WestlakeNLP

Natural Language Processing

Yue Zhang
Westlake University

西湖大學
WESTLAKE UNIVERSITY

Chapter 2

Counting Relative Frequencies

Contents

WestlakeNLP

Models

- What is a “ model”:
  - An imaginary abstract and simplified version of a subject
  - Makes mathematical calculation feasible
- Probabilistic model:
  - Calculate the probability of a random event
- Take probabilistic language modelling for example:
  - Assign a probability to words or sentences

    e.g. P(I know it) > P(eye no it)

Contents

WestlakeNLP

From coin tossing experiments

Intuition

MLE leads to counting relative frequencies

WestlakeNLP

- Training data: $\mathrm{D} = \{y_1, y_2, \dots, y_n\}$
- Training example: $y_i \in \{head, tail\}$
- Parameter: $P(head) = \theta$
- Condition: the tosses are independent and identically $(i.i.d.)$ distributed
- Training objective: The log likelihood

$$\hat{\theta} = \underset{\theta}{\operatorname{argmax}}\, \mathrm{P(D)} = \underset{\theta}{\operatorname{argmax}}\, \log \mathrm{P(D)}$$

MLE leads to counting relative frequencies

WestlakeNLP

- Derivation

$$P(D) = \theta^k (1-\theta)^{N-k}$$

Let $\frac{\delta log\ P(D)}{\delta\theta} = 0$, we have:

$$\frac{\partial \log P(D)}{\partial\theta} = \frac{\partial\left(\log \theta^k (1-\theta)^{N-k}\right)}{\partial\theta}$$

$$= \frac{\partial\left(k\log\theta + (N-k)\log(1-\theta)\right)}{\partial\theta}$$

$$= \frac{k}{\theta} - \frac{N-k}{1-\theta} = 0 \Rightarrow \hat{\theta} = \frac{k}{N}$$

Contents

Casino dice casting

Casino dice casting

Outcomes: 6

Parameters: $\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6$

Constraint: $\sum_{i=1}^{6} \theta_i = 1$

Parameter estimating using MLE:

If out of N trails, k_i gives the outcome of i, then $\theta_i = \frac{k_i}{N}$

Training a word model

Vocabulary: $V = \{w_1, w_2, \dots, w_{|V|}\}$

$|V|$: the number of words in V

Corpus D

MLE training:

$$P(w) = \frac{\#w \in D}{\sum_{w' \in V}(\#w' \in D)}$$

Contents

Review

- Probabilistic Models (e.g. P(head))
- Model Parameters (e.g. θ)
- Model Training (e.g. , $\theta = \frac{k}{N}$)

  Parameter Estimation (e.g. MLE)

  - Training Data (e.g. D $= \{y_1, y_2, \ldots, y_n\}$)
  - Training Example (e.g. $y_i (i \in [1,2, \ldots, n])$)

Terminology

WestlakeNLP

- **Random variable**:

distinct outcome of a random **event** using a distinct **value**

e.g., head = 0 tail = 1

- **Parameterisation**:

specifies a **calculable equation** to compute probabilities

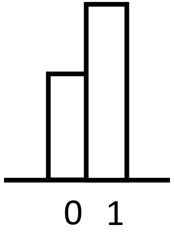
involving the definition of **model parameters**

Probability Distributions

The probabilities of all possible values of a discrete random variable is a **probability distribution**

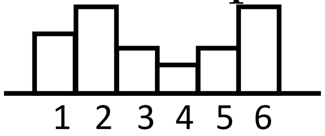
A **Bernoulli distribution** example:

coin tossing



A **categorical distribution** (multinoulli distribution) example:

dice casting, word drawing



MLE training for i.i.d. Bernoulli random variables and categorical random variables leads to **relative frequencies**

Probability Distributions

A **binomial distribution**: the results of n i.i.d. Bernoulli distributions, e.g., for coin tossing problem:

$$P_{BIN}(k,n) = \frac{n!}{k!(n-k)!} P_{BER}(heads)^k \, P_{BER}(tails)^{n-k}$$

A **multinomial distribution**: the results of n i.i.d. categorical distributions e.g., for dice casting problem:

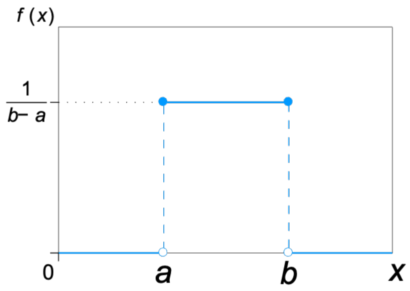
$$P_{MUL}(c_1, c_2, \ldots c_6, n) = \frac{n!}{c_1! \ldots c_6!} P_{CAT}(1)^{c_1} \ldots P_{CAT}(6)^{c_6}$$

Probability Distributions

Continuous random variable:

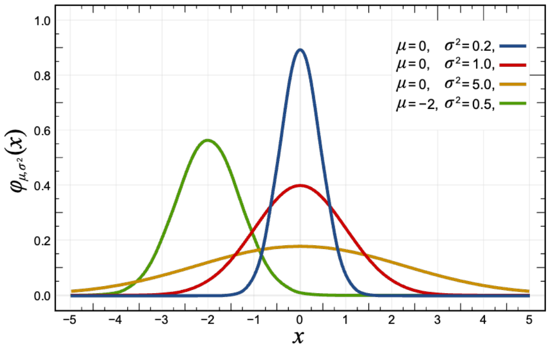
A uniform distribution:

$$f(y) = \frac{1}{b-a} \; for \; y \in [a, b]$$



A Gaussian distribution (or normal distribution):

$$f(y) = \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{(y-\mu)^2}{2\sigma^2})$$



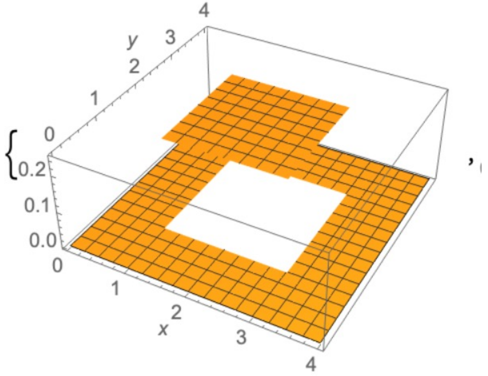
Probability Distributions

Vector random variable:

A uniform distribution:

$$f(x_1, x_2, \dots, x_n) =$$

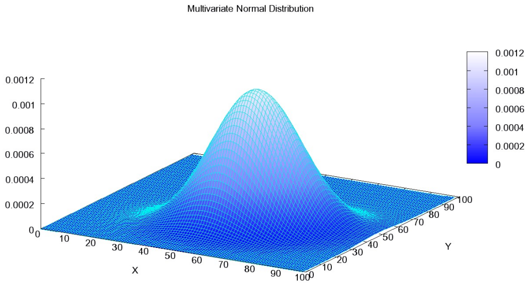
$$\frac{1}{\prod_i^n (H_i - L_i)},\ for\ L_i \le x_i \le H_I, 1 \le i \le n$$



A Gaussian distribution (or normal distribution):

$$f(x_1, x_2, \dots, x_n) =$$

$$\frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}}\ \exp(-\frac{1}{2}(\vec{X} - \vec{\mu})^T \Sigma^{-1} (\vec{X} - \vec{\mu}))$$



Contents

WestlakeNLP

Language Model

A **language model** (LM) measures the probability of natural language sentences, by means of simpler patterns, such as:

- words

  "thanks" is more probable than *"markov"*

- phrases

  P("eat pizza") > P("drink pizza")

- sentences

  P("he said hi") > P("said he hi")

N-gram

- Unigram (bag-of-words)

  hello, hyperbole

- Bigram

  eat pizza, drink pizza

- Trigram

  cat eat mouse, mouse eat cat

Contents

Unigram Language Models

WestlakeNLP

- *i.i.d.* assumption between words in a sentence

$$P(s) = P(w_1)P(w_2)\dots P(w_n) = \prod_f P(w_i)$$

- **Parameter type** : The probability of a word
- **Parameter instances** : |V|

Unigram Language Models

- **out-of-vocabulary** (OOV) word in test data

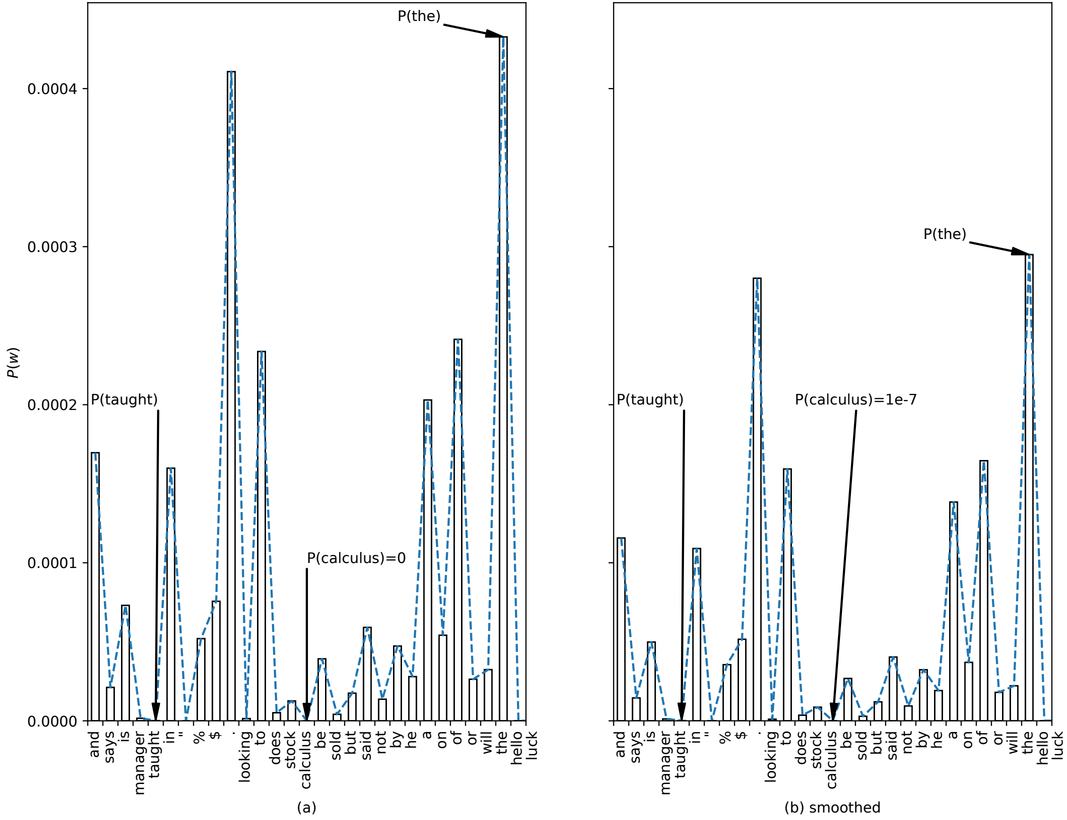
  - *not seen in the training data*

  - *P(OOV) = 0*

  - *P(S) = 0, if OOV ∈ S*

- **add-one smoothing**

$$P(w) = \frac{(\#\mathbf{w} \in D)+1}{\sum_{\mathbf{w}' \in V}((\#\mathbf{w}' \in D)+1)} = \frac{(\#\mathbf{w} \in D)+1}{|V|+\sum_{\mathbf{w}' \in V}(\#\mathbf{w}' \in D)}$$



(a) Unigram distributions. (b) Unigram distributions with add-10 smoothing.

Add-α Smoothing

- Dealing with OOV problem in test data: a more general form of add-one smoothing
- **Hyper-parameter:**
  - fixed in advance and not trained during training
  - can be tuned, selected empirically to improve performance
- Add-α smoothing: introduces a hyper-parameter α

$$P(w) = \frac{(\#w \in D) + \alpha}{\sum_{w' \in V}((\#w' \in D) + \alpha)}$$

Contents

Conditional probability

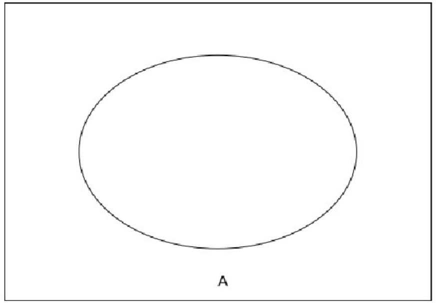
- Unigram language models face challenge in comparing *"he ate pizza"* and *"he drank pizza"*, which requires knowledge on verb-object relations

- While bigram language models compute **conditional probabilities** $P(w_2|w_1)$.

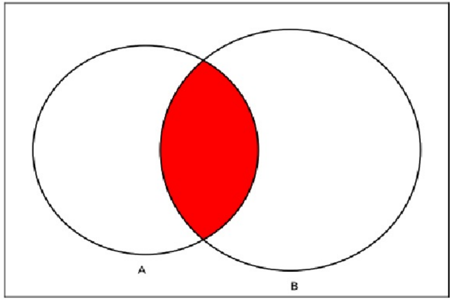
  e.g. $P(pizza|ate) > P(pizza|drank)$

Conditional probability

Unconditional probabilities and conditional probabilities



(a) $P(A) = \frac{\text{AREA}(A)}{\text{AREA}(\Box)}$



(b) $P(B|A) = \frac{\text{AREA}(A \cap B)}{\text{AREA}(A)}$

$$P(B|A) = \frac{AREA(A \cap B)}{AREA(A)} = \frac{\frac{AREA(A \cap B)}{AREA(A \cap B)}}{\frac{AREA(A)}{AREA(\Box)}} = \frac{P(A,B)}{P(A)}$$

P(A,B) is the **joint probability** of A and B

Training bigram language models

WestlakeNLP

- Bigram language models compute conditional probabilities $P(w_2|w_1)$ for bigrams w_1w_2
- Training data: D consisting of a set of sentences
- Given D: MLE for the conditional probabilities:

$$P(\mathbf{w}_2|\mathbf{w}_1) = \frac{(\#\mathbf{w}_1\mathbf{w}_2 \in D)}{\sum_{\mathbf{w}\in V}(\#\mathbf{w}_1\mathbf{w} \in D)}$$

Training bigram language models

- Reducing sparsity:

  Back-off

$$P_{\text{backoff}}\left(\mathbf{w}_2 | \mathbf{w}_1\right) = \lambda P\left(\mathbf{w}_2 | \mathbf{w}_1\right) + (1-\lambda) P\left(\mathbf{w}_2\right)$$

  λ is a hyper-parameter which can be set empirically.

Calculating the probability of a sentence

- Sentence: $S = \langle s \rangle w_1 w_2 \ldots w_n \langle /s \rangle$

  $\langle s \rangle$: the beginning of a sentence

  $\langle /s \rangle$: the end of a sentence

- Conditional probabilities of bigrams: $P(w_i|w_{i-1})$
- According to bigram language model:

$$P(s) = P\left(w_1 w_2 \ldots w_n \langle /s \rangle | \langle s \rangle\right)$$
$$= P\left(w_1 | \langle s \rangle\right) P\left(w_2 | w_1\right) \ldots$$
$$P\left(w_n | w_{n-1}\right) P\left(\langle /s \rangle | w_n\right)$$

Derivation

- **Chain rule**

$$P(s) = P(w_1 w_2 \dots w_n \langle/s\rangle | \langle s\rangle)$$
$$= P(w_1|\langle s\rangle) P(w_2|\langle s\rangle w_1) \dots$$
$$P(\langle/s\rangle|\langle s\rangle\, w_1 w_2 \dots w_n)$$

- **Conditional independence** assumptions in bigram language models:

$$P(w_i|\langle s\rangle w_1 \dots w_{i-1}) = P(w_i\, |w_{i-1})$$

- Result

$$P(s) = P(\langle s\rangle) P(w_1|\langle s\rangle) P(w_2|w_1) \dots P(\langle/s\rangle|w_n)$$

Contents

Trigram language modelling

WestlakeNLP

- Random variable

$$s = w_1 w_2 \dots w_n \quad \Rightarrow \quad s = \langle s \rangle \langle s \rangle w_1 w_2 \dots w_n \langle /s \rangle$$

- Modelling target P(s)
- Parameterisation
  1. Chain rule

$$P(s) = P(w_1 w_2 \dots w_n \langle /s \rangle | \langle s \rangle \langle s \rangle)$$
$$= P(w_1 | \langle s \rangle \langle s \rangle)\, P(w_2 | \langle s \rangle \langle s \rangle w_1)\, P(w_3 | \langle s \rangle \langle s \rangle w_1 w_2)$$
$$\dots P(\langle /s \rangle | \langle s \rangle \langle s \rangle w_1 w_2 \dots w_n)$$

  1. Independence assumptions

$$P(w_\mathrm{i}) = P(\langle s \rangle \langle s \rangle w_1 w_2 \dots w_{\mathrm{i}-1})$$
$$\Rightarrow P(s) = P(w_1 | \langle s \rangle \langle s \rangle)\, P(w_2 | \langle s \rangle w_1) \dots P(\langle /s \rangle | w_{n-1} w_n)$$

Trigram language modelling

WestlakeNLP

- Modelling target: P(s)

- Parameterised model form

$$P(s) = P(w_1|\langle s\rangle\langle s\rangle)\ P(w_2|\langle s\rangle w_1)\ \ldots P(\langle/s\rangle|w_{n-1}w_n)$$

- Parameters
  - One type: $P(w_3|w_1w_2)$
  - O($|V|^3$) instances

Trigram language modelling

WestlakeNLP

- Training — MLE

$$P(w_3|w_1w_2) = \frac{(\#w_1w_2w_3 \in D)}{\sum_{w \in V}(\#w_1w_2w \in D)}$$

  - Relative frequency of w_3 under the **context** (or **history**) w_1w_2

- **Sparsity** — backoff

$$P_{backoff}(w_3|w_1w_2) = \lambda_1 P(w_3|w_1w_2) + \lambda_2 P(w_3|w_2) + \lambda_3 P(w_3)$$

$$\text{s.t., } \lambda_1 + \lambda_2 + \lambda_3 = 1;\ \lambda_i > 0, i \epsilon \{1,2,3\}$$

  - $P(w_3)$ can be smoothed
  - Can $P(w_3|w_1w_2)$ be smoothed be smoothed directly?

Methods to address sparsity

- add-one smoothing: add one to the count of all words
- add-α smoothing: add α to the count of all words
- back-off: use lower order n-gram probabilities to approximate high order n-gram probabilities
- Good-Turing smoothing: make a rational guess of the count of OOV words
- Knesser-Ney smoothing: work with back-off, consider the history context of lower order n-gram

Log-probability models

Calculating $logP(s)$ to avoid small values:

$$\log\left(\prod_{i=1}^{n+1} P\left(w_i \middle| w_{i-2} w_{i-1}\right)\right) = \sum_{i=1}^{n+1} \log P\left(w_i \middle| w_{i-2} w_{i-1}\right)$$

Different n-grams

WestlakeNLP

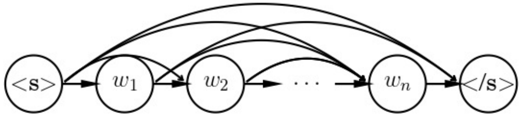
Model	Samples
Unigram	out this like there Against me you, made? he Cupid to thou too thee My he tricks that heart one thing face as not fear she on face Athens. let Good and and, kiss affection a PRINCE ?
Bigram	All my sometime like himself, –What's master. As much good news? tell you foolish thought. Can it like a man whom there but it is eaten up Lancaster and it, sir? Away! why
Trigram	Where is the lady of the house of York. My servant, Ariel, thy blood and made to understand you, hear me speak a word, Mortimer! We should have had such faults; makes him to this woman to bear him home. Those that betray them do it secretly, alone, and I will believe thou hast done!

Contents

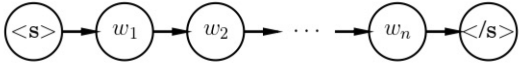
WestlakeNLP

Generative Models

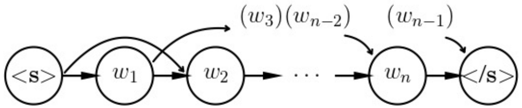
- **Generative story** treats sentence as being generated from left to right



  - First-order Markov model



  - Second-order Markov model



  - Naïve Bayes model

Hands on

1. A small corpus abridged from Alice in Wonderland:

Do cats eat bats?

Do bats eat cats?

Now, Dinah, tell me the truth: did you ever eat bats?

Bigram model: $P(w_i|w_{i-1}) = \frac{\#(w_{i-1}, w_i)}{\#(w_{i-1})}$

$P(\text{bats} \mid \text{eat}) = \frac{2}{3}$

$P(\text{cats} \mid \text{eat}) = \frac{1}{3}$

$P(\langle/s\rangle \mid ?) = \frac{3}{3} = 1$

Hands on

2. Berkeley Restaurant Project (BeRP):

http://www.icsi.berkeley.edu/ftp/pub/speech/wooters/berp.tgz

Examples:

- can you tell me about any good cantonese restaurants close by
- mid priced thai food is what i'm looking for
- tell me about chez panisse
- can you give me a list of the kinds of food that are available
- i'm looking for a good place to eat breakfast
- when is caffe venezia open during the day

Unigram counts

i	want	to	eat	Chinese	food	lunch	spend
2533	927	2417	746	158	1093	341	278

Hands on

Normalize by unigrams

-	i	want	to	eat	Chinese	food
i	0.002	0.33	0	0.0036	0	0
want	0.0022	0	0.66	0.0011	0.0065	0.0065
to	0.00083	0	0.0017	0.28	0.00083	0
eat	0	0	0.0027	0	0.021	0.0027
chinese	0.0063	0	0	0	0	0.52
food	0.014	0	0.014	0	0.00092	0.0037
lunch	0.0059	0	0	0	0	0.0029
spend	0.0036	0	0.0036	0	0	0

Bigram estimates of sentence probabilities

$P(\langle s\rangle \mid I\ want\ Chinese\ food\ \langle/s\rangle)$

$= P(I|\langle s\rangle \times P(want|I) \times P(chinese|want) \times P(food|chinese) \times$

$P(\langle/s\rangle|food) = 0.000183$

- P(chinese | want)=0.0065
- PP(english | want)=0.0011
- PP(to | want)=0.66
- PP(eat | to)=0.28
- PP(food | to)=0
- PP(want | spend)=0
- PP(I | $\langle s\rangle$)=0.25

These values tell facts about world or grammar

Contents

Contents

WestlakeNLP

Text classification under MLE

- Input: text document $d = w_1 w_2 \dots w_n$
- Output: class $c \in C$
- Corpus of documents: $D = \{(d_i, c_i)\}|_{i=1}^{N}$
- Modeling Target: P(c | d)
- Parameterisation: taking $P(c|d)$ as model parameters directly?

$$P(c|d) = \frac{\#(d,c) \in D}{\#d \in D},$$

too **sparse**.

Needs more computable parameterisation

The Bayes rule

- From the equation of conditional probability:

$$P(B|A) = \frac{P(AB)}{P(A)}$$

- We have

$$P(AB) = P(A|B)P(B) = P(B|A)P(A)$$

- Therefore

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

Which is the **Bayes rule.**

Naïve Bayes model parameterisation process

WestlakeNLP

- Given a document d and a class c

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

- Under conditional independent assumption (bag of words):

$$P(d|c) = P(w_1|c)P(w_2|c)\dots P(w_n|c)$$

- The final form of a **Naïve Bayes Classifier** is

$$P(c|d) \propto P(d|c)P(c) \approx \prod_i P(w_i|c)P(c)$$

Training a Naïve Bayes classifier

WestlakeNLP

- Given $D = \{(d_i, c_i)\}|_{i=1}^{N}$,

  the probability P(c) can be estimated using MLE:

$$P(c) = \frac{\#c \in D}{\sum_{c'} (\#c' \in D)} = \frac{\#c \in D}{|D|}$$

- For each w and c pair, $P(w|c)$ can be estimated using MLE:

$$P(\mathbf{w}|c) = \frac{\#(\mathbf{w}, c) \in D}{\sum_{\mathbf{w}'} (\# (\mathbf{w}', c) \in D)}$$

Calculating $logP(c)$ and $logP(w|c)$ as model parameters

log$(c|d)$ to score candidate class labels.

Naïve Bayes text classification

WestlakeNLP

- Testing:

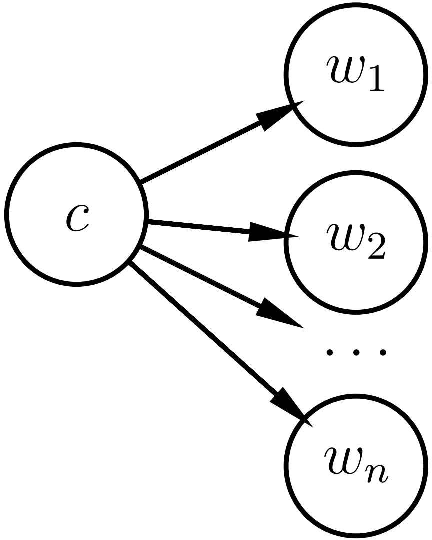
$$\hat{c} = \arg\max_{c \in C} P(c|d)$$

$$= \arg\max_{c \in C} \frac{P(d|c)P(c)}{P(d)} = \arg\max_{c \in C} P(d|c)P(c)$$

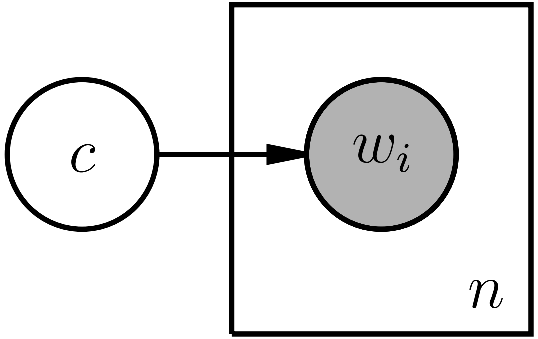
$$= \arg\max_{c \in C} P(c)\, P(w_1|c)P(w_2|c) \ldots P(w_n|c)$$

- Parameters
  - Two types: $P(c)$, $P(w|c)$
  - $|C| + |V||C|$ instances

Generative models



(a) Naïve Bayer model



(b) Naïve Bayer model (nested plate notation)

Hands on

3.International news classification

a: US news, i: Iran news, $D = d_i|_{i=1}^{4}$

-	Doc	Words	Class
Training	1	US, Washington, US	a
	2	US, US, New York	a
	3	US, The White House	a
	4	Tehran, Iran, US	i
Test	5	US, US, US, Tehran, Iran	?

Hands on

Calculate with add-one smoothing:

$$\hat{P}(c) = \frac{\#c \in D}{|D|}, P(w|c) = \frac{\#(w,c) \in D + 1}{\sum_{w'}(\#(w',c) \in D) + |V|}$$

Priors

$$P(a) = \frac{3}{4}, P(i) = \frac{1}{4}$$

Conditional Probabilities

$$P(US|a) = \frac{5+1}{8+6} = \frac{3}{7}, P(Tehran|a) = \frac{0+1}{8+6} = \frac{1}{14}$$

$$P(Iran|a) = \frac{0+1}{8+6} = \frac{1}{14}, P(US|i) = \frac{1+1}{3+6} = \frac{2}{9}$$

$$P(Tehran|i) = \frac{1+1}{3+6} = \frac{2}{9}, P(Iran|i) = \frac{1+1}{3+6} = \frac{2}{9}$$

Hands on

Text classification with Naïve Bayes classifier:

$$P(c|d) \propto P(d|c)P(c) \approx \prod_i P(w_i|c)p(c)$$

The probable class of test data:

$$P(a|d_5) \propto \frac{3}{4} \times \frac{3}{7} \times \frac{1}{14} \times \frac{1}{14} = 0.0003$$

$$P(i|d_5) \propto \frac{1}{4} \times \frac{2}{9} \times \frac{2}{9} \times \frac{2}{9} = 0.0001$$

So test data d_5 is assigned to US news.

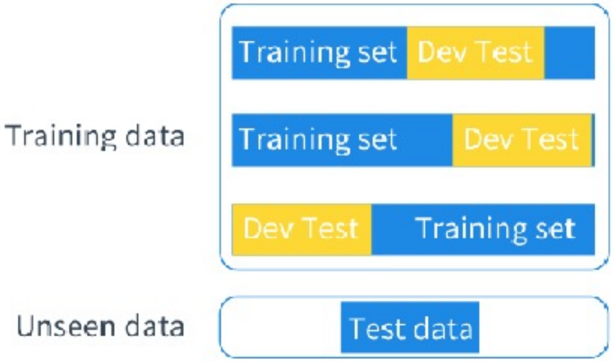
Contents

WestlakeNLP

Evaluating a Text Classifier

Data

- Training set: estimate model parameters
- Test set: get final results
- Development set: adjust hyper-parameters



Process

Train -> dev test ->Train -> dev test -> … Train -> dev test -> test

strong results

only once

Evaluation metric

- Accuracy

$$Acc = \frac{\#\,Correct}{\#\,Total}$$

Contents

Features in NLP

- Features are patterns that are used to parameterise a model
  - word: $P(w)$
  - n-gram: $P(w_2 | w_1)$, $P(w_3 | w_1\ w_2)$
  - word-class pair: $P(w | c)$
- With more features, we can obtain more evidences for making a correct prediction
- But we need to avoid **overlapping features** for generative models (e.g., $P(w)$, $P(w | c)$)

Summary

- Probabilistic modelling and parametrisation techniques
- Maximum likelihood estimation
- N-gram language models
- Naive Bayes models for text classification

Resources

- Language modelling toolkits:

SRILM
http://www.speech.sri.com/projects/srilm/

Google N-gram Release
http://googleresearch.blogspot.com/2006/08/all-our-n-gram-are-belong-to-you.html

Google Book N-grams
http://ngrams.googlelabs.com/