Generative Models for Supervised Learning,

CS772A: Probabilistic Machine Learning

Piyush Rai

Generative Supervised Learning

- The conditional distribution $p(y|x)$ can also be defined as

$$p(y|x) = \frac{p(x,y)}{p(x)}$$

Requires modeling the joint distribution of the inputs and outputs

In the discriminative approach for learning $p(y|x)$, we didn't model the inputs x but treated them as "given"

- Generative sup. learning is usually more work because $p(x,y)$ has to be estimated
- However, there are some benefits as well. For example, for classification

$p(y)$ is called the "class-prior" or "class-marginal" distribution

Can incorporate knowledge of frequency ("size") of each class in training data

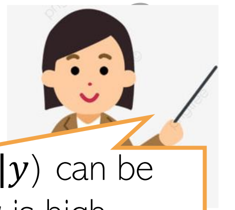
Can incorporate knowledge of the distribution ("shape") of each class in training data

$$p(y|x) = \frac{p(x,y)}{p(x)} = \frac{p(y)p(x|y)}{p(x)}$$

Can assume simple/sophisticated types of distributions for the "class-conditional" distribution $p(x|y)$ and learned them using the training data of each class

- Once class-marginal and class-conditionals are estimated, we can make predictions for the label of test input as $p(y_* = k|x_*) \propto p(y_* = k)p(x_*|y_* = k)$

Generative Supervised Learning



- The generative classification model

Probability of belonging to class $\boldsymbol{k}$, conditioned on the input $\boldsymbol{x}$

Marginal probability of belonging to class $\boldsymbol{k}$

Probability (density) of input $\boldsymbol{x}$ under class $\boldsymbol{k}$

$$p(y = k|\boldsymbol{x}) = \frac{p(y = k)p(\boldsymbol{x}|y = k)}{\sum_k p(y = k)p(\boldsymbol{x}|y = k)}$$

Note: Estimating $\boldsymbol{p(x|y)}$ can be difficult especially if $\boldsymbol{x}$ is high-dimensional and we don't have enough data from each class

A way to handle this is to assume simpler forms for $p(\boldsymbol{x}|y)$ (e.g., Gaussian with diagonal/spherical covar – naïve Bayes) but it might sacrifice accuracy too

- We need to learn $\boldsymbol{p(y)}$ and $\boldsymbol{p(x|y)}$ here given training data $(\boldsymbol{X}, \boldsymbol{y}) = \{(x_n, y_n)\}_{n=1}^N$

- Class prior/marginal distribution $\boldsymbol{p(y)}$ will always be a discrete distribution, e.g.,
  - For $y \in \{0,1\}$, $p(y) = p(y|\pi) = \text{Bernoulli}(y|\pi)$ with $\boldsymbol{\pi} \in (0,1)$
  - For $y \in \{1,2,\ldots,K\}$, $p(y) = p(y|\boldsymbol{\pi}) = \text{multinoulli}(y|\boldsymbol{\pi})$ where $\boldsymbol{\pi} = [\pi_1, \ldots, \pi_K]$

$\sum_{k=1}^{K} \pi_k = 1$

- Class conditional distribution $\boldsymbol{p(x|y)}$ will depend on the nature of inputs, e.g.,
  - For $\boldsymbol{x} \in \mathbb{R}^D$, $p(\boldsymbol{x}|y = k)$ can be a multivariate Gaussian (one per class)

For π, can use Beta or Dirichlet (we have already seen these examples)

Note: When estimating θ_k, we only need inputs from class k
$\boldsymbol{X}_k = \{\boldsymbol{x}_n : y_n = k\}$

$$p(\boldsymbol{x}|y = k) = p(\boldsymbol{x}|\theta_k) = \mathcal{N}(\boldsymbol{x}|\mu_k, \Sigma_k)$$

Will need appropriate prior distributions for π and $\{\theta_k\}_{k=1}^K$

- Can estimate π and $\{\theta_k\}_{k=1}^K$ using $(\boldsymbol{X}, \boldsymbol{y})$ via point est. or fully Bayesian infer.

Generative Classification: Making Predictions

- Once $\boldsymbol{\pi}$ and $\{\theta_k\}_{k=1}^{K}$ are learned, we are ready to make prediction for any test input $\boldsymbol{x}_*$
- Two ways to make the prediction
- Approach 1: If we have point estimates for $\boldsymbol{\pi}$ and $\{\theta_k\}_{k=1}^{K}$, say $\hat{\boldsymbol{\pi}}$ and $\{\hat{\theta}_k\}_{k=1}^{K}$. Then

$$p(y_* = k|\boldsymbol{x}_*) = \frac{p(y_* = k|\hat{\pi})p(\boldsymbol{x}_*|\hat{\theta}_k)}{\sum_k p(y = k|\hat{\pi})p(\boldsymbol{x}|\hat{\theta}_k)} \propto \hat{\pi}_k p(\boldsymbol{x}_*|\hat{\theta}_k)$$

Compute for every value of k and normalize

- Approach 2: If we have the full posterior for $\boldsymbol{\pi}$ and $\{\theta_k\}_{k=1}^{K}$. Then

PPD of y_*

  - Instead of using $p(y_* = k|\hat{\boldsymbol{\pi}})$, we will use $p(y_* = k|\boldsymbol{y}) = \int p(y_* = k|\pi)p(\pi|\boldsymbol{y})d\pi$
  - Instead of using $p(\boldsymbol{x}_*|\hat{\theta}_k)$, we will use $p(\boldsymbol{x}_*|\boldsymbol{X}_k) = \int p(\boldsymbol{x}_*|\theta_k)p(\theta_k|\boldsymbol{X}_k)d\theta_k$ PPD of $\boldsymbol{x}_*$
  - Using these quantities, the prediction will be made as

Compute for every value of k and normalize

$$p(y_* = k|x_*, \boldsymbol{X}, \boldsymbol{y}) = \frac{p(y_* = k|\boldsymbol{y})p(\boldsymbol{x}_*|\boldsymbol{X}_k)}{\sum_k p(y_* = k|\boldsymbol{y})p(\boldsymbol{x}_*|\boldsymbol{X}_k)} \propto p(y_* = k|\boldsymbol{y})p(\boldsymbol{x}_*|\boldsymbol{X}_k)$$

Note that we aren't using a single "best" value of the params π and θ_k unlike Approach 1

Generative Classification: A Generative Story

- Assuming binary labels, can define a "generative story" for each example $(\boldsymbol{x}_i, y_i)$
  - First draw ("generate") a binary label $y_i \in \{0,1\}$

$$y_i \sim \text{Bernoulli}(\pi)$$

For multi-class problems, we will have a multinoulli instead

  - Now draw ("generate") the input $\boldsymbol{x}_i$ from the distribution of class $y_i \in \{0,1\}$

$$\boldsymbol{x}_i | y_i \sim p(\boldsymbol{x}|\theta_{y_i})$$

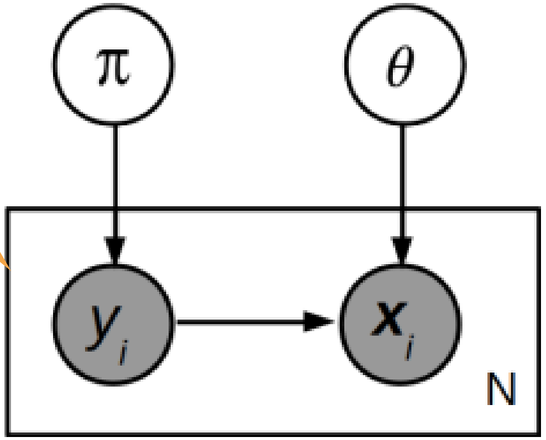
Most generative models (supervised as well as unsupervised/semi-supervised) can be expressed via such a story

- Writing $\boldsymbol{\theta} = (\theta_0, \theta_1)$, the above generative model shown in "plate notation"

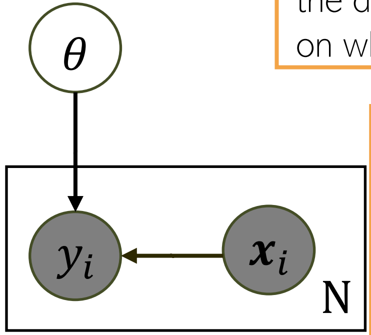
Order of generation in this story depends on what part of the data/parameters depend on what data/params

Note that in this generative process, we assume y is generated first since the generation of $\boldsymbol{x}$ depends on what $\boldsymbol{y}$ is

Note that in this generative process, we assume y is generated first since the generation of $\boldsymbol{x}$ depends on what $\boldsymbol{y}$ is



Compare this with the plate notation diagram of a discriminative model such as prob linear regression or logistic regression



A discriminative model (no model for $\boldsymbol{x}_i$'s)

Often we also show the generation of parameters/unknowns as well (via their respective distributions)

Generative Sup. Learning: Some Comments

- A very flexible approach for classification

Incorporate info about how frequent each class is in the training data ("class prior")

Incorporate info about the shape of each class

Consequently, can naturally learn nonlinear boundaries, too (without using kernel methods or deep learning)

$$p(y_* = k|\boldsymbol{x}_*) = \frac{p(y_* = k)p(\boldsymbol{x}_*|y_* = k)}{\sum_k p(y_* = k)p(\boldsymbol{x}_*|y_* = k)}$$

- Can handle missing labels and missing features
  - These can be treated as latent variables as estimated using methods such as EM

Will discuss this later

- Ability to handle missing labels makes it suitable for semi-supervised learning
- The choice of the class-conditional and proper estimation is important
  - Can leverage advances in deep generative models to learn very flexible forms for $\boldsymbol{p}(\boldsymbol{x}|\boldsymbol{y})$
- Can also use it for regression (define $\boldsymbol{p}(\boldsymbol{x}, \boldsymbol{y})$ via some distr. and obtain $\boldsymbol{p}(\boldsymbol{y}|\boldsymbol{x})$ from it)
- Can also <u>combine</u> generative and discriminative approaches for supervised learning

Hybrids of Discriminative and Generative Models

- Both discriminative and generative models have their strengths/shortcomings
- Some aspects about discriminative models for sup. learning

Recall prob linear regression and logistic reg

  - Discriminative models have usually fewer parameters (e.g., just a weight vector)
  - Given "plenty" of training data, disc. models can usually outperform generative models
- Some aspects about generative models for sup. learning
  - Can be more flexible (we have seen the reasons already)
  - Usually have more parameters to be learned
  - Modeling the inputs (learning $\boldsymbol{p}(\boldsymbol{x}|\boldsymbol{y})$) can be difficult for high-dim inputs
- Some prior work on combining discriminative and generative models. Examples:

$$\alpha \log p(y|x;\theta) + \beta \log p(x;\theta)$$

$$p(x, y, \theta_d, \theta_g) = p_{\theta_d}(y|x) p_{\theta_g}(x) p(\theta_d, \theta_g)$$

$$p(x, y, z) = p(y|x, z) \cdot p(x, z)$$

Approach 1 (McCullum et al, 2006) – modeling the joint $p(x, y|\theta)$ using a multi-conditional likelihood

Approach 2 (Lasserre et al, 2006) – Coupled parameters between discriminative and generative models

Approach 3 (Kuleshov and Ermon, 2017) – Coupling discriminative and generative models via a latent variable z (see "Deep Hybrid Models: Bridging Discriminative and Generative Approaches", UAI 2017)