Recap from last time...
Three people have been fatally shot, and five people, including a mayor, were seriously wounded as a result of a Shining Path attack today.

$p_{\theta}(\cdot)$
Chain Rule
+
Backoff (Markov assumption)
=
n-grams
# N-Gram Terminology

how to (efficiently) compute $p$(Colorless green ideas sleep furiously)?

<table>
<thead>
<tr>
<th>$n$</th>
<th>Commonly called</th>
<th>History Size (Markov order)</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>unigram</td>
<td>0</td>
<td>$p$(furiously)</td>
</tr>
<tr>
<td>2</td>
<td>bigram</td>
<td>1</td>
<td>$p$(furiously</td>
</tr>
<tr>
<td>3</td>
<td>trigram (3-gram)</td>
<td>2</td>
<td>$p$(furiously</td>
</tr>
<tr>
<td>4</td>
<td>4-gram</td>
<td>3</td>
<td>$p$(furiously</td>
</tr>
<tr>
<td>$n$</td>
<td>$n$-gram</td>
<td>$n$-1</td>
<td>$p(w_i</td>
</tr>
</tbody>
</table>
Count-Based N-Grams (Unigrams)

\[ p(z) \propto \text{count}(z) \]
\[ = \frac{\text{count}(z)}{\sum_v \text{count}(v)} \]
Count-Based N-Grams (Trigrams)

\[ p(z|x, y) \propto \frac{\text{count}(x, y, z)}{\sum_v \text{count}(x, y, v)} \]
Add-$\lambda$ estimation

Laplace smoothing,
Lidstone smoothing

Pretend we saw each word $\lambda$ more times than we did

$$p(z) \propto \text{count}(z) + \lambda$$

Add $\lambda$ to all the counts
Linear Interpolation

\[ p(y \mid x) = \lambda p_2(y \mid x) + (1 - \lambda) p_1(y) \]

\[ 0 \leq \lambda \leq 1 \]

Simple interpolation --> Averaging
Discounted Backoff

Trust your statistics, up to a point

\[
p(z|x, y) = \begin{cases} 
    p_3(z|x, y) - d & \text{if } \text{count}(x, y, z) > 0 \\
    \beta(x, y)p_2(z|y) & \text{otherwise}
\end{cases}
\]
Evaluation Framework

**What is “correct?”**

**What is working “well?”**

- **Training Data**: acquire primary statistics for learning model parameters
- **Dev Data**: fine-tune any secondary (hyper)parameters
- **Test Data**: perform final evaluation

**DO NOT ITERATE ON THE TEST DATA**
Setting Hyperparameters

Use a development corpus

Choose λs to maximize the probability of dev data:
  Fix the N-gram probabilities counts (on the training data)
  Search for λs that give largest probability to held-out set
Evaluating Language Models

What is “correct?”

What is working “well?”

**Extrinsic:** Evaluate LM in downstream task

Test an MT, ASR, etc. system and see which LM does better

Propagate & conflate errors

**Intrinsic:** Treat LM as its own downstream task

Use perplexity (from information theory)
Perplexity

Lower is better: lower perplexity --> less surprised

\[ \text{perplexity} = \exp\left(\frac{-1}{M} \sum_{i=1}^{M} \log p(w_i | h_i)\right) \]
Maximum Likelihood Estimates

\[ p(\text{item}) \propto \text{count}(\text{item}) \]

Maximizes the likelihood of the training set

Do different corpora look the same?

For large data: can actually do reasonably well
Implementation: Unknown words

Create an unknown word token <UNK>

Training:
1. Create a fixed lexicon L of size V
2. Change any word not in L to <UNK>
3. Train LM as normal

Evaluation:
Use UNK probabilities for any word not in training
p(Colorless green ideas sleep furiously) =
   p(Colorless | <BOS> <BOS>) *
   p(green | <BOS> Colorless) *
   p(ideas | Colorless green) *
   p(sleep | green ideas) *
   p(furiously | ideas sleep) *
   p(<EOS> | sleep furiously)

**Consistent notation:** Pad the left with <BOS> (beginning of sentence) symbols

**Fully proper distribution:** Pad the right with a single <EOS> symbol
Implementation: EOS Padding

Create an end of sentence (“chunk”) token \(<\text{EOS}>\)

Don’t estimate \(p(<\text{BOS}> \mid <\text{EOS}>)\)

Training & Evaluation:

1. Identify “chunks” that are relevant (sentences, paragraphs, documents)
2. Append the \(<\text{EOS}>\) token to the end of the chunk
3. Train or evaluate LM as normal
Other Kinds of Smoothing

Interpolated (modified) Kneser-Ney

Idea: How “productive” is a context?
How many different word types \( v \) appear in a context \( x, y \)

Good-Turing

Partition words into classes of occurrence
Smooth class statistics
Properties of classes are likely to predict properties of other classes

Witten-Bell

Idea: Every observed type was at some point novel
Give MLE prediction for novel type occurring
Bayes Rule $\rightarrow$ NLP Applications

$$p(X \mid Y) = \frac{p(Y \mid X) \cdot p(X)}{p(Y)}$$

- posterior probability
- likelihood
- prior probability
- marginal likelihood (probability)
Two Different Philosophical Frameworks

\[ p(X \mid Y) = \frac{p(Y \mid X) \times p(X)}{p(Y)} \]

- **Posterior Classification/Decoding**
  - maximum a posteriori

- **Noisy Channel Model Decoding**

*there are others too (CMSC 478/678)*
Two Different Philosophical Frameworks

\[ p(X \mid Y) = \frac{p(Y \mid X) \times p(X)}{p(Y)} \]

\( p(Y \mid X) \) \textit{likelihood}
\( p(X) \) \textit{prior probability}
\( p(Y) \) \textit{marginal likelihood (probability)}

Posterior Classification/Decoding

\( p(X \mid Y) \) \textit{posterior probability}

Noisy Channel Model Decoding

\textit{maximum a posteriori}

\textit{there are others too (CMSC 478/678)}
Three people have been fatally shot, and five people, including a mayor, were seriously wounded as a result of a Shining Path attack today against a community in Junin department, central Peruvian mountain region.
Three people have been fatally shot, and five people, including a mayor, were seriously wounded as a result of a Shining Path attack today against a community in Junin department, central Peruvian mountain region.
Electronic alerts have been used to assist the authorities in moments of chaos and potential danger: after the Boston bombing in 2013, when the Boston suspects were still at large, and last month in Los Angeles, during an active shooter scare at the airport.
Electronic alerts have been used to assist the authorities in moments of chaos and potential danger: after the Boston bombing in 2013, when the Boston suspects were still at large, and last month in Los Angeles, during an active shooter scare at the airport.
Classify with Uncertainty

\[ \text{best label} = \arg \max_{\text{label}} P(\text{label}|\text{example}) \]

Use probabilities
Classify with Uncertainty

\[
\text{best label} = \arg \max_{\text{label}} P(\text{label}|\text{example})
\]

*Use probabilities*

*There are non-probabilistic ways to handle uncertainty... but probabilities sure are handy!*
Electronic alerts have been used to assist the authorities in moments of chaos and potential danger: after the Boston bombing in 2013, when the Boston suspects were still at large, and last month in Los Angeles, during an active shooter scare at the airport.
Text Classification

Assigning subject categories, topics, or genres
Spam detection
Authorship identification

Age/gender identification
Language Identification
Sentiment analysis
...
Text Classification

Assigning subject categories, topics, or genres
Spam detection
Authorship identification

Age/gender identification
Language Identification
Sentiment analysis

Input:
- a document
- a fixed set of classes $C = \{c_1, c_2, ..., c_J\}$

Output: a predicted class $c$ from $C$
Text Classification

Assigning subject categories, topics, or genres
Spam detection
Authorship identification

Age/gender identification
Language Identification
Sentiment analysis

Input:
a document linguistic blob
a fixed set of classes \( C = \{c_1, c_2, \ldots, c_J\} \)

Output: a predicted class \( c \) from \( C \)
Text Classification: Hand-coded Rules?

Assigning subject categories, topics, or genres
Spam detection
Authorship identification

Age/gender identification
Language Identification
Sentiment analysis
...

Rules based on combinations of words or other features
spam: black-list-address OR (“dollars” AND “have been selected”)

Accuracy can be high
If rules carefully refined by expert

Building and maintaining these rules is expensive
Can humans faithfully assign uncertainty?
Text Classification: Supervised Machine Learning

Assigning subject categories, topics, or genres
Spam detection
Authorship identification
Age/gender identification
Language Identification
Sentiment analysis

**Input:**
- a document $d$
- a fixed set of classes $C = \{c_1, c_2, \ldots, c_J\}$
- A training set of $m$ hand-labeled documents $(d_1, c_1), \ldots, (d_m, c_m)$

**Output:**
- a learned classifier $\gamma$ that maps documents to classes
Text Classification: Supervised Machine Learning

Assigning subject categories, topics, or genres
Spam detection
Authorship identification
Age/gender identification
Language Identification
Sentiment analysis

Input:
- a document \( d \)
- a fixed set of classes \( C = \{ c_1, c_2, \ldots, c_J \} \)
- A training set of \( m \) hand-labeled documents \( (d_1, c_1), \ldots, (d_m, c_m) \)

Output:
a learned classifier \( \gamma \) that maps documents to classes

Naïve Bayes
Logistic regression
Support-vector machines
k-Nearest Neighbors

...
Text Classification: Supervised Machine Learning

Assigning subject categories, topics, or genres
Spam detection
Authorship identification
Age/gender identification
Language Identification
Sentiment analysis

Input:
- a document \(d\)
- a fixed set of classes \(C = \{c_1, c_2, \ldots, c_J\}\)
- A training set of \(m\) hand-labeled documents \((d_1, c_1), \ldots, (d_m, c_m)\)

Output:
a learned classifier \(\gamma\) that maps documents to classes

Naïve Bayes
Logistic regression
Support-vector machines
k-Nearest Neighbors
Probabilistic Text Classification

Assigning subject categories, topics, or genres
Spam detection
Authorship identification

Age/gender identification
Language Identification
Sentiment analysis

...
Probabilistic Text Classification

Assigning subject categories, topics, or genres
Spam detection
Authorship identification

Age/gender identification
Language Identification
Sentiment analysis

...
Probabilistic Text Classification

Assigning subject categories, topics, or genres
Spam detection
Authorship identification

Age/gender identification
Language Identification
Sentiment analysis

...
Probabilistic Text Classification

Assigning subject categories, topics, or genres

Spam detection

Authorship identification

Age/gender identification

Language Identification

Sentiment analysis

MAP (maximum a posteriori classification)

\[ p(X \mid Y) = \frac{p(Y \mid X) \ast p(X)}{p(Y)} \]

- class
- observed data
- class-based likelihood (language model)
- prior probability of class
- observation likelihood (averaged over all classes)
Two Different Philosophical Frameworks

\[ p(X \mid Y) = \frac{p(Y \mid X) \cdot p(X)}{p(Y)} \]

- \( p(X \mid Y) \): posterior probability
- \( p(Y \mid X) \): likelihood
- \( p(X) \): prior probability
- \( p(Y) \): marginal likelihood

Posterior Classification/Decoding

maximum a posteriori

Noisy Channel Model Decoding

there are others too (CMSC 478/678)
Noisy Channel Model
what I want to tell you “sports”
Noisy Channel Model

what I want to tell you “sports”

what you actually see “The Os lost again...”
what I want to tell you "sports"

what you actually see "The Os lost again..."

Decode

hypothesized intent "sad stories" "sports"
Noisy Channel Model

what I want to tell you “sports”

what you actually see “The Os lost again…”

hypothesized intent “sad stories” “sports”

reweight according to what’s likely “sports”
Noisy Channel

Machine translation  
Part-of-speech tagging
Speech-to-text  
Morphological analysis
Spelling correction  
Image captioning
Text normalization

\[ p(X \mid Y) = \frac{p(Y \mid X) \ast p(X)}{p(Y)} \]

possible (clean) output
observed (noisy) text
translation/decode model
(clean) language model
observation (noisy) likelihood
Noisy Channel

Possible (clean) output
possible (noisy) text

\[ p(X \mid Y) = \frac{p(Y \mid X) \ast p(X)}{p(Y)} \]

Translation/decode model
(clean) language model

Observation (noisy) likelihood

Machine translation
Speech-to-text
Spelling correction
Text normalization

Part-of-speech tagging
Morphological analysis
Image captioning
Language Model

Use any of the language modeling algorithms we’ve learned

Unigram, bigram, trigram

Add-\(\lambda\), interpolation, backoff

(Later: Maxent, RNNs, hierarchical Bayesian LMs, ...)
Two Different Philosophical Frameworks

\[ p(X \mid Y) = \frac{p(Y \mid X) \cdot p(X)}{p(Y)} \]

- **posterior probability**
- **likelihood**
- **prior probability**
- **marginal likelihood**

Posterior Classification/Decoding

*maximum a posteriori*

Noisy Channel Model Decoding

*there are others too (CMSC 478/678)*
Classification with Bayes Rule

$$\text{argmax}_x p(X \mid Y)$$
Classification with Bayes Rule

\[
\arg\max_X \frac{p(Y \mid X) \ast p(X)}{p(Y)}
\]
Classification with Bayes Rule

\[
\text{argmax}_X \frac{p(Y \mid X) \ast p(X)}{p(Y)}
\]

constant with respect to X
Classification with Bayes Rule

$$\text{argmax}_x p(Y \mid X) \ast p(X)$$
MY HOBBY:
SITTING DOWN WITH GRAD STUDENTS AND Timing HOW LONG IT TAKES THEM TO FIGURE OUT THAT I'M NOT ACTUALLY AN EXPERT IN THEIR FIELD.
MY HOBBY:
SITTING DOWN WITH GRAD STUDENTS AND TIMING
HOW LONG IT TAKES THEM TO FIGURE OUT THAT
I'M NOT ACTUALLY AN EXPERT IN THEIR FIELD.

ENGINEERING:
OUR BIG PROBLEM IS HEAT DISSIPATION
HAVE YOU TRIED LOGARITHMS?
48 SECONDS

LINGUISTICS:
AH, SO DOES THIS FINNO-UGRIC FAMILY INCLUDE,
SAY, KLINGON?

LINGUISTICS:
63 SECONDS

SOCIOLGY:
YEAH, MY LATEST WORK IS ON RANKING PEOPLE
FROM BEST TO WORST.

4 MINUTES

LITERARY CRITICISM:
YOU SEE, THE DECONSTRUCTION IS INEXTRICABLE FROM NOT ONLY
THE TEXT, BUT ALSO THE SELF.

EIGHT PAPERS AND TWO BOOKS AND THEY HAVEN'T CAUGHT ON.
Classification with Bayes Rule

\[
\text{argmax}_X \log p(Y \mid X) + \log p(X)
\]
Evaluation: the 2-by-2 contingency table

<table>
<thead>
<tr>
<th></th>
<th>Actually Correct</th>
<th>Actually Incorrect</th>
</tr>
</thead>
<tbody>
<tr>
<td>Selected/Guessed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Not selected/not guessed</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Evaluation: the 2-by-2 contingency table

<table>
<thead>
<tr>
<th></th>
<th>Actually Correct</th>
<th>Actually Incorrect</th>
</tr>
</thead>
<tbody>
<tr>
<td>Selected/</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Guessed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Not selected/</td>
<td></td>
<td></td>
</tr>
<tr>
<td>not guessed</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Classes/Choices

[Diagram of classes/choices]
Evaluation: the 2-by-2 contingency table

<table>
<thead>
<tr>
<th></th>
<th>Actually Correct</th>
<th>Actually Incorrect</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Selected/Guessed</strong></td>
<td>True Positive  (TP)</td>
<td></td>
</tr>
<tr>
<td>Correct</td>
<td>[Blue Circle]</td>
<td>[Blue Circle]</td>
</tr>
<tr>
<td>Guessed</td>
<td>[Blue Circle]</td>
<td>[White Circle]</td>
</tr>
<tr>
<td><strong>Not selected/not guessed</strong></td>
<td>[Black Background]</td>
<td>[Black Background]</td>
</tr>
</tbody>
</table>
Evaluation: the 2-by-2 contingency table

<table>
<thead>
<tr>
<th>Selected/ guessed</th>
<th>Actually Correct</th>
<th>Actually Incorrect</th>
</tr>
</thead>
<tbody>
<tr>
<td>True Positive (TP)</td>
<td>Correct guessed</td>
<td>False Positive (FP)</td>
</tr>
<tr>
<td>Not selected/ not guessed</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Evaluation: the 2-by-2 contingency table

<table>
<thead>
<tr>
<th></th>
<th>Actually Correct</th>
<th>Actually Incorrect</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Selected/Guessed</strong></td>
<td>True Positive</td>
<td>False Positive</td>
</tr>
<tr>
<td></td>
<td>(TP) Correct</td>
<td>(FP) Correct</td>
</tr>
<tr>
<td></td>
<td>Guessed</td>
<td>Guessed</td>
</tr>
<tr>
<td><strong>Not selected/not guessed</strong></td>
<td>False Negative</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(FN) Correct</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Guessed</td>
<td></td>
</tr>
</tbody>
</table>
Evaluation: the 2-by-2 contingency table

<table>
<thead>
<tr>
<th>Selected/ Guessed</th>
<th>Actually Correct</th>
<th>Actually Incorrect</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>True Positive (TP)</td>
<td>False Positive (FP)</td>
</tr>
<tr>
<td>Correct</td>
<td>circle</td>
<td>circle</td>
</tr>
<tr>
<td>Guessed</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>False Negative (FN)</td>
<td>True Negative (TN)</td>
</tr>
<tr>
<td>Correct</td>
<td>circle</td>
<td>circle</td>
</tr>
<tr>
<td>Guessed</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Accuracy, Precision, and Recall

**Accuracy**: % of items correct

\[
\frac{TP + TN}{TP + FP + FN + TN}
\]

<table>
<thead>
<tr>
<th></th>
<th>Actually Correct</th>
<th>Actually Incorrect</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Selected/Guessed</strong></td>
<td>True Positive (TP)</td>
<td>False Positive (FP)</td>
</tr>
<tr>
<td><strong>Not select/not guessed</strong></td>
<td>False Negative (FN)</td>
<td>True Negative (TN)</td>
</tr>
</tbody>
</table>
Accuracy, Precision, and Recall

**Accuracy**: % of items correct
\[
\frac{TP + TN}{TP + FP + FN + TN}
\]

**Precision**: % of selected items that are correct
\[
\frac{TP}{TP + FP}
\]

<table>
<thead>
<tr>
<th>Selected/Guessed</th>
<th>Actually Correct</th>
<th>Actually Incorrect</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actually Correct</td>
<td>True Positive (TP)</td>
<td>False Positive (FP)</td>
</tr>
<tr>
<td>Not select/not guessed</td>
<td>False Negative (FN)</td>
<td>True Negative (TN)</td>
</tr>
</tbody>
</table>
Accuracy, Precision, and Recall

**Accuracy:** % of items correct

\[
\frac{TP + TN}{TP + FP + FN + TN}
\]

**Precision:** % of selected items that are correct

\[
\frac{TP}{TP + FP}
\]

**Recall:** % of correct items that are selected

\[
\frac{TP}{TP + FN}
\]

<table>
<thead>
<tr>
<th>Actually Correct</th>
<th>Actually Incorrect</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Selected/Guessed</strong></td>
<td>True Positive (TP)</td>
</tr>
<tr>
<td><strong>Not select/not guessed</strong></td>
<td>False Positive (FP)</td>
</tr>
<tr>
<td><strong>False Negative (FN)</strong></td>
<td>True Negative (TN)</td>
</tr>
</tbody>
</table>
A combined measure: \( F \)

Weighted (harmonic) average of **Precision & Recall**

\[
F = \frac{1}{\alpha \frac{1}{P} + (1 - \alpha) \frac{1}{R}}
\]
A combined measure: $F$

Weighted (harmonic) average of Precision & Recall

$$F = \frac{1}{\alpha \frac{1}{P} + (1 - \alpha) \frac{1}{R}} = \frac{(1 + \beta^2) \cdot P \cdot R}{(\beta^2 \cdot P) + R}$$

(Not important)
A combined measure: $F$

Weighted (harmonic) average of $P$recision & $R$ecall

$$F = \frac{(1 + \beta^2) \times P \times R}{(\beta^2 \times P) + R}$$

Balanced F1 measure: $\beta=1$

$$F_1 = \frac{2 \times P \times R}{P + R}$$
Micro- vs. Macro-Averaging

If we have more than one class, how do we combine multiple performance measures into one quantity?

**Macroaveraging**: Compute performance for each class, then average.

**Microaveraging**: Collect decisions for all classes, compute contingency table, evaluate.
**Micro- vs. Macro-Averaging: Example**

<table>
<thead>
<tr>
<th>Class 1</th>
<th>Class 2</th>
<th>Micro Ave. Table</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classifier: yes</td>
<td>Truth: yes</td>
<td>10</td>
</tr>
<tr>
<td>Classifier: no</td>
<td>Truth: yes</td>
<td>10</td>
</tr>
<tr>
<td>Classifier: yes</td>
<td>Truth: no</td>
<td>10</td>
</tr>
<tr>
<td>Classifier: no</td>
<td>Truth: no</td>
<td>10</td>
</tr>
</tbody>
</table>

Macroaveraged precision: \((0.5 + 0.9)/2 = 0.7\)

Microaveraged precision: \(100/120 = .83\)

Microaveraged score is dominated by score on common classes
Language Modeling as Naïve Bayes Classifier

\[ p(X | Y) = \frac{p(Y | X) \cdot p(X)}{p(Y)} \]

- **Posterior Classification/Decoding**
  - maximum a posteriori

- **Noisy Channel Model Decoding**
  - class-based likelihood (language model)
  - prior probability of class

- **Posterior probability**
- **Observed data**
- **Class**
I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun... It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet!
I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun... It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet!
The Bag of Words Representation

I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun... It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet!
Bag of Words Representation

\[ \gamma(\ldots) = C \]

<table>
<thead>
<tr>
<th>Word</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>seen</td>
<td>2</td>
</tr>
<tr>
<td>sweet</td>
<td>1</td>
</tr>
<tr>
<td>whimsical</td>
<td>1</td>
</tr>
<tr>
<td>recommend</td>
<td>1</td>
</tr>
<tr>
<td>happy</td>
<td>1</td>
</tr>
</tbody>
</table>

...
Language Modeling as Naïve Bayes Classifier

$$\text{argmax}_X p(Y \mid X) \ast p(X)$$

Start with Bayes Rule
Language Modeling as Naïve Bayes Classifier

$$\text{argmax}_X \prod_i p(Y_i | X) \ast p(X)$$

Adopt naïve bag of words representation $Y_i$
Language Modeling as Naïve Bayes Classifier

$$\arg\max_X \prod_{i} p(Y_i|X) \times p(X)$$

Adopt naïve bag of words representation $Y_i$

Assume position doesn’t matter
Language Modeling as Naïve Bayes Classifier

$$\arg\max_X \prod_i p(Y_i|X) \times p(X)$$

Adopt naïve bag of words representation $Y_i$

Assume position doesn’t matter

Assume the feature probabilities are independent given the class $X$
Multinomial Naïve Bayes: Learning

From training corpus, extract *Vocabulary*
Multinomial Naïve Bayes: Learning

From training corpus, extract *Vocabulary*

**Calculate** $P(c_j)$ **terms**
For each $c_j$ in $C$ do

$docs_j = \text{all docs with } \text{class } = c_j$

$$p(c_j) = \frac{|docs_j|}{\# \text{ docs}}$$
Multinomial Naïve Bayes: Learning

From training corpus, extract *Vocabulary*

**Calculate $P(c_j)$ terms**
For each $c_j$ in $C$ do
- $docs_j = $ all docs with class $= c_j$

**Calculate $P(w_k | c_j)$ terms**
For each word $w_k$ in *Vocabulary*
- $Text_j = $ single doc containing all $docs_j$
- $n_k = $ # of occurrences of $w_k$ in $Text_j$

\[ p(c_j) = \frac{|docs_j|}{\# \text{ docs}} \]

\[ p(w_k | c_j) = \text{class LM} \]
Naïve Bayes and Language Modeling

Naïve Bayes classifiers can use any sort of feature

But if, as in the previous slides

We use only word features

we use all of the words in the text (not a subset)

Then

Naïve Bayes has an important similarity to language modeling
Naïve Bayes as a Language Model

<table>
<thead>
<tr>
<th>Positive Model</th>
<th>Negative Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1 l</td>
<td>0.2 l</td>
</tr>
<tr>
<td>0.1 love</td>
<td>0.001 love</td>
</tr>
<tr>
<td>0.01 this</td>
<td>0.01 this</td>
</tr>
<tr>
<td>0.05 fun</td>
<td>0.005 fun</td>
</tr>
<tr>
<td>0.1 film</td>
<td>0.1 film</td>
</tr>
</tbody>
</table>
Naïve Bayes as a Language Model

Which class assigns the higher probability to $s$?

<table>
<thead>
<tr>
<th>Positive Model</th>
<th>Negative Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1 I</td>
<td>0.2 I</td>
</tr>
<tr>
<td>0.1 love</td>
<td>0.001 love</td>
</tr>
<tr>
<td>0.01 this</td>
<td>0.01 this</td>
</tr>
<tr>
<td>0.05 fun</td>
<td>0.005 fun</td>
</tr>
<tr>
<td>0.1 film</td>
<td>0.1 film</td>
</tr>
</tbody>
</table>
Naïve Bayes as a Language Model

Which class assigns the higher probability to $s$?

<table>
<thead>
<tr>
<th>Positive Model</th>
<th>Negative Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1 l</td>
<td>0.2 l</td>
</tr>
<tr>
<td>0.1 love</td>
<td>0.001 love</td>
</tr>
<tr>
<td>0.01 this</td>
<td>0.01 this</td>
</tr>
<tr>
<td>0.05 fun</td>
<td>0.005 fun</td>
</tr>
<tr>
<td>0.1 film</td>
<td>0.1 film</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>l</th>
<th>love</th>
<th>this</th>
<th>fun</th>
<th>film</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positive Model</td>
<td>0.1</td>
<td>0.1</td>
<td>0.01</td>
<td>0.05</td>
<td>0.1</td>
</tr>
<tr>
<td>Negative Model</td>
<td>0.2</td>
<td>0.001</td>
<td>0.01</td>
<td>0.005</td>
<td>0.1</td>
</tr>
</tbody>
</table>
Naïve Bayes as a Language Model

Which class assigns the higher probability to $s$?

<table>
<thead>
<tr>
<th>Positive Model</th>
<th>Negative Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1 l</td>
<td>0.2 l</td>
</tr>
<tr>
<td>0.1 love</td>
<td>0.001 love</td>
</tr>
<tr>
<td>0.01 this</td>
<td>0.01 this</td>
</tr>
<tr>
<td>0.05 fun</td>
<td>0.005 fun</td>
</tr>
<tr>
<td>0.1 film</td>
<td>0.1 film</td>
</tr>
</tbody>
</table>

$5e-7 \approx P(s|\text{pos}) > P(s|\text{neg}) \approx 1e-9$
Brill and Banko (2001)

With enough data, the classifier may not matter
Summary: Naïve Bayes is Not So Naïve

Very Fast, low storage requirements

Robust to Irrelevant Features

Very good in domains with many equally important features

Optimal if the independence assumptions hold

Dependable baseline for text classification (but often not the best)
But: Naïve Bayes Isn’t Without Issue

Model the posterior in one go?

Are the features really uncorrelated?

Are plain counts always appropriate?

Are there “better” ways of handling missing/noisy data? (automated, more principled)