# Assignment 2a

## Problem 1

Let $f(x,y) = x^2 + xy + y^2$.

(a) Compute the gradient $\nabla f(x,y)$.

(b) Starting from ($\mathbf{x}0 = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$*) and using step size (*$\eta$*=0.1), perform* ***two*** *steps of gradient descent:*
$\mathbf{x}k+1 = \mathbf{x}_k - \eta, \nabla f(\mathbf{x}_k)$,
and report ($\mathbf{x}_1$) and ($\mathbf{x}_2$).

(c) Write the iteration in matrix form ($\mathbf{x}_{k+1} = (I - \eta A)\mathbf{x}_k$) by identifying the matrix (A) for the given function.

## (a) Compute the gradient (∇f(x,y)).

Compute partial derivatives:

- $\dfrac{\partial f}{\partial x} = \dfrac{\partial}{\partial x}(x^2 + xy + y^2) = 2x + y.$

  *(Why? differentiate (x^2) → (2x); (xy) w.r.t. (x) gives (y); ($y^2$) is constant in (x).)*
- $\dfrac{\partial f}{\partial y} = \dfrac{\partial}{\partial y}(x^2 + xy + y^2) = x + 2y.$

  *(Why? differentiate (y^2) → (2y); (xy) w.r.t. (y) gives (x); ($x^2$) is constant in (y).)*

So the gradient vector is

$$\boxed{\nabla f(x,y) = \begin{bmatrix} 2x + y \\ x + 2y \end{bmatrix}.}$$

## (b) Two steps of gradient descent from ($\mathbf{x}_0 = [1, -1]^T$) with ($\eta = 0.1$).

**Step 0 → Step 1**

First, evaluate the gradient at ($\mathbf{x}_0$).

$$\nabla f(\mathbf{x}_0) = \begin{bmatrix} 2(1) + (-1) \\ 1 + 2(-1) \end{bmatrix}$$

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

$$= \begin{bmatrix} 1 \\ -1 \end{bmatrix}.$$

*(Why? direct substitution into the  gradient formula.)*

Now apply the gradient descent update:

$$\mathbf{x}_1 = \mathbf{x}_0 - 0.1 \cdot \nabla f(\mathbf{x}_0)$$

$$= \begin{bmatrix} 1 \\ -1 \end{bmatrix} - 0.1 \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 - 0.1 \\ -1 - (-0.1) \end{bmatrix}$$

$$= \begin{bmatrix} 0.9 \\ -0.9 \end{bmatrix}.$$

*(Why? step is ($x - \eta \nabla f$); multiply gradient by step size and subtract componentwise.)*

so,

$$\boxed{\mathbf{x}_1 = \begin{bmatrix} 0.9 \\ -0.9 \end{bmatrix}.}$$

---

**Step 1 → Step 2**

Evaluate the gradient at ($\mathbf{x}_1$):

$$\nabla f(\mathbf{x}_1) = \begin{bmatrix} 2(0.9) + (-0.9) \\ 0.9 + 2(-0.9) \end{bmatrix}$$

$$= \begin{bmatrix} 1.8 - 0.9 \\ 0.9 - 1.8 \end{bmatrix}$$

$$= \begin{bmatrix} 0.9 \\ -0.9 \end{bmatrix}.$$

*(Why? again substitute into gradient; note it equals (0.9) times ($[1, -1]^T$).)*

Update:

$$\mathbf{x}_2 = \mathbf{x}_1 - 0.1 \cdot \nabla f(\mathbf{x}_1)$$

$$= \begin{bmatrix} 0.9 \\ -0.9 \end{bmatrix} - 0.1 \begin{bmatrix} 0.9 \\ -0.9 \end{bmatrix}$$

$$= \begin{bmatrix} 0.9 - 0.09 \\ -0.9 - (-0.09) \end{bmatrix}$$
$$= \begin{bmatrix} 0.81 \\ -0.81 \end{bmatrix}.$$

*(Why? same update; arithmetic: ($0.1 \times 0.9 = 0.09$), subtract componentwise.)*

$$\boxed{\mathbf{x}_2 = \begin{bmatrix} 0.81 \\ -0.81 \end{bmatrix}.}$$

**Remark/pattern:** the iterates scale by (0.9) each step along the direction $([1, -1]^T)$ because the gradient is linear in ($\mathbf{x}$) and the initial vector lies on an eigen-direction of that linear map (see part (c)).

---

## (c) Matrix form and identification of (A).

Write ($\nabla f(\mathbf{x})$) as a matrix times ($\mathbf{x}$). From part (a),

$$\nabla f(x, y) = \begin{bmatrix} 2x + y \\ x + 2y \end{bmatrix}$$
$$= \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}.$$

So the matrix is

$$\boxed{A = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}.}$$

Therefore the gradient-descent iteration becomes

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \eta A \mathbf{x}_k = (I - \eta A)\mathbf{x}_k.$$

With ($\eta = 0.1$) we have

$$I - \eta A = I - 0.1 \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$$
$$= \begin{bmatrix} 1 - 0.2 & -0.1 \\ -0.1 & 1 - 0.2 \end{bmatrix}$$
$$= \begin{bmatrix} 0.8 & -0.1 \\ -0.1 & 0.8 \end{bmatrix}.$$

Check:

$$((I - \eta A)\mathbf{x}_0 = \begin{bmatrix} 0.8 & -0.1 \\ -0.1 & 0.8 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \end{bmatrix} = \begin{bmatrix} 0.8 + 0.1 \\ -0.1 - 0.8 \end{bmatrix} = \begin{bmatrix} 0.9 \\ -0.9 \end{bmatrix})$$

, which matches ($\mathbf{x}_1$). Re-applying gives ($\mathbf{x}_2$).

*(Why this matrix form? Because (f) is a quadratic ($= \frac{1}{2}\mathbf{x}^T(2A)\mathbf{x}$) up to constants; in general for quadratic forms the gradient is linear: ($\nabla f(\mathbf{x}) = A\mathbf{x}$), so gradient descent is multiplication by ($I - \eta A$).)*

---

## Final answers (boxed)

$$(\nabla f(x,y) = \begin{bmatrix} 2x+y \\ x+2y \end{bmatrix}.)$$

$$(\mathbf{x}_1 = \begin{bmatrix} 0.9 \\ -0.9 \end{bmatrix}, \qquad \mathbf{x}_2 = \begin{bmatrix} 0.81 \\ -0.81 \end{bmatrix}.)$$

$$(A = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}, \quad I - \eta A = \begin{bmatrix} 0.8 & -0.1 \\ -0.1 & 0.8 \end{bmatrix} \text{ for } \eta = 0.1.)$$

---

# Problem 2

Consider the 1D function

$f(x) = x^4 - 2x^2.$

(a) Compute (f'(x)) and write the gradient-descent update ($x_{k+1} = x_k - \eta f'(x_k)$). Assume ($x_0$ =0) and ($\eta$=0.1); write 2 steps.

(b) With ($x_0$=0.1) and ($\eta$=0.1), compute two iterations ($x_1, x_2$).

(c) With ($x_0$=-0.1) and ($\eta$=0.1), compute two iterations ($x_1, x_2$).

(d) Comment on why GD can appear to "stall" near (x=0) even though (x=0) is not a minimum, and explain the differences in direction in cases (b) and (c).

---

## (a) Derivative and update rule

Compute derivative:

$$f'(x) = \frac{d}{dx}(x^4 - 2x^2) = 4x^3 - 4x = 4x(x^2 - 1).$$

*(Why? Differentiate termwise: $d(x^4)/dx = 4x^3$, $d(-2x^2)/dx = -4x$).*

Gradient-descent update (1D):

$$x_{k+1} = x_k - \eta f'(x_k) = x_k - \eta\big(4x_k^3 - 4x_k\big).$$

*(Why? Standard GD: move opposite the derivative scaled by step size ($\eta$).)*

Now with ($x_0$=0) and ($\eta$=0.1):

- Evaluate ($f'(0) = 4 \cdot 0 \cdot (0^2 - 1) = 0$).

  *(Why? because (x=0) makes factor (4x) zero.)*

- Update ($x_1$ = 0 - 0.1·0 = 0.)
- Then ($x_2 = x_1 - 0.1 \cdot f'(x_1) = 0 - 0.1 \cdot 0 = 0$)

$$\boxed{x_1 = 0, \qquad x_2 = 0.}$$

*(Why this happens: if gradient is **exactly** zero, GD does not move.)*

---

## (b) ($x_0 = 0.1,\ \eta = 0.1$) — two iterations

We use ($f'(x) = 4x(x^2 - 1)$).

**Step 0 → 1**

- Compute $f'(0.1) = 4(0.1)((0.1)^2 - 1) = 0.4(0.01 - 1) = 0.4(-0.99) = -0.396.$

  *(Why? substitute (x=0.1) into derivative.)*

- Update:

  $x_1 = 0.1 - 0.1 \cdot (-0.396) = 0.1 + 0.0396 = 0.1396.$

**Step 1 → 2**

- Compute ($x_1^2 = (0.1396)^2 = 0.01948816.$)

  Then ($x_1^2 - 1 = -0.98051184.$)

- Compute ($f'(0.1396) = 4(0.1396)(-0.98051184).$)

  First ($4 \cdot 0.1396 = 0.5584$).

  Then,

  $$0.5584 \times 0.98051184 \approx 0.547517811456,$$

  so ($f'(0.1396) \approx -0.547517811456$) (negative).

  *(Why? algebraic multiplication; sign negative because (($x^2$-1)<0) here.)*

- Update:

  $$x_2 = 0.1396 - 0.1 \cdot (-0.547517811456)$$
  $$= 0.1396 + 0.0547517811456 \approx 0.194351781146.$$

So (rounded reasonably)

$$\boxed{x_1 \approx 0.1396, \qquad x_2 \approx 0.19435178.}$$

*(Why these steps: evaluate derivative at current point, multiply by ($\eta$), subtract; signs determine whether you move left or right.)*

---

## (c) ($x_0 = -0.1,\ \eta = 0.1$) — two iterations

This is symmetric to part (b) because (f'(x)) is an odd function in (x) (check: ($f'(-x) = -f'(x)$ )).

**Step 0 → 1**

- $f'(-0.1) = 4(-0.1)(0.01 - 1) = -0.4(-0.99) = +0.396.$
- Update:

  $x_1 = -0.1 - 0.1 \cdot (0.396) = -0.1 - 0.0396 = -0.1396.$

**Step 1 → 2**

- We already computed for (0.1396) that ($f'(0.1396) \approx -0.547517811456$). Using oddness, ($f'(-0.1396) \approx +0.547517811456.$)
- Update:

  $x_2 = -0.1396 - 0.1 \cdot (0.547517811456)$

  $= -0.1396 - 0.0547517811456 \approx -0.194351781146.$

So

$$\boxed{x_1 \approx -0.1396, \qquad x_2 \approx -0.19435178.}$$

*(Why symmetric? because the function and its derivative are symmetric/odd, so positive starts move positive, negative starts move negative.)*

---

## (d) Comments: "stall" at (x=0), stability and direction differences

1. **Why GD can appear to stall at (x=0):**

   (x=0) is a stationary point because (f'(0)=0). Gradient descent updates are ($x_{k+1} = x_k - \eta f'(x_k)$). If ($f'(x_k)$=0) exactly, then the update is zero and the iterate **stops**. That is a purely mechanical reason: zero gradient ⇒ no move.

   However, a vanishing gradient does **not** imply a minimum — it could be a maximum or saddle. For this function the critical points are at (x=0,$\pm$ 1). Check second derivative:

   $$f''(x) = 12x^2 - 4, \qquad f''(0) = -4 < 0,$$

   so (x=0) is a local **maximum** (not a minimum). GD at exactly (x=0) will “stall” (stay at 0) because the method uses only first derivative information.

2. **Why starting slightly away from zero moves you away (cases (b) and (c)):**

   If you start at a small nonzero (x), the derivative is small but nonzero and points away from 0 (its sign pushes (x) further from 0). For small (x), linearizing $(f'(x)) gives (f'(x) \approx -4x)$ (dominant term), so the GD update is approximately

   $x_{k+1} \approx x_k - \eta(-4x_k) = (1 + 4\eta)x_k.$

   With $(\eta = 0.1), (1 + 4\eta = 1.4 > 1)$, so the magnitude of (x) **grows** away from 0 (geometric growth) until other nonlinear terms change behavior. That is why in (b) the positive start moves to larger positive values, and in (c) the negative start moves to larger negative values — directions are preserved (sign of (x) remains the same).

3. **When does GD actually "stall" (slow progress) vs. move?**

   - Exact stall occurs only if you land exactly on a point where gradient is zero (machine-exact zero) — then no update.
   - Slow progress (apparent stall) happens in **flat** regions where gradients are tiny (so updates $(\eta f')$ are tiny). Which effect dominates depends on the function shape and the step size $(\eta)$. In this specific (f), for very small nonzero (x) the multiplicative factor $(1 + 4\eta)$ causes growth rather than slow decay, but with a much smaller $(\eta)$ the immediate change per step could be very small and appear slow.

4. **Where do iterates ultimately go?**

   For this function the minima are at $(x = \pm 1)$ (since (f'(x)=0) also at $(\pm 1)$ and $(f''(\pm 1) = 8 > 0)$). Starting near (0) but not exactly zero, iterates will be pushed away from 0 and (with a suitable (\eta)) can move toward one of the minima (+1) or (-1) depending on sign of the initial point.

---

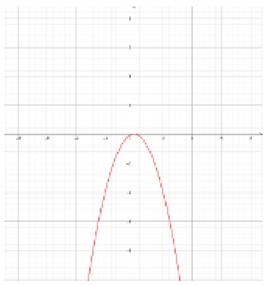
## Final numeric summary

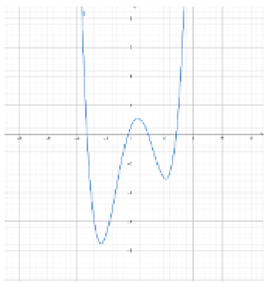
- $(a) With (x_0 = 0,\ \eta = 0.1:\ x_1 = 0,\ x_2 = 0.)$
- $(b) With (x_0 = 0.1,\ \eta = 0.1:\ x_1 \approx 0.1396,\ x_2 \approx 0.19435178.)$
- $(c) With (x_0 = -0.1,\ \eta = 0.1:\ x_1 \approx -0.1396,\ x_2 \approx -0.19435178.)$

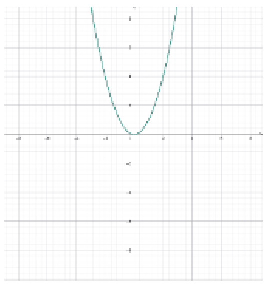
---

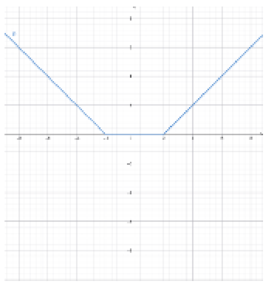
# Problem 3

1. Consider the four figures (a)–(d) shown below. For each figure, indicate whether the underlying function can have a **unique minimiser that gradient descent will converge to** (for a suitable fixed step size) and result in a single minima (unique value rounded to assumed tolerance). Answer: for each (a)–(d), write "Yes/No".


(a)


(b)


(c)


(d)

### Answers (a)–(d) with explanations

**(a) — No.**

The plot in (a) looks like a concave “cap” (a local **maximum** shape), not a bowl. A concave function does not have a unique global minimiser in the usual sense (it tends to (-∞) toward the sides or has minima at the boundaries), and gradient-descent updates will not converge to a well-defined interior minimiser. So you cannot expect a unique minimiser that GD converges to.

**(b) — No.**

Picture (b) shows a non-convex, multimodal shape with several local minima and maxima. For such functions, GD typically converges to *one of the local minima,* depending on initialization — different starts give different limits. Therefore, there is **no single unique minimiser** that GD will always find for all starts (so the answer is No).

**(c) — Yes.**

Figure (c) is a smooth convex “bowl” (like a parabola). A convex, smooth function has a single global minimiser. For a suitable fixed step size (small enough to ensure stability), gradient descent will converge to that unique minimum. So (c) is **Yes**.

**(d) — Yes (with a caveat).**

Figure (d) is a V-shaped convex function (like ($|x|$)), which is convex and has a unique minimiser at the cusp (the vertex). The gradient is not defined exactly at the cusp, but the function is convex and has a unique minimiser; a subgradient method or GD applied with appropriate handling (or a sufficiently small fixed step size) will drive iterates into a small neighbourhood of the minimiser and produce a single value within any reasonable tolerance. So the correct answer for the existence of a unique minimiser and convergence (for a suitable fixed step) is **Yes**, noting the practical detail that one must handle the nondifferentiable point (or use subgradient methods) — otherwise standard GD can bounce around the cusp but still settle within tolerance for a suitable step size.

# Problem 4

A dataset contains 5,000,000 data points, of which 40% are used for training.

A researcher uses mini-batch gradient descent with:

- Batch size = 0.2% of the training data
- Learning rate η = 0.1

(a) How many samples are used in the training set?

(b) How many samples are in each mini-batch?

(c) How many weight updates (iterations) are performed in one epoch?

(d) If training runs for 20 epochs, how many total updates of **w** occur?

---

## (a) How many samples are used in the training set?

The training set is 40% of the total 5,000,000 data points:

$$\text{Training samples} = 0.40 \times 5,000,000 = 2,000,000.$$

**Answer:**

$$\boxed{2,000,000 \text{ training samples}}$$

*(Why? Multiply the total dataset size by the % used for training.)*

---

## (b) How many samples are in each mini-batch?

Batch size = 0.2% of the training data:

$$\text{Mini-batch size} = 0.002 \times 2,000,000 = 4,000.$$

**Answer:**

$$\boxed{4,000 \text{ samples per mini-batch}}$$

*(Why? Convert 0.2% to decimal = 0.002, multiply by training size.)*

---

## (c) How many weight updates per epoch?

One epoch means going once through all 2,000,000 training samples.

$$\text{Updates per epoch} = \frac{\text{training samples}}{\text{batch size}} = \frac{2,000,000}{4,000} = 500.$$

**Answer:**

$$\boxed{500 \text{ updates per epoch}}$$

*(Why? Each batch produces one update. Divide total samples by batch size.)*

### (d) Total weight updates for 20 epochs

$$\text{Total updates} = 20 \times 500 = 10,000.$$

**Answer:**

$$\boxed{10,000 \text{ total updates of } w}$$

*(Why? Number of epochs × updates per epoch.)*

## Problem 5

Consider the binary classification problem of predicting whether a person will buy a product (Yes/No) based on their characteristics using the Naïve Bayes classifier.

Dataset (10 examples):

| Age | Income | Student | Credit Rating | Buys Product (Y/N) |
|---|---|---|---|---|
| ≤ 30 | High | No | Fair | No |
| ≤ 30 | High | No | Excellent | No |
| 31–40 | High | No | Fair | Yes |
| > 40 | Medium | No | Fair | Yes |
| > 40 | Low | Yes | Fair | Yes |
| > 40 | Low | Yes | Excellent | No |
| 31–40 | Low | Yes | Excellent | Yes |
| ≤ 30 | Medium | No | Fair | No |
| ≤ 30 | Low | Yes | Fair | Yes |
| > 40 | Medium | Yes | Fair | Yes |

Predict whether a **31–40** year-old with **Medium** income, who **is a Student** and has **Excellent** credit rating will buy the product.

We want ($P(\text{Buy} = \text{Yes} \mid \text{features})$) vs ($P(\text{No} \mid \text{features})$) under Naïve Bayes.

### Step 1 — Prior class probabilities

Count how many Yes / No in the dataset:

- Number of **Yes**: 6 (rows 3,4,5,7,9,10).
- Number of **No**: 4 (rows 1,2,6,8).

So

$$P(\text{Yes}) = \frac{6}{10} = 0.6, \qquad P(\text{No}) = \frac{4}{10} = 0.4.$$

*(Why? Priors = class frequencies.)*

---

## Step 2 — Count conditional frequencies for each feature

We need for the query ($x = (\text{Age} = 31!-!40,\ \text{Income} = Medium,\ \text{Student} = Yes,\ \text{Credit} = Excellent)$) the conditional probabilities ($P(\text{feature}|,\text{Yes})$) and ($P(\text{feature}|,\text{No})$).

**Counts among Yes (6 examples):**

List Yes examples (age, income, student, credit):

- (31–40, High, No, Fair)
- (>40, Medium, No, Fair)
- (>40, Low, Yes, Fair)
- (31–40, Low, Yes, Excellent)
- (≤30, Low, Yes, Fair)
- (>40, Medium, Yes, Fair)

From these 6:

- Age = 31–40: appears **2** times → ($P(\text{Age} = 31!-!40, |, \text{Yes}) = 2/6$).
- Income = Medium: appears **2** times → ($P(\text{Income} = Medium, |, \text{Yes}) = 2/6$).
- Student = Yes: appears **4** times → ($P(\text{Student} = Yes, |, \text{Yes}) = 4/6$).
- Credit = Excellent: appears **1** time → ($P(\text{Credit} = Excellent, |, \text{Yes}) = 1/6$).

**Counts among No (4 examples):**

List No examples (age, income, student, credit):

- (≤30, High, No, Fair)
- (≤30, High, No, Excellent)
- (>40, Low, Yes, Excellent)
- (≤30, Medium, No, Fair)

From these 4:

- Age = 31–40: appears **0** times → ($P(\text{Age} = 31!-!40, |, \text{No}) = 0/4 = 0$.)

- Income = Medium: appears **1** time → ($P(\text{Income} = Medium, |, \text{No}) = 1/4.$)
- Student = Yes: appears **1** time → ($P(\text{Student} = Yes, |, \text{No}) = 1/4.$)
- Credit = Excellent: appears **2** times → ($P(\text{Credit} = Excellent, |, \text{No}) = 2/4 = 1/2.$)

*(Why? Naïve Bayes treats each conditional independently; we use class-conditional frequencies.)*

---

## Step 3 — Compute (unnormalized) posterior without smoothing

Naïve Bayes (conditional independence) gives

$$P(\text{Yes} \mid x) \propto P(\text{Yes}) \prod_i P(x_i \mid \text{Yes}),$$

and similarly for No.

Plugging numbers **without smoothing**:

- For **Yes**:

$$P(\text{Yes} \mid x) \propto 0.6 \times \frac{2}{6} \times \frac{2}{6} \times \frac{4}{6} \times \frac{1}{6}.$$

  Multiply: nonzero small, positive value ≈ ($0.6 \times (16/1296) \approx 0.00741.$)

- For **No**:

$$P(\text{No} \mid x) \propto 0.4 \times \underbrace{0}_{P(\text{Age}=31-40|\text{No})} \times \frac{1}{4} \times \frac{1}{4} \times \frac{2}{4} = 0.$$

  Because ($P(\text{Age} = 31!-!40|\text{No}) = 0$), the whole product is zero.

**Decision (no smoothing):** ($P(\text{No} \mid x) = 0$), so the classifier picks **Yes** (all posterior mass goes to Yes).

*(Why? Direct Naïve Bayes multiplies conditional probabilities — a single zero factor forces that class probability to zero.)*

---

## Step 4 — Discussion about zero probabilities and Laplace smoothing

A zero conditional probability is often undesirable (it makes a class impossible regardless of other evidence). A common fix is **Laplace (add-one) smoothing**: add 1 to every count and add the number of categories to the denominator.

I'll show the **add-one smoothed** result for comparison.

- For categorical features, number of possible values:
  - Age has 3 values (≤30, 31–40, >40).
  - Income has 3 values (High, Medium, Low).
  - Student has 2 values (Yes, No).
  - Credit has 2 values (Fair, Excellent).

Apply add-one smoothing (k = 1):

**Smoothed conditionals for Yes class (denominator = 6 + #values):**

- $(P(\text{Age} = 31!-!40|\text{Yes}) = (2+1)/(6+3) = 3/9 = 1/3.)$
- $(P(\text{Income} = Medium|\text{Yes}) = (2+1)/(6+3) = 3/9 = 1/3.)$
- $(P(\text{Student} = Yes|\text{Yes}) = (4+1)/(6+2) = 5/8.)$
- $(P(\text{Credit} = Excellent|\text{Yes}) = (1+1)/(6+2) = 2/8 = 1/4.)$

**Smoothed conditionals for No class (denominator = 4 + #values):**

- $(P(\text{Age} = 31!-!40|\text{No}) = (0+1)/(4+3) = 1/7.)$
- $(P(\text{Income} = Medium|\text{No}) = (1+1)/(4+3) = 2/7.)$
- $(P(\text{Student} = Yes|\text{No}) = (1+1)/(4+2) = 2/6 = 1/3.)$
- $(P(\text{Credit} = Excellent|\text{No}) = (2+1)/(4+2) = 3/6 = 1/2.)$

Now compute smoothed unnormalized posteriors:

- For **Yes** (smoothed):

$$\tilde{P}(\text{Yes} \mid x) = 0.6 \times \frac{1}{3} \times \frac{1}{3} \times \frac{5}{8} \times \frac{1}{4} \approx 0.00260417.$$

- For **No** (smoothed):

$$\tilde{P}(\text{No} \mid x) = 0.4 \times \frac{1}{7} \times \frac{2}{7} \times \frac{1}{3} \times \frac{1}{2} \approx 0.00272108.$$

Normalize to compare:

$$P_{\text{smoothed}}(\text{No} \mid x) \approx \frac{0.00272108}{0.00272108 + 0.00260417} \approx 0.511,$$

$$P_{\text{smoothed}}(\text{Yes} \mid x) \approx 0.489.$$

**Decision (with Laplace smoothing):** slightly favors **No** (≈51.1% vs 48.9%).

*(Why show both? Without smoothing, a single zero count forces a 0 posterior for a class, which can be misleading when data are sparse. Laplace smoothing avoids zeros and often gives more sensible, robust results.)*

---

## Final answer/takeaway

- **Using plain Naïve Bayes (no smoothing):** the classifier predicts **Yes** (because ($P(\text{Age} = 31-40|\text{No}) = 0) makes (P(\text{No} \mid x) = 0)$).
- **Using Laplace (add-one) smoothing,** the classifier predicts **No** (slightly higher posterior ≈ 51.1% vs 48.9%).

**Practical recommendation:** use Laplace (or another smoothing) in small datasets to avoid zero-probability artifacts. For your notes, you can state both results and explain why smoothing changes the prediction.

---

# Problem 6

Consider a binary classification problem modeled using Gaussian Discriminant Analysis (GDA). Suppose we have two classes with parameters

$$\mu_0 = \begin{bmatrix} 1 \\ 2 \end{bmatrix}, \qquad \mu_1 = \begin{bmatrix} 3 \\ 4 \end{bmatrix}, \qquad \Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \qquad \phi = 0.5.$$

For a new observation

$$x = \begin{bmatrix} 2 \\ 3 \end{bmatrix},$$

compute the posterior probability ($P(y = 1 \mid x)$) and determine the predicted class.

(a) Compute the Gaussian likelihoods ($p(x \mid y = 0)$) and ($p(x \mid y = 1)$).

(b) Using Bayes' rule, compute ($P(y = 1 \mid x)$).

(c) Determine the predicted class based on which posterior is greater.

(d) Interpret the result geometrically in terms of the decision boundary between the two classes.

---

### Useful formula (why we use it)

For a (d)-dimensional Gaussian with mean ($\mu$) and covariance ($\Sigma$),

$$p(x) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp!\Big( -\tfrac{1}{2}(x-\mu)^\top \Sigma^{-1}(x-\mu)\Big).$$

Here (d=2), ($\Sigma = I$) so ($|\Sigma| = 1$) and ($\Sigma^{-1} = I$). Hence the normalizing constant is ($\frac{1}{2\pi}$).

---

## (a) Compute ($p(x \mid y=0)$) and ($p(x \mid y=1)$)

Compute the residuals and squared Mahalanobis distances:

- For class 0: ($x - \mu_0 = \begin{bmatrix}2\\3\end{bmatrix} - \begin{bmatrix}1\\2\end{bmatrix} = \begin{bmatrix}1\\1\end{bmatrix}$.)

  Squared norm: ($(x-\mu_0)^\top(x-\mu_0) = 1^2 + 1^2 = 2$.)
- For class 1: ($x - \mu_1 = \begin{bmatrix}2\\3\end{bmatrix} - \begin{bmatrix}3\\4\end{bmatrix} = \begin{bmatrix}-1\\-1\end{bmatrix}$.)

  Squared norm: ($(x-\mu_1)^\top(x-\mu_1) = (-1)^2 + (-1)^2 = 2$.)

(Why? With ($\Sigma = I$) the exponent is ($-\frac{1}{2}$) times the squared Euclidean distance.)

Therefore both classes have the same Mahalanobis distance of 2, so their likelihoods are equal:

$p(x \mid y=0) = p(x \mid y=1) = \frac{1}{2\pi}\exp!\big(-\frac{1}{2}\cdot 2\big) = \frac{1}{2\pi}e^{-1}.$

Numeric value:

$\frac{1}{2\pi}e^{-1} \approx 0.15915494 \times 0.36787944 \approx 0.0585498.$

**Answer (a):**

$$\boxed{p(x \mid y=0) = p(x \mid y=1) = \frac{1}{2\pi}e^{-1} \approx 0.05855.}$$

---

## (b) Compute posterior ($P(y=1 \mid x)$) (Bayes' rule)

Bayes' rule:

$$P(y=1 \mid x) = \frac{p(x \mid y=1)P(y=1)}{p(x \mid y=0)P(y=0) + p(x \mid y=1)P(y=1)}.$$

Given ($P(y=1) = \phi = 0.5$) and ($P(y=0) = 0.5$), and the likelihoods are equal, the numerator and denominator simplify:

$$P(y=1 \mid x) = \frac{p \cdot 0.5}{p\cdot 0.5 + p \cdot 0.5} = \frac{0.5}{0.5+0.5} = \frac{1}{2}.$$

So

$$\boxed{P(y=1 \mid x) = 0.5.}$$

(Why? equal likelihoods and equal priors give equal posterior mass.)

---

## (c) Predicted class

Since ($P(y=1 \mid x) = P(y=0 \mid x) = 0.5$), neither posterior is greater **— we have a tie**.

Practical choices:

- You can declare the classifier **undecided** (on the decision boundary).
- Many implementations break ties by a fixed rule (e.g., choose the class with larger prior, or choose class 1 by convention). With equal priors, that still leaves a tie; one might then choose class 0 or class 1 arbitrarily.

**Conclusion:** mathematically there is **no strict winner**; (x) lies exactly on the decision boundary, so the prediction is ambiguous (both classes equally likely). If forced to pick, you must specify a tie-breaking rule.

$$\boxed{\text{No strict predicted class (tie): } P(y{=}1 \mid x) = P(y{=}0 \mid x) = 0.5.}$$

---

## (d) Geometric interpretation (decision boundary)

For GDA with shared covariance ($\Sigma = I$) and priors ($\phi$), the decision boundary between classes is linear (a hyperplane). In particular, the log-odds decision rule simplifies to

$$(\mu_1 - \mu_0)^\top x = \tfrac{1}{2}(\mu_1^\top \mu_1 - \mu_0^\top \mu_0) + \log \frac{P(y=1)}{P(y=0)}.$$

With equal priors the log ratio term is zero. Compute the numbers:

- ($\mu_1 - \mu_0 = \begin{bmatrix} 2 \\ 2 \end{bmatrix}$.)
- ($\mu_1^\top \mu_1 = 3^2 + 4^2 = 25, \qquad \mu_0^\top \mu_0 = 1^2 + 2^2 = 5.$)
- $RHS = \tfrac{1}{2}(25 - 5) = 10.$

So the boundary is

$$[2\ 2] \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = 10 \quad \Longleftrightarrow \quad x_1 + x_2 = 5.$$

Our point ($x = [2, 3]^\top$) satisfies (2+3=5), so it lies **exactly on the decision boundary** — hence equal posterior probabilities.

Geometric summary:

- The decision boundary is the line ($x_1 + x_2$=5), which is perpendicular to vector ($\mu_1 - \mu_0 = [2, 2]^\top$).
- Points on one side of this line are classified as class 1, points on the other side as class 0.
- The given (x) is the midpoint along the direction between the two means (equidistant from ($\mu_0$) and ($\mu_1$)), so it lies on the boundary.

I'll break this into 4 very simple steps:

---

## Step 1 — What is the decision boundary in GDA?

In Gaussian Discriminant Analysis with **shared covariance matrix** ($\Sigma$),

- Each class is modeled by a Gaussian with mean ($\mu_0$) and ($\mu_1$)
- The covariance matrix ($\Sigma$) is the same for both classes
- The decision rule is:

  **Choose the class whose Gaussian assigns a higher probability to x**.

The **decision boundary** is the set of points where the two class probabilities are *equal*:

$$p(x \mid y = 0)P(y = 0) = p(x \mid y = 1)P(y = 1).$$

This is NOT a formula — it's just the definition:

**Points where the classifier is "undecided."**

---

## Step 2 — Because Σ = I (identity), likelihood depends only on distance to the mean

Since ($\Sigma = I$), the Gaussian density simplifies to:

$$p(x \mid y = k) = C \cdot \exp\left(-\tfrac{1}{2}|x - \mu_k|^2\right),$$

where ($C = \frac{1}{2\pi}$).

That means:

A point x is more likely under class 1 than class 0 IF it is closer (in Euclidean distance) to $\mu_1$ than to $\mu_0$.

That's it.

No complicated formula — just comparing distances.

Thus the decision boundary is:

$$|x - \mu_0|^2 = |x - \mu_1|^2.$$

This is the key geometric idea.

## Step 3 — Expand the distance equality (no tricks)

Take the condition:

$$|x - \mu_0|^2 = |x - \mu_1|^2.$$

Expand both sides:

Left side:

$$(x - \mu_0)^\top (x - \mu_0) = x^\top x - 2\mu_0^\top x + \mu_0^\top \mu_0.$$

Right side:

$$(x - \mu_1)^\top (x - \mu_1) = x^\top x - 2\mu_1^\top x + \mu_1^\top \mu_1.$$

Set them equal:

$$x^\top x - 2\mu_0^\top x + \mu_0^\top \mu_0 = x^\top x - 2\mu_1^\top x + \mu_1^\top \mu_1$$

Cancel ($x^\top x$) on both sides:

$$-2\mu_0^\top x + \mu_0^\top \mu_0 = -2\mu_1^\top x + \mu_1^\top \mu_1$$

Rearrange:

$$(\mu_1 - \mu_0)^\top x = \tfrac{1}{2}\big(\mu_1^\top \mu_1 - \mu_0^\top \mu_0\big).$$

This equation describes a **line** (a linear decision boundary).

## Step 4 — Plug in the actual numbers

You were given:

$$\mu_0 = \begin{bmatrix}1\ 2\end{bmatrix}, \quad \mu_1 = \begin{bmatrix}3\ 4\end{bmatrix}$$

Compute the difference:

$$\mu_1 - \mu_0 = \begin{bmatrix}2\ 2\end{bmatrix}$$

Compute the constants:

$$\mu_1^\top \mu_1 = 3^2 + 4^2 = 25, \quad \mu_0^\top \mu_0 = 1^2 + 2^2 = 5.$$

Plug into the boundary formula:

$$\begin{bmatrix}2 & 2\end{bmatrix} x = \tfrac{1}{2}(25 - 5) = 10.$$

Rewrite:

$$2x_1 + 2x_2 = 10 \quad \iff \quad x_1 + x_2 = 5.$$

**This is the line that separates the classes.**

## Step 5 — Why is the decision boundary the line $x_1 + x_2 = 5$?

Because:

- Points with **x₁ + x₂ > 5** are closer to ($\mu_1$=[3,4]), so classify as **class 1**
- Points with **x₁ + x₂ < 5** are closer to ($\mu_0$=[1,2]), so classify as **class 0**
- Points with **x₁ + x₂ = 5** are exactly **equidistant from both means**, so both classes are equally likely.

Your point:

$$x = \begin{bmatrix} 2 \\ 3 \end{bmatrix}$$

satisfies:

2 + 3 = 5,

so it lies **exactly on** the decision boundary → both class probabilities = 0.5.

---

## Summary (the geometric intuition)

- GDA with shared covariance compares **distances to the means**.
- Decision boundary = points equidistant from ($\mu_0$) and ($\mu_1$).
- With ($\Sigma = I$), the boundary is a straight line halfway between the means.
- For this problem the boundary is

  $$x_1 + x_2 = 5,$$

  which is a line perpendicular to the vector ($\mu_1 - \mu_0$ = [2,2]).
- Your test point lies exactly on this line → probability = 0.5 → tie.

---

### Final concise answers

- (a) ($p(x \mid y = 0) = p(x \mid y = 1) = \frac{1}{2\pi} e^{-1} \approx 0.05855.$)
- (b) ($P(y = 1 \mid x) = 0.5.$)
- (c) No strict predicted class (tie); (x) lies on the decision boundary.
- (d) Decision boundary is the line ($x_1 + x_2 = 5$); (x=(2,3)) satisfies this, hence it is equidistant from the two class means.

---

# Problem 7

Explain the core assumptions of Gaussian Discriminant Analysis (GDA). How does GDA differ from Logistic Regression in terms of modeling approach, assumptions, and decision boundaries?

---

## ✅ Core Assumptions of Gaussian Discriminant Analysis (GDA)

GDA is a **generative model**.

It models **how the data is generated** for each class.

It makes **three main assumptions**:

---

## 1. Class-conditional features follow a multivariate Gaussian distribution

For each class ( $y \in 0, 1$ ), the feature vector (x) is assumed to be drawn from a Gaussian:

$$x \mid y = 0 \sim \mathcal{N}(\mu_0, \Sigma), \qquad x \mid y = 1 \sim \mathcal{N}(\mu_1, \Sigma).$$

This says:

- Each class has its **own mean vector**.
- All points in a class cluster around that mean, following a **Gaussian-shaped distribution**.

---

## 2. Shared covariance matrix between classes

GDA assumes:

$$\Sigma_0 = \Sigma_1 = \Sigma.$$

Meaning:

- Both classes have the same shape, orientation, and spread.
- The only difference between classes is their means, not their variability.

(If covariances differ, the model becomes **Quadratic Discriminant Analysis (QDA)**.)

---

## 3. Prior class probabilities are Bernoulli

Class labels are assumed to follow:

$$P(y = 1) = \phi, \qquad P(y = 0) = 1 - \phi.$$

This means:

- Labels come from a Bernoulli distribution.
- ($\phi$) is estimated from the proportion of positive samples.

---

### Why these assumptions matter

They allow GDA to compute:

$$P(x \mid y), \quad P(y), \quad \text{and then } P(y \mid x)$$

using Bayes′ rule:

$$P(y=1 \mid x) = \frac{p(x \mid y=1)\phi}{p(x \mid y=0)(1-\phi) + p(x \mid y=1)\phi}.$$

## ✅ How GDA Differs from Logistic Regression

We compare them across **three major dimensions**:

### 1. Modeling Approach

### GDA → Generative model

GDA models the **data generation process**:

$$p(x \mid y), \quad p(y)$$

Then applies Bayes' rule to find:

$$p(y \mid x)$$

GDA asks:

> "How is data for each class distributed?"

### Logistic Regression → Discriminative model

Logistic regression directly models:

$$p(y \mid x)$$

It makes **no assumptions** about ($p(x)$) or ($p(x \mid y)$).

Logistic regression asks:

> "How does the boundary between classes look?"

## 2. Assumptions

### Assumptions in GDA

- Class conditional distributions are Gaussian.
- Both classes share the same covariance matrix.
- Priors are Bernoulli.
- Data of each class cluster around a mean.

These assumptions are **strong** — they shape the form of the decision boundary.

## Assumptions in Logistic Regression

Only assumptions:

- Linearity in the **log-odds**:

$$\log \frac{P(y=1 \mid x)}{P(y=0 \mid x)} = \theta^\top x$$

- No functional assumption on the distribution of (x) itself.
- Does **not** assume Gaussianity or equal covariances.

These assumptions are **much weaker**, since LR does not try to model the class distributions.

## 3. Decision Boundaries

**GDA**

Because ($\Sigma$) is shared,

$$\log \frac{p(x \mid y=1)}{p(x \mid y=0)}$$

becomes a **linear function of x**.

Thus, **GDA produces a linear decision boundary**.

BUT the decision boundary is:

- perpendicular to the direction connecting (\mu_0) and (\mu_1)
- located where the two Gaussian densities are equal

**Geometric interpretation**:

The boundary is the set of points **equidistant in Mahalanobis distance** from the two class means.

**Logistic Regression**

Also yields a **linear decision boundary**, because:

$$P(y=1 \mid x) = \sigma(\theta^\top x)$$

Decision boundary:

$\theta^\top x = 0$

Geometric interpretation:

- LR finds the hyperplane that best separates labels, maximizing likelihood.
- It does not care about the Gaussian structure or distances to class means.

## 4. When does GDA outperform Logistic Regression?

GDA wins when:

- Data for each class truly clusters like a Gaussian
- Covariances across classes are similar
- Feature distributions match GDA's model

In this case:

- GDA is **asymptotically more efficient**
- Needs fewer samples to estimate the optimal boundary

## 5. When does Logistic Regression outperform GDA?

Logistic regression wins when:

- Data is **not Gaussian**
- Covariance structures differ
- Class distributions are multimodal or irregular
- You want fewer modeling assumptions
- You have large datasets (LR is very robust)

## Summary Table

| Aspect | GDA | Logistic Regression |
|---|---|---|
| Type | **Generative** | **Discriminative** |
| Models | $(p(x \mid y)), (p(y))$ | $(p(y \mid x))$ |
| Assumes X distribution? | **Yes (Gaussian)** | **No** |
| Covariance | **Shared ($\Sigma$)** | Not used |
| Decision boundary | Linear (if shared ($\Sigma$)) | Linear |
| Boundary depends on | Means + covariance | ($\theta$) learned directly |
| More efficient when | Gaussian assumptions true | No distributional assumptions |
| Robust to violations | ❌ Sensitive | ✓ More robust |

## Final Intuition

### GDA

Models how each class *looks* in the feature space → compares distances → draws a boundary halfway between Gaussian clusters.

### Logistic Regression

Directly models the boundary that best separates the classes → ignores how the data was generated.

**Both produce linear boundaries**, but for *totally different reasons*.

---

## Problem 8

Clear — let's derive the GDA decision boundary step by step, then explain exactly how different assumptions on the covariance matrices ($\Sigma$) change the geometry of that boundary.

---

## Setup (two-class GDA)

Class priors ($P(y=1)=\pi_1; P(y=0)=\pi_0$).

Class-conditionals are Gaussian:

$$x \mid y=k \sim \mathcal{N}(\mu_k, \Sigma_k), \qquad k \in 0,1.$$

Decision boundary = set of (x) where posterior probabilities are equal (i.e. classifier is indifferent):

$$P(y=1 \mid x) = P(y=0 \mid x) \quad \Longleftrightarrow \quad p(x \mid y=1), \pi_1 = p(x \mid y=0), \pi_0.$$

We will take logs to simplify.

---

## General log-form (no assumption ($\Sigma_0 = \Sigma_1$) yet)

Gaussian density:

$$p(x \mid y=k) = \frac{1}{(2\pi)^{d/2}|\Sigma_k|^{1/2}} \exp\Big(-\tfrac{1}{2}(x-\mu_k)^\top \Sigma_k^{-1}(x-\mu_k)\Big).$$

Set equality and take natural log:

$$\log \pi_1 - \tfrac{1}{2}\log|\Sigma_1| - \tfrac{1}{2}(x-\mu_1)^\top \Sigma_1^{-1}(x-\mu_1)$$

$$=$$

$$\log \pi_0 - \tfrac{1}{2}\log|\Sigma_0| - \tfrac{1}{2}(x-\mu_0)^\top \Sigma_0^{-1}(x-\mu_0).$$

Move terms to one side and multiply by 2:

$$(x-\mu_0)^\top \Sigma_0^{-1}(x-\mu_0) - (x-\mu_1)^\top \Sigma_1^{-1}(x-\mu_1) = 2\log\frac{\pi_0}{\pi_1} + \log\frac{|\Sigma_0|}{|\Sigma_1|}.$$

Expand the quadratic forms (keep ($x^\top Ax$) style):

$$x^\top(\Sigma_0^{-1} - \Sigma_1^{-1})x - 2(\mu_0^\top\Sigma_0^{-1} - \mu_1^\top\Sigma_1^{-1})x + (\mu_0^\top\Sigma_0^{-1}\mu_0 - \mu_1^\top\Sigma_1^{-1}\mu_1) = 2\log\frac{\pi_0}{\pi_1} + \log\frac{|\Sigma_0|}{|\Sigma_1|}.$$

This is a **quadratic equation in (x)**. Hence:

- **If ($\Sigma_0 \neq \Sigma_1$):** the leftmost term ($x^\top(\Sigma_0^{-1} - \Sigma_1^{-1})x$) is (generically) nonzero → the boundary is **quadratic** (conic surface: ellipsoid/hyperbola/parabola in 2D; a quadric in higher-D).

  This is the *QDA* boundary.

---

## Special case: shared covariance ($\Sigma_0 = \Sigma_1 = \Sigma$) (standard GDA assumption)

If ($\Sigma_0 = \Sigma_1 = \Sigma$), the quadratic ($x^\top(\Sigma^{-1} - \Sigma^{-1})x$) term cancels and we get a **linear** equation in (x). Starting from the expanded form, cancel the quadratic term and simplify:

$$-2(\mu_0^\top\Sigma^{-1} - \mu_1^\top\Sigma^{-1})x + (\mu_0^\top\Sigma^{-1}\mu_0 - \mu_1^\top\Sigma^{-1}\mu_1) = 2\log\frac{\pi_0}{\pi_1}.$$

Rearrange to standard linear form (move constants to RHS and flip sign):

$$(\mu_1-\mu_0)^\top\Sigma^{-1}x = \tfrac{1}{2}\big(\mu_1^\top\Sigma^{-1}\mu_1 - \mu_0^\top\Sigma^{-1}\mu_0\big) + \log\frac{\pi_1}{\pi_0}.$$

### Boxed: linear decision boundary (hyperplane)

$$\boxed{(\mu_1-\mu_0)^\top\Sigma^{-1}x = \tfrac{1}{2}\big(\mu_1^\top\Sigma^{-1}\mu_1 - \mu_0^\top\Sigma^{-1}\mu_0\big) + \log\frac{\pi_1}{\pi_0}.}$$

Equivalently, write as

$w^\top x + b = 0 \quad \text{with} \quad w = \Sigma^{-1}(\mu_1-\mu_0), b = -\tfrac{1}{2}(\mu_1^\top\Sigma^{-1}\mu_1 - \mu_0^\top\Sigma^{-1}\mu_0) - \log\tfrac{\pi_1}{\pi_0}.$

**Geometric interpretation:**

- Normal vector to the separating hyperplane is ($w = \Sigma^{-1}(\mu_1-\mu_0)$).

So the direction separating the classes is the difference of means **reweighted by ($\Sigma^{-1}$)** (Mahalanobis scaling).

- The constant (offset) depends on the squared Mahalanobis norms of the means and the log prior ratio ($\log(\pi_1/\pi_0)$), so **priors shift the plane** along the normal.

---

## Important special sub-cases & intuition

1. **($\Sigma = \sigma^2 I$)** (isotropic covariance):

   ($\Sigma^{-1} = (1/\sigma^2)I \Rightarrow w \propto (\mu_1 - \mu_0)$). The boundary is the hyperplane perpendicular to the vector ($\mu_1 - \mu_0$). If priors are equal, it is the **(Euclidean) perpendicular bisector** of the two means (points equidistant to the means).

2. **($\Sigma$) diagonal but not scalar:**

   Coordinates are scaled by inverse variances: the boundary is perpendicular in the Mahalanobis metric (features with small variance get a larger weight).

3. **($\Sigma_0 \neq \Sigma_1$)** (QDA):

   The ($x^\top(\Sigma_0^{-1} - \Sigma_1^{-1})x$) term makes the boundary quadratic. That boundary can be an ellipse, hyperbola, parabola, etc., depending on the matrices — geometrically, it means the classifier compares **Mahalanobis distances using different metrics for each class**; the locus where the weighted squared-distance difference equals a constant is generally quadratic.

---

## Summary — how ($\Sigma$) affects the shape

- **Shared covariance (($\Sigma_0 = \Sigma_1$))** → **Linear** decision boundary (hyperplane). The orientation is determined by ($\Sigma^{-1}(\mu_1 - \mu_0)$); priors shift offset.
- **Different covariances (($\Sigma_0 \neq \Sigma_1$))** → **Quadratic** decision boundary (general quadric surface). Different covariance shapes/orientations make the boundary curve to reflect differing class spreads.

---

## Final remarks / practical notes

- GDA (shared ($\Sigma$)) is elegant and yields linear separators but is only appropriate when class spreads are similar.
- QDA (separate ($\Sigma_k$)) is more flexible (can capture curved boundaries) but requires estimating more parameters (risk of overfitting with small data).
- The Mahalanobis metric ($|x - \mu|_{\Sigma^{-1}}^2 = (x - \mu)^\top \Sigma^{-1}(x - \mu)$) is the natural distance GDA uses — this is why ($\Sigma$) appears as a reweighting in the boundary.

---

## Problem 9

You have trained a logistic regression model to predict the probability of a customer churning (y=1). The model uses two features: `customer_age` (in years) and `monthly_spend` (in dollars). The fitted model equation for the log-odds (logit) is:

$$\log\left(\frac{p}{1-p}\right) = -2.5 - 0.05 \cdot (\text{customer-age}) + 0.08 \cdot (\text{monthly-spend})$$

(a) Holding `monthly_spend` constant, what is the effect on the **odds** of churning for every 1-year increase in `customer_age`? (Provide a specific factor and show your calculation).

(b) Holding `customer_age` constant, by what **factor** do the odds of churning change for every $10 increase in `monthly_spend`?

(c) What is the predicted **probability** (p) of churning for a 40-year-old customer who spends $100 per month?

---

## Quick reminder (why we do this)

Logistic regression models the log-odds linearly:

$$\text{logit}(p) = \log\frac{p}{1-p} = \beta_0 + \beta_1 x_1 + \beta_2 x_2.$$

A change ($\Delta x_j$) in feature ($x_j$) multiplies the **odds** by ($\exp(\beta_j \Delta x_j)$) because

$$\Delta(\log \text{odds}) = \beta_j \Delta x_j \quad \Longrightarrow \quad \text{odds multiplier} = e^{\beta_j \Delta x_j}.$$

---

### (a) Effect of +1 year in age (monthly_spend held constant)

Coefficient for `customer_age` is ($\beta_{\text{age}} = -0.05$).

For a 1-year increase:

$$\text{odds multiplier} = \exp(-0.05 \times 1) = e^{-0.05}.$$

Compute numerically (digit-by-digit):

- ($e^{0.05} \approx 1.051271$) so ($e^{-0.05} = 1/e^{0.05} \approx 1/1.051271 \approx 0.951229.$)

**Answer:** The odds are multiplied by ($\boxed{e^{-0.05} \approx 0.95123}$).

*(Why? a factor <1 means odds decrease — here ~0.951, i.e. about a 4.88% decrease in odds per additional year: ($1 - 0.95123 \approx 0.04877$).)*

---

### (b) Effect of +$10 in monthly_spend (age held constant)

Coefficient for `monthly_spend` is ($\beta_{\text{spend}}$=0.08).

For an increase of \$10:

$$\text{odds multiplier} = \exp(0.08 \times 10) = \exp(0.8).$$

Compute numerically:

- $(e^{0.8} \approx 2.225540)$.

**Answer:** The odds are multiplied by ( $\boxed{e^{0.8} \approx 2.22554}$ ).

*(Why? odds more than double — specifically, about a 122.6% increase in odds for every \$10 increase in monthly spend.)*

---

### (c) Predicted probability for age = 40, spend = 100

First compute the logit (plug values into the model):

$$\text{logit}(p) = -2.5 - 0.05 \cdot 40 + 0.08 \cdot 100.$$

Compute each term:

- (-0.05 $\times$ 40 = -2.)
- (0.08 $\times$ 100 = 8.)

So

$$\text{logit}(p) = -2.5 - 2 + 8 = 3.5.$$

*(Why? evaluate linear expression.)*

Convert logit to probability using the sigmoid:

$$p = \frac{1}{1+e^{-\text{logit}}} = \frac{1}{1+e^{-3.5}}.$$

Compute numerically:

- ($e^{-3.5} \approx 0.03019738$).
- So ($p \approx \dfrac{1}{1+0.03019738} \approx \dfrac{1}{1.03019738} \approx 0.97068777$.)

**Answer:** ( $\boxed{p \approx 0.9707\ (97.07\ \%\ )}$ .)

*(Extra intuition: the odds = ($e^{3.5} \approx 33.11545$), i.e. roughly 33.1 to 1 in favor of churn, which corresponds to probability ($33.11545/(1+33.11545) \approx 0.9707$).)*

---

## One-line summary

- (a) Odds multiplier per +1 year age = ($e^{-0.05} \approx 0.95123$) (≈4.9% decrease).
- (b) Odds multiplier per +\$10 spend = ($e^{0.8} \approx 2.22554$) (≈122.6% increase).

- (c) For age 40 and spend \$100: ($p \approx 0.9707$) (≈97.1% chance of churning).

---