

Basic Linear Regression

by Grady Jensen

Reading Material:

"Pattern Recognition and Machine Learning" by Bishop [ch. 3.1, 3.2, 3.6]

Further extra reading:

"Machine Learning: A Probabilistic Perspective" by Murphy [ch. 7.2 - 7.3, 7.5.4,]

Note: these slides are adapted from slides originally by Christian Osendorfer
Most figures are from C. Bishop: "Pattern Recognition and Machine Learning"

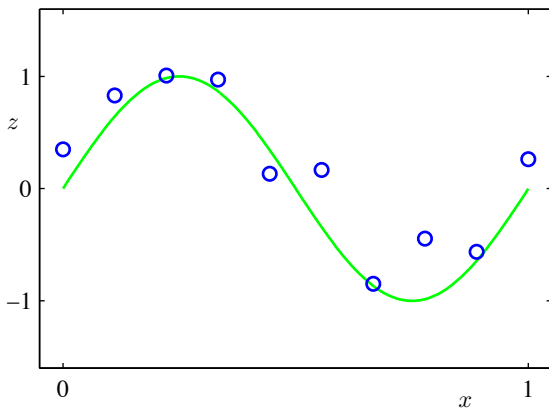
Notation

Symbol	Meaning
--------	---------

s	a scalar number is lowercase and not bold
S	A vector is uppercase
\mathbf{S}	a matrix is uppercase and bold
$y(X)$	predicted value of inputs X
z or Z	vector of targets
z_i or Z_i	The target of the i 'th example
w_0	a bias term (not to be confused with bias in general)
$\phi()$	a basis function
$E()$	an error function
$\tilde{E}()$	regularized error function
\mathcal{D}	The training data
Φ^\dagger	Moore-Penrose pseudoinverse of Φ

There is not a special symbol for vectors or matrices augmented by the bias term, w_0 . Assume it is always included.

A noisy real-valued function



$$\text{inputs: } X = (x_1, \dots, x_N)^T \quad (1)$$

$$\text{targets: } z = (z_1, \dots, z_N)^T, \quad z_i = y(x_i) + \epsilon = \sin(2\pi x_i) + \epsilon \quad (2)$$

Fitting a function to data.

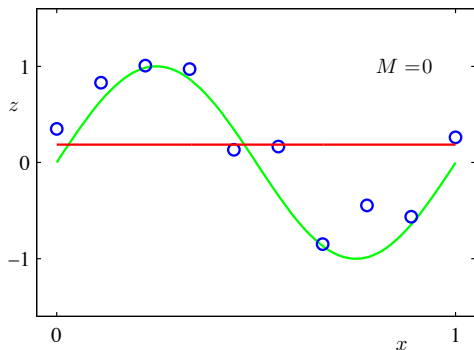
Common approach: **Sum of squared Errors**

Try to minimize:

$$E_{\mathcal{D}}(W) = \frac{1}{2} \sum^n [y(x_n, W) - z_n]^2 \quad (3)$$

We learn the coefficients, W , by minimizing some of the error function that gives us a measure for the "misfit" between our model and the data points we have.

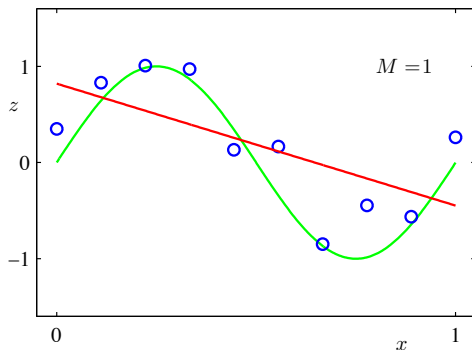
Model: 0th order polynomial



$$y(x_n, w) = w_0$$

$$E_{\mathcal{D}}(w_0) = \frac{1}{2} \sum^n (w_0 - z_n)^2$$

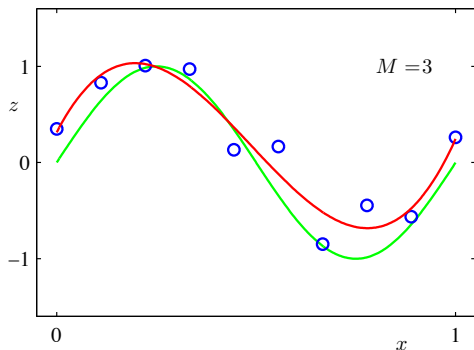
Model: 1st order polynomial



$$y(x_n, W) = w_0 + w_1 x$$

$$E_{\mathcal{D}}(W) = \frac{1}{2} \sum^N (w_0 + w_1 x_n - z_n)^2$$

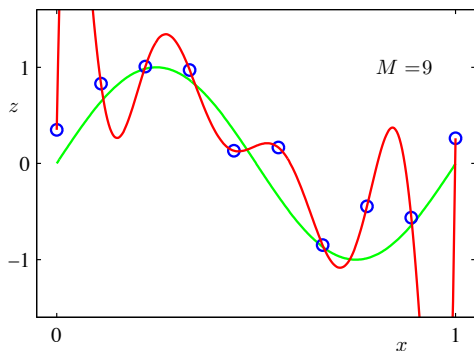
Model: 3rd order polynomial



$$y(x_n, W) = w_0 + w_1 x + w_2 x^2 + w_3 x^3$$

$$E_{\mathcal{D}}(W) = \frac{1}{2} \sum^n [y(x_n, W) - z_n]^2$$

Model: 9th order polynomial



$$y(x_n, W) = \sum_{j=0}^M w_j x^j$$

$$E_{\mathcal{D}}(W) = \frac{1}{2} \sum_{n=1}^N [y(x_n, W) - z_n]^2$$

Problem Definition

We have input vectors X and associated output values z . We want to describe the underlying functional relation between them.

What about the following simple model? (Look familiar?)

$$y(X_n, W) = w_0 + \sum_{j=1}^{M-1} w_j \phi_j(X_n) = W^T \Phi(X_n) \quad (4)$$

where

- ϕ **basis function** — many choices, can be nonlinear
- w_0 **bias** — equivalent to defining $\phi_0 \equiv 1$

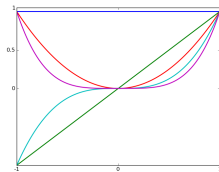
It is **linear** in W !

$$E_{\mathcal{D}}(W) = \frac{1}{2} \sum^n [W^T \Phi(x_n) - z_n]^2 \quad (5)$$

Typical Basis Functions

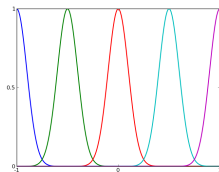
Polynomials

$$\phi_j(x) = x^j$$



Gaussian

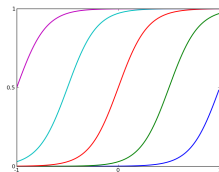
$$\phi_j(x) = e^{-\frac{(x-\mu_j)^2}{2s^2}}$$



Logistic Sigmoid

$$\phi_j(x) = \sigma\left(\frac{x-\mu_j}{s}\right),$$

where $\sigma(a) = \frac{1}{1+e^{-a}}$



Optimal Solution

To compute the optimal solution, we need to minimise the error function $E_{\mathcal{D}}$ with respect to W :

$$E_{\mathcal{D}}(W) = \frac{1}{2} \sum^n (W^T \Phi(x_n) - z_n)^2 = \frac{1}{2} (\Phi W - z)^T (\Phi W - z) \quad (6)$$

with

$$\Phi = \begin{pmatrix} \phi_0(X_1) & \phi_1(X_1) & \dots & \phi_{M-1}(X_1) \\ \phi_0(X_2) & \phi_1(X_2) & & \vdots \\ \vdots & \vdots & \ddots & \\ \phi_0(X_N) & \phi_1(X_N) & \dots & \phi_{M-1}(X_N) \end{pmatrix}$$

being the **design matrix** of ϕ .

Optimal Solution

Therefore we can compute the gradient error function w.r.t. W by taking the derivative of $E_{\mathcal{D}}(W)$.

$$\nabla_W E_{\mathcal{D}}(W) = \frac{\partial}{\partial W} \frac{1}{2} (\Phi W - Z)^T (\Phi W - Z) \quad (7)$$

$$= -\Phi^T (\Phi W - Z) \quad (8)$$

$$= 0 \quad (9)$$

(The steps from (7) to (8) are not trivial, but Eqs. (71) and (81) of the matrix cook book can help you.) Setting the gradient to zero we get **normal equations** of (ordinary) least squares problem:

$$W_{\text{optimal}} = \underbrace{(\Phi^T \Phi)^{-1} \Phi^T}_{=\Phi^\dagger} z. \quad (10)$$

Φ^\dagger is called **Moore-Penrose pseudo-inverse** of Φ (because for an *invertible* square matrix, $\Phi^\dagger = \Phi^{-1}$).

Second derivative: $\Phi^T \Phi$ is (semi) positive definite.

Computational aspect

Computing the solution W_{Optimal} using the normal equations is not such a great idea. Why?

- ▶ If Φ is not full rank, $(\Phi^T \Phi)^{-1}$ does not exist (why? how can rank deficiency happen?)
- ▶ Even if Φ is full rank, it can be *ill-conditioned* (???), (i.e., $\kappa(\Phi)$ is large), $\Phi^T \Phi$ will be even worse ($\kappa(\Phi^T \Phi) = \kappa(\Phi)^2$).

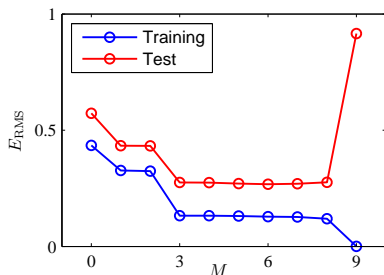
Applied numerical computing :)

Fighting Overfitting (Split up data)

Intuitively, the use of high level polynomials causes the equation to fit itself to the noise in the data.

- ▶ Use training and test sets of the data
- ▶ Use Root Mean Square (RMS) to normalize for differences in set sizes.

$$E_{\text{RMS}} = \sqrt{\frac{2E(W^*)}{N}}$$

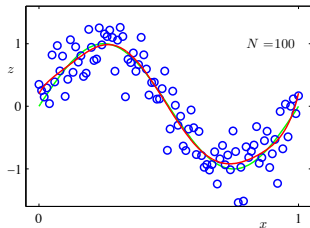
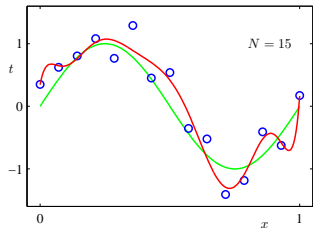


W^* is the value of W that minimizes the error function E

Fighting Overfitting

Heuristic:

of data points $>$ 5 or 10 times
number of parms.



Sadly, the number of parameters isn't always a good measure of complexity. A much better approach is to choose complexity based on problem to be solved.

Controlling overfitting with regularization

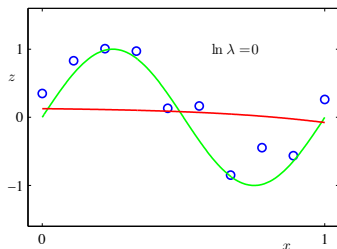
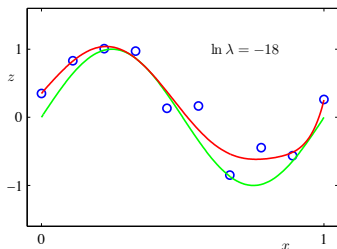
MLE often suffers from overfitting \rightsquigarrow use **regularisation**:

$$\widetilde{E}_{\mathcal{D}}(W) = \frac{1}{2} \sum^n [W^T \Phi(X_n) - Z_n]^2 + \frac{\lambda}{2} \|W\|_2^2 \quad (11)$$

Where,

$$\|W\|_2^2 \equiv W^T W = w_0^2 + w_1^2 + w_2^2 + \dots + w_m^2$$

λ = weight of regularization term



Penalty term discourages coefficients from reaching large values...

Techniques of regularization, like sum of squares, are called shrinkage methods in statistics because they reduce the value of coefficients.

Regularization

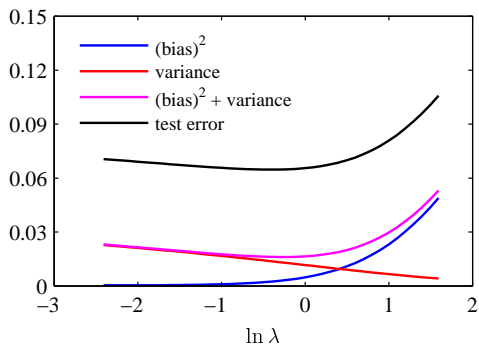
$$\widetilde{E}_{\mathcal{D}}(W) = \frac{1}{2} \sum^n [W^T \phi(x_n) - z_n]^2 + \frac{\lambda}{2} \|W\|_q \quad (12)$$

⇒ this is like a Lagrange term specifying an additional constraint:

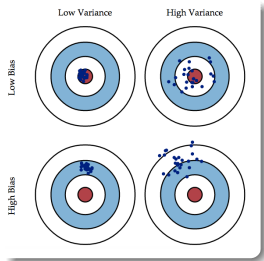
$$\sum^M \|w_j\|_q \leq \eta$$

↪ most often, use quadratic regulariser (ℓ_2 , ridge regression): $q = 2$, i.e.

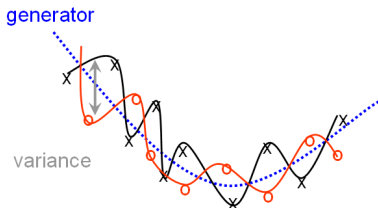
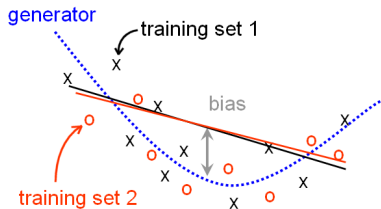
$$\sum^M \|w_j\|_2 = W^T W$$



Bias and Variance



→ **Regularization** ←
mediates between these extremes.



Bayesian Linear Regression

by Grady Jensen

Reading Material:

"Pattern Recognition and Machine Learning" by Bishop [ch. 3.3 - 3.3.2, 3.4 - 3.5]

Further extra reading:

"Machine Learning: A Probabilistic Perspective" by Murphy [ch. 7.4 - 7.6]

Note: these slides are adapted from slides originally by Christian Osendorfer
Most figures are from C. Bishop: "Pattern Recognition and Machine Learning"

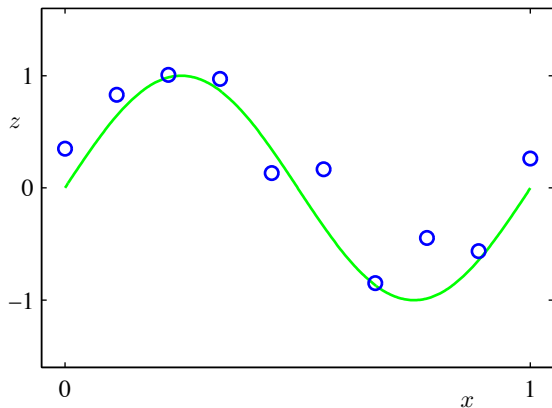
Notation

Symbol	Meaning
--------	---------

s	a scalar number is lowercase and not bold
S	A vector is uppercase
\mathbf{S}	a matrix is uppercase and bold
$y(X)$	predicted value of inputs X
z or Z	vector of targets
z_i or Z_i	The target of the i 'th example
w_0	a bias term (not to be confused with bias in general)
$\phi()$	a basis function
$E()$	an error function
$\tilde{E}()$	regularized error function
\mathcal{D}	The training data
Φ^\dagger	Moore-Penrose pseudoinverse of Φ

There is not a special symbol for vectors or matrices augmented by the bias term, w_0 . Assume it is always included.

A noisy real-valued function



$$\text{inputs: } X = (x_1, \dots, x_N)^T \quad (1)$$

$$\text{targets: } z = (z_1, \dots, z_N)^T, \quad z_i = y(x_i) + \epsilon = \sin(2\pi x_i) + \epsilon \quad (2)$$

Problem Definition

We have input vectors X and associated output values z . We want to describe the underlying functional relation between them.

What about the following simple model? (Look familiar?)

$$y(X, W) = w_0 + \sum_{j=1}^{M-1} w_j \phi_j(X) = W^T \Phi(X) \quad (3)$$

where

- ϕ **basis function** — many choices, can be nonlinear
- w_0 **bias** — equivalent to defining $\phi_0 \equiv 1$

$$E_{\mathcal{D}}(W) = \frac{1}{2} \sum^n [W^T \Phi(x_n) - z_n]^2 \quad (4)$$

Bayesian Regression

Remember from eq (2) at the start of the lecture,

$$z(X_n) = y(X_n, W) + \underbrace{\epsilon}_{\text{noise}}$$

Let's assume the values of the data have gaussian noise with a mean equal to $y(X_n, W) = w_0 + w_1 X_n + w_2 X_n^2 + \dots + w_m X_n^M = \sum_{j=0}^M w_j X_n^j$
then,

$$p(Z_n | X_n, W, \beta) = \mathcal{N}(Z_n | y(X_n, W), \beta^{-1}) \approx y(X_n, W) \quad (5)$$

Remember: Without any loss of generality, $y(X_n, W) = W^T \phi(X_n)$

Bayesian Likelihood

Assume the points are drawn i.i.d. and we get a likelihood function of

$$p(Z | X, W, \beta) = \prod_{n=1}^N \mathcal{N}(Z_n | y(X_n, W), \beta^{-1}) \quad (6)$$

Or formulated in a more explicit way:

$$\ln p(Z | X, W, \beta) = \frac{-\beta}{2} \sum_{n=1}^N [y(X_n, W) - Z_n]^2 + \frac{N}{2} \ln \beta - \frac{N}{2} \ln 2\pi \quad (7)$$

Why do we use \ln ?

Bayesian Maximum Likelihood

$$W_{\text{ML}} \Leftrightarrow \frac{\partial}{\partial W} \ln p(Z | X, W, \beta) = 0 \quad (8)$$

$$\Leftrightarrow \frac{\partial}{\partial W} \frac{-\beta}{2} \sum_{n=1}^N [y(X_n, W) - Z_n]^2 = 0 \quad (9)$$

$$\Leftrightarrow \frac{\partial}{\partial W} \frac{-1}{2} \sum_{n=1}^N [y(X_n, W) - Z_n]^2 = 0$$

$$\Leftrightarrow \frac{\partial}{\partial W} \frac{1}{2} \sum_{n=1}^N [y(X_n, W) - Z_n]^2 = 0$$

$$= E_{\mathcal{D}}(W)$$

Scaling likelihood by a constant only changes height, not location with respect to w

Minimize instead of maximize

see equation 4 from the Problem definition slide

What does this mean?

Bayesian ML continued...

Now let's take the derivative with respect to β to get the maximum likelihood value for β :

$$\frac{\partial}{\partial \beta} \ln p(Z | X, W, \beta) = 0 \quad (10)$$

$$\Leftrightarrow \frac{\partial}{\partial \beta} \frac{-\beta}{2} \sum_{n=1}^N [y(X_n, W) - Z_n]^2 + \frac{N}{2} \ln \beta - \frac{N}{2} \ln 2\pi = 0 \quad (11)$$

$$\Leftrightarrow \frac{-1}{2} \sum_{n=1}^N [y(X_n, W) - Z_n]^2 + \frac{N}{2\beta} = 0 \quad (12)$$

$$\Leftrightarrow \frac{1}{\beta_{\text{ML}}} = \frac{1}{N} \sum_{n=1}^N [y(X_n, W_{\text{ML}}) - Z_n]^2 \quad (13)$$

Predictive

Plugging in the W_{ML} and β_{ML} into our likelihood we get a predictive distribution that allows us to make predictions for new values of x .

$$p(z | x, W_{ML}, \beta_{ML}) = \mathcal{N}(z | y(x, W_{ML}), \beta_{ML}^{-1}) \quad (14)$$

Up until now we have been using maximum likelihood to solve our problems. An issue with ML approaches is that they can **overfit** the data (when there isn't a lot of data) because it chooses the best parameters to fit the training data. In practice, this means that any new points, that were not trained on, will classify horribly.



Maximum A Posteriori (a.k.a. MAP)

Recap: For the coin flip experiment, we introduced prior information to prevent **overfitting**. By analogy:

	train data	likelihood	prior	posterior
coin:	$\mathcal{D} = X$	$p(\mathcal{D} \theta)$	$p(\theta a, b)$	$p(\theta \mathcal{D})$
regr.:	$\mathcal{D} = \{X, Z\}$	$p(Z X, W, \beta)$	$p(W \cdot)$	$p(W X, Z, \cdot)$

Prior: How to find a good one?

A Gaussian prior encourages small parameter values; this discourages the curve of the function from varying wildly.

- ▶ recall (from Eq. (6)) that our likelihood function $p(Z | X, W, \beta)$ is a Gaussian,
- ▶ treat precision $\beta = 1/\sigma^2$ as a known parameter (for now),
- ▶ know that the **conjugate prior** for a Gaussian with known variance is also a Gaussian.

Posterior Distribution

We can actually find a general closed expression for the posterior! Yay, conjugate priors!

Posterior parameter distribution

$$p(W | Z) = \mathcal{N}(W | M_N, \mathbf{S}_N) \quad (15)$$

with

$$\text{Mean} = M_N = \mathbf{S}_N \left(\mathbf{S}_0^{-1} M_0 + \beta \Phi^T Z \right) \quad (16)$$

$$\text{Covariance} = \mathbf{S}_N^{-1} = \mathbf{S}_0^{-1} + \beta \Phi^T \Phi \quad (17)$$

\mathbf{S}_0 = prior's covariance

Properties of the posterior:

- ▶ Since we again have a Gaussian, the MAP solution (i.e the mode) equals the mean: $W_{\text{MAP}} = M_N$.
- ▶ In the limit of an infinitely broad prior, $\mathbf{S}_0 \rightarrow 0$, $W_{\text{MAP}} \rightarrow W_{\text{ML}}$
- ▶ For $N = 0$, i.e. no data points, we get the prior back.

Φ is the design matrix, i.e. a matrix of basis functions $\phi_m(X)$, for each m in M .

Prior distribution example

Let's introduce an isotropic gaussian distribution with zero mean prior over w .

$$p(W | \alpha) = \mathcal{N}(W | 0, \alpha^{-1}\mathbf{I}) = \left(\frac{\alpha}{2\pi}\right)^{\frac{(M+1)}{2}} e^{(-\frac{\alpha}{2} W^T W)} \quad (18)$$

Where, $\alpha =$ precision of the distribution

$M + 1 =$ # of elements in the vector W for an M th order polynomial .

This means that the posterior will have the same form as equation 15, with the following differences:

$$M_n = \beta S_n \Phi^T Z$$
$$S_N^{-1} = \alpha I + \beta \Phi^T \Phi$$

MAP Solution

$$\text{Since } p(W | X, Z, \alpha, \beta) \propto p(Z | X, W, \beta) p(W | \alpha), \quad (19)$$

we have a new error function to which we can apply negative log and minimize.

$$E_{MAP} = -\ln p(W | X, Z, \alpha, \beta)$$

$$E_{MAP} = \frac{\beta}{2} \sum_{n=1}^N [y(X_n, W) - Z_n]^2 - \frac{N}{2} \ln \beta + \frac{N}{2} \ln 2\pi - \ln\left(\frac{\alpha}{2\pi}\right)^{\frac{M+1}{2}} + \frac{\alpha}{2} W^T W$$

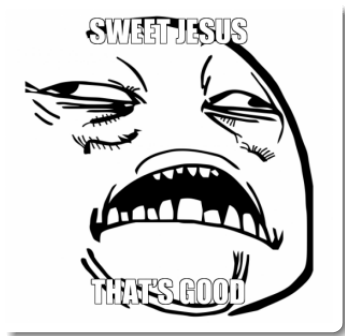
$$\frac{\partial}{\partial W} \frac{\beta}{2} \sum_{n=1}^N [y(X_n, W) - Z_n]^2 - \frac{N}{2} \ln \beta + \frac{N}{2} \ln 2\pi - \ln\left(\frac{\alpha}{2\pi}\right)^{\frac{M+1}{2}} + \frac{\alpha}{2} W^T W = 0$$

$$\Leftrightarrow \frac{\partial}{\partial W} \frac{\beta}{2} \sum_{n=1}^N [y(X_n, W) - Z_n]^2 + \frac{\alpha}{2} W^T W = 0$$

$$\Leftrightarrow \frac{\partial}{\partial W} \underbrace{\frac{1}{2} \sum_{n=1}^N [W^T \phi(X_n) - Z_n]^2}_{\Rightarrow \widetilde{E}_{\mathcal{D}}(W)} + \frac{\lambda}{2} \|W\|_2^2 = 0, \text{ where } \lambda = \frac{\alpha}{\beta}$$

Examples of Closed Form Posteriors

Likelihood	Prior	Posterior Name
Gaussian	Uniform	Least Squares
Gaussian	Gaussian	Ridge
Gaussian	Laplace	Lasso
Laplace	Uniform	Robust Regression
Student	Uniform	Robust Regression



Online Linear Regression

by Grady Jensen

Reading Material:

"Pattern Recognition and Machine Learning" by Bishop[ch. 3.1.3, 3.1.4, 3.3.2]

Further extra reading:

"Machine Learning: A Probabilistic Perspective" by Murphy [ch. 7.6.2 - 7.6.3]

Note: these slides are adapted from slides originally by Christian Osendorfer
Most figures are from C. Bishop: "Pattern Recognition and Machine Learning"

Notation

Symbol	Meaning
--------	---------

s	a scalar number is lowercase and not bold
-----	---

S	A vector is uppercase
-----	-----------------------

\mathbf{S}	a matrix is uppercase and bold
--------------	--------------------------------

$y(X)$	predicted value of inputs X
--------	-------------------------------

z or Z	vector of targets
------------	-------------------

z_i or Z_i	The target of the i 'th example
----------------	-----------------------------------

w_0	a bias term (not to be confused with bias in general)
-------	--

$\phi()$	a basis function
----------	------------------

$E()$	an error function
-------	-------------------

$\tilde{E}()$	regularized error function
---------------	----------------------------

\mathcal{D}	The training data
---------------	-------------------

Φ^\dagger	Moore-Penrose pseudoinverse of Φ
----------------	---------------------------------------

There is not a special symbol for vectors or matrices augmented by the bias term, w_0 . Assume it is always included.

Data Processing



The algorithm for learning maximum likelihood estimates for W assumes that all data points are available at once (*offline* learning, *batch* learning).

What if large data sets are involved so that batch processing of all points at once is infeasible? What if data points arrive over time (sequentially), and possibly should be discarded as soon as possible?

Online Learning: The least-mean-squares (LMS) algorithm

The basic idea is to update W after each newly arriving data point by applying the following technique which is also called **stochastic gradient descent**.

$$W_{(\tau+1)} = W_{\tau} - \eta \nabla E_n \quad (1)$$

E_n represents the error function of the n th data point and we assume that the overall error is given by $E = \sum_n E_n$ and τ is the iteration number.

The parameter η is called the **learning rate** and needs to be chosen carefully in order to achieve convergence of the algorithm.

For the least squares algorithm, its update would look like this:

$$W_{(\tau+1)} = W_{\tau} - \eta (Z_n - W_{\tau}^T \phi(X_n)) \phi(X_n)^T \quad (2)$$

Recursive Least Squares

Determine the *optimal* learning rate for linear regression.

Defining $\mathbf{R}_\tau = \Phi_\tau^T \Phi_\tau$, we get a recursive relationship over time:

$$W_{(\tau+1)} = W_\tau + \mathbf{R}_{(\tau+1)}^{-1} X_{(\tau+1)} (Z_{(\tau+1)} - X_{(\tau+1)}^T W_\tau)$$

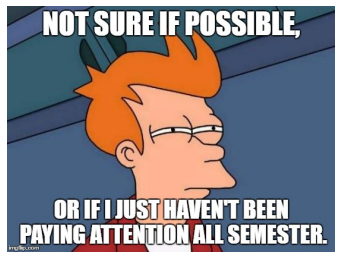
$$\mathbf{R}_{(\tau+1)}^{-1} = \mathbf{R}_\tau^{-1} - \frac{\mathbf{R}_\tau^{-1} X_{(\tau+1)} X_{(\tau+1)}^T \mathbf{R}_\tau^{-1}}{(1 + X_{(\tau+1)}^T \mathbf{R}_\tau^{-1} X_{(\tau+1)})}$$

- ▶ Initial value for $\mathbf{R}^{(0)-1}$ is usually a diagonal matrix with large entries on its diagonal.
- ▶ No matrix inversions are necessary!
- ▶ Versions with weighting/forgetting factors are also possible.

Bayesian Online

What if we don't think of this as an optimization problem, but rather take a Bayesian viewpoint?

Is it possible for us to frame the problem in such a way that takes advantage of Baye's rule to make sequential updates to our approximation to the hidden distribution we are trying to model?



MAP Solution

Remember, if we introduce a gaussian prior:

$$p(W) = \mathcal{N}(W | M_0, \mathbf{S}_0) \quad [M_0: \text{mean}, \mathbf{S}_0: \text{covariance matrix}] \quad (3)$$

Often we don't know much about the prior distribution anyway. For a suitably designed model with independent parameters W , the following prior is usually reasonable (i.e. an isotropic gaussian):

$$p(W | \alpha) = \mathcal{N}(W | M_0 = 0, \mathbf{S}_0 = \alpha^{-1} \mathbf{I}) \quad (4)$$

This results in the posterior:

$$p(W | Z, \alpha, \beta) \propto p(Z | W, \beta)p(W | \alpha)$$

Posterior Distribution for a Simple Prior

If we look at our simplified case as shown in equation 4 from the previous slide:

$$p(W | \alpha) = \mathcal{N}(W | M_0 = 0, \mathbf{S}_0 = \alpha^{-1} \mathbf{I})$$

the posterior parameters simplify to:

$$M_N = \beta \mathbf{S}_N \Phi^T Z \quad (5)$$

$$\mathbf{S}_N^{-1} = \alpha \mathbf{I} + \beta \Phi^T \Phi \quad (6)$$

Predictive Distribution

Usually, we want to know output z for new values of X —the model parameters W are just a means to achieve this. To predict z , evaluate

$$\underbrace{p(z | X, Z, \alpha, \beta)}_{\text{new posterior}} = \int_W \underbrace{p(z | X, W, \beta)}_{\text{likelihood}} \underbrace{p(W | Z, \alpha, \beta)}_{\text{old posterior}} dW \quad (7)$$

(coin flip analogy: $p(x | \mathcal{D}, a, b) = \int_0^1 p(x | \theta) p(\theta | \mathcal{D}, a, b) d\theta$)

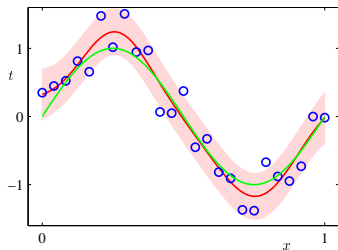
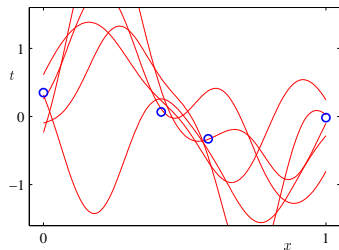
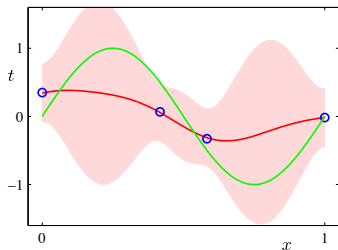
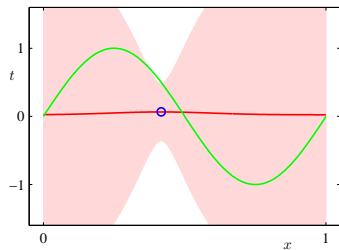
Predictive distribution

$$p(z | X, Z, \alpha, \beta) = \mathcal{N}(z | M_N^T \phi(X), \sigma_N^2(X)) \quad (8)$$

with variance

$$\sigma_N^2(X) = \frac{1}{\beta} + \phi(X)^T \mathbf{S}_N \phi(X). \quad (9)$$

Example of Posterior Predictive



Green: Underlying function, Blue: Observations, Dark-Red: Mode

A simple example

Bayesian regression for the target values

$$z_n = -0.3 + 0.5x_n + \epsilon$$

where ϵ is a Gaussian noise term ($\sigma = 0.2$).

To model this, we set $\phi(x) = \begin{bmatrix} 1 \\ x \end{bmatrix}$ and thus

$$y(x, W) = w_0 + w_1x$$

Sequential Estimation: The demo shows how the posterior's breadth gets smaller as more and more points t are taken into account, and how its mode converges to the optimum (=correct) values of the weights (white cross).

