Report of Support Vector Machine Project

Name ID: Yutong Wang 11611808
Department: Computer Science and Technology
Institution: Southern University of Science and Technology
Email: 11611808@mail.sustc.edu.cn

I. Preliminaries

A. Problem Description

In this project, Support Vector Machine(SVM, also support vector network[1]) is used to classify data into two classes. A SVM classifies data by finding the best hyperplane that separates all data points of one class from those of the other class. The best hyperplane for an SVM means the one with the largest margin between the two classes. Margin means the maximal width of the slab parallel to the hyperplane that has no interior data points.

The support vectors are the data points that are closest to the separating hyperplane; these points are on the boundary of the slab. The following figure illustrates these definitions, with + indicating data points of type 1, and – indicating data points of type –1.

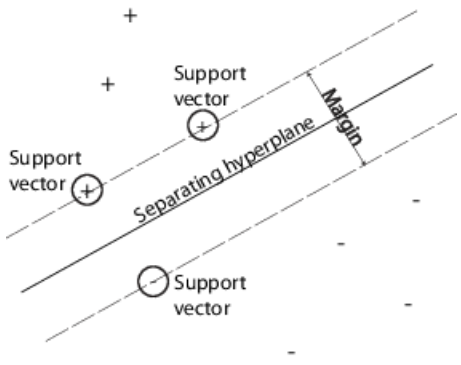

Fig 1. SVM.

B. Problem Applications

SVM has multiple applications and can be used to solve various real world problems:

- Text and hypertext categorization.
- Classification of images.
- Recognizing hand-written characters.
- Classification of proteins with up to 90% of the compounds classified correctly in the biological and other sciences.
- ...

II. Methodology

A. Notation

Notation	Meaning
w	Weight vector of the hyperplane of SVM.
b	Offset vector of the hyperplane of SVM.
$train\_x$	x in training dataset.
$train\_y$	y in training dataset.
$test\_x$	x in testing dataset.
$test\_y$	y in testing dataset.
$Q(w)$	Loss function of SVM.
η	Learning rate.
$epoch$	Number of times in training process.
Err	Error rate during the test.
T	Running time of the program.

Table 1. Notations table.

B. Data Structure

There are not much data structure used in this project. The most often used one is list including 1d and 2d array. For instance, I use 2d-array to store train data $train\_x$ and test data $test\_x$ and use 1d-array for train data $train\_y$, test data $test\_y$ and final result.

C. Model Design

- How to fomulate the problem?

  Given a training dataset of n points of the form $(\vec{x}_1, y_1)$, ..., $(\vec{x}_n, y_n)$ where y_i are whether 1 or -1, each indicating the class to which the point $\vec{x}_i$ belongs. This problem aims to find the "maximum-margin hyperplane" that divides the group of points $\vec{x}_i$ for which $\vec{y}_i = 1$ from the group of points for which $\vec{y}_i = -1$, which is defined so that the distance between the hyperplane and the nearest point $\vec{x}_i$ from either group is maximized. Any hyperplane can be written as the set of points $\vec{x}_i$ satisfying $\vec{w} \cdot \vec{x} - b = 0$.

- How to solve the problem?

  To solve this problem, I use two methods to compare the efficiency and effectiveness of this problem which are stochastic gradient descent and sequential minimal optimization(SMO).
  - Stochastic Gradient Descent(SGD):

    In stochastic gradient descent, the

true gradient of $Q(w)$ is approximated by a gradient at a single example: $w = w - \eta \nabla Q_i(w)$ where η is a step size (sometimes called the learning rate in machine learning) and $Q_i(w)$ is the value of the loss function at i-th example.

– Sequential Minimal Optimization(SMO):

SMO is an iterative algorithm for solving the optimization problem described above. SMO breaks this problem into a series of smallest possible sub-problems, which are then solved analytically. Because of the linear equality constraint involving the Lagrange multipliers α_i, the smallest possible problem involves two such multipliers. Then, for any two multipliers α_1 and α_2, the constraints are reduced to: $0 \leq \alpha_1, \alpha_2 \leq C$, $y_1\alpha_1 + y_2\alpha_2 = k$, and this reduced problem can be solved analytically: one needs to find a minimum of a one-dimensional quadratic function. k is the negative of the sum over the rest of terms in the equality constraint, which is fixed in each iteration.

D. Details of Algorithm

Algorithm 1 SGD:

Different values of w and b determine whether the hyperplane is robust enough to do the binary classification. So the process of training is to find w and b good enough to achieve optimal separation effect. In order to let the SVM's output to be as close as the real value, we should compare the difference between the prediction value and the real value and modify the model according to the differnece.

Stochastic gradient descend method is used to modify the model. The pseudo-code is as follows:

```
Choose an initial vector of parameters w and learning rate e
Repeat until an approximate minimum is obtained:
        Randomly shuffle examples in the training set.
        For i  = 1,2, ..., n, do:
            w = w - e * (-y * x)
```

Loss function is used to compare the difference between the prediction value and the real value. The pseudo-code is as follows:

```
function get_loss(x, y):
    loss = max{0, 1 - y * <x, w>}
```

Training process is as follows:

```
Initialize epoch.
function train:
    Repeat epoch times:
        Randomly shuffle the training set.
        For each (x, y) pair in training set:
            loss  = loss + get_loss(x, y)
            Modify w with SGF method.
```

Algorithm 2 SMO:

The algorithm proceeds as follows.

1. Find a Lagrange multiplier α_1 that violates the KKT conditions for the optimization problem.

2. Pick a second multiplier α_2 and optimize the pair (α_1, α_2).

3. Repeat steps 1 and 2 until convergence.

III. Empirical Verification

A. Dataset

I use the given training data set. There are 1334 ten dimension samples in the data set. For dataset 1, 2/3 of it is to train and 1/3 is for testing the result. For dataset 2, 3/4 of it is to train and 1/4 is for testing the result.

B. Performance measure

Test environment: MacOS Sierra 10.12.6; Processor 2.9 GHz Intel Core i5; Memory 8 GB 2133 MHz LPDDR3.

How to meassure:

I calculate the error rate Err to measure the performance of either SGD method or SMO method. The error rate equals to the number of wrong prediction outputs over the total size of test data. Besides, running time is also taken into consideration.

C. Hyperparameters

The hyperparameter refers to *epoch* and learning rate η of both SGD and SMO. After experiments, I set *epoch* to be 200 and η to be 0.01.

D. Experimental results

1. Dataset 1: Training set from 1st sample to 890th sample, testing set from 891th sample tp 1334th sample.

epoch	Err_{SGD}	$T_{SGD}(s)$	Err_{SMO}	$T_{SMO}(s)$
200	0.0045	0.74	0.0224	10.58
2000	0.0045	6.18	0.0315	15.14
20000	0.0023	56.31	0.0297	15.44

Table 2. Varying epoch, costant $\eta = 0.01$

η	Err_{SGD}	$T_{SGD}(s)$	Err_{SMO}	$T_{SMO}(s)$
0.1	0.0045	0.73	0.027	0.15
0.01	0.0045	0.74	0.0224	10.58
0.001	0.0022	6.28	0.0315	11.17

Table 3. Varying η, costant epoch = 200

2. Dataset 2: Training set from 335st sample to 1334th sample, testing set from 1st sample tp 334th sample.

epoch	Err_{SGD}	$T_{SGD}(s)$	Err_{SMO}	$T_{SMO}(s)$
200	0.003	0.82	0.0059	15.75
2000	0.0	7.7	0.006	23.42
20000	0.0	67.09	0.0059	23.64

Table 4. Varying epoch, costant $\eta = 0.01$

η	Err_{SGD}	$T_{SGD}(s)$	Err_{SMO}	$T_{SMO}(s)$
0.1	0.0	0.82	0.051	0.16
0.01	0.0	0.78	0.0089	15.13
0.001	0.0	0.82	0.0089	15.75

Table 5. Varying η, costant epoch = 200

3. One more experiment is to check the convergence of *epoch* when $\eta = 0.01$. X-axis refers to *epoch* and Y-axis refers to the error rate. As can be seen from the figure below, the error rate becomes stable when *epoch* achieves around 200.

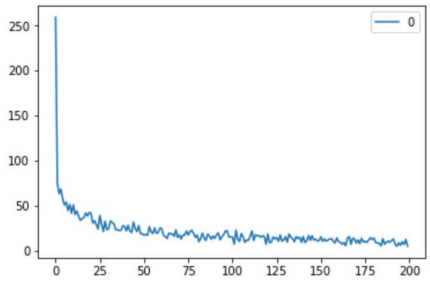

Fig 2. Convergence of *epoch*.

E. Conclusion

To summarize, SMO uses Platt's Sequential Minimal Optimization algorithm (a batch algorithm) for learning linear and non-linear support vector machines. SGD uses stochastic gradient descent to learn linear models (linear SVM, logistic regression and multiple linear regression). Stochastic gradient descent is an incremental anytime algorithm, so it can be applied to data streams or data sets that are too large to fit into main memory.

From the experiment, I learn that the effect of two methods SGD and SMO depends greatly on the training data set. Although SGD seems to have a better effect on this problem because it can achieve smaller error rate in less time, SMO also has its advantage and may be more proper to other problems and datasets. Because we can see that when η is constant and *epoch* becomes larger, the running time of SMO doesn't increase as SGD. More experiments should be done in the future and from the result, I choose SGD in this project with η equaling to 0.01 and *epoch* equaling to 200 because of its convergence.

References

[1] Cortes, Corinna; Vapnik, Vladimir N. "Support-vector networks". Machine Learning. 20 (3): 273–297, 1995.