1. Ordinary Linear Regression

Statement of problem: Given a training set *X* (m by n matrix with each row as one input), and a corresponding observed outputs(m by 1 vector, each element corresponds to one input row), we hope to find a parameter vector (n by 1 vector), such that the system is linearly expressed as:

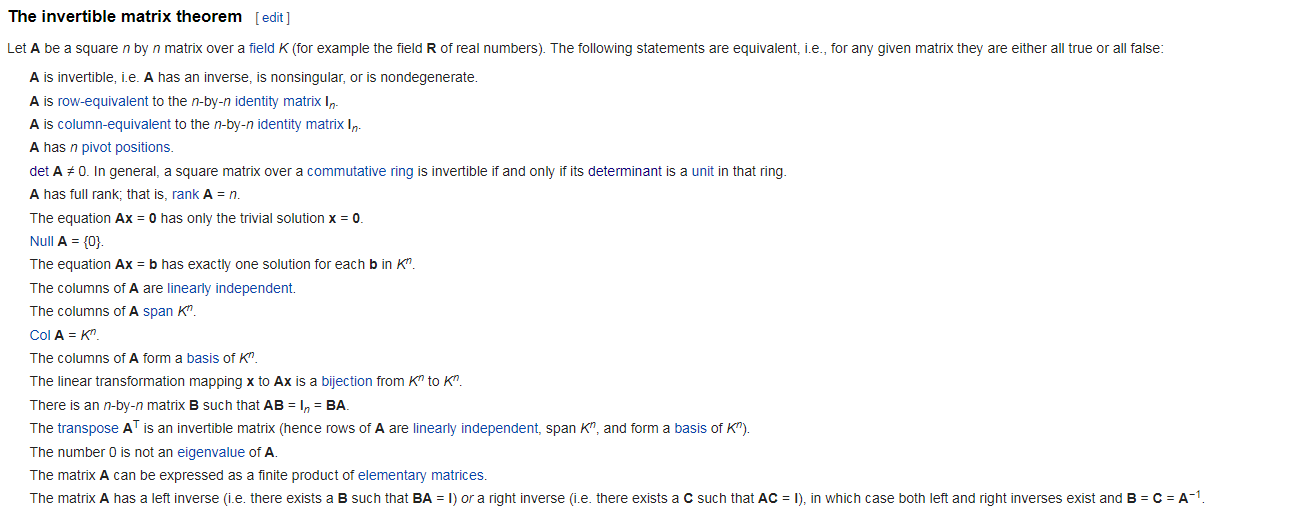
Where represents perturbations such as noise.

We can use least squares as the cost function in this problem:

By taking the derivative of *S* over and set it to zero we could find the most suitable values for :

If is invertible (equivalent to *X* being full column rank!), then we can directly solve :

Note: rank = rank . See: <https://math.stackexchange.com/questions/349738/prove-rank-ata-rank-a-for-any-a-m-times-n>.



However, in more general cases, is not invertible. Hence we can take the derivative of log-likelihood for only one training data pair () over and use **stochastic gradient descent** method to numerically solve for the best :

For gradient **descent**, we use the opposite number to update the parameters.

We carry out the iterative method as:

Loop{

for i = 1:m {//Every training set

for j = 1:n { //Every parameter

}

}

}

Note: Whereas batch gradient descent has to scan through the entire training set before taking a single step—a costly operation if m is large—stochastic gradient descent can start making progress right away, and continues to make progress with each example it looks at. Often, stochastic gradient descent gets θ “close” to the minimum much faster than batch gradient descent. (Note however that it may never “converge” to the minimum, and the parameters θ will keep oscillating around the minimum of J(θ); but in practice most of the values near the minimum will be reasonably good approximations to the true minimum.

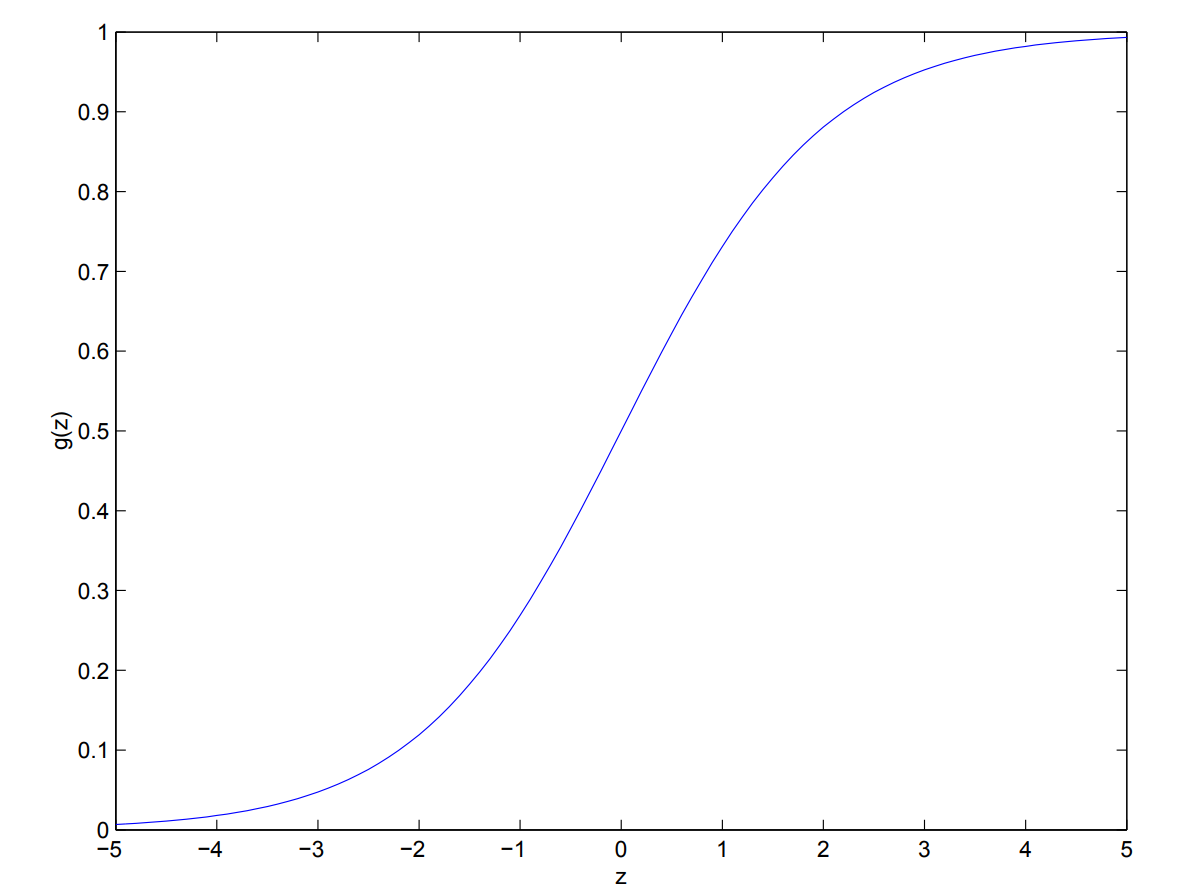
**Probabilistic interpretation**:

For the system mentioned in the first equation, we can also approach the problem using maximum-likelihood method. If we assume that the distribution of is i.i.d. normal, then the maximum-likelihood method results in:

**The above equation is EXACTLY the OLS fitting method! (Hence we can also say that the OLS method is a correct estimation iff is i.i.d. normal distribution)**

1. Logistic Regression

The form of logistic function:



The purpose that logistic regression was introduced was to solve classification problems. Remember the derivative of *g(z)*:

The above form turned out to be VERY useful in later derivations!

Statement of problem in a classification problem: Given a training set *X* (m by n matrix with each row as one input), and a corresponding observed outputs(m by 1 vector, each element corresponds to one input row and **CAN ONLY BE 0 OR 1**), we hope to find a parameter vector (n by 1 vector), such that the probability for an input set to belong to class 1 is expressed as:

Denote:

We can re-write the probability function as:

Note: be careful that here represents the OBSERVED class, instead of the predicted possibility .

Again, we can use maximum-likelihood method to solve for the optimum parameter :

Again, we can take the derivative of log-likelihood for only one training data pair () over and use **stochastic gradient ascent** method to numerically solve for the best !

We carry out the iterative method as:

Loop{

for i = 1:m {//Every training set

for j = 1:n { //Every parameter

}

}

}

**Note: The above stochastic update equation is exactly the same as that for OLS, even though we started out from a totally different data model!**