# POIR 613: Computational Social Science

**Pablo Barberá**

School of International Relations
University of Southern California
pablobarbera.com

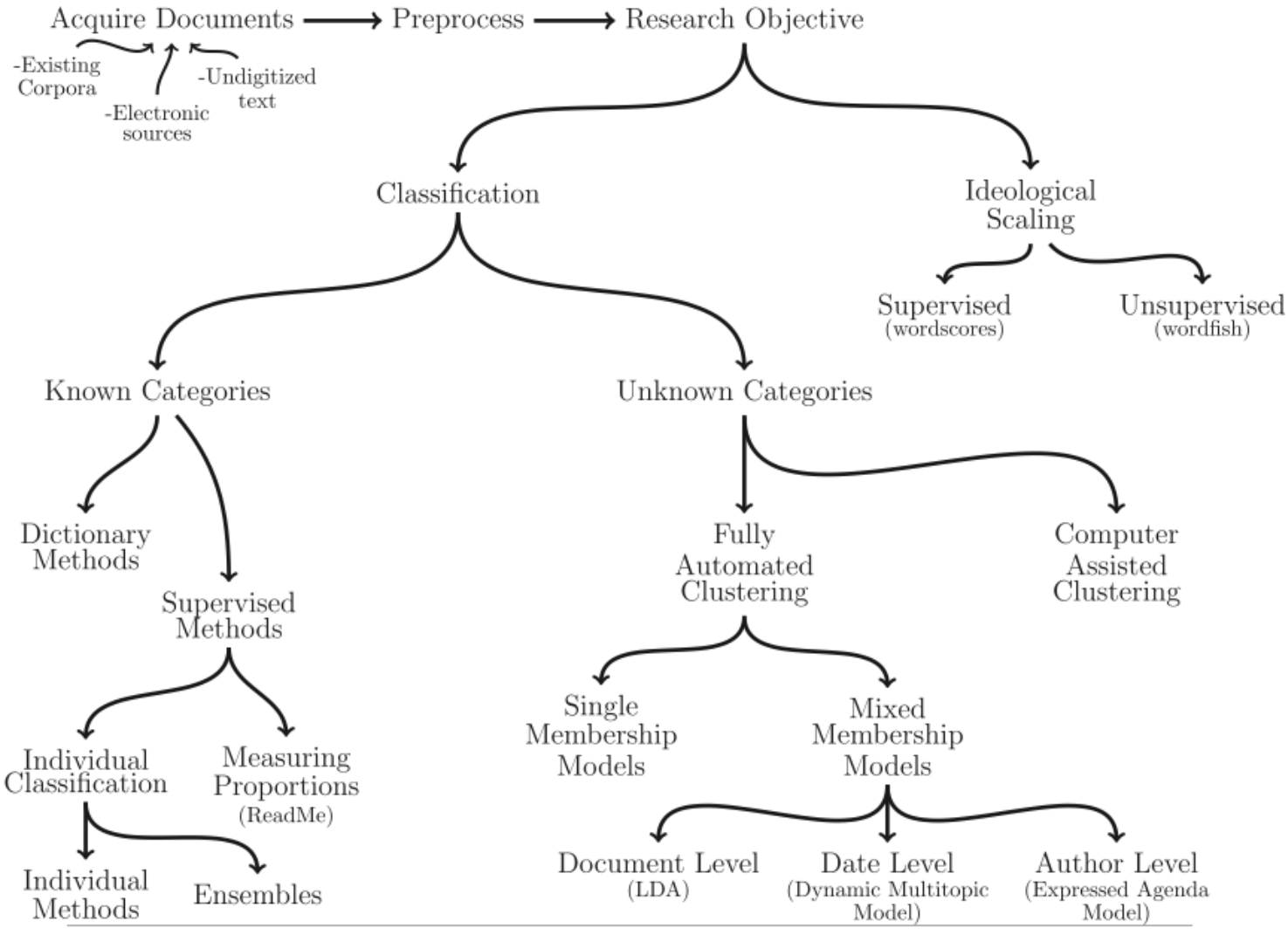
Course website:
pablobarbera.com/POIR613/

# Today

1. Project
   - Two-page summary was due on Monday
   - Peer feedback due next Monday
   - See my email for additional details
2. Machine learning
3. Solutions to challenge 5
4. Examples of supervised machine learning

# Supervised machine learning

# Overview of text as data methods



# Outline

- Supervised learning overview
- Creating a labeled set and evaluating its reliability
- Classifier performance metrics
- One classifier for text
  - Regularized regression

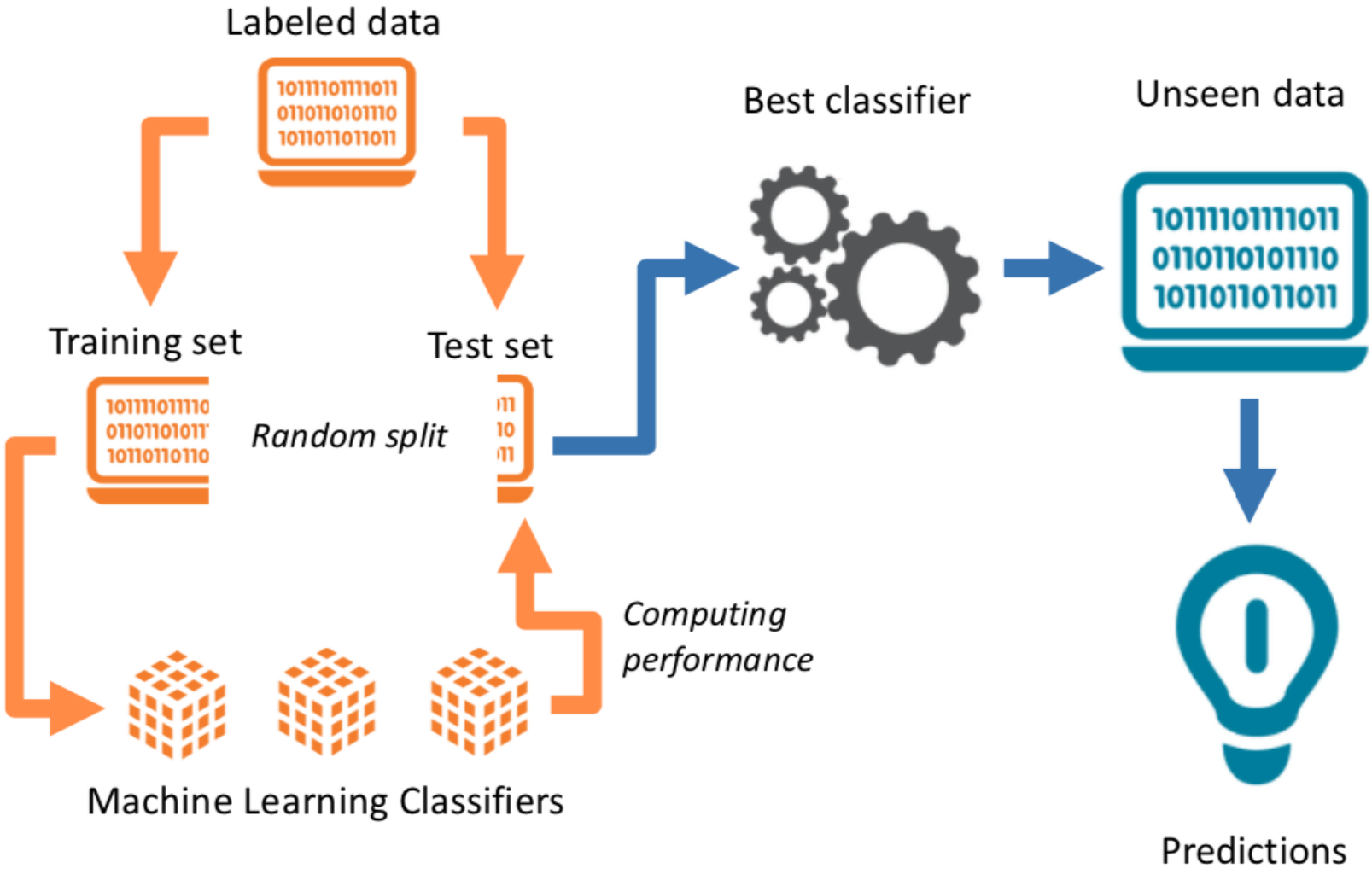
# Supervised machine learning

**Goal**: classify documents into pre existing categories.

e.g. authors of documents, sentiment of tweets, ideological position of parties based on manifestos, tone of movie reviews...

**What we need**:

- Hand-coded dataset (labeled), to be split into:
  - Training set: used to train the classifier
  - Validation/Test set: used to validate the classifier
- Method to extrapolate from hand coding to unlabeled documents (classifier):
  - Naive Bayes, regularized regression, SVM, K-nearest neighbors, BART, ensemble methods...
- Performance metric to choose best classifier and avoid overfitting: confusion matrix, accuracy, precision, recall...


Labeled data
Training set
Test set
Random split
Best classifier
Unseen data
Computing performance
Machine Learning Classifiers
Predictions

# Basic principles of supervised learning

- Generalization: A classifier or a regression algorithm learns to correctly predict output from given inputs not only in previously seen samples but also in previously unseen samples
- Overfitting: A classifier or a regression algorithm learns to correctly predict output from given inputs in previously seen samples but fails to do so in previously unseen samples. This causes poor prediction/generalization.

# Supervised v. unsupervised methods compared

- The goal (in text analysis) is to differentiate *documents* from one another, treating them as "bags of words"
- Different approaches:
  - *Supervised methods* require a training set that exemplify contrasting classes, identified by the researcher
  - *Unsupervised methods* scale documents based on patterns of similarity from the term-document matrix, without requiring a training step
- Relative advantage of supervised methods:

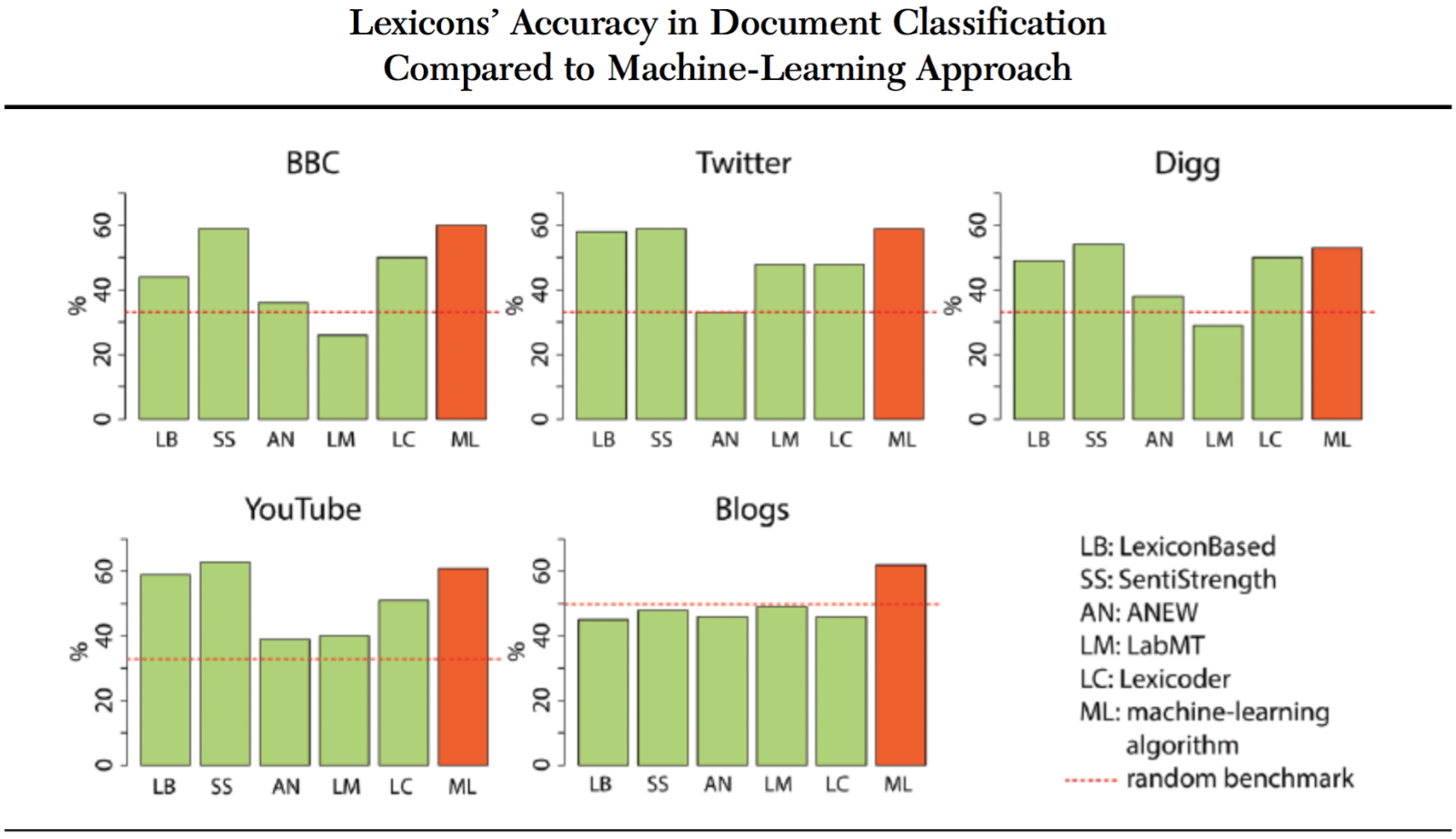
  You already know the dimension being scaled, because you set it in the training stage
- Relative disadvantage of supervised methods:

  You *must* already know the dimension being scaled, because you have to feed it good sample documents in the training stage

# Supervised learning v. dictionary methods

- Dictionary methods:
  - Advantage: not corpus-specific, cost to apply to a new corpus is trivial
  - Disadvantage: not corpus-specific, so performance on a new corpus is unknown (domain shift)
- Supervised learning can be conceptualized as a generalization of dictionary methods, where features associated with each categories (and their relative weight) are learned from the data
- By construction, they will outperform dictionary methods in classification tasks, as long as training sample is large enough

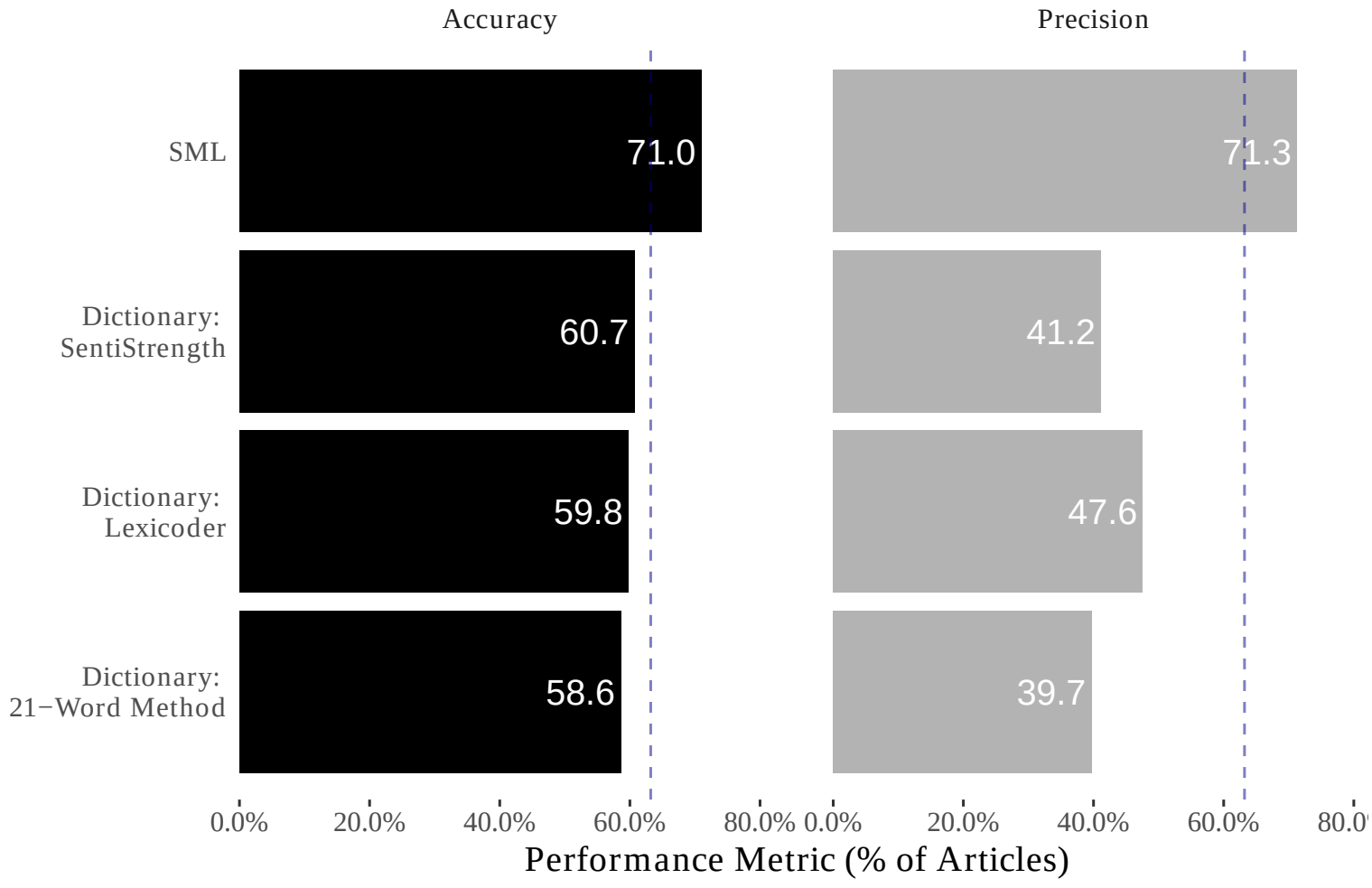
# Dictionaries vs supervised learning




**Source**: González-Bailón and Paltoglou (2015)

# Dictionaries vs supervised learning

Application: sentiment analysis of NYTimes articles




**Source**: Barberá et al (2019)

# Outline

- Supervised learning overview
- Creating a labeled set and evaluating its reliability
- Classifier performance metrics
- One classifier for text
  - Regularized regression

# Creating a labeled set

How do we obtain a **labeled set**?

- External sources of annotation
  - Disputed authorship of Federalist papers estimated based on known authors of other documents
  - Party labels for election manifestos
  - Legislative proposals by think tanks (text reuse)
- Expert annotation
  - "Canonical" dataset in Comparative Manifesto Project
  - In most projects, undergraduate students (expertise comes from training)
- Crowd-sourced coding
  - **Wisdom of crowds**: aggregated judgments of non-experts converge to judgments of experts at much lower cost (Benoit et al, 2016)
  - Easy to implement with FigureEight or MTurk

# Code the Content of a Sample of Tweets

Instructions ▲

In this job, you will be presented with tweets about the recent protests related to race and law enforcement in the U.S.

You will have to read the tweet and answer a set of questions about its content.

Read the tweet below paying close attention to detail:

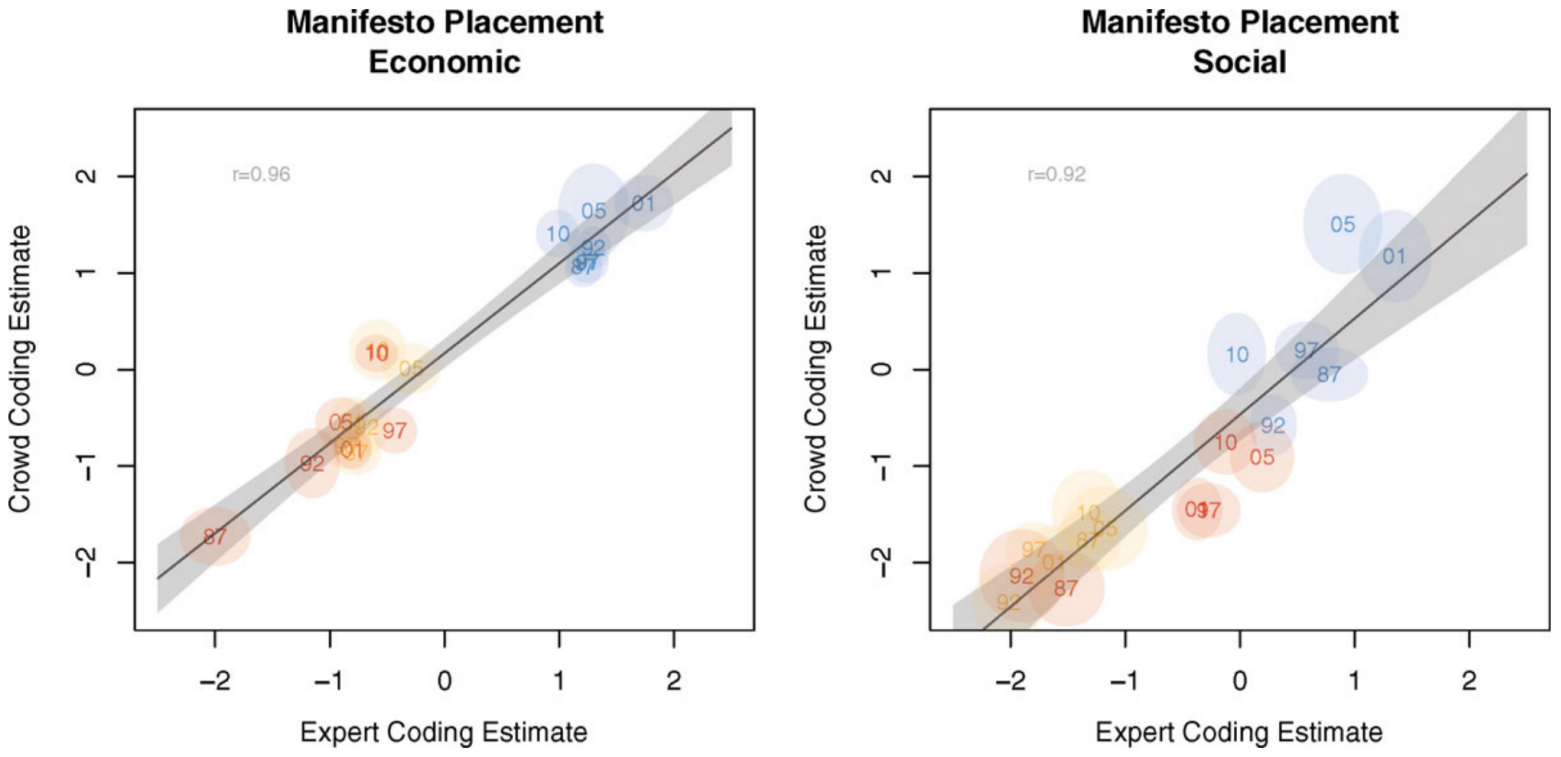
Tweet ID: **447**


**Is this tweet related to the ongoing debate about law enforcement and race in the United States?**

- ○ Yes
- ○ No
- ○ Don't Know

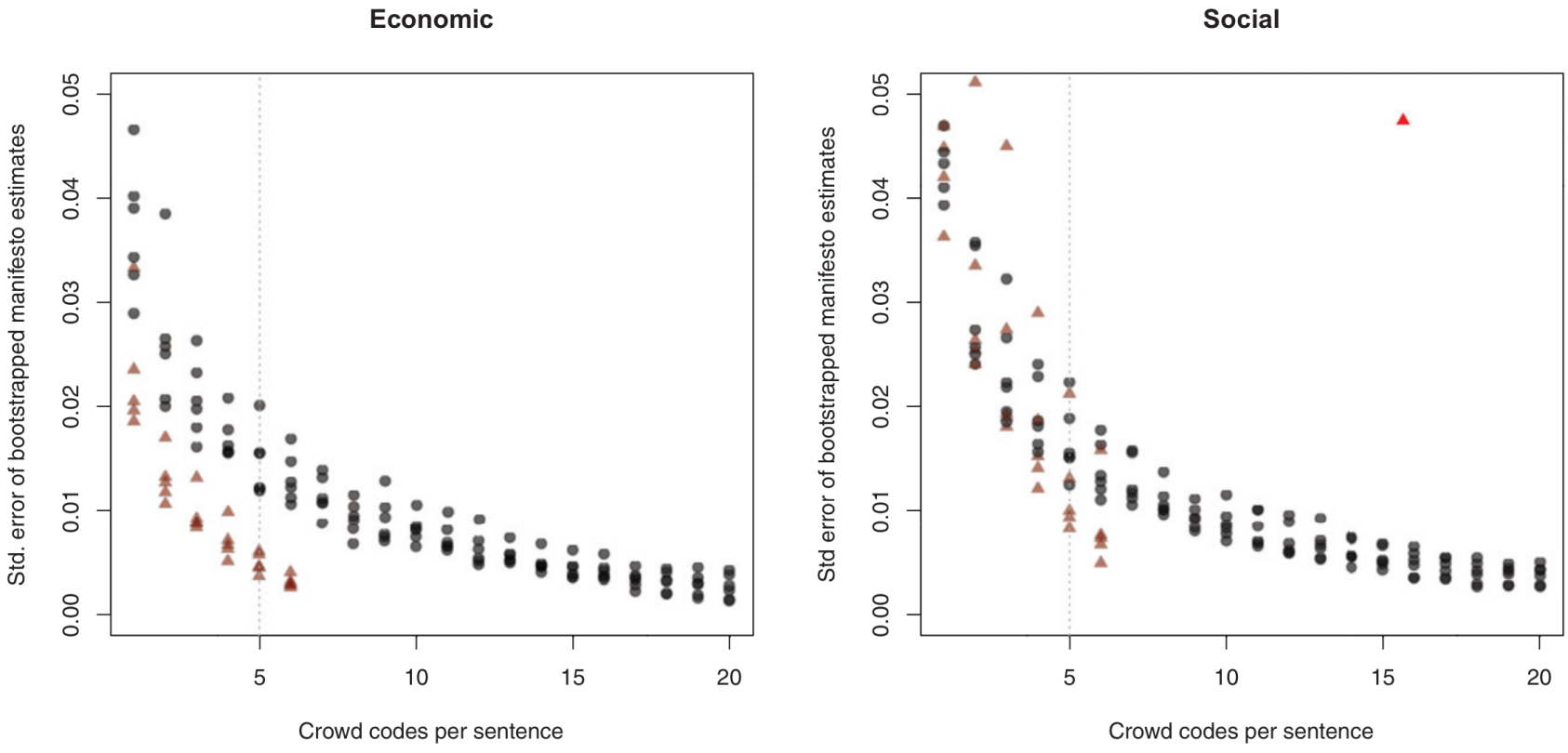
# Crowd-sourced text analysis (Benoit et al, 2016 APSR)

**FIGURE 3. Expert and Crowd-sourced Estimates of Economic and Social Policy Positions**



# Crowd-sourced text analysis (Benoit et al, 2016 APSR)

**FIGURE 5. Standard Errors of Manifesto-level Policy Estimates as a Function of the Number of Workers, for the Oversampled 1987 and 1997 Manifestos**




*Note:* Each point is the bootstrapped standard deviation of the mean of means aggregate manifesto scores, computed from sentence-level random n subsamples from the codes.

# Evaluating the quality of a labeled set

Measures of agreement:

- Percent agreement Very simple:

  (number of agreeing ratings) / (total ratings) * 100%

- Correlation
  - (usually) Pearson's *r*, aka product-moment correlation
  - Formula: $r_{AB} = \frac{1}{n-1} \sum_{i=1}^{n} \left( \frac{A_i - \bar{A}}{s_A} \right) \left( \frac{B_i - \bar{B}}{s_B} \right)$
  - May also be ordinal, such as Spearman's rho or Kendall's tau-b
  - Range is [0,1]
- Agreement measures
  - Take into account not only observed agreement, but also *agreement that would have occurred by chance*
  - Cohen's $\kappa$ is most common
  - Krippendorf's $\alpha$ is a generalization of Cohen's $\kappa$
  - Both range from [0,1]

# Outline

- Supervised learning overview
- Creating a labeled set and evaluating its reliability
- Classifier performance metrics
- One classifier for text
  - Regularized regression

# Computing performance

**Binary** outcome variables:

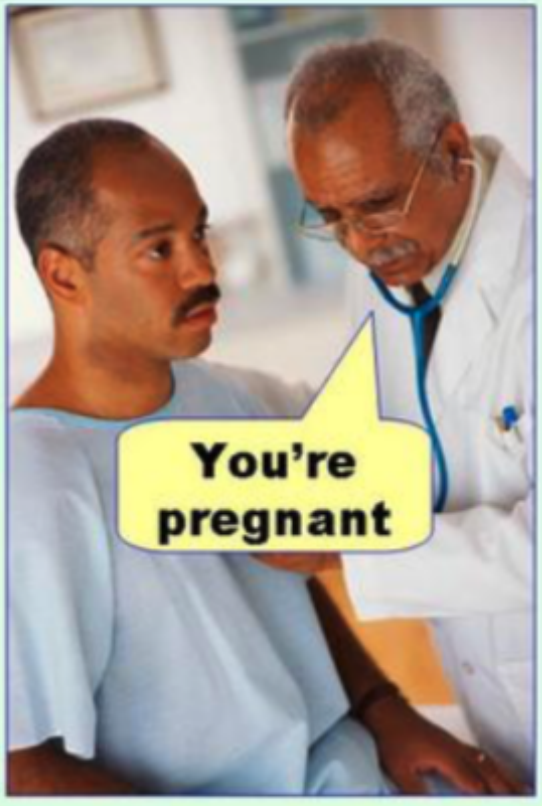
| | Actual value | |
|---|---|---|
| Classification | Ham | Spam |
| Ham | True negative | False negative |
| Spam | False positive | True positive |

Confusion matrix:

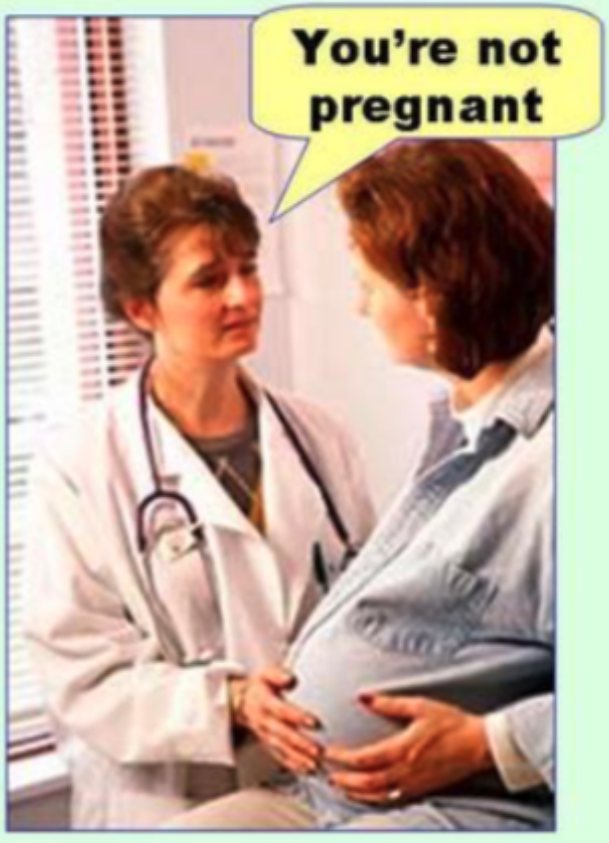
- True negatives and true positives are **correct** predictions (to maximize)
- False positives and false negatives are **incorrect** predictions (to minimize)

# Computing performance

**Type I error**
(false positive)




**Type II error**
(false negative)



# Computing performance: an example

- Performance **metrics**

| | Actual value | |
|---|---|---|
| Classification | Ham | Spam |
| Ham | True negative | False negative |
| Spam | False positive | True positive |

  - **Accuracy**: correct predictions / total of predictions
    - % of units that are correctly predicted
  - **Precision** for positive labels: (true positive) / (false positive + true positive)
    - % of units predicted to be positive that are indeed positive
  - **Recall** for positive labels: (true positive) / (true positive + false negative)
    - % of units that are positive and are predicted as such

# Computing performance: an example

| | Actual value | |
|---|---|---|
| **Classification** | Ham | Spam |
| Ham | 600 | 300 |
| Spam | 200 | 900 |

**Accuracy =** (600 + 900) / (600 + 900 + 200 + 300)

= 0.75

75% of all emails are correctly classified

# Computing performance: an example

| | **Actual value** | |
|---|---|---|
| **Classification** | Ham | Spam |
| Ham | 600 | 300 |
| Spam | 200 | 900 |

**Precision =** (900) / (900 + 200)

= 0.82

83% of all emails predicted to be spam are indeed **spam**

# Computing performance: an example

| | **Actual value** | |
|---|---|---|
| **Classification** | Ham | Spam |
| Ham | 600 | 300 |
| Spam | 200 | 900 |

**Recall =** (900) / (900 + 300)

= 0.75

75% of all **spam** emails are correctly classified

# Computing performance: an example

| | **Actual value** | |
|---|---|---|
| **Classification** | Ham | Spam |
| Ham | 1700 | 50 |
| Spam | 150 | 100 |

Total emails:
Ham = 1850
Spam = 150

**Accuracy =** (1700+100) / (1700+50+150+100) = 0.90
**Precision (spam) =** (100) / (150+100) = 0.40
**Recall (spam) =** (100) / (50+100) = 0.67

Accuracy can be misleadingly high!

Imagine extreme scenario: we classify everything as ham – accuracy would be 92.5%

# The trade-off between precision and recall

Two extreme scenarios (but same underlying data):

1) Model predicts *always spam*

| | **Actual value** | |
|---|---|---|
| **Classif.** | Ham | Spam |
| Ham | 0 | 0 |
| Spam | 800 | 1200 |

**Accuracy** = 1200 / 2000 = 0.60
**Precision (spam)** = 1200 / (800 + 1200) = 0.60
**Recall (spam)** = 1200 / 1200 = 1.00

High recall but low precision.

2) Model predicts *(almost) always ham*
(e.g. only emails with 10+ links as spam)

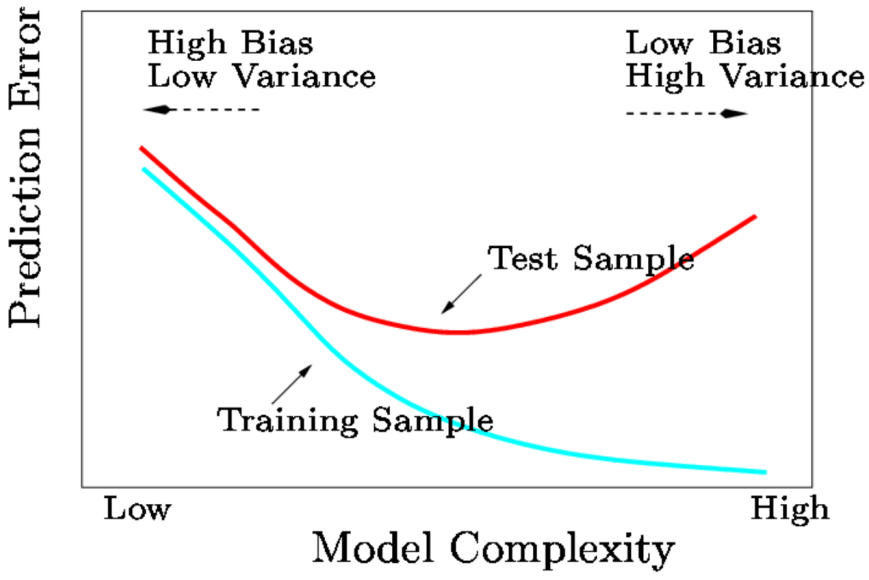
| | **Actual value** | |
|---|---|---|
| **Classif.** | Ham | Spam |
| Ham | 800 | 1190 |
| Spam | 0 | 10 |

**Accuracy** = (800 + 10) / 2000 = 0.40
**Precision (spam)** = 10 / 10 = 1.0
**Recall (spam)** = 10 / 1200 = 0.01

High precision but low recall.

# Measuring performance

- Classifier is trained to maximize in-sample performance
- But generally we want to apply method to new data
- Danger: overfitting



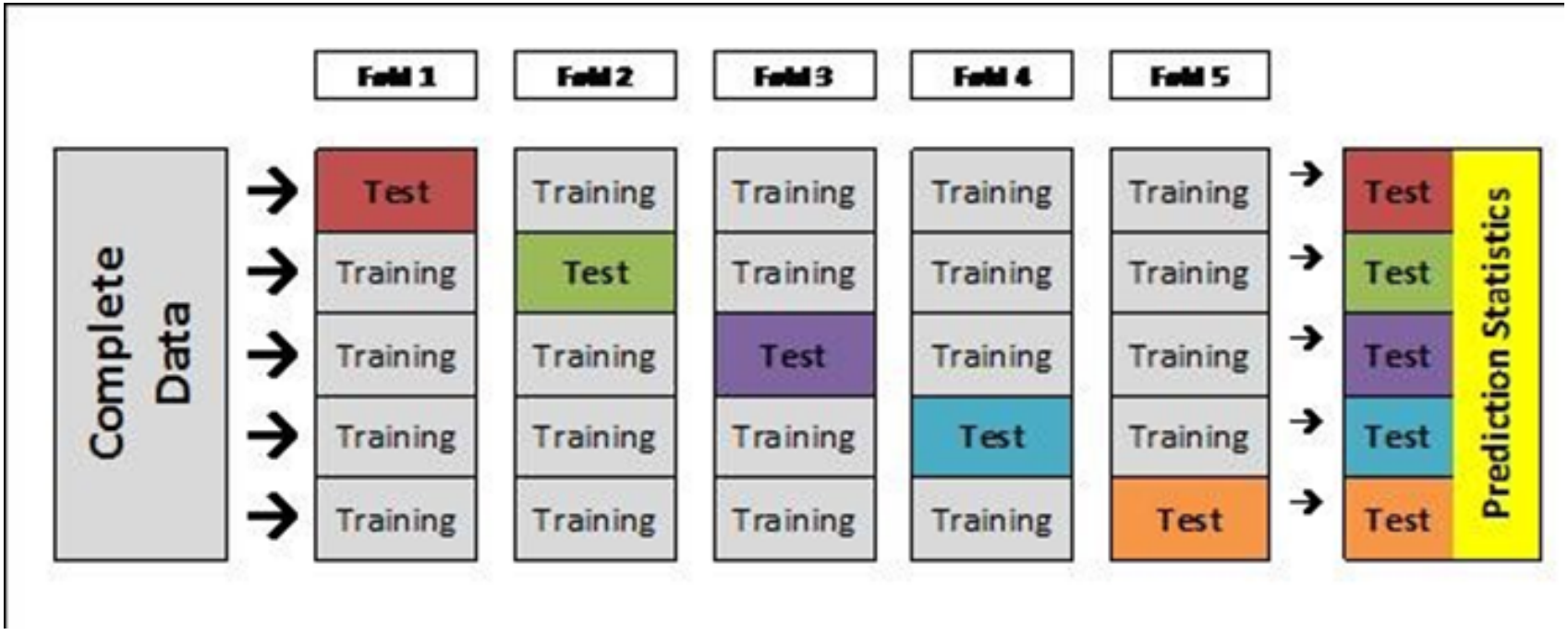

- Model is too complex, describes noise rather than signal
- Focus on features that perform well in labeled data but may not generalize (e.g. “inflation” in 1980s)
- In-sample performance better than out-of-sample performance

- Solutions?
  - Randomly split dataset into training and test set
  - Cross-validation

# Cross-validation

Intuition:

- Create K training and test sets ("folds") within training set.
- For each k in K, run classifier and estimate performance in test set within fold.
- Choose best classifier based on cross-validated performance



# Outline

- Supervised learning overview
- Creating a labeled set and evaluating its reliability
- Classifier performance metrics
- One classifier for text
  - Regularized regression

# Types of classifiers

General thoughts:

- Trade-off between accuracy and interpretability
- Parameters need to be cross-validated

Frequently used classifiers:

- Naive Bayes
- Regularized regression
- SVM
- Others: k-nearest neighbors, tree-based methods, etc.
- Ensemble methods

# Regularized regression

Assume we have:

- $i = 1, 2, \ldots, N$ documents
- Each document $i$ is in class $y_i = 0$ or $y_i = 1$
- $j = 1, 2, \ldots, J$ unique features
- And $x_{ij}$ as the count of feature $j$ in document $i$

We could build a linear regression model as a classifier, using the values of $\beta_0, \beta_1, \ldots, \beta_J$ that minimize:

$$RSS = \sum_{i=1}^{N} \left( y_i - \beta_0 - \sum_{j=1}^{J} \beta_j x_{ij} \right)^2$$

But can we?

- If $J > N$, OLS does not have a unique solution
- Even with $N > J$, OLS has low bias/high variance (overfitting)

# Regularized regression

What can we do? Add a penalty for model complexity, such that we now minimize:

$$\sum_{i=1}^{N}\left(y_i-\beta_0-\sum_{j=1}^{J}\beta_j x_{ij}\right)^2+\lambda\sum_{j=1}^{J}\beta_j^2 \rightarrow \text{ridge regression}$$

or

$$\sum_{i=1}^{N}\left(y_i-\beta_0-\sum_{j=1}^{J}\beta_j x_{ij}\right)^2+\lambda\sum_{j=1}^{J}|\beta_j| \rightarrow \text{lasso regression}$$

where $\lambda$ is the **penalty parameter** (to be estimated)

# Regularized regression

Why the penalty (shrinkage)?

- Reduces the variance
- Identifies the model if J $>$ N
- Some coefficients become zero (feature selection)

The penalty can take different forms:

- Ridge regression: $\lambda \sum_{j=1}^{J} \beta_j^2$ with $\lambda > 0$; and when $\lambda = 0$ becomes OLS
- Lasso $\lambda \sum_{j=1}^{J} |\beta_j|$ where some coefficients become zero.
- Elastic Net: $\lambda_1 \sum_{j=1}^{J} \beta_j^2 + \lambda_2 \sum_{j=1}^{J} |\beta_j|$ (best of both worlds?)

How to find best value of $\lambda$? Cross-validation.
Evaluation: regularized regression is easy to interpret, but often outperformed by more complex methods.