10-Generative Models

Discriminative models vs. Generative models

- In the training of discriminative models, we have a generative story for t given $\boldsymbol{x}$. For example,
  - Linear regression: $t|\boldsymbol{x} \sim N(\boldsymbol{w}^T\boldsymbol{x}+b, \sigma_\varepsilon^2)$
  - Softmax classification: $t|\boldsymbol{x} \sim \text{Multinomial}(\text{softmax}(\boldsymbol{W}\boldsymbol{x}+\boldsymbol{b}))$

  And we model the conditional probability of t given $\boldsymbol{x}$.

  $$p(t|\boldsymbol{x}; \textcircled{H})$$

  In discriminative models, we do NOT have a generative story of $\boldsymbol{x}$. In other words, we do not have a model for $p(\boldsymbol{x})$.
- In generative models, we will have a generative story concerning how both $\boldsymbol{x}$ and are generated, i.e., we will have a probabilistic model of $p(\boldsymbol{x}, t)$.
  - First generate $\boldsymbol{x}$, and then generate t given $\boldsymbol{x}$.

    $$p(\boldsymbol{x}, t) = p(\boldsymbol{x})p(t|\boldsymbol{x})$$

    $\boldsymbol{x}$ generated is more complicated than y, but we model $\boldsymbol{x}$ iid. We then model $p(t|\boldsymbol{x})$, which is not different from the discriminative model.
    => Not quite meaningful

  - First generate t and then generate $\boldsymbol{x}$ given t.

    $$p(\boldsymbol{x}, t) = p(t)p(\boldsymbol{x}|t)$$

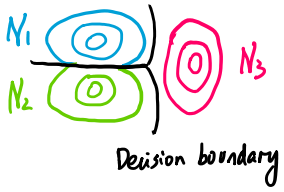
    For classification, y is a categorical variable. $p(t)$ models class frequency, and $p(\boldsymbol{x}|t)$ models the distribution of $\boldsymbol{x}$ in a certain category.

  Since we will model how $\boldsymbol{x}$ is obtained given t, we need to distinguish the type of $\boldsymbol{x}$, i.e., continuous variables or discrete variables.

Gaussian discriminant analysis

- Generative story for discrete y and continuous features $\boldsymbol{x}$.

  $$t \sim \text{Multinomial}(\pi_1, \cdots, \pi_K)$$
  $$\boldsymbol{x}|t = k \sim N(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$



- Parameter estimation: Maximum likelihood estimation

$$\hat{\boldsymbol{\pi}}, \hat{\boldsymbol{\mu}}_1, \cdots \hat{\boldsymbol{\mu}}_K, \hat{\boldsymbol{\Sigma}}_1, \cdots, \hat{\boldsymbol{\Sigma}}_K = \operatorname{argmax} \log p(\boldsymbol{X}, \boldsymbol{t}) \quad \text{[Generative model maximizes joint likelihood]}$$

$$= \operatorname{argmax} \sum_{m=1}^{M} \log p(\boldsymbol{x}^{(m)}, t^{(m)}) \quad \text{[data iid]}$$

$$= \operatorname{argmax} \sum_{m=1}^{M} \log \left[p(t^{(m)})\, p(\boldsymbol{x}^{(m)} | t^{(m)}) \right] \quad \text{[Generative story]}$$

$$= \operatorname{argmax} \sum_{m=1}^{M} \left[\log p(t^{(m)}) + \log p(\boldsymbol{x}^{(m)} | t^{(m)}) \right]$$

$$= \operatorname{argmax} \left[\sum_{m=1}^{M} \log p(t^{(m)}; \boldsymbol{\pi}) + \sum_{m=1}^{M} \log p(\boldsymbol{x}^{(m)} | t^{(m)}; \boldsymbol{\mu}_{t^{(m)}}, \boldsymbol{\Sigma}_{t^{(m)}}) \right]$$

$$= \operatorname{argmax} \left[\sum_{m=1}^{M} \log p(t^{(m)}; \boldsymbol{\pi}) + \sum_{k=1}^{K} \sum_{\substack{m=1..M \\ t^{(m)}=k}} p(\boldsymbol{x}^{(m)} | t^{(m)}=k; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right]$$

[rearrange of terms]

This shows that all parameters decompose into the following groups:

$$\boldsymbol{\pi}, (\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1), \cdots, (\boldsymbol{\mu}_K, \boldsymbol{\Sigma}_K)$$

We may optimize each group separately.

- MLE for multinomial distribution
  - Quick result (just counting):

$$\hat{\pi}_k = \frac{\sum_{m=1}^{M} \mathbb{1}\{t^{(m)} = k\}}{M} \quad \text{for } k = 1, \cdots, K$$

  - 1st Proof: Set $\pi_K = 1 - \pi_1 - \cdots - \pi_{K-1}$ and seek for a closed form solution
  - 2nd Proof (Lagrange multiplier method):

$$\log p(t^{(1)}, \cdots t^{(M)}) = \sum_{m=1}^{M} \log p(t^{(m)})$$

$$= \sum_{m=1}^{M} \log \pi_{t^{(m)}}$$

$$= \sum_{m=1}^{M} \sum_{k=1}^{K} \mathbb{1}\{t^{(m)} = k\} \log \pi_k$$

The optimization problem becomes

$$\text{minimize} \quad -\sum_{m=1}^{M} \sum_{k=1}^{K} \mathbb{1}\{t^{(m)} = k\} \log \pi_k$$

$$\text{subject to} \quad \pi_1 + \pi_2 + \cdots + \pi_K = 1$$

$$\text{Lagrangian: } \mathcal{L}(\boldsymbol{\pi}, \lambda) = -\sum_{m=1}^{M} \sum_{k=1}^{K} \left[\mathbb{1}\{t^{(m)} = k\} \cdot \log \pi_k \right] + \lambda(\pi_1 + \cdots + \pi_K - 1)$$

Optimality conditions:

$$\begin{cases} \dfrac{\partial \mathcal{L}(\boldsymbol{\pi}, \lambda)}{\partial \pi_k} = 0 & (1) \\ \dfrac{\partial \mathcal{L}(\boldsymbol{\pi}, \lambda)}{\partial \lambda} = 0 & (2) \end{cases}$$

(1) implies

$$\frac{\partial \mathcal{L}(\boldsymbol{\pi}, \lambda)}{\partial \pi_k} = -\sum_{m=1}^{M} \mathbb{1}\{t^{(m)} = k\} \cdot \frac{1}{\pi_k} + \lambda = 0 \qquad (3)$$

(2) implies

$$\pi_1 + \cdots \pi_K = 1 \qquad (4)$$

(3) implies

$$\lambda \pi_k = \sum_{m=1}^{M} \mathbb{1}\{t^{(m)} = k\} \qquad (5)$$

Put (5) in (4):

$$\sum_{k=1}^{K} \sum_{m=1}^{M} \mathbb{1}\{t^{(m)} = k\} = \lambda$$

which implies $\lambda = M$ (6)

Put (6) back in (5):

$$\hat{\pi}_k = \frac{\sum_{m=1}^{M} \mathbb{1}\{t^{(m)} = k\}}{M}$$

- MLE for Gaussian distributions

$$\hat{\boldsymbol{\mu}}_k = \frac{1}{M_k} \sum_{m=1}^{M} \mathbb{1}\{t^{(m)} = k\} \boldsymbol{x}_k$$

$$\hat{\boldsymbol{\Sigma}}_k = \frac{1}{M_k} \sum_{m=1}^{M} \mathbb{1}\{t^{(m)} = k\} \cdot (\boldsymbol{x}^{(m)} - \boldsymbol{\mu}_k)(\boldsymbol{x}^{(m)} - \boldsymbol{\mu}_k)^T$$

where $M_k = \sum_{m=1}^{M} \mathbb{1}\{t^{(m)} = k\}$

If we have an assumption that all covariance $\boldsymbol{\Sigma}_1, \cdots, \boldsymbol{\Sigma}_K$ are shared as $\boldsymbol{\Sigma}$, then

$$\hat{\boldsymbol{\Sigma}} = \frac{M_1}{M} \hat{\boldsymbol{\Sigma}}_1 + \cdots + \frac{M_K}{M} \hat{\boldsymbol{\Sigma}}_K$$

- Decision boundary for Gaussian discriminant analysis

Max *a posteriori* inference: $\hat{t} = \operatorname{argmax} \hat{p}(t|\boldsymbol{x}) = \operatorname{argmax} \hat{p}(t)\, p(\boldsymbol{x}|t)$

Considering any two categories i, j.

$\hat{t} = i$ being preferred than $\hat{t} = j$

$$\Leftrightarrow \quad \hat{p}(t=i|\boldsymbol{x}) \geq \hat{p}(t=j|\boldsymbol{x})$$

$$\Leftrightarrow \quad \pi_i \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \geq \pi_j \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)$$

$$\Leftrightarrow \quad \pi_i \frac{1}{(2\pi)^{d/2}|\boldsymbol{\Sigma}_i|^{1/2}} \exp\left\{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_i)\right\}$$

$$\geq \pi_j \frac{1}{(2\pi)^{d/2}|\boldsymbol{\Sigma}_j|^{1/2}} \exp\left\{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_j)^T \boldsymbol{\Sigma}_j^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_j)\right\}$$

$$\Leftrightarrow$$

$$-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_i) + \log\frac{\pi_i}{|\boldsymbol{\Sigma}_i|^{1/2}} \geq -\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_j)^T \boldsymbol{\Sigma}_j^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_j) + \log\frac{\pi_j}{|\boldsymbol{\Sigma}_j|^{1/2}}$$

If $\boldsymbol{\Sigma}_i \neq \boldsymbol{\Sigma}_j$, then the decision boundary is quadratic.

If $\boldsymbol{\Sigma}_i = \boldsymbol{\Sigma}_j$, then the decision boundary is linear

[See the above RGB cartoon]

Naïve Bayes for discrete variables

- For simplicity, we only consider binary features

  $x_i \in \{0,1\}$, i.e., $\boldsymbol{x} \in \{0,1\}^d$
- Generative story for discrete y and (binary) discrete features $\boldsymbol{x}$

  $t \sim \text{Multinomial}(\pi_1, \cdots, \pi_K)$

  $\boldsymbol{x}_i | t = k \sim \text{Bernoulli}(p_{k,i})$

  - Such model can be used to represent a document in text classification. For example, $t = 1$ indicates Spam, $t = 0$ indicates NotSpam. $x_i \in \{0,1\}$ indicates if the ith word in the vocabulary occurs in the document.

- MLE for Naïve Bayes [HW]

  Please show that the parameters of naïve Bayes decompose, i.e., the probability factorizes (for the same reason as Gaussian discriminative analysis). Write out the MLE for naïve Bayes (which is simply counting). No proof is needed for the second part, because the MLE for

No proof is needed for the second part, because the MLE for multinomial distribution has been clear in the Gaussian discriminative analysis.

- Inference with the naïve Bayes model

  Category k is preferred than Category $j64$

$$\Leftrightarrow \quad \pi_k \prod_{i=1}^{d} p_{k,i}^{x_i} (1-p_{k,i})^{1-x_i} \geq \pi_j \prod_{i=1}^{d} p_{j,i}^{x_i} (1-p_{j,i})^{1-x_i}$$

$$\Leftrightarrow \quad \log \pi_k + \sum_{i=1}^{d} \left[x_i \log p_{k,i} + (1-x_i) \log (1-p_{k,i}) \right]$$

$$\geq \log \pi_j + \sum_{i=1}^{d} \left[x_i \cdot \log p_{j,i} + (1-x_i) \log (1-p_{j,i}) \right]$$

  The decision boundary is again linear.

Comparing discriminative models and generative models

- For naïve Bayes and Gaussian discriminative analysis with shared covariance, the decision boundary is linear, which yield the same hypothesis class as logistic regression.
  - Even for Gaussian discriminative analysis with different covariance matrices, the decision boundary is quadratic. We may introduce quadratic features for logistic regression, too.

  Thus, generative models do not enhance the hypothesis class, compared with discriminative models.

- Generative and discriminative models have the same inference criterion (when we concern the accuracy as our measure of success.

$$\hat{t}_* = \operatorname{argmax} \hat{p}(t \mid \boldsymbol{x})$$

  However, they differ in the training objective. Discriminative models treat input $\mathbf{X}$ as constants (unkown variables), whereas generative models model X as random variables.

$$\hat{\boldsymbol{\theta}} = \operatorname{argmax} p(\mathbf{y} \mid \mathbf{X}; \boldsymbol{\theta}) \quad \text{vs.} \quad \hat{\boldsymbol{\theta}} = \operatorname{argmax} p(\mathbf{X}, \mathbf{y}; \boldsymbol{\theta})$$

  Note: MAP parameter estimation can be applied to both.

- Generative models usually have far more parameters than discriminative models, thus requiring more data for training.

- Generative models work for more scenarios than classification.

- Unsupervised learning. MLE is still well defined in generative models if **X** is not given. [Additional lecture]
- Inferring some of x_i during prediction, given other features and labels.