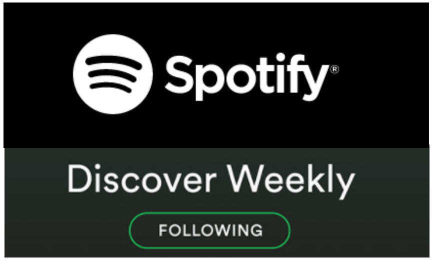
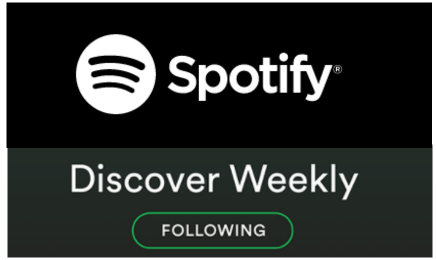
Announcements

- My **office hours TODAY 3:30 pm - 4:30 pm CSE 666**
- Poster Session - Pick **one**
  - First poster session **TODAY 4:30 pm - 7:30 pm CSE Atrium**
  - Second poster session December 12 4:30 pm - 7:30 pm **CSE Atrium**
  - Support your peers and check out the posters!
  - Poster description from website:
    "We will hold a poster session in the Atrium of the Paul Allen Center. Each team will be given a stand to present a poster summarizing the project motivation, methodology, and results. The poster session will give you a chance to show off the hard work you put into your project, and to learn about the projects of your peers. We will provide poster boards that are 32x40 inches. Both one large poster or several pinned pages are OK (fonts should be easily readable from 5 feet away)."
- Course Evaluation: https://uw.iasystem.org/survey/200308 (or on MyUW)
- Other anonymous Google form course feedback: https://bit.ly/2rmdYAc
- Homework 3 Problem 5 "revisited".
  - **Optional.** Can only increase your grade, but will not hurt it.



You may also like…

ML uses past data to make personalized predictions



You may also like…

ML uses past data to make personalized predictions

Basics of Fair ML

You work at a bank that gives loans based on credit score.

You have historical data: $\{(x_i, y_i)\}_{i=1}^n$

credit score $x_i \in \mathbb{R}$

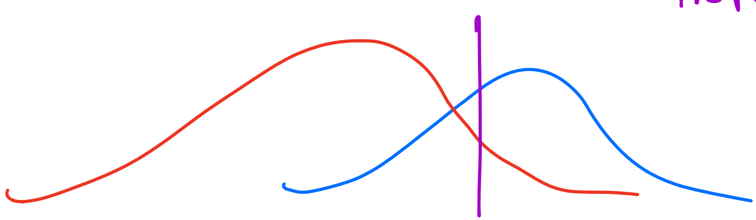
paid back loan $y_i \in \{0, 1\}$

If the loan ~~defaults~~ is paid back ($y_i = 1$) you receive \$300 in interest

If the loan defaults ($y_i = 0$) you lose \$700

For some threshold t

$$\text{Profit} = 300 \cdot \mathbb{P}(x_i > t \mid y_i = 1) - 700 \cdot \mathbb{P}(x_i > t \mid y_i = 0)$$



Discussion based on [Hardt, Price, Srebro '16]. See http://www.fatml.org for more resources.

Basics of Fair ML

You work at a bank that gives loans based on credit score.

Boss tells you "make sure it doesn't discriminate on race"

You have historical data: $\{(x_i, a_i, y_i)\}_{i=1}^n$

credit score $x_i \in \mathbb{R}$

paid back loan $y_i \in \{0, 1\}$

race $a_i \in \{\text{asian}, \text{white}, \text{hispanic}, \text{black}\}$

paid back

If the ~~loan defaults~~ $(y_i = 1)$ you receive \$300 in interest

If the loan defaults $(y_i = 0)$ you lose \$700

Discussion based on [Hardt, Price, Srebro '16]. See http://www.fatml.org for more resources.

Basics of Fair ML

You work at a bank that gives loans based on credit score.

Boss tells you "make sure it doesn't discriminate on race"

You have historical data: $\{(x_i, a_i, y_i)\}_{i=1}^n$

credit score $x_i \in \mathbb{R}$

paid back loan $y_i \in \{0, 1\}$

race $a_i \in \{\text{asian}, \text{white}, \text{hispanic}, \text{black}\}$

- ***Fairness through unawareness***. Ignore a_i, everyone gets same threshold
    - **Pro:** simple,
    - **Con:** features are often proxy for protected group

$$\mathbb{P}(x_i > t | a_i = \square) = \mathbb{P}(x_i > t)$$

Discussion based on [Hardt, Price, Srebro '16]. See http://www.fatml.org for more resources.

Basics of Fair ML

You work at a bank that gives loans based on credit score.
Boss tells you "make sure it doesn't discriminate on race"

You have historical data: $\{(x_i, a_i, y_i)\}_{i=1}^n$

credit score $x_i \in \mathbb{R}$

paid back loan $y_i \in \{0, 1\}$

race $a_i \in \{\text{asian}, \text{white}, \text{hispanic}, \text{black}\}$

- ***Demographic parity***. proportion of loans to each group is the same
  - **Pro:** sounds fair,
  - **Con:** groups more likely to pay back loans penalized

$$\mathbb{P}(x_i > t_\square | a_i = \square) = \mathbb{P}(x_i > t_\diamond | a_i = \diamond)$$

Discussion based on [Hardt, Price, Srebro '16]. See http://www.fatml.org for more resources.

Basics of Fair ML

You work at a bank that gives loans based on credit score.

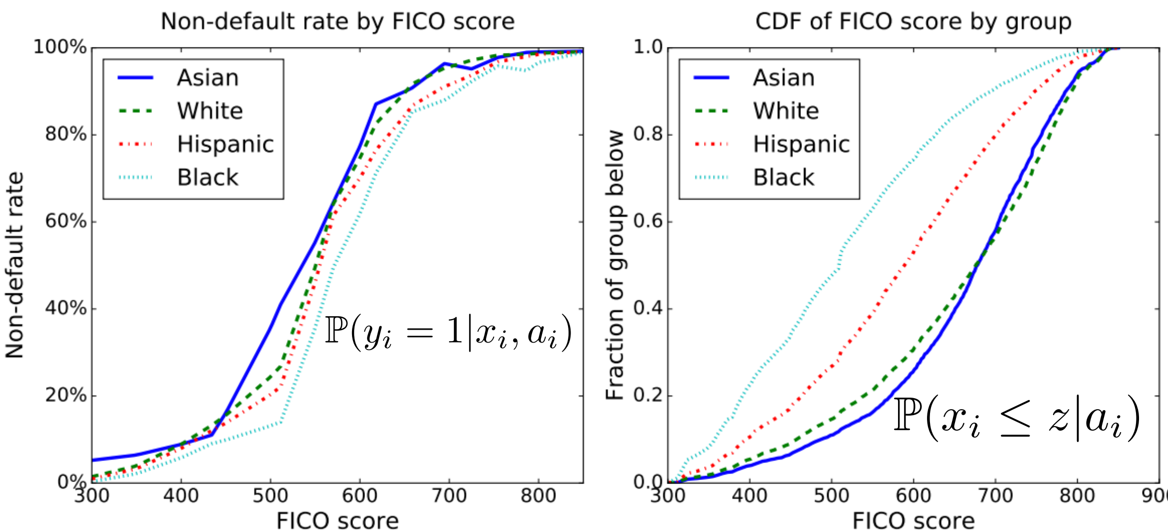
Boss tells you “make sure it doesn’t discriminate on race”

You have historical data: $\{(x_i, a_i, y_i)\}_{i=1}^n$

credit score $x_i \in \mathbb{R}$

paid back loan $y_i \in \{0, 1\}$

race $a_i \in \{\text{asian}, \text{white}, \text{hispanic}, \text{black}\}$



Discussion based on [Hardt, Price, Srebro ’16]. See http://www.fatml.org for more resources.

Basics of Fair ML

You work at a bank that gives loans based on credit score.
Boss tells you "make sure it doesn't discriminate on race"

You have historical data: $\{(x_i, a_i, y_i)\}_{i=1}^n$

credit score $x_i \in \mathbb{R}$

paid back loan $y_i \in \{0, 1\}$

race $a_i \in \{\text{asian}, \text{white}, \text{hispanic}, \text{black}\}$

$$P(x > t \mid y = 1) = \frac{\mathbb{P}(x > t, y = 1)}{\mathbb{P}(y = 1)}$$

- ***Equal opportunity***. proportion of those who would pay back loans equal
  - **Pro:** Bayes optimal if conditional distributions are the same,
  - **Con:** needs one class to be "good", another "bad"

TPR=equal

$$\mathbb{P}(x_i > t_\square \mid y_i = 1, a_i = \square) = \mathbb{P}(x_i > t_\diamond \mid y_i = 1, a_i = \diamond)$$

Discussion based on [Hardt, Price, Srebro '16]. See http://www.fatml.org for more resources.

Basics of Fair ML

You work at a bank that gives loans based on credit score.

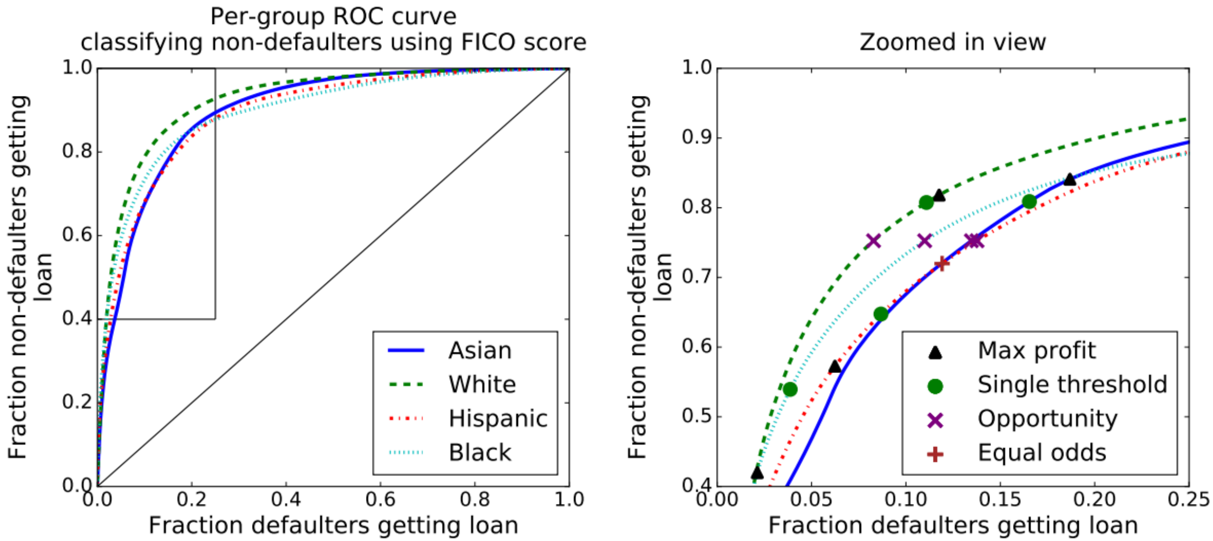
Boss tells you "make sure it doesn't discriminate on race"

You have historical data: $\{(x_i, a_i, y_i)\}_{i=1}^{n}$

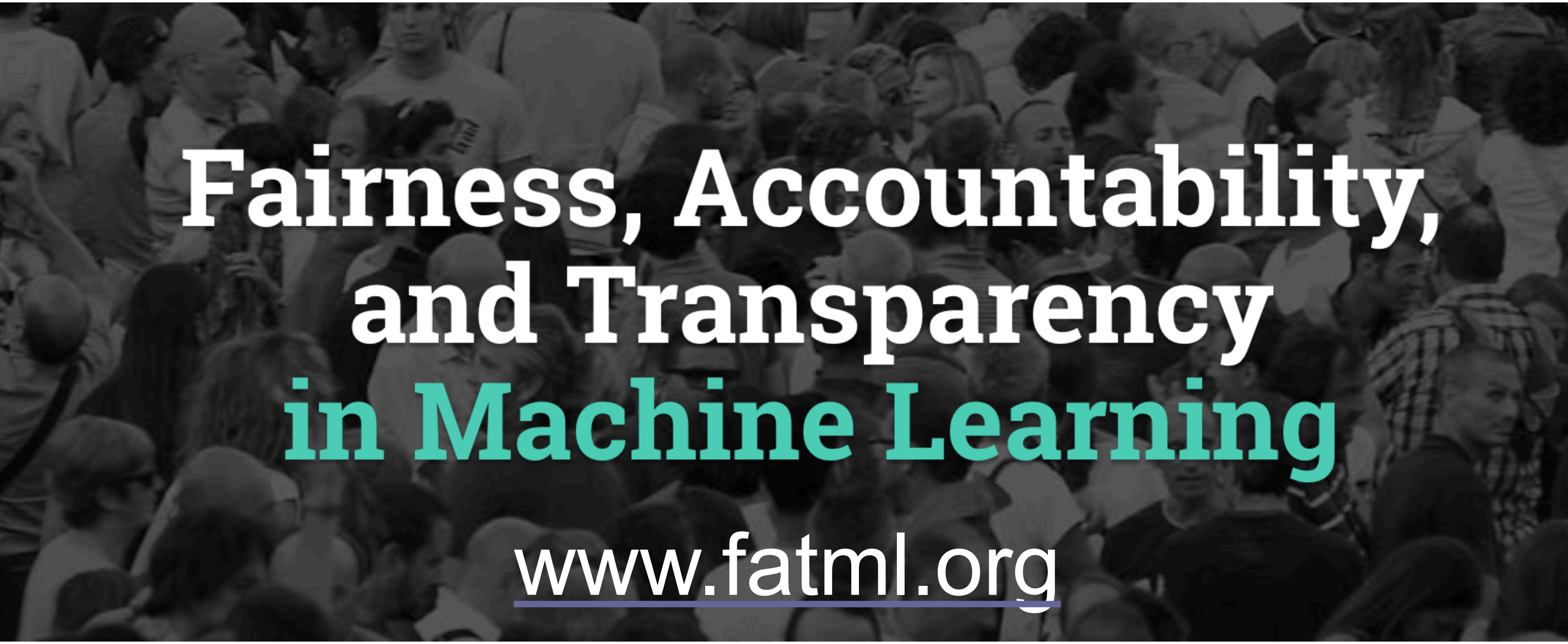
credit score $x_i \in \mathbb{R}$

paid back loan $y_i \in \{0, 1\}$

race $a_i \in \{\text{asian}, \text{white}, \text{hispanic}, \text{black}\}$



Discussion based on [Hardt, Price, Srebro '16]. See http://www.fatml.org for more resources.



Discussion based on [Hardt, Price, Srebro '16]. See http://www.fatml.org for more resources.

Trees

Machine Learning – CSE546

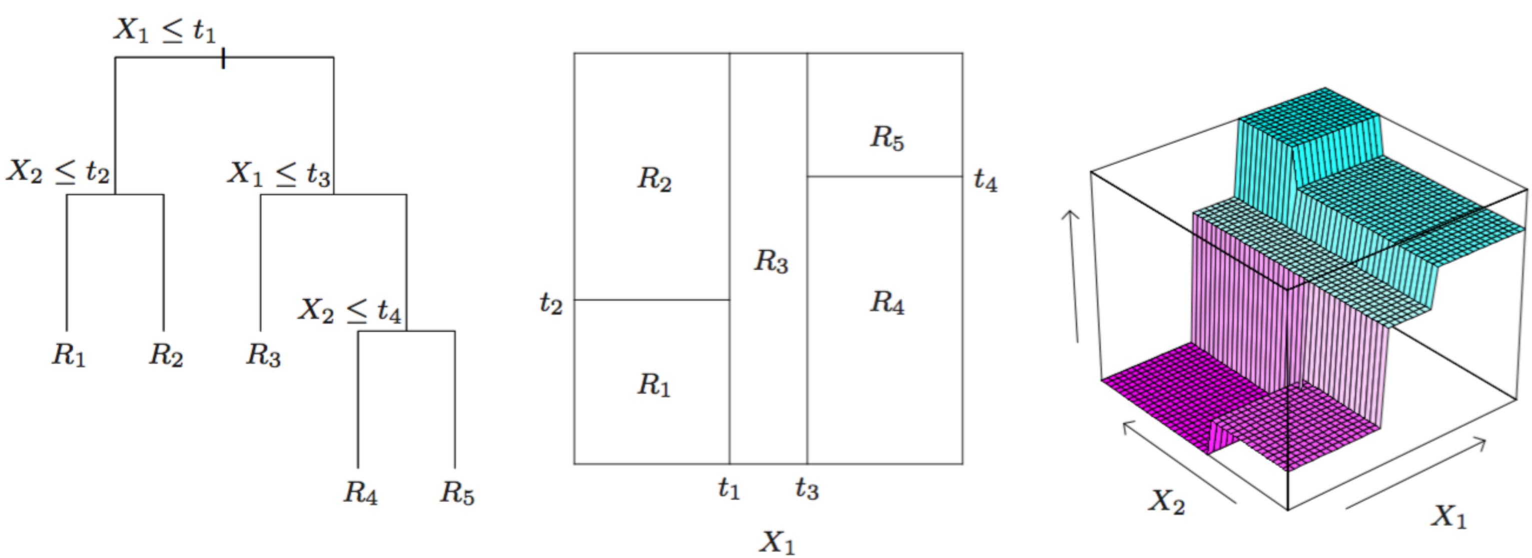
Kevin Jamieson
University of Washington

December 4, 2018

Trees

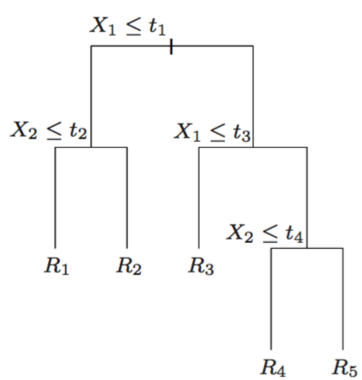
$$f(x) = \sum_{m=1}^{M} c_m I(x \in R_m).$$

Build a binary tree, splitting along axes



Learning decision trees

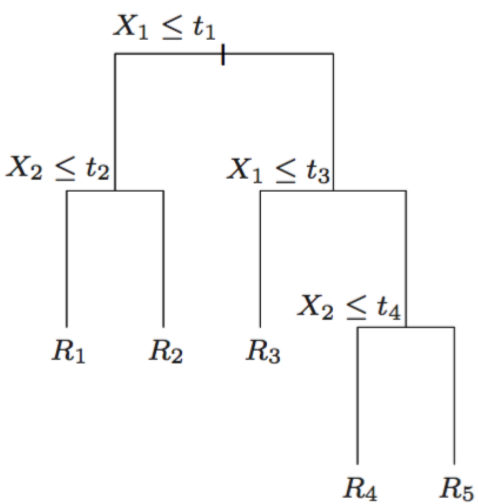
- Start from empty decision tree
- Split on **next best attribute (feature)**
  - Use, for example, information gain to select attribute
  - Split on $\arg\max_i IG(X_i) = \arg\max_i H(Y) - H(Y \mid X_i)$
- Recurse
- Prune



$$f(x) = \sum_{m=1}^{M} c_m I(x \in R_m).$$

Trees

$$f(x) = \sum_{m=1}^{M} c_m I(x \in R_m).$$



- Trees
  - **have low bias, high variance**
  - deal with categorial variables well
  - intuitive, interpretable
  - good software exists
  - Some theoretical guarantees

Random Forests

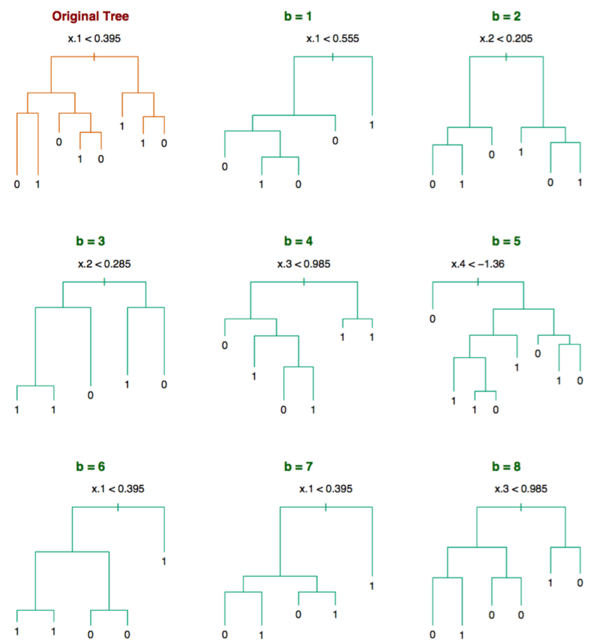
Machine Learning – CSE546
Kevin Jamieson
University of Washington

December 4, 2018

Random Forests

Tree methods have **low bias** but **high variance**.

One way to reduce variance is to construct a lot of “lightly correlated” trees and average them:



“Bagging:” Bootstrap aggregating

Random Forrests

Algorithm 15.1 *Random Forest for Regression or Classification.*

1. For $b = 1$ to B:

   (a) Draw a bootstrap sample $\mathbf{Z}^*$ of size N from the training data.

   (b) Grow a random-forest tree T_b to the bootstrapped data, by recursively repeating the following steps for each terminal node of the tree, until the minimum node size n_{min} is reached.

      i. Select m variables at random from the p variables.

      ii. Pick the best variable/split-point among the m.

      iii. Split the node into two daughter nodes.

2. Output the ensemble of trees $\{T_b\}_1^B$.

To make a prediction at a new point x:

Regression: $\hat{f}_{\mathrm{rf}}^B(x) = \frac{1}{B}\sum_{b=1}^B T_b(x)$.

m~p/3

Classification: Let $\hat{C}_b(x)$ be the class prediction of the bth random-forest tree. Then $\hat{C}_{\mathrm{rf}}^B(x) = \textit{majority vote}\ \{\hat{C}_b(x)\}_1^B$.

m~sqrt(p)

Random Forests

- Random Forests
  - **have low bias, low variance**
  - deal with categorial variables well
  - not that intuitive or interpretable
  - Notion of confidence estimates
  - good software exists
  - Some theoretical guarantees
  - **works well with default hyperparameters**

Boosting

Machine Learning – CSE546
Kevin Jamieson
University of Washington

December 4, 2018

Boosting

- 1988 Kearns and Valiant: "Can **weak learners** be combined to create a **strong learner?**"

Weak learner definition (informal):

An algorithm $\mathcal{A}$ is a *weak learner* for a hypothesis class $\mathcal{H}$ that maps $\mathcal{X}$ to $\{-1, 1\}$ if for all input distributions over $\mathcal{X}$ and $h \in \mathcal{H}$, we have that $\mathcal{A}$ correctly classifies h with error at most $1/2 - \gamma$

Boosting

- 1988 Kearns and Valiant: "Can **weak learners** be combined to create a **strong learner?**"

Weak learner definition (informal):

An algorithm $\mathcal{A}$ is a *weak learner* for a hypothesis class $\mathcal{H}$ that maps $\mathcal{X}$ to $\{-1, 1\}$ if for all input distributions over $\mathcal{X}$ and $h \in \mathcal{H}$, we have that $\mathcal{A}$ correctly classifies h with error at most $1/2 - \gamma$

- 1990 Robert Schapire: "Yup!"
- 1995 Schapire and Freund: "Practical for 0/1 loss" AdaBoost

Boosting

- 1988 Kearns and Valiant: "Can **weak learners** be combined to create a **strong learner?**"

Weak learner definition (informal):

An algorithm $\mathcal{A}$ is a *weak learner* for a hypothesis class $\mathcal{H}$ that maps $\mathcal{X}$ to $\{-1,1\}$ if for all input distributions over $\mathcal{X}$ and $h \in \mathcal{H}$, we have that $\mathcal{A}$ correctly classifies h with error at most $1/2 - \gamma$

- 1990 Robert Schapire: "Yup!"
- 1995 Schapire and Freund: "Practical for 0/1 loss" AdaBoost
- 2001 Friedman: "Practical for arbitrary losses"

Boosting

- 1988 Kearns and Valiant: "Can **weak learners** be combined to create a **strong learner?**"

Weak learner definition (informal):

An algorithm $\mathcal{A}$ is a *weak learner* for a hypothesis class $\mathcal{H}$ that maps $\mathcal{X}$ to $\{-1, 1\}$ if for all input distributions over $\mathcal{X}$ and $h \in \mathcal{H}$, we have that $\mathcal{A}$ correctly classifies h with error at most $1/2 - \gamma$

- 1990 Robert Schapire: "Yup!"
- 1995 Schapire and Freund: "Practical for 0/1 loss" AdaBoost
- 2001 Friedman: "Practical for arbitrary losses"
- 2014 Tianqi Chen: "Scale it up!" XGBoost

Boosting and Additive Models

Machine Learning – CSE546
Kevin Jamieson
University of Washington

December 4, 2018

Additive models

- Consider the first algorithm we used to get good classification for MNIST. Given: $\{(x_i, y_i)\}_{i=1}^n \quad x_i \in \mathbb{R}^d, y_i \in \{-1, 1\}$
- Generate **random** functions: $\phi_t : \mathbb{R}^d \to \mathbb{R} \quad t = 1, \dots, p$
- Learn some weights: $$\widehat{w} = \arg\min_w \sum_{i=1}^n \text{Loss}\left(y_i, \sum_{t=1}^p w_t \phi_t(x_i) \right)$$
- Classify new data: $$f(x) = \text{sign}\left(\sum_{t=1}^p \widehat{w}_t \phi_t(x) \right)$$

Additive models

- Consider the first algorithm we used to get good classification for MNIST. Given: $\{(x_i, y_i)\}_{i=1}^n \quad x_i \in \mathbb{R}^d, y_i \in \{-1, 1\}$
- Generate **random** functions: $\phi_t : \mathbb{R}^d \to \mathbb{R} \quad t = 1, \dots, p$
- Learn some weights: $$\widehat{w} = \arg\min_w \sum_{i=1}^n \text{Loss}\left(y_i, \sum_{t=1}^p w_t \phi_t(x_i) \right)$$
- Classify new data: $$f(x) = \text{sign}\left(\sum_{t=1}^p \widehat{w}_t \phi_t(x) \right)$$

An interpretation:
Each $\phi_t(x)$ is a classification rule that we are assigning some weight $\widehat{w}_t$

Additive models

- Consider the first algorithm we used to get good classification for MNIST. Given: $\{(x_i, y_i)\}_{i=1}^n \quad x_i \in \mathbb{R}^d, y_i \in \{-1, 1\}$
- Generate **random** functions: $\phi_t : \mathbb{R}^d \to \mathbb{R} \quad t = 1, \dots, p$
- Learn some weights: $$\widehat{w} = \arg\min_w \sum_{i=1}^n \text{Loss}\left(y_i, \sum_{t=1}^p w_t \phi_t(x_i)\right)$$
- Classify new data: $$f(x) = \text{sign}\left(\sum_{t=1}^p \widehat{w}_t \phi_t(x)\right)$$

An interpretation:
Each $\phi_t(x)$ is a classification rule that we are assigning some weight $\widehat{w}_t$

$$\widehat{w}, \widehat{\phi}_1, \dots, \widehat{\phi}_t = \arg\min_{w, \phi_1, \dots, \phi_p} \sum_{i=1}^n \text{Loss}\left(y_i, \sum_{t=1}^p w_t \phi_t(x_i)\right)$$

is in general computationally hard

Forward Stagewise Additive models

$b(x, \gamma)$ is a function with parameters γ

Examples: $b(x, \gamma) = \dfrac{1}{1 + e^{-\gamma^T x}}$

$b(x, \gamma) = \gamma_1 \mathbf{1}\{x_3 \leq \gamma_2\}$

Algorithm 10.2 *Forward Stagewise Additive Modeling.*

1. Initialize $f_0(x) = 0$.
2. For $m = 1$ to M:
   (a) Compute
   $$(\beta_m, \gamma_m) = \arg\min_{\beta,\gamma} \sum_{i=1}^{N} L(y_i, f_{m-1}(x_i) + \beta b(x_i; \gamma)).$$
   (b) Set $f_m(x) = f_{m-1}(x) + \beta_m b(x; \gamma_m)$.

Idea: greedily add one function at a time

Forward Stagewise Additive models

$b(x, \gamma)$ is a function with parameters γ

Examples: $b(x, \gamma) = \dfrac{1}{1 + e^{-\gamma^T x}}$

$b(x, \gamma) = \gamma_1 \mathbf{1}\{x_3 \leq \gamma_2\}$

Algorithm 10.2 *Forward Stagewise Additive Modeling.*

1. Initialize $f_0(x) = 0$.
2. For $m = 1$ to M:
   (a) Compute
   $$(\beta_m, \gamma_m) = \arg\min_{\beta, \gamma} \sum_{i=1}^{N} L(y_i, f_{m-1}(x_i) + \beta b(x_i; \gamma)).$$
   (b) Set $f_m(x) = f_{m-1}(x) + \beta_m b(x; \gamma_m)$.

Idea: greedily add one function at a time

AdaBoost: $b(x, \gamma)$: classifiers to $\{-1, 1\}$

$$L(y, f(x)) = \exp(-y f(x))$$

Forward Stagewise Additive models

$b(x,\gamma)$ is a function with parameters γ

Examples: $b(x,\gamma) = \dfrac{1}{1+e^{-\gamma^T x}}$

$b(x,\gamma) = \gamma_1 \mathbf{1}\{x_3 \leq \gamma_2\}$

Algorithm 10.2 *Forward Stagewise Additive Modeling.*

1. Initialize $f_0(x) = 0$.
2. For $m = 1$ to M:
   (a) Compute
   $$(\beta_m, \gamma_m) = \arg\min_{\beta,\gamma} \sum_{i=1}^{N} L(y_i, f_{m-1}(x_i) + \beta b(x_i;\gamma)).$$
   (b) Set $f_m(x) = f_{m-1}(x) + \beta_m b(x;\gamma_m)$.

Idea: greedily add one function at a time

Boosted Regression Trees:

$L(y, f(x)) = (y - f(x))^2$

$b(x,\gamma)$: regression trees

Forward Stagewise Additive models

$b(x,\gamma)$ is a function with parameters γ

Examples: $b(x,\gamma) = \dfrac{1}{1+e^{-\gamma^T x}}$

$b(x,\gamma) = \gamma_1 \mathbf{1}\{x_3 \leq \gamma_2\}$

Algorithm 10.2 *Forward Stagewise Additive Modeling.*

1. Initialize $f_0(x) = 0$.
2. For $m = 1$ to M:

   (a) Compute

   $$(\beta_m, \gamma_m) = \arg\min_{\beta,\gamma} \sum_{i=1}^{N} L(y_i, f_{m-1}(x_i) + \beta b(x_i;\gamma)).$$

   (b) Set $f_m(x) = f_{m-1}(x) + \beta_m b(x;\gamma_m)$.

Idea: greedily add one function at a time

Boosted Regression Trees: $L(y, f(x)) = (y - f(x))^2$

$$\begin{aligned} L(y_i, f_{m-1}(x_i) + \beta b(x_i;\gamma)) &= (y_i - f_{m-1}(x_i) - \beta b(x_i;\gamma))^2 \\ &= (r_{im} - \beta b(x_i;\gamma))^2, \qquad r_{im} = y_i - f_{m-1}(x_i) \end{aligned}$$

Efficient: No harder than learning regression trees!

Forward Stagewise Additive models

$b(x, \gamma)$ is a function with parameters γ

Examples: $b(x, \gamma) = \dfrac{1}{1 + e^{-\gamma^T x}}$

$b(x, \gamma) = \gamma_1 \mathbf{1}\{x_3 \leq \gamma_2\}$

Algorithm 10.2 *Forward Stagewise Additive Modeling.*

1. Initialize $f_0(x) = 0$.
2. For $m = 1$ to M:
   (a) Compute
   $$\boxed{(\beta_m, \gamma_m) = \arg\min_{\beta,\gamma} \sum_{i=1}^{N} L(y_i, f_{m-1}(x_i) + \beta b(x_i; \gamma)).}$$
   (b) Set $f_m(x) = f_{m-1}(x) + \beta_m b(x; \gamma_m)$.

Idea: greedily add one function at a time

Boosted Logistic Trees: $L(y, f(x)) = y \log(f(x)) + (1-y)\log(1 - f(x))$

$b(x, \gamma)$: regression trees

Computationally hard to update

Gradient Boosting

Least squares, exponential loss easy. But what about cross entropy? Huber?

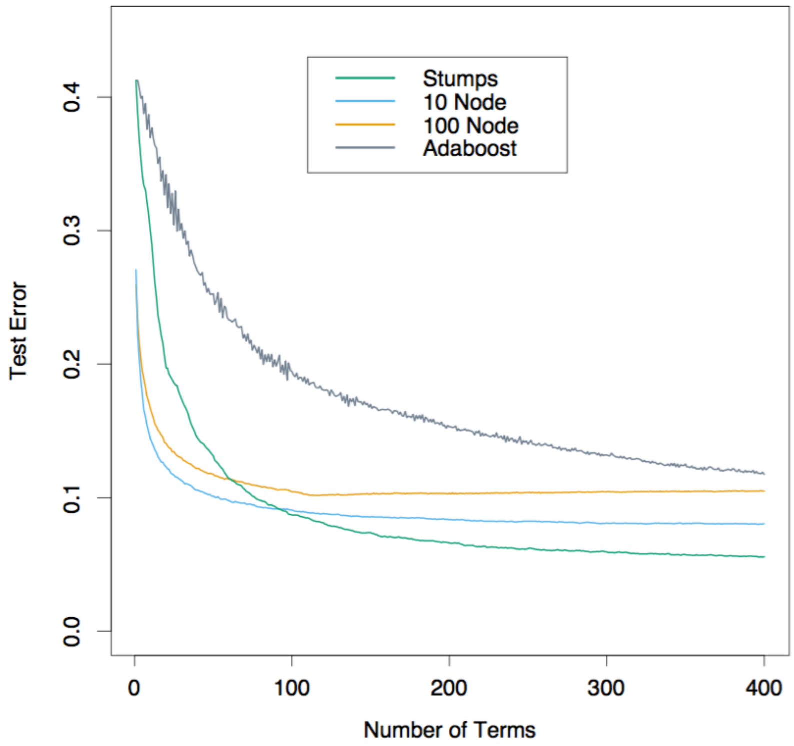
Algorithm 10.3 *Gradient Tree Boosting Algorithm.*

1. Initialize $f_0(x) = \arg\min_\gamma \sum_{i=1}^N L(y_i, \gamma)$.
2. For $m = 1$ to M:
   (a) For $i = 1, 2, \ldots, N$ compute
   $$r_{im} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f=f_{m-1}}.$$
   (b) Fit a regression tree to the targets r_{im} giving terminal regions $R_{jm},\ j = 1, 2, \ldots, J_m$.
   (c) For $j = 1, 2, \ldots, J_m$ compute
   $$\gamma_{jm} = \arg\min_\gamma \sum_{x_i \in R_{jm}} L\left(y_i, f_{m-1}(x_i) + \gamma\right).$$
   (d) Update $f_m(x) = f_{m-1}(x) + \sum_{j=1}^{J_m} \gamma_{jm} I(x \in R_{jm})$.
3. Output $\hat{f}(x) = f_M(x)$.

LS fit regression tree to n-dimensional gradient, take a step in that direction

Gradient Boosting

Least squares, 0/1 loss easy. But what about cross entropy? Huber?



AdaBoost uses 0/1 loss,
all other trees are minimizing
binomial deviance

Additive models

- Boosting is popular at parties: Invented by theorists, heavily adopted by practitioners.

Additive models

- Boosting is popular at parties: Invented by theorists, heavily adopted by practitioners.
- Computationally efficient with “weak” learners. But can also use trees! Boosting can scale.
- Kind of like sparsity?

Additive models

- Boosting is popular at parties: Invented by theorists, heavily adopted by practitioners.
- Computationally efficient with “weak” learners. But can also use trees! Boosting can scale.
- Kind of like sparsity?
- Gradient boosting generalization with good software packages (e.g., *XGBoost*). Effective on Kaggle
- Robust to overfitting and can be dealt with with “shrinkage” and “sampling”

Bagging versus Boosting

- Bagging *averages* many **low-bias**, **lightly dependent** classifiers to reduce the variance
- Boosting *learns* linear combination of **high-bias**, **highly dependent** classifiers to reduce error
- Empirically, boosting appears to outperform bagging

Which algorithm do I use?

TABLE 10.1. *Some characteristics of different learning methods. Key:* ▲ *= good,* ◆ *=fair, and* ▼ *=poor.*

Characteristic	Neural Nets	SVM	Trees	MARS	k-NN, Kernels
Natural handling of data of "mixed" type	▼	▼	▲	▲	▼
Handling of missing values	▼	▼	▲	▲	▲
Robustness to outliers in input space	▼	▼	▲	▼	▲
Insensitive to monotone transformations of inputs	▼	▼	▲	▼	▼
Computational scalability (large N)	▼	▼	▲	▲	▼
Ability to deal with irrelevant inputs	▼	▼	▲	▲	▼
Ability to extract linear combinations of features	▲	▲	▼	▼	◆
Interpretability	▼	▼	◆	▲	▼
Predictive power	▲	▲	▼	◆	▲

$\mathcal{X}$ - sample space | P_{XY} distribution on $\mathcal{X} \times \mathcal{Y}$

$\mathcal{Y} : \{0, 1\}$

Given $f : \mathcal{X} \to \{0, 1\}$, sample $\{(x_i, y_i)\}_{i=1}^{n}$

Define

Empirical Loss: $\hat{R}_n(f) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}\{f(x_i) \neq y_i\}$

True Loss: $R(f) = \mathbb{E}_{P_{XY}}[f(X) \neq Y]$

Empirical Risk Minimization

$$\hat{f} = \min_{f \in \mathcal{H}} \hat{R}_n(f)$$

logistic, linear, trees, NN architecture

→ How well does $\hat{f}$ generalize?

PAC Learning

all functions $\mathcal{X} \to \{0,1\}$

A hypothesis class $\mathcal{H} \subseteq \mathcal{Y}^{\mathcal{X}}$ is PAC-learnable if there exists a function $m_{\mathcal{H}} : [0,1]^2 \to \mathbb{N}$, and an algorithm $\mathcal{A}$, such that: For every $\varepsilon, \delta \in (0,1)$ and every P_{XY}, when running $\mathcal{A}$ on $m \geq m_H(\varepsilon, \delta)$ i.i.d examples from P_{XY} the algorithm returns $h \in \mathcal{H}$ s.t. w/ probability $> 1-\delta$

$$R(h) \leq \min_{h' \in \mathcal{H}} R(h') + \varepsilon$$

generalization