

CS 2731

Introduction to Natural Language Processing

Session 6: Classifier evaluation

Michael Miller Yoder

September 16, 2024



University of
Pittsburgh

School of Computing and Information

Course logistics

- [Homework 1](#) **due this Thu Sep 19**
 - Feel free to ask questions in the Canvas discussion forum, email Jayden or Michael
- [Project idea submission form](#) **due this Fri Sep 20 (pushed back)**
 - All students must submit one idea for credit
 - You can submit one of the example project ideas from the [website](#)
 - Ideas do not need to be fully developed

Lecture overview: classifier evaluation

- Overview of CRC resources, including using GPU (Jayden)
 - Jupyter OnDemand manual: <https://crc-pages.pitt.edu/user-manual/web-portals/jupyter-ondemand/>
 - Resources on the gpu cluster are described here, <https://crc.pitt.edu/resources/computing-hardware>
- Precision, recall, f1-score
- Train/dev/test and cross-validation sets
- Statistical significance testing between models
- Harms in classification
- Coding activity
 - 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.

We Can Evaluate a Classifier Using Accuracy

Accuracy is our first shot.

- Accuracy:

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

where p is the true distribution over data.

- Error is $1 - A$.
- Accuracy is estimated using a test set $\{(\bar{d}_1, \bar{\ell}_1), (\bar{d}_2, \bar{\ell}_2), \dots, (\bar{d}_m, \bar{\ell}_m)\}$

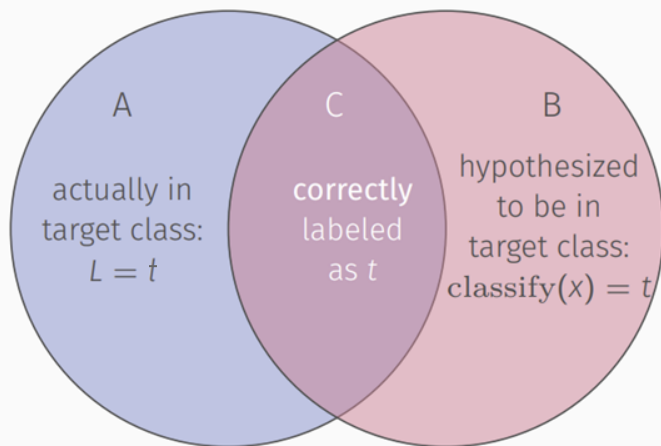
$$\hat{A}(\text{classify}) = \frac{1}{m} \sum_{i=1}^m \{\text{classify}(\bar{d}_i) = \bar{\ell}_i\} \quad (2)$$

Issues with using test set accuracy

- Imagine an “important email” classifier that notifies you when you get an important email
- Suppose that 99% of the messages you receive are junk and not important (we’re being realistic here)
- An easy important email classifier: classify **nothing** as important
 - You would get lots of work done, because you wouldn’t be distracted by email
 - The email classifier would have an accuracy of ~99%
 - Everybody would be happy except for your boss
- You must take the relative importance of the classes into account, and the cost of the error types

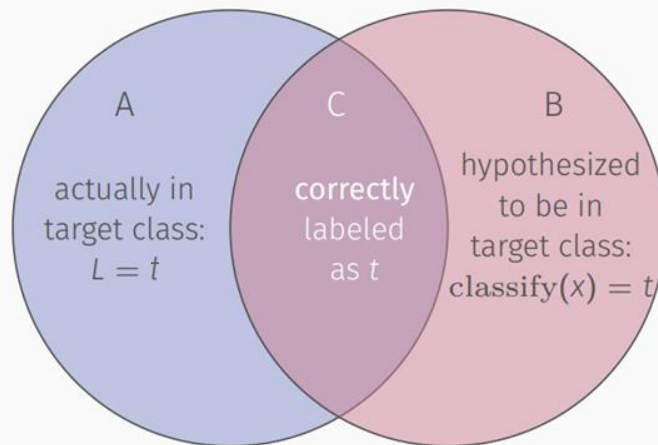
Evaluation in the Two-class case

- 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



- 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)
- $F_1 = 2 \frac{\hat{P} \cdot \hat{R}}{\hat{P} + \hat{R}}$

A Different View – Contingency Tables



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

$$\text{precision} = \frac{\text{tp}}{\text{tp} + \text{fp}}$$

$$\text{recall} = \frac{\text{tp}}{\text{tp} + \text{fn}}$$

Why precision and recall

- 2-way precision and recall are specific to a target class
- Accuracy=99% on important email detection
 - 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

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., $\beta = 1$) . Harmonic mean

$$F_1 = \frac{2PR}{P + R}$$

Confusion matrix for 3-class classification

		<i>gold labels</i>			
		urgent	normal	spam	
<i>system output</i>	urgent	8	10	1	precision_u = $\frac{8}{8+10+1}$
	normal	5	60	50	precision_n = $\frac{60}{5+60+50}$
	spam	3	30	200	precision_s = $\frac{200}{3+30+200}$
		recall_u = $\frac{8}{8+5+3}$	recall_n = $\frac{60}{10+60+30}$	recall_s = $\frac{200}{1+50+200}$	

- **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

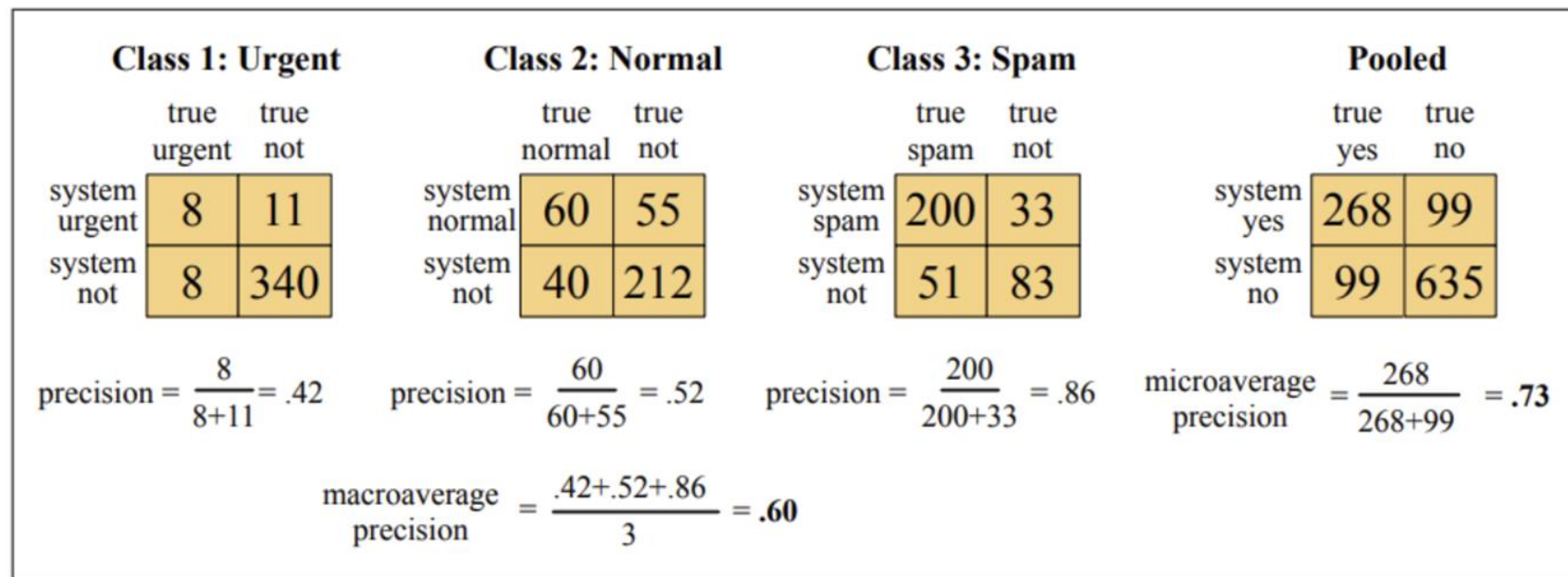


Figure 4.6 Separate confusion matrices for the 3 classes from the previous figure, showing the pooled confusion matrix and the microaveraged and macroaveraged precision.

Train/dev/test splits and cross-validation

Development Sets ("Devsets") and Cross-validation

Training set

Development Set

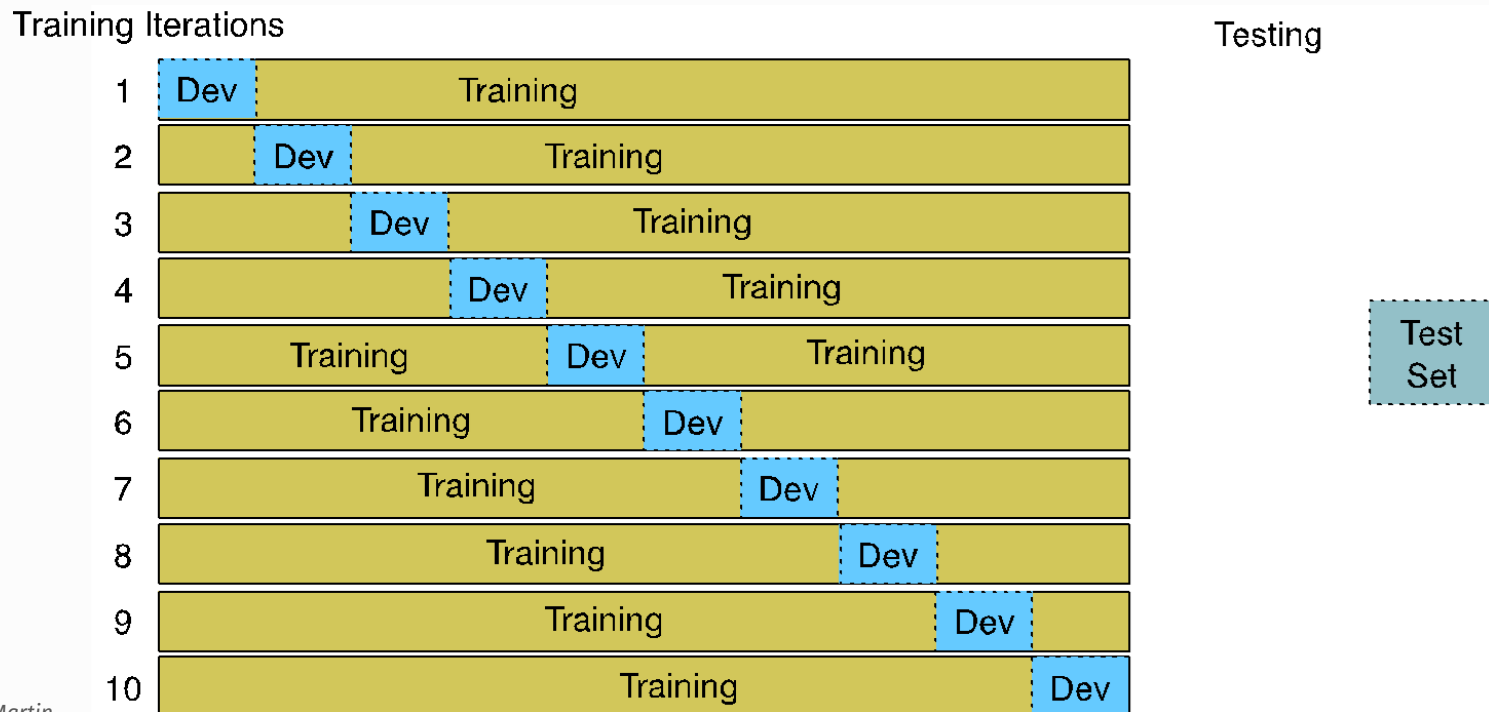
Test Set

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?

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)



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
- $\delta(x)$: 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

- $\delta(x)$ is called the **effect size**
- Suppose we look and see that $\delta(x)$ is positive. Are we done?
- No! This might be just an accident of this one test set, or circumstance of the experiment. Instead...

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 $\delta(x)$ we did see?

- Formalized as the p-value:

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

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 $\delta(x)$ assuming H_0 (=A is not better than B).
 - If H_0 is true but $\delta(x)$ 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.

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 $\delta(x')$ 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 $\delta(x) > \delta(x')$
 - We conclude that our original test set delta was a real delta and not an artifact.

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.

See the textbook section 4.9 for further details (or ask me later!)

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

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.

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**

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)

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

In an area of research or industry you are familiar with (in CS or outside CS), do you see issues with transparency of data and/or models?

Is it clear under what circumstances datasets were collected? Are the intended uses of machine learning models clearly stated?

Group activity: text classification

Clickbait classification code walk-through

- What steps are needed to go from labeled text data to a classifier?
 - Load in data, matched with labels
 - Preprocessing (tokenize, remove punctuation? lowercase?)
 - Extract features (bag of words, etc)
 - Train classifier
 - Test classifier
 - Interpret classifier
- Colab notebook here: <https://colab.research.google.com/drive/1J-FcHTFYXTsNgcEB19kQcqJnE8CsbSe7?usp=sharing>

Questions?

Homework 1 due this Thu Sep 19