ELL409: Assignment 2

Anand Kumar Verma
2016EE10459

Part 1

1. Binary Classification using convex optimization (CVXOPT)

SVM dual problem and convex optimization package

$$\begin{aligned} &\min \frac{1}{2}x^T P x + q^T x \\ s.t. \quad & Gx \preceq h \\ & Ax = b \end{aligned}$$

The general steps to solve the SVM problem are the following:

- Create $\mathbf{P}$ where $H_{i,j} = y^{(i)}y^{(j)} < x^{(i)}x^{(j)} >$
- Calculate $\mathbf{w} = \sum_i^m y^{(i)}\alpha_i x^{(i)}$
- Determine the set of support vectors S by finding the indices such that $\alpha_i > 0$
- Calculate the intercept term using $b = y^{(s)} - \sum_{m \in S} \alpha_m y^{(m)} < x^{(m)}x^{(s)} >$
- For each new point x' classify according to $y' = sign(w^T x' + b)$

Relevant code for fitting using convex optimisation -

```
def cvx_fit(C,X,y) :

    m,n = X.shape
    y = y.reshape(-1,1) * 1.
    X_dash = y * X
    H = np.dot(X_dash , X_dash.T) * 1.


    P = matrix(H)
    q = matrix(-np.ones((m, 1)))
    G = matrix(np.vstack((np.eye(m)*-1,np.eye(m))))
    h = matrix(np.hstack((np.zeros(m), np.ones(m) * C)))
    A = matrix(y.reshape(1, -1))
    b = matrix(np.zeros(1))

    solvers.options['show_progress'] = False
    sol = solvers.qp(P, q, G, h, A, b)
    alphas = np.array(sol['x'])

    return alphas
```

The code which takes the CVX output and uses it to construct the actual classifier to be run on test data :

```
w = np.sum(alphas * y[:, None] * x, axis = 0)
cond = (alphas > 1e-4).reshape(-1)
b = y[cond] - np.dot(x[cond], w)
```

Linear Classifier

Class A = 0

Class B = 1

C = 1

CVX OPT results:

w = [0.50903405 0.2151743 -0.07727033 0.06086342 0.0815154 -0.06419666
-0.07647347 0.0908558 -0.04182757 0.143932 0.28851191 0.16998405
0.01984412 -0.02207359 -0.26463624 0.12112439 0.11990436 -0.18536296
-0.10984664 -0.03110624 -0.00899966 0.10722886 -0.01574708 -0.11689553
-0.12377568]

b = 0.1450705

train accuracy = 1

test accuracy = 1

LIBSVM results:

w = [[0.50891035 0.21537839 -0.07721695 0.06081666 0.08153991 -0.06415033
-0.07644515 0.09095017 -0.04190984 0.14413728 0.28852729 0.16997611
0.01967119 -0.02200081 -0.26477857 0.12103369 0.11999122 -0.18536999
-0.10968765 -0.03099504 -0.00902105 0.10743674 -0.01565741 -0.11685295
-0.12383827]]

b = [0.14523601]

Polynomial Classifier

C = 10

Gamma = 1

CVXOPT result :

Alphas = [0.0223547 0.00532456 0.10067115 0.11565338 0.05470203 0.03265724
0.03697998 0.03342269 0.02443309 0.00291996 0.03779205 0.03806548
0.00581801 0.03597707 0.11595474]

w = [0.50903405 0.2151743 -0.07727033 0.06086342 0.0815154 -0.06419666
-0.07647347 0.0908558 -0.04182757 0.143932 0.28851191 0.16998405
0.01984412 -0.02207359 -0.26463624 0.12112439 0.11990436 -0.18536296
-0.10984664 -0.03110624 -0.00899966 0.10722886 -0.01574708 -0.11689553
-0.12377568]

b = 0.14507054446950562

2. Binary Classification

Train data set – 500

Test data set - 100

5 fold cross- validation

10 features

Linear Kernel

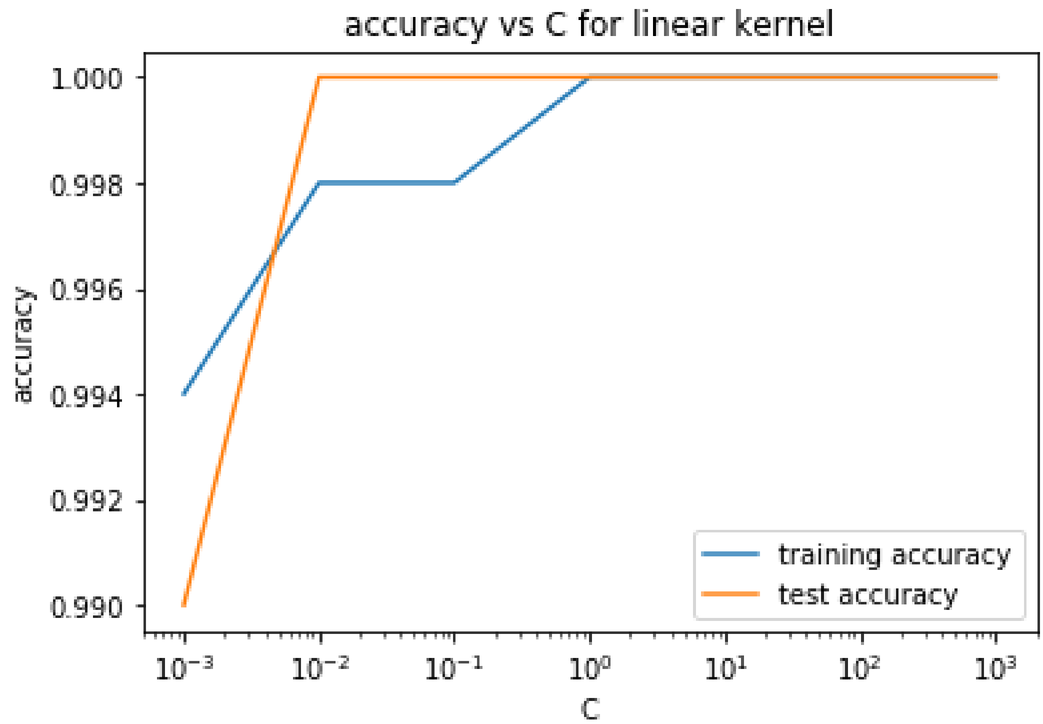
Class A = 0 & Class B = 1

Best Hyperparameter:

    'SVM__C': 1

Train Acccuracy = 1.0

Train Acccuracy = 1.0



Polynomial Kernel

Class A = 0 & Class B = 1

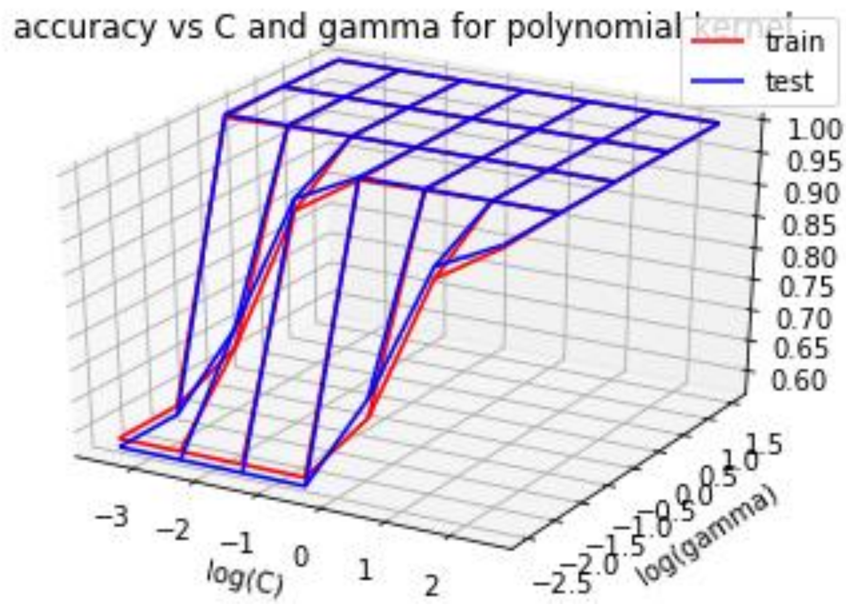
Best Hyperparameter:

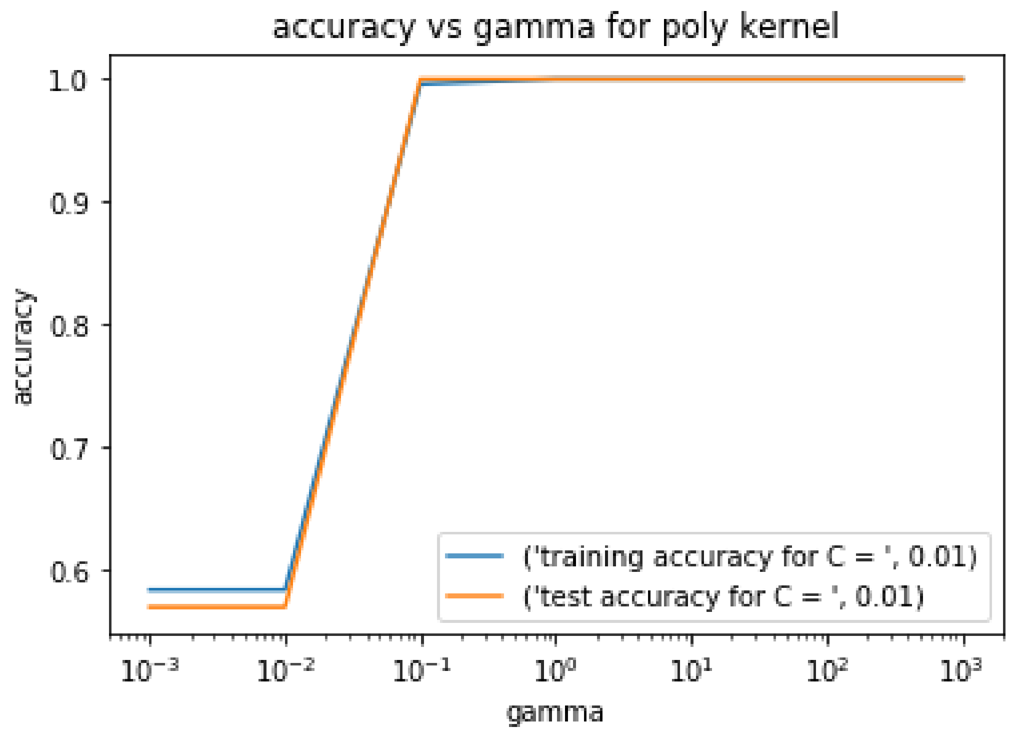
    'SVM__C': 0.01

    'SVM__gamma': 10

Train Accuracy = 1.0

Test accuracy = 0.99





Rbf kernel:

Class A = 0 & Class B = 1

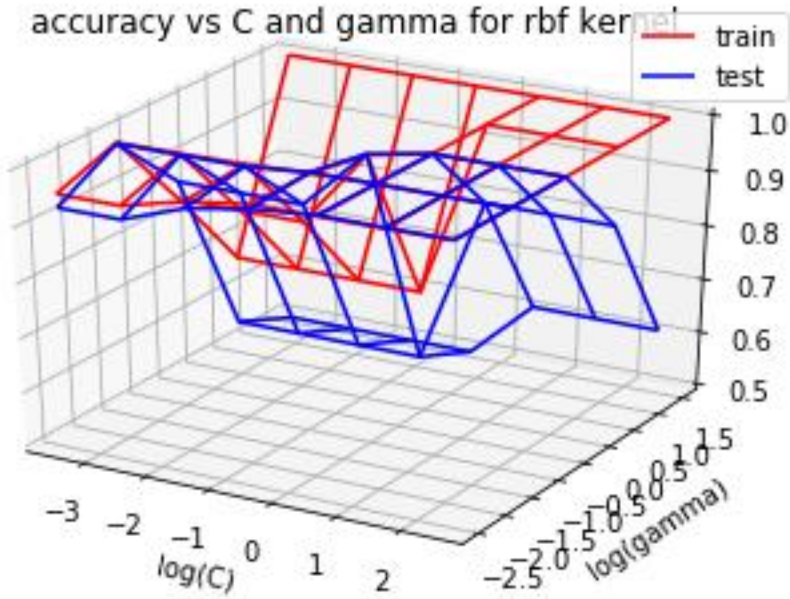
Best Hyperparameter:

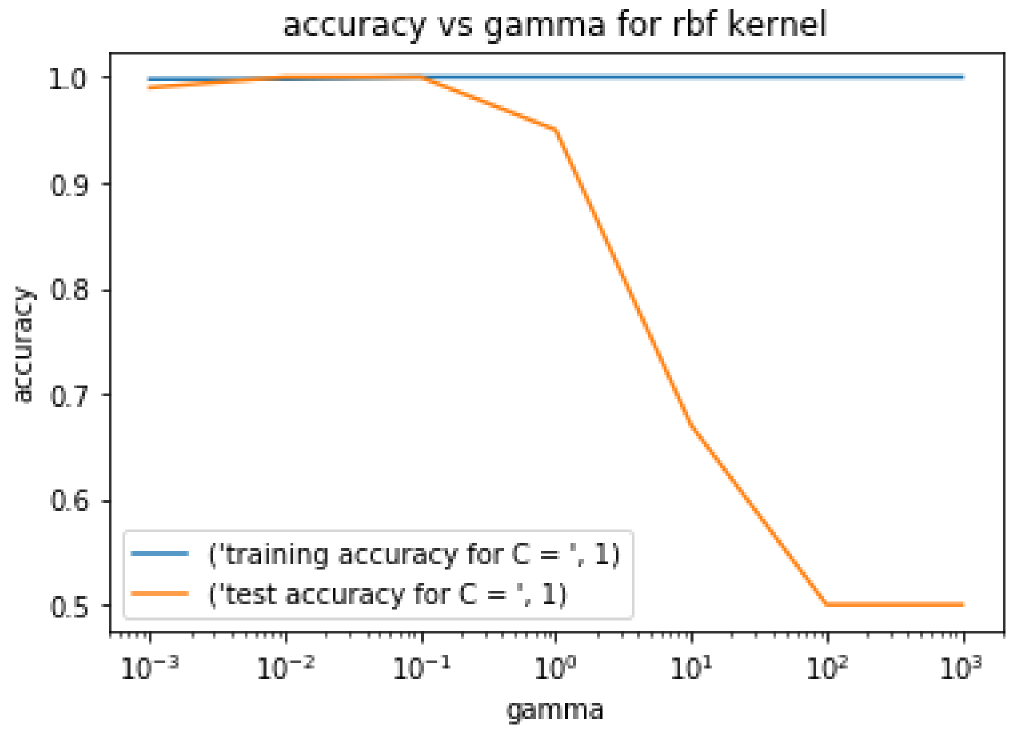
'SVM__C': 1,

'SVM__gamma': 0.01

Train Accuracy = 1.0

Test accuracy = 1.0





25 features

Linear Kernel

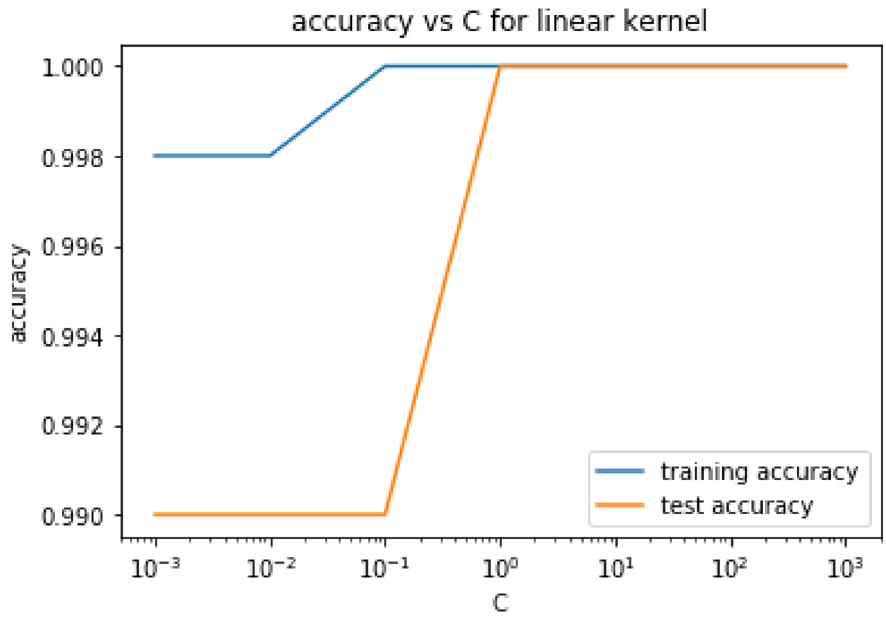
Class A = 0 & Class B = 1

Best Hyperparameter:

'SVM__C': 1

Train Acccuracy = 1.0

Train Acccuracy = 1.0



Polynomial Kernel

Class A = 0 & Class B = 1

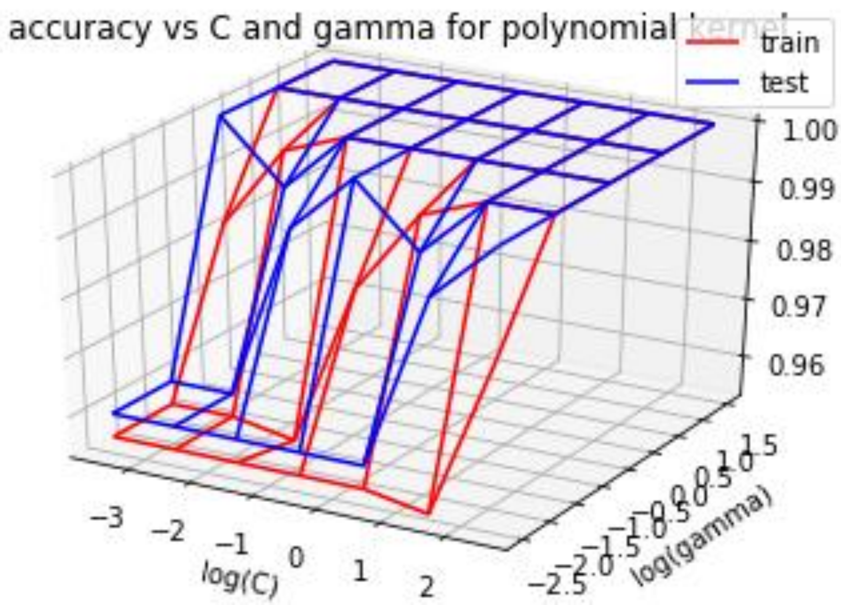
Best Hyperparameter:

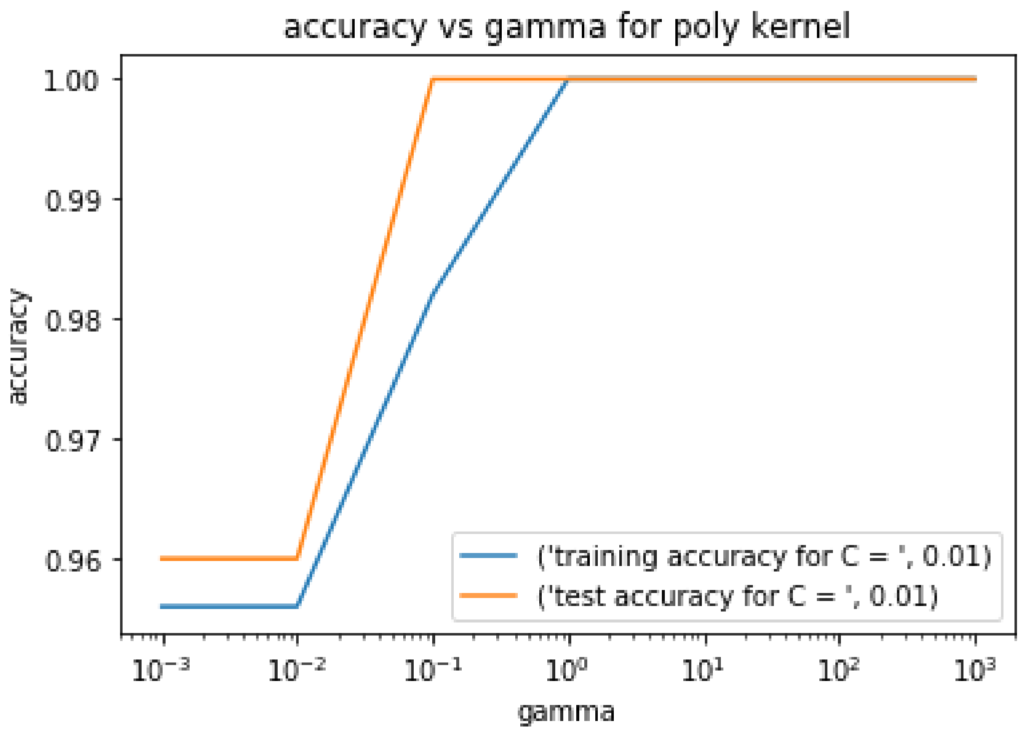
'SVM__C': 0.01

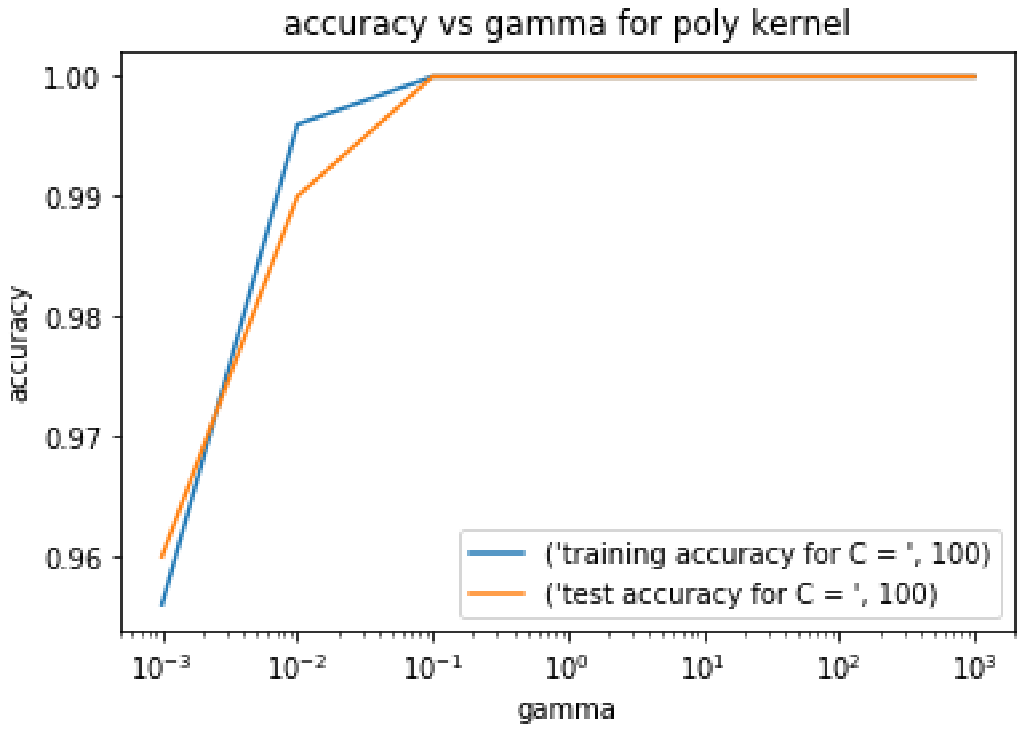
'SVM__gamma': 0.1

Train Accuracy = 1.0

Test accuracy = 1.0







Rbf kernel:

Class A = 0 & Class B = 1

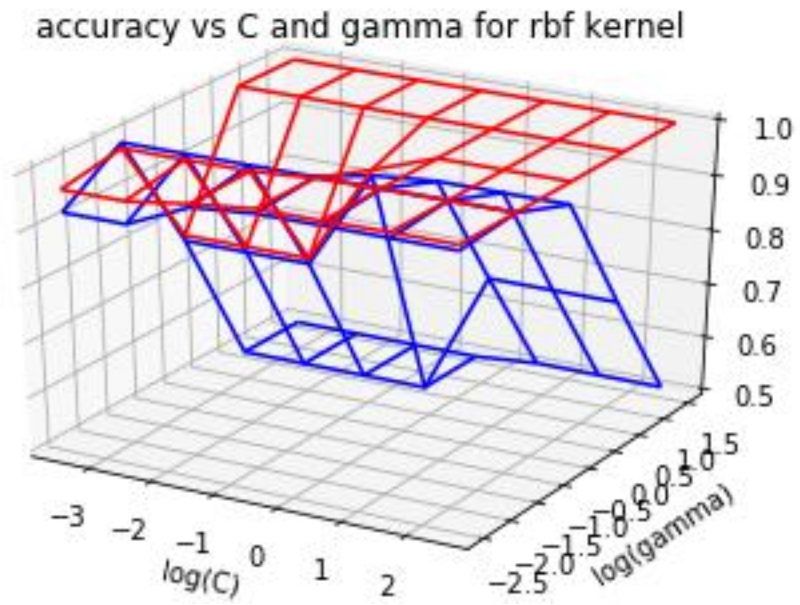
Best Hyperparameter:

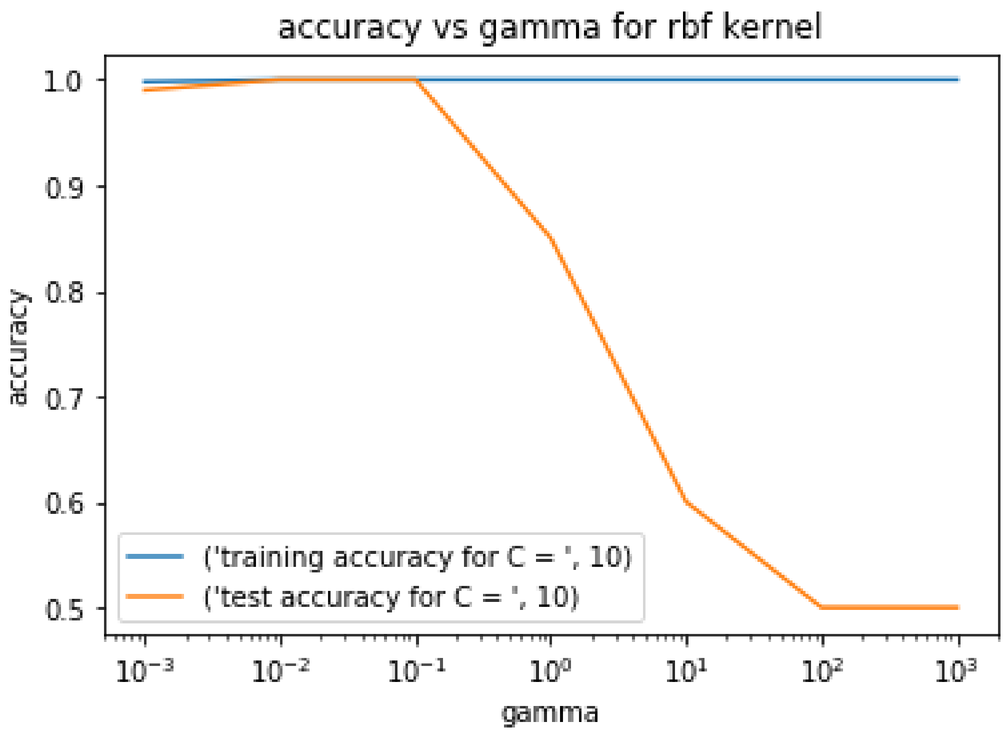
'SVM__C': 10,

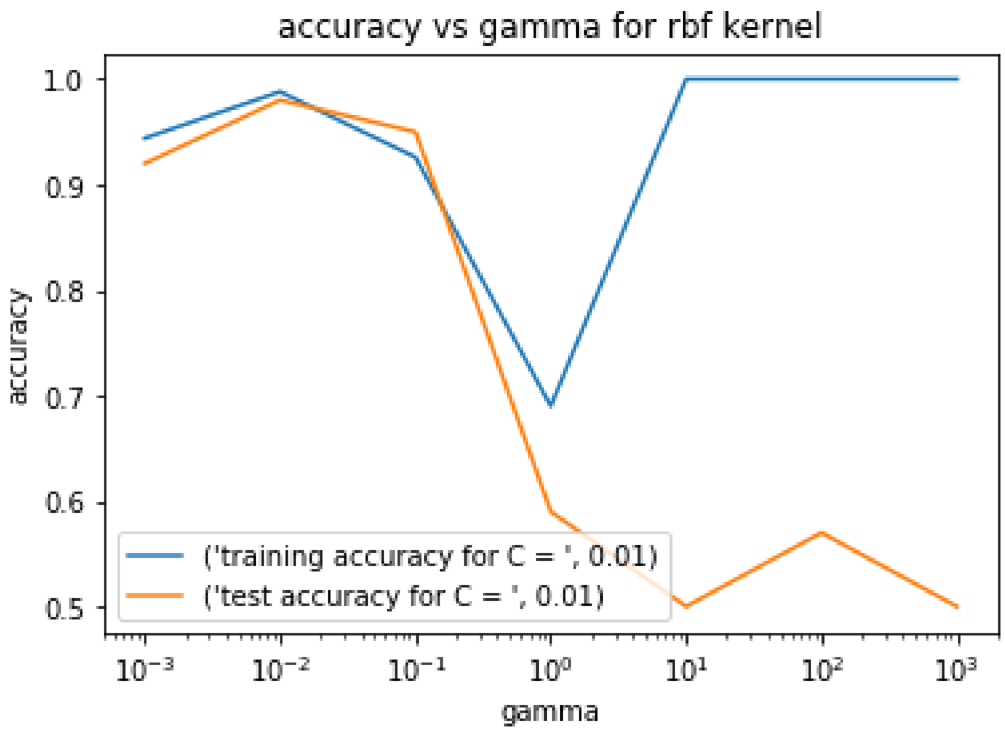
'SVM__gamma': 0.01

Train Accuracy = 1.0

Test accuracy = 1.0







Summary of Binary Classification

Train data set – 500
Test data set - 100
5 fold cross- validation
25 features

Class A	Class B	Kernel	Best C	Best Gamma	Train Accuracy	Test Accuracy
0	1	Linear	1.0		1.0	1.0
0	1	Poly	0.1	0.1	1.0	1.0
0	1	rbf	10	0.01	1.0	1.0
8	9	Linear	100		0.99	0.96
8	9	Poly	0.1	0.1	1.0	0.98
8	9	Rbf	10	0.01	1.0	0.97
3	6	Linear	1.0		1.0	1.0
3	6	Poly	0.01	0.1	1.0	0.99
3	6	rbf	1	0.1	1.0	1.0

- For binary classification of given data, both linear and non-linear kernel are equally good.
  Reason:
  No. of feature = 25

No. of class = 2

Since, feature > class. So, linear kernel will also work fine.

- Different class need different hyper parameter setting.
- Linear –
    - C increment leads to over fitting
- Polynomial –
    - C increment leads to over fitting
    - gamma increment leads to overfitting
- rbf –
    - C increment leads to over fitting
    - gamma increment leads to overfitting

3. Multiclass classification

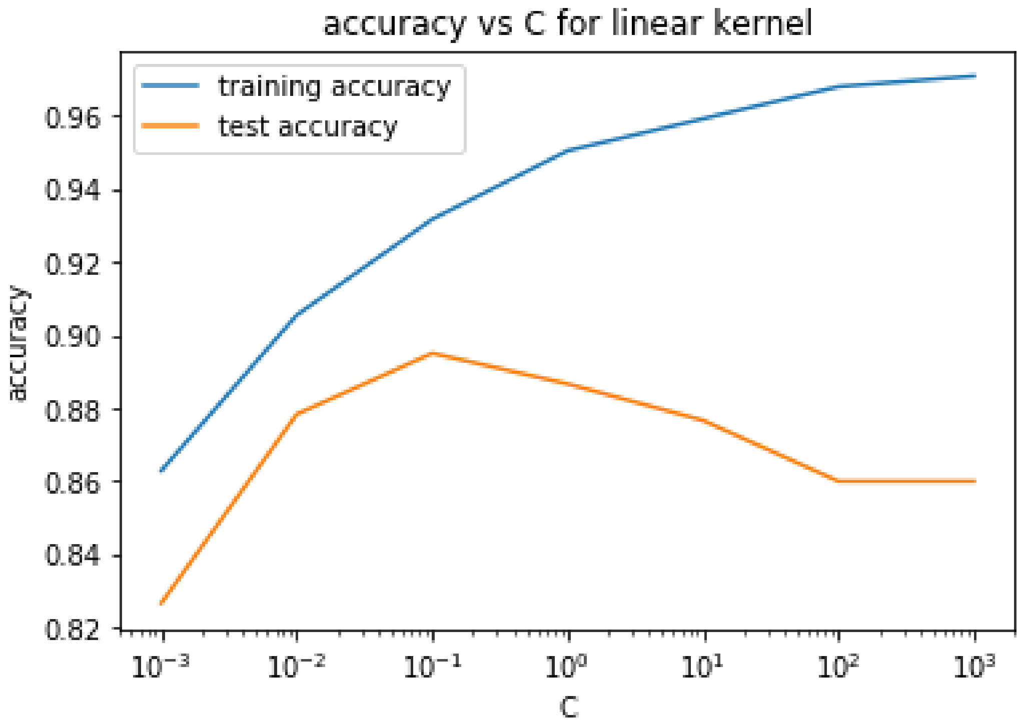
25 Features

Linear Kernel

Best Hyperparameter:
'SVM__C': 0.1,

Train Accuracy = 0.925

Test accuracy = 0.89

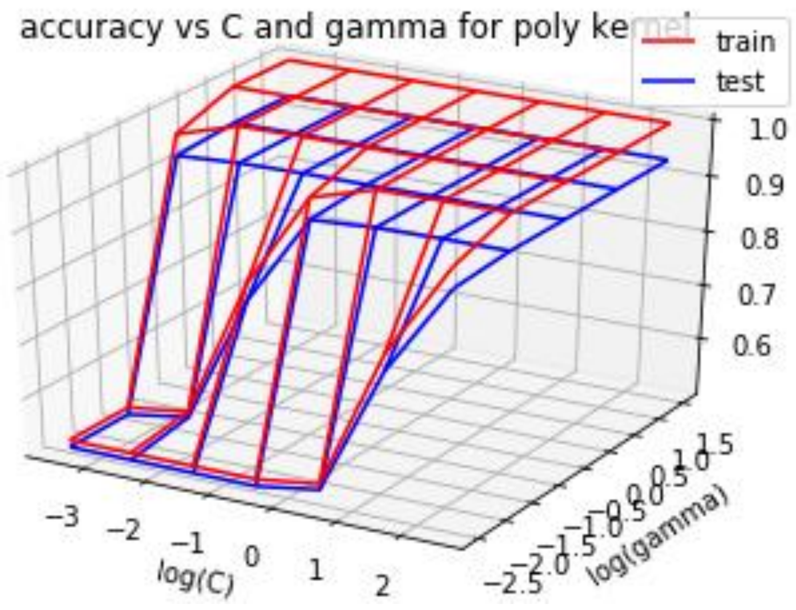


Polynomial Kernel

Best Hyperparameter:
'SVM__C': 0.01,
'SVM__gamma': 10

Train Accuracy = 1.0

Test accuracy = 0.935



RBF Kernel

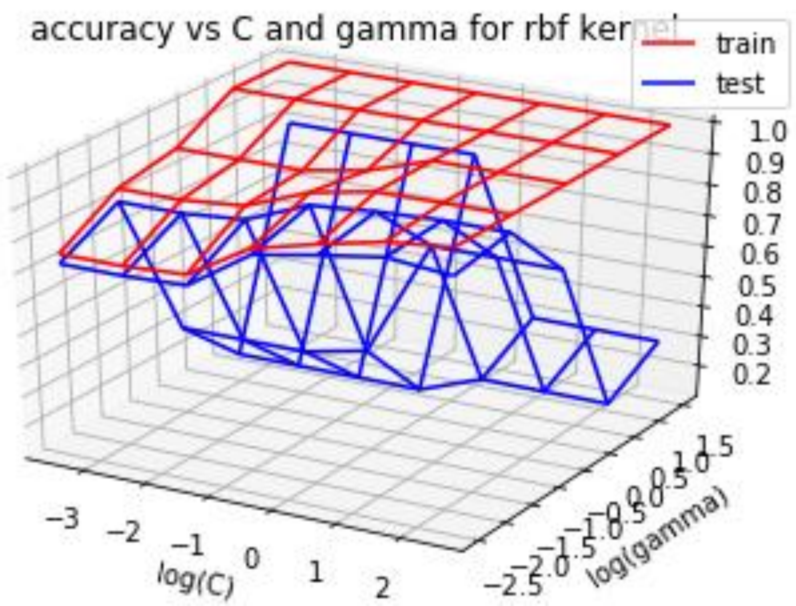
Best Hyperparameter:

'SVM__C': 10,

'SVM__gamma': 0.1

Train Accuracy = 1.0

Test accuracy = 0.9333



10 Features

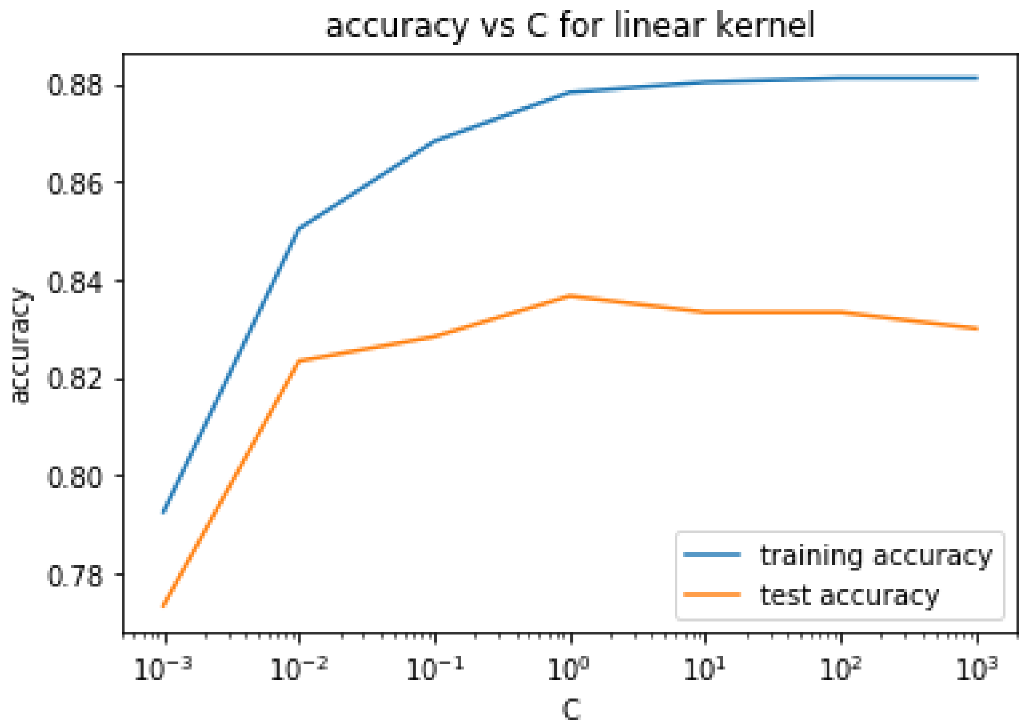
Linear Kernel

Best Hyperparameter:

'SVM__C': 1,

Train Accuracy = 0.87625

Test accuracy = 0.825



Polynomial Kernel

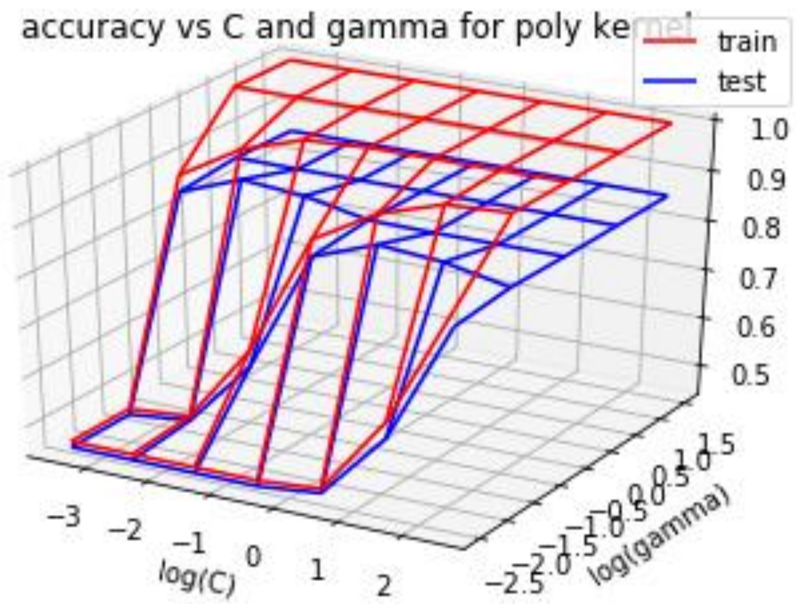
Best Hyperparameter:

'SVM__C': 1,

'SVM__gamma': 0.1

Train Accuracy = 0.9475

Test accuracy = 0.88



RBF Kernel

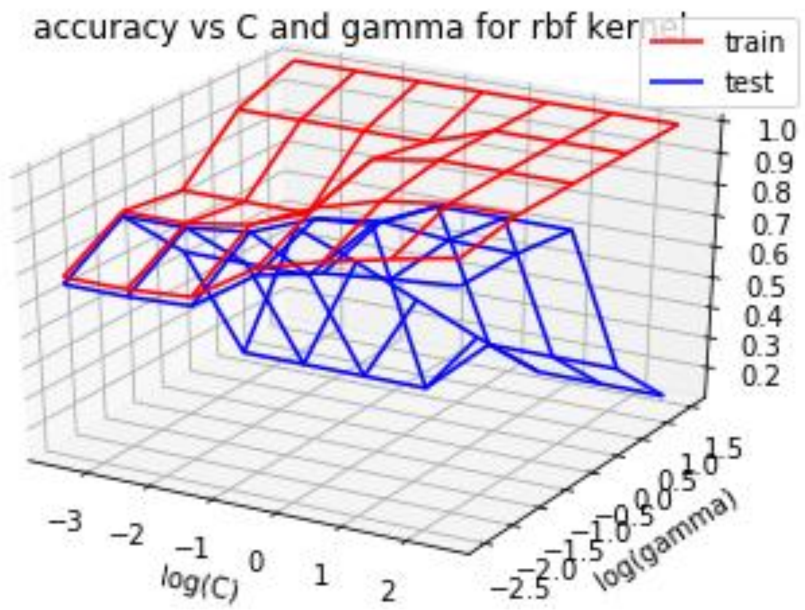
Best Hyperparameter:

'SVM__C': 10,

'SVM__gamma': 0.1

Train Accuracy = 0.9925

Test accuracy = 0.8916



- Tuned values for muti-classification are different from binary classification. As, in binary classification, hyper parameters value are dependent on the class to be classified.
- Classification using 10 features has less accuracy than one with 25 features. As no. of feature provide more dimension, so data of different classes can be classified better.

Part 2

Approach:

- Scaled down data by 1000 to make data from -10 to 10.
- Fine tuning for hyper parameters through greed search.
- Test accuracy = 0.9588