
Amir M. Mir$^1$ and Jalal A. Nasiri$^1$

1 Iranian Research Institute for Information Science and Technology (IranDoc), Tehran, Iran

Abstract

This paper presents the LightTwinSVM program and its features. It is a simple and fast implementation of twin support vector machine algorithm (TwinSVM). Numerical experiments on benchmark datasets show the effectiveness of the LightTwinSVM program in terms of accuracy. This program can be used by researchers, students, and practitioners to solve classification tasks.

Introduction

Classification is a widely-used learning method in machine learning. The task of classification consists of training samples for which the class labels are available. On the basis of such training samples, a classifier learns a rule for predicting an unseen sample (Shalev-Shwartz & Ben-David, 2014). To do a classification task, many algorithms have been proposed in machine learning literature such as Artificial Neural Network (ANN), Support Vector Machine (SVM), K-nearest neighbors (KNN), and Decision Trees. Among these classification algorithms, SVM classifier has relatively better prediction accuracy and generalization (Kotsiantis, Zaharakis, & Pintelas, 2007). The main idea of SVM is to find the optimal separating hyperplane with the largest margin between the two classes. Figure 1 shows the geometric illustration of SVM classifier.

Over the past decade, researchers have proposed new classifiers based on the SVM (Nayak, Naik, & Behera, 2015). Of these extensions of SVM, the twin support vector machine (TwinSVM) (Jayadeva, Khemchandani, & Chandra, 2007) has received more attention from scholars in the field of SVM research. This may be due to the novel idea of TwinSVM which is doing classification using two non-parallel hyperplanes. Each of which is as close as possible to samples of its own class and far from samples of the other class. To show the central idea of TwinSVM graphically, Figure 2 shows the geometric illustration of TwinSVM classifier.

For SVMs, there exist several stable software packages and implementations such as LIBSVM (C.-C. Chang & Lin, 2011) and LIBLINEAR (Fan, Chang, Hsieh, Wang, & Lin, 2008). These packages were used to implement SVM in scikit-learn (Pedregosa et al., 2011) which is a widely-used machine learning package for Python programming language. To solve a classification problem with the SVM algorithm, one can use scikit-learn with only a few lines of code in Python.

Even though TwinSVM is a popular classification algorithm in the field of SVM research, to the best of our knowledge, there exists no free and reliable implementation with a user guide on the internet. This motivated us to develop LightTwinSVM program to
Figure 1: Geometric illustration of SVM classifier.

Figure 2: Geometric illustration of TwinSVM classifier.

help researchers, practitioners, and students build their own classifier on the basis of LightTwinSVM. Moreover, this program can be used to solve classification problems. In the next section, we present LightTwinSVM, its features, and compare it with scikit-learn’s SVM.

**LightTwinSVM**

The LightTwinSVM program is a simple and fast implementation of the TwinSVM classifier. It is mostly written in Python and its main design goals are simplicity and speed. Also, this program is free, open source, and licensed under the terms of GNU GPL v3¹. LightTwinSVM is built on top of NumPy (Walt, Colbert, & Varoquaux, 2011), scikit-learn (Pedregosa et al., 2011), and pandas (McKinney, 2011).

LightTwinSVM program can be used by both researchers in the field of SVM research and by students in courses on pattern recognition and machine learning. Moreover, this software can be applied to a wide variety of research applications such as text classification, image or video recognition, medical diagnosis, and bioinformatics. For example, Light TwinSVM was used for the numerical experiments in our previous research paper (Mir & Nasiri, 2018).

The main features of the LightTwinSVM program are the following:

- To make its usage simple, a command-line application was created to help users solve classification tasks step-by-step.
- Since speed is one of the design goals, the clipDCD optimization algorithm (Peng, Chen, & Kong, 2014) is employed which is a simple and fast external optimizer. It was improved and implemented in C++.
- In order to solve linear or non-linear classification problems, both linear and RBF kernels are supported.
- Multi-class classification problems can be solved using either One-vs-One or One-vs-All scheme.
- The One-vs-One classifier is scikit-learn compatible. Therefore, scikit-learn tools such as GridSearchCV and cross_val_score can be employed.
- To evaluate the performance of the classifier, K-fold cross-validation and train/test split are supported.
- The optimal values of hyper-parameters can be found with grid search.
- CSV and LIBSVM formats are supported for importing datasets.
- Detailed classification results are saved in a spreadsheet file so that results can be analyzed and interpreted.

The source code of LightTwinSVM, its installation guide and usage example can be found at https://github.com/mir-am/LightTwinSVM.

**Numerical Experiments**

In this section, we conducted experiments to show the efficiency of LightTwinSVM program, and compared it with the implementation of SVM in scikit-learn on the UCI² benchmark datasets. To evaluate the classifiers’ performance, 5-fold cross-validation is used. For both standard SVM and TwinSVM, the penalty parameter $C$ was selected

¹https://opensource.org/licenses/GPL-3.0
²http://archive.ics.uci.edu/ml/datasets.html
from the set \( \{2^i \mid i = -10, -9, \ldots, 5\} \). Moreover, the RBF kernel was used and its parameter \( \gamma \) was chosen over the range \( \{2^i \mid i = -14, \ldots, 5\} \). Since the classification performance of standard SVM and TwinSVM depends heavily on the optimal choice of hyper-parameters, grid search is used to find the optimal values of hyper-parameters.

To analyze the classification performance of LightTwinSVM and scikit-learn’s SVM, the results on benchmark datasets are summarized in Table 1.

**Table 1:** The accuracy comparison between LightTwinSVM and scikit-learn’s SVM

<table>
<thead>
<tr>
<th>Datasets</th>
<th>LightTwinSVM</th>
<th>scikit-learn’s SVM</th>
<th>Difference in Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pima-Indian</td>
<td>78.91±3.73</td>
<td>78.26±2.62</td>
<td>0.65</td>
</tr>
<tr>
<td>Australian</td>
<td>87.25±2.27</td>
<td>86.81±3.22</td>
<td>0.44</td>
</tr>
<tr>
<td>Haberman</td>
<td>76.12±4.79</td>
<td>76.80±2.68</td>
<td>-0.68</td>
</tr>
<tr>
<td>Cleveland</td>
<td>85.14±5.45</td>
<td>84.82±4.04</td>
<td>0.32</td>
</tr>
<tr>
<td>Sonar</td>
<td>84.62±4.89</td>
<td>64.42±6.81</td>
<td>20.2</td>
</tr>
<tr>
<td>Heart-Statlog</td>
<td>85.56±2.96</td>
<td>85.19±2.62</td>
<td>0.37</td>
</tr>
<tr>
<td>Hepatitis</td>
<td>86.45±5.16</td>
<td>83.23±3.55</td>
<td>3.22</td>
</tr>
<tr>
<td>WDBC</td>
<td>98.24±1.36</td>
<td>98.07±0.85</td>
<td>0.17</td>
</tr>
<tr>
<td>Spectf</td>
<td>81.68±5.35</td>
<td>79.78±0.19</td>
<td>1.9</td>
</tr>
<tr>
<td>Titanic</td>
<td>81.93±2.59</td>
<td>82.27±1.83</td>
<td>-0.34</td>
</tr>
<tr>
<td>Mean Accuracy</td>
<td>84.59</td>
<td>81.94</td>
<td>2.65</td>
</tr>
</tbody>
</table>

From the Table 1, it can be seen that LightTwinSVM outperforms scikit-learn’s SVM on most datasets. For instance, the accuracy difference in Sonar dataset is as high as 20.2% which is a significant result. However, in consideration of the mean accuracy, one may notice that the difference in accuracy between the two classifiers is not very large. To show whether a significant difference exists, statistical tests are often used in research papers on classification (Demšar, 2006). Due to the limited space, comprehensive experiments with statistical tests are skipped in this paper. In summary, the experiment indicates that the LightTwinSVM program can be used for classification tasks and it may produce a better prediction accuracy.

**Acknowledgments**

This research work was carried out at the machine learning lab of IranDoc Institution. We would like to thank the director of IranDoc Institution for providing us with research facilities.

**References**


