em três conjuntos distintos: treinamento, teste e validação. Essa divisão desempenha um papel crucial na avaliação do desempenho e da capacidade de generalização do modelo.

O conjunto de dados de treinamento é utilizado para alimentar o modelo, permitindo que ele aprenda os padrões e as relações presentes nas variáveis. O conjunto de dados de teste é empregado para avaliar o desempenho do modelo em dados que não foram "vistos" durante o treinamento. Ele simula um ambiente real em que o modelo será aplicado e permite verificar como ele se comporta em situações não encontradas anteriormente, avaliando sua capacidade de generalização. Por sua vez, o conjunto de validação é utilizado para ajustar os hiperparâmetros do modelo. Ao realizar experimentos com diferentes valores de hiperparâmetros e avaliar o desempenho do modelo no conjunto de validação, é possível selecionar as configurações que resultam nos melhores resultados, de acordo com as métricas definidas para tal.

Para realizar essa divisão entre os conjuntos mencionados anteriormente, foi utilizada a biblioteca "*rsample*", que permite realizar a divisão com códigos simples. A proporção adotada para o conjunto de treinamento e validação foi de 75%, enquanto o conjunto de testes ficou com 25% dos dados. Essa divisão é ilustrada pela Figura 13.

Figura 13. Proporção da divisão do conjunto de dados entre Treino e Teste



Fonte: Autor (2023).

3.2.4 Visualização de dados

A etapa anterior que consiste em organizar e transformar o conjunto deixou os dados mais confiáveis e práticos para serem trabalhados no momento da interpretação gráfica.

Após a realização de todos os procedimentos anteriores, foram realizadas análises gráficas utilizando a biblioteca *ggplot2* para que fosse possível compreender como os dados se comportam e como os mesmos são distribuídos dentro do conjunto. Gráficos de dispersão, histogramas e tabelas foram gerados para as variáveis categóricas e não categóricas.

Para fins de comparação na assertividade da importação, limpeza e organização de dados, foram calculados os valores de metro quadrado dividindo as coluna "Preço" pela coluna "Área" para cada uma das observações sendo comparados diretamente com os dados do relatório do Índice FIPEZAP+ (Abril/2023).

3.2.5 Modelagem

Após a definição das variáveis, filtragem inicial e organização dos dados, três modelos de aprendizado de máquina foram utilizados para prever o preço de um imóvel conforme o objetivo do presente estudo. São eles: XGBoost, Regressão Lasso e Random Forest. Tais modelos foram selecionados devido à sua capacidade de lidar com problemas de regressão, bem como a ampla presença de outros estudos baseados na utilização desses modelos preditivos.

A execução dos três modelos (treinamento, e testes) foi dividida em duas etapas distintas. Na primeira etapa, o treinamento dos três modelos utilizaram hiperparâmetros pré-definidos, enquanto na segunda etapa, diversas combinações foram testadas. Essa abordagem de duas etapas objetivou parametrizar os resultados e observar a melhoria existente antes e depois do processo de tunagem de hiperparâmetros (combinação e teste entre os valores).

Para a implementação dos modelos foram empregadas as bibliotecas "ranger" para o algoritmo Random Forest, "xgboost" para o XGBoost e "glmnet" para o Lasso. Além disso, todos os modelos em ambas as etapas utilizaram a técnica de validação cruzada com 10 dobras da biblioteca "parnsip", visando aumentar a robustez através de diferentes combinações de conjuntos de treinamento.

3.2.5.1 Hiperparâmetros pré-definidos

A aplicação inicial dos modelos, sem a otimização dos hiperparâmetros, foi realizada para permitir uma comparação posterior com os resultados obtidos com ajuste dos parâmetros nos três modelos testados nos dados de entrada.

Para o modelo Random Forest, foram utilizados os hiperparâmetros "mtry = 2", "min_n = 2" e "trees = 500". No modelo XGBoost, os seguintes hiperparâmetros foram selecionados: "trees = 500", "mtry = 2", "min_n = 2", "learn_rate = 0.1" e "tree_depth = 3". Os demais referentes ao número de árvores, número de variáveis selecionadas aleatoriamente para cada divisão do nó e o número mínimo de observações necessárias para criar um novo nó foram mantidos iguais em ambos os modelos visando realizar a comparação direta de desempenho.

No caso do modelo Lasso, foram selecionados os seguintes valores: "penalty = 1" e "mixture = 1". Importante pontuar que o selecionado para "mixture" foi de uso proposital para que o modelo considerasse apenas a regressão Lasso no momento de treino.

3.2.5.2 Otimização de Hiperparâmetros

Para otimizar os resultados dos modelos, foram realizados testes com diferentes combinações de parâmetros, também chamado de tunagem de modelos. Essa otimização (tunagem) é permitida através do pacote *"dials"* que realiza a criação de grades de busca que vão testando diferentes parâmetros dentro da mesma aplicação.

O Quadro 3 ilustra os hiperparâmetros de grade de busca utilizados na linguagem de programação visando a otimização de resultados. O intervalo discreto é representado pelos valores entre parênteses ao lado de cada hiperparâmetro listado.

Quadro 3. Definição de hiperparâmetros para otimização utilizando busca de grade

XGBoost	Random Forest	Lasso
mtry (1-12) min_n (1-12) tree_depth (1-6) levels (6)	mtry (1-12) min_n (1-12) levels (6)	levels (100) penalty()

Fonte: Autor (2023).

No caso dos modelos Random Forest e XGBoost, o número de árvores anteriormente definidas como 500 foi aumentado para 1000 nos novos treinamentos. Os demais hiperparâmetros foram testados com diferentes combinações, visando encontrar a configuração que proporciona melhores resultados a fim de minimizar os erros entre os valores previstos e os valores reais do preço dos imóveis. O mesmo procedimento foi aplicado ao modelo Lasso, onde buscou-se otimizar a penalidade aplicada ao modelo.

3.2.6 Análise de resultados

Depois de realizados os ajustes dentro das etapas anteriores foi possível prosseguir para a última etapa do trabalho que consiste na análise de resultados. Durante essa etapa foram comparadas todas as métricas obtidas nos modelos para selecionar aquele que poderia desempenhar a melhor previsão. As métricas utilizadas durante essa etapa foram: RMSE, RSR, R^2 e MAPE. Além da comparação direta entre os modelos, também foram coletadas e parametrizadas as métricas entre os modelos que utilizaram a tunagem de hiperparâmetros e aqueles que não utilizaram.

Por fim, outras análises gráficas e conclusões também foram realizadas utilizando os pacotes *ggplot2* e *cowpolot*, a fim de promover a discussões e o entendimento dos resultados obtidos ao longo de todo o processo do estudo.

4. RESULTADOS

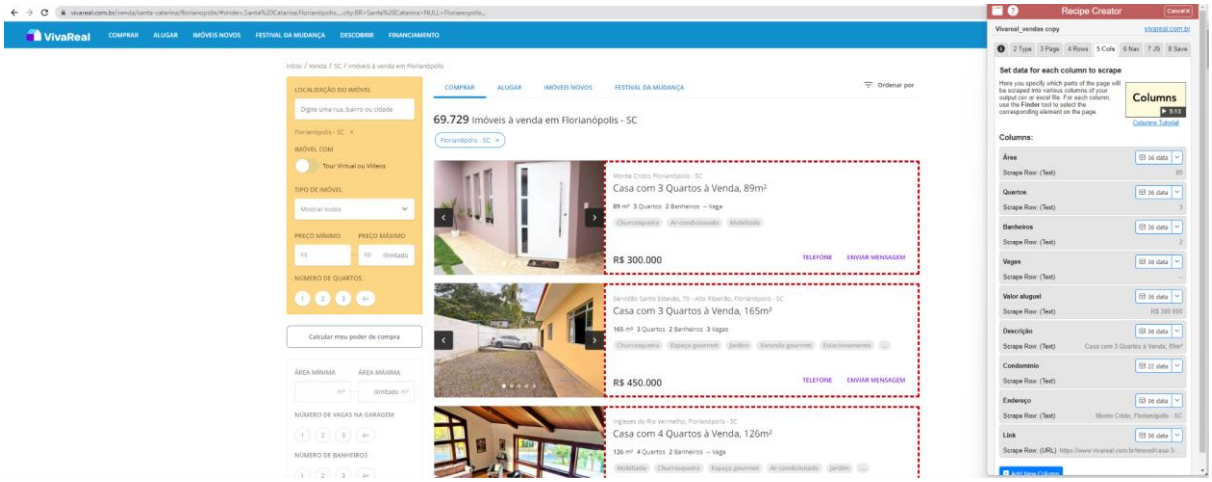
Neste capítulo, serão apresentados os resultados do trabalho. Todos os modelos de ML foram programados dentro do mesmo conjunto de dados (teste, treinos e validação). Os resultados são demonstrados por meio de análises gráficas relativas aos erros percentuais dos preços previstos e preços reais e através das métricas de desempenho RMSE, R^2, RSR e MAPE.

Devido às limitações de processamento computacional, os modelos foram treinados dentro das capacidades máximas disponíveis. Além das restrições de capacidade computacional, é importante ressaltar que o desenvolvimento dos modelos neste estudo foi desafiador devido à grande quantidade de dados incompletos e despadronizados, o que, consequentemente, exigiu etapas adicionais de ordenação e limpeza das variáveis.

4.1 IMPORTAÇÃO

A escolha das variáveis que fizeram parte da extração de dados do site foi pautada de acordo com a disponibilidade das informações, bem como informações que corriqueiramente são utilizadas como métricas básicas de um imóvel: área, localização, tipo de imóvel, etc. A Figura 14 ilustra a aplicação da ferramenta dentro do *website* em conjunto com a seleção de dados que seriam extraídos.

Figura 14. Utilização da ferramenta Data Miner no portal imobiliário



Fonte: Autor (2023).

As informações selecionadas foram aquelas disponíveis na página inicial de listagem de imóveis, sendo distribuídas em 9 colunas distintas para cada observação: Área, Quartos, Vagas, Banheiros, Preço, Descrição do imóvel, Condomínio, Endereço e Link. Devido às limitações da ferramenta, não foi possível obter informações adicionais, como descrições detalhadas ou atributos e facilidades específicas dos imóveis.

Quadro 4. Variáveis importadas e formatos interpretados pelo software R

Índice	**Coluna**	**Tipo de variável**
1	Área	<chr>
2	Quartos	<chr>
3	Banheiros	<chr>
4	Vagas	<chr>
5	Preço	<chr>
6	Descrição	<chr>
7	Condomínio	<chr>
8	Endereço	<chr>
9	Link	<chr>

Fonte: Autor (2023).

O banco de dados resultante foi armazenado em formato CSV, totalizando 10.800 observações (Figura 15). No entanto, é importante destacar que nem todas as observações possuíam informações completas em todas as colunas. Além disso, não há uma padronização específica para a descrição ou o endereço dos imóveis, o que significa que algumas informações estavam ausentes e eram apresentadas de maneiras diferentes.

Figura 15. Banco de dados resultante em formato CSV.

```
Rows: 10,800
Columns: 9
$ Área       <chr> "90", "53-264", "81", "64", "1494", "180", "62", "33", "70", "73", "95", "75", "115", "135", "9…
$ Quartos    <chr> "3", "1-4", "2", "--", "--", "3", "2", "--", "2", "2", "2", "2", "3", "3", "2", "1", "2", "3", …
$ Banheiros  <chr> "2", "2-5", "2", "1", "--", "2", "2", "1", "3", "1", "2", "1", "2", "3", "3", "1", "3", "2", "2…
$ Vagas      <chr> "2", "1-4", "1", "--", "--", "2", "1", "--", "1", "--", "1", "1", "2", "2", "2", "1", "2", "1",…
$ Preço      <chr> "R$ 276.000\n                          \n                          Preço abaixo do mercado", "De R$…
$ Descrição  <chr> "Apartamento com 3 Quartos à Venda, 90m²", "D/Yard Home Design", "Casa com 2 Quartos à Venda, 8…
$ Condomínio <chr> "R$ 200", NA, NA, NA, "R$ 1", NA, "R$ 499", "R$ 264", NA, "R$ 600", "R$ 480", "R$ 100", NA, NA,…
$ Endereço   <chr> "Servidão Fermino Severino Sagaz, 125 - Ingleses do Rio Vermelho, Florianópolis - SC", "Rua Pre…
$ Link       <chr> "https://www.vivareal.com.br/imovel/apartamento-3-quartos-ingleses-do-rio-vermelho-bairros-flor…
```

Apesar da existência de dados ausentes e demais pontos abordados anteriormente, a extração e importação dos dados utilizando a ferramenta Data Miner teve resultados

satisfatórios, sendo capaz de coletar cerca de 15% de todos os imóveis anunciados dentro da plataforma imobiliária (referente ao mês de Abril/2023). Mesmo sendo uma ferramenta gratuita, se mostrou extremamente capaz e poderosa para realizar a extração de dados de *webpages*.

4.2 ANÁLISE EXPLORATÓRIA

Após concluir a etapa inicial de preparação do banco de dados, ocorreu uma significativa redução no número de observações. Inicialmente, o banco de dados continha 10.800 observações, porém com as etapas de tratamento de dados passou a possuir 6.734 observações. Uma redução próxima a 38% frente ao número de observações inicialmente coletadas.

Para compreender como cada uma das variáveis independentes se comportam foi feita a análise exploratória de dados. No primeiro momento as análises foram separadas entre variáveis categóricas e não categóricas e posteriormente combinadas para formar tabelas e demais conjuntos de dados. O quadro 5 condensa as principais métricas para as variáveis categóricas do conjunto de dados.

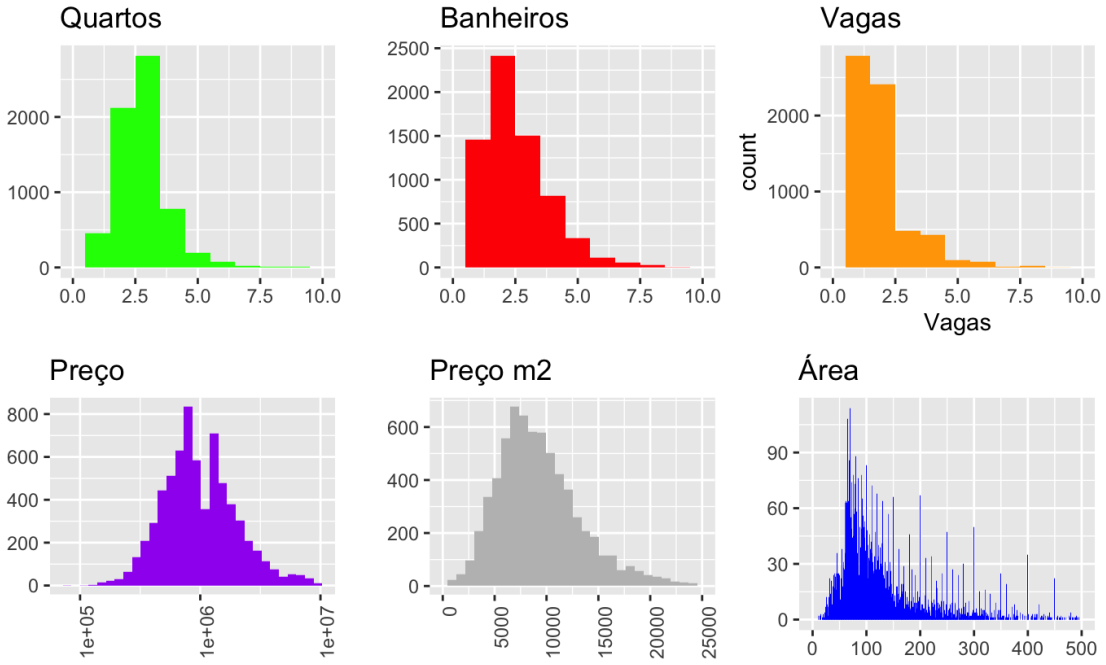
Quadro 5. Avaliação preliminar do comportamento das variáveis numéricas

Variável	**Média**	**Desvio Padrão**	**Mínimo**	**P25**	**P50**	**P75**	**Máximo**	**Mediana**
Área	192,1417	1784,194	10	74	107	175	130000	107
Quartos	2,788515	1,076957	1	2	3	3	16	3
Banheiros	2,540244	1,326567	1	2	2	3	10	2
Vagas	1,946398	1,524931	1	1	2	2	50	2
Preço	R$ 1.286.352,00	R$ 11.695.820,00	R$ 82.000,00	R$ 606.066,00	R$ 900.000,00	R$ 1.500.000,00	R$ 10.000.000,00	R$ 900.000,00
Preço_metro_quadrado	R$ 9.400,74	R$ 4.575,26	R$ 5,76	R$ 6.465,00	R$ 8.767,00	R$ 11.538,00	R$ 122.400,00	R$ 8.767,28

Fonte: Autor (2023).

Além das variáveis numéricas, haviam também variáveis categóricas consideradas dentro do modelo. Essas variáveis não categóricas também foram representadas em formato de gráficos histograma como ilustrado na Figura 16.

Figura 16. Histograma de variáveis não categóricas



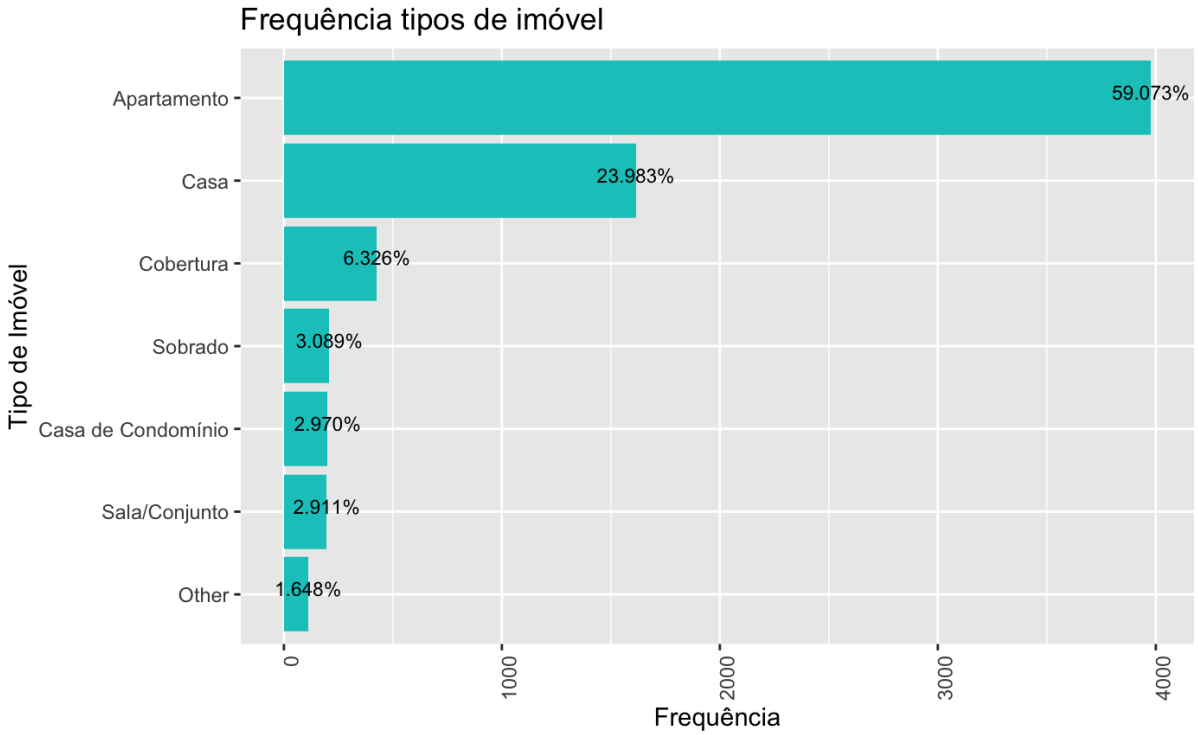
Fonte: Autor (2023).

A partir da ilustração gráfica das variáveis numéricas é possível verificar que há uma maior densidade de imóveis que possuem menor tamanho, dadas as quantidades de quartos, banheiros, vagas e área dos mesmos. Esse resultado já era esperado, uma vez que o mercado imobiliário tem como objetivo atender à demanda predominante de residências que comportam até 4 ocupantes na média.

Observa-se também que a distribuição da variável “Preço m2” apresenta forma semelhante a uma curva normal, o que comparativamente às demais variáveis pode ser interpretada como uma distribuição mais homogênea e com menores tendências.

Prosseguindo para as variáveis independentes, "Bairro" e "Tipo_Imovel" foram impressas em histogramas na Figura 17 e Figura 18 que possuem a informação da representatividade em porcentagem de cada uma no conjunto de dados tratados .

Figura 17. Histograma da variável “Tipo_Imovel”



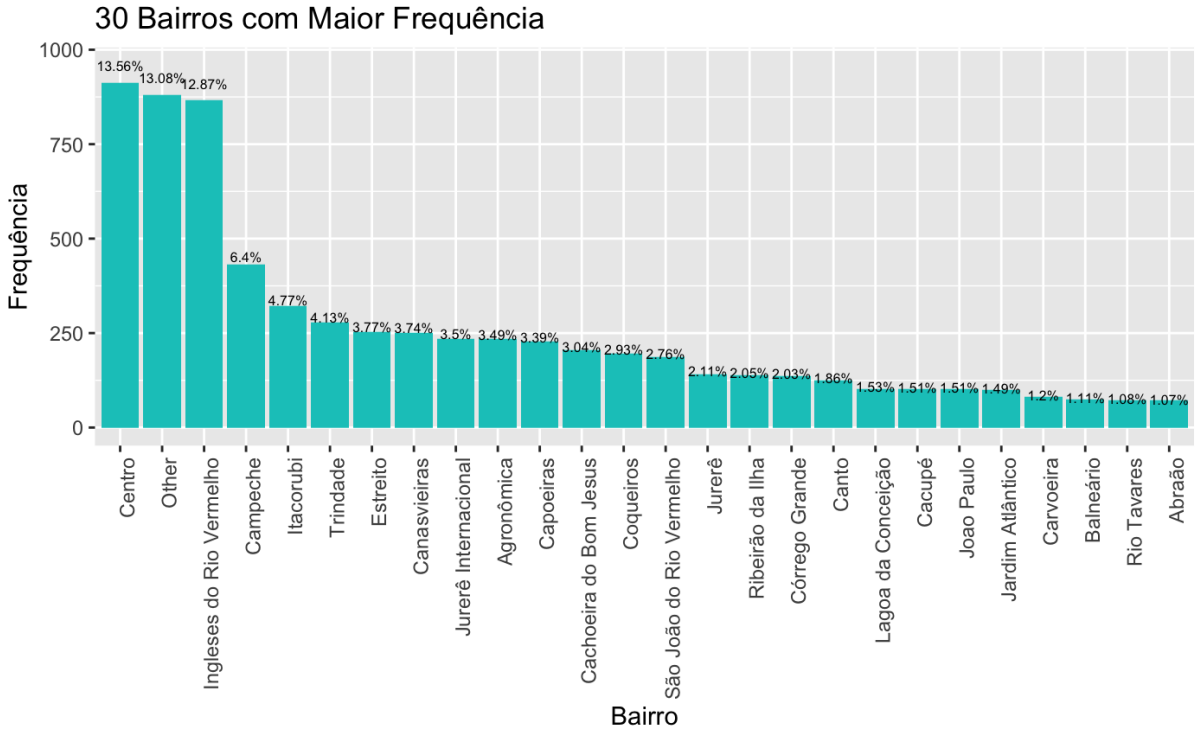
Fonte: Autor (2023).

Através da Figura 17 que representa o histograma da variável "Tipo_Imovel" é possível perceber que a maior concentração está presente entre imóveis do tipo "Apartamento" e do tipo "Casa", totalizando um valor próximo à 82% de todo o conjunto de dados. As observações do tipo "Other" são quase insignificantes para o conjunto, reforçando que a concentração de imóveis à venda encontra-se dentro das duas categorias pontuadas anteriormente.

Ao observar novamente o histograma da variável "Tipo_Imovel" vê-se que a grande concentração dos imóveis encontra-se em apartamentos (59,07%), o que naturalmente se reflete na concentração dos dados para as variáveis numéricas. Diferentemente de uma casa, um apartamento possui maiores limitações de espaço, o que consequentemente impacta na disponibilidade de quartos, vagas, banheiros e área útil total do imóvel e, como verfica-se, a maior concentração dos imóveis encontra-se mais próxima aos limites inferiores para os valores possíveis de cada variável.

Já quando interpreta-se a distribuição da variável "Bairro", a Figura 18 ilustra os 30 bairros que mais aparecem no conjunto e vê-se que a mesma é mais homogênea entre as observações, resultado que também já era esperado, dada a natureza dessa variável que possui um maior espectro de valores possíveis se comparado à coluna "Tipo_Imovel".

Figura 18. Histograma da variável “Bairro”



Fonte: Autor (2023).

A concentração dos dados encontra-se nos bairros onde há a maior densidade populacional da cidade que nesse caso, são representadas por "Centro" e "Ingleses do Rio Vermelho".

Um ponto relevante e de interesse para a análise é o fator "Other" que dentro do contexto da variável "Bairro" possui grande relevância, representando 13,08% das observações registradas. A representatividade desse valor dentro do conjunto de dados, no entanto, indica que há uma grande parcela de bairros da cidade que possuem poucas unidades imobiliárias à venda, o que, por consequência, impede que o modelo seja capaz de atuar de maneira efetiva para todos os bairros existentes, uma vez que muitos deles ficam mascarados dentro do valor "Other".

4.3 VALIDAÇÃO DO BANCO DE DADOS

O setor imobiliário é uma das que mais carecem de dados, não havendo uma centralização e nem uma convenção única para a fonte deles. No entanto, o Índice FIPEZAP+ (2023), apesar de não ser um índice oficial, ainda sim pode ser considerado amplamente relevante para medir diversos indicadores desse setor. Portanto, para avaliar os resultados obtidos com a importação, organização e o tratamento de dados, os indicadores de preço por metro quadrado foram calculados, já que essa é a métrica principal utilizada dentro do índice.

O Quadro 6 representa a comparação do preço de metro quadrado para alguns bairros existentes para Florianópolis dentro do índice onde a escala de cores na coluna "Variação" indica o distanciamento percentual absoluto entre os valores do conjunto de dados e os valores divulgados no relatório. Já a coluna "Representatividade" enumera a frequência percentual que cada observação aparece no conjunto de dados coletados do portal imobiliário.

Quadro 6.Comparativo Índice x Conjunto de dados para a variável “Bairro”

Bairo	**Valor m2**		**Variação**	
	FIPEZAP+	**Banco de dados**		**Representatividade**
Centro	R$ 10.531	R$ 10.410	-1,16%	13,56%
Ingleses do Rio Vermelho	R$ 7.562	R$ 7.001	-8,01%	12,87%
Itacorubi	R$ 10.321	R$ 10.193	-1,26%	4,77%
Trindade	R$ 10.141	R$ 9.741	-4,11%	4,13%
Estreito	R$ 7.374	R$ 8.081	+8,75%	3,77%
Agronômica	R$ 12.852	R$ 12.276	-4,69%	3,49%
Córrego Grande	R$ 11.449	R$ 10.125	-13,08%	2,03%
Coqueiros	R$ 8.429	R$ 8.211	-2,65%	2,93%

Fonte: Autor (2023).

A mesma métrica foi calculada e sumarizada para variável "Quartos", conforme o Quadro 7.

Quadro 7. Comparativo Índice x Conjunto de dados para a variável “Quartos”

Quartos	**Valor m2**	**Variação**	**Representatividade**

	FIPEZAP+	Banco de dados		
3	9813	9343	-5,03%	41,75%
2	9750	9393	-3,80%	31,40%
4	9318	9268	-0,54%	11,52%
1	10828	11957	+9,44%	6,78%

Fonte: Autor (2023).

Quando analisada a coluna "Variação", em ambos os casos a diferença existente entre os valores do índice e os valores calculados são baixos, o que demonstra que o conjunto de dados coletados e tratados é capaz de representar de maneira próxima a realidade os preços praticados pelos anúncios, tornando útil a aplicação dos mesmos nos algoritmos de ML.

Ademais, constata-se também que os atributos com maiores representatividades não contemplam erros tão grandes se comparados aos de menor representatividade. Ao levar em conta a inacessibilidade à toda população, pode-se verificar que esse conjunto é capaz de representar os preços de imóveis em Florianópolis/SC.

4.4 COMPARAÇÃO DOS RESULTADOS

Os algoritmos utilizados no estudo foram treinados, validados e testados com o emprego de hiperparâmetros pré-definidos e otimizados. Essa abordagem permitiu uma comparação entre os algoritmos, a fim de determinar qual deles apresentava o melhor desempenho na previsão de preços de imóveis em Florianópolis/SC. Para essa avaliação, foram utilizadas métricas de RMSE, RSR, R^2 e MAPE . O quadro 8 sintetiza de forma concisa as três métricas mencionadas anteriormente para os modelos testados, exibindo os valores correspondentes aos melhores desempenhos obtidos em cada uma delas.

Quadro 8.Métricas dos modelos de Machine Learning

Método	RMSE	R^2	RSR	MAPE
Lasso	7,58E+05	0,572	0,648	41,95%
Lasso (tune)	7,12E+05	0,654	0,609	40,01%
Random Forest	6,84E+05	0,733	0,585	42,23%
Random Forest (tune)	5,16E+05	0,818	0,441	25,84%
XGBoost	5,86E+05	0,762	0,501	31,51%

XGBoost (tune)	5,19E+05	0,814	0,443	27,79%

Fonte: Autor (2023).

No contexto da aplicação de otimização de hiperparâmetros evidencia-se que a mesma teve a capacidade de reduzir o erro e aumentar a acurácia nos três modelos. Esse já era um resultado previsto inicialmente no estudo, já que essa técnica ajuda na capacidade de desempenho de algoritmos. Para o modelo de *randon forest* otimizado o resultado que apresentou menor RMSE teve os seguintes parâmetros: "mtry = 12", "min_n = 1" e "trees = 1000". Já para o modelo de XGBoost os parâmetros que obtiveram o menor RMSE foram: "trees = 1000", "mtry = 1", "min_n = 12", "tree_depth = 6".O maior beneficiado da otimização foi o modelo Random Forest, que anteriormente na etapa de hiperparâmetros pré-definidos tinha como resultado um RMSE de 6,84E+05 e não foi capaz de desempenhar tão bem quanto o XGBoost que retornou o valor de 5,86E+05. O Quadro 9 ilustra a variação existente através da coluna "Δ% " que aponta a porcentagem de redução entre os modelos com e sem a utilização de tuning na comparativa direta da métrica de erro RMSE.

Quadro 9.Comparativo entre modelos otimizados e não otimizados através da métrica RMSE

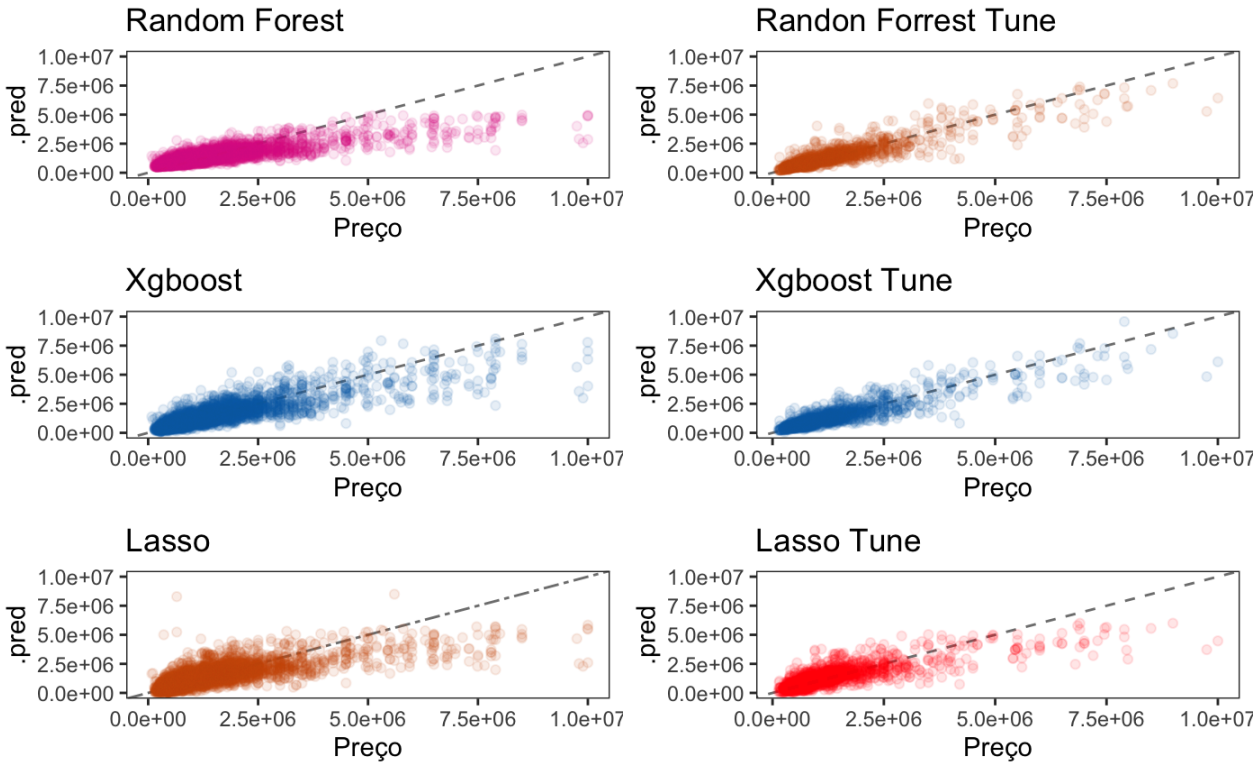
ML	**RMSE**		**Δ%**
	Pré-definidos	**Tune**	
Lasso	7,58E+05	7,12E+05	6,08%
Random Forest	6,84E+05	5,16E+05	24,53%
XGBoost	5,86E+05	5,19E+05	11,49%

Fonte: Autor (2023).

Por meio da análise anterior compreende-se que os modelos de busca de grades foram extremamente efetivos para a otimização dos parâmetros, o que consequentemente resultou na redução do erro RMSE em todos os três modelos utilizados.

Outra maneira de demonstrar os resultados e entender como a otimização dos hiperparâmetros foi de suma importância é através do gráfico de dispersão ilustrado na Figura 19, onde a esquerda encontram-se os modelos sem otimização e a direita com.

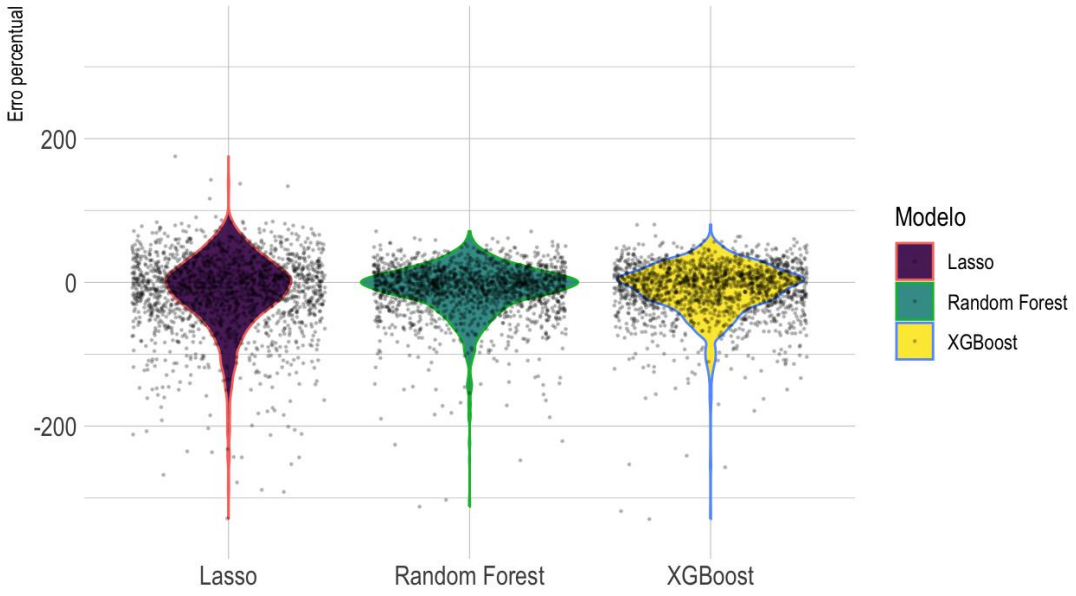
Figura 19 . Gráfico de dispersão entre valores reais e valores previstos para todos os modelos treinados



Fonte: Autor (2023).

A linha tracejada em cinza, utilizada como linha diagonal de apoio apresenta a paridade entre preços previstos e preços reais. O adensamento dos pontos previstos na linha de apoio em todos os modelos de *tuning* reforça o efeito positivo da busca de grades neste estudo, em especial para o *random forest*. No entanto, ao visualizar esse gráfico existe dificuldade em compreender se há uma diferença notória entre o XGBoost e o Random Forest. Visando complementar essa análise, as métricas de erro percentuais também foram operadas através da representação gráfica. A Figura 20 demonstra um gráfico do tipo *boxplot* que representa a concentração dos erros nos modelos com hiperparâmetros otimizados.

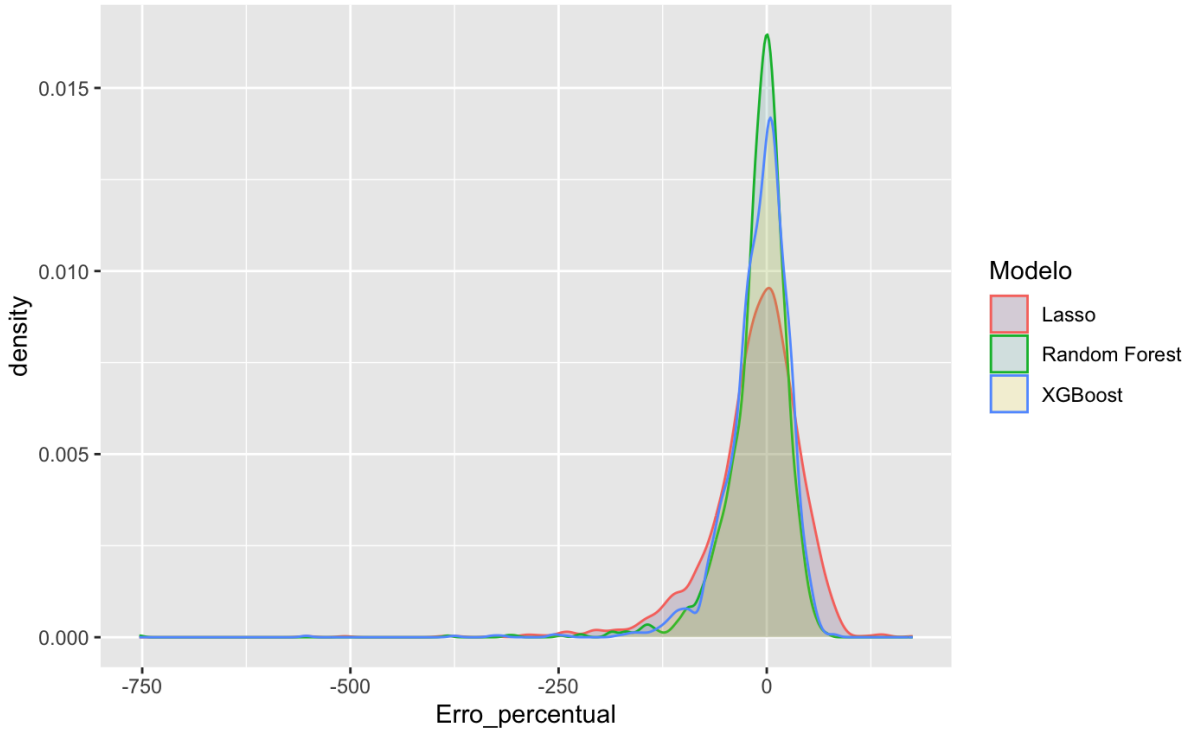
Figura 20. Boxplot dos erros de algoritmos de Machine Learning Otimizados



Fonte: Autor (2023).

A partir da imagem anterior nota-se que o modelo *Random Forest* é aquele que possui a menor taxa de erros percentuais e, portanto, a maior concentração de erros próxima a zero comparativamente aos demais modelos. A Figura 21 também foi utilizada como apoio para representar a comparação direta entre os erros existentes dentro dos modelos.

Figura 21. Densidade acumulado dos erros de algoritmos de Machine Learning Otimizados

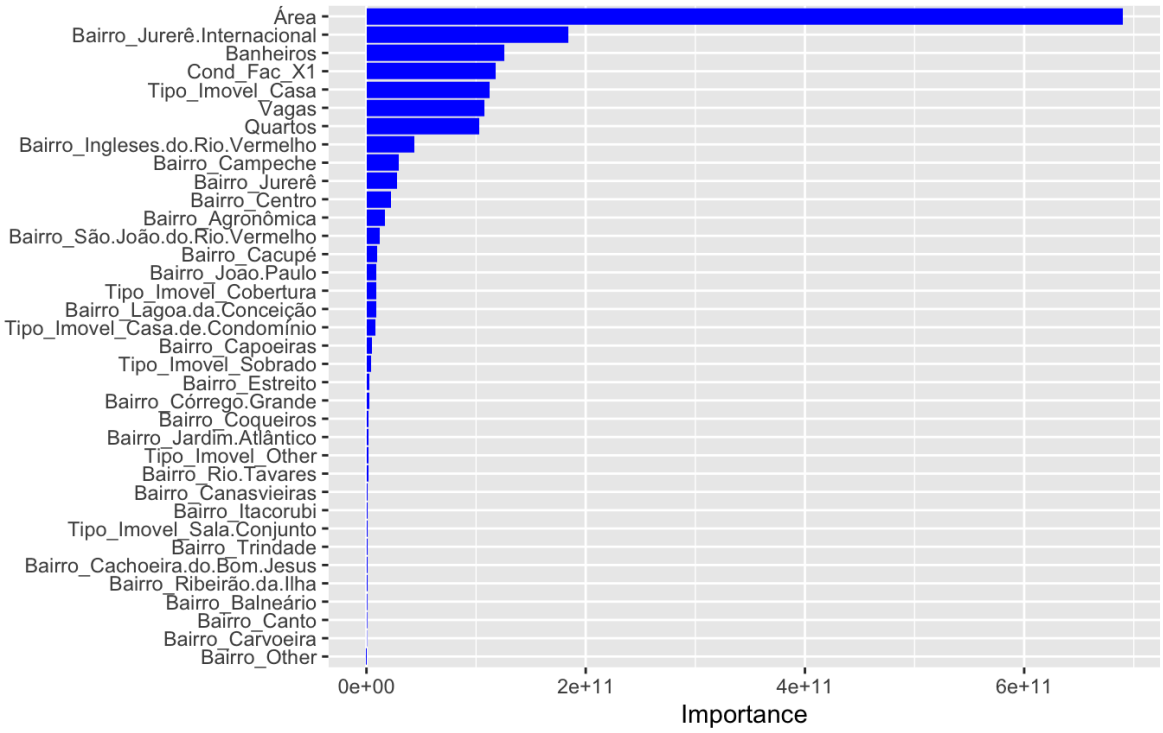


Fonte: Autor (2023).

Nota-se que os modelos de XGBoost e Random Forest desempenham de forma semelhante, mas o modelo Random Forest tem ligeira vantagem como pode-se evidenciar dada a menor dispersão para erros maiores. O gráfico justifica e ilustra o desempenho comparativo da métrica MAPE, ficando mais evidente que o modelo Lasso possui a maior dispersão de erros acumulados dentro todos os algoritmos testados, confirmando que é aquele que possui o pior desempenho.

Conforme mencionado anteriormente, o modelo de Random Forest é aquele que apresenta o melhor desempenho nas quatro métricas avaliadas para a modelagem de preços de imóveis. A fim de compreender também quais são os fatores mais relevantes para a formação do preço de um imóvel, foi elaborado um gráfico que demonstra esses atributos e sua contribuição. A Figura 22 ilustra todos os fatores abordados.

Figura 22. Importância dos atributos para o Random Forest



Fonte: Autor (2023).

Evidencia-se que a área privativa desempenha um papel fundamental na modelagem de preços, o que é esperado, uma vez que muitos indicadores se baseiam nessa variável. Um aspecto relevante do estudo está relacionado à importância dos bairros no modelo, destacando-se o Bairro Jurerê Internacional, que demonstrou ter uma influência maior do que outros atributos, como o número de quartos, banheiros e vagas. Esses fatores estão intrinsecamente ligados a atributos como segurança, qualidade de vida, acessibilidade a hospitais, escolas e outros, que não foram mapeados por bairro, mas estão diretamente relacionados aos valores mais altos ou mais baixos de cada região. No entanto, devido à limitação dos dados que contêm apenas o nome do bairro, não é possível avaliar outras condições que contribuem para a importância de cada um deles.

Na comparação com demais estudos que também possuem como abordagem a modelagem de valores imobiliários com base em variáveis dispoíniveis a respeito dos imóveis o estudo mostrou um distanciamento considerável com as métricas obtidas pelos estudos. No caso do estudo de Zhao (2022) o R2 atingido com a utilização do XGBoost foi de 0.943 e no de Li (2021), o que comparativamente ao modelo testado no presente estudo obteve um resutlado de 0.814, um distanciamento considerável da regressão. Ao analisar o estudo de

Hong (2020) e a utulização do Random Forest, os resultados para o MAPE e R2 foram respectivamente 5,48% e 0,971, enquanto o atingido com o Random Forest Tune no modelo foram respectivamente 24,53% e 0,818.

O distanciamento entre as métricas alcançadas pelos estudos abordados e o presente estudo pode ser explicado por diversos fatores relevantes. Um deles é a quantidade maior de dados analisados pelos estudos comparativos, assim como a quantidade de variáveis consideradas nos três casos abordados para comparação. Esses estudos possuíam um conjunto mais amplo de informações disponíveis para a modelagem de valores imobiliários. Além disso, foram aplicados hiperparâmetros mais avançados para prever os valores, o que resultou em treinamentos mais precisos, permitindo que os algoritmos testassem combinações e quantidades mais significativas de valores, resultando em melhores métricas de erro.

No entanto, é importante ressaltar que a comparação entre esses resultados deve ser contextualizada devido às diferenças nos mercados analisados em relação ao mercado brasileiro. Cada região possui seus próprios valores e características específicas de imóveis, o que deve ser considerado na modelagem de cada valor imobiliário.

5. CONCLUSÃO

O presente estudo teve por finalidade realizar a predição de preços para imóveis na cidade de Florianópolis/SC, através da aplicação e escolha entre três modelos de ML. Para conduzir o trabalho, foram utilizadas as etapas propostas por Wickham (2023) em seu livro R for Data Science. Essas etapas estão descritas detalhadamente dentro do tópico 3 deste documento.

A etapa inicial de coleta e extração de dados obteve um resultado satisfatório com a utilização da ferramenta Data Miner, retornando 10800 observações divididas entre 9 colunas distintas de um total disponível de aproximadamente 68.000 observações existentes para imóveis à venda na cidade de Florianópolis/SC no mês de Abril/2023. No entanto, dada a limitação da ferramenta, muitas informações disponíveis no portal sobre os imóveis não foram possíveis de serem utilizadas dentro do estudo, uma vez que a ferramenta não conseguiu capturar o link de acesso das mesmas.

Durante a etapa de organização e transformação de dados o objetivo principal o objetivo era tratar as variáveis para que as mesmas fossem representativas e pudessem ser utilizadas dentro dos modelos de ML. A validação da base de dados veio através da comparação dos Índices FIPEZAP+ (2023) de vendas para o mês de Abril/2023. Essa comparação elucidou a paridade existente entre os indicadores testados. Portanto, o resultado da aplicação de diferentes ferramentas no conjunto de dados foi satisfatório e garantiu que a amostra de 6.734 observações pudesse representar de maneira representativa o conjunto de imóveis à venda da cidade.

Os algoritmos de ML utilizados no presente trabalho também obtiveram um resultado positivo. Destaque para o modelo de *Random Forest* com hiperparâmetros otimizados, que apesar de apresentar um desempenho semelhante ao XGBoost, ainda foi capaz de retornar os melhores indicadores nas 4 métricas avaliadas (RMSE, RSR, R2, MAPE) respectivamente: 5,16E+05, 0,441, 0,818 e 25,84%. Nesse caso, a otimização de hiperparâmetros foi essencial para o desempenho do algoritmo, resultando em uma redução do RMSE em 24,53%.

Cabe ressaltar que apesar do modelo *Random Forest* retornar um bom valor para a predição de preço de venda é preciso compreender que o mercado imobiliário possui diversas particularidades que influenciam no preço de um imóvel. A ampla gama de possibilidades existentes para os imóveis dificultam a generalização de uma única modelagem com as variáveis que estavam disponíveis para a realização do presente trabalho.

Outro ponto de extrema relevância é que tanto os resultados em comparação com o Índice FIPEZAP+ (2023) quanto os resultados dos três modelos de aprendizado de máquina demonstraram uma tendência de subvalorização dos imóveis em todas as métricas analisadas. Essa tendência pode ser potencialmente explicada pela inacessibilidade ao conjunto completo de anúncios de imóveis. Como foi realizado um web scraping sem uma ordenação prévia, os imóveis selecionados podem estar, em média, abaixo dos preços efetivamente praticados, o que potencialmente contribui para a subvalorização observada.

Portanto, como sugestão para próximos trabalhos propõe-se a utilização de outras variáveis que possam contabilizar com maior precisão a geolocalização (baseadas em coordenadas) interpretando a proximidade a locais de interesse público (centros de saúde, centros comerciais, áreas de lazer, etc.), particularidades do imóvel e suas características (novo, antigo, bem conversado ou não, etc.) e outras variáveis que possam auxiliar na identificação, parametrização e previsão de preços para imóveis. Outra sugestão pertinente é a combinação de modelos de ML que podem ser capazes de retornar bons valores sem a presença de *overfitting* durante a seleção da importância das variáveis. Outro tópico de interesse para futuras pesquisas refere-se à coleta de dados do mercado imobiliário. Como o acesso aos dados é de suma importância para a realização de um estudo de ML, recomenda-se o desenvolvimento de um algoritmo de *web scraping* mais eficiente, que possa ser aplicado em outras localidades, além da cidade abordada no decorrer do estudo. Dessa forma, seria possível coletar mais informações e complementar o modelo de ML, aumentando a representatividade do banco de dados e, consequentemente, a acurácia atingida com as previsões.

REFERÊNCIAS

ALVARENGA-JÚNIOR, W. J. de. **Métodos De Otimização Hiperparamétrica: Um Estudo Comparativo Utilizando Árvores De Decisão E Florestas Aleatórias Na Classificação Binária**. 2018. 82 f. Dissertação (Mestrado) - Curso de Engenharia Elétrica, Escola de Engenharia, Ufmg, Belo Horizonte, 2018. Cap. 27.

BADILLO, S.; BANFAI, B.; BIRZELE, F.DAVYDOV, I. I.; HUTCHINSON, L.; KAM-THONG, T.; SIEBOURG-POLSTER, J.; STEIERT, B.; ZHANG, J. D. An Introduction to Machine Learning. **Clinical Pharmacology & Therapeutics**, v. 107. 2020. Wiley. http://dx.doi.org/10.1002/cpt.1796

BALDOMINOS, A.; BLANCO, I.; MORENO, A.; ITURRARTE, R.; BERNÁRDEZ, Ó.; AFONSO, C. Identifying Real Estate Opportunities Using Machine Learning. **Applied Sciences**, v. 8. 2018.

BERRAR, D. Cross-Validation. Encyclopedia Of Bioinformatics And Computational **Biology**, v. 1. 2023.

BIAU, G. et al. A random forest guided tour. **Test**, v. 25. 2016.

BREIMAN, L. Random forests. **Machine learning**, v. 45, n. 1, p. 5-32, 2001.

BLUEPRINT. **Entendendo a participação da construção civil no PIB brasileiro ao longo dos anos**. 2022. Disponível em: https://blueprint.apto.vc/entendendo-a-participacao-da-construcao-civil-no-pib-brasileiro-ao-longo-dos-anos. Acesso em: 24 maio 2023.

BORDE, S. et al. Real Estate Investment Advising Using Machine Learning. International **Research Journal Of Engineering And Technology (Irjet), Mumbai**, v. 04. 2017.

CBIC. **Indicadores Imobiliários Nacionais**: 1º trimestre de 2023. 2023. Disponível em: https://cbic.org.br/wp-content/uploads/2023

CHEN, T.; GUESTRIN, C**. XGBoost.** Proceedings Of The 22Nd Acm Sigkdd International Conference On Knowledge Discovery And Data Mining. 2016.

CHIRAG-GOYAL. **Bagging- 25 Questions to Test Your Skills on Random Forest Algorithm.** 2021. Disponível em: https://www.analyticsvidhya.com/blog/2021/05/bagging-25-questions-to-test-your-skills-on-random-forest-algorithm/. Acesso em: 04 jun. 2023.

CHICCO, D.; WARRENS, M. J.; JURMAN, G. The coefficient of determination R-squared is more informative than SMAPE, MAE, MAPE, MSE and RMSE in regression analysis evaluation. **Peerj Computer Science**, v. 7. 2021.

CHOU, J., S.; FLESHMAN, D., B.; TRUONG, D. N. Comparison of machine learning models to provide preliminary forecasts of real estate prices. **Journal Of Housing And The Built Environment**, v. 37. 2022.

DIN, A. et al. Environmental Variables and Real Estate Prices. **Urban Studies**, v. 38. 2000,
PROBST, P.; BISCHL, B.; BOULESTEIX, A., L. Tunability: importance of hyperparameters of machine learning algorithms. **Journal Of Machine Learning Research** v. 1.

DONG, G. Feature Engineering for Machine Learning and Data Analytics. **Boca Raton: Taylor & Francis Group**, 2018.

FIPEZAP+ (São Paulo). **Relatório Índice Residencial Informe Fevereiro 2023**. 2023. Disponível em: https://fipezap.zapimoveis.com.br/wp-content/uploads/2022/03/FIPEZAPVenda_202203.pdf. Acesso em: 25 abr. 2023.

FIPEZAP. FIPEZAP. 2023. Disponível em: https://fipezap.zapimoveis.com.br/. Acesso em: 21 abr. 2023.

FONTI, V.; BELITSER, E. Feature selection using lasso. **VU Amsterdam research paper in business analytics,** v. 30, p. 1-25, 2017.

GALDINO, I. M.; GALLINDO, E. de L.; MOREIRA, M. W. L. Utilização de Bots para Obtenção Automática de Dados Públicos usando as Técnicas de Web Crawling e Web Scraping. **Anais do Workshop de Computação Aplicada em Governo Eletrônico** (Wcge 2020), v. 1. 2020. http://dx.doi.org/10.5753/wcge.2020.11269.

GOLMOHAMMADI, G.; PRASHER, S.; MADANI, A.; RUDRA, R. Evaluating Three Hydrological Distributed Watershed Models: mike-she, apex, swat. **Hydrology**, v. 1. 2014.

HAIR JUNIOR, J.F. Partial Least Squares Structural Equation Modeling (PLS-SEM) Using R: a workbook. 2. ed. Suíça: **Springer Nature Switzerland Ag**, 2021.

HOMMA, E. J.; BONFIM, M. A.; ALMEIDA, R. F. F.; SILVA, A. O.; RIGONATI, G. C.; RAMINELLI, D. G. T. L.; SANTOS, B. S.; LIMA, R. H. P.. Aplicações de machine learning nas áreas da engenharia de produção. **Revista Brasileira de Administração Científica**, v.13, n.1, p.368-384, 2022.

HONG, Jengei; CHOI, Heeyoul; KIM, Woo-Sung. A HOUSE PRICE VALUATION BASED ON THE RANDOM FOREST APPROACH: the mass appraisal of residential property in south korea. **International Journal Of Strategic Property Management**, [S.L.], v. 24, n. 3, p. 140-152, 3 fev. 2020. Vilnius Gediminas Technical University. http://dx.doi.org/10.3846/ijspm.2020.11544.

IBGE. **Características gerais dos domicílios e dos moradores 2019**. 2020. Disponível em: https://biblioteca.ibge.gov.br/visualizacao/livros/liv101707_informativo.pdf. Acesso em: 01 jun. 2023.

INDEED EDITORIAL TEAM. **15 Web Scraping Tools** (Plus Applications and Purpose). 2023. Disponível em: https://www.indeed.com/career-advice/career-development/web-scraping-tools. Acesso em: 25 maio 2023.

JIANG, Tammy et al. Supervised Machine Learning: A Brief Primer. **Behaviour Therapy**, v. 51. 2020.

KREMER, J. **Mercado Imobiliário**. Inadaial: Uniasselvi, 2008. 155 p.

KROTOV, V.; TENNYSON, M. SCRAPING Financial Data from the Web Using the R Language. **Journal of Emerging Technologies in Accounting**. 2018

LI, S. et al. Understanding the Effects of Influential Factors on Housing Prices by Combining Extreme Gradient Boosting and a Hedonic Price Model. **Land**, v. 10. 2021.

LORENZ, F.; WILLWERSCH, J.; CAJIAS, M.; FUERST, F. Interpretable machine learning for real estate market analysis. **Real Estate Economics**, v. 1. 2022.

MARTINS, M. E. G. Coeficiente de determinação. **Revista de Ciência Elementar**, v. 6. 2018.

MATOS, D.; BARTKIW, P. I. N. **Introdução ao Mercado Imobiliário**. Curitiba: Instituto Federal do Paraná, 2011. 122 p. Disponível em: https://encurtador.com.br/kDM37. Acesso em: 24 maio 2023.

NAQA, I. E.; MURPHY, M. J. What Is Machine Learning? Machine Learning In Radiation **Oncology**, v. 2, n. 1, p. 3-11, 2015.

NAIR, V. G. Getting Started with Beautiful Soup. **Birmingham**: Packt Publishing, 2014.

NEPOMUCEN, T**. Basic Web Scraping**. 2016. Disponível em: https://neps.academy/br/course/web-scraping/lesson/introducao. Acesso em: 25 maio 2023.

OZDEMIR, S. Feature Engineering Bookcamp. **Shelter Island, Ny**: Manning Publications Co., 2022.

PINA, D. B. et al. Análise de Hiperparâmetros em Aplicações de Aprendizado Profundo por meio de Dados de Proveniência. **Anais do Simpósio Brasileiro de Banco de Dados** (Sbbd), v. 1. 2019.

RANSTAM, J. et al. LASSO regression. **British Journal Of Surgery**, v. 105. 2018.

RASCHKA, S. et al. **Python Machine Learning:** machine learning and deep learning wiht python, scikit-learn, and tensorflow 2. 3. ed. Birmingham: Packt Publishing Ltd., 2019.

REIS, E. A.; REIS, I. A. **Análise Descritiva de Dad**os. Minas Gerais: Departamento de Estatística da Ufmg, 2002. 64 p. Disponível em: https://www.est.ufmg.br/portal/arquivos/rts/rte0202.pdf. Acesso em: 02 jun. 2023.

ROLLI, C. S. **Zillow Home Value Prediction (Zestimate) By Using XGBoost**. 2020. 47 f. Dissertação (Mestrado) - Curso de Computer Science, Computer Science & Information Systems, California State University, San Marcos, San Marcos, 2020.

SANTOS, R. V. dos. Planejamento do preço de venda**. Caderno de Estudos**. 1997.

SANTOS-JUNIOR, E. S., MACHADO, A. de O., MACEDO, M. C. F.; dos SANTOS SOUZA, A. C. Reinforcement Learning para treino do Pac-Man em Speedrun. **Brazilian Journal of Development**, v. 5. 2019

SCACCIA, K. **Validação Cruzada Aninhada com Scikit-learn**. 2022. Disponível em: https://dataml.com.br/validacao-cruzada-aninhada-com-scikit-learn/. Acesso em: 26 maio 2023.

SILVESTRE, A. L. **Análise de dados e estatística descritiva**. Escolar Editora, 2007.

SOETEWEY, A. **Web scraping in R**. 2023. Disponível em: https://statsandr.com/blog/web-scraping-in-r/. Acesso em: 25 maio 2023.

SOUZA, H. B. et al. **Uma aplicação na base de dados de artigos de periódicos científicos das áreas de informação**. 2008. 155 f. Tese (Doutorado) - Curso de Engenharia Elétrica, Universidade de Brasília, Brasília, 2008.

TASKAN, B.; JUNÇA-SILVA, A.; CAETANO, A. Clarifying the conceptual map of VUCA: a systematic review. **International Journal Of Organizational Analysis**, v. 30. 2022.

TIBSHIRANI, R. Regression Shrinkage and Selection Via the Lasso**. Journal Of The Royal Statistical Society**: Series B (Methodological), [S.L.], v. 58. 1996.

VERZANI, J. **Getting Started with Rstudio**. Sebastopol: Safari, 2011. Disponível em: https://books.google.com.br/books?id=q95Nyozjda4C&printsec=frontcover&hl=pt-BR#v=onepage&q&f=false. Acesso em: 14 maio 2023.

ZHOU, Z. H. Machine Learning. Nanjing: **Springer Nature Singapore**, 2016.

ZHANG, P. Model Selection Via Multifold Cross Validation. **The Annals Of Statistics**, v. 21. 1993.

ZHAO, Y. et al. PATE: property, amenities, traffic and emotions coming together for real estate price prediction. **The University Of Hong Kong**. 2022.

WICKHAM, H.; GROLEMUND, G. R for **Data Science**. 2017. http://r4ds.had.co.nz/.

WICKHAM, H. **R for Data Science**. Sebastopol. 2023.

WU, J. et al. Hyperparameter Optimization for Machine Learning Models Based on Bayesian Optimization. **Journal Of Electronic Science And Technology**, v. 17. 2019.

ANEXO 1. CÓDIGO COMPLETO

````
---
title: "Compilado código TCC"
author: "Lucca Magri Zaghi"
date: "2023-05-18"
output: html_document
---

#Carregar pacotes

{r setup, include=FALSE, echo=FALSE}
library("dplyr")
library("tidyverse")
library("readr")
library("ggplot2")
library("rmarkdown")
library("forcats")
library("xts")
library("PerformanceAnalytics")
library("broom")
library("caret")
library("readxl")
library("tidyr")
library("ranger")
library("glmnet")
library("xgboost")
````

#Limpeza de dados

```{r setup, include=FALSE, echo=FALSE}

#Carregar os dados .csv de alugueis de florianópolis
library(readr)
#juntar os df
t1 <- read_csv("~/Downloads/Vendas_1.csv")
t2 <- read_csv("~/Downloads/Vendas_2.csv")

vendas_florianopolis_original <-  rbind(t1, t2)
names(vendas_florianopolis_original) <- c("Área", "Quartos", "Banheiros", "Vagas", "Preço",
"Descrição", "Condomínio", "Endereço", "Link")

head(vendas_florianopolis_original, ncol = Inf)
glimpse(vendas_florianopolis_original)
skim(vendas_florianopolis_original)
#Carregar pacotes para tratar dados
library(tidyr)
library(dplyr)
```

```
#transformar o que é char em numeric e rertirar eventualmente erros de digitação que existem, linhas duplicadas e filtrar por valores menores que 10KK e 11 banheiros
vendas_florianopolis_tratado <- vendas_florianopolis_original %>%
  mutate(Área = as.numeric(gsub("[^[:digit:].]", "", Área)),
         Quartos = as.numeric(gsub("[^[:digit:].]", "", Quartos)),
         Vagas = as.numeric(gsub("[^[:digit:].]", "", Vagas)),
         Banheiros = as.numeric(gsub("[^[:digit:].]", "", Banheiros)),
         Preço = as.numeric(gsub("[^[:digit:]]", "", gsub("\\.", "", Preço))),
         Condomínio = as.numeric(gsub("[^[:digit:]]", "", gsub("\\.", "", Condomínio)))) %>%
  subset(Banheiros <=11 & Preço <=10000000) %>%
  distinct(Preço, Endereço, Descrição, .keep_all = TRUE)
str(vendas_florianopolis_tratado)


#Identificando o tipo de imóvel com base no que o Vivareal usa e nas descrições existentes
vendas_florianopolis_tratado <- vendas_florianopolis_tratado %>%
  mutate(Tipo_Imovel = case_when(
    grepl("(?i)Apartamento", Descrição) ~ "Apartamento",
    grepl("(?i)Casa de Condomínio", Descrição) ~ "Casa de Condomínio",
    grepl("(?i)Casa", Descrição) & !grepl("(?i)Casa de Condomínio", Descrição) ~ "Casa",
    grepl("(?i)Chácara", Descrição) ~ "Chácara",
    grepl("(?i)Cobertura", Descrição) ~ "Cobertura",
    grepl("(?i)Flat", Descrição) ~ "Flat",
    grepl("(?i)Kitnet/Conjugado", Descrição) ~ "Kitnet/Conjugado",
    grepl("(?i)Lote/Terreno", Descrição) & !grepl("(?i)Imóvel Comercial|Ponto
Comercial/Loja/Box", Descrição) ~ "Lote/Terreno",
    grepl("(?i)Sobrado", Descrição) ~ "Sobrado",
    grepl("(?i)Edifício Residencial", Descrição) ~ "Edifício Residencial",
    grepl("(?i)Fazenda/Sítios/Chácaras", Descrição) ~ "Fazenda/Sítios/Chácaras",
    grepl("(?i)Consultório", Descrição) ~ "Consultório",
    grepl("(?i)Galpão/Depósito/Armazém", Descrição) ~ "Galpão/Depósito/Armazém",
    grepl("(?i)Imóvel Comercial", Descrição) & !grepl("(?i)Lote/Terreno", Descrição) ~
"Imóvel Comercial",
    grepl("(?i)Ponto Comercial/Loja/Box", Descrição) & !grepl("(?i)Lote/Terreno", Descrição)
~ "Ponto Comercial/Loja/Box",
    grepl("(?i)Sala/Conjunto", Descrição) ~ "Sala/Conjunto",
    grepl("(?i)Prédio/Edifício Inteiro", Descrição) ~ "Prédio/Edifício Inteiro",
    TRUE ~ NA_character_
  ))
vendas_florianopolis_tratado <- vendas_florianopolis_tratado %>%
  filter(!is.na(Tipo_Imovel))

#Identificando os bairros conforme o vivareal e a tabela de bairros que peguei do wikipedia
tipos_imoveis <- c("Centro", "Capoeiras", "Trindade", "Agronômica", "Saco dos Limões",
"Coqueiros", "Monte Cristo", "Jardim Atlântico", "Itacorubi", "Costeira do Pirajubaé",
"Capivari", "Tapera da Base", "Estreito", "Monte Verde", "Balneário", "São João do Rio
Vermelho", "Canto", "Abraão", "Santa Mônica", "Lagoa da Conceição", "Saco Grande",
"Córrego Grande", "Canasvieiras", "Pantanal", "Coloninha", "Barra da Lagoa", "Carianos",
"José Mendes", "Ingleses Centro", "João Paulo", "Campeche Leste", "Campeche Sul", "Rio
```

```
Tavares Central", "Santinho", "Ponta das Canas", "Vargem do Bom Jesus", "Armação",
"Cachoeira do Bom Jesus Leste", "Pântano do Sul", "Itaguaçu", "Jurerê Leste", "Campeche
Norte", "Vargem Grande", "Campeche", "Ressacada", "Morro das Pedras", "Alto Ribeirão
Leste", "Alto Ribeirão", "Ribeirão da Ilha", "Santo Antônio", "Sambaqui", "Ingleses Sul",
"Bom Abrigo", "Jurerê Internacional","Jurerê", "Porto da Lagoa", "Cachoeira do Bom Jesus",
"Rio Tavares do Norte", "Pedregal", "Ratones", "Canto da Lagoa", "Retiro", "Cacupé",
"Lagoa Pequena", "Barra do Sambaqui", "Caiacanga", "Lagoinha do Norte", "Base Aérea",
"Pedrita", "Açores", "Costeira do Ribeirão", "Moenda", "Tapera", "Daniela", "Vargem
Pequena", "Canto dos Araçás", "Recanto dos Açores", "Canto do Lamim", "Vargem de Fora",
"Dunas da Lagoa", "Autódromo", "Forte", "Ingleses Norte", "Caieira", "Praia Brava",
"Ingleses do Rio Vermelho", "Canasvieiras", "Joao Paulo", "Carvoeira", "Beira Mar", "Rio
Tavares", "Parque São Jorge", "Praia dos Ingleses", "Canajure", "Barreiros", "Campinas",
"Kobrasol", "Praia Mole")

#identificando todas as linhas para verificar qual o bairro ele se encaixa
vendas_florianopolis_tratado$Bairro <- NA
for (i in 1:nrow(vendas_florianopolis_tratado)) {
  endereco <- vendas_florianopolis_tratado$Endereço[i]
  for (bairro in tipos_imoveis) {
    if (grepl(bairro, endereco, ignore.case = TRUE, fixed=FALSE)) {
      vendas_florianopolis_tratado$Bairro[i] <- bairro
      break
    }
  }
}

#removendo as linhas que não seja possível identificar o modelo

vendas_florianopolis_tratado <- subset(vendas_florianopolis_tratado, !is.na(Bairro))
dadossemna <- vendas_florianopolis_tratado

dadossemna %>%
  filter(Tipo_Imovel == "Casa") %>%
  select(Condomínio, Link, Tipo_Imovel) %>%
  filter(!is.na(Condomínio))

dados <- dadossemna %>%
  mutate(Tipo_Imovel = ifelse(!is.na(Condomínio) & Tipo_Imovel == "Casa", "Sobrado",
Tipo_Imovel)) %>%
  mutate(Condomínio = ifelse(Condomínio == 1, NA, Condomínio))

dados %>%
  filter(Tipo_Imovel == "Sobrado") %>%
  select(Condomínio, Link, Tipo_Imovel)
dados %>%
  group_by(Tipo_Imovel) %>%
  summarise(ok = sum(!is.na(Condomínio)),
            na = n()-sum(!is.na(Condomínio)),
            tot = ok+na,
            prop = 100*na/tot) %>%
```

```
  arrange(-tot)

semcond <- c("Casa","Sala/Conjunto" , "Ponto Comercial/Loja/Box","Chácara")
dados_df <- dados %>%
  mutate(Cond_Fac = ifelse(Tipo_Imovel %in% semcond, 0, 1)) %>%
  mutate(Cond_Fac = as_factor(Cond_Fac)) %>%
  select(-Condomínio, -media_preco, -desvio_padrao_preco, -Link, -Descrição, -
Endereço)%>%
  mutate_if(is.character, factor)


##Análise de dados

{r setup, include=FALSE, echo=FALSE}

#Skim
tabela1<-skim(dados)
tabela1

#Histograma 1
dados %>%
  count(Bairro) %>%
  arrange(desc(n)) %>%
  head(30) %>%
  mutate(Percent = n / sum(n) * 100) %>%
  ggplot(aes(x = reorder(Bairro, -n), y = n)) +
  geom_col(fill = "red") +
  geom_text(aes(label = paste0(round(Percent, 2), "%")),
            position = position_stack(vjust = 1.05),
            color = "black",
            size = 1.8) +
  xlab("Bairro") +
  ylab("Frequência") +
  ggtitle("30 Bairros com Maior Frequência") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  theme(legend.position = "none")

#Histograma 2
dados %>%
  count(Tipo_Imovel) %>%
  arrange(n) %>%
  mutate(Tipo_Imovel = factor(Tipo_Imovel, levels = Tipo_Imovel)) %>%
  mutate(Percent = n / sum(n) * 100) %>%
  ggplot(aes(x = Tipo_Imovel, y = n)) +
  geom_col(fill = "red") +
  geom_text(aes(label = scales::percent(Percent / 100), y = n),
            vjust = 0.1,
            color = "black",
            size = 1.8) +
  coord_flip() +
```

```
  ylab("Frequência") +
  xlab("Tipo de Imóvel") +
  ggtitle("Frequência tipos de imóvel") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))

#Análise de concentração de dados de cada uma das colunas
plot_area <- ggplot(dados_df, aes(x = Área)) +
  geom_histogram(fill = "blue", color = "black", binwidth = 10, bins = 25) +
  labs(x = "", y = "") +
  ggtitle("Área") +
  xlim(0, 1000)

plot_quartos <- ggplot(dados_df, aes(x = Quartos)) +
  geom_histogram(fill = "green", color = "black", binwidth = 1, bins = 1) +
  labs(x = "", y = "") +
  ggtitle("Quartos") +
  xlim(0, 10)

plot_banheiros <- ggplot(dados_df, aes(x = Banheiros)) +
  geom_histogram(fill = "red", color = "black", binwidth = 1, bins = 1) +
  labs(x = "", y = "") +
  ggtitle("Banheiros") +
  xlim(0, 10)

plot_vagas <- ggplot(dados_df, aes(x = Vagas)) +
  geom_histogram(fill = "orange", color = "black", binwidth = 1, bins = 1) +
  labs(x = "", y = "") +
  ggtitle("Vagas") +
  xlim(0, 10)

plot_preço <- ggplot(dados_df, aes(x = Preço)) +
  geom_histogram(fill = "purple", color = "black", binwidth = 25, bins = 25) +
  labs(x = "", y = "") +
  ggtitle("Preço") +
  xlim(0, 10000000) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))

# Combinação dos histogramas usando plot_grid
combined_plot <- plot_grid(plot_area, plot_quartos, plot_banheiros, plot_vagas, plot_preço)

# Exibição da imagem combinada
combined_plot

#Sumarizado de colunas

##Organização de dados

{r setup, include=FALSE, echo=FALSE}
set.seed(1212)
```

```
split <- initial_split(dados_df, prop = 0.75)
train <- training(split)
test <- testing(split)
set <- validation_split(train)

desvio_padrao_train<- sd(train$Preço)
desvio_padrao_test<- sd(test$Preço)
desvio_padrao_dados<- sd(dados_df$Preço)

#Folds para validação cruzada

fold <- vfold_cv(train)

#Criando o workflow para o modelo

rec <- recipe(Preço ~ ., data = train) %>%
  step_dummy(all_nominal(), -all_outcomes()) %>%
  step_impute_knn(Quartos, Vagas)
wf <- workflow() %>%
  add_recipe(rec)



###Algoritmos sem otimização (Consrtrução + Treinamento)

{r setup, include=FALSE, echo=FALSE}

#Definição de parametros iniciais
rf_spec <- rand_forest(trees=500, mtry = 2, min_n = 2) %>%
  set_engine("ranger") %>%
  set_mode("regression")

lasso_spec <- linear_reg(penalty = 1, mixture = 1) %>%
  set_engine("glmnet") %>%
  set_mode("regression")

xgboost_spec <- boost_tree(trees = 500, mtry = 2, min_n = 2, learn_rate = 0.1, tree_depth = 3)
%>%
 set_engine("xgboost") %>%
  set_mode("regression")

#Treinamento inicial para o modelo de acordo com o paralel
doParallel::registerDoParallel()

rf <- wf %>%
  add_model(rf_spec) %>%
  fit_resamples(resamples=fold, control=control_resamples(save_pred=TRUE))

xgboost <- wf %>%
  add_model(xgboost_spec) %>%
```

```
  fit_resamples(resamples=fold,
                control=control_resamples(save_pred=TRUE))

lasso <- wf %>%
  add_model(lasso_spec) %>%
  fit_resamples(resamples = fold, control = control_resamples(save_pred = TRUE))

#Verificação de resultados para o treinamento
metrics_rf <- collect_metrics(rf)%>%
  mutate(Model = "Random Forest")

metrics_xgb <- collect_metrics(xgboost) %>%
  mutate(Model = "XGBoost")

metrics_lasso <- collect_metrics(lasso) %>%
  mutate(Model = "Lasso")

all_metrics <- bind_rows(metrics_rf, metrics_xgb, metrics_lasso)
all_metrics <- arrange(all_metrics, mean)
all_metrics

#Visualização da dispersão de cada Fold (10) treinadas
folds_rf <- collect_predictions(rf) %>%
  ggplot(aes(Preço, .pred, color = id)) +
  geom_abline(lty = 2, color = "red") +
  geom_point(alpha = 0.1) +
  facet_wrap(~ id) +
  labs(title = "Random Forest") +
  theme_linedraw() +
  theme(legend.position = "none", axis.text.x = element_blank()) +
  coord_equal(ratio = 1)

folds_xgboost <- collect_predictions(xgboost) %>%
  ggplot(aes(Preço, .pred, color = id)) +
  geom_abline(lty = 2, color = "red") +
  geom_point(alpha = 0.1) +
  facet_wrap(~ id) +
  labs(title = "XGBoost") +
  theme_linedraw() +
  theme(legend.position = "none", axis.text.x = element_blank()) +
  coord_equal(ratio = 1)

folds_lasso <- collect_predictions(lasso) %>%
  ggplot(aes(Preço, .pred, color = id)) +
  geom_abline(lty = 2, color = "red") +
  geom_point(alpha = 0.1) +
  facet_wrap(~ id) +
  labs(title = "Lasso") +
  theme_linedraw() +
   theme(legend.position = "none", axis.text.y = element_blank())
```

```
  coord_equal(ratio = 1)

  folds_lasso
  folds_rf
  folds_xgboost

#Plot do melhor fold para cada modelo
library(cowplot)

plot_rf <- collect_predictions(rf)%>%
  ggplot(aes(Preço, .pred)) +
  geom_abline(lty = 2, color = "gray50") +
  geom_point(alpha = 0.1, color = "#e32d91") +
  theme(legend.position = "none", axis.text.x = element_blank())+
  labs(title = "Random Forest") +
  ylim(0, 5e+06) +
  theme_test()

plot_xgboost <- collect_predictions(xgboost)%>%
  ggplot(aes(Preço, .pred)) +
  geom_abline(lty = 2, color = "gray50") +
  geom_point(alpha = 0.1, color = "#0072B2") +
  labs(title = "Xgboost") +
   theme(legend.position = "none", axis.text.x = element_blank())+
  ylim(0, 5e+06)+
  theme_test()

plot_lasso <- collect_predictions(lasso)%>%
  ggplot(aes(Preço, .pred)) +
  geom_abline(lty = 12, color = "gray50") +
  geom_point(alpha = 0.1, color = "#D55E00") +
  theme(legend.position = "none", axis.text.x = element_blank())+
  labs(title = "Lasso" )+
  ylim(0, 5e+06)+
  theme_test()

plot_grid(plot_rf, plot_xgboost, plot_lasso, nrow = 1, align = "h")


###Algoritmos com otimização (Construção + Treinamento)

{r setup, include=FALSE, echo=FALSE}

#realizar o tune e seleçãõ de parametros para serem tunados
tune_spec_rand <-
  rand_forest(
    mtry = tune(),
    trees = 1000,
    min_n = tune()
  ) %>%
```

```
  set_engine("ranger") %>%
  set_mode("regression")

tune_spec_xgb <- boost_tree(
    mtry = tune(),
    trees = 1000,
    min_n = tune(), tree_depth = tune()
  ) %>%
  set_engine("xgboost") %>%
  set_mode("regression")

tune_spec_lasso <- linear_reg(penalty = tune(), mixture = 0.75) %>%
  set_engine("glmnet")

#Confecção de modelos de grid (range de cada parametro)
random_grid <- grid_regular(mtry(range = c(1,12)),
                      min_n(range = c(1,12)),
                      levels = 4)

xgb_grid <- grid_regular(mtry(range = c(1,12)),
                      min_n(range = c(1,12)), tree_depth(range = c(1,6)),
                      levels = 4)

lasso_grid <- grid_regular(penalty(), levels = 100)

#Treinamento dos modelos para otimização
doParallel::registerDoParallel()
tree_tune_rand <- wf %>%
  add_model(tune_spec_rand) %>%
  tune_grid(resamples = fold,
            grid = random_grid)

tree_tune_xgb <- wf %>%
  add_model(tune_spec_xgb) %>%
  tune_grid(resamples = fold,
            grid = xgb_grid)

tree_tune_lasso <- tune_grid(
  wf %>% add_model(tune_spec_lasso),
  resamples = fold,
  grid = lasso_grid
)

#selecionando o melhor modelo de acordo com o RMSE
best_tree_rand <- tree_tune_rand %>%
  select_best(metric = "rmse")

best_tree_xgb <- tree_tune_xgb %>%
  select_best(metric = "rmse")
```

```
best_tree_lasso <- tree_tune_lasso %>%
  select_best(metric = "rmse")

#Visualizar os melhores parametros para os menores RMSE
best_tree_rand
best_tree_xgb
best_tree_lasso

#Adicionando todo o modelo dentro da análise
final_tree_wf_rand <- wf %>%
  add_model(tune_spec_rand) %>%
  finalize_workflow(best_tree_rand)

final_tree_wf_xgb <- wf %>%
  add_model(tune_spec_xgb) %>%
  finalize_workflow(best_tree_xgb)

final_tree_wf_lasso <- wf %>%
  add_model(tune_spec_lasso) %>%
  finalize_workflow(best_tree_lasso)

#Coletando resultados
final_tree_fit_rand <- final_tree_wf_rand %>%
  last_fit(split)

results_rand_tune <- final_tree_fit_rand %>%
  collect_metrics(summarise = FALSE) %>%
  mutate(Model = "Random Forest Tune")

rmse_rand <- final_tree_fit_rand %>%
  collect_metrics() %>%
  filter(.metric == "rmse") %>%
  pull(.estimate)

tabela_rand <-collect_predictions(final_tree_fit_rand)

final_tree_fit_lasso <- final_tree_wf_lasso %>%
  last_fit(split)

results_lasso_tune <- final_tree_fit_lasso %>%
  collect_metrics(summarise = FALSE) %>%
  mutate(Model = "Lasso Tune")

rmse_lasso <- final_tree_fit_lasso %>%
  collect_metrics() %>%
  filter(.metric == "rmse") %>%
  pull(.estimate)

tabela_lasso <-collect_predictions(final_tree_fit_lasso)
```

```
final_tree_fit_xgb <- final_tree_wf_xgb %>%
  last_fit(split)

results_xgb_tune <- final_tree_fit_xgb %>%
  collect_metrics(summarise = FALSE)%>%
  mutate(Model = "XGBoost Tune")

rmse_xgb <- final_tree_fit_xgb %>%
  collect_metrics() %>%
  filter(.metric == "rmse") %>%
  pull(.estimate)

all_metrics_tune <- bind_rows(results_lasso_tune, results_xgb_tune, results_rand_tune)
all_metrics_tune

tabela_xgb <-collect_predictions(final_tree_fit_xgb)




##Comparação de resultados (Tunados)

{r setup, include=FALSE, echo=FALSE}

p2 <- tabela_xgb%>%
  ggplot(aes(Preço, .pred)) +
  geom_abline(lty = 2, color = "gray50") +
  geom_point(alpha = 0.1, color = "#0072B2") +
  labs(title = "Xgboost Tune") +
  ylim(0, 5e+06)+
  theme_test()

p1 <- tabela_rand %>%
  ggplot(aes(Preço, .pred)) +
  geom_abline(lty = 2, color = "gray50") +
  geom_point(alpha = 0.1, color = "#D55E00") +
  labs(title = "Randon Forrest Tune" )+
  ylim(0, 5e+06)+
  theme_test()

p3 <- tabela_lasso %>%
  ggplot(aes(Preço, .pred)) +
  geom_abline(lty = 2, color = "gray50") +
  geom_point(alpha = 0.1, color = "red") +
  labs(title = "Lasso Tune" )+
  ylim(0, 5e+06)+
  theme_test()
```

```
plot_grid(p1, p2, p3, nrow = 1, align = "h")

#Coletando o RSR para cada resultado
rsr_rand <- rmse_rand/desvio_padrao_test
rsr_xgb <- rmse_xgb/desvio_padrao_test
rsr_lasso <- rmse_lasso/desvio_padrao_test


#Plot comparativo dispersão + linha tendência Antes e depois da aplicação

#Lasso
x<- plot_grid(plot_rf,p1, nrow = 1, align = "h")

#XGB
y<- plot_grid(plot_xgboost,p2, nrow = 1, align = "h")

#Rand
z<- plot_grid(plot_lasso,p3, nrow = 1, align = "h")
plot_grid(x,y,z, align = "h", nrow = 3)

library(reshape2)
library(ggplot2)

#Parametrizando mesmo nome de colunas para ccada tabela
names(tabela_rand) <- c("ID", "Preço previsto", "Linha_referencia", "Preço anúncio",
"Configurador")
tabela_rand <- tabela_rand %>%
        select(-c(ID, Configurador))


names(tabela_xgb) = c("ID", "Preço previsto", "Linha_referencia", "Preço anúncio",
"Configurador")
tabela_xgb <- tabela_xgb %>%
          select(-c(ID, Configurador))


names(tabela_lasso) = c("ID", "Preço previsto", "Linha_referencia", "Preço anúncio",
"Configurador")
tabela_lasso <- tabela_lasso %>%
          select(-c(ID, Configurador))

#Criando a variavel erro para todas as tabelas
tabela_rand$Erro_absoluto <- abs(tabela_rand$`Preço anúncio` - tabela_rand$`Preço
previsto`)
tabela_rand$Erro_percentual <- ((tabela_rand$`Preço anúncio` - tabela_rand$`Preço
previsto`) / tabela_rand$`Preço anúncio`) * 100
tabela_rand$Erro_percentual <- round(tabela_rand$Erro_percentual, 1)

tabela_lasso$Erro_absoluto <- abs(tabela_lasso$`Preço anúncio` - tabela_lasso$`Preço
previsto`)
```

```
tabela_lasso$Erro_percentual <- ((tabela_lasso$`Preço anúncio` - tabela_lasso$`Preço previsto`) / tabela_lasso$`Preço anúncio`) * 100
tabela_lasso$Erro_percentual <- round(tabela_lasso$Erro_percentual, 1)


tabela_xgb$Erro_absoluto <- abs(tabela_xgb$`Preço anúncio` - tabela_xgb$`Preço previsto`)
tabela_xgb$Erro_percentual <- ((tabela_xgb$`Preço anúncio` - tabela_xgb$`Preço previsto`) / tabela_xgb$`Preço anúncio`) * 100
tabela_xgb$Erro_percentual <- round(tabela_xgb$Erro_percentual, 1)

library(reshape2)
library(ggplot2)
library(tidyverse)
library(hrbrthemes)
library(viridis)
library(cowplot)

# criar coluna "Modelo" em cada tabela
tabela_rand$Modelo <- "Random Forest"
tabela_xgb$Modelo <- "XGBoost"
tabela_lasso$Modelo <- "Lasso"

# combinar tabelas para obter as comparações
tabela_combinada <- rbind(tabela_rand, tabela_xgb, tabela_lasso)
ggplot(tabela_combinada, aes(x = `Preço anúncio` , y = Erro_percentual, group = Modelo, fill = Modelo, color = Modelo)) +
  geom_point( alpha =0.2) +
  labs(x = "Preço anúncio", y = "Erro percentual") +
scale_y_log10()

library(ggplot2)

#Curvas de densidade por modelo
ggplot(tabela_combinada, aes(x = Erro_percentual, color = Modelo, group = Modelo, fill = Modelo)) +
  geom_density(alpha = 0.15)

# plotar boxplots
ggplot(tabela_combinada, aes(x = Modelo, y = Erro_percentual, fill = Modelo)) +
  geom_boxplot() +
  scale_fill_viridis(discrete = TRUE, alpha=0.9) +
    geom_jitter(color="black", size=0.2, alpha=0.2) +
    theme_ipsum() +
  labs(x = "", y = "Erro percentual")+
  ylim(-350,350)

# Carregar pacotes
library(ggplot2)
library(scales)
```

```
# Plotar a curva de densidade acumulada
acumulado_rand <- ggplot(tabela_rand, aes(x = Erro_percentual)) +
  stat_ecdf(geom = "step", inherit.aes = TRUE) +
  scale_x_continuous("Erro percentual", labels = percent) +
  scale_y_continuous("Proporção acumulada", labels = percent) +
  theme_bw()+
  xlim(-200,200)+
  ggtitle("Random Forest")

acumulado_xgb <- ggplot(tabela_xgb, aes(x = Erro_percentual)) +
  stat_ecdf(geom = "step", inherit.aes = TRUE) +
  scale_x_continuous("Erro percentual", labels = percent) +
  scale_y_continuous("Proporção acumulada", labels = percent) +
  xlim(-200,200)+
  ggtitle("XGboost") +
  theme_bw()

acumulado_lasso <- ggplot(tabela_lasso, aes(x = Erro_percentual)) +
  stat_ecdf(geom = "step", inherit.aes = TRUE) +
  scale_x_continuous("Erro percentual", labels = percent) +
  scale_y_continuous("Proporção acumulada", labels = percent) +
  xlim(-200,200)+
  ggtitle("Lasso")+
  theme_bw()

ggplot(tabela_combinada, aes(x = Erro_percentual, color = Modelo, group = Modelo,)) +
  stat_ecdf(geom = "step", inherit.aes = TRUE) +
  scale_x_continuous("Erro percentual", labels = percent) +
  scale_y_continuous("Proporção acumulada", labels = percent) +
  theme_bw()+
  ggtitle("Curvas acumuladas") +
   xlim(-300,300)

plot_grid(acumulado_rand, acumulado_xgb,acumulado_lasso, ncol = 3)
```