COMP 4970/7970 Project 3: 15 points 15% Credit
Final Submission due before 11:59 PM Tuesday March 27

Instructions:

1. This is a group project. You should do your own work while working collaboratively as a group. Any evidence of copying either from a public source or from the works of other groups without due credits will result in a zero grade and additional penalties/actions against **all members of the involved groups.**
2. Select one member from your team as a group leader for the project who did not serve as group leader before during the course. The group leader will be responsible for the task allocation, progress tracking, combining works from the team members, maintaining all communications with the instructor, and for certifying efforts of the team members. **The group leader will also deliver the project presentation in class and upload the final submission to Canvas on behalf of the entire group.**
3. **No show in project presentation or final submissions by email or late submissions (even by minutes) will receive a zero grade for the entire group.** No makeup will be offered unless prior permission has been granted, or there is a valid and verifiable excuse.

Project presentation (5 points):

1. All group presentations scheduled **in class on Tuesday March 27**.
2. The group leader to deliver a PowerPoint presentation for 13 minutes followed by additional 2 minutes of Q&A.
3. Effort citification (5-point scale) for each member of the team must be presented at the end of the presentation.

Final Submission (10 points):

1. Python source files containing your code only (no test data needed).
2. Completed report document (Word or PDF format) and Readme.txt file (template provided).
3. Group leader uploads all files to Canvas as a single zipped folder before the deadline on the due date.

Implementing Gaussian Naïve Bayes for protein SS prediction

Objective: Implement Gaussian Naïve Bayes for protein 3 class secondary structure prediction.

You must use standard Python programming language. You are NOT allowed to use existing packages or libraries (e.g. Biopython).

A: Raw Data:

Two data files (*Proteins.fa* and *Proteins.sa*) are supplied. The *Proteins.fa* file contains 5,772 protein sequences in Multi-FASTA format. A multi-FASTA file contains multiple FASTA formatted sequences.

```
>sequenceID-001 description
AAGTAGGAATAATATCTTATCATTATAGATAAAAACCTTCTGAATTTGCTTAGTGTGTAT
ACGACTAGACATATATCAGCTCGCCGATTATTTGGATTATTCCCTG
>sequenceID-002 description
CAGTAAAGAGTGGATGTAAGAACCGTCCGATCTACCAGATGTGATAGAGGTTGCCAGTAC
AAAAATTGCATAATAATTGATTAATCCTTTAATATTGTTTAGAATATATCCGTCAGATAA
TCCTAAAAATAACGATATGATGGCGGAAATCGTC
>sequenceID-003 description
CTTCAATTACCCTGCTGACGCGAGATACCTTATGCATCGAAGGTAAAGCGATGAATTTAT
CCAAGGTTTTAATTTG
```

The true 3-class secondary structure (SS) labels of these proteins can be found in the *Proteins.ss* file. This file is also in Multi-FASTA format. SS labels have three possible values:

```
'H': helix
'E': strand
'C': coil
```

N.B. The true SS labels are calculated using the DSSP (Dictionary of Protein Secondary Structure: Pattern Recognition of Hydrogen-Bonded and Geometrical Features. Kabsch and Sander, 1983) software and the resulting 8-class assignment has been transformed into 3-class.

B: Curating Training and Test Datasets:

Divide the raw data into non-overlapping sets of training (~75%) and test (~25%) datasets using simple random sampling without replacement.

C. Position Specific Scoring Matrix (PSSM) generation:

Step 1: Download NCBI PSIBLAST to `<your_blast_path>` as follows:

```
$ cd <your_blast_path>
$ wget https://zhanglab.ccmb.med.umich.edu/PSSpred/blastv2.6.tar.gz
$ tar xvfz blastv2.6.tar.gz
```

The executable programs can be found at `<your_blast_path>/blast/bin/`

Step 2: Download NR database to `<your_nr_path>` as follows:

```
$ cd  <your_nr_path>
$ wget http://calla.rnet.missouri.edu/qprob/nr_database.tar.gz
$ tar xvfz nr_database.tar.gz
```

The nr database can be found at `<your_nr_path>/nr_database`

Step 3: Test PSSM generation for a test protein sequence located at `<your_test_path>/test.fa`:

```
$cd <your_test_path>
$ <your_blast_path>/blast/bin/blastpgp -d <your_nr_path>/nr_database/nr -j 3 -b
1 -a 4 -i test.fa -Q test.pssm
```

The output PSSM file can be located at `<your_test_path>/test.pssm`

N.B. Sample test.fa and test.pssm files are included in the project.

D. Feature Generation:

Use the 20 PSSM values for each residue in a protein from the .pssm file generated by PSIBLAST. Fist few lines for the test.pssm file are given below. For a protein sequence of N residues, there will be N x 20 PSSM values.

```
           A  R  N  D  C  Q  E  G  H  I  L  K  M  F  P  S  T  W  Y  V
    1 E   -3 -2 -2  3 -6  4  5 -4 -3 -6 -5  3 -4 -6 -4 -3 -3 -5 -5 -5
    2 A    1 -2 -2  2 -4  1  1  0 -4 -3 -1 -2 -2 -5  2  3  1 -2 -5 -1
    3 E   -1  4 -2  1 -5  1  0 -3  0 -2 -3  3 -2 -4  3  0  0 -5 -4 -1
    4 A    0  0 -2  0 -5  2  1 -1 -2 -4 -5  0 -3 -6  6 -1 -1 -6 -5 -4
    5 S    1 -2  1  5 -5 -1  3 -1 -1 -5 -5  0 -3 -6  0  1 -1 -6 -5 -4
    6 I   -1 -2 -5 -5 -4 -4 -3 -5  0  3  0 -3 -1  5 -2 -4 -1 -2  1  3
    7 C   -2 -6 -5 -6 11 -6 -6 -5 -6 -4 -4 -6 -4 -5 -6 -3 -3 -5 -5 -3
    8 S   -1  0  1 -1 -3  1  0 -3  3 -2  2  0  0  0 -5  1  0 -1  1 -2
    9 E   -1 -3 -5 -3 -4  3  2 -5 -3 -2  5 -3  2 -1 -3 -3 -3 -2 -3 -1
   10 P   -2 -3 -4 -3 -5 -1  0 -5 -3 -5 -4 -2 -4 -6  8 -1 -3 -6 -5 -4
   11 K    2  0 -3 -3 -5  0 -1 -4 -1 -2 -2  4  1 -5  4  0 -3 -5 -4  0
   12 K   -1 -2 -1  5 -5 -2  3 -4 -2 -2 -2  2 -2 -4 -4 -2 -3 -5  0  2
   13 V   -1  0 -3 -3 -3  0  0 -3 -4 -1 -3  0 -2 -4  3  1  3 -5 -3  3
   14 G   -2 -4 -3 -4 -5 -4 -4  7 -5 -7 -6 -4 -5 -6 -4 -3 -4 -5 -6 -6
   15 R   -3  2 -2 -3 -5 -3 -2 -4  1 -1 -2 -1 -2 -3  7 -2 -1 -3 -2 -2
```

Additionally, use a sliding window of 5 around the central residue (i.e. 2 residues on both sides) for feature generation and use the central residue's true SS type (from the *Proteins.ss* file) for class label assignment (H/E/C). Therefore, there will be 20 x 5 = 100 PSSM values for each non-terminal residue. For terminal residues that do not have one or more neighbors on either side, use -1 as dummy PSSM values. E.g., for the first residue at the N-terminal, the feature vector will start with forty (20 x 2) dummy PSSM values of -1. Similarly, there will be twenty dummy PSSM values of -1 at the beginning of the feature vector for the second residue. There will be symmetrically opposite effect at the C-terminal with feature vectors ending with forty dummy PSSM values of -1 for the last residue and twenty dummy PSSM values of -1 for the second last residue.

E. Gaussian Naïve Bayes Learning on Training Set:

Implement the Gaussian Naïve Bayes learning algorithm that learns class priors and class conditional means and variances, assuming each P $(X_i \mid Y = y_k)$ to follow Gaussian distribution.

F. Gaussian Naïve Bayes Classification on Test Set:

Implement Gaussian Naïve Bayes classifier that takes the parameters learned during training to calculate the probability of each class and predict the class labels on any given test dataset by selecting the most probable class assignment (or a single sequence).

N.B. Gaussian Naïve Bayes is an offline-learning algorithm. Therefore, training and classification should be implemented separately. The classification algorithm should take a test dataset in Multi-FASTA format as an input and predict the class labels (i.e. H, E, C) in Multi-FASTA format (just like *Proteins.ss* file) in a standalone mode. You may save the parameters learned during training in a file that can be fed into the classifier, in an offline mode.

G. Evaluate Accuracy:

Use Q3 accuracy to evaluate the classification performance on the test dataset.