به نام خداوند بخشنده مهربان

عنوان پروژه

پیش‌بینی قیمت منزل با استفاده از رگرسیون خطی

عنوان درس

یادگیری ماشین

استاد

دکتر الهام قصرالدشتی

دستیاران آموزشی

مهرداد قصابی

مریم صفوی

گردآورنده

سید حسین حسینی

بهار ۱۴۰۴

دانشکده مهندسی کامپیوتر

دانشگاه اصفهان

➢ کتابخانه‌ها

- Numpy : برای کار با آرایه‌ها و داده‌ها و انجام عملیات جبری مناسب
- Pandas : برای تحلیل داده‌ها و همچنین انجام عملیات مناسب بر روی دیتافریم‌ها
- Seaborn : برای مصورسازی داده‌ها و روابط بین آن ها و انجام تحلیل بر روی آن‌ها
- Matplotlib : برای مصورسازی داده‌ها و روابط بین آن ها و انجام تحلیل بر روی آن‌ها
- Sklearn : یکی از کتابخانه‌های یادگیری ماشین برای انجام عملیات‌های مرتبط با آن
- Gdown : برای دانلود دیتاست از لینک دریافتی

➢ پیش‌پردازش

- حذف داده‌های NULL

در ابتدا با توجه به اینکه داده‌های NULL باعث کاهش قدرت مدل میشوند باید در ابتدا از دیتاست حذف و با داده مناسب جایگذاری شوند، که در اینجا برای اینکه بتوانیم تعادل مناسبی بین داده‌ها برقرار کنیم از میانگین در داده‌های عددی و از مد در داده‌های غیر عددی استفاده کردیم. در جزئیاتی که در زیر مشاهده میکنید این نوع داده‌ها بسیار کم بوده و نیاز به پردازش‌های خیلی خاص و دقیقی نداشته است و همچنین این روش جز یکی از روش های مرسوم جایگذاری است.

```
RangeIndex: 128 entries, 0 to 127
Data columns (total 7 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   Price         128 non-null    int64
 1   SqFt          126 non-null    float64
 2   Bedrooms      125 non-null    float64
 3   Bathrooms     127 non-null    float64
 4   Offers        126 non-null    float64
 5   Brick         128 non-null    object
 6   Neighborhood  128 non-null    object
dtypes: float64(4), int64(1), object(2)
memory usage: 7.1+ KB
Dataset Info:
 None

Statistical Description:
                Price         SqFt    Bedrooms   Bathrooms      Offers
count      128.000000   126.000000  125.000000  127.000000  126.000000
mean    130427.343750  2001.666667    3.032000    2.448819    2.563492
std      26868.770371   212.387382    0.728852    0.514992    1.069550
min      69100.000000  1450.000000    2.000000    2.000000    1.000000
25%     111325.000000  1882.500000    3.000000    2.000000    2.000000
50%     125950.000000  2000.000000    3.000000    2.000000    3.000000
75%     148250.000000  2140.000000    3.000000    3.000000    3.000000
max     211200.000000  2590.000000    5.000000    4.000000    6.000000

Number of NULL values per column:
 Price           0
SqFt             2
Bedrooms         3
Bathrooms        1
Offers           2
Brick            0
Neighborhood     0
dtype: int64
```

- حذف داده‌های پرت

با توجه به اینکه داده‌های پرت باعث می‌شود که دچار بیش برازش شویم، در نتیجه نیاز است که شناسایی و حذف شوند و عموما از داده‌های آموزش حذف میشوند ولی در اینجا به علت اینکه تعداد داده‌های پرت خیلی محدود هستند و داده‌های آموزش و تست شباهت زیادی بهم دارند، این مورد در هر دو داده اتفاق می‌افتد و همچنین به علت اینکه با استفاده از روش‌هایی مثل IQR و Z_Score بخش زیادی از داده‌ها به عنوان داده‌های پرت شناسایی شدند، در این زمینه از تحلیل دستی استفاده شده و بر اساس یکسری از متریک‌ها بخشی از داده‌ها شناسایی و حذف شدند که در زیر قابل مشاهده است. (مثلا در روش IQR به علت محدود بودن بازه عددی باعث میشد هر عددی غیر از 3 به عنوان داده پرت شناسایی شود.)

```
# Price thresholds - remove extremely cheap or expensive houses
clean_df = clean_df[(clean_df['Price'] >= 70000) & (clean_df['Price'] <= 200000)]

# Square footage thresholds - remove very small or very large houses
clean_df = clean_df[(clean_df['SqFt'] >= 1500) & (clean_df['SqFt'] <= 2500)]

# Bedrooms threshold - remove houses with more than 4 bedrooms
clean_df = clean_df[clean_df['Bedrooms'] <= 4]

# Offers threshold - remove houses with more than 5 offers
clean_df = clean_df[clean_df['Offers'] <= 5]
```

```
Initial row count: 128

Initial descriptive statistics:
               Price         SqFt    Bedrooms   Bathrooms      Offers
count     128.000000   128.000000  128.000000  128.000000  128.000000
mean   130427.343750  2001.640625    3.031250    2.445312    2.570312
std     26868.770371   210.708506    0.720209    0.514492    1.062486
min     69100.000000  1450.000000    2.000000    2.000000    1.000000
25%    111325.000000  1887.500000    3.000000    2.000000    2.000000
50%    125950.000000  2000.000000    3.000000    2.000000    3.000000
75%    148250.000000  2140.000000    3.000000    3.000000    3.000000
max    211200.000000  2590.000000    5.000000    4.000000    6.000000

Row count after outlier removal: 120
Number of rows removed: 8

Descriptive statistics after cleaning:
               Price         SqFt    Bedrooms   Bathrooms      Offers
count     120.000000   120.000000  120.000000  120.000000  120.000000
mean   129585.000000  1991.500000    2.991667    2.425000    2.525000
std     24606.843518   187.422365    0.667314    0.496416    1.003879
min     81300.000000  1520.000000    2.000000    2.000000    1.000000
25%    111550.000000  1887.500000    3.000000    2.000000    2.000000
50%    125700.000000  2000.000000    3.000000    2.000000    3.000000
75%    147750.000000  2130.000000    3.000000    3.000000    3.000000
max    188300.000000  2440.000000    4.000000    3.000000    5.000000

Sample of removed outliers:
     Price    SqFt  Bedrooms  Bathrooms  Offers
14  176800  2590.0       4.0        3.0     4.0
28   69100  1600.0       2.0        2.0     3.0
33  139600  2280.0       5.0        3.0     4.0
47   90300  2050.0       3.0        2.0     6.0
65  111100  1450.0       2.0        2.0     1.0
```

➢ مصورسازی و تحلیل

تحلیل داده‌های قیمت مسکن

نمودار پراکندگی: ارتباط متراژ و قیمت

- توضیحات :این نمودار نشان می‌دهد که با افزایش متراژ (SqFt) قیمت نیز افزایش می‌یابد.
- نتیجه :ارتباط مثبت بین متراژ و قیمت نشان‌دهنده این است که بزرگ‌تر بودن خانه‌ها معمولاً به قیمت‌های بالاتر منجر می‌شود.

نمودار جعبه‌ای: تعداد اتاق خواب‌ها و قیمت

- توضیحات :این نمودار توزیع قیمت‌ها را بر اساس تعداد اتاق خواب نشان می‌دهد.
- نتیجه :خانه‌های با تعداد بیشتر اتاق خواب‌ها قیمت‌های بالاتری دارند. همچنین، در خانه‌هایی با اتاق خواب کمتر، قیمت‌ها تنوع بیشتری دارند.

نمودار جعبه‌ای: تعداد حمام‌ها و قیمت

- توضیحات :این نمودار ارتباط بین تعداد حمام‌ها و قیمت را نمایش می‌دهد.
- نتیجه :مشابه اتاق خواب، تعداد بیشتر حمام‌ها معمولاً به قیمت‌های بالاتر منجر می‌شود. قیمت‌ها برای خانه‌هایی با تعداد کم حمام، بیشتر پراکنده هستند.

نمودار جعبه‌ای: تعداد پیشنهادات و قیمت

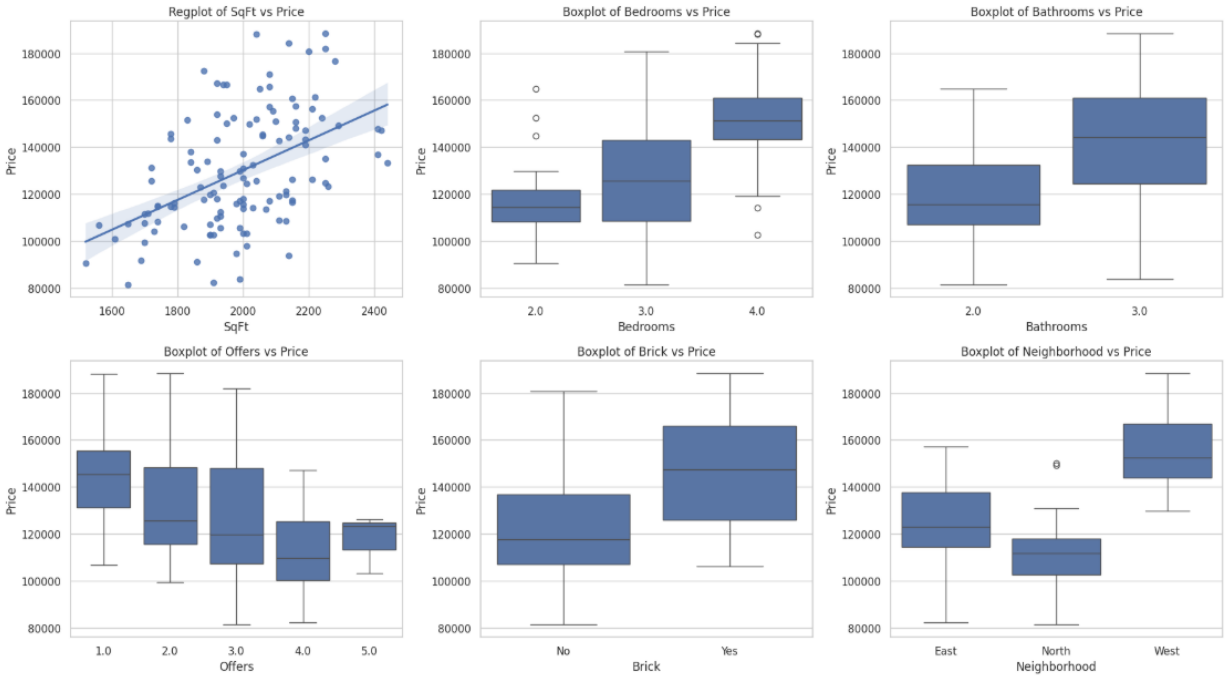
- توضیحات :این نمودار نشان می‌دهد که چگونه تعداد پیشنهادات دریافتی برای یک ملک بر قیمت تأثیر می‌گذارد.
- نتیجه :خانه‌هایی که پیشنهادات بیشتری دارند، قیمت‌های کمتری نیز دارند. این می‌تواند نشان‌دهنده تقاضای بیشتر برای آن ملک باشد و همچنین نشان دهنده سطح توان مالی افراد را نیز در بر داشته باشد.

نمودار جعبه‌ای: نوع ساخت (آجر) و قیمت

- توضیحات :این نمودار قیمت‌گذاری را بر اساس نوع ساخت، یعنی «آجر» یا «غیراجر» نشان می‌دهد.
- نتیجه :خانه‌های ساخته شده از آجر معمولاً قیمت‌های بالاتری دارند که می‌تواند به دلیل کیفیت بالاتر و جذابیت بیشتر این نوع ساخت باشد.

نمودار جعبه‌ای: محله و قیمت

- توضیحات :این نمودار تفاوت قیمت‌ها را بر اساس محله نمایش می‌دهد.
- نتیجه :به نظر می‌رسد که محله‌های غرب (West) قیمت‌های بالاتری تولید می‌کنند در حالی که محله‌های شرق (East) قیمت‌های کمتری دارند. این می‌تواند به عوامل مختلفی از جمله امکانات اطراف و دسترسی به خدمات مرتبط باشد.




```
Results Analysis:
Correlation between SqFt and Price: 0.48
Average price for 2.0 bedrooms: 116696.30
Average price for 4.0 bedrooms: 151492.31
Average price for 3.0 bedrooms: 126277.61
Average price for 2.0 bathrooms: 119865.22
Average price for 3.0 bathrooms: 142735.29
Average price for 2.0 offers: 132861.11
Average price for 3.0 offers: 126546.67
Average price for 1.0 offers: 144280.95
Average price for 4.0 offers: 112673.33
Average price for 5.0 offers: 117533.33
Average price for Brick houses: Yes: 145918.42, No: 122015.85
Average price in East neighborhood: 124904.55
Average price in North neighborhood: 111617.07
Average price in West neighborhood: 156517.14
```

➢ تقسیم بندی داده‌های آموزش و تست

در این مرحله با توجه به اینکه نیاز است مدل را با بخشی از داده‌ها آموزش دهیم و با بخشی دیگر تست کنیم، بعد از عملیات‌های پیش پردازش که انجام شد به سراغ این تقسیم بندی به صورت 90 درصد داده‌های آموزش و 10 درصد داده‌های تست می‌رویم.

```
X = df_clean[['SqFt', 'Bedrooms', 'Bathrooms', 'Offers', 'Brick', 'Neighborhood']]
y = df_clean['Price']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.1, random_state=42)
```

➢ انکودینگ و نرمال‌سازی

برای مدل‌هایی مانند رگرسیون خطی که به داده‌ها و ارتباط بین آن ها حساس هستند، نرمال‌سازی یک امر بسیار مهم است، از این رو ابتدا داده‌های آموزش را بر اساس میانگین 0 و انحراف معیار 1 نرمال میکنیم تا بتوانیم به درستی روابط بین داده‌ها را در آموزش مدل ترسیم کنیم و همچنین بعد از نرمال‌سازی داده‌های آموزش از جزئیات آن برای نرمال سازی داده‌های تست استفاده میکنیم و بعد از آن به سراغ انکود کردن و تبدیل داده‌های دسته‌ای به عددی میرویم، که به آن انکود کردن می گویند. با توجه به اینکه چه در لیبل انکودینگ و چه در وان هات انکودینگ از 0 و 1 فراتر نمیرویم نیازی به نرمال‌سازی نیست.

```
✅ Manual preprocessing completed.
📌 X_train shape: (108, 8)
📌 X_test shape: (12, 8)
📌 y_train mean: 129182.41, std: 24605.27
📌 Final columns: ['SqFt', 'Bedrooms', 'Bathrooms', 'Offers', 'Brick', 'Neighborhood_East', 'Neighborhood_North', 'Neighbor
hood_West']

🔍 Sample of X_train (encoded & normalized):
        SqFt  Bedrooms  Bathrooms    Offers  Brick  Neighborhood_East  \
11  -0.628203 -1.468229  -0.857360 -0.532140      1                  1
39  -1.299944  0.041949  -0.857360 -0.532140      0                  0
94   0.818625  1.552128   1.155572  0.441947      1                  0
96   2.317125  0.041949   1.155572  0.441947      0                  1
117 -0.369841  0.041949  -0.857360 -0.532140      0                  0

     Neighborhood_North  Neighborhood_West
11                    0                  0
39                    1                  0
94                    0                  1
96                    0                  0
117                   1                  0

🔍 Sample of y_train (normalized):
11    -0.251264
39    -0.852761
94     1.276864
96     0.167346
117   -0.462600
Name: Price, dtype: float64
```

➢ پیاده‌سازی مدل و تابع هدف

این بخش مهم‌ترین مرحله در این ساختار است و در اینجا از مدل رگرسیون خطی استفاده می‌کنیم که یکی از مدل‌های پایه در زمینه هوش مصنوعی شناخته میشود، که روش عملکرد آن با استفاده از فرمول خط هست، که در آن شیب خط و عرض از مبدا تعریف میکنیم و در اینجا این‌ها به عنوان پارامترها و یا وزن‌های مدل تلقی میشوند و شیب را نیز بایاس میخوانند.

بعد از آن استفاده از تابع هدف یا لاس هست که میزان عملکرد مدل را بررسی میکند، تا بتواند آپدیت‌های مورد نیاز را بر روی وزن‌ها انجام دهد و مدل را به‌درستی آموزش دهد، که در اینجا از Root Mean Square Error استفاده میکنیم که یکی از تابع‌های لاس معروف به شمار میرود.

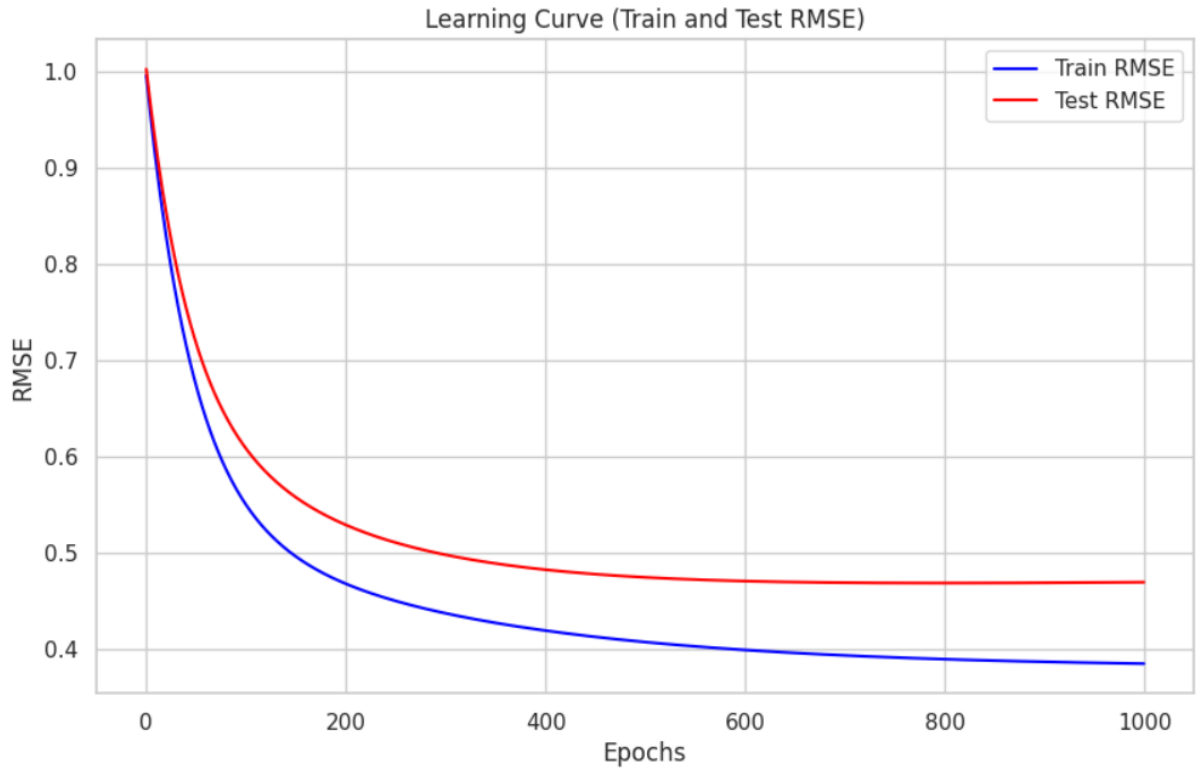
حال با استفاده از این موارد نیاز است که در هر دور وزن‌ها آپدیت شوند، که در اینجا بر اساس یکی از روش‌های مرسوم به نام گرادیان کاهشی این اتفاق می‌افتد و تا زمانی که به مقدار مناسب و بهینه همگرا نشویم این روند متوقف نخواهد شد تا اینکه تعداد دورهای مشخص شده را به پایان برسانیم.

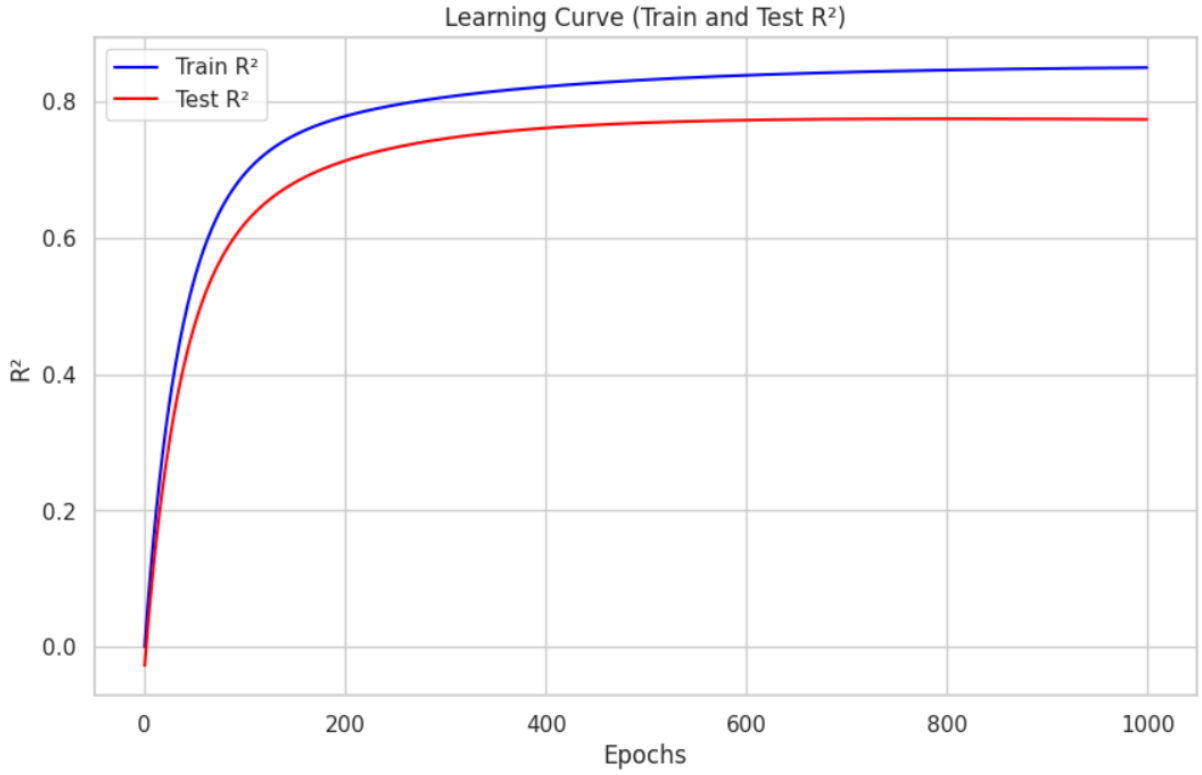
➢ ارزیابی و نتیجه‌گیری

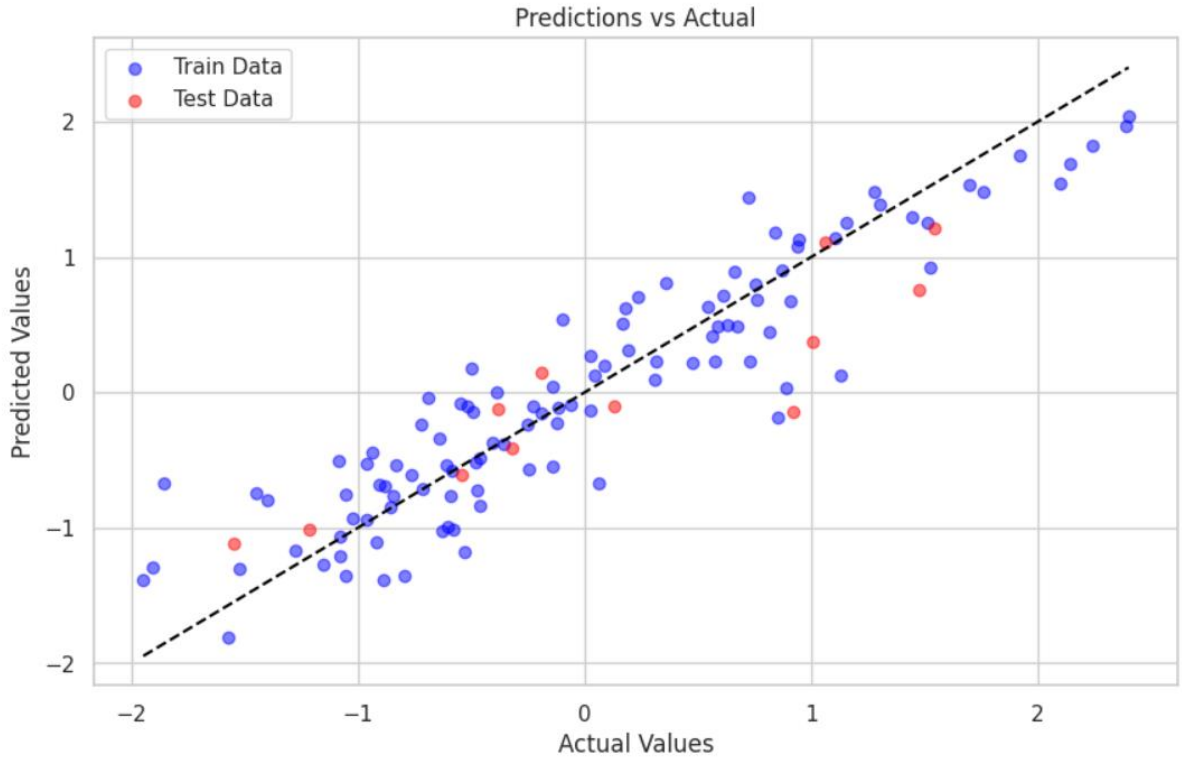
در پایان نتایج به دست آمده را بررسی میکنیم و سعی میکنیم نرخ آموزش و تعداد دور آموزش را به درستی تعیین کنیم تا بهینه ترین خروجی را دریافت کنیم که در اینجا نرخ آموزش را 0.05 و تعداد دور آموزش را 1000 در نظر گرفته ایم تا بهینه‌ترین حالت ممکن که در آن نیز باید مقدار تفاوت خطا داده آموزش و تست هم کم باشد را شناسایی کنیم به دلیل اینکه با مشکل بیش‌برازش مواجه نشویم و همچنین برای مقایسه از R^2 نیز بهروری کردیم. (نمودارهای تحلیلی در زیر قابل مشاهده است)

```
Epoch 0: Train RMSE = 0.9954, Test RMSE = 1.0020
Epoch 0: Train R² = 0.0000, Test R² = -0.0274
Epoch 100: Train RMSE = 0.5498, Test RMSE = 0.6081
Epoch 100: Train R² = 0.6948, Test R² = 0.6215
Epoch 200: Train RMSE = 0.4680, Test RMSE = 0.5290
Epoch 200: Train R² = 0.7789, Test R² = 0.7136
Epoch 300: Train RMSE = 0.4374, Test RMSE = 0.4980
Epoch 300: Train R² = 0.8069, Test R² = 0.7462
Epoch 400: Train RMSE = 0.4195, Test RMSE = 0.4826
Epoch 400: Train R² = 0.8224, Test R² = 0.7616
Epoch 500: Train RMSE = 0.4075, Test RMSE = 0.4747
Epoch 500: Train R² = 0.8324, Test R² = 0.7694
Epoch 600: Train RMSE = 0.3994, Test RMSE = 0.4708
Epoch 600: Train R² = 0.8390, Test R² = 0.7732
Epoch 700: Train RMSE = 0.3937, Test RMSE = 0.4692
Epoch 700: Train R² = 0.8435, Test R² = 0.7747
Epoch 800: Train RMSE = 0.3898, Test RMSE = 0.4688
Epoch 800: Train R² = 0.8466, Test R² = 0.7751
Epoch 900: Train RMSE = 0.3871, Test RMSE = 0.4691
Epoch 900: Train R² = 0.8488, Test R² = 0.7748

Final Train RMSE: 0.3852
Final Test RMSE: 0.4697
Final Train R²: 0.8503
Final Test R²: 0.7742
```


Learning Curve (Train and Test RMSE)
Train RMSE
Test RMSE
RMSE
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0
200
400
600
800
1000
Epochs


Learning Curve (Train and Test R²)
Train R²
Test R²
R²
0.8
0.6
0.4
0.2
0.0
0
200
400
600
800
1000
Epochs



در پایان خطی که به صورت خط‌چین مشاهده می‌شود همان خط مورد نظر ما برای پیش‌بینی قیمت منازل خواهد بود.

➢ منابع

- Ng, Andrew. Machine Learning (Coursera)
- دوره یادگیری ماشین، دانشگاه صنعتی شریف .شریفی زارچی، علی
- Géron, Aurélien. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow. O'Reilly Media, 2019
- OpenAI. ChatGPT

سید حسین حسینی دولت آبادی

موفق باشید 😊 😅