NLP 201: Probability and Language Modeling

Jeffrey Flanigan

University of California Santa Cruz

jmflanig@ucsc.edu

Fall 2023

Outline

- Probability Review
- Naive Bayes Classifier (Review)
- Language Modeling

Probability Review

For more, see readings on Canvas

Probability concepts

- Event space
- Random variables
- Probability distributions
- Parameters

Event space Ω

The space of events or possible outcomes (a set).

Example:

- Experiment which consists of 2 coin tosses
- $\Omega = \{(h, h), (h, t), (t, h), (t, t)\}$

Random variable (R.V.)

A function of elements of the event space.

Usually written in capital letters like X, Y, or W_i (a collection of R.V.s indexed by i)

Example:

- Experiment which consists of 2 coin tosses
- The number of heads, let's denote H (real-valued R.V.)
- The number of tails, let's denote $T = 2 - H$ (real-valued R.V.)
- Result of the ith toss: $X_i \in \{h, t\}$ (categorical R.V.)

Another example: Sentences

Experiment: Sample a sentence uniformly from a given corpus.

- Let S denote the sentence.
- Let W_i denote the i word in the sentence.
- Sometimes we will write $S = (W_1 \dots W_n)$, where n is the length of the sentence.
- Note: n is a R.V. even though it's lowercase.

Probability distribution

A **probability distribution**:

- takes either a set (a subset of the sample space, such as $\{(h,h),(h,t)\} \subset \Omega$) or a condition (such as $H = 2$)
- returns a value ≥ 0
- the sum over all disjoint possibilities must sum (or integrate if continuous) to 1
- if the sets or conditions are disjoint, then the probabilities add

Examples: 2 fair coin tosses

- $Pr(\{(h,h),(h,t)\}) = .5$
- $Pr(X_1 = h) = .5$
- $Pr(X_1 = h, X_2 = t) = .25$ *Note: comma denotes "and"*

Other notations

- Sometimes will write $Pr(X = h)$ or $p(X = h)$ or $P(X = h)$
- Sometimes leave off value of the condition $Pr(X)$. You can think of this as a function you haven't given the value yet
- Sometimes leave off the R.V. and only write the value $Pr(x)$

Parameters of a Distribution

We often **parameterize** a distribution with a number, collection of numbers, or a vector.
Example: coin toss of a single weighted coin.

- $Pr(X = h) = \theta$, where θ is a parameter
- Then $Pr(X = t) = 1 - \theta$
- This is called a **Bernoulli distribution**
- X is called a **Bernoulli R.V.**

Notation:

- Sometimes write the parameters: $Pr(X|\theta)$ or $Pr(X; \theta)$, sometimes omit parameters: $Pr(X)$.

Recap and Example

Random variable

- A variable that can take values within a fixed set (discrete) or within some range (continuous).

$$X \in \{1, 2, 3, 4, 5, 6\}$$

$$X \in \{the, a, dog, cat, runs, to, store\}$$

$P(X = x)$

Probability that the random variable X takes the value x (e.g., 1)

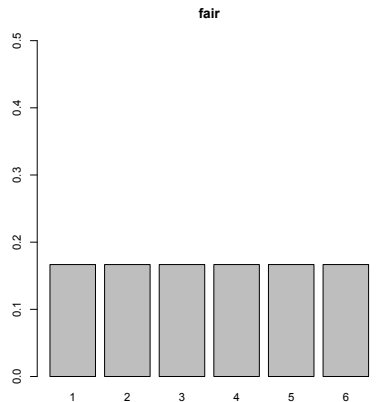
$$X \in \{1, 2, 3, 4, 5, 6\}$$

Two conditions:

1. Between 0 and 1: $0 \leq P(X = x) \leq 1$
2. Sum of all probabilities = 1 $\sum_{x} P(X = x) = 1$

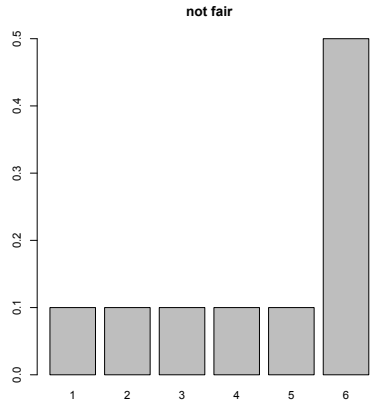
Fair dice

$X \in \{1, 2, 3, 4, 5, 6\}$



Weighted dice

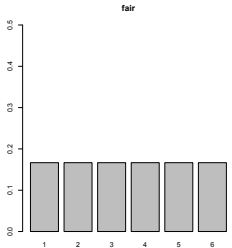
$X \in \{1, 2, 3, 4, 5, 6\}$



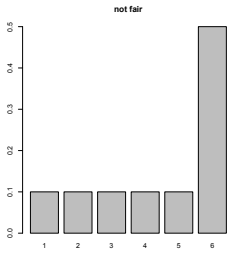
Inference

$$X \in \{1, 2, 3, 4, 5, 6\}$$

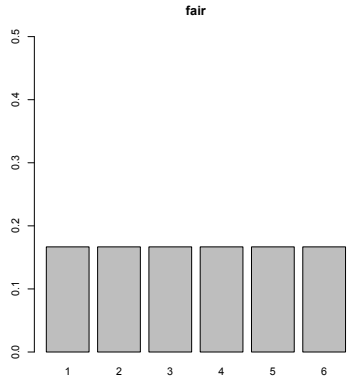
We want to *infer* the probability distribution that generated the data we see.

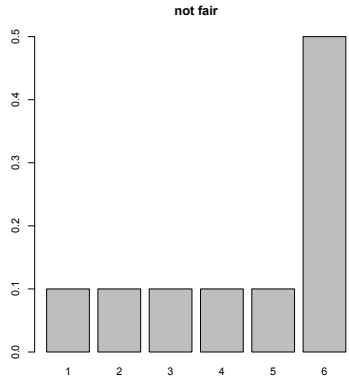


?

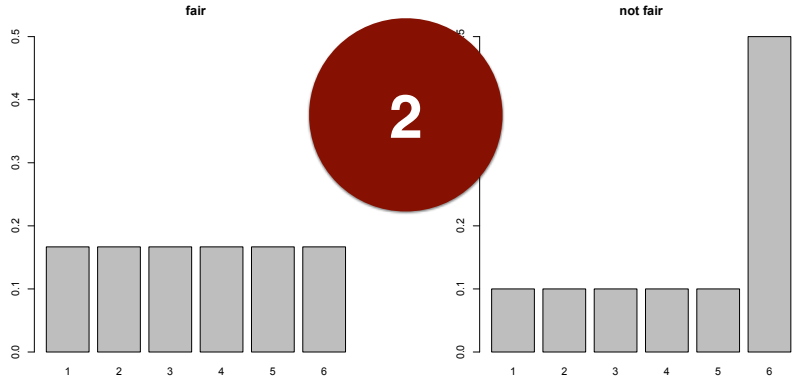


Probability



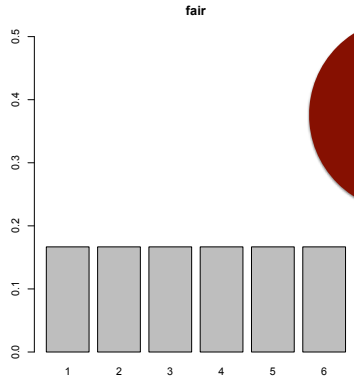


Probability

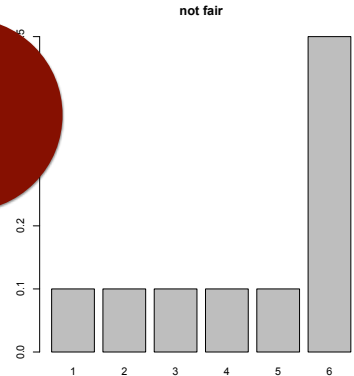


Probability

2

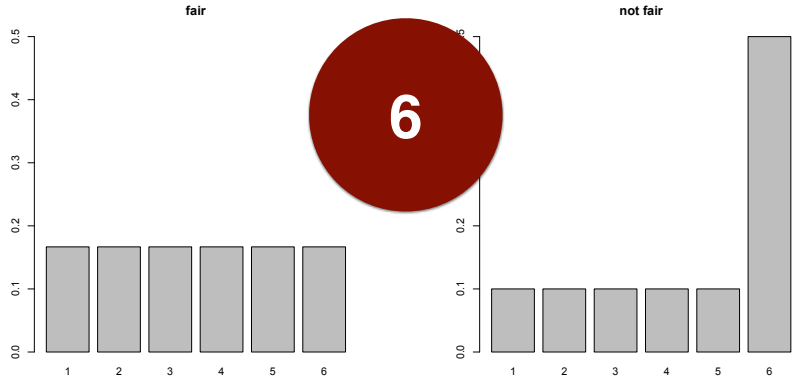


6

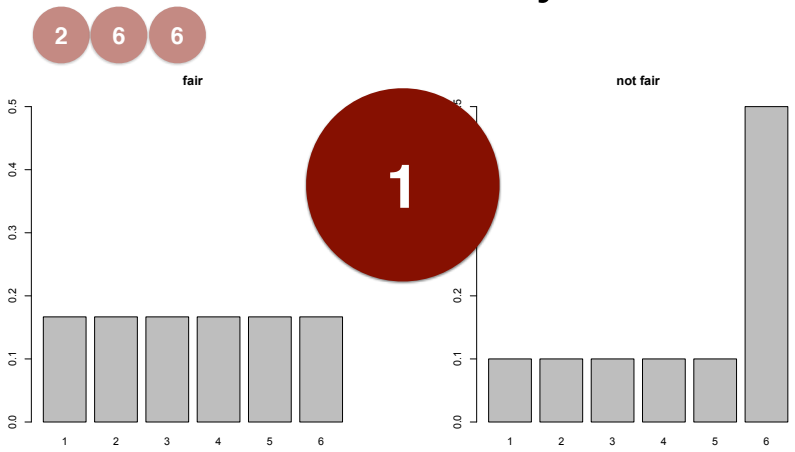


Probability

2 6

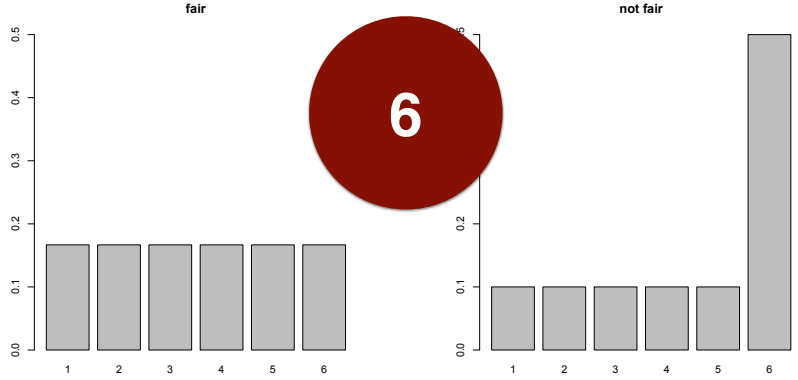


Probability



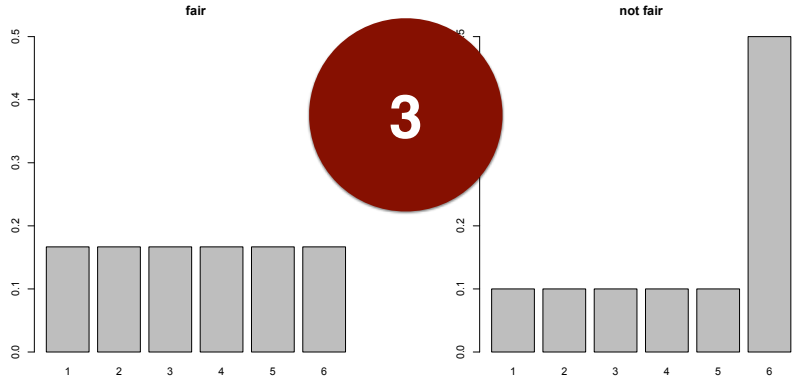
Probability

2 6 6 1



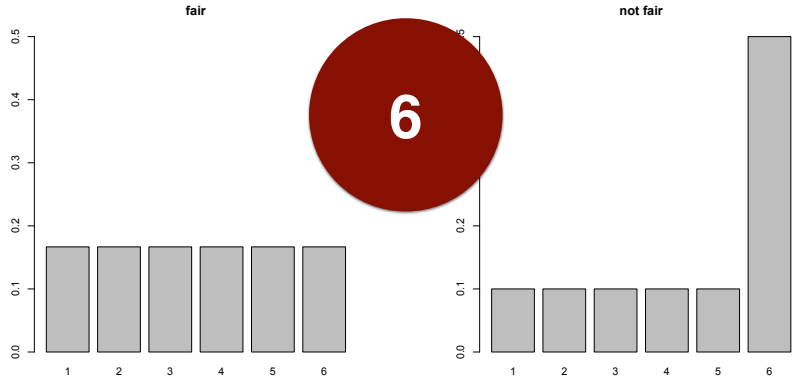
Probability

2 6 6 1 6



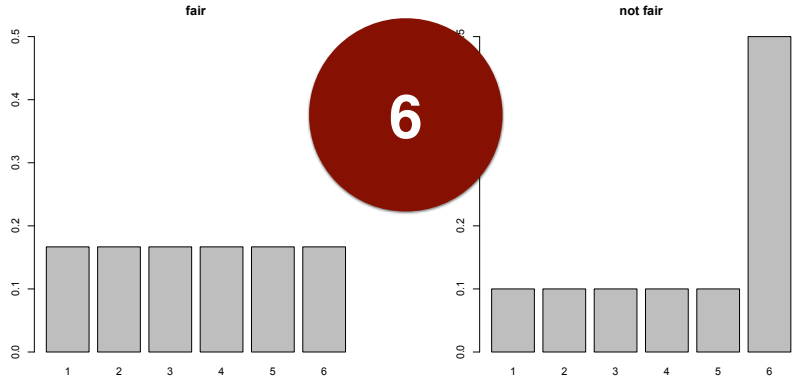
Probability

2 6 6 1 6 3



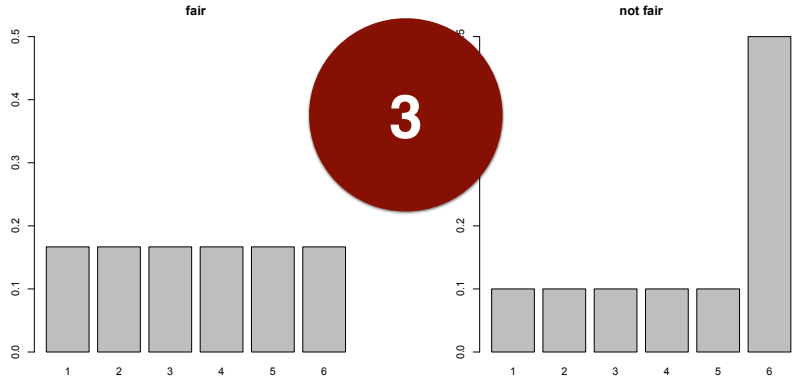
Probability

2 6 6 1 6 3 6



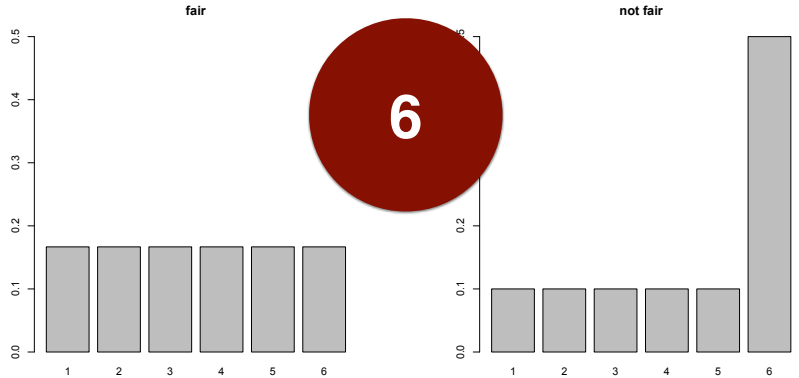
Probability

2 6 6 1 6 3 6 6



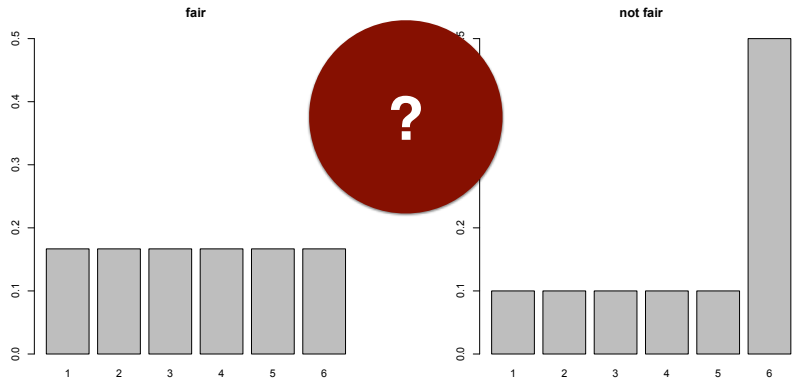
Probability

2 6 6 1 6 3 6 6 3



Probability

2 6 6 1 6 3 6 6 3 6



Board Work

We roll a dice 10 times. Let X_i be the result of the ith roll.
$X_1 = 2, X_2 = 6, X_3 = 6, X_4 = 1, X_5 = 6, X_6 = 3, X_7 = 6, X_8 = 6, X_9 = 3, X_{10} = 6$

- (3 Minutes) On your whiteboard or in the chat, compute the parameter vector θ:
  - Let θ_k be the probability of rolling k. Compute θ_k for $k = 1 \dots 6$.
- (3 minutes) Discuss with the person next to you or with the others in Zoom

Board work (more difficult): MLE for binomial distribution

We flip a weighted coin n times.
$X_i = 1$ denotes heads, $X_i = 0$ denotes tails.
$Pr(X_i = 1) = \theta$ θ **is a parameter**
$Pr(X_i = 0) = 1 - \theta$
(5 minutes: on your board)

- Write an expression for the probability of getting k heads.
- Write an expression for maximum likelihood estimate for θ after observing k heads. How would you solve this equation?

$$\hat{\theta} = \underset{\theta}{\operatorname{argmax}}\, Pr(\sum_{i=1}^{n} X_i = k; \theta) = ?$$

(3 minutes: discuss)

Very Quick Review of Probability

- Event space (e.g., $\mathcal{X}$, $\mathcal{Y}$)—in this class, usually discrete
- Random variables (e.g., X, Y)
- Typical statement: "random variable X takes value $x \in \mathcal{X}$ with probability $p(X = x)$, or, in shorthand, $p(x)$"
- Joint probability: $p(X = x, Y = y)$
- Marginal distribution (or just **marginal**): $p(X = x) = \sum_y p(X = x, Y = y)$ and $p(Y = y) = \sum_x p(X = x, Y = y)$
- The difference between *true* and *estimated* probability distributions

Independence

- Two random variables are independent if:

$$P(A, B) = P(A) \times P(B)$$

- In general:

$$P(x_1, \ldots, x_n) = \prod_{i=1}^{N} P(x_i)$$

- Information about one random variable (B) gives no information about the value of another (A)

$$P(A) = P(A \mid B) \qquad P(B) = P(B \mid A)$$

Very Quick Review of Probability

- Conditional probability: $p(X = x \mid Y = y) = \dfrac{p(X = x, Y = y)}{p(Y = y)}$
- Always true:
  $p(X = x, Y = y) = p(X = x \mid Y = y) \cdot p(Y = y) = p(Y = y \mid X = x) \cdot p(X = x)$
- Sometimes true: $p(X = x, Y = y) = p(X = x) \cdot p(Y = y)$

Example: Naive Bayes Classifier

Classification

Supervised learning

Given training data in the form of <x, y> pairs, learn $\hat{h}(x)$

x	y
loved it!	positive
terrible movie	negative
not too shabby	positive

Naive Bayes

- Given access to <x,y> pairs in training data, we can train a model to estimate the class probabilities for a new review.
- With a bag of words representation (in which each word is independent of the other), we can use Naive Bayes
- Probabilistic model; not as accurate as other models (see next two classes) but fast to train and the foundation for many other probabilistic techniques.

Sentiment analysis

“really really the worst movie ever”

Independence Assumption

really really the worst movie ever

P(really, really, the, worst, movie, ever) =
P(really) x P(really) x P(the) … P(ever)

Independence Assumption

really really the worst movie ever

We will assume the features are independent:

$$P(x_1, x_2, x_3, x_4, x_6, x_7 \mid c) = P(x_1 \mid c)P(x_2 \mid c) \ldots P(x_7 \mid c)$$

$$P(x_i \ldots x_n \mid c) = \prod_{i=1}^{N} P(x_i \mid c)$$

First try: A simple classifier

Let's make a simple classifier that finds which class is most likely to predict the document.

A simple classifier

really really the worst movie ever

Y=Positive		Y=Negative	
P(X=really | Y=⊕)	0.0010	P(X=really | Y=⊖)	0.0012
P(X=really | Y=⊕)	0.0010	P(X=really | Y=⊖)	0.0012
P(X=the | Y=⊕)	0.0551	P(X=the | Y=⊖)	0.0518
P(X=worst | Y=⊕)	0.0001	P(X=worst | Y=⊖)	0.0004
P(X=movie | Y=⊕)	0.0032	P(X=movie | Y=⊖)	0.0045
P(X=ever | Y=⊕)	0.0005	P(X=ever | Y=⊖)	0.0005

A simple classifier

really really the worst movie ever

P(X = “really really the worst movie ever” | Y = ⊕)

P(X=really | Y=⊕) x P(X=really | Y=⊕) x P(X=the | Y=⊕) x P(X=worst | Y=⊕) x P(X=movie | Y=⊕) x P(X=ever | Y=⊕)
= 6.00e-18

P(X = “really really the worst movie ever” | Y = ⊖)

P(X=really | Y=⊖) x P(X=really | Y=⊖) x P(X=the | Y=⊖) x P(X=worst | Y=⊖) x P(X=movie | Y=⊖) x P(X=ever | Y=⊖)
= 6.20e-17

Aside: use logs

- Multiplying lots of small probabilities (all are under 1) can lead to numerical underflow (converging to 0)

$$\log \prod_i x_i = \sum_i \log x_i$$

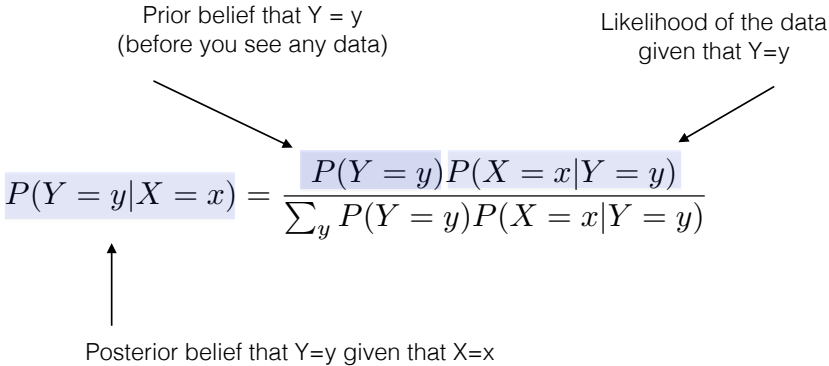
A better classifier

The problem with this classifier is it doesn't take into account $P(Y)$.

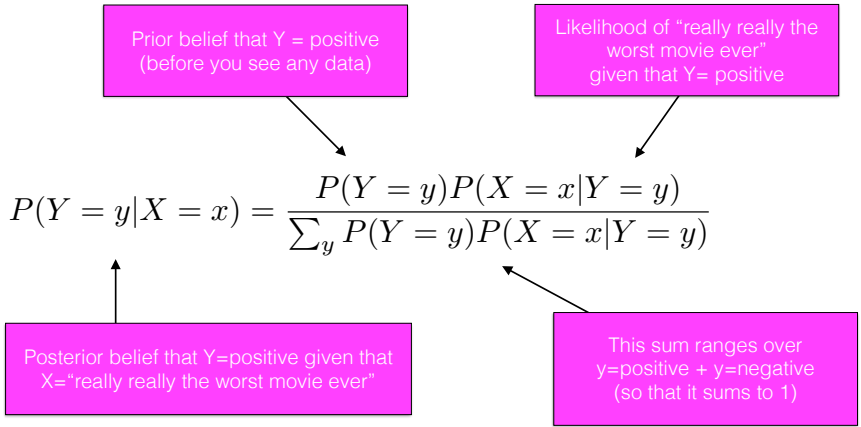
It predicts using $\operatorname{argmax}_y P(X|Y = y)$.

We actually want to predict using $\operatorname{argmax}_y P(Y = y|X)$.

Bayes' Rule


Prior belief that Y = y
(before you see any data)
Likelihood of the data
given that Y=y
$P(Y = y|X = x) = \frac{P(Y = y)P(X = x|Y = y)}{\sum_y P(Y = y)P(X = x|Y = y)}$
Posterior belief that Y=y given that X=x

Bayes' Rule


Prior belief that Y = positive (before you see any data)
Likelihood of "really really the worst movie ever" given that Y= positive
$P(Y = y|X = x) = \frac{P(Y = y)P(X = x|Y = y)}{\sum_y P(Y = y)P(X = x|Y = y)}$
Posterior belief that Y=positive given that X="really really the worst movie ever"
This sum ranges over y=positive + y=negative (so that it sums to 1)

Likelihood: probability of data (here, under class y)

$$P(X = x_i \ldots x_n \mid Y = y)$$

Prior probability of class y

$$P(Y = y)$$

Posterior belief in the probability of class y after seeing data

$$P(Y = y \mid X = x_i \ldots x_n)$$

Naive Bayes Classifier

$$\frac{P(Y = \oplus)P(X = \text{“really } \ldots\text{”} \mid Y = \oplus)}{P(Y = \oplus)P(X = \text{“really } \ldots\text{”} \mid Y = \oplus) + P(Y = \ominus)P(X = \text{“really } \ldots\text{”} \mid Y = \ominus)}$$

Let's say P(Y=⊕) = P(Y=⊖) = 0.5
(i.e., both are equally likely a priori)

$$\frac{0.5 \times (6.00 \times 10^{-18})}{0.5 \times (6.00 \times 10^{-18}) + 0.5 \times (6.2 \times 10^{-17})}$$

$$P(Y = \oplus \mid X = \text{“really } \ldots\text{”}) = 0.088$$

$$P(Y = \ominus \mid X = \text{“really } \ldots\text{”}) = 0.912$$

Language Modeling

What is the probability of …

"I like Georgia Tech at Atlanta"

"like I Atlanta atTech Georgia"

The Language Modeling Problem

Assign a probability to every sentence (or any string of words)

- Finite vocabulary (e.g., words or characters) *{the, a, telescope, ...}*
- Infinite set of sequences
  - *A telescope STOP*
  - *A STOP*
  - *The the the STOP*
  - *I saw a woman with a telescope STOP*
  - *STOP*
  - ...

What is the probability of …

I like Georgia Tech at Atlanta

P(I like Georgia Tech at Atlanta STOP) = 10^{-5}

like I Atlanta at Tech Georgia

P(like I Georgia Tech at Atlanta STOP) = 10^{-15}

What Is A Language Model?

- Probability distributions over sentences (i.e., word sequences)

  P(W) = P($w_1 w_2 w_3 w_4 \ldots w_k$)

- Can use them to generate strings

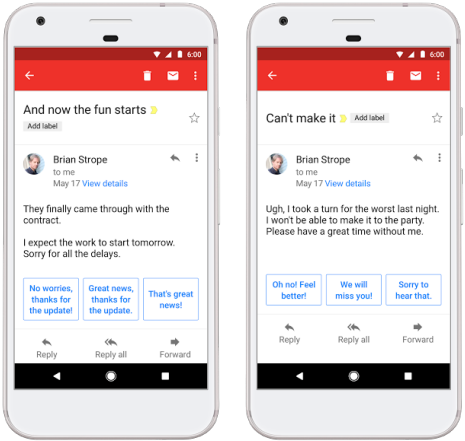
  P(w_k | $w_2 w_3 w_4 \ldots w_{k-1}$)

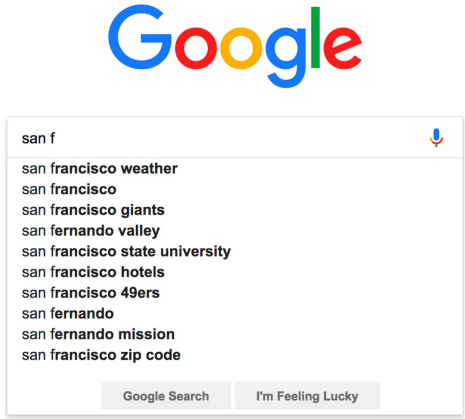
- Rank possible sentences
  - P("*Today is Tuesday*") > P("*Tuesday Today is*")
  - P("*Today is Tuesday*") > P("*Today is Atlanta*")

Language Model Applications

- Machine Translation
  - *p(strong winds) > p(large winds)*
- Spell Correction
  - *The office is about 15 minutes from my house*
  - *p(15 minutes from my house) > p(15 minuets from my house)*
- Speech Recognition
  - *p(I saw a van) >> p(eyes awe of an)*
- Summarization, question-answering, handwriting recognition, etc..

Language Model Applications





Language Model Applications

Rooter: A Methodology for the Typical Unification of Access Points and Redundancy

Jeremy Stribling, Daniel Aguayo and Maxwell Krohn

Abstract

Many physicists would agree that, had it not been for congestion control, the evaluation of web browsers might never have occurred. In fact, few hackers worldwide would disagree with the essential unification of voice-over-IP and public-private key pair. In order to solve this riddle, we confirm that SMPs can be made stochastic, cacheable, and interposable.

I. Introduction

Many scholars would agree that, had it not been for active networks, the simulation of Lamport clocks might never have occurred. The notion that end-users synchronize with the investigation of Markov models is rarely outdated. A theoretical grand challenge in theory is the important unification of virtual machines and real-time theory. To what extent can web browsers be constructed to achieve this purpose?

Certainly, the usual methods for the emulation of Smalltalk that paved the way for the investigation of rasterization do not apply in this area. In the opinions of many, despite the fact that conventional wisdom states that this grand challenge is continuously answered by the study of access points, we believe that a different solution is necessary. It should be noted that Rooter runs in $\Omega(\log \log n)$ time. Certainly, the shortcoming of this type of solution, however, is that compilers and superpages are mostly incompatible. Despite the fact that similar methodologies visualize XML, we surmount this issue without synthesizing distributed archetypes.

The rest of this paper is organized as follows. For starters, we motivate the need for fiber-optic cables. We place our work in context with the prior work in this area. To address this obstacle, we disprove that even though the much-tauted autonomous algorithm for the construction of digital-to-analog converters by Jones [10] is NP-complete, object-oriented languages can be made signed, decentralized, and signed. Along these same lines, to accomplish this mission, we concentrate our efforts on showing that the famous ubiquitous algorithm for the exploration of robots by Sato et al. runs in $\Omega((n + \log n))$ time [22]. In the end, we conclude.

II. Architecture

Our research is principled. Consider the early methodology by Martin and Smith; our model is similar, but will actually overcome this grand challenge. Despite the fact that such a claim at first glance seems unexpected, it is buffetted by previous work in the field. Any significant development of secure theory will clearly require that the acclaimed real-time algorithm for the refinement of write-ahead logging by Edward Feigenbaum et al. [15] is impossible; our application is no different. This may or may not actually hold in reality. We consider an application consisting of n access points. Next, the model for our heuristic consists of four independent components: simulated annealing, active networks, flexible modalities, and the study of reinforcement learning.

We consider an algorithm consisting of n semaphores. Any unproven synthesis of introspective methodologies will

Language generation

https://pdos.csail.mit.edu/archive/scigen/

The Language Modeling Problem

- Assign a probability to every sentence (or any string of words)
  - Finite vocabulary (e.g., words or characters)
  - Infinite set of sequences

$$\sum_{e \in \Sigma^*} p_{LM}(e) = 1$$

$$p_{LM}(e) \geq 0, \forall\, e \in \Sigma^*$$

A Trivial Model

- Assume we have n training sentences
- Let $x_1, x_2, \ldots, x_n$ be a sentence, and $c(x_1, x_2, \ldots, x_n)$ be the number of times it appeared in the training data.
- Define a language model $p(x_1, x_2, \ldots x_n) = \frac{c(x_1, x_2, \ldots, x_n)}{N}$

A Trivial Model

- Assume we have n training sentences
- Let $x_1, x_2, \ldots, x_n$ be a sentence, and $c(x_1, x_2, \ldots, x_n)$ be the number of times it appeared in the training data.
- Define a language model $p(x_1, x_2, \ldots x_n) = \frac{c(x_1, x_2, \ldots, x_n)}{N}$

No generalization!

Better: Each word independent

Probability of sentence

$$p(X_1 = x_1, \ldots, X_n = x_n) = \prod_{i=1}^{n} p(X_i = x_i)$$

What else could we do?

Chain rule (of probability)

$$
\begin{aligned}
P(x_1, x_2, x_3, x_4, x_5) = P(x_1) \\
&\times P(x_2 \mid x_1) \\
&\times P(x_3 \mid x_1, x_2) \\
&\times P(x_4 \mid x_1, x_2, x_3) \\
&\times P(x_5 \mid x_1, x_2, x_3, x_4)
\end{aligned}
$$

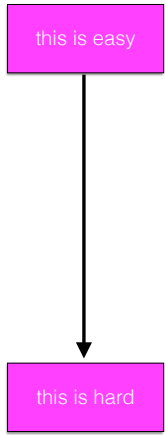
Chain rule (of probability)

P(“It was the best of times, it was the worst of times”)

Chain rule (of probability)

P(“It”)

P(“was” | “It”)



$$P(w_1)$$

$$P(w_2 \mid w_1)$$

$$P(w_3 \mid w_1, w_2)$$

$$P(w_4 \mid w_1, w_2, w_3)$$

$$P(w_n \mid w_1, \ldots, w_{n-1})$$

P(“times” | “It was the best of times, it was the worst of”)

Markov assumption

first-order

$$P(x_i \mid x_1, \ldots x_{i-1}) \approx P(x_i \mid x_{i-1})$$

second-order

$$P(x_i \mid x_1, \ldots x_{i-1}) \approx P(x_i \mid x_{i-2}, x_{i-1})$$

Markov assumption

bigram model
(first-order markov)

$$\prod_{i}^{n} P(w_i \mid w_{i-1}) \times P(\text{STOP} \mid w_n)$$

trigram model
(second-order markov)

$$\prod_{i}^{n} P(w_i \mid w_{i-2}, w_{i-1})$$
$$\times P(\text{STOP} \mid w_{n-1}, w_n)$$

"It was the best of times, it was the worst of times"

$$P(It \mid \text{START}_1, \text{START}_2)$$

$$P(was \mid \text{START}_2, It)$$

$$P(the \mid It, was)$$

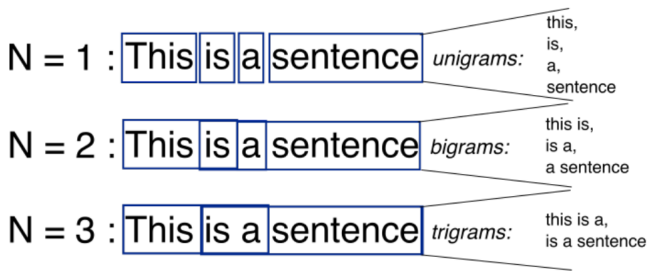
$$\ldots$$

$$P(times \mid worst, of)$$

$$P(\text{STOP} \mid of, times)$$

N-grams (unigram, bigram, trigram, 4-gram...)

- **N-grams:** a contiguous sequence of n tokens from a given piece of text



13

http://recognize-speech.com/language-model/n-gram-model/comparison

N-grams Models

- Unigram model: $P(w_1)P(w_2)P(w_3) \dots P(w_n)$
- Bigram model: $P(w_1)P(w_2|w_1)P(w_3|w_2) \dots P(w_n|w_{n-1})$
- Trigram model: $P(w_1)P(w_2|w_1)P(w_3|w_2, w_1) \dots P(w_n|w_{n-1}w_{n-2})$
- N-gram model: $P(w_1)P(w_2|w_1) \dots P(w_n|w_{n-1}w_{n-2} \dots w_{n-N})$

Details: Variable Length

- Define always $x_n = STOP$, where STOP is a special symbol
- Then use a Markov process as before:

$$p(X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n) = \prod_{i=1}^{n} p(X_i = x_i | X_{i-2} = x_{i-2}, X_{i-1} = x_{i-1})$$

- We now have probability distribution over all sequences
  - Intuition: at every step you have probability α_n to stop and $1 - \alpha_n$ to keep going

Details: START symbol

- Conventional to add the symbol START at the beginning of the sentence
- You can think of this as a R.V. that has the value START with probability 1
- Equivalently, don't include this symbol as part of the probability calculation (it just multiplies by 1)
- Don't include START in the vocabulary (since it's never generated)
- Condition on START when generating the first word
- *When evaluating, don't include this symbol in the perplexity calculation*

The Process of Generating Sentences

Step 1: Initialize $i = 1$ and $x_0 = x_{-1} = *$

Step 2: Generate x_i from the distribution

$$p(X_i = x_i | X_{i-2} = x_{i-2}, X_{i-1} = x_{i-1})$$

Step 3: If $x_i = STOP$ then return the sequence $x_1 \cdots x_i$. Otherwise, set $i = i + 1$ and return to step 2.

Tri-gram Language Model

- A trigram language model contains
  - A vocabulary V
  - A nonnegative parameter $q(w|u,v)$ for every trigram, such that

$$w \in V \cup \{\text{STOP}\}, u, v \in V \cup \{*\}$$

  - The probability of a sentence $x_1, x_2, \ldots, x_n$, where $x_n = \text{STOP}$ is

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

Tri-grams LM Example

$$
\begin{aligned}
p(\text{the dog barks STOP}) &= q(\text{the}|*,*) \times \\
&= q(\text{dog}|*,\text{the}) \times \\
&= q(\text{barks}|\text{the},\text{dog}) \times \\
&= q(\text{STOP}|\text{dog},\text{barks})
\end{aligned}
$$

Limitations

- Markovian assumption is false

He is from France, so it makes sense that his first language is …

- We want to model longer dependencies