GitHub Diary – Project in Molecular Life Science

15 February (Start-Introduction)

Introduction to the project. Read all the documentation regarding Bash, Git and "How to organize your project" to become familiar with the tools to use and the structure of the project.

I wrote a bash script that creates all the different folders that are needed to store the information I will be collecting during the development of the project. For a first try, I created a big directory (*MTLSProject*) inside which I created five other repositories (*Projects, Datasets, Results, Scripts, Bin*). These can be modified later on, when a clearer structure of the project is perceived and more files need to be stored inside the different repositories.
Then, this was uploaded on GitHub inside a repository that was previously created. The link of this repository was sent to the TAs so they can check everything has been done correctly.

Bash script:

```
mkdir MTLSProject
cd MTLSProject

mkdir Projects
touch Projects/readme.txt
touch Projects/commands.txt

mkdir Datasets
#touch Datasets/namefile    #touch creates a file inside a directory

mkdir Results
mkdir Scripts
mkdir bin
```

16 February (Lab group meeting)

I attended the lab group meeting at SciLife and listened to the presentations of the group members, who explained what they were exactly working on, as well as the most interesting and relevant scientific findings of their last week.

19 February (Project help + software carpentry submission)

Deadline for software carpentry tasks. The link to GitHub was already sent to TAs last week.

20 February (Project introduction)

Start of project. Mandatory parts to pass the project:

- *Extract the feature from your dataset*
- *Create cross-validated sets*
- *Train a SVM using single sequence information, using sklearn*
- *Check different window sizes for the inputs*
- *Analyze the results and compare it to previous work*
- *Review the state of art for your predictor*
- *Write a report*

The first step is to look at my dataset and understand what is consists of. It has 42 proteins, each of them with a given ID, its amino acid sequence and the corresponding topology. First of all, I have to read my dataset line by line and create a dictionary to store all the information it contains. The keys will be the IDs and the values will be one list. This list contains two elements: the amino acid sequence and the topology.

21 February (Project help)

Since the inputs from SVM must be numbers, it is necessary to convert the protein sequences and topology to a different format. This will then give us the output from which we obtain the predicted topology of the input sequence. In order to do this, the amino acid sequences are encoded into pure binary: every letter is represented by a vector of 20 elements, in which 19 of them are 0 and the other one is a 1. Depending on the position of the 1 in the vector, it will represent one of the 20 amino acids. For example:

["A"]=[1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0], ["C"]=[0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0] and so on.

The topologies also need to be encoded as numbers. In my dataset there are 3 possible topologies: i, M and o. In the new format: i = 0; M = 1; o = 2.
Once both the amino acid sequences and the topologies are transformed into numbers, they are stored in the original dictionary, so that now it has the keys with the sequence ID, and the values with a list containing the amino acid sequences and the topologies in binary format.

23 February (Project plan + 5 most relevant papers + Journal club + Lab group meeting)

I submitted a list of the 5 most relevant papers regarding β-barrel prediction that I found online. One of them will be selected for my individual presentation at the end of the course. Also, I also sent in one-page project plan describing how the realization of the project will look like and what steps will be followed to achieve the topology prediction of an input amino acid sequence.

I attended the Journal Club at SciLife for a discussion about deep learning and the consequent explanation of the paper "*Visualizing and Understanding Convolutional Networks*" I had read beforehand. Then we had a lab group meeting where we listened to the new findings of the group members.

26 February (Project help)

The next step in the project is to build the sliding windows. This is done to take into consideration the amino acids that surround a central one, and consider the topology of such amino acid accounting for the influence of the surrounding ones. For the sliding windows of the first and last amino acids we have to add flanking vectors to the left and the right of the binary amino acid sequence, respectively; this flanking vector contains 20 zeros and the number of times it should be added depends on the size of the sliding window.

28 February (Project help)

In order to train the SVM, we need to input the sliding windows of the amino acid sequences and the corresponding topologies for each sliding window, which will be determined by the central amino acid of each window.

Once the model is trained, it is time to predict the topology of an input protein sequence. For this aim, the file testfile.txt needs to be read and its sequences converted into binary. Following the same process as in the main script, these input sequences undergo the sliding window process to give rise to the testsetaa, which will be the input to predict with clf.predict(). The topology is predicted and shown in the command window.

It is still necessary to create all the modules that will make the script look more compact and clear, but this will be done next week!

1 March (Paper presentation with colleagues)

We organized groups of 5 students and went over the each of the presentations. Then I wrote a short evaluation of each of the presentations and submitted it.

2 March (Journal Club + Lab group meeting)

I attended the Journal Club at SciLife for a background presentation about deep transfer and how to make use of datasets when there is not enough data available. Then we listened to a presentation about face recognition using deep learning based on whales. Then we had a lab group meeting where we listened to the new findings of the group members.

5 March (Project help)

The implementations I performed today are the following:

- Convert the predicted topology back to letters, so that it does not return a vector of 0, 1 and 2 but the topology each of these numbers represents.
- Use "OrderedDict" when creating the dictionaries so that the values that are stored in them keep the same order that is given, and not a random order. In this way, we can directly relate the predicted topology with the input sequences, since they will be output in the same order as the sequences were input.
- Build all the functions (modules) and design the main script accordingly.
- Let the program run overnight on the whole dataset to figure out the best sliding window. The kernel used is linear and C=1. These parameters will be checked later when a good sliding window is found.

6 March

After approximately 20h, the sliding windows that have been computed vary in a range from 3 to 31, both included, taking only the odd numbers. The code was stop due to lack of time, and the best accuracy (**0.63731 (+/+ 0.06770)**) was obtained with the last sliding window computed, 31. The accuracy tends to increase as the sliding window is increased, but due to a lack of time, we will keep 31 as a good sliding window (although waiting for accuracy of bigger sliding windows could provide better results).

FINDING A GOOD SLIDING WINDOW

[0.47090663 0.4705617   0.47326867 0.46717798 0.45487365]
The predictor accuracy with a sliding window of 3 is: 0.46736 (+/+ 0.01307)
[0.52232747 0.51815926 0.50214302 0.49785698 0.47518051]
The predictor accuracy with a sliding window of 5 is: 0.50313 (+/+ 0.03352)
[0.56134416 0.57297541 0.52988947 0.53282202 0.504287]
The predictor accuracy with a sliding window of 7 is: 0.54026 (+/+ 0.04876)
[0.59111412 0.59147304 0.54139409 0.549515   0.53091155]
The predictor accuracy with a sliding window of 9 is: 0.56088 (+/+ 0.05105)
[0.60532251 0.60568464 0.56056846 0.5659824   0.54467509]
The predictor accuracy with a sliding window of 11 is: 0.57645 (+/+ 0.04948)

[0.62449256 0.60951951 0.57207309 0.57929168 0.55505415]
The predictor accuracy with a sliding window of 13 is: 0.58809 (+/+ 0.05067)
[0.62652233 0.63072411 0.58380329 0.57951726 0.55843863]
The predictor accuracy with a sliding window of 15 is: 0.59580 (+/+ 0.05635)
[0.63351376 0.63839386 0.59621024 0.59327769 0.55888989]
The predictor accuracy with a sliding window of 17 is: 0.60406 (+/+ 0.05841)
[0.6488498   0.64628919 0.60365441 0.60275209 0.56723827]
The predictor accuracy with a sliding window of 19 is: 0.61376 (+/+ 0.06117)
[0.65809653 0.648545   0.60658696 0.61222648 0.57468412]
The predictor accuracy with a sliding window of 21 is: 0.62003 (+/+ 0.06040)
[0.66869644 0.64922175 0.60884277 0.6167381   0.57694043]
The predictor accuracy with a sliding window of 23 is: 0.62409 (+/+ 0.06407)
[0.6736581   0.64899617 0.61109858 0.63049853 0.57536101]
The predictor accuracy with a sliding window of 25 is: 0.62792 (+/+ 0.06686)
[0.67298151 0.65192872 0.61335439 0.63613806 0.57490975]
The predictor accuracy with a sliding window of 27 is: 0.62986 (+/+ 0.06740)
[0.67456022 0.65847056 0.61312881 0.63546131 0.58145307]
The predictor accuracy with a sliding window of 29 is: 0.63261 (+/+ 0.06593)
[0.68335589 0.66117753 0.6167381   0.63907061 0.58619134]
The predictor accuracy with a sliding window of 31 is: 0.63731 (+/+ 0.06770)

Once a good sliding window is identified, it is time to change the kernel, C and gamma, and run the script with only the good sliding window. When good parameters are found, I will calculate confusion matrix. This matrix allows to evaluate the accuracy of the classification our script performs. In this matrix, the elements that are in the diagonal represent the amount of points for which the predicted label is equal to the true label. The rest of the elements in the matrix represent those points for which these predicted and true labels do not match. Therefore, higher values in the diagonal will indicate that the classifier was able to make more correct predictions.

Finally, I fixed the sliding window = 31, C=1 and a default gamma value, and computed the accuracy for the different kernels. I let the program run overnight on the entire dataset.

7 March (Project help)

After 24 hours, none of the accuracies for the different kernels have been shown. Therefore, I have predicted the accuracy of the model when running it against 15 (out of 42) representative sequence from my dataset, taking both positive and negative examples to get a good estimate of the accuracy that I would get on the entire dataset if I kept waiting. In this case, the results are:

FINDING THE BEST KERNEL TYPE:

[0.64129401 0.63333333 0.69113441 0.62154433 0.61641221]
The predictor accuracy with a linear kernel is: 0.64074 (+/+ 0.05333)
[0.53092293 0.53809524 0.57959962 0.53670162 0.57824427]

The predictor accuracy with a rbf kernel is: 0.55271 (+/+ 0.04308)
[0.40057088 0.40095238 0.4013346   0.4013346   0.40076336]
The predictor accuracy with a polynomial kernel is: 0.40099 (+/+ 0.00061)
[0.40057088 0.40666667 0.4013346   0.4013346   0.40076336]
The predictor accuracy with a sigmoid kernel is: 0.40213 (+/+ 0.00457)

The highest accuracy is obtained with the linear kernel, so up to now we have fixed: the sliding window = 31, kernel = linear and gamma is set as default since it is only necessary to modify it for rbf, sigmoid and poly kernels.

The next step is to get a good value for C. Trying different values, these are the results I got when sliding window = 31 and kernel = linear:

FINDING A GOOD VALUE FOR C

[0.66127498 0.65238095 0.68541468 0.6491897   0.67080153]
The predictor accuracy with C=0.01 is: 0.66381 (+/+ 0.02633)
[0.67269267 0.67333333 0.69685415 0.64442326 0.65267176]
The predictor accuracy with C=0.1 is: 0.66800 (+/+ 0.03659)
[0.63939106 0.64761905 0.69590086 0.64346997 0.625   ]
The predictor accuracy with C=0.3 is: 0.65028 (+/+ 0.04810)
[0.63558516 0.64857143 0.68541468 0.64823642 0.61736641]
The predictor accuracy with C=0.5 is: 0.64703 (+/+ 0.04462)
[0.64129401 0.63333333 0.69113441 0.62154433 0.61641221]
The predictor accuracy with C=1 is: 0.64074 (+/+ 0.05333)
[0.63177926 0.6352381   0.68446139 0.61773117 0.61832061]
The predictor accuracy with C=1.5 is: 0.63751 (+/+ 0.04901)
[0.63273073 0.63333333 0.67874166 0.61963775 0.61832061]
The predictor accuracy with C=2 is: 0.63655 (+/+ 0.04403)
[0.64129401 0.63333333 0.69113441 0.62154433 0.61641221]

SUMMARY OF SVM PARAMETERS

Thus, the selected parameters are:

- Sliding window size = 31
- Kernel = Linear
- C = 0.1

Once all these parameters are defined, it is time to train the model and save it as "*finalizedmodel.pkl*", so that it is only necessary to load it when predicting a protein's topology and everything works much faster:

- Train on the whole dataset: "*finalizedmodel.pkl*

  The model was successfully saved!
  [0.68380695 0.65982405 0.60861719 0.63816828 0.58212996]
  The predictor accuracy with a sliding window of 31 is: ***0.63451*** *(+/+ 0.07210).*
  The size of this file is too big (77MB). I will try training the model with parts of the dataset:

- Train on 21 sequences from the dataset: (finalizedmodel2.pkl)
  The model was successfully saved!
  [0.7129981 0.61301045 0.65622032 0.69563153 0.55175689]
  The predictor accuracy with a sliding window of 31 is: **0.64592** (+/+ 0.11671)
  Size of the file is still too big!

- Train on 16 sequences from the dataset: (finalizedmodel3.pkl)
  The model was successfully saved!
  [0.6883378 0.7156271 0.620389 0.62885906 0.68322148]
  The predictor accuracy with a sliding window of 31 is: **0.66729** (+/+ 0.07327)

  The size of this file is good enough to upload it to Github.

8 March (Presentations)

Today we listened to every colleague's presentation of the paper we got assigned in regard to the predictor we need to create for our project; each of us presented our paper for around 12 minutes and answered the questions we got from the rest. Then I wrote the self-evaluation that we have to send to Arne.

9 March (Journal Club + Lab group meeting)

I attended the Journal Club at SciLife for a background presentation about predicting the subcellular localization of a protein when only the sequence information is available. It is focused on a paper called "*DeepLoc: prediction of protein subcellular localization using deep learning*". Then we had a lab group meeting where we listened to the new findings of the group members.

Then I submitted the tweaked version of the predictor for this week. This time I have 4 important scripts:

- One script for training the model: *training.py*
- One script for the modules of the training script: *trainingmodules.py*
- One script for predicting a topology: *prediction.py*
- One script for the modules of the prediction script: *predictionmodules.py*

The parameters for SVM training are:

- Sliding window size = 31
- Kernel = Linear
- C = 0.1

The trained model is saved as "*finalizedmodel3.pkl*". This model has been trained with 16 sequences out of the 42 sequences of the whole dataset, since the size of the model trained on the whole dataset was too big for uploading to GitHub (77MB).

You can test the topologies contained in "*testfile.txt*" with the script "*prediction.py*".

12 March (Project help)

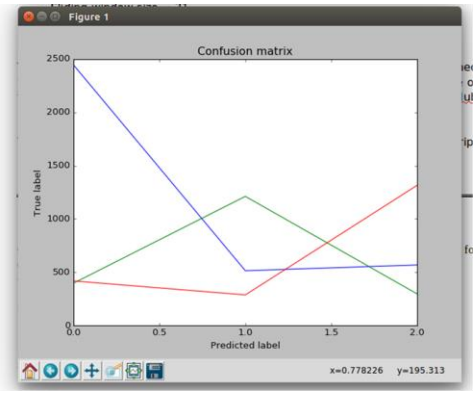
The first thing today was to run the psi blast bash script (using Uniref90) to get all the psiblast and pssm files (4h running). Then I created a script for PSSM files conversion for training of a model. Train the model and save as PSSMmodel.pkl.

With regard to the 3 linear models I have, the accuracy is not everything that matters!

CONFUSION MATRIX LINEAR MODEL:

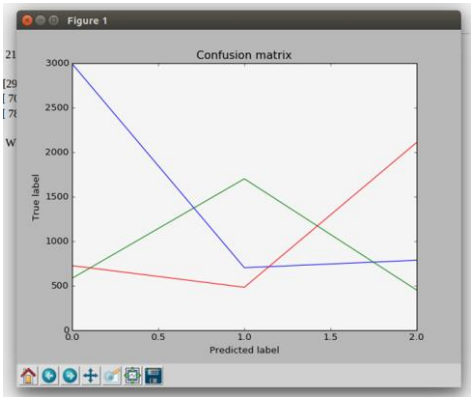
- Linear model trained with 16 sequences:

[[2446 397 418]
[514 1212 287]
[568 296 1316]]



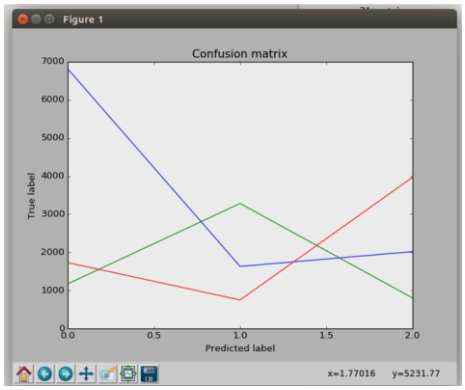
- Linear model trained with 21 sequences:

[[2993 583 725]
 [702 1701 482]
 [786 451 2109]]



- Linear model trained with the whole dataset:

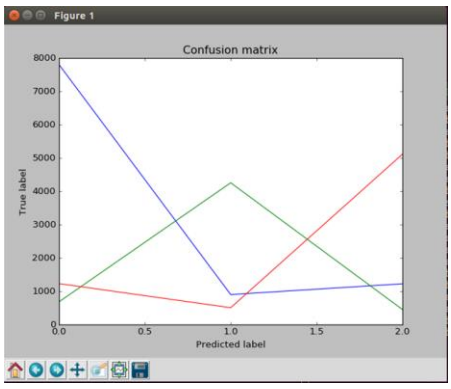
[[6814 1172 1729]
 [1630 3282 749]
[2018 803 3968]]



Since the elements in the diagonal are much bigger in the case of the whole dataset, this is the model I will keep using.

<u>CONFUSION MATRIX MODEL PSSM</u>

[[7805 681 1229]
[900 4255 506]
 [1224 451 5114]]



Running to save the PSSMmodel…

[0.77852954 0.808933 0.77847958 0.76404241 0.74413357]
The PSSM predictor accuracy with a sliding window of 31 is: 0,77482 (+/+ 0.04241)

The model is saved as PSSMmodel.pkl

13 March:

RANDOM FOREST:

FINDING GOOD PARAMETERS FOR RANDOM FOREST: Try different values for n_estimators and min_samples_split. Since it runs very fast, it is possible to try a lot of different combinations and keep the one with the best accuracy:

Scores for n_estimators = 10 and min_samples_split = 2 is: [0.66035183 0.6638845 0.69591699 0.67990074 0.65094765]
The predictor accuracy is: 0.67020 (+/+ 0.03179)
Scores for n_estimators = 10 and min_samples_split = 3 is: [0.6646369 0.67245658 0.6686217 0.67042635 0.62883574]
The predictor accuracy is: 0.66100 (+/+ 0.03257)
Scores for n_estimators = 10 and min_samples_split = 4 is: [0.65223275 0.68035191 0.68734491 0.67944958 0.63289711]
The predictor accuracy is: 0.66646 (+/+ 0.04125)
Scores for n_estimators = 10 and min_samples_split = 5 is: [0.64569238 0.67223099 0.68305888 0.67065193 0.63447653]
The predictor accuracy is: 0.66122 (+/+ 0.03626)
Scores for n_estimators = 10 and min_samples_split = 6 is: [0.65268381 0.67674261 0.68215655 0.6934356 0.64214801]
The predictor accuracy is: 0.66943 (+/+ 0.03812)
Scores for n_estimators = 10 and min_samples_split = 7 is: [0.65403699 0.67877284 0.69659373 0.68418678 0.63944043]
The predictor accuracy is: 0.67061 (+/+ 0.04169)
Scores for n_estimators = 10 and min_samples_split = 8 is: [0.66599008 0.67990074 0.69208211 0.69366118 0.64214801]
The predictor accuracy is: 0.67476 (+/+ 0.03820)
Scores for n_estimators = 10 and min_samples_split = 9 is: [0.66441137 0.6850891 0.69636815 0.68824724 0.65500903]
The predictor accuracy is: 0.67782 (+/+ 0.03106)

Scores for n_estimators = 50 and min_samples_split = 2 is: [0.7232747 0.7723889 0.76855403 0.73764945 0.7017148]
The predictor accuracy is: 0.74072 (+/+ 0.05376)
Scores for n_estimators = 50 and min_samples_split = 3 is: [0.73387461 0.77690052 0.76584706 0.74802617 0.69855596]
The predictor accuracy is: 0.74464 (+/+ 0.05471)
Scores for n_estimators = 50 and min_samples_split = 4 is: [0.73026613 0.77644936 0.76607264 0.74306339 0.71502708]
The predictor accuracy is: 0.74618 (+/+ 0.04513)
Scores for n_estimators = 50 and min_samples_split = 5 is: [0.73951286 0.78479585 0.77035867 0.75479359 0.70983755]
The predictor accuracy is: 0.75186 (+/+ 0.05180)
Scores for n_estimators = 50 and min_samples_split = 6 is: [0.72553 0.77171216 0.77396797 0.75366569 0.70645307]
The predictor accuracy is: 0.74627 (+/+ 0.05280)
Scores for n_estimators = 50 and min_samples_split = 7 is: [0.73319802 0.77396797 0.78005865 0.75411685 0.71231949]
The predictor accuracy is: 0.75073 (+/+ 0.05055)
Scores for n_estimators = 50 and min_samples_split = 8 is: [0.73184484 0.76945635 0.77193774 0.75344011 0.70916065]
The predictor accuracy is: 0.74717 (+/+ 0.04757)
Scores for n_estimators = 50 and min_samples_split = 9 is: [0.73545332 0.77780284 0.77080984 0.7464471 0.71615523]
The predictor accuracy is: 0.74933 (+/+ 0.04537)
Scores for n_estimators = 90 and min_samples_split = 2 is: [0.74718088 0.78727724 0.78637492 0.75750056 0.7265343]
The predictor accuracy is: 0.76097 (+/+ 0.04669)
Scores for n_estimators = 90 and min_samples_split = 3 is: [0.73951286 0.78344236 0.7818633 0.75817731 0.71750903]
The predictor accuracy is: 0.75610 (+/+ 0.05044)
Scores for n_estimators = 90 and min_samples_split = 4 is: [0.74312134 0.79178886 0.78637492 0.75298895 0.71886282]
The predictor accuracy is: 0.75863 (+/+ 0.05458)
Scores for n_estimators = 90 and min_samples_split = 5 is: [0.74379793 0.78863072 0.7829912 0.75885405 0.72360108]
The predictor accuracy is: 0.75957 (+/+ 0.04847)
Scores for n_estimators = 90 and min_samples_split = 6 is: [0.74515111 0.78817956 0.78817956 0.76088428 0.72066787]
The predictor accuracy is: 0.76061 (+/+ 0.05180)

Scores for n_estimators = 90 and min_samples_split = 7 is: [0.7413171 0.79088653 0.79539815 0.75930521 0.72608303]
The predictor accuracy is: 0.76260 (+/+ 0.05421)
Scores for n_estimators = 90 and min_samples_split = 8 is: [0.75304465 0.79224002 0.78975863 0.76426799 0.72563177]
The predictor accuracy is: 0.76499 (+/+ 0.04938)
Scores for n_estimators = 90 and min_samples_split = 9 is: [0.74921065 0.78276562 0.78411911 0.75501917 0.72788809]
The predictor accuracy is: 0.75980 (+/+ 0.04264)
Scores for n_estimators = 130 and min_samples_split = 2 is: [0.74267028 0.79449583 0.79517257 0.76133544 0.74052347]
The predictor accuracy is: 0.76684 (+/+ 0.04796)
Scores for n_estimators = 130 and min_samples_split = 3 is: [0.74785747 0.79291676 0.79291676 0.75907963 0.72608303]
The predictor accuracy is: 0.76377 (+/+ 0.05211)
Scores for n_estimators = 130 and min_samples_split = 4 is: [0.74560217 0.79787954 0.79562373 0.77013309 0.72450361]
The predictor accuracy is: 0.76675 (+/+ 0.05689)
Scores for n_estimators = 130 and min_samples_split = 5 is: [0.7541723 0.79923303 0.78795398 0.75727498 0.73352888]
The predictor accuracy is: 0.76643 (+/+ 0.04780)
Scores for n_estimators = 130 and min_samples_split = 6 is: [0.75372124 0.79201444 0.78998421 0.7617866 0.73262635]
The predictor accuracy is: 0.76603 (+/+ 0.04503)
Scores for n_estimators = 130 and min_samples_split = 7 is: [0.76071267 0.79968419 0.79314234 0.75885405 0.73127256]
The predictor accuracy is: 0.76873 (+/+ 0.04995)
Scores for n_estimators = 130 and min_samples_split = 8 is: [0.74853406 0.79427025 0.79787954 0.76539589 0.7301444]
The predictor accuracy is: 0.76724 (+/+ 0.05214)
Scores for n_estimators = 130 and min_samples_split = 9 is: [0.74988724 0.79178886 0.78998421 0.76268892 0.73691336]
The predictor accuracy is: 0.76625 (+/+ 0.04342)
Scores for n_estimators = 170 and min_samples_split = 2 is: [0.75349571 0.79968419 0.79607489 0.76674938 0.72721119]
The predictor accuracy is: 0.76864 (+/+ 0.05415)
Scores for n_estimators = 170 and min_samples_split = 3 is: [0.74424899 0.80126325 0.79742838 0.77013309 0.73894404]
The predictor accuracy is: 0.77040 (+/+ 0.05182)

Scores for n_estimators = 170 and min_samples_split = 4 is: [0.75597654 0.79201444 0.79201444 0.76539589 0.72879061]
The predictor accuracy is: 0.76684 (+/+ 0.04763)
Scores for n_estimators = 170 and min_samples_split = 5 is: [0.75372124 0.80509813 0.80148883 0.76742612 0.73533394]
The predictor accuracy is: 0.77261 (+/+ 0.05413)
Scores for n_estimators = 170 and min_samples_split = 6 is: [0.75011276 0.80103767 0.80148883 0.76832845 0.73104693]
The predictor accuracy is: 0.77040 (+/+ 0.05564)
Scores for n_estimators = 170 and min_samples_split = 7 is: [0.75304465 0.80081209 0.79336792 0.76381683 0.74323105]
The predictor accuracy is: 0.77085 (+/+ 0.04503)
Scores for n_estimators = 170 and min_samples_split = 8 is: [0.75507442 0.80036093 0.80036093 0.76742612 0.73307762]
The predictor accuracy is: 0.77126 (+/+ 0.05237)
Scores for n_estimators = 170 and min_samples_split = 9 is: [0.75304465 0.79336792 0.79043537 0.76517031 0.74187726]
The predictor accuracy is: 0.76878 (+/+ 0.04057)
Scores for n_estimators = 210 and min_samples_split = 2 is: [0.75214253 0.80509813 0.79404467 0.76855403 0.7407491]
The predictor accuracy is: 0.77212 (+/+ 0.04870)
Scores for n_estimators = 210 and min_samples_split = 3 is: [0.7564276   0.80284232 0.79449583 0.76607264 0.73240072]
The predictor accuracy is: 0.77045 (+/+ 0.05131)
Scores for n_estimators = 210 and min_samples_split = 4 is: [0.75349571 0.79787954 0.79607489 0.76968193 0.73646209]
The predictor accuracy is: 0.77072 (+/+ 0.04776)
Scores for n_estimators = 210 and min_samples_split = 5 is: [0.7586829   0.80013535 0.79630047 0.7606587   0.74435921]
The predictor accuracy is: 0.77203 (+/+ 0.04429)
Scores for n_estimators = 210 and min_samples_split = 6 is: [0.75552548 0.80397022 0.79427025 0.76426799 0.73646209]
The predictor accuracy is: 0.77090 (+/+ 0.04985)
Scores for n_estimators = 210 and min_samples_split = 7 is: [0.75349571 0.80464697 0.79968419 0.76855403 0.73691336]
The predictor accuracy is: 0.77266 (+/+ 0.05227)
Scores for n_estimators = 210 and min_samples_split = 8 is: [0.75845737 0.80690277 0.79855628 0.76584706 0.73601083]
The predictor accuracy is: 0.77315 (+/+ 0.05241)

Scores for n_estimators = 210 and min_samples_split = 9 is: [0.75078935 0.81209113 0.79652605 0.76517031 0.73962094]
The predictor accuracy is: 0.77284 (+/+ 0.05477)
Scores for n_estimators = 250 and min_samples_split = 2 is: [0.75913396 0.80735394 0.79923303 0.76449357 0.73916968]
The predictor accuracy is: 0.77388 (+/+ 0.05118)
Scores for n_estimators = 250 and min_samples_split = 3 is: [0.75552548 0.79787954 0.79675164 0.76359125 0.73307762]
The predictor accuracy is: 0.76937 (+/+ 0.04984)
Scores for n_estimators = 250 and min_samples_split = 4 is: [0.7564276 0.80013535 0.79517257 0.76742612 0.74052347]
The predictor accuracy is: 0.77194 (+/+ 0.04546)
Scores for n_estimators = 250 and min_samples_split = 5 is: [0.7564276 0.80261674 0.80013535 0.76471915 0.73759025]
The predictor accuracy is: 0.77230 (+/+ 0.05066)
Scores for n_estimators = 250 and min_samples_split = 6 is: [0.75732972 0.80487255 0.80600045 0.76607264 0.73962094]
The predictor accuracy is: 0.77478 (+/+ 0.05289)
Scores for n_estimators = 250 and min_samples_split = 7 is: [0.75484889 0.80058651 0.79878186 0.77103542 0.73420578]
The predictor accuracy is: 0.77189 (+/+ 0.05105)
Scores for n_estimators = 250 and min_samples_split = 8 is: [0.75462336 0.80509813 0.79923303 0.76877961 0.73623646]
The predictor accuracy is: 0.77279 (+/+ 0.05235)
Scores for n_estimators = 250 and min_samples_split = 9 is: [0.75439783 0.79945861 0.79742838 0.76517031 0.73916968]
The predictor accuracy is: 0.77112 (+/+ 0.04759)
Scores for n_estimators = 290 and min_samples_split = 2 is: [0.75327018 0.8041958 0.7972028 0.7676517 0.73759025]
The predictor accuracy is: 0.77198 (+/+ 0.05080)
Scores for n_estimators = 290 and min_samples_split = 3 is: [0.75349571 0.80148883 0.80216558 0.76720054 0.73871841]
The predictor accuracy is: 0.77261 (+/+ 0.05100)
Scores for n_estimators = 290 and min_samples_split = 4 is: [0.75484889 0.79945861 0.80171441 0.77171216 0.73240072]
The predictor accuracy is: 0.77203 (+/+ 0.05291)
Scores for n_estimators = 290 and min_samples_split = 5 is: [0.75597654 0.80464697 0.80058651 0.76720054 0.74052347]
The predictor accuracy is: 0.77379 (+/+ 0.05010)

Scores for n_estimators = 290 and min_samples_split = 6 is: [0.75439783 0.81006091 0.7972028 0.76990751 0.74413357]
The predictor accuracy is: 0.77514 (+/+ 0.05000)
Scores for n_estimators = 290 and min_samples_split = 7 is: [0.75687866 0.80577487 0.80126325 0.76607264 0.73713899]
The predictor accuracy is: 0.77343 (+/+ 0.05266)
Scores for n_estimators = 290 and min_samples_split = 8 is: [0.75462336 0.79878186 0.79675164 0.76720054 0.74187726]
The predictor accuracy is: 0.77185 (+/+ 0.04527)
Scores for n_estimators = 290 and min_samples_split = 9 is: [0.75462336 0.80329348 0.79742838 0.76697496 0.73601083]
The predictor accuracy is: 0.77167 (+/+ 0.05097)
Scores for n_estimators = 330 and min_samples_split = 2 is: [0.75958502 0.80938416 0.7983307 0.76110986 0.74706679]
The predictor accuracy is: 0.77510 (+/+ 0.04848)
Scores for n_estimators = 330 and min_samples_split = 3 is: [0.75845737 0.80960975 0.80261674 0.76968193 0.7450361]
The predictor accuracy is: 0.77708 (+/+ 0.05011)
Scores for n_estimators = 330 and min_samples_split = 4 is: [0.75800631 0.80148883 0.79787954 0.76697496 0.74435921]
The predictor accuracy is: 0.77374 (+/+ 0.04480)
Scores for n_estimators = 330 and min_samples_split = 5 is: [0.7586829 0.81051207 0.79742838 0.76720054 0.74232852]
The predictor accuracy is: 0.77523 (+/+ 0.05027)
Scores for n_estimators = 330 and min_samples_split = 6 is: [0.76138926 0.8041958 0.80216558 0.76945635 0.74029783]
The predictor accuracy is: 0.77550 (+/+ 0.04907)
Scores for n_estimators = 330 and min_samples_split = 7 is: [0.75981055 0.80148883 0.80013535 0.76990751 0.7407491]
The predictor accuracy is: 0.77442 (+/+ 0.04700)
Scores for n_estimators = 330 and min_samples_split = 8 is: [0.75507442 0.81096323 0.80194 0.76426799 0.74323105]
The predictor accuracy is: 0.77510 (+/+ 0.05322)
Scores for n_estimators = 330 and min_samples_split = 9 is: [0.75439783 0.79855628 0.79607489 0.7665238 0.74187726]
The predictor accuracy is: 0.77149 (+/+ 0.04500)
Scores for n_estimators = 370 and min_samples_split = 2 is: [0.75687866 0.80577487 0.80622603 0.77284006 0.74120036]
The predictor accuracy is: 0.77658 (+/+ 0.05204)

Scores for n_estimators = 370 and min_samples_split = 3 is: [0.7564276 0.808933 0.7983307 0.771261 0.74097473]
The predictor accuracy is: 0.77519 (+/+ 0.05069)
Scores for n_estimators = 370 and min_samples_split = 4 is: [0.75169147 0.80825626 0.80442139 0.76720054 0.73736462]
The predictor accuracy is: 0.77379 (+/+ 0.05646)
Scores for n_estimators = 370 and min_samples_split = 5 is: [0.76026161 0.80442139 0.7983307 0.76990751 0.74819495]
The predictor accuracy is: 0.77622 (+/+ 0.04349)
Scores for n_estimators = 370 and min_samples_split = 6 is: [0.75823184 0.80757952 0.80058651 0.77306564 0.74368231]
The predictor accuracy is: 0.77663 (+/+ 0.04873)
Scores for n_estimators = 370 and min_samples_split = 7 is: [0.76003608 0.81006091 0.80036093 0.76855403 0.7450361]
The predictor accuracy is: 0.77681 (+/+ 0.04915)
Scores for n_estimators = 370 and min_samples_split = 8 is: [0.76003608 0.80577487 0.80081209 0.76629822 0.74165162]
The predictor accuracy is: 0.77491 (+/+ 0.04919)
Scores for n_estimators = 370 and min_samples_split = 9 is: [0.75823184 0.80554929 0.80103767 0.77035867 0.74165162]
The predictor accuracy is: 0.77537 (+/+ 0.04920)

Select the best parameters among those calculated, compute the confusion matrix for the model with these specific parameters and save the model as rfmodel.pkl. Summary of results:

Scores for n_estimators = 330 and min_samples_split = 3 is: [0.75845737 0.80960975 0.80261674 0.76968193 0.7450361]
The predictor accuracy is: 0.77708 (+/+ 0.05011)

In this case, the best parameters are n_estimators=330 and min_samples_split=3
Model saved as rfmodel.pkl.

The Confusion matrix for the model with these specific parameters is:

[[8350 442 923]
 [901 4427 333]
 [1964 406 4419]]

DECISION TREE:

Trying different min_samples_split values to choose a good one before saving the model:

Scores for min_samples_split = 2 are: [0.56044204 0.57748703 0.58515678 0.58470562 0.55437726]
The decision tree predictor accuracy is: 0.57243 (+/+ 0.02542)
Scores for min_samples_split = 3 are: [0.56698241 0.57861493 0.59418001 0.57884051 0.55595668]
The decision tree predictor accuracy is: 0.57491 (+/+ 0.02565)
Scores for min_samples_split = 4 are: [0.55728462 0.57838935 0.58854049 0.58177307 0.55189531]
The decision tree predictor accuracy is: 0.57158 (+/+ 0.02870)
Scores for min_samples_split = 5 are: [0.55593144 0.57184751 0.5907963 0.58244981 0.54963899]
The decision tree predictor accuracy is: 0.57013 (+/+ 0.03103)
Scores for min_samples_split = 6 are: [0.5563825 0.57590796 0.59372885 0.58064516 0.55009025]
The decision tree predictor accuracy is: 0.57135 (+/+ 0.03205)
Scores for min_samples_split = 7 are: [0.55593144 0.57681029 0.58944282 0.58267539 0.55685921]
The decision tree predictor accuracy is: 0.57234 (+/+ 0.02725)
Scores for min_samples_split = 8 are: [0.55209743 0.57478006 0.59372885 0.58425446 0.55234657]
The decision tree predictor accuracy is: 0.57144 (+/+ 0.03360)
Scores for min_samples_split = 9 are: [0.55345061 0.57274983 0.59169862 0.57816377 0.55482852]
The decision tree predictor accuracy is: 0.57018 (+/+ 0.02897)

The best accuracy among those tested results with min_samples_split = 3.
Save the model with this parameter as decisiontreemodel.pkl and get the confusion matrix:

[[5860 1354 2501]
 [1372 3368 921]
 [2404 921 3464]]

14 March (Project help)

For PSSM model prediction, the input must be converted to a PSSM file first. For this aim, I took three FASTA proteins examples and I run the psi-blast bash that generates the psiblast and the pssm files. Then, the input for PSSM model is obtained in the same way as the input for training of the model was obtained. Now it is possible to use the PSSM model previously created to predict the topology of the PSSM files.

Computing the F1 score gives information about how accurate the prediction is. It is calculated by comparing the predicted topology by the model with the real topology (in case it is known). Since I

selected three proteins for PSSM input whose topology is known, it is possible to compute the F1 score for such predictions:

F1 score for "testPSSM.txt" prediction with PSSM model is = 0.612357838919

50 EXTRA PROTEIN SEQUENCES

Extract other protein sequences and topologies from TOPDB. There are 50 beta barrel proteins that I downloaded and saved in *newdataset.txt*. In order to ease the use of the script I have, I have adjusted the topology types I downloaded to match those of my script. This means, I have created a script that makes the following changes: "m" to "M"; "I" to "i"; "O" to "o"; Apart from these three letters, some of the proteins also had "X" at the beginning of the topology line. Looking at the dataset that was assigned to me, all the proteins contained a long stretch of "i" at the beginning, so this is why I have decided to turn all these "X" topologies into "i". The sequences with these changes are saved in *modifiednewdataset.txt.* Having all this done, the original script is now able to predict the *fastadataset.fasta* file that contains the 50 extracted sequences. The predicted topologies are saved in predicted50seqs.txt.
Therefore, I have the predictions in predicted50seqs.txt, and the real topologies in *modifiednewdataset.txt*. I can now calculate the F1 score of the prediction.

F1 score of 50 new extracted proteins prediction with linear model = 0.547542177694

I do the same for the testfile I created at the beginning:

F1 score for testfile.txt prediction with linear model = 0.633603071884

Table F1 scores: (random forest model only prints "iiiiiiiii" as predicted topologies)

	Finalizedmodel	rfmodel	decisiontreemodel	PSSMmodel
testfile.txt	**0.633603071884**	**0.14507361961**	**0.35832945942**	x
fastadataset.txt	**0.547542177694**	**0.103803850074**	**0.286370584233**	x
testPSSM.txt	x	x	x	**0.612357838919**

After doing all these steps, the only thing that is left is to analyze all the data and write the final report. There you can find all the conclusions and deeper explanations of the results shown here.

16 March (Project help)

I attended the Journal Club at SciLife for a background presentation about protein contact prediction with bioinformatics tools, in particular deep learning. It is focused on a paper called "*Accurate De Novo Prediction of Protein Contact Map by Ultra-Deep Learning Model".* We had a lab group meeting where we listened to the new findings of the group members.