Przewidywanie ceny mieszkań na podstawie danych dostępnych na serwisie OTODOM

Jakub Stępkowski
Politechnika Wrocławska

Czerwiec 2024

1 Wstęp

2 Pozyskanie zbioru danych

opisz scraper

3 Przetwarzanie wstępne

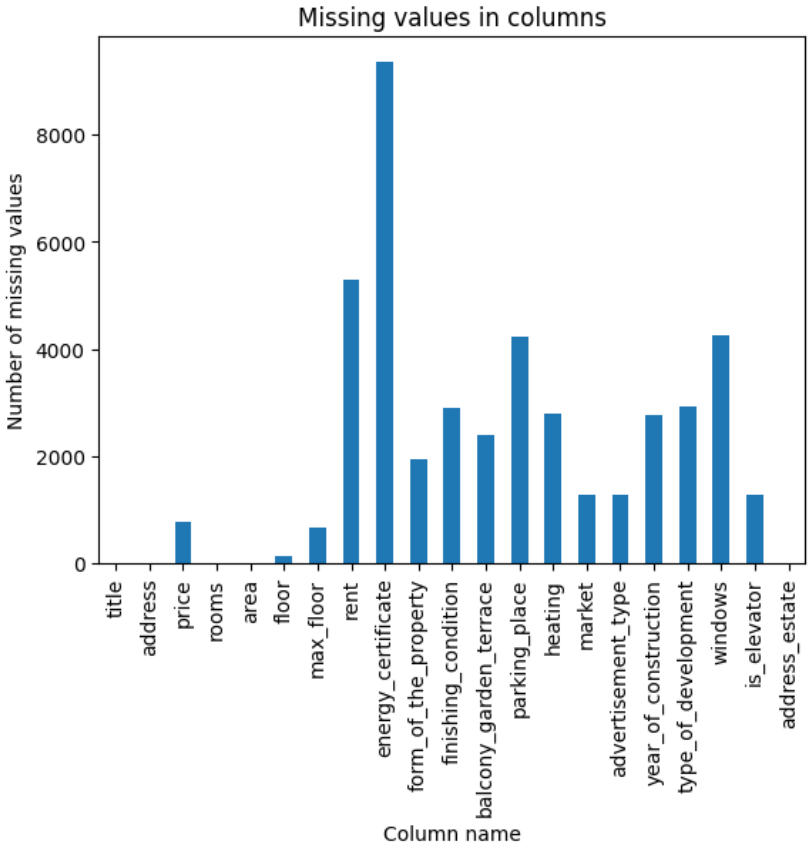
Zbiór danych zawiera następujące cechy:

1. title - tytuł ogłoszenia
2. address - adres mieszkania
3. price - cena mieszkania
4. rooms - liczba pokoi
5. area - powierzchnia w m^2
6. floor - piętro na którym znajduje się mieszkanie
7. max_floor - liczba pięter w budynku
8. rent - wysokość renty
9. energy_certificate -
10. form_of_the_property
11. finishing_condition
12. balcony_garden_terrace
13. parking_place

14. heating
15. market
16. advertisement_type
17. year_of_construction
18. type_of_development
19. finishing_condition
20. windows
21. is_elevator

... dopisz opisy

Z adresu została wykorzystania wyłącznie nazwa osiedla, na którym znajduje się mieszkanie. Umieszczono ją w kolumnie *adress_estate*. Cały zbiór zawiera 9843 mieszkań, jednak nie wszystkie oferty mają uzupełnione wszystkie cechy. Poniższy wykres pokazuje, dla każdej cechy, w ilu próbkach jest ona pusta:



Po analizie brakujących danych postanowiono wykorzystać następujące cechy: rooms, area, address_estate, floor, max_floor, market, year_of_construction, finishing_condition, is_elevator. Wszystkie oferty, które nie zawierały jakiejś z tych cech, zostały usunięte. W ten sposób zbiór danych został zredukowany do 4571 elementów.

[ewentualnie opisz 10+ itd ale mowil ze go nie interesuje oczyszczanie xzbioru wiec pomijam]

Zbiór zawiera następujące cechy kategorialne: adress_estate, market, finishing_condition, is_elevator. Cechy binarne (market i is_elevator) zakodowano przez jedną cechę liczbową, gdzie 0 oznacza jedną z klas a 1 drugą. Cechy zawierające więcej kategorii (finishing_condition i is_elevator) zakodowano wykorzystując one hot encoding.

Wyizolowano cechę price jako cel predykcji. Ze względu na stabilność modeli postanowniono przewidywać cenę za metr kwadratowy, zamiast całkowitej ceny mieszkania.

Zbiór został podzielony na 3 podzbiory - treningowy, walidacyjny i testowy

Tak przygotowany zbiór danych przeskalowano z wykorzystaniem standardowego skalera, dopasowanego do zbioru treningowego (zarówno dla cech wejściowych jak i dla celu).

4 Eksploracja danych

wykresy

5 Wybór i trening modeli

5.1 Model liniowy

5.2 Uogólniony model liniowy

5.3 Sieć neuronowa

Architektura wybranej sieci składała się z następujących warstw:

1. Warstwa wejściowa - 11 neuronów
2. Warstwa ukryta 1 - 256 neuronów z funkcją aktywacji ReLU
3. Warstwa ukryta 2 - 128 neuronów z funkcją aktywacji ReLU
4. Warstwa wyjściowa - 1 neuron

Zastosowano gęste połączenia między kolejnymi warstwami. Została użyta funkcja aktywacji ReLU ze względu na jej niski koszt obliczeniowy. Po każdej z warstw ukrytych, na czas treningu, zastosowano dropout wielkości 30% każdy w celu uniknięcia nadmiernego dopasowania do zbioru treningowego.

trening

Do treningu został użyty algorytm optymalizacji Adam ze współczynnikiem uczenia 0.001. Pętla ucząca została wykonana 100 razy. Za każdym powtórzeniem odbywała się optymalizacja współczynników sieci na podstawie całego zbioru danych, pogrupowanego w serie po 16 próbek, w celu przyspieszenia obliczeń i lepszemu uogólnianiu. Wykorzystana funkcja straty to błąd średniokwadratowy.

Do implementacji sieci został wykorzystany moduł PyTorch.

6 Porównanie modeli

7 Wnioski