Naive Bayes Classifier

Why Naive Bayes

- n << p (# of features)
- n somewhat small **or**
- n quite large:
  - Streams of input data (online learning)
  - Not bounded by memory (usually)
- Multi-class

Background - Discriminative vs Generative

- We’ve mostly discussed “discriminative” models so far, which predict $P(Y|X)$ directly.

- Today we’ll look at a “generative” model, which predicts $P(X|Y)$ and $P(Y)$

Naive Bayes Derivation

Naive Bayes

Bayes Theorem: $$p(C_k|\mathbf{x}) = \frac{p(C_k)\ p(\mathbf{x}|C_k)}{p(\mathbf{x})}.$$

Assuming independence:

$$\boxed{p(C_k|x_1,\dots,x_n) = \frac{1}{Z}p(C_k)\prod_{i=1}^{n} p(x_i|C_k)}$$

where:

$$Z = p(\mathbf{x})$$

Naive Bayes Classification

$$\hat{y} = \underset{k \in \{1,\dots,K\}}{\operatorname{argmax}} \; p(C_k) \prod_{i=1}^{n} p(x_i | C_k).$$

Can ignore 1/Z term when predicting classes.

Naive Bayes with Text Data

$$P(c) = \frac{\text{\# of articles of class } c}{\text{total \# of articles}}$$

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

Naive Bayes in Practice

Dealing with zeros: Laplace smoothing

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

Avoiding underflow: Log Likelihood

$$\log(P(c|X)) = \log(P(c)) + \log(P(x_1|c)) + \log(P(x_2|c)) + \ldots + \log(P(x_n|c))$$

Special Cases of Naive Bayes

- Multinomial: estimate $P(x|c)$ directly via counts. (introduced above)
- Gaussian: model $P(x|c)$ assuming that the data are Normally (aka Gaussian) distributed.
- Bernoulli: used with strictly binary input data.
  - Text data: use word occurrence instead of counts

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

Notation used in Sprint

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

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

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

Naive Bayes Text Classifier Derivation

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

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

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

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

Variants of Naive Bayes

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

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:

Nonfiction Corpus:

“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

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

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

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

Laplace Smoothing

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

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