Midterm Review

Nan Li

2011.10.24
Things to Think about…

• How to train it?
  • What is the objective function?
  • Which algorithm could be used?
    • E.g. Gradient descent, EM
  • Optimality?

• What are the assumptions?
  • Linear boundary?

• What are the pros and cons comparing with other algorithms?
Decision Tree

- Input: a vector of attributes
- Output: a class, e.g. cheating or not
- Express any function of the input variables
- Overfitting
- Prefer compact tree
- Learning:
  - Greedily select the most informative attribute
  - Information gain

\[
GAIN_{split} = \text{Entropy}(p) - \left( \frac{1}{k} \sum_{i=1}^{k} \frac{n_i}{n} \text{Entropy}(i) \right)
\]

Parent Node, p is split into k partitions; n_i is number of records in partition i
K-Nearest Neighbour

- Learning is just storing all training examples
- Bayes classifier is the best classifier which minimizes the probability of classification error
- The error-rate of 1NN is less than twice the Bayes error – Error rate of classifier knowing model of generated data
  - What if the density estimates converge to the true densities?
Naïve Bayes

- Generative
  - Model $P(X, Y)$
- $P(Y|X) = P(X|Y)P(Y)/P(X)$
- Conditional Independence Assumption
  - $P(X_1, X_2 \ldots X_n|Y) = P(X_1|Y)P(X_2|Y)\ldots P(X_n|Y)$
- Why do we want to make this assumption?
- Training:
  - MLE
- Connection to logistic regression
Logistic Regression

- Discriminative
  - Directly model $P(Y|X)$

The condition distribution: a Bernoulli

$$p(y | x) = \mu(x)^y (1 - \mu(x))^{1-y}$$

where $\mu$ is a logistic function

$$\mu(x) = \frac{1}{1 + e^{-\theta^T x}}$$

- Training:
  - MCLE
  - Gradient ascent
  - Concave -> Global optimum

- Linear decision boundary
Naïve Bayes vs Logistic Regression

- When model assumptions correct
  - NB = LR

- When model assumptions incorrect
  - LR is less biased

- Convergence rate
  - NB order log m (m = # attributes in X)
  - LR order m
Linear Regression

- Assume that $Y$ (target) is a linear function of $X$ (features)
  \[ \hat{y} = \theta_0 + \theta_1 x^1 + \theta_2 x^2 + \ldots + \theta_k x^k \]
- Can be with non-linear basis functions

- Training
  - Least Mean Square (LMS)
    \[ J(\theta) = \frac{1}{2} \sum_{i=1}^{n} (x_i^T \theta - y_i)^2 \]
    1. Gradient descent, global optimum
    2. Set derivative to zero
       \[ \theta^* = (X^T X)^{-1} X^T \bar{y} \]
- Equivalence of LMS and MLE
Ridge Regression vs Lasso

$$\min_{\beta} (X\beta - Y)^T (X\beta - Y) + \lambda \text{pen}(\beta) = \min_{\beta} J(\beta) + \lambda \text{pen}(\beta)$$

Prior belief that $\beta$ is Gaussian with zero-mean biases solution to “small” $\beta$

Ridge Regression:
$$\text{pen}(\beta) = \|\beta\|_2^2$$

Prior belief that $\beta$ is Laplace with zero-mean biases solution to “small” $\beta$

Lasso:
$$\text{pen}(\beta) = \|\beta\|_1$$

$\beta$s with constant $J(\beta)$ (level sets of $J(\beta)$)

$\beta$s with constant $l_2$ norm

$\beta$s with constant $l_1$ norm

Lasso ($l_1$ penalty) results in sparse solutions – vector with more zero coordinates
Good for high-dimensional problems – don’t have to store all coordinates!
Neural Networks

- Perceptron
  - Training: Gradient descent
  - Decision boundary?
    - Linear if using sigmoid.
    - Not true in general.

- Neural Networks
  - Highly expressive
  - Training:
    - Backpropagation
    - Minimizing sum of squared training errors
  - May stuck in local optimum
Learning Theory

- A lot of definitions...
- And theorems...

- **Definition:** The *Vapnik-Chervonenkis dimension*, \( VC(H) \), of hypothesis space \( H \) defined over instance space \( X \) is the size of the largest finite subset of \( X \) shattered by \( H \). If arbitrarily large finite sets of \( X \) can be shattered by \( H \), then \( VC(H) = \infty \).

**Definition:**
Given a set \( S = \{ x(1), \ldots, x(d) \} \) of points \( x(i) \in X \), we say that \( H \) shatters \( S \) if \( H \) can realize any labeling on \( S \).
Overfitting and Model Selection

- Training Error vs Testing Error
- Cross Validation
  - Enough to run it once?
- Regularization
  - L1, L2
- Feature Selection
  - Filter: direct feature ranking
  - Wrapper: determine feature based on performance under the learning algorithm
  - Simultaneous learning and feature selection
Clustering

- Unsupervised learning
- Various distance metrics
  - E.g. Euclidean distance, Manhattan distance ...
- Hierarchical clustering
  - Bottom-up
  - Top-down
- K-Means
  - Known to converge
  - Sensitive to initial points
Expectation
Maximization

- Why do we need it?
  - Used when there are hidden variables

- Complete log likelihood
  \[ \ell_c(\theta; x, z) \]

- Incomplete log likelihood
  \[ \ell_c(\theta; x) \]

- Lower bound
  \[ F(q, \theta) = \sum_z q(z | x) \log \frac{p(x, z | \theta)}{q(z | x)} \leq \ell(\theta; x) \]

- Algorithm (May refer to the Bishop book for details)
  - E-step: filling in the latent variables using the best guess
  - M-step: updating the parameters based on this guess

- Local optimum

- A soft version of K-means
Hidden Markov Model

- Assumption
  - How many parameters needed?

\[
p(x, y) = p(x_1 \ldots x_T, y_1, \ldots, y_T) \quad \text{(Joint probability)}
\]
\[
= p(y_1) p(x_1 | y_1) p(y_2 | y_1) p(x_2 | y_2) \ldots p(y_T | y_{T-1}) p(x_T | y_T)
\]
\[
= p(y_1) P(y_2 | y_1) \ldots p(y_T | y_{T-1}) \times p(x_1 | y_1) p(x_2 | y_2) \ldots p(x_T | y_T)
\]
HMM

- **Forward**
  \[ \alpha(y_t^k = 1) = \alpha_t^k = P(x_1, \ldots, x_t, y_t^k = 1) \]
  \[ \alpha_t^k = p(x_t | y_t^k = 1) \sum_i \alpha_{t-1}^i a_{i,k} \]

- **Backward**
  \[ \beta_t^k = P(x_{t+1}, \ldots, x_T | y_t^k = 1) \]
  \[ \beta_t^k = \sum_i a_{k,i} p(x_{t+1} | y_{t+1}^i = 1) \beta_{t+1}^i \]

- **Viterbi**
  \[ V_t^k = \max_{\{y_1, \ldots, y_{t-1}\}} P(x_1, \ldots, x_{t-1}, y_1, \ldots, y_{t-1}, x_t, y_t^k = 1) \]
  \[ V_t^k = p(x_t | y_t^k = 1) \max_i a_{i,k} V_{t-1}^i \]

- **Learning**
  - Baum-Welch (EM)
  - Please refer to the paper if you are interested in knowing the details
Good Luck 😊