Applications of Logistic Regression in Protein Contact Map Prediction

Sritika Chakladar[1, *, +], **Rahul Alapati**[1, +], **Vineet Nayak**[1, +], **Anuj Gupta**[1, +] **and Austin Ream**[2, +]

[1]Graduate Student, Computer Science and Software Engineering, Auburn University, 36830, USA [2] Undergraduate Student, Computer Science and Software Engineering, Auburn University, 36830, USA

[*] szc0098@auburn.edu

[+] these authors contributed equally to this work

ABSTRACT

Prediction of protein contact map is of greater importance since it can improve the prediction of protein 3D structure. Protein contacts contain key information for the understanding of protein structure and function. This report lays out the implementation of Logistic Regression a non-linear regression algorithm, that has been used to train a binary logistic model to estimate the probability of contacts based on the position specific scoring matrices generated by PSI-BLAST. Two variants of gradient ascent namely: Batch & Stochastic and MCLE estimation have been used to learn the weight vectors during model training. Accuracies of top L/10, L/5 and L/2 predicted contacts are calculated to evaluate the classification performance of the model.

1. Introduction

To understand the mechanism of protein functions at the molecular level, it is usually necessary to determine their structures. However, determining 3D protein structure by experimental methods are high-cost, time-consuming and challenging. Fortunately, protein contact map or residue-residue contact map prediction simplifies the problem of protein structure prediction. The contacts in a protein sequence are predicted in the form a contact map, and then used in 3D protein structure prediction.

Contact Map of a protein, is a binary symmetric matrix and is a useful and simplified representation of the 3D protein structure. Contact Map contains important structural information of a protein, especially the backbone information and hence is crucial in predicting the 3D protein structure. The protein contact map prediction refers to the problem of predicting the spatial closeness of residue pairs in a folded protein structure. Contacts occurring between

sequentially distant residues, i.e. long-range contacts, impose strong constraints on the 3D structure of a protein and are particularly important for structural analysis, understanding the folding process and predicting the 3D structure. Even a small set of correctly predicted long-range contacts can be useful for improving ab initio structure prediction for proteins without known templates.

Logistic regression is a classification algorithm which is used to assign observations to a discrete set of classes. It transforms the input training dataset using a logistic function to return a probability which is mapped to discrete response classes based upon the prediction threshold. The learning algorithm uses regression coefficient to reduce the prediction error. It discovers the best value of the coefficient based on the prediction error and iteratively updates them until the stopping criteria is met.

Gradient ascent optimization is an iterative algorithm for finding the maximum of the cost function by taking steps proportional to the positive of the gradient at the current point. It has two variants: Batch and Stochastic Gradient Ascent. Batch gradient ascent calculates the gradient of the cost function by using the sum of the cost of each sample in every iteration. On the other hand, stochastic gradient ascent is a stochastic approximation of the gradient ascent optimization algorithm and it's an iterative method for maximizing the cost function over a single instance in the dataset. We use the gradient ascent optimization algorithm to update the regression coefficient vector to reach convergence quickly.

In this project, we train a binary logistic model to estimate the probability of contacts based on the position specific scoring matrices generated by PSI-BLAST. We implement both the variants of gradient ascent namely: Batch & Stochastic and MCLE estimation to learn the weight vectors during model training. We use these models to predict the residue-residue contacts in a protein sequence.

2. Methods

The protein sequences in Multi-FASTA format, the actual intra residue distances (d) from the rr file and the position specific scoring matrices have been used to train and test the binary logistic model. Gradient ascent optimization algorithms and MCLE Estimation have been used to learn the weight vector. The learned weight vector is used in residue-residue contact prediction in a protein sequence.

The input protein sequences and their respective actual intra residue distances (d) classes have been curated into non-overlapping sets of Training (75%) and

Test (25%) datasets using the simple random sampling without replacement.

Now, PSIBLAST and the nr database have been used to generate train and test PSSMs for protein sequences in train and test datasets, respectively.

BLAST (Basic Local Alignment Search Tool) is a sequence similarity search method, in which a query protein is compared to protein sequences in a target database to identify regions of local alignment and report those alignments that score above a given score threshold. Position-Specific Iterative (PSI)-BLAST is a protein sequence profile search method that builds off the alignments generated by a run of the BLASTp program.

A sample PSSM for a sequence is shown below in figure 1:

>sequence
TIKVLFVDDHEMVRIGISSYLSTQSDIEVVGEGASGKEAIAKAHELKPDLILMDLLMEDMDGVEATT
QIKKDLPQIKVLMLTSFIEDKEVYRALDAGVDSYILKTTSAKDIADAVRKTSRGESVFEPEVLVKMR
NRMKKRAELYEMLTEREMEILLLIAKGYSNQEIASASHITIKTVKTHVSNILSKLEVQDRTQAVIYAF
QHNLIQ

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

Figure 1. Sample PSSM for a sequence.

The PSSM contains 20 values for each residue in a protein sequence. For a protein sequence of N residues, there will be N * 20 PSSM Values.

Now, for feature generation a sliding window of 5 around central residue is used, as shown in Figure 2.

1	T	-1	-3	-2	-3	-3	-2	-2	-3	-3	-3	-3	-2	-3	-4	-3	3	6	-4	-3	-2
2	I	-3	-5	-6	-6	-3	-5	-5	-6	-6	7	0	-5	2	-2	-5	-5	-2	-5	-4	2
3	K	-3	7	-2	-3	-5	0	-2	-4	-3	-5	-5	4	-4	-5	-4	-1	1	-5	-4	-4
4	V	-2	-5	-5	-6	-3	-5	-5	-6	-6	3	1	-5	-1	-3	-5	-4	-2	-5	-4	6
5	L	-2	-5	-6	-6	-3	-4	-5	-6	-5	1	5	-5	3	-1	-5	-4	-3	-4	-3	2
6	F	-3	-5	-6	-6	-3	-5	-5	-6	-5	5	4	-5	-1	1	-5	-5	-3	-4	-3	3
7	V	4	-4	-5	-5	1	-4	-4	-4	-5	0	-2	-4	-2	-4	-4	-3	-2	-5	-4	6
8	D	-4	-4	-1	8	-6	-2	0	-4	-3	-6	-6	-3	-6	-6	-4	-2	-3	-7	-6	-6
9	D	-4	-4	-1	8	-6	-3	-1	-4	-3	-6	-6	-3	-6	-6	-4	-2	-3	-7	-6	-6
10	H	-3	-2	-2	-3	-5	6	-1	-4	9	-5	-5	-2	-3	-4	-4	-3	-3	-5	-1	-5

Figure 2. Sliding window of 5 around the central residue 'L'.

For the residue L, we consider the PSSM values of K & V above it and the PSSM values of F & V below it to generate a feature vector. Therefore, there will be 20 * 5 = 100 PSSM values for each non-terminal residue like L. For terminal residues like T that do not have one or more neighbors on either side, rows containing 20 values of -1 are used for feature generation.

The N * 20 PSSM values are now converted into N * 100 Feature vectors, as shown below in Figure 3.

20 PSSM values for residue 'L':

5	L	-2	-5	-6	-6	-3	-4	-5	-6	-5	1	5	-5	3	-1	-5	-4	-3	-4	-3	2

100 Feature vector for residue 'L':

5	L	-2	-5	-6	-6	-3	-4	-5	-6	-5	1	5	-5	3	-1	-5	-4	-3	-4	-3	2
		-2	-5	-5	-6	-3	-5	-5	-6	-6	3	1	-5	-1	-3	-5	-4	-2	-5	-4	6
		-3	7	-2	-3	-5	0	-2	-4	-3	-5	-5	4	-4	-5	-4	-1	1	-5	-4	-4
		-3	-5	-6	-6	-3	-5	-5	-6	-5	5	4	-5	-1	1	-5	-5	-3	-4	-3	3
		4	-4	-5	-5	1	-4	-4	-4	-5	0	-2	-4	-2	-4	-4	-3	-2	-5	-4	6

Figure 3. Feature generation from N * 20 PSSM Values.

Now, we combine the 100 PSSM values of each pair of residues (i, j), where $j > i + 5$, in a protein sequence i.e. for a protein sequence of length N, we will have N * 200 PSSM values.

The actual intra residue distance (d) of the two residues (i, j) from the corresponding .rr files has been used to generate the class labels as 1 if (i, j) is in contact i.e. $d < 8$, 0 otherwise. Also, all the residue-residue which are not in contact with a class label 0, have been considered in feature generation.

A sample training instance is shown in Figure 4. Each training instance consists of i, j, 200 PSSM values and the corresponding class label for the i, j residue pair.

A sample training instance:

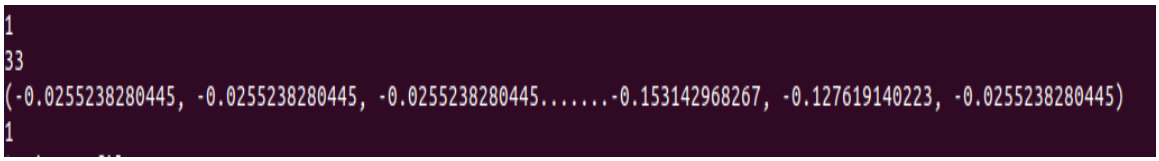

Figure 4. Feature generation (i, j, 200 PSSM Values, class label).

2.1. Logistic Regression Learning on Training Set:

Initially, we implemented the Stochastic gradient ascent based optimization algorithm and MCLE estimation to learn the weight vector of Logistic Regression using the training set.

In Stochastic Gradient Ascent, we begin by selecting a single training instance for each iteration using simple random sampling with replacement and initially all the weights are set to zero. In every iteration, we calculate a prediction based on the logistic function using the training instance and the weight vectors as follows:

$$P(z) = \frac{1}{1+e^{z}}$$

where, P(z) = Probability estimate

z = input to the function ($w0 + \sum_i w_i x_i$)

The weight vectors are updated in every iteration using the MCLE estimation as follows:

$$w_i \leftarrow w_i + \eta X_i (Y - P(Z))$$

where,

w_i : Weights

η : learning rate

X_i : Features from the selected training instance

Y : Prediction

P(z) = Probability estimate

Assumption : $X_0 = 1$ for W_o

The learning algorithm uses gradient ascent to discover best values of regression coefficients based on the error in prediction on training data in an iterative process, until convergence is reached. Given a dataset, we want to find the parameter vector '**w'** which maximizes the likelihood.

During the training, we use a variable learning rate (η) which follows an exponential function and varies from 0.045 to 0.009405, i.e. the learning rate is high at higher gradient value and reduces as gradient decreases and we are closer to the maxima.

The above process is repeated until the error in our prediction doesn't change over a few iterations, i.e. change in the weights is less than $\epsilon = 0.0005$ for 10 iterations.

Also, we have implemented mini batch gradient ascent optimization algorithm in combination with MCLE estimation to learn the weight vector of Logistic Regression using the training set.

In mini batch gradient ascent, we begin by selecting a subset of the large training dataset, say, 500 training instances using random sampling with replacement and repeat the above mentioned process of learning weights.

2.2. Challenges faced in training:

1. **Normalization of dataset:** We observed that, the PSSM values for the 150 protein sequences were not normalized, due to which weights learned using gradient ascent optimization were too large and non-converging with the use of general learning rates and stopping criteria.

   Hence, we normalized our features vectors using Euclidean distances. Using Euclidean distances, our feature vector values were normalized in the range of 0 and 1.

2. **Batch Gradient Ascent:** The implementation of batch gradient ascent on the whole training dataset was computationally intensive and time taking. Hence, we used mini batch gradient ascent where we learn the weights on a randomly selected small subset of the training dataset.

2.3. Logistic Regression Classification on Test Set:

For classification, we use the weights learned as a part of training the binary logistic model and the logistic sigmoid function to classify the residue-residue pairs in the protein sequence as 1 if in contact, 0 otherwise.

The residue-residue pairs are classified as in contact (1) if the probability of the prediction is greater 0.5, not in contact (0) otherwise.

The prediction is calculated using the following equation:

$$P(z) = \frac{1}{1+e^{z}}$$

where, P(z) = Probability estimate

z = input to the function ($w0 + \sum_i w_i x_i$)

Accuracy:

The 'top' L/10, L/5, L/2 predicted contacts are used to evaluate the classification performance averaged over the proteins in the test dataset, where L is the length of the protein sequence and 'top' contacts are the contacts predicted with high probabilities.

3. Results

The following is a sample output showing the residue-residue contacts in a protein sequence in RR format. The contacts are sorted in the non-increasing their probabilities:

```
KTRWTREEDEKLKKLVEQNGTDDWKVIANYLPNRTDVQCQHRWQKVLNPE
8 17 0 8 0.510280046958
8 50 0 8 0.510017163785
9 17 0 8 0.509908357755
9 50 0 8 0.509645466791
7 17 0 8 0.509619764289
7 50 0 8 0.509356867476
8 36 0 8 0.509345703421
8 22 0 8 0.509339780959
8 48 0 8 0.509103081724
8 32 0 8 0.509072664174
8 21 0 8 0.509068859846
9 36 0 8 0.50897398746
9 22 0 8 0.508968064837
8 29 0 8 0.508807248786
9 48 0 8 0.508731359239
9 32 0 8 0.508700940884
9 21 0 8 0.508697136455
7 36 0 8 0.508685373931
7 22 0 8 0.508679451187
8 33 0 8 0.508534607826
7 48 0 8 0.50844274083
9 29 0 8 0.508435518584
7 32 0 8 0.508412321872
7 21 0 8 0.508408517369
10 17 0 8 0.508232047148
9 33 0 8 0.508162870742
7 29 0 8 0.508146894409
23 50 0 8 0.507986955242
8 23 0 8 0.507979473289
10 50 0 8 0.507969124657
8 20 0 8 0.507962141478
7 33 0 8 0.507874241431
8 45 0 8 0.507822080078
8 41 0 8 0.507695762295
9 23 0 8 0.507607722876
9 20 0 8 0.507590390664
8 14 0 8 0.5075903665
```

The following are the accuracies of our logistic model trained using **Stochastic Gradient Ascent:**

Protein	L10	L5	L2	TP	TN	FP	FN
1gzc.rr	82.6086956522 %	91.4893617021 %	93.2773109244 %	0	26535	0	726
1jos.rr	90.0 %	95.0 %	98.0 %	0	4296	0	169
5ptp.rr	95.4545454545 %	97.7272727273 %	98.1981981982 %	0	22783	0	653
1dbx.rr	93.3333333333 %	93.3333333333 %	97.3333333333 %	0	10053	0	387
1lo7.rr	92.8571428571 %	96.4285714286 %	98.5714285714 %	0	8758	0	287
1w0h.rr	94.7368421053 %	97.3684210526 %	98.9690721649 %	0	17363	0	403
1aba.rr	87.5 %	94.1176470588 %	97.6744186047 %	0	3181	0	140
1kw4.rr	83.3333333333 %	92.3076923077 %	97.0588235294 %	0	1881	0	72
1h4x.rr	90.9090909091 %	90.9090909091 %	94.5454545455 %	0	5265	0	195
1i1j.rr	90.0 %	95.2380952381 %	98.1132075472 %	0	4795	0	255
1hfc.rr	93.3333333333 %	96.7741935484 %	97.4358974359 %	0	11137	0	339
1tif.rr	85.7142857143 %	93.3333333333 %	94.7368421053 %	0	2369	0	116
1g2r.rr	88.8888888889 %	88.8888888889 %	95.7446808511 %	0	3757	0	159
2mhr.rr	90.9090909091 %	95.652173913 %	96.6101694915 %	0	6179	0	149
1pch.rr	87.5 %	94.1176470588 %	95.4545454545 %	0	3218	0	185
1bdo.rr	75.0 %	87.5 %	90.0 %	0	2583	0	192
2cua.rr	91.6666666667 %	95.8333333333 %	98.3606557377 %	0	6473	0	313
1c52.rr	84.6153846154 %	92.3076923077 %	96.9230769231 %	0	7639	0	236
1aoe.rr	94.7368421053 %	97.3684210526 %	98.9583333333 %	0	16988	0	403
1ag6.rr	88.8888888889 %	94.7368421053 %	97.9591836735 %	0	4134	0	237
1ku3.rr	83.3333333333 %	91.6666666667 %	93.3333333333 %	0	1483	0	57
1cjw.rr	81.25 %	84.8484848485 %	86.7469879518 %	0	12531	0	349
1a3a.rr	92.8571428571 %	96.5517241379 %	98.6111111111 %	0	9405	0	325
1mk0.rr	77.7777777778 %	89.4736842105 %	91.6666666667 %	0	4005	0	181
1ek0.rr	93.75 %	96.9696969697 %	98.8095238095 %	0	12807	0	396
1i4j.rr	90.9090909091 %	95.4545454545 %	98.1818181818 %	0	5230	0	230
1i5g.rr	92.8571428571 %	92.8571428571 %	95.8333333333 %	0	9291	0	300
1xdz.rr	95.652173913 %	97.8723404255 %	98.3193277311 %	0	26483	0	545
1dqg.rr	84.6153846154 %	88.4615384615 %	91.0447761194 %	0	7910	0	346
2arc.rr	93.75 %	96.875 %	98.75 %	0	11742	0	348
1fqt.rr	90.0 %	85.7142857143 %	90.7407407407 %	0	5075	0	281
1jbk.rr	94.4444444444 %	97.2972972973 %	96.8085106383 %	0	16481	0	355
1a70.rr	88.8888888889 %	94.7368421053 %	97.9166666667 %	0	3966	0	220
1i58.rr	94.4444444444 %	97.2972972973 %	98.9361702128 %	0	16476	0	360
1g9o.rr	77.7777777778 %	83.3333333333 %	88.8888888889 %	0	3472	0	183
1i1n.rr	95.4545454545 %	97.7272727273 %	99.1071428571 %	0	23301	0	570
1brf.rr	80.0 %	90.0 %	84.6153846154 %	0	1037	0	91
1atl.rr	95.0 %	97.5 %	99.0 %	0	18462	0	453

Average L/10 = 89.0723292642 %
Average L/5 = 93.5544516791 %
Average L/2 = 95.8219740864 %
Average accuracy positive contacts = 0 %
Average accuracy negetive contacts = 96.9693035835 %

We achieve an **accuracy of 0% in stochastic gradient ascent** because our model predicts all the probabilities as 0's. This is due to the fact that the weight bias W_0, dominates all the other 200 weights trained by our model. The following are the values of the weights learned using stochastic gradient ascent:

```
=============== recent 10 deltas ===============
[1.1822106803050405e-11, 1.642955435095226e-11, 4.284811873538376e-11, 4.776910565393038e-11, 4.871994928761673e-11, 1.6294877444644457e-11, 3.1248414926257 5e-11, 3.599949094557515e-11, 4.068426132521334e-11, 7.662119415931481e-11]
============================ weights ===================================
[14.862120852093954, -0.3068340424467362, -0.5645102324261682, -0.5919559154002827, -0.6359521475922265, -0.8748815468892239, -0.4930407319865049, -0.5192237281350535, -0.6559887144139089, -0.6135184239245799, -0.5006423028157083, -0.5893203095213102, -0.4742182450250632 7, -0.5062396524322125, -0.7225858119307782, -0.8418707566299786, -0.4067223814493749, -0.3887836292866165, -1.1022047140919125, -0.7037658946550015, -0.38782066586006514, -0.2673175014104722, -0.6081460744947886, -0.7260299514299328, -0.7701833988239132, -0.7844040343754791, -0.5674813421965329, -0.6592163318764798, -0.782373449386169, -0.6756088830884719, -0.38810283789551325, -0.4957328315670076 5, -0.5565488819552907, -0.4233849129404532, -0.6866249586606149, -0.942730478345319, -0.44457529304206433, -0.3395958090799228, -1.0289972891813013, -0.6775480823792572, -0.2345973516195558, -0.2971783680596982, -0.8202687171059093, -0.8419641149305267, -0.9571630252278982, -0.7106034838502338, -0.7275304927400609, -0.8308606933011238, -0.8313687918335759, -0.7880890997852897, -0.24865811386406916, -0.32046043592714624, -0.7750730247996082, -0.3779294986001840 4, -0.5268967874788394, -0.9838331550393444, -0.5382814518071383, -0.40888660292559964, -1.04346560804217 54, -0.6383827883754704, -0.14296756621174211, -0.29955405587812983, -0.6357747003804924, -0.6582931279919 88, -0.7056795720337099, -0.8623305508327862, -0.5360030479949217, -0.5702834623984521, -0.8112868939519353, -0.6982666127062951, -0.4595123598929642 4, -0.5644537034243267, -0.5729699179522403, -0.4934420229057796, -0.6840522383234774, -0.9300190289102966, -0.43464555588488873, -0.3637465717309397, -1.0584446805221 29, -0.658876917953763, -0.3209936341638313, -0.2725481201379408, -0.5370703675909478, -0.5864056080441187, -0.6381152991298793, -0.8911546351416636, -0.43835700520997056, -0.4803976950147731 4, -0.6522588864040227, -0.579370385899744, -0.6270229894223799, -0.6553513221862953, -0.4317197540002587 5, -0.5527034201451749, -0.7612189075704986, -0.8715088557530678, -0.3704513597312684 7, -0.3764505707895551, -1.0266825870434748, -0.6477946573351032, -0.5401732888750778, -0.2293515798132885 5, -0.4616355242205963 4, -0.5249407311202 1, -0.5186123352469275, -0.851482757921263, -0.4034664522620266, -0.3748581124863471 5, -0.5689470643496338, -0.5524490695347273, -0.6026261077040717, -0.6322938331144 86, -0.3692758511667172 4, -0.5266724295636918, -0.8191060868467 33, -0.8601078114164373, -0.3884641489970654 4, -0.3700855802945121 6, -1.1052898489781595, -0.6782881331165902, -0.4230513615614961, -0.2655168652825049, -0.4549173250861444, -0.5328051732449073, -0.5595455418112547, -0.8933479746895162, -0.4182447631019241 6, -0.4393369603708605, -0.7394503204376078, -0.5726206122446678, -0.4878254376395328, -0.5412159530039465, -0.4012026308896511, -0.5091434567054046, -0.7066775692842923, -0.8051664131632604, -0.3900219575859059, -0.3275434512708515, -0.9983915598783 11, -0.5997998555174555, -0.3539545353238802, -0.2907433797572786, -0.7202704133415879, -0.8816494779228029, -0.9045629211139206, -0.7524811502714239, -0.6240268044696639, -0.7398345845164525, -0.8517072885428768, -0.7428198322940265, -0.2470007821114372, -0.3465931310337500 6, -0.6768083630881659, -0.3716479014658918 3, -0.5746864563552234, -0.8757203789313087, -0.5531760684414949, -0.4299000939732966 5, -1.074797296649172, -0.6326278010709231, -0.1746628251173054 4, -0.3222467172900929, -0.5455829994134067, -0.6244041197361697, -0.6801017685153092, -0.8533423867813058, -0.4887570633403971, -0.5413205621046804, -0.8045174102918962, -0.5821368955369175, -0.4170819589112943, -0.4776618509231465, -0.4817044895884192 7, -0.4812423178598619, -0.649461295994092, -0.8479289111841034 8, -0.4619574515987373 6, -0.3412058266012254, -1.0258541407272794, -0.6127633521247882, -0.2765058429895548, -0.2431375893237791 6, -0.4892409421193145 7, -0.4989886611584737, -0.491963474320 96, -0.9664212764498331, -0.3923015515547562 7, -0.4041978919394768, -0.6187359622471781, -0.6017395104632 88, -0.5972147387224545, -0.6516094742849202, -0.3838024658387346, -0.5649166860074479, -0.8627911391225757, -0.6848822856379218, -0.3666018421248239, -0.3996698383437451, -1.1263752778859805, -0.7260850391107 58, -0.4826798651361109 4]
```

The value of W_0 is 14.06, whereas all the other weights are negative values.

The following are the accuracies of our logistic model trained using **mini Batch Gradient Ascent:**

Protein	L10	L5	L2	TP	TN	FP	FN
1gzc.rr	0.0 %	0.0 %	0.0 %	123	18401	8134	603
1jos.rr	0.0 %	0.0 %	0.0 %	17	2893	1403	152
5ptp.rr	0.0 %	0.0 %	0.0 %	84	17166	5617	569
1dbx.rr	0.0 %	0.0 %	0.0 %	43	7340	2713	344
1lo7.rr	0.0 %	0.0 %	0.0 %	25	7009	1749	262
1w0h.rr	0.0 %	0.0 %	1.03092783505 %	62	12179	5184	341
1aba.rr	0.0 %	0.0 %	0.0 %	11	2130	1051	129
1kw4.rr	0.0 %	0.0 %	0.0 %	10	1358	523	62
1h4x.rr	9.09090909091 %	4.54545454545 %	1.81818181818 %	28	3642	1623	167
1i1j.rr	0.0 %	0.0 %	0.0 %	59	2903	1892	196
1hfc.rr	0.0 %	0.0 %	1.28205128205 %	57	7751	3386	282
1tif.rr	0.0 %	0.0 %	0.0 %	14	1291	1078	102
1g2r.rr	0.0 %	0.0 %	0.0 %	28	2129	1628	131
2mhr.rr	9.09090909091 %	8.69565217391 %	3.38983050847 %	32	4020	2159	117
1pch.rr	0.0 %	0.0 %	0.0 %	39	2114	1104	146
1bdo.rr	0.0 %	0.0 %	0.0 %	31	1826	757	161
2cua.rr	0.0 %	0.0 %	0.0 %	60	4308	2165	253
1c52.rr	0.0 %	0.0 %	0.0 %	79	4190	3449	157
1aoe.rr	0.0 %	0.0 %	0.0 %	81	11403	5585	322
1ag6.rr	0.0 %	10.5263157895 %	8.16326530612 %	43	2694	1440	194
1ku3.rr	16.6666666667 %	8.33333333333 %	6.66666666667 %	15	822	661	42
1cjw.rr	0.0 %	0.0 %	0.0 %	47	9160	3371	302
1a3a.rr	0.0 %	0.0 %	1.38888888889 %	46	5941	3464	279
1mk0.rr	0.0 %	0.0 %	0.0 %	21	2804	1201	160
1ek0.rr	0.0 %	0.0 %	0.0 %	45	9123	3684	351
1i4j.rr	0.0 %	9.09090909091 %	7.27272727273 %	40	3508	1722	190
1i5g.rr	0.0 %	0.0 %	0.0 %	46	6073	3218	254
1xdz.rr	0.0 %	0.0 %	0.0 %	79	19530	6953	466
1dqg.rr	0.0 %	0.0 %	0.0 %	83	4299	3611	263
2arc.rr	0.0 %	0.0 %	0.0 %	68	8445	3297	280
1fqt.rr	0.0 %	0.0 %	1.85185185185 %	40	3754	1321	241
1jbk.rr	5.55555555556 %	2.7027027027 %	1.06382978723 %	74	10201	6280	281
1a70.rr	0.0 %	0.0 %	0.0 %	34	2735	1231	186
1i58.rr	0.0 %	0.0 %	0.0 %	81	10909	5567	279
1g9o.rr	0.0 %	0.0 %	0.0 %	29	2091	1381	154
1i1n.rr	0.0 %	0.0 %	0.892857142857 %	92	16499	6802	478
1brf.rr	0.0 %	0.0 %	0.0 %	15	762	275	76
1atl.rr	0.0 %	0.0 %	0.0 %	44	14056	4406	409

Average L/10 = 1.06326422116 %
Average L/5 = 1.15511493778 %
Average L/2 = 0.916344167371 %
Average accuracy positive contacts = 1.61633159153 %
Average accuracy negetive contacts = 96.3475315371 %

We achieve an **accuracy of 1.06% in mini Batch gradient ascent**. This is due to the fact that the weight bias W_0, dominates all the other 200 weights trained by our model. Due to the dominating W_0 value, our model is forced to predict more number of False Positives and less number of True Positives. The following are the values of the weights learned using stochastic gradient ascent:

========== recent 10 deltas ==========
[0.0004962227714559144, 0.0002981198868611205, 0.0002987962723519193, 0.0001005203567537975, 0.0003997944024781512, 0.0003993377492147533, 0.0001997630556112721 4, 0.0002000596041590747, 0.0002004433654922637, 0.0002985813859702825]
========== weights ==========
[-0.0004141671803291079 4, 2.5155895854366857e-05, -0.0002296455335513812, -0.0002324525612163712 4, -0.0001310992922446798 2, 0.0001634068725326265 3, -0.0001340985343908932, -0.0001371337569584525 2, -5.946714937757645 5e-05, 1.9092613158317915e-05, 0.0002447120861776044 5, -5.823227237711335e-05, -0.0001692606807960826, -3.935695625902801e-05, 4.9089793579278 51e-05, -6.982776468458289e-05, -4.8614338823970666e-05, -1.6257342340671 4e-05, -9.748038321925846e-05, -9.874376898552726e-06, 0.0002465298568977053, -6.126983694134217e-05, -0.0002009747028652733 7, -0.0004769636125386596, -0.0006018571101277125, 0.0003924593692384824, -0.0004771613422377 1975, -0.0005512073524640802, -0.0002509948886822563, -0.0003492047931727 68, 0.0004057905075090915, 0.0001892610347755330 1, -0.0003707923947783986, 0.0001791688444449830 4, 0.0001280053737896472 3, 1.9128151672717846e-05, -0.0002717288040855121, -0.0001100052632443575 9, 0.0001925498312769200 5, -1.2514564324737794e-05, 0.0004054702565407813 5, -0.0002528814163002283 3, -0.0009546360002935029, -0.000919788848697887, -0.0011834792382001242, 0.0006934284739534 01, -0.0009823700079957674, -0.0013280043583451656, -0.0005738481078271662, -0.0005328291802303111, 0.0010416869755527 83, 0.0006217038274499545, -0.0010820381338226977, 0.0004034340175808462 7, 0.0009073396569629247, -0.0005385034649523752, -0.0005826832617877308, -0.0002943233241214008, 0.0004940455039870742, 0.0004985608321752142, 0.0008573255780513987, -7.778189236161496e-05, -0.0001216028283836542 4, -0.0004588720681829073, -0.0004009919495683538, 0.0002484401771879336 5, -0.0002561765720355885, -0.0001600828688756757 3, -0.0004537958104683958 5, -0.0003772908730425127, 0.0004864807264717010 6, 0.0004023664654321177 1, -0.0001588245671960726 4, 0.0002507164305061206, 0.0005312079291558 55, -3.7454783162185464e-05, -0.0002264315767209109 6, 0.0001411779358259851 5, 0.0003653161341679060 3, 0.0003857212294246352 7, 0.0004602502841304144, -0.0002890834683766678 7, -2.9213134259182635e-05, 0.0001710660153322145 2, 0.0001613625721766455 5, 0.0001794299036162898 6, -5.177880379812575e-05, 8.0686481437201436e-05, 0.0002562977353410643, 0.0002521284213348209, -0.0001680847662615145, -0.0001587345893219544, -0.0001641879269420880 5, -0.0001298948249671039, 0.0001447878640314995 1, -8.191234398355285e-05, 8.257967368359922e-05, -0.0001359254577958535 4, -5.213524881972107e-07, 0.0001332339738511676, -0.0001665788197281601 5, -0.0001740607740154289, -0.0001889985371020 09, -0.0001362202992949036 3, 5.455768356790573e-05, 0.0001386873321743362 6, -7.846527632168012e-05, 0.0001409751505920046, 2.6402448808735014e-05, -3.222194054768465e-05, 0.0002320677263028652 3, 4.4118688390180376e-05, 7.881892342764353e-05, -8.532438259544798e-05, 0.0001017927420596632 6, 1.6922905139028 42e-05, -0.0001561147608412354, -2.2763838577324896e-05, -1.0006765166621265e-05, 1.7514748368956183e-05, 0.0003399116311761613, -0.0001765511948268039 4, -6.403166162003415e-05, -0.0001425050267692200 5, -0.0002917692109689933, 0.0003950975199273933, -0.0001181461314850777 7, -5.8491198631683296e-05, -0.0002398880526638107 6, 0.0001142069931911618 8, 0.0004218020684582002 5, 0.0002818897626472474, -0.0001837430255659598 2, 0.0001503879071289444 8, 0.0001834042577984833 7, -0.0001802188710011136 3, -0.0001270113842133576 7, 0.0001801007680928135, 4.1551676149304324e-05, 0.0001218958120324228 15, 0.0004980490834059674, -0.0001585932359863328, -0.0009934995322410291, -0.0010068852235562 55, -0.0015564231443022 65, 0.0007779427005910186, -0.0010073673970618338, -0.0014098434346578263, -0.0005964993527854978, -0.0004445096501645521 3, 0.0011430557955736012, 0.0010856445318117316, -0.0011685473430408813, 0.0006639338095727135, 0.0008725562615835435, -0.0006365888226644468 7, -0.0005327857369648616, -7.878590742940436e-05, 0.0004721226693270511 4, 0.0008331067114085 94, 0.0009789900415117458, -0.0002567055229833395 5, -0.0001922785914615914 2, -9.058075693677188e-05, -0.0003230971121202341, 0.0004331978388793443, -0.0002925452609283615, -0.0003285277385171846, -0.0002068991563588341 6, -8.460883917648554e-05, 0.0002879989526956179, 6.462632005979517e-05, -0.0002518613440901785, -2.5714115773742 14e-05, 0.0002026362887707731 5, -0.0002276627939349332 4, -0.0001591593281821929 5, 5.228944705596676e-05, 0.0001608628584332654 5, 0.0001643118550006359, 0.0003298112532174322, -0.0002669787882153092, -0.0001181958820402733, 5.533792353966006e-05, 0.0002051255841069131 9, 0.0002944935923741761, -5.0743878521652254e-05, -1.6340319263010 2e-05, -2.7389887487740786e-05, 5.0301625360754 67e-05, -3.3039074963952085e-05, -0.0001977856600160178 5, -0.0002773644352000947, -0.0001846515801735244, 3.8831419415839 29e-05, -0.0003026495017419 65, -5.765081808876261e-05, -0.0001386261332545275 7, 0.0001301401114918692 3, -4.6987109960023 54e-05, -1.7206180711757466e-05]

4. Challenges due to Class Imbalance

We observed that, the number of non-contact pairs in the training dataset largely outnumbered the number of contact pairs. Due to this class imbalance, initially our model was incorrectly trained to predict non-contact pairs.

Hence, we balanced our dataset with equal number of contact and non-contact pairs. We randomly selected non-contact pairs to equate them to the number of contact pairs.

Before class balancing, our overall prediction accuracy was 97% and both of our models (stochastic & batch) were only predicting non-contact pairs. Due to class balancing, both of our models improved, and the model trained using mini batch gradient ascent gained a contact accuracy of 1.06%.

5. Discussion

We performed different experiments as discussed above to improve the accuracy of our binary logistic model. Our contact accuracy was the 0 % and our model was only predicting non-contacts due to class imbalance. When the contact and non-contact pairs in our training dataset were balanced, our accuracy increased to 1.06 %. These experiments outlay the importance of class imbalance, feature normalization and mini batch gradients.

5. References

1. Wikipedia contributors. (2018, March 21). Protein structure. In Wikipedia, The Free Encyclopedia. Retrieved 19:23, March 26, 2018, from https://en.wikipedia.org/w/index.php?title=Protein_structure&oldid=831674619
2. Pauling, L., Corey, R. B., & Branson, H. R. (1951). The structure of proteins: two hydrogen-bonded helical configurations of the polypeptide chain. Proceedings of the National Academy of Sciences, 37(4), 205-211.
3. Alberts, B., Johnson, A., Lewis, J., Raff, M., Roberts, K., & Walter, P. (2002). The shape and structure of proteins.
4. Anfinsen, C. B. (1973). Principles that govern the folding of protein chains. Science, 181(4096), 223-230.
5. Zhang, H. (2004). The optimality of naive Bayes. AA, 1(2), 3.
6. Schmidler, S. C., Liu, J. S., & Brutlag, D. L. (2000). Bayesian

segmentation of protein secondary structure. Journal of computational biology, 7(1-2), 233-248.
7. Robles, V., Larrañaga, P., Peña, J. M., Menasalvas, E., Pérez, M. S., Herves, V., & Wasilewska, A. (2004). Bayesian network multi-classifiers for protein secondary structure prediction. Artificial Intelligence in Medicine, 31(2), 117-136.
8. Di Lena, P., Nagata, K., & Baldi, P. (2012). Deep architectures for protein contact map prediction. Bioinformatics, 28(19), 2449-2457.
9. Tress, M. L., & Valencia, A. (2010). Predicted residue–residue contacts can help the scoring of 3D models. Proteins: Structure, Function, and Bioinformatics, 78(8), 1980-1991.