華南理工大學

South China University of Technology

The Experiment Report of Machine Learning

SCHOOL: SCHOOL OF SOFTWARE ENGINEERING

SUBJECT: SOFTWARE ENGINEERING

Author:
Shoukai Xu and Yaofu Chen

Supervisor:
Mingkui Tan or Qingyao Wu

Student ID：201530612231

Grade:
Undergraduate

December 9, 2017

Logistic Regression, Linear Classification and Stochastic Gradient Descent

Abstract—

I. Introduction

In statistics, logistic regression, or logit regression, or logit model[1] is a regression model where the dependent variable (DV) is categorical. This article covers 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. Cases where the dependent variable has more than two outcome categories may be analysed in multinomial logistic regression, or, if the multiple categories are ordered, in ordinal logistic regression.[2] In the terminology of economics, logistic regression is an example of a qualitative response/discrete choice model.

A linear classifier achieves this by making a classification decision based on the value of a linear combination of the characteristics. An object's characteristics are also known as feature values and are typically presented to the machine in a vector called a feature vector. Such classifiers work well for practical problems such as document classification, and more generally for problems with many variables (features), reaching accuracy levels comparable to non-linear classifiers while taking less time to train and use.

Stochastic gradient descent (often shortened to SGD), also known as incremental gradient descent, is a stochastic approximation of the gradient descent optimization and iterative method for minimizing an objective function that is written as a sum of differentiable functions. In other words, SGD tries to find minima or maxima by iteration.

II. Methods and Theory

Logistic regression is an important machine learning algorithm. The goal is to model the probability of a random variable Y being 0 or 1 given experimental data.

Consider a generalized linear model function parameterized by W

$$h_{\mathbf{w}}(\mathbf{x}) = g(\mathbf{w}^\top \mathbf{x}) = \frac{1}{1+e^{-\mathbf{w}^\top \mathbf{x}}}$$

Probability:

$$p = \begin{cases} h_{\mathbf{w}}(\mathbf{x}_i) & y_i = 1 \\ 1 - h_{\mathbf{w}}(\mathbf{x}_i) & y_i = 0 \end{cases}$$

Log-likehood loss function :

$$J(\mathbf{w}) = -\frac{1}{n}\left[\sum_{i=1}^{n} y_i \log h_{\mathbf{w}}(\mathbf{x}_i) + (1-y_i)\log\left(1-h_{\mathbf{w}}(\mathbf{x}_i)\right)\right]$$

Use the Gradient Descent to Get w:

For a sample:

$$\frac{\partial J(\mathbf{w})}{\partial \mathbf{w}} = \left(h_{\mathbf{w}}(\mathbf{x}) - y\right)\mathbf{x}$$

$$\mathbf{w} := \mathbf{w} - \alpha\left(h_{\mathbf{w}}(\mathbf{x}) - y\right)\mathbf{x}$$

For all samples:

$$\frac{\partial J(\mathbf{w})}{\partial \mathbf{w}} = \frac{1}{n}\sum_{i=1}^{n}\left(h_{\mathbf{w}}(\mathbf{x}_i) - y\right)\mathbf{x}_i$$

$$\mathbf{w} := \mathbf{w} - \frac{1}{n}\sum_{i=1}^{n}\alpha\left(h_{\mathbf{w}}(\mathbf{x}_i) - y_i\right)\mathbf{x}_i$$

A linear classifier achieves this by making a classification decision based on the value of a linear combination of the characteristics.,If the input feature vector to the classifier is a real vector $\vec{x}_i$,then the output score is

$$y = f(\vec{w}\cdot\vec{x}) = f\left(\sum_j w_j x_j\right)$$

where $\vec{w}$ is a real vector of weights and f is a function that converts the dot product of the two vectors into the desired output. (In other words, $\vec{w}$ is a one-form or linear functional mapping $\vec{x}$ onto R.) The weight vector $\vec{w}$ is learned from a set of labeled training samples.

Discriminative training of linear classifiers usually proceeds in a supervised way, by means of an optimization algorithm that is given a training set with desired outputs and a loss function that measures the discrepancy between the classifier's outputs and the desired outputs. Thus, the learning algorithm solves an optimization problem of the form

$$\underset{\mathbf{w}}{\operatorname{argmin}}\ R(\mathbf{w}) + C\sum_{i=1}^{N} L(y_i, \mathbf{w}^\top \mathbf{x}_i)$$

SGD tries to find minima or maxima by iteration. Both statistical estimation and machine learning consider the problem of minimizing an objective function that has the form of a sum:

$$Q(w) = \frac{1}{n}\sum_{i=1}^{n} Q_i(w),$$

When used to minimize the above function, a standard (or "batch") gradient descent method would perform the following iterations :

$$w := w - \eta \nabla Q(w) = w - \eta \sum_{i=1}^{n} \nabla Q_i(w)/n,$$

where η is a step size (sometimes called the learning rate in machine learning).

Parameter updates

Nesterov Momentum is a slightly different version of the momentum update that has recently been gaining popularity. It enjoys stronger theoretical converge guarantees for convex functions and in practice it also consistenly works slightly better than standard momentum.

$$\mathbf{g}_t \leftarrow \nabla J(\boldsymbol{\theta}_{t-1} - \gamma \mathbf{v}_{t-1})$$
$$\mathbf{v}_t \leftarrow \gamma \mathbf{v}_{t-1} + \eta \mathbf{g}_t$$
$$\boldsymbol{\theta}_t \leftarrow \boldsymbol{\theta}_{t-1} - \mathbf{v}_t$$

RMSprop. The RMSProp update adjusts the Adagrad method in a very simple way in an attempt to reduce its aggressive, monotonically decreasing learning rate. In particular, it uses a moving average of squared gradients instead:

$$\mathbf{g}_t \leftarrow \nabla J(\boldsymbol{\theta}_{t-1})$$
$$G_t \leftarrow \gamma G_t + (1-\gamma)\mathbf{g}_t \odot \mathbf{g}_t$$
$$\boldsymbol{\theta}_t \leftarrow \boldsymbol{\theta}_{t-1} - \frac{\eta}{\sqrt{G_t + \epsilon}} \odot \mathbf{g}_t$$

Adam. Adam is a recently proposed update that looks a bit like RMSProp with momentum. The (simplified) update looks as follows:

$$\mathbf{g}_t \leftarrow \nabla J(\boldsymbol{\theta}_{t-1})$$
$$\mathbf{m}_t \leftarrow \beta_1 \mathbf{m}_{t-1} + (1-\beta_1)\mathbf{g}_t$$
$$G_t \leftarrow \gamma G_t + (1-\gamma)\mathbf{g}_t \odot \mathbf{g}_t$$
$$\alpha \leftarrow \eta \frac{\sqrt{1-\gamma^t}}{1-\beta^t}$$
$$\boldsymbol{\theta}_t \leftarrow \boldsymbol{\theta}_{t-1} - \alpha \frac{\mathbf{m}_t}{\sqrt{G_t + \epsilon}}$$

Adadelta is an extension of Adagrad that seeks to reduce its aggressive, monotonically decreasing learning rate. Instead of accumulating all past squared gradients, Adadelta restricts the window of accumulated past gradients to some fixed size w

$$\mathbf{g}_t \leftarrow \nabla J(\boldsymbol{\theta}_{t-1})$$
$$G_t \leftarrow \gamma G_t + (1-\gamma)\mathbf{g}_t \odot \mathbf{g}_t$$
$$\Delta\boldsymbol{\theta}_t \leftarrow -\frac{\sqrt{\Delta_{t-1} + \epsilon}}{\sqrt{G_t + \epsilon}} \odot \mathbf{g}_t$$
$$\boldsymbol{\theta}_t \leftarrow \boldsymbol{\theta}_{t-1} + \Delta\boldsymbol{\theta}_t$$
$$\Delta_t \leftarrow \gamma \Delta_{t-1} + (1-\gamma)\Delta\boldsymbol{\theta}_t \odot \Delta\boldsymbol{\theta}_t$$

III. EXPERIMENT

Motivation

1. Compare and understand the difference between gradient descent and stochastic gradient descent.
2. Compare and understand the differences and relationships between Logistic regression and linear classification.
3. Further understand the principles of SVM and practice on larger data.

Dataset

Experiment uses a9a of LIBSVM Data, including 32561/16281(testing) samples and each sample has 123/123 (testing) features. Please download the training set and validation set.

Logistic Regression and Stochastic Gradient Descent

1. Load the training set and validation set.
2. Initalize logistic regression model parameters, you can consider initalizing zeros, random numbers or normal distribution.
3. Select the loss function and calculate its derivation, find more detail in PPT.
4. Calculate gradient toward loss function from partial samples.
5. Update model parameters using different optimized methods(NAG，RMSProp，AdaDelta and Adam).
6. Select the appropriate threshold, mark the sample whose predict scores greater than the threshold as positive, on the contrary as negative. Predict under validation set and get the different optimized method loss L(NAG) ,L(RMSProp) L(AdaDelta) and L(Adam)
7. Repeate step 4 to 6 for several times, and drawing graph of L(NAG) ,L(RMSProp) L(AdaDelta) and L(Adam), and with the number of iterations.

Linear Classification and Stochastic Gradient Descent

1.Load the training set and validation set.

2.Initalize SVM model parameters, you can consider initalizing zeros, random numbers or normal distribution.

3.Select the loss function and calculate its derivation, find more detail in PPT.

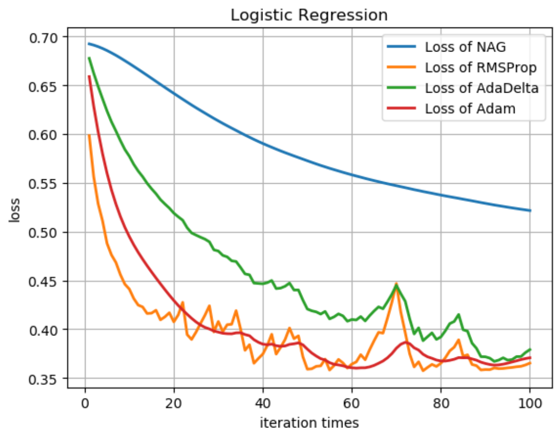
4.Calculate gradient toward loss function from partial samples.

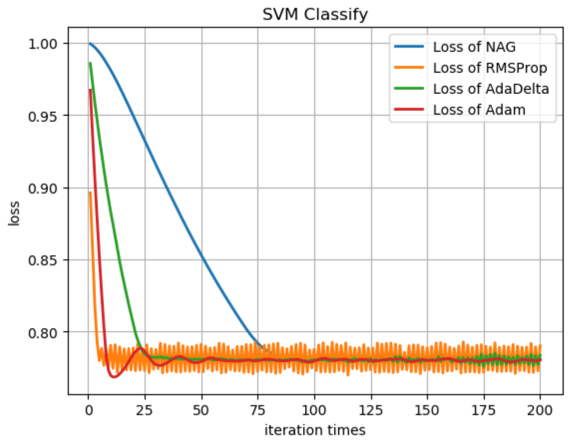
5.Update model parameters using different optimized methods(NAG，RMSProp，AdaDelta and Adam).

6.Select the appropriate threshold, mark the sample whose predict scores greater than the threshold as positive, on the contrary as negative. Predict under validation set and get the different optimized method loss L(NAG) ,L(RMSProp) L(AdaDelta) and L(Adam)

7. Repeate step 4 to 6 for several times, and drawing graph of L(NAG) ,L(RMSProp) L(AdaDelta) and L(Adam)with the number of iterations.

The result:





IV. CONCLUSION

Through the experiment ,I understand the difference between gradient descent and stochastic gradient descent.get the main points of the differences and relationships between Logistic regression and linear classification.Furtherly, I am touched of the principles of SVM while practicing on larger data. I feel very excited of the results!