House Prices

Final Project

Lecturer: Dr. Tamar Shrot
By:
Name: Coral Avital ,ID: 205871163
Name: Yoni Ifrah ,ID: 313914723

The Problem

Predicting house prices based on historical data

Area

House Age

Popul
ation

Number
of
Rooms

Number of
Bedrooms

Area
Income

Address

Introduction

This project will deal with valuing an apartment through machine learning.
You can specify as follows:

"Ask an apartment buyer to describe his dream home, and it will probably not start with the height of the basement ceiling or the proximity to the east-west railroad tracks.
Our data set proves that these data have a greater impact on the valuation of an apartment in negotiations than the number of bedrooms or a white picket fence."

Predicting house prices can help buyers to know the price of the home so they can buy at fair value.
Similarly, predicting house prices can help homeowners seeking to sell to accurately price their listing.
Knowing the fair value, they can assess and plan the maximum discount rate they would be willing to sell for.
Lastly, property investors can know the trend of housing prices in a certain location.
Investors can use this model to predict home prices against the fair value or sale price to know if there is an arbitrage to leverage in their investing activities.

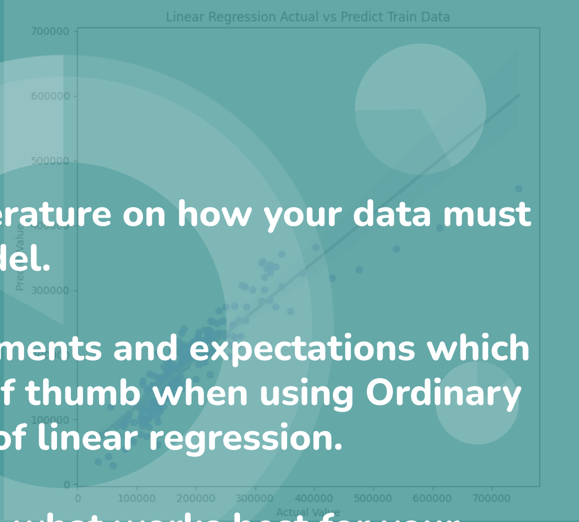
In this project, we will try to build a model that can fit the valuation of an apartment by samples of the features.
We will treat the problem as a regression problem

The Algorithms

The approaches we chose to implement:

- Linear Regression
- K nearest neighbors

Linear Regression



Linear regression is been studied at great length, and there is a lot of literature on how your data must be structured to make the best use of the model.

As such, there is a lot of sophistication when talking about these requirements and expectations which can be intimidating. In practice, you can use these rules more like rules of thumb when using Ordinary Least Squares Regression, the most common implementation of linear regression.

Try different preparations of your data using these heuristics and see what works best for your problem.

Linear Assumption. Linear regression assumes that the relationship between your input and output is linear. It does not support anything else. This may be obvious, but it is good to remember when you have a lot of attributes. You may need to transform data to make the relationship linear (e.g. log transform for an exponential relationship).

Remove Noise. Linear regression assumes that your input and output variables are not noisy. Consider using data cleaning operations that let you better expose and clarify the signal in your data. This is most important for the output variable and you want to remove outliers in the output variable (y) if possible.

Linear Regression

Remove Collinearity. Linear regression will over-fit your data when you have highly correlated input variables. Consider calculating pairwise correlations for your input data and removing the most correlated.

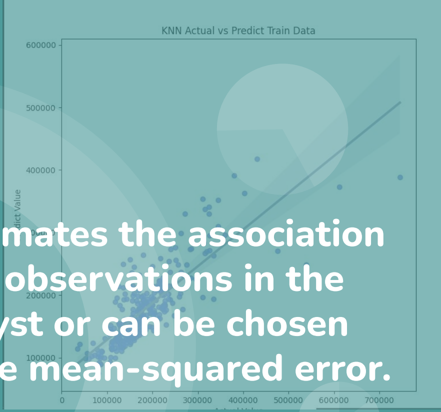
Gaussian Distributions. Linear regression will make more reliable predictions if your input and output variables have a Gaussian distribution. You may get some benefit using transforms (e.g. log or BoxCox) on your variables to make their distribution more Gaussian looking.

Rescale Inputs: Linear regression will often make more reliable predictions if you rescale input variables using standardization or normalization.

Why we chose to use Linear regression?

It is an algorithm of supervised machine learning where the predicted output is continuous with a constant slope determined by the prices of existing dwellings. It is used to predict values in a continuous range instead of classifying the values into categories. Linear regression is used to perform various tasks like predicting apartment prices.

K - nearest neighbors

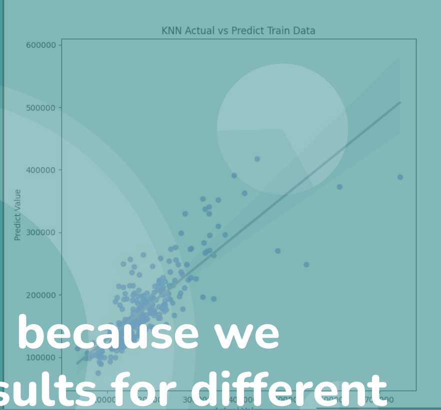


KNN regression is a non-parametric method that, in an intuitive manner, approximates the association between independent variables and the continuous outcome by averaging the observations in the same neighborhood. The size of the neighborhood needs to be set by the analyst or can be chosen using cross-validation (we will see this later) to select the size that minimizes the mean-squared error.

KNN is a lazy learning, non-parametric algorithm. It uses data with several classes to predict the classification of the new sample point.
KNN is non-parametric since it doesn't make any assumptions on the data being studied, i.e., the model is distributed from the data.

What does it mean to say KNN is a lazy algorithm? It means it doesn't use the training data points to make any generalizations.
You expect little to no explicit training phase,
The training phase is pretty fast,
KNN keeps all the training data since they are needed during the testing phase.

K - nearest neighbors



- <u>**Why we chose to use KNN algorithm?**</u>

We chose this approach to address the problem we raised because we recognized that there are features that could yield identical results for different apartments, i.e. using similarity in Hyperparameters for different apartments, we can deduce the value of the apartment being examined based on existing data.

- **We create 4 different models - 3 regressions of K neighbors :**

one where k = 3, one where k = 8, one where k = 10
and one where Hyperparameter.

Our System

The input

The data available (input) for predicting house prices is shown below

```
Index(['Address',
       'Price',
       'Area Population',
       'Avg. Area Number of Bedrooms',
       'Avg. Area Number of Rooms',
       'Avg. Area House Age',
       'Avg. Area Income'],
       dtype='object')
```

```
 #   Column                        Non-Null Count  Dtype
---  ------                        --------------  -----
 0   Avg. Area Income              5000 non-null   float64
 1   Avg. Area House Age           5000 non-null   float64
 2   Avg. Area Number of Rooms     5000 non-null   float64
 3   Avg. Area Number of Bedrooms  5000 non-null   float64
 4   Area Population               5000 non-null   float64
 5   Price                         5000 non-null   float64
 6   Address                       5000 non-null   object
```

The input

We are going to use the USA_Housing dataset. Since house price is a continuous variable, this is a regression problem. The data contains the following columns:

- **'Avg. Area Income': Avg. Income of residents of the city house is located in.**
- **'Avg. Area House Age': Avg Age of Houses in same city**
- **'Avg. Area Number of Rooms': Avg Number of Rooms for Houses in same city**
- **'Avg. Area Number of Bedrooms': Avg Number of Bedrooms for Houses in same city**
- **'Area Population': Population of city house is located in**
- **'Price': Price that the house sold at**
- **'Address': Address for the house**

The Data on Home Prices

5000

Examples in dataset

This refers to over 1400 houses evaluated.

7

Features in dataset

7 columns in total including Price (the target value).

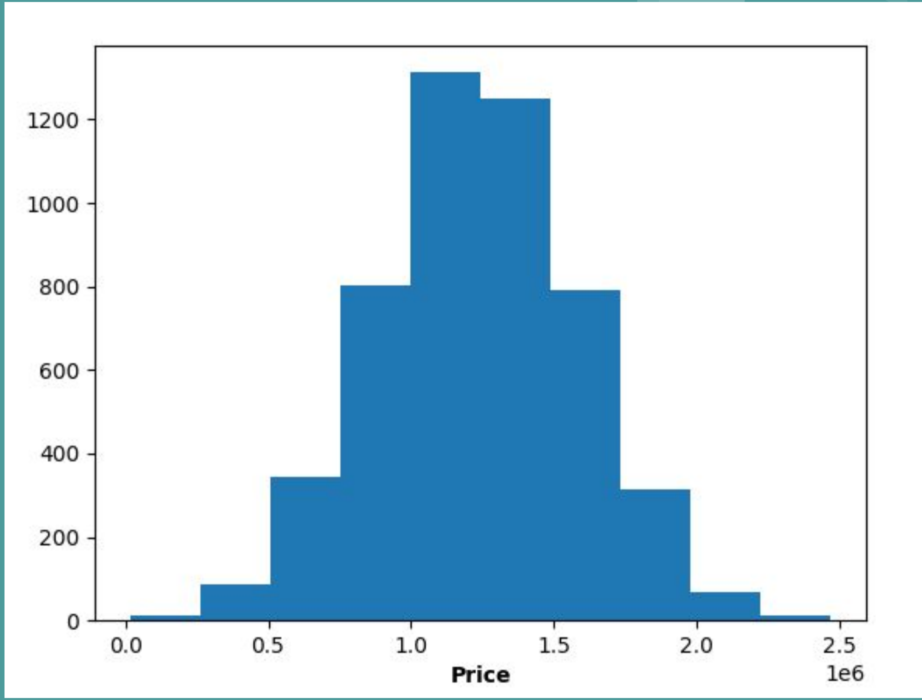
3500/1500

70/30 split

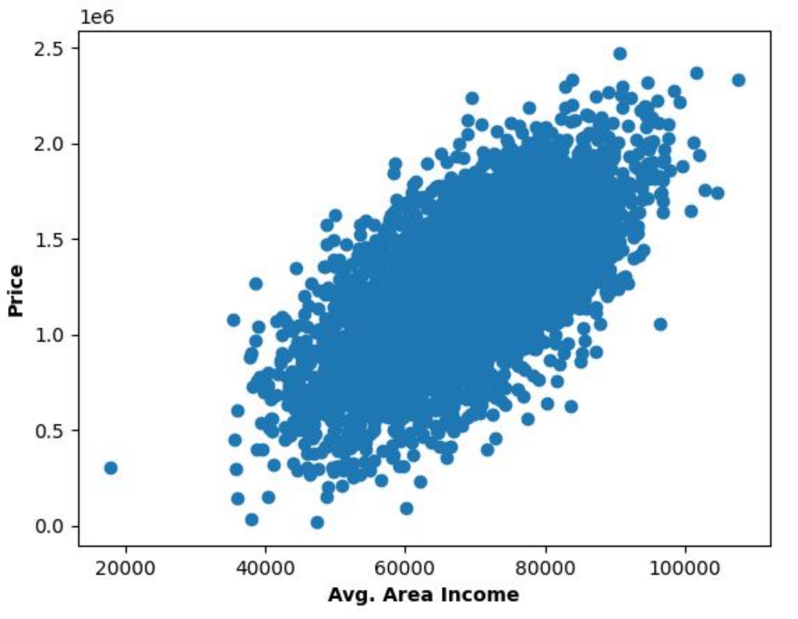
The data was split 3500 training and 1500 testing.

Price

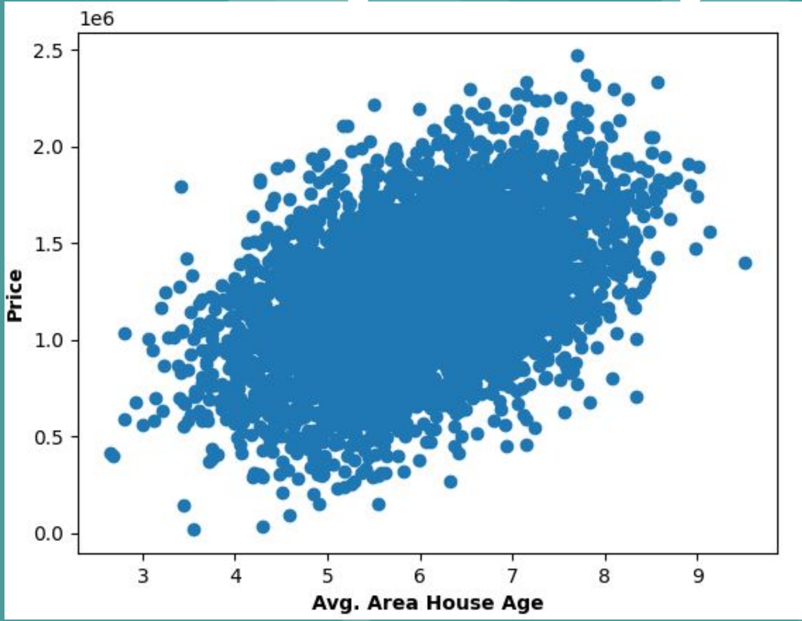
Price ratio of the houses that were sold



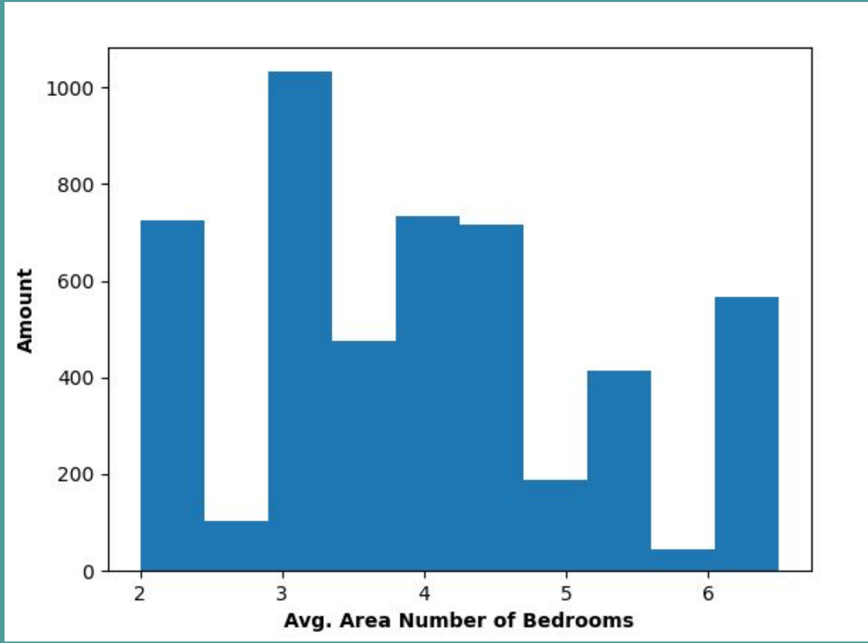
Price vs Avg.Area income



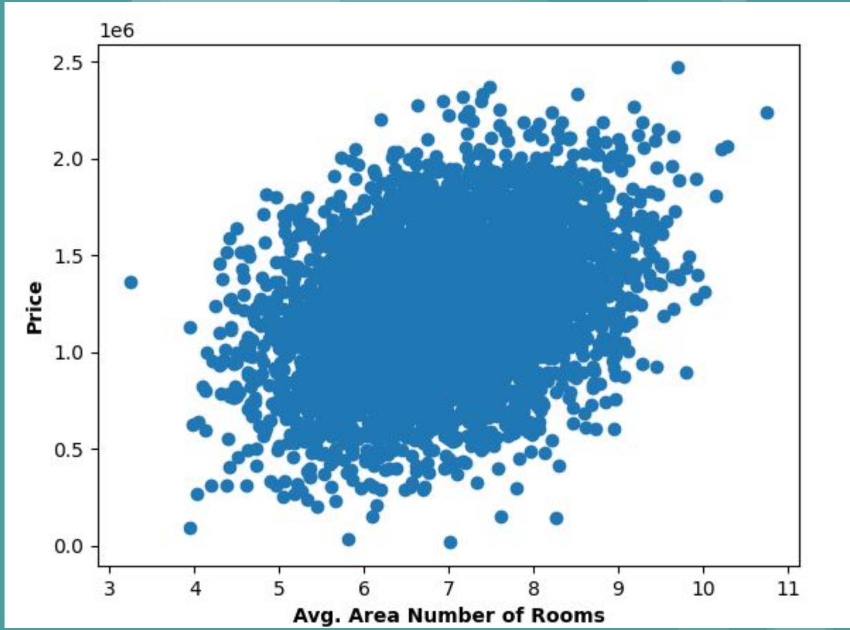
Price vs Avg.Area House Age



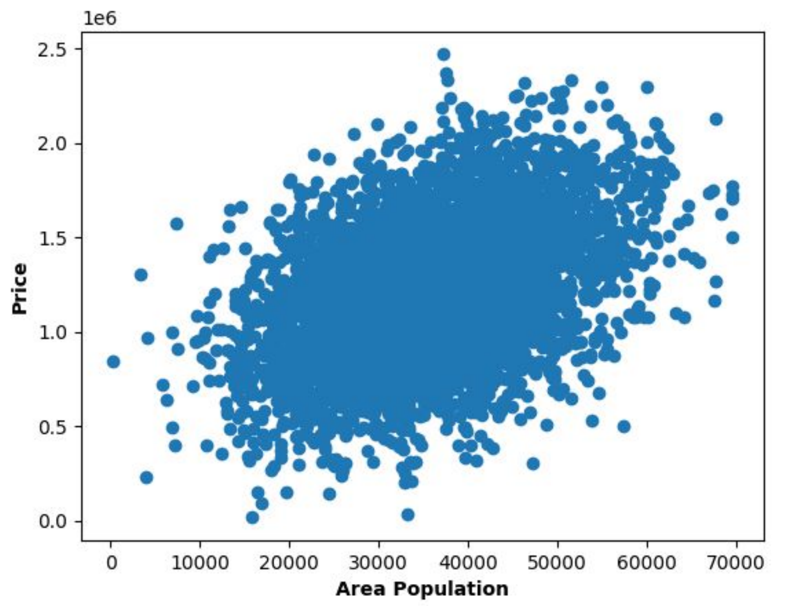
Price vs Avg.Area Number of BedRooms



Price vs Avg.Area Number of Rooms



Price vs Avg.Area Population



Here we can see all the correlation between all the feature with their ratio to each other

Evaluation

We used sklearn's MAE, MSE, RMSE R2 for reviewing our model's performance.

- MAE is the easiest to understand, because it's the average error.
- MSE is more popular than MAE, because MSE "punishes" larger errors, which tends to be useful in the real world.
- RMSE is even more popular than MSE, because RMSE is interpretable in the "y" units.
- R2 statistical measure of fit that indicates how much variation of a dependent variable is explained by the independent variable(s) in a regression model.

Hyperparameters for KNN

After calculating the values of the apartments using the KNN algorithm, we tried to optimize our assessment using Hyperparameters we found by using GridSearchCV.

GridSearchCV tries all the combinations of the values passed in the dictionary and evaluates the model for each combination using the Cross-Validation method. Hence after using this function we get accuracy/loss for every combination of hyperparameters and we can choose the one with the best performance.

Algorithm Performance

Linear Regression Algorithm

```
Model number 1 - Linear Regression
                                    Coefficient
Avg. Area Income                  232679.724643
Avg. Area House Age               163841.046593
Avg. Area Number of Rooms         121110.555478
Avg. Area Number of Bedrooms        2892.815119
Area Population                   151252.342377
Test set evaluation:
_____________________________________
MAE: 81135.56609336878
MSE: 10068422551.40088
RMSE: 100341.52954485436
R2 Square 0.9146818498754016
__________________________________
Train set evaluation:
_____________________________________
MAE: 81480.4997317489
MSE: 10287043161.197224
RMSE: 101425.06180031257
R2 Square 0.9192986579075526
__________________________________
```

KNN Algorithm

```
Model number 2 - KNN with 3 neighbors
                              Coefficient
Avg. Area Income                 0.927144
Avg. Area House Age              0.927144
Avg. Area Number of Rooms        0.927144
Avg. Area Number of Bedrooms     0.927144
Area Population                  0.927144
Test set evaluation:
_____________________________________
MAE: 110416.92048474538
MSE: 18721119493.630993
RMSE: 136825.14203767886
R2 Square 0.8413603247873304
__________________________________
Train set evaluation:
_____________________________________
MAE: 77330.57340404775
MSE: 9286934654.968918
RMSE: 96368.74314303843
R2 Square 0.9271444594100811
__________________________________
```

```
Model number 2 - KNN with 8 neighbors
                              Coefficient
Avg. Area Income                 0.901984
Avg. Area House Age              0.901984
Avg. Area Number of Rooms        0.901984
Avg. Area Number of Bedrooms     0.901984
Area Population                  0.901984
Test set evaluation:
_____________________________________
MAE: 99802.51338430308
MSE: 15644238959.878263
RMSE: 125076.93216528083
R2 Square 0.8674333023519881
__________________________________
Train set evaluation:
_____________________________________
MAE: 89134.64666406093
MSE: 12494121124.127361
RMSE: 111777.1046508513
R2 Square 0.9019842410318681
__________________________________
```

```
Model number 2 - KNN with 10 neighbors
                              Coefficient
Avg. Area Income                 0.898455
Avg. Area House Age              0.898455
Avg. Area Number of Rooms        0.898455
Avg. Area Number of Bedrooms     0.898455
Area Population                  0.898455
Test set evaluation:
_____________________________________
MAE: 98991.63253116103
MSE: 15392853397.117458
RMSE: 124067.9386349167
R2 Square 0.8695635021000906
__________________________________
Train set evaluation:
_____________________________________
MAE: 90617.98276965883
MSE: 12944035896.934874
RMSE: 113771.85898514128
R2 Square 0.8984546820105022
__________________________________
```

You can see that for 10 neighbors we got the best score

KNN with Hyperparameter

```
Best n_neighbors: 14
                                  Coefficient
Avg. Area Income                      0.89383
Avg. Area House Age                   0.89383
Avg. Area Number of Rooms             0.89383
Avg. Area Number of Bedrooms          0.89383
Area Population                       0.89383

Model number 2 - KNN with Hyperparameters
Test set evaluation:
_____________________________________
MAE: 98560.4669536799
MSE: 15429624130.213854
RMSE: 124216.03813603883
R2 Square 0.8692519129797002
__________________________________
Train set evaluation:
_____________________________________
MAE: 92408.04533992754
MSE: 13533576505.608768
RMSE: 116333.90093007614
R2 Square 0.8938297652494409
__________________________________
```


```
Test set evaluation:
_____________________________________
MAE: 81135.56609336878
MSE: 10068422551.40088
RMSE: 100341.52954485436
R2 Square 0.9146818498754016
__________________________________
Train set evaluation:
_____________________________________
MAE: 81480.4997317489
MSE: 10287043161.197224
RMSE: 101425.06180031257
R2 Square 0.9192986579075526
__________________________________
```

According to the results from R square we can see that the best score from all the models is 91% accuracy which belongs to the linear regression.

Our insights

First of all we learned how much we are in the right field!
We are familiar with the 3 pillars of AI that are now freely available:
1. Data and Algorithms - Kaggle, Google Dataset Search
2. Computing - Google Colab, Kaggle kernels
3. Education - School of Artificial Intelligence, KhanAcademy, Fast .AI

Second, we learned how to use some of these APIs,
all of which are very similar -
Create a model instance, use a matching method to train it, predict a prediction method, etc..
We're glad we chose to build this project from scratch,
It helped us understand exactly how to collect data, look for features within
Data, create a training kit and test kit, create and train our models, and then use them to predict different data!

- **We learned about different algorithms of regression learning: knn regressor and linear regression**
- **We learned about different algorithms statistical measure: MAE, MSE, RMSE and R2**

Conclusions

We were able to build a model with good performance for identifying the degree of value of an apartment!
the model produced excellent results.

Bibliography

- Predicting House Prices EDA+ Model
https://www.kaggle.com/miraculusik/predicting-house-prices-eda-model

- House Prices
https://www.kaggle.com/dataabel/house-prices

- Linear Regression House Prices Predict
https://www.kaggle.com/faressayah/linear-regression-house-price-prediction

- Hyperparameters for KNN
https://medium.datadriveninvestor.com/k-nearest-neighbors-in-python-hyperparameters-tuning-716734bc557f