Algorithms KNN, SVM and KMeans Report

**Author:** Livan Miranda **PID:** 6392173

# 1. Introduction

This project evaluates supervised and unsupervised learning methods on the lncRNA\_5\_Cancers dataset with five cancer classes. For classification, we apply K-Nearest Neighbors (KNN) and Support Vector Machines (SVM) with Linear, Polynomial, and RBF kernels. All classifiers are trained and assessed using 5-fold stratified cross-validation to preserve class proportions. We report macro Accuracy, Precision, Recall, and F1, and provide overall confusion matrices, as well as OvR ROC-AUC and OvR PR-AUC curves to capture performance across all classes. Prior to modeling, features are standardized, and class distribution is visualized to check for imbalance. For unsupervised analysis, we apply K-Means clustering with K = 2, 3, 4, 5, 6, 7. We visualize clusters in PCA-reduced 2D space (points colored by true labels for reference) and estimate the optimal number of clusters via visual inspection, the Elbow method (Inertia vs. K), and the Silhouette score.

**A graph of blue rectangular bars

AI-generated content may be incorrect.**

**Figure 1. Class counts in the Dataset**

# 2. Methods / Algorithms / Tools

For each approach, we describe: (a) What it is / what it does, (b) How it works, (c) Application to this dataset.

## 2.1 k-Nearest Neighbors (KNN)

(a) What: KNN is a lazy, instance-based classifier that predicts a sample’s label by majority vote among its k closest training points. It has no explicit training phase beyond storing the data.

(b) How: We use Euclidean distance on standardized features so each dimension contributes comparably. With k=5 neighbors, the predicted class is the most common among the 5 nearest points; ties are broken by class order and distances as implemented in scikit-learn.

(c) Application: Implemented as KNeighborsClassifier(n\_neighbors=5) inside a Pipeline with StandardScaler. Evaluated with 5-fold Stratified CV. We report macro-averaged metrics and plot OvR ROC/PR curves from predict\_proba outputs.

A graph of a positive rate

AI-generated content may be incorrect. **A graph of a number of objects

AI-generated content may be incorrect.**

**Figure 2. ROC-AUC curves for KNN Figure 3. PR-AUC curves KNN**

A blue squares with white text

AI-generated content may be incorrect.

**Figure 4. KNN Confusion Matrix**

## 2.2 Support Vector Machines (SVM)

### 2.2.1 Linear SVM

(a) What: Finds a separating hyperplane that maximizes the margin between classes in the original feature space.

(b) How: Optimizes hinge-loss with C regularization. Only support vectors affect the boundary. Scores come from the signed distance to the hyperplane.

(c) Application: LinearSVC(dual="auto", max\_iter=5000) within a scaled Pipeline. We compute decision\_function scores for OvR ROC/PR across five classes.

A graph of a function

AI-generated content may be incorrect. **A graph of a number of objects

AI-generated content may be incorrect.**

**Figure 5. ROC-AUC curves for SVM LINEAR Figure 6. PR-AUC curves SVM LINEAR**

**A graph of a diagram

AI-generated content may be incorrect.**

**Figure 7. SVM LINEAR Confusion Matrix**

### 2.2.2 Polynomial SVM

(a) What: Introduces nonlinear decision boundaries by implicitly considering polynomial feature interactions up to degree d.

(b) How: Uses the polynomial kernel K(x,x') = (γ x·x' + coef0)^d. γ controls scaling of the dot product, coef0 shifts the polynomial, and d sets interaction order.

(c) Application: SVC(kernel="poly", degree=2, gamma="scale", coef0=1.0, C=1.0, probability=False) in a scaled Pipeline. Decision scores from decision\_function feed OvR ROC/PR.

**A graph of a function

AI-generated content may be incorrect.**   **A graph of a number of objects

AI-generated content may be incorrect.**

**Figure 8. ROC-AUC curves for SVM POLYNOMIAL Figure 9. PR-AUC curves SVM POLYNOMIAL**

**A graph of a diagram

AI-generated content may be incorrect.**

**Figure 10. SVM POLYNOMIAL Confusion Matrix**

### 2.2.3 RBF SVM

(a) What: Models complex, localized boundaries using the Gaussian (RBF) kernel.

(b) How: Kernel K(x,x') = exp(-γ ||x - x'||^2). Larger γ yields more local, flexible boundaries; C trades margin width for slack (misclassification).

(c) Application: SVC(kernel="rbf", gamma="scale", C=1.0, probability=False). We standardize features and use 5-fold Stratified CV. Note: RBF can run slower on high-dimensional datasets but often improves accuracy/AUC when classes are not linearly separable.

**A graph of a function

AI-generated content may be incorrect.**   **A graph of a number of objects

AI-generated content may be incorrect.**

**Figure 11. ROC-AUC curves for SVM RBF Figure 12. PR-AUC curves SVM RBF**

**A graph of a diagram

AI-generated content may be incorrect.**

**Figure 13. SVM RBF Confusion Matrix**

## 2.3 KMeans Clustering

(a) What: Unsupervised partitioning of the data into K clusters by minimizing within-cluster sum of squares (inertia).

(b) How: The algorithm alternates assignments and centroid updates until convergence. We assess the number of clusters via the Elbow method (inertia vs K) and the Silhouette score (mean per-sample cohesion vs separation).

(c) Application: KMeans with n\_init=10 evaluated for K=2..7 on standardized features. We visualize in PCA(2D), coloring points by true labels for reference and marking centroids.

**Figure 14-19. Cluster’s Graph for raw features [PCA(2)] for visualization only**

**A diagram of different colored dots

AI-generated content may be incorrect.A diagram of different colored squares

AI-generated content may be incorrect.**

**A diagram of different colored squares

AI-generated content may be incorrect.A diagram of a diagram

AI-generated content may be incorrect.**

**A diagram of different colored shapes

AI-generated content may be incorrect.A diagram of a graph

AI-generated content may be incorrect.**

**Figure 20 - 25. Cluster’s Graph for features [PCA(100)]**

**A screenshot of a computer screen

AI-generated content may be incorrect.A screenshot of a computer screen

AI-generated content may be incorrect.**

**A screenshot of a computer game

AI-generated content may be incorrect.A screenshot of a computer game

AI-generated content may be incorrect.**

**A screen shot of a computer screen

AI-generated content may be incorrect.** **A screenshot of a computer screen

AI-generated content may be incorrect.**

**Figure 26 - 31. Cluster’s Graph for features [PCA(100)] + UMAP**

*A graph with a line

AI-generated content may be incorrect.*

**Figure 32. Elbow**

*[Insert Figure 8 here: results/figures/task3\_k5\_pca.png]*

**Figure 15. Silhouette score vs K.**

# 3. Results

Table 1. Calculated Mean from the Aggregated Cross-Validation Metrics (Macro)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Model** | **Accuracy** | **Precision** | **Recall** | **F1** |
| KNN | 0.954 | 0.955 | 0.954 | 0.953 |
| Linear SVM | 0.980 | 0.981 | 0.980 | 0.980 |
| Poly SVM | 0.981 | 0.981 | 0.980 | 0.980 |
| RBF SVM | 0.957 | 0.962 | 0.957 | 0.958 |

Description: Macro-averaged metrics across 5 folds of Stratified CV on standardized features.

SVM scores use decision\_function

KNN uses predict\_proba.

Observations:

Linear & Poly SVM lead (≈0.98 across metrics), essentially tied.

RBF SVM trails (≈0.96), suggesting data is near-linearly separable or needs tuning.

KNN is solid but lower (≈0.95); distance-based methods suffer in high-dimensional space.

Macro metrics are closely matched. No single class dominates performance.

Conclusion:

Pick Linear SVM: best accuracy–efficiency balance and simplest to deploy.

Poly SVM: near-identical accuracy if mild nonlinearity is desired.

RBF SVM: consider only with proper hyperparameter tuning.

KNN: keep as a baseline/reference.

Observation:

* The dataset contains a total of 2,529 samples across five cancer types.
* KIRC has the highest number of samples (527), while PRAD has the fewest (493).
* The counts across cancer types are relatively balanced, with differences between the largest and smallest class being less than 7% of the total dataset size. This balance is advantageous for model training, as it reduces the risk of bias towards a single class.

Conclusion:

The dataset is well-suited for classification tasks since all classes have similar representation. No additional resampling or class-weighting techniques appear necessary before training. Balanced class distribution helps ensure that evaluation metrics like accuracy, F1-score, and AUC remain reliable and unbiased.

**Figure 8. PCA visualization for KMeans (K=5) with points colored by true labels.**

# 5. Conclusion