Political Methodology III: Model Based Inference

Justin Grimmer

Associate Professor
Department of Political Science
Stanford University

May 22, 2017

Supervised Learning: Ensemble Learning

1) Task

Supervised Learning: Ensemble Learning

1) Task
    - Learn a Conditional Expectation Function

Supervised Learning: Ensemble Learning

1) Task
    - Learn a Conditional Expectation Function
2) Objective function

Supervised Learning: Ensemble Learning

1) Task
   - Learn a Conditional Expectation Function
2) Objective function

$$\Pr(Y_i = C_k|\boldsymbol{x}_i) \quad = \quad f(\boldsymbol{x}_i, \boldsymbol{\beta}, \boldsymbol{\lambda})$$

Supervised Learning: Ensemble Learning

1) Task
   - Learn a Conditional Expectation Function
2) Objective function

$$\Pr(Y_i = C_k|\boldsymbol{x}_i) \quad = \quad f(\boldsymbol{x}_i, \boldsymbol{\beta}, \boldsymbol{\lambda})$$

   - "Coefficients" $\boldsymbol{\beta}$

Supervised Learning: Ensemble Learning

1) Task
   - Learn a Conditional Expectation Function
2) Objective function

$$\Pr(Y_i = C_k|\boldsymbol{x}_i) \quad = \quad f(\boldsymbol{x}_i, \boldsymbol{\beta}, \boldsymbol{\lambda})$$

   - "Coefficients" $\boldsymbol{\beta}$
   - Tuning parameters $\boldsymbol{\lambda}$

Supervised Learning: Ensemble Learning

1) Task
   - Learn a Conditional Expectation Function
2) Objective function

$$\Pr(Y_i = C_k|\boldsymbol{x}_i) \quad = \quad f(\boldsymbol{x}_i, \boldsymbol{\beta}, \boldsymbol{\lambda})$$

   - "Coefficients" $\boldsymbol{\beta}$
   - Tuning parameters $\boldsymbol{\lambda}$
   - Functional form f

Supervised Learning: Ensemble Learning

1) Task
   - Learn a Conditional Expectation Function
2) Objective function

$$\Pr(Y_i = C_k | \boldsymbol{x}_i) \quad = \quad f(\boldsymbol{x}_i, \boldsymbol{\beta}, \boldsymbol{\lambda})$$

- "Coefficients" $\boldsymbol{\beta}$
- Tuning parameters $\boldsymbol{\lambda}$
- Functional form f
- Necessarily requires consequential assumptions

Supervised Learning: Ensemble Learning

1) Task
   - Learn a Conditional Expectation Function
2) Objective function

$$\Pr(Y_i = C_k | \boldsymbol{x}_i) \quad = \quad f(\boldsymbol{x}_i, \boldsymbol{\beta}, \boldsymbol{\lambda})$$

- "Coefficients" $\boldsymbol{\beta}$
- Tuning parameters $\boldsymbol{\lambda}$
- Functional form f
- Necessarily requires consequential assumptions
- Ensembles: aggregate models to increase performance

Supervised Learning: Ensemble Learning

1) Task
   - Learn a Conditional Expectation Function
2) Objective function

$$\Pr(Y_i = C_k | \boldsymbol{x}_i) \quad = \quad f(\boldsymbol{x}_i, \boldsymbol{\beta}, \boldsymbol{\lambda})$$

   - "Coefficients" $\boldsymbol{\beta}$
   - Tuning parameters $\boldsymbol{\lambda}$
   - Functional form f
   - Necessarily requires consequential assumptions
   - Ensembles: aggregate models to increase performance
3) Optimization

Supervised Learning: Ensemble Learning

1) Task
   - Learn a Conditional Expectation Function
2) Objective function

$$\Pr(Y_i = C_k | \boldsymbol{x}_i) \quad = \quad f(\boldsymbol{x}_i, \boldsymbol{\beta}, \boldsymbol{\lambda})$$

   - "Coefficients" $\boldsymbol{\beta}$
   - Tuning parameters $\boldsymbol{\lambda}$
   - Functional form f
   - Necessarily requires consequential assumptions
   - Ensembles: aggregate models to increase performance
3) Optimization
   - Optimization of individual classifiers$\rightsquigarrow$ methods already discussed + other methods

Supervised Learning: Ensemble Learning

1) Task
   - Learn a Conditional Expectation Function
2) Objective function

$$\Pr(Y_i = C_k | \boldsymbol{x}_i) \quad = \quad f(\boldsymbol{x}_i, \boldsymbol{\beta}, \boldsymbol{\lambda})$$

   - "Coefficients" $\boldsymbol{\beta}$
   - Tuning parameters $\boldsymbol{\lambda}$
   - Functional form f
   - Necessarily requires consequential assumptions
   - Ensembles: aggregate models to increase performance
3) Optimization
   - Optimization of individual classifiers$\rightsquigarrow$ methods already discussed + other methods
   - Determination of weights on models$\rightsquigarrow$ equal (bagging), out of sample performance via cross validation (super learning)

Supervised Learning: Ensemble Learning

1) Task
   - Learn a Conditional Expectation Function
2) Objective function

$$\Pr(Y_i = C_k|\boldsymbol{x}_i) \quad = \quad f(\boldsymbol{x}_i, \boldsymbol{\beta}, \boldsymbol{\lambda})$$

   - "Coefficients" $\boldsymbol{\beta}$
   - Tuning parameters $\boldsymbol{\lambda}$
   - Functional form f
   - Necessarily requires consequential assumptions
   - Ensembles: aggregate models to increase performance
3) Optimization
   - Optimization of individual classifiers$\rightsquigarrow$ methods already discussed + other methods
   - Determination of weights on models$\rightsquigarrow$ equal (bagging), out of sample performance via cross validation (super learning)
4) Validation

Supervised Learning: Ensemble Learning

1) Task
   - Learn a Conditional Expectation Function
2) Objective function

$$\Pr(Y_i = C_k|\boldsymbol{x}_i) \quad = \quad f(\boldsymbol{x}_i, \boldsymbol{\beta}, \boldsymbol{\lambda})$$

   - "Coefficients" $\boldsymbol{\beta}$
   - Tuning parameters $\boldsymbol{\lambda}$
   - Functional form f
   - Necessarily requires consequential assumptions
   - Ensembles: aggregate models to increase performance
3) Optimization
   - Optimization of individual classifiers$\rightsquigarrow$ methods already discussed + other methods
   - Determination of weights on models$\rightsquigarrow$ equal (bagging), out of sample performance via cross validation (super learning)
4) Validation
   - Out of sample predictive performance

Ensemble Learning: Intuition

Heuristic (upon which we'll improve):

Ensemble Learning: Intuition

Heuristic (upon which we'll improve): if regressions are accurate and diverse$\rightarrow$ ensemble methods improve

Ensemble Learning: Intuition

Heuristic (upon which we'll improve): if regressions are accurate and diverse$\rightarrow$ ensemble methods improve

Intuition:

Ensemble Learning: Intuition

Heuristic (upon which we'll improve): if regressions are accurate and diverse$\rightarrow$ ensemble methods improve

Intuition:

- Classify observations into two categories (Category 1, Category 2).

Ensemble Learning: Intuition

Heuristic (upon which we'll improve): if regressions are accurate and diverse$\rightarrow$ ensemble methods improve

Intuition:

- Classify observations into two categories (Category 1, Category 2).
- True labels: evenly distributed across two categories

Ensemble Learning: Intuition

Heuristic (upon which we'll improve): if regressions are accurate and diverse$\rightarrow$ ensemble methods improve

Intuition:

- Classify observations into two categories (Category 1, Category 2).
- True labels: evenly distributed across two categories
- Three classifiers with 75% accuracy, but independent

Ensemble Learning: Intuition

Heuristic (upon which we'll improve): if regressions are accurate and diverse$\rightarrow$ ensemble methods improve

Intuition:

- Classify observations into two categories (Category 1, Category 2).
- True labels: evenly distributed across two categories
- Three classifiers with 75% accuracy, but independent
- Implement majority voting rule

Ensemble Learning: Intuition

Heuristic (upon which we'll improve): if regressions are accurate and diverse$\rightarrow$ ensemble methods improve

Intuition:

- Classify observations into two categories (Category 1, Category 2).
- True labels: evenly distributed across two categories
- Three classifiers with 75% accuracy, but independent
- Implement majority voting rule

$$\Pr(\text{Correct Guess}|\text{Votes})$$

Ensemble Learning: Intuition

Heuristic (upon which we'll improve): if regressions are accurate and diverse$\rightarrow$ ensemble methods improve

Intuition:

- Classify observations into two categories (Category 1, Category 2).
- True labels: evenly distributed across two categories
- Three classifiers with 75% accuracy, but independent
- Implement majority voting rule

$$\Pr(\text{Correct Guess}|\text{Votes}) \quad = \quad \Pr(3 \text{ correct}) + \Pr(2 \text{ correct})$$

Ensemble Learning: Intuition

Heuristic (upon which we'll improve): if regressions are accurate and diverse$\rightarrow$ ensemble methods improve

Intuition:

- Classify observations into two categories (Category 1, Category 2).
- True labels: evenly distributed across two categories
- Three classifiers with 75% accuracy, but independent
- Implement majority voting rule

$$\begin{aligned} \Pr(\text{Correct Guess}|\text{Votes}) &= \Pr(3 \text{ correct}) + \Pr(2 \text{ correct}) \\ &= 0.75^3 + 3 \times (0.75^2 \times 0.25) \end{aligned}$$

Ensemble Learning: Intuition

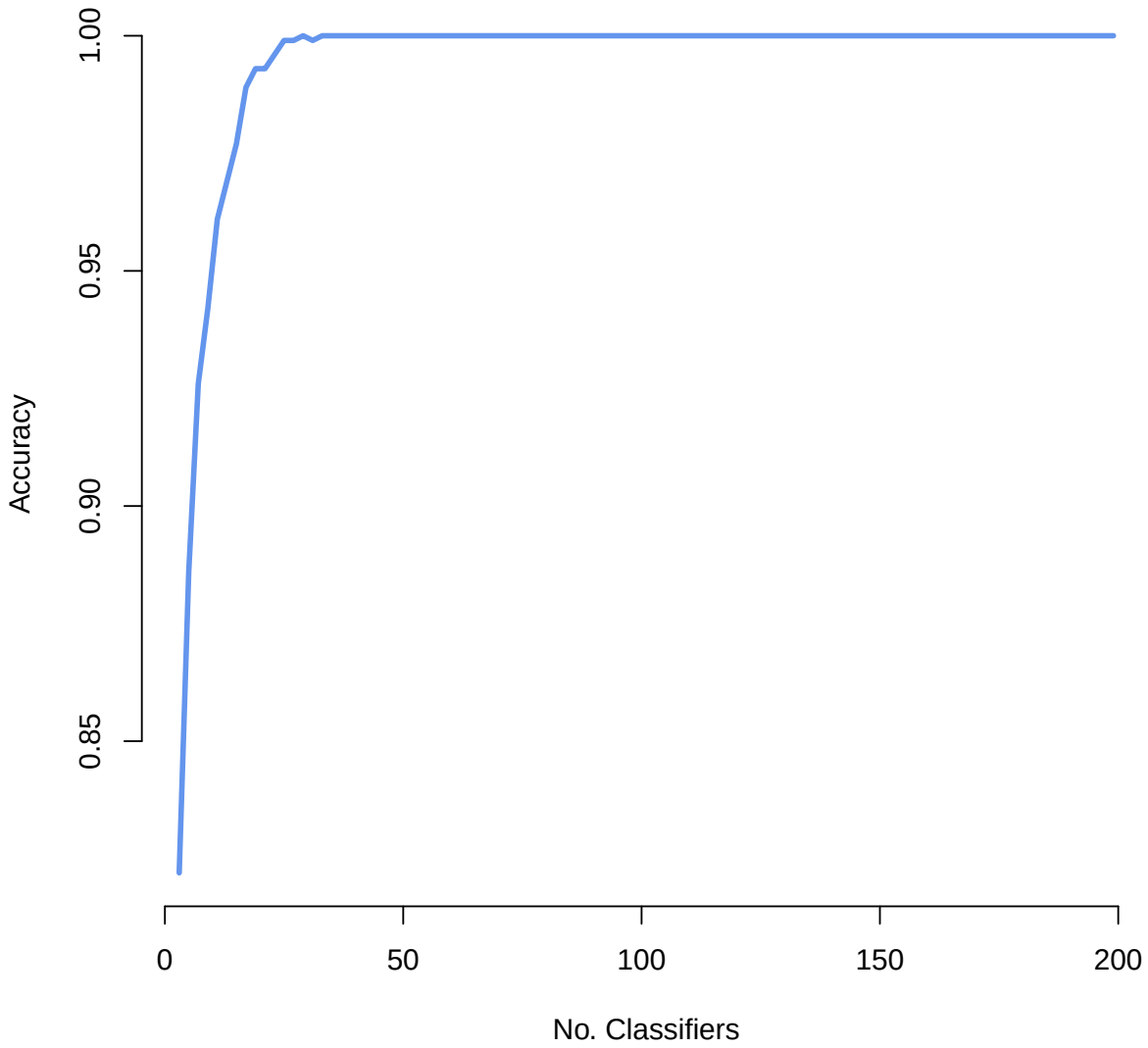
Heuristic (upon which we'll improve): if regressions are accurate and diverse$\rightarrow$ ensemble methods improve

Intuition:

- Classify observations into two categories (Category 1, Category 2).
- True labels: evenly distributed across two categories
- Three classifiers with 75% accuracy, but independent
- Implement majority voting rule

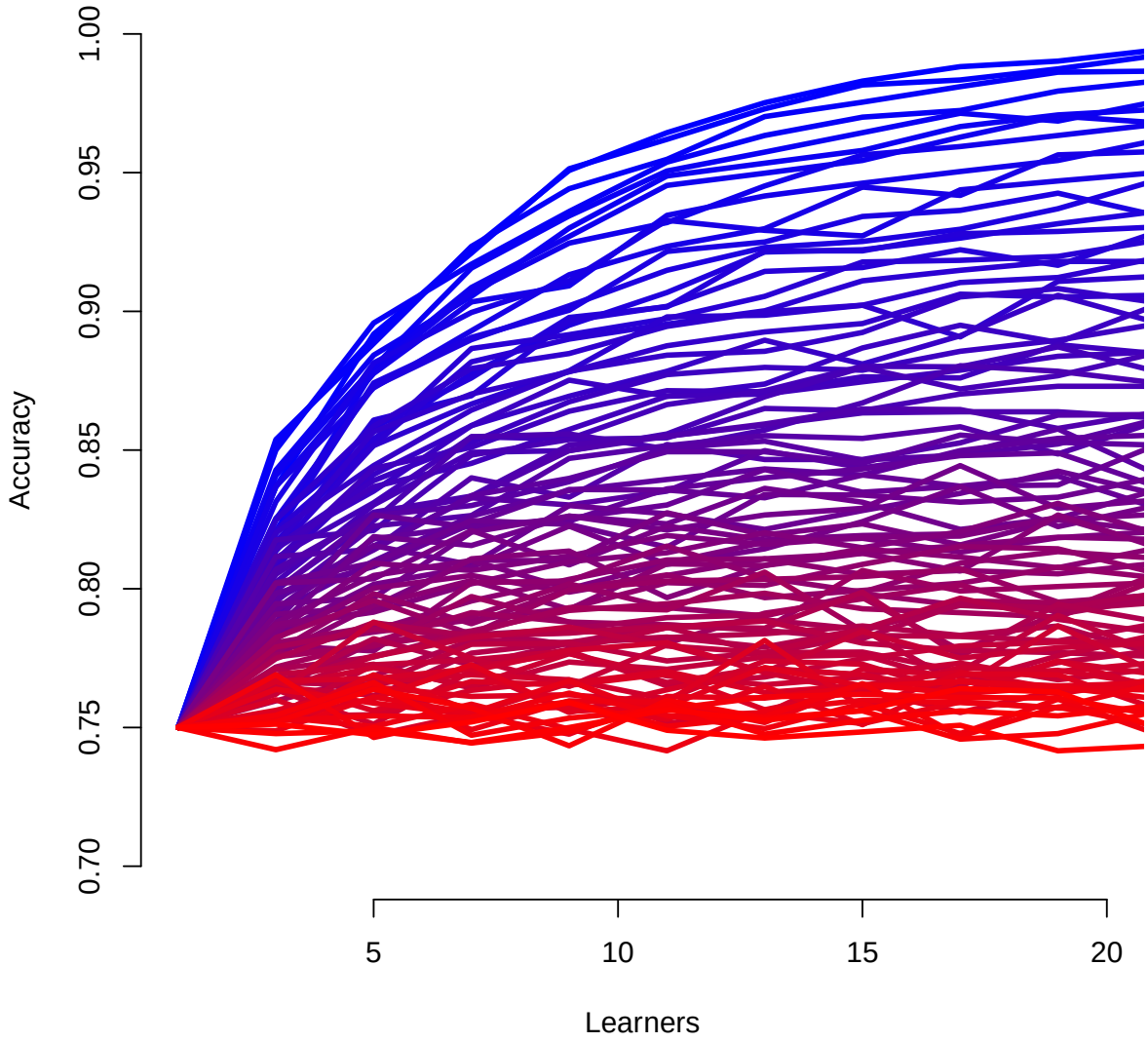
$$\begin{aligned} \Pr(\text{Correct Guess}|\text{Votes}) &= \Pr(3 \text{ correct}) + \Pr(2 \text{ correct}) \\ &= 0.75^3 + 3 \times (0.75^2 \times 0.25) \\ &= 0.844 \end{aligned}$$

Ensemble Learning: Intuition



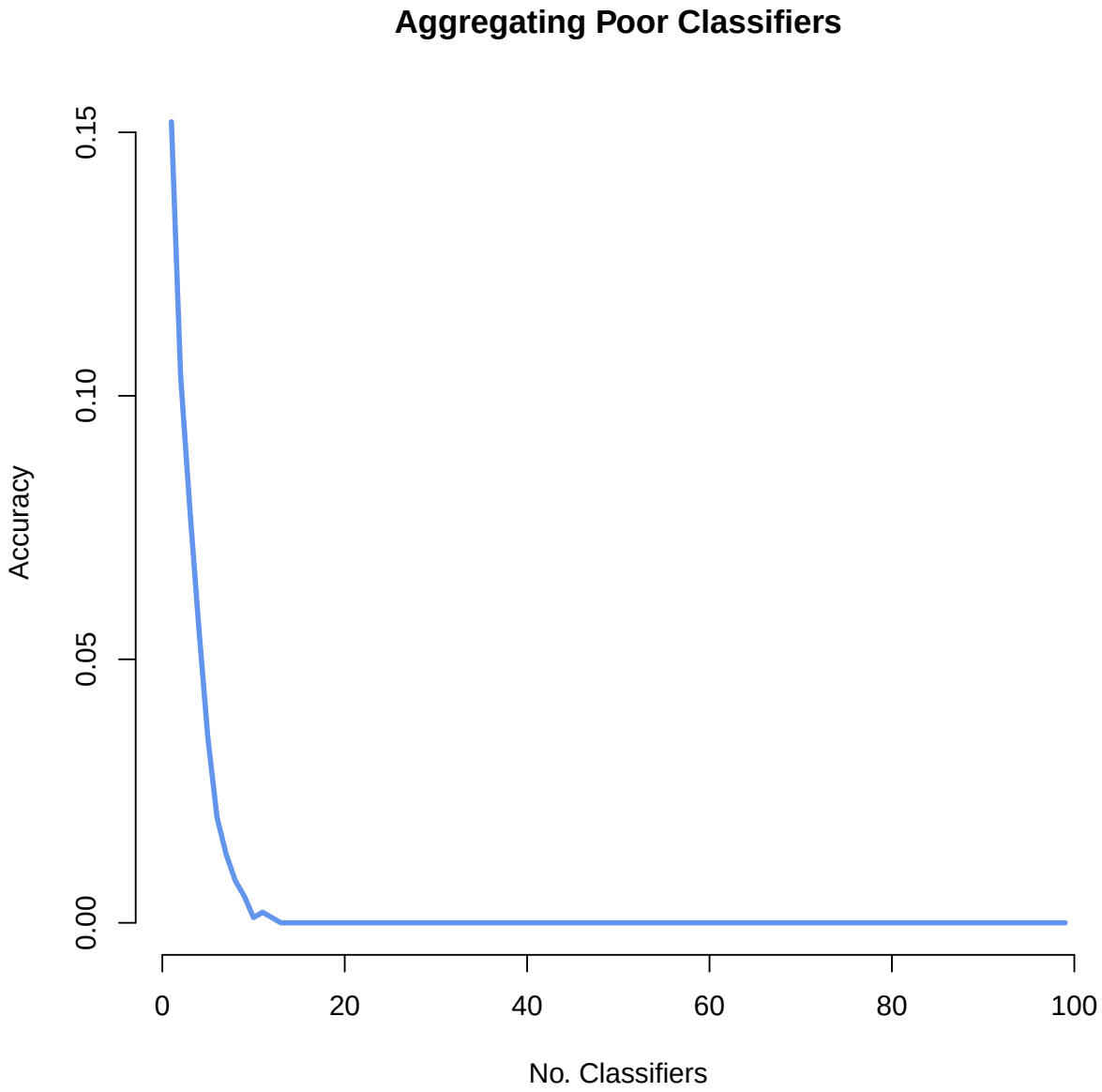
Ensemble Learning: Intuition

Diverse and Accurate matter.



Ensemble Learning: Intuition

Diverse and Accurate matter.



Wisdom of the Crowds:

Goal: estimate an observation's category $\rightsquigarrow Y \in \{0, 1\}$

Wisdom of the Crowds:

Goal: estimate an observation's category$\rightsquigarrow Y \in \{0, 1\}$
Classifiers: (suppose) a sequence of identically distributed (not necessarily independent) random variables.

Wisdom of the Crowds:

Goal: estimate an observation's category$\rightsquigarrow Y \in \{0, 1\}$
Classifiers: (suppose) a sequence of identically distributed (not necessarily independent) random variables.
Suppose $Y = 1$

Wisdom of the Crowds:

Goal: estimate an observation's category$\rightsquigarrow Y \in \{0, 1\}$
Classifiers: (suppose) a sequence of identically distributed (not necessarily independent) random variables.
Suppose $Y = 1$
Guess from classifer m is B_m with $\Pr(B_i = 1) = p > 0.5$.

Wisdom of the Crowds:

Goal: estimate an observation's category$\rightsquigarrow Y \in \{0, 1\}$
Classifiers: (suppose) a sequence of identically distributed (not necessarily independent) random variables.
Suppose $Y = 1$
Guess from classifer m is B_m with $\Pr(B_i = 1) = p > 0.5$.

$$\bar{B} = \sum_{m=1}^{M} \frac{B_m}{M}$$

Wisdom of the Crowds:

Goal: estimate an observation's category$\rightsquigarrow Y \in \{0, 1\}$
Classifiers: (suppose) a sequence of identically distributed (not necessarily independent) random variables.
Suppose $Y = 1$
Guess from classifer m is B_m with $\Pr(B_i = 1) = p > 0.5$.

$$\bar{B} = \sum_{m=1}^{M} \frac{B_m}{M}$$

Wisdom of crows (Condorcet Jury Theorem)

Wisdom of the Crowds:

Goal: estimate an observation's category$\rightsquigarrow Y \in \{0, 1\}$
Classifiers: (suppose) a sequence of identically distributed (not necessarily independent) random variables.
Suppose $Y = 1$
Guess from classifer m is B_m with $\Pr(B_i = 1) = p > 0.5$.

$$\bar{B} = \sum_{m=1}^{M} \frac{B_m}{M}$$

Wisdom of crows (Condorcet Jury Theorem)

$$\lim_{M \to \infty} P(\bar{B} > 0.5) = 1$$

Wisdom of the Crowds

Suppose B_m have variance σ^2 and pairwise correlation ρ.

Wisdom of the Crowds

Suppose B_m have variance σ^2 and pairwise correlation ρ.
Then,

Wisdom of the Crowds

Suppose B_m have variance σ^2 and pairwise correlation ρ. Then,

$$\text{var}(\bar{B})$$

Wisdom of the Crowds

Suppose B_m have variance σ^2 and pairwise correlation ρ. Then,

$$\text{var}(\bar{B}) = \text{var}\left(\sum_{i=1}^{M} \frac{B_i}{M}\right)$$

Wisdom of the Crowds

Suppose B_m have variance σ^2 and pairwise correlation ρ. Then,

$$\begin{aligned}
\text{var}(\bar{B}) &= \text{var}\left(\sum_{i=1}^{M} \frac{B_i}{M}\right) \\
&= \frac{1}{M^2}\sum_{i=1}^{M} \text{var}\left(B_i\right) + \frac{2}{M^2}\sum_{i<j} \text{cov}(B_i, B_j)
\end{aligned}$$

Wisdom of the Crowds

Suppose B_m have variance σ^2 and pairwise correlation ρ. Then,

$$\begin{aligned}
\text{var}(\bar{B}) &= \text{var}\left(\sum_{i=1}^{M} \frac{B_i}{M}\right) \\
&= \frac{1}{M^2}\sum_{i=1}^{M} \text{var}\left(B_i\right) + \frac{2}{M^2}\sum_{i<j} \text{cov}(B_i, B_j) \\
&= \frac{M\sigma^2}{M^2} + \frac{2}{M^2}\rho\sigma^2\binom{M}{2}
\end{aligned}$$

Wisdom of the Crowds

Suppose B_m have variance σ^2 and pairwise correlation ρ. Then,

$$\begin{aligned}
\text{var}(\bar{B}) &= \text{var}\left(\sum_{i=1}^{M} \frac{B_i}{M}\right) \\
&= \frac{1}{M^2}\sum_{i=1}^{M} \text{var}\left(B_i\right) + \frac{2}{M^2}\sum_{i<j} \text{cov}(B_i, B_j) \\
&= \frac{M\sigma^2}{M^2} + \frac{2}{M^2}\rho\sigma^2\binom{M}{2} \\
&= \underbrace{\rho\sigma^2}_{\text{Resolve with independence}} + \underbrace{\frac{1-\rho}{M}\sigma^2}_{\text{Resolve with } \uparrow\text{classifiers}}
\end{aligned}$$

Bagging: bootstrap aggregation

Creating Weak Classifiers with resampling:

Bagging: bootstrap aggregation

Creating Weak Classifiers with resampling:

- Suppose we have dependent variables $\boldsymbol{Y}$ and data $\boldsymbol{X}$

Bagging: bootstrap aggregation

Creating Weak Classifiers with resampling:

- Suppose we have dependent variables $\boldsymbol{Y}$ and data $\boldsymbol{X}$
- For each bootstrap step m, $(m = 1, 2, \ldots, M)$ draw N observations with replacement, $\tilde{\boldsymbol{Y}}_m$, $\tilde{\boldsymbol{X}}_m$.

Bagging: bootstrap aggregation

Creating Weak Classifiers with resampling:

- Suppose we have dependent variables $\boldsymbol{Y}$ and data $\boldsymbol{X}$
- For each bootstrap step m, $(m = 1, 2, \ldots, M)$ draw N observations with replacement, $\tilde{\boldsymbol{Y}}_m$, $\tilde{\boldsymbol{X}}_m$.
- Train classifier on bootstrapped data,

Bagging: bootstrap aggregation

Creating Weak Classifiers with resampling:

- Suppose we have dependent variables $\boldsymbol{Y}$ and data $\boldsymbol{X}$
- For each bootstrap step m, $(m = 1, 2, \ldots, M)$ draw N observations with replacement, $\tilde{\boldsymbol{Y}}_m$, $\tilde{\boldsymbol{X}}_m$.
- Train classifier on bootstrapped data,

$$\tilde{\boldsymbol{Y}}_m \quad = \quad f^m(\tilde{\boldsymbol{X}}_m, \widehat{\boldsymbol{\beta}}, \boldsymbol{\lambda})$$

Bagging: bootstrap aggregation

Creating Weak Classifiers with resampling:

- Suppose we have dependent variables $\boldsymbol{Y}$ and data $\boldsymbol{X}$
- For each bootstrap step m, $(m = 1, 2, \ldots, M)$ draw N observations with replacement, $\tilde{\boldsymbol{Y}}_m$, $\tilde{\boldsymbol{X}}_m$.
- Train classifier on bootstrapped data,

$$\begin{aligned} \tilde{\boldsymbol{Y}}_m &= f^m(\tilde{\boldsymbol{X}}_m, \widehat{\boldsymbol{\beta}}, \boldsymbol{\lambda}) \\ \hat{f}^m(\boldsymbol{x}_i, \widehat{\boldsymbol{\beta}}, \boldsymbol{\lambda}) &= \text{Classifier from m}^{\text{th}} \text{ iteration at } \boldsymbol{x}_i \end{aligned}$$

Bagging: bootstrap aggregation

Creating Weak Classifiers with resampling:

- Suppose we have dependent variables $\boldsymbol{Y}$ and data $\boldsymbol{X}$
- For each bootstrap step m, $(m = 1, 2, \ldots, M)$ draw N observations with replacement, $\tilde{\boldsymbol{Y}}_m$, $\tilde{\boldsymbol{X}}_m$.
- Train classifier on bootstrapped data,

$$\begin{aligned} \tilde{\boldsymbol{Y}}_m &= f^m(\tilde{\boldsymbol{X}}_m, \widehat{\boldsymbol{\beta}}, \boldsymbol{\lambda}) \\ \hat{f}^m(\boldsymbol{x}_i, \widehat{\boldsymbol{\beta}}, \boldsymbol{\lambda}) &= \text{Classifier from m}^{\text{th}} \text{ iteration at } \boldsymbol{x}_i \end{aligned}$$

- Aggregating across classifiers,

Bagging: bootstrap aggregation

Creating Weak Classifiers with resampling:

- Suppose we have dependent variables $\boldsymbol{Y}$ and data $\boldsymbol{X}$
- For each bootstrap step m, $(m = 1, 2, \ldots, M)$ draw N observations with replacement, $\tilde{\boldsymbol{Y}}_m$, $\tilde{\boldsymbol{X}}_m$.
- Train classifier on bootstrapped data,

$$\begin{aligned} \tilde{\boldsymbol{Y}}_m &= f^m(\tilde{\boldsymbol{X}}_m, \widehat{\boldsymbol{\beta}}, \boldsymbol{\lambda}) \\ \hat{f}^m(\boldsymbol{x}_i, \widehat{\boldsymbol{\beta}}, \boldsymbol{\lambda}) &= \text{Classifier from m}^{\text{th}} \text{ iteration at } \boldsymbol{x}_i \end{aligned}$$

- Aggregating across classifiers,

$$f_{\text{bag}}(\boldsymbol{x}_i) = \frac{1}{M} \sum_{m=1}^{M} \hat{f}^m(\boldsymbol{x}_i, \widehat{\boldsymbol{\beta}}, \boldsymbol{\lambda})$$

Bagging: bootstrap aggregation

Creating Weak Classifiers with resampling:

- Suppose we have dependent variables $\boldsymbol{Y}$ and data $\boldsymbol{X}$
- For each bootstrap step m, $(m = 1, 2, \ldots, M)$ draw N observations with replacement, $\tilde{\boldsymbol{Y}}_m$, $\tilde{\boldsymbol{X}}_m$.
- Train classifier on bootstrapped data,

$$\begin{aligned} \tilde{\boldsymbol{Y}}_m &= f^m(\tilde{\boldsymbol{X}}_m, \widehat{\boldsymbol{\beta}}, \boldsymbol{\lambda}) \\ \hat{f}^m(\boldsymbol{x}_i, \widehat{\boldsymbol{\beta}}, \boldsymbol{\lambda}) &= \text{Classifier from m}^{\text{th}} \text{ iteration at } \boldsymbol{x}_i \end{aligned}$$

- Aggregating across classifiers,

$$f_{\text{bag}}(\boldsymbol{x}_i) = \frac{1}{M} \sum_{m=1}^{M} \hat{f}^m(\boldsymbol{x}_i, \widehat{\boldsymbol{\beta}}, \boldsymbol{\lambda})$$

- Only leads to a difference in estimate if classifiers are non-linear.

Bagging: bootstrap aggregation

Creating Weak Classifiers with resampling:

- Suppose we have dependent variables $\boldsymbol{Y}$ and data $\boldsymbol{X}$
- For each bootstrap step m, $(m = 1, 2, \ldots, M)$ draw N observations with replacement, $\tilde{\boldsymbol{Y}}_m$, $\tilde{\boldsymbol{X}}_m$.
- Train classifier on bootstrapped data,

$$\begin{aligned} \tilde{\boldsymbol{Y}}_m &= f^m(\tilde{\boldsymbol{X}}_m, \widehat{\boldsymbol{\beta}}, \boldsymbol{\lambda}) \\ \hat{f}^m(\boldsymbol{x}_i, \widehat{\boldsymbol{\beta}}, \boldsymbol{\lambda}) &= \text{Classifier from m}^{\text{th}} \text{ iteration at } \boldsymbol{x}_i \end{aligned}$$

- Aggregating across classifiers,

$$f_{\text{bag}}(\boldsymbol{x}_i) = \frac{1}{M} \sum_{m=1}^{M} \hat{f}^m(\boldsymbol{x}_i, \widehat{\boldsymbol{\beta}}, \boldsymbol{\lambda})$$

- Only leads to a difference in estimate if classifiers are non-linear.
- Strong Correlation between classifiers (recall optimal division from previous slide)

Classification and Regression Trees (CART): Intuition

Consider regression $E[Y|\boldsymbol{x}_i]$.

Classification and Regression Trees (CART): Intuition

Consider regression $E[Y|\boldsymbol{x}_i]$.

With no assumptions, stratify$\rightsquigarrow$ different mean for unique values of $\boldsymbol{x}_i$

Classification and Regression Trees (CART): Intuition

Consider regression $E[Y|\boldsymbol{x}_i]$.

With no assumptions, stratify$\rightsquigarrow$ different mean for unique values of $\boldsymbol{x}_i$

- Within each strata p, compute average Y

Classification and Regression Trees (CART): Intuition

Consider regression $E[Y|\boldsymbol{x}_i]$.

With no assumptions, stratify$\rightsquigarrow$ different mean for unique values of $\boldsymbol{x}_i$

- Within each strata p, compute average Y

$$\bar{Y}|\boldsymbol{x}_p = \sum_{i=1}^{N} \frac{I(\boldsymbol{x}_i = \boldsymbol{x}_p)Y_i}{\sum_{t=1}^{N} I(\boldsymbol{x}_t = \boldsymbol{x}_p)}$$

Classification and Regression Trees (CART): Intuition

Consider regression $E[Y|\boldsymbol{x}_i]$.

With no assumptions, stratify$\rightsquigarrow$ different mean for unique values of $\boldsymbol{x}_i$

- Within each strata p, compute average Y

$$\bar{Y}|\boldsymbol{x}_p = \sum_{i=1}^{N} \frac{I(\boldsymbol{x}_i = \boldsymbol{x}_p)Y_i}{\sum_{t=1}^{N} I(\boldsymbol{x}_t = \boldsymbol{x}_p)}$$

Implies that for test data we would fit:

Classification and Regression Trees (CART): Intuition

Consider regression $E[Y|\boldsymbol{x}_i]$.

With no assumptions, stratify$\rightsquigarrow$ different mean for unique values of $\boldsymbol{x}_i$

- Within each strata p, compute average Y

$$\bar{Y}|\boldsymbol{x}_p = \sum_{i=1}^{N} \frac{I(\boldsymbol{x}_i = \boldsymbol{x}_p)Y_i}{\sum_{t=1}^{N} I(\boldsymbol{x}_t = \boldsymbol{x}_p)}$$

Implies that for test data we would fit:

$$\hat{f}(\boldsymbol{x}_i) = \sum_{p=1}^{P} \bar{Y}|\boldsymbol{x}_p I(\boldsymbol{x}_i = \boldsymbol{x}_p)$$

Classification and Regression Trees (CART): Intuition

Consider regression $E[Y|\boldsymbol{x}_i]$.

With no assumptions, stratify$\rightsquigarrow$ different mean for unique values of $\boldsymbol{x}_i$

- Within each strata p, compute average Y

$$\bar{Y}|\boldsymbol{x}_p = \sum_{i=1}^{N} \frac{I(\boldsymbol{x}_i = \boldsymbol{x}_p)Y_i}{\sum_{t=1}^{N} I(\boldsymbol{x}_t = \boldsymbol{x}_p)}$$

Implies that for test data we would fit:

$$\begin{aligned}\hat{f}(\boldsymbol{x}_i) &= \sum_{p=1}^{P} \bar{Y}|\boldsymbol{x}_p I(\boldsymbol{x}_i = \boldsymbol{x}_p) \\ &= \sum_{p=1}^{P} c_p I(\boldsymbol{x}_i = \boldsymbol{x}_p)\end{aligned}$$

Classification and Regression Trees (CART): Intuition

Consider regression $E[Y|\boldsymbol{x}_i]$.

With no assumptions, stratify$\rightsquigarrow$ different mean for unique values of $\boldsymbol{x}_i$

- Within each strata p, compute average Y

$$\bar{Y}|\boldsymbol{x}_p = \sum_{i=1}^{N} \frac{I(\boldsymbol{x}_i = \boldsymbol{x}_p)Y_i}{\sum_{t=1}^{N} I(\boldsymbol{x}_t = \boldsymbol{x}_p)}$$

Implies that for test data we would fit:

$$\begin{aligned}\hat{f}(\boldsymbol{x}_i) &= \sum_{p=1}^{P} \bar{Y}|\boldsymbol{x}_p I(\boldsymbol{x}_i = \boldsymbol{x}_p) \\ &= \sum_{p=1}^{P} c_p I(\boldsymbol{x}_i = \boldsymbol{x}_p)\end{aligned}$$

Curse of dimensionality(!!!)

Classification and Regression Trees (CART): Intuition

Consider regression $E[Y|\boldsymbol{x}_i]$.

With no assumptions, stratify$\rightsquigarrow$ different mean for unique values of $\boldsymbol{x}_i$

- Within each strata p, compute average Y

$$\bar{Y}|\boldsymbol{x}_p = \sum_{i=1}^{N} \frac{I(\boldsymbol{x}_i = \boldsymbol{x}_p)Y_i}{\sum_{t=1}^{N} I(\boldsymbol{x}_t = \boldsymbol{x}_p)}$$

Implies that for test data we would fit:

$$\begin{aligned}\hat{f}(\boldsymbol{x}_i) &= \sum_{p=1}^{P} \bar{Y}|\boldsymbol{x}_p I(\boldsymbol{x}_i = \boldsymbol{x}_p) \\ &= \sum_{p=1}^{P} c_p I(\boldsymbol{x}_i = \boldsymbol{x}_p)\end{aligned}$$

Curse of dimensionality(!!!)

Approximate with regions$\rightsquigarrow$ search for splits of data to approximate stratification

Classification and Regression Trees (CART): Objective function

Labels $\boldsymbol{Y}_i$ and documents $\boldsymbol{x}_i$

$$\begin{aligned} E[Y|\boldsymbol{x}_i] &= \widehat{f}(\boldsymbol{x}_i) \\ &= \sum_{p=1}^{P} c_p I(\boldsymbol{x}_i \in R_p) \end{aligned}$$

where:

- R_p describes a region $\rightsquigarrow$ node
- c_p describes values of Y_i for document in R_p

Classification and Regression Trees (CART): Optimization function

Suppose we want to minimize sum of squared residuals with each node

Classification and Regression Trees (CART): Optimization function

Suppose we want to minimize sum of squared residuals with each node
Then c_p = Average Y for documents assigned to R_p

Classification and Regression Trees (CART): Optimization function

Suppose we want to minimize sum of squared residuals with each node
Then c_p = Average Y for documents assigned to R_p

$$\widehat{c}_p = \sum_{i=1}^{N} \frac{Y_i I(\boldsymbol{x}_i \in R_p)}{\sum_{j=1}^{N} I(\boldsymbol{x}_j \in R_p)}$$

Classification and Regression Trees (CART): Optimization function

Suppose we want to minimize sum of squared residuals with each node
Then c_p = Average Y for documents assigned to R_p

$$\widehat{c}_p = \sum_{i=1}^{N} \frac{Y_i I(\boldsymbol{x}_i \in R_p)}{\sum_{j=1}^{N} I(\boldsymbol{x}_j \in R_p)}$$

Determining an optimal partition$\rightsquigarrow$ NP-Hard.

Classification and Regression Trees (CART): Optimization function

Suppose we want to minimize sum of squared residuals with each node
Then c_p = Average Y for documents assigned to R_p

$$\widehat{c}_p = \sum_{i=1}^{N} \frac{Y_i I(\boldsymbol{x}_i \in R_p)}{\sum_{j=1}^{N} I(\boldsymbol{x}_j \in R_p)}$$

Determining an optimal partition$\rightsquigarrow$ NP-Hard.
Suppose we are in some node (perhaps at the start).

Classification and Regression Trees (CART): Optimization function

Suppose we want to minimize sum of squared residuals with each node
Then c_p = Average Y for documents assigned to R_p

$$\widehat{c}_p = \sum_{i=1}^{N} \frac{Y_i I(\boldsymbol{x}_i \in R_p)}{\sum_{j=1}^{N} I(\boldsymbol{x}_j \in R_p)}$$

Determining an optimal partition$\rightsquigarrow$ NP-Hard.

Suppose we are in some node (perhaps at the start).
Greedy algorithm:

Classification and Regression Trees (CART): Optimization function

Suppose we want to minimize sum of squared residuals with each node
Then c_p = Average Y for documents assigned to R_p

$$\widehat{c}_p = \sum_{i=1}^{N} \frac{Y_i I(\boldsymbol{x}_i \in R_p)}{\sum_{j=1}^{N} I(\boldsymbol{x}_j \in R_p)}$$

Determining an optimal partition $\rightsquigarrow$ NP-Hard.

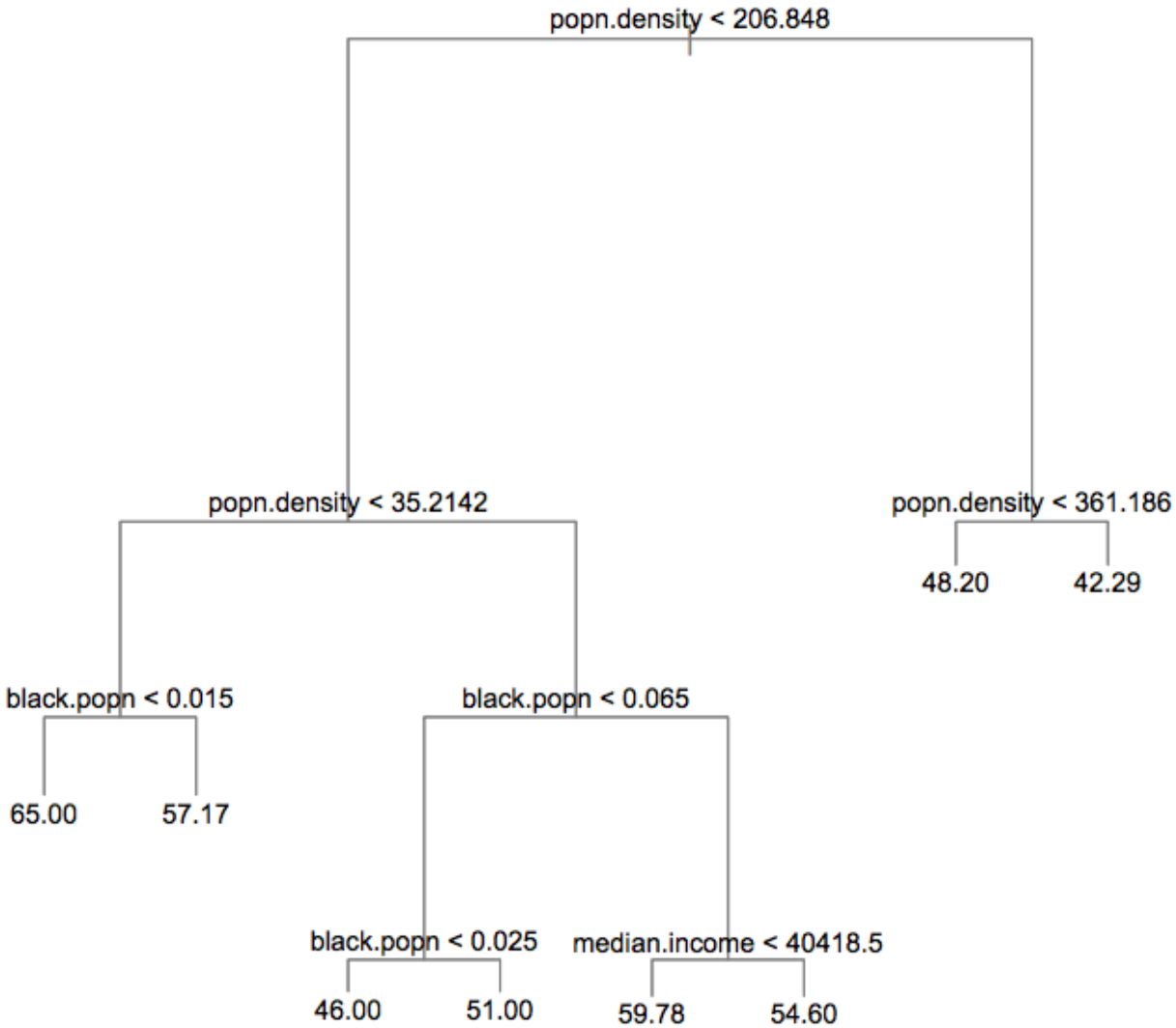
Suppose we are in some node (perhaps at the start).
Greedy algorithm:

$$(j^*, s^*) = \text{arg min}_{j,s} \left[\underbrace{\text{min}_{c_1} \sum_{i=1}^{N} I(x_{ij} < s)(Y_i - c_1)^2}_{\text{“cost” group 1}} + \underbrace{\text{min}_{c_2} \sum_{i=1}^{N} I(x_{ij} > s)(Y_i - c_2)^2}_{\text{“cost” group 2}} \right]$$

Classification and Regression Trees (CART): Algorithm

- Start in Node
- Partition according to Greedy algorithm
- Continue until some stopping rule: number of observations per node

CART Picture (Spirling 2008)



Forests and Trees

Recall: accurate (unbiased) and uncorrelated classifiers

Forests and Trees

Recall: accurate (unbiased) and uncorrelated classifiers

- Grow trees deeply$\rightsquigarrow$ unbiased classifers, though high variance

Forests and Trees

Recall: accurate (unbiased) and uncorrelated classifiers

- Grow trees deeply$\rightsquigarrow$ unbiased classifers, though high variance
- Average$\rightsquigarrow$ reduces variance, but will be correlated

Forests and Trees

Recall: accurate (unbiased) and uncorrelated classifiers

- Grow trees deeply$\rightsquigarrow$ unbiased classifers, though high variance
- Average$\rightsquigarrow$ reduces variance, but will be correlated
- Random forest$\rightsquigarrow$ introduce additional sampling to induce independence$\rightsquigarrow$ Only split on subset of variables

Random Forest Algorithm (ESL, 588)

Random Forest Algorithm (ESL, 588)

1) For m bootstrap samples ($m = 1, \ldots, M$), draw N observations with replacement, $\tilde{\boldsymbol{Y}}_m, \tilde{\boldsymbol{X}}_m$

Random Forest Algorithm (ESL, 588)

1) For m bootstrap samples ($m = 1, \ldots, M$), draw N observations with replacement, $\tilde{\boldsymbol{Y}}_m, \tilde{\boldsymbol{X}}_m$
2) Until a minimum node size is reached:

Random Forest Algorithm (ESL, 588)

1) For m bootstrap samples ($m = 1, \ldots, M$), draw N observations with replacement, $\tilde{\boldsymbol{Y}}_m, \tilde{\boldsymbol{X}}_m$
2) Until a minimum node size is reached:
   i) Select z of the J variables$\rightsquigarrow$ introduces independences across the trees

Random Forest Algorithm (ESL, 588)

1) For m bootstrap samples ($m = 1, \ldots, M$), draw N observations with replacement, $\tilde{\boldsymbol{Y}}_m, \tilde{\boldsymbol{X}}_m$
2) Until a minimum node size is reached:
   i) Select z of the J variables$\rightsquigarrow$ introduces independences across the trees
   ii) Among those z, select the best split node

Random Forest Algorithm (ESL, 588)

1) For m bootstrap samples ($m = 1, \ldots, M$), draw N observations with replacement, $\tilde{\boldsymbol{Y}}_m, \tilde{\boldsymbol{X}}_m$
2) Until a minimum node size is reached:
   i) Select z of the J variables$\rightsquigarrow$ introduces independences across the trees
   ii) Among those z, select the best split node
   iii) Split into daughter nodes

Random Forest Algorithm (ESL, 588)

1) For m bootstrap samples $(m = 1, \ldots, M)$, draw N observations with replacement, $\tilde{\boldsymbol{Y}}_m, \tilde{\boldsymbol{X}}_m$
2) Until a minimum node size is reached:
   i) Select z of the J variables$\rightsquigarrow$ introduces independences across the trees
   ii) Among those z, select the best split node
   iii) Split into daughter nodes
3) The result is an ensemble (forest) of trees $\boldsymbol{T} = (T_1, T_2, \ldots, T_M)$,

Random Forest Algorithm (ESL, 588)

1) For m bootstrap samples ($m = 1, \ldots, M$), draw N observations with replacement, $\tilde{\boldsymbol{Y}}_m, \tilde{\boldsymbol{X}}_m$
2) Until a minimum node size is reached:
   i) Select z of the J variables $\rightsquigarrow$ introduces independences across the trees
   ii) Among those z, select the best split node
   iii) Split into daughter nodes
3) The result is an ensemble (forest) of trees $\boldsymbol{T} = (T_1, T_2, \ldots, T_M)$,

$$\hat{f}(\boldsymbol{x}_i) \quad = \quad \frac{1}{M} \sum_{m=1}^{M} T_m(\boldsymbol{x}_i)$$

Random Forest Algorithm (ESL, 588)

1) For m bootstrap samples ($m = 1, \ldots, M$), draw N observations with replacement, $\tilde{\boldsymbol{Y}}_m, \tilde{\boldsymbol{X}}_m$
2) Until a minimum node size is reached:
   i) Select z of the J variables$\rightsquigarrow$ introduces independences across the trees
   ii) Among those z, select the best split node
   iii) Split into daughter nodes
3) The result is an ensemble (forest) of trees $\boldsymbol{T} = (T_1, T_2, \ldots, T_M)$,

$$\hat{f}(\boldsymbol{x}_i) \quad = \quad \frac{1}{M} \sum_{m=1}^{M} T_m(\boldsymbol{x}_i)$$

`RandomForest`$\rightsquigarrow$ Not a silver bullet!

Random Forest Algorithm (ESL, 588)

1) For m bootstrap samples $(m = 1, \ldots, M)$, draw N observations with replacement, $\tilde{\boldsymbol{Y}}_m, \tilde{\boldsymbol{X}}_m$
2) Until a minimum node size is reached:
   i) Select z of the J variables$\rightsquigarrow$ introduces independences across the trees
   ii) Among those z, select the best split node
   iii) Split into daughter nodes
3) The result is an ensemble (forest) of trees $\boldsymbol{T} = (T_1, T_2, \ldots, T_M)$,

$$\hat{f}(\boldsymbol{x}_i) \quad = \quad \frac{1}{M} \sum_{m=1}^{M} T_m(\boldsymbol{x}_i)$$

`RandomForest`$\rightsquigarrow$ Not a silver bullet!

- With many poor predictors$\rightsquigarrow$ the p selected may be meaningless

Random Forest Algorithm (ESL, 588)

1) For m bootstrap samples $(m = 1, \ldots, M)$, draw N observations with replacement, $\tilde{\boldsymbol{Y}}_m, \tilde{\boldsymbol{X}}_m$
2) Until a minimum node size is reached:
   i) Select z of the J variables$\rightsquigarrow$ introduces independences across the trees
   ii) Among those z, select the best split node
   iii) Split into daughter nodes
3) The result is an ensemble (forest) of trees $\boldsymbol{T} = (T_1, T_2, \ldots, T_M)$,

$$\hat{f}(\boldsymbol{x}_i) \quad = \quad \frac{1}{M} \sum_{m=1}^{M} T_m(\boldsymbol{x}_i)$$

`RandomForest`$\rightsquigarrow$ Not a silver bullet!

- With many poor predictors$\rightsquigarrow$ the p selected may be meaningless
- Wager and Athey (2015): Random Forest for estimating heterogeneous effects

Super Learning

1) Set of hand labeled documents. For each i, $(i = 1, \ldots, N_{\text{train}})$

Super Learning

1) Set of hand labeled documents. For each i, $(i = 1, \ldots, N_{\text{train}})$

$$Y_{i,\text{train}} \in \{C_1 C_2, \ldots C_K\}$$

Super Learning

1) Set of hand labeled documents. For each i, $(i = 1, \ldots, N_{\text{train}})$
   $Y_{i,\text{train}} \in \{C_1 C_2, \ldots C_K\}$
2) Estimate relationship between labels and words

Super Learning

1) Set of hand labeled documents. For each i, $(i = 1, \ldots, N_{\text{train}})$
   $Y_{i,\text{train}} \in \{C_1 C_2, \ldots C_K\}$
2) Estimate relationship between labels and words
   - Each document i is a count vector of K words

Super Learning

1) Set of hand labeled documents. For each i, $(i = 1, \ldots, N_{\text{train}})$

   $Y_{i,\text{train}} \in \{C_1 C_2, \ldots C_K\}$

2) Estimate relationship between labels and words

   - Each document i is a count vector of K words

     $\mathbf{x}_{i,\text{train}} = (X_{i1}, X_{i2}, \ldots, X_{iK})$

Super Learning

1) Set of hand labeled documents. For each i, $(i = 1, \ldots, N_{\text{train}})$
   $Y_{i,\text{train}} \in \{C_1 C_2, \ldots C_K\}$
2) Estimate relationship between labels and words
   - Each document i is a count vector of K words
     $\mathbf{x}_{i,\text{train}} = (X_{i1}, X_{i2}, \ldots, X_{iK})$

$$\Pr(Y_i = C_k | \mathbf{x}_i)$$

Super Learning

1) Set of hand labeled documents. For each i, $(i = 1, \ldots, N_{\text{train}})$
   $Y_{i,\text{train}} \in \{C_1 C_2, \ldots C_K\}$
2) Estimate relationship between labels and words
   - Each document i is a count vector of K words
     $\mathbf{x}_{i,\text{train}} = (X_{i1}, X_{i2}, \ldots, X_{iK})$

$$\Pr(Y_i = C_k | \mathbf{x}_i)_{\text{train}}$$

Super Learning

1) Set of hand labeled documents. For each i, $(i = 1, \ldots, N_{\text{train}})$
   $Y_{i,\text{train}} \in \{C_1 C_2, \ldots C_K\}$
2) Estimate relationship between labels and words
   - Each document i is a count vector of K words
     $\mathbf{x}_{i,\text{train}} = (X_{i1}, X_{i2}, \ldots, X_{iK})$

$$\Pr(Y_i = C_k | \mathbf{x}_i)_{\text{train}} \quad = \quad \widehat{g}(\mathbf{x}_i)_{\text{train}}$$

- Identify systematic relationship between words, labels

Super Learning

1) Set of hand labeled documents. For each i, $(i = 1, \ldots, N_{\text{train}})$
   $Y_{i,\text{train}} \in \{C_1 C_2, \ldots C_K\}$
2) Estimate relationship between labels and words
   - Each document i is a count vector of K words
     $\mathbf{x}_{i,\text{train}} = (X_{i1}, X_{i2}, \ldots, X_{iK})$

$$\Pr(Y_i = C_k | \mathbf{x}_i)_{\text{train}} \quad = \quad \widehat{g}(\mathbf{x}_i)_{\text{train}}$$

- Identify systematic relationship between words, labels $\rightsquigarrow$ Data and assumptions

Super Learning

1) Set of hand labeled documents. For each i, $(i = 1, \ldots, N_{\text{train}})$
   $Y_{i,\text{train}} \in \{C_1 C_2, \ldots C_K\}$
2) Estimate relationship between labels and words
   - Each document i is a count vector of K words
     $\mathbf{x}_{i,\text{train}} = (X_{i1}, X_{i2}, \ldots, X_{iK})$

$$\Pr(Y_i = C_k | \mathbf{x}_i)_{\text{train}} \quad = \quad \widehat{g}(\mathbf{x}_i)_{\text{train}}$$

- Identify systematic relationship between words, labels $\rightsquigarrow$ Data and assumptions
  - LASSO (Tibshirani 1996): sparsity

Super Learning

1) Set of hand labeled documents. For each i, $(i = 1, \ldots, N_{\text{train}})$
   $Y_{i,\text{train}} \in \{C_1 C_2, \ldots C_K\}$
2) Estimate relationship between labels and words
   - Each document i is a count vector of K words
     $\mathbf{x}_{i,\text{train}} = (X_{i1}, X_{i2}, \ldots, X_{iK})$

$$\Pr(Y_i = C_k | \mathbf{x}_i)_{\text{train}} \quad = \quad \widehat{g}(\mathbf{x}_i)_{\text{train}}$$

- Identify systematic relationship between words, labels $\rightsquigarrow$ Data and assumptions
  - LASSO (Tibshirani 1996): sparsity
  - KRLS (Hainmueller and Hazlett 2013): dense, flexible surface

Super Learning

1) Set of hand labeled documents. For each i, $(i = 1, \ldots, N_{\text{train}})$
   $Y_{i,\text{train}} \in \{C_1 C_2, \ldots C_K\}$
2) Estimate relationship between labels and words
   - Each document i is a count vector of K words
     $\mathbf{x}_{i,\text{train}} = (X_{i1}, X_{i2}, \ldots, X_{iK})$

$$\Pr(Y_i = C_k | \mathbf{x}_i)_{\text{train}} \quad = \quad \widehat{g}(\mathbf{x}_i)_{\text{train}}$$

- Identify systematic relationship between words, labels $\rightsquigarrow$ Data and assumptions
  - LASSO (Tibshirani 1996): sparsity
  - KRLS (Hainmueller and Hazlett 2013): dense, flexible surface
  - Ridge, Elastic-Net, SVM, Random Forests, BART, ...

Super Learning

1) Set of hand labeled documents. For each i, $(i = 1, \ldots, N_{\text{train}})$
   $Y_{i,\text{train}} \in \{C_1 C_2, \ldots C_K\}$
2) Estimate relationship between labels and words
   - Each document i is a count vector of K words
     $\mathbf{x}_{i,\text{train}} = (X_{i1}, X_{i2}, \ldots, X_{iK})$

$$\Pr(Y_i = C_k | \mathbf{x}_i)_{\text{train}} \quad = \quad \widehat{g}(\mathbf{x}_i)_{\text{train}}$$

- Identify systematic relationship between words, labels $\rightsquigarrow$ Data and assumptions
  - LASSO (Tibshirani 1996): sparsity
  - KRLS (Hainmueller and Hazlett 2013): dense, flexible surface
  - Ridge, Elastic-Net, SVM, Random Forests, BART, ...
- Which model? Difficult to know before hand

Super Learning

1) Set of hand labeled documents. For each i, $(i = 1, \ldots, N_{\text{train}})$
   $Y_{i,\text{train}} \in \{C_1 C_2, \ldots C_K\}$
2) Estimate relationship between labels and words
   - Each document i is a count vector of K words
     $\mathbf{x}_{i,\text{train}} = (X_{i1}, X_{i2}, \ldots, X_{iK})$

$$\Pr(Y_i = C_k | \mathbf{x}_i)_{\text{train}} \quad = \quad \widehat{g}(\mathbf{x}_i)_{\text{train}}$$

- Identify systematic relationship between words, labels $\rightsquigarrow$ Data and assumptions
  - LASSO (Tibshirani 1996): sparsity
  - KRLS (Hainmueller and Hazlett 2013): dense, flexible surface
  - Ridge, Elastic-Net, SVM, Random Forests, BART, ...
- Which model? Difficult to know before hand
- Assess out of sample performance with cross validation

Super Learning

1) Set of hand labeled documents. For each i, $(i = 1, \ldots, N_{\text{train}})$
   $Y_{i,\text{train}} \in \{C_1 C_2, \ldots C_K\}$
2) Estimate relationship between labels and words
   - Each document i is a count vector of K words
     $\mathbf{x}_{i,\text{train}} = (X_{i1}, X_{i2}, \ldots, X_{iK})$

$$\Pr(Y_i = C_k | \mathbf{x}_i)_{\text{train}} \quad = \quad \widehat{g}(\mathbf{x}_i)_{\text{train}}$$

- Identify systematic relationship between words, labels $\rightsquigarrow$ Data and assumptions
  - LASSO (Tibshirani 1996): sparsity
  - KRLS (Hainmueller and Hazlett 2013): dense, flexible surface
  - Ridge, Elastic-Net, SVM, Random Forests, BART, ...
- Which model? Difficult to know before hand
- Assess out of sample performance with cross validation

Weighted ensemble: weights determined by (unique) out of sample predictive performance

Committee Methods:
Fit many methods, average with equal weights

- Voting (classification)
- Averaging (predictions)

Problem: many poor methods may overwhelm high quality fit (remember earlier figures)
Solution: learn weights via cross validation

Weighted Ensemble to Classify Documents

- Suppose we have M $(m = 1, \ldots, M)$ models.

Weighted Ensemble to Classify Documents

- Suppose we have M $(m = 1, \ldots, M)$ models.

$$\Pr(Y_i = \mathsf{C}_1 | \mathbf{x}_i)_{\text{train}} = \sum_{m=1}^{M} \widehat{\pi}_m \widehat{g}_m(\mathbf{x}_i)$$

Weighted Ensemble to Classify Documents

- Suppose we have M $(m = 1, \ldots, M)$ models.

$$\Pr(Y_i = \mathsf{C}_1 | \mathbf{x}_i)_{\text{train}} = \sum_{m=1}^{M} \widehat{\pi}_m \widehat{g}_m(\mathbf{x}_i)$$

Weighted Ensemble to Classify Documents

- Suppose we have M $(m = 1, \ldots, M)$ models.

$$\Pr(Y_i = \mathsf{C}_1 | \mathbf{x}_i)_{\text{train}} = \sum_{m=1}^{M} \widehat{\pi}_m \widehat{g}_m(\mathbf{x}_i)$$

Weighted Ensemble to Classify Documents

- Suppose we have M $(m = 1, \ldots, M)$ models.

$$\Pr(Y_i = \mathsf{C}_1 | \mathbf{x}_i)_{\text{train}} = \sum_{m=1}^{M} \widehat{\pi}_m \widehat{g}_m(\mathbf{x}_i)$$

- Estimate weights $(\widehat{\pi}_m)$

Weighted Ensemble to Classify Documents

- Suppose we have M $(m = 1, \ldots, M)$ models.

$$\Pr(Y_i = \mathsf{C}_1 | \mathbf{x}_i)_{\text{train}} \quad = \quad \sum_{m=1}^{M} \widehat{\pi}_m \widehat{g}_m(\mathbf{x}_i)$$

- Estimate weights $(\widehat{\pi}_m)$
  - K-fold cross validation: generate M out of sample predictions for each document in training set

Weighted Ensemble to Classify Documents

- Suppose we have M $(m = 1, \ldots, M)$ models.

$$\Pr(Y_i = \mathsf{C}_1 | \mathbf{x}_i)_{\text{train}} = \sum_{m=1}^{M} \widehat{\pi}_m \widehat{g}_m(\mathbf{x}_i)$$

- Estimate weights $(\widehat{\pi}_m)$
  - K-fold cross validation: generate M out of sample predictions for each document in training set
    $\widehat{\mathbf{Y}}_i = (\widehat{Y}_{i1}, \widehat{Y}_{i2}, \ldots, \widehat{Y}_{iM})$

Weighted Ensemble to Classify Documents

- Suppose we have M $(m = 1, \ldots, M)$ models.

$$\Pr(Y_i = \mathsf{C}_1 | \mathbf{x}_i)_{\text{train}} \quad = \quad \sum_{m=1}^{M} \widehat{\pi}_m \widehat{g}_m(\mathbf{x}_i)$$

- Estimate weights $(\widehat{\pi}_m)$
  - K-fold cross validation: generate M out of sample predictions for each document in training set
    $\widehat{\mathbf{Y}}_i = (\widehat{Y}_{i1}, \widehat{Y}_{i2}, \ldots, \widehat{Y}_{iM})$
  - Estimate weights with constrained regression:

Weighted Ensemble to Classify Documents

- Suppose we have M $(m = 1, \ldots, M)$ models.

$$\Pr(Y_i = \mathsf{C}_1 | \mathbf{x}_i)_{\text{train}} \quad = \quad \sum_{m=1}^{M} \widehat{\pi}_m \widehat{g}_m(\mathbf{x}_i)$$

- Estimate weights $(\widehat{\pi}_m)$
  - K-fold cross validation: generate M out of sample predictions for each document in training set
    $\widehat{\mathbf{Y}}_i = (\widehat{Y}_{i1}, \widehat{Y}_{i2}, \ldots, \widehat{Y}_{iM})$
  - Estimate weights with constrained regression:

$$Y_i \quad = \quad \sum_{m=1}^{M} \pi_m \hat{Y}_{im} + \epsilon_i$$

Weighted Ensemble to Classify Documents

- Suppose we have M $(m = 1, \ldots, M)$ models.

$$\Pr(Y_i = \mathsf{C}_1 | \mathbf{x}_i)_{\text{train}} \quad = \quad \sum_{m=1}^{M} \widehat{\pi}_m \widehat{g}_m(\mathbf{x}_i)$$

- Estimate weights $(\widehat{\pi}_m)$
  - K-fold cross validation: generate M out of sample predictions for each document in training set
    $\widehat{\mathbf{Y}}_i = (\widehat{Y}_{i1}, \widehat{Y}_{i2}, \ldots, \widehat{Y}_{iM})$
  - Estimate weights with constrained regression:

$$Y_i \quad = \quad \sum_{m=1}^{M} \pi_m \hat{Y}_{im} + \epsilon_i$$

where we impose constraints: $\pi_m \geq 0$ and $\sum_{m=1}^{M} \pi_m = 1$.

Weighted Ensemble to Classify Documents

- Suppose we have M $(m = 1, \ldots, M)$ models.

$$\Pr(Y_i = \mathsf{C}_1 | \mathbf{x}_i)_{\text{train}} \quad = \quad \sum_{m=1}^{M} \widehat{\pi}_m \widehat{g}_m(\mathbf{x}_i)$$

- Estimate weights $(\widehat{\pi}_m)$
  - K-fold cross validation: generate M out of sample predictions for each document in training set
    $\widehat{\mathbf{Y}}_i = (\widehat{Y}_{i1}, \widehat{Y}_{i2}, \ldots, \widehat{Y}_{iM})$
  - Estimate weights with constrained regression:

$$Y_i \quad = \quad \sum_{m=1}^{M} \pi_m \hat{Y}_{im} + \epsilon_i$$

    where we impose constraints: $\pi_m \geq 0$ and $\sum_{m=1}^{M} \pi_m = 1$.
  - Result $\widehat{\pi}_m$ for each method

Weighted Ensemble to Classify Documents

- Suppose we have M $(m = 1, \ldots, M)$ models.

$$\Pr(Y_i = \mathsf{C}_1 | \mathbf{x}_i)_{\text{train}} \quad = \quad \sum_{m=1}^{M} \widehat{\pi}_m \widehat{g}_m(\mathbf{x}_i)$$

- Estimate weights $(\widehat{\pi}_m)$
- Estimate $\widehat{g}_m(\mathbf{x}_i) \rightsquigarrow$ Apply all M models to entire training set

Weighted Ensemble to Classify Documents

- Suppose we have M $(m = 1, \ldots, M)$ models.

$$\Pr(Y_i = \mathsf{C}_1 | \mathbf{x}_i)_{\text{train}} \quad = \quad \sum_{m=1}^{M} \widehat{\pi}_m \widehat{g}_m(\mathbf{x}_i)$$

- Estimate weights $(\widehat{\pi}_m)$
- Estimate $\widehat{g}_m(\mathbf{x}_i) \rightsquigarrow$ Apply all M models to entire training set

3) For each document i in test set, $\mathbf{x}_{i,\text{test}}$

Weighted Ensemble to Classify Documents

- Suppose we have M $(m = 1, \ldots, M)$ models.

$$\Pr(Y_i = \mathsf{C}_1 | \mathbf{x}_{i,\text{test}})_{\text{train}} = \sum_{m=1}^{M} \widehat{\pi}_m \widehat{g}_m(\mathbf{x}_{i,\text{test}})$$

- Estimate weights $(\widehat{\pi}_m)$
- Estimate $\widehat{g}_m(\mathbf{x}_i) \rightsquigarrow$ Apply all M models to entire training set

3) For each document i in test set, $\mathbf{x}_{i,\text{test}}$

Weighted Ensemble to Classify Documents

- Suppose we have M $(m = 1, \ldots, M)$ models.

$$\Pr(Y_i = \mathsf{C}_1 | \mathbf{x}_{i,\text{test}})_{\text{train}} = \sum_{m=1}^{M} \widehat{\pi}_m \widehat{g}_m(\mathbf{x}_{i,\text{test}})$$

- Estimate weights $(\widehat{\pi}_m)$
- Estimate $\widehat{g}_m(\mathbf{x}_i) \rightsquigarrow$ Apply all M models to entire training set

3) For each document i in test set, $\mathbf{x}_{i,\text{test}}$
(Classify if above threshold)

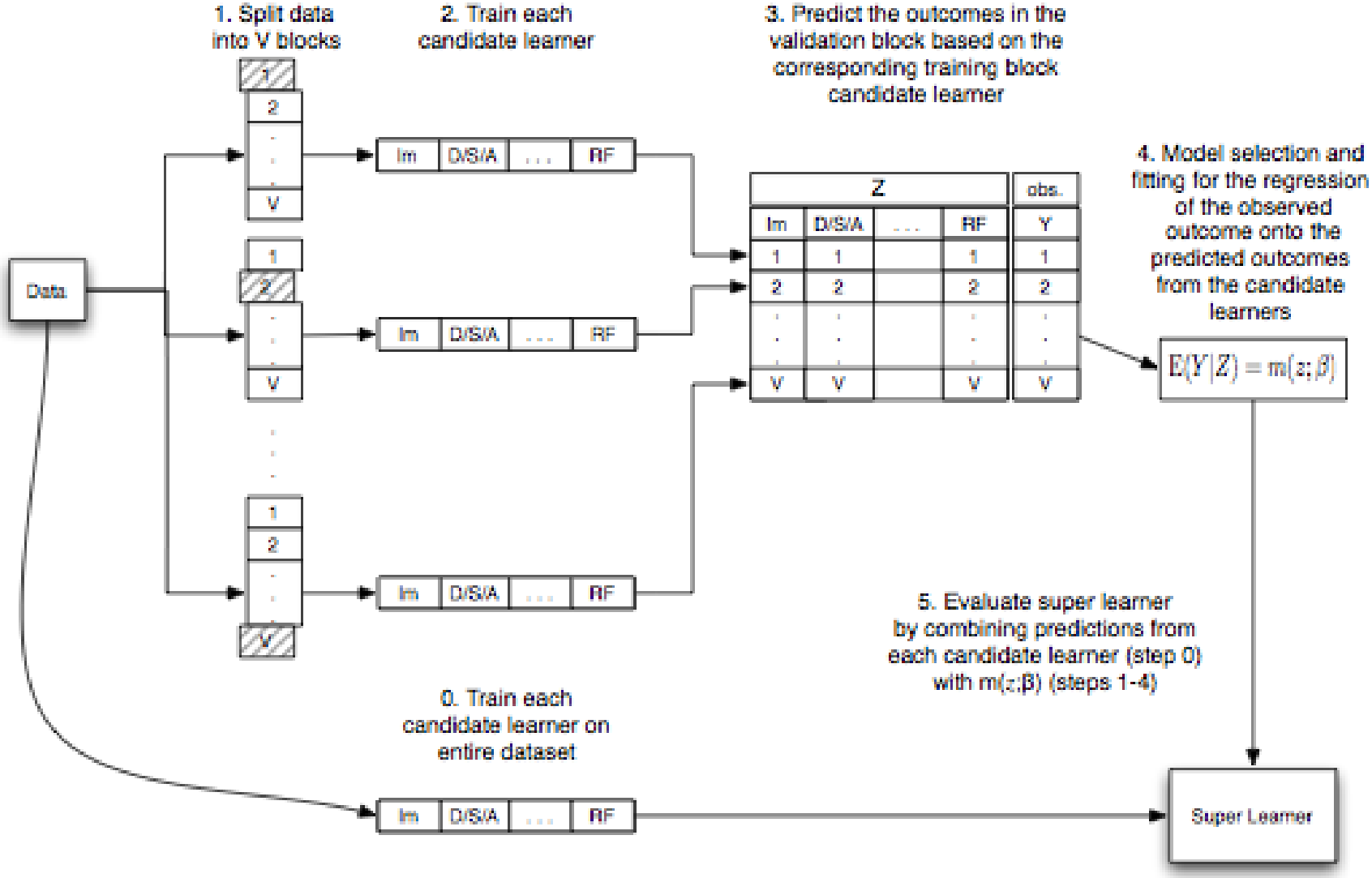


Figure 1: Flow Diagram for Super Learner

Why Super Learn?

van der Laan et al (2007) prove:

- Asymptotically: super learners will perform as well the best candidates for data
- Oracle: performs like the best possible method among candidate methods
  - Asymptotically outperforms constituent methods
  - Performs as well as optimal combinations of those methods

Practical questions:

- Final regression:
  - Logistic
  - Linear
  - Could super learn again!
- How Many Folds?
  - van der Laan et al's proofs rely on growing folds with N (but slowly)
  - Use 10-fold cross validation for simulations

Impression of Influence

Estimate: $Y_i \in \{\text{Credit}, \text{Not Credit}\}$

Impression of Influence

Estimate: $Y_i \in \{\text{Credit}, \text{Not Credit}\}$

- Triple hand code 800 press releases

Impression of Influence

Estimate: $Y_i \in \{\text{Credit}, \text{Not Credit}\}$

- Triple hand code 800 press releases
- Resolve disagreement with voting$\rightsquigarrow$ few disagreements

Impression of Influence

Estimate: $Y_i \in \{\text{Credit}, \text{Not Credit}\}$

- Triple hand code 800 press releases
- Resolve disagreement with voting$\rightsquigarrow$ few disagreements

Use five classifiers to form Ensemble (cross validating within each to tune parameters)

Impression of Influence

Estimate: $Y_i \in \{\text{Credit}, \text{Not Credit}\}$

- Triple hand code 800 press releases
- Resolve disagreement with voting$\rightsquigarrow$ few disagreements

Use five classifiers to form Ensemble (cross validating within each to tune parameters)

- LASSO

Impression of Influence

Estimate: $Y_i \in \{\text{Credit}, \text{Not Credit}\}$

- Triple hand code 800 press releases
- Resolve disagreement with voting$\rightsquigarrow$ few disagreements

Use five classifiers to form Ensemble (cross validating within each to tune parameters)

- LASSO
- Elastic-Net

Impression of Influence

Estimate: $Y_i \in \{\text{Credit}, \text{Not Credit}\}$

- Triple hand code 800 press releases
- Resolve disagreement with voting$\rightsquigarrow$ few disagreements

Use five classifiers to form Ensemble (cross validating within each to tune parameters)

- LASSO
- Elastic-Net
- Random Forest

Impression of Influence

Estimate: $Y_i \in \{\text{Credit}, \text{Not Credit}\}$

- Triple hand code 800 press releases
- Resolve disagreement with voting$\rightsquigarrow$ few disagreements

Use five classifiers to form Ensemble (cross validating within each to tune parameters)

- LASSO
- Elastic-Net
- Random Forest
- A Support Vector Machine

Impression of Influence

Estimate: $Y_i \in \{\text{Credit}, \text{Not Credit}\}$

- Triple hand code 800 press releases
- Resolve disagreement with voting$\rightsquigarrow$ few disagreements

Use five classifiers to form Ensemble (cross validating within each to tune parameters)

- LASSO
- Elastic-Net
- Random Forest
- A Support Vector Machine
- Kernel Regularized Least Squares (KRLS, Hainmueller and Hazlett 2014)

Impression of Influence

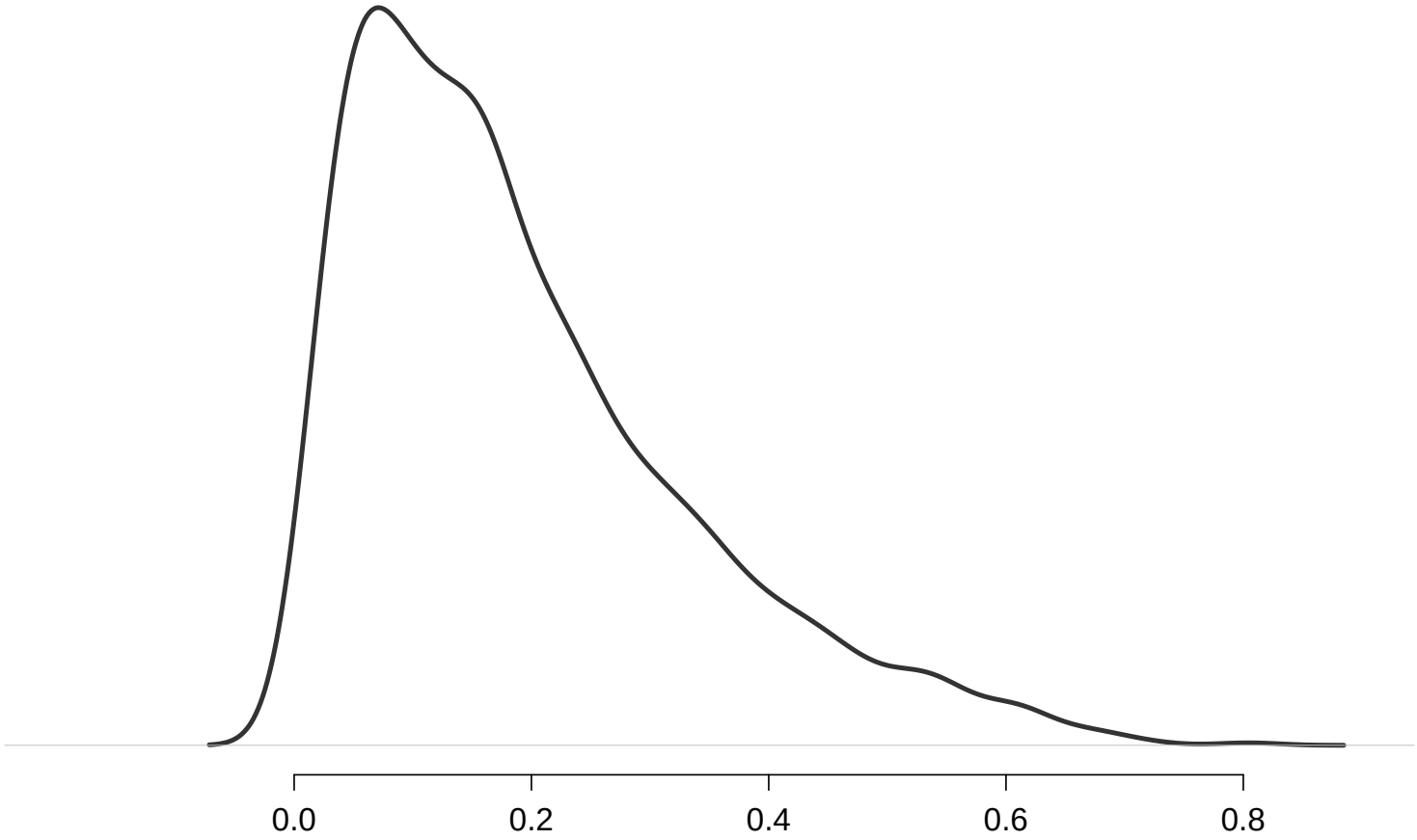
Estimate: $Y_i \in \{\text{Credit}, \text{Not Credit}\}$

- Triple hand code 800 press releases
- Resolve disagreement with voting$\rightsquigarrow$ few disagreements

Use five classifiers to form Ensemble (cross validating within each to tune parameters)

- LASSO 0
- Elastic-Net 23%
- Random Forest 61%
- A Support Vector Machine16%
- Kernel Regularized Least Squares (KRLS, Hainmueller and Hazlett 2014) 0

Strategic Credit Claiming to Build a Personal Vote



Strategic Credit Claiming to Build a Personal Vote

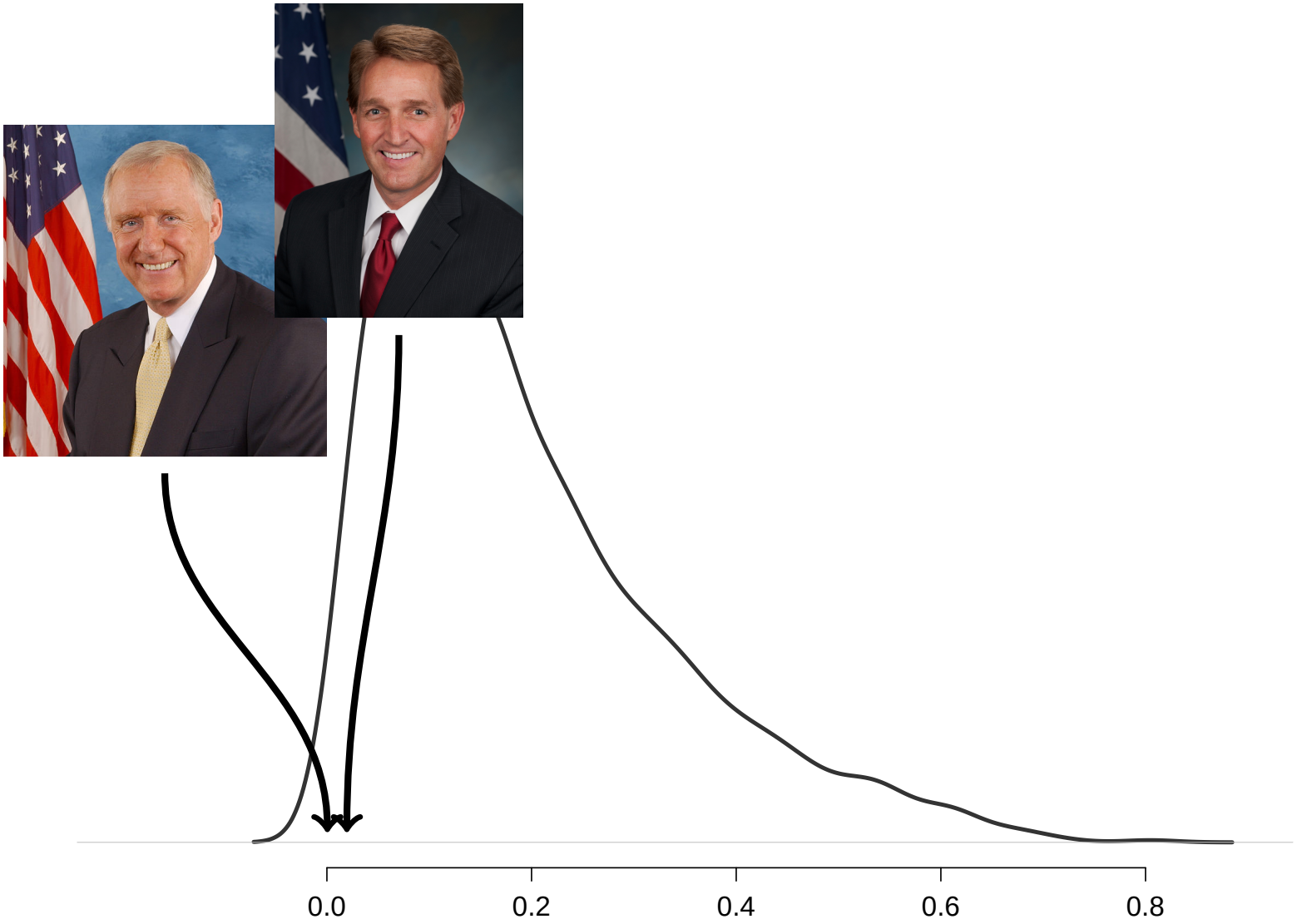
Strategic Credit Claiming to Build a Personal Vote

John McGroff: "voted for every spending bill " that went through the office"

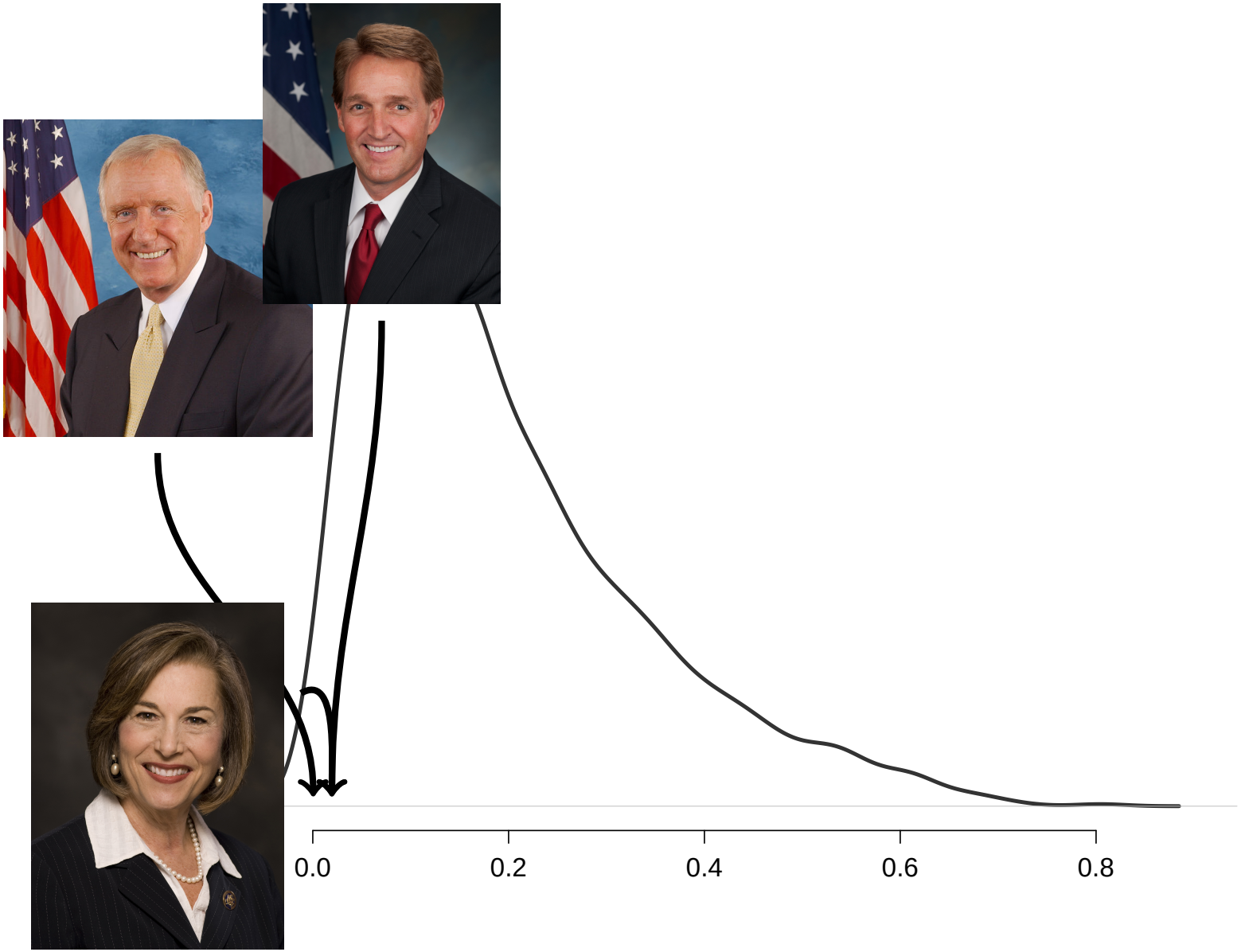
Strategic Credit Claiming to Build a Personal Vote

John McGroff: "Not the actions of a fiscally "
conservative congressman who
cares about personal responsibility"

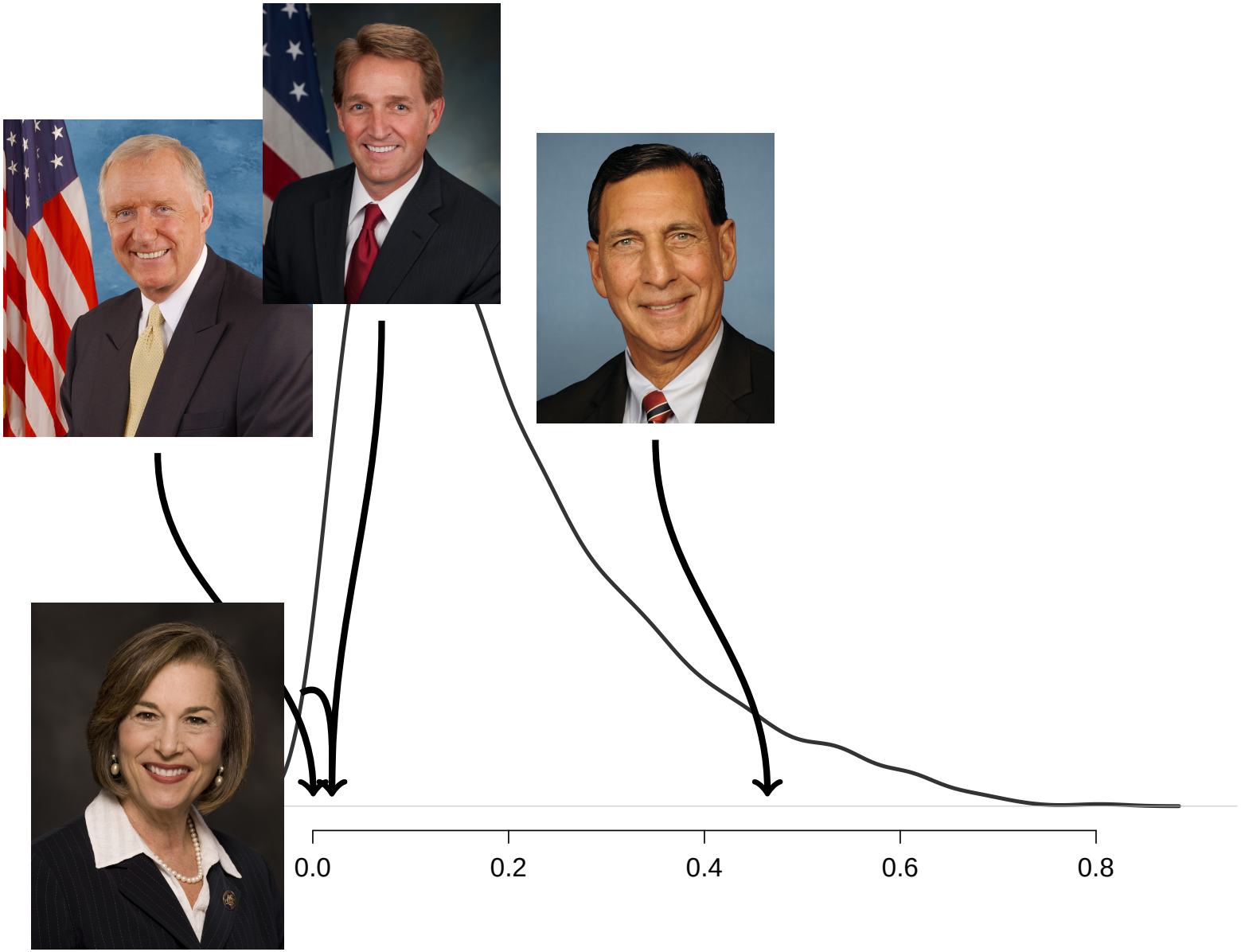
Strategic Credit Claiming to Build a Personal Vote



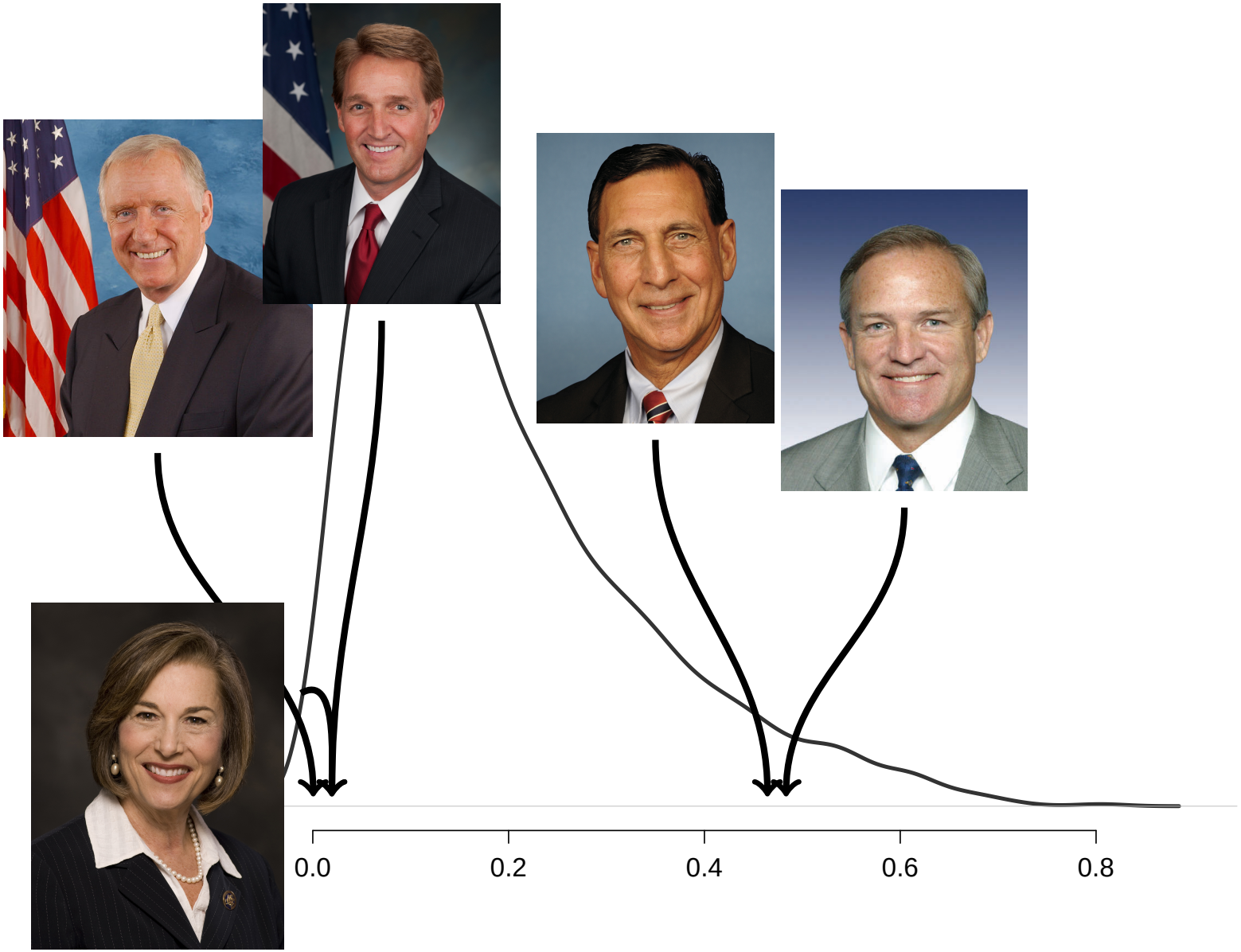
Strategic Credit Claiming to Build a Personal Vote



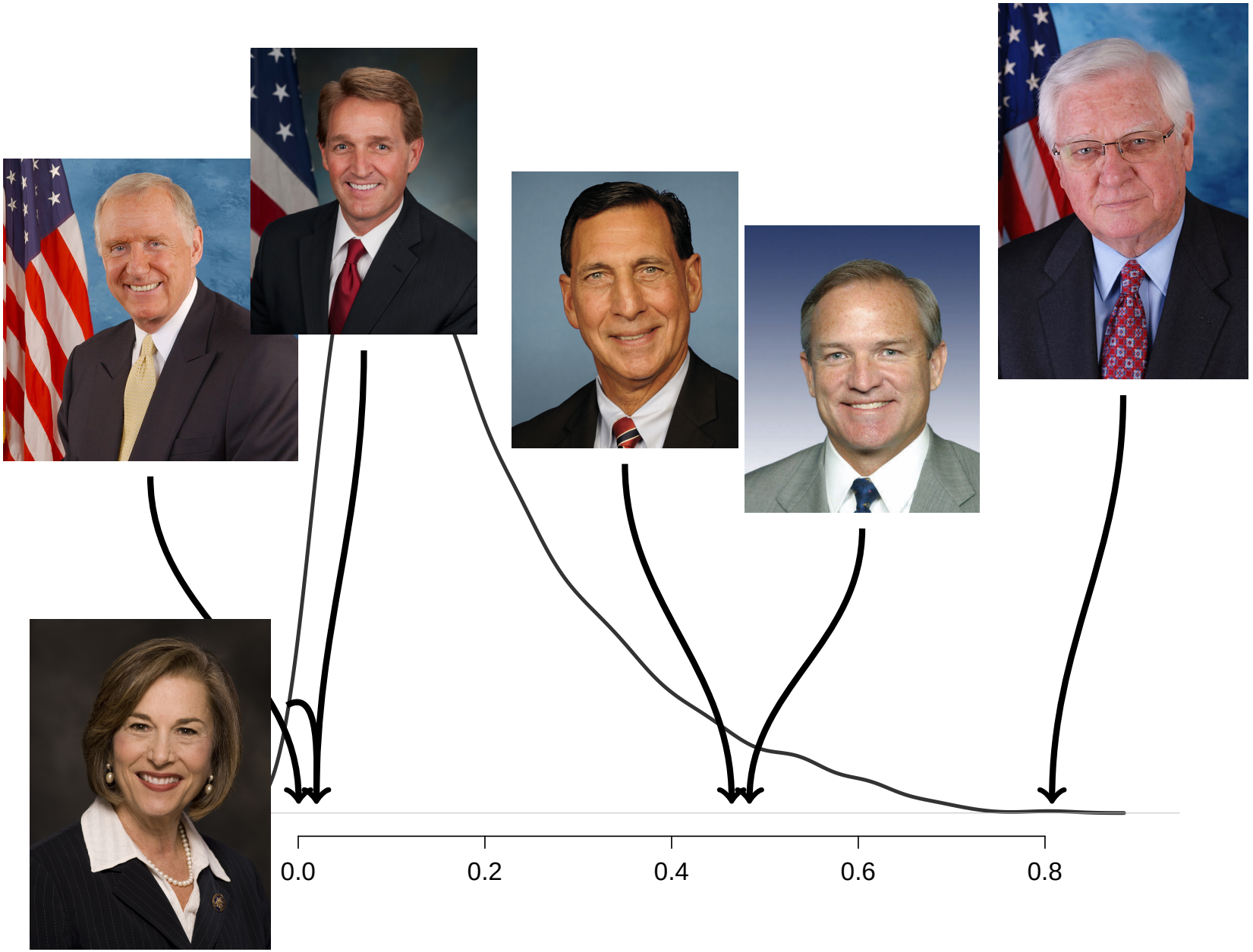
Strategic Credit Claiming to Build a Personal Vote



Strategic Credit Claiming to Build a Personal Vote

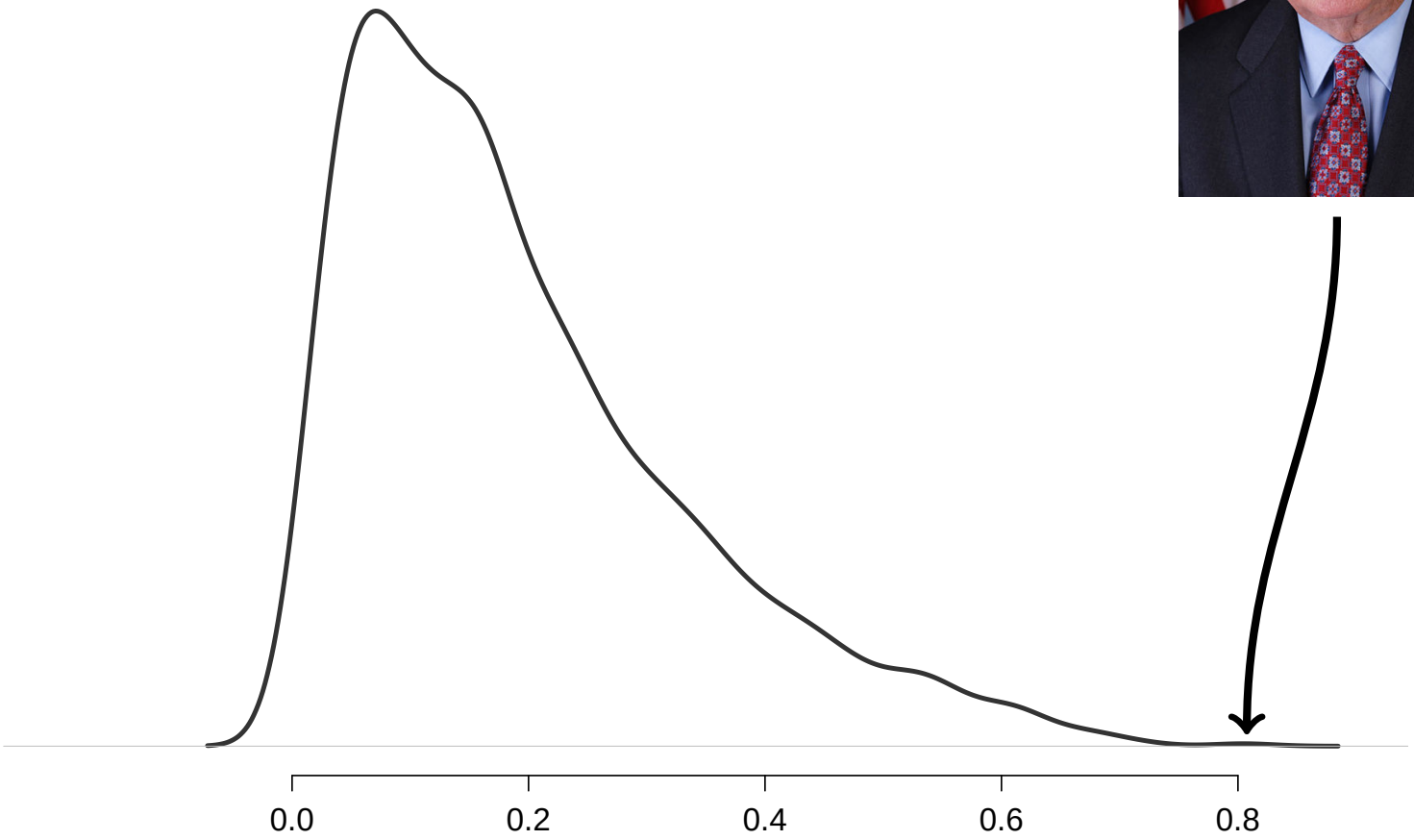


Strategic Credit Claiming to Build a Personal Vote



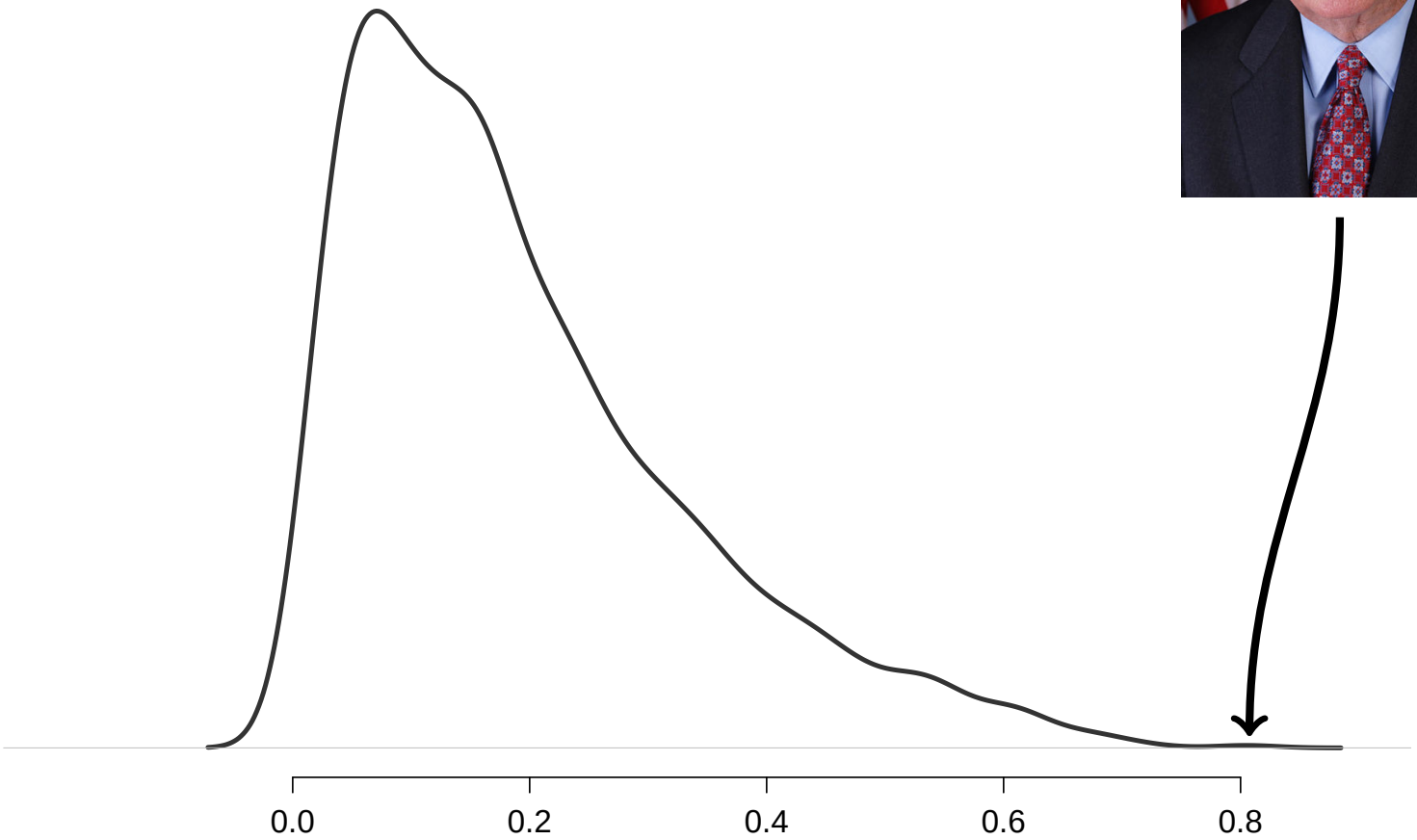
Strategic Credit Claiming to Build a Personal Vote

"We just can't afford luxuries like ideology"



Strategic Credit Claiming to Build a Personal Vote

Lexington Herald-Leader: Prince of Pork



Other Reasons to Ensemble (Dietterich 2000)

Statistical

Other Reasons to Ensemble (Dietterich 2000)

Statistical

- With little data, many algorithms offer similar performance

Other Reasons to Ensemble (Dietterich 2000)

Statistical

- With little data, many algorithms offer similar performance
- Ensemble ensures we avoid wrong model in test set

Other Reasons to Ensemble (Dietterich 2000)

Statistical

- With little data, many algorithms offer similar performance
- Ensemble ensures we avoid wrong model in test set

Computational

Other Reasons to Ensemble (Dietterich 2000)

Statistical

- With little data, many algorithms offer similar performance
- Ensemble ensures we avoid wrong model in test set

Computational

- Methods stuck in local modes

Other Reasons to Ensemble (Dietterich 2000)

Statistical

- With little data, many algorithms offer similar performance
- Ensemble ensures we avoid wrong model in test set

Computational

- Methods stuck in local modes
- Result: no one run provides best model

Other Reasons to Ensemble (Dietterich 2000)

Statistical

- With little data, many algorithms offer similar performance
- Ensemble ensures we avoid wrong model in test set

Computational

- Methods stuck in local modes
- Result: no one run provides best model
- Averages of runs may perform better

Other Reasons to Ensemble (Dietterich 2000)

Statistical

- With little data, many algorithms offer similar performance
- Ensemble ensures we avoid wrong model in test set

Computational

- Methods stuck in local modes
- Result: no one run provides best model
- Averages of runs may perform better

Complex "true" functional forms

Other Reasons to Ensemble (Dietterich 2000)

Statistical

- With little data, many algorithms offer similar performance
- Ensemble ensures we avoid wrong model in test set

Computational

- Methods stuck in local modes
- Result: no one run provides best model
- Averages of runs may perform better

Complex "true" functional forms

- One method may be unable to approximate true DGP

Other Reasons to Ensemble (Dietterich 2000)

Statistical

- With little data, many algorithms offer similar performance
- Ensemble ensures we avoid wrong model in test set

Computational

- Methods stuck in local modes
- Result: no one run provides best model
- Averages of runs may perform better

Complex "true" functional forms

- One method may be unable to approximate true DGP
- Mixtures of methods may approximate better

Other Reasons to Ensemble (Dietterich 2000)

Statistical

- With little data, many algorithms offer similar performance
- Ensemble ensures we avoid wrong model in test set

Computational

- Methods stuck in local modes
- Result: no one run provides best model
- Averages of runs may perform better

Complex "true" functional forms

- One method may be unable to approximate true DGP
- Mixtures of methods may approximate better