**Model working**

**File structure**

App.py

Source.py

Data.csv

Clf.sav

---------

Static

{ Multiple files }

Template

Inex.html

----------

**Programming language used**

* Python
* Libraries used in python code

|  |  |
| --- | --- |
| **Python libraries** | **versions** |
| **Flask** | **1.1.2** |
| **pandas** | **1.2.0** |
| **Numpy** | **1.19.3** |
| **matplotlib** | **3.4.0** |
| **Seaborn** | **0.11.1** |
| **tensorflow** | **2.5.0** |
| **Scikit learn** | **0.24.2** |

**Scripting language used**

* HTML, CSS, Js

**Flow of code**

Input protein sequence

0

1

1

0

0

1

1

1

0

0

1

1

1

0

0

0

A

A

M

M

B

B

K

K

K

L

M

S

M

M

L

L

Endolysins

Machine learning

model

Python

code

Training

Machine

Antolysins

**1.Dataset**

In dataset we have collected protein data structure sequences. The dataset consists of 21 endolysins and 21 antolysins out of 42 protein data structure. This data is collected from uniport universal data bank. The sequence of protein code is called by FASTA code

AAAABBBBBDDDDMMMMMMLLLLLAAAALLLJJFGHH

FASTA code

The protein dataset taken according to the authors “Ding, Hui & Luo, Liaofu & Lin, Hao” mentioned in their research paper prediction of cell wall Lytic Enzymes.

**2.Python code**

Source.py

In this source code we are taken the protein dataset which is in the form of comma separated values (csv) and we did quick preprocessing the dataset and removed unnecessary data columns.

We then converted the protein sequence data to machine code that is in the form of 0 and 1.

**3.Machine learning model**

To train the model we need x and y values here x is the input dataset (protein sequence) and y is the output dataset (endolysins/ antolysins).

For training and testing we need to split the dataset suppose in a 100% of dataset we can train the machine by 80% dataset and use the 20% dataset to test the model.

Here in this code we have used scikit-learn machine learning library to split the dataset and then we have given the training data x\_train and y\_train values to the support vector machine(svm). After the model is built, we can check the model predictions using y\_test values

**Support vector machine(svc)**

Search and Take the data from the internet

svc will take the inputs ( x\_train , y\_train ) and train the machine learning model. We now predict the protein predictions over cmd.

**4.app.py**

this app.py code is used to integrate the machine learning model to webpage so the we can take the input from webpage and convert the protein sequence to machine code and show the results on the same page.

**Input**

**Protein sequence**

**Output**

**E/A**

**source.py**

**machine**

**learning**

**model**

**app.py**

**converting text**

**to machine language**

**webpage**

**5.Frontend**

We used web Technology such as HTML,CSS,JS to design the frontend part of the website

{ you can write much info about html, css, js }

**steps to run the application**:

1. Click **app.py** file

CMD file will open and starts the server on your local machine wait for few seconds the url text will generate there you need to copy the URL generated by your machine.

1. Past the copied URL on your local machine browser
2. The website will load and you can make predictions.

{Backend the CMD file be running if you close the CMD, server will end the connections and you cant make predictions.}