CMPE 462: Machine Learning

PROJECT 2

Implementing an SVM Classifier

Prepared By:

Fish & Fishers

Burak Ömür – 2016400186

Hasan Ramazan Yurt – 2016400078

Ömer Faruk Deniz – 2016400003

Overview

In this project, we are expected to implement SVM by using a sofware package called LIBSVM. Also, a .mat dataset is provided for training. There are some variables used throughout the project: Train data is loaded to array X_train, train label is loaded to array y_train, test data is loaded array X_test, and train label is loaded to y_test.

Task 1

In this task, we first import svmutil package, then define a problem by using svm_problem. To train the data, we used svm_train with parameters "-t 0 -c 1e8". -t means kernel type and 0 means linear kernel type. Therefore, data is trained by linear kernel. In order to train a hard margin linear SVM, we chose C value as 1e8. By using the labels, we run predict method. Here are the results of our prediction:

```
Accuracy = 74.6667% (112/150) (classification) ← Train
Accuracy = 77.5% (93/120) (classification)     ← Test
```

Task 2

Actually, this task is very similar to first one in the implementation side. Unlike the first, we u sed different kernel functions denoted as kernel_funtion in the code. In addition, because it is a soft margin SVM there is a parameter C. We use 4 kernel funtions (linear, polynomial, rbf, s igmoid) with different C values. Here is the table that we obtained:

	linear	sv_l	polynomial	sv_p	rbf	sv_r	sigmoid	sv_s
C-Values								
0.01	84.1667	118	58.3333	141	58.3333	140	58.3333	140
0.16	85.0000	69	58.3333	141	84.1667	117	84.1667	115
0.31	83.3333	67	84.1667	141	84.1667	103	85.0000	98
0.46	85.0000	63	79.1667	134	83.3333	92	84.1667	89
...	...	...	...	...	...	...	...	...
9.31	81.6667	51	80.8333	85	77.5000	74	81.6667	56
9.46	81.6667	51	80.8333	85	77.5000	74	81.6667	56
9.61	81.6667	51	80.8333	85	77.5000	74	80.8333	56
9.76	81.6667	51	80.8333	85	77.5000	74	81.6667	57
9.91	81.6667	51	80.8333	84	77.5000	74	84.1667	55

68 rows × 8 columns

Task 3

- In theory, as the value of C increases, the number of support vectors decrease, and accuracy of the model up to a c-value.

- We know that if C is very high, model can turn into hard margin svm, if C is very low, model can ignore the error term and accuracy can be low.
- The decrease in support vectors means that there are less vectors that are supporting the margin. This can be because of we apply some regularization called soft margin, we ignore some support vectors and increase the fatness. Therefore, the number of supporting vectors decrease.
- The decrease in support vector can be seen clearly from previous figure, which is said also in theory, so correlation make our work right.
    - With polynomial, rbf and sigmoid we see many errors with small C value. As C increases, they reach a top point and then start to decrease again. This is because it went to hard margin svm and lost some regularization

Task 4

In this task, we construct the hyperplane by using support vectors. Then, we deleted a support vector and trained data again. We repeat this process for one data point that is not a support vector. In the case of support vector, we realized that number of support vectors is changed. Thus, the hyperplane is changed. More specifically, in the code there were 58 support vectors. After deletion, this number is decreased to 55. In contrast, when we delete a data point that is not a support vector, the number of support vectors and shape of the hyperplane did not change.

Bonus Task

In this task, we changed the representation of our problem fitting to the one in CVXOPT documentation given below:

$$\min_{\alpha} \quad \frac{1}{2}\alpha^T H\alpha - 1^T$$
$$s.t. \ -\alpha_i \leq 0$$
$$s.t. \ y^T\alpha = 0$$

Then we run the QPSolver with by assigning necessary parameters to the solver. Since this is more of a format conversion task rather than an algorithmic problem, we had to use a code template that was taken from the documentation and used everywhere. We got significant help to initialize and run the QPSolver from the following tutorial: https://xavierbourretsicotte.github.io/SVM_implementation.html

After running the QPSolver, we got the test accuracy as `50%`.