**FINAL PROJECT**

**Step 3:**

**CREATING A MACHINE LEARNING MODEL TO PREDICT HOUSING PRICES**

The next step would be to use the data to estimate housing prices.

|  |  |
| --- | --- |
| Int64Index: 110723 entries, 1 to 134877 |  |
| Data columns (total 46 columns): |  |
| # Column Non-Null Count Dtype | # Column Non-Null Count Dtype |
| --- ------ -------------- ----- | --- ------ -------------- ----- |
| 0 address 110723 non-null object | 29 ProfessionalUse 110723 non-null float64 |
| 1 postal\_code 110723 non-null object | 30 DistanceFromSea 110723 non-null object |
| 2 floor 110723 non-null float64 | 31 EnergyCat 110723 non-null object |
| 3 heating 110723 non-null object | 32 Luxury 110723 non-null float64 |
| 4 homeType 110723 non-null object | 33 StudentAccomodation 110723 non-null float64 |
| 5 numBathrooms 110723 non-null float64 | 34 SummerHouse 110723 non-null float64 |
| 6 numBedrooms 110723 non-null float64 | 35 BuildingZone 110723 non-null object |
| 7 parking 110723 non-null float64 | 36 category 110723 non-null float64 |
| 8 price 110723 non-null float64 | 37 lat 110723 non-null float64 |
| 9 pricePerSqm 110723 non-null float64 | 38 lng 110723 non-null float64 |
| 10 area 110723 non-null float64 | 39 place\_id 110723 non-null object |
| 11 year 110723 non-null float64 | 40 numWCBath 110723 non-null float64 |
| 12 LivingRooms 110723 non-null float64 | 41 numRooms 110723 non-null float64 |
| 13 Kitchens 110723 non-null float64 | 42 areaPerRoom 110723 non-null float64 |
| 14 WC 110723 non-null float64 | 43 areaPerWCBath 110723 non-null float64 |
| 15 Orientation 110723 non-null object | 44 posFeatures 110723 non-null float64 |
| 16 NewlyBuilt 110723 non-null float64 | 45 negFeatures 110723 non-null float64 |
| 17 Storage 110723 non-null float64 |  |
| 18 Views 110723 non-null float64 |  |
| 19 RoofTop 110723 non-null float64 |  |
| 20 SwimmingPool 110723 non-null float64 |  |
| 21 RoadFront 110723 non-null float64 |  |
| 22 Corner 110723 non-null float64 |  |
| 23 Renovated 110723 non-null float64 |  |
| 24 YearRenovated 110723 non-null object |  |
| 25 NeedsRenovation 110723 non-null float64 |  |
| 26 ListedBuilding 110723 non-null float64 |  |
| 27 Neoclassico 110723 non-null float64 |  |
| 28 Unfinished 110723 non-null float64 |  |
| dtypes: float64(36), object(10) |  |
| memory usage: 39.7+ MB |  |

**Linear Regression**

I first applied a linear regression model (Multiple Linear Regression) to see if there is a strong linear relationship between the pricePerSqm and price (as y-dependent variable) and the rest of the data as independent variables.

Apply scaling to the variables to help see all the variables from the same lens (same scale), it will also help my models learn faster.

First, I tried to see if any of the variables individually explained the variation in either ‘price’ or ‘pricePerSqm’ by running a linear regression based on each variable alone.

The results show area explained 63% of variation in price (see VarScore), understandable, but very little of pricePerSqm.



Decided to run multivariable analysis with the ones that showed some explanation of variance >= 0.05 for both variables.



It did improve explanation of variance to 66% for price and 23% for pricePerSqm.

The residual error distributions for above table seen in graphs below:

Chart, histogram

Description automatically generatedChart, histogram

Description automatically generated

I also tested if running this separate for each homeType or category provided for a stronger explanation of variance but it does not seem to improve the VarScore:

The residual error distributions for above table seen in graphs below:

Chart, histogram

Description automatically generatedChart, histogram

Description automatically generatedChart, histogram

Description automatically generatedChart, histogram

Description automatically generated

Finally, I ran the multi-variate regressions for all the different home types to see if this model does particularly well predicting the dependent variable for any individual home type.

Also I could not find any significant improvement in focusing on just on home type even if listing classified as Houses or Apartments, the two main categories, did have slightly improved VarScores.



**Deep Learning – Keras Regressions**

Created a neural network model for predicting house prices with as many neurons as the X independent variables we have as a start, 4 hidden layers and 1 output layer due to predict house price, using the Adam optimization algorithm to optimize Mean Squared Error (MSE). Also ran same model to predict pricePerSqm instead of price. Initially ran model for batches of 32 and 64 epochs using 67% of sample for testing and 33% for validation**. My initial run already shows an improved variance score of 0.70 from 0.66 with respect to predicting price and 0.54 from 0.23 with respect to predicting pricePerSqm vs linear regression.**

Checking for the optimal list of x-variables, I ran the above model for different set of x-variables.



Based on the above results, I decided to use the full 37 independent variables against both price and pricePerSqm. To find the optimal model and parameters that minimize MSE and explain most of the variance, I run a series of model training sessions with all the combinations of the following parameters and different models:

Batch Size = 16, 32

Models = Adam, Adadelta, Adamax, Nadam, RMSprop

Learning Rate = 0.1, 0.01, 0.001

Epochs = 64, 128, 256, 516

Dropout Probability = 0.05, 0.5

It took around 30 hours for my python interpreter running on CPU & GPU to calculate all the permutations. The following table shows some of the main results. **My deep learning model shows considerable improvement in explaining price from 0.62 in multi-variate linear regression to 0.73-0.75 and for pricePerSqm from 0.23 is in multi-variate linear regression to 0.55-0.57**:



Highlighted the most promising models for both price and pricePerSqm. When looking at the history.history convergence/learning rate graphs for both the training and testing datasets included below, I realised that more epochs may be needed to have a smoother projection of the MSE error for testing set. From the analysis, the model seems to achieve higher prediction rate and faster/smoother learning if dropout rate probability at 0.05 and lower learning rate 0.01 – 0.001.

Chart, line chart

Description automatically generated

Chart, line chart

Description automatically generated

Chart, line chart

Description automatically generated

I further tested for 256 and 516 epochs and increased batch size to 32 and got improved for the Adamax models that seems to give the better results.



Based on the above results, I chose the highest scoring variance settings for predicting price and pricePerSqm. The final model used had the following parameters:

Batch Size = 32

Models = Adamax

Learning Rate = 0.01

Epochs = 516

Dropout Probability = 0.05

As you can see in graphs on following 4 pages, the model converges quickly and relatively smoothly to the minimum MSE, for price and pricePerSqm. I get slightly higher variance score for LR=0.01 than LR=0.001 but the convergence of the learning graphs is somewhat smoother for LR=0.001. I still used the parameters that gave me the highest variance score.

As a final step, I estimated the model predicted values across the entire cleaned up database of listings without dropping out the listings with tail results for price and pricePerSqm (by uploading file FinalDataSetNoTrim.txt) . I stored the resulting database into a ‘~’ delimited text file REListingsPredictions516\_32\_01.txt, and run against these listing the prediction model.

Stored the results in extra fields showing the model predicitons for each listing of price and pricePerSqm as well as how many standard deviations is the prediction off the actual listed price or pricePerSqm.

The Django-based app will load into its model/SQL the data from the above file.

**Model for price, LR=0.001. Batch Size=32, Dropout Rate=0.05, Epochs = 516, Model = Adamax**

**VARIANCE SCORE = 0.70**Chart, line chart

Description automatically generated

**Model for pricePerSqm, LR=0.001. Batch Size=32, Dropout Rate=0.05, Epochs = 516, Model = Adamax**

**VARIANCE SCORE = 0.57**Chart, line chart

Description automatically generated

**Model for price, LR=0.01. Batch Size=32, Dropout Rate=0.05, Epochs = 516, Model = Adamax**

**VARIANCE SCORE = 0.75**

Chart, line chart

Description automatically generated

**Model for pricePerSqm, LR=0.01. Batch Size=32, Dropout Rate=0.05, Epochs = 516, Model = Adamax**

**VARIANCE SCORE = 0.59**

Chart, line chart

Description automatically generated