AI Project Report (CS410 2018 Autumn)

Name: Kaiyu Yan SNo: 516021910203 Teammate: Yufeng Deng

Abstract—This course project is about the classification problem of cells with different characteristics in the gene chip data. The classification problem is a very important part of machine learning. Through a lot of training, the machine sorts the input data into useful knowledge output and predicts the category from the result. And The classification includes binary classification and multivariate classification. This report proposed the Logistic Regression method and the Support Vector Machine method to train a classifier for binary and multivariate classification problem. And the model uesd principal component analysis to reduce the dimension of original data. In the other hand, the report tried to implement a deep learning model to do the same work as LR and SVM. By solving the same problem with different methods, we can understand the model more deeply and understand the difference between these methods.

Index Terms—Binary Classification, Multivariate Classification, Logistic Regression, Support Vector Machine, Deep Learning

1 Methods

In this course project we used a number of different methods to build the models. Each of these methods has its own characteristics, which we will introduce separately.

1.1 Principal Component Analysis (PCA)

When using statistical analysis methods to study multivariate problems, too many variables will increase the complexity of the problem. When there is a certain correlation between two variables, it can be explained that these two variables reflect a certain overlap of the information of this topic.

Principal component analysis is to delete all the variables that were originally proposed, and to eliminate redundant variables (close-knit variables), and to create as few new variables as possible, so that these new variables are irrelevant, and these new variables are reflected. Keep the original information as much as possible in the information aspect of the subject [1].

Some major mathematical basics are as follows:

1) Covariance between sample X and sample Y:

$$Cov(X,Y) = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{(n-1)}$$

When the samples are n-dimensional data, their covariance is actually a covariance matrix (symmetric matrix). For example, for 3-dimensional data (x, y, z), calculate its covariance is:

$$X = \begin{vmatrix} cov(x,x) & cov(x,y) & cov(x,z) \\ cov(y,x) & cov(y,y) & cov(y,z) \\ cov(z,x) & cov(z,y) & cov(z,z) \end{vmatrix}$$

2) If AX=λX, then is said to be the eigenvalue of A, and X is the corresponding eigenvector. When A is an n-th order invertible matrix, A is similar to $P^{-1}AP$, and the similarity matrix has the same eigenvalue.In particular, when A is a symmetric matrix, the singular value of A is equal to the eigenvalue of A, and there is an orthogonal matrix Q ($Q^{-1} = Q^T$) such that:

$$Q^TAQ = \begin{pmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_n \end{pmatrix}$$

Performing singular value decomposition on A can find all eigenvalues and Q matrices. AQ=QD, D is a diagonal matrix composed of eigenvalues, and the column vector of Q is the eigenvector of A.

If some of the eigenvalues are much larger than the others, we only need to focus on those eigenvalues that are particularly large, because they contain most of the information about the distribution of the data. Conversely, those eigenvalues close to 0 contain little information and can be ignored in the new feature space.

PCA usually has the following six steps [2]:

a) Remove the category feature of the data, and take the removed d-dimensional data as a sample.
b) Calculate the d-dimensional mean vector (that is, the mean of each dimension vector of all data).
c) Calculate the covariance matrix of all data.
d) Calculate eigenvalues (e_1, e_2, ..., e_d) and corresponding eigenvectors(λ_1, λ_2, ..., λ_d).
e) Sort the feature vectors in descending order according to the size of the feature values, and select the top k largest feature vectors to form a matrix W of d*k dimensions (each column represents a feature vector).
f) The sample data is transformed into a new subspace using the feature vector matrix W of d*K. Expression in mathematics is $y = W^T \times X$, Where X is a d*1 dimensional vector representing a sample and y is a K*1 dimensional vector in the new subspace.

When choosing the number of principal components, that is, the dimension K after dimension reduction, we use the following two concepts [1]:

• *Kaiyu Yan in the School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University, Shanghai, China*

Project Report Finished on Jan 13th, 2019.

average squared projection error (Where $x^{(i)}_{approx}$ is the mapped value):

$$\frac{1}{m}\sum_{i=0}^{m}\|x^{(i)}-x^{(i)}_{approx}\|^2$$

total variaton in the data:

$$\frac{1}{m}\sum_{i=0}^{m}\|x^{(i)}\|^2$$

Select different K values, and then use the following formula to continuously calculate and select the minimum K value that can satisfy the following formula conditions.

$$\frac{\frac{1}{m}\sum_{i=0}^{m}\|x^{(i)}-x^{(i)}_{approx}\|^2}{\frac{1}{m}\sum_{i=0}^{m}\|x^{(i)}\|^2}\leq t$$

The value of t can be set. For example, if the value of t is 0.01, it represents that the PCA algorithm retains 99% of the main information.

1.2 Logistic Regression (LR)

Logistic regression is a classification algorithm rather than a regression algorithm. It is common to use known independent variables to predict the value of a discrete dependent variable (like binary value 0/1, yes/no, true/false). Simply put, it is by fitting a logic function to predict the probability of an event occurring. So it predicts a probability value, naturally, its output value should be between 0 and 1.

General steps are as follows [3]:

1) Construct a prediction function $h(x)$. There is a function g called sigmoid function(The function image is shown in Figure 1):

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

For the case of a linear boundary, the boundary form is as follows(The sample data is vector x and the parameter is vector θ):

$$z=\theta^Tx=\theta_0x_0+\theta_1x_1+\cdots+\theta_nx_n=\sum_{i=0}^{n}\theta_ix_i$$

Then the prediction function is:

$$h_\theta(x)=g(\theta^Tx)=\frac{1}{1+e^{-\theta^Tx}}$$

For input x, the probability that the classification result is category 1 and category 0 is:

$$P(y=1|x;\theta)=h_\theta(x)$$

$$P(y=0|x;\theta)=1-h_\theta(x)$$

2) Construct a loss function $J(\theta)$ which is derivation based on maximum likelihood estimation and gradient descent method.

$$J(\theta)=\frac{1}{m}[\sum_{i=1}^{m}(y_ilogh_\theta(x_i)+(1-y_i)log(1-h_\theta(x_i)))]$$

Update θ: $\theta_j:=\theta_j-\frac{\alpha}{m}\sum_{i=1}^{m}(h_\theta(x_i)-y_i)x_i^j$

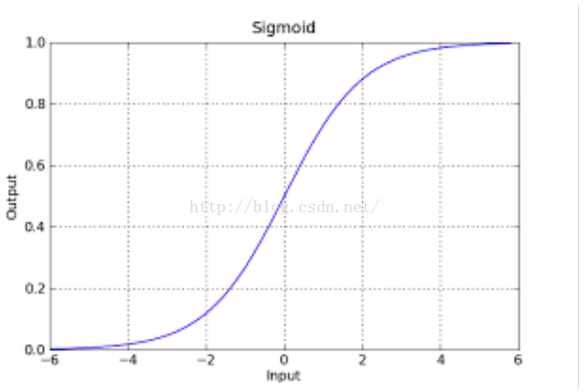

Fig. 1. The image of sigmoid function

Over-fitting is an over-fitting of training data, which increases the complexity of the model and poor generalization ability. The solution can be to remove some feature quantities, or regularization. Taking the squared loss in the regression problem, that is, the L2 norm of the parameter, the loss function of the model becomes(λ is a regular term coefficient):

$$J(\theta)=\frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x_i)-y_i)^2+\lambda\sum_{j=1}^{m}\theta_j^2$$

After regularization, the update of θ becomes:

$$\theta_j:=\theta_j-\frac{\alpha}{m}\sum_{i=1}^{m}(h_\theta(x_i)-y_i)x_i^j-\frac{\lambda}{m}\theta_j$$

1.3 Support Vector Machine (SVM)

Support vector machines are used for clustering data into two categories according to maximum boundary geometry. In the SVM training algorithm, new examples are assigned to one category or another as either nonlinear or linear binary classifiers obtained from a set of training examples.

Considering a linear two-class problem, there are two sample points x on a two-dimensional plane, and the target values are respectively labeled as $\{-1,1\}$, and an infinite number of straight lines $\omega^Tx+b=0$ can be made. We can use the sign of ω^Tx+b to determine the classification of points, and write the decision function $f(x,\omega,b)=sign(\omega^Tx+b)$ to separate the two types of points.

The expansion from a two-dimensional line $\omega^Tx+b=0$ to a high dimension is called a hyperplane (ω,b) (See Figure 2). The distance between a point and the hyperplane can indicate the degree of confidence in the classification prediction. The farther the sample point is from the hyperplane, the greater the confidence that the classification is correct. And if the classification is correct, the signs of $y^{(i)}$ and $\omega^Tx^{(i)}+b$ are the same.

The function interval of the hyperplane (ω,b) with respect to the sample points $(x^{(i)},y^{(i)})$ is [4]:

$$\hat{\gamma}^{(i)}=y^{(i)}(\omega^Tx^{(i)}+b)$$

The geometric interval of the hyperplane (ω,b) with respect to the sample points $(x^{(i)},y^{(i)})$ is:

$$\gamma^{(i)}=y^{(i)}(\frac{\omega^Tx^{(i)}+b}{\|\omega\|})$$

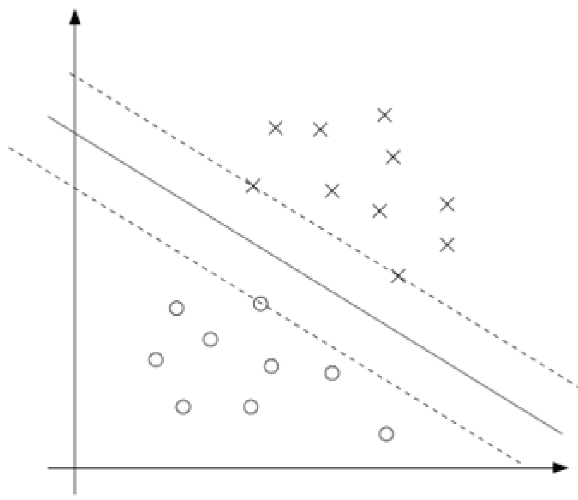
Fig. 2. The image of hyperplane

Define the hyperplane's geometric interval γ for the sample set S as the minimum of the hyperplane (ω, b) and the geometric interval of all sample points in S. The basic idea of a support vector machine is to solve a separate hyperplane that correctly partitions the training data set and has the largest geometric spacing. In order to maximize the interval, only need to maximize γ. A main constraint problem can be obtained by the relationship between function interval and geometric interval.

$$min_{\gamma,\omega,b} \quad \frac{1}{2}\|\omega\|^2$$
$$s.t. \quad y^{(i)}(\omega^T x^{(i)} + b) \geq 1, \quad i = 1, \cdots, m$$

With Lagrange duality and SMO algorithm, we can get the best classification hyperplane (α is the Lagrangian multiplier).

$$\begin{aligned}\omega^T x^{(i)} + b &= (\sum_{i=1}^{m} \alpha_i y^{(i)} x^{(i)})^T + b \\ &= \sum_{i=1}^{m} \alpha_i y^{(i)} \left\langle x^{(i)}, x \right\rangle + b\end{aligned}$$

But sometimes the data is not linearly separable. So we can set a relaxation factor $\xi_i \geq 0$ for each data point to make the function interval and relaxation factor satisfy the relationship $\gamma_i + \xi_i \geq 1$. And assign a price C to each relaxation factor.

$$min_{\gamma,\omega,b} \quad \frac{1}{2}\|\omega\|^2 + C\sum_{i=1}^{m}\xi_i$$
$$s.t. \quad y^{(i)}(\omega^T x^{(i)} + b) \geq 1 - \xi_i, \quad i = 1, \cdots, m$$
$$\xi_i \geq 0, \quad i = 1, \cdots, m$$

For nonlinear support vector machines, the original sample data is linearly inseparable, However, nonlinear transformation $\phi(x)$ is applied to the original data to map the original data from low-dimensional to high-dimensional, and the data on the high-dimensional may become linearly separable. To reduce the amount of computation on high dimensions, we can define kernel functions $K(x, z) = \phi(x)^T\phi(z)$.

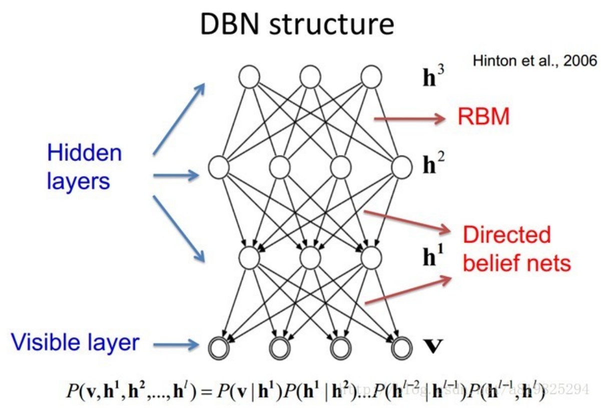

Fig. 3. The structure of the DBN

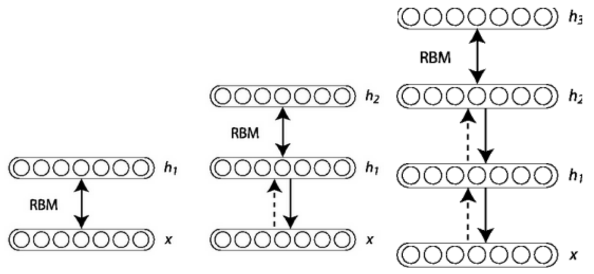

Fig. 4. Layer-by-layer learning parameters

1.4 Deep Belief Network (DBN)

In the deep learning part we used a model called deep belief network. DBN is a probability generation model. Compared with the neural network of the traditional discriminant model, the generation model is to establish a joint distribution between observation data and labels, and evaluate both P($Observation|Label$) and P($Label|Observation$). The discriminant model has only evaluated the latter, which is P ($Label|Observation$).

DBN consist of multiple restricted Boltzmann machines (RBM) layers. A typical network structure is shown in Figure 3. These networks are "restricted" to a visible layer and multiple hidden layers, with connections between the layers, but there are no connections between the cells within the layer. The hidden layer unit is trained to capture the correlation of higher order data represented in the visible layer [5].

A layer-by-layer unsupervised method is used to learn the parameters during training. As shown in Figure 4, the data vector x and the first layer hidden layer are first used as an RBM, and the parameters of the RBM are trained, and then the parameters of the RBM are fixed, h_1 is regarded as a visible vector, and h_2 is regarded as a hidden vector, and training is performed. The second RBM gets its parameters, then fixes these parameters and trains the RBMs composed of h_2 and h_3.

DBN is mainly divided into two steps in the process of training the model:

1) Each layer of RBM network is trained separately and unsupervised to ensure that feature information

is preserved as much as possible when mapping feature vectors to different feature spaces.

2) The BP network is set up in the last layer of the DBN, and the output feature vector of the RBM is received as its input feature vector, and the entity relationship classifier is supervisedly trained.
Each layer of RBM network can only ensure that the weights in its own layer are optimal for the feature vector mapping of the layer, instead of optimizing the feature vector mapping of the entire DBN. Therefore, the back propagation network propagates the error information from top to bottom to each layer of RBM and fine-tunes the DBN network.
The layer with supervised learning at the top can be replaced with any classifier model depending on the specific application domain, not necessarily the BP network. So in the project we use the logistic regression classifier.

2 Results

We will introduce the results of this project, using traditional classification methods and deep learning. In the traditional method, we first carry out dimensionality reduction and then carry out the training model, while in deep learning, we train layer by layer.

All the programs and output files of this project can be downloaded from my github.

2.1 PCA

In the Gene_Chip_Data dataset, file microarray.oroginal.txt is the data file, which saves 22283 rows and 5896 columns matrix data. Every row shows the feature and every column shows the observation. Dimensional processing is performed on the data.

We set the contribution ratio of PCA as 0.9 and we finally get 112 main features.

Another file E-TBM-185.sdrf.txt saves the labels for examples from $material$ to $disease\ status$, etc. Our experiment uses the $material$ labels as the binary classification standard and uses the $disease\ status$ labels as the multiclass classification standard. There are 2 different label, organism_part and cell_line under the $material$ and 194 kinds of disease status labels under $disease\ status$.

As for the testing, we use the $k-fold$ cross-validation method to test our models. The original data is divided into k groups and we make k-1 groups be the training set and 1 group left be the test set. Then change the training set and test set in turn for k times so that every group has the chance to be tested. Finally we will calculate the average accuracy of k times cross-validations.

2.2 LR

The knowledge of the basis of logistic regression has been mentioned above, so that we can write programs and fit the data. And its important to transfer the label to vectors. We used one-hot code to represent different kind of labels. For example, we could use [1, 0] to represent organism_part and use [0, 1] to represent cell_line. We trained the model for 100 iterations and every iteration has 100 batches. Figure 5 is the running result of the LR.

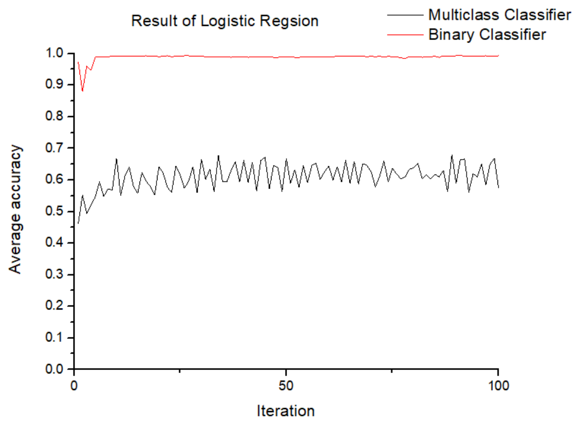

Fig. 5. The result of LR

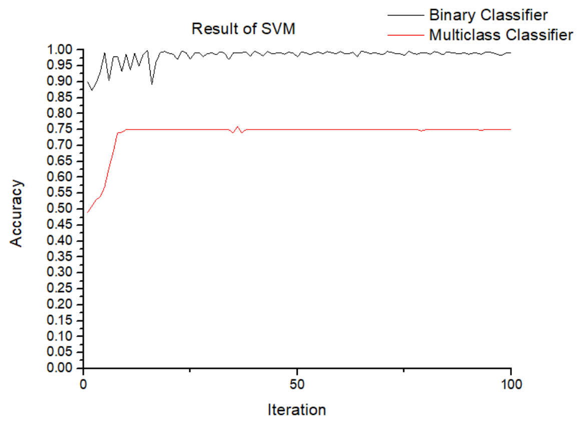

Fig. 6. The result of SVM

2.3 SVM

We use another method that is similar to one-hot code in LR, which uses -1 to replace 0 in one-hot code. For example, we use [1, -1] to represent organism_part and use [-1, 1] to represent cell_line. We trained the model for 100 iterations and every iteration has 100 batches. Figure 6 is the running result of the SVM.

2.4 DBN

After the DBN program outputs the result, we find that the accuracy of every epoch is almost the same. We think the reason is pre-training of deep learning. However, because the number of observations under every label is much too insufficient that it cant satisfy the standard of deep learning, it performs poor in classification problems. The accuracy of multiclass classification problem is only about 0.2 and the accuracy of binary classification problem is about 0.8, which is far away from LR and SVM. Figure 7 is the running result of the DBN.

3 Discussion

In order to test the performance of multiclass classifier and binary classifier, we conducted two kind of experiment for every methods. Multiclass test requires classifiers to tell all

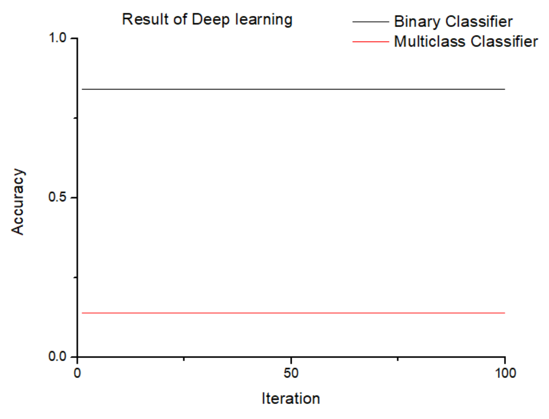

Fig. 7. The result of DBN

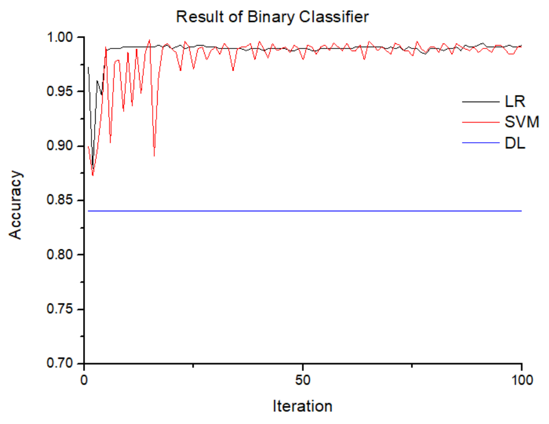

Fig. 8. The result of Binary classification

194 kinds of status, and binary class test only requires classifiers to tell whether the observations comes from organism or cell.

I will explain why we used the $material$ label as the standard of binary classification problem rather than use $disease status$. In the dataset, there are 4752 observations with organism_part label and 1142 observations with cell_line label. The ratio is about 4:1. We would have decided to use $normal$ and $disease$ as the standard. However, there are only 85 $normal$ observations and 4917 of all observations are $disease$, where the ratio is about 58:1. The incommensurable ratio will cause big error in statistics. So we decided to use $material$ label.

3.1 Comparison of results of each method

Figure 8 and Figure 9 are the Comprehensive comparison. We can draw the conclusion that LR performs better when there are fewer target labels while SVM performs better when there are more target labels, and DBN has advantages in convergence but it doesnt have high accuracy because of the restriction of data quantities.

3.2 LR

When we focus on Figure 5, we can find that LR model did well in binary classification problems and the accuracy

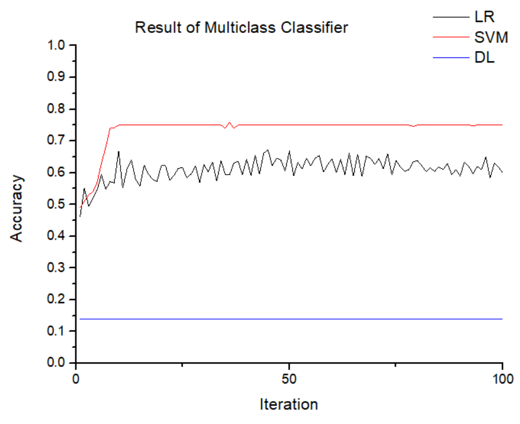

Fig. 9. The result of Multiclass classification

reached 0.96. It also had an excellent performance in convergence. However, it didnt have a high accuracy in multiclass classification problems. Although the overall tends to be stable, the fluctuation is large and the curve is not smooth.

3.3 SVM

When we focus on Figure 6, it is obvious that SVM models perfectly fitted multiclass classification problem that its convergence is excellent and can reach 0.99. At the same time, it also has good convergence in binary classification, reaching 75%. In general its undeniable that SVM models have higher accuracy than LR.

3.4 DBN

When we focus on Figure 7, we can find that the accuracy of every epoch is almost the same. We think the reason is pre-training of deep learning. However, because the number of observations under every label is much too insufficient that it cant satisfy the standard of deep learning, it performs poor in classification problems. In terms of binary classification, its accuracy is still ok, reaching 0.8, but it is poor in multi-classification, only 0.2, not as good as LR and SVM.

Acknowledgments

We sincerely appreciate this course for offering valuable techniques and broadening our horizons. Moreover, we would like to show our deepest gratitude to Prof. Bo Yuan and assistants.

My github is marked [6].

References

[1] [Online]. Available: https://www.cnblogs.com/steed/p/7454329.html
[2] [Online]. Available: https://www.cnblogs.com/charlotte77/p/5625984.html
[3] [Online]. Available: https://blog.csdn.net/chibangyuxun/article/details/53148005
[4] [Online]. Available: https://blog.csdn.net/luoshixian099/article/details/51073885
[5] [Online]. Available: https://blog.csdn.net/a819825294/article/details/53608141
[6] [Online]. Available: https://github.com/mangotuanzi/AI-and-Gene_Chip_Data-project