Independence

Two variables are independent if they do not affect each other.

For two independent variables, knowing the value of one does not change the probability distribution of the other variable:
- The result of the toss of a coin is independent of a roll of a dice.
- Price of tea in England is independent of whether or not you get an A in NLP.

Independent or Dependent?

- You catching a cold and a butterfly flapping its wings in Africa.
- Miles per gallon and driving habits.
- Height and longevity of life.
Independent variables

How does independence affect our probability equations/properties?

If A and B are independent, written $A \perp B$

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

What does that mean about $P(A,B)$?

Conditional Independence

Dependent events can become independent given certain other events

Examples:
- height and length of life
- "correlation" studies
- size of your lawn and length of life

If A, B are conditionally independent given C, $A \perp B | C$

- $P(A,B|C) = P(A|C)P(B|C)$
- $P(A|B,C) = P(A|C)$
- $P(B|A,C) = P(B|C)$
- but $P(A,B) \neq P(A)P(B)$
Assume independence

Sometimes we will assume two variables are independent (or conditionally independent) even though they’re not

Why?
- Creates a simpler model
  - $P(X,Y)$ many more variables than just $P(X)$ and $P(Y)$
- May not be able to estimate the more complicated model

Language modeling

What does natural language look like?

More specifically in NLP, probabilistic model

$p(\text{sentence})$
- $p(\text{"I like to eat pizza"})$
- $p(\text{"pizza like I eat"})$

Often is posed as: $p(\text{word} | \text{previous words})$
- $p(\text{"pizza"} | \text{"I like to eat"})$
- $p(\text{"garbage"} | \text{"I like to eat"})$
- $p(\text{"run"} | \text{"I like to eat"})$

Language modeling

How might these models be useful?
- Language generation tasks
  - machine translation
  - summarization
  - simplification
  - speech recognition
  - ...
- Text correction
  - spelling correction
  - grammar correction

Ideas?

$p(\text{"I like to eat pizza"})$

$p(\text{"pizza like I eat"})$

$p(\text{"pizza"} | \text{"I like to eat"})$

$p(\text{"garbage"} | \text{"I like to eat"})$

$p(\text{"run"} | \text{"I like to eat"})$
Look at a corpus

Language modeling

I think today is a good day to be me

Language modeling is about dealing with data sparsity!

Probabilistic Language modeling

A probabilistic explanation of how the sentence was generated

Key idea:
- break this generation process into smaller steps
- estimate the probabilities of these smaller steps
- the overall probability is the combined product of the steps

Language modeling

Many approaches:
- n-gram language modeling
  - Start at the beginning of the sentence
  - Generate one word at a time based on the previous words
- syntax-based language modeling
  - Construct the syntactic tree from the top down
  - e.g. context free grammar
  - eventually at the leaves, generate the words

Pros/cons?
**n-gram language modeling**

I think today is a good day to be me

Google: “I think”

Web: Results 1 - 10 of about 564,000,000 for “I think” (0.33 seconds)

Google: “today is a good day”

Web: Results 1 - 10 of about 10,100,000 for “today is a good day”

Google: “to be me”

Web: Results 1 - 10 of about 70,200,000 for “to be me”

---

**Our friend the chain rule**

Step 1: decompose the probability

\[
P(I \text{ think today is a good day to be me}) =
\]

\[
P(I | <\text{start}> ) \times
\]

\[
P(\text{think} | I ) \times
\]

\[
P(\text{today} | I \text{ think}) \times
\]

\[
P(\text{is} | I \text{ think today}) \times
\]

\[
P(\text{a} | I \text{ think today is}) \times
\]

\[
P(\text{good} | I \text{ think today is a}) \times
\]

... 

How can we simplify these?

---

**The n-gram approximation**

Assume each word depends only on the previous n-1 words (e.g. trigram: three words total)

\[
P(\text{is} | I \text{ think today}) = P(\text{is} | \text{think today})
\]

\[
P(\text{a} | I \text{ think today is}) = P(\text{a} | \text{today is})
\]

\[
P(\text{good} | I \text{ think today is a}) = P(\text{good} | \text{is a})
\]

---

**Estimating probabilities**

How do we find probabilities? 

P(is | think today)

Get real text, and start counting (MLE!)

\[
P(is | \text{think today}) = \frac{\text{count}(\text{think today is})}{\text{count}(\text{think today})}
\]
Estimating from a corpus

Corpus of sentences
(e.g. gigaword corpus)

\[ \vdots \]

\[ \vdots \]

n-gram language model

\[ ? \]

Estimating from a corpus

\[ I \text{ am a happy Pomona College student .} \]

\[ \text{count all of the trigrams} \]

\[ \text{\langle start \rangle} \text{ \langle start \rangle} \text{ I} \]
\[ \text{\langle start \rangle} \text{ I am} \]
\[ \text{I am a} \]
\[ \text{a happy} \]
\[ \text{happy Pomona} \]
\[ \text{Pomona College student} \]
\[ \text{College student} \]
\[ . \]
\[ . \]
\[ . \]

\[ \text{why do we need} \]
\[ \text{\langle start \rangle} \text{ and \langle end \rangle}? \]

Estimating from a corpus

\[ I \text{ am a happy Pomona College student .} \]

\[ \text{count all of the bigrams} \]

\[ \text{\langle start \rangle} \text{ \langle start \rangle} \text{ I} \]
\[ \text{\langle start \rangle} \text{ I am} \]
\[ \text{I am a} \]
\[ \text{a happy} \]
\[ \text{happy Pomona} \]
\[ \text{Pomona College student} \]
\[ \text{College student} \]
\[ . \]
\[ . \]
\[ . \]

\[ p(c | a \ b) = \frac{\text{count}(a \ b \ c)}{\text{count}(a \ b)} \]
Estimating from a corpus

1. Go through all sentences and count trigrams and bigrams
   - usually you store these in some kind of data structure

2. Now, go through all of the trigrams and use the count and the bigram count to calculate MLE probabilities
   - do we need to worry about divide by zero?

Applying a model

Given a new sentence, we can apply the model

\[ p(\text{Pomona College students are the best.}) = \] ?

- \[ p(\text{Pomona | <start> <start> }) \]
- \[ p(\text{College | <start> Pomona }) \]
- \[ p(\text{students | Pomona College }) \]
- \[ \vdots \]
- \[ p(\text{<end> | , <end>}) \]

Generating examples

We can also use a trained model to generate a random sentence

Ideas?

\[ <\text{start}> <\text{start}> \]

We have a distribution over all possible starting words

\[ p(\text{A | <start> <start> }) \]
\[ p(\text{Apples | <start> <start> }) \]
\[ p(\text{I | <start> <start> }) \]
\[ p(\text{The | <start> <start> }) \]
\[ \vdots \]
\[ p(\text{Zebras | <start> <start> }) \]

Drawing one from this distribution

Generating examples

\[ <\text{start}> <\text{start}> \text{Zebras} \]

Repeat!

\[ p(\text{are | <start> Zebras}) \]
\[ p(\text{eat | <start> Zebras}) \]
\[ p(\text{think | <start> Zebras}) \]
\[ p(\text{and | <start> Zebras}) \]
\[ \vdots \]
\[ p(\text{mostly | <start> Zebras}) \]
### Generation examples

#### Unigram

<table>
<thead>
<tr>
<th>Unigram</th>
</tr>
</thead>
<tbody>
<tr>
<td>are were that ėres mammal naturally built describes jazz territory</td>
</tr>
<tr>
<td>heteromyids film tenor prime live founding must an vas feet negro</td>
</tr>
<tr>
<td>legal gate in on beside . provincial sim ; stephens simply spaces</td>
</tr>
<tr>
<td>stretched performance double-entry grove replacing station across to</td>
</tr>
<tr>
<td>burma . repairing ėres capital about double reached omnibus et time</td>
</tr>
<tr>
<td>believed what hotels parameter jurisprudence words syndrome to ėres</td>
</tr>
<tr>
<td>profanity is administrators ėres offices hilarious institutionalized</td>
</tr>
<tr>
<td>remains writer royalty dennis , ėres tyson , and objective ,</td>
</tr>
<tr>
<td>instructions seem timekeeper has ėres valley ėres ” magnitudes for</td>
</tr>
<tr>
<td>love on ėres from allsasket , , one central enlightened . to ėres</td>
</tr>
<tr>
<td>belongs fame they the corrected , on in pressure %NUMBER% her</td>
</tr>
<tr>
<td>flavored ėres derogatory is won metcard indirectly of crop duty</td>
</tr>
<tr>
<td>learn northbound ėres ėres dancing similarity ėres named ėres</td>
</tr>
<tr>
<td>berkley , off-scale overtime . each manifold stripes ähnlich</td>
</tr>
<tr>
<td>traffic assets and at alpha popularity town</td>
</tr>
</tbody>
</table>

#### Bigrams

<table>
<thead>
<tr>
<th>Bigrams</th>
</tr>
</thead>
<tbody>
<tr>
<td>the wikipedia county , mexico .</td>
</tr>
<tr>
<td>maurice ravel . it is require that is sparta , where functions .</td>
</tr>
<tr>
<td>most widely admired .</td>
</tr>
<tr>
<td>hologens chamiali cast jason against test site .</td>
</tr>
</tbody>
</table>

#### Trigrams

<table>
<thead>
<tr>
<th>Trigrams</th>
</tr>
</thead>
<tbody>
<tr>
<td>is widespread in north africa in june %NUMBER% %NUMBER% units were</td>
</tr>
<tr>
<td>built by with .</td>
</tr>
<tr>
<td>jewish video spiritual are considered rival , this season was an</td>
</tr>
<tr>
<td>extratropical cyclone .</td>
</tr>
<tr>
<td>the british railways ’ s strong and a spot .</td>
</tr>
</tbody>
</table>

### Evaluation

We can train a language model on some data

How can we tell how well we’re doing?

- for example
  - bigrams vs. trigrams
  - 100K sentence corpus vs. 100M
  - ...
Evaluation

A very good option: extrinsic evaluation

If you're going to be using it for machine translation
- build a system with each language model
- compare the two based on their approach for machine translation

Sometimes we don't know the application

Can be time consuming

Granularity of results

Evaluation

Common NLP/machine learning/AI approach

Training sentences

All sentences

Testing sentences

Evaluation

A good model should do a good job of predicting actual sentences

Ideas?
Evaluation

Pros: Fine for comparing two models
Cons: Doesn’t give us a sense of how well any model is doing

Test sentences

model 1

model 2

The problem

Which of these sentences will have a higher probability based on a language model?

I like to eat banana peels.

I like to eat banana peels with peanut butter.

Since probabilities are multiplicative (and between 0 and 1), they get smaller for longer sentences.

The solution: perplexity

\[
prob(w_{1:n}) = \prod_{i=1}^{n} p(w_i|w_{1:i-1})
\]

average the probabilities

geometric mean

\[
PP(w_{1:n}) = \sqrt[n]{\prod_{i=1}^{n} p(w_i|w_{1:i-1})}
\]
Calculating perplexity in practice

\[
\log \left( \sqrt[1/n]{ \prod_{i=1}^{n} p(w_i|w_{1:i-1}) } \right) = \log \left( \frac{1}{\prod_{i=1}^{n} p(w_i|w_{1:i-1})} \right) \\
= \log \left( \frac{1}{\prod_{i=1}^{n} p(w_i|w_{1:i-1})} \right) \\
= -\log(\prod_{i=1}^{n} p(w_i|w_{1:i-1})) \\
= -\frac{\sum_{i=1}^{n} \log p(w_i|w_{1:i-1})}{n}
\]

What is this?

Average logprob per word!

Calculating perplexity

\[
PP(w_{1:n}) = \sqrt[1/n]{ \prod_{i=1}^{n} p(w_i|w_{1:i-1}) } \\
= 10^{-\frac{\sum_{i=1}^{n} \log p(w_i|w_{1:i-1})}{n}}
\]

- This is often how it’s calculated (and how we’ll calculate it)
- Avoid underflow from multiplying too many small probabilities together

Another view of perplexity

Weighted average branching factor
- number of possible next words that can follow a word or phrase
- measure of the complexity/uncertainty of text (as viewed from the language models perspective)
Smoothing

What if our test set contains the following sentence, but one of the trigrams never occurred in our training data?

\[ P(\text{I think today is a good day to be me}) = \]

\[ P(\text{I} \mid \text{<start>} \text{<start>}) \times P(\text{think} \mid \text{I}) \times P(\text{today} \mid \text{think}) \times P(\text{is} \mid \text{think today}) \times P(\text{a} \mid \text{today is}) \times P(\text{good} \mid \text{is a}) \times \ldots \]

If any of these has never been seen before, prob = 0!

A better approach

\[ p(z \mid x y) = ? \]

Suppose our training data includes

\[ \ldots x y a \ldots \]
\[ \ldots x y d \ldots \]
\[ \ldots x y d \ldots \]
but never: xyz

We would conclude

\[ p(a \mid x y) = 1/3? \]
\[ p(d \mid x y) = 2/3? \]
\[ p(z \mid x y) = 0/3? \]

Is this ok?

Intuitively, how should we fix these?

Smoothing the estimates

Basic idea:

- \( p(a \mid x y) = 1/3 \) reduce
- \( p(d \mid x y) = 2/3 \) reduce
- \( p(z \mid x y) = 0/3 \) increase

Discount the positive counts somewhat

Reallocate that probability to the zeroes

Remember, it needs to stay a probability distribution

Other situations

\[ p(z \mid x y) = ? \]

Suppose our training data includes

\[ \ldots x y a \ldots (100 \text{ times}) \]
\[ \ldots x y d \ldots (100 \text{ times}) \]
\[ \ldots x y d \ldots (100 \text{ times}) \]
but never: xyz

Suppose our training data includes

\[ \ldots x y a \ldots \]
\[ \ldots x y d \ldots \]
\[ \ldots x y d \ldots (300 \text{ times}) \]
but never: xyz

Is this the same situation as before?
### Smoothing the estimates

Should we conclude

\[ p(a \mid xy) = \frac{1}{3}? \]
\[ p(d \mid xy) = \frac{2}{3}? \]
\[ p(z \mid xy) = \frac{0}{3}? \]

Reduce \( p(c \mid a b) = \frac{\text{count}(a b c)}{\text{count}(a b)} \)

Increase

Readjusting the estimate is particularly important if:

- the denominator is small …
  - \( \frac{1}{3} \) probably too high, \( \frac{100}{300} \) probably about right
- numerator is small …
  - \( \frac{1}{300} \) is probably too high, \( \frac{100}{300} \) probably about right

### Add-one (Laplacian) smoothing

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>1/3</th>
<th>2</th>
<th>2/29</th>
</tr>
</thead>
<tbody>
<tr>
<td>xya</td>
<td>1</td>
<td>1/3</td>
<td>2</td>
<td>2/29</td>
</tr>
<tr>
<td>xyb</td>
<td>0</td>
<td>0/3</td>
<td>1</td>
<td>1/29</td>
</tr>
<tr>
<td>xyc</td>
<td>0</td>
<td>0/3</td>
<td>1</td>
<td>1/29</td>
</tr>
<tr>
<td>xyd</td>
<td>2</td>
<td>2/3</td>
<td>3</td>
<td>3/29</td>
</tr>
<tr>
<td>xye</td>
<td>0</td>
<td>0/3</td>
<td>1</td>
<td>1/29</td>
</tr>
<tr>
<td>…</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>xyz</td>
<td>0</td>
<td>0/3</td>
<td>1</td>
<td>1/29</td>
</tr>
<tr>
<td>Total xy</td>
<td>3</td>
<td>3/3</td>
<td>29</td>
<td>29/29</td>
</tr>
</tbody>
</table>

**Add-one (Laplacian) smoothing**

300 observations instead of 3 – better data, less smoothing

<table>
<thead>
<tr>
<th></th>
<th>100</th>
<th>100/300</th>
<th>101</th>
<th>101/326</th>
</tr>
</thead>
<tbody>
<tr>
<td>xya</td>
<td>100</td>
<td>100/300</td>
<td>101</td>
<td>101/326</td>
</tr>
<tr>
<td>xyb</td>
<td>0</td>
<td>0/300</td>
<td>1</td>
<td>1/326</td>
</tr>
<tr>
<td>xyc</td>
<td>0</td>
<td>0/300</td>
<td>1</td>
<td>1/326</td>
</tr>
<tr>
<td>xyd</td>
<td>200</td>
<td>200/300</td>
<td>201</td>
<td>201/326</td>
</tr>
<tr>
<td>xye</td>
<td>0</td>
<td>0/300</td>
<td>1</td>
<td>1/326</td>
</tr>
<tr>
<td>…</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>xyz</td>
<td>0</td>
<td>0/300</td>
<td>1</td>
<td>1/326</td>
</tr>
<tr>
<td>Total xy</td>
<td>300</td>
<td>300/300</td>
<td>326</td>
<td>326/326</td>
</tr>
</tbody>
</table>

**Add-one (Laplacian) smoothing**

What happens if we’re now considering a vocabulary of 20,000 words?

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>1/3</th>
<th>2</th>
<th>2/29</th>
</tr>
</thead>
<tbody>
<tr>
<td>xya</td>
<td>1</td>
<td>1/3</td>
<td>2</td>
<td>2/29</td>
</tr>
<tr>
<td>xyb</td>
<td>0</td>
<td>0/3</td>
<td>1</td>
<td>1/29</td>
</tr>
<tr>
<td>xyc</td>
<td>0</td>
<td>0/3</td>
<td>1</td>
<td>1/29</td>
</tr>
<tr>
<td>xyd</td>
<td>2</td>
<td>2/3</td>
<td>3</td>
<td>3/29</td>
</tr>
<tr>
<td>xye</td>
<td>0</td>
<td>0/3</td>
<td>1</td>
<td>1/29</td>
</tr>
<tr>
<td>…</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>xyz</td>
<td>0</td>
<td>0/3</td>
<td>1</td>
<td>1/29</td>
</tr>
<tr>
<td>Total xy</td>
<td>3</td>
<td>3/3</td>
<td>29</td>
<td>29/29</td>
</tr>
</tbody>
</table>
Add-one (Laplacian) smoothing

20,000 words, not 26 letters

<table>
<thead>
<tr>
<th>Event</th>
<th>Observation</th>
<th>Count</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>see the abacus</td>
<td>1</td>
<td>1/3</td>
<td>2/20003</td>
</tr>
<tr>
<td>see the abbot</td>
<td>0</td>
<td>0/3</td>
<td>1/20003</td>
</tr>
<tr>
<td>see the abduct</td>
<td>0</td>
<td>0/3</td>
<td>1/20003</td>
</tr>
<tr>
<td>see the above</td>
<td>2</td>
<td>2/3</td>
<td>3/20003</td>
</tr>
<tr>
<td>see the Abram</td>
<td>0</td>
<td>0/3</td>
<td>1/20003</td>
</tr>
<tr>
<td>...</td>
<td>0</td>
<td>0/3</td>
<td>1/20003</td>
</tr>
</tbody>
</table>

Total: 3 3/3 20003

Any problem with this?

Add-lambda smoothing

A large dictionary makes novel events too probable.

Instead of adding 1 to all counts, add $\lambda = 0.01$?

<table>
<thead>
<tr>
<th>Event</th>
<th>Observation</th>
<th>Count</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>see the abacus</td>
<td>1</td>
<td>1/3</td>
<td>1.01 1.01/203</td>
</tr>
<tr>
<td>see the abbot</td>
<td>0</td>
<td>0/3</td>
<td>0.01 0.01/203</td>
</tr>
<tr>
<td>see the abduct</td>
<td>0</td>
<td>0/3</td>
<td>0.01 0.01/203</td>
</tr>
<tr>
<td>see the above</td>
<td>2</td>
<td>2/3</td>
<td>2.01 2.01/203</td>
</tr>
<tr>
<td>see the Abram</td>
<td>0</td>
<td>0/3</td>
<td>0.01 0.01/203</td>
</tr>
<tr>
<td>...</td>
<td>0</td>
<td>0/3</td>
<td>0.01 0.01/203</td>
</tr>
</tbody>
</table>

Total: 3 3/3 203