Using Gradient Descent Method to Predict House Price

Peilin Liu, kirby school

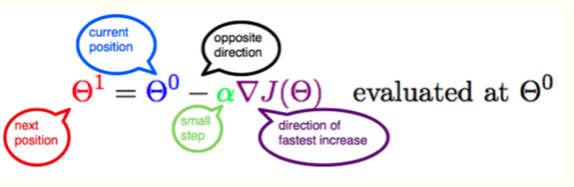
# **Abstract**

The purpose of the project is to study the gradient descent algorithm in machine learning, choose linear regression models, obtain the linear regression function through the gradient descent algorithm to predict the house price, use Python programming to implement the algorithm.

# **1.Principle of gradient and gradient descent algorithm**

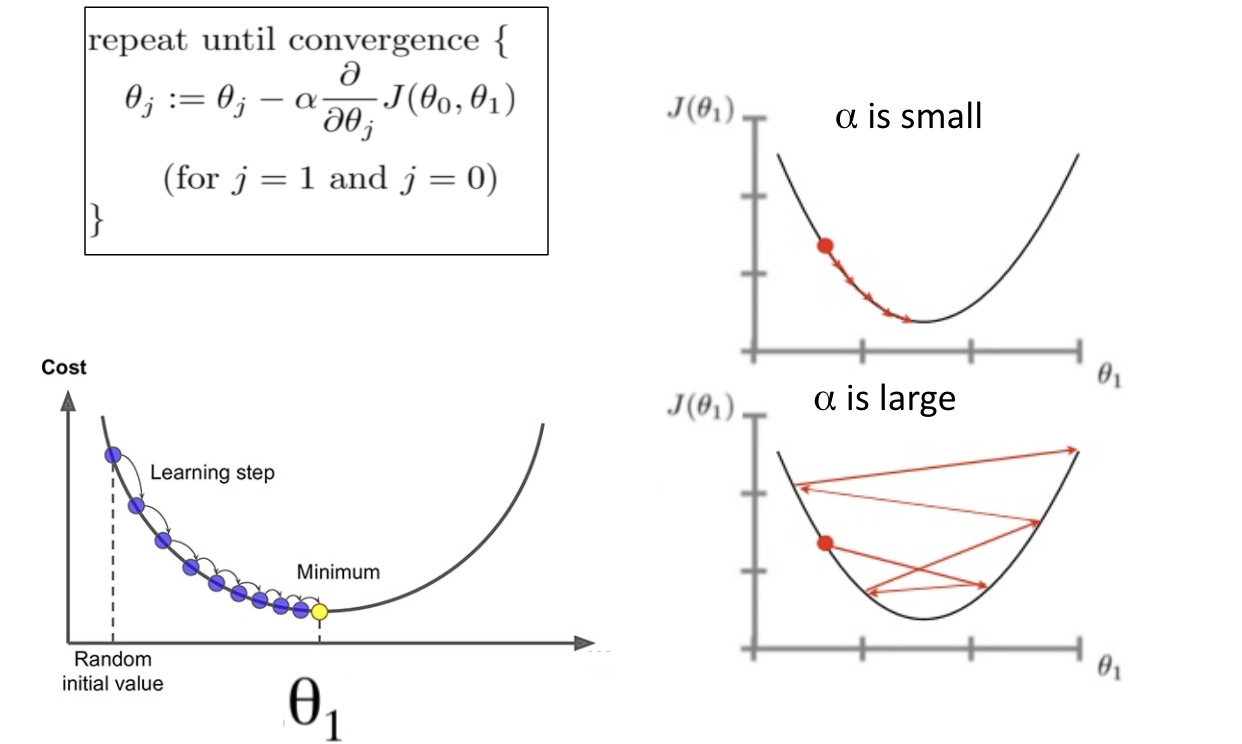
Gradient descent algorithm is widely used in machine learning, its principle is walking along the opposite direction of the function gradient to reach the local minimum of the function by taking repeated steps.

Gradient descent formula is ：



In the formula ,is the current value， is the value of next step， *J*() is function of,can be multiple variables , *∇J*( is the gradient of *J*( )，**-α*∇J*(**is the opposite direction of the gradient，from walks a distance of step length α to reach point.α is the gradient descent algorithm learning rate or step length, which means that we can through α control the distance of each step.α cannot be too large or too small. If it is too small, it may lead to delays in walking，if it is too large, it will cause the lowest point to be missed!

Following graph describes the gradient iterate algorithm in computer ：



**2.Using the gradient descent method to predict house price**

There are lots of factors can affect house price, including area, house age，incoming level，population density, etc. Project adopts linear regression model，use gradient descent algorithm to learn the coefficient of house price the prediction linear regression function. The learning data can be from Zillow or house company history reports.

For machine learning ,it needs to choose a loss function , in general, regression models choose  [***MSE（mean square error)***](https://link.jianshu.com/?t=https%253A%252F%252Fen.wikipedia.org%252Fwiki%252FLeast_squares)as loss function. Set y\_pred is the prediction function，y is the true value of a house，then ***loss function =*** ***MSE(y\_pred, y) =*** , m is the number of learning data samples. For MSE, the smaller is the better.

### **2.1 single variable linear regression**

Assume house price and the area is a linear relationship ，then we can use single variable linear regression function ***f (x) = w\*x + b*** to predict the price of a house， where f denotes the house price, x represents the house area, so ***Loss(w, b)=***  ,where:

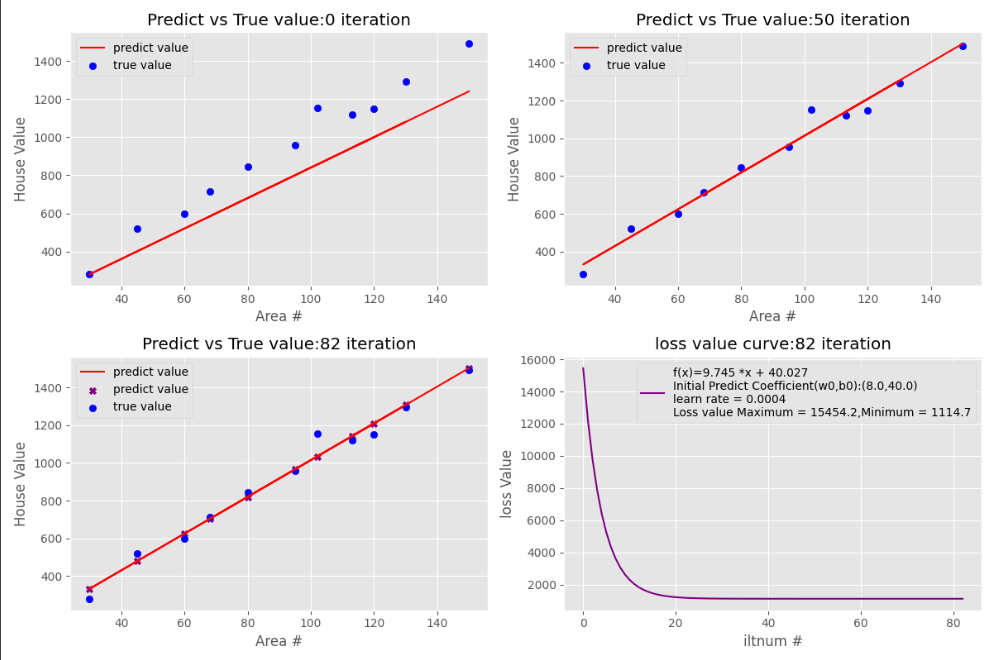
* m is the number of samples in house data set, e.g. m =11 means there are 11 samples in house data;
* is the true price of each house in the data set;
* For every house area , the prediction value is f()=w+b.

. In the loss function, there is two variables w and b, here try to use the gradient descent method to make *Loss*(w, b) to take the minimum value, then,

***∂Loss/∂w = w+b-) /m*** ①

***∂Loss/∂b = w+b-)/m*  ②**

Using Python to implement the gradient descent algorithm, give suitable ***initial value of( )*** and ***learning rate α***, we can get following figures of prediction and loss values:



A linear regression function learnt is: ***f(x) = 9.745\*x+40.27.***

From the Predict-true value contrast graph, we can find that the predicted regression function can match true value closely, the loss value convergent to a small digit, that means gradient descent method is valid.

### **2.2 multiple variables linear regression**

Single variable linear regression can be used to predict the price of house in the same district, but is too simple for the real world, so I try to use multiple variables linear regression. In three-variables linear regression, use area(x1), house age(x2), district(x3) as factors, linear regression ***f (, , ) = \*+ \*+ \* + b***, still adopt MSE as loss function, then,

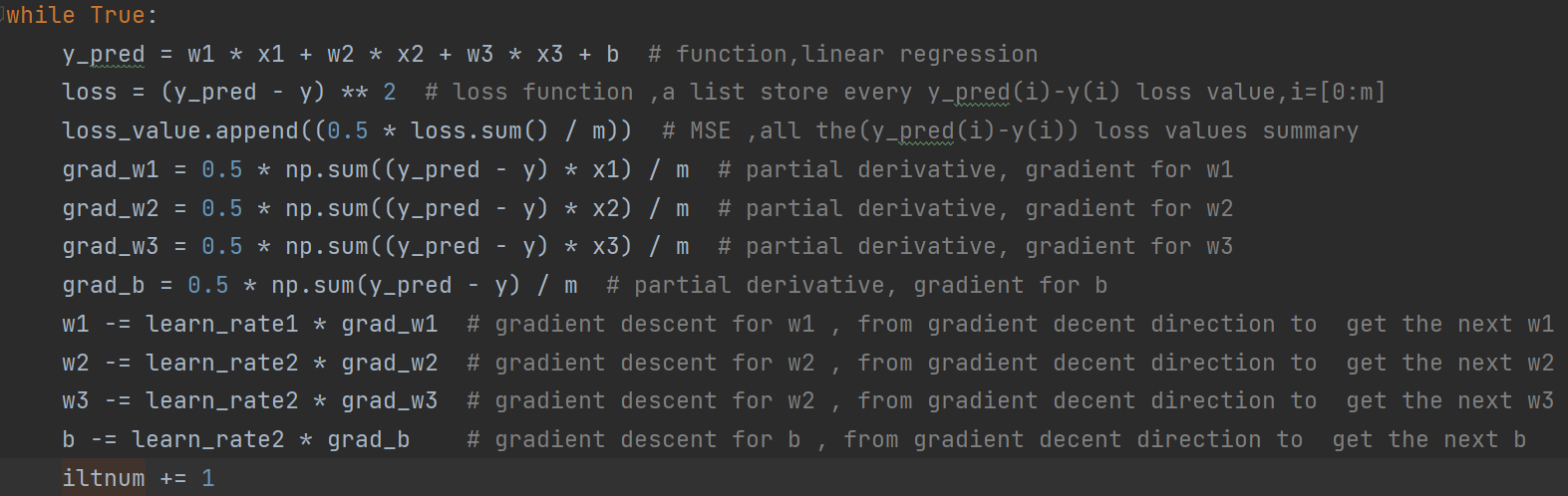
***∂Loss/∂ = + -) /m* ①**

***∂Loss /∂ = + -) /m* ②**

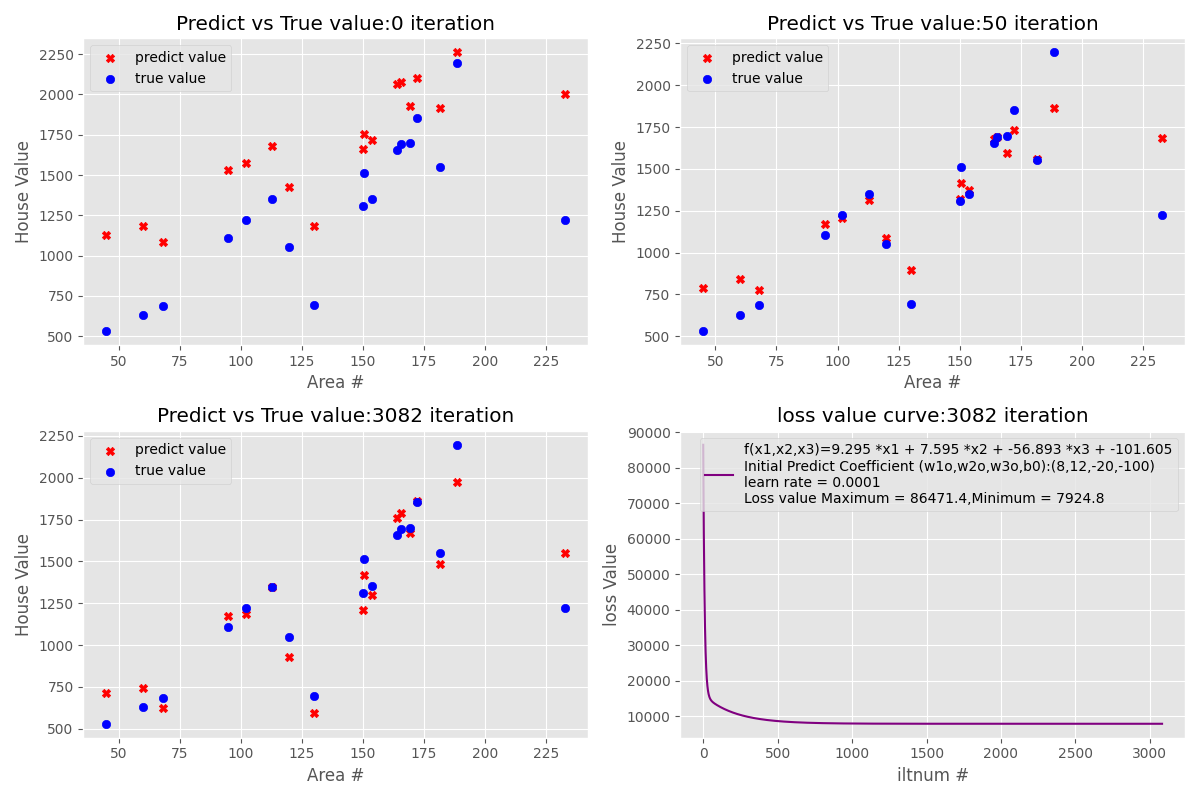
***∂Loss /∂ = + -) /m* ③**

***∂Loss /∂b = + -) /m* ④**

gradient descent algorithm main Python code segment:



Give suitable ***initial value of()*** and ***learning rate α***, after thousands of iteration computing, we can get figures of prediction and loss values:



A linear regression function learnt is: ***f (, , ) = 9.295\*+ 7.595\*-56.893\* -101.605***.

# **3.Python Programming**

In the Project, I use ***numpy*** to compute arrays and matrices, use ***Matplotlib***  for producing figures  of regression functions and loss-value, and ***Pandas*** to read house data stored in Excel [house\_value.xlsx](https://github.com/johnsonPcj/PythonProject/blob/master/house_value.xlsx).

All Python codes and test data are released at [*https://github.com/johnsonPcj/PythonProject*](https://github.com/johnsonPcj/PythonProject), [gradient\_descent1.py](https://github.com/johnsonPcj/PythonProject/blob/master/gradient_descent1.py) is a single-variable linear regression file , [gradient\_descent2.py](https://github.com/johnsonPcj/PythonProject/blob/master/gradient_descent2.py) is a two-variable linear regression file ,[gradient\_descent3.py](https://github.com/johnsonPcj/PythonProject/blob/master/gradient_descent3.py) is a three-variable linear regression file.

# **4.Conclusions**

It is found in the project that in order to minimize the loss function value and the prediction accuracy, setting learning rate and coefficient initial value is a difficult problem. I output many figures of the predicted house price and loss values under different learning rates and coefficient initial values for further study of more complex gradient descent algorithms.