Week3 Report

Chengze Cai

caicz18@mails.tsinghua.edu.cn

1 Logistic Regression

To model the conditional probability $Pr(y = 1|x) \sim p(x)$ with a linear function, one of the most obvious way is let $\ln(\frac{p(x)}{1-p(x)}) = w^T x + w_0$, which leads to $p(x) = \frac{1}{1+\exp(-w^T x - w_0)}$.
The parameters w, w_0 can be found by maximize the log likelyhood:
$L = \sum_{y_i=1} \ln(p(x)) + \sum_{y_i=0} \ln(1 - p(x))$
$= \sum_{y_i=1} - \ln(1 + \exp(-(w^T x + w_0))) + \sum_{y_i=0} \ln(1 + \exp(-(w^T x + w_0)))$,
which is a convex function. This can be solved by gradient based optimization methods.
When data is linear seperatable, maximize the log likelyhood may lead to infinite coefficients. Large coefficients usually mean over-fitting. Adding an L2 regular term $-c\|w\|^2$ or L1 regular term $-c\|w\|$ may avoid over-fitting.
Logistic regression can be changed to deal with multi-classification problem:
$Pr(y = i|x) \sim \frac{\exp(w^{(i)T} x + w_0^{(i)})}{\sum_{j=1}^{c} \exp(w^{(j)T} x + w_0^{(j)})}$

2 Bagging

2.1 Basis

Training on different data would yield a different base model. If we combine these models, the variance of models will be decreased. However, we only have one dataset, so we need to use bootstrap sampling to generate different datasets.

2.2 Algorithm

input: dataset S
for t = 1,2, ... ,T:
    create bootstrap sample D_t from S
    train a classifier h_t on D_t
Classify new instance by majority vote of h_t (equal weights)

3 AdaBoost

3.1 Basis

Weak learners often have low variance and high bias. It's often possible to reduce the model bias by training a sequence of weak learners and give them different voting weight.

3.2 Analysis

In a binary classification problem, we represent each sample by (x_i, y_i), and $y_i = 1$. Using the exponential loss function, we are going to minimize $L = \sum_{i=1}^{m} \exp(-y_i H_t(x_i))$, where $H_t = \sum_{i=1}^{t} \alpha_t h_t$ is the ensemble of t weak learners $h_1, h_2, ..., h_t$.

$L = \sum_{i=1}^{m} \exp(-y_i H_t(x_i))$

$= \sum_{i=1}^{m} \exp(-y_i H_{t-1}(x_i) - y_i h_t(x_i; \theta_t))$

$= \sum_{i=1}^{m} w_i^{(t)} \exp(-y_i \alpha_t h_t(x_i; \theta_t))$

$= \exp(\alpha_t) \sum_{y_i h_t(x_i)=-1} w_i^{(t)} + \exp(-\alpha_t) \sum_{y_i h_t(x_i)=1} w_i^{(t)}$

$= \epsilon_t \exp(\alpha_t) + (1 - \epsilon_t) \exp(-\alpha_t)$

where $w_i^{(1)} = 1, w_i^{(t+1)} = w_i^{(t)} \exp(-y_i \alpha_t h_t(x_i))$ is the weight of i-th sample when training the t-th weak learner. To minimize the loss, we should select $\alpha_t = \frac{1}{2} \ln(\frac{1-\epsilon_t}{\epsilon_t})$, where ϵ_t is sum of the weights of all samples misclassified by h_t. Choosing different loss functions leads to different algorithms.

3.3 Algorithm

input: dataset S
Initially assign an equal weight to each example
for t = 1,2, ... ,T:
    train a classifier h_t on S with weights
    let ϵ_t = sum of the weights of all misclassified samples
    for each misclassified sample, multiply its weight by $\frac{1-\epsilon_t}{\epsilon_t}$
Combine all h_t with the voting weight $\ln(\frac{1-\epsilon_t}{\epsilon_t})$.