CS 412

APR 9TH – RANDOM FORESTS

+concentration bounds

HW4
out tonight/
tomorrow
Ensemble
+
Testing

< 2hrs

Midterm Exam

Midterm, Thursday April 9th (Today), 12-8pm CDT

on piazza

- Download from piazza
- Submit to gradescope
- No late submissions
- Lots of explanations, please don't skimp here

open-book, open-note

4 questions

- Short Answer (8 parts)
- SVM
- NN
- Boosting

small coding portion

for the exam on slack #exam

NO OH

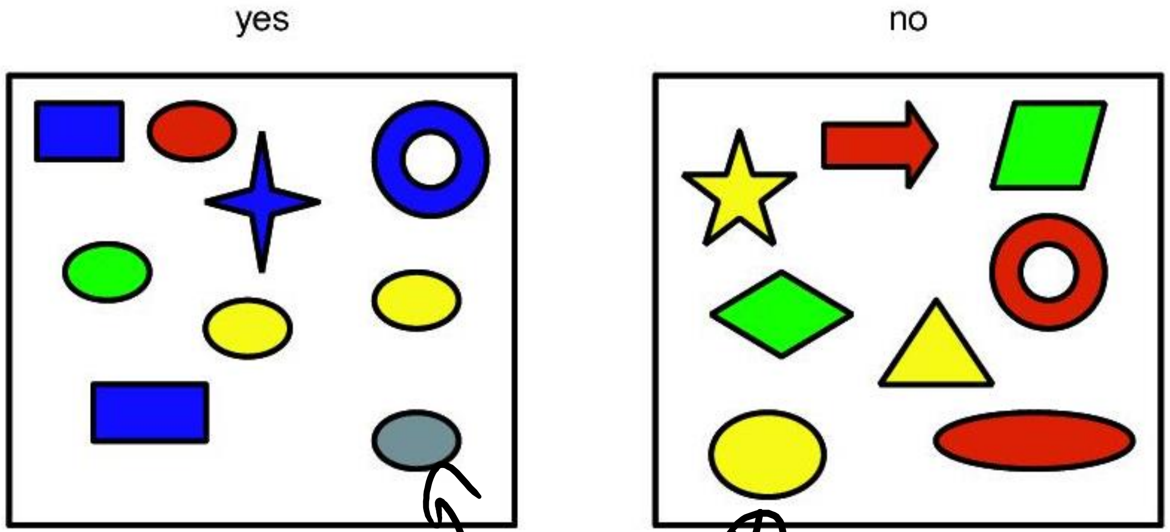
Please post clarification Q's publicly

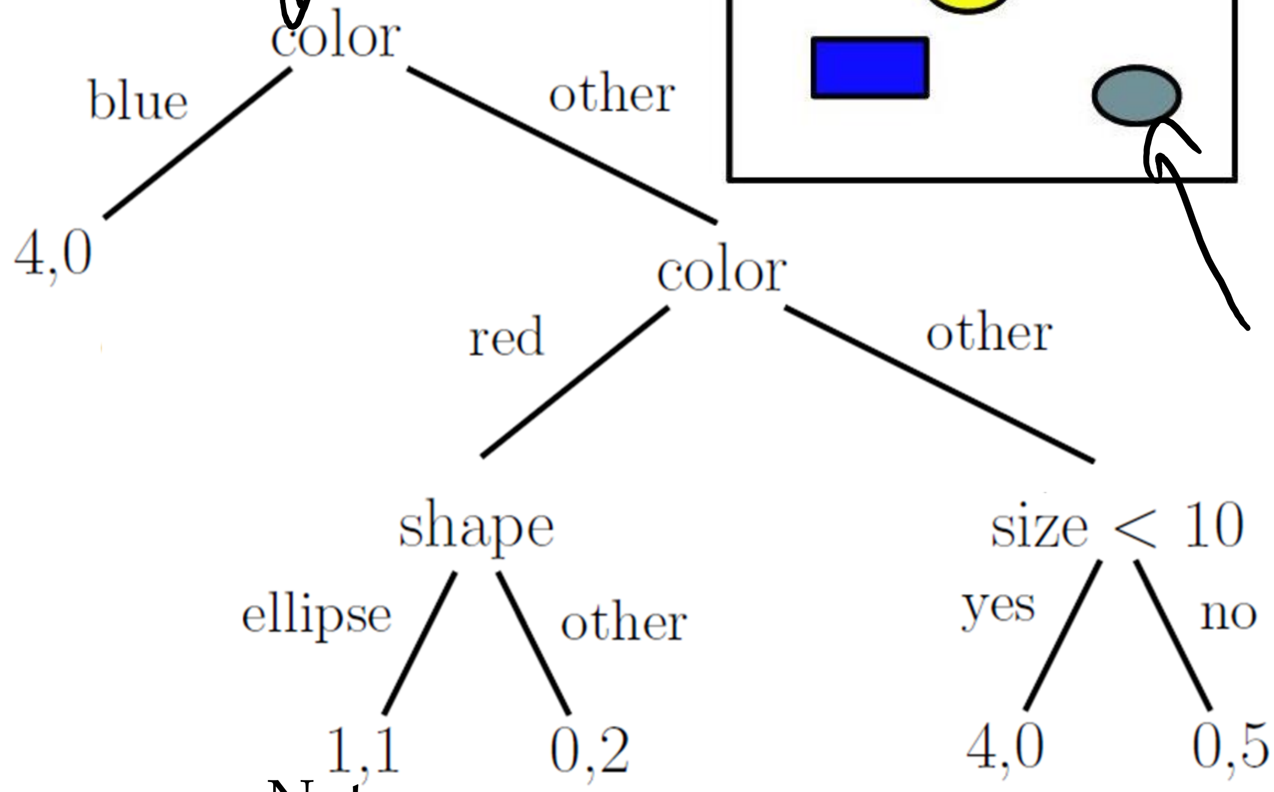
Decision Trees

"20 questions game for each possible outcome"

Nodes: test the value of feature x_j, branch based on result

Leafs: provide the class (prediction)


yes
no

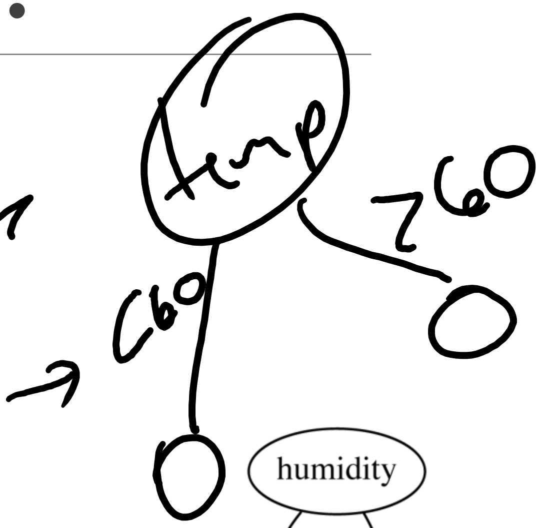

color
blue
other
4,0
color
red
other
shape
size < 10
ellipse
other
yes
no
1,1
0,2
4,0
0,5

Not
pure

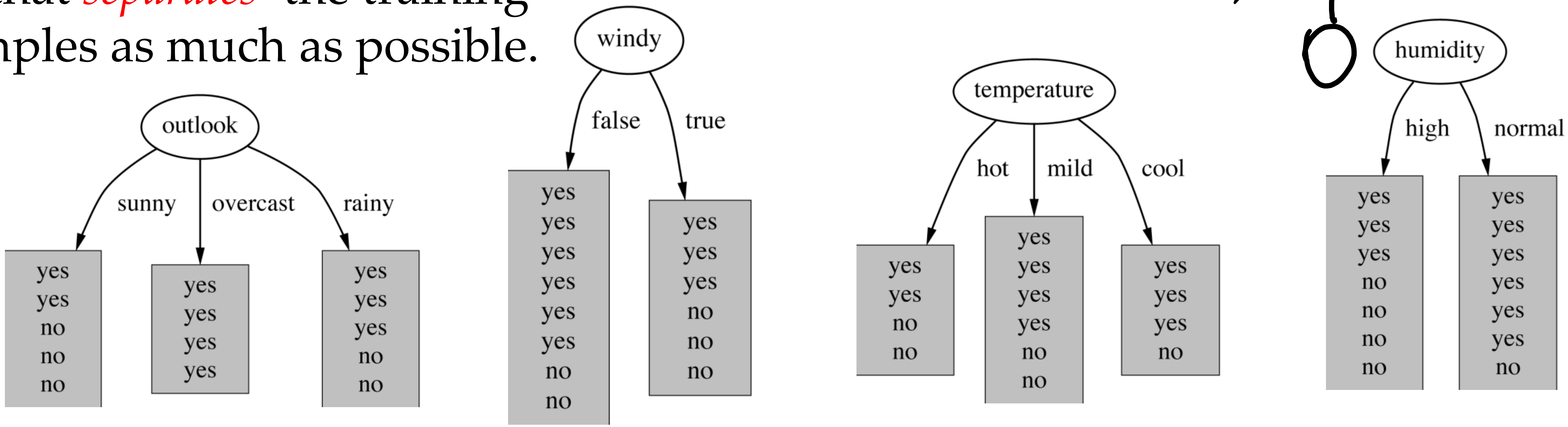
How do we choose the test ?

Which attribute should be used as the test?

discrete



Intuitively, you would prefer the one that *separates* the training examples as much as possible.



Decision tree: divide and conquer

Supervised learning: classification and regression

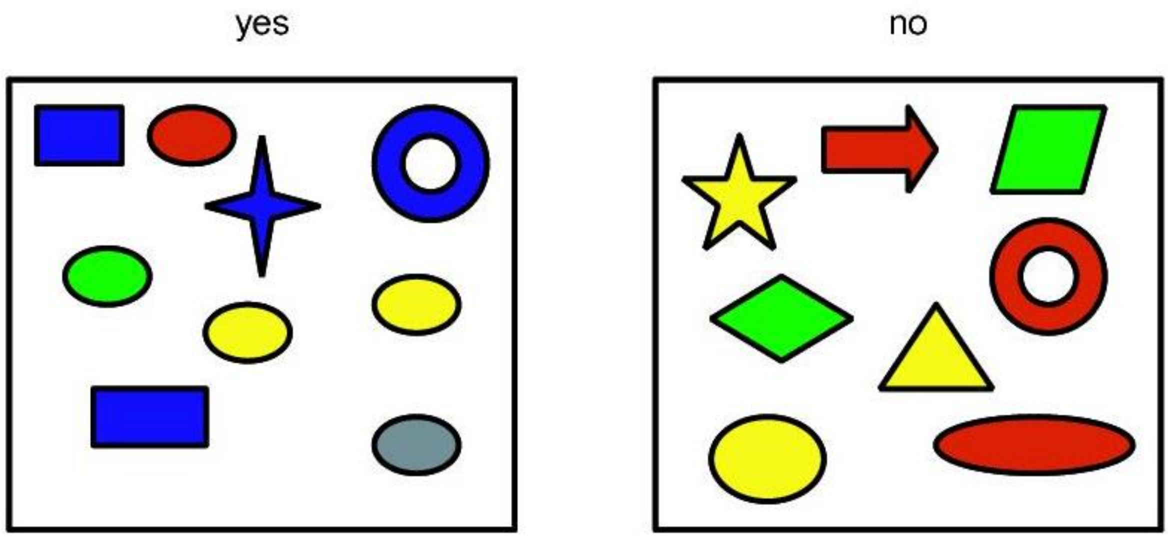
Internal decision nodes implements a test function

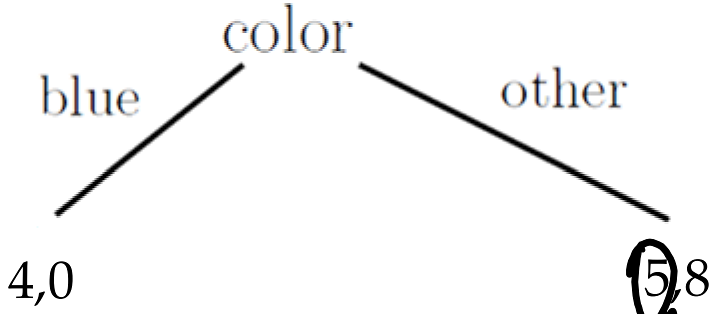
- Univariate: Uses a single attribute, x_i – **This is used most frequently**
  - Numeric x_i : Binary split : $x_i > wm$
  - Discrete x_i : n-way split for n possible values or binary
- Multivariate: Uses multiple/all attributes, x

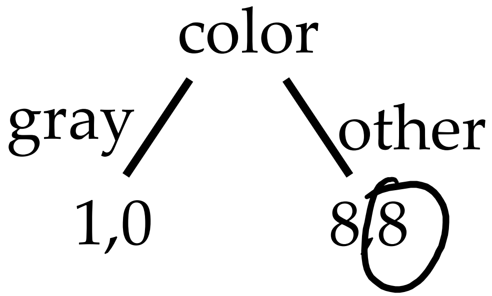
Leaves

- Classification: Class labels, or proportions
- Regression: Numeric; r average, or local fit

Highly interpretable







Error rate: 5/17

Error rate: 8/17

Use majority label at the leaf, then compute error rate

Accuracy score pitfall

Feature 1

400,200 400,200

Error rate: (200+200)/1200

Feature 2

split 250,240 550,160 ← leaf

25%

Error rate: (240+160)/1200

Both have the same error rate!!!

Which is "progressing more" towards a lower error?

Best split in classification

For node m, N_m instances reach m,

N^i_m belong to C_i, then $\hat{P}(C_i \mid \mathbf{x}, m) \equiv p^i_m = \frac{N^i_m}{N_m}$

Node m is pure if p^i_m is 0 or 1

If node m is pure, generate a leaf and stop, otherwise split and continue recursively

Measure of impurity is entropy

worst entropy for binary classification @ 0.5

$$\mathcal{I}_m = -\sum_{i=1}^{K} p^i_m \log_2 p^i_m$$

Entropy as an impurity measure

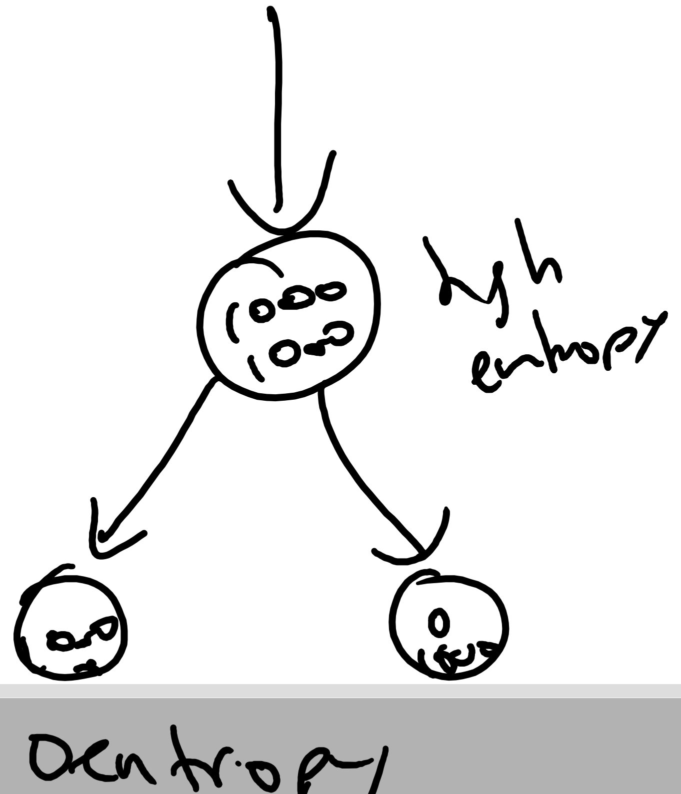
Measure of (degree of) uncertainty

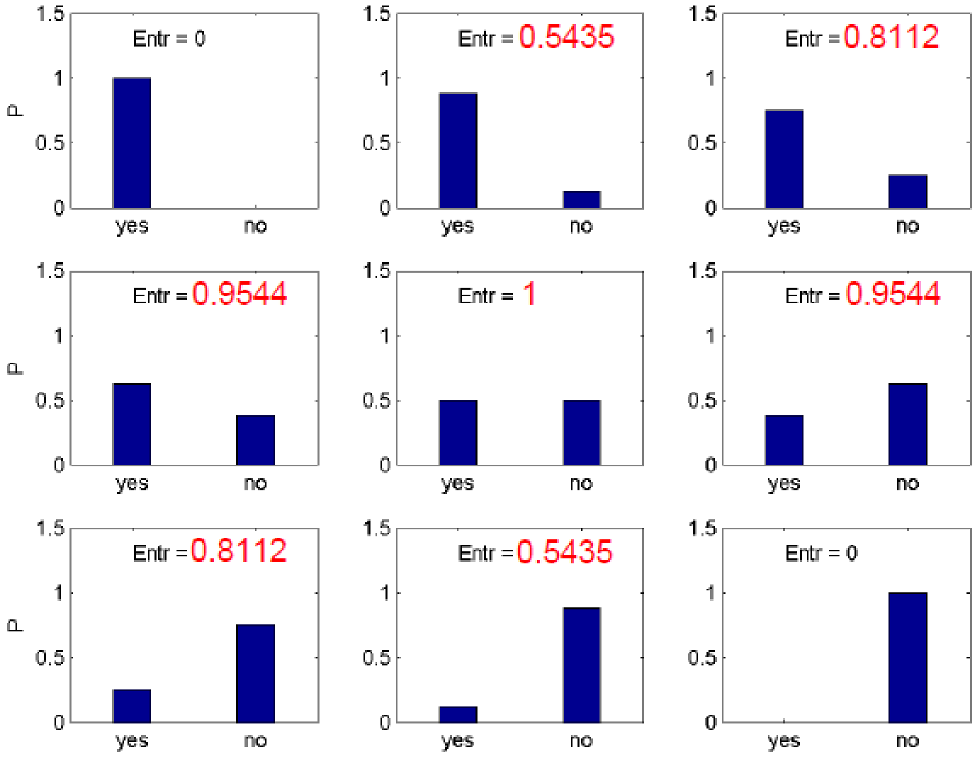
- The more clueless I am about the answer initially, the more information is contained in the answer
- Information in an answer when prior is (P1, …, Pn)

$$\Sigma_{i=1}^{n} - P_i \log_2 P_i$$

- Scale: 1 bit = answer to Boolean question with equal prior
- Roll of a 4-sided die has 2 bits of information.
- Acquisition of information leads to reduction in entropy

how much does entropy improve



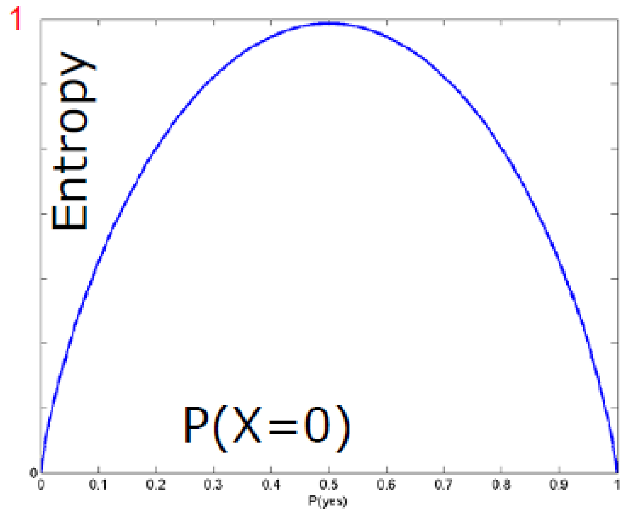


The entropy is maximal when all possibilities are equally likely. for binary $P(X)=0.5$

The goal of the decision tree is to decrease the entropy in each node.

Entropy is zero in a pure "yes" node (or pure "no" node).

Entropy is a measure of "uncertainty" of a random variable.



Caveats

The number of possible values influences the information gain.

Splitting which leaf gives us the largest decrease in entropy

The more possible values, the higher the gain (the more likely it is to form small, but pure partitions)

Best split in classification

If node m is pure, generate a leaf and stop, otherwise split and continue recursively

Impurity after split: N_{mj} of N_m take branch j. N^i_{mj} belong to C_i

$$\mathcal{I}'_m = -\sum_{j=1}^{n} \frac{N_{mj}}{N_m} \sum_{i=1}^{K} p^i_{mj} \log_2 p^i_{mj}$$

Find the variable and split that best reduces impurity (among all variables -- and split positions for numeric variables)

$$\hat{P}(C_i \mid \mathbf{x}, m, j) \equiv p^i_{mj} = \frac{N^i_{mj}}{N_{mj}}$$

800,400

Feature 1

400,200 400,200

Error rate: 400/1200
Entropy: 0.92

Feature 2

250,240 550,160

Error rate: 400/1200
Entropy: 0.86

Most of the [illegible]

not sufficient

Greedy

GenerateTree($\mathcal{X}$)
  If NodeEntropy($\mathcal{X}$)$< \theta_I$ /* eq. 9.3
    Create leaf labelled by majority class in $\mathcal{X}$
    Return
  $i \leftarrow$ SplitAttribute($\mathcal{X}$)
  For each branch of $\boldsymbol{x}_i$
    Find $\mathcal{X}_i$ falling in branch
    GenerateTree($\mathcal{X}_i$)

SplitAttribute($\mathcal{X}$)
  MinEnt$\leftarrow$ MAX
  For all attributes $i = 1, \ldots, d$
    If $\boldsymbol{x}_i$ is discrete with n values
      Split $\mathcal{X}$ into $\mathcal{X}_1, \ldots, \mathcal{X}_n$ by $\boldsymbol{x}_i$
      e $\leftarrow$ SplitEntropy($\mathcal{X}_1, \ldots, \mathcal{X}_n$) /* eq. 9.8 */
      If e<MinEnt MinEnt $\leftarrow$ e; bestf $\leftarrow$ i
    Else /* $\boldsymbol{x}_i$ is numeric */
      For all possible splits
        Split $\mathcal{X}$ into $\mathcal{X}_1, \mathcal{X}_2$ on $\boldsymbol{x}_i$
        e$\leftarrow$SplitEntropy($\mathcal{X}_1, \mathcal{X}_2$)
        If e<MinEnt MinEnt $\leftarrow$ e; bestf $\leftarrow$ i
  Return bestf

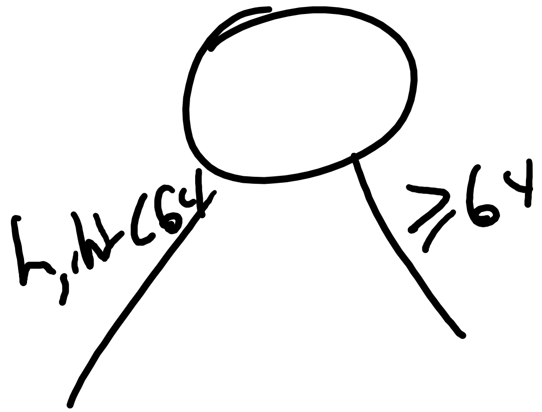
all possible splits

entropy cutoff heuristic. (prevent overfitting

pre prun

get the entropy

split point

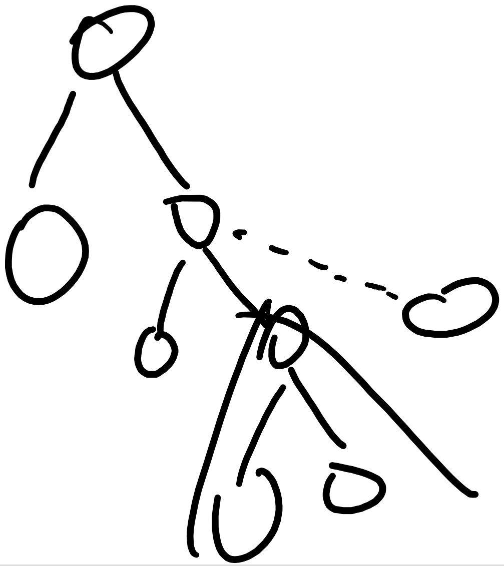


larger tree → more likely to overfit

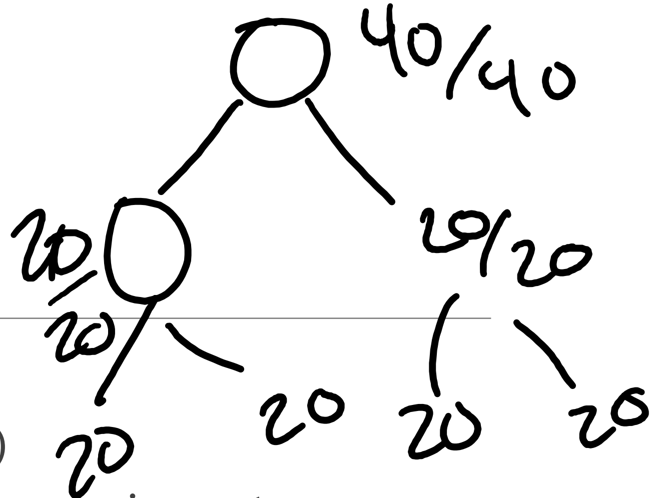
Pruning Trees

Remove subtrees for better generalization (decrease variance)

- Prepruning: Early stopping (e.g. < 5% points or small change in entropy)
- Postpruning: Grow the whole tree then prune subtrees that overfit on the pruning set
- Set aside a subset of data for pruning /validation set



Pruning Trees



Remove subtrees for better generalization (decrease variance)

- Prepruning: Early stopping (e.g. < 5% points or small change in entropy)
- Postpruning: Grow the whole tree then prune subtrees that overfit on the pruning set
- Set aside a subset of data for pruning

greedy search

Prepruning is faster, postpruning is more accurate (requires a separate pruning set)

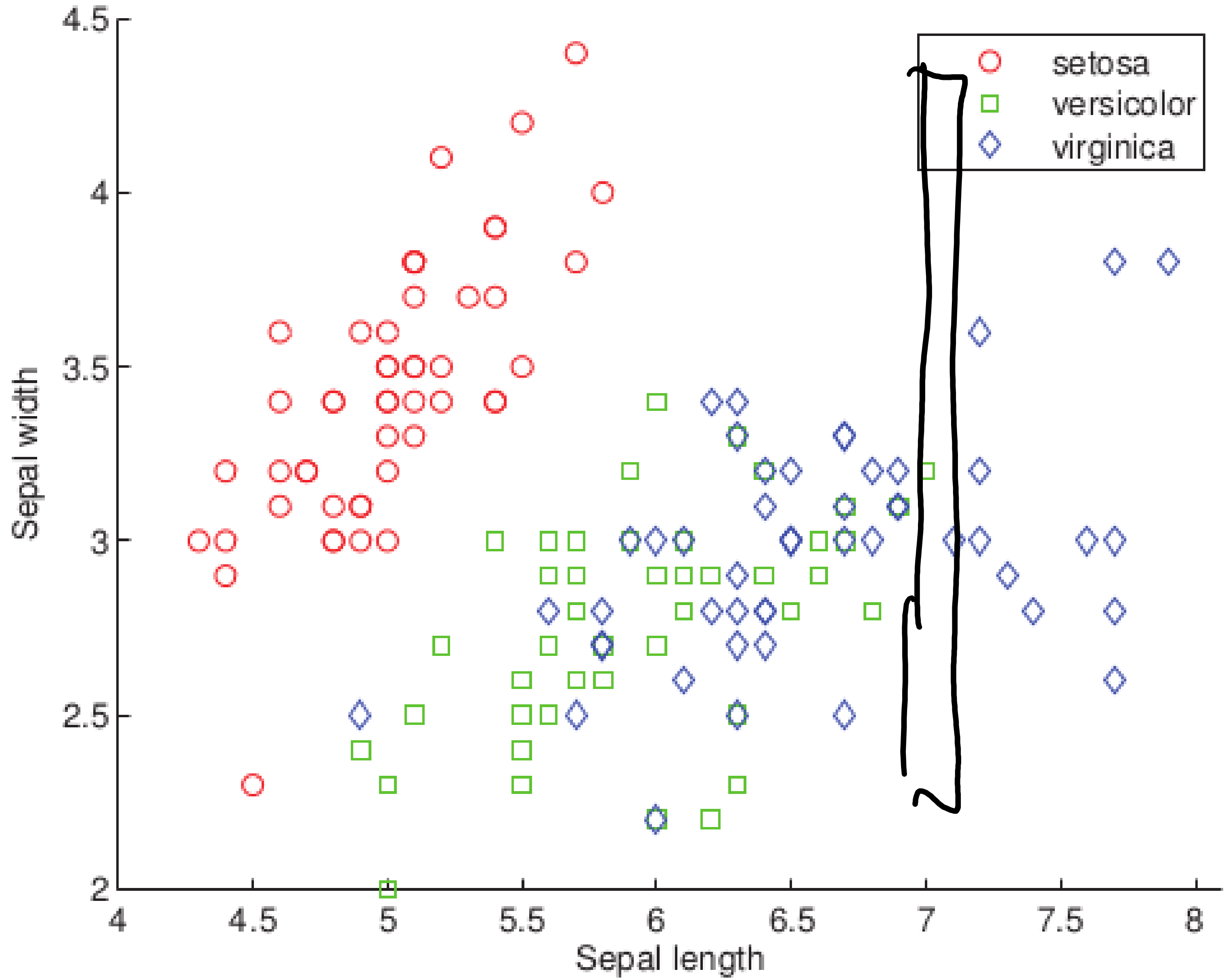
add'l negative consequences of stopping too early

Post-pruning – create several candidate trees of less depth and use the one with lowest validation error (This is separate from 10-fold cross validation)

pruning error

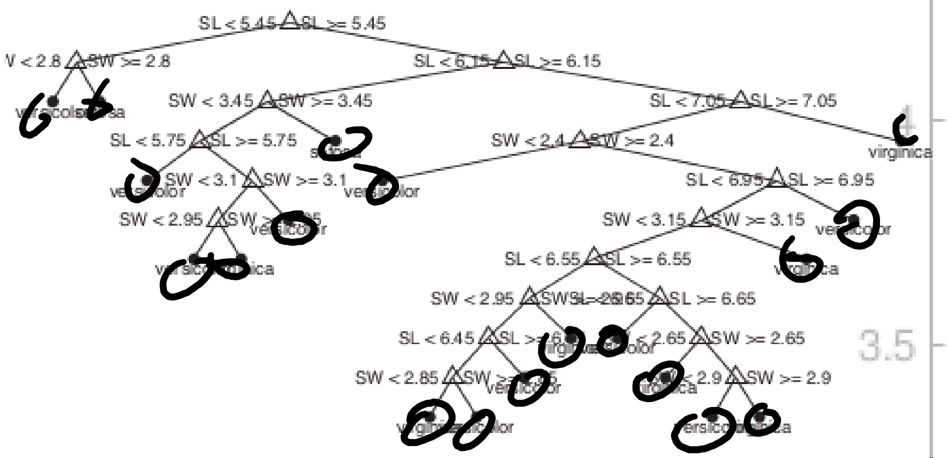
computationally intensive

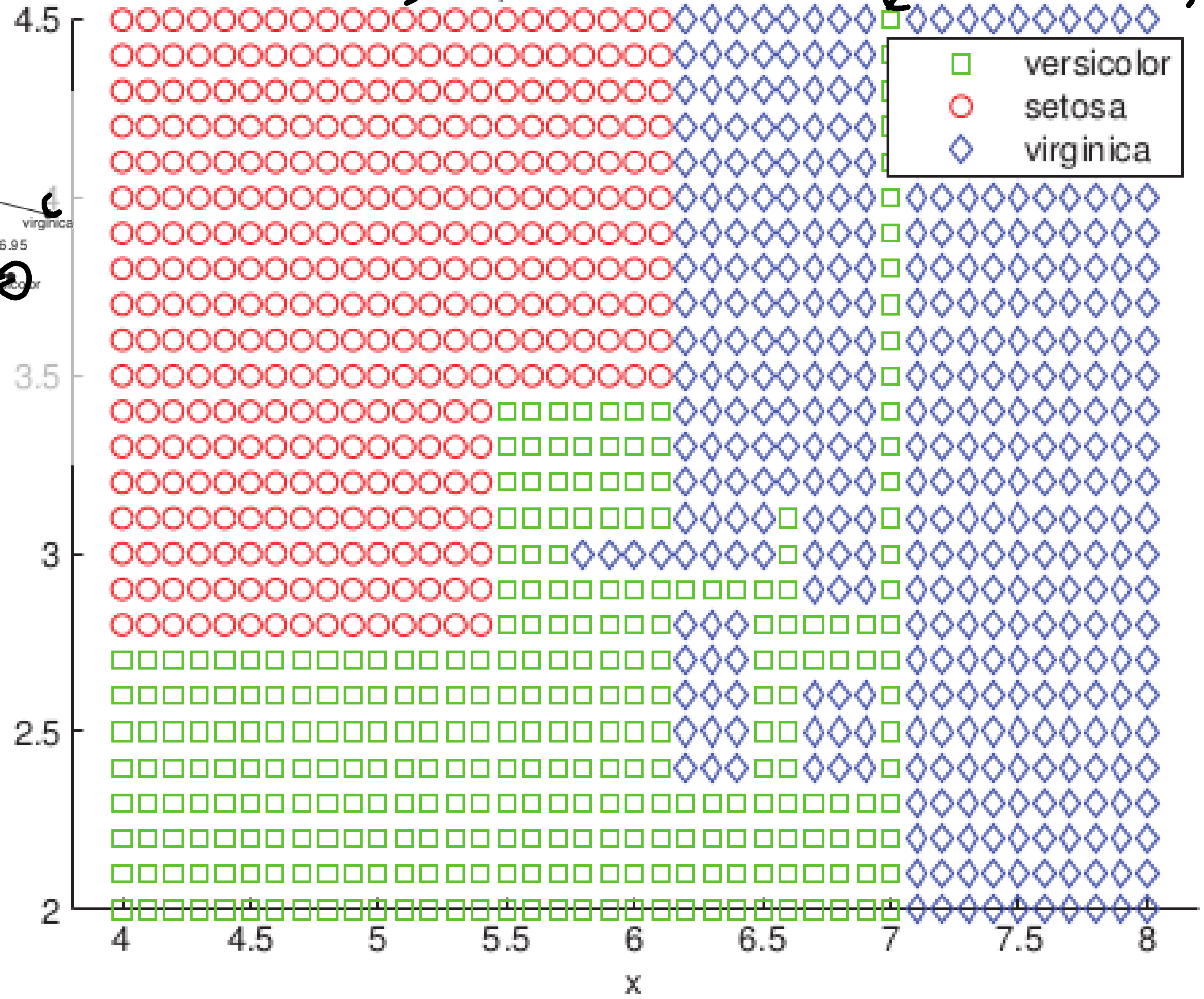
combine nodes


setosa
versicolor
virginica
4.5
4
3.5
3
2.5
2
Sepal width
4
4.5
5
5.5
6
6.5
7
7.5
8
Sepal length

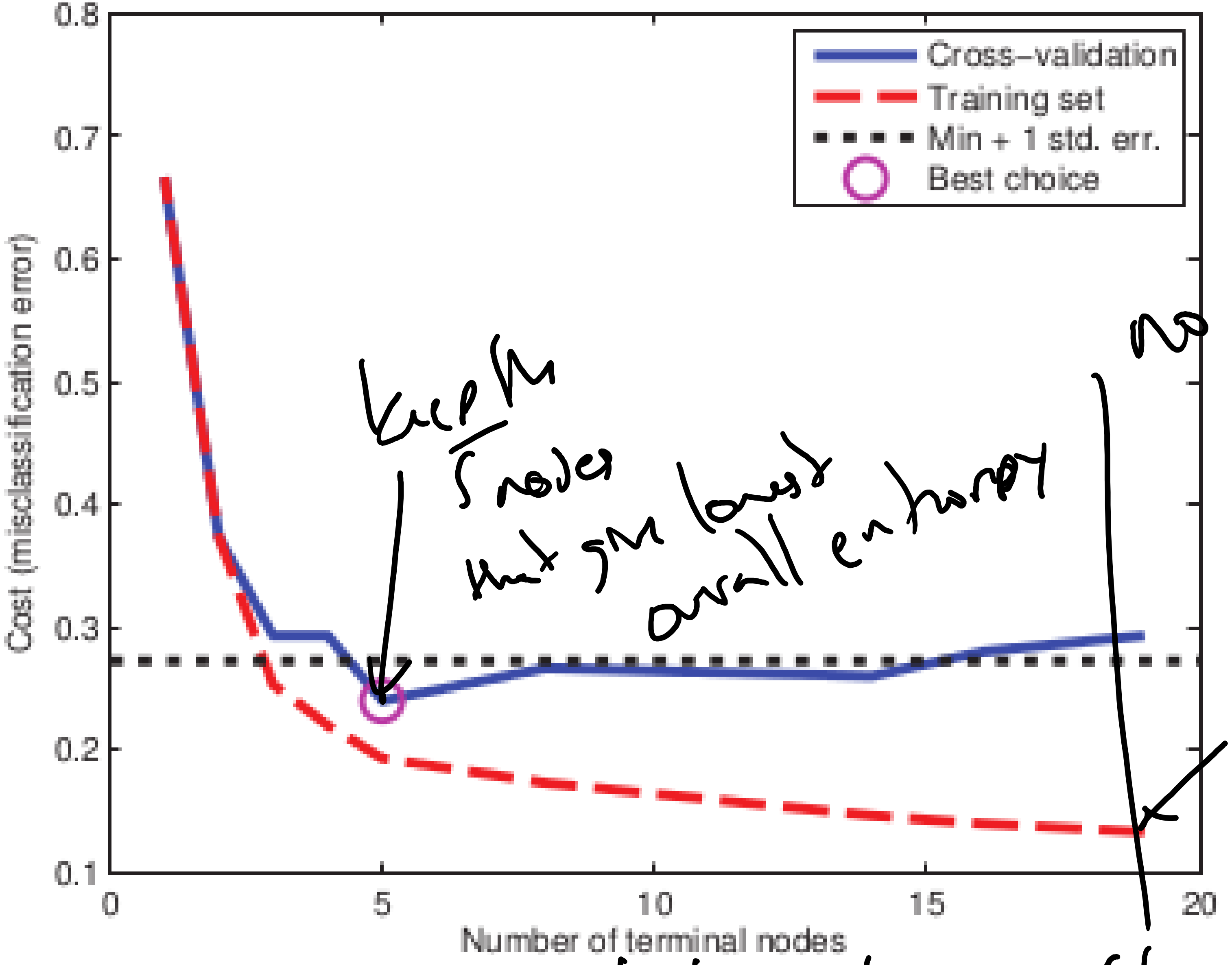
unpruned decision tree

overfitting





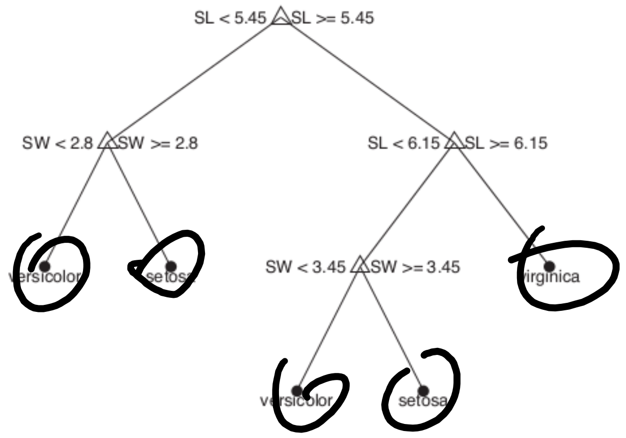
15



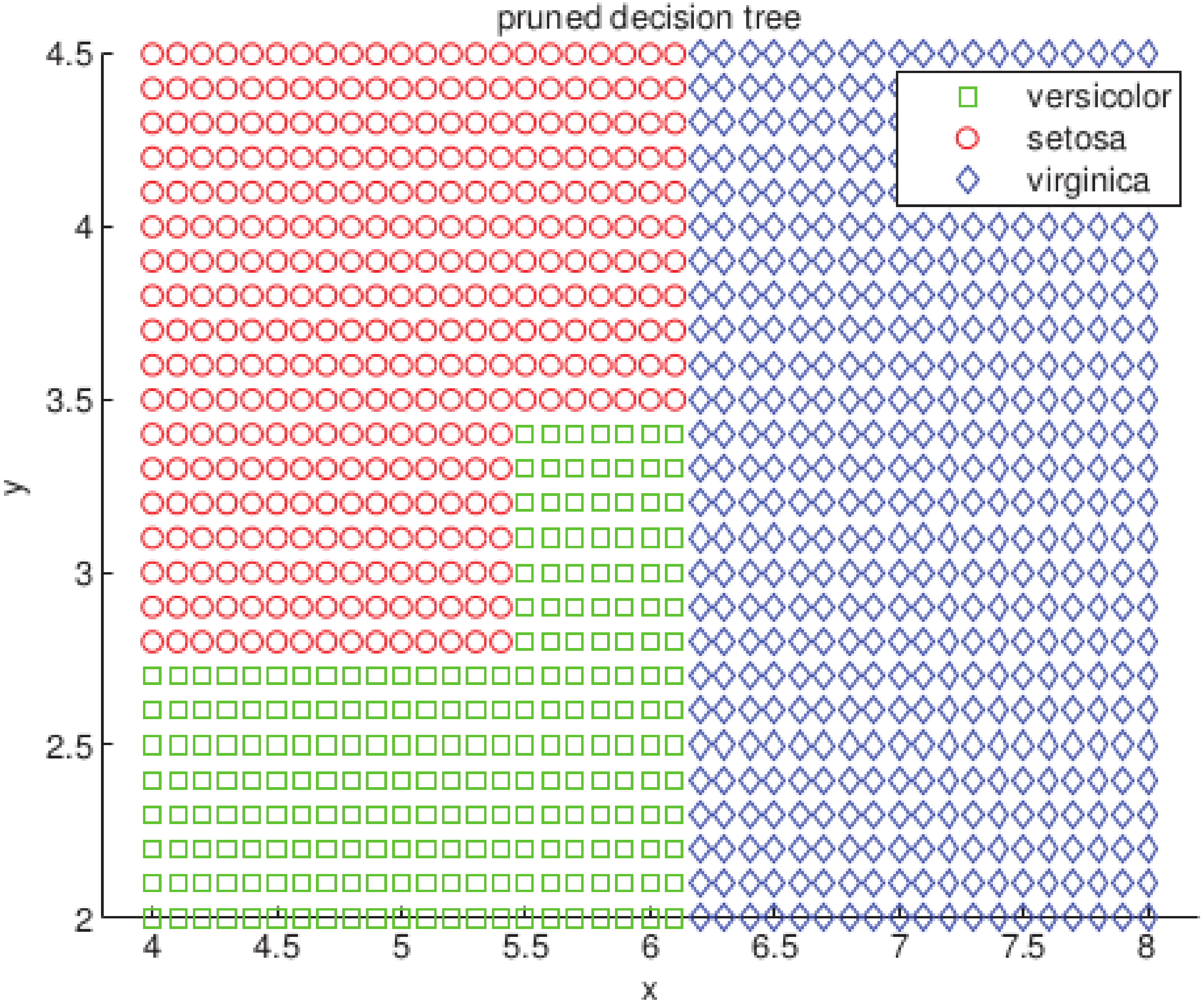
keep the 5 nodes that give lowest overall entropy

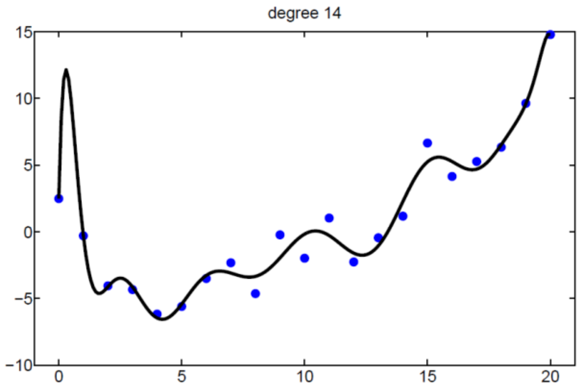
no pruning at all

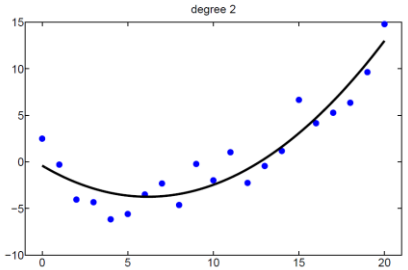
that we keep after post-pruning

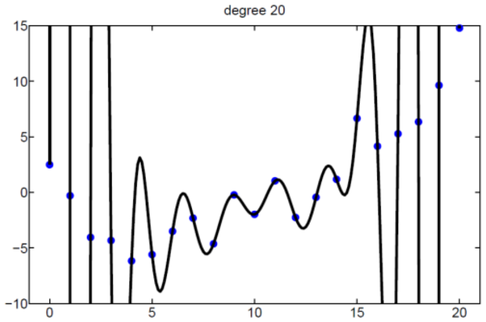


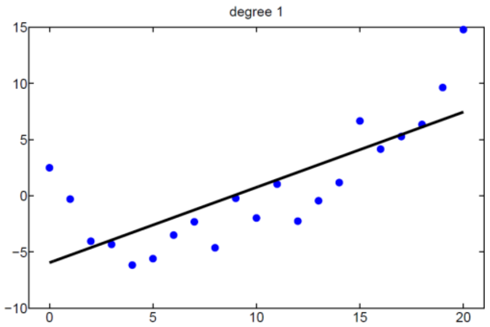
Post-prun is robust











Overfitting in Regression

Decision trees are particularly vulnerable to overfitting

Be prepared to recognize the signs

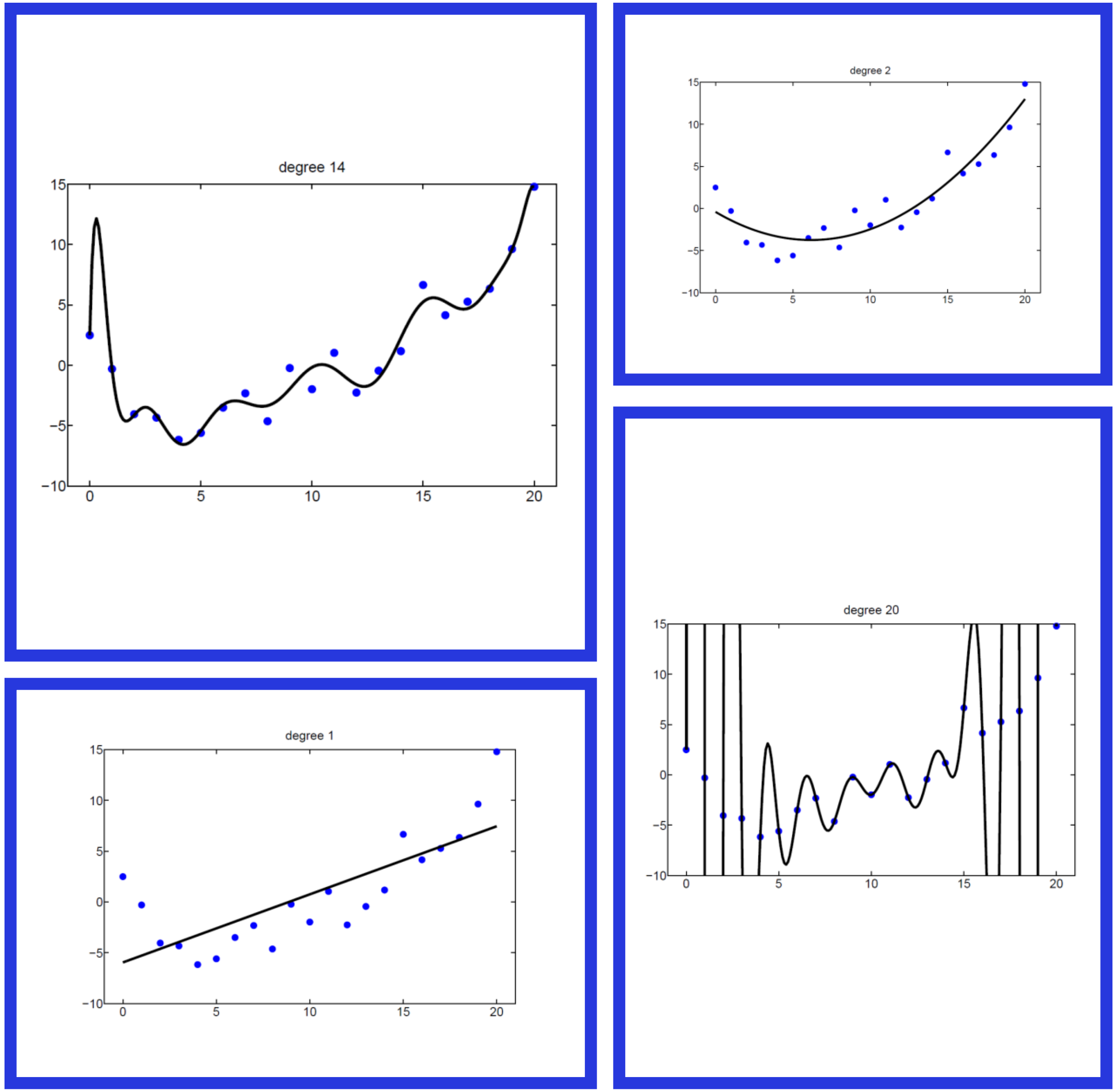
How can you prevent?

- Prune the tree

Is there any benefit to this?

Why use decision trees?

Simplicity



Overfitting in Regression

Decision trees are particularly vulnerable to overfitting

Be prepared to recognize the signs

How can you prevent?

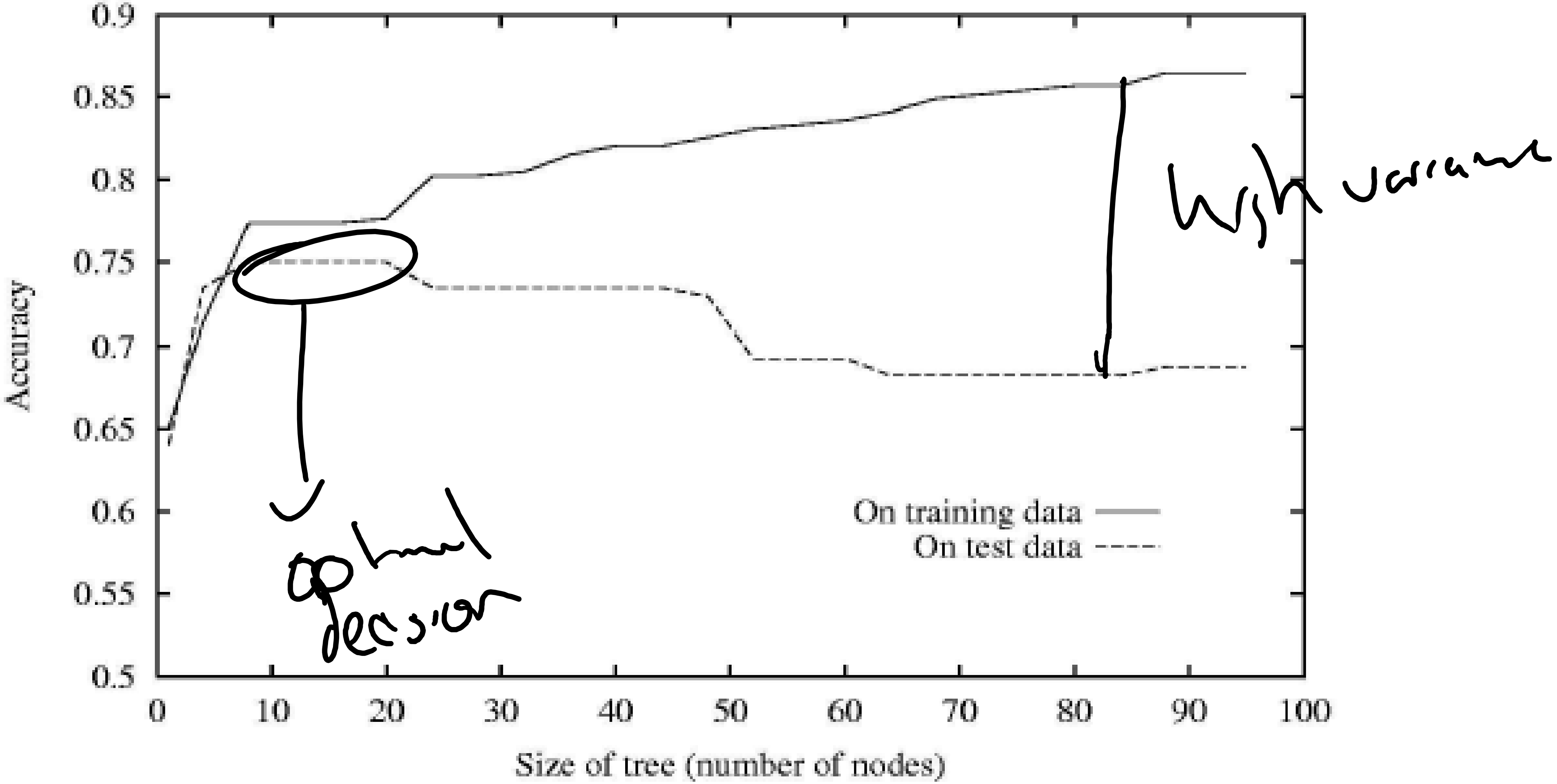
- Prune the tree

Is there any benefit to this?

- Decision trees are very unstable, and therefore make very good weak classifiers for ensemble methods

Commonly most ensemble methods use DTrees

Decision Tree Overfitting



Decision Tree Conclusion

Decision trees are very interpretable

(Relatively) easy to implement

Finding optimal decision tree is impractical — exponential

Less accurate than many other methods

high accuracy → overfitting

Often unstable

- Small changes in training data lead to very different decision boundaries
- Which means? Good for bagging!

Bagging : an simulated example

Generated a sample of size N = 30, with two classes and p = 5 features, each having a standard Gaussian distribution with pairwise Correlation 0.95.

X_1 is the relevant feature

The response Y was generated according to

$\Pr(Y = 1 \mid x1 \leq 0.5) = 0.2$,

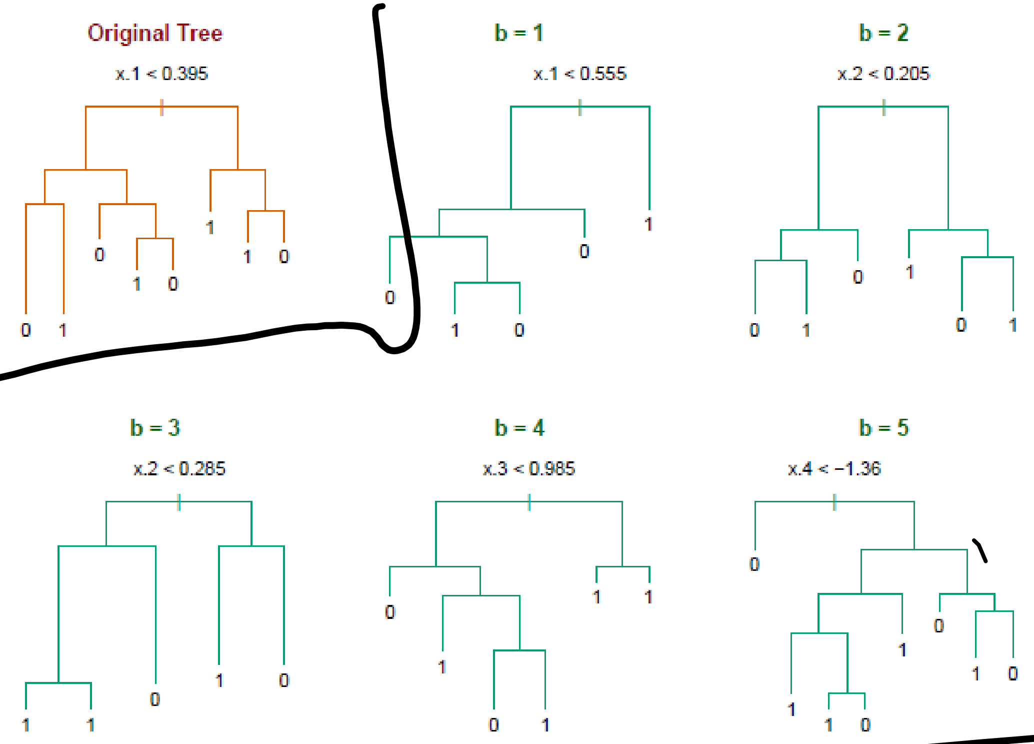
$\Pr(Y = 0 \mid x1 > 0.5) = 0.8$.

$X_2\, X_3\, X_4\, X_5$ do explain the data but only because they are correlated w/ X_1

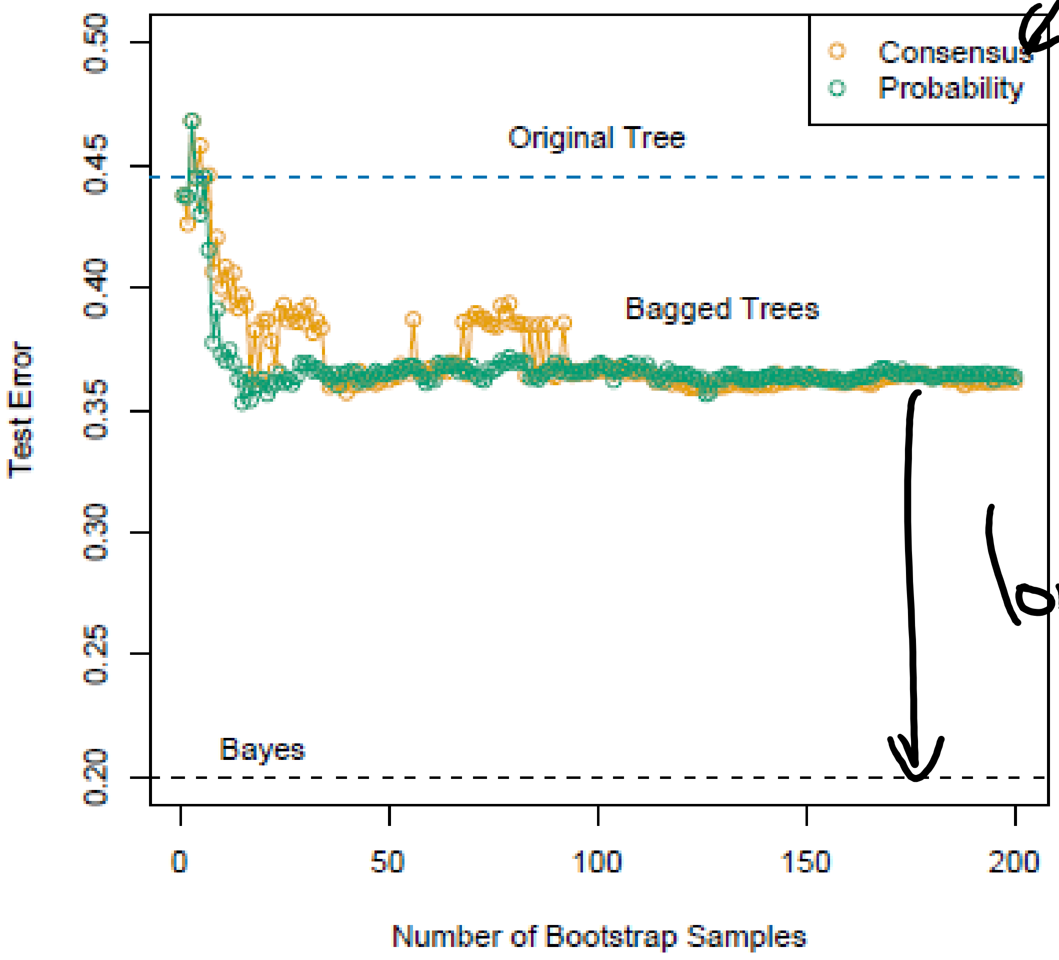
Bagging

Notice the bootstrap trees are different than the original tree

w/ whole dataset



Bootstrap Samples



Bagging

Because of the instability of the classifier, bagging often lowers error for decision trees

long way from "optimal"

Random Forest classifier

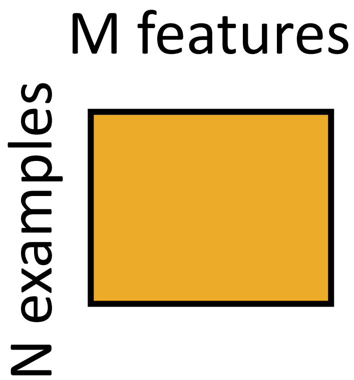
Random forest classifier, an extension to bagging which uses de-correlated trees.

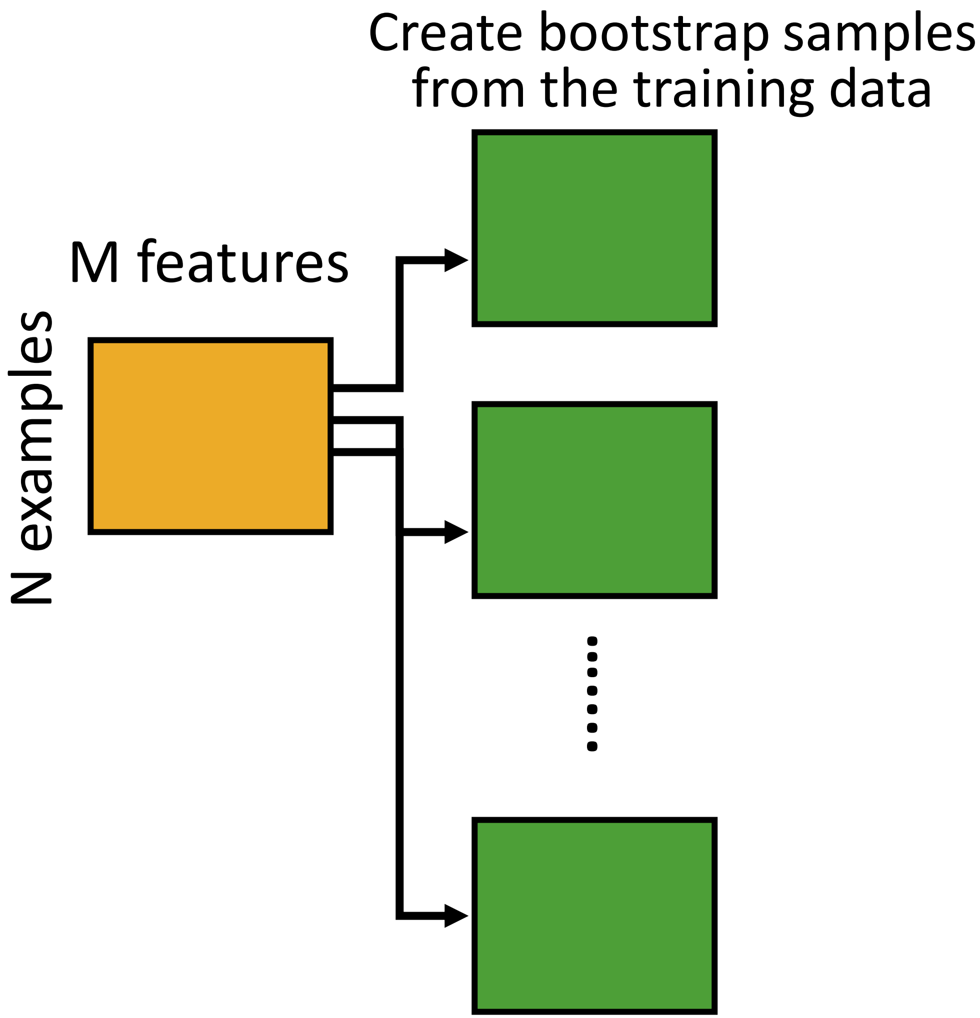
remember: less correlation → more independence

more benefit from ensemble

Random Forest Classifier

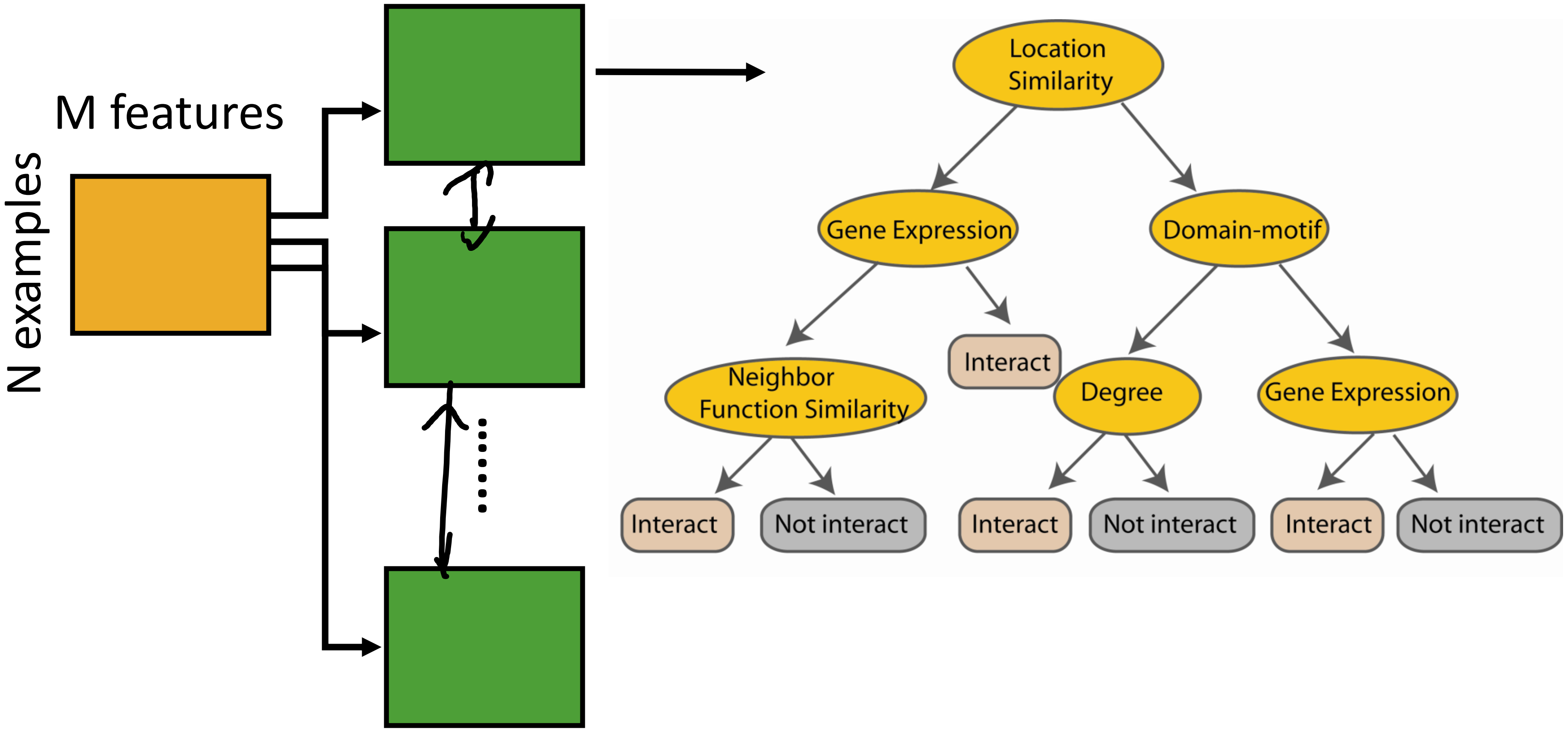
Training Data



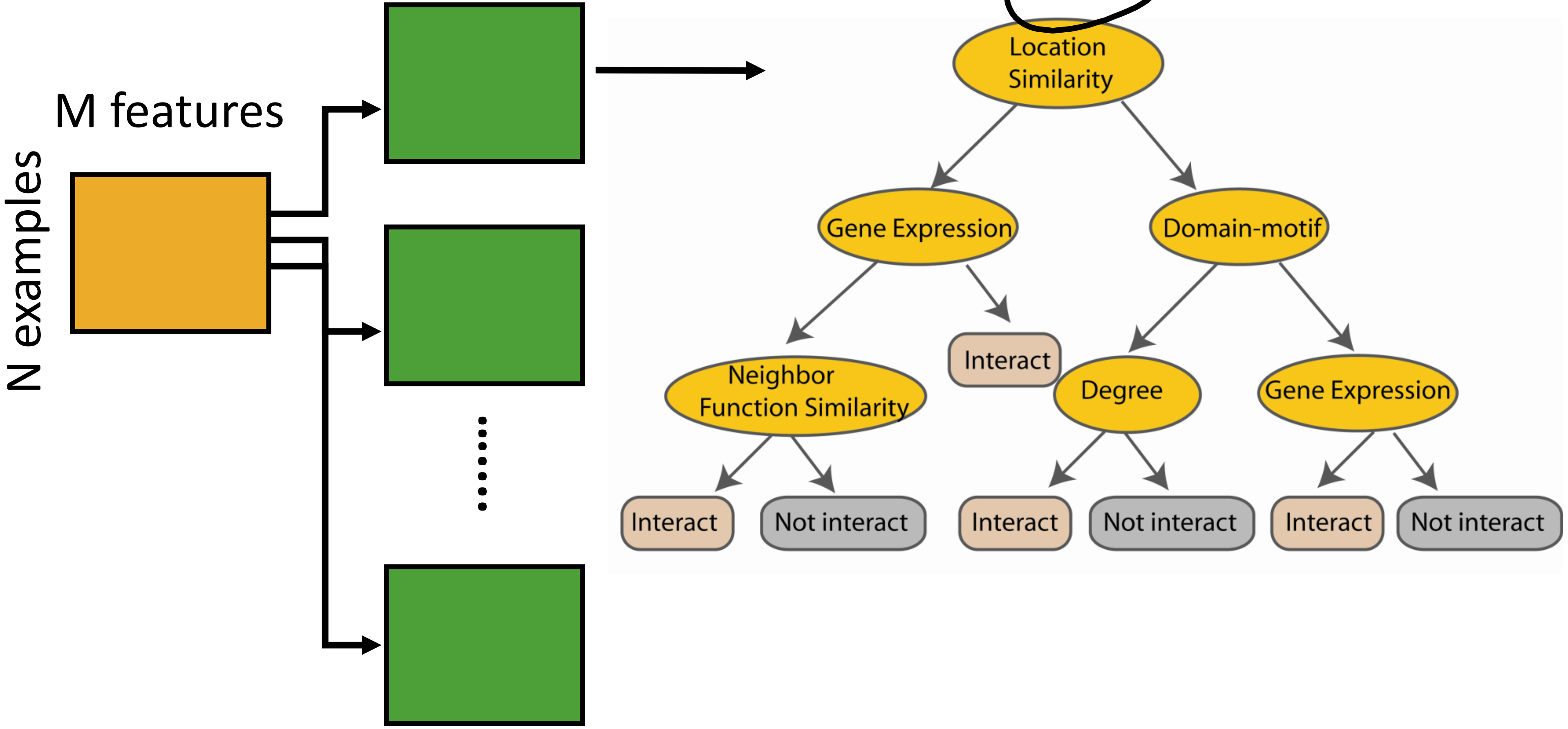

Create bootstrap samples
from the training data
M features
N examples

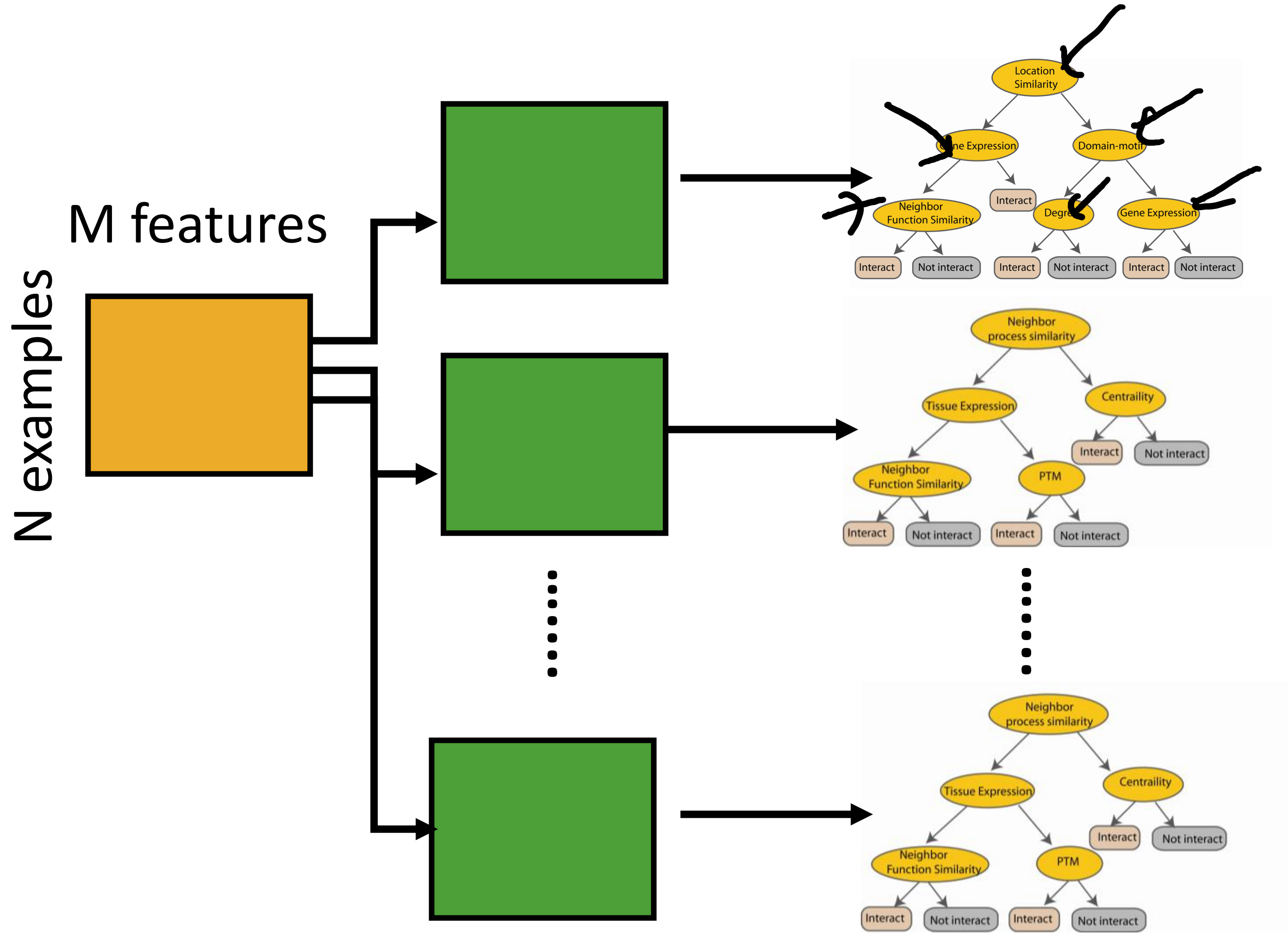
Standard bagging

Construct a decision tree

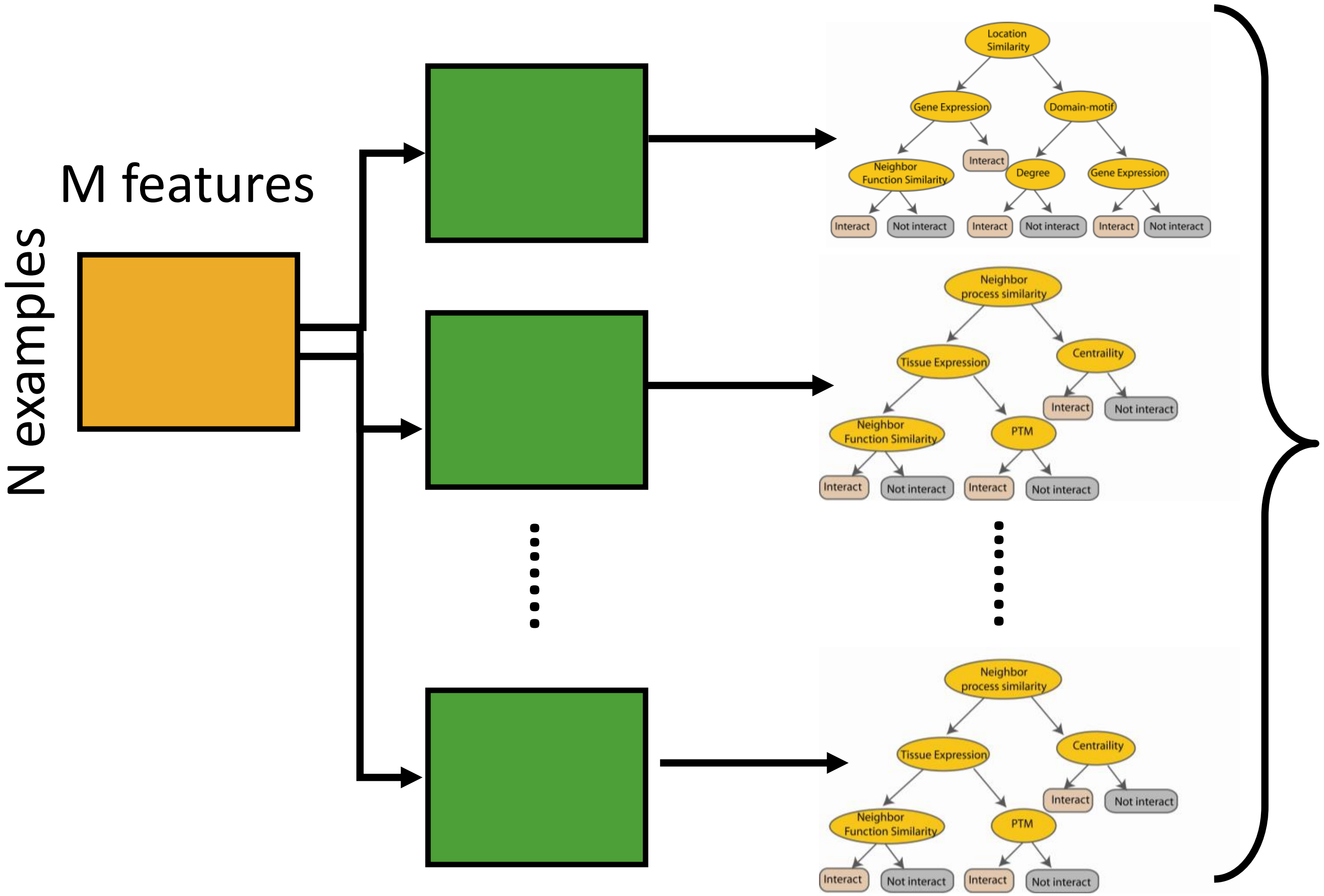


At each node in choosing the split feature choose only among $m<M$ features





each tree in forest gets a different random subset of the features



Take he majority vote

(or consensus)

Random Forests Issues

When the number of variables is large, but the fraction of relevant variables is small, random forests are likely to perform poorly when m is small

m is the # of features

for example X_1 is the only "relevant" feature

Why?

Because:

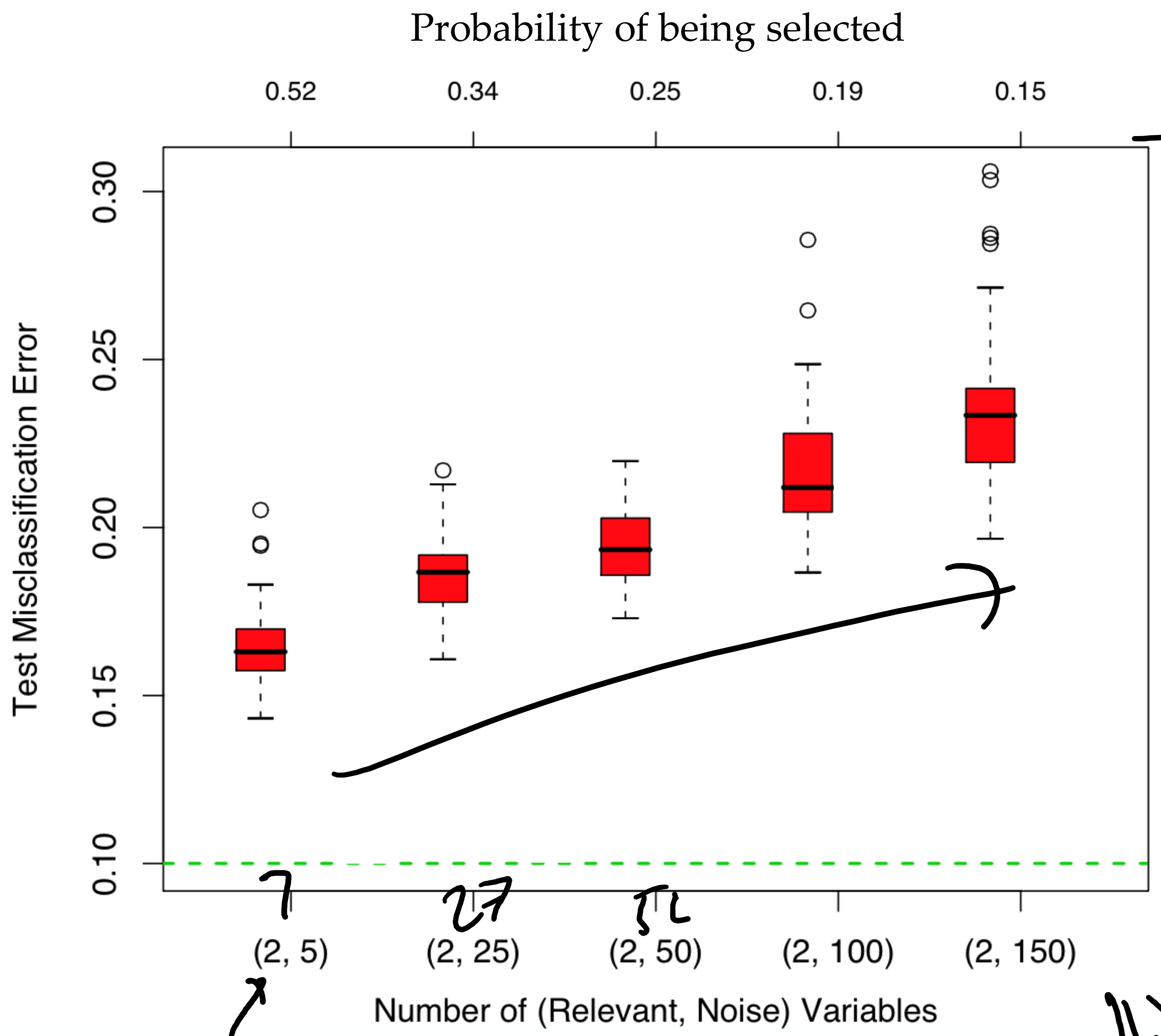
At each ~~split~~ the chance can be small that the relevant variables will be selected

ensemble tree

For example, with 3 relevant and 100 not so relevant variables the probability of any of the relevant variables being selected at any split is ~0.25

for m =

$$1-(.97)^m = 0.25$$



Subset selection

more noise variable

Can RF overfit?

Random forests "cannot overfit" the data wrt to number of trees.

More trees cannot overfit the model

Why?

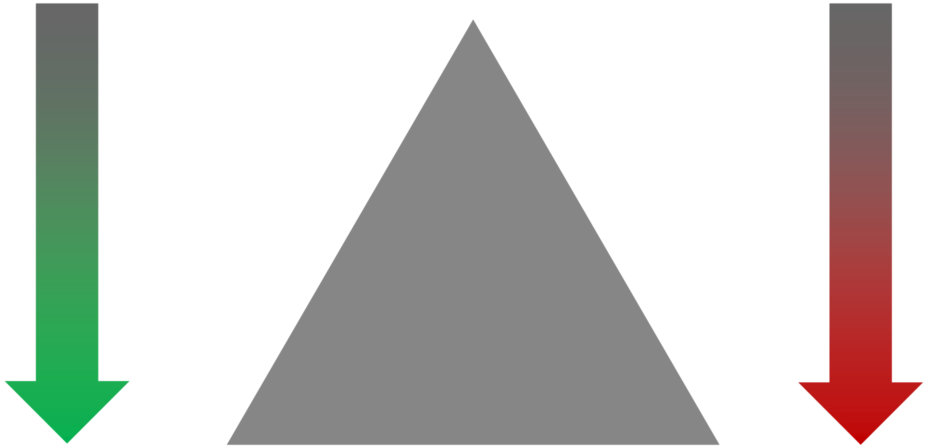
The number of trees, B does not mean increase in the flexibility of the model

above a certain threshold

Decision Tree Models

- As tree depth increases, bias decreases and variance generally increases. Why? (Hint: think about k-NN)

does increase complexity



Bias decreases with tree depth

Variance increases with tree depth

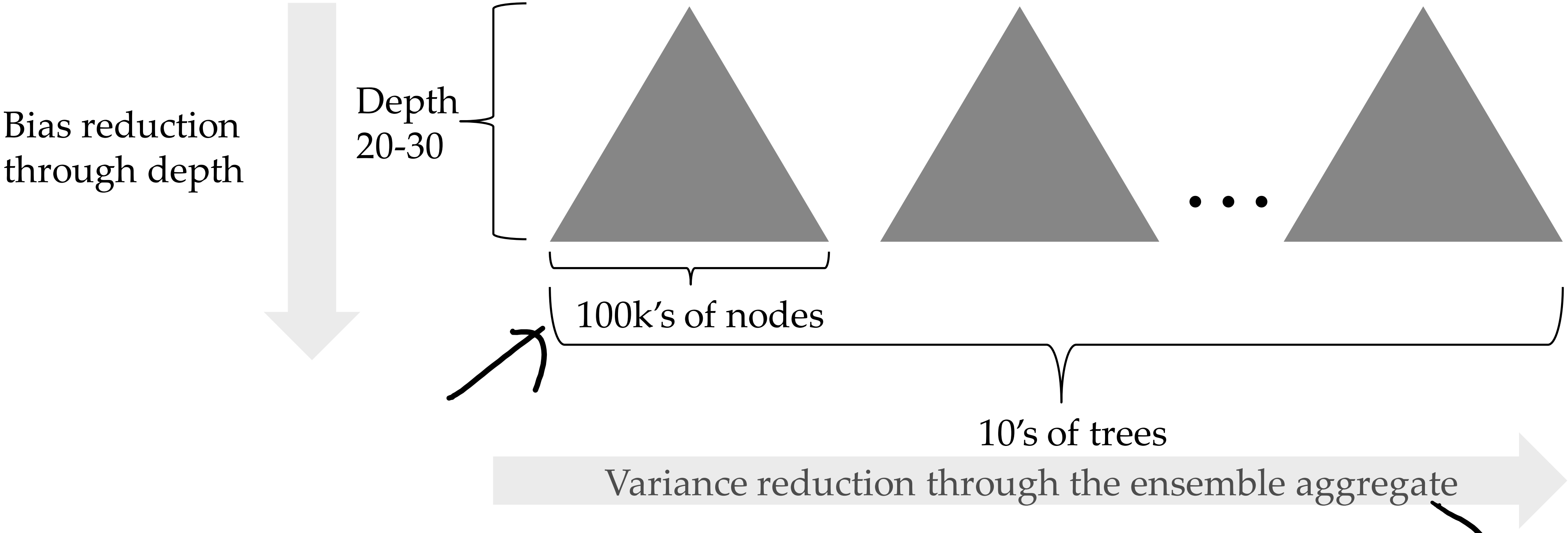
high instability

Random Forests vs Boosted Trees

- The "geometry" of the methods is very different (MNIST data):

smaller numbers of deeper trees

- Random forest use 10's of deep, large trees:

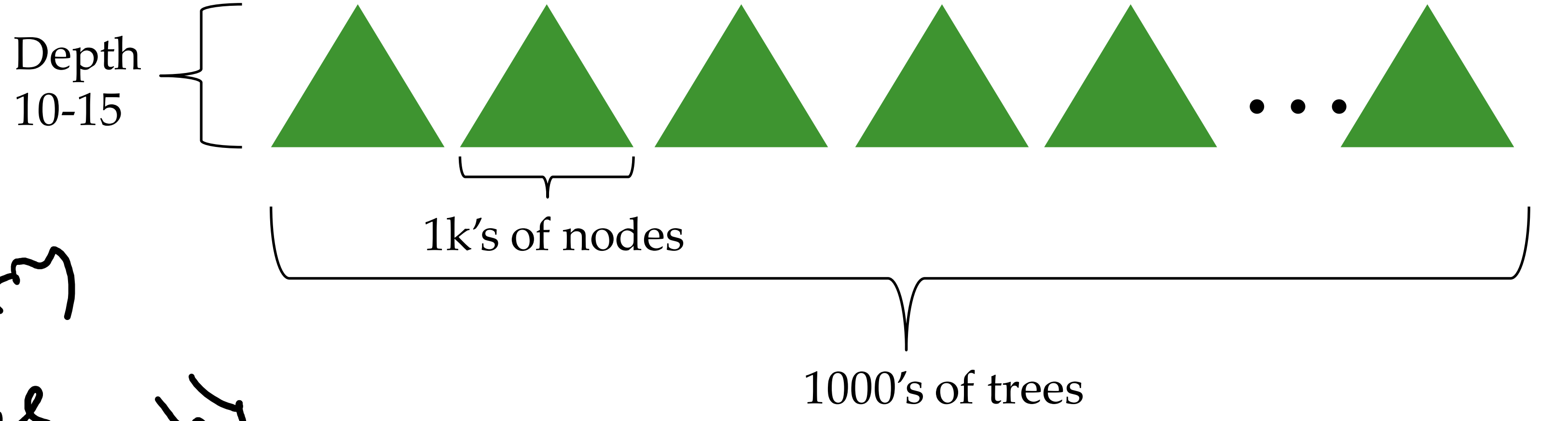


Random Forests vs Boosted Trees

- The "geometry" of the methods is very different (MNIST data):
- Boosted decision trees use 1000's of shallow, small trees:

variance reduction

limiting size of ensemble trees



Bias reduction through boosting – variance already low

however, w/ boosting, can overfit w/ more trees

Back to statistics

Suppose ~~the average~~ a student carries on average \$20 in cash

- What is the probability a particular student carries \$100 in cash?

Back to statistics

Suppose the average student carries $20 in cash

- What is the probability a particular student carries $100 in cash?

What is the mathematical notation for this problem?

random probability

X – # of dollars in a students pocket

$E(X) = \$20$

Back to statistics

What is the highest percentage of students w/ 100.

Suppose the average student carries $20 in cash

- What is the probability a particular student carries $100 in cash?

at least

What is the mathematical notation for this problem?

- Define a random variable: let X be the number of dollars in a student's pocket
- So, what is $20? $E(x)$

Ideally, we could analyze money in pockets

$$1 \geq P(X=20) \geq 0$$

80 students
$0

$$P(X>100) \leq 20\%$$

20% students
$100

Back to statistics

Suppose the average student carries \$20 in cash

- What is the probability a particular student carries \$100 in cash?

What is the mathematical notation for this problem?

- Define a random variable: let X be the number of dollars in a student's pocket
- So, what is \$20? E[X]
- What are we trying to find? P(X>100)
- *Note: X must be non-negative (pretend debt isn't real)*

$$\mathrm{E}(X) = \int_{-\infty}^{\infty} x\,f(x)\,dx$$ *where f(x) is the pdf*

20% chance to have \$100

Back to statistics

can't have debt in your pocket

$X \geq 0$

Suppose the average student carries \$20 in cash

- What is the probability a particular student carries \$100 in cash?

What is the mathematical notation for this problem?

- Define a random variable: let X be the number of dollars in a student's pocket
- So, what is \$20? E[X]
- What are we trying to find? P(X>100)
- *Note: X must be non-negative (pretend debt isn't real)*

$$\mathrm{E}(X) = \int_{-\infty}^{\infty} x f(x)\, dx$$ *where f(x) is the pdf*

$$\mathrm{E}(X) = \int_0^a x f(x)\, dx + \int_a^{\infty} x f(x)\, dx \geq \int_a^{\infty} x f(x)\, dx \geq \int_a^{\infty} a f(x)\, dx = a \int_a^{\infty} f(x)\, dx = a \Pr(X \geq a)$$

say all students have \$100

- **Markov's Inequality:** $\Pr(X \geq a) \leq \mathrm{E}(X)/a$

100

$\frac{20}{100}$

$P(X \geq 100) \leq 20\%$

Markov's Inequality

Markov's Inequality: $\Pr(X \geq a) \leq \mathrm{E}(X)/a$

This is a concentration bound, it shows us a bound on how the data is going to be concentrated

Suppose the average student carries $20 in cash

- What is the probability a particular student carries $100 in cash?

Markov's Inequality

Markov's Inequality: $\Pr(X \geq a) \leq \mathrm{E}(X)/a$

This is a concentration bound, it shows us a bound on how the data is going to be concentrated

Suppose the average student carries $20 in cash

- What is the probability a particular student carries $100 in cash?
- 20/100 = 0.2

Markov's Inequality

$P(E_{rror} > \epsilon) \leq \delta$

Markov's Inequality: $\Pr(X \geq a) \leq \mathrm{E}(X)/a$

This is a concentration bound, it shows us a bound on how the data is going to be concentrated

Suppose the average student carries \$20 in cash

- What is the probability a particular student carries \$100 in cash?
- 20/100 = 0.2

Why is this going to be useful? What non-negative random variable do we care about?

- E_{out}! What is the actual error our model is going to have?:
- E_{cv} and E_{test} are just estimators. A lot of this math is also directly applicable to polling
- Are E_{cv} and E_{test} unbiased?

it only just
it one

$P(E_{rror} > 0.05) \leq 0.05$

Motivation

Suppose there was a measure on the ballot, and I'm trying to determine what proportion of the population (A) supports it.

- More critically, what am I interested in?

Motivation

Suppose there was a measure on the ballot, and I'm trying to determine what proportion of the population (A) supports it.

- More critically, what am I interested in? P(A>0.5)

Motivation

Suppose there was a measure on the ballot, and I'm trying to determine what proportion of the population (A) supports it.

- More critically, what am I interested in? P(A>0.5)
- How would I estimate this value? A poll? What are some problems?
  - How do I ask the question?
  - Who do I ask?
  - When do I ask?
  - How many people do I ask?
- Which of these are applicable to our problem of E_{out}?
  - Is my data representative of what I'm going to predict?
  - What if I had 15 polls? What should my reported value for A be?
  - Maybe take an average, but in ML, we want the **best** model, it makes sense that we should pay some penalty
  - This also explains why we only want to use the test data once – keeps the estimate unbiased

Survey design

more people → higher confidence

↳ also explains why testing set is large

Motivation

Markov's Inequality: $\Pr(X \geq a) \leq \mathbf{E}(X)/a$

What is the probability our actual error $X=E_{out}$ is double our expected error E_{test}?

Motivation

0.02 E_{test}

0.04 E_{out} wp 0.5

Markov's Inequality: $\Pr(X \geq a) \leq \mathrm{E}(X)/a$

What is the probability our actual error X=E_{out} is double our expected error E_{test}?

- ½ -- that's not great
- What is our 95% confidence interval?

$$P(X \geq 2E(x)) \leq \frac{E(x)}{2E(x)} = 0.5$$

$$\frac{E(x)}{20E(x)} = 0.05$$

0.02 E_{test}

0.40 E_{out} wp 95%

Motivation

Markov's Inequality: $\Pr(X \geq a) \leq \mathrm{E}(X)/a$

What is the probability our actual error $X=E_{out}$ is double our expected error E_{test}?

- ½ -- that's not great
- What is our 95% confidence interval?
- $E_{out} = 20*E_{test}$

Motivation

Markov's Inequality: $\Pr(X \geq a) \leq \mathrm{E}(X)/a$

What is the probability our actual error $X=E_{out}$ is double our expected error E_{test}?

- ½ -- that's not great
- What is our 95% confidence interval?
- $E_{out} = 20^*E_{test}$ – ouch

Is this the best we can do?

- What else do we know about our error? Currently, what are we using?

Motivation

Markov's Inequality: $\Pr(X \geq a) \leq \mathrm{E}(X)/a$

What is the probability our actual error $X=E_{out}$ is double our expected error E_{test}?

- ½ -- that's not great
- What is our 95% confidence interval?
- $E_{out} = 20*E_{test}$ – ouch

Is this the best we can do?

- What else do we know about our error? Currently, what are we using?
- Only that it is non-negative! What else can we use?

Motivation

Markov's Inequality: $\Pr(X \geq a) \leq \mathrm{E}(X)/a$

What is the probability our actual error $X=E_{out}$ is double our expected error E_{test}?

- ½ -- that's not great
- What is our 95% confidence interval?
- $E_{out} = 20*E_{test}$ – ouch

Is this the best we can do?

- What else do we know about our error? Currently, what are we using?
- Only that it is non-negative! What else can we use?
- That we might know it's standard deviation, and that the error is bounded in [0,1]