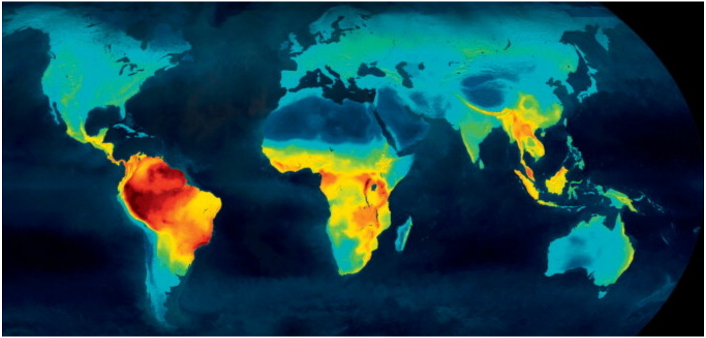
Ensemble Methods (Part I)
Bagging and Random Forests

Schwartz

September 12, 2016

Trees!

A recent study published in the Proceedings of the National Academy of Sciences estimates that there are somewhere between 40,000 to 50,000 unique tree species in the American neotropics and Indo-Pacific tropics, respectively, and that tropical Africa includes an additional approximately 10,000 species. This amount of tree diversity far outpaces tree diversity in temperate regions. For example, temperate forests in Europe have only 124 unique tree species and there are only approximately 1,000 tree species in North America. Another recent study published in Nature estimates that there are 3.04 trillion trees on earth – 422 per person – and that 45% of the trees are in the tropics. Tropical forests make up 2% of the earth's landmass. By area then, tropical jungles produce tree diversity at a rate of 5000:1 and tree density at a rate of 40:1 compared the rest of the earth's dry landmasses.



Objectives

- What is *Bagging*?

Objectives

- What is *Bagging*?
    - How do you do it?

Objectives

- What is *Bagging*?
  - How do you do it?
    - Why is it good?

Objectives

- What is *Bagging*?
  - How do you do it?
    - Why is it good?
- What are *Random Forests*?

Objectives

- What is *Bagging*?
  - How do you do it?
    - Why is it good?
- What are *Random Forests*?
  - How do you make them?

Objectives

- What is *Bagging*?
    - How do you do it?
        - Why is it good?
- What are *Random Forests*?
    - How do you make them?
        - Why might they be an improvement over bagging?

Objectives

- What is *Bagging*?
  - How do you do it?
    - Why is it good?
- What are *Random Forests*?
  - How do you make them?
    - Why might they be an improvement over bagging?

- What is "out of bag error" (OOB) relative to K-folds?

Objectives

- What is *Bagging*?
  - How do you do it?
    - Why is it good?
- What are *Random Forests*?
  - How do you make them?
    - Why might they be an improvement over bagging?

- What is “out of bag error” (OOB) relative to K-folds?
- How can one interpret feature importance in tree ensembles?

Objectives

- What is *Bagging*?
  - How do you do it?
    - Why is it good?
- What are *Random Forests*?
  - How do you make them?
    - Why might they be an improvement over bagging?

- What is "out of bag error" (OOB) relative to K-folds?
- How can one interpret feature importance in tree ensembles?

- Be able to implement tree-ensemble methods

Objectives

- What is *Bagging*?
  - How do you do it?
    - Why is it good?
- What are *Random Forests*?
  - How do you make them?
    - Why might they be an improvement over bagging?

- What is "out of bag error" (OOB) relative to K-folds?
- How can one interpret feature importance in tree ensembles?

- Be able to implement tree-ensemble methods
- Review what you know about trees

Single predictors

- A single tree is pretty great!

Single predictors

- A single tree is pretty great!
- Well, it's not that good, actually

Single predictors

- A single tree is pretty great!
- Well, it's not that good, actually
  What could be wrong with it?

Single predictors

- A single tree is pretty great!
- Well, it's not that good, actually

  What could be wrong with it?

  Biased?

  High variance?

Single predictors

- A single tree is pretty great!
- Well, it's not that good, actually

  What could be wrong with it?

  Biased?
  High variance?
  How so with trees?

Single predictors

- A single tree is pretty great!
- Well, it's not that good, actually

  What could be wrong with it?

  Biased?
  High variance?
  How so with trees?

  Do we think a tree prediction is a good one?

Single predictors

- A single tree is pretty great!
- Well, it's not that good, actually

  What could be wrong with it?

  Biased?
  High variance?
  How so with trees?

  Do we think a tree prediction is a good one?

- What if we had a lot of prediction trees

Single predictors

- A single tree is pretty great!
- Well, it's not that good, actually

  What could be wrong with it?

  Biased?
  High variance?
  How so with trees?

  Do we think a tree prediction is a good one?

- What if we had a lot of prediction trees *and averaged them?*

Challenge

Suppose you are trying to predict the election...

You have 5 expert opions, each with a 70% chance of being right, and each experts pick is *independent* of the other experts pick.

How could you leverage expert picks to improve accuracy and how often would you be right?

Challenge

Suppose you are trying to predict the election...

You have 5 expert opions, each with a 70% chance of being right, and each experts pick is *independent* of the other experts pick.

How could you leverage expert picks to improve accuracy and how often would you be right?

How many independent predictors would you need to get a 99.9% accuracy?

Averaging Predictions

- Let $\hat{f}^{(j)}(\boldsymbol{x}_0)$ be an estimator of outcome $f(\boldsymbol{x}_0)$ constructed from a training set with features $\boldsymbol{x}^{(j)}$ and outcomes $\boldsymbol{Y}^{(j)}$

Averaging Predictions

- Let $\hat{f}^{(j)}(\boldsymbol{x}_0)$ be an estimator of outcome $f(\boldsymbol{x}_0)$ constructed from a training set with features $\boldsymbol{x}^{(j)}$ and outcomes $\boldsymbol{Y}^{(j)}$
- If each $\hat{f}^{(j)}(\boldsymbol{x}_0)$ is an unbiased estimator of $f(\boldsymbol{x}_0)$ then

$$\mathrm{E}\left[\sum \frac{1}{m}\hat{f}^{(j)}(\boldsymbol{x}_0)\right] = f(\boldsymbol{x}_0)$$

Averaging Predictions

- Let $\hat{f}^{(j)}(\boldsymbol{x}_0)$ be an estimator of outcome $f(\boldsymbol{x}_0)$ constructed from a training set with features $\boldsymbol{x}^{(j)}$ and outcomes $\boldsymbol{Y}^{(j)}$
- If each $\hat{f}^{(j)}(\boldsymbol{x}_0)$ is an unbiased estimator of $f(\boldsymbol{x}_0)$ then

$$\mathrm{E}\left[\sum \frac{1}{m}\hat{f}^{(j)}(\boldsymbol{x}_0)\right] = f(\boldsymbol{x}_0)$$

- And if $\mathrm{Var}\left[\hat{f}^{(j)}(\boldsymbol{x}_0)\right] = \sigma^2$, then

$$\mathrm{Var}\left[\sum \frac{1}{m}\hat{f}^{(j)}(\boldsymbol{x}_0)\right] = \ ?$$

Averaging Predictions

- Let $\hat{f}^{(j)}(\boldsymbol{x}_0)$ be an estimator of outcome $f(\boldsymbol{x}_0)$ constructed from a training set with features $\boldsymbol{x}^{(j)}$ and outcomes $\boldsymbol{Y}^{(j)}$
- If each $\hat{f}^{(j)}(\boldsymbol{x}_0)$ is an unbiased estimator of $f(\boldsymbol{x}_0)$ then

$$\mathrm{E}\left[\sum \frac{1}{m}\hat{f}^{(j)}(\boldsymbol{x}_0)\right] = f(\boldsymbol{x}_0)$$

- And if $\mathrm{Var}\left[\hat{f}^{(j)}(\boldsymbol{x}_0)\right] = \sigma^2$, then

$$\mathrm{Var}\left[\sum \frac{1}{m}\hat{f}^{(j)}(\boldsymbol{x}_0)\right] = \; ?$$

$$\frac{\sigma^2}{m}?$$

Bootstrapping Aggregation (Bagging)

- *Bagging* is simply estimating $f(\boldsymbol{x}_0)$ with $\frac{1}{m}\sum \hat{f}^{(j)}(\boldsymbol{x}_0)$

Bootstrapping Aggregation (Bagging)

- *Bagging* is simply estimating $f(\boldsymbol{x}_0)$ with $\frac{1}{m}\sum \hat{f}^{(j)}(\boldsymbol{x}_0)$
- I.e., the *average* of a bunch of predictions

Bootstrapping Aggregation (Bagging)

- *Bagging* is simply estimating $f(\boldsymbol{x}_0)$ with $\frac{1}{m}\sum \hat{f}^{(j)}(\boldsymbol{x}_0)$
- I.e., the *average* of a bunch of predictions
- Because averages are more stable than a single prediction*

Bootstrapping Aggregation (Bagging)

- *Bagging* is simply estimating $f(\boldsymbol{x}_0)$ with $\frac{1}{m}\sum \hat{f}^{(j)}(\boldsymbol{x}_0)$
- I.e., the *average* of a bunch of predictions
- Because averages are more stable than a single prediction*

- What are the $\hat{f}^{(j)}$'s? I.e, what are $\boldsymbol{x}^{(j)}$'s and $\boldsymbol{Y}^{(j)}$'s?

Bootstrapping Aggregation (Bagging)

- *Bagging* is simply estimating $f(\boldsymbol{x}_0)$ with $\frac{1}{m}\sum \hat{f}^{(j)}(\boldsymbol{x}_0)$
- I.e., the *average* of a bunch of predictions
- Because averages are more stable than a single prediction*

- What are the $\hat{f}^{(j)}$'s? I.e, what are $\boldsymbol{x}^{(j)}$'s and $\boldsymbol{Y}^{(j)}$'s?

- $\boldsymbol{x}^{(j)}$ is a *bootstrapped* sample from $\boldsymbol{X}$

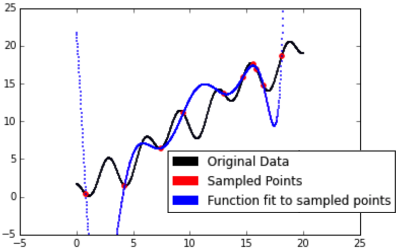
Bootstrapping Aggregation (Bagging)

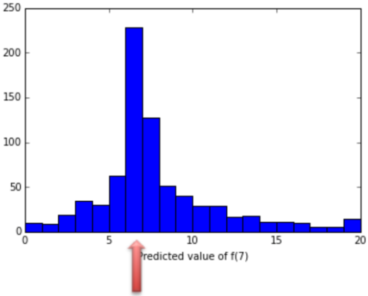
- *Bagging* is simply estimating $f(\boldsymbol{x}_0)$ with $\frac{1}{m}\sum \hat{f}^{(j)}(\boldsymbol{x}_0)$
- I.e., the *average* of a bunch of predictions
- Because averages are more stable than a single prediction*

- What are the $\hat{f}^{(j)}$'s? I.e, what are $\boldsymbol{x}^{(j)}$'s and $\boldsymbol{Y}^{(j)}$'s?

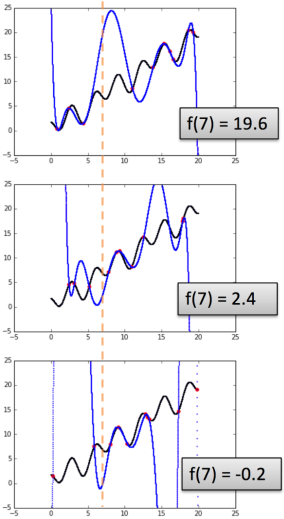
- $\boldsymbol{x}^{(j)}$ is a *bootstrapped* sample from $\boldsymbol{X}$
- $\hat{f}^{(j)}$ is the model fit on that bootstrapped training set

Why does this work?





Actual value: f(7) = 6.8



"Out of Bag" (OOB) testing error

- Let $\boldsymbol{x}_0^{(j)}$ and $\boldsymbol{Y}_0^{(j)}$ be the features and outcomes not in $\boldsymbol{x}^{(j)}$
- OOB prediction error is calculated from

“Out of Bag” (OOB) testing error

- Let $\boldsymbol{x}_0^{(j)}$ and $\boldsymbol{Y}_0^{(j)}$ be the features and outcomes not in $\boldsymbol{x}^{(j)}$
- OOB prediction error is calculated from

  the prediction $\hat{\boldsymbol{Y}}_0^{(j)}$ from $\hat{f}^{(j)}\left(\boldsymbol{x}_0^{(j)}\right)$ versus the actual $\boldsymbol{Y}_0^{(j)}$:

$$\boldsymbol{Y}_0^{(j)} \text{ vs } \hat{f}^{(j)}\left(\boldsymbol{x}_0^{(j)}\right)$$

"Out of Bag" (OOB) testing error

- Let $\boldsymbol{x}_0^{(j)}$ and $\boldsymbol{Y}_0^{(j)}$ be the features and outcomes not in $\boldsymbol{x}^{(j)}$
- OOB prediction error is calculated from

  the prediction $\hat{\boldsymbol{Y}}_0^{(j)}$ from $\hat{f}^{(j)}\left(\boldsymbol{x}_0^{(j)}\right)$ versus the actual $\boldsymbol{Y}_0^{(j)}$:

$$\boldsymbol{Y}_0^{(j)} \text{ vs } \hat{f}^{(j)}\left(\boldsymbol{x}_0^{(j)}\right)$$

- When the number of bootstrapped samples $j = 1, \cdots m$ is large, this closely approximates LOO test error estimation

"Out of Bag" (OOB) testing error

- Let $\boldsymbol{x}_0^{(j)}$ and $\boldsymbol{Y}_0^{(j)}$ be the features and outcomes not in $\boldsymbol{x}^{(j)}$
- OOB prediction error is calculated from

  the prediction $\hat{\boldsymbol{Y}}_0^{(j)}$ from $\hat{f}^{(j)}\left(\boldsymbol{x}_0^{(j)}\right)$ versus the actual $\boldsymbol{Y}_0^{(j)}$:

$$\boldsymbol{Y}_0^{(j)} \text{ vs } \hat{f}^{(j)}\left(\boldsymbol{x}_0^{(j)}\right)$$

- When the number of bootstrapped samples $j = 1, \cdots m$ is large, this closely approximates LOO test error estimation

  So test error estimation is available by default

“Out of Bag” (OOB) testing error

- Let $\boldsymbol{x}_0^{(j)}$ and $\boldsymbol{Y}_0^{(j)}$ be the features and outcomes not in $\boldsymbol{x}^{(j)}$
- OOB prediction error is calculated from

  the prediction $\hat{\boldsymbol{Y}}_0^{(j)}$ from $\hat{f}^{(j)}\left(\boldsymbol{x}_0^{(j)}\right)$ versus the actual $\boldsymbol{Y}_0^{(j)}$:

$$\boldsymbol{Y}_0^{(j)} \text{ vs } \hat{f}^{(j)}\left(\boldsymbol{x}_0^{(j)}\right)$$

- When the number of bootstrapped samples $j = 1, \cdots m$ is large, this closely approximates LOO test error estimation

  So test error estimation is available by default

Bootstrapped samples of size n leave out $\sim \frac{1}{3}$ of the data

But wait...

$$\mathrm{Var}\left[\sum \frac{1}{m}\hat{f}^{(j)}(\boldsymbol{x}_0)\right] \neq \frac{\sigma^2}{m}$$

But wait...

$$\mathrm{Var}\left[\sum \frac{1}{m}\hat{f}^{(j)}(\boldsymbol{x}_0)\right] \neq \frac{\sigma^2}{m}$$

- Why?

But wait...

$$\mathrm{Var}\left[\sum \frac{1}{m}\hat{f}^{(j)}(\boldsymbol{x}_0)\right] \neq \frac{\sigma^2}{m}$$

- Why?

$$\mathrm{Var}[X+Y] = \mathrm{Var}[X] + \mathrm{Var}[Y] + 2\mathrm{Cov}[X, Y]$$

I.e.,

$$\begin{aligned}\mathrm{Var}\left[f^{(j)}(\boldsymbol{x}_0) + f^{(k)}(\boldsymbol{x}_0)\right] &= \mathrm{Var}\left[f^{(j)}(\boldsymbol{x}_0)\right] + \mathrm{Var}\left[f^{(k)}(\boldsymbol{x}_0)\right] \\ &\quad + 2\mathrm{Cov}\left[f^{(j)}(\boldsymbol{x}_0), f^{(k)}(\boldsymbol{x}_0)\right]\end{aligned}$$

$f^{(j)}(\boldsymbol{x}_0)$ and $f^{(k)}(\boldsymbol{x}_0)$ correlated?

$f^{(j)}(\boldsymbol{x}_0)$ and $f^{(k)}(\boldsymbol{x}_0)$ correlated?

- $f^{(j)}$ is estimated from $\boldsymbol{x}^{(j)}$

$f^{(j)}(\boldsymbol{x}_0)$ and $f^{(k)}(\boldsymbol{x}_0)$ correlated?

- $f^{(j)}$ is estimated from $\boldsymbol{x}^{(j)}$
- $f^{(k)}$ is estimated from $\boldsymbol{x}^{(k)}$

$f^{(j)}(\boldsymbol{x}_0)$ and $f^{(k)}(\boldsymbol{x}_0)$ correlated?

- $f^{(j)}$ is estimated from $\boldsymbol{x}^{(j)}$
- $f^{(k)}$ is estimated from $\boldsymbol{x}^{(k)}$

- $\boldsymbol{x}^{(j)}$ and $\boldsymbol{x}^{(k)}$ are bootstrapped from *the same* $\boldsymbol{X}$

$f^{(j)}(\boldsymbol{x}_0)$ and $f^{(k)}(\boldsymbol{x}_0)$ correlated?

- $f^{(j)}$ is estimated from $\boldsymbol{x}^{(j)}$
- $f^{(k)}$ is estimated from $\boldsymbol{x}^{(k)}$

- $\boldsymbol{x}^{(j)}$ and $\boldsymbol{x}^{(k)}$ are bootstrapped from *the same* $\boldsymbol{X}$
- So they are very similar samples (indeed, they overlap)

How good is bagging?

- If $\mathrm{Cor}\left[f^{(j)}, f^{(k)}\right] = \rho$ for all j and k

How good is bagging?

- If $\mathrm{Cor}\left[f^{(j)}, f^{(k)}\right] = \rho$ for all j and k then
$$\mathrm{Var}\left[\sum f^{(j)}(\boldsymbol{x}_0)\right] = \rho\sigma^2 + (1-\rho)\frac{\sigma^2}{m}$$

How good is bagging?

- If $\text{Cor}\left[f^{(j)}, f^{(k)}\right] = \rho$ for all j and k then
$$\text{Var}\left[\sum f^{(j)}(\boldsymbol{x}_0)\right] = \rho\sigma^2 + (1-\rho)\frac{\sigma^2}{m}$$
- Only "uncorrelated parts" of $f^{(j)}$ and $f^{(k)}$ get "CLT effect"

How good is bagging?

- If $\mathrm{Cor}\left[f^{(j)}, f^{(k)}\right] = \rho$ for all j and k

  then
$$\mathrm{Var}\left[\sum f^{(j)}(\boldsymbol{x}_0)\right] = \rho\sigma^2 + (1-\rho)\frac{\sigma^2}{m}$$
- Only "uncorrelated parts" of $f^{(j)}$ and $f^{(k)}$ get "CLT effect"
- So is there any way to get ρ close to 0?

Decorrelating trees

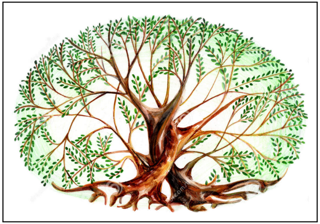
We might try to decorrelate any class of predictors $\hat{f}^{(j)}$

But let's focus on 

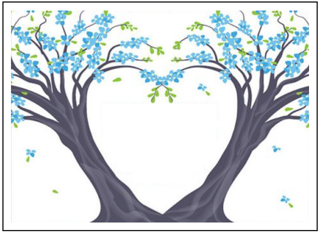
Decorrelating trees

We might try to decorrelate any class of predictors $\hat{f}^{(j)}$

But let's focus on 



$$\Downarrow$$

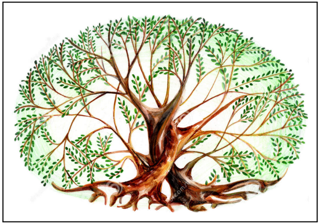


Decorrelating trees

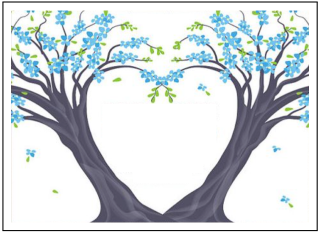
We might try to decorrelate any class of predictors $\hat{f}^{(j)}$

But let's focus on 

Trees are "correlated" because



$$\Downarrow$$



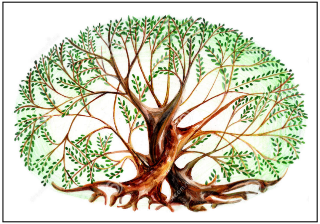
Decorrelating trees

We might try to decorrelate any class of predictors $\hat{f}^{(j)}$

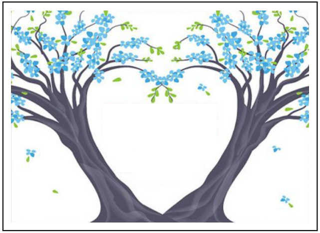
But let's focus on 

Trees are "correlated" because

1. bootstrap samples are about the same [nothing to do there]



$$\Downarrow$$



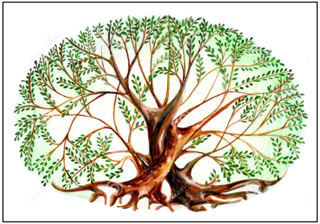
Decorrelating trees

We might try to decorrelate any class of predictors $\hat{f}^{(j)}$

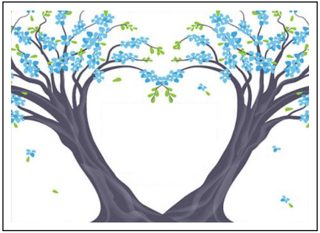
But let's focus on 

Trees are "correlated" because

1. bootstrap samples are about the same [nothing to do there]
2. influential features tend to be same [this we could influence]



$\Downarrow$



Decorrelating trees

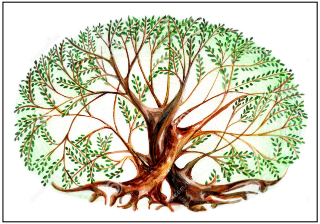
We might try to decorrelate any class of predictors $\hat{f}^{(j)}$

But let's focus on 

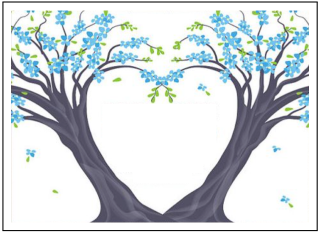
Trees are "correlated" because

1. bootstrap samples are about the same [nothing to do there]
2. influential features tend to be same [this we could influence]

How can we make constructed tree not exactly the same?



$\Downarrow$



Random Forests

- What is the tree construction process?

Random Forests

- What is the tree construction process?
- Create branch by "splitting on" a feature.

Random Forests

- What is the tree construction process?
- Create branch by "splitting on" a feature.
- What if we restrict the features that can be used to split?

Random Forests

- What is the tree construction process?
- Create branch by "splitting on" a feature.
- What if we restrict the features that can be used to split?
- I.e., consider only a random set of features **at each stage**?

Random Forests

- What is the tree construction process?
- Create branch by "splitting on" a feature.
- What if we restrict the features that can be used to split?
- I.e., consider only a random set of features **at each stage**?

- Typically we consider $\sqrt{p}$ and $\frac{p}{3}$ of the p predictors for classification and regression, respectively **at each stage**

Random Forests

- What is the tree construction process?
- Create branch by "splitting on" a feature.
- What if we restrict the features that can be used to split?
- I.e., consider only a random set of features **at each stage**?

- Typically we consider $\sqrt{p}$ and $\frac{p}{3}$ of the p predictors for classification and regression, respectively **at each stage**

- This could be a tuning parameter...

Random Forests

- What is the tree construction process?
- Create branch by "splitting on" a feature.
- What if we restrict the features that can be used to split?
- I.e., consider only a random set of features **at each stage**?

- Typically we consider $\sqrt{p}$ and $\frac{p}{3}$ of the p predictors for classification and regression, respectively **at each stage**

- This could be a tuning parameter...
- More features means stronger, but more correlated trees...

Random Forests

- What is the tree construction process?
- Create branch by "splitting on" a feature.
- What if we restrict the features that can be used to split?
- I.e., consider only a random set of features **at each stage**?

- Typically we consider $\sqrt{p}$ and $\frac{p}{3}$ of the p predictors for classification and regression, respectively **at each stage**

- This could be a tuning parameter...
- More features means stronger, but more correlated trees...
- But features excluded at one level could be included later...

Random Forest *Tuning*

$g(p)$: Number of features considered at each split

Random Forest *Tuning*

$g(p)$: Number of features considered at each split

m: Number of trees (i.e., number of bootstrapped samples)

Random Forest *Tuning*

$g(p)$: Number of features considered at each split

m: Number of trees (i.e., number of bootstrapped samples)

n_b: Sample size of each bootstrapped sample

Random Forest *Tuning*

$g(p)$: Number of features considered at each split

m: Number of trees (i.e., number of bootstrapped samples)

n_b: Sample size of each bootstrapped sample

χ: Tree characteristics $\Longrightarrow \sum(\boldsymbol{Y}_i - \hat{\boldsymbol{Y}}_i)^2 + \alpha|T|$

- Number of nodes (T), minimum node size, feature subsets?

Random Forest *Tuning*

$g(p)$: Number of features considered at each split

m: Number of trees (i.e., number of bootstrapped samples)

n_b: Sample size of each bootstrapped sample

χ: Tree characteristics $\Longrightarrow \sum(\boldsymbol{Y}_i - \hat{\boldsymbol{Y}}_i)^2 + \alpha|T|$

- Number of nodes (T), minimum node size, feature subsets?

- Trees can be computationally expensive to fit

Random Forest *Tuning*

$g(p)$: Number of features considered at each split

m: Number of trees (i.e., number of bootstrapped samples)

n_b: Sample size of each bootstrapped sample

χ: Tree characteristics $\Longrightarrow \sum(\boldsymbol{Y}_i - \hat{\boldsymbol{Y}}_i)^2 + \alpha|T|$

- Number of nodes (T), minimum node size, feature subsets?

- Trees can be computationally expensive to fit
  - But they can be fit in parallel

Random Forest *Tuning*

$g(p)$: Number of features considered at each split

m: Number of trees (i.e., number of bootstrapped samples)

n_b: Sample size of each bootstrapped sample

χ: Tree characteristics $\Longrightarrow \sum(\boldsymbol{Y}_i - \hat{\boldsymbol{Y}}_i)^2 + \alpha|T|$

- Number of nodes (T), minimum node size, feature subsets?

- Trees can be computationally expensive to fit
  - But they can be fit in parallel
  - At some point $\rho\sigma^2 + (1-\rho)\frac{\sigma^2}{m}$ is reached

Random Forest *Tuning*

$g(p)$: Number of features considered at each split

m: Number of trees (i.e., number of bootstrapped samples)

n_b: Sample size of each bootstrapped sample

χ: Tree characteristics $\Longrightarrow \sum(\boldsymbol{Y}_i - \hat{\boldsymbol{Y}}_i)^2 + \alpha|T|$

- Number of nodes (T), minimum node size, feature subsets?

- Trees can be computationally expensive to fit
  - But they can be fit in parallel
  - At some point $\rho\sigma^2 + (1-\rho)\frac{\sigma^2}{m}$ is reached
- Cross-validation can be computationally demanding

Random Forest *Tuning*

$g(p)$: Number of features considered at each split

m: Number of trees (i.e., number of bootstrapped samples)

n_b: Sample size of each bootstrapped sample

χ: Tree characteristics $\implies \sum(\boldsymbol{Y}_i - \hat{\boldsymbol{Y}}_i)^2 + \alpha|T|$

- Number of nodes (T), minimum node size, feature subsets?

- Trees can be computationally expensive to fit
  - But they can be fit in parallel
  - At some point $\rho\sigma^2 + (1-\rho)\frac{\sigma^2}{m}$ is reached
- Cross-validation can be computationally demanding
  - Parameters can be tuned using OOB

Random Forest *Tuning*

$g(p)$: Number of features considered at each split

m: Number of trees (i.e., number of bootstrapped samples)

n_b: Sample size of each bootstrapped sample

χ: Tree characteristics $\Longrightarrow \sum(\boldsymbol{Y}_i - \hat{\boldsymbol{Y}}_i)^2 + \alpha|T|$

- Number of nodes (T), minimum node size, feature subsets?

- Trees can be computationally expensive to fit
  - But they can be fit in parallel
  - At some point $\rho\sigma^2 + (1-\rho)\frac{\sigma^2}{m}$ is reached
- Cross-validation can be computationally demanding
  - Parameters can be tuned using OOB
  - Cross-validation is still needed for methodology comparisons...

Random Forest *Tuning*

$g(p)$: Number of features considered at each split

m: Number of trees (i.e., number of bootstrapped samples)

n_b: Sample size of each bootstrapped sample

χ: Tree characteristics $\Longrightarrow \sum(\mathbf{Y}_i - \hat{\mathbf{Y}}_i)^2 + \alpha|T|$

- Number of nodes (T), minimum node size, feature subsets?

- Trees can be computationally expensive to fit
  - But they can be fit in parallel
  - At some point $\rho\sigma^2 + (1-\rho)\frac{\sigma^2}{m}$ is reached
- Cross-validation can be computationally demanding
  - Parameters can be tuned using OOB
  - Cross-validation is still needed for methodology comparisons...

- Random forests are quite robust to these choices

Random Forest *Tuning*

$g(p)$: Number of features considered at each split

m: Number of trees (i.e., number of bootstrapped samples)

n_b: Sample size of each bootstrapped sample

χ: Tree characteristics $\Longrightarrow \sum(\boldsymbol{Y}_i - \hat{\boldsymbol{Y}}_i)^2 + \alpha|T|$

- Number of nodes (T), minimum node size, feature subsets?

- Trees can be computationally expensive to fit
  - But they can be fit in parallel
  - At some point $\rho\sigma^2 + (1-\rho)\frac{\sigma^2}{m}$ is reached
- Cross-validation can be computationally demanding
  - Parameters can be tuned using OOB
  - Cross-validation is still needed for methodology comparisons...

- Random forests are quite robust to these choices
- A *very large* number of "bushy" trees *do not overfit*

Random Forest *Tuning*

$g(p)$: Number of features considered at each split

m: Number of trees (i.e., number of bootstrapped samples)

n_b: Sample size of each bootstrapped sample

χ: Tree characteristics $\implies \sum(\boldsymbol{Y}_i - \hat{\boldsymbol{Y}}_i)^2 + \alpha|T|$

- Number of nodes (T), minimum node size, feature subsets?

- Trees can be computationally expensive to fit
  - But they can be fit in parallel
  - At some point $\rho\sigma^2 + (1-\rho)\frac{\sigma^2}{m}$ is reached
- Cross-validation can be computationally demanding
  - Parameters can be tuned using OOB
  - Cross-validation is still needed for methodology comparisons...

- Random forests are quite robust to these choices
- A *very large* number of "bushy" trees *do not overfit*

  and approach "state of the art" performance "out of the box"

Interpreting Tree Ensembles

"Plentie is nodeintie, ye see not your owne ease. I see, ye can not see the wood for trees."

– J. Heywood (1546)

"You can't see the forest for the treeeeeeees!"

– M. Manson (1996)

Variable Importance (v1.0)

1. Make split on feature V that minimizes RSS or Gini/Entropy

Variable Importance (v1.0)

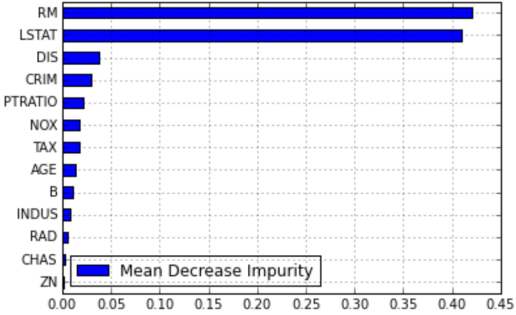
1. Make split on feature V that minimizes RSS or Gini/Entropy
2. For split s, record the absolute explanatory contribution $\xi_V^{(s)}$

Variable Importance (v1.0)

1. Make split on feature V that minimizes RSS or Gini/Entropy
2. For split s, record the absolute explanatory contribution $\xi_V^{(s)}$
3. Average the scores $\xi_V^{(s)}$ for each feature V **across all trees**

Variable Importance (v1.0)

1. Make split on feature V that minimizes RSS or Gini/Entropy
2. For split s, record the absolute explanatory contribution $\xi_V^{(s)}$
3. Average the scores $\xi_V^{(s)}$ for each feature V **across all trees**



In Boston, the most relevant associations with neighborhood home values are (1) number of rooms and (2) proportion of low income households in the neighborhood

Variable Importance (v2.0)

1. Predict OOB accuracy using all features, including original V

Variable Importance (v2.0)

1. Predict OOB accuracy using all features, including original V
2. Predict OOB accuracy using all features & V^*, a permuted V

Variable Importance (v2.0)

1. Predict OOB accuracy using all features, including original V
2. Predict OOB accuracy using all features & V^*, a permuted V
3. Compare the differences between 1 and 2 for all features

Variable Importance (v2.0)

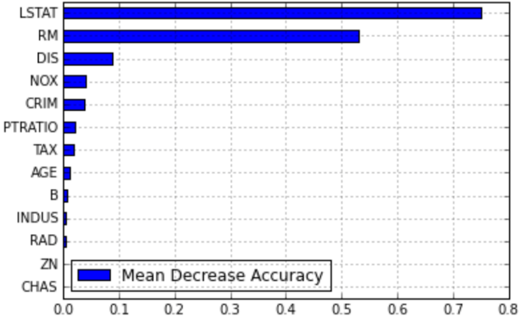
1. Predict OOB accuracy using all features, including original V
2. Predict OOB accuracy using all features & V^*, a permuted V
3. Compare the differences between 1 and 2 for all features

   Results are predicted fitted on trees with V/V^* and averaged

Variable Importance (v2.0)

1. Predict OOB accuracy using all features, including original V
2. Predict OOB accuracy using all features & V^*, a permuted V
3. Compare the differences between 1 and 2 for all features

   Results are predicted fitted on trees with V/V^* and averaged



This approach suggests prediction is more sensitive to (1) low income proportion rather than (2) room number

Variable Importance (v3.0)

1. Follow test samples down the tree through each split s on V

Variable Importance (v3.0)

1. Follow test samples down the tree through each split s on V
2. Record the change in prediction $\epsilon_V^{(s)}$ due to V split at s

Variable Importance (v3.0)

1. Follow test samples down the tree through each split s on V
2. Record the change in prediction $\epsilon_V^{(s)}$ due to V split at s
3. Sum up all the changes $\epsilon_V^{(s)}$ due to feature V

Variable Importance (v3.0)

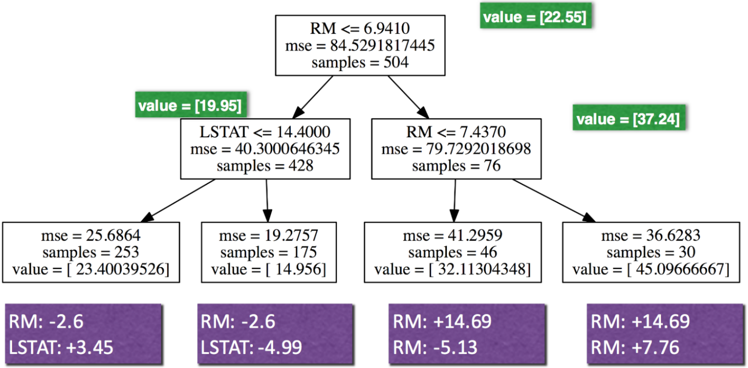
1. Follow test samples down the tree through each split s on V
2. Record the change in prediction $\epsilon_V^{(s)}$ due to V split at s
3. Sum up all the changes $\epsilon_V^{(s)}$ due to feature V

   These values are then averaged by tree

Variable Importance (v3.0)

1. Follow test samples down the tree through each split s on V
2. Record the change in prediction $\epsilon_V^{(s)}$ due to V split at s
3. Sum up all the changes $\epsilon_V^{(s)}$ due to feature V

   These values are then averaged by tree



Variable Importance (v3.0)

1. Follow test samples down the tree through each split s on V
2. Record the change in prediction $\epsilon_V^{(s)}$ due to V split at s
3. Sum up all the changes $\epsilon_V^{(s)}$ due to feature V

   These values are then averaged by tree

Distance from city-center

Fraction of lower-class residents

Average number of rooms

	DIS	INDUS	LSTAT	NOX	PTRATIO	RAD	RM	TAX	ZN
Prediction 1	6.11	0.10	2.67	-0.02	-0.19	0.06	-2.63	-0.20	0.03
Prediction 2	6.22	0.16	2.56	-0.01	-0.19	0.06	-3.15	-0.16	0.03
Prediction 3	-0.70	0.04	7.42	-0.11	0.42	-0.02	1.10	-0.14	-0.05
Prediction 4	-0.53	0.25	3.50	0.16	1.46	0.13	2.21	-0.24	0.11
Prediction 5	-0.68	0.15	7.86	0.03	0.85	0.01	-1.14	-0.16	0.18
Prediction 6	0.18	-0.26	8.62	-0.19	-0.02	-0.07	-1.83	-0.34	-0.05

Features effects can be characterized on individual samples (!)

Variable Importance (v4.0)

1. Calculate proportion of samples visiting feature V in each tree

Variable Importance (v4.0)

1. Calculate proportion of samples visiting feature V in each tree (conditions higher in the tree typically visited by more data)

Variable Importance (v4.0)

1. Calculate proportion of samples visiting feature V in each tree (conditions higher in the tree typically visited by more data)
2. Average the “proportion of samples visited” across all trees

Tree $<$ **Bagging** $<$ **Random Forests** $<$ **________?**