Tumor Type Detection on Gene Expression Cancer RNA-Seq Data

Sai Pavan Kamma, Krishna Murthy Takkillapati, Sankalp Pandey, Nitin Ramesh

1 Department of Computer and Information Science and Engineering, University of Florida, Gainesville, Florida, United States of America

I Abstract

Cancer can be defined as the unfettered growth of cells in the body. One of the main reasons for cancer is genetics, this is usually due to mutations in the DNA or by inheritance. Due to the large size of genetic data, it is impossible to understand the significance a gene has in causing cancer. With the advancement of machine learning, we can leverage modern computing power to deal with such large datasets. As the size of data increases the performance of the model, whereas the dimensionality reduces the same. In this paper, we are using machine learning algorithms to classify 801 (Ribonucleic acid)RNA-Seq gene expression samples to predict whether a particular sample is likely to be one of the five cancers namely Breast cancer (BRCA), Renal Cancer (KIRC), Colon Cancer (COAD), Lung Cancer (LUAD) and Prostate Cancer (PRAD). Each sample is composed of 20,531 genes. To reduce the dimensionality and get optimal features we used principal component analysis (PCA). For classification, we have used random forest, naive Bayes, and ANN(Artificial Neural Networks). An accuracy of 98.69% was achieved using the random forest. Our result demonstrates that using PCA for feature selection in RNA-seq data improves the accuracy of the machine learning algorithms.

II Author summary

In this paper, we have attempted to classify the detected cancer sub-types. Our technique follows a twofold approach for the classification. The data input size being large is first processed via Principal Component Analysis (referred to as PCA). This helps reduce the dimensionality and thereby increases the quality of the input feature set. Having a reduced PCA space, the features are then processed by various methods for the classification of the cancer type. The methods we have selected for this purpose include multiple machine learning algorithms such as Random Forest, ANN(Artificial Neural Network), and Naive Bayes. We have facilitated easy analysis of the output by visualizing the output in the form of a graph indicating the type of cancer.

III Introduction

RNA sequencing (RNA-seq) was first created more than a decade ago [1, 2] and has since become a common technique in molecular biology, influencing practically every area of our understanding of genomic function. RNA-Seq is emerging as a robust approach for transcriptome analysis that will eventually supplant microarrays in gene expression analysis [3]. Cancer categorization based on genetic profiles obtained by sequencing enables differentiation between healthy and ill people, as well as between different kinds and subtypes of cancer [4, 5]. This finding aids in the diagnosis, modification, and improvement of patient therapies.

The Hughes Effect [6], which argues that increasing dimensionality affects the reliability of the estimate of the statistical parameters necessary to calculate probabilities, is one of the most complex problems in cancer classification. Thousands of genes are represented in the DNA sequences, despite the small number of samples. Furthermore, the DNA sequences contain a large number of genes that are unrelated to the target cancer types since they have no bearing on categorization. As a result, the classification algorithms' performance suffers significantly.

As a result, various research concentrating on the selection of a limited set of genes that are important for categorization has been created. Many authors have used feature selection approaches in machine learning algorithms to tackle the challenge of detecting disease genetic markers. Filter selection approaches to rank the features in a collection to find the ones that are most useful for categorization. Pavithra et al. [7] set the filters according to the mutual information, Guyon et al. [8] ordered them according to the weights of a recursive Support Vector Machine (SVM) trained for classification. As the first step to classification is, utilizing linear discriminant Analysis(LDA), or quadratic discriminant analysis. Nguyen and Rocke (2002) and Gosh (2002) recommended employing the partial least squares (PLS) approach for dimension reduction (QDA). During our literature survey we noticed that there is no work which uses PCA for feature selection on this dataset.

In this paper, we use and compare different machine learning models' accuracy to classify an unbalanced database of 801 samples of gene expression RNA-Seq data in 5 types of cancer [9]. Here, we apply Principal Components Analysis(PCA) to feature selection. Principal Components Analysis (PCA) is a well-known unsupervised dimensionality reduction approach that creates meaningful features/variables by combining the original variables in linear (linear PCA) or non-linear (kernel PCA) ways (features). By linearly reducing correlated data into a smaller number of uncorrelated variables, useful features may be constructed. This is accomplished by projecting (dot producing) the original data into the reduced PCA space using the covariance/correlation matrix eigenvectors, also known as the principal components (PCs). This reduced uncorrelated dimensionality space improves the performance of the classification algorithms.

Random decision forests (RF) methods for classification or regression work by averaging multiple decision trees, each being trained on different parts of the training set, the method successfully reduces variance while significantly boosting the performance.

PCA here plays an important role in ensuring the efficiency and accuracy of the above methods. [10] They overcome the limitations mentioned by reducing the dimensionality of the features and encapsulating all collinear features as a single PCA component, thus helping RF to perform with higher accuracy. For example, Random Forests do not perform when the selected features are the monotonic transformation of other features. Additionally, in cases where features outnumber the samples, RF tends to overfit the data. PCA's reduction of dimensionality and clustering of similar items increases accuracy and efficiency.

The paper is structured as follows: Section IV describes the dataset from UCI [8] used to carry out the experiments and the later part explains the theoretical concepts of the algorithms such as ANN, Random Forest and Naive Bayes. Section V presents the results obtained for the data analyzed. Section VI offers the discussions of the research conducted. Finally, the conclusions of the research are included in Section VII.

IV Materials and methods

IV.1 Data

We have used the standard 'gene expression cancer RNASeq' [9] Dataset from UCI. The data set was collected from 801 patients, which consist of random extraction of gene expression. Each patient has suffered from one type of tumor: BRCA, KIRC, COAD, LUAD, and PRAD. Table 1 shows the details of the dataset.

Dataset Characteristics	Multivariate
Associated Tasks	Classification, Clustering
Attribute Characteristics	Real
Area	Life
Number of Attributes	20531
Number of Instances	801

Table 1. Gene expression cancer RNA-Seq Dataset Description

Each sample has 20,531 genes, the features of each sample are RNA-Seq gene expression levels that are measured with the Illumina HiSeq platform. Composition is 136 samples are patients with PRAD tumor, 300 samples are patients with BRCA tumor, 141 samples are patients with LUAD tumor, 146 samples are patients with KIRC tumor and the remaining 78 samples are patients with COAD tumor. So all samples are divided into 5 classes as 5 tumor types as given in Table 2. The data set contains 20531 features of each sample. [11]

Available Cancer Types	Abbreviation	Number of Samples
Breast invasive carcinoma	BRCA	300
Kidney renal clear cell carcinoma	KIRC	146
Lung adenocarcinoma	LUAD	141
Prostate adenocarcinoma	PRAD	136
Colon adenocarcinoma	COAD	78

Table 2. Distribution of patients to tumors classes in Gene Expression Cancer RNA-Seq Data Set

The data were divided into training and testing sets. The training set is used to train the model while the test set is used for gauging the accuracy of the model. The split is along 85% for the training set and 15% for the testing set. The source of all our qualitative coding results can be found at
`https://github.com/saipavan10-git/Genomics-Project`

IV.2 Primary Methods

IV.2.1 ANN

ANNs are biologically inspired computer programs designed to simulate how the human brain processes information. ANNs gather their knowledge by detecting the patterns and relationships in data and learn (or are trained) through experience, not from the programming. [11] An ANN is formed from hundreds of single units, artificial neurons, or processing elements (PE), connected with coefficients (weights), which constitute the neural structure and are organized in layers. The power of neural computations comes from connecting neurons in a network. Each PE has weighted inputs, a transfer function, and one output. The weights are the adjustable parameters, and, in that sense, a neural network is a parameterized system.

$$y = \sum w_i * x_i + b; n \in i = 1, where :$$

$w_i \rightarrow$ weights of a neuron
$x_i \rightarrow$ input of a neuron
b $\rightarrow$ bias

The weighted sum of the inputs as shown in the above equation constitutes the activation of the neuron.

The activation signal is passed through the transfer function to produce a single output of the neuron. The transfer function introduces non-linearity to the network. During training, the inter-unit connections are optimized until the error in predictions is minimized and the network reaches the specified level of accuracy. [12] Once the network is trained and tested it can be given new input information to predict the output.

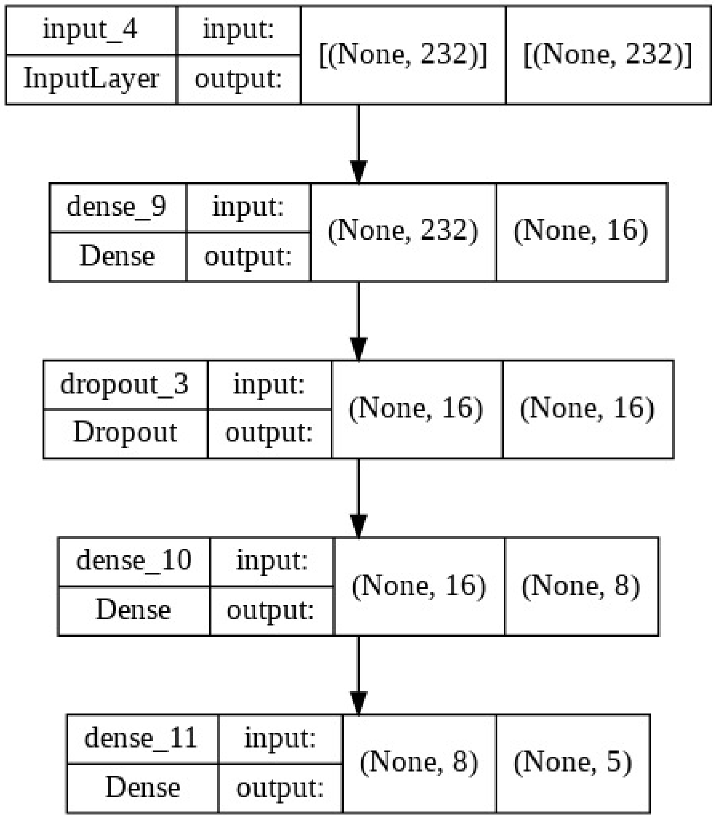

Fig 1. The figure shows the architecture of ANN we used in our work. The ANN contains an input layer, two hidden layers, and an output layer. We also added dropout to the second hidden layer to avoid the possibility of overfitting. We are also using the softmax activation function for the output layer as softmax returns the probabilities of each class thereby readily classifying the output of the ANN.

IV.2.2 Random Forest

Random Forest is a supervised machine learning algorithm that uses an ensemble of decision trees, trained with a “Bagging” method. It can produce better results even without hyper-parameter tuning. One big advantage of random forest is that it can be used for both classification and regression problems, which form most current machine learning problems. Another great quality of the random forest algorithm is that it is very easy to measure the relative importance of each feature on the prediction. Sci-kit-Learn enables a great framework for this that measures a feature’s importance by looking at how much the tree nodes that use that feature reduce impurity across all trees in the forest. It computes this score automatically for each feature after training and scales the results, so the sum of all importance is equal to one. The problem of overfitting is circumvented by random forests if there are enough trees in the forest.

Feature Randomness: In a vanilla decision tree, when it’s time to separate a node, we consider every possible feature and pick the one that produces the foremost separation between the observations in the left node vs. those in the right node. On the contrary, each decision tree during a random forest can choose only from a random subset of features. This ensures even more multifariousness amongst the trees within the model and resulting in a lower correlation across trees and more diversification. This results in a wide medley that generally results in a better model. [14] The strength of random forest is that a significant number of unrelated trees functioning as a cluster will outperform any of the discrete constituent models.

Algorithm 1 PseudoCode for RandomForest

```
Precondition : A training set S := $(x_1, y_1), ..., (x_n, y_n)$, features F, and number of
               trees in forest B.
Function RandomForest($S$,$F$):
    $H \leftarrow \phi$
    for $i \leftarrow 1$ to $B$ do
        $S^i \leftarrow$ A bootstrap sample from S
        $h_i \leftarrow$ RandomTreeLearn($S^i$, F)
        H $\leftarrow H \cup h_i$
    end
    return H
End Function
Function RandomTreeLearn($S$,$F$):
    At each node:
        f $\leftarrow$ very small subset of F
        Split on best feature in f
    return The learned tree
End Function
```

IV.2.3 Naive Bayes

Naive Bayes models are a group of extremely fast and simple classification algorithms that are often suitable for very high-dimensional datasets. Because they are so fast and have so few tunable parameters, they end up being very useful as a quick-and-dirty baseline for a classification problem. [16] Naive Bayes classifiers build on Bayesian classification methods. These rely on Bayes's theorem, which is an equation describing the relationship of conditional probabilities of statistical quantities. In Bayesian classification, we're interested in finding the probability of a label given some observed features, which we can write as P(L | features):

$$P(L \mid features) = \frac{P(features \mid L)P(L)}{P(features)}$$

If we are trying to decide between two labels—let's call them L_1 and L_2—then one way to make this decision is to compute the ratio of the posterior probabilities for each label. All we need now is some model by which we can compute P(features | L_i) for each label. Such a model is called a generative model because it specifies the hypothetical random process that generates the data. Specifying this generative model for each label is the main piece of the training of such a Bayesian classifier.

The basic model was used for training and gave significantly lower accuracy. To improve performance, we used var_smoothing hyperparameter tuning along with GridSearchCV to get the best value for our model.

IV.3 Supporting Methods - Feature Selection

IV.3.1 PCA

Given a set of N features a PCA analysis will produce the linear combination of the features with the highest variance, the linear combination with the highest variance in the subspace orthogonal to the first PCA component, etc. (under the constraint that the coefficients of the combination form a vector with the unit norm) (under the constraint that the coefficients of the combination form a vector with the unit norm) It depends on what you're attempting to forecast if the linear combination with the highest variance is a "good" feature. Measurements of all the original variables are used in the projection to the lower-dimensional space.

Thousands of genes are represented in the DNA sequences, despite the small number of samples. Furthermore, the DNA sequences contain a large number of genes that are unrelated to the target cancer types since they have no bearing on categorization. Our dataset samples are composed of 20531 genes. We apply principal component analysis on the dataset to remove multicollinearity which improves the interpretation of the parameters of our model. After applying this dimension reduction technique, our data contains a new sub-space of 692 genes.

Initially, in our implementation, we are fitting our pre-processor on the entire dataset that might cause data leakage from the test data, since the parameters of the pre-processing model will be fitted with knowledge of the test set. We rectified this problem by splitting the data in train/test first. Then, we fit our PCA model on only the training set and run the transform method on the test data.

IV.3.2 GridSearchCV

The performance of a model significantly depends on the value of hyperparameters. There is no way to know in advance the best values for hyperparameters so ideally, we need to try all possible values to know the optimal values. Doing this manually could take a considerable amount of time and resources and thus we use GridSearchCV to automate the tuning of hyperparameters. GridSearchCV is a function that comes in Scikit-learn's model_selection package. This function helps to loop through predefined hyperparameters and fit your estimator (model) on your training set. So, in the end, we can select the best parameters from the listed hyperparameters. We have used GridSearchCV to search for the best possible value of var_smoothing hyperparameter in our Naives Bayes algorithm.

IV.4 Measures for Performance Evaluation

The PCA was designed to retain 85% variance of the original data(Fig 2. shows the distribution of data after applying PCA). The result was that 236 principal components were identified. For the random forest, we used a 100 estimator(decision trees (fig 5.)). Our initial naive Bayes model produced lesser accuracy compared to random forest. So we applied hyper-parameter tuning. For hyper-parameter tuning, we used grid search CV.

The grid search algorithm maps out the search parameters(in this case var_smoothing) in a grid and searches for the optimal parameter for the data. Along with grid search cross-validation is also applied, we applied 10 folds for 50 random candidates(features) 100 times. The ANN model for our work was designed using a sequential model (which is a linear stack of layers) in Keras. There are two hidden layers containing 16 units(neurons) each and having relu as the activation function. The model was trained for 10 epochs with batch sizes of 8. We also added a callback to prevent possible over-fitting, we reduced the learning rate by 0.001 if the loss plateaued after 3 epochs.

The performance of our methods was evaluated using three types of evaluation measurements: precision, f1-score, and accuracy, which is explained by the confusion matrix 5.

	Predicted Positive	**Predicted Negative**
Actual Positive	**TP (True Positive)**	**FN (False Negative)**
Actual Negative	**FP (False Positive)**	**TN (True Negative)**

Table 3. Confusion Matrix

V Results

With 20531 features, the application of PCA helped identify the dimensions of the data in the dataset with the largest variance out of the overall variance and helped to reduce the dimensionality of the data set by linearly reducing correlated data from a smaller number of uncorrelated variables. This aided in the extraction of the high-quality features in the reduced PCA space using covariance/correlation matrix eigenvectors.

The PCA produced a cumulative variance of 85% and discovered a total of 236 principal components. During which, the PCA computes the principal components and applies them to perform a change of basis on the available data. This sometimes leads to the usage of just the first few principal components while ignoring the rest. After the application of PCA, the resulting dataset was divided proportionally into the training and test data sets. Where the training set was used to train the algorithm and the test set was used to measure the effective performance. We have opted for a base configuration to use 75% for the training set and 25% for the testing set.

The ANN model was implemented using the keras module. We designed a ANN (Fig 1) with two hidden layers. The network was fitted on the data transformed by the PCA. Using this optimized PCA space, we allocated 75% of the data for training 25% for testing purpose. After successfully running, the samples were accurately classified with an overall accuracy of 98.18%

Models	**Accuracy**	**Precision**	**F1 score**
Random Forest	98.69	98.41	98.9
Naive Bayes	69.17	71.04	69.96
Naive Bayes(with Hyperparamter tuning)	98.17	98.8	98.96
Artificial Neural Network	98.18	99.53	98.7

Table 4. Performance metrics of different models

Models	AUC ROC Score (macro)		AUC ROC Score (weighted)	
	One vs One	One vs Rest	One vs One	One vs Rest
Random Forest	99.5	99.9	99.9	99.9
Naive Bayes(without tuning)	89.7	89.1	89.07	87.7

Table 5. Performance metrics of different models

The Random Forest model was given the input data in the ratios of 75:25 for training and testing respectively with n_estimators of 100 and min_samples_leaf of 80. Following the application of the model to test data, 52 samples are correctly classified as BRCA, 9 samples as COAD, 21 samples as KIRC, 20 as LUAD, and 17 samples as PRAD (Fig 3). This produced an overall accuracy of 98.69%.

The Naive Bayes model worked with a similar 75:25 configuration. After applying the model on test data, 25 samples are correctly classified as BRCA, 7 samples as COAD, 16 samples as KIRC, 16 as LUAD, and 15 samples as PRAD (Fig 4). The basic model was used for training and produced 69.17% accuracy. We used var_smoothing which is a stability calculation to widen (or smooth) the curve and therefore account for more samples that are further away from the distribution mean. After applying this hyperparamter tuning, the model accuracy improved to 98.17%.

All experiments were performed with a 3.2 GHz AMD Ryzen 7 3750H processor. The results presented below are the average values obtained from 10 repetitions of training and testing. In our experimentation, we found Random Forest to be the best among all the models across all the metrics. This was the reason why we did not use any search algorithm for hyperparameter tuning in random forest.

The application of the various algorithms presented certain issues with some algorithms producing less accurate results than others, thus presenting an opportunity to fine-tune. The application of the Naive Bayes algorithm generated a less than expected accuracy of 69.17%. This suggests the requirement of additional hyperparameter tuning can be applied in this situation. Similarly to standardize the sampling, the implementation of stratified cross-validation can be done to the other models implemented. Rigorous cross-validation of results across additional models can be applied to ensure better overall accuracy. During the training of ANN, we noticed that the loss remained stagnant through some epochs so we used a validation technique to reduce the learning rate to prevent the issue again.

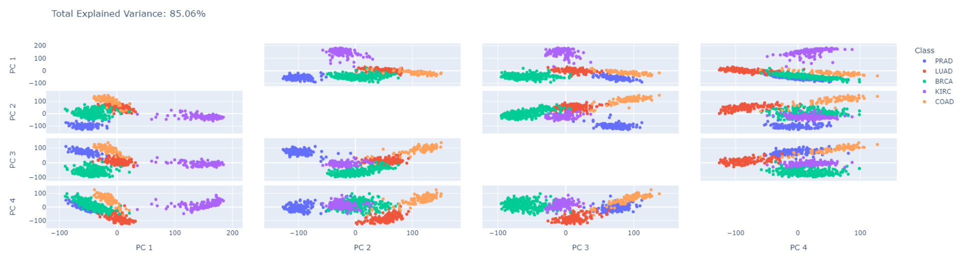

Fig 2. This image represents how the data are distributed after applying PCA. The image shows the resulting principal components, ordered by how much variance they can explain form. The importance of explained variance is that it represents how much of the original variance is still present after applying PCA. The subplot between PC3 and PC4 is unable to separate each class, whereas the subplot between PC1 and PC2 shows a clear separation between each cancer type. However, Principal components are linear combinations of the features of the original data, which is not easy to interpret. Additionally, there could be information loss because of dimensionality reduction, and this requires a compromise in the trade-off.

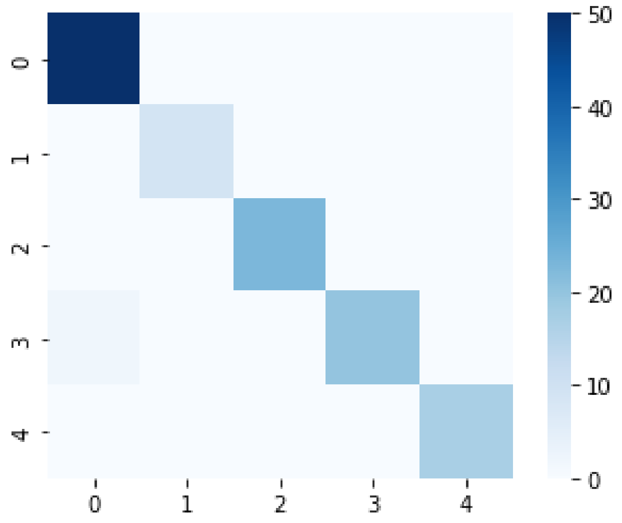

Fig 3. A confusion matrix plot showing correctly classified samples in each class (diagonal) by a Random Forest algorithm. After running the model on test data, it is observed that 52 samples are correctly classified as BRCA, 9 samples as COAD, 21 samples as KIRC, 20 as LUAD, and 17 samples as PRAD.

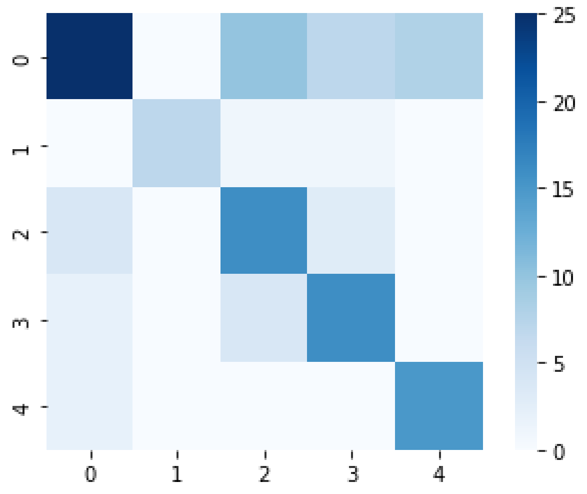

Fig 4. A confusion matrix plot showing correctly classified samples in each class (diagonal) by a Naive Bayes algorithm. After running the model on test data, it is observed that 25 samples are correctly classified as BRCA, 7 samples as COAD, 16 samples as KIRC, 16 as LUAD, and 15 samples as PRAD. It is also observed that 50% of BRCA samples are wrongly classified.

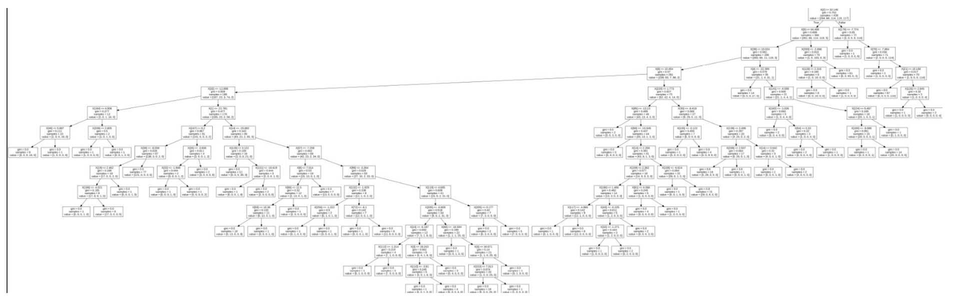

Fig 5. The above figure shows an estimator(tree) in the random forest. We had 100 estimators in our model. Random forest while growing the trees., instead of searching for the most important feature while splitting a node, it searches for the best feature among a random subset of features. This results in a wide diversity that generally results in a better model. The tree shows how a decision was made while splitting a node, one hyperparameter which played a role in the splitting is min_leaf_samples. If the number of leaf samples after splitting is less than 85 the node is not divided. Another factor is the Gini index, Gini index is a metric commonly used in random forests. The Gini index measures the variance in the samples, so the more the Gini index decreases for a characteristic after a split, the more significant it is.

VI Discussion

We have applied the Random Forest, Naive Bayes, and ANN to the database from the UCI Repository for the classification of DNA samples according to different types of cancer. The database taken from the UCI Repository collects a total of 801 samples belonging to 5 types of cancer: breast, kidney, colon, lung, and prostate (Table 2). This database is not balanced because the number of samples of each type is quite different:

the most numerous samples are those of breast cancer (300 in total) and the least frequent are those of colon cancer (78 samples in total). With this unbalanced database, several groups of genes are obtained that allow a classification acceptable accuracy.

We proposed an approach that uses PCA for feature selection to bring out the best features using covariance/correlation matrix eigenvectors and then used various machine learning algorithms and compared their performance on the principal components produced by the PCA.

We observed that the random forest produced better accuracy initially because of bagging. The random forest algorithm uses the bagging technique for building an ensemble of decision trees. Bagging is known to reduce the variance of the algorithm. Another aspect that might be the reason for the random forest to get a higher accuracy is the randomness of the model. In traditional bagging with decision trees, the constituent decision trees may end up being very correlated because the same features will tend to be used repeatedly to split the bootstrap samples. By restricting each split-test to a small, randomly picked sample of features, we can decrease the correlation between trees in the ensemble. This decreases the time taken to build the trees.

The ANN also produced high accuracy because it can handle complex natural input spaces. However, ANN is a black box learning approach, it cannot interpret the relationship between input and output and cannot deal with uncertainties. But since we have used feature selection along with ANN we have improved interpretability.

The obvious increase in accuracy of the Naive Bayes model after hyperparameter tuning might be attributed to the reason that the suitable hyperparameter for the data was obtained while tuning and cross-validation.

The methods implemented produced promising results across the selected features with room for additional improvement in the future. Random Forest, Naive Bayes, and Artificial Neural Network models were applied which produced an accuracy of 96.69, 65.27, and 98.27 respectively. Precision from Random Forest was 98.41, Naive Bayes produced 71.04, and precision from ANN was 84.47. The F1 scores from Random Forest, Naive Bayes, and ANN were 95, 69.96, and 87.12 respectively.

VII Conclusion

The fusion of genomics and machine learning ushered in a new era in the field of cancer detection and diagnosis. This paper aimed to compare the various machine learning methods and evaluate their results and efficiency to determine the most effective algorithm for the classification of the type of cancer. The large amount of genomics data being generated in recent years introduced a problem that is central to the concept of the Hughes effect. The affected reliability due to increasing dimensionality is one of the problems addressed in the paper. The unbalanced database containing 801 RNA Seq gene expression samples was composed of 20,531 samples. This required the reduction of dimensionality and that helped increase the quality of the throughput and the results.

Our novel approach to applying PCA in the project resulted in a quality input feature set to the methods and resulted in reduced dimensionality. Thus the methods of Random Forest, Naive Bayes, and ANN gave significantly better efficiency in the reduced PCA space. In terms of overall accuracy, the Random Forest gave the best results, followed by the ANN and lastly the Naive Bayes method. The usage of multiple methods to compare the results helped determine the best average accuracy of the prediction.

Overall the implemented solution presented a viable way of detecting the cancer types with the provided data. Future research in this area can include the implementation of additional techniques and methods to evaluate the performance and deliver better accuracy. Additional improvements in this space can be performed to get better feature selection techniques by using genetic algorithms.

References

1. Emrich, S. J., Barbazuk, W. B., Li, L., Schnable, P. S. Gene discovery and annotation using LCM-454 transcriptome sequencing.. Genome Res. 17, 69–73 (2007).

2. Lister, R. et al. Highly integrated single-base resolution maps of the epigenome in Arabidopsis. London: George Alien & Unwin Ltd. Berlin, Heidelberg and New York: Cell 133, 523–536 (2008).

3. Daniel Ramsköld, Ersen Kavak, Rickard Sandberg How to analyze gene expression using RNA-sequencing data

4. T.R. Golub, D.K. Slonim, P. Tamayo, C. Huard, et al. Molecular classification of cancer: class discovery and class prediction by gene expression monitoring Science, 286 (5439) (1999), pp. 531-537

5. A.A. Alizadeh, M.B. Eisen, R.E. Davis, C. Ma, et al. Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling Nature, 403 (6769) (2000), p. 503

6. Hughes, G.F. (January 1968). "On the mean accuracy of statistical pattern recognizers". IEEE Transactions on Information Theory. 14 (1): 55–63. https://ieeexplore.ieee.org/document/1054102

7. D. Pavithra, B. Lakshmanan Feature selection and classification in gene expression cancer data 2017 International Conference on Computational Intelligence in Data Science, IEEE (2017), pp. 1-6

8. I. Guyon, J. Weston, S. Barnhill, V. Vapnik Gene selection for cancer classification using support vector machines Mach. Learn., 46 (1–3) (2002), pp. 389-422

9. "UCI Machine Learning Repository: gene expression cancer RNASeq Data Set." [Online]. Available: https://archive.ics.uci.edu/ml/datasets/gene+expression+cancer+RNASeq

10. https://towardsdatascience.com/dimensionality-reduction-does-pca-really-improve-classification-outcome-6e9ba21f0a32

11. Pretrained Convolutional Neural Networks for Cancer Genome Classification https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=arnumber=9096808

12. Artificial Neural Networks Steven Walczak, Narciso Cerpa https://www.sciencedirect.com/topics/engineering/artificial-neural-network

13. Basic concepts of artificial neural network (ANN) modeling and its application in pharmaceutical research https://www.sciencedirect.com/science/article/pii/S0731708599002721

14. https://towardsdatascience.com/understanding-random-forest-58381e0602d2

15. $https://www.saedsayad.com/k_nearest_neighbors.htm$

16. $https://cocalc.com/share/public_paths/8b892baf91f98d0cf6172b872c8ad6694d0f7204/r$

17. Margaret H. Danham, and S. Sridhar, " Data mining, Introductory and Advanced Topics", Person education , 1st ed., 2006.