Pattern Recognition and Machine Learning
Report for : Lab 10

Name : Khushi Parikh
Roll No : B20EE029

Methodology

- Support Vector Machine (SVM) is a type of unsupervised learning algorithm. It basically creates a **decision boundary** that separates an n-dimensional space into divisions such that each division represents a certain class.
- Say we are taking a linear line to separate two classes. The two closest points, one on each side of the hyperplane are known as **support vectors** and these affect the position of the plane. It tries to **maximize the distance (margin)** between itself and these two points, the output hyperplane is known as the optimal hyperplane. This is done by the **linear kernel.**
- Similarly, the **polynomial** kernel creates a decision boundary that is **non-linear.**
- The third kernel is **RBF(radial basis function)**. It's formula for distance between two points, σ here is variance -

$$K(X_1, X_2) = exp(-\frac{||X_1 - X_2||^2}{2\sigma^2})$$

  RBF is the default kernel in sklearn svc function.
- An SVM kernel transforms the input data to its desired form. A major advantage of SVM is that it is **efficient in high dimensional spaces.**
- To implement SVM, I have used the **sklearn library.**

- The data was pre-processed and no null values were found. It was scaled using the standard scaler and was split into train(70%) and test(30%) sets.

Accuracies

- Linear Kernel : `Accuracy : 93.48298334540188`
- Quadratic Kernel : `Accuracy : 83.7798696596669`
- RBF Kernel : `Accuracy : 92.97610427226647`

Generalization Constant C

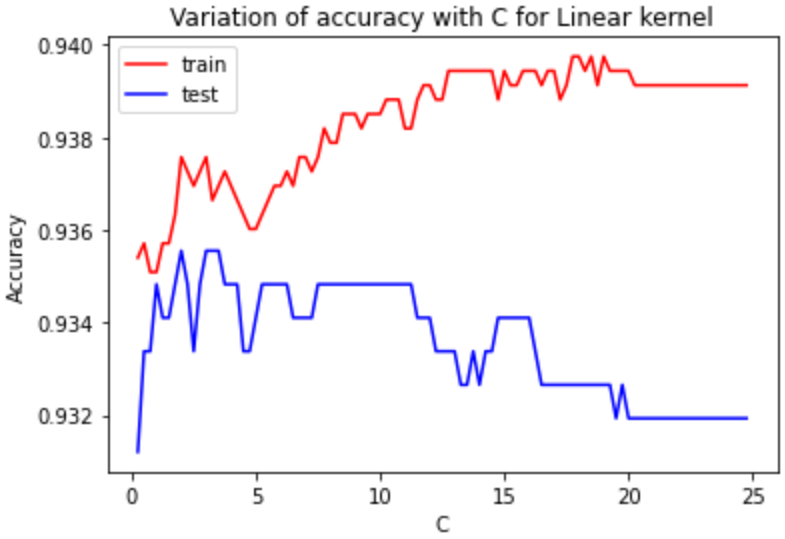
- The C parameter of an SVM classification shows how much you want to avoid misclassifying each training example.
- For large values of C, the optimization will choose a smaller-margin hyperplane if that hyperplane does a better job of accurately classifying all the training points.
- Conversely, a very small value of C will encourage the optimizer to look for a larger-margin separating hyperplane, even if that hyperplane misclassifies more points.

Finding optimal value of C based on test(majorly) and train accuracy :

- The first column is the training accuracy and the second is the testing accuracy.

- **Linear Kernel**

```
0.25 :   0.9354037267080745      0.9312092686459088
0.5 :    0.9357142857142857      0.9333816075307748
0.75 :   0.9350931677018633      0.9333816075307748
1.0 :    0.9350931677018633      0.9348298334540188
1.25 :   0.9357142857142857      0.9341057204923968
1.5 :    0.9357142857142857      0.9341057204923968
1.75 :   0.9363354037267081      0.9348298334540188
2.0 :    0.9375776397515528      0.9355539464156408
2.25 :   0.9372670807453416      0.9348298334540188
2.5 :    0.9369565217391305      0.9333816075307748
2.75 :   0.9372670807453416      0.9348298334540188
3.0 :    0.9375776397515528      0.9355539464156408
3.25 :   0.9366459627329192      0.9355539464156408
3.5 :    0.9369565217391305      0.9355539464156408
3.75 :   0.9372670807453416      0.9348298334540188
```

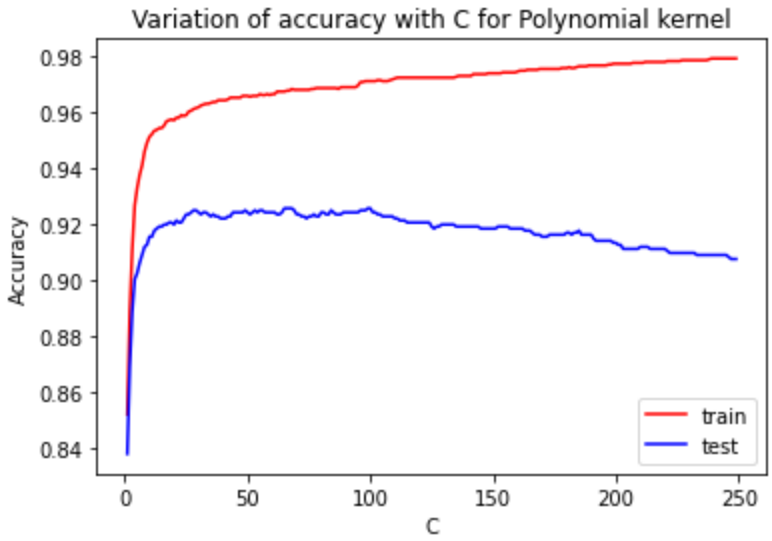


For C=10.5,

```
TRAINING SET
[[1886   80]
 [ 117 1137]]
Accuracy : 93.88198757763975
------------------------------
TESTING SET
[[792  30]
 [ 60 499]]
Accuracy : 93.48298334540188
```

- **Quadratic Kernel**

```
1 :     0.8518633540372671     0.8377986965966691
2 :     0.8888198757763975     0.8689355539464156
3 :     0.9108695652173913     0.88848660391021
4 :     0.9263975155279504     0.9000724112961622
5 :     0.9326086956521739     0.9022447501810282
6 :     0.9372670807453416     0.9058653149891384
7 :     0.9406832298136646     0.9087617668356264
8 :     0.9456521739130435     0.9116582186821144
9 :     0.9487577639751553     0.9123823316437364
10 :    0.9509316770186336     0.9152787834902245
11 :    0.9518633540372671     0.9152787834902245
12 :    0.9531055900621118     0.9174511223750905
13 :    0.953416149068323      0.9181752353367125
14 :    0.9540372670807453     0.9188993482983345
15 :    0.9540372670807453     0.9188993482983345
16 :    0.9549689440993789     0.9196234612599565
17 :    0.9565217391304348     0.9196234612599565
18 :    0.9568322981366459     0.9203475742215785
19 :    0.9571428571428572     0.9203475742215785
20 :    0.9568322981366459     0.9196234612599565
```

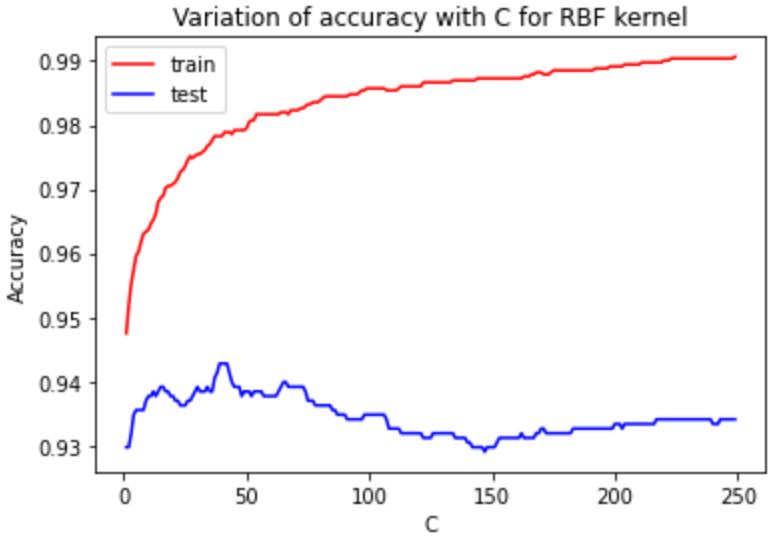


For C=98,

```
TRAINING SET
[[1950   16]
 [  78 1176]]
Accuracy : 97.0807453416149
------------------------------
TESTING SET
[[783  39]
 [ 65 494]]
Accuracy : 92.46922519913106
```

- **RBF Kernel**

```
1 :     0.9475155279503106      0.9297610427226647
2 :     0.9518633540372671      0.9297610427226647
3 :     0.9552795031055901      0.9319333816075308
4 :     0.9574534161490683      0.9348298334540188
5 :     0.9596273291925466      0.9355539464156408
6 :     0.9602484472049689      0.9355539464156408
7 :     0.9618012422360248      0.9355539464156408
8 :     0.9630434782608696      0.9355539464156408
9 :     0.9633540372670808      0.9370021723388848
10 :    0.9636645962732919      0.9377262853005068
11 :    0.9645962732919254      0.9377262853005068
12 :    0.9652173913043478      0.9384503982621288
13 :    0.9661490683229814      0.9377262853005068
14 :    0.9680124223602484      0.9384503982621288
15 :    0.9686335403726708      0.939174511223751
16 :    0.968944099378882       0.939174511223751
17 :    0.9701863354037267      0.9384503982621288
18 :    0.9704968944099379      0.9384503982621288
```



For C=40,

```
TRAINING SET
[[1949   17]
 [  53 1201]]
Accuracy : 97.82608695652173
------------------------------
TESTING SET
[[795  27]
 [ 52 507]]
Accuracy : 94.27950760318609
```

- **Heads of the output table of pandas dataframe -**

	C for Linear	Train Accuracy	Test Accuracy
0	0.25	0.935404	0.931209
1	0.50	0.935714	0.933382
2	0.75	0.935093	0.933382
3	1.00	0.935093	0.934830
4	1.25	0.935714	0.934106

	C for Quadratic	Train Accuracy	Test Accuracy
0	1	0.851863	0.837799
1	2	0.888820	0.868936
2	3	0.910870	0.888487
3	4	0.926398	0.900072
4	5	0.932609	0.902245

	C for RBF	Train Accuracy	Test Accuracy
0	1	0.947516	0.929761
1	2	0.951863	0.929761
2	3	0.955280	0.931933
3	4	0.957453	0.934830
4	5	0.959627	0.935554

Note: The values of C for linear ranged from 0.25 to 25 in intervals of 25. For quadratic and rbf kernels, it ranged from 1 to 250 during which it had started overfitting. The linear kernel had higher accuracy for smaller intervals and took a lot of time to run for higher C, aling with giving less train and test accuracy.