哈尔滨工业大学深圳研究生院

机器学习 Project 报告

DNA-binding protein prediction

组　　员

报告日期

Contents

1. Introduction

1.1 Motivation and background

As one of the material bases of various life activities, protein is not only an important component of the structure of all cells, but also an important part of life activities. Therefore, the study of protein molecular structure and function is of great significance to biomedicine, production practice and human life.

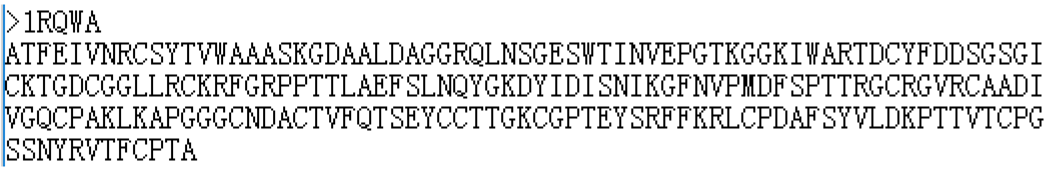
DNA-binding proteins refer to a class of proteins in the organism's proteome that bind to DNA and form complexes. DNA-binding proteins represent a broad class of proteins that are diverse in sequence and structure, as well as in function. Structure can be divided into eight structural groups, further divided into 54 structural families. Functionally, it has different functional roles throughout the genome. It is known that 6-7% of the proteins in the eukaryotic proteome are DNA-binding proteins and are involved in many life activities closely related to the life process of cells, such as DNA transcription, replication, modification, folding, recombination and other DNA-related basic activities.

Histones are a common example of DNA-binding proteins in eukaryotes. Histones can form a disc-like composite structure with DNA called nucleosomes through ionic bonds formed by the basic residues and the DNA acid phosphate backbone. Nucleosomes are an important complex in the chromosome that can be made DNA is organized into a tightly packed chromatin structure. Methylation, phosphorylation and acetylation modifications on basic amino acids serve to modulate the intensity of the interaction with DNA and alter the rate of transcription. Human helicase is also a DNA-binding protein that interacts with single-stranded DNA and is involved in processes such as DNA replication, recombination, and DNA modification.

Therefore, the prediction of DNA-binding protein can help people to further understand the mechanism of protein-nucleic acid interaction and promote the study of the essence of human life.

1.2 Overview of Data

In this study, we have 550 non-DNA-binding proteins sequences in file of protein_neg.txt, and 525 DNA-binding proteins sequences in file of protein_pos.txt. These file includes the highest number of protein sequences with low similarity, which is desirous for model evaluation. The predictor accuracy was tested on this data sets.



```
>1RQWA
ATFEIVNRCSYTVWAAASKGDAALDAGGRQLNSGESWTINVEPGTKGGKIWARTDCYFDDSGSGI
CKTGDCGGLLRCKRFGRPPTTLAEFSLNQYGKDYIDISNIKGFNVPMDFSPTTRGCRGVRCAADI
VGQCPAKLKAPGGGCNDACTVFQTSEYCCTTGKCGPTEYSRFFKRLCPDAFSYVLDKPTTVTCPG
SSNYRVTFCPTA
```

1-1. One kind of training example

For this negative training example, the “1RQWA” is the name of the protein and the following rows are the sequences.

The basic knowledge of protein is that among them, there are 20 letter sequences represent the amino acid sequence of the protein.

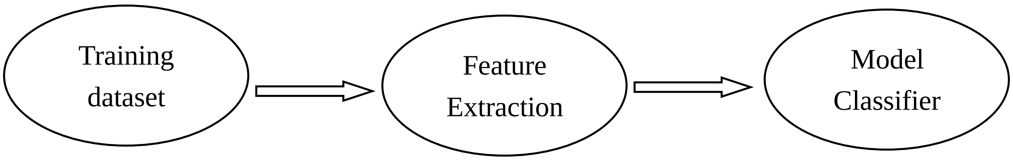
Amino Acid	Symbol
Alanine	A
Cysteine	C
Aspartic	D
Glutamic	E
Phenylalanine	F
Glycine	G
Histidine	H
Isoleucine	I
Lysine	K
Leucine	L
Methionine	M
Asparagine	N
Proline	P
Glutamine	Q
Arginine	R
Serine	S
Threonine	T
Valine	V
Tryptophan	W
Tyrosine	Y

1-2. 20 kinds of amino acids

1.3 Evaluation Standard

In this study, we use the –fold cross validation to evaluate the goodness of our model. Cross-validation is a way to predict the fit of a model to a hypothetical validation set when an explicit validation set is not available. In k-fold cross-validation, the original sample is randomly partitioned into k equal sized subsamples. Of the k subsamples, a single subsample is retained as the validation data for testing the model, and the remaining k-1 subsamples are used as training data. The cross-validation process is then repeated k times (the folds), with each of the k subsamples used exactly once as the validation data. The k results from the folds can then be averaged to produce a single estimation. The advantage of this method over repeated random sub-sampling is that all observations are used for both training and validation, and each observation is used for validation exactly once. 10-fold cross-validation is commonly used. So in the process of training models, I apply 10-fold cross-validation.

1.4 Framework



1-3 framework of our study

In the process of data extraction, we mainly extract four kinds of features: 1) the frequencies of 20 kinds of amino acid of protein. It is 20D. 2) the best 10 of the top-2-gram base on the work of Zhou[1]. It is 10D. 3) local-DPP feature based on the PSSM, according to the work of Wei[2]. It is about 120D. 4) PSSM-DT feature based on the PSSM, according to the work of Zhou[1]. It is 2000D. So, we extract 2150D effective features.

In the process of choosing classifier and building model. We have tried different classifier models: logistic regression, support vector machine, gradient boosting, xgboost and random forest.

We do a lot of work on the feature extraction and choose classification model. Finally, we can get the 2150D features step by step. What’s more, different classification will have great effect on the accuracy of result. So, we do the comparison of them to choose the best.

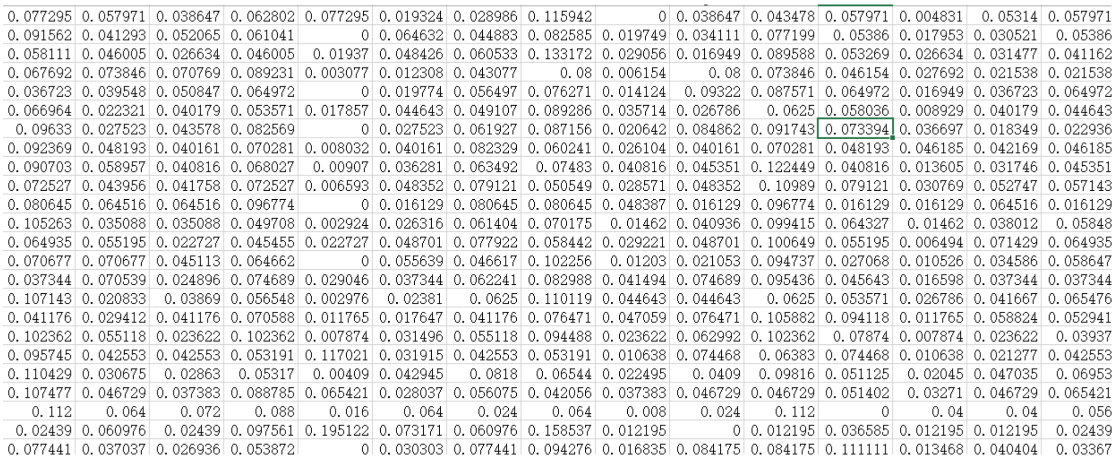
Finally, we can get the best accuracy of 84.0% using the 1250D feature and the gradient boosting classification model.

2. Feature Extraction

2.1 Feature 1 the frequencies of 20 kinds of amino acid

As we all know, there are 20 kinds of amino acid. So, we can easily get the basic feature of the frequencies of the 20 kinds of amino acid of the protein. It is a basic feature and the dimension is 20. The order of the 20 kinds of amino acid is ACDEFGHIJKLMNPQRSTVWY.

The frequencies can be calculated by $\frac{the\ number\ of\ amino\ acid}{L}$, here L means the length of the protein sequence.



0.077295	0.057971	0.038647	0.062802	0.077295	0.019324	0.028986	0.115942	0	0.038647	0.043478	0.057971	0.004831	0.05314	0.057971
0.091562	0.041293	0.052065	0.061041	0	0.064632	0.044883	0.082585	0.019749	0.034111	0.077199	0.05386	0.017953	0.030521	0.05386
0.058111	0.046005	0.026634	0.046005	0.01937	0.048426	0.060533	0.133172	0.029056	0.016949	0.089588	0.053269	0.026634	0.031477	0.041162
0.067692	0.073846	0.070769	0.089231	0.003077	0.012308	0.043077	0.08	0.006154	0.08	0.073846	0.046154	0.027692	0.021538	0.021538
0.036723	0.039548	0.050847	0.064972	0	0.019774	0.056497	0.076271	0.014124	0.09322	0.087571	0.064972	0.016949	0.036723	0.064972
0.066964	0.022321	0.040179	0.053571	0.017857	0.044643	0.049107	0.089286	0.035714	0.026786	0.0625	0.058036	0.008929	0.040179	0.044643
0.09633	0.027523	0.043578	0.082569	0	0.027523	0.061927	0.087156	0.020642	0.084862	0.091743	0.073394	0.036697	0.018349	0.022936
0.092369	0.048193	0.040161	0.070281	0.008032	0.040161	0.082329	0.060241	0.026104	0.040161	0.070281	0.048193	0.046185	0.042169	0.046185
0.090703	0.058957	0.040816	0.068027	0.00907	0.036281	0.063492	0.07483	0.040816	0.045351	0.122449	0.040816	0.013605	0.031746	0.045351
0.072527	0.043956	0.041758	0.072527	0.006593	0.048352	0.079121	0.050549	0.028571	0.048352	0.10989	0.079121	0.030769	0.052747	0.057143
0.080645	0.064516	0.064516	0.096774	0	0.016129	0.080645	0.080645	0.048387	0.016129	0.096774	0.016129	0.016129	0.064516	0.016129
0.105263	0.035088	0.035088	0.049708	0.002924	0.026316	0.061404	0.070175	0.01462	0.040936	0.099415	0.064327	0.01462	0.038012	0.05848
0.064935	0.055195	0.022727	0.045455	0.022727	0.048701	0.077922	0.058442	0.029221	0.048701	0.100649	0.055195	0.006494	0.071429	0.064935
0.070677	0.070677	0.045113	0.064662	0	0.055639	0.046617	0.102256	0.01203	0.021053	0.094737	0.027068	0.010526	0.034586	0.058647
0.037344	0.070539	0.024896	0.074689	0.029046	0.037344	0.062241	0.082988	0.041494	0.074689	0.095436	0.045643	0.016598	0.037344	0.037344
0.107143	0.020833	0.03869	0.056548	0.002976	0.02381	0.0625	0.110119	0.044643	0.044643	0.0625	0.053571	0.026786	0.041667	0.065476
0.041176	0.029412	0.041176	0.070588	0.011765	0.017647	0.041176	0.076471	0.047059	0.076471	0.105882	0.094118	0.011765	0.058824	0.052941
0.102362	0.055118	0.023622	0.102362	0.007874	0.031496	0.055118	0.094488	0.023622	0.062992	0.102362	0.07874	0.007874	0.023622	0.03937
0.095745	0.042553	0.042553	0.053191	0.117021	0.031915	0.042553	0.053191	0.010638	0.074468	0.06383	0.074468	0.010638	0.021277	0.042553
0.110429	0.030675	0.02863	0.05317	0.00409	0.042945	0.0818	0.06544	0.022495	0.0409	0.09816	0.051125	0.02045	0.047035	0.06953
0.107477	0.046729	0.037383	0.088785	0.065421	0.028037	0.056075	0.042056	0.037383	0.046729	0.046729	0.051402	0.03271	0.046729	0.065421
0.112	0.064	0.072	0.088	0.016	0.064	0.024	0.064	0.008	0.024	0.112	0	0.04	0.04	0.056
0.02439	0.060976	0.02439	0.097561	0.195122	0.073171	0.060976	0.158537	0.012195	0	0.012195	0.036585	0.012195	0.012195	0.02439
0.077441	0.037037	0.026936	0.053872	0	0.030303	0.077441	0.094276	0.016835	0.084175	0.084175	0.111111	0.013468	0.040404	0.03367

2-1. the 10D feature1

2.2 Feature 2 the best 10 of the top-2-gram

Protein sequence frequency profile is a matrix of dimension N*L, can be expressed as formula

$$M = \begin{bmatrix} m_{1,1} & m_{1,2} & \cdots & m_{1,L} \\ m_{2,1} & m_{2,2} & \cdots & m_{2,L} \\ \vdots & \vdots & \vdots & \vdots \\ m_{N,1} & m_{N,2} & \cdots & m_{N,L} \end{bmatrix} \qquad （1）$$

L is the number of amino acid residues contained in the protein sequence and N is the number of standard amino acids, usually taken as 20.

The elements in M indicate the frequency of appearance of the corresponding amino acid in the sequence at a specific position in the evolution of the protein sequence. The sequence of M is the frequency spectrum of the amino acid at a certain position in the sequence, that is, a 20-dimensional vector whose 20 elements are the frequencies of occurrence of 20 standard amino acids at this position respectively. So $\sum_{i=1}^{N} m_{ji} = 1$ Where j is a position on the sequence.

For each column in M, 20 of these amino acids are listed in descending order of frequency of occurrence. The ordered matrix M can be expressed as follows:

$$M^{\downarrow} = \begin{bmatrix} m_{1,1}^{\downarrow} & m_{1,2}^{\downarrow} & \cdots & m_{1,L}^{\downarrow} \\ m_{2,1}^{\downarrow} & m_{2,2}^{\downarrow} & \cdots & m_{2,L}^{\downarrow} \\ \vdots & \vdots & \vdots & \vdots \\ m_{20,1}^{\downarrow} & m_{20,2}^{\downarrow} & \cdots & m_{20,L}^{\downarrow} \end{bmatrix} \qquad (2)$$

$$m_{1,j}^{\downarrow} \geq m_{2,j}^{\downarrow} \geq \cdots \geq m_{20,j}^{\downarrow}$$

In the Top-n-gram method, for each position on the protein sequence, the n amino acids that have the highest probability of appearance are sorted and combined according to their appearance frequency, and this combination is a Top-n-gram, in which the frequency of occurrence is the largest Of the amino acids arranged in the Top-n-gram of the first position, the frequency of appearance of amino acids arranged in the second position,

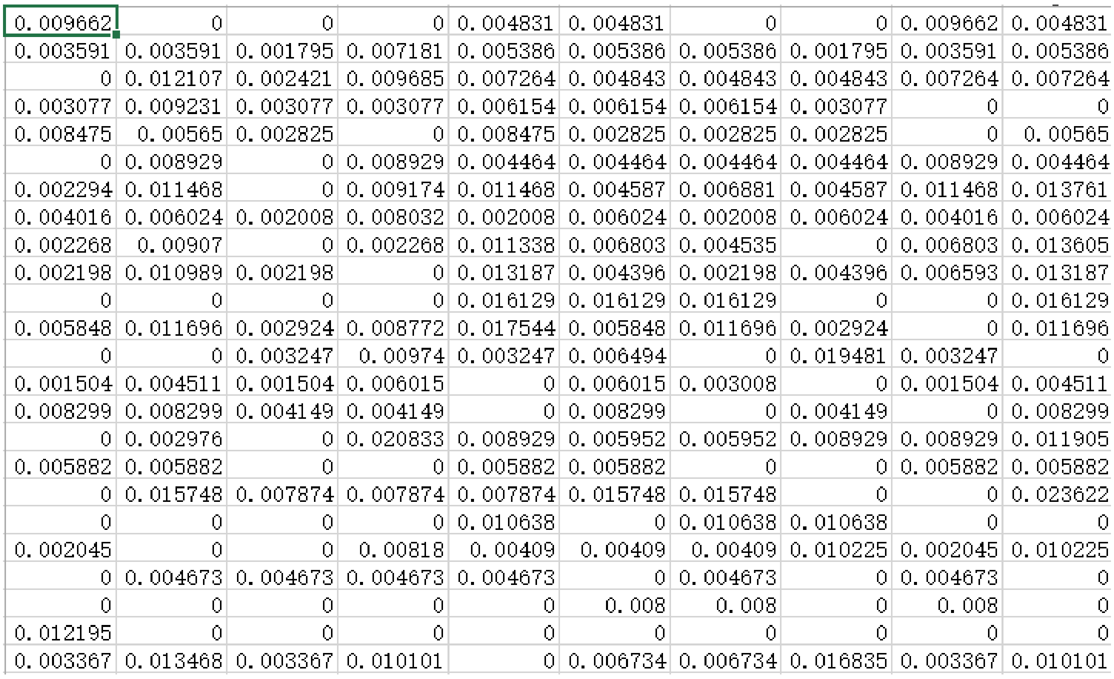
According to the work of Zhou[2], we can get the e top 10 features and their discriminant weights among the 400 features as follow:

numerical order	feature	weight
1	KR	407.312
2	LV	-322.105
3	RK	304.152
4	GA	-250.014
5	VL	-247.943
6	AG	-228.982
7	AV	-197.469
8	EK	-185.799
9	GD	-178.154
10	AL	-176.483

2-2 the top 10 weights of Top-2-gram

So, we can get the feature 2 of 10D as the following figure shows.

2-



0.009662	0	0	0	0.004831	0.004831	0	0	0.009662	0.004831
0.003591	0.003591	0.001795	0.007181	0.005386	0.005386	0.005386	0.001795	0.003591	0.005386
0	0.012107	0.002421	0.009685	0.007264	0.004843	0.004843	0.004843	0.007264	0.007264
0.003077	0.009231	0.003077	0.003077	0.006154	0.006154	0.006154	0.003077	0	0
0.008475	0.00565	0.002825	0	0.008475	0.002825	0.002825	0.002825	0	0.00565
0	0.008929	0	0.008929	0.004464	0.004464	0.004464	0.004464	0.008929	0.004464
0.002294	0.011468	0	0.009174	0.011468	0.004587	0.006881	0.004587	0.011468	0.013761
0.004016	0.006024	0.002008	0.008032	0.002008	0.006024	0.002008	0.006024	0.004016	0.006024
0.002268	0.00907	0	0.002268	0.011338	0.006803	0.004535	0	0.006803	0.013605
0.002198	0.010989	0.002198	0	0.013187	0.004396	0.002198	0.004396	0.006593	0.013187
0	0	0	0	0.016129	0.016129	0.016129	0	0	0.016129
0.005848	0.011696	0.002924	0.008772	0.017544	0.005848	0.011696	0.002924	0	0.011696
0	0	0.003247	0.00974	0.003247	0.006494	0	0.019481	0.003247	0
0.001504	0.004511	0.001504	0.006015	0	0.006015	0.003008	0	0.001504	0.004511
0.008299	0.008299	0.004149	0.004149	0	0.008299	0	0.004149	0	0.008299
0	0.002976	0	0.020833	0.008929	0.005952	0.005952	0.008929	0.008929	0.011905
0.005882	0.005882	0	0	0.005882	0.005882	0	0	0.005882	0.005882
0	0.015748	0.007874	0.007874	0.007874	0.015748	0.015748	0	0	0.023622
0	0	0	0	0.010638	0	0.010638	0.010638	0	0
0.002045	0	0	0.00818	0.00409	0.00409	0.00409	0.010225	0.002045	0.010225
0	0.004673	0.004673	0.004673	0.004673	0	0.004673	0	0.004673	0
0	0	0	0	0	0.008	0.008	0	0.008	0
0.012195	0	0	0	0	0	0	0	0	0
0.003367	0.013468	0.003367	0.010101	0	0.006734	0.006734	0.016835	0.003367	0.010101

2-2 the frequencies of top 10 of the top-2-gram.

2.3 Feature 3 local-DPP

The feature 3 is called local-DPP according to the work of Wei[1].

First we introduce of PSSM**.** A given protein sequence S is represented as $S_{1,}$ $S_{2,}$ . . . S_L, where S_i(1 ≤ i ≤ L) rep-resents the amino acid appearing in the ith position of S, and L is the length of S.

The so-called evolutionary profile of S is the position-specific scoring matrix (PSSM), generated by three iterations of a PSI-BLAST search of the protein database nrdb90. The E-value (expectation value) cutoff for the multiple sequence alignment was 0.001.

The PSSM contains the probability that each type of amino acid is found at each residue position of the protein sequence during the evolutionary process. Hence, the PSSM measures the residue conservation at a given location. The evolutionary information in the PSSM is stored in a matrix of dimensions L * 20 (L rows and 20 columns), formulated as follows:

$$\boldsymbol{P_{original}} = \begin{bmatrix} p_{1,1} & p_{1,2} & \cdots & p_{1,20} \\ p_{2,1} & p_{2,2} & \cdots & p_{2,20} \\ \vdots & \vdots & \vdots & \vdots \\ p_{i,1} & p_{i,2} & \cdots & p_{i,20} \\ \vdots & \vdots & \vdots & \vdots \\ p_{L,1} & p_{L,2} & \cdots & p_{L,20} \end{bmatrix}_{L\times 20} \qquad (3)$$

where a row denotes the corresponding position of the sequence S. For example, the first, second and Lth rows refer to the first, second and Lth positions of S, respectively. The columns represent the corresponding residue type of the 20 amino acids {A, C, D, E, F, G, H, I, K, L, M, N, P, Q, R, S, T, V, W, Y}; for example, the first, second and 20th columns refer to A, C and Y, respectively. An entry $p_{i,j}$ represents the score of the residue at the ith position of S being mutated to residue type j during the evolutionary process ($1 \le i \le L,\ 1 \le j \le 20$). The higher the $p_{i,j}$ score, the more frequent the mutation (in general). Residues at highly mutable sites are likely to be functional.

The steps of extract of local-DPP: 1). The original PSSM is normalized as follows:

$$f_{i,j} = \frac{p_{i,j} - \frac{1}{20}\sum_{k=1}^{20} p_{i,k}}{\sqrt{\frac{1}{20}\sum_{l=1}^{20}\left(p_{i,l} - \frac{1}{20}\sum_{k=1}^{20} p_{i,k}\right)^2}}, \quad (i = 1, 2, \ldots, L;\ j = 1, 2, \ldots, 20) \qquad (4)$$

where p_{ij} represent the original scores of PSSM. The normalized scores ($f_{i,j}$) have a zero mean over the 20 amino acids. A positive (negative) score indicates that the corresponding mutation occurs more (less) frequently in the multiple alignment than expected by chance. The normalized PSSM is represented by

$$\boldsymbol{P_{normalized}} = \begin{bmatrix} f_{1,1} & f_{1,2} & \cdots & f_{1,20} \\ f_{2,1} & f_{2,2} & \cdots & f_{2,20} \\ \vdots & \vdots & \vdots & \vdots \\ f_{i,1} & f_{i,2} & \cdots & f_{i,20} \\ \vdots & \vdots & \vdots & \vdots \\ f_{L,1} & f_{L,2} & \cdots & f_{L,20} \end{bmatrix}_{L\times 20} \qquad (5)$$

2). This step row-fragments the normalized matrix $P_{normalized}$ into n($\geq$ 1) sub-matrices. For convenience, we denote the kth sub-matrix as $P^{k}_{normalized}$($1 \le k \le n$). Each of the first (n − 1) sub-k matrices has L/n rows and 20 columns; the final sub-matrix ($P^{n}_{normalized}$) has (L − (n − 1) ∗ L/n) rows and 20 columns. Note L. that the sizes of the first (n − 1) sub-matrices and the last sub-matrix are equal if and only if L/n is an integer. Moreover, every sub-matrix retains the evolutionary information contained in the original PSSM. Importantly, the fragmentation operation captures the local conservation information, because this information is always embedded in the local regions.

3） To quantize the local conservation information of the protein S, we computed the local Pse-PSSM features for all sub-matrices. However, $P^{k}_{normalized}(1 \leq k \leq n - 1)$ and $P^{n}_{normalized}$ are not necessarily equal in size. Therefore, the features were separately computed for $P^{k}_{normalized}(1 \leq k \leq n - 1)$ and $P^{n}_{normalized}$. For each of the first (n − 1) sub-matrices ($P^{k}_{normalized}(1 \leq k \leq n - 1)$), we computed 20 local features by incorporating the evolutionary information as follows:

$$Part_1 = \{F_j(k) = \frac{1}{L/n} \sum_{i=(k-1)L/n}^{k*L/n} f_{i,j} | 1 \leq k \leq n-1; j = 1, 2, \ldots, 20\} \qquad (6)$$

where $F_j(k)$ denotes the average probability that each residue position in the k^{th} fragmented sequence mutates to residue type j during the evolutionary process. Thus, we obtained (n − 1) × 20 local features containing evolutionary information for the first (n − 1) sub-matrices. To incorporate the sequence-order information, we represent the protein S by

$$Part_2 = \left\{ \Phi_j^{\xi}(k) = \frac{1}{\frac{L}{n} - \xi} \sum_{i=(k-1)\frac{L}{n}}^{k*\frac{L}{n}-\xi} \left(f_{i,j} - f_{(i+\xi),j}\right)^2 | \ \xi = 1, \ldots, \lambda; 1 < \lambda < \frac{L}{n} \right\} \qquad (7)$$

where $\phi_j^{\varepsilon}(k)$ is the average correlation between two coupled residues separated by ξ for amino acid type j in the kth sub- matrix. For example, $\phi_j^{1}(k)$ and $\phi_j^{2}(k)$ are the correlation factors obtained by coupling contiguous residues and every two residues along the protein chain, respectively, for amino acid type j in the kth sub-matrix. The maximum L should be the minimum length of the sequences in the dataset.

After combining the local features containing evolutionary information ($Part_1$) and sequence-order information ($Part_2$), we obtained $20(n-1)(1+\lambda)$ local Pse-PSSM features for the first $(n-1)$ sub-matrices. The space representation of the features is given by

$$FV(n\text{-}1) = (part1, part2) \tag{8}$$

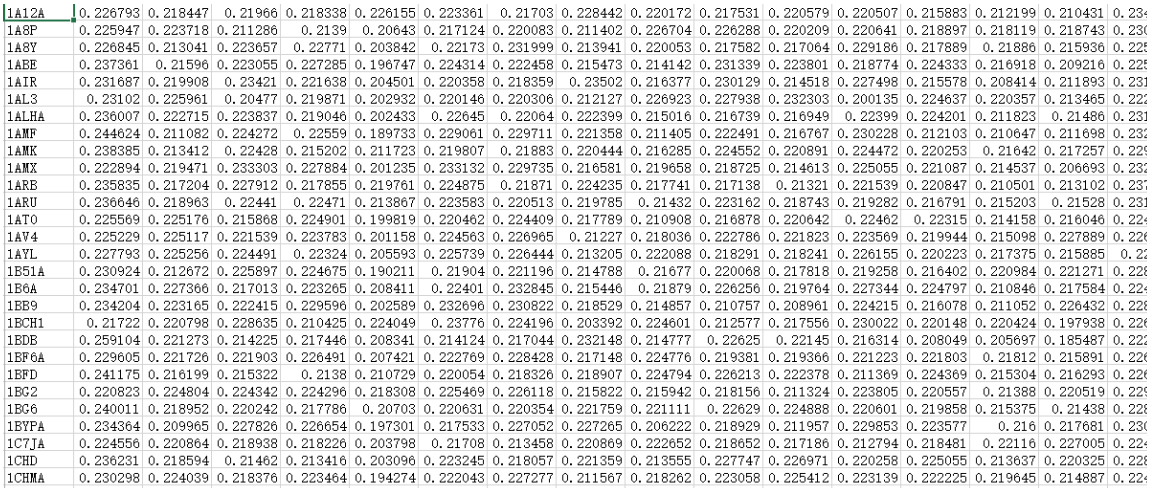
The local Pse-PSSM or the last sub-matrix ($P^{n}_{normalized}$) is given by

$$FV(n) = \left(F_1(n), \ldots, F_{20}(n), \Phi_1^1(n), \ldots, \Phi_{20}^1(n), \ldots, \Phi_1^{\lambda}(n), \ldots, \Phi_{20}^{\lambda}(n)\right) \tag{9}$$

where $F_j(n)$ and $\phi_j^1(k)$ are computed as described for the first $(n-1)$ sub-matrices. The final feature vector combines the feature vectors $FV(n-1)$ and $FV(n)$ to give

$$FV = (FV(n-1), FV(n)) \tag{10}$$

Here, we selected the best-performing parameters ($\lambda = 1$ and $n = 3$) as the default parameters. The protein sequence is finally represented as a 120D feature vector.



1A12A	0.226793	0.218447	0.21966	0.218338	0.226155	0.223361	0.21703	0.228442	0.220172	0.217531	0.220579	0.220507	0.215883	0.212199	0.210431	0.23
1A8P	0.225947	0.223718	0.211286	0.2139	0.20643	0.217124	0.220083	0.211402	0.226704	0.226288	0.220209	0.220641	0.218897	0.218119	0.218743	0.23
1A8Y	0.226845	0.213041	0.223657	0.22771	0.203842	0.22173	0.231999	0.213941	0.220053	0.217582	0.217064	0.229186	0.217889	0.21886	0.215936	0.22
1ABE	0.237361	0.21596	0.223055	0.227285	0.196747	0.224314	0.222458	0.215473	0.214142	0.231339	0.223801	0.218774	0.224333	0.216918	0.209216	0.22
1AIR	0.231687	0.219908	0.23421	0.221638	0.204501	0.220358	0.218359	0.23502	0.216377	0.230129	0.214518	0.227498	0.215578	0.208414	0.211893	0.231
1AL3	0.23102	0.225961	0.20477	0.219871	0.202932	0.220146	0.220306	0.212127	0.226923	0.227938	0.232303	0.200135	0.224637	0.220357	0.213465	0.22
1ALHA	0.236007	0.222715	0.223837	0.219046	0.202433	0.22645	0.22064	0.222399	0.215016	0.216739	0.216949	0.22399	0.224201	0.211823	0.21486	0.231
1AMF	0.244624	0.211082	0.224272	0.22559	0.189733	0.229061	0.229711	0.221358	0.211405	0.222491	0.216767	0.230228	0.212103	0.210647	0.211698	0.23
1AMK	0.238385	0.213412	0.22428	0.215202	0.211723	0.219807	0.21883	0.220444	0.216285	0.224552	0.220891	0.224472	0.220253	0.21642	0.217257	0.22
1AMX	0.222894	0.219471	0.233303	0.227884	0.201235	0.233132	0.229735	0.216581	0.219658	0.218725	0.214613	0.225055	0.221087	0.214537	0.206693	0.23
1ARB	0.235835	0.217204	0.227912	0.217855	0.219761	0.224875	0.21871	0.224235	0.217741	0.217138	0.21321	0.221539	0.220847	0.210501	0.213102	0.23
1ARU	0.236646	0.218963	0.22441	0.22471	0.213867	0.223583	0.220513	0.219785	0.21432	0.223162	0.218743	0.219282	0.216791	0.215203	0.21528	0.231
1ATO	0.225569	0.225176	0.215868	0.224901	0.199819	0.220462	0.224409	0.217789	0.210908	0.216878	0.220642	0.22462	0.22315	0.214158	0.216046	0.22
1AV4	0.225229	0.225117	0.221539	0.223783	0.201158	0.224563	0.226965	0.21227	0.218036	0.222786	0.221823	0.223569	0.219944	0.215098	0.227889	0.22
1AYL	0.227793	0.225256	0.224491	0.22324	0.205593	0.225739	0.226444	0.213205	0.222088	0.218291	0.218241	0.226155	0.220223	0.217375	0.215885	0.2
1B51A	0.230924	0.212672	0.225897	0.224675	0.190211	0.21904	0.221196	0.214788	0.21677	0.220068	0.217818	0.219258	0.216402	0.220984	0.221271	0.22
1B6A	0.234701	0.227366	0.217013	0.223265	0.208411	0.22401	0.232845	0.215446	0.21879	0.226256	0.219764	0.227344	0.224797	0.210846	0.217584	0.22
1BB9	0.234204	0.223165	0.222415	0.229596	0.202589	0.232696	0.230822	0.218529	0.214857	0.210757	0.208961	0.224215	0.216078	0.211052	0.226432	0.22
1BCH1	0.21722	0.220798	0.228635	0.210425	0.224049	0.23776	0.224196	0.203392	0.224601	0.212577	0.217556	0.230022	0.220148	0.220424	0.197938	0.22
1BDB	0.259104	0.221273	0.214225	0.217446	0.208341	0.214124	0.217044	0.232148	0.214777	0.22625	0.22145	0.216314	0.208049	0.205697	0.185487	0.22
1BF6A	0.229605	0.221726	0.221903	0.226491	0.207421	0.222769	0.228428	0.217148	0.224776	0.219381	0.219366	0.221223	0.221803	0.21812	0.215891	0.22
1BFD	0.241175	0.216199	0.215322	0.2138	0.210729	0.220054	0.218326	0.218907	0.224794	0.226213	0.222378	0.211369	0.224369	0.215304	0.216293	0.22
1BG2	0.220823	0.224804	0.224342	0.224296	0.218308	0.225469	0.226118	0.215822	0.215942	0.218156	0.211324	0.223805	0.220557	0.21388	0.220519	0.22
1BG6	0.240011	0.218952	0.220242	0.217786	0.20703	0.220631	0.220354	0.221759	0.221111	0.22629	0.224888	0.220601	0.219858	0.215375	0.21438	0.22
1BYPA	0.234364	0.209965	0.227826	0.226654	0.197301	0.217533	0.227052	0.227265	0.206222	0.218929	0.211957	0.229853	0.223577	0.216	0.217681	0.23
1C7JA	0.224556	0.220864	0.218938	0.218226	0.203798	0.21708	0.213458	0.220869	0.222652	0.218652	0.217186	0.212794	0.218481	0.22116	0.227005	0.22
1CHD	0.236231	0.218594	0.21462	0.213416	0.203096	0.223245	0.218057	0.221359	0.213555	0.227747	0.226971	0.220258	0.225055	0.213637	0.220325	0.22
1CHMA	0.230298	0.224039	0.218376	0.223464	0.194274	0.222043	0.227277	0.211567	0.218262	0.223058	0.225412	0.223139	0.222225	0.219645	0.214887	0.22

2-3 local-DPP feature

2.4 Feature 4 PSSM-DT

The feature 4 is called PSSM-DT according to the work of Zhou[2]. PSSM-DT is the Position-Specific Scoring Matrix Distance Transformation. It mainly includes two kinds of variables, which are the transformation variables of the same kind of amino acid distance, respectively Same Amino Acid Distance Transformation, SDT and Transformation variables of the same amino acid distance, different Property Distance Transformation DDT.

The variable SDT is used to calculate the average probability of any amino acid with the same amino acid content at two locations in different sequences. It can be calculated by the following formula,

$$SDT(i,lg)=\sum_{j=1}^{L-lg} S_{i,j}\times S_{i,j+lg}\,/\,(L-lg) \tag{11}$$

Here, i is a kind of protein and L is the length of amino acid and S is the score value of the position of i,j.

The variable DDT is used to calculate the average probability of the occurrence of two amino acids and amino acids in different locations at a distance of LG, and the formula is as follows:

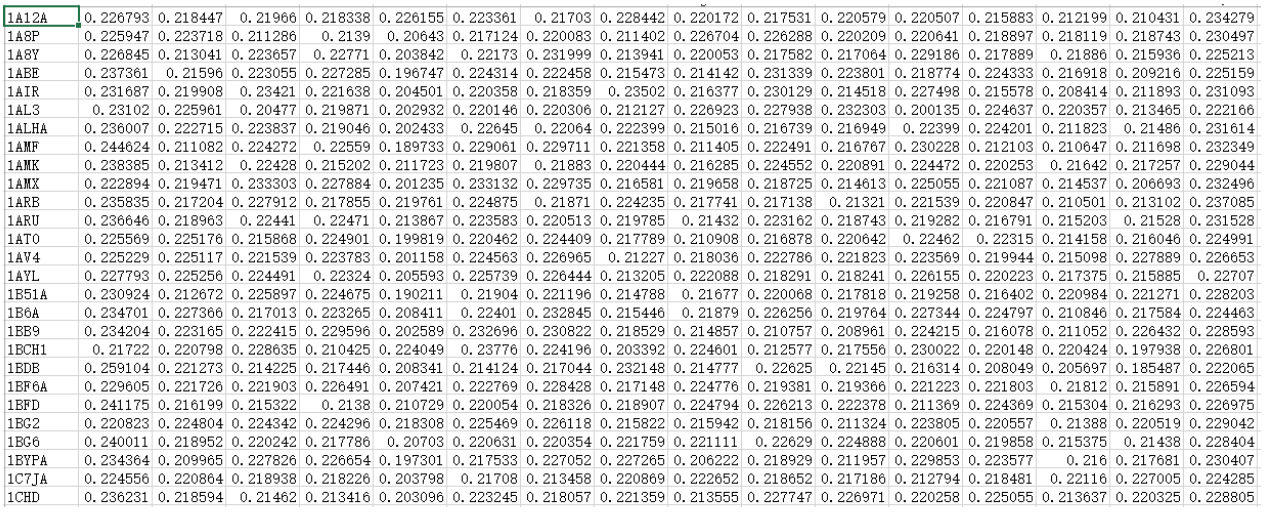
$$DDT(i1,i2,lg)=\sum_{j=1}^{L-lg} S_{i1,j}\times S_{i2,j+lg}\,/\,(L-lg) \tag{12}$$

Here, i1,i2 is two different kinds of amino acid.

The SDT can get 20*LG dimensions features and DDT is 380*LG. And according to the work of Zhou, we can set LG=5 to get the best result. So, we can get 2000D features.

2.5 Final Features

Finally, we summary the above four kinds of features and get the 2150D features.



1A12A	0.226793	0.218447	0.21966	0.218338	0.226155	0.223361	0.21703	0.228442	0.220172	0.217531	0.220579	0.220507	0.215883	0.212199	0.210431	0.234279
1A8P	0.225947	0.223718	0.211286	0.2139	0.20643	0.217124	0.220083	0.211402	0.226704	0.226288	0.220209	0.220641	0.218897	0.218119	0.218743	0.230497
1A8Y	0.226845	0.213041	0.223657	0.22771	0.203842	0.22173	0.231999	0.213941	0.220053	0.217582	0.217064	0.229186	0.217889	0.21886	0.215936	0.225213
1ABE	0.237361	0.21596	0.223055	0.227285	0.196747	0.224314	0.222458	0.215473	0.214142	0.231339	0.223801	0.218774	0.224333	0.216918	0.209216	0.225159
1AIR	0.231687	0.219908	0.23421	0.221638	0.204501	0.220358	0.218359	0.23502	0.216377	0.230129	0.214518	0.227498	0.215578	0.208414	0.211893	0.231093
1AL3	0.23102	0.225961	0.20477	0.219871	0.202932	0.220146	0.220306	0.212127	0.226923	0.227938	0.232303	0.200135	0.224637	0.220357	0.213465	0.222166
1ALHA	0.236007	0.222715	0.223837	0.219046	0.202433	0.22645	0.22064	0.222399	0.215016	0.216739	0.216949	0.22399	0.224201	0.211823	0.21486	0.231614
1AMF	0.244624	0.211082	0.224272	0.22559	0.189733	0.229061	0.229711	0.221358	0.211405	0.222491	0.216767	0.230228	0.212103	0.210647	0.211698	0.232349
1AMK	0.238385	0.213412	0.22428	0.215202	0.211723	0.219807	0.21883	0.220444	0.216285	0.224552	0.220891	0.224472	0.220253	0.21642	0.217257	0.229044
1AMX	0.222894	0.219471	0.233303	0.227884	0.201235	0.233132	0.229735	0.216581	0.219658	0.218725	0.214613	0.225055	0.221087	0.214537	0.206693	0.232496
1ARB	0.235835	0.217204	0.227912	0.217855	0.219761	0.224875	0.21871	0.224235	0.217741	0.217138	0.21321	0.221539	0.220847	0.210501	0.213102	0.237085
1ARU	0.236646	0.218963	0.22441	0.22471	0.213867	0.223583	0.220513	0.219785	0.21432	0.223162	0.218743	0.219282	0.216791	0.215203	0.21528	0.231528
1AT0	0.225569	0.225176	0.215868	0.224901	0.199819	0.220462	0.224409	0.217789	0.210908	0.216878	0.220642	0.22462	0.22315	0.214158	0.216046	0.224991
1AV4	0.225229	0.225117	0.221539	0.223783	0.201158	0.224563	0.226965	0.21227	0.218036	0.222786	0.221823	0.223569	0.219944	0.215098	0.227889	0.226653
1AYL	0.227793	0.225256	0.224491	0.22324	0.205593	0.225739	0.226444	0.213205	0.222088	0.218291	0.218241	0.226155	0.220223	0.217375	0.215885	0.22707
1B51A	0.230924	0.212672	0.225897	0.224675	0.190211	0.21904	0.221196	0.214788	0.21677	0.220068	0.217818	0.219258	0.216402	0.220984	0.221271	0.228203
1B6A	0.234701	0.227366	0.217013	0.223265	0.208411	0.22401	0.232845	0.215446	0.21879	0.226256	0.219764	0.227344	0.224797	0.210846	0.217584	0.224463
1BB9	0.234204	0.223165	0.222415	0.229596	0.202589	0.232696	0.230822	0.218529	0.214857	0.210757	0.208961	0.224215	0.216078	0.211052	0.226432	0.228593
1BCH1	0.21722	0.220798	0.228635	0.210425	0.224049	0.23776	0.224196	0.203392	0.224601	0.212577	0.217556	0.230022	0.220148	0.220424	0.197938	0.226801
1BDB	0.259104	0.221273	0.214225	0.217446	0.208341	0.214124	0.217044	0.232148	0.214777	0.22625	0.22145	0.216314	0.208049	0.205697	0.185487	0.222065
1BF6A	0.229605	0.221726	0.221903	0.226491	0.207421	0.222769	0.228428	0.217148	0.224776	0.219381	0.219366	0.221223	0.221803	0.21812	0.215891	0.226594
1BFD	0.241175	0.216199	0.215322	0.2138	0.210729	0.220054	0.218326	0.218907	0.224794	0.226213	0.222378	0.211369	0.224369	0.215304	0.216293	0.226975
1BG2	0.220823	0.224804	0.224342	0.224296	0.218308	0.225469	0.226118	0.215822	0.215942	0.218156	0.211324	0.223805	0.220557	0.21388	0.220519	0.229042
1BG6	0.240011	0.218952	0.220242	0.217786	0.20703	0.220631	0.220354	0.221759	0.221111	0.22629	0.224888	0.220601	0.219858	0.215375	0.21438	0.228404
1BYPA	0.234364	0.209965	0.227826	0.226654	0.197301	0.217533	0.227052	0.227265	0.206222	0.218929	0.211957	0.229853	0.223577	0.216	0.217681	0.230407
1C7JA	0.224556	0.220864	0.218938	0.218226	0.203798	0.21708	0.213458	0.220869	0.222652	0.218652	0.217186	0.212794	0.218481	0.22116	0.227005	0.224285
1CHD	0.236231	0.218594	0.21462	0.213416	0.203096	0.223245	0.218057	0.221359	0.213555	0.227747	0.226971	0.220258	0.225055	0.213637	0.220325	0.228805

2-4.The final 1250D features.

3. Classification Models

3.1 Logistic regression

Logistic regression was developed by statistician David Cox in 1958. The binary logistic model is used to estimate the probability of a binary response based on one or more predictor (or independent) variables (features). It allows one to say that the presence of a risk factor increases the odds of a given outcome by a specific factor.

In statistics, logistic regression, or logit regression, or logit model is a regression model where the dependent variable (DV) is categorical. The case of a binary dependent variable—that is, where the output can take only two values, "0" and "1", which represent outcomes such as pass/fail, win/lose, alive/dead or healthy/sick.

The case of DNA-binding protein prediction exactly is binary Classification. The Logistic regression should be valid.

3.2 Support vector machines

In machine learning, support vector machines (SVM) are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier (although methods such as Platt scaling exist to use SVM in a probabilistic classification setting). An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall. In addition to performing linear classification, SVMs can efficiently perform a non-linear classification using what is called the kernel trick, implicitly mapping their inputs into high-dimensional feature spaces.

3.3 Gradient boosting

Gradient boosting is a machine learning technique for regression and classification problems, which produces a prediction model in the form of an ensemble of weak prediction models, typically decision trees. It builds the model in a stage-wise fashion like other boosting methods do, and it generalizes them by allowing optimization of an arbitrary differentiable loss function.

Like other boosting methods, gradient boosting combines weak "learners" into a single strong learner in an iterative fashion. It is easiest to explain in the least-squares regression setting, where the goal is to "teach" a model F to predict values in the form $\hat{y} = F(x)$ by minimizing the mean squared error $(\hat{y} - y)^2$, averaged over some training set of actual values of the output variable y .

3.4 Xgboost

.Xgboost initially started as a research project by Tianqi Chen as part of the Distributed (Deep) Machine Learning Community (DMLC) group. Initially, it began as a terminal application which could be configured using a libsvm configuration file. After winning the Higgs Machine Learning Challenge, it became well known in the ML competition circles. This brought the library to more developers and became popular among the Kaggle community where it has been used for a large number of competitions.

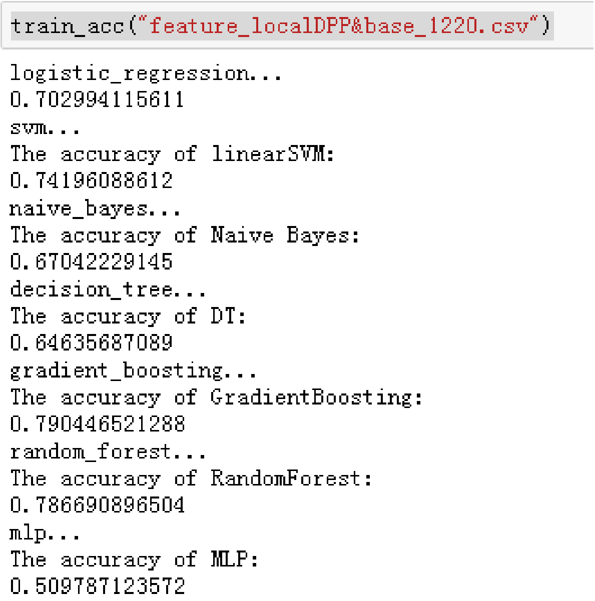
3.5 Random forests

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks, that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Random decision forests correct for decision trees' habit of overfitting to their training set.

4. Experiment Result

First, we get four different features one step by and step and test accuracy using different classification models by 10-fold-crossvalidation. 1) the frequencies of 20 kinds of amino acid of protein. It is 20D. 2) the best 10 of the top-2-gram. It is 10D. 3) local-DPP feature. It is about 120D. 4) PSSM-DT feature. It is 2000D. So, we extract 2150D effective features.

1. Using feature 1 and feature 3. (130D)

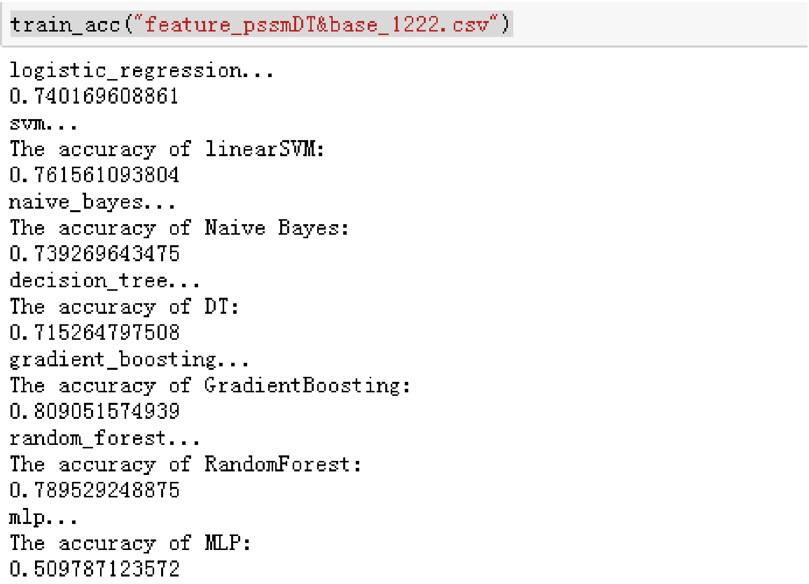


```
train_acc("feature_localDPP&base_1220.csv")
```

```
logistic_regression...
0.702994115611
svm...
The accuracy of linearSVM:
0.74196088612
naive_bayes...
The accuracy of Naive Bayes:
0.67042229145
decision_tree...
The accuracy of DT:
0.64635687089
gradient_boosting...
The accuracy of GradientBoosting:
0.790446521288
random_forest...
The accuracy of RandomForest:
0.786690896504
mlp...
The accuracy of MLP:
0.509787123572
```

The best accuracy is about 79.0% using gradient boosting.

2. Using feature 1 and feature 4.(2010D)

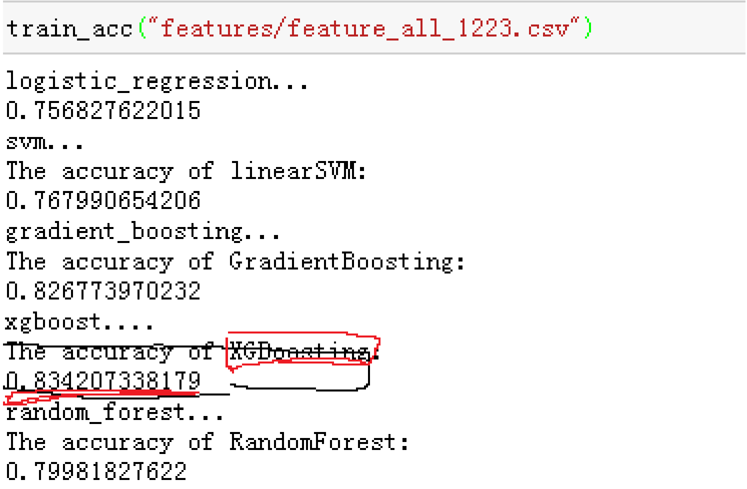


```
train_acc("feature_pssmDT&base_1222.csv")
```

```
logistic_regression...
0.740169608861
svm...
The accuracy of linearSVM:
0.761561093804
naive_bayes...
The accuracy of Naive Bayes:
0.739269643475
decision_tree...
The accuracy of DT:
0.715264797508
gradient_boosting...
The accuracy of GradientBoosting:
0.809051574939
random_forest...
The accuracy of RandomForest:
0.789529248875
mlp...
The accuracy of MLP:
0.509787123572
```

The best accuracy is about 80.9% using gradient boosting.

3. Using feature 1,feature 2 and feature 3.(2130D)

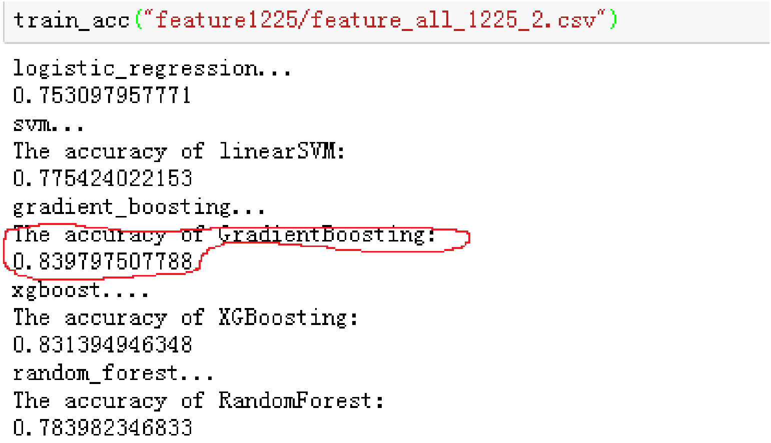


```
train_acc("features/feature_all_1223.csv")

logistic_regression...
0.756827622015
svm...
The accuracy of linearSVM:
0.767990654206
gradient_boosting...
The accuracy of GradientBoosting:
0.826773970232
xgboost....
The accuracy of XGBoosting:
0.834207338179
random_forest...
The accuracy of RandomForest:
0.79981827622
```

The best accuracy can reach 83.4% using xgboosting.

4. Using feature 1, feature 2, feature 3 and feature 4.(2150D)



```
train_acc("feature1225/feature_all_1225_2.csv")

logistic_regression...
0.753097957771
svm...
The accuracy of linearSVM:
0.775424022153
gradient_boosting...
The accuracy of GradientBoosting:
0.839797507788
xgboost....
The accuracy of XGBoosting:
0.831394946348
random_forest...
The accuracy of RandomForest:
0.783982346833
```

The best accuracy can reach 84.0% using gradient boosting.

As we can see, the result are getting better as the feature get more effective. And the best accuracy can reach 84.0% by the gradient boosting classification models.

We can see the 2150D features are effective and the result reach a high level.

5. References

[1]周继云. 基于序列信息的 DNA 结合蛋白质预测方法研究[D]. 哈尔滨工业大学, 2014.

[2] Wei L, Tang J, Zou Q. Local-DPP: An improved DNA-binding protein prediction method by exploring local evolutionary information[J]. Information Sciences, 2016, 384.