README OF ASSIGNMENT-2

PREDICTION OF ANTI FUNGAL PEPTIDES

TEAM NAME- haaye garmi

INPUT-:

We used **pandas** library to take input.
1)There is a function **read_csv()** in pandas library which helps to read the csv file and store it in a dataset.
2)We read **train.csv** and stored its content in train_data and then we read **test.csv** and stored its contents in test_data.
3)We also took the charge, pi value and mass of each sequence using peptides package of RStudio and stored those value in **final.cs**v for training data set and **final1.csv** for testing data set.
4)We then read **final.csv** and **final1.csv** using read_csv() function of pandas library and stored that data in mass_data and mass_data_t arrays respectively.
5)From train_data we retrieved its sequences by train_data[“Sequence”] column.In that column we iterated all sequences of the peptides one by one and then worked on them to get our desired output.We did the same for testing data set too.

PROCEDURE-:

WE USED THE FOLLOWING METHODS TO ACHIEVE OUR RESULT-:

1)BINARY PROFILE

We make a dictionary of all 20 amnio acids with the initial count set to 0.Now we iterate over all sequences of training data set and when processing each sequence we count the number of occurrence of each amino acid in that sequence.We maintain the count of every amnio acid for every sequence we process.We store the entire data in a 2-D array.We then do the same for testing data set and maintain it in a separate array.This is how you maintain binary profile or composition of the sequences of training and testing data set.We will use this method to find binary profile of sequences in the methods given below too.

2)DIPEPTIDE

Here we make a dictionary where the key is a dipeptide.We make a dictionary of all possible dipeptides formed from the 20 available amino acids.Initially we set the count for every dipeptide as 0.Then we iterate over every sequence of training data set and while processing every sequence we check the amino acid and the amino acid next to it, form a dipeptide for the same and check its occurrence.So when we go through the sequence we count the occurrence of every dipeptide which can be formed by adjacent amino acids of the sequence.We keep the dipeptide count for all sequences of training data in an array.Then we do the same for all sequences of testing data set and keep it in a separate array.

3)N5C5

If the length of the sequence is less than 10 than we take the entire sequence for binary profiling but if the length of the sequence is greater than 10 than we take just the first 5 and last 5 amino

acids of the peptide sequence together and take their binary profile into account.We do the same for all sequences of training and testing data set and store it in arr2 and arr2_t respectively.

4)N10C10

If the length of the sequence is less than 20 than we take the entire sequence for binary profiling but if the length of the sequence is greater than 20 than we take just the first 10 and last 10 amino acids of the peptide sequence together and take their binary profile into account.We do the same for all sequences of training and testing data set and store it in arr3 and arr3_t respectively.

5)N15C15

If the length of the sequence is less than 30 than we take the entire sequence for binary profiling but if the length of the sequence is greater than 30 than we take just the first 15 and last 15 amino acids of the peptide sequence together and take their binary profile into account.We do the same for all sequences of training and testing data set and store it in arr4 and arr4_t respectively.

6)N5

Here if the length of the sequence is greater than 5 then we take the 5 amino acids from N terminal into account and find binary profile for those 5 amino acids only.If the length of the sequence is less than 5 then we find the binary profile of the entire sequence.We do this for all sequences of training and testing data set and store the binary profiles obtained in arr5 and arr5_t respectively.

7)N10

Here if the length of the sequence is greater than 10 then we take the 10 amino acids from N terminal into account and find binary profile for those 10 amino acids only.If the length of the sequence is less than 10 then we find the binary profile of the entire sequence.We do this for all sequences of training and testing data set and store the binary profiles obtained in arr6 and arr6_t respectively.

8)N15

Here if the length of the sequence is greater than 15 then we take the 15 amino acids from N terminal into account and find binary profile for those 15 amino acids only.If the length of the sequence is less than 15 then we find the binary profile of the entire sequence.We do this for all sequences of training and testing data set and store the binary profiles obtained in arr7 and arr7_t respectively.

9)C5

Here if the length of the sequence is greater than 5 then we take the 5 amino acids from C terminal into account and find binary profile for those 5 amino acids only.If the length of the sequence is less than 5 then we find the binary profile of the entire sequence.We do this for all sequences of training and testing data set and store the binary profiles obtained in arr8 and arr8_t respectively.

9)C10

Here if the length of the sequence is greater than 10 then we take the 10 amino acids from C terminal into account and find binary profile for those 10 amino acids only.If the length of the sequence is less than 10 then we find the binary profile of the entire sequence.We do this for all sequences of training and testing data set and store the binary profiles obtained in arr9 and arr9_t respectively.

9)C15

Here if the length of the sequence is greater than 15 then we take the 15 amino acids from C terminal into account and find binary profile for those 15 amino acids only.If the length of the sequence is less than 15 then we find the binary profile of the entire sequence.We do this for all sequences of training and testing data set and store the binary profiles obtained in arr10 and arr10_t respectively.

Now we have made array for all features of training and testing data sets independently.

Now we make columns like b_profile, dipep, N5C5, N10C10, N15C15, N5, N10,N15,C5,C10,C15 in the training data set and store the corresponding arrays made above in it.
Then we make the same columns in testing data set and in those columns store the corresponding arrays made above.

Now we add all the arrays of training data set for all features in feature_df and all arrays of testing data set for all features in feature_dft.Now we use feature_df and feature_dft to find the desired output.

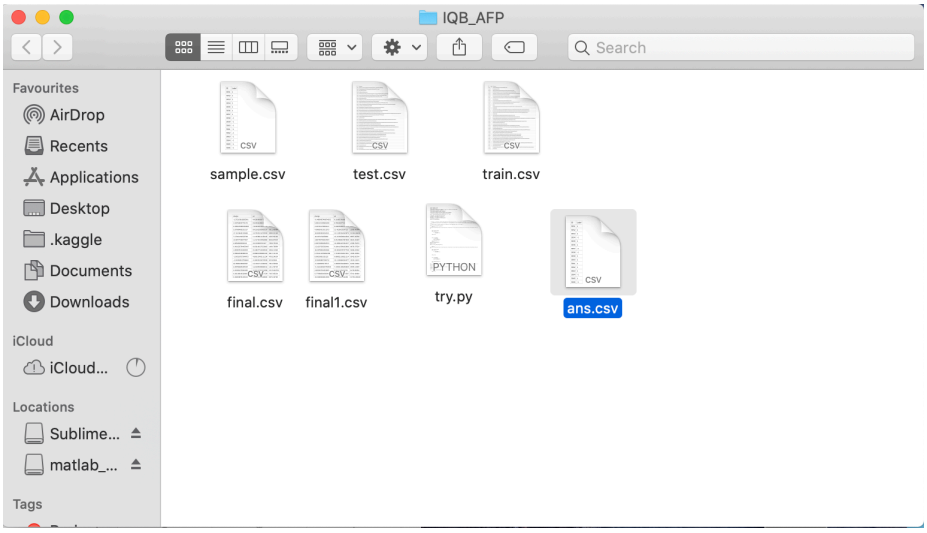
<u>OUTPUT</u>-:

Our code on implementation prints prediction of labels of all sequences for the testing data set on terminal.

```
SHREEYAs-MacBook-Pro-11421:IQB_AFP shreeyagarg$ python3 try.py
/Library/Frameworks/Python.framework/Versions/3.7/lib/python3.7/site-packages/sklearn/svm/base.py:929: ConvergenceWarning: Liblinear failed to converge, increase the number of iterations.
  "the number of iterations.", ConvergenceWarning)
/Library/Frameworks/Python.framework/Versions/3.7/lib/python3.7/site-packages/sklearn/svm/base.py:193: FutureWarning: The default value of gamma will change from 'auto' to 'scale' in version 0.22 to accou
nt better for unscaled features. Set gamma explicitly to 'auto' or 'scale' to avoid this warning.
  "avoid this warning.", FutureWarning)
Predicting labels for testing data set by a trained model
['1' '1' '1' '-1' '1' '1' '-1' '-1' '1' '1' '1' '1' '1' '1' '1' '-1' '-1'
 '1' '-1' '-1' '1' '1' '1' '1' '1' '1' '1' '-1' '-1' '1' '1' '1' '-1' '1' '1'
 '-1' '-1' '1' '-1' '-1' '1' '1' '1' '1' '-1' '-1' '-1' '-1' '-1' '1' '-1'
 '-1' '-1' '-1' '1' '1' '-1' '1' '-1' '-1' '-1' '1' '-1' '-1' '1' '-1' '1' '1'
 '-1' '1' '1' '-1' '1' '1' '1' '1' '-1' '-1' '-1' '1' '-1' '-1' '1' '-1'
 '-1' '-1' '-1' '-1' '1' '1' '1' '-1' '-1' '1' '-1' '-1' '1' '1' '1' '-1'
 '1' '-1' '1' '1' '1' '-1' '-1' '1' '1' '1' '1' '1' '-1' '-1' '1' '-1' '-1'
 '1' '-1' '-1' '-1' '-1' '-1' '1' '1' '-1' '-1' '-1' '-1' '-1' '-1' '-1' '-1'
 '-1' '1' '1' '1' '1' '1' '1' '-1' '-1' '1' '-1' '1' '1' '1' '-1' '1' '-1'
 '-1' '-1' '-1' '1' '1' '1' '-1' '1' '-1' '1' '1' '-1' '-1' '1' '1' '-1'
 '1' '-1' '1' '1' '1' '-1' '-1' '1' '1' '-1' '1' '-1' '-1' '-1' '1' '-1'
 '-1' '1' '-1' '1' '-1' '1' '-1' '1' '-1' '1' '-1' '1' '1' '1' '-1' '-1'
 '-1' '-1' '1' '1' '1' '1' '-1' '-1' '1' '1' '-1' '-1' '1' '1' '-1' '-1' '1'
 '-1' '-1' '-1' '-1' '1' '-1' '-1' '1' '-1' '1' '-1' '1' '-1' '1' '1' '-1'
 '-1' '-1' '-1' '1' '1' '1' '1' '-1' '1' '-1' '1' '1' '1' '1' '-1' '1'
 '-1' '-1' '-1' '1' '1' '1' '-1' '1' '1' '-1' '-1' '1' '1' '1' '1' '1' '1'
 '1' '-1' '-1' '-1' '1' '-1' '-1' '1' '1' '1' '-1' '1' '-1' '-1' '-1' '-1'
 '1' '-1' '-1' '1' '1' '1' '-1' '-1' '-1' '1' '1' '-1' '-1' '1' '-1' '-1' '1'
 '-1' '-1' '1' '-1' '-1' '1' '-1' '1' '-1' '1' '1' '-1' '-1' '-1' '1' '-1'
 '-1' '1' '1' '1' '-1' '-1' '-1' '-1' '1' '1' '1' '1' '-1' '1' '-1' '1'
 '-1' '-1' '1' '-1' '-1' '1' '1' '1' '1' '1' '-1' '-1' '1' '-1' '-1' '1' '1'
 '-1' '1' '-1' '1' '1' '-1' '1' '1' '-1' '-1' '1' '1' '1' '1' '1' '-1' '1'
 '1' '-1' '1' '1' '1' '-1' '1' '1' '-1' '1' '-1' '-1' '1']
Sub success
SHREEYAs-MacBook-Pro-11421:IQB_AFP shreeyagarg$
```

You will observe that Sub-Success is printed when the entire execution is completed.

Our code also generates a **ans.csv** file which predicts the label for testing data set.It displays the ID and the corresponding label of all the sequences of testing data set in a tabular form.A table named “Table 1” is generated to display the final data in ans.csv.



This is how **ans.csv** looks when opened.

Table 1

ID	Label
3551	1
3552	1
3553	1
3554	-1
3555	1
3556	1
3557	-1
3558	-1
3559	1
3560	1
3561	1
3562	1
3563	1
3564	1
3565	1
3566	-1
3567	-1
3568	1
3569	-1
3570	-1
3571	1
3572	1
3573	1
3574	1

HOW DID WE ACHIEVE THIS OUTPUT-:

SUPPORT VECTOR MACHINE (SVM)

We made feature_df as a sum of some columns of train_data.The columns of train_data which we included in feature_df are b_profile, dipep, N5C5, N10C10, N15C15, N5, N10, N15, C5, C10, C15. Similarly we added the same category columns of test_data in feature_dft.We used linear SVC model to train our data.We used feature_df for training our model and feature_dft for testing our model.We trained our data using train_data["Label"] to make the model understand how to give label for peptides sequences.We used **feature_selection** and **classification** primarily from SVM Model to train our dataset.When the model is trained we finally passed x_test to predict the results.This predicted result is shown on terminal as well as a csv file named **ans.csv** is also formed to display the labels of all sequences given in testing data set.In the output you will observe that if the label is 1 than the sequence may be an anti fungal peptide and if the sequence has label -1 than that sequence is not a anti fungal peptide.

NAME AND ROLL NOs OF THE TEAM-:

SHREEYA GARG(2018415)
YASHDEEP PRASAD(2018121)
SEJAL SINGH(2018413)