Data da versão atual: 22 de Julho, 2023

Estimação do preço de imóveis utilizando rede perceptron multicamadas

VICTOR ASSIS DE OLIVEIRA, KATARINE MELO LUCAS, FELIPE VASCONCELLOS NUNES GURGEL FARIAS
Departamento de Engenharia Eletrônica e de Computação, Universidade Federal do Rio de Janeiro, Rio de Janeiro, RJ 21941-914 Brazil
Corresponding author: Victor Assis de Oliveira (e-mail: victor$_assis@poli.ufrj.br$).

RESUMO O presente relatório tem como objetivo explorar a aplicação de uma rede perceptron multicamadas para a estimação de preço de imóveis. Serão abordados detalhes sobre o pré processamento dos dados utilizados, a construção da rede em função dos seus hiperparâmetros, incluindo o treinamento e validação e, por fim, uma conclusão sobre trabalho desenvolvido.

PALAVRAS CHAVE redes neurais, perceptron multicamadas, MLP, hiperparâmetros, aprendizado de máquina, imóveis.

I. INTRODUÇÃO

O mercado imobiliário é influenciado por diversas variáveis que impactam o preço dos imóveis. No entanto, muitas vezes, para aferir esses preços, é necessário recorrer a especialistas, pois faltam recursos tradicionais eficientes para auxiliar nessas estimativas ou validá-las. Os métodos tradicionais de estimação frequentemente enfrentam dificuldades ao identificar as relações não lineares entre essas variáveis, o que resulta em estimativas altamente especulativas e pouco fundamentadas cientificamente.

Nesse contexto, o objetivo deste experimento é desenvolver um modelo de rede neural ideal capaz de identificar essas relações não lineares e, assim, aferir preços que refletem o valor real de imóveis similares praticados no mercado imobiliário. A criação dessa ferramenta valiosa será benéfica tanto para profissionais do mercado quanto para proprietários e compradores, auxiliando-os a tomar decisões mais embasadas. Com essa abordagem, será possível tornar as estimativas de preços mais precisas e confiáveis, proporcionando maior segurança e eficiência ao mercado imobiliário.

II. CONJUNTO DE DADOS

A base de dados utilizada nessa pesquisa contém 13.378 registros e 21 atributos de imóveis da cidade de Melborn, Austrália. Os atributos e suas especificações foram listados na tabela 1. Alguns atributos foram modificados para melhor se adequarem ao processo de treinamento da rede. Todos os tratamentos de dados serão abordados na sessão III - Pré-processamento.

TABELA 1. Especificações do conjunto de dados

Atributo	Tipo	Descrição
Suburb	object	Bairro
Address	object	Rua e número
Rooms	int64	Número de cômodos
Type	object	h - casa; u - apartamento
Price	float64	Preço de venda
Method	object	S - imóvel vendido; SP - imóvel vendido antes; PI - imóvel não arrematado; PN - venda anterior não divulgada; SN - venda não divulgada; NB - sem oferta; VB - oferta do vendedor; W - retirado antes do leilão; SA - vendido após o leilão; SS - vendido após o leilão, preço não divulgado. N/A - preço ou oferta mais alta não disponível.
SellerG	object	Agente imobiliário
Date	object	Data da venda
Distance	float64	Distância do centro comercial
Postcode	float64	Código postal
Bedroom2	float64	Número de quartos
Bathroom	float64	Número de banheiros
Car	float64	Número de vagas para automóveis
Landsize	float64	Tamanho do terreno
BuildingArea	float64	Tamanho do imóvel
YearBuilt	float64	Ano de construção
CouncilArea	object	Distrito governamental
Lattitude	float64	Tatidude
Longtitude	float64	Longitude
Regionname	object	Região geral (Norte, Sul, Noroeste, Oeste, etc)
Propertycount	float64	Número de imóveis existentes no bairro

III. PRÉ-PROCESSAMENTO

O tratamento da base de dados foi uma etapa fundamental para atingir um dos melhores resultados possíveis neste projeto. Após a análise dos dados, foram encontrados valores nulos, não numéricos, valores muito discrepantes dentro de um único conjunto de informações (outliers) e atributos sem relação alguma com o alvo da nossa análise, o preço dos imóveis.

Dessa forma, dividimos o pré-processamento dos dados em 5 etapas, remoção de atributos, tratamento dos dados

faltantes, tratamento dos dados não numéricos, remoção dos valores atípicos (*outliers*) e normalização dos dados.

A. REMOÇÃO DE ATRIBUTOS

Após uma breve análise do conjunto de dados, decidimos remover quatro atributos que julgamos ser irrelevantes para a construção do preço dos imóveis. São eles: ***SellerG***, ***Date***, e ***Method***. Além desses, ***Address*** também foi removido.

Sobre o atributo ***Address***, é importante ressaltar que sim, seria possível utilizá-lo após um certo tratamento. Sabemos que ruas específicas dentro de um bairro ou até mesmo números específicos dentro de uma rua podem fazer diferença no preço de imóveis. No entanto, para simplificar a complexidade do projeto em um primeiro momento, optamos por remover esse atributo, mas deixando claro que esse seria um ótimo ponto de possível melhoria para uma continuação da pesquisa.

B. TRATAMENTO DOS DADOS FALTANTES

Durante a análise exploratória, foi observado que quatro atributos apresentavam valores nulos, como mostra a figura 1. Esses valores precisavam ser transformados para que fosse possível usá-los no treinamento da rede. Incorporamos dois tipos de tratamento para esses atributos.

1) Substituição dos valores

Para ***Car***, determinamos que seus valores nulos seríam transformados em 0, assumindo que os imóveis que não informaram o número de vagas para automóveis não possuem vagas.

2) Remoção do atributo

Para ***BuildingArea***, ***YearBuilt*** e ***ConcilArea***, devido ao alto número de registros nulos, decidimos removê-los completamente do conjunto de dados.

```
Suburb            0
Address           0
Rooms             0
Type              0
Price             0
Method            0
SellerG           0
Date              0
Distance          0
Postcode          0
Bedroom2          0
Bathroom          0
Car              62
Landsize          0
BuildingArea   6450
YearBuilt      5375
CouncilArea    1369
Lattitude         0
Longtitude        0
Regionname        0
Propertycount     0
dtype: int64
```

FIGURA 1. Contagem de valores nulos para cada atributo.

C. TRATAMENTO DOS DADOS NÃO NUMÉRICOS

Da mesma forma que precisamos transformar valores nulos para adequação ao processo de treinamento da rede, também é necessário transformar valores não numéricos. Na figura 2, podemos ver quais dos atributos restantes apresentam dados não numéricos. São eles ***Suburb***, ***Type***, e ***Regionname***.

Para resolver essa questão, foram utilizadas duas abordagens diferentes:

1) One-hot encoding

Para o atributo ***Type***, que apresenta variáveis categóricas, foi utilizada a técnica de codificação one-hot encondig, onde para cada categoria, isto é, para cada valor único do atributo, é criado um novo atributo binário. Cada novo atributo binário representa a presença ou não da categoria original. Dessa forma, 3 novas colunas foram adicionadas ao conjunto de dados, ***Type_h***, ***Type_t***, e ***Type_u***. Isto, pois apenas 3 categorias foram encontradas para o atributo original no conjunto de dados.

```
Suburb          object
Rooms            int64
Type            object
Price          float64
Distance       float64
Postcode       float64
Bedroom2       float64
Bathroom       float64
Car            float64
Landsize       float64
Lattitude      float64
Longtitude     float64
Regionname      object
Propertycount  float64
dtype: object
```

FIGURA 2. Tipos de dados de cada atributo.

2) Mean encoding

Para os atributos ***Suburb*** e ***Regionname*** foi utilizada a técnica de codificação mean enconding, onde se é calculada a média do preço dos imóveis para cada categoria e, então, o valor encontrado é usado como substituto do nome da categoria. Isto é, para o atributo ***Suburb***, por exemplo, o valor referente à média do preço dos imóveis de um bairro é usado como rótulo para esse bairro. Dessa forma, conseguimos criar uma relação mais evidente entre o preço e o bairro dos imóveis. O mesmo é feito para o atributo ***Regionname***.

	Type		Type_h	Type_t	Type_u
0	h	0	1	0	0
1	h	1	1	0	0
2	h	2	1	0	0
3	h	3	1	0	0
4	h	4	1	0	0
5	h	5	1	0	0
6	h	6	1	0	0
7	h	7	1	0	0
8	u	8	0	0	1
9	h	9	1	0	0

FIGURA 3. Aplicação da técnica One-hot encoding no atributo ***Type***.

D. REMOÇÃO DOS VALORES ATÍPICOS *(OUTLIERS)*

Para focar a análise em cima de imóveis mais comuns, e evitar casos muito específicos que fogem do padrão, fizemos uma análise dos *outliers* e removemos imóveis com valores acima de AU$2,100.00.

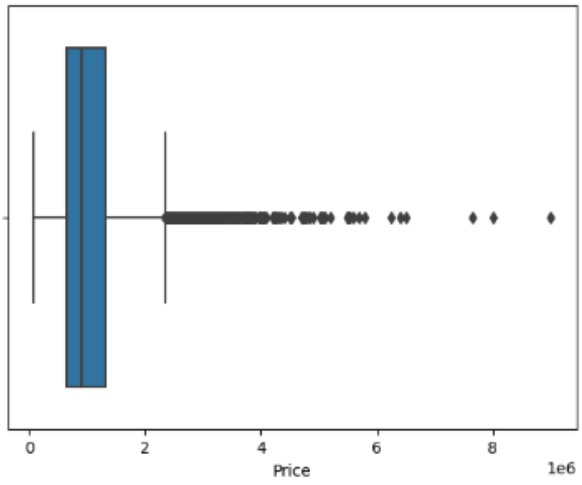

FIGURA 4. *Outliers* de preço

E. NORMALIZAÇÃO DOS DADOS

Por fim, é feita a normalização dos dados, que consiste em reescalar os valores das variáveis para o intervalo entre -10 e 10 para eliminar o viés na escala dos dados.

F. MAPA DE CALOR

Após todas as transformações obtivemos o mapa de calor representado na figura 7 na sessão VI. Apêndices.

IV. CONSTRUÇÃO DA REDE

Para a construção da rede foram utilizadas as bibliotecas 'Keras' e 'Tensorflow'. Na inicialização, a função de modelo sequencial foi implementada para a formação das camadas. Além disso, para a escolha da função de ativação, alguns testes foram necessários para verificar qual delas proporcionava o melhor resultado e, portanto, as escolhas foram a 'relu' para as camadas ocultas e 'linear' para a saída.

TABELA 2. Primeira rodade de testes realizados

Camadas	Batch size	Epocs	Validation split	Resultado
15,15,15,15	32	50	0.2	Loss-179716.141 R²-0.545 DR-0.212
15,7,7,4	32	50	0.2	Loss-192877.437 R²-0.489 DR-0.223
13,13,7	32	50	0.2	Loss-216290.703 R²-0.347 DR-0.254
13,13,13,13,13	32	50	0.2	Loss-168873.703 R²-0.587 DR-0.194
15,13,13,7	32	50	0.4	Loss-187139.656 R²-0.511 DR-0.219
15,15,15,15,15	32	100	0.2	Loss-162242.391 R²-0.609 DR-0.179
15,15,15,15,15,15	32	100	0.2	Loss-163125.250 R²-0.606 DR-0.178

As métricas utilizadas para avaliar o modelo foram *R-squared* e discrepância relativa, a primeira fornece uma indicação do quão bem as previsões do modelo se ajustam aos dados observados, já o segundo é possível ver o quão próximo o modelo está prevendo os valores corretos em termos relativos.

Na compilação da rede utilizamos um otimizador conhecido como 'adam' e para constatar a perda foi utilizado o erro absoluto médio.

Antes de abordar os testes vale ressaltar que os valores e funções para testes foram decididos de forma arbitrária sem a ajuda de algoritmos para atingir o melhor resultado.

A. PRIMEIRA RODADA DE TESTES

Na primeira rodada de testes, foram aplicadas as funções 'relu' na entrada das primeiras camadas e, na saída, a função 'linear'. Quanto a quantidade de camadas, os valores testados foram três, quatro e seis com o mesmo número de neurônios e, também, variando. Cabe destacar também que foram realizados alguns testes na primeira camada oculta começando em 13 até 15, na segunda tentativa de 7 a 15, na terceira de 7 a 15, quarta de 4 a 15, quinta de 13 a 15 e na sexta 15. No inicializador kernel foi utilizado o valor 'normal' e 'random_normal.

Além disso, para treinar a rede foi utilizado batch sizes entre 16 e 64, as épocas variadas de 50 a 500 e para testar os dados foi feita uma divisão 80/20 entre treinamento e teste. Vale ressaltar que a função de erro também foi alterada entre erro absoluto médio e erro quadrático médio. Outras funções de perda também foram testadas, porém, elas apresentaram resultados significativamente piores comparando com as citadas anteriormente. Nessa primeira rodada de testes foi-se obtido uma perda de 162242.3906, R2-score de 0.6093 e uma discrepância relativa de 0.1795.

Alguns dos testes realizados podem ser encontrados na Tabela IV-A, vale ressaltar que, enem todos os hiperparâmetros testados estão representados nas tabelas de testes, pois

TABELA 3. Segunda rodada de testes

Camadas	Batch size	Epocs	Learning rate	Validation split	Resultado
38,13	32	300	0.2		Loss-163439.094 R²-0.605 DR-0.182
38,13	32	350	0.3		Loss-162138.125 R²-0.610 DR-0.180
45,13	32	400	0.2		Loss-163190.922 R²-0.607 DR-0.181
45,13	32	480	0.2		Loss-161820.687 R²-0.612 DR-0.179
20,20	16	2500	0.2	0.0015	Loss-161664.672 R²-0.614 DR-0.179
20,20	20	1000	0.2	0.0030	Loss-159369.922 R²-0.622 DR-0.176
20,20	20	400	0.2	0.0040	Loss-162257.469 R²-0.610 DR-0.179
30,30,30	32	250	0.2	0.0030	Loss-145537.234 R²-0.6909 DR-0.157
30,30,30	32	150	0.2	0.0040	Loss-150246.219 R²-0.667 DR-0.166
30,30,30	32	150	0.2	0.0030	Loss-148831.531 R²-0.664 DR-0.174
30,30,30	32	250	0.2	0.0010	Loss-155042.812 R²-0.638 DR-0.186

esses não apresentaram resultados significativamente diferentes dos que já estão representados.

B. SEGUNDA RODADA DE TESTES

O objetivo da segunda rodada foi melhorar o resultado antes obtido, diminuindo a quantidade de camadas escondidas atingindo o valor de máximo três. Para isso ser possível, tornou-se necessário aumentar a quantidade de neurônios na segunda e, quando adicionada, na terceira camada, do mesmo modo, o número de épocas foi aumentado até alcançar o valor de 2500, também foi importante modificar os valores do otimizador definindo um valor de 0,0010 até 0,0040 para o campo 'learning rate' no modelo 'adam' e, no campo 'batch size', os números variaram entre 16 a 32 novamente. Após aplicar estas as variações, foi-se observado resultados melhores comparado a primeira rodada de testes, atingindo 145537.23 perdas, R-squared 0.6909 e discrepância relativa 0.1566, respectivamente.

V. CONCLUSÃO

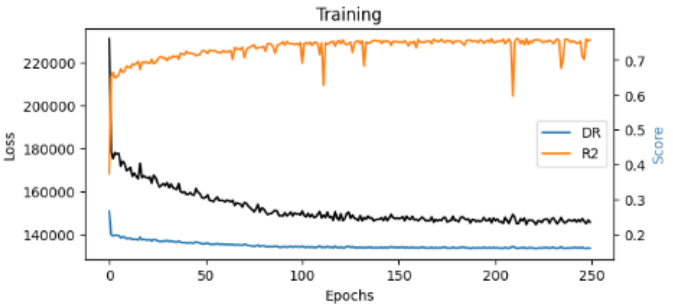

FIGURA 5. *Outliers* de preço

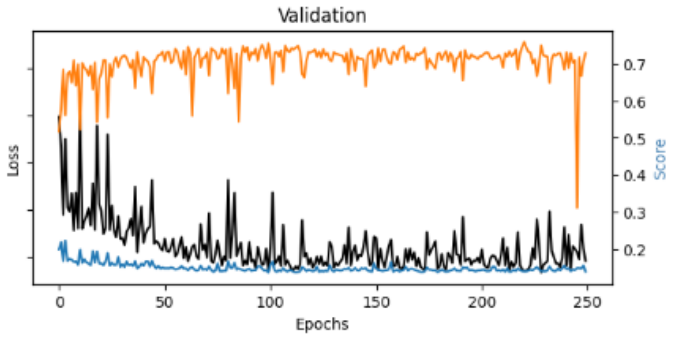

FIGURA 6. *Outliers* de preço

Após realizar várias combinações de alterações nos hyperparâmetros da rede, obtivemos o melhor resultado com uma perda de AU$145,537.23, um R2_score de 0.69 e uma discrepância relativa de 0.15.

No entanto, o resultado alcançado não atingiu nossas expectativas. Notamos que, independentemente das diversas combinações de parâmetros testadas, os resultados finais foram muito semelhantes. Acreditamos que uma possível razão para isso seja a falta de informações complementares sobre os imóveis. Diversos atributos não mencionados podem ter uma influência significativa no valor de um imóvel, tais como a presença de piscina ou churrasqueira, o tamanho em metros quadrados, o ano de construção, o estado de conservação, entre outros.

Para futuras análises, planejamos testar novos conjuntos de dados que contenham uma maior riqueza de detalhes sobre os imóveis. Acreditamos que essas informações adicionais podem fornecer uma visão mais completa e precisa para o treinamento do modelo, o que pode resultar em resultados mais satisfatórios.

REFERENCES

[1] Deep Learning with Python: Neural Networks (complete tutorial) [Online]. Available: https://towardsdatascience.com/ deep-learning-with-python-neural-networks-complete-tutorial.
[2] Melbourne Housing Snapshot Snapshot of Tony Pino's Melbourne Housing Dataset Available: https://www.kaggle.com/datasets/dansbecker/melbourne-housing-snapshot
[3] ChatGPT Available: https:https://chat.openai.com/

VI. APÊNDICES

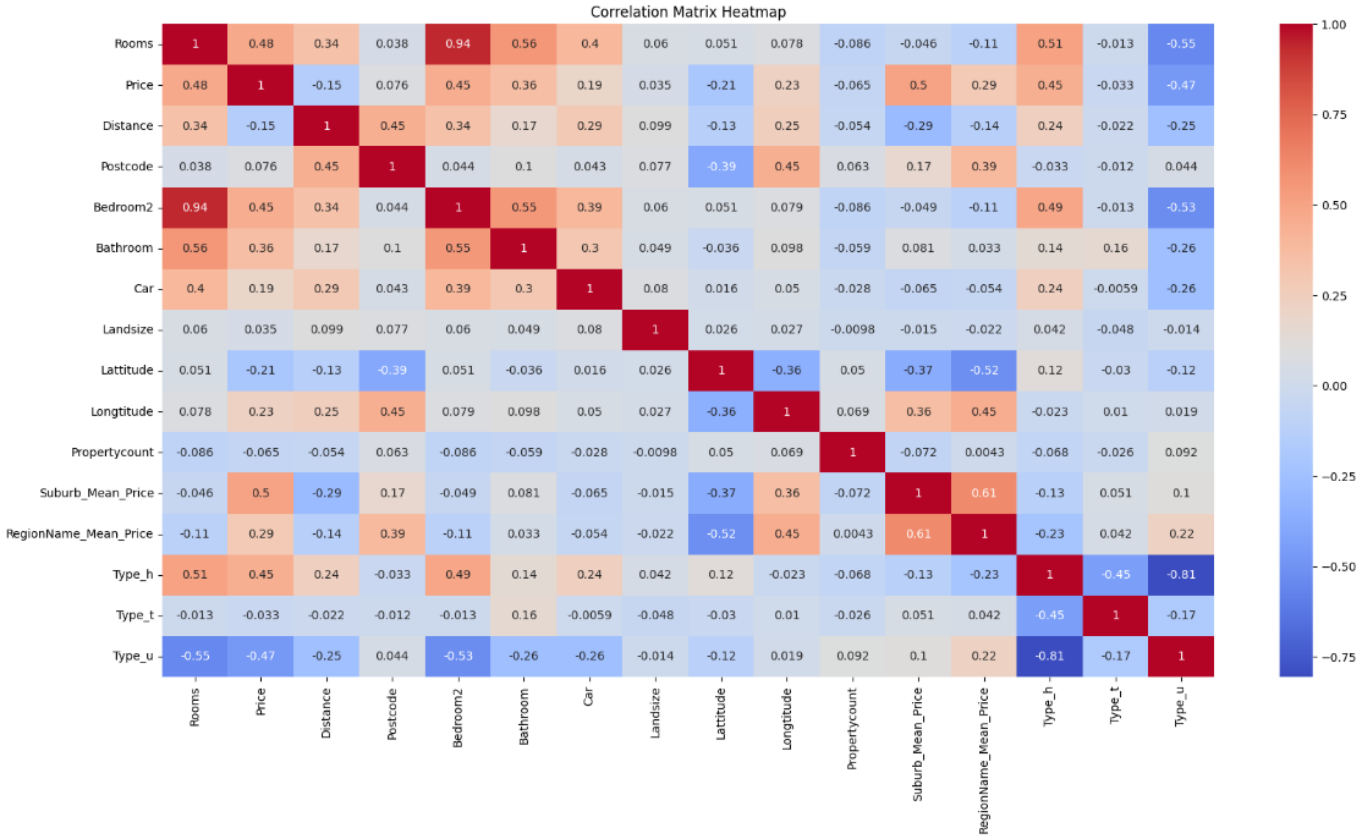

FIGURA 7. Mapa de calor após o pré-processamento