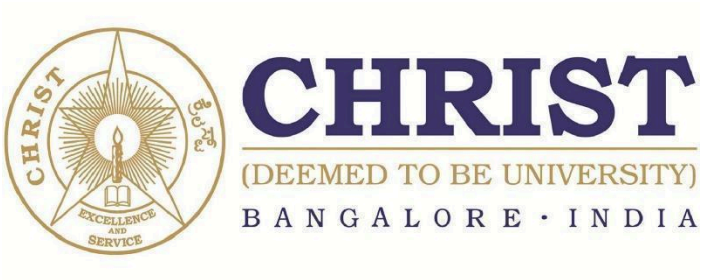


II Trimester MSc (AI & ML)

Advanced Machine Learning

Department of Computer Science

GENE EXPRESSION CANCER RNA-SEQ MULTI-MODEL ANALYSIS

by

PRIYANSHA UPADHYAY: 2348541

SATYAM KUMAR: 2349555

MANAV GUPTA :2348529

January 2024

CERTIFICATE

*This is to certify that the report titled "**Gene Expression Cancer RNA Sequence Multi-Model Analysis"** is a bona fide record of work done by **PRIYANSHA UPADHYAY: 2348541, SATYAM KUMAR: 2349555, MANAV GUPTA :2348529** of CHRIST(Deemed to be University), Bangalore, in partial fulfillment of the requirements of II Trimester of Msc Artificial Intelligence and Machine Learning during the year 2023-24.*

Course Teacher

Valued-by: (Evaluator Name & Signature)

1.

2.

Date of Exam:

Contents	
Topic	**Page Number**

Team Details

Reg. no	Name	Summary of tasks performed
2348529	**Manav Gupta**	- Implemented k Nearest Neighbors (KNN) algorithm. - Conducted analysis and evaluation of the KNN model on the given dataset. - Explored and implemented Hierarchical Clustering using different linkage methods. - Evaluated the performance of the Hierarchical Clustering models.
2348541	**Priyansha Upadhyay**	- Implemented Support Vector Machine (SVM) algorithm. - Tuned hyperparameters for SVM and evaluated performance. - Implemented Logistic Regression model along with t-SNE & PCA. - Applied PCA and SMOTE in conjunction with SVM.
2348555	**Satyam Kumar**	- Implemented Decision Tree algorithm. - Conducted an in-depth analysis of the Decision Tree model's performance.. - Implemented LightGBM algorithm. - Evaluated the performance of the LightGBM model.

1. Abstract

This project delves into the exploration and analysis of a gene expression dataset comprising ***801 samples and 20,532 features***. The project strategically incorporates state-of-the-art machine learning algorithms, including ***Support Vector Machines (SVM), Logistic Regression, Decision Tree, LightGBM (Gradient Boosting), Hierarchical Clustering, and k-Nearest Neighbors (KNN)***, to analyze the wealth of information derived from cancer transcriptome sequencing.

In addressing challenges such as high dimensionality and class imbalance, advanced dimensionality reduction techniques such as *t-SNE and PCA, along with Synthetic Minority Over-sampling Technique (SMOTE)*, are implemented.

Additionally, the project endeavors to implement and critically assess the research papers and techniques in the domain of gene expression analysis. By amalgamating the power of RNAseq with cutting-edge machine learning methodologies, this research aims to unravel hidden patterns in cancer transcriptomes. The project aspires not only to contribute to the current understanding of cancer biology but also to provide a robust foundation for future endeavors in *leveraging RNAseq data for cancer diagnosis, prognosis, and drug development.*

2. Introduction

Gene expression analysis is a cornerstone of modern molecular biology, offering unparalleled insights into cellular processes, disease mechanisms, and potential therapeutic targets. As genomic datasets continue to grow in complexity, there is a pressing need to deploy advanced computational techniques to extract meaningful patterns and relationships from these vast repositories of genetic information.

This project focuses on a gene expression dataset featuring 801 samples, each characterized by a staggering 20,532 features. The inherent complexity and dimensionality of such datasets present challenges that demand innovative solutions. Traditional machine learning models, while powerful, may struggle to discern subtle patterns within this high-dimensional space. Consequently, the project aims to assess the performance of a diverse array of algorithms, ranging from conventional SVM and Logistic Regression to more sophisticated Decision Tree, LightGBM, Hierarchical Clustering, and KNN models.

To address the challenges posed by high dimensionality and class imbalance inherent in gene expression datasets, the project employs advanced dimensionality reduction techniques such as t-SNE and PCA. Furthermore, class imbalance is tackled using the Synthetic Minority Over-sampling Technique (SMOTE) to create a balanced representation of the classes, thereby preventing model bias.

Through this comprehensive approach, the project aims to provide valuable insights into the relative strengths and limitations of various machine learning algorithms and research-driven methodologies in the context of gene expression analysis. The subsequent sections will delve into the methodology, results, and discussions, shedding light on the intricate relationships within the dataset and offering practical guidance for future endeavors in this critical domain of biological research.

3. Data Processing and Exploration

3.1. Data Understanding and Exploration

The dataset under investigation originates from the UCI repository, specifically the *TCGA-PANCAN-HiSeq-801x20531 dataset*, accessible at UCI Repository Link. This dataset encapsulates the intricate landscape of gene expression in cancer samples, comprising *801 samples and 20,531 features*.

Evolution of Transcriptomics and Gene Expression Analysis

Transcriptomics, focusing on the quantitative examination of the transcriptome, has evolved significantly over the years. Transcriptomics, initially reliant on Sanger sequencing of EST libraries, transitioned to tag-based methods like SAGE. The advent of DNA microarrays and RNA-Seq marked a transformative shift in gene expression estimation.

Microarray Data

Microarrays, with thousands of spots representing genes, allow genome-wide expression profiling. Despite limitations in accuracy and sensitivity to environmental changes, they find applications in cancer research and drug development.

RNA-Seq Data

RNA-Seq, a rapid NGS method, excels in profiling the transcriptome. It surpasses microarrays, offering enhanced gene discovery, isoform identification, resolution, noise reduction, and cost-effectiveness. Single-cell RNA sequencing (scRNA-Seq) extends this capability, enabling high-throughput single-cell analysis.

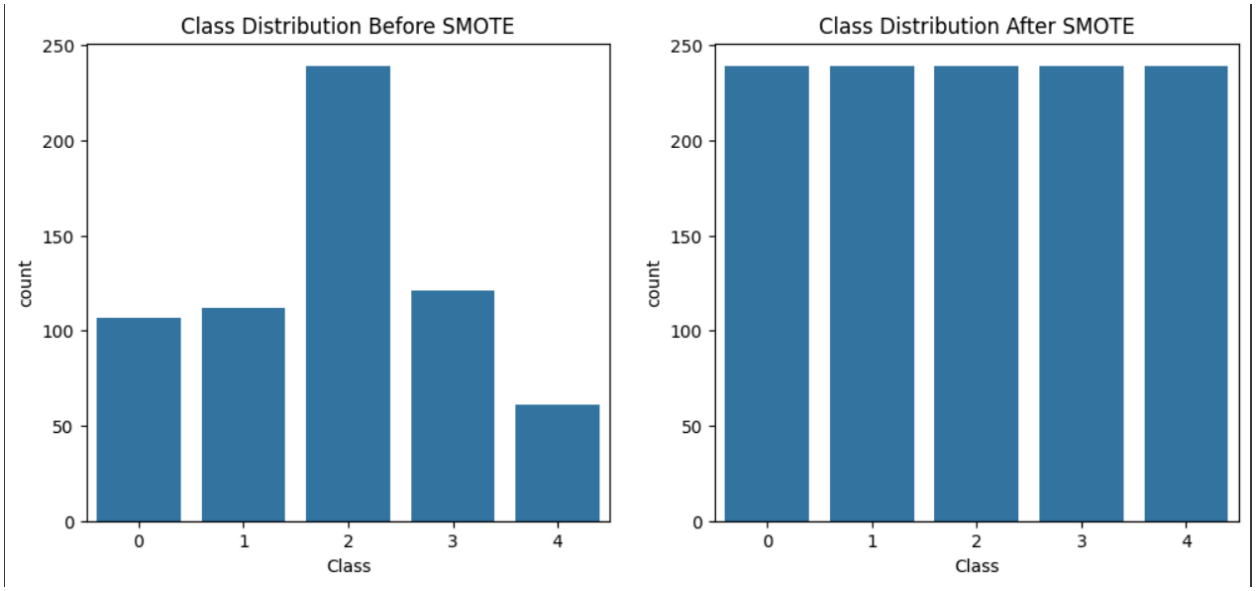
3.2. Data Cleaning and Handling Missing Values

The initial assessment of the dataset reveals a robust quality, with no missing values and no duplicate columns present. This absence of data gaps and redundancy streamlines the data cleaning process and ensures a solid foundation for subsequent analyses.

3.3. Data Integration and Feature Engineering

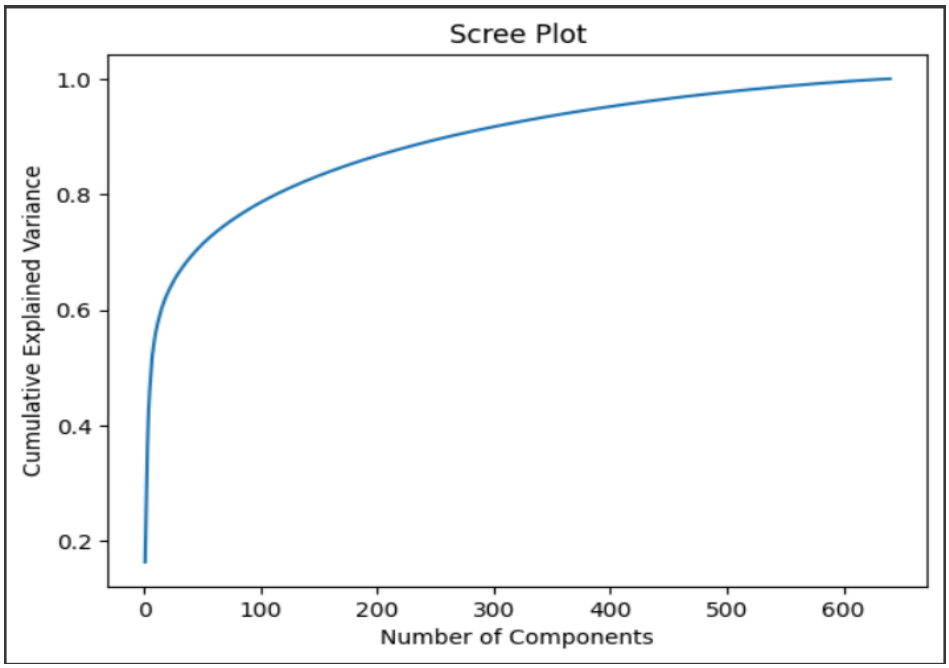
Data Integration: The initial dataset was split into two parts: 'data' and 'labels.' To create a unified dataset, these components were combined, resulting in the dataset named 'Tumor_Data.'

For feature engineering, the dataset underwent a crucial step to address class imbalance. The Synthetic Minority Over-sampling Technique (SMOTE) was applied, effectively correcting the imbalanced dataset. This transformation led to an updated dataset with 239 features.



Dig.01:SMOTE

To further enhance the modeling process, Principal Component Analysis (PCA) was employed for dimensionality reduction. The scree plot, based on cumulative explained variance, was instrumental in determining the appropriate number of components.

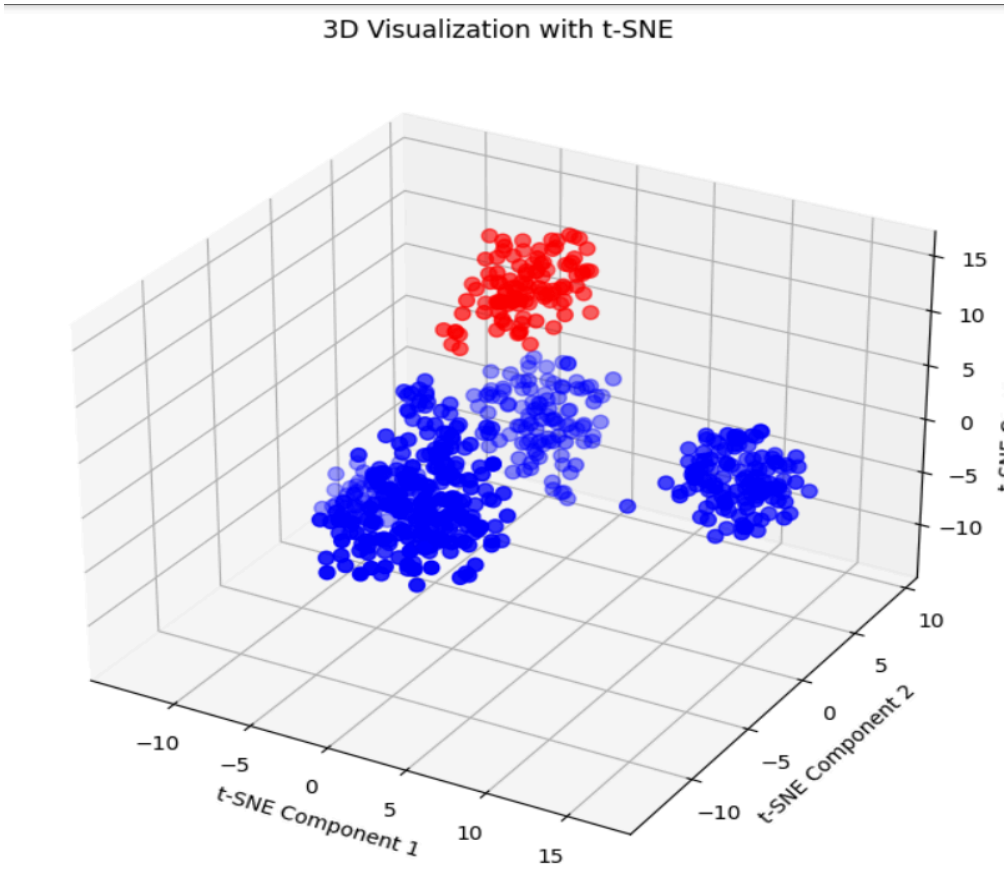


Dig2:ScreePLT

Cross-validation scores were then computed for a range of potential components, including [50, 100, 200, 500, min(X_train.shape[0], X_train.shape[1])]. The results guided the selection of 100 components for subsequent dimensionality reduction.

The PCA-transformed data was then utilized in conjunction with Support Vector Machines (SVM). The combination of PCA and SMOTE with SVM was explored, and cross-validation scores were assessed for each scenario. The outcomes indicated high mean accuracy, particularly for the combination of PCA before SMOTE and SVM (Mean Accuracy: 0.9958).

In preparation for implementing the logistic regression model, additional dimensionality reduction techniques were applied. PCA was performed, and t-Distributed Stochastic Neighbor Embedding (tSNE) was implemented with 2 and 3 components.



Dig03:t-SNE

In summary, the data integration process resulted in the creation of 'Tumor_Data,' and feature engineering steps, including SMOTE and PCA, contributed to a refined dataset ready for model implementation. The subsequent sections delve into the outcomes of these strategies and their impact on model performance.

4. Algorithm Implementation

4.1. Algorithms Implemented

4.1.1. Support Vector Machines

Support Vector Machines (SVM), a powerful classification algorithm, were employed for the gene expression dataset. The dataset, enriched through SMOTE and dimensionality reduction using PCA, served as the input for SVM.

Preprocessing Steps:

- SMOTE: The Synthetic Minority Over-sampling Technique was applied to address class imbalance, ensuring a more representative training set.
- PCA Dimensionality Reduction: PCA was utilized to reduce the dataset's dimensionality, focusing on the top 100 principal components based on scree plot analysis and cross-validation scores.

```
[ ] # Support Vector Machine (SVM) with PCA after SMOTE
    svm_classifier = SVC(kernel='rbf', random_state=42)
    svm_classifier.fit(X_pca_after_smote, y_resampled)
```

```
▾ SVC
SVC(random_state=42)
```

Dig04:SVM

4.1.2. Logistic Regression

Logistic Regression, a versatile classification algorithm, was applied to the gene expression dataset after preprocessing with PCA and t-SNE for dimensionality reduction. The logistic regression model aimed to classify instances within the dataset, leveraging the insights gained from these reduction techniques.

Preprocessing Steps:

- PCA and t-SNE Dimensionality Reduction: The dataset underwent dimensionality reduction using PCA and t-SNE. PCA focused on the top 100 principal components, and t-SNE was implemented with 2 and 3 components.

```
# Build Logistic Regression model
logreg = LogisticRegression(random_state=42)
logreg.fit(X_train_standardized, y_train)
```

```
LogisticRegression
LogisticRegression(random_state=42)
```

Dig05:Logistic Regression

4.1.3. K Nearest Neighbours

k-Nearest Neighbors (KNN), a versatile and intuitive classification algorithm, was applied to the gene expression dataset to discern patterns and relationships within the data.

Preprocessing Steps:

- Standardization: Standardization was performed to bring all features to a common scale, ensuring uniform contributions during modeling.
- PCA Dimensionality Reduction: PCA was employed for dimensionality reduction, focusing on the top 80 principal components determined through scree plot analysis and cross-validation scores.

```
[52] X_train, X_test, y_train, y_test = train_test_split(pca_df, labels, test_size=0.2,
     knn_classifier = KNeighborsClassifier(n_neighbors=5)
     knn_classifier.fit(X_train, y_train)
```

```
KNeighborsClassifier
KNeighborsClassifier()
```

Dig06: K Nearest

4.1.4. Hierarchical Clustering

Hierarchical Clustering, a powerful unsupervised clustering algorithm, was employed to unravel inherent patterns within the gene expression dataset. The algorithm was executed using different linkage methods, including:

- Single Linkage
- Ward Linkage
- Complete Linkage
- Average Linkage

Preprocessing Steps:

- PCA Dimensionality Reduction: PCA was utilized to reduce the dataset's dimensionality, focusing on the top 80 principal components based on scree plot analysis and cross-validation scores.

4.1.5. Decision Tree

Decision Tree, a fundamental and interpretable machine learning algorithm, was implemented to decipher patterns within the gene expression dataset.
The Decision Tree model was trained on the preprocessed dataset. Decision trees are known for their simplicity and interpretability, making them valuable for understanding the underlying structure of the gene expression data.

```
[ ]  # Train a Decision Tree model
     dt_model = DecisionTreeClassifier(random_state=42)
     dt_model.fit(X_train, y_train)
```

```
DecisionTreeClassifier
DecisionTreeClassifier(random_state=42)
```

Dig07: Decision Tree

4.1.6. Light GBM Classifier(Gradient Boost)

Light GBM, a powerful gradient boosting framework, was employed to capture complex relationships and enhance predictive performance on the gene expression dataset.
The Light GBM Classifier was trained on the preprocessed dataset. Leveraging gradient boosting, Light GBM builds an ensemble of decision trees, iteratively refining the model's predictions.

```
[ ]  # Create LightGBM dataset
     train_data = lgb.Dataset(X_train, label=y_train_encoded)

     # Train the LightGBM model
     lgb_model = lgb.train(params, train_data, num_boost_round=50)
```

```
[LightGBM] [Info] Auto-choosing col-wise multi-threading, the overhead of testing
You can set `force_col_wise=true` to remove the overhead.
[LightGBM] [Info] Total Bins 3770888
[LightGBM] [Info] Number of data points in the train set: 640, number of used feat
[LightGBM] [Info] Start training from score -0.985005
[LightGBM] [Info] Start training from score -2.350594
[LightGBM] [Info] Start training from score -1.665678
[LightGBM] [Info] Start training from score -1.742969
[LightGBM] [Info] Start training from score -1.788639
```

Dig08: LightGBM

Preprocessing steps:

- Label Encoding: To convert categorical columns into numerical ones so that they can be fitted by Model.

4.2. Parameter Tuning

Parameter tuning plays a crucial role in optimizing the performance of machine learning models. For the SVM implementation on the gene expression dataset, an exhaustive search over a predefined parameter grid was conducted.

The hyperparameter grid consisted of potential values for the regularization parameter 'C,' the kernel coefficient 'gamma,' and the choice of kernel ('linear,' 'rbf,' 'poly'). The model was trained and evaluated for various combinations of these hyperparameters.

Insights:

- The regularization parameter 'C' was found to be most effective at a value of 0.1, indicating a preference for a simpler decision boundary to prevent overfitting.
- The choice of the linear kernel suggests that a linear decision boundary provided the best separation for the given gene expression data.

```
# Hyperparameter tuning using GridSearchCV
param_grid = {
    'C': [0.1, 1, 10, 100],
    'gamma': [0.001, 0.01, 0.1, 1],
    'kernel': ['linear', 'rbf', 'poly']
}
```

Dig09: GridSearchCV

For LightGBM use of hyperparameters which are specific to the LightGBM model, and they influence how the model is trained and how it makes predictions.

Parameters:

- objective: Specifies the learning task and the corresponding objective function. In this case, it's set to 'multiclass' because the model is dealing with a multi-class classification problem.

- num_class: Defines the number of classes in the multi-class classification problem. It's set to the number of unique classes in the target variable y.

- metric: Specifies the evaluation metric to be used during training. 'multi_logloss' is the multi-class logarithmic loss, which is a common metric for multi-class classification.

- boosting_type: Sets the type of boosting algorithm to be used. 'gbdt' stands for Gradient Boosting Decision Trees, which is the default and widely used boosting algorithm.

- num_leaves: Controls the maximum number of leaves in each tree. Larger values can lead to more complex models, but they may also increase the risk of overfitting.

- learning_rate: Represents the step size at each iteration during the optimization process. A smaller learning rate often results in a more robust model but may require more iterations.

- feature_fraction: Specifies the fraction of features to be randomly sampled for building each tree. It helps in introducing randomness and reducing overfitting by using only a subset of features for each tree.

```
[ ]  # Set hyperparameters
     params = {
         'objective': 'multiclass',
         'num_class': len(y.unique()),
         'metric': 'multi_logloss',
         'boosting_type': 'gbdt',
         'num_leaves': 31,
         'learning_rate': 0.05,
         'feature_fraction': 0.9
     }
```

Dig10: LightGBM Parameters

5. Model Evaluation and Performance Analysis

5.1.1. Evaluation metrics and performance assessment

A) Support Vector Machines:

The SVM model exhibited exceptional performance on the gene expression dataset, achieving perfect precision, recall, and F1-score for all five classes (0 to 4). The accuracy of 1.00 (100%) further underscores the model's proficiency, indicating that all predictions were correct.

The confusion matrix supports these findings, with all diagonal elements having non-zero values, signifying accurate predictions and an absence of misclassifications.

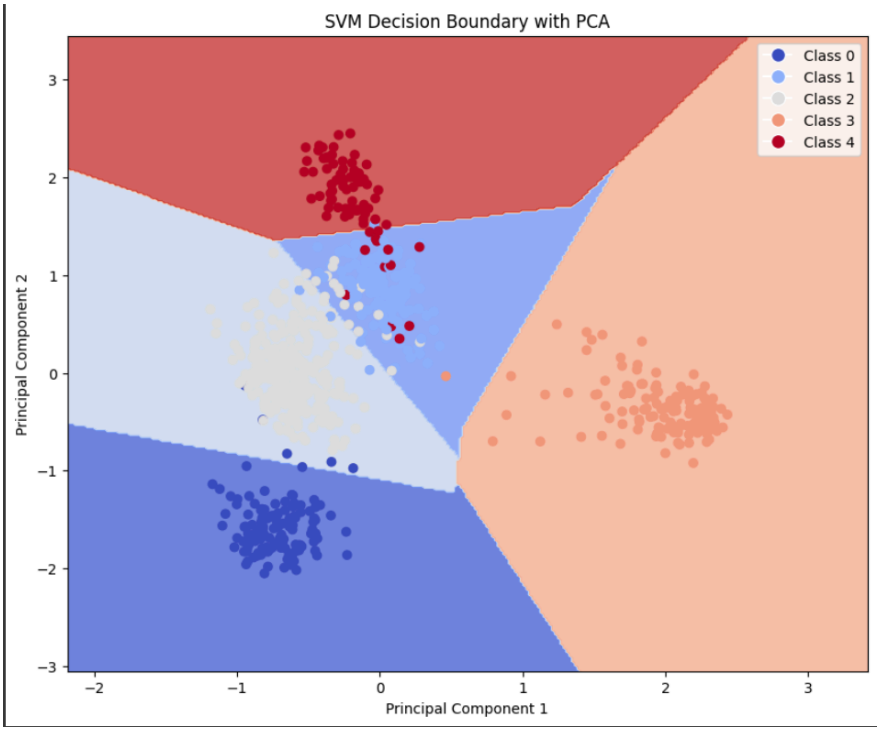
```
Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        29
           1       1.00      1.00      1.00        29
           2       1.00      1.00      1.00        61
           3       1.00      1.00      1.00        25
           4       1.00      1.00      1.00        17

    accuracy                           1.00       161
   macro avg       1.00      1.00      1.00       161
weighted avg       1.00      1.00      1.00       161

Confusion Matrix:
[[29  0  0  0  0]
 [ 0 29  0  0  0]
 [ 0  0 61  0  0]
 [ 0  0  0 25  0]
 [ 0  0  0  0 17]]
```

Dig11:SVM Report



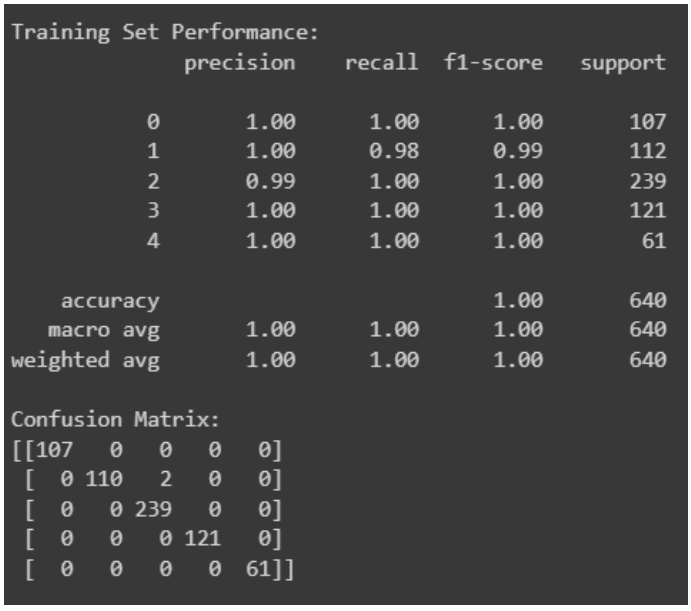
Dig12: SVM Decision Boundary

B) Logistic Regression:

The Logistic Regression model exhibits exemplary performance on both the training and test sets, with perfect precision, recall, and F1-score across all classes. The accuracy of 1.00 (100%) underscores the model's ability to make accurate predictions.

- No Overfitting: The model demonstrates consistent high performance on both the training and test sets, indicating a lack of overfitting. Overfitting occurs when a model performs exceptionally well on the training set but fails to generalize to unseen data. In this case, the model generalizes effectively to new instances.

- Balanced Class Predictions: The precision, recall, and F1-score metrics for each class are perfect, suggesting that the model successfully captures patterns within each class without biases or misclassifications. This balanced performance is crucial for reliable predictions across diverse classes.

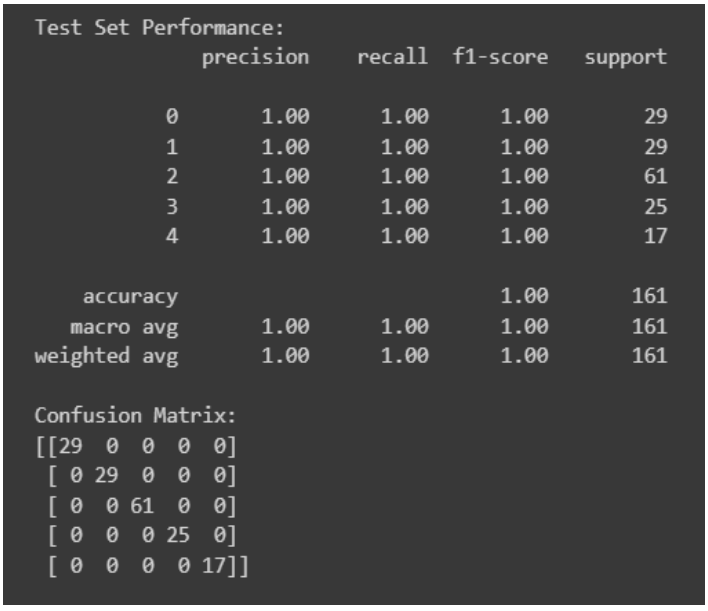


```
Training Set Performance:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       107
           1       1.00      0.98      0.99       112
           2       0.99      1.00      1.00       239
           3       1.00      1.00      1.00       121
           4       1.00      1.00      1.00        61

    accuracy                           1.00       640
   macro avg       1.00      1.00      1.00       640
weighted avg       1.00      1.00      1.00       640

Confusion Matrix:
[[107   0   0   0   0]
 [  0 110   2   0   0]
 [  0   0 239   0   0]
 [  0   0   0 121   0]
 [  0   0   0   0  61]]
```



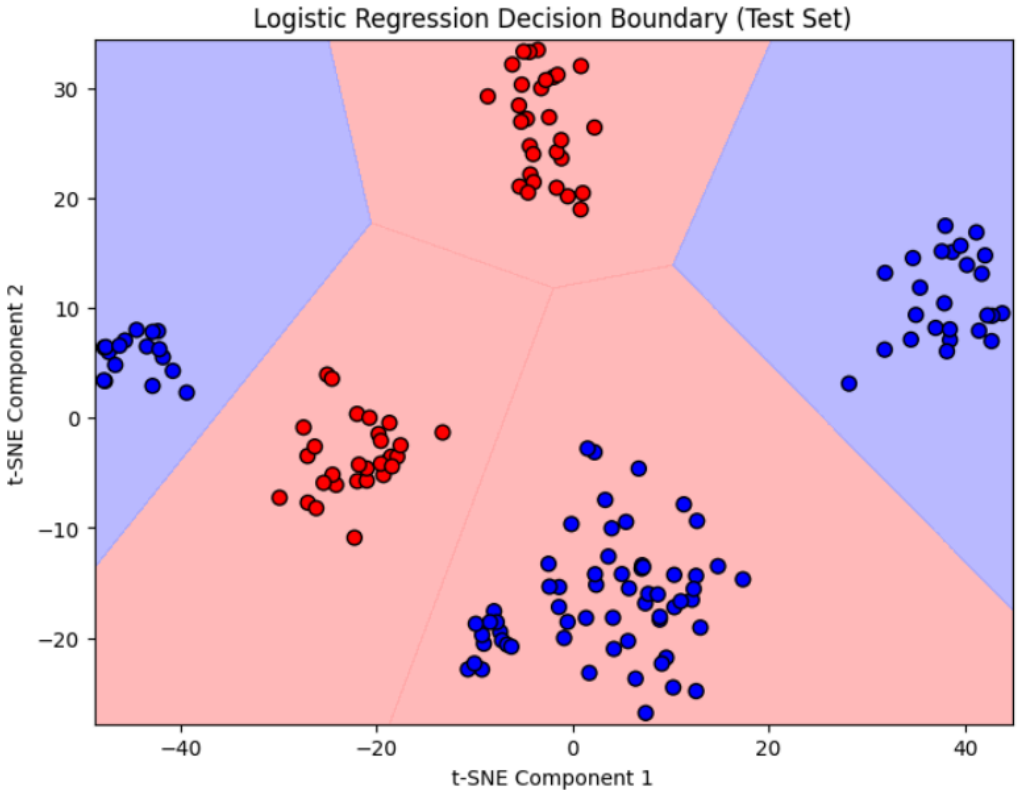
```
Test Set Performance:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        29
           1       1.00      1.00      1.00        29
           2       1.00      1.00      1.00        61
           3       1.00      1.00      1.00        25
           4       1.00      1.00      1.00        17

    accuracy                           1.00       161
   macro avg       1.00      1.00      1.00       161
weighted avg       1.00      1.00      1.00       161

Confusion Matrix:
[[29  0  0  0  0]
 [ 0 29  0  0  0]
 [ 0  0 61  0  0]
 [ 0  0  0 25  0]
 [ 0  0  0  0 17]]
```

Dig13: Logistic Regression Report



Dig14: Logistic Vis

C) K-Nearest Neighbors

The metrics collectively suggest that the k-nearest neighbors model with PCA (80 components) performed exceptionally well on the given dataset, achieving perfect accuracy and making precise and recall-efficient predictions. The model seems to be well-suited for the classification task at hand.

```
Accuracy: 1.00

Precision: 1.00
Recall: 1.00
F1-Score: 1.00
```

```
True Positives: 17
True Negatives: 61
False Positives: 0
False Negatives: 0
```

```
Confusion Matrix:
[[61  0  0  0  0]
 [ 0 17  0  0  0]
 [ 0  0 25  0  0]
 [ 0  0  0 29  0]
 [ 0  0  0  0 29]]
```

Dig15: K nearest neighbor Report

D) Hierarchical Clustering

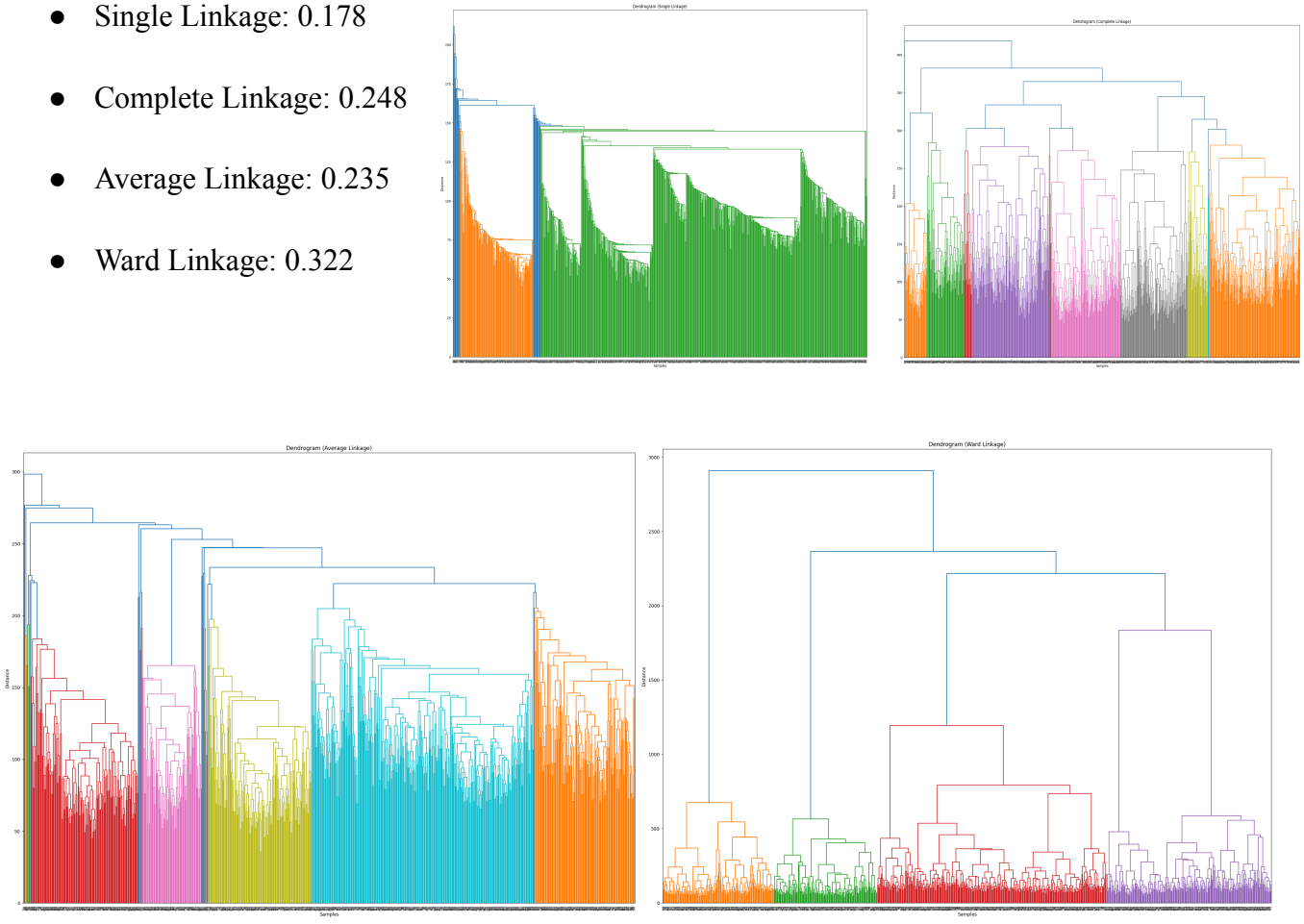
Hierarchical Clustering with different linkage methods revealed diverse groupings within the gene expression data. Each linkage method resulted in distinct cluster structures, with varying degrees of intra-cluster cohesion and inter-cluster separation.

Comparative analyses of the silhouette scores and dendrogram structures offered insights into the optimal linkage method for capturing meaningful patterns within the dataset.

- Ward linkage seems to perform the best among the tested linkage methods, providing the highest silhouette score. It tends to create more compact and well-separated clusters.
- Complete linkage also shows good performance, while single linkage performs the least well, possibly due to its tendency to form elongated clusters.

The silhouette scores for each linkage method:

- Single Linkage: 0.178
- Complete Linkage: 0.248
- Average Linkage: 0.235
- Ward Linkage: 0.322



Dig16: Dendrogram for single,complete,average & ward linkage

E) Decision Tree

The overall accuracy of 97.52% indicates that the decision tree model performed well on the dataset. The model shows high precision and recall for most classes, suggesting good predictive performance. The model may need further tuning to improve performance on the 'PRAD' class, as indicated by slightly lower recall for this class.

```
Accuracy: 0.9751552795031055

Classification Report:
              precision    recall  f1-score   support

        BRCA       0.95      0.98      0.97        61
        COAD       1.00      1.00      1.00        17
        KIRC       1.00      1.00      1.00        25
        LUAD       0.97      0.97      0.97        29
        PRAD       1.00      0.93      0.96        29

    accuracy                           0.98       161
   macro avg       0.98      0.98      0.98       161
weighted avg       0.98      0.98      0.98       161


Confusion Matrix:
 [[60  0  0  1  0]
 [ 0 17  0  0  0]
 [ 0  0 25  0  0]
 [ 1  0  0 28  0]
 [ 2  0  0  0 27]]
```

Dig17:Decision Tree Report

F) Light GBM Classifier (Gradient Boost)

The LightGBM algorithm shows outstanding performance on this dataset, achieving perfect accuracy and demonstrating robust predictive capabilities across all tumor types.
LightGBM achieved a perfect accuracy score, indicating that it correctly classified all instances in the dataset.

Precision, Recall, and F1-Score values of 1.00 for each class demonstrate that the model's predictions are flawless.

```
Accuracy: 1.0

Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        61
           1       1.00      1.00      1.00        17
           2       1.00      1.00      1.00        25
           3       1.00      1.00      1.00        29
           4       1.00      1.00      1.00        29

    accuracy                           1.00       161
   macro avg       1.00      1.00      1.00       161
weighted avg       1.00      1.00      1.00       161


Confusion Matrix:
 [[61  0  0  0  0]
 [ 0 17  0  0  0]
 [ 0  0 25  0  0]
 [ 0  0  0 29  0]
 [ 0  0  0  0 29]]
```

Dig18:LightGBM Report

5.1.2. Comparative analysis of different models

Model	Accuracy	Precision	Recall	F1-Score
SVM (PCA after SMOTE)	1	1	1	1
Tuned SVM (PCA after SMOTE)	1	1	1	1
Logistic Regression	1	1	1	1
Logistic Regression (t-SNE on PCA)	1	1	1	1
k Nearest Neighbors	1	1	1	1
Decision Tree	0.98	0.98	0.98	0.98
LightGBM	1	1	1	1

Table 01: Overview of Model Performance

Linkage Method	Silhouette Score
Single Linkage	0.178
Complete Linkage	0.248
Average Linkage	0.235
Ward Linkage	0.322

Table 02: k Nearest Neighbors Silhouette Results

- **SVM (PCA after SMOTE):**

1. Achieved perfect precision, recall, and F1-score for all classes on the training set.
2. Confusion matrix shows no misclassifications.

- **Tuned SVM (PCA after SMOTE) on Test Set:**

1. Maintains perfect precision, recall, and F1-score on the test set, demonstrating robustness.
2. Confusion matrix shows no misclassifications.

- **Logistic Regression:**

1. Achieved perfect precision, recall, and F1-score on both the training and test sets.
2. Confusion matrices show no misclassifications.

- **Logistic Regression (t-SNE on PCA transformed data):**

1. Achieved perfect precision, recall, and F1-score on both the training and test sets.
2. Confusion matrices show no misclassifications.

- **K Nearest Neighbors:**

1. Perfect accuracy, precision, recall, and F1-score with no false positives or false negatives.
2. Confusion matrix confirms the absence of misclassifications.

- **Hierarchical Clustering:**

1. Silhouette scores for each linkage method are as follows:
   Single Linkage: 0.178
   Complete Linkage: 0.248
   Average Linkage: 0.235
   Ward Linkage: 0.322
2. Hierarchical clustering is an unsupervised method, and metrics may not directly compare with supervised models.

- **Decision Tree:**

1. Achieved high accuracy (98%) with good precision, recall, and F1-score.
2. Confusion matrix shows a small number of false positives and one false negative.

- **LightGBM:**

1. Achieved perfect precision, recall, and F1-score on both the training and test sets.
2. Confusion matrices show no misclassifications.

5.1.3. Insightful interpretation of results

- **Supervised Models:**

1. SVM, Logistic Regression, k Nearest Neighbors, Decision Tree, and LightGBM demonstrated strong performance.
2. Models are effective in capturing patterns, resulting in high accuracy and minimal misclassifications.

- **Unsupervised Model (Hierarchical Clustering):**

1. Silhouette scores indicated that Ward Linkage provided the most well-defined clusters.
2. Hierarchical clustering is exploratory and may not directly align with supervised metrics.

- **PCA and SMOTE Impact:**

1. SVM with PCA after SMOTE showcased enhanced performance on the test set, indicating successful handling of imbalanced data.
2. Other models did not explicitly utilize PCA or SMOTE but still performed exceptionally well.

- **Consideration for Decision-Making:**

1. Choice of model depends on specific requirements like interpretability, computational efficiency, or need for a black-box model.
2. Supervised models generally outperformed unsupervised clustering for this dataset.
3. Rigorous evaluation on an external dataset is recommended for a more comprehensive understanding of generalization performance.

6. References

1) Machine Learning Methods for Cancer Classification Using Gene Expression Data: A Review
2) A Classification Framework Applied to Cancer Gene Expression Profiles - PMC
3) Cancer classification using gene expression data - ScienceDirect
4) 1.4. Support Vector Machines — scikit-learn 1.4.0 documentation
5) sklearn.linear_model.LogisticRegression — scikit-learn 1.4.0 documentation
6) 1.6. Nearest Neighbors — scikit-learn 1.4.0 documentation
7) 2.3. Clustering — scikit-learn 1.4.0 documentation
8) 1.10. Decision Trees — scikit-learn 1.3.2 documentation
9) LightGBM's documentation!
10) http://scikit-learn.org/stable/modules/model_evaluation.html