Introduction to Machine Learning

Answers to Exercise 6
Generative Models

Jingtao Min

July 26, 2022

1 Discriminative and generative models

(a) Naive Bayes classifier is a generative model; logistic regression, SVM and neural networks are examples of discriminative models.

(b) When using a discriminative model, only the posterior $P(Y|X)$ is trained and used for prediction.

(c) When using a generative model, the joint distribution $P(X, Y)$ is trained and used for prediction. In many cases the likelihood $P(X|Y)$, the prior $P(Y)$ are obtained along the way, and if the evidence $P(X)$ is also modelled / calculated, the posterior $P(Y|X)$ can also be obtained.

(d) With the model prior $P(Y)$ and likelihood $P(X|Y)$, the joint distribution can be calculated as $P(X, Y) = P(X|Y)P(Y)$. Marginalizing the joint distribution, one gets the evidence $P(X)$, and finally the posterior $P(Y|X) = P(X, Y)/P(X)$.

Suppose a Gaussian Bayes classifier is used for binary classification ($y \in \{-1, +1\}$).

(e) Linear Discriminant Analysis (LDA) assumes shared covariance, i.e. $\Sigma_+ = \Sigma_-$ between the two classes;

(f) Fisher's Linear Discriminant Analysis is a term used almost interchangeably with LDA, but in some contexts it assumes homogeneous prior $P(Y = y) = \frac{1}{2}$ in addition to $\Sigma_+ = \Sigma_-$;

(g) Quadratic Discriminant Analysis makes no such assumptions;

(h) Gaussian Naive Bayes (GNB) classifier makes the assumption that the feature elements are independent random variables, i.e. covariance is diagonal $\Sigma_y = \operatorname{diag}\left(\sigma_{y,i}^2\right)$.

(i) With generative modelling it is possible to explicitly include a bias in the model by defining the structure of likelihood $P(X|Y)$.

Suppose we have a very large dataset $\{(x_i, y_i)\}_{i=1}^n$ with $x_i \in \mathbb{R}$ and $y_i \in \{\pm 1\}$, and each sample is drawn i.i.d. from the joint distribution $P(X, Y)$ as shown in the plot.

(j) To train a model to predict y_{new} based on new feature x_{new}, a Gaussian Bayes classifier should be used. The covariance is clearly not the same across classes, so LDA cannot be used; the decision boundary is clearly non-linear, so Logistic regression cannot be used.

2 Gaussian-mixture Bayes classifier

A Gaussian-mixture Bayes classifier is similar to Gaussian Bayes classifier, but with a richer likelihood which is modelled as a mixture of Gaussian distributions:

$$p_{X|Y}(x|y; k, w, \mu, \Sigma) = \sum_{j=1}^{k} w_j^{(y)} \mathcal{N}\left(x; \mu_j^{(y)}, (\sigma_j^{(y)})^2\right) \tag{1}$$

Suppose we have a dataset $\{(x_i, y_i)\}_{i=1}^{10000}$ with $x_i \in \mathbb{R}$ and $y_i \in \{\pm 1\}$, and each sample is drawn i.i.d. from the joint distribution $P(X, Y)$, as shown in the histogram.

(a) A Gaussian-mixture Bayes classifier would clearly outperform Gaussian Bayes classifier, as neither class distribution can be modelled well with a single Gaussian.

(b) For the same reason I would assume that both Gaussian Bayes classifier and its special variant Gaussian Naive Bayes classifier should work poorly. If I had to choose I would say GNB would perform worse.

For a Gaussian-mixture Bayes classifier,

(c) a number of $k = 3$ mixtures can be chosen to model the distributions well.

(d) Choosing $k = 10$ might not deteriorate the prediction significantly;

(e) but the classification performance is expected to decrease strongly when k is comparable to or larger than the number of samples.

(f) Let p_{+1} be the parameter that models $P(Y = +1)$, from a probabilistic point of view the likelihood function for the label distribution

$$\begin{aligned} P(y_{1\cdots n}) &= \prod_{i=1}^{n} P(Y = y_i) = P(Y = +1)^{n_+} P(Y = -1)^{n_-} = p_{+1}^{n_+} (1 - p_{+1})^{n_-} \\ \log P(y_{1\cdots n}) &= n_+ \log p_{+1} + n_- \log (1 - p_{+1}) . \end{aligned} \tag{2}$$

To maximize the logarithmic likelihood we can find its stationary point, which yields

$$\begin{aligned} &\frac{\partial}{\partial p_{+1}} \log P(y_{1\cdots n}) = \frac{n_+}{p_{+1}} - \frac{n_-}{1 - p_{+1}} = \frac{n_+ - (n_+ + n_-) p_{+1}}{p_{+1} (1 - p_{+1})} \\ &\implies \quad p_{+1} = \frac{n_+}{n_+ + n_-} = \frac{n_+}{n} = \frac{\{\#y = +1\}}{\{\#y = +1\} + \{\#y = -1\}} . \end{aligned} \tag{3}$$

(g) Training the parameters $w_j^{(y)}$, $\mu_j^{(y)}$ and $\Sigma_j^{(y)}$ require solving the following optimization problem, stemming from maximum likelihood estimation (MLE)

$$\left(w_{1\cdots k}^{(y),*}, \mu_{1\cdots k}^{(y),*}, \Sigma_{1\cdots k}^{(y),*} \right) = \arg \min_{w,\mu,\Sigma} \sum_{i, y_i = y} \left[-\log \sum_{j=1}^{k} w_j^{(y)} \mathcal{N} \left(x_i; \mu_j^{(y)}, \Sigma_j^{(y)} \right) \right] . \tag{4}$$

Hereafter I drop the y superscript, as it is clear the optimization is done class-wise.

(h) The training can be performed using the Expected-Maximum-likelihood (EM) algorithm. The E step is used to derive the latent variable z_i that determines the membership of the sample. Using a hard EM, where each sample is exclusively attributed to one component of the mixture, the E-step can be written as

$$\begin{aligned} z_i &= \arg \max_{z \in \{1\cdots k\}} P(z | x_i, w_z^{(y)}, \mu_z^{(y)}, \Sigma_z^{(y)}) = \arg \max_{z \in \{1\cdots k\}} P(z | w_z^{(y)}, \mu_z^{(y)}, \Sigma_z^{(y)}) P(x_i | z, w_z^{(y)}, \mu_z^{(y)}, \Sigma_z^{(y)}) \\ &= \arg \max_{z \in \{1\cdots k\}} w_z^{(y)} \mathcal{N} \left(x_i; \mu_z^{(y)}, \Sigma_z^{(y)} \right) \\ &= \arg \max_{z \in \{1\cdots k\}} \log w_z^{(y)} - (x_i - \mu_z^{(y)})^T \left(\Sigma_z^{(y)} \right)^{-1} (x_i - \mu_z^{(y)}) \end{aligned} \tag{5}$$

(i) Hard EM has the potential problem when dealing with overlapping components/clusters. This is the case with the $y = -1$ dataset, where two clusters seem to overlap.

3 EM algorithm for mixture of distributions

For an integer random variable x over the values $\{1, 2, 3\}$, a generative model uses the following 2 distributions

$$p_1(x) = \begin{cases} \alpha, & x = 1 \\ 1 - \alpha, & x = 2 \\ 0, & x = 3 \end{cases} \qquad p_1(x) = \begin{cases} 0, & x = 1 \\ 1 - \beta, & x = 2 \\ \beta, & x = 3 \end{cases} . \tag{6}$$

The overall model reads $p(x) = \gamma p_1(x) + (1 - \gamma) p_2(x)$. The numbers of observations in each classes are $k_1, k_2, k_3 = \{30, 20, 60\}$, respectively. EM algorithm is initialized with parameters $\alpha_0, \beta_0, \gamma_0 = \left(\frac{1}{2}, \frac{1}{2}, \frac{1}{2} \right)$.

(a) As in the case of Gaussian mixtures, the latent variable z denotes the membership of the sample, and takes the values $\{1, 2\}$. Its distribution is given by the mixture weights

$$p(z) = \begin{cases} \gamma, & z = 1 \\ 1-\gamma, & z = 2 \end{cases} . \tag{7}$$

The joint distribution over the observed variable x and the latent variable z is given by

$$p(x, z) = p(x|z)p(z) = \begin{cases} p(x|z=1)p(z=1), & z = 1 \\ p(x|z=2)p(z=2), & z = 2 \end{cases} . \tag{8}$$

The distribution can be tabulated

$z\backslash P(x,z)\backslash x$	1	2	3
1	$\gamma\alpha$	$\gamma(1-\alpha)$	0
2	0	$(1-\gamma)(1-\beta)$	$(1-\gamma)\beta$

(b) For a given x_i, the responsibility z_i is evaluated in the E-step as

$$\begin{aligned} z_i &= \arg\max_z P(z|x_i) = \arg\max_z \frac{P(x_i, z)}{P(x_i)} = \arg\max_z P(x_i, z) \\ &= \begin{cases} 1, & x = 1 \quad \text{or} \quad x = 2, \frac{\gamma}{1-\gamma} \geq \frac{1-\beta}{1-\alpha} \\ 2, & x = 3 \quad \text{or} \quad x = 2, \frac{\gamma}{1-\gamma} \leq \frac{1-\beta}{1-\alpha} \end{cases} . \end{aligned} \tag{9}$$

(c) Once the latent variables are assigned, the parameters can be re-evaluated in the M-step as

$$\begin{aligned} \gamma &= \frac{\{\#z_i = 1\}}{\{\#z_i = 1\} + \{\#z_i = 2\}}, \\ \alpha &= \frac{\{\#(x_i, z_i) = (1,1)\}}{\{\#(x_i, z_i) = (1,1)\} + \{\#(x_i, z_i) = (2,1)\}}, \\ \beta &= \frac{\{\#(x_i, z_i) = (3,2)\}}{\{\#(x_i, z_i) = (3,2)\} + \{\#(x_i, z_i) = (2,2)\}}. \end{aligned} \tag{10}$$

(d) Given the initial conditions and the observations, we obtain the parameters

$$\alpha = \frac{3}{5}, \quad \beta = 1, \quad \gamma = \frac{5}{11} \tag{11}$$

4 Generative adversarial networks (GANs)

Let the discriminator and the generator be deonted as $\mathcal{D}$ and $\mathcal{G}$, respectively. The training objective for GAN is given by

$$\min_{\mathcal{G}} \max_{\mathcal{D}} \mathbb{E}_{\mathbf{x}}[\log \mathcal{D}(\mathbf{x})] + \mathbb{E}_{\mathbf{z}}\left[\log\left(1 - \mathcal{D}(\mathcal{G}(\mathbf{z}))\right)\right], \tag{12}$$

where $\mathbf{z}$ is the random input variable and $\mathbf{z} \sim \mathcal{N}(0, \mathbf{I}) \in \mathbb{R}^n$. We assume the true data has the distribution $\mathbf{x} \sim p_{\text{data}}$.

(a) If $\mathcal{D}$ and $\mathcal{G}$ has enough capacity, the optimal generator would be such that

$$\mathcal{G}(\mathbf{z}) \sim p_{\text{data}} \tag{13}$$

(b) The objective can be interpreted as a two-player game.

(c) In its formal expression, the discriminator strives to maximize the objective

$$\begin{aligned} &\max_{\mathcal{D}} \mathbb{E}_{\mathbf{x}\sim\rho_d} \log \mathcal{D}(\mathbf{x}) + \mathbb{E}_{\mathbf{x}\sim\rho_G} \log\left(1 - \mathcal{D}(\mathbf{x})\right) \\ &= \max_{\mathcal{D}} \int_{\Omega_x} \left[\rho_d(\mathbf{x}) \log \mathcal{D}(\mathbf{x}) + \rho_G(\mathbf{x}) \log\left(1 - \mathcal{D}(\mathbf{x})\right)\right] d\mathbf{x} \end{aligned} \tag{14}$$

The optimized $\mathcal{D}$ should be able to maximize the point-wise integrand, leading to

$$\max_{\mathcal{D}} \rho_d(\mathbf{x}) \log \mathcal{D}(\mathbf{x}) + \rho_G(\mathbf{x}) \log\left(1 - \mathcal{D}(\mathbf{x})\right) \tag{15}$$

The optimal condition then yields

$$\frac{\partial}{\partial \mathcal{D}(\mathbf{x})} \left[\rho_d(\mathbf{x}) \log \mathcal{D}(\mathbf{x}) + \rho_G(\mathbf{x}) \log\left(1 - \mathcal{D}(\mathbf{x})\right)\right] = 0 \quad \Longrightarrow \quad \mathcal{D}(\mathbf{x}) = \frac{\rho_d(\mathbf{x})}{\rho_d(\mathbf{x}) + \rho_G(\mathbf{x})}. \tag{16}$$

This is the predicted probability that $\mathbf{x} \in \rho_{\text{data}}$; inversely we have the probability that the sample is generated by $\mathcal{G}$

$$\frac{\rho_G(\mathbf{x})}{\rho_d(\mathbf{x}) + \rho_G(\mathbf{x})}. \tag{17}$$