Generative Models for Classification

CS6780 – Advanced Machine Learning
Spring 2015

Thorsten Joachims
Cornell University

Reading:
Murphy 3.5, 4.1, 4.2, 8.6.1

Generative vs. Conditional vs. ERM

- Empirical Risk Minimization
  - Find $h = \operatorname*{argmin}_{h \in H} Err_S(h)$ s.t. overfitting control
  - Pro: directly estimate decision rule
  - Con: committed to loss, X, Y
- Discriminative Conditional Model
  - Find P(Y|X), then derive h(x) via Bayes rule
  - Pro: not committed to loss
  - Con: committed to X, Y; conditional distributions more complex than decision rule
- Generative Model
  - Find P(X,Y), then derive h(x) via Bayes rule
  - Pro: not committed to loss function, X, and Y; often computationally easy
  - Con: Model dependencies in X

Bayes Decision Rule

- Assumption:
  - learning task P(X,Y)=P(Y|X) P(X) is known
- Question:
  - Given instance x, how should it be classified to minimize prediction error?
- Bayes Decision Rule:

$$h_{bayes(\vec{x})} = argmax_{y \in Y}[P(Y = y|X = \vec{x})]$$
$$= argmax_{y \in Y}[P(X = \vec{x}|Y = y)P(Y = y)]$$

Bayes Theorem

- It is possible to "switch" conditioning according to the following rule
- Given any two random variables X and Y, it holds that

$$P(Y = y|X = x) = \frac{P(X = x|Y = y)P(Y = y)}{P(X = x)}$$

- Note that

$$P(X = x) = \sum_{y \in Y} P(X = x|Y = y)P(Y = y)$$

Naïve Bayes' Classifier (Multivariate)

- Model for each class

$$P(X = \vec{x}|Y = +1) = \prod_{i=1}^{N} P(X_i = x_i|Y = +1)$$

$$P(X = \vec{x}|Y = -1) = \prod_{i=1}^{N} P(X_i = x_i|Y = -1)$$

- Prior probabilities

$$P(Y = +1), P(Y = -1)$$

- Classification rule:

$$h_{naive}(\vec{x}) = \underset{y\in\{+1,-1\}}{\operatorname{argmax}} \left\{ P(Y = y) \prod_{i=1}^{N} P(X_i = x_i|Y = y) \right\}$$

fever (h,l,n)	cough (y,n)	pukes (y,n)	flu?
high	yes	no	1
high	no	yes	1
low	yes	no	-1
low	yes	yes	1
high	no	yes	???

Estimating the Parameters of NB

fever (h,l,n)	cough (y,n)	pukes (y,n)	flu?
high	yes	no	1
high	no	yes	1
low	yes	no	-1
low	yes	yes	1
high	no	yes	???

- Count frequencies in training data
  - n: number of training examples
  - n_+ / n_- : number of pos/neg examples
  - $\#(X_i=x_i, y)$: number of times feature X_i takes value x_i for examples in class y
  - $|X_i|$: number of values attribute X_i can take
- Estimating P(Y)
  - Fraction of positive / negative examples in training data

$$\hat{P}(\mathrm{Y}=+1)=\frac{n_+}{n} \qquad \hat{P}(\mathrm{Y}=-1)=\frac{n_-}{n}$$

- Estimating P(X|Y)
  - Maximum Likelihood Estimate

$$\hat{P}(X_i=x_i|Y=y)=\frac{\#(X_i=x_i,y)}{n_y}$$

  - Smoothing with Laplace estimate

$$\hat{P}(X_i=x_i|Y=y)=\frac{\#(X_i=x_i,y)+1}{n_y+|X_i|}$$

Linear Discriminant Analysis

- Spherical Gaussian model with unit variance for each class

$$P(X = \vec{x} | Y = +1) \sim \exp\left(-\frac{1}{2}(\vec{x} - \vec{\mu}_{+})^2\right)$$

$$P(X = \vec{x} | Y = -1) \sim \exp\left(-\frac{1}{2}(\vec{x} - \vec{\mu}_{-})^2\right)$$

- Prior probabilities

$$P(Y = +1), P(Y = -1)$$

- Classification rule

$$h_{LDA}(\vec{x}) \quad = \operatorname*{argmax}_{y \in \{+1,-1\}} \left\{ P(Y = y) exp\left(-\frac{1}{2}\left(\vec{x} - \vec{\mu}_y\right)^2\right)\right\}$$

$$\operatorname*{argmax}_{y \in \{+1,-1\}} \left\{ \log(P(Y = y)) - \frac{1}{2}\left(\vec{x} - \vec{\mu}_y\right)^2 \right\}$$

- Often called “Rocchio Algorithm” in Information Retrieval

Estimating the Parameters of LDA

- Count frequencies in training data
  - $(\vec{x}_1, \vec{y}_1), \dots, (\vec{x}_n, \vec{y}_n) \sim P(X, Y)$: training data
  - *n*: number of training examples
  - n_+ / n_-: number of positive/negative training examples
- Estimating P(Y)
  - Fraction of pos / neg examples in training data

$$\hat{P}(\mathrm{Y} = +1) = \frac{n_+}{n} \qquad \hat{P}(\mathrm{Y} = -1) = \frac{n_-}{n}$$

- Estimating class means

$$\vec{\mu}_+ = \frac{1}{n_+} \sum_{\{i: y_i = 1\}} \vec{x}_i \qquad \vec{\mu}_- = \frac{1}{n_-} \sum_{\{i: y_i = -1\}} \vec{x}_i$$

Naïve Bayes Classifier (Multinomial)

- Application: Text classification ($x = (w_1, \ldots, w_l)$ sequence)

text	CS?
$x_1 = (The, art, of, Programming)$	+1
$x_2 = (Introduction, to, Calculus)$	-1
$x_3 = (Introduction, to, Complexity, Theory)$	+1
$x_4 = (Introduction, to, Programming)$	??

- Assumption

$$P(X = x | Y = +1) = \prod_{i=1}^{l} P(W = w_i | Y = +1)$$

$$P(X = x | Y = -1) = \prod_{i=1}^{l} P(W = w_i | Y = -1)$$

- Classification Rule

$$h_{naive}(x) = \underset{y \in \{+1,-1\}}{\operatorname{argmax}} \left\{ P(Y = y) \prod_{i=1}^{l} P(W = w_i | Y = y) \right\}$$

Estimating the Parameters of Multinomial Naïve Bayes

- Count frequencies in training data
  - *n*: number of training examples
  - n_+ / n_- : number of pos/neg examples
  - *#(W=w, y)*: number of times word *w* occurs in examples of class *y*
  - l_+ / l_- : total number of words in pos/neg examples
  - | V |: size of vocabulary

text	CS?
$x_1 = (The, art, of, Programming)$	+1
$x_2 = (Introduction, to, Calculus)$	-1
$x_3 = (Introduction, to, Complexity, Theory)$	+1
$x_4 = (Introduction, to, Programming)$	??

- Estimating P(Y)

$$\hat{P}(\mathrm{Y} = +1) = \frac{n_+}{n} \qquad \hat{P}(\mathrm{Y} = -1) = \frac{n_-}{n}$$

- Estimating P(X|Y) (smoothing with Laplace estimate):

$$\hat{P}(W = w|Y = y) = \frac{\#(W = w, y) + 1}{l_y + |V|}$$