

Generative Models for Classification

CS4780/5780 – Machine Learning
Fall 2019

Nika Haghtalab & Thorsten Joachims
Cornell University

Reading: UML 24.2, 24.3

Generative vs. Conditional vs. ERM

- Empirical Risk Minimization
 - Find $h = \operatorname{argmin}_{h \in H} \text{Err}_S(h)$ s.t. overfitting control
 - Pro: directly estimate decision rule
 - Con: need to commit to loss, input, and output before training
- Discriminative Conditional Model
 - Find $P(Y|X)$, then derive $h(x)$ via Bayes rule
 - Pro: not yet committed to loss during training
 - Con: need to commit to input and output before training; learning conditional distribution is harder than learning decision rule
- Generative Model
 - Find $P(X,Y)$, then derive $h(x)$ via Bayes rule
 - Pro: not yet committed to loss, input, or output during training; often computationally easy
 - Con: Needs to 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 (for zero/one loss):

$$h_{\text{bayes}}(\vec{x}) = \operatorname{argmax}_{y \in Y} [P(Y = y | X = \vec{x})]$$

Example: Modeling Flu Patients

- Data:

fever (h, 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	???
- Approach: One model for flu, one for not-flu.

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:

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

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

fever (h,j,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 $P(Y)$
 - Fraction of positive / negative examples in training data

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

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

$$\hat{P}(X_i = x_j | Y = y) = \frac{\#(X_i = x_j, y)}{n_y}$$
 - Smoothing with Laplace estimate

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

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

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

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

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

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

- Estimating class means

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

Naïve Bayes Classifier (Multinomial)

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

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

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 $P(Y)$
 - Fraction of positive / negative examples in training data

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

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

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