Digging in Logistic Regression Using binary and Multiclass Datasets

Xinji Wang 32026312

SCC 461

Lancaster University

January 14, 2018

Abstract

This study implements logistic regression from scratch on both binary and multiclass datasets. With a accuracy and other performance metrics higher than baseline, the report shows in detail about how a model was tuned and analyze the result from a statistical and computer science perspective. Though reducing the binary datasets and getting different number of attributes, the result of tuning the model shows a specific pattern. Only the number calculation (e.g. Number of epochs, iterations, observations and features) will affect the runtime but all other parameters will change the confusion matrix and the result of prediction. So the report detailed all the result of how the model was tuned and tried to explain how the model works. In the multiclass classification area, the model is adjusted to fit in the situation by involving a voting criteria and maximum confidence probability method. In the one-vs.-all LR, 100% accuracy was obtained and in the on-vs.-one LR reach 88%.

1 Introduction

1.1 Objectives

The aim of this report is an attempt at implementing logistic regression (2.1) on both binary datasets and multiclass datasets. When digging into the implementation of multiclass datasets, a one-Vs.-all logistic regression will be used section (2.2). An auxiliary line of the report focuses on presenting comparisons about how function parameters tuning, changing of the size of datasets and features affect the performance of the model and results will be analyzed in both machine learning and computer science perspective. All of the tunings of the model will be tune and record specific to different datasets and the outcome will be presented in a form of comparison using tables and graphs. From here, all models with or without pre-processed data will be compared using a pre-defined set of performance metrics section (2.3). This first section focuses on setting out the motivations and objectives of this project, along with a brief introduction to the dataset and any related work. Section 2 will focus on the underlying methodologies behind the algorithms used in this report, whilst Section 3 will provide a detailed breakdown of exactly how each model was built and tuned along with the results produced by each model. The final section of the report's main body will focus on the limitations of this project, giving out conclusions of the project and providing potential explanations.

The specific objectives of this project are to present the process of implementation of logistic regression from scratch, using Python as programming language without any toolkits and using R as an analytical tool, showing readers how this machine learning algorithm works for an arbitrary number of features in real datasets.

1.2 Dataset Introduction

This subsection will introduce the datasets used in the project; two of the datasets are binary label datasets whilst the third dataset is a multiclass datasets with 3 labels. All of the datasets can be download from the UCI machine learning repository.

Logistic regression and the task of binary classification have been extensively associated in the different area. In two of the binomial datasets, the project will focus on model construction, studying how parameters tuning like a change of the threshold, learning rate and the number of epochs, size of datasets change the results. And later in the multiclass classification problem, the task will focus on not only tuning of the model but also voting criteria and confidence of probability.

The Pima Indians Diabetes Dataset is a real dataset contributed by the National Institute of Diabetes and Digestive and Kidney Diseases. With 738 observations and 8 attributes in the dataset, the patients in record were divided into two groups. Class value '1' is interpreted as "tested positive for diabetes". With 500 of the observations are labeled '0' and 268 observations being labeled '1'.

The Sonar dataset is the dataset used by Gorman and Sejnowski in their study of the classification of sonar signals using a neural network. The dataset contains 60 patterns obtained from the sonar signals off a metal cylinder at various angles and under various conditions. The label associated with each record contains the letter "R" if the object is a rock and "M" if it is a mine (metal cylinder). The numbers in the labels are in increasing order of aspect angle, but they do not encode the angle directly.

The Wine dataset are the results of a chemical analysis of wines grown in the same region in Italy but derived from three different cultivars. The analysis determined the quantities of 13 constituents found in each of the three types of wines. The three types of the label represent for three kinds of wine with each class has around 60 observations.

1.3 relate works

The word 'logit' was first presented by Bliss(1934) as a comparison to the word 'probit'. As an alternative and better choice of Gaussian distribution, logistic regression has a wilder usage. Adam(1958) used logistic regression on economics demand analysis and Theodossiou(1998) analysis unemployment effect on psychological well-being research. On facing a medical research which involves binary classification prediction, a logistic regression performs well and therefore is wildly use in medical research, Bender (1997) and Bestall (1999). When focus on feature selection, LR model can take the role as a tool to reduce dimension of dataset, Hughes(1992) ,Cooper (2000).

2 Methods

2.1 Logistic regression on binary classification

To solve a binary classification problem, a most obvious idea is to let $p(y=1 \mid X=x)$ be the output of the model, calculating the probability of how possible an observation is predicted to be the class label '1'. With a given dataset, each attribute contributes certain weights to the probability, so finally after estimated the weights using certain statistic methods, and constraining the output between 0 and 1, a logistic regression can give the maximum likelihood probability of a certain observation classified as a certain class.

A logistic regression model is like a linear regression model, in order to constrain our output between 0 and 1, a sigmoid function is used, which is defined by the following equation:

$$h_\theta(x) = \frac{1}{1+e^{-\theta^T x}} \qquad (2\text{-}1)$$

The task of the logistic regression model is to find best-fitted hypothesis function ($h_\theta(x)$) of the dataset so that a probability of an object can be predicted. And the corresponding function image is presented in the graph bellow:

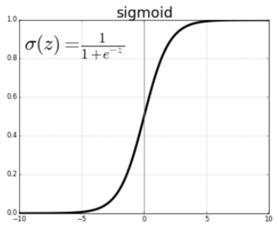


Figure (2-1)

All the output after the function is between 0 and 1, the threshold of the function is usually set to be 0.5 as it is the value of $h_\theta(0)$ while $g(x) = w_0 + w_1x_1 + ... + w_nx_n = 0$. If the outcome of the function $h_\theta(x) > 0.5$, we class the object to be class '1', else the object is assert to be class '0'. This lead to the next step of the model, which is to calculate the weighs of each attributes.

Using the theory in maximum likelihood, assuming m observations are given and the corresponding label $y_1, y_2, y_3 \ldots . y_m$, denote p_r be the probability of an object being labeled '1'($y_i = 1$), so the probability of the object being labeled '0'($y_i = 0$) is $1 - p_r$. The odds ratio can be present as equation (2-2):

$$\frac{p(y=1|x)}{p(y=0|x)} = \frac{p}{1-p} = e^{g(x)} \qquad (2\text{-}2)$$

While after logarithm transformation, the equation is:

$$\ln\left(\frac{p}{1-p}\right) = h_\theta(x) = w_0 + w_1x_1 + w_2x_2 + \cdots + w_mx_m \qquad (2\text{-}3)$$

And to evaluate the parameters in $g(x)$, following the maximum likelihood theory, the joint likelihood function can be present as follow:

$$L(\omega) = \prod_{i=1}^{m}(h_\theta(x_i)^{y_i})(1 - h_\theta(x_i)^{1-y_i}) \qquad (2\text{-}4)$$

Also, after a logarithm transformation on equation (2-4), the log-likelihood function is:

$$\iota(\omega) = \sum_{i=1}^{m}\left(y_i * \log(h_\theta(x_i))\right) + (1 - y_i) * \log(1 - h_\theta(x_i)) \qquad (2\text{-}5)$$

So, maximize equation (2-5), which means get the best estimate of parameter θ, we use the stochastic gradient descent method. For attribute x_j, the corresponding weight θ_j is renewed following equation (2-6), while y^i is the label of the i_{th} observation, and $h_\theta(x^i)$ is the predicted probability of the i_{th} item in the dataset.

$$\theta_i := \theta_i - \alpha * (h_\theta(x_i) - y_i) * h_\theta(x_i) * (1 - h_\theta(x_i) * x_i) \qquad (2\text{-}6)$$

And:

$$\theta_0 := \theta_i - \alpha * (h_\theta(x_i) - y_i) * h_\theta(x_i) * (1 - h_\theta(x_i)) \qquad (2\text{-}7)$$

After each calculation, the weights θ_j are renewed once immediately so that the cost will converge faster compare to the gradient descent method.

After a number of epochs and iterations, the weights will converge after training and can be used to predict new objects or test the model performance.

2.2 One vs. all logistic regression

A one vs. all logistic regression is an adjustment of binary logistic regression so that it can be used while facing multiclass classification problem. In this kind of regression model, by subtracting one of the classes from an n-class dataset and turn them into positive examples while the rest of the class are turned into negative class, binary logistic regression was trained. After the first model was built, the repeat the process until the model of all the class was trained. Therefore, with an n-class dataset, n models were built and for each object, n results were given.

$$h(i)\theta(x)=P(y=i|x;\theta),i=1,2,3\ldots n \tag{2-8}$$

Having calculated the vector $h_\theta(x) = [h_\theta^{(1)}, h_\theta^{(2)}, h_\theta^{(3)} \ldots h_\theta^{(n)}]$, we choose the maximum value $h_\theta^{(k)}$ and label the object ‘k’. The implementation of this model will also be realized using python and without toolkits or library.

For a 4-class dataset in picture (2-2), the diamond class in the top-left corner is separated and turn into positive while the rest of the data graphics are classed negative. Therefore the first model can be constructed. After all four model were built, a new object will be label to the class with highest probability among the four model

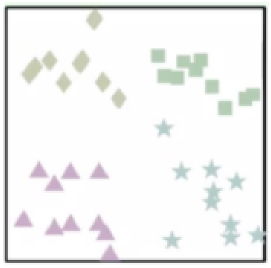
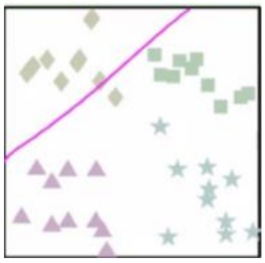


Figure (2-2)



Figure(2-3)

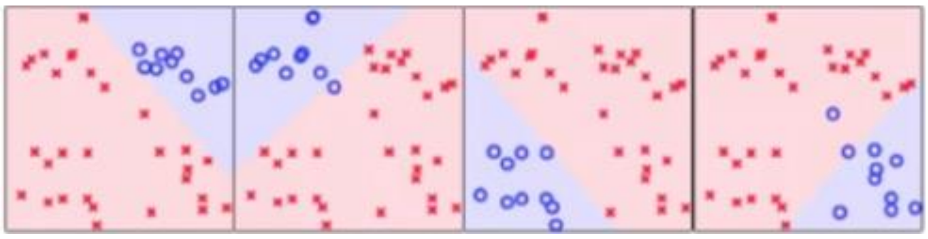


Figure (2-4)

2.3 One vs. one logistic regression[1]

A one vs. one logistic regression is another kind of adjustment of binary logistic regression while facing multiclass classification tasks. And the procedure is slightly different from the one vs. all

[1]Figure(2-2,3,4) 2015,Source: http://kubicode.me/2015/08/30/Machine%20Learning/Multiclass-Classification/

logistic regression. By traverse all the class in the dataset, in each construction of the model, two classed were picked and a corresponding model will be trained. After C_N^2 of the models were built, a voting strategy was brought in to assess the final result. For an object to be predicted, the object will be label to the class if it gets the most votes, that is to say it will be class as the class which win most of the time against other class.

2.4 Model Evaluation

During model evaluation, K-fold cross validation works by first splitting the dataset into k, equally sized partitions. Of these partitions, one is randomly selected and used as a validation or testing set and the remaining k-1 samples are then used for training the model. The model is then trained; using the k-1 partitions and predictions are then made by passing the testing data through the trained model. This process is then repeated with every one of the k partitions of the data used as the testing set. The results of each test are then averaged at the end to determine a final model accuracy and standard errors can be found be calculating $\frac{1}{\sqrt{n}}\sum x_i$ where x_i is the accuracy at each stage of cross-validation and n is the number of observations. A 10-fold cross validation method has been shown been shown y Kohavi et al. (1995) to be most effective when assessing a models accuracy and for this reason, it will be the initial metric of choice when assessing each model's performance on both the training and testing data. But deal with time-consuming reason, a 5-fold cross validation is used in this project.

Besides accuracy, other machine learning performance metrics will be bringing in use. A precision is a positive predict value, reflecting the percentage of true positive among those predicted positive by the model. A recall is a metric that evaluates the probability of true positive prediction among the entire positive object in the dataset.

The formulas are present as follows, where TP are the true positive object and FP are the false positive object, FN are abbreviation for false negative:

$$\text{Precision} = \frac{TP}{TP+FP} \tag{2-9}$$

$$\text{Recall} = \frac{TP}{TP+FN} \tag{2-10}$$

$$\text{F1} - \text{score} = 2 * \frac{Precision*Recall}{Precision+Recall} \tag{2-11}$$

3 Results

3.1 pre-processing

Four datasets are trained and test on the LR model, two original dataset Pima-Indian-diabetes and sonar dataset with two corresponding PCA-processed dataset, as can be seen in the file *pca_sonar.csv and pca_pima.csv.*

From a computer science perspective, time-consuming problem while the size of a dataset changed is concerned. By processing Principle component analysis (PCA) in two binary dataset Pimi-Indian-diabetes and Sonar dataset, we resize the dataset from 738*8 to 738*2 and from 208*60 to 208*7, the size of sonar dataset drop from 84 kb to only 22 kb. Meanwhile the time consuming on the same number of epochs drop. See in figure (3-1) bellow.

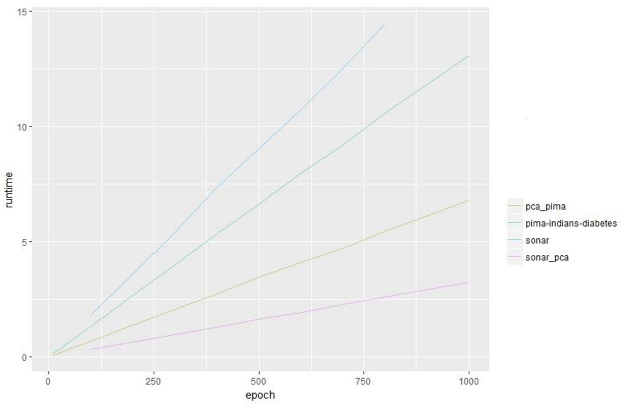


Figure (3-1)

Keeping other parameters unchanged, the graph shows a clear linear relationship. In minimizing the time consume while training a logistic regression model, a pre-processed dataset or a dimension reduced dataset takes less time in training, this can be seen in a vertical comparison that the biggest dataset sonar takes longest time in training and the smallest dataset sonar-pca takes the shortest time in training. And as epochs, data size (number of attributes or number of observation) increase, the time consuming will increase even if a stochastic gradient descent method are used to reduce training time, not to say using gradient descent method or a batch gradient decent. When facing a massive data, time consuming problem should take into considered.

To prove that size of feature statistically significant effect the runtime, a two sample t-test is used to exam the result (code can be found in appendix). Running the model 20 times in each binary dataset, and finally get the result that the runtime of Pima. Dataset and the Pca_pima dataset differ, as shown in the table below. Same result happened on the sonar and correspond dimension reduced dataset which will not be presented here.

Table 3-1

Two Sample t-test				
t	df	p-value	CI_lower	CI_upper
72.048	37.999	2.20E-16	3.515982	3.719278

3.2 binary logistic regression construction

Exact code for architecture of the logistic regression can be found in the logistic.py file, raw record of tuning can be found in the 'result' document.

Considering only using stochastic gradient descent method to renew weights, the adjustment of parameters are more like a thumb of rule. The Gaussian logit response curve is then just a convenient starting point in the process of model building Ter Braak et.al.(1986). By initializing weights obeying the Gaussian (normal) distribution, tuning the model by changing learning rate, number of epochs, number of folds, all the results are recorded and can be found in the *result.xlsx file.* Although a full breakdown can be found in Table 4-1, optimal results were achieved by the original dataset of Pima-Indian-Diabetes dataset using a 0.15 learning rate and 300 epochs. Note that all the results of the model are close because epochs are carried out to provide sufficient propagation to set the weights correctly.

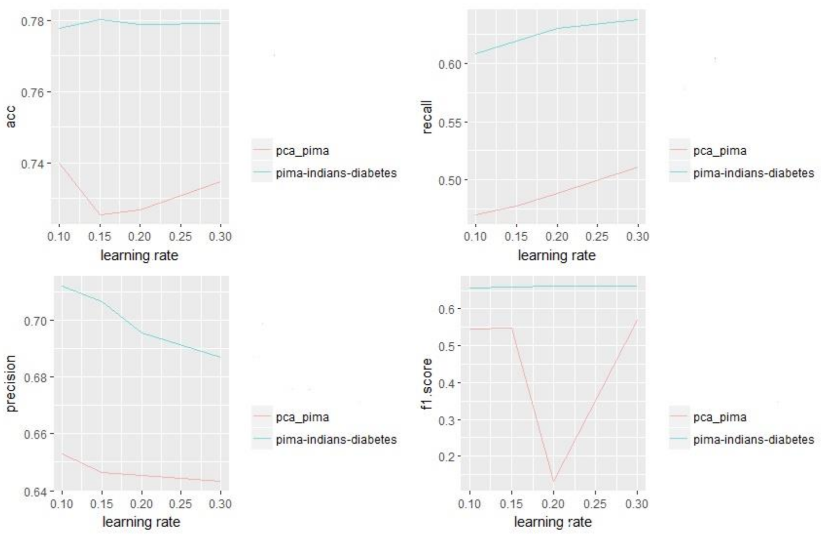


Figure (3-2) result of 'pima' dataset under different learning rate

Figure (3-2) shows a clear difference that the original data are better than the data after PCA process in all performance metrics, which means PCA pre-process lost some of the information of the raw dataset. Same things happened in the Sonar data and so a lower performance is obtained as showed in Figure (3-3).

According to result table (4-1), the predict model dealing with diabetes dataset parameter was set under a 0.15 learning rate and number of epochs = 300 due to its highest accuracy. And the sonar-predict model parameter was set under a 0.25 learning rate and 400 epochs.

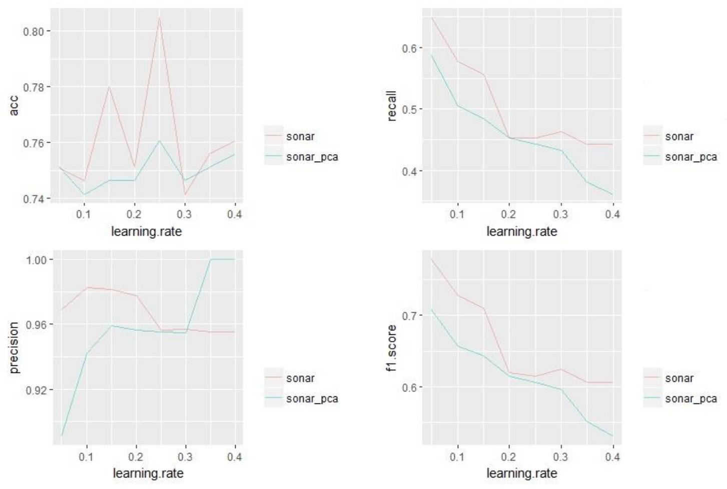


Figure (3-3) result of 'sonar' dataset under different learning rate

Focusing in the top-left figure in figure(3-4), the accuracy rises at first and drops later, that means be precisely adjusting the value of threshold one can possibly find the optimum value for a predict model. The value of recall and precision are more like a trade of while one raises the other drops. Finally, the f1-score are a balanced metric of precision and recall which can be seen from not only the trend of figure but also from equation (2-11).

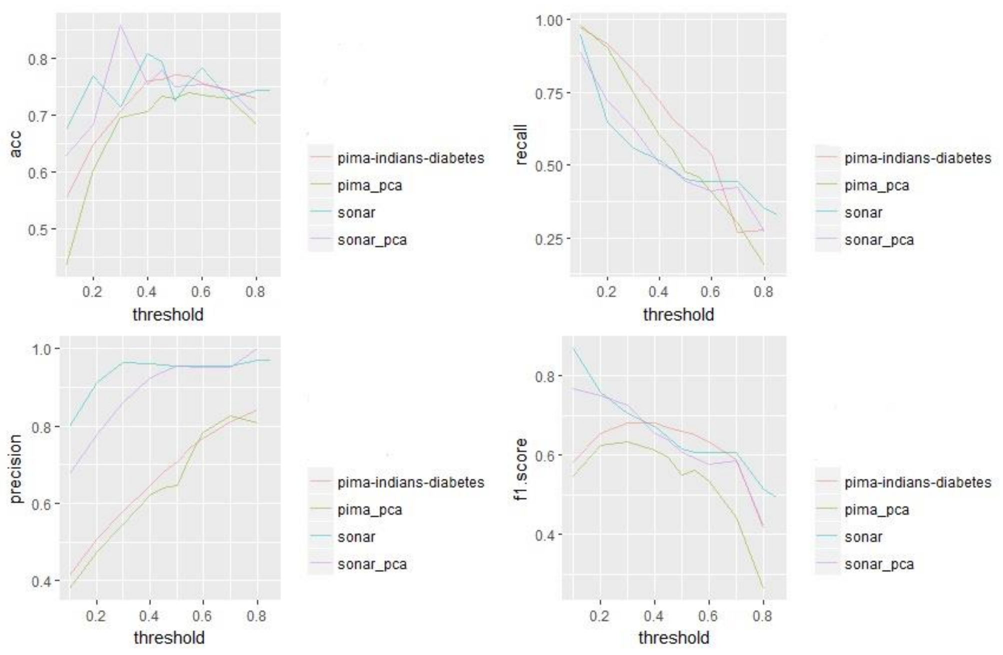


Figure (3-4) tuning of threshold

3.3 one-vs.-all logistic regression construction

Exact code for architecture of this subsection can be found in the logistic.py file, raw record can be found in result_wine.xlsx.

In the implementation of one-vs.-all logistic regression three classifiers was built, in each of the classifier, to be, the result is show in the following table:

Table 3-2 classifiers' prediction result

	TP	FP	TN	FN
wine1	59	0	119	0
wine2	70	0	107	1
wine3	48	0	130	0

Each classifier correctly predicts almost all the objects in its own dataset that turns positive with only one classifier make one mistakes. Due to the multi-classification case in this analysis, metrics such as sensitivity and specificity are not as clear therefore accuracy will be the primary metric considered. By finding the maximum probability of each object among three pre-built-models, final result is 100% accuracy.

3.4 one-vs.-one logistic regression construction

Exact code for architecture of this subsection can be found in the one_vs_one.py file, note that the method was implemented using the library scikit-learn

In the construction of one-vs.-one logistic regression, using wine dataset, $C_3^2 = 3$ classifier was built and the final result was evaluated using a vote criteria. The final accuracy reaches 88.2%. As the number of class increases, unlike the one-vs.-all LR, the require classifier raise significantly and therefore the timing will go up faster than the other model.

4 Discussion

In the previous section, an implementation process and tuning of the model is shown. The consuming of time while training a model is concerned with the size of datasets. The runtime increases as the number of attributes and number of observations increases, showing a linear relationship. Besides, number of epochs and iterations, number of folds in cross-validation also shows a linear relationship to the runtime and conclusion is proved significantly differs using a t-test. An explanation is the runtime is more consistent with the number of calculation, regardless of the value of learning rate and other value which do not change the number of calculation.

The direction of tuning a logistic regression is more like a thumb of rule, with no standard. This is because the situation of each dataset and engineers' purpose of training a model differs. From figure(3-2,3), precision metrics and recall metric are antithetical, therefore choosing a better model is more like a trade-off, Peter A. Flach et.al. (2015). A balanced way to choose among all the tuned model is concerned more about f1-score and using other model evaluation metrics such as AUC and ROC curve.

Naturally, with more time, a more realistic and dirty dataset could be used in this project and test how the tuning works. The datasets used in this project are lacking preprocessing, only with standardize and PCA method. By involving feature selection to reduce the number of attributes, more comparison pattern of the graphs can be plot and analysis. No matter how the user concerned about time constraints and accuracy, it has been shown that transforming the data through PCA resulting worst accuracy and only marginal improvements in runtime over the original dataset. The possible explanation could be the principles choose was set around only 80%, which means 80% of the information was preserved after PCA. But in the meantime it could be seen as a reduction of the possibility of overfitting of the dataset, meaning that the model contains much less bias and can, therefore, adapt better to the unseen testing data.

Another line concerning voting criteria, other algorithms such as k-means and KNN can be used as an ensemble classifier. With each model vote for the objects' prediction, the expected metrics may rise. However, due to time constraints, it was not a line of analysis considered in this project.

Table 4-1 Model result

Dataset	Epoch	L. rate	Threshold	Acc	Precision	Recall	F1-score	Std.Err
sonar	800	0.1	0.5	0.7365	0.97183	**0.711**	**0.821**	0.1353
sonar	400	0.1	0.5	0.746341	**0.98245**	0.57732	0.7272	0.136416
sonar	400	0.25	0.5	**0.80488**	0.9565	0.4536	0.6153	0.137297
sonar_pca	400	0.4	0.5	0.756	**1**	0.3608	0.5303	0.134656
pima	300	0.15	0.5	**0.7802**	0.70638	0.6194	0.66	0.1353
pima	300	0.1	0.5	0.77124	0.711	**0.6082**	**0.6559**	0.13773
pca_pima	10	0.1	0.5	0.7137	**0.81308**	0.3246	0.464	**0.12877**
pima	300	**0.1**	0.45	**0.7605**	0.645484	0.720149	0.6807	0.13396[2]

[2] Note that table 4-1 is part of the result among all the tuning

5 Bibliography

Figure(2-2,3,4) source from :http://kubicode.me/2015/08/ 30/Machine%20Learning/ Multiclass – Classification

Bliss C I. The method of probits[J]. Science, 1934, 79(2037): 38-39.

Lemeshow S, Sturdivant R X, Hosmer D W. Applied Logistic Regression (Wiley Series in Probability and Statistics)[M]. Wiley, 2013.

Theodossiou I. The effects of low-pay and unemployment on psychological well-being: a logistic regression approach[J]. Journal of health economics, 1998, 17(1): 85-104.

Hughes A J, Ben-Shlomo Y, Daniel S E, et al. What features improve the accuracy of clinical diagnosis in Parkinson's disease A clinicopathologic study[J]. Neurology, 1992, 42(6): 1142-1142.

Bender R, Grouven U. Ordinal logistic regression in medical research[J]. Journal of the Royal College of physicians of London, 1997, 31(5): 546-551.

Bestall J C, Paul E A, Garrod R, et al. Usefulness of the Medical Research Council (MRC) dyspnoea scale as a measure of disability in patients with chronic obstructive pulmonary disease[J]. Thorax, 1999, 54(7): 581-586.

Steyerberg, E. W., Harrell, F. E., Borsboom, G. J., Eijkemans, M., Vergouwe, Y. & Habbema, J. D. F. (2001), ‘Internal validation of predictive models: efficiency of some procedures for logistic regression analysis’, Journal of clinical epidemiology 54(8), 774–781.

Cooper N R, Blakey G, Sherwin C, et al. The use of GIS to develop a probability-based trunk mains burst risk model[J]. Urban Water, 2000, 2(2): 97-103.

Ter Braak C J F, Looman C W N. Weighted averaging, logistic regression and the Gaussian response model[J]. Vegetatio, 1986, 65(1): 3-11.

Hosmer Jr D W, Lemeshow S, Sturdivant R X. Applied logistic regression[M]. John Wiley & Sons, 2013.

McDonald B W. Estimating logistic regression parameters for bivariate binary data[J]. Journal of the Royal Statistical Society. Series B (Methodological), 1993: 391-397.

Flach P, Kull M. Precision-recall-gain curves: PR analysis done right[C]//Advances in Neural Information Processing Systems. 2015: 838-846.

Appendix Ⅰ

```r
#r script for time comsuming analysis
setwd('C:\\Users\\Hasee\\Desktop')
library(dplyr)
library(ggplot2)
library(RColorBrewer)
library(stringr)
#function of multiplot source from http://blog.csdn.net/tanzuozhev/article/details/51112223
#the function is use to covert multi-plots in one picture
multiplot <- function(..., plotlist=NULL, file, cols=1, layout=NULL) {
  library(grid)
  # Make a list from the ... arguments and plotlist
  plots <- c(list(...), plotlist)
  numPlots = length(plots)
  # If layout is NULL, then use 'cols' to determine layout
  if (is.null(layout)) {
    # Make the panel
    # ncol: Number of columns of plots
    # nrow: Number of rows needed, calculated from # of cols
    layout <- matrix(seq(1, cols * ceiling(numPlots/cols)),
                     ncol = cols, nrow = ceiling(numPlots/cols))
  }
  if (numPlots==1) {
    print(plots[[1]])
  } else {
    # Set up the page
    grid.newpage()
    pushViewport(viewport(layout = grid.layout(nrow(layout), ncol(layout))))

    # Make each plot, in the correct location
    for (i in 1:numPlots) {
      # Get the i,j matrix positions of the regions that contain this subplot
      matchidx <- as.data.frame(which(layout == i, arr.ind = TRUE))

      print(plots[[i]], vp = viewport(layout.pos.row = matchidx$row,
                                      layout.pos.col = matchidx$col))
    }
  }
}

time = read.csv('time.csv')
head(time)
timing = group_by(time,dataset,runtime,add = FALSE)
```

```
summarise(timing,count = n())


ggplot(timing)+
  geom_line(aes(x = epoch,y = runtime,group = as.factor(dataset),colour = as.factor(dataset),
                linetype = 'dashed'),
            alpha = 0.5)
#analyze of pima dataset
pima    = read.csv('PIMA.csv')
pima = pima[,c(-2,-3,-5)]
pima
pima_group = group_by(pima,dataset,acc,add = FALSE)
summarise(pima_group)
p1 = ggplot(pima_group)+
  geom_line(aes(x = learning.rate,y = acc,group = as.factor(dataset),colour = as.factor(dataset),
                linetype = 'dashed'),
            alpha = 0.5)
pima_group2 = group_by(pima,dataset,precision,add = FALSE)
p2 = ggplot(pima)+
  geom_line(aes(x = learning.rate,y = precision,group = as.factor(dataset),colour = as.factor(dataset),
                linetype = 'dashed'),
            alpha = 0.5)
pima_group3 = group_by(pima,dataset,recall,add = FALSE)
p3 = ggplot(pima_group3)+
  geom_line(aes(x = learning.rate,y = recall,group = as.factor(dataset),colour = as.factor(dataset),
                linetype = 'dashed'),
            alpha = 0.5)
pima_group4 = group_by(pima,dataset,f1.score,add = FALSE)
p4 = ggplot(pima_group4)+
  geom_line(aes(x = learning.rate,y = f1.score,group = as.factor(dataset),colour = as.factor(dataset),
                linetype = 'dashed'),
            alpha = 0.5)
multiplot(p1, p2, p3, p4, cols=2)
#analyze of threshold
th    = read.csv('threshold.csv')
th_group = group_by(th,dataset,acc,add = FALSE)
summarise(th_group)
p1 = ggplot(th_group)+
  geom_line(aes(x = threshold,y = acc,group = as.factor(dataset),colour = as.factor(dataset),
                linetype = 'dashed'),
            alpha = 0.5)
th_group2 = group_by(th,dataset,precision,add = FALSE)
p2 = ggplot(th)+
  geom_line(aes(x = threshold,y = precision,group = as.factor(dataset),colour = as.factor(dataset),
                linetype = 'dashed'),
                    alpha = 0.5)

th_group3 = group_by(th,dataset,recall,add = FALSE)
p3 = ggplot(th_group3)+
   geom_line(aes(x = threshold,y = recall,group = as.factor(dataset),colour = as.factor(dataset),
                          linetype = 'dashed'),
                    alpha = 0.5)
th_group4 = group_by(th,dataset,f1.score,add = FALSE)
p4 = ggplot(th_group4)+
   geom_line(aes(x = threshold,y = f1.score,group = as.factor(dataset),colour = as.factor(dataset),
                          linetype = 'dashed'),
                    alpha = 0.5)
multiplot(p1, p2, p3, p4, cols=2)
################### t-test for time    ######################
time = read.csv('t_test_time.csv')
result = data.frame(
   ID = 1:20,
   group1 = time$time[1:20],
   group2 = time$time[21:40])

t.test(time~group,time)
```

Appendix Ⅱ

Fake code for one-vs-one LR：

```
Begin:
for first_class in dataset:
        for second_class in dataset:
              if first_class >second_class:
                     train LR model
                     record coef
                     predict and record result
For object in datasets:
        Count number of prediction from each classifier
        label object as maximum vote class
evaluate result

End
```