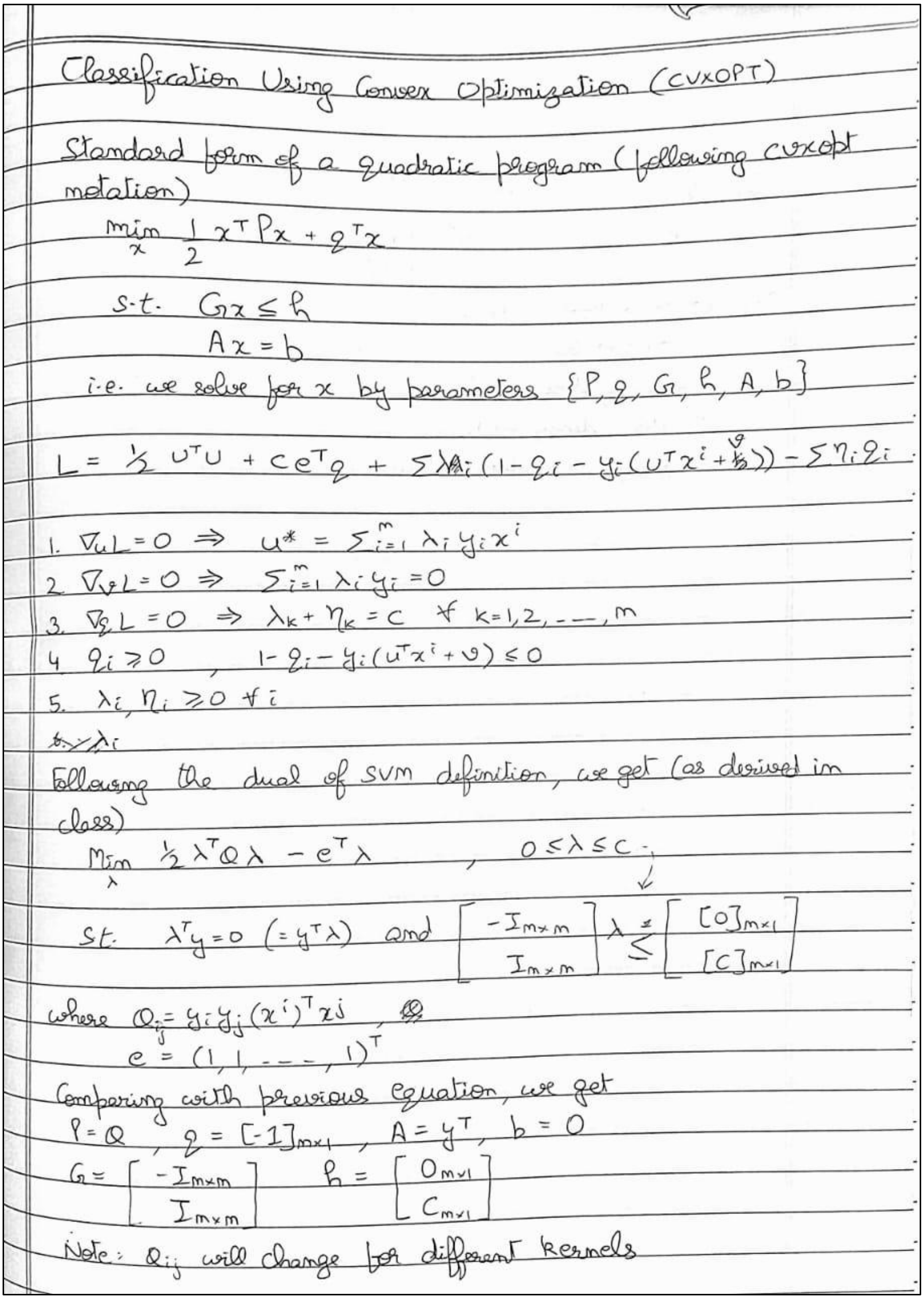
ELL409 Assignment 2

Vansh Gupta
2019EE10143

Appendix:

- CVXOPT derivations useful for the code:
  Note: Derivations for the parts derived in class have been skipped, but the final results are mentioned.



Classification Using Convex Optimization (CVXOPT)

Standard form of a quadratic program (following cvxopt notation)

$$\min_{x} \frac{1}{2} x^T P x + q^T x$$

$$\text{s.t. } Gx \leq h$$
$$Ax = b$$

i.e. we solve for x by parameters $\{P, q, G, h, A, b\}$

$$L = \tfrac{1}{2} u^T u + c e^T \xi + \sum \lambda_i (1 - \xi_i - y_i(u^T x^i + \overset{\vartheta}{\cancel{b}})) - \sum \eta_i \xi_i$$

1. $\nabla_u L = 0 \Rightarrow u^* = \sum_{i=1}^{m} \lambda_i y_i x^i$
2. $\nabla_\vartheta L = 0 \Rightarrow \sum_{i=1}^{m} \lambda_i y_i = 0$
3. $\nabla_\xi L = 0 \Rightarrow \lambda_k + \eta_k = c \quad \forall\, k = 1, 2, \ldots, m$
4. $\xi_i \geq 0 \quad , \quad 1 - \xi_i - y_i(u^T x^i + \vartheta) \leq 0$
5. $\lambda_i, \eta_i \geq 0 \; \forall i$

~~6. λ_i~~

Following the dual of SVM definition, we get (as derived in class)

$$\min_{\lambda} \tfrac{1}{2} \lambda^T Q \lambda - e^T \lambda \quad , \quad 0 \leq \lambda \leq c$$

$$\text{s.t. } \lambda^T y = 0 \; (= y^T \lambda) \text{ and } \begin{bmatrix} -I_{m\times m} \\ I_{m\times m} \end{bmatrix} \lambda \leq \begin{bmatrix} [0]_{m\times 1} \\ [c]_{m\times 1} \end{bmatrix}$$

where $Q_{ij} = y_i y_j (x^i)^T x^j$,
$e = (1, 1, \ldots, 1)^T$

Comparing with previous equation, we get

$P = Q$, $q = [-1]_{m\times 1}$, $A = y^T$, $b = 0$

$$G = \begin{bmatrix} -I_{m\times m} \\ I_{m\times m} \end{bmatrix} \quad h = \begin{bmatrix} 0_{m\times 1} \\ C_{m\times 1} \end{bmatrix}$$

Note: Q_{ij} will change for different kernels

In the expression for Q_{ij}, the $(x^i)^T x^j$ defines the kernel and this value is for the linear kernel
For liner kernels, $K(x, x') = x^T x'$

Poly : $K(x, x') = (1 + x^T x')^d$

RBF/Gaussian : $K(x, x') = \exp(-\|x - x'\|^2 / 2\sigma^2)$

In lecture we found ~~so~~ how the offset term v is given by $v = y_i - U^T x^i$
Also, for a new dataset x

$$U = \sum_{j=1}^{m} \lambda_j y_j \phi(x^j)$$

$$U^T \phi(x) + v = \sum_{j=1}^{M} \lambda_j y_j \phi(x^j)^T \phi(x^j) + v$$

$$\text{OR} \quad \overset{\text{target}}{y} = \sum_{j=1}^{m} \lambda_j y_j K(x, x^j) + v$$

Part 1A

- **Binary Classification**:
  - **Linear Kernel**
  - Note, here's my understanding of the parameter C, based on this: *The C parameter tells the SVM optimization 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 getting all the training points classified correctly. Conversely, a very small value of C will cause the optimizer to look for a larger-margin separating hyperplane, even if that hyperplane misclassifies more points. For very tiny values of C, you should get misclassified examples, often even if your training data is linearly separable.* i.e., it kind of acts as a regularization constant

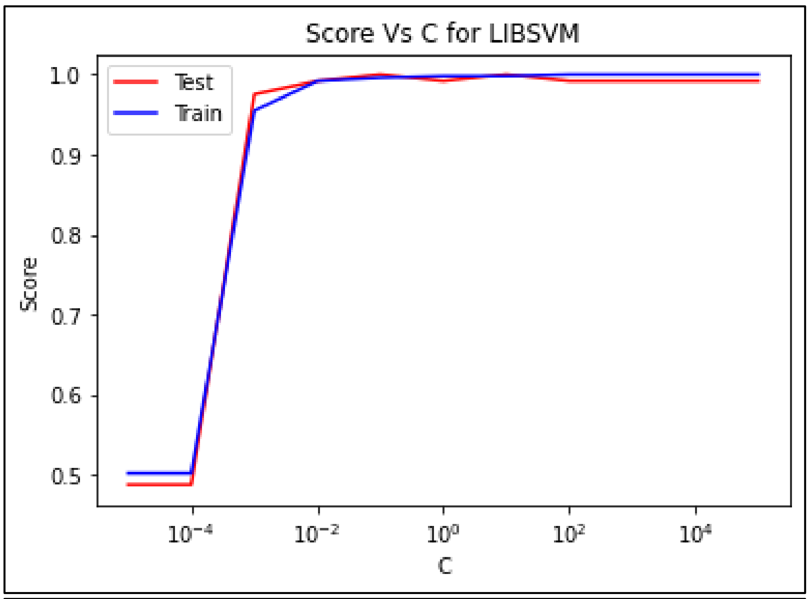
First, I take the linear kernel, for which, the only parameter is C (Because the degree is introduced in the separate poly kernel). Further, I first consider only 1st 10 features for this sub-part and then 25 features for each pair of target values. I have reported the combined results in table 1.

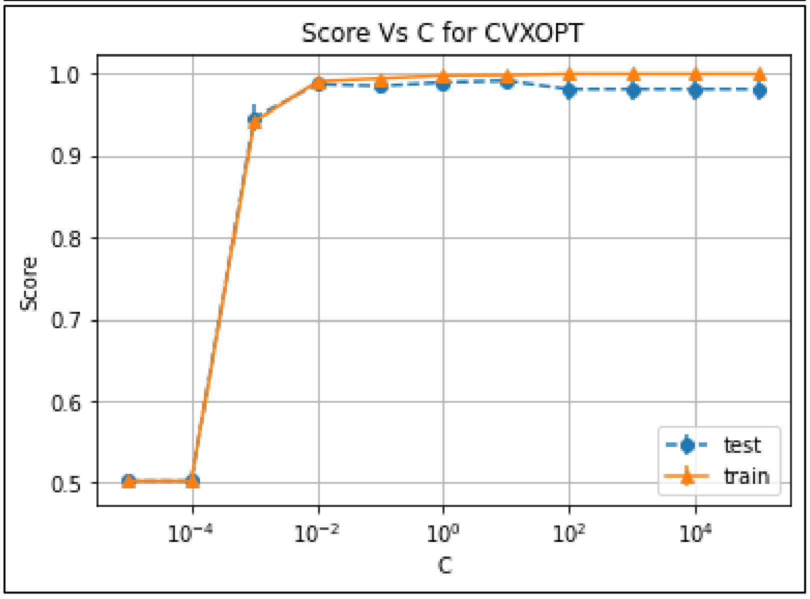
First, I take the first 10 features, split the dataset in 4:1 training:test set and run a 5-fold CV on the training set for the results. I took the best parameters returned by the grid search, and used them to train an SVM classifier but without the standard scaler. I again took these "best parameters" to train the model through CVXOPT and reported the score (accuracies) and support vectors for each of the cases.
We know that a greater number of attributes/features would make the model more complex and increase the number of support vectors. It is also confirmed by the outputs.
The plots are of 2 types. First is the score (mean accuracy) for different values of C, and the second is an error bar for the mean score for the 5-fold CV, spanning the standard deviation about mean. A similar approach is repeated while considering all of the 25 features.

Note: Most of the report below is directly copied from the output of a nice pipeline that I made

```
#################################################################################
Labels:  0 ,  1

Number of features:  10

#################################################################################
Number of training examples:  (484, 10) (484, 1)
Number of test examples:  (121, 10) (121, 1)

--------------------------LIBSVM----------------------------

The Best parameters according to grid search are:  {'SVM__C': 10.0}
Training score for LIBSVM with best parameters:  100.0 %
Test score for LIBSVM with best parameters:  99.17355371900827 %

Indices of support vectors as returned by LIBSVM: [41, 78, 261, 275, 280, 284, 346, 382, 432, 472]
```

```
-------------------------CVXOPT---------------------------

Test score for CVXOPT with best parameters:  99.17355371900827 %
Indices of support vectors as returned by CVXOPT: [41, 78, 261, 275, 280, 284, 346,
382, 432, 472]
```



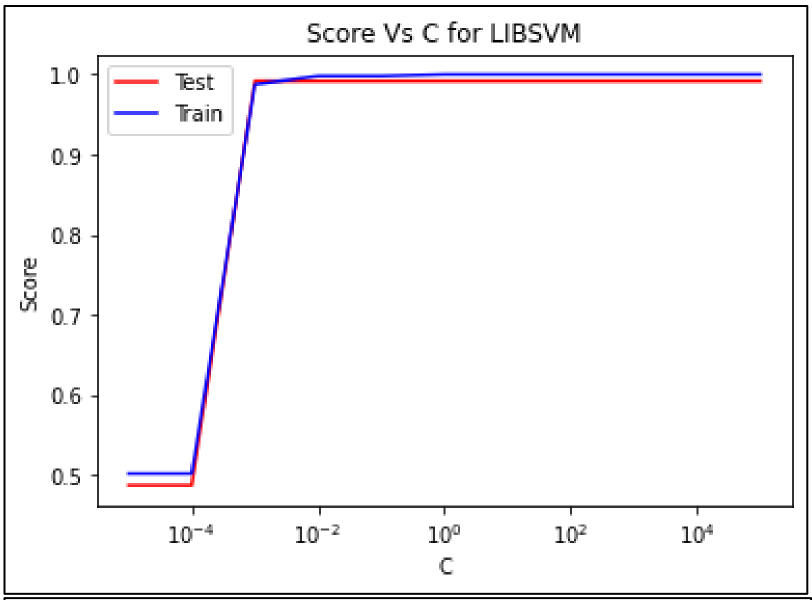


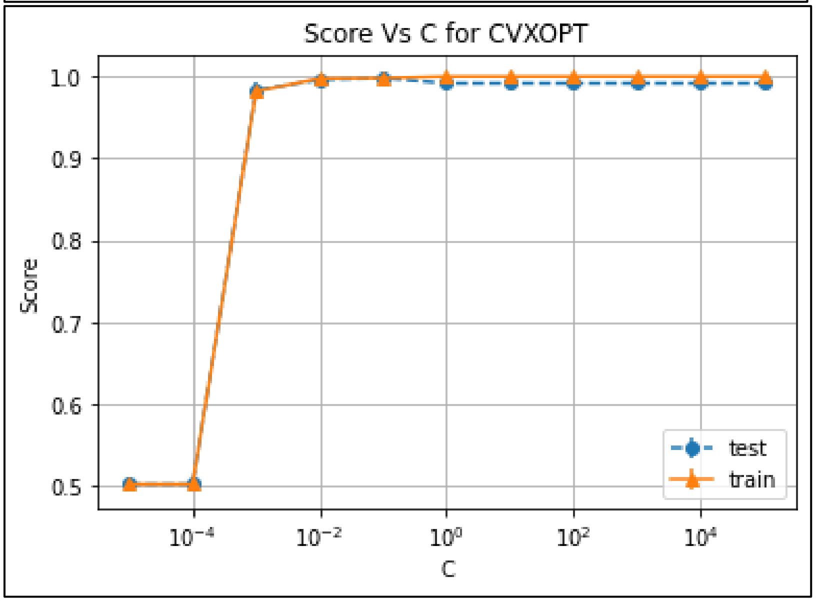
```
####################################################################################
Labels:  0 ,  1
Number of features:  25

####################################################################################
Number of training examples: (484, 25) (484, 1)
Number of test examples: (121, 25) (121, 1)

-------------------------LIBSVM---------------------------

The Best parameters according to grid search are: {'SVM__C': 0.1}
Training score for LIBSVM with best parameters:  100.0 %
```

Test score for LIBSVM with best parameters: 99.17355371900827 %
Indices of support vectors as returned by LIBSVM: [13, 19, 27, 78, 84, 96, 149, 176, 261, 275, 282, 285, 322, 346, 372, 382, 401, 403, 472]

-------------------------CVXOPT----------------------------

Test score for CVXOPT with best parameters: 99.17355371900827 %
Indices of support vectors as returned by CVXOPT: [13, 19, 27, 78, 84, 96, 149, 176, 261, 275, 282, 285, 322, 346, 372, 382, 401, 403, 472]



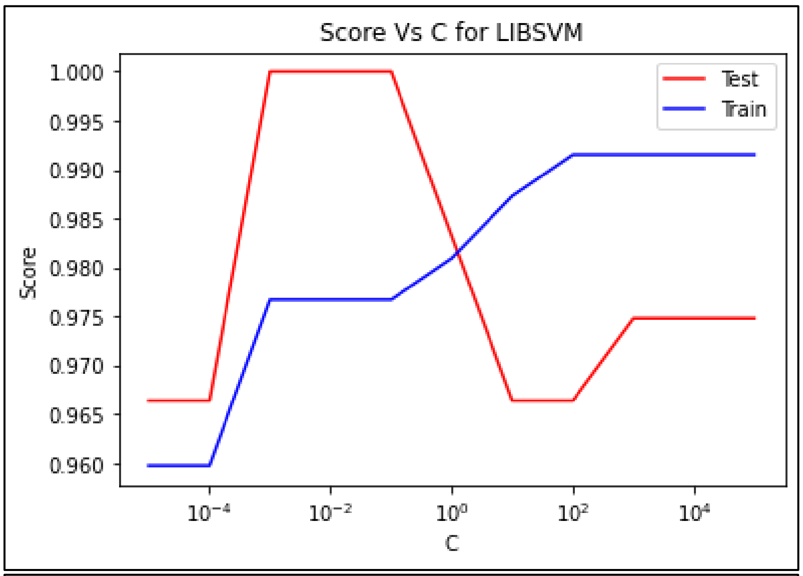


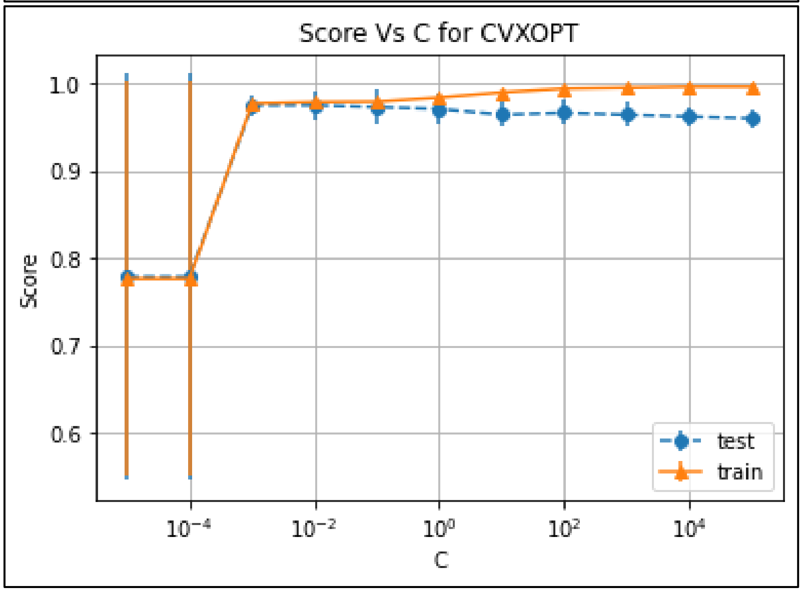
###
Labels: 4 , 6

Number of features: 10

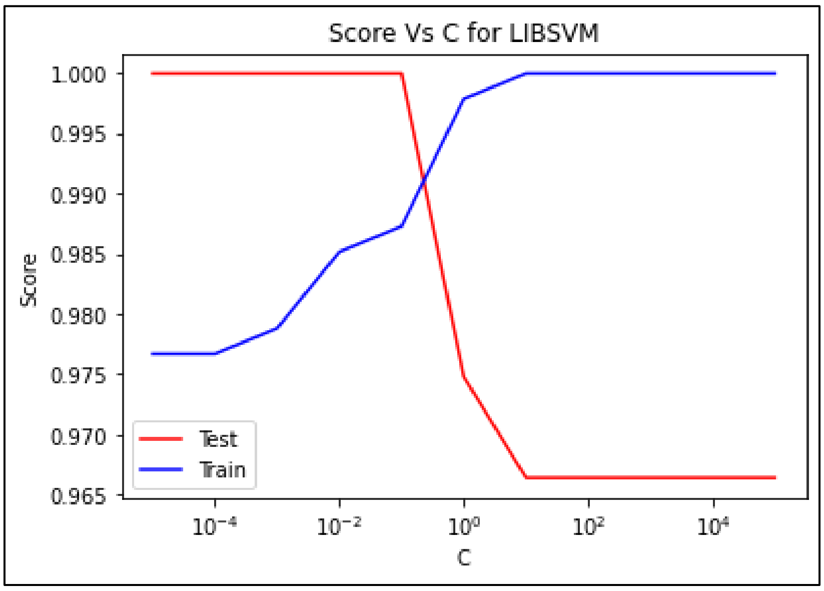
###
Number of training examples: (472, 10) (472, 1)
Number of test examples: (119, 10) (119, 1)

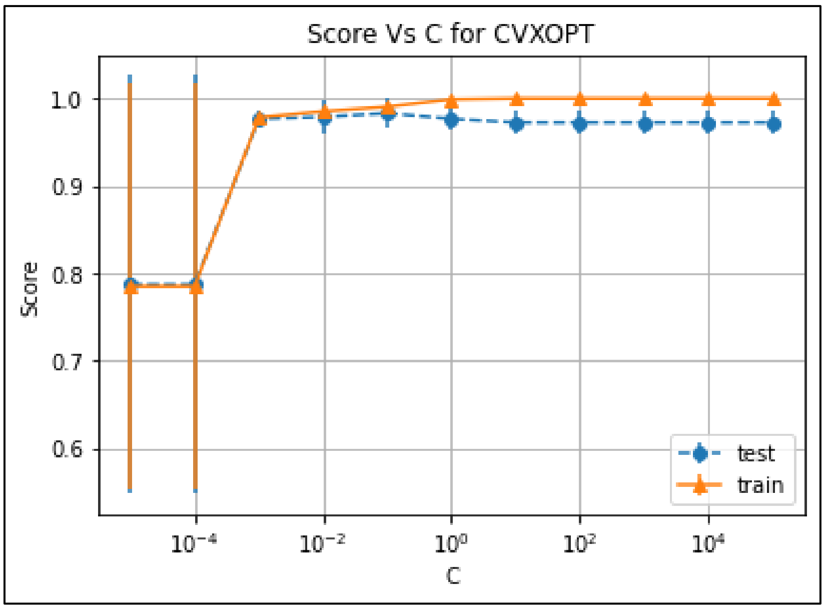
```
--------------------------LIBSVM----------------------------

The Best parameters according to grid search are:  {'SVM__C': 0.01}
Training score for LIBSVM with best parameters:  97.66949152542372 %
Test score for LIBSVM with best parameters:  100.0 %
Indices of support vectors as returned by LIBSVM:  [0, 1, 3, 11, 14, 18, 32, 33, 39, 40, 43, 44,
46, 47, 51, 52, 54, 58, 60, 61, 66, 71, 73, 78, 80, 92, 93, 110, 116, 120, 123, 126, 130, 139, 140, 141,
142, 150, 153, 162, 164, 168, 170, 181, 183, 194, 197, 199, 202, 213, 221, 235, 236, 238, 239, 244,
246, 255, 257, 258, 262, 267, 269, 272, 273, 275, 276, 278, 280, 289, 293, 294, 295, 302, 303, 305,
306, 319, 331, 332, 340, 341, 348, 352, 353, 364, 370, 375, 376, 380, 381, 382, 389, 390, 392, 393,
395, 402, 403, 406, 411, 412, 415, 418, 420, 421, 427, 435, 440, 450, 464, 465, 468, 469]

--------------------------CVXOPT----------------------------

Test score for CVXOPT with best parameters:  100.0 %
Indices of support vectors as returned by CVXOPT:  [0, 1, 3, 11, 14, 18, 32, 33, 39, 40, 43,
44, 46, 47, 51, 52, 54, 58, 60, 61, 66, 71, 73, 78, 80, 92, 93, 110, 116, 120, 123, 126, 130, 139, 140,
141, 142, 150, 153, 162, 164, 168, 170, 181, 183, 194, 197, 199, 202, 213, 221, 235, 236, 238, 239,
244, 246, 255, 257, 258, 262, 267, 269, 272, 273, 275, 276, 278, 280, 289, 293, 294, 295, 302, 303,
305, 306, 319, 331, 332, 340, 341, 348, 352, 353, 364, 370, 375, 376, 380, 381, 382, 389, 390, 392,
393, 395, 402, 403, 406, 411, 412, 415, 418, 420, 421, 427, 435, 440, 450, 464, 465, 468, 469]
```



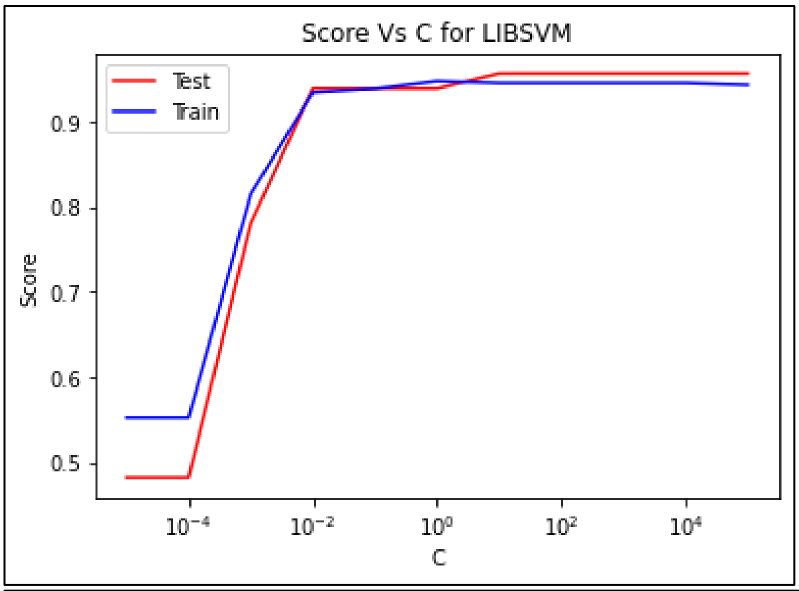


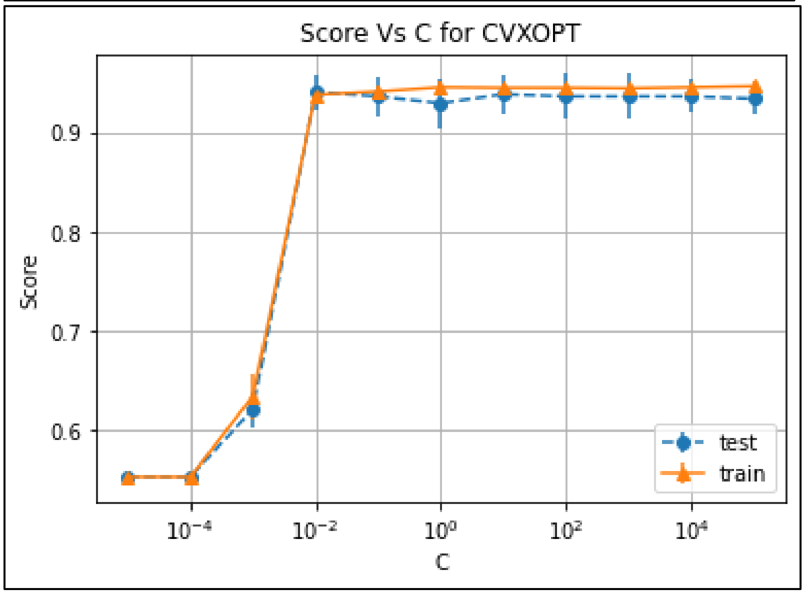
```
######################################################################################
Labels:  4 ,  6

Number of features:  25

######################################################################################
Number of training examples:  (472, 25) (472, 1)
Number of test examples:  (119, 25) (119, 1)

-------------------------LIBSVM----------------------------

The Best parameters according to grid search are:  {'SVM__C': 0.1}
Training score for LIBSVM with best parameters:  98.9406779661017 %
Test score for LIBSVM with best parameters:  100.0 %
Indices of support vectors as returned by LIBSVM:  [0, 11, 14, 43, 44, 51, 52, 61, 66, 73,
83, 94, 133, 140, 142, 150, 153, 164, 170, 194, 199, 221, 234, 235, 272, 275, 276, 280, 289,
294, 302, 319, 352, 381, 390, 395, 414, 420, 426, 435, 440, 450]

-------------------------CVXOPT----------------------------

Test score for CVXOPT with best parameters:  100.0 %
Indices of support vectors as returned by CVXOPT:  [0, 11, 14, 43, 44, 51, 52, 61, 66, 73,
83, 94, 133, 140, 142, 150, 153, 164, 170, 194, 199, 221, 234, 235, 272, 275, 276, 280, 289,
294, 302, 319, 352, 381, 390, 395, 414, 420, 426, 435, 440, 450]
```





```
################################################################################
Labels:  8 ,  9

Number of features:  10

################################################################################
Number of training examples: (454, 10) (454, 1)
Number of test examples: (114, 10) (114, 1)

-------------------------LIBSVM----------------------------

The Best parameters according to grid search are:  {'SVM__C': 0.01}
Training score for LIBSVM with best parameters:  92.95154185022027 %
Test score for LIBSVM with best parameters:  93.85964912280701 %
Indices of support vectors as returned by LIBSVM:  [3, 7, 8, 10, 13, 14, 15, 17, 20, 24, 29, 33,
34, 36, 37, 42, 45, 53, 57, 59, 62, 70, 75, 79, 82, 83, 86, 90, 97, 101, 108, 109, 112, 114, 132, 134,
137, 138, 141, 146, 148, 149, 150, 153, 157, 163, 167, 172, 180, 186, 194, 198, 199, 201, 202, 209,
211, 212, 215, 221, 223, 224, 225, 226, 229, 230, 233, 236, 237, 242, 248, 251, 256, 260, 261, 263,
265, 266, 267, 273, 275, 276, 278, 282, 283, 286, 287, 288, 294, 297, 298, 299, 300, 315, 322, 323,
325, 326, 329, 330, 331, 337, 340, 342, 351, 355, 358, 362, 367, 369, 370, 371, 375, 376, 380, 381,
384, 386, 388, 389, 391, 396, 398, 399, 400, 403, 406, 407, 408, 410, 412, 414, 415, 416, 418, 419,
420, 421, 423, 427, 430, 431, 432, 433, 435, 439, 440, 441, 448, 450]

-------------------------CVXOPT----------------------------

Test score for CVXOPT with best parameters:  93.85964912280701 %
Indices of support vectors as returned by CVXOPT:  [3, 7, 8, 10, 13, 14, 15, 17, 20, 24, 29, 33,
34, 36, 37, 42, 45, 53, 57, 59, 62, 70, 75, 79, 82, 83, 86, 90, 97, 101, 108, 109, 112, 114, 132, 134,
137, 138, 141, 146, 148, 149, 150, 153, 157, 163, 167, 172, 180, 186, 194, 198, 199, 201, 202, 209,
211, 212, 215, 221, 223, 224, 225, 226, 229, 230, 233, 236, 237, 242, 248, 251, 256, 260, 261, 263,
265, 266, 267, 273, 275, 276, 278, 282, 283, 286, 287, 288, 294, 297, 298, 299, 300, 315, 322, 323,
325, 326, 329, 330, 331, 337, 340, 342, 351, 355, 358, 362, 367, 369, 370, 371, 375, 376, 380, 381,
384, 386, 388, 389, 391, 396, 398, 399, 400, 403, 406, 407, 408, 410, 412, 414, 415, 416, 418, 419,
420, 421, 423, 427, 430, 431, 432, 433, 435, 439, 440, 441, 448, 450]
```



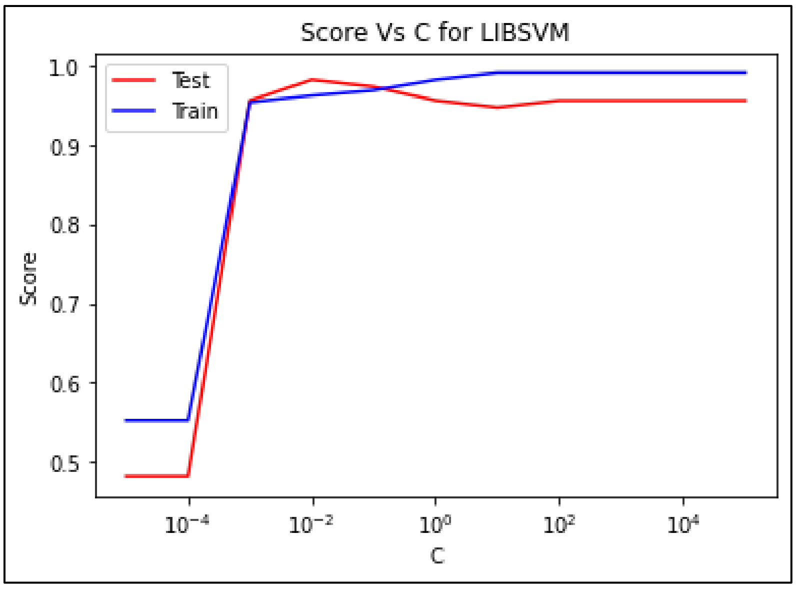


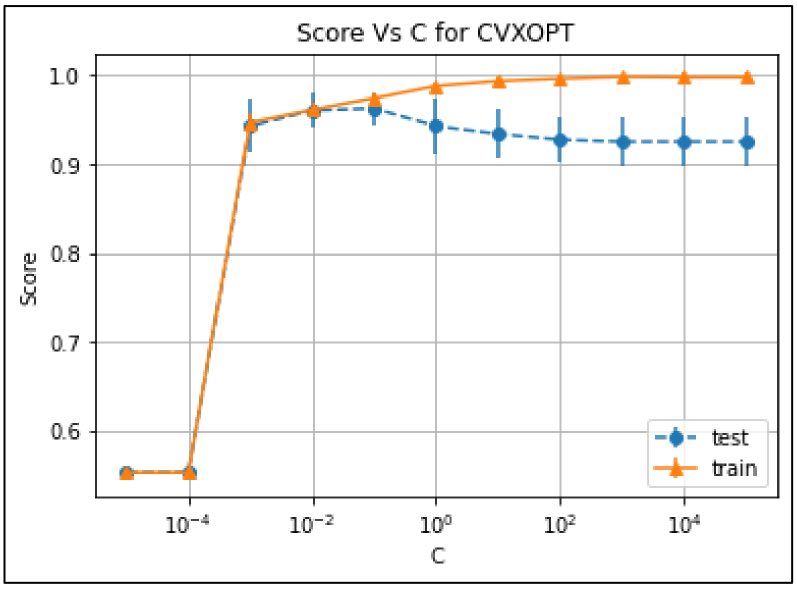
```
##################################################################################
Labels:  8 ,  9

Number of features:  25

##################################################################################
Number of training examples:  (454, 25) (454, 1)
Number of test examples:  (114, 25) (114, 1)

--------------------------LIBSVM----------------------------

The Best parameters according to grid search are:  {'SVM__C': 0.1}
Training score for LIBSVM with best parameters:  97.13656387665198 %
Test score for LIBSVM with best parameters:  97.36842105263158 %
Indices of support vectors as returned by LIBSVM:  [3, 13, 14, 24, 28, 33, 34, 37, 45, 53,
57, 62, 82, 90, 101, 132, 149, 153, 155, 157, 172, 186, 190, 191, 199, 201, 202, 212, 230,
```

231, 236, 244, 260, 261, 267, 288, 314, 322, 323, 331, 340, 351, 355, 362, 369, 377, 380, 381, 392, 398, 400, 403, 406, 407, 415, 416, 419, 420, 423, 439, 441, 442, 448, 450]

-------------------------CVXOPT----------------------------

Test score for CVXOPT with best parameters: 97.36842105263158 %
Indices of support vectors as returned by CVXOPT: [3, 13, 14, 24, 28, 33, 34, 37, 45, 53, 57, 62, 82, 90, 101, 132, 149, 153, 155, 157, 172, 186, 190, 191, 199, 201, 202, 212, 230, 231, 236, 244, 260, 261, 267, 288, 314, 322, 323, 331, 340, 351, 355, 362, 369, 377, 380, 381, 392, 398, 400, 403, 406, 407, 415, 416, 419, 420, 423, 439, 441, 442, 448, 450]





Linear kernel

Class A	Class B	Number of features	Best C	SVM training accuracy	SVM test accuracy	CVXOPT test accuracy
0	1	10	10	100%	99.17%	99.17%

0	1	25	0.1	100%	99.17%	99.17%
4	6	10	0.01	97.67%	100%	100%
4	6	25	0.1	98.94%	100%	100%
8	9	10	0.01	92.95%	93.86%	93.86%
8	9	25	0.1	97.14%	97.37%	97.37%

Table 1.

Discussion

As we can see, the test accuracy is same for Libsvm and CVXOPT. Even the support vectors for both are same (Printed in sorted order, so that can also be confirmed by human inspection)

By the various graphs, one can see that the smaller values of C lead to underfitting, while the higher value of C leads to overfitting. This is not *that* clearly visible because we have 25 features which makes the model complex enough to prevent overfitting.

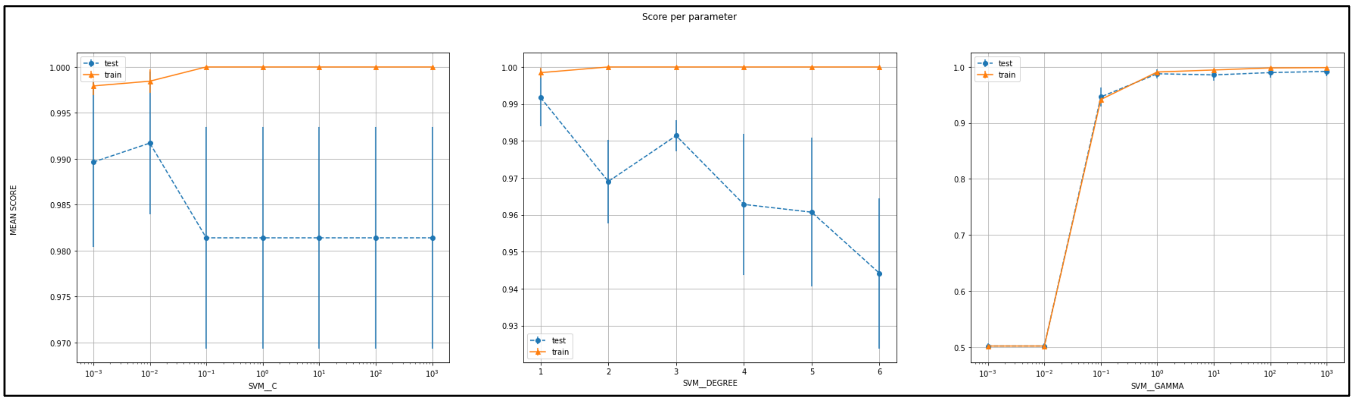
By looking at the table 1, we can easily conclude that as the number of features increase, the test accuracy either remains the same or increases, which is also what we expect because we are increasing the complexity of the model. Since it is a linear classifier, maybe that is why for the pair (0,1) there isn't any increase when number of features are increase 2.5 times

- **Poly Kernel**
- For the poly kernel, we have 2 more parameters, gamma (kernel coefficient) and degree (of the polynomial). The default value of the *independent term* is 0 by default. Since it has little to no bearing on the result (since I am anyway using a standard scaler), I keep it as 0

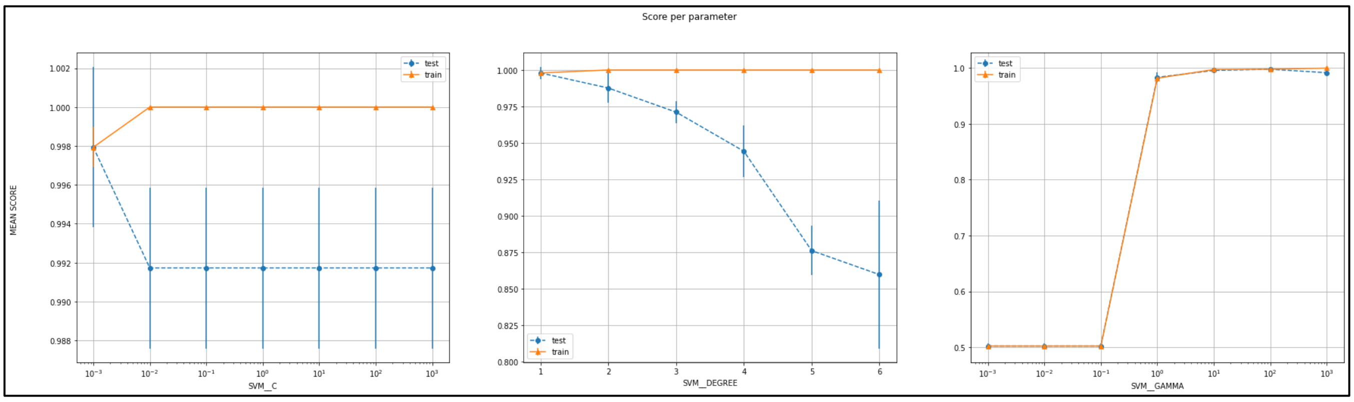
I first consider only 1st 10 features for this sub-part, followed by all 25 features to finally report the results in the table 2.

I take the first 10 features, split the dataset in 4:1 training:test set and run a 5-fold CV on the training set for the results. Having 3 different parameters, there was no definite way to represent the scores on a graph (4D), therefore I only have single set of graphs for this, which are error bars for the mean score for the 5-fold CV, spanning the std dev about mean for all the parameters. Same is done for 25 features iteration
<u>Note</u>: Zoom in on the graph for better visibility

```
###################################################################################
Labels: 0 , 1
Number of features: 10

###################################################################################
Number of training examples: (484, 10) (484, 1)
Number of test examples: (121, 10) (121, 1)

-------------------------LIBSVM----------------------------

The Best parameters according to grid search are: {'SVM__C': 0.01, 'SVM__degree':
1.0, 'SVM__gamma': 1000.0}
Training score for LIBSVM with best parameters: 100.0 %
Test score for LIBSVM with best parameters: 99.17355371900827 %
Indices of support vectors as returned by LIBSVM: [41, 78, 261, 275, 280, 284, 346,
382, 432, 472]

-------------------------CVXOPT----------------------------

Test score for CVXOPT with best parameters: 99.17355371900827 %
Indices of support vectors as returned by CVXOPT: [41, 78, 261, 275, 280, 284, 346,
382, 432, 472]
```



```
##########################################################################################
Labels: 0 , 1
Number of features: 25

##########################################################################################
Number of training examples: (484, 25) (484, 1)
Number of test examples: (121, 25) (121, 1)

--------------------------LIBSVM----------------------------

The Best parameters according to grid search are: {'SVM__C': 0.001, 'SVM__degree':
1.0, 'SVM__gamma': 100.0}
Training score for LIBSVM with best parameters: 100.0 %
Test score for LIBSVM with best parameters: 99.17355371900827 %
Indices of support vectors as returned by LIBSVM:  [13, 19, 27, 78, 84, 96, 149, 176,
261, 275, 282, 285, 322, 346, 372, 382, 401, 403, 472]

--------------------------CVXOPT----------------------------

Test score for CVXOPT with best parameters: 99.17355371900827 %
Indices of support vectors as returned by CVXOPT:  [13, 19, 78, 96, 149, 176, 261,
275, 282, 285, 346, 372, 382, 403, 472]
```

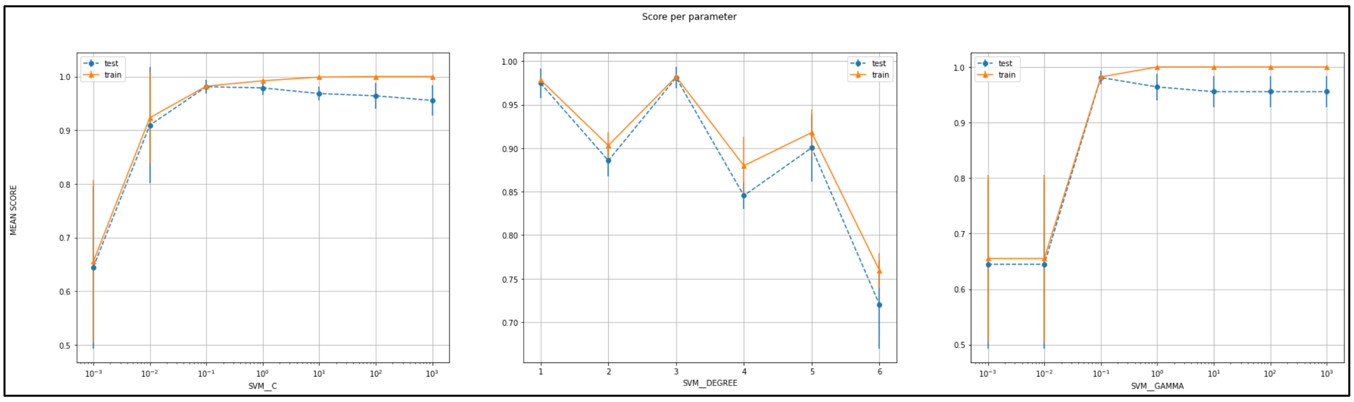


```
##########################################################################################
Labels: 4 , 6

Number of features: 10

##########################################################################################
Number of training examples: (472, 10) (472, 1)
Number of test examples: (119, 10) (119, 1)

--------------------------LIBSVM----------------------------

The Best parameters according to grid search are: {'SVM__C': 0.1, 'SVM__degree':
3.0, 'SVM__gamma': 0.1}
Training score for LIBSVM with best parameters: 99.36440677966102 %
```

Test score for LIBSVM with best parameters: 100.0 %
Indices of support vectors as returned by LIBSVM: [0, 2, 3, 11, 18, 39, 43, 44, 46, 47, 51, 52, 58, 60, 64, 66, 71, 73, 78, 80, 82, 89, 92, 93, 99, 110, 116, 126, 131, 136, 140, 142, 150, 153, 164, 168, 174, 181, 183, 194, 202, 213, 227, 231, 234, 236, 238, 239, 244, 246, 267, 269, 272, 275, 276, 278, 280, 286, 287, 289, 294, 302, 303, 305, 306, 314, 319, 349, 352, 353, 356, 360, 363, 364, 367, 374, 375, 376, 380, 382, 390, 393, 395, 402, 403, 414, 415, 418, 420, 421, 426, 427, 428, 430, 435, 440, 450, 464, 465, 466, 468, 469]

-------------------------CVXOPT----------------------------

Test score for CVXOPT with best parameters: 100.0 %
Indices of support vectors as returned by CVXOPT: [0, 2, 3, 11, 18, 39, 43, 44, 46, 47, 51, 52, 58, 60, 64, 66, 71, 73, 78, 80, 82, 89, 92, 93, 99, 110, 116, 126, 131, 136, 140, 142, 150, 153, 164, 168, 174, 181, 183, 194, 199, 202, 213, 227, 231, 234, 236, 238, 239, 244, 246, 267, 269, 272, 275, 276, 278, 280, 286, 287, 289, 294, 302, 303, 305, 306, 314, 319, 349, 352, 353, 356, 360, 363, 364, 367, 374, 375, 376, 380, 382, 390, 393, 395, 402, 403, 414, 415, 418, 420, 421, 426, 427, 428, 430, 435, 440, 450, 464, 465, 466, 468, 469]



##
Labels: 4 , 6

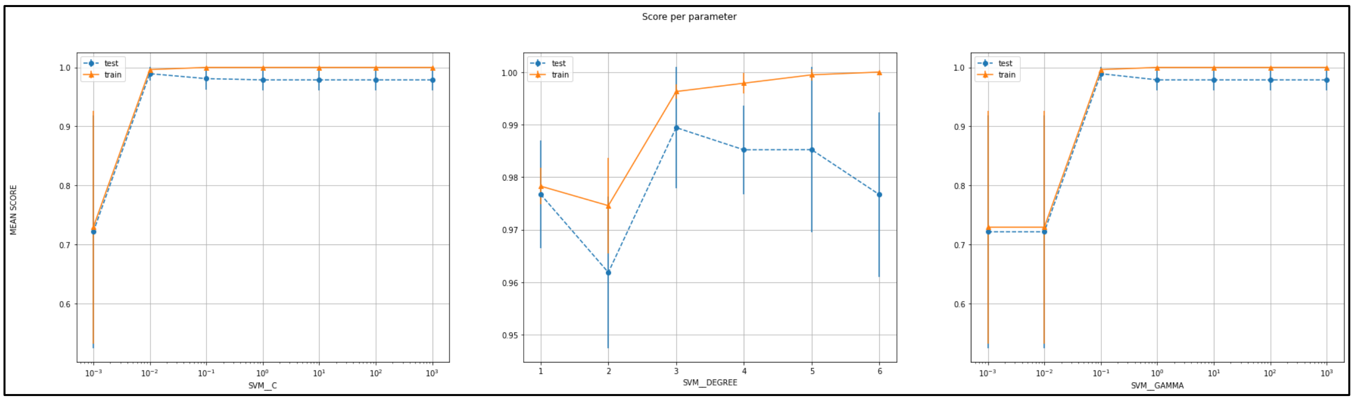
Number of features: 25

##
Number of training examples: (472, 25) (472, 1)
Number of test examples: (119, 25) (119, 1)

-------------------------LIBSVM----------------------------

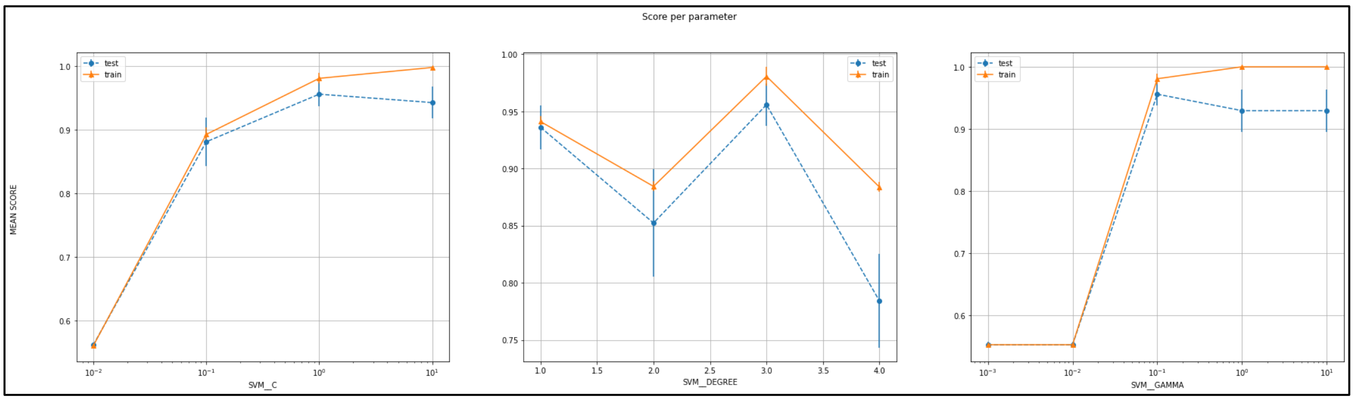
The Best parameters according to grid search are: {'SVM__C': 0.01, 'SVM__degree': 3.0, 'SVM__gamma': 0.1}
Training score for LIBSVM with best parameters: 99.57627118644068 %
Test score for LIBSVM with best parameters: 98.31932773109243 %
Indices of support vectors as returned by LIBSVM: [0, 2, 7, 11, 14, 18, 19, 23, 25, 32, 33, 34, 38, 39, 41, 43, 44, 45, 46, 47, 51, 52, 53, 58, 60, 61, 64, 66, 67, 71, 73, 78, 80, 81, 82, 83, 89, 92, 93, 94, 99, 102, 103, 106, 110, 116, 123, 126, 129, 130, 131, 132, 133, 136, 140, 141, 142, 144, 146, 150, 152, 153, 157, 162, 163, 164, 166, 168, 170, 172, 174, 177, 178, 181, 183, 194, 196, 197, 199, 202, 207, 211, 213, 214, 217, 219, 221, 223, 225, 227, 231, 234, 235, 236, 238, 239, 244, 246, 248, 252, 259, 267, 272, 273, 275, 276, 278, 280, 284, 286, 287, 289, 290, 294, 302, 303, 305, 306, 309, 312, 314, 315, 316, 318, 319, 326, 327, 328, 330, 332, 336, 340, 343, 348, 349, 352, 353, 354, 356, 360,

```
361, 362, 363, 364, 367, 368, 374, 375, 376, 382, 388, 389, 390, 392, 393, 395, 402, 403, 405, 406,
412, 414, 415, 416, 417, 418, 420, 421, 426, 427, 428, 430, 435, 438, 440, 441, 442, 443, 445, 450,
460, 464, 465, 466, 469, 470]

-------------------------CVXOPT----------------------------

Test score for CVXOPT with best parameters: 99.15966386554622 %
Indices of support vectors as returned by CVXOPT:  [0, 2, 7, 11, 14, 18, 19, 23, 25, 32, 33,
34, 38, 39, 41, 43, 44, 45, 46, 47, 51, 52, 53, 58, 60, 61, 64, 66, 67, 71, 73, 78, 80, 81, 82, 83, 89, 92,
93, 94, 99, 102, 103, 106, 110, 116, 123, 126, 129, 130, 131, 132, 133, 136, 140, 141, 142, 144, 146,
150, 152, 153, 157, 162, 163, 164, 166, 168, 170, 172, 174, 177, 178, 181, 183, 194, 196, 197, 199,
202, 207, 211, 213, 214, 217, 219, 221, 223, 225, 227, 231, 234, 235, 236, 238, 239, 244, 246, 248,
252, 259, 267, 272, 273, 275, 276, 278, 280, 284, 286, 287, 289, 290, 294, 302, 303, 305, 306, 309,
312, 314, 315, 316, 318, 319, 326, 327, 328, 330, 332, 336, 340, 343, 348, 349, 352, 353, 354, 356,
360, 361, 362, 363, 364, 367, 368, 374, 375, 376, 382, 388, 389, 390, 392, 393, 395, 402, 403, 405,
406, 412, 414, 415, 416, 417, 418, 420, 421, 426, 427, 428, 430, 435, 438, 440, 441, 442, 443, 445,
450, 460, 464, 465, 466, 469, 470]
```



```
#####################################################################################
Labels: 8 , 9
Number of features: 10

#####################################################################################
Number of training examples: (454, 10) (454, 1)
Number of test examples: (114, 10) (114, 1)

-------------------------LIBSVM----------------------------

The Best parameters according to grid search are: {'SVM__C': 1.0, 'SVM__degree':
3.0, 'SVM__gamma': 0.1}
Training score for LIBSVM with best parameters: 99.77973568281938 %
Test score for LIBSVM with best parameters: 94.73684210526315 %
Indices of support vectors as returned by LIBSVM:  [3, 12, 13, 14, 20, 24, 32, 33, 36, 37,
39, 42, 44, 45, 51, 57, 63, 70, 76, 79, 80, 82, 90, 132, 137, 141, 149, 157, 172, 191, 199, 202,
212, 218, 221, 224, 225, 229, 230, 232, 236, 242, 245, 248, 256, 260, 261, 263, 266, 283,
286, 289, 294, 297, 301, 326, 329, 337, 340, 345, 351, 355, 362, 370, 371, 373, 374, 375,
376, 381, 398, 399, 402, 407, 408, 410, 412, 415, 416, 418, 430, 439, 447, 448]

-------------------------CVXOPT----------------------------
```

Test score for CVXOPT with best parameters: 94.73684210526316 %
Indices of support vectors as returned by CVXOPT: [3, 12, 13, 14, 20, 24, 32, 33, 36, 37, 39, 42, 44, 45, 51, 57, 63, 70, 76, 79, 80, 82, 90, 132, 137, 141, 149, 157, 172, 191, 199, 202, 212, 218, 221, 224, 225, 229, 230, 232, 236, 242, 245, 248, 256, 260, 261, 263, 266, 283, 286, 289, 294, 297, 301, 326, 329, 337, 340, 345, 351, 355, 362, 370, 371, 373, 374, 375, 376, 381, 398, 399, 402, 407, 408, 410, 412, 415, 416, 418, 430, 439, 447, 448]



##
Labels: 8 , 9
Number of features: 25

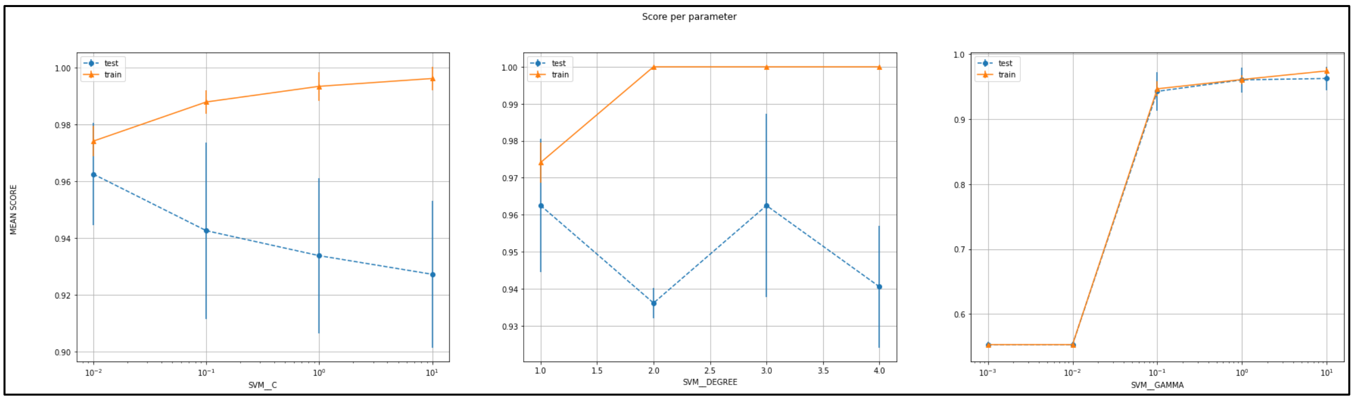
##
Number of training examples: (454, 25) (454, 1)
Number of test examples: (114, 25) (114, 1)

-------------------------LIBSVM---------------------------

The Best parameters according to grid search are: {'SVM__C': 0.01, 'SVM__degree': 1.0, 'SVM__gamma': 10.0}
Training score for LIBSVM with best parameters: 97.13656387665198 %
Test score for LIBSVM with best parameters: 97.36842105263158 %
Indices of support vectors as returned by LIBSVM: [3, 13, 14, 24, 28, 33, 34, 37, 45, 53, 57, 62, 82, 90, 101, 132, 149, 153, 155, 157, 172, 186, 190, 191, 199, 201, 202, 212, 230, 231, 236, 244, 260, 261, 267, 288, 314, 322, 323, 331, 340, 351, 355, 362, 369, 377, 380, 381, 392, 398, 400, 403, 406, 407, 415, 416, 419, 420, 423, 439, 441, 442, 448, 450]

-------------------------CVXOPT---------------------------

Test score for CVXOPT with best parameters: 97.36842105263158 %
Indices of support vectors as returned by CVXOPT: [3, 13, 14, 24, 28, 33, 34, 37, 45, 53, 57, 62, 82, 90, 101, 132, 149, 153, 155, 157, 172, 186, 190, 191, 199, 201, 202, 212, 230, 231, 236, 244, 260, 261, 267, 288, 314, 322, 323, 331, 340, 351, 355, 362, 369, 377, 380, 381, 392, 398, 400, 403, 406, 407, 415, 416, 419, 420, 423, 439, 441, 442, 448, 450]



Poly Kernel

Class A	Class B	Num of features	Best C	Best gamma	Best degree	SVM train score	SVM test score	CVXOPT test score
0	1	10	0.01	1000	1	100%	99.17%	99.17%
0	1	25	0.001	100	1	100%	99.17%	99.17%
4	6	10	0.1	0.1	3	99.36%	100%	100%
4	6	25	0.01	0.1	3	99.58%	98.32%	99.16%
8	9	10	1	0.1	3	99.78%	94.74%	94.74%
8	9	25	0.01	10	1	97.14%	97.37%	97.37%

Table 2.

Discussion

By looking at the table 2, we can conclude that as the number of features increase, the test accuracy either remains the same or increases, which is also what we expect because we are increasing the complexity of the model. For the pair (8,9), the training score is decreasing which indicates that for 10 features, it was experiencing overfitting

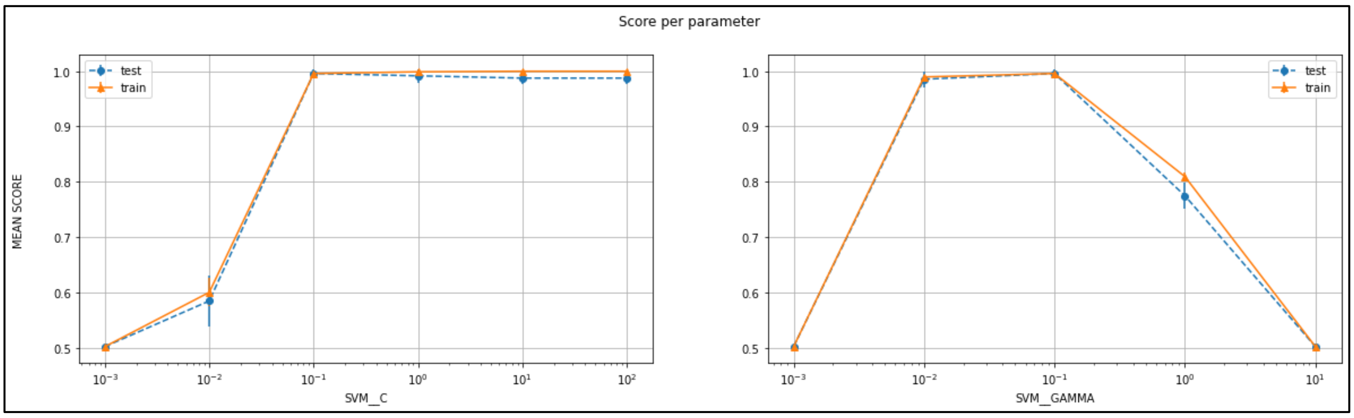
When we look at the graphs that we plotted, we see a similar trend across all the parameters. The training error and the test error start low (Underfitting region) and the training keeps on increasing while the test error reaches a maximum (Optimal best value) and starts decreasing (Overfitting). However, the trend is not exactly the same and we can conclude that degree and C are much more prone to overfitting compared with gamma, which barely shows this trend

- **RBF Kernel**
- For the rbf kernel, we have 2 parameters, gamma (kernel coefficient) and C (degree of margin). The default value of the *independent term* is 0 by default. Since it has little to no bearing on the result (asI am anyway using a standard scaler), I keep it as 0

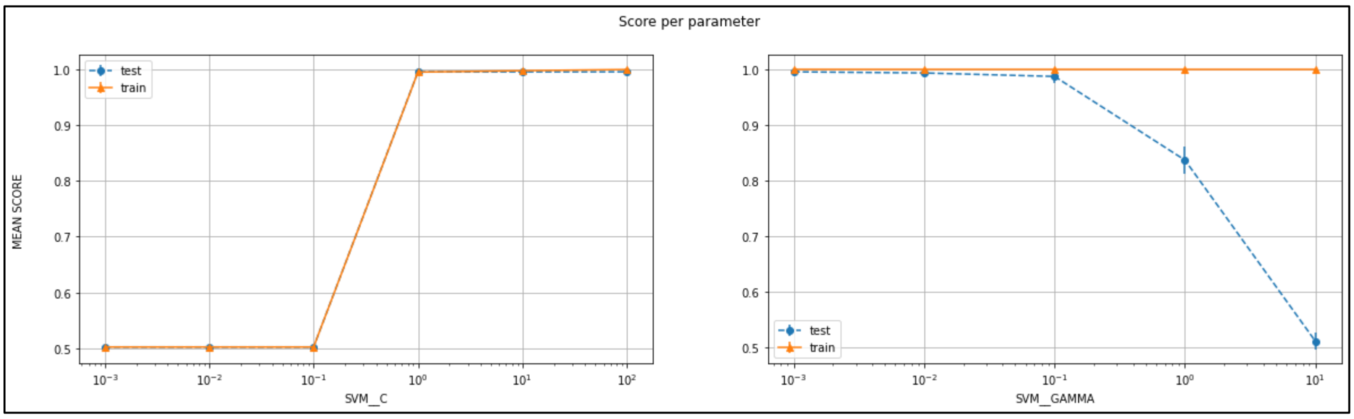
I first consider only 1st 10 features for this sub-part, followed by all 25 features to finally report the results in the table 3.

I take the first 10 features, split the dataset in 4:1 training:test set and run a 5-fold CV on the training set for the results. Having 2 different parameters, there was a definite way to represent the scores on a grap, but lacking continuous values, the contour plot was not very informative, and the wireframe was illegible. Therefore, I only have single set of graphs for this, which are error bars for the mean score for the 5-fold CV, spanning the std dev about mean for all the parameters. Same is done for 25 features iteration.
Note: Zoom in on the graph for better visibility

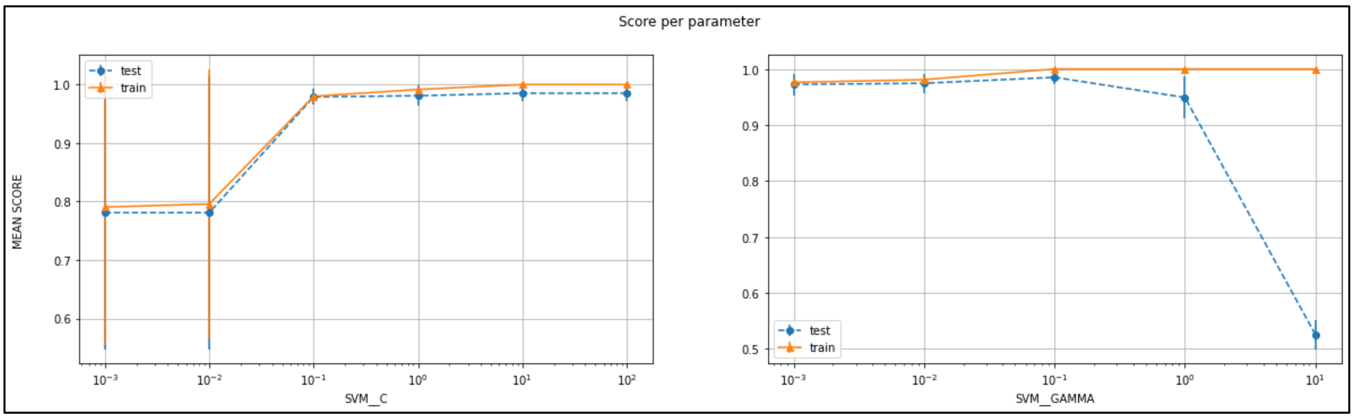
```
##############################################################################
Labels: 0 , 1
Number of features: 10

##############################################################################
Number of training examples: (484, 10) (484, 1)
Number of test examples: (121, 10) (121, 1)

--------------------------LIBSVM----------------------------

The Best parameters according to grid search are: {'SVM__C': 0.1, 'SVM__gamma':
0.1}
Training score for LIBSVM with best parameters: 99.58677685950413 %
Test score for LIBSVM with best parameters: 99.17355371900827 %
Indices of support vectors as returned by LIBSVM: [4, 5, 6, 7, 13, 14, 17, 18, 19, 20, 24, 27,
28, 30, 34, 39, 40, 41, 45, 46, 48, 61, 62, 78, 82, 84, 86, 87, 89, 94, 95, 96, 98, 100, 103, 108, 109,
110, 114, 116, 118, 120, 121, 125, 127, 128, 131, 132, 133, 134, 135, 138, 140, 142, 144, 146, 149,
150, 160, 161, 162, 163, 164, 166, 167, 169, 174, 175, 176, 179, 185, 188, 189, 191, 192, 193, 196,
197, 199, 201, 205, 211, 212, 213, 214, 215, 217, 218, 219, 222, 223, 224, 225, 228, 232, 233, 235,
236, 238, 239, 241, 244, 247, 249, 251, 252, 253, 254, 259, 261, 262, 263, 265, 266, 271, 273, 275,
276, 280, 282, 283, 284, 285, 286, 287, 288, 294, 295, 298, 299, 300, 304, 309, 310, 313, 321, 324,
326, 330, 334, 335, 336, 339, 346, 352, 353, 358, 361, 362, 363, 364, 366, 374, 376, 380, 382, 384,
390, 391, 395, 399, 401, 402, 403, 404, 405, 407, 408, 414, 421, 424, 427, 430, 432, 436, 438, 440,
442, 443, 444, 446, 449, 454, 455, 456, 457, 459, 466, 469, 471, 472, 474, 475, 477, 479, 480, 483]


--------------------------CVXOPT----------------------------

Test score for CVXOPT with best parameters: 99.17355371900827 %
Indices of support vectors as returned by CVXOPT: [4, 5, 6, 7, 13, 14, 17, 18, 19, 20, 24, 27,
28, 30, 34, 39, 40, 41, 45, 46, 48, 61, 62, 75, 78, 82, 84, 86, 87, 89, 94, 95, 96, 98, 100, 103, 108, 109,
110, 114, 116, 118, 120, 121, 125, 127, 128, 131, 132, 133, 134, 135, 138, 140, 142, 144, 146, 149,
150, 160, 161, 162, 163, 164, 166, 167, 169, 174, 175, 176, 179, 185, 188, 189, 191, 192, 193, 196,
197, 199, 201, 205, 211, 212, 213, 214, 215, 217, 218, 219, 222, 223, 224, 225, 228, 232, 233, 235,
236, 238, 239, 241, 244, 247, 249, 251, 252, 253, 254, 259, 261, 262, 263, 265, 266, 271, 273, 275,
276, 280, 282, 283, 284, 285, 286, 287, 288, 294, 295, 298, 299, 300, 304, 309, 310, 313, 321, 324,
326, 330, 334, 335, 336, 339, 346, 352, 353, 358, 361, 362, 363, 364, 366, 374, 376, 380, 382, 384,
390, 391, 395, 399, 401, 402, 403, 404, 405, 407, 408, 414, 421, 424, 427, 430, 432, 436, 438, 440,
442, 443, 444, 446, 449, 454, 455, 456, 457, 459, 466, 469, 471, 472, 474, 475, 477, 479, 480, 483]
```



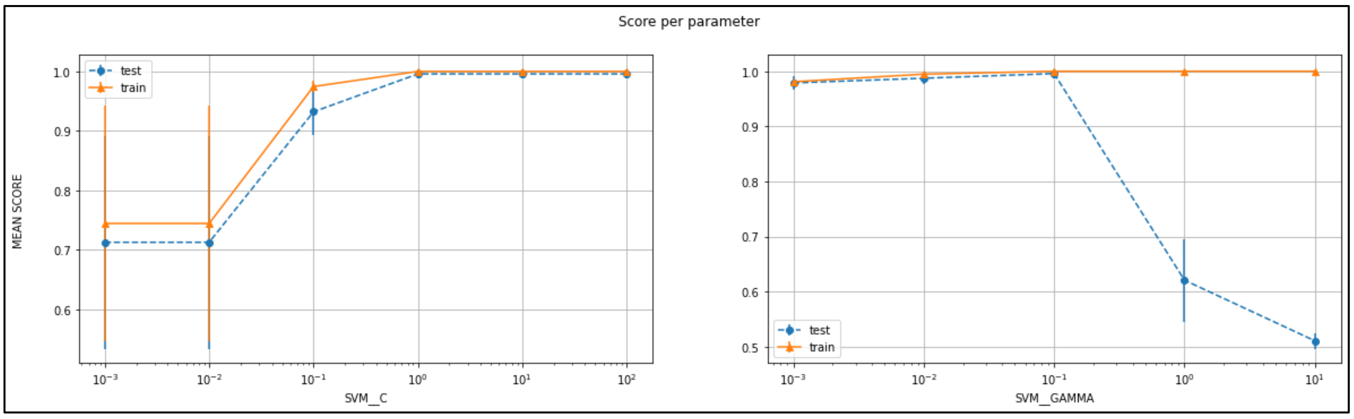
```
##########################################################################################
Labels: 0 , 1
Number of features: 25

##########################################################################################
Number of training examples: (484, 25) (484, 1)
Number of test examples: (121, 25) (121, 1)

--------------------------LIBSVM----------------------------

The Best parameters according to grid search are: {'SVM__C': 100.0, 'SVM__gamma': 0.001}
Training score for LIBSVM with best parameters: 100.0 %
Test score for LIBSVM with best parameters: 99.17355371900827 %
Indices of support vectors as returned by LIBSVM:  [13, 19, 27, 78, 149, 176, 238, 261,
275, 282, 346, 372, 382, 401, 403, 432, 469, 472]


--------------------------CVXOPT----------------------------

Test score for CVXOPT with best parameters: 99.17355371900827 %
Indices of support vectors as returned by CVXOPT:  [13, 19, 27, 78, 149, 176, 238,
239, 261, 275, 282, 322, 346, 372, 382, 401, 403, 432, 469, 472]
```



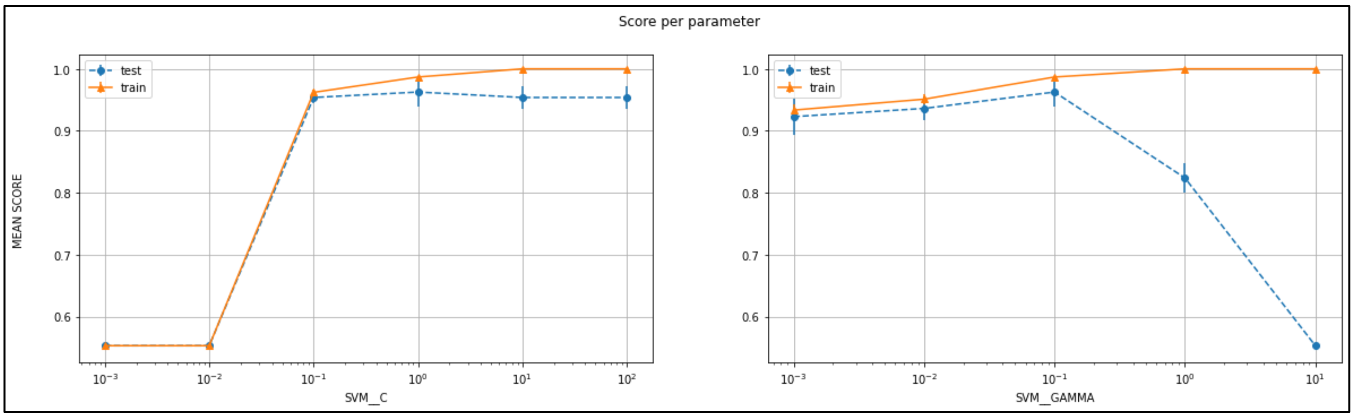
```
##############################################################################
Labels: 4 , 6
Number of features: 10

##############################################################################
Number of training examples: (472, 10) (472, 1)
Number of test examples: (119, 10) (119, 1)

--------------------------LIBSVM----------------------------

The Best parameters according to grid search are: {'SVM__C': 10.0, 'SVM__gamma':
0.1}
Training score for LIBSVM with best parameters: 100.0 %
Test score for LIBSVM with best parameters: 100.0 %
Indices of support vectors as returned by LIBSVM:  [0, 3, 4, 5, 11, 14, 18, 19, 20, 22, 33, 37,
38, 39, 43, 44, 51, 52, 54, 56, 61, 62, 66, 71, 73, 88, 89, 93, 98, 110, 113, 120, 121, 126, 127, 136,
139, 140, 143, 147, 148, 153, 158, 164, 166, 168, 172, 175, 179, 180, 181, 182, 183, 185, 188, 209,
213, 217, 221, 234, 235, 236, 239, 242, 243, 245, 251, 256, 260, 262, 263, 264, 266, 269, 270, 271,
276, 280, 289, 294, 302, 305, 306, 314, 318, 319, 321, 326, 327, 329, 338, 350, 351, 352, 356, 367,
368, 374, 378, 379, 381, 382, 390, 393, 397, 401, 403, 406, 414, 415, 417, 420, 421, 425, 428, 435,
440, 447, 448, 450, 459, 464, 468]


--------------------------CVXOPT----------------------------

Test score for CVXOPT with best parameters: 100.0 %
Indices of support vectors as returned by CVXOPT:  [0, 3, 4, 5, 7, 11, 14, 18, 19, 20, 22, 33,
37, 38, 39, 43, 44, 51, 52, 54, 56, 59, 61, 62, 66, 71, 73, 88, 89, 93, 98, 110, 113, 120, 121, 126, 127,
136, 137, 139, 140, 143, 147, 148, 153, 158, 164, 166, 168, 172, 175, 179, 180, 181, 182, 183, 185,
188, 209, 213, 217, 221, 234, 235, 236, 239, 242, 243, 245, 250, 251, 256, 260, 262, 263, 264, 266,
269, 270, 271, 272, 276, 280, 284, 289, 294, 302, 305, 306, 314, 318, 319, 321, 326, 327, 329, 338,
349, 350, 351, 352, 356, 367, 368, 374, 376, 378, 379, 381, 382, 390, 393, 397, 401, 403, 406, 414,
415, 417, 420, 421, 425, 428, 435, 440, 447, 448, 450, 459, 464, 468]
```



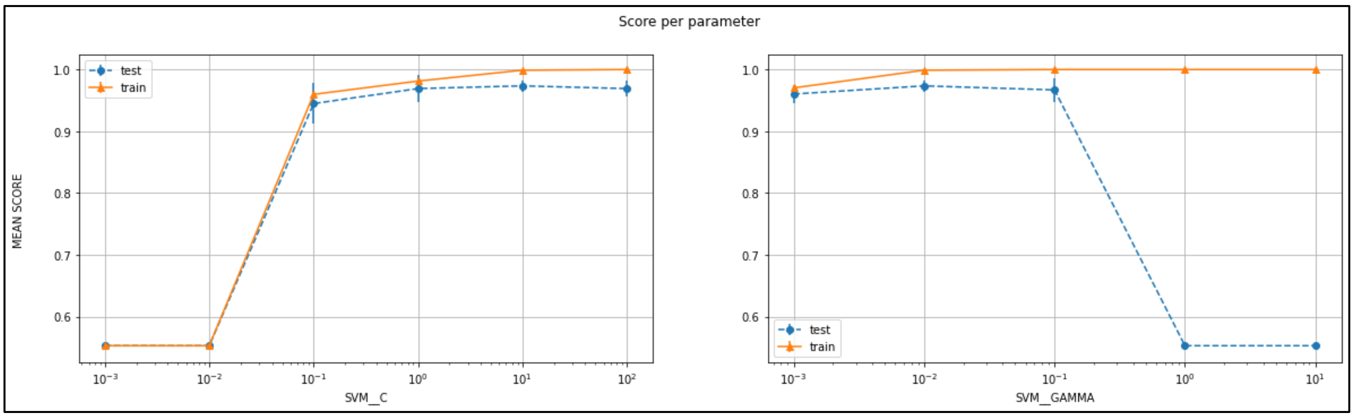
```
####################################################################################
Labels: 4 , 6
Number of features: 25

####################################################################################
Number of training examples: (472, 25) (472, 1)
Number of test examples: (119, 25) (119, 1)

-------------------------LIBSVM----------------------------

The Best parameters according to grid search are: {'SVM__C': 1.0, 'SVM__gamma':
0.1}
Training score for LIBSVM with best parameters: 100.0 %
Test score for LIBSVM with best parameters: 99.15966386554622 %
Indices of support vectors as returned by LIBSVM: [0, 1, 2, 3, 4, 5, 7, 8, 10, 11, 12, 14, 15,
16, 18, 19, 20, 22, 25, 27, 29, 30, 31, 32, 33, 36, 37, 38, 39, 40, 43, 44, 45, 46, 48, 51, 52, 54, 56, 58,
59, 60, 61, 62, 64, 65, 66, 67, 68, 69, 70, 71, 73, 75, 76, 77, 78, 80, 81, 83, 85, 88, 89, 91, 92, 93, 94,
95, 96, 97, 98, 104, 106, 108, 110, 111, 113, 116, 120, 121, 124, 125, 126, 127, 128, 129, 130, 132,
133, 134, 136, 137, 139, 140, 141, 142, 143, 144, 145, 148, 150, 153, 155, 156, 158, 159, 160, 163,
164, 166, 168, 170, 171, 172, 174, 175, 176, 179, 180, 181, 182, 183, 185, 186, 187, 188, 191, 193,
194, 197, 199, 201, 204, 206, 209, 210, 211, 213, 215, 216, 217, 220, 221, 222, 223, 224, 229, 230,
231, 232, 234, 235, 236, 238, 239, 240, 241, 242, 243, 245, 246, 247, 250, 251, 252, 256, 257, 258,
259, 260, 262, 263, 264, 265, 266, 267, 269, 270, 271, 272, 275, 276, 278, 279, 280, 281, 286, 287,
288, 289, 290, 292, 293, 294, 295, 296, 297, 299, 302, 303, 304, 305, 306, 307, 308, 309, 311, 314,
315, 316, 318, 319, 320, 321, 324, 326, 327, 329, 330, 331, 334, 336, 337, 338, 339, 340, 341, 342,
343, 346, 347, 348, 349, 350, 351, 352, 353, 356, 358, 361, 362, 364, 365, 366, 367, 368, 370, 372,
373, 374, 375, 376, 378, 379, 381, 382, 383, 384, 385, 388, 389, 390, 393, 394, 395, 396, 397, 399,
401, 403, 404, 405, 406, 407, 410, 411, 412, 414, 415, 417, 418, 419, 420, 421, 423, 424, 425, 426,
428, 429, 431, 433, 435, 437, 438, 439, 440, 441, 442, 446, 447, 448, 449, 450, 454, 455, 457, 458,
459, 460, 462, 463, 464, 465, 467, 468, 469, 471]


-------------------------CVXOPT----------------------------

Test score for CVXOPT with best parameters: 99.15966386554622 %
Indices of support vectors as returned by CVXOPT: [0, 1, 2, 3, 4, 5, 7, 8, 10, 11, 12, 14, 15,
16, 18, 19, 20, 22, 25, 27, 29, 30, 31, 32, 33, 36, 37, 38, 39, 40, 43, 44, 45, 46, 48, 51, 52, 54, 56, 58,
59, 60, 61, 62, 64, 65, 66, 67, 68, 69, 70, 71, 73, 75, 76, 77, 78, 80, 81, 83, 85, 88, 89, 91, 92, 93, 94,
95, 96, 97, 98, 102, 104, 106, 108, 110, 111, 113, 116, 120, 121, 124, 125, 126, 127, 128, 129, 130,
132, 133, 134, 136, 137, 139, 140, 141, 142, 143, 144, 145, 148, 150, 153, 155, 156, 158, 159, 160,
163, 164, 166, 168, 170, 171, 172, 174, 175, 176, 179, 180, 181, 182, 183, 185, 186, 187, 188, 191,
193, 194, 197, 199, 201, 204, 206, 209, 210, 211, 213, 215, 216, 217, 220, 221, 222, 223, 224, 229,
230, 231, 232, 234, 235, 236, 238, 239, 240, 241, 242, 243, 245, 246, 247, 250, 251, 252, 256, 257,
258, 259, 260, 262, 263, 264, 265, 266, 267, 269, 270, 271, 272, 275, 276, 278, 279, 280, 281, 286,
287, 288, 289, 290, 292, 293, 294, 295, 296, 297, 299, 302, 303, 304, 305, 306, 307, 308, 309, 311,
314, 315, 316, 318, 319, 320, 321, 324, 326, 327, 329, 330, 331, 334, 336, 337, 338, 339, 340, 341,
342, 343, 346, 347, 348, 349, 350, 351, 352, 353, 356, 358, 361, 362, 364, 365, 366, 367, 368, 370,
372, 373, 374, 375, 376, 378, 379, 381, 382, 383, 384, 385, 388, 389, 390, 393, 394, 395, 396, 397,
399, 401, 403, 404, 405, 406, 407, 410, 411, 412, 414, 415, 417, 418, 419, 420, 421, 423, 424, 425,
426, 428, 429, 431, 433, 435, 437, 438, 439, 440, 441, 442, 446, 447, 448, 449, 450, 454, 455, 457,
458, 459, 460, 462, 463, 464, 465, 467, 468, 469, 471]
```



```
################################################################################
Labels: 8 , 9
Number of features: 10

################################################################################
Number of training examples: (454, 10) (454, 1)
Number of test examples: (114, 10) (114, 1)

-------------------------LIBSVM----------------------------

The Best parameters according to grid search are: {'SVM__C': 1.0, 'SVM__gamma':
0.1}
Training score for LIBSVM with best parameters: 98.45814977973568 %
Test score for LIBSVM with best parameters: 98.24561403508771 %
Indices of support vectors as returned by LIBSVM:  [3, 8, 10, 11, 12, 13, 14, 20, 21, 24, 30,
32, 33, 34, 36, 37, 40, 42, 44, 45, 46, 49, 51, 53, 57, 58, 62, 63, 64, 70, 77, 79, 82, 83, 90, 96, 97, 101,
106, 108, 112, 116, 127, 130, 131, 132, 133, 137, 139, 141, 142, 144, 149, 153, 157, 163, 169, 172,
180, 181, 182, 184, 188, 190, 191, 194, 199, 202, 207, 212, 213, 216, 220, 221, 223, 224, 225, 227,
229, 230, 231, 236, 239, 242, 244, 245, 248, 252, 253, 259, 260, 261, 263, 265, 266, 267, 268, 272,
273, 275, 276, 278, 280, 283, 284, 288, 289, 293, 297, 298, 299, 301, 303, 304, 310, 313, 315, 318,
322, 323, 326, 329, 331, 334, 337, 340, 349, 350, 351, 353, 355, 359, 360, 362, 364, 365, 371, 375,
376, 377, 380, 381, 384, 388, 390, 392, 394, 395, 396, 397, 398, 399, 402, 403, 404, 407, 408, 410,
412, 414, 415, 416, 418, 423, 425, 426, 428, 430, 433, 435, 436, 437, 439, 442, 448, 450]



-------------------------CVXOPT----------------------------

Test score for CVXOPT with best parameters: 98.24561403508773 %
Indices of support vectors as returned by CVXOPT:  [3, 8, 10, 11, 12, 13, 14, 20, 21, 24, 30,
32, 33, 34, 36, 37, 40, 42, 44, 45, 46, 49, 51, 53, 57, 58, 62, 63, 64, 70, 77, 79, 82, 83, 90, 96, 97, 101,
106, 108, 112, 113, 116, 127, 130, 131, 132, 133, 137, 139, 140, 141, 142, 144, 149, 153, 157, 163,
164, 169, 172, 180, 181, 182, 184, 188, 190, 191, 194, 199, 202, 207, 212, 213, 216, 220, 221, 223,
224, 225, 227, 229, 230, 231, 236, 239, 242, 244, 245, 248, 252, 253, 256, 259, 260, 261, 263, 265,
266, 267, 268, 272, 273, 275, 276, 278, 280, 283, 284, 288, 289, 293, 297, 298, 299, 301, 303, 304,
310, 313, 315, 318, 322, 323, 326, 329, 331, 334, 337, 340, 349, 350, 351, 353, 355, 359, 360, 362,
364, 365, 371, 375, 376, 377, 380, 381, 384, 388, 390, 392, 394, 395, 396, 397, 398, 399, 402, 403,
404, 407, 408, 410, 412, 414, 415, 416, 418, 423, 425, 426, 428, 430, 433, 435, 436, 437, 439, 442,
448, 450]
```



```
################################################################################
Labels: 8 , 9
Number of features: 25

################################################################################
Number of training examples: (454, 25) (454, 1)
Number of test examples: (114, 25) (114, 1)
--------------------------LIBSVM----------------------------
The Best parameters according to grid search are: {'SVM__C': 10.0, 'SVM__gamma':
0.01}
Training score for LIBSVM with best parameters: 100.0 %
Test score for LIBSVM with best parameters: 98.24561403508771 %
Indices of support vectors as returned by LIBSVM:  [3, 28, 33, 44, 45, 57, 58, 62, 64, 70,
76, 79, 84, 101, 102, 132, 149, 153, 155, 157, 172, 180, 191, 199, 201, 202, 212, 224, 229,
230, 231, 236, 244, 245, 253, 260, 261, 266, 267, 278, 288, 297, 304, 315, 323, 326, 330,
331, 340, 351, 355, 369, 373, 377, 380, 381, 390, 392, 400, 403, 406, 407, 408, 412, 414,
415, 416, 431, 439, 443, 448, 450]

--------------------------CVXOPT----------------------------
Test score for CVXOPT with best parameters: 98.24561403508773 %
Indices of support vectors as returned by CVXOPT:  [3, 28, 33, 44, 45, 57, 58, 62, 64, 70,
76, 79, 84, 101, 102, 132, 149, 153, 155, 157, 172, 180, 191, 199, 201, 202, 212, 224, 229,
230, 231, 236, 244, 245, 253, 260, 261, 266, 267, 278, 288, 297, 304, 315, 323, 326, 330,
331, 340, 351, 355, 369, 373, 377, 380, 381, 390, 392, 400, 403, 406, 407, 408, 412, 414,
415, 416, 431, 439, 443, 448, 450]
```



RBF Kernel

Class A	Class B	Num of features	Best C	Best gamma	SVM train score	SVM test score	CVXOPT test score
0	1	10	0.1	0.1	99.59%	99.17%	99.17%
0	1	25	100	0.001	100%	99.17%	99.17%
4	6	10	10	0.1	100%	100%	100%
4	6	25	1	0.1	100%	99.16%	99.16%
8	9	10	1	0.1	98.46%	98.25%	98.25%
8	9	25	10	0.01	100%	98.25%	98.25%

Table 3

Discussion

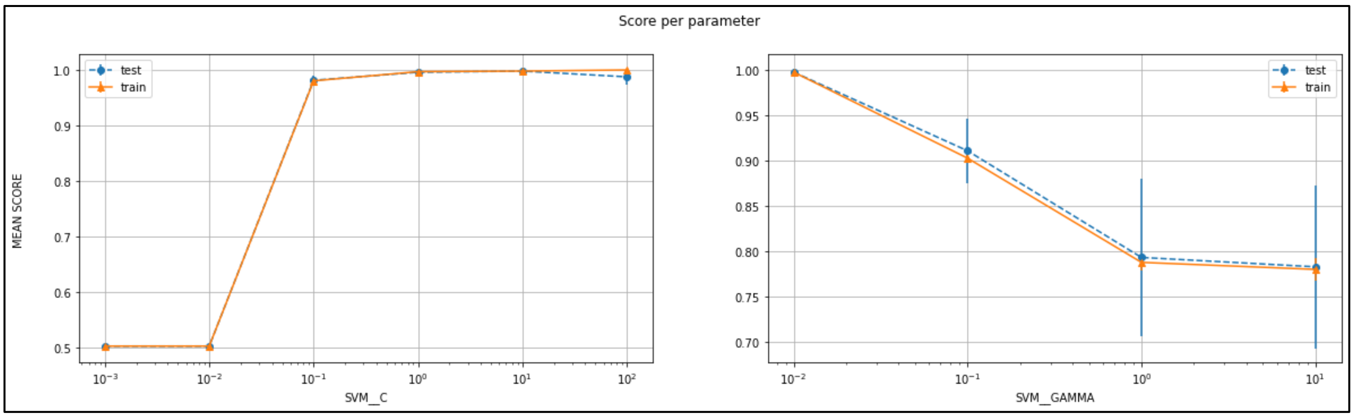
By looking at the table 3, we can see that as the number of features increase, the test accuracy either remains the same or increases, which is also what we expect because we are increasing the complexity of the model. However, there is a slight anomaly for the pair (4,6) as increasing features is actually hurting its test set performance. I do not know why this might be happening.

As stated in the discussion for the poly kernel, we see that at lower values of the parameters, the model underfits the data and as we keep increasing their value, the training accuracy also keeps increasing. Same, however, cannot be said about the test error which is also what we expect

- **Sigmoid Kernel**
- The sigmoid kernel has the same parameters as the RBF kernel, and is therefore iterated over and plotted in the same fashion. The results are tabulated in table 4

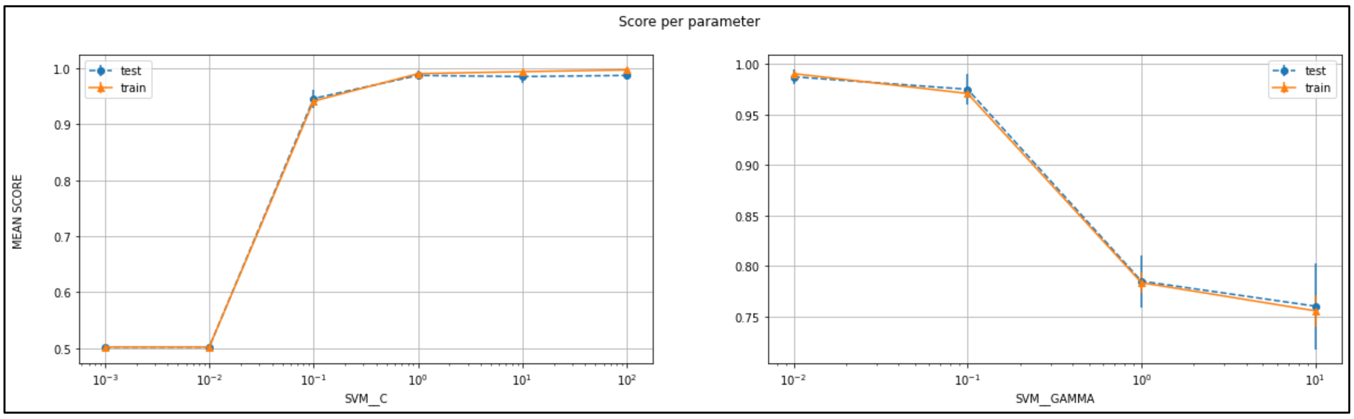
  Note: Zoom in on the graph for better visibility

```
#####################################################################################
Labels: 0 , 1
Number of features: 25

#####################################################################################
Number of training examples: (484, 25) (484, 1)
Number of test examples: (121, 25) (121, 1)

-------------------------LIBSVM----------------------------

The Best parameters according to grid search are: {'SVM__C': 10.0, 'SVM__gamma':
0.01}
Training score for LIBSVM with best parameters: 100.0 %
Test score for LIBSVM with best parameters: 99.17355371900827 %
Indices of support vectors as returned by LIBSVM:  [13, 19, 78, 96, 149, 176, 261, 275,
282, 285, 322, 346, 372, 382, 403, 472]


-------------------------CVXOPT----------------------------

Test score for CVXOPT with best parameters: 99.17355371900827 %
Indices of support vectors as returned by CVXOPT:  [13, 19, 27, 78, 96, 149, 176, 261,
275, 282, 285, 322, 346, 372, 382, 403, 472]
```



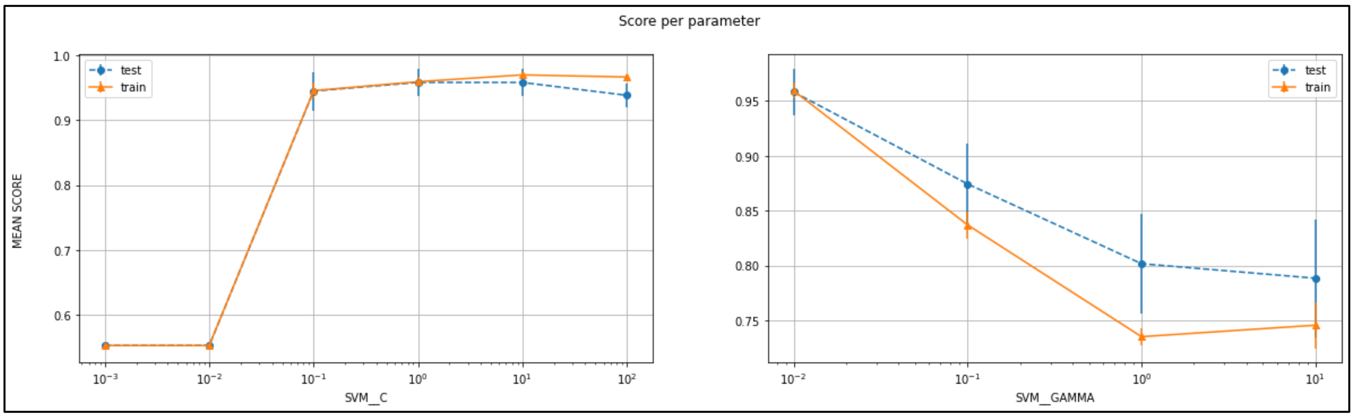
```
#####################################################################################
Labels: 0 , 1
Number of features: 10

#####################################################################################
Number of training examples: (484, 10) (484, 1)
Number of test examples: (121, 10) (121, 1)
```

```
-------------------------LIBSVM----------------------------
The Best parameters according to grid search are: {'SVM__C': 1.0, 'SVM__gamma':
0.01}
Training score for LIBSVM with best parameters: 99.58677685950413 %
Test score for LIBSVM with best parameters: 100.0 %
Indices of support vectors as returned by LIBSVM:  [4, 13, 14, 19, 24, 41, 78, 84, 96,
118, 119, 133, 135, 144, 149, 161, 162, 163, 176, 192, 197, 199, 261, 275, 280, 284,
285, 304, 313, 346, 382, 395, 403, 432, 445, 454, 456, 467, 469, 472]

-------------------------CVXOPT----------------------------
Test score for CVXOPT with best parameters: 100.0 %
Indices of support vectors as returned by CVXOPT:  [4, 13, 14, 19, 24, 41, 78, 84, 96,
118, 119, 133, 135, 144, 149, 161, 162, 163, 176, 192, 197, 199, 261, 275, 280, 284,
285, 304, 313, 346, 382, 395, 403, 432, 445, 454, 456, 467, 469, 472]
```

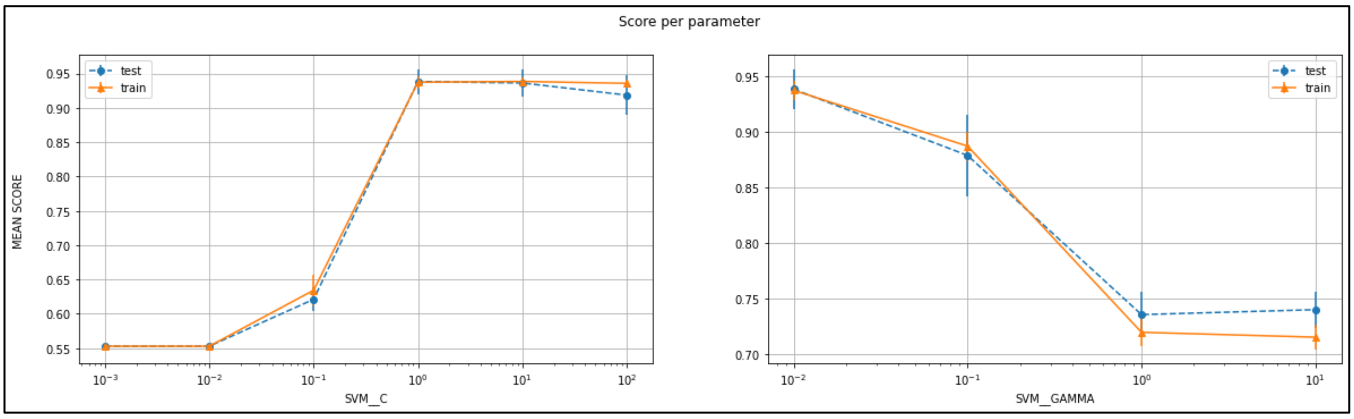


```
####################################################################################
Labels: 8 , 9
Number of features: 25

####################################################################################
Number of training examples: (454, 25) (454, 1)
Number of test examples: (114, 25) (114, 1)
-------------------------LIBSVM----------------------------
The Best parameters according to grid search are: {'SVM__C': 1.0, 'SVM__gamma':
0.01}
Training score for LIBSVM with best parameters: 96.47577092511013 %
Test score for LIBSVM with best parameters: 96.49122807017544 %
Indices of support vectors as returned by LIBSVM:  [1, 3, 7, 8, 13, 14, 15, 17, 20, 24, 28, 29,
33, 34, 37, 42, 44, 45, 53, 57, 58, 59, 62, 72, 76, 79, 82, 83, 90, 95, 101, 108, 109, 112, 117, 132, 134,
137, 149, 153, 157, 172, 180, 186, 190, 191, 199, 201, 202, 212, 223, 224, 225, 226, 229, 230, 231,
233, 236, 237, 244, 248, 251, 253, 256, 260, 261, 267, 275, 278, 281, 282, 283, 288, 294, 297, 298,
299, 301, 314, 315, 322, 323, 325, 326, 327, 330, 331, 340, 351, 355, 358, 362, 369, 373, 376, 377,
380, 381, 390, 392, 398, 399, 400, 403, 406, 407, 408, 410, 412, 414, 415, 416, 418, 419, 420, 427,
431, 433, 435, 437, 439, 441, 443, 448, 450]
```

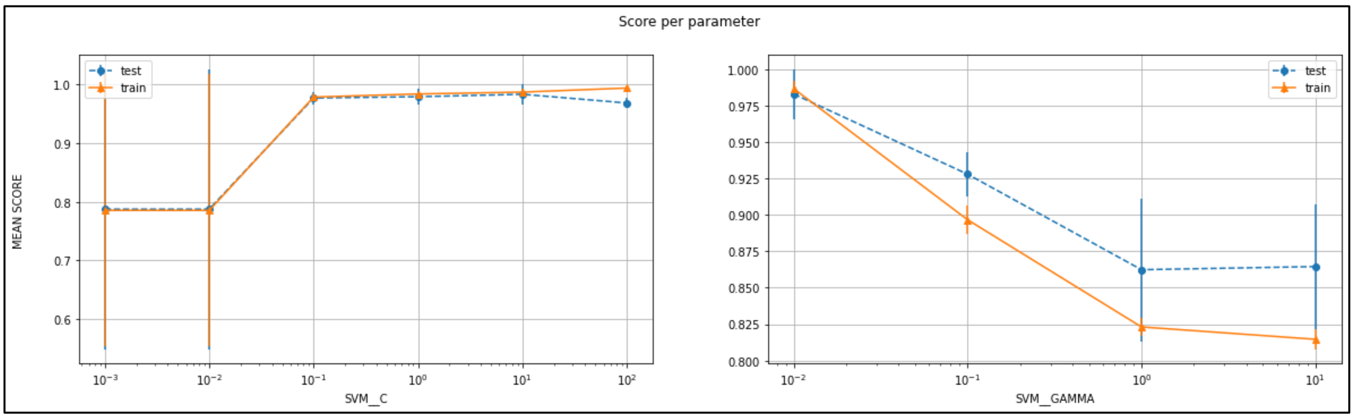
```
-------------------------CVXOPT---------------------------
Test score for CVXOPT with best parameters: 96.49122807017544 %
Indices of support vectors as returned by CVXOPT:  [1, 3, 7, 8, 13, 14, 15, 17, 20, 24, 28, 29,
33, 34, 37, 42, 44, 45, 53, 57, 58, 59, 62, 72, 76, 79, 82, 83, 90, 95, 101, 108, 109, 112, 117, 132, 134,
137, 149, 153, 157, 172, 180, 186, 190, 191, 199, 201, 202, 212, 223, 224, 225, 226, 229, 230, 231,
233, 236, 237, 244, 248, 251, 253, 256, 260, 261, 267, 275, 278, 281, 282, 283, 288, 294, 297, 298,
299, 301, 314, 315, 322, 323, 325, 326, 327, 330, 331, 340, 351, 355, 358, 362, 369, 373, 376, 377,
380, 381, 390, 392, 398, 399, 400, 403, 406, 407, 408, 410, 412, 414, 415, 416, 418, 419, 420, 427,
431, 433, 435, 437, 439, 441, 443, 448, 450]
```



```
################################################################################
Labels: 8 , 9
Number of features: 10

################################################################################
Number of training examples: (454, 10) (454, 1)
Number of test examples: (114, 10) (114, 1)
-------------------------LIBSVM---------------------------
The Best parameters according to grid search are: {'SVM__C': 1.0, 'SVM__gamma':
0.01}
Training score for LIBSVM with best parameters: 92.95154185022027 %
Test score for LIBSVM with best parameters: 93.85964912280701 %
Indices of support vectors as returned by LIBSVM:  [3, 7, 8, 10, 13, 14, 15, 20, 24, 29, 33, 34,
36, 37, 42, 45, 53, 57, 59, 62, 70, 75, 79, 82, 83, 86, 90, 97, 101, 108, 109, 112, 114, 132, 134, 137,
138, 141, 146, 148, 149, 150, 153, 157, 163, 167, 172, 180, 186, 194, 198, 199, 201, 202, 209, 211,
212, 215, 221, 223, 224, 225, 226, 229, 230, 233, 236, 237, 242, 248, 251, 256, 260, 261, 263, 265,
266, 267, 273, 275, 276, 278, 282, 283, 286, 287, 288, 294, 297, 298, 299, 300, 315, 322, 323, 325,
326, 329, 330, 331, 332, 337, 340, 342, 351, 355, 358, 362, 367, 369, 370, 371, 375, 376, 380, 381,
384, 386, 388, 389, 391, 396, 398, 399, 400, 403, 406, 407, 408, 410, 412, 414, 415, 416, 418, 419,
420, 421, 423, 427, 430, 431, 432, 433, 435, 439, 440, 441, 448, 450]

-------------------------CVXOPT---------------------------
Test score for CVXOPT with best parameters: 93.85964912280701 %
Indices of support vectors as returned by CVXOPT:  [3, 7, 8, 10, 13, 14, 15, 17, 20, 24, 29,
33, 34, 36, 37, 42, 45, 53, 57, 59, 62, 70, 75, 79, 82, 83, 86, 90, 97, 101, 108, 109, 112, 114, 132, 134,
137, 138, 141, 146, 148, 149, 150, 153, 157, 163, 167, 172, 180, 186, 194, 198, 199, 201, 202, 209,
211, 212, 215, 221, 223, 224, 225, 226, 229, 230, 233, 236, 237, 242, 248, 251, 256, 260, 261, 263,
265, 266, 267, 273, 275, 276, 278, 282, 283, 286, 287, 288, 294, 297, 298, 299, 300, 309, 315, 322,
323, 325, 326, 329, 330, 331, 332, 337, 340, 342, 351, 355, 358, 362, 367, 369, 370, 371, 375, 376,
380, 381, 384, 386, 388, 389, 391, 396, 398, 399, 400, 403, 406, 407, 408, 410, 412, 414, 415, 416,
418, 419, 420, 421, 423, 427, 430, 431, 432, 433, 435, 439, 440, 441, 448, 450]
```



```
####################################################################################
Labels: 4 , 6
Number of features: 25

####################################################################################
Number of training examples: (472, 25) (472, 1)
Number of test examples: (119, 25) (119, 1)
--------------------------LIBSVM----------------------------
The Best parameters according to grid search are: {'SVM__C': 10.0, 'SVM__gamma':
0.01}
Training score for LIBSVM with best parameters: 98.30508474576271 %
Test score for LIBSVM with best parameters: 100.0 %
Indices of support vectors as returned by LIBSVM:  [0, 11, 14, 43, 44, 51, 52, 61, 66,
73, 83, 106, 133, 140, 142, 150, 153, 164, 170, 194, 199, 234, 235, 272, 275, 276, 280,
289, 294, 302, 306, 319, 352, 381, 390, 393, 395, 414, 420, 440, 450]

--------------------------CVXOPT----------------------------
Test score for CVXOPT with best parameters: 100.0 %
Indices of support vectors as returned by CVXOPT:  [0, 11, 14, 43, 44, 51, 52, 61, 66,
73, 83, 106, 133, 140, 142, 150, 153, 164, 170, 194, 199, 234, 235, 236, 272, 275, 276,
280, 289, 294, 302, 319, 352, 381, 382, 390, 395, 414, 420, 440, 450]
```

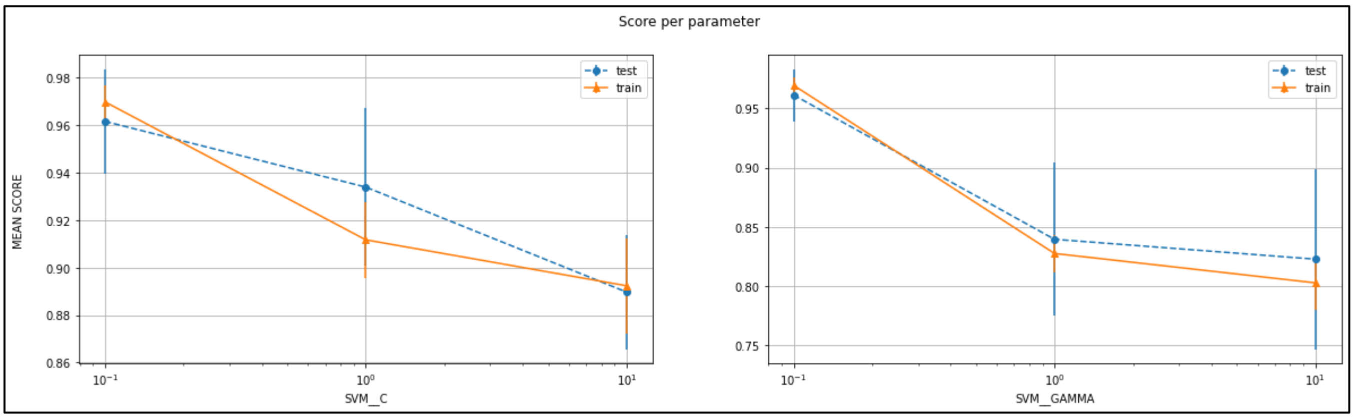


##
Labels: 4 , 6
Number of features: 10

##
Number of training examples: (472, 10) (472, 1)
Number of test examples: (119, 10) (119, 1)
--------------------------LIBSVM----------------------------
The Best parameters according to grid search are: {'SVM__C': 0.1, 'SVM__gamma': 0.1}
Training score for LIBSVM with best parameters: 94.91525423728814 %
Test score for LIBSVM with best parameters: 94.11764705882352 %
Indices of support vectors as returned by LIBSVM: [0, 1, 3, 4, 10, 11, 14, 24, 25, 33, 40, 43, 44, 46, 47, 51, 52, 54, 58, 60, 61, 62, 66, 71, 76, 77, 91, 92, 93, 94, 108, 109, 110, 117, 120, 123, 126, 139, 140, 141, 142, 145, 147, 149, 150, 153, 158, 160, 161, 162, 164, 170, 174, 180, 182, 202, 210, 215, 217, 220, 221, 224, 228, 230, 235, 236, 238, 239, 241, 242, 244, 246, 255, 256, 257, 258, 262, 265, 266, 267, 271, 272, 273, 275, 276, 277, 280, 281, 289, 294, 296, 297, 299, 300, 302, 303, 305, 308, 314, 316, 319, 321, 328, 331, 332, 335, 339, 340, 341, 342, 348, 357, 358, 364, 370, 375, 376, 378, 380, 381, 382, 384, 385, 389, 390, 393, 394, 395, 397, 403, 406, 409, 411, 412, 415, 420, 421, 423, 427, 435, 436, 440, 447, 450, 455, 460, 468, 469]

--------------------------CVXOPT----------------------------

My CVXOPT returned an arithmetic error related to rank for this which I could not rectify



Sigmoid Kernel

Class A	Class B	Num of features	Best C	Best gamma	SVM train score	SVM test score	CVXOPT test score

0	1	10	1	0.01	99.59%	100%	100%
0	1	25	10	0.01	100%	99.17%	99.17%
4	6	10	0.1	0.1	94.92%	94.12%	-
4	6	25	10	0.01	98.31%	100%	100%
8	9	10	1	0.01	92.95%	93.86%	93.86%
8	9	25	1	0.01	96.48%	96.49%	96.49%

Table 4

Discussion

By looking at the table 4, we can easily conclude that as the number of features increase, the test accuracy either remains the same or increases, which is also what we expect because we are increasing the complexity of the model.

The one thing that makes this kernel different from the rest is that instead of just test error decreasing for larger values of gamma, both training and test error decrease. i.e., there is something in its internal functioning which has negative effect by larger values of gamma

➢ Conclusion (Part 1A, Binary Classification)

As mentioned in the individual sub-sections, it almost never hurt to increase the number of features as it only increased the accuracy. Moreover, other than gamma in few cases, the underfitting to overfitting change followed a similar trend.

One very important and validating (for me) thing that I also noticed was how the support vectors returned by CVXOPT and LIBSVM were almost exactly the same. I returned the indices of the support vector for that particular training set so anyone can manually confirm this. Also, the test score was the same for libsvm and cvxopt (barring a few exceptions) which is not surprising as the support vectors are the deciding factor for the classifier.

Now, I present the best parameters and scores for each of the kernels for each set of pairs for 25 features:

Pair	Kernel	Best C	Best degree	Best gamma	SVM train score	SVM test score	CVXOPT test score
(0,1)	Linear	0.1	-	-	100%	99.17%	99.17%
(0,1)	Poly	0.001	1	100	100%	99.17%	99.17%
(0,1)	RBF	100	-	0.001	100%	99.17%	99.17%
(0,1)	Sigmoid	10	-	0.01	100%	99.17%	99.17%
(4,6)	Linear	0.1	-	-	98.94%	100%	100%
(4,6)	Poly	0.01	3	0.1	99.58%	98.32%	99.16%
(4,6)	RBF	1	-	0.1	100%	99.16%	99.16%
(4,6)	Sigmoid	10	-	0.01	98.31%	100%	100%
(8,9)	Linear	0.1	-	-	97.14%	97.37%	97.37%
(8,9)	Poly	0.01	1	10	97.14%	97.37%	97.37%
(8,9)	RBF	10	-	0.01	100%	98.25%	98.25%
(8,9)	Sigmoid	1	-	0.01	96.48%	96.49%	96.49%

Table 5

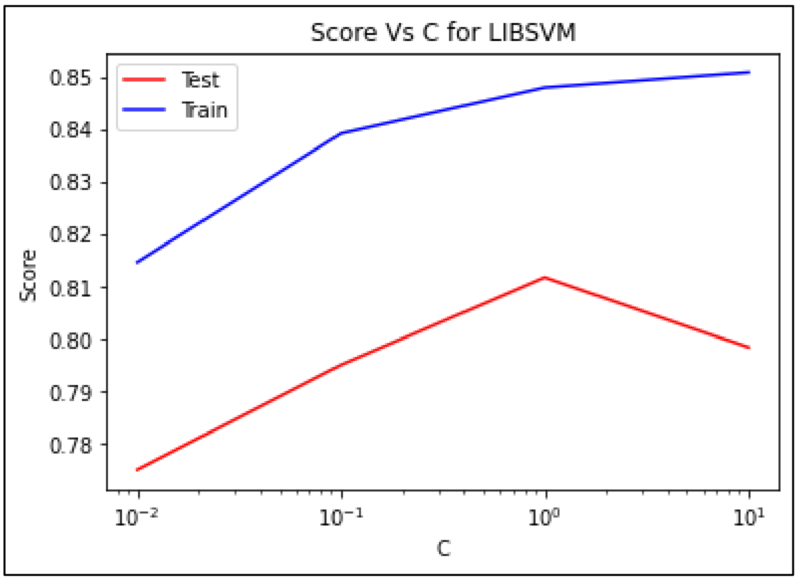
Interestingly enough, a simple model with linear kernel performs really well for the dataset at hand. This might be because 25 features are good enough and we jut happen to have a good dataset. Or perhaps the projection to lower dimensions does our job for us

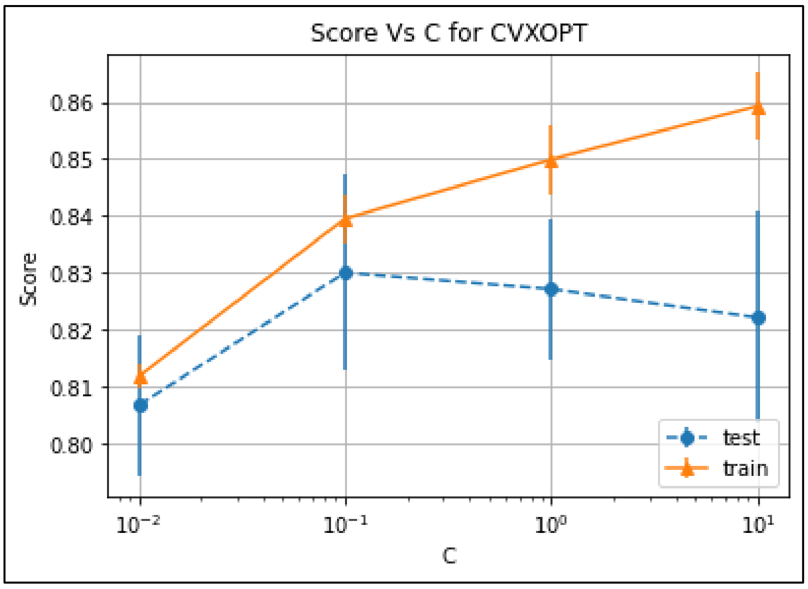
Although it is not guaranteed that the same hyperparameters give the best results (As we can see in our tables). But, barring a few outliers, the best score is generally for a small range of hyperparameters only

- **Multi-class Classification**:
  - In this part, we will be classifying our data not on randomly chosen pairs, but on all the labels together. Libsvm uses a one-vs-one technique to handle this which has been discussed in class in short
  - My method to approach this remains the same. A 5-fold cross validation for each of the 4 kernels across 4-5 values of each of the hyperparameters. A linear space for degree and logspace for C and gamma.
  - The graphs are also presented in a similar fashion

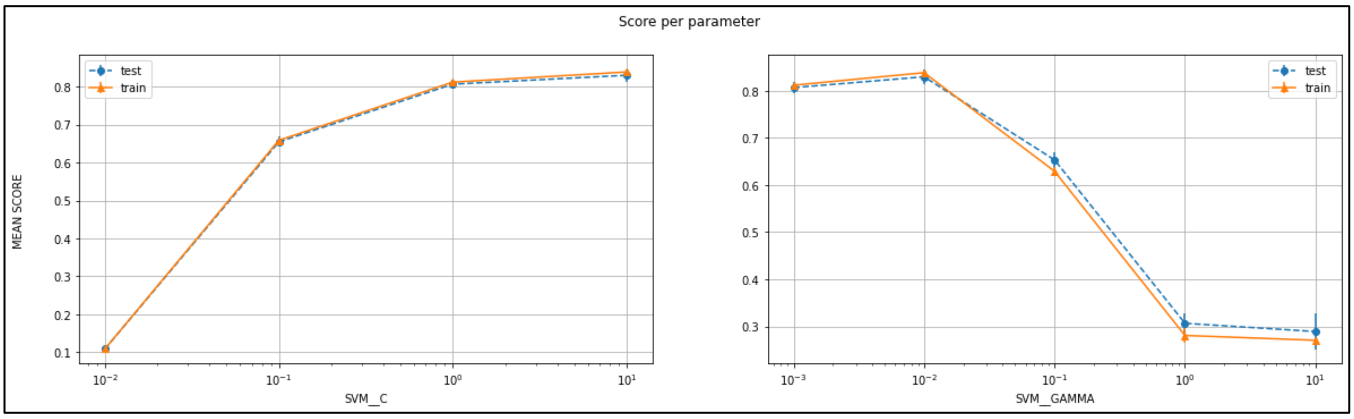
Note, zoom in on the graphs for better visibility. They are in good quality.

```
##########################################################################
Number of features: 10
##########################################################################
Number of training examples: (2400, 10) (2400, 1)
Number of test examples: (600, 10) (600, 1)
-------------------------LIBSVM for LINEAR for 10 features----------------------------
The Best parameters according to grid search are: {'SVM__C': 0.1}
Best training score from grid search pipeline: 83.91666666666666 %
Best test score from grid search pipeline: 79.5 %
Training score for LIBSVM with best parameters: 84.75 %
Test score for LIBSVM with best parameters: 79.83333333333333 %
```



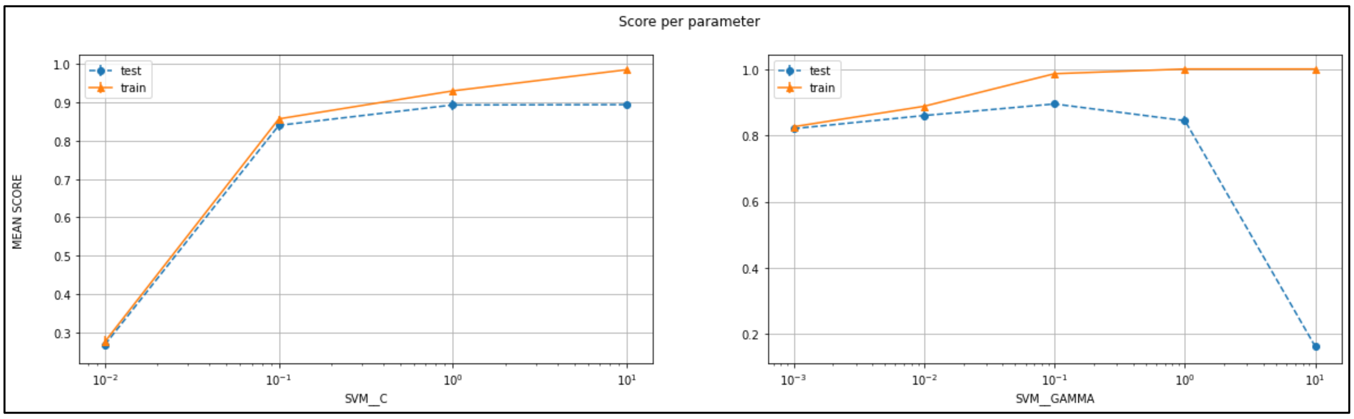


```
################################################################################
Number of features: 10
################################################################################
Number of training examples: (2400, 10) (2400, 1)
Number of test examples: (600, 10) (600, 1)
-------------------------LIBSVM for SIGMOID for 10 features----------------------------
The Best parameters according to grid search are: {'SVM__C': 10.0, 'SVM__gamma':
0.01}
Best training score from grid search pipeline: 83.83333333333334 %
Best test score from grid search pipeline: 79.83333333333333 %
Training score for LIBSVM with best parameters: 83.0 %
Test score for LIBSVM with best parameters: 79.16666666666666 %
```

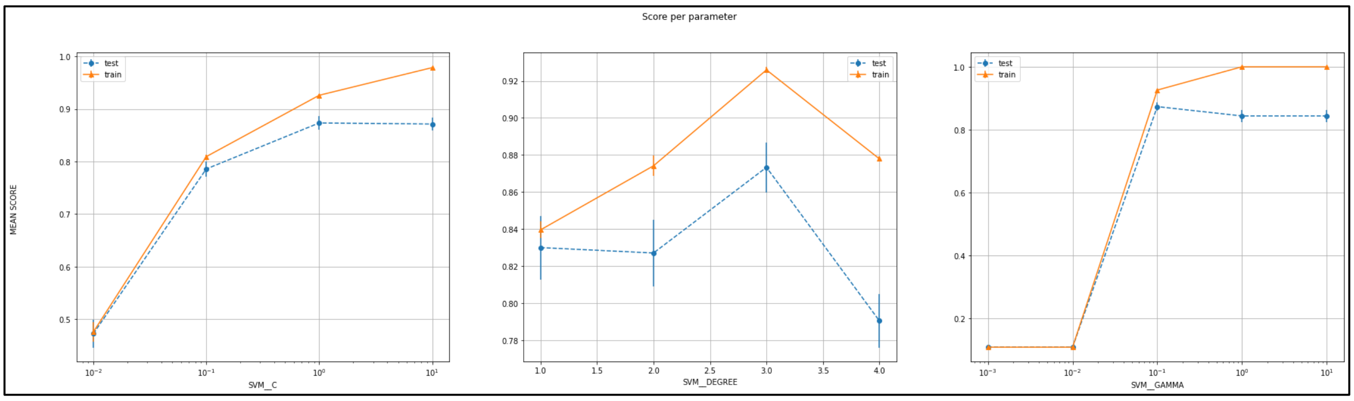


```
################################################################################
Number of features: 10
################################################################################
Number of training examples: (2400, 10) (2400, 1)
Number of test examples: (600, 10) (600, 1)
-------------------------LIBSVM for RBF for 10 features----------------------------
```

The Best parameters according to grid search are: {'SVM__C': 10.0, 'SVM__gamma': 0.1}
Best training score from grid search pipeline: 98.41666666666666 %
Best test score from grid search pipeline: 88.0 %
Training score for LIBSVM with best parameters: 99.91666666666667 %
Test score for LIBSVM with best parameters: 89.66666666666666 %

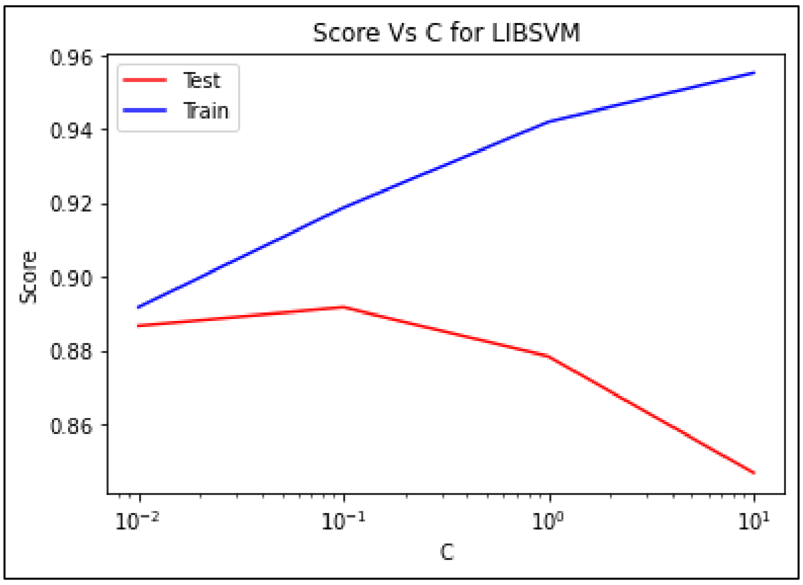


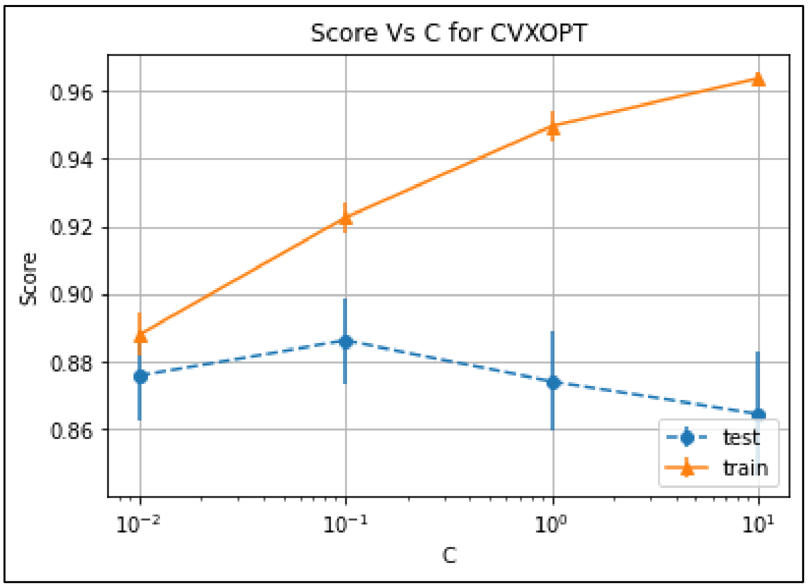
##
Number of features: 10
##
Number of training examples: (2400, 10) (2400, 1)
Number of test examples: (600, 10) (600, 1)
-------------------------LIBSVM for **POLY** for 10 features----------------------------
The Best parameters according to grid search are: {'SVM__C': 1.0, 'SVM__degree': 3.0, 'SVM__gamma': 0.1}
Best training score from grid search pipeline: 92.41666666666667 %
Best test score from grid search pipeline: 85.0 %
Training score for LIBSVM with best parameters: 97.91666666666666 %
Test score for LIBSVM with best parameters: 86.66666666666667 %



##
Number of features: 25
##
Number of training examples: (2400, 25) (2400, 1)

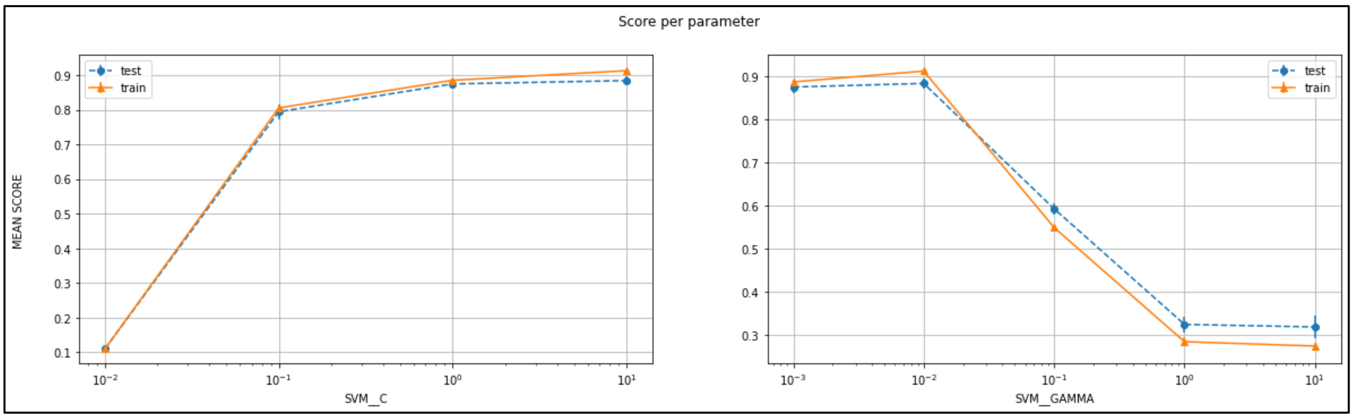
Number of test examples: (600, 25) (600, 1)
-------------------------LIBSVM for **LINEAR** for 25 features----------------------------
The Best parameters according to grid search are: {'SVM__C': 0.1}
Best training score from grid search pipeline: 91.875 %
Best test score from grid search pipeline: 89.16666666666667 %
Training score for LIBSVM with best parameters: 92.375 %
Test score for LIBSVM with best parameters: 89.83333333333333 %



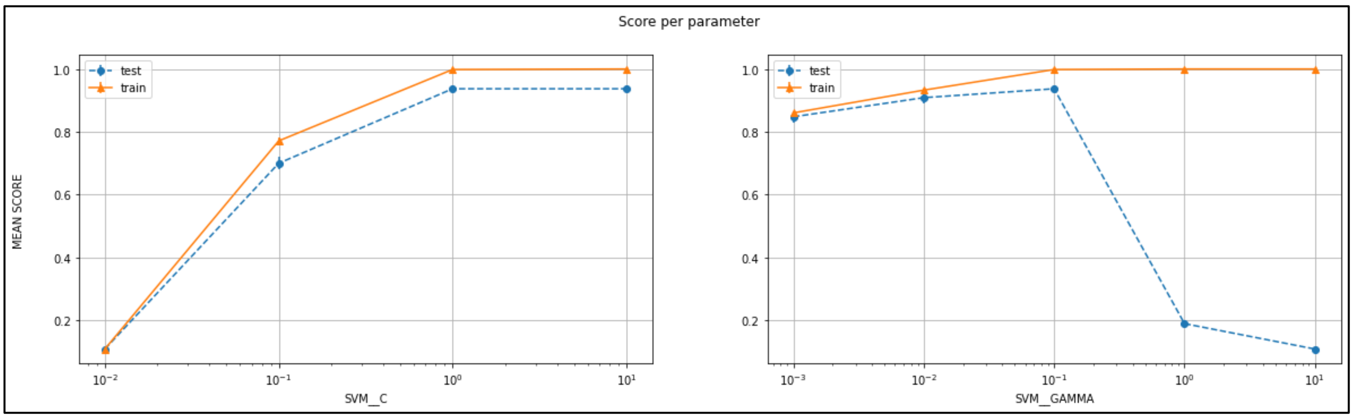


###
Number of features: 25
###
Number of training examples: (2400, 25) (2400, 1)
Number of test examples: (600, 25) (600, 1)
-------------------------LIBSVM for **SIGMOID** for 25 features----------------------------
The Best parameters according to grid search are: {'SVM__C': 10.0, 'SVM__gamma': 0.01}
Best training score from grid search pipeline: 90.91666666666667 %
Best test score from grid search pipeline: 89.0 %

```
Training score for LIBSVM with best parameters: 90.25 %
Test score for LIBSVM with best parameters: 87.66666666666667 %
```

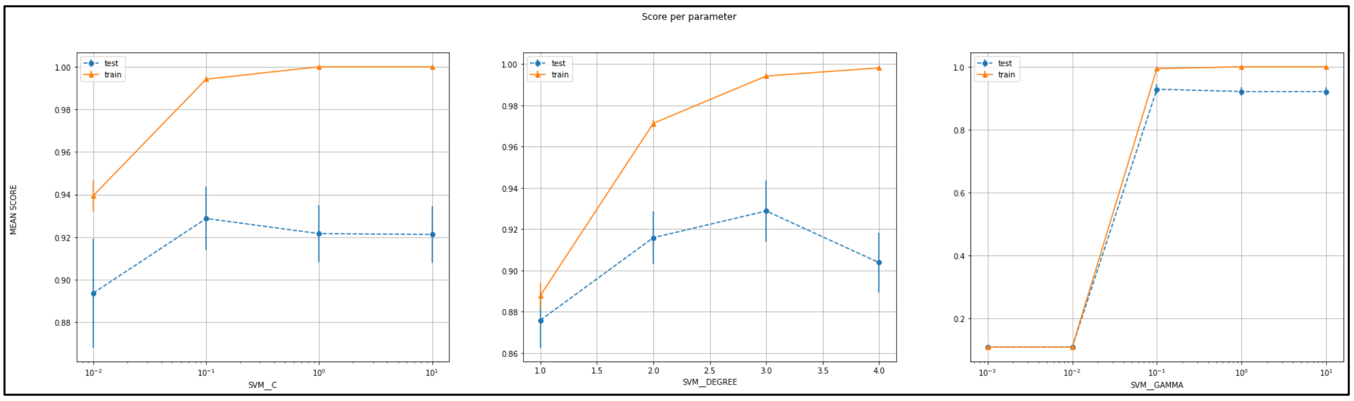


```
##############################################################################
Number of features: 25
##############################################################################
Number of training examples: (2400, 25) (2400, 1)
Number of test examples: (600, 25) (600, 1)
-------------------------LIBSVM for RBF for 25 features----------------------------
The Best parameters according to grid search are: {'SVM__C': 1.0, 'SVM__gamma':
0.1}
Best training score from grid search pipeline: 99.875 %
Best test score from grid search pipeline: 94.16666666666667 %
Training score for LIBSVM with best parameters: 99.83333333333333 %
Test score for LIBSVM with best parameters: 93.66666666666667 %
```



```
##############################################################################
Number of features: 25
##############################################################################
Number of training examples: (2400, 25) (2400, 1)
Number of test examples: (600, 25) (600, 1)
-------------------------LIBSVM for POLY for 25 features----------------------------
The Best parameters according to grid search are: {'SVM__C': 0.1, 'SVM__degree':
3.0, 'SVM__gamma': 0.1}
```

Best training score from grid search pipeline: 99.41666666666666 %
Best test score from grid search pipeline: 92.83333333333333 %
Training score for LIBSVM with best parameters: 99.75 %
Test score for LIBSVM with best parameters: 93.83333333333333 %



Summary:

Num of features	Kernel	Best C	Best gamma	Best degree	Training accuracy	Test accuracy
10	Linear	0.1	-	-	84.75%	79.83%
10	Poly	1	0.1	3	97.92%	86.67%
10	Sigmoid	10	0.01	-	83.83%	79.83%
10	**RBF**	**10**	**0.1**	**-**	**99.92%**	**89.67%**
25	Linear	0.1	-	-	92.38%	89.83%
25	Poly	0.1	0.1	3	99.75%	93.83%
25	Sigmoid	10	0.01	-	90.92%	89%
25	**RBF**	**1**	**0.1**	**-**	**99.88%**	**94.17%**

Table 6

The values written in purple are with the standard scaler in the pipeline and the one in the black are without it. The best of the 2 scores is reported. The ones in bold represent the best values for particular number of features.
In both the cases, RBF outperforms all the other models by some significant value.
It should also be noted that in both the cases, sigmoid works better when a standard scaler is NOT used. This could come in handy for the next part of the assignment.
In terms of computation time, RBF and poly kernels take a lot more time than the other kernels and for purposes of fast results, these should be avoided (especially during hyperparameter sweeps)

Other conclusions include the visible increase in score by as much as 10% as the number of features increase to 25. Since both the training and test scores increased with the number of features, it's not just that the models were overfitting earlier. But that the models simply have more attributes to classify the data.

Finally, we should also observe how the accuracy scores (especially for the test set) differ vastly from the binary classification task. This is indeed expected because now

there simply are more classes to classify which makes it much more prone to errors.
The SVM.SVC from sklearn that I use is based on the Libsvm library and thus it uses a one-vs-one technique for multi-class classification. i.e., it generates ${}^{N}C_{2}$ classifiers (where N is the number of classes) for categorizing the data.

The gamma parameter for the sigmoid kernel showed the same trend as observed earlier. It first increases the accuracy and then decreases it.
For other kernels and parameters, it’s almost always the case that as the value increases, the model goes from a state of underfitting to overfitting.
One noteworthy mention here is the degree parameter for poly kernel with 10 features. The training accuracy along with the test accuracy reaches a maximum and then starts decreasing. I can only speculate the reason to have something to do with the even degree for the kernel.

Part 1B

So… I used the pseudocode given in the supplementary material given with this assignment to implement the simplified SMO algorithm and ran that, Libsvm and CVXOPT in the same block to rightfully compare their computation times and accuracy. I chose value of C to be 0.1 as it gave the best performance earlier. The results are as follows:

```
################################################################################
Labels: 0 , 1
Number of features: 10
Current kernel: linear
################################################################################
Number of training examples: (484, 10) (484, 1)
Number of test examples: (121, 10) (121, 1)
Training accuracy by simplified SMO is: 99.79338842975206 %
Test accuracy by simplified SMO is: 99.17355371900827 %
Time taken by simplified SMO for labels (0,1) for 10 features is 5.328125 seconds

Training accuracy for LIBSVM is: 99.79338842975206 %
Test accuracy for LIBSVM is: 99.17355371900827 %
Time taken by LIBSVM for labels (0,1) for 10 features is 0.0 seconds

Test accuracy for CVXOPT is: 99.17355371900827 %
Time taken by CVXOPT for labels (0,1) for 10 features is 0.578125 seconds


################################################################################
Labels: 0 , 1
Number of features: 10
Current kernel: sigmoid
################################################################################
Number of training examples: (484, 10) (484, 1)
Number of test examples: (121, 10) (121, 1)
Test accuracy by simplified SMO is: 98.34710743801652 %
Time taken by simplified SMO for labels (0,1) for 10 features is 5.421875 seconds

Training accuracy for LIBSVM is: 99.58677685950413 %
Test accuracy for LIBSVM is: 100.0 %
Time taken by LIBSVM for labels (0,1) for 10 features is 0.0 seconds

Test accuracy for CVXOPT is: 100.0 %
Time taken by CVXOPT for labels (0,1) for 10 features is 0.484375 seconds


################################################################################
Labels: 0 , 1
Number of features: 25
Current kernel: linear
################################################################################
```

```
Number of training examples: (484, 25) (484, 1)
Number of test examples: (121, 25) (121, 1)
Training accuracy by simplified SMO is: 100.0 %
Test accuracy by simplified SMO is: 99.17355371900827 %
Time taken by simplified SMO for labels (0,1) for 25 features is 106.640625 seconds

Training accuracy for LIBSVM is: 100.0 %
Test accuracy for LIBSVM is: 99.17355371900827 %
Time taken by LIBSVM for labels (0,1) for 25 features is 0.015625 seconds

Test accuracy for CVXOPT is: 99.17355371900827 %
Time taken by CVXOPT for labels (0,1) for 25 features is 0.359375 seconds


##########################################################################
Labels: 0 , 1
Number of features: 25
Current kernel: sigmoid
##########################################################################
Number of training examples: (484, 25) (484, 1)
Number of test examples: (121, 25) (121, 1)
Test accuracy by simplified SMO is: 99.17355371900827 %
Time taken by simplified SMO for labels (0,1) for 25 features is 23.640625 seconds

Training accuracy for LIBSVM is: 99.79338842975206 %
Test accuracy for LIBSVM is: 99.17355371900827 %
Time taken by LIBSVM for labels (0,1) for 25 features is 0.015625 seconds

Test accuracy for CVXOPT is: 100.0 %
Time taken by CVXOPT for labels (0,1) for 25 features is 0.546875 seconds


##########################################################################
Labels: 4 , 6
Number of features: 10
Current kernel: linear
##########################################################################
Number of training examples: (472, 10) (472, 1)
Number of test examples: (119, 10) (119, 1)
Training accuracy by simplified SMO is: 97.66949152542372 %
Test accuracy by simplified SMO is: 100.0 %
Time taken by simplified SMO for labels (4,6) for 10 features is 23.109375 seconds

Training accuracy for LIBSVM is: 97.88135593220339 %
Test accuracy for LIBSVM is: 100.0 %
Time taken by LIBSVM for labels (4,6) for 10 features is 0.015625 seconds

Test accuracy for CVXOPT is: 99.15966386554622 %
Time taken by CVXOPT for labels (4,6) for 10 features is 0.640625 seconds
```

```
##########################################################################
Labels: 4 , 6
Number of features: 10
Current kernel: sigmoid
##########################################################################
Number of training examples: (472, 10) (472, 1)
Number of test examples: (119, 10) (119, 1)
Test accuracy by simplified SMO is: 100.0 %
Time taken by simplified SMO for labels (4,6) for 10 features is 3.6875 seconds

Training accuracy for LIBSVM is: 97.66949152542372 %
Test accuracy for LIBSVM is: 100.0 %
Time taken by LIBSVM for labels (4,6) for 10 features is 0.0 seconds

Test accuracy for CVXOPT is: 100.0 %
Time taken by CVXOPT for labels (4,6) for 10 features is 0.40625 seconds


##########################################################################
Labels: 4 , 6
Number of features: 25
Current kernel: linear
##########################################################################
Number of training examples: (472, 25) (472, 1)
Number of test examples: (119, 25) (119, 1)
Training accuracy by simplified SMO is: 98.9406779661017 %
Test accuracy by simplified SMO is: 100.0 %
Time taken by simplified SMO for labels (4,6) for 25 features is 106.3125 seconds

Training accuracy for LIBSVM is: 98.9406779661017 %
Test accuracy for LIBSVM is: 100.0 %
Time taken by LIBSVM for labels (4,6) for 25 features is 0.015625 seconds

Test accuracy for CVXOPT is: 100.0 %
Time taken by CVXOPT for labels (4,6) for 25 features is 0.53125 seconds


##########################################################################
Labels: 4 , 6
Number of features: 25
Current kernel: sigmoid
##########################################################################
Number of training examples: (472, 25) (472, 1)
Number of test examples: (119, 25) (119, 1)
Test accuracy by simplified SMO is: 100.0 %
Time taken by simplified SMO for labels (4,6) for 25 features is 22.28125 seconds

Training accuracy for LIBSVM is: 98.51694915254238 %
Test accuracy for LIBSVM is: 100.0 %
Time taken by LIBSVM for labels (4,6) for 25 features is 0.0 seconds
```

```
Test accuracy for CVXOPT is: 100.0 %
Time taken by CVXOPT for labels (4,6) for 25 features is 0.34375 seconds


##########################################################################
Labels: 8 , 9
Number of features: 10
Current kernel: linear
##########################################################################
Number of training examples: (454, 10) (454, 1)
Number of test examples: (114, 10) (114, 1)
Training accuracy by simplified SMO is: 94.27312775330397 %
Test accuracy by simplified SMO is: 93.85964912280701 %
Time taken by simplified SMO for labels (8,9) for 10 features is 16.90625 seconds

Training accuracy for LIBSVM is: 94.27312775330397 %
Test accuracy for LIBSVM is: 93.85964912280701 %
Time taken by LIBSVM for labels (8,9) for 10 features is 0.0 seconds

Test accuracy for CVXOPT is: 92.98245614035088 %
Time taken by CVXOPT for labels (8,9) for 10 features is 0.578125 seconds


##########################################################################
Labels: 8 , 9
Number of features: 10
Current kernel: sigmoid
##########################################################################
Number of training examples: (454, 10) (454, 1)
Number of test examples: (114, 10) (114, 1)
Test accuracy by simplified SMO is: 92.10526315789474 %
Time taken by simplified SMO for labels (8,9) for 10 features is 3.921875 seconds

Training accuracy for LIBSVM is: 92.95154185022027 %
Test accuracy for LIBSVM is: 93.85964912280701 %
Time taken by LIBSVM for labels (8,9) for 10 features is 0.0 seconds

Test accuracy for CVXOPT is: 92.10526315789474 %
Time taken by CVXOPT for labels (8,9) for 10 features is 0.484375 seconds


##########################################################################
Labels: 8 , 9
Number of features: 25
Current kernel: linear
##########################################################################
Number of training examples: (454, 25) (454, 1)
Number of test examples: (114, 25) (114, 1)
Training accuracy by simplified SMO is: 97.13656387665198 %
Test accuracy by simplified SMO is: 97.36842105263158 %
Time taken by simplified SMO for labels (8,9) for 25 features is 205.296875 seconds
```

```
Training accuracy for LIBSVM is: 97.13656387665198 %
Test accuracy for LIBSVM is: 97.36842105263158 %
Time taken by LIBSVM for labels (8,9) for 25 features is 0.0 seconds

Test accuracy for CVXOPT is: 97.36842105263158 %
Time taken by CVXOPT for labels (8,9) for 25 features is 0.46875 seconds


###########################################################################
Labels: 8 , 9
Number of features: 25
Current kernel: sigmoid
###########################################################################
Number of training examples: (454, 25) (454, 1)
Number of test examples: (114, 25) (114, 1)
Test accuracy by simplified SMO is: 96.49122807017544 %
Time taken by simplified SMO for labels (8,9) for 25 features is 20.03125 seconds

Training accuracy for LIBSVM is: 96.47577092511013 %
Test accuracy for LIBSVM is: 96.49122807017544 %
Time taken by LIBSVM for labels (8,9) for 25 features is 0.0 seconds

Test accuracy for CVXOPT is: 96.49122807017544 %
Time taken by CVXOPT for labels (8,9) for 25 features is 0.296875 seconds
```

For linear kernel:

Label Pair	**Num of features**	**Algorithm**	**Training Accuracy**	**Test Accuracy**	**Time Taken (seconds)**
(0, 1)	10	Simplified SMO	99.79%	99.17%	5.328
(0, 1)	10	LIBSVM	99.79%	99.17%	0.0
(0, 1)	10	CVXOPT	-	99.17%	0.578
(0, 1)	25	Simplified SMO	100%	99.17%	106.644
(0, 1)	25	LIBSVM	100%	99.17%	0.016
(0, 1)	25	CVXOPT	-	99.17%	0.359
(4,6)	10	Simplified SMO	97.67%	100%	23.10
(4,6)	10	LIBSVM	97.88%	100%	0.015
(4,6)	10	CVXOPT	-	99.16%	0.640
(4,6)	25	Simplified SMO	98.94%	100%	106.313
(4,6)	25	LIBSVM	98.94%	100%	0.0
(4,6)	25	CVXOPT	-	100%	0.4375
(8,9)	10	Simplified SMO	94.27%	93.86%	16.907
(8,9)	10	LIBSVM	94.27%	93.86%	0.0

(8,9)	10	CVXOPT	-	92.98%	0.578
(8,9)	25	Simplified SMO	97.14%	97.37%	205.297
(8,9)	25	LIBSVM	97.14%	97.37%	0.0
(8,9)	25	CVXOPT	-	97.37%	0.469

Table 7

For sigmoid kernel:

Label Pair	**Num of features**	**Algorithm**	**Training Accuracy**	**Test Accuracy**	**Time Taken (seconds)**
(0, 1)	10	Simplified SMO	99.38%	98.35%	5.422
(0, 1)	10	LIBSVM	99.58%	100%	0.0
(0, 1)	10	CVXOPT	-	100%	0.484
(0, 1)	25	Simplified SMO	99.79%	99.17%	23.64
(0, 1)	25	LIBSVM	99.79%	99.17%	0.016
(0, 1)	25	CVXOPT	-	100%	0.546
(4,6)	10	Simplified SMO	97.88%	100%	3.688
(4,6)	10	LIBSVM	97.67%	100%	0.00
(4,6)	10	CVXOPT	-	100%	0.4
(4,6)	25	Simplified SMO	98.94%	100%	22.281
(4,6)	25	LIBSVM	98.52%	100%	0.0
(4,6)	25	CVXOPT	-	100%	0.344
(8,9)	10	Simplified SMO	93.17%	92.11%	3.92
(8,9)	10	LIBSVM	92.95%	93.86%	0.0
(8,9)	10	CVXOPT	-	92.11%	0.484
(8,9)	25	Simplified SMO	95.82%	96.49%	20.031
(8,9)	25	LIBSVM	96.47%	96.49%	0.0
(8,9)	25	CVXOPT	-	96.49%	0.30

Table 8

As we can see in the table, the time taken by the simplified SMO algorithm was much more than the other algorithms. This is most probably because simplified SMO is an iterative algorithm whereas others use vector operations entirely. The accuracy and running times will also depend on other hyperparameters like tolerance and max passes but those haven't been covered here.

There is barely any difference in the accuracies of the 3 algorithms. However, if we were to consider absolute numbers, SMO did manage to outperform at least in training accuracies (in most cases). And as mentioned before, the scores can further be improved by sweeping different values of the hyperparameters tolerance and max number of passes.

One reason why people might prefer simplified SMO over other algorithms is that it has space complexity of O(n) while the others have O(n^2)

I was able to implement simplified SMO for different kernels to return the alphas and offset, but I was unable to use these and get new predictions for these non-linear kernels.

The alphas and b are used accordingly if the type is linear or non-linear. The code for linear is fairly straightforward, but for the non-linear, my code is derived from the non-linear one. One thing that is different is that I am not limiting removing the alphas with small values, but just assuming that they would automatically decrease the effect of that vector in the final prediction

Part 1C

I tried to follow the pseudo-code given in the research paper but my offset value was blowing up to 1e16 every time. After taking help from the simplified SMO paper and building upon it, I was able to get some really good values, the results of which are presented below:

```
##############################################################################
Labels: 0 , 1
Number of features: 10
Current kernel: linear
##############################################################################
Number of training examples: (484, 10) (484, 1)
Number of test examples: (121, 10) (121, 1)
Training accuracy by SMO is: 99.79338842975206 %
Test accuracy by SMO is: 100.0 %
Time taken by simplified SMO for labels (0,1) for 10 features is 36.484375 seconds


##############################################################################
Labels: 0 , 1
Number of features: 10
Current kernel: sigmoid
##############################################################################
Number of training examples: (484, 10) (484, 1)
Number of test examples: (121, 10) (121, 1)
Training accuracy by simplified SMO is: 99.58677685950413 %
Test accuracy by simplified SMO is: 100.0 %
Time taken by simplified SMO for labels (0,1) for 10 features is 5.421875 seconds


##############################################################################
Labels: 0 , 1
Number of features: 25
Current kernel: linear
##############################################################################
Number of training examples: (484, 25) (484, 1)
Number of test examples: (121, 25) (121, 1)
Training accuracy by SMO is: 100.0 %
Test accuracy by SMO is: 99.17355371900827 %
Time taken by simplified SMO for labels (0,1) for 25 features is 12.28125 seconds


##############################################################################
Labels: 0 , 1
Number of features: 25
Current kernel: sigmoid
##############################################################################
Number of training examples: (484, 25) (484, 1)
Number of test examples: (121, 25) (121, 1)
Training accuracy by simplified SMO is: 99.79338842975207 %
Test accuracy by simplified SMO is: 99.17355371900827 %
Time taken by simplified SMO for labels (0,1) for 25 features is 11.96875 seconds


########################################################################
Labels: 4 , 6
Number of features: 10
Current kernel: linear
########################################################################
Number of training examples: (472, 10) (472, 1)
Number of test examples: (119, 10) (119, 1)
Training accuracy by SMO is: 98.30508474576271 %
Test accuracy by SMO is: 97.47899159663865 %
Time taken by simplified SMO for labels (4,6) for 10 features is 17.8125 seconds


########################################################################
Labels: 4 , 6
Number of features: 10
Current kernel: sigmoid
########################################################################
Number of training examples: (472, 10) (472, 1)
Number of test examples: (119, 10) (119, 1)
Training accuracy by simplified SMO is: 97.66949152542372 %
Test accuracy by simplified SMO is: 100.0 %
Time taken by simplified SMO for labels (4,6) for 10 features is 13.046875 seconds


########################################################################
Labels: 4 , 6
Number of features: 25
Current kernel: linear
########################################################################
Number of training examples: (472, 25) (472, 1)
Number of test examples: (119, 25) (119, 1)
Training accuracy by SMO is: 100.0 %
Test accuracy by SMO is: 98.31932773109243 %
Time taken by simplified SMO for labels (4,6) for 25 features is 35.9375 seconds


########################################################################
Labels: 4 , 6
Number of features: 25
Current kernel: sigmoid
########################################################################
Number of training examples: (472, 25) (472, 1)
Number of test examples: (119, 25) (119, 1)
Training accuracy by simplified SMO is: 98.51694915254237 %
Test accuracy by simplified SMO is: 100.0 %
```

```
Time taken by simplified SMO for labels (4,6) for 25 features is 69.28125 seconds


################################################################################
Labels: 8 , 9
Number of features: 10
Current kernel: linear
################################################################################
Number of training examples: (454, 10) (454, 1)
Number of test examples: (114, 10) (114, 1)
Training accuracy by SMO is: 94.7136563876652 %
Test accuracy by SMO is: 95.6140350877193 %
Time taken by simplified SMO for labels (8,9) for 10 features is 72.015625 seconds


################################################################################
Labels: 8 , 9
Number of features: 10
Current kernel: sigmoid
################################################################################
Number of training examples: (454, 10) (454, 1)
Number of test examples: (114, 10) (114, 1)
Training accuracy by simplified SMO is: 92.95154185022027 %
Test accuracy by simplified SMO is: 93.85964912280701 %
Time taken by simplified SMO for labels (8,9) for 10 features is 10.71875 seconds


################################################################################
Labels: 8 , 9
Number of features: 25
Current kernel: linear
################################################################################
Number of training examples: (454, 25) (454, 1)
Number of test examples: (114, 25) (114, 1)
Training accuracy by SMO is: 98.6784140969163 %
Test accuracy by SMO is: 96.49122807017544 %
Time taken by simplified SMO for labels (8,9) for 25 features is 370.078125 seconds


################################################################################
Labels: 8 , 9
Number of features: 25
Current kernel: sigmoid
################################################################################
Number of training examples: (454, 25) (454, 1)
Number of test examples: (114, 25) (114, 1)
Training accuracy by simplified SMO is: 96.47577092511013 %
Test accuracy by simplified SMO is: 96.49122807017544 %
Time taken by simplified SMO for labels (8,9) for 25 features is 28.40625 seconds
```

For linear kernel:

Label Pair	Num of features	Algorithm	Training Accuracy	Test Accuracy	Time Taken (seconds)
(0, 1)	10	SMO	99.79%	100%	36.48
(0, 1)	10	LIBSVM	99.79%	99.17%	0.0
(0, 1)	10	CVXOPT	-	99.17%	0.578
(0, 1)	25	SMO	100%	99.17%	12.28
(0, 1)	25	LIBSVM	100%	99.17%	0.016
(0, 1)	25	CVXOPT	-	99.17%	0.359
(4,6)	10	SMO	97.30%	97.48%	17.82
(4,6)	10	LIBSVM	97.88%	100%	0.015
(4,6)	10	CVXOPT	-	99.16%	0.640
(4,6)	25	SMO	100%	98.32%	35.94
(4,6)	25	LIBSVM	98.94%	100%	0.0
(4,6)	25	CVXOPT	-	100%	0.4375
(8,9)	10	SMO	94.71%	95.61%	72.02
(8,9)	10	LIBSVM	94.27%	93.86%	0.0
(8,9)	10	CVXOPT	-	92.98%	0.578
(8,9)	25	SMO	98.68%	96.49%	370.07
(8,9)	25	LIBSVM	97.14%	97.37%	0.0
(8,9)	25	CVXOPT	-	97.37%	0.469

Table 9

For sigmoid kernel:

Label Pair	Num of features	Algorithm	Training Accuracy	Test Accuracy	Time Taken (seconds)
(0, 1)	10	SMO	99.59%	100%	5.422
(0, 1)	10	LIBSVM	99.58%	100%	0.0
(0, 1)	10	CVXOPT	-	100%	0.484
(0, 1)	25	SMO	99.79%	99.17%	11.97
(0, 1)	25	LIBSVM	99.79%	99.17%	0.016
(0, 1)	25	CVXOPT	-	100%	0.546
(4,6)	10	SMO	97.67%	100%	13.04
(4,6)	10	LIBSVM	97.67%	100%	0.00
(4,6)	10	CVXOPT	-	100%	0.4
(4,6)	25	SMO	98.52%	100%	69.281
(4,6)	25	LIBSVM	98.52%	100%	0.0
(4,6)	25	CVXOPT	-	100%	0.344
(8,9)	10	SMO	92.95%	93.86%	10.72
(8,9)	10	LIBSVM	92.95%	93.86%	0.0
(8,9)	10	CVXOPT	-	92.11%	0.484
(8,9)	25	SMO	96.47%	96.49%	20.031
(8,9)	25	LIBSVM	96.47%	96.49%	0.0
(8,9)	25	CVXOPT	-	96.49%	0.30

Table 10

We notice that for the full SMO algorithm, the time taken is more than the simplified version. However, its accuracy exactly matches the one by Libsvm (And therefore CVXOPT, as they both had the same accuracies) for the sigmoid kernel. This, loosely, confirms that the full SMO algorithm, is in fact correct.

In order to choose which Lagrange multipliers to optimize, I, very obviously, read the research paper. I found that SMO first computes the constraints on these multipliers and then solves for the constrained minimum. In my code, i1 denotes the first multiplier and the i2 denotes the second.

Now we know that each step of the SMO will optimize 2 Lagrange multipliers (α_1 and α_2). The objective function mentioned in the paper can be defined as

$$\psi = \frac{1}{2} K_{11}\alpha_1^2 + \frac{1}{2} K_{22}\alpha_2^2 + sK_{12}\alpha_1\alpha_2 + y_1\alpha_1 v_1 + y_2\alpha_2 v_2 - \alpha_1 - \alpha_2 + \psi_{constant}$$

Where K_{ii} is the kernel function with parameters $X_i \ and \ X_j$
And $v_i = \sum_{j=3}^{N} y_j \alpha_j^* K_{ij} = u_i + b^* - y_1\alpha_1^* K_{1i} - y_2\alpha_2^* K_{2i}$

Here, the starred variables are because this is an iterative algorithm. They denote the values from previous iteration.

In the full SMO, we find minima along a line with constraint:

$$\alpha_1 + s\alpha_2 = \alpha_1^* + s\alpha_2^* = w$$

Therefore, we can substitute the value of α_1 to reduce the expression to only depend on α_2

To find the minima, we differentiate the objective function w.r.t α_2 and equate it to 0 (Assuming second derivative to be 0)

i.e., $-sK_{11}(w - s\alpha_2) + K_{22}\alpha_2 + K_{12}\alpha_2 + sK_{12}(w - s\alpha_2) - y_2 v_1 + s + y_2 v_2 - 1 = 0$

$$\rightarrow \alpha_2(K_{11} + K_{22} - 2K_{12}) = s(K_{11} - K_{22})w + y_2(v_1 - v_2) + 1 - s$$

$$\rightarrow \alpha_2(K_{11} + K_{22} - 2K_{12}) = \alpha_2^*(K_{11} + K_{22} - 2K_{12}) + y_2(u_1 - u_2 + y_2 - y\_2)$$

$$\rightarrow \boldsymbol{\alpha_2^{new} = \alpha_2 + \frac{y_2(E_1 - E_2)}{\eta}}$$

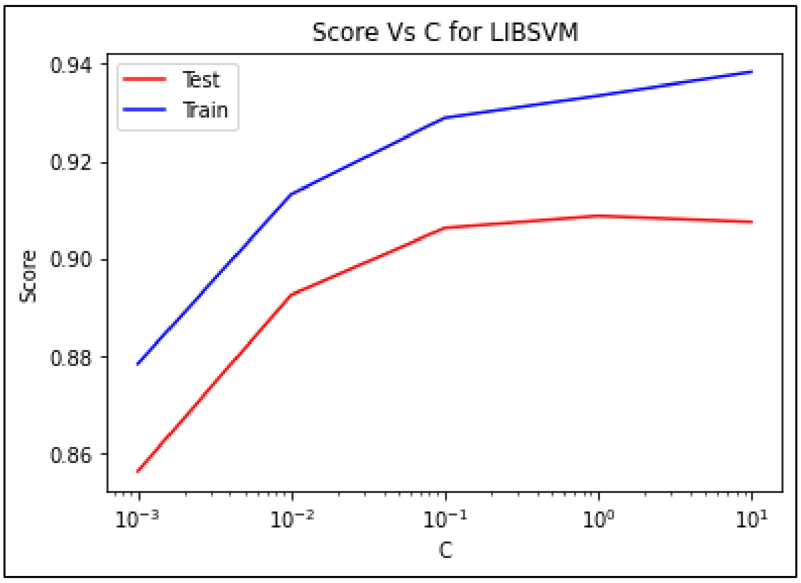
Part 2

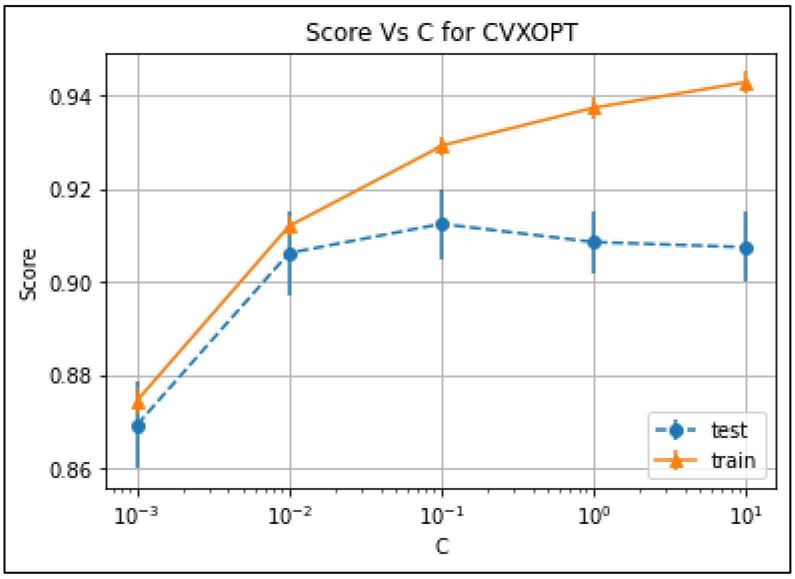
My initial basic approach was to simply do a hyperparameter sweep for different kernels for the LIBSVM library. The pipeline contained a standard scaler followed by the SVM.SVC model. The results are as follows:

Number of training examples: (7200, 25) (7200, 1)
Number of test examples: (800, 25) (800, 1)

-------------------------LIBSVM for LINEAR----------------------------

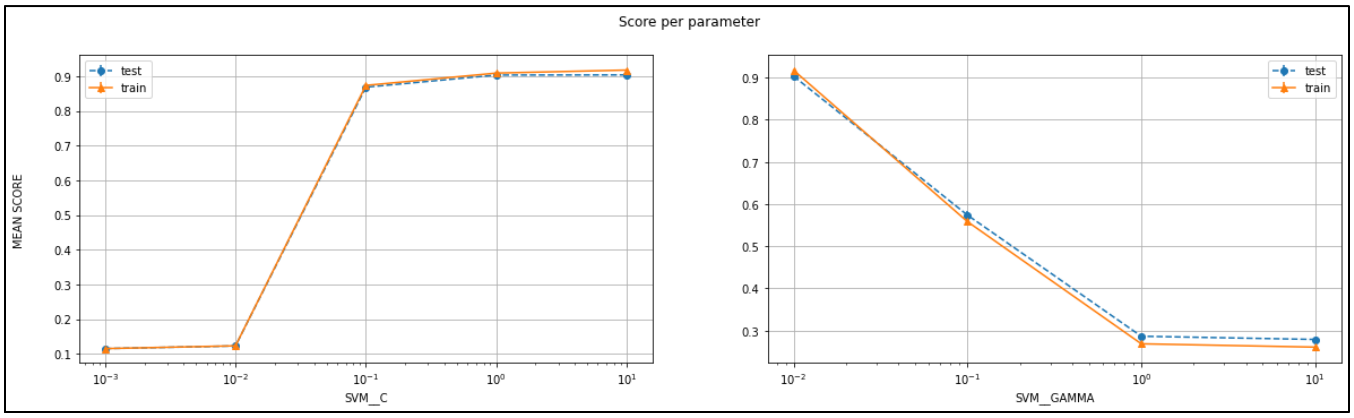
Grid scaled training score for libsvm is: 92.88888888888889 %
Grid scaled test score for libsvm is: 90.625 %
The Best parameters according to grid search are: {'SVM__C': 0.1}





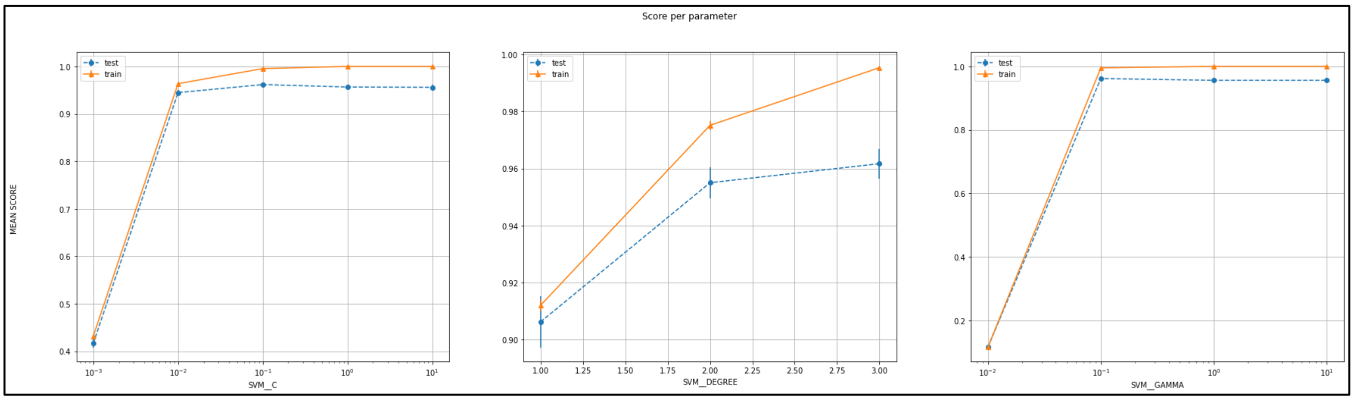
-------------------------LIBSVM for SIGMOID----------------------------

Grid scaled training score for libsvm is: 91.625 %
Grid scaled test score for libsvm is: 90.25 %
The Best parameters according to grid search are: {'SVM__C': 10.0, 'SVM__gamma': 0.01}



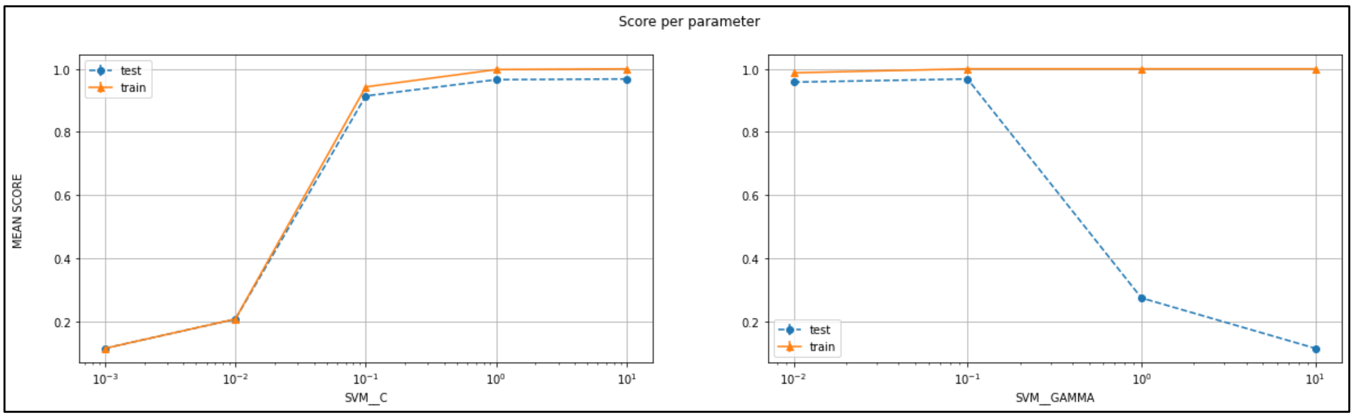
--------------------------LIBSVM for POLY----------------------------

Grid scaled training score for libsvm is: 99.58333333333333 %
Grid scaled test score for libsvm is: 96.5 %
The Best parameters according to grid search are: {'SVM__C': 0.1, 'SVM__degree': 3.0, 'SVM__gamma': 0.1}



--------------------------LIBSVM for RBF----------------------------

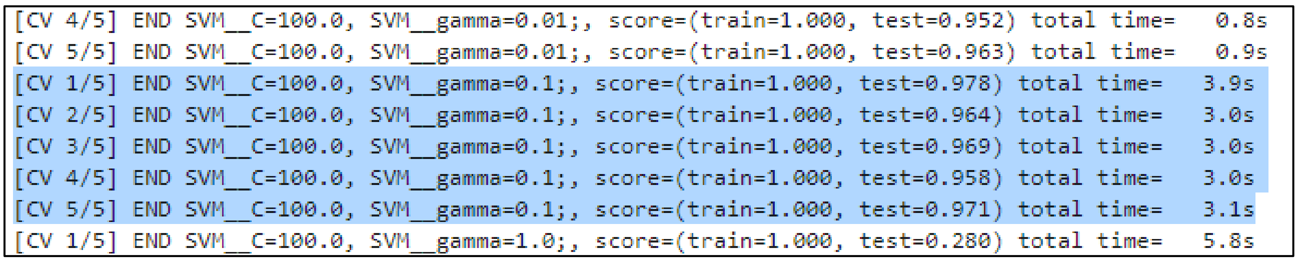
Grid scaled training score for libsvm is: 100.0 %
Grid scaled test score for libsvm is: 96.875 %
The Best parameters according to grid search are: {'SVM__C': 10.0, 'SVM__gamma': 0.1}



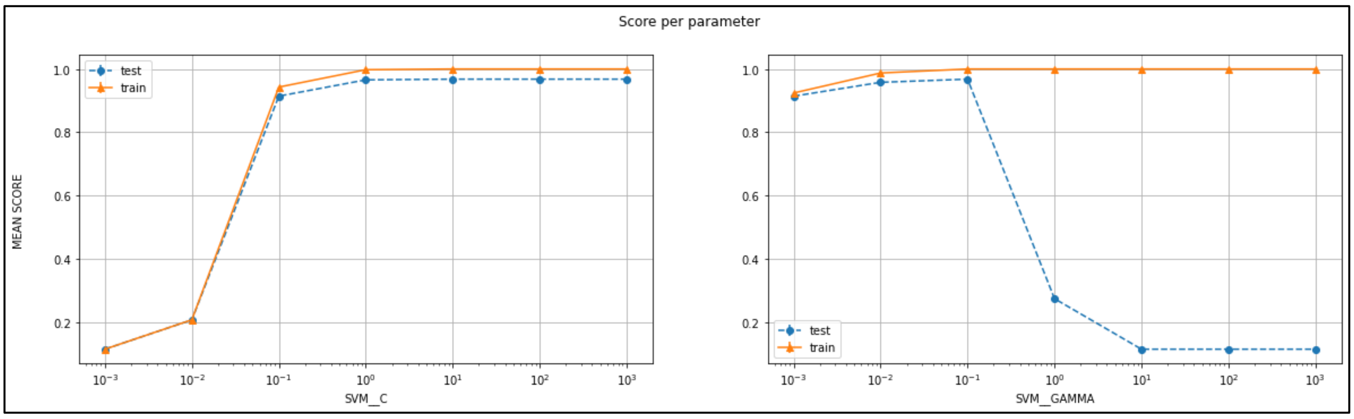
The best score was, as can be seen, given by RBF kernel, with parameters C=10 and gamma=0.1
This (First_draft.csv) gave me an accuracy of 95.5% on the leaderboard

Since 15 submissions were allowed, I tried out the poly one too, with parameters C=0.1, degree=3, gamma=0.1
This (second_draft.csv) gave a worse accuracy of 94.5%

Therefore, I decided to do a bigger sweep on just the RBF kernel. The results were as f ollows:
And eyeballing the different scores, I found C=100 to be better than C=10 for the same gamma of 0.1. This did, in fact, give me a better performance (95.65%).

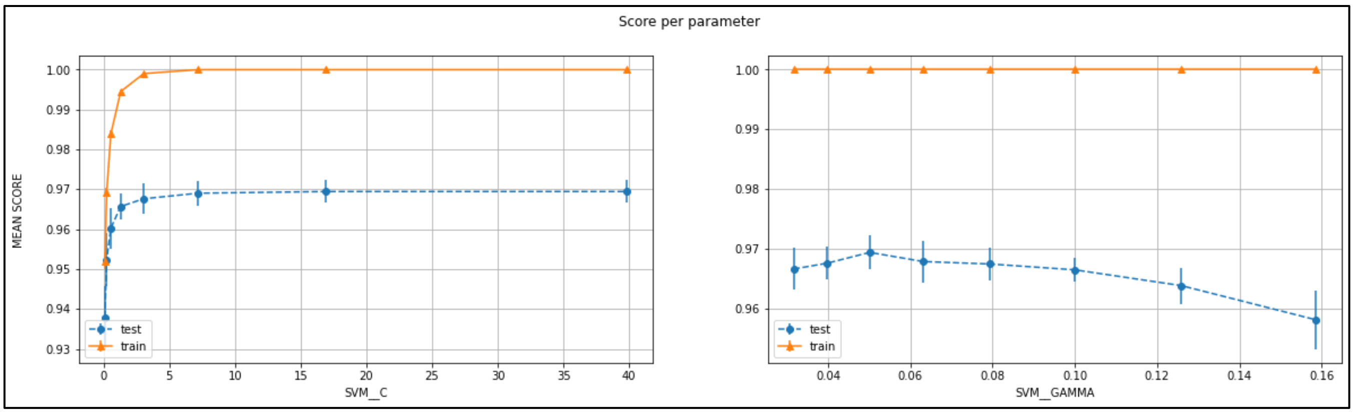


```
[CV 4/5] END SVM__C=100.0, SVM__gamma=0.01;, score=(train=1.000, test=0.952) total time=   0.8s
[CV 5/5] END SVM__C=100.0, SVM__gamma=0.01;, score=(train=1.000, test=0.963) total time=   0.9s
[CV 1/5] END SVM__C=100.0, SVM__gamma=0.1;, score=(train=1.000, test=0.978) total time=   3.9s
[CV 2/5] END SVM__C=100.0, SVM__gamma=0.1;, score=(train=1.000, test=0.964) total time=   3.0s
[CV 3/5] END SVM__C=100.0, SVM__gamma=0.1;, score=(train=1.000, test=0.969) total time=   3.0s
[CV 4/5] END SVM__C=100.0, SVM__gamma=0.1;, score=(train=1.000, test=0.958) total time=   3.0s
[CV 5/5] END SVM__C=100.0, SVM__gamma=0.1;, score=(train=1.000, test=0.971) total time=   3.1s
[CV 1/5] END SVM__C=100.0, SVM__gamma=1.0;, score=(train=1.000, test=0.280) total time=   5.8s
```



Before moving on to some other approaches, I decided to sweep more values of C and gamma but in a constricted region for RBF kernel. After training 64 models, I got the following results (C= np.logspace(-1, 1.6, num=8), gamma= np.logspace(-1.5, -0.8, num=8))

Grid scaled training score for libsvm is: 100.0 %
Grid scaled test score for libsvm is: 96.5 %
The Best parameters according to grid search are: {'SVM__C': 16.92666615037876, 'SVM__gamma': 0.05011872336272722}
Training score for LIBSVM with best parameters: 100.0 %
Test score for LIBSVM with best parameters: 97.0 %

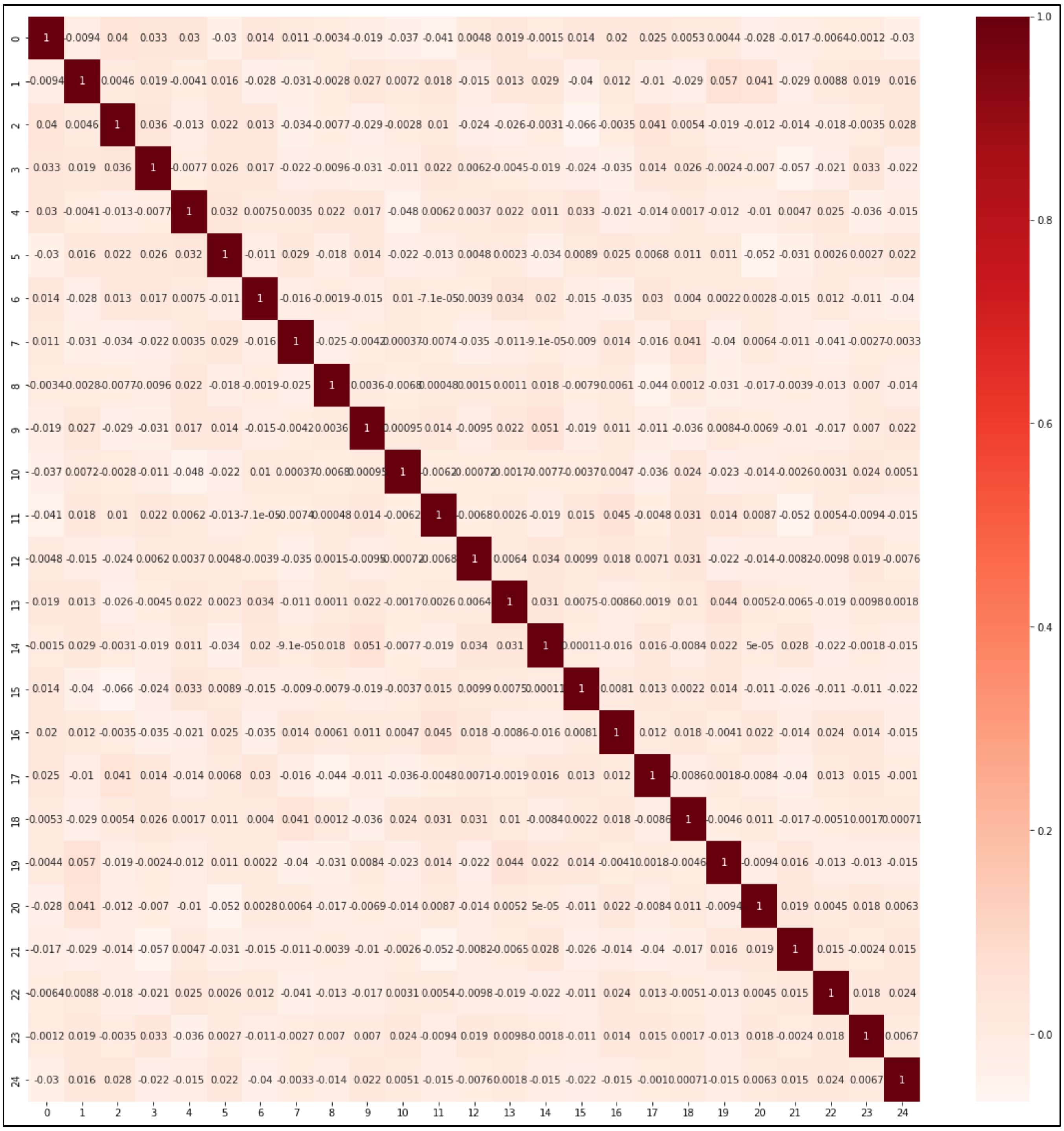


So, I used the predictions without any scaling and with these parameters. This gave me an accuracy of 96.875% on the leaderboard

After that, I restricted the C and gamma even more, and sweeped over the last parameter coef0 to get the same accuracy but a much better dev set accuracy (97.916%). The parameters used were: {'SVM__C': 25.118864315095795, 'SVM__coef0': -0.4, 'SVM__gamma': 0.05623413251903491}

My next step would be to do feature selection using some tools provided by the sklearn library. I can plot the correlation values among different features and the P value, and eliminate the highly dependent ones or the unimportant ones.

So first, I create the correlation matrix for all the features of our data. The result are as follows:



As we can see, there is no highly dependent attributes in our data and thus we cannot eliminate any feature. I confirm this using the recursive feature elimination specifically on my best model using the sklearn API.
Note: It wasn't working on my system, but worked on Google colab. The results were:

Rfe.support_ returned: [True True True True True True True True True True True True True True True True True True True True True True True True True]
Implying that all the features are important for training and fitting the model

Rfe.ranking_ returned: [1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]
Implying that all the features fit in the 'most important' category

Thus, a bit of feature engineering did us no good.

Now, I will retrain all the good performing models that I have used till now and use a voting classifier. i.e., I will pick the label which is returned by majority of the classifiers (This is 'hard' voting classifier. Alternatively, I can have a soft voting classifier that takes probabilities into account). If no such label exists, I will provide some weight to different models based on their validation set accuracies and choose accordingly.

I trained 3 such ensemble classifiers. One of type "hard", one of type "soft" and one of type "soft" with qualitative weights based on performance on the dev set from earlier.

The 8 models taken were:

```
clf1 = svm.SVC(kernel='rbf', C=10, gamma=0.1, probability=True)
clf2 = svm.SVC(kernel='poly', C=0.1, gamma=0.1, degree=3, probability=True)
clf3 = svm.SVC(kernel='rbf', C=100, gamma=0.1, probability=True)
clf4 = svm.SVC(kernel='rbf', C=16.92666615037876, gamma=0.05011872336272722, pr
obability=True)
clf5 = svm.SVC(kernel='rbf', C=25.118864315095795, gamma=0.05623413251903491, c
oef0=-0.4, probability=True) #Best
clf6 = svm.SVC(kernel='rbf', C=2.31, gamma=0.059, probability=True)
clf7 = svm.SVC(kernel='rbf', C=7.19, gamma=0.067, probability=True)
clf8 = svm.SVC(kernel='rbf', C=6.31, gamma=0.052, probability=True)
```

The training accuracy for all 3 types was 100%
The test accuracy for hard voting was 97.125% and it was 97.25% for the other 2

All of them gave the same 96.875% accuracy on the leaderboard.

I decided to go with the prediction that gave the best dev set score of 97.916% for my final submission

References (For Code)

Note: It is possible that codes in some of these references are not present in my current version of the code. But they certainly influenced my code

- https://stats.stackexchange.com/questions/31066/what-is-the-influence-of-c-in-svms-with-linear-kernel
- https://stackoverflow.com/questions/37161563/how-to-graph-grid-scores-from-gridsearchcv
- https://courses.csail.mit.edu/6.867/wiki/images/a/a7/Qp-cvxopt.pdf
- https://www.robots.ox.ac.uk/~az/lectures/ml/lect3.pdf
- https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html
- https://xavierbourretsicotte.github.io/SVM_implementation.html
- https://www.discoverbits.in/701/find-the-indices-of-the-values-common-two-arrays-lists-python
- https://jakevdp.github.io/PythonDataScienceHandbook/04.12-three-dimensional-plotting.html
- https://matplotlib.org/2.0.2/mpl_toolkits/mplot3d/tutorial.html
- https://linuxtut.com/en/faba9e8b9c764a045e8e/ (Based on Ch7 PRML)
- https://towardsdatascience.com/feature-selection-with-pandas-e3690ad8504b
- https://www.geeksforgeeks.org/ml-voting-classifier-using-sklearn/
- https://machinelearningmastery.com/voting-ensembles-with-python/
- https://cs229.stanford.edu/materials/smo.pdf
- https://web.iitd.ac.in/~sumeet/tr-98-14.pdf