Natural Language Processing

Text classification

Yulia Tsvetkov

yuliats@cs.washington.edu

PAUL G. ALLEN SCHOOL
OF COMPUTER SCIENCE & ENGINEERING

Announcements

- HW1 overview
  - https://gitlab.cs.washington.edu/cse447-au22/internal/assignment-1-public-ready/-/blob/main/pset1.ipynb
- Quiz 1 is on Wednesday

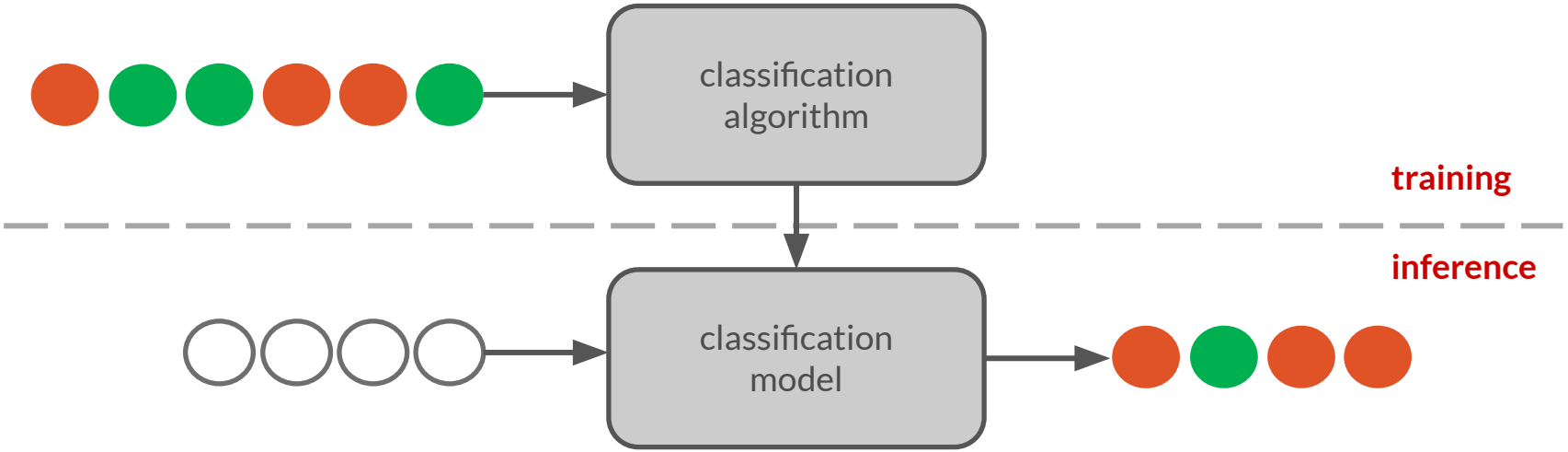
Compute resources for completing HW

- Access to non-CS students
  - Response from CSE support: Based on day 5 enrollment, I have finished catching up on this, this morning. Students should check their email, maybe spam folder for the CSE account email. There was a user or 2 that previously had a non-major account that I reactivated and they may need to reset their password at password.cs.washington.edu. If anyone has not had an account created (very late add, etc.) please let me know and I'll make the accounts for them.
- Google cloud credits
  - Email TAs

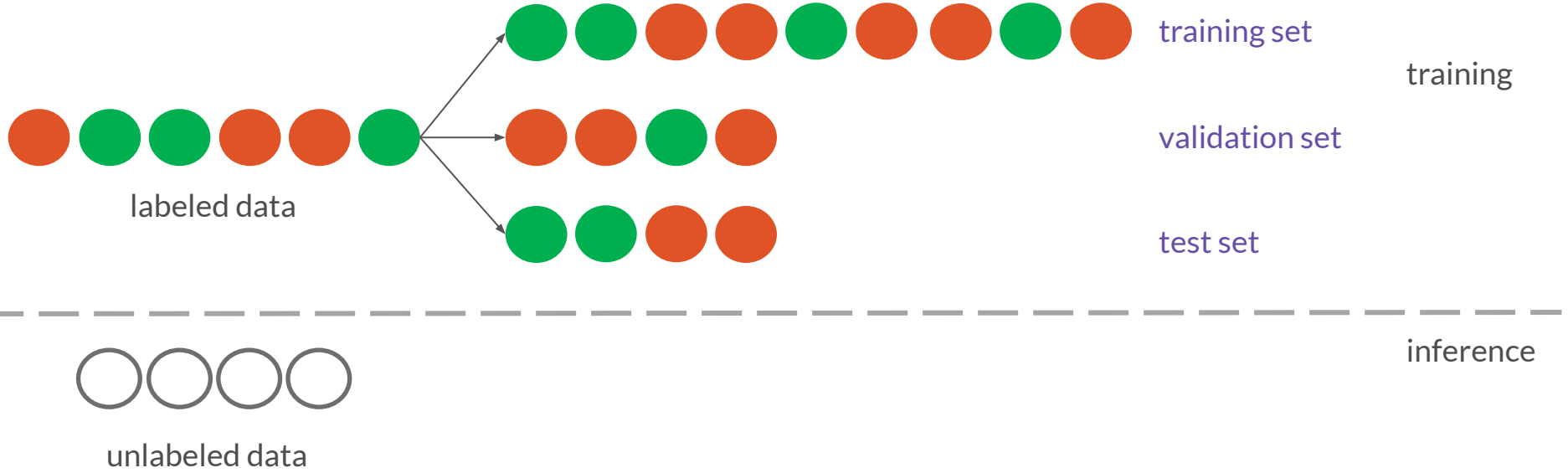
Readings

- Eis 2 https://github.com/jacobeisenstein/gt-nlp-class/blob/master/notes/eisenstein-nlp-notes.pdf
- J&M III 4 https://web.stanford.edu/~jurafsky/slp3/4.pdf
- Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up? Sentiment Classification using Machine Learning Techniques. In Proceedings of EMNLP, 2002
- Andrew Y. Ng and Michael I. Jordan, On discriminative vs. generative classifiers: A comparison of logistic regression and naive Bayes, In Proceedings of NeurIPS, 2001.

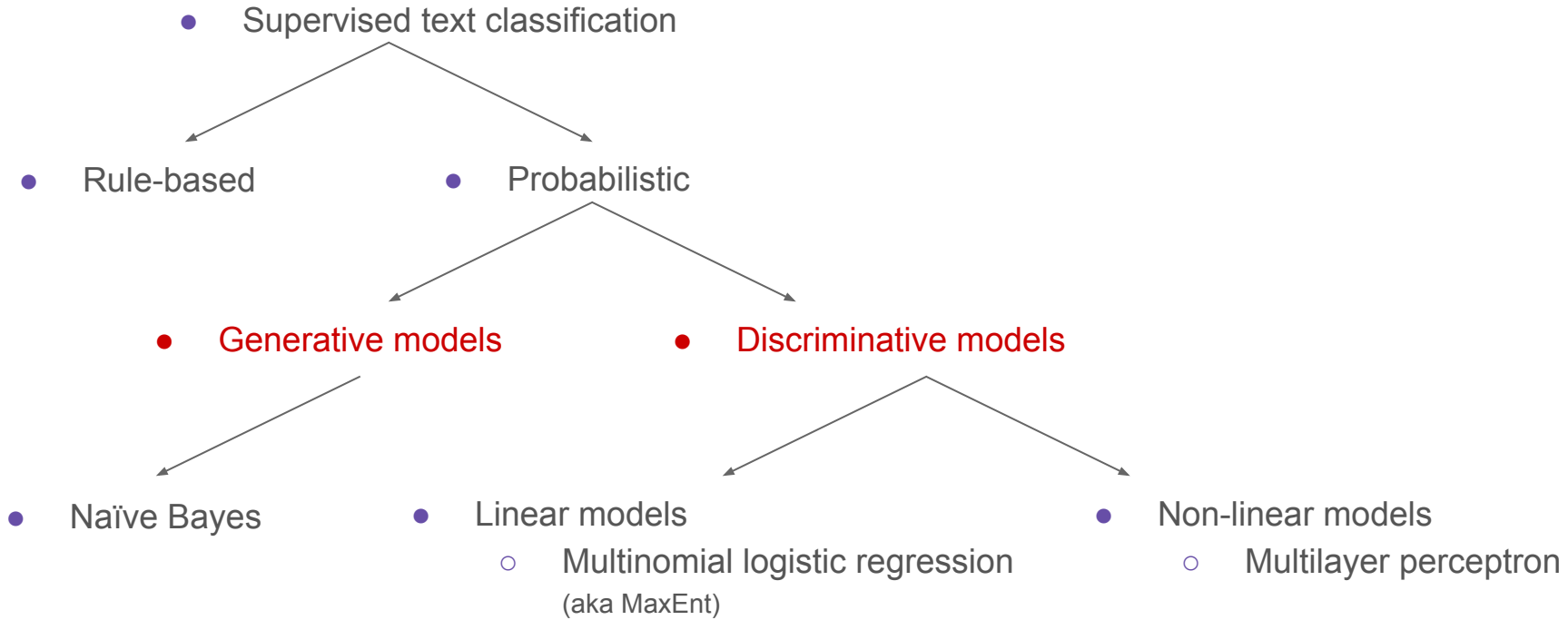
Supervised classification



Training, validation, and test sets



We consider alternative models for classification



Generative and discriminative models

imagenet

imagenet

Generative model

- Build a model of what's in a cat image
  - Knows about whiskers, ears, eyes
  - Assigns a probability to any image:
  - how cat-y is this image?
- Also build a model for dog images

imagenet

imagenet

Now given a new image:

Run both models and see which one fits better

Discriminative model

Just try to distinguish dogs from cats

Oh look, dogs have collars! Let's ignore everything else

Generative and discriminative models

- Generative model: a model that calculates the probability of the input data itself

$$P(X, Y)$$

joint

- Discriminative model: a model that calculates the probability of a latent trait given the data

$$P(Y \mid X)$$

conditional

Generative and discriminative models

- Generative text classification: Learn a model of the joint P(X, y), and find

$$\hat{y} = \underset{\tilde{y}}{\text{argmax}}\ P(X, \tilde{y})$$

- Discriminative text classification: Learn a model of the conditional P(y | X), and find

$$\hat{y} = \underset{\tilde{y}}{\text{argmax}}\ P(\tilde{y}|X)$$

Andrew Y. Ng and Michael I. Jordan, On discriminative vs. generative classifiers: A comparison of logistic regression and naive Bayes, Advances in Neural Information Processing Systems 14 (NIPS), 2001.

Finding the correct class c from a document d in Generative vs Discriminative Classifiers

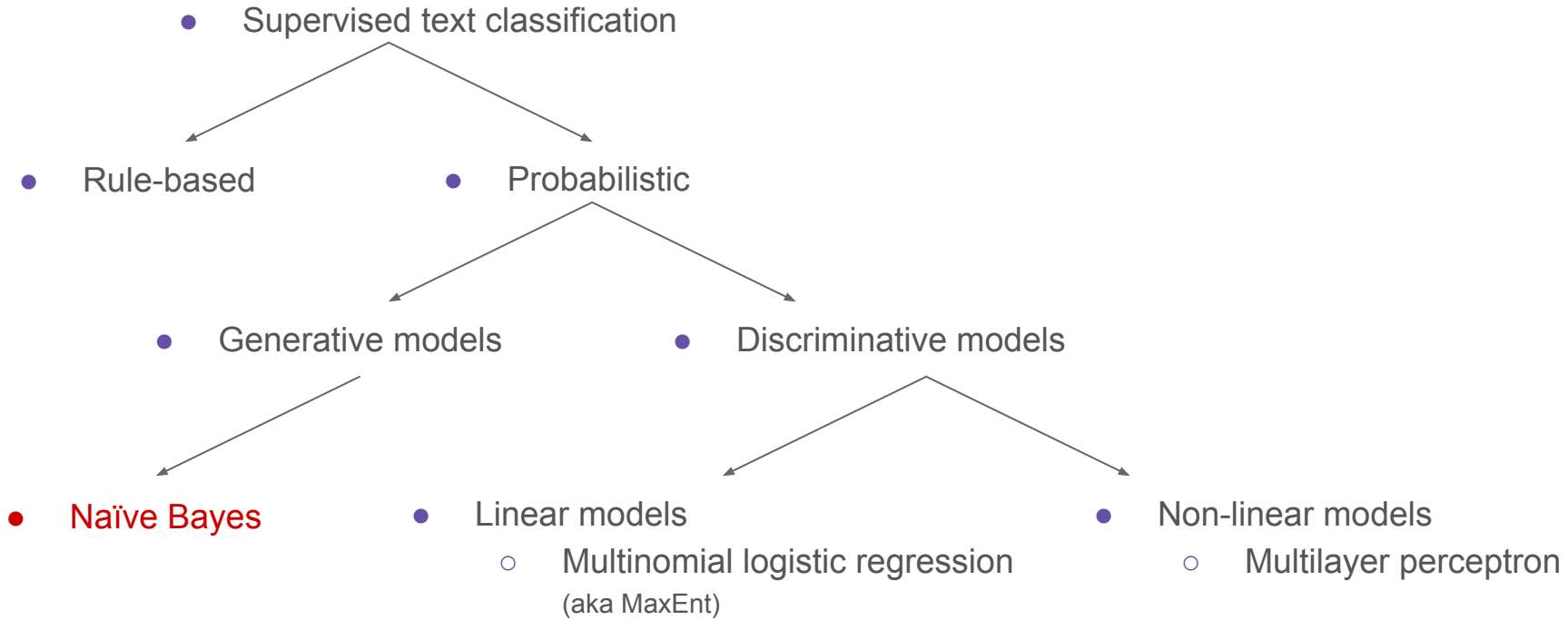
- Naive Bayes

$$\hat{c} = \underset{c \in C}{\text{argmax}} \overbrace{P(d|c)}^{\text{likelihood}} \overbrace{P(c)}^{\text{prior}}$$

- Logistic Regression

$$\hat{c} = \underset{c \in C}{\text{argmax}} \overbrace{P(c|d)}^{\text{posterior}}$$

We'll consider alternative models for classification



Generative text classification: Naïve Bayes

$$\mathrm{C}_{NB} = \underset{c}{\operatorname{argmax}} P(c|d) = \underset{c}{\operatorname{argmax}} \frac{P(d|c)P(c)}{P(d)} \propto$$ Bayes rule

$$\underset{c}{\operatorname{argmax}} P(d|c)P(c) =$$ same denominator

$$\underset{c}{\operatorname{argmax}} P(w_1, w_2, \ldots, w_n|c)P(c) =$$ representation

$$\underset{c_j}{\operatorname{argmax}}\ P(c_j) \prod_i P(w_i|c)$$ conditional independence

Underflow prevention: log space

- Multiplying lots of probabilities can result in floating-point underflow
- Since log(xy) = log(x) + log(y)
  - better to sum logs of probabilities instead of multiplying probabilities
- Class with highest un-normalized log probability score is still most probable

$$\mathrm{C}_{NB} = \underset{c_j}{\operatorname{argmax}}\ P(c_j) \prod_i P(w_i|c)$$

$$\mathrm{C}_{NB} = \underset{c_j}{\operatorname{argmax}}\ log(P(c_j)) + \sum_i log(P(w_i|c))$$

- Model is now just max of sum of weights

Learning the multinomial naïve Bayes

- How do we learn (train) the NB model?

Learning the multinomial naïve Bayes

- How do we learn (train) the NB model?
- We learn P(c) and $P(w_i|c)$ from training (labeled) data

$$C_{NB} = \underset{c_j}{\text{argmax}}\ log(P(c_j)) + \sum_i log(P(w_i|c))$$

Parameter estimation for NB

- Parameter estimation during training
- Concatenate all documents with category c into one mega-document
- Use the frequency of w_i in the mega-document to estimate the word probability

$$C_{NB} = \underset{c_j}{\text{argmax}}\ log(P(c_j)) + \sum_i log(P(w_i|c))$$

$$\hat{P}(c_j) = \frac{doccount(C = c_j)}{N_{doc}}$$

$$\hat{P}(w_i|c_j) = \frac{count(w_i, c_j)}{\sum_{w \in V} count(w, c_j)}$$

Parameter estimation for NB

$$\hat{P}(w_i|c_j) = \frac{count(w_i, c_j)}{\sum_{w \in V} count(w, c_j)}$$

- fraction of times word w_i appears among all words in documents of topic c_j

- Create mega-document for topic j by concatenating all docs in this topic
  - Use frequency of w in mega-document

Problem with Maximum Likelihood

- What if we have seen no training documents with the word “fantastic” and classified in the topic positive?

Problem with Maximum Likelihood

- What if we have seen no training documents with the word “fantastic” and classified in the topic positive?

$$\hat{P}(\text{“}fantastic\text{”}|c = \text{positive}) = \frac{count(\text{“}fantastic\text{”}, \text{positive})}{\sum_{w \in V} count(w, \text{positive})} = 0$$

- Zero probabilities cannot be conditioned away, no matter the other evidence!

$$\underset{c_j}{\text{argmax}}\ P(c_j) \prod_i P(w_i|c)$$

Laplace (add-1) smoothing for naïve Bayes

$$\hat{P}(w_i|c_j) = \frac{count(w_i, c_j) + 1}{\sum_{w \in V}(count(w, c_j) + 1)}$$

Laplace (add-1) smoothing for naïve Bayes

$$\hat{P}(w_i|c_j) = \frac{count(w_i, c_j) + 1}{\sum_{w \in V}(count(w, c_j) + 1)}$$

$$= \frac{count(w_i, c_j) + 1}{(\sum_{w \in V}(count(w, c_j)) + |V|}$$

- Note about log space

Example

	Doc	Words	Class
Training	1	Chinese Beijing Chinese	c
	2	Chinese Chinese Shanghai	c
	3	Chinese Macao	c
	4	Tokyo Japan Chinese	j
Test	5	Chinese Chinese Chinese Tokyo Japan	?

Example

	Doc	Words	Class
Training	1	Chinese Beijing Chinese	c
	2	Chinese Chinese Shanghai	c
	3	Chinese Macao	c
	4	Tokyo Japan Chinese	j
Test	5	Chinese Chinese Chinese Tokyo Japan	?

$$\hat{P}(c) = \frac{N_c}{N}$$

Priors:

$P(c) = \frac{3}{4}$

$P(j) = \frac{1}{4}$

Example

	Doc	Words	Class
Training	1	Chinese Beijing Chinese	c
	2	Chinese Chinese Shanghai	c
	3	Chinese Macao	c
	4	Tokyo Japan Chinese	j
Test	5	Chinese Chinese Chinese Tokyo Japan	?

$$\hat{P}(c) = \frac{N_c}{N}$$

$$\hat{P}(w \mid c) = \frac{count(w,c)+1}{count(c)+|V|}$$

Priors:

$P(c) = \frac{3}{4}$

$P(j) = \frac{1}{4}$

Conditional Probabilities:

P(Chinese|*c*) = (5+1) / (8+6) = 6/14 = 3/7

P(Tokyo|*c*) = (0+1) / (8+6) = 1/14

P(Japan|*c*) = (0+1) / (8+6) = 1/14

P(Chinese|*j*) = (1+1) / (3+6) = 2/9

P(Tokyo|*j*) = (1+1) / (3+6) = 2/9

P(Japan|*j*) = (1+1) / (3+6) = 2/9

Example

	Doc	Words	Class
Training	1	Chinese Beijing Chinese	c
	2	Chinese Chinese Shanghai	c
	3	Chinese Macao	c
	4	Tokyo Japan Chinese	j
Test	5	Chinese Chinese Chinese Tokyo Japan	?

$$\hat{P}(c) = \frac{N_c}{N} \qquad \hat{P}(w \mid c) = \frac{count(w,c)+1}{count(c)+|V|}$$

Priors:

$P(c) = \frac{3}{4}$

$P(j) = \frac{1}{4}$

Conditional Probabilities:

P(Chinese|*c*) = (5+1) / (8+6) = 6/14 = 3/7
P(Tokyo|*c*) = (0+1) / (8+6) = 1/14
P(Japan|*c*) = (0+1) / (8+6) = 1/14
P(Chinese|*j*) = (1+1) / (3+6) = 2/9
P(Tokyo|*j*) = (1+1) / (3+6) = 2/9
P(Japan|*j*) = (1+1) / (3+6) = 2/9

Choosing a class:

$P(c|d5) \propto 3/4 * (3/7)^3 * 1/14 * 1/14 \approx 0.0003$

$P(j|d5) \propto 1/4 * (2/9)^3 * 2/9 * 2/9 \approx 0.0001$

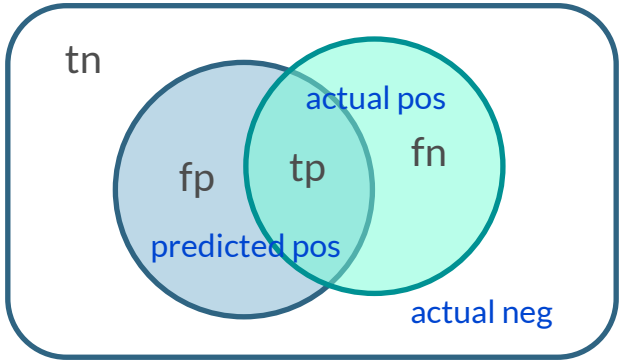
Summary: naïve Bayes is not so naïve

- Naïve Bayes is a probabilistic model
- Naïve because it assumes features are independent of each other for a class
- Very fast, low storage requirements
- Robust to Irrelevant Features
- Very good in domains with many equally important features
- Optimal if the independence assumptions hold: If assumed independence is correct, then it is the Bayes Optimal Classifier for problem
- A good dependable baseline for text classification
  - But we will see other classifiers that give better accuracy

Classification evaluation

- Contingency table: model's predictions are compared to the correct results
  - a.k.a. confusion matrix

	actual pos	actual neg
predicted pos	true positive (tp)	false positive (fp)
predicted neg	false negative (fn)	true negative (tn)



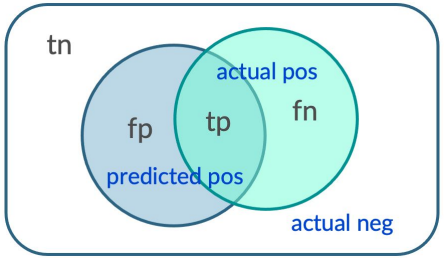
Classification evaluation

- Borrowing from Information Retrieval, empirical NLP systems are usually evaluated using the notions of precision and recall

Classification evaluation

- Precision (P) is the proportion of the selected items that the system got right in the case of text categorization
  - it is the % of documents classified as “positive” by the system which are indeed “positive” documents
- Reported per class or average

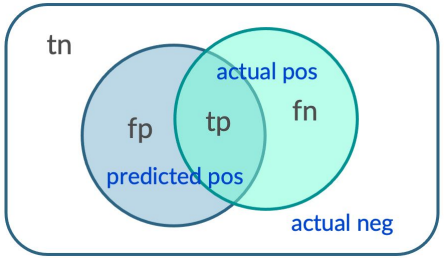
$$\text{precision} = \frac{true\ positives}{true\ positives + false\ positives} = \frac{tp}{tp + fp}$$



Classification evaluation

- Recall (R) is the proportion of actual items that the system selected in the case of text categorization
  - it is the % of the “positive” documents which were actually classified as “positive” by the system
- Reported per class or average

$$\text{recall} = \frac{true\ positives}{true\ positives + false\ negatives} = \frac{tp}{tp + fn}$$



Classification evaluation

- We often want to trade-off precision and recall
  - typically: the higher the precision the lower the recall
  - can be plotted in a precision-recall curve
- It is convenient to combine P and R into a single measure
  - one possible way to do that is F measure

$$F_\beta = \frac{(\beta^2+1)PR}{\beta^2 P+R} \quad \text{for } \beta=1,\ F_1 = \frac{2PR}{P+R}$$

Classification evaluation

- Additional measures of performance: accuracy and error
  - accuracy is the proportion of items the system got right
  - error is its complement

	actual pos	actual neg
predicted pos	true positive (tp)	false positive (fp)
predicted neg	false negative (fn)	true negative (tn)

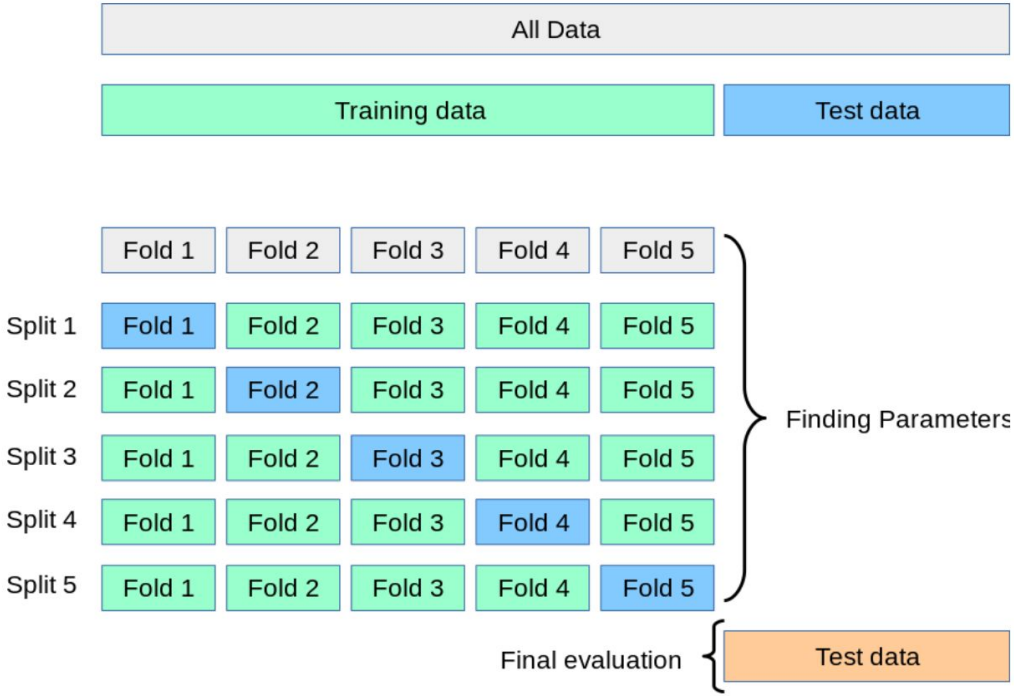
$$\text{accuracy} = \frac{tp+tn}{tp+fp+tn+fn}$$

Micro- vs. macro-averaging

If we have more than one class, how do we combine multiple performance measures into one quantity?

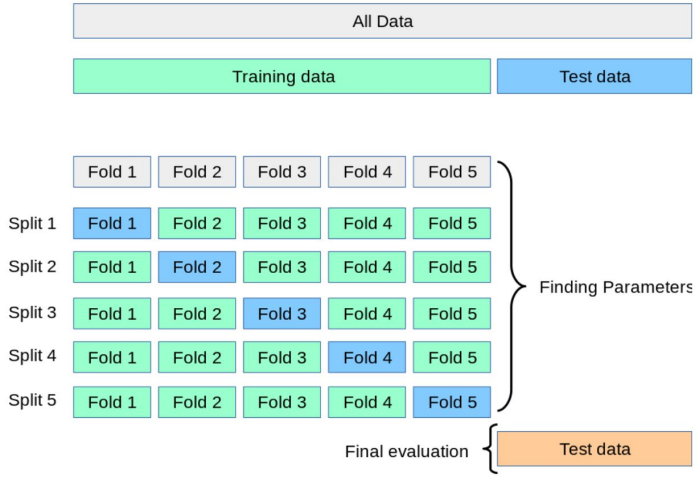
- Macroaveraging
  - Compute performance for each class, then average.
- Microaveraging
  - Collect decisions for all classes, compute contingency table, evaluate.

K-fold cross-validation

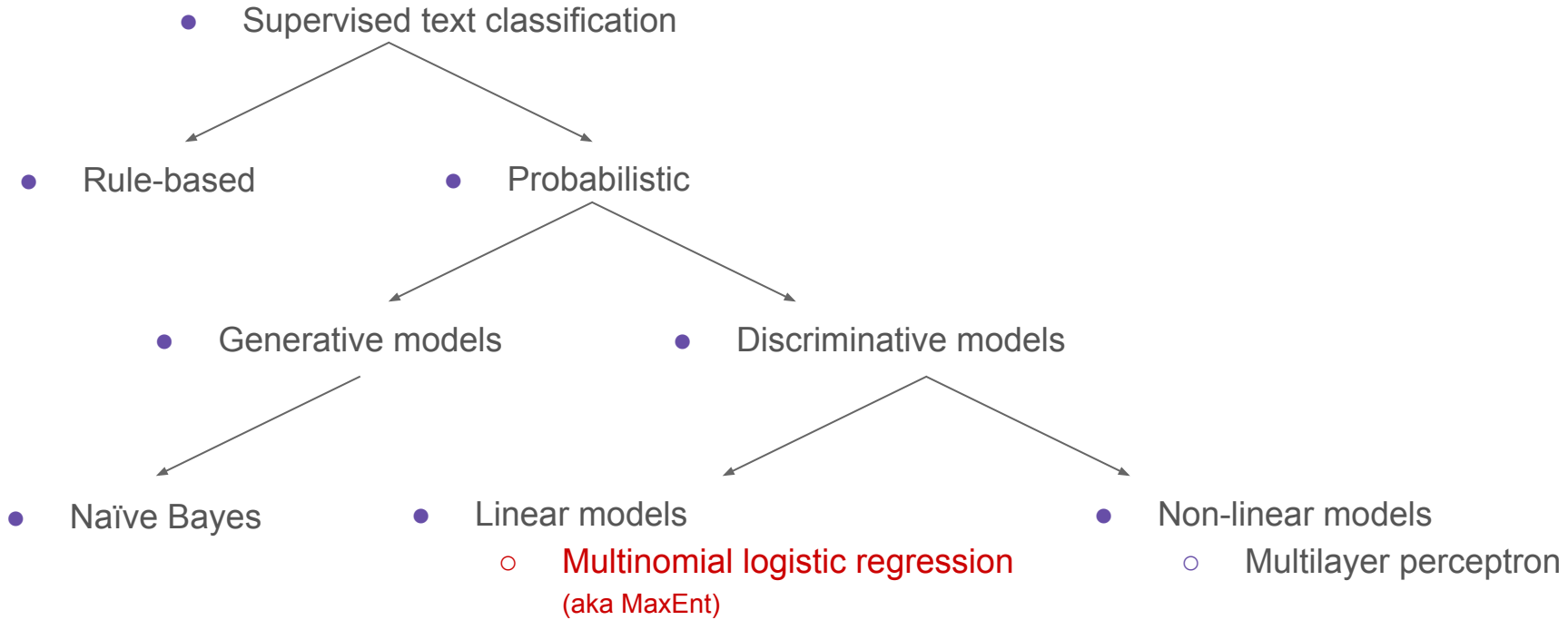


K-fold cross-validation

- Metric: P/R/F1 or Accuracy
- Unseen test set
  - avoid overfitting (‘tuning to the test set’)
  - more conservative estimate of performance
- Cross-validation over multiple splits
    - Handles sampling errors from different datasets
  - Pool results over each split
  - Compute pooled dev set performance



Next class: Logistic regression



Logistic regression classifier

- Important analytic tool in natural and social sciences
- Baseline supervised machine learning tool for classification
- Is also the foundation of neural networks