CAP 5610: Machine Learning

Lecture 5:
Regression:
Logistic and Linear

Instructor: Dr. Gita Sukthankar
Email: gitars@eecs.ucf.edu

Reference Materials From: GuoJun Qi

Bayes Classifier: A Generative model

- Model the posterior distribution P(Y|X)
  - Estimate class-conditional distribution P(X|Y) for each Y
  - Estimate prior distribution P(Y)
  - Predict the target value Y by Maximum A Posteriori (MAP)

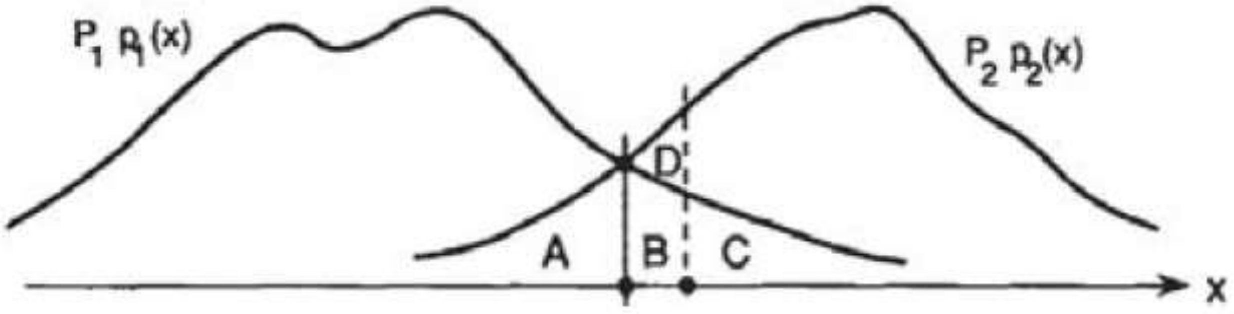
$$Y^{*} = \mathrm{argmax}_{Y} P(Y|X) = \mathrm{argmax}_{Y} P(X|Y)P(Y)$$

- It is a generative model, because
  - Given each class Y, we can draw (generate) a sample from P(Y|X)

Bayes Classifier is the optimal classifier

- Any classifier can never yield smaller prediction error than Bayes classifier

$$error_{true}(h_{Bayes})) \leq error_{true}(h), \ \forall h(\mathbf{x})$$



Drawback of a generative model

- It is an indirect method
  - We need to estimate P(Y) and P(X|Y) from the training set at first
  - For a classification problem, we do not need any generative information contained in P(X|Y)
  - Unnecessary assumptions are usually needed to estimate P(X|Y)
    - Naive Bayes – independence between features conditional on the label Y
- How about directly learn P(Y|X)?

Discriminative model

- Directly model the posterior distribution P(Y|X)
  - Assume some functional forms for P(Y|X), e.g., a linear model
  - Estimate the parameter of the proposed model directly from training set,
    - Option 1: Maximum likelihood estimation (MLE)
    - Option 2: Maximum A Posteriori (MAP)

Hints from generative model

- Suppose
  - X is a vector of continuous features: $X=(X_1, \ldots, X_N)$
  - Y is a binary random variable {0,1}
  - Naive Bayesian classifier: X_i's are independent given Y
  - P(Y) is a Bernoulli distribution
- Find the form for P(Y=1|X), by Bayes rule

$$P(Y=1|X) = \frac{P(Y=1)P(X|Y=1)}{P(Y=1)P(X|Y=1) + P(Y=0)P(X|Y=0)}$$

Find P(Y=1|X)

$$P(Y=1|X)=\frac{P(Y=1)P(X|Y=1)}{P(Y=1)P(X|Y=1)+P(Y=0)P(X|Y=0)}$$

Bayes rule

$$=\frac{1}{1+\frac{P(Y=0)P(X|Y=0)}{P(Y=1)P(X|Y=1)}}$$

$$=\frac{1}{1+\exp(\ln\frac{P(Y=0)P(X|Y=0)}{P(Y=1)P(X|Y=1)})}$$

exp and $\ln$ will cancel out.

$$=\frac{1}{1+\exp(\ln\frac{P(\mathrm{Y}=0)}{P(Y=1)}+\sum_{i=1}^{N}\ln\frac{P(X_i\,|\,Y=0)}{P(X_i\,|\,Y=1)})}$$

Assume $P(X_i|Y)$ are Gaussian with the same variance but different mean

$$=\frac{1}{1+\exp(\ln\frac{P(\mathrm{Y}=0)}{P(Y=1)}+\sum_{i=1}^{N}\ln\frac{\mathrm{N}(X_i;\mu_{i0},\sigma_i^2)}{\mathrm{N}(X_i;\mu_{i1},\sigma_i^2)})}=\frac{1}{1+\exp(\ln\frac{P(\mathrm{Y}=0)}{P(Y=1)}+\sum_{i=1}^{N}\left(\frac{\mu_{i0}-\mu_{i1}}{\sigma_i^2}X_i+\frac{\mu_{i1}^2-\mu_{i0}^2}{2\sigma_i^2}\right))}$$

Find P(Y|X=1)

- Logistic function

$$P(Y=1|X)=\frac{1}{1+\exp(\sum_{i=1}^{N}\underbrace{\frac{\mu_{i0}-\mu_{i1}}{\sigma_i^2}}_{\omega_i}X_i+\underbrace{\frac{\mu_{i1}^2-\mu_{i0}^2}{2\sigma_i^2}+\ln\frac{P(\mathrm{Y}=0)}{P(Y=1)}}_{\omega_0})}$$

$$=\frac{1}{1+\exp(\sum_{i=1}^{N}\omega_i X_i+\omega_0)}$$

Linear classifier

- When Y=0

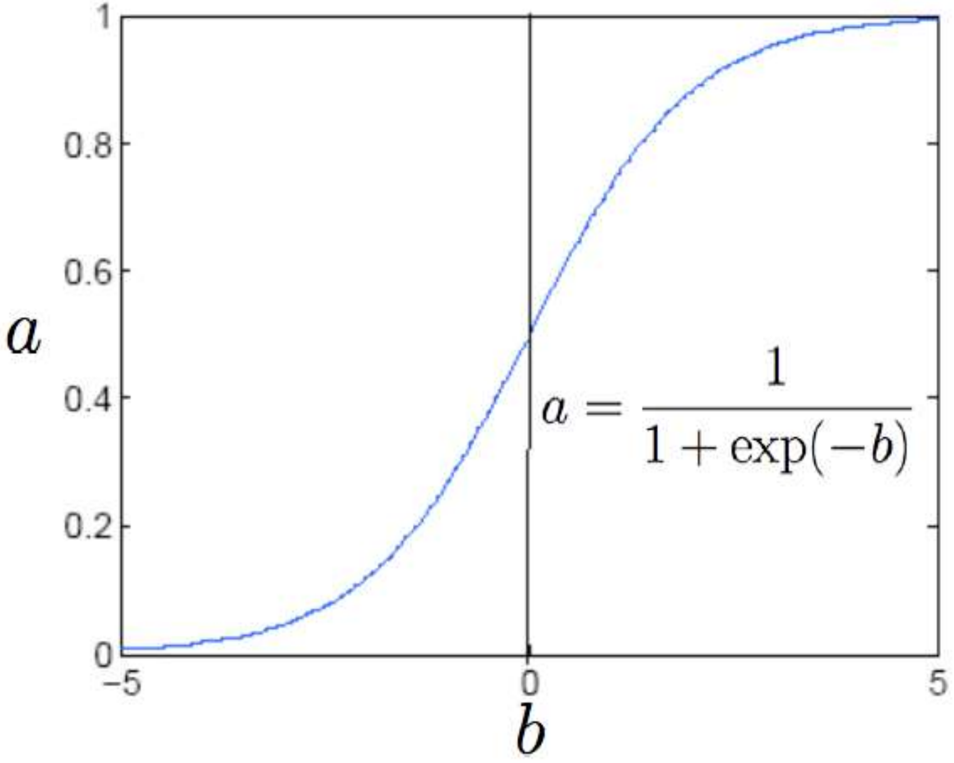
$$P(Y=0|X)=1-P(Y=1|X)=\frac{\exp(\sum_{i=1}^{N}\omega_i X_i+\omega_0)}{1+\exp(\sum_{i=1}^{N}\omega_i X_i+\omega_0)}$$

- Log likelihood ratio: linear in X and w

$$\log\frac{P(Y=1|X)}{P(Y=0|X)}=\log\frac{1}{\exp(\sum_{i=1}^{N}\omega_i X_i+\omega_0)}=-\sum_{i=1}^{N}\ \omega_i X_i+\omega_0$$

Logistic function

- Smooth in b: differentiable and continuously differentiable
- The logarithm of logistic function is concave.



Extend to multiple classes

- Y={0,1,…,C}, e.g., C=10 in MNIST dataset
- Assume $$P(Y=c\mid X)\propto \exp(\sum_{i=1}^{N} w_{ci}X_i + w_{c0})$$
- Normalizing P(Y|X) such that the summation over C classes is one.
  - Softmax: the maximum exponential function (above) for class c will have the largest posterior probability

$$P(Y=c\mid X)=\frac{\exp(\sum_{i=1}^{N} w_{ci}X_i + w_{c0})}{\sum_{c'=1}^{C}\exp(\sum_{i=1}^{N} w_{c'i}X_i + w_{c'0})}$$

- Binary logistic regression is a special case when C=2, and w_{1i}=0

Learning logistic regression model

- Given a set of training example {($X^{(m)}$,$Y^{(m)}$|m=1,…, M}, MLE corresponds to

$$L(\mathrm{w}) = \sum_{m=1}^{M} \log P(\mathrm{Y}^{(m)} \mid \mathrm{X}^{(m)}, \mathrm{w})$$

$$= \sum_{m=1}^{M} \mathrm{Y}^{(m)} \log P(\mathrm{Y}^{(m)} = 1 \mid \mathrm{X}^{(m)}, \mathrm{w}) + (1 - \mathrm{Y}^{(m)}) \log P(\mathrm{Y}^{(m)} = 0 \mid \mathrm{X}^{(m)}, \mathrm{w})$$

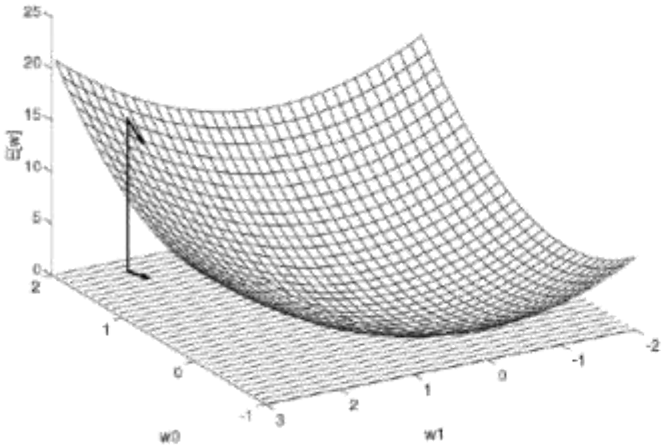
$$= \sum_{m=1}^{M} \left\{ \mathrm{Y}^{(m)} \log \frac{P(\mathrm{Y}^{(m)} = 1 \mid \mathrm{X}^{(m)}, \mathrm{w})}{P(\mathrm{Y}^{(m)} = 0 \mid \mathrm{X}^{(m)}, \mathrm{w})} + \log P(\mathrm{Y}^{(m)} = 0 \mid \mathrm{X}^{(m)}, \mathrm{w}) \right\}$$

$$= -\sum_{m=1}^{M} \mathrm{Y}^{(m)} \left(\sum_{i=1}^{N} w_i X_i^{(m)} + w_0 \right) + \log \left\{ 1 + \exp(-\sum_{i=1}^{N} w_i X_i^{(m)} - w_0) \right\}$$

LR Optimization problem

- The optimal solution to w is

$$\mathrm{w}^* = \arg\max{}_w - \sum_{m=1}^{M} \mathrm{Y}^{(m)} \left(\sum_{i=1}^{N} w_i X_i^{(m)} + w_0 \right) + \log \left\{ 1 + \exp\left(-\sum_{i=1}^{N} w_i X_i^{(m)} - w_0\right) \right\}$$



- The objective function is concave.
  - Logistic function is concave in its logarithmic form.
  - It has global optimal point, avoiding local optimum.
  - Unfortunately, LR optimization problem does not have closed-form solution.

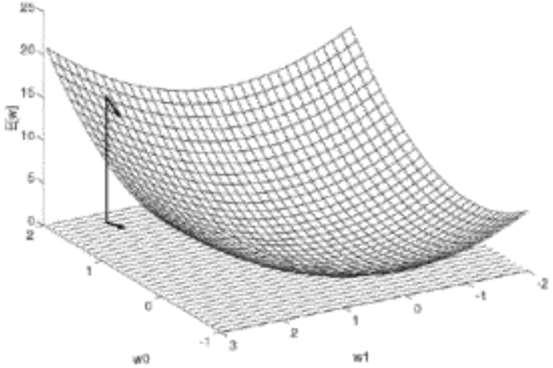
Gradient Ascent Method

- Gradient descent method is an iterative algorithm
  - hill climbing method to find the peak point of a "mountain"
  - At each point, compute its gradient

$$\nabla L = \left[\frac{\partial L}{\partial w_0}, \frac{\partial L}{\partial w_1}, \ldots, \frac{\partial L}{\partial w_N} \right]$$

  - Gradient is a vector that points to the steepest direction climbing up the mountain.
  - At each point, w is updated so it moves a size of step λ in the gradient direction

$$w \leftarrow w + \lambda \nabla L(w)$$



Gradient Ascent Method

- LR Log likelihood:

$$L(W) = -\sum_{m=1}^{M} \mathrm{Y}^{(m)} \left(\sum_{i=1}^{N} w_i X_i^{(m)} + w_0 \right) + \log \left\{ 1 + \exp(-\sum_{i=1}^{N} w_i X_i^{(m)} - w_0) \right\}$$

- Derivative to each weight w_i

$$\frac{\partial L}{w_0} = -\sum_{m=1}^{M} \mathrm{Y}^{(m)} - P(Y=1 | X^{(m)})$$

$$\frac{\partial L}{w_i} = -\sum_{m=1}^{M} \mathrm{Y}^{(m)} X_i^{(m)} - X_i^{(m)} P(Y=1 | X^{(m)})$$

- Update rule

$$w_0 \leftarrow w_0 - \lambda(\sum_{m=1}^{M} \mathrm{Y}^{(m)} + P(Y=1 | X^{(m)}))$$

$$w_i \leftarrow w_i - \lambda(\sum_{m=1}^{M} \mathrm{Y}^{(m)} X_i^{(m)} + X_i^{(m)} P(Y=1 | X^{(m)}))$$

Training algorithm

- Input: a set of training example {(X$^{(m)}$,Y$^{(m)}$|m=1,…, M}
- Initialize w=[w_0, w_1, …, w_N]
- Repeat

$$w_0 \leftarrow w_0 - \lambda\left(\sum_{m=1}^{M} \mathrm{Y}^{(m)} + P(Y=1 \mid X^{(m)})\right)$$

$$w_i \leftarrow w_i - \lambda\left(\sum_{m=1}^{M} \mathrm{Y}^{(m)} X_i^{(m)} + X_i^{(m)} P(Y=1 \mid X^{(m)})\right)$$

- (optional) check if the loglikelihood increases after each update. If not, shrink the step size λ
- Until convergence

Training algorithm

- Input: a set of training example $\{(X^{(m)},Y^{(m)} | m=1,\ldots, M\}$
- Initialize $w=[w_0, w_1, \ldots, w_N]$
- Repeat

$$w_0 \leftarrow w_0 - \lambda(\sum_{m=1}^{M} \mathrm{Y}^{(m)} + P(Y=1 | X^{(m)}))$$

$$w_i \leftarrow w_i - \lambda(\sum_{m=1}^{M} \mathrm{Y}^{(m)} X_i^{(m)} + X_i^{(m)} P(Y=1 | X^{(m)}))$$

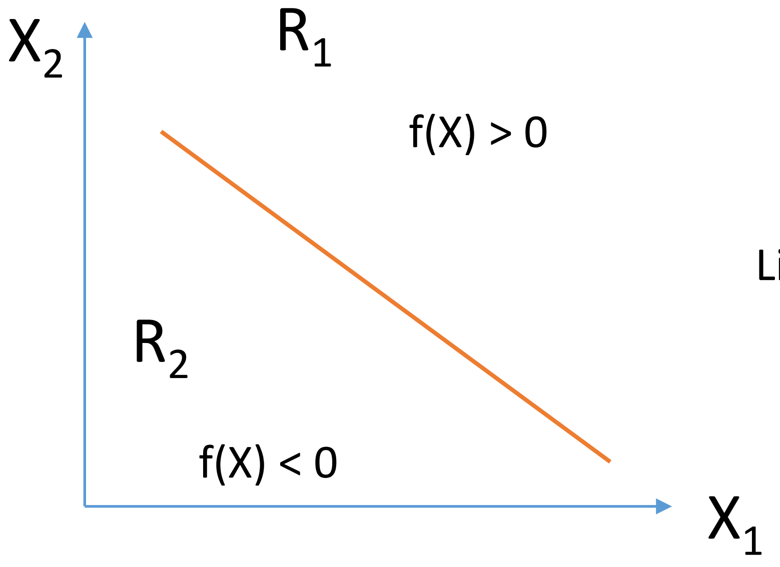
- (optional) check if the log likelihood increases after each update. If not, shrink the step size λ
- Until convergence

Test algorithm

- Computing the log likelihood ratio

$$\mathrm{f(X)} = \log \frac{P(Y=1|X)}{P(Y=0|X)} = \log \frac{1}{\exp(\sum_{i=1}^{N} \omega_i X_i + \omega_0)} = -\sum_{i=1}^{N} \omega_i X_i + \omega_0$$

- If >0, X is positive (Y=1)
- If <0, X is negative (Y=0)



Linear decision bounda

Stochastic Gradient Ascent Method

- Making the learning algorithm scalable to big data
- Typical gradient ascent method
  - Each update needs to go over all M examples to compute the derivatives

$$\frac{\partial L}{w_0} = -\sum_{m=1}^{M} \mathrm{Y}^{(m)} - P(Y = 1 | X^{(m)}) \qquad \frac{\partial L}{w_i} = -\sum_{m=1}^{M} \mathrm{Y}^{(m)} X_i^{(m)} - X_i^{(m)} P(Y = 1 | X^{(m)})$$

  - It will save us a huge amount of computations if only a small portion of examples are used

Stochastic Gradient Ascent Method

- A random subset Ω of examples are drawn from training set, which are used to compute derivatives

$$\frac{\partial L}{w_0} = -\sum_{m=1}^{M} \mathrm{Y}^{(m)} - P(Y=1|X^{(m)}) \qquad \frac{\partial L}{w_i} = -\sum_{m=1}^{M} \mathrm{Y}^{(m)} X_i^{(m)} - X_i^{(m)} P(Y=1|X^{(m)})$$

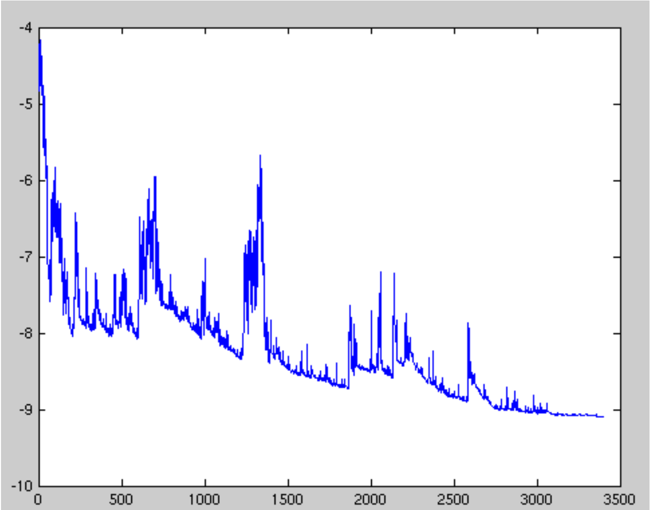
$$\Downarrow \qquad\qquad \Downarrow$$

$$\frac{\partial L}{w_0} = -\sum_{m\in\Omega} \mathrm{Y}^{(m)} - P(Y=1|X^{(m)}) \qquad \frac{\partial L}{w_i} = -\sum_{m\in\Omega} \mathrm{Y}^{(m)} X_i^{(m)} - X_i^{(m)} P(Y=1|X^{(m)})$$

- Especially, only one example is drawn from training set to compute the derivative

Stochastic Gradient Ascent Method

- Pros: Each round of update only needs very small number of examples
  - Time complexity: More rapid update
  - Space complexity: Only upload the examples drawn in each round to the memory
- Cons: Cannot guarantee convergence
- Practice: very efficient



Option 2: Maximum A Posteriori (MAP)

- Add a prior distribution on w
  - P(w) = N(0, σ^2 I)
- Posterior distribution on w can be given by

$$\log P(w \mid D) \propto \log P(w) + \sum_{m=1}^{M} \log P(\mathrm{Y}^{(m)} \mid \mathrm{X}^{(m)}, \mathrm{w})$$
$$= \log P(w) + L(w)$$

- The optimal solution

$$\mathrm{w}^* = \mathrm{argmax}_w \mathrm{L(w)} + \log P(w) = \mathrm{argmax}_w \mathrm{L(w)} - \frac{1}{2\sigma^2} \| w \|_2^2$$

L_2 regularizer

- L_2 Regularized loglikelihood objective function

$$\mathrm{w}^* = \mathrm{argmax}_w \mathrm{L(w)} + \log P(w) = \mathrm{argmax}_w \mathrm{L(w)} - \frac{1}{2\sigma^2} \| w \|_2^2$$

- Role of a regularizer
  - Reduce the overfitting, by imposing a prior knowledge on w
  - keeping the weights closer to zero

Philosophy behind L_2 regularizer

- Why should we prefer a zero weight ***w*** in LR?
  - Maximum entropy principle: do NOT make a **bias** towards one of two outcomes unless you have enough information (least bias).
  - A coin flipping: how do you predict the outcome if you do not know any attributes about the coin?

- When ***w*** = 0, P(Y=1|X)=P(Y=0|X)=0.5 (equal probability)

$$P(Y=1|X)=\frac{1}{1+\exp(\sum_{i=1}^{N}\omega_i X_i+\omega_0)}$$

Comparison between Naive Bayes and LR

- Number of model parameters
  - Naive Bayes: for each attribute, two parameters (mean and variance), for two classes; two class prior distribution – 4N+2
  - LR: N weight coefficients
  - More parameters usually mean more training examples to estimate them.
- Assumptions:
  - Naive Bayes: conditional independence given the class label
  - LR: no such independence assumption (why?)
  - With weaker assumption, LR may outperform Naive Bayes if the assumption does not hold.

Summary: Logistic Regression

- Derive LR from Naive Bayes with Gaussian class-conditional distribution
- MLE: Estimate the coefficient weights in LR
  - Gradient ascent method
  - Stochastic gradient ascent method
- MAP: add a prior on w
  - A L_2 regularizer

Linear vs. Logistic Regression

- Logistic regression is used when the dependent variable is categorical to establish the probability of the event.
  - Logistic loss function causes errors to be penalized to an asymptotic constant.
- Linear regression is used when the dependent variable is continuous.
  - Solved by minimizing least squares error so large errors are penalized quadratically.
- Reading: PRML Chapter 3

What's a regression problem?

- Given an input vector *X*, predict the target value associated with *X*.
  - Input is size of a house, target variable is its price.
  - Predict the stock prices based their history performances.
  - Predict the person's age from the face image.

Regression

- Learn a predictor f: $X \mapsto Y$ where X is a feature vector and Y is the target variable
- Outline (compared with discrete case)
  - Design an optimal predictor (Bayes classifier)
  - Linear model based on Gaussian assumption
  - Estimate the parameters of linear model
    - MLE
    - MAP
  - Bias-variance decomposition

Optimal regression model

- Given a joint distribution over (X,Y), what's the optimal predictor?
  - Minimizing the expected least-square loss wrt P(X,Y)

$$E|Y-f(X)|^2=\iint_{Y,X}|Y-f(X)|^2P(X,Y)dXdY$$

  - Compared with classification problem, where we minimize the expected prediction errors.

Optimal regression model

- Variational optimization method that minimizes wrt a function *f*(X) rather than a single variable (Appendix D in PRML)

$$E|Y - f(X)|^2 = \iint_{Y,X} |Y - f(X)|^2 P(X,Y)\,dX\,dY$$

- Set the differential wrt *f*(X) to 0

$$\frac{\partial E|Y - f(X)|^2}{\partial f(X)} = 2\int (f(X) - Y)\, P(X,Y)\,dY = 0$$

$$f(X)P(X) - P(X)E(Y|X) = 0$$

$$f(X) = E(Y|X) = \int Y P(Y|X)\, dY$$

Optimal regression model

- Optimal regression model is the expectation of Y conditional on the input vector X

$$f(X) = E(Y|X) = \int YP(Y|X)\, dY$$

- Compared with optimal classification model (Bayes Classifier)

$$Y = \mathrm{argmax}_Y\, P(Y|X)$$

Practical Issue

- We do not know the true posterior distribution P($Y|X$), just as in the classification problem

- How about using multivariate Gaussian distribution to estimate the joint distribution P(X,Y)?

The optimal Gaussian regression model

- Joint distribution

$$P(X, Y) = N(\boldsymbol{m}_X, m_Y; C)$$

- Conditional distribution of Y given X
- An important property: the conditional distribution P(Y|X) from a multivariate Gaussian distribution is still Gaussian.

$$P(Y|X) = \mathrm{N}(m_Y + C_{YX}C_{XX}^{-1}(X - mX), C_{Y|X})$$

- Optimal predictor

$$\mathrm{E}(Y|X) = mY + C_{YX}C_{XX}^{-1}(X - mX)$$

  - This is a linear predictor, although we do not know these parameters.
  - That's why we often assume data are generated by Gaussian distribution, because the regression model is very simple.

Drawbacks

- It is an indirect method
  - We have to estimate the joint distribution before finding the optimal predictor (then computing expected *Y* conditional on *X*).
  - Too many parameters to estimate
    - For N attributes in X and target variable *Y*, there are (N+1)(N+2)/2 parameters in covariance matrix, along with N+1 parameters for the mean of X and Y.
  - We do not care about the covariance and mean of the attributes;
    - We only need to know the covariance **between** X and Y (N parameters) and the mean of Y (1 parameter): N+1 parameters totally that are of true interest to the regression problem.
    - A direct method, please!

Discriminative method

- Directly estimate P(Y|X) with a specified form, e.g., Gaussian again.

$$P(Y|X) = N(f(X), \sigma^2)$$

- Assume *f*(X) is linear in X

$$f(X) = \sum_{i=1}^{N} W_i X_i + W_0 = W^T X$$

where we denote by *X* an extended vector with an additional entry

$X=[1, X_1, X_2, \ldots, X_N]$, $W=[W_0, W_1, W_2, \ldots, W_N]$.

- It is discriminative since we only estimate the posterior distribution
  - No generating model will be estimated from which we can draw X given a Y.

Two Methods

- Option 1: MLE (Maximum Likelihood Estimation)
- Option 2: MAP (Maximum A Posteriori)

MLE

- Maximizing the log likelihood on a training set $\{(X^l, Y^l) \mid l=1,\ldots,L\}$

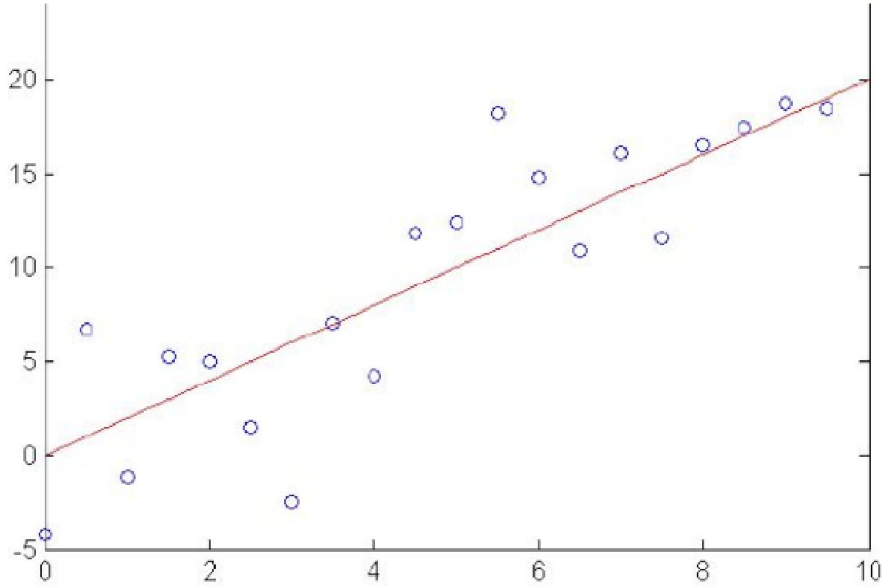
$$W^* = \text{argmax}_W \sum_{l=1}^{L} \log P(Y^l | X^l; W)$$

Where $P(Y|X) = N(f(X), \sigma^2) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left\{-\frac{(Y-f(X))^2}{2\sigma^2}\right\}$.

MLE

- The objective is equivalent to

$$W^* = \text{argmax}_W \sum_{l=1}^{L} \big(Y - f(X)\big)^2$$



where *f*(*X*) is a linear function, so it finds a line in hyperspace which best fits the training data, i.e., linear regression.

Solving linear regression problem

- Take derivative of each square term to *W,* by a chain rule

$$\frac{\partial \sum_l (y - f(x;W))^2}{\partial w_i} = \sum_l 2(y - f(x;W)) \frac{\partial (y - f(x;W))}{\partial w_i}$$

$$= \sum_l -2(y - f(x;W)) \frac{\partial f(x;W)}{\partial w_i}$$

Solving linear regression problem

- Set the derivative to zero,

$$\sum_{l=1}^{L} (Y^l - W^T X^l) X^l = 0$$

- An alternative matrix representation

$$\mathbf{XY} - \mathbf{XX}^{\mathrm{T}}\mathrm{W} = 0$$

Where $\mathbf{X}$ is data matrix, with each column being a training example, Y is the vector of target variables.

$$\mathbf{X} = [X^1, X^2, \dots, XL], \mathbf{Y} = [Y^1, Y^2, \dots, YL]^T$$

Pseudo Inverse

- Solving $\mathbf{XY} - \mathbf{XX}^{\mathrm{T}}\mathrm{W} = 0$

$$W = (\boldsymbol{X}\boldsymbol{X}^{\boldsymbol{T}})^{-\mathbf{1}}\boldsymbol{X}\boldsymbol{Y}$$

Where $\boldsymbol{X}^{+} = (\boldsymbol{X}\boldsymbol{X}^{\boldsymbol{T}})^{-\mathbf{1}}\boldsymbol{X}$ is called pseudo inverse. When $\boldsymbol{X}$ is invertible, then it is the inverse of the matrix.

- Compared with logistic regression, we have closed-form solution to the linear regression problem.

(pseudo inverse can be calculated for a rectangular matrix)

MAP solution

- Add a prior distribution on *W*, e.g., Gaussian.

  P(w) = N(0, σ^2 I)

- Maximizing the posterior distribution on W over the training set D

$$\log P(w|D) \propto log P(w) + \sum_{m=1}^{M} \log(P^l|X^l, w)$$

$$= \log P(w) + L(w)$$

- The optimal solution

$$w^* = \arg\max_w L(W) + \log P(W) = \arg\max_w \sum_{l=1}^{L} (Y^l - W^T X^l)^2 + \gamma ||w||_2^2$$

MAP solution

- Closed-form solution

$$W = (\boldsymbol{X}\boldsymbol{X}^T + \boldsymbol{\gamma}\boldsymbol{I})^{-1}\boldsymbol{X}\boldsymbol{Y}$$

- Improved performance
  - avoiding overfitting with prior information about W
- Improved numerical stability
  - avoiding ill-conditioned inverse matrix

Bias-Variance Decomposition

- Where does a prediction error come?
  - Suppose (X,Y) is drawn from a joint (unknown true) distribution
  - Perfect predictor is the conditional expectation E(Y|X)

$$E|Y - f(X)|^2 = \iint_{Y,X} |Y - f(X)|^2 P(X,Y)\,dX\,dY$$

$$= \iint_{Y,X} |Y - E(Y|X)|^2 P(X,Y)\,dX\,dY + \iint_{Y,X} |f(X) - E(Y|X)|^2 P(X)\,dX\,dY$$

  - The first term is unavoidable, causing by the uncertainty of data.
    - Given a X, its true variable is uncertain.
      - E.g., for a house, its “true” price is bargainable

Bias-Variance Decomposition

- Suppose a training set D, and *f*(*X;D*) is learned from this set.

$$\iint_{Y,X} |f(X) - E(Y|X)|^2 P(X)dXdY$$

$$= \int \{E_D[f(X;D)] - E(Y|X)\}^2 P(X)dX + \int E_D\{f(X;D) - E_D[f(X;D)]\}^2 P(X)dX$$

where $E_D[f(X;D)]$ is an average predictor over training sets, e.g., cross-validation.

- The first term is called bias: the deviation of the average predictor from the optimal one.
- The second term is called variance: how stable the learned predictors are over different training sets?

What we learn from bias-variance decomposition?

- A good predictor shall have
  - Small bias: be as close to the optimal predictor E[Y|X] as possible
  - Small variance: Be stable to the change of training set.
    - No matter how the training examples are drawn, the learned predictors shall not change too much.
- Unfortunately the two goals are not consistent
  - Flexible model usually can yield smaller bias but larger variance
    - Smaller γ in MAP leads to flexible model
    - Flexible enough to capture the subtle changes in training set
  - Rigid model usually has larger bias but smaller variance
    - Larger γ in MAP leads to rigid model
    - Stronger prior causes the model less flexible

More examples: KNN

- KNN becomes more rigid when K increases
  - As K increases, the neighborhood of a test example becomes larger and the prediction result will become less sensitive to the change of few neighbors.
- When K is small, KNN can be better adapted to the training set
  - A small change in training set will affect the predictions made by KNN
  - In this case, KNN is more flexible
- Making trade-off between larger K (rigid) and smaller K (flexible).

Summary: Linear Regession

- Optimal predictor in expected least square error sense
- Gaussian assumption leads to optimal linear predictor
- Learning linear regression model
  - MLE
  - MAP
- Bias-variance Decomposition

End of Fundamentals

Now you know all the fundamentals for machine learning!

Reading

- Chap 1-4 in Bishop (check the appendices for useful info)
- Chap 1-8 in Murphy
- Chap 1 in Marsland

Next lecture will start talking about specific classifiers: kernel methods (support vector machines) and neural networks