Machine Learning(H) Midterm Exam

SID: 12012919

Name: 廖铭骞

Problem I. Least Square (15 points)

1)

$$
\begin{aligned}
Y = AX + V, V \sim N(v|0, Q) \\
E(X) = \frac{1}{2}(y - AX)^T Q^{-1}(y - AX) \\
\frac{\partial E(X)}{\partial X} = 0 \Rightarrow A^T Q^{-1}(y - AX^*) = 0 \\
\Rightarrow X^* = (A^T Q^{-1} A)^{-1} A^T Q^{-1} y
\end{aligned} \tag{1}
$$

2)

The cost function of 2) is $E_1 + E_2$

which is

$$
\frac{1}{2}(y - AX)^T Q^{-1}(y - AX) + \lambda(bX - c) \tag{2}
$$

$$
\Rightarrow \hat{X} = X - (A^T A) b^T [b(A^T A)^{-1} b^T]^{-1} (b\hat{X} - c) \tag{3}
$$

Where $X = (A^T \Sigma^{-1} A)^{-1} A^T \Sigma^{-1} Y$ is the solution in (1), which also means:

$$\hat{X} = (A^T\Sigma^{-1}A)^{-1}A^T\Sigma^{-1}Y - (A^TA)b^T[b(A^TA)^{-1}b^T]^{-1}\hat{b}$$
$$((A^T\Sigma^{-1}A)^{-1})A^T\Sigma^{-1}Y) - c \tag{4}$$

3)

$$min(AX-Y)^T(AX-Y)$$
$$s.t.\, bX - c = 0 \;\; X^TX = d$$
$$L(X)\cdot(AX-Y)^T(AX-Y) - \lambda(bX-c) - N(X^TX-d) \tag{5}$$
$$\Rightarrow X_{optimal} = (2A^TA - 2DI)^{-1}(2A^TY - \lambda b)$$

Problem II. Linear Gaussian System (10 points)

Problem II. Linear Gaussian System (10 points)

Consider $Y = AX + V$, where X and V are Gaussian, $X \sim \mathcal{N}(\boldsymbol{x}|\boldsymbol{m}_0, \boldsymbol{\Sigma}_0)$, $V \sim \mathcal{N}(\boldsymbol{v}|\boldsymbol{0}, \boldsymbol{\beta}^{-1}\boldsymbol{I})$.

What are the conditional distribution, $p(Y \mid X)$, the joint distribution $p = (Y, X)$, the marginal distribution, $p(Y)$, the posterior distribution, $p(X|Y = \boldsymbol{y}, \boldsymbol{\beta}, \boldsymbol{m}_0, \boldsymbol{\Sigma}_0)$, and the posterior predictive distribution, $p\left(\hat{Y}\middle|Y = \boldsymbol{y}, \boldsymbol{\beta}, \boldsymbol{m}_0, \boldsymbol{\Sigma}_0\right)$, respectively?

$$P(Y|X) = N(Y|AX, \beta^{-1}I)$$
$$p(X,Y) = P(X)\cdot P(Y|X) = N(X|m_0, \Sigma_0)\cdot N(y|AX, \beta^{-1}I)$$
$$P(Y) = N(Y|Am_0, \beta^{-1}I + A\Sigma_0A^T)$$
$$P(X|Y) = N(X|\Sigma(A^T(\beta I)Y + \Sigma_0^{-1}m_0), \Sigma) \tag{6}$$
$$where \Sigma = (\Sigma_0^{-1} + A^T(\beta I)A)^{-1}$$
$$P(\hat{Y}|Y) = N(\hat{Y}|A\Sigma(A^T(\beta I)Y + \Sigma_0^{-1}m_0), \beta^{-1}I + A\Sigma A^T)$$

Problem III. Linear Regression (10 points)

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

Posterior distribution:

$$
\begin{aligned}
p(t|t, \alpha, \beta) &= \int p(t|w, \beta) p(w|t, \alpha, \beta) dw \\
p(t|x, t, \alpha, \beta) &= N(t|m_N^T \emptyset(x), \sigma_N^2(x)) \\
\sigma_N^2(x) &= \frac{1}{\beta} + \emptyset(x)^T S_N \emptyset(x)
\end{aligned} \tag{7}
$$

Posterior predictive distribution:

Regarding the discussion of the Bayesian method of linear fitting, we first introduce the prior distribution probability of the model parameter w. At this time, we regard the noise precision parameter β as a known parameter (if unknown, the Gauss-Gamma distribution can be introduced as a priori distribution) :

$$
p(w) = N(w|m_0, S_0) \tag{8}
$$

Next, we compute the posterior distribution, which is proportional to the product of the likelihood function and the prior distribution:

$$
\begin{aligned}
p(w|t) = N(w|m_N, S_N), where \ \ m_N &= S_N(S_0^{-1} m_0 + \beta \emptyset^T t) \\
S_N^{-1} &= S_0^{-1} + \beta \emptyset^T \emptyset
\end{aligned} \tag{9}
$$

For simplicity, we will consider a specific form of the Gaussian prior. Specifically, we consider a zero-mean isotropic Gaussian distribution governed by a precision parameter α, namely:

$$
p(w|\alpha) = N(w|0, \alpha^{-1} I) \tag{10}
$$

where

$$
\begin{aligned}
m_N &= \beta S_N \emptyset^T t \\
S_N^{-1} &= \alpha I + \beta \emptyset^T \emptyset
\end{aligned}
\tag{11}
$$

Problem IV. Logistics Regression (10 points)

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

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

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

First, select the Gaussian distribution as the prior probability, and then calculate the posterior probability distribution of w, taking the logarithm:

$$
\begin{aligned}
&\ln p(w|t) = -\frac{1}{2}(w - w_0)^T S_0^{-1}(w - w_0) + \\
&\sum_{n=1}^{N}\{t_n \ln y_n + (1 - t_n)\ln(1 - y_n)\} + const
\end{aligned}
\tag{12}
$$

To obtain a Gaussian approximation of the posterior probability, we first maximize the posterior probability distribution to obtain the MAP solution w_{MAP}, which defines the mean of the Gaussian distribution. Thus the covariance matrix is the inverse matrix of the second-order derivative matrix of the negative log-likelihood function, in the form

$$S_N^{-1} = -\nabla\nabla \ln p(w|t) = s_0^{-1} + \sum_{n=1}^{N} y_n(1-y_n)\emptyset_n\emptyset_n^T \tag{13}$$

and we can get the Gaussian similarity of posterior probability:

$$q(w) = N(w|w_{MAP}, S_N) \tag{14}$$

Finally, we can get a similar form of predictive distribution:

$$p(C_1|t) = \int \sigma(a)p(a)da = \int \sigma(a)N(a|\mu_a, \sigma_a^2)da \tag{15}$$

Problem V. Neural Network (10 points)

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

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

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

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

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

(1)

$$\begin{aligned}
\frac{\partial y}{\partial a_2} &= \frac{\partial \sigma(a_2)}{\partial a_2} = y(1-y) \\
\frac{\partial y}{\partial w^{(2)}} &= \frac{\partial y}{\partial a_2}\frac{\partial a_2}{\partial w^{(2)}} = y(1-y)z \\
\frac{\partial y}{\partial a_1} &= \frac{\partial y}{\partial a_2}\frac{\partial a_2}{\partial z}\frac{\partial z}{\partial a_1} = y(1-y)w^{(2)}h^{'}(a_1) \\
\frac{\partial y}{\partial w^{(1)}} &= \frac{\partial y}{\partial a_2}\frac{\partial a_2}{\partial z}\frac{\partial z}{\partial a_1}\frac{\partial a_1}{\partial w^{(1)}} = y(1-y)w^{(2)}h^{'}(a_1)x \\
\frac{\partial y}{\partial x} &= \frac{\partial y}{\partial a_2}\frac{\partial a_2}{\partial z}\frac{\partial z}{\partial a_1}\frac{\partial a_1}{\partial x} = y(1-y)w^{(2)}h^{'}(a_1)w^{(1)}
\end{aligned} \tag{16}$$

(2)

$$
\begin{aligned}
\frac{\partial E}{\partial y} &= \frac{1}{2}\frac{\partial(y^2 - 2yt + t^2)}{\partial y} = y - t \\
\nabla w^{(2)} &= -\alpha\frac{\partial E}{\partial w^{(2)}} = -\alpha\frac{\partial E}{\partial y}\frac{\partial y}{\partial w^{(1)}} \\
&= -\alpha(y-t)y(1-y)w^{(2)}h^{'}(a_1)x \\
\nabla w^{(1)} &= -\alpha\frac{\partial E}{\partial w^{(1)}} = -\alpha\frac{\partial E}{\partial y}\frac{\partial y}{\partial w^{(1)}} \\
&= -\alpha(y-t)y(1-y)w^{(2)}h^{'}(a_1)x
\end{aligned}
\tag{17}
$$

Problem VI. Bayesian Neural Network (20 points)

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

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

a) The regression of the network:

$$
\begin{aligned}
P(t|w, x, \beta) &= N(t|y(x, w), \beta^{-1}) \\
p(w) &= N(w|0, \alpha^{-1}T) \\
P(w|t) &\propto P(w) \cdot P(t|w, x, \beta)
\end{aligned}
\tag{18}
$$

$$E = regularization + squreLoss = \frac{\alpha}{2}w^T w + \frac{\beta}{2}\sum\{y(x_n, w) - t_n\}^2$$

$$\nabla E(w) = \alpha w + \beta\sum(y_n - t_n)g_n, \quad g_n = \nabla_w y(x, w)$$

$w_{MAP} = w^{dd} - (\nabla\nabla E(w))^{-1}\nabla E(w),\ \ q(w) = N(w|w_{MAP}, (\nabla^2 E)^{-1})$

b) The classification of the network:

$E = regularization + crossEntrophy = \frac{\alpha}{2}w^T w - \sum[t_n|ny_n + (1-t_n)\ln(1-y_n)]$

$\nabla E = \alpha w + \sum(y_n - t_n)g_n$

$A = \nabla^2 E$

Finally, $w_{new} = w_{old} - A^{-1}\nabla E \to w_{MAP}$

Problem VII. Critical Analyses (20 Points)

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

Since at the last of the deduction process, we can get

$$L(w, b, a) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{n} a_i(y_i(w^T \cdot x_i + b) - 1) \tag{19}$$

where a is a new parameter brought by the Lagrange multiplier method.

Next, the question becomes

$$\min_{w,b}\max_{a} L(w, b, a) \tag{20}$$

To make the following calculation more convenient, we convert the formula to its dual form,

$$\max_{a}\min_{w,b} L(w, b, a) \tag{21}$$

Because after the dual conversion, we can use the kernel function to deal with the data.

b) Please explain, in terms of cost functions, constraints, and predictions, i) what are the differences between SVM classification and logistic regression; ii) what are the differences between v-SVM regression and least square regression.

i) The difference in the cost function lies in that, the logistic regression uses the log loss while the SVM uses the hinge loss;

The difference in the constraints lies in that, the cost function brings a regularizer own to serve as constraints while the logistic function should add a regularizer to constraints the result. Apart from that, the support vector machine only considers the points near the local boundary line, while the logistic regression considers all the points.

The difference in the predictions lies in that, changing non-support vector samples in SVM will not cause changes in the decision-making surface, while changing any samples in logistic regression will cause changes in the decision-making surface. And the SVM uses the kernel function to calculate the decision plane.

ii)In ordinary least square regression, to find the line of best fit, we use the L2 (squared loss) function, which finds a line with minimum distances from the observations. In Linear-SVR, we use epsilon insensitive L1 loss function, i.e if observations are within the threshold of epsilon produced no error, only the observation outside of the epsilon range produces an error.

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

Because the logistic activation functions are non-linear, the combination of a linear function is still a linear function, which will make the superposition of neural network layers becomes meaningless, and the accuracy of classification cannot be further improved.

d) Please explain i) what are the differences between the *logistic* activation function and other activation functions (e.g., $relu$, *tanh*), and ii) when these activation functions should be used.

i) The logistic activation function is differentiable and monotonic, and its value varies from 0 to 1.

ii)When you use the neural network to get a non-linear output such as in a classification task, the activation function should be used.

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

Introduce the F-norm of the Jacobian matrix to make the learned features locally invariant. And in machine learning optimization, after we converge to a critical point using gradient descent, we need to check the Hessian eigenvalue to determine whether it is a minimum, maximum, or saddle point. Studying the properties of the eigenvalues can tell us about the convexity of a function.

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

So in most cases, we will artificially specify a certain form of probability distribution (such as Gaussian distribution or Bernoulli distribution, etc.). In this way, the learning of the probability function is transformed into the learning of function parameters, which reduces the difficulty of learning; we only need to store the statistics we are interested in (for example, for Gaussian distribution, we only need to store the mean and variance; for Bernoulli Profit distribution, we only need to store the probability of taking a positive class), which reduces the demand for storage space. Of course, since the probability distribution form is artificially limited, we need to choose different distributions according to different problems, just like choosing different machine learning models for different problems.

The exponential family distribution is a commonly used distribution model, which has many excellent properties.

Examples:

Uniform distribution

Cauchy distribution

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

Used to measure the similarity of two probability distributions.

In the Spam Classification Task or the ImageNet competition, the KL divergence can be used to measure the similarity of two distributions.

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

Regularization techniques prevent overfitting in networks with more parameters than input data. Regularization helps the algorithm generalize by avoiding training coefficients that perfectly fit the data samples. To prevent overfitting, increasing training samples is a good solution, and data enhancement can achieve this goal by increasing training samples.

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

Because the mean of the Gaussian distribution is 0 and the variance is 1, and the properties of the entire distribution only depend on the mean and variance, the computational complexity is very small, and uncorrelation is equal to independence.

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

In machine learning problems, it is often impossible to determine the specific density function of a probability distribution, so it will be very difficult or even impossible to perform subsequent operations on this distribution. At this time, Laplace approximation can be used to approximate a complex distribution using Gaussian distribution, which makes subsequent operations easier.

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

- Simplicity and complexity
- Overfitting
- Bias-Variance Tradeoff

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.

For regression, take house price prediction as an example, we should choose 80% of the dataset to train and 20% of the dataset to test.

For classification, take Sonar classification as an example, we should also choose 80% of the dataset as a training dataset and 20% of the dataset to test.

m) Please explain why the MAP model is usually preferred over the ML model.

In machine learning, Maximum Posteriori optimization provides a Bayesian probability framework for fitting model parameters to training data and an alternative and sibling to the perhaps more common Maximum Likelihood Estimation framework.

Problem VIII. Discussion (5 Points)

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?

In the case of generative models, 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):**

$$P(Y|X) = \frac{P(Y) \cdot P(X|Y)}{P(X)} \tag{22}$$

In the case of discriminative models, 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.

In `On Discriminative vs Generative classifiers: A comparison of logistic regression and naive Bayes` written by Andrew Y.Ng, the comparison of discriminative and generative learning as typified by logistic regression and naive Bayes, while discriminative learning has a lower asymptotic error, a generative classifier may also approach its asymptotic error much faster.