Análise de Preços de Imóveis
Relatório Final
Introdução ao Aprendizado de Máquina - 2024/2

Andrew da Silva Faria[1], **Gustavo Vilares Mariz de Oliveira**[1], **Matheus da Cruz Percine Pinto**[1]

[1] Instituto de Computação – Universidade Federal do Rio de Janeiro
Rio de Janeiro, Brasil.

{andrewsf,gustavovmo,matheuscpp}@dcc.ufrj.br

1. Introdução

A utilização de modelos de aprendizado de máquina na tentativa de realizar previsões e classificação de eventos e diferentes tipos de informações, como previsões de moedas e valores no mercado financeiro, é algo muito relevante no contexto atual. Este trabalho busca desenvolver modelos que utilizam técnicas/métodos de aprendizado de máquina capazes de realizar a previsão dos preços de aluguel de apartamentos nos Estados Unidos, com base em diversos outros dados relativos aos apartamentos e o anúncio de seu aluguel. Com isso, a motivação do estudo é poder manusear a base de dados escolhida, realizando seu processamento e utilizá-la no treinamento de modelos que usam diferentes algoritmos — Regressão Linear e Floresta Aleatória —, podendo compará-los, observando os resultados e analisando qual se adequou melhor para o objetivo proposto.

2. Base de dados

A base de dados selecionada apresenta informações detalhadas de aluguéis de apartamentos. Nela, observa-se a presença de 99.492 instâncias, porém a base de dados não explicita a proveniencia dos dados nem método realizado na coleta, sendo apenas registrado a licença do MIT.

2.1. Conteúdo

Cada instância presente na base de dados possui os seguintes atributos:

- ID: identificador único do anúncio
- Category: categoria em que o anúncio foi publicado
- Title: título do anúncio
- Body: corpo do anúncio
- amenities: comodidades junto ao aluguel do apartamento
- bathrooms: número de banheiros no apartamento
- bedrooms: número de quartos no apartamento
- Currency: moeda de pagamento
- Fee: presença (ou não) de taxa no aluguel
- Has_photo: indica se o apartamento tem foto no anúncio
- Pets_allowed: indica a permissão de animais de estimação
- Price: preço do aluguel
- Price_display: preço mostrado no anúncio
- Square_feet: tamanho do apartamento (em metros quadrados)

- Address: endereço do apartamento
- CityName: cidade onde o apartamento está localizado
- State: estado onde o apartamento está localizado
- Latitude: valor de latitude de onde o apartamento está localizado
- Longitude: valor de longitude de onde o apartamento está localizado
- Source: fonte do anúncio
- Time: tempo desde a criação do anúncio

Para o treinamento dos modelos, é necessário tratar os dados.

3. Pré-processamento

Se tentarmos utilizar a base de dados ”crua”para a construção dos modelos, não será possível sequer treiná-lo adequadamente, visto que é necessário tratar os dados, buscando deixá-los apropriados para sua utilização.

3.1. Retirando Atributos não relevantes

Inicialmente, vamos verificar quais atributos existentes na base de dados não contém informação relevante para o treinamento do modelo. Após analisar os atributos e que tipo de informação eles estão guardando, concluímos em excluir atributos segundo 2 motivos:

1. Atributos que em seu significado não somam nenhuma informação ou temos outro atributo existente que seria seu equivalente. Neste raciocínio, excluimos: ID, Body, Title, Price_display, Address, Source.
2. Atributos que possuem variância igual (ou muito próxima) de zero, indicando que, para aquele atributo, quase todas as instâncias possuem o mesmo valor. Com isso, excluimos: Category, Currency, Fee, Price_type.

3.2. Completude e Redudância dos Dados

Para que seja possível utilizar a base de dados é necessário realizar o tratamento de instâncias que não estiverem corretamente preenchidas. Assim, teremos que cuidar das instâncias que os dados de um determinado atributo estiver ausente e também daquelas que estiverem duplicadas.

3.2.1. Dados Ausentes

Observando os dados restantes referentes à localização (latitude, longitude, estado, cidade), temos algumas instâncias que apresentam o valor Null para essas features. Como a quantidade de instâncias com essas características é pequena (302 instâncias para cityname e state, 25 para latitude e longitude), e não temos como substituir esse valor por outro (não temos uma média ou mediana adequado para esses valores), decidimos excluir essas instâncias.

Para bathrooms e bedrooms, assumimos que os valores ausentes existentes seriam equivalentes a não existência de banheiros ou quartos respectivas instâncias, por isso atribuimos o valor 0.

3.2.2. Instâncias Duplicadas

Importante também verificarmos a existência de instâncias repetidas no meio da base de dados. Desse modo, identificamos a existência de 129 instâncias repetidas, que removemos do dataset.

3.3. Outliers

Na base selecionada, percebe-se a existência de dados discrepantes, sendo necessário sua exclusão, visto que temos modelos que são sensíveis a presença de outliers (como na Regressão Linear). Dessa forma, observando o atributo price, decidimos excluir as instâncias que estavam à mais de 3 desvios padrões de distância da média (ou seja, o valor z é maior ou igual à 3). Como resultado, retiramos 998 instâncias.

3.4. Transformação dos Dados

Algumas colunas apresentam informações que consideramos que seriam relevantes para a construção de modelos, mas que sua tipagem não será adequada no treinamento, como utilizar atributos categóricos no modelo de Regressão Linear. Por isso, é necessário que seja realizada uma transformação da representação desses atributos para valores numéricos.

Para os atributos amenities e pets_allowed decidimos utilizar a Variável Dummy, criando uma nova coluna para comodidade (no caso de amenities) e para carrocho/gato (para pets_allowed). Foram criadas um total de 29 colunas (27 para amenities, 2 para pets_allowed) nesse processo.

Além disso, no atributo has photo, temos o seguinte conjunto de possíveis valores: No, Thumbnail, Yes. Para essa coluna, decidimos usar uma codificação ordinal, visto que interpretamos o ”Yes”como tendo várias fotos, Thumbnail como só havendo uma única foto (como uma ”capa”), e o ”No”como não tendo nenhuma foto. Portanto, ficamos com a mudança: $\{\text{No, Thumbnail, Yes}\} \rightarrow \{0,1,2\}$.

No fim do pré-processamento, obtemos um dataset com um total de 39 colunas e 98.058 instâncias.

4. Métodos Usados

Com a base de dados selecionada, consideramos que o mais adequado seria usá-la para treinar modelos que possam realizar previsões de preços dos imóveis cujas informações estão disponíveis. Dessa forma, foi pensado que seria razoável a utilização dos seguintes métodos para a construção dos modelos: Regressão Linear e Floresta Aleatória.

Para a construção e aplicação dos modelos, foi usado majoritariamente a biblioteca *scikit-learn* na linguagem Python, disponível em um conjunto de Google Colaboratory (Pré-processamento, Regressão Linear, Floresta Aleatória)

4.1. Regressão Linear

A Regressão Linear é uma técnica de análise de dados que prevê o valor de dados desconhecidos usando outro valor de dados relacionado e conhecido. Ele modela matematicamente a variável desconhecida ou dependente e a variável conhecida ou independente como uma equação linear [Amazon Web Services (AWS) 2024].

Nosso problema que envolve a previsão de preços se encaixa no tipo de problemas que a regressão linear pode ser aplicada, e supor que os atributos presentes na base podem ter uma relação linear com o valor de preço de aluguel do apartamento é algo possível, por isso, consideramos que este poderia ser um bom candidato de modelo para utilizar.

4.2. Floresta Aleatória

Floresta Aleatória (random forest) é um algoritmo de aprendizado de máquina que cria uma combinação (ensemble) de árvores de decisão. Por sua simplicidade e flexibilidade, podendo ser utilizado tanto para tarefas de classificação quanto regressão, esse algoritmo acaba sendo um dos mais utilizados.

Dentre os motivos da escolha desse algoritmo, podemos citar sua característica de poder ser usado para regressão, que encaixa com o problema de previsão que está sendo trabalhado nesse estudo. Além disso, a floresta aleatória nos permite uma maior interpretabilidade dos resultados, na medida em que podemos verificar quais atributos foram considerados mais importantes para a predição dos valores.

5. Projetos Relacionados

No kaggle existem alguns trabalhos relacionados que usam o mesmo dataset. Dentre eles, temos ANN-Random-Forest-Regressor(Apartment) [Wizbik 2024]; EDA and 9 models optimized [Godo Istvan 2024]; Apartment Rental Data Analysis [Joseph De Mey 2024]

Em relação o pré-processamento (limpeza e transformação) dos dados, observa-se que dos trabalhos tiveram uma abordagem semelhante, com pequenas variações. No geral, foi realizada limpeza dos dados ausentes, transformação de variáveis categóricas em numéricas, normalização de dados e eliminação de algumas colunas.

Na construção de modelos, nota-se que em alguns trabalhos houve o foco em um método específico (como em [Wizbik 2024]) e em outros foram construídos modelos com diferentes métodos, como SVR, Regressão linear, Árvore de Decisão, Floresta Aleatória, LGBM, XGB, CatBoost, AdaBoost, Bagging, que ocorre em [Godo Istvan 2024].

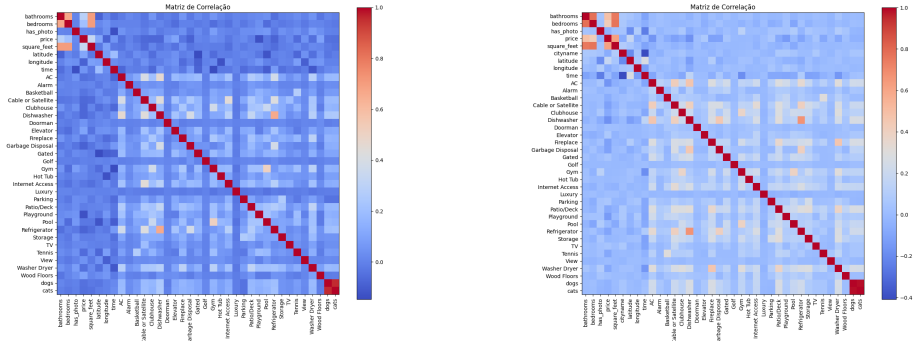
6. Experimentos Realizados

6.1. Regressão Linear

O objetivo principal foi realizar um total de 4 experimentos distintos, com a seguinte estrutura:

1. Modelo utilizando todas as features disponíveis.
2. Modelo utilizando apenas as K melhores features.
3. Modelo considerando apenas as instâncias de um estado e todas as features.
4. Modelo considerando apenas as instâncias de um estado e as K melhores features.

Antes de partirmos para a construção dos modelos de regressão linear e a divisão do conjunto de dados em grupos de treinamento e teste, buscamos analisar as características da nossa base de dados. Isso foi realizado para identificar quais atributos seriam mais relevantes para o modelo, e quais poderiam ser desconsiderados. Para tal análise, construímos a matriz de correlação entre os atributos, com o objetivo de identificar as



(a) Todas as instâncias **(b)** Apenas Texas

Figura 1. Matrizes de correlação

variáveis mais relacionadas com a feature target, ”price”. Dessa forma, pudemos também eliminar atributos com alta correlação entre si, visando reduzir redundâncias no modelo.

Em nosso experimento inicial, optamos por excluir as colunas “cityname” e “state”, por serem categóricas e acreditarmos que a localização pode ser representada adequadamente pelas coordenadas de latitude e longitude. Já nos experimentos (3) e (4), pegamos apenas instâncias referentes a um estado, e mapeamos os nomes das cidades para valores numéricos. O estado escolhido foi TX, por ter o maior número de instâncias. Também, aplicamos a regressão para os outros estados, para comparação, mas não daremos muito foco nesses resultados, apenas comentando de forma mais generalizada sobre eles.

Para selecionar as melhores features, utilizamos a função SelectKBest da biblioteca sklearn, a qual realiza a escolha dos K atributos com as maiores pontuações segundo um critério específico. No nosso caso, o critério escolhido foi a função f_regression, que calcula o F-valor entre a label e cada feature, sendo uma abordagem comum para tarefas de regressão. Essa etapa permitiu a construção de modelos tanto com todas as features quanto apenas com as selecionadas, possibilitando comparações de desempenho.

Quanto à divisão do dataset em conjuntos de treinamento e teste, adotamos o método de K-fold cross-validation, com $k = 5$. Assim, o conjunto de dados foi dividido em 5 grupos de treinamento e teste, sendo que cada grupo conta com 80% dos dados no treinamento e 20% no teste. Para cada experimento, construímos cinco modelos correspondentes aos diferentes folds e calculamos a média dos erros e o valor médio do coeficiente de determinação (R^2), para avaliar a consistência dos resultados.

6.2. Floresta Aleatória

Depois das avaliações realizadas na Regressão Linear ficamos insatisfeitos com a performance de previsão encontrada, apesar das diferentes formas de tratamento e processamento dos dados. Assim, baseados no trabalho visto em ”EDA and 9 models optimized”[Godo Istvan 2024] testamos diferentes modelos de aprendizado supervisionado buscando uma melhor performance, chegando no Random Forest Regressor como melhor candidato. O código utilizado para este experimento foi implementado em Python, com as bibliotecas scikit-learn, xgboost, lightgbm e ferramentas de otimização de hiper-

parâmetros, como o Optuna.

Para isso, os dados foram importados e processados para tratar variáveis categóricas com LabelEncoder. Os dados foram divididos em conjuntos de treino (80%) e teste (20%) com a função train_test_split. O atributo alvo foi price, enquanto as demais colunas foram usadas como variáveis independentes.

Diversos modelos de aprendizado supervisionado foram utilizados, incluindo:

- *Regressão Linear*: Modelo simples e eficaz para problemas lineares.
- *Support Vector Regressor (SVR)*: Baseado em margens com *kernel*.
- *Árvores de Decisão e Random Forest*: Modelos capazes de capturar padrões não lineares.
- *Modelos Ensemble*: Como AdaBoost, Bagging, LGBM e XGBoost.

Cada modelo foi avaliado com base no coeficiente de determinação (R^2) calculado no conjunto de teste, de modo que o Random Forest Regressor foi o com melhor resultado e, portanto, o escolhido para o treinamento e otimização de hiperparâmetros. Esse usa a combinação do resultado de várias árvores aleatórias para previnir o overfitting e melhorar a precisão.

O *Optuna* foi utilizado para otimizar os hiperparâmetros. Ele é uma biblioteca de otimização de hiperparâmetros automatizada, projetada para encontrar os valores ideais de parâmetros em problemas de aprendizado de máquina. Ele utiliza métodos como a busca bayesiana para explorar eficientemente o espaço de hiperparâmetros, equilibrando exploração (tentar novos valores) e exploração refinada (focar em regiões promissoras). A otimização de hiperparâmetros é crucial para melhorar o desempenho de modelos complexos, como o Random Forest. Métodos tradicionais, como busca em grade (Grid Search) ou busca aleatória (Random Search), podem ser computacionalmente caros e menos eficientes, especialmente quando o espaço de busca é grande. O Optuna supera essas limitações ao:

- *R*ealizar busca adaptativa com base em resultados intermediários.
- *P*ermitir a definição de métricas de desempenho, como o coeficiente de determinação (R^2).
- *I*mplementar paralelismo e suporte à execução distribuída para otimizações mais rápidas.

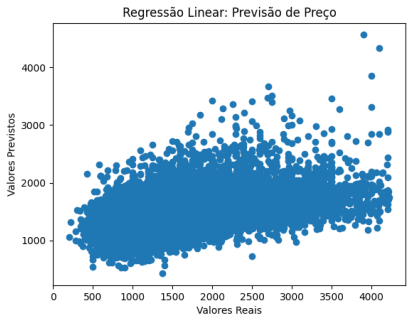
Parâmetros como número de árvores, profundidade máxima e número mínimo de amostras por nó foram ajustados, de modo que o melhor conjunto de hiperparâmetros foi identificado buscando maximizar a função objetivo (R^2).

7. Resultados

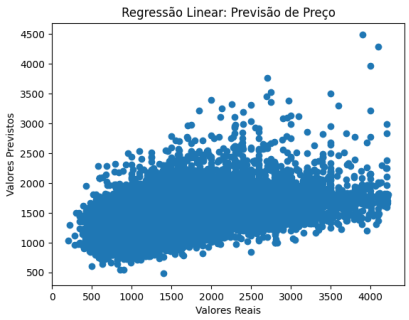
7.1. Regressão Linear

Depois de treinar os modelos, foi realizada a construção de gráfico um para mostrar a comparação de valores reais (presente na base de dados) e dos valores previstos por nossos modelos. Além disso, criamos um gráfico para comparar os erros (mean squared errors) e coeficientes de determinação (R^2).

Ao analisar os gráficos, vemos que os experimentos apresentam resultados que refletem as matrizes de correlação mostradas anteriormente. Nos experimentos (1) e (2),

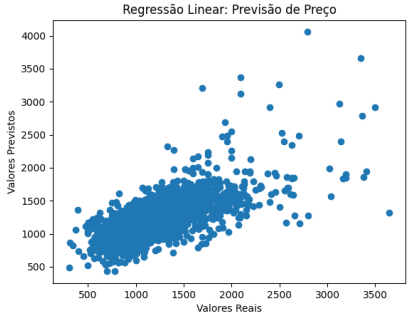


(a) Usando todos os atributos

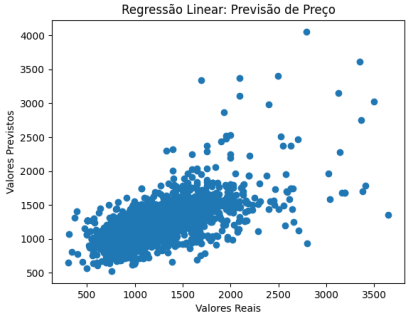


(b) Usando 10 melhores atributos

Figura 2. Gráfico Valores reais X Valores previstos, com todas as instâncias



(a) Usando todos os atributos



(b) Usando 10 melhores atributos

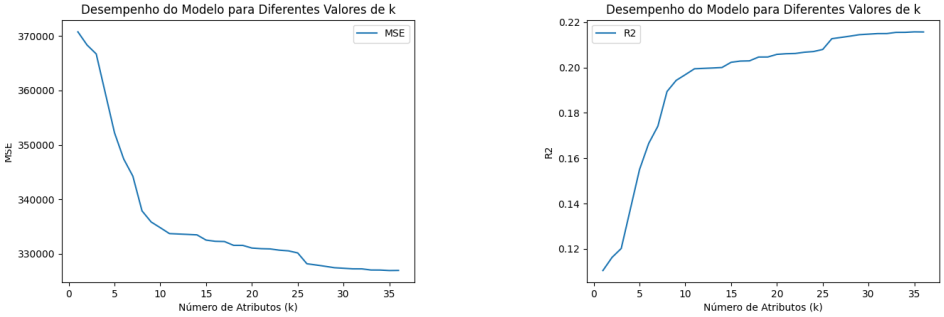
Figura 3. Gráfico Valores reais X Valores previstos, com instâncias do Texas

que utilizam todas as instâncias do dataset (após processamento), era possível verificar na matriz de correlação que poucos atributos tinham uma correlação considerável com a feature target (price). Desse modo, independente do número de atributos escolhidos, não foi possível obter uma boa relação linear entre os atributos para poder realizar a previsão desejada.

Enquanto isso, nos experimentos (3) e (4), ao limitar as instâncias com apenas aquelas referentes ao estado de TX, era perceptível na matriz de correlação a presença de mais correlações no geral, inclusive com o atributo price. Assim, vemos um modelo que conseguiu realizar uma melhor previsão.

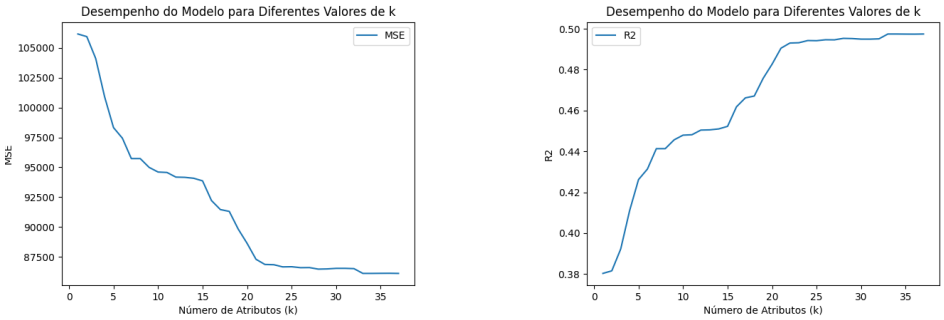
Em relação ao número de atributos selecionados, buscamos realizar uma comparação entre escolhas de diferentes quantidades de features, e construímos gráficos para facilitar a análise.

Nota-se que a utilização de poucas features resulta em erros muito altos e, a medida que aumentamos o valor de K, obtemos melhores modelos. No fim, perbe-se que os melhores resultados para ambos os tipos de experimentos (tanto analisando todo o dataset quando analisando para o TX) vemos que os modelos com melhores desempenhos são aqueles que utilizam todas (ou quase todas) as features em sua construção.



(a) MSE **(b)** R^2

Figura 4. Relação Métrica X Número de Atributos, com toodas as instâncias



(a) MSE **(b)** R^2

Figura 5. Relação Métrica X Número de Atributos, com instâncas do Texas

Em comparação com os trabalhos relacionados de outros autores, observando aqueles que fizeram um modelo de regressão linear, é possível perceber que houveram diferenças nos resultados, o que provavelmente está relacionado ao pré-processamento dos dados e na escolha dos atributos. Portanto, percebe-se que a regressão linear, para a base de dados escolhida, não possui um bom funcionamento, considerando que ela ficará razoável apenas ao limitar a análise das instâncias para um estado (e isso nem garante que o resultado não será ruim).

7.2. Regressão por Floresta Aleatória

Para o Random Forest Regressor, os parâmetros ideais encontrados pela otimização de hiperparâmetros foram:

- 'n_estimators': 932
- 'max_depth': 30
- 'min_samples_split': 4
- 'min_samples_leaf': 1
- 'bootstrap': True
- 'max_features': sqrt

Treinando o modelo em cima da base de dados processada e com os melhores hiperparâmetros encontrados obtivemos um (R^2) score de 0.857. Ao testar com o K-Fold

estratificado, o modelo não teve uma melhora significativa, com o (R^2) score aumentando em 0,01. Vamos gerar alguns gráficos para visualizar melhor os resultados da Regressao por Floresta Aleatória.

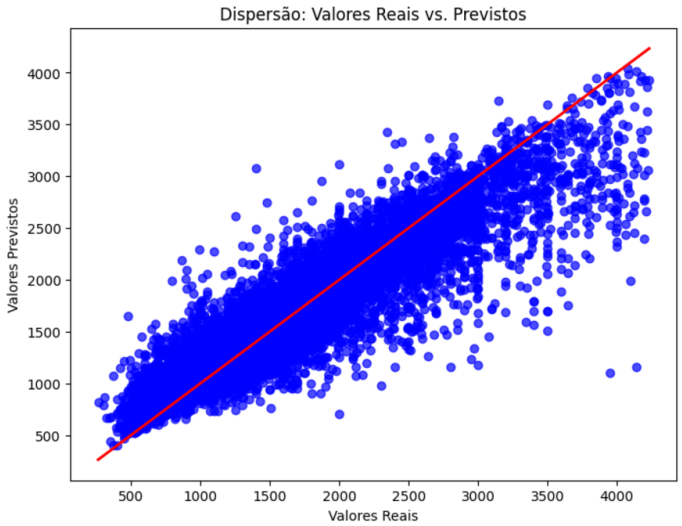


Figura 6. Gráfico de Dispersão - Random Forest Regressor

Observando o gráfico de Dispersão, temos que apesar de não ser exato, o modelo faz uma previsão, na maior parte dos casos, próxima da real.

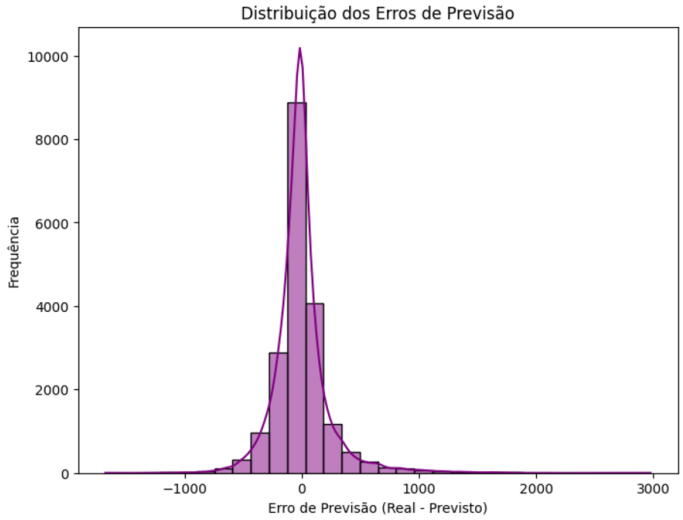


Figura 7. Distribuição dos Erros - Random Forest Regressor

Ao analisar o gráfico percebe-se que as previsões tem um bom índice de acertos e seus maiores erros (fora alguns outliers) estão em torno de 1000 dólares para mais ou para menos.

Assim, notamos que a partir do método Random Forest Regressor foi possível atingir um resultado mais satisfatório em relação a construção de um modelo para previsão de preços, comparando com a Regressão Linear.

Além disso, na Figura 8 é possíver ver a importância de cada atributo para o

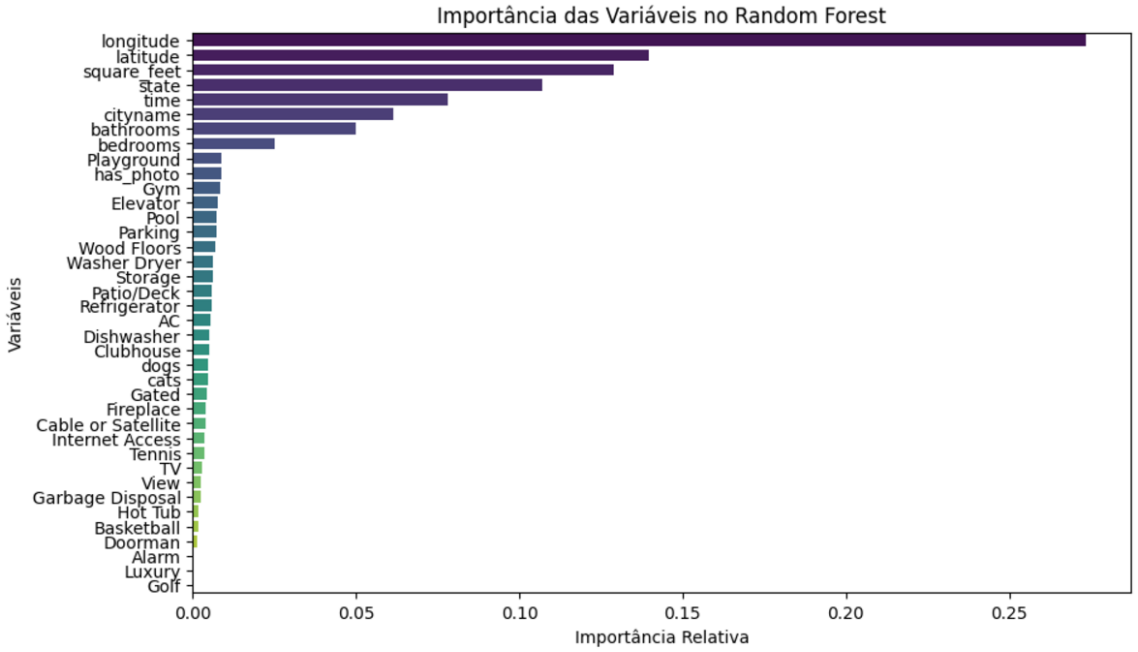


Figura 8. Importância dos Atributos - Random Forest Regressor

método e notar que, fatores que têm maior importância na vida real como posição, tamanho e quantidade de quartos, por exemplo, foram notados como de maior importância pelo modelo para prever o valor.

8. Conclusão

Neste trabalho, analisamos diferentes abordagens para prever preços, começando com a regressão linear, que apresentou desempenho limitado devido à incapacidade de capturar padrões complexos nos dados. Modelos mais avançados, como Random Forest e XGBoost, foram então testados, com o Random Forest Regressor destacando-se pela robustez e alta precisão após a otimização dos hiperparâmetros via Optuna. O estudo evidenciou que, para problemas em que os padrões nos dados não são encontrados de forma simples, modelos baseados em ensemble, aliados a estratégias modernas de otimização, são significativamente mais eficazes do que abordagens lineares simples.

Referências

Amazon Web Services (AWS) (2024). What is Linear Regression? Accessed: 2024-11-01.

Godo Istvan (2024). EDA and 9 Models Optimized. Accessed: 2024-11-01.

Joseph De Mey (2024). Apartment Rental Data Analysis. Accessed: 2024-11-01.

Manning, C. D. (2008). *Introduction to information retrieval*. Syngress Publishing,. Capítulos 1, 2, 6 e 8.

Wizbik (2024). ANN Random Forest Regressor: Apartment Rent Prediction. Accessed: 2024-11-01.