**Project plan**

**- for prediction of membrane alpha helix topology**

**Information about my dataset:-**

The Dataset is of membrane alpha helices. The data file is in 3 lines per a protein sequence with a protein id, protein sequence and the feature sequence. The features have 3 possibilities Inside the cell, in the membrane or outside the membrane

**Overview:**

**Preparation:**

* Created a bash script and a git hub repository which has structure for my project

**Main objects**

* 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
* **Add evolutionary information by running psi-blast and extracting the information**
* **Train a SVM using multiple sequence information**
* **Optimize the performance of the SVM**
* Analyze the results and compare it to previous work
* **Use random forests and a simple decision tree and compare the performance with the SVM performance.**
* **Extract the data from 50 other proteins and test the performance**
* Review the state of art for your predictor
* **Write a report**

**In-depth steps:**

**1. Extract the feature from your dataset**

**Parsing: -** A new python script was created and label *parse.py*. It objective is to open my data file at the beginning of the script and parse the file by sorting through the dataset text file *membrane-alpha.3line.txt* and separating the elements line by line. The separation of the different elements was done by the **s**plitting the elements into their own list with the condition if/else of the remainder by the modulator of 3.

- id labels → idlist

- sequences → seqlist

- features → feat\_list

The lists were indexed with the use of an enumeration of the list.The different elements were simulatanoesly written to a output file in the forloop – *idlist.txt*, *seqlist.txt* and *feat\_list.txt*.

**Note: I am to decide if to leave this as a script or if to define it as a function. This is a question I going to decide on as I code more as I want to make it easy to reuse code and also to create flow in the final program.**

**2.Create cross-validated sets.**

**Separating into train and test datasets:**

To extract the features using the sci-kit learn OneHotencoder or the Dictvectorizer into a sparse matrix format on which I can use for an input into sklearn.

Either Using the cross\_valid\_sort in scikit learn

I will need to divide the dataset into 3 or 5 different file on which to use for cross-validation in order to train and test a SVModel. Train a SVM on sequence info with sklearn. After which the model will be tested on a test set.

to be continued………...