Generative Supervised Learning

CS772A: Probabilistic Machine Learning

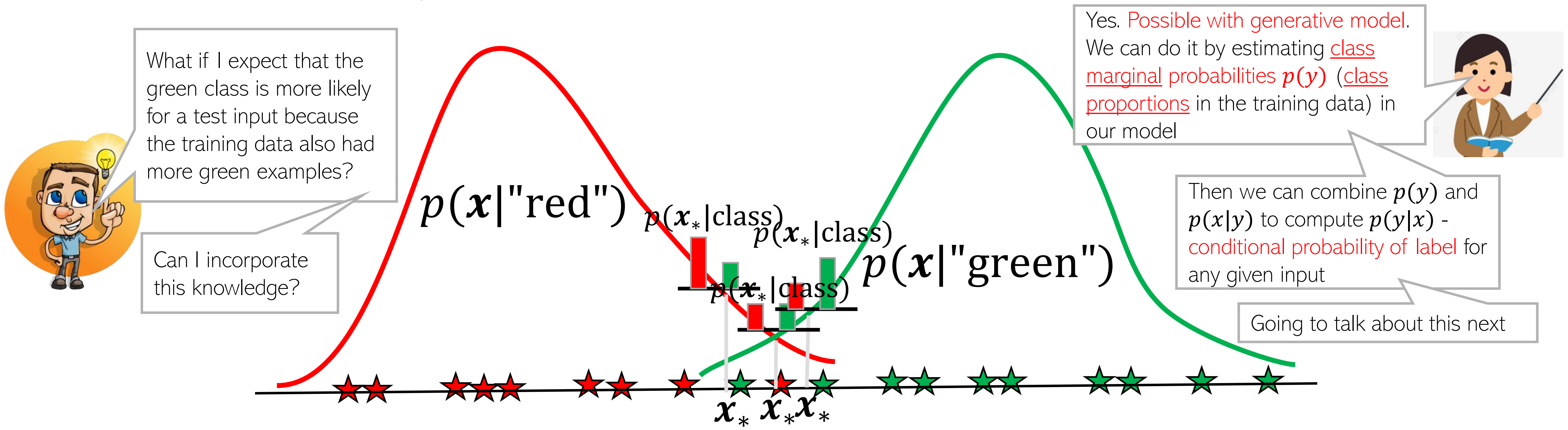
Piyush Rai

Announcement

- Students facing issues with marking biometric attendance: Please visit Biometrics office (located below L-16) to get the issue fixed.

Generative Classification: A Basic Idea

- Learn the probability distribution $p(x|y=k)$ of inputs from each class k


What if I expect that the green class is more likely for a test input because the training data also had more green examples?
Can I incorporate this knowledge?
Yes. Possible with generative model. We can do it by estimating class marginal probabilities $p(y)$ (class proportions in the training data) in our model
Then we can combine $p(y)$ and $p(x|y)$ to compute $p(y|x)$ - conditional probability of label for any given input
Going to talk about this next
$p(\boldsymbol{x}|\text{"red"})$
$p(\boldsymbol{x}|\text{"green"})$
$p(\boldsymbol{x}_*|\text{class})$
$p(\boldsymbol{x}_*|\text{class})$
$p(\boldsymbol{x}_*|\text{class})$
$\boldsymbol{x}_*$
$\boldsymbol{x}_*$
$\boldsymbol{x}_*$

- We usually assume some form for $p(x|y=k)$ (e.g., Gaussian) and estimate the parameters of that distribution (MLE/MAP/fully posterior)
- We then predict label of test input $\boldsymbol{x}_*$ by comparing probabilities under each class
  - Or can report the probability of belonging to each class (soft prediction)

Generative Classification

- Suppose we have training data $\{(\boldsymbol{x}_n, y_n)\}_{n=1}^{N}$ from K classes
- The conditional probability of label y_n given the input $\boldsymbol{x}_n$

$$p(y_n = k|x_n) = \frac{p(\boldsymbol{x}_n, y_n = k)}{p(x_n)}$$

Known as the "class-conditional" distribution

Probability distribution of the inputs from class $\boldsymbol{k}$

Known as "class-marginal" or "class-prior" distribution

Marginal distribution of just the labels (not looking at the inputs) – Bernoulli/multinoulli

$$= \frac{p(y_n = k) \times p(\boldsymbol{x}_n|y_n = k)}{p(\boldsymbol{x}_n)}$$

The numerator (joint distribution of x_n and y_n) summed over all K values of y_n

Marginal distribution of the input $\boldsymbol{x}_n$

- We use the training data to estimate the class-marginal and class-conditionals

Estimating Class Marginals

- Estimating class marginals $p(y = k)$ is usually straightforward

If only two classes, assume Bernoulli

- Since labels are discrete, we assume class marginal $p(y)$ to be a multinoulli

$\pi_k = p(y = k)$

These probabilities sum to 1: $\sum_{k=1}^{K} \pi_k = 1$

$$p(y|\boldsymbol{\pi}) = \text{multinoulli}(y|\pi_1, \pi_2, \ldots, \pi_K) = \prod_{k=1}^{K} \pi_k^{\mathbb{I}[y=k]}$$

- Given N i.i.d. labelled examples $\{(x_n, y_n)\}_{n=1}^{N}$, $y_n \in \{1,2,\ldots,K\}$ the MLE soln

$$\boldsymbol{\pi}_{MLE} = \underset{\boldsymbol{\pi}}{\text{argmax}} \sum_{n=1}^{N} \log p(y_n|\boldsymbol{\pi})$$

Subject to constraint $\sum_{k=1}^{K} \pi_k = 1$

- MLE solution is $p(y = k) = \pi_k = N_k/N$ where $N_k = \sum_{n=1}^{N} \mathbb{I}[y = k]$
  - Thus $p(y = k) = \pi_k$ is simply the fraction of inputs from class k
- Can also compute MAP estimate or full posterior of $\boldsymbol{\pi}$ using a Dirichlet prior

Estimating Class-Conditionals

To be estimated using the N_k training inputs $\{\boldsymbol{x}_n : y_n = k\}$ from class k

- Can assume a distribution $p(\boldsymbol{x}|y = k) = p(\boldsymbol{x}|\theta_k)$ for inputs of each class k
- If $\boldsymbol{x}$ is D-dimensional, $p(\boldsymbol{x}|\theta_k)$ will be a D-dimensional distribution
- Can compute MLE/MAP estimate or full posterior of θ_k
  - This essentially is a density estimation problem for the class-cond.
  - In principle, can use any density estimation method

E.g., if $p(\boldsymbol{x}|\theta_k)$ is multivariate Gaussian then assume it to have a diagonal covariance matrix instead of full covariance matrix

Such assumptions greatly reduce the number of parameters to be estimated

- Choice of the form of $p(\boldsymbol{x}|\theta_k)$ depends on various factors
  - Nature of input features, e.g.,
    - If $\boldsymbol{x} \in \mathbb{R}^D$, can use a D-dim Gaussian $\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$
    - If $\boldsymbol{x} \in \{0,1\}^D$, can use D Bernoullis (one for each feature)
    - Can also choose other more sophisticated distributions
  - Amount of training data available (important)
    - If D large and N_k small, it will be difficult to get a good estimate θ_k

In such cases, we may need to regularize θ_k or make some simplifying assumptions on $p(\boldsymbol{x}|\theta_k)$, such as features being conditionally independent given class e.g., $p(\boldsymbol{x}|\theta_k) = \prod_{d=1}^{D} p(x_d|\theta_{kd})$ - naïve Bayes

Especially if the number of features (D) is very large because large value of D means k consists of a large number of parameters (e.g., in the Gaussian case, $\theta_k = (\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k), D$ params for $\boldsymbol{\mu}_k$ and $O(D^2)$ params for $\boldsymbol{\Sigma}_k$. Can overfit

Generative Classification: At Test Time

- Recall the form of the conditional distribution of the label

Class-marginal accounts for the frequency of class k labels in the training data

Class-conditional distribution of inputs accounts for the shape/spread of class k

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

Probability of $\boldsymbol{x}_*$ belonging to class k is proportional to the fraction of training inputs from class k times the probability of $\boldsymbol{x}_*$ under the distribution of inputs from class k

$$\propto p(y_* = k) \times p(\boldsymbol{x}_*|y_* = k)$$

- If we assume the class-marginal to be uniform $(p(y_* = k) = 1/K)$ then

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

Basically, the probability of input under class k distribution

- The most likely label is $y_* = \text{argmax}_{k \in \{1,2,\ldots,K\}}\, p(y_* = k|\boldsymbol{x}_*)$

Generative Classification: At Test Time

- Prediction rule is

$$p(y_* = k|\boldsymbol{x}_*) \propto p(y_* = k) \times p(\boldsymbol{x}_*|y_* = k)$$

- If we have point estimates for π and $\{\theta_k\}_{k=1}^K$, say $\hat{\pi}$ and $\{\hat{\theta}_k\}_{k=1}^K$

$$p(y_* = k) = p(y_* = k|\hat{\pi}) = \hat{\pi}_k$$

$$p(\boldsymbol{x}_*|y_* = k) = p(\boldsymbol{x}_*|\hat{\theta}_k)$$

- If we have posteriors for π and $\{\theta_k\}_{k=1}^K$ then

PPD of y_*

$$p(y_* = k) = p(y_* = k|\boldsymbol{y}) = \int p(y_* = k|\pi)p(\pi|\boldsymbol{y})d\pi$$

PPD of $\boldsymbol{x}_*$

$$p(\boldsymbol{x}_*|y_* = k) = p(\boldsymbol{x}_*|\boldsymbol{X}_k) = \int p(\boldsymbol{x}_*|\theta_k)p(\theta_k|\boldsymbol{X}_k)d\theta_k$$

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 $\boldsymbol{p(x|y)}$

- Can also use it for regression (define $\boldsymbol{p(x,y)}$ via some distr. and obtain $\boldsymbol{p(y|x)}$ from it)

- Can also combine 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
  - Discriminative models have usually fewer parameters (e.g., just a weight vector)
  - Given "plenty" of training data, disc. models can usually outperform generative models

Recall prob linear regression and logistic reg

- 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 $\boldsymbol{p(x|y)}$) can be difficult for high-dim inputs
- Some prior work on combining discriminative and generative models. Examples:

$$\alpha \log p(y|x;\theta) + \beta \log p(x;\theta) \qquad p(x, y, \theta_d, \theta_g) = p_{\theta_d}(y|x) p_{\theta_g}(x) p(\theta_d, \theta_g)$$

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

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)