Date of submission April 10, 2021, date of current version April 10, 2021

Prostate Cancer Detection Using Machine Learning Techniques (April 2021)

C.Dickie [1], Senior Student, L.Khalil [1], Senior Student.

[1]University of Windsor, Windsor, Ontario, Canada
Corresponding authors: First C.Dickie (e-mail: dickiec@uwindsor.ca). Second L.Khalil (e-mail: khali121@uwindsor.ca)

ABSTRACT Prostate cancer is the world's most frequently diagnosed cancer in men. It is the second leading cause of death in older men. Prostate cancer, however, is not always fatal, especially when diagnosed early [1]. The majority of the symptoms associated with this type of cancer are similar to those associated with a variety of non-cancerous conditions. Prostate cancer, as a result, can go undetected for a long time. An elevated Prostate-specific antigen (PSA) level, for example, is not always due to cancer. That being said, if a patient's PSA level is high, he will almost always need a biopsy, after a series of cancer screening tests. Then, a Gleason scoring of 1 to 10 is used to classify the level of prostate cancer severity based on similarity to healthy cells. A low Gleason score (1-6) indicates that the cells are not too far removed from healthy cells and that the risk is low. A high Gleason score (7-10) indicates that cancer cells are distinct from healthy cells and are high risk [1]. Biopsies are widely used in cancer diagnosis because of their accuracy. A biopsy, however, is an invasive procedure that may result in post-surgery complications [2]. Innovations in early cancer detection and prediction methods have resulted from advances in data science, machine learning and the study of the genetic basis of common diseases. In this paper, we use various machine learning techniques to predict the Gleason score of patients based on their RNA genetic profile using a prostate adenocarcinoma dataset of 494 samples. This dataset was downloaded from the cBioPortal for cancer genomics [3]. The accuracy of our model ranges from 96 to 99 percent. Different feature selection methods and a group of classifiers with different hyperparameters are used to achieve this result. These results suggest that RNA expression tests may serve as a suitable and less invasive substitute for biopsies in Prostate Cancer detection and assessment.

INDEX TERMS Gleason Score, Genetic Expression, Machine Learning, Prostate Cancer diagnosis.

1. INTRODUCTION

Many biomedical interventions aim to improve medical diagnosis and therapy of terminal illnesses such as prostate cancer [4]. The number of machine learning applications for cancer diagnosis is rapidly increasing, leading some to wonder whether the need for biopsies will eventually disappear. Many bioinformatic studies aim to eliminate surgical interventions in cancer diagnosis. To help in the fight against prostate cancer, we designed a machine learning model which can predict and classify cancer's severity. Our objective is to accurately predict the Gleason Score, a measure of Prostate Cancer risk and severity, using only information about a person's mRNA transcript levels. Our model produces highly accurate and dependable results that can be used for prostate cancer diagnosis and prognosis in patients who have no symptoms and without a biopsy. The model was trained using a historical dataset that contains biopsy result samples of 494 individuals along with their gene expressions. Additionally, the model utilized a diverse combination of classifiers and feature selection methods to accomplish its goal. More details about the dataset,

classifiers and feature selection methods will be provided in the General Concepts section.

2. GENERAL CONCEPTS

2.1 Dataset

The cancer genomics dataset was obtained from cBioPortal. It contains historical records of 494 patients. Their identities are represented by their Cancer Genome Atlas *(TCGA) IDs* and their age, calculated in days since birth. This data was collected over five years (from 2011 to 2015). The biopsy diagnosis results for the patient are included in the dataset, as well as the Gleason scores. A second spreadsheet lists the expression levels for over 60, 000 exons for each of the 494 individuals.

2.2 Gleason Grading System

The Gleason Grading System is used in the evaluation and prognosis of prostate cancer. A grade is assigned based on visual assessment of a prostate biopsy [5]. A primary grade, ranging from 1 to 5, is assigned based on the dominant pattern identified in the tumor biopsy. See the figure in appendix B for the characteristics observed at each grade level. A secondary grade, also ranging from 1 to 5, is assigned to the next most frequent pattern identified. These two scores are added together to form the Gleason Score which will generally fall between 6 and 10. The Gleason Grade Group is another popular metric that assigns individuals a score of 1 to 5 based on the primary and secondary pattern scores. A tumor biopsy with a Gleason Score of 6 or less is considered low risk.

2.3 Ensemble Stable IDs

Stable identifiers are used to label genes, transcripts, exons, or proteins in a database. The IDs are comprised of letters and numbers, and each ID in the prostate cancer dataset begins with an 'E.' The E stands for exon; a distinct piece of an mRNA transcript. The expression levels of these transcripts can be measured (in fragments per kilobase of exon per million reads) and then linked to conditions such as prostate cancer.

2.4 Classification

Classification is a predictive modelling problem in machine learning, in which a class label is predicted for a certain input [6]. In classification problems the model approximates a function that maps the input variable to the predicated output variable i.e. $y = f(x)$, where y is a representation of x [6]. In this model, data = $\{x1, x2, \dots, xn\}$, where $x \in R^d$, and x represents gene expressions. The class labels: ω = Gleason_score ={6, 7, 8, 9, 10}.

2.5 Supervised Learning Classifiers

A supervised learning classifier uses a predefined dataset for training purpose. The algorithm then predicts how an unknown input variable will be categorized according to the training dataset labels. In this case, known Gleason scores were used as training data. If the supervised learning classifier is accurately trained, it will be able to predict the Gleason score of new, unlabeled samples. In the following sections we describe the classifiers used in this project.

Naive Bayes (NB) is a supervised learning classifier that uses the Bayes' theorem and assumes that the values of each feature are completely independent from one another. The Bayes classifier can be used for binary and multiclass classification [6].

Support Vector Machine (SVM) mainly aims at detecting the hyperplane which generates the largest boundary between samples in the dataset. SVM is a powerful classifier, as it can be used with different kernels (linear, polynomial, sigmoid, and RBF). This, combined with the "kernel trick," allows for efficient identification of relationships in high dimension space.

Random Forest (RF) extends the decision tree model. It works perfectly on multiclass classification problems in that it selects the best result made by a variety of decision trees. Random forests usually produce highly accurate and interpretable results.

K Nearest Neighbor (KNN) works well on non-linear datasets. It predicts the class of the new observations based on the company they keep. Samples are labeled based on the identity of the 'k' nearest neighbors (where k is a natural number). Distance can be measured in several ways, such as the Euclidean distance and the Mahalanobis distance.

Hyperparameters are the parameters whose values are used to determine the behaviours of a classifier. For instance, we can use the Support Vector Machine with different Kernels. As a further example, a hyperparameter is used to set the number of neighbouring samples to be considered when using a KNN classifier. Classifiers can be used with the default parameters or with a modified version of them. Hyperparameter tweaking, using methods such as Grid Search, can improve the effectiveness of classification models.

2.6 Classifier Evaluation

Classifier evaluation techniques are used to ensure that a classification model is producing accurate results, that these results will be reproduceable, and that the classifier is not overfitting the data. The methods used in this work are described below.

K-fold cross validation evaluates the anticipated performance of the model. First, split the dataset into K equal subsets. One subset is selected to validate the model, while the remaining subsets are used to train the model. This process continues until all the subsets are used to validate the model.

Confusion Matrix is used to compare a model's predictions to the actual identities of the training/test population. Metrics used to evaluate model performance such as accuracy, precision, false alarm rate, Negative predicted value, Specificity, Sensitivity and Accuracy can be calculated using the matrix.

		Actual	
		+	-
Predicted	+	TPs	FPs
	-	FNs	TNs

- **True Positive:** the patient's RNA profile is correctly classified as positive prostate cancer instance.
- **False Positive:** the patient's RNA profile is wrongly classified as a prostate cancer positive instance.
- **True Negative:** the patient's RNA profile is correctly classified as negative prostate cancer instance.
- **False Negative:** the patient's RNA profile is wrongly classified as negative prostate cancer instance.

Predictive Value (PPV)or Precision:

A measure of the trustworthiness of positive prostate cancer classification. Precision is calculated as

$$PPV = \frac{TP}{TP + FP}$$

Negative Predicted Values (NPV):

A measure of the trustworthiness of negative prostate cancer classification. NPV is calculated as

$$NPV = \frac{TN}{TN + FN}$$

Specificity: A measure for the model's ability to generate negative results for patients who do not have prostate cancer. It is calculated as

$$\text{Specificity} = \frac{TN}{TN + FP}$$

Sensitivity: A measure to check how often the model generates positive results for patients who have prostate cancer. It is calculated as

$$\text{Sensitivity} = \frac{TP}{TP + FN}$$

Accuracy: A measure of the model's ability to distinguish between labels. It is calculated as

$$\text{Accuracy} = \frac{TP + TN}{numbr\ of\ samples}$$

2.7 Interpretability

The interpretability of a Machine Learning (ML) system is defined as the degree to which a cause and effect can be identified within the system. In other words, can we easily and fully understand how the model derives its predictions. Interpretability is less important when the goal is strictly to make a classification based on a set of input features. If we wish to understand the relationship between the features and the dependant variable, however, interpretability is critical.

2.8 Multi-Class Classification

The problem of classifying input variable instances into more than two classes is known as multiclass classification. In this case, we are classifying samples into five different classes based on the Gleason score provided in the training dataset. There are several strategies for

handling multiclass problems, such as “one versus rest” and “one versus all.” Different classifiers require different approaches. Fortunately, the Scikit Learn library handles these distinctions internally in most cases.

2.9 Feature Selection

We reduce the number of input variables through a process known as feature selection. This can be important for several reasons: increasing model interpretability, improving runtime, and increasing accuracy. We apply statistical methods to select the input variables in data = $\{x1, x2, \ldots, xn\}$ that contribute the most to the models training process. There are a variety of feature selection techniques available to scientists. Filter methods like chi-Square, Wrapper methods like forward and backward selections and Embedding methods like random forest feature selection are just a few examples. There are various methods that can be used for feature selection. Here, we will define the ones that we used in our study.

C**hi-square** also known as, Chi-2, is one statistical measure used in feature selection. Chi-2 quantifies the degree of relatedness between each feature, individually, and the dependant variable. It is a popular as a first layer of feature selection because it can be applied quickly [6].

Standard Scaler(Normalization) is a method that is used in machine learning algorithms to scale the numerical values into a standard range [7]. The normalization technique is one of the scaling approaches used with Standard Scaler. The normalizer scales each floating-point value of the input variables into a range, based on the normal distribution, to get the most precision.

Random Forest is a powerful method for selecting features. It takes advantage of decision trees' strengths by employing a group of them to rank features based on how well they can be used to distinguish members of different classes [8]. For our project, we chose random forest feature selector because it is extremely accurate and interpretable. This is critical in the classification of prostate cancer because it is crucial to understand which genes are linked to the severity of the disease.

Dimensionality Reduction reduces the feature space by using linear combinations of the features in place of the original features. This method can improve both accuracy and runtime dramatically. Because the original features are lost, however, it often removes or reduces our ability to link individual features to the target variable.

Linear Discriminant Analysis (LDA) is a tool used for both dimensionality reduction and classification. LDA generates linear combinations of features that best preserve the characteristics of the original features [6].

2.10 Class Imbalance

Class imbalance occurs when one class is overrepresented compared to the other(s). This is problematic because underrepresented classes may contain significant values that the classifier may overlook during the prediction process. There are several strategies for dealing with this problem, such as oversampling of the minority groups and undersampling of the majority groups to balance the classes. The oversampling technique we apply in this work is SMOTE, which stands for Synthetic Minority Over-sampling Technique. We apply an edited nearest neighbor (ENN) technique for undersampling.

SMOTE generates new members of the under-represented class by generating synthetic data points between the minority sample and its ‘k’ nearest neighbors [9]. This technique is thought to be superior to simply creating copies of the minority samples because new information is generated.

ENN removes members of the over-represented class by removing samples whose identity differs from that of their ‘k’ nearest neighbors. This not only reduces the class imbalance but also helps to sharpen the boundaries between classes [10].

3. APPROACH

In this order, our project went through four stages: examining the dataset, dataset preprocessing, feature selection, testing the classifiers with a variety of hyperparameters and finally, applying dimensionality reduction after achieving 98 percent accuracy.

3.1 Dataset Examination

An Excel Worksheet was used to store the raw data. To get started, we had to save it as a comma-separated values (CSV) file. Then, we noticed that the patient's ID row

appears twice in the table, once at the top and once at the bottom. As a result, the data loading function treated all the values as if they were strings. We also noticed that some of the rows in the dataset have only zero values. Including these values will only slow down the model and will have no effect on the classification process. Furthermore, each row in the dataset represented the characteristics of a single exon for different patients. Therefore, the original dataset had to be transposed so that the rows of the original file became the features columns (a requirement for proper processing using Scikit Learn). Quick analysis also revealed a class imbalance.

3.2 Data Preprocessing

Primarily, we had to delete the patient's ID row from the bottom of the spread sheet. This solution helped the loading function to recognise the values as floats. Then, using the Pandas library, we created the function (*load_and_process_data*) to load data from a CSV file saved locally. The function first loads the data from the CSV into memory and then transposes the resulting Dataframe. After the transformation, the original dataset's rows become the features columns. The function also removes any features with zero values and adds a new target variable (*GRADE_GROUP*) as a dependent variable, if desired. The *GRADE_GROUP* variable can be one of five groups based on the Gleason score of the patient. Then, the function initialises the X, the feature matrix, and y, the target variable matrix. The target variable can also be adjusted by the user (Gleason Score, Grade Group, Primary Gleason Grade, Secondary Gleason Grade).

3.3 Feature Selection Methods

As mentioned previously, various methods can be used for the selection of features. We used a combination of theory and empirical testing to develop an effective feature selection pipeline. We apply several feature selection methods. We begin with Chi-2, and reduce the feature space from roughly 57 000 to 15 000. We then normalize the data using a Standard Scaler and further reduce the feature space using a Random Forest and Gini impurity. We run the classifiers at this point and then apply dimensionality reduction, via Linear Discriminant Analysis, to further improve accuracy (although this distorts the relationships between the individual features and the target).

3.4 Class Imbalance

To deal with the class imbalance problem, we apply both oversampling and undersampling techniques.. To increase the number of samples in the underrepresented group, the oversampling technique adds copies of the class that are different than the existing samples. Additionally, using the undersampling technique eliminates instances of overrepresented classes which are most unlike the other members of their class.

3.5 Classifiers and Hyperparameters

To achieve the highest accuracy, we made an educated choice of the classifiers on which we tested our model. The dataset is not linear. As such, we avoided classifiers that work best with linear data, such as LDA and SVMs with linear kernels. We instead chose classifiers that worked well with datasets that are not linearly separable. Additionally, even though Neural Networks produce extremely accurate results, we avoided them because they can be overly complex, slow to run, and difficult to interpret. Knowing the precise chain of reasoning behind the predicted Gleason score is an important part of the diagnosis and prognosis of the disease in this model, and it cannot be compromised. Thus, based on both theory and experience, we narrowed our choice down to four classifiers:

- **Naive Bayes (NB):** We used naïve Bayes with the Gaussian Distribution statistical model (GaussianNB). Our choice was based on the distribution of the input variables in the dataset after feature selection and resampling.
- **Support Vector Machine (SVM):** We used SVM with the default Radial Basis Function (RFB) kernel, as the dataset is not linearly separable. RBF acts as a weighted nearest neighbour model. As a result, the classification of the new values is heavily influenced by the nearest neighbours. The neighbours are all

grouped together in this dataset. Thus, the SVM with RBF Kernel achieved high accuracy.

- **Random Forest:** The accuracy and interpretability of the predictions are crucial in this experiment. Thus, we chose this classifier as it suits the main objective of the project.
- **K Nearest Neighbor:** We used KNN with the default Euclidean Distance function, however we modified the default number of neighbours by finding the best K value. Prior to performing classification we determine the best K value, in a range from 1 to 20, based on accuracy.

4. EXPERIMENTAL SETUP

Our experiments were performed using the Prostate Adenocarcinoma (TCGA, Firehose Legacy) dataset obtained through cBioPortal. The columns, which each represent a patient (patient IDs are found in row 1 and begin with TCGA), are used as the samples. The first 484 rows, which represent RNA exon expression levels for roughly 60,000 genes, serve as the features (independent variables) for our model. The last five rows can each be used as the target/dependant variable. As our target for classification in this dataset, we chose the Gleason Score, which ranges from 6 to 10 in this dataset.

We designed our classifier using Python 3.7 and the following libraries:

- Numpy (Numpy arrays)
- Pandas (Dataframes)
- Scikit Learn 0.24.1 (Machine Learning Models)
- Imbalanced-Learn 0.8.0 (SMOTE / ENN)

We used PyCharm and Spyder/Anaconda for our integrated development environments. Our code is publicly available through GitHub at this link. Our driver program is **main.py**. Wrappers functions for the major components of our classification pipeline (loading the data, feature selection, class rebalancing, dimensionality reduction and classification) are in **pipeline.py**.

Classifier accuracy was determined using 8-fold cross validation.

5. RESULTS AND DISCUSSION

5.1 Results

Our classification model can accurately determine the Gleason Score of samples taken from the "prad_tcga_genes" dataset. After the data had undergone dimensionality reduction, the highest classification accuracy of 99.8 percent was achieved by Naïve Bayes (NB). The rest of the classifiers achieved above 99 percent accuracy. These results can be found in appendix A, Figure 2, and Table 1. We also measured accuracy using data that had not undergone dimensionality reduction – since this process can distort the effect of individual features. The best accuracy achieved without dimensionality reduction was 99.31 percent, which was achieved using a KNN model. Other classifiers also performed well, with the SVM model achieving 97.47 percent accuracy, the RF model achieving 96.32 percent accuracy and the NB model achieving 89.66 percent accuracy. These results are captured in appendix A, Figure 1, and Table 2.

Table 1: Classification with Dimensionality Reduction – Average Accuracy and other Performance Measures (%)

	Accuracy	Sensitivity	Specificity	PPV	NPV
NB	**99.78**	99.83	99.94	99.82	99.94
RF	99.77	99.82	99.94	99.83	99.94
SVM	99.77	99.82	99.94	99.83	99.94
KNN	99.77	99.82	99.94	99.83	99.94

Table 2: Classification without Dimensionality Reduction – Average Accuracy and other Performance Measures (%)

	Accuracy	Sensitivity	Specificity	PPV	NPV
KNN	**99.31**	95.92	99.85	99.43	99.85
SVM	97.47	86.59	99.43	95.88	99.43
RF	96.32	80.47	99.18	96.95	99.18
NB	89.66	79.3	97.41	91.22	97.41

The results were multiplied by 100 and rounded up to two decimal places. They are listed in increasing order and the best results are highlighted.

We also identified three exon IDs that seem to be especially co-related with Gleason Score:

1. ENSG00000124233.11

2. ENSG00000202198.1
3. ENSG00000124157.6

5.2 Discussion

Our goal was to determine the relationship between the gene / RNA profiles and the Gleason Scores of the 494 individuals belonging to the "prad_tcga_genes" dataset. Understanding this connection could lead to a classification model able to identify the Gleason Score of a new, unlabelled sample. The Gleason Scores found in this dataset range from 6 to 10.

When first presented with the dataset, we struggled to achieve classification accuracies exceeding 50 percent. We attempted performing several layers of feature selection, including passing the data through a Chi2 filter, selecting features based on Gini impurity using a random forest and some forward / backwards search methods. Unfortunately, depending on the number of features retained after passing the data through a Chi2 filter, search methods either took an unreasonable amount of time or failed to improve accuracy for any of the models listed in section 5.1.

We studied the distribution of the sample among each of Gleason Scores and identified a class imbalance. As per appendix A, figure 3, 246 of the 494 samples have a Gleason Score of 7, while only 4 samples have a Gleason Score of 10. The other scores were more evenly distributed, however a score of 9 was still significantly more common than a score of 6 or 8. To address the class imbalance we applied a combination of oversampling and undersampling. The oversampling technique we applied is called Synthetic Minority Over-Sampling Technique or SMOTE. SMOTE generates additional copies of the minority class that are distinct from the existing samples, which will often aid the classification process more than simply adding copies of existing samples. The undersampling technique we applied is called Edited Nearest Neighbor or ENN. ENN reduces the number of samples in the majority class by eliminating a member whose class label differs from the class of at least half of its k nearest neighbors. Thus, the class imbalance is addressed, and the most ambiguous members of the majority class are eliminated. After applying both SMOTE and ENN to the data the class balance was improved (see appendix A, figure 3), and the classifiers began achieving much greater accuracies.

Our final classification pipeline performed the following steps:

1. Eliminate features whose values were the same across all members of the sample population.
2. Apply a Chi2 filter to reduce the number of features by roughly ¾ (most of our test runs use 15000).
3. Transform the data so that it fits a normal distribution.
4. Perform a second round of feature selection using Gini impurity (this generally left less than 5000 features).
5. Apply undersampling and oversampling.
6. Use Linear Discriminant Analysis (LDA) to perform dimensionality reduction (optional)
7. Perform classification using the classifiers outlined in section 5.1

This series of steps led to classification that was both highly accurate and reasonably quick. Each of the classifiers tested achieved almost perfect accuracy after we applied dimensionality reduction. Without dimensionality reduction the Naïve Bayes model achieves an accuracy just under 90 percent, while the other models each achieve accuracies over 95 percent.

All models struggled the most with classifying members of group 7 and, to a lesser extent group 8. Members of these groups probably have a mix of low- and high-grade regions leading them to have characteristics similar to the other groups. Examination of the confusion matrices for classifiers without dimensionality reduction shows that members of the class with Gleason Score 7 are often classified as belonging to the groups with a Gleason Score of 8 or less than or equal to 6. The behavior of the ENN undersampling provides further support for this theory. The ENN process drastically reduces the number of samples with a Gleason Score of 7, suggesting that

samples in this group are often more like members of other groups than their own.

We note that the best value for K was 1 (a chart is provided in appendix A). This may occur because of the ENN undersampling performed.

We attempted additional strategies, such as eliminating outliers using an isolation forest, but the additional techniques that we applied increased the time required for classification and did not improve the accuracy.

We were also able to identify three exons that seem to be highly corelated with a samples Gleason Score: ENSG00000124233.11, ENSG00000202198.1, and ENSG00000124157.6. These exons were identified when applying the initial feature selection filter (Chi2). A chart showing the top 10 features by Chi2 score can be found in appendix A (figure 4).

We considered classifying samples based on their Grade Group, which was defined by the International Society of Urological Pathology in 2014. Under this system a Gleason Score of 6 or less is considered group 1, a Gleason Score of 7 can be considered either group 2 or group 3 depending on scores at the primary and secondary site, a Gleason Score of 8 is considered group 4 and a Gleason Score of 9 or 10 is considered group 5. The classifier was unable to accurately classify samples using these categories. Namely, all classifiers struggled to classify members of groups 2 and 5. Additionally, our oversampling and undersampling technique exaggerated the class imbalance, rather than mitigating it. We hypothesize that this may have occurred due to similarities between grade groups – such as groups 2 and 3. Further investigation is required and falls outside of the scope of this study.

There is also concern that our classifier may be overfitting the data. The SMOTE/ENN technique used to address the class imbalance alters the data and increases the sample size significantly. This may lead to a classifier that is well suited for identifying members of each class within this dataset but struggles to achieve the same results when presented with new, unlabeled samples.

6. CONCLUSION

Our results show that exon expression levels can be used to accurately predict the Gleason Scores which we might expect from a biopsy performed on a person with those expression levels. As Gleason Score is correlated with Cancer risk and severity, this study shows that tests of exon expression, which are significantly less invasive than biopsies, may serve as accurate predictors of prostate cancer. Further work is needed to ascertain the generalizability of these results.

7. FUTURE WORKS

Even though this model produces good results, we cannot know, at this point, whether they will be generalizable. As we mentioned in the results section, there is some concern that the model may be overfitting the data. This work should be validated using fresh samples that were not part of the dataset used to train the model.

We would also like to better understand why the model performed well when classifying the data by Gleason Score, but failed to produce similar results when we attempted classification using the Grade Group. A study examining how gene expression varies based on the Gleason Pattern found at the primary site versus the secondary site may help to explain the differences in classifier performance. Finally, we wondered how trustworthy the results would be in a few years. What if machine learning becomes the norm for diagnosing prostate cancer in the future? Will there be less biopsies as a result? If that is the case, how can we obtain more data to train the algorithm in the future? These are just some of the topics to consider and study.

8. ACKNOWLEDGEMENT

We would like to thank scikit-learn for assisting us with the implementation of all the techniques used in this project. We would also like to thank Dr. Luis Rueda for conceiving the project's idea and providing the necessary datasets.

9. REFERENCES

[1] M. C. Staff, "Prostate cancer Health Information," mayoclinic.org, 12 11 2020.

[Online]. Available: https://www.mayoclinic.org/diseases-conditions/prostate-cancer/diagnosis-treatment/drc-20353093. [Accessed 08 04 2021].

[2] "TCGA's Study of Prostate Carcinoma," National Cancer Institute, [Online]. Available: https://www.cancer.gov/about-nci/organization/ccg/research/structural-genomics/tcga/studied-cancers/prostate. [Accessed 08 04 2021].

[3] "Prostate Adenocarcinoma," cBioPortal For Cancer Genomics, [Online]. Available: https://www.cbioportal.org/study/summary?id=prad_tcga. [Accessed 08 04 2021].

[4] V. B. P. Ames, "Stereotactically guided breast biopsy: a review. Insights Imaging," Insights Into Imaging, vol. 2, pp. 171-176, 2011.

[5] P. A. Humphrey, "Gleason grading and prognostic factors in carcinoma of the prostate," Modern Pathology, vol. 17, no. 3, pp. 292-306, 2004.

[6] D. L. Rueda, COMP-4740 Advanced Selected Topcs in Machine Learning, Windsor, Ontario: PowerPoint Slides, 2021.

[7] J. Brownlee, "How to Use StandardScaler and MinMaxScaler Transforms," Machine Learning Mastery, 10 June 2020. [Online]. Available: https://machinelearningmastery.com/standardscaler-and-minmaxscaler-transforms-in-python/. [Accessed 8 April 20201].

[8] A. Dubey, "Feature Selection Using Random forest," Towards Data Science , 14 December 2018. [Online]. Available: https://towardsdatascience.com/feature-selection-using-random-forest-26d7b747597f. [Accessed 08 April 2021].

[9] K. W. B. L. O. H. W. P. K. Nitesh V. Chawla, "SMOTE: Synthetic Minority Over-sampling Technique," Journal of Artificial Inteligence Research, vol. 16, pp. 321-357, 2002.

[10] D. Wilson, "Asymptotic Properties of Nearest Neighbor Rules Using Edited Data," IEEE Transactions on Systems, Man, and Cybernetics , Vols. SMC-2, no. 3, pp. 408-421, 1972.

10. APPENDICES

10.1 Appendix A - Results

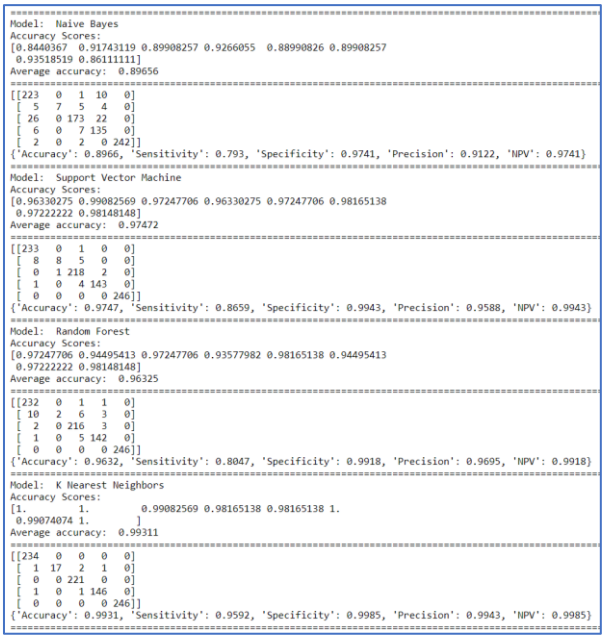

Figure 1 – Model Accuracies

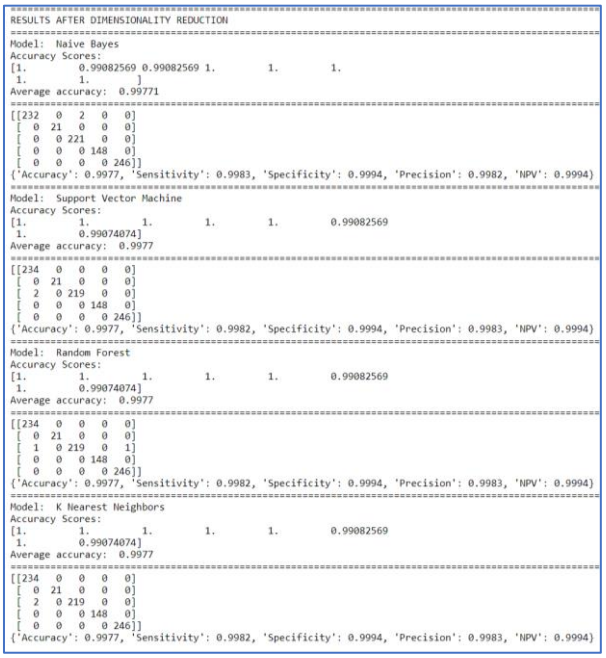

Figure 2 – Accuracies with Dimensionality Reduction

```
DATA PROFILE -  INITIAL
------------
Number of samples:  494
Number of features:  57465
Class Distribution:
Counter({7.0: 246, 9.0: 137, 8.0: 63, 6.0: 44, 10.0: 4})
================================================================================
================================================================================
DATA PROFILE -  IMBALANCE ADJUSTED
------------
Number of samples:  847
Number of features:  4106
Class Distribution:
Counter({10.0: 246, 6.0: 237, 8.0: 218, 9.0: 125, 7.0: 21})
```

Figure 3 – Class Distribution Before/After SMOTE/ENN

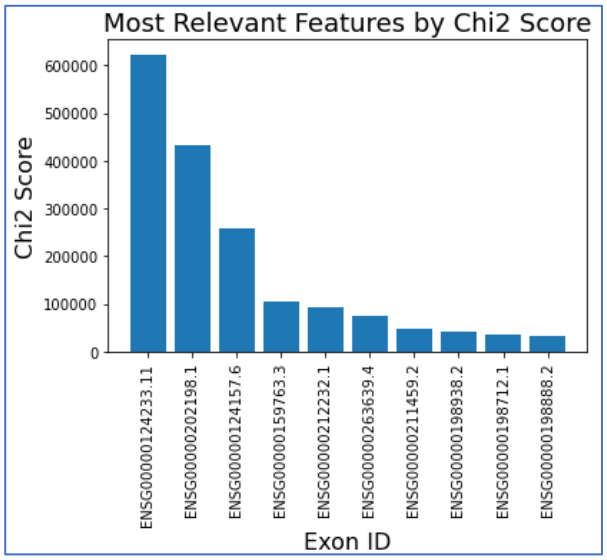

Figure 4 - Most Relevant Features

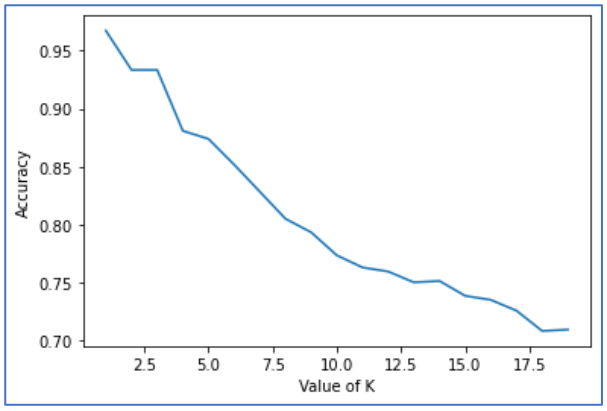

Figure 5 - Best K

10.2 Appendix B – Figures

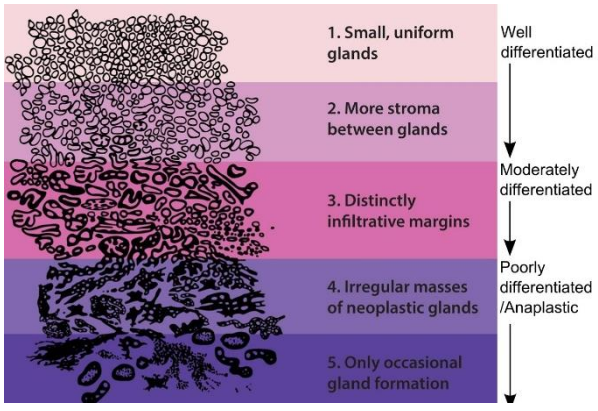

Figure 6 – Gleason Pattern Recognition

10.3 Appendix C – Main Program

```python
import pipeline as pl

SEED = 42       # The random seed used
K_BEST = 15000  # The number of features to select using Chi2 filter
N_DIM = 3       # The number of dimensions (dimensionality reduction)
N_FOLDS = 8     # The number of folds to use in cross-fold validation

# Load and process the data and return the feature and target matrices -
X, y = pl.load_and_process_data('prad_tcga_genes.csv', 'GLEASON_SCORE')

# Display the data profile ----------------------------------------------
pl.data_profile(X, y, 'INITIAL')

# Perform feature selection ---------------------------------------------
X_select = pl.feature_selection(X, y, K_BEST)

# Perform up-sampling and down-sampling to address class imbalance -----
X_balanced, y_balanced = pl.fix_imbalance(X_select, y, SEED)

# Display the updated data profile --------------------------------------
pl.data_profile(X_balanced, y_balanced, 'IMBALANCE ADJUSTED')

# Perform dimensionality reduction (optional) ---------------------------
X_reduced = pl.reduce_dimensions(X_balanced, y_balanced, N_DIM)

# Perform classification ------------------------------------------------
model_names = ['Naive Bayes',
               'Support Vector Machine',
               'Random Forest',
               'K Nearest Neighbors']
n_models = len(model_names)
scores = []
averages = []
models = []
confusions = []
metrics = []
for i in range(n_models):
    m, s, a, c, z = pl.classify(X_balanced, y_balanced,
                                model_names[i], N_FOLDS)
    models.append(m)
    scores.append(s)
    averages.append(a)
    confusions.append(c)
    metrics.append(z)

# Display results -------------------------------------------------------
for i in range(n_models):
    pl.display_results(model_names[i], scores[i], averages[i])
    print(confusions[i])
    print(metrics[i])

# Perform classification after dimensionality reduction ----------------
pl.print_divider()
pl.print_divider()
print("RESULTS AFTER DIMENSIONALITY REDUCTION")
model_names = ['Naive Bayes',
               'Support Vector Machine',
               'Random Forest',
               'K Nearest Neighbors']
n_models = len(model_names)
scores = []
averages = []
models = []
confusions = []
metrics = []
for i in range(n_models):
    m, s, a, c, z= pl.classify(X_reduced, y_balanced,
                               model_names[i], N_FOLDS)
    models.append(m)
    scores.append(s)
    averages.append(a)
    confusions.append(c)
    metrics.append(z)

# Display results -------------------------------------------------------
for i in range(n_models):
    pl.display_results(model_names[i], scores[i], averages[i])
    print(confusions[i])
    print(metrics[i])
```

Figure 7 - Main Program

10.4 Appendix D – Github Link

Github Repository:

https://github.com/LamaKha/COMP4740_Prostate_Cancer_Detection

Git Cloning:

https://github.com/LamaKha/COMP4740_Prostate_Cancer_Detection.git