An Analysis Report on Advanced approach to identify Antimicrobial Peptides and their function types for Penaeus through Machine Learning Strategies

Durga Prasad Rangavajjala
Master of Computer Science
University of Ottawa,
Ottawa, Canada
300123236
drang041@uottawa.ca

***Abstract*—Antimicrobial peptides (AMP's) are important components to innate immune system and protect host from various pathogenic bacteria. This paper aims to analyze the report "An advanced approach to identify antimicrobial peptides and their function types for penaeus through machine learning strategies" [1] by applying Supervised Machine Learning concepts. The analysis starts with data preprocessing then continued with normalization, sampling and feature selection on AMP data then builds classification models using pandas, sklearn, NumPy ,then analyze and compare with distance based, tree based, linear based and ensemble learning and visualize performance accuracies and make effective classification of AMP and also analyze its function type which is referred as Multi Label Classification in the second part of reference paper. The result of the paper is the binary classification analysis to predict whether the given feature set is AMP and also classify the AMP function type and compare the results with the reference paper.**

Keywords— Antimicrobial peptides, Classification model, Multi Label Classification, supervised machine learning, data preprocessing, normalization, sampling, feature selection, rule-based classification, tree based, ensemble learning, linear classification, distance based

I. INTRODUCTION

The aim of this project is to analyze the paper – "An advanced approach to identify antimicrobial peptides and their function types for Penaeus through machine learning strategies" [1], and experiment with the implementation mentioned in reference paper and additional with other ML algorithms and compare the accuracies and try to make comments on the reference paper.

The AMP classification is a two-step classification problem. In the first step, we try to analyze the given feature set is AMP or not as a Binary Classification Problem proceeding with classification of type of AMP which is a Multi Label Classification problem.

II. DATA SOURCE

In the source paper [1], the authors took raw Penaeus data and then performed data cleaning, integration part and made available open as GitHub repo.
Data source reference URL:
https://github.com/JianyuanLin/SupplementaryData

III. DATA DESCRIPTION

The Analysis is divided into two stages first, we classify whether the given data belong to Anti-Microbial Peptide (AMP) or Non-AMP. AMP classification data set consists of 6990 rows with 188-D feature set constructed from SVM-Prot feature [32, 33], which maps Penaeus peptide sequence to numeric feature vectors and class label to classify whether the feature vector is AMP or not.

In the second stage of analysis deals with data to classify the function type of AMP which consists of 2619 AMP features with 16 different types AMP function types.

IV. DATA ANALYSIS

A. Class Label Distribution

The paper analyzed the data and visualizes the distribution of dataset. In the reference data set, the first step is the binary classification problem but, the class label is found to be highly imbalance where the Non-AMP data dominates over AMP with the ratio nearly 2:1

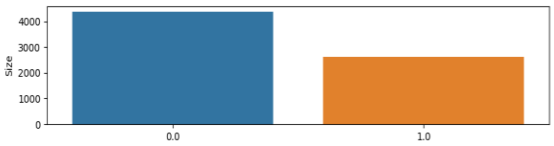

Figure 1: Raw Data Class Label

In the second step, we deal with multi label classification with 16 outcomes where Antibacterial function type dominates over all other labels in the dataset. Out of 2618 AMP data, we have 1297 Antibacterial function types only. Which resembles 50 percent of AMP data in belongs to Antibacterial function type and we have only 4 of 2618 AMP data as Anti-protist function type.

Function	Dataset	Function type	Sequence
AMPs	s_1^{AMPs}	Wound healing	18
	s_2^{AMPs}	Spermicidal	13
	s_3^{AMPs}	Insecticidal	28
	s_4^{AMPs}	Chemotactic	57
	s_5^{AMPs}	Antifungal	593
	s_6^{AMPs}	Anti-protist	4
	s_7^{AMPs}	Antioxidant	22
	s_8^{AMPs}	Antibacterial	1297
	s_9^{AMPs}	Antibiotic	32
	s_{10}^{AMPs}	Antimalarial	25
	s_{11}^{AMPs}	Antiparasital	101
	s_{12}^{AMPs}	Antiviral	125
	s_{13}^{AMPs}	Anticancer	125
	s_{14}^{AMPs}	Anti-HIV	109
	s_{15}^{AMPs}	Proteinase inhibitor	26
	s_{16}^{AMPs}	Surface immobilized	43
	s^{AMPs}		2618
non-AMPs	$s^{non-AMPs}$		4371

Figure 2: AMP data set and its functional types

B. Correlation

Correlation analysis is used to understand the strength between the features. It lies between -1 to +1 and 0 correlation says that there is no relation between the features. The best way to visualize the correlation is heat map [4].

A heat map (or heatmap) is a graphical representation of data where the individual values contained in a matrix are represented as colors [5].

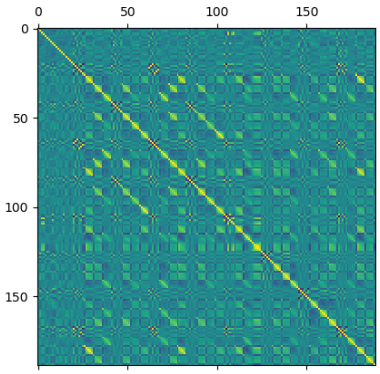

Figure 3: Heat Map Correlation

C. Skewness Analysis

An important task in any statistical analysis is understanding the variability of the data. In this application, we evaluated the property called Skewness.

Skewness is a measure of symmetry, or more precisely, a measure of lack of symmetry. A distribution or data set is symmetrical if it looks the same on the left and right of the center point

$$a_3 = \sum \frac{(X_i - \bar{X})^3}{ns^3}$$

Figure 4: Skew Formula

Most of the features in the data set have the skewness values within the range -0.5 to 0.5 which results in symmetric data and few are moderately and highly skewed [6].

Stage -1 Data Set Skewness analysis

	Skewness		...
0	1.765715	159	-0.470152
1	1.330276	160	-1.633069
2	1.522396	161	-3.604814
3	1.810549	162	2.737349
4	1.604491	163	0.060193
5	2.212492	164	-0.697922
6	4.382993	165	-1.608047
7	1.382738	166	-3.307266
8	1.353007	167	0.230837
9	1.448532	168	0.258186
10	1.598834	169	0.396841
11	1.740086	170	0.519897
12	3.166102	171	0.569339
13	1.769125	172	0.765779
14	2.868514	173	4.632403
15	1.234617	174	1.183293
16	1.693695	175	-0.087567
17	1.119482	176	-1.217685
18	6.334725	177	-3.331878
19	1.824585	178	2.527240
20	0.258186	179	-0.044635
21	0.537892	180	-0.702882
22	0.046968	181	-1.668205
23	0.421510	182	-3.573688
24	0.929400	183	2.449385
25	0.483560	184	0.496350
26	2.527240	185	-0.420031
27	-0.044635	186	-1.358517
28	-0.702882	187	-3.022603
29	-1.668205	188	0.518322

Figure 5: Skewness for dataset-1

Stage -2 Data Set skewness analysis

Skewness	
wound-healing-peptides	5.114143
spermicidal-peptides	6.379924
insecticidal-peptides	6.268579
chemotactic-peptides	3.558991
antifungal-peptides	0.085689
anti-protist-peptides	16.977119
antioxidant-peptides	6.886273
antibacterial-peptides	-1.470148
antibiofilm-peptides	4.794503
antimalarial-peptides	5.348616
Antiparasitic-peptides	2.494195
antiviral-peptides	1.927124
anticancer-peptides	2.223639
anti-HIV-peptides	2.324356
protease-inhibitors	5.384457
surface-immobilized-peptides	4.591944
Feature1	1.417344
Feature2	1.197436
Feature3	1.645926
Feature4	2.330547
Feature5	1.329103
Feature6	2.160636
Feature7	3.852203
Feature8	1.039400
Feature9	1.091350
Feature10	1.335125

	...
Feature159	0.694146
Feature160	-0.367905
Feature161	-1.269333
Feature162	-3.013095
Feature163	2.622246
Feature164	0.663233
Feature165	-0.342768
Feature166	-1.220783
Feature167	-2.482679
Feature168	-0.057830
Feature169	0.234925
Feature170	0.567333
Feature171	0.351009
Feature172	0.516604
Feature173	1.032474
Feature174	4.290664
Feature175	0.880964
Feature176	-0.298519
Feature177	-1.093125
Feature178	-2.416259
Feature179	1.868166
Feature180	0.450310
Feature181	-0.271876
Feature182	-1.290247
Feature183	-3.059462
Feature184	1.803905
Feature185	1.004430
Feature186	-0.122323
Feature187	-0.893953
Feature188	-2.268564

Figure 6: skewness for dataset-2

V. Data Preprocessing

Data preprocessing is a data mining technique that involves converting raw data into an understandable format. Data in the real world are often incomplete, inconsistent and / or lack certain behaviors or trends, and may contain many errors. Data preprocessing is an effective method to solve such problems [7].

In the real world, data is often incomplete: missing attribute values, missing certain attributes of interest, or containing only summary data. Noisy: Contains errors or outliers. Inconsistencies: Include differences in code or name

1. Normalization

Normalization is a technique often used when preparing machine learning data. The goal of normalization is to change the value of a numeric column in a data set to a common scale without distorting the difference in value range. For machine learning, there is no need to standardize every data set. Only required if the functions have different ranges [10].

1.1 Min Max Normalization

Min Max Normalization is the process of converting values with different ranges to between 0 and 1. Min Max Normalization is used to transform data based on Max and Min Values of feature column data.

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

Figure 7: Min Max Normalization Formula

In our data set the data is sparse with different ranges. When different columns are spread over different ranges then Machine Learning algorithms adds weights or get biased towards the features with highest values.so as to improve performance and ensure no weights and biasing we perform normalization in the range 0 to 1 to give equal weightage to all the features and scale normally.

2. Sampling the imbalance data

Training machine learning models on unbalanced data sets can pose unique challenges to learning problems. Data imbalance usually refers to a classification problem in which the number of observations in each category is unevenly distributed. Usually you

will have a lot of data / observation data class (referred to as the majority class), and much fewer observations for one or more other classes (referred to as the minority classes) [13].

2.1 Over sampling

Oversampling is the method of adding more of the minority class, so it has more effect on the machine learning algorithm

SMOTE

Smote is a nearest neighbor technique based on Euclidean distance between data points in feature space.
There is an oversampling percentage that indicates the number of synthetic samples to be created, and the oversampling percentage parameter is always a multiple of 100. If the oversampling percentage is 100, for each instance, a new sample will be created. As a result, the number of instances of a few classes will double. Similarly, if the oversampling percentage is 200, the total number of samples in the minority class will triple [15].

Figure 8: Over Sampling

2.2 Under sampling

Under sampling is the method of removing some of the majority class so it has less effect on the machine learning algorithm.

Neighbourhood Cleaning Rule

Editing the close distance removes samples from most categories that differ from one of the nearest neighbours. Sieves can be repeated, which is the principle of repeatedly editing adjacent distances. By changing the parameters of the internal nearest neighbour algorithm and increasing it at each iteration, all KNNs are slightly different from repeated edited nearest neighbours.
The compressed nearest neighbour uses 1-NN iteration to determine whether the sample should be kept in the data set. The problem is
that compressed nearest neighbours are sensitive to noise by retaining noise samples. One-sided selection also uses 1-NN and removes noise samples using Tomek links. The neighbour cleaning rule deletes some samples using the edited nearest neighbour distance. In addition, they use the 3 nearest neighbours to remove samples that are inconsistent with the rules [16].

Figure 9: Under Sampling

2.3 Balanced sampling

Balanced Sampling is defined as Hybrid sampling, which is the mixture of both oversampling and under sampling techniques.

SMOTE Tomek

Tomek's link and edited nearest neighbors are the two cleaning methods that have been added to the pipeline after applying SMOTE over-sampling to obtain a cleaner space. The ready-to use class imbalanced-learn implements for combining over- and under sampling methods is SMOTE Tomek.

Figure 10: Balanced Sampling

3. Feature Selection

Feature selection is one of the core concepts in machine learning, which greatly affects the performance of the model. The data capabilities we use to train your machine learning models have a huge impact on the performance you can achieve. Unrelated or partially related features may have a negative impact on model performance.
Feature selection is the process by which you automatically or manually select those features that contribute the most to the predictor or output you are interested in.
The best feature of the feature selection algorithm is to reduce overfitting, improve accuracy and reduce training time [17].

3.1 Variance Threshold Feature Selection

This method removes features with variation below a certain cut off. The idea is when a feature doesn't vary much within itself, it generally has very little predictive power. Variance Threshold doesn't consider the relationship of features with the target variable.
Motivated by the idea that low variance features contain less information. So, calculate variance of each feature, then drop features with variance below some threshold [18].

```
Decision Tree
[[347  67]
 [ 58 394]]
              precision    recall  f1-score   support

         0.0       0.86      0.84      0.85       414
         1.0       0.85      0.87      0.86       452

    accuracy                           0.86       866
   macro avg       0.86      0.85      0.86       866
weighted avg       0.86      0.86      0.86       866
```

Figure 11: Variance Threshold Feature selection for Decision Tree

```
K-NN Algo
[[294 120]
 [ 14 438]]
              precision    recall  f1-score   support

         0.0       0.95      0.71      0.81       414
         1.0       0.78      0.97      0.87       452

    accuracy                           0.85       866
   macro avg       0.87      0.84      0.84       866
weighted avg       0.87      0.85      0.84       866
```

Figure 12: Variance Threshold Feature selection for K-NN Tree

```
Naive Bayes
[[298 116]
 [134 318]]
              precision    recall  f1-score   support

         0.0       0.69      0.72      0.70       414
         1.0       0.73      0.70      0.72       452

    accuracy                           0.71       866
   macro avg       0.71      0.71      0.71       866
weighted avg       0.71      0.71      0.71       866
```

Figure 13: Variance Threshold Feature selection for Naive Bayes

```
Random Forests
[[257 157]
 [ 24 428]]
              precision    recall  f1-score   support

         0.0       0.91      0.62      0.74       414
         1.0       0.73      0.95      0.83       452

    accuracy                           0.79       866
   macro avg       0.82      0.78      0.78       866
weighted avg       0.82      0.79      0.78       866
```

Figure 14: Variance Threshold Feature selection for Random Forests

```
Support Vector Machine
[[305 109]
 [ 86 366]]
              precision    recall  f1-score   support

         0.0       0.78      0.74      0.76       414
         1.0       0.77      0.81      0.79       452

    accuracy                           0.77       866
   macro avg       0.78      0.77      0.77       866
weighted avg       0.78      0.77      0.77       866
```

Figure 15: Variance Threshold Feature selection for SVM

	decisiontree	naive	knn	random	svm
0	0.607390	0.371824	0.630485	0.630485	0.630485
1	0.571594	0.461894	0.598152	0.598152	0.598152
2	0.585450	0.612009	0.612009	0.612009	0.612009
3	0.681293	0.794457	0.847575	0.847575	0.847575
4	0.882217	0.859122	0.926097	0.926097	0.926097
5	0.903002	0.886836	0.916859	0.916859	0.916859
6	0.937644	0.845266	0.958430	0.958430	0.958430
7	0.956120	0.821016	0.961894	0.961894	0.961894
8	0.957275	0.855658	0.926097	0.926097	0.926097
9	0.583333	0.833333	0.884259	0.884259	0.884259

Figure 16: Table of accuracies for Variance Threshold Feature selection with 10-fold cross validation.

Algorithm evaluations for Variance Threshold

For Variance Threshold feature selection, Decision Tree algorithm was performing more accurate than others.

Ranking

1. Decision Tree
2. K-NN
3. Support Vector Machine
4. Random Forests
5. Naïve Bayes

3.2 Select K Best Feature Selection

It takes a scoring function as a parameter, which must be applied to a pair (X, y). The score function must return an array of scores, each of the features X of the score X [:, i] X [:, i] (Also, it can return a p-value, but it is neither required nor required). Then, SelectKBest retains only the top kk features of the highest-scoring X.

So, for example, if you pass chi2 as a scoring function, SelectKBest will calculate the chi2 statistic between each feature of X and y (assuming class labels). Smaller values indicate that the feature is independent of y. Larger values indicate that the feature is non-randomly related to y and are therefore likely to provide important information. Keep only k elements.

```
Decision Tree
[[331  83]
 [ 86 366]]
              precision    recall  f1-score   support

         0.0       0.79      0.80      0.80       414
         1.0       0.82      0.81      0.81       452

    accuracy                           0.80       866
   macro avg       0.80      0.80      0.80       866
weighted avg       0.80      0.80      0.80       866
```

Figure 17: Select K Best Feature selection for Decision Tree

```
K-NN Algo
[[294 120]
 [ 62 390]]
              precision    recall  f1-score   support

         0.0       0.83      0.71      0.76       414
         1.0       0.76      0.86      0.81       452

    accuracy                           0.79       866
   macro avg       0.80      0.79      0.79       866
weighted avg       0.79      0.79      0.79       866
```

Figure 18: Select K Best Feature selection for K-NN

```
Naive Bayes
[[285 129]
 [134 318]]
              precision    recall  f1-score   support

         0.0       0.68      0.69      0.68       414
         1.0       0.71      0.70      0.71       452

    accuracy                           0.70       866
   macro avg       0.70      0.70      0.70       866
weighted avg       0.70      0.70      0.70       866
```

Figure 19: Select K Best Feature selection for Naive Bayes

```
Random Forests
[[277 137]
 [ 71 381]]
              precision    recall  f1-score   support

         0.0       0.80      0.67      0.73       414
         1.0       0.74      0.84      0.79       452

    accuracy                           0.76       866
   macro avg       0.77      0.76      0.76       866
weighted avg       0.76      0.76      0.76       866
```

Figure 20: Select K Best Feature selection for Random Forests

```
Support Vector Machine
[[281 133]
 [104 348]]
              precision    recall  f1-score   support

         0.0       0.73      0.68      0.70       414
         1.0       0.72      0.77      0.75       452

    accuracy                           0.73       866
   macro avg       0.73      0.72      0.72       866
weighted avg       0.73      0.73      0.73       866
```

Figure 21: Select K Best Feature selection for SVM

	decisiontree	naive	knn	random	svm
0	0.560046	0.415704	0.557737	0.557737	0.557737
1	0.568129	0.467667	0.600462	0.600462	0.600462
2	0.543880	0.600462	0.527714	0.527714	0.527714
3	0.663972	0.750577	0.714781	0.714781	0.714781
4	0.766744	0.826790	0.764434	0.764434	0.764434
5	0.722864	0.854503	0.804850	0.804850	0.804850
6	0.863741	0.847575	0.862587	0.862587	0.862587
7	0.884527	0.847575	0.860277	0.860277	0.860277
8	0.887991	0.855658	0.836028	0.836028	0.836028
9	0.618056	0.826389	0.839120	0.839120	0.839120

Figure 22: Table of accuracies for Select K Best Feature selection

Algorithm evaluations for Select K Best feature selection

For Select K best feature selection, Ensemble using Decision Tree algorithm was performing more accurate than others.

Ranking

1. Decision Tree
2. K-NN
3. Random Forests
4. Support Vector Machine
5. Naïve Bayes

VI. Model Evaluation

1. Stage -1: Binary Classification of AMP

A. Decision Tree

Decision tree learning is to construct a decision tree from labeled training tuples. A decision tree is a structure similar to a flowchart, where each internal (non-leaf) node represents a test of an attribute, each branch represents the test result, and each leaf (or terminal) node has a class label. The highest node in the tree is the root node [21].

a) Raw

```
Decision Tree
[[355  74]
 [ 61 209]]
              precision    recall  f1-score   support

         0.0       0.85      0.83      0.84       429
         1.0       0.74      0.77      0.76       270

    accuracy                           0.81       699
   macro avg       0.80      0.80      0.80       699
weighted avg       0.81      0.81      0.81       699
```

Figure 23:Decision Tree accuracy with Raw Data

b) Over Sampled

```
Decision Tree
[[380  71]
 [ 61 363]]
              precision    recall  f1-score   support

         0.0       0.86      0.84      0.85       451
         1.0       0.84      0.86      0.85       424

    accuracy                           0.85       875
   macro avg       0.85      0.85      0.85       875
weighted avg       0.85      0.85      0.85       875
```

Figure 24:Decision Tree accuracy with Over Sampled Data

c) Under Sampled

```
Decision Tree
[[280  28]
 [ 18 244]]
              precision    recall  f1-score   support

         0.0       0.94      0.91      0.92       308
         1.0       0.90      0.93      0.91       262

    accuracy                           0.92       570
   macro avg       0.92      0.92      0.92       570
weighted avg       0.92      0.92      0.92       570
```

Figure 25: Decision Tree accuracy with Under Sampled Data

d) Balanced Sampled

```
Decision Tree
[[350  64]
 [ 56 396]]
              precision    recall  f1-score   support

         0.0       0.86      0.85      0.85       414
         1.0       0.86      0.88      0.87       452

    accuracy                           0.86       866
   macro avg       0.86      0.86      0.86       866
weighted avg       0.86      0.86      0.86       866
```

Figure 26: Decision Tree accuracy with Balanced Sampled Data

Sampling technique evaluations for Decision Tree

For Decision Tree algorithm, Under Sampling was performing more accurate than others.

Ranking based on different sampling technique

1. Under Sampling
2. Balanced Sampling
3. Over Sampling
4. Raw Data

B. Naïve Bayes

A Naive Bayes classifier is a probabilistic linear machine learning model that's used for classification task. The crux of the classifier is based on the Bayes theorem.

$$p(C_k \mid \mathbf{x}) = \frac{p(C_k)\, p(\mathbf{x} \mid C_k)}{p(\mathbf{x})}$$

Figure 27: Naive Formula

Using Bayes theorem, we can find the probability of A happening, given that B has occurred. Here, B is the evidence and A is the hypothesis. The assumption made here is that the predictors/features are independent. That is presence of one particular feature does not affect the other. Hence it is called naïve [20].

a) Raw

Naive Bayes
[[346 83]
[93 177]]

	precision	recall	f1-score	support
0.0	0.79	0.81	0.80	429
1.0	0.68	0.66	0.67	270
accuracy			0.75	699
macro avg	0.73	0.73	0.73	699
weighted avg	0.75	0.75	0.75	699

Figure 28: Naive Bayes accuracy with Raw Data

b) Over Sampling

Naive Bayes
[[342 109]
[115 309]]

	precision	recall	f1-score	support
0.0	0.75	0.76	0.75	451
1.0	0.74	0.73	0.73	424
accuracy			0.74	875
macro avg	0.74	0.74	0.74	875
weighted avg	0.74	0.74	0.74	875

Figure 29:Naive Bayes accuracy with Over Sampled Data

c) Under Sampled

Naive Bayes
[[267 41]
[59 203]]

	precision	recall	f1-score	support
0.0	0.82	0.87	0.84	308
1.0	0.83	0.77	0.80	262
accuracy			0.82	570
macro avg	0.83	0.82	0.82	570
weighted avg	0.82	0.82	0.82	570

Figure 30: Naive Bayes accuracy with Under Sampled Data

d) Balance Sampled

Naive Bayes
[[298 116]
[134 318]]

	precision	recall	f1-score	support
0.0	0.69	0.72	0.70	414
1.0	0.73	0.70	0.72	452
accuracy			0.71	866
macro avg	0.71	0.71	0.71	866
weighted avg	0.71	0.71	0.71	866

Figure 31: Naive Bayes accuracy with Balanced Sampled Data

Sampling technique evaluations for Decision Tree

For Naïve Bayes algorithm, Under Sampling was performing more accurate than others.

Ranking based on different sampling technique

1. Under Sampling
2. Over Sampling
3. Balanced Sampling
4. Raw Data

C. K-NN

In k-NN classification, the output is a class member. An object is classified by multiple votes of its neighbors, and the object is assigned to the most common category among its nearest k neighbors (k is a positive integer, usually small). If k = 1, simply assign the object to the class of that single nearest neighbor [22].

In *k-NN regression*, the output is the property value for the object. This value is the average of the values of *k* nearest neighbors.

a) Raw

K-NN Algo
[[344 85]
[56 214]]

	precision	recall	f1-score	support
0.0	0.86	0.80	0.83	429
1.0	0.72	0.79	0.75	270
accuracy			0.80	699
macro avg	0.79	0.80	0.79	699
weighted avg	0.80	0.80	0.80	699

Figure 32: K-NN accuracy with Raw Data

b) Over Sampled

K-NN Algo
[[323 128]
[22 402]]

	precision	recall	f1-score	support
0.0	0.94	0.72	0.81	451
1.0	0.76	0.95	0.84	424
accuracy			0.83	875
macro avg	0.85	0.83	0.83	875
weighted avg	0.85	0.83	0.83	875

Figure 33: K-NN accuracy with Over Sampled Data

c) Under Sampled

```
K-NN Algo
[[276  32]
 [ 11 251]]
              precision    recall  f1-score   supp

         0.0       0.96      0.90      0.93
         1.0       0.89      0.96      0.92

    accuracy                           0.92
   macro avg       0.92      0.93      0.92
weighted avg       0.93      0.92      0.92
```

Figure 34: K-NN accuracy with Under Sampled Data

d) Balance Sampled

```
K-NN Algo
[[294 120]
 [ 14 438]]
              precision    recall  f1-score   support

         0.0       0.95      0.71      0.81       414
         1.0       0.78      0.97      0.87       452

    accuracy                           0.85       866
   macro avg       0.87      0.84      0.84       866
weighted avg       0.87      0.85      0.84       866
```

Figure 35: K-NN accuracy with Balanced Data

Sampling technique evaluations for K-NN

For K-NN algorithm, Under Sampling was performing more accurate than others.

Ranking based on different sampling technique

1. Under Sampling
2. Balanced Sampling
3. Over Sampling
4. Raw Data

D. Random Forests

Random forest is a meta-estimator that fits many decision tree classifiers on various sub-samples of a dataset and uses averages to improve prediction accuracy and control overfitting. The subsample size is always the same as the original input sample size, but if bootstrap = True (the default), the sample is drawn with replacement [23].

a) Raw

```
Random Forests
[[335  94]
 [ 49 221]]
              precision    recall  f1-score   support

         0.0       0.87      0.78      0.82       429
         1.0       0.70      0.82      0.76       270

    accuracy                           0.80       699
   macro avg       0.79      0.80      0.79       699
weighted avg       0.81      0.80      0.80       699
```

Figure 36: Random Forests accuracy with Raw Data

b) Over Sampled

```
Random Forests
[[292 159]
 [ 18 406]]
              precision    recall  f1-score   support

         0.0       0.94      0.65      0.77       451
         1.0       0.72      0.96      0.82       424

    accuracy                           0.80       875
   macro avg       0.83      0.80      0.79       875
weighted avg       0.83      0.80      0.79       875
```

Figure 37: Random Forests accuracy with Over Sampled Data

c) Under Sampled

```
Random Forests
[[238  70]
 [ 11 251]]
              precision    recall  f1-score   support

         0.0       0.96      0.77      0.85       308
         1.0       0.78      0.96      0.86       262

    accuracy                           0.86       570
   macro avg       0.87      0.87      0.86       570
weighted avg       0.88      0.86      0.86       570
```

Figure 38: Random Forests accuracy with Under Sampled Data

d) Balance Sampled

```
Random Forests
[[257 157]
 [ 24 428]]
              precision    recall  f1-score   support

         0.0       0.91      0.62      0.74       414
         1.0       0.73      0.95      0.83       452

    accuracy                           0.79       866
   macro avg       0.82      0.78      0.78       866
weighted avg       0.82      0.79      0.78       866
```

Figure 39: Random Forests accuracy with Balanced Data

Sampling technique evaluations for Random Forests

For Random Forests algorithm, Under Sampling was performing more accurate than others.

Ranking based on different sampling technique

1. Under Sampling
2. Over Sampling
3. Balance Sampling
4. Raw Data

E. Support Vector Machine

Support vector machine (SVM) is a discriminative classifier formally defined by a separation hyperplane. In other words, given labeled training data (supervised learning), the algorithm will output the optimal hyperplane that classifies the new examples. In two-dimensional space, this hyperplane is a line that divides the plane into two parts, each part on each side [28].

a) Raw

Support Vector Machine
[[362 67]
 [87 183]]

	precision	recall	f1-score	support
0.0	0.81	0.84	0.82	429
1.0	0.73	0.68	0.70	270
accuracy			0.78	699
macro avg	0.77	0.76	0.76	699
weighted avg	0.78	0.78	0.78	699

Figure 40: Support Vector Machine accuracy with Raw Data

b) Over Sampled

Support Vector Machine
[[343 108]
 [59 365]]

	precision	recall	f1-score	support
0.0	0.85	0.76	0.80	451
1.0	0.77	0.86	0.81	424
accuracy			0.81	875
macro avg	0.81	0.81	0.81	875
weighted avg	0.81	0.81	0.81	875

Figure 41: Support Vector Machine accuracy with Over Sampled Data

c) Under Sampled

Support Vector Machine
[[268 40]
 [31 231]]

	precision	recall	f1-score	support
0.0	0.90	0.87	0.88	308
1.0	0.85	0.88	0.87	262
accuracy			0.88	570
macro avg	0.87	0.88	0.87	570
weighted avg	0.88	0.88	0.88	570

Figure 42: Support Vector Machine accuracy with Under Sampled Data

d) Balance Sampled

Support Vector Machine
[[305 109]
 [86 366]]

	precision	recall	f1-score	support
0.0	0.78	0.74	0.76	414
1.0	0.77	0.81	0.79	452
accuracy			0.77	866
macro avg	0.78	0.77	0.77	866
weighted avg	0.78	0.77	0.77	866

Figure 43: Support Vector Machine accuracy with Balanced Data

Sampling technique evaluations for SVM algorithm

For SVM algorithm, Under Sampling was performing more accurate than others.
Ranking based on different sampling technique

1. Under Sampling
2. Over Sampling
3. Balanced Sampling
4. Raw Data

2. Stage -2: Multi Label Classification of AMP Feature

A. Random Forests

	precision	recall	f1-score	support
0	0.00	0.00	0.00	6
1	0.00	0.00	0.00	4
2	0.00	0.00	0.00	5
3	0.00	0.00	0.00	13
4	0.64	0.21	0.31	121
5	0.00	0.00	0.00	1
6	0.00	0.00	0.00	5
7	0.78	1.00	0.88	204
8	0.00	0.00	0.00	8
9	0.00	0.00	0.00	10
10	0.00	0.00	0.00	28
11	0.00	0.00	0.00	39
12	0.00	0.00	0.00	34
13	0.00	0.00	0.00	37
14	0.00	0.00	0.00	9
15	1.00	0.22	0.36	9
micro avg	0.76	0.43	0.55	533
macro avg	0.15	0.09	0.10	533
weighted avg	0.46	0.43	0.41	533
samples avg	0.76	0.53	0.59	533

0.3015267175572519

Figure 44: Random Forests

Accuracy: 30.15%

B. Binary Relevance using Gaussian NB

	precision	recall	f1-score	support
0	0.14	0.83	0.24	6
1	0.06	0.50	0.11	4
2	0.04	0.60	0.07	5
3	0.13	0.92	0.23	13
4	0.49	0.68	0.57	121
5	1.00	1.00	1.00	1
6	0.11	1.00	0.20	5
7	0.84	0.82	0.83	204
8	0.10	0.62	0.18	8
9	0.10	1.00	0.19	10
10	0.17	0.89	0.29	28
11	0.19	0.38	0.25	39
12	0.21	0.71	0.32	34
13	0.22	0.38	0.28	37
14	0.08	0.78	0.15	9
15	0.17	0.56	0.26	9
micro avg	0.29	0.72	0.41	533
macro avg	0.25	0.73	0.32	533
weighted avg	0.50	0.72	0.55	533
samples avg	0.31	0.74	0.39	533

0.022900763358778626

Figure 45: Gaussian NB

Accuracy: 2.22%

C. Classification Chain Algorithm – Decision Tree

	precision	recall	f1-score	support
0	0.55	1.00	0.71	6
1	0.67	1.00	0.80	4
2	0.60	0.60	0.60	5
3	0.68	1.00	0.81	13
4	0.76	0.87	0.81	121
5	1.00	1.00	1.00	1
6	0.50	0.80	0.62	5
7	0.88	0.90	0.89	204
8	0.89	1.00	0.94	8
9	0.77	1.00	0.87	10
10	0.69	0.86	0.76	28
11	0.67	1.00	0.80	39
12	0.73	0.88	0.80	34
13	0.69	0.95	0.80	37
14	0.80	0.89	0.84	9
15	0.56	1.00	0.72	9
micro avg	0.77	0.91	0.83	533
macro avg	0.71	0.92	0.80	533
weighted avg	0.78	0.91	0.83	533
samples avg	0.72	0.83	0.75	533

0.6297709923664122

Figure 46: Classification Chain using Decision Tree

Accuracy: 62.9%

D. Classification Chain Algorithm – Ridge Classifier CV

	precision	recall	f1-score	support
0	1.00	0.17	0.29	6
1	0.00	0.00	0.00	4
2	0.00	0.00	0.00	5
3	0.00	0.00	0.00	13
4	0.54	0.53	0.54	121
5	0.00	0.00	0.00	1
6	0.00	0.00	0.00	5
7	0.78	1.00	0.88	204
8	1.00	0.12	0.22	8
9	0.00	0.00	0.00	10
10	0.00	0.00	0.00	28
11	0.50	0.03	0.05	39
12	0.00	0.00	0.00	34
13	1.00	0.05	0.10	37
14	0.00	0.00	0.00	9
15	1.00	0.44	0.62	9
micro avg	0.71	0.52	0.60	533
macro avg	0.36	0.15	0.17	533
weighted avg	0.57	0.52	0.48	533
samples avg	0.71	0.60	0.61	533

0.29770992366412213

Figure 47: Classification Chain using Ridge Classifier CV

Accuracy: 29.7%

VII. Evaluation Techniques

A. K-fold Cross Validation

Cross-validation is a resampling process used to evaluate machine learning models on limited data samples. The process has a single parameter called k, which represents the number of groups into which a given data sample is split. Therefore, this process is often called k-fold cross validation. When selecting a specific value for k, you can use a specific value for k in the model reference, for example, k = 10 becomes 10 times the cross-validation. Cross-validation is mainly used in machine learning applications to estimate the skills of machine learning models based on invisible data. That is, use limited samples to estimate how the model is typically expected to perform when used to make predictions on unused data during model training [24].

B. Train Test Split

The concept of test sequence segmentation is a technique used in machine learning to divide a data set into training and test data. This can be done easily by passing a panda data frame to the train_test_split () function or the scikit learning library. We will need to provide the proportion of the dataset that needs to be split. Based on experience or the most commonly used convention, we use 70:30 as the ratio of dividing the data set into training and test data sets. Similarly, we can try to build models for other ratios and observe performance. The test sequence split also provides an option to specify something called "random state", which is just an integer. This helps ensure that we don't get a different training and test dataset each time we split the dataset. The following is a count of split test and training data sets.

C. ROC Curve

The receiver operating characteristic curve or ROC curve is a graphical diagram that illustrates the diagnostic capabilities of the binary classifier system when the discrimination threshold changes.

ROC curves were drawn by plotting the true positive rate (TPR) and positive rate (FPR) at various threshold settings. The true positive rate is also called sensitivity, recall rate, or detection probability in machine learning. The false positive rate is also called false alarm probability and can be calculated as (1-specificity).

Raw

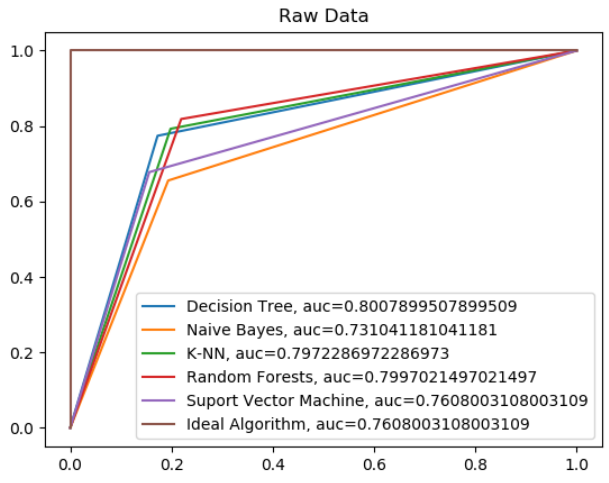

Figure 48: ROC for Raw Data

Over Sampled

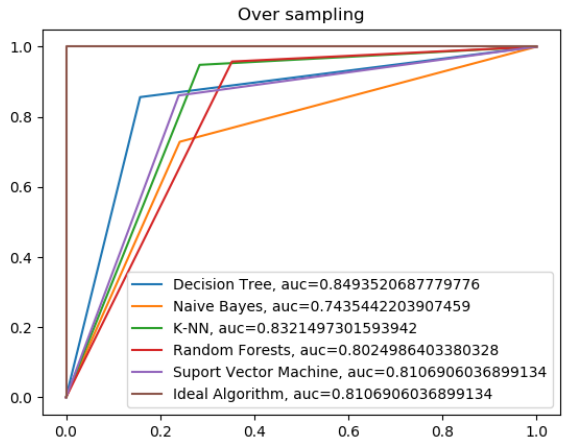

Figure 49: ROC for Over Sampled Data

Under Sampled

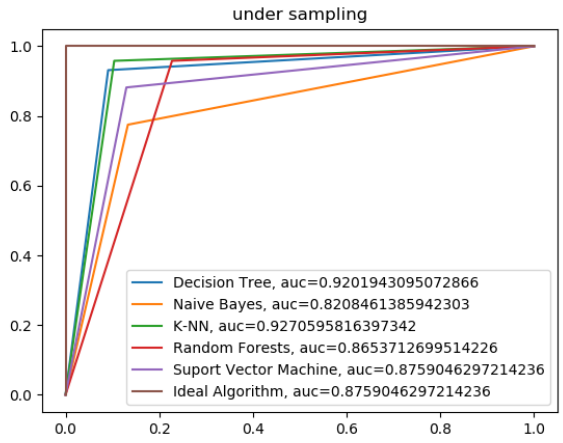

Figure 50: ROC for Under Sampled Data

Balance Sampled

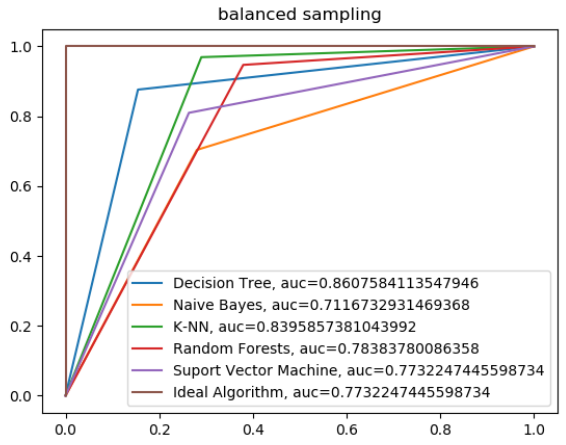

Figure 51: ROC for Balanced Sampled Data

D. Confusion Matrix

A table of confusion (sometimes also called a confusion matrix), is a table with two rows and two columns that reports the number of *false positives*, *false negatives*, *true positives*, and *true negatives[25]*.

		predicted	
actual		negative	positive
	negative	TN True positive	FP False Positive
	positive	FN False negative	TP True positive

Figure 52: Confusion Matrix

E. Accuracy

When we use the term accuracy, we usually mean classification accuracy. It is the ratio of the number of correct predictions to the total number of input samples. It works only if the number of samples belonging to each category is equal [27].

Accuracy:

$$AC = \frac{TN + TP}{TN + FP + FN + TP}$$

Figure 53: Accuracy

F. Precision

Precision (P) is defined as the number of true positives (TP) over the number of true positives plus the number of false positives (FP)[27].

Precision:

$$precision : \frac{TP}{FP + TP}$$

Figure 54: Precision

G. Recall

Recall (R) is defined as the number of true positives (TP) over the number of true positives plus the number of false negatives (FN).

Recall aka. True Positive Rate:

$$recall = \frac{TP}{FN + TP}$$

Figure 55: Recall

H. F1-score

F1-score is defined as the harmonic mean of precision and recall.

$$F_1 = \left(\frac{2}{\text{recall}^{-1} + \text{precision}^{-1}} \right) = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}.$$

Figure 56: F1-Score

I. Friedman Testing

The Friedman test is a non-parametric version of repeated measures analysis or repeated measures analysis of variance. This test can be seen as an extension of the Kruskal-Wallis H

test to more than two samples.
The default or null hypothesis is that multiple pairs of samples have the same distribution. A negative negating null hypothesis indicates that one or more of the paired samples have different distributions.
Cannot reject H0: Paired sample distributions are equal.
Reject H0: Paired sample distributions are not equal [31].

	decisiontree	naive	knn	random	svm
0	0.596491	0.385965	0.680702	0.680702	0.680702
1	0.600000	0.617544	0.770175	0.770175	0.770175
2	0.770175	0.856140	0.966667	0.966667	0.966667
3	0.942105	0.921053	0.968421	0.968421	0.968421
4	0.924561	0.864912	0.968421	0.968421	0.968421
5	0.922807	0.859649	0.970175	0.970175	0.970175
6	0.933333	0.835088	0.957895	0.957895	0.957895
7	0.957821	0.901582	0.963093	0.963093	0.963093
8	0.978873	0.885563	0.978873	0.978873	0.978873
9	0.801056	0.904930	0.948944	0.948944	0.948944

Figure 57: Table of Accuracies with 10 fold cross validation

```
stat=16.000, p=0.003019164
Probably the same distribution
```

Figure 58: Friedman Test Results

VIII. Results

In the stage -1, for binary classification Under sampling was giving more accuracy compared to other sampling techniques.

As the data set is dealing with many features, it's important to understand main features that gives more precision and accuracy. The project is tested with Variance Threshold and Select K Best features selection algorithms out of both Variance Threshold was giving more accuracy results.

The project is modelled with Decision Tree, Naïve Bayes, K-NN, Support Vector Machine and Random Forests Algorithms. Out of 5 algorithms Decision tree was giving more accuracy.

With comparison with the reference paper, the paper was giving accuracy with 90% with random forests algorithm but,
With Under Sampling Decision tree was giving 92% which is better than reference paper proposed accuracy.

In the stage-2, for Multi label classification, the project is tested with Random Forests, Gaussain NB, Decision Tree and Ridge Classifier CV, out of which Decision Tree was giving more accuracy.

With comparison with the reference, Decision tree was giving 62% accuracy whereas reference paper proposed 85% accuracy with Random Forests Algorithm.

IX. Future Work

The First stage was giving really good results, but in Second step its lacking due to high imbalance data. So, a good multi label sampling technique can be used for more accuracy.

X. References.

[1] Yuan Lin, Yinyin Cai, Juan Liu, Chen Lin, and Xiangrong Liu. An advanced approach to identify antimicrobial peptides and their function types for penaeus through machine learning strategies. BMC Bioinformatics, 20(Suppl 8):291, Jun 2019.

[2] https://scikit-learn.org/stable/modules/impute.html

[3] https://www.itl.nist.gov/div898/handbook/eda/section3/eda35b.htm#:~:targetText=Skewness%20is%20a%20measure%20of,relative%20to%20a%20normal%20distribution.

[4] sciencedirect.com/topics/medicine-and-dentistry/correlation-analysis

[5] https://en.wikipedia.org/wiki/Heat_map#:~:targetText=A%20heat%20map%20(or%20heatmap,existed%20for%20over%20a%20century.

[6] https://www.r-bloggers.com/how-to-use-data-analysis-for-machine-learning-example-part-1/

[7] https://hackernoon.com/what-steps-should-one-take-while-doing-data-preprocessing-502c993e1caa#:~:targetText=Data%20preprocessing%20is%20a%20data,method%20of%20resolving%20such%20issues.

[8] https://hackernoon.com/what-is-one-hot-encoding-why-and-when-do-you-have-to-use-it-e3c6186d008f#:~:targetText=One%20hot%20encoding%20is%20a,a%20better%20job%20in%20prediction.

[9] https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.OneHotEncoder.html

[10] https://medium.com/@urvashilluniya/why-data-normalization-is-necessary-for-machine-learning-models-681b65a05029#:~:targetText=Normalization%20is%20a%20technique%20often,dataset%20does%20not%20require%20normalization.

[11] https://www.statisticshowto.datasciencecentral.com/probability-and-statistics/z-score/#:~:targetText=A%20z%2Dscore%20is%20also,of%20the%20normal%20distribution%20curve).

[12] https://en.wikipedia.org/wiki/Dimensionality_reduction#:~:targetText=In%20statistics%2C%20machine%20learning%2C%20and,feature%20selection%20and%20feature%20extraction.

[13] https://www.jeremyjordan.me/imbalanced-data/

[14] http://www.chioka.in/class-imbalance-problem/

[15] https://medium.com/towards-artificial-intelligence/application-of-synthetic-minority-over-sampling-technique-smote-for-imbalanced-data-sets-509ab55cfdaf

[16] https://imbalanced-learn.readthedocs.io/en/stable/auto_examples/under-sampling/plot_comparison_under_sampling.html.

[17] https://towardsdatascience.com/why-how-and-when-to-apply-feature-selection-e9c69adfabf2

[18] https://chrisalbon.com/machine_learning/feature_selection/variance_thresholding_for_feature_selection/.

[19] https://en.wikipedia.org/wiki/Linear_classifier#:~:targetText=I

n%20the%20field%20of%20machine,linear%20combination%20of%20the%20characteristics.

[20] https://towardsdatascience.com/naive-bayes-classifier-81d512f50a7c

[21] https://en.wikipedia.org/wiki/Decision_tree_learning

[22] https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm

[23] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

[24] https://machinelearningmastery.com/k-fold-cross-validation/

[25] https://en.wikipedia.org/wiki/Confusion_matrix

[26] https://towardsdatascience.com/metrics-to-evaluate-your-machine-learning-algorithm-f10ba6e38234

[27] https://scikitlearn.org/stable/auto_examples/model_selection/plot_precision_recall.html

[28] https://medium.com/machine-learning-101/chapter-2-svm-support-vector-machine-theory-f0812effc72#:~:targetText=A%20Support%20Vector%20Machine%20(SVM,hyperplane%20which%20categorizes%20new%20examples.

[29] Cai CZ, Han LY, Ji ZL, Chen X, Chen YZ. SVM-Prot: web-based support vector machine software for functional classification of a protein from its primary sequence. Nucleic Acids Res. 2003. https://doi.org/10.1093/nar/gkg600.

[30] Li YH. SVM-Prot: SVM-Prot 2016: A web-server for machine learning prediction of protein functional families from sequence irrespective of similarity. PloS ONE. 2016. https://doi.org/10.1371/journal.pone.0155290.

[31] https://machinelearningmastery.com/nonparametric-statistical-significance-tests-in-python/