Statistics Review

junyan.xu[*]

April 14, 2019

Contents

[*] junyanxu5513@gmail.com

1 Support Vector Machine and Regression

1.1 Separable

A distance of a point a to a plane $w \cdot x + b = 0$ is $\frac{a \cdot w + b}{\|w\|}$. Define margin ζ to be the value that all (x_i, y_i) are having distance larger that this.

$$\frac{x_i \cdot w + b}{\|w\|} \geq \zeta \tag{1}$$

Then if we fix ζ to be 1, mathematically the margin is $\frac{1}{\|w\|}$. The we are actually minimize $\frac{1}{2}\|w\|^2$

1.2 Non Separable

If the data is not separable, then we add penalty to the optimization function (constant muliply soft margin).

$$\begin{aligned} loss &= \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i \\ s.t.\forall i, y_i(w.x_i + b) &>= 1 - \xi_i \end{aligned} \tag{2}$$

The x_i represent the hinge loss, an "option" style loss function. Which could be rewrite as $max\{0, 1 - y(w \cdot x + b)\}$

1.3 Solver

Currently according to Stanford course this one should use SGD to solve.

1.4 Dual Problem

According to representer theorem the w can be a linear combination of x_i and is $w = \sum_{i=1}^{n} \alpha_i x_i$. Then we have $f(x_i) = y_i \sum_{j=1}^{n} \alpha_j x_j^T x_i + b$ The primal problem of this is minimize:

$$\frac{1}{2}\sum_{jk}\alpha_j\alpha_k y_j y_k(x_j^\top) + \sum_{i=1}^{n} C \cdot max(0, 1 - y_i \sum_{j=1}^{n}\alpha_j x_j^T x_i + b) \tag{3}$$

The dual problem is maximize

$$\begin{aligned} &-\frac{1}{2}\sum_{jk}\alpha_j\alpha_k y_j y_k(x_j^\top) + \sum_{i=1}^{n}\alpha_i \\ &0 \leq \alpha_i \leq C \\ &\sum_i \alpha_i y_i = 0 \end{aligned} \tag{4}$$

1.5 Regression

For regression part. The we assume the regression lies in range ϵ. Also the penalty of upper ϵ is ξ_u and down is ξ_d So the primal problem is

$$loss = \frac{1}{2}\|w\|^2 + C\sum_{i=1}(max\{f(x_i) - y_i - \epsilon, 0\} + max\{y_i - f(x_i) - \epsilon, 0\}) \tag{5}$$

1.6 Summary

1. GOOD: SVM is good for handling high dimensional data
2. GOOD: SVM has customize kernal
3. BAD: No probability estimate

2 Kalman Filter and Hidden Markov Model

2.1 Basic Definition

Kalman filter has definition x_t and the transition matrix A_t. The x_t follows the formula

$$\begin{aligned} x_{t+1} &= A_t \cdot x_t + Normal(0, Q_t) + b_t \\ z_t &= C_t \cdot x_t + Normal(0, R_t) + d_t \end{aligned} \tag{6}$$

All the state transitions and observations are linear with Gaussian distributed noise, then the estimation can be represented by a mean plus a Gaussian distribution.

The kalman gain is a critical term in Kalman Filter. In the formula above, kalman gain can be defined as the coeffcient of innovation error that we need to update our prior estimation of x_t based on the current observation z_t. The idea is quite straight forward

$$\hat{x}_t|t = \hat{x}_t|t-1 + K_t \cdot (z_t - C_t \cdot \hat{x}_t|t-1)$$

The solving of kalman gain can be defined as minimize covariance of time t posterior estimation of x. Which is actually miniize the trace of error matrix.

$$K_t = \arg\min_K trace(cov(x_t - x_t|t-1 - K(C \cdot x_t + v_t - C_t \cdot x_t|t-1))) \tag{7}$$

What need to be noticed is that Q_t and R_t can be defined with some assumptions (exogeneous input), which is not a constant. The system can work fine too. The predicted variance follows

$$\begin{aligned} P_{t|t-1} &= A_t Q_t A_t^\top + R_t \\ P_{t|t} &= (I - K_t C_t) P_{t|t-1} \end{aligned} \tag{8}$$

2.2 EM Algorithm

Two step, the E step of this algorithm first assumed we have θ, then we can get distribution of hidden x. Then we calculate the expectation $E_x(l)$ of the likelyhood under hidden variable$x$. Next step is the M step, caclculate the max E to get theta

2.3 HMM vs Kalman (State Space Model)

1. HMM has discrte hidden state (support categorical, that's why second property holds), while Kalman has continuous hidden state (Gaussian transition).
2. HMM discrete state only have transition matrix, but no transition noise.

3 Trees

3.1 Decision Tree

The decsition tree algo can be think of a iterative constructing tree. Suppose current stage t we have tree K_t and node $k_{1 \ldots t} \in K_t$. The input data has M features. Then we will expand the tree by adding new m_{t+1} using

$$m_{t+1}, \theta_{t+1} = \underset{i \in t, m \in M, \theta \in \mathbb{R}}{\arg\min} \frac{n_{left}}{N_m} H(\{(x,y)|x_j \leq \theta\}) + \frac{n_{right}}{N_m} H(\{(x,y)|x_j \geq \theta\}) \tag{9}$$

For classification, H can be defined as $-\sum_k p_{mk} \cdot log(p_{mk})$. For regression, H can be defined as $-\sum_k (y_i - \hat{y}_m)^2$. The key thing about decision tree is that the definition is recursive which is different from SVM's global optimization.

The strength and weakness of this algo is:

1. GOOD: It is implicity doing data selection. Easy interpretation.
2. GOOD: Non-linearity low effection.
3. BAD: Inaccurate often
4. BAD: Weak in dealing with continuous output.

3.2 Random Forest

Perform both sample bagging and feature bagging. In Sklearn, the bagging are performed by only sample bagging, where sub-sample each time is the same size as original input.

3.3 XGBoost

Different from random forest, XGBoost start from building sequential trees. It tells you how much you should build extra tree to minimize loss given previous structure.

$$
\begin{aligned}
L^t - L^{t-1} &= \sum_{i=1}^{n} l(y_i, \hat{y}_i^{t-1} + f_t(x_i)) + \Omega(f_t) - \sum_{i=1}^{n} l(y_i, \hat{y}_i^{t-1}) \\
\Omega(f_t) &= \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} ||w_j||^2
\end{aligned}
\tag{10}
$$

However we don't know the structure of the next tree yet, but we do know the give a fixed structure of next tree, the weight w_j on leaf j should follow some constrain.

Define $I_j = \{i|q(x_i) = j\}$ as instance set on leaf j. The loss can be further write as:

$$
\sum_{j=1}^{T} [g(y_i)w_i + 0.5(\sum_{i \in I_j} h_i + \lambda)w_j^2] + \gamma T \tag{11}
$$

Taking derivative for w_i, we have optimal $w_i = \frac{\sum_{i \in I_j} g_i}{\sum_{i \in I_j} h_i + \lambda}$. This L can implied the optimal loss of adding a substree as well as a score function for coordinating a split(Recall in a single tree mode our loss function is just the change of the entropy). Then we can use normal greedy algorithm to find out the split of next tree.

4 Factor Selection

4.1 Stepwise Selection

For stepwise selection, we have forward and backward type. Backward start from the full model and remove the least impact variable. Forward start from 0 and add most impact model. They are greedy algorithm

4.2 Stagewise Selection

It should be properly named as (Error Correlation Gradient Descent). Because the algo is adding variable selected by their correlation with the error term (with small step ϵ)

$$
\hat{c} = c(\hat{\mu}) = X^{\top}(y - \hat{\mu}) \tag{12}
$$

The next μ is $\hat{j} = argmax|\hat{c}_j|$ then $\mu = \mu + \epsilon \cdot sign(\hat{c}_j) \cdot x_j$