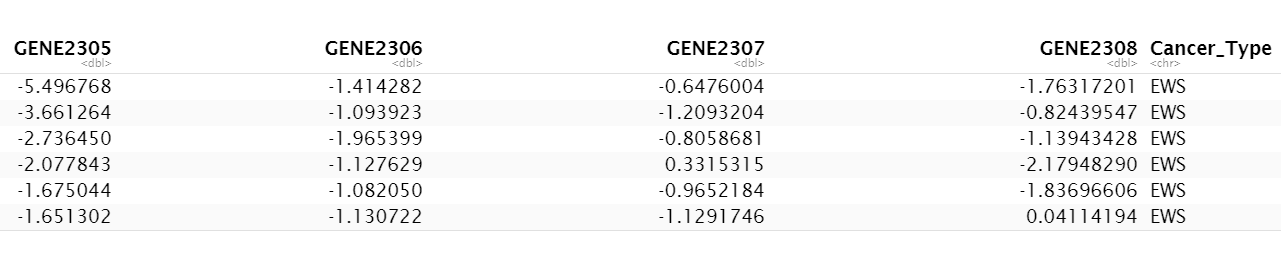
Tumor type classification using KNN, Decision Tree and SVM model

**Data Handling:**

We were given a dataset with 2309 rows and 63 columns, which we then had to transpose to classify the cancer types based on genes.

Here we have displayed just top 6 rows and the last 5 columns of our dataset of 63 rows and 2309 columns

As of now the cancer type variable is of data type string, so we have to convert it into type factor, post which we will see that we have 4 levels in the dependent variable, **Cancer\_Type**

## Levels: BL EWS NB RMS

**KNN Model Building:**

On checking the dataset we have found there are no null/missing values present, so we do not need to impute any data. Next we have used the function **findCorrelation**, available inside package **caret**, to find the highly correlated independent variables and remove them from our dataset. We have kept the **cut-off for correlation at 0.5**, which has then reduced the dataset into a dimension of 63 rows X 257 columns.

We have then used the 80-20 ratio to split our data into training and testing dataset, post which we have scaled both the datasets so that our machine learning algorithm does not give more weightage to higher values and less weightage to lower values irrespective of their units. We used the function **scale** in R for scaling the datasets.

Next we will now classify our data using **KNN** without any kind of dimensionality reduction. We are using the library **class** for this purpose.

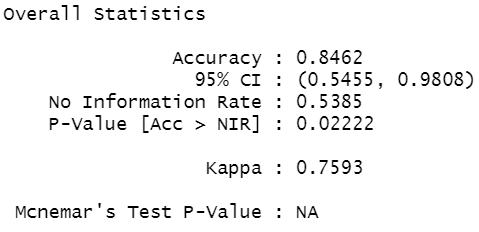
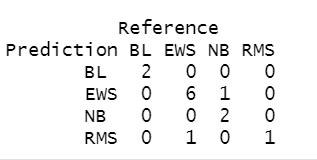
Post **KNN** if we look at the predicted classes and the actual classes we will notice that some classes has been incorrectly predicted.

y\_predicted

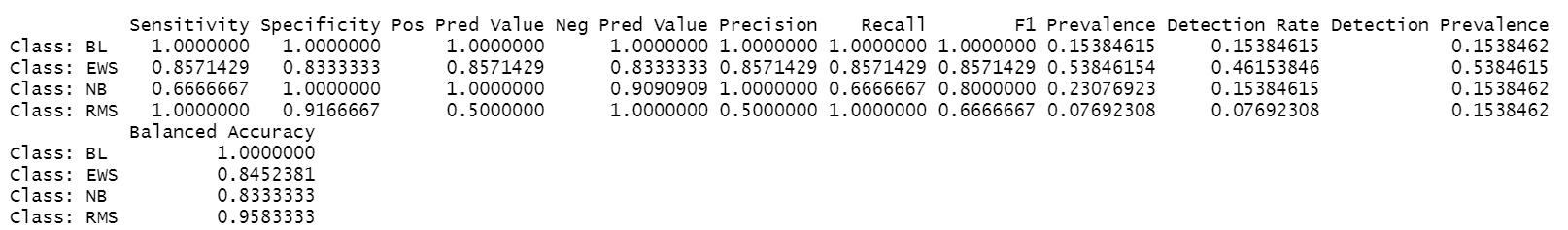
## [1] EWS EWS RMS EWS EWS EWS EWS RMS NB NB EWS BL BL   
y\_actual

## [1] EWS EWS EWS EWS EWS EWS EWS RMS NB NB NB BL BL

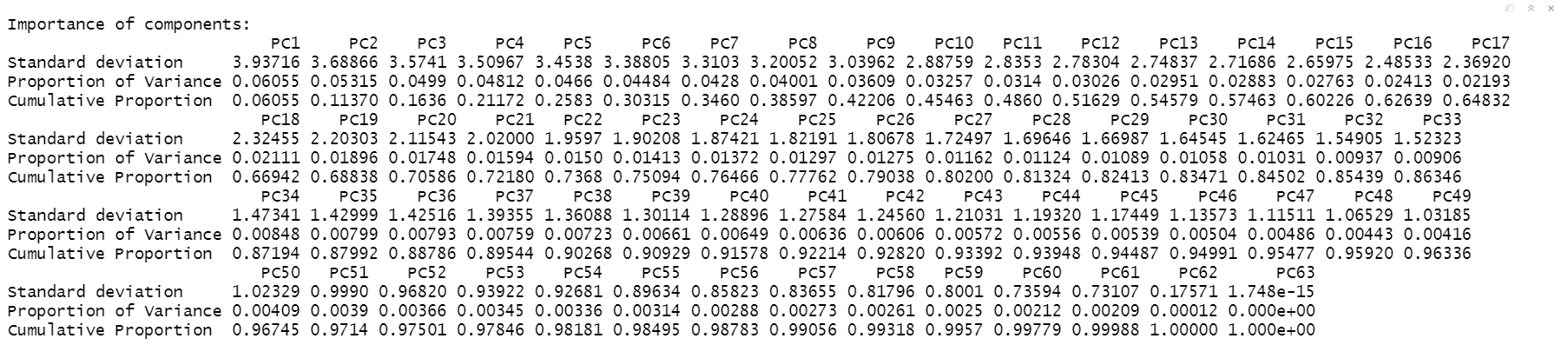
Thus we will now look at the accuracy of our findings using a confusion matrix, for which we will be using the library **caret.**



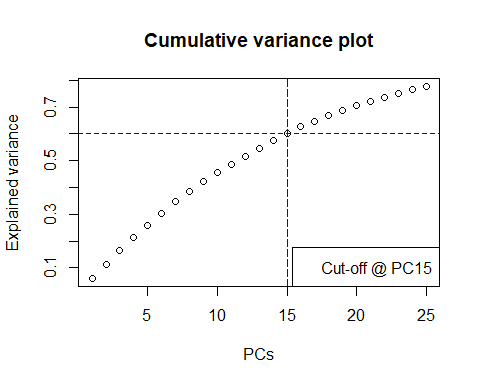
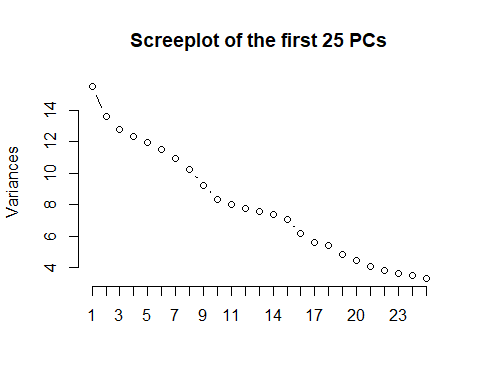
From the confusion matrix it is clear that all the classes except class EWS has been predicted correctly. There are in total 7 EWS cancer data in our actual training data, however only 6 has been correctly classified and the rest 1 has been misclassified as of type NB. The same applies for type RMS, which has been wrongly predicted 1 time as EWS. Also the **overall accuracy stands at 84.62%**



We can see that the **F1 scores for class BL, EWS, NB, RMS are 1, 0.86, 0.67, 1** in that order, which even though it is good but can still be increased by dimensionality reduction of our dataset, and that we will perform next.



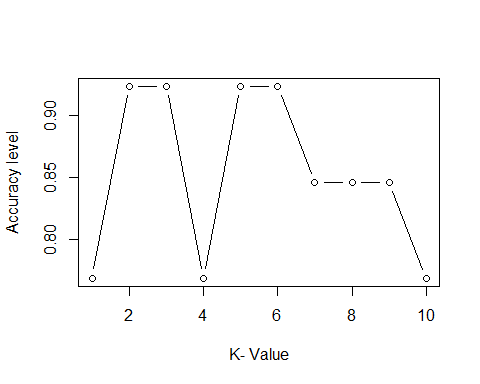
From the above summary of the PCA components we can see that **PC1** is explaining only **6% variance** of the dataset and **PC30** is explaining **84% variance.** We have then tried building our model with 30 PC component, then with 25, then again with 20 and then finally with **15 PC components**. Our finding was that the **accuracy** remained the same from 30->15 components, i.e. **92.307 %.** Thus finally we have used **15 PC components for our model.**



Next we built a process to find out the best **k value** for our **KNN** model using **15 PC components** by calculating the accuracy for each **k**, starting from 1 to 10. We have used the library **class** for this purpose.

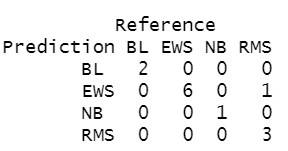
Accuracy range for k: 1->10

****

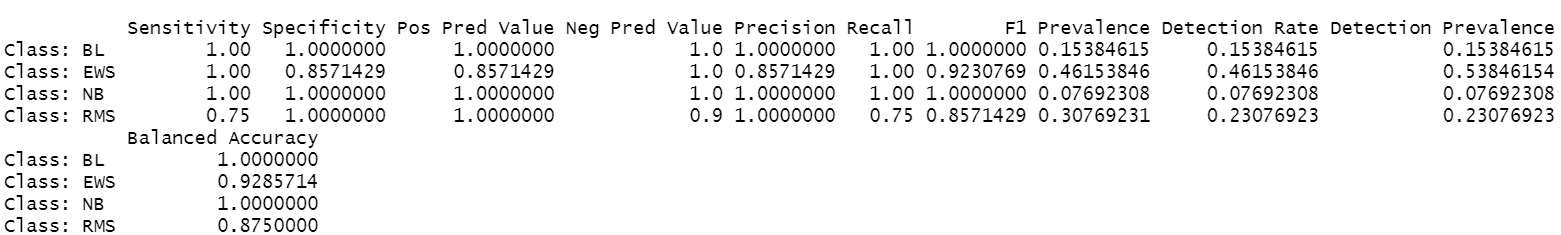


From the above plot we can see that with **k=5 we are getting an optimal accuracy (92.307%)** that does not seem to overfit our model **unlike k=4 or k=7**. Also with **5** being an odd number there **are no chances of a tie** being possible when the model is trying to classify the cancer types.

Let’s check the confusion matrix now with the KNN model when predicted upon the test dataset, where **k=5** and **15 PC components** has been used. The library used here is **caret.**

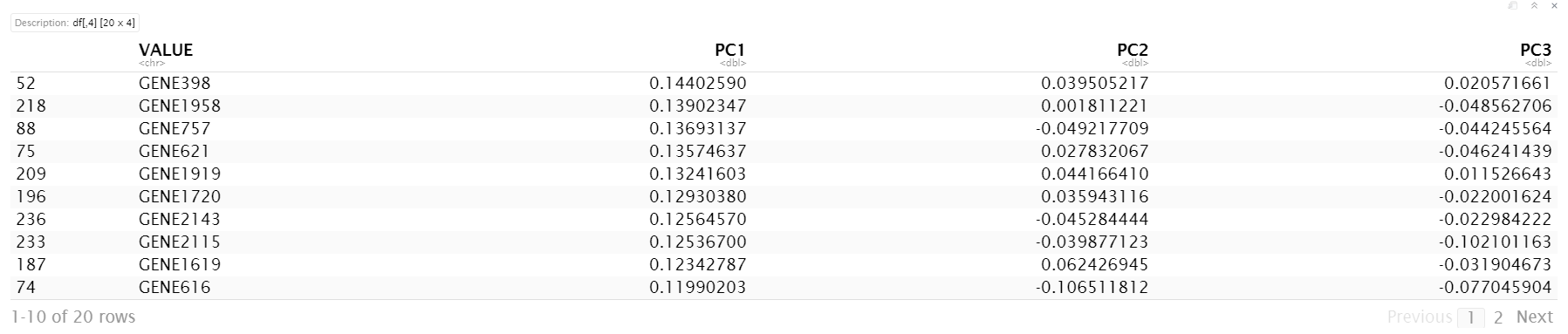


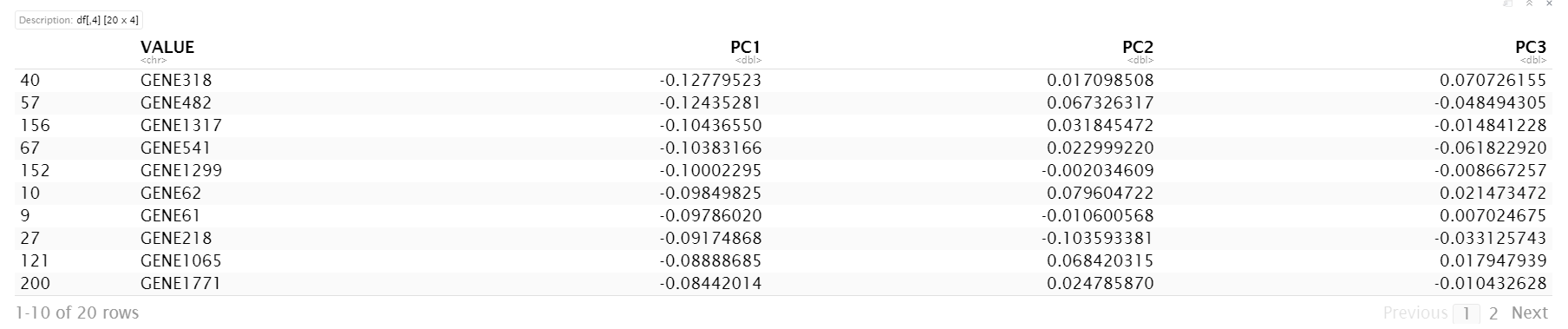
It’s apparent that our prediction now has quite improved from before and if we look at the F1 scores below, we will see it has improved drastically.



Next we have used the **library pROC**  to check the **AUROC, which is 0.9643.** This indicates how accurate our model is now.



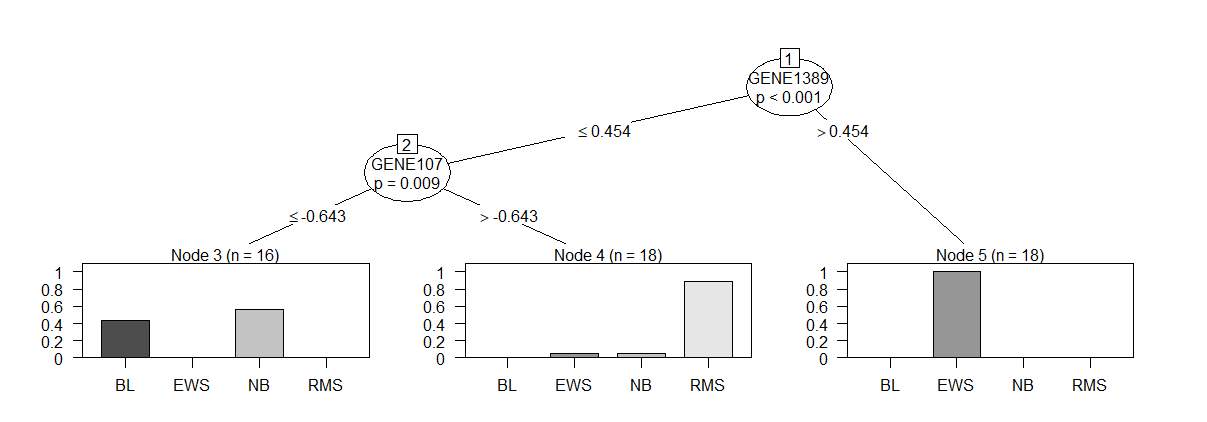
Upon **tracing back the loadings of our PCA components** we were able to identify the top 20 genes which are responsible for the cancer types in our dataset.



Thus with **KNN**, preceded by correlation removal and PCA we are getting an overall accuracy of **92.307%** which then helps us in understanding the **top 20 genes** that are responsible for the cancer types in our dataset. The **overall F1 score** stands at **0.98** and the **AUROC** stands at **0.9643**

**Decision Tree Model Building:**

* The **ctree()** function belonging to **‘party’** package is used to apply Decision Tree clustering
* The model is trained using the training data
* The model is plotted. Below is the graph:



* The test dataset is then applied on the model to predict. Below is the outcome:

**y\_pred = [1] EWS EWS EWS EWS RMS RMS RMS EWS NB NB NB**

**Levels: BL EWS NB RMS**

* The confusion matrix is calculated. Below is the outcome:

Reference

Prediction BL EWS NB RMS

BL 0 0 1 0

EWS 0 4 0 0

NB 0 0 2 0

RMS 0 1 0 3

**Overall Statistics:**

Accuracy : 0.8182

95% CI : (0.4822, 0.9772)

No Information Rate : 0.4545

P-Value [Acc > NIR] : 0.01598

Kappa : 0.7349

Mcnemar's Test P-Value : NA

**Statistics by Class:**

Class: BL Class: EWS Class: NB Class: RMS

Sensitivity NA 0.8000 0.6667 1.0000

Specificity 0.90909 1.0000 1.0000 0.8750

Pos Pred Value NA 1.0000 1.0000 0.7500

Neg Pred Value NA 0.8571 0.8889 1.0000

Prevalence 0.00000 0.4545 0.2727 0.2727

Detection Rate 0.00000 0.3636 0.1818 0.2727

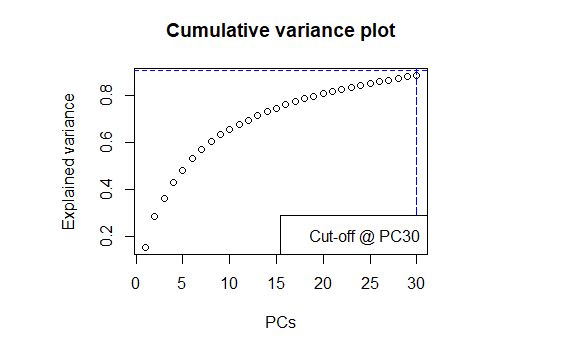
Detection Prevalence 0.09091 0.3636 0.1818 0.3636

Balanced Accuracy NA 0.9000 0.8333 0.9375

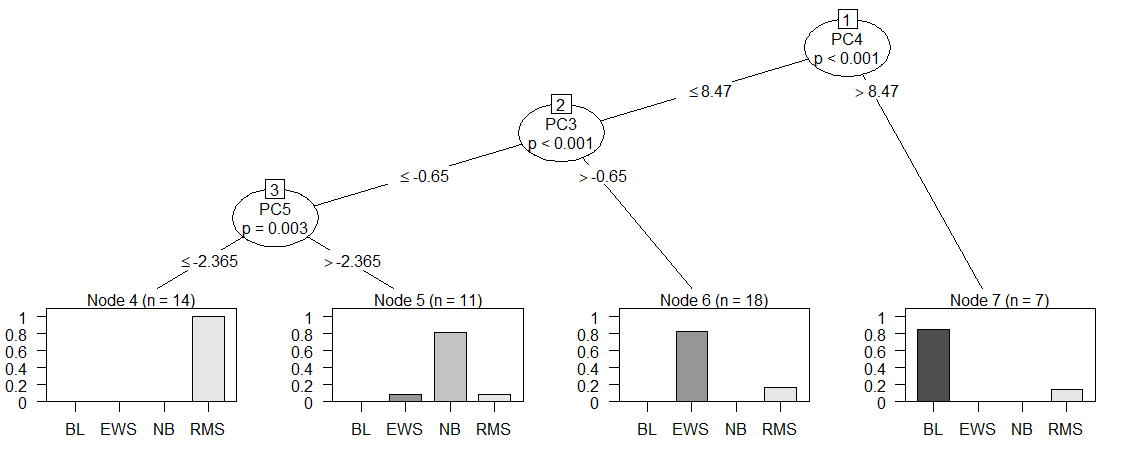
**Thus, the model gives an overall accuracy of 82%**

**Dimension Reduction**

* Principal component analysis is applied and first 30 components are selected. Below is the component vs variance graph:



* Train and test sets are prepared using the extracted features with a ratio of 20:80
* The model is trained using the training data
* The model is plotted. Below is the graph:



The test dataset is then applied on the model to predict. Below is the outcome:

**y\_predicted = [1] EWS EWS EWS EWS EWS EWS EWS RMS NB NB NB BL BL**

**Levels: BL EWS NB RMS**

* The confusion matrix is calculated. Below is the outcome:

Reference

Prediction BL EWS NB RMS

BL 2 0 0 0

EWS 0 7 0 0

NB 0 0 3 0

RMS 0 0 0 1

**Overall Statistics**

Accuracy : 1

% CI : (0.7529, 1)

No Information Rate : 0.5385

P-Value [Acc > NIR] : 0.0003199

Kappa : 1

Mcnemar's Test P-Value : NA

**Statistics by Class:**

Class: BL Class: EWS Class: NB Class: RMS

Sensitivity 1.0000 1.0000 1.0000 1.00000

Specificity 1.0000 1.0000 1.0000 1.00000

Pos Pred Value 1.0000 1.0000 1.0000 1.00000

Neg Pred Value 1.0000 1.0000 1.0000 1.00000

Prevalence 0.1538 0.5385 0.2308 0.07692

Detection Rate 0.1538 0.5385 0.2308 0.07692

Detection Prevalence 0.1538 0.5385 0.2308 0.07692

Balanced Accuracy 1.0000 1.0000 1.0000 1.00000

**The overall accuracy after application of pca is 100%. This indicates the model is over trained.**

**SVM Mode Building:**

Data Preprocessing:

* Data is divided into train and test in 80:20 ratio
* Checked for the null values and removed if any.
* Scaled train and test data individually.
* Then I have applied **svm** function on the training data by importing ‘**e1071’** package which consists of **svm**().

Graphical user interface, text, application

Description automatically generated

* Then predict function is used to obtain the predicted values when the above obtained ‘model’ is applied on test data.

Text, letter

Description automatically generated

* Then confusion matrix function is used to obtain the confusion matrix for better interpretation of correctly and wrongly predicted values, then sequentially I found the model accuracy.

Table

Description automatically generated

* Chart, scatter chart

  Description automatically generatedThe accuracy obtained is 100% due to overfitting.

Chart, scatter chart

Description automatically generated

* As the classification is not done properly due to overfitting, PCA is used in the further steps.

**Dimensionality Reduction using PCA:**

* PCA is applied on the entire data for getting columns which have more impact on the target value.
* To interpret PCA a Scree plot is plotted.

Chart, scatter chart

Description automatically generated

* After analyzing the principal components obtained till PC30 a cumulative proportion of 88% (0.88643) is obtained.
* Then PC scores from 1-30 are taken and divided into train and test in 80:20 ratio.
* Then svm function is applied on the updated train data.

Graphical user interface, text, application

Description automatically generated

* Then predict function is used to obtain the predicted values when the above obtained model is applied on test data.

A picture containing calendar

Description automatically generated

A picture containing text

Description automatically generated

* Then confusion matrix function is used to obtain the confusion matrix for better interpretation of correctly and wrongly predicted values, then sequentially I found the model accuracy.

A close-up of a document

Description automatically generated with low confidence

* The accuracy obtained is 92.307%. So, there is no overfitting.



**Findings & Conclusion:**

|  |  |
| --- | --- |
| Topic | Findings |
| KNN before PCA | Gives an accuracy of 84.62% |
| KNN post PCA (15 PCs) | Gives an accuracy of 92.307% and an F1 score of 0.98 |
| Decision Tree before PCA | Gives an accuracy of 82% |
| Decision Tree post PCA (30 Pcs) | Overfits with 100% accuracy |
| SVM before PCA | Overfits with 100% accuracy |
| SVM post PCA | Gives an accuracy of 92.307% |

From the above table it’s quite clear that KNN and SVM are the two best models that gives an accuracy of 92.307% in our prediction of cancer types, where KNN is providing an F1 score of 0.98. Also by backtracking the loadings of PC components in the KNN model we were able to predict the 20 genes that were most responsible in predicting the cancer types.

**Future Scope:**

Even though we were able to predict quite accurately with the 3 models, but we still had the challenge of using a **very small dataset** of 63 rows. Also the **class imbalance** in this dataset might have led to the overfitting of few of the models. We thus **aim** to create a better model with **many more samples** and **balanced classes**. This will help in reducing the overfitting and the prediction rates will be higher.