ISyE 6740 - Spring 2021
Final Report

Team Member Names:	Saptarshi Basu
Project Title:	Multiclass tumor classification based on RNA-Seq Gene Expression Data

Problem Statement

It is sobering to consider than annually about 9.5 million people die due to cancer with the number being around 600,000 in the United States. A significant portion of these deaths can be prevented by accurate diagnosis of cancer at early stages. Current diagnosis is driven by imaging (ultrasound, CT-scan, MRI), biopsy, and blood tests. Recently, with the advent of cost efficient gene sequencing, RNA-seq arrays are increasingly used for cancer diagnosis. The hope is that it will eventually transfer to liquid biopsy technologies where small fragments of DNA in the blood can be sequenced to identify cancer even before onset based on DNA mutations. While, liquid biopsy is still a work in development, gene expression data is being increasingly used to analyze tumor cells to identify tumor type or detect cancer. Gene expression is the way the genetic information encoded in DNA is first transcripted to RNA and then translated to a protein. The protein controls the cell functioning. Hence, the presence of different proteins as shown in the gene expression panel can be indicative of cancer type or modified cell functioning. As the DNA mutates, the genetic information is modified, resulting in a differential gene expression or protein translation.

The present work focuses on analyzing the gene expression data from multiple samples to classify the tumor type. A total of 801 samples are analyzed and five tumor types (PRAD, LUAD, BRCA, COAD, and KIRC) were studied. The objective is to accurately predict the tumor type based on available gene expression data. The problem is multi class classification and the independent feature set is high-dimensional (20531). Thus, the major challenge is feature selection. A variety of methods: f_classif, mutual_info_classif, PCA, and LDA is used for feature selection and/or dimension reduction. Next, a wide spectrum of classification algorithms was tested to compare the predictive performance. The classification methods considered in the study are as follows:

- Non-parametric: k-NN
- Bayesian: LDA, Naive Bayes
- Support Vector Machine: Linear and RBF Kernel SVM
- Generative: Logistic Regression (Elastic Net)
- Tree based: Decision Tree

- Bagging: Random Forest
- Boosting: AdaBoost

Data Source
The data was obtained from UCI Machine Learning Repository: UCI machinelearning

Samples (instances) are stored row-wise. Variables (attributes) of each sample are RNA-Seq gene expression levels measured by Illumina HiSeq platform. A dummy name (gene_XX) is given to each attribute. Total of 801 samples, 20531 attributes, and five tumor classes are included in the dataset.

Methodology
The high-level analysis workflow in Python is outlined below..
D**ata Preprocessing**

1. Perform data cleanup and impute missing data, if needed. (Numpy, Pandas). No missing data identified.
2. Perform exploratory data analysis: heatmaps, 2D plots, multicolinearity assessed later. (numpy, matplotlib, Seaborn,Statsmodel)
3. Determine class distribution and identify the key classification metrics
4. Standardize each feature using sklearn.preprocessing.StandardScaler

Dimensional Reduction

1. Identify key features using ANOVA F-test (f_classif) and Mutual Info approach (mutual_info_classif).
2. Reduce the high dimensional feature set using PCA and LDA
3. Plot reduced dimension heat maps, scree plots, and 2D clustering.
4. Different classification methods will be tested using the reduced or projected dimensions.
5. Number of features and the embedding/feature selection will be determined by classification performance.

Classification Methods

1. Split the data into 80/20 training and test split (sklearn.model_selection.train_test_split)
2. Cross-validation to be used on training set to tune hyperparameters including regularization parameters.
3. Bayes classifiers: Linear Discriminant Analysis (sklearn.discriminant_analysis.LinearDiscriminantAnalysis), Naive Bayes (sklearn.naive_bayes.GaussianNB)
4. Non-parametric methods like KNN Classifier. The number of neighbors to be determined using cross-validation (sklearn.neighbors.KNeighborsClassifier)
5. Logistic Regression elastic net. (sklearn.linear_model.LogisticRegression)
6. Support Vector Machine: Linear and RBF Kernel (sklearn.svm.SVC)
7. Evaluate decision tree based methods. Tune hyperparameters to prevent overfitting. (sklearn.tree.DecisionTreeClassifier)

8. Evaluate different ensemble methods based on bagging and boosting (Random Forest and AdaBoost)

Evaluation

1. Test each individual classifier on the test set
2. Plot the confusion matrix for each classifier
3. Measure the classification metrics (accuracy, precision, recall, and F1 score) to identify the best classifier.
4. Visualize the data in 2D embeddings and plot decision boundary
5. Make final recommendations

Exploratory Data Analysis and Feature Selection
The class distribution of the dependent variable or tumor label is as follows:

Table 1: Tumor Label Class Distribution

Tumor Label	No. of samples	Ordinal Label
BRCA	300	0
KIRC	146	2
LUAD	141	3
PRAD	136	4
COAD	78	1
Total	801	

Since, the classes are fairly well distributed; accuracy can be used as the classification metric during cross-validation. For further analysis, precision, recall, and F1-score will be utilized. The high-dimensional features were standardized and then represented in the heatmap. As shown in Fig. 1 , significant differentiation is not observed due to the "curse of dimensionality". A number of feature selection methods will be utilized to reduce the dimension by either projection (PCA/LDA) or feature selection.

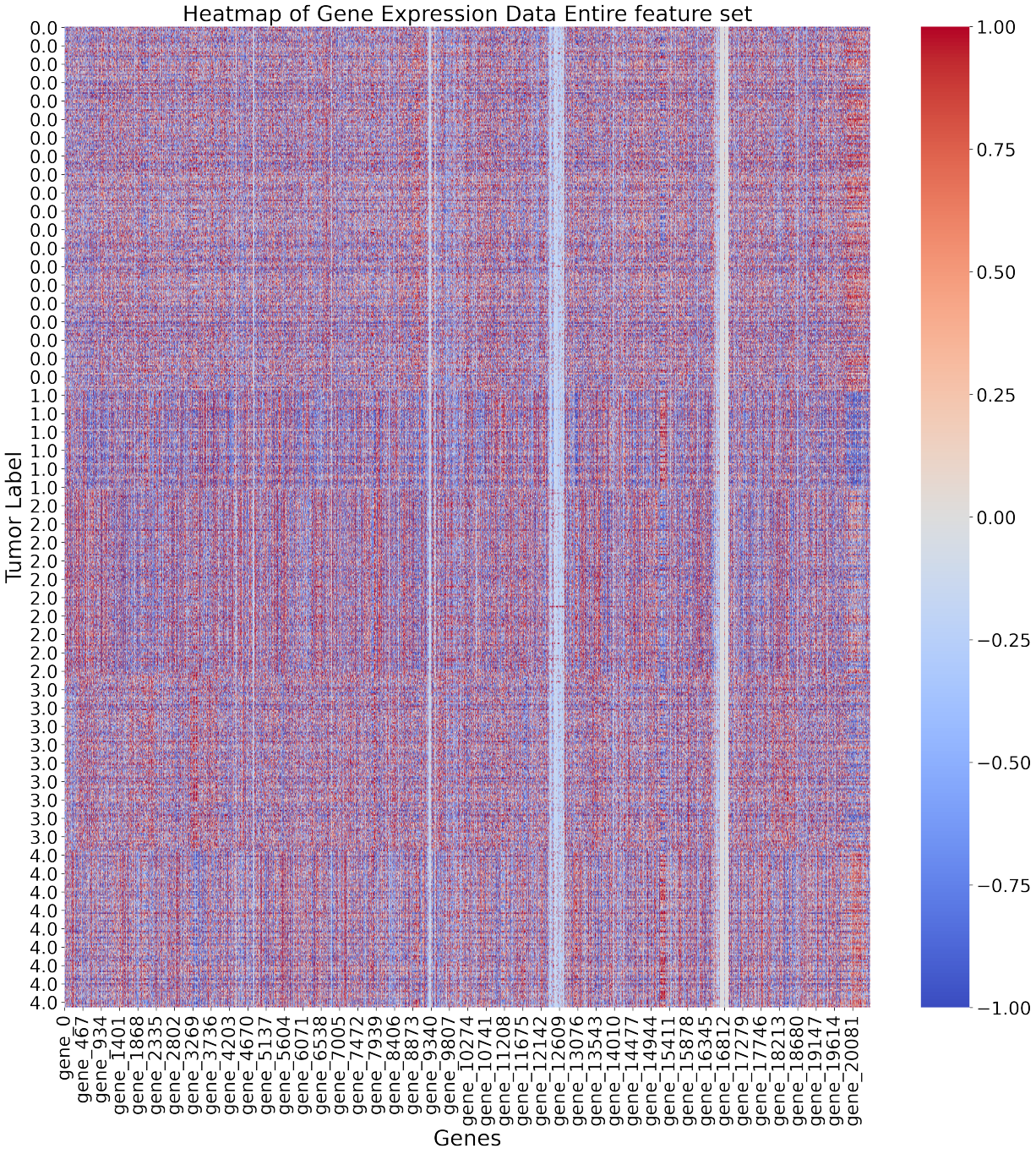


Figure 1: Heatmap of Gene Expression Data (20531 features)

The different feature selection and dimensional reduction methods are discussed next. Although a comprehensive analysis of the number of features was done, only the final results are presented for the feature selection section. Interested readers can refer to the provided Jupyter notebook for the alternate analysis and plots.

- **ANOVA F Feature Selection**
  f_classif was used with SelectKBest package from sklearn to select the top 5 and top 20 features. Based on performance testing with the classification algorithms, the top 20 features were used in the model as the accuracy was higher than with 5 features without exhibiting overfit. The 20 feature heatmap is shown in Fig. 2 . Significant differentiation is observed among the different tumor labels for the selected features. Different combinations of genes are expressed more in different tumor types. The x labels provided the name of the selected features. However, when f_classif was used to plot the data in 2D, the different clusters were not well separated (Fig. 3)

- **Mutual Info Feature Selection**
  For feature selection based on non-linear relationship, mutual_info_classif was used with SelectKBest. In this case,top 20 features were considered for classification.. Different combination of genes was differentially expressed in the different tumor classes (Fig. 4). The 2D plot also showed clustering of the tumor classes (Fig. 5)

- **PCA**
  PCA is a popular linear dimension reduction method based on maximizing variance across the projected principal components. However, PCA may not be suited for the current dataset with more than 20000 features as the top principal components do not account for a large proportion of the total variance. In our testing, the top 300 eigenvalues accounted for about 86% of the variance. This is evident from the scree plot (Fig. 6) with the top 20 components accounting for 53% of the variance. The top 20 principal components were chosen based on some margin on the elbow in the variance plot around 10 principal components. This was further tested and verified on different classification algorithms.

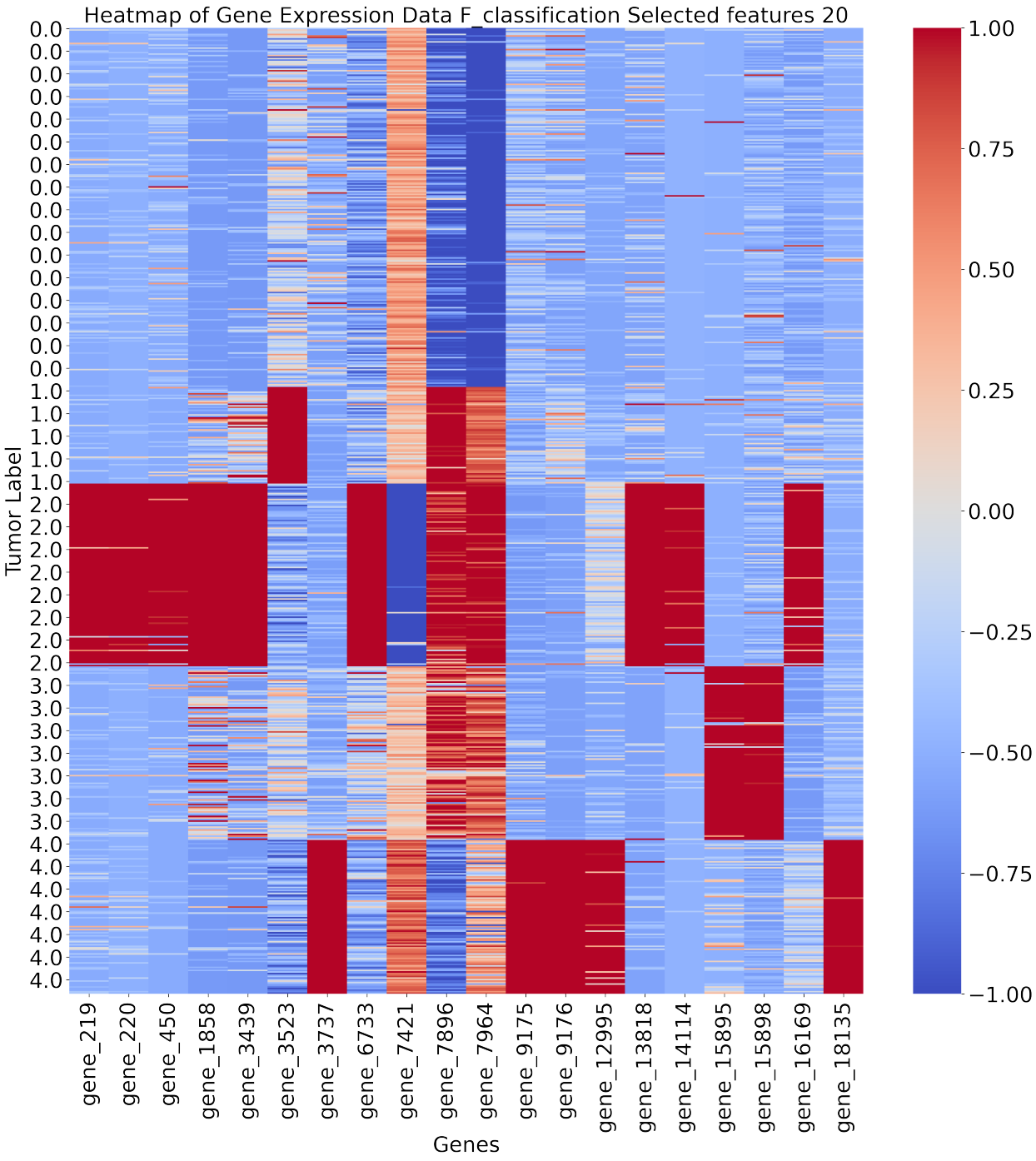


Figure 2: Selected top 20 genes (f_classif)

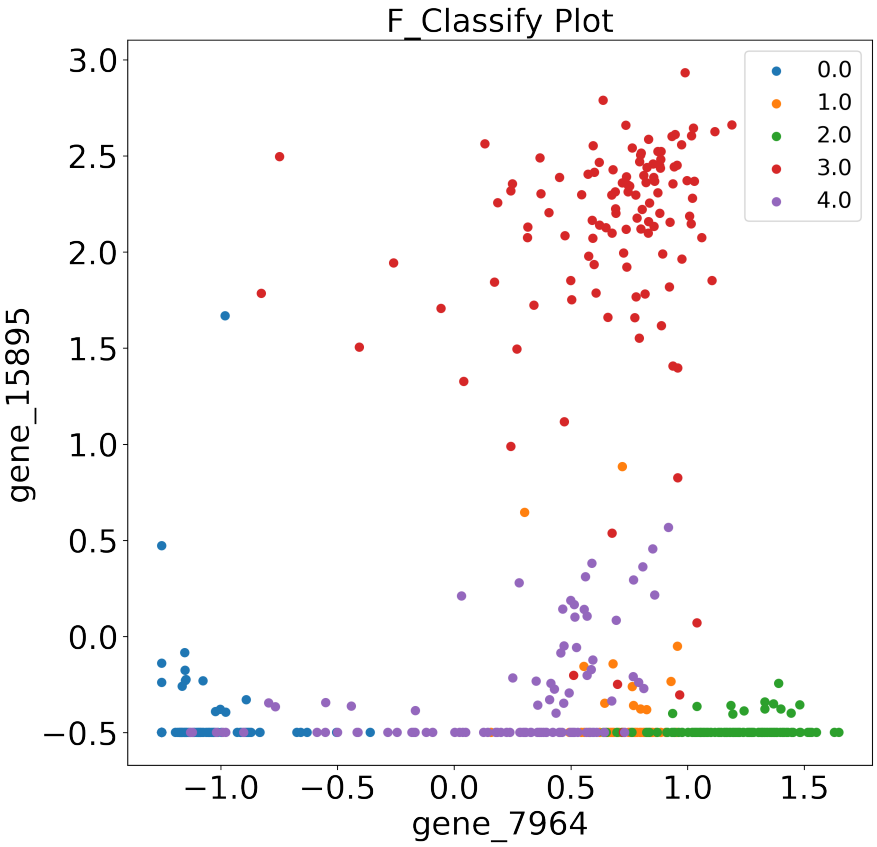


Figure 3: F_classif 2D plot

- **LDA**
  In order to address some of the limitations of the PCA with a high-dimension feature set, Linear Discriminant Analysis was used to project the original feature set to 4 dimensions. This is based on the idea of separating the classes as evident in the 2D embeddings (Fig. 8). LDA shows five separate clusters corresponding to the five different classes although classes 1 and 3 are not well differentiated in two dimensions. The corresponding explained variance plot for LDA is available in Jupyter notebook.

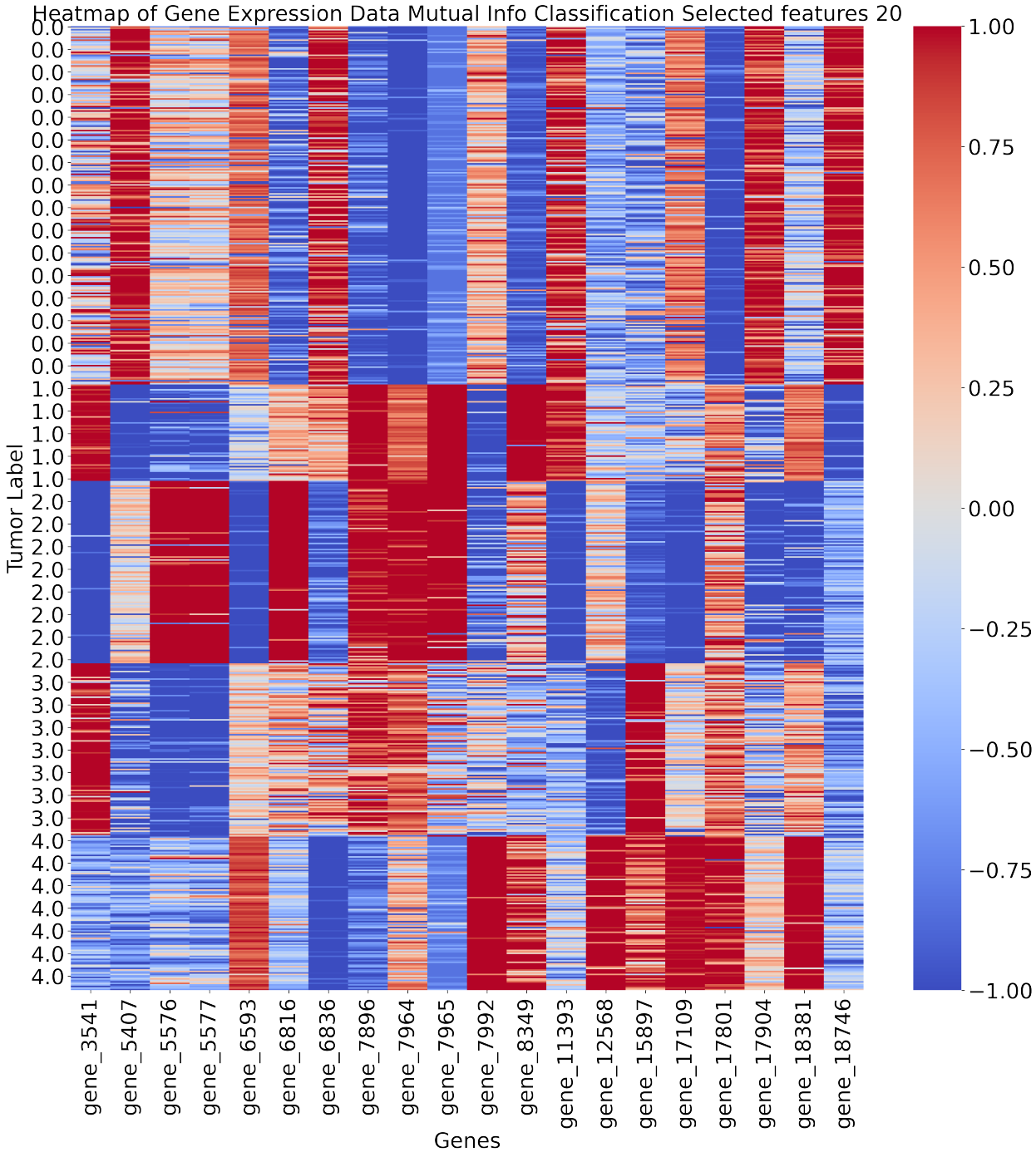


Figure 4: Selected top twenty genes (mutual_info_classif)

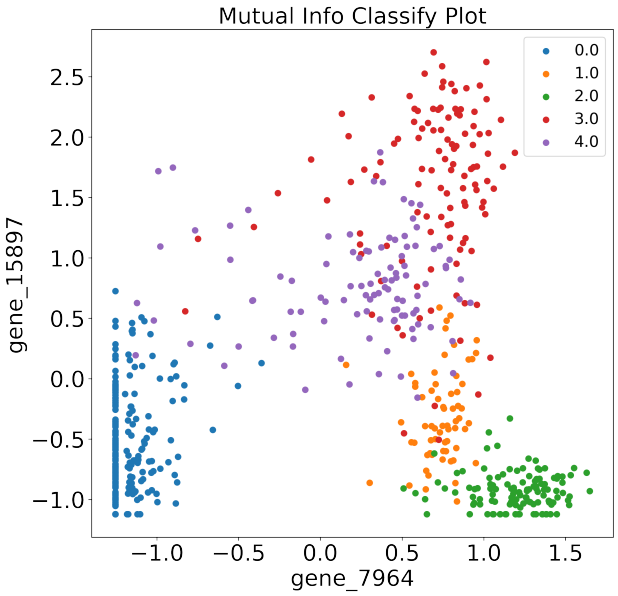


Figure 5: Mutual_info_classif 2D plot

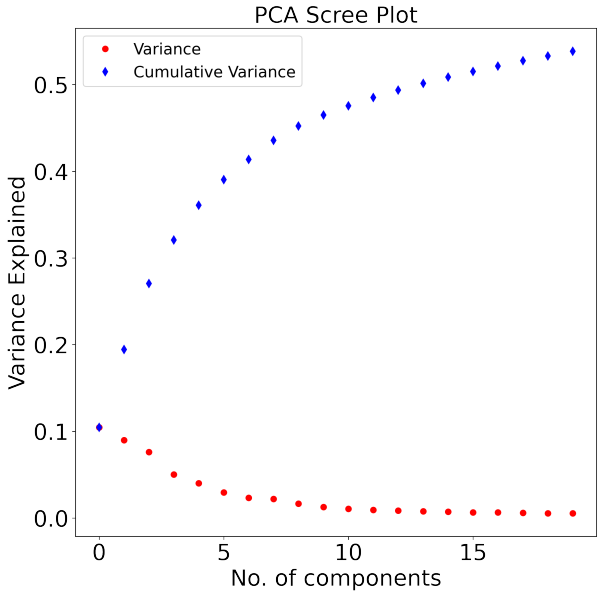


Figure 6: PCA Scree Plot

The 2D plot for PCA does not separate the classes very well. The principal components maximize variance but it is unsupervised and does not distinguish between the class labels.

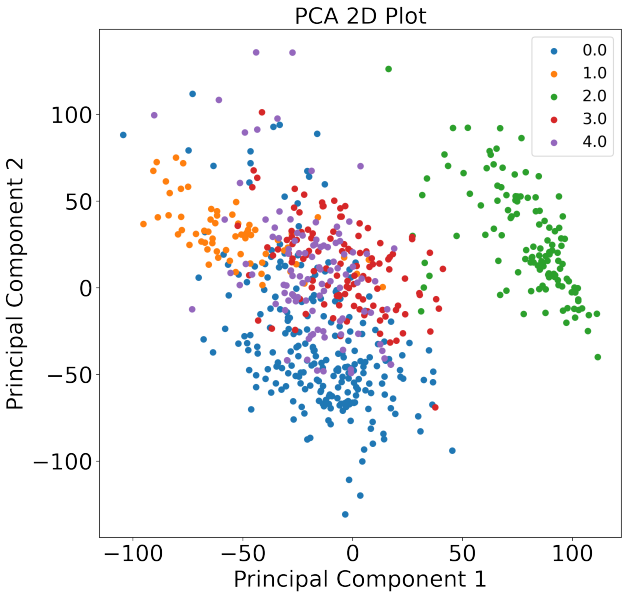


Figure 7: PCA 2D plot

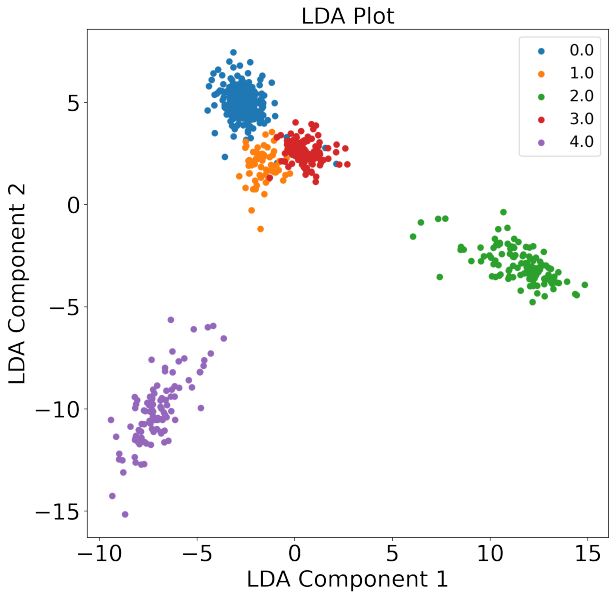


Figure 8: LDA 2D plot

Notes:
Standard Scaler was used on each of the independent parameters, hence the variance is same

across all the independent variables. As seen in Table 3 , there is significant multicollinearity between the independent features. Only, PCA addresses the multicollinearity between the attributes. The PCA principal components by design are orthogonal to each other and hence, not correlated. The distribution of the selected features based on different feature selection/sample reduction algorithms are shown in Fig. 9 . Based on visual inspection, only the PCA principal components have a close to normal distribution. The original features selected by f_classif and mutual_info_classif as well as the LDA decomposition variables either showed a bimodal or a skewed distribution. **Therefore, only the PCA features were used in the LDA and Naive Bayes classifiers**.

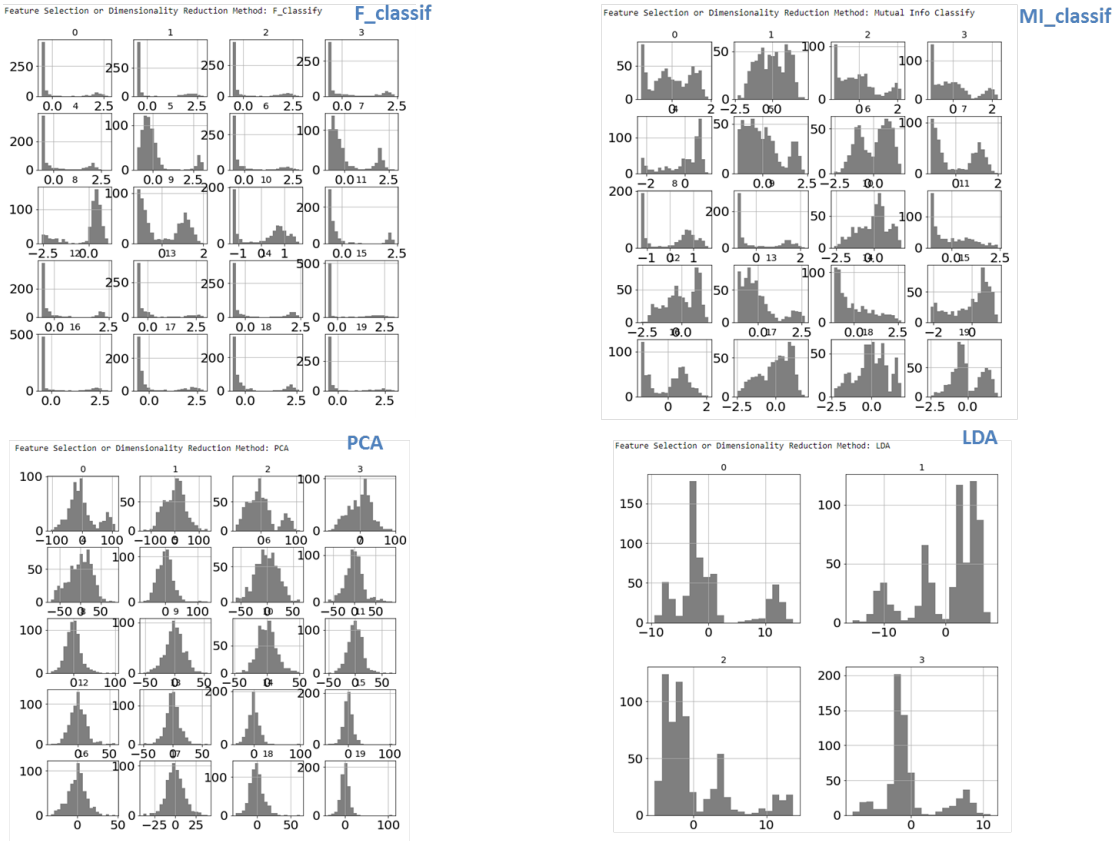


Figure 9: Distribution of Features/Projected Dimensions

Classification Models: Hyperparameter Tuning and Performance

A multitude of classification algorithms were tested on the data set to determine the predictive performance of the same. The data set was split into a 80-20 training and test set randomly. Thereafter, the hyperparameters were tuned for each algorithm using a 5 fold cross validation

with accuracy as metric. This tuning was done on all the four feature sets selected: f_classif, mutual_info_classif. PCA, and LDA. Finally, the test set performance was measured using accuracy, precision, recall, and F1-score. The confusion matrix was also determined for each of the selected feature sets. Since, multiple selected feature sets were compared, the optimum hyperparameter is approximated based on overall visual inspection rather than tuned for each individual feature set. The analysis and performance summary for the different algorithms are discussed as follows:

1. **K- Nearest Neighbor Classifier**
   K-nn classifier was implemented with number of neighbors as the hyperparameter. The cross-validation score is shown in Fig. 10. Based on the CV-score plot, k=5 was selected as the optimum number of neighbors.

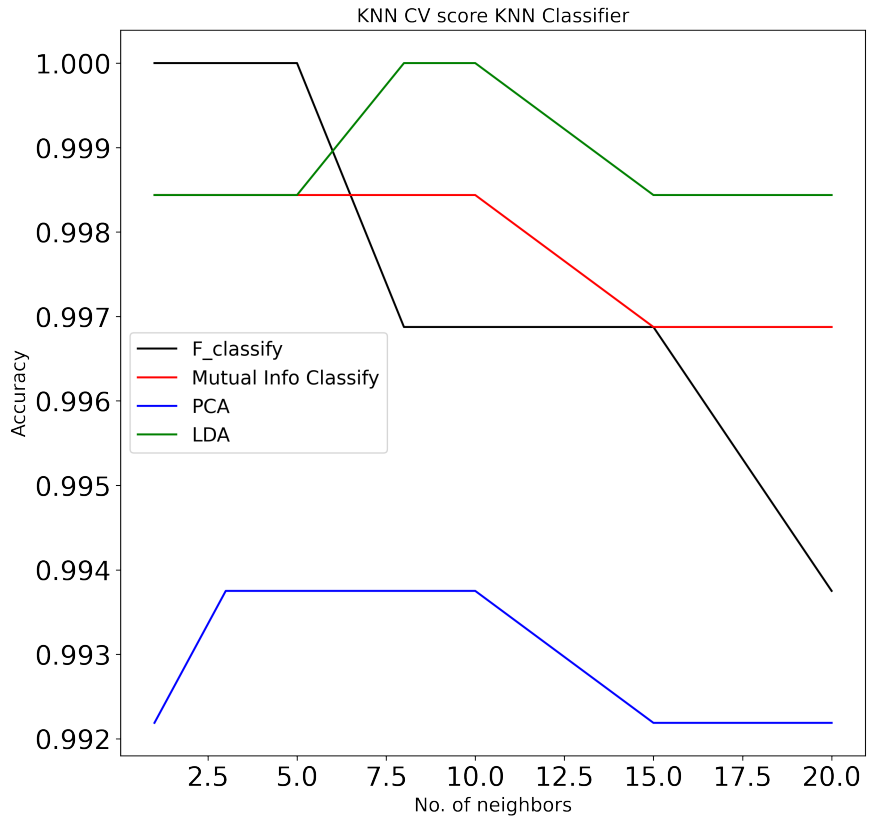


Figure 10: K-NN Classifier CV Score Plot

The test set performance with k-nn classifier is summarized in Figs. 11 and 12.

Model:k-nn with f_classify,Accuracy:1.0
Model: k-nn with f_classify

	precision	recall	f1-score	support
0	1.00	1.00	1.00	62
1	1.00	1.00	1.00	14
2	1.00	1.00	1.00	25
3	1.00	1.00	1.00	26
4	1.00	1.00	1.00	34
accuracy			1.00	161
macro avg	1.00	1.00	1.00	161
weighted avg	1.00	1.00	1.00	161

a) F_classif

Model:k-nn with mutual info classify,Accuracy:0.9937888198757764
Model: k-nn with mutual info classify

	precision	recall	f1-score	support
0	0.98	1.00	0.99	62
1	1.00	1.00	1.00	14
2	1.00	1.00	1.00	25
3	1.00	0.96	0.98	26
4	1.00	1.00	1.00	34
accuracy			0.99	161
macro avg	1.00	0.99	0.99	161
weighted avg	0.99	0.99	0.99	161

b) MI_classif

Model:k-nn with PCA,Accuracy:0.9875776397515528
Model: k-nn with PCA

	precision	recall	f1-score	support
0	0.97	1.00	0.98	62
1	1.00	1.00	1.00	14
2	1.00	0.96	0.98	25
3	1.00	0.96	0.98	26
4	1.00	1.00	1.00	34
accuracy			0.99	161
macro avg	0.99	0.98	0.99	161
weighted avg	0.99	0.99	0.99	161

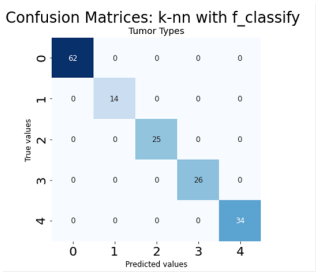
c) PCA

Model:k-nn with LDA,Accuracy:0.9813664596273292
Model: k-nn with LDA

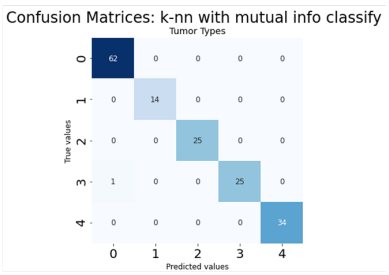
	precision	recall	f1-score	support
0	1.00	0.98	0.99	62
1	0.93	1.00	0.97	14
2	1.00	0.96	0.98	25
3	1.00	0.96	0.98	26
4	0.94	1.00	0.97	34
accuracy			0.98	161
macro avg	0.98	0.98	0.98	161
weighted avg	0.98	0.98	0.98	161

d) LDA

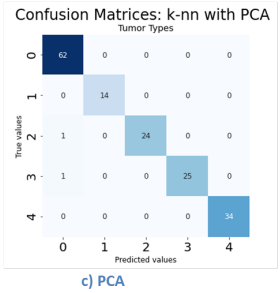
Figure 11: k-nn Classification Report (Test set)



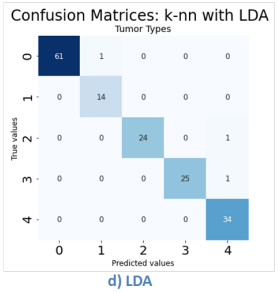
a) F_classif



b) MI_classif



c) PCA



d) LDA

Figure 12: k-nn Confusion Matrix (Test set)

2. **Naive Bayes Classifier**
   The Naive Bayes classifier was implemented with variance smoothing as the hyperpa-

rameter. **Naive Bayes classifier assumes independence of features and normal distribution. Thus, it is strictly applicable only for the PCA projected data set.** Nevertheless, although the assumptions are not strictly followed for the other selected feature sets, the classifier was still tested with the other selected feature sets - f_classif, mutual_info_classif, and LDA. Based on the cross-validation score shown in Fig. 13 , Naive Bayes model was implemented with variance smoothing factor of 0.1.

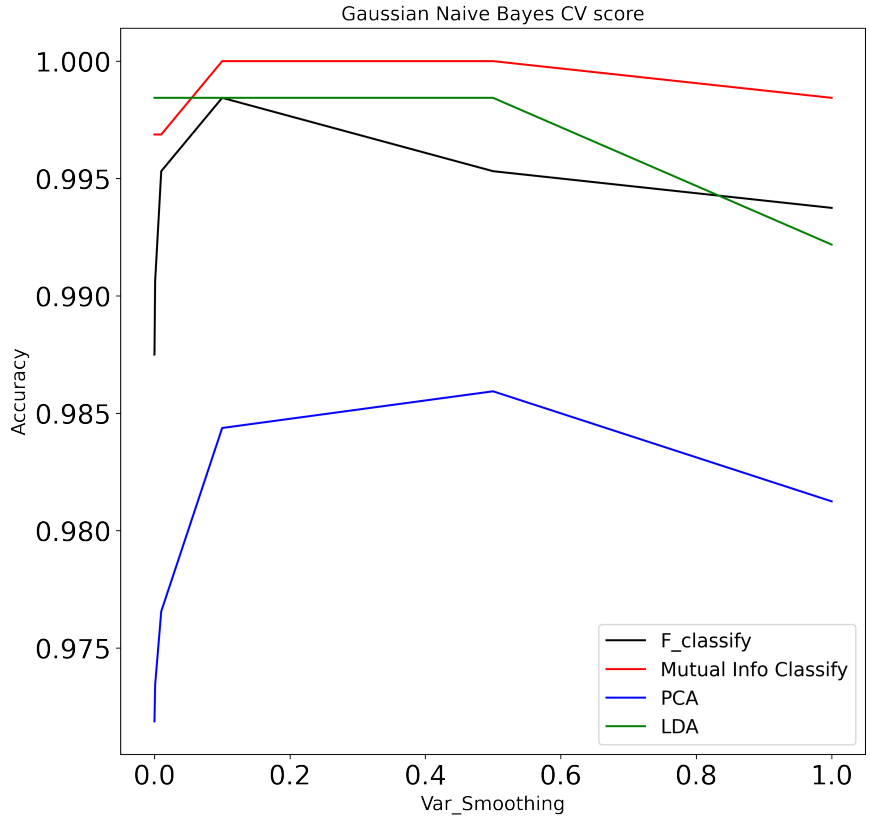


Figure 13: Gaussian Naive Bayes Classifier CV Score Plot

The test set performance is summarized in the classification report and confusion matrix in Figs. 14 and 15 . All four feature sets exhibit excellent accuracy with small variations in precision and recall in some classes. The LDA which uses only four projected features performed slightly poorer compared to the other sets which consist of 20 features. Due to the assumptions, it is recommended that Naive Bayes is only utilized with PCA data set and caution should be used for the other cases.

Model:GaussNB with f_classify,Accuracy:1.0
Model: GaussNB with f_classify

	precision	recall	f1-score	support
0	1.00	1.00	1.00	62
1	1.00	1.00	1.00	14
2	1.00	1.00	1.00	25
3	1.00	1.00	1.00	26
4	1.00	1.00	1.00	34
accuracy			1.00	161
macro avg	1.00	1.00	1.00	161
weighted avg	1.00	1.00	1.00	161

a) F_classif

Model:GaussNB with mutual info classify,Accuracy:0.9937888198757764
Model: GaussNB with mutual info classify

	precision	recall	f1-score	support
0	0.98	1.00	0.99	62
1	1.00	1.00	1.00	14
2	1.00	1.00	1.00	25
3	1.00	0.96	0.98	26
4	1.00	1.00	1.00	34
accuracy			0.99	161
macro avg	1.00	0.99	0.99	161
weighted avg	0.99	0.99	0.99	161

b) MI_classif

Model:GaussNB with PCA,Accuracy:0.9875776397515528
Model: GaussNB with PCA

	precision	recall	f1-score	support
0	0.97	1.00	0.98	62
1	1.00	1.00	1.00	14
2	1.00	1.00	1.00	25
3	1.00	0.96	0.98	26
4	1.00	0.97	0.99	34
accuracy			0.99	161
macro avg	0.99	0.99	0.99	161
weighted avg	0.99	0.99	0.99	161

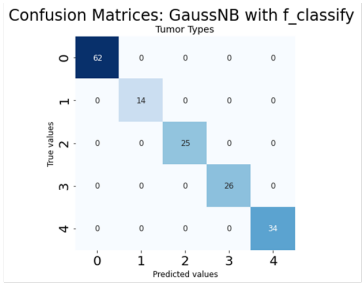
c) PCA

Model:GaussNB with LDA,Accuracy:0.9751552795031055
Model: GaussNB with LDA

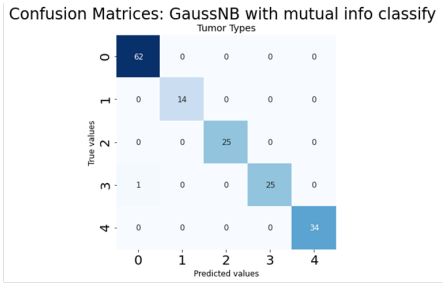
	precision	recall	f1-score	support
0	1.00	0.98	0.99	62
1	0.88	1.00	0.93	14
2	1.00	0.96	0.98	25
3	1.00	0.96	0.98	26
4	0.94	0.97	0.96	34
accuracy			0.98	161
macro avg	0.96	0.98	0.97	161
weighted avg	0.98	0.98	0.98	161

d) LDA

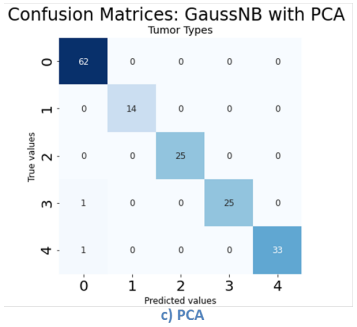
Figure 14: Gaussian Naive Bayes Classification Report (Test set)



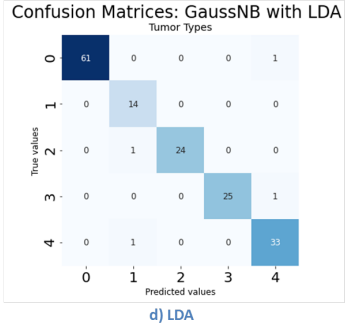
a) F_classif



b) MI_classif



c) PCA



d) LDA

Figure 15: Gaussian Naive Bayes Confusion Matrix (Test set)

3. **Linear Discriminant Analysis (LDA)**

Similar to Naive Bayes, LDA assumptions are independence of the features, same variance, and normal distribution. These constraints are met only for the PCA principal components. Although, the LDA classification algorithm is tested with the other feature sets, caution should be applied in those circumstances. LDA was applied with default variables, and the cross-validation performance was as follows:

Table 2: LDA Cross Validation Score

f_classif (20)	MI_classif (20)	PCA (20)	LDA (4)
0.99	1.00	0.99	1.0

The test set performance is captured in classification report and confusion matrix (Figs. 16 and 17). Similar to the other algorithms, the accuracy was >0.99 for all the four feature sets.

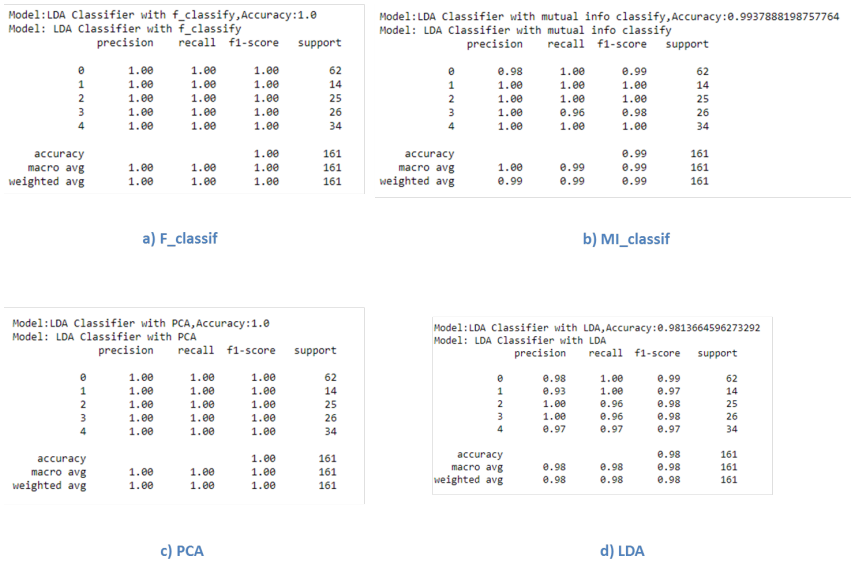


Figure 16: LDA Classification Report (Test set)

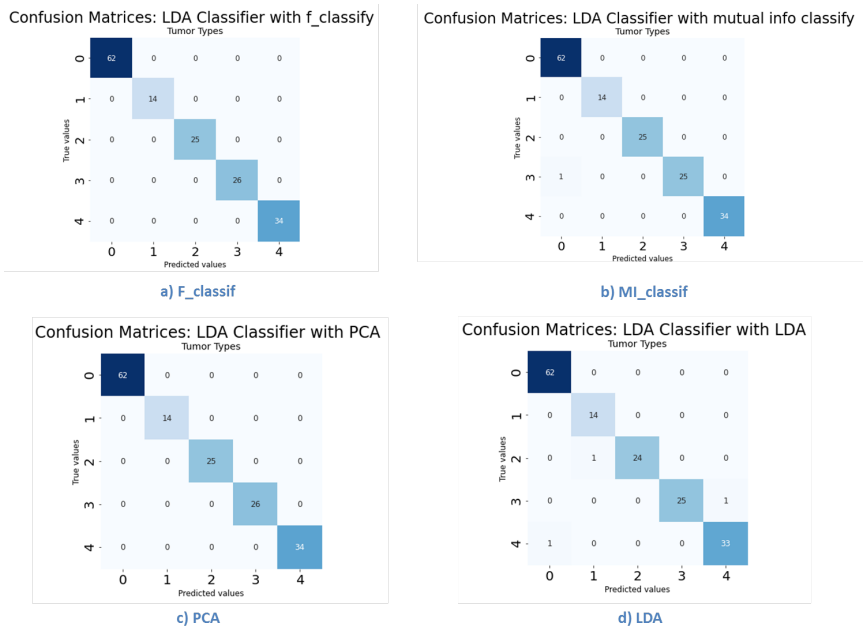


Figure 17: LDA Confusion Matrix (Test set)

4. **Logistic Regression**
   Logistic Regression was implemented with elastic net. The hyperparameters included l1 ratio and inverse of the regularization parameter. Based on the cross-validation plot shown in Fig. 18 ,l1_ratio = 0.5 and C =1 was selected as optimum hyperparameter. The saga solver was used with multinomial classification as this is a multi-class problem. The fitted model was then tested on the test set and the performance is summarized in Figs. 19 and 20 .

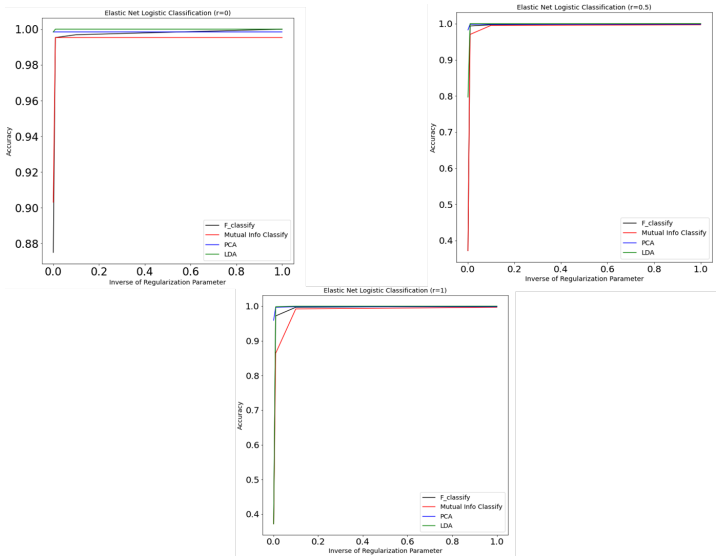


Figure 18: Logistic Regression Classifier CV Score Plot

Model:Logistic Regression with f_classify,Accuracy:0.9937888198757764
Model: Logistic Regression with f_classify

	precision	recall	f1-score	support
0	0.98	1.00	0.99	62
1	1.00	1.00	1.00	14
2	1.00	1.00	1.00	25
3	1.00	1.00	1.00	26
4	1.00	0.97	0.99	34
accuracy			0.99	161
macro avg	1.00	0.99	1.00	161
weighted avg	0.99	0.99	0.99	161

a) F_classif

Model:Logistic Regression with mutual info classify,Accuracy:0.9937888198757764
Model: Logistic Regression with mutual info classify

	precision	recall	f1-score	support
0	0.98	1.00	0.99	62
1	1.00	1.00	1.00	14
2	1.00	1.00	1.00	25
3	1.00	0.96	0.98	26
4	1.00	1.00	1.00	34
accuracy			0.99	161
macro avg	1.00	0.99	0.99	161
weighted avg	0.99	0.99	0.99	161

b) MI_classif

Model:Logistic Regression with PCA,Accuracy:1.0
Model: Logistic Regression with PCA

	precision	recall	f1-score	support
0	1.00	1.00	1.00	62
1	1.00	1.00	1.00	14
2	1.00	1.00	1.00	25
3	1.00	1.00	1.00	26
4	1.00	1.00	1.00	34
accuracy			1.00	161
macro avg	1.00	1.00	1.00	161
weighted avg	1.00	1.00	1.00	161

c) PCA

Model:Logistic Regression with LDA,Accuracy:0.9813664596273292
Model: Logistic Regression with LDA

	precision	recall	f1-score	support
0	0.98	1.00	0.99	62
1	0.93	1.00	0.97	14
2	1.00	0.96	0.98	25
3	0.96	1.00	0.98	26
4	1.00	0.94	0.97	34
accuracy			0.98	161
macro avg	0.98	0.98	0.98	161
weighted avg	0.98	0.98	0.98	161

d) LDA

Figure 19: Logistic Regression Classification Report (Test set)

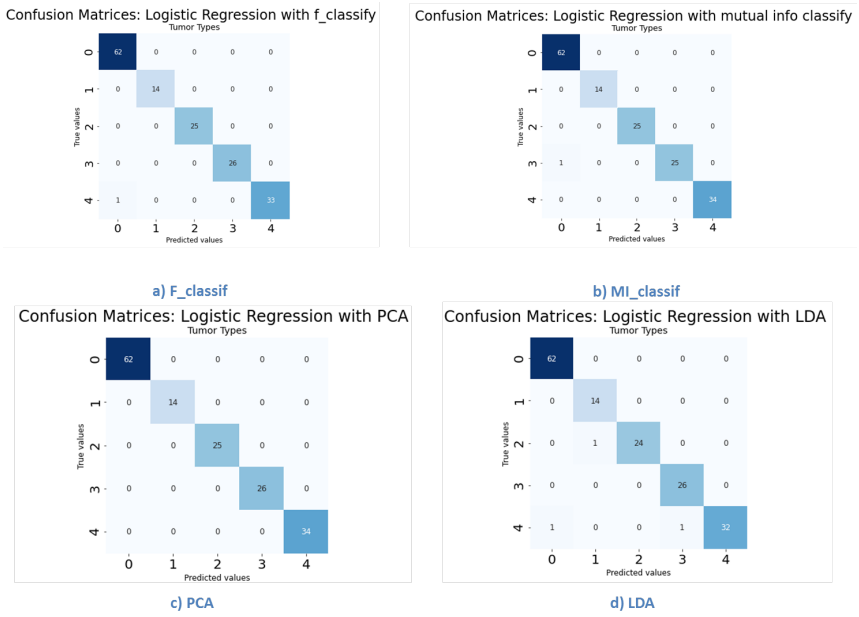


Figure 20: Logistic Regression Confusion Matrix (Test set)

Table 3: Logisitc Regression Coefficients and VIF for f_classif feature sets

Genes	Features	Label 0	Label 1	Label 2	Label 3	Label 4	VIF
0	gene_219	0	0	0.24	-0.03	0	87.69
1	gene_220	0	0	0.25	0	0	100.55
2	gene_450	0	0	0.30	-0.04	0	19.55
3	gene_1858	0	0	0.69	0	0	12.03
4	gene_3439	-0.12	0	0.58	0	0	11.93
5	gene_3523	0	2.01	-0.02	-0.57	0	4.19
6	gene_3737	-0.17	0	0	0	0.85	10.91
7	gene_6733	0	0	0.65	0	0	13.34
8	gene_7421	0	0	-0.20	0	0	13.03
9	gene_7896	-1.18	0.44	0	0.25	0	9.99
10	gene_7964	-2.00	0.09	0	0.39	0	11.18
11	gene_9175	-0.04	0	0	0	0.87	34.50
12	gene_9176	-0.15	0	0	-0.09	0.82	31.70
13	gene_12995	-0.26	0	0	0	0.55	10.83
14	gene_13818	0	0	0.51	-0.07	0	15.32
15	gene_14114	0	0	0.15	0	0	18.06
16	gene_15895	-0.39	0	0	1.18	0	13.79
17	gene_15898	-0.34	0	0	1.31	0	15.28
18	gene_16169	0	0	0.27	-0.03	0	18.38

The coefficients of the logistic regression model for the f_classif dataset are shown in Table 3. The vif factors indicate that there is considerable amount of multicollinearity between the attributes as it is expected that the gene expressions are dependent on each other. Using elastic net, the coefficients of several of these attributes are reduced to zero.

5. **SVM Linear**
   SVM Linear was fitted with the inverse of regularization parameter as the only hyper-parameter. A value of 0.1 was chosen based on the cv plot shown in Fig. 21. One vs. rest was used for decision function shape. The test set performance is shown in Figs. 22 and 23 .

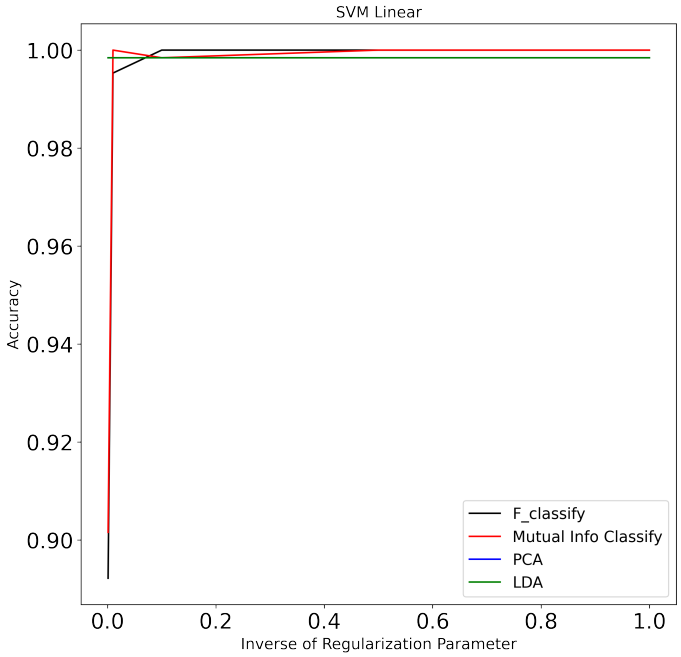


Figure 21: SVM Linear Classifier CV Score Plot

Model:SVM Linear with f_classify,Accuracy:1.0
Model: SVM Linear with f_classify

	precision	recall	f1-score	support
0	1.00	1.00	1.00	62
1	1.00	1.00	1.00	14
2	1.00	1.00	1.00	25
3	1.00	1.00	1.00	26
4	1.00	1.00	1.00	34
accuracy			1.00	161
macro avg	1.00	1.00	1.00	161
weighted avg	1.00	1.00	1.00	161

a) F_classif

Model:SVM Linear with mutual info classify,Accuracy:0.9937888198757764
Model: SVM Linear with mutual info classify

	precision	recall	f1-score	support
0	0.98	1.00	0.99	62
1	1.00	1.00	1.00	14
2	1.00	1.00	1.00	25
3	1.00	0.96	0.98	26
4	1.00	1.00	1.00	34
accuracy			0.99	161
macro avg	1.00	0.99	0.99	161
weighted avg	0.99	0.99	0.99	161

b) MI_classif

Model:SVM Linear with PCA,Accuracy:1.0
Model: SVM Linear with PCA

	precision	recall	f1-score	support
0	1.00	1.00	1.00	62
1	1.00	1.00	1.00	14
2	1.00	1.00	1.00	25
3	1.00	1.00	1.00	26
4	1.00	1.00	1.00	34
accuracy			1.00	161
macro avg	1.00	1.00	1.00	161
weighted avg	1.00	1.00	1.00	161

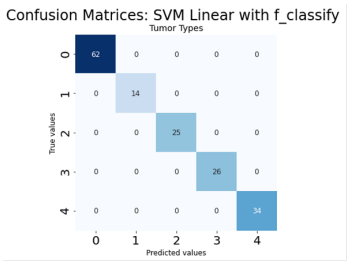
c) PCA

Model:SVM Linear with LDA,Accuracy:0.9813664596273292
Model: SVM Linear with LDA

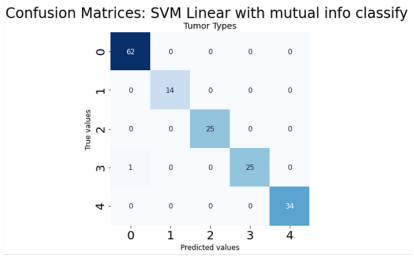
	precision	recall	f1-score	support
0	0.98	1.00	0.99	62
1	0.93	1.00	0.97	14
2	1.00	0.96	0.98	25
3	1.00	0.96	0.98	26
4	0.97	0.97	0.97	34
accuracy			0.98	161
macro avg	0.98	0.98	0.98	161
weighted avg	0.98	0.98	0.98	161

d) LDA

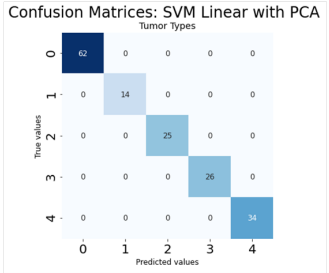
Figure 22: SVM Linear Classification Report (test set)



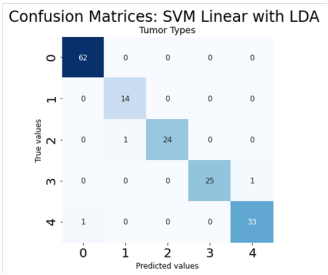
a) F_classif



b) MI_classif



c) PCA



d) LDA

Figure 23: SVM Linear Confusion Matrix (test set)

6. **SVM RBF Kernel**

The kernelized version of Support Vector Machine was implemented with RBF kernel

and hyperparameters: inverse of regularization parameter and gama. Scale, auto, and selected floats are used for gama. Some selected plots for cross validation score are shown in Fig. 24 . Based on the plots, C=0.1 and scale version of gamma are used in the final model that is tested on the test set (Figs. 25 and 26).

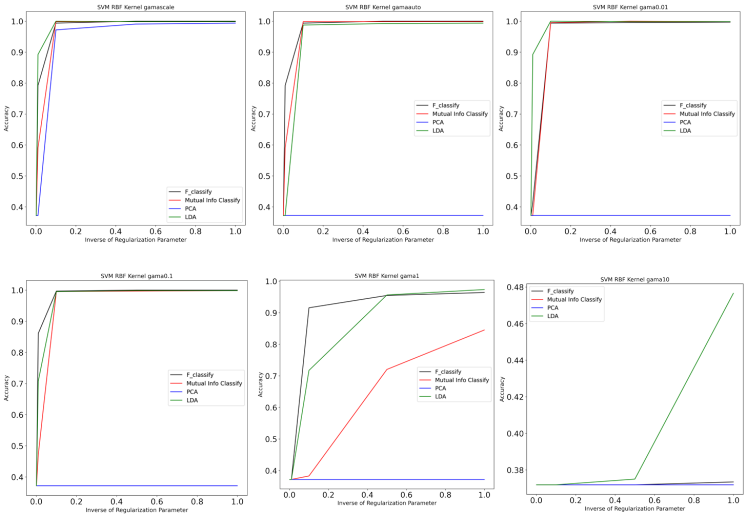


Figure 24: SVM RBF Kernel Classifier CV Score Plot

Model:SVM RBF Kernel with f_classify,Accuracy:1.0
Model: SVM RBF Kernel with f_classify

	precision	recall	f1-score	support
0	1.00	1.00	1.00	62
1	1.00	1.00	1.00	14
2	1.00	1.00	1.00	25
3	1.00	1.00	1.00	26
4	1.00	1.00	1.00	34
accuracy			1.00	161
macro avg	1.00	1.00	1.00	161
weighted avg	1.00	1.00	1.00	161

a) F_classif

Model:SVM RBF Kernel with mutual info classify,Accuracy:0.9937888198757764
Model: SVM RBF Kernel with mutual info classify

	precision	recall	f1-score	support
0	0.98	1.00	0.99	62
1	1.00	1.00	1.00	14
2	1.00	1.00	1.00	25
3	1.00	0.96	0.98	26
4	1.00	1.00	1.00	34
accuracy			0.99	161
macro avg	1.00	0.99	0.99	161
weighted avg	0.99	0.99	0.99	161

b) MI_classif

Model:SVM RBF Kernel with PCA,Accuracy:0.968944099378882
Model: SVM RBF Kernel with PCA

	precision	recall	f1-score	support
0	0.93	1.00	0.96	62
1	1.00	1.00	1.00	14
2	1.00	0.96	0.98	25
3	1.00	0.88	0.94	26
4	1.00	0.97	0.99	34
accuracy			0.97	161
macro avg	0.99	0.96	0.97	161
weighted avg	0.97	0.97	0.97	161

c) PCA

Model:SVM RBF Kernel with LDA,Accuracy:0.9751552795031055
Model: SVM RBF Kernel with LDA

	precision	recall	f1-score	support
0	1.00	0.98	0.99	62
1	1.00	1.00	1.00	14
2	1.00	0.96	0.98	25
3	0.87	1.00	0.93	26
4	1.00	0.94	0.97	34
accuracy			0.98	161
macro avg	0.97	0.98	0.97	161
weighted avg	0.98	0.98	0.98	161

d) LDA

Figure 25: SVM RBF Kernel Classification Report (test set)

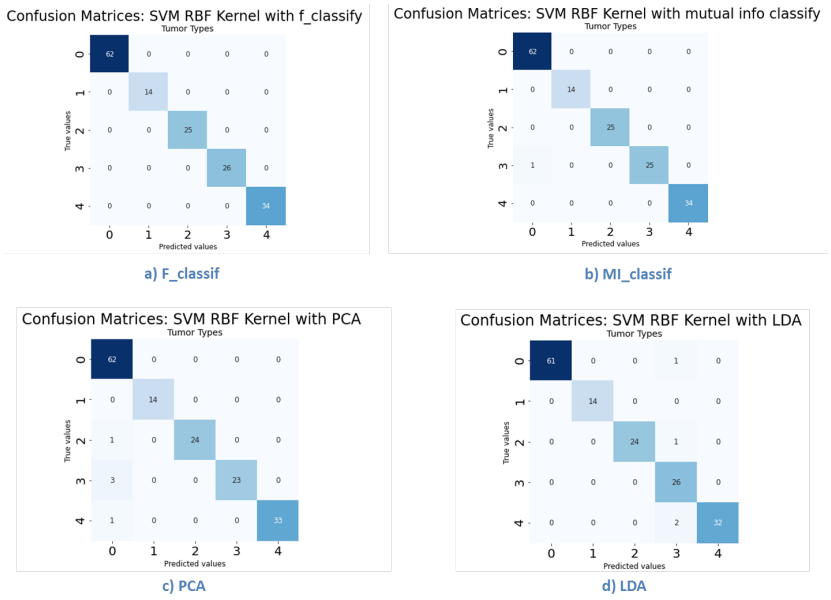


Figure 26: SVM RBF Kernel Confusion Matrix (test set)

7. **Decision Tree**

   A classification decision tree was next trained and cross-validation score was used to identify the optimum depth of 6 (Fig. 27). Max_Depth was the only hyperparameter considered in the present study. As the cross-validation score and later, test performance score was above 95%, pruning or additional hyperparameters were not needed. The actual depth of the tree was also quite small as shown in Fig. 28 . The test performance is summarized in Figs. 29 and 30 .

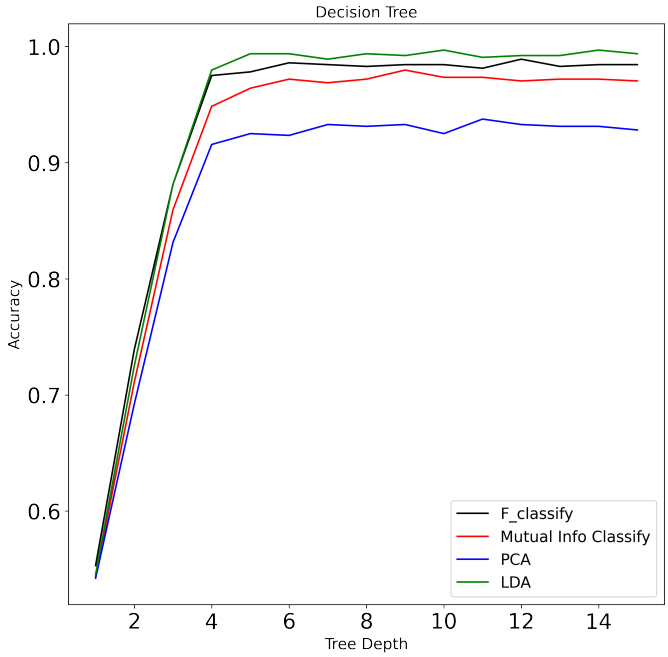


Figure 27: Decision Tree Classifier CV Score Plot

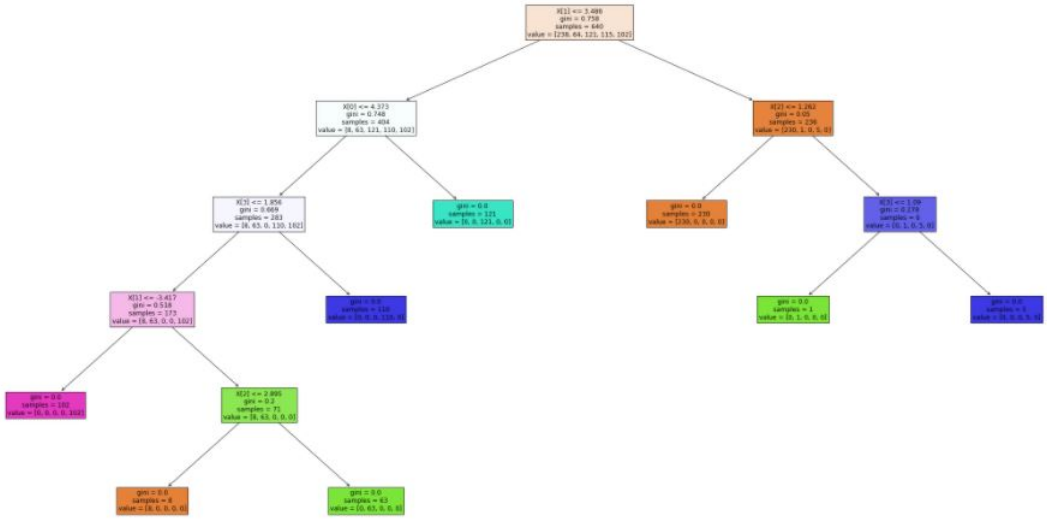


Figure 28: Decision Tree Plot

Model:Decision Tree with f_classify,Accuracy:0.9751552795031055
Model: Decision Tree with f_classify

	precision	recall	f1-score	support
0	0.98	1.00	0.99	62
1	1.00	1.00	1.00	14
2	1.00	0.88	0.94	25
3	0.89	0.96	0.93	26
4	1.00	1.00	1.00	34
accuracy			0.98	161
macro avg	0.98	0.97	0.97	161
weighted avg	0.98	0.98	0.98	161

a) F_classif

Model:Decision Tree with mutual info classify,Accuracy:0.9627329192546584
Model: Decision Tree with mutual info classify

	precision	recall	f1-score	support
0	0.93	1.00	0.96	62
1	1.00	1.00	1.00	14
2	1.00	0.96	0.98	25
3	1.00	0.92	0.96	26
4	0.97	0.91	0.94	34
accuracy			0.96	161
macro avg	0.98	0.96	0.97	161
weighted avg	0.96	0.96	0.96	161

b) MI_classif

Model:Decision Tree with PCA,Accuracy:0.9316770186335404
Model: Decision Tree with PCA

	precision	recall	f1-score	support
0	0.91	0.95	0.93	62
1	0.88	1.00	0.93	14
2	0.96	0.92	0.94	25
3	0.92	0.85	0.88	26
4	1.00	0.94	0.97	34
accuracy			0.93	161
macro avg	0.93	0.93	0.93	161
weighted avg	0.93	0.93	0.93	161

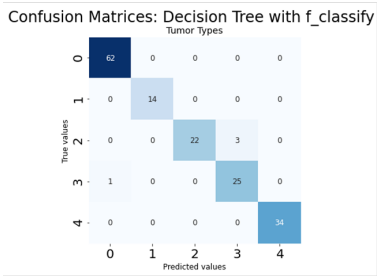
c) PCA

Model:Decision Tree with LDA,Accuracy:0.9751552795031055
Model: Decision Tree with LDA

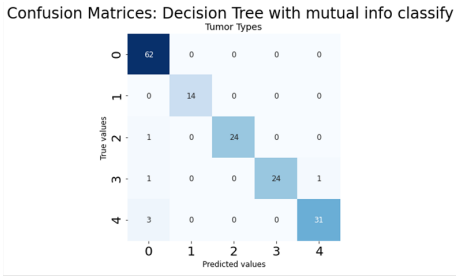
	precision	recall	f1-score	support
0	0.98	1.00	0.99	62
1	0.93	1.00	0.97	14
2	0.92	0.96	0.94	25
3	1.00	0.96	0.98	26
4	1.00	0.94	0.97	34
accuracy			0.98	161
macro avg	0.97	0.97	0.97	161
weighted avg	0.98	0.98	0.98	161

d) LDA

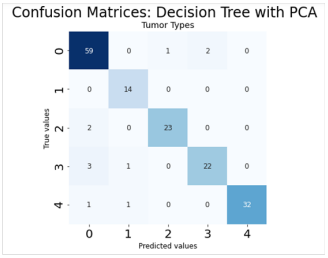
Figure 29: Decision Tree Classification Report (test set)



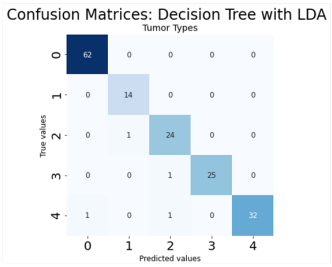
a) F_classif



b) MI_classif



c) PCA



d) LDA

Figure 30: Decision Tree Confusion Matrix (test set)

The feature importance as calculated by decision tree algorithm using f_classif feature set is provided in Fig. 31 .

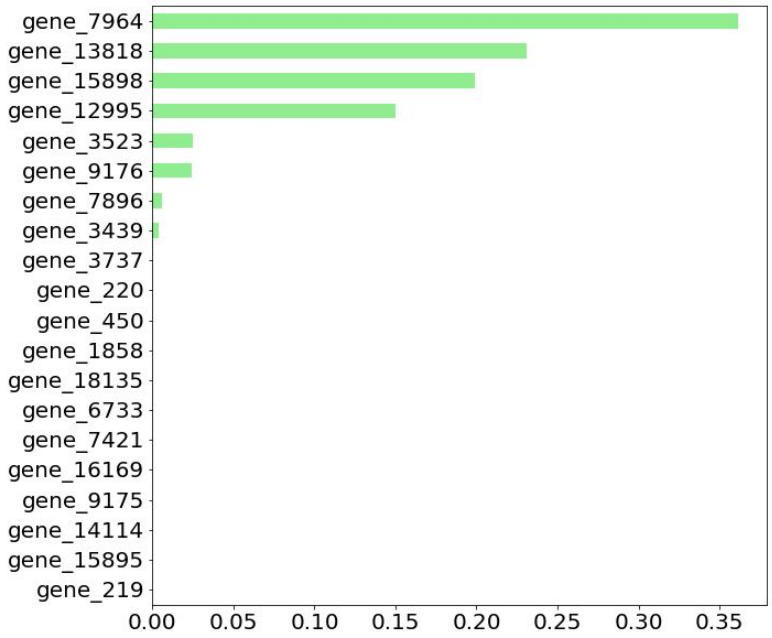


Figure 31: Decision Tree: Feature Importance

8. **Random Forest**
   Random Forest uses bagging by fitting a deep decision tree on a random set of samples and random set of features. The algorithm uses "gini" and "sqrt" of total features. Based on Fig. 32 , the number of estimators was determined to be 20. The test performance is shown in Fig. 33 and 34 . The feature importance for random forest algorithm for the f_classif feature set is shown in Fig. 35.

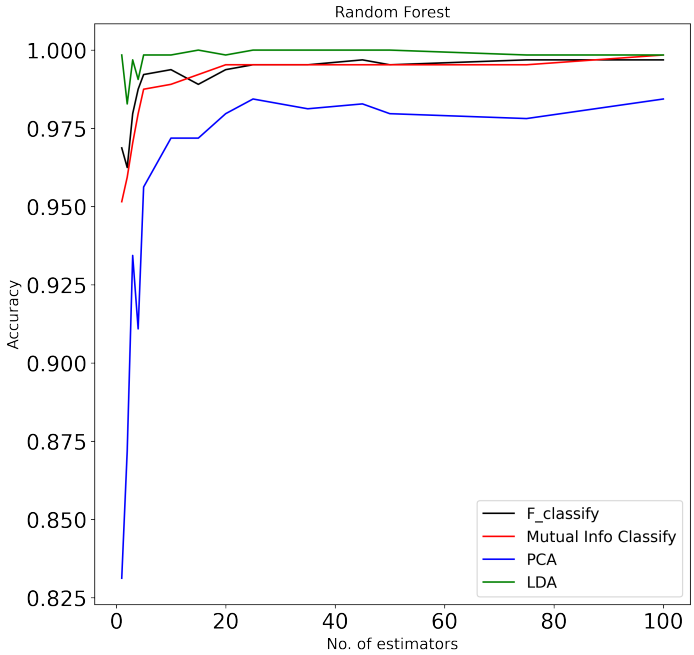


Figure 32: Random Forest Classifier CV Score Plot

Model:Random Forest with f_classify,Accuracy:0.9875776397515528
Model: Random Forest with f_classify

	precision	recall	f1-score	support
0	0.98	1.00	0.99	62
1	1.00	1.00	1.00	14
2	1.00	0.96	0.98	25
3	0.96	0.96	0.96	26
4	1.00	1.00	1.00	34
accuracy			0.99	161
macro avg	0.99	0.98	0.99	161
weighted avg	0.99	0.99	0.99	161

a) F_classif

Model:Random Forest with mutual info classify,Accuracy:0.9875776397515528
Model: Random Forest with mutual info classify

	precision	recall	f1-score	support
0	0.98	1.00	0.99	62
1	1.00	1.00	1.00	14
2	1.00	1.00	1.00	25
3	0.96	0.96	0.96	26
4	1.00	0.97	0.99	34
accuracy			0.99	161
macro avg	0.99	0.99	0.99	161
weighted avg	0.99	0.99	0.99	161

b) MI_classif

Model:Random Forest with PCA,Accuracy:0.968944099378882
Model: Random Forest with PCA

	precision	recall	f1-score	support
0	0.97	1.00	0.98	62
1	0.93	1.00	0.97	14
2	0.96	1.00	0.98	25
3	0.96	0.85	0.90	26
4	1.00	0.97	0.99	34
accuracy			0.97	161
macro avg	0.96	0.96	0.96	161
weighted avg	0.97	0.97	0.97	161

c) PCA

Model:Random Forest with LDA,Accuracy:0.9751552795031055
Model: Random Forest with LDA

	precision	recall	f1-score	support
0	0.97	1.00	0.98	62
1	1.00	1.00	1.00	14
2	0.92	0.96	0.94	25
3	1.00	0.96	0.98	26
4	1.00	0.94	0.97	34
accuracy			0.98	161
macro avg	0.98	0.97	0.98	161
weighted avg	0.98	0.98	0.98	161

d) LDA

Figure 33: Random Forest Classification Report (test set)

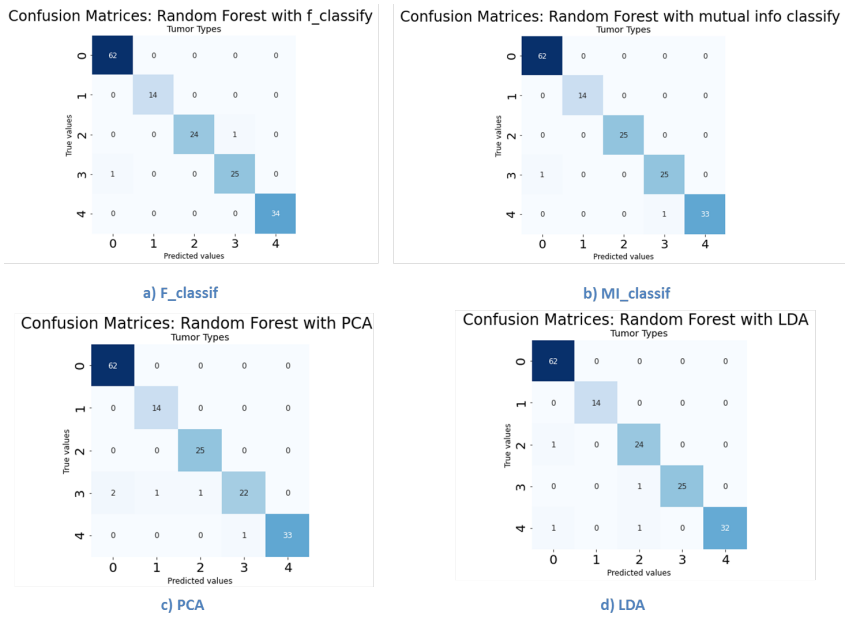


Figure 34: Random Forest Confusion Matrix (test set)

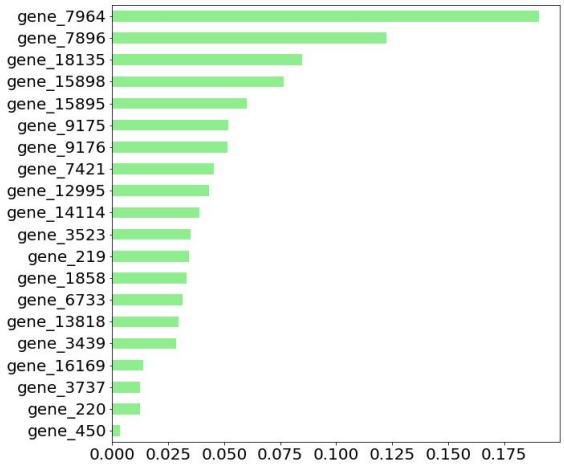


Figure 35: Random Forest Feature Importance

9. **AdaBoost**
AdaBoost is also an ensemble learning technique using boosting and shallow decision stumps. Learning rate and number of estimators were used as hyperparameters. Based on cross validation score, the final model was fitted with 300 estimators and a learning rate of 0.5. Selected cross validation plots are shown in Fig. 36 . The test set performance

is shown in Figs. 37 and 38.

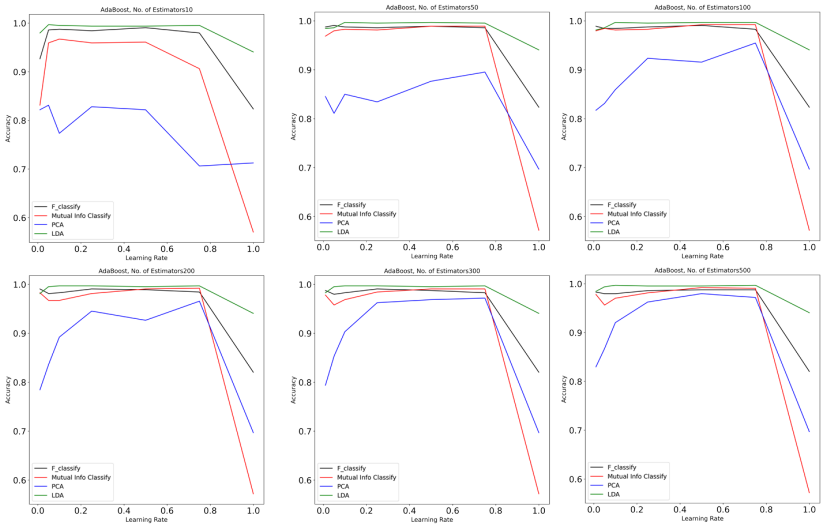


Figure 36: AdaBoost CV Score Plot

Model:AdaBoost with f_classify,Accuracy:0.9751552795031055
Model: AdaBoost with f_classify

	precision	recall	f1-score	support
0	0.98	1.00	0.99	62
1	1.00	1.00	1.00	14
2	1.00	0.88	0.94	25
3	0.89	0.96	0.93	26
4	1.00	1.00	1.00	34
accuracy			0.98	161
macro avg	0.98	0.97	0.97	161
weighted avg	0.98	0.98	0.98	161

a) F_classif

Model:AdaBoost with mutual info classify,Accuracy:0.9751552795031055
Model: AdaBoost with mutual info classify

	precision	recall	f1-score	support
0	0.98	1.00	0.99	62
1	1.00	1.00	1.00	14
2	1.00	1.00	1.00	25
3	0.89	0.96	0.93	26
4	1.00	0.91	0.95	34
accuracy			0.98	161
macro avg	0.98	0.97	0.97	161
weighted avg	0.98	0.98	0.98	161

b) MI_classif

Model:AdaBoost with PCA,Accuracy:0.968944099378882
Model: AdaBoost with PCA

	precision	recall	f1-score	support
0	0.94	1.00	0.97	62
1	1.00	1.00	1.00	14
2	1.00	0.96	0.98	25
3	0.96	0.88	0.92	26
4	1.00	0.97	0.99	34
accuracy			0.97	161
macro avg	0.98	0.96	0.97	161
weighted avg	0.97	0.97	0.97	161

c) PCA

Model:AdaBoost with LDA,Accuracy:0.9751552795031055
Model: AdaBoost with LDA

	precision	recall	f1-score	support
0	0.97	1.00	0.98	62
1	0.93	1.00	0.97	14
2	1.00	0.96	0.98	25
3	1.00	0.96	0.98	26
4	0.97	0.94	0.96	34
accuracy			0.98	161
macro avg	0.97	0.97	0.97	161
weighted avg	0.98	0.98	0.98	161

d) LDA

Figure 37: AdaBoost Classification Report (test set)

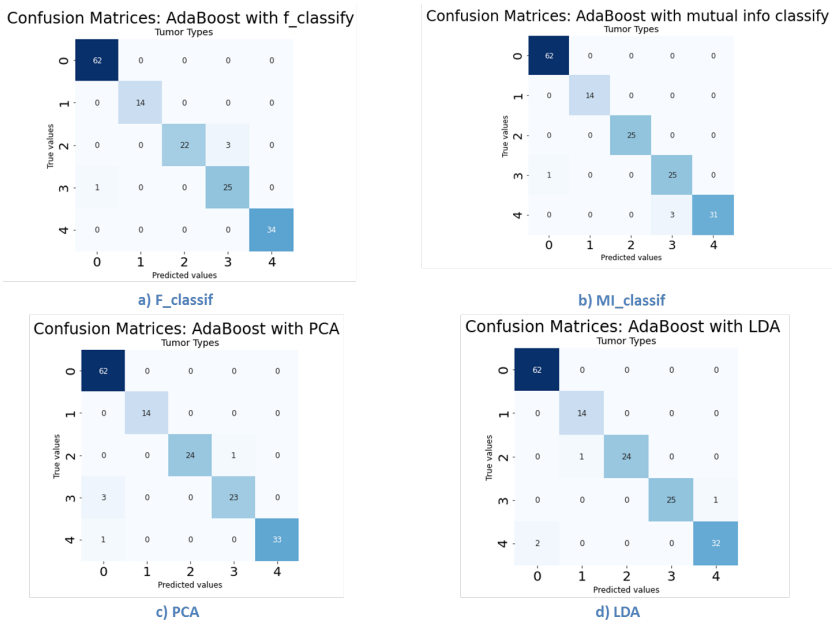


Figure 38: AdaBoost Confusion Matrix (test set)

Evaluation and Final Results

The performance of different classification algorithms are summarized in Table in terms of accuracy and Table in terms of worst f1-score for an individual tumor type. Accuracy is a valid metric for the present work as all tumor classes are adequately represented without significant under or over representation. F1-score captures the combined measure of the precision and recall of the classifier. All four classifier performed very well with accuracy greater than 0.93 in worst case and mostly greater than 0.96. The f1-score was also higher than 0.88 in worst case and mostly greater than 0.93. Among the classification algorithms, the simpler algorithms like k-NN , logistic regression, SVM Linear, and decision tree performed quite well. More complex algorithms like SVM Kernel, AdaBoost, and Random Forest did not show significant improvement for complexity. Naive Bayes and LDA are only applicable for the PCA feature set due to the approximations related to equal variance, independence of features, and normality.

Feature selection or dimensional reduction was a key part of this work. f_classif and mutual_info_classif both preserved the original dimensions and selected top 20 features. f_classif exhibited marginally better performance in general compared to mutual_info for the non-tree based methods and marginally poorer for the tree based methods. However, it takes considerably longer to run the mutual info algorithm. In general, PCA with 20 components performed better than LDA with four components in the non-tree based classification algorithms but the results were better for LDA with the tree based methods as they depend more on separation of classes than maximizing variance.

The decision boundary of the different classification algorithms are shown in the 2D LDA embeddings in Fig. 39. Although all the classification algorithms perform very well for dif-

ferent feature selection methods, there is considerable difference in the decision boundaries, ranging from the jagged k-NN boundary to the linear logistic regression, SVM Linear, and LDA boundaries to the squared tile decision boundary for tree and ensemble methods. Although, the accuracy of the test prediction is very high, it is not always reflected in the 2D embeddings as it might require higher dimensions to separate the classes.

Table 4: Summary Table of Accuracy

Method	f_classif	MI_classif	PCA	LDA
k-NN	1.00	0.99	0.99	0.98
Naive Bayes	1.00	0.99	0.99	0.98
LDA	1.00	0.99	1.00	0.98
Logistic Regression	0.99	0.99	1.00	0.97
SVM Linear	1.00	0.99	1.00	0.98
SVM Kernel	1.00	0.99	0.97	0.98
Decision Tree	0.98	0.96	0.93	0.98
Random Forest	0.99	0.99	0.96	0.98
AdaBoost	0.98	0.98	0.97	0.98

Table 5: Summary Table of Worst F1-Score

Method	f_classif	MI_classif	PCA	LDA
k-NN	1.00	0.98	0.98	0.97
Naive Bayes	1.00	0.98	0.98	0.93
LDA	1.00	0.98	1.00	0.97
Logistic Regression	0.99	0.98	1.00	0.97
SVM Linear	1.00	0.98	1.00	0.98
SVM Kernel	1.00	0.98	0.94	0.93
Decision Tree	0.93	0.94	0.88	0.94
Random Forest	0.96	0.96	0.90	0.94
AdaBoost	0.93	0.93	0.92	0.96

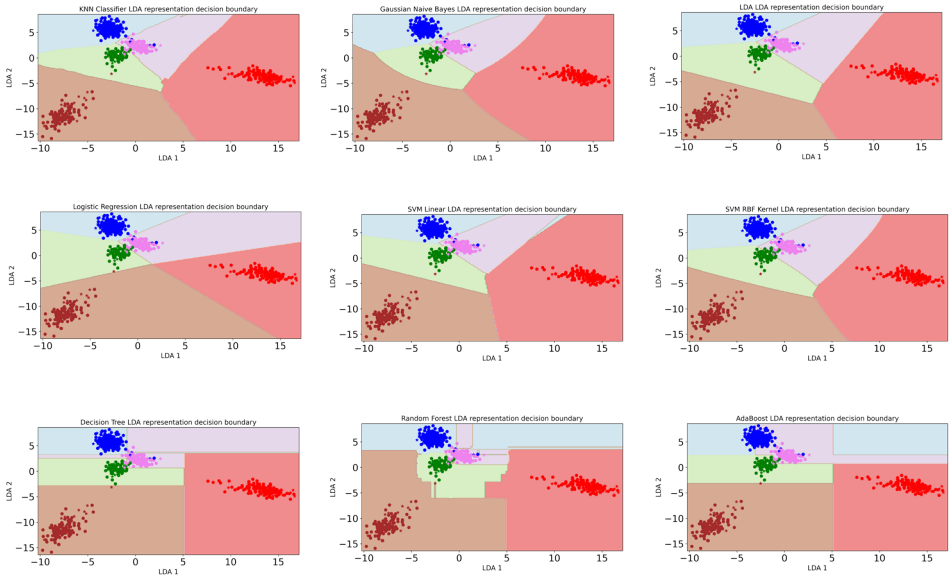


Figure 39: Decision Boundary: LDA 2D Representation

Conclusions

A total of two feature selection (f_classif and mutual_info_classif) and two dimensional reduction algorithms (PCA and LDA) were used to reduce or select features from the high-dimensional gene expression data. Thereafter, nine different classification algorithms were tested and results indicated excellent performance for all the algorithms. Each of the algorithms was capable of predicting the tumor class with high accuracy, precision, and recall. This demonstrates that gene expression data can be efficiently utilized to determine tumor class. It is the recommendation of the author to use f_classify with twenty features and use logistic regression to further reduce the number of variables by regularization and understand the impact of an individual gene on the tumor type. Elastic net will also address the issue related to multicollinearity. For predictive accuracy, k-nearest neighbor is easy to implement and performed very well for the present dataset.

References

Weinstein, John N., et al. 'The cancer genome atlas pan-cancer analysis project.' Nature genetics 45.10 (2013): 1113-1120.