Predizendo o preço de casas em São Paulo com algoritmos de regressão

João Pedro Assumpção Evaristo

Universidade Federal de São Paulo

joao.evaristo@unifesp.br

***Resumo* - Este trabalho trará uma abordagem de algoritmos estudados durante o curso de Inteligência Artificial aplicados a um problema de predição do preço de casas na cidade de São Paulo. Para isso, foi usado uma base de dados com cerca de dez mil amostras e algoritmos de regressão, sendo eles a Regressão Linear e o K-Vizinhos Mais Próximos (KNN). Ao final, foi possível verificar qual deles obtinham o melhor resultado nessa aplicação.**

***Palavras-chave* — Inteligência artificial, regressão, knn, casas, São Paulo.**

I. Introdução

A inteligência artificial tem sido aplicada cada vez mais no cotidiano a fim de otimizar e automatizar tarefas que, anteriormente, eram feitas por humanos, de forma que, estavam suscetíveis a mais erros e demandavam um tempo maior para serem concluídas. Atualmente, essas aplicações cercam as tarefas cotidianas de forma tão natural que muitas vezes não são nem percebidas. Pode-se citar o seu uso em aplicativos de locomoção, assistentes pessoais, mecanismos de busca, publicidade, controles de estoque e logística, entre outras. Dessa forma, também é possível utilizar esses algoritmos para fazer predições, seja para metereologia, comportamentos, preço de commodities ou para avaliar quanto um imovel vale, sendo essa a abordagem a ser tratada em diante.

II. Trabalhos relacionados

Alguns trabalhos e publicações já têm explorado e mostrado o potencial dessa área de predição de preços de imóveis através de algoritmos de regressão. A seguir, será citado alguns e como estes ajudaram a propor esse trabalho.

A. Housing Price Prediction (Linear Regression), Ashish.[1]

O autor, nesse notebook, tem como objetivo treinar um algoritmo de Regressão Linear para que esse possa predizer o preço de propriedades na região de Delhi. Para isso, ele utiliza uma base com 546 amostras de imóveis, contendo informações como preço, área, quartos, banheiros, se conta com quartos de hóspedes, com porão entre outros aspectos. Antes de tudo, o autor faz alguns tratamentos nos dados, a fim de identificar e tratar outliers e saber a correlação entre as características dos imóveis. Foi baseado nessas técnicas que foi feito o tratamento dos dados no trabalho aqui proposto.

B. Linear Regression in Python with Scikit-Learn, Cássia Sampaio.[2]

Nesse artigo, a autora mostra qual a teoria por trás da regressão linear e como fazer um algoritmo utilizando essa técnica através da biblioteca Scikit-Learn, do ambiente Python. Essa publicação foi importante para o entendimento sobre as funções da biblioteca e como fazer o uso dessas na base de dados explorada.

C. Housing Price Prediction Using Linear Regression, Siddhant Burse et al.[3]

No trabalho desenvolvido pelos autores, e mostrando as etapas para fazer a predição do preço de imóveis situados na cidade de Mumbai, na Índia. Assim, é abordado desde a importância desse tipo de aplicação até em quais funcionalidades o algoritmo desenvolvido pode ser utilizado, dado a sua acurácia atingida de 86%.

D. Designing an optimal KNN regression model for predicting house price with Boston Housing Dataset, Kishor Keshav.[4]

A publicação traz uma abordagem da predição do preço de casas também, mas dessa vez, com a aplicação do algoritmo KNN com regressão. A base de dados utilizados nesse caso se refere a características de 506 residências situadas na cidade de Boston. Nesse caso, o autor utilizou algumas bibliotecas apenas para funções auxiliares, mas

implementou o algoritmo KNN de maneira manual, e isso auxiliou para o entendimento de como o mesmo funciona.

III. Metodologia

Primeiramente, para o desenvolvimento correto do trabalho, foi necessário fazer um tratamento dos dados obtidos da base de dados da Kaggle[5]. A base de dados original conta com dez mil e trinta e quatro amostras de casas, porém, algumas delas estão com precificação relacionadas a aluguel ou não estão localizadas na cidade de São Paulo, mas em outros municípios do estado. Outro obstáculo para o desenvolvimento do trabalho foi a localização das casas, que nesse caso não seguia um padrão, podendo ser descrita pela rua com número, ou apenas pelo bairro e em algumas ocorrências, eram localizadas em estradas.

Portanto, para esse primeiro tratamento e padronização dos dados, fiz a remoção das casas que estavam precificadas com o aluguel, já que o objetivo deste trabalho é predizer o preço de venda dos imóveis. Após essa etapa, desenvolvi um webdriver que tinha por objetivo a obtenção do Código de Endereçamento Postal (CEP) dos endereços apontados na base de dados, assim, o algoritmo itera sobre as linhas da base e procura a rua designada no site Busca CEP [6] do Correios. Para a construção desse algoritmo, utilizei a biblioteca selenium, do python, auxiliada da pandas, para criar um csv já com a adição de uma coluna designada ao CEP. Decidi fazer a utilização do CEP para a localização dos imóveis pois, além do fácil acesso, gera bons aglomerados de localizações para o caso dos imóveis [7]. Nos casos em que o endereço era dado apenas pelo bairro, o driver pegava o CEP da primeira ocorrência que constava no site dos Correios.

Ademais, outra inconsistência nos dados era a respeito de características apontadas sem precisão, como em casos que era fornecido um intervalo de valores:

Itaberaba, São Paulo	300	3	5	4	R$ 2.232.000
Santo Amaro, São Paulo	772	7	7	6	R$ 2.700.000
Vila Nova Conceição, São Paulo	605	3	3	4	R$ 4.190.000
Jardim da Saúde, São Paulo	110	3	2	5	R$ 860.000
Vila Poupança, Santana de Parnaíba	66-95	2-3	2	1-2	R$ 375.000
Jardim Santo Antônio, São Paulo	250	4	4	4	R$ 650.000
Vila Mazzei, São Paulo	120	2	1	1	R$ 490.000
Vila Madalena, São Paulo	170	2	2	1	R$ 1.485.000
Jardim Ivana, São Paulo	250	3	1	5	R$ 650.000
Vila Campo Grande, São Paulo	110	3	3	2	R$ 550.000
Vila Gomes Cardim, São Paulo	120	3	2	2	R$ 1.100.000

Fig. 1. Nesse caso, o número de vagas de uma residência era apontado como 1-2 vagas.

Para esses casos, arredondei os valores dos intervalos para cima, sendo que a ocorrência: vagas = 1-2, se tornou vagas = 2.

Na base, também havia casos de residências que se encontravam dentro de condomínios, assim, o número de vagas para estacionamento era muito alto e fora da realidade, pois nesses casos foi contabilizado o número de total de vagas do condomínio. Para essas ocorrências, considerei o número de vagas como sendo quatro.

Alto da Lapa, São Paulo	118	3	4	2	R$ 1.200.000
Morumbi, São Paulo	800	7	7	4	R$ 3.500.000
Vila Olímpia, São Paulo	130	3	2	1	R$ 1.100.000
Cambuci, São Paulo	220	3	4	1	R$ 1.000.000
SP	150	2	3	1	R$ 2.500.000
Jardim Paulista, São Paulo	1000	5	6	7	R$ 8.000.000
Santo Amaro, São Paulo	500	4	10	50	R$ 3.700.000
Vila Carrão, São Paulo	147	3	3	2	R$ 1.300.000
Aclimação, São Paulo	250	6	4	2	R$ 1.940.000
Vila Nova Conceição, São Paulo	240	1	3	3	R$ 6.000.000

Fig. 2. Exemplo de residência em que era considerado o número de vagas total do condomínio.

Por fim, foi necessário identificar onde havia a presença de outliers e fazer o tratamento desses posteriormente. Para isso, foi utilizado a função boxplot, da biblioteca Seaborn, que plota gráficos do tipo box plot e permite a identificação de outliers.

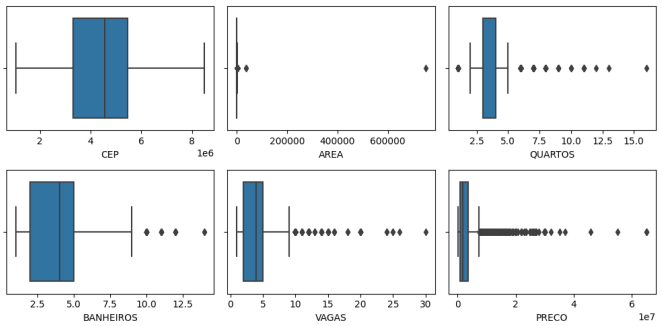

Fig. 3. Visualização dos outliers antes de um tratamento adequado.

Como é possível visualizar, há a presença de muitos outliers em campos como área, quartos, vagas e no preço. Assim, para fazer os tratamentos desses dados, utilizei o método Interquartile Range (IQR). Dessa forma, eliminei dados que estavam 1,5 vezes fora do intervalo entre os quartis 1º (Q1) e 3º (Q3). O resultado obtido após esse tratamento foi o seguinte:

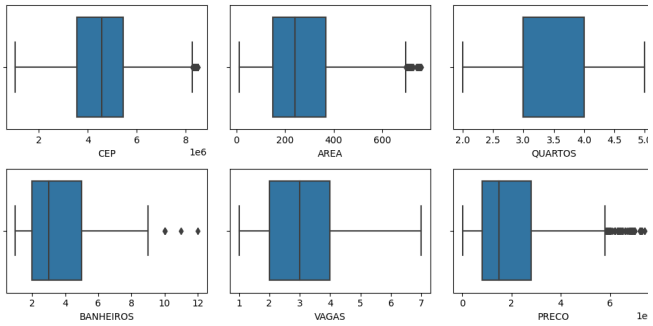

Fig. 4. Visualização da distribuição dos quartis após o tratamento dos outliers.

Após esse tratamento, a base ficou com um total de cinco mil quatrocentos e duas amostras. Nesse ponto, já era possível visualizar com mais detalhes a relação que as variáveis da base de dados tinham entre si. Para tal finalidade, utilizei a função heatmap da biblioteca Seaborn. Segue o resultado obtido:

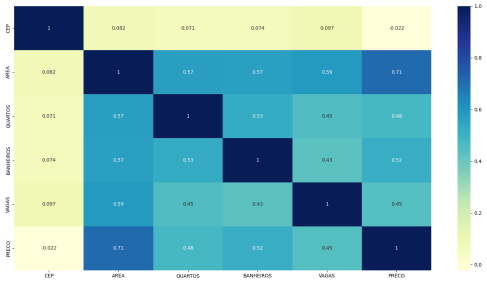

Fig. 5. Visualização da distribuição dos quartis após o tratamento dos outliers.

O mapa de calor das relações mostra bem como a área da casa e o que mais importa na sua precificação, sendo que as outras características tinham uma relação muito baixa com o preço. Adiante, isso será tratado com análises.

Para a parte prática dos algoritmos de regressão para predição dos preços, utilizei o modelo de Regressão Linear e KNN com regressão. A escolha pelo algoritmo de Regressão Linear se deu porque esse é um clássico em modelos de predição de preços, como mostrado nos trabalhos já citados. Dessa forma, partiria dele como um benchmarking para o teste no KNN. O KNN foi selecionado para esse teste devido a sua simplicidade e bom desempenho, possuindo uma boa relação com o objetivo do trabalho, visto que uma casa não varia muito suas características físicas e não há uma flutuação proporcional nos preços, considerando que se encontram na ordem de grandeza entre 5 e 6.

Para ambos os algoritmos, a implementação foi feita através da biblioteca Scikit-Learn, do Python.

IV. Análise Experimental

A. *Conjunto de dados*

Como dito anteriormente, o conjunto de dados analisados conta com dados sobre casas em São Paulo, trazendo informações sobre rua, cidade, área em metros quadrados, quantidade de quartos, banheiros, vagas e o seu preço. Após todos os tratamentos abordados anteriormente, a base de dados ficou da seguinte maneira:

Tab. 1.Pequena amostragem dos dados que foram usados no trabalho.

RUA	CEP	AREA	QUARTOS	BANHEIROS	VAGAS	PRECO
Avenida Itacira	4061000	1000	4	8	6	7000000
Rua Aurelia Perez Alvarez	4642020	524	6	6	4	3700000
Rua Alba Valdez	4845200	125	4	3	2	380000
Jardim Morumbi	5693000	310	3	2	4	685000
Rua Tobias Barreto	3176000	100	3	2	2	540000
Rua Graham Bell	4737030	440	4	4	6	1980000
Rua Francisco Paulo	3306050	145	4	4	2	850000
Rua Guilherme Valente	5818280	150	2	2	2	450000
Rua Sagrada Família	8140520	50	2	1	1	199900
Rua Tapaji	3207050	114	3	3	2	585000
Rua Vigário João Álvares	1551040	261	4	4	3	700000

B. *Configuração do algoritmo e do ambiente computacional*

Ambiente (hardware e software) utilizado para a execução dos algoritmos:

- CPU: Intel(R) Xeon(R) CPU E5-2620 v3 @ 3.20GHz
- Memoria: 16GB DDR4 (1333 MHz)

- Sistema Operacional: Windows 10 Home
- Versão: 21H2
- Compilador: Python 3.10
- Pacotes e versões: chromedriver-binary 104.0.5112.29, selenium 4.3.0, matplotlib 3.5.2, numpy 1.23.1, pandas 1.4.3, scikit-learn 1.1.1, sklearn 1.1.1 e seaborn 0.11.2.

C. Critérios de análise

Como tratado anteriormente, o algoritmo de regressão linear será utilizado como benchmark, tanto em tempo de execução quanto em acurácia obtida. Uma abordagem que achei válido testar neste trabalho é a diferença entre as análises com todas as características (cep, rua, área em metros quadrados, quantidade de quartos, banheiros e vagas) e utilizando apenas a área no conjunto de treino, tendo em vista que a área, como mencionado anteriormente, e a variável com maior relação com o preço. Para as diferentes execuções, foi usado diferentes parâmetros para o random_state, que serve para fazer o embaralhamento do conjunto, variando de um até 30. A seguir, tem-se os resultados:

- random_state=1
  Score de acuracia Regressao Linear: 0.5491026760641642
  Tempo de execucao regressao linear: 0.00699615478515625

  Score de acuracia KNN regressao: 0.6767106594301516
  Tempo de execucao knn: 0.0139923095703125

  Analise apenas com a area:

  Score de acuracia Regressao Linear: 0.5194882291656173
  Tempo de execucao regressao linear: 0.0009989738464355469

  Score de acuracia KNN regressao: 0.420256003669753
  Tempo de execucao knn: 0.0049970149993896484

- random_state=2
  Score de acuracia Regressao Linear: 0.5523181359302324
  Tempo de execucao regressao linear 0.0069959163665771484

  Score de acuracia KNN regressao: 0.6524029210184049
  Tempo de execucao knn 0.012992620468139648

  Analise apenas com a area:

  Score de acuracia Regressao Linear: 0.5212296075820053
  Tempo de execucao regressao linear 0.001999378204345703

  Score de acuracia KNN regressao: 0.42335405768734047
  Tempo de execucao knn 0.004997730255126953

- random_state=3
  Score de acuracia Regressao Linear: 0.533587583754179
  Tempo de execucao regressao linear: 0.006996631622314453

  Score de acuracia KNN regressao: 0.6247242896302049
  Tempo de execucao knn: 0.013991832733154297

  Analise apenas com a area:

  Score de acuracia Regressao Linear: 0.502729963527206
  Tempo de execucao regressao linear: 0.0019979476928710938

  Score de acuracia KNN regressao:

0.3528418078031922
Tempo de execucao knn: 0.004997968673706055

- random_state=4
  Score de acuracia Regressao Linear: 0.532113749435363
  Tempo de execucao regressao linear: 0.0069942474365234375

  Score de acuracia KNN regressao: 0.6395773389381033
  Tempo de execucao knn: 0.012992620468139648

  Analise apenas com a area:

  Score de acuracia Regressao Linear: 0.5108245958155528
  Tempo de execucao regressao linear: 0.0009996891021728516

  Score de acuracia KNN regressao: 0.4313866066667199
  Tempo de execucao knn: 0.00499725341796875

- random_state=5
  Score de acuracia Regressao Linear: 0.5282593528643715
  Tempo de execucao regressao linear: 0.0059964656829833984

  Score de acuracia KNN regressao: 0.6316652803736498
  Tempo de execucao knn: 0.012992620468139648

  Analise apenas com a area:

  Score de acuracia Regressao Linear: 0.507112507480713
  Tempo de execucao regressao linear: 0.0019981861114501953

  Score de acuracia KNN regressao: 0.4432439948911777
  Tempo de execucao knn: 0.004996776580810547

Como é perceptível que o tempo de execução não varia significativamente com o random_state, a partir desse ponto, deixei de trazer ele como parte da saída.

- random_state=6
  Score de acuracia Regressao Linear: 0.5332108526456134

  Score de acuracia KNN regressao: 0.6677954828594808

  Analise apenas com a area:

  Score de acuracia Regressao Linear: 0.5079619981765742

  Score de acuracia KNN regressao: 0.4132174894037627

- random_state=7
  Score de acuracia Regressao Linear: 0.5243089886898392

  Score de acuracia KNN regressao: 0.636372072620937

  Analise apenas com a area:

  Score de acuracia Regressao Linear: 0.507152469337959

  Score de acuracia KNN regressao: 0.4281027359276044

- random_state=8
  Score de acuracia Regressao Linear: 0.5593283933427866

  Score de acuracia KNN regressao: 0.6217312802764933

Analise apenas com a area:

Score de acuracia Regressao Linear:
0.5388779671894217

Score de acuracia KNN regressao:
0.43688308324140845

- random_state=9

  Score de acuracia Regressao Linear:
  0.5262089254681082

  Score de acuracia KNN regressao:
  0.65943815853249

  Analise apenas com a area:

  Score de acuracia Regressao Linear:
  0.5028241202321848

  Score de acuracia KNN regressao:
  0.4158825775927

- random_state=10
  0.5432268784275549

  Score de acuracia KNN regressao:
  0.6262499999266586

  Analise apenas com a area:

  Score de acuracia Regressao Linear:
  0.5193712964920044

  Score de acuracia KNN regressao:
  0.4217314627862717

Nesse ponto fica claro que, apesar de ser a característica que mais influência sobre o preço do imovel, considerar apenas a área do imovel, não é uma boa estratégia. A acurácia quando apenas a área faz parte do conjunto de treinamento se mostrou baixa em todas as execuções, não ultrapassando nenhuma vez a acurácia quando outros elementos são levados em conta. Uma conclusão que se pode tirar do uso de apenas um parâmetro é que o KNN é o algoritmo mais prejudicado por essa abordagem, tendo sua acurácia diminuída em até 36%.

A partir desse ponto, passei a desconsiderar a abordagem com apenas a área.

- random_state=11
  Score de acuracia Regressao Linear:
  0.5427539978432692

  Score de acuracia KNN regressao:
  0.622823520285577

- random_state=12
  Score de acuracia Regressao Linear:
  0.549169569381186

  Score de acuracia KNN regressao:
  0.6551011964502226

- random_state=13
  Score de acuracia Regressao Linear:
  0.5433677713293474

  Score de acuracia KNN regressao:
  0.6329967469359516

- random_state=14
  Score de acuracia Regressao Linear:
  0.5205138439796345

  Score de acuracia KNN regressao:
  0.6353919315578269

- random_state=15
  Score de acuracia Regressao Linear:
  0.5258743513893411

  Score de acuracia KNN regressao:
  0.6456607492738704

- random_state=16
  Score de acuracia Regressao Linear:
  0.5428985325186955

  Score de acuracia KNN regressao:
  0.6329486120404664

- random_state=17
  Score de acuracia Regressao Linear:
  0.5316062378634718

  Score de acuracia KNN regressao:
  0.6214291312099178

- random_state=18
  Score de acuracia Regressao Linear:
  0.5451078016782218

  Score de acuracia KNN regressao:
  0.6418427757844203

- random_state=19
  Score de acuracia Regressao Linear:
  0.5497578738359479

  Score de acuracia KNN regressao:
  0.6372233029248584

- random_state=20
  Score de acuracia Regressao Linear:
  0.5357012760779918

  Score de acuracia KNN regressao:
  0.644507531607537

  …..

- random_state=25
  Score de acuracia Regressao Linear:
  0.5378254851120259

  Score de acuracia KNN regressao:
  0.623442057982295

  ….

- random_state=30
  Score de acuracia Regressao Linear:
  0.5106956935379083

  Score de acuracia KNN regressao:
  0.6230019589865987

Após essas execuções, pode-se concluir que de fato, para esse problema, apesar do tempo de execução em média 85% maior que o de Regressão Linear, o KNN com regressão se saiu melhor em acurácia, sendo em média, 23% mais assertivo.

D. Resultados e discussões

Como visto anteriormente, o KNN se saiu melhor em acurácia em todos os testes, chegando a uma taxa 23% mais assertiva. Isso pode se observar a partir dos seguintes gráficos:

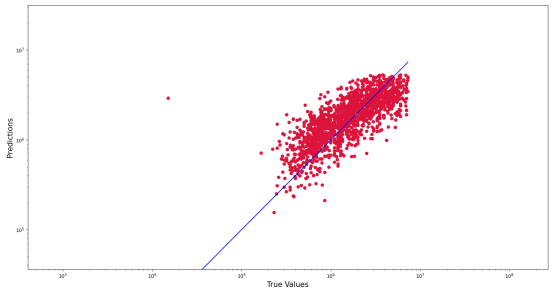

Fig. 6. Visualização da distribuição dos preços dados pelo algoritmo de regressão linear.

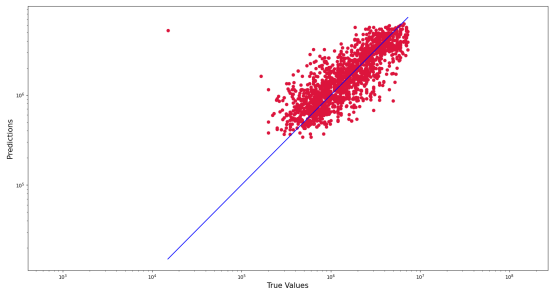

Fig. 7. Visualização da distribuição dos preços dados pelo algoritmo KNN.

Pode-se notar que os preços sugeridos pelo algoritmo KNN são menos espaçados da linha

esperada, se comparado com os resultados do algoritmo de Regressão Linear. Abaixo, estão algumas saídas de comparações de preços esperados versus preços preditos pelos algoritmos:

Tab. 2.Pequena amostragem dos preços fornecidos pela Regressão Linear.

Preco Atual	Preço previsto pela regressão
1380000	1236068
740000	607317
1200000	1996056
370000	670913
1810000	1159445
980000	1046533
420000	439802
490000	966454
1450000	2288322
800000	1169321
5500000	2908171

Tab. 3.Pequena amostragem dos preços fornecidos pelo KNN.

Preco Atual	Preço previsto pela regressão
1380000	1372500
740000	1170000
1200000	1672500
370000	1016500
1810000	1106500
980000	1636250
420000	2990000
490000	717500
1450000	1523000
800000	817000
5500000	3027500

Percebe-se que nenhum dos dois algoritmos conseguiu uma amostragem com 100% de acerto entre o preço esperado e o preço predito. Apesar disso, os preços dados se aproximam dos reais, sendo uma base satisfatória para o que se propõe.

Outra questão a se levar em conta é o tempo de execucao, que nesse caso, pode ser um importante fator de decisão entre os dois algoritmos, dependendo do uso. Como já foi levantado, a diferença entre os dois algoritmos chega a ser de 85%, com vantagem para o algoritmo de Regressão Linear. Uma diferença significativa que pode influenciar muito em bases de dados com maior número de amostras.

V. Conclusões

Com os resultados obtidos neste trabalho, pode-se concluir que apesar de práticos, os algoritmos abordados aqui não foram 100% precisos, atingindo um nível de acurácia de aproximadamente 68%, um nível satisfatório, mas não suficiente, ainda mais quando se trata de dinheiro. Os modelos abordados podem ser usados como apoio para a predição do preço de imóveis, assim, se tem uma noção se o preço se aproxima ou não do que é cobrado. Ademais, melhorias podem ser feitas no tratamento de dados, ou nos métodos utilizados, para que se alcance uma acurácia maior, além do tempo de execução, que aumenta significativamente no caso do KNN para se obter pouco ganho relativo de assertividade.

Referencias

[1] ASHISH, *Housing Price Prediction (Linear Regression)*, Kaggle, 2019. Disponível em: https://www.kaggle.com/code/ashydv/housing-price-prediction-linear-regression/notebook.

[2] SAMPAIO, Cássia, *Linear Regression in Python with Scikit-Learn*, StackAbuse, 2022. Disponível em: https://stackabuse.com/linear-regression-in-python-with-scikit-learn/.

[3] BURSE, Siddhant et al., *Housing Price Prediction Using Linear Regression*, Jetir, volume 8, 2021. Disponível em: https://www.jetir.org/papers/JETIR2110302.pdf.

[4] KESHAV, Kishor, *Designing an optimal KNN regression model for predicting house price with Boston Housing Dataset*, Medium, 2021. Disponível em: https://medium.com/mlearning-ai/designing-a-optimal-knn-regression-model-for-predicting-house-price-with-boston-housing-dataset-faef377536e3.

[5] SHASHANKK, *House price data of Sao Paulo,* Kaggle. Disponível em: https://www.kaggle.com/datasets/kaggleshashankk/house-price-data-of-sao-paulo?resource=download.

[6] Busca CEP, Correios. Disponível em: https://buscacepinter.correios.com.br/app/endereco/index.php

[7] O que é e o que significa o CEP, Significados. Disponível em: https://www.significados.com.br/cep/.