CS 2731 Introduction to Natural Language Processing

Session 5: Classifier evaluation

Michael Miller Yoder

September 13, 2023

University of Pittsburgh | School of Computing and Information

Course logistics

- Homework 1
  - Download new versions:
    - skeleton.py
    - snli.csv
  - **Deadline extended:** Sun 09-14 at 11:59pm
- Project area and contribution form will be due Thu 09-21
  - Not released yet
  - Please plan meeting with groups to discuss project ideas
- Discussion posts

Lecture overview: classifier evaluation

- Precision, recall, f1-score
- Train/dev/test and cross-validation sets
- Statistical significance testing between models
- Harms in classification
- Activity in breakout groups
    - Clickbait classification with Naive Bayes

How to evaluate your classifier

Gold labels and predicted labels

Document	gold label	predicted label
just plain boring	–	–
entirely predictable	–	–
no surprises and very few laughs	–	+
very powerful	+	–
the most fun film of the summer	+	+

The **gold** label is the label that a human assigned to the document.

The **predicted** or **hypothesized** label is the label that the classifier assigned to the document.

Slide credit: David Mortensen

We Can Evaluate a Classifier Using Accuracy

Accuracy is our first shot.

- Accuracy:

$$\mathrm{A}(\texttt{classify}) = \sum_{\substack{\mathbf{d}\in\mathcal{V}^{+}, \ell\in\mathcal{L}, \\ \texttt{classify}(\mathbf{d})=\ell}} p(\mathbf{x}, \ell) \tag{1}$$

  where p is the true distribution over data.
- Error is $1 - A$.
- Accuracy is estimated using a test set $\{(\bar{\mathbf{d}}_1, \bar{\ell}_1), (\bar{\mathbf{d}}_2, \bar{\ell}_2), \cdots, (\bar{\mathbf{d}}_m, \bar{\ell}_m)\}$

$$\hat{\mathrm{A}}(\texttt{classify}) = \frac{1}{m}\sum_{i=1}^{m}\{\texttt{classify}(\bar{\mathbf{d}}_i) = \bar{\ell}_i\} \tag{2}$$

Slide credit: David Mortensen

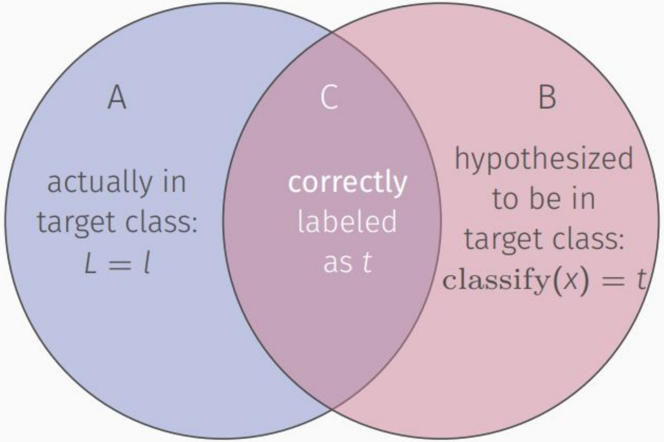
Issues with Using Test Set Accuracy

- Suppose that 99% of the messages you get are spam and 1% are not (we're being realistic here).
- Suppose, too, that you have a spam filter that **always** classifies messages as spam.
  1. You would get lots of work done, because you wouldn't have to answer email.
  2. The email classifier would have $\hat{A} \approx 0.99$.
  3. Everybody would be happy, except for your boss.
- You must take other things into account:
  - Relative importance of the classes the cost of the error types.
  - Variance due to the test data.

Slide credit: David Mortensen

Evaluation in the Two-class case

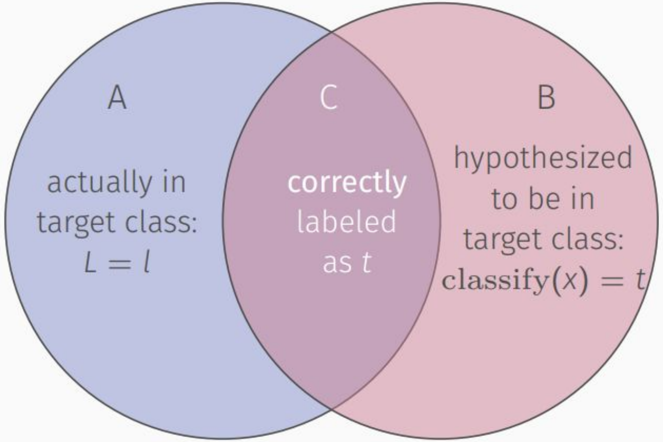
- Suppose we have one of the classes $t \in \mathcal{L}$ as the target class.
- We would like to identify documents with label t in the test data.
- We get

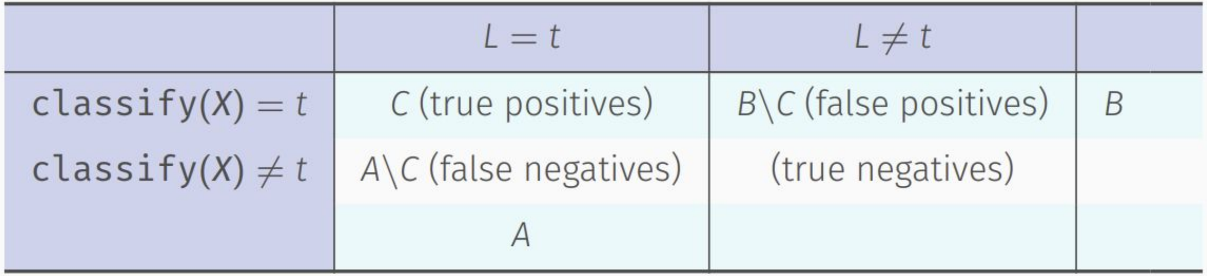

A
actually in target class: $L = l$
C
correctly labeled as t
B
hypothesized to be in target class: $\mathrm{classify}(x) = t$

- Precision $\hat{P} = \frac{C}{B}$ (percentage of documents `classify` correctly labeled as t)
- Recall $\hat{R} = \frac{C}{A}$ (percentage of actual t labeled documents correctly labeled as t)
- $F_1 = 2\frac{\hat{P} \cdot \hat{R}}{\hat{P} + \hat{R}}$

Slide credit: David Mortensen

A Different View – Contingency Tables





	$L = t$	$L \neq t$	
$\texttt{classify}(X) = t$	C (true positives)	$B \backslash C$ (false positives)	B
$\texttt{classify}(X) \neq t$	$A \backslash C$ (false negatives)	(true negatives)	
	A		

precision = tp/(tp+fp)

recall = tp/(tp+fn)

Slide credit: David Mortensen

Why precision and recall

- Our dumb spam detector: labels everything as spam (99% of email is)
- = a dumb “important email” detector that labels nothing as important
    - 2-way precision and recall are specific to a target class
- Accuracy=99%

  but

- Recall = 0 (out of all actually important emails, got none)
- Precision and recall, unlike accuracy, emphasize true positives: finding the things that we are supposed to be looking for

Slide adapted from Jurafksy & Martin

A combined measure: F

F measure: a single number that combines P and R:

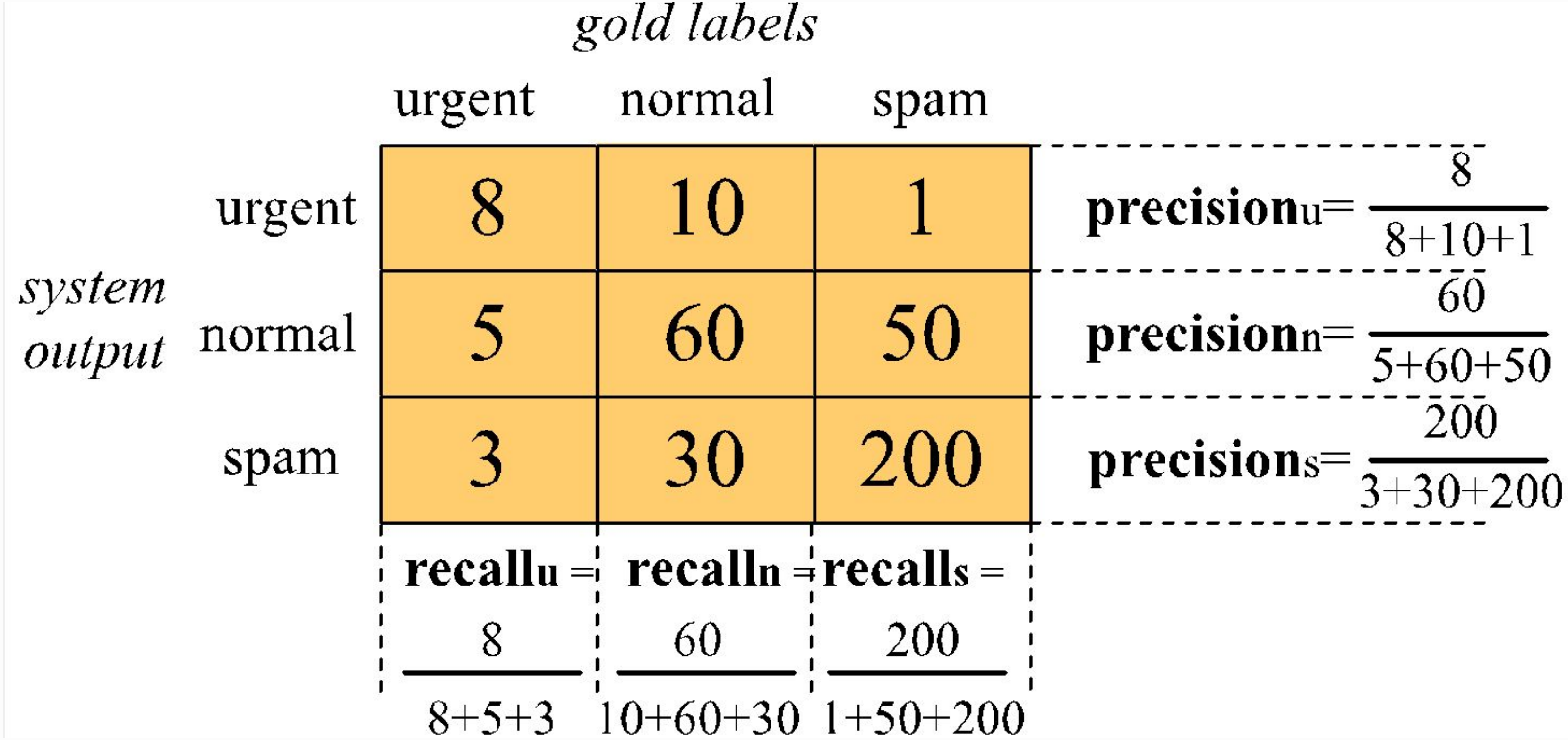
$$F_\beta = \frac{(\beta^2+1)PR}{\beta^2 P + R}$$

We almost always use balanced F_1 (i.e., β = 1). Harmonic mean

$$\mathrm{F}_1 = \frac{2PR}{P+R}$$

Slide adapted from Jurafksy & Martin

Confusion matrix for 3-class classification



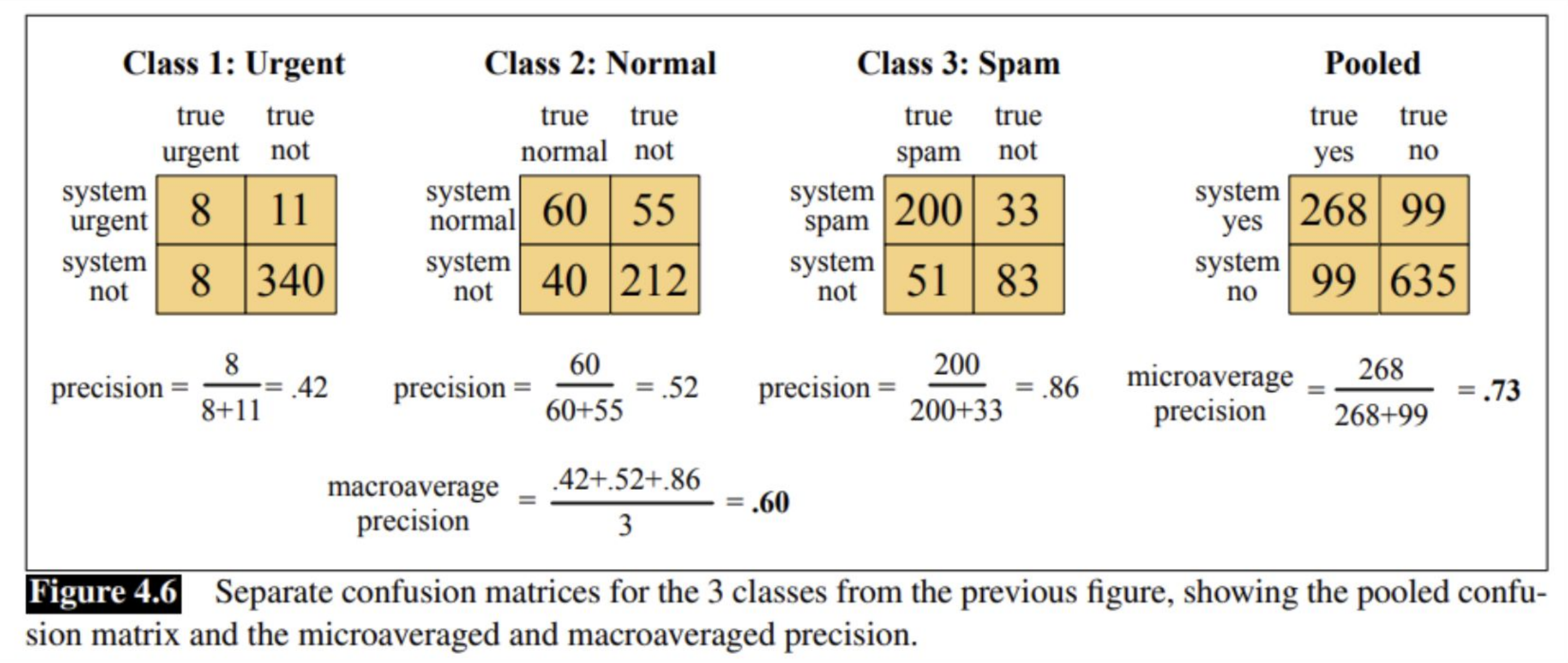
		gold labels			
		urgent	normal	spam	
system output	urgent	8	10	1	$\textbf{precision}_{u} = \frac{8}{8+10+1}$
	normal	5	60	50	$\textbf{precision}_{n} = \frac{60}{5+60+50}$
	spam	3	30	200	$\textbf{precision}_{s} = \frac{200}{3+30+200}$
		$\textbf{recall}_{u} = \frac{8}{8+5+3}$	$\textbf{recall}_{n} = \frac{60}{10+60+30}$	$\textbf{recall}_{s} = \frac{200}{1+50+200}$	

Slide adapted from Jurafksy & Martin

Evaluation with > 2 Classes

- **Macroaveraged precision and recall**: let each class be the target and report the average $\hat{P}$ and $\hat{R}$ across all classes.
- **Microaveraged precision and recall**: pool all one-vs.-rest decisions into a single contingency table, calculate $\hat{P}$ and $\hat{R}$ from that.

Example of more than two classes



Class 1: Urgent

	true urgent	true not
system urgent	8	11
system not	8	340

$$\text{precision} = \frac{8}{8+11} = .42$$

Class 2: Normal

	true normal	true not
system normal	60	55
system not	40	212

$$\text{precision} = \frac{60}{60+55} = .52$$

Class 3: Spam

	true spam	true not
system spam	200	33
system not	51	83

$$\text{precision} = \frac{200}{200+33} = .86$$

Pooled

	true yes	true no
system yes	268	99
system no	99	635

$$\begin{array}{c}\text{microaverage}\\ \text{precision}\end{array} = \frac{268}{268+99} = \mathbf{.73}$$

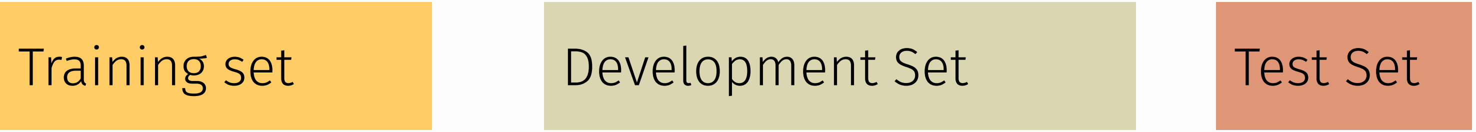
$$\begin{array}{c}\text{macroaverage}\\ \text{precision}\end{array} = \frac{.42+.52+.86}{3} = \mathbf{.60}$$

Figure 4.6 Separate confusion matrices for the 3 classes from the previous figure, showing the pooled confusion matrix and the microaveraged and macroaveraged precision.

Slide credit: David Mortensen, Jurafksy & Martin

Train/dev/test splits and cross-validation

Development Sets ("Devsets") and Cross-validation


Training set

Development Set

Test Set

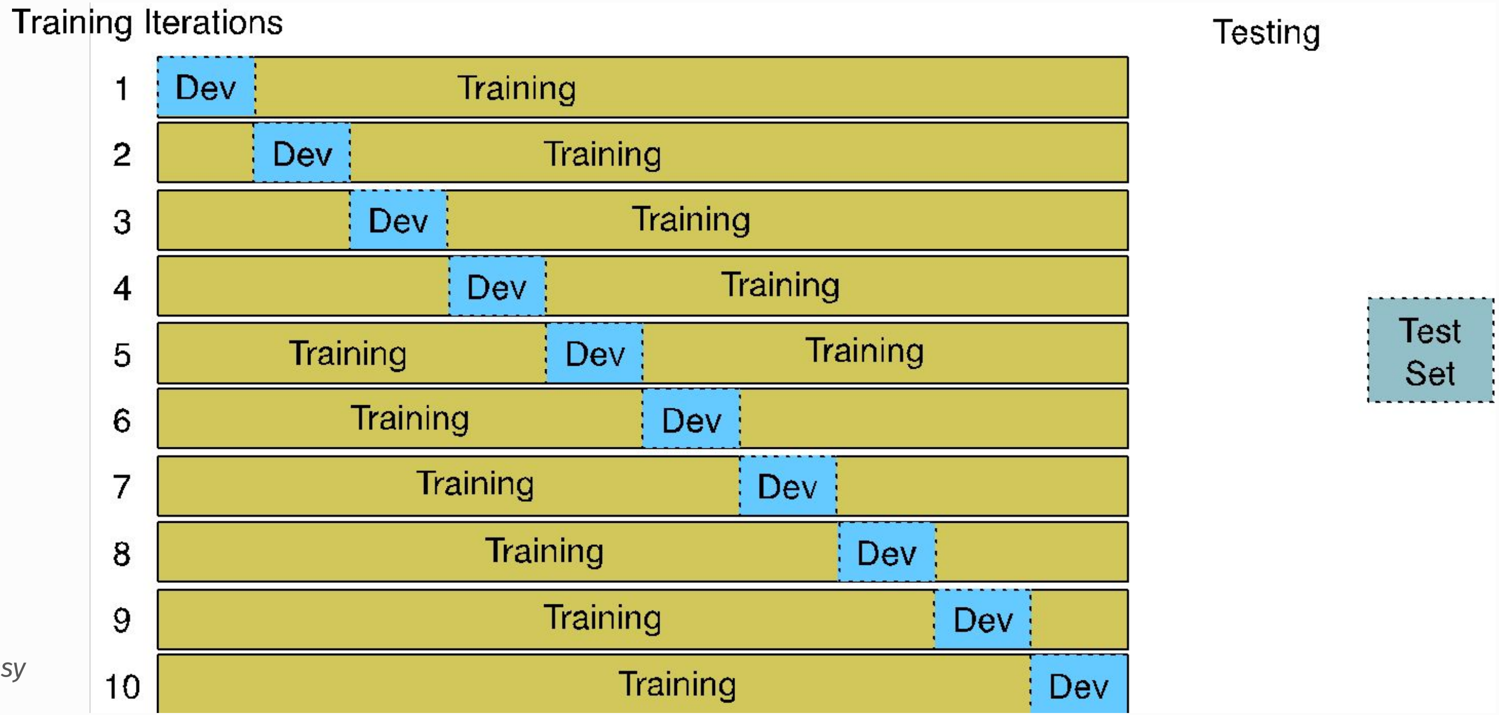
Train on training set, tune on dev set, report on test set

- **Do not look at test set**
- Using a dev set avoids overfitting (‘tuning to the test set’)
- More conservative estimate of performance
- But paradox: want as much data as possible for training, and as much for dev; how to split?

Slide adapted from Jurafksy & Martin

Cross-validation: multiple splits

- Pool results over splits, Compute pooled dev performance
- Good for when you don't have much data (<10k instances rule of thumb)



Training Iterations

Testing

1 Dev Training

2 Dev Training

3 Dev Training

4 Dev Training

5 Training Dev Training

6 Training Dev

7 Training Dev

8 Training Dev

9 Training Dev

10 Training Dev

Test Set

Slide adapted from Jurafksy & Martin

Statistical significance testing among models

How do we know if one classifier is better than another?

Given:

- Classifier A and B
- Metric M: M(A,x) is the performance of *A* on test set *x*
- δ(x): the performance difference between A, B on x:
  - $\delta(x) = M(A,x) - M(B,x)$

- We want to know if $\delta(x)>0$, meaning A is better than B
- δ(x) is called the **effect size**
- Suppose we look and see that δ(x) is positive. Are we done?
- No! This might be just an accident of this one test set, or circumstance of the experiment. Instead…

Slide adapted from Jurafksy & Martin

Statistical Hypothesis Testing

Consider two hypotheses:

- ○ Null hypothesis: A isn't better than B
- ○ A is better than B

$$H_0 \ : \ \delta(x) \leq 0$$

$$H_1 \ : \ \delta(x) > 0$$

We want to rule out H_0

We create a random variable X ranging over test sets

And ask, how likely, if H_0 is true, is it that among these test sets we would see the δ(x) we did see?

- Formalized as the p-value:

$$P(\delta(X) \geq \delta(x) | H_0 \text{ is true})$$

Slide adapted from Jurafksy & Martin

Statistical Hypothesis Testing

$$P(\delta(X) \geq \delta(x) | H_0 \text{ is true})$$

- In our example, this p-value is the probability that we would see δ(*x*) assuming H_0 (=*A* is not better than *B*).
    - If H_0 is true but δ(*x*) is huge, that is surprising! Very low probability!
- A very small p-value means that the difference we observed is very unlikely under the null hypothesis, and we can reject the null hypothesis
- Very small: .05 or .01
- A result (e.g., "*A* is better than *B*") is **statistically significant** if the δ we saw has a probability that is below the threshold and we therefore reject this null hypothesis.

Slide adapted from Jurafksy & Martin

Statistical Hypothesis Testing

- How do we compute this probability?
- In NLP, we don't tend to use parametric tests (like t-tests)
- Instead, we use non-parametric tests based on sampling: artificially creating many versions of the setup.
- For example, suppose we had created zillions of test sets x' with an assumption that the null hypothesis is true
  - Now we measure the value of δ(x') on each test set
  - That gives us a distribution
  - Now set a threshold (say .01).
  - So if we see that in 99% of the test sets δ(x) > δ(x’)
    - We conclude that our original test set delta was a real delta and not an artifact.

Slide adapted from Jurafksy & Martin

Bootstrap test [Efron & Tibshirani 1993]

Can apply to any metric (accuracy, precision, recall, F1, etc).

Bootstrap means to repeatedly draw large numbers of smaller samples with replacement (called **bootstrap samples**) from an original larger sample.

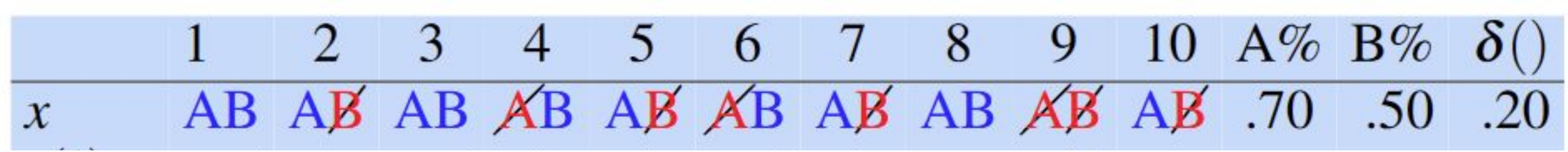
Slide adapted from Jurafksy & Martin

Bootstrap example

Consider a baby text classification example with a test set x of 10 documents, using accuracy as metric.

Suppose these are the results of systems A and B on x, with 4 outcomes (A & B both right, A & B both wrong, A right/B wrong, A wrong/B right):

either A+B both correct, or



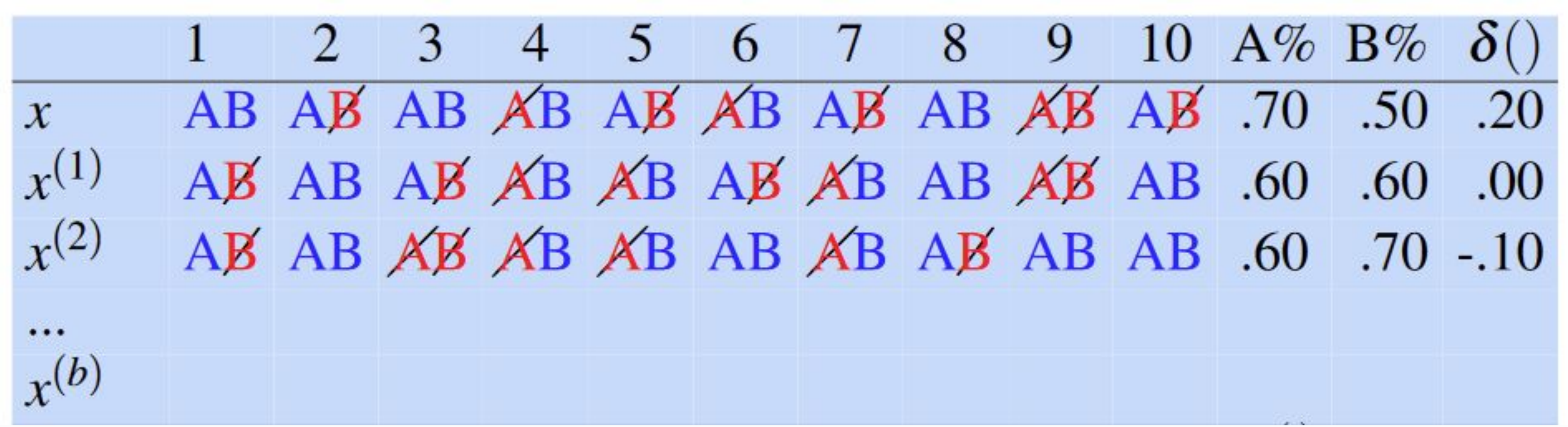
	1	2	3	4	5	6	7	8	9	10	A%	B%	δ()
x	AB	A~~B~~	AB	~~A~~B	A~~B~~	~~A~~B	A~~B~~	AB	~~AB~~	A~~B~~	.70	.50	.20

Slide adapted from Jurafksy & Martin

Bootstrap example

Now we create, many, say, *b*=10,000 virtual test sets *x*(*i*), each of size *n* = 10.

To make each *x*(*i*), we randomly select a cell from row *x*, with replacement, 10 times:



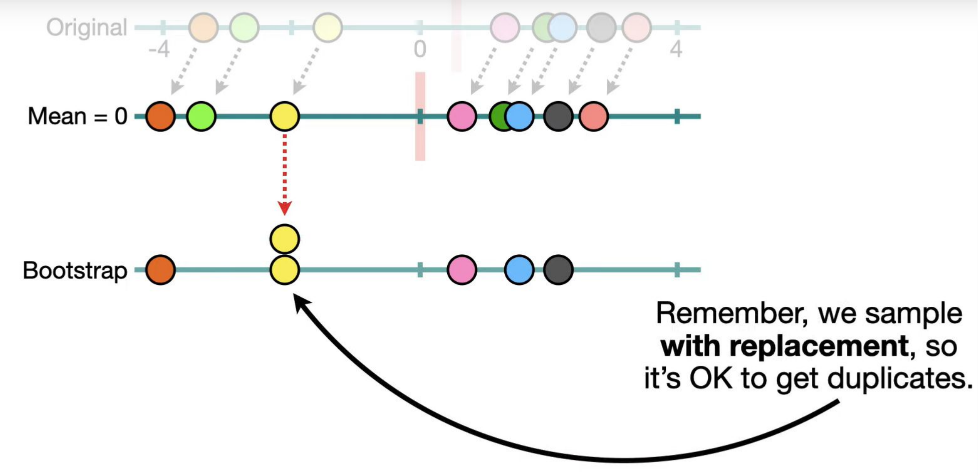
	1	2	3	4	5	6	7	8	9	10	A%	B%	$\delta()$
x	AB	A~~B~~	AB	~~A~~B	A~~B~~	~~A~~B	A~~B~~	AB	~~AB~~	A~~B~~	.70	.50	.20
$x^{(1)}$	A~~B~~	AB	A~~B~~	~~A~~B	~~A~~B	A~~B~~	~~A~~B	AB	~~AB~~	AB	.60	.60	.00
$x^{(2)}$	A~~B~~	AB	~~AB~~	~~A~~B	~~A~~B	AB	~~A~~B	A~~B~~	AB	AB	.60	.70	-.10
…													
$x^{(b)}$													

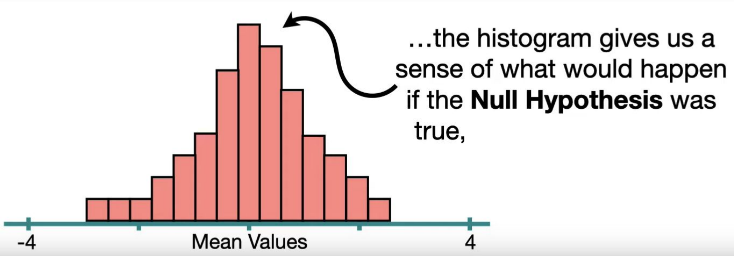
Slide adapted from Jurafksy & Martin

Bootstrap example

Now we have a distribution! But that distribution was drawn from the test set, which we know has a bias of its mean.

To represent the null hypothesis, we can shift values to mean = 0 and calculate the mean values

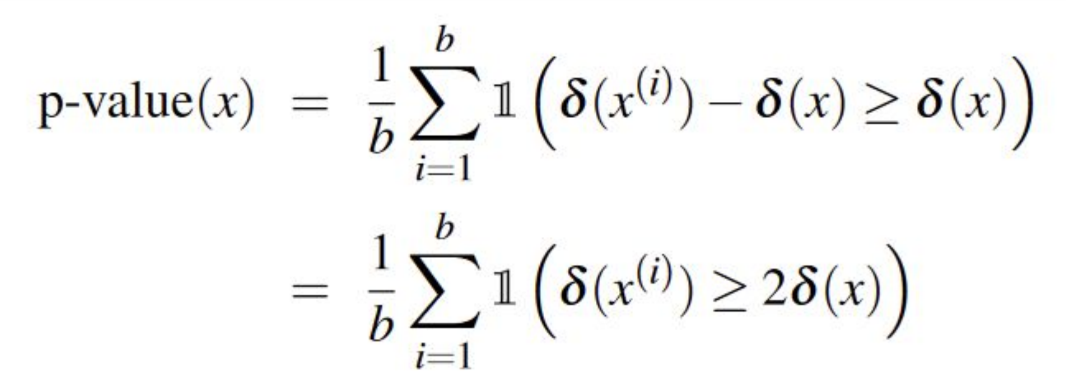

Original
-4
0
4
Mean = 0
Bootstrap
Remember, we sample **with replacement**, so it's OK to get duplicates.



Slide adapted from Jurafksy & Martin and StatsQuest

Bootstrap example

- Compute the p-value by counting how often δ(*x*′) exceeds the expected value of δ(*x*) by δ(*x*) or more
- This is how likely/surprising our observed δ(*x*) is under a null hypothesis distribution



$$
\begin{aligned}
\text{p-value}(x) &= \frac{1}{b}\sum_{i=1}^{b} \mathbb{1}\left(\boldsymbol{\delta}(x^{(i)}) - \boldsymbol{\delta}(x) \geq \boldsymbol{\delta}(x)\right) \\
&= \frac{1}{b}\sum_{i=1}^{b} \mathbb{1}\left(\boldsymbol{\delta}(x^{(i)}) \geq 2\boldsymbol{\delta}(x)\right)
\end{aligned}
$$

Slide adapted from Jurafksy & Martin

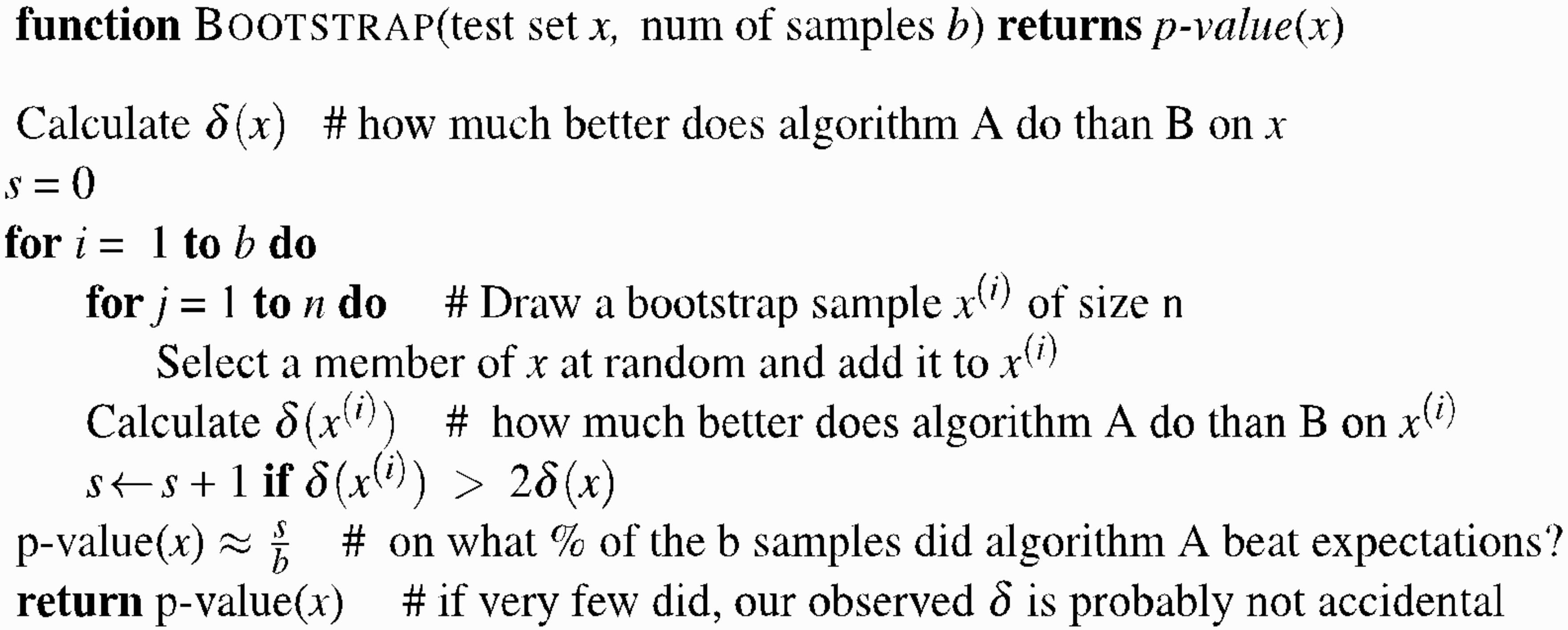
Bootstrap example

Suppose:

- We have 10,000 test sets $x(i)$ and a threshold of .01
- And in only 47 of the test sets do we find that $\delta(x(i)) \geq 2\delta(x)$
- The resulting p-value is .0047
- This is smaller than .01, indicating $\delta\ (x)$ is indeed sufficiently surprising
- And we reject the null hypothesis and conclude *A* is better than *B*.

Slide adapted from Jurafksy & Martin

Paired bootstrap example [Berg-Kirkpatrick et al. 2012]



```
function BOOTSTRAP(test set $x$, num of samples $b$) returns $p$-value($x$)

 Calculate $\delta(x)$   # how much better does algorithm A do than B on $x$
$s = 0$
for $i$ = 1 to $b$ do
    for $j$ = 1 to $n$ do     # Draw a bootstrap sample $x^{(i)}$ of size n
        Select a member of $x$ at random and add it to $x^{(i)}$
    Calculate $\delta(x^{(i)})$    # how much better does algorithm A do than B on $x^{(i)}$
    $s \leftarrow s + 1$ if $\delta(x^{(i)}) > 2\delta(x)$
 p-value($x$) $\approx \frac{s}{b}$    # on what % of the b samples did algorithm A beat expectations?
 return p-value($x$)    # if very few did, our observed $\delta$ is probably not accidental
```

Slide adapted from Jurafksy & Martin

Harms in classification in NLP

Harms in sentiment classifiers

Kiritchenko and Mohammad (2018) found that most sentiment classifiers assign lower sentiment and more negative emotion to sentences with African American names in them.

This perpetuates negative stereotypes that associate African Americans with negative emotions

Slide adapted from Jurafksy & Martin

Harms in toxicity classification

Toxicity detection is the task of detecting hate speech, abuse, harassment, or other kinds of toxic language

But some toxicity classifiers incorrectly flag as being toxic sentences that are non-toxic but simply mention identities like blind people, women, or gay people.

This could lead to censorship of discussion about these groups.

Slide adapted from Jurafksy & Martin

What causes these harms?

Can be caused by:

- Problems in the training data; machine learning systems are known to amplify the biases in their training data.
- Problems in the human labels
- Problems in the resources used (like lexicons)
- Problems in model architecture (like what the model is trained to optimized)

Mitigation of these harms is an open research area

Can’t fully “remove” bias because exists in societies that produced texts we use

So need to be explicit about what those biases may be through **data statements** and **model cards**

Slide adapted from Jurafksy & Martin

Data statements [Bender & Friedman 2018]

For each dataset you release, document:

- Curation rationale: why were certain texts selected
- Language variety
- Speaker demographic
- Annotator demographic
- Speech situation
  - Time and place, modality, scripted vs spontaneous, intended audience
- Text characteristics
  - Genre, topic
- Recording quality (for speech)

Slide adapted from Jurafksy & Martin

Model cards [Mitchell et al. 2019]

For each algorithm you release, document:

- training algorithms and parameters
- training data sources, motivation, and preprocessing
- evaluation data sources, motivation, and preprocessing
- intended use and users
- model performance across different demographic or other groups and environmental situations

Discussion forum themes

- Transparency in specific fields
  - Robotics, embedded systems, healthcare
- Challenges
  - Profit incentives. Is bad PR enough to encourage transparency?
  - Competitive advantages to keeping training data secret
  - Legal obligations to privacy
  - Evolving/online training for model cards, changing definitions of e.g. hate speech
  - Biases in deciding what are potential harms, which subpopulations to test
- Also need
  - 3rd-party test sets for bias (HateCheck)
  - Explainability
  - To work to mitigate/address biases instead of just naming them

Activity in breakout groups

Clickbait classification activity

- Groups of ~5 with people who
  - Gather/find ideas
  - Organize/plan
  - Implement
- Download clickbait data from Kaggle (“Clickbait Dataset” by Aman Anand)
- Come up with an approach, implement and evaluate it
  - Use Naive Bayes
    - scikit-learn’s MultinomialNB is a good option
  - Only use the text of the headline
  - Be able to explain your code
- Use anything: example notebooks, lexicons, Internet resources, generative AI
- Questions to consider
  - What kind of preprocessing will you do? (tokenization, stemming, etc)
  - How will you divide the dataset into training, dev, test sets?
  - What features will you use (ngrams as well as unigrams bag of words? Any custom features?)
  - What’s the fewest number of features you can use for good performance? (feature selection)
  - What are most associated features with each class?

Questions?

Homework 1 due Sun, Sep 17