CS303 Project4: Support Vector Machine Using PEGASOS Algorithm

Shijie Chen 11612028
Department of Computer Science and Engineering
Southern University of Science and Technology
Shenzhen, Guangdong, China
Email: 11612028@mail.sustc.edu.cn

Abstract—Support Vector Machine (SVM) is a classic machine learning approach to solve classicification problems. In this project, I implemented a SVM using the PEGASOS algorithm and compared the performance of PEGASOS algorithm with SMO.

I. Preliminaries

A. Notation

The following notations are used in this report:

TABLE I
representation

Name	Variable
Weight Matrex	w
Vector	x
Class label	y
Cernel Function	$K(x, y)$
Lagrange Multipliers	α_i

B. Support Vector Machine

In machine learning, support vector machines (SVMs) are supervised models associated with classification and regression problems.

Consider a classification problem given a set of training examples, $x_1, y_1, \ldots, x_n, y_n$, where x_i is an input vector and $y_i \in -1, 1$ is the correspoding class label. A SVM is trained with training examples and assigns class label $y \in -1, 1$to a new vector x.

SVMs construct a decision plane, which is a hyperplane $wx+b = 0$ to decide the category of a given vector. Since a larger margin, distance of the nearest training pattern to the decision plane, leads to better generalizing ability, the problem of getting a good decision plane can be expressed as the following convex quadratic programming problem:

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & \frac{1}{2}|w|^2 \\ \text{subject to} \quad & y_i((w \cdot x_i) + b) \geq 1,\ i = 1, \ldots, l. \end{aligned}$$

Alternatively, we can solve its dual problem:

$$\begin{aligned} \underset{x}{\text{maximize}} \quad & \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} y_i y_j K(x_i, x_j) \alpha_i \alpha_j \\ \text{subject to} \quad & 0 \leq \alpha_i \leq C, \sum_{i=1}^{n} y_i \alpha_i = 0,\ i = 1, \ldots, l. \end{aligned}$$

where C is an hyperparameter and $K(x_i, y_i)$ is the kernel function, both given by the user. α_i are Lagrange multipliers used to construct the dual problem.

In this project, such a problem is solved using two methods: Primal Estimated sub-GrAdient SOlver for SVM (PEGASOS) and Sequential Minimal Optimization (SMO).

Performance is measured by the accuracy of the classifier on test dataset and the time needed for training.

II. Methodology

A. Data Structure

Data structure used in this project is numpy.array.

B. The PEGASOS Algorithm

The PEGASOS algorithm is a kind of gradient descendent method. It performs gradient descent on each training data with a decreasing step size [1].

Algorithm 1 PEGASOS

```
function PEGASOS(data, y)
    w_1 ← 0
    t ← 0
    for iter = 1, ..., 20 do
        for j = 1, ..., |data| do
            t ← t + 1
            η_t ← 1/(tλ)
            if y_j(w_t x_j < 1 then
                w_{t+1} ← (1 − η_t λ)w_t + η_t y_j x_j
            else
                w_{t+1} ← (1 − η_t λ)w_t
            end if
        end for
    end for
     return w_{t+1}
end function
```

C. The SMO algorithm

The SMO algorithm [2] breaks the dual problem to its smallest parts. The smallest subproblem is to solve a problem with only two lagrange multipliers α_1, α_2 analytically with constraints:

$$0 \leq \alpha_1 \alpha_2 \leq C$$

$$y_1 \alpha_1 + y_2 \alpha_2 = k$$

where k is the negative of the sum of the rest equality constraints.

The SMO algorithm works as follows:

1) Find a α_1 that violates the Karush-Kuhn-Tucker (KKT) conditions the most.
2) Find a α_2 that optimizes (α_1, α_2)
3) Repeat from 1) until KKT condition is satisfied.

Since the SMO algorithm checks the KKT conditions, it guarantees an optimal solution if the training set is linearly seperable.

However, since there're $n(n+1)/2$ combinations of α_1 and α_2, hueristics are used to cut computation time in popular implementations.

D. Parameters

The λ in PEGASOS is a hyperparameter that controls the learning rate and is given by the user. Through experiments, I found $\lambda = 0.05$ a good choice in this project.

III. Validation

A. Environment

- OS: windows 10
- RAM: 16G
- CPU: Intel Core i7 8700k @ 3.7GHz
- Python: 3.7.1

B. Benchmark

I used the breast cancer dataset to validate my implementation of PEGASOS algorithm. For comparison, I used the SVM.SVC from $scikit-learn$ package whose implementation is based on SMO algorithm. Kernel function of SMO is set as $radial\ basis\ function(rbf)$ and $linear$,

Note that SVM of *sklearn* is implementated in C and PEGASOS is implementated in python. Running time is not compared in this report.

TABLE II
representation

Benchmark	Dim	Model	N_{train}	N_{test}	Accuracy
train_data	10	PEGASOS	2000	2000	0.991
train_data	10	PEGASOS	2000	600	0.990
train_data	10	SMO(rbf)	2000	2000	1.000
train_data	10	SMO(linear)	2000	2000	0.998
train_data	10	SMO(rbf)	2000	600	1.000
train_data	10	SMO(linear)	2000	600	1.000
Breast Cancer	9	PEGASOS	2000	2000	0.737
Breast Cancer	9	PEGASOS	2000	700	0.710
Breast Cancer	9	SMO(rbf)	2000	2000	0.906
Breast Cancer	9	SMO(linear)	2000	2000	0.710
Breast Cancer	9	SMO(rbf)	2000	700	0.867
Breast Cancer	9	SMO(linear)	2000	700	0.736

IV. Conclusion

The PEGASOS algorithm is easy to implement and performs well when the training data is linearly seperable.

Also note that SMO has better performance even with a linear kernel. A larger leap in accuracy can be get if we can choose the right kernel function, as shown in the Breast Cancer dataset.

V. Acknowledgement

The author would like to thank Ying Zhou for providing the preprocessed Breast Cancer dataset.

References

[1] S. Shalev-Shwartz, Y. Singer, N. Srebro, and A. Cotter, "Pegasos: Primal estimated sub-gradient solver for svm," Mathematical programming, vol. 127, no. 1, pp. 3–30, 2011.

[2] J. Platt, "Sequential minimal optimization: A fast algorithm for training support vector machines," 1998.