Natural Language Processing

Text classification

Yulia Tsvetkov

yuliats@cs.washington.edu

PAUL G. ALLEN SCHOOL
OF COMPUTER SCIENCE & ENGINEERING

Announcements

- HW1 is out, please start early!
  - Han will describe it in the last 10 minutes of the class
  - Use OHs to ask questions
- First quiz - next Wednesday, will cover introduction and text classification
  - We'll practice taking quizzes in Friday

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.

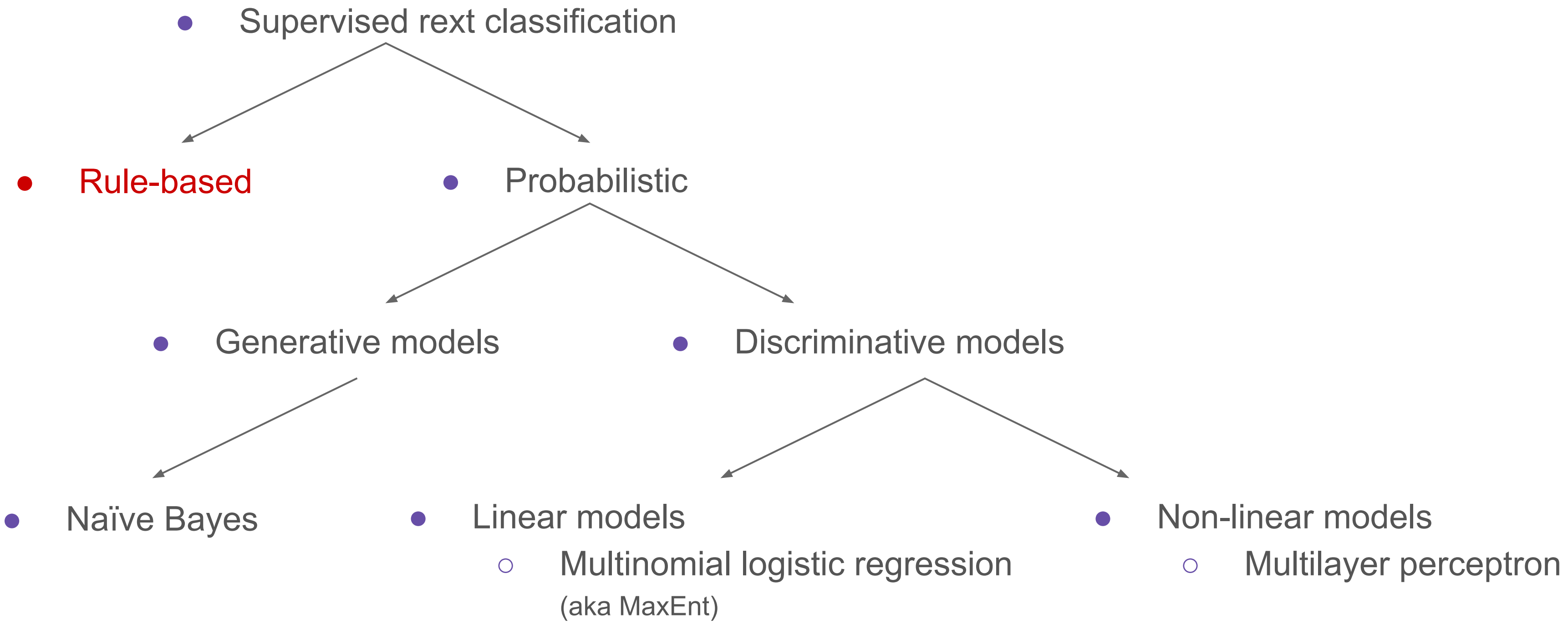
Text classification

- We might want to categorize the content of the text:
    - Spam detection (binary classification: spam/not spam)
    - Sentiment analysis (binary or multiway)
        - movie, restaurant, product reviews (pos/neg, or 1-5 stars)
        - political argument (pro/con, or pro/con/neutral)
        - Topic classification (multiway: sport/finance/travel/etc)
    - Language Identification (multiway: languages, language families)
    - …
- Or we might want to categorize the author of the text (authorship attribution)
    - Human- or machine generated?
    - Native language identification (e.g., to tailor language tutoring)
    - Diagnosis of disease (psychiatric or cognitive impairments)
    - Identification of gender, dialect, educational background, political orientation (e.g., in forensics [legal matters], advertising/marketing, campaigning, disinformation)
    - …

Classification: learning from data

- Supervised
  - labeled examples
    - Binary (true, false)
    - Multi-class classification (politics, sports, gossip)
    - Multi-label classification (#party #FRIDAY #fail)
- Unsupervised
  - no labeled examples
- Semi-supervised
  - labeled examples + non-labeled examples
- Weakly supervised
  - heuristically-labeled examples

We’ll consider alternative models for classification



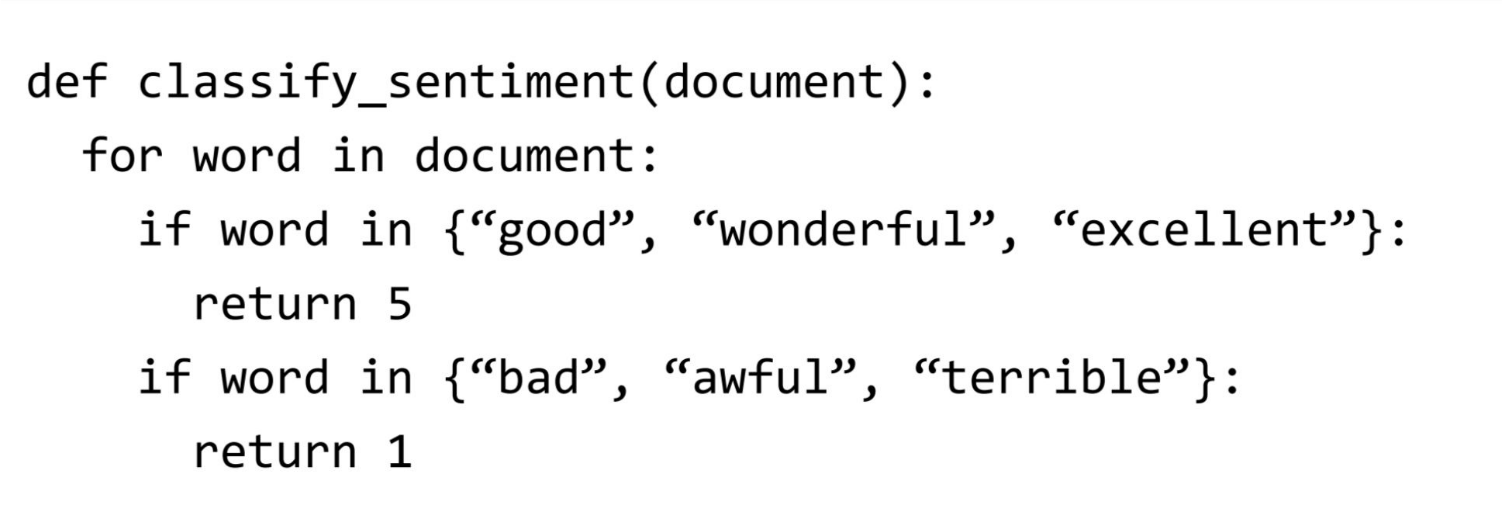
- Supervised rext classification

- Rule-based
- Probabilistic

- Generative models
- Discriminative models

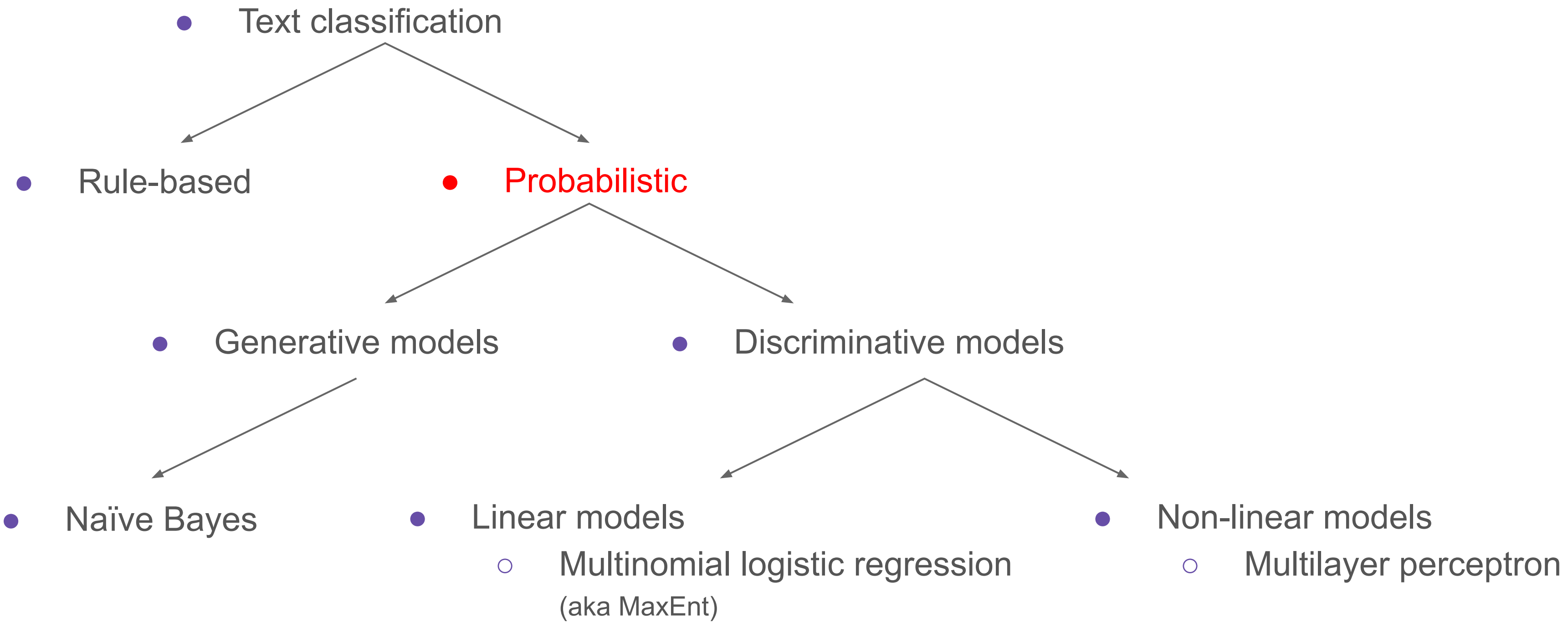
- Naïve Bayes
- Linear models
  - Multinomial logistic regression (aka MaxEnt)
- Non-linear models
  - Multilayer perceptron

Rule-based classifier



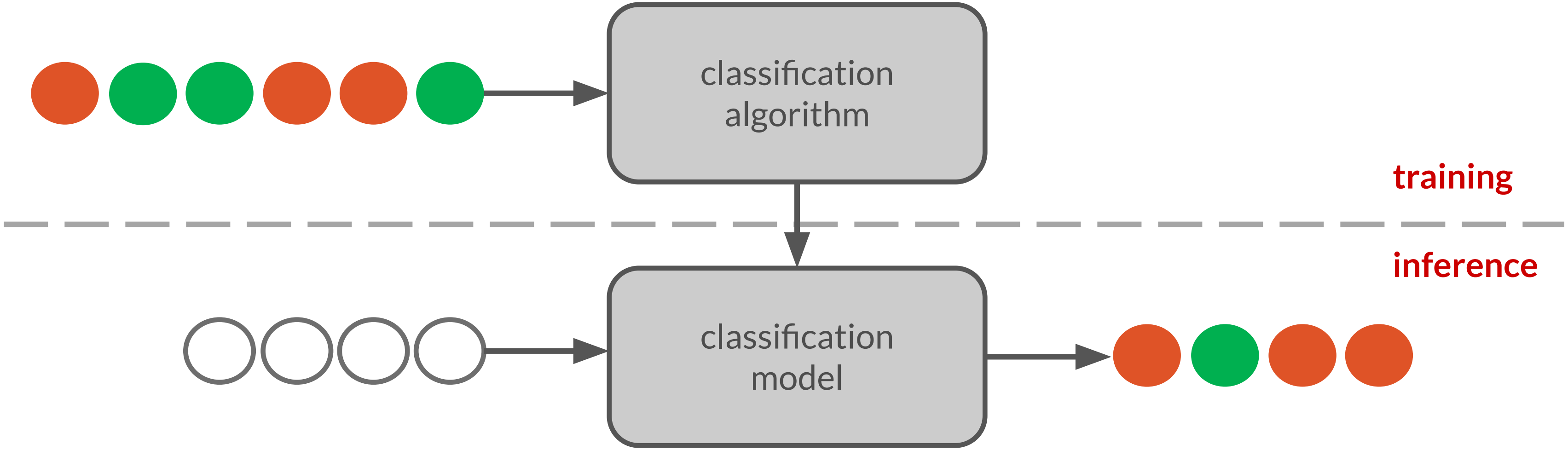
```
def classify_sentiment(document):
  for word in document:
    if word in {“good”, “wonderful”, “excellent”}:
      return 5
    if word in {“bad”, “awful”, “terrible”}:
      return 1
```

We’ll consider alternative models for classification



- Text classification
- Rule-based
- Probabilistic
- Generative models
- Discriminative models
- Naïve Bayes
- Linear models
  - Multinomial logistic regression (aka MaxEnt)
- Non-linear models
  - Multilayer perceptron

Supervised classification



Supervised classification: formal setting

- Learn a classification model from labeled data on
  - properties ("features") and their importance ("weights")
- X: set of attributes or features $\{x_1, x_2, \ldots, x_n\}$
  - e.g. word counts extracted from an input documents
- y: a "class" label from the label set $Y = \{y_1, y_2, \ldots, y_k\}$
  - e.g., spam/not spam, positive/negative/neutral

Supervised classification: formal setting

- Learn a classification model from labeled data on
  - properties (“features”) and their importance (“weights”)
- X: set of attributes or features $\{x_1, x_2, \ldots, x_n\}$
  - e.g. word counts extracted from an input documents
- y: a “class” label from the label set $Y = \{y_1, y_2, \ldots, y_k\}$
  - e.g., spam/not spam, positive/negative/neutral

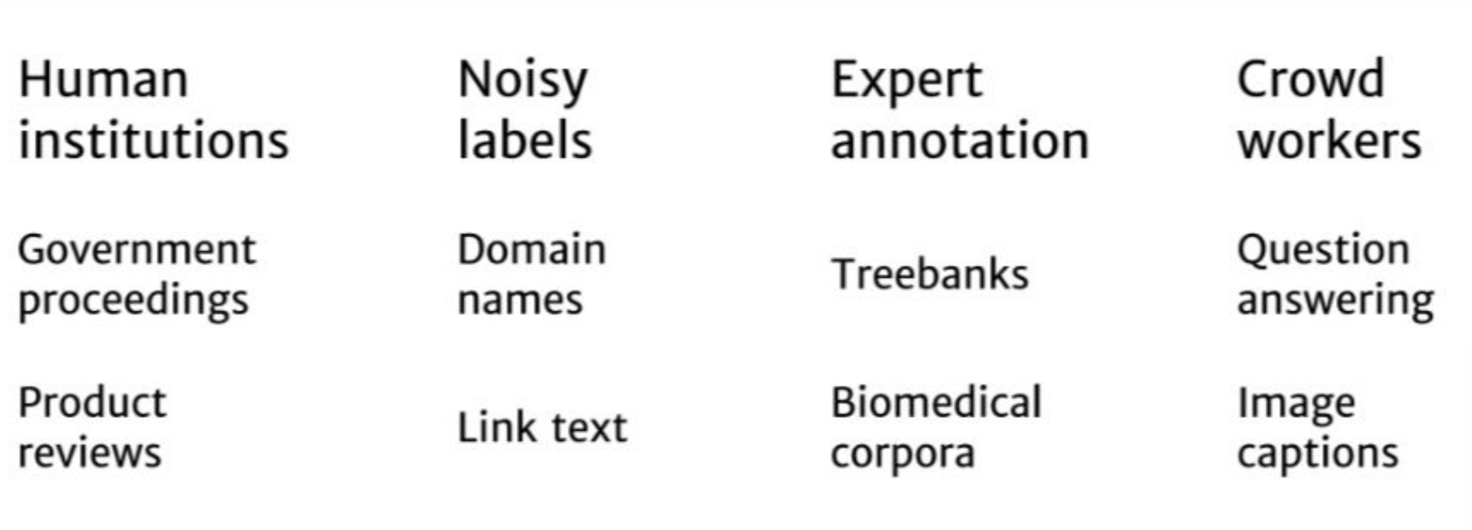
- Given data samples $\{x_1, x_2, \ldots, x_n\}$ and corresponding labels $Y = \{y_1, y_2, \ \ldots, y_k\}$
- We train a function $f: x \in X \rightarrow y \in Y$ (the model)

Supervised classification: formal setting

- Learn a classification model from labeled data on
  - properties (“features”) and their importance (“weights”)
- X: set of attributes or features $\{x_1, x_2, \ldots, x_n\}$
  - e.g. word counts extracted from an input documents
- y: a “class” label from the label set $Y = \{y_1, y_2, \ldots, y_k\}$
  - e.g., spam/not spam, positive/negative/neutral

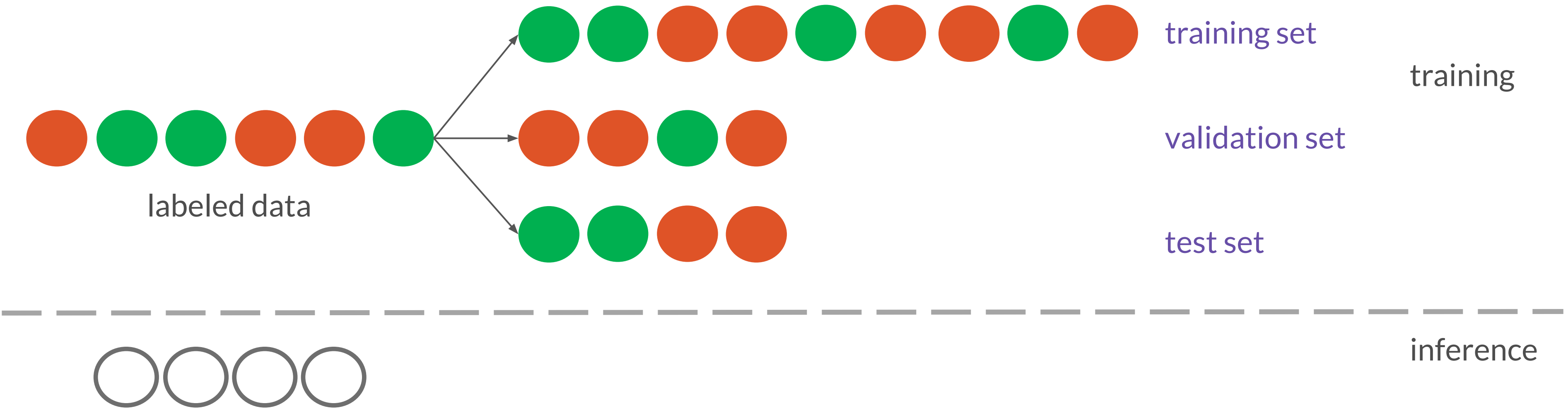
- At inference time, apply the model on new instances to predict the label

Where do datasets come from?



Human institutions	Noisy labels	Expert annotation	Crowd workers
Government proceedings	Domain names	Treebanks	Question answering
Product reviews	Link text	Biomedical corpora	Image captions

Training, validation, and test sets



training set

training

validation set

labeled data

test set

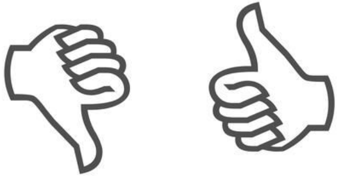
inference

unlabeled data

Text classification – feature extraction

What can we measure over text? Consider this movie review:

I love this movie! It’s sweet, but with satirical humor. The dialogue is great, and the adventure scenes are fun… It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it just to about anyone. I’ve seen it several times, and I’m always happy to see it again whenever I have a friend who hasn’t seen it before.



Text classification – feature extraction

What can we measure over text? Consider this movie review:

I love this movie! It's sweet, but with satirical humor. The dialogue is great, and the adventure scenes are fun… It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it just to about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it before.

Text classification – feature extraction

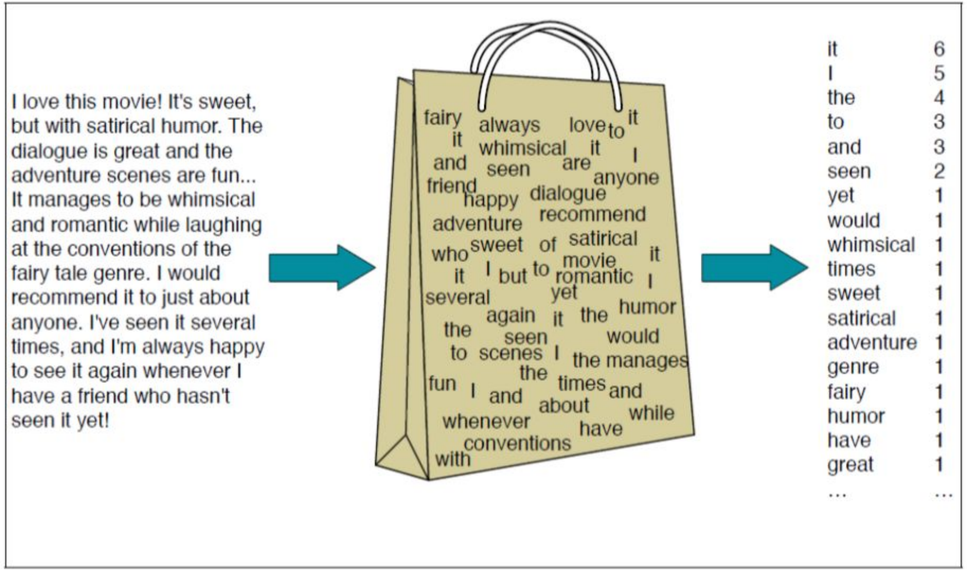
I love this movie! It's sweet, but with satirical humor. The dialogue is great, and the adventure scenes are fun… It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it just to about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it before.

(almost) the entire lexicon

word	count	relative frequency
love	10	0.0007
great	…	
recommend		
laugh		
happy		
…		
several		
boring		
…		

Bag-of-Words (BOW)

- Given a document d (e.g., a movie review) – how to represent d ?



I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun... It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet!

it	6
I	5
the	4
to	3
and	3
seen	2
yet	1
would	1
whimsical	1
times	1
sweet	1
satirical	1
adventure	1
genre	1
fairy	1
humor	1
have	1
great	1
...	...

Figure 7.1 Intuition of the multinomial naive Bayes classifier applied to a movie review. The position of the words is ignored (the *bag of words* assumption) and we make use of the frequency of each word.

Figure from J&M 3rd ed. draft, sec 7.1

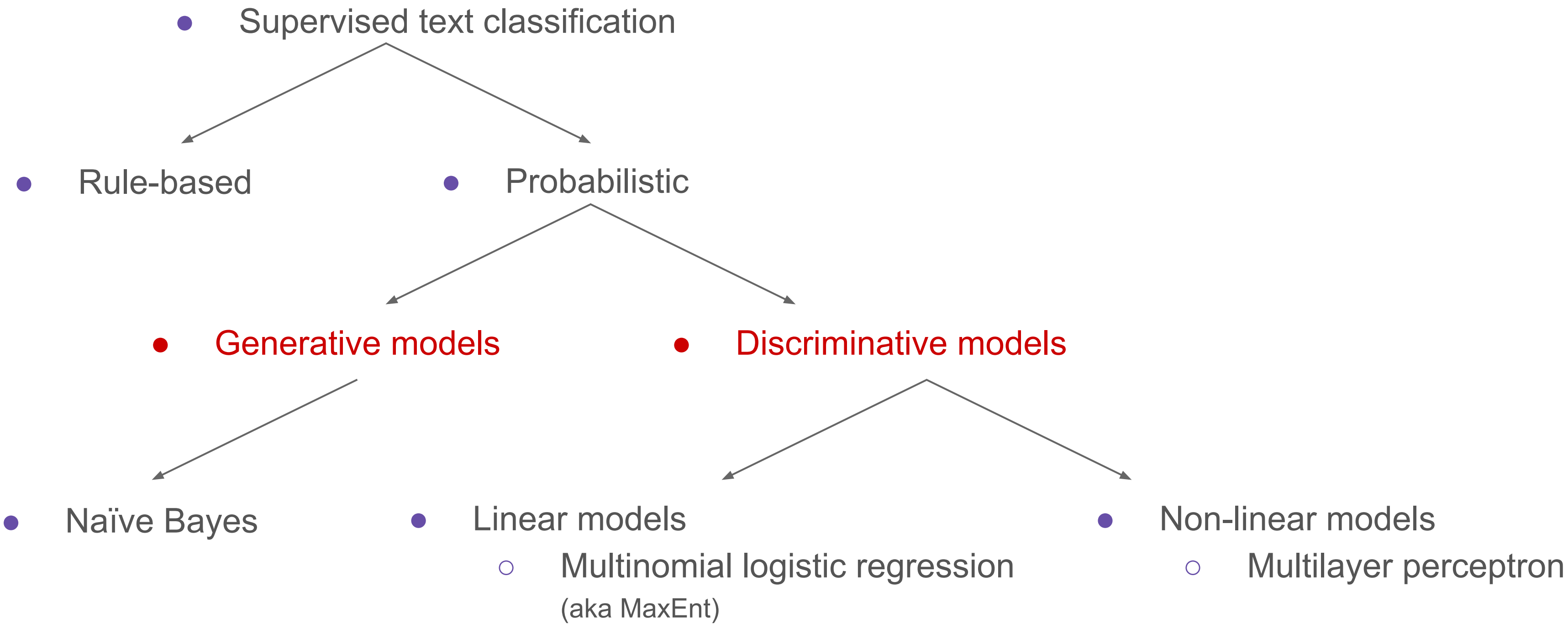
Types of textual features

- Words
  - content words, stop-words
  - punctuation? tokenization? lemmatization? lowercase?
- Word sequences
  - bigrams, trigrams, n-grams
- Grammatical structure, sentence parse tree
- Words’ part-of-speech
- Word vectors
- …

Possible representations for text

- Bag-of-Words (BOW)
  - Easy, no effort required
  - Variable size, ignores sentential structure
- Hand-crafted features
  - Full control, can use NLP pipeline, class-specific features
  - Over-specific, incomplete, makes use of NLP pipeline
- Learned feature representations
  - Can learn to contain all relevant information
  - Needs to be learned

We’ll consider alternative models for classification



- Supervised text classification
- Rule-based
- Probabilistic
- Generative models
- Discriminative models
- Naïve Bayes
- Linear models
  - Multinomial logistic regression (aka MaxEnt)
- Non-linear models
  - Multilayer perceptron

Generative and discriminative models

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

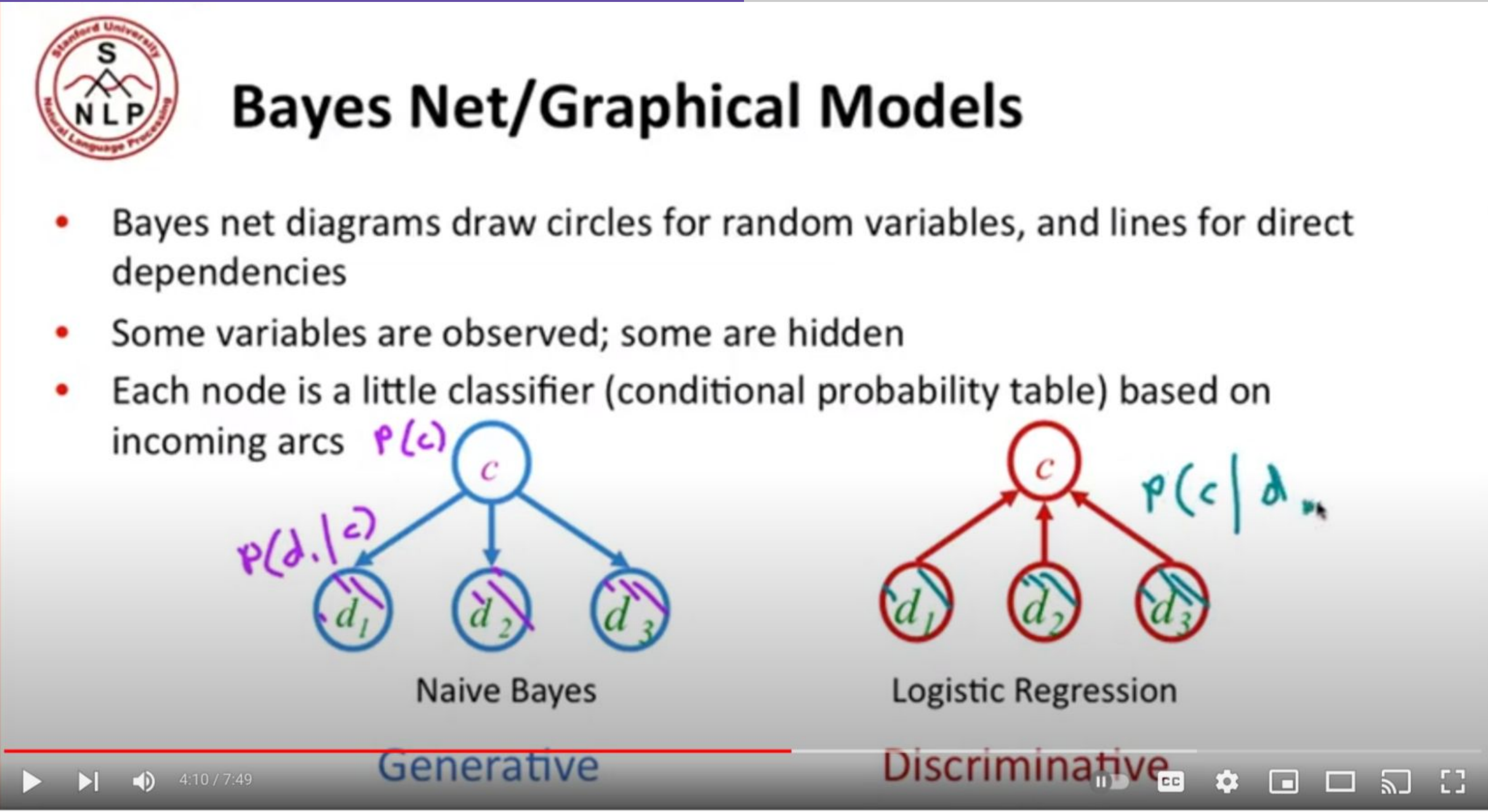
$$P(X, Y)$$

joint

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

$$P(Y \mid X)$$

conditional



Bayes Net/Graphical Models

- Bayes net diagrams draw circles for random variables, and lines for direct dependencies
- Some variables are observed; some are hidden
- Each node is a little classifier (conditional probability table) based on incoming arcs

Generative and discriminative models

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

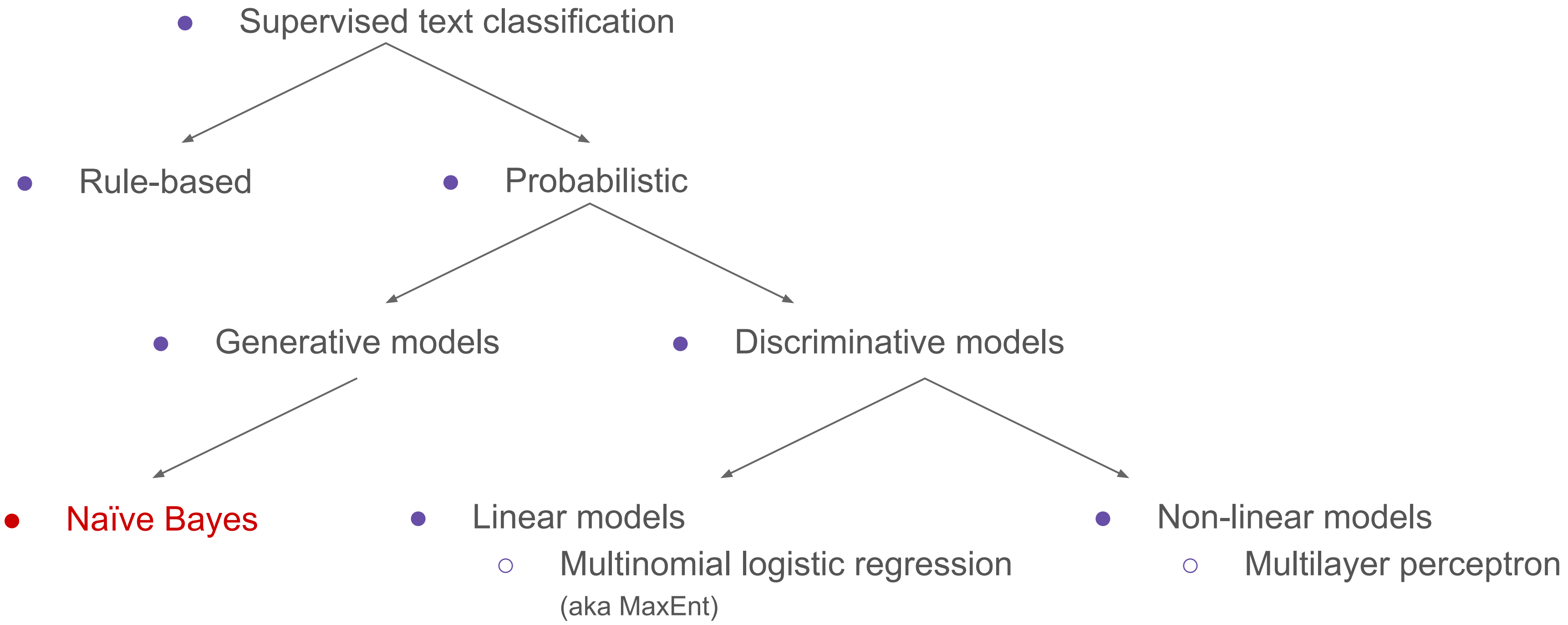
$$\hat{y} = \underset{\tilde{y}}{\operatorname{argmax}}\, P(X, \tilde{y})$$

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

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

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.

We’ll consider alternative models for classification



- Supervised text classification
- Rule-based
- Probabilistic
- Generative models
- Discriminative models
- Naïve Bayes
- Linear models
  - Multinomial logistic regression (aka MaxEnt)
- Non-linear models
  - Multilayer perceptron

Generative text classification: naïve Bayes

- Simple ("naïve") classification method
  - based on the Bayes rule
- Relies on a very simple representation of documents
  - bag-of-words, no relative order
- A good baseline for more sophisticated models

Naïve Bayes

Sentiment analysis: movie reviews

- Given a document d (e.g., a movie review)
- Decide which class c it belongs to: positive, negative, neutral
- Compute P(c | d) for each c
    - P(positive | d), P(negative | d), P(neutral | d)
    - select the one with max P

Naïve Bayes

- Given a document d and a class c, Bayes' rule:

$$P(c|d) = \frac{P(d|c)P(c)}{P(d)}$$

Naïve Bayes

- Given a document d and a class c, Bayes’ rule:

$$P(c|d) = \frac{P(d|c)P(c)}{P(d)}$$

$$P(\text{‘positive’}|d) \propto \underset{\substack{\Big\downarrow \\ \text{likelihood}}}{P(d|\text{‘positive’})}\;\underset{\substack{\Big\downarrow \\ \text{prior}}}{P(\text{‘positive’})}$$

Naïve Bayes

- Given a document d and a class c, Bayes’ rule:

$$P(c|d) = \frac{P(d|c)P(c)}{P(d)}$$

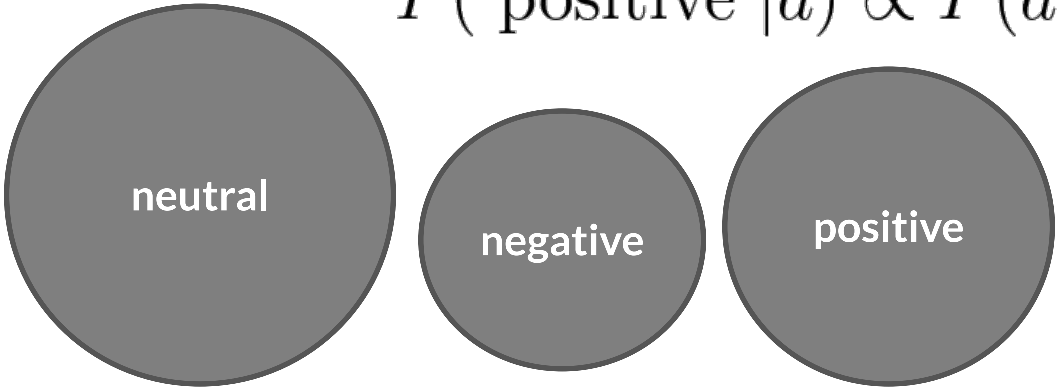
$$P(\text{‘positive’}|d) \propto P(d|\text{‘positive’})\underset{\overset{\big\downarrow}{\text{prior}}}{P(\text{‘positive’})}$$

Naïve Bayes

- Given a document d and a class c, Bayes' rule:

$$P(c|d) = \frac{P(d|c)P(c)}{P(d)}$$

$$P(\text{`positive'}|d) \propto P(d|\text{`positive'})P(\text{`positive'})$$



prior

Naïve Bayes

- Given a document d and a class c, Bayes' rule:

$$P(c|d) = \frac{P(d|c)P(c)}{P(d)}$$

$$P(\text{`positive'}|d) \propto \underbrace{P(d|\text{`positive'})}_{\text{likelihood}}P(\text{`positive'})$$

Naïve Bayes

- Given a document d and a class c, Bayes' rule:

$$P(c|d) = \frac{P(d|c)P(c)}{P(d)}$$

$$P(\text{`positive'}|d) \propto P(d|\text{`positive'})P(\text{`positive'})$$

$$\downarrow$$

likelihood

$$P(w_1, w_2, \ldots, w_n|c)$$

Naïve Bayes independence assumptions

$$P(w_1, w_2, \ldots, w_n | c)$$

- **Bag of Words assumption**: Assume word position doesn't matter
- **Conditional Independence**: Assume the feature probabilities $P(w_i | c_j)$ are independent given the class c

$$P(w_1, w_2, \ldots, w_n | c) = P(w_1|c) \times P(w_2|c) \times P(w_3|c) \times \ldots \times P(w_n|c)$$

Document representation

I love this movie. It’s sweet but with satirical humor. The dialogue is great and the adventure scenes are fun… it manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I’ve seen it several times, and I’m always happy to see it again whenever I have a friend who hasn’t seen it yet!

→ **bag of words (BOW)** →

it	6
I	5
the	4
to	3
and	3
seen	2
yet	1
would	1
whimsical	1
times	1
sweet	1
satirical	1
adventure	1
genre	1
fairy	1
humor	1
have	1
great	1
…	…

Document representation

I love this movie. It’s sweet but with satirical humor. The dialogue is great and the adventure scenes are fun… it manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I’ve seen it several times, and I’m always happy to see it again whenever I have a friend who hasn’t seen it yet!

→ **bag of words (BOW)** →

it	6
I	5
the	4
to	3
and	3
seen	2
yet	1
would	1
whimsical	1
times	1
sweet	1
satirical	1
adventure	1
genre	1
fairy	1
humor	1
have	1
great	1
…	…

$$P(d|c) = P(w_1, w_2, \ldots, w_n|c) = \prod_i P(w_i|c)$$

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

$$\underset{c_j}{\operatorname{argmax}} \; P(c_j) \textstyle\prod_i P(w_i|c)$$

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

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

Parameter estimation

- 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

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

$$\hat{P}(c_j) = \frac{doccount(C = c_j)}{N_{doc}} \qquad \hat{P}(w_i|c_j) = \frac{count(w_i, c_j)}{\sum_{w \in V} count(w, c_j)}$$

Parameter estimation

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

- 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

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

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

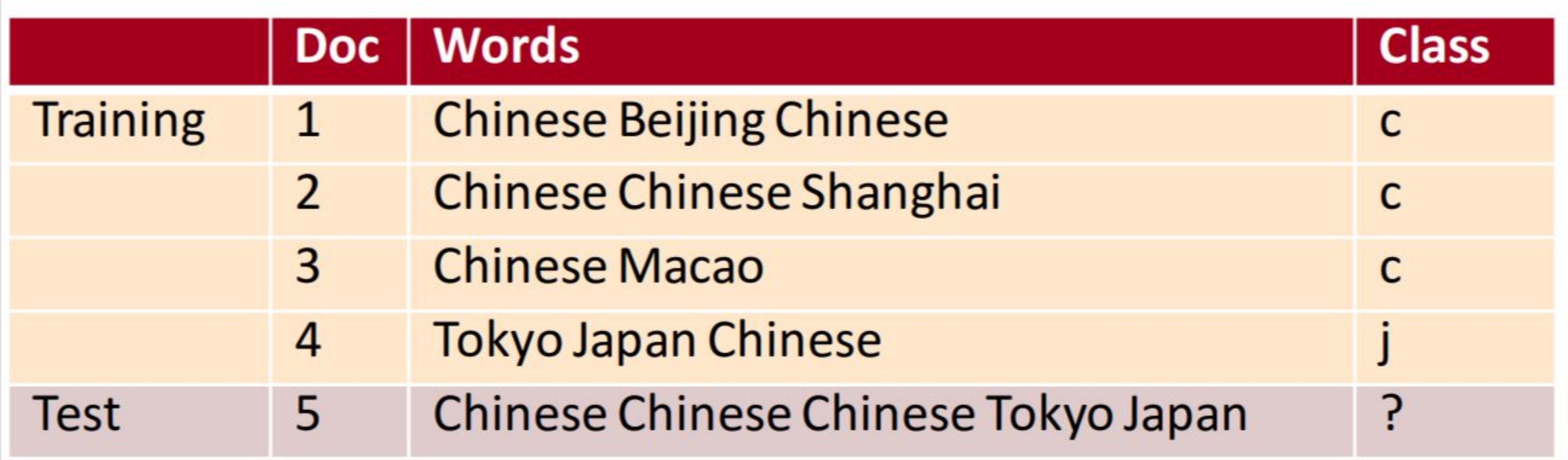
Multinomial naïve Bayes : learning

- From training corpus, extract *Vocabulary*
- Calculate $P(c_j)$ terms
  - For each c_j do
    - $docs_j \leftarrow$ all docs with class = c_j
    - $P(c_j) \leftarrow \dfrac{|docs_j|}{total\ \#\ documents}$

Multinomial naïve Bayes : learning

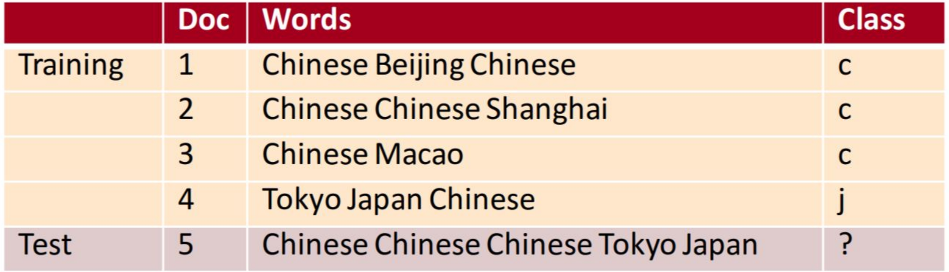
- From training corpus, extract *Vocabulary*
- Calculate $P(c_j)$ terms
  - For each c_j do
    - $docs_j \leftarrow$ all docs with class = c_j
    - $P(c_j) \leftarrow \dfrac{|docs_j|}{total\ \#\ documents}$
- Calculate $P(w_i | c_j)$ terms
  - $Text_j \leftarrow$ single doc containing all docs_j
  - For each word w_i in *Vocabulary*
    - $n_i \leftarrow$ # of occurrences of w_i in $Text_j$
    - $P(w_j | c_j) \leftarrow \dfrac{n_i + \alpha}{n + \alpha|Vocabulary|}$

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	?

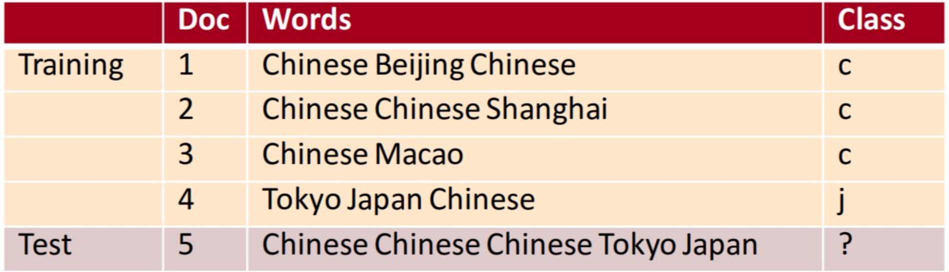
$$\hat{P}(c) = \frac{N_c}{N}$$

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:

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

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

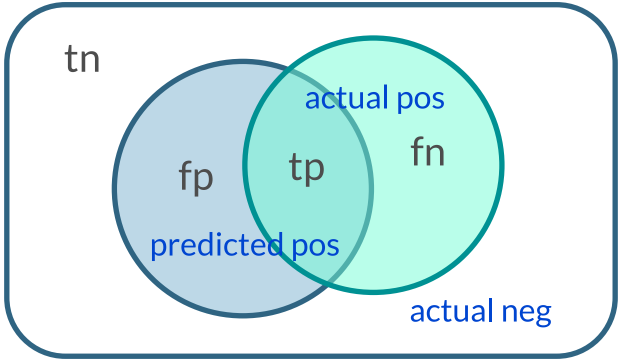
Summary: naïve Bayes is not so naïve

- Naïve Bayes is a probabilistic model
- Naïve because is assumes features are independent of each other for a class
- Very fast, low storage requirements
- Robust to Irrelevant Features
  - Irrelevant Features cancel each other without affecting results
- 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)



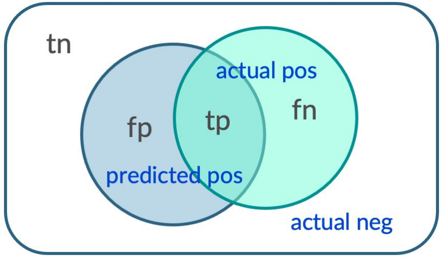
Classification evaluation

- 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

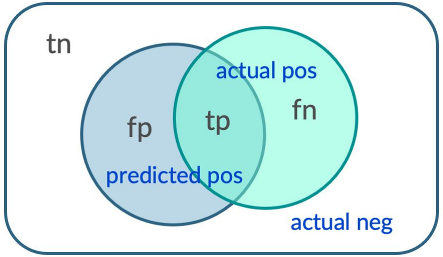
$$\text{precision} = \frac{true\ positives}{true\ positives + false\ positives} = \frac{tp}{tp + fp}$$



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

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

Classification evaluation

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

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

Micro- vs. macro-averaging

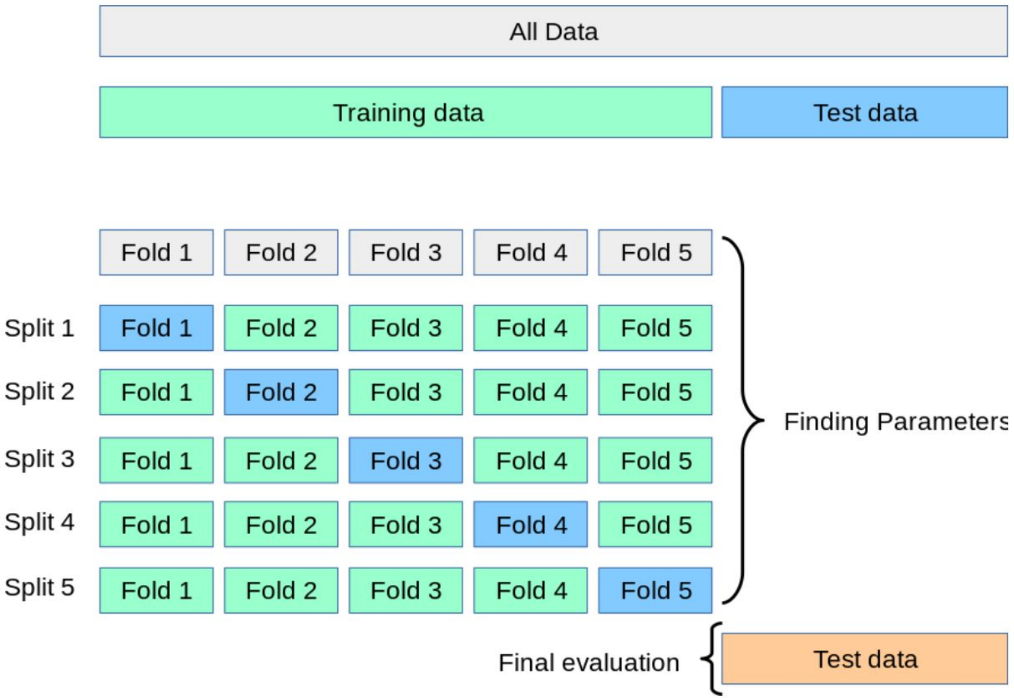
If we have more than one class, how do we combine multiple performance measures into one quantity?

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

Classification common practices

- Divide the training data into k folds (e.g., k=10)
- Repeat k times: train on k-1 folds and test on the holdout fold, cyclically
- Average over the k folds' results

K-fold cross-validation



All Data

Training data | Test data

Fold 1 | Fold 2 | Fold 3 | Fold 4 | Fold 5

Split 1: Fold 1 | Fold 2 | Fold 3 | Fold 4 | Fold 5

Split 2: Fold 1 | Fold 2 | Fold 3 | Fold 4 | Fold 5

Split 3: Fold 1 | Fold 2 | Fold 3 | Fold 4 | Fold 5

Split 4: Fold 1 | Fold 2 | Fold 3 | Fold 4 | Fold 5

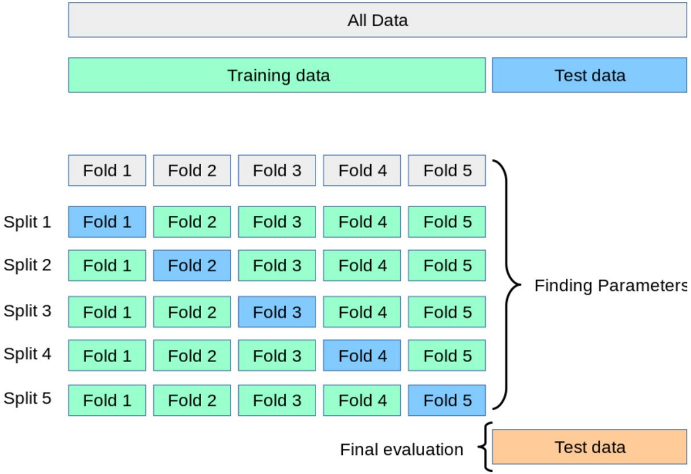
Split 5: Fold 1 | Fold 2 | Fold 3 | Fold 4 | Fold 5

Finding Parameters

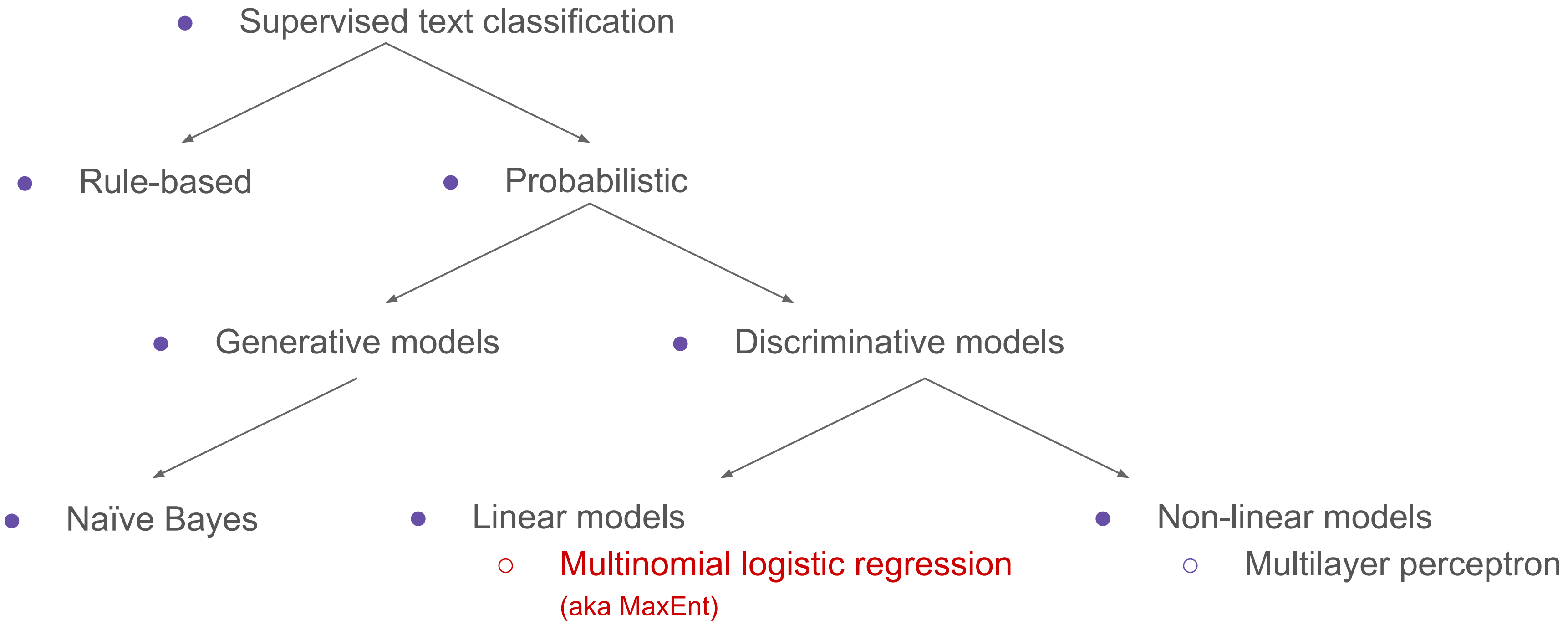
Final evaluation: Test data

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



- Supervised text classification
- Rule-based
- Probabilistic
- Generative models
- Discriminative models
- Naïve Bayes
- Linear models
    - Multinomial logistic regression (aka MaxEnt)
- Non-linear models
    - Multilayer perceptron