Machine Learning Midterm Exam Answer Sheet

Course: Machine Learning(H)(CS329) - Instructor: Qi Hao

Name: Jianan Xie(谢嘉楠)

SID: 12110714

Problem 1. Least Square (15 pts)

a) Consider $Y = AX + V$ and $V \sim N(v|0, Q)$, what is the least square solution of X ?

Ans:

$$E(X) = \frac{1}{2}(\mathbf{y} - \mathbf{AX})^{\mathrm{T}}\mathrm{Q}^{-1}(\mathbf{y} - \mathbf{AX})$$

find $\hat{X} = argmin(E(X))$:

$$\frac{\partial E(X)}{\partial \mathbf{X}} = \mathbf{A}^T Q^{-1}\mathbf{AX} - \mathbf{A}^{\mathrm{T}}\mathrm{Q}^{-1}\mathbf{y} = 0$$

Then we find $\hat{X} = (\mathbf{A}^{\mathrm{T}}\mathrm{Q}^{-1}\mathbf{A})^{-1}\mathbf{A}^{\mathrm{T}}\mathrm{Q}^{-1}\mathbf{y}$.

b) If there is a constraint of $b^{\mathrm{T}}X = c$, what is the optimal solution of X ?

Ans:

$$E(X) = \frac{1}{2}(\mathbf{y} - \mathbf{AX})^{\mathrm{T}}\mathrm{Q}^{-1}(\mathbf{y} - \mathbf{AX}) + \lambda(\mathbf{b}^{\mathrm{T}}\mathbf{X} - \mathrm{c})$$

find $\hat{X} = argmin(E(X))$:

$$\frac{\partial E(X)}{\partial \mathbf{X}} = \mathbf{A}^T Q^{-1}\mathbf{AX} - \mathbf{A}^{\mathrm{T}}\mathrm{Q}^{-1}\mathbf{y} + \lambda\mathbf{b} = 0$$
$$\frac{\partial E(X)}{\partial \lambda} = \mathbf{b}^{\mathrm{T}}\mathbf{X} - \mathrm{c} = 0$$

Then we derive that

$$\hat{X} = (\mathbf{A}^{\mathrm{T}}\mathrm{Q}^{-1}\mathbf{A})^{-1}(\mathbf{A}^{\mathrm{T}}\mathrm{Q}^{-1}\mathbf{y} - \lambda\mathbf{b}), \quad \mathbf{b}^{\mathrm{T}}\hat{\mathbf{X}} = \mathrm{c}$$

Thus, $\mathbf{b}^{\mathrm{T}}\hat{\mathrm{X}} = \mathbf{b}^{\mathrm{T}}(\mathbf{A}^{\mathrm{T}}\mathrm{Q}^{-1}\mathbf{A})^{-1}(\mathbf{A}^{\mathrm{T}}\mathrm{Q}^{-1}\mathbf{y} - \lambda\mathbf{b}) = \mathbf{b}^{\mathrm{T}}(\mathbf{A}^{\mathrm{T}}\mathrm{Q}^{-1}\mathbf{A})^{-1}\mathbf{A}^{\mathrm{T}}\mathrm{Q}^{-1}\mathbf{y} - \lambda\mathbf{b}^{\mathrm{T}}(\mathbf{A}^{\mathrm{T}}\mathrm{Q}^{-1}\mathbf{A})^{-1}\mathbf{b} = \mathrm{c}$, which gives $\lambda = [\mathbf{b}^{\mathrm{T}}(\mathbf{A}^{\mathrm{T}}\mathrm{Q}^{-1}\mathbf{A})^{-1}\mathbf{b}]^{-1}\mathbf{b}^{\mathrm{T}}(\mathbf{A}^{\mathrm{T}}\mathrm{Q}^{-1}\mathbf{A})^{-1}\mathbf{A}^{\mathrm{T}}\mathrm{Q}^{-1}\mathbf{y} - [\mathbf{b}^{\mathrm{T}}(\mathbf{A}^{\mathrm{T}}\mathrm{Q}^{-1}\mathbf{A})^{-1}\mathbf{b}]^{-1}\mathrm{c}$. For convenience, we denote $\hat{X_1} = (\mathbf{A}^{\mathrm{T}}\mathrm{Q}^{-1}\mathbf{A})^{-1}\mathbf{A}^{\mathrm{T}}\mathrm{Q}^{-1}\mathbf{y}$, which is the least square solution of X in (a). Then $\hat{X} = \hat{X_1} - \lambda(\mathbf{A}^{\mathrm{T}}\mathrm{Q}^{-1}\mathbf{A})^{-1}\mathbf{b}$, $\lambda = [\mathbf{b}^{\mathrm{T}}(\mathbf{A}^{\mathrm{T}}\mathrm{Q}^{-1}\mathbf{A})^{-1}\mathbf{b}]^{-1}(\mathbf{b}^{\mathrm{T}}\hat{\mathrm{X}_1} - \mathrm{c})$.

Therefore, $\hat{X} = \hat{X_1} - (\mathbf{A}^{\mathrm{T}}\mathrm{Q}^{-1}\mathbf{A})^{-1}\mathbf{b}[\mathbf{b}^{\mathrm{T}}(\mathbf{A}^{\mathrm{T}}\mathrm{Q}^{-1}\mathbf{A})^{-1}\mathbf{b}]^{-1}(\mathbf{b}^{\mathrm{T}}\hat{\mathrm{X}_1} - \mathrm{c})$

c) If there is an additional constraint of $X^{\mathrm{T}}X = d$, in addition to the constraint in b), what is the optimal solution of X ?

Ans:

$$E(X) = \frac{1}{2}(\mathbf{y} - \mathbf{AX})^{\mathrm{T}}\mathrm{Q}^{-1}(\mathbf{y} - \mathbf{AX}) + \lambda_1(\mathbf{b}^{\mathrm{T}}\mathbf{X} - \mathrm{c}) + \lambda_2(\mathbf{X}^{\mathrm{T}}\mathbf{X} - \mathrm{d})$$

find $\hat{X} = argmin(E(X))$:

$$\frac{\partial E(X)}{\partial \mathbf{X}} = \mathbf{A}^T Q^{-1}\mathbf{AX} - \mathbf{A}^{\mathrm{T}}\mathrm{Q}^{-1}\mathbf{y} + \lambda_1\mathbf{b} + 2\lambda_2\mathbf{X} = 0$$
$$\frac{\partial E(X)}{\partial \lambda_1} = \mathbf{b}^{\mathrm{T}}\mathbf{X} - \mathrm{c} = 0$$
$$\frac{\partial E(X)}{\partial \lambda_2} = \mathbf{X}^{\mathrm{T}}\mathbf{X} - d = 0$$

Then we derive that

$$\hat{X} = (\mathbf{A}^{\mathrm{T}}\mathrm{Q}^{-1}\mathbf{A} + 2\lambda_2\mathbf{I})^{-1}(\mathbf{A}^{\mathrm{T}}\mathrm{Q}^{-1}\mathbf{y} - \lambda_1\mathbf{b}), \quad \mathbf{b}^{\mathrm{T}}\hat{\mathbf{X}} = \mathrm{c}, \quad \mathbf{X}^{\mathrm{T}}\mathbf{X} - \mathrm{d} = 0$$

Thus, after we solve the equations above to get the λ_1 and λ_2, take them back into the equation $\hat{X} = (\mathbf{A}^{\mathrm{T}}\mathrm{Q}^{-1}\mathbf{A} + 2\lambda_2\mathbf{I})^{-1}(\mathbf{A}^{\mathrm{T}}\mathrm{Q}^{-1}\mathbf{y} - \lambda_1\mathbf{b})$ to obtain the optimal solution of X.

d) If both A and X are unknown, how to solve A and X alternatively by using two constraints of $X^{\mathrm{T}}X = d$ and $Trace(A^{\mathrm{T}}A) = e$?

Ans:

$$E(X) = \frac{1}{2}(\mathbf{y} - \mathbf{AX})^{\mathrm{T}}\mathrm{Q}^{-1}(\mathbf{y} - \mathbf{AX}) + \lambda_1(\mathbf{X}^{\mathrm{T}}\mathbf{X} - \mathrm{d}) + \lambda_2(\mathrm{Tr}(\mathbf{A}^{\mathbf{T}}\mathbf{A}) - \mathrm{e})$$

find $\hat{A}, \hat{X} = argmin(E(X))$:

$$\begin{aligned}
\frac{\partial E(X)}{\partial \mathbf{X}} &= \mathbf{A}^{T}Q^{-1}\mathbf{AX} - \mathbf{A}^{\mathrm{T}}\mathrm{Q}^{-1}\mathbf{y} + 2\lambda_1\mathbf{X} = 0 \\
\frac{\partial E(X)}{\partial \mathbf{A}} &= Q^{-1}\mathbf{AXX}^{\mathrm{T}} - \frac{1}{2}Q^{-1}\mathbf{y}\mathbf{X}^{\mathrm{T}} - \frac{1}{2}\mathbf{X}(Q^{-1}\mathbf{y})^{\mathrm{T}} + 2\lambda_2\mathbf{A} = 0 \\
\frac{\partial E(X)}{\partial \lambda_1} &= \mathbf{X}^{\mathrm{T}}\mathbf{X} - d = 0 \\
\frac{\partial E(X)}{\partial \lambda_2} &= Tr(\mathbf{A}^{\mathbf{T}}\mathbf{A}) - e = 0
\end{aligned}$$

Then we first suppose a proper $\mathbf{A_0}$, then take it into equations above to find $\hat{X_0}$ and then take $\hat{X_0}$ into equations above to find $\hat{\mathbf{A_1}}$. We do these two steps iteratively and derive the final optimal answer we want after certain amount of steps.

Problem 2. Linear Gaussian System (10 pts)

Consider $Y = AX + V$, where X and V are Gaussian, $X \sim N(x|\mathbf{m}_0, \mathbf{\Sigma}_0)$, $V \sim N(v|\mathbf{0}, \beta^{-1}\mathbf{I})$. What are the conditional distribution, $p(Y|X)$, the joint distribution $p(Y, X)$, the marginal distribution, $p(Y)$, the posterior distribution, $p(X|Y = \mathbf{y}, \beta, \mathbf{m}_0, \mathbf{\Sigma}_0)$, the posterior predictive distribution, $p(\hat{Y}|Y = \mathbf{y}, \beta, \boldsymbol{m}_0, \mathbf{\Sigma}_0)$, and the prior predictive distribution, $p(Y|\beta, \boldsymbol{m}_0, \Sigma_0)$, respectively?

Ans:

Introduce $Z = \begin{pmatrix} X \\ Y \end{pmatrix}$, so $\mathbb{E}[Z] = \begin{pmatrix} \mathbf{m_0} \\ A\mathbf{m_0} \end{pmatrix}$, $\mathbb{C}ov[Z] = \begin{pmatrix} \mathbf{\Sigma_0} & \mathbf{\Sigma_0}A^{\mathrm{T}} \\ A\mathbf{\Sigma_0} & \beta^{-1}\mathbf{I} + A\mathbf{\Sigma}_0A^{\mathrm{T}} \end{pmatrix}$. And we have learned that for $\begin{pmatrix} x_a \\ x_b \end{pmatrix}$, $\mathbb{E}[x_a|x_b] = \mu_a + \Sigma_{ab}\Sigma_{bb}^{-1}(x_b - \mu_b)$, $Var[x_a|x_b] = \Sigma_{aa} - \Sigma_{ab}\Sigma_{bb}^{-1}\Sigma_{ba}$.

Then we get:

$$\begin{aligned}
p(Y|X) &= N(Y|AX, \beta^{-1}\mathbf{I}) \\
p(Y, X) &= p(X)p(Y|X) = N(x|\mathbf{m}_0, \mathbf{\Sigma}_0)N(Y|AX, \beta^{-1}\mathbf{I}) \\
p(Y) &= \int p(Y, X)dX = \int N(x|\mathbf{m}_0, \mathbf{\Sigma}_0)N(Y|AX, \beta^{-1}\mathbf{I})dX = N(Y|A\mathbf{m_0}, A\mathbf{\Sigma}_0A^{\mathrm{T}} + \beta^{-1}\mathbf{I}) \\
p(X|Y = \mathbf{y}, \beta, \mathbf{m}_0, \mathbf{\Sigma}_0) &= N(X|\mathbf{\Sigma}(\mathbf{A}^{\mathrm{T}}(\beta\mathbf{I})Y + \mathbf{\Sigma_0}^{-1}\mathbf{m_0}), \mathbf{\Sigma}) \\
where \quad \mathbf{\Sigma} &= (\mathbf{\Sigma_0}^{-1} + \mathbf{A}^{\mathrm{T}}(\beta\mathbf{I})\mathbf{A})^{-1} \\
p(\hat{Y}|Y = \mathbf{y}, \beta, \mathbf{m}_0, \mathbf{\Sigma}_0) &= \int p(\hat{Y}|X)p(X|Y = \mathbf{y}, \beta, \mathbf{m}_0, \mathbf{\Sigma}_0)dX \\
&= N(\hat{Y}|A\mathbf{\Sigma}(\mathbf{A}^{\mathrm{T}}(\beta\mathbf{I})Y + \mathbf{\Sigma_0}^{-1}\mathbf{m_0}), \beta^{-1}\mathbf{I} + A\mathbf{\Sigma}A^{\mathrm{T}}) \\
p(Y|\beta, \mathbf{m}_0, \Sigma_0) &= \int p(\hat{Y}|X)p(X)dX \\
&= N(Y|A\mathbf{m_0}, A\mathbf{\Sigma}_0A^{\mathrm{T}} + \beta^{-1}\mathbf{I})
\end{aligned}$$

Problem 3. Linear Regression (10 pts)

Consider $y = \mathbf{w}^{\mathrm{T}}\phi(\mathbf{x}) + v$, where v is Gaussian, i.e., $v \sim N(v|0, \beta^{-1})$, and $\mathbf{w}$ has a Gaussian prior, i.e., $\mathbf{w} \sim N(\mathbf{w}|\mathbf{m}_0, \alpha^{-1}\mathbf{I})$. Assume that $\phi(x)$ is known, please derive the posterior distribution, $p(\mathbf{w}|D, \beta, \mathbf{m}_0, \alpha)$, the posterior predictive distribution, $p(\hat{y}|\hat{x}, D, \beta, \mathbf{m}_0, \alpha)$, and the prior predictive distribution, $p(D|\beta, \mathbf{m}_0, \alpha)$, respectively, where $D = \{\phi_n, y_n\}$, $n = 1, \ldots, N$, is the training data set and $\phi_n = \phi(\mathbf{x}_n)$.

Ans:

The posterior distribution is:

$$p(\mathbf{w}|D, \beta, \mathbf{m}_0, \alpha) = N(\mathbf{w}|\mathbf{m_N}, \mathbf{S_N})$$

where

$$\begin{aligned}
\mathbf{m_N} &= \mathbf{S_N}(\alpha\mathbf{I}\mathbf{m_0} + \beta\mathbf{\Phi}^{\mathrm{T}}\mathbf{t}) \\
\mathbf{S_N}^{-1} &= \alpha\mathbf{I} + \beta\mathbf{\Phi}^{\mathrm{T}}\mathbf{\Phi} \\
\mathbf{\Phi} &= (\phi_1, \phi_2, \ldots, \phi_N)^{\mathrm{T}} \\
\mathbf{t} &= (y_1, y_2, \ldots, y_N)^{\mathbf{T}}
\end{aligned}$$

The posterior predictive distribution is:

$$p(\hat{y}|\hat{x}, D, \beta, \mathbf{m}_0, \alpha) = N(\hat{y}|\mathbf{m_N}^{\mathrm{T}}\phi(\hat{x}), \sigma_N^2(\hat{x}))$$

where

$$\sigma_N^2(x) = \frac{1}{\beta} + \phi(x)^{\mathrm{T}}\mathbf{S_N}\phi(x)$$

The prior predictive distribution is:

$$\begin{aligned}
p(D|\beta, \mathbf{m}_0, \alpha) &= \int p(D|\mathbf{w})p(\mathbf{w})d\mathbf{w} \\
&= \int \prod_{n=1}^{N} N(y_n|\mathbf{w}^{\mathrm{T}}\phi_n, \beta^{-1})N(\mathbf{w}|\mathbf{m_0}, \alpha^{-\mathbf{1}}\mathbf{I})d\mathbf{w} \\
&= \prod_{n=1}^{N} N(y_n|\mathbf{m}_0^T\phi_n, \phi_n^{\mathrm{T}}(\alpha^{-1}\mathbf{I})\phi_n + \beta^{-1})
\end{aligned}$$

Problem 4. Logistics Regression (10 pts)

Consider a two-class classification problem with the logistic sigmoid function, $y = \sigma(\mathbf{w}^{\mathrm{T}}\phi(\mathbf{x}))$, for a given data set $D = \{\phi_n, t_n\}$, where $t_n \in \{0, 1\}$, $\phi_n = \phi(\mathbf{x}_n)$, n = 1,..., N, and the likelihood function is given by

$$p(\mathbf{t}|\mathbf{w}) = \prod_{n=1}^{N} y_n^{t_n}(1 - y_n)^{1-t_n}$$

where $\mathbf{w}$ has a Gaussian priori, i.e., $\mathbf{w} \sim N(\mathbf{w}|\mathbf{m}_0, \alpha^{-1}\mathbf{I})$. Please derive the posterior distribution, $p(\mathbf{w}|D, \mathbf{m}_0, \alpha)$, the posterior predictive distribution, $p(t|x, D, \mathbf{m}_0, \alpha)$, and the prior predictive distribution, and $p(D|\mathbf{m}_0, \alpha)$, respectively. (Hint: using Delta approximation and Laplace approximation properly).

Ans:

The posterior distribution is:

$$p(\mathbf{w}|D, \mathbf{m}_0, \alpha) \propto p(\mathbf{w})p(\mathbf{t}|\mathbf{w})$$

$$\Rightarrow \ln p(\mathbf{w}|\mathbf{t}) = -\frac{\alpha}{2}(\mathbf{w} - \mathbf{m}_0)^{\mathrm{T}}(\mathbf{w} - \mathbf{m_0}) + \sum_{n=1}^{N}\{t_n \ln y_n + (1 - t_n)\ln(1 - y_n)\} + const$$

To obtain a Gaussian approximation of posterior distribution, we need to maximize $\ln p(\mathbf{w}|\mathbf{t})$ first in order to get the MAP solution $\mathbf{w}_{MAP}$, which defines the mean of the Gaussian distribution.

$$-\nabla \ln p(\mathbf{w}|\mathbf{t}) = \alpha(\mathbf{w} - \mathbf{m}_0) + \sum_{n=1}^{N}(y_n - t_n)\phi_n$$

$$S_N^{-1} = -\nabla\nabla \ln p(\mathbf{w}|\mathbf{t}) = \alpha\mathbf{I} + \sum_{n=1}^{N} y_n(1 - y_n)\phi_n\phi_n^{\mathrm{T}}$$

then we use $\mathbf{w}_{MAP} \rightarrow \mathbf{w}_{new} = \mathbf{w}_{old} - S_N(-\nabla \ln p(\mathbf{w}|\mathbf{t}))$ to find the $\mathbf{w}_{MAP}$. And the Gaussian approximation posterior distribution is:

$$q(\mathbf{w}) = N(\mathbf{w}|\mathbf{w}_{MAP}, S_N)$$

The posterior predictive distribution is:

$$p(t|x, D, \mathbf{m}_0, \alpha) \backsimeq \int \sigma(\mathbf{w}^{\mathrm{T}}\phi(x))q(\mathbf{w})d\mathbf{w}$$

to calculate this, using delta approximation: (take $a = \mathbf{w}^{\mathrm{T}}\phi$)

$$\int \sigma(\mathbf{w}^{\mathrm{T}}\phi(x))q(\mathbf{w})d\mathbf{w} = \int \sigma(a)p(a)da$$
$$where \quad p(a) = \int \delta(a - \mathbf{w}^{\mathrm{T}}\phi)q(\mathbf{w})d\mathbf{w}$$

then we calculate mean and variance of $p(a)$:

$$\mu_a = \mathbb{E}[a] = \int ap(a)da = \int q(\mathbf{w})\mathbf{w}^{\mathrm{T}}\phi d\mathbf{w} = \mathbf{w}_{MAP}^{\mathrm{T}}\phi$$
$$\sigma_a^2 = var[a] = \int p(a)\{a^2 - \mathbb{E}[a]^2\}da = \int q(\mathbf{w})\{(\mathbf{w}^{\mathrm{T}}\phi)^2 - (\mathbf{m}_N^{\mathrm{T}}\phi)^2\}d\mathbf{w} = \phi^{\mathrm{T}}S_N\phi$$

Thus the approximation of posterior predictive distribution is:

$$p(t|x, D, \mathbf{m}_0, \alpha) \backsimeq \int \sigma(a)N(a|\mu_a, \sigma_a^2)da$$

Furthermore, we can use inverse probit function to get a more explicit approximation of $p(t|x, D, \mathbf{m}_0, \alpha)$:

$$p(t|x, D, \mathbf{m}_0, \alpha) \backsimeq \sigma(\kappa(\sigma_a^2)\mu_a)$$
$$where \quad \kappa(\sigma^2) = (1 + \frac{\pi\sigma^2}{8})^{-\frac{1}{2}}$$

The prior predictive distribution is:

$$p(D|\mathbf{m}_0, \alpha) = \int p(D|\mathbf{w})p(\mathbf{w})d\mathbf{w}$$

Using Laplace approximation:

$$-\ln p(D|\mathbf{w}) - \ln p(\mathbf{w}) = -\ln p(D|\mathbf{w}_N) - \ln p(\mathbf{w}_N) + \frac{1}{2}(\mathbf{w} - \mathbf{m}_N)^{\mathrm{T}}\Sigma_N^{-1}(\mathbf{w} - \mathbf{m}_N)$$

Then we derive

$$p(D|\mathbf{w}_N) = \prod_{n=1}^{N}[\sigma(\phi_n^{\mathrm{T}}\mathbf{m_N})]^{t_n}[1 - \sigma(\phi_n^T\mathbf{m}_N)]^{(1-t_n)}$$
$$\begin{aligned}\ln p(D|\mathbf{m_0}, \alpha) &= \ln \int p(D|\mathbf{w})p(\mathbf{w})d\mathbf{w} \\ &= \ln p(D|\mathbf{w_N}) + \ln p(\mathbf{w_N}) + \frac{M}{2}\ln 2\pi - \frac{1}{2}\ln|\Sigma_N| \\ &= \sum_{n=1}^{N}[t_n \ln \sigma(\phi_n^{\mathrm{T}}\mathbf{m_N}) + (1 - t_n)\ln(1 - \sigma(\phi_n^{\mathrm{T}}\mathbf{m_N}))] + \frac{NM}{2}\ln\alpha - \frac{1}{2}\ln|\Sigma_N| - \frac{\alpha}{2}(\mathbf{m}_N - \mathbf{m}_0)^{\mathrm{T}}(\mathbf{m}_N - \mathbf{m}_0)\end{aligned}$$

Thus, $p(D|\mathbf{m}_0, \alpha) = \exp(\ln p(D|\mathbf{m}_0, \alpha))$.

Problem 5. Neural Network (10 pts)

Consider a two-layer neural network described by following equations:

$$a_1 = \mathbf{w^{(1)}}\mathbf{x}, \quad a_2 = \mathbf{w^{(2)}}\mathbf{z}, \quad z = h(a_1), \quad y = \sigma(a_2)$$

where $\mathbf{x}$ and y are the input and output, respectively, of the neural network, $h(\cdot)$ is a nonlinear function, and $\sigma(\cdot)$ is the sigmod function.

(1)Please derive the following gradients: $\frac{\partial y}{\partial \mathbf{w}^{(1)}}$, $\frac{\partial y}{\partial \mathbf{w}^{(2)}}$, $\frac{\partial y}{\partial a_1}$, $\frac{\partial y}{\partial a_2}$, and $\frac{\partial y}{\partial \mathbf{x}}$.

Ans:

we know that

$$\begin{aligned}
\frac{\partial a_1}{\partial \mathbf{x}} &= \mathbf{w^{(1)}}, \quad \frac{\partial a_1}{\partial \mathbf{w^{(1)}}} = \mathbf{x} \\
\frac{\partial a_2}{\partial \mathbf{z}} &= \mathbf{w^{(2)}}, \quad \frac{\partial a_2}{\partial \mathbf{w^{(2)}}} = \mathbf{z} \\
\frac{\partial \mathbf{z}}{\partial a_1} &= h'(a_1), \quad \frac{\partial y}{\partial a_2} = y(1-y)
\end{aligned}$$

Then we derive the following gradients

$$\begin{aligned}
\frac{\partial y}{\partial \mathbf{w^{(1)}}} &= \frac{\partial y}{\partial a_2}\frac{\partial a_2}{\partial \mathbf{z}}\frac{\partial \mathbf{z}}{\partial a_1}\frac{\partial a_1}{\partial \mathbf{w^{(1)}}} = y(1-y)\mathbf{w}^{(2)}h'(a_1)\mathbf{x} \\
\frac{\partial y}{\partial \mathbf{w^{(2)}}} &= \frac{\partial y}{\partial a_2}\frac{\partial a_2}{\partial \mathbf{w}^{(2)}} = y(1-y)\mathbf{z} \\
\frac{\partial y}{\partial a_1} &= \frac{\partial y}{\partial a_2}\frac{\partial a_2}{\partial \mathbf{z}}\frac{\partial \mathbf{z}}{\partial a_1} = y(1-y)\mathbf{w}^{(2)}h'(a_1) \\
\frac{\partial y}{\partial a_2} &= y(1-y) \\
\frac{\partial y}{\partial \mathbf{x}} &= \frac{\partial y}{\partial a_2}\frac{\partial a_2}{\partial \mathbf{z}}\frac{\partial \mathbf{z}}{\partial a_1}\frac{\partial a_1}{\partial \mathbf{x}} = y(1-y)\mathbf{w}^{(2)}h'(a_1)\mathbf{w}^{(1)}
\end{aligned}$$

(2) Please derive the updating rules for $\mathbf{w^{(1)}}$ and $\mathbf{w^{(2)}}$ given the classification errors between y and t, where t is the ground truth of the output y.

Ans:

the classification errors function: (take it as binary-classification)

$$E = -[t\ln(y) + (1-t)\ln(1-y)]$$

Thus

$$\begin{aligned}
\frac{\partial E}{\partial \mathbf{w^{(1)}}} &= \frac{\partial E}{\partial y}\frac{\partial y}{\partial \mathbf{w^{(1)}}} = (-\frac{t}{y} + \frac{1-t}{1-y})y(1-y)\mathbf{w}^{(2)}h'(a_1)\mathbf{x} = (y-t)\mathbf{w}^{(2)}h'(a_1)\mathbf{x} \\
\frac{\partial E}{\partial \mathbf{w^{(2)}}} &= \frac{\partial E}{\partial y}\frac{\partial y}{\partial \mathbf{w^{(2)}}} = (-\frac{t}{y} + \frac{1-t}{1-y})y(1-y)\mathbf{z} = (y-t)\mathbf{z}
\end{aligned}$$

So, $\mathbf{w^{(1)}} \leftarrow \mathbf{w^{(1)}} - \eta(y-t)\mathbf{w}^{(2)}h'(a_1)\mathbf{x}$ and $\mathbf{w^{(2)}} \leftarrow \mathbf{w^{(2)}} - \eta(y-t)\mathbf{z}$, where η is the learning rate.

Problem 6. Bayesian Neural Network (20 pts)

a) Consider a neural network for regression, $t = y(\mathbf{w}, \mathbf{x}) + v$, where v is Gaussian, i.e., $v \sim N(v|0, \beta^{-1})$, and $\mathbf{w}$ has a Gaussian priori, i.e., $\mathbf{w} \sim N(\mathbf{w}|\mathbf{m}_0, \alpha^{-1}\mathbf{I})$. Assume that $y(\mathbf{w}, \mathbf{x})$ is the neural network output please derive the posterior distribution, $p(\mathbf{w}|D, \beta, \mathbf{m}_0, \alpha)$, the posterior predictive distribution, $p(t|x, D, \beta, \mathbf{m}_0, \alpha)$, and the prior predictive distribution, $p(D|\beta, \mathbf{m}_0, \alpha)$, where $D = \{x_n, t_n\}$, $n = 1, \ldots, N$, is the training data set.

Ans:

The posterior distribution is:

$$\begin{aligned}
p(\mathbf{w}|D, \beta, \mathbf{m}_0, \alpha) &\propto p(\mathbf{w})p(D|\mathbf{w}) \\
where \quad p(D|\mathbf{w}) &= \prod_{n=1}^{N} N(t_n|y(x_n, \mathbf{w}), \beta^{-1}) \qquad (p(D|\mathbf{w}) \text{ is the likelihood})
\end{aligned}$$

We also use the Laplace approximation to find a Gaussian approximation of $p(\mathbf{w}|D, \beta, \mathbf{m}_0, \alpha)$, first we need to maximize $\ln p(\mathbf{w}|D, \beta, \mathbf{m}_0, \alpha)$, where
$\ln p(\mathbf{w}|D, \beta, \mathbf{m}_0, \alpha) = -\frac{\alpha}{2}(\mathbf{w} - \mathbf{m_0})^{\mathrm{T}}(\mathbf{w} - \mathbf{m_0}) - \frac{\beta}{2}\sum_{n=1}^{N}\{y(\mathbf{x_n}, \mathbf{w}) - t_n\}^2 + const$, to find $\mathbf{w}_{MAP}$, which is the mean of Gaussian distribution. The error function $E(\mathbf{w}) = -\ln p(\mathbf{w}|D, \beta, \mathbf{m}_0, \alpha)$.

$$E(\mathbf{w}) = \frac{\alpha}{2}(\mathbf{w}-\mathbf{m_0})^{\mathrm{T}}(\mathbf{w}-\mathbf{m_0}) + \frac{\beta}{2}\sum_{n=1}^{N}\{y(\mathbf{x_n},\mathbf{w}) - t_n\}^2 + const$$

$$\nabla E(\mathbf{w}) = \alpha(\mathbf{w}-\mathbf{m_0}) + \beta\sum_{n=1}^{N}(y_n - t_n)g_n, where\ y_n = y(\mathbf{x_n},\mathbf{w}), g_n = \nabla_{\mathbf{w}} y_n$$

$$A = \nabla\nabla E(\mathbf{w}) = \alpha\mathbf{I} + \beta\mathbf{H}, where\ \mathbf{H}\ is\ Hessian\ matrix\ of\ the\ sum-of-error\ function$$

Then we use $\mathbf{w}_{MAP} \rightarrow \mathbf{w}_{new} = \mathbf{w}_{old} - A^{-1}\nabla E(\mathbf{w})$ to find the $\mathbf{w}_{MAP}$.And the Gaussian approximation posterior distribution is:

$$q(\mathbf{w}) = N(\mathbf{w}|\mathbf{w}_{MAP}, A^{-1})$$

The posterior predictive distribution is:

$$p(t|x, D, \beta, \mathbf{m}_0, \alpha) = \int p(t|x,\mathbf{w})q(\mathbf{w}|D)d\mathbf{w}$$

As the relationship between $y(x,\mathbf{w})$ and $\mathbf{w}$ is non-linear, it's impossible to get a analytical solution of $p(t|x, D, \beta, \mathbf{m}_0, \alpha)$. So, we first do Taylor expansion for $y(x,\mathbf{w})$ near $\mathbf{w}_{MAP}$:

$$y(x,\mathbf{w}) \backsimeq y(x,\mathbf{w}_{MAP}) + g^{\mathrm{T}}(\mathbf{w}-\mathbf{w}_{MAP}), where\ g = \nabla_{\mathbf{w}} y(x,\mathbf{w})|_{\mathbf{w}=\mathbf{w}_{MAP}}$$

Then get a linear Gaussian distribution of $p(t|x,\mathbf{w})$

$$p(t|x,\mathbf{w}) \backsimeq N(t|y(x,\mathbf{w}_{MAP}) + g^{\mathrm{T}}(\mathbf{w}-\mathbf{w}_{MAP}), \beta^{-1})$$

Then we get

$$p(t|x, D, \beta, \mathbf{m}_0, \alpha) = N(t|y(x,\mathbf{w}_{MAP}), \sigma^2(x))$$

where $\sigma^2(x) = \beta^{-1} + g^{\mathrm{T}}A^{-1}g$.

The prior predictive distribution is:

$$p(D|\beta, \mathbf{m}_0, \alpha) = \int p(D|\mathbf{w},\beta)p(\mathbf{w})d\mathbf{w}$$

Using Laplace approximation like in problem4, we derive:

$$\ln p(D|\beta, \mathbf{m}_0, \alpha) \backsimeq -\frac{\beta}{2}\sum_{n=1}^{N}\{y(x_n,\mathbf{w}_{MAP}) - t_n\}^2 - \frac{\alpha}{2}(\mathbf{w}_{MAP}-\mathbf{m}_0)^{\mathrm{T}}(\mathbf{w}_{MAP}-\mathbf{m}_0)$$
$$-\frac{1}{2}\ln|A| + \frac{MN}{2}\ln\alpha + \frac{N}{2}\ln\beta - \frac{N}{2}\ln 2\pi$$

b) Consider a neural network for two-class classification, $y = \sigma(a(\mathbf{w},\mathbf{x}))$ and a data set $D = \{x_n, t_n\}$, where $t_n \in \{0,1\}$, $\mathbf{w}$ has a Gaussian priori, i.e., $\mathbf{w} \sim N(\mathbf{w}|\mathbf{0}, \alpha^{-1}\mathbf{I})$, and $a(\mathbf{w},\mathbf{x})$ is the neural network model. Please derive the posterior distribution, $p(\mathbf{w}|D,\alpha)$, posterior predictive distribution, $p(t|\mathbf{x}, D, \alpha)$, and the prior predictive distribution, $p(D|\alpha)$, respectively.

Ans:

The posterior distribution is:

$$p(\mathbf{w}|D,\alpha) \propto p(\mathbf{w})p(D|\mathbf{w})$$

We also use the Laplace approximation to find a Gaussian approximation of $p(\mathbf{w}|D,\alpha)$, first we need to maximize $\ln p(\mathbf{w}|D,\alpha)$, where $\ln p(\mathbf{w}|D,\alpha) = -\frac{\alpha}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w} + \sum_{n=1}^{N}[t_n \ln y_n + (1-t_n)\ln(1-y_n)]$, to find $\mathbf{w}_{MAP}$, which is the mean of Gaussian distribution. The error function $E(\mathbf{w}) = -\ln p(\mathbf{w}|D,\alpha)$.

$$E(\mathbf{w}) = \frac{\alpha}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w} - \sum_{n=1}^{N}[t_n \ln y_n + (1-t_n)\ln(1-y_n)]$$

$$\nabla E(\mathbf{w}) = \alpha\mathbf{w} + \sum_{n=1}^{N}(y_n - t_n)g_n, where\ y_n = y(\mathbf{x_n},\mathbf{w}), g_n = \nabla_{\mathbf{w}} y_n$$

$$A = \nabla\nabla E(\mathbf{w}) = \alpha\mathbf{I} + \mathbf{H}, where\ \mathbf{H}\ is\ Hessian\ matrix\ of\ the\ cross-entropy\ function$$

Then we use $\mathbf{w}_{MAP} \rightarrow \mathbf{w}_{new} = \mathbf{w}_{old} - A^{-1}\nabla E(\mathbf{w})$ to find the $\mathbf{w}_{MAP}$.And the Gaussian approximation posterior distribution is:

$$q(\mathbf{w}) = N(\mathbf{w}|\mathbf{w}_{MAP}, A^{-1})$$

The posterior predictive distribution is:

$$p(t|\mathbf{x}, D, \alpha) = \int p(t|\mathbf{x}, \mathbf{w})q(\mathbf{w}|D)d\mathbf{w}$$

It's reasonable to do linear approximation on activate function $a(x, \mathbf{w})$:

$$a(x, \mathbf{w}) \backsimeq a_{MAP}(x) + \mathbf{b}^{\mathrm{T}}(x)(\mathbf{w} - \mathbf{w}_{MAP})$$
$$where \quad a_{MAP}(x) = a(x, \mathbf{w}_{MAP}), \mathbf{b} = \nabla a(x, \mathbf{w}_{MAP})$$

Then we get the distribution of a is :

$$p(a|x, D) = \int \delta(a - a_{MAP}(x) - \mathbf{b}^{\mathrm{T}}(x)(\mathbf{w} - \mathbf{w}_{MAP}))q(\mathbf{w}|D)d\mathbf{w}$$

which is also a Gaussian distribution from what we have learned, whose mean is $a_{MAP} = a(x, \mathbf{w}_{MAP})$ and variance is $\sigma_a^2(x) = \mathbf{b}^{\mathrm{T}}(x)A^{-1}\mathbf{b}(x)$.

Last we use the Gaussian approximation distribution we obtained to get the posterior predictive distribution:

$$p(t = 1|\mathbf{x}, D, \alpha) = \int \sigma(a)p(a|\mathbf{x}, D)da$$

Furthermore, we can use inverse probit function to get a more explicit approximation of $p(t = 1|\mathbf{x}, D, \alpha)$

$$p(t = 1|\mathbf{x}, D, \alpha) = \sigma(\kappa(\sigma_a^2)a_{MAP})$$
$$where \quad \kappa(\sigma^2) = (1 + \frac{\pi\sigma^2}{8})^{-\frac{1}{2}}$$

The prior predictive distribution is:

$$p(D|\alpha) = \int p(D|\mathbf{w})p(\mathbf{w})d\mathbf{w}$$

Using Laplace approximation like in problem4, we derive:

$$\ln p(D|\alpha) \backsimeq -\frac{\alpha}{2}\mathbf{w}_{\mathbf{MAP}}^{\mathrm{T}}\mathbf{w}_{\mathbf{MAP}} + \sum_{n=1}^{N}[t_n \ln y_n + (1 - t_n)\ln(1 - y_n)] - \frac{1}{2}\ln|A| + \frac{NM}{2}\ln\alpha$$

Finally, the prior predictive distribution $p(D|\alpha) = \exp(\ln p(D|\alpha))$.

Problem 7. Critical Analyses (20 pts)

a) Please explain why the dual problem formulation is used to solve the SVM machine learning problem.

Ans: The SVM machine learning problem is $\underset{w,b}{min}\underset{\alpha}{max}L(w, b, \alpha)$, where $L(w, b, \alpha) = \frac{1}{2}||w||^2 - \sum_{i=1}^{n}\alpha_i(y_i(w^{\mathrm{T}}x_i + b) - 1)$. And it's a convex optimization problem. It can be time-consuming on large-scale datasets as it involves a number of variables equal to the number of features. And the equivalent dual problem formulation of it is $\underset{\alpha}{max}\underset{w,b}{min}L(w, b, \alpha)$, which is a concave optimization problem. In dual problem, data only appears in the form of dot products. This allows us to use the so-called "Kernel Trick" to map data into a high-dimensional space, thereby solving problems that are not linearly separable in the original space and the number of variables involved is equal to the number of samples, which is usually much smaller than the number of features.

b) Please explain, in terms of cost functions, constraints and predictions,

- **i) what are the differences between SVM classification and logistic regression;**
- **Ans:**
    - **Cost Function**: The cost function of Logistic Regression is the log loss, which is a smooth function that gives a probabilistic interpretation to the outputs. On the other hand, SVM uses the hinge loss, which does not output probabilities and is not differentiable at zero. This makes SVM more robust to outliers compared to Logistic Regression.
    - **Constraints**: SVM tries to find the hyperplane that maximizes the margin between the closest points of different classes (these points are called support vectors). Logistic Regression does not have this property.

    - **Predictions**: Logistic Regression outputs probabilities, which can be thresholded to get class predictions. SVM directly outputs class predictions and does not naturally provide probability estimates.

- **ii) what are the differences between v-SVM regression and least square regression.**

- **Ans:**

    - **Cost Function**: Least Square Regression minimizes the sum of squared residuals, which makes it sensitive to outliers. v-SVR uses the ε-insensitive loss function, which ignores errors that are within a certain distance ε from the true value, making it more robust to outliers.
    - **Constraints**: v-SVR tries to find a function that fits most of the data points within a specified margin, while allowing some violations (points outside the margin). These violations are penalized based on their distance from the margin. The parameter v controls the number of support vectors and training errors, providing an upper bound on the fraction of margin errors and a lower bound on the fraction of support vectors. Least Square Regression does not have this property.
    - **Predictions**: Both methods output continuous values. However, due to the different cost functions and constraints, the predictions can be quite different, especially for datasets with outliers. In particular, v-SVR can control the trade-off between model complexity (number of support vectors) and the amount of tolerable errors, which can be beneficial in situations where a certain amount of error is acceptable.

c) Please explain why neural network (NN) based machine learning algorithms use logistic activation functions ?

Ans: Neural networks use activation functions to introduce non-linearity into the model. And logistic activation function is non-linear. Besides, it outputs a value between 0 and 1, which can be interpreted as probability and used in classification tasks. In addition, it's differentiable everywhere, which is important for gradient-based optimization methods like backpropagation.

d) Please explain

- **i) what are the differences between the logistic activation function and other activation functions (e.g., relu, tanh)**

    - Ans:

        - **Logistic Activation Function**: The output of the sigmoid function ranges from 0 to 1, making it particularly useful in the output layer of binary classification problems, where we can interpret the output as a probability. However, the sigmoid function has a gradient close to 0 when the absolute value of its input is large, which can lead to the vanishing gradient problem, making the network difficult to learn.
        - **ReLU Function**: The ReLU function outputs 0 when the input is less than 0, and outputs the input itself when the input is greater than 0. This allows the ReLU function to maintain the linear properties of the input when dealing with positive values, while completely suppressing the input when dealing with negative values. This characteristic of the ReLU function makes it very effective in many deep learning models. However, the gradient of the ReLU function is 0 when the input is less than 0, which can lead to "dead neurons", i.e., they may stop learning during training.
        - **Tanh Function**: The output of the tanh function ranges from -1 to 1, making its output more balanced between negative and positive values. However, like the sigmoid function, the tanh function has a gradient close to 0 when the absolute value of its input is large, which can lead to the vanishing gradient problem.
        - **Softmax**: The softmax function is often used in the output layer of a multi-class classification problem. It outputs a vector that represents the probability distributions of a list of potential outcomes. It's a generalization of the sigmoid function that can handle more than two classes.

- **ii) when these activation functions should be used**

    - Ans:

        - **Logistic Activation Function**: The sigmoid function is often used in the output layer of a binary classification problem where the output is interpreted as a probability of belonging to the positive class.
        - **ReLU**: The ReLU function is the most commonly used activation function in the hidden layers of neural networks for a wide range of tasks. It helps to alleviate the vanishing gradient problem and is computationally efficient.
        - **Tanh**: The tanh function can be used in the hidden layers of a neural network when the output needs to be normalized to range between -1 and 1.
        - **Softmax**: The softmax function is often used in the output layer of a multi-class classification problem where it provides a probability distribution over N different possible outcomes.

e) Please explain why Jacobian and Hessian matrices are useful for machine learning algorithms.

Ans:

- **Jacobian Matrix**: The Jacobian matrix is a matrix of all first-order partial derivatives of a vector-valued function. It provides us with information about the gradient of a function at a particular point. In machine learning, we often need to minimize a cost function, and the Jacobian matrix helps us understand the direction of steepest ascent or descent at a given point. This is crucial for optimization algorithms like gradient descent.
- **Hessian Matrix**: The Hessian matrix is a square matrix of second-order partial derivatives of a function. It gives us information about the curvature of the function. In the context of optimization, the Hessian can be used to determine whether a critical point is a local maximum, local minimum, or a saddle point. This is useful in second-order optimization methods like Newton's method. Furthermore, the Hessian is used in determining the convergence rates of optimization algorithms.

f) Please explain why exponential family distributions are so common in engineering practice. Please give some examples which are NOT exponential family distributions.

Ans: Exponential family distributions are practical in engineering practice due to the following reasons:

- **Simplicity**: Exponential family distributions have a simple and consistent mathematical form, which makes them easy to work with.
- **Conjugate Priors**: In Bayesian statistics, exponential family distributions have conjugate priors. This means that if the prior and the likelihood are in the same distribution family, the posterior distribution is also in that family, simplifying the computation.
- **Sufficient Statistics**: Exponential family distributions have the property of sufficiency, meaning that all the information in the data about the parameters of the distribution can be summarized in a fixed-size set of statistics.
- **Versatility**: The exponential family includes many of the most common distributions used in practice, such as the normal, exponential, Bernoulli, Poisson, and gamma distributions. This makes them versatile for modeling different types of data.

The followings are some distributions which are **NOT** in the exponential family: Cauchy Distribution, F-Distribution, Uniform Distribution.

g) Please explain why KL divergence is useful for machine learning? Please provide two examples of using KL divergence in machine learning

Ans: KL divergence is a measure of how one probability distribution diverges from a second, expected probability distribution. It can be used as a loss function to measure the difference between the predicted probability distribution and the true distribution. Minimizing the KL divergence helps to improve the model's predictions. Besides, KL divergence can be used to compare different models. The model that results in the smallest KL divergence from the true distribution is considered the best.Two examples of using KL divergence: Variational Autoencoders(VAE), t-SNE algorithm.

h) Please explain why data augmentation techniques are a kind of regularization skills for NNs.

Ans: It helps the model generalize better to unseen data, thus preventing overfitting. And by creating variations of the training data, data augmentation techniques force the model to learn more robust features. Besides, data augmentation effectively increases the size of the training dataset and prevents overfitting by providing more training examples, making it harder for the model to memorize the training data and thus encouraging it to learn more general patterns.

i) Please explain why Gaussian distributions are preferred over other distributions for many machine learning models?

Ans: Gaussian distributions are often preferred in many machine learning models due to a combination of factors. The Central Limit Theorem, which states that the sum of a large number of independent and identically distributed random variables tends towards a Gaussian distribution, makes the Gaussian distribution a natural choice for many real-world phenomena. Additionally, Gaussian distributions are mathematically tractable, meaning they have desirable properties when it comes to calculus operations. For instance, Gaussians remain Gaussians under linear transformations and the convolution of two Gaussian distributions is another Gaussian, making them easier to work with in many machine learning algorithms. Furthermore, Gaussian distributions are parameter efficient as they are fully described by just two parameters - the mean and variance, making them very efficient in terms of encapsulating information about the data. Lastly, many natural phenomena follow a Gaussian distribution due to the accumulation of many small, independent effects, making Gaussian distributions a good first assumption in many cases.

j) Please explain why Laplacian approximation can be used for many cases?

Ans: The Laplacian approximation is widely used because it simplifies complex problems by approximating a difficult-to-handle distribution with a Gaussian distribution, which has desirable mathematical properties. This method is computationally efficient as it only requires the calculation of the mode and the curvature at the mode, making it quick even for high-dimensional problems. Furthermore, its versatility allows it to be applied to a broad range of problems, not limited to specific types of distributions or models.

k) What are the fundamental principles for model selection (degree of complexity) in machine learning?

Ans: The fundamental principles for model selection, or determining the degree of complexity in machine learning, revolve around finding a balance between bias and variance, also known as the bias-variance tradeoff. Bias refers to the error introduced by approximating a real-world problem, which may be extremely complicated, by a much simpler model. High bias can cause a model to miss relevant relations between features and target outputs (underfitting).Variance, on the other hand, is the amount by which our model would change if we estimated it using a different training dataset. High variance can cause a model to model the random noise in the training data, rather than the intended outputs (overfitting). Other principles include Cross-Validation, Regularization, Occam's Razor and so on.

l) How to choose a new data sample (feature) for regression and classification model training, respectively? How to choose it for testing? Please provide some examples.

Ans: When choosing new data sample(feature) during training, we should make sure the features should be relevant to the problem and be independent of each other, as multicollinearity can make the model unstable and the results hard to interpret. And for testing, you should ideally choose a set of data that the model hasn't seen during training. This is often done by splitting the original dataset into a training set and a test set.

- Regression: If you're predicting the price of a house, you might choose features like the size of the house, the number of rooms, the age of the house, the location, etc. You would then split your dataset into a training set (e.g., 80% of the data) and a test set (e.g., 20% of the data).
- Classification: If you're building a spam classifier, you might choose features like the length of the email, the number of capital letters, whether it contains certain keywords, etc. Again, you would split your dataset into a training set and a test set.

m) Please explain why the MAP model is usually more preferred than the ML model?

Ans: MAP allows for the inclusion of prior knowledge about the parameters through a prior distribution. This is especially useful when data is limited, as the prior can guide the estimation towards more plausible values. Additionally, the prior distribution in MAP estimation can act as a regularizer, helping to prevent overfitting, a common problem in ML estimation. MAP estimation can also be more robust than ML, particularly with small datasets, as ML estimates can be overly influenced by the data and lead to overfitting.

Problem 8. Discussions (10pts)

(1) What are the generative and discriminative approaches to machine learning, respectively? Can you explain the advantages and disadvantages of these two approaches and provide a detailed example to illustrate your points?

Ans:

- generative approaches: to find the conditional probability $P(Y|X)$, they estimate the prior probability $P(Y)$ and likelihood probability $P(X|Y)$ with the help of the training data and use the Bayes Theorem to calculate the posterior probability $P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)}$. They are good at handling missing data and require less data, but make strong assumptions about data distribution. Example: In a spam detection problem, a generative model like Naive Bayes would learn the distribution of words in spam and non-spam emails ($P(X|Y)$) and the overall probabilities of spam and non-spam emails ($P(Y)$). It could then classify a new email by seeing which class is more likely to have generated it.
- discriminative approaches: to find the probability, they directly assume some functional form for $P(Y|X)$ and then estimate the parameters of $P(Y|X)$ with the help of the training data. They can model complex relationships and are often more accurate, but require more data and can't handle missing values as naturally. Example: In the spam detection problem, a discriminative model like Logistic Regression would learn the boundary between spam and non-spam emails based on the words in the emails. It would classify a new email by seeing which side of the boundary it falls on.

(2) How do you analyze the GAN model from the generative and discriminative perspectives?

Ans: GAN model is a type of generative model that use both generative and discriminative components to generate new data, which consists of two parts: a generator and a discriminator. The generator's job is to generate new data and the discriminator's job is to distinguish between real data (from the training set) and fake data (from the generator). The two models are trained together in a game-theoretic framework, where the generator tries to fool the discriminator and the discriminator tries to correctly classify real vs. fake. The competition between these two forces leads the generator to produce increasingly realistic data. From the generative perspective, GANs are powerful because they can learn to mimic any distribution of data, meaning they can generate new data that's very similar to the training data. This makes them useful for tasks like image synthesis, super-resolution, and image-to-image translation. From the discriminative perspective, the discriminator in a GAN learns to accurately distinguish real data from fake, which can be a useful skill in itself.