南京大學
NANJING UNIVERSITY

Text Classification

Xinyu Dai

2020-10

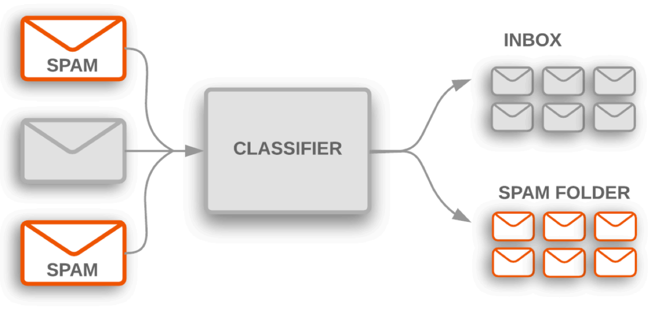
Classification

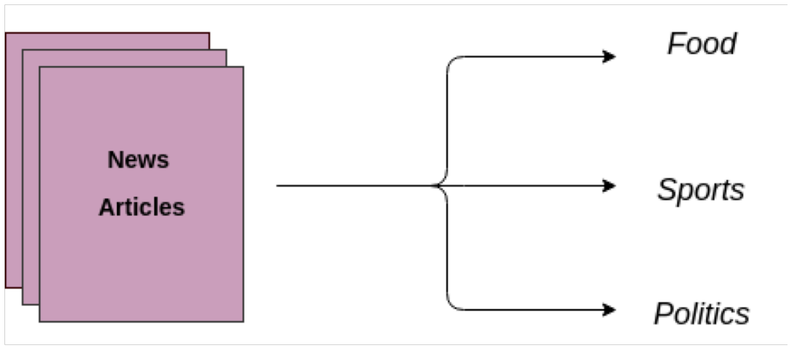
- Automatically make a decision about inputs
  - Example: document → category
  - Example: image of digit → digit
  - Example: image of object → object type
  - Example: query + webpages → best match
  - Example: symptoms → diagnosis
- Four main ideas
  - Representation as feature vectors
  - Model
  - Training
  - Inference

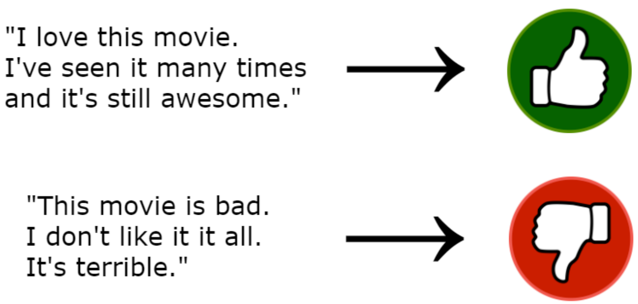
Example: Text Classification

- We want to classify documents into semantic categories

DOCUMENT	CATEGORY
… win the election …	*POLITICS*
… win the game …	*SPORTS*
… see a movie …	*OTHER*







Naïve Bayes Model for Text Classification

- Document D, with class c_k
- Naïve Bayes Model: Classify D as the class with the highest posterior probability:

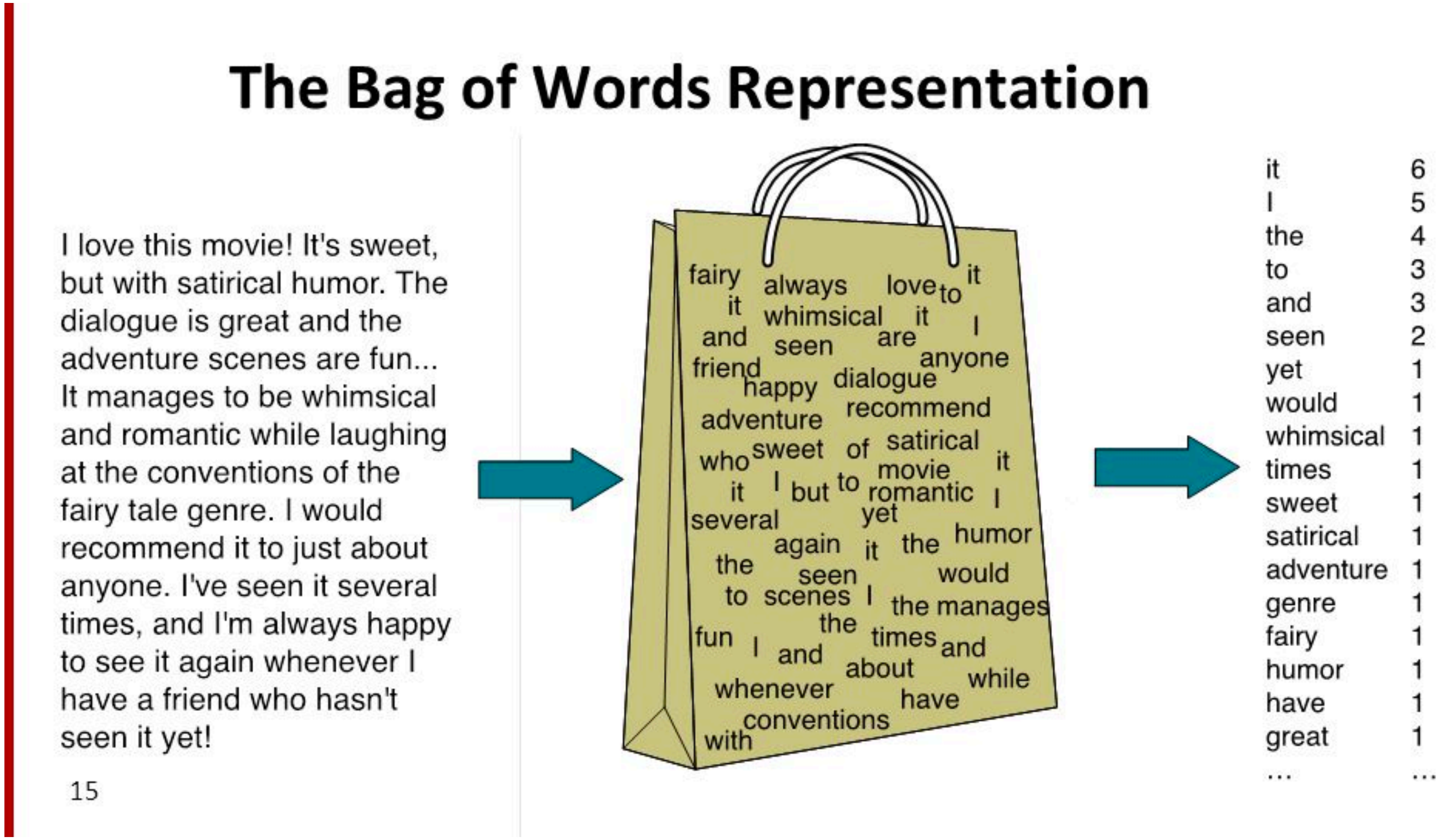
Why Bayes?

$$\operatorname{argmax}_{c_k} P(c_k \mid D) = \operatorname{argmax}_{c_k} \frac{P(D \mid c_k)P(c_k)}{P(D)} = \operatorname{argmax}_{c_k} P(D \mid c_k)P(c_k)$$

- How to represent D?
- How to inference $P(D \mid c_k)$ and $P(c_k)$?

Text Representation

- Bag-of-Words



- To be continued……

Naïve Bayes Model for text classification

- **Bernoulli document model：**

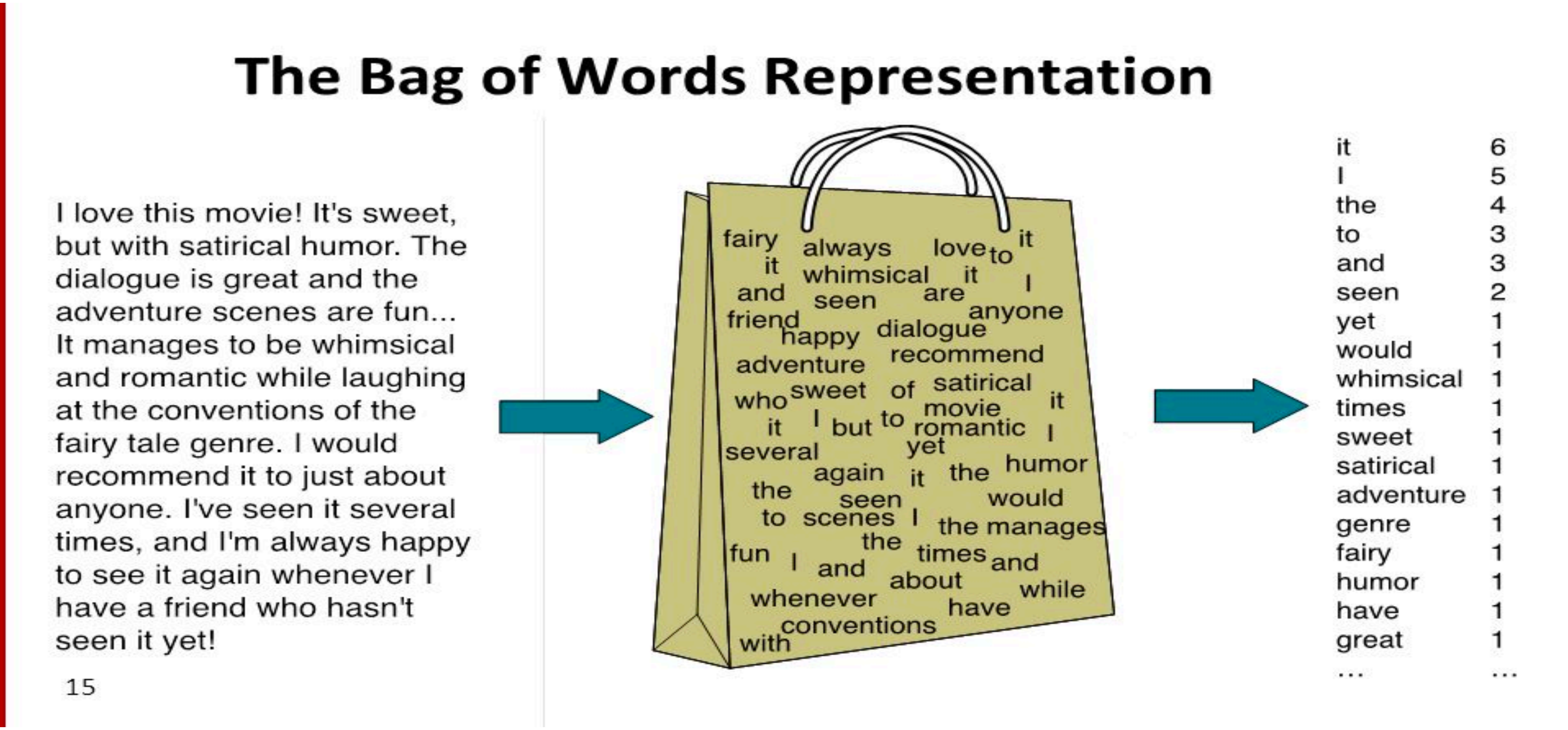
  a document is represented by a binary feature vector, whose elements indicate absence or presence of corresponding word in the document

- **Multinomial document model**:

  a document is represented by an integer feature vector, whose elements indicate frequency of corresponding word in the document

Naïve Bayes Model for text classification

- **Bernoulli document model:**
  a document is represented by a binary feature vector, whose elements indicate absence or presence of corresponding word in the document
-



Naive Bayes
-- Bernoulli Document Model

- A vocabulary *V* containing a set of |*V*| words
- A documents *D* is represented as a *V* dimensional 0-1 vector.
- Generative model:
  - for each word w
  - flip a coin, with probability of heads $P(w|\ c_k)$
  - if heads, w is included the document
- We thus generate a document containing the selected words
- But no count information for each word

Naive Bayes
-- Bernoulli Document Model

- D_i is the feature vector for the i^{th} document.
- D_{it}, is either 0 or 1 representing the absence or presence of the word w_t in D_i.
- $P(w_t|c_k)$ is the probability of word w_t occurring in document of class c_k ; $(1 - P(w_t| c_k))$ is probability of w_t not occurring

$$P(D_{it} \mid c_k) = D_{it} P(w_t \mid c_k) + (1 - D_{it})(1 - P(w_t \mid c_k))$$

$$P(D_i \mid c_k) = \prod_{t=1}^{|V|} P(D_{it} \mid c_k) = \prod_{t=1}^{|V|} [D_{it} P(w_t \mid c_k) + (1 - D_{it})(1 - P(w_t \mid c_k))]$$

Training a Bernoulli Documents Model

- Parameters:
  - likelihoods of each word given the class $P(w_t|c_k)$
  - prior probabilities $P(c_k)$
- Let $n_k(w_t)$ be the number of documents of class c_k in which w_t is observed, and let N_k be the total number of documents in c_k
- Estimate the word likelihoods as: $$\hat{P}(w_t \mid c_k) = \frac{n_k(w_t)}{N_k}$$
- Estimate priors as: $$\hat{P}(c_k) = \frac{N_k}{N}$$

Training a Bernoulli Documents Model

- We have labeled documents set.
- Define the vocabulary V， the number of words in the vocabulary defines the dimension of the feature vectors.
- Count in the training set:
  - N (number of documents)
  - N_k (number of documents of class c_k)
  - $n_k(w_t)$ (number of documents of class c_k containing w_t)
- Estimate likelihoods $P(w_t|c_k)$
- Estimate priors $P(c_k)$

Classifying with the Bernoulli Model

- To classify an unlabelled document *D*, we estimate the posterior probability for each class, and find which class gives maximum probability:

$$\operatorname{argmax}_{c_k} P(c_k \mid D_j) = \operatorname{argmax}_{c_k} P(D_j \mid c_k) P(c_k)$$

$$= \operatorname{argmax}_{c_k} P(c_k) \prod_{t=1}^{|V|} [D_{jt} P(w_t \mid c_k) + (1 - D_{jt})(1 - P(w_t \mid c_k))]$$

Example

- Consider a set of documents each of which is related either to *Sports (S)* or to *Informatics (I)*.
- We define a vocabulary *V* of eight words:
  - w1 = goal
  - w2 = tutor
  - w3 = variance
  - w4 = speed
  - w5 = drink
  - w6 = defence
  - w7 = performance
  - w8 = field

Example

- Training data (each corresponds to a document, each column corresponds to a word):

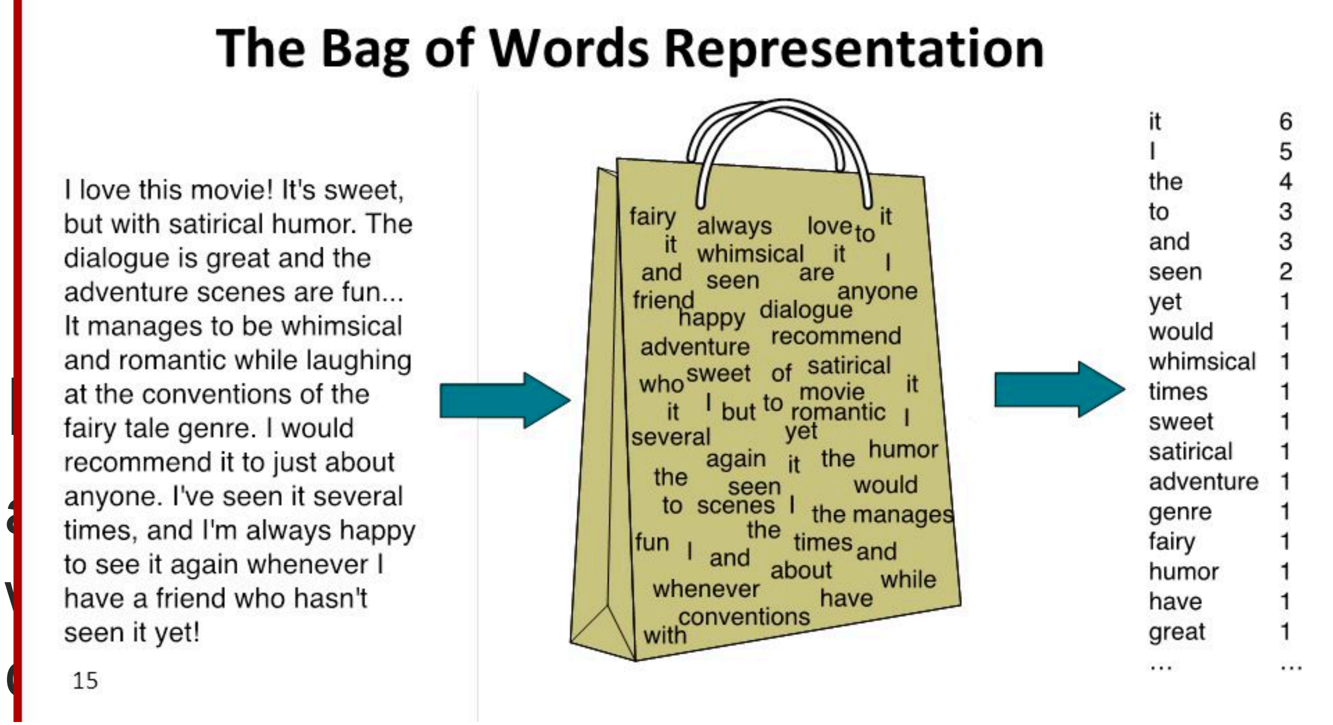
$$\mathbf{B}^{\text{Sport}} = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \end{pmatrix}$$

$$\mathbf{B}^{\text{Inf}} = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \end{pmatrix}$$

Example

- Classify the new sample
  - B1 = [1 0 0 1 1 1 0 1]
  - B2 = [0 1 1 0 1 0 1 0]
- ……

Naïve Bayes Model for text classification



- ...e vector, ...ce of

- **Multinomial document model**:

  a document is represented by an integer feature vector, whose elements indicate frequency of corresponding word in the document

Multinomial model

- Document feature vectors capture word frequency information (not just presence or absence)
- As in the Bernoulli model
  - Vocabulary V containing a set of $|V|$ words
  - Dimension t of a document vector corresponds to word w_t in the vocabulary
  - $P(w_t|c_k)$ is the probability of word w_t occurring in document of class c_k
- Multinomial generative model
  - consider a $|V|$-sided dice
  - each side i corresponds to word w_i with probability $P(w_i|c_k)$
  - at each position in the document roll the dice and insert the corresponding word
- Generates a document as a bag of words — includes what words are in the document, and how many times they occur

Multinomial model for TC

- D_i is the feature vector for the i^{th} document.
- D_{it}, is the number of times word w_t occurs in D_i.

$n_i = \sum_t D_{it}$ is the total number of words in D_i.

- $P(w_t|c_k)$ is the probability of word w_t occurring in document of class c_k based on multinomial distribution.

$$P(D_i \mid c_k) = \frac{n_i!}{\prod_{t=1}^{|V|} D_{it}!} \prod_{t=1}^{|V|} P(w_t \mid c_k)^{D_{it}}$$

Training a Multinomial Documents Model

- Parameters, the same as Bernoulli Model :
  - likelihoods of each word given the class $P(w_t|c_k)$
  - prior probabilities $P(c_k)$
- Let $z_{ik}=1$ when the i^{th} *Documents* has class c_k; otherwise $z_{ik}=0$
- N_k is the total number of documents in c_k
- N is the total number of documents.
- Estimate the word likelihoods as:

$$\hat{P}(w_t \mid c_k) = \frac{\sum_{i=1}^{N} D_{it} z_{ik}}{\sum_{s=1}^{|V|} \sum_{i=1}^{N} D_{is} z_{ik}}$$

  - relative frequency of w_t in documents of class c_k with respect to the total number of words in documents of that class

- Estimate priors as:

$$\hat{P}(c_k) = \frac{N_k}{N}$$

Training a Multinomial Documents Model

- We have labeled documents set.
- Define the vocabulary *V*， the number of words in the vocabulary defines the dimension of the feature vectors.
- Count in the training set:
  - N (number of documents)
  - N_k (number of documents of class c_k)
  - D_{it} *the frequency of a word* w_t *in a document* D_i *for all words in V and all documents*
- Estimate likelihoods $P(w_t|c_k)$
- Estimate priors $P(c_k)$

Classifying with Multinomial Model

- To classify an unlabelled document *D*, we estimate the posterior probability for each class, and find which class gives maximum probability:

$$\text{argmax}_{c_k} P(c_k \mid D_j) = \text{argmax}_{c_k} P(D_j \mid c_k) P(c_k)$$

$$= \text{argmax}_{c_k} P(c_k) \frac{n_i!}{\prod_{t=1}^{|V|} D_{it}!} \prod_{t=1}^{|V|} P(w_t \mid c_k)^{D_{it}}$$

$$= \text{argmax}_{c_k} P(c_k) \prod_{t=1}^{|V|} P(w_t \mid c_k)^{D_{it}}$$

$$= \text{argmax}_{c_k} P(c_k) \prod_{h=1}^{len(D_i)} P(u_h \mid c_k)$$

u_h: the word in D_i

Zero probability problem and smoothing

- If a word does not occur in the training data for a class that does not mean it cannot occur in any document of that class
- Add-one smoothing

Summary and question

- Task description
- Naïve Bayes model for Text classification
  - Bernoulli Model
  - Multinomial Model
- Zero probability problem
- Consider the word “the” or the Chinese word “的”. What will be the approximate value of the probability P(“the” | ck) in
  - the Bernoulli model？
  - the multinomial model?

- Reformulate the text classification task
- Linear model
- Text Representation
- Feature Selection

Task again

- We want to classify documents into semantic categories

DOCUMENT	CATEGORY
… win the election …	*POLITICS*
… win the game …	*SPORTS*
… see a movie …	*OTHER*

Formulate the task again

INPUTS	$\mathbf{x}_i$	… win the election …
CANDIDATE SET	$\mathcal{Y}(\mathbf{x})$	[… win the election … SPORTS], [… win the election … POLITICS], [… win the election … OTHER]
CANDIDATES	$\mathbf{y}$	[… win the election … SPORTS]
TRUE OUTPUTS	$\mathbf{y}_i^*$	[… win the election … POLITICS]
FEATURE VECTORS	$\mathbf{f}_i(\mathbf{y})$	[0 0 0 0 1 0 1 0 0 0 0 0]

SPORTS ∧ "win"

POLITICS ∧ "win"

POLITICS ∧ "election"

Remember: if y contains x, we also write $f(y)$

Feature vectors

- Sometimes, we think of the input as having features, which are multiplied by outputs to form the candidates

$\mathbf{x}$ … *win the election* …

"$\mathbf{f}(\mathbf{x})$" $[1\ 0\ 1\ 0]$

"*win*" "*election*"

… *win the election* …

$$\mathbf{f}(\textit{SPORTS}) = [1\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0]$$

… *win the election* …

$$\mathbf{f}(\textit{POLITICS}) = [0\ 0\ 0\ 0\ 1\ 0\ 1\ 0\ 0\ 0\ 0\ 0]$$

… *win the election* …

$$\mathbf{f}(\textit{OTHER}) = [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 1\ 0]$$

Linear Model

- Simply, each feature gets a weight *w*

$$\mathbf{f}(\overset{\text{... win the election ...}}{POLITICS}) = [\ 0\ \ 0\ \ 0\ \ 0\ \ 1\ \ 0\ \ 1\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0]$$

$$\mathbf{f}(\overset{\text{... win the election ...}}{SPORTS}) = [\ 1\ \ 0\ \ 1\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0]$$

$$\mathbf{w} = [\ 1\ \ 1\ \ -1\ \ -2\ \ 1\ \ -1\ \ 1\ \ -2\ \ -2\ \ -1\ \ -1\ \ 1]$$

- Define a linear function to score the hypothesis. Score the hypothesis by multiplying features and weights:

$$score(\mathbf{y}, \mathbf{w}) = \mathbf{w}^\top \mathbf{f}(\mathbf{y})$$

$$\mathbf{f}(\overset{\text{... win the election ...}}{POLITICS}) = [\ 0\ \ 0\ \ 0\ \ 0\ \ 1\ \ 0\ \ 1\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0]$$

$$\mathbf{w} = [\ 1\ \ 1\ \ -1\ \ -2\ \ 1\ \ -1\ \ 1\ \ -2\ \ -2\ \ -1\ \ -1\ \ 1]$$

$$score(\overset{\text{... win the election ...}}{POLITICS}, \mathbf{w}) = 1 \times 1 + 1 \times 1 = 2$$

Linear Model: prediction

$$prediction(\text{... win the election ...}, \mathbf{w}) = \arg\max_{\mathbf{y}\in\mathcal{Y}(\mathbf{x})} \mathbf{w}^\top \mathbf{f}(\mathbf{y})$$

$$score(\overset{\text{... win the election ...}}{SPORTS}, \mathbf{w}) = 1 \times 1 + (-1) \times 1 = 0$$

$$score(\overset{\text{... win the election ...}}{POLITICS}, \mathbf{w}) = 1 \times 1 + 1 \times 1 = 2$$

$$score(\overset{\text{... win the election ...}}{OTHER}, \mathbf{w}) = (-2) \times 1 + (-1) \times 1 = -3$$

$$prediction(\text{... win the election ...}, \mathbf{w}) = \overset{\text{... win the election ...}}{POLITICS}$$

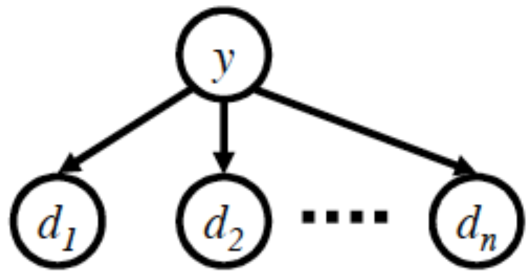
Linear Model: Naïve Bayes

- Naïve-Bayes is a linear model, where:

$$\mathbf{x}^i = d_1, d_2, \cdots d_n$$

$$\begin{array}{lcllllllll}
\mathbf{f}_i(\mathbf{y}) & = & [& \cdots 0 \cdots & 1, & \#v_1, & \#v_2, & \cdots & \#v_{|V|} & \cdots 0 \cdots &] \\
\mathbf{w} & = & [& \cdots & \log \mathsf{P}(y), & \log \mathsf{P}(v_1|y), & \log \mathsf{P}(v_2|y), & \cdots & \log \mathsf{P}(v_n|y) & \cdots &]
\end{array}$$

$$\begin{aligned}
\text{score}(\mathbf{x}_i, \mathbf{y}, \mathbf{w}) &= \mathbf{w}^\top \mathbf{f}_i(\mathbf{y}) \\
&= \log \mathsf{P}(\mathbf{y}) + \sum_k \#v_k \log \mathsf{P}(v_k|\mathbf{y}) \\
&= \log \left(\mathsf{P}(\mathbf{y}) \prod_k \mathsf{P}(v_k|\mathbf{y})^{\#v_k} \right) \\
&= \log \left(\mathsf{P}(\mathbf{y}) \prod_{d \in \mathbf{x}^i} \mathsf{P}(d|\mathbf{y}) \right) \\
&= \log \mathsf{P}(\mathbf{x}^i, \mathbf{y})
\end{aligned}$$



Learning the weight

- Goal:
  - Choose a "best" vector *w* given the training data
  - The ideal: the weights which have greatest test set accuracy or F1.
  - we want weights which give best training set accuracy?
- MLE in Naïve Bayes Model (Maximum Likelihood)
- Based on some error-related criterion
  - Perceptron
  - Logistic Regression (Maximum Entropy)
  - SVM

Minimize Training Error

- A loss function declares how costly each mistake is

$$\ell_i(\mathbf{y}) = \ell(\mathbf{y}, \mathbf{y}_i^*)$$

- We could, in principle, minimize training loss:

$$\min_{\mathbf{w}} \sum_i \ell_i \left(\arg\max_{\mathbf{y}} \mathbf{w}^\top \mathbf{f}_i(\mathbf{y}) \right)$$

Learning the weight

- Goal:
  - Choose a “best” vector *w* given the training data
  - The ideal: the weights which have greatest test set accuracy or F1.
  - we want weights which give best training set accuracy?
- MLE in Naïve Bayes Model (Maximum Likelihood)
- Based on some error-related criterion
  - Perceptron: 0-1 loss
  - Logistic Regression (Maximum Entropy): log-loss
  - SVM: hinge-loss

Linear Models: Maximum Entropy

- Maximum entropy (logistic regression)
  - Use the scores as probabilities:

$$P(\mathbf{y}|\mathbf{x},\mathbf{w}) = \frac{\exp(\mathbf{w}^\top \mathbf{f}(\mathbf{y}))}{\sum_{\mathbf{y}'} \exp(\mathbf{w}^\top \mathbf{f}(\mathbf{y}'))} \quad \begin{array}{l} \longleftarrow \text{Make positive} \\ \longleftarrow \text{Normalize} \end{array}$$

  - Maximize the (log) conditional likelihood of training data

$$L(\mathbf{w}) = \log \prod_i P(\mathbf{y}_i^*|\mathbf{x}_i,\mathbf{w}) = \sum_i \log \left(\frac{\exp(\mathbf{w}^\top \mathbf{f}_i(\mathbf{y}_i^*))}{\sum_{\mathbf{y}} \exp(\mathbf{w}^\top \mathbf{f}_i(\mathbf{y}))} \right)$$

$$= \sum_i \left(\mathbf{w}^\top \mathbf{f}_i(\mathbf{y}_i^*) - \log \sum_{\mathbf{y}} \exp(\mathbf{w}^\top \mathbf{f}_i(\mathbf{y})) \right)$$

Maximum Entropy II

- Motivation for maximum entropy:
  - Connection to maximum entropy principle (sort of)
  - Might want to do a good job of being uncertain on noisy cases...
- Regularization (smoothing)

$$\max_{\mathbf{w}} \sum_i \left(\mathbf{w}^\top \mathbf{f}_i(\mathbf{y}_i^*) - \log \sum_{\mathbf{y}} \exp(\mathbf{w}^\top \mathbf{f}_i(\mathbf{y})) \right) - k||\mathbf{w}||^2$$

$$\min_{\mathbf{w}} \ k||\mathbf{w}||^2 - \sum_i \left(\mathbf{w}^\top \mathbf{f}_i(\mathbf{y}_i^*) - \log \sum_{\mathbf{y}} \exp(\mathbf{w}^\top \mathbf{f}_i(\mathbf{y})) \right)$$

Log-Loss

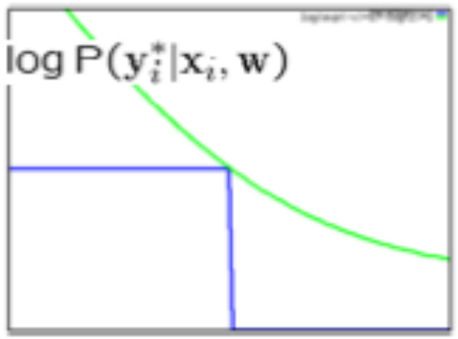
- We view maxent as a minimization problem:

$$\min_{\mathbf{w}} \quad k||\mathbf{w}||^2 - \sum_i \left(\mathbf{w}^\top \mathbf{f}_i(\mathbf{y}_i^*) - \log \sum_{\mathbf{y}} \exp(\mathbf{w}^\top \mathbf{f}_i(\mathbf{y})) \right)$$

- This minimizes the "log loss" on each example

$$-\left(\mathbf{w}^\top \mathbf{f}_i(\mathbf{y}_i^*) - \log \sum_{\mathbf{y}} \exp(\mathbf{w}^\top \mathbf{f}_i(\mathbf{y})) \right) = -\log \mathrm{P}(\mathbf{y}_i^* | \mathbf{x}_i, \mathbf{w})$$

$$step\left(\mathbf{w}^\top \mathbf{f}_i(\mathbf{y}_i^*) - \max_{\mathbf{y} \neq \mathbf{y}_i^*} \mathbf{w}^\top \mathbf{f}_i(\mathbf{y}) \right)$$



Back to NLP from ML ☺

- Text representation
  - Feature vectors
  - Features Engineering
  - Weight
    - 0-1 vectors
    - Count vectors
    - tf*idf

Features

- Bag of words
- Phrase-based
- N-gram
- Hypernym Representation
  - Using some lexicon or thesaurus
- Graph-based Representation
- Distributed Representation：word2vec, sen2vec, doc2vec……

Feature selection

- High dimensional space
- Eliminating noise features from the representation increases efficiency and effectiveness of text classification.
- Selecting a subset of relevant features for building robust learning models.
- Actually feature filtering
  - assign heuristic score to each feature *f* to filter out the “obviously” useless ones.

Different feature selection methods

- A feature selection method is mainly defined by the feature utility measures it employs.
- Feature utility measures:
  - Stop words
  - Frequency – select the most frequent terms
  - Mutual information – select the terms with the highest mutual information (mutual information is also called information gain in this context)
  - X^2 (Chi-square)

Mutual Information

- Formally, the mutual information of two discrete random variables *X* and *Y* can be defined as:

$$I(X;Y) = \sum_{y \in Y} \sum_{x \in X} p(x,y) \log\left(\frac{p(x,y)}{p(x)\,p(y)}\right)$$

Mutual Information

- Based on maximum likelihood estimates, the formula we actually use is:

$$I(U;C) = \frac{N_{11}}{N}\log_2\frac{NN_{11}}{N_{1.}N_{.1}} + \frac{N_{10}}{N}\log_2\frac{NN_{10}}{N_{1.}N_{.0}} \quad (1)$$
$$+\frac{N_{01}}{N}\log_2\frac{NN_{01}}{N_{0.}N_{.1}} + \frac{N_{00}}{N}\log_2\frac{NN_{00}}{N_{0.}N_{.0}}$$

N_{11}: # of documents that contain t ($e_t = 1$) and are in c ($e_c = 1$)
N_{10}: # of documents that contain t ($e_t = 1$) and not in c ($e_c = 0$)
N_{01}: # of documents that don't contain t ($e_t = 0$) and in c ($e_c = 1$)
N_{00}: # of documents that don't contain t ($e_t = 0$) and not in c ($e_c = 0$)

$N = N_{00} + N_{01} + N_{10} + N_{11}$

$p(t,c) \approx N_{11}/N$, $p(\overline{t},c) \approx N_{01}/N$, $p(t,\overline{c}) \approx N_{10}/N$, $p(\overline{t},\overline{c}) \approx N_{00}/N$

$N_{1.} = N_{10} + N_{11}$: # documents that contain t, $p(t) \approx N_{1.}/N$
$N_{.1} = N_{01} + N_{11}$: # documents in c, $p(c) \approx N_{.1}/N$
$N_{0.} = N_{00} + N_{01}$: # documents that don't contain t, $p(\overline{t}) \approx N_{0.}/N$
$N_{.0} = N_{00} + N_{10}$: # documents not in c, $p(\overline{c}) \approx N_{.0}/N$

MI example for poultry/export in Reuters

	$e_c = e_{\text{POULTRY}} = 1$	$e_c = e_{\text{POULTRY}} = 0$
$e_t = e_{export} = 1$	$N_{11} = 49$	$N_{10} = 141$
$e_t = e_{export} = 0$	$N_{01} = 27{,}652$	$N_{00} = 774{,}106$

Plug these values into formula:

$$
\begin{aligned}
I(U;C) &= \frac{49}{801{,}948}\log_2\frac{801{,}948\cdot 49}{(49+27{,}652)(49+141)} \\
&\quad+\frac{141}{801{,}948}\log_2\frac{801{,}948\cdot 141}{(141+774{,}106)(49+141)} \\
&\quad+\frac{27{,}652}{801{,}948}\log_2\frac{801{,}948\cdot 27{,}652}{(49+27{,}652)(27{,}652+774{,}106)} \\
&\quad+\frac{774{,}106}{801{,}948}\log_2\frac{801{,}948\cdot 774{,}106}{(141+774{,}106)(27{,}652+774{,}106)} \\
&\approx 0.000105
\end{aligned}
$$

MI feature selection on Reuters

Terms with highest mutual information for three classes:

COFFEE

coffee	0.0111
bags	0.0042
growers	0.0025
kg	0.0019
colombia	0.0018
brazil	0.0016
export	0.0014
exporters	0.0013
exports	0.0013
crop	0.0012

SPORTS

soccer	0.0681
cup	0.0515
match	0.0441
matches	0.0408
played	0.0388
league	0.0386
beat	0.0301
game	0.0299
games	0.0284
team	0.0264

POULTRY

poultry	0.0013
meat	0.0008
chicken	0.0006
agriculture	0.0005
avian	0.0004
broiler	0.0003
veterinary	0.0003
birds	0.0003
inspection	0.0003
pathogenic	0.0003

$I(export, \text{POULTRY}) \approx .000105$ not among the ten highest for class POULTRY, but still potentially significant.

X^2 Feature selection

χ^2 tests independence of two events, $p(A,B) = p(A)p(B)$
(or $p(A|B) = p(A)$, $p(B|A) = p(B)$) .
Test occurrence of the term, occurrence of the class, rank w.r.t.:

$$X^2(D,t,c) = \sum_{e_t \in \{0,1\}} \sum_{e_c \in \{0,1\}} \frac{(N_{e_t e_c} - E_{e_t e_c})^2}{E_{e_t e_c}}$$

where N = *observed* frequency in D, E = *expected* frequency (e.g., E_{11} is the expected frequency of t and c occurring together in a document, assuming term and class are independent)

High value of X^2 indicates independence hypothesis is incorrect, i.e., observed and expected are too dissimilar.
If occurrence of term and class are dependent events, then occurrence of term makes class more (or less) likely, hence helpful as feature.

X^2 Feature selection: example

Are class POULTRY and term *export* interdependent by χ^2 test?

	$e_c = e_{\text{POULTRY}} = 1$	$e_c = e_{\text{POULTRY}} = 0$
$e_t = e_{export} = 1$	$N_{11} = 49$	$N_{10} = 141$
$e_t = e_{export} = 0$	$N_{01} = 27{,}652$	$N_{00} = 774{,}106$

$$N = N_{11} + N_{10} + N_{01} + N_{00} = 801948$$

Identify:

$$p(t) = \frac{N_{11}+N_{10}}{N},\ p(c) = \frac{N_{11}+N_{01}}{N},\ p(\overline{t}) = \frac{N_{01}+N_{00}}{N},\ p(\overline{c}) = \frac{N_{10}+N_{00}}{N}$$

Then estimate expected frequencies:

	$e_c = e_{\text{POULTRY}} = 1$	$e_c = e_{\text{POULTRY}} = 0$
$e_t = e_{export} = 1$	$E_{11} = Np(t)p(c)$	$E_{10} = Np(t)p(\overline{c})$
$e_t = e_{export} = 0$	$E_{01} = Np(\overline{t})p(c)$	$E_{00} = Np(\overline{t})p(\overline{c})$

e.g., $E_{11} = N \cdot p(t) \cdot p(c) = N \cdot \frac{N_{11}+N_{10}}{N} \cdot \frac{N_{11}+N_{01}}{N}$

$$= N \cdot \frac{49+141}{N} \cdot \frac{49+27652}{N} \approx 6.6$$

Expected Frequencies

From

	$e_c = e_{\text{POULTRY}} = 1$	$e_c = e_{\text{POULTRY}} = 0$
$e_t = e_{export} = 1$	$E_{11} = Np(t)p(c)$	$E_{10} = Np(t)p(\overline{c})$
$e_t = e_{export} = 0$	$E_{01} = Np(\overline{t})p(c)$	$E_{00} = Np(\overline{t})p(\overline{c})$

the full table of expected frequencies is

	$e_c = e_{\text{POULTRY}} = 1$	$e_c = e_{\text{POULTRY}} = 0$
$e_t = e_{export} = 1$	$E_{11} \approx 6.6$	$E_{10} \approx 183.4$
$e_t = e_{export} = 0$	$E_{01} \approx 27694.4$	$E_{00} \approx 774063.6$

Compared to the original data:

	$e_c = e_{\text{POULTRY}} = 1$	$e_c = e_{\text{POULTRY}} = 0$
$e_t = e_{export} = 1$	$N_{11} = 49$	$N_{10} = 141$
$e_t = e_{export} = 0$	$N_{01} = 27{,}652$	$N_{00} = 774{,}106$

the question is now whether a quantity like the surplus $N_{11} = 49$ over the expected $E_{11} \approx 6.6$ is statistically significant.

Expected Frequencies (cont`)

For these values of N and E, the result for X^2 is

$$X^2(D,t,c) = \sum_{e_t \in \{0,1\}} \sum_{e_c \in \{0,1\}} \frac{(N_{e_t e_c} - E_{e_t e_c})^2}{E_{e_t e_c}} \approx 284$$

We are testing the assumption that the values of the $N_{e_t e_c}$ are generated by two independent probabilities, fitting the three ratios with two parameters $p(t)$ and $p(c)$, leaving one degree of freedom.

There is a tabulated distribution, called the χ^2 distribution (in this case with one degree of freedom) which assesses the statistical likelihood of any value of X^2, as defined above (and is analogous to likelihood of standard deviations from the mean of a gaussian distribution):

p	χ^2 critical
.1	2.71
.05	3.84
.01	6.63
.005	7.88
.001	10.83

The above $X^2 \approx 284 > 10.83$, i.e., giving a less than .1% chance that so large a value of X^2 would occur if *export*/POULTRY were really independent (equivalently a 99.9% chance they're dependent).

Back to NLP from ML ☺

- Text representation
  - Feature vectors
  - Features Engineering
  - Weight
    - 0-1 vectors
    - Count vectors
    - tf*idf

Term Weights: Term Frequency

- More frequent terms in a document are more important, i.e. more indicative of the topic.

  f_{ij} = frequency of term i in document j

- May want to normalize *term frequency* (*tf*) by dividing by the frequency of the most common term in the document:

  $tf_{ij} = f_{ij} / max_i\{f_{ij}\}$

Term Weights: Inverse Document Frequency

- Terms that appear in many *different* documents are *less* indicative of overall topic.

  df_i = document frequency of term *i*

  = number of documents containing term *i*

  idf_i = inverse document frequency of term *i*,

  $= \log_2 (N / df_i)$

  (*N*: total number of documents)

- An indication of a term's *discrimination* power.
- Log used to dampen the effect relative to *tf*.

TF-IDF Weighting

- A typical combined term importance indicator is *tf-idf weighting*:

$$w_{ij} = tf_{ij}\, idf_i = tf_{ij} \log_2 (N / df_i)$$

- A term occurring frequently in the document but rarely in the rest of the collection is given high weight.
- Experimentally, *tf-idf* has been found to work well.

Evaluation of Text Classification

- Benchmark data
  - Reuters-21578
  - 20Newspaper
  - ……

Measures of performance

- If binary classification of M texts as members or not members of class c

Predicted / Actual	c	not c
c	True Positive TP	False Negative FN
not c	False Positive FP	True Negative TN

- Accuracy = (TP + TN) / (TP+FP+TN+FN)
- Precision = TP / (TP + FP)
- Recall = TP / (TP + FN)
- F-measure: trade-off between recall and precision:

$$F = \frac{2PR}{P+R} = \frac{2}{\frac{1}{R}+\frac{1}{P}}$$

- • What about more than 2 classes?

Measures of performance

- Macro-averaging:
  - Compute performance for each class, then average.
- Micro-averaging
  - Collect decisions for all classes, compute contingency table, evaluate.

$$P_{macro} = \frac{1}{n}\sum_{i=1}^{n} P_i$$

$$R_{macro} = \frac{1}{n}\sum_{i=1}^{n} R_i$$

$$F_{macro} = \frac{2 \times P_{macro} \times R_{macro}}{P_{macro} + R_{macro}}$$

$$P_{micro} = \frac{\overline{TP}}{\overline{TP} + \overline{FP}} = \frac{\sum_{i=1}^{n} TP_i}{\sum_{i=1}^{n} TP_i + \sum_{i=1}^{n} FP_i}$$

$$R_{micro} = \frac{\overline{TP}}{\overline{TP} + \overline{FN}} = \frac{\sum_{i=1}^{n} TP_i}{\sum_{i=1}^{n} TP_i + \sum_{i=1}^{n} FN_i}$$

$$F_{micro} = \frac{2 \times P_{micro} \times R_{micro}}{P_{micro} + R_{micro}}$$

Summary

- Represent texts
- Select a classifier model
- Evaluate your system

Thanks

- Some slides are from informatics 2B in University of Edinburgh http://www.inf.ed.ac.uk/teaching/courses/inf2b/ and CS 288 in UC Berkeley http://www.cs.berkeley.edu/~klein/cs288/sp10/