Part A — Calculation

Q1. Decision Stump Prediction

Stump: predict + if `Sneezing=Yes`, – otherwise.
Dataset (4 examples):

1. (Yes, +) → predicts + → **correct**
2. (No, -) → predicts - → **correct**
3. (Yes, -) → predicts + → **incorrect**
4. (No, -) → predicts - → **correct**
5. Training error = 1 misclassification out of 4 → **error = 1/4 = 0.25 (25%)**.
6. Memorizer (perfect) would have **error 0**. The stump is worse (25% error).

Q2. Training Error as Splitting Criterion

Data (6 records):

Row	Age (x1)	Exercise (x2)	Diet (x3)	Label
1	Young	High	Poor	Yes
2	Young	Medium	Good	Yes
3	Mid	Low	Poor	No
4	Old	Medium	Poor	No
5	Old	High	Good	Yes
6	Mid	Low	Poor	No

Compute training error for splitting on each feature (choose majority label in each child node; ties result in at least one error in that node).

- Split on **Age (x1)**:
  - Young: rows 1,2 → (Yes, Yes) → predict Yes → errors 0
  - Mid: rows 3,6 → (No, No) → predict No → errors 0
  - Old: rows 4,5 → (No, Yes) → best single-class prediction causes 1 error
  - Total errors = 1 → training error = **1/6 ≈ 0.1667 (16.67%)**
- Split on **Exercise (x2)**:
  - High: rows 1,5 → (Yes, Yes) → errors 0
  - Medium: rows 2,4 → (Yes, No) → 1 error
  - Low: rows 3,6 → (No, No) → errors 0
  - Total errors = 1 → **1/6 ≈ 16.67%**
- Split on **Diet (x3)**:

- Poor: rows 1,3,4,6 → labels (Yes, No, No, No) → majority No → 1 error (row1)
- Good: rows 2,5 → (Yes, Yes) → errors 0
- Total errors = 1 → **1/6 ≈ 16.67%**

2. All three features yield the same training error **1/6**. So all are equally good by this criterion.

Q3. Entropy & Information Gain (same dataset)

Label counts: 3 Yes, 3 No → p(Yes)=0.5, p(No)=0.5.

1. Entropy of labels:
   $H(Y) = -0.5\log_2 0.5 - 0.5\log_2 0.5 = 1.0\text{bit}.$
2. Entropy after splitting on **Exercise (x2)**:
   - High (2 samples: Yes, Yes) → entropy 0
   - Medium (2 samples: Yes, No) → entropy = 1
   - Low (2 samples: No, No) → entropy 0
     Weighted entropy = (2/6)*0 + (2/6)*1 + (2/6)*0 = 2/6 = **1/3 ≈ 0.3333 bits**
3. Information gain:
   $IG = H(Y) - H(Y \mid x2) = 1 - 1/3 = 2/3 \approx 0.6667\text{bits}.$
4. Is Exercise a good split? Yes — it reduces entropy substantially (gain ≈ 0.667 bits) and perfectly separates two of the three child nodes.

Q4. Confusion Matrix Metrics

Confusion matrix on 100 samples:

	Pred +	**Pred -**
Actual +	25	5
Actual -	15	55

Compute metrics:

- Accuracy = (TP + TN)/Total = (25 + 55)/100 = **0.80 (80%)**
- Precision = TP / (TP + FP) = 25 / (25 + 15) = 25/40 = **0.625**
- Recall (Sensitivity) = TP / (TP + FN) = 25 / (25 + 5) = 25/30 ≈ **0.8333**
- Specificity = TN / (TN + FP) = 55 / (55 + 15) = 55/70 ≈ **0.7857**
- F1-score = 2 * (precision * recall) / (precision + recall) ≈ **0.7143**

2.If dataset was imbalanced (e.g., 80 negatives, 20 positives), **accuracy can be misleading**. The most informative metrics are **precision, recall, and F1-score** (F1 is a single-number tradeoff). For class-imbalanced problems, also consider Precision-Recall and per-class metrics.

Q5. Distance Calculations (kNN)

Points:

- A = (2,4), label Red
- B = (4,4), label Blue
- C = (4,6), label Red
- P = (5,4) — classify

1. Euclidean distances (compute digit-by-digit):
   - d(P,A) = sqrt((5−2)^2 + (4−4)^2) = sqrt(3^2 + 0) = sqrt(9) = **3.0**
   - d(P,B) = sqrt((5−4)^2 + (4−4)^2) = sqrt(1 + 0) = **1.0**
   - d(P,C) = sqrt((5−4)^2 + (4−6)^2) = sqrt(1 + 4) = sqrt(5) ≈ **2.2361**
2. 1-NN: nearest is B (Blue) → **predict Blue**.
3. 3-NN: three nearest are B (Blue), C (Red), A (Red) → votes: Red=2, Blue=1 → **predict Red**.

Q6. K-fold Cross-Validation

Fold errors:

Fold	k=1	k=3	k=5
1	0.20	0.15	0.10
2	0.25	0.20	0.15
3	0.15	0.10	0.10
4	0.30	0.20	0.20

1. Mean CV error:

- k=1: mean = (0.20 + 0.25 + 0.15 + 0.30)/4 = 0.90/4 = **0.225**
- k=3: mean = (0.15 + 0.20 + 0.10 + 0.20)/4 = 0.65/4 = **0.1625**
- k=5: mean = (0.10 + 0.15 + 0.10 + 0.20)/4 = 0.55/4 = **0.1375**

2. Best generalization (lowest mean CV error) is **k=5** (error 0.1375).