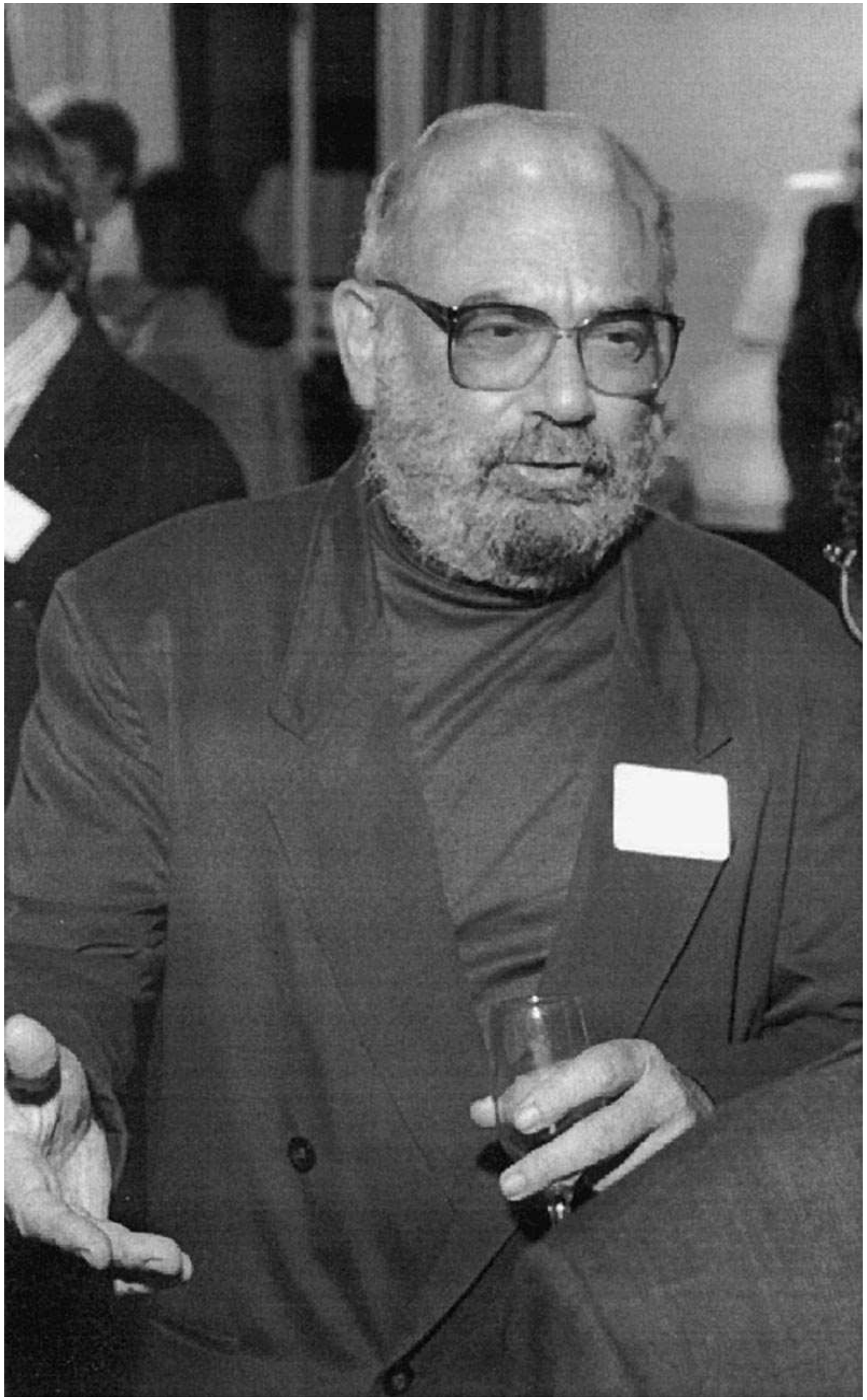
Random Forests – A Statistical Tool for the Sciences

Adele Cutler
Utah State University



Based on joint work with
Leo Breiman, UC Berkleley.

Thanks to Andy Liaw,
Merck.

Neural net research, 1987 – 1990 (Perrone, 1992)

Bayesian BP (Buntine & Weigend 92)

Hierarchical NNs (Ersoy & Hong 90)

Hybrid NNs (Cooper 91, Scofield et al. 87, Reilly 88, 87)

Local experts (Jacobs et al. 1991)

Neural trees (Perrone 92, Sankar 90)

Stacked generalization (Wolpert 90)

Synergy (Lincoln & Skrzypek 90)

$$\left.\begin{array}{l}\text{- many learning algorithms}\\ \text{- many possible architectures}\\ \text{- many local minima}\end{array}\right\}\ \begin{array}{l}\text{many}\\ \text{disagreeing}\\ \text{networks}\end{array}$$

Naïve estimate – choose the best

Better estimate – COMBINE networks “ensembles”

Boosting

Michael Kearns (1988):
"Can a set of weak learners create a single strong learner?"

Weak Learnability (Schapire 90)
Boosting by majority (Freund 95)
Game theory and boosting (Freund & Schapire 96)
Adaboost (Freund & Schapire 97)
Boosting the margin (Schapire et al. 97)
Ref: http://www.cs.princeton.edu/~schapire/boost.html

Leo, 4/24/2000:

Some of my latest efforts are to understand Adaboost better. Its really a strange algorithm with unexpected behavior. … Its become like searching for the Holy Grail!!"

Breiman, 1992 – 1999

1992: Stacked regressions

1993: Nonnegative garrote

1994: Bagging predictors

1996: Bias, variance and arcing classifiers

1997: Arcing the edge

1998: Prediction games and arcing algorithms

1998: Using convex pseudo data to increase prediction accuracy

1998: Randomizing outputs to increase prediction accuracy

1998: Half & half bagging and hard boundary points

1999: Using adaptive bagging to de-bias regressions

1999: Random forests

Motivation: to provide a tool for the understanding and prediction of data.

Leo, 8/16/2000:

"My work on random forests opens up glorious opportunities for graphical displays to exhibit what is driving the classification. Are you interested??"

10/20/2000:

"Let's talk about where to go with this-- one idea I had was to interface it to R. Or maybe S+. I prefer R because its freeware."

Leo, 4/4/2003:

“Sometimes I think that with RF we've got a tiger by the tail - it keeps growing and growing. Oh, well.”

The Random Forest Classifier

Create a collection (ensemble) of trees. Grow each tree on an independent bootstrap sample from the data.

At each node:

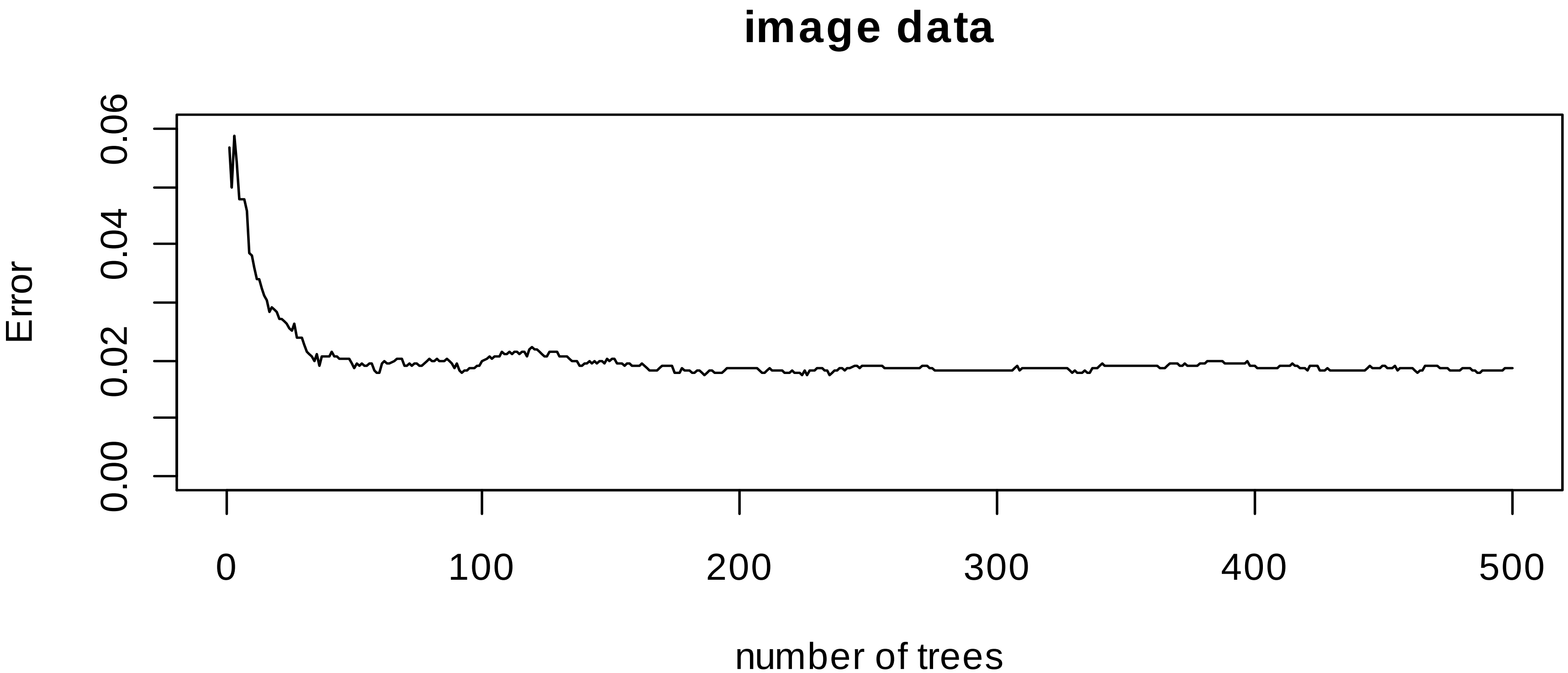
Randomly select ***mtry*** variables out of all ***m*** possible variables (independently for each node).

Find the best split on the selected ***mtry*** variables.

Grow the trees to maximum depth – do not prune.

Vote the trees to get predictions for new data.

“OOB data is used to get a running unbiased estimate of the classification error as trees are added to the forest.”



Out of bag data

Think about a single tree from a random forest:

We grow the tree on a bootstrap sample (“the bag”). About two-thirds of the cases are in the bag.

The remaining one-third are “out-of-bag”.

The out-of-bag data are like a test set for this tree – pass them down the tree and compute their error rate.

Out of bag errors

```
> rfout = randomForest( class ~ . , data = train )
```

```
> mean( predict( rfout ) != train$class )
```

OOB error rate on the training data.

```
> mean( predict( rfout, newdata = train ) != train$class )
```

Zero!

```
> mean( predict( rfout, newdata = test ) != test$class )
```

Error rate on the test data.

The RF Classifier

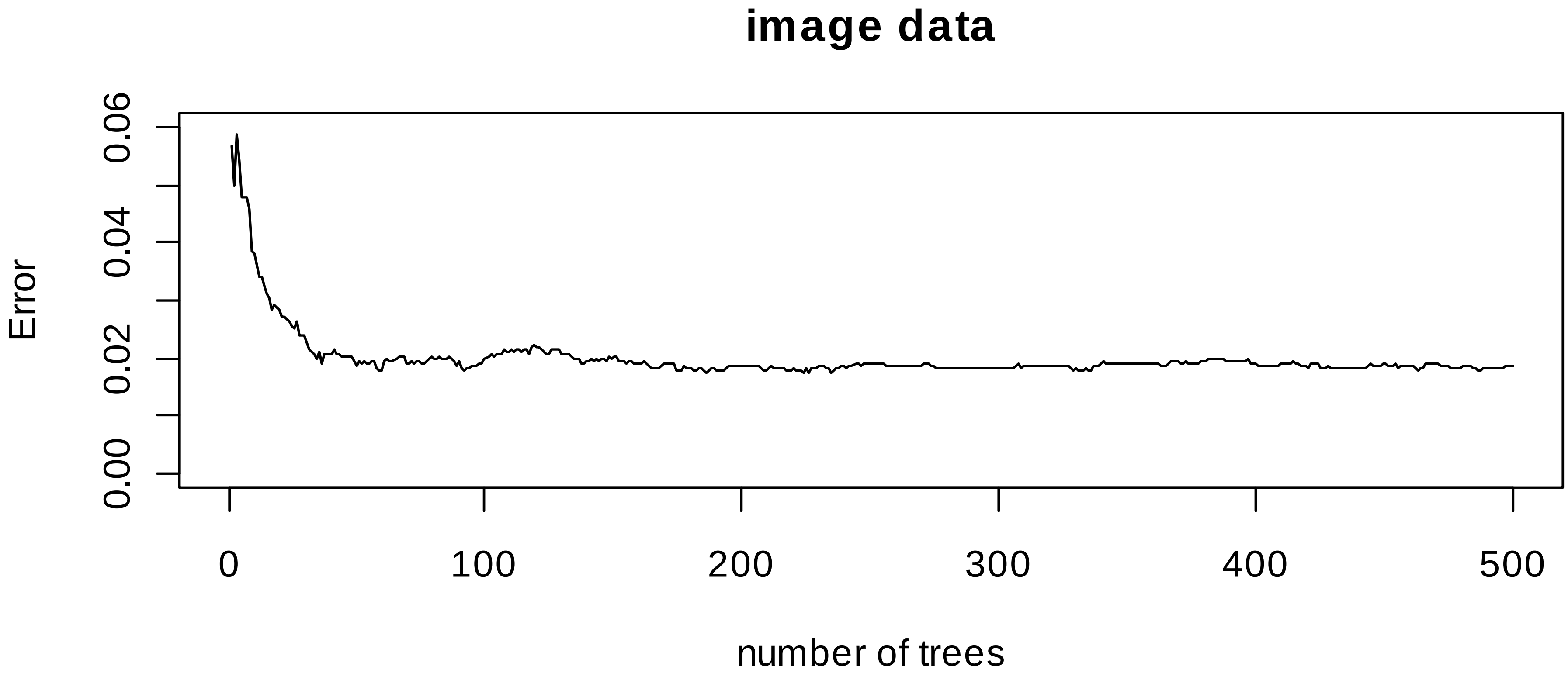
For cases in the training data, vote the trees for which the case is out-of-bag.

$\rightarrow$ "OOB" estimate of error rate.

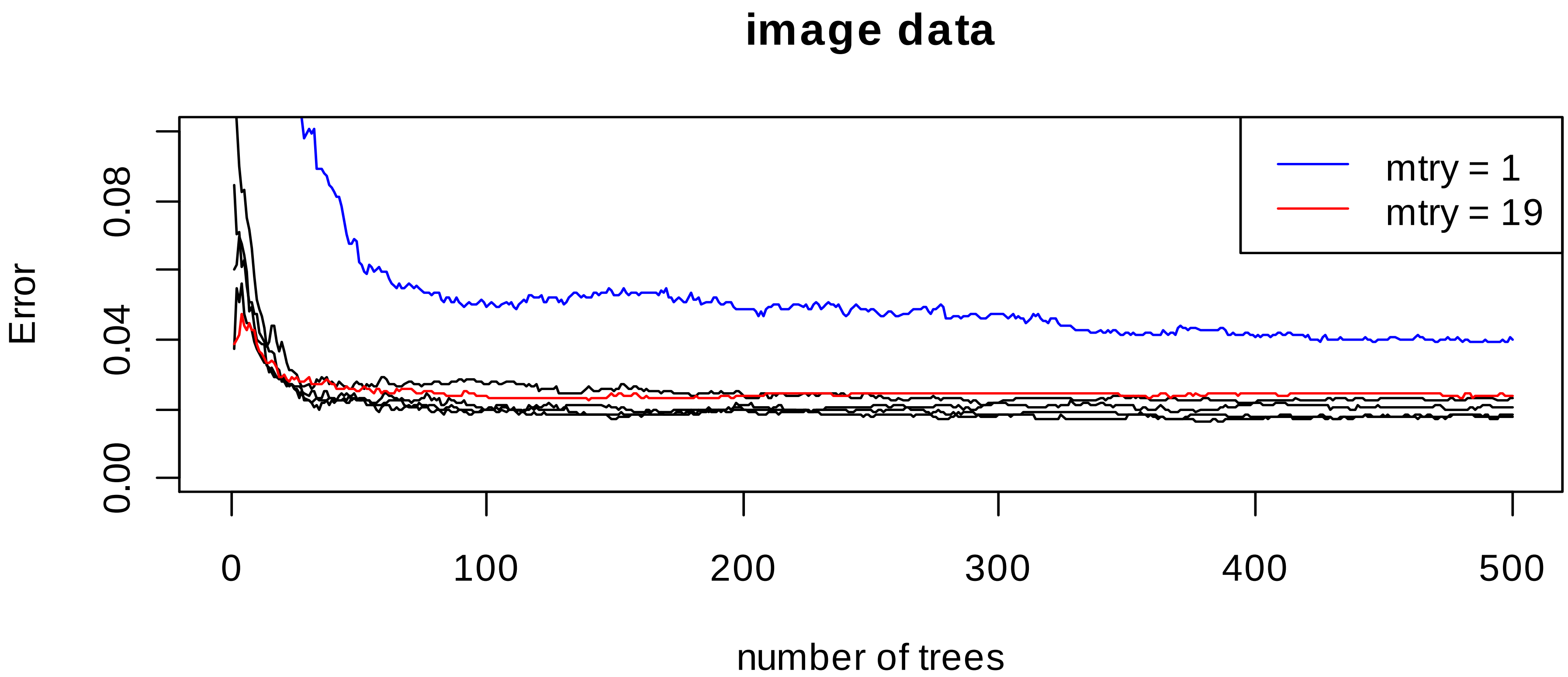
For new cases, vote *all* the trees.

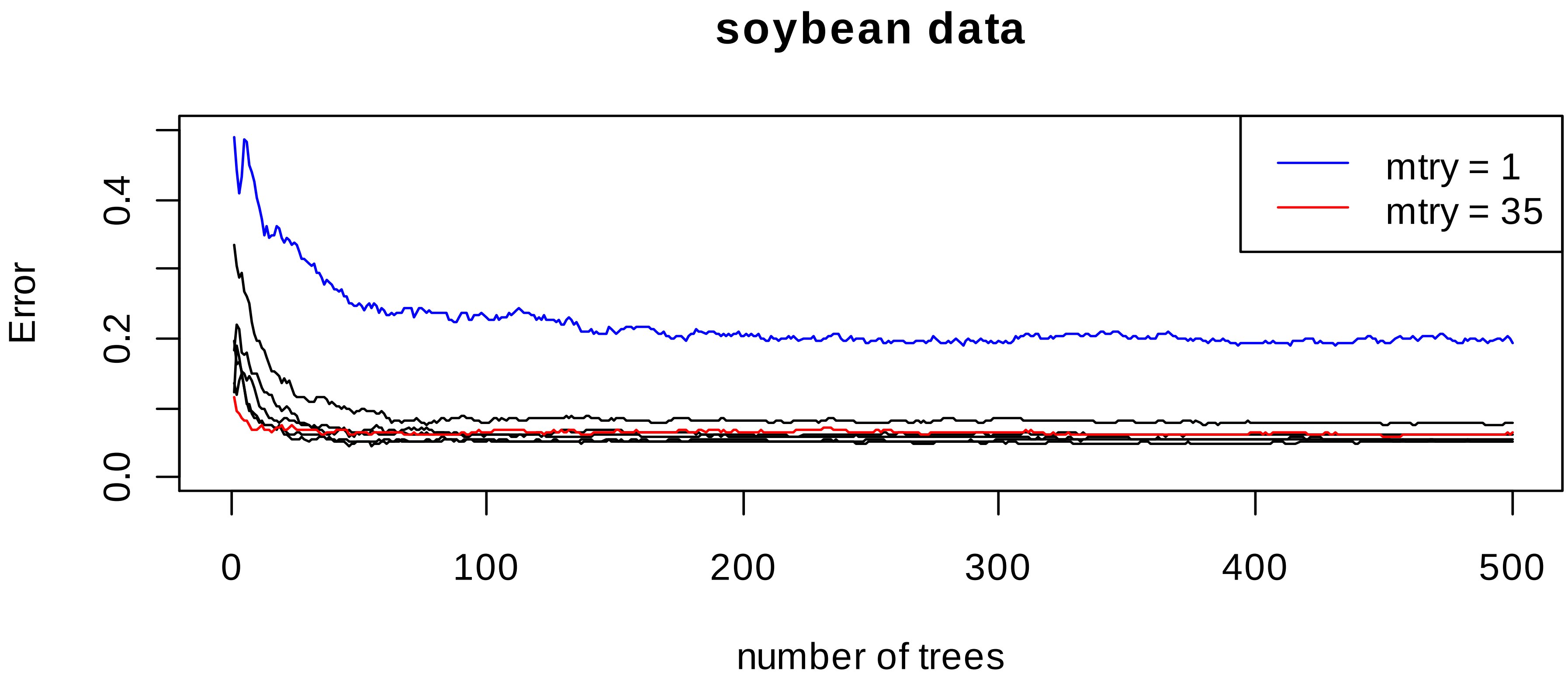
If there are duplicates in the population, the OOB error rate will have negative bias.

“RF does not overfit as more trees are added to the forest.”

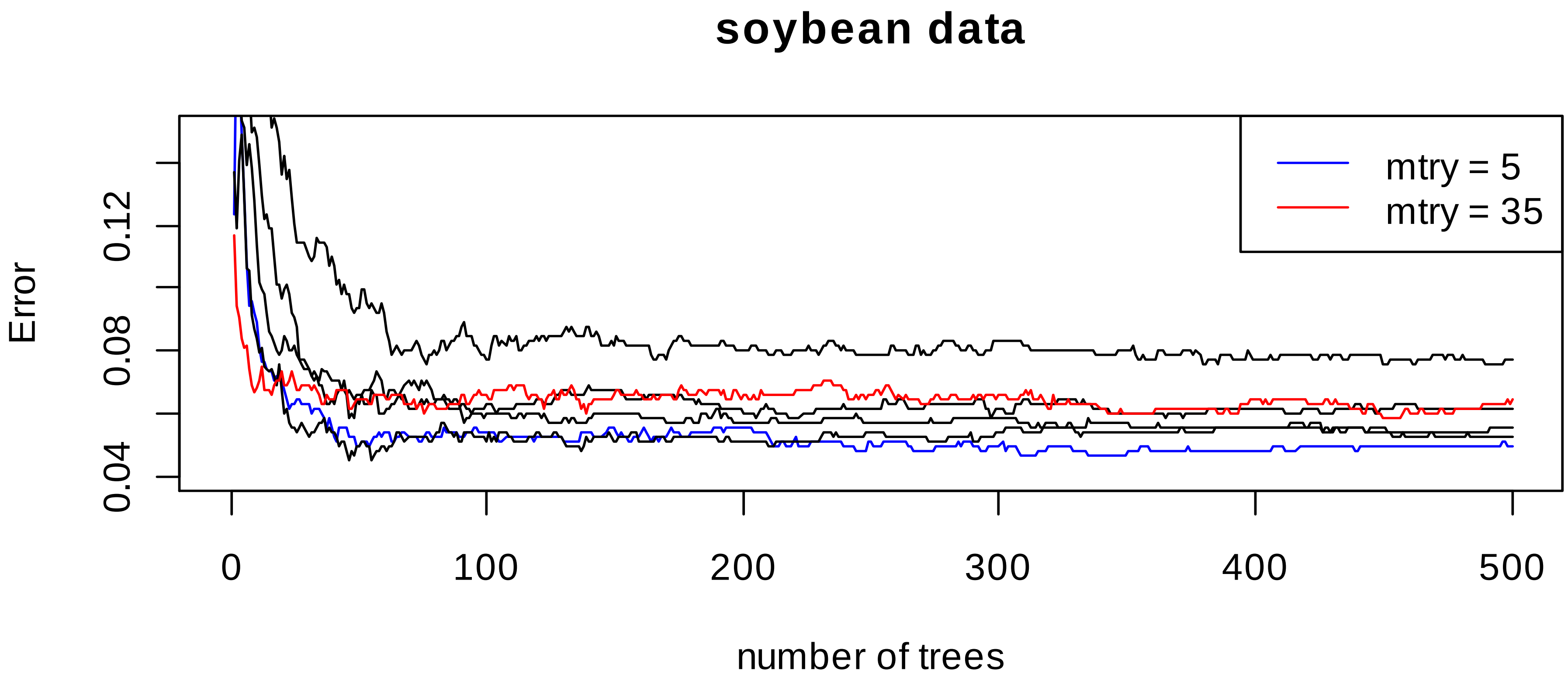


"The error rate in RF is not sensitive to the value of mtry over a very wide range."





soybean data
Error
0.0
0.2
0.4
mtry = 1
mtry = 35
0
100
200
300
400
500
number of trees


soybean data
Error
0.04
0.08
0.12
0
100
200
300
400
500
number of trees
mtry = 5
mtry = 35

Choosing ***mtry***

Start with ***mtry*** equal to the square root of the total number of predictors.

Double it, halve it $\rightarrow$ three OOB error estimates.

If the minimum is at one of the endpoints, try doubling or halving again.

e.g. soybean data, 35 predictors:

mtry = 2,OOB error = .078
mtry = 5,OOB error = .050 $\leftarrow$ use *mtry* = 5
mtry = 10, OOB error = .054
mtry = 6,OOB error = .053

Variable importance

For each tree, look at the out-of-bag data:

Randomly permute the OOB values of variable *j.*

Pass OOB data down the tree $\rightarrow$ predictions.

Subtract:

OOB error rate with variable *j* permuted	–	OOB error rate without permutation

$\rightarrow$ variable importance score

RF error rates with additional noise variables

	No noise added	***10 noise variables***		***100 noise variables***	
Dataset	***Error rate***	***Error rate***	***Ratio***	***Error rate***	***Ratio***
breast	3.1	2.9	**0.93**	2.8	**0.91**
diabetes	23.5	23.8	**1.01**	25.8	**1.10**
ecoli	11.8	13.5	**1.14**	21.2	**1.80**
german	23.5	25.3	**1.07**	28.8	**1.22**
glass	20.4	25.9	**1.27**	37.0	**1.81**
image	1.9	2.1	**1.14**	4.1	**2.22**
iono	6.6	6.5	**0.99**	7.1	**1.07**
liver	25.7	31.0	**1.21**	40.8	**1.59**
sonar	15.2	17.1	**1.12**	21.3	**1.40**
soy	5.3	5.5	**1.06**	7.0	**1.33**
vehicle	25.5	25.0	**0.98**	28.7	**1.12**
votes	4.1	4.6	**1.12**	5.4	**1.33**
vowel	2.6	4.2	**1.59**	17.9	**6.77**

RF variable importance with additional noise variables

Dataset	m	10 noise variables: Number in top m	10 noise variables: Percent	100 noise variables: Number in top m	100 noise variables: Percent
breast	9	9.0	**100.0**	9.0	**100.0**
diabetes	8	7.6	**95.0**	7.3	**91.2**
ecoli	7	6.0	**85.7**	6.0	**85.7**
german	24	20.0	**83.3**	10.1	**42.1**
glass	9	8.7	**96.7**	8.1	**90.0**
image	19	18.0	**94.7**	18.0	**94.7**
ionosphere	34	33.0	**97.1**	33.0	**97.1**
liver	6	5.6	**93.3**	3.1	**51.7**
sonar	60	57.5	**95.8**	48.0	**80.0**
soy	35	35.0	**100.0**	35.0	**100.0**
vehicle	18	18.0	**100.0**	18.0	**100.0**
votes	16	14.3	**89.4**	13.7	**85.6**
vowel	10	10.0	**100.0**	10.0	**100.0**

RF error rates with additional noise variables

Error rates		*Number of noise variables*			
Dataset	*No noise added*	*10*	*100*	*1,000*	*10,000*
breast	3.1	2.9	2.8	3.6	8.9
glass	20.4	25.9	37.0	51.4	61.7
votes	4.1	4.6	5.4	7.8	17.7

Number in top m		*Number of noise variables*			
Dataset	*m*	*10*	*100*	*1,000*	*10,000*
breast	9	9.0	9.0	9	9
glass	9	8.7	8.1	7	6
votes	16	14.3	13.7	13	13

Proximities

Proximity: Pass all the data down all the trees. Proximity between two cases is the proportion of the trees in which the cases end up in the same terminal node.

Proximities *don't* just measure similarity - they take into account the importance of variables.

Two items with ***different*** values on the variables can have ***large*** proximity if they differ only on ***unimportant*** variables.

Two items with ***similar*** values of the variables can have ***small*** proximity if they differ on ***important*** variables.

Getting Pictures from Proximities

To “look” at the data we use classical multidimensional scaling (MDS) to get a picture in 2-D or 3-D:

Proximities $\xrightarrow{\text{MDS}}$ visualization

Idea: points that appear similar to the forest (often in the same terminal node) will be close together on the plot.

Visualizing using proximities

- at-a-glance information about which classes are overlapping, which classes differ
- find clusters within classes
- find easy/hard/unusual cases

With a good tool we can also

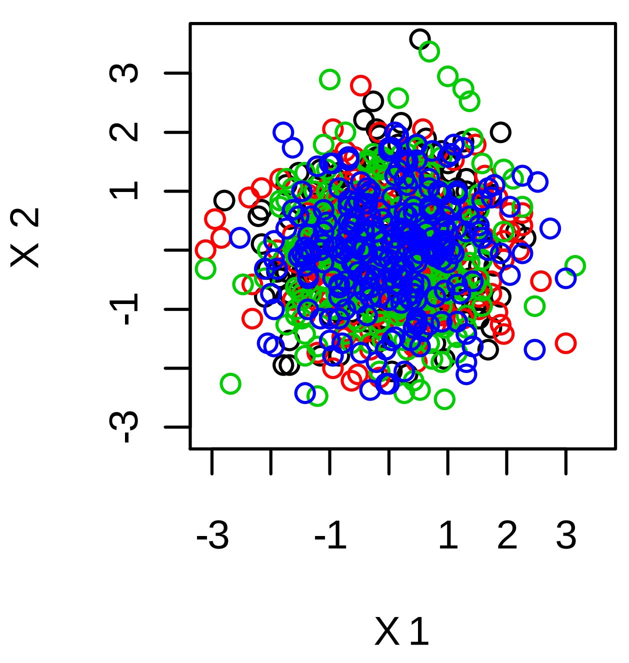
- identify characteristics of unusual points
- see which variables are locally important
- see how clusters or unusual points differ

The Problem with Proximities

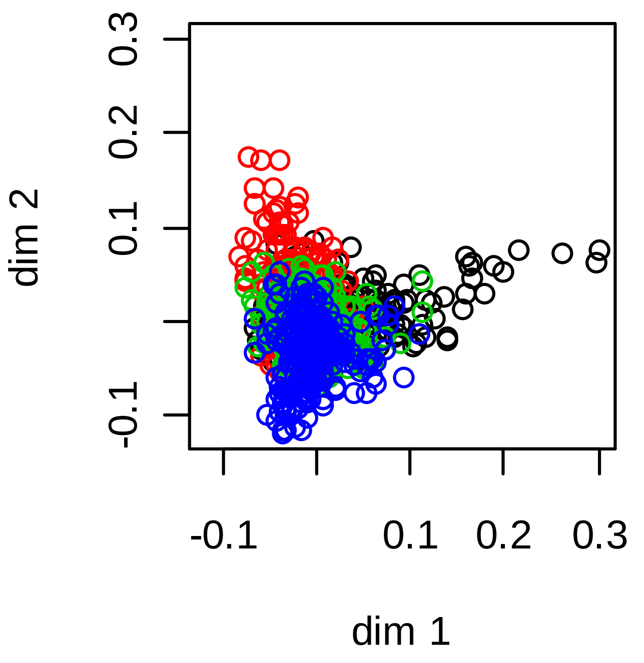
Proximities based on *all* the data overfit!

e.g. two cases from different classes must have proximity zero if trees are grown deep.

Data



MDS



Proximity-weighted Nearest Neighbors

RF is like a nearest-neighbor classifier:

- Use the proximities as weights for nearest-neighbors.
- Classify the training data.
- Compute the error rate.

Want the error rate to be close to the RF OOB error rate.

If we compute proximities from trees in which both cases are OOB, we don't get good accuracy!

Proximity-weighted Nearest Neighbors

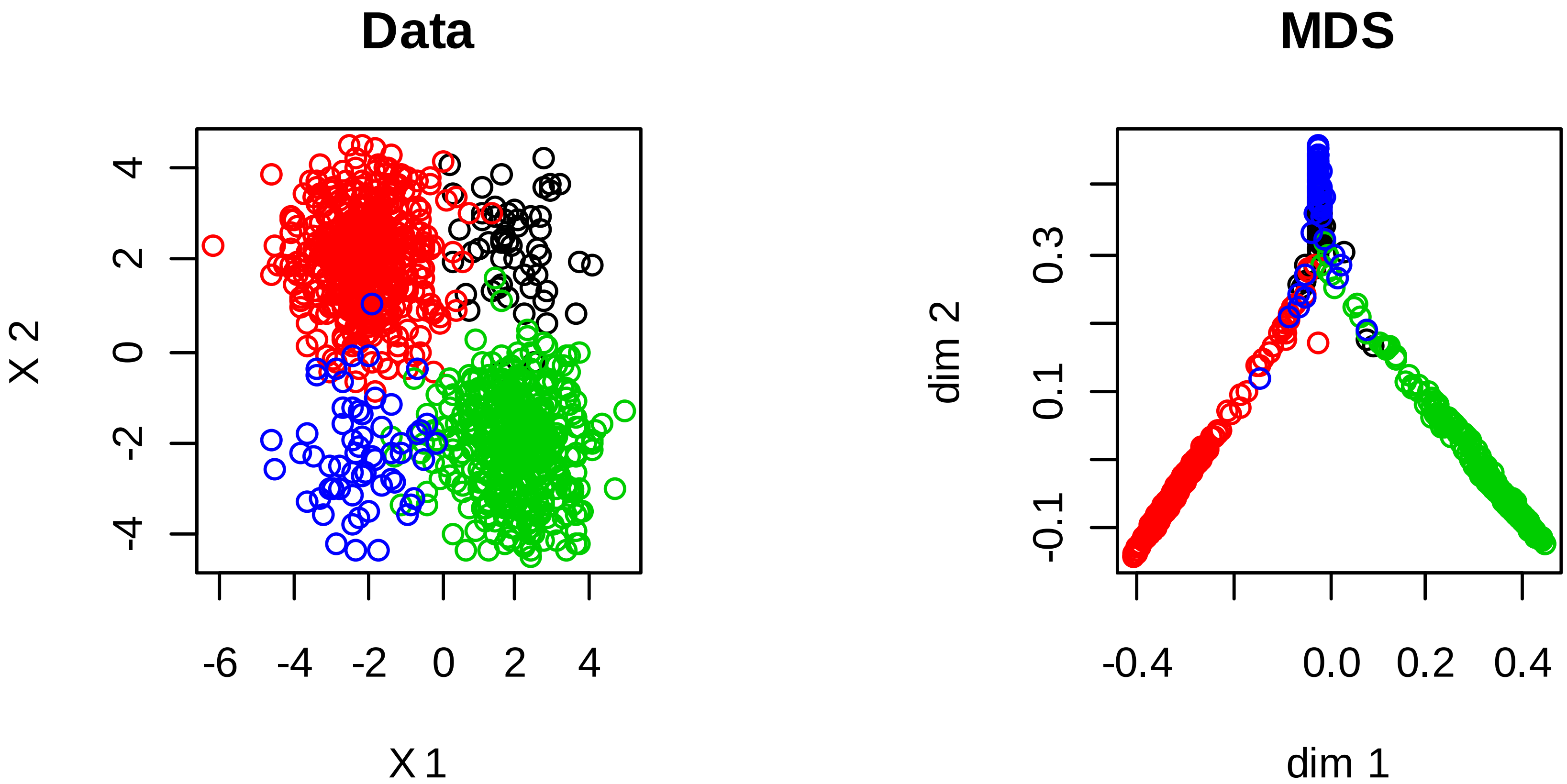
Dataset	*RF*	*OOB*	*New*
breast	2.6	2.9	2.6
diabetes	24.2	23.7	24.4
ecoli	11.6	12.5	11.9
german	23.6	24.1	23.4
glass	20.6	23.8	20.6
image	1.9	2.1	1.9
iono	6.8	6.8	6.8
liver	26.4	26.7	26.4
sonar	13.9	21.6	13.9
soy	5.1	5.4	5.3
vehicle	24.8	27.4	24.8
votes	3.9	3.7	3.7
vowel	2.6	4.5	2.6

Proximity-weighted Nearest Neighbors

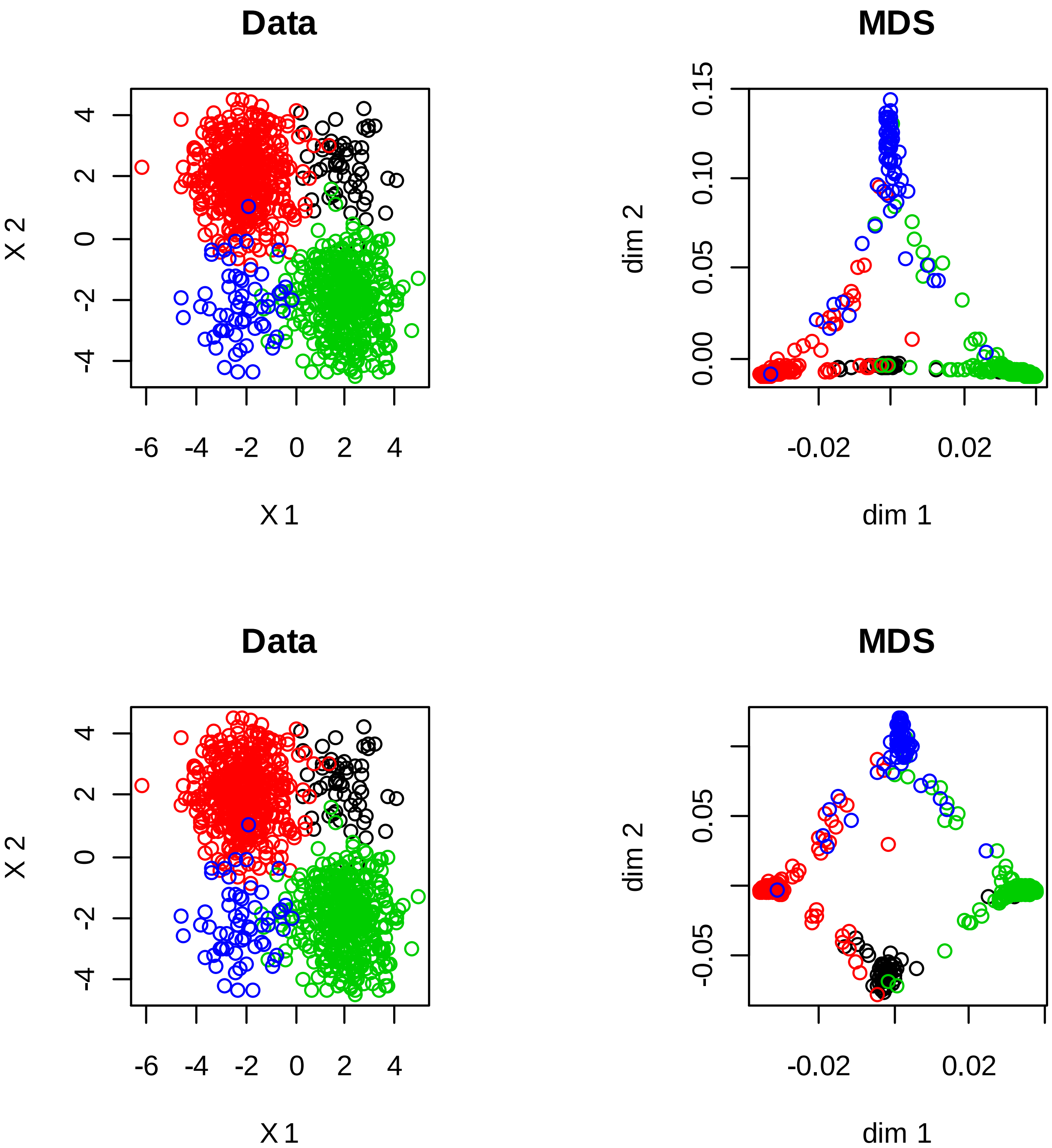
Dataset	*RF*	*OOB*	*New*
Waveform	15.5	16.1	15.5
Twonorm	3.7	4.6	3.7
Threenorm	14.5	15.7	14.5
Ringnorm	5.6	5.9	5.6

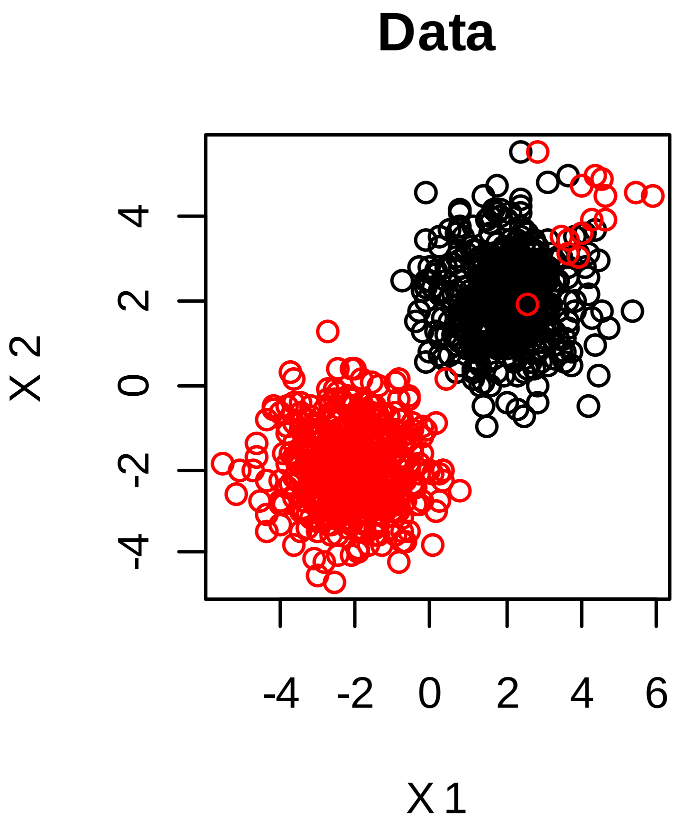
New method to get proximities for observation i:

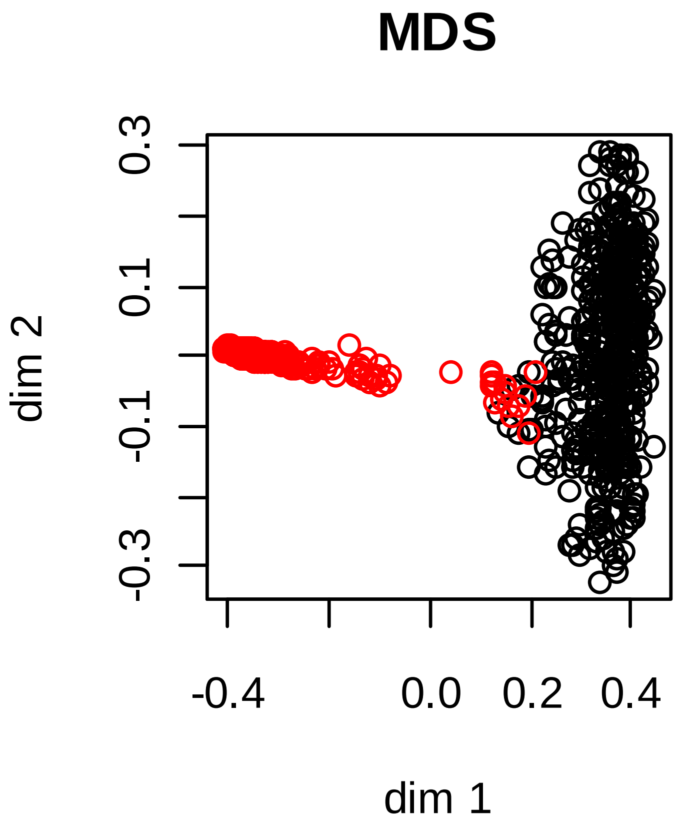
- Pass it down the trees in which it is OOB.
- Increase its proximity to the *k* in-bag cases that are in the same terminal node, by amount 1/k.

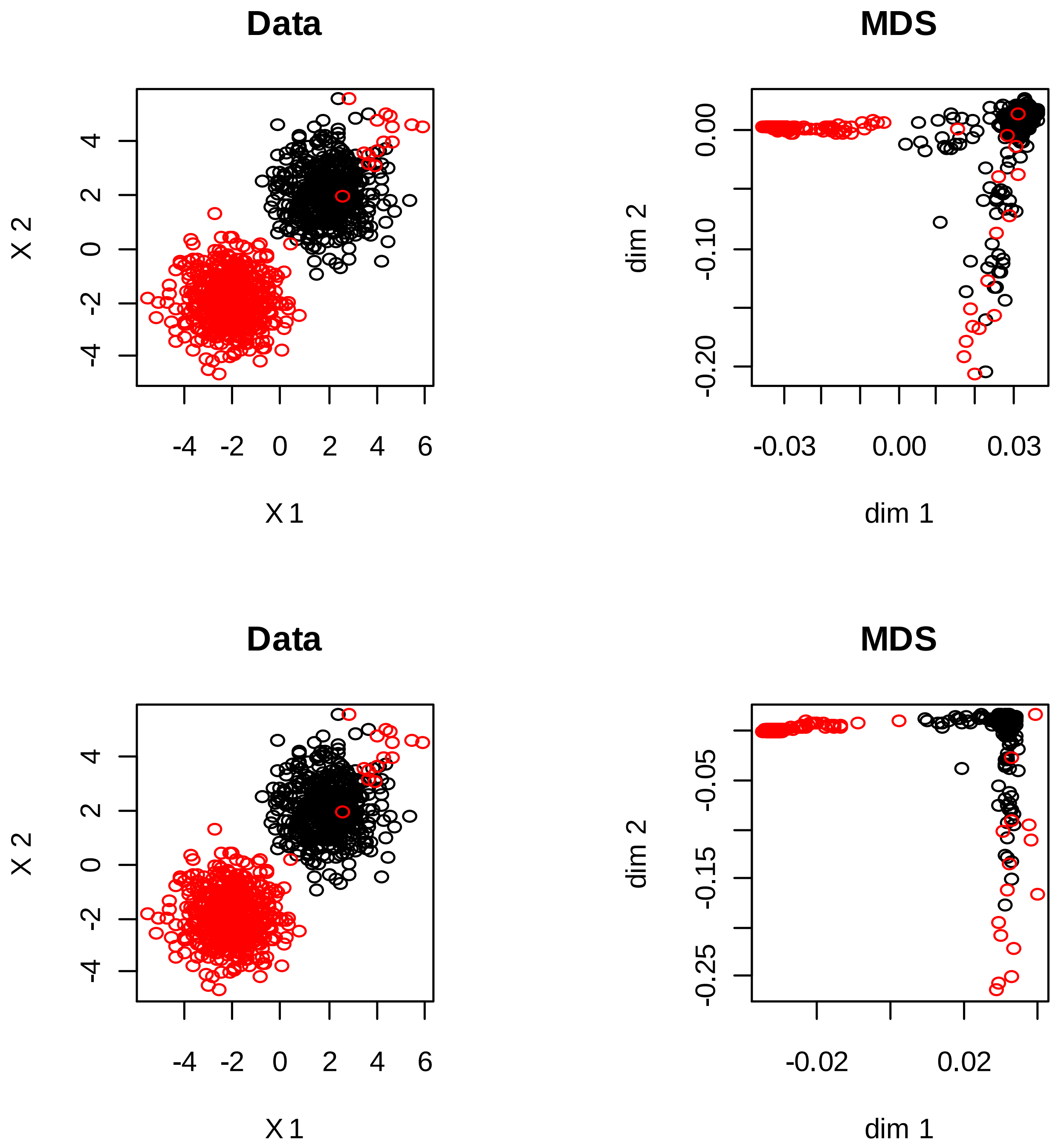


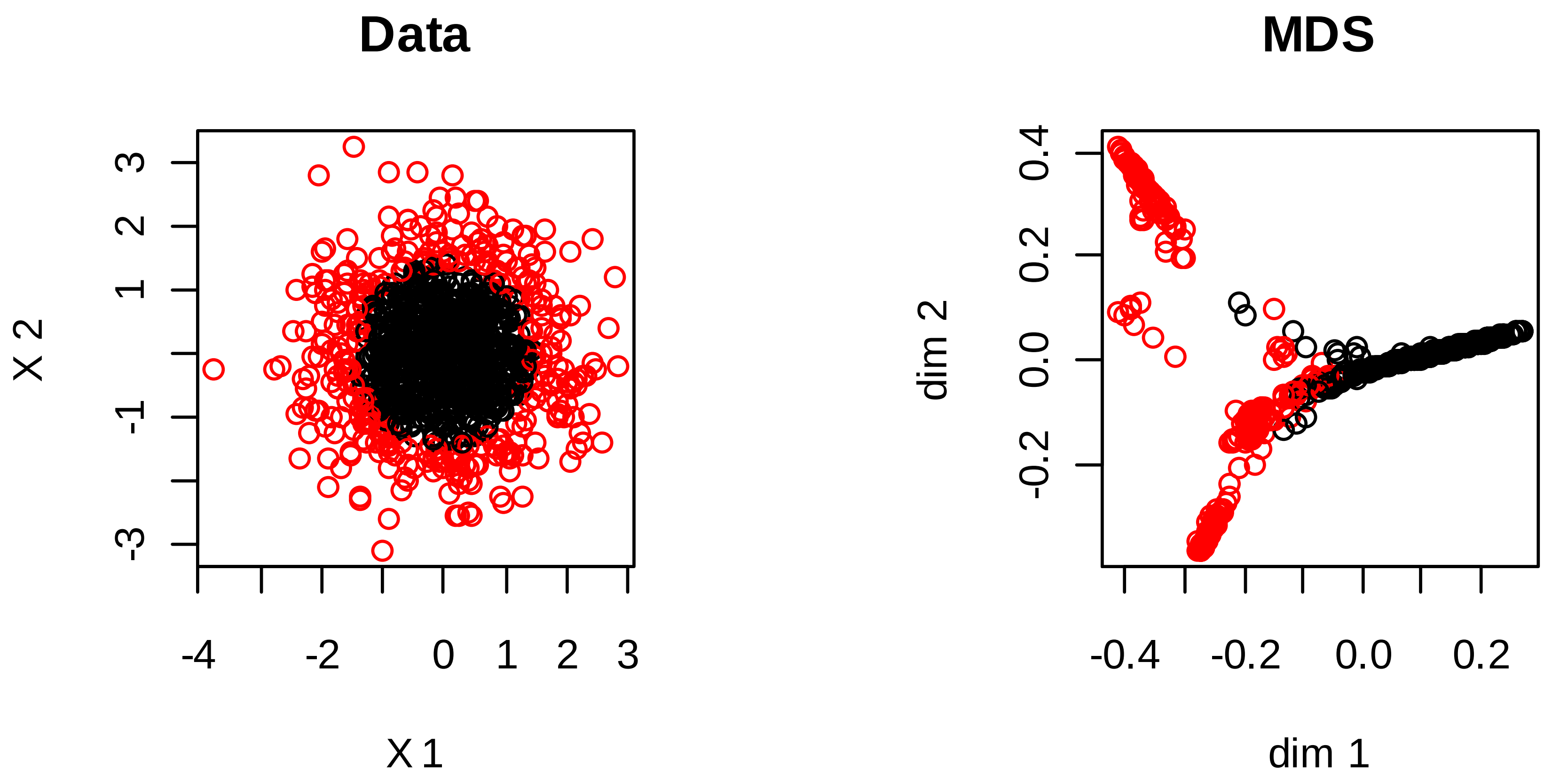
Data
X 2
X 1
4
2
0
-2
-4
-6
-4
-2
0
2
4
MDS
dim 2
dim 1
0.3
0.1
-0.1
-0.4
0.0
0.2
0.4


Data
MDS
X 2
X 1
dim 2
dim 1
-6 -4 -2 0 2 4
4 2 0 -2 -4
-0.02 0.02
0.00 0.05 0.10 0.15
Data
MDS
X 2
X 1
dim 2
dim 1
-6 -4 -2 0 2 4
4 2 0 -2 -4
-0.02 0.02
-0.05 0.05

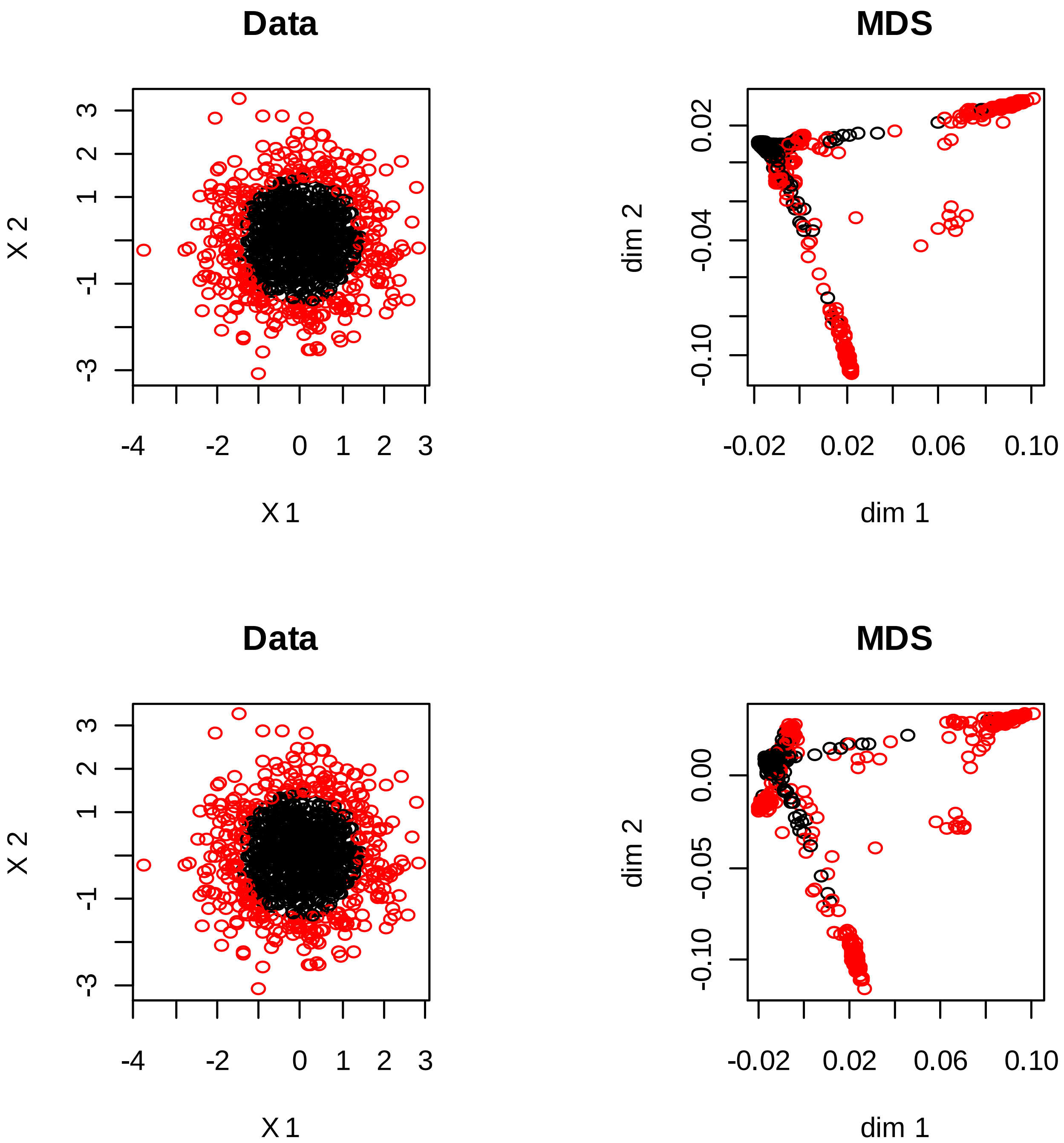

Data
X 2
X 1
-4
-2
0
2
4
6


MDS
dim 2
dim 1
-0.4
0.0
0.2
0.4
0.3
0.1
-0.1
-0.3


Data
X 2
4
2
0
-2
-4
-4
-2
0
2
4
6
X 1
MDS
dim 2
0.00
-0.10
-0.20
-0.03
0.00
0.03
dim 1
Data
X 2
4
2
0
-2
-4
-4
-2
0
2
4
6
X 1
MDS
dim 2
-0.05
-0.15
-0.25
-0.02
0.02
dim 1



Data
X 2
X 1
-4
-2
0
1
2
3
-3
-1
1
2
3
MDS
dim 2
dim 1
-0.4
-0.2
0.0
0.2
0.4


Data
X 1
X 2
-4
-2
0
1
2
3
-3
-1
1
2
3
MDS
dim 1
dim 2
-0.02
0.02
0.06
0.10
-0.10
-0.04
0.02
Data
X 1
X 2
MDS
dim 1
dim 2
-0.02
0.02
0.06
0.10
-0.10
-0.05
0.00

RAFT
RAndom Forests graphics Tool

- java-based, stand-alone application
- uses output files from the fortran code
- download RAFT from

  www.math.usu.edu/~adele/forests/cc_graphics.htm

Raft uses

  VisAD www.ssec.wisc.edu/~billh/visad.html

  ImageJ http://rsb.info.nih.gov/ij