PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS
NÚCLEO DE EDUCAÇÃO A DISTÂNCIA
Pós-graduação *Lato Sensu* em Inteligência Artificial e Aprendizado de Máquina

Marcelo de Oliveira

APLICAÇÃO DE MACHINE LEARNING NA PREVISÃO DE PREÇOS DE IMÓVEIS EM BELO HORIZONTE

Belo Horizonte
2022

Marcelo de Oliveira

APLICAÇÃO DE MACHINE LEARNING NA PREVISÃO DE PREÇOS DE IMÓVEIS EM BELO HORIZONTE

Trabalho de Conclusão de Curso apresentado ao Curso de Especialização em Inteligência Artificial e Aprendizado de Máquina como requisito parcial à obtenção do título de especialista.

Belo Horizonte
2022

SUMÁRIO

1. Introdução

1.1. Contextualização

O preço dos imóveis reflete diretamente a economia de um país, possuindo forte relação com o crescimento econômico, crescimento das cidades, além de ser um dos principais investimentos procurados pelos brasileiros, muitas vezes tratado como reserva “segura”. É neste contexto que surge a necessidade de se compreender a valorização dos imóveis nas mais diversas regiões e cidades do Brasil.

O objetivo deste trabalho é demonstrar, através de *machine learning*, que a partir de informações comuns e publicamente disponíveis de imóveis, é possível prever seu valor. Apesar de utilizarmos dados da cidade de Belo Horizonte, a abordagem poderá ser utilizada em dados de qualquer cidade ou região brasileira, que possuam as informações necessárias para a análise.

1.2. O Problema Proposto

Poder prever o valor de um imóvel é de suma importância para qualquer investidor do setor imobiliário. Principalmente se isso puder ser feito a partir de atributos simples e de fácil obtenção.

Os dados analisados são da cidade de Belo Horizonte e foram obtidos de um site imobiliário, e se referem ao período de novembro de 2021.

O objetivo é tentar prever o valor do imóvel com base em seus atributos mais comuns: área, endereço, valor de condomínio, vagas de garagem, número de quartos, bairro e localização geográfica.

2. Coleta de Dados

A base de dados utilizada nesse trabalho contém 5981 registros de imóveis da cidade de Belo Horizonte. Os dados foram obtidos de um site imobiliário brasileiro e

correspondem a informações reais de casas e apartamentos. O conjunto foi extraído em novembro de 2021.

O *dataset* foi obtido no site **Kaggle**[1], no formato CSV, que pode ser observado em detalhe na **Tabela 1**. O Kaggle é uma plataforma para competições de Data Science e disponibiliza conjuntos de dados sobre os mais variados assuntos. A plataforma também possui fóruns para troca de conhecimentos entre seus usuários.

NOME DA COLUNA	DESCRIÇÃO	TIPO
Cidade	Nome da cidade	String
Endereço	Endereço do imóvel	String
Condomínio	Valor da taxa de condomínio	Float
Vagas	Número de vagas de garagem	Int
Quartos	Número de quartos	Int
Área	Área do imóvel em m2	Int
Bairro	Nome do bairro	String
Latitude	Latitude	Float
Longitude	Longitude	Float
Preço	Valor do imóvel	Float

Tabela 1 – Formato do Dataset

3. Processamento/Tratamento de Dados

O conjunto de dados possui 5981 registros e 10 colunas, que correspondem as 9 variáveis de entrada (preditoras) e uma variável alvo (preço do imóvel) (**Figura 1**).

```
# leitura do arquivo
bh = pd.read_csv('data.csv')
bh.info()
```

```
RangeIndex: 5981 entries, 0 to 5980
```

[1] https://www.kaggle.com/datasets/guilherme26/house-pricing-in-belo-horizonte/download

```
Data columns (total 10 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   address        5981 non-null   object
 1   adm-fees       3977 non-null   float64
 2   garage-places  5981 non-null   object
 3   price          5951 non-null   float64
 4   rooms          5981 non-null   object
 5   square-foot    5981 non-null   object
 6   neighborhood   5957 non-null   object
 7   city           5981 non-null   object
 8   latitude       5981 non-null   float64
 9   longitude      5981 non-null   float64
dtypes: float64(4), object(6)
memory usage: 467.4+ KB
```

Figura 1 - Variáveis

Inicialmente foi aplicado novos nomes às colunas (**Figura 2**).

```
# renomear colunas
bh = bh.rename(columns={
   'city':'cidade',
   'address':'endereco',
   'adm-fees':'condominio',
   'garage-places':'vagas',
   'price':'preco',
   'rooms':'quartos',
   'square-foot':'area',
   'neighborhood':'bairro',
})
```

Figura 2 - Variáveis renomeadas

Em seguida verificou-se que o conjunto continha 32 registros que não pertenciam à cidade de Belo Horizonte, que foram excluídos. Como a análise envolverá imóveis apenas da cidade de Belo Horizonte, a coluna “cidade” foi removida. Sua presença não traria contribuições para o modelo (**Figura 3**).

```
# remoção dos imóveis não pertencentes à Belo Horizonte
bh = bh.loc[bh.cidade.str.contains('Belo Horizonte')].drop(columns=['cidade'])
```

Figura 3 - Exclusão da coluna Cidade

Observou-se que em 26 registros o preço do imóvel estava ausente. Devido à pequena quantidade de registros, decidiu-se por excluí-los, uma vez que essa quantidade pouco afetaria o resultado da análise (**Figura 4**).

```
# preço: remover registros nulos
bh = bh.loc[bh.preco.notna()]
```

Figura 4 - Exclusão de registros sem preço

Uma rápida olhada nos preços de imóveis identificou alguns valores bastante altos (*outliers*), chegando a R$ 130.000.000, o que pode afetar a qualidade do modelo. É provável que tais ocorrências representem erros no registro ou problemas na coleta da informação. Da mesma forma, valores muito baixos foram encontrados e decidiu-se por removê-los. Para esta análise iremos considerar preços entre R$ 80.000 e R$ 5.000.000, o que parece ser bastante razoável para a realidade do mercado local (**Figura 5**).

```
# preço: remoção dos outliers
bh = bh[(bh.preco >= 80000) & (bh.preco <= 5000000)]
```

Figura 5 - Exclusão de registros com preços fora de limites

Verificando a coluna "condomínio", constatou-se uma grande quantidade de valores ausentes. A ausência desse valor representa “nenhum custo”. Neste caso optou-se por atribuir zero aos valores ausentes. A substituição não causará perda de informação, pois representará o real significado da informação (**Figura 6**).

```
# preecher com zero os imóveis sem custo de condomínio
bh = bh.fillna({'condominio': 0})
bh = bh.astype({'condominio': int})
```

Figura 6 - Preenchimento da coluna Condomínio

A análise das colunas “quartos”, “vagas” e “área” indicou que 665 registros estavam sem valor em pelo menos uma das colunas. Devido à grande influência destas informações na definição do valor do imóvel, optou-se por remover as linhas com valores faltantes.

Entretanto, analisando estas mesmas colunas, foram encontrados diversos valores no formato N-N, o que sugere a representação de uma faixa de valores no formato “MÍNIMO-MÁXIMO”. Constatou-se que não havia nenhum valor discrepante entre mínimo e máximo, e por isso optou-se pelo aproveitamento dos registros (**Figura 7**).

```
# Amostra de valores no formato N-N
bh.loc[(~bh.quartos.str.isnumeric())
            & (~bh.vagas.str.isnumeric())
            & (~bh.area.str.isnumeric())
            & (~bh.vagas.str.contains('--')), ['quartos', 'vagas', 'area']]

quartos vagas     area
   3-4   2-3   87-134
   3-4   2-3  103-128
   2-3   2-3    63-83
   2-3   1-2    55-75
```

Figura 7 - Colunas com intervalo de valores

Era necessário que as colunas “quartos”, “vagas” e “área” estivessem no formato numérico. Várias abordagens poderiam ter sido aplicadas na transformação da informação: assumir o valor mínimo, máximo ou a média dos valores. Optou-se por utilizar o valor máximo, uma vez que aparentava ser, na maioria dos casos, o valor mais adequado para a coluna em questão (**Figura 8**).

```
# Remover valores ausentes e transformar valores N-N em numérico
for col in ['quartos', 'vagas', 'area']:
    bh = bh.loc[~bh.loc[:,col].str.startswith('--')]
    vals = bh.loc[~bh.loc[:,col].str.isnumeric(), col].str.split('-',expand=True)
    bh.loc[~bh.loc[:,col].str.isnumeric(), col] = vals[1]
```

```
bh = bh.astype({col: int})
```

Figura 8 - Transformação de colunas

A verificação do campo endereço indicou que a informação estava bem formatada e sem valores faltantes. Não foram encontrados endereços com erros de digitação, o que demostrou rigor na captação da informação.

Foram encontrados dois imóveis com a informação “bairro” gravadas no campo endereço. Por se tratar de apenas dois registros, optou-se por obter manualmente o endereço dos imóveis no Google Maps a partir das coordenadas geográficas disponíveis no dataset.

Entretanto, muitos registros estavam sem a informação “número”. Visto que a informação mais significativa (nome da rua) estava disponível em 100% dos registros, optou-se por remover o número do campo endereço (**Figura 9**).

```
# remover número do endereço
bh['endereco'] = bh['endereco'].str.split(',', expand=True)[0].str.strip()

# remover brancos do campo bairro
bh.bairro = bh.bairro.str.strip()

# preenchimento de endereços faltante
bh.loc[bh.endereco.str.contains('Nova Vista'), 'bairro'] = 'Nova Vista'
bh.loc[bh.endereco.str.contains('Nova Vista'), 'endereco'] = 'Rua São Fidélis'

bh.loc[bh.endereco.str.contains('Vila da Serra'), 'bairro'] = 'Vila da Serra'
bh.loc[bh.endereco.str.contains('Vila da Serra'), 'endereco'] = 'Rua do Vale'
```

Figura 9 - Preenchimento de dados faltantes

Da mesma forma que o endereço, a análise do campo bairro indicou uma boa qualidade da informação, sem erros de digitação ou valores similares. Entretanto, 20 registros estavam sem a informação. Primeiramente tentou-se preencher a informação obtendo o bairro de outros imóveis localizados na mesma rua, mas nenhum foi encontrado.

A segunda opção foi obter o bairro a partir da localização geográfica do imóvel (latitude/longitude), o que se demonstrou totalmente viável. Os 20 registros estavam concentrados em apenas quatro bairros. Dessa forma, a partir das coordenadas geográficas, os quatro bairros faltantes foram obtidos manualmente no Google Maps e inseridos no dataset (**Figura 10**).

```
# preencher bairros faltantes
bh.loc[(bh.bairro.isna()) & (bh.endereco.str.startswith('Rua Beija')), 'bairro'] = 'Planalto'
bh.loc[(bh.bairro.isna()) & (bh.endereco.str.startswith('Rua Flor')), 'bairro'] = 'Beija Flor'
bh.loc[(bh.bairro.isna()) & (bh.endereco.str.startswith('Rua Lírios')), 'bairro'] = 'Monte Verde'
bh.loc[(bh.bairro.isna()) & (bh.endereco.str.startswith('Rua Pium')), 'bairro'] = 'Anchieta'
```

Figura 10 - Preenchimento de bairros faltantes

Ao final do processo de tratamento dos dados, o dataset resultante ficou com **5040 registros**. A análise dos dados se mostrou muito importante para garantir uma entrada de dados com boa qualidade e homogeneidade.

4. Análise e Exploração dos Dados

Realizou-se a Análise de correlação das variáveis, para identificar problemas que comprometessem os resultados, como a colinearidade, que é a relação entre duas (colinearidade) ou mais (multicolinearidade) variáveis independentes (**Figura 11**).

```
# Correlação
plt.figure(figsize=(12, 6))
hm = sns.heatmap(bh.corr(), vmin=-1, vmax=1, annot=True)
hm.set_title('Correlação', pad=12);
plt.show()
```

Figura 11 - Análise de Correlação de variáveis

Com exceção das variáveis latitude e longitude, que naturalmente estão relacionadas, a análise de correlação demonstrou que nenhuma das variáveis apresentou forte correlação, o que eliminará do modelo o problema de colinearidade.

Em relação à variável alvo (preço), as varáveis "vagas", "quartos" e "área" apresentaram correlação moderada, portanto não foram identificadas correlações altas o suficiente para justificar a remoção de variáveis, como pode ser observado no **Gráfico 1**.

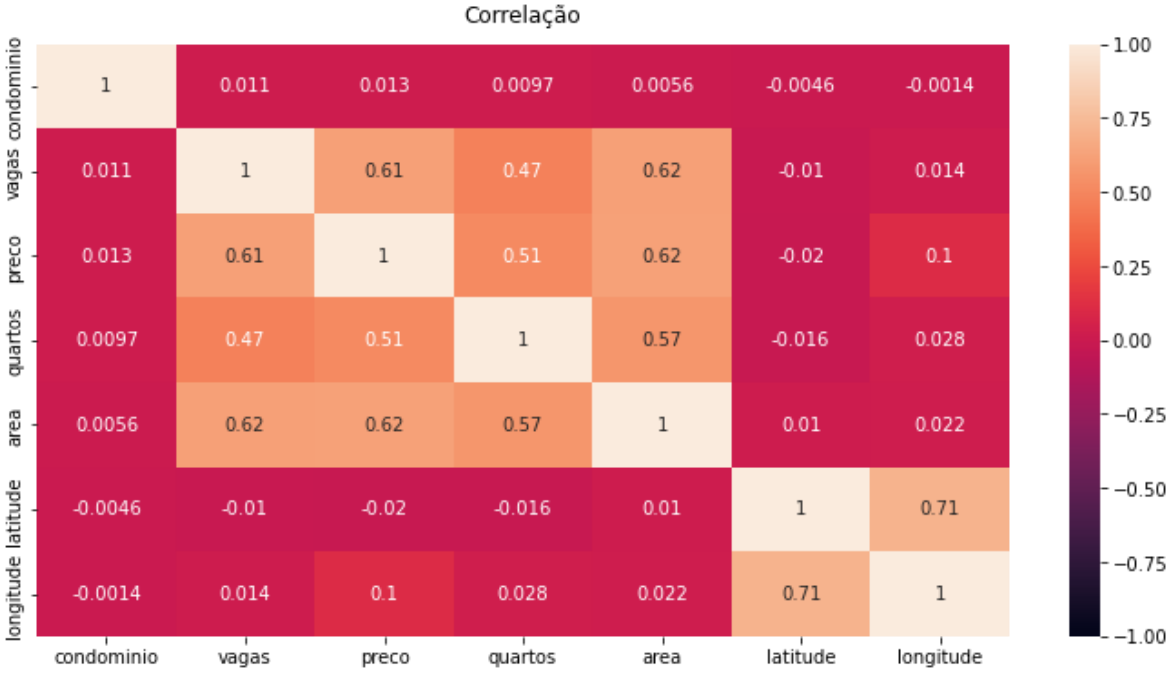

Gráfico 1 – Resultado da Análise de Correlação de variáveis

O histograma da varável "preço" revelou uma assimetria à esquerda, demonstrando alta concentração de imóveis com valores entre 400 e 900 mil reais (**Gráfico 2**).

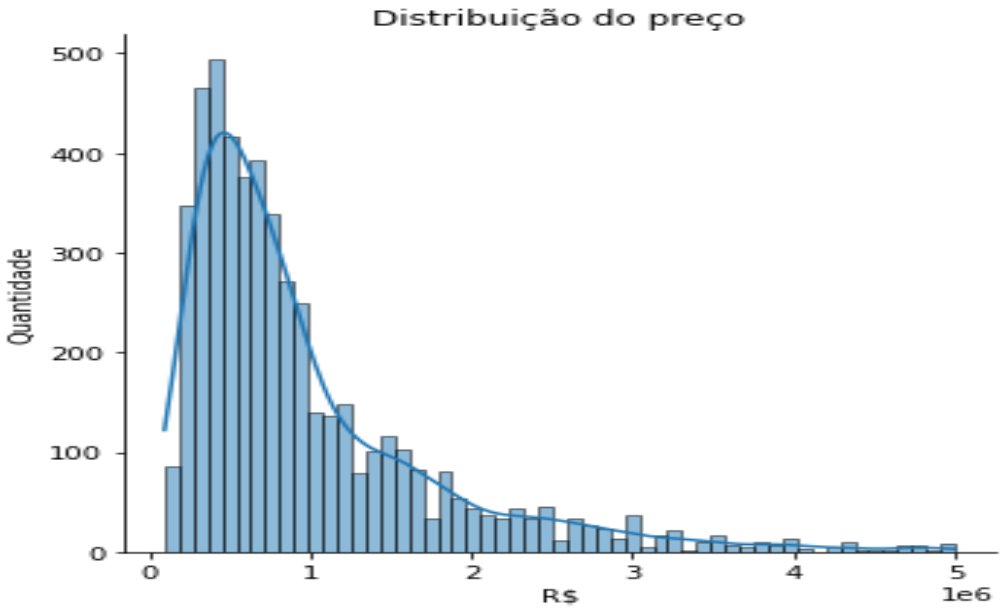

Gráfico 2 - Histograma da variável Preço

A distribuição no "número de vagas", "número de quartos" e "área" seguiram a mesma tendência, como já demonstrado no mapa de correlação, o que pode ser comparado nos **Gráficos 3, 4 e 5**.

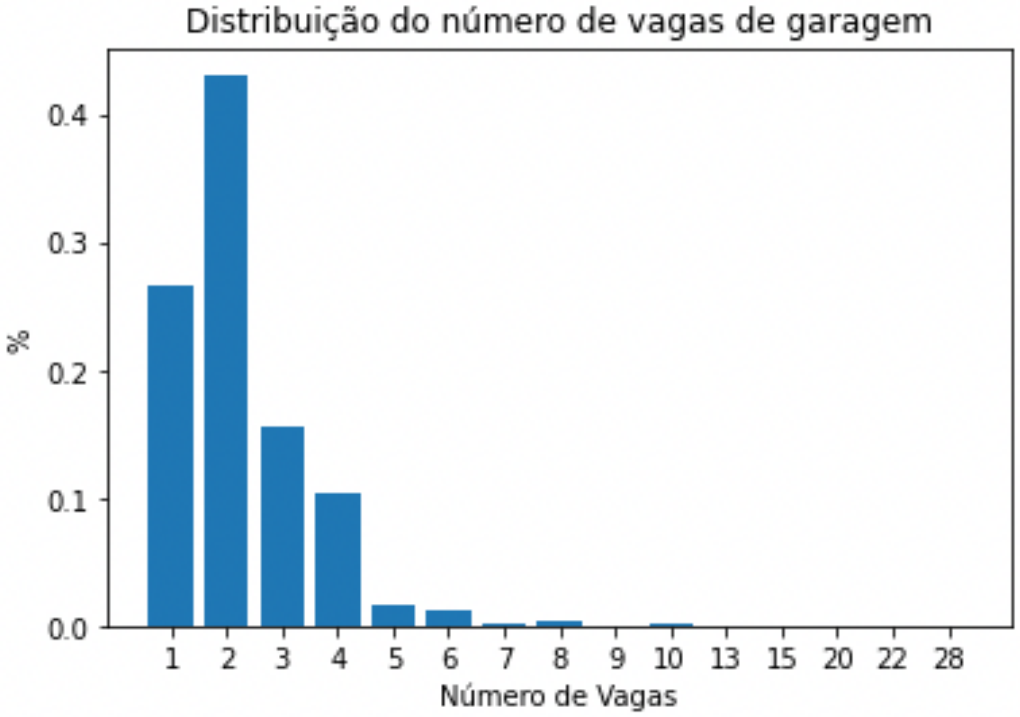

Gráfico 3 - Histograma da variável Número de Vagas

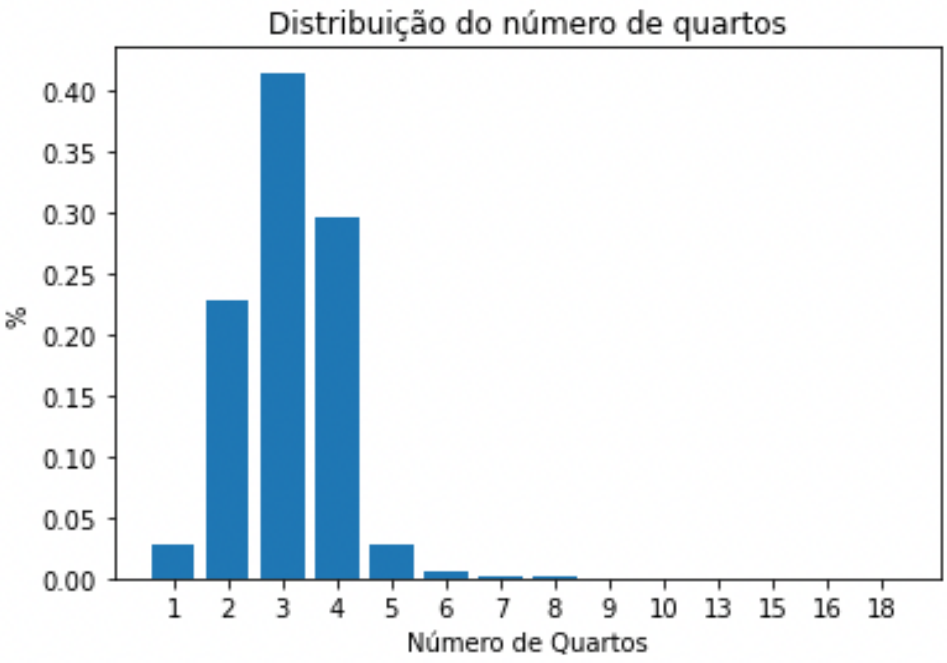

Gráfico 4 - Histograma da variável Número de Quartos

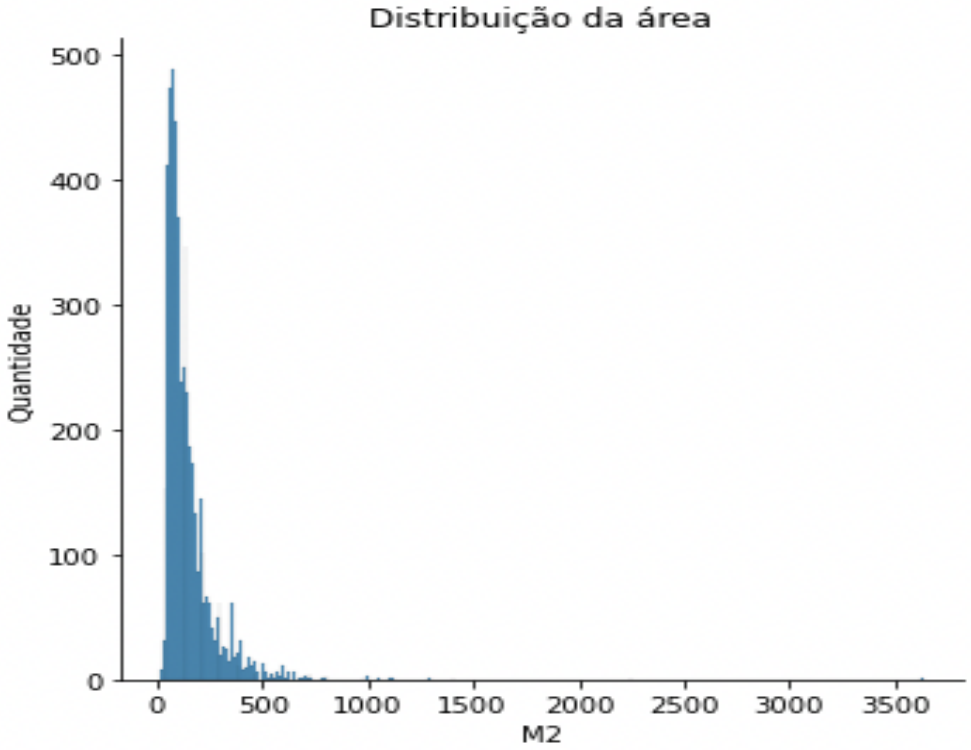

Gráfico 5 - Histograma da variável Área

5. Criação de Modelos de Machine Learning

Para a criação dos modelos optou-se por utilizar a biblioteca scikit-learn. O scikit-learn é uma biblioteca Python de código aberto desenvolvida para a aplicação da aprendizagem de máquina. Ela dispõe de ferramentas simples, eficientes, reutilizáveis e possui diversos algoritmos para a criação de modelos de regressão linear. No treinamento foi utilizado um subconjunto dos algoritmos disponíveis a fim de experimentar e selecionar o algoritmo ideal.

O dataset foi dividido em dois conjuntos, para as fases de treinamento e teste, sendo um com 70% dos dados, para a fase de treinamento, e outro com 30%, para a fase teste.

Inicialmente foi utilizada a métrica "R2" (coeficiente de determinação), muito útil para uma avaliação inicial. O treinamento em apenas um conjunto de dados torna a avaliação menos abrangente, por isso optou-se por utilizar a técnica de validação cruzada (*cross validation*). Neste método o dataset é particionado em conjuntos menores, mutuamente exclusivos, o que permite treinar e testar vários conjuntos diferentes. O dataset foi particionado em cinco grupos e o valores abaixo (**Tabela 1**) representam o coeficiente médio.

Regressor	**R2**
RandomForestRegressor	0.825536
ExtraTreesRegressor	0.813019
GradientBoostingRegressor	0.802545
DecisionTreeRegressor	0.702833
LinearRegression	0.476343

Tabela 1 – Resultados de Coeficiente médio

Três algoritmos mostraram melhor desempenho: *ExtraTreesRegressor, RandomForestRegressor* e *GradientBoostingRegressor*. Entretanto, dois atributos categóricos foram removidos para permitir o treinamento: **endereco** e **bairro**.

Para incluir estes atributos numa nova bateria de treinamento foi necessário transformá-los para o tipo numérico, utilizando a técnica *One Hot Encoding*, onde cada

valor único de categoria é convertido em uma coluna independente do dataset indicando se a categoria está presente (1) ou ausente (0) no respectivo registro.

Houve uma melhora visível no resultado considerando os três melhores algoritmos selecionados no primeiro treinamento, como pode ser comparado na **Tabela 2**.

Regressor	R2
ExtraTreesRegressor	0.836633
RandomForestRegressor	0.831068
GradientBoostingRegressor	0.804662

Tabela 2 – Resultados após conversão das colunas Endereço e Bairro

Foram feitos alguns treinamentos removendo algumas variáveis para verificar sua influência no modelo. Todas elas modificaram sensivelmente o resultado, demostrando que eram relevantes. A **Tabela 3** apresenta o resultado de um treinamento onde a coluna "quartos" foi removida.

Regressor	R2
ExtraTreesRegressor	0.832762
RandomForestRegressor	0.830659
GradientBoostingRegressor	0.802858

Tabela 3 – Resultados de Validação Cruzada com remoção da coluna Quartos

Como já visto, nosso dataset possui vários campos numéricos, alguns deles com alta escala de variação. Os treinamentos até aqui foram feitos sem nenhuma transformação nesses valores. Foram feitos alguns treinamentos onde tais informações foram "padronizadas". Dois métodos foram experimentados:

1. **StandardScaler**: padronização da informação (subtrai a média e, em seguida, escala para a variância da unidade).
2. **MinMaxScaler**: escala todos os valores no intervalo [0, 1] (também pode ser feito no [-1, 1]).

O resultado demonstrou que a transformação influenciou muito pouco nos resultados.

Regressor	Sem escala	StandardScaler	MinMaxScaler
ExtraTreesRegressor	0. 836633	0.835989	0.836121
RandomForestRegressor	0. 831068	0.830910	0.830777
GradientBoostingRegressor	0. 804662	0.804815	0.804726

Tabela 4 – Resultados com padronização de campos numéricos

Como próximo passo, foi incluída a métrica RMSE (*Root Mean Squared Error)* na análise, para fornecer mais insumos nas avaliações do modelo. A métrica MSE penaliza os erros ao elevá-los ao quadrado, sendo a RMSE a raiz quadrada da MSE. Ela foi escolhida por estar na mesma unidade da variável alvo (o preço do imóvel). A **Tabela 5** a seguir demonstra os resultados para cada algoritmo.

Regressor	R2	RMSE
ExtraTreesRegressor	0. 836633	320120.88
RandomForestRegressor	0. 831068	325046.24
GradientBoostingRegressor	0. 804662	350380.76

Tabela 5 – Resultados com aplicação de RMSE

6. Apresentação dos Resultados

Depois de diversas análises e comparações, com detalhado exame dos resultados obtidos, foi possível demonstrar que dois algoritmos obtiveram melhor desempenho (maior R2 e menor RMSE): *ExtraTreesRegressor* e *RandomForestRegressor*.

O modelo selecionado foi o *ExtraTreesRegressor*, que apresentou erros ligeiramente menores que o *RandomForestRegressor*. O RMSE ficou significativamente alto, o que ainda representa uma diferença importante para um valor de imóvel.

O **Gráfico 6** apresenta o gráfico de dispersão com a diferença entre valores de preço de imóveis reais e previstos no conjunto de teste.

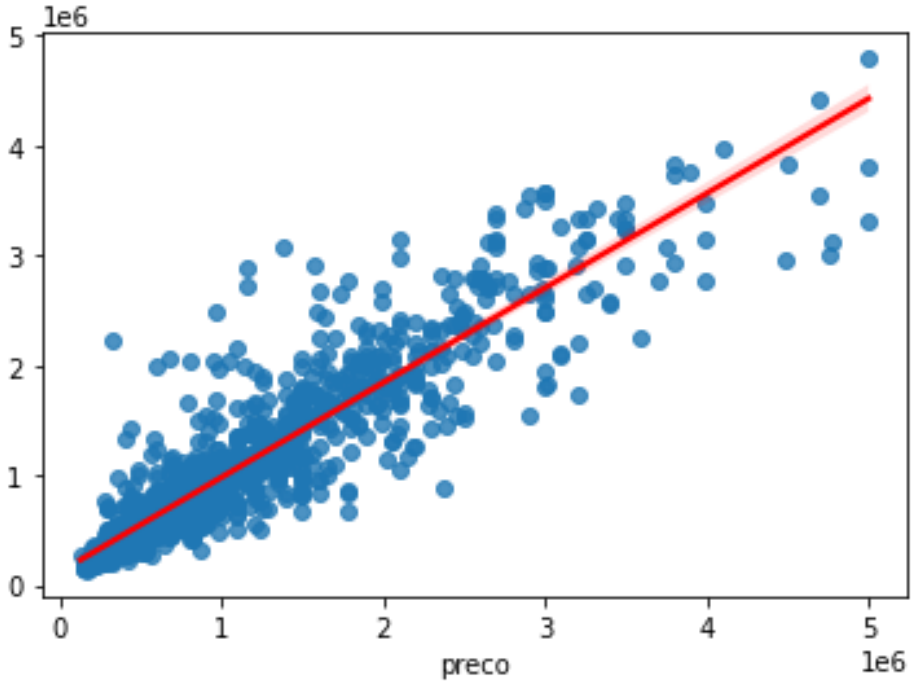

Gráfico 6 - Análise de Dispersão de Preços

O gráfico de resíduos (**Gráfico 7)** mostra diferença entre o valor real e o valor da predição dos preços de imóveis. O gráfico exibe uma simetria relativa dos valores, indicando que houve pouca interferência de valores fora do esperado (*outliers*).

Destaque para a quantidade sensivelmente maior de números negativos na faixa dos imóveis de maior valor, mostrando que a predição ficou acima do valor real na maioria dos casos.

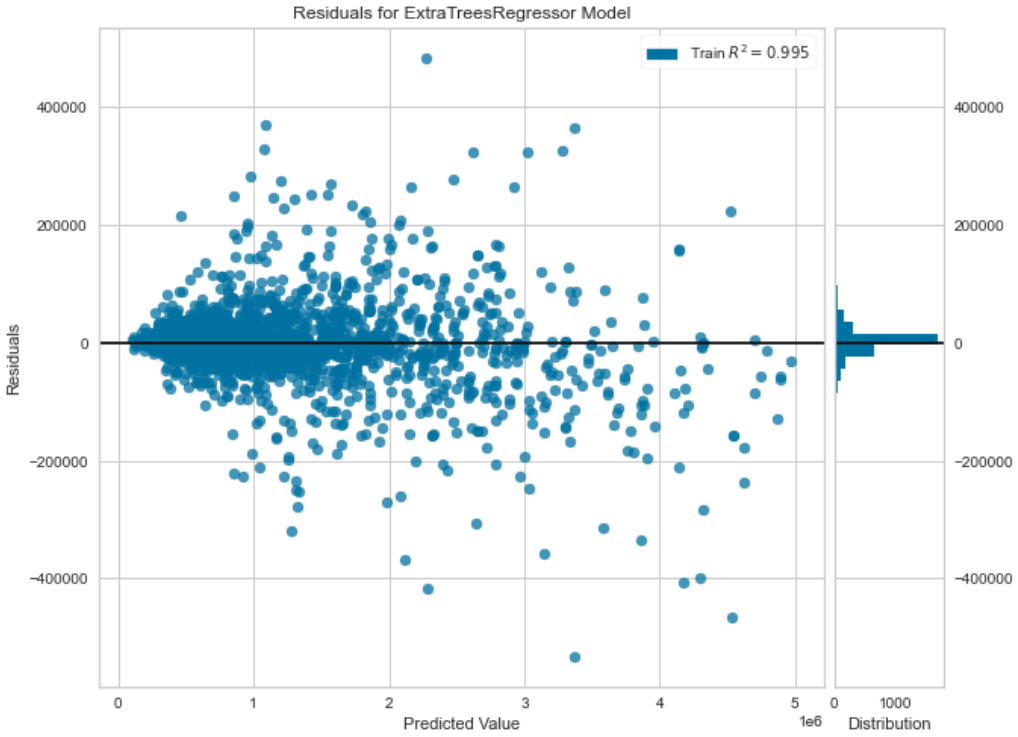

Gráfico 7 - Análise de resíduos

Assim, concluiu-se que existe potencial positivo (e inexplorado) para a utilização do modelo resultante (*ExtraTreesRegressor*) na predição de valores de imóveis. Essa ferramenta seria capaz de dar suporte a decisões de precificação para todo o ecossistema que trabalha com o mercado imobiliário.

Neste trabalho foram utilizados os atributos mais comuns, disponíveis publicamente em sites de comércio de imóveis, demonstrando sua fácil aplicabilidade e capacidade de disponibilização de forma escalável.

E mais: a depender do site utilizado, muitas outras informações poderão ser obtidas, criando um dataset mais rico e consistente, o que pode proporcionar predições ainda melhores e percepções valiosas sobre o mercado imobiliário.

7. Fontes

- Código Fonte (acessado em: 09/05/2022)
  https://github.com/mobhz/predicao_imoveis

- Dataset (acessado em: 05/05/2022)
  https://www.kaggle.com/datasets/guilherme26/house-pricing-in-belo-horizonte/download

- Scikit-Learn (acessado em: 05/05/2022)
  https://scikit-learn.org