Political Science 452: Text as Data

Justin Grimmer

Assistant Professor
Department of Political Science
Stanford University

June 8th, 2011

Where We've Been, Where We're Going

- Class 1: Finding Text Data
- Class 2: Representing Texts Quantitatively
- Class 3: Dictionary Methods for Classification
- Class 4: Comparing Language Across Groups
- Class 5: Texts in Space
- Class 6: Clustering
- Class 7: Topic models
- Class 8: Supervised methods for classification
- Class 9: Ensemble methods for classification
- Class 10: Scaling Speech

Cross validation, Ensembles, and Super learning

Classification:

Cross validation, Ensembles, and Super learning

Classification:

- Models for categorizing texts

Cross validation, Ensembles, and Super learning

Classification:

- Models for categorizing texts
- Know (develop) categories before hand

Cross validation, Ensembles, and Super learning

Classification:

- Models for categorizing texts
- Know (develop) categories before hand
- Hand coding: assign documents to categories
- Infer: new document assignment to categories (distribution of documents to categories)

Cross validation, Ensembles, and Super learning

Classification:

- Models for categorizing texts
- Know (develop) categories before hand
- Hand coding: assign documents to categories
- Infer: new document assignment to categories (distribution of documents to categories)
- This week: how to select method?
- Combining many methods

Cross validation, Ensembles, and Super learning

Classification:

- Models for categorizing texts
- Know (develop) categories before hand
- Hand coding: assign documents to categories
- Infer: new document assignment to categories (distribution of documents to categories)
- This week: how to select method?
- Combining many methods

Scaling: (when we get there!)

Methods for Classification

Three supervised methods (there are many!)

1) Naive Bayes:
   - Training set: Construct model of what documents "look like"
   - Test set: Assign documents to categories, based on similarity to categories
2) ReadMe:
   - Focus on estimating proportions only
   - Training set: construct model of stem profiles in categories
   - Test set: linear algebra solution to problem (modulo dimensionality)
3) Support Vector Machines
   - Training set: identify separating plane between two classes
   - Test set: classify based on location to separating plane

Support Vector Machines

Document i is an $M \times 1$ vector of counts

$$\mathbf{y}_i = (y_{1i}, y_{2i}, \ldots, y_{Mi})$$

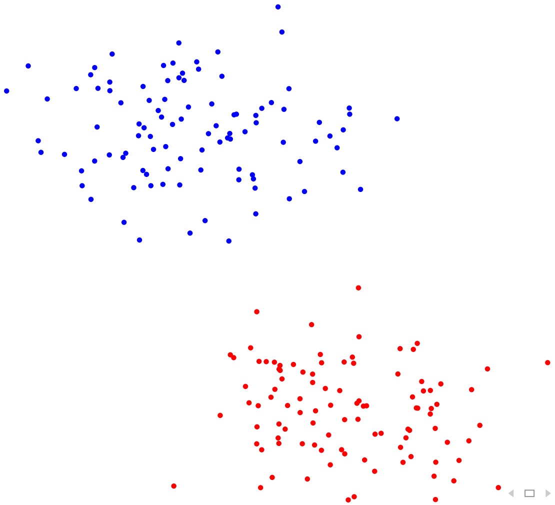
Suppose we have two classes, c_1, c_2.

$$t_i = 1 \text{ if } i \text{ is in class 1}$$
$$t_i = -1 \text{ if } i \text{ is in class 2}$$

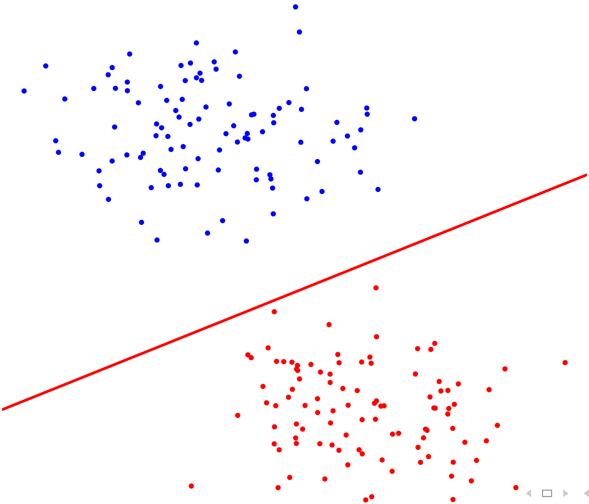
Suppose they are separable:

- Draw a line between groups
- Goal: identify the line in the middle
- Maximum margin

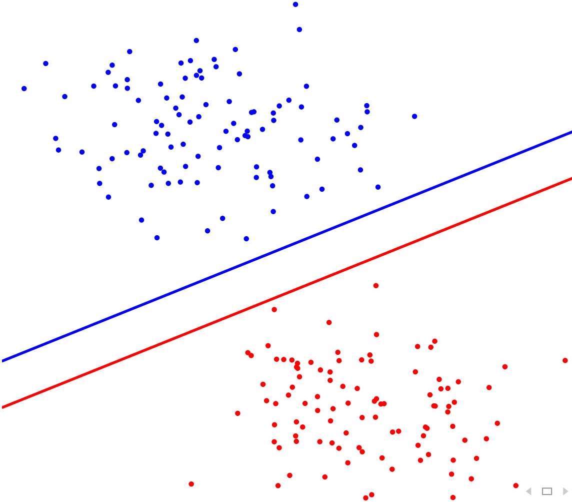
Support Vector Machines: Maximum Margin Classifier (Bishop 2006)



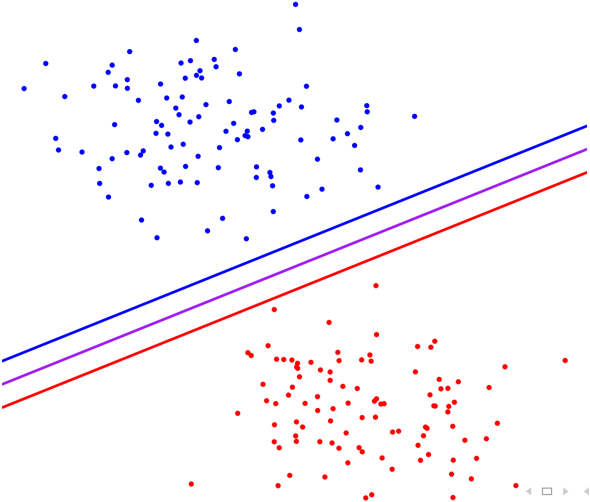
Support Vector Machines: Maximum Margin Classifier (Bishop 2006)



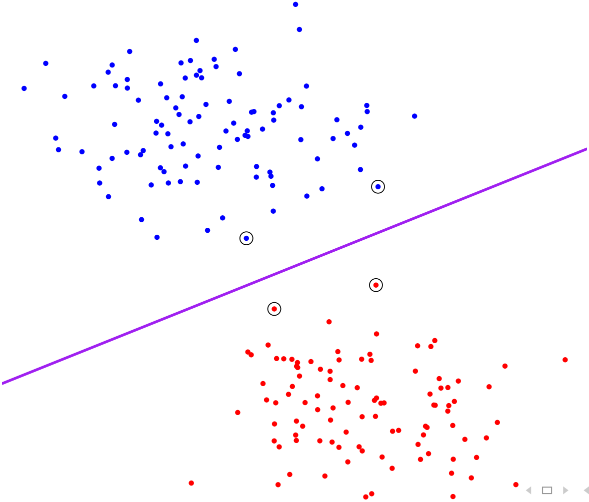
Support Vector Machines: Maximum Margin Classifier (Bishop 2006)



Support Vector Machines: Maximum Margin Classifier (Bishop 2006)



Support Vector Machines: Maximum Margin Classifier (Bishop 2006)



Support Vector Machines: Algebra (Bishop 2006)

Goal create a score to classify:

$$s(\mathbf{y}_i) \;\; = \;\; \boldsymbol{\beta}^{'}\mathbf{y}_i + b$$

- $\boldsymbol{\beta}$ Determines orientation of surface (slope)
- b determines location (moves surface up or down)
- If $s(\mathbf{y}_i) > 0 \rightarrow$ class 1
- If $s(\mathbf{y}_i) < 0 \rightarrow$ class 2
- $\frac{|s(\mathbf{y}_i)|}{||\boldsymbol{\beta}||} =$ Document distance from decision surface (margin)

Support Vector Machines: Algebra (Bishop 2006)

Objective function: maximum margin

Support Vector Machines: Algebra (Bishop 2006)

Objective function: maximum margin

$\min_i [\, |(s(\mathbf{y}_i)| \,]$: Point closest to decision surface

Support Vector Machines: Algebra (Bishop 2006)

Objective function: maximum margin

$\min_i[\ |(s(\mathbf{y}_i)|\ ]$: Point closest to decision surface

We want to identify β and b to maximize the margin:

Support Vector Machines: Algebra (Bishop 2006)

Objective function: maximum margin

$\min_i[\ |(s(\mathbf{y}_i)|\ ]$: Point closest to decision surface

We want to identify β and b to maximize the margin:

$$\arg\max_{\beta,b}\left\{\frac{1}{||\beta||}\min_i[\ |(s(\mathbf{y}_i)|\ ]\right\}$$

Support Vector Machines: Algebra (Bishop 2006)

Objective function: maximum margin

$\min_i[\ |(s(\mathbf{y}_i)|\ ]$: Point closest to decision surface

We want to identify $\boldsymbol{\beta}$ and b to maximize the margin:

$$\arg\max_{\boldsymbol{\beta},b}\left\{\frac{1}{||\boldsymbol{\beta}||}\min_i[\ |(s(\mathbf{y}_i)|\ ]\right\}$$

$$\arg\max_{\boldsymbol{\beta},b}\left\{\frac{1}{||\boldsymbol{\beta}||}\min_i[\ |\boldsymbol{\beta}^{'}\mathbf{y}_i + b|\ ]\right\}$$

Support Vector Machines: Algebra (Bishop 2006)

Objective function: maximum margin

$\min_i[\ |(s(\mathbf{y}_i)|\ ]$: Point closest to decision surface

We want to identify $\boldsymbol{\beta}$ and b to maximize the margin:

$$\arg\max_{\boldsymbol{\beta}, b} \left\{ \frac{1}{||\boldsymbol{\beta}||} \min_i[\ |(s(\mathbf{y}_i)|\ ] \right\}$$

$$\arg\max_{\boldsymbol{\beta}, b} \left\{ \frac{1}{||\boldsymbol{\beta}||} \min_i[\ |\boldsymbol{\beta}^{'} \mathbf{y}_i + b|\ ] \right\}$$

Constrained optimization problem

What About Overlap? (Bishop 2006)

- Rare that classes are separable.

What About Overlap? (Bishop 2006)

- Rare that classes are separable.
- Define:

What About Overlap? (Bishop 2006)

- Rare that classes are separable.
- Define:

$$\xi_i = 0 \text{ if correctly classified}$$

What About Overlap? (Bishop 2006)

- Rare that classes are separable.
- Define:

$$\begin{aligned} \xi_i &= 0 \text{ if correctly classified} \\ \xi_i &= |s(\mathbf{y}_i)| \text{ if incorrectly classified} \end{aligned}$$

What About Overlap? (Bishop 2006)

- Rare that classes are separable.
- Define:

$$
\begin{aligned}
\xi_i &= 0 \text{ if correctly classified} \\
\xi_i &= |s(\mathbf{y}_i)| \text{ if incorrectly classified}
\end{aligned}
$$

Tradeoff:

What About Overlap? (Bishop 2006)

- Rare that classes are separable.
- Define:

$$\begin{aligned}
\xi_i &= 0 \text{ if correctly classified} \\
\xi_i &= |s(\mathbf{y}_i)| \text{ if incorrectly classified}
\end{aligned}$$

Tradeoff:

- Maximize margin between correctly classified groups

What About Overlap? (Bishop 2006)

- Rare that classes are separable.
- Define:

$$\begin{aligned} \xi_i &= 0 \text{ if correctly classified} \\ \xi_i &= |s(\mathbf{y}_i)| \text{ if incorrectly classified} \end{aligned}$$

Tradeoff:

- Maximize margin between correctly classified groups
- Minimize error from misclassified documents

What About Overlap? (Bishop 2006)

- Rare that classes are separable.
- Define:

$$
\begin{aligned}
\xi_i &= 0 \text{ if correctly classified} \\
\xi_i &= |s(\mathbf{y}_i)| \text{ if incorrectly classified}
\end{aligned}
$$

Tradeoff:

- Maximize margin between correctly classified groups
- Minimize error from misclassified documents

$$
\text{arg max}_{\boldsymbol{\beta},b}\left\{ C\sum_{i=1}^{N}\xi_i + \frac{1}{||\boldsymbol{\beta}||}\ \text{min}_i[\ |\boldsymbol{\beta}^{'}\mathbf{y}_i + b|\ ]\right\}
$$

What About Overlap? (Bishop 2006)

- Rare that classes are separable.
- Define:

$$\begin{aligned} \xi_i &= 0 \text{ if correctly classified} \\ \xi_i &= |s(\mathbf{y}_i)| \text{ if incorrectly classified} \end{aligned}$$

Tradeoff:

- Maximize margin between correctly classified groups
- Minimize error from misclassified documents

$$\text{arg max}_{\boldsymbol{\beta},b} \left\{ C \sum_{i=1}^{N} \xi_i + \frac{1}{||\boldsymbol{\beta}||} \text{min}_i[\ |\boldsymbol{\beta}^{'} \mathbf{y}_i + b|\ ] \right\}$$

C captures tradeoff

What About Overlap? (Bishop 2006)

- Rare that classes are separable.
- Define:

$$
\begin{aligned}
\xi_i &= 0 \text{ if correctly classified} \\
\xi_i &= |s(\mathbf{y}_i)| \text{ if incorrectly classified}
\end{aligned}
$$

Tradeoff:

- Maximize margin between correctly classified groups
- Minimize error from misclassified documents

$$
\text{arg max}_{\boldsymbol{\beta},b} \left\{ C \sum_{i=1}^{N} \xi_i + \frac{1}{||\boldsymbol{\beta}||} \text{ min}_i [\ |\boldsymbol{\beta}^{'} \mathbf{y}_i + b|\ ] \right\}
$$

C captures tradeoff

How to Handle Multiple Comparisons?

- Rare that we only want to classify two categories

How to Handle Multiple Comparisons?

- Rare that we only want to classify two categories
- How to handle classification into K groups?

How to Handle Multiple Comparisons?

- Rare that we only want to classify two categories
- How to handle classification into K groups?
  1) Set up K classification problems:

How to Handle Multiple Comparisons?

- Rare that we only want to classify two categories
- How to handle classification into K groups?
    1) Set up K classification problems:
        - Compare each class to all other classes

How to Handle Multiple Comparisons?

- Rare that we only want to classify two categories
- How to handle classification into K groups?
  1) Set up K classification problems:
     - Compare each class to all other classes
     - Problem: can lead to inconsistent results

How to Handle Multiple Comparisons?

- Rare that we only want to classify two categories
- How to handle classification into K groups?
    1) Set up K classification problems:
        - Compare each class to all other classes
        - Problem: can lead to inconsistent results
        - Solution(?): select category with largest "score"

How to Handle Multiple Comparisons?

- Rare that we only want to classify two categories
- How to handle classification into K groups?
    1) Set up K classification problems:
        - Compare each class to all other classes
        - Problem: can lead to inconsistent results
        - Solution(?): select category with largest "score"
        - Problem: scales are not comparable

How to Handle Multiple Comparisons?

- Rare that we only want to classify two categories
- How to handle classification into K groups?
  1) Set up K classification problems:
     - Compare each class to all other classes
     - Problem: can lead to inconsistent results
     - Solution(?): select category with largest “score”
     - Problem: scales are not comparable
  2) Common solution: set up $K(K-1)/2$ classifications

How to Handle Multiple Comparisons?

- Rare that we only want to classify two categories
- How to handle classification into K groups?
  1) Set up K classification problems:
     - Compare each class to all other classes
     - Problem: can lead to inconsistent results
     - Solution(?): select category with largest "score"
     - Problem: scales are not comparable
  2) Common solution: set up $K(K-1)/2$ classifications
  - Perform vote to select class (still suboptimal)

How to Handle Multiple Comparisons?

- Rare that we only want to classify two categories
- How to handle classification into K groups?
  1) Set up K classification problems:
     - Compare each class to all other classes
     - Problem: can lead to inconsistent results
     - Solution(?): select category with largest "score"
     - Problem: scales are not comparable
  2) Common solution: set up $K(K-1)/2$ classifications
     - Perform vote to select class (still suboptimal)
  3) Simultaneous estimation possible, much slower

R Code to Run SVMs

```
library(e1071)
fit<- svm(T . , as.data.frame(tdm) , method ='C',
kernel='linear')
```

where: `method = 'C'` → Classification

`kernel='linear'` → allows for distortion of feature space. Options:

- Linear
- Polynomial
- Radial
- sigmoid

```
preds<- predict(fit, data =
as.data.frame(tdm[-c(1:no.train),]))
```

Example of SVMs in Political Science Research

Hillard, Purpura, Wilkerson: SVMs to code topic/sub topics for policy agendas project

TABLE 3. Bill Title Interannotator Agreement for Five Model Types

	SVM	MaxEnt	Boostexter	Naïve Bayes
Major topic *N* = 20	88.7% (.881)	86.5% (.859)	85.6% (.849)	81.4% (.805)
Subtopic *N* = 226	81.0% (.800)	78.3% (.771)	73.6% (.722)	71.9% (.705)

SVMs are under utilized in political science

Assessing Models (Elements of Statistical Learning)

- Model Selection: tuning parameters to select final model
- Model assessment : after selecting model, estimating error in classification (last week's discussion)

How Do We Build A Model?

There are many ways to fit models

And many choices made when performing model fit

How do we choose?

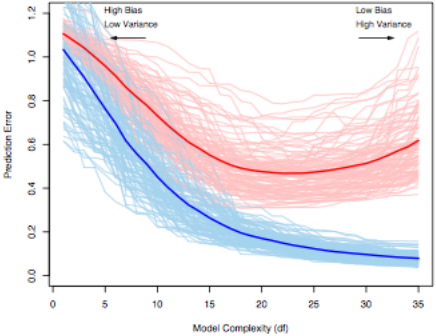


FIGURE 7.1. *Behavior of test sample and training sample error as the model complexity is varied. The light blue curves show the training error* $\overline{\mathrm{err}}$, *while the light red curves show the conditional test error* $\mathrm{Err}_{\mathcal{T}}$ *for 100 training sets of size 50 each, as the model complexity is increased. The solid curves show the expected test error* Err *and the expected training error* $\mathrm{E}[\overline{\mathrm{err}}]$.

How Do We Build A Model?

There are many ways to fit models

And many choices made when performing model fit

How do we choose?

Bad way to choose:

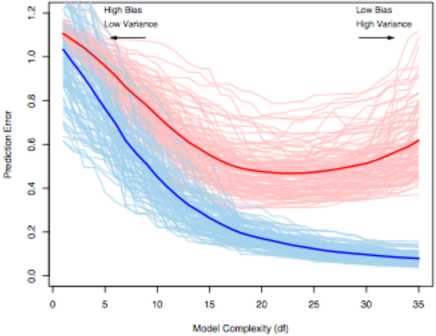


FIGURE 7.1. *Behavior of test sample and training sample error as the model complexity is varied. The light blue curves show the training error* $\overline{\text{err}}$*, while the light red curves show the conditional test error* $\text{Err}_{\mathcal{T}}$ *for 100 training sets of size 50 each, as the model complexity is increased. The solid curves show the expected test error* Err *and the expected training error* $\text{E}[\overline{\text{err}}]$.

How Do We Build A Model?

There are many ways to fit models

And many choices made when performing model fit

How do we choose?

Bad way to choose: within sample model fit (HTF Figure 7.1)

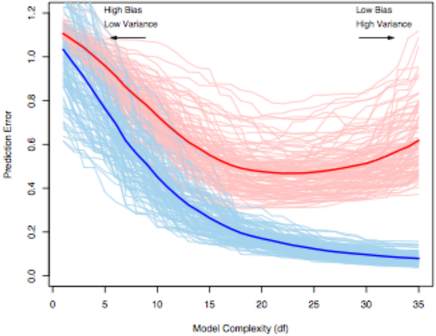


FIGURE 7.1. *Behavior of test sample and training sample error as the model complexity is varied. The light blue curves show the training error* $\overline{\text{err}}$*, while the light red curves show the conditional test error* $\text{Err}_{\mathcal{T}}$ *for 100 training sets of size 50 each, as the model complexity is increased. The solid curves show the expected test error* Err *and the expected training error* $\text{E}[\overline{\text{err}}]$.

How Do We Build A Model?

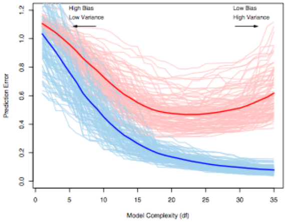


FIGURE 7.1. *Behavior of test sample and training sample error as the model complexity is varied. The light blue curves show the training error* $\overline{\text{err}}$*, while the light red curves show the conditional test error* $\text{Err}_{\mathcal{T}}$ *for 100 training sets of size 50 each, as the model complexity is increased. The solid curves show the expected test error* Err *and the expected training error* $\text{E}[\overline{\text{err}}]$.

Model overfit:

How Do We Build A Model?

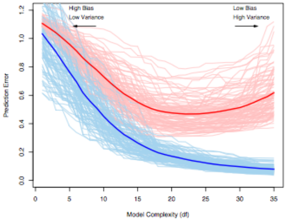


FIGURE 7.1. *Behavior of test sample and training sample error as the model complexity is varied. The light blue curves show the training error* $\overline{\mathrm{err}}$*, while the light red curves show the conditional test error* $\mathrm{Err}_{\mathcal{T}}$ *for 100 training sets of size 50 each, as the model complexity is increased. The solid curves show the expected test error* Err *and the expected training error* $\mathrm{E}[\overline{\mathrm{err}}]$.

Model overfit:

- Some model complexity captures systematic features of the data

How Do We Build A Model?

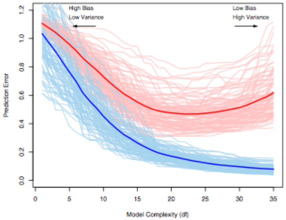


FIGURE 7.1. *Behavior of test sample and training sample error as the model complexity is varied. The light blue curves show the training error* $\overline{\text{err}}$, *while the light red curves show the conditional test error* $\text{Err}_{\mathcal{T}}$ *for 100 training sets of size 50 each, as the model complexity is increased. The solid curves show the expected test error* Err *and the expected training error* $\text{E}[\overline{\text{err}}]$.

Model overfit:

- Some model complexity captures systematic features of the data
- Characteristics found in both training and test set

How Do We Build A Model?

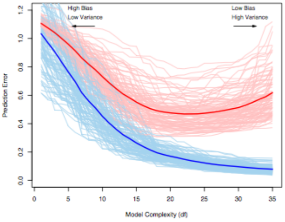


FIGURE 7.1. *Behavior of test sample and training sample error as the model complexity is varied. The light blue curves show the training error* $\overline{\text{err}}$*, while the light red curves show the conditional test error* $\text{Err}_{\mathcal{T}}$ *for 100 training sets of size 50 each, as the model complexity is increased. The solid curves show the expected test error* Err *and the expected training error* $\text{E}[\overline{\text{err}}]$.

Model overfit:

- Some model complexity captures systematic features of the data
- Characteristics found in both training and test set
- Reduces error in both training and test set

How Do We Build A Model?

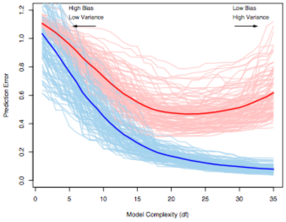


FIGURE 7.1. *Behavior of test sample and training sample error as the model complexity is varied. The light blue curves show the training error* $\overline{\text{err}}$*, while the light red curves show the conditional test error* $\text{Err}_{\mathcal{T}}$ *for 100 training sets of size 50 each, as the model complexity is increased. The solid curves show the expected test error* Err *and the expected training error* $\text{E}[\overline{\text{err}}]$.

Model overfit:

- Some model complexity captures systematic features of the data
- Characteristics found in both training and test set
- Reduces error in both training and test set
- Additional model complexity: idiosyncratic features of the training set

How Do We Build A Model?

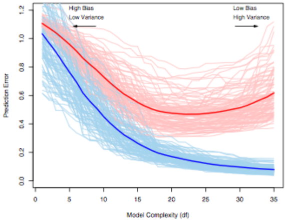


FIGURE 7.1. *Behavior of test sample and training sample error as the model complexity is varied. The light blue curves show the training error* $\overline{\text{err}}$*, while the light red curves show the conditional test error* $\text{Err}_{\mathcal{T}}$ *for 100 training sets of size 50 each, as the model complexity is increased. The solid curves show the expected test error* Err *and the expected training error* $\text{E}[\overline{\text{err}}]$.

Model overfit:

- Some model complexity captures systematic features of the data
- Characteristics found in both training and test set
- Reduces error in both training and test set
- Additional model complexity: idiosyncratic features of the training set
- Reduces error in training set, increases error in test set

How Do We Select A Model?

Analytic statistics for selection, include penalty for complexity

How Do We Select A Model?

Analytic statistics for selection, include penalty for complexity

- AIC : Akaka Information Criterion

How Do We Select A Model?

Analytic statistics for selection, include penalty for complexity

- AIC : Akaka Information Criterion
- BIC: Bayesian Information Criterion

How Do We Select A Model?

Analytic statistics for selection, include penalty for complexity

- AIC : Akaka Information Criterion
- BIC: Bayesian Information Criterion
- DIC: Deviance Information Criterion

How Do We Select A Model?

Analytic statistics for selection, include penalty for complexity

- AIC : Akaka Information Criterion
- BIC: Bayesian Information Criterion
- DIC: Deviance Information Criterion

Can work well, but...

How Do We Select A Model?

Analytic statistics for selection, include penalty for complexity

- AIC : Akaka Information Criterion
- BIC: Bayesian Information Criterion
- DIC: Deviance Information Criterion

Can work well, but...

- Rely on specific loss function

How Do We Select A Model?

Analytic statistics for selection, include penalty for complexity

- AIC : Akaka Information Criterion
- BIC: Bayesian Information Criterion
- DIC: Deviance Information Criterion

Can work well, but...

- Rely on specific loss function
- Rely on asymptotic argument

How Do We Select A Model?

Analytic statistics for selection, include penalty for complexity

- AIC : Akaka Information Criterion
- BIC: Bayesian Information Criterion
- DIC: Deviance Information Criterion

Can work well, but...

- Rely on specific loss function
- Rely on asymptotic argument
- Rely on estimate of number of parameters

How Do We Select A Model?

Analytic statistics for selection, include penalty for complexity

- AIC : Akaka Information Criterion
- BIC: Bayesian Information Criterion
- DIC: Deviance Information Criterion

Can work well, but...

- Rely on specific loss function
- Rely on asymptotic argument
- Rely on estimate of number of parameters
- Extremely model dependent

How Do We Select A Model?

Analytic statistics for selection, include penalty for complexity

- AIC : Akaka Information Criterion
- BIC: Bayesian Information Criterion
- DIC: Deviance Information Criterion

Can work well, but...

- Rely on specific loss function
- Rely on asymptotic argument
- Rely on estimate of number of parameters
- Extremely model dependent

Need: general tool for evaluating models, replicates decision problem

Cross-Validation: Some Intuition

Recall Optimal division of data:

Cross-Validation: Some Intuition

Recall Optimal division of data:

- Train: build model

Cross-Validation: Some Intuition

Recall Optimal division of data:

- Train: build model
- Validation: assess model

Cross-Validation: Some Intuition

Recall Optimal division of data:

- Train: build model
- Validation: assess model
- Test: classify remaining documents

Cross-Validation: Some Intuition

Recall Optimal division of data:

- Train: build model
- Validation: assess model
- Test: classify remaining documents

K-fold Cross-validation idea: create many training and test sets.

Cross-Validation: Some Intuition

Recall Optimal division of data:

- Train: build model
- Validation: assess model
- Test: classify remaining documents

K-fold Cross-validation idea: create many training and test sets.

- Perform on labeled data

Cross-Validation: Some Intuition

Recall Optimal division of data:

- Train: build model
- Validation: assess model
- Test: classify remaining documents

K-fold Cross-validation idea: create many training and test sets.

- Perform on labeled data

  Suppose each document i belongs to class c_i.

Cross-Validation: Some Intuition

Recall Optimal division of data:

- Train: build model
- Validation: assess model
- Test: classify remaining documents

K-fold Cross-validation idea: create many training and test sets.

- Perform on labeled data

  Suppose each document i belongs to class c_i.

  Let $\mathbf{c} = (c_1, c_2, \ldots, c_N)$

Cross-Validation: Some Intuition

Recall Optimal division of data:

- Train: build model
- Validation: assess model
- Test: classify remaining documents

K-fold Cross-validation idea: create many training and test sets.

- Perform on labeled data

  Suppose each document i belongs to class c_i.

  Let $\mathbf{c} = (c_1, c_2, \ldots, c_N)$
- Idea: use observations both in training and test sets

Cross-Validation: Some Intuition

Recall Optimal division of data:

- Train: build model
- Validation: assess model
- Test: classify remaining documents

K-fold Cross-validation idea: create many training and test sets.

- Perform on labeled data

  Suppose each document i belongs to class c_i.

  Let $\mathbf{c} = (c_1, c_2, \ldots, c_N)$
- Idea: use observations both in training and test sets
- Each step: use held out data to evaluate performance

Cross-Validation: Some Intuition

Recall Optimal division of data:

- Train: build model
- Validation: assess model
- Test: classify remaining documents

K-fold Cross-validation idea: create many training and test sets.

- Perform on labeled data

  Suppose each document i belongs to class c_i.

  Let $\mathbf{c} = (c_1, c_2, \ldots, c_N)$
- Idea: use observations both in training and test sets
- Each step: use held out data to evaluate performance
- Avoid overfitting

Cross-Validation: A How To Guide

Process:

Cross-Validation: A How To Guide

Process:

- Randomly partition data into K groups.

Cross-Validation: A How To Guide

Process:

- Randomly partition data into K groups.
  (Group 1, Group 2, Group3, ..., Group K)

Cross-Validation: A How To Guide

Process:

- Randomly partition data into K groups.
  (Group 1, Group 2, Group3, . . ., Group K)
- Rotate through groups as follows

Cross-Validation: A How To Guide

Process:

- Randomly partition data into K groups.
  (Group 1, Group 2, Group3, . . ., Group K)
- Rotate through groups as follows

Step	Training	Validation ("Test")

Cross-Validation: A How To Guide

Process:

- Randomly partition data into K groups.
  (Group 1, Group 2, Group3, ..., Group K)
- Rotate through groups as follows

Step	Training	Validation ("Test")
1	Group2, Group3, Group 4, ..., Group K	Group 1

Cross-Validation: A How To Guide

Process:

- Randomly partition data into K groups.
  (Group 1, Group 2, Group3, ..., Group K)
- Rotate through groups as follows

Step	Training	Validation ("Test")
1	Group2, Group3, Group 4, ..., Group K	Group 1
2	Group 1, Group3, Group 4, ..., Group K	Group 2

Cross-Validation: A How To Guide

Process:

- Randomly partition data into K groups.
  (Group 1, Group 2, Group3, ..., Group K)
- Rotate through groups as follows

Step	Training	Validation ("Test")
1	Group2, Group3, Group 4, ..., Group K	Group 1
2	Group 1, Group3, Group 4, ..., Group K	Group 2
3	Group 1, Group 2, Group 4, ..., Group K	Group 3

Cross-Validation: A How To Guide

Process:

- Randomly partition data into K groups.
  (Group 1, Group 2, Group3, ..., Group K)
- Rotate through groups as follows

Step	Training	Validation ("Test")
1	Group2, Group3, Group 4, ..., Group K	Group 1
2	Group 1, Group3, Group 4, ..., Group K	Group 2
3	Group 1, Group 2, Group 4, ..., Group K	Group 3
⋮	⋮	⋮

Cross-Validation: A How To Guide

Process:

- Randomly partition data into K groups.
  (Group 1, Group 2, Group3, ..., Group K)
- Rotate through groups as follows

Step	Training	Validation ("Test")
1	Group2, Group3, Group 4, ..., Group K	Group 1
2	Group 1, Group3, Group 4, ..., Group K	Group 2
3	Group 1, Group 2, Group 4, ..., Group K	Group 3
⋮	⋮	⋮
K	Group 1, Group 2, Group 3, ..., Group K - 1	Group K

Cross-Validation: A How To Guide

Step	Training	Validation ("Test")
1	Group2, Group3, Group 4, ..., Group K	Group 1
2	Group 1, Group3, Group 4, ..., Group K	Group 2
3	Group 1, Group 2, Group 4, ..., Group K	Group 3
$\vdots$	$\vdots$	$\vdots$
K	Group 1, Group 2, Group 3, ..., Group K - 1	Group K

Cross-Validation: A How To Guide

Step	Training	Validation ("Test")
1	Group2, Group3, Group 4, ..., Group K	Group 1
2	Group 1, Group3, Group 4, ..., Group K	Group 2
3	Group 1, Group 2, Group 4, ..., Group K	Group 3
⋮	⋮	⋮
K	Group 1, Group 2, Group 3, ..., Group K - 1	Group K

Strategy:

Cross-Validation: A How To Guide

Step	Training	Validation ("Test")
1	Group2, Group3, Group 4, ..., Group K	Group 1
2	Group 1, Group3, Group 4, ..., Group K	Group 2
3	Group 1, Group 2, Group 4, ..., Group K	Group 3
⋮	⋮	⋮
K	Group 1, Group 2, Group 3, ..., Group K - 1	Group K

Strategy:

- Divide data into K groups

Cross-Validation: A How To Guide

Step	Training	Validation ("Test")
1	Group2, Group3, Group 4, ..., Group K	Group 1
2	Group 1, Group3, Group 4, ..., Group K	Group 2
3	Group 1, Group 2, Group 4, ..., Group K	Group 3
$\vdots$	$\vdots$	$\vdots$
K	Group 1, Group 2, Group 3, ..., Group K - 1	Group K

Strategy:

- Divide data into K groups
- Train data on $K - 1$ groups. Create function $f : \mathbf{Y} \rightarrow \mathbf{C}$

Cross-Validation: A How To Guide

Step	Training	Validation ("Test")
1	Group2, Group3, Group 4, ..., Group K	Group 1
2	Group 1, Group3, Group 4, ..., Group K	Group 2
3	Group 1, Group 2, Group 4, ..., Group K	Group 3
⋮	⋮	⋮
K	Group 1, Group 2, Group 3, ..., Group K - 1	Group K

Strategy:

- Divide data into K groups
- Train data on $K-1$ groups. Create function $f : \mathbf{Y} \rightarrow \mathbf{C}$
- Predict values for K^{th}

Cross-Validation: A How To Guide

Step	Training	Validation ("Test")
1	Group2, Group3, Group 4, ..., Group K	Group 1
2	Group 1, Group3, Group 4, ..., Group K	Group 2
3	Group 1, Group 2, Group 4, ..., Group K	Group 3
⋮	⋮	⋮
K	Group 1, Group 2, Group 3, ..., Group K - 1	Group K

Strategy:

- Divide data into K groups
- Train data on $K - 1$ groups. Create function $f : \mathbf{Y} \rightarrow \mathbf{C}$
- Predict values for K^{th}
- Summarize performance with loss function: $L(\mathbf{C}_i, f^{-k}(\mathbf{y}_i))$

Cross-Validation: A How To Guide

Step	Training	Validation ("Test")
1	Group2, Group3, Group 4, ..., Group K	Group 1
2	Group 1, Group3, Group 4, ..., Group K	Group 2
3	Group 1, Group 2, Group 4, ..., Group K	Group 3
⋮	⋮	⋮
K	Group 1, Group 2, Group 3, ..., Group K - 1	Group K

Strategy:

- Divide data into K groups
- Train data on $K-1$ groups. Create function $f : \mathbf{Y} \rightarrow \mathbf{C}$
- Predict values for K^{th}
- Summarize performance with loss function: $L(\mathbf{C}_i, f^{-k}(\mathbf{y}_i))$
  - Mean square error, Absolute error, Prediction error, ...

Cross-Validation: A How To Guide

Step	Training	Validation ("Test")
1	Group2, Group3, Group 4, ..., Group K	Group 1
2	Group 1, Group3, Group 4, ..., Group K	Group 2
3	Group 1, Group 2, Group 4, ..., Group K	Group 3
$\vdots$	$\vdots$	$\vdots$
K	Group 1, Group 2, Group 3, ..., Group K - 1	Group K

Strategy:

- Divide data into K groups
- Train data on $K-1$ groups. Create function $f : \mathbf{Y} \rightarrow \mathbf{C}$
- Predict values for K^{th}
- Summarize performance with loss function: $L(\mathbf{C}_i, f^{-k}(\mathbf{y}_i))$
    - Mean square error, Absolute error, Prediction error, ...

$$\text{CV(ind. classification)} = \frac{1}{N}\sum_{i=1}^{N} L(\mathbf{C}_i, f^{-k}(\mathbf{y}_i))$$

Cross-Validation: A How To Guide

Step	Training	Validation ("Test")
1	Group2, Group3, Group 4, ..., Group K	Group 1
2	Group 1, Group3, Group 4, ..., Group K	Group 2
3	Group 1, Group 2, Group 4, ..., Group K	Group 3
$\vdots$	$\vdots$	$\vdots$
K	Group 1, Group 2, Group 3, ..., Group K - 1	Group K

Strategy:

- Divide data into K groups
- Train data on $K-1$ groups. Create function $f : \mathbf{Y} \rightarrow \mathbf{C}$
- Predict values for K^{th}
- Summarize performance with loss function: $L(\mathbf{C}_i, f^{-k}(\mathbf{y}_i))$
    - Mean square error, Absolute error, Prediction error, ...

CV(ind. classification) $= \frac{1}{N}\sum_{i=1}^{N} L(\mathbf{C}_i, f^{-k}(\mathbf{y}_i))$

CV(proportions) $=$
$\frac{1}{K}\sum_{j=1}^{K}$ Mean Square Error Proportions from Group j

Cross-Validation: A How To Guide

Step	Training	Validation ("Test")
1	Group2, Group3, Group 4, ..., Group K	Group 1
2	Group 1, Group3, Group 4, ..., Group K	Group 2
3	Group 1, Group 2, Group 4, ..., Group K	Group 3
$\vdots$	$\vdots$	$\vdots$
K	Group 1, Group 2, Group 3, ..., Group K - 1	Group K

Strategy:

- Divide data into K groups
- Train data on $K-1$ groups. Create function $f : \mathbf{Y} \rightarrow \mathbf{C}$
- Predict values for K^{th}
- Summarize performance with loss function: $L(\mathbf{C}_i, f^{-k}(\mathbf{y}_i))$
  - Mean square error, Absolute error, Prediction error, ...

  CV(ind. classification) $= \frac{1}{N}\sum_{i=1}^{N} L(\mathbf{C}_i, f^{-k}(\mathbf{y}_i))$

  CV(proportions) $=$
  $\frac{1}{K}\sum_{j=1}^{K}$ Mean Square Error Proportions from Group j
- Final choice: model with highest CV score

How Do We Select K? (HTF, Section 7.10)

Common values of K

- $K = 5$: Five fold cross validation
- $K = 10$: Ten fold cross validation
- $K = N$: Leave one out cross validation

Considerations:

- How sensitive are inferences to number of coded documents? (HTF, pg 243-244)
- 200 labeled documents
    - $K = N \rightarrow 199$ documents to train,
    - $K = 10 \rightarrow 180$ documents to train
    - $K = 5 \rightarrow 160$ documents to train
- 50 labeled documents
    - $K = N \rightarrow 49$ documents to train,
    - $K = 10 \rightarrow 45$ documents to train
    - $K = 5 \rightarrow 40$ documents to train
- How long will it take to run models?
    - $K-$fold cross validation requires $K \times$ One model run
- What is the correct loss function?

If you cross validate, you really need to cross validate

From Section 7.10.2 of HTF

- Use CV to estimate prediction error
- All supervised steps performed in cross-validation
- Underestimate prediction error
- Could lead to selecting lower performing model

Cross Validation In `R`

`library(bootstrap)` Contains a cross validation (bootstrap and Jackknife function as well)

Ensemble Learning

Many methods for classification

Ensemble Learning

Many methods for classification

- SVM (linear, Gaussian, ...)

Ensemble Learning

Many methods for classification

- SVM (linear, Gaussian, ...)
- Naive Bayes

Ensemble Learning

Many methods for classification

- SVM (linear, Gaussian, ...)
- Naive Bayes
- Max-Entropy

Ensemble Learning

Many methods for classification

- SVM (linear, Gaussian, ...)
- Naive Bayes
- Max-Entropy
- Lasso/Ridge

Ensemble Learning

Many methods for classification

- SVM (linear, Gaussian, ...)
- Naive Bayes
- Max-Entropy
- Lasso/Ridge
- Random Forests

Ensemble Learning

Many methods for classification

- SVM (linear, Gaussian, ...)
- Naive Bayes
- Max-Entropy
- Lasso/Ridge
- Random Forests
- ...

Ensemble Learning

Many methods for classification

- SVM (linear, Gaussian, ...)
- Naive Bayes
- Max-Entropy
- Lasso/Ridge
- Random Forests
- ...

Cross-validation: selection of one model

Ensemble Learning

Many methods for classification

- SVM (linear, Gaussian, ...)
- Naive Bayes
- Max-Entropy
- Lasso/Ridge
- Random Forests
- ...

Cross-validation: selection of one model

- Oracle property: selects best model for underlying data [this is amazing]

Ensemble Learning

Many methods for classification

- SVM (linear, Gaussian, ...)
- Naive Bayes
- Max-Entropy
- Lasso/Ridge
- Random Forests
- ...

Cross-validation: selection of one model

- Oracle property: selects best model for underlying data [this is amazing]
- But what do we do with the other models we fit?

Ensemble Learning

Many methods for classification

- SVM (linear, Gaussian, ...)
- Naive Bayes
- Max-Entropy
- Lasso/Ridge
- Random Forests
- ...

Cross-validation: selection of one model

- Oracle property: selects best model for underlying data [this is amazing]
- But what do we do with the other models we fit?
- Ensemble methods: combine learners to improve model fit

Ensemble Learning

Many methods for classification

- SVM (linear, Gaussian, ...)
- Naive Bayes
- Max-Entropy
- Lasso/Ridge
- Random Forests
- ...

Cross-validation: selection of one model

- Oracle property: selects best model for underlying data [this is amazing]
- But what do we do with the other models we fit?
- Ensemble methods: combine learners to improve model fit
- Simplest form: methods vote on category, majority wins

Ensemble Learning: Intuition

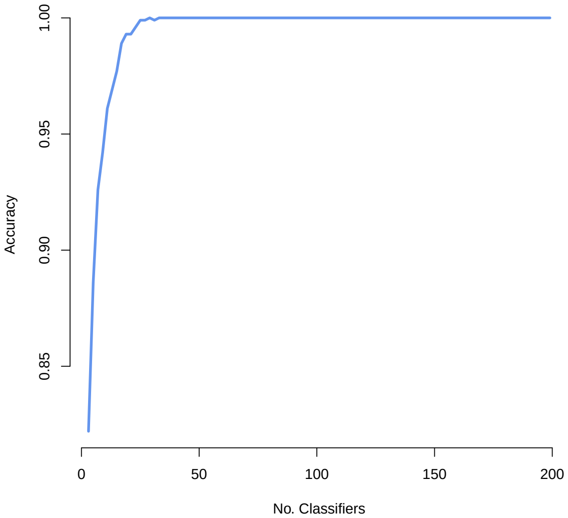
Heuristic: if classifiers are accurate and diverse $\rightarrow$ ensemble methods improve

Intuition:

- Classify documents 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

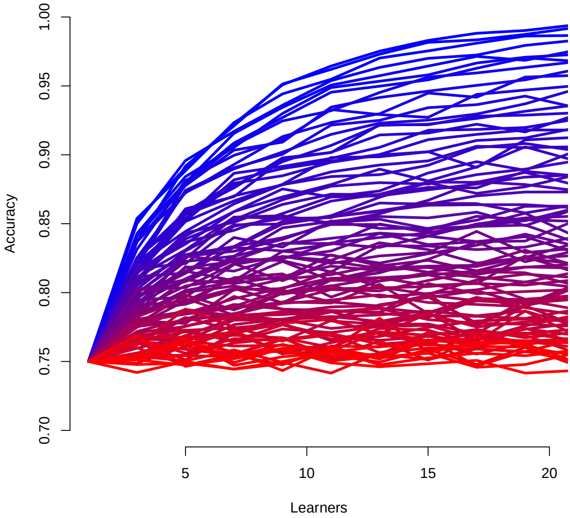
$$\begin{aligned}
\Pr(\text{Correct Guess}|\text{Votes}) &= \Pr(3 \text{ correct}) + \Pr(2 \text{ correct}) \\
&= 0.75^3 + 3 \times (0.75^2 \times 0.25) \\
&= 0.844
\end{aligned}$$

Ensemble Learning: Intuition



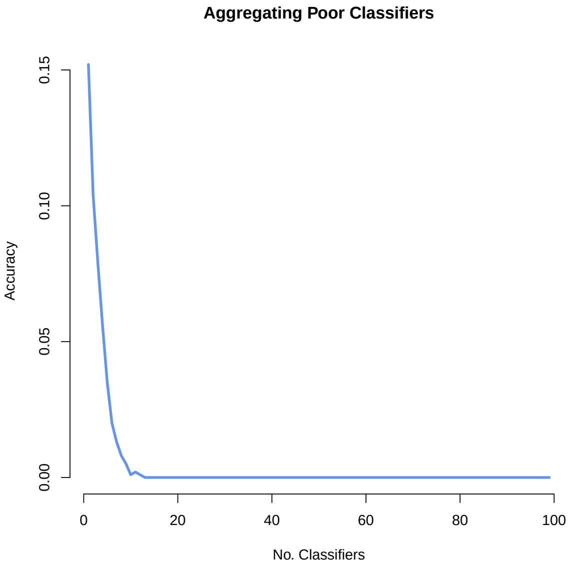
Ensemble Learning: Intuition

Diverse and Accurate matter.



Ensemble Learning: Intuition

Diverse and Accurate matter.



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

Common Ensemble Methods

Committee Methods

- Voting (classification)
- Averaging (predictions)

Bagging: bootstrap aggregation

- Need method to produce variability between models in data set
- Bootstrap M data sets (draw N observations, with replacement)
- Apply classifier to each data set
- Aggregation across classified data sets
- Dietterich 2000: works well for unstable classifiers (lots of diversity across samples)

Common Ensemble Methods

Boosting: sequential training of weak classifiers

- Method for combining several weak classifiers
- Basic idea:
    - Model 1: classify initially based on all data (equal weight)
    - Model 2: classify all data, more weight to incorrectly classified data
    - Model 3: classify all data, more weight to incorrectly classified data

      ...
    - Model M: classify all data, more weight to incorrectly classified data
- Aggregate using weighted committee

Ensembles in R

`ADABoost` :

`Bagging:` :

Post code to Piazzza (from Solomon)

Super learning

van der Laan, et. al (2007): Develop a cross-validation heavy method for aggregating classifiers
Best name in statistics?
Notation we'll need:

$$\begin{aligned}
\mathbf{y}_i &= \text{M x 1 vector of data} \\
C_i &= \text{Category for observation } i \text{ (need pre-labeled data)} \\
M &= \text{Number of methods included in ensemble} \\
\mathbf{Z}_i &= (Z_{i1}, Z_{i2}, Z_{i3}, \ldots, Z_{iM}) \\
&= \text{Predictions for } i \text{ across } M \text{ methods} \\
K &= \text{Number of Folds in Cross Validation}
\end{aligned}$$

Super Learning Algorithm

Training:

Super Learning Algorithm

Training:

1) Split data into K blocks (K-fold cross validation)

Super Learning Algorithm

Training:

1) Split data into K blocks (K-fold cross validation)
2) Fit M models on training data K times (K-fold cross validation)

Super Learning Algorithm

Training:

1) Split data into K blocks (K-fold cross validation)
2) Fit M models on training data K times (K-fold cross validation)
    - Block 1: train on Blocks 2 to K

Super Learning Algorithm

Training:

1) Split data into K blocks (K-fold cross validation)
2) Fit M models on training data K times (K-fold cross validation)
    - Block 1: train on Blocks 2 to K
    - Block 2: train on Blocks 1, 3 to K, ...

Super Learning Algorithm

Training:

1) Split data into K blocks (K-fold cross validation)
2) Fit M models on training data K times (K-fold cross validation)
   - Block 1: train on Blocks 2 to K
   - Block 2: train on Blocks 1, 3 to K, ...
   - Block K: train on Blocks 1 to $K - 1$

Super Learning Algorithm

Training:

1) Split data into K blocks (K-fold cross validation)
2) Fit M models on training data K times (K-fold cross validation)
    - Block 1: train on Blocks 2 to K
    - Block 2: train on Blocks 1, 3 to K, ...
    - Block K: train on Blocks 1 to $K-1$
3) For each of K blocks, generate M predictions for each observation

Super Learning Algorithm

Training:

1) Split data into K blocks (K-fold cross validation)
2) Fit M models on training data K times (K-fold cross validation)
   - Block 1: train on Blocks 2 to K
   - Block 2: train on Blocks 1, 3 to K, ...
   - Block K: train on Blocks 1 to $K-1$
3) For each of K blocks, generate M predictions for each observation
   - Block 1: make predictions for observations in Block 1 using models

Super Learning Algorithm

Training:

1) Split data into K blocks (K-fold cross validation)
2) Fit M models on training data K times (K-fold cross validation)
   - Block 1: train on Blocks 2 to K
   - Block 2: train on Blocks 1, 3 to K, ...
   - Block K: train on Blocks 1 to $K-1$
3) For each of K blocks, generate M predictions for each observation
   - Block 1: make predictions for observations in Block 1 using models
   - Carry out for all blocks, produce vector of predictions

Super Learning Algorithm

Training:

1) Split data into K blocks (K-fold cross validation)
2) Fit M models on training data K times (K-fold cross validation)
    - Block 1: train on Blocks 2 to K
    - Block 2: train on Blocks 1, 3 to K, ...
    - Block K: train on Blocks 1 to $K-1$
3) For each of K blocks, generate M predictions for each observation
    - Block 1: make predictions for observations in Block 1 using models
    - Carry out for all blocks, produce vector of predictions
        - $\mathbf{Z}_i = (Z_{i1}, Z_{i2}, Z_{i3}, \ldots, Z_{iM})$

Super Learning Algorithm

Training:

1) Split data into K blocks (K-fold cross validation)
2) Fit M models on training data K times (K-fold cross validation)
    - Block 1: train on Blocks 2 to K
    - Block 2: train on Blocks 1, 3 to K, ...
    - Block K: train on Blocks 1 to $K-1$
3) For each of K blocks, generate M predictions for each observation
    - Block 1: make predictions for observations in Block 1 using models
    - Carry out for all blocks, produce vector of predictions
        - $\mathbf{Z}_i = (Z_{i1}, Z_{i2}, Z_{i3}, \ldots, Z_{iM})$
    - New data: $(C_i, \mathbf{Z}_i)$

Super Learning Algorithm

Training:

1) Split data into K blocks (K-fold cross validation)
2) Fit M models on training data K times (K-fold cross validation)
    - Block 1: train on Blocks 2 to K
    - Block 2: train on Blocks 1, 3 to K, ...
    - Block K: train on Blocks 1 to $K-1$
3) For each of K blocks, generate M predictions for each observation
    - Block 1: make predictions for observations in Block 1 using models
    - Carry out for all blocks, produce vector of predictions
        - $\mathbf{Z}_i = (Z_{i1}, Z_{i2}, Z_{i3}, \ldots, Z_{iM})$
    - New data: $(C_i, \mathbf{Z}_i)$
    - C_i: dependent variable

Super Learning Algorithm

Training:

1) Split data into K blocks (K-fold cross validation)
2) Fit M models on training data K times (K-fold cross validation)
   - Block 1: train on Blocks 2 to K
   - Block 2: train on Blocks 1, 3 to K, ...
   - Block K: train on Blocks 1 to $K-1$
3) For each of K blocks, generate M predictions for each observation
   - Block 1: make predictions for observations in Block 1 using models
   - Carry out for all blocks, produce vector of predictions
     - $\mathbf{Z}_i = (Z_{i1}, Z_{i2}, Z_{i3}, \ldots, Z_{iM})$
   - New data: $(C_i, \mathbf{Z}_i)$
   - C_i: dependent variable
   - $\mathbf{Z}_i$: $M \times 1$ vector of predictions (covariates)?

Super Learning Algorithm

Training:

1) Split data into K blocks (K-fold cross validation)
2) Fit M models on training data K times (K-fold cross validation)
    - Block 1: train on Blocks 2 to K
    - Block 2: train on Blocks 1, 3 to K, ...
    - Block K: train on Blocks 1 to $K-1$
3) For each of K blocks, generate M predictions for each observation
    - Block 1: make predictions for observations in Block 1 using models
    - Carry out for all blocks, produce vector of predictions
        - $\mathbf{Z}_i = (Z_{i1}, Z_{i2}, Z_{i3}, \ldots, Z_{iM})$
    - New data: $(C_i, \mathbf{Z}_i)$
    - C_i: dependent variable
    - $\mathbf{Z}_i$: $M \times 1$ vector of predictions (covariates)?
4) Regress $C_i \sim \mathbf{Z}_i$

Super Learning Algorithm

Training:

1) Split data into K blocks (K-fold cross validation)
2) Fit M models on training data K times (K-fold cross validation)
    - Block 1: train on Blocks 2 to K
    - Block 2: train on Blocks 1, 3 to K, ...
    - Block K: train on Blocks 1 to $K-1$
3) For each of K blocks, generate M predictions for each observation
    - Block 1: make predictions for observations in Block 1 using models
    - Carry out for all blocks, produce vector of predictions
        - $\mathbf{Z}_i = (Z_{i1}, Z_{i2}, Z_{i3}, \ldots, Z_{iM})$
    - New data: $(C_i, \mathbf{Z}_i)$
    - C_i: dependent variable
    - $\mathbf{Z}_i$: $M \times 1$ vector of predictions (covariates)?
4) Regress $C_i \sim \mathbf{Z}_i$
    - Use any previous methods

Super Learning Algorithm

Training:

1) Split data into K blocks (K-fold cross validation)
2) Fit M models on training data K times (K-fold cross validation)
    - Block 1: train on Blocks 2 to K
    - Block 2: train on Blocks 1, 3 to K, ...
    - Block K: train on Blocks 1 to $K-1$
3) For each of K blocks, generate M predictions for each observation
    - Block 1: make predictions for observations in Block 1 using models
    - Carry out for all blocks, produce vector of predictions
        - $\mathbf{Z}_i = (Z_{i1}, Z_{i2}, Z_{i3}, \ldots, Z_{iM})$
    - New data: $(C_i, \mathbf{Z}_i)$
    - C_i: dependent variable
    - $\mathbf{Z}_i$: $M \times 1$ vector of predictions (covariates)?
4) Regress $C_i \sim \mathbf{Z}_i$
    - Use any previous methods
    - Linear Regression, Lasso, Ridge, ...

Super Learning Algorithm

Training:

1) Split data into K blocks (K-fold cross validation)
2) Fit M models on training data K times (K-fold cross validation)
    - Block 1: train on Blocks 2 to K
    - Block 2: train on Blocks 1, 3 to K, ...
    - Block K: train on Blocks 1 to $K - 1$
3) For each of K blocks, generate M predictions for each observation
    - Block 1: make predictions for observations in Block 1 using models
    - Carry out for all blocks, produce vector of predictions
        - $\mathbf{Z}_i = (Z_{i1}, Z_{i2}, Z_{i3}, \ldots, Z_{iM})$
    - New data: $(C_i, \mathbf{Z}_i)$
    - C_i: dependent variable
    - $\mathbf{Z}_i$: $M \times 1$ vector of predictions (covariates)?
4) Regress $C_i \sim \mathbf{Z}_i$
    - Use any previous methods
    - Linear Regression, Lasso, Ridge, ...
    - Produce function that maps from $\mathbf{Z}_i$ to classes C_i

Super Learning Algorithm

Training:

1) Split data into K blocks (K-fold cross validation)
2) Fit M models on training data K times (K-fold cross validation)
    - Block 1: train on Blocks 2 to K
    - Block 2: train on Blocks 1, 3 to K, ...
    - Block K: train on Blocks 1 to $K-1$
3) For each of K blocks, generate M predictions for each observation
    - Block 1: make predictions for observations in Block 1 using models
    - Carry out for all blocks, produce vector of predictions
        - $\mathbf{Z}_i = (Z_{i1}, Z_{i2}, Z_{i3}, \ldots, Z_{iM})$
    - New data: $(C_i, \mathbf{Z}_i)$
    - C_i: dependent variable
    - $\mathbf{Z}_i$: $M \times 1$ vector of predictions (covariates)?
4) Regress $C_i \sim \mathbf{Z}_i$
    - Use any previous methods
    - Linear Regression, Lasso, Ridge, ...
    - Produce function that maps from $\mathbf{Z}_i$ to classes C_i
    - Example: $\frac{1}{1+\exp(-\boldsymbol{\beta}'\mathbf{Z}_i)}$

Super Learning Algorithm

Super Learning Algorithm

5) Fit all M models to entire data set, produce

Super Learning Algorithm

5) Fit all M models to entire data set, produce $\tilde{\mathbf{Z}}_i = (\tilde{Z}_{i1}, \tilde{Z}_{i2}, \ldots, \tilde{Z}_{iM})$

Super Learning Algorithm

5) Fit all M models to entire data set, produce $\tilde{\mathbf{Z}}_i = (\tilde{Z}_{i1}, \tilde{Z}_{i2}, \ldots, \tilde{Z}_{iM})$
6) Use function from Step 4 to produce classifications for all observations in training set

Super Learning Algorithm

5) Fit all M models to entire data set, produce $\tilde{\mathbf{Z}}_i = (\tilde{Z}_{i1}, \tilde{Z}_{i2}, \ldots, \tilde{Z}_{iM})$
6) Use function from Step 4 to produce classifications for all observations in training set
7) Evaluate super learner performance:

Super Learning Algorithm

5) Fit all M models to entire data set, produce $\tilde{\mathbf{Z}}_i = (\tilde{Z}_{i1}, \tilde{Z}_{i2}, \ldots, \tilde{Z}_{iM})$
6) Use function from Step 4 to produce classifications for all observations in training set
7) Evaluate super learner performance:
    - Built in method for assessing super learner's performance

Super Learning Algorithm

5) Fit all M models to entire data set, produce $\tilde{\mathbf{Z}}_i = (\tilde{Z}_{i1}, \tilde{Z}_{i2}, \ldots, \tilde{Z}_{iM})$
6) Use function from Step 4 to produce classifications for all observations in training set
7) Evaluate super learner performance:
   - Built in method for assessing super learner's performance

Test Set

Super Learning Algorithm

5) Fit all M models to entire data set, produce $\tilde{\mathbf{Z}}_i = (\tilde{Z}_{i1}, \tilde{Z}_{i2}, \ldots, \tilde{Z}_{iM})$
6) Use function from Step 4 to produce classifications for all observations in training set
7) Evaluate super learner performance:
    - Built in method for assessing super learner's performance

Test Set

- Use model from Step 5 to generate predictions for all data

Super Learning Algorithm

5) Fit all M models to entire data set, produce $\tilde{\mathbf{Z}}_i = (\tilde{Z}_{i1}, \tilde{Z}_{i2}, \ldots, \tilde{Z}_{iM})$
6) Use function from Step 4 to produce classifications for all observations in training set
7) Evaluate super learner performance:
    - Built in method for assessing super learner's performance

Test Set

- Use model from Step 5 to generate predictions for all data
- Use function from Step 4 to generate predictions for all observations

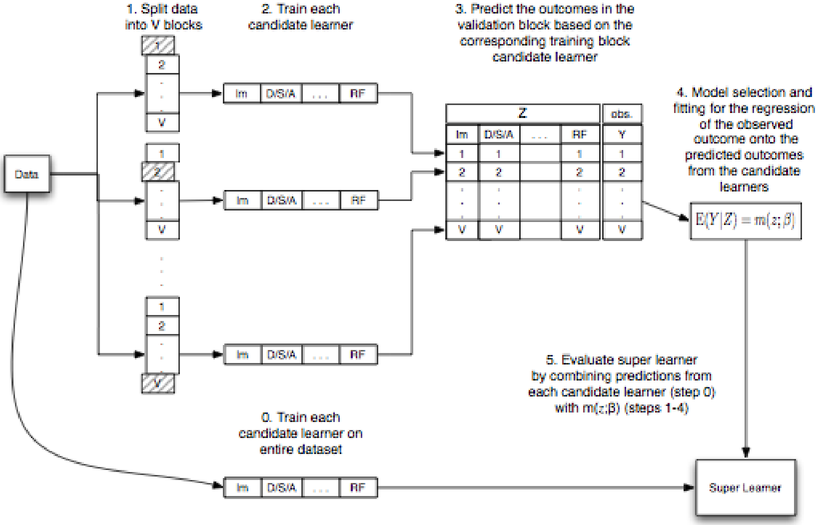


Figure 1: Flow Diagram for Super Learner

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

Super learner in R

`Superlearner()` is a package available in R (off of GitHub, not CRAN)
`https://github.com/ecpolley/SuperLearner`
Automatic selection of methods, prediction, and many other features
Code yourself:

- Perform cross validation
- Apply methods/get predictions
- Final regression
- Complicated but not technically hard (rely on canned programs throughout)

Scaling Political Text

Scaling:

- Spatial model of politics (median voters, proposal games, pivotal politics, veto players, bargaining)
- Retrieve space to test spatial theories of politics
- Stanford: spatial modeling center
- Space from votes: Poole and Rosenthal; Clinton, Jackman, Rivers;...
- Space from contributions: Wand; Bonica; ...
- Space from votes and survey responses : Bafumi and Herron; Lauderdale; Rodden and Warshaw; Tausanovitch and Warshaw...

Goal:

- Low level summary of actors' political beliefs
- Problem: often difficult to collect data

Wouldn't it be great: if we could use text to retrieve low level spatial locations?

- Existing Models ?
- How do we evaluate? (What is the goal when using text?) (What is ideology?)
- Prediction? Description? Summary? ...

Beauchamp (2011): summary of methods, simulations, and attempts to approximate roll call scalings

Scaling and US Congress: Roll Call Votes

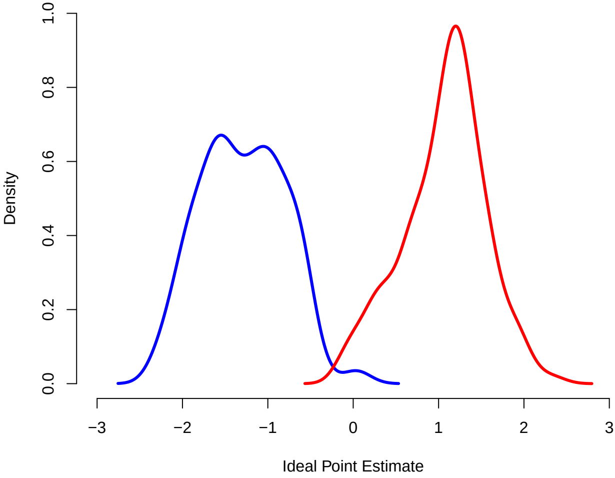
Poole and Rosenthal Scores:

- (Essentially): Factor analysis of roll call votes
- Simple (crazy!) procedure reveals highly informative ordering of legislators
- Highly predictive of Congressional voting/expected behavior, ...

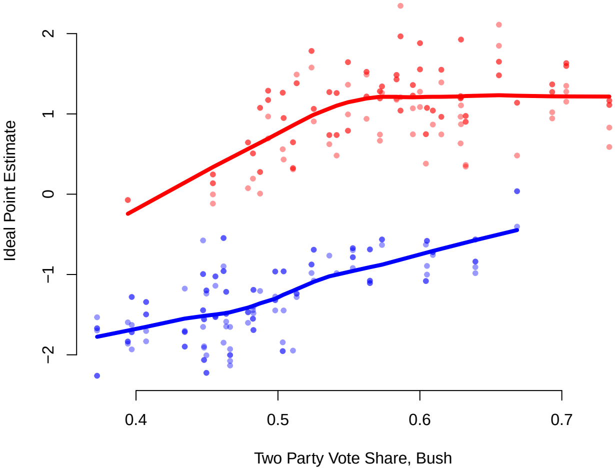
For a variety of reasons, low-dimensional summaries of roll call voting behavior is useful

Allows approximation of ideology with low-dimensional (unidimensional) summaries

Scaling and US Congress: Roll Call Votes



Scaling and US Congress: Roll Call Votes



Scaling and US Congress: Roll Call Votes

But this is not true in other settings

- Without votes, hard to scale other actors
- Wand; Bonica great results with campaign contributions
- But what if there are no campaign contributions?
- In highly disciplined parliaments, hard to scale in meaningful (something like ideological) way
  - Spirling and Maclean: Poole's OC algorithm yields scaling that deviates from qualitative expectations
  - Quinn and Spirling: standard methods of scaling group together rebels and conservatives

But everybody talks!

WordScores (Laver, Benoit, and Garry 2003)

Running example :

- Develop scaling of US Senate in 2005, based on press releases
- Monroe, Colaresi, and Quinn (2010) [conference presentation], Beauchamp (2011), Cormack (2011)
- First try: `wordscores`

Wordscores proceeds as follows:

- Identify set of reference texts
- Determine how well words separate reference texts (week 4 problem)
- Using this score, we assess new documents
- Generates scaling for all documents

WordScores (Laver, Benoit, and Garry 2003)

Suppose we have reference texts:

- Liberal: Ted Kennedy, L
    - Document: $\mathbf{y}_L$
    - Total words: $W_L = \sum_{m=1}^{M} y_{mL}$
- Conservative: Tom Coburn, C, $\mathbf{y}_C$
    - Document: $\mathbf{y}_C$
    - Total words: $W_C = \sum_{m=1}^{M} y_{mC}$

WordScores (Laver, Benoit, and Garry 2003)

Our first task: score each word

- How well does each word separate speakers?
- Lowe (2008) and Beauchamp (2011): approximately ask $p(L|y_{ij} = z)$
- Laver, Benoit, and Garry (2003) compute:

$$P_{zL} \equiv \frac{\frac{y_{zL}}{W_L}}{\frac{y_{zL}}{W_L} + \frac{y_{zC}}{W_C}}$$

The score for word z is then,

$$S_z = P_{zR} - P_{zL}$$

For all other documents, compute their scores:

$$\text{Score}_i = \sum_{m=1}^{M} \frac{y_{mi}}{W_i} S_z$$

Generalize to groups, multiple dimensions [not necessary for intuition, though]

Where have we seen this before?

Where have we seen this before?

Reference texts

Where have we seen this before?

Reference texts $\rightsquigarrow$ training set

Where have we seen this before?

Reference texts $\rightsquigarrow$ training set
Virgin texts

Where have we seen this before?

Reference texts $\rightsquigarrow$ training set
Virgin texts $\rightsquigarrow$ test set

Where have we seen this before?

Reference texts $\rightsquigarrow$ training set
Virgin texts $\rightsquigarrow$ test set
Goal: construct dictionary to score test set according to scale in training set

Where have we seen this before?

Reference texts $\rightsquigarrow$ training set

Virgin texts $\rightsquigarrow$ test set

Goal: construct dictionary to score test set according to scale in training set

This a dictionary method, with features selected from training set

Where have we seen this before?

Reference texts $\rightsquigarrow$ training set

Virgin texts $\rightsquigarrow$ test set

Goal: construct dictionary to score test set according to scale in training set

This a dictionary method, with features selected from training set

Or: this is isomorphic to method used in Week 3 and Federalist papers

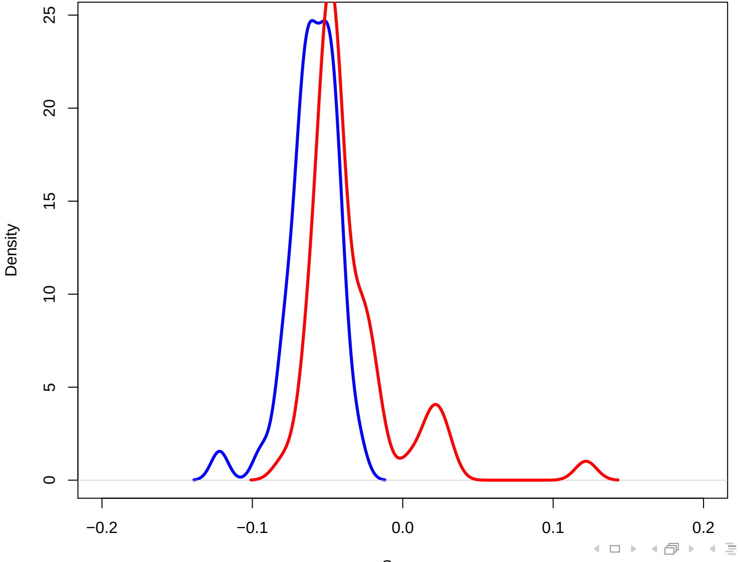
Wordscores in R

You know how to run this model! (Check Your Dictionary Homework!!)

- Generate dictionary weights using simple algorithm described above
- Score documents according to weights

Transformations (standardize results)

Applying to Senate Press Releases



Lowe (2008) & Beauchamp 2011

Beauchamp (2011): shows wordscores and Naive Bayes (where training set are reference documents) produce similar scalings:

Why?

Generalize:

- Take any week 4 method (includes all supervised learning algorithms that produce "coefficients")
- Create scores using these coefficients

Lowe (2008): Discusses potentially problematic wordscore properties

1) Each word is weighted equally (fixable with different scoring procedure)
2) Unique words are conflated with centrist (fixable with MCQ fightin' words style algorithm)
3) General problem: hard to interpret and no statistical model makes inference more difficutl

To be fair: fast, nonparametric, and novel [trailblazing] method for scoring documents (starts conversation)

WordFish

Monroe and Maeda (2005) and Slopkin and Proksch (2008):
Develop Item-Response style model for analyzing political texts.
Basic idea:

- Parties have underlying latent position
- This is associated with word usage
- Some words discriminate better than others
- Fit large model to estimate

For more on IRT:
Clinton, Jackman, Rivers (2003) : IRT for roll call votes
Rivers (2002): Identification for factor analysis models

WordFish

Suppose we have individual i. (We'll ignore temporal component for now).

$$
\begin{aligned}
y_{ij} &\sim \text{Poisson}(\lambda_{ij}) \\
\lambda_{ij} &= \exp(\alpha_i + \psi_j + \beta_j \times \theta_i)
\end{aligned}
$$

Where,

$$
\begin{aligned}
\lambda_{ij} &= \text{Rate individual } i \text{ uses word } j \\
\alpha_i &= \text{Individual } i \text{ loquaciousness} \\
\psi_j &= \text{Word } j\text{'s frequency} \\
\beta_j &= \text{Word } j\text{'s discrimination} \\
\theta_i &= \text{Legislator } i\text{'s latent positions}
\end{aligned}
$$

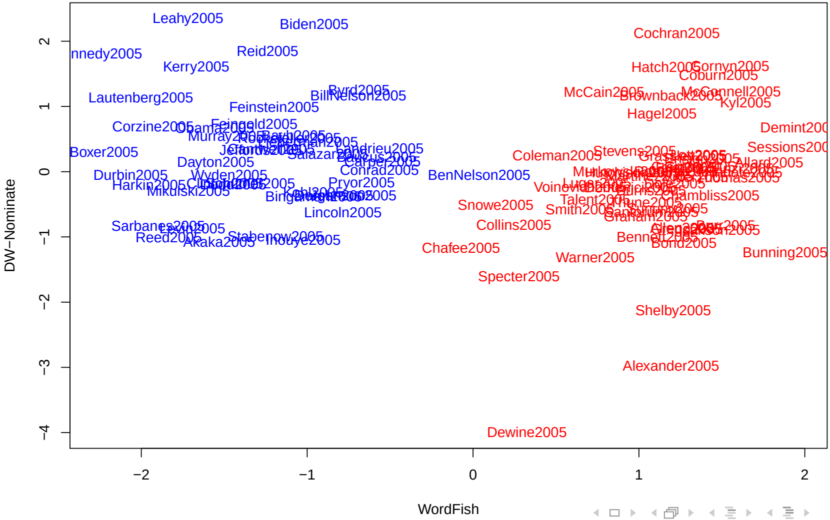
Benoit and Lowe (2010, 2011): Poisson functional form probably wrong

Running `WordFish` in R

Slapkin and Proksch have code available at :
`http://www.wordfish.org`
Apply simply to term document matrix.

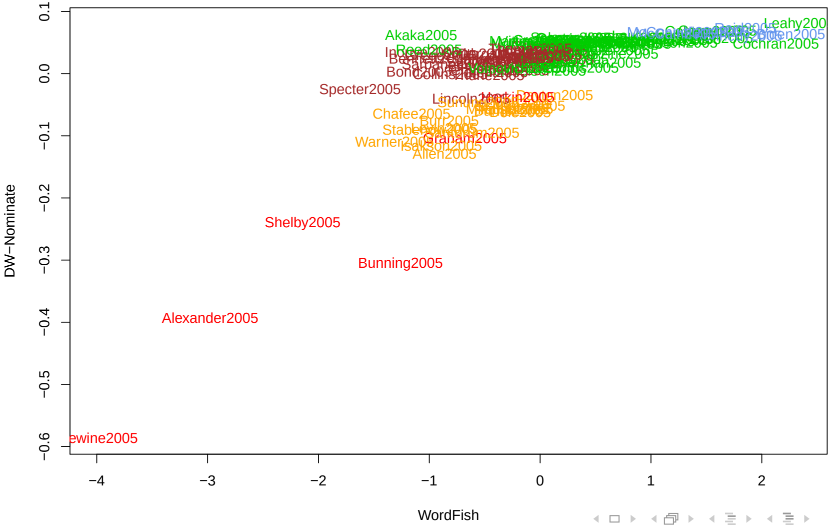
WordFish on Senate Press Releases

Fit model (using defaults).



WordFish on Senate Press Releases

Fit model (using defaults).



Scaling Wrap-up

Goal:

- What exactly do we want when we scale?
- Submit: just as ambiguous as clustering problem (perhaps more?)
- Without goal $\rightsquigarrow$ hard to validate, hard to make real progress

Problem:

- US Congress has been easy
- Text is harder
- Goal cannot be replication of voting scales
- Need more supervision (survey-like questions to classify texts)
- Makes clear immediately what we want: low-level summary of supervised components?

Where We've Been

- Class 1: Finding Text Data
- Class 2: Representing Texts Quantitatively
- Class 3: Dictionary Methods for Classification
- Class 4: Comparing Language Across Groups
- Class 5: Texts in Space
- Class 6: Clustering
- Class 7: Topic models
- Class 8: Supervised methods for classification
- Class 9: Ensemble methods for classification
- Class 10: Scaling Speech

Class Theme: Think!

Statistical/Algorithmic tools for text create new possibilities

Do not eliminate the need to THINK

When applying methods:

- Think: is this a useful model
- Can I accomplish my goal using a different tool?
- How do I validate my results

From here:

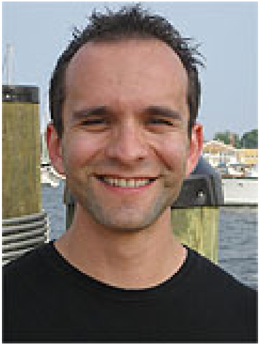
1) Natural Language Processing Courses
   - Part of speech tagging
   - Sentence parsing
   - ...
2) Machine Learning
   - Bayesian statistics
   - High dimensional data
   - ...

Thanks!

Thanks!

Jackie!

Thanks!



$< / \text{Course} >$