Project Diary

Name: Youcheng ZHANG
Programme: Molecular Technique in Life Science
Course: Project in molecular Life science (KB8024/KB8025)

Date	Work
Feb 15th, 2018	1) Read *Noble WS (2009) A Quick Guide to Organizing Computational Biology Projects. PLOS Computational Biology 5(7): e1000424.* *https://doi.org/10.1371/journal.pcbi.1000424*
	2) Read Linux Tips and Tricks + How to organize your project
	3) Created GitHub account
Feb 16th	1) Practice Bash Command: http://swcarpentry.github.io/shell-novice/
	2) Practice Git Command: http://swcarpentry.github.io/git-novice
Feb 17th	1) Practiced Bash Command, Write bash scripts
	2) Start literature search about “Protein secretion”, “Signal peptide”, and “Gram-negative bacteria”
Feb 18th	Literature review
Feb 19th, 2018	1) Reorganized folders on GitHub
	2) Finished week1 assignment: createfolder.sh https://github.com/YouchengZHANG/project/tree/master/assignment/week1
Feb 20th	1) Add Project Diary
	2) Start literature search about “Signal peptide prediction”, “Machine learning approaches”, “Neural network method” and “Random Forest”
	3) Start week2 assignment: Summary of 5 relevant papers
	4) Finished Python Command: http://swcarpentry.github.io/python-novice-inflammation/
Feb 21st	1) Write and Add week2 assignment: List_of_papers_and_Summary.pdf https://github.com/YouchengZHANG/project/tree/master/assignment/week2
	2) Read *Costa, T. R. et al. Secretion systems in Gram-negative bacteria: structural and mechanistic insights. Nature reviews. Microbiology* ***13****, 343-359, doi: 10.1038/nrmicro3456 (2015).*
	3) Read *Chou, K. C. Prediction of protein signal sequences. Current protein & peptide science* ***3****, 615-622 (2002).*
	4) Write week2 assignment: Project_plan
Feb 22nd	1) Read *Zeiler, M.D. and Fergus, R., 2014, September. Visualizing and understanding convolutional networks. In European conference on computer vision (pp. 818-833). Springer, Cham*.
	2) Learn sklearn and one-hot encoding
	3) Extract feature using part of the raw data
Feb 23rd	1) Journal Club and Learn convolutional networks
	2) Learn sklearn.svm and cross-validation
Feb 24th	1) Write Feature Extractor
	2) Run SVM with different kernel and parameters and Run different cross-validation
	3) Write Window size operator
	4) Learn how to save and load the trained model

Feb 25th

1) Read and Learn how to process PSSM *Jones, D.T., 1999. Protein secondary structure prediction based on position-specific scoring matrices1. Journal of molecular biology, 292(2), pp.195-202*.

2) Read literatures about "PSSM normalization from raw profile matrix value" and "scaling window size in signal peptide prediction":

Sharma, R., Sharma, A. et al. 2015. Predict gram-positive and gram-negative subcellular localization via incorporating evolutionary information and physicochemical features into Chou's general PseAAC. IEEE transactions on nanobioscience, 14(8), pp.915-926.

Chou, K.C., 2001. Prediction of signal peptides using scaled window. peptides, 22(12), pp.1973-1979.

3) Try to write both bash and python scripts for separating each sequence information into single files and then running PSI-BLAST locally

4) Learn additional Bash Command: variable assignment, calculation, input/output, $ sed/awk

5) Learn how to manage background jobs: $ &, $ nohup, $ screen

6) Write PSSM extractor

Feb 26th , 2018

1) Test PSI-BLAST locally, change parameter e.g. -num_thread, -word_size to estimate running time

2) Learn background command: $ jobs, $ ps -aux, $ kill, $ lscpu, $ top -H

2) Write PSSM window size operator

3) Modify the window size parameter from [(i – n)…i…(i + n)] to [(i – m)…i…(i + n)] where the two edges of window size could be different

Feb 27th

1) Test PSI-BLAST with uniref90, uniref50 and swissprot database, as well as with different evalue

2) Run SVM on various number of samples in raw dataset to evaluate and estimate the running time

3) Look for solutions to speed up the training process

Feb 28th

1) Modify the Feature Extractor without using OneHotEncoder() command

2) Modify the Feature Extractor to a user-friendly program with sys.argv

3) Run PSI-BLAST on the raw dataset (9357 sequences) and Get the .align / .pssm files

4) Test the PSSM editor and extractor

5) Learn different classifiers in sklearn: RandomForestClassifier; OneVsRestClassifier; BaggingClassifier, etc.

6) Try to find solutions of 'Memory Error' problem

7) Practice presentation

Mar 1st

1) Split the original dataset into different number of partitions

2) Test every different subsets used to train model

3) Practice presentation

4) Finished week3 assignment:

https://github.com/YouchengZHANG/project/tree/master/assignment/week3

Mar 2nd

1) Do presentation in the group and Write peer review on presentation by group member

2) Try to find solutions on dealing with unbalanced dataset

3) Solve 'Memory Error' problem by splitting the original dataset into subsets, and use the subsets as the dataset for model training

4) Read articles about how to preprocess the sequence data when predicting signal peptides: *Petersen, T.N., Brunak, S., von Heijne, G. and Nielsen, H., 2011. SignalP 4.0: discriminating signal peptides from transmembrane regions. Nature methods, 8(10), p.785.*

5) Create PSSMeditor

Mar 3rd

1) Use CD-HIT to perform homology reduction on large dataset: http://weizhongli-lab.org/cd-hit/

2) Process dataset and cut down the length of every sequence to the first 70 amino acid

3) Create PSSMeditor with window-size operating function

4) Try different PSSM normalization functions

Mar 4th

1) Learn how to handle imbalanced dataset: up/downsampling, changing performance matrix, penalize algorithms(cost-sensitive training), tree algorithms

2) Learn how to evaluate the trained model: accuracy, AUROC, MCC

Mar 5th , 2018

1) Write python script to test all the possible parameters automatically

2) Find 50 other proteins with the known structure(only with signal peptides, with transmembrane regions and with neither signal peptides nor transmembrane regions) for further prediction

3) Learn how to evaluate the trained model: sensitivity, ROC, recall, precision

Mar 6th

1) Run python script to test possible parameters(window-size, SVM kernel, class_weight) automatically: optimizeP.py (single sequence) , PSSMeditor_2.py (PSSM)

2) Read article about performance evaluations of model: *Powers, D.M., 2011. Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation.*

3) Learn matplotlib.pyplot command

4) Prepare for presentation

Mar 7th

1) Prepare for presentation

2) Learn GridSearchCV() for optimization

Mar 8th

1) Write three optimizer python scripts using GridSearchCV(): SVM_optimizer, PSSM_SVM_optimizer, RFC_optimzer

2) Optimize parameters on both single sequence model and PSSM model: window-size, SVM kernel, class_weight, number of residues used for training (still waiting for the results)

3) Optimize parameters on single sequence model: C, gamma, degree (still waiting for the results)

4) Learn GridSearchCV()

5) Write self-evaluation for the presentation

6) Add week4 assignment:

https://github.com/YouchengZHANG/project/tree/master/assignment/week4

Mar 9th

1) Try to write modules and to import functions when needed

2) Read website article about Random Forest: *TAVISH SRIVASTAVA , JUNE 9, 2015, Tuning the parameters of your Random Forest model*

3) Read website article about Decision Tree: *ANALYTICS VIDHYA CONTENT TEAM , APRIL 12, 2016, A Complete Tutorial on Tree Based Modeling from Scratch (in R & Python)*

4) Optimize RandomForestClassifier parameters on both single sequence model and PSSM model: n_estimators, max_depth, max_features

5) Optimize DecisionTreeClassifier parameters on single sequence model: min_samples_split, max_depth, max_features

Mar 10th

1) Optimize DecisionTreeClassifier parameters on both single sequence and PSSM model: min_samples_split, max_depth, max_features

2) Try BaggingClassifier to speed up SVM (Failed)

3) Reduce the dataset used to optimize SVM parameters and modify some parameters (C, kernel) to speed up SVM, and rerun SVM optimizer

	4) Start writing final report: Introduction, Result(dataset extraction, parameters)
Mar 11th	1) Organize files and scripts on GitHub, make sure the path correct
	2) Improve the predictor by using modules: Table_Creater.py, Window_Sizer_SS.py, Window_Sizer_PSSM.py
	3) Write final report: Introduction, Result(dataset extraction, parameters)
Mar 12th , 2018	1) Add position frequency matrix model and use module: Window_Sizer_PFM.py
	2) Write final report: Introduction, Result(dataset extraction, parameters)
Mar 13th	1) Process data analysis
	2) Write final report: Introduction, Result, Discussion
Mar 14th	1) Add modules for final predictor: Predict_Separater.py, Predict_Preprocessor.py, Predict_PSSM_Processor.py, Predict_PFM_Processor.py
	2) Process data
	3) Write final report: Result, Discussion
Mar 15th	1) Check every scripts and modules
	2) Write final report: Add figures
Mar 16th	1) Write final report: Add figures
	2) Upload final models and predictors: PSSM_Based_Predictor.py, PFM_Based_Predictor.py, Sequences_Based_Predictor.py https://github.com/YouchengZHANG/project/tree/master/final
	3) Add User_Manual (contains required steps to run the predictors properly): User_Manual.txt https://github.com/YouchengZHANG/project/tree/master/final
Mar 17th	1) Add Example fasta file and Example predicted output: 50_proteins.txt, /50_proteins_result/
	2) Finish final report
	3) Double check
Mar 18th	1) Triple Check
Mar 19th , 2018	1) Submit Final Report and Code