Proceedings of the 3rd International Conference on Signal Processing and Machine Learning
DOI: 10.54254/2755-2721/6/20230928

Optimization of linear regression in house price prediction

Liangji Zhu

University of Science and Technology of China, Anhui, China

ustc_zlj@mail.ustc.edu.cn

Abstract. House price prediction plays a very important role in housing transactions. Linear regression based algorithms show good effects in predicting house prices. They have strong interpretability and fast operation speed. However, people ignore the estimation of deviations in linear regression (LR) algorithms. In this paper, k-nearest neighbor (KNN) algorithm is supposed to estimate deviations that are added to the result of linear regression to predict house prices accurately. Furthermore, deviation regression (DR) algorithm is supposed to make the prediction result more accurate. By utilizing Boston House Price data from Kaggle, extensive experiments are conducted and demonstrate the superior performance and compatibility of DR.

Keywords: Linear Regression, Deviation, Penalty Coefficient.

1. Introduction

In a real estate transaction, the pricing of the house is very important for both sellers and buyers. An accurate valuation algorithm can help real estate developers get the best selling price for each house, so that they can be more competitive in the fierce real estate competition market and gain more benefits. At the same time, it can also help buyers to judge the value of houses and avoid losses. Therefore, the prediction of the house price is necessary.

The linear regression algorithm is suitable for house price prediction [1-3]. This algorithm can make a reasonable estimate of the house price. However, it can be found that when using the linear regression algorithm, no matter how the learning rate is adjusted, the loss function will gradually converge to a big value [4,5]. This shows that there is an upper limit to the accuracy of the predicted value obtained by the linear regression algorithm. Therefore, in this paper, the author supposes to improve the linear regression algorithm to make the predicted value more accurate.

First of all, by training the linear regression model, the author gets a linear regression equation, which can be used to predict the house price. At this time, the author gets inaccurate prediction results. These predicted values have some deviations from the actual values. For the data in the training set, these deviations can be calculated. After that, KNN algorithm [6] is used to predict the deviation for a piece of house data through the deviations of data in the training set. In this way, deviations can be multiplied by the parameter β as a new feature and added to the original new regression equation. Eventually, a more accurate prediction model is obtained by training this new linear regression equation.

In this work, the author makes the following contributions: first, the KNN algorithm is used to predict the deviation of each house data; second, the linear regression algorithm is combined with the KNN algorithm to get a more accurate house price prediction algorithm; third, DR is proposed to

DOI: 10.54254/2755-2721/6/20230928

predict the house price more accurately; at last, the Boston house price data set is tested and it is verified that the new algorithm increases the prediction accuracy.

2. Related work

2.1. Multiple linear regression model

If the value to be predicted has a linear relationship with $X^1, X^2, \cdots, X^n$, they satisfy the linear regression model and it can be formulated as:

$$h_\theta(X) = \theta_0 + \theta_1 X_1 + \theta_2 X_2 + \cdots + \theta_n X_n \tag{1}$$

In order to express the accuracy of linear regression results, the cost function is:

$$J(\theta_0, \theta_1, \cdots, \theta_n) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(X^{(i)}) - y^{(i)})^2 \tag{2}$$

When the cost function reaches the minimum value, it shows that the linear regression has the best prediction effect.

2.2. Gradient descent

Gradient descent algorithm [7,8] can make $h_\theta(X)$ gradually approach to the minimum value through iteration. Simultaneously update all theta values for every iteration:

$$\theta_j = \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1, \cdots, \theta_n) \tag{3}$$

$$\frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1, \cdots, \theta_n) = \frac{1}{m} \sum_{i=1}^{m} (h_\theta(X^{(i)}) - y^{(i)}) X_j^{(i)} \tag{4}$$

α stands for the learning rate, which is obtained through debugging. Update theta and recalculate $J(\theta_0, \theta_1, \cdots, \theta_n)$. Until $J(\theta_0, \theta_1, \cdots, \theta_n)$ reaches the minimum or the number of iterations reaches the upper limit, the theta value corresponding to the model can be obtained.

2.3. Feature scaling

In order to reduce the number of iterations and speed up the operation, feature scaling is used to scale all the values of X_i to the range of zero to one:

$$X_i = \frac{X_i - \min(X_i)}{\max(X_i) - \min(X_i)} \tag{5}$$

2.4. KNN algorithm

KNN is a Supervised Learning algorithm that classifies new data points into the target class, depending on the features of their neighbouring data points. K needs to be tested to get the optimal value.

DOI: 10.54254/2755-2721/6/20230928

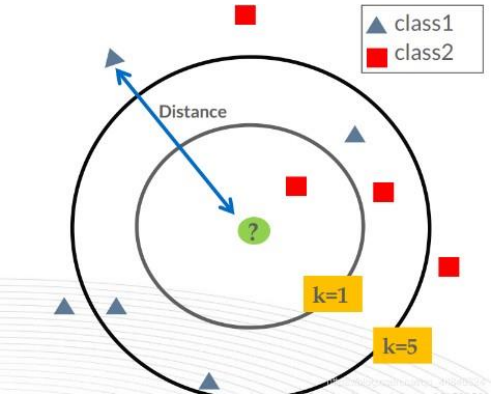

Figure 1. An example of KNN. When k is 1, the nearest point to X belongs to class 2, so X should be classified as class 2. When k is 5, since most of the five points closest to X belong to class 1, X should be classified as class 1 [9].

3. Methodology

3.1. Problem formulation

In the Boston house price data set, each data contains 13 features. Therefore, the price of a house is expected to be predicted by those 13 features. The linear regression model is as follows:

$$h_\theta(X) = \theta_0 + \theta_1 X_1 + \theta_2 X_2 + \cdots + \theta_{13} X_{13} \tag{6}$$

Data in the training set are used to train this linear regression model to get the value of the parameters $\theta_0, \theta_1, \cdots, \theta_{13}$.

3.2. LR algorithm

The linear regression algorithm updates the parameter $\theta_0, \theta_1, \cdots, \theta_{13}$ by using gradient descent until the cost function reaches the minimum value. The minimum loss function means that the linear regression model has the best fitting effect.

Algorithm 1 LR algorithm

Input: input training set

Output: output $\theta_0, \theta_1, \cdots, \theta_{13}$

1: Feature scaling
2: Initialize θ and α
3: Calculate cost function
4: **for** cost function is not minimum or iterations does not reach the upper limit **do**
5: Gradient descent update θ
6: Calculate cost function
7: **return** $\theta_0, \theta_1, \cdots, \theta_{13}$

3.3. LR-KNN algorithm

After executing LR, deviations are predicted by KNN; deviations of the house data in the training set are calculated; m_i is supposed to represent the deviation of the house X_i:

$$m_i = y_i - h_\theta(X_i) \tag{7}$$

DOI: 10.54254/2755-2721/6/20230928

Here, the KNN algorithm is used to predict deviations for data in the test set. For a house in the test set, K houses that are most similar to it need to be found in the training set. This similarity should be determined by the value of features because similar houses should have similar features. Suppose there are n features and the distance function is as follows:

$$D(X_i, X_j) = \sum_{k=1}^{n} (X_i^{(k)} - X_j^{(k)})^2 \tag{8}$$

When the features of two houses are similar, the value of the corresponding distance function will be smaller. Therefore, for each house (X_i) and each house (X_j) in the training set, sort the values of $D(X_i, X_j)$ and find K minimum values. From this, K houses that are most similar to the target house X_i are selected. Store the serial numbers of these K houses in set S_i. Then the estimated deviation of X_i can be calculated:

$$\hat{m}_i = \frac{1}{K} \sum_{t=1}^{K} m_j, j \in S_i \tag{9}$$

Suppose the new prediction value for X_i is Y_i, the algorithm is as follows:

Algorithm 2 LR-KNN algorithm

Input: input training set & test set
Output: output prediction value of house price
1: Feature scaling
2: Initialize θ and α
3: Calculate cost function
4: **for** cost function is not minimum or iterations does not reach the upper limit **do**
5: Gradient descent update θ
6: Calculate cost function
7: **for** each X_i in the training set **do**
8: Calculate m_i
9: **for** each X_i in the test set **do**
10: **for** each X_j in the training set **do**
11: Calculate $D(X_i, X_j)$
12: Sort $D(X_i, X_j)$
13: Store j corresponding to the K smallest $D(X_i, X_j)$ in set S_i
14: $\hat{m}_i = \frac{1}{K} \sum_{t=1}^{K} m_j, j \in S_i$
15: $\hat{Y}_i = h_\theta(X_i) + \hat{m}_i$
16: return Results

3.4. DR algorithm

In the LR-KNN algorithm, the deviation is directly added to the result of the LR algorithm to generate a new predicted value. At this point, deviation can be used as a new feature of the data set, and linear regression can be carried out again to get the influence of deviation on the prediction results. On the basis of the LR-KNN algorithm, add a penalty coefficient β for m_i, and the new linear regression equation and new cost function become as follows:

DOI: 10.54254/2755-2721/6/20230928

$$h'_\theta(X_i) = \theta_0 + \theta_1 X_1 + \theta_2 X_2 + \cdots + \theta_{13} X_{13} + \beta \hat{m}_i \tag{10}$$

$$J(\theta_0, \theta_1, \cdots, \theta_{13}, \beta) = \frac{1}{2m} \sum_{i=1}^{m} (h'_\theta(X^{(i)}) - y^{(i)})^2 \tag{11}$$

Linear regression is used again and the new linear regression equation $h'_\theta(X_i)$ is trained. The algorithm is as follows:

Algorithm 3 DR algorithm

Input: input training set & test set

Output: output $\theta_0, \theta_1, \cdots, \theta_{13}, \beta$

1: Feature scaling
2: Initialize θ and α
3: Calculate cost function
4: **for** cost function is not minimum or iterations does not reach the upper limit **do**
5: Gradient descent update θ
6: Calculate cost function
7: **for** each X_i in the training set **do**
8: Calculate m_i
9: **for** each X_i in the train set & test set **do**
10:**for** each X_j in the training set **do**
11: Calculate $D(X_i, X_j)$
12: Sort $D(X_i, X_j)$
13: Store j corresponding to the K smallest $D(X_i, X_j)$ in set S_i
14: $\hat{m}_i = \frac{1}{K} \sum_{t=1}^{K} m_j,\ j \in S_i$
15: Feature scaling
16: Initialize θ & α & β
17: Calculate cost function
18: **for** cost function is not minimum or iterations does not reach the upper limit **do**
19: Gradient descent update θ & β
20: Calculate cost function
21: **return** $\theta_0, \theta_1, \cdots, \theta_{13}, \beta$

4. Experiments

4.1. Data set

The Boston house price data set from Kaggle is used to test the accuracy of 3 algorithms. Each house data has 13 features and their real prices:

Table 1. Features of Boston house price data set [10].

No.	features	Definition
1	CRIM	per capita crime rate by town
2	ZN	proportion of residential land zoned for lots over 25,000 sq.ft.
3	INDUS	proportion of non-retail business acres per town
4	CHAS	Charles River dummy variable

DOI: 10.54254/2755-2721/6/20230928

5	NOX	Charles River dummy variable
6	RM	average number of rooms per dwelling

Table 1. (continue)

7	AGE	proportion of owner-occupied units built prior to 1940
8	DIS	weighted distances to five Boston employment centers
9	RAD	index of accessibility to radial highways
10	TAX	full-value property-tax rate per $10, 000
11	PTRATIO	pupil-teacher ratio by town
12	B	1000(Bk − 0.63)2 where Bk is the proportion of blacks by town
13	LSTAT	13% lower status of the population
14	MEDV	Median value of owner-occupied homes in $1000s

There are 500 pieces of data in the data set. 450 pieces of data are used as the training set and the remaining 50 pieces of data are used as the test set.

4.2. Evaluation function

Use MAE to test the accuracy of prediction results.

$$\mathrm{MAE} = \frac{1}{50}\sum_{i=1}^{50}|h_\theta(X_i - y_i)|i\epsilon\text{test set} \tag{12}$$

The smaller the MAE, the more accurate the algorithm is in predicting house prices.

4.3. Parameter testing

In the LR algorithm and DR algorithm, α is tested in the order of [0.1,0.05,0.001,0.005...] until the result of the algorithm is the most accurate, which means MAE is the least. Because the LR-KNN algorithm is on the base of the LR algorithm, the value of α in the LR algorithm is the same as that in the LR-KNN algorithm. In addition, K in LR-KNN is tested in the order of [5,10,15,20...] until the least MAE is found.

4.4. Result

4.4.1. LR algorithm: MAE is 2.80, which is large. The results are reported in Figure 2(a). The predicted value of the house price is roughly the same as the actual value. But the predicted house prices are not accurate enough.

4.4.2. LR KNN algorithm: MAE is 2.38, which reduces 15% compared to the MAE of the LR algorithm. The results are reported in Figure 2(b). It makes the following observations: 1). House prices are predicted more accurately by the LR-KNN algorithm than the LR algorithm; 2). The estimation of deviation effectively increases the accuracy of the algorithm; 3). The KNN algorithm can successfully predict deviations.

4.4.3. DR algorithm: MAE is 2.16, which reduces 22.8% compared to the MAE of the LR algorithm and 9.2% compared to the MAE of the LR-KNN algorithm. The results are reported in Figure 2(c). It makes the following observations: 1). The DR algorithm has the best prediction effect whose predicted value is the most accurate; 2). It is necessary to introduce the penalty coefficient β, which effectively increases the accuracy of the algorithm in predicting house prices.

DOI: 10.54254/2755-2721/6/20230928

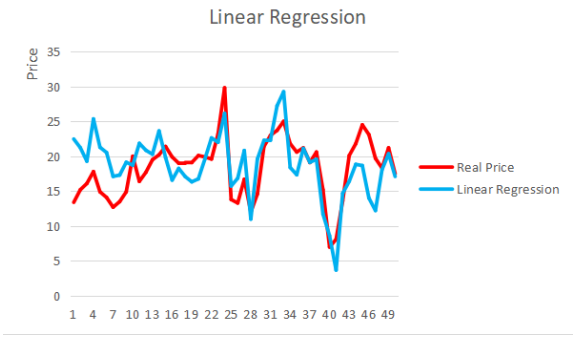

Figure 2. Predicted values of 3 algorithms.(a). Predicted value of LR

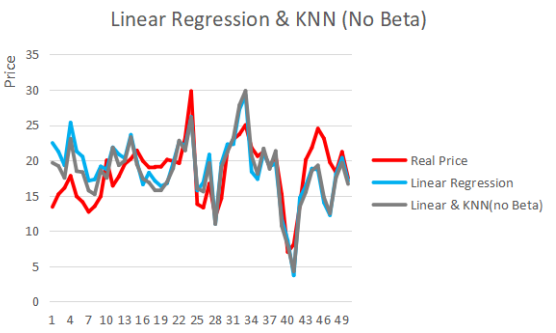

Figure 2. Predicted values of 3 algorithms.(b). Predicted value of LR & LR-KNN

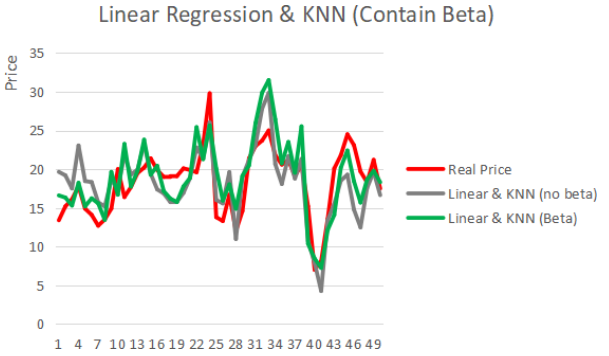

Figure 2. Predicted values of 3 algorithms.(c). Predicted value of LR-KNN & DR

5. Conclusion

In this paper, on the basis of linear regression, the DR algorithm is supposed to predict house prices more accurately by estimating deviations using the KNN algorithm. Furthermore, a penalty coefficient β is introduced in front of deviations. The author takes deviations as a new feature and uses the linear regression algorithm again to get the value of β. The author provides the feasibility of estimating the deviations by the similarity among houses. The formula $D(X_i, X_j)$ is defined to judge the similarity among houses. Additionally, the results of experiments indicate that estimating deviations and adding them to the linear regression results can improve the prediction accuracy. The accuracy is further increased after the introduction of the penalty coefficient β.

References

[1] Zhang, K., Zhu, X.: Analysis of Z City House Price Forecast based on Application of Multiple Linear Regression. The First International Symposium on Economics, Management, and Sustainable Development (2019).

[2] Tianjun, W.U., Fang, Y.: The City's House Price Evaluation Model Based on the Regional Characteristics. Journal of Geomatics Science and Technology (2012).

[3] Zhao, X.: Analysis of factors influencing real estate price in China based on linear regression. Journal of Anhui Jianzhu University (2018).

[4] Cui, Z., Chen, R., Hong, S., Liu, D., Liu, L., Tang, S., et al.: A method of predicting housing price using the method of combining multiple source data with mathematical model (2020).

[5] Yang, G., Deng, X.: Regression Analysis and After-Simulation of House Price in Chengdu City. Value Engineering (2007).

[6] Wilson, A., Sukumar, R. and Hemalatha, N.: Machine learning model for rice yield prediction using KNN regression (2021).

DOI: 10.54254/2755-2721/6/20230928

[7] Sun, Y., et al.: Application of gradient descent method in machine learning. Journal of Suzhou University of Science and Technology (Natural Science Edition) (2018).
[8] Schleich, M. Learning Linear Regression Models over Factorized Joins. International Conference on Management of Data ACM (2016).
[9] Back garden. This article makes it easy for you to understand KNN classification algorithm. https://blog.csdn.net/weixin 45192980/article/details/118534309.
[10] Boston housing dataset. https://www.kaggle.com/datasets/altavish/boston-housing-dataset.