Logistic Regression

Logistic Regression

Logistic regression is a linear model for classification rather than regression. We fit the implementation of logistic regression with L2 regularization. As an optimization problem, binary class L2 penalized logistic regression minimizes the following cost function:

$$\min_{w,c} \frac{1}{2} w^T w + C \sum_{i=1}^{n} \log(\exp(-y_i(X_i^T w + c)) + 1)$$

Within the loss function of this version, the observation y_i takes values in the set $\{-1, 1\}$ at trial i.

With the weight decay term (for any $\lambda > 0$), the cost function $J(\theta)$ is strictly convex, and is guaranteed to have a unique solution. Therefore, we use gradient descent as algorithm which is guaranteed to converge to the global minimum. The gradient of loss function $J(\theta)$ is the following derivative:

$$\nabla_{\theta_j} J(\theta) = -\frac{1}{m} \sum_{i=1}^{n} \left[-\frac{\exp(-y_i w^T x_i)}{1 + \exp(-y_i w^T x_i)} y_i x_i \right] + \lambda w$$

We use gradient descent to optimize the problem and the complete algorithm is the following pseudo code:

使用梯度下降法(Gradient Descent)求解逻辑回归

算法（梯度下降法求解逻辑回归）

输入：目标函数：$J(w)$|(对数似然损失函数)，梯度函数：$g(w) = \nabla J(w)$|，计算精度ϵ|

输出：$J(w)$|的极小值点w^*|

过程：

(1) 取初始值$w_0 \in \mathbf{R}^n$|，令$k = 0$|

(2) 计算$J(w_k)$|

$$J(w_k) = -\frac{1}{N} lnL(w_k) \Rightarrow -lnL(w_k)|$$
$$= \sum[y_i(w_k \cdot x_i) - ln(1 + e^{w_k \cdot x_i})]|$$

(3) 计算梯度$g_k = g(w_k) = \nabla J(w)$|

$$g(w_k) = \sum[x_i \cdot y_i - \frac{x_i \cdot e^{w_k \cdot x_i}}{1 + e^{w_k \cdot x_i}}]|$$
$$= \sum[x_i \cdot y_i - \pi(x_i)]|$$

若$||g_k|| < \epsilon$|，停止迭代，令

$$w^* = w_k|$$

否则，令$p_k = -g(w_k)$|，求λ_k|，使得

$$J(w_k + \lambda_k p_k) = min(J(w_k + \lambda p_k))|$$

(4) 令$w_{k+1} = w_k + \lambda_k p_k$|，计算$J(w_{k+1})$|

当$||J(w_{k+1}) - J(w_k)|| < \epsilon$|或$||w_{k+1} - w_k|| < \epsilon$|，停止迭代，令

$$w^* = w_{k+1}|$$

(5) 否则，令$k = k + 1$|，转**(3)**.

Performance Comparison

Iris Dataset

10-Fold Cross-Validation

We use cross validation to compare the self-implemented algorithm with sklearn one. Iris dataset is a classical dataset for classification task.

Classes	3
Samples per class	50
Samples total	150
Dimensionality	4
Features	real, positive

Both algorithms reach 1.0 out-of-sample precision. Moreover, the performance evaluated with ROC curve shows that both algorithms have a considerable generalization ability within Iris dataset. The ROC curves of self-implemented algorithm and sklearn are as following:

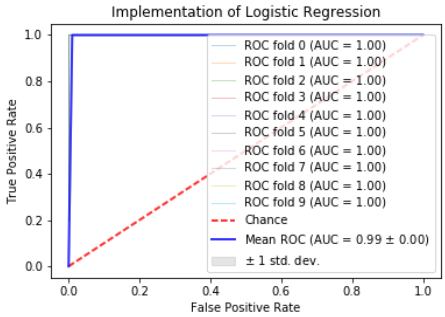


Figure 1.Self-implemented

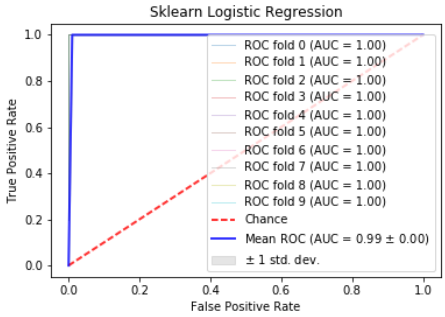


Figure 2.Sklearn

Leave One Out

We carry out Leave One Out to further evaluate the performance of both algorithms. Due to the result, both algorithms generates out-of-sample precision of 1.0. However, with respect to the efficiency of algorithm, the self-implemented one turns out to be about 800 times slower than sklearn. With examination on implementation of sklearn, it uses Cython to accelerate speed.

Algo	Time
Self-implementd	6.32s

Algo	Time
Sklearn	0.08s

Balance Scale Data Set

This data set was generated to model psychological experimental results. Each example is classified as having the balance scale tip to the right, tip to the left, or be balanced. The attributes are the left weight, the left distance, the right weight, and the right distance. The correct way to find the class is the greater of (left-distance * left-weight) and (right-distance * right-weight). If they are equal, it is balanced.

Attribute Name	No.	Element
Class Name	3	(L, B, R)
Left-Weight	5	(1, 2, 3, 4, 5)
Left-Distance	5	(1, 2, 3, 4, 5)
Right-Weight	5	(1, 2, 3, 4, 5)
Right-Distance	5	(1, 2, 3, 4, 5)

10-Fold Cross Validation

We choose label, "L" and "R" where each label has 46.08% samples. With observation on the result, both algorithms display a satisfactory performance shown below. With respect to the precision metric, self-implemented method reaches 93.39% on the test set while sklearn reaches 92.87%. The difference between these two methods may result from different random seeds.

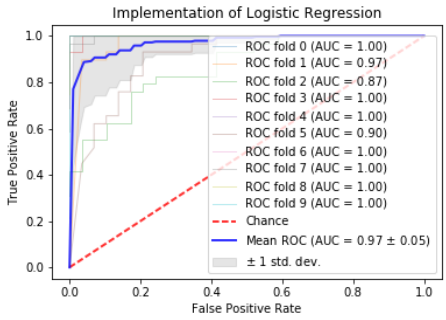


Figure 3.Self-implemented

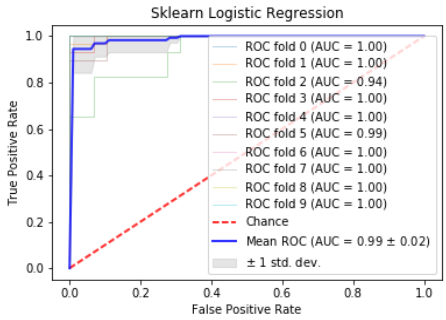


Figure 4.Sklearn

Leave One Out

We test two algorithms with Leave One Out where both methods display 91.67% on average. The difference is the running time where Sklearn takes 0.74s while self-implemented method costs 11.21s. This turns out an excellent optimization of Sklearn.

Softmax Regression

Softmax Regression

Softmax generalizes logistic regression to classification problems in which class label y can take more than two possible values. Softmax is a supervised learning method but it is often implemented in deep learning problem.

In softmax regression setting, multi-class classification is of interest(as opposed to only binary classification), and so the label y can take on k different values, rather than only two. Thus, in the training set $\{x^{(1)}, y^{(1)}), \ldots, (x^{(m)}, y^{(m)})$ we now have that $y^{(i)} \in \{1, 2, \ldots, k\}$.

Given an input x, the output from softmax regression will be the probability of the class label taking on each of the label $j = \{1, 2, \ldots, k\}$. The hypothesis probability $h_\theta(x)$ are defined as:

$$h_\theta(x^{(i)}) = \begin{bmatrix} p(y^{(i)} = 1|x^{(i)}; \theta) \\ p(y^{(i)} = 2|x^{(i)}; \theta) \\ \vdots \\ p(y^{(i)} = k|x^{(i)}; \theta) \end{bmatrix} = \frac{1}{\sum_{j=1}^{k} e^{\theta_j^T x^{(i)}}} \begin{bmatrix} e^{\theta_1^T x^{(i)}} \\ e^{\theta_2^T x^{(i)}} \\ \vdots \\ e^{\theta_k^T x^{(i)}} \end{bmatrix}$$

Here, paramters are of the form $\theta_1, \theta_2, \ldots, \theta_k \in \mathcal{R}^{n+1}$.

Cost Function of softmax regression are analogous to logistic regression,

$$\begin{aligned} J(\theta) &= -\frac{1}{m} \left[\sum_{i=1}^{m} (1 - y^{(i)}) \log(1 - h_\theta(x^{(i)})) + y^{(i)} \log h_\theta(x^{(i)}) \right] \\ &= -\frac{1}{m} \left[\sum_{i=1}^{m} \sum_{j=0}^{1} 1 \left\{ y^{(i)} = j \right\} \log p(y^{(i)} = j|x^{(i)}; \theta) \right] \end{aligned}$$

The minimization of softmax regression is similar to logistic regression and iterative optimization algorithms such as gradient descent and newton conjugate gradient method are implemented. Taking derivatives, the gradient is:

$$\nabla_{\theta_j} J(\theta) = -\frac{1}{m} \sum_{i=1}^{m} \left[x^{(i)} \left(1\{y^{(i)} = j\} - p(y^{(i)} = j|x^{(i)}; \theta) \right) \right]$$

Armed with this formula for the derivative, we can then plug it into an optimization algorithm, and have it minimize $J(\theta)$. Thus, on each iteration we would perform the update $\theta_j := \theta_j - \alpha \nabla_{\theta_j} J(\theta)$ (for each $j = (1, \ldots, k)$).

Softmax Regression vs. k Binary Classifiers

Compared with k binary classifiers, softmax regression holds with an assumption that the data classes are mutually independent. If the assumption holds, each sample will fall into one class otherwise every sample could share with several labels. In such condition, k binary classiers are suitable for this task.

Comparison with OvO and OvR

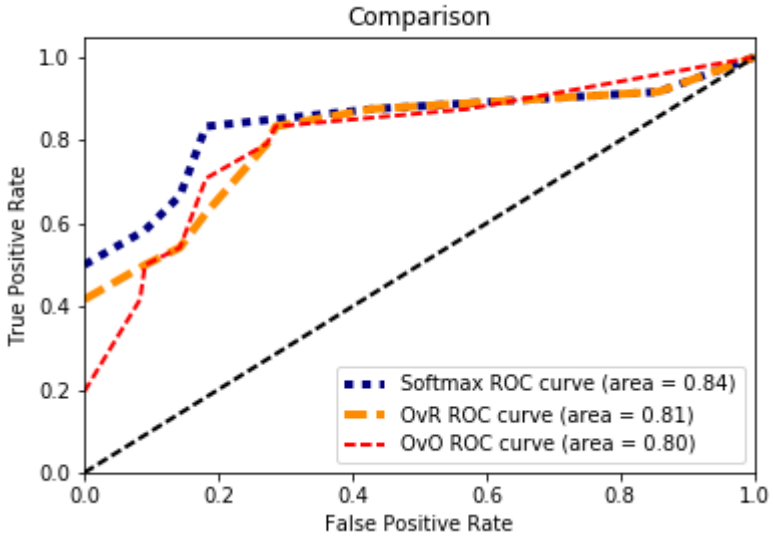
Iris Dataset

The iris dataset is a classic and very easy multi-class classification dataset.This data sets consists of 3 different types of irises' (Setosa, Versicolour, and Virginica) petal and sepal length and the classes are supposed to be mutually exclusive. Therefore, before experiment, assumption is made that the classification performance of softmax should overweight OvO and OvR.

Classes	3
Samples per class	50
Samples total	150
Dimensionality	4
Features	real, positive

We use 10-fold cross-validation to compare the performances of three methods. With respect to the chart below, softmax has a better test set prediction accuracy than One versus One and One versus All. For ROC curve, the curve of softmax completely contains both of OvO and OvR, indicating that softmax has a even more general and model-based performance than OvO and OvR with respect to Iris Dataset.

Accuracy Score		
Softmax	One vs. One	One vs. All
0.73	0.6	0.666



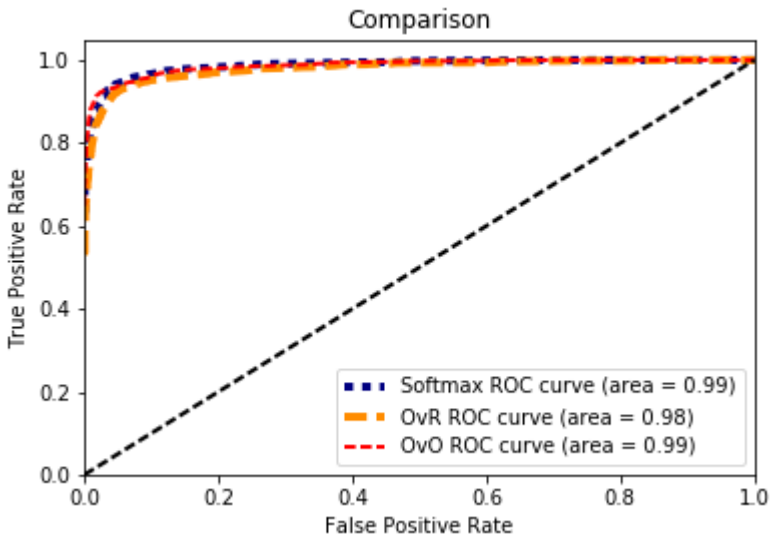
Mnist Dataset

The Mnist dataset is a classic multi-class classification dataset. Each datapoint is a 8x8 image of a digit.

Mnist Dataset	
Classes	10
Samples per class	~180
Samples total	1797
Dimensionality	64
Features	integers 0-16

We use 10-fold cross-validation to compare the performances of three methods. With observation on the results, all three methods display a similar performance of prediction accuracy, where OvO has a 2% accuracy more than softmax. In the meanwhile, all three methods presents a perfect ROC curve. One assumption for similar performances is that compared with sample size, the feature size is sufficient enough to ensure generalization ability. In conclusion, within Mnist Dataset, Softmax, One vs. One and One vs. Rest have a relatively analogous classification performance.

Accuracy Score		
Softmax	One vs. One	One vs. All
0.91	0.93	0.90



Balanced Scaled Dataset

This data set was generated to model psychological experimental results. Each example is classified as having the balance scale tip to the right, tip to the left, or be balanced. The attributes are the left weight, the left distance, the right weight, and the right distance. The correct way to find the class is the greater of (left-distance * left-weight) and (right-distance * right-weight). If they are equal, it is balanced.

Attribute Name	No.	Element
Class Name	3	(L, B, R)

Attribute Name	No.	Element
Left-Weight	5	(1, 2, 3, 4, 5)
Left-Distance	5	(1, 2, 3, 4, 5)
Right-Weight	5	(1, 2, 3, 4, 5)
Right-Distance	5	(1, 2, 3, 4, 5)

We use 10-fold cross-validation to compare the performances of three methods. Referred to the results, Softmax overweights the performances of the other two methods. As shown below, the ROC curve of Softmax completely contains other two methods. Moreover, Softmax dominates the out-of-sample test set precision by about 8%. Since the dataset classes are unbalanced, we modify the class weight to balance.

Accuracy Score		
Softmax	One vs. One	One vs. All
0.70	0.62	0.62

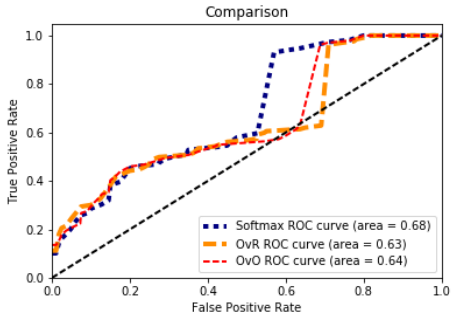


Figure 4.Comparison on Balanced Dataset

Cost Sensitive

Cost sensitive takes different penalties of classification errors into consideration. In many real-world domains, such as fraud detection and medical diagnosis, the class distribution of the data is skewed and the cost of misclassifying a minority-class example is substantially greater than the cost of misclassifying a majority-class example. Cost sensitive incorporates the misclassification costs into the learning algorithm. While referred to related articles that makes comparison between cost sensitive method, oversampling and undersampling, there is no clear winner when applying to different problems. However, worth noticing that if focusing exclusively on datasets with more than 10,000 examples, then the cost sensitivelearning algorithm consistently outperforms thesampling methods (oversampling appears to be the best method for small data sets).

[1] In recommendation system, incorrect recommendations to users should have higher penalty since they could arouse dissatisfaction among users.

[2] In junk mail screening, misscreening should be put on higher penalty weight because it brings about trouble when useful and significant mails are thrown into junk file.

[3] In bank loaning, banks need to predict the probability of default. If setting negative sample as default, false positive on large-amount cases should be put on more penalty because this may bring about financial blow on the bank.

[4] In earthquake prediction, set correct prediction as positive sample. False positive should be put on more penalty because the model should increase the ratio of true positive. An incorrect positive case could bring about a far more disastrous influence.

[5] In intrusion detection system, mistaking attack as harmless will bring about far more loss than mistaking the harmless as harmful.

[6] In medical problem as cancer diagnosis, since cancer patients are minority, the cost of classify a normal person is far less than omit a cancer patient.

[7] In VISA fraud problem, omitting a potential fraud is more harmful than misclassify as an ordinary case as fraud.

[8] In facial recognition applying to electronic payment system, mistaking a similar or falsely recognizing faces has more cost than incorrectly classify the right face.

[9] In facial recognition applying to photo function such as Faceu or Instagram, failure to recognize the right region of face takes more cost because this will decrease users' satisfaction with the application.

[10] In demand prediction problem for suppliers, insufficient prediction of demand takes more cost than excessive ones because insufficient supply amount could lead to customer loss.

SMOTE

SMOTE Method

SMOTE proposes an over-sampling approach in which the minority class is over-sampled by creating “synthetic” examples rather than by over-sampling with replacement. SMOTE generates synthetic examples in a less application-specific manner, by operating in “feature space” rather than “data space”. The minority class is over-sampled by taking each minority class sample and introducing synthetic examples along the line segments joining any/all of the k minority class nearest neighbors. Depending upon the amount of over-sampling required, neighbors from the k nearest neighbors are randomly chosen.

Procedure

Synthetic samples are generated in the following way: Take the difference between the feature vector (sample) under consideration and its nearest neighbor. Multiply this difference by a random number between 0 and 1, and add it to the feature vector under consideration. This causes the selection of a random point along the line segment between two specific features. This approach effectively forces the decision region of the minority class to become more general.

Pros and Cons

The synthetic examples cause the classifier to create larger and less specific decision regions, rather than smaller and more specific regions. More general regions are now learned for the minority class samples rather than those being subsumed by the majority class samples around them. The effect is that decision trees generalize better. Compared with traditional method, replication method, the

decision region that results in a classification decision for the minority class can actually become smaller and more specific as the minority samples in the region are replicated. This is the opposite of the desired effect. While synthetic over-sampling works to cause the classifier to build larger decision regions that contain nearby minority class points. However, synthetic method still learns from the information provided in the dataset, albeit with different cost information.