**IDENTIFICATION OF PATHOGEN-KILLING PEPTIDES**

**USING MACHINE LEARNING**

**A Project Report for CA1**

**Submitted to**

**Dr. SAGAR PANDE**

**SCHOOL OF COMPUTER SCIENCE AND ENGINEERING**

**LOVELY PROFESSIONAL UNIVERSITY**

****

**Submitted by**

**Name: Neil Sagar Bhosale**

**Registration Number: 11902386**

**Section: KM055, Roll No.: 56**

**Date: 30th of March 2021**

**TABLE OF CONTENTS:**

|  |  |  |
| --- | --- | --- |
| **Chapter No.** | **Title** | **Page**  **No.** |
| **1** | **Introduction** | **3** |
| **2** | **Underlying Biological Concepts of The Project** | **6** |
| **3** | **Machine Learning Paradigms Used in The Project** | **4** |
| **4** | **Tools/Software Description** | **5** |
| **5** | **Steps/Flow of the project, Circuit description** | **6** |
| **6** | **Results** | **10** |
| **7** | **Conclusion and Future Scope** | **16** |
| **8** | **References/Bibliography** | **19** |

CHAPTER 1

INTRODUCTION

**1.1 Overview and importance of project:**

In **medical science**, there is a very urgent need to find new and better medicinal chemicals that will provide more effective cures. Every year, many new and mutated forms of pathogens (microbes) are found in sick and deceased individuals. Thus, antimicrobial resistance is an urgent and global health problem as existing drugs are becoming ineffective against the treatment of microbial infections.

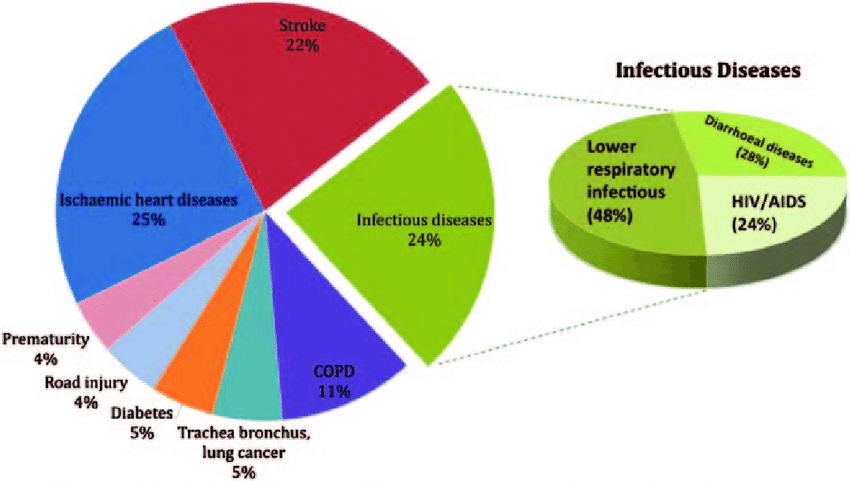


Figure: A major number of deaths are caused by **infectious diseases caused by microbes**

Peptides are some of the bio-molecules that show resistance against these harmful foreign pathogens by stopping their growth or outright killing them. Hence, this makes them an essential part of pharmaceutical drugs. When the genomics researchers find out the amino-acid structure and composition of a new pathogen, **new peptides need to be tested quickly** to find out which one can kill the new microbe.

**Antimicrobial peptides (AMPs)** are a valuable source of antimicrobial agents and a potential solution to the multi-drug resistance problem (in which, after continuous administration of a drug to a long-term patient, the patient becomes resistant to it and a new drug is required to treat the same illness and with the same effectiveness).

In particular, **short-length AMPs** have been shown to have enhanced antimicrobial activities, higher stability, and lower toxicity to human cells. In this project I will be analyzing 2 types of short length AMPs, single amino acid chain (by composition) and dipeptide chain.

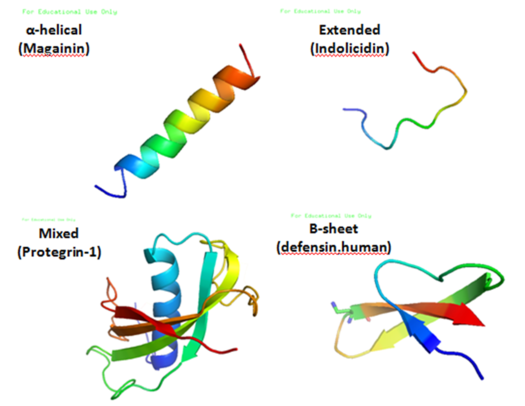


Figure: Various anti-microbial peptides that can be identified by the machine learning work in this project.

Machine learning techniques are used in these research projects. This area of research comes under the sub-domain of computational biology.

In this project, a very important part of the above drug-discovery research has been implemented using machine learning. Namely: Identifying the amino-acid compositions and dipeptide compositions that can kill the microbes, out of the known peptide chains.

2 datasets procured from peptide research projects have been procured and used in this project. The 1st dataset consists of peptide chains that are known to effect microbes and the sum total of the contents of the other dataset is known to not have any effect. This forms the positive and negative classes of the binary classification that has been done in this project.

Further, these ascii based datasets have been processed by me into csv files by using custom functions that turn the peptide data into **quantitative compositional data that can be** fed to the several machine learning models to find the best one for anti-microbial peptide identification.

**1.2 Objectives of the project:**

* Create a model-feedable dataset by making functions to form csv data out of ascii/txt data.
* To build a machine learning model that, when given a new peptide chain, can predict whether it will be effective on killing harmful microbes or not (binary classification).
* To evaluate the model using various performance metrics.
* To try 3 different ML algorithms with various hyperparameters and choose the one with highest accuracy.
* To find short accuracy results on 30 ML models using LazyClassifier tool.

CHAPTER 2

UNDERLYING BIOLOGICAL CONCEPTS OF THE PROJECT

**2.1: Microbes**

Microbes are living organisms of microscopic size that exist in single celled form or in a colony of cells. It was discovered only in the 1880s [Robert Koch et al] that microbes exist and can be harmful as well as essential for the life cycle on planet Earth.

The microbes of interest in this project are harmful microbes. Microorganisms are the causative agents (pathogens) in many infectious diseases. The organisms involved include **pathogenic bacteria**, causing diseases such as **plague, tuberculosis and anthrax**; protozoan parasites, causing diseases such as **malaria**, **sleeping sickness**, **dysentery** and toxoplasmosis; and also, fungi causing diseases such as **ringworm, candidiasis or histoplasmosis**.

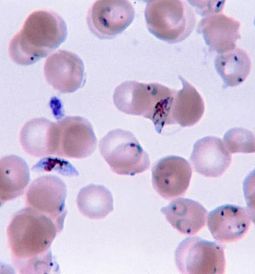


Figure 2.1: Disease causing parasitic microbes as seen under the microscope.

All of the above types of microbes can be resisted or downright killed by **peptides.** These peptides are built by amino acids.

**2.2 Amino acids (Features for prediction)**

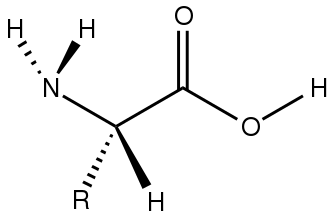


Figure 2.2: Structure of an L amino acid

Amino acids are organic compounds that contain amino and carboxylate (as shown above) groups, along with a side chain **(R group) specific to each amino acid**.

Amino acids by themselves are not effective on the harmful microbes. But, when these amino acids form bonds and chains with one another, **they become peptides.** These peptides can then be very powerful against the microbes or completely useless, entirely based upon the **combinations and permutations** of the below amino acids:

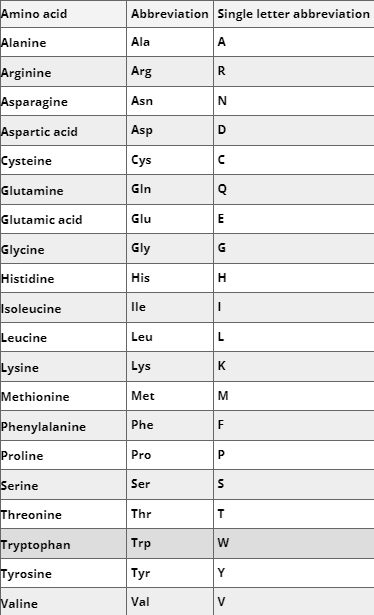


Figure 2.3: These are the 20 types of amino acids (with their single letter abbreviations), the various combinations of which can be powerful for making anti-pathogenic medicines.

There are 20 types of amino acids. **In this project** I have formed two datasets, 1st: with single amino acid composition in each peptide chain and 2nd: with dipeptide combinations that are formed in each peptide chain.

**2.3 Peptides (Rows of the custom formed dataset)**

Peptides are the central actor in this project. Peptides are short chains of the above-define amino acids linked together by peptide bonds.

The peptides of concern in this project are **Anti-Microbial-Peptides**:

Antimicrobial peptides (AMPs), also called host defense peptides (HDPs) are part of the innate immune response found among all classes of life. But, when the body cannot produce these peptides, **they have to be artificially administered by outside means**, which is the aim of this project.

In this project, I have analyzed **single amino acid chains** and **dipeptide (bonded) chains** that can be formed from a given peptide composition. There are 20 amino acids known to scientists. So, to form dipeptide chains (chains with two amino-acids at each node, bonded with a peptide bond), there can be 400 different dipeptides (repeated over and again) in a single peptide chain.

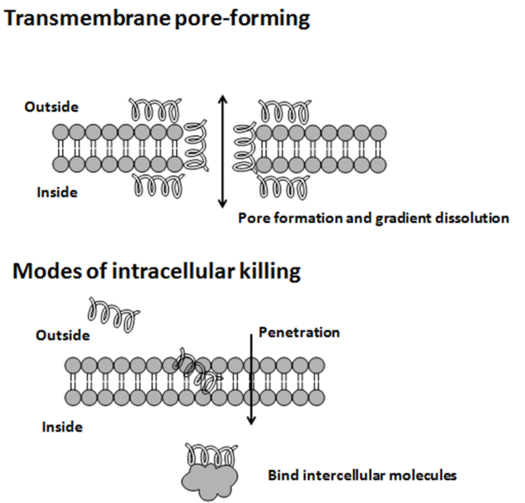


Figure 2.4: Shows Dipeptide chains (which are analyzed in the second custom dataset of this project)

CHAPTER 3

THE MACHINE LEARNING PARADIGMS USED IN THE PROJECT

**3.1 Proposed Algorithms:**

The central problem of this project is to classify peptides into microbicidal and non-microbicidal ones. This is clearly a **binary classification problem** and can be solved using the following supervised learning algorithms (in that order, retrospectively): Support Vector Machine, K-Nearest-Neighbors, Decision tree, Random Forest Algorithm. Then, various model-evaluation metrics can be used to find the accuracy and scope for improvement in the model

**3.2 Support Vector Machine**

Support Vector Machine or SVM is a supervised and linear Machine Learning algorithm most commonly used for solving classification problems and is also referred to as Support Vector Classification. The objective of SVM is to draw a line that best separates the two classes of data points. (This is very useful as the given problem is a binary classification one.)

In SVM, the line is determined by the margins and the support vectors.

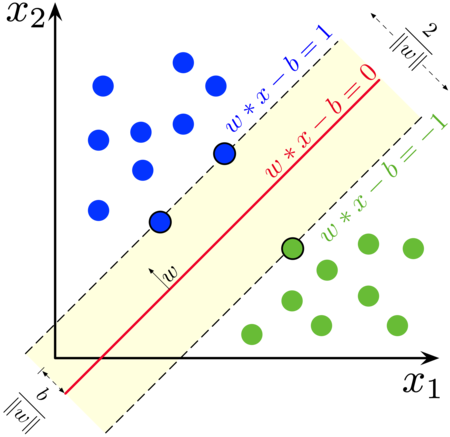
The more the margin the better the classes are separated.

The points that lie on the margins are called support vectors as they contribute to the margins and hence the classifier itself. These support vectors are simply the data points lying closest to the border of either of the classes which has a probability of being in either one.

The SVM then generates a hyperplane which has the maximum margin, that separates the two classes which is at an optimum distance between both the classes.

In case of more than 2 features and multiple dimensions, the line is replaced by a hyperplane that separates multidimensional spaces.

Implementation of linear SVM for binary classification:



We are given a training dataset of n points of the form:



We want to find the "maximum-margin hyperplane" that divides the group of points xi for which yi=1 from the group of points for which yi=0 so that the distance between the hyperplane and the nearest point xi from either group is maximized. Any hyperplane can be written as the set of points **x** satisfying:



If the training data is linearly separable, we can select two parallel hyperplanes that separate the two classes of data, so that the distance between them is as large as possible.



Anything on or above this boundary will have class label 1



Anything on or below this boundary will have class label 0 or -1

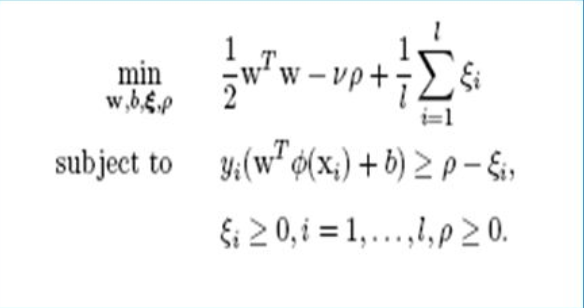
**3.3 Nu SVC**

Nu-SVC: Nu-Support Vector Classification

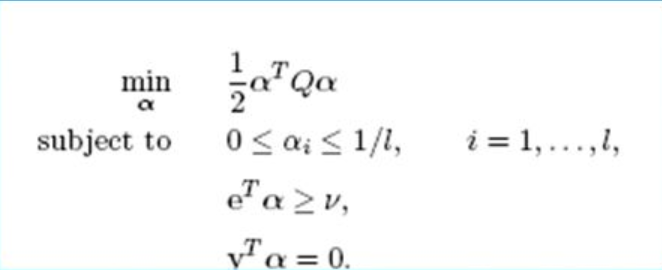
**Nu-SVC** is similar to Support Vector Classification/Machine but it uses a new parameter nu which controls the number of support vectors and training errors.

The parameter **nu** is an upper bound on the fraction of training errors and a lower bound of the fraction of support vectors.

The value of nu should be in the interval (0,1] (0 non-inclusive and 1 inclusive)



Nu calculation shown above and below is the nu-SVP margin calculation



**3.4 K NEAREST NEIGHBOURS**

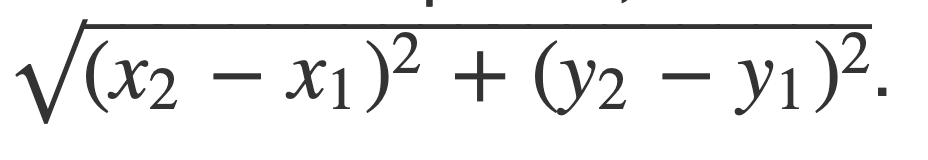
This is one of the Supervised learning algorithms mostly used for classification of data on the basis how it’s neighbors are classified. KNN stores all available cases and classifies new cases based on a similarity measure. K in KNN is a parameter that refers to the number of the nearest neighbors to include in the majority voting process.

How K is chosen:

Sqrt(n), where n is a total number of data-points (if in case n is even we have to make the value odd by adding 1 or subtracting 1 that helps in select better). KNN is a lazy learner algorithm.

Implementation

To find the distance between any two points, Euclidean distance measure is used. Manhattan measure and hamming measure can also be used.



Similarly, we find out all distance one by one.

We calculate K factor for each row and find the nearest neighbour to that value. The row is then assigned to that target class which is nearest.

**3.5 Decision tree Algorithm**

There are various subtypes in the decision tree algorithm family. But, when we implement it using Sci-Kit Learn library, the library chooses on it’s own which would be best to use. The types of Decision tree learning algorithms are:

* ID3
* C4.5
* C5.0
* CART
* Etc

The final aim of all these algorithms is to eventually arrive at a tree that will have leave nodes that will be classifiers. It approximates discrete-valued target functions while being robust to noisy data and learns complex patterns in the data.

They classify the instances by sorting down the tree from root to a leaf node that provides the classification of the instance.

All these algorithm differ only in the manner in which they go about the problem of choosing a node that will be further split (a parent node). There are various metrics available that do it. In this project, clearly the metric of GINI INDEX was used (which has been visualized for random forest algorithm in the project code and later in the report)

The main idea of a decision tree is to identify the features which contain the most information regarding the target feature and then split the dataset along the values of these features such that the target feature values at the resulting nodes are as pure as possible.

Gini Index: It is calculated by subtracting the sum of squared probabilities of each class from one. It favors larger partitions and easy to implement whereas information gain favors smaller partitions with distinct values.



A feature with a lower Gini index is chosen for a split.

The classic CART algorithm uses the Gini Index for constructing the decision tree.

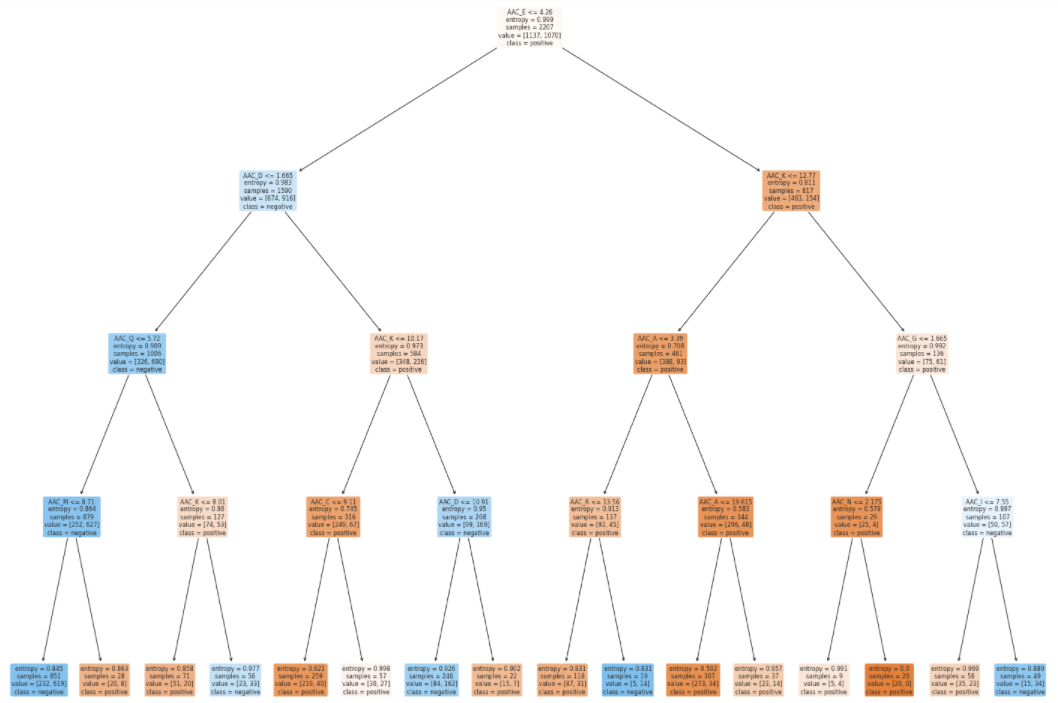
OTHER METRICS THAT CAN BE USED TO SELECT NODES:

* Entropy:



* Information gain:



****

**Figure: Visualization of the decision tree made for Amino Acid Composition Dataset**

**3.6 Random Forest Algorithm**

Random forest is a Supervised Machine Learning Algorithm that is used widely in Classification. It builds decision trees on different samples and takes their majority vote for classification and average in case of regression.

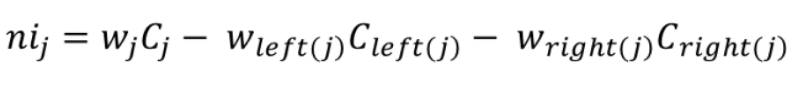
This makes it an *ensemble learning technique* for classification.

it can handle the data set containing continuous variables as in the case of regression and categorical variables as in the case of classification. It performs better results for classification problems.

It is an extension on the decision tree algorithms. Decision trees show heavy overfitting and are not optimal for datasets with many features as they tend to overfit. (This project also uses Decision trees and overfitting is certainly observed) Random forests (RF) construct many individual decision trees at training.

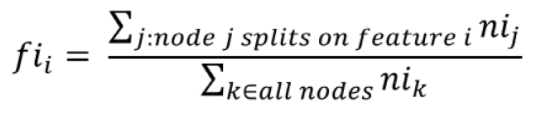
**Implementation in Scikit-learn**

For each decision tree, Scikit-learn calculates a nodes importance using Gini Importance, assuming only two child nodes (binary tree):



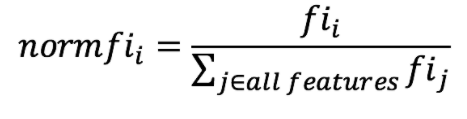
* ni sub(j)= the importance of node j
* w sub(j) = weighted number of samples reaching node j
* C sub(j)= the impurity value of node j
* left(j) = child node from left split on node j
* right(j) = child node from right split on node j

The importance for each feature on a decision tree is then calculated as:

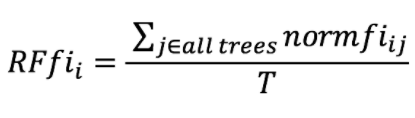


* fi sub(i)= the importance of feature i
* ni sub(j)= the importance of node j

This result is normalized by:



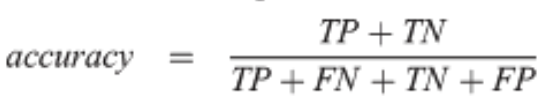
The final feature importance, at the Random Forest level, is it’s average over all the trees. The sum of the feature’s importance value on each trees is calculated and divided by the total number of trees:



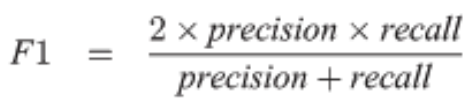
**EVALUATION METRICS USED:**

**Accuracy Score:**

This is a metric for direct ratio of right classifications to misclassifications:



**FI SCORE:**



**CONFUSION MATRIX:**

We aim to maximize the value of the diagonal elements (left to right diagonal) in a confusion matrix as these values represent the correct classification that has been made by our model.

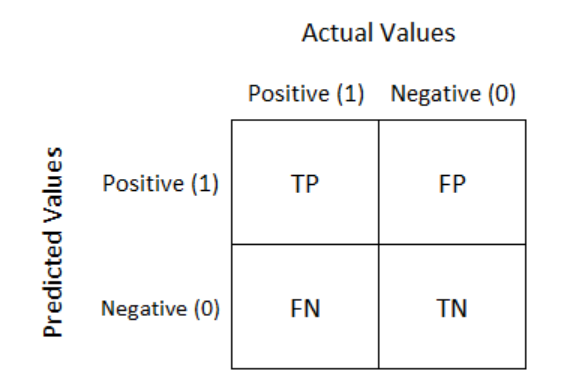


Figure 3.: A blueprint for a confusion matrix

CHAPTER 4

TOOLS AND LIBRARIES USED (SOFTWARE DESCRIPTION)

**4.1 Conda:**

Version: Miniconda3-py37\_4.8

The purpose of this technology is to be able to use the CD-HIT feature

**4.2 Python sys module:**

This helps to manipulate the runtime environment

**4.3 Pfeature Library**

Pfeature is an experimental library built in the systems biology research group of IIITD.

This is a powerful library that provides various functions to manipulate raw peptide chain data in such a way that all the possible combinations of peptide bonded chains can be formed.

These peptide chains can be for the entire raw chain or only using parts of it (The start of the chain or the end of it).

70000 such different permutations and peptide bonded combinations can be formed out of these pfeature functions from only a bare raw text file with amino acid chain Initials.

**4.4 CD-HIT**

Cd HIT is a conda functionality that

**4.5 Pandas**

All data manipulation rests on Pandas.\

**4.6 Seaborn, Matplotlib**

All data visualization has been done using seaborn and matplotlib for both feature selection phase and result visualization phase.

**4.8 LazyClassifier**

This is a powerful tool that aides in model selection. It has been used by me in this project for finding the top 5 performing algorithms both on the training and testing datasets. Based on that, I decided which classifiers to implement.

**4.8 Sci-Kit Learn**

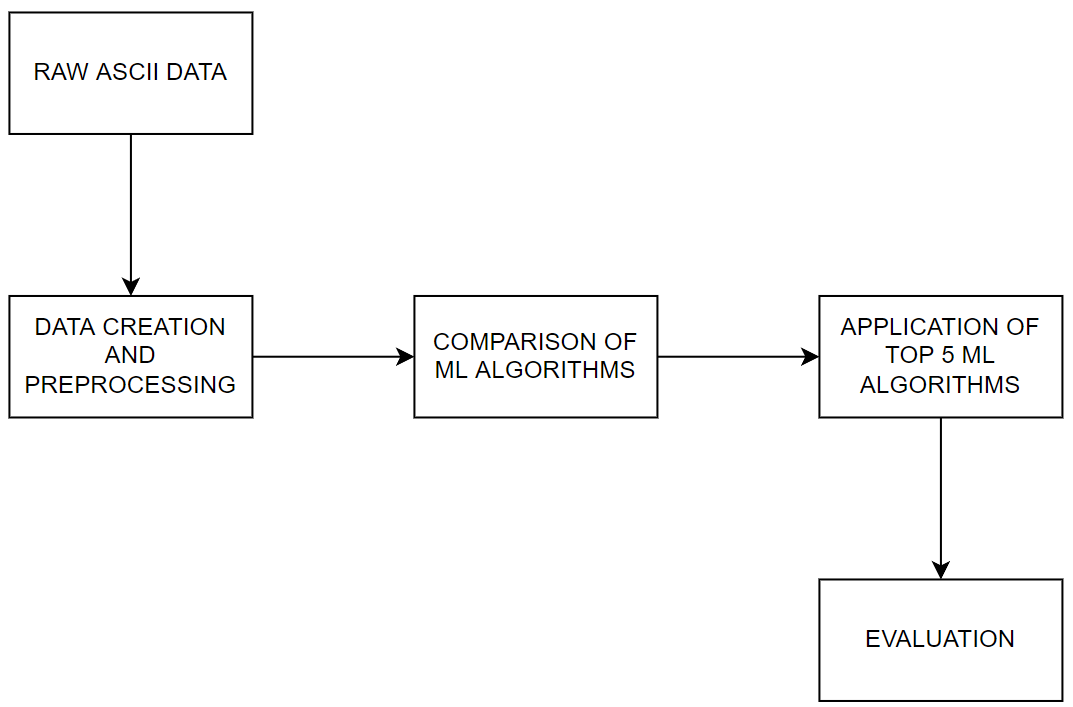
This library is at the base of all the machine learning activities performed in this project, whether that may be splitting data, training it, testing it, making predictions or evaluating the already trained model.

It has in-built functions for each machine learning algorithm and method, thus allowing ease of use and more focus on Data preprocessing rather than training.

CHAPTER 5

FLOW/STEPS OF THE PROJECT AND CIRCUIT DESCRIPTION

The general flow of the project is as follows:



**I WILL FURTHER ELABORATE ON EACH OF THE ABOVE STEPS:**

**5.1 RAW ASCII DATA TO PREPARED DATA**

The data was not in the form of a csv file (relational)

The data was in the following form in an ascii text file:

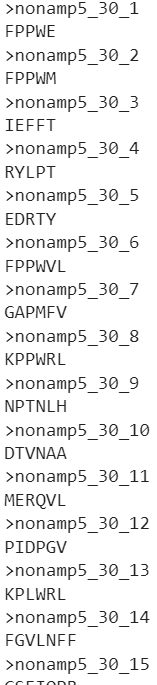
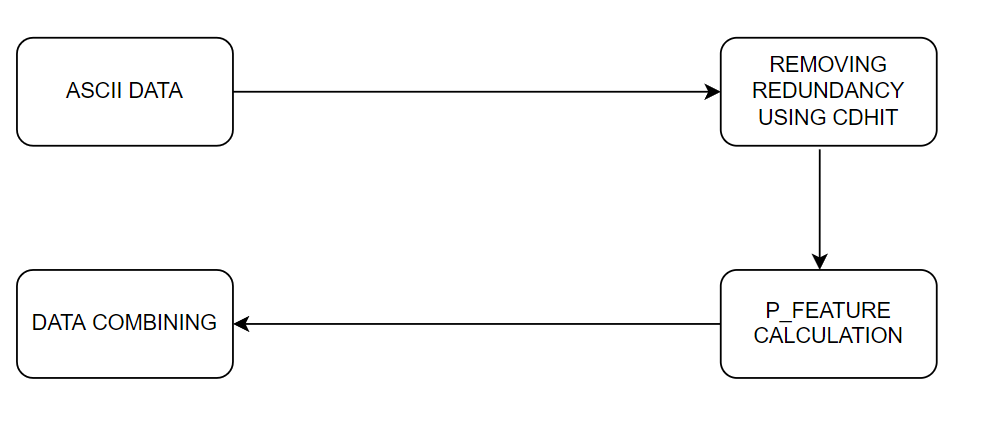


Figure: A snippet from the negative dataset

In this dataset, the number of peptide chain is given, followed by the actual Amino acid sequence in that peptide. Eg: peptide 14 is FGVLNFF. This is from the negative datasets.

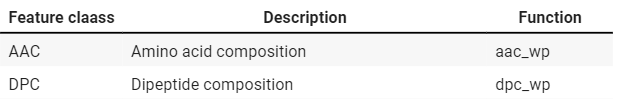
NOTE: The positive peptides and negative peptides are in separate classes. They will have to be processed and combined as follows 🡪

To create the csv dataset for classifying peptides using “Amino Acid Composition”, I used the **Pfeature** library



First: All the un-required peptide chains that are similar to other chains are removed. In this case 50 peptides (columns) are dropped from the positive and 175 peptides (columns) are dopped from the negative peptide files.

Second: The following functions are available in Pfeature to calculate the values of each column in our new dataset.



Custom function to calculate AAC features:

1. Change the file extension of the file from txt to csv
2. Use aac\_wp() function (shown above) to calculate the composition of each amino acid in a given peptide chain.
3. Form dataset with Individual Amino Acids Composition as columns and each peptide as Rows. (1337 rows × 20 columns)
4. Read this newly formed dataset using pd.readcsv() function.

Third: Calculate feature for both positive and negative classes + combines the two classes + merge with class labels. Using the following custom-made function:

1. Takes attributes: positive dataset, negative dataset, above created custom function.
2. Create class labels (positive and negative)
3. Vertically concatenate positive and negative datasets.
4. Horizontally concatenate Feature variables and class variables.

WITH THE SUM CUMULATION OF THE ABOVE PROCESSES, MY DATA SET WITH THE AMINO \_ ACID \_M COMPOSITION OF EACH PEPTIDE IS READY:

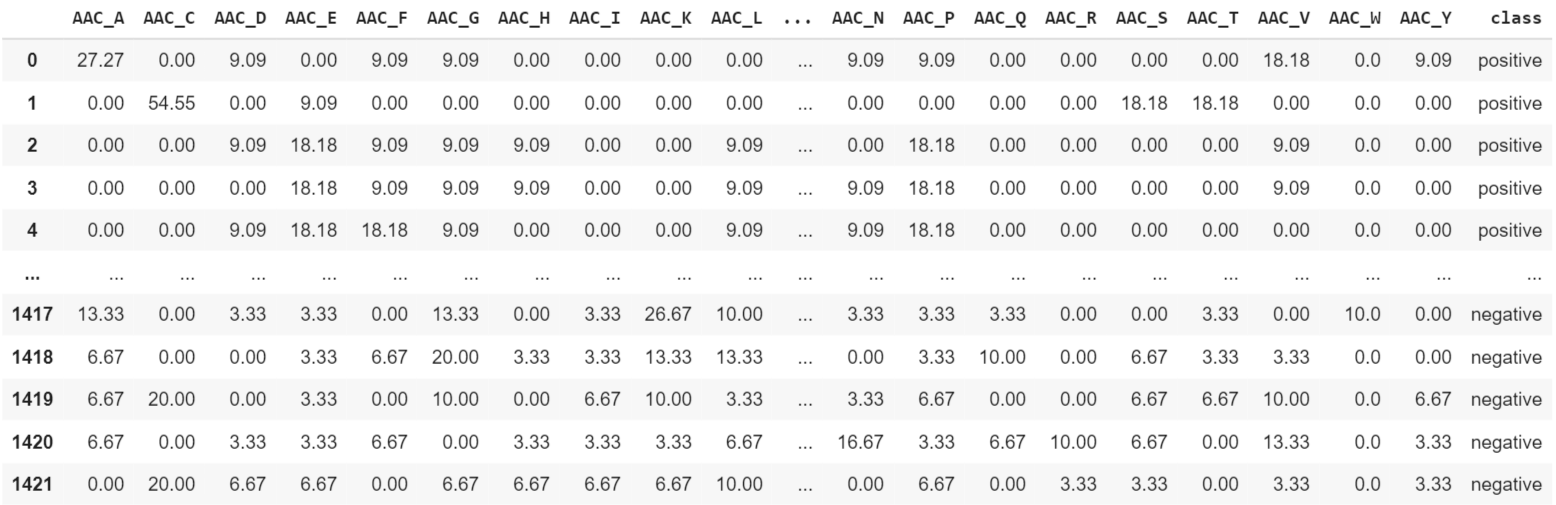
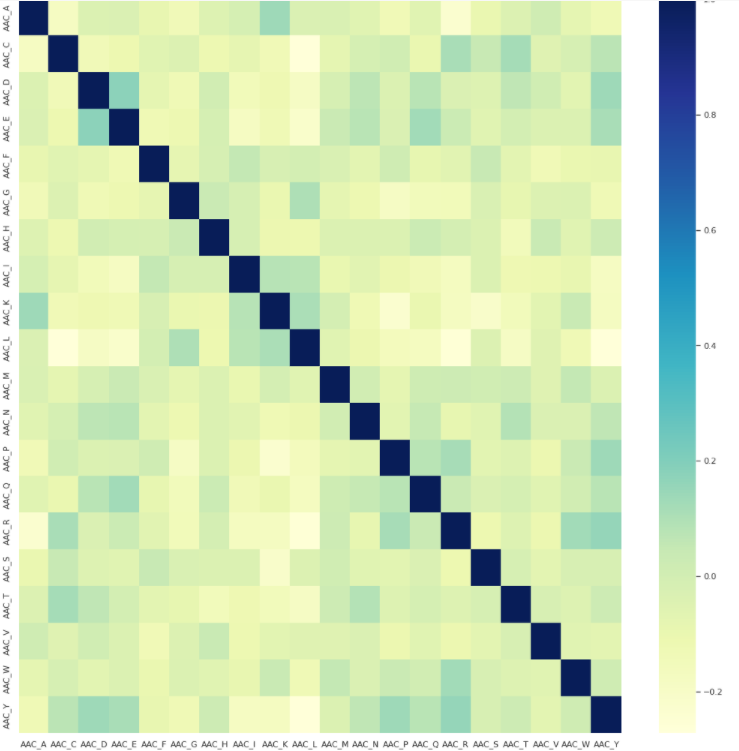


Figure: ataset is ready for further preprocessing.

**5.2 DATA PREPROCESSING**

Dividing the data into Features (X) and Target (y)

Apart from the CDHIT dropping of peptide chains, the formed aac dataset was checked for variance and correlation:

****

**Figure: Correlation of all the amino-acids (features)**

As seen above, D and E amino acids are fairly highly correlated, so one of them can be dropped. Same is true for Y and R amino acids and some other. But dropping might be less or avoided all together as no real model-degrading correlation is seen in the above heatmap.

Calculation of variance threshold using VarianceThreshold function in sklearn.feature\_ selection. Using this to Drop all columns with variance less than 0.1.

(Very few columns are dropped as CD-HIT has already done some essential preprocessing)

**Lastly, performing train\_test\_split using sklearn.model\_selection. (20% testing and 80 percent training data)**

**5.3 COMPARISON OF ML ALGORITHMS:**

In this project I have used **lazypredict** and **LazyClassifier** to compare 27 ML algorithms:



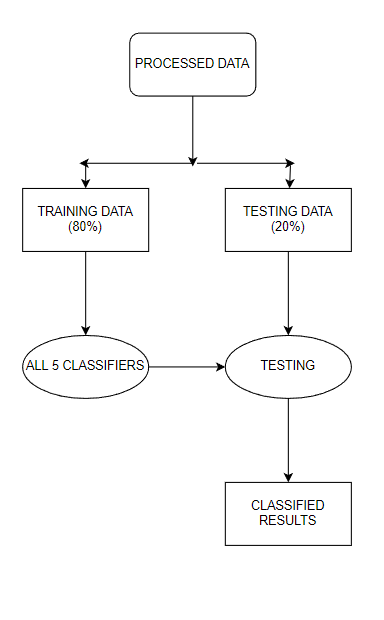
Figure: The performance of 27 different ML algorithms on the training dataset



Figure: Show the average performance of all 27 algorithms on the testing set.

BASED ON ABOVE RESULTS AND SIMPLICITY OF IMPLEMENTATION, THE FOLLOWING 5 ALGORITHMS ARE CHOSEN:

1. Support Vector Machine, 2. Nu-SVM, 3. K Nearest Neighbors, 4. Decision Tree, 5. Random Forest



CHAPTER 6

RESULTS AND HYPER PAREMETER TUNING

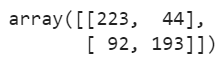
Given below are the results of each of the 5 ML algorithms used and their comparison/Visualization.

**6.1 Result of SVM:**

Accuracy on test data: 75.362 percent

F1 SCORE: 76.632 percent

Confusion Matrix

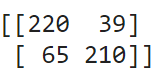


**6.2 Result of Nu-SVC:**

Accuracy on training data: 85 percent

Accuracy on testing data: 79.345 percent

Confusion Matrix



**6.3 Result of K-Nearest-Neighbor**

Accuracy Score: 74.567 percent

F1\_score: 72.958 percent

Confusion Matrix:



**6.4 Result of DECISION TREE LEARNING**

Accuracy Score: 69.993 percent

F1\_Score: 70.506 percent

Confusion Matrix:



**6.5 Result of RANDOM FOREST**

Preliminary result:

Accuracy Score: 73.913 percent

F1\_Score: 76.366 percent

**Results after performing Hyper parameter Tuning Using Grid-Search CV:**

Accuracy Score: 81.733 percent

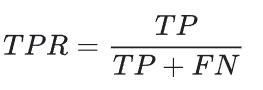
F1\_Score: 82 percent

**NOTE:** In this project, I am working with RAW real world BIOLOGICAL DATA. In this case, I have created the dataset from a raw ascii file of amino acid chain sequences. This Even the above accuracies are hugely corroborative of the hypothesis of the-zusefulness of the given classifier on a given dataset.

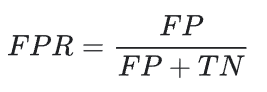
Hence, from the above findings we can attest that Random-Forest algorithm gives me the best results. Performing further analysis of the results:

An ROC curve (receiver operating characteristic curve) is a graph showing the performance of a classification model at all classification thresholds. This curve plots two parameters:

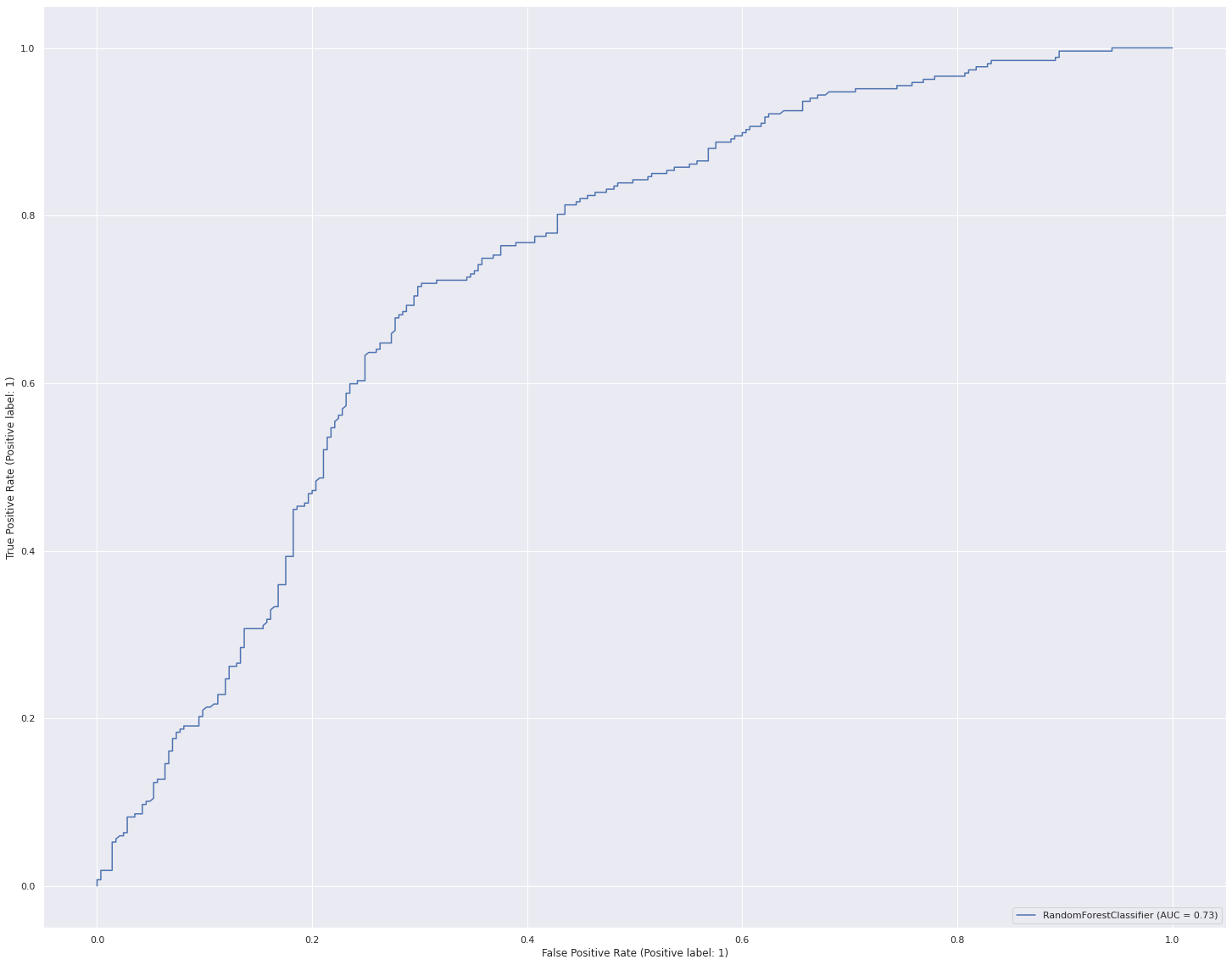
True Positive Rate



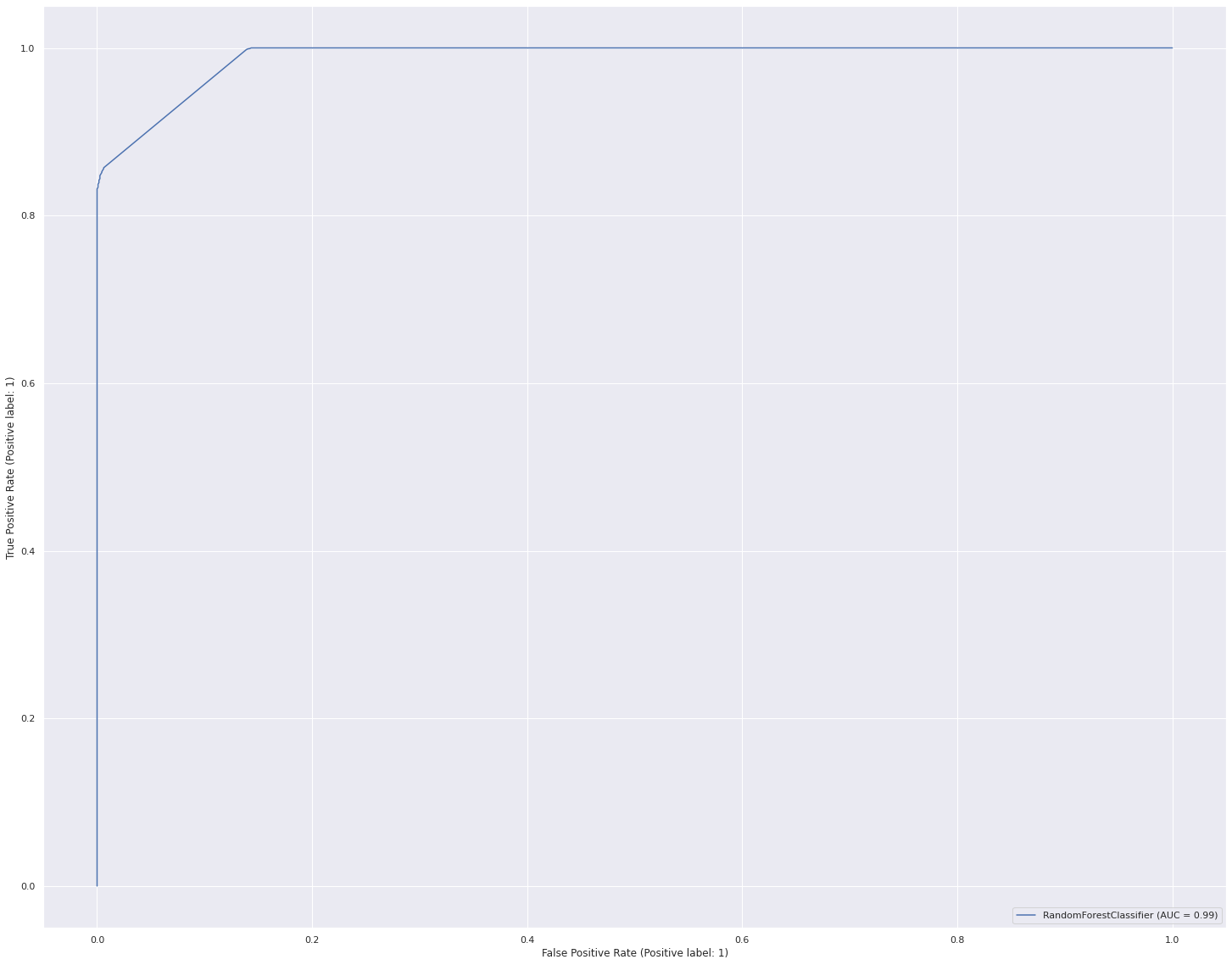
False Positive Rate



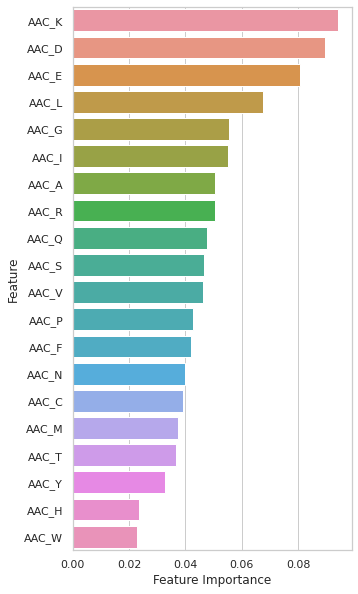
An ROC curve plots TPR vs. FPR at different classification thresholds. Lowering the classification threshold classifies more items as positive, thus increasing both False Positives and True Positives.



ROC CURVE FOR Testing DATA



ROC CURVE FOR TRAINING DATA



Above shown is the most important visual metric in the used Random-Forest Algorithm.

This Bar Plot is instrumental in finding which Features have the highest GINI INDEX (plotted in descending order).

CHAPTER 7

CONCLUSION AND FUTURE SCOPE

**7.1 Conclusions made from the project**

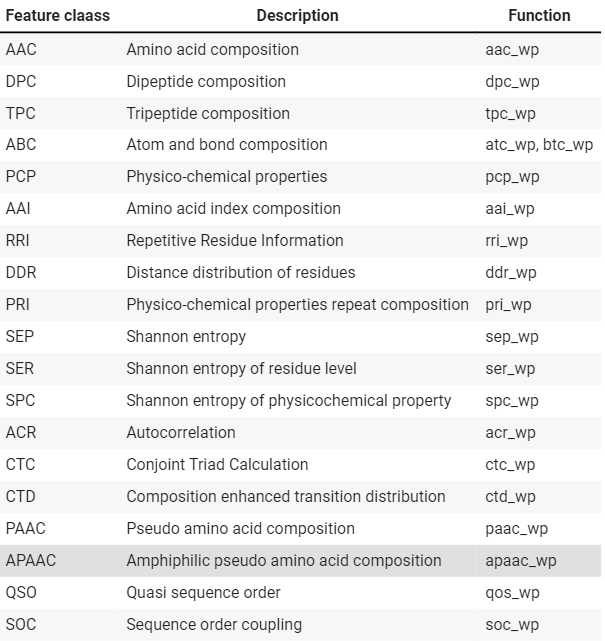
This project was instrumental in understanding how both the combinations and individual compositions of Amino acids in peptide chains affects the power of peptide chains in killing microbes or being completely inactive against them.

This makes it very clear that K, D and E Amino Acids are the most important Amino Acids when it comes to antimicrobial activities.

Y, H and W amino acids are least useful and their presence in the amino acid chain does not matter much.

**7. Future Scope**

Similar to the process performed for Amino acid composition, the same peptide chain ascii file text data can give rise to relational datasets of the following kind:



Given above are the multitudinous functionalities available in the pfeature library [Raghava Et Al]

These are all the peptide-bonded combinations that can be formed from the given chains. The functions stated next to each combination can be used to quantitatively state the composition of each combination formed in a given raw peptide chain and thus form a value for that column in the peptide row.

For example: In the 2nd combination stated in the above table: DPC (dipeptide composition):

* There are a maximum of 20 amino acids found in nature.
* Dipeptide bonds hold 2 of these together at a node (whether unique or repeated)
* So, the number of features in this new dataset will be equal to the total number of dipeptide combinations possible. i.e:
* 20\*20 = 400 Feature columns (for each peptide)

The same can be done for all other combinations as well.

The pfeature library is capable of producing 70,000 such compositional combinations, which makes it a very powerful tool in the realm of polypeptide-related systems biology applications and judging the various behaviors of peptides in the world.

This is not only limited to microbicidal properties by many other realms such as:

* Skincare products
* Hair growth products
* Antidepressants
* Anti-Indigestion (Non-microbial)

Any of the above research areas are open for the peptide behavioural researchers in the future.