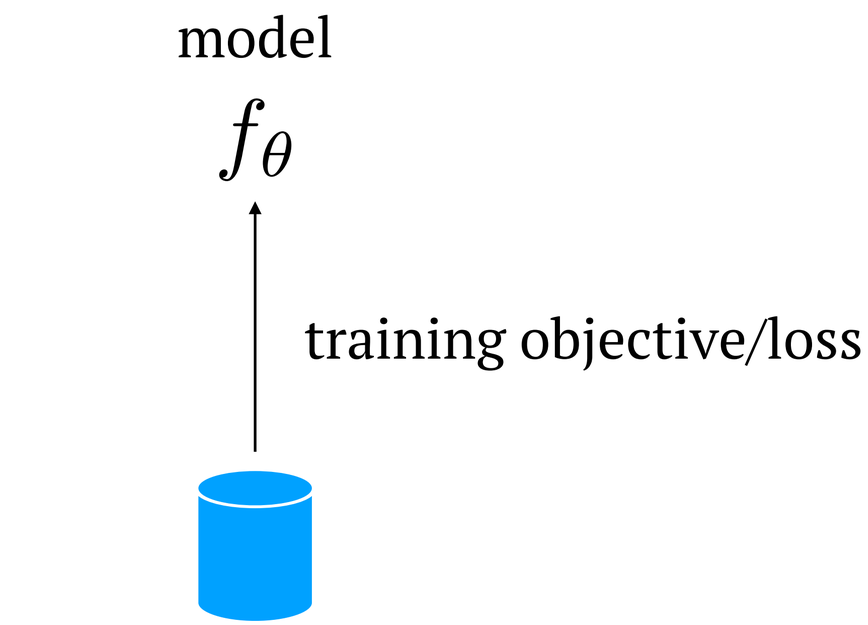
Introduction to NLP, Language Models

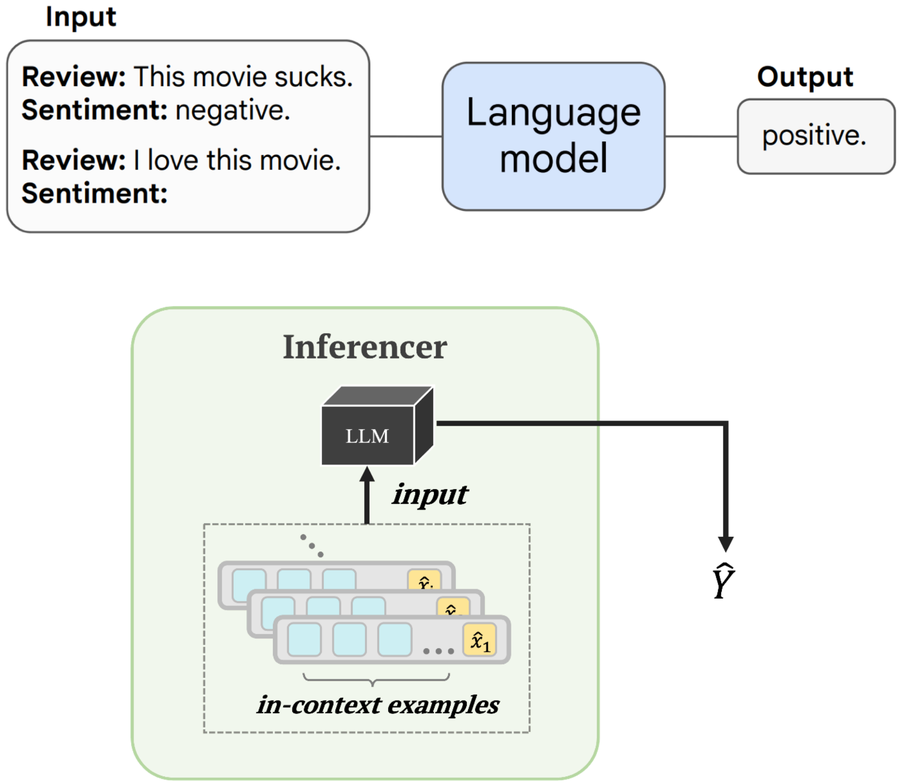
COMP7607 — Week 1

Lingpeng Kong
Department of Computer Science, The University of Hong Kong

Machine Learning (Today)



Supervised Learning

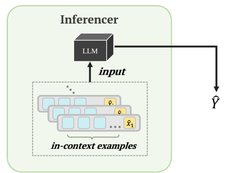


Zero-shot / In-Context Learning

Testing the Upper Limits of Reasoning Ability

Success (w/ Subgoal-Based Proof)

```
theorem mathd_numbertheory_435:
  fixes k :: nat
  assumes "0 < k"
    and "\<forall> n. gcd (6 * n + k) (6 * n + 3) = 1"
    and "\<forall> n. gcd (6 * n + k) (6 * n + 2) = 1"
    and "\<forall> n. gcd (6 * n + k) (6 * n + 1) = 1"
  shows "5 \<le> k"
proof -
  (* We will show that for any positive integer n, 6n + k is relatively prime to each of
6n + 3, 6n + 2, and 6n + 1 (This is true by assumption). *)
  have "\<forall> n. gcd (6 * n + k) (6 * n + 3) = 1"
    using assms(2)
    sledgehammer
  moreover have "\<forall> n. gcd (6 * n + k) (6 * n + 2) = 1"
    using assms(3)
    sledgehammer
  moreover have "\<forall> n. gcd (6 * n + k) (6 * n + 1) = 1"
    using assms(4)
    sledgehammer
  (* We will show that for any positive integer n, 6n + k is relatively prime to each of
6n + 3, 6n + 2, and 6n + 1 (This is true by assumption). *)
  ultimately have "\<forall> n. coprime (6 * n + k) (6 * n + 3) \<and> coprime (6 * n
+ k) (6 * n + 2) \<and> coprime (6 * n + k) (6 * n + 1)"
    sledgehammer
  (* Now we will show that k is at least 5. Suppose that k < 5. Then k = 1, 2, 3, 4. We will
show that for each of these values of k, there exists a positive integer n such that 6n +
k is not relatively prime to one of 6n + 3, 6n + 2, and 6n + 1. *)
  have "k \<ge> 5"
  proof (rule ccontr)
    assume "\<not> 5 \<le> k"
    then have "k < 5"
      sledgehammer
    with assms(1)
    consider "k=1" | "k=2" | "k=3" | "k=4"
      sledgehammer
    then show False
    proof cases
      case 1
      then have "gcd (6 * 2 + k) (6 * 2 + 1) \<noteq> 1"
        sledgehammer
      then show False
        sledgehammer
    next
      case 2
      then have "gcd (6 * 5 + k) (6 * 5 + 3) \<noteq> 1"
        sledgehammer
      then show False
        sledgehammer
    next
      case 3
      then have "gcd (6 * 4 + k) (6 * 4 + 2) \<noteq> 1"
        sledgehammer
      then show False
        sledgehammer
    next
      case 4
      then have "gcd (6 * 2 + k) (6 * 2 + 3) \<noteq> 1"
        sledgehammer
      then show False
        sledgehammer
    qed
  qed
  (* Thus k is at least 5. *)
  then show?thesis
    sledgehammer
qed
```



1

Statement

If $\gcd(n, 4) = 1$ and $\mathrm{lcm}(n, 4) = 28$, show that n is 7.

x

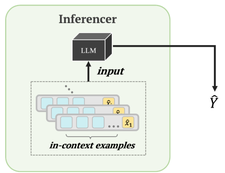
Informal proof

We know that $\gcd(a, b) \cdot \mathrm{lcm}(a, b) = ab$, hence $1 \cdot 28 = n \cdot 4$.

Then $n = 1 \cdot 28/4 = 7$,

completing the proof. ■

y



2

Informal proof

We know that $\gcd(a, b) \cdot \mathrm{lcm}(a, b) = ab$, hence $1 \cdot 28 = n \cdot 4$.

Then $n = 1 \cdot 28/4 = 7$,

completing the proof. ■

x

Formal sketch

```
have c1: "1*28 = n*4"
using assms
  <proof>
then have c2: "n = 1*28/4"
  <proof>
then show ?thesis
  <proof>
```

y

SubgoalXL successfully solves **41 AMC12, 9 AIME**, and **3 IMO** problems from miniF2F. achieves a new state-of-the-art performance of 56.1% in Isabelle on the standard miniF2F dataset

Xueliang Zhao et al., 2023, 2024; https://www.arxiv.org/pdf/2408.11172; https://arxiv.org/abs/2305.16366

Why not every AI is that powerful?

What can I help you with?

Play a good song.

Sorry, I couldn’t find ‘a good song’ in your music.

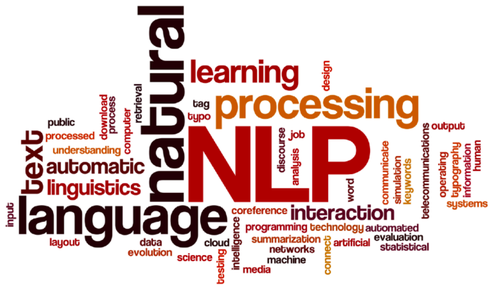
You need to do a better job understanding me.

Noted.

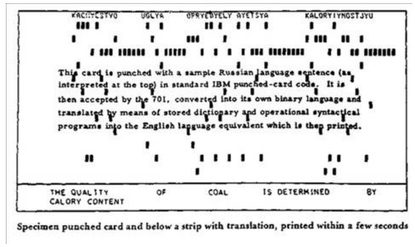
Yeah, make a note of that.

Here’s your note:

Demystify — what's inside?

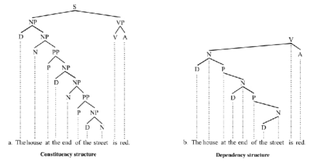


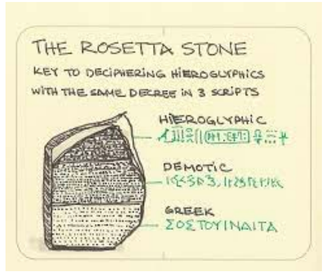
Symbolic NLP



Specimen punched card and below a strip with translation, printed within a few seconds

Statistical NLP





Neural NLP

1950s - early 1990s

1990s - 2010s

2010s - present

What is NLP? Wait, what is language?

The abstraction of the real world — different languages take you to different worlds!

餃子

CX 844
Hong Kong -> New York

dumplings

嗩吶

CX 596
Hong Kong -> Osaka

チャルメラ

Something that makes sharp long voice, like screaming???

Do AI “understand”? Let’s play a game!

The cat is thrown out of the _____

door, window, dog

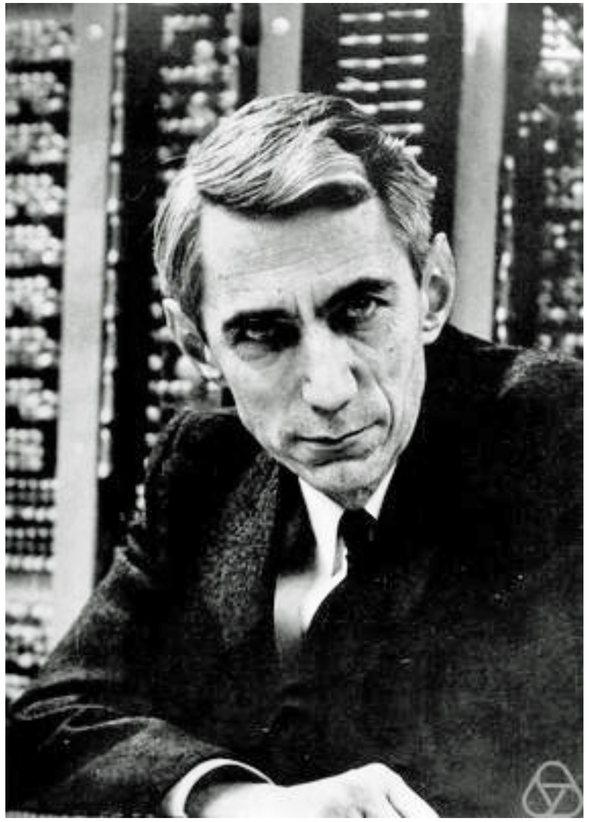
This year, I am going to do an internship in _____

Queen Mary Hospital, HSBC, Google, Amazon

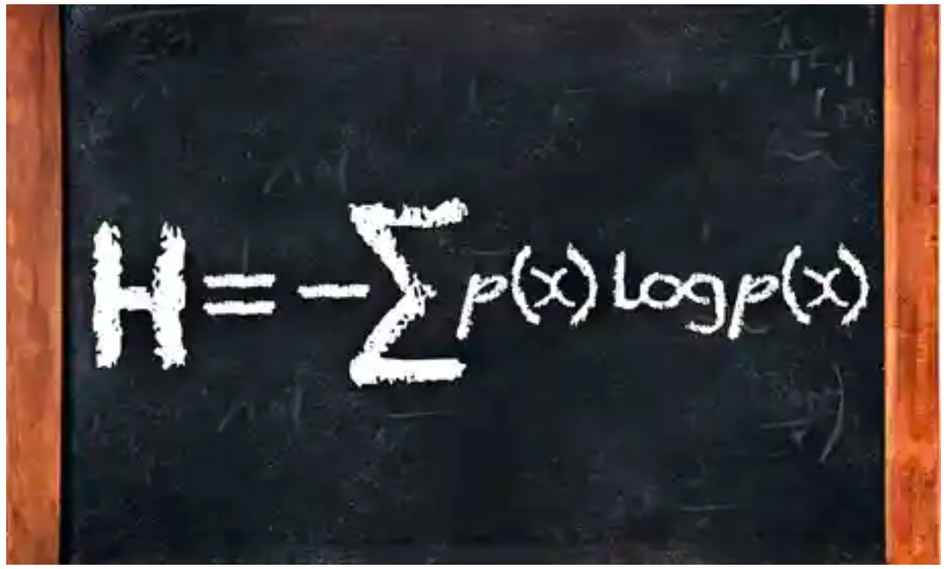
Majoring in computer science, this year, I am going to do an internship in _____

Queen Mary Hospital, HSBC, Google, Amazon

Shannon Game



Claude Elwood Shannon
(April 30, 1916 – February 24, 2001)



Information Theory; Entropy

Language models, and how to build it.

Dice, and how do we roll them
(probabilistic model)

Transformers, neural networks and many others
(powerful functions, and inside configurations)

Generative Language Model

I am going to do an internship in Google

Making the dice

bag of words
(@Carnegie Mellon University)

1	Belief
2	Evidence
3	Reason
4	Claim
5	Think
6	Justify
7	Also
	…
99	Therefore
100	Google

Vocabulary

Generative Language Model

I

I

Generative Language Model

I am

am

Generative Language Model

I am going

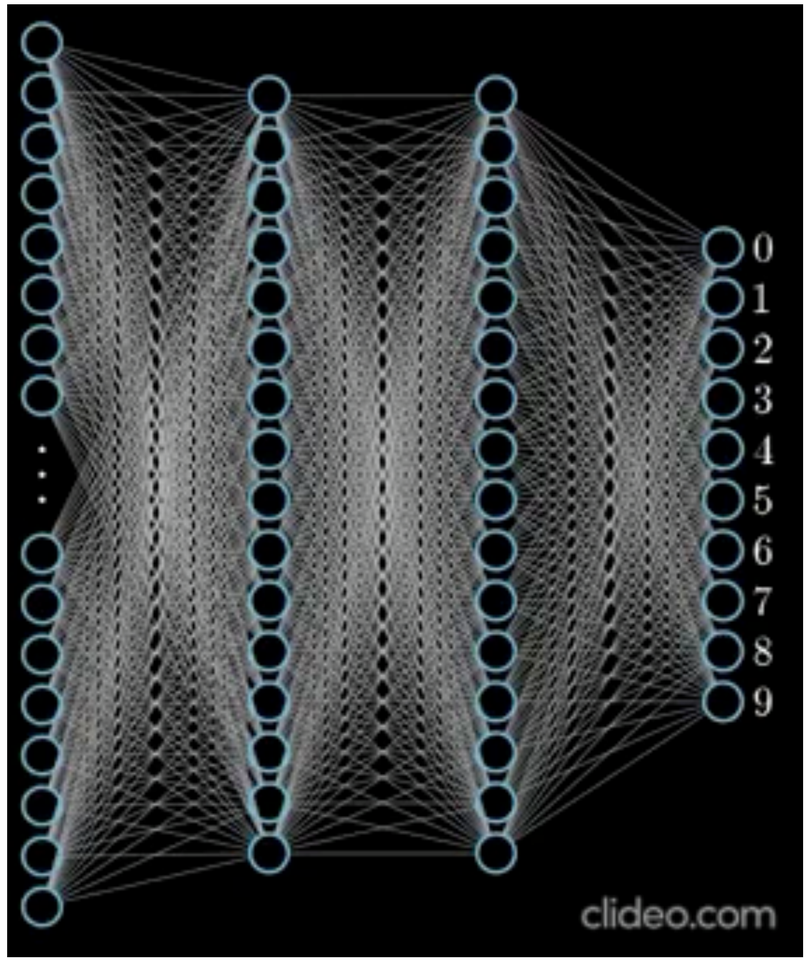
going

Generative Language Model

I am going to do an internship in Google

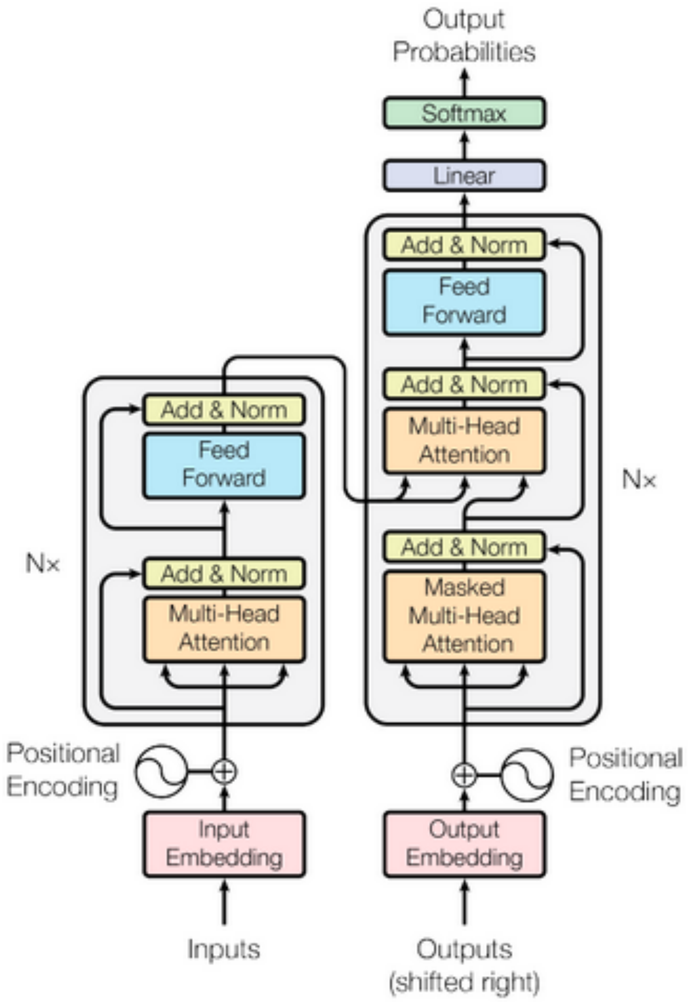
Google

Neuralize the dice!



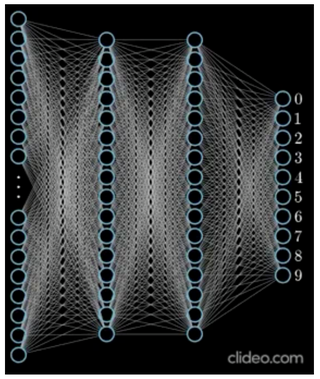
3Blue1Brown

Neural Networks (e.g. Transformers)



Generative Language Model

I am going to do an internship in Google



Google

Language models, and how to build it.

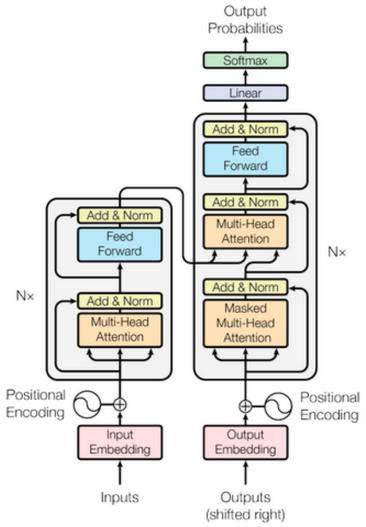
Dice, and how do we roll them
(probabilistic model)

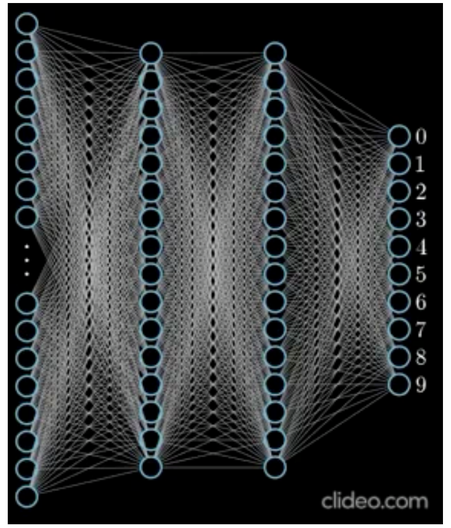
Transformers, neural networks and many others
(powerful functions)

$$p(\boldsymbol{x}) = \prod_i p(x_i|\boldsymbol{x}_{<i})$$

Learn

Parameterize





First problem — the language modeling problem

Given a finite vocabulary

$$\mathcal{V} = \{\text{belief}, \text{evidence}, \text{reason}, \text{claim}, \ldots \text{Google}, \text{therefore}\}$$

We have an infinite set of strings, $\mathcal{V}^\dagger$

<s> I am going to an internship in Google </s>
<s> an internship in Google </s>
<s> I am going going </s>
<s> Google is am </s>
<s> internship is going </s>

Formally:

$$p(x_1, x_2, \ldots x_n)$$
$$p(x_i \mid x_{i-1}, x_{i-2}, \ldots x_1)$$

Can we learn a "model" for this "generative process"? We need to "learn" a probability distribution:

$$\sum_{x \in \mathcal{V}^\dagger} p(x) = 1, \; p(x) \geq 0 \text{ for all } x \in \mathcal{V}^\dagger$$

Learn from what we've seen

The Language Modeling Problem

Given a *training sample* of example sentences, we need to "learn" a probabilistic model that assigns probabilities to every possible string:

p(<s> I am going to an internship in Google </s>) = 10^{-12}

p(<s> an internship in Google </s>) = 10^{-8}

p(<s> I am going going </s>) = 10^{-15}

…

It is a probability distribution p over strings, i.e., p is a function that satisfies

$$\sum_{x \in \mathcal{V}^\dagger} p(x) = 1,\ p(x) \geq 0 \text{ for all } x \in \mathcal{V}^\dagger$$

The Language Modeling Problem

<s> Sam I am </s>

<s> I am Sam </s>

<s> I do not like green eggs and ham </s>

Training Corpus

Is this a good model?

P(<s> Sam I am </s>) = 1/3

P(<s> I am Sam </s>) = 1/3

P(<s> I do not like green eggs and ham </s>) = 1/3

P(<s> I am </s>) = 0

P(<s> green Sam </s>) = 0

Naïve Model

The Language Modeling Problem

<s> Sam I am </s>

<s> I am Sam </s>

<s> I do not like green eggs and ham </s>

Training Corpus

The probability of the word “<s>” followed by the word “I”:

$$P(I \mid <s>) = 2/3$$

The Language Modeling Problem

<s> Sam I am </s>

<s> I am Sam </s>

<s> I do not like green eggs and ham </s>

Training Corpus

P(I | <s>) = 2/3

P(</s> | Sam) = 1/2

Naïve Model

The Language Modeling Problem

<s> Sam I am </s>

<s> I am Sam </s>

<s> I do not like green eggs and ham </s>

Training Corpus

P(I | <s>) = 2/3

P(am | I) = ?

P(</s> | Sam) = 1/2

Naïve Model

The Language Modeling Problem

<s> Sam I am </s>

<s> I am Sam </s>

<s> I do not like green eggs and ham </s>

Training Corpus

P(I | <s>) = 2/3

P(am | I) = 2/3

P(</s> | Sam) = 1/2

Naïve Model

The Language Modeling Problem

<s> Sam I am </s>

<s> I am Sam </s>

<s> I do not like green eggs and ham </s>

Training Corpus

$$P(\text{<s> Sam I am </s>}) = P(\text{Sam} \mid \text{<s>}) * P(\text{I} \mid \text{Sam}) * P(\text{am} \mid \text{I}) * P(\text{</s>} \mid \text{am})$$

Bi-gram Model

Course Logistics

Course Logistics

Website:

https://nlp.cs.hku.hk/comp7607-fall2024

Prerequisites:

COMP3314 or COMP3340, MATH1853

We will assume a lot things from Machine Learning, Statistics, and Programming

This NLP course will be very difficult if you haven't taken these courses.

Assessment:

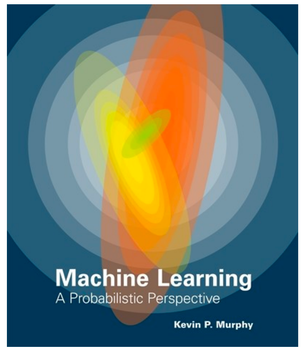
50% continuous assessment, 50% final project

TA:

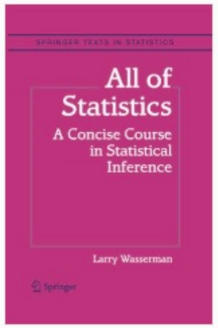
Qington Li (https://qtli.github.io/), Qiushi Sun (https://qiushisun.github.io/)

Course Logistics

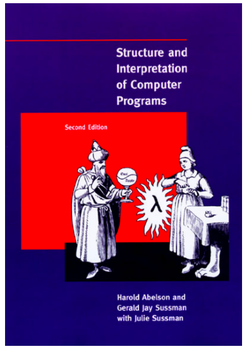
We will assume a lot things from Machine Learning, Statistics, and Programming



Supervised learning, unsupervised learning, regression, classification, loss function, neural networks, regularization … (COMP3314)



Random variables, joint probability, conditional probability, Bayes' theorem …

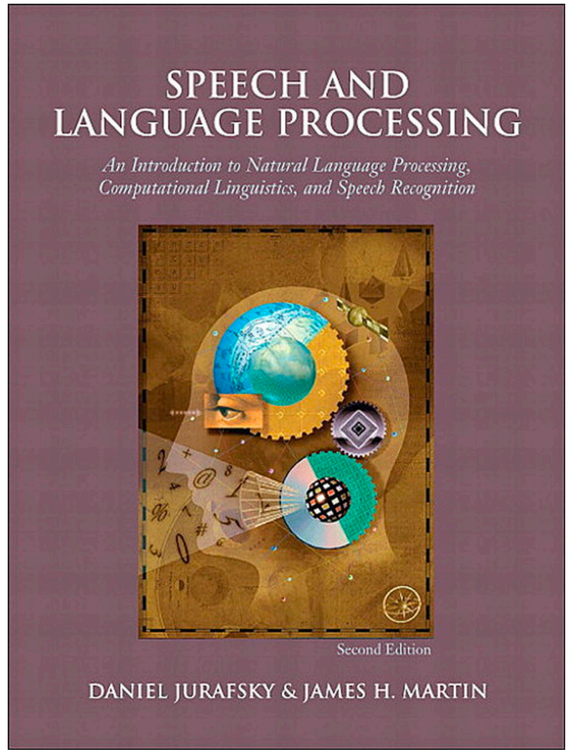


Data structures, dynamic programming, time/space complexity …

Course Logistics

Textbook recommendation (J&M):

https://web.stanford.edu/~jurafsky/slp3/



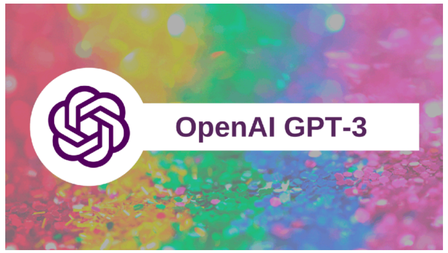
Assessments (in total ~3):

- Programming problems
- Problem sets

Honor code:

> You are free to form study groups and discuss homeworks and projects. However, you must write up homeworks and code from scratch independently, and you must acknowledge in your submission all the students you discussed with.

What's Next?



BERT, GPT-3, Word2vec, Glove, T5 ...

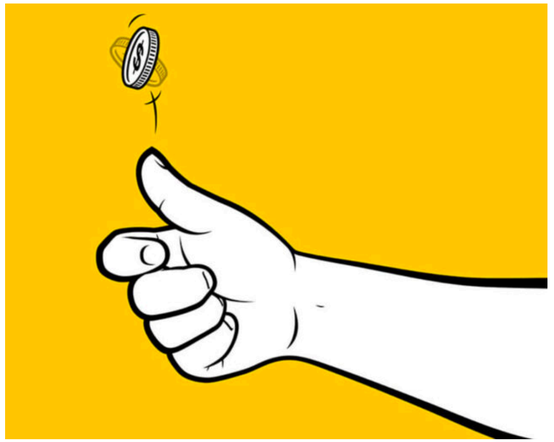
N-Gram Models, Hidden Markov Models ...

LSTMs, Recurrent Neural Networks, MLP, Transformers ...

What is probability?

P(head) = 50%

Frequentist



events (repeated trails)

Bayesian

uncertainty (ignorance)

Probability

Joint probability:

$$\Pr(A \wedge B) = \Pr(A, B)$$

Probability of a union of two events:

$$\Pr(A \vee B) = \Pr(A) + \Pr(B) - \Pr(A \wedge B)$$

Conditional probability:

$$\Pr(B|A) \triangleq \frac{\Pr(A, B)}{\Pr(A)}$$

If A and B are independent events:

$$\Pr(A, B) = \Pr(A) \Pr(B)$$

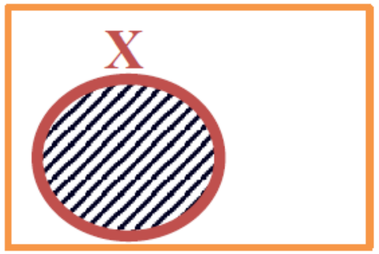
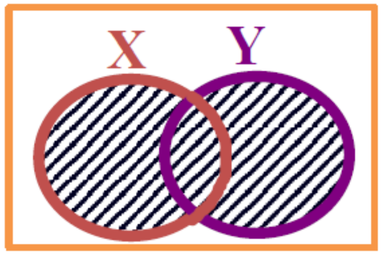
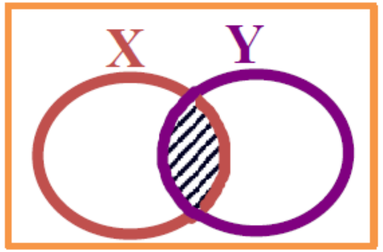
If the events are mutually exclusive:

$$\Pr(A \vee B) = \Pr(A) + \Pr(B)$$

Conditional independence:

$$\Pr(A, B|C) = \Pr(A|C) \Pr(B|C)$$

Probability

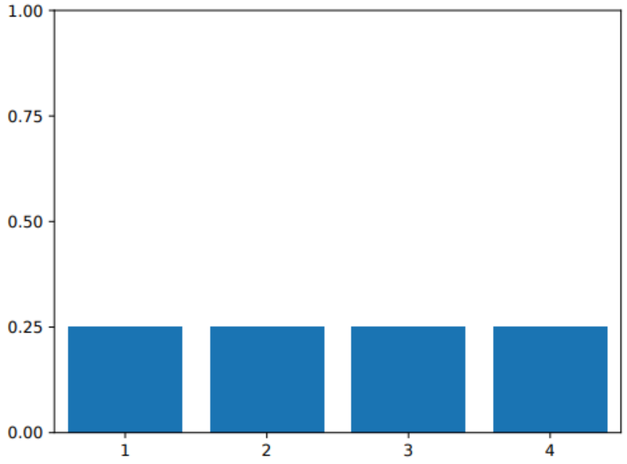
Marginal	Union	Joint	Conditional
$P(X)$	$P(X \cup Y)$	$P(X \cap Y)$	$P(X \mid Y)$
The probability of **X** occurring	The probability of **X or Y** occurring	The probability of **X and Y** occurring	The probability of **X occurring given that Y** has occurred
			

https://medium.com/budding-data-scientist/data-analytics-using-python-part-2-d55715e5c1f9

Random Variables

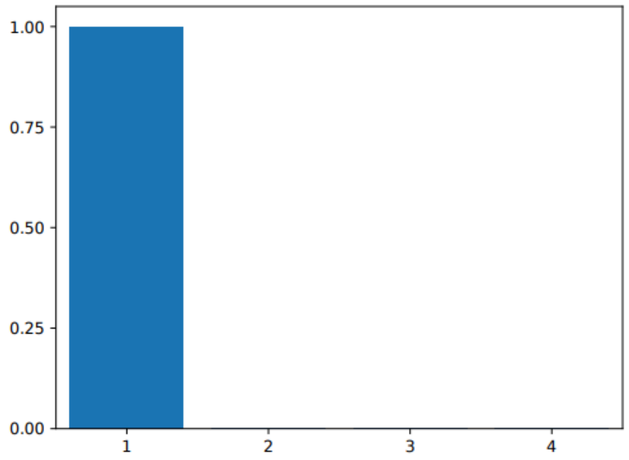
X represents some unknown quantity of interest

We call it a **random variable** if the value of X is unknown and/or could change.



(a)

uniform distribution

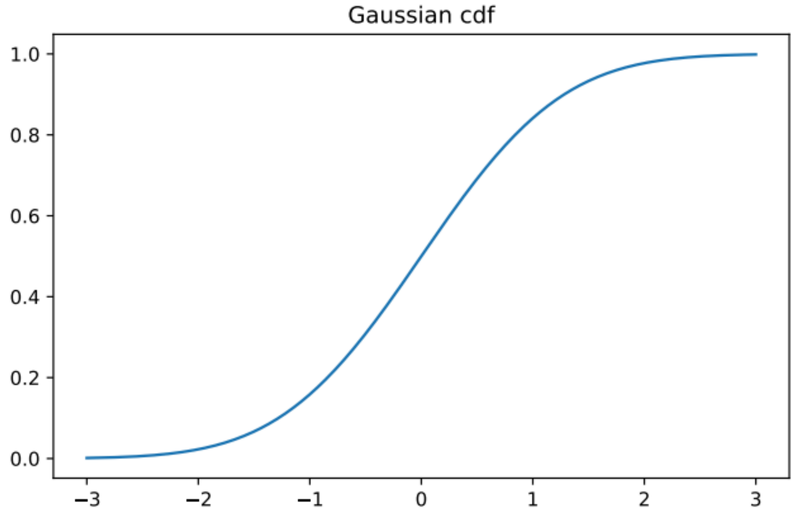


(b)

degenerate distribution
(delta function)

probability mass function (pmf): $p(x) \triangleq \Pr(X = x)$

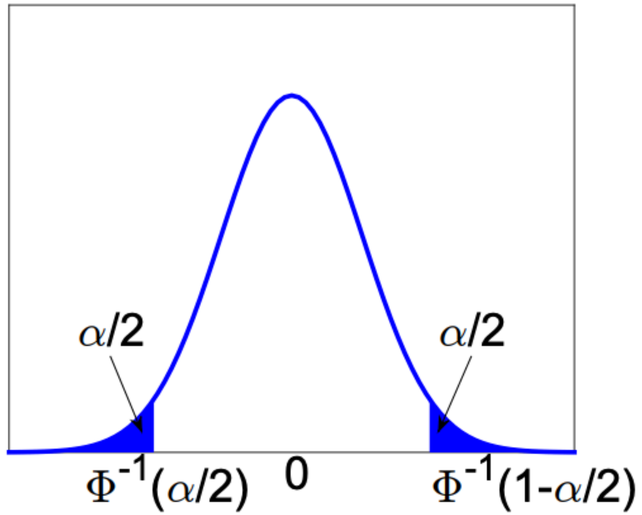
Continuous Random Variables



(a)

cumulative distribution function (cdf)

$$P(x) \triangleq \Pr(X \leq x)$$



(b)

probability density function (pdf)

$$p(x) \triangleq \frac{d}{dx} P(x)$$

$$\Pr(a < X \leq b) = \int_a^b p(x)dx = P(b) - P(a)$$

$$\Pr(x \leq X \leq x + dx) \approx p(x)dx$$

Sets of Related Random Variables

$p(X,Y)$	$Y=0$	$Y=1$
$X=0$	0.2	0.3
$X=1$	0.3	0.2

marginal distribution

$$p(X=x)=\sum_y p(X=x,Y=y)$$

conditional distribution

$$p(Y=y|X=x)=\frac{p(X=x,Y=y)}{p(X=x)}$$

$$p(x,y)=p(x)p(y|x)$$

chain rule

$$p(\boldsymbol{x}_{1:D})=p(x_1)p(x_2|x_1)p(x_3|x_1,x_2)p(x_4|x_1,x_2,x_3)\ldots p(x_D|\boldsymbol{x}_{1:D-1})$$

Bayes' rule

$$p(H = h|Y = y) = \frac{p(H = h)p(Y = y|H = h)}{p(Y = y)}$$

prior distribution — what we know about possible values of H before we see any data

$$p(H)$$

observation distribution — possible outcomes Y we expect to see if H =h

$$p(Y|H = h)$$

likelihood — evaluate the observation distribution at a point corresponding to the actual observations, y

$$p(Y = y|H = h)$$

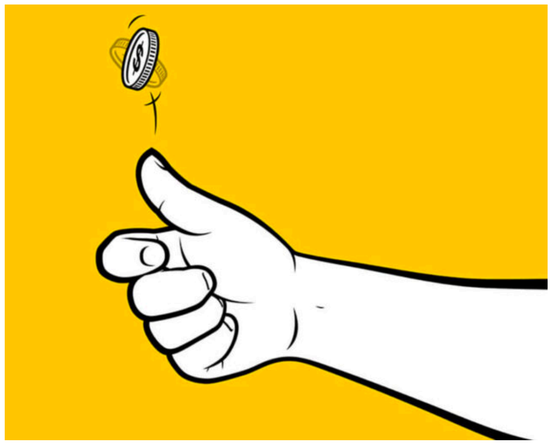
marginal likelihood

$$p(Y = y) = \sum_{h' \in \mathcal{H}} p(H = h')p(Y = y|H = h') = \sum_{h' \in \mathcal{H}} p(H = h', Y = y)$$

posterior — our new belief state about the possible value of H

$$p(H = h|Y = y)$$

A Simple Question?



What's the probability of head up? P(Heads)

0.8?

A Simple Question?

$$P(\text{Heads}) = \theta \qquad P(\text{Tails}) = 1 - \theta$$

$$D = \{\text{head}, \text{head}, \text{head}, \text{head}, \text{tail}\}$$

α_H total number of heads α_T total number of tails

$$P(D|\theta) = P(\alpha_H, \alpha_T \mid \theta) = \theta^{\alpha_H}(1-\theta)^{\alpha_T}$$

Maximum Likelihood Estimation

MLE: Choose θ that maximizes the probability of observed data

$$\hat{\theta} = \arg\max_{\theta} P(D \mid \theta)$$

$$= \arg\max_{\theta} \log P(D \mid \theta)$$

Set derivative to zero

$$\frac{\partial \log P(D \mid \theta)}{\partial \theta} = 0$$

Maximum Likelihood Estimation

Set derivative to zero

$$\frac{\partial \log P(D \mid \theta)}{\partial \theta} = 0$$

$$\frac{\partial \log \theta^{\alpha_H}(1-\theta)^{\alpha_T}}{\partial \theta} = 0$$

$$\hat{\theta}_{\text{MLE}} = \frac{\alpha_H}{\alpha_H + \alpha_T} = \frac{4}{4+1} = 0.8$$

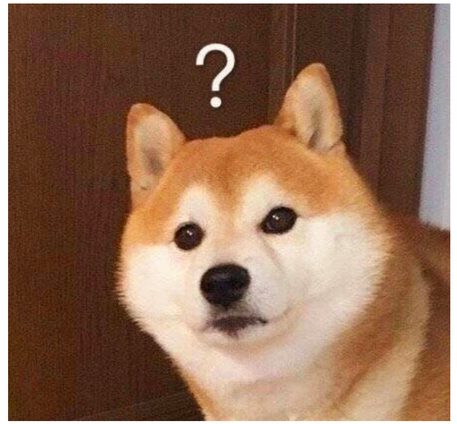
Maximum Likelihood Estimation

If you flip the coin 5 times, get 4 times head

$$\hat{\theta}_{\mathrm{MLE}} = \frac{\alpha_H}{\alpha_H + \alpha_T} = \frac{4}{4+1} = 0.8$$

If you flip the coin 5000 times, get 4000 times head

$$\hat{\theta}_{\mathrm{MLE}} = \frac{\alpha_H}{\alpha_H + \alpha_T} = \frac{4000}{4000+1000} = 0.8$$



Maximum Likelihood Estimation

MLE: Choose θ that maximizes the probability of observed data

$$\hat{\theta} = \arg\max_{\theta} P(D \mid \theta)$$

$$= \arg\max_{\theta} \log P(D \mid \theta)$$

Set derivative to zero

$$\frac{\partial \log P(D \mid \theta)}{\partial \theta} = 0$$

$$\hat{\theta}_{\text{MLE}} = \frac{\alpha_H}{\alpha_H + \alpha_T}$$

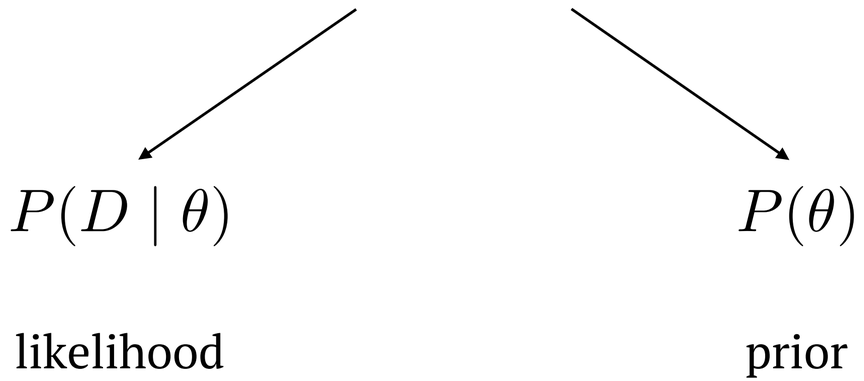
Bayesian Learning

$$P(\theta \mid D) = \frac{P(D \mid \theta)P(\theta)}{P(D)}$$

posterior

$$\arg\max_{\theta} P(\theta \mid D) = \arg\max_{\theta} \frac{P(D \mid \theta)P(\theta)}{P(D)}$$

$$= \arg\max_{\theta} P(D \mid \theta)P(\theta)$$



$P(D \mid \theta)$ likelihood

$P(\theta)$ prior

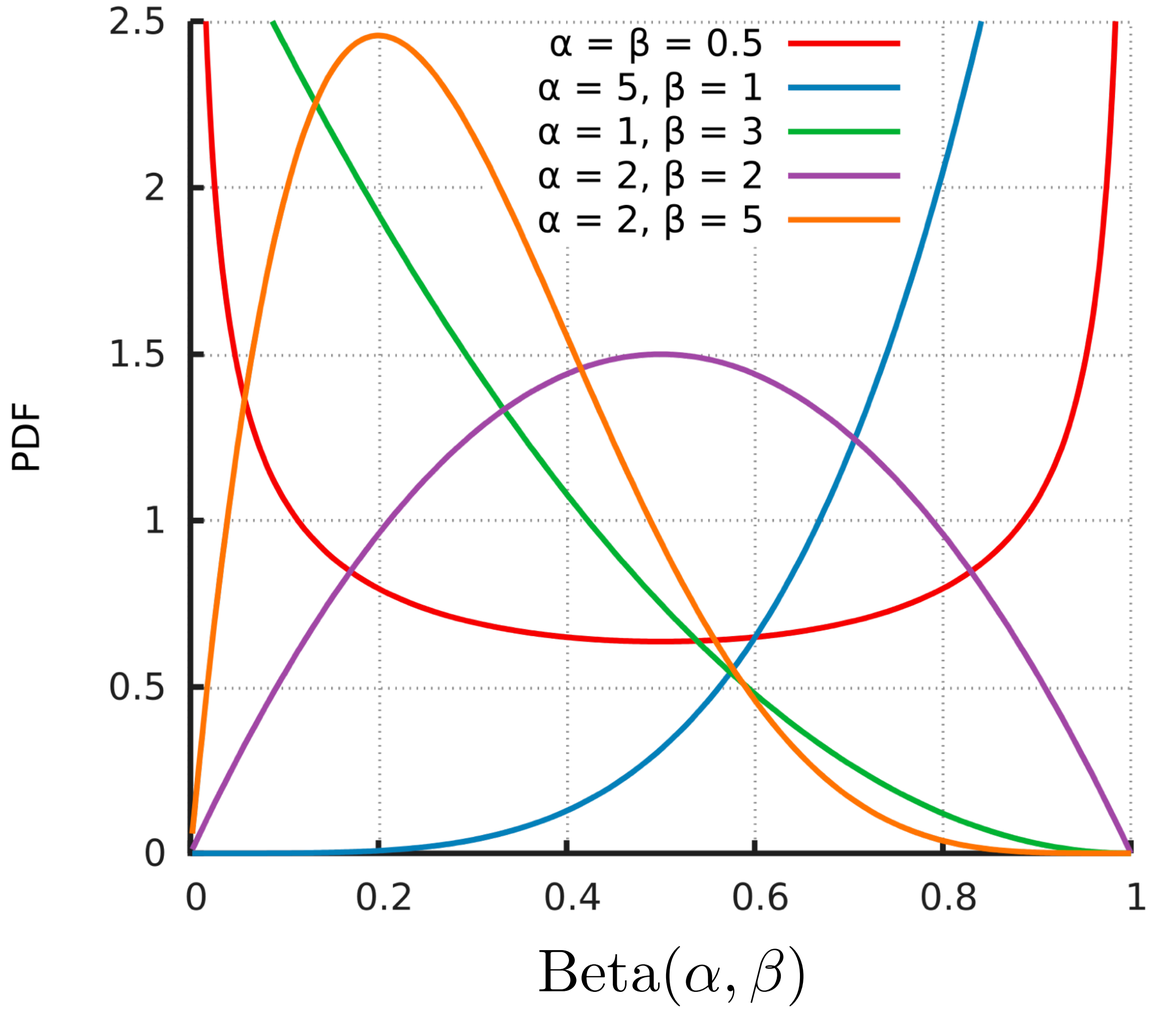
Bayesian Learning

prior distribution

$$P(\theta) \sim \text{Beta}(\beta_H, \beta_T)$$

Beta Distribution

$$P(\theta) = \frac{\theta^{\beta_H - 1}(1-\theta)^{\beta_T - 1}}{B(\beta_H, \beta T)}$$



Bayesian Learning

prior distribution

$$P(\theta) \sim \text{Beta}(\beta_H, \beta_T)$$

$$P(\theta) = \frac{\theta^{\beta_H - 1}(1-\theta)^{\beta_T - 1}}{B(\beta_H, \beta T)}$$

likelihood function

$$P(D|\theta) = P(\alpha_H, \alpha_T \mid \theta) = \theta^{\alpha_H}(1-\theta)^{\alpha_T}$$

posterior distribution

$$P(\theta \mid D) \propto \theta^{\alpha_H + \beta_H - 1}(1-\theta)^{\alpha_T + \beta_T - 1}$$

$$P(\theta \mid D) \sim \beta(\beta_H + \alpha_H, \beta_T + \alpha_T)$$

Estimating Parameters

MLE: Maximum Likelihood Estimate, choose θ that maximizes the probability of observed data

$$\hat{\theta} = \arg\max_{\theta} P(D \mid \theta)$$

MAP: Maximum a Posteriori, choose θ that is most probable given prior probability and the data

$$\hat{\theta} = \arg\max_{\theta} P(\theta \mid D) = \arg\max_{\theta} \frac{P(D \mid \theta)P(\theta)}{P(D)}$$

Recap: Supervised Learning in a Nutshell

$\boldsymbol{x} \in \mathcal{X}$

inputs/features/covariates/predictors

model

f_θ

training objective/loss

$\mathcal{D} = \{(\boldsymbol{x}_n, \boldsymbol{y}_n)\}_{n=1}^N$

training dataset

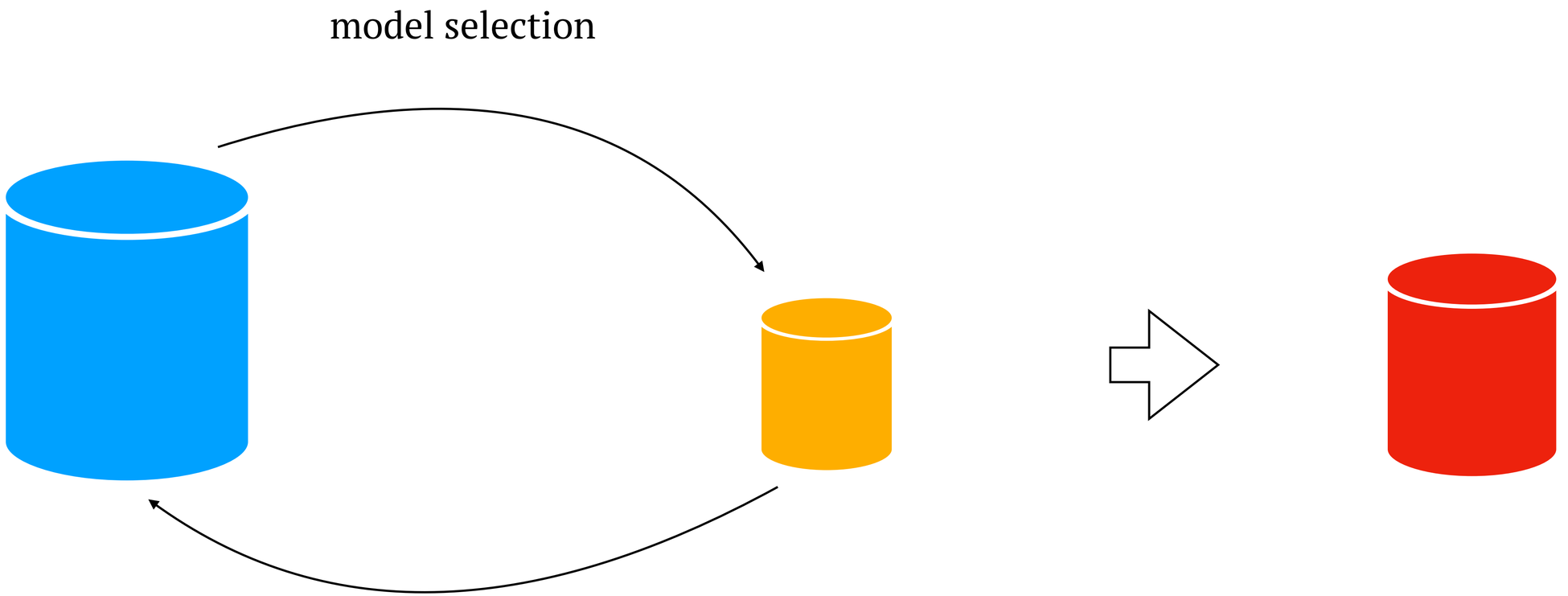
$\boldsymbol{y} \in \mathcal{Y}$

outputs/label/target/response

Case Study: Play Golf

	Outlook	Temperature	Humidity	Windy	Play Golf
0	Rainy	Hot	High	FALSE	No
1	Rainy	Hot	High	TRUE	No
2	Overcast	Hot	High	FALSE	Yes
3	Sunny	Mild	High	FALSE	Yes
4	Sunny	Cool	Normal	FALSE	Yes
5	Sunny	Cool	Normal	TRUE	No
6	Overcast	Cool	Normal	TRUE	Yes
7	Rainy	Mild	High	FALSE	No
8	Rainy	Cool	Normal	FALSE	Yes
9	Sunny	Mild	Normal	FALSE	Yes
10	Rainy	Mild	Normal	TRUE	Yes
11	Overcast	Mild	High	TRUE	Yes
12	Overcast	Hot	Normal	FALSE	Yes
13	Sunny	Mild	High	TRUE	No

Train / Dev / Test Set



$\mathcal{D}_{\text{train}} = \{(\boldsymbol{x}^{(n)}, \boldsymbol{y}^{(n)})\}_{n=1}^{N_{\text{train}}}$

$\mathcal{D}_{\text{dev}} = \{(\boldsymbol{x}^{(n)}, \boldsymbol{y}^{(n)})\}_{n=1}^{N_{\text{dev}}}$

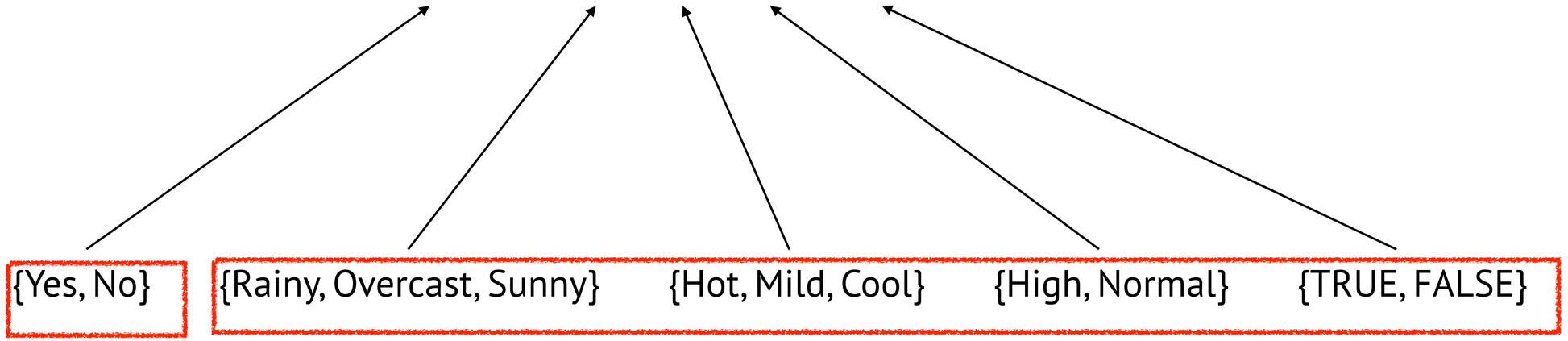
$\mathcal{D}_{\text{test}} = \{(\boldsymbol{x}^{(n)}, \boldsymbol{y}^{(n)})\}_{n=1}^{N_{\text{test}}}$

Case Study: Play Golf

	x_1	x_2	x_3	x_4	y
	Outlook	Temperature	Humidity	Windy	Play Golf
0	Rainy	Hot	High	FALSE	No
1	Rainy	Hot	High	TRUE	No
2	Overcast	Hot	High	FALSE	Yes
3	Sunny	Mild	High	FALSE	Yes
4	Sunny	Cool	Normal	FALSE	Yes
5	Sunny	Cool	Normal	TRUE	No
6	Overcast	Cool	Normal	TRUE	Yes
7	Rainy	Mild	High	FALSE	No
8	Rainy	Cool	Normal	FALSE	Yes
9	Sunny	Mild	Normal	FALSE	Yes
10	Rainy	Mild	Normal	TRUE	Yes
11	Overcast	Mild	High	TRUE	Yes
12	Overcast	Hot	Normal	FALSE	Yes
13	Sunny	Mild	High	TRUE	No

Case Study: Play Golf

$$y \quad = f(\ x_1\ ,\ x_2\ \ ,x_3\ \ ,\ x_4\ )$$

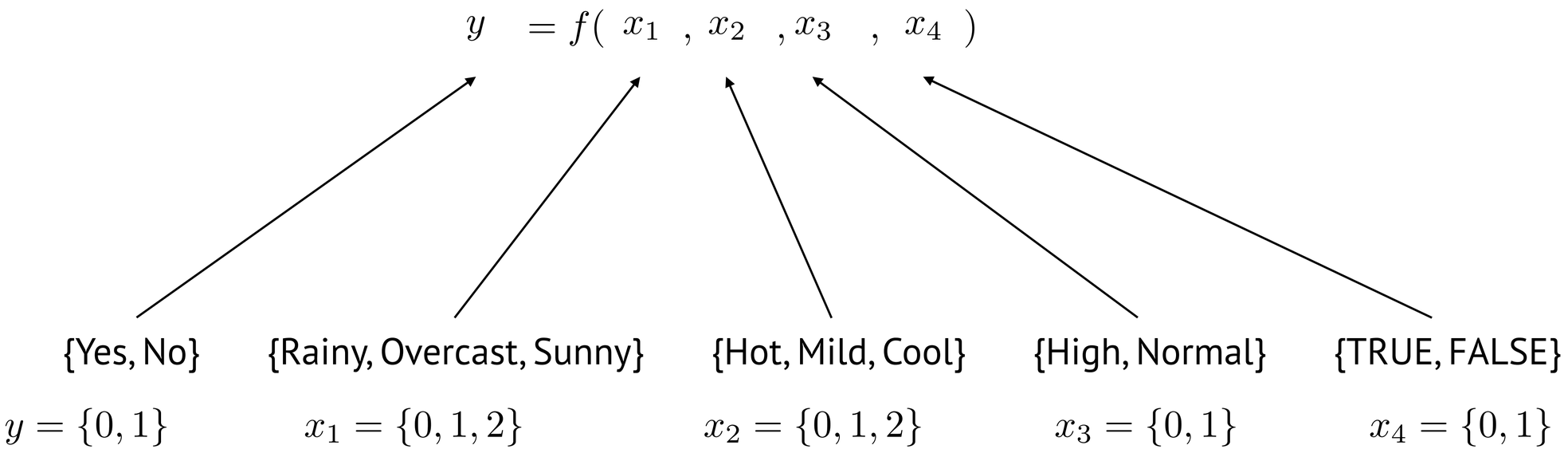


Binary Classification Problem

Discrete Features

What's in our toolbox?

Case Study: Play Golf



	Outlook	Temperature	Humidity	Windy	Play Golf
0	Rainy	Hot	High	FALSE	No

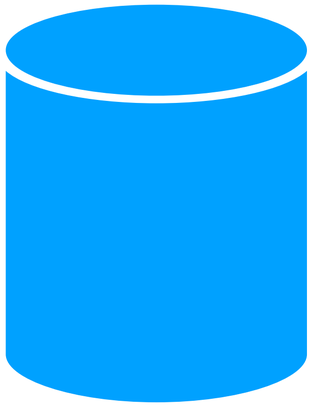
$$(\boldsymbol{x}^{(i)} = (0, 0, 0, 1), \boldsymbol{y}^{(i)} = 1)$$

Perceptron

$$f(\boldsymbol{x}; \boldsymbol{\theta}) = \mathbb{I}(\boldsymbol{w}^\top \boldsymbol{x} + b > 0)$$

Inference / Prediction: $(\boldsymbol{x}^{(i)} = (0, 0, 0, 1), \boldsymbol{y}^{(i)} = 1)$ $\boldsymbol{\theta} = \{\boldsymbol{w} = (3, -4, 5, -2), b = 1\}$

Learning / Training:



Update Rules

Objective / Loss

$\hat{\boldsymbol{\theta}}$

$$\mathcal{D}_{\text{train}} = \{(\boldsymbol{x}^{(n)}, \boldsymbol{y}^{(n)})\}_{n=1}^{N_{\text{train}}}$$

Perceptron

$$f(\boldsymbol{x};\boldsymbol{\theta}) = \mathbb{I}(\boldsymbol{w}^\top \boldsymbol{x} + b > 0)$$

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \eta_t(\hat{y} - y)\boldsymbol{x}$$

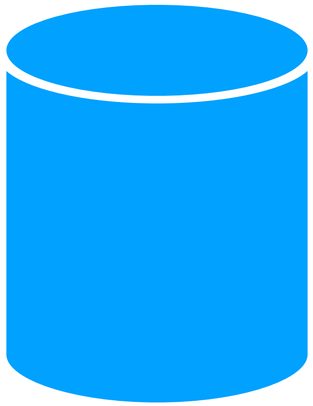
$(\boldsymbol{x}^{(i)} = (0,0,0,1), \boldsymbol{y}^{(i)} = 1)$ $\boldsymbol{\theta} = \{\boldsymbol{w} = (3,-4,5,-2), b = 1\}$ $\eta_t = 1$

$$\hat{y} =$$

Supervised Learning

Model / Function $\boldsymbol{y} = f(\boldsymbol{x}; \boldsymbol{\theta})$

Learning / Training:

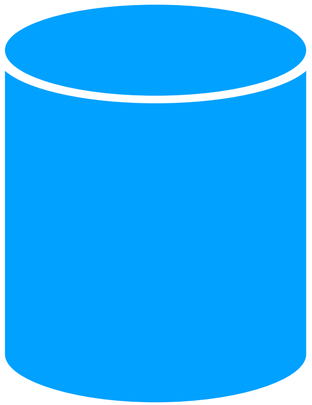


Objective / Loss

$\hat{\boldsymbol{\theta}}$

$\mathcal{D}_{\text{train}} = \{(\boldsymbol{x}^{(n)}, \boldsymbol{y}^{(n)})\}_{n=1}^{N_{\text{train}}}$

Probabilistic Models



$$\mathcal{D}_{\text{train}} = \{(\boldsymbol{x}^{(n)}, \boldsymbol{y}^{(n)})\}_{n=1}^{N_{\text{train}}}$$

How does this happen?

They all sampled from a distribution!

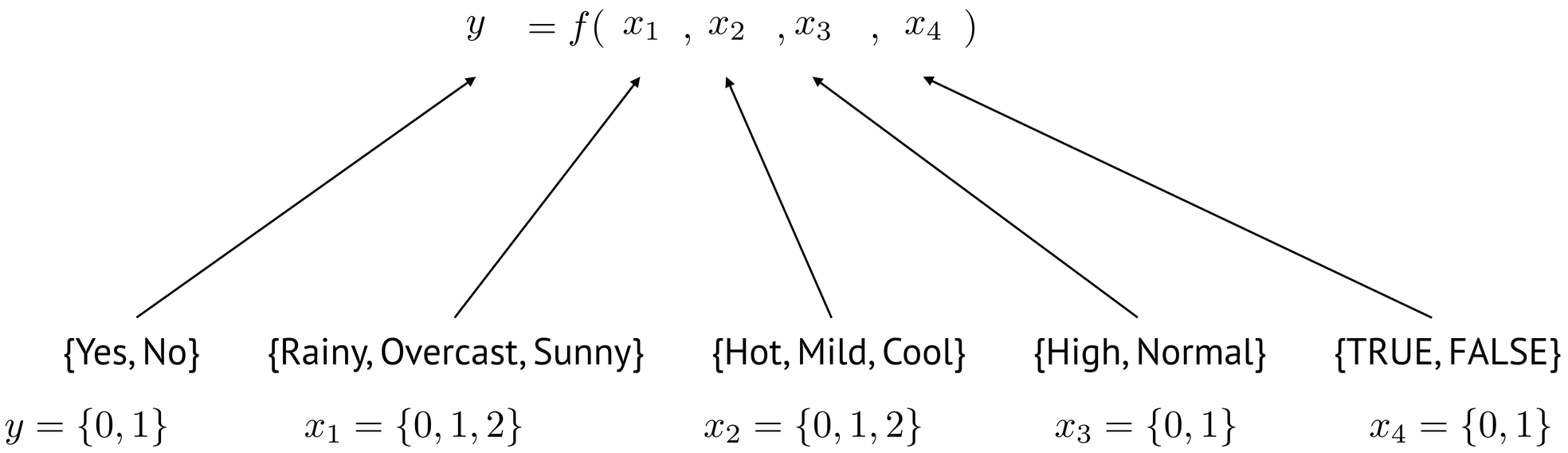
$p(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{\theta})$ Model / Function

$$p(\mathcal{D}_{\text{train}} \mid \boldsymbol{\theta}) = \prod_{n=1}^{N_{\text{train}}} p(\boldsymbol{y}^{(n)} \mid \boldsymbol{x}^{(n)}, \boldsymbol{\theta})$$

Objective / Loss

Learning / Training $\longrightarrow$ $\hat{\boldsymbol{\theta}}$

Case Study: Play Golf



	Outlook	Temperature	Humidity	Windy	Play Golf
0	Rainy	Hot	High	FALSE	No

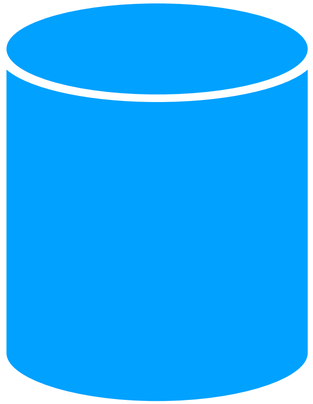
$$(\boldsymbol{x}^{(i)} = (0, 0, 0, 1), \boldsymbol{y}^{(i)} = 1)$$

Logistic Regression

$$p(y \mid \boldsymbol{x}, \boldsymbol{\theta}) = \mathrm{Ber}(y|\sigma(\boldsymbol{w}^\top \boldsymbol{x} + b))$$

Inference / Prediction: $(\boldsymbol{x}^{(i)} = (0, 0, 0, 1), \boldsymbol{y}^{(i)} = 1)$ $\boldsymbol{\theta} = \{\boldsymbol{w} = (3, -4, 5, -2), b = 1\}$

Learning / Training:



$\mathcal{D}_{\text{train}} = \{(\boldsymbol{x}^{(n)}, \boldsymbol{y}^{(n)})\}_{n=1}^{N_{\text{train}}}$

$\longrightarrow \hat{\boldsymbol{\theta}}$

Objective / Loss:

$$p(\mathcal{D} \mid \boldsymbol{\theta}) \longrightarrow \arg\min_{\boldsymbol{\theta}} [-\log P(\mathcal{D} \mid \boldsymbol{\theta})]$$

$$-\sum_{n=1}^{N_{\text{train}}} [(y^{(n)}) \log p(y = 1 \mid \boldsymbol{x}^{(n)}, \boldsymbol{\theta}) + (1 - y^{(n)}) \log p(y = 0 \mid \boldsymbol{x}^{(n)}, \boldsymbol{\theta})]$$

Update rule:

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \eta_t(\sigma(\boldsymbol{w}_t \boldsymbol{x} + b) - y)\boldsymbol{x}$$

Probabilistic Language Models

Assign a probability to a sentence

P(“I am going to school”) > P(“I are going to school”) — Grammar Checking

I had some coffee this morning.

P(“我今早喝了一些咖啡”) > P(“我今早吃了一些咖啡”) — Machine translation

P(“Can we put an elephant into the refrigerator? No, we can’t.”) > P(“Can we put an elephant into the refrigerator? Yes, we can.”) — Question Answering

Probabilistic Language Models

$$\mathcal{V} = \{\text{the}, \text{dog}, \text{laughs}, \text{saw}, \text{barks}, \text{cat}, \ldots\}$$

A sentence in the language is a sequence of words

$$x_1 x_2 \ldots x_n$$

For example

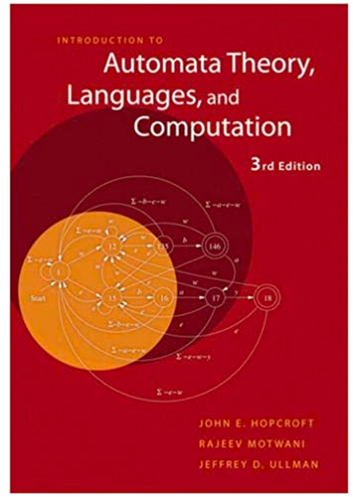
the dog barks STOP

the cat saw the dog STOP

...

Definition (Language Mode)

1. For any $\langle x_1 \ldots x_n \rangle \in \mathcal{V}^\dagger$, $p(x_1, x_2, \ldots x_n) \geq 0$
2. In addition, $\sum_{\langle x_1 \ldots x_n \rangle \in \mathcal{V}^\dagger} p(x_1, x_2, \ldots x_n) = 1$



(Hopcroft, Motwani, Ullman)

A (very bad) method for learning a LM

Number of times the sentence $x_1 \dots x_n$ is seen in the training corpus

$$c(x_1 \dots x_n)$$

Total number of sentences in the training corpus N

$$p(x_1 \dots x_n) = \frac{c(x_1 \dots x_n)}{N}$$

Why this is very bad?

Markov Models

Consider a sequence of random variables $X_1, X_2, \ldots, X_n$, each take any value in $\mathcal{V}$

The joint probability of a sentence is

$$P(X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n)$$

$$= P(X_1 = x_1) \prod_{i=2}^{n} P(X_i = x_i | X_1 = x_1, \ldots, X_{i-1} = x_{i-1})$$

$$= P(X_1 = x_1) \prod_{i=2}^{n} P(X_i = x_i | X_{i-1} = x_{i-1})$$

First-order Markov Assumption

Trigram Language Models

A trigram language model consists of a finite set $\mathcal{V}$, and a parameter $q(w \mid u, v)$

For each trigram u, v, w , such that $w \in \mathcal{V} \cup \{\text{STOP}\}$, $u, v \in \mathcal{V} \cup \{*\}$.

$q(w \mid u, v)$ can be interpreted as the probability of seeing the word w immediately after the bigram (u, v) .

For any sentence $x_1 \dots x_n$, where $x_i \in \mathcal{V}$ for $i = 1 \dots (n-1)$, and $x_n = \text{STOP}$

$$p(x_1 \dots x_n) = \prod_{i=1}^{n} q(x_i \mid x_{i-2}, x_{i-1})$$

where we define $x_0 = x_{-1} = *$

Trigram Language Models

For example, for the sentence

the dog barks STOP

$$p(\text{the dog barks STOP}) = q(\text{the} \mid *, *) \times q(\text{dog} \mid *, \text{the}) \times q(\text{barks} \mid \text{the}, \text{dog}) \times q(\text{STOP} \mid \text{dog}, \text{barks})$$

Problem solved? How can we find $q(w \mid u, v)$

Parameters (of the model)

$q(w \mid u, v)$

How many parameters?

How to “estimate” them from training data?

Trigram Language Models

Parameters (of the model)

$$q(w \mid u, v)$$

How many parameters?

$$|\mathcal{V}|^3$$

How to "estimate" them from training data?

$$q(w \mid u, v) = \frac{c(u, v, w)}{c(u, v)}$$

$$q(\text{barks} \mid \text{the}, \text{dog}) = \frac{c(\text{the}, \text{dog}, \text{barks})}{c(\text{the}, \text{dog})}$$

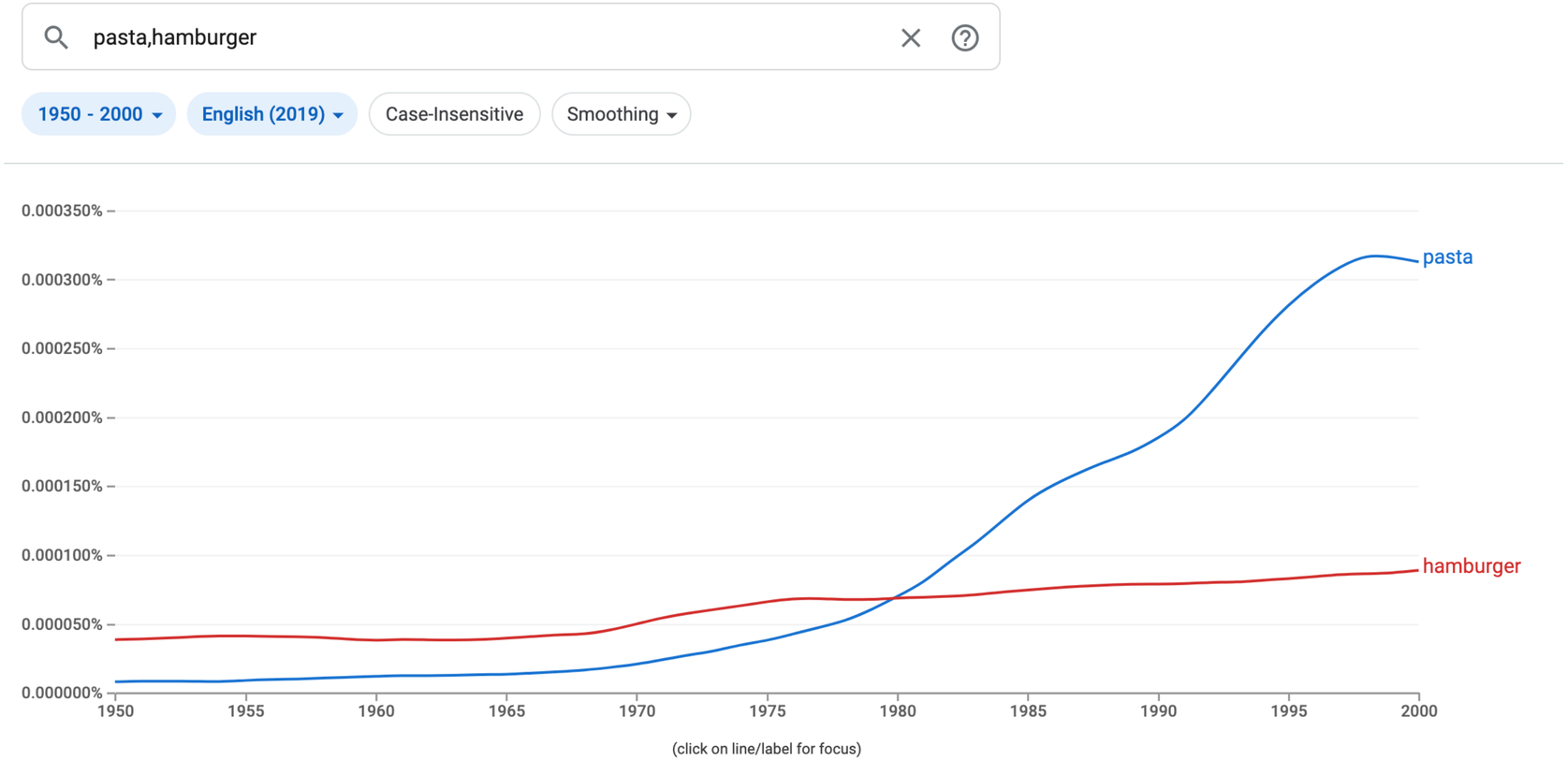
Trigram Language Models

How to “estimate” them from training data?

$$q(w \mid u, v) = \frac{c(u, v, w)}{c(u, v)}$$

$$q(\text{barks} \mid \text{the}, \text{dog}) = \frac{c(\text{the}, \text{dog}, \text{barks})}{c(\text{the}, \text{dog})}$$

N-gram counts!



Pasta v.s. Hamburger (Google Books Ngram Viewer)

Sparse Data Problems

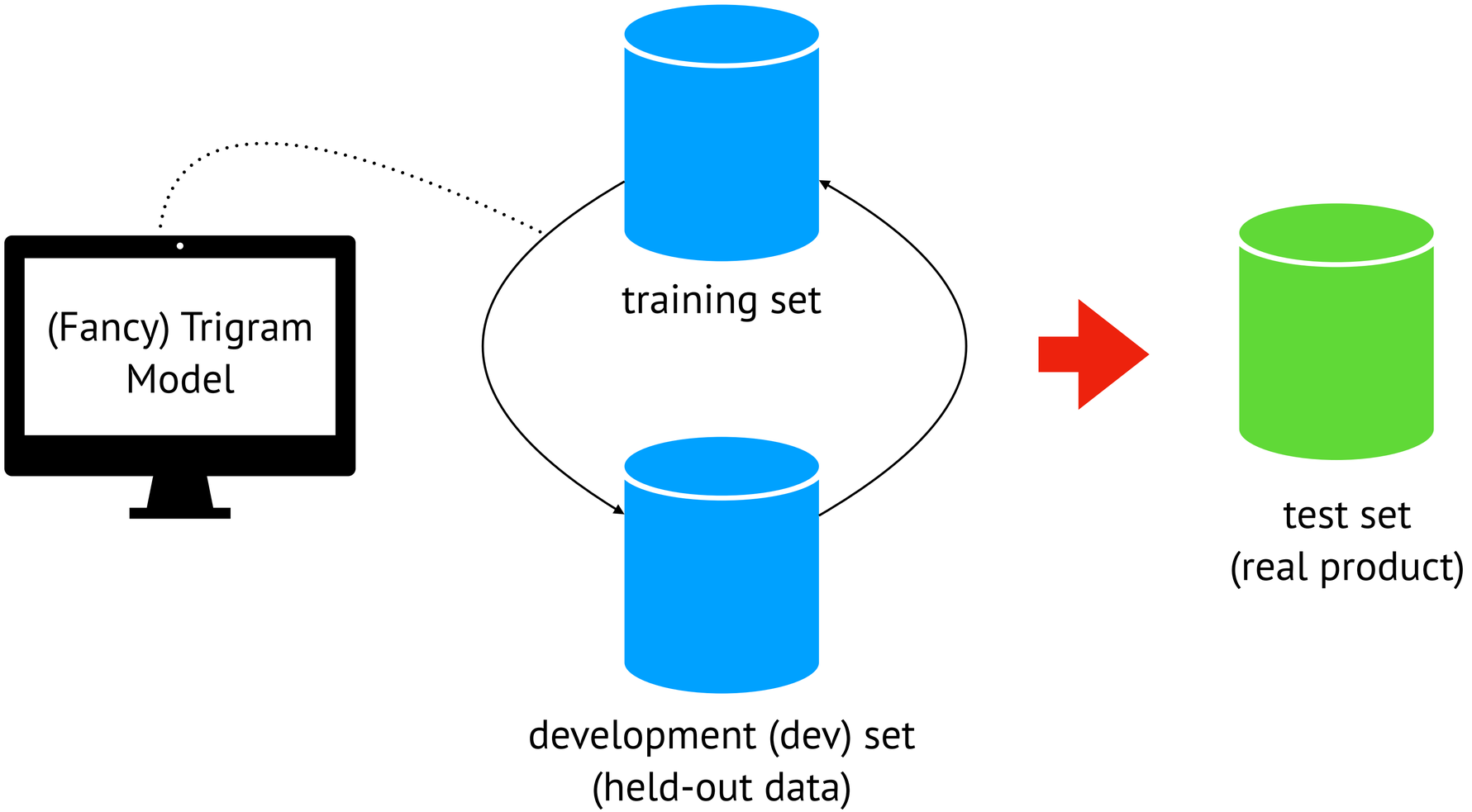
Maximum likelihood estimate:

$$q(w \mid u, v) = \frac{c(u, v, w)}{c(u, v)}$$

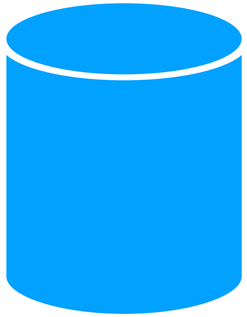
$$q(\text{barks} \mid \text{the}, \text{dog}) = \frac{c(\text{the}, \text{dog}, \text{barks})}{c(\text{the}, \text{dog})}$$

$|\mathcal{V}|^3$ Say vocabulary size is 20000. We have 8 *10^{12} parameters!!

Evaluating Language Models: Perplexity



Evaluating Language Models: Perplexity



development (dev) set
(held-out data)

. . .

$x^{(i)}$ the cat laughs STOP

$x^{(i+1)}$ the dog laughs at the cat STOP

. . .

We can compute the probability it assigns to the entire set of test sentences

$$\prod_{i=1}^{m} p(x^{(i)})$$

The higher this quantity is, the better the language model is at modeling unseen sentences.

Evaluating Language Models: Perplexity

The higher this quantity is, the better the language model is at modeling unseen sentences.

$$\prod_{i=1}^{m} p(x^{(i)})$$

Perplexity on the test corpus is derived as a direction transformation of this.

$$\text{ppl} = 2^{-l}$$

$$l = \frac{1}{M} \sum_{i=1}^{m} \log_2 p(x^{(i)})$$

M is the total length of the sentences in the test corpus.

What if the model estimate $q(w \mid u, v) = 0$ and the trigram appears in the dataset?

Wait, why we love this number in the first place?

Let the model predicts $q(w \mid u, v) = 1/N$

$$l = \frac{1}{M} \sum_{i=1}^{m} \log_2 p(x^{(i)})$$

$$\text{ppl} = 2^{-l} = N$$

A uniform probability model — The perplexity is equal to the vocabulary size!

Perplexity can be thought of as the effective vocabulary size under the model! For example, the perplexity of the model is 120 (even though the vocabulary size is say 10,000), then this is roughly equivalent to having an effective vocabulary of 120.

How much your language model updates the uniform guess!

Bayes Factors:
https://www.youtube.com/watch?v=lG4VkPoG3ko

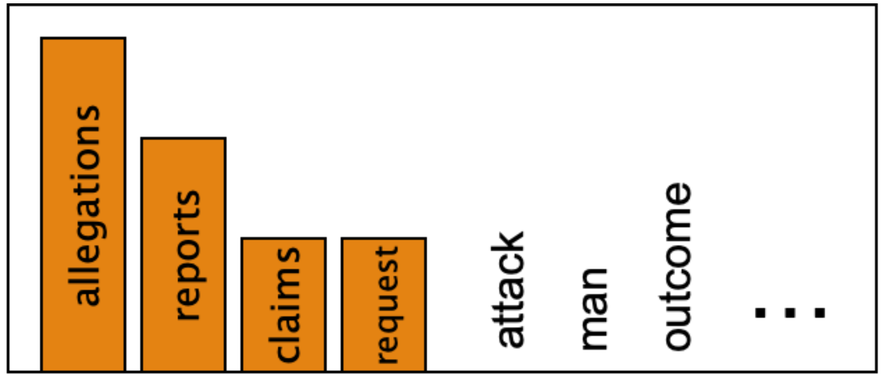
Smoothing for Language Models

If the model estimate $q(w \mid u, v) = 0$ and the trigram appears in the test data, ppl goes up to infinity.

When we have sparse statistics:

P(w | denied the)
- 3 allegations
- 2 reports
- 1 claims
- 1 request

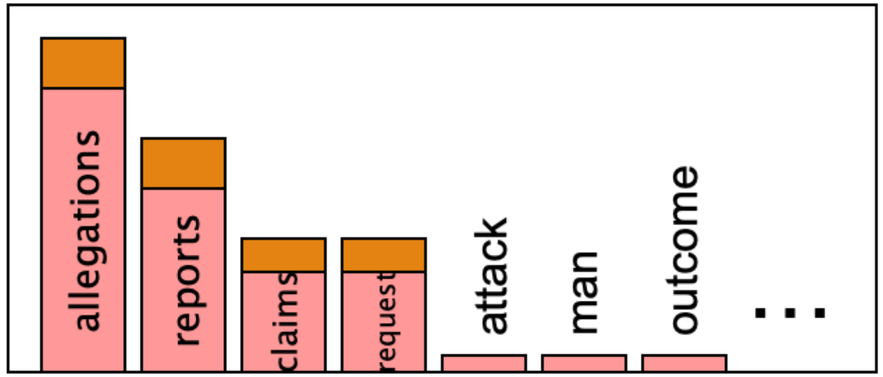
7 total



Steal probability mass to generalize better:

P(w | denied the)
- 2.5 allegations
- 1.5 reports
- 0.5 claims
- 0.5 request
- 2 other

7 total



Example from Dan Klein

Add-one (Laplace) smoothing

Considering a bigram model here, pretend we saw each word one more time than we did.

MLE estimate:

$$q_{\text{MLE}}(w_i|w_{i-1}) = \frac{c(w_{i-1}, w_i)}{c(w_{i-1})}$$

Add-one smoothing:

$$q_{\text{Laplace}}(w_i|w_{i-1}) = \frac{c(w_{i-1}, w_i) + 1}{c(w_{i-1}) + |\mathcal{V}|}$$

Linear Interpolation (Stupid Backoff)

Trigram Model, Bigram Model, Unigram Model

Trigram maximum-likelihood estimate: $q(w_i \mid w_{i-2}, w_{i-1}) = \frac{c(w_{i-2}, w_{i-1}, w_i)}{c(w_{i-2}, w_{i-1})}$

Bigram maximum-likelihood estimate: $q(w_i \mid w_{i-1}) = \frac{c(w_{i-1}, w_i)}{c(w_{i-1})}$

Unigram maximum-likelihood estimate: $q(w_i) = \frac{c(w_i)}{c(\cdot)}$

Which one suffers from the data sparsity problem the most?
Which one is more accurate?

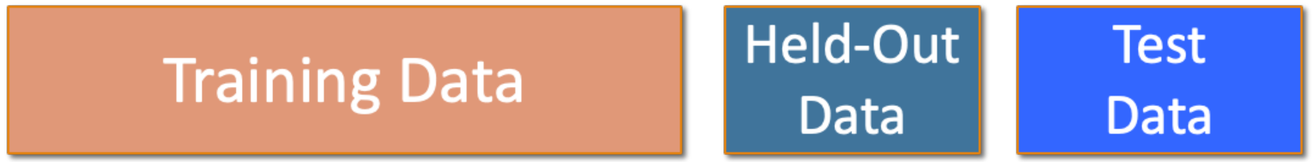
Linear Interpolation (Stupid Backoff)

$$
\begin{aligned}
q(w_i \mid w_{i-2}, w_{i-1}) = \quad & \lambda_1 \times q_{\mathrm{ML}}(w_i \mid w_{i-2}, w_{i-1}) \\
& + \lambda_2 \times q_{\mathrm{ML}}(w_i \mid w_{i-1}) \\
& + \lambda_3 \times q_{\mathrm{ML}}(w_i)
\end{aligned}
$$

$$\text{where } \lambda_1 + \lambda_2 + \lambda_3 = 1, \text{ and } \lambda_i \geq 0 \text{ for all } i.$$

How to choose the value of $\lambda_1, \lambda_2, \lambda_3$

Use the held-out corpus Hyperparameters



maximize the probability of held-out data.

Markov Models in Retrospect

Consider a sequence of random variables $X_1, X_2, \ldots, X_n$, each take any value in $\mathcal{V}$

The joint probability of a sentence is

$$P(X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n)$$

$$= P(X_1 = x_1) \prod_{i=2}^{n} P(X_i = x_i | X_1 = x_1, \ldots, X_{i-1} = x_{i-1})$$

$$= P(X_1 = x_1) \prod_{i=2}^{n} P(X_i = x_i | X_{i-1} = x_{i-1})$$

First-order Markov Assumption

$$P(X_i = x_i | X_1 = x_1, \ldots, X_{i-1} = x_{i-1})$$

Is it possible to directly model this probability?