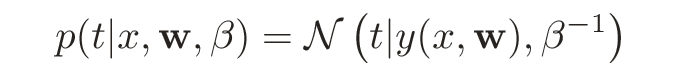
# Bayesian Curve Fitting

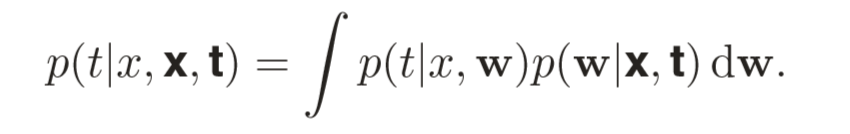
In [statistics](https://en.wikipedia.org/wiki/Statistics), Bayesian linear regression is an approach to [linear regression](https://en.wikipedia.org/wiki/Linear_regression) in which the statistical analysis is undertaken within the context of [Bayesian inference](https://en.wikipedia.org/wiki/Bayesian_inference). We will find the likelihood function and max it to fit curve.

The goal in the curve ﬁtting problem is to be able to make predictions for the target variable t given some new value of the input variable x on the basis of a set of training data comprising N input values x = (x1,...,xN)T and their corresponding target values t = (t1,...,tN)T. We can express our uncertainty over the value of the target variable using a probability distribution. For this purpose, we shall assume that, given the value of x, the corresponding value of t has a Gaussian distribution with a mean equal to the value y(x,w) of the polynomial curve. We have:

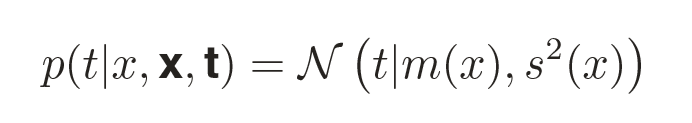


where, for consistency with the notation in later chapters, we have deﬁned a precision parameter β corresponding to the inverse variance of the distribution.

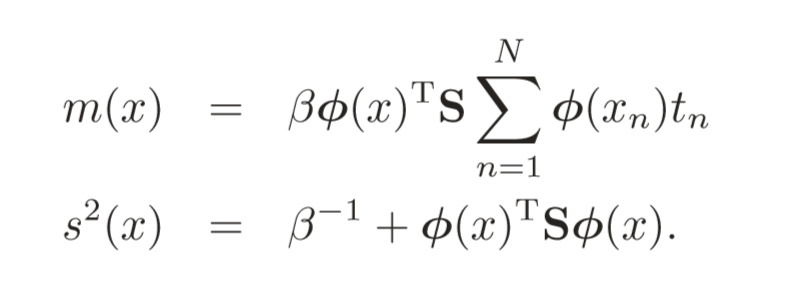
In the curve ﬁtting problem, we are given the training data x and t, along with a new test point x, and our goal is to predict the value of t. We therefore wish to evaluate the predictive distribution p(t|x,x,t). Here we shall assume that the parameters α and β are ﬁxed and known in advance (in later chapters we shall discuss how such parameters can be inferred from data in a Bayesian setting). A Bayesian treatment simply corresponds to a consistent application of the sum and product rules of probability, which allow the predictive distribution to be written in the form



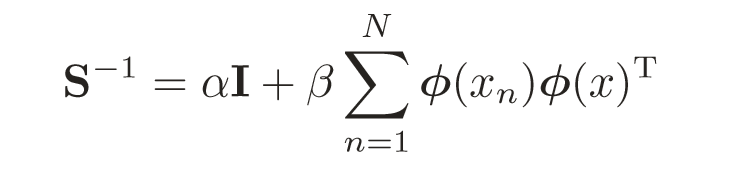
Here p(t|x,w) is given above, and we have omitted the dependence on α and β to simplify the notation. Here p(w|x,t) is the posterior distribution over parameters. We shall see that, for problems such as the curve-ﬁtting example, this posterior distribution is a Gaussian and can be evaluated analytically. Similarly, the integration in above can also be performed analytically with the result that the predictive distribution is given by a Gaussian of the form



where the mean and variance are given by



Here the matrix S is given by



where I is the unit matrix, and we have deﬁned the vector φ(x) with elements φi(x) = xi for i = 0,...,M.

How to use Bayesian Curve Fitting in stock prediction

We apply Bayesian Curve Fitting in short term prediction.

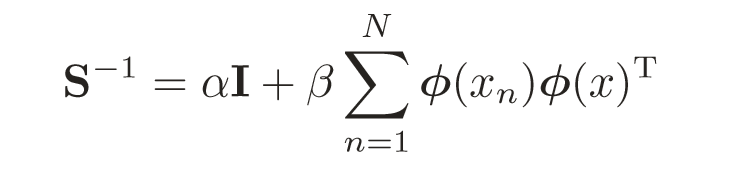
Basically, you need to get the training data to fit the curve, so the first step is choose appropriate data to do the prediction.

We choose the latest 10 real-time prices of one stock. And assuming that every price has an index from 1 to 10. So we can get a set of training data comprising N input values x = (x1,...,xN)T and their corresponding target values t = (t1,...,tN)T.

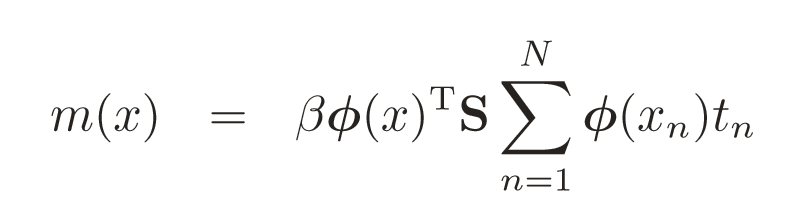
Then, we can increase the index to get the new value of the input variable x.

As for the parameters α and β, we can assume that the values are fixed. We set α=0.0047 and β= 0.225.

After these steps, we can calculate



For the stock prediction, we only care about the price which will has the greatest probability. So we only need to calculate the m(x):



# Artificial Neural Network

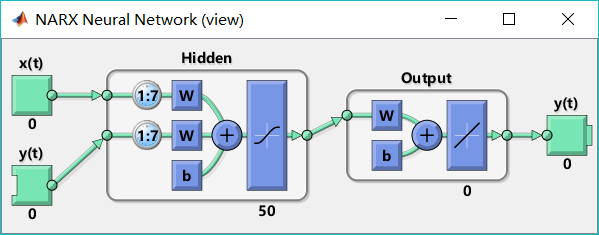
Artificial neural networks (ANNs) are a family of models inspired by [biological neural networks](https://en.wikipedia.org/wiki/Biological_neural_network) (the [central nervous systems](https://en.wikipedia.org/wiki/Central_nervous_system) of animals, in particular the [brain](https://en.wikipedia.org/wiki/Brain)) and are used to estimate or [approximate](https://en.wikipedia.org/wiki/Universal_approximation_theorem) [functions](https://en.wikipedia.org/wiki/Function_(mathematics)) that can depend on a large number of [inputs](https://en.wikipedia.org/wiki/Argument_of_a_function) and are generally unknown. Artificial neural networks are generally presented as systems of interconnected "[neurons](https://en.wikipedia.org/wiki/Artificial_neuron)" which exchange messages between each other. The connections have [numeric weights](https://en.wikipedia.org/wiki/Weighting) that can be tuned based on experience, making neural nets adaptive to inputs and capable of learning.

These networks are excellent for designing the behavior of more complicated structures because of their ability to learn. A major advantage in using this method for what we seek to accomplish is that they can be used to model a system of events, stock trends, that are totally unknown and how it will react to distortion and interference or in our case the imperfections of the stock market.

How to use ANN in stock prediction

We use MATLAB neural network toolbox to generate the ANN. This means through the use of this software we can adapt the information we know about stocks to their system that is already set up to handle neural networks.

In order to use toolbox of neural network in MATLAB, we must prepare data to train the network. This data is imported from database which we fetch from stock website. And we will redefine this data to fit the toolbox.



We use the whole one past year’s close price as the train data. The date will be redefined as a number as input, and the close price can be target output. Once the data is ready, the neutral network can be trained.

We set the delay of input is 7, and the number of hidden layer is 50.

After the networks has been set, we will redefine the next days as numbers, and input these numbers to the network. The network will calculate the predictive value of current input under the training weights. So, we can get the predictive results of the next days.