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

Lucas Guimarães Cavalheiro

Inteligência Artificial e predição de preços de aluguel de imóveis: Abordagens voltadas ao consumidor

Belo Horizonte
Agosto de 2023

Lucas Guimarães Cavalheiro

Inteligência Artificial e predição de preços de aluguel de imóveis: Abordagens voltadas ao consumidor

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
Agosto de 2023

SUMÁRIO

1. Introdução

O mercado imobiliário é um setor de grande importância na economia e na sociedade como um todo, sendo o preço do aluguel de um imóvel algo altamente importante para o cidadão brasileiro. Com o avanço da tecnologia no campo da Inteligência Artificial e Aprendizado de Máquina, surgem novas oportunidades para aprimorar a precisão e eficiência das previsões de preços de imóveis, tando para compra quanto para aluguel. Esta pesquisa propõe explorar o uso destas tecnologias para prever os preços de aluguel de imóveis, considerando suas potenciais aplicações, desafios e benefícios.

Abordaremos as principais técnicas de Inteligência Artificial e Aprendizado de Máquina empregadas no contexto de previsão de preços de aluguéis imobiliários, tendo como foco o consumidor final, sendo este considerado o grupo que maior necessita deste tipo de auxílio. Além disso, analisaremos as fontes de dados relevantes e as características mais significativas para a construção de modelos preditivos precisos.

Espera-se que esta pesquisa contribua para a expansão do conhecimento sobre o uso de Inteligência Artificial e Aprendizado de Máquina na previsão de preços de aluguel de imóveis, destacando as possibilidades e limitações dessa abordagem. Aprofundar o entendimento nessa área pode levar a avanços significativos no mercado imobiliário, tornando-o mais transparente, eficiente e acessível para todas as partes envolvidas.

2. Descrição do Problema e da Solução Proposta

O preço do aluguel é influenciado por diversos fatores, como localização, tamanho, andar do imóvel e condições de entrega e moradia (se é ou não mobiliado, por exemplo). A capacidade de prever com precisão os preços de aluguel é essencial para inquilinos em busca de uma moradia adequada, que se encaixe na sua situação atual de vida e financeira.

A justificativa para este estudo reside na necessidade de melhorar a acurácia das previsões de aluguel, o que pode ser alcançado através da aplicação de técnicas avançadas de Aprendizado de Máquina e Inteligência Artificial. A crescente disponibilidade de dados relacionados a imóveis, combinada com o poder

computacional aprimorado, torna possível explorar modelos preditivos complexos para identificar padrões e correlações que podem passar despercebidos em análises tradicionais.

A motivação que inspira esta pesquisa é proporcionar benefícios para os inquilinos, que poderão tomar decisões mais informadas ao buscar moradia, estimando melhor o custo das propriedades em diferentes localidades e ajustando suas escolhas de acordo com suas preferências e orçamentos.

Os principais objetivos desta pesquisa são:

1. Explorar diferentes técnicas de Aprendizado de Máquina, como regressão, árvores de decisão, entre outras, para determinar aquelas que melhor se adaptam ao problema em questão.
2. Utilizar técnicas de pré-processamento de dados para lidar com possíveis ruídos e valores ausentes, garantindo a qualidade das previsões.
3. Avaliar a eficácia dos modelos desenvolvidos e validar os resultados em um conjunto de dados diversificado.

Para alcançar esses objetivos, as tarefas de Aprendizado de Máquina a serem executadas incluem a coleta e preparação de dados, seleção de características relevantes, escolha de algoritmos adequados e treinamento dos modelos. Além disso, será necessário realizar a avaliação e ajuste dos modelos para garantir que eles sejam generalizáveis e precisos o suficiente para prever preços de aluguel em novos cenários.

Como soluções, espera-se desenvolver modelos de Aprendizado de Máquina capazes de prever os preços de aluguel com acurácia satisfatória, fornecendo ferramentas valiosas para inquilinos. A utilização destas técnicas avançadas permitirá revelar padrões complexos e relacionamentos ocultos nos dados, gerando insights valiosos para tomada de decisões informadas nesse setor em constante evolução.

3. Coleta de Dados

Os dados usados nesta pesquisa foram coletados do site kaggle.com no dia 5 de Agosto de 2023, este dataset contém mais de 10 mil registros de imóveis de aluguel nas cidades do Rio de Janeiro, São Paulo, Belo Horizonte, Campinas e

Porto Alegre, estes dados foram coletados por meio de *web crawler* durante o ano de 2020.

Nome do dataset: Brazilian houses to rent

Descrição: Dataset coletado durante o ano de 2020, possui 10962 casas para alugar com 13 classes diferentes

Link: https://www.kaggle.com/datasets/0fc2c2957155f98e380a0e5e7db219aff29962b37b13e6ac5a569389cfe26e83?select=houses_to_rent_v2.csv

Nome do Atributo	Descrição	Tipo
City	Cidade onde se encontra o imóvel	string
Area	Área do imóvel	int64
Rooms	Quantidade de quartos	int64
Bathroom	Quantidade de banheiros	int64
Parking spaces	Quantidade de vagas para estacionar	int64
Floor	Andar onde se encontra o imóvel	int64
Animal	Define se o imóvel aceita animais	string
Furniture	Define se o imóvel é ou não mobiliado	string
Hoa (R$)	Valor da taxa condominial	int64
Rent Amount (R$)	Valor do aluguel	int64
Property Tax (R$)	Valor do IPTU	int64
Fire Insurance (R$)	Valor do seguro contra incêndio	int64
Total (R$)	Somatório de todas as taxas	int64

4. Processamento/Tratamento de Dados

Para o tratamento inicial dos dados, o dataset foi inspecionado com o objetivo de encontrar e corrigir: registros duplicados, valores não-numéricos, valores nulos e *outliers.* Foi utilizado um jupyter-notebook dentro da ferramenta JupyterLabs do pacote Anaconda para explicar e demonstrar os passos necessários deste processo. Quanto as bibliotecas Python utilizadas, estas foram Pandas e matplotlib. O notebook referenciado nesta parte da pesquisa se encontra na pasta “A3” com o nome de “tratamento-de-dados.ipynb”.

Para detecção de registros duplicados, foram utilizadas as funções *duplicated* e *value_counts* para separar e contar as entradas únicas e duplicadas, depois, foi usada a função *drop_duplicates* para de fato remover as duplicidades, note que o parâmetro *inplace* se faz necessário para persistir as mudanças no dataset, ao invés de executa-las em uma cópia do mesmo:

```
data.duplicated().value_counts()
```

```
False    10334
True       358
dtype: int64
```

```
data.drop_duplicates(inplace=True)
data.duplicated().value_counts()
```

```
False    10334
dtype: int64
```

Para tratamento de valores não-numéricos, foram observadas as classes *city, furniture, animal* e *floor,* cada uma destas classes foi tratada de um modo diferente. Começando com a classe *floor,* podemos notar que imóveis térreos possuem o valor “-”, logo, podemos substituir este valor por 0:

```
data.loc[data.floor == '-', 'floor'] = 0

data.floor.value_counts()
```

No tratamento das colunas *animal* e *furniture,* podemos notar que os valores não-numéricos possuem função de sim/não, verdadeiro/falso, permitido/proibido etc, logo, podemos substituir “sim” por 1 e “não” por 0:

```
data.loc[data.animal == 'acept', 'animal'] = 1
data.loc[data.animal == 'not acept', 'animal'] = 0

data.loc[data.furniture == 'furnished', 'furniture'] = 1
data.loc[data.furniture == 'not furnished', 'furniture'] = 0

data.head()
```

	area	rooms	bathroom	parking spaces	floor	animal	furniture	hoa (R$)	rent amount (R$)	property tax (R$)	fire insurance (R$)	total (R$)	city_Belo Horizonte	city_Campinas	cit
0	70	2	1	1	7	1	1	2065	3300	211	42	5618	0	0	
1	320	4	4	0	20	1	0	1200	4960	1750	63	7973	0	0	
2	80	1	1	1	6	1	0	1000	2800	0	41	3841	0	0	
3	51	2	1	0	2	1	0	270	1112	22	17	1421	0	0	
4	25	1	1	0	1	0	0	0	800	25	11	836	0	0	

Por fim, notamos na coluna *city* que cada valor não-numérico representa a localidade de cada imóvel, neste caso, podemos transformar cada um destes valores em uma coluna própria, com o valor 1 caso o imóvel esteja localizado naquela cidade, ou 0 caso ele não esteja. Em seguida, podemos remover de forma segura a coluna *city,* pois continuamos retendo a informação desta coluna:

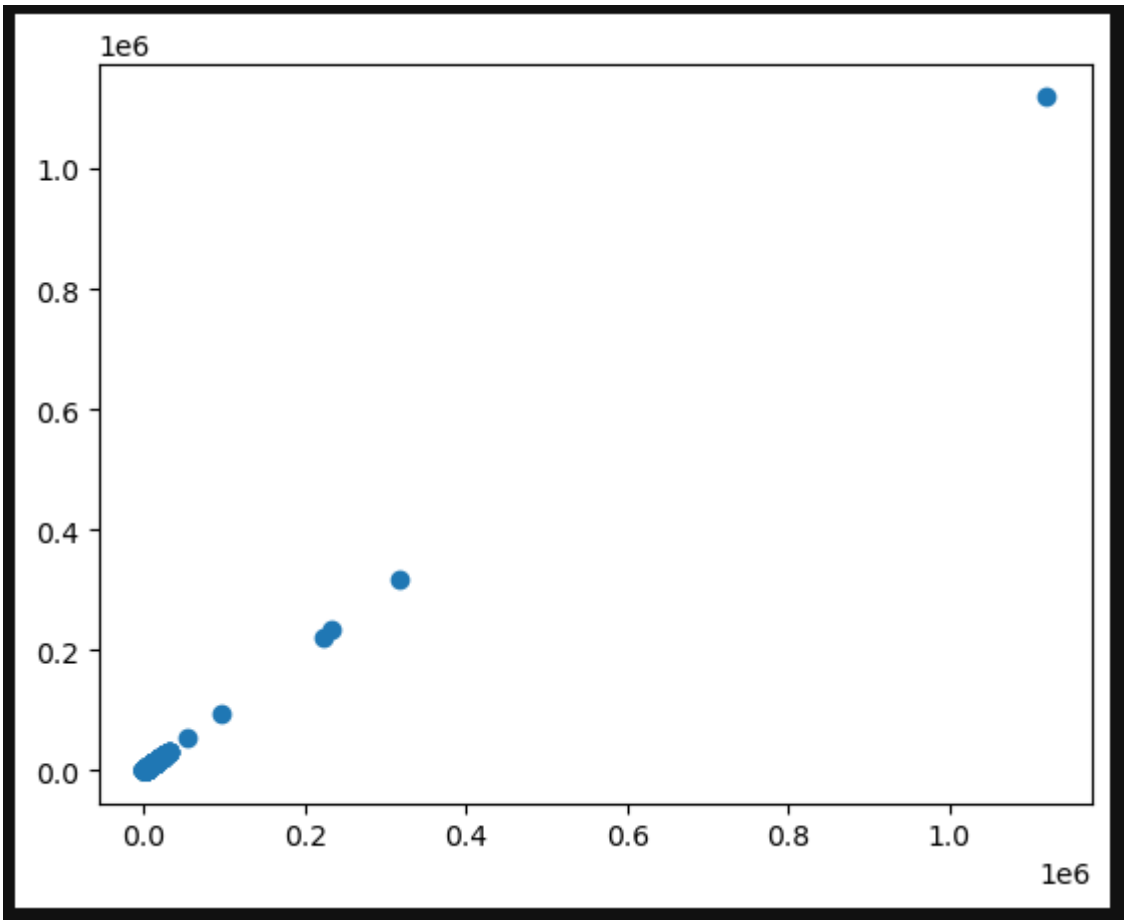
```
data = data.join(pd.get_dummies(data.city, prefix='city')).drop(['city'], axis=1)
data.head()
```

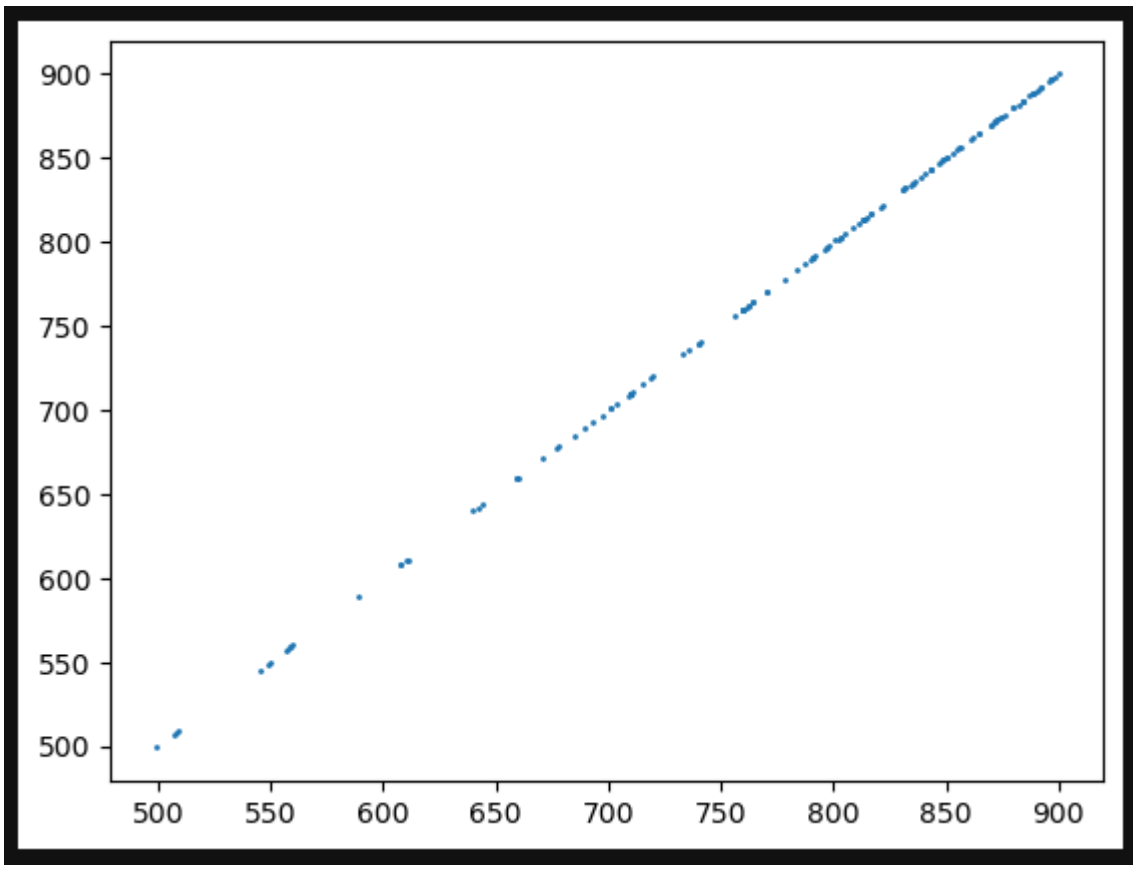
	area	rooms	bathroom	parking spaces	floor	animal	furniture	hoa (R$)	rent amount (R$)	property tax (R$)	fire insurance (R$)	total (R$)	city_Belo Horizonte	city_Campinas	cit
0	70	2	1	1	7	acept	furnished	2065	3300	211	42	5618	0	0	
1	320	4	4	0	20	acept	not furnished	1200	4960	1750	63	7973	0	0	
2	80	1	1	1	6	acept	not furnished	1000	2800	0	41	3841	0	0	
3	51	2	1	0	2	acept	not furnished	270	1112	22	17	1421	0	0	
4	25	1	1	0	1	not acept	not furnished	0	800	25	11	836	0	0	

Para procurarmos por valores nulos, basta usar o método *info* do DataFrame, podemos notar no seu retorno que temos 10334 registros, e para cada coluna, temos 10334 não-nulos, sendo assim, não é necessária nenhuma ação de remoção de dados nulos:

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 10334 entries, 0 to 10691
Data columns (total 17 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   area                  10334 non-null  int64
 1   rooms                 10334 non-null  int64
 2   bathroom              10334 non-null  int64
 3   parking spaces        10334 non-null  int64
 4   floor                 10334 non-null  uint8
 5   animal                10334 non-null  uint8
 6   furniture             10334 non-null  uint8
 7   hoa (R$)              10334 non-null  int64
 8   rent amount (R$)      10334 non-null  int64
 9   property tax (R$)     10334 non-null  int64
 10  fire insurance (R$)   10334 non-null  int64
 11  total (R$)            10334 non-null  int64
 12  city_Belo Horizonte   10334 non-null  uint8
 13  city_Campinas         10334 non-null  uint8
 14  city_Porto Alegre     10334 non-null  uint8
 15  city_Rio de Janeiro   10334 non-null  uint8
 16  city_São Paulo        10334 non-null  uint8
dtypes: int64(9), uint8(8)
memory usage: 1.1 MB
```

Quanto a *outliers*, foram inspecionadas as classes *total* e *area*, em ambos os casos, o dataset foi colocado em um *scatter plot,* afim de identificar agrupamentos de *outliers* em ambas as extremidades do gráfico, em seguida, filtros foram criados para remover entradas a partir de um limiar especificado após inspeção dos gráficos. A seguir, este processo é exemplificado usando a coluna *total*:


1e6
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
1e6


900
850
800
750
700
650
600
550
500
500
550
600
650
700
750
800
850
900

```
data = data[data.total.between(750, 20000)]

data.sort_values(by=['total'], ascending=False)['total']
```

```
428     19990
5278    19940
9452    19940
3716    19890
4008    19860
        ...
3702      760
7553      760
216       760
960       760
4850      756
Name: total, Length: 10162, dtype: int64
```

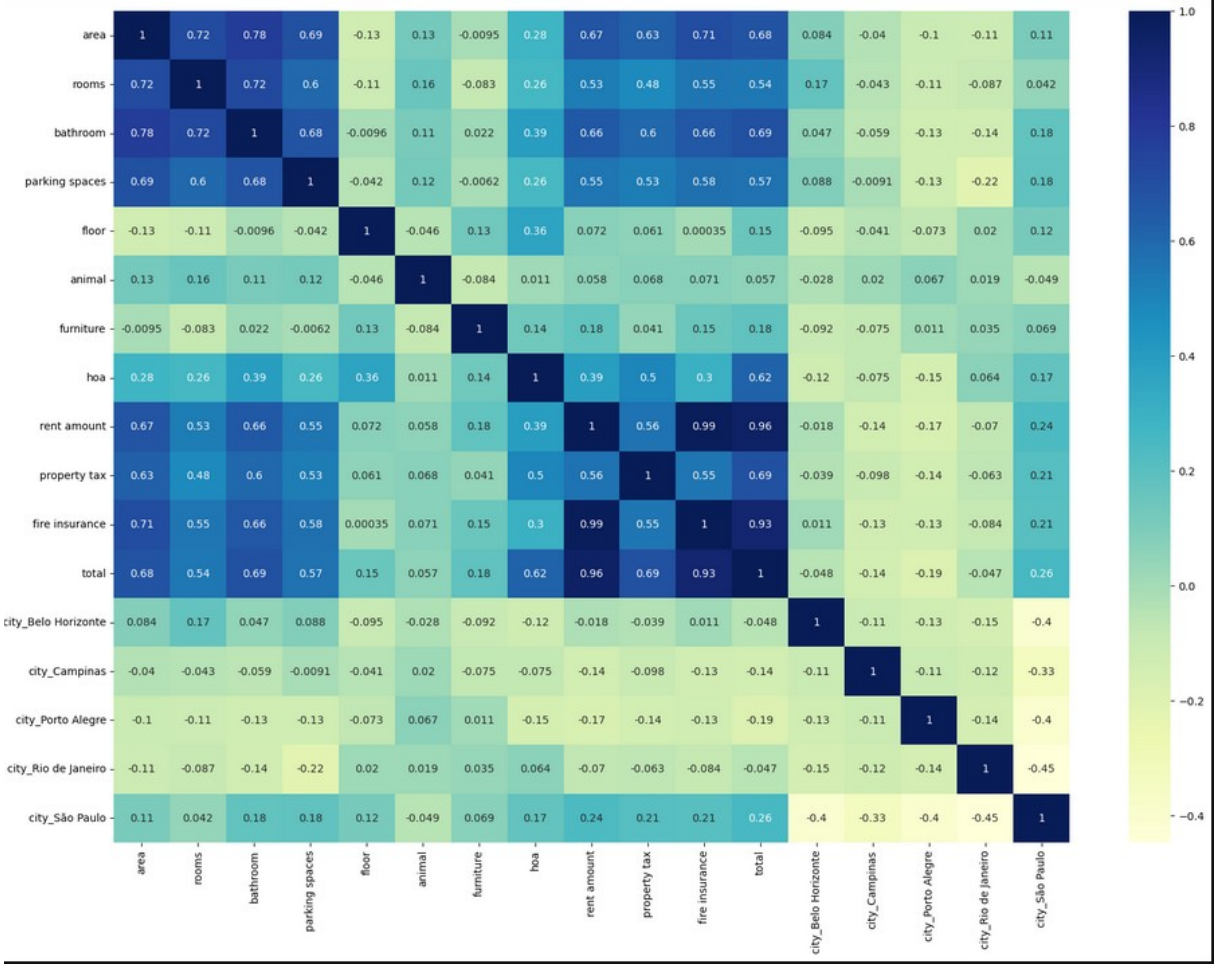
5. Análise e Exploração dos Dados

Nesta etapa, iremos usar o notebook “exploração-de-dados.ipynb”, também localizado na pasta “A3”, este notebook requer o arquivo csv com os dados já previamente tratados no notebook da etapa 4, este arquivo é o “tratado_houses_to_rent_v2.csv”, caso ele não esteja presente, basta abrir o notebook anterior e rodá-lo de novo.

Iremos explorar os dados predominantemente pelo uso de matrizes de correlação, assim, poderemos identificar as classes que melhor influenciam o preço do aluguel de um imóvel, indicado pela classe *total*. Usaremos o médoto *corr* para gerar esta matriz de correlação e em seguida, alimentaremos um mapa de calor com esta matriz para melhor visualização:

```
plt.figure(figsize=(20,14))
sb.heatmap(data.corr(), annot=True, cmap='YlGnBu')
```

```
<Axes: >
```



Podemos notar fortes correlações entre as colunas “hoa”, “rent amount”, “property tax” e “fire insurance” com a coluna “total”, isto faz sentido, pois a coluna “total” é a soma destas outras colunas. Como não temos mais nenhuma relação significativa usando estas colunas que também não possa ser inferida pela coluna “total”, podemos remover as colunas “hoa”, “rent amount”, “property tax” e “fire insurance” em prol de um dataset mais enxuto.

Podemos notar também fortes correlações entre as colunas “area”, “rooms”, “bathroom” e “parking spaces”, isto faz sentido quando falamos da área de um imóvel (mais cômodos pode implicar em maior área), mas não podemos concluir nada com relação as outras áreas no momento, por hora, iremos manter estas colunas intactas, no futuro, durante a fase de preparação de dados para os modelos de machine learning, poderemos usar estas colunas na criação de novos atributos, como por exemplo, uma razão entre número de quartos e número de banheiros.

Já que temos como foco a predição do valor do aluguel de um imóvel, vamos ordenar os valores de correlação com a coluna “total”:

```
data.corr()['total'].round(decimals=2).sort_values(ascending=False)
```

```
total                 1.00
bathroom              0.69
area                  0.68
parking spaces        0.57
rooms                 0.54
city_São Paulo        0.26
furniture             0.18
floor                 0.15
animal                0.06
city_Belo Horizonte  -0.05
city_Rio de Janeiro  -0.05
city_Campinas        -0.14
city_Porto Alegre    -0.19
Name: total, dtype: float64
```

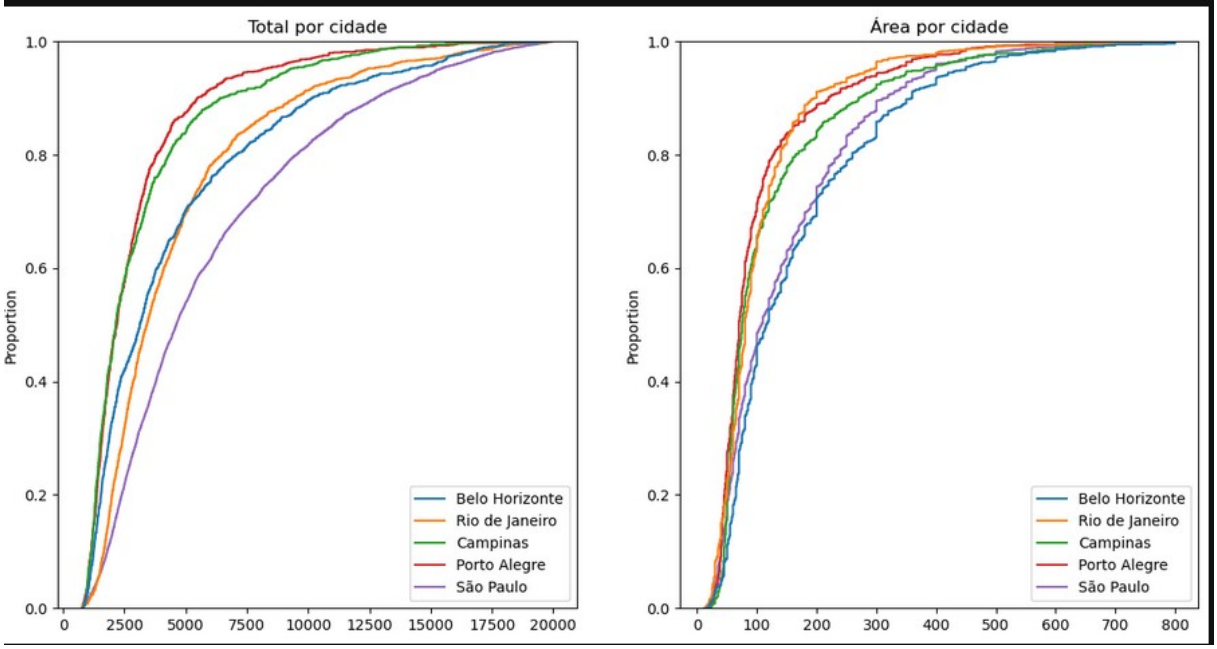
Podemos ver que imóveis localizados em São Paulo tendem a ser mais caros, devido a relação positiva com a coluna “city_São Paulo”, logo, podemos concluir que caso um usuário queira pagar menos aluguel e localidade não seja uma restrição, as cidades de Campinas e Porto Alegre tendem a ser mais favoráveis.

Também podemos notar, como fatores que tendem a aumentar o valor do aluguel de um imóvel, o andar onde o imóvel está localizado e se ele já está mobiliado, logo, caso um usuário deseje pagar menos pelo aluguel, imóveis de andar menor e que não estejam mobiliados tendem a ter melhores preços.

Para verificarmos mais a fundo nossa hipótese de que o aluguel em São Paulo é mais caro, podemos comparar o total e a área dos imóveis, agrupados por cidade, para isso, utilizaremos um gráfico *ecdfplot* da biblioteca Seaborn, este tipo de visualização nos ajudará a observar as proporções de valores das classes *total* e *area,* agrupados por cidade:

```python
def generate_chart_data(column: str):
    chart_data = {}
    cols = ['city_Belo Horizonte', 'city_Rio de Janeiro', 'city_Campinas', 'city_Porto Alegre', 'city_São Paulo']

    for col in cols:
        entries = data[data[col] == 1]
        chart_data[col.replace('city_', '')] = entries[column]

    return chart_data

plt.figure(figsize=(14,7))

plot1 = plt.subplot(1,2,1)
plot1.set_title('Total por cidade')
sb.ecdfplot(generate_chart_data('total'))

plot2 = plt.subplot(1,2,2)
plot2.set_title('Área por cidade')
sb.ecdfplot(generate_chart_data('area'))
```

```
<Axes: title={'center': 'Área por cidade'}, ylabel='Proportion'>
```



Podemos observar nos gráficos acima que, enquanto o custo de alguel em São Paulo é maior, a área dos imóveis não é necessariamente maior, pois podemos observar que Belo Horizonte possui imóveis com mais área, porém, o custo total do aluguel é mais baixo, dando mais credibilidade a nossa hipótese.

Podemos ir ainda mais a fundo e tirar a razão entre as médias aritméticas do total e da área do imóvel para as duas cidades:

```
data_sao_paulo = data[data['city_São Paulo'] == 1]
data_belo_horizonte = data[data['city_Belo Horizonte'] == 1]

ratio_sao_paulo = (data_sao_paulo.total.mean() / data_sao_paulo.area.mean()).round(decimals=2)
ratio_belo_horizonte = (data_belo_horizonte.total.mean() / data_belo_horizonte.area.mean()).round(decimals=2)

table_data = [
    ['São Paulo', ratio_sao_paulo],
    ['Belo Horizonte', ratio_belo_horizonte]
]

fig, ax = plt.subplots()
fig.patch.set_visible(False)
ax.axis('off')
ax.axis('tight')

table = ax.table(cellText=table_data, colLabels=[None, 'Razão total/area'], loc='center')
table.set_fontsize(14)
table.scale(1,4)
```

	Razão total/area
São Paulo	40.19
Belo Horizonte	27.31

Também podemos medir a frequência de imóveis mobiliados e não mobiliados entre as duas cidades, nota-se que Belo Horizonte também possui mais imóveis sem mobilia, o que pode contribuir para aluguéis mais baixos:

```python
def get_frequency(city: str, col: str):
    return (data[data[city] == 1][col].value_counts(normalize=True) * 100).round(decimals=2)

furniture_sp = get_frequency('city_São Paulo', 'furniture')
furniture_bh = get_frequency('city_Belo Horizonte', 'furniture')

table_data = [
    ['São Paulo', *[f'{f}%' for f in furniture_sp]],
    ['Belo Horizonte', *[f'{f}%' for f in furniture_bh]]
]

fig, ax = plt.subplots()
fig.patch.set_visible(False)
ax.axis('off')
ax.axis('tight')

table = ax.table(cellText=table_data, colLabels=[None, 'Sem mobilia', 'Mobiliado'], loc='center')
table.set_fontsize(14)
table.scale(1,4)
```

	Sem mobilia	Mobiliado
São Paulo	72.87%	27.13%
Belo Horizonte	86.56%	13.44%

Também podemos medir a frequência dos primeiros andares de cada cidade, isto nos mostra que, embora ambas as cidades tenham frequências similares de imóveis térreos, Belo Horizonte possui maior oferta de imóveis nos primeiros quatro andares, afirmando mais ainda nossa hipótese quanto ao custo benefício dos imóveis da cidade:

```
floor_sp = get_frequency('city_São Paulo', 'floor').sort_index()[:5]
floor_bh = get_frequency('city_Belo Horizonte', 'floor').sort_index()[:5]

table_data = [
    ['São Paulo', *[f'{f}%' for f in floor_sp]],
    ['Belo Horizonte', *[f'{f}%' for f in floor_bh]]
]

fig, ax = plt.subplots()
fig.patch.set_visible(False)
ax.axis('off')
ax.axis('tight')

table = ax.table(cellText=table_data, colLabels=[None, *range(5)] ,loc='center')
table.set_fontsize(14)
table.scale(2,4)
```

	0	1	2	3	4
São Paulo	27.2%	8.92%	6.07%	5.71%	4.92%
Belo Horizonte	26.88%	11.73%	12.07%	12.16%	9.76%