Main Project Stage 1

Data Source
King's County Housing Data
Sourced from Kaggle: [https://www.kaggle.com/datasets/harlfoxem/housesalesprediction]
It is under a CC0: Public Domain license.
The dataset is for housing in King's Country sold between May 2014 and May 2015, with metrics for each individual house sale.

Cleaning
Not too much cleaning had to be done. I removed data which I felt was unusable (the specific ID of the sale and the zip code). I moved the price of the house to the first column, and I also cleaned up the date of sale so that it was numeric.

Visualization 1: Histograms of Some Housing Attributes
I showed the bedroom count, bathroom count, and floor count in a histogram. You can see the general distributions of how many bedrooms/bathrooms/floors each house has. Note that the bedroom count histogram appears to go up to 20+: this is because there's a single house at element 15870 with 33 bedrooms. I chose not to clean this because I do not know the true bedroom count, but I am noting that this is likely an error in the data.

Visualization 2: House Sale Locations
These are the locations of house sales in the dataset. The reason I did not clean out the location data is because, as seen in the visualization, it forms coherent shapes that may help the neural network.

Visualization 3: Histogram of House Price (in millions)
Thsi is the distribution of house prices, with the x-axis labeled in millions of dollars. The neural network will predict housing prices, so this is the distribution of the ouput parameter.

Main Project Stage 2

Data Processing
I picked the house price as the output data and everything else in the cleaned data as input. I chose 'price' as the ouput data because I felt like it was the parameter which would be the most correlated to everything else (floor size, whether it was a waterfront property, location, date of sale etc.). I was able to use everything else as input because it was already in a numeric form.

I split off 40000 of the 21613 total input cases as test cases, which is about 20% of the data. I felt like this was enough testing data to get an idea of how accurate the neural net was, while still leaving enough data for training purposes.

Basic Model
I used a mean squared error loss function, as that is appropriate for house prices which are not categorical and not bound between 0 and 1. I used a 128 and a 64 node fully connected layer just as a basic neural network which should be able to do some level of predictions.

I chose 10**(-9) as the learning rate because anything significantly higher would lead to divergent behaviour. I chose 10 epochs as that allowed it mostly converge.

Results
It ended with a mean squared error of 27421886464.0 on the training data and 31759446016.0 on the test data. Converting it to the standard error, it was 165 595.6 on training and 178 211.8 on test. Given that the prices were generally around 500 000, this error is quite large but indicates that the model is recognizing some patterns.

Main Project Stage 3

Since the houses have longitude/latitude information, I plotted the locations where the house prices are overestimated vs. where they are underestimated in the testing data set. (Blue is underestimate, orange or red is overestimate.) We can tell on the graph that the model consistently underestimates house prices near the middle of the county.

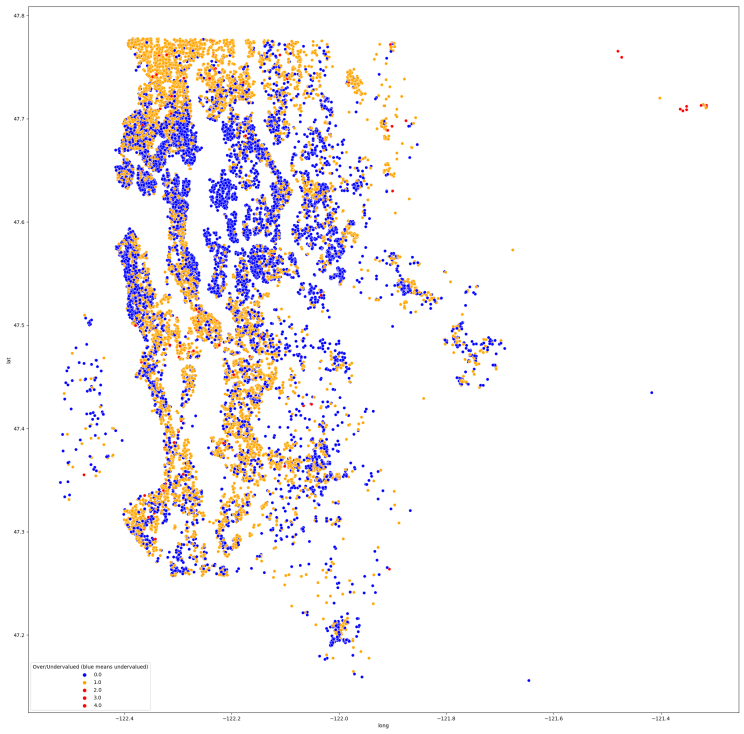

Figure 1: House Price and Location

This means that the model is not correctly considering the affect of location on house price.
To fix this, I added two more layers to the neural network so it could handle more complexity, changed the optimizer to ADAM, and increased the number of epochs to 100. This means that it should actually fit to the location data.

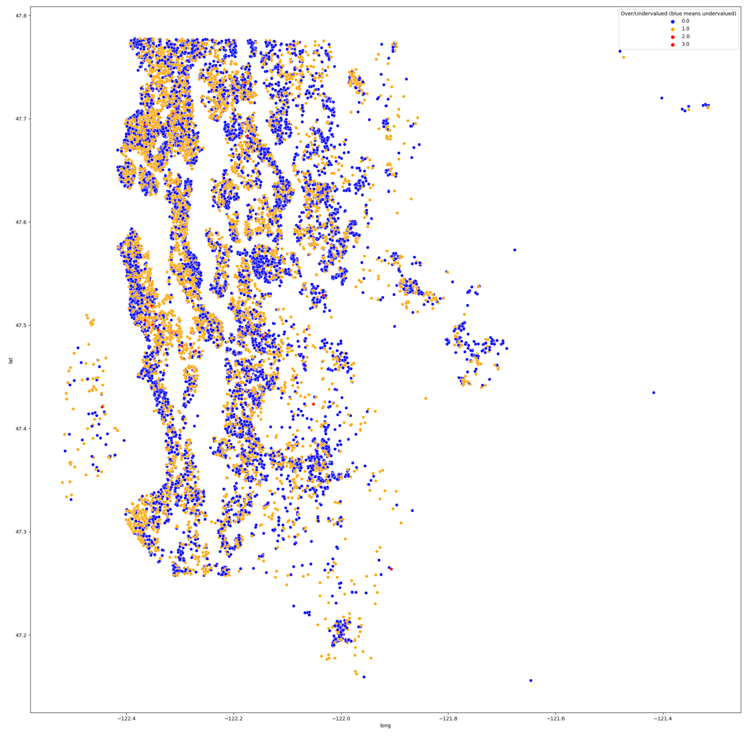

Figure 2: House Price and Location, revised

Here, the chance that the model over- or underpredicts the price is much more homogenous.

Main Project Stage 4

Model Updates

Categoric data was already functional, did not need to process any further.

The model had weird outliers (the location being far away from everything else, or the number of bedrooms being labeled as 33). Therefore I added a L1 normalizer to the first layer, which shoudl reduce the imapct of outliers on the model.

I also doubled the learning rate and halved the number of epochs to speed up training, since the model was not overfitting due to learning rate.

Model Results
Train / Test Error: 101671.4 standard error / 122714.0 standard error