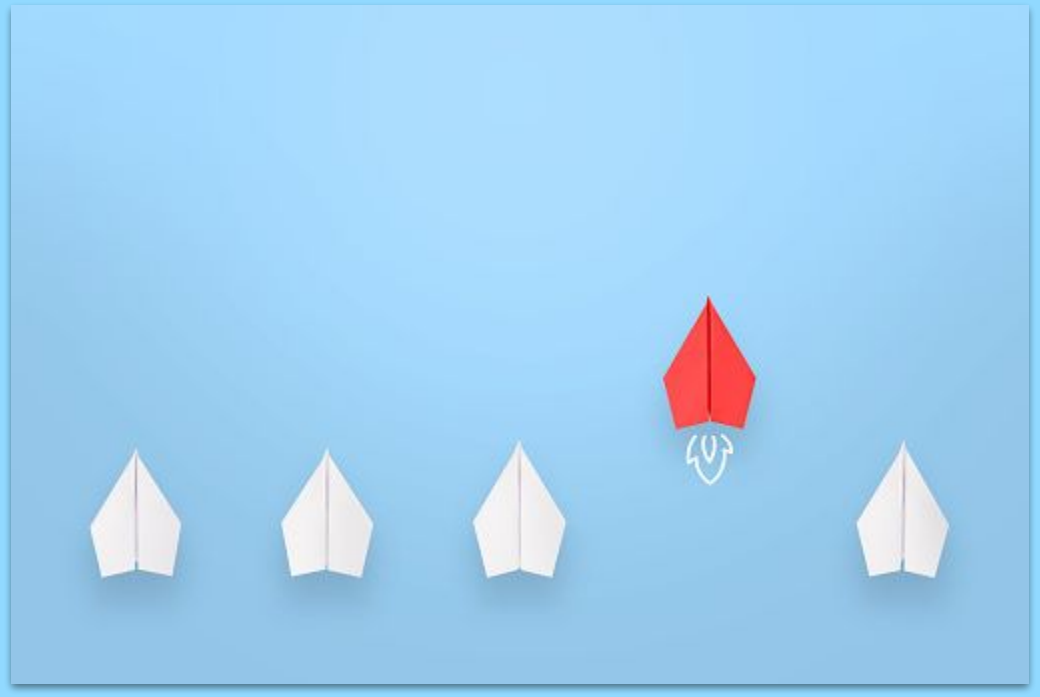


Algumas Provocações

4 verbos da inovação: rir, proibir, comprar e chorar


SEMANA DE 4 DIAS DE TRABALHO
Você apoia essa ideia?
Dom Seg Ter Qua Qui Sex Sab
Dom Seg Ter Qua Qui Sex Sab
Dom Seg Ter Qua Qui Sex Sab
@Robson Sampaio

UFRN
UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE

NE 4.0

Aprendizado de Máquina

K-Vizinhos Mais Próximos (KNN)

Ivanovitch Silva
ivanovitch.silva@ufrn.br

Assumindo que ....

Aula 05 Fundamentos de Aprendizado de Máquina PDF

- O que é aprendizado de máquina? Video
- Tipos de aprendizado de máquina Video
- Principais desafios
  - Variáveis, fluxos e controlando o caos Video
  - Instâncias de treinamento, desenvolvimento/validação e testes Video
  - Viés e variância Video

Você saberia estimar quanto custa um determinado veículo?

Ford ka 2010
40.000km
Ar condicionado
...

Você saberia estimar o custo de um projeto?

Você saberia estimar o aumento ou diminuição no ganho de produção caso a proporção de um insumo seja alterado?

O que esses exemplos têm em comum?

Problema de regressão!!!

http://insideairbnb.com/

Inside Airbnb
Adding data to the debate

Data ▾ About Support ▾ Organise ▾ Donate!

Donate! Support is essential to help Inside Airbnb add data to the debate. Donations fund the collection and hosting of data, development of analytic and activist tools, and help sustain the project.

How is Airbnb really being used in and affecting the neighbourhoods of your city?

Airbnb claims to be part of the "sharing economy" and disrupting the hotel industry. However, data shows that the majority of Airbnb listings in most cities are entire homes, many of which are rented all year round - disrupting housing and communities.

Browse the data for your city below, and see for yourself.

Explore the Data

Inside Airbnb has collected data on dozens of cities and countries around the world (more coming soon!).

Choose a city... »

or browse all cities

Selected Cities

Research and Reports

Inside Airbnb: Dallas
May 3, 2022

New data by Inside Airbnb for the City of Dallas shows concerning growth, entire homes and absentee "hosts" dominating the platform, and significant impact on rental housing.

O que é gentrificação e o que isso tem haver com a Airbnb?

THE HOUSING MARKET | MAY 17, 2022

New York Now Has More Airbnb Listings Than Apartments for Rent

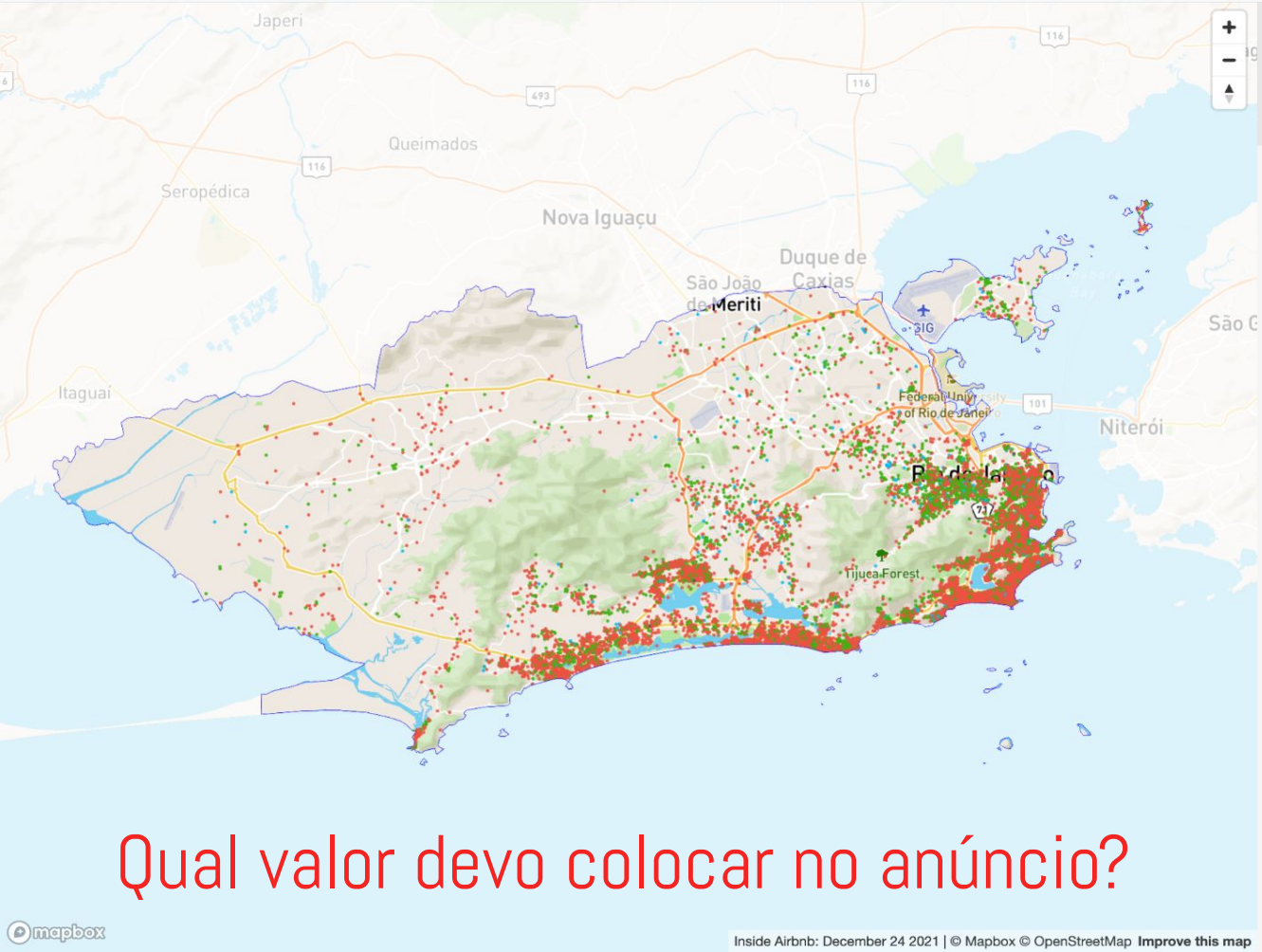
By Kim Velsey

Photo-Illustration: Curbed; Photo: Getty Images

MOST VIEWED STORIES

1. New York Now Has More Airbnb Listings Than Apartments for Rent
2. The Nolitafication of Dimes Square
3. A Fidi Loft With Room for the Pokémon: Before and After Photos
4. Century 21 Lives!
5. The Ugliest Divorce in Manhattan Real Estate

https://www.curbed.com/2022/05/new-york-more-airbnb-listings-apartments-rentals.html



Rio de Janeiro

Filter by:

Rio de Janeiro

24,549
out of **24,549** listings (100.0%)

Room Type

☐ Only entire homes/apartments

Airbnb hosts can list entire homes/apartments, private, shared rooms, and more recently hotel rooms.

Depending on the room type and activity, a residential airbnb listing could be more like a hotel, disruptive for neighbours, taking away housing, and illegal.

75.6%
entire homes/apartments

18,564 (75.6%)
entire home/apartments

5,415 (22.1%)
private rooms

494 (2.0%)
shared rooms

76 (0.3%)
hotel rooms

Activity

☐ Only **recent** and **frequently** booked

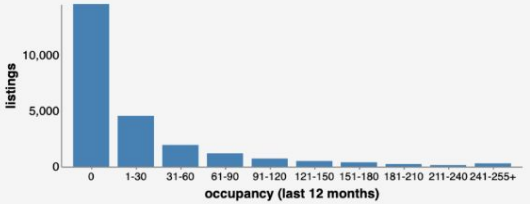
The minimum stay, price and number of reviews have been used to estimate the the number of **nights booked** and the **income** for each listing, for the last 12 months.

Is the home, apartment or room rented frequently and displacing units of housing and residents? Does the income from Airbnb incentivise short-term rentals vs long-term housing?

23
average nights booked

R$1030
price/night

R$16,927
average income



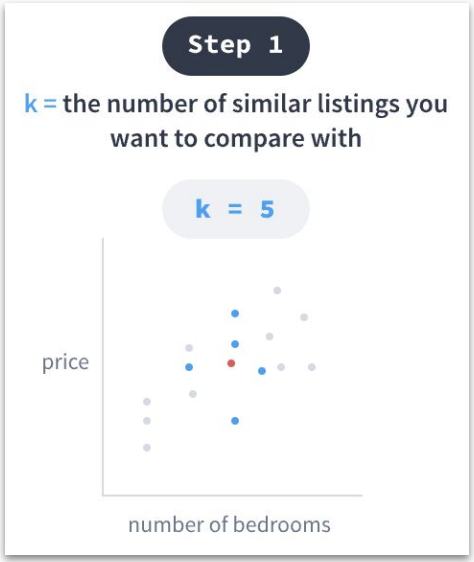
Qual valor devo colocar no anúncio?

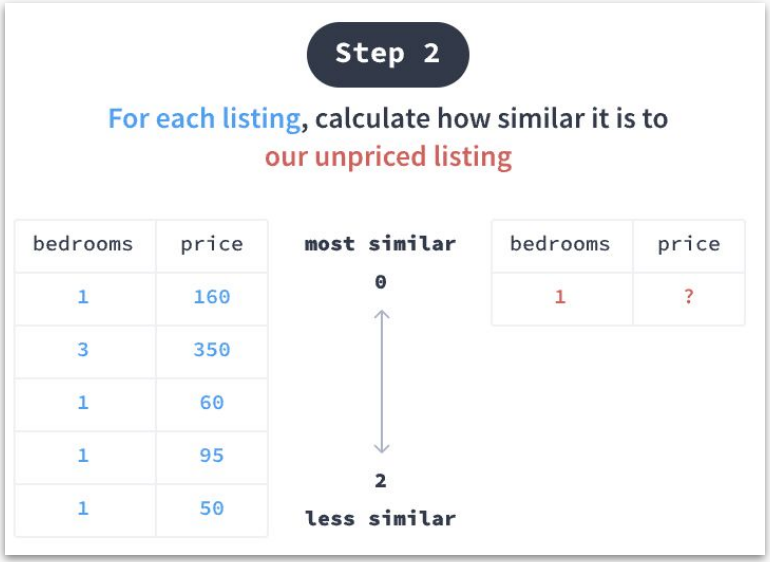
- **host_response_rate**: the response rate of the host
- **host_acceptance_rate**: number of requests to the host that convert to rentals
- **host_listings_count**: number of other listings the host has
- **latitude**: latitude dimension of the geographic coordinates
- **longitude**: longitude part of the coordinates
- **accommodates**: the number of guests the rental can accommodate
- **room_type**: the type of living space (Private room, Shared room or Entire home/apt
- **bedrooms**: number of bedrooms included in the rental
- **bathrooms**: number of bathrooms included in the rental
- **beds**: number of beds included in the rental
- **price**: nightly price for the rental
- **minimum_nights**: minimum number of nights a guest can stay for the rental
- **maximum_nights**: maximum number of nights a guest can stay for the rental
- **number_of_reviews**: number of reviews that previous guests have left

24k linhas

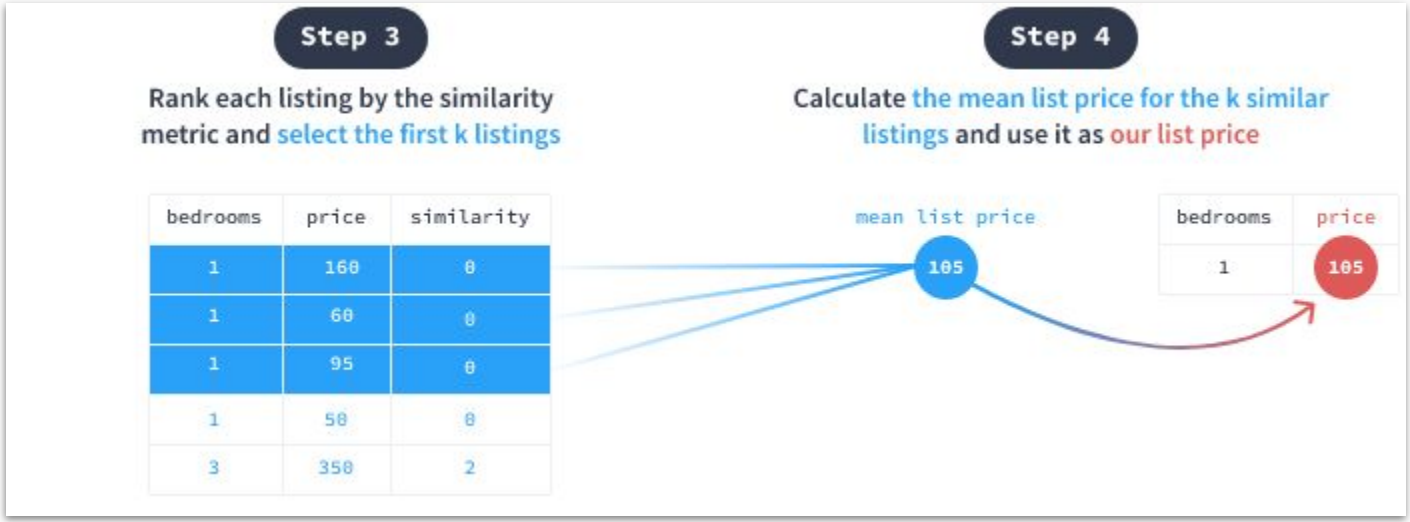
74 cols

Como funciona o KNN?





Como funciona o KNN?



Como encontrar matematicamente essa similaridade ("distância") entre dois anúncios?

Distância euclidiana

32 línguas

Origem: Wikipédia, a enciclopédia livre.

Em matemática, **distância euclidiana** (ou **distância métrica**) é a distância entre dois pontos, que pode ser provada pela aplicação repetida do teorema de Pitágoras. Aplicando essa fórmula como distância, o espaço euclidiano torna-se um espaço métrico.

Índice [esconder]

Definição

A distância euclidiana entre os pontos $P = (p_1, p_2, \ldots, p_n)$ e $Q = (q_1, q_2, \ldots, q_n)$, num espaço euclidiano n-dimensional, é definida como:

$$\sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \cdots + (p_n - q_n)^2} = \sqrt{\sum_{i=1}^{n} (p_i - q_i)^2}.$$

Um exemplo univariado

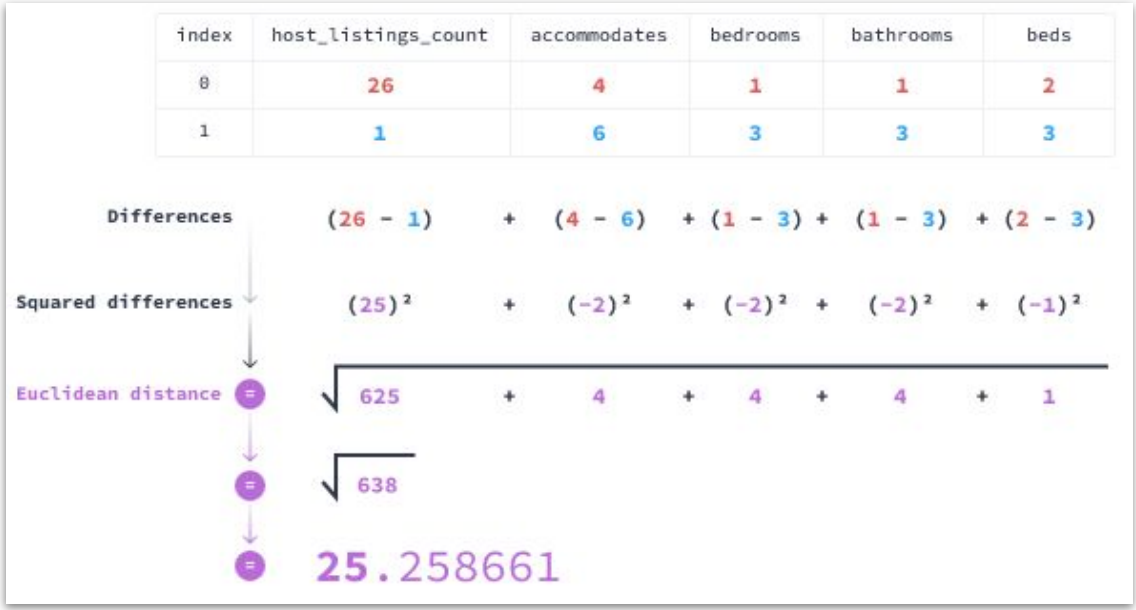
our listings

accommodates
8

rio listings

index	accommodates	distance	calculated distance
0	4	|4 - 8|	4
1	6	|6 - 8|	2
0	1	|1 - 8|	7
1	2	|2 - 8|	6

Um exemplo multivariado



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 24549 entries, 0 to 24548
Data columns (total 15 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   price                 24549 non-null  object
 1   host_response_rate    20668 non-null  object
 2   host_acceptance_rate  18902 non-null  object
 3   host_listings_count   24534 non-null  float64
 4   latitude              24549 non-null  float64
 5   longitude             24549 non-null  float64
 6   accommodates          24549 non-null  int64
 7   neighbourhood         13212 non-null  object
 8   room_type             24549 non-null  object
 9   bedrooms              23059 non-null  float64
 10  bathrooms             0 non-null      float64
 11  beds                  24202 non-null  float64
 12  minimum_nights        24549 non-null  int64
 13  maximum_nights        24549 non-null  int64
 14  number_of_reviews     24549 non-null  int64
dtypes: float64(6), int64(4), object(5)
memory usage: 2.8+ MB
```

Desafios?

Preparação da base

	price	host_response_rate	host_acceptance_rate	host_listings_count	latitude	longitude	accommodates
0	$350.00	100%	96%	2.0	-22.96599	-43.17940	5
1	$296.00	100%	100%	0.0	-22.98405	-43.20189	2
2	$387.00	80%	41%	3.0	-22.97735	-43.19105	3
3	$172.00	100%	NaN	1.0	-22.98839	-43.19232	2
4	$260.00	100%	97%	1.0	-22.98107	-43.19136	2

```
1 rio_listings.isnull().sum()
```

```
price                      0
host_response_rate      3881
host_acceptance_rate    5647
host_listings_count       15
latitude                   0
longitude                  0
accommodates               0
neighbourhood          11337
room_type                  0
bedrooms                1490
bathrooms              24549
beds                     347
minimum_nights             0
maximum_nights             0
number_of_reviews          0
dtype: int64
```

Preparação da base

Que problema é esse?

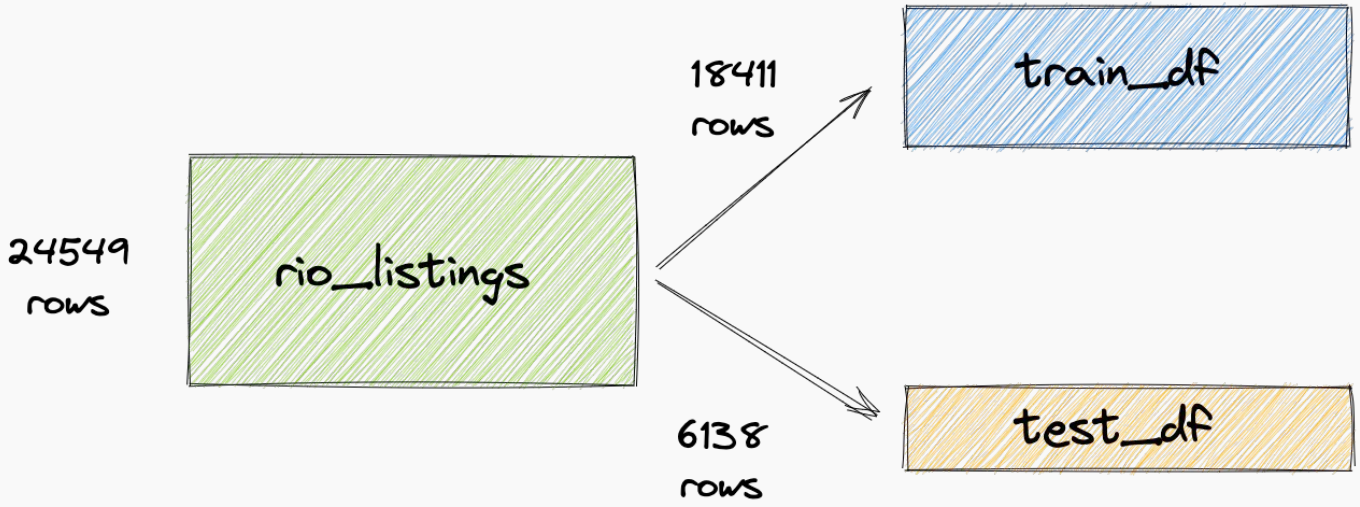
1.0 Introduction to K-Nearest Neighbors

- 1.1 Introduction to the data
- 1.3 K-nearest neighbors
- 1.4 Euclidean distance
- 1.5 Calculate distance for all observations
- 1.6 Randomizing, and sorting
- 1.7 Average price
- 1.8 Function to make predictions
- 1.9 Next steps

- Carregar a base
- Ler, verificar e executar os exercícios
- Entender as respostas
- Avaliar os valores das variáveis
- Provocações
  - Reproducibilidade (seed)
- Duração: 25 a 30 min

Como avaliar o nosso modelo?

Precisamos segregar a base!!!



Métricas de erro

Mean Absolute Error (erro médio absoluto)

$$MAE = \frac{|actual_1 - predicted_1| + |actual_2 - predicted_2| + \ \ldots + |actual_n - predicted_n|}{n}$$

Mean Squared Error (erro médio quadrático)

$$MSE = \frac{(actual_1 - predicted_1)^2 + (actual_2 - predicted_2)^2 + \ \ldots + (actual_n - predicted_n)^2}{n}$$

Root Mean Squared Error (raiz quadrada do erro médio quadrático)

$$RMSE = \sqrt{MSE}$$

2.0 Evaluating Model Performance

2.1 Testing quality of predictions

2.2 Error Metrics

2.3 Mean Squared Error

2.4 Training another model

2.5 Root Mean Squared Error

- Perceber a divisão treinamento vs teste
- A execução é rápida? Por que?
- O erro foi baixo ou alto? Por que?
- Duração: 25 a 30min

Melhorando o modelo

Cenário multivariado

Desafios

- Variáveis categóricas
- Dados faltantes
- Variáveis numéricas que não são ordinais
- Variáveis difusas com problema

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 24549 entries, 3757 to 235
Data columns (total 15 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   host_response_rate    20668 non-null  object
 1   host_acceptance_rate  18902 non-null  object
 2   host_listings_count   24534 non-null  float64
 3   neighbourhood         13212 non-null  object
 4   latitude              24549 non-null  float64
 5   longitude             24549 non-null  float64
 6   room_type             24549 non-null  object
 7   accommodates          24549 non-null  int64
 8   bathrooms             0 non-null      float64
 9   bedrooms              23059 non-null  float64
 10  beds                  24202 non-null  float64
 11  price                 24549 non-null  float64
 12  minimum_nights        24549 non-null  int64
 13  maximum_nights        24549 non-null  int64
 14  number_of_reviews     24549 non-null  int64
dtypes: float64(7), int64(4), object(4)
memory usage: 3.0+ MB
```

Melhorando o modelo

Removendo variáveis (*features*)

```
drop_columns = ['room_type', 'neighbourhood',
                'latitude','longitude',
                'host_response_rate',
                'host_acceptance_rate',
                'host_listings_count']
rio_listings = rio_listings.drop(drop_columns,
                                 axis=1)
```

	room_type	neighbourhood	latitude	longitude	host_response_rate	host_acceptance_rate	host_listings_count
3757	Entire home/apt	NaN	-22.94613	-43.18182	100%	100%	4.0
15234	Entire home/apt	NaN	-22.97502	-43.18756	100%	100%	1.0
18723	Entire home/apt	NaN	-22.96965	-43.38803	100%	86%	2.0
20524	Entire home/apt	Copacabana, Rio de Janeiro, Brazil	-22.96822	-43.18333	0%	20%	0.0
12620	Entire home/apt	Rio de Janeiro, Brazil	-22.98209	-43.20406	100%	73%	3.0
...	...	...	...	...	...	...	...
10955	Entire home/apt	Rio de Janeiro, Brazil	-22.98474	-43.20802	90%	29%	1.0
17289	Private room	Rio de Janeiro , Rio de Janeiro, Brazil	-22.96789	-43.18720	59%	56%	4.0
5192	Private room	Rio de Janeiro, Brazil	-23.01546	-43.46930	NaN	NaN	2.0
12172	Entire home/apt	Pedra de Guaratiba, Rio de Janeiro, Brazil	-22.99348	-43.62161	NaN	NaN	1.0
235	Entire home/apt	Rio de Janeiro, Brazil	-22.98434	-43.20793	100%	33%	2.0

Melhorando o modelo

Dados faltantes

```
rio_listings.isnull().sum()
```

```
accommodates          0
bathrooms ❌       24549
bedrooms  ⚠️        1490
beds      ⚠️         347
price                 0
minimum_nights        0
maximum_nights        0
number_of_reviews     0
dtype: int64
```

```
rio_listings.isnull().sum()/rio_listings.shape[0]
```

```
accommodates          0.000000
bathrooms             1.000000
bedrooms              0.060695
beds                  0.014135
price                 0.000000
minimum_nights        0.000000
maximum_nights        0.000000
number_of_reviews     0.000000
dtype: float64
```

```
rio_listings.drop(['bathrooms'], axis=1)
```

```
rio_listings.dropna(axis=0)
```

Melhorando o modelo

Normalização de variáveis

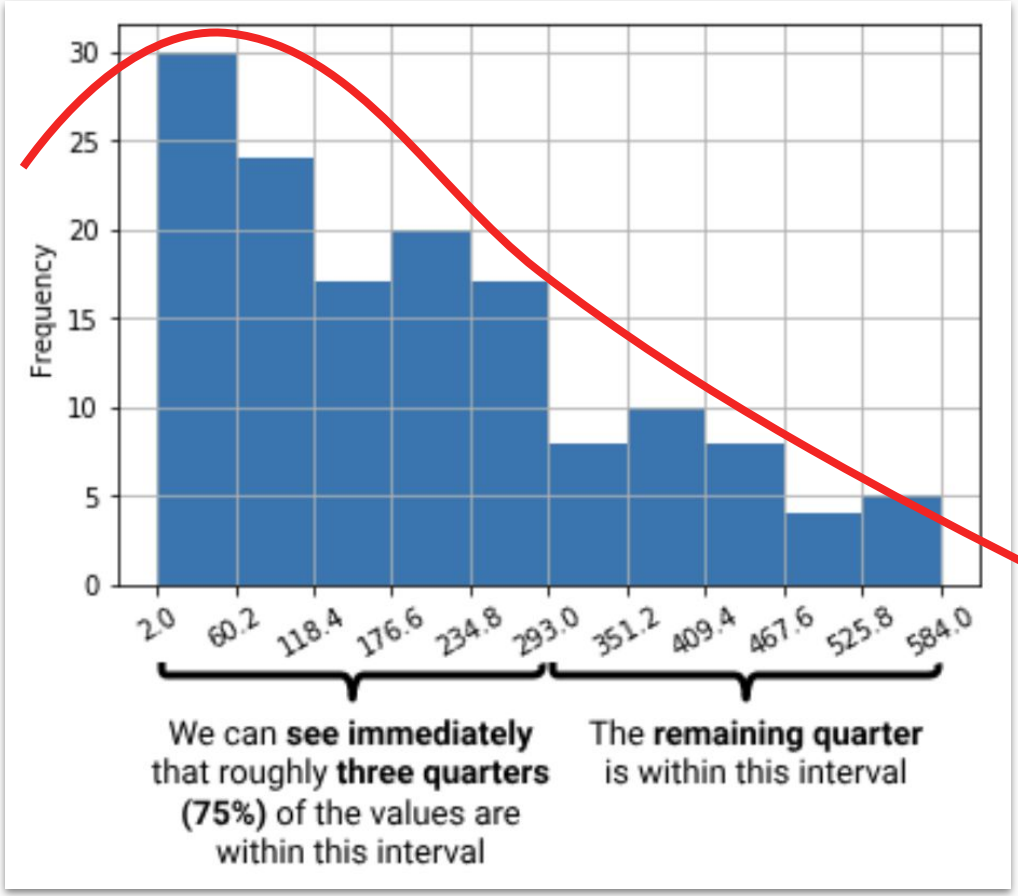
	accommodates	bedrooms	beds	price	minimum_nights	maximum_nights	number_of_reviews
3757	4	1.0	1.0	280.0	2	30	8
15234	6	3.0	5.0	1000.0	5	30	17
20524	6	2.0	3.0	180.0	2	30	0
12620	4	2.0	3.0	943.0	6	1125	8
4527	4	1.0	3.0	227.0	1	1125	0

	accommodates	bedrooms	beds	price	minimum_nights	maximum_nights	number_of_reviews
3757	4	1.0	1.0	280.0	2	30	8
15234	6	3.0	5.0	1000.0	5	30	17
20524	6	2.0	3.0	180.0	2	30	0
12620	4	2.0	3.0	943.0	6	1125	8
4527	4	1.0	3.0	227.0	1	1125	0

	accommodates	bedrooms	beds	price	minimum_nights	maximum_nights	number_of_reviews
3757	-0.101490	-0.647043	-0.728645	280.0	-0.144334	-0.909669	-0.180958
15234	0.697541	1.151181	1.004717	1000.0	-0.004319	-0.909669	0.086595
20524	0.697541	0.252069	0.138036	180.0	-0.144334	-0.909669	-0.418783
12620	-0.101490	0.252069	0.138036	943.0	0.042353	0.834585	-0.180958
4527	-0.101490	-0.647043	0.138036	227.0	-0.191006	0.834585	-0.418783

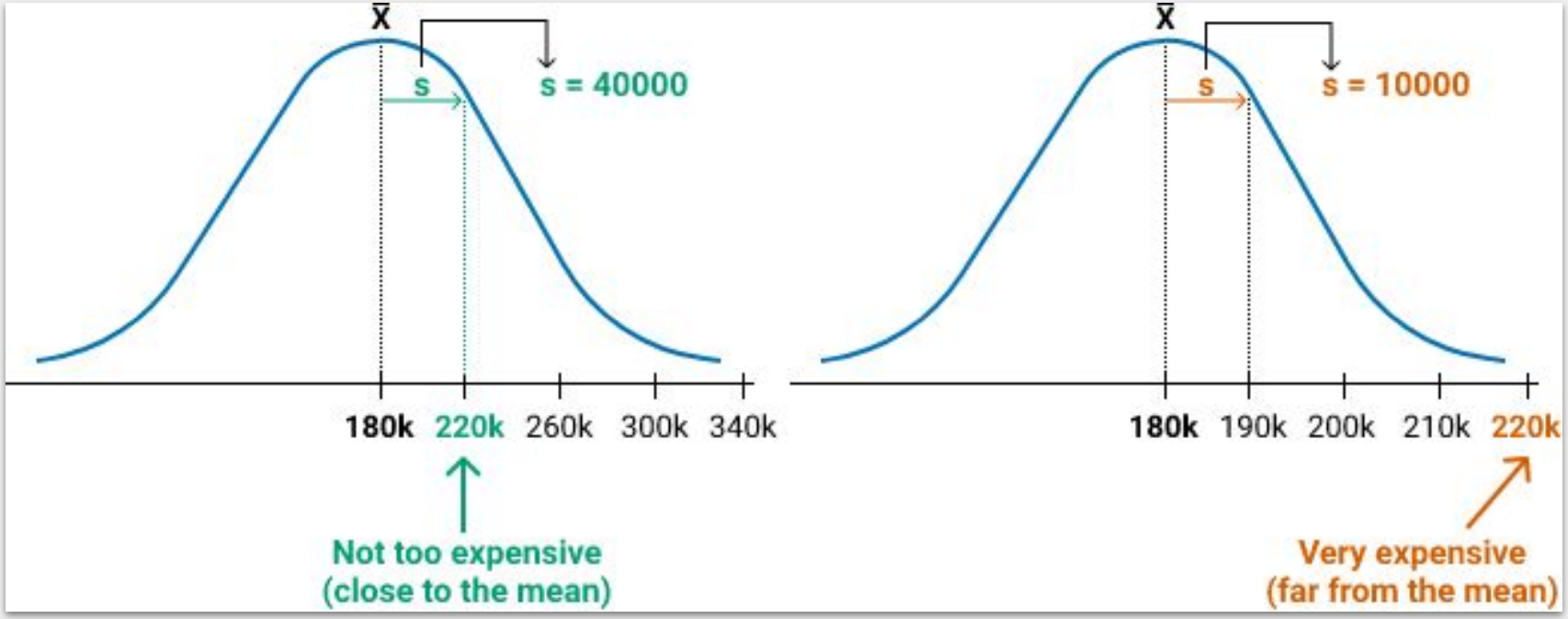
Vocês conhecem o Z-Score?

Revisão sobre Histograma

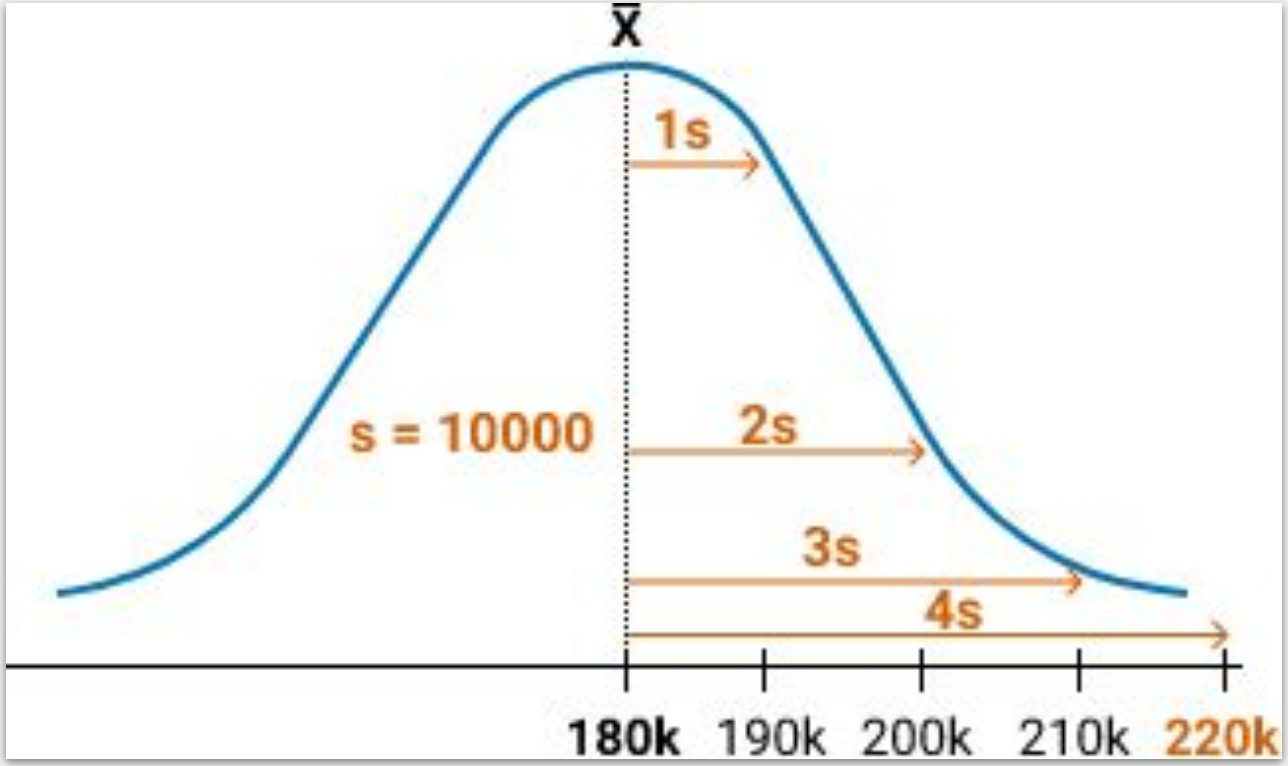


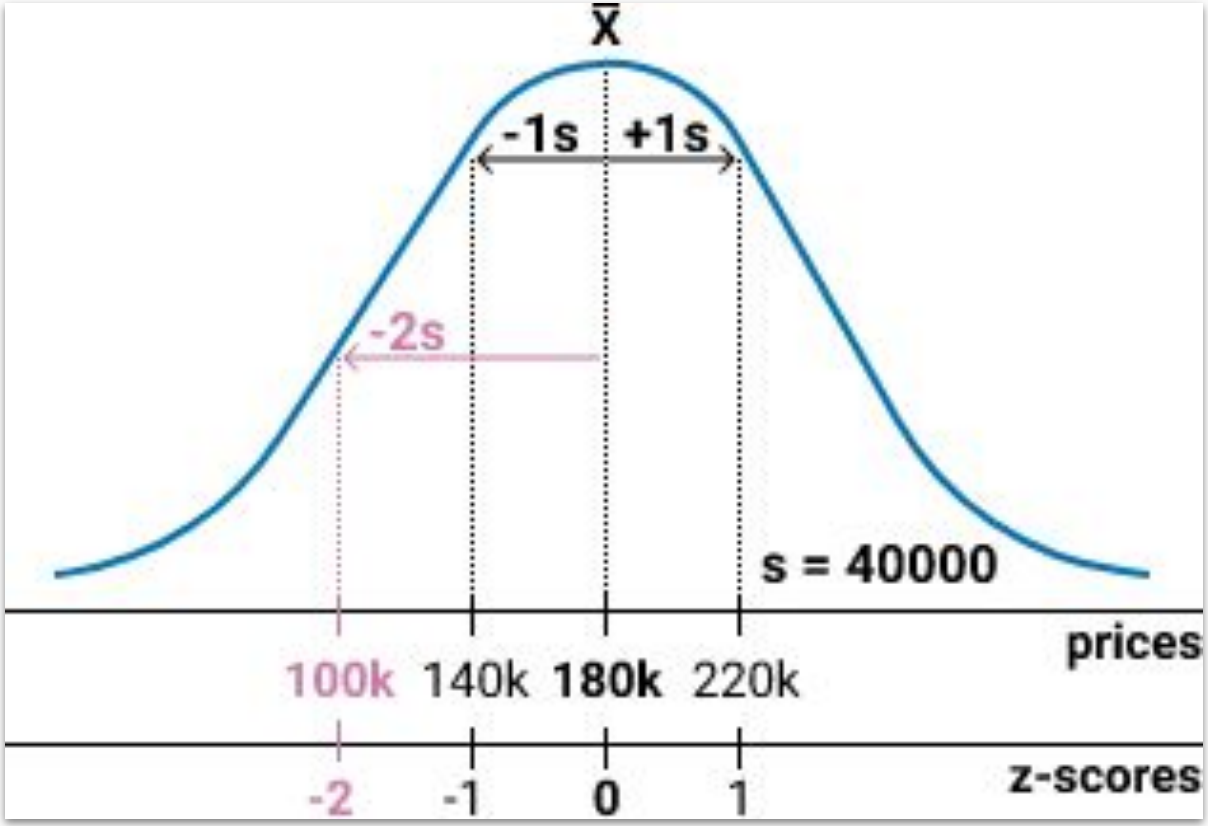

```
count     143.000000
mean      201.790210
std       153.381548
min         2.000000
25%        75.000000
50%       177.000000
75%       277.500000
max       584.000000
Name: PTS, dtype: float64
```

Duas curvas com a mesma média mas diferentes desvios padrões



Distância em unidades de desvio padrão para a média





Z-Score

$$z = \frac{x - \mu}{\sigma}$$

Usamos o Z-Score!!!

	accommodates	bedrooms	beds	price	minimum_nights	maximum_nights	number_of_reviews
3757	4	1.0	1.0	280.0	2	30	8
15234	6	3.0	5.0	1000.0	5	30	17
20524	6	2.0	3.0	180.0	2	30	0
12620	4	2.0	3.0	943.0	6	1125	8
4527	4	1.0	3.0	227.0	1	1125	0

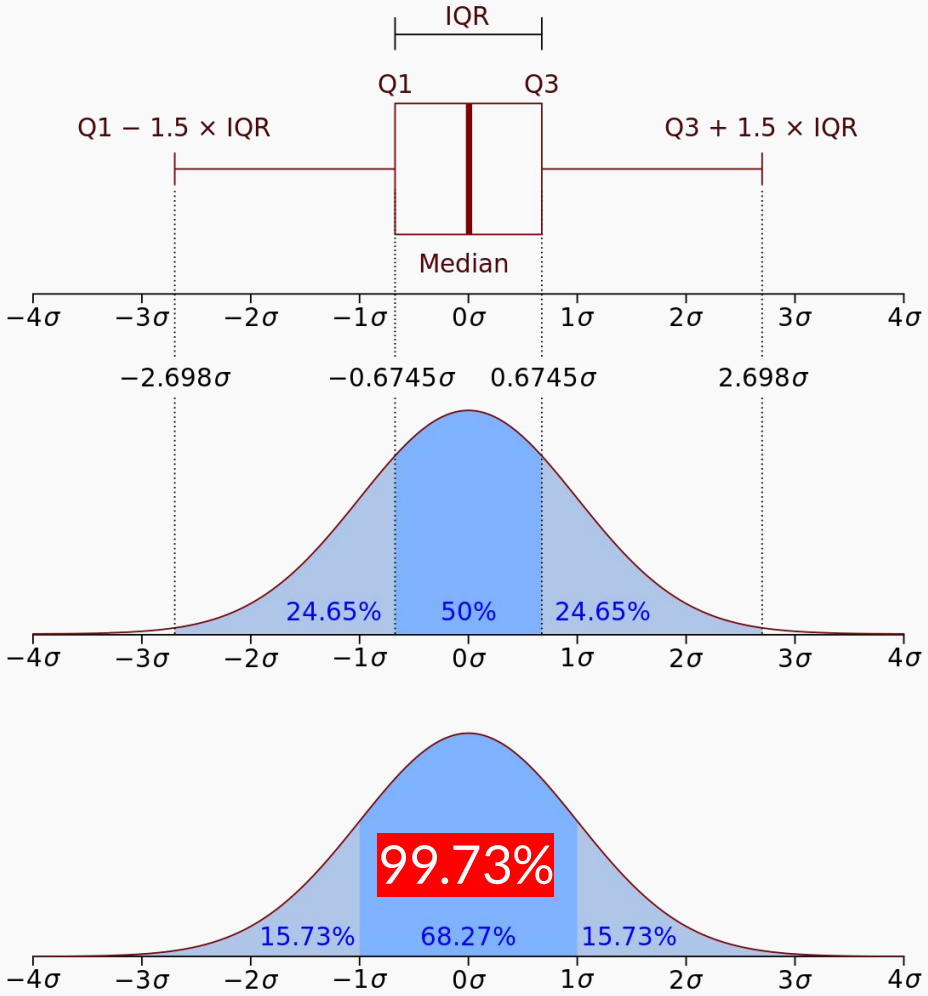
	accommodates	bedrooms	beds	price	minimum_nights	maximum_nights	number_of_reviews
3757	-0.101490	-0.647043	-0.728645	280.0	-0.144334	-0.909669	-0.180958
15234	0.697541	1.151181	1.004717	1000.0	-0.004319	-0.909669	0.086595
20524	0.697541	0.252069	0.138036	180.0	-0.144334	-0.909669	-0.418783
12620	-0.101490	0.252069	0.138036	943.0	0.042353	0.834585	-0.180958
4527	-0.101490	-0.647043	0.138036	227.0	-0.191006	0.834585	-0.418783

Como isso foi implementado?

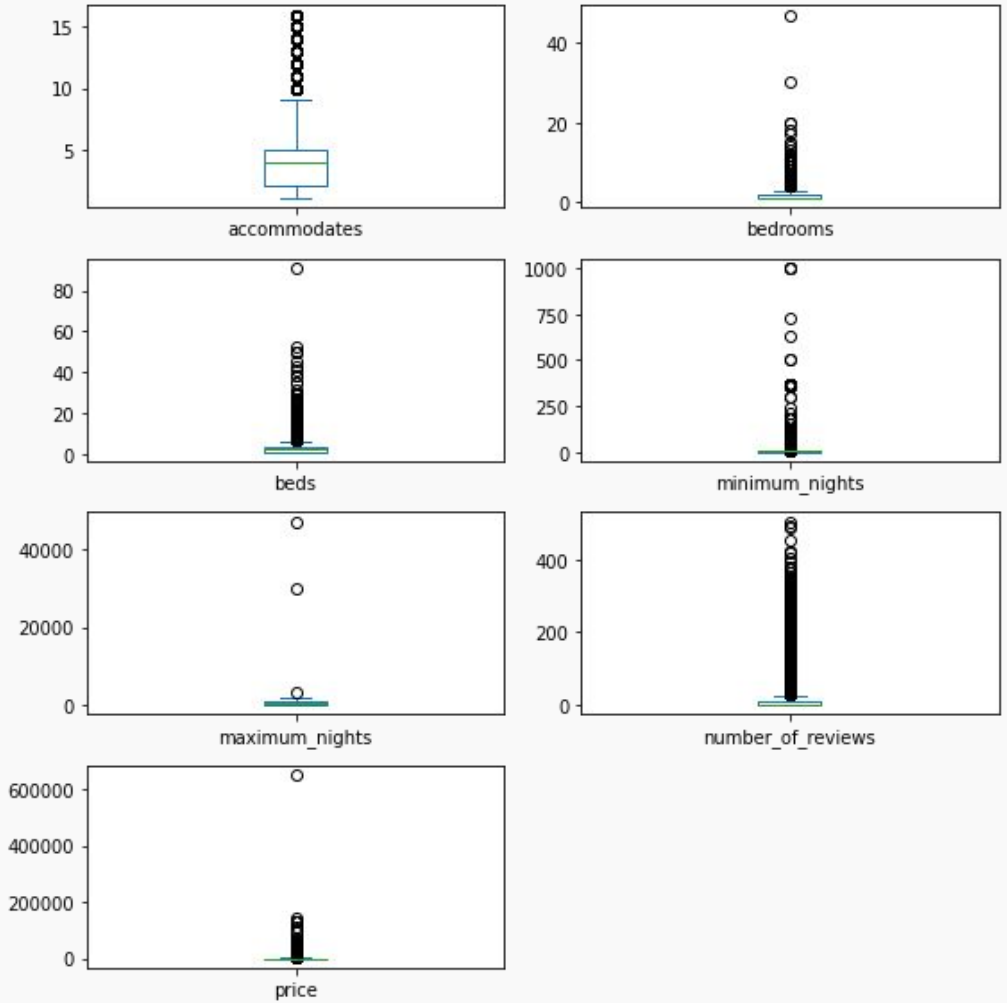
Z-Score

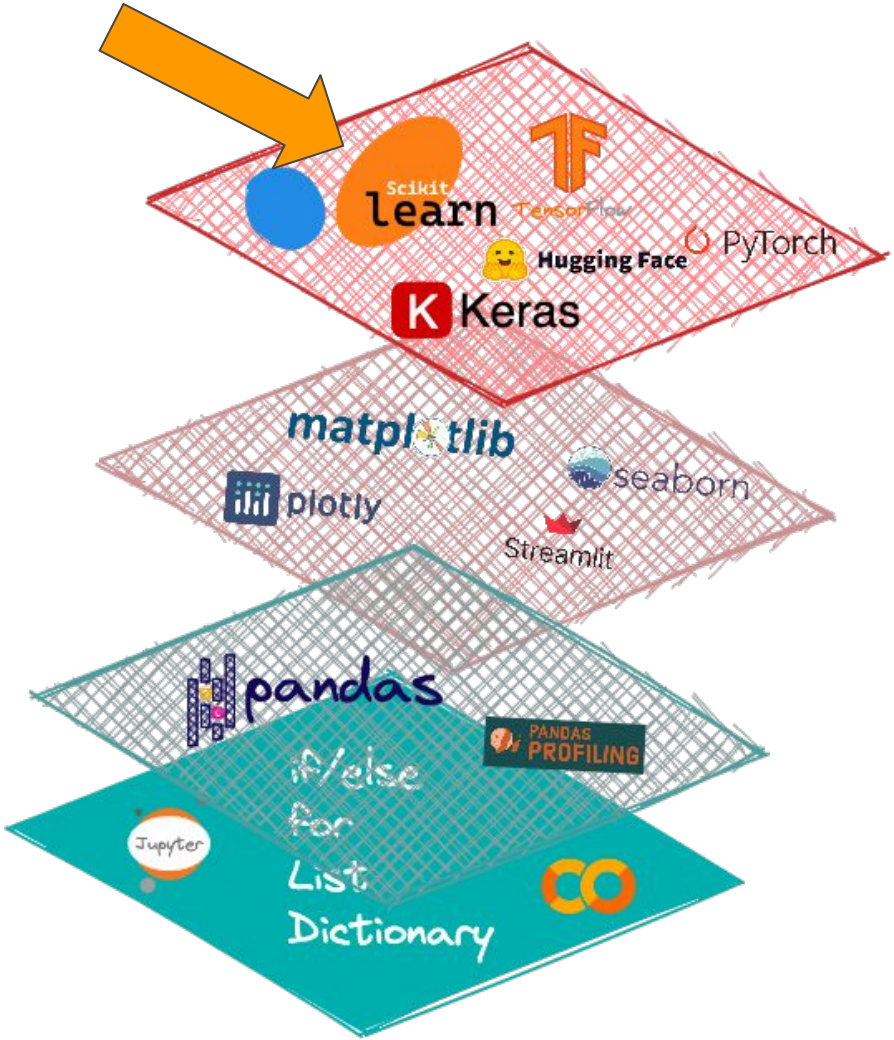
$$z = \frac{x - \mu}{\sigma}$$

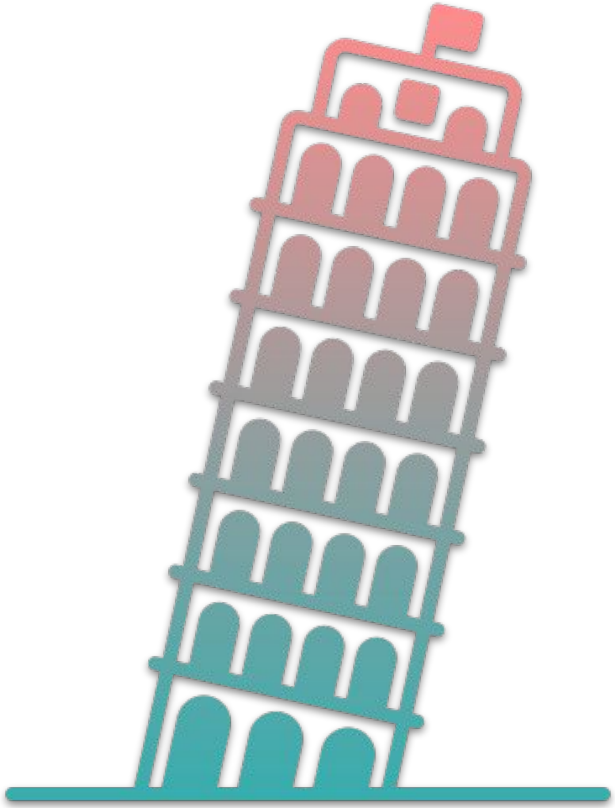
```
normalized_listings = (rio_listings - rio_listings.mean())/(rio_listings.std())
normalized_listings.loc[:,'price'] = rio_listings.loc[:,'price']
```



O Z-Score pode também ser usado para detectar valores atípicos (*outliers*)


15
10
5
accommodates
40
20
0
bedrooms
80
60
40
20
0
beds
1000
750
500
250
0
minimum_nights
40000
20000
0
maximum_nights
400
200
0
number_of_reviews
600000
400000
200000
0
price


Scikit
learn
TensorFlow
PyTorch
Hugging Face
Keras
matplotlib
plotly
seaborn
Streamlit
pandas
PANDAS PROFILING
if/else
for
List
Dictionary
Jupyter
CO



scikit-learn

Machine Learning in Python

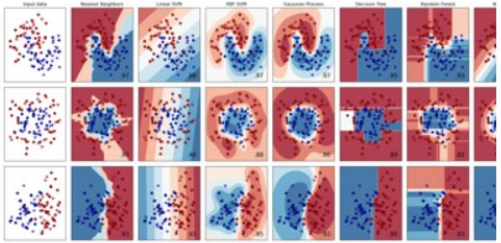
Getting Started | Release Highlights for 1.1 | GitHub

- Simple and efficient tools for predictive data analysis
- Accessible to everybody, and reusable in various contexts
- Built on NumPy, SciPy, and matplotlib
- Open source, commercially usable - BSD license

Classification

Identifying which category an object belongs to.

Applications: Spam detection, image recognition.
Algorithms: SVM, nearest neighbors, random forest, and more...

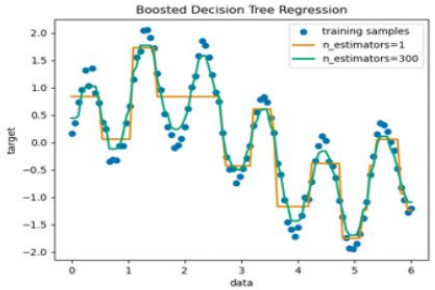


Examples

Regression

Predicting a continuous-valued attribute associated with an object.

Applications: Drug response, Stock prices.
Algorithms: SVR, nearest neighbors, random forest, and more...

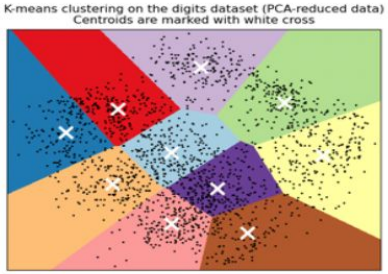


Examples

Clustering

Automatic grouping of similar objects into sets.

Applications: Customer segmentation, Grouping experiment outcomes
Algorithms: k-Means, spectral clustering, mean-shift, and more...



Examples

sklearn.neighbors.KNeighborsRegressor

class sklearn.neighbors.**KNeighborsRegressor**(*n_neighbors=5*, ***, *weights='uniform'*, *algorithm='auto'*, *leaf_size=30*, *p=2*, *metric='minkowski'*, *metric_params=None*, *n_jobs=None*) ¶ [source]

Regression based on k-nearest neighbors.

The target is predicted by local interpolation of the targets associated of the nearest neighbors in the training set.

Read more in the User Guide.

New in version 0.9.

Parameters:

n_neighbors : *int, default=5*
Number of neighbors to use by default for `kneighbors` queries.

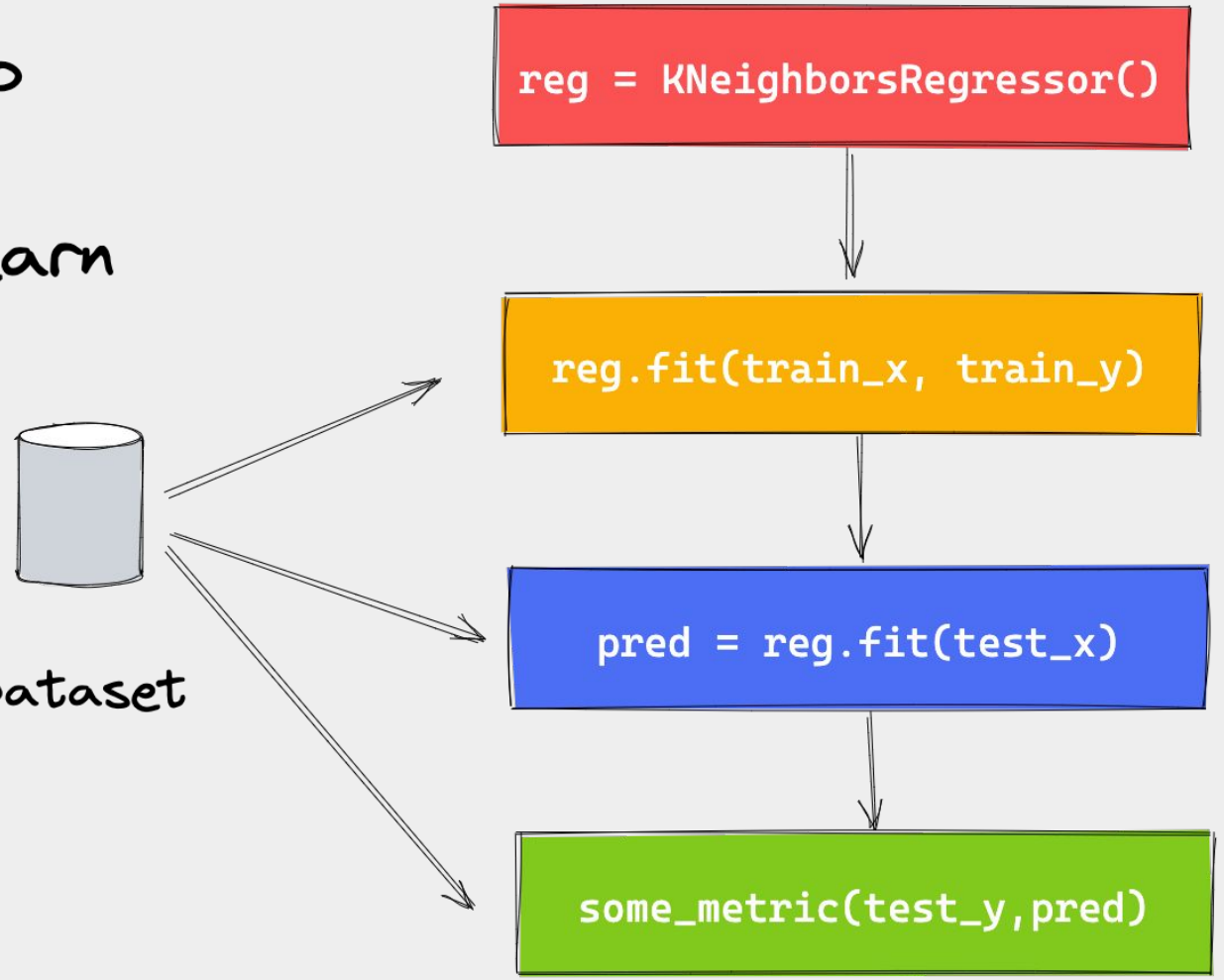
weights : *{'uniform', 'distance'} or callable, default='uniform'*
Weight function used in prediction. Possible values:

- 'uniform' : uniform weights. All points in each neighborhood are weighted equally.
- 'distance' : weight points by the inverse of their distance. in this case, closer neighbors of a query point will have a greater influence than neighbors which are further away.
- [callable] : a user-defined function which accepts an array of distances, and returns an array of the same shape containing the weights.

Uniform weights are used by default.

algorithm : *{'auto', 'ball_tree', 'kd_tree', 'brute'}, default='auto'*
Algorithm used to compute the nearest neighbors:

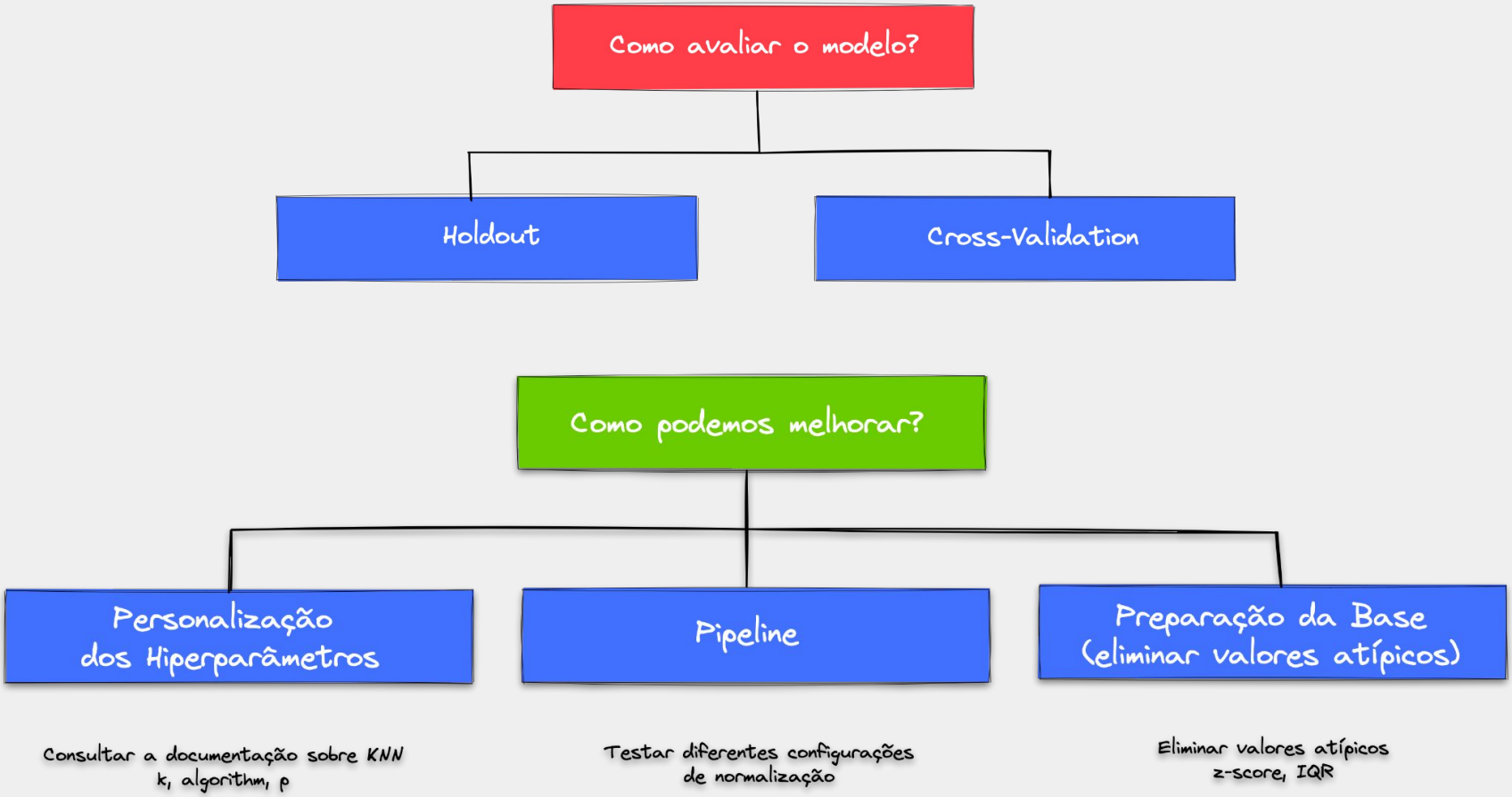
Um fluxo típico do Sckit-learn



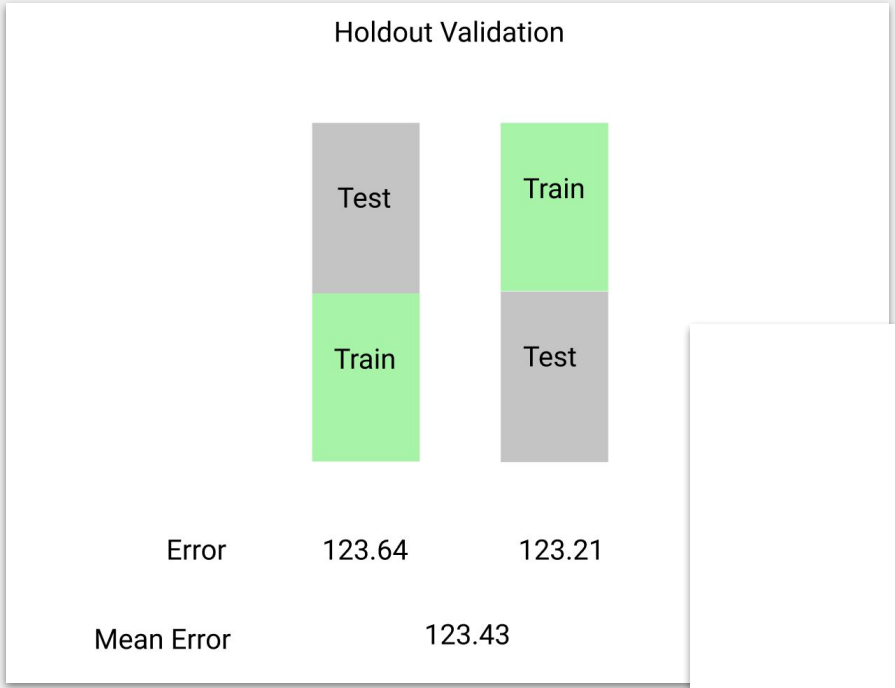
3.0 Multivariate K-Nearest Neighbors

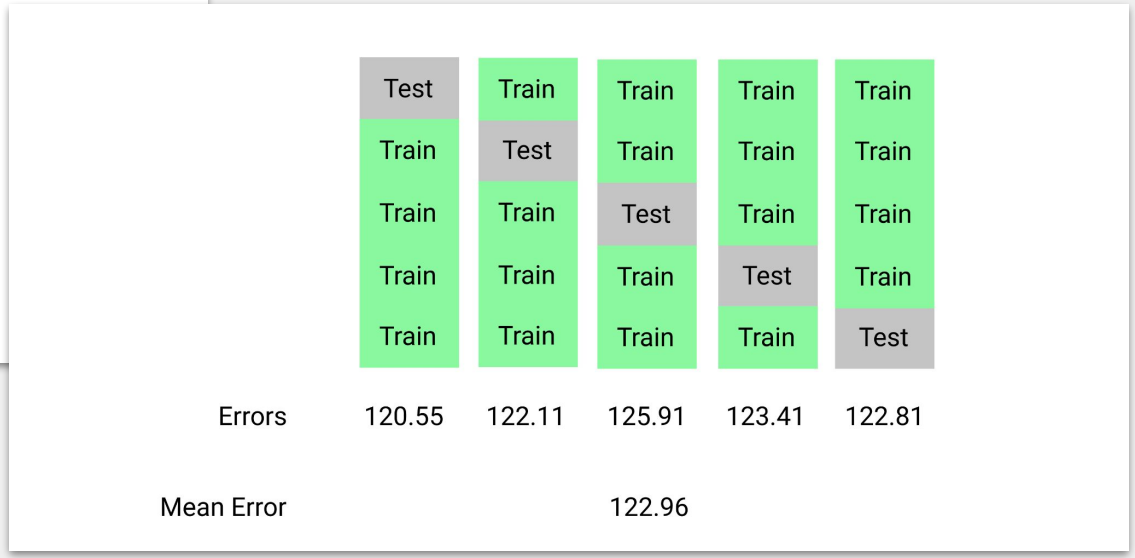
- 3.1 Recap
- 3.2 Removing features
- 3.3 Handling missing values
- 3.4 Normalize columns
- 3.5 Euclidean distance for multivariate case
- 3.6 Introduction to scikit-learn
- 3.7 Fitting a model and making predictions
- 3.8 Calculating MSE using Scikit-Learn

- Identificar a normalização da base
- Identificar a remoção das colunas e linhas
- Como a distância euclidiana é utilizada
- Os resultados foram satisfatórios? Como podemos melhorar?
- Duração: 25 a 30 min


Como avaliar o modelo?
Holdout
Cross-Validation
Como podemos melhorar?
Personalização dos Hiperparâmetros
Pipeline
Preparação da Base (eliminar valores atípicos)
Consultar a documentação sobre KNN
k, algorithm, p
Testar diferentes configurações de normalização
Eliminar valores atípicos
z-score, IQR

Cross-Validation

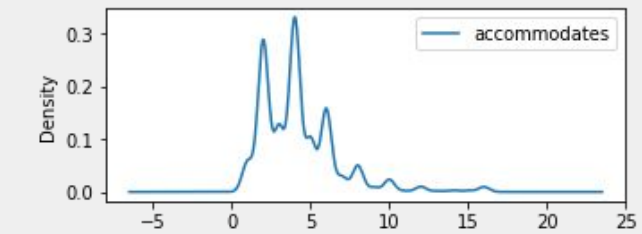


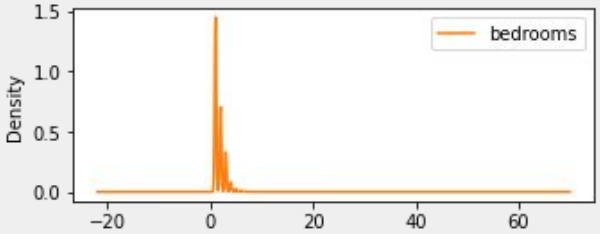


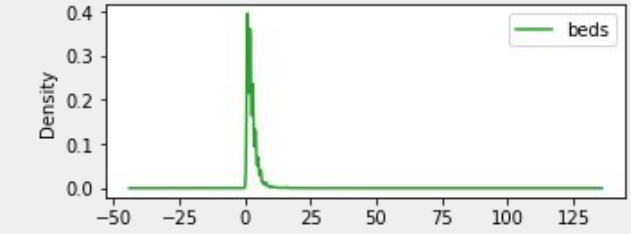
Utilizando o cross-validation para avaliar o modelo

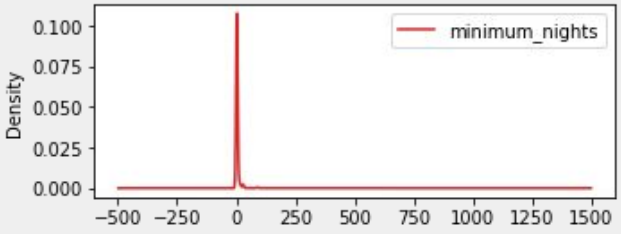
```
from sklearn.model_selection import cross_val_score, KFold

# kfold instance
kf = KFold(5, shuffle=True, random_state=1)

# knn model
model = KNeighborsRegressor()

# cross validation (knn,x,y,scoring,kfold)
mses = cross_val_score(model,
                       normalized_listings[["accommodates"]],
                       normalized_listings["price"],
                       scoring="neg_mean_squared_error",
                       cv=kf)

# root mean squared error
rmses = np.sqrt(np.absolute(mses))

# average error
avg_rmse = np.mean(rmses)
```

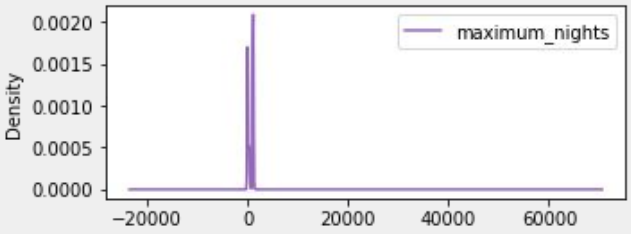
```
[10166.40476622  3845.96656612  3735.46970741
2862.45030037 3829.44837424]
4887.94794287297
```

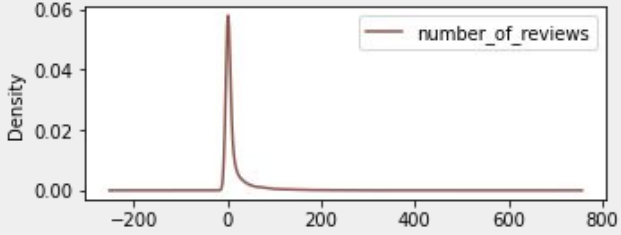


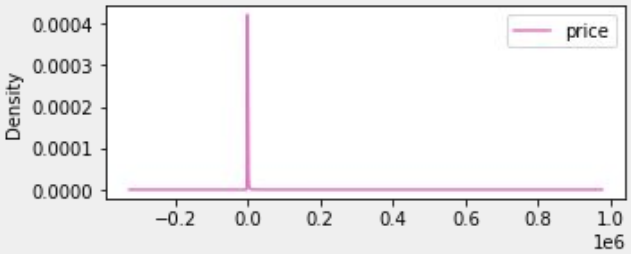






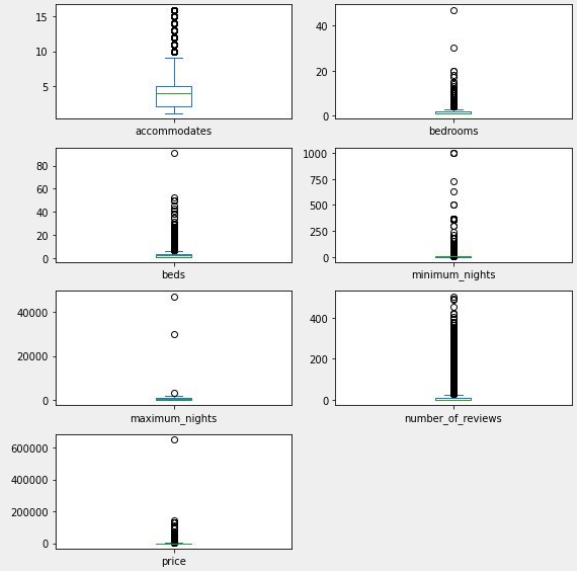


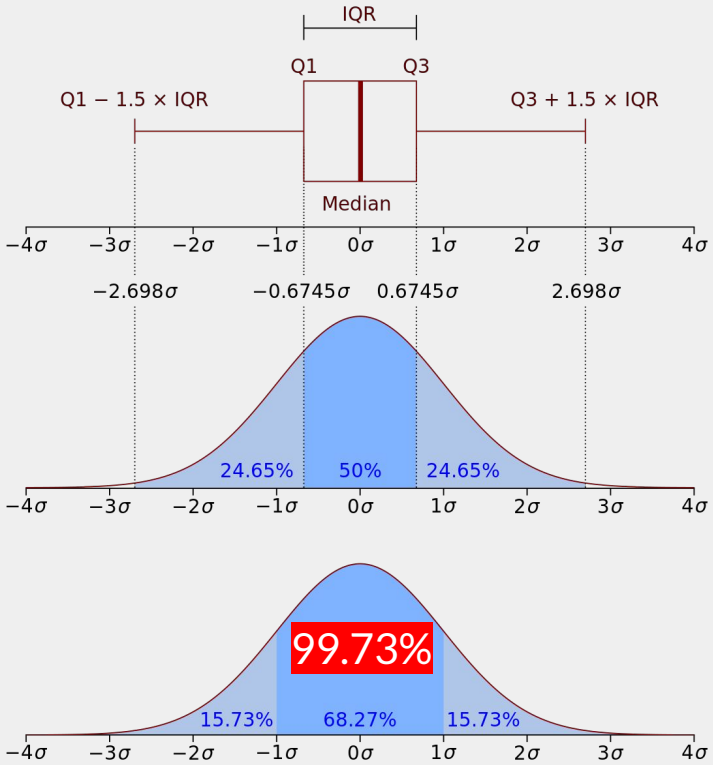




Onde podemos melhorar?

Valores atípicos são um problema real aqui!!!!



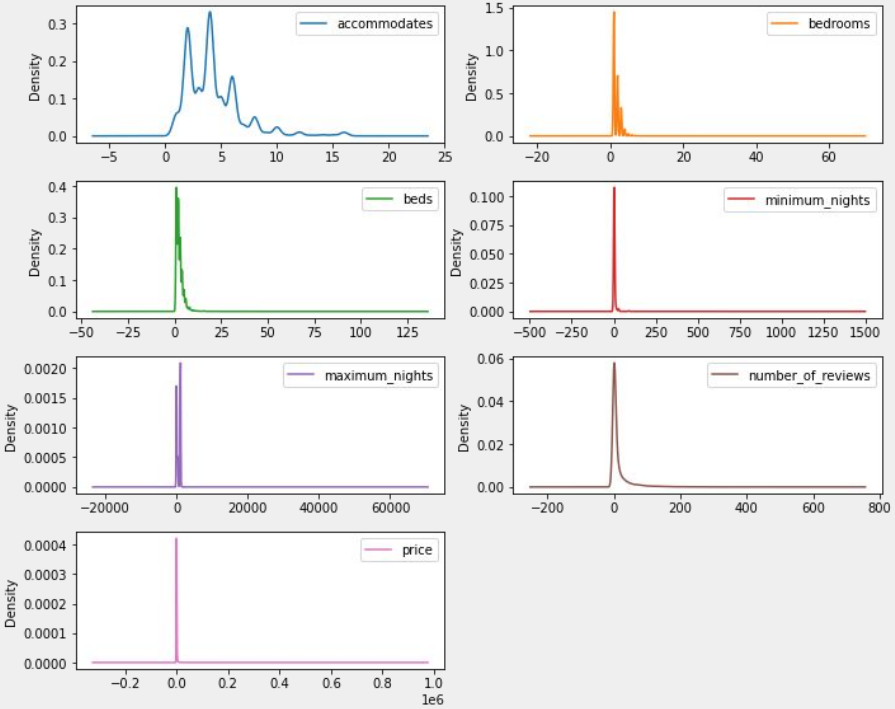


Como definir um valor atípico usando o Z-Score?

```
# remove outliers
rio_z_scored = rio_z_scored[(rio_z_scored < 2.698).all(axis=1)
                & (rio_z_scored > -2.698).all(axis=1)]
```

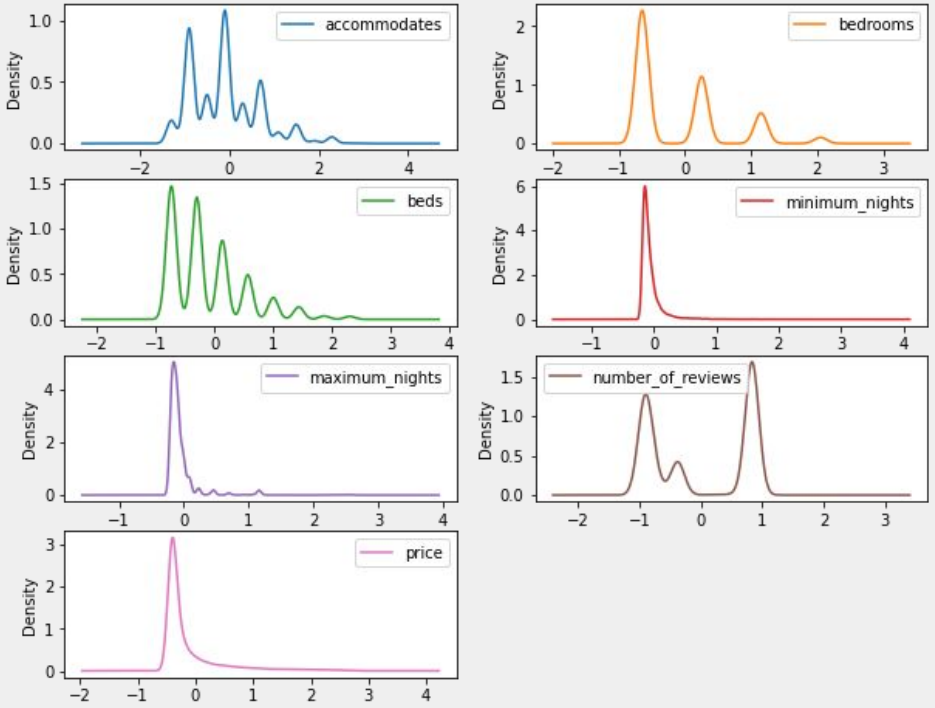
Antes da remoção

(22757, 7)



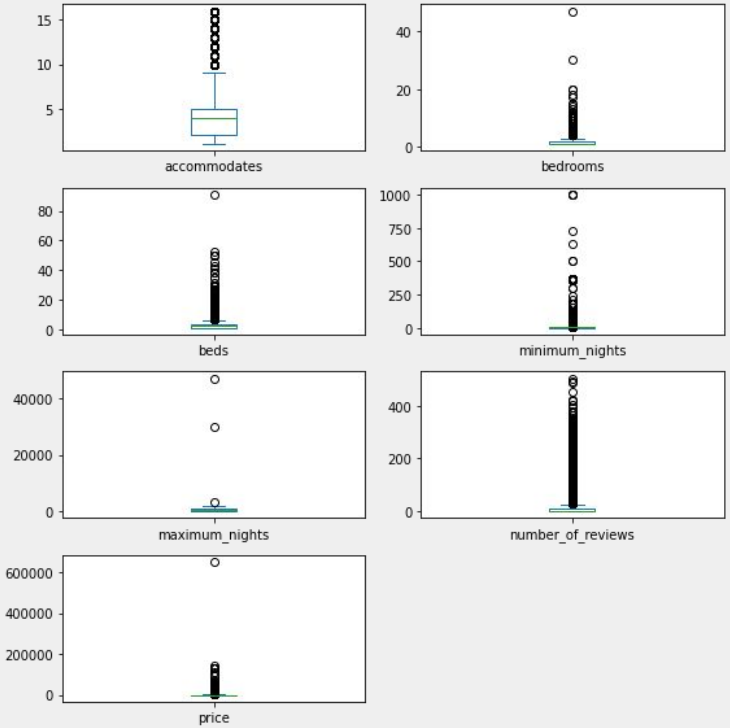
Após a remoção

(21077,7)



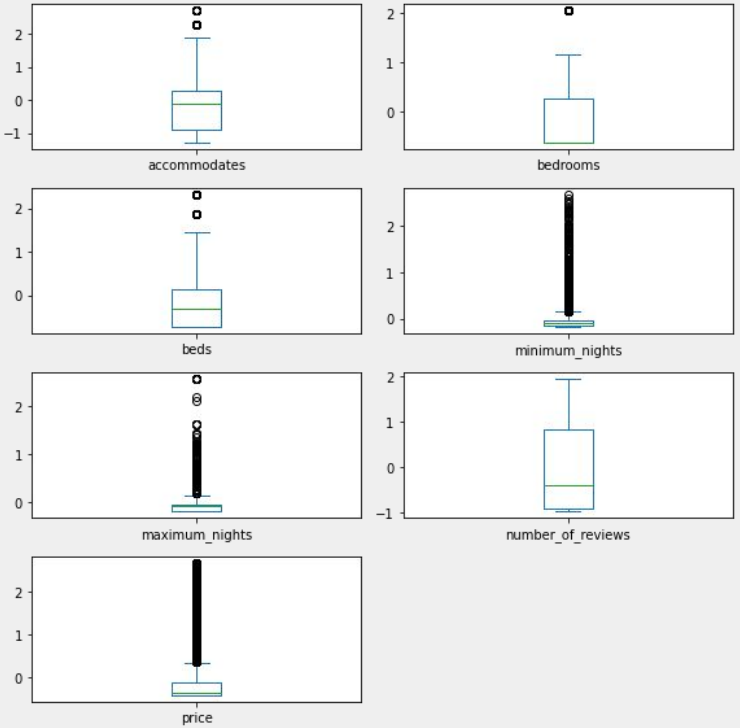
Antes da remoção

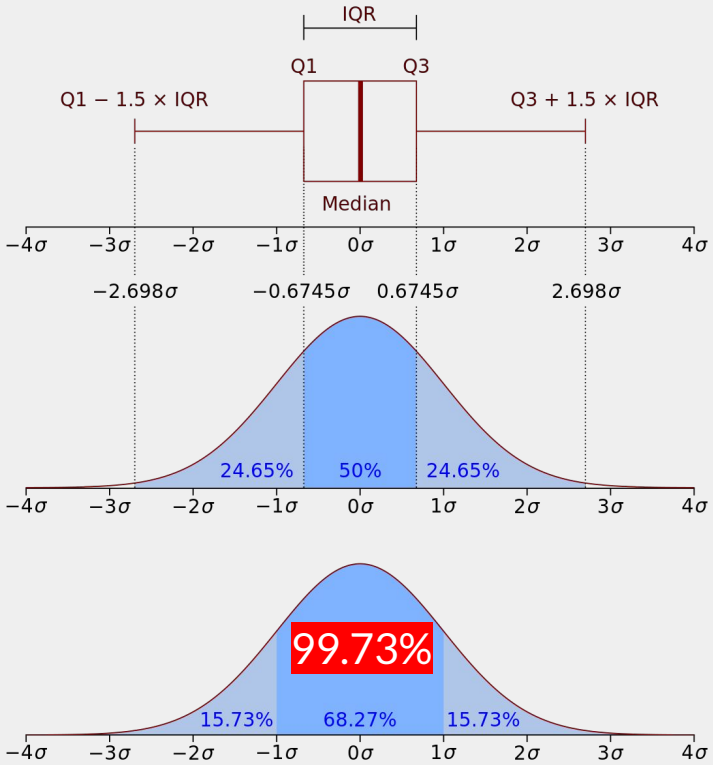
(22757, 7)



Após a remoção

(21077,7)





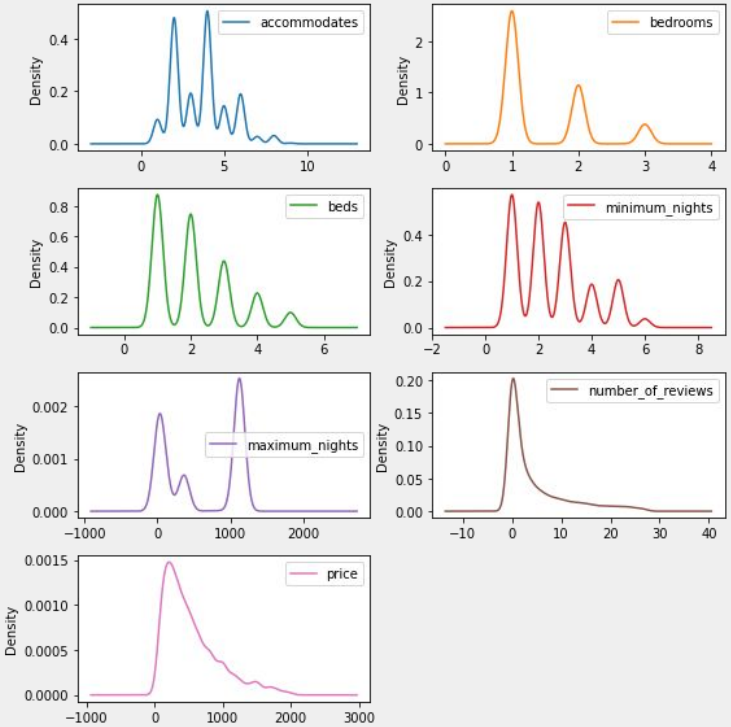
Como definir um valor atípico usando o IQR?

```
rio_iqr = rio_listings[target_columns].copy()
Q1 = rio_iqr.quantile(0.25)
Q3 = rio_iqr.quantile(0.75)
IQR = Q3 - Q1
low = Q1 - 1.5 * IQR
up = Q3 + 1.5 * IQR
```

```
# remove outliers
rio_iqr = rio_iqr[((rio_iqr > low).all(axis=1) & (rio_iqr < up).all(axis=1))]
```

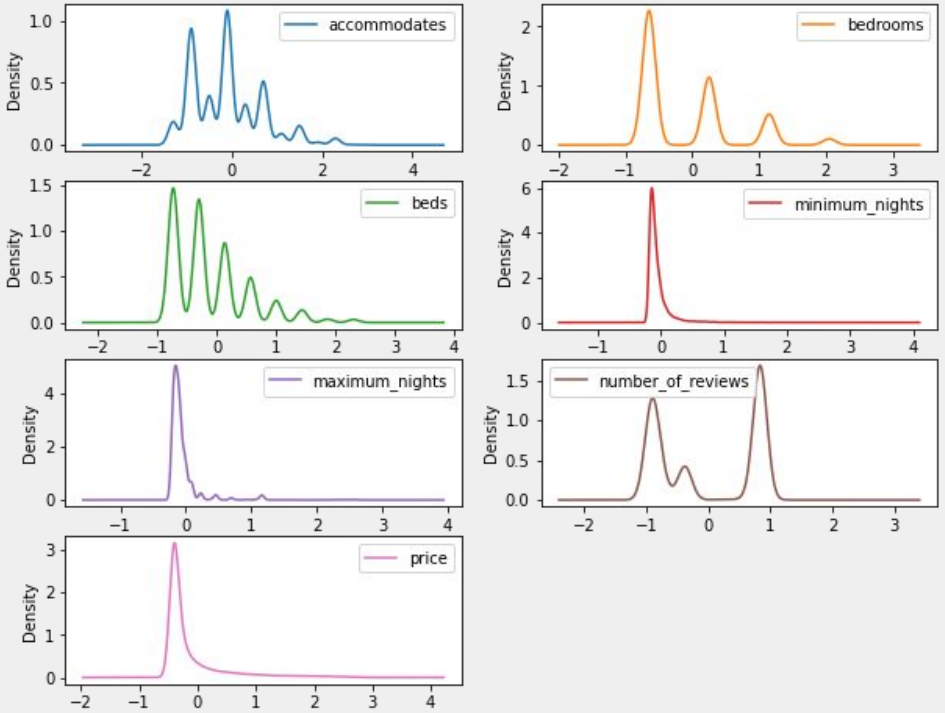
Remoção IQR

(14414, 7)



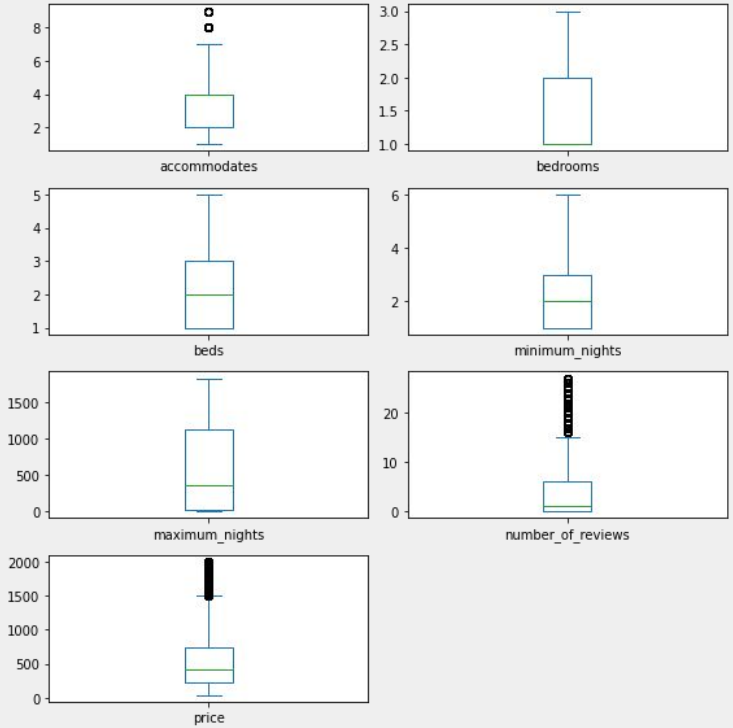
Remoção Z-Score

(21077,7)



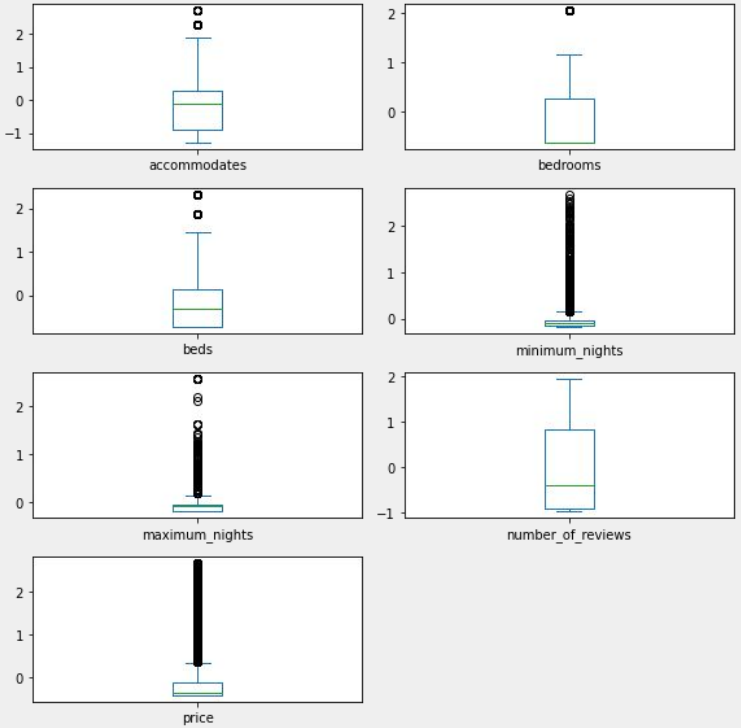
Remoção IQR

(14414, 7)

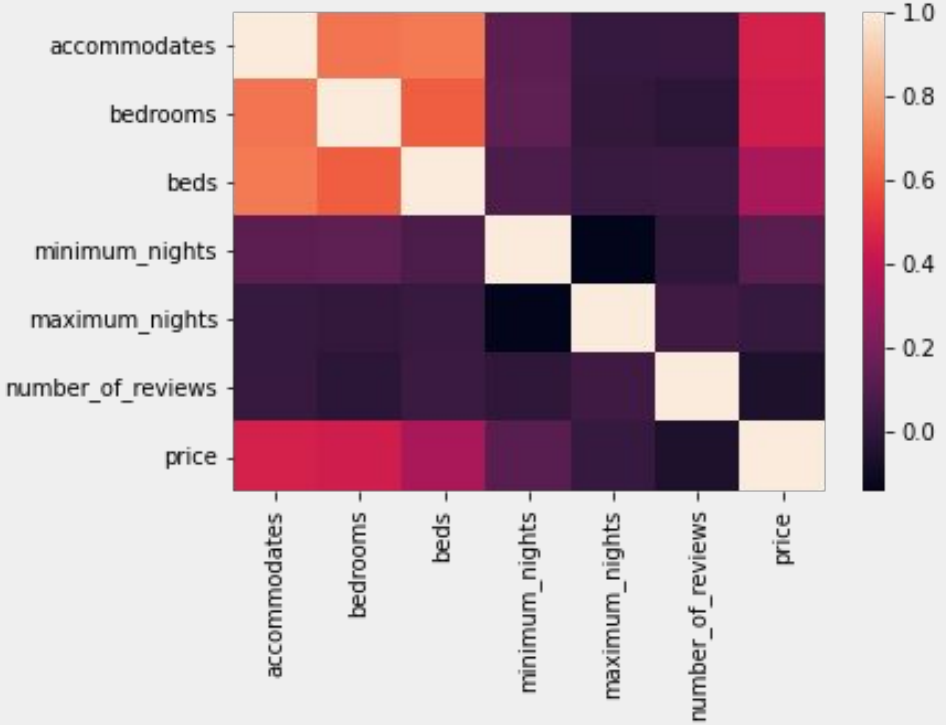


Remoção Z-Score

(21077,7)



Correlação entre as variáveis




```
import seaborn as sns
sns.heatmap(rio_iqr.corr())
```

Separando a base em treinamento e teste

	accommodates	bedrooms	beds	minimum_nights	maximum_nights	number_of_reviews	price
15	4	3.0	4.0	4	1125	1	220.0
19	2	1.0	1.0	5	120	14	851.0
23	2	1.0	1.0	2	60	14	243.0
25	2	1.0	1.0	5	30	7	500.0
26	2	1.0	2.0	1	90	0	851.0

```
# Using IQR

# Separate Data into a Training and Validation Datasets
from sklearn.model_selection import train_test_split

test_size = 0.20
seed = 20

# note that X_train, X_test, Y_train, Y_test are not standardized
# using IQR to eliminate outliers
X_train, X_test, Y_train, Y_test = train_test_split(rio_iqr.drop(axis=1,labels=["price"]),
                                                    rio_iqr["price"],
                                                    test_size=test_size,
                                                    random_state=seed)

print(X_train.shape,X_test.shape)
(11531, 6) (2883, 6)
```

```
from sklearn.neighbors import KNeighborsRegressor
from sklearn.metrics import mean_squared_error
import numpy as np

# instantiate a knn object
knn = KNeighborsRegressor(n_neighbors=5, n_jobs=-1,algorithm="brute")

# train the model
# Note that X and Y are not normalized
knn.fit(X_train,Y_train)

# predict
predict = knn.predict(X_test)

# evaluate
rmse = np.sqrt(mean_squared_error(Y_test,predict))

print(rmse)
378.2413288696399
```

Um primeiro modelo

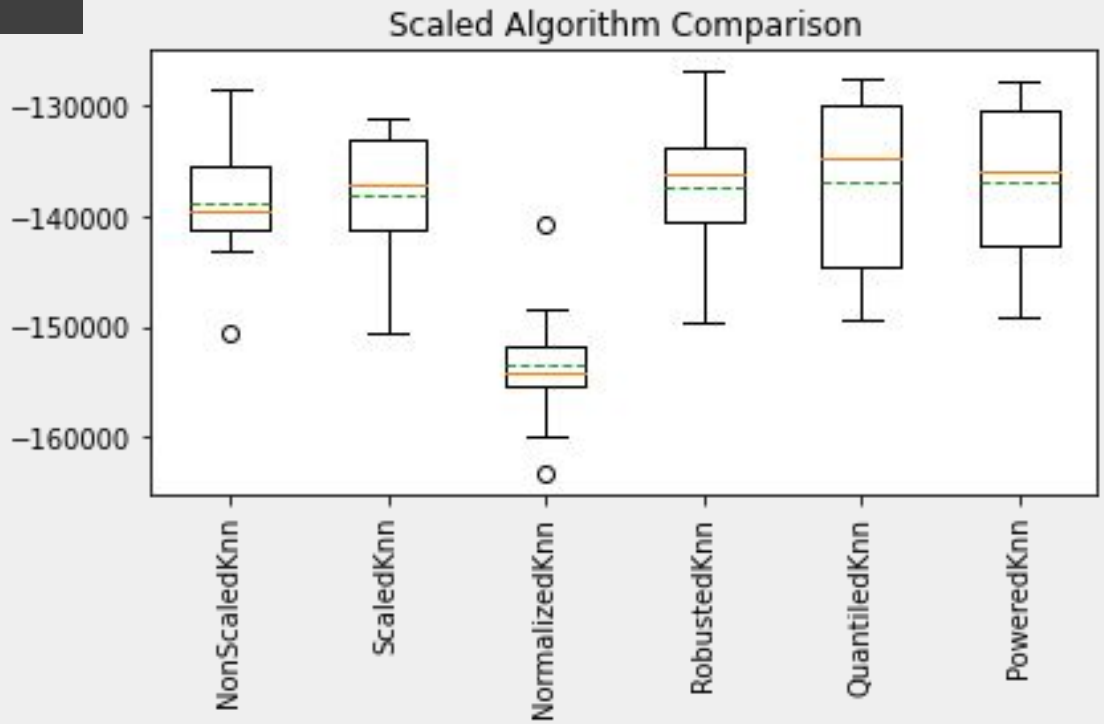
Qual a melhor técnica de normalização para esse problema (#pipeline)?

```
# Standardize the dataset
pipelines = []
pipelines.append(('NonScaledKnn',Pipeline([('KNN',KNeighborsRegressor(n_neighbors=5, algorithm="brute",n_jobs=-1))])))
pipelines.append(('ScaledKnn',Pipeline([('Scaler',StandardScaler()),
                                        ('KNN',KNeighborsRegressor(n_neighbors=5,n_jobs=-1))])))

pipelines.append(('NormalizedKnn',Pipeline([('Normalizer',Normalizer()),
                                            ('KNN',KNeighborsRegressor(n_neighbors=5, n_jobs=-1))])))

pipelines.append(('RobustedKnn',Pipeline([('Robust',RobustScaler()),
                                          ('KNN',KNeighborsRegressor(n_neighbors=5, n_jobs=-1))])))

pipelines.append(('QuantiledKnn',Pipeline([('Quantile',QuantileTransformer()),
                                           ('KNN',KNeighborsRegressor(n_neighbors=5, n_jobs=-1))])))

pipelines.append(('PoweredKnn',Pipeline([('Power',PowerTransformer()),
                                         ('KNN',KNeighborsRegressor(n_neighbors=5, n_jobs=-1))])))
```

Cross-Validation + Pipeline

```
from sklearn.model_selection import KFold
from sklearn.model_selection import cross_val_score
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import Normalizer
from sklearn.preprocessing import RobustScaler
from sklearn.preprocessing import QuantileTransformer
from sklearn.preprocessing import PowerTransformer

# Test options and evaluation metric
num_folds = 10
scoring = 'neg_mean_squared_error'
results = []
names = []

for name, model in pipelines:
 kfold = KFold(n_splits=num_folds)
 cv_results = cross_val_score(model, X_train, Y_train, cv=kfold, scoring=scoring)
 results.append(cv_results)
 names.append(name)
 print("%s Mean: %f Std: %f" % (name,
                                 np.sqrt(-cv_results.mean()),
                                 np.sqrt(cv_results.std())))
```

```
NonScaledKnn Mean: 372.632283 Std: 76.457267
ScaledKnn Mean: 371.578204 Std: 79.002880
NormalizedKnn Mean: 391.795614 Std: 76.538094
RobustedKnn Mean: 370.750472 Std: 83.735166
QuantiledKnn Mean: 370.147303 Std: 88.905904
PoweredKnn Mean: 369.933964 Std: 86.191869
```



Cross-Validation

Pipeline

Hyperparameter-Tuning?

- Identificar e compreender a personalização dos hiperparâmetros
- Entender a diferença entre a validação Holdout e a Cruzada (cross-validation)
- Navegar pela documentação do scikit-learn e estudar os hiperparâmetros do KNN-Regressor. Executar e compreender os códigos do notebook.
- Estudar e reproduzir a seção 6.
- Analisar as seções 6.4 e 6.5 com atenção!!!
- Duração: 60 a 90min