Naive Bayes Classifier

Background - Discriminative vs Generative

- We’ve mostly discussed “discriminative” models so far, which predict P(Y|X)
- Today we’ll look at a “generative” model, which predicts P(X|Y) and P(Y)

Example Problem

- Goal: predict whether a borrower will default on loan
- Data: 50 observations, 100 features
    - Features: 50 Bernoulli, 50 Gaussian (e.g. past default, normalized credit score, etc.)
- Problem: Not enough data to estimate joint distribution directly

Naive Bayes Derivation

Naive Bayes for Text Classification

- Author randomly picks a category (e.g. fiction, nonfiction)
  - according to prior distribution $P(Y)$
- Then randomly draws from bag of words with replacement
  - according conditional distribution $P(x|y)$
- Known as the “multinomial event model”

Naive Bayes Text Classifier Derivation

Estimating class prior distribution:

$$P(y = \text{"}sports\text{"}) = \frac{\text{number of sports articles}}{\text{total number of articles}}$$

Naive Bayes Text Classifier Derivation

Estimating conditional word distribution from bag of words:

Fiction Corpus:

"the cat in the hat"

"the cat in the tree"

"the cow jumped over the moon"

P(word = "cat" | fiction) = 2/15

P(word = "jumped" | fiction) = 1/15

Naive Bayes Text Classifier Derivation

Estimating conditional word distribution from bag of words:

<u>Nonfiction Corpus:</u>

“the giants won the game”

“the stock market was up today”

“the candidate won the election”

P(word = “giants” | nonfiction) = 1/15

P(word = “won” | nonfiction) = 2/15

Naive Bayes Text Classifier Derivation

$$P(y|doc = \text{"the cat in the hat"}) =$$

$$= \frac{P(doc = \text{"the cat in the hat"}|y)P(y)}{P(doc = \text{"the cat in the hat"})} \propto$$

$$= P(doc = \text{"the cat in the hat"}|y)P(y)$$

Naive Bayes Text Classifier Derivation

$$P(doc = \text{"the cat in the hat"}|y)P(y) =$$

$$P(y)P(\text{"}the\text{"}|y)P(\text{"}cat\text{"}|y)P(\text{"}in\text{"}|y)P(\text{"}the\text{"}|y)P(\text{"}hat\text{"}|y) =$$

$$P(y)P(\text{"}the\text{"}|y)^2 P(\text{"}cat\text{"}|y)^1 P(\text{"}in\text{"}|y)^1 P(\text{"}hat\text{"}|y)^1 =$$

$$P(y) \prod_{w \in vocab} P(w|y)^{x_w} =$$

Naive Bayes Text Classifier Derivation

$$P(y) \prod_{w \in vocab} P(w|y)^{x_w} =$$

$$log(P(y)) + \sum_{w \in vocab} x_w log(P(w|y)) =$$

$$\Rightarrow \widehat{y} = argmax_y \left(log(P(y)) + \sum_{w \in vocab} x_w log(P(w|y)) = \right)$$

Laplace Smoothing

$$P(y) \prod_{w \in vocab} P(w|y)^{x_w}$$

What happens if a word doesn't appear in a class?

Laplace Smoothing

$$P(x|c) = \frac{(\#\text{ of times } x \text{ appears in articles of class } c) + \alpha}{(\text{total } \# \text{ of words in articles of class } c) + \alpha \cdot (\# \text{ of words in corpus})}$$

- add constant (usually 1) to each word’s frequency
- as if we saw each word more than we actually did
- prevents zero division

Details

Pros

- Good with "wide data"
  (i.e. more features than observations)
- Fast to train / good at online learning
- Simple to implement

Cons

- Can be hampered by irrelevant features
- Sometimes outperformed by other models

Details: "Tackling the Poor Assumptions of Naive Bayes Classifiers" http://machinelearning.wustl.edu/mlpapers/paper_files/icml2003_RennieSTK03.pdf

Variants of Naive Bayes

- Feature weighting (source)
- Use other distributions to model term frequency (source)