دانشگاه تهران
پردیس دانشکده‌های فنی
دانشکده برق و کامپیوتر

یادگیری ماشین

تمرین شماره 4

نام و نام خانوادگی
سید محمد جزایری

شماره دانشجویی
810101399

دی 1404

فهرست

فهرست شکل‌ها

فهرست جدول‌ها

چکیده

In this homework we implement different feature selection algorithms and compare them on a real dataset. We do the same for clustering algorithms. Furthermore we explore PCA and LDA algorithms and apply them to the Iris dataset and visualize it as well.

پرسش 1 - عنوان پرسش

1-1. عنوان بخش

For this part I am going to compare the square of the Euclidean distance since: $A^2 < B^2 \rightarrow A < B$

P_1:

Distance to cluster 0: $(5.5 - 5.3)^2 + (3.1 - 3.5)^2 = 0.2$

Distance to cluster 1: $(5.5 - 5.1)^2 + (3.1 - 4.2)^2 = 1.37$

Conclusion: assign P_1 to cluster 0.

P_2:

Distance to cluster 0: $(5.1 - 5.3)^2 + (4.8 - 3.5)^2 = 1.73$

Distance to cluster 1: $(5.1 - 5.1)^2 + (4.8 - 4.2)^2 = 0.36$

Conclusion: assign P_2 to cluster 1.

P_3:

Distance to cluster 0: $(6.6 - 5.3)^2 + (3.0 - 3.5)^2 = 1.94$

Distance to cluster 1: $(6.6 - 5.1)^2 + (3.0 - 4.2)^2 = 3.69$

Conclusion: assign P_3 to cluster 0.

P_4:

Distance to cluster 0: $(5.5 - 5.3)^2 + (4.6 - 3.5)^2 = 1.25$

Distance to cluster 1: $(5.5 - 5.1)^2 + (4.6 - 4.2)^2 = 0.32$

Conclusion: assign P_4 to cluster 1.

P_5:

Distance to cluster 0: $(6.8 - 5.3)^2 + (3.8 - 3.5)^2 = 2.34$

Distance to cluster 1: $(6.8 - 5.1)^2 + (3.8 - 4.2)^2 = 3.05$

Conclusion: assign P_5 to cluster 0.

2-1. عنوان بخش

Cluster 0:

$X_{new} = 6.3 \qquad Y_{new} = 3.3$

Cluster 1:

$X_{new} = 5.3 \qquad Y_{new} = 4.7$

3-1. عنوان بخش

$\mu_0 = (6.3,\ 3.3)$

$\mu_1 = (5.3,\ 4.7)$

4-1. عنوان بخش

Three points belong to cluster 0, therefore its dimensionality is 3.

پرسش 2 - عنوان پرسش

1-2. عنوان بخش

Minimum distance algorithms take $O(n^2)$ or $O(n \log n)$ in efficient cases where n is number of data samples, whereas maximum distance algorithms take $O(n^2 \log n)$ or $O(n^3)$ in naive cases. In conclusion, minimum distance is more efficient.

2-2. عنوان بخش

Single Linkage (The Chaining Problem): This method merges clusters if any two points are close. A chain of noisy points can bridge two distinct clusters together like a bridge. It is sensitive to outliers.

Complete Linkage (Compact Clusters): This method only merges clusters if the farthest points are close enough. This forces clusters to stay compact and round. An outlier that is far away will not easily be merged because it would drastically increase the cluster diameter.

Conclusion: Complete Linkage is more robust to outliers. It avoids the chaining effect where a single noise point merges two unrelated clusters.

3-2. عنوان بخش

First Merges (Closest Points):

- P_4 and P_7: Distance ≈1.41. Merge them. (Cluster A)
- P_5 and P_8: Distance ≈1.41. Merge them. (Cluster B)

Second Round:

- P_1 and P_2: Distance ≈ 2.24. Merge them. (Cluster C)
- Check P_3 against Cluster B (P_5 and P_8):
    - Dist to P_5 ≈ 2.24.

- Dist to $P_8 \approx 2.24$.
- Max is 2.24. Merge P_3 into Cluster B. (New Cluster B: P_3, P_5, P_8).

Third Round:

- Check P_9 against Cluster C (P_1, P_2):
  - Dist to $P_1 = 3.0$.
  - Dist to $P_2 \approx 2.82$.
  - Max is 3.0. Merge P_9 into Cluster C. (New Cluster C: P_1, P_2, P_9).

Fourth Round:

- We have P_6 left alone.
- Dist P_6 to Cluster A (P_4, P_7): Max dist is to P_4(d=6.0).
- Dist P_6 to Cluster B (P_3, P_5, P_8): Max dist is to P_3 (d=6.0).
- It's a tie. Geometrically, P_6 is on the right side with Cluster B. Let's Merge P_6 into Cluster B. (New Cluster Right: P_3, P_5, P_6, P_8).

Fifth Round:

- Now we have the Left Cluster (P_1, P_2, P_9) and Top-Left Cluster (P_4, P_7).
- Max distance between them is P_1 to P_7:
  $\sqrt{(4-1)^2 + (8-1)^2} = \sqrt{9+49} \approx 7.6$.
- Merge these two. (New Cluster Left-Top).

Final Merge:

- Merge the big Left-Top group with the big Right group.
- The max distance is between the farthest points: $P_1\,(1,1)$ and $P_6\,(9,7)$.
- Distance $= \sqrt{8^2 + 6^2} = 10.0$.

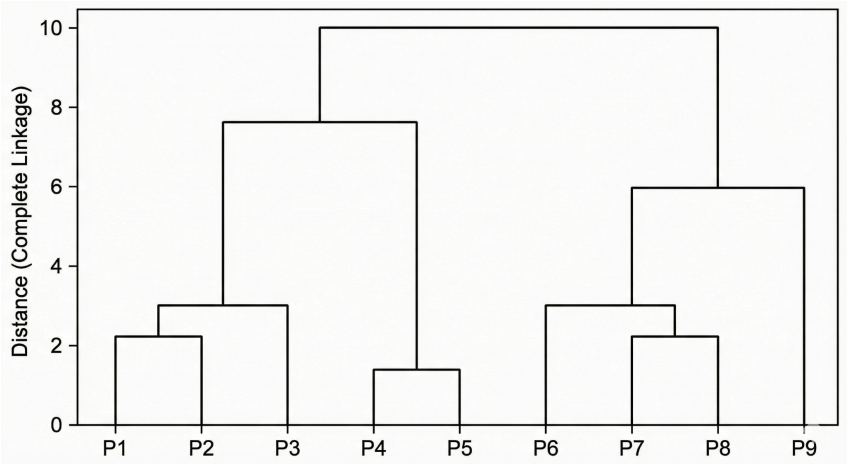

Fig 1. The resulting dendrogram

پرسش 3 - عنوان پرسش

Scenario 1: Linearly Correlated Gaussian Data

- Best Method: PCA (Principal Component Analysis)
- Why?
    - Geometry: Gaussian data forms a simple cloud or ellipsoid shape. The most interesting features are simply the directions where the cloud is widest (most variance).
    - Assumption: PCA assumes that the data lies on a linear subspace and that variance equals information. Since the data is linearly correlated, a simple rotation of axes (which PCA does) captures the structure perfectly.

Scenario 2: Two Concentric Circles

- Best Method: Kernel PCA
- Why?
    - Geometry: One circle is inside the other. No straight line can separate them, and projecting them onto a straight line (Standard PCA) would just smash them together, losing the structure.
    - The Trick: Kernel PCA projects the data into a higher-dimensional space (like a 3D hill). In that space, the inner circle might become the "peak" and the outer circle the base, making them easy to distinguish. It captures non-linear structure.

Scenario 3: High-Dimensional Data with Strong Class Overlap

- Best Method: LDA (Linear Discriminant Analysis)
- Why?
    - Objective Function: PCA focuses on *total variance*, ignoring class labels. If the noise is loud (high variance), PCA will preserve the noise and ignore the classes.
    - LDA's Advantage: LDA specifically looks at the class labels. Its goal is to maximize the distance *between* class means ($\mu_1 - \mu_2$) while minimizing

the spread *within* each class. Even if the overlap is strong, LDA is the only method here that actively tries to find the specific angle that separates the classes best.

Scenario 4: Data Containing Many Irrelevant or Noisy Features

- Best Method: PCA
- Why?
    - Assumption: In most datasets, the signal (useful information) creates correlations and high variance, while noise is random and has lower variance.
    - How it works: PCA packs the correlated signal into the top few Principal Components (large eigenvalues). The random noise gets pushed into the bottom components (small eigenvalues). By dropping the bottom components, we effectively denoise the data.

پرسش 4 - عنوان پرسش

We look at the Between-Class Scatter Matrix (S_B), which measures how far the class centers are from the global center.

$$S_B = \sum_{c=1}^{C} N_c(\mu_c - \mu)(\mu_c - \mu)^T$$

μ_c is the mean of class c.

μ is the global mean of all data.

N_c is the number of samples in class c.

The matrix S_B is constructed from C specific vectors:

$$v_c = \mu_c - \mu$$

This means the "rank" (dimensionality) of S_B depends on how many of these v_c vectors are linearly independent.

The global mean μ is just the weighted average of the class means:

$$\mu = \sum_{c=1}^{C} \frac{N_c}{N}\mu_c$$

If we move terms around, we see that the sum of weighted deviations is zero:

$$\sum_{c=1}^{C} N_c (\mu_c - \mu) = 0$$

Because these vectors must sum to zero, they are not fully independent. If you know the first C-1 vectors, the last one is fixed (it has to balance the equation to zero). Therefore, the maximum number of independent vectors is C-1.The rank of S_B is $\leq C - 1$ Consequently, there are at most C-1 nonzero eigenvalues (discriminant directions).

In Kernel PCA, we map our data x into a feature space $\Phi(x)$.

- The Problem: This feature space is often infinite-dimensional (e.g., when using the RBF Kernel).
- The Covariance Matrix: To do PCA, we need the covariance matrix C. If the space is infinite-dimensional, C would be an $\infty \times \infty$ matrix. We cannot store it or compute its eigenvectors directly.

We want to find an eigenvector V (a direction in feature space) such that:

$$C V = \lambda V$$

where the covariance matrix is the sum of the outer products of the data:

$$C = \frac{1}{N} \sum_{i=1}^{N} \Phi(x_i) \Phi(x_i)^T$$

Substitute C into the eigenvector equation:

$$\left(\frac{1}{N} \sum_{i=1}^{N} \Phi(x_i) \Phi(x_i)^T \right) V = \lambda V$$

Rearrange the terms:

$$\frac{1}{N} \sum_{i=1}^{N} \Phi(x_i) (\Phi(x_i)^T V) = \lambda V$$

Notice that $\Phi(x_i)^T V$ is just a scalar number (a dot product).

This equation tells us that V (times λ) is equal to a weighted sum of the data points $\Phi(x_i)$.

Thus, V lies in the span of the mapped data points:

$$V = \sum_{i=1}^{N} \alpha_i \Phi(x_i)$$

This is crucial. It means we don't need to look at the whole infinite space; we only need to find the weights α_i for our N data points.

Now, substitute $V = \sum_{j=1}^{N} \alpha_j \Phi(x_j)$ back into the eigen-equation:

$$C V = \lambda V$$

$$\frac{1}{N} \sum_{i=1}^{N} \Phi(x_i) \Phi(x_i)^T \left(\sum_{j=1}^{N} \alpha_j \Phi(x_j) \right) = \lambda \sum_{j=1}^{N} \alpha_j \Phi(x_j)$$

To solve this, we multiply both sides by $\Phi(x_k)^T$(project onto a data point x_k):

$$\frac{1}{N} \sum_{i=1}^{N} \Phi(x_k)^T \Phi(x_i) \left(\sum_{j=1}^{N} \alpha_j \Phi(x_i)^T \Phi(x_j) \right) = \lambda \sum_{j=1}^{N} \alpha_j \Phi(x_k)^T \Phi(x_j)$$

Recognize that $\Phi(a)^T \Phi(b)$is the Kernel function K(a, b).

Using matrix notation (K is the $N \times N$ kernel matrix):

$$\frac{1}{N} K^2 \alpha = \lambda K \alpha$$

Multiply by K^{-1} (assuming invertibility for simplicity) or simply match terms:

$$K \alpha = \lambda N \alpha$$

Once we solve $K\alpha = \lambda' \alpha$ to get the weights α, we can project a new test point x_{new} onto this principal component V:

$$Projection = V^T \Phi(x_{new})$$

Since $V = \sum_i \alpha_i \Phi(x_i)$:

$$Projection = \sum_{i=1}^{N} \alpha_i \Phi(x_i)^T \Phi(x_{new})$$

$$Projection = \sum_{i=1}^{N} \alpha_i K(x_i, x_{new})$$

This allows us to compute the projection using only kernel evaluations, never entering the infinite-dimensional space explicitly.

پرسش 5 - عنوان پرسش

Class 1:

$$\mu_{1x} = 3$$

$$\$\mu_{1y} = 3.8$$

$$\mu_1 = (3, 3.8)$$

Class 2:

$$\mu_{2x} = 8.4$$

$$\$\mu_{2y} = 7.6$$

$$\mu_2 = (8.4, 7.6)$$

Since $N_1 = N_2$, the global mean is just the average of the two class means.

$$\mu = (5.7, 5.7)$$

Within-Class Scatter Matrix $S_W = S_1 + S_2$, where $S_i = \sum_i (x - \mu_i)(x - \mu_i)^T$.

$S_1 =$

4	-1
-1	8.8

$S_2 =$

9.2	-0.2
-0.2	13.2

$S_W = S_1 + S_2 =$

13.2	-1.2
-1.2	22.0

Between-Class Scatter Matrix (S_B)

For two classes, S_B is proportional to the outer product of the difference in means $(\mu_1 - \mu_2)(\mu_1 - \mu_2)^T$.

Mean Difference: $\mu_1 - \mu_2 = (3 - 8.4, 3.8 - 7.6) = (-5.4, -3.8)$

For a binary classification problem, we do not need to perform full eigen-decomposition. The optimal vector w is given directly by:

$$w \propto S_W^{-1}(\mu_1 - \mu_2)$$

$S_W^{-1} = \frac{1}{288.96}$

$$\begin{vmatrix} 22 & 1.2 \\ 1.2 & 13.2 \end{vmatrix}$$

We can ignore the scalar $\frac{1}{288.96}$ as it only scales the magnitude, not the direction.

$S_W^{-1}(\mu_1 - \mu_2) = [-123.36 \quad -56.64]^T$

This vector points in the direction of Class 1. We can simplify this vector by dividing by -56.64 to get a cleaner ratio:

$$w \approx (2.18, 1)$$

Class 1:

(4,2): 2.18(4) + 2 = 10.72

(2,4): 2.18(2) + 4 = 8.36

(2,3): 2.18(2) + 3 = 7.36

(3,6): 2.18(3) + 6 = 12.54

(4,4): 2.18(4) + 4 = 12.72

Class 2:

(9,10): 2.18(9) + 10 = 29.62

(6,8): 2.18(6) + 8 = 21.08

(9,5): 2.18(9) + 5 = 24.62

(8,7): 2.18(8) + 7 = 24.44

(10,8): 2.18(10) + 8 = 29.80

Result: Class 1 clusters around 7-12, and Class 2 clusters around 21-30. The classes are perfectly separated.

پرسش 6 - عنوان پرسش

6-1. عنوان بخش

1. Sequential Forward Selection (SFS)

Initialization: Start with an empty set (no features).

Update Step:

1. Train the model on every single feature individually (Feature A only, Feature B only, etc.).
2. Pick the winner (say, Feature A).
3. Now, try adding every remaining feature to the winner (A+B, A+C, etc.).
4. Pick the winner (say, A+C).
5. Repeat, adding one feature at a time.

Stopping Criterion: Stop when adding a new feature doesn't improve the model's accuracy, or when you reach a specific number of features (e.g., "I only want 5 features").

2. Sequential Backward Selection (SBS)

Initialization: Start with all features (the full set).

Update Step:

1. Train the model with everything.
2. Try removing one feature at a time and see which removal hurts the model the least (or maybe even improves it by removing noise).
3. Permanently delete that feature.
4. Repeat.

Stopping Criterion: Stop when removing any more features causes the accuracy to drop significantly.

3. Bidirectional (Stepwise) Selection

This combines both. In each round, you try to add the best feature, but you also check if you should delete any of the features you added earlier (maybe Feature A was good at the start, but now that we have Feature C and D, Feature A is useless).

They are called greedy because they make the best immediate decision at every single step without looking at the big picture.

- SFS says: "Which feature is best *right now*?"
- It does **not** say: "Maybe I should pick a weak feature now because it combines perfectly with another feature later."
- Once a decision is made (adding or removing), it is never reconsidered (unless using bidirectional).

2-6. عنوان بخش

Because these methods are greedy, they are not guaranteed to find the absolute best combination of features. To find the perfect set, you would need to check every possible combination (2^N), which is impossible for large datasets.

Concrete Example of Failure (The "XOR" Problem):

Imagine you are predicting if a customer will buy a product.

- Feature 1: "Is rich?" (Randomly predicts 50% correct)
- Feature 2: "Is frugal?" (Randomly predicts 50% correct)
- Feature 3: "Lives in City" (Predicts 60% correct by itself)

The Truth: Let's say rich people who are *also* frugal always buy. (Interaction between 1 and 2 is 100% accurate).

How SFS Fails:

1. Round 1: SFS tests all three.
    - Feat 1 alone: 50% acc.
    - Feat 2 alone: 50% acc.
    - Feat 3 alone: 60% acc.
2. Decision: SFS picks **Feature 3** because it looks best individually.
3. Result: It might never pick Features 1 and 2 because individually they looked like garbage. It missed the "interaction" because it was being greedy for immediate results.

3-6. عنوان بخش

Table 1. Comparison of different methods

Feature	Sequential (SFS/SBS)	Recursive Feature Elimination (RFE)	L1 Regularization (Lasso)
How it works	Adds/Removes features by re-training the model over and over.	Trains once (or few times), looks at weights, and drops the smallest weights.	Solves a single math equation that force some weights to become exactly zero.
Speed (Complexity)	Very Slow. Training a model 1000 times takes forever.	Faster. Only trains a few times.	Fastest. It's just one training run.
High Dimensions	Terrible. Too slow for thousands of features.	Good. Can handle many features.	Excellent. Designed for high-dimensional data.
Correlations	Can handle them, but slow.	If features are correlated, RFE randomly drops one and keeps the other.	Lasso randomly picks one correlated feature and sets the other to zero (instability).
Feature Interactions	Poor (misses XOR problems as explained above).	Better, as it starts with all features and sees the	Okay, but primarily focuses on linear individual contributions.

		interactions before deleting.	

پرسش 7 - عنوان پرسش

7-1. عنوان بخش

The Iris dataset has 4 features (Sepal Length, Sepal Width, Petal Length, Petal Width).

```
--- Part A: PCA on Iris Dataset ---
Eigenvalues (variance explained by each component): [2.93808505 0.9201649 ]
Explained Variance Ratio: [0.72962445 0.22850762]
Total Variance Explained: 95.81%
```

Fig 2. Result of PCA

As we can see the two components capture almost 96% of the variance which is significant! Why*?* PC1 aligns with the "overall size" of the flower. A big flower tends to have both big petals and big sepals. Since most features correlate (grow together), this single direction explains most differences between flowers.

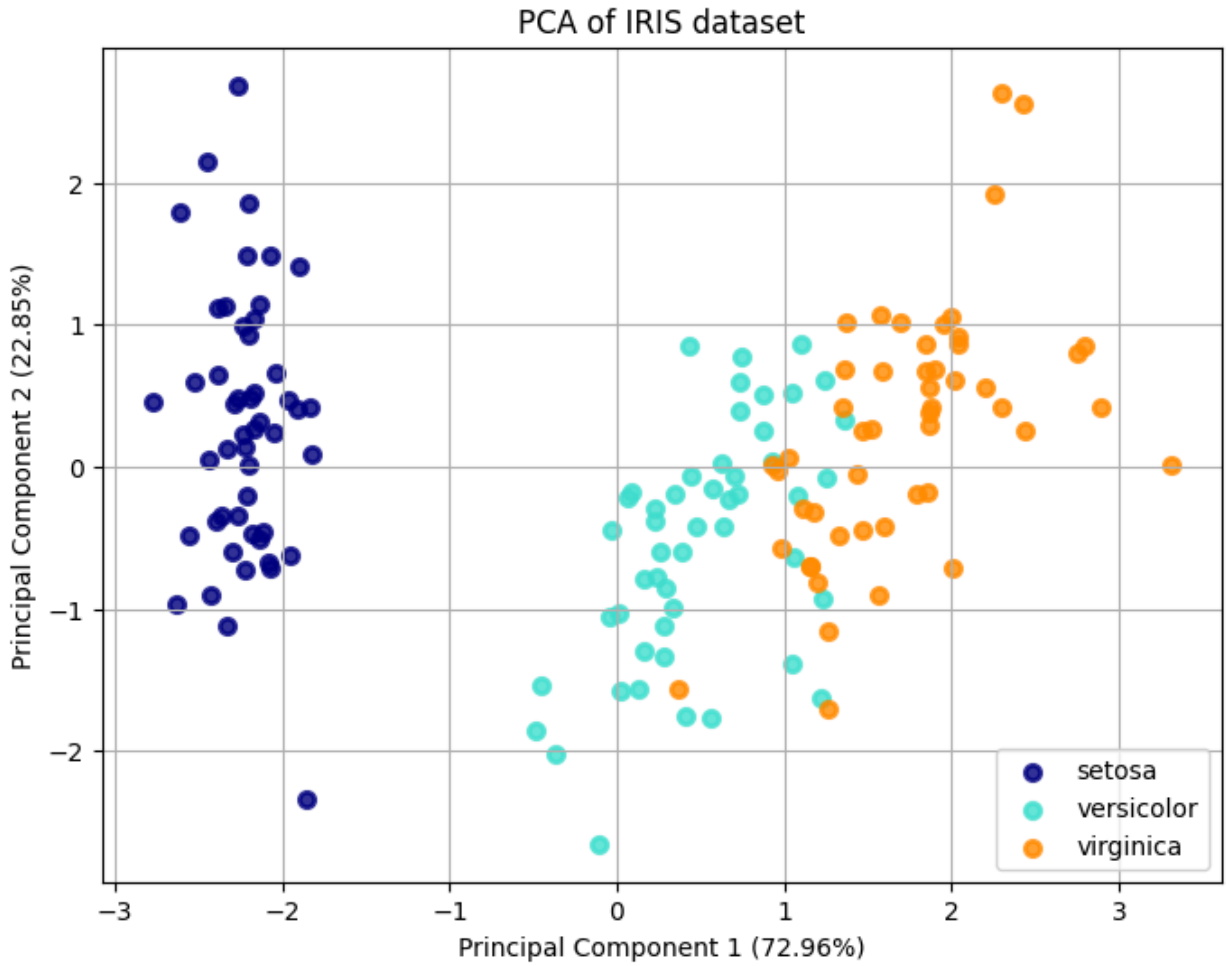

Fig 3. Plot of PCA

One class (*Setosa*) is perfectly separated from the others. The other two classes (*Versicolor* and *Virginica*) overlap slightly.

PCA tries to preserve the "spread" of the data, not the "groups." Since the different species have different sizes, preserving the size information (variance) coincidentally helps separate them, but it's not guaranteed.

PCA is an **unsupervised** technique. It does not know that classes exist. It only cares about variance. If the noise in your data has higher variance than the difference between classes, PCA will preserve the noise and discard the useful class info.

PCA Projection: It spreads the data out as much as possible like a pancake. It ignores color (labels).

LDA Projection: It specifically tries to group points of the same color together and push different colors apart.

Result: If we used LDA, the overlap between *Versicolor* and *Virginica* would likely disappear or be minimized, because LDA explicitly calculates the projection that maximizes that separation.

2-7. عنوان بخش

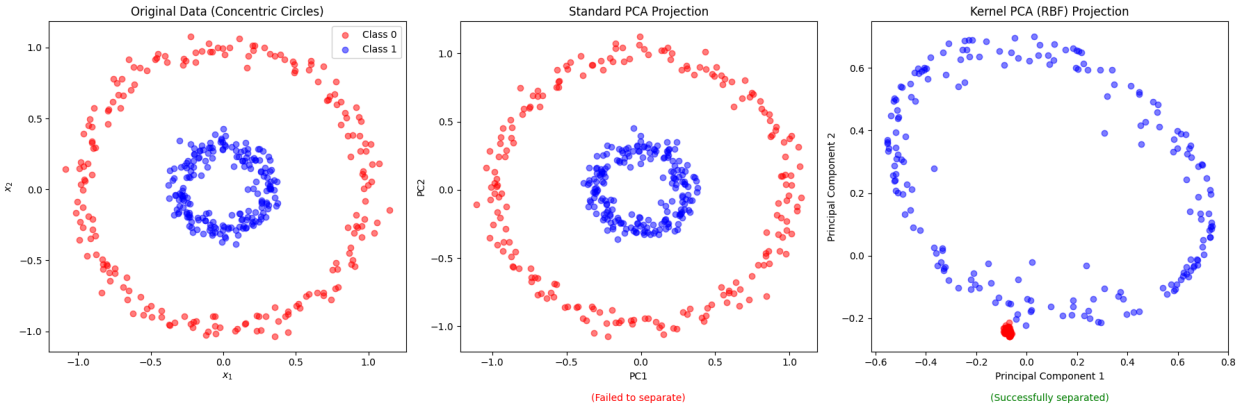

Fig 4. Plot of PCA on two circular datasets

As it appears the standard PCA fails to separate the two datasets because it can only rotate and squash; it can't bend or cut; besides it can only separate classes via lines and can’t find non-linear discriminators that’s why the PCA output looks exactly like the input. Whereas kernel PCASucceeds because it implicitly maps data to a higher dimension where the concentric circles become linearly separable (like a cone), allowing the principal components to capture the non-linear structure.

پرسش 8 - عنوان پرسش

8-1. عنوان بخش

Correlation only checks for linear relationships between two variables.

- **The Flaw:** It ignores the target variable (y). Two features might be highly correlated with each other (redundant), but dropping one might lose subtle information if the noise in them is different.
- **The Trap:** It cannot detect interactions. Feature A and Feature B might look useless individually (low correlation with y), but together they might perfectly predict y (like an XOR function).

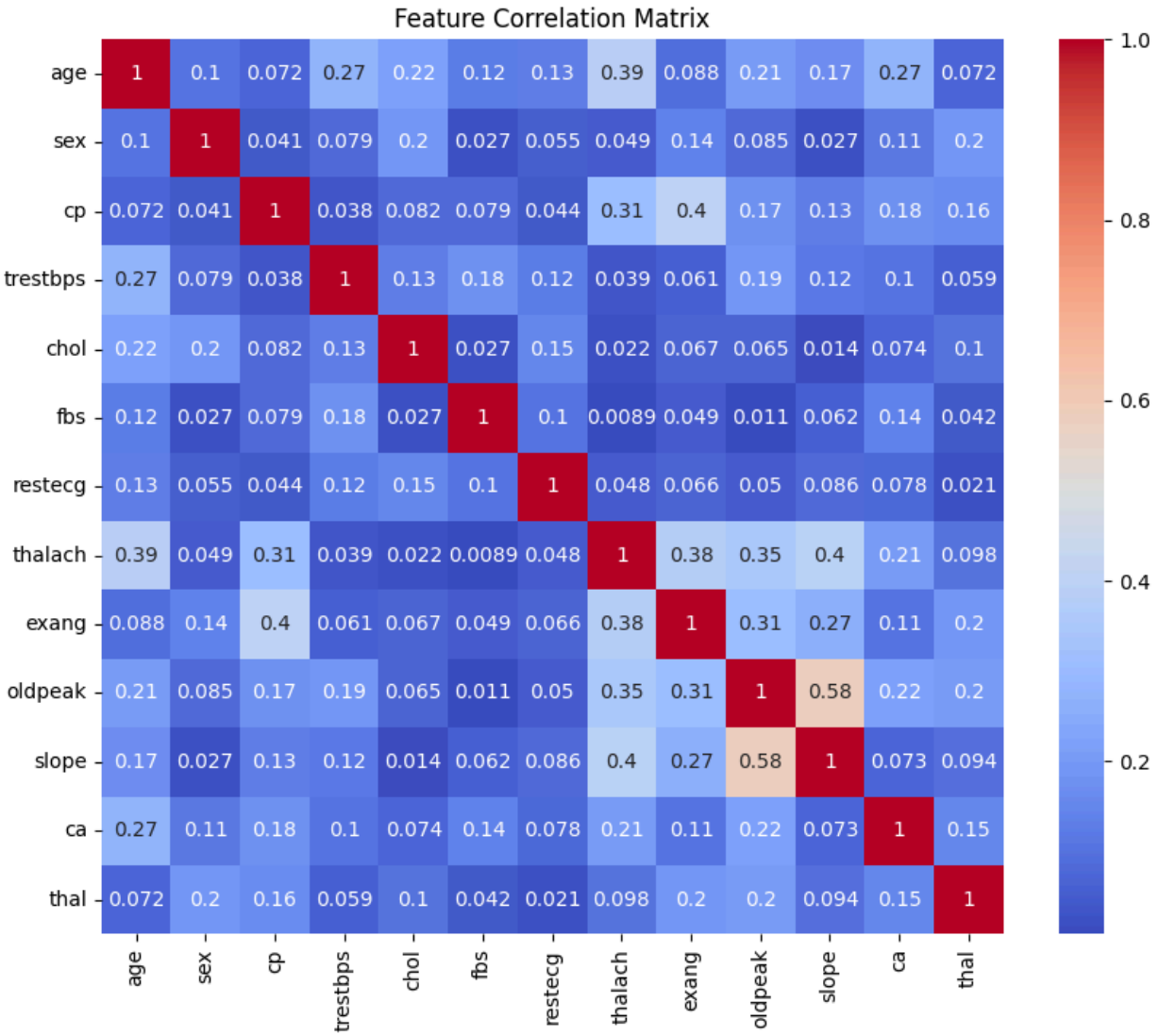

Fig 5. Confusion matrix

Most of the grid is blue, indicating low correlation (values near 0). This suggests that the features in the Heart Disease dataset are largely independent of each other. This is good for many models (like Naive Bayes or Logistic Regression) that assume feature independence.

There is a distinct lighter square where oldpeak and slope intersect. This indicates a moderate positive correlation (0.58). This makes medical sense. oldpeak measures ST depression induced by exercise, and slope measures the slope of the peak exercise ST segment. Both are derived from the same ECG readings during a stress test.

Conclusion**:** Since no off-diagonal squares are dark red (correlation > 0.85), no features will be removed by a standard correlation filter. The features measure distinct physiological factors (cholesterol vs. age vs. heart rate), so they don't overlap much.

2-8. عنوان بخش

In this part I will use a logistic regression model for the classification task and information gain as the feature selection method.

```
--- 2. Univariate Selection (Mutual Information) ---
Top 5 Features (Univariate): ['cp', 'chol', 'thalach', 'oldpeak', 'thal']
Accuracy with Top-5 Univariate features: 0.8156
Accuracy with all features: 0.8585
```

Fig 6. Result of information gain

As we can see the top 5 features don't perform a lot worse than the whole dataset.

3-8. عنوان بخش

```
--- 3. Recursive Feature Elimination (RFE) ---
Top 5 Features (RFE): ['sex', 'cp', 'thalach', 'oldpeak', 'ca']
Accuracy with RFE features: 0.8059

--- 4. RFECV (Automatic Number of Features) ---
Optimal number of features: 12
Selected Features (RFECV): ['age', 'sex', 'cp', 'trestbps', 'chol', 'restecg', 'thalach', 'exang', 'oldpeak', 'slope', 'ca', 'thal']
Accuracy with RFECV features: 0.8585
Best Cross-Validation Score: 0.8478
```

Fig 7. Result of RFE and RFECV

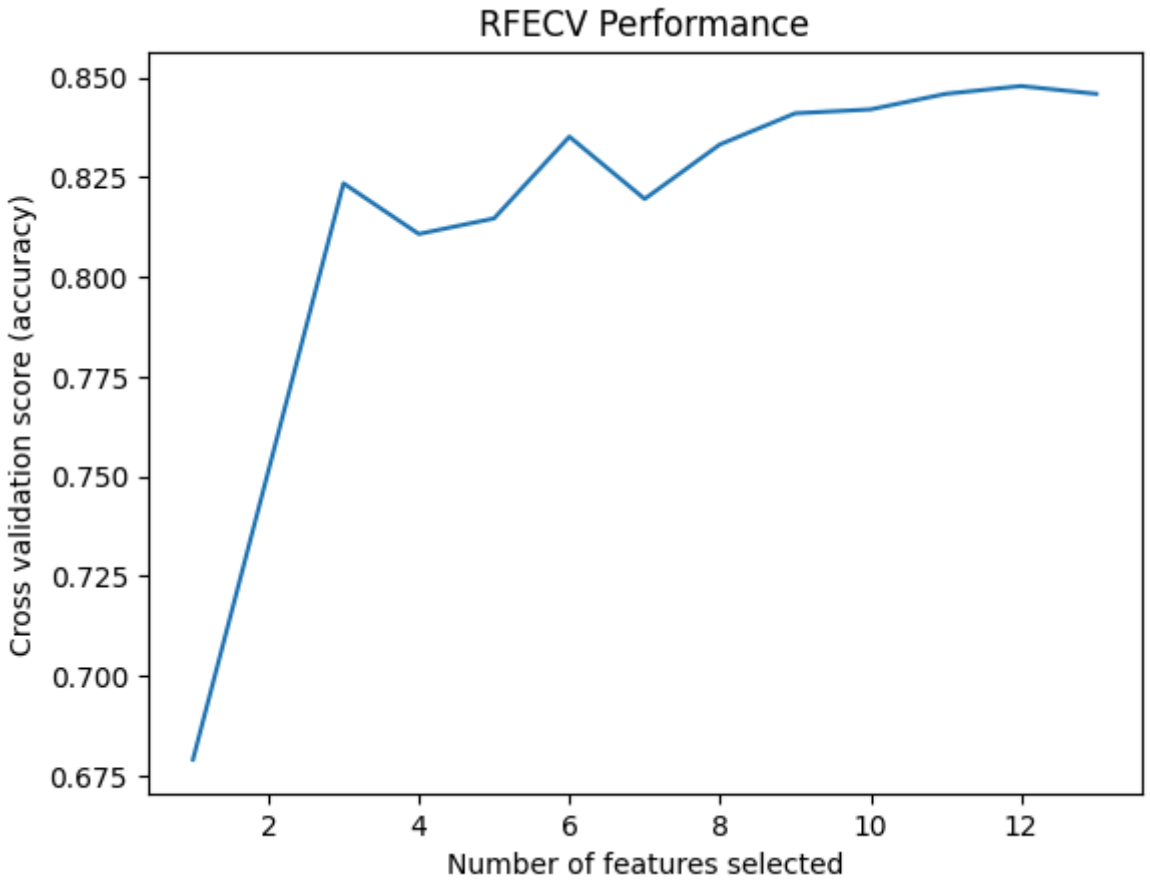

Fig 8. Cross-validation accuracy of RFECV

RFE doesn't achieve the best accuracy possible and it even under-performs the Univariate Selection method but the RFECV method uses 12 features but achieves the accuracy we'd achieve using 13 (all) features.

RFE requires you to guess the number of features (k). If you guess wrong, you might toss out good features or keep noise.

RFECV is robust. It tests the model performance at every step. If RFECV selects all features, it implies your dataset is "hard" (as is our case)—every single bit of data is needed to distinguish the classes, or the sample size is too small to statistically rule out noise.

4-8. عنوان بخش

```
--- 5. Embedded Method (Lasso L1) ---
Features selected by Lasso (non-zero weights):
['age', 'sex', 'cp', 'trestbps', 'chol', 'restecg', 'thalach', 'exang', 'oldpeak', 'slope', 'ca', 'thal']
Accuracy with Lasso L1: 0.8449
Lasso Coefficients:
 age        -0.059491
sex         -0.651373
cp           0.720967
trestbps    -0.214001
chol        -0.165820
restecg      0.138685
thalach      0.430719
exang       -0.402547
oldpeak     -0.565493
slope        0.256392
ca          -0.632488
thal        -0.447367
dtype: float64
```

Fig 9. Lasso L1 result

Observation on Sparsity: the L1 regularization (Lasso) did not set any coefficients to exactly zero. All 12 features were retained.

- **Implication:** This suggests that with the regularization strength used, the model found that every single feature contributed at least some unique information to the prediction. This aligns with our RFECV result, which also concluded that the optimal number of features is 12 (the maximum).

Ranking by Importance (Absolute Coefficient Magnitude): Since Lasso provides weights, we can rank the features by how much they influence the decision boundary (magnitude of coefficient):

1. **cp (Chest Pain):** 0.72 (Strongest positive predictor)
2. **sex:** 0.65 (Strongest negative predictor)
3. **ca (Major Vessels):** 0.63
4. **oldpeak (ST Depression):** 0.56
5. **thal (Thalassemia):** 0.45
6. **thalach (Max Heart Rate):** 0.43
7. **exang (Exercise Angina):** 0.40 ...
8. **age:** 0.06 (Weakest predictor)

All three methods agree that cp (Chest Pain) and oldpeak (ST depression) are critical. These are likely the strongest, most distinct signals in the dataset.

Univariate selected chol in the top 5.

Lasso ranked chol very low (Coefficient ~0.16, Rank 10).

RFE also dropped chol from the top 5.

Reason: Univariate selection looks at features in isolation. Cholesterol might correlate with heart disease individually, but once you know a patient's Age and Chest Pain, cholesterol adds very little new information. Lasso and RFE (multivariate methods) realize this redundancy and penalize it, whereas Univariate blindly picks it.

Univariate missed sex and ca in its top 5.

Lasso and RFE both identified them as top-tier features (Rank 2 and 3 for Lasso).

Reason: This is the strength of model-based selection (Wrapper/Embedded). Sex might not look like a strong predictor alone (low correlation), but combined with thalach or cp, it becomes a powerful discriminator. Univariate methods fail to capture these conditional dependencies.

5-8. عنوان بخش

I applied Principal Component Analysis (PCA) as an exploratory tool to determine the intrinsic dimensionality of the Heart Disease dataset. The cumulative explained variance plot reveals a gradual increase, requiring 12 principal components to explain 95% of the total variance.

1. Complexity of the Dataset:

The fact that we need 12 components (out of 13 total features) indicates that the dataset contains very little redundancy. The variance is distributed across many orthogonal directions, meaning the data cannot be easily compressed into a lower-dimensional space without significant information loss.

2. Alignment with RFE and RFECV:

This finding strongly corroborates our results from Recursive Feature Elimination with Cross-Validation (RFECV), which also identified 12 features as the optimal subset. Both the unsupervised method (PCA) and the supervised method (RFECV) agree that nearly all features are necessary to capture the underlying structure of the data.

3. Interpretability Trade-off:

While PCA and Feature Selection yielded a similar count of necessary dimensions (12), Feature Selection is superior for this specific medical application.

- **Feature Selection:** Preserves the original, clinically meaningful variables (e.g., *Chest Pain Type*, *ST Depression*), allowing physicians to understand *risk factors*.
- **PCA:** Transforms these into abstract linear combinations (e.g., $0.3 \times Age - 0.5 \times Cholesterol$), which have no direct medical interpretation. Given that PCA achieved almost no dimensionality reduction (13 →12), it offers no computational advantage to outweigh this loss of interpretability.

```
--- 6. PCA Analysis ---
Number of PCA components to explain 95% variance: 12
Number of original features: 13
```

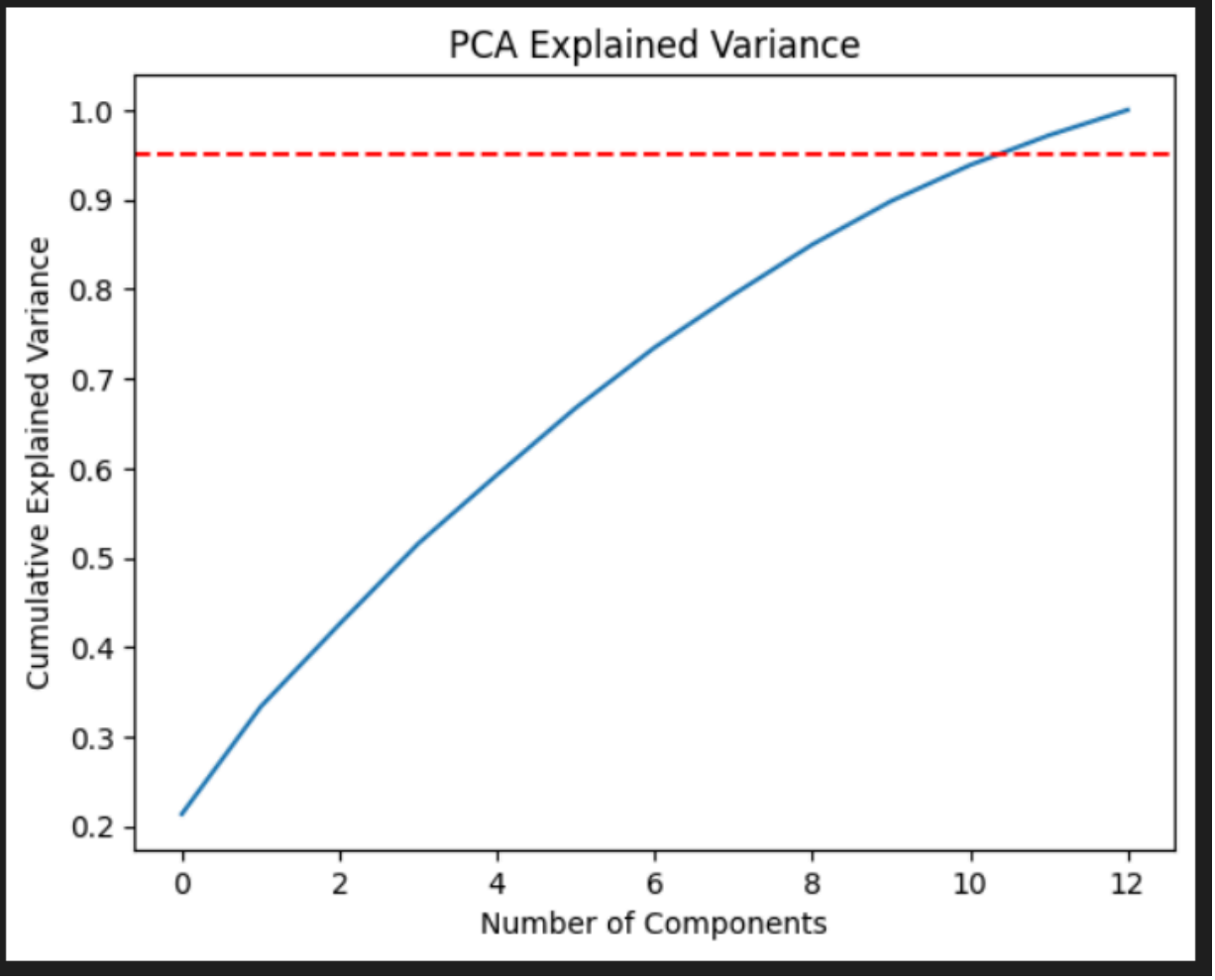

Fig 10. PCA and variance

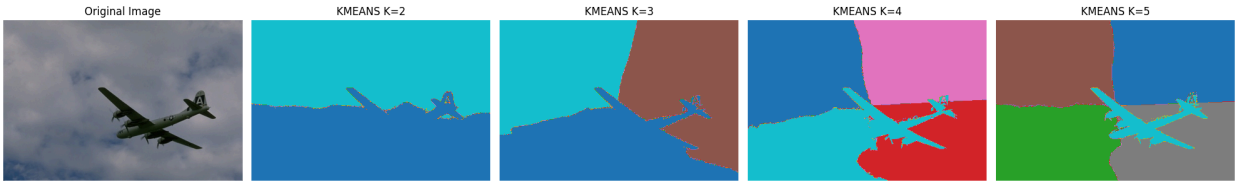

Fig 11. KMeans on image1

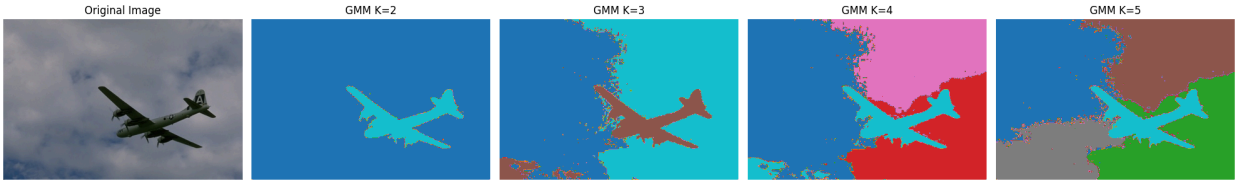

Fig 12. GMM in image1

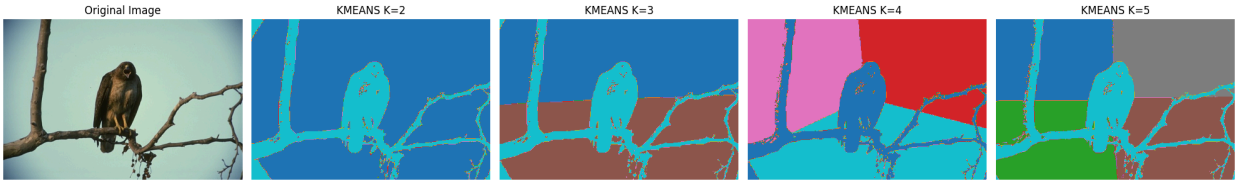

Fig 13. KMeans on image2

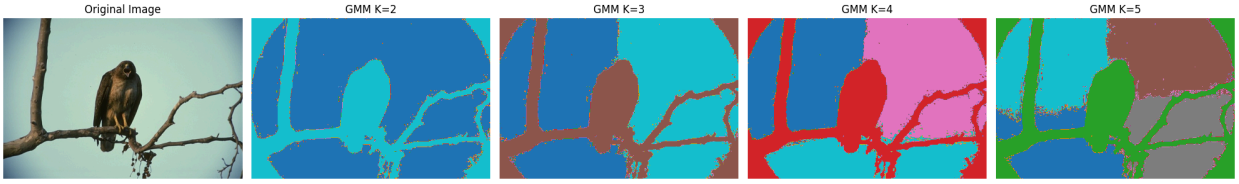

Fig 14. GMM in image2

1. General Observation: Geometric vs. Organic Boundaries

- **K-Means Results:** The most striking artifact in the K-Means results is the tendency to produce straight, geometric boundaries within uniform regions. This is clearly visible in the sky of the Hawk image (K=3, 4, 5) and the Airplane image (K=3, 4).
    - **Reason:** K-Means assumes clusters are spherical in the 5-dimensional feature space (row, col, R, G, B). When the color variance is low (e.g., a solid blue sky), the algorithm minimizes the total variance by splitting the large spatial area into smaller, compact spatial chunks. It behaves like a Voronoi diagram, creating arbitrary "territories" based on spatial coordinates rather than actual texture differences.
- **GMM Results:** GMM produced organic, irregular boundaries that better followed the natural textures of the image, such as the shape of the clouds in the "Airplane" image.
    - **Reason:** GMM learns a full covariance matrix for each cluster, allowing for ellipsoidal shapes in the 5D space. This means it can model a cluster that is narrow in color (very specific blue) but wide in space (spanning the

whole image), without feeling the need to chop it into smaller spatial blocks.

2. Case Study 1: The Hawk (Uniform Background)

Observation: Both K-Means (K=3) and GMM (K=3) split the sky into two separate clusters.

Why this happens: This occurs because we included spatial coordinates (Row, Col) in the feature vector.

Since the sky occupies a massive portion of the image, the spatial variance is huge. Even GMM finds it statistically "cheaper" to fit two separate Gaussian distributions (one for the left side, one for the right) rather than trying to stretch a single Gaussian across the entire width of the image.

The Critical Difference:

- **K-Means Split:** Look at the boundary in KMEANS K=3 (Hawk). It is a sharp, straight line that looks like a geometric cut. This is the "Voronoi" artifact.
- **GMM Split:** Look at the boundary in GMM K=3 (Hawk). The boundary between the blue and brown sky regions is softer and more curved. While it still unnecessarily splits the sky, the boundary follows the slight lighting gradient (vignetting) of the photo rather than an arbitrary straight line.

3. Case Study 2: The Airplane (Textured Background)

In the Airplane image, the sky represents a highly textured, high-variance region (clouds).

- **K-Means Failure:** At K=4, K-Means ignores the texture and imposes arbitrary linear boundaries (visible as large geometric polygons), partitioning the sky purely based on spatial distance to minimize variance.
- **GMM Behavior:** GMM (K=4) also incorrectly fragments the sky, but the boundaries are jagged and irregular rather than geometric. This fragmentation occurs because the algorithm is forced to fit multiple Gaussian distributions to the single, high-variance cloud texture. The resulting segments likely correspond to subtle shifts in illumination or cloud density across the image, rather than meaningful semantic objects.

Comparative Conclusion:

Both algorithms struggled to treat the large background regions (sky) as a single coherent segment when K was set higher than necessary (K > 2). However, their failure modes highlight their fundamental theoretical differences:

- **K-Means (Geometric Artifacts):** Due to its assumption of spherical clusters, K-Means was heavily biased by the spatial features (X, Y). It partitioned the continuous sky into rigid, polygonal regions with straight boundaries (Voronoi

artifacts) to minimize total variance, effectively ignoring the visual continuity of the scene.
- **GMM (Statistical Overfitting):** GMM, with its ability to model covariance, was less constrained by geometry but more sensitive to local texture variance. Instead of straight lines, it produced jagged, noisy segments that overfit the subtle variations in cloud density and lighting.

Ultimately, neither method is purely correct for these specific images at high K. K-Means imposes an artificial geometric structure, while GMM gets distracted by internal texture variance. This demonstrates that adding spatial coordinates to color features introduces a trade-off: it helps group local objects but forces algorithms to arbitrarily split large, continuous background regions.