A Comparative Analysis to Predict p53 Activity Using Classification Models

by

Priyanka V. Setty

BE Biotechnology, P.E.S Institute of Technology, India, 2017

Submitted to the Graduate Faculty of

the Department of Biostatistics

Graduate School of Public Health in partial fulfillment

of the requirements for the degree of

Master of Science

University of Pittsburgh

2019

UNIVERSITY OF PITTSBURGH

GRADUATE SCHOOL OF PUBLIC HEALTH

This thesis was presented

by

Priyanka V. Setty

It was defended on

July 29, 2019

and approved by

A Comparative Analysis to Predict p53 Activity Using Classification Models

Priyanka V. Setty, MS

University of Pittsburgh, 2019

Abstract

Mutation studies of *TP53*, the gene coding the tumor protein p53, have become increasingly common in cancer research to understand its structural changes and its implications for tumor suppression. The protein's structure is built with four identical chains containing 393 amino acids per chain. This homo-tetrameric configuration of p53 plays an important role in suppressing tumors and it is important to understand the structure-function dynamics and their role in cancer development.

A p53 mutant dataset was obtained from the University of California at Irvine (UCI) Machine Learning Repository to infer p53 protein's ability to suppress tumors based on its two-dimensional (2D) and three-dimensional (3D) structural features. The dataset consisted of 31,283 instances (observations) and 5,408 numerical features. Among the total features, the first 4,826 accounted for 2D structural features which were based on electrostatic and surface properties. The remaining 582 3D features were the distance maps between mutant and wild type p53. After selecting a subset of the features that were statistically relevant in predicting the outcome (n=100), three classification algorithms, Logistic Regression (LR), Support Vector Machine (SVM) and Random Forest (RF), were fit to the data and trained using a cross-validation scheme to obtain good parameters to classify an active p53 mutant from its inactive counterparts. Performance metrics in terms of accuracy and area-under-the-curve (AUC) were utilized in order to evaluate a particular classification model. Among the three different algorithms used to predict the outcome, LR seemed to outperform SVM and RF with an accuracy ranging from 0.75 to 0.81 and AUC ranging from 0.75 to 0.88.

The LR model identified 2D feature numbers 60,74,49,40, and 73 as features of high importance in predicting the activity of p53. The public health significance of this study is that it advances the understanding of p53, which is critical to cancer tumor suppression, by helping to predict p53 activation using set of structural features obtained from simple classification models.

Table of Contents

List of Tables

List of Figures

Preface

I would like to thank my thesis advisors, Dr. Jeanine Buchanich and Dr. Arvind Ramanathan for all the support, guidance, and encouragement throughout this endeavor. Dr. Jeanine Buchanich's constant replies and corrections helped me get the work done on time. She has been a wonderful advisor throughout my master's journey. Dr. Arvind Ramanathan's guidance over the past year has not only helped me understand and utilize the research tools I needed for this study, but also better prepared me for further studies, and helped me develop research and organizational skills that will continue to benefit my future work as well.

I would like to thank Drs. Jenna Carlson and John R. Shaffer for agreeing to serve on my thesis committee on a short notice and also for their feedback during this process. Beyond giving me further insights on my analysis, their comments and advice also helped me to better present my research. I also want to thank all the faculty in the Biostatistics department at the University of Pittsburgh, who have helped me continuously over the past two years. Their guidance has been invaluable during my time as a master's student and will continue to be so throughout my career.

I would like to thank my family, particularly my parents (Varadaraj Setty and Dr. Savitha V. Setty) and grandparents (K. V. Nagaraj, Leela Raj, B. N Narayana Setty and Gangalakshmamma) , who have always believed in me more than I do. They have always been a source of inspiration when I am low on confidence. Without their blessings and constant encouragement I wouldn't have completed my journey as a Master's student.

Finally, I would like to thank my friends Akash Parvatikar, Shashank B.R, Avantika Srivastava and Kushal Ganesh without whom this journey would be incomplete. In particular

Akash helped me understand concepts concerned with my study and also his help had a significant contribution towards finishing my work on time.

1.0 Introduction

Uncontrolled cell division that proliferates into surrounding tissues is termed as cancer. According to the National Cancer Institute, approximately, 1,735,350 new cancer cases were anticipated in the year 2018 in the United States, making it a significant public health problem [1]. There are several factors that can cause uncontrolled cell division, and one which is widely studied is the mutation in gene *TP53* that codes for protein p53 [2]–[4]. The p53 protein is a tumor suppressor which represses uncontrolled cell division. But, when there is a mutation in *TP53* gene, which might lead to an amino acid substitution in the protein p53, it results in the protein losing its tumor suppressor nature [2]. This leads to uncontrolled cell division thus leading to cancer.

p53, which has been widely recognized as "the guardian of the genome", plays a key role in suppressing tumor formation [5]. In the dataset that we have chosen for our study, the structural features of the protein p53 were used to predict the active or inactive status of the protein. Here, active refers to the "on" state, whereas inactive refers to the "off" state. A protein in the active state can function to repress the formation of tumor. However, an inactive protein loses its function to repress tumor formation. Due to this relationship between the state of the protein and its activity to suppress tumors, predicting the tumor suppression can be linked to predicting the state of p53 protein and thus, the goal is reduced to find if this protein is in active or inactive state. The prediction of states, active and inactive, is linked to the biological function of protein p53 (to suppress or not) and can be considered as a classification problem. Classification is a technique to predict the category of a sample based on its inherent properties and function. The algorithm tries to model the relationship between the input and outcome variables. In our study, the outcome variable has two classes: active (1) and inactive (0), and the input variables are the structural features of p53 protein. Prior to finding the state of protein, it is important to understand the structure of protein and its role in cancer development which is discussed in the following sections.

1.1 Structure of p53 Protein

A protein is essentially made up of amino acids which are organic compounds containing both carboxyl (-COOH) and amine group ($-NH_2$) along with functional groups (-R) that determine its function. Each amino acid is further made up of nucleotides which are organic compounds that are building blocks of DNA (Deoxyribo-Nucleic Acid) and RNA (Ribo-Nucleic Acid). According to "Central Dogma of Molecular Biology", DNA makes RNA and RNA makes protein. There are 20 different amino acids, and each is coded by a stretch of DNA made up of three nucleotides which is known as a codon. DNA is made of 4 unique nucleotides, Adenine (A), Guanine (G), Cytosine (C) and Thymine (T). Out of the 4 nucleotides, 3 nucleotides in a particular orientation results in an amino acid and specific arrangement of different amino acids results in a protein. For example, TTC configuration of the nucleotides results in an amino acid called lysine which is basic (i.e., $pH > 7$) in nature. If TTC changes to TGC, that is, if the thymine in position two is replaced with a guanine, it results in a neutral ($pH = 7$) amino acid Threonine. This process of changing an amino acid in a particular position of protein due to replacement of different nucleotides in the codon is termed as "point mutation". During the occurrence of a point mutation, a protein's structure undergoes changes which in turn changes its functional properties [6].

In our study, p53 is the protein of interest. p53 is a homotetramer (four identical protein chains) with 393 amino acid residues in each chain. The protein is made up of five functional domains, (i) Transactivation domain (TAD) which is further sub-divided as TAD1 and TAD2, (ii)

Proline-rich region (PRR), (iii) DNA binding region or the core domain, (iv) Tetramerization domain (OD) and (v) Regulatory domain at the extreme carboxyl terminal (CTD) [7]. The domain that we are interested in studying further is the DNA binding region, as it has been shown that most of the cancer mutations are seen in this region [8]. Figure 1 (below) shows the binding of DNA to the core domain/DNA binding domain of the protein p53.

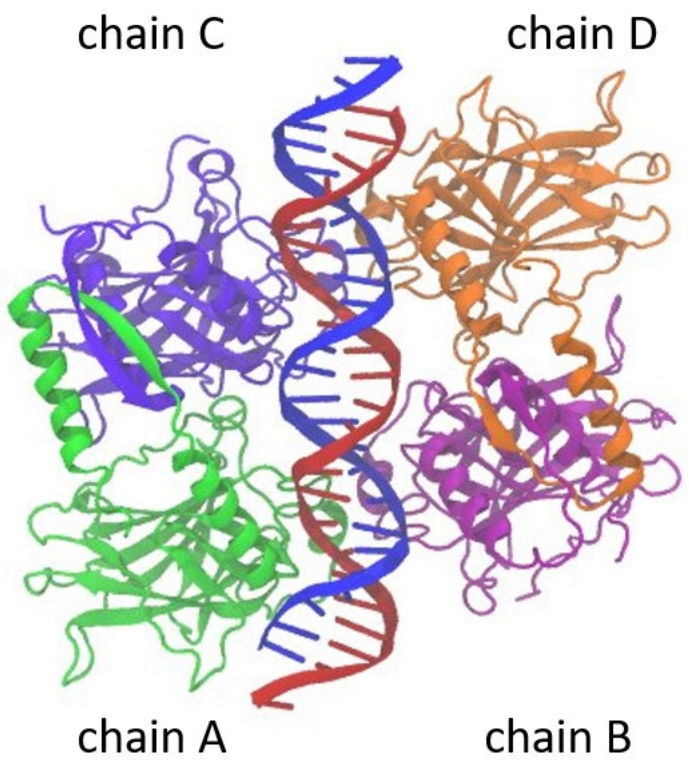


Figure 1. Illustrates the core domain/DNA binding domain in chains A,B,C and D of p53 protein interacting with DNA molecule

1.1.1 Structure of DNA-Binding Region of p53

Among the 393 residues in each chain of p53 protein, the DNA binding site is made up of 197 residues which span from 94th residue to 292nd residue. The DNA binding surface is made up of loop-sheet-helix motif and two large loops (L2 and L3). When the secondary structures of proteins like loop, sheet and helix are arranged in a particular geometric arrangement it forms different motif or super-secondary structures. The structure of L2 and L3 are stabilized by amino acids, cysteine in 176th , 238th , 242nd positions and histidine at position 179 in all four chains of p53 protein (tetrahedrally) [5]–[7]. The DNA binding domain of p53 has been used to extract the 2D and 3D features in the dataset that has been used in this study.

1.2 Surface Maps of Proteins

Surface maps help in mapping a protein's three-dimensional (3D) structure. In order to obtain the surface/structural features of a protein, its 3D structure is projected onto a plane [9]. During the occurrence of point mutation, a protein's structure undergoes changes which in turn changes its functional properties [10]. In practical scenarios, calculating surface maps for both mutant and wild type and then subtracting one from another (*wild type* - *mutant*) yields useful results that can help explain the effect of point mutation [9]. Due to this structure-function inter-relationship, the 2D features obtained from surface maps in UCI dataset will help in predicting the p53 mutant's activity.

1.3 Distance Maps of Proteins

Distance maps help in mapping distances between amino acid residues in a protein. This distance map is an N × N matrix, where N is the number of amino acids, and the matrix is filled with distances between corresponding residues. As a result of point mutation (explained in the previous section) the distance between pairs of amino acids and position where the mutation has occurred also changes [9], [11]. Thus, the changes in protein's structure induced by mutations can be deduced by looking at the 3D distance map. This provides another useful direction to infer a protein's activity status based on the distances between their corresponding amino acids.

1.4 2D Features

The 2D features for each mutant protein were obtained through homology modeling, a technique which is prominent in obtaining required structural models that are useful for inferring a protein's function [3], [4]. The wild type p53 was substituted with the mutant amino acid and this configuration was used to simulate the structures of mutant p53 protein. These structural features were further extracted from the corresponding mutant model [2]. The extracted features were labelled as electrostatic or h-bond acceptor/donor using AMBER 6 [12]. The steric (structural changes between the mutant p53 and the wild type p53) and depth (distance between the amino acids of mutant and wild type p53) information was obtained by mapping the molecular surface onto a sphere which was then mapped to a plane. The surface map (2D representation of the 3D surface) that resulted from this mapping was subtracted from the wild type map, hence obtaining 2D features in the dataset [2], [4], [9].

1.5 3D Features

The 3D features were calculated by taking the difference between mutant and wild type p53 distance maps (distance between all amino acid residue pairs). The core domain of the protein has 197 residues which resulted in a 197-dimensional square matrix that was collapsed to get a distance vector. This 197-length vector map showed three features of each residue (i.e., i, j, k directional vectors). Out of the 591 features that were obtained for a single mutant, only 582 significant features were retained [2]–[4]. The 3D features symbolize the changes in the magnitude of 3D distance between the mutant and the wild type p53.

1.6 Previous Work

Danziger and collaborators have worked on finding useful p53 protein mutants to reveal important drug targets that would restore its activity [11]. In doing so, they have worked with 123 already known putative cancer rescue mutants of p53 among which 52 were active (suppressor mutant: suppresses cancer) and 71 were inactive (cancer mutant: does not suppress cancer). Using this prior knowledge, they used Bayesian statistics to infer if the protein was an active or inactive mutant. For this work, they have used 1D sequence information, 2D surface details, 3D structure data, and 4D trajectory information-based features which were extracted from these mutant proteins to construct a Support Vector Machine classifier. This model was tested on 71 newly found p53 mutants using a double-blinded prediction study. An overall accuracy of 74% was achieved using 1D features alone. 2D and 3D features achieved an accuracy of 64.2% and 73.2% respectively. 4D features was able to obtain an accuracy of 47.2% [11].

In a subsequent study, Lathrop et al. used several active learning methods to explore p53 cancer rescue mutants and also compared the different methods [4]. Active learning is a technique for building a classifier that uses unlabeled examples (unknown activity state of mutant) in an iterative fashion. In this study, a total of 204 mutants were used in the training set and 57 putative p53 cancer rescue mutant which was not previously classified (active vs inactive) was used as a test set. They used 10 different active learning methods and 3 control types, out of which, Maximum Curiosity performed best overall with an accuracy of around 77%. This method results in a high correlation coefficient when the mutant is correctly paired with its activity status [4].

In another study by Danziger et. al, they extended the active learning method by developing a novel method known as *Most Informative Positive* (MIP) to find mutations in p53 that represses tumor formation [2]. Here, the novelty achieved was with respect to a faster computational time. MIP was able to discover active p53 mutants in-silico by carrying out 33% fewer experiments in comparison to the non-MIP approach which saved computational time. However, not much difference in the accuracies of active learning models using traditional method (non-MIP) described in the previous section versus MIP was found [2].

1.7 Public Health Impact

In developed countries, the deaths caused due to cancer are 9.6 million among the total deaths that occur every year which is approximately, 55.3 million people [13]. Thus, making it a public health problem which requires attention. We have inferred that p53 protein plays a key role in suppressing tumor formation. Also, from the previous work conducted we have seen that mutant studies of p53 protein and its structural features have a key role in predicting the p53 protein's activity to suppress the tumor formation. Hence, finding an optimal model that uses these structural features to predict the outcome (activity of protein p53) helps in solving global burden of cancer.

The public health relevance of the study that has been presented here would be to help clinicians predict the occurrence of cancer given the right set of structural features of protein p53 using simple classification models. This also helps in predicting the occurrence of cancer early in life so that the necessary precautions and preparations can be made.

1.8 Plan of Action

Predicting the activity of p53 mutant protein seems to be an important problem in order to address the cancer burden globally. Another challenging task is to use the right feature set for predicting an observation of interest. This motivates understanding both the data and use of features in a way that is scientifically acceptable. In this study, I propose a statistical approach towards selection of features and comparing several classification models for optimal prediction of the outcome.

2.0 Materials and Methods

2.1 Dataset

In this study the p53 mutant dataset from UCI machine learning was used (https://archive.ics.uci.edu/ml/datasets/p53+Mutants) [2]–[4]. The dataset consisted of 31,283 instances (observations) and 5,408 numerical features. Among the total features, the first 4,826 accounted for two-dimensional (2D) structural features which were based on electrostatic and surface properties. The remaining 582 three-dimensional (3D) features were the distance maps between mutant and wild type p53. The 2D and 3D features were obtained through biophysical models of p53 mutant. Based on these features, the transcriptional activity of the p53 mutant (active or inactive) was predicted through in-vivo assays [9]. Active label (denoted by 1) represented a mutation that retains the p53 activity, whereas inactive (labelled as 0) represented the mutation that inactivates its normal activity. Among the 31,283 instances, 150 are active and the rest 31,133 instances were inactive. This showed a highly imbalanced dataset, which needed to be handled during classification tasks.

2.2 Methodology

The pipeline that was followed for analyzing the dataset is illustrated in Figure 1.

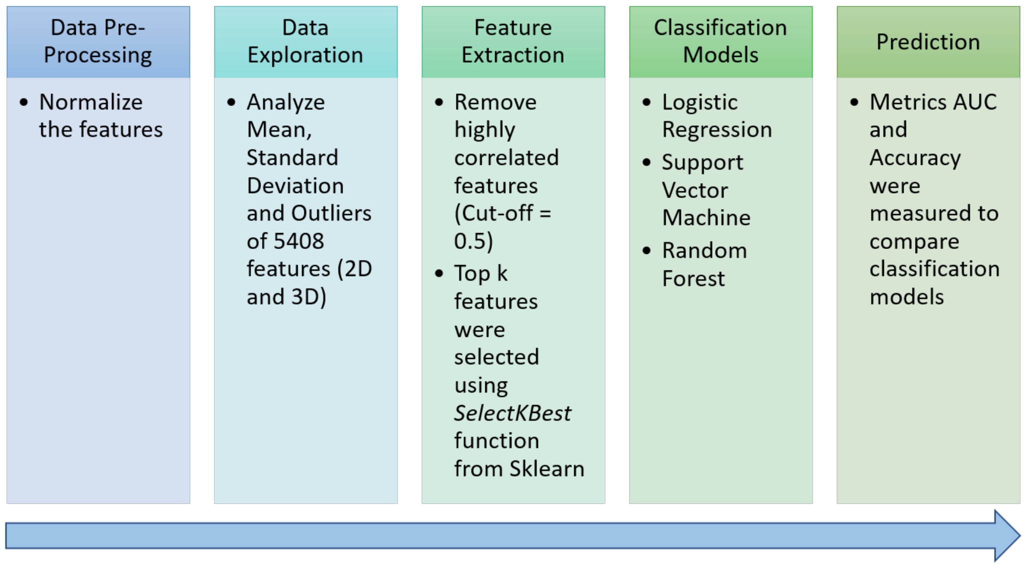

Data Pre-Processing
Normalize the features
Data Exploration
Analyze Mean, Standard Deviation and Outliers of 5408 features (2D and 3D)
Feature Extraction
Remove highly correlated features (Cut-off = 0.5)
Top k features were selected using *SelectKBest* function from Sklearn
Classification Models
Logistic Regression
Support Vector Machine
Random Forest
Prediction
Metrics AUC and Accuracy were measured to compare classification models

Figure 2 Pipeline of the Method

Firstly, the data was pre-processed to get a more informative and understandable dataset. The features in the dataset considered for our study were all quantitative in nature but the metric that was used to measure them was unclear, hence normalizing helped in scaling the whole dataset to a new understandable scale and created uniformity in the dataset. Next, the distribution of data was understood by plotting mean and standard deviation (SD) of the features. The missing data and outliers were counted to further understand the nature of the dataset. Proceeding further, the pre-processed data was used to extract features to reduce the dimensionality of the dataset and to reduce the computational time. An additional step of removing highly correlated features was also performed. For this purpose, the correlation among all 5408 features were computed and the features which had a correlation coefficient |r| greater than 0.5 were filtered which resulted in around 500 features. This feature set was sent through a feature extraction pipeline provided by sklearn's *SelectKBest* to obtain 100 most relevant features [14]. Then, suitable classification models were fit using the selected features. Finally, to compare the classifiers and their prediction ability of the outcome (transcriptional activity of p53 mutant), metrics like accuracy and AUC were calculated. The steps shown in Figure 2 are explained in detail in further sections.

2.2.1 Data Preprocessing

The features in the dataset were first normalized prior to exploring some of its statistical measures. Normalization of the numeric features in the dataset helped in scaling them to a common scale without distorting the broad range of values they possess. The normalized dataset was obtained by applying the formula below for each data entry,

$$x_{new} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

Where, x represented the individual observation in the dataset, x_{new} was the new observation got after normalizing, x_{min} was the minimum observation and x_{max} was the maximum observation. The x_{min} and x_{max} values were calculated for respective features and not the whole dataset. The method to normalize was implemented using sklearn's data preprocessing package called "*Normalizer*".

Further, the dataset had 31,283 instances and 5,408 features which consisted of around 0.6 million missing values that accounted for 0.35% of the total data. Among the total number of missing values, all of them were present in 2D features. 3D features on the other hand did not have any missing value. The number of missing values in each 2D feature ranged from 0 to 150 in number. In total 598,424 missing values were present among 169,178,464 (5,408 × 31,283) entries.

The missing value in each feature did not seem to be very large and hence imputing those with the mean of the respective feature column seemed more appropriate, rather than deleting the entries.

2.2.2 Data Exploration

In order to understand the p53 mutant data, a simple analysis of calculating the mean and standard deviation (SD) for each feature was done. Then, the Coefficient of variation was calculated as a percent using the formula,

$$CV = \frac{s}{\bar{x}} \times 100$$

- Where, CV stands for coefficient of variation, s is the standard deviation and $\bar{x}$ stands for the mean of each feature. The plot for coefficient of variation for all the features with respect to the percent of variability can be seen in Figure 3 below.

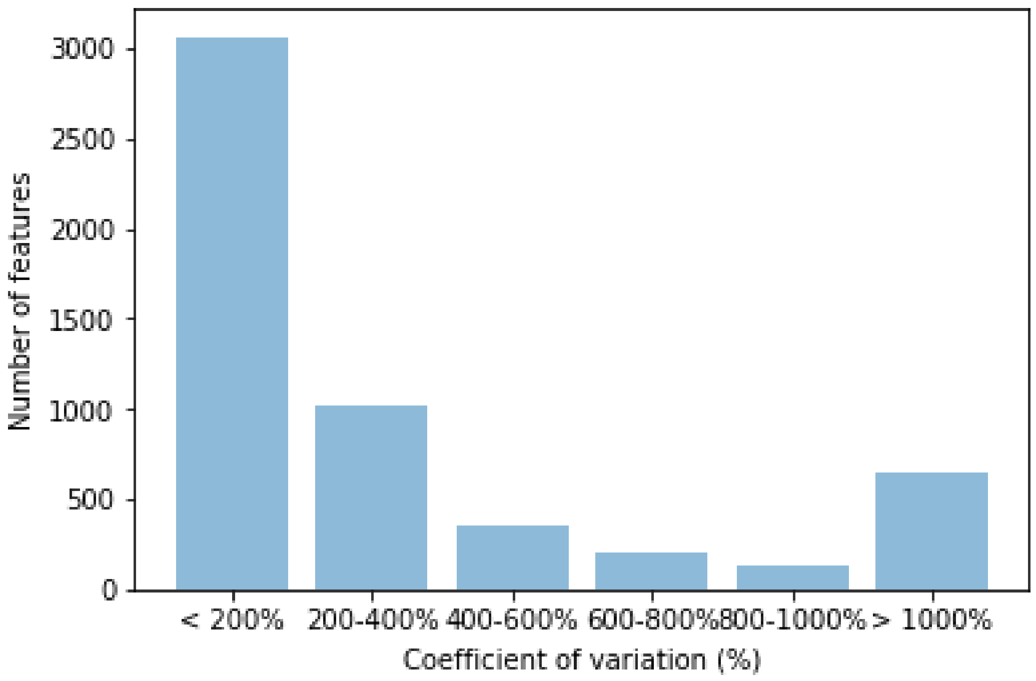


Figure 3 Coefficient of variation among all the features

Coefficient of variation measured the expected variability relative to the mean. Here, mean between features were really small and SD seemed to be high which implied there was high variability in these features. We can see that there were around 3000 features with coefficient of variation less than 200%, around 500 features with CV between 400-1000% and around 700 features with greater than 1000% CV. Also this implied that there might be large number of outliers.

The outliers for each feature were calculated using the interquartile range (IQR) technique. In this technique, the first quartile (Q1) and third quartile (Q3) values for each feature was calculated. Then, the difference between Q1 and Q3 was calculated which was termed as the interquartile range (IQR). The outliers were the points which were less than *Q1 – 1.5 * IQR* or greater than *Q3 + 1.5 * IQR* values. Then, the percentage of outliers were calculated which was used to plot the histogram in Figure 4. This plot helped us visualize the trend in percentage of outliers with respect to the number of features. The number of features with outliers decreased as the percentage of outlier range increased. As seen in Figure 4, there were around 3000 features with 0-5%, around 1500 features with 5-10% and very few features with 35-40% of outliers respectively.

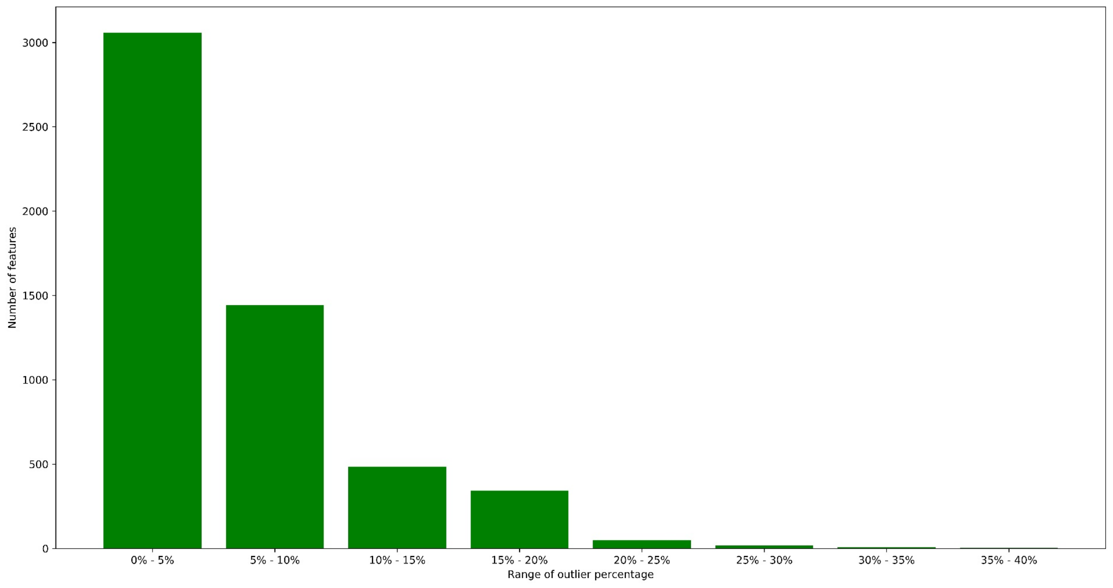


Figure 4 Trend in percentage of outliers with respect to number of features

In order to understand the probability distribution of the UCI dataset, the fourth-order moment (kurtosis) was computed. The data followed a highly non-Gaussian distribution with an overall kurtosis of around 230. A Gaussian distribution would technically have a kurtosis equal to 3 and anything above 3 was labeled as non-Gaussian. Kurtosis was also computed at a feature level. Majority of the features had a kurtosis value greater than 3. Thus, PCA among other dimensionality reduction methods which assumes that the data distribution should be Gaussian cannot be used in this context.

2.2.3 Feature Extraction

Efficient models are built by selecting the right set of features. Careful selection of these features will not only save computational time but also lead to an improvement in the prediction accuracy. The p53 mutant dataset used for the current analysis had dimensions of 5,408 features × 31,283 instances. Handling such a large dataset would not only put a burden on computational time but also affect the performance. Using all 5,408 features would also result in overfitting (training accuracy being significantly greater than test accuracy) [15], [16].

In order to handle the issues of computational time and overfitting, feature selection was performed. First, the correlation matrix was constructed which had the correlation coefficient values between all the features. Then, features with correlation coefficient greater than 0.5 were removed from the upper triangle of the correlation matrix in order to reduce the multicollinearity problem. This resulted in 500 features among 5,408 total features that were present. These 500 features were further reduced to 100 features by using *SelectKBest* function from sklearn which is explained in the following section.

2.2.3.1 Feature Selection: SelectKBest and f_classif scoring function

To reduce the dimensionality of the dataset and for easier comparison between the classifiers, feature selection was performed. From sklearn, the function "*SelectKBest"* was used to select the top 'k' features [14]. As there was no specific way to select a value for 'k', 'k=100' was chosen to get the features in order to reduce the computational time and to reduce the overfitting issue. Choosing 50% of features or using all the features took over 2-3 days to run the models like Support Vector Machine (SVM) and Random Forest (RF). Due to time constraint, only 100 features were used for further analysis.

This function aids in univariate selection of features. Univariate feature selection is a way of selecting features by calculating the univariate statistical tests between features and the outcome variable. There are numerous scoring function options in sklearn's *"SelectKBest"* which can be used based on the nature of features. Since, the features in our dataset were all numerical in nature and the outcome variable was binary the appropriate scoring function was *"f_classif"*. This scoring function calculated an ANOVA F-value between each feature and the outcome variable. The F-value was useful to see if the mean value of features significantly differed by p53 activation status [14]. Based on the scoring function *"f_classif",* "*SelectKBest"* function performed appropriate statistical test and selected 'k' features with highest score.

2.2.4 Train Test Split

Once the features were selected, the dataset was split into training and test sets in the ratio 70-30. Selecting an optimum train-to-test ratio is tricky. Having less data to train leads to greater variances in parameter estimates. However, having less test data will lead to high variation in performance statistics [17]. Hence, to have ample data to train and test, the dataset was split in 70-30 ratio of training and testing sets using sklearn's "*train_test_split*" function. The test set was untouched until the end and was used to analyze the accuracy and discrimination ability of the final classification models.

2.2.4.1 Oversampling technique SMOTE (Synthetic Minority Over-Sampling Technique)

The training set was oversampled using SMOTE (Synthetic Minority Over-Sampling Technique). It helped in improving the decision boundaries in imbalanced data by interpolating the minority class and at the same time under sampling the majority class [18]. The oversampled training set was used for training and validating different classifiers.

The outcome variable in our dataset had two classes (binary outcome), active and inactive. The number of instances in the active and inactive classes were 150 and 31,133 respectively. This posed a problem of imbalance, that is number of observations with active and inactive outcomes were unequal. To handle the data in such situations, we could either sample with replacement from the minority class (the outcome that has occurred less number of times) or reduce the samples from the majority class (the outcome that has occurred highest number of time) to make the two outcomes equal in the dataset. These procedures are called oversampling and under sampling, respectively [19]. In oversampling, the samples are duplicated. Under sampling reduces the observations in the dataset, therefore reducing the data available for training. Also, there were possibilities that the important information might be lost in under sampling. On the other hand, oversampling would solve the issues associated with under sampling but, the training of classification models might be incomplete as these classifiers learn the same thing many times. This would affect the prediction capability of the classification models.

SMOTE is an oversampling technique in which the samples are extrapolated rather than simply duplicating [18]. This is done by obtaining data samples from feature space for both active and inactive target variable and its closest neighbors. Further, it generates new data samples by taking into account the both features of neighbors and features of target variable. Since it takes care of the faults in both under sampling and random oversampling, this method was used in our analyses. Also, SMOTE has proven to be a suitable technique when number of features are not larger than the number of instances which was the case in our data (31,283 instances and 5,408 features) [20].

2.2.5 Method Implemented

The models are trained by looping the dataset through the steps illustrated in figure 5.

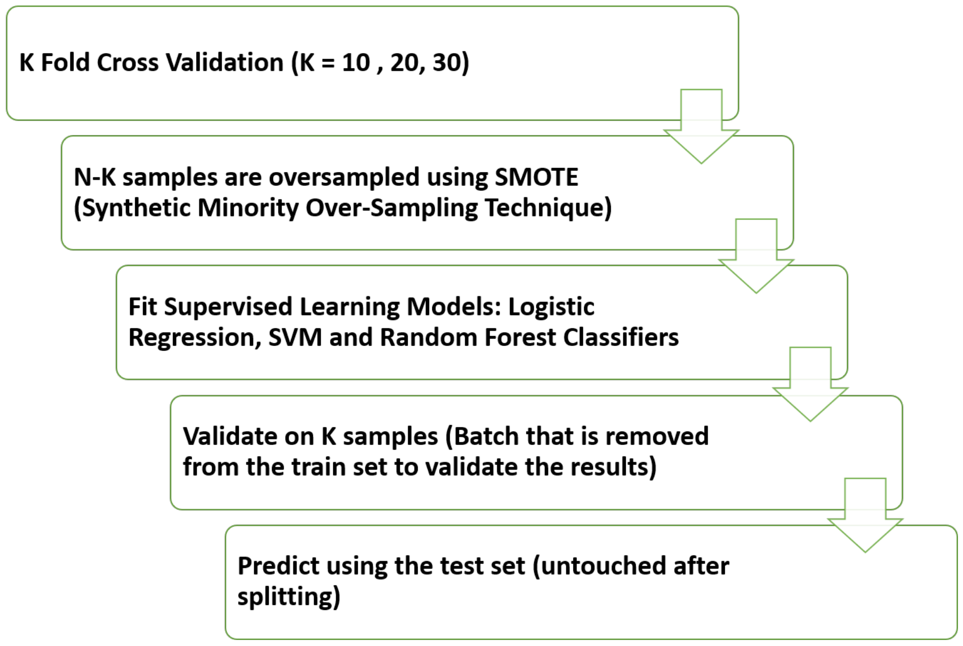

K Fold Cross Validation (K = 10 , 20, 30)
N-K samples are oversampled using SMOTE (Synthetic Minority Over-Sampling Technique)
Fit Supervised Learning Models: Logistic Regression, SVM and Random Forest Classifiers
Validate on K samples (Batch that is removed from the train set to validate the results)
Predict using the test set (untouched after splitting)

Figure 5 Illustrates the steps followed to fit classification models

K-Fold cross validation is also called as out-of-sample testing or rotation estimation. In this method, the whole dataset was divided into K batches. I used K = 10, 20 and 30 for analysis of p53 mutant dataset. The k splits were all same for all the classification models were training using the k-fold cross validation technique. For training, N-K samples were used as depicted in Figure 5, where N denoted the total number of training samples. For validation, K samples were used to predict the activity of p53 mutant based on the model trained. Finally, the 30% of the data used for testing was deployed to predict the outcome using the same 100 features as in training.

2.2.6 Classification Model: Logistic Regression

Since we were dealing with categorical outcome, logistic regression was selected as one of the choices for performing classification.

Logistic regression is an extension of simple linear regression which uses a logistic function to model the binary dependent variable [21]. It is used to model the probability of active or inactive status of p53 mutant in this study. The model is defined as,

$$ln\left(\frac{p}{1-p}\right) = X\beta$$

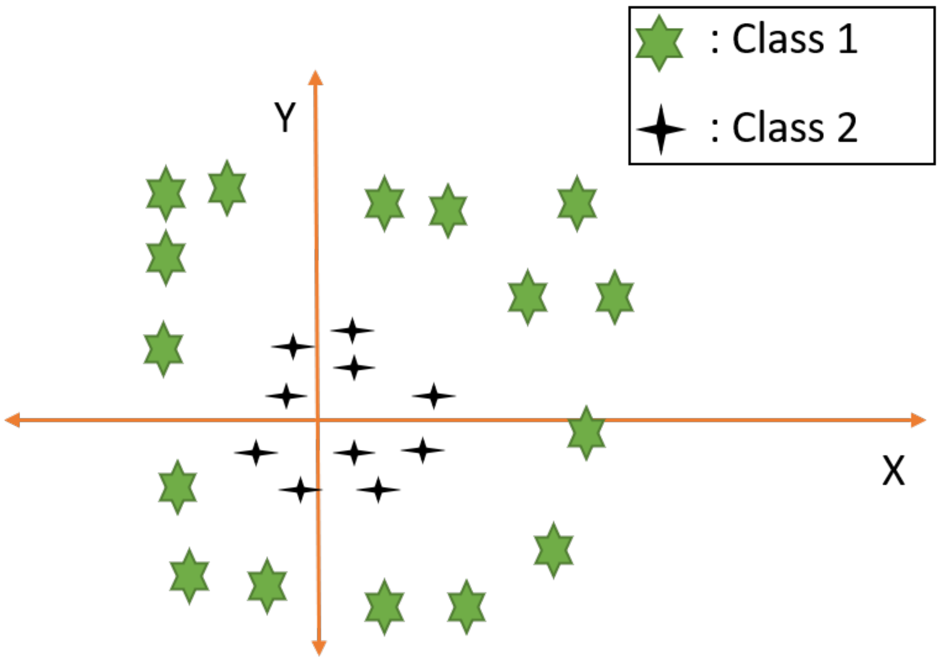
Where, 'p' is the probability of active status of p53 mutant, X is the covariate matrix and β is the parameter vector.

The regression model was implemented using sklearn's "*LogisticRegression"* function. In order to evaluate the model, the training dataset was further subdivided into training and validation set with a K-fold cross validation strategy (where K=10,20 and 30). Prior to feeding the training set into the algorithmic pipeline, it was oversampled using SMOTE technique as described previously. Further, the training and validation performance metrics were reported, and this model was evaluated on the test set. In order to run the model, default parameters were used with regularizing cost function value *C=1*, penalty parameter *L2*, and the number of iterations for convergence *max_iter=100*. Again, sklearn's "*predict*" function was used to obtain the label predictions based on feature dataset using the logistic regression module. Performance metrics such as accuracy and area-under-the-curve (AUC) were measured on the training, validation and testing set separately.

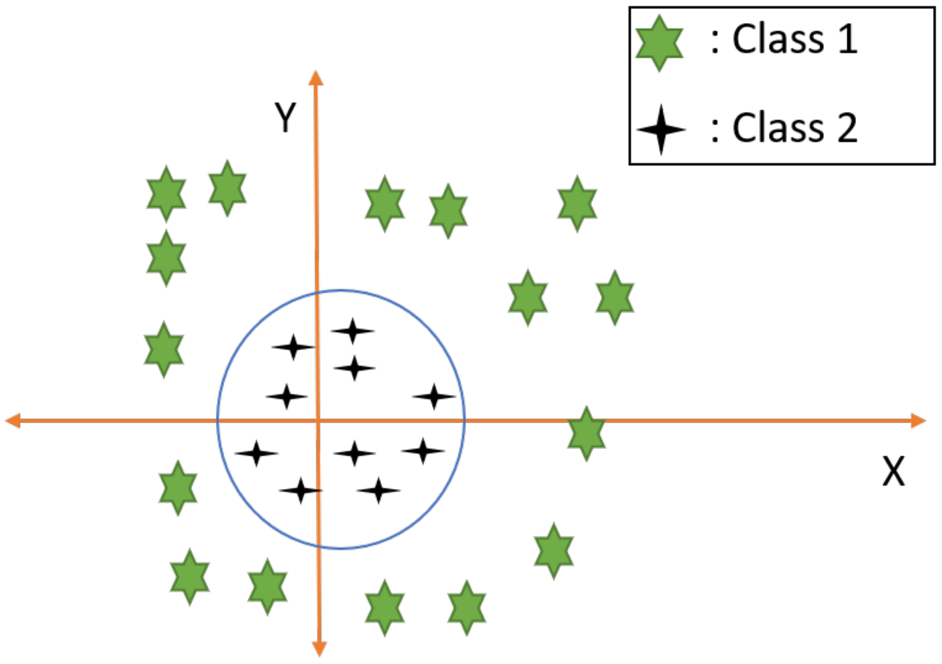
2.2.7 Classification Model: Support Vector Machine (SVM)

Due to its ability to find decision surfaces in a high dimensional space through non-linear planes, SVM was utilized. Also, it handles imbalanced data well [22]. Therefore, SVM was selected as one of the models to predict the p53 activity.

SVM is a type of supervised learning (presence of input and outcome variable to train the data) model. Given the features and an outcome variable, SVM finds decision boundaries that discriminate one category of the data from another. In doing so, it tries to optimize an objective function defined by weights which depend upon the distance between the data points to the decision surface. Thus, it tries to classify by drawing a hyperplane which transforms to a line in two-dimensional (2D) space [23]. For example, if we have distribution of classes as seen in Figure 6a, it would be difficult to separate the classes in x-y plane or by drawing a line. SVM would apply an appropriate transformation to data, in this case, it would add one more dimension or z-axis and separate the data in three-dimensional space which would look like the plot shown in Figure 6b in a two-dimensional space.



6(a)



6(b)

Figure 6 SVM discriminates the two classes in 2D space

For implementation, "*SVC*", a C-Support Vector Classification (SVM module) provided by sklearn was used. The pipeline for training and testing the model was like the one used to implement logistic regression. Here, however, the parameters used to obtain hyperplanes generated by the Support Vector Machine classifier was different. The regularization parameter value of *C=1* was used. Kernel coefficient *gamma* was set to *auto* with a non-linear *rbf* kernel which finds non-linear decision boundaries (curves) to classify.

2.2.8 Classification Model: Random Forest

Random forest is very robust and handles overfitting issue is minimized when it is used for classification tasks [24], [25]. Hence, this was selected as one of the models to train on the 100 features selected in the feature extraction step.

Random forest (RF) uses multiple decision trees to make the final decision about the outcome prediction. Decision tree can be thought of as multiple yes/no questions which would lead to the predicted class. Each decision tree is built using a bootstrap sample of the data and random sample of predictors at each node of the tree. Each tree gives a classification vote and estimated probability, then the final decision is made based on the majority vote or average prediction. The prediction accuracy increases with the number of trees obtained by the number of data samples generated. The parameter, i.e., the number of estimators, is used to define the number of random decision trees used for predicting the class of the outcome variable which is obtained by taking the majority among the decisions provided by each tree.

Since, we are dealing with a classification problem, Gini impurity criterion was used for data splitting in the decision tree. This criterion measures the frequency of misclassification of a randomly selected instance if it were labeled randomly. The number of samples that is needed to split an internal node (a node that can be split further) and number of samples needed to be at a leaf node (terminal node which is not split further) should be specified as well for training an RF algorithm.

Random Forest was implemented using sklearn's ensemble package "*RandomForestClassifier*". Once again, the same pipeline for evaluating the model on test set was followed similar to SVM and Logistic Regression. Here, the default parameters provided by the software was used. Specifically, the number of estimators *n_estimators=10*, *criterion=gini*, minimum number of samples at leaf node *min_samples_leaf=1, min_samples_split=2* was selected for generating the RF model.

2.2.9 Metrics Used for Comparison

The metrics used to compare the results between train, validation and test sets were accuracy and AUC (Area Under the Curve). AUC is the area under the Receiver Operating Characteristic curve (ROC) which is a plot of sensitivity (true positive rate) v/s 1-specificity (false positive rate). AUC helped us understand how well the model discriminated between the two classes, active and inactive. Accuracy measured how precisely the model predicted the outcome using the features selected in the feature selection step. Accuracy and AUC are given by,

$$Accuracy = \frac{T_p + T_n}{T_p + T_n + F_p + F_n}$$

$$Sensitivity = \frac{T_p}{T_p + F_n}$$

$$Specificity = \frac{T_n}{T_n + F_p}$$

$$False\ Positive\ Rate = \frac{F_p}{T_n + F_p}$$

$$AUC = Area\ under\ the\ ROC\ plot$$

Where, T_p (true positives), T_n (true negatives), F_p (false positives) and F_n stands for (false negatives).

Typically, an AUC value of greater than 75%-80% discriminates the classes well. Similarly, an accuracy of over 80% is reasonable.

3.0 Results

3.1 Feature extraction results

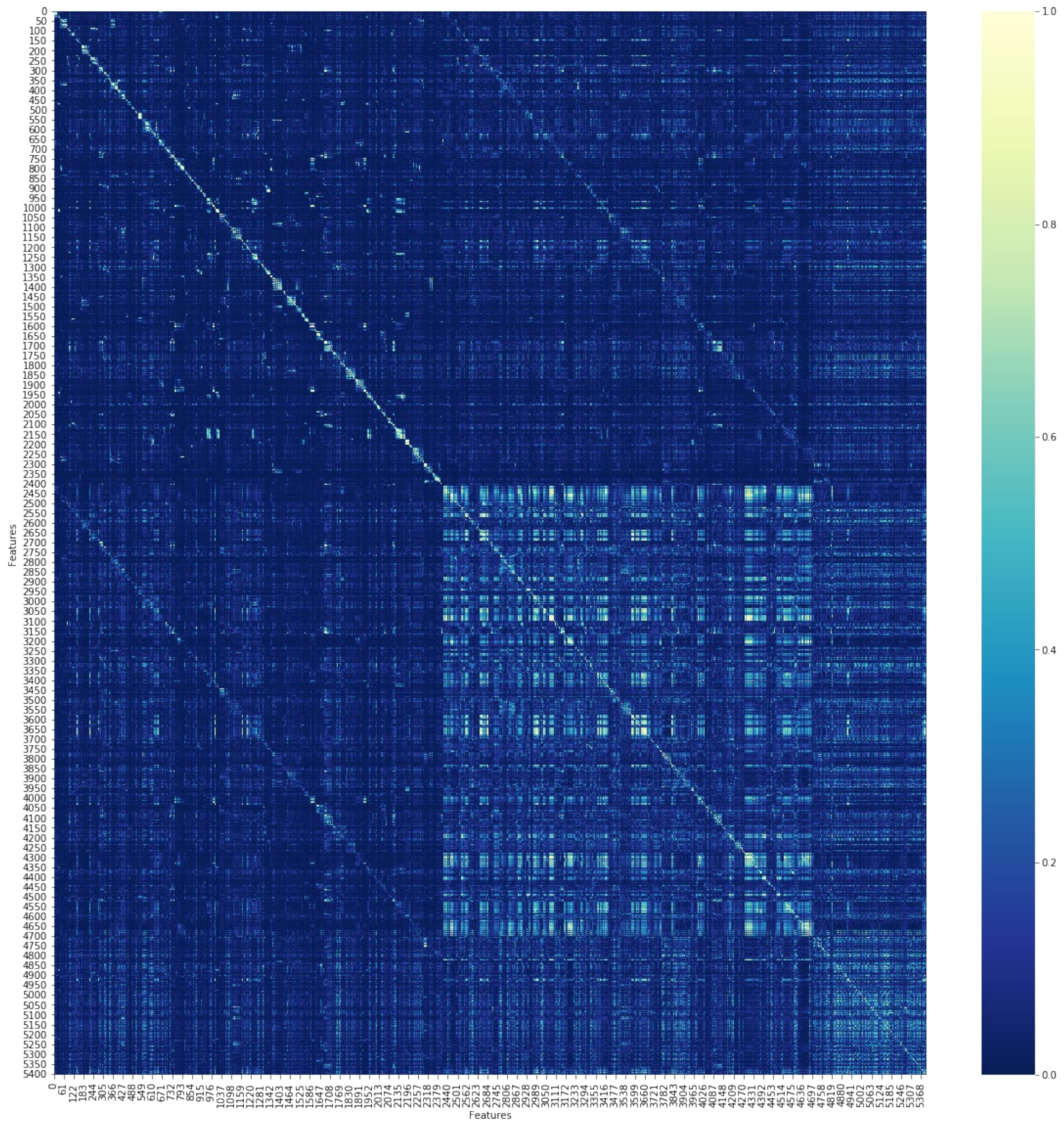


Figure 7 Heatmap of correlation-coefficients of all 5,408 features

The Figure 7 shows the heatmap of correlation coefficient among all 5,408 features. Once the correlation coefficient matrix was constructed the absolute values were considered to plot the heatmap. As we can see there were many highly correlated features with correlation coefficient greater than 0.5.

As explained in the method after constructing the correlation coefficient matrix, using |r| >0.5 as a threshold the features were removed from the upper triangular region of the correlation coefficient matrix. This way of removing the features made sure one of the highly correlated feature was still present and this can be seen in Figure 8 below.

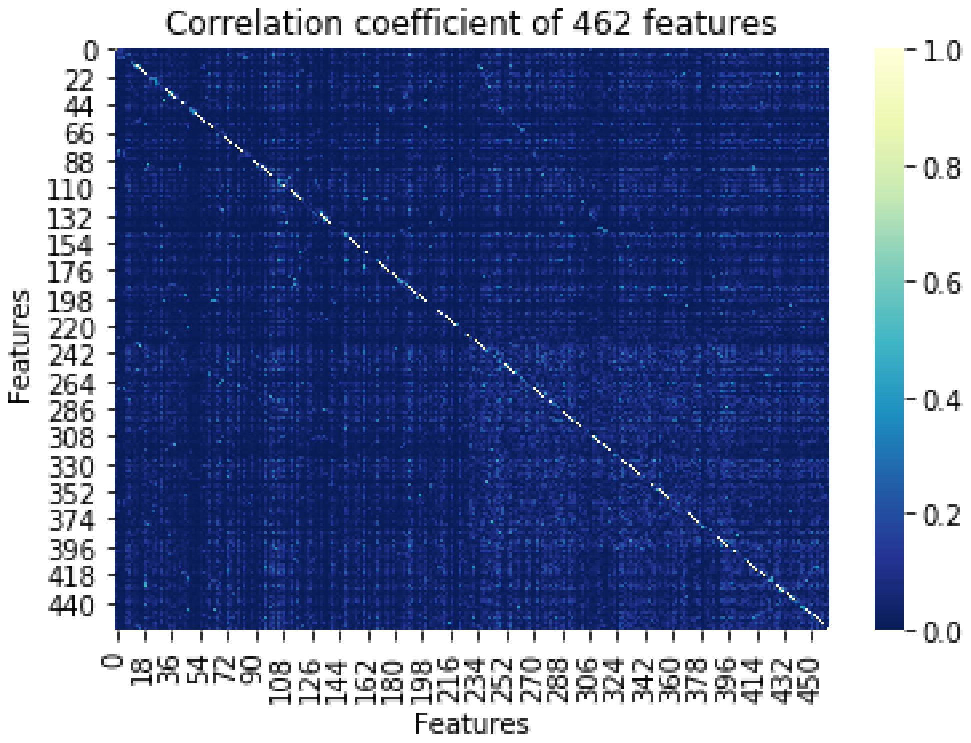


Figure 8 Heatmap of 462 features that were obtained after removing the higly correlated features

Further, as described in the previous section, prior to training the models on dataset, the first step was to select 100 features in order to reduce the computational time. Th3 462 features were further downsized to 100 features using *SelectKBest* function. The heatmap in Figure 9 below shows the correlation between final 100 features which were used for training. All the 100 features which were selected had missing values of 150 originally before imputing them with respective feature column mean. Almost all of the features had a correlation that spanned between 0 to 0.25. All the heatmaps were plotted with the absolute values of correlation coefficient.

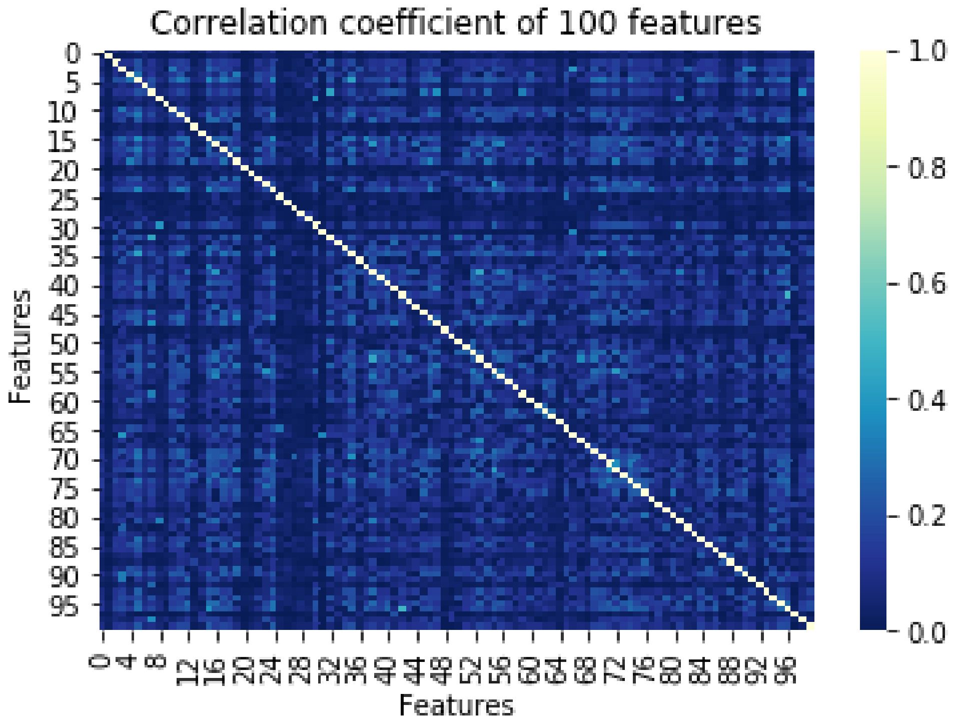


Figure 9 Heatmap of 100 features which were selected for training the models

After feature selection, three different classification algorithms: Logistic Regression (LR), Support Vector Machine (SVM), and Random Forest (RF) was implemented on the UCI p53 mutant dataset. A K-fold cross validation approach was used for each of the learning model to deal with the problem of overfitting and also as an attempt to improve accuracy on the unseen data (test set). For the purpose of evaluation, model accuracies and area-under-the-curve (AUC) was compared for a variety of combinations of algorithms and folds for cross validation. Following section elucidates the performance of each model on training and validation set followed by test set.

3.2 Performance on Training and Validation data

Three classification models (LR, SVM, and RF) were used to fit the training data which was oversampled using SMOTE, a technique used to handle class-imbalance. Further, this trained model was evaluated on the validation set that was generated using K-fold cross-validation (K=10, 20 and 30). Predictions on the train set and validation set were used to compute accuracy and measures of class discrimination using AUC. Results from respective algorithms is described in the following section.

3.2.1 Performance of Logistic Regression

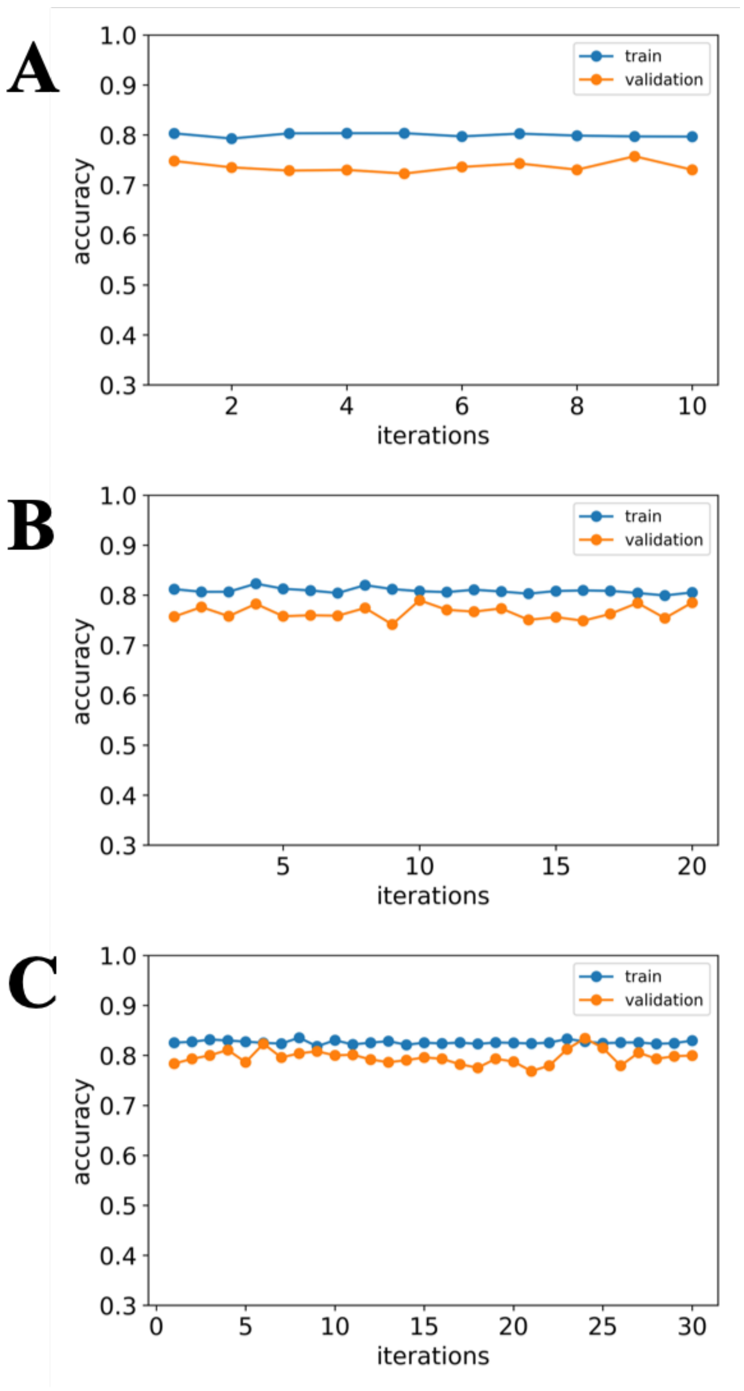


Figure 10 Accuracy measure of LR on train and validation data for three different folds (where A: K=10, B: K=20; C: K=30)

The panel of plots shown in Figure 10 describes the performance of logistic regression on train and validation set using default parameters provided by sklearn package which was mentioned in the Methods section of this report. Based on the results in Figure 10 (10A, 10B, 10C) which visualizes the accuracy score on both the sets, it can be observed that the training accuracy is slightly greater than validation accuracy. On an average, the validation accuracy fluctuates around 75% mark for 20 and 30-fold cross-validation set. Looking at the 10-fold set, the average validation accuracy is around 79%. Training accuracy on the other hand has an average accuracy approximately equal to 80% for all the flavors of cross-validation.

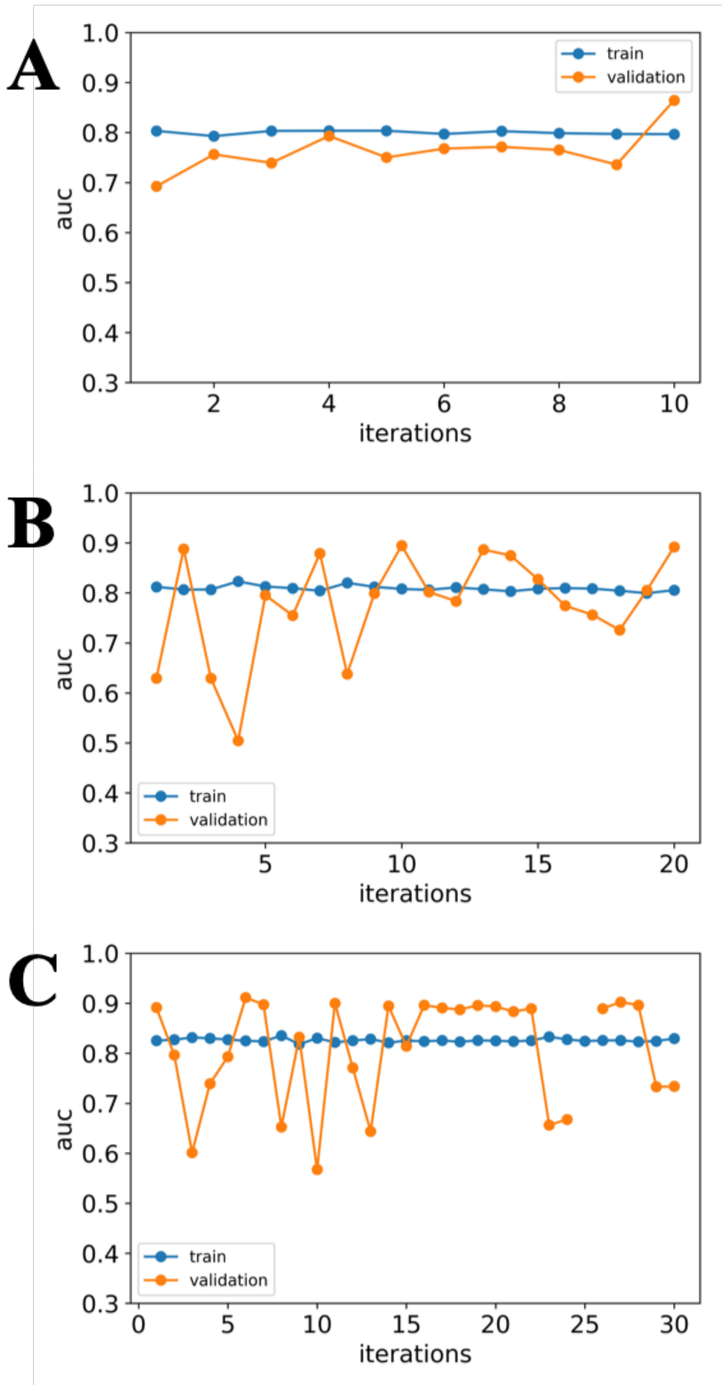


Figure 11 AUC measure of LR on train and validation data for three different folds (where A: K=10, B: K = 20, C: K=30)

Figure 11 shows the AUC score for each iteration over a span of different K-fold cross-validations for LR model. It is interesting to observe that, the AUC on train data is approximately consistent for each of the cross-validation set with a score of 0.80. However, the AUC scores for validation set for each iteration and across different K (10, 20 or 30) values seems to be fluctuating between 0.50 and 0.90. For the 10-fold cross-validation, AUC score ranges between 0.70 and 0.86 at first and last iterations respectively. Only one score out of ten falls above the training AUC mark of 0.80, while the rest are below that value. Similarly, for 20-fold cross-validation, eight AUCs are above and twelve are below the threshold defined by training AUC. We can observe that the 4th iteration performs worst with an AUC value of around 0.5 when trying to infer the correctness in differentiating the activity of p53 to be active vs inactive. Two such samples are visible in 30-fold cross-validation at 3rd and 10th iteration run when the AUC is the least. The large variability in the AUC scores between all the examples can be attributed to the model fitting and randomness in generating the validation data. Compared to the 10-fold set, the range of AUC values span from 0.5 to 0.9 for 20-fold variation and 0.6 to 0.9 for the 30-fold cross-validated dataset.

3.2.2 Performance of Support Vector Machine

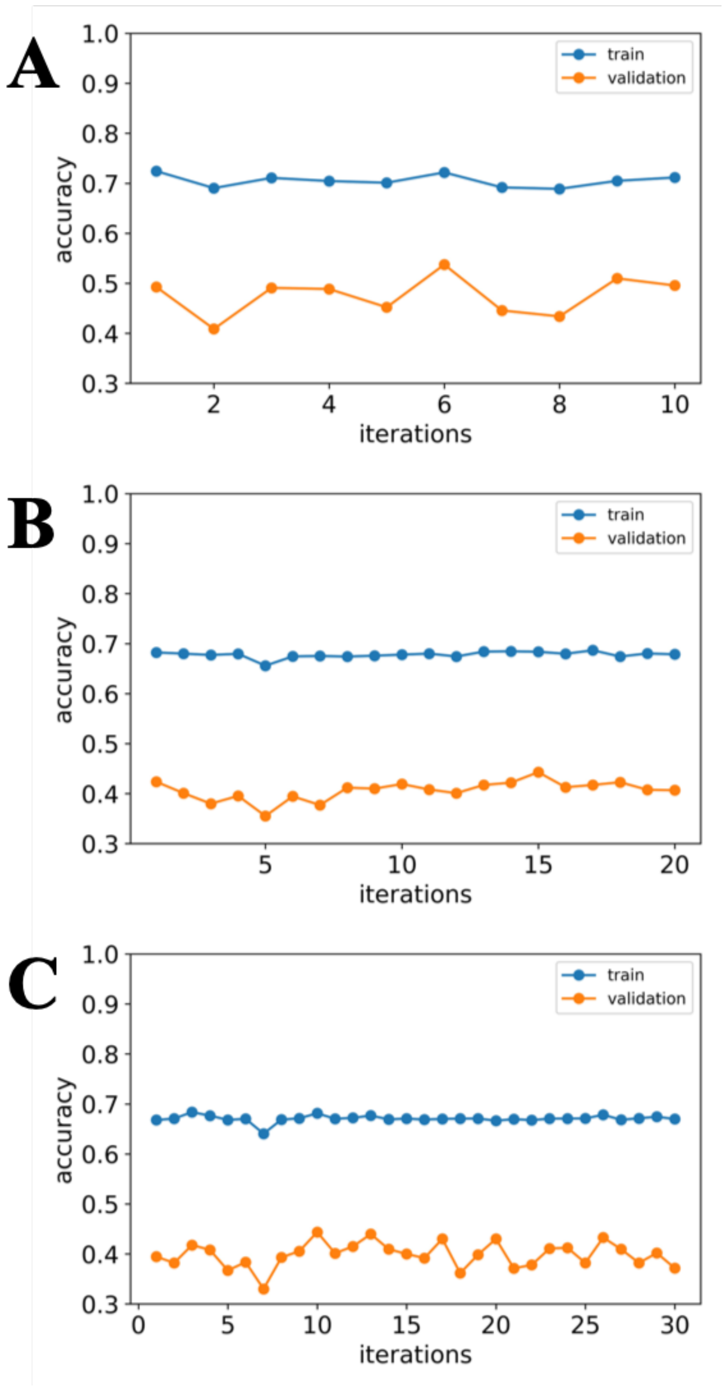


Figure 12 Accuracy measure of SVM on train and validation data for three different folds (where A: K=10, B: K = 20, C: K=30)

The performance of SVM classifier on the training and validation data follows a similar trend when compared to logistic regression as it can be seen from Figure 12. But, looking at the blue line plot that indicates accuracy for train data, it shows that there is no much variability in the values. Almost all of the training samples for three different cross-validation folds shows an accuracy in between the range 0.65-0.70. But, there is a large variation in the validation accuracy of SVM classifier. For the 10-fold variation, the accuracies range from 0.4 to 0.5. For the other two categories (20 and 30-fold) the accuracies range from 0.35 to 0.42 and 0.3 to 0.43 respectively. This can be observed from the fluctuations in the orange line depicting validation accuracies in Figure 12.

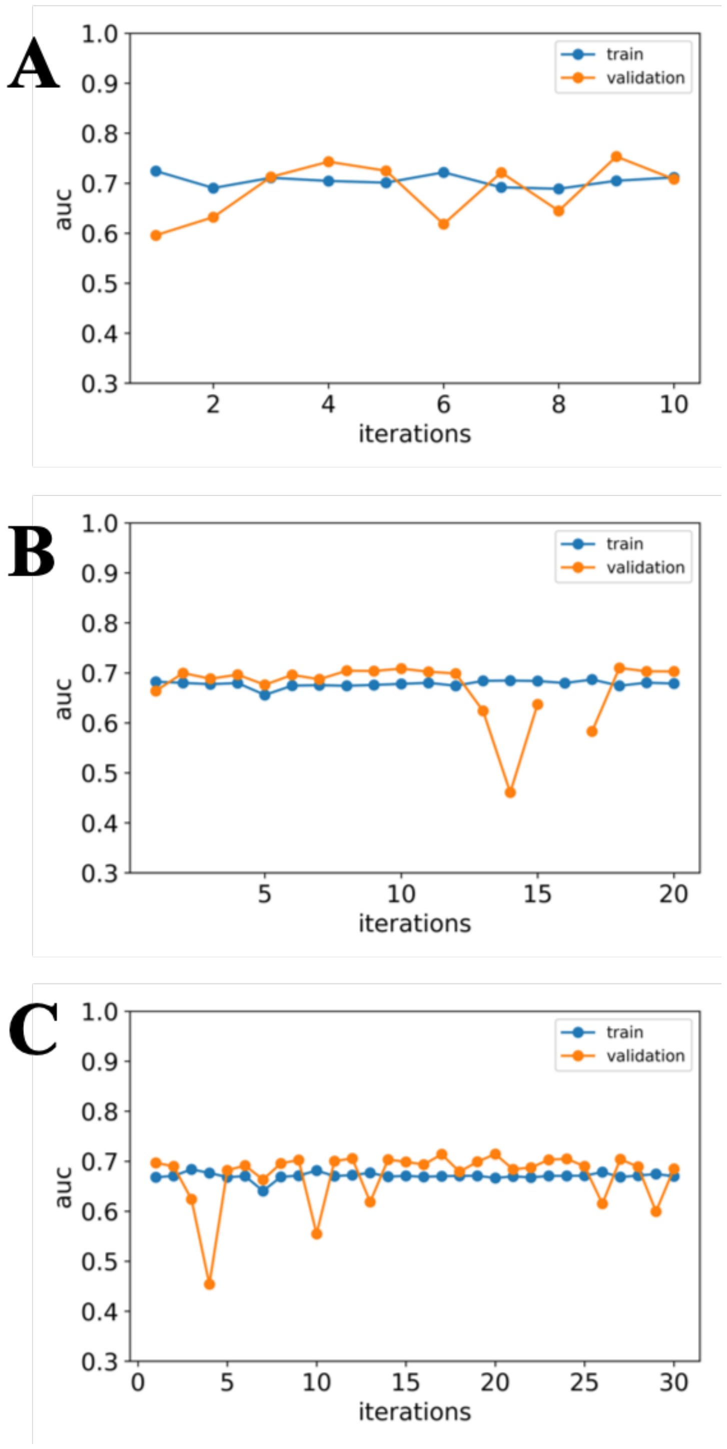


Figure 13 AUC measure of SVM on train and validation data for three different folds (A: K=10, B: K= 20 and C: K=30)

Figure 13 shows the AUC curve for different cross-validation based generated samples. The AUC values for 20-fold and 30-fold cross-validation falls in the range 0.42-0.70 which can be observed from the linear curve in Figure 13B and 13C. However, from Figure 13A we can see that there are slight fluctuations in how well the algorithm (here SVM) is able to differentiate between the two classes and the values lie between 0.60 and 0.72.

3.2.3 Performance of Random Forest

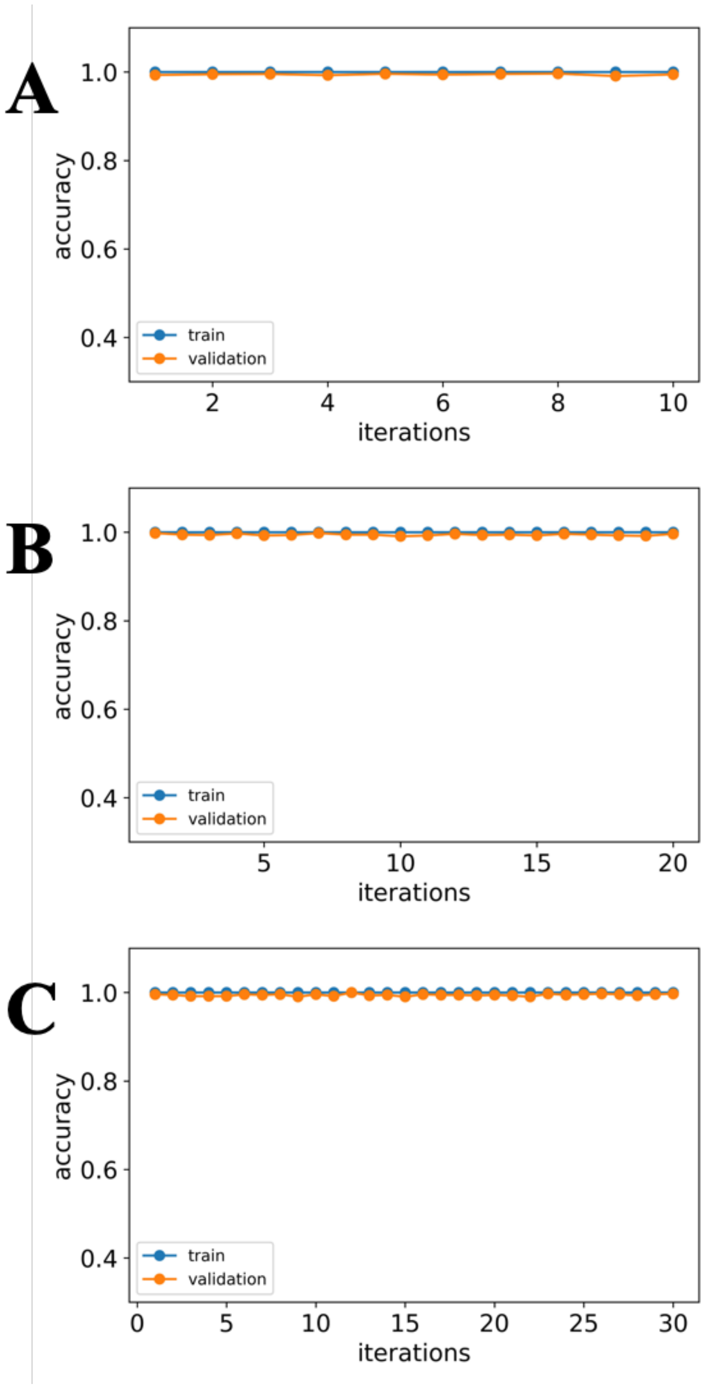


Figure 14 Accuracy measure of RF on train and validation data for three different folds (A: K=10, B: K=20 and C: K= 30)

The third classification algorithm that was implemented for the classification task was random forest. Contrary to the results obtained from LR and SVM on training and validation data, here as we can observe from Figure 14, the training data seems to perform slightly better than validation set with a larger accuracy. But we can see that the trend in both the accuracies is observed to follow a similar pattern indicated by an almost straight blue and orange linear curve with less fluctuations in the accuracy values for different samples. In case of random forest, the training data seems to perform exceptionally well, with almost a 100% accuracy for all of the data samples for three different cross-validation folds (blue line shown in Figure 14). Random forest applied to validation set also performs considerably well compared to its counterpart, LR and SVM, with a range 99%-99.9%. This shows the effect of using RF on this particular dataset to obtain a greater accuracy.

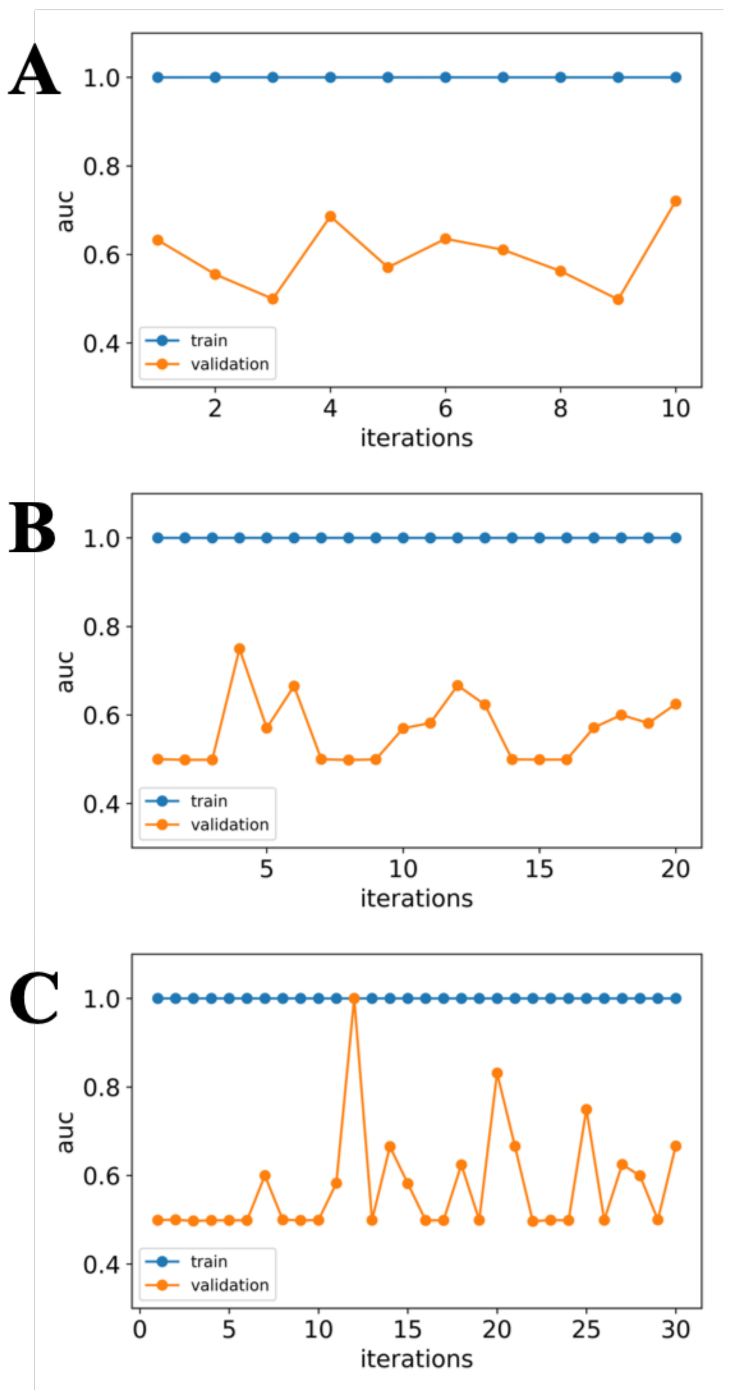


Figure 15 AUC measure of RF on train and validation data for three different folds (A: K=10, B: K= 20 and C: K=30)

As we previously observed that, the training accuracy was higher than validation accuracy, similarly, even in the AUC plot, train data seems to perform better with a maximum AUC of around 1 as shown by the blue line in Figure 15. The discriminating score of validation data, however, fluctuates between 0.5 and 1 as it can be observed from the orange curve in Figure 15. For the 10-fold cross-validation, the minimum AUC achieved is around 0.50 for the third iteration and the highest AUC reached is 0.74 for the tenth iteration. In the 2nd set that accounts for 20-fold cross-validation, the lowest AUC is around 0.5 that can be seen for the 1st, 2nd, 3rd, 7th, 8th, 9th 14th, 15th, and 16th iteration. In the last, 3-fold variation data, larger number of samples are poorly classified with an AUC of around 0.5 for 18 iterations,that is around 60% of the runs do not perform well. A highest AUC can be observed for 12th iteration that almost perfectly classifies the binary data.

3.3 Performance on Test data

As mentioned previously, 30% of the total dataset was set aside for testing the model performance. Specific parameter sets were chosen across three different classification algorithms in order to predict the activity status of p53 mutant. This section describes the model performance on the test data which was not seen by the algorithm while trying to learn the objective function for classification.

Table 1 Test accuracy for different combinations of classification algorithms and folds for cross-validation

TEST ACCURACY			
K-FOLD CV	LR	SVM	RF
10	0.730	0.505	0.995
20	0.770	0.415	0.993
30	0.799	0.410	0.995

According to Table 1 which describes the performance of LR, SVM and RF for 10,20 and 30-fold cross-validation techniques to train the data, an overall good accuracy of greater than 94% can be observed when trained using RF. In terms of accuracy alone, random forest is observed to outperform SVM and LR by a huge margin achieving an average accuracy of around 99.4%. LR falls second in the line with an average accuracy of around 76.6%. SVM on the other hand performs poorly when compared to RF and LR with an average accuracy of only 44.3%. The possible reasons for this large variation in average accuracy is described in the Discussion section.

Table 2 Test AUC for different combinations of classification algorithms and folds for cross-validation

TEST AUC			
K-FOLD CV	LR	SVM	RF
10	0.768	0.727	0.706
20	0.760	0.687	0.595
30	0.737	0.694	0.646

Since, accuracy alone cannot be a good measure to gauge how well the classifier is performing, AUC was calculated for each of the data variations. It is interesting to observe that, LR which had an accuracy of around 76.6% seems to have the best AUC score with 0.77 for 10-fold cross-validation set and 0.76 for 20 and 0.74 for 30-fold dataset variation. In case of SVM, it achieves a moderate AUC of around 0.70 on an average. This is surprising since SVM was able to achieve a lower accuracy of around 44%. RF on the other hand is able to discriminate between the classes rightly only 65% although it achieved a high accuracy.

3.4 Summary

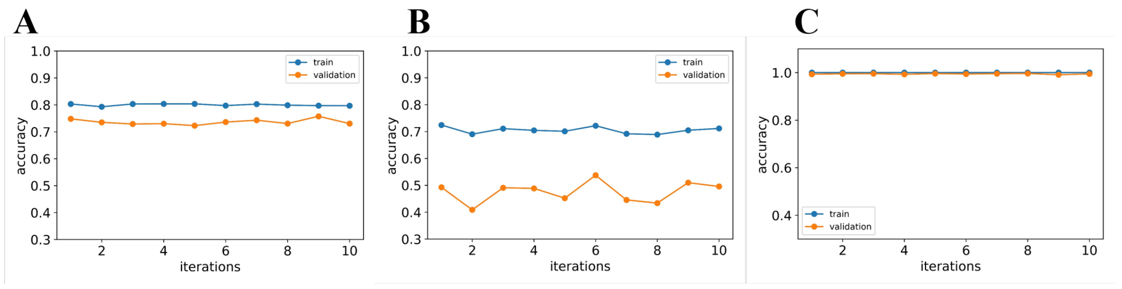


Figure 16 Illustrates the accuracy for LR, SVM and RF for K=10 (A: LR, B: SVM, C: RF)

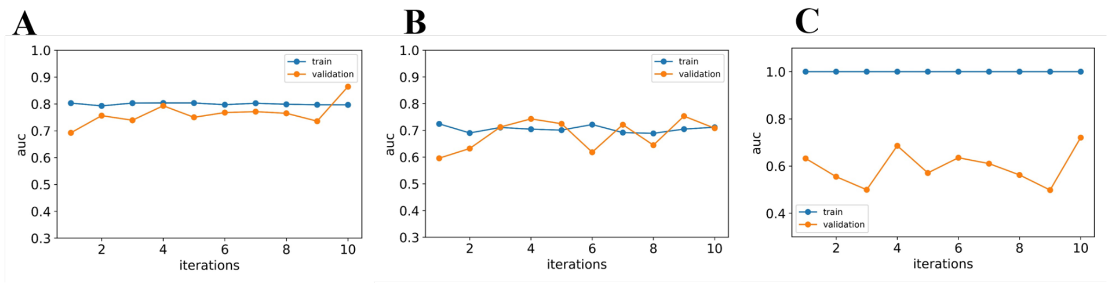


Figure 17 Illustrates the AUC for LR, SVM and RF for K=10 (A: LR, B: SVM, C: RF)

Among the K=10, 20 and 30, K = 10 seemed to perform better among the three. Figures 16 and 17 above shows the accuracy and AUC of LR, SVM and RF respectively. Thus, we observed the accuracies and AUC obtained by logistic regression, support vector machine and random forest for three different cross-validation folds from Figures 10-15 for train, validation, and test dataset. Performance of train data over validation data was always higher when using LR, SVM, and RF. We observed a large difference in the accuracies achieved by all the algorithms across different variations. But, in terms of AUC metric, LR outperformed SVM and RF. In the following section, the results obtained will be discussed in detail and possible explanations for some of the limitations of the model will also be highlighted.

3.4.1 Importance of features while training the models

Once the classification models LR, SVM and RF were trained we wanted to see which among the 100 features were important for the prediction. The Figure 18 and 19 shows the importance of features for LR and RF models. Due to the non-linear kernel that was used to train the SVM model, the transformation that was applied in high dimensional space to classify the classes were unclear and there was no way to extract them. Hence, the feature importance could not be extracted for SVM.

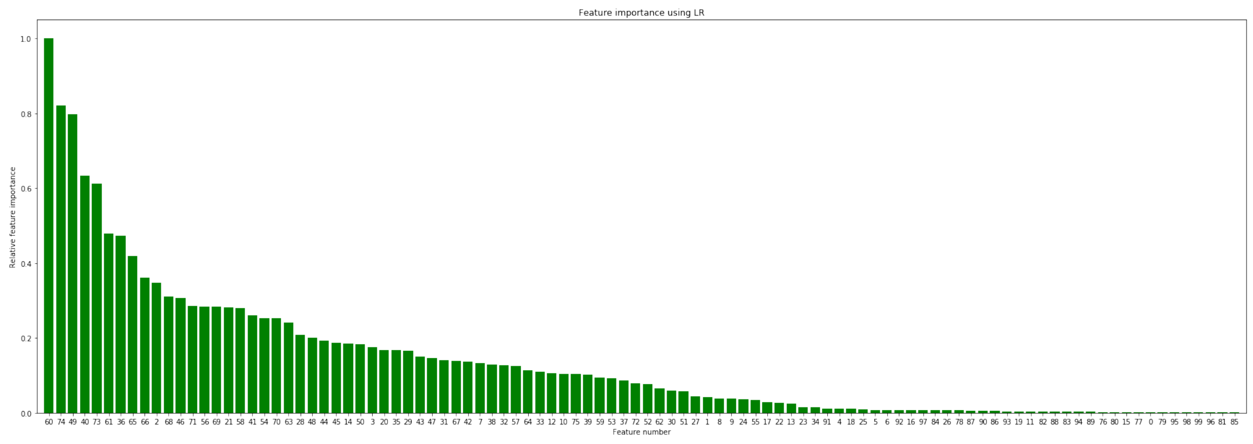


Figure 18 Feature Importance for training LR

We can observe in Figure 18, that the top 5 feature of importance for training a LR model on this dataset were feature numbers 60, 74, 49, 40, and 73.

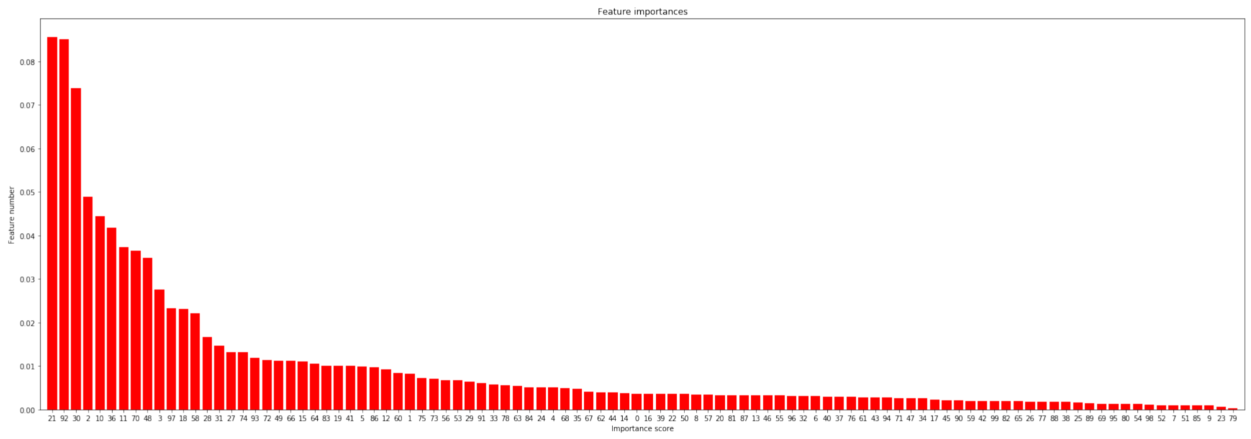


Figure 19 Feature Importance for RF

We can observe in Figure 19, that top 5 feature of importance for training a LR model on this dataset were feature number 21,92,30,2 and10. It looks like the top 5 features in LR and RF are totally different. This indicated that the important features depend on the model trained.

4.0 Discussion

For this report, an evaluation of three classifiers: Logistic regression (LR), Support Vector Machine (SVM) and Random Forest (RF) was deployed using a three-fold cross-validation strategy with number of folds including 10, 20 and 30 to identify the activity of p53 mutant based on 2D and 3D features. The dataset consisted of 5,408 numerical features from 31,283 observations with around 600,000 missing values. The missing values were imputed using the average of the respective columns as a first step towards preprocessing the data.

Further, due to the high intrinsic data dimensionality, it was not feasible to work with the entire 5,408 features without using some sort of efficient computational strategy. Thus, it was necessary to work with the most important features that affected the target variable, in our case, the activity status of p53 mutant. Since the data failed to follow a Gaussian distribution, classic dimensionality reduction techniques such as Principal Component Analysis (PCA), is a tool in machine learning that is used to examine interrelations among set of variables which further helps in dimensionality reduction of large sets of features. Here, this tool could not be applied.

Apart from having the advantage of reducing computational time, proper feature selection also reduces the problem of overfitting, thereby improving accuracy. Features that do not directly contribute towards the binary classification problem might affect learning algorithms such as LR to a greater extent compared to SVM and RF. Some of the feature extraction algorithms were attempted to select the most relevant attributes. Among these was pyMRMR [26], an algorithm that works towards finding minimum redundancy and maximum relevance between the features such that they are mutually distant from each other but still contribute towards having a correlation with the output variable. But this method took over 48 hours to obtain 100 features from the UCI dataset, and thus it was discarded.

Next, a combined technique based on correlation-coefficient that filters the features with greater than 0.5 correlation and F-statistic was used to obtain a smaller feature set that can explain the outcome variable. For this, sklearn's "*SelectKBest*" was used that takes as an input, a scoring function based statistical metric computed between the feature set and returns *k* highest scoring features that are most relevant in predicting the target variable. The package offers *chi2* and *f_classif* as scoring functions. *SelectKBest* used with *chi2* as the input parameter computes the chi2 statistic between each feature variable and the output variable. A lower value of this chi2 statistic indicates statistical independence between the feature and outcome. On the other hand, a larger value indicates the opposite, that is, the feature has a relationship with the target and the importance of this relationship can be quantified numerically based on the score. However, this is true only for non-negative features and therefore this scoring function was not used. Also, *chi2* is less meaningful if the features and target variable have a non-linear relationship which is true in this case.

The second scoring option provided by sklearn is *f_classif* which computes F-statistic for each feature assuming the outcome variable to be a class label. The statistic is measured as ratio between the explained variance to unexplained variance which signifies the proportion of variability between groups compared to within groups. The number of features *K* (groups) that is required as an input is used to compute those number of non-repeating values of the target variable based on F-statistic. In this case, a high F-score indicates that the averages of these groups are not equal and also it would indicate that the feature values are derived from a Gaussian distribution which is contradicting to the data under consideration. Thus, lower values of this scoring function are meaningful and is considered for selecting the features. Based on this, 100 features (K=100) were selected for further analysis to predict the activity of p53 mutant. An exhaustive analysis for choosing the right *K* value was not performed. Further, other feature extraction techniques that may be better suited for data which does not follow a normal distribution was not evaluated. Also, the 100 features that were extracted from the pipeline explained in method section were all 2D in nature and the interesting fact is that they all had missing values of 150 before imputing them with the mean. The outliers in these 100 features ranged from 0 to 400. This seems to be interesting thing to consider for further analysis. The feature importance plot that was obtained for LR and RF had top 5 features with missing values of 150 and outliers ranged from 93 to 384.

Among several classification models available, the reduced dataset was trained using LR, SVM, and RF. Since the dataset had only 150 active instances compared to 31,133 inactive instances, rejecting any further data processing techniques might have led to obtaining high accuracy but low AUC score. Thus, in order to deal with class-imbalance SMOTE technique was used which is explained in detail in the Methods section [27]. It is a popular method to generate synthetic samples of data that can correct for issues arising from class-imbalance. Another approach that was tried for oversampling was by making duplicate copies of the data, but, when trying to perform cross-validation, the algorithm was exposed to the same data either twice or at least once. This had a negative effect on obtaining the right parameter set for prediction. Although, certain randomization techniques could have been tried to avoid this problem, in this situation, established oversampling methods like SMOTE was used. Next, the oversampled data was modeled for prediction using cross-validation. This technique was useful to assess the performance of algorithms and evaluate their outcome on new dataset. While training the models such as LR, SVM or RF, it saw only the train data and learnt the parameters for prediction. However, in order to have a more realistic direction towards data modeling, it needs to perform well even on the test set. Two methods for cross-validation could have been employed. First, is the leave-one-out cross-validation which leaves one instance out for validation and trains on the remaining instances and this is iterated until all the instances is left out for validation once. Since this method was computationally expensive and given the large number of instances provided by UCI dataset, the other approach using K-fold cross-validation was used. In this method, instead of leaving one instance out for validation, a block of K-instances was left out for validation and the algorithm was trained on remaining instances. Three different K values of 10, 20 and 30 was used and the performance for different models were compared.

The results from logistic regression showed that the algorithm performed better on training data over validation data. The accuracies obtained by training set were up by a margin of around 5% for 10-fold and 20-fold cross-validation set and 1% for 30-fold set in contrast to the validation data. Training error is bound to be lower than validation error. In all the cases for model training using LR, that is, using K=10, 20 and 30, it is intuitive to observe that the validation error is indeed greater than the training error. The possible explanation may be due to a situation where the training set might have obtained easier data to predict compared to the validation data or a possible overfitting. This 'overfitting' obtained by looking at the accuracy curve can further be validated by looking at the AUC scores from Figure 9. As we can see, almost half of the iterations belonging to the validation set of 20 and 30-fold data have a larger AUC. Based on these two metric results while training the model, we can observe from Table 1 and Table 2 that LR performs well in predicting the class (active vs inactive) in terms of both accuracy and AUC. We can also see that as the *K*-value increases from 10 to 30, the accuracy seems to increase from 0.730 to 0.799. On the contrary, AUC decreases from 0.768 to 0.737. A 4% decrease in AUC can be observed when

K is changed from 10-fold to 30-fold. But when changed from K=20 to K=30, a 3% change in AUC score was observed. Overall, logistic regression seems to perform well in differentiating between active and inactive class of p53 mutant data based on feature set.

Next, in an attempt to find decision surfaces in high-dimensional plane, SVM was implemented. While training the model using three different folds for cross-validation, the training accuracy was much larger than validation accuracy by a margin of around 30%. The validation accuracy is found to be seemingly low with values ranging from 31% - 55%. But in the case for 10-fold and 20-fold cross-validation, the smaller number of iterations had AUC scores greater than training AUC which was also true for 30-fold scenario which had over 60% of the validation runs to have greater AUC. From Figure 11 it is intuitive to observe that SVM's validation accuracy was much lower than LR and also in terms of AUC it does not seem to outperform the regression model. Even for the training data, the average AUC was around 0.68 which is 12% lesser than the average AUC achieved by LR on train data. We can also observe from Figure 11 that fewer number of iterations have AUC value around 0.5 (no discrimination) especially for the 20-fold and 30-fold case. Following the "No free lunch theorem" which is established in the machine learning literature, there is no one model that was previously expected to perform the best on this dataset. Thus, SVM's lower performance compared to LR may be due to the intrinsic nature of the data not being linearly separable. On the test data, SVM did not perform well, with an accuracy of around 45% and with an increase in the K-folds, the accuracy was observed to decrease; however, the difference was not seemingly significant for 20-fold and 30-fold set. In terms of AUC, SVM had an average score of around 70% which did not seem to vary much by changing *K* folds number. The default *C* and *gamma* parameters was used to train the SVM model. A detailed hyperparameter tuning of these two parameters was not carried out which could have improved its performance both in terms of accuracy and AUC.

The third classifier used was RF (Figures 12 and 13). Following the performance of LR and SVM on train and validation set respectively, the RF trained model to work better on train set in comparison to validation set. The largest margin between train and validation accuracies was around 2% and the average margin considering all three-fold variations was around 1% which can be seen from Figure 12. According to results from Figure 13, there was a large difference between train AUC and validation AUC with around 30% for 10-fold cross-validation, 35% on 20-fold data and 30% for the 30-fold variation. This large difference can indicate a possibility of overfitting. The performance of random forest on the 30% testing data (out of the total available data) was not impressive. The 30-fold cross-validation achieved the highest AUC of 1.0. In case of random forest, there are certain parameters like number of trees and number of splits at a node. In this case, default parameters offered by sklearn was used (very low value of 1 and 2 for number of nodes and splits respectively were used). An improvement in the algorithm's performance could have been seen as a result of hyperparameter tuning of these parameters which was not done in this study. Also, if we had increased the value of these parameters, prediction may have improved. On the test set, RF achieved the highest accuracy of around 99.4% in comparison to LR and SVM. This score was 25% larger compared to LR and 54% larger compared to SVM. But the AUC score seems to be comparable to the other algorithms. The AUC decreases by 15% when increasing the k-value from 10 to 20. However, when further increasing the k value to 30, the score increases by 8.5%.

4.1 Summary

Overall, logistic regression seemed to outperform support vector machine and random forest by taking into account accuracy and AUC scores. Christodoulou et. al and team had conducted a clinical study which showed that for classification problems on medical data, machine learning algorithms did not significantly outperform logistic regression [28]. The results obtained in this study were in line with their findings. We found that the importance, or not, of the features depended on the model. Ideally, we would have liked to find a strong signal for certain specific features. However, this study still advanced the study of p53 in identifying only 2D features for inclusion. Future work should focus on understanding the distribution of feature and how they are important for each model. Removing the less important feature and training the models and comparing the results with models trained with 100 features also might help us understand more about these features.

Bibliography

[1] National cancer institue, "Cancer Statistics," 2018.

[2] S. A. Danziger *et al.*, "Predicting positive p53 cancer rescue regions using Most Informative Positive (MIP) active learning," *PLoS Comput. Biol.*, 2009.

[3] S. A. Danziger *et al.*, "Functional census of mutation sequence spaces: The example of p53 cancer rescue mutants," *IEEE/ACM Trans. Comput. Biol. Bioinforma.*, 2006.

[4] S. A. Danziger, J. Zeng, Y. Wang, R. K. Brachmann, and R. H. Lathrop, "Choosing where to look next in a mutation sequence space: Active Learning of informative p53 cancer rescue mutants," in *Bioinformatics*, 2007.

[5] T. Soussi, "The p53 tumor suppressor gene," *Adv. Genome Biol.*, 1995.

[6] L. Guruprasad, "Protein Structure," *Resonance*, 2019.

[7] A. C. Joerger and A. R. Fersht, "The tumor suppressor p53: from structures to drug discovery.," *Cold Spring Harbor perspectives in biology*. 2010.

[8] A. . Petitjean, S. . Mathe, E.; Kato, C. . Ishioka, S. V. . Tavtigian, P. . Hainaut, and M. Olivier, "Impact of Mutant p53 Functional Properties on TP53 Mutation Patterns and Tumor Phenotype: Lessons from Recent Developments in the IARC TP53 Database," *Hum. Mutat.*, 2006.

[9] R. Geetha Ramani and S. G. Jacob, "Prediction of P53 Mutants (Multiple Sites) Transcriptional Activity Based on Structural (2D&3D) Properties," *PLoS One*, 2013.

[10] J. A. Littlechild, "Protein structure and function," in *Introduction to Biological and Small Molecule Drug Research and Development: Theory and Case Studies*, 2013.

[11] D. S.A. *et al.*, "Functional census of mutation sequence spaces: the example of p53 cancer rescue mutants," *IEEE/ACM Trans. Comput. Biol. Bioinforma.*, 2006.

[12] R. Luo, L. David, and M. K. Gilson, "Accelerated Poisson-Boltzmann calculations for static and dynamic systems," *J. Comput. Chem.*, 2002.

[13] World Health Organisation, "Latest global cancer data," *Int. Agency Res. cancer*, 2018.

[14] I. Guyon and A. Elisseeff, "Feature Extraction, Foundations and Applications: An introduction to feature extraction," in *Feature Extraction. Studies in Fuzziness and Soft Computing, vol 207*, 2006.

[15] I. Guyon and A. Elisseeff, “An Introduction to Feature Extraction,” in *Feature Extraction*, 2008.

[16] F. Pedregosa *et al.*, “Scikit-learn: Machine Learning in Python Gaël Varoquaux Bertrand Thirion Vincent Dubourg Alexandre Passos PEDREGOSA, VAROQUAUX, GRAMFORT ET AL. Matthieu Perrot,” *J. Mach. Learn. Res.*, 2011.

[17] I. Guyon, “A scaling law for the validation-set training-set size ratio,” *AT&T Bell Lab.*, 1997.

[18] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “SMOTE: Synthetic minority over-sampling technique,” *J. Artif. Intell. Res.*, 2002.

[19] N. Japkowicz, “The Class Imbalance Problem: Significance and Strategies,” in *Proceedings of the 2000 International Conference on Artificial Intelligence*, 2000.

[20] R. Blagus and L. Lusa, “SMOTE for high-dimensional class-imbalanced data,” *BMC Bioinformatics*, 2013.

[21] L. Regression, “Logit Models for Binary Data,” *Bernoulli*, 1978.

[22] Y. Tang, Y. Q. Zhang, and N. V. Chawla, “SVMs modeling for highly imbalanced classification,” *IEEE Trans. Syst. Man, Cybern. Part B Cybern.*, 2009.

[23] B. Schölkopf, “An Introduction to Support Vector Machines,” in *Recent Advances and Trends in Nonparametric Statistics*, 2003.

[24] L. Breiman, “Random Forrests,” *Mach. Learn.*, 2001.

[25] Y. Qi, “Random forest for bioinformatics,” in *Ensemble Machine Learning: Methods and ApplicatiOns*, 2012.

[26] H. Peng, F. Long, and C. Ding, “Feature selection based on mutual information: Criteria of Max-Dependency, Max-Relevance, and Min-Redundancy,” *IEEE Trans. Pattern Anal. Mach. Intell.*, 2005.

[27] C. Elkan and K. Noto, “Learning classifiers from only positive and unlabeled data,” in *Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD 08*, 2008.

[28] E. Christodoulou, J. Ma, G. S. Collins, E. W. Steyerberg, J. Y. Verbakel, and B. Van Calster, “A systematic review shows no performance benefit of machine learning over logistic regression for clinical prediction models,” *Journal of Clinical Epidemiology*. 2019.