Ensemble Learning

C. Andy Tsao

Institute of Statistics/Department of Applied Math
National Dong Hwa University, Hualien

March, 2015
Kaohsiung, Taiwan

Outline

- Overview
- Regression: theme and variations
- CART and tree-based methods
- Random Forests
- Boosting: AdaBoost and its variants
- Concluding Remarks

Ensemble Learning

Figure : What are they?

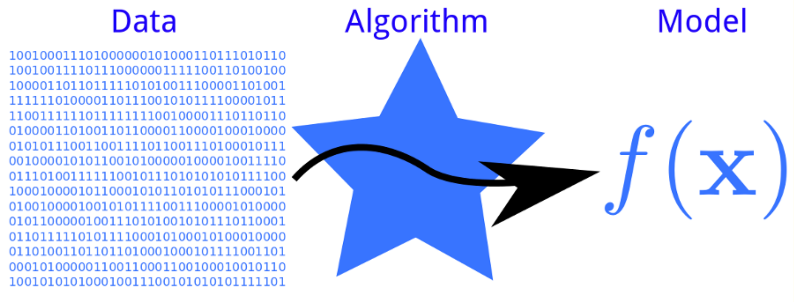


Figure : Machine Learning

Figure : (Jazz) Ensemble

Figure : Ensemble 13

Supervised Learning

- Training data:
  $\{(x_i, y_i)\}_{i=1}^n$, where $x_i \in \mathcal{X} \subset \mathcal{R}^p$ and $y_i \in \mathcal{Y} = \{\pm 1\}$ for classification;
  ($\mathcal{Y} = R = (-\infty, \infty)$ for regression)
- Testing (generalization) data: $\{(x'_j, y'_j)\}_{j=1}^m$
- Data: $(x, y) \overset{from}{\leftarrow} (X, Y) \overset{iid}{\sim} P_{X,Y}$
- Machine or classifier: $\widehat{F} \in \mathcal{F}$ such that $\widehat{F} : \mathcal{X} \to \mathcal{Y}$

Supervised Learning

- Training data:
  $\{(x_i, y_i)\}_{i=1}^n$, where $x_i \in \mathcal{X} \subset \mathcal{R}^p$ and $y_i \in \mathcal{Y} = \{\pm 1\}$ for classification;
  ($\mathcal{Y} = R = (-\infty, \infty)$ for regression)
- Testing (generalization) data: $\{(x'_j, y'_j)\}_{j=1}^m$
- Data: $(x, y) \overset{from}{\leftarrow} (X, Y) \overset{iid}{\sim} P_{X,Y}$
- Machine or classifier: $\widehat{F} \in \mathcal{F}$ such that $\widehat{F} : \mathcal{X} \to \mathcal{Y}$
- Training error:

$$TE = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}_{[y_i \neq \widehat{F}(x_i)]} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}_{[y_i \widehat{F}(x_i) < 0]}$$

- Testing (generalization) error:

$$\widehat{GE} = \frac{1}{m} \sum_{j=1}^{m} \mathbf{1}_{[y'_j \widehat{F}(x'_j) < 0]} \text{ and } GE = E_{X,Y}\{\mathbf{1}_{[YF(X)<0]}\}$$

Supervised Learning-II

With respect to a loss L

$$TE(F) = \frac{1}{n}\sum_{i=1}^{n} L(y_i, F(x_i)), \quad GE(F) = \frac{1}{m}\sum_{j=1}^{m} L(y_j', F(x_j'))$$

again GE is an estimate for $E_{Y,X} L(Y, F(X))$.

For regression, $L(y, F(x)) = (y - F(x))^2$ is widely used.

Regression-theme

(Classical) Regression

- Data: $(x_i, y_i)_{i=1}^n$, where $x_i \in \mathcal{X} = \mathcal{R}^p, y_i \in \mathcal{Y} = \mathcal{R}$.
  Distribution: $(y_i|x_i)_{i=1}^n \sim_{indep.} P_{Y|x}$.
- Class of learners: $\mathcal{F} = \{f(X) : f(X) = \beta_0 + \beta' X = \beta_0 + \sum_{j=1}^p \beta_j X_j,$ for $\beta_0 \in \mathcal{R}, \beta \in \mathcal{R}^p\}$.
- Construction: Least square errors (LSE)

$$SSE(\hat{F}) = ||Y - \hat{Y}||^2 = \sum_{i=1}^n (y_i - \hat{F}(x_i))^2 = \min_{F \in \mathcal{F}} \sum_{i=1}^n (y_i - F(x_i))^2$$

  where $\widehat{F}(x) = \hat{\beta}_0 + \hat{\beta}' x$.
- Evaluation: Sum of square errors (SSE or equivalently MSE).

Regression-v-class

Naive regression (Classification)

- Data:
  $(x_i, y_i)_{i=1}^n$, where $x_i \in \mathcal{X} = \mathcal{R}^p$, $y_i \in \mathcal{Y} = \{\pm 1\}$.
  Distribution: $(y_i, x_i)_{i=1}^n \sim_{indep.} P_{Y,X}$.
- Class of learners: $\mathcal{F}$, the collection of linear functions of $1, X_1, \cdots, X_p$.
- Construction: Least square errors (LSE)
- Evaluation: TE, GE (with respect to zero-one loss function)

$$TE = \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}_{[y_i \widehat{F}(x_i)<0]}, \quad GE = \frac{1}{m}\sum_{j=1}^{m} \mathbf{1}_{[y_j' \widehat{F}(x_j')<0]}$$

Regression-v-random

(random ensemble) Regression

- Data: $(x_i, y_i)_{i=1}^n$, where $x_i \in \mathcal{X} = \mathcal{R}^p$, $y_i \in \mathcal{Y} = \mathcal{R}$.
  Distribution: $(y_i, x_i)_{i=1}^n \sim_{indep.} P_{Y,X}$.
- Class of base learners: $\mathcal{F}_B = \{1, X_1, \cdots, X_p\}$
- Construction: Random subset regression
  - For $k = 1, 2, \cdots, K$
    1. Randomly choose m base learners $f \in \mathcal{F}_B$, $m < p + 1$.
    2. Fit a (subset) regression (LSE): $\hat{f}_k$, i.e. regress Y on the chosen m independent variables.
  - $\hat{F} = \sum_{k=1}^K w_k \hat{f}_k$ where w's are the weights of the k-th learner. Usually $\sum_k w_k = 1, 0 < w_k < 1$.
- Evaluation: Sum of square errors (SSE or equivalently mean square errors MSE)

Random (Average) Regression

- Data:
  $(x_i, y_i)_{i=1}^n$, where $x_i \in \mathcal{X} = \mathcal{R}^p, y_i \in \mathcal{Y} = \mathcal{R} = (-\infty, \infty)$.
  Distribution: $(y_i, x_i)_{i=1}^n \sim_{indep.} P_{Y,X}$.
- Class of base learners:
  $\mathcal{F} = \{1, X_1, \cdots, X_p\}$
- Construction: Random subset regression
  - Repeat 1, 2 for K times
    1. Randomly choose m X's from $\mathcal{F}$, $m < p + 1$.
    2. Fit(LSE) a (subset) regression: $\hat{f}_k$, i.e. regress Y on the chosen m independent variables.
  - $\hat{F}(x) = \frac{1}{K}\sum_{k=1}^{K} \hat{f}_k(x)$.
- Evaluation: SSE or MSE

Recap

Random Average Scheme (RA) + Base Learners

- Random Average Regression: RA+(subset) Regression
- Random Forests: RA+(subset) CART

Boosting and variations

- Weighted average of CARTs with reweighted data-feeds
- The population version of AdaBoost is a Newton-like updates for minimizing exponential criterion $E_{Y|x}\{e^{-YF(x)}\}$ (loss for construction)

Friedman, Hastie and Tibishirani (2000)

CART Algorithm (Regression)

- Data: $(x_i, y_i)_{i=1}^n$, where $x_i \in \mathcal{X} = \mathcal{R}^p, y_i \in \mathcal{Y} = \mathcal{R} = (-\infty, \infty)$ where $x_i = (x_{i1}, \cdots, x_{ip})', i = 1, \cdots, n$.
- Greedy recursive binary partition
  1. Find the split variable/point (j, s) that solve

$$SSE_1 = \min_{j,t} \left[\min_{c_1} \sum_{x_i \in R_1(j,t)} (y_i - c_1)^2 + \min_{c_2} \sum_{x_i \in R_2(j,t)} (y_i - c_2)^2 \right] \tag{1}$$

     where $R_1(j,t) = \{X | X_j \leq t\}, R_2(j,t) = \{X | X_j > t\}$
  2. Given (j, t), $(\hat{c}_1, \hat{c}_2)$ solves the inner minimization and $\hat{c}_l = ave(y_i | x_i \in R_l(j,t)), l = 1, 2.$
  3. Continue adding split one at a time $\rightsquigarrow R_1, \cdots, R_M$
- $\hat{F}(x) = \sum_{m=1}^M \hat{c_m} 1_{[x \in R_m]}$.
- Evaluation: SSE or MSE

Hastie, Tibshirani and Friedman (2001).

RF and Boosting R package

- Titanic: Getting Started With R
- `rpart`, `randomForest` packages at `CRAN`
- Boosting algorithms
- `ada`, `adabag`, `gbm`, `mboost` packages
- Link: Rstudio, kaggle

Random Forests (for regression)

- Data: $(x_i, y_i)_{i=1}^n$, where $x_i \in \mathcal{X} = \mathcal{R}^p, y_i \in \mathcal{Y} = \mathcal{R}$.
  Distribution: $(y_i, x_i)_{i=1}^n \sim_{iid} P_{Y,X}$.
- Class of base learners:
  $\mathcal{F}_B$: the collection of regression trees with independent variables being a subset of $X_1, \cdots, X_p$.
- Construction: Random Forests
  - For $k = 1, 2, \cdots, K$
    1. Draw a bootstrap sample Z^* of size n from the data.
    2. Randomly choose m independent variables among $X_1, \cdots, X_p$
    3. Fit a regression tree (CART): i.e. Construct $\hat{f}_k$ for Y on the chosen m independent variables with $^Z*$
  - $\hat{F} = \frac{1}{K} \sum_{k=1}^K \hat{f}_k$.
- Evaluation criterion: SSE, MSE.

Random Forests (classification)

The classification algorithm is similar except

- class of base learners = the collection of classification trees
- In choosing the split variable/point, square error is substituted by classification impurity measures, say misclassification error.
- $\hat{F}(x) = \sum_{m=1}^{M} c_m 1_{[x \in R_m]}$ where $c_m = majority(y_i | x_i \in R_m)$.

AdaBoost-I

Given $(x_i, y_i)_{i=1}^n$ where $x_i \in \mathcal{X}$ and $y_i \in \mathcal{Y} = \{\pm 1\}$.

Algorithm

1. Initialize $D_1(i) = n^{-1}$ for $i = 1, 2, \cdots, n$
   $D_t(i)$ = the weight on the i-th case in the t-th iteration.
2. For $t = 1, 2, \cdots, T$
   - Construct the (trained) learner $h_t : \mathcal{X} \to \mathcal{Y}$ using the weight D_t on the data.
   - Compute the error $\epsilon_t = \sum_{i=1}^n D_t(i) 1_{[h_t(x_i) y_i < 0]}$. Then $\alpha_t = \ln(\frac{1-\epsilon_t}{\epsilon_t})$.
   - Update the weights $D_{t+1}(i) = D_t(i) e^{\alpha_t 1_{[h_t(x_i) y_i < 0]}} \quad \forall i$
   - Normalization $D_{t+1}(i) = D_{t+1}(i) \left(\sum_{i=1}^n D_{t+1}(i)\right)^{-1} \quad \forall i$
3. Output the final hypothesis $F(x) = \text{sgn}\left[\sum_{t=1}^T \alpha_t h_t(x)\right]$.

ABC about Boosting

- Start with a simple type learner (reg, tree, etc)
- Train the learner several times (learning process) and obtain the fitted learner using data
- Higher weights to misclassified cases each time
- Final prediction: weighted majority vote

ABC about Boosting

- Start with a simple type learner (reg, tree, etc)
- Train the learner several times (learning process) and obtain the fitted learner using data
- Higher weights to misclassified cases each time
- Final prediction: weighted majority vote
- Criteria: TE: Training error and GE: Testing error

ABC about Boosting

- Start with a simple type learner (reg, tree, etc)
- Train the learner several times (learning process) and obtain the fitted learner using data
- Higher weights to misclassified cases each time
- Final prediction: weighted majority vote
- Criteria: TE: Training error and GE: Testing error

- Under weak base hypothesis assumption, TE $\to 0$.
- Rather immune to overfitting for less noisy data in apps.
- Breiman (1996): "Best off-the-shelf classifier in the world"

FHT's Results

Friedman, Hastie, and Tibishirani (2000)

- The population version of AdaBoost is a Newton-like updates for minimizing exponential criterion $E_{Y|x}\{e^{-YF(x)}\}$.

FHT's Results

Friedman, Hastie, and Tibishirani (2000)

- The population version of AdaBoost is a Newton-like updates for minimizing exponential criterion $E_{Y|x}\{e^{-YF(x)}\}$.
- Let $J_e(F) = E_{Y|x}\{e^{-YF(x)}\}$ and

$$\mathop{\mathbf{arg\ min}}_{f} \tilde{J}_e(F+f) \rightsquigarrow \mathop{\mathbf{arg\ min}}_{s,c} \tilde{J}_e(F+sc),$$

where $\tilde{J}_e$ is an approximation of J_e.

FHT's Results

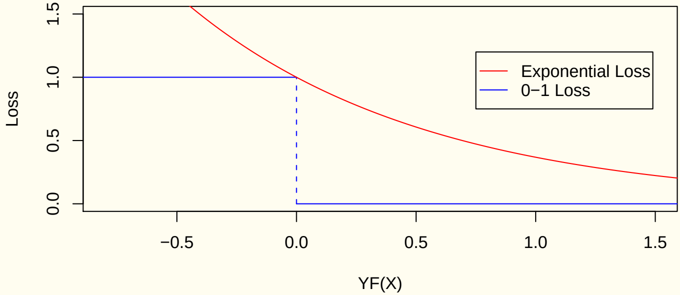
Friedman, Hastie, and Tibishirani (2000)

- The population version of AdaBoost is a Newton-like updates for minimizing exponential criterion $E_{Y|x}\{e^{-YF(x)}\}$.
- Let $J_e(F) = E_{Y|x}\{e^{-YF(x)}\}$ and
$$\mathbf{arg\,min}_f \tilde{J}_e(F+f) \rightsquigarrow \mathbf{arg\,min}_{s,c} \tilde{J}_e(F+sc),$$
where $\tilde{J}_e$ is an approximation of J_e.
- The solutions are
$$s = \begin{cases} +1, & \text{if } E_{w_e}\{Y|x\} > 0, \\ -1, & \text{otherwise,} \end{cases} \quad \text{and } c = \frac{1}{2}\ln\left(\frac{1-\mathbf{err}}{\mathbf{err}}\right),$$
where $w_e = e^{-YF(x)}$ and $\mathbf{err} = E_{w_e}\left\{\mathbf{1}_{[Y \neq s(x)]}\right\}$.

Exponential Loss

The 0-1 loss can be bounded by exponential loss

$$\mathbf{1}_{[Y \neq F(x)]} = \mathbf{1}_{[YF(x)<0]} \leq e^{-YF(x)} \overset{def}{=} L_e[Y, F(x)].$$



Concluding remarks

- Framework
- RF, Boosting are powerful and better "off-the-shelf"(?) learners
- High dimensional problem ready

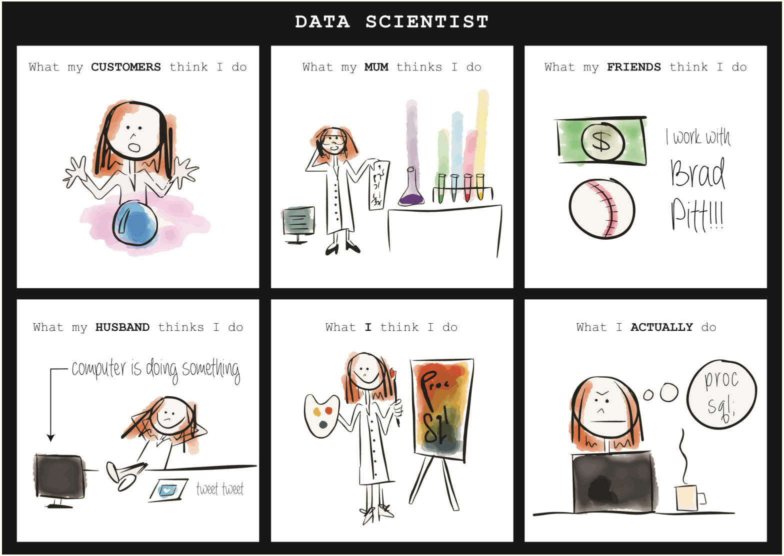


Figure : What does a Data Scientist do?

Thanks for your attention!

References

- Biau, G., Devroye, L. and Lugosi, G. (2008). Consistency of random forests and other average classifiers, *Journal of Machine Learning and Research*, **9**, 2039–2057.
- Breiman, L. (2000). Some infinite theory for predictor ensembles. Technical Report 577, Statistics Department, UC Berkeley, 2000.
- Breiman, L. (2001). Random Forests, *Machine Learning*, **45**, 5–32.
- Breiman, L., Friedman, J., Olshen, R. and Stone, C. (1984). Classification and Regression Trees. Wadsworth.
- Hastie, T., Tibshirani, R. and Friedman, J. (2001). The Element of Statistical Learning: Data mining, inference and prediction. Springer-Verlag.
- FRIEDMAN, J. H., HASTIE, T. AND TIBISHIRANI, R. (2000). Additive logistic regression: a statistical view of boosting. *Annals of Statistics*, **28**, 337–407.
- Hong, B.-Z. (2013). Random Average Regression Methods. Master Thesis. National Dong Hwa University. Taiwan.
- Tsao, C. A. (2014). A Statistical Introduction to Ensemble Learning Methods. *Journal of Chinese Statistical Association*, **52**, 115–132.

Performance of Average Learner

For the training data $D = (y_i, x_i)_{i=1}^n$, the MSE for learner f_k is

$$MSE(f_k) = \frac{1}{n}\sum_{i=1}^{n}(y_i - f_k(x_i))^2.$$

Let $w_k =$ the weight of f_k with $\sum_{k=1}^K w_k = 1$ and $0 < w_k < 1$. Note

$$E_w MSE(f_k) = \sum_{k=1}^{K} w_k \left(\frac{1}{n}\sum_{i=1}^{n}(y_i - f_k(x_i))^2\right) \geq \frac{1}{n}\sum_{i=1}^{n}(y_i - E_w f_w(x_i))^2 = MSE(E_w f_w)$$

where $E_w f_w(x) = \sum_{k=1}^K w_k f_k(x)$.

When $w_k = 1/K$, $E_w f_w(x)$ is the average of the K learners.

$$MSE(E_w f_w) \leq \text{average}_k MSE(f_k) = \frac{1}{K} \sum_k MSE(f_k).$$

And it is not necessarily

$$MSE(E_w f_w) \leq MSE(f_k) \quad \textit{for all } k.$$