COMP 5970/6970 Project 3: 15 points 15% Credit
Final Submission due before 11:59 PM Monday March 25

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. **No show in project presentation or final submissions by email or late submissions (even by minutes) will receive a zero grade.** No makeup will be offered unless prior permission has been granted, or there is a valid and verifiable excuse.

Submission:

For 5970, one member from each group will upload the following to canvas before 11:59 PM Monday March 25:

1. **Source Code (Member 1)**: Python source files (upload .zip file in case of multiple files) containing your code only (no test data needed) and ReadMe.txt file (template provided) describing how to run your code. Note that we will NOT debug your code. If your code does not execute as described in ReadMe.txt, you will receive a zero grade.
2. **Presentation Slide (Member 2)**: One slide only in PPT/PPTX/PDF format to be used during the oral presentations (see below). If you submitted file span more than a page, we will extract the first page for the oral presentation.
3. **Project Report (Member 3)**: Completed report document in PDF format using template provided. Make sure to have all necessary sections of scientific writing: abstract, introduction, methods, results, discussion, references.

For 6970, one member from each group will upload the following to canvas before 11:59 PM Monday March 25:

1. **Source Code and Project Report (Member 1)**: (i) Python source files (upload .zip file in case of multiple files) containing your code only (no test data needed) and ReadMe.txt file (template provided) describing how to run your code. Note that we will NOT debug your code. If your code does not execute after following your instructions laid out in ReadMe.txt, you will receive a zero grade. (ii) Completed report document in PDF format using template provided. Make sure to have all necessary sections of scientific writing: abstract, introduction, methods, results, discussion, references.
2. **Presentation Slide and Video Demo (Member 2)**: (i) One slide only in PPT/PPTX/PDF format to be used during the oral presentations (see below). If you submitted file span more than a page, we will extract the first page for the oral presentation. (ii) A video demonstration not more than 5 minutes in duration containing a creative demonstration of the working dynamics of your program and the results achieved. Creative ways of visualization and use of graphic tools are encouraged. Please use widely recognized formats for videos.

Presentations:

Presentation will be during the class on **Wednesday March 27** and **Friday March 29**.

For 5970, the member submitting presentation slide will deliver 5 minutes flash presentation accompanied by the submitted slide:

1. At the least, your presentation should contain methods (i.e. implementation), results (e.g. output), and conclusion.
2. Practice your talk not to exceed the time limit or finish too early.
3. No need to bring your slides. We will set things up and decide the presentation sequence.

For 6970, the member submitting presentation slide and video demo will deliver 5 minutes flash presentation accompanied by the submitted slide followed by additional 5 minutes of demo accompanied by the submitted video:

1. At the least, your presentation should contain methods (i.e. implementation), results (e.g. output), and conclusion.
2. Practice your talk not to exceed the time limit or finish too early.
3. The video demo may be accompanied by oral presentation.
4. No need to bring your slides/demo. We will set things up and decide the presentation sequence.

Implementing Gaussian Naïve Bayes for protein Secondary Structure prediction

Objective: Implement decision tree for protein relative solvent accessibility prediction.

Note: You must use standard Python programming language. You are NOT allowed to use non-standard packages or libraries (e.g. Biopython, scikit-learn, SciPy, NumPy, etc.).

A: Raw Data:

Two directories (*fasta* and *sa*) are supplied. The *fasta* directory contains 150 protein sequences in FASTA format. A FASTA file is as follows:

```
>sequenceID
AAGTAGGAATAATATCTTATCATTATAGATAAAAACCTTCTGAATTTGCTTAGTGTGTATACGACTAGACATATATCAG
CTCGCCGATTATTTGGATTATTCCCTG
```

The true 3-class secondary structure (SS) labels of these proteins can be found in the *ss* directory. Files in this directory are also in 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) Files:

In addition to the *fasta* and *sa* directories, a *pssm* directory is also provided containing the evolutionary profile of the 150 proteins (one *pssm* file for each *fasta* sequence). Every row in a *pssm* file corresponds to the evolutionary profile of one amino acid. *pssm* files are generated using NCBI PSIBLAST software.

D. Feature Generation:

Use the values in the first 20 PSSM columns for each amino acid in a protein from the *pssm* file generated by PSIBLAST. Fist few lines for a sample *pssm* file are given below. For a protein sequence of N amino acids, 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 corresponding *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 file in FASTA format as an input (passed as a command line argument to your program) and predict the class labels (i.e. H, E, C) in FASTA format (just like *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.