**Text Categorization**

*Given a new document, which of several classes is it?*

**Many Applications**

- Topic of a news article (classic example)
- Sentiment of a movie or product review
- Email into (spam, not), or (business, personal, bills, ...)
- Reading level (K-12) of an article or essay
- Author of a document (e.g., Federalist papers)
- Genre of a document (report, editorial, advertisement, blog)
- Language identification
Some Notation

• $x_1, ..., x_N$: our training documents
• $y_1, ..., y_N$: their classes (for now, assume only two values)

• What we want: a classifier $c$ that takes a new document, e.g., "$x_{N+1}$" and returns its class.
  • $c(x) \rightarrow y$

Linear Classifiers

• Map each document into a $d$-dimensional vector
  $f(x) \in \mathbb{R}^d$
Linear Classifiers

- Learn a weight vector $w \in \mathbb{R}^d$ which defines a separating hyperplane

$$f(x)^Tw = 0$$

Linear Classifiers

- If $f(x)^Tw < 0$, predict class 1
- If $f(x)^Tw > 0$, predict class 2

$f(x)$

- Maps a document into a numerical vector.
- A collection of “feature functions.”
- In principle, can be any measurement on the document.
- Common feature functions:
  - Number of occurrences of word $u$
  - Is word $u$ present? (0 or 1; binary feature)
  - Number of occurrences of bigram $uv$
  - 1 (bias)
  - ...
Hyperplanes and Separability

There may be more than one good solution!

Hyperplanes and Separability

In some cases, there's no solution!
How to fix this problem?
Hyperplanes and Separability

In some cases, there's no solution!
How to fix this problem?

Semantics of “0”

When some dimensions of $w$ are 0, we are effectively disregarding some features. We don’t need the $z$ dimension here!

Semantics of “0”

Same classifier, with the “vertical” feature explicitly removed ...
Semantics of “0”

Same classifier, with the “vertical” feature explicitly removed ... and another feature removed!

High-Level Thinking

• If we can come up with enough good features, we believe we can find a hyperplane (i.e., weights \( w \)) that separates the training data.
  • How?

Perceptron

• A very simple algorithm guaranteed to eventually find a linear separator, if one exists.
• If one doesn’t, the perceptron will oscillate.

• The algorithm:
  
  \[
  w^{(0)} \leftarrow \{0, \ldots, 0\} \\
  \text{for } t = 1 \text{ to } T:\n  \quad \text{for } n = 1 \text{ to } N:\n  \quad \quad a \leftarrow y_n - c(x_n; w) \\
  \quad \quad w \leftarrow w + af(x_n) \\
  \text{return } w
  \]
Extensions

• Because the data may not be separable, people sometimes use:
  • averaged perceptron: \( w \leftarrow \text{average}(w(1), ..., w(T)) \)
  • voted perceptron: \( c(x) = \text{mode}(c(1)(x), ..., c(T)(x)) \)

Modern Methods

• Maximum margin techniques try to find a separating hyperplane that's far from the training examples.
• Kernel methods use tricks to get much higher-dimensional spaces, almost for free.
• Feature selection tries to find the most important features and throw out the rest.

• Take a class in machine learning to learn more!

What if there are more than 2 classes?

My favorite solution:
• Change \( f_1, ..., f_d \) to be functions of both \( x \) and \( y \)
  \[ f_{\text{test}}(x, y) = \text{count of "diploma" in } x \text{ if } y = \text{"spam"}, 0 \text{ otherwise} \]
• The decision function then becomes:
  \[ c(x) = \arg\max_y f(x, y) \mathbf{w} \]

Another way to do it: a bunch of "one vs. all" classifiers.
Alternative Approach

Noisy Channel

\[ c(x) = \arg \max_y p(y \mid x) = \arg \max_y p(y) p(x \mid y) \]

Source and Channel

- **Source**: prior probability of each class
- **Channel**: generates the documents
  - Could be a language model!
  - Could be something else!
  - Important special case: Naive Bayes model
Naive Bayes

\[ c(x) = \arg \max_y p(y) \times p(x \mid y) \]

\[ p(x \mid y) = p(f(x) \mid y) = \prod_{i=1}^{d} p(f_i(x) \mid y) \]

Feature values are all independent of each other, given the class.

Naive Bayes

- How to estimate \( p(y) \)?
- How to estimate \( p(f(x) \mid y) \) for different features \( f \)?

Noisy Channel with Language Models

- Source: randomly pick a language (this models the fact that some “languages”/“classes” are more likely than others)
- Channel: generate text from the language model.

- To “win” an \( x \), a class \( y \) has to be likely (source) and likely to produce the document \( x \) (channel)!
Modern Methods

• Assuming independence among features (given the class) is, indeed, naive
• Instead, model interactions among features: max ent (or exponential) models.
• Structured classification: different labels are interdependent! (E.g., two documents from the same web site might be likely to have the same topic.)

Best Practices

• Training/development/testing
• Evaluation: accuracy, precision, and recall
  • Per-class precision and recall
• Lower bound: most probable class? Random?
• Upper bound: human agreement