Linear Regression

Maximum Likelihood Estimation

- Given data(x) predict value (t)predict for new data -> model predictive distribution of p(t/x) so every distribution has a predictive distribution and we choose the appropriate distribution for the pbm.
- Since both weights and i/p are linear there are limitations. So extend by linear combinations of fixed nonlinear functions of i/p y(x,w) = w0+summation of w(j)Phi(x) Phi(x) is basis fn w0 is a bias parameter

Generalization of linear regression- replace each input with a function of that input.

- feature extraction- features are expressed in terms of basis functions Phi(x)
- (ex: robot body - arms egs body head - > basis functions / features)

- polynomial fns are global to remove this limitation in modeling use
- spline fns - different polynomial for each region of i/p space

- gaussian, sigmoid,- space and
- fourier, sinusoidal fnc - specific frequency and infinite space
- wavelets - localized in both space and frequency

- likelihood fn- $p(a/b)= p(b/a)p(a) / p(b)$
- $p(b/a)$ = likelihood function
- max likelihood fn = 'a' is set to value that maximizes $p(b/a)$

- Relation between Least Squares of Error and Maximum Likelihood:
- $T=y(x,w)+$*Epsilon (Epsilon is gaussian random variable with precision* β.
- $p(t|x,w,\beta)=N(t|y(x,w),\beta-1)=\surd(\beta/2\pi)\exp(-\beta/2(t-y$ *square of$(x,w))$

- Use N independent identically distributed observations x1,...xn with corresponding target functions t1...tn
- Jt conditional probability of t/X is
- $p(t|X,w,\beta)=\prod N(ti|w\varphi(xi),inv(\beta))$ <- Likelihood Function
- $\log p(t|w,\beta)=(N/2)\log\beta-(N/2)\log 2\pi-\beta Ed(w)$
- Ed(w) is the sum of squares error function
- $Ed(w)=(\tfrac{1}{2})\sum$square of$(ti-w\varphi(xi))$ $=(\tfrac{1}{2})$square of$\|t-\Phi w\|$
- $\varphi(xi)$ is Design Matrix (N x M matrix)

- Maximizing log likelihood (= minimizing the sum-of-squares error function) w.r.t. W

-> maximum likelihood estimate of parameters w.

Maximum likelihood estimation ->over-fitting if complex models (e.g. polynomial regression models of high order) are fit to datasets of limited size.

Prevent over-fitting - add a regularization term to error function. ->

by a Bayesian approach (or a Gaussian approach)

- Sum of squares error function is -
- Ed(w) =½ Summ (square of {tn − wTϕ(xn)})

- Maximizing likelihood fn under Gaussian noise distribution for a linear model is equivalent to minimizing sum of squares error fn. given Ed(w)

- Gradient of log likelihood fn
- ∇ln p(t|w, β) =Summation(tn − wTϕ(xn)ϕ(xn)T.
- Set gradient to zero
- 0 =Summation (tnϕ(xn)T − wT(Summation(ϕ(xn)ϕ(xn)T)
- Solve for W
- Wml=inv(ΦTΦ) ΦTt <-- Normal equation for least squares pbm (Wml is Weight for Maximum Likelihood)
- Φ is design matrix
- Φ† ≡inv(ΦTΦ) ΦT <-- Moore Penrose pseudo inverse of matrix Φ

- Bias compensates for difference between averages of target values and weighted sum of averages of basis function values

- Sequential Learning (Online Learning)
- Model updates after each data I/p
- Use Stochastic gradient descent
- $w(\tau+1) = w(\tau) - \eta\nabla En$
- For sum of squares error fn:
- $w(\tau+1) = w(\tau) + \eta(tn - w(\tau)T\phi n)\phi n$
- Above is Least Means Square (LMS) algorithm

Regularization

- Error function : Ed(w) + λEw(w)   (Ed - data dependent error)
- Ew is regularization term and λ - control impact of this term
- Simple form of regularization – sum of squares of weight vector terms = Ew(w) =½*transpose(w)*w
- Add E(w) =½ Summation square of{tn − wTϕ(Xn)} to get
- ½ Summation square of {tn − wTϕ(xn)}2 + λ /2 * transpose(w)*w
- Above is the Weight decay / Parameter shrinkage

Regularization -2

- More generalized regularizer:
- }½*Summation square of {tn − wTϕ(xn)} + λ/2 *Summation|Wj| power of q
- If q=1 the function is called Lasso - > regularly used in Deep Learning
- In Lasso when λ is large enough, some of the coefficients of Wj become Zero – > Sparse model -> Basis fn plays no role
- => Avoid overfitting with correct value for regularization
- Generalized to multiple outputs and - solution decouples between different target variables
- Generalize to general Gaussian noise distributions

Bias Variance Decomposition

- Pbms of overfitting and limiting number of basis functions
- Need to determine correct value for λ - (regularization coefficient)

1. Bias Variance trade off
2. Handle overfitting by Bayesian approach (avoid maximum likelihood)

Assuming enough data sets, obtain different prediction fn for each data set. Take average of these functions (squared loss)

Expected squared loss -> E[L] =Integral(square of{y(x) − h(x)} p(x) dx + Integral (square of {h(x) − t}p(x, t) dx dt.

Term 2 represents Noise

Bias Variance decomposition -2

- (Bayesian perspective: posterior distribution over w)
- Average of squared loss over ensemble of data sets , ...
- ED {y(x;D) − h(x)}2 = {ED[y(x;D)] − h(x)}2 + ED{y(x;D) − ED[y(x;D)]}2
- Square(Bias) Variance
- Expected squared difference between y and regression fn h = **squared Bias** (Error of the average prediction over all data sets + **Variance** (sol. varies around average –> sensitivity of y to choice of data)
- Small values of λ -> model become finely tuned to noise on each individual data set -> large variance and low bias
- Large value of λ -> weight parameters to zero -> low variance and large bias.

- Very flexible models <-> low Bias High Variance
- Relatively rigid models <-> High Bias Low Variance
- Need a balance

- Sol: weighted averaging of multiple solutions
- Used in Bayesian approach (Posterior distribution)
- Note: small values of λ -> model is finely tuned to noise on each individual data set -> to large variance
- While large value of λ -> weight parameters to zero -> large bias.

- Bias variance decomposition – uses average over large data sets - which are not available
- Bayesian Linear Regression:
- **Avoid overfitting & determine model complexity**
- Model complexity (# of Basis functions) ∞ size of data set
- Parameter distribution:
- Predictive distribution:
- Equivalent Kernel:

Parameter distribution:

- Prior probability distribution over model parameters 'w'
- (noise precision parameter is considered as a constant)
- P(t/w) is the exponential of a quadratic function of 'w'
- Conjugate prior of likelihood function (posterior and prior in same distribution) -> Gaussian distribution ->
- Normal distribution of (w/m0, S0) m0 – mean, S0 – covariance
- Calculate posterior distribution - ∞ (likelihood fn *prior)
- Evaluate this by completing square in exponential and find normalization co-efficient (using standard result for normalized Gaussian)

- P(w/t) = Normalized fn(w/m_n, S_n)
- Where $M_n = S_n(S_0^{-1}m_0+\beta(Phi)^Tt$
- $S_n^{-1}= S_0^{-1}+\beta(Phi)^TPhi$
- Max posterior Weight vector = $w_{map}=m_N$
- Infinitely broad prior then mean of posterior distribution = w_{ml}
- If N=0 then posterior = prior
- If data points arrive sequentially then posterior = prior for subsequent data points

Consider Zero Mean Isotropic Gaussian with a single precision parameter α
=>

$$p(\mathbf{w}|\alpha) = \mathcal{N}(\mathbf{w}|\mathbf{0}, \alpha^{-1}\mathbf{I})$$

Corresponding posterior distribution is $p(\mathbf{w}|\mathbf{t}) = \mathcal{N}(\mathbf{w}|\mathbf{m}_N, \mathbf{S}_N)$
where

$$\begin{aligned} \mathbf{m}_N &= \beta \mathbf{S}_N \boldsymbol{\Phi}^{\mathrm{T}} \mathbf{t} \\ \mathbf{S}_N^{-1} &= \alpha \mathbf{I} + \beta \boldsymbol{\Phi}^{\mathrm{T}} \boldsymbol{\Phi}. \end{aligned}$$

Gaussian => mode =mean => Maximum Posterior weight vector $\mathbf{w}_{\mathrm{MAP}} = \mathbf{m}_N$

Log of posterior distribution = Sum of log likelihood and log of prior

$$\ln p(\mathbf{w}|\mathbf{t}) = -\frac{\beta}{2} \sum_{n=1}^{N} \{t_n - \mathbf{w}^{\mathrm{T}} \boldsymbol{\phi}(\mathbf{x}_n)\}^2 - \frac{\alpha}{2} \mathbf{w}^{\mathrm{T}} \mathbf{w} + \mathrm{const.}$$

Maximization of posterior distribution wrt w <=>

Linear Models for Regression

$$p(\mathbf{w}|\alpha) = \left[\frac{q}{2}\left(\frac{\alpha}{2}\right)^{1/q}\frac{1}{\Gamma(1/q)}\right]^M \exp\left(-\frac{\alpha}{2}\sum_{j=1}^{M}|w_j|^q\right)$$

Predictive Distribution

$$p(t|\mathbf{t},\alpha,\beta) = \int p(t|\mathbf{w},\beta)p(\mathbf{w}|\mathbf{t},\alpha,\beta)\,\mathrm{d}\mathbf{w}$$

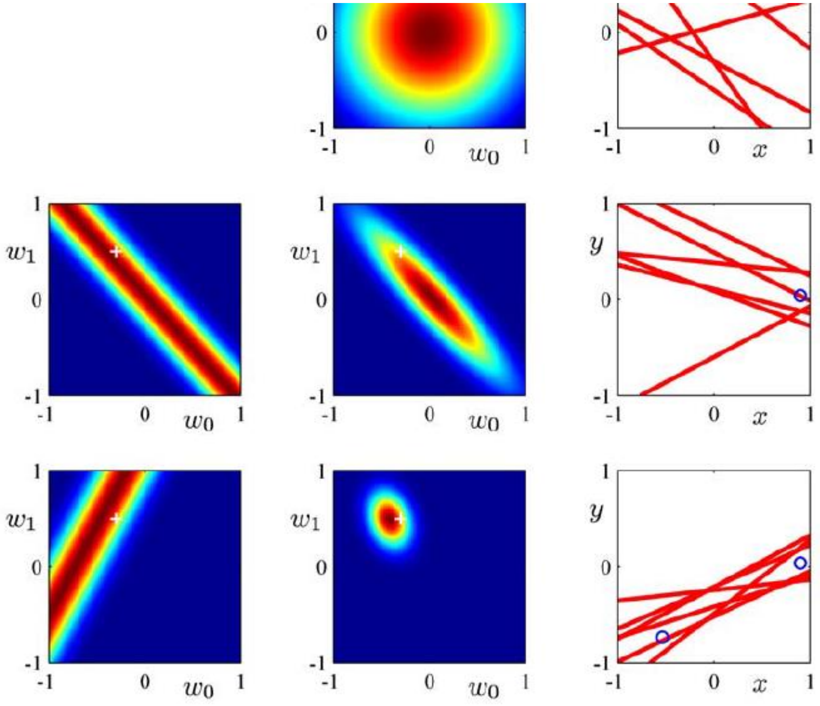
$$p(t|\mathbf{x},\mathbf{t},\alpha,\beta) = \mathcal{N}(t|\mathbf{m}_N^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}),\sigma_N^2(\mathbf{x}))$$

where the variance $\sigma_N^2(\mathbf{x})$ of the predictive distribution is given by

$$\sigma_N^2(\mathbf{x}) = \frac{1}{\beta} + \boldsymbol{\phi}(\mathbf{x})^{\mathrm{T}}\mathbf{S}_N\boldsymbol{\phi}(\mathbf{x}).$$

Illustration: Bayesian Linear regression

- Input variable X and target T . Linear model-> $y(x, \mathrm{w}) = w_0 + w_1 x.$
- Only 2 adaptive parameters =>
- Generate synthetic data using f(x,a) =a_0+a_1x
- Values –> a_0=-0.3 a_1=0.5 by choosing x_n values from Uniform distribution U(x|-1,1) , evaluate f(x_n,a) Add Gaussian Noise with Std. deviation of 0.2 to get t_n
- Objective: Recover values of a_0 and a_1 and study effect of size of data set
- Set Noise β = 25 and α to 2.0



- **Likelihood Prior/ Posterior Data Space**
- 1st row – initial – Prior distribution (6 samples)
- 2nd row – single data point – rt column – data. Lt column – plot of p(t|x,w)
- Sequential nature of Bayesian learning - Current posterior distribution becomes prior when new data is added

Predictive Distribution

- This is the real interest - predict t for new values of x

$$p(t|\mathbf{t},\alpha,\beta) = \int p(t|\mathbf{w},\beta)p(\mathbf{w}|\mathbf{t},\alpha,\beta)\,\mathrm{d}\mathbf{w}$$

- Conditional distribution of t & posterior weight distribution
- 2 Gaussian distributions => Predictive distribution is

$$p(t|\mathbf{x},\mathbf{t},\alpha,\beta) = \mathcal{N}(t|\mathbf{m}_N^{\mathrm{T}}\phi(\mathbf{x}),\sigma_N^2(\mathbf{x}))$$

- Variance of predictive distribution is

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

- 1/β y in Parameter 'w'
- With more sample data posterior distribution gets narrower => Variance becomes lesser. Uncertainty in 'w' become zero and only Noise results in a Variance
- Pbm: Areas away from basis function centers also only noise as the predicted variance value
- Sol: Adopt a Gaussian approach instead of Regression (pl refer book)

Equivalent Kernel

- Kernel methods use

$$\begin{aligned} \mathbf{m}_N &= \beta \mathbf{S}_N \boldsymbol{\Phi}^{\mathrm{T}} \mathbf{t} \\ \mathbf{S}_N^{-1} &= \alpha \mathbf{I} + \beta \boldsymbol{\Phi}^{\mathrm{T}} \boldsymbol{\Phi}. \end{aligned}$$

- => Predictive mean: $$y(\mathbf{x}, \mathbf{w}) = \sum_{j=0}^{M-1} w_j \phi_j(\mathbf{x}) = \mathbf{w}^{\mathrm{T}} \boldsymbol{\phi}(\mathbf{x})$$

- becomes:

$$y(\mathbf{x}, \mathbf{m}_N) = \mathbf{m}_N^{\mathrm{T}} \boldsymbol{\phi}(\mathbf{x}) = \beta \boldsymbol{\phi}(\mathbf{x})^{\mathrm{T}} \mathbf{S}_N \boldsymbol{\Phi}^{\mathrm{T}} \mathbf{t} = \sum_{n=1}^{N} \beta \boldsymbol{\phi}(\mathbf{x})^{\mathrm{T}} \mathbf{S}_N \boldsymbol{\phi}(\mathbf{x}_n) t_n$$

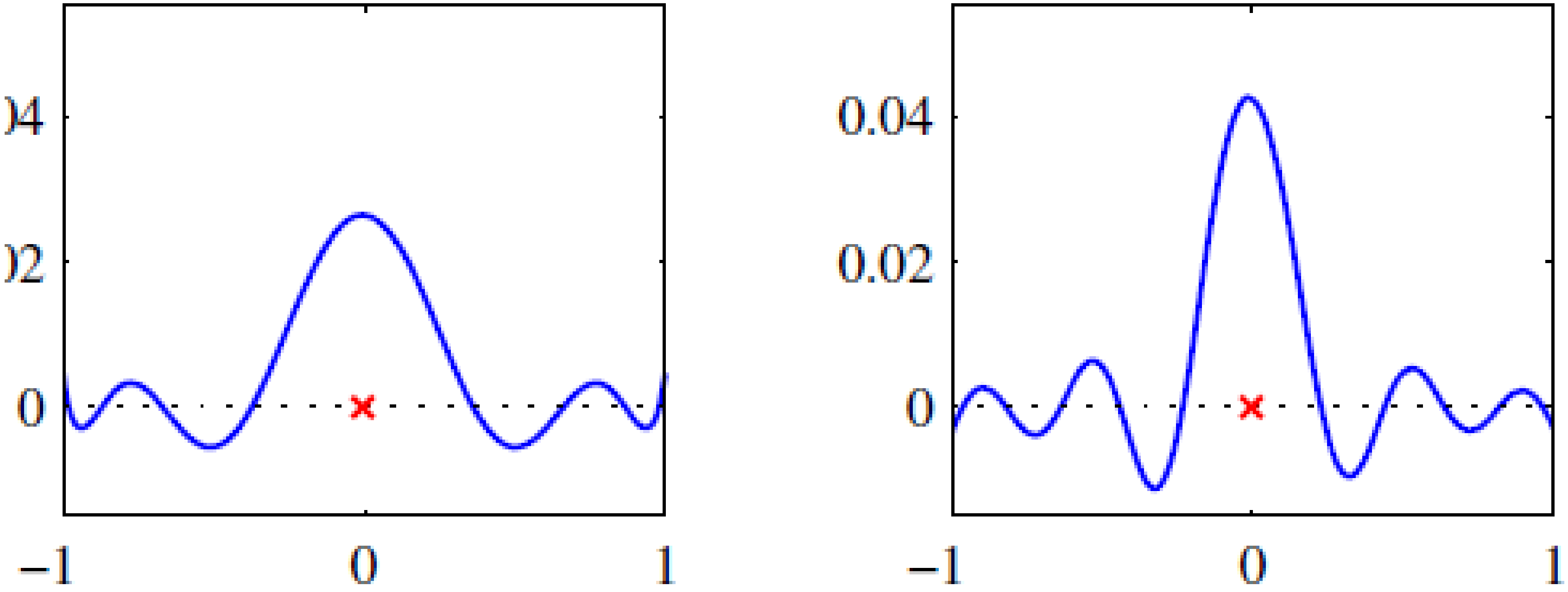
- Predictive distribution at point x is a linear combination of target variables

$$y(\mathbf{x}, \mathbf{m}_N) = \sum_{n=1}^{N} k(\mathbf{x}, \mathbf{x}_n) t_n$$

$$k(\mathbf{x}, \mathbf{x}') = \beta \boldsymbol{\phi}(\mathbf{x})^{\mathrm{T}} \mathbf{S}_N \boldsymbol{\phi}(\mathbf{x}')$$

- Is called "smoother matrix" or "equivalent kernel"
- Linear Smoothers: Linear combinations of training sets

- Visualization of equivalent kernel for Gaussian Basis functions:
- K(x,x') plotted as a function of x for three values => localized around x
- => Mean of predictive distribution function -> gives more weight to data points close to 'x'
-



- Similar inference from covariance between y(x) and y(x')

$$\begin{aligned} \operatorname{cov}[y(\mathbf{x}), y(\mathbf{x}')] &= \operatorname{cov}[\boldsymbol{\phi}(\mathbf{x})^{\mathrm{T}}\mathbf{w}, \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}')] \\ &= \boldsymbol{\phi}(\mathbf{x})^{\mathrm{T}}\mathbf{S}_N\boldsymbol{\phi}(\mathbf{x}') = \beta^{-1}k(\mathbf{x}, \mathbf{x}') \end{aligned}$$

- Predictive means for nearby points is highly correlated
- ---
- Instead of using a set of basis functions , we can define a Localized Kernel and use this to make predictions <= Gaussian Process
- Effective kernel defines the weights and these weights sum to 'One'

$$\sum_{n=1}^{N} k(\mathbf{x}, \mathbf{x}_n) = 1$$

- The summation is = predictive means for target data where t_n=1
- Requirement :
- Basis functions are linearly independent,
- More data points than basis functions
- One Basis point is constant (Bias)
- Then => fit training data exactly

- Also $k(\mathbf{x}, \mathbf{x}') = \beta \phi(\mathbf{x})^{\mathrm{T}} \mathbf{S}_N \phi(\mathbf{x}')$
- Equivalent to inner product w.r.t vector $\psi(\mathbf{x})$ of non linear fn

$$k(\mathbf{x}, \mathbf{z}) = \psi(\mathbf{x})^{\mathrm{T}} \psi(\mathbf{z})$$

- Where

$$\psi(\mathbf{x}) = \beta^{1/2} \mathbf{S}_N^{1/2} \phi(\mathbf{x})$$