ASSIGNMENT 3

Deadline: 11:59 pm, March 31, 2024

Submit via Blackboard with VeriGuide receipt.

Please follow the course policy and the school's academic honesty policy.

1. Deterministic noise depends on $\mathcal{H}$, as some models approximate f better than others (referring to week 7's slide page 8).

   (1) Assume $\mathcal{H}$ is fixed and we increase the complexity of f. Will deterministic noise in general go up or down? Is there a higher or lower tendency to overfit? Please provide a detailed analysis.

   Hint: Compare the complexity between $\mathcal{H}$ and f.

   (2) Assume f is fixed and we decrease the complexity of $\mathcal{H}$. Will deterministic noise in general go up or down? Is there a higher or lower tendency to overfit? Please provide a detailed analysis.

   Hint: There is a race between two factors that affect overfitting in opposite ways, but one wins.

2. In this problem, we will focus on the regularization (referring to week 7's slide page 19 and page 26).

   Supposed that the regularized weight $\mathbf{w}_{\text{reg}}$ is a solution to:

$$\begin{aligned} &\text{min: } E_{\text{in}}(\mathbf{w}) = \frac{1}{N}\sum_{n=1}^{N}(\mathbf{w}^\intercal \mathbf{x}_n - y_n)^2 \\ &\text{subject to: } \mathbf{w}^\intercal \Gamma^\intercal \Gamma \mathbf{w} \leq C \end{aligned}$$

   where Γ is a matrix. If $\mathbf{w}_{\text{lin}}^\intercal \Gamma^\intercal \Gamma \mathbf{w}_{\text{lin}} \leq C$, where $\mathbf{w}_{\text{lin}}$ is the linear regression solution, then what is the regularized weight $\mathbf{w}_{\text{reg}}$?

   [a] $\mathbf{w}_{\text{reg}} = \mathbf{w}_{\text{lin}}$

   [b] $\mathbf{w}_{\text{reg}} = C\mathbf{w}_{\text{lin}}$

   [c] $\mathbf{w}_{\text{reg}} = \Gamma\mathbf{w}_{\text{lin}}$

   [d] $\mathbf{w}_{\text{reg}} = C\Gamma\mathbf{w}_{\text{lin}}$

   [e] $\mathbf{w}_{\text{reg}} = \Gamma^\intercal\Gamma\mathbf{w}_{\text{lin}}$

   [f] None of the above

   Please select **one correct option** from the above options and briefly discuss your reason.

   Hint: $w_{lin} = (Z^\intercal Z)^{-1}Z^\intercal y$ and $w_{reg} = (Z^\intercal Z + \lambda * I)^{-1}Z^\intercal y$.

3. Supposed that 60% of mobile phones are model A and 40% are model B. Due to the high similarity of the mobile phones, the probability of a tech enthusiast correctly identifying the phone model is 90%. If a tech enthusiast claims to have just seen a mobile phone of model A, what is the probability that the mobile phone is indeed model A?

   Hint: Bayes rule (referring to week 8's slide page 8): $p(y|x) = \frac{p(x|y)p(y)}{p(x)}$.

4. The Poisson distribution is one of the most popular distributions in statistics. You can learn more about it on Wikipedia. The probability density function (PDF) of the Poisson distribution is given as:

$$P(X = x) = \frac{\lambda^x e^{-\lambda}}{x!}$$

Given a sample of n measured values $x_i \in \{0, 1, ...\}$, for $i = 1, ..., n$, we wish to estimate the value of the parameter λ of the Poisson population from which the sample was drawn. Please derive the maximum likelihood estimate $\hat{\lambda}$.

Hint: You could calculate the log likelihood function first and utilize the derivative condition for maximum value to calculate the result.

5. Gaussian Discriminant Analysis (GDA) can be utilized for binary classification tasks (referring to week 8's slide page 27). Considering the datasets in Figure 1, please identify which dataset is more suitable for GDA and briefly discuss your reason.

(a) Dataset A (b) Dataset B

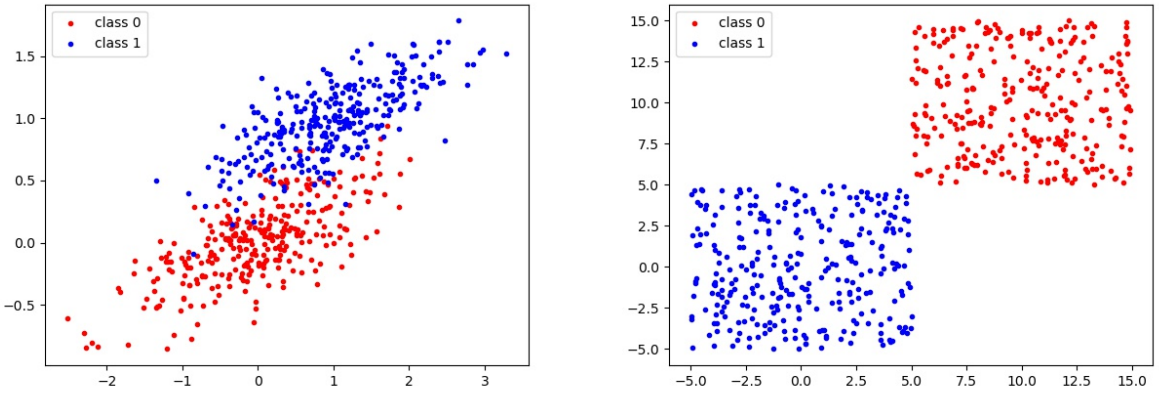


Figure 1

6. Given the GDA model's discriminant function

$$\log p(y = c \mid \boldsymbol{x}, \boldsymbol{\theta}) = \log \pi_c - \frac{1}{2} \log |2\pi \boldsymbol{\Sigma}_c| - \frac{1}{2} (\boldsymbol{x} - \boldsymbol{\mu}_c)^\top \boldsymbol{\Sigma}_c^{-1} (\boldsymbol{x} - \boldsymbol{\mu}_c) + \text{ const} . \quad (1)$$

Please describe why the decision boundary between any two classes will be a quadrative function of x. In addition, in which case the LDA (figure 2 (right), referring to week 8's slide page 29) will be a special case of GDA and please explain that according to the decision boundaries that can be derived from the equation 1.

7. Python Programming Task

(1) Generate a dataset containing 6000 data points from three different Gaussian distributions.

For Gaussian distribution 1: means=$[0, 0]^T$, variance=$diag([1, 2]^T)$.

For Gaussian distribution 2: means=$[4, 0]^T$, variance=$diag([2, 2]^T)$.

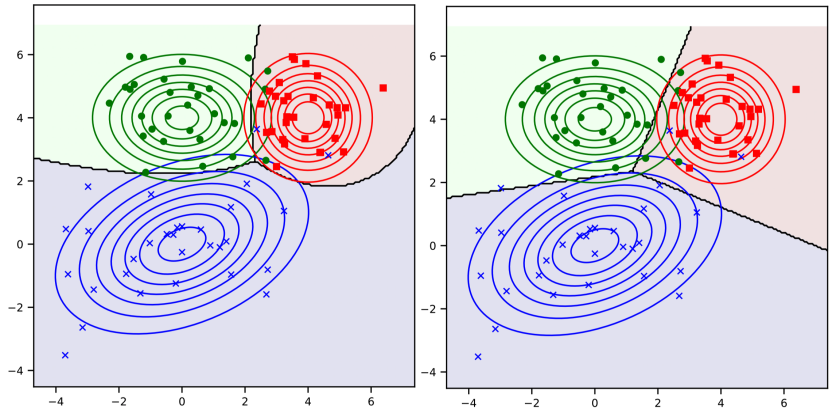


Figure 2

For Gaussian distribution 3: means=$[2, 0]^T$, variance=$diag([2, 1]^T)$.

The intensity ratio of the three Gaussian distributions is 3:2:2. After generating, visualize the results and mark data from different distributions with different colored points.

(2) Fit a Gaussian Mixture Model with three components to the dataset generated in the previous step. Classify each point in the dataset with the fitted model. Provide the precision and recall of the fitting results for each class and plot the prediction result in different colors. Compare the figure you get here with that in the first step. Note that for simplicity, we assume the covariance matrices of each Gaussian distribution are diagonal. There are different prediction-groundtruth matching choices. Use the one can maximize the precision!

(3) In the second step, remove the assumption that the covariance matrices of each Gaussian distribution are diagonal, refit, and predict the category of each point. Provide the new precision and recall. Plot the prediction results in a figure and compare it with that in the second step.

(4) In the third step, repeat the fitting with component numbers of 2, 3, 10, and 100. Plot the prediction results in a figure. Discuss how the hyperparameter "number of components" affects the fitting results.

Hint: You can refer to some python packages from numpy, matplotlib and sklearn. For example, we can directly use **from sklearn.mixture import GaussianMixture** and **np.random.multivariate_normal()**.

Note: Please submit your python implementation (e.g., Q7.py, Q7.ipynb) along with the report. Here is a submission example via Blackboard for your reference: **report.pdf**, **Q7.py**, **veriguide_receipt.pdf**. Please note that there are no restrict limitations regarding the submission format.

*** END ***

1. Deterministic noise depends on $\mathcal{H}$, as some models approximate f better than others (referring to week 7's slide page 8).

(1) Assume $\mathcal{H}$ is fixed and we increase the complexity of f. Will deterministic noise in general go up or down? Is there a higher or lower tendency to overfit? Please provide a detailed analysis.

Hint: Compare the complexity between $\mathcal{H}$ and f.

(2) Assume f is fixed and we decrease the complexity of $\mathcal{H}$. Will deterministic noise in general go up or down? Is there a higher or lower tendency to overfit? Please provide a detailed analysis.

Hint: There is a race between two factors that affect overfitting in opposite ways, but one wins.

(1).

①: when $\mathcal{H}$ is less complex than f.

then the best approximation g from $\mathcal{H}$ is less complex than f and it cannot model y exactly and will interpret the errors as noise, since error will increase when f gets more complex, deterministic noise will increase.

since $\mathcal{H}$ is not changing and deterministic noise increases, g tends to fit more noise and it'll be harder for $\mathcal{H}$ to fit f

so deterministic noise will increase, and ~~tendency to overfit~~ will increase

②: when $\mathcal{H}$ is more complex than f.

then when complexity of f increases, the complexity of $\mathcal{H}$ and f will first be more similar then be more different.

then g can first more and more fit to f but when complexity of f surpasses g, g will less and less fit to f

so errors will first decrease then increase

since these error are interpreted as deterministic noise, deterministic noise will first decrease then increase

since $\mathcal{H}$ is not changing and deterministic noise will first decrease then increase

g tends to first fit less noise then fit more noise

so deterministic noise will first decrease then increase and ~~tendency to overfit~~ will first decrease then increase

(2).

①: when $\mathcal{H}$ is less complex than f:

then the best approximation g from $\mathcal{H}$ is less complex than f and it cannot model y exactly and will interpret the errors as noise, since error will increase when complexity of $\mathcal{H}$ decreases (because its too simple to model complex f)

deterministic noise will increase.

since noise increases, g tends to fit more noise and it'll be harder for $\mathcal{H}$ to fit f

so tendency to overfit will increase

⇒ deterministic noise will increase and ~~tendency to overfit~~ will increase

②: when $\mathcal{H}$ is more complex than f

then when complexity of $\mathcal{H}$ decreases the complexity of g and f will first be more similar then be more different.

before g has same complexity as f, the g can fit less and less actual noises of f which it won't interpret them as deterministic noise, so deterministic noise will decrease

since when complexity of g decreases, g less likely to fit the real noise of data set, tendency to overfit will decrease

after g has same complexity as f, g can less and less fit the dataset

and these error are interpreted as deterministic noise, so deterministic noise will increase

since deterministic noise will first decrease then increase, then tendency to overfit will always first decrease then increase

so deterministic noise will first decrease then increase and ~~tendency to overfit~~ will always first decrease then increase

2. the answer is [a] $w_{reg} = w_{lin}$

proof:

$\because$ linear regression solution is w_{lin}

$\therefore$ for any $w \in \mathcal{H}$, $E_{in}(w_{lin}) < E_{in}(w)$

then under constrian, assume the constrained hypothesis set is $\mathcal{H}'$, So $\mathcal{H}' \subseteq \mathcal{H}$ and $w_{lin} \in \mathcal{H}'$

So it still satisfies that for any $w \in \mathcal{H}'$, $E_{in}(w_{lin}) < E_{in}(w)$

$\Rightarrow w_{lin} = \min_{w} E_{in}(w)$ s.t. $w^T T^T T w \leq C$

$\Rightarrow w_{reg} = w_{lin}$

3.

let M_A define phone is indeed A, and T_A define techician thinks its A.

then $P(M_A) = 60\%$, $P(M_A^c) = 40\%$

probability of correct identification = 90% $\Rightarrow P(T_A|M_A) = P(T_A^c|M_A^c) = 90\%$

$\Rightarrow P(T_A|M_A^c) = 1 - P(T_A^c|M_A^c) = 1 - 90\% = 10\%$

probability phones is indeed A condition to technician thinks it's A:

$$P(M_A|T_A) = \frac{P(T_A|M_A) \cdot P(M_A)}{P(T_A)}$$

$$= \frac{P(T_A|M_A) \cdot P(M_A)}{P(T_A|M_A) \cdot P(M_A) + P(T_A|M_A^c) \cdot P(M_A^c)}$$

$$= \frac{90\% \times 60\%}{90\% \times 60\% + 10\% \times 40\%}$$

$$= \frac{27}{29}$$

$$\approx 93.10\%$$

4.

Since we assume $x_1, \dots x_n$ independent

likelihood of the result: $P(x_1, \dots, x_n | \lambda) = P(x_1|\lambda) \cdot P(x_2|\lambda) \cdot \cdots P(x_n|\lambda) = \prod_{i=1}^{n} P(x_i|\lambda)$

by maximum likelihood estimation:

$$\hat{\lambda} = \arg\max_{\lambda} P(x_1, \dots, x_n | \lambda) = \arg\max_{\lambda} \prod_{i=1}^{n} P(x_i|\lambda)$$

$\ln e^{-\lambda}$

$$= \arg\max_{\lambda} \sum_{i=1}^{n} \ln P(x_i|\lambda) = \arg\max_{\lambda} \sum_{i=1}^{n} \ln \frac{\lambda^{x_i} e^{-\lambda}}{x_i!} = \arg\max_{\lambda} \sum_{i=1}^{n} [x_i \ln\lambda - \lambda - \ln(x_i!)]$$

$$= \arg\max_{\lambda} \left[\left(\sum_{i=1}^{n} x_i\right) \cdot \ln\lambda - n\lambda \right]$$

let $\frac{\partial}{\partial \lambda} \left[\left(\sum_{i=1}^{n} x_i\right) \cdot \ln\lambda - n\lambda \right] = \frac{\sum_{i=1}^{n} x_i}{\lambda} - n = 0 \quad \Rightarrow \quad \lambda = \frac{\sum_{i=1}^{n} x_i}{n}$

$\because \sum_{i=1}^{n} x_i > 0$

$\Rightarrow$ when $0 < \lambda \leq \frac{\sum_{i=1}^{n} x_i}{n}$, $\frac{\partial}{\partial \lambda} \left[\left(\sum_{i=1}^{n} x_i\right) \cdot \ln\lambda - n\lambda \right] \geq 0$, $\left[\left(\sum_{i=1}^{n} x_i\right) \cdot \ln\lambda - n\lambda \right]$ increases

when $\lambda > \frac{\sum_{i=1}^{n} x_i}{n}$, $\frac{\partial}{\partial \lambda} \left[\left(\sum_{i=1}^{n} x_i\right) \cdot \ln\lambda - n\lambda \right] < 0$, $\left[\left(\sum_{i=1}^{n} x_i\right) \cdot \ln\lambda - n\lambda \right]$ decreases

$\Rightarrow \hat{\lambda} = \arg\max_{\lambda} \left[\left(\sum_{i=1}^{n} x_i\right) \cdot \ln\lambda - n\lambda \right] = \frac{\sum_{i=1}^{n} x_i}{n}$

1

5.

I think dataset A is more suitable for GDA model.

this is because GDA model has the prior asumption that the data distribution follows Gaussian distribution

in Dataset A, both blue and red data give a eliptical, Gaussian-distribution-like shape,

but in Dataset A, both blue and red data give a square shape which are not likely to be Gaussian distribution

thus contradicts the prior, shouldn't be fit with GDA model.

So dataset A is more suitable for GDA model

6

for any point on the boundary, the probability that the point belongs to each center should be the

i.e. the boundary formula for any center c_1, c_2 is:

$$\frac{1}{2}\log|2\pi\Sigma_{c_1}| - \frac{1}{2}(x-\mu_{c_1})^T\Sigma_{c_1}^{-1}(x-\mu_{c_1}) + const_1 = \frac{1}{2}\log|2\pi\Sigma_{c_2}| - \frac{1}{2}(x-\mu_{c_2})^T\Sigma_{c_2}^{-1}(x-\mu_{c_2}) + const_2$$

Since $\frac{1}{2}(x-\mu_{c_1})^T\Sigma_{c_1}^{-1}(x-\mu_{c_1})$ and $(x-\mu_{c_2})^T\Sigma_{c_2}^{-1}(x-\mu_{c_2})$ are quardratic

and generally $\Sigma_{c_1} \neq \Sigma_{c_2}$ so that $\frac{1}{2}(x-\mu_{c_1})^T\Sigma_{c_1}^{-1}(x-\mu_{c_1}) - (x-\mu_{c_2})^T\Sigma_{c_2}^{-1}(x-\mu_{c_2})$ is generally quadratic

So the decision boundary will be quadrative

in LDA, we assume that the covariance matrix are the same for all clusters i.e. for any center c_1, c_2: $\Sigma_{c_1} = \Sigma_{c_2} = \Sigma$

So descriminant function becomes

$$\frac{1}{2}\log|2\pi\Sigma| - \frac{1}{2}(x-\mu_c)^T\Sigma^{-1}(x-\mu_{c_1}) + const_1$$
$$= \frac{1}{2}\log|2\pi\Sigma| - \frac{1}{2}\mu_c^T\Sigma^{-1}\mu_c + X^T\Sigma^{-1}\mu_c + const - \frac{1}{2}X^T\Sigma^{-1}X$$

So the boundary formula for any center c_1, c_2 is:

$$\frac{1}{2}\log|2\pi\Sigma| - \frac{1}{2}\mu_{c_1}^T\Sigma^{-1}\mu_{c_1} + X^T\Sigma^{-1}\mu_{c_1} + const_1 - \frac{1}{2}X^T\Sigma^{-1}X = \frac{1}{2}\log|2\pi\Sigma| - \frac{1}{2}\mu_{c_2}^T\Sigma^{-1}\mu_{c_2} + X^T\Sigma^{-1}\mu_{c_2} + const_2 - \frac{1}{2}X^T\Sigma^{-1}X$$

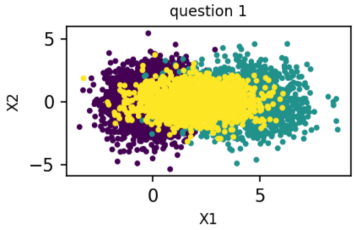
$$\Rightarrow X^T\Sigma^{-1}\mu_{c_1} - X^T\Sigma^{-1}\mu_{c_2} - \frac{1}{2}\mu_{c_1}^T\Sigma^{-1}\mu_{c_1} + \frac{1}{2}\mu_{c_2}^T\Sigma^{-1}\mu_{c_2} + const_1 - const_2 = 0$$

the quadratic term $X^T\Sigma^{-1}X$ is the same on both sides and is eliminated, and highest order is 1

so boundary is linear

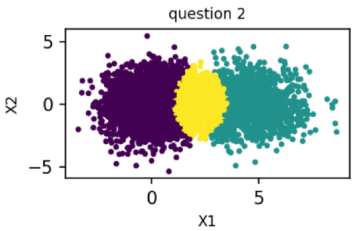
7. the code is attached in the zip file

(1) the output is shown below



(2)

predicted data:



precision and recall are shown in the output.

```
question (2)
precision for class 1, 2, 3 is [0.8833138856476079, 0.7036172695449242, 0.5918367346938775]
recall for class 1, 2, 3 is [0.8231243204059442, 0.8271604938271605, 0.5692652832305104]
```

compare:

the centers of the three classes are predicted approximately correct.

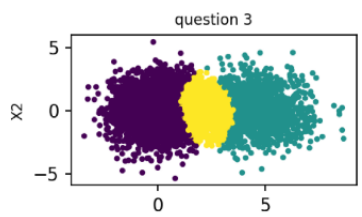
the purple and cyan class are predicted almost correct with high accuracy of more than 70%

the many yellow points that are closed to the purple and cyan center are classified as purple and cyan

So the accuracy is relatively low

(3)

predicted data



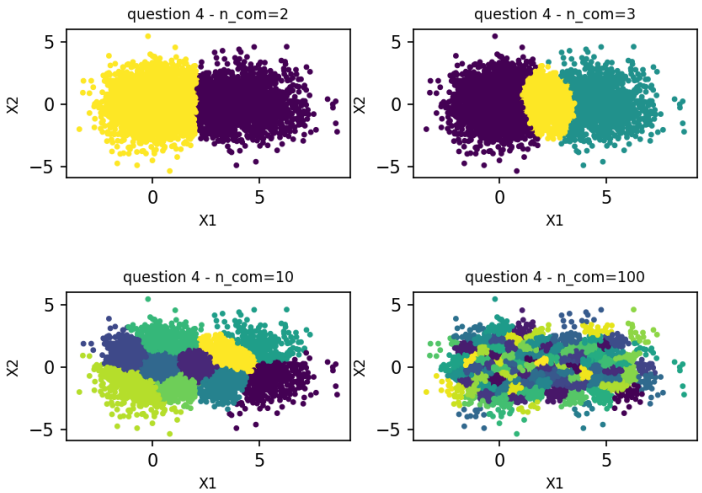
precision an recall are shown in the output.

```
question (3):
new precision for class 1, 2, 3 is [0.8856476079346558, 0.7007001166861143, 0.5912536443148688]
new recall for class 1, 2, 3 is [0.8220216606498195, 0.8282758620689655, 0.5696629213483146]
```

in terms of the plot, question 2 and question 3 are almost the same, but the yellow cluster is slight different since the yellow cluster in question 3 is more tilted in the top left - bottom right diagonal.

the accuracy is almost the same

(E) predicted data:



when number of data increases, the data is separated into more clusters, and each clusters have less points and the size are smaller