MACHINE LEARNING

SUPPORT VECTOR MACHINE

October 12, 2018

NAME : HIMANSHU
ROLL NO: 16MA20020
Assignment Number 5

Contents

0.1 METHODOLOGY

SVM is one of the state of the art among supervised learning algorithms. How good this classifier works depend upon various parameters. In this report we compare different values of parameter C (described below) in different types of kernels (**Linear**, **Quadratic** and **RBF**) for classification of Spam base data-set.

0.1.1 Theory

Our training sample should be divided into two classes. Let us consider a binary class with only two features. We can represent each of the training sample as a point in a 2D plane. Then they can be divided by a line if the given sample is linearly separable. Moreover we can also learn non-linear boundaries by using kernel tricks. A kernel basically projects the given data to a higher dimensional space where it is linearly separable. In 3D samples can be divided by a plane. Actually what has been shown above in 2D can easily be generalized for any finite dimension. Samples in 'n' dimensional space (Each sample has n features) can then be separated using 'n-1' dimensional hyper-plane.
The aim is to find the best hyper-plane which classifies our training samples correctly and have the maximum possible margin. Margin is the maximum possible distance between the hyper-plane and nearest point. For any hyper-plane we can rather draw two more parallel lines passing through the two most nearest points on either side of it. We find that finding the maximum margin translates into finding the maximum possible distance between those two lines. So, we can start with taking all the possible parallel lines and take that pair which classifies the entire training examples and has the largest distance between them. We call this Primal problem, as shown below.

$$Min\,\frac{1}{2}\,\|w\|^2 \quad \text{such that} \quad y_i(wX_i+b)\ \geq\ 1$$

Above problem has a complex constraint and smaller problem, which is computationally expensive to solve as it is. We have following dual problem which has a simpler constraint.This problem is used to train the classifier and kernel trick is applied as mentioned above to get the classification using non-linear boundaries.

$$min_w, bL = \|w\|^2\ -\ \sum \lambda(y_i(wX_i+b)-1)$$

0.1.2 API

PACKAGE : **Scikit Learn**

Svm is one of the many classifiers provided in scikit learn. It resides as :

```
sklearn.svm.SVC()
```

Parameters:

C

It is the regularization parameter used to select the optimum value of margin. Higher value of C means choosing smaller margin hyper-plane and so get larger accuracy on training set. Lower value of C means larger margin hyper-plane and this means greater miss-classification of training data by the hyper-plane.

Kernel

This specifies which kernel to be used in the algorithm. Provided inbuilt kernel types in this package are "linear", "poly", "rbf" and "sigmoid".

Degree

The degree of polynomial in poly kernel. It is ignored in rest of the kernels.

Gamma

Kernel coefficient for "rbf", "poly" and "sigmoid".

max_iter

It is the maximum iteration to allow in the solver. -1 means no limit. It has been fixed to 10^5 in all the experiment.

0.2 Experimental Results

For comparison purpose different values of C has been tested using same other constraints as provided in each type of kernel.

0.2.1 Linear Kernel

maximum iteration	10^5
kernel	linear

Table 1: Training and testing accuracy of **linear kernel** using different values of C

Value of C	Training Accuracy	Testing Accuracy
2^{-7}	0.897515	0.89427
2^{-6}	0.909316	0.903692
2^{-5}	0.917081	0.915278
2^{-4}	0.922981	0.918899
2^{-3}	0.927018	0.916727
2^{-2}	0.931055	0.923968
2^{-1}	0.932608	**0.926864**
2^{0}	0.934472	**0.926140**
2^{1}	0.935403	0.924692
2^{2}	0.935714	0.924692
$\mathbf{2^{3}}$	**0.935403**	**0.927589**
2^{4}	0.933229	0.923244
2^{5}	0.935714	**0.926864**
2^{6}	0.930124	0.923244
2^{7}	0.799068	0.792179
2^{8}	0.643167	0.650253
2^{9}	0.668012	0.653874
2^{10}	0.623291	0.623461
2^{11}	0.766770	0.7682839
2^{12}	0.584472	0.586531
2^{13}	0.724534	0.727733
2^{14}	0.713043	0.711078
2^{15}	0.642236	0.650977
2^{16}	0.696583	0.692251
2^{17}	0.661801	0.667632

In above table we clearly see that with very low values of C, accuracy increases with increase in C. It achieves best value w.r.t test accuracy around $\mathbf{C = 2^3}$.

0.2.2 Quadratic Kernel

maximum iteration	10^5
kernel	poly
degree	2
gamma	auto

Table 2: Training and testing accuracy of **poly kernel** using different values of C

Value of C	Training Accuracy	Testing Accuracy
2^{8}	0.620186	0.585083
2^{9}	0.628260	0.588703
2^{10}	0.670496	0.626357
2^{11}	0.704347	0.670528
2^{12}	0.743167	0.709631
2^{13}	0.776397	0.753802
2^{14}	0.804347	0.782042
2^{15}	0.820497	0.803765
2^{16}	0.839751	0.824040
2^{17}	0.855590	0.838522
2^{18}	0.876708	0.857349
2^{19}	0.891925	0.870383
2^{20}	0.906832	0.875452
2^{21}	0.922670	0.885590
2^{22}	0.938509	0.892107
2^{23}	0.952484	0.905865
$\mathbf{2^{24}}$	**0.959627**	**0.913106**
$\mathbf{2^{25}}$	**0.968012**	**0.913106**
2^{26}	0.974223	0.910934
2^{27}	0.979813	0.910210
2^{28}	0.984472	0.905141
2^{29}	0.986956	0.902969
2^{30}	0.987577	0.901520
2^{31}	0.985714	0.895004
2^{32}	0.980124	0.886314

In above table for Quadratic kernel best possible value of accuracy is achieved for $\mathbf{C} = \mathbf{2^{25}}$.

0.2.3 RBF Kernel

maximum iteration	10^5
kernel	rbf
gamma	auto

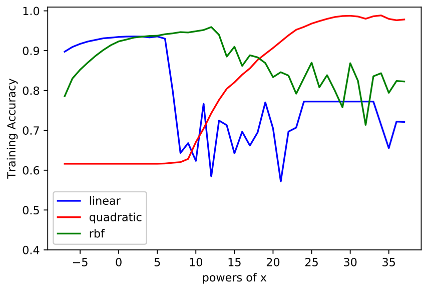
Table 3: Training and testing accuracy of **RBF kernel** using different values of C

Value of C	Training Accuracy	Testing Accuracy
2^{-7}	0.785714	0.772628
2^{-6}	0.830124	0.822592
2^{-5}	0.852173	0.852281
2^{-4}	0.870186	0.872556
2^{-3}	0.886956	0.888486
2^{-2}	0.901552	0.896452
2^{-1}	0.913975	0.910209
2^{0}	0.923291	0.921072
2^{1}	0.927950	0.925416
2^{2}	0.932919	0.929761
2^{3}	0.935093	0.931933
2^{4}	0.936956	0.934105
2^{5}	0.937577	0.935553
2^{6}	0.941304	0.937726
2^{7}	0.943478	**0.941346**
2^{8}	0.946583	**0.940623**
2^{9}	0.945652	**0.940623**
$\mathbf{2^{10}}$	**0.948757**	**0.946415**
2^{11}	0.951863	0.939899
2^{12}	0.959316	**0.944243**
2^{13}	0.939751	0.931209
2^{14}	0.885093	0.873280
2^{15}	0.909937	0.905141
2^{16}	0.861801	0.862418
2^{17}	0.888509	0.860970

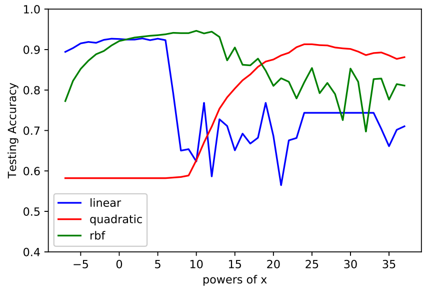
In above table of accuracy for different value of C best accuracy is achieved when $\mathbf{C = 2^{10}}$.

Table 4: Comparison of best C in different kernels

Type of kernel	C value for best accuracy	Train accuracy	Test accuracy
Linear	2^3	0.935403	0.927589
Quadratic	2^{25}	0.968012	0.913106
RBF	2^{10}	0.948757	0.946415

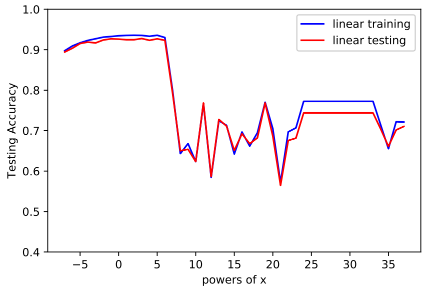


(a) Training Accuracy

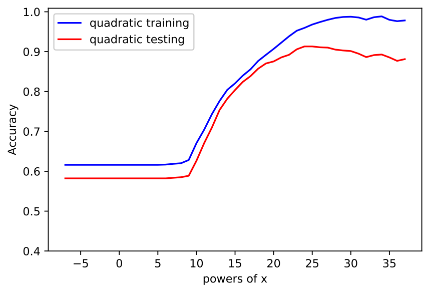


(b) Testing Accuracy

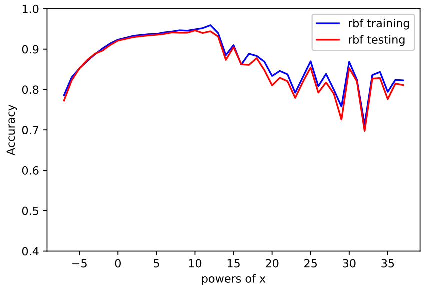
Figure 1: Comparison different kernels for training and testing accuracy respectively



(a) Linear Kernel



(b) Quadratic kernel



(c) RBF kernel

Figure 2: Comparison training and testing errors of each kernel