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**

Interpretation

Bar chart of samples per class shows a fairly balanced dataset: KIRC (527), LUAD (510), LUSC (498), PRAD (493), THCA (501).

Observation

Only mild imbalance (KIRC a bit higher, PRAD a bit lower). This reduces the risk of biased metrics and makes macro-averaging appropriate.

Conclusion

No heavy rebalancing was required; stratified CV should preserve these proportions across folds.

# 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**

Interpretation

ROC curves (TPR vs FPR) for each class using one-vs-rest; the diagonal is chance.

Precision–Recall curves per class under one-vs-rest.

Observations

**ROC:** All curves hug the top-left; macro AUC ≈ 0.99. Per-class AUC is ~0.98–1.00; LUAD/LUSC are very strong but marginally below THCA/PRAD, which are essentially perfect.

**PR:** Curves stay near the top edge (high precision across recalls). PRAD and THCA are ~1.00 AP; KIRC is ~0.99; LUSC ~0.96; LUAD ~0.94, indicating LUAD is the hardest class for KNN at high recall.

Conclusion

KNN separates classes extremely well; the classifier achieves near-perfect ranking performance across classes.

A blue squares with white text

AI-generated content may be incorrect.

**Figure 4. KNN Confusion Matrix**

Interpretation

Counts of true vs predicted labels across all test folds.

Observation

Strong diagonal dominance:

KIRC: 517/527 correct; a few mislabels to THCA/LUSC.

LUAD: 444/510 correct; main confusions to LUSC (27) and THCA (38).

LUSC: 457/498 correct; confusions to LUAD (29) and THCA (10).

PRAD: 493/493 correct (perfect).

THCA: 501/501 correct (perfect).

Conclusion

Errors concentrate between LUAD and LUSC (biologically plausible similarity). PRAD and THCA are very distinctive for KNN; KIRC is also highly separable.

## 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**

Interpretation

ROC curves per class; TPR vs FPR in one-vs-rest.

Precision vs Recall per class using one-vs-rest; legend shows AP.

Observation

Curves hug the top-left; AUC ≈ 1.00 for KIRC/PRAD/THCA and ~0.99 for LUAD/LUSC.

PRAD/THCA ≈ 1.00 across recalls; KIRC ~0.99; LUAD/LUSC dip modestly at high recall (AP ~0.97–0.98).

Conclusion

Linear SVM achieves near-perfect separability; only slight degradation for LUAD/LUSC at very low FPR.

Precision remains high throughout; minor losses for LUAD/LUSC reflect harder decision boundaries.

**A graph of a diagram

AI-generated content may be incorrect.**

**Figure 7. SVM LINEAR Confusion Matrix**

Interpretation

Aggregated over 5 folds; entries show counts of true vs predicted classes.

Observation

Strong diagonal dominance (e.g., KIRC/PRAD/THCA nearly perfect). Most residual errors are LUAD ↔ LUSC.

Conclusion

Linear SVM provides robust class separation overall; remaining confusion is concentrated between the two lung cancer subtypes.

### 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**

Interpretation

ROC curves show TPR vs FPR per class; AUC summarizes class-wise separability.

Precision vs Recall per class; AP summarizes precision across all recall levels.

Observation

**ROC-AUC:** Curves hug the top-left; AUC ≈ 1.00 for KIRC/PRAD/THCA and ~0.99 for LUAD/LUSC.

**PR-AUC :** PRAD/THCA ~1.00 AP; KIRC ~0.99; LUAD/LUSC ~0.98 with a slight precision drop at extreme recall.

Conclusion

Poly SVM achieves near-perfect discrimination; only minor degradation for LUAD/LUSC at very low FPR.

Model maintains very high precision across recalls; LUAD/LUSC are comparatively harder but still strong.

**A graph of a diagram

AI-generated content may be incorrect.**

**Figure 10. SVM POLYNOMIAL Confusion Matrix**

Interpretation

Counts of true vs predicted labels aggregated over CV folds.

Observation

Dominant diagonal: most errors are LUAD ↔ LUSC confusions. KIRC/PRAD/THCA are almost perfectly classified.

Conclusion

Polynomial SVM provides robust class separation; remaining mistakes align with the biologically similar LUAD/LUSC pair.

### 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**

Interpretation

ROC shows TPR vs FPR per class using one-vs-rest.

PR shows precision vs recall per class with AP in the legend.

Observation

**ROC-AUC :** All curves hug the top-left; KIRC/PRAD/THCA ≈ 1.00 AUC; LUAD/LUSC ≈ 0.99 but slightly below the others at low FPR.

**PR-AUC :** PRAD/THCA sustain ≈1.00 precision across recalls; KIRC ≈1.00 AP; LUAD (~0.96) and LUSC (~0.97) dip as recall approaches 1.0.

Conclusion

RBF achieves near-perfect separability overall; the slight gap for LUAD/LUSC mirrors their harder boundary.

RBF maintains very strong precision, with minor degradation for LUAD/LUSC at high recall, consistent with their overlap.

**A graph of a diagram

AI-generated content may be incorrect.**

**Figure 13. SVM RBF Confusion Matrix**

Interpretation

Counts of true vs predicted labels aggregated across folds.

Observation.

Strong diagonal dominance: most residual errors are LUAD ↔ LUSC and a few KIRC ↔ LUSC/PRAD. THCA is nearly perfectly classified.

Conclusion

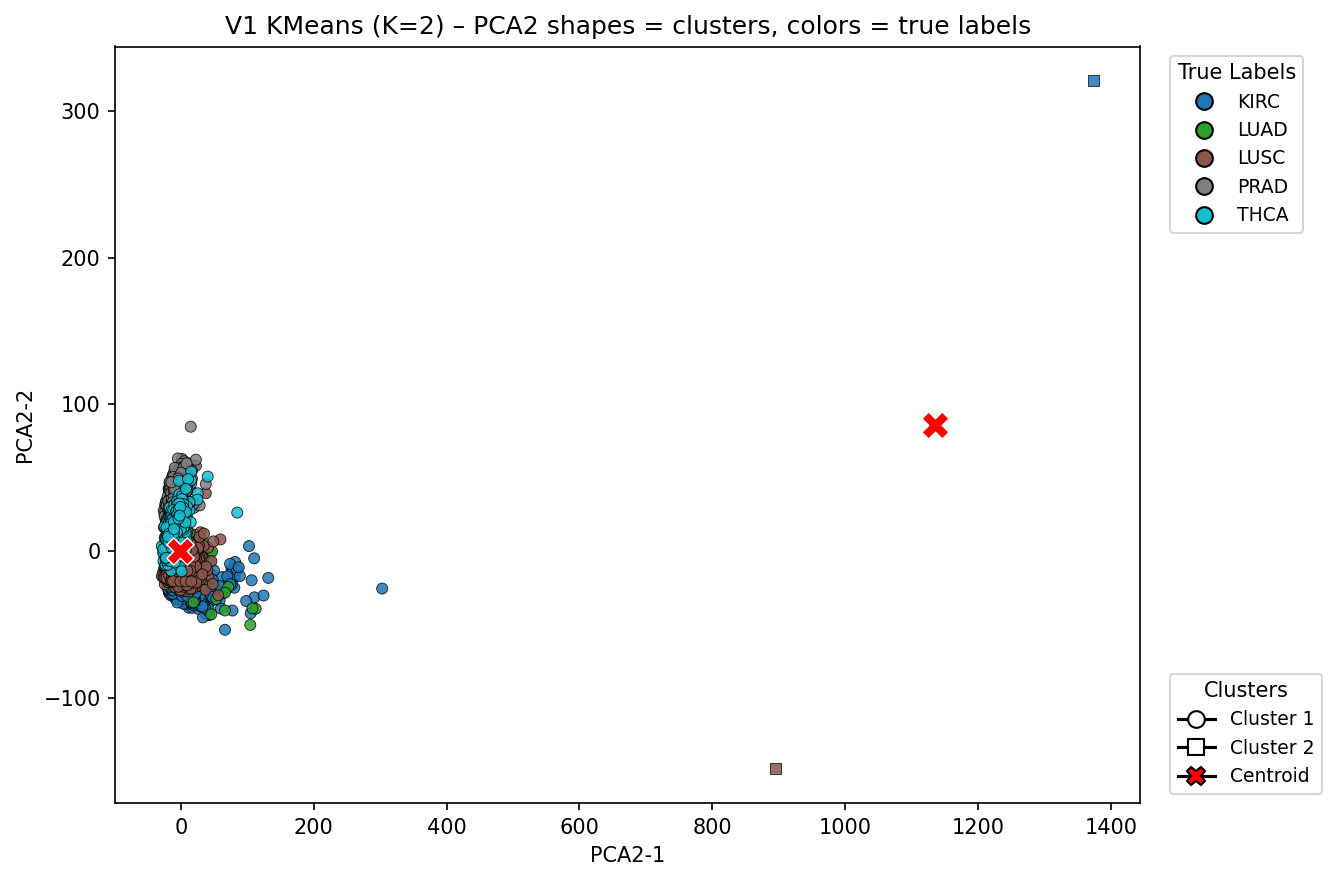
RBF provides robust class separation; remaining mistakes concentrate in the biologically similar LUAD/LUSC pair.

## 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.

A screen shot of a computer

AI-generated content may be incorrect.

**A screen shot of a graph

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

AI-generated content may be incorrect.**

**A screen shot of a computer

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

AI-generated content may be incorrect.**

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

Interpretation

Points are KMeans clusters trained on the full standardized feature space and then projected to PCA(2) for visualization; the red “X” marks are the cluster centroids shown in that same 2-D space.

Observation

Several red centroids sit far from the dense point clouds even though most samples are tightly packed around the origin.

Conclusion

Outliers pull: Centroids are means, so even a few extreme samples in a cluster strongly pull the mean in PCA(2).

Projection effect: Clustering happens in high-D; the PCA(2) projection magnifies variance along the top components. A small number of high-leverage points get pushed far out in 2-D, so the mean of projected points can land well away from the main mass.

Imbalanced cluster density: When a cluster contains many tight points plus a tiny outlier subset, the mean shifts toward the outliers, making the centroid appear “in empty space.”

KMeans sensitivity: Despite k-means++ and multiple restarts, KMeans still optimizes inertia with L2 and is not robust to outliers.

Practical fixes (already reflected in V2 and recommended for V1):

Compute centroids in high-D and then project the centroids via the same PCA transform (rather than averaging after projection).

Down-weight or mask outliers for centroid computation only (keep them visible), e.g., z-score > 3 per PC.

Prefer clustering on PCA(100) (In V2): it regularizes variance and produces centroids that sit inside the visible cluster bodies (In PCA/UMAP plots confirm this).

**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)]**

Interpretation

Each scatter plot shows the samples embedded into 2D using PCA (after first selecting the top 5,000 most variable genes and compressing with PCA(100)).

Color = true cancer label (KIRC, LUAD, LUSC, PRAD, THCA).

Marker shape = KMeans-assigned cluster ID.

Red X = centroid of each cluster in this 2D space.

So visually we can ask: do clusters the algorithm finds line up with the “real” cancer types? If yes, most points of one color should share the same marker shape.

Observation.

For K = 2, the data mostly splits into two broad groups. One group is dominated by KIRC + LUAD + LUSC (mostly teal/green/brown), and the other group tends to pick up PRAD and THCA (gray/cyan), but there is still heavy mixing across cancers. Two clusters are clearly too coarse to reflect five biological classes.

For K = 3 and K = 4, we start to see separation: some cancer types (especially PRAD and THCA) concentrate into specific clusters with distinct shapes. Centroids are now sitting closer to dense clouds, not off in extreme outlier space, which means they better represent the “center” of each discovered group in this PCA representation.

Around K = 5 or K = 6, structure becomes more biologically meaningful:

We start seeing clusters that align strongly with a single color.

LUAD vs LUSC (which are known to be harder to distinguish) still partially overlap, but PRAD and THCA tend to occupy tighter regions with their own cluster shapes.

The centroids (red X’s) sit in the middle of dense local subclouds rather than being dragged into noise, suggesting the algorithm is stabilizing.

By K = 7, we get additional micro-clusters. These extra clusters often “slice up” what used to be one cancer region into multiple subclusters, including gradients within LUAD/LUSC. That can mean KMeans is starting to overfit fine-grained gene-expression substructure instead of giving clean cancer-type partitions.

Conclusion.

The dataset clearly does not behave like “just 2 clusters.”

As K increases from 2 → 6, clusters align more closely with true cancer classes: PRAD and THCA become almost visually separable, and KIRC carves out its own region.

Past K ≈ 5–6, adding more clusters mostly fragments existing cancer groups instead of revealing brand-new biology.

**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**

Interpretation

UMAP projects the PCA(100) features to 2D; points are colored by true labels and marker shape = cluster ID. Red “X” marks are the K-Means centroids computed in the UMAP space. Compact, well-separated blobs indicate structure K-Means can capture.

Observations

K=2. Two large clusters split the space roughly along a global trend. Several cancer types (e.g., PRAD, THCA) sit mostly inside a single blob; LUAD/LUSC share a mixed region. Centroids land near the densest cores.

K=3. The mixed middle region starts to separate; one centroid moves into the LUAD/LUSC band while the other two anchor the outer blobs (e.g., PRAD/THCA vs. KIRC area).

K=4. Clearer partition of the central band; PRAD and THCA remain tight/compact; KIRC’s spread is captured with a dedicated centroid. Overall separation improves without obvious fragmentation.

K=5. Substructure emerges: LUAD vs. LUSC become more distinguishable, and one extra centroid captures a denser sub-region of KIRC/THCA. Gains are visible but smaller than K=3→4.

K=6. Further refinement of the central band; centroids track local density pockets. Some clusters get small/tight (risk of over-splitting begins).

K=7. Marginal changes vs. K=6; additional centroid mostly slices an already-coherent blob—limited new structure captured.

Conclusion.

UMAP reveals five compact, well-structured regions with a moderately mixed LUAD/LUSC band. Visual quality improves notably up to K≈4–5; beyond that, splits look incremental and risk over-fragmentation. Combined with the Elbow/Silhouette trends, K=4 or K=5 is a defensible choice for this representation.

**A graph with a line

AI-generated content may be incorrect.** **A graph with orange lines

AI-generated content may be incorrect.**

**Figure 32. Elbow Method for raw features Figure 33. Silhouette score for raw features**

Interpretation

Inertia (within-cluster SSE) vs. K; look for the “elbow” where added clusters stop yielding large SSE drops.

Mean silhouette in −1,1-1,1−1,1; higher is better (dense, well-separated clusters).

Observation

**Elbow:** Big decreases from K=2→3→4, then clearly diminishing returns after K≈4–5 (curve flattens).

**Silhouette:** Very high scores at K=2 (~0.9) and K=3 (~0.88–0.9), followed by a sharp drop for K≥4 (≈0.12 at K=4 and near-zero afterward).

Conclusion

High silhouette at small K and rapid degradation as K grows is typical for high-dimensional gene data: most separation is captured by a few broad groups, and extra clusters fragment coherent structure. The PCA/UMAP views on the reduced features (V2) help visualize this and can refine the choice, but V1 metrics alone point to K≈2–5.

**A graph with a green line

AI-generated content may be incorrect.** **A graph with a red line

AI-generated content may be incorrect.**

**Figure 34. Elbow Method for PCA(100) Figure 35. Silhouette score for PCA(100)**

Interpretation

Inertia vs K after variance filtering + PCA; look for elbow.

Average silhouette after dimensionality reduction.

Observation

**Elbow:** Clearer curvature: large drop up to K=4–5, then noticeably smaller gains toward K=6–7.

**Silhouette:** Observation. Scores improve from K=2 → 4, peak around K=4–6 (highest at K=6 by a small margin), then dip slightly at 7.

Conclusion

V2 suggests K=4–6 as a sensible range, with K≈4–5 a strong balance of fit vs complexity.

Best separation in V2 occurs around K=4–6 (often K=4 or K=6 depending on our preference for fewer/more clusters).

# 3. Results

Table 1. Calculated Mean from the Aggregated Cross-Validation Metrics (Macro)per class in KNN

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Accuracy** | **Precision** | **Recall** | **F1** |
| KIRC | 0.996 | 0.998 | 0.981 | 0.989 |
| LUAD | 0.962 | 0.937 | 0.871 | 0.902 |
| LUSC | 0.972 | 0.940 | 0.918 | 0.929 |
| PRAD | 0.999 | 0.996 | 1 | 0.998 |
| THCA | 0.978 | 0.901 | 1 | 0.948 |

Table 2. Calculated Mean from the Aggregated Cross-Validation Metrics (Macro)per class in SVM-L

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Accuracy** | **Precision** | **Recall** | **F1** |
| KIRC | 0.999 | 0.998 | 0.996 | 0.997 |
| LUAD | 0.981 | 0.938 | 0.971 | 0.954 |
| LUSC | 0.981 | 0.967 | 0.934 | 0.950 |
| PRAD | 1 | 1 | 1 | 1 |
| THCA | 1 | 1 | 1 | 1 |

Table 3. Calculated Mean from the Aggregated Cross-Validation Metrics (Macro)per class in SVM-POLY

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Accuracy** | **Precision** | **Recall** | **F1** |
| KIRC | 0.998 | 0.998 | 0.994 | 0.996 |
| LUAD | 0.981 | 0.934 | 0.971 | 0.954 |
| LUSC | 0.981 | 0.969 | 0.936 | 0.952 |
| PRAD | 0.999 | 0.998 | 1 | 0.999 |
| THCA | 1 | 1 | 1 | 1 |

Table 4. Calculated Mean from the Aggregated Cross-Validation Metrics (Macro)per class in SVM-RBF

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Accuracy** | **Precision** | **Recall** | **F1** |
| KIRC | 0.995 | 1 | 0.975 | 0.988 |
| LUAD | 0.960 | 0.852 | 0.971 | 0.907 |
| LUSC | 0.970 | 0.951 | 0.892 | 0.920 |
| PRAD | 0.992 | 1 | 0.961 | 0.980 |
| THCA | 0.997 | 1 | 0.984 | 0.992 |

Table 5. 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.

# 5. 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.