Decision Trees & Ensemble Methods

Decision trees: nonlinear; greedy, top-down, recursive partitioning

Looking for a split s_p: $s_p(j,t) = (\underbrace{\{x|x_j < t, x_j \in R_p\}}_{R_1}, \underbrace{\{x|x_j \geq t, x_j \in R_p\}}_{R_2})$

Define $L(R)$: loss on region R

Given C classes, define $\hat{p}_c$ to be the proportion of examples in R that are of class C

Objective: $\max L(R_p) - (L(R_1) + L(R_2))$

Uses cross entropy loss: $L_{cross} = -\Sigma_c(\hat{p}_c \cdot \log_2 \hat{p}_c)$

Regularization of decision trees:

1. min leaf size
2. max depth
3. max number of nodes
4. min decrease in loss
5. pruning with validation set

Runtime: n examples, f features, d depth (usually $d < log_2 n$)

Test time $O(d)$

Train time: $O(nfd)$ since each point is part of $O(d)$ nodes, cost of point at each node is $O(f)$

Strength: 1. easy to explain; 2. interpretable; 3. categorical variables; 4. fast

Weakness: 1. no additive structure; 2. high variance, easy to overfit; 3. due to 1, 2 → low predictive accuracy

Ensembling

Take x_i's which are random variables (RV) that are independent identically distributed (i.i.d.),

$Var(x_i) = \sigma^2, Var(\bar{x}) = \frac{\sigma^2}{n}$

Drop independence assumption, so now x_i's are i.d., x_i's correlated by p,

$Var(\bar{X}) = p\sigma^2 + \frac{1-p}{n}\sigma^2$

Ways to ensemble:

1. different algorithms
2. different training sets
3. bagging (e.g. random forests)
4. boosting (e.g. Adaboost, xgboost)

Bagging - Bootstrap AGGregatING

True population P, Training set S~P

Assume P=S, Booststrap samples Z_1, …, Z_M ~ S

Train model G_m on Z_m,

$G(m) = \frac{\Sigma_{m=1}^{M} G_m(x)}{M}$

Bias-Variance Analysis: $Var(\bar{X}) = p\sigma^2 + \frac{1-p}{n}\sigma^2$

Bootstrapping is driving down p, more M → less variance; bias slightly increases

DT are high variance low bias, ideal fit for bagging → random forests

Boosting

Decrease bias, addictive

Adaboost: Determine for classifier G_m a weight $\alpha_m = \log(\frac{1-err_m}{err_m})$, then $G(x) = \Sigma \alpha_m G_m$