Solution 1 (a)

$$\boldsymbol{\mu} = [\mu_x, \mu_y]^T$$

$$\boldsymbol{\Sigma} = \begin{bmatrix} \sigma_x^2 & \rho\sigma_x\sigma_y \\ \rho\sigma_x\sigma_y & \sigma_y^2 \end{bmatrix}$$

- Mean vector: $\boldsymbol{\mu}$
- Covariance matrix: $\boldsymbol{\Sigma}$

Let $\mu_x = 2$, $\mu_y = 2$, $\sigma_x = 1$, $\sigma_y = 0.5$, and $\rho = -0.5$

Solve for mean vector $\boldsymbol{\mu}$:

$$\boldsymbol{\mu} = \begin{bmatrix} \mu_x & \mu_y \end{bmatrix}^T = \begin{bmatrix} \mu_x \\ \mu_y \end{bmatrix} = \begin{bmatrix} 2 \\ 2 \end{bmatrix}$$

Solve for covariance matrix $\boldsymbol{\Sigma}$:

$$\boldsymbol{\Sigma} = \begin{bmatrix} \sigma_x^2 & \rho\sigma_x\sigma_y \\ \rho\sigma_x\sigma_y & \sigma_y^2 \end{bmatrix} = \begin{bmatrix} 1^2 & (-0.5)(1)(0.5) \\ (-0.5)(1)(0.5) & 0.5^2 \end{bmatrix} = \begin{bmatrix} 1 & -0.25 \\ -0.25 & 0.25 \end{bmatrix}$$

$\therefore$ the bivariate Gaussian has parameters:

$$\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \mathcal{N}\left(\begin{bmatrix} 2 \\ 2 \end{bmatrix}, \begin{bmatrix} 1 & -0.25 \\ -0.25 & 0.25 \end{bmatrix}\right)$$

Solution 1 (b)

For a bivariate Gaussian distribution, we need to specify the following parameters:

$$\boldsymbol{\mu} = [\mu_x, \mu_y]^T$$

$$\boldsymbol{\Sigma} = \begin{bmatrix} \sigma_x^2 & \rho\sigma_x\sigma_y \\ \rho\sigma_x\sigma_y & \sigma_y^2 \end{bmatrix}$$

- Mean vector: $\boldsymbol{\mu}$
- Covariance matrix: $\boldsymbol{\Sigma}$

Let $\mu_x = 1$, $\mu_y = 2$, $\sigma_x = 1$, $\sigma_y = 0$

Solve for mean vector $\boldsymbol{\mu}$:

$$\boldsymbol{\mu} = \begin{bmatrix} \mu_x & \mu_y \end{bmatrix}^T = \begin{bmatrix} \mu_x \\ \mu_y \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

Solve for covariance matrix $\boldsymbol{\Sigma}$:

$$\boldsymbol{\Sigma} = \begin{bmatrix} \sigma_x^2 & \rho\sigma_x\sigma_y \\ \rho\sigma_x\sigma_y & \sigma_y^2 \end{bmatrix} = \begin{bmatrix} 1^2 & 0 \\ 0 & 0^2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$$

$\therefore$ the bivariate Gaussian has parameters:

$$\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \mathcal{N}\left(\begin{bmatrix} 1 \\ 2 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}\right)$$

Solution 2

Step 1

The Bayesian decision rule for generative classifiers is defined as:

$$\hat{y} = \arg\max_{y \in \{+,-\}} p(y|x) = \arg\max_{y \in \{+,-\}} \frac{p(x|y)p(y)}{p(x)}$$

- Since $p(x)$ is constant for both classes, the decision rule simplifies to:

$$\hat{y} = \arg\max_{y \in \{+,-\}} p(x|y)p(y)$$

Step 2

The $(+)$ class is predicticted when the folowing holds:

$$p(x|+)p(+) > p(x|-)p(-)$$

Step 3

Identify possible reasons for always predicting the positive class.

1. **Highly imbalanced prior probabilities:** If $p(+) \gg p(-)$, the classifier might always predict the positive class because the prior term dominates the decision, regardless of the likelihood term. This occurs when the training data contains many more $+$ examples than $-$ ones.
2. **Poor estimation of class-conditional densities:** If $p(x|+)$ is consistently overestimated or $p(x|-)$ is consistently underestimated across the input space, the classifier will favor the $+$ class.

$\therefore$ the classifier that predicts $+$ for all points x in the input space is likely due to a combination of highly imbalanced prior probabilities and poor estimation of class-conditional densities.

Solution 3 (a)

Step 1

From Bayes Theorem we have:

$$p(y|x) = \frac{p(x|y)p(y)}{p(x)}$$

$$or$$

$$p(y|x) \propto p(x|y)p(y)$$

Step 2

The class probabilities are given as:

$$\pi_1 = 0.33 \;\; \pi_2 = 0.39 \;\; \pi_3 = 0.28$$

From the plot the density for each class is:

$$P(x = 12.0|Class_1) = 0.4 \;\; P(x = 12.0|Class_2) = 0.05 \;\; P(x = 12.0|Class_3) = 0.0025$$

Step 3

Use class probabilities and densities from *Step 2*, the apply Bayes Theorem

$$P(Class_1|x = 12.0) \propto P(x = 12.0|Class_1)P(Class_1) = 0.4 \times 0.33 = 0.132$$

$$P(Class_2|x = 12.0) \propto P(x = 12.0|Class_2)P(Class_2) = 0.05 \times 0.39 = 0.0195$$

$$P(Class_3|x = 12.0) \propto P(x = 12.0|Class_3)P(Class_3) = 0.0025 \times 0.28 = 0.0007$$

$\therefore$ the label $Class_1$ would be assigned at x=12.0

Solution 3 (b)

Step 1

From Bayes Theorem we have:

$$p(y|x) = \frac{p(x|y)p(y)}{p(x)}$$

$$\textit{or}$$

$$p(y|x) \propto p(x|y)p(y)$$

Step 2

The class probabilities are given as:

$$\pi_1 = 0.33 \;\; \pi_2 = 0.39 \;\; \pi_3 = 0.28$$

From *Figure 1*, the density for each class is:

$$P(x = 12.5|Class_1) = 0.6 \;\; P(x = 12.5|Class_2) = 0.3 \;\; P(x = 12.5|Class_3) = 0.05$$

Step 3

Use class probabilities and densities from *Step 2*, then apply Bayes Theorem

$P(Class_1|x = 12.5) \propto P(x = 12.5|Class_1)P(Class_1) = 0.6 \times 0.33 = 0.198$

$P(Class_2|x = 12.5) \propto P(x = 12.5|Class_2)P(Class_2) = 0.3 \times 0.39 = 0.117$

$P(Class_3|x = 12.5) \propto P(x = 12.5|Class_3)P(Class_3) = 0.05 \times 0.28 = 0.014$

$\therefore$ the label $Class_1$ would be assigned at x=12.5

Solution 3 (c)

Step 1

From Bayes Theorem we have:

$$p(y|x) = \frac{p(x|y)p(y)}{p(x)}$$

$$\textit{or}$$

$$p(y|x) \propto p(x|y)p(y)$$

Step 2

The class probabilities are given as:

$$\pi_1 = 0.33 \;\; \pi_2 = 0.39 \;\; \pi_3 = 0.28$$

From *Figure 1*, the density for each class is:

$$P(x = 13.0|Class_1) = 0.3 \;\; P(x = 13.0|Class_2) = 0.6 \;\; P(x = 13.0|Class_3) = 0.2$$

Step 3

Use class probabilities and densities from *Step 2*, then apply Bayes Theorem

$P(Class_1|x = 13.0) \propto P(x = 13.0|Class_1)P(Class_1) = 0.3 \times 0.33 = 0.099$

$P(Class_2|x = 13.0) \propto P(x = 13.0|Class_2)P(Class_2) = 0.6 \times 0.39 = 0.234$

$P(Class_3|x = 13.0) \propto P(x = 13.0|Class_3)P(Class_3) = 0.2 \times 0.28 = 0.056$

$\therefore$ the label $Class_2$ would be assigned at x=13.0

Solution 3 (d)

Step 1

From Bayes Theorem we have:

$$p(y|x) = \frac{p(x|y)p(y)}{p(x)}$$

$$or$$

$$p(y|x) \propto p(x|y)p(y)$$

Step 2

The class probabilities are given as:

$$\pi_1 = 0.33 \;\; \pi_2 = 0.39 \;\; \pi_3 = 0.28$$

From *Figure 1*, the density for each class is:

$$P(x = 13.5|Class_1) = 0.1 \;\; P(x = 13.5|Class_2) = 0.7 \;\; P(x = 13.5|Class_3) = 0.4$$

Step 3

Use class probabilities and densities from *Step 2*, then apply Bayes Theorem

$P(Class_1|x = 13.5) \propto P(x = 13.5|Class_1)P(Class_1) = 0.1 \times 0.33 = 0.033$

$P(Class_2|x = 13.5) \propto P(x = 13.5|Class_2)P(Class_2) = 0.7 \times 0.39 = 0.273$

$P(Class_3|x = 13.5) \propto P(x = 13.5|Class_3)P(Class_3) = 0.4 \times 0.28 = 0.112$

$\therefore$ the label $Class_2$ would be assigned at x=13.5

Solution 3 (e)

Step 1

From Bayes Theorem we have:

$$p(y|x) = \frac{p(x|y)p(y)}{p(x)}$$

$$or$$

$$p(y|x) \propto p(x|y)p(y)$$

Step 2

The class probabilities are given as:

$$\pi_1 = 0.33 \;\; \pi_2 = 0.39 \;\; \pi_3 = 0.28$$

From *Figure 1*, the density for each class is:

$$P(x = 14.0|Class_1) = 0.05 \;\; P(x = 14.0|Class_2) = 0.2 \;\; P(x = 14.0|Class_3) = 0.8$$

Step 3

Use class probabilities and densities from *Step 2*, then apply Bayes Theorem

$$P(Class_1|x = 14.0) \propto P(x = 14.0|Class_1)P(Class_1) = 0.05 \times 0.33 = 0.0165$$

$$P(Class_2|x = 14.0) \propto P(x = 14.0|Class_2)P(Class_2) = 0.2 \times 0.39 = 0.078$$

$$P(Class_3|x = 14.0) \propto P(x = 14.0|Class_3)P(Class_3) = 0.8 \times 0.28 = 0.224$$

$\therefore$ the label $Class_3$ would be assigned at x=14.0

Solution 4 (a)

Step 1

Analyze μ

- Since $\mu = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$, the center of the Gaussian is at the origin.

Step 2

Analyze Σ

- The covariance matrix $\Sigma = \begin{bmatrix} 9 & 0 \\ 0 & 1 \end{bmatrix}$ indicates that the variance in the x-direction is 9 and in the y-direction is 1. This means that the Gaussian will be elongated along the x-axis.

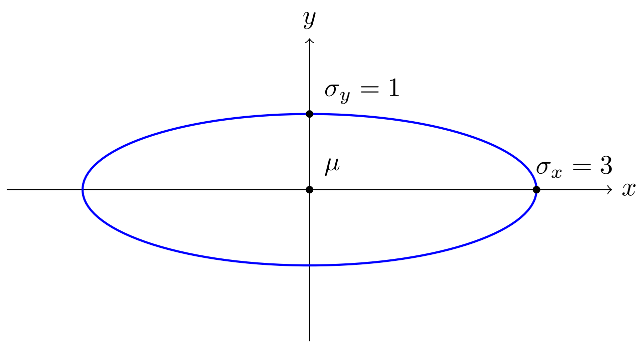
Step 3

Look at the standard deviation to determine the shape, since Σ is a diagonal matrix

- The standard deviation in the x-direction is $\sigma_x = \sqrt{9} = 3$
- The standard deviation in the y-direction is $\sigma_y = \sqrt{1} = 1$

Step 4

Sketch the Gaussian



Solution 4 (b)

Step 1

Analyze μ

- Since $\mu = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$, the center of the Gaussian is at the origin.

Step 2

Analyze Σ

- The covariance matrix $\Sigma = \begin{bmatrix} 1 & -0.75 \\ -0.75 & 1 \end{bmatrix}$ has non-zero off-diagonal elements.
- This indicates a correlation between variables.
- The negative correlation (-0.75) means that as one variable increases, the other tends to decrease.
- Look at the eigenvalues and eigenvectors to determine the shape since we don't have a diagonal matrix in this case.

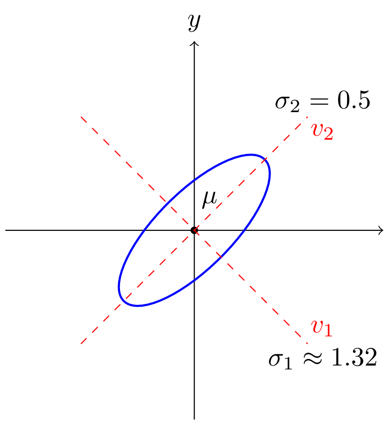
Step 3

Find eigenvalues and eigenvectors

- The eigenvalues of Σ are $\lambda_1 = 1 + 0.75 = 1.75$ and $\lambda_2 = 1 - 0.75 = 0.25$.
- The corresponding eigenvectors are $v_1 = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$ and $v_2 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$.
- The standard deviations along the principal axes are $\sigma_1 = \sqrt{1.75} \approx 1.32$ and $\sigma_2 = \sqrt{0.25} = 0.5$.

Step 4

Sketch the Gaussian



Solution 5

A unit vector in $\mathbb{R}^2$ is a vector of the form:

$$\begin{bmatrix} x \\ y \end{bmatrix}$$

To be orthogonal to $\begin{bmatrix} 1 \\ 1 \end{bmatrix}$, the dot product must equal zero:

$$\begin{bmatrix} 1 \\ 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix} = 0$$

This gives the equation:

$$x + y = 0$$

This means that $y = -x$.
Now, we need to find the unit vectors. A unit vector has a magnitude of 1:

$$\sqrt{x^2 + y^2} = 1$$

Substituting $y = -x$ into the equation:

$$\sqrt{x^2 + (-x)^2} = 1$$
$$\sqrt{2x^2} = 1$$
$$\sqrt{2}|x| = 1$$
$$|x| = \frac{1}{\sqrt{2}}$$

This gives two solutions for x:

$$x = \frac{1}{\sqrt{2}} \quad \text{or} \quad x = -\frac{1}{\sqrt{2}}$$

Substituting back to find y:

$$y = -\frac{1}{\sqrt{2}} \quad \text{or} \quad y = \frac{1}{\sqrt{2}}$$

$\therefore$ the unit vectors orthogonal to $\begin{bmatrix} 1 \\ 1 \end{bmatrix}$ are:

$$\begin{bmatrix} \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix}$$

Solution 6

Let $x \cdot x = 25, \ \forall \, x \in \mathbb{R}^d$

$$x \cdot x = \|x\|^2 = 25 \rightarrow \sqrt{\|x\|^2} = \sqrt{25} \rightarrow \|x\| = 5$$

Hence, the vector x has a magnitude of 5.

$\therefore$ the set of all points $x \in \mathbb{R}^d$ with $x \cdot x = 25$ is a sphere of radius 5 centered at the origin in d-dimensional space.

Solution 7

The function $f(x) = 2x_1 - x_2 + 6x_3$ can be expressed in the form of a dot product:

$$f(x) = w \cdot x$$

Where w is a vector in $\mathbb{R}^3$ and x is a vector in $\mathbb{R}^3$.

We can then express $f(x)$ as the dot product of two matrices:

$$f(x) = \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = w_1 \cdot x_1 + w_2 \cdot x_2 + w_3 \cdot x_3$$

It follows that: $w_1 = 2 \;\; w_2 = -1 \;\; w_3 = 6$

$\therefore$ the vector w is:

$$w = \begin{bmatrix} 2 \\ -1 \\ 6 \end{bmatrix}$$

Solution 8

Let the dimensions of A be $m \times n$ and the dimensions of B be $n \times p$.

The product AB will have dimensions $m \times p$.

Given that AB has dimensions 10×20, and $A = m \times 30$.

$$m = 10 \qquad p = 20 \qquad n = 30$$

We know that the product AB can be expressed as:

$$AB = (m \times n) \times (n \times p)$$

$\therefore$ the dimensions of A and B are:

$$A : 10 \times 30$$
$$B : 30 \times 20$$

Solution 9 (a)

The matrix X has n rows and d columns, so the dimension of X is:

$$X \in \mathbb{R}^{n \times d}$$

This means that X has n data points, each with d features.

$\therefore$ the dimension of X is $n \times d$.

Solution 9 (b)

The matrix XX^T is the product of an $n \times d$ matrix and a $d \times n$ matrix.

The resulting matrix will have dimensions $n \times n$.

$\therefore$ the dimension of XX^T is $n \times n$.

Solution 9 (c)

The (i, j) entry of $X^T X$ is the dot product of the i-th row of X and the j-th column of X^T

This is simply the sum of the products of the corresponding elements:

$$(X^T X)_{ij} = \sum_{k=1}^{d} x_k^{(i)} x_k^{(j)}$$

This is the inner product of the i-th and j-th data points.

$\therefore$ the (i, j) entry of $X^T X$ is the inner product of the i-th and j-th data points or $(x^{(n)}, x^{(n)})$.

Solution 10

Step 1

Compute x^T

$$x = \begin{bmatrix} 1 \\ 3 \\ 5 \end{bmatrix} \rightarrow x^T = \begin{bmatrix} 1 & 3 & 5 \end{bmatrix}$$

Step 2

Compute $x^T x$

$$x^T x = \begin{bmatrix} 1 & 3 & 5 \end{bmatrix} \begin{bmatrix} 1 \\ 3 \\ 5 \end{bmatrix} = 1^2 + 3^2 + 5^2 = 1 + 9 + 25 = 35$$

Step 3

Compute xx^T

$$xx^T = \begin{bmatrix} 1 \\ 3 \\ 5 \end{bmatrix} \begin{bmatrix} 1 & 3 & 5 \end{bmatrix} = \begin{bmatrix} 1^2 & 1 \cdot 3 & 1 \cdot 5 \\ 3 \cdot 1 & 3^2 & 3 \cdot 5 \\ 5 \cdot 1 & 5 \cdot 3 & 5^2 \end{bmatrix} = \begin{bmatrix} 1 & 3 & 5 \\ 3 & 9 & 15 \\ 5 & 15 & 25 \end{bmatrix}$$

$\therefore$ the result of $x^T x$ is a scalar 35 and the result of xx^T is a matrix:

$$\begin{bmatrix} 1 & 3 & 5 \\ 3 & 9 & 15 \\ 5 & 15 & 25 \end{bmatrix}$$

Solution 11

Let $f(x) = 3x_1^2 + 2x_1x_2 - 4x_1x_3 + 6x_3^2$

Step 1

Define x, x^T, and M

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

$$x^T = \begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix}$$

$$M = \begin{bmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{bmatrix}$$

Step 2

Expand $x^T Mx$ so it is in the same form as $f(x)$

$$x^T Mx = m_{11}x_1^2 + m_{12}x_1x_2 + m_{13}x_1x_3 + m_{21}x_2x_1 + m_{22}x_2^2 + m_{23}x_2x_3 + m_{31}x_3x_1 + m_{32}x_3x_2 + m_{33}x_3^2.$$

Step 3

Match coeficients of $x^T Mx$ with $f(x)$

$$m_{11} = 3 \quad m_{12} = 2 \quad m_{13} = -4$$
$$m_{21} = 2 \quad m_{22} = 0 \quad m_{23} = 0$$
$$m_{31} = -4 \quad m_{32} = 0 \quad m_{33} = 6$$

Step 4

Plug in values and express the matrix M

$$M = \begin{bmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{bmatrix} = \begin{bmatrix} 3 & 2 & -4 \\ 2 & 0 & 0 \\ -4 & 0 & 6 \end{bmatrix}$$

$\therefore$ the matrix M is:

$$M = \begin{bmatrix} 3 & 2 & -4 \\ 2 & 0 & 0 \\ -4 & 0 & 6 \end{bmatrix}$$

Solution 12 (a)

The determinant of a diagonal matrix is the product of its diagonal elements:

$$|A| = 1 \cdot 2 \cdot 3 \cdot 4 \cdot 5 \cdot 6 \cdot 7 \cdot 8 = 8! = 40320$$

$\therefore$ the determinant of $A = 40320$

Solution 12 (b)

Since all values above and below the diagonal are zero the inverse of A is obtained by taking the reciprocal of each diagonal element:

$$A^{-1} = diag\left(\frac{1}{1}, \frac{1}{2}, \frac{1}{3}, \frac{1}{4}, \frac{1}{5}, \frac{1}{6}, \frac{1}{7}, \frac{1}{8}\right)$$

Checking the definition of the inverse $AA^{-1} = I$:

$$AA^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 6 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 7 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 8 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{3} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{4} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{5} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{6} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{7} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{8} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$\therefore A^{-1}$ is:

$$A^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{3} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{4} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{5} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{6} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{7} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{8} \end{bmatrix}$$

Solution 13 (a)

Pseudocode for training procedure:

```
FUNCTION fit_generative_model(x, y):
    SET k = 10 (number of classes)
    SET d = number of features in x
    INITIALIZE mu AS kxd matrix of zeros
    INITIALIZE sigma AS kxdxd matrix of zeros
    INITIALIZE pi AS vector of k zeros

    Normalize x to range [0,1]

    Split data into training (80%) and validation (20%) sets

    FUNCTION calc_priors_means_covariances(x, y, c):
        FOR each class j from 0 to k-1:
            Find all samples where y equals j
            CALCULATE prior probability pi[j] AS fraction of samples in class j
            CALCULATE mean vector mu[j] from samples in class j
            IF c > 0:
                CALCULATE covariance matrix sigma[j] with regularization c
            ELSE:
                CALCULATE covariance matrix sigma[j] without regularization
        RETURN mu, sigma, pi

    FUNCTION fit(x_train, y_train, x_val, y_val, c):
        CALCULATE mu, sigma, pi using training data with regularization c
        INITIALIZE score matrix for validation samples

        FOR each class label from 0 to k-1:
            Create multivariate normal distribution with mean mu[label] and covariance
                sigma[label]
            CALCULATE log probability for all validation samples (vectorized)

        Make predictions by finding class with highest score for each sample
        Count number of errors (predictions != y_val)
        RETURN number of errors

    FUNCTION find_best_c():
        INITIALIZE list of c values to test
        INITIALIZE empty list for errors
        INITIALIZE dictionary to store c values and errors

        FOR each c value:
            CALCULATE error using fit function
            Store c and error in lists and dictionary

        Find c with minimum error
        RETURN best c, c_error_dictionary

    CALCULATE best c using find_best_c function

    CALCULATE final mu, sigma, pi using entire dataset with best c

    RETURN mu, sigma, pi
END FUNCTION
```

Python code for training procedure:

```python
def fit_generative_model(x,y):
    k = 10  # labels 0,1,...,k-1
    d = (x.shape)[1]  # number of features
    mu = np.zeros((k,d))
    sigma = np.zeros((k,d,d))
    pi = np.zeros(k)

    ## normalize data [0,1]
    x = x.astype(np.float32)/255.0
```

```

    ## initialize training set and validation set to find best regularization parameter c
    ## note: we will do a 80/20 split of x,y
    x_train, x_val, y_train, y_val = train_test_split(x, y, test_size=0.2, random_state=17)

    def calc_priors_means_covariances(x,y,c):
        ## calc priors, means, and covariances
        if c > 0:
            for j in range(k):
                indices = (y==j).flatten()
                x_j = x[indices]
                pi[j] = x_j.shape[0]/x.shape[0]
                mu[j] = np.mean(x_j, axis=0)
                sigma[j] = np.cov(x_j.T) + c*np.eye(d)
        else:
            for j in range(k):
                indices = (y==j).flatten()
                x_j = x[indices]
                pi[j] = x_j.shape[0]/x.shape[0]
                mu[j] = np.mean(x_j, axis=0)
                sigma[j] = np.cov(x_j.T)
        return mu, sigma, pi

    ## evaluate c using multivariate normal distribution to the training set and validation set
    def fit(x_train,y_train,x_val,y_val,c):
        print(f'calculating mu, sigma, pi for c: {c}')
        mu, sigma, pi = calc_priors_means_covariances(x_train,y_train,c)
        score = np.zeros((len(x_val),k))
        print(f'calc score for c: {c}')
        for label in range(0,k):
            rv = multivariate_normal(mean=mu[label], cov=sigma[label],allow_singular=True)
            score[:,label] = np.log(pi[label]) + rv.logpdf(x_val[:,:])
            #for i in range(0,len(x_val)):
                #score[i,label] = np.log(pi[label]) + rv.logpdf(x_val[i,:])
        predictions = np.argmax(score, axis=1)
        errors = np.sum(predictions != y_val)
        print(f'c: {c}, errors: {errors}')
        return errors

    ## find best c
    def find_best_c():
        ## initialize c values to test and an empty list to store errors
        c_values = [0,0.0001,0.001, 0.01, 0.1, 1, 10, 100, 1000, 10000]
        errors = []
        ## loop through c values and calculate error
        for c in c_values:
            print("Calculating error for c: ", c)
            error = fit(x_train,y_train,x_val,y_val,c)
            errors.append(error)
            print("c: ", c, " error: ", error)
            c_error_dict['c_value'].append(c)
            c_error_dict['error'].append(error)
        ## find best c where error is minimum
        best_c = c_values[np.argmin(errors)]

        return best_c

    ## calculate best c
    best_c, c_error_dict = find_best_c()

    ## calc mu, sigma, pi using best c
    mu, sigma, pi = calc_priors_means_covariances(x,y,best_c)

    # Halt and return parameters
    return mu, sigma, pi
```

Solution 13 (b)

- A single value of $c = 0.1$ for all ten classes.

- The value of c was chosen based on the validation set error rate.

  - The model returned 608 errors for the 12000 validation labels from the training labels.

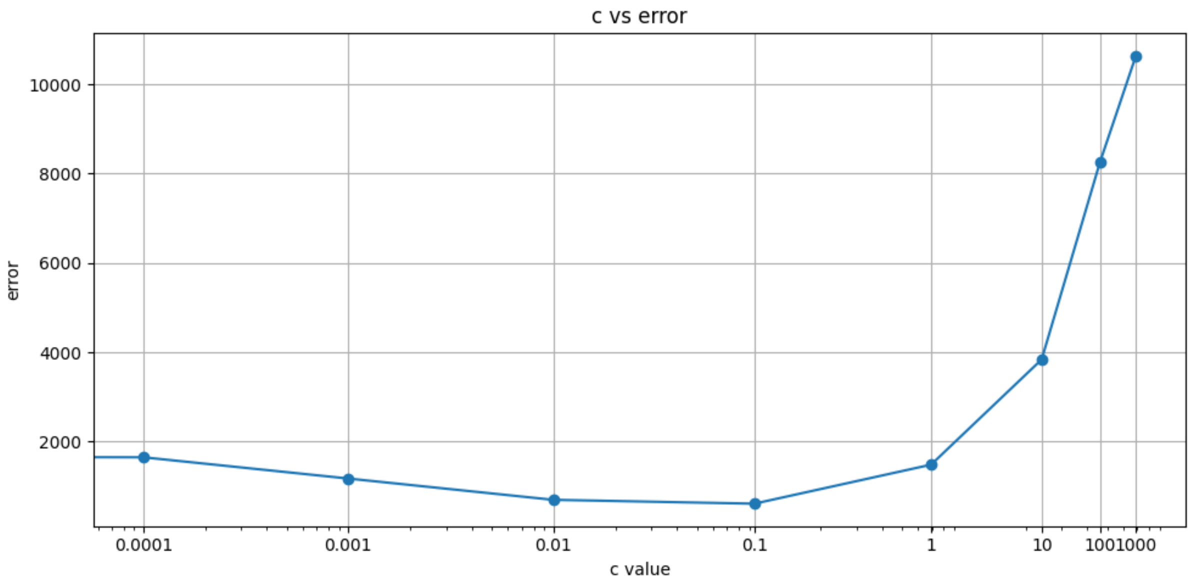


Figure 1: Results from c value testing

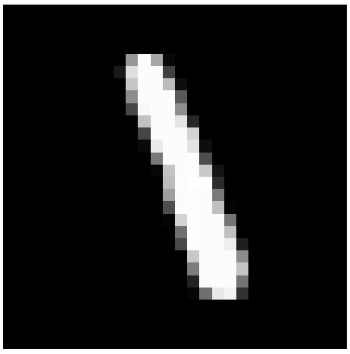
Solution 13 (c)

- The accuracy for $c = 0.1$ on the test data was 0.9423 or 94.23%.

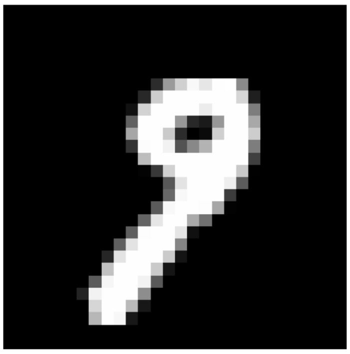
- The model with $c = 0.1$ predicted 577 out of 10,000 incorrectly.

$\therefore$ The error rate on the MNIST test set was 0.0577 or 5.77%.

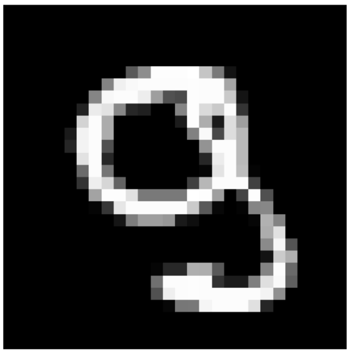
Solution 13 (d)



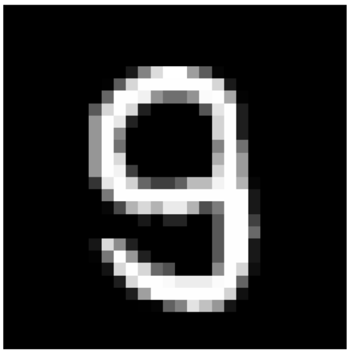
(a) 1 classified as 8



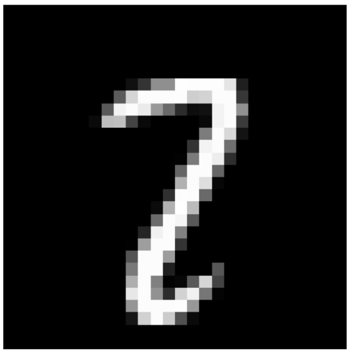
(b) 9 classified as 7



(c) 9 classified as 8



(d) 9 classified as 8



(e) 2 classified as 7

Figure 2: Examples of misclassified digits from the test set