Determining the Best Model among Candidate Machine Learning Models for Chicago Suburb House Price Data

Rui Zhang

Illinois Institute of Technology,Chicago,60616,USA

Abstract. With the development of the society, there is an increasing need of predicting the house price in order to purchase properties economically. Myriads of machine learning methods have been developed to achieve this goal. In this paper, we narrow our focus on three types of machine learning methods-Linear Regression, Polynomial Regression, and Neural Network (Multilayer Perceptron Regressor) and train all of them on Chicago suburb house price dataset and employ all of them to make predictions. Performance is evaluated using mean squared error (MSE) and R square (R2) metrics. Preliminary experimental results demostrate that Neural Network outperforms the other candidate methods in all cases. Polynomial Regression of degree 2 outperforms Linear Regression in most cases but performs arbitrarily bad in worst cases. Therefore, Neural Network (MLP Regressor) is the best method for predicting Chicago suburb house price.

Keywords: model selection, Linear Regressioin, Polynomial Regression, Neural Network, house price.

1. Introduction

With the development of the society, more and more opulent people are focusing on purchasing their individual properties-especially town houses, apartments, condominiums. However, Much information on house price is still unavailable to most people because of various ulterior reasons. Therefore, being able to predict a house price before purchasing it renders a buyer to gain an advantage in a property transaction.

Myriads of scholars, technicians and data scientists have developed methods for predicting the house price. Most of their mehods have been proven to be effective. Robin A, Dubin predicts the house price by considering the correlation between the prices of neighboring houses [2]. J.J. Wang et al. employ a memristor-based ANN to predict the house prices of some Boston towns [3]. J Manasa et al. adopt various regression techniques such as linear regression, Lasso and Ridge regression models, support vector regression to predict the house price [4]. Archith J. Bency et al. predict the housing prices of London, Birmingham and Liverpool with the aid of satelite images [5]. Jorge Chica-Olmo et al. present a multi-equational hedonic regression model with coregionalized disturbances to predict the house price [6]. Nuri Hacıevliyagil et al. adopt the DMA method to predict the house price in Turkey [7].

In this paper, we leverage the Chicago suburb house price dataset from Kaggle. We use 5-fold cross-validation to split dataset into a training set and a testing set. We train the following models (Linear Regression, Polynomial Regression and Neural Network) on the training set and predict the house price on the test set. We evaluate the performance of the prediction using mean squared error (MSE), R square (R^2), average MSE and average R^2 metrics. Based on the evaluation, we conclude that Neural Network performs much better than polynomial regression of degree 2 and linear regression models in all cases.

2. Methodology

In this section, we introduce the dataset we use for training and testing, and methods for preprocessing the data. We present the 5-fold cross-validation and describe how to use it to split the data and explain the reason for using this method. Then we give a brief summary of the models we use for training and prediction because numerous textbooks have detailed them thoroughly. Last, we describe the performance metrics for evaluating those models and present the formulas for calulating them.

2.1 Dataset

The dataset we use for this project is Chicago house price dataset consisting of 156 rows of data and including price and 8 corresponding features. These features are bedroom, space, room, lot, tax, bathroom, garage, condition. Their explanations are as follows (See Table 1).

Table 1 Explanations of Column Items in the Chicago House Price Dataset[8]

Column Item	Explanation
Price	Price of house
Bedroom	Number of bedrooms
Room	Number of rooms
Space	Size of house (in square feet)
Lot	Width of a lot
Tax	Amount of annual tax
Bathroom	Number of bathrooms
Garage	Number of garages
Condition	Condition of house (1 if good , 0 otherwise)

This paper focus on how to predict the price based on the remaining 8 determinants.

2.2 Preprocessing Data

2.2.1 Handing Missing Value

There are about 30 missing values in this dataset, indicated by NA. We fill the missing value through interpolation. That is, we replace the missing value with the mean of the adjacent values in the same column. There is no exception of this rule since there are no missing values in the first and the last rows of data.

2.2.2 Feature Scaling

Observing that the values in Space, Room, Lot, Tax columns are far more greater than 1, we use formula (1) to normalize those features to achieve faster coverage in Gradient Descend.

$$x_{new} = \frac{\mathrm{x}-\mathrm{x}_{\mathrm{mean}}}{\mathrm{x}_{\mathrm{max}}-\mathrm{x}_{\mathrm{min}}} \tag{1}$$

After normalizing, all values in above columns falls into [-1,1].

2.3 5-fold Cross-validation

Cross-validation is a resampling procedure used to evaluate machine learning models on a limited data sample. 5 refers to the number of groups that the data is splitted into [9]. We use this procedure to split the data because we have a limited number of data and we want to how well the model generally works on this limited amount of data and we do not want the performace that depends on the choice of splitting the data. The entire procedure is as follows:

Shuffle the dataset randomly.

Split the dataset into 5 groups

For each unique group:

Take the group as a hold out or test dataset.
Take the remaining groups as a training dataset.
Fit a model on the training set and evaluate it on the test set.
Retain the evaluation score and discard the model[9].
Calculate the average score based on above evaulation scores.

2.4 Three Models

2.4.1 Linear Regression

Linear regression is the regression in which the target value is a linear combination of the features while it tries to minimize the squared error function.

2.4.2 Polynomial Regression

Similar to Linear regression, polynomial regression also tries to minimize the squared error function but the hypothesis is a polynomial.

2.4.3 Neutral Network

We use Multilayer Perceptron Regressor (MLP Regressor) for training and predition. Multi-layer Perceptron (MLP) is a supervised learning algorithm that learns a function f(*):R^m->R^o by training on a dataset, where m is the number of dimensions for input and o is the number of dimensions for output. MLP Regressor implements a multi-layer perceptron (MLP) that trains using backpropagation with no activation function in the output layer, which can also be seen as using the identity function as activation function. Therefore, it uses the square error as the loss function, and the output is a set of continuous values [1].

2.5 Performance Metrics

2.5.1 Mean Squared Error (MSE)

Mean Square Error is an absolute measure of the goodness for the fit. MSE is calculated by the sum of square of prediction error which is real output minus predicted output and then divide by the number of data points (See formula (2)). It gives you an absolute number on how much your predicted results deviate from the actual number [10]. The smaller the value is, the better the model is.

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{2}$$

2.5.2 R Square*(R^2)*

R square measures how much variability in dependent variable can be explained by the model. R square is calculated by the sum of squared of prediction error divided by the total sum of the square which replaces the calculated prediction with mean (See formula (3)). R square is a good measure to determine how well the model fits the dependent variables [10]. The value is at most 1, the closer the value is to 1, the better the model fits the data.

$$R^2 = 1 - \frac{SS_{Regression}}{SS_{Total}} = 1 - \frac{\sum_i(y_i - \hat{y}_i)^2}{\sum_i(y_i - \bar{y})^2} \tag{3}$$

Average MSE

Average MSE equals sum of MSEs divided by 5, where 5 is the number of training-testing processes in 5-fold cross-validation.

Average R^2

Average R^2 equals sum of R^2s divided by 5, where 5 is the number of training-testing processes in 5-fold cross-validation.

3. Experiments

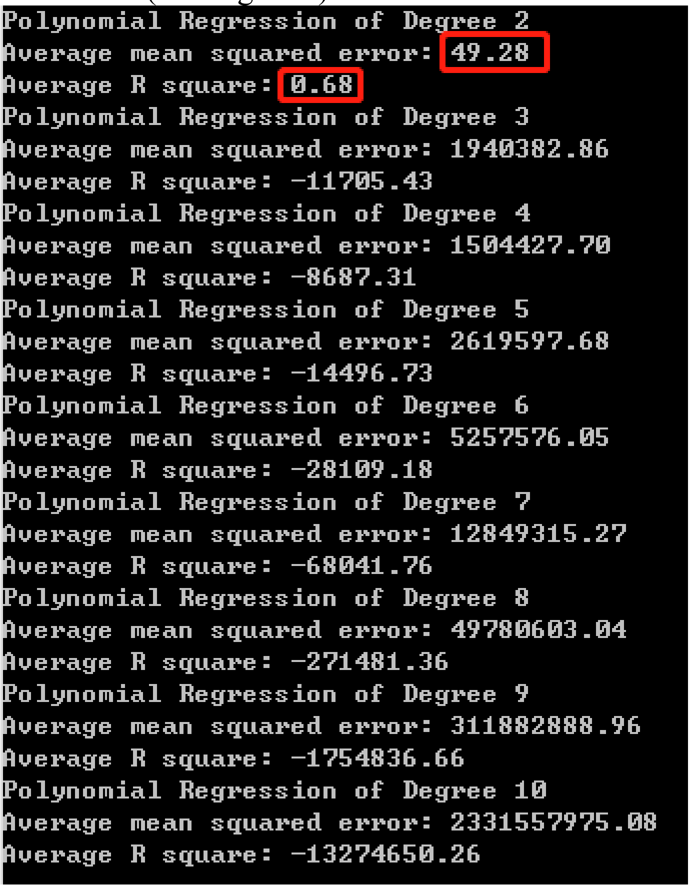
In this section, we first briefly describe the program. Then we end this section by describing the experiments and corresponding results. We draw immediate conclusions based on the experimental results.

3.1 The Program

The program is implemented in Python using scikit learn package. The program consists of 2 files: *start.py* and *tool.py*. *start.py* includes the main function that defines the experiments we conduct. *tool.py* includes the functions that preprocess the data, and that split the data using 5-fold cross-validation, and that train and test the Linear Regression, Polynomial Regression and Neural Network models, and that evaluate the prediction performance of these models. The total file size is 8KB and relatively small, which does not take too much space in the computer.

3.2 Experiment 1:Determing the Best Polynomial Regression Model

For simplicity, we restrict the degree of a polynomial to an integer. We set the degree to be values from 2 to 10 and then execute the program to train the Polynomial Regression models. The prediction performance results are as follows (See Figure 1):



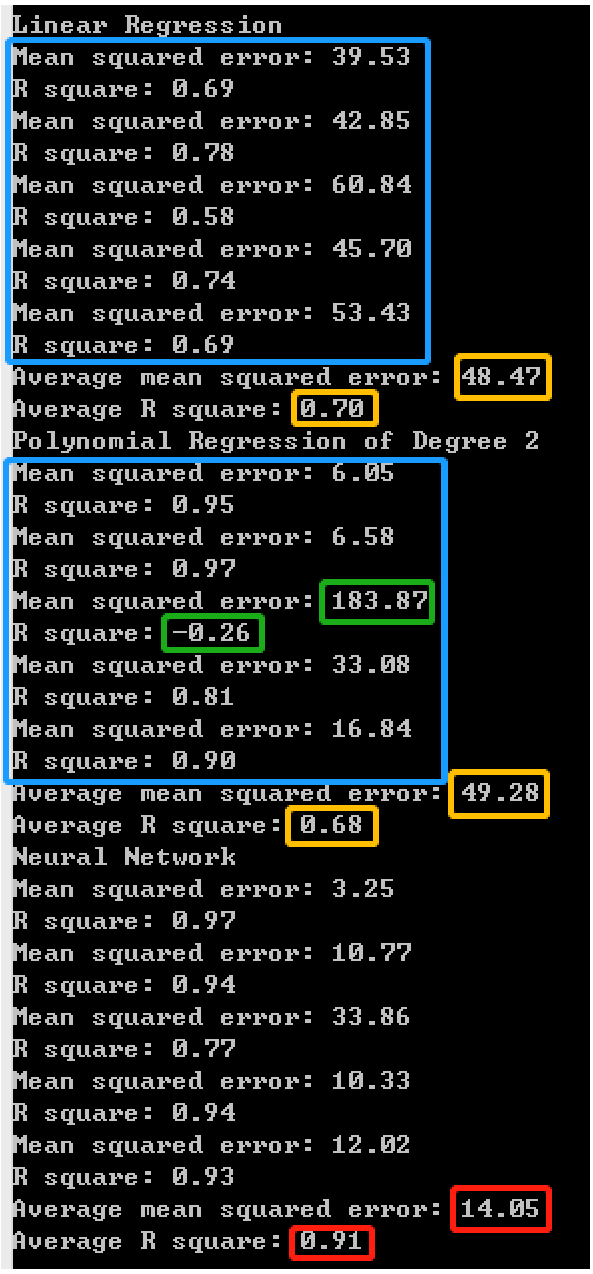
```
Polynomial Regression of Degree 2
Average mean squared error: 49.28
Average R square: 0.68
Polynomial Regression of Degree 3
Average mean squared error: 1940382.86
Average R square: -11705.43
Polynomial Regression of Degree 4
Average mean squared error: 1504427.70
Average R square: -8687.31
Polynomial Regression of Degree 5
Average mean squared error: 2619597.68
Average R square: -14496.73
Polynomial Regression of Degree 6
Average mean squared error: 5257576.05
Average R square: -28109.18
Polynomial Regression of Degree 7
Average mean squared error: 12849315.27
Average R square: -68041.76
Polynomial Regression of Degree 8
Average mean squared error: 49780603.04
Average R square: -271481.36
Polynomial Regression of Degree 9
Average mean squared error: 311882888.96
Average R square: -1754836.66
Polynomial Regression of Degree 10
Average mean squared error: 2331557975.08
Average R square: -13274650.26
```

Figure 1 Results of Experiment 1

As shown in Figure 1, among all 9 models, the Polynomial Regression of Degree 2 has the smallest average MSE (49.28) and the largest average R^2 (0.68) (See the figures in red boxes). In addition, as the degree increases, the average MSE increases and the average R^2 decreases. It indicates that as the complexity of the model increases, the model becomes worse. Therefore, we can conclude that the Polynomial Regression of Degree 2 is the best Polynomial Regression Model.

3.3 Experiment 2: Determining the Best Model among 3 Candidate Models

In this experiment, we execute the program to train Linear Regression model, Polynomial Regression model of degree 2, and Neural Network Model. The prediction performance results are as follows (See Figure 2):



```
Linear Regression
Mean squared error: 39.53
R square: 0.69
Mean squared error: 42.85
R square: 0.78
Mean squared error: 60.84
R square: 0.58
Mean squared error: 45.70
R square: 0.74
Mean squared error: 53.43
R square: 0.69
Average mean squared error: 48.47
Average R square: 0.70
Polynomial Regression of Degree 2
Mean squared error: 6.05
R square: 0.95
Mean squared error: 6.58
R square: 0.97
Mean squared error: 183.87
R square: -0.26
Mean squared error: 33.08
R square: 0.81
Mean squared error: 16.84
R square: 0.90
Average mean squared error: 49.28
Average R square: 0.68
Neural Network
Mean squared error: 3.25
R square: 0.97
Mean squared error: 10.77
R square: 0.94
Mean squared error: 33.86
R square: 0.77
Mean squared error: 10.33
R square: 0.94
Mean squared error: 12.02
R square: 0.93
Average mean squared error: 14.05
Average R square: 0.91
```

Figure 2 Results of Experiment 2

As shown in Figure 2, Neural Network has the smallest average MSE (14.05) and the largest average R^2 (0.91) (See the figures in red boxes), compared to the other 2 models. Therefore, Neural Network is the best model among 3 candidate models. Also observing that althrough Polynomial Regression of degree 2 and Linear Regression have similar average MSEs and averge R^2s (Compare the corresponding figures of these two models in yellow boxes), the former outperforms the latter in most iterations according to MSE and R^2 for each iteration (Compare the figures of these two models in blue boxes). However, in the third iteration, Polynomial Regression of degree 2 has MSE (183.87) and R^2 (-0.26) (See the figures in green boxes), which greatly reduce its overall performance. It indicates that in this worst case, Polynomial Regression of degree 2 fails to make accurate prediction on testing data.

4. Conclusion

In this paper, we have trained the candidate machine learning models and have evaluated the prediction performance of all these models on Chicago suburb house price dataset. After two experiments, we conclude that among the candidate models, the prediction performance of Neural Network is the best. Therefore, we select the Neural Network model for predicting the Chicago suburb house price. We also conclude that Polynomial Regression model of degree 2 is the best Polynomial Regression model and it outperforms Linear Regression model in most cases, but its prediction performance is very poor in the worst cases.

There are still myriads of other methods for predicting house price as they are mentioned in section 1. In addtion, this paper does not tackle time-based dataset. However, some house price datasets are related to time. If time is considered, the methods for processing the data in this paper may not work. New methods should be applied to this type of data.

References

[1] Buitinck et al. (2013). Neural network models (supervised). [Online]. Available:https://scikitlearn.org/stable/modules/neural_networks_supervised.html

[2] R. A. Dubin, “Predicting House Prices Using Multiple Listings Data,” The Journal of Real Estate Finance and Economics, vol. 17, no. 1, pp. 35–59, 1998, doi: 10.1023/A:1007751112669.

[3] J. J. Wang et al., “Predicting House Price With a Memristor-Based Artificial Neural Network,” IEEE Access, vol. 6, pp. 16523–16528, 2018, doi: 10.1109/ACCESS.2018.2814065.

[4] J. Manasa, R. Gupta, and N. S. Narahari, “Machine Learning based Predicting House Prices using Regression Techniques,” in 2020 2nd International Conference on Innovative Mechanisms for Industry Applications (ICIMIA), Bangalore, India: IEEE, Mar. 2020, pp. 624–630. doi: 10.1109/ICIMIA48430.2020.9074952.

[5] A. J. Bency, S. Rallapalli, R. K. Ganti, M. Srivatsa, and B. S. Manjunath, “Beyond Spatial Auto-Regressive Models: Predicting Housing Prices with Satellite Imagery,” in 2017 IEEE Winter Conference on Applications of Computer Vision (WACV), Santa Rosa, CA, USA: IEEE, Mar. 2017, pp. 320–329. doi: 10.1109/WACV.2017.42.

[6] J. Chica-Olmo, R. Cano-Guervos, and M. Chica-Olmo, “A Coregionalized Model to Predict Housing Prices,” Urban Geography, vol. 34, no. 3, pp. 395–412, May 2013, doi: 10.1080/02723638.2013.778662.

[7] N. Hacıevliyagil, K. Drachal, and I. H. Eksi, “Predicting House Prices Using DMA Method: Evidence from Turkey,” Economies, vol. 10, no. 3, p. 64, Mar. 2022, doi: 10.3390/economies10030064.

[8] Tawfik Elmetwally, (2023). Chicago House Price. [Online]. Available: https://www.kaggle.com/datasets/tawfikelmetwally/chicago-house-price?resource=download

[9] Jason Brownlee, (2018). A Gentle Introduction to K-fold Cross-Validation. [Online]. Available: https://machinelearningmastery.com/k-fold-cross-validation/

[10] Songhao Wu. (2020) 3 Best Metrics to Evaluate Regression Model? [Online]. Available: https://towardsdatascience.com/what-are-the-best-metrics-to-evaluate-your-regression-model-418ca481755b