Ashit Patel
Jaimin Patel
Cmsi 643
Dr. Lei huang

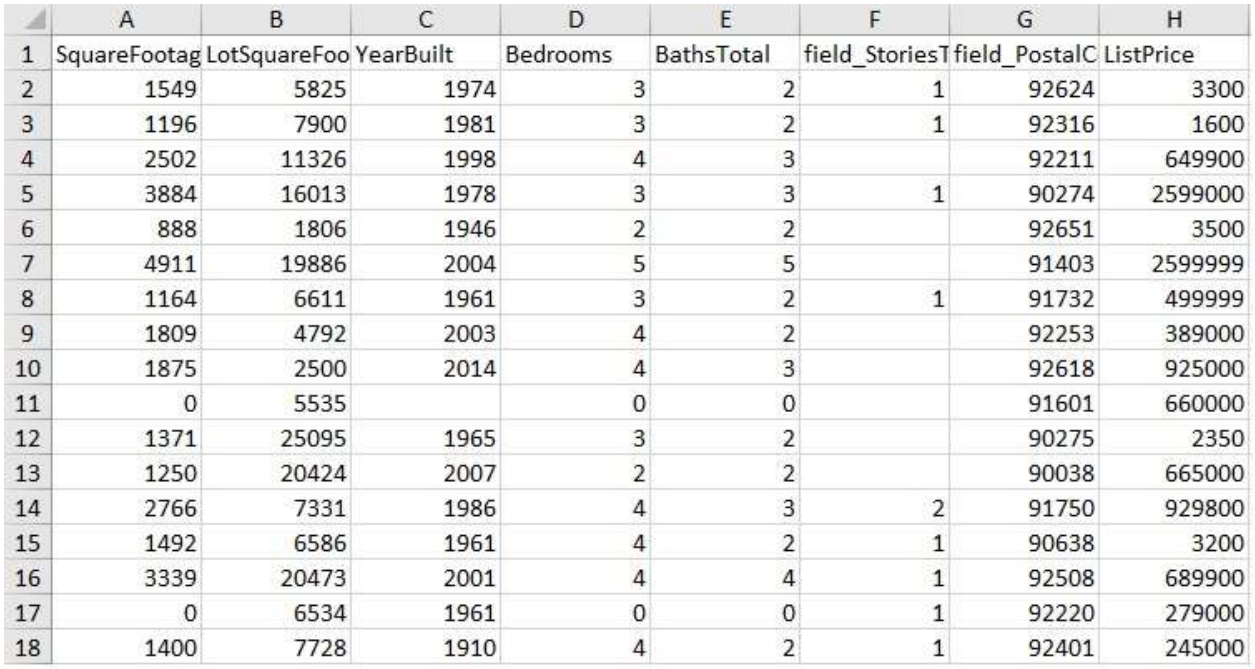
California Housing Price Estimation Model Report

Introduction

The goal of this project is to build a model that can estimate the price of houses in southern California. The given set data consists of features like square footage, lot square footage, year built, number of bedrooms, number of bathrooms, number of field stories, and the zip code, and at the end List Price to Predict as shown in the image in bottom of the page.

We were given a dataset that included 10,000 samples, in excel datasheet. We used pandas in order to import data into our program and preprocess it because of much noise or improper data samples and then build a model to train on the data, and eventually analyze the results.

Since this is a real-life dataset, it included many improper data for model, missing values, invalid values, etc. Therefore, we had to filter out and clean the dataset before training the model.



	A	B	C	D	E	F	G	H
1	SquareFootag	LotSquareFoo	YearBuilt	Bedrooms	BathsTotal	field_StoriesT	field_PostalC	ListPrice
2	1549	5825	1974	3	2	1	92624	3300
3	1196	7900	1981	3	2	1	92316	1600
4	2502	11326	1998	4	3		92211	649900
5	3884	16013	1978	3	3	1	90274	2599000
6	888	1806	1946	2	2		92651	3500
7	4911	19886	2004	5	5		91403	2599999
8	1164	6611	1961	3	2	1	91732	499999
9	1809	4792	2003	4	2		92253	389000
10	1875	2500	2014	4	3		92618	925000
11	0	5535		0	0		91601	660000
12	1371	25095	1965	3	2		90275	2350
13	1250	20424	2007	2	2		90038	665000
14	2766	7331	1986	4	3	2	91750	929800
15	1492	6586	1961	4	2	1	90638	3200
16	3339	20473	2001	4	4	1	92508	689900
17	0	6534	1961	0	0	1	92220	279000
18	1400	7728	1910	4	2	1	92401	245000

Proposed Solution

Data Preprocessing

The first task for the dataset was to deal will null values. We just filled null values to '0' so that we can deal with such values later.

The second step was to make sure that each of the 7 features has a direct correlation with the price of the house. For example, the lot square footage has a direct correlation with the price because a property with a higher lot square footage is usually more expensive than a property with a lower lot square footage. The first 6 features all have direct correlations with the price, but the last one, zip code, does not have a correlation with the price of the property. A higher zip code does not necessarily mean higher prices for the properties in that zip code. Therefore, we just removed the column of zip code.

We didn't replace anything with zip code because any such data may cause trouble for our model, may be the data would make model worse or maybe it didn't match out existing data, so we just decided to go with just removing the zip code column.

We then tried to train the model using the data after these few modifications, but our model did not perform well and the validation error was still very high as shown in Figure 1.

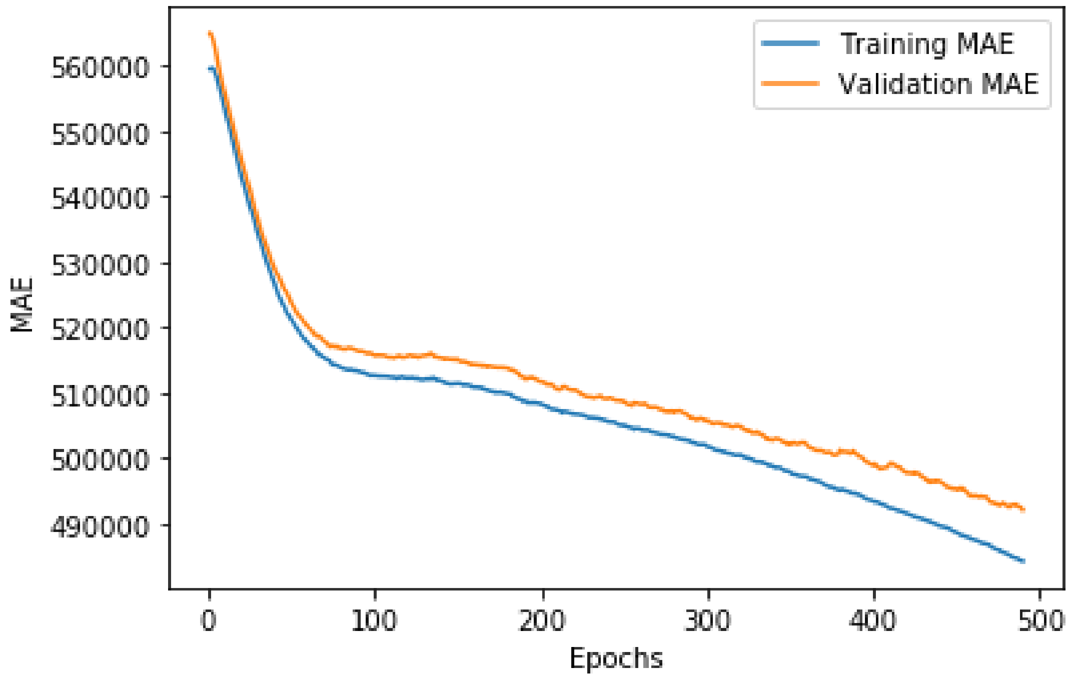


Figure 1

Since our model performed very badly with testing score of 555981.2, we decided to filter out the data even more to remove outliers. To do so, we graphed the data distribution of each of the 6 features and studied the graphs to know the outliers for each feature and delete entries that include features, which are recognized as outliers.

- Square Footage Data Distribution

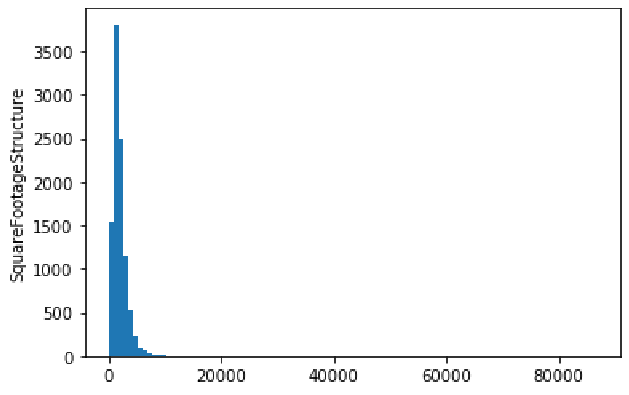


Figure 2

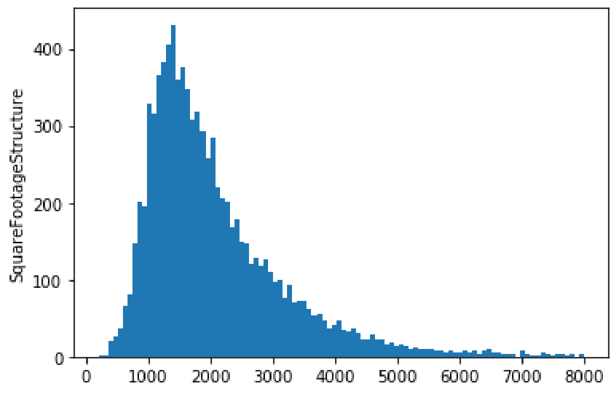


Figure 3

As shown in Figure 2, the square footage ranges from 0ft^2 to almost 80,000ft^2; however, most entries have a square footage of less than 8000ft^2. Therefore, we removed entries with square footage of more than 8,000ft^2, and the new distribution is shown Figure 3.

- Lot Square Footage Data Distribution

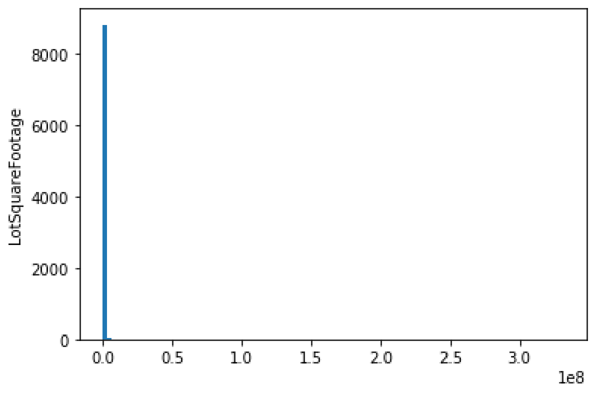


Figure 4

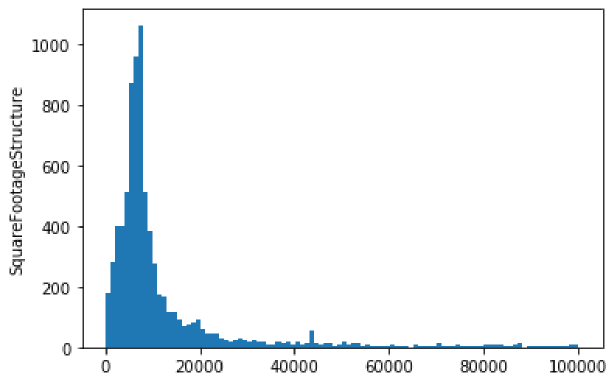


Figure 5

As shown in Figure 4, the lot square footage ranges from 0ft^2 to 300,000,000ft^2; however, most entries have a lot square footage of less than 10,000 ft^2. Therefore, we removed entries with lot square footage of more than 100,000ft^2, and the new distribution is shown Figure 5.

- Year Built Data Distribution

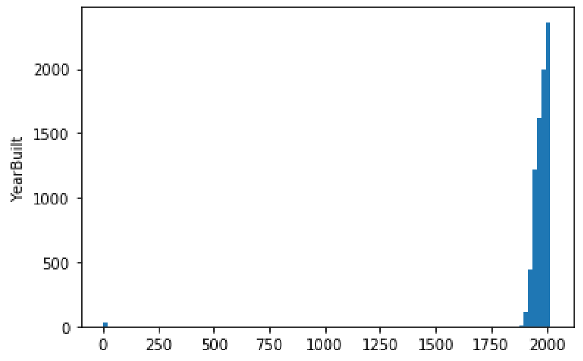


Figure 6

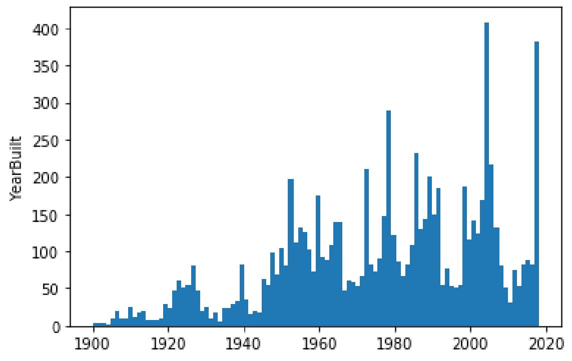


Figure 7

As shown in Figure 6, most entries were built around the year 2000, and some entries had a '0' for the year feature.
There were also a couple of entries that were built in early 1900's. Therefore, we removed entries that had no value for the year built or built before 1900, and the new distribution is shown in Figure 7.

- Number of Bedrooms Data Distribution

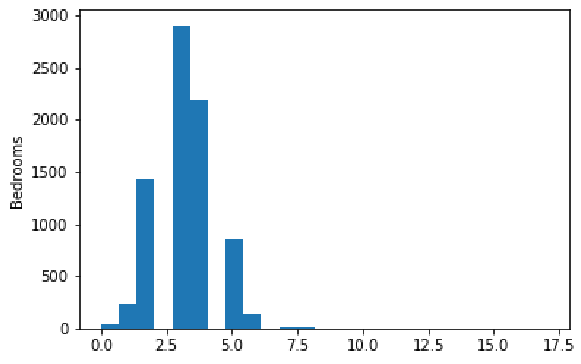


Figure 8

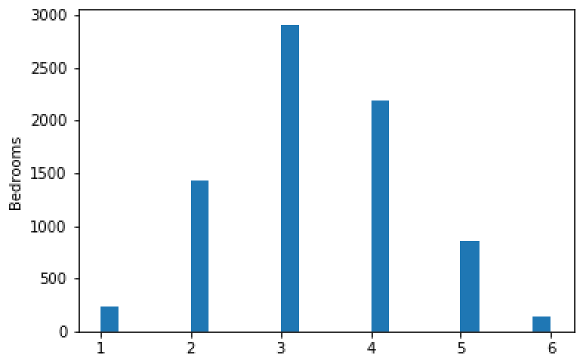


Figure 9

As shown in Figure 8, the number of bedrooms ranged from 0 bedrooms to 17 bedrooms. However, most entries have from 1 to 6; therefore, we eliminated entries that had 0 bedrooms or more than 6 bedrooms, and the new distribution for the bedroom count is shown in Figure 9.

- Number of Bathrooms Data Distribution

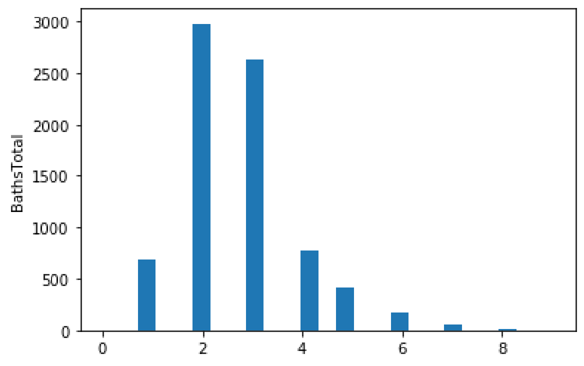


Figure 10

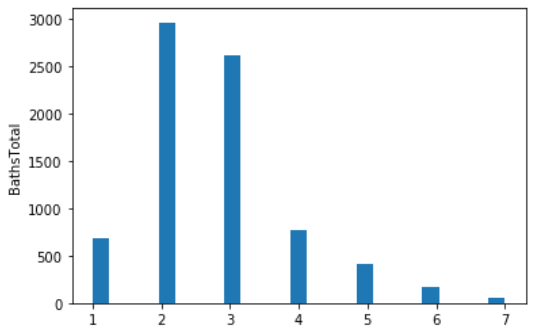


Figure 11

As shown in Figure 10, the number of bathrooms ranges from 0 to 10; therefore, we eliminated entries that had 0 bathrooms or more than 7 bathrooms, and the new distribution for the bathroom count is shown Figure 11.

- Number of Field Stories Data Distribution

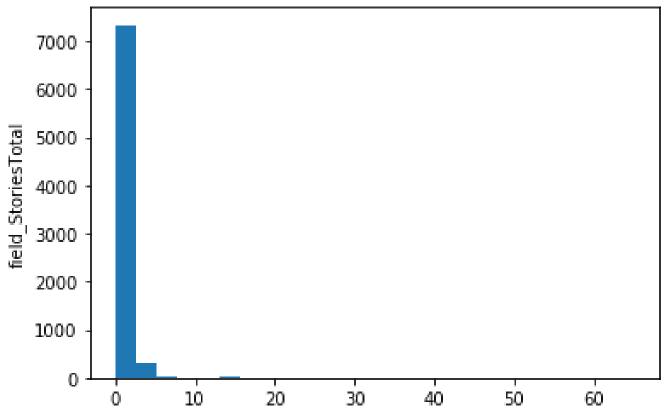


Figure 12

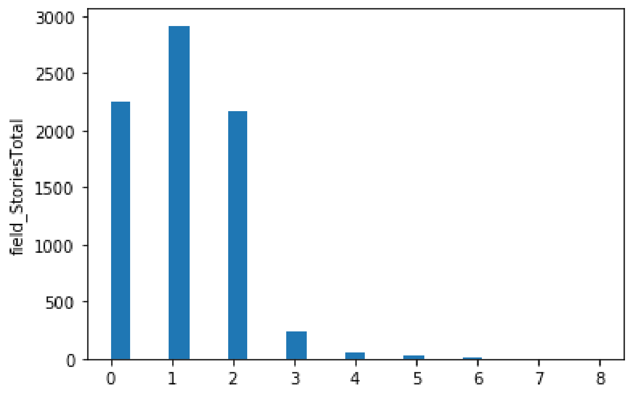


Figure 13

As shown in Figure 12, the number of field stories ranges from 0 to 60; however, most entries have from 0 to 5; therefore, we eliminated entries that had more than 3 field stories, and the new distribution for the field stories is shown Figure 13.

Eventually, we removed a total of 2381 entries from the original 10,000 entries, and we were left with 7619 entries. The 7619 were then split into 2 datasets: training set (6095 entries) and testing set (1524 entries) i.e. 80-20% ratio of training and testing. After splitting the datasets, we normalized datasets by mean and standard deviation as shown in the book.

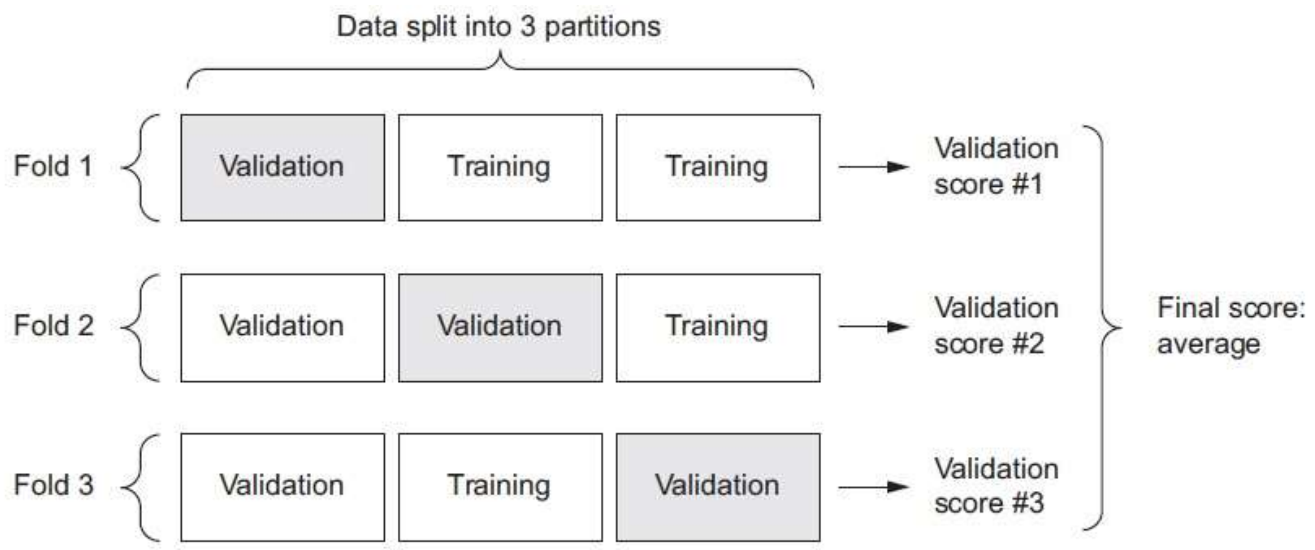
Model

We built a 4-layer fully-connected sequential neural network. The first layer (input) has 128 neurons, the second layer (hidden) has 64 neurons, the third layer (hidden) has 16 neurons and the fourth layer (output) has 1 neuron.

For the model, we used the RMSprop (Root Mean Square Propagation) optimizer with learning rate of 0.004. The loss function is MSE (Mean Squared Error), and the metric is MAE (Mean Absolute Error). We tried other models with different layers, neuron count, and optimizers, and this was the best performing model.

Training & Validation

To train the model, we used the K-fold cross validation method with K = 4. More specifically, in the first fold the training data will be divided into 3 parts, 2 parts of training and 1 of validation, in the second fold 2 parts of validation and 1 part of training as shown in the figure below and so on.



K-Fold Validation

We trained the model using 300 epochs and batch size of 2 and kept a history of the training and validation MAE's and training and validation Loss. Both are shown below as graphs.

Training VS Validation MAE

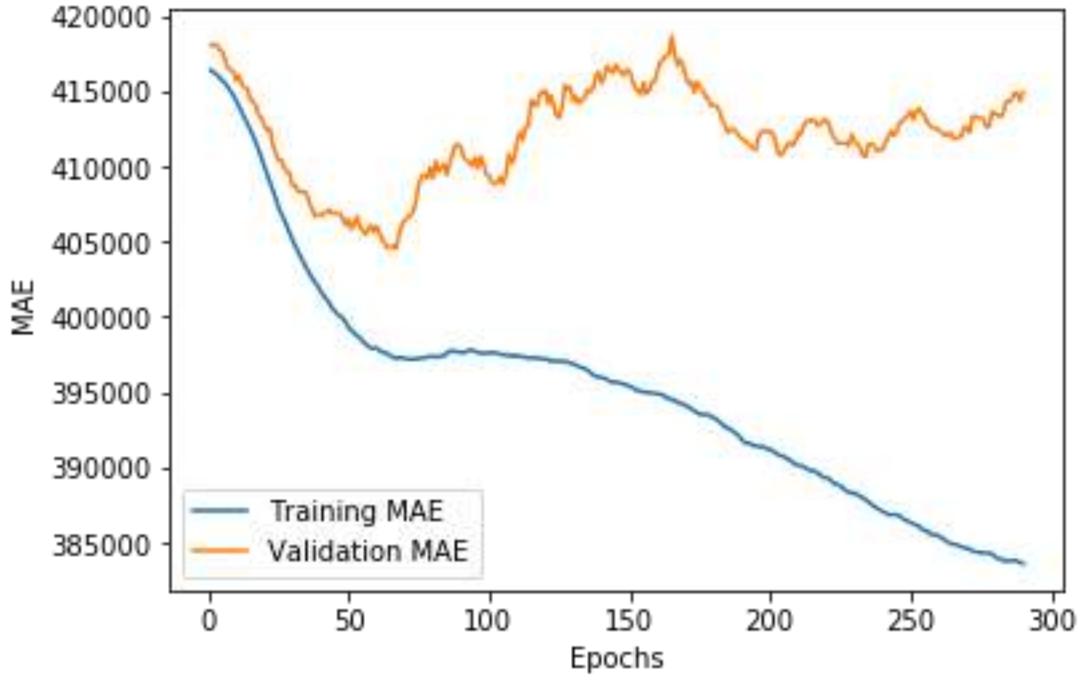


Figure 16

Training VS Validation Loss

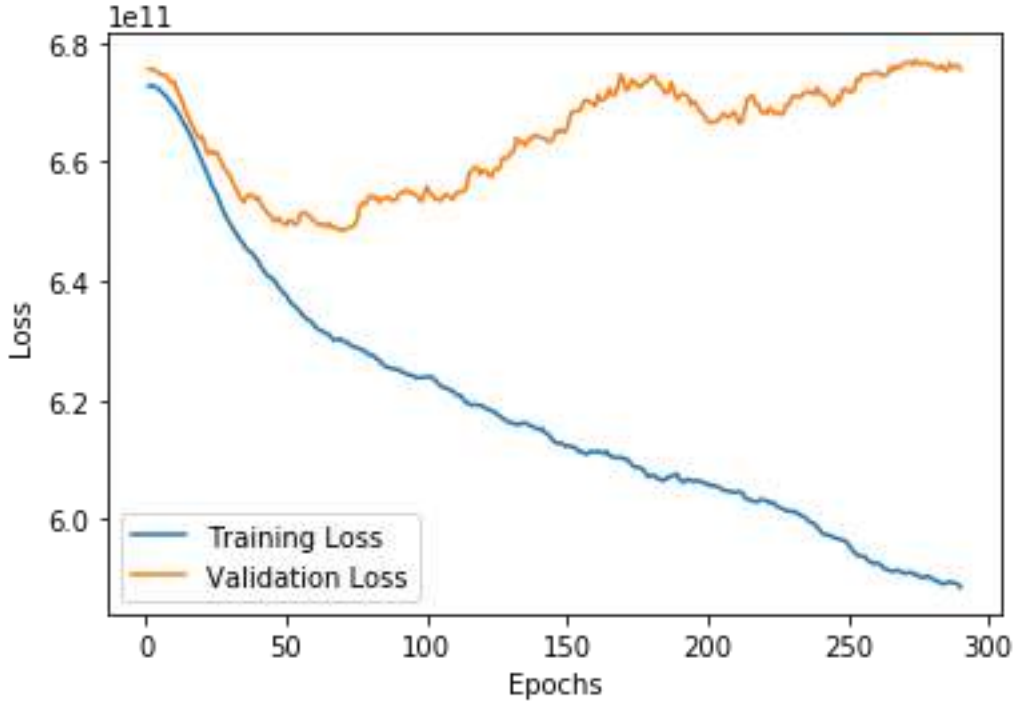


Figure 17

Results

After training and validating, we ran the new model using the test set, and the model was trained using the full training set with 75 epochs and batch size of 2 as it seems appropriate from validation MAE graph.

The test set MAE = 405201.75.

Conclusion

The performance of the model definitely improved after removing all the outliers. Figure 1 shows the performance of the model before removing the outliers; the validation MAE plateaued at around 555981.2 after 500 epochs. Figure 16 shows the performance of the model after removing the outliers; the validation MAE plateaued at around 405,201 after 75 epochs.

For future improvements, we should handle preprocessing of data in more detailed and professional manner. Also, Construction of model according to the data would help as well.