Linear Models for Regression using Basis Functions
Overview

• Supervised learning starting with regression
• Goal: predict value of one or more target variables \( t \) given value of \( D \)-dimensional vector \( \mathbf{x} \) of input variables
• When \( t \) is continuous valued we call it regression, if \( t \) has a value consisting of labels it is called classification
• Polynomial is a specific example of curve fitting called linear regression models
Types of Linear Regression Models

• Simplest form of linear regression models:
  – linear functions of input variables

• More useful class of functions:
  – linear combination of non-linear functions of input variables called *basis functions*

• They are linear functions of parameters (which gives them simple analytical properties), yet are nonlinear with respect to input variables
Task at Hand

- Predict value of continuous target variable $t$ given value of $D$-dimensional input variable $x$
  - $t$ can also be a set of variables

- Seen earlier
  - Polynomial
  - Input variable scalar

- This discussion
  - Linear functions of adjustable parameters
  - Specifically linear combinations of nonlinear functions of input variable
Probabilistic Formulation

• Given:
  – Data set of \( n \) observations \( \{x_n\}, n=1,\ldots, N \)
  – Corresponding target values \( \{t_n\} \)

• Goal:
  – Predict value of \( t \) for a new value of \( x \)

• Simplest approach:
  – Construct function \( y(x) \) whose values are the predictions

• Probabilistic Approach:
  – Model predictive distribution \( p(t|x) \)
    • It expresses uncertainty about value of \( t \) for each \( x \)
  – From this conditional distribution
    • Predict \( t \) for any value of \( x \) so as to minimize a loss function
    • Typical loss is squared loss for which the solution is the conditional expectation of \( t \)
Linear Regression Model

- Simplest linear model for regression with $D$ input variables
  \[ y(x, w) = w_0 + w_1 x_1 + \ldots + w_D x_D \]
  \text{where } x = (x_1, \ldots, x_D)^T \text{ are the input variables}

- Called Linear Regression since it is a linear function of
  - parameters $w_0, \ldots, w_D$
  - input variables
Linear Basis Function Models

- Extended by considering nonlinear functions of input variables

\[ y(x, w) = w_0 + \sum_{j=1}^{M-1} w_j \phi_j(x) \]

- where \( \phi_j(x) \) are called Basis functions
- There are now \( M \) parameters instead of \( D \) parameters
- Can be written as

\[ y(x, w) = \sum_{j=0}^{M-1} w_j \phi_j(x) = w^T \varphi(x) \]
Some Basis Functions

- **Linear Basis Function Model**
  \[ y(x, w) = \sum_{j=0}^{M-1} w_j \phi_j(x) = w^T \varphi(x) \]

- **Polynomial**
  - In polynomial regression seen earlier, there is a single variable \( x \) and \( \phi_j(x) = x^j \) with degree \( M \) polynomial
  - Disadvantage
    - Global

- **Gaussian**
  \[ \phi_j = \exp \left( \frac{(x - \mu_j)^2}{2s^2} \right) \]

- **Sigmoid**
  \[ \phi_j = \sigma \left( \frac{x - \mu_j}{s} \right) \text{ where } \sigma(a) = \frac{1}{1 + \exp(-a)} \]

- **Tan h**
  \[ \tanh(a) = 2\sigma(a) - 1 \]
Other Basis Functions

• Fourier
  – Expansion in sinusoidal functions
  – Infinite spatial extent

• Signal Processing
  – Functions localized in time and frequency
  – Called wavelets
    • Useful for lattices such as images and time series

• Further discussion independent of choice of basis
Maximum Likelihood Formulation

- Target variable $t$ given by deterministic function $y(x,w)$ with additive Gaussian noise:
  $$t = y(x,w) + \varepsilon$$
  $\varepsilon$ is zero-mean Gaussian with precision $\beta$
- Thus distribution of $t$ is normal:
  $$p(t|x,w,\beta) = \mathcal{N}(t|y(x,w), \beta^{-1})$$
  \begin{align*}
  \text{mean} & \quad \text{variance}
  \end{align*}
Likelihood Function

- Data set $X = \{ x_1, \ldots, x_N \}$ with values $t = \{ t_1, \ldots, t_N \}$
- Likelihood

\[
p(t \mid X, w, \beta) = \prod_{n=1}^{N} N(t_n \mid w^T \phi(x_n), \beta^{-1})
\]

- Log-likelihood

\[
\ln p(t \mid X, w, \beta) = \prod_{n=1}^{N} \ln N(t_n \mid w^T \phi(x_n), \beta^{-1})
\]

- Where

\[
E_D(w) = \frac{1}{2} \sum_{n=1}^{N} \left( t_n - w^T \phi(x_n) \right)^2
\]

- Called Sum-of-squares Error Function

- Maximizing Likelihood with Gaussian noise is equivalent to minimizing $E_D(w)$
Maximum Likelihood for weight parameter \( w \)

- Gradient of log-likelihood wrt \( w \)

\[
\nabla \ln p(t \mid X, w, \beta) = \sum_{n=1}^{N} \left\{ t_n - w^T \phi(x_n) \right\} \phi(x_n)^T
\]

- Setting to zero and solving for \( w \)

\[
w_{ML} = \Phi^+ t
\]

- Where \( \Phi^+ = (\Phi^T \Phi)^{-1} \Phi^T \) is the Moore-Penrose pseudo inverse of \( N \times M \) Design Matrix

\[
\Phi = \begin{pmatrix}
\phi_0(x_1) & \phi_1(x_1) & \ldots & \phi_{M-1}(x_1) \\
\phi_0(x_2) \\
\phi_0(x_N) & \phi_{M-1}(x_N)
\end{pmatrix}
\]
Maximum Likelihood for precision $\beta$

- Similarly gradient wrt $\beta$ gives

$$\frac{1}{\beta_{ML}} = \frac{1}{N} \sum_{n=1}^{N} \{ t_n - w_{ML}^T \phi(x_n) \}^2$$

Residual variance of the target values around the regression function
Geometry of Least Squares Solution

- \( N \)-dimensional space (target values of \( N \) data points)
- Axes given by \( t_n \)
- Each basis function \( \phi_j(x_n) \), corresponding to \( j^{th} \) column of \( \Phi \), is represented in this space
- If \( M < N \) then \( \phi_j(x_n) \) are in a subspace \( S \) of dimensionality \( M \)
- Solution \( y \) is choice of \( y \) that lies in subspace \( S \) that is closest to \( t \)
  - Corresponds to orthogonal projection of \( t \) onto \( S \)
- When two or more basis functions are collinear, \( \Phi^T \Phi \) is singular
  - Singular Value Decomposition is used
Sequential Learning

• Batch techniques for m.l.e. can be computationally expensive for large data sets

• If error function is a sum over $n$ data points

\[ E = \sum_n E_n \]

then update parameter vector $w$ using

\[ w^{(\tau+1)} = w^{(\tau)} - \eta \nabla E_n \]

• For Sum-of-squares Error Function

\[ w^{(\tau+1)} = w^{(\tau)} + \eta (t_n - w^{(\tau)} \phi(x_n)) \phi(x_n) \]

• Known as Least Mean Squares Algorithm
Regularized Least Squares

- Adding regularization term to error controls over-fitting

\[ E_D(w) + \lambda E_W(w) \]

- Where \( \lambda \) is the regularization coefficient that controls importance of data-dependent error \( E_D(w) \) and the regularization term \( E_W(w) \)
- Simple form of regularizer

\[ E_W(w) = \frac{1}{2} w^T w \]

- Total error function becomes

\[ \frac{1}{2} \sum_{n=1}^{N} \left\{ t_n - w^T \phi(x_n) \right\}^2 + \frac{\lambda}{2} w^T w \]

- Called weight decay because in sequential learning weight values decay towards zero unless supported by data
More general regularizer

- Regularized Error

\[ \frac{1}{2} \sum_{n=1}^{N} \left\{ t_n - w^T \phi(x_n) \right\}^2 + \frac{\lambda}{2} \sum_{j=1}^{M} |w_j|^q \]

- Where \( q=2 \) corresponds to the quadratic regularizer
- \( q=1 \) is known as lasso
- Regularization allows complex models to be trained on small data sets without severe overfitting
- Contours of regularization term: \(|w_j|^q\)
Height of Emperor of China

True height is 200 (measured in cm, about 6’6”).
Poll a random American: ask “How tall is the emperor?”
We want to determine how wrong they are, on average

- **Scenario 1**
  - Every American believes it is 180
  - The answer is always 180
  - The error is always -20
  - Average squared error is 400
  - Average error is 20

- **Scenario 2**
  - Americans have normally distributed beliefs with mean 180 and standard deviation 10
  - Poll two Americans. One says 190 and other 170
  - Bias Errors are -10 and -30
    - Average bias error is -20
  - Squared errors: 100 and 900
    - Ave squared error: 500
  - $500 = 400 + 100$

- **Scenario 3**
  - Americans have normally distributed beliefs with mean 180 and standard deviation 20
  - Poll two: One says 200 and other 160
  - Errors: 0 and -40
    - Ave error is -20
  - Squared errors: 0 and 1600
    - Ave squared error: 800
  - $800 = 400 + 400$

Average Squared Error (500) = Square of bias (-20) + variance (100)
Total squared error = square of bias error + variance
Bias and Variance Formulation

- \( y(x) \): estimate of the value of \( t \) for input \( x \)
- \( h(x) \): optimal prediction
  \[
  h(x) = E[t \mid x] = \int t p(t \mid x) dt
  \]
- If we assume loss function \( L(t, y(x)) = (y(x) - t)^2 \)
- \( E[L] \) can be written as
  expected loss = (bias)\(^2 + \) variance + noise
- where
  \[
  (\text{bias})^2 = \int \{E_{D}[y(x; D)] - h(x)\}^2 p(x) dx
  \]
  variance = \[
  \int E_{D}\left[\{y(x; D) - E_{D}[y(x; D)]\}^2\right] p(x) dx
  \]
  noise = \[
  \int \{h(x) - t\}^2 p(x, t) dx dt
  \]
Dependence of Bias-Variance on Model Complexity

- \( h(x) = \sin(2\pi x) \)
- Regularization parameter \( \lambda \)
- \( L = 100 \) data sets
- Each has \( N = 25 \) data points
- 24 Gaussian Basis functions
  - No of parameters \( M = 25 \)

Result of averaging multiple solutions with complex model gives good fit

Weighted averaging of multiple solutions is at heart of Bayesian approach: not wrt multiple data sets but wrt posterior distribution of parameters
Bayesian Linear Regression

• Prior probability distribution over model parameters $w$

• Assume precision $\beta$ is known

• Since Likelihood function $p(t|w)$ with Gaussian noise has an exponential form
  
  – Conjugate prior is given by Gaussian
  
  $p(w) = N(w|m_0, S_0)$ with mean $m_0$ and covariance $S_0$
Posterior Distribution of Parameters

- Given by product of likelihood function and prior
  \[ p(w|D) = p(D|w)p(w)/p(D) \]
- Due to choice of conjugate Gaussian prior, posterior is also Gaussian
- Posterior can be written directly in the form
  \[ p(w|t) = N(w|m_N, S_N) \] where
  \[ m_N = S_N(S_0^{-1}m_0 + \beta \Phi^T t), \quad S_N^{-1} = S_0^{-1} + \beta \Phi^T \Phi \]
Bayesian Linear Regression Example

- Straight line fitting
- Single input variable $x$
- Single target variable $t$
- Linear model $y(x, w) = w_0 + w_1x$
  - $t$ and $y$ used interchangeably here
- Since there are only two parameters
  - We can plot prior and posterior distributions in parameter space
Sequential Bayesian Learning

- Synthetic data generated from 
  \( f(x,a) = a_0 + a_1 x \) with parameter values 
  \( a_0 = -0.3 \) and \( a_1 = 0.5 \)

- By first choosing \( x_n \) from \( U(x|-1,1) \), then evaluating \( f(x_n,a) \) and then adding Gaussian noise with std dev 0.2 to obtain target values \( t_n \)

- Goal is to recover values of \( a_0 \) and \( a_1 \)

With infinite points posterior is a delta function centered at true parameters (white cross)
Predictive Distribution

• We are usually not interested in the value of \( w \) itself
• But predicting \( t \) for new values of \( x \)
• We evaluate the predictive distribution

\[
p(t \mid x, t, \alpha, \beta) = N(t \mid m_N^T \phi(x), \sigma_N^2(x))
\]

where \( \sigma_N^2(x) = \frac{1}{\beta} + \phi(x)^T S_N \phi(x) \)

Noise in data
Uncertainty associated with parameters \( w \)