Lecture 12: Ensemble Methods

Isabelle Guyon guyoni@inf.ethz.ch

Book Chapter 7

Introduction

Weighted Majority

- Assume K experts f₁, f₂, ...f_K (base learners)
- Each expert makes a decision $f_k(\mathbf{x}) = \pm 1$
- Improve predictions by making the experts "vote" according to how good they are:

$$\mathsf{F}(\mathbf{x}) = \sum_{k} \alpha_{k} f_{k}(\mathbf{x}) \qquad 0 \leq \alpha_{k} \leq 1$$

$$\sum_{k} \alpha_{k} = 1$$

• Decision: sign[F(x)]







Bias: $[E_D f(\mathbf{x}, D) - y]^2$

- E_Df(x) : your "ideal" committee machine.
- Bias : what your ideal committee can't learn (from m training examples)
- E_Df(x) has the same bias as E_Df(x) but no variance.
- Note: Each committee member was trained on a different set on m examples...

Variance: $E_D[f(\mathbf{x}, D) - E_D f(\mathbf{x}, D)]^2$

- E_Df(x) : your "ideal" committee machine.
- Variance : how far apart on average your solution f(x,D) is from your "ideal" committee machine.
- If the variance is high but the bias is low: there is hope that a committee can improve performance.
- Note: Subsampling introduces extra bias ...



- 1) Mixture of decision stumps ⇔ feature selection
- 2) Merging expert feature rankings:
 - Average ranking index: $C_i = \sum_k \alpha_k C_{ik}$
 - Average rank: $C_j = \Sigma_k \alpha_k (R_{max}-R_{jk})$
- 3) Merging feature sets selected by experts: - Ranking index: $C_i = \sum_k \alpha_k \delta_{ik}$ ($\delta_{ik}=1$ if feat j selected
 - by expert k, 0 otherwise)
 - $S^* = \operatorname{argmax}_k \min_k |S_k \cap S_k|$ (most "stable" subset)
 - _ $R^* = \operatorname{argmin}_k \max_{k'} \operatorname{dist}(R_k, R_{k'})$
 - http://people.revoledu.com/kardi/tutorial/Similarity/OrdinalVariables.html
- 4) Sensitivity-based (special for bagging)







Difficulties

Continuous case: Infinitely many experts, we can't try them all!
Idea:

Let's take a sample...

• How?

Grid, heuristic search, stochastic search

• Important: Avoid sampling "poor" experts or "redundant" experts.





Variable-dimension MCMC

Vehtari and Lampinen, 2002

- Some steps include removal or addition of a feature
- We obtain P(model,feature-subset|D) for some samples of models and feature subsets
- Subset relevance can be computed by marginalization (averaging over the functions using the same subset)
- Feature relevance can also be computed by marginalization (averaging over all subsets containing that feature)

Performance Gain?

- If we draw M classifiers \boldsymbol{f}_k according to $P(\boldsymbol{f}|\boldsymbol{D}),$ we can approximate

 $\mathsf{P}(y|\mathbf{x},\mathsf{D}) = \int_{\mathsf{f}} \mathsf{P}(\mathsf{f}|\mathsf{D}) \ \mathsf{P}(y|\mathbf{x},\mathsf{f},\mathsf{D}) \ \mathsf{df}$ by

$$P(y|\mathbf{x},D) \sim \sum_{k=1:M} P(y|\mathbf{x},f_k,D)$$

• Relative error difference with optimum Bayes classifier decays with O(1/M) (Ng, Jordan, 2001)

Non-Bayesian Approaches

- Parallel ensembles: bagging
- Serial ensembles: boosting

Bagging

Breiman, 1996

- Bootstrap Aggregation:
 - Draw with replacement m samples from the original training set of size m
 - Train a learning machine
 - Repeat many time
 - On average, each example appears in the training set (1-1/m)^m~1-e⁻¹~0.632 times

Random Forests

Breiman, 2001

- A number n is specified much smaller than the total number N of variables (typically n ~ sqrt(N))
- 2. Each tree of maximum depth is grown on a bootstrap sample of the training set
- 3. At each node, n variables are selected at random out of the N $\,$
- 4. The split used is the best split on these n variables









Sensitivity-based Scoring

Breiman, 2001

- Classify the OOB cases and count the number of votes cast for the correct class in every tree grown in the forest
- Randomly permute the values of feature f in the OOB cases and classify these cases down the trees
- Subtract the number of votes for the correct class in the feature-f permuted OOB data from the untouched OOB data
- Average this number over all trees in the forest to obtain the importance score R(f)

Cross-validated Committee

Parmanto et al., 1996

- Any learning machine
- Any method of splitting the (training) data many times into training set and validation set (vset)
- Perturb feature f randomly in vset (pvset)
- R(f) = mean[num-correct-class(vset) num-correct-class(pvset)]
- Zscore = R(f)/stderror

Boosting

- Adaboost (Freund and Schapire, 1996): At every step add a new base learner that is forced (by re-weighting the training data) to concentrate on misclassified examples.
- Forward stagewise boosting (Breiman, 1997, Friedman et al., 2000)
 - 1. Initialize F(x)=0
 - 2. For k=1 to M

$$\begin{split} \mathsf{F}(\boldsymbol{x}) \leftarrow \mathsf{F}(\boldsymbol{x}) + \alpha \ \mathsf{f}(\boldsymbol{x}) \\ (\alpha_{\mathsf{k}}, \ \mathsf{f}_{\mathsf{k}}) = \text{argmin}_{\alpha,\mathsf{f}} \ \Sigma_{\mathsf{t}} \ \text{exp}(\text{-}\mathsf{y}_{\mathsf{i}} \ \mathsf{F}(\boldsymbol{x}_{\mathsf{i}})) \end{split}$$

3. Output $F(\mathbf{x}) = \sum_{k=1:M} \alpha_k f_k(\mathbf{x})$

L(y, f(x)) 4 Margin Decision 3.5 boundary Adaboost³ SVC loss, $\beta=2$ $\max(0, (1 - z))^2$ loss e-z 25 logistic loss square loss SVC loss, β= log(1+e-z) 15 $(1 - z)^2$ $\max(0, 1-z)$ 0/1Perceptrom max(0, -z $z \rightarrow z = y f(\mathbf{x})$ missclassified well classified

Loss Functions

Conclusion

- Ensemble methods help reducing the "variance"
- They benefit most to "low bias" base learners
- One should not confuse feature set variability and variance in predictions
- CV committees allow to rank features according to sensitivity ans compute zscores.

Exercise Class

Arcene Boosting



Forward Stagewise Boosting

- 1. Initialize F(x)=0
- 2. For k=1 to M
 - $\mathsf{F}(\mathbf{x}) \leftarrow \mathsf{F}(\mathbf{x}) + \alpha \, \mathsf{f}(\mathbf{x})$
 - $(\boldsymbol{\alpha}_{k},\,f_{k}) = argmin_{\!\boldsymbol{\alpha},f}\,\boldsymbol{\Sigma}_{\!i}\,\exp(\!\!\cdot\!\boldsymbol{y}_{i}\,F(\boldsymbol{x}_{i}))$
- 3. Output $F(x) = \sum_{k=1:M} \alpha_k f_k(\mathbf{x})$

At step t:

 $\begin{aligned} (\alpha_k, \ f_k) &= argmin_{\alpha, f} \ \Sigma_i \ exp[-y_i \ (F_{t-1}(\boldsymbol{x}_i) + \alpha \ f(\boldsymbol{x}_i))] \\ Compute \ \alpha_k, \ f_k \ for \ decision \ stumps \end{aligned}$



