PPOL 628: Text as Data – Computational Linguistics for Social Scientists

Class 9: Supervised Learning

Today

- Lecture: an overview of supervised learning for text.
- Lab: supervised_learning.R
- Website: github.com/matthewjdenny/PPOL_628_Text_As_Data

Supervised Learning with Text

- **What it is:** Using document covariates (such as term counts in those documents) to predict a label for those documents.

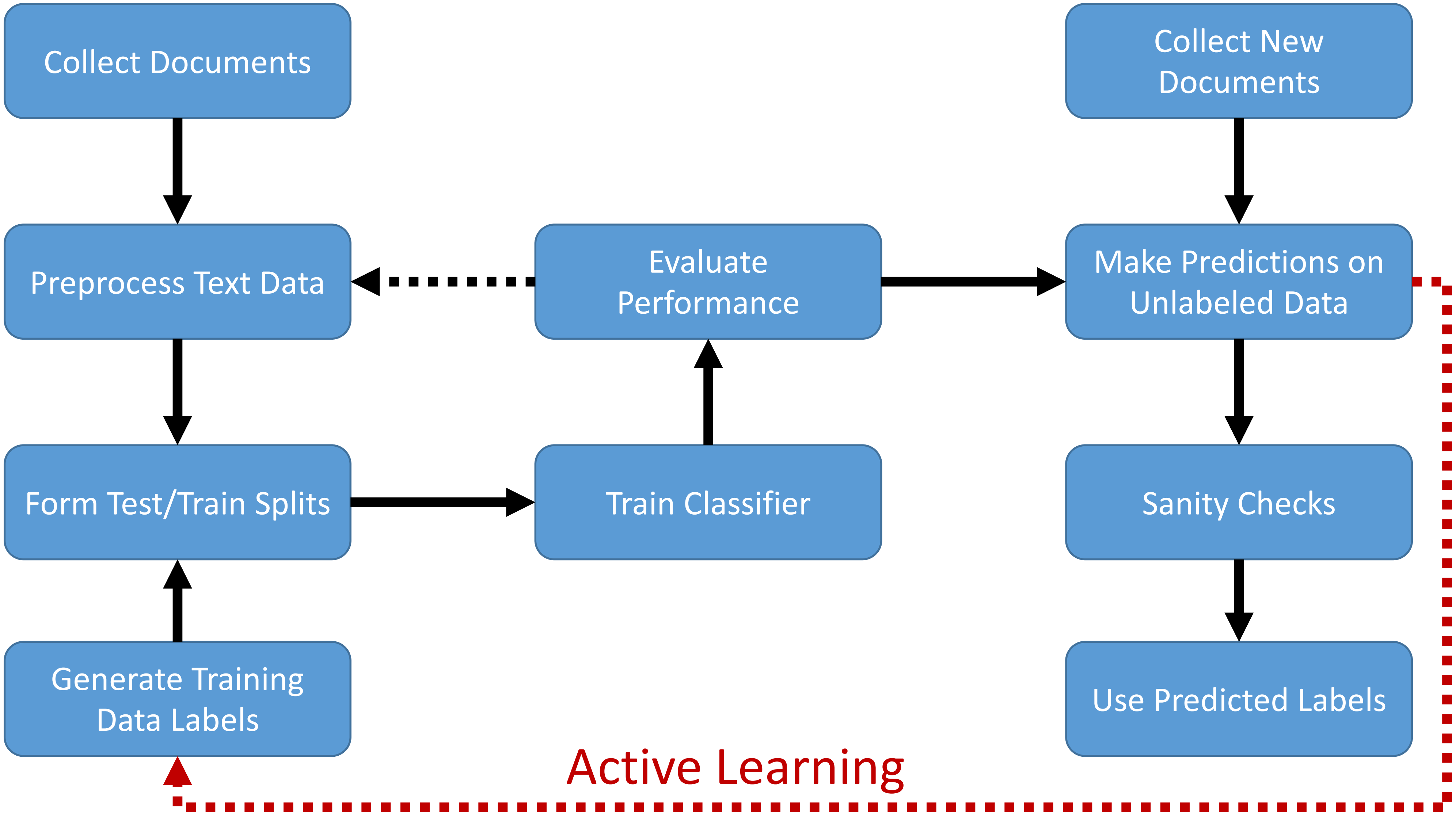
- **Prerequisites:** You need to have training data.
    - Usually in the form of some number of hand labeled documents.
    - Can also take the form of document metadata (labels assigned by some other process).

- **When we use it:** For problems where we cannot just mechanically look for the answer in the text, and where we care about a categorical class label at the document level.
    - e.g. document was ESL author vs. most important topic is the economy.

Supervised Learning with Text (continued)

- **Binary vs. Multiclass:** How many different categories you want to make predictions for.
  - Binary is most common, by far the easiest. Multi-class can be turned into binary problem.
  - Multiclass: the more classes you have, the harder the problem and the more training cases you need.

- **Key challenges with text**: high dimensionality, worse sense ambiguity.
  - Removing stop words and infrequent terms, **regularization**.
  - Be careful about preprocessing with respect to term ambiguity, with respect to prediction problem (e.g. same word has different meanings across contexts).
  - Generalization to new documents (vocabulary overlap).


Collect Documents
Collect New Documents
Preprocess Text Data
Evaluate Performance
Make Predictions on Unlabeled Data
Form Test/Train Splits
Train Classifier
Sanity Checks
Generate Training Data Labels
Use Predicted Labels
Active Learning

Constructing Training Data

- **Determine what to annotate**
  - Need a theory about what categories you expect.
  - Need enough examples from each category.
- **Formalize annotation instructions**
  - Write them down! Provide examples, talk through how to adjudicate unclear cases.
- **Perform pilot annotation**
  - Need to assess how quick it is to annotate a single document, and how much training a person needs to do the annotation.
  - Need to determine whether coders are able to agree on annotations.
- **Annotate data**
- **Assess inter-coder reliability**
  - If inter-coder reliability is unacceptably low, need to improve instructions, better coder training, recode data.
- **Release data**

Preprocessing Text

- Tradeoff between including lots of features which may improve performance, and tractability/speed.
- Generally want to remove terms that will not help distinguish classes.
- Very infrequent terms are often not useful.
- Including n-grams (1-3 is usually sufficient), phrases will typically improve performance.
- Think about terms that may be used in different contexts in different classes – may want to remove these terms or only include in n-grams.
- **Now is the time to experiment with different preprocessing specifications and see which ones yield highest accuracy/AUC**.
- **Can combine text data with metadata to train model**.

Classifiers

- There are many different classifiers out there – **no one best classifier for all use cases.**
- Text features are interpretable, so may want to prioritize models where we can assess feature importance.
- Number of distinct features will often be larger than number of documents, so you will want a classifier that can handle this.
  - **Regularization:** Penalty that shrinks most feature parameters to zero.
- It’s ok to try a few different classifiers, compare performance.
- LASSO (glmnet) and boosted trees (xgboost) are common classifiers used with text – handle large numbers of features well.
- **This is not a class about machine learning – I expect you have/will put in the time to learn how to use classifiers appropriately**.

Iteration and Active Learning

- Because we have a well defined problem (maximize classifier accuracy), it is ok to iterate, tweak things.
  - This only holds so long as you evaluate on held-out data/employ cross validation.
- **Active Learning:** train an classifier, classify new documents, use the newly classified documents as input to be coded and added to training data, repeat until classifier is stable.
  - Useful if you are trying to classify a rare class – can be hard to get enough positive cases. Using predictions to find new positive cases to code can iteratively improve model performance.
  - Want to keep going until you do not add many/any new cases.
  - Calculate accuracy on final test split.

Sanity Checks

- Look at results, spot check some predictions to see if they make sense on new data.
- Apply coding criteria to cases classifier found difficult.
  - Can feed back into Active Learning process.
  - Can report results as a robustness check.
- Come up with tests based on available metadata
  - Try to think of associations that should theoretically hold between your class labels and some other available metadata and check to see if they hold.
  - For example, if you expect that 5-10% of your documents should be in a particular class and you see 40%, need to investigate and square with your theory.
- Report all sanity checks in your write up.

Types of Classification Problems

- **Infrequent class(es) of interest:** The class(es) of documents that you care about identifying are relatively rare within the corpus.
  - Assess source of errors. May want to employ active learning.

- **Balanced classes**: All of the classes occur with reasonably similar frequency in the corpus.

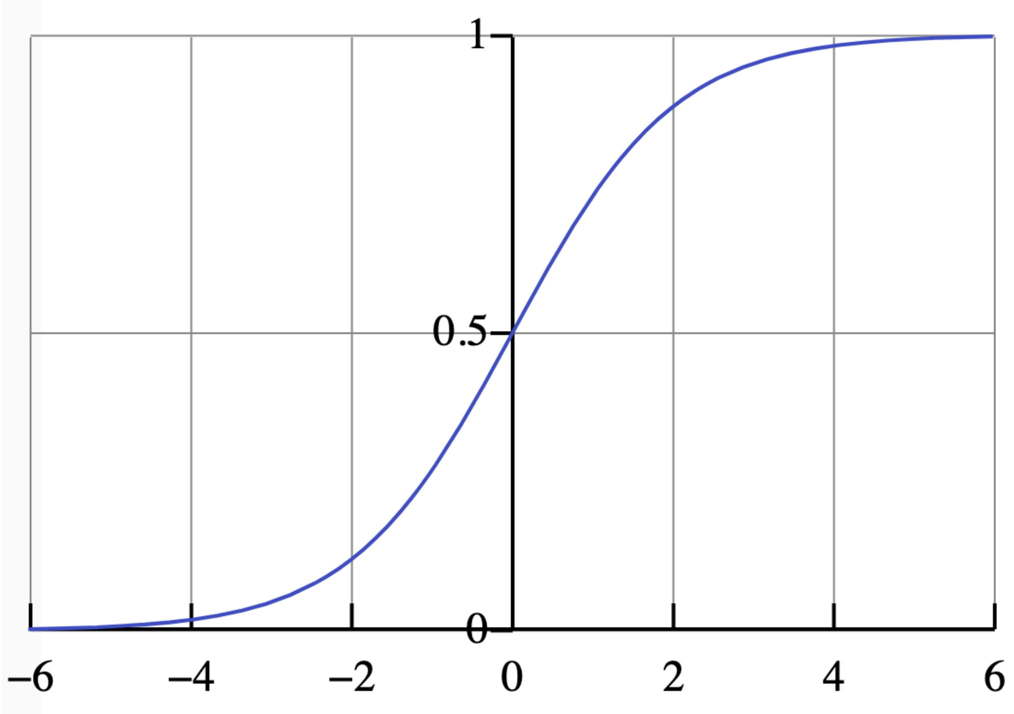
- Relative importance of **false positives vs. false negatives**.
  - False positives worse – favor high precision, low recall.
    Example: classifying a document as terrorist recruitment materials.
  - False negatives worse – favor low precision, high recall.
    Example: identifying hate speech for further review.

Assessing Classifier Performance

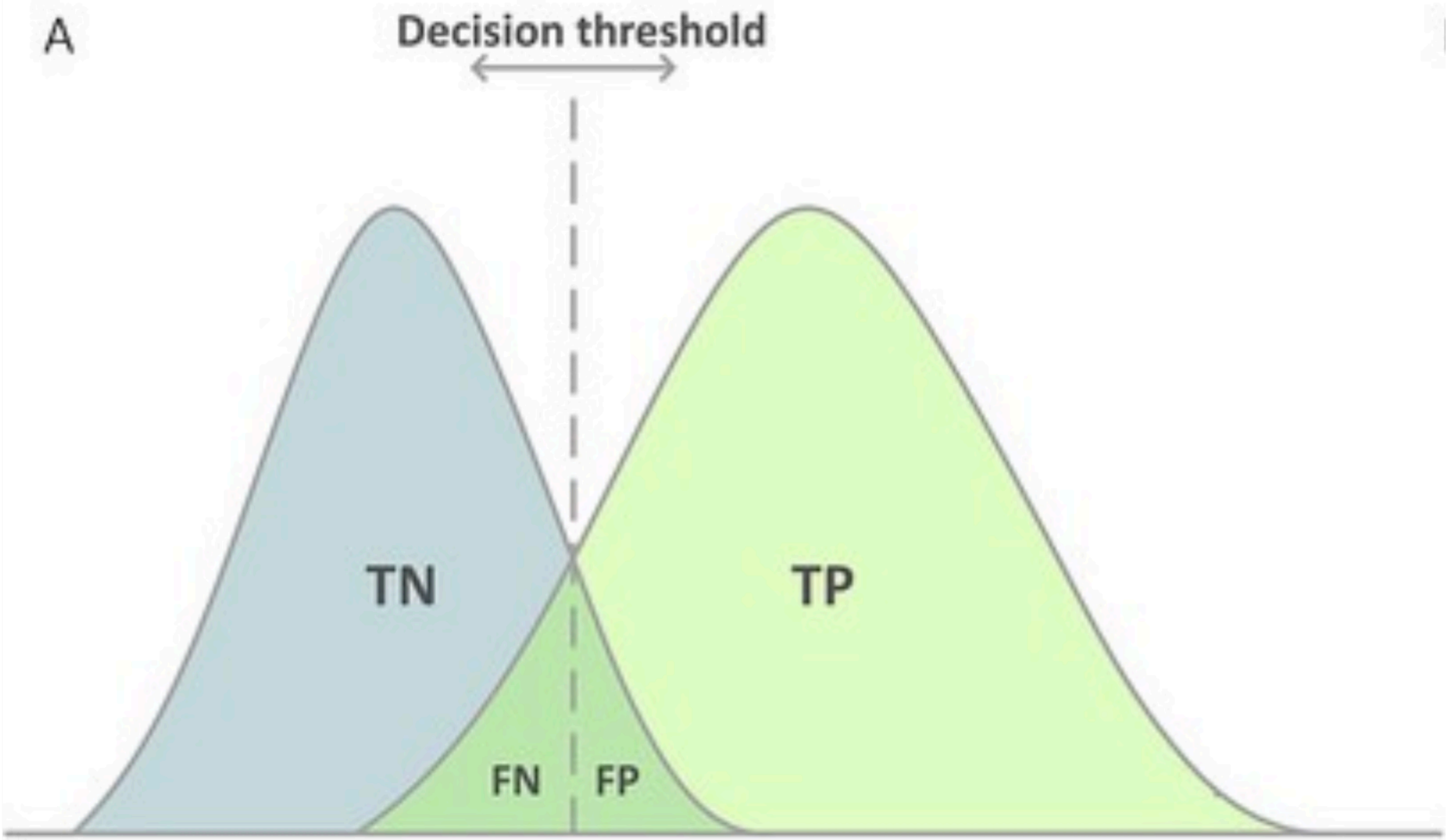
- **Metrics:** Accuracy, precision, recall, AUC, experiments.
  - Report/evaluate based on multiple measures.
  - Different applications → different points on P/R curve.
- **Held out test set, cross validation and generalization error:**
  - Always evaluate model performance on something other than training set.
- **Selecting a high quality training set:**
  - Enough cases, representative sample, labels not derived from input features.
  - Active learning as an option: repeated optimization.
- **Helpful resource**: http://brenocon.com/confusion_matrix_diagrams.pdf

Performance is Always Relative to a Threshold

- All classifiers output a score which can be transformed into a probability of being in a class (easy to see with logistic regression).
- In order to make 0, 1 predictions, we have to select a classification threshold.
- Choice of threshold can have significant impact on accuracy, precision, recall.



Performance Depends on Threshold Selection



Confusion Matrices:

- **Predicted vs. Observed**
- **True Positive:** prediction in class matches human label in class.
- **True Negative**: prediction not in class matches human label not in class.
- **False Positive**: prediction in class does not match human label not in class
- **False Negative**: prediction not in class does not match human label in class.

		pred	
		1	0
gold	1	True Pos	**False Neg**
	0	**False Pos**	True Neg

Accuracy, Precision, Recall

- **Accuracy:** prop. of predictions classifier got correct on the test set. Can also calculate accuracy on training set.

$$\mathrm{acc}(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \frac{1}{N}\sum_{i}^{N} \delta(y^{(i)} = \hat{y})$$

- **Recall:** the proportion of instances in the test set where the classifier correctly predicted that observations in the focal class were in that class.

- **Precision**: the proportion of focal class predictions that were correct.

$$\mathrm{RECALL}(\boldsymbol{y}, \hat{\boldsymbol{y}}, k) = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}$$

$$\mathrm{PRECISION}(\boldsymbol{y}, \hat{\boldsymbol{y}}, k) = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}.$$

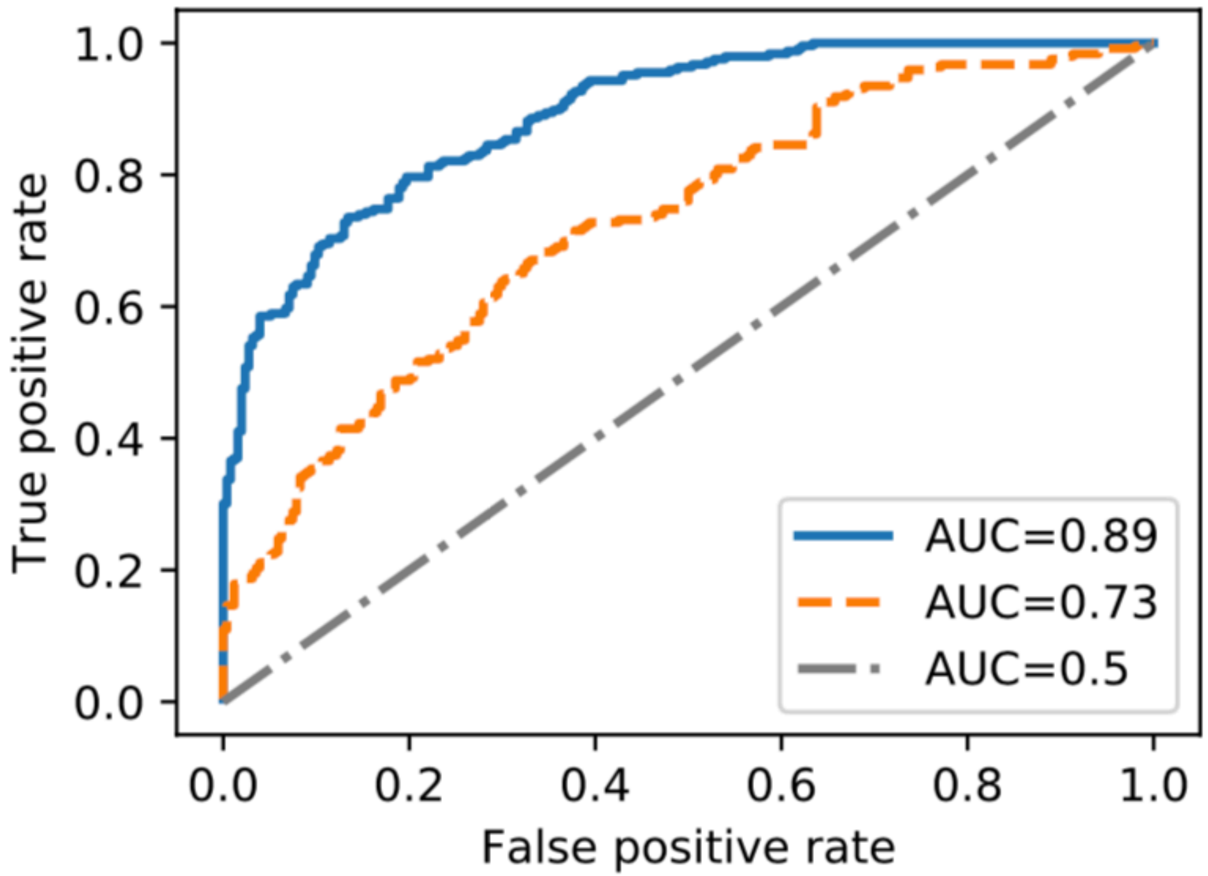
F-score

- An F score is just the harmonic mean of precision (p) and recall (r) for a given classifier.
- Will take a maximum value of 1 when precision and recall are both 1.
- Can also calculate a macro-F measure in the multiclass case.
  - For each class, make that the focal class and calculate precision + recall with some given threshold.
  - Take average across these individual class F measures.

$$F\text{-}\textsc{measure}(\boldsymbol{y}, \hat{\boldsymbol{y}}, k) = \frac{2rp}{r+p} \qquad \text{Macro-}F(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \frac{1}{|\mathcal{K}|}\sum_{k\in\mathcal{K}} F\text{-}\textsc{measure}(\boldsymbol{y}, \hat{\boldsymbol{y}}, k)$$

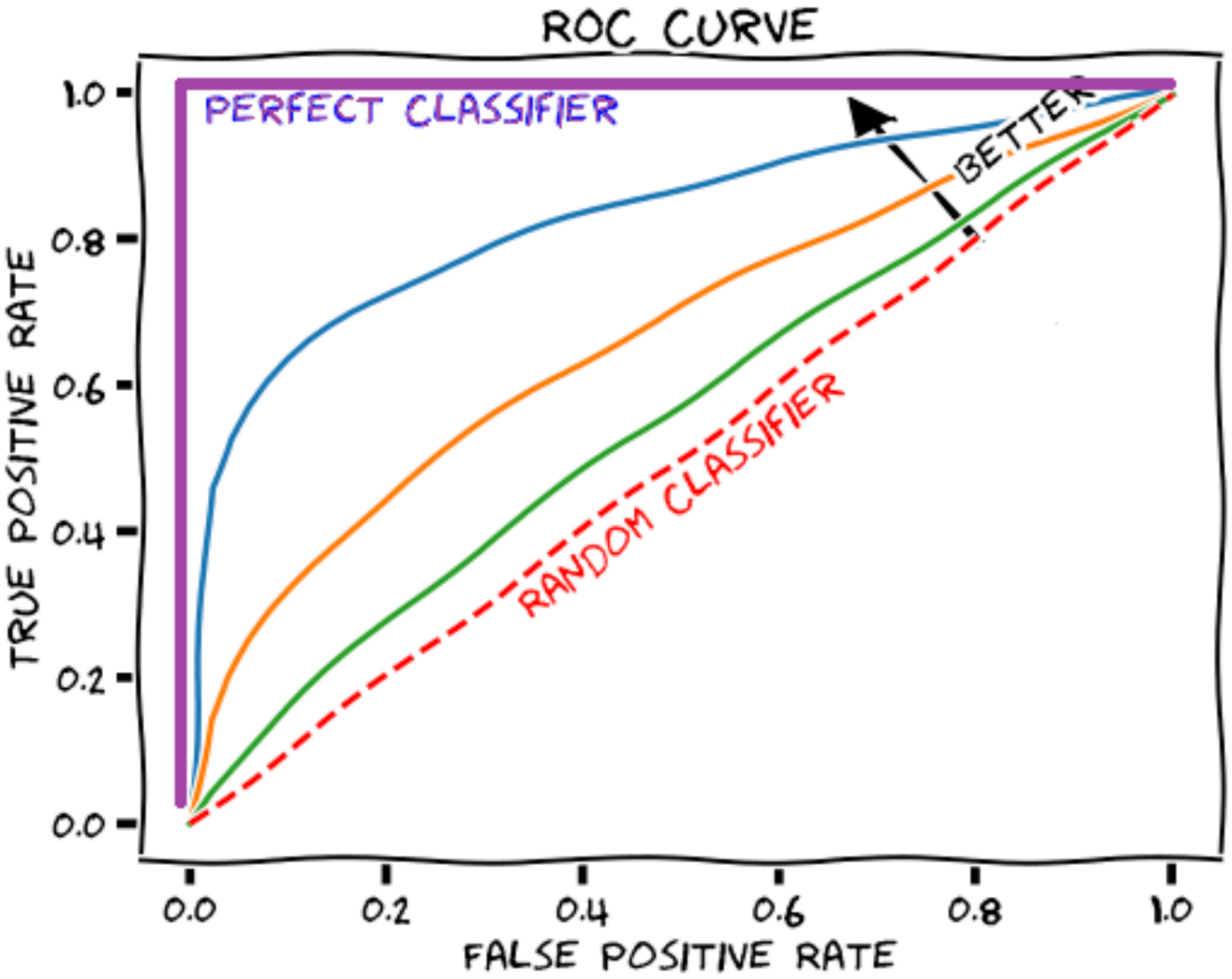
Classification Thresholds → ROC and AUC

- We can vary the classification threshold from zero to 1 to generate a receiver operating characteristic curve (ROC) curve.
- Shows tradeoff between TPR and FPR.
- Area Under the Curve (AUC) is literally the integral of the ROC curve (ranges from 0 to 1).



ROC, AUC and Performance

- A perfect classifier will have an AUC of 1 – can predict 100% of true positives with zero false positives, for some threshold.
- Random classifier will have AUC of 0.5 – TPR and FPR will track each other ~1:1.
- Higher AUC is better.



Tradeoffs

- **Precision-Recall Tradeoff:** Except in the case of a perfect classifier, setting a classification threshold to increase recall will lower precisions, and vice versa.
  - Need to think about your application and the relative costs.

- **Feature Complexity:** Adding tons of features (e.g. 1-12 grams) will improve in-sample accuracy, may improve held out accuracy, but may also lead to greater generalization error.
  - How likely are the terms I am using to appear in the unseen documents I am classifying.
  - Will they have the same meaning?

General Takeaways

- Supervised learning with text is like supervised learning with any other features, ours just happen to be counts of words.
- Optimize text preprocessing towards accuracy/AUC.
- Iterative process -- can take the form of active learning.
- No one universally optimal classifier.
- Validation and replicability are key!
    - Document the process you went through to arrive at final specification.
    - Clearly lay out coding guidelines – with theory ahead of time.
    - Checks to determine if your results “make sense”, comparison to other approaches, metadata.