**House Price Prediction Machine Learning Model**

**Sahito, Saad Nisar**

**Wei-Chia, Hsu**

**Cheng-Yu, Lin**

**Introduction**

Our project involves regression analysis to predict the cost of houses based on different attributes, including but not limited to the city, location, and type of house.

To start, we load and analyse the dataset and focus on identifying the most significant features for prediction.

Afterward, we utilize the important features to make predictions and enhance the accuracy of our model by selecting the most crucial features.

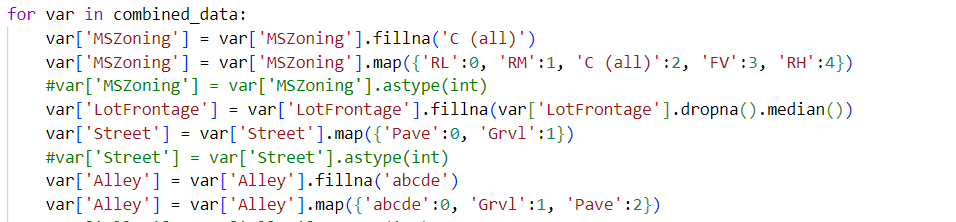
**Methodology**

After sorting the dataset and loading the train and test datasets, different machine learning algorithms were formed for testing. The three major models that were used were Linear Regression, Random Forest Regressor, and Neural Networks. The models were tested with different parameters that fit the dataset as accurately as possible. The accuracy of the model was tested through plots of generalization error by comparing the MSE for the training dataset and the testing dataset, for validation. The closer the two MSE were after training the better the model is when it comes to reliability. A scatter plot was also plotted for the predicted sale price and actual sale price for a visual comparison of the models.

A plot of the most significant classes was also made after the initial iteration of the models which helped to hand pick the top classes to be used for the next iteration.

**Preprocessing the Dataset**

The dataset for this project was taken from Kaggle.com, the dataset was very vast with more than 80 classes that define the sale price of a particular house. So in order to make sense of all the classes and feed them into a machine learning model, the classes sub-classes were labelled with numbers starting from 0. This is so that the predictions could be based on a number that corresponds to a particular sub-class of a classes. The image below shows the preprocessing of the data for a few classes od the dataset.



Chart

Description automatically generated with low confidenceSome data in the dataset was missing so a mean value with respect to nearby data points was taken to fill in that missing data. After an initial iteration of the model, we plotted an importance factor graph for the classes of the dataset.

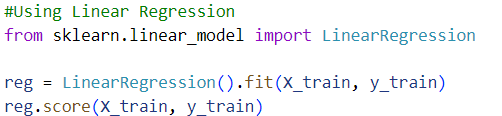
**Model Selection**

**Linear Regression**

Linear regression is a statistical method used to establish a relationship between a dependent variable and one or more independent variables. It is a linear approach to modelling the relationship between a dependent variable and one or more independent variables.

The linear regression model assumes that there is a linear relationship between the dependent variable and the independent variables. It is a simple model that uses a straight line to represent the relationship between the variables.

The linear regression model was simple to implement with no real parameters to tweak.



The generalization error was very high for this.

Chart, line chart

Description automatically generated

**Random Forest Regressor**

Random Forest Regressor is a supervised machine learning algorithm that belongs to the family of ensemble methods. It is a powerful algorithm used for regression analysis and is based on decision trees.

In Random Forest Regressor, many decision trees are created, and their results are combined to make a final prediction. Each decision tree in the forest is built independently, using a random sample of the data and a random subset of the features. This randomness helps to reduce overfitting and improve the accuracy of the predictions.

The algorithm works by creating decision trees using random subsets of the available features and data, and then averaging their results to produce a final prediction. Each decision tree in the forest is built using a different subset of the data and features, so the trees are diverse and independent.

When a new data point is presented to the model, each decision tree in the forest makes a prediction, and the final prediction is obtained by averaging the predictions of all the trees.

A hyper-parameter grid search was done to find the best parameters for the random forest regressor as shown below.

Text

Description automatically generated

In the end the following parameters were chosen for the model.

{'max\_depth': 20, 'min\_samples\_split': 2, 'n\_estimators': 300}

However, the generalization error for this model also turned out to be very poor and unusable.

**Neural Network Model**

A neural network model, also known as an artificial neural network (ANN), is a computational model inspired by the structure and function of the human brain. It is composed of layers of interconnected nodes, or neurons, that process and transmit information.

A neural network model typically consists of three types of layers: input layer, hidden layer, and output layer. The input layer receives input data, which is then processed by the hidden layers, and the output layer produces the final output.

Each neuron in a neural network model performs a simple computation and is connected to other neurons through weighted connections. The weights of these connections are adjusted during the training process to optimize the performance of the model.

Neural network models can learn complex patterns and relationships in data and can be used for a wide range of tasks, including classification, regression, and image and speech recognition.

The neural network was initially tested with a single hidden layer as follow.

model = Sequential()

model.add(Dense(128, activation='relu', input\_shape=(X\_train.shape[1],)))

model.add(Dense(64, activation='relu'))

Chart

Description automatically generatedmodel.add(Dense(1, activation='linear'))

Considering the improved generalization error of the model compared to the other models, we decided to go along with this model.

Chart, scatter chart

Description automatically generatedA scatter plot was made for the predicted vs actual sale price for this model, though generalization error was low, the scatter plot portrayed a different perspective of the accuracy related to this model.

The line of best fit here should be as close to the middle of the plot as possible, meaning that it should have a gradient of exactly one. The above plot was not quite the result we had hoped for as the gradient was way off the mark, for an actual price of $300,000 the predicted price was closer to $200,000. So, we improved the model in the next step by using the most significant classes of the dataset and adding 4 more hidden layers, along with a L2 regularizer.

model = Sequential()

model.add(Dense(32, activation='relu',kernel\_regularizer=regularizers.l2(0.001)))

model.add(Dense(64, activation='relu',kernel\_regularizer=regularizers.l2(0.001)))

model.add(Dense(128, activation='relu',  input\_shape=(X\_train.shape[1],)))

model.add(Dense(64, activation='relu',kernel\_regularizer=regularizers.l2(0.001)))

model.add(Dense(32, activation='relu',kernel\_regularizer=regularizers.l2(0.001)))

model.add(Dense(1, activation='linear'))

The output of the scatter plot improved significantly after this.



The gradient of the slope improved and was closer to ideal value. For a house priced at $300,000 the predicted sale price was around $240,000. The hidden layers had improved the accuracy of the model greatly.

To try to improve it further, two more hidden layers were added like so.

model.add(Dense(32, activation='relu',kernel\_regularizer=regularizers.l2(0.001)))

model.add(Dense(64, activation='relu',kernel\_regularizer=regularizers.l2(0.001)))

model.add(Dense(64, activation='relu',kernel\_regularizer=regularizers.l2(0.001)))

model.add(Dense(64, activation='relu',  input\_shape=(X\_train.shape[1],)))

model.add(Dense(64, activation='relu',kernel\_regularizer=regularizers.l2(0.001)))

model.add(Dense(64, activation='relu',kernel\_regularizer=regularizers.l2(0.001)))

model.add(Dense(32, activation='relu',kernel\_regularizer=regularizers.l2(0.001)))

This further improved the accuracy to the $300,000 price for $260,000.

Chart, scatter chart

Description automatically generated

However, the model did not improve further, so we concluded the current model to be the best one for predicting house sale prices.

**Conclusion**

The best model was declared to be the 7-hidden layer neural network model with ‘relu’ activation function along with a L2 regularizer. The house sale price was predicted closer to its actual model using this model, with MSE score as shown.

Mean Squared Error: 1724162048.0

This model had the best generalization error compared to other types of models, with training and validation errors intersecting at just 10 epochs. The top features were selected as being most significant ones and used for the prediction model.