Supervised Learning (Github)

(Report no.1)

Bhagyesh Kumar (2409661330)
Cyber Physical Systems

Keywords Linear Regression · Logistic regression · Gradient Descent · Regularisation · Cost function

1.Supervised Learning

It is a part of field where computers are given ability to learn without being explicitly programmed. In supervised learning it learns after given right answers.

1.1 Regression:
Predicts numbers of infinitely many possible outcomes.
1.2 Classification
Predict categories small number of possible outcomes.

1.1 Applications

- Spam filtering
- Speech recognition
- machine translation
- online advertising
- Self driving car
- Visual Inspector

With the addition of computer vision for object detection and real-time decision-making, robots can operate more effectively in real-world industries such as healthcare, manufacturing, and autonomous vehicles.

2. Unsupervised Learning

Finds pattern in the given data without being fed right answers. Clustering, Anomaly detection, Dimensional reduction. Algorithm finds structure in the given data.

3 Linear regression

3.1 Overview

It involves predicting an appropriate output after using given data sets to train. The deviation from the desired output of each dataset is calculated as cost and we aim to minimise the cost to present us with a linear function which can give output based on results.

$$f_{w,b}(x) = wx + b$$

f(x) is used to represent the function, where w and b are parameters, and x and y is input and output of our model.

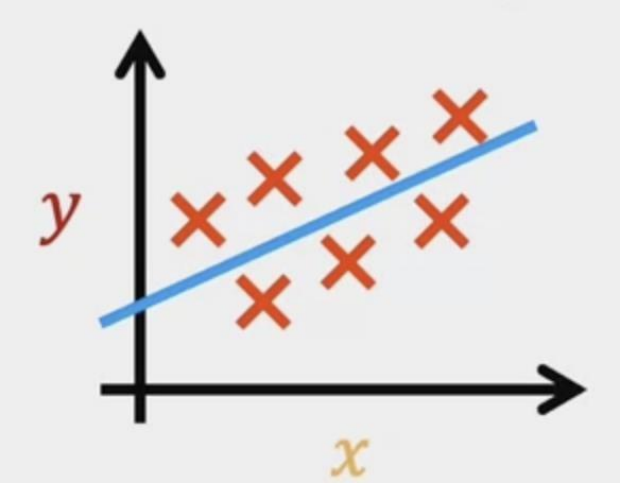


Fig.1 Linear regression function

The red cross in the fig.1 represents the various output in the dataset. To get a linear function which can best predict the output closest to these outputs is the goal here.
•It is a single variable

3.2 Cost function

- To present us with a model which can predict the values, we begin by defining a cost function which takes the difference from the target result to our output.
- The result is squared for each data set and after repeating of this process for all the training examples, summation is taken and is represented:

$$J(w,b) = \frac{1}{2m}\sum_{i=1}^{m}\left(\underbrace{\hat{y}^{(i)} - y^{(i)}}_{\text{error}}\right)^2$$

m = number of training examples

- This type of function is called squared cost function.

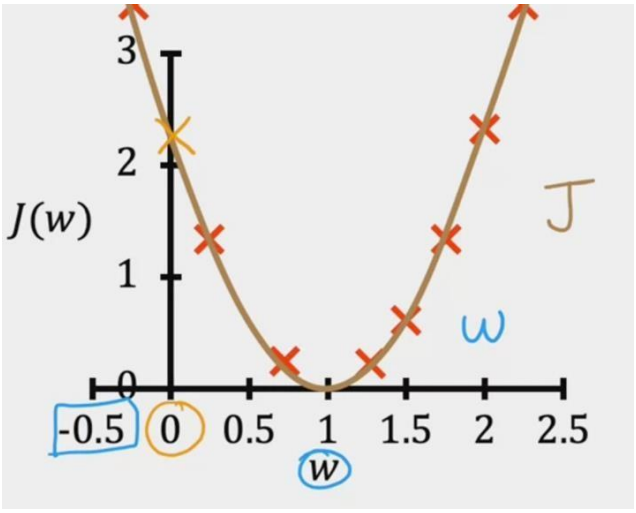


Fig. 2 Cost function

- If we consider a single parameter to be active than the above graph shows us the cost function which provided us insight about how adjustment of w parameter can affect the cost function.
- We are looking for a parameter where we can find the minimum cost in training the model as it will be closest to the target y.
- Therefore, next we will focus on minimising the cost function.

3.3 Gradient descent:

It is algorithm responsible for finding us the minimum values for our cost function. It is widely used un training linear regression, and in addition to it, it is also used in training neural networks, deep learning models.

Before beginning, we have to know how to visualize cost function, In the previous section we used only 1 parameter but, actually there are two parameter used w and b, which results in a 3d graph of cost function.

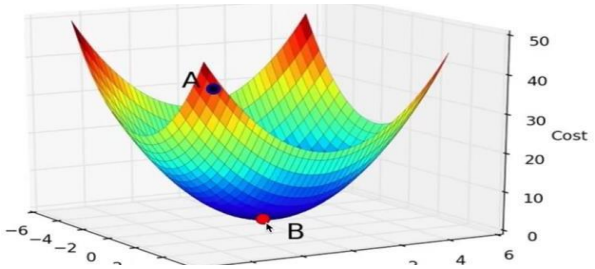


Fig3. Cost function of 2 parameter.

We can clearly see the minima in the depicted figure, and to find the same we do it finding the partial derivative of cost function with respect to both parameters.

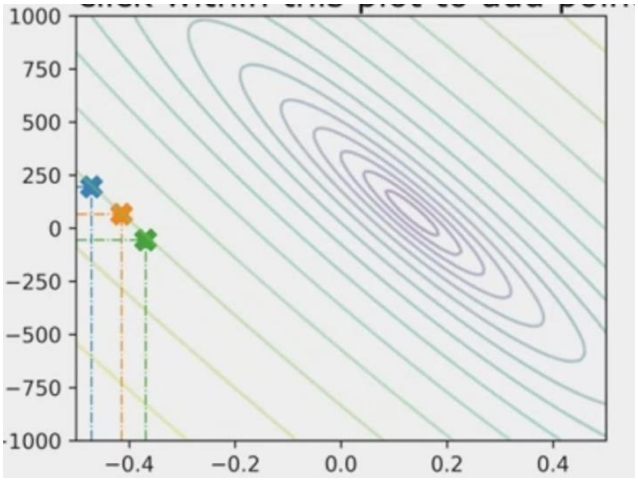


Fig4. Contour graph can also be used to represent the 3d graphs like of cost function.

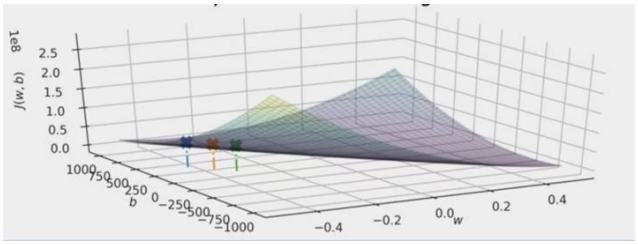


Fig5. Another example of 3d graph of cost function

We start at any point on the 3d graph of cost function and starts to take small steps towards our minima, to replicate this behaviour using maths we use:

$$tmp\_w = w - \alpha \frac{\partial}{\partial w} J(w,b)$$

$$tmp\_b = b - \alpha \frac{\partial}{\partial b} J(w,b)$$

Here α represent the learning rate. We are trying to adjust the parameter by subtracting it with the derivative of cost function, which basically gives us the direction where the local minima would lie. α gives us the rate of our descent, which have to be carefully manged as taking a large value would mean we might overshoot our minima, and a lower value would result in lower efficiency of the learning model.

- One key point to remember is to always assign the value of parameters simultaneously.

We must repeat it over multiple training sets until convergence is achieved to achieve a proper working model.

The Gradient descent algorithm can also be represented by following which can be obtained substituting values of partial function in case of both parameters.

$$w = w - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(f_{w,b}\left(x^{(i)}\right) - y^{(i)}\right) x^{(i)}$$

Fig8 For w parameter

$$b = b - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(f_{w,b}\left(x^{(i)}\right) - y^{(i)}\right)$$

Fig6 For b parameter

3.4 Learning Rate

It is helpful in improving the efficiency of the gradient descent and thereby of our model.

- If α is very small, our descent could be incredibly slow, and it would take more time to find a minimal. It won't be preferred as the efficiency of the model reduced significantly

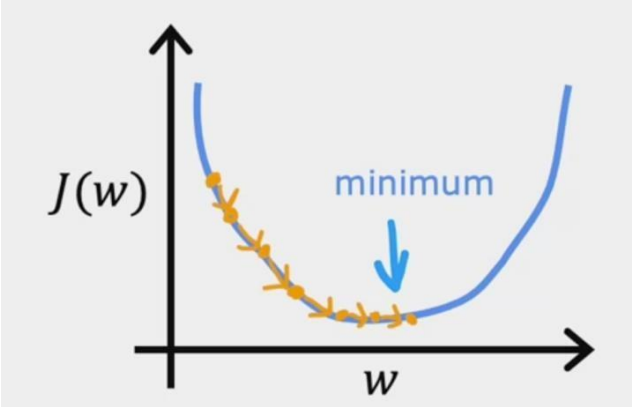


Fig6. When α with very small value.

- However, if the α is very large, as mentioned earlier it may overshoot and fail to converge.

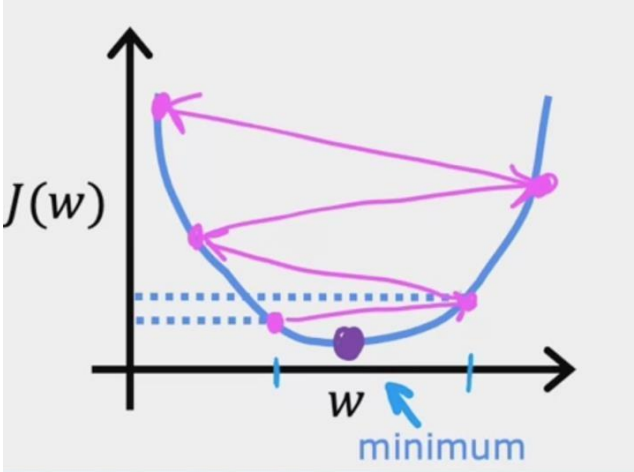


Fig7. When α is very large

After training the model over a large set of values, we can find a suitable pair of parameters where we can find the minima of our cost function uing the gradient descent algorithm.

4. Linear regression (Multiple Variable)

4.1 Overview

Instead of relying on just one feature or input, we can make use of multi variable in the linear regression. Similar to single variable function, we represent it by:

$$f_{w,b}(x) = w_1 x_1 + w_2 x_2 + \cdots + w_n x_n + b$$
$$\vec{w} = [w_1 \; w_2 \; w_3 \ldots w_n]$$
$$b \text{ is a number}$$

We make use of very important feature of NumPy library here by using vectorisation.

4.2 Vectorisation

It can help in making the model overall faster and also makes our codes shorter, as it doesn't rely on normal loop or arithmetic multiplication methods to calculate the above function.

When we solve their above function using loop or arithmetic multiplication, we tend to repeat the same process over large number of times.

$$f_{\vec{w},b}(\vec{x}) = \vec{w} \cdot \vec{x} + b =$$

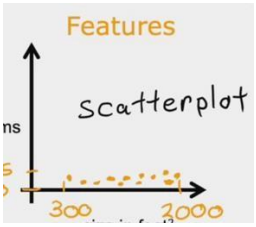
dot product

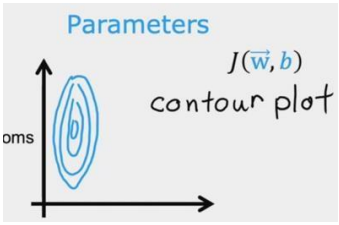
But relying on NumPy library, we can use dot product to solve the same. It makes uses of SIMD to do multiple computation at a single time saving our time and cost.

w[0]	w[1]	...	w[15]
*	*	...	*
x[0]	x[1]	...	x[15]

4.3 Feature Scaling

It is used to standardise a range of input variable in a dataset to make sure no particular dominate the training process due to its large range. We scale the feature over a particular range to avoid this using:





a. Mean Normalisation- Feature scaling where the feature values centres around zero.
b. Z-Score Normalisation- When the feature values are scaled to have a mean of 0 and s standard deviation of 1.

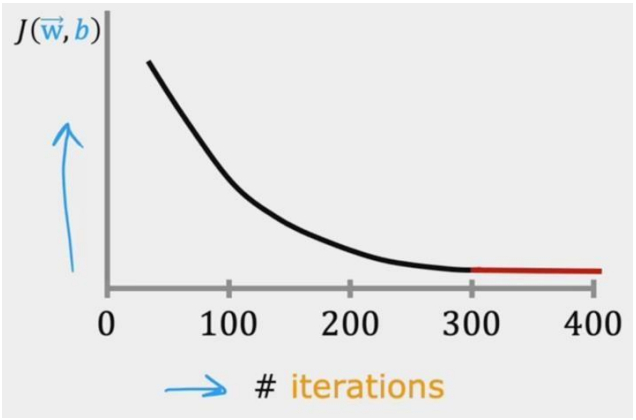
It helps in make the gradient descent runs faster.

4.4 Checking Gradient Descent

To ensure our algorithm is working properly, we can check it by plotting a cost graph with number of iterations. Gradient descent objective: min [J(w, b)]

- If the learning curve doesn't converge that means either there is bug in our code, or the alpha is too large.
- Automatic convergence can be used to detect it by declaring convergence if our cost function J (w, b) decreases by epsilon(10^-3) in one convergence.
- We can find plot using low learning rate, and if it doesn't work, that means, there is a bug in our code.

This method can be used to check if it doesn't work even though we have correct learning rate.



4.5 Feature Engineering

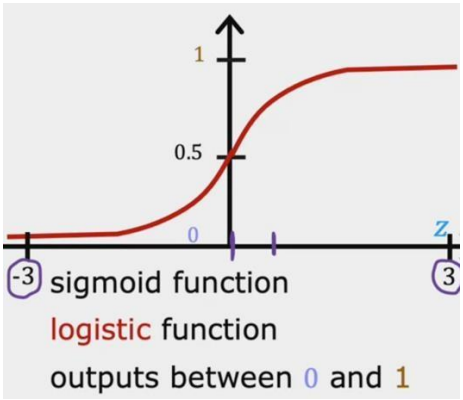
- Use of intuition to design a new feature by transforming or combining features
- It can make use of features which can be more useful for finding desired output

$$x_3 = x_1 x_2$$

new feature

5. Logistic Regression

It is another type of supervised learning where the goal is to classify the input into a category unlike finding a numerical representation in linear regression. Linear regression can't be used for same task as it can misclassify the data in some cases if weights are added. Do it might work sometimes but it can't be used for classification into categories.



For this purpose, sigmoid function is used gives output 0 and 1. It is represented by:

$$g(z) = \frac{1}{1+e^{-z}}$$

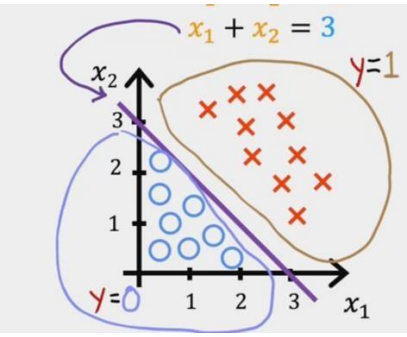
5.1 Decision Boundary

We are returned a probability between 0 and 1. To map this value in a category we select a threshold between 0 and 1.

For g(z) ≥ 0.5, y =1
For g(z) < 0.5, y = 0

Now, 0.5 act as a decision boundary. Similar example can be

g(z)=w1x1+w2x2+b
z = w1x1 +w2x2+ b =0 (w=1, w2=1, b=-3)
z = x1+x2-3 =0



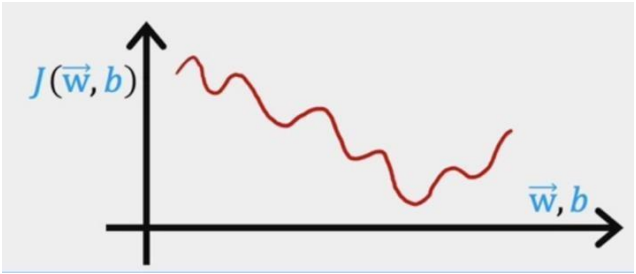
If y is labelled as 0 then f (w, b)(x)= 0.7 represents that there is 70% chance that it our model gives 0.

$$P(y = 0) + P(y = 1) = 1$$

This type of representation is used as we go forward, indicating that sum of probability when y=0 and when y=1 is equal to 1

5.2 Squared error cost

Unlike Linear regression, it can't be used here as we have too many local minima here, we get a wiggly shape in the surface graph. It results into a non-convex graph.



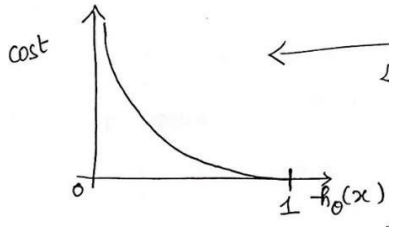
5.3 Logistics Loss Function:

Our new function make use of 2 values representing two categories, 1(Positive) and 0 (Negative).

$L\left(f_{\vec{w},b}(\vec{x}^{(i)}), y^{(i)}\right)$

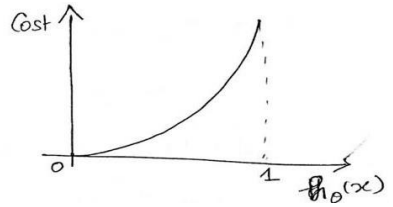
For y = 1 we use loss function as,

$-\log\left(f_{\vec{w},b}(\vec{x}^{(i)})\right)$



And for y =0 we have L=

$-\log\left(1 - f_{\vec{w},b}(\vec{x}^{(i)})\right)$



The loss represents the loss in the confidence of correctly labelling y as 0 or 1. The new function can be represented in a simplified form as

$$L\left(f_{\vec{w},b}(\vec{x}^{(i)}), y^{(i)}\right) = -y^{(i)}\log\left(f_{\vec{w},b}(\vec{x}^{(i)})\right) - (1 - y^{(i)})\log\left(1 - f_{\vec{w},b}(\vec{x}^{(i)})\right)$$

The cost function can then be calculated by:

$$\text{cost}\quad J(\vec{w}, b) = \frac{1}{m}\sum_{i=1}^{m}\left[L\left(f_{\vec{w},b}(\vec{x}^{(i)}), y^{(i)}\right)\right]$$

In the final step we can calculate the gradient descent using the same formula, but the value of function is represented by sigmoid this time.

$$w_j = w_j - \alpha\left[\frac{1}{m}\sum_{i=1}^{m}\left(f_{\vec{w},b}(\vec{x}^{(i)}) - y^{(i)}\right)x_j^{(i)}\right]$$

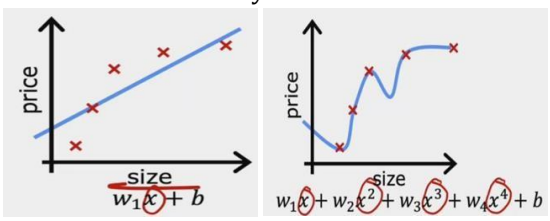
$$b = b - \alpha\left[\frac{1}{m}\sum_{i=1}^{m}\left(f_{\vec{w},b}(\vec{x}^{(i)}) - y^{(i)}\right)\right]$$

- We find similarity in linear and logistic regression in terms of monitoring gradient descent, vectorisation implementation, and feature scaling.

5.4 Problem of Overfitting

We can face problem when our algorithm either not fits our dataset or it fits too well.

a. Underfit: When our algorithm does not fit our training data very well because it limited by having a bias. It tends to have high bias. It can be due to preconception like of a linear function expecting ideal function linearly.



b. Overfit: When the data fits our algorithm too perfectly, it results in having high variance and it function changes very rapidly even if a little variance is seen in data set.

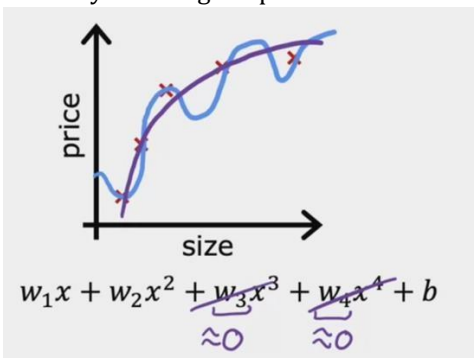
Similar problems are faced in classification with linear showing underfit, and polynomial of higher degree shows overfit.

5.5 Addressing overfit

- Overfit can be solved by collecting more data for our training set
- We can select feature to include/ exclude to make our algorithm fit just right.
- Regularisation is used to reduce size of the parameters.

5.6 Regularisation

- It is used to overcome overfit in a training model by reducing the parameter w.



If we are not aware of which parameter to reduce to solve overfit, we can introduce a new term regularisation parameter which is then added to it to overcome overfitting.

$$\frac{\lambda}{2m}\sum_{j=1}^{n} w_j^2$$

For Linear regression,

$$J(\vec{w},b) = \frac{1}{2m}\sum_{i=1}^{m}\left(f_{\vec{w},b}(\vec{x}^{(i)}) - y^{(i)}\right)^2 + \frac{\lambda}{2m}\sum_{j=1}^{n} w_j^2$$

"lambda"

If lambda is too small, it could lead to underfitting, and overfitting if the term is too large.

We have to balance it to find the just right, to make sure both of them doesn't happen. Our new gradient descent will be:

$$w_j = w_j - \alpha\left[\frac{1}{m}\sum_{i=1}^{m}\left[\left(f_{\vec{w},b}(\vec{x}^{(i)}) - y^{(i)}\right)x_j^{(i)}\right] + \frac{\lambda}{m}w_j\right]$$

Th parameter doesn't change as we don't regulate b parameter. We are basically shrinking our parameter w with every step in the gradient descent.

Similarly, Regularised logistic regression is used as it also tends to get overfit:

$$J(\vec{w},b) = -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}\log\left(f_{\vec{w},b}(\vec{x}^{(i)})\right) + (1-y^{(i)})\log\left(1 - f_{\vec{w},b}(\vec{x}^{(i)})\right)\right] + \frac{\lambda}{2m}\sum_{j=1}^{n} w_j^2$$

$\min_{\vec{w},b}$

And gradient descent algorithm is changed by adding lambda parameter.

repeat {

$$w_j = w_j - \alpha\frac{\partial}{\partial w_j}J(\vec{w},b) = \frac{1}{m}\sum_{i=1}^{m}\left(f_{\vec{w},b}(\vec{x}^{(i)}) - y^{(i)}\right)x_j^{(i)} + \frac{\lambda}{m}w_j$$

$j = 1 \ldots n$

B parameter isn't regularised.

6. Miscellaneous

Randomness and unpredictability are the two main components of a regression model.

Prediction = Deterministic + Statistic

Deterministic part is covered by the predictor variable in the model. Stochastic part reveals the fact that the expected and observed value is unpredictable. There will always be some information that are missed to cover. This information can be obtained from the residual information.

Characteristics of a residue:

1)Residuals do not exhibit any pattern

2)Adjacent residuals should not be same as they indicate that there is some information missed by system.

- Regression of sum of squares:

$$\text{Error} = \sum_{i=1}^{n} (\text{Predicted\_output} - \text{average\_of\_actual\_output})^2$$

- Total sum of squares

$$\text{Error} = \sum_{i=1}^{n} (\text{Actual\_output} - \text{average\_of\_actual\_output})^2$$

- Corelation coefficient(r)-

$r = (+/-) \sqrt{r^2}$

- Null hypothesis is the initial claim that researcher specify using previous research or knowledge.
- Low P-value: Rejects null hypothesis indicating that the predictor value is related to the response
- High P-value: Changes in predictor are not associated with change in target

Reference:

1. Supervised Learning by Andrew
2. Linear Regression TowardsDataScience
3. Regularization
4. Simple Linear Regression Tutorial
5. Linear Regression Cheat Sheet
6. Gradient Descent for Machine Learning
7. Logistic Regression Overview
8. Logistic Regression — Detailed Overview
9. Logistic Regression for Machine Learning
10. Logistic Regression Cheat Sheet