SVM1

Irene Volpe
r0740784

1 Exercises

1.1 A Simple Example: Two Gaussians

Fig 1: red dots represent one class and blue dots the other.
Task is to find a linear separator, ie a hyperplane, that separates the two classes with the largest margin.
The margin is the distance of such hyperplane to a few close points called support vectors, and it gets wider the more distinct the classes are.

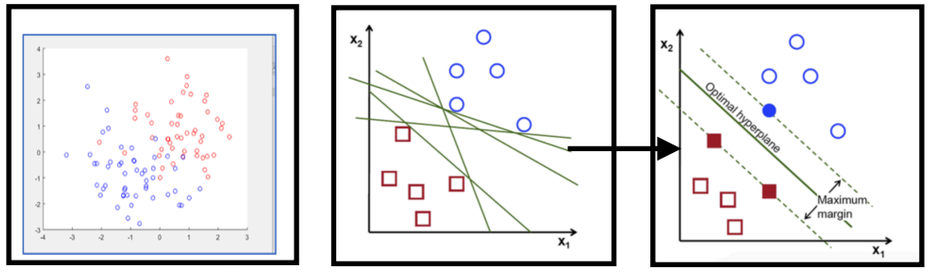


fig 1

Fig.2: the two classes have a Gaussian distributions with identical covariance and different geographic coordinates;
The linear decision boundary holds true for this classification problem even if the two distributions overlap, due to:
identical shapes of the two distributions,
approximately linear data
absence of many outliers

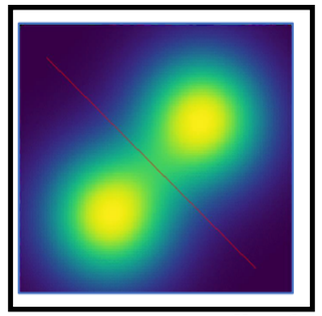


fig 2

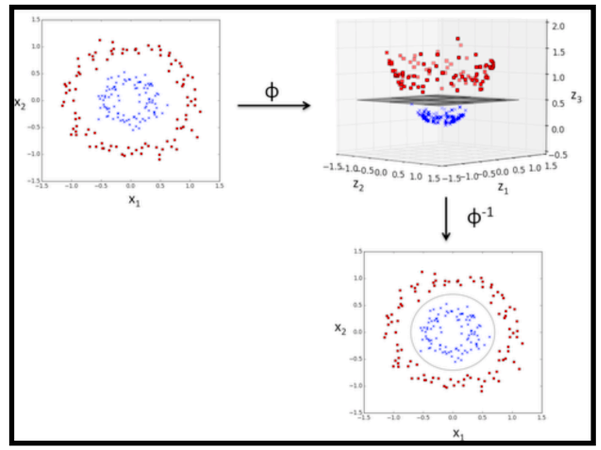


fig 3

1.2 SVM Classifier

1.2.1 Linear and RBF kernel for classification

Fig 3:kernel functions map the non-linear separable input space into a higher dimensional linear separable feature space.
The kernel method then maps the solutions back, so that in the non-linear separable input space you then have a non-linear solution.

Figure 4 to 13 show the effects of adding data on the decision boundary
Adding data points on the right side:
Fig 4: Initial decision boundary
Fig 5: The initial decision boundary does not change much
Fig 6: Adding data closer to support vectors causes a shift in angle and width of the margin.

Adding data points on the wrong side:
Fig 7: Since this is a small dataset, addition of even one datapoint changes the decision surface
Fig 8: Adding more than one datapoint on the wrong side gives a larger margin.
Fig 9-10: Adding more datapoints on wrong side completely changes the decision surface width and orientation.

Adding points near outliers (In the original dataset: one green outlier) :
Fig 11: Adding a point near this outlier widens the margin to minimize the error classification.
Fig 12: Adding a second datapoint near this outlier shifts the angle of the margin.
Fig 13: Adding a red datapoint results in outliers added on both of wrong sides which completely changes the decision boundary so that the SVM classifier can include more datapoints in the correct decision boundary.

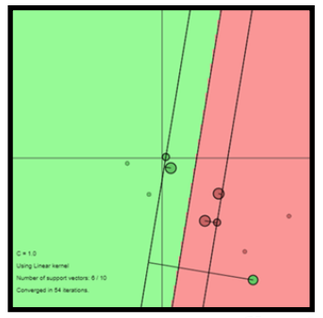


fig 4

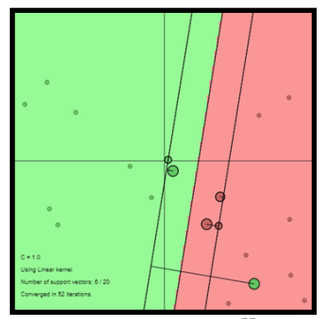


fig 5

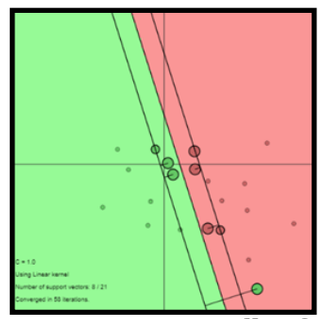


fig 6

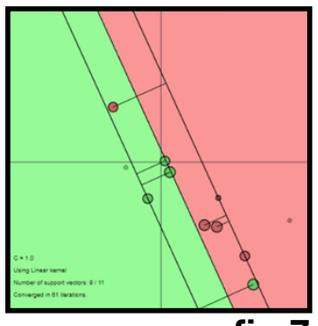


fig 7

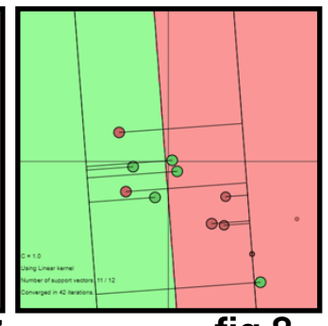


fig 8

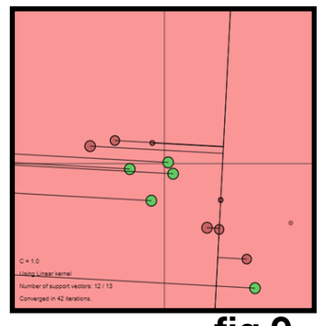


fig 9

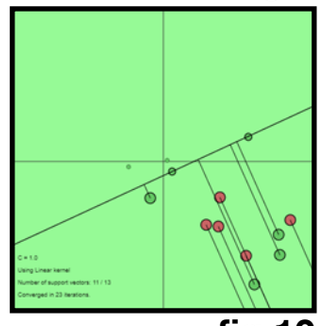


fig 10

fig 11

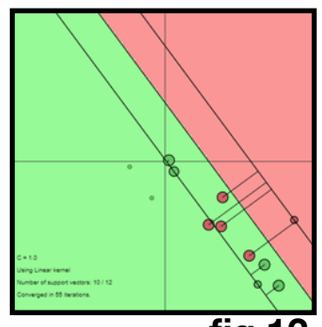


fig 12

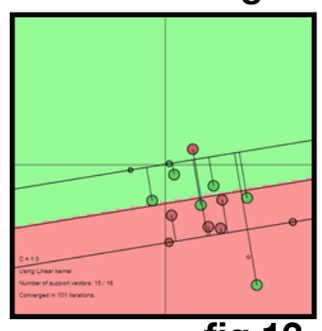


fig 13

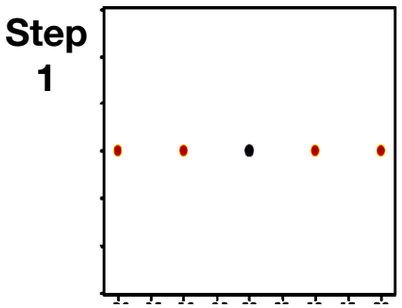
RBF kernel: a numerical example

RBF It is a general-purpose kernel used to handle classification of non-linearly separable classes when there is no prior knowledge about the data.
Equation is:

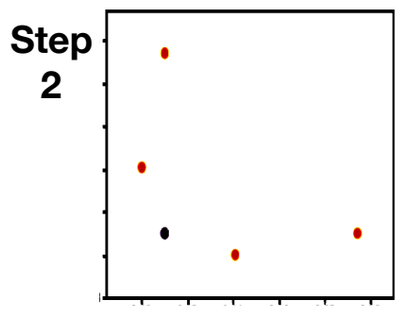
```
k(x,y) = exp(−gamma ∗ ||x−y||^2)
```

To illustrate the benefit of applying a Gaussian rbf (gamma = 0.1), let's use the same example:

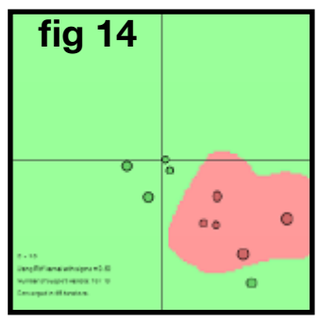
We repeat the same steps as above with RBF kernel to differentiate how RBF performs in comparison to linear kernel.
Fig. 14: Initial Decision Surface
Fig 15: Adding red datapoints
Fig 16: Adding green datapoints

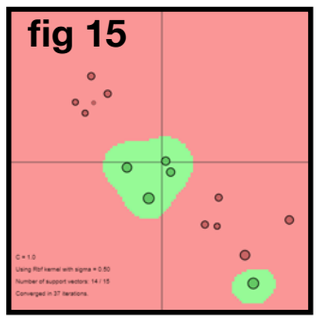


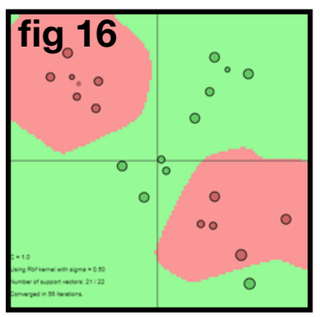
Existing Feature: X = np.array([-2,-1,0, 1,2])
Label: Y = np.array([1,1,0,1,1])
It's impossible for us to find a line to separate the dots,



If we apply Gaussian RBF transformation using two centers (-1,0) and (2,0) to get new features, we will then be able to draw a line to separate the yellow purple dots:
X_new1 = array([1.01, 1.00, 1.01, 1.04, 1.09])
X_new2 = array([1.09, 1.04, 1.01, 1.00, 1.01])



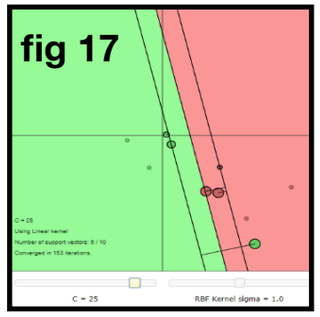


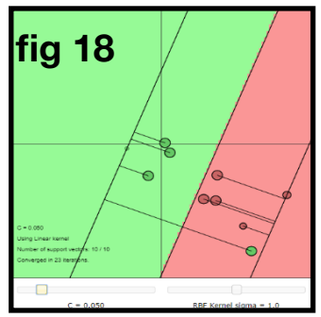


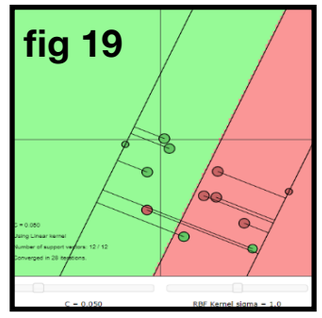
1.2.2 C and sigma (aka gamma)

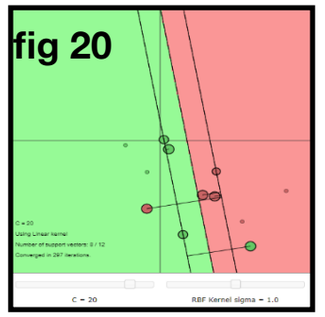
Regularization Hyperparameter 'C':
In case of overlapping class distributions, to relax our constraints when we cannot find a hyperplane that separates the two classes precisely, we use C as tunable parameter that controls how much error we are willing to afford in the training data.
Higher value of C —> higher misclassification "penalty" value —> narrower margins

Fig 17: Higher value of 'C' narrower margin
Fig 18: Lower value of 'C' larger margin
Fig 19: Smaller 'C' with more non-overlapping class data-points
Fig 20: Higher 'C' with more overlapping classes data-points







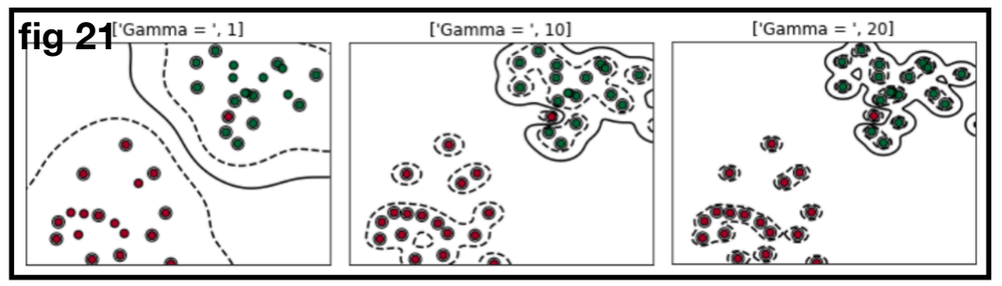


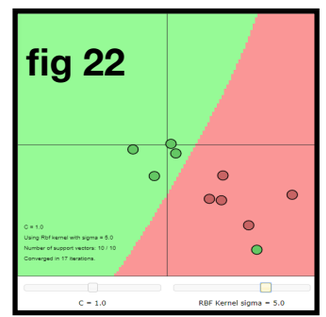
Kernel Parameter Sigma/Gamma:
Similar to C in the soft margin in Linear SVM, Gamma is a hyper-parameter that determines how much misclassification error we allow in the training data.
Fig 21:
Gamma is small, we allow a lot of misclassified data-points
Gamma gets bigger, influence increase, the decision boundary get wiggled.

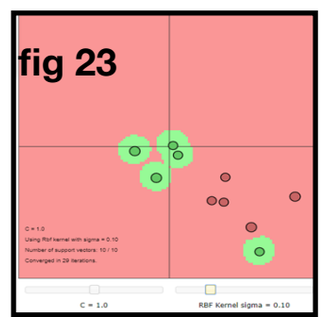
Sigma(Gamma regularization in MATLAB):

```
k_rbf(x,y) = exp(−1/(2*sigma^2) ∗ ||x−y||^2).
```

Fig 22: Higher value of sigma tends to make a more general classifier that considers nearly all data points and we get a smoother decision boundary.
Fig 23: Smaller sigma tends to make a local classifier with stricter decision boundary.







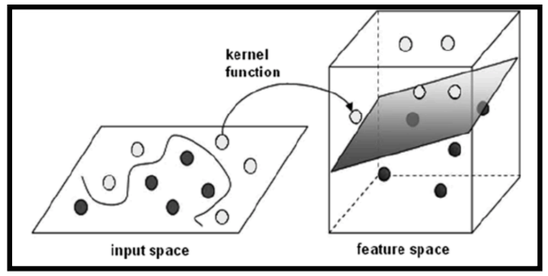
1.2.3 SVM Classification Linear kernel versus RBF kernel

SVM classification: Linear kernel vs RBF kernel

Fig 24: linear kernel performs best for linear problems
Fig 25: RBF kernel classifies the non-linearly separable data more efficiently by applying clustering technique.

1)Linear kernel is a better choice when data is classifiable by both, because RBF is a non-parametric model: it handles (in this case unnecessary) complexity at the expenses of an increased computational demand for training to tune more hyperparameters and is susceptible to overfitting when there is some noise in the data.
2) For the same reasons with unlimited data and very weak assumptions about the problem, RBF will be a better choice.

fig 24



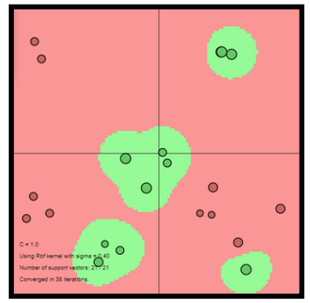
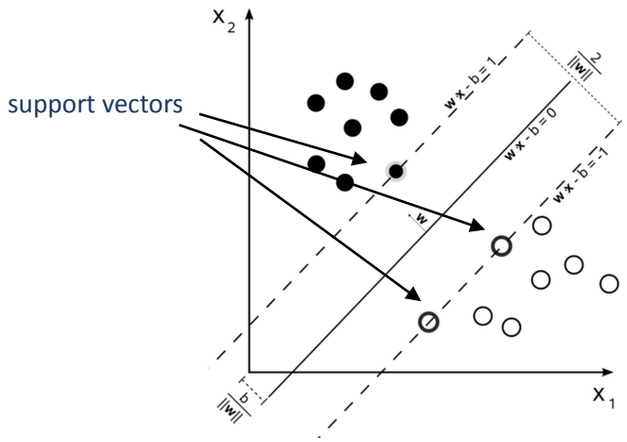


fig 25

To sum up, SVM in the linear non separable cases:

- By combining the soft margin C (tolerance of misclassifications) and RBF kernel trick together, Support Vector Machine is able to structure the decision boundary for linear non-separable cases.
- Hyper-parameters like C or Gamma control how wiggling the SVM decision boundary could be.
- the higher the C, the more penalty SVM was given when it misclassified, and therefore the less wiggling the decision boundary will be
- the higher the gamma, the more influence the feature data points will have on the decision boundary, thereby the more wiggling the boundary will be



1.2.4 Support Vectors

1.2.5 When the importance of Support Vectors change?

The aim of SVMs is to find the decision boundary that is furthest away from any data point.
Since the separating hyper-plane can be expressed in terms of the data points that are closest to the boundary, and these points are the support vectors, it follows that the importance of these lies in how close it is to the decision boundary.
The closer it is the more affect it has on the angle and width of the decision boundary.

Fig 26: initial state
Fig 27: adding points far from the decision boundary —> angle and width does not change
Fig 28: adding points near the decision boundary —> angle changes and width is narrower to avoid miscalssification

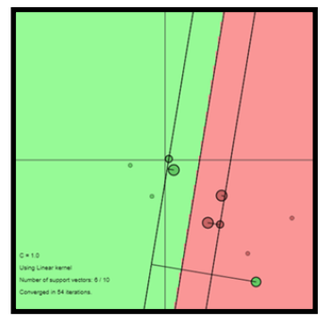


fig 26

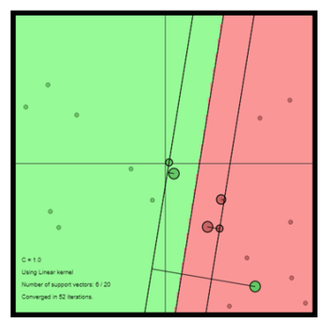


fig 27

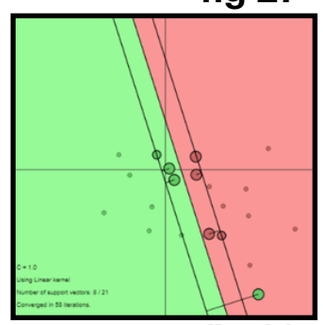


fig 28

1-3 LS-SVM Classifier

1.3.1 Influence of hyperparameters (regularization constant) and kernel parameters

In this section we evaluate the performance of the Least Square SVM with different polynomial kernels and regularization constant gamma as a hyperparameter.
The iris dataset is used which contains two classes with a total 100 datapoints for training and 20 datapoints for testing.

1.3.1.1 Polynomial kernel with varying degree - performance on the test set

The degree of the polynomial kernel controls the flexibility of the resulting classifier.
Figure.29: Performance of polynomial kernels of degree 1 to 20 are evaluated with gamma fixed at 1. A polynomial kernel with degree 1 performs like a linear kernel;
Higher-degree polynomial kernels allow a more flexible decision boundary, and the model is more likely to overfit. We get a misclassification error 20% on validation miss-class 4.

1.3.1.3 different sig2 values and fixed Gamma as kernel parameters

Figure 30: performance on test set of RBF kernel, sig2 values from 0.01 to 20 —>
sig2 = 0.01 —> 90% accuracy,
sig^2 0.1 —> 100% accuracy.
sig2 > 12 —> accuracy decreases.

Table.1: increasing value of sigma improves classification up to 12.50; after this threshold we get more misclassifications.

Sigma value	Error Rate	Misclassifications
0.01	10%	2
0.10	0 %	0
12.50	5%	1
13.00	10 %	2
14.50	15 %	3
15.50	30%	6
16.50	40%	8
17.50	50%	10
20.00	50%	10

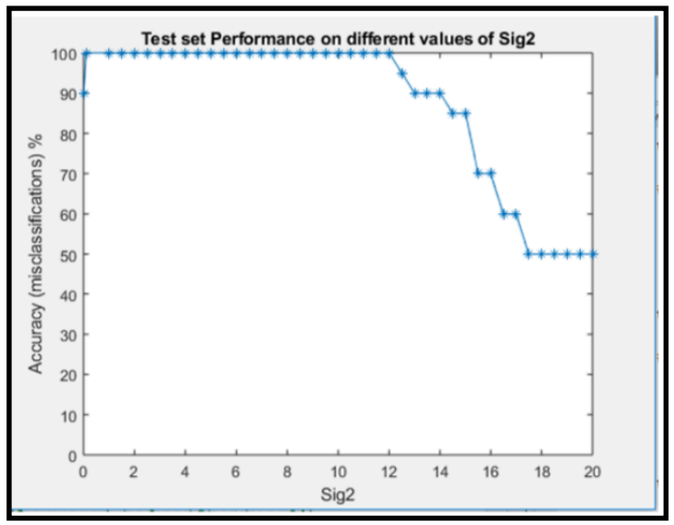


Table 1

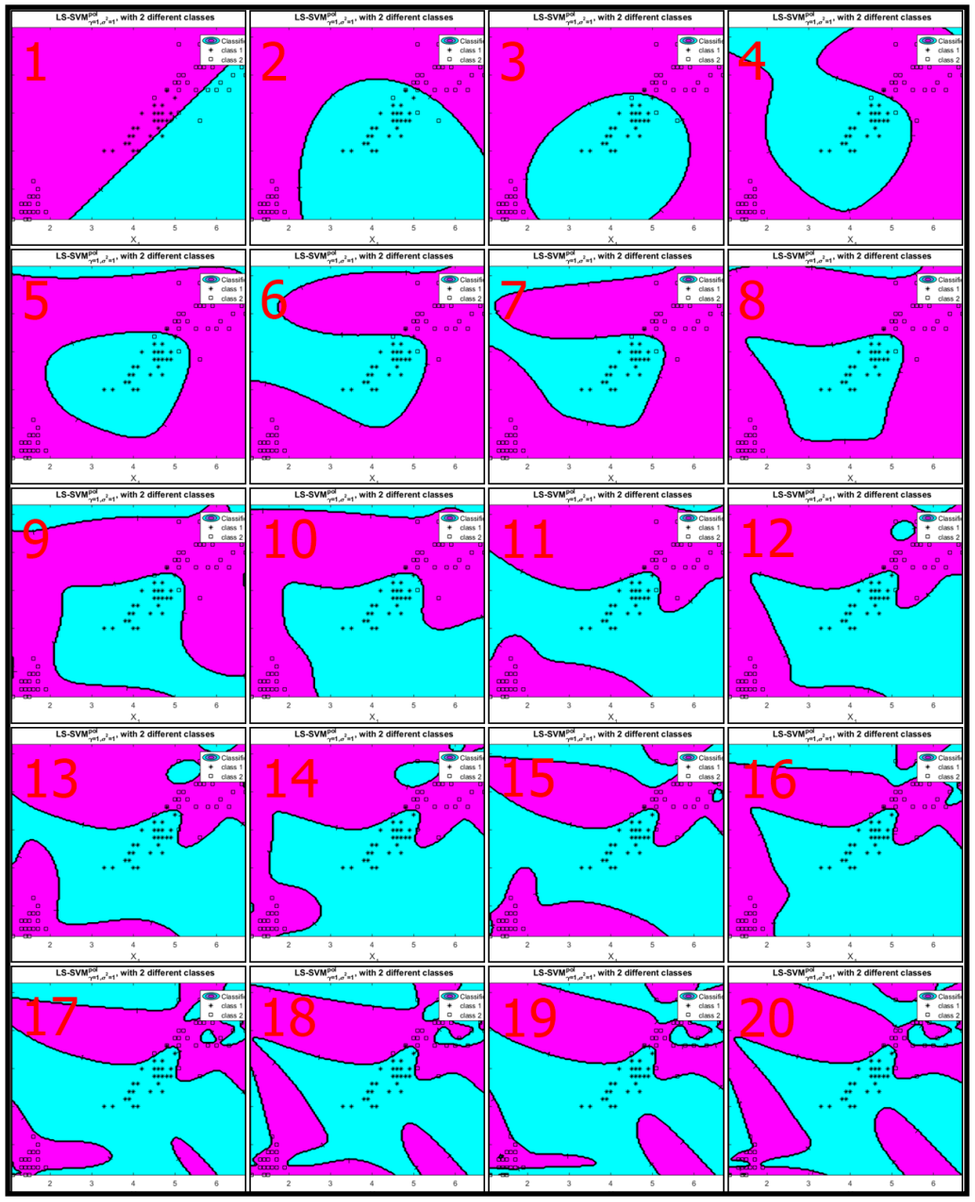


fig 29

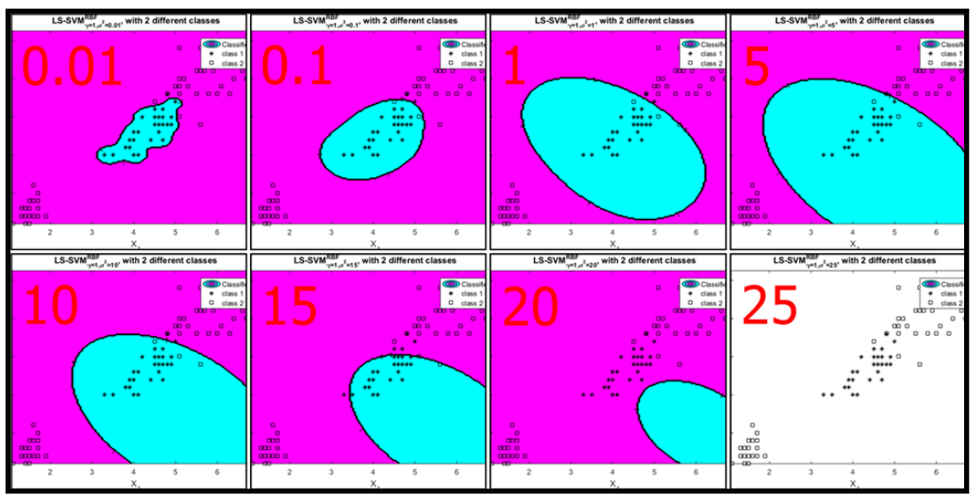


fig 30

1.3.1.4 different Gamma and fixed Sigma as kernel parameters

Fixing the sigma2 value at 1, we experiment on the gam parameter of the RBF kernel.

Fig 31: larger gamma = less generalized classification (it defines a smaller Gaussian variance).
Fig 32:
Only gamma 0.01 of gives a 90% accuracy on the test set;
for all higher values we get 100% classification accuracy —-> depending on sig2 value, a good value for gam should be above 0.01.

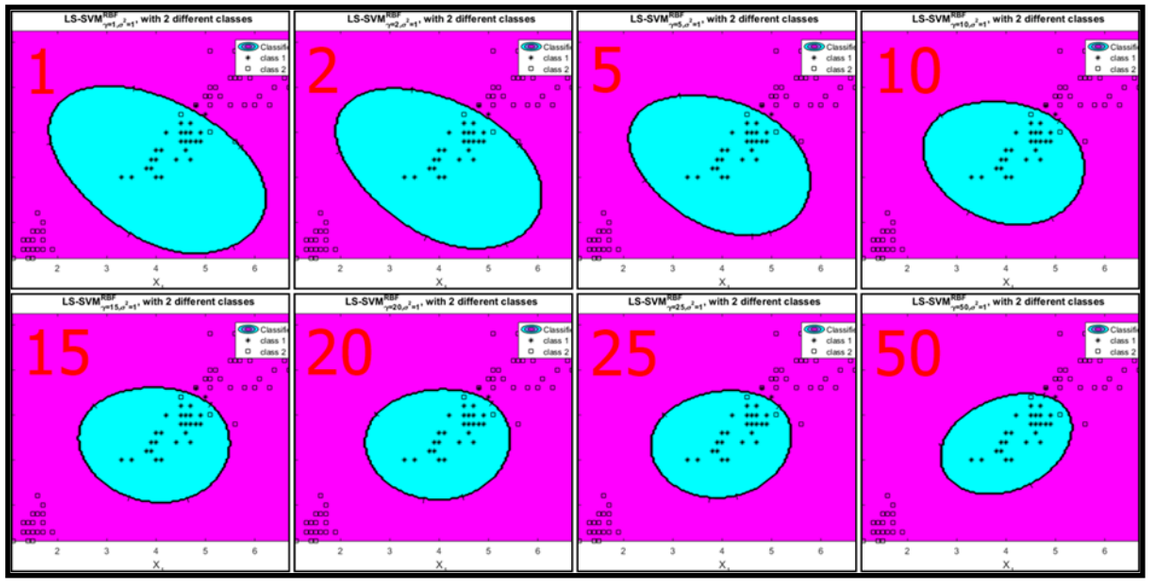


fig 31

1.3.1.5 comparing results with Sample Script of Iris Data

Comparing the results obtained with given sample script of iris database we notice that there aren't any significant differences:

1) Linear kernel gets 11 misclassifications (55% error rate).
2) Polynomial kernel (degree 5; gam, t = 1) achieve 0% error rate.

3) RBF kernel (gamma=1), Sig^2 [0.01, 0.1, 1, 5, 10, 25] —->
Fig 33: misclassification plot with different values of gam
Fig 34: misclassification plot with different values of Sig^2.

A certain range of sigma's values lead to higher accuracy and very low or high values of this parameter leads to more misclassifications.
All these results are consistent with the experiments we run earlier on iris dataset.

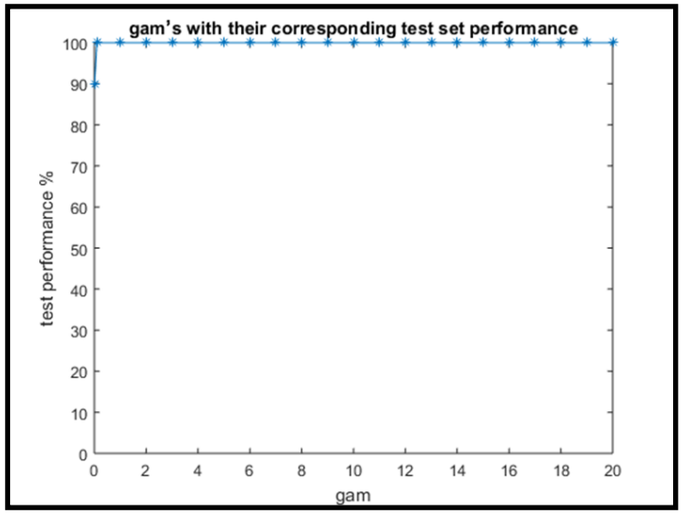


fig 32

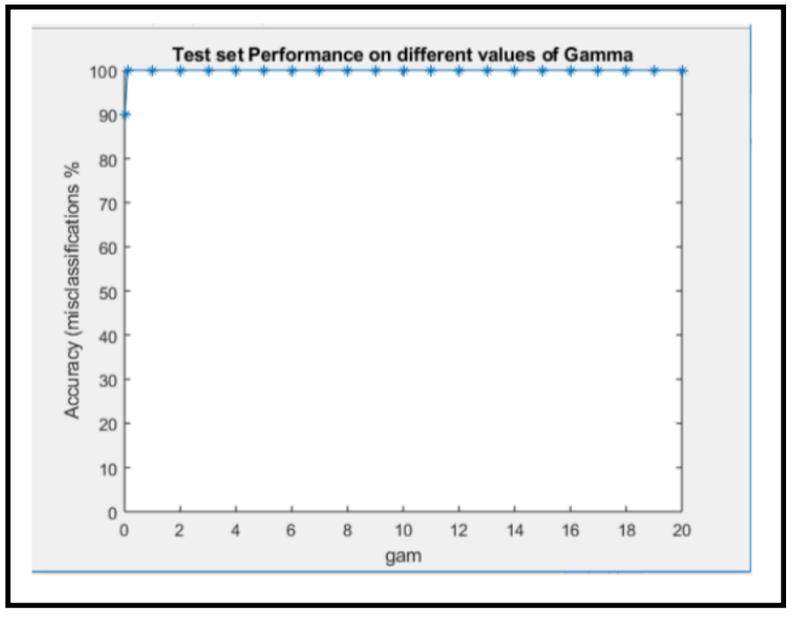


fig 33

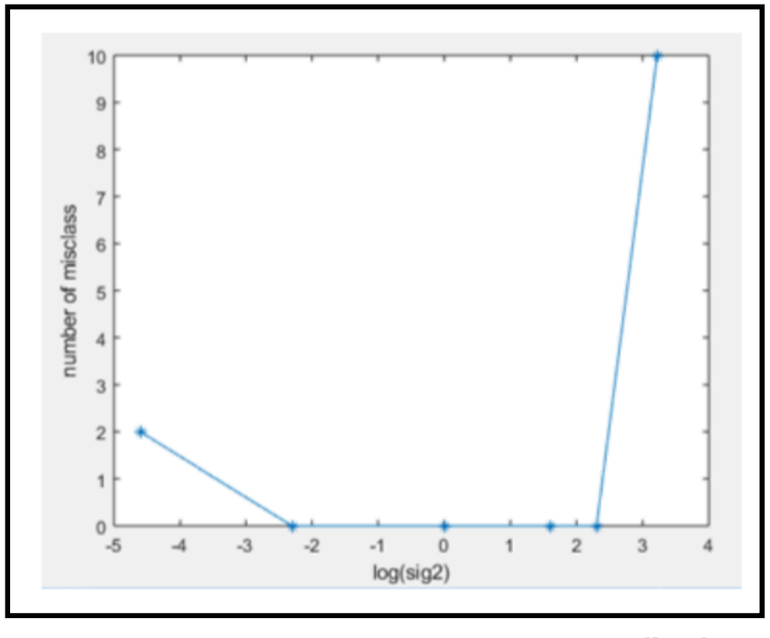


fig 34

1.3.2 Tuning parameters using validation

1.3.2.1 Performance of different validation methods using range of gamma and sig2 values

Random/simple split validation: We test on the same iris dataset, using 80% of data for training set and 20% for the test set.
We use range of gamma and Sig^2 values and choose RBF kernel.

Figure 35 shows the performance of model in terms of misclassification error on the random split set. The plot shows that for low values of gam and high values of sig^2 the error is low but increases significantly for larger values of gamma and low values of sig^2. We conclude that a good combination of tuned hyperparameters would be gam 1 to 50 and $sig^2 \geq 1$.

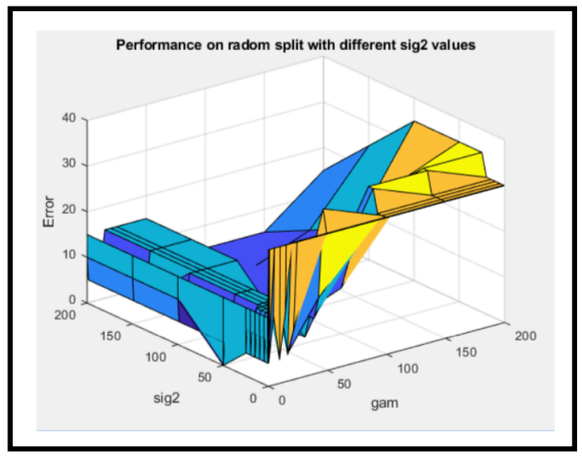


fig 35

Fig 36: 10-fold cross-validation: We use 10-fold cross-validation to check the performance on different combination of gamma and sigma values and update the hyperparameters accordingly. The final performance score is called the cross-validation error which is the mean of all the validation errors of the k-folds

Fig. 37: Leave-one-out validation Method: This is another form of testing the performance of the model. It works by using just one datapoint as the validation set and all the rest are used as training set.

Comparing the plots in figure 36 and 37, we can see the leave one out method is gives slightly better results than simple and 10-cross validation.
This can be because it essentially gets to train on more data than the k-fold validation method.
In practice, leave one out is less computationally expensive, but k fold validation gives much robust performance scores.

1.3.2.2 Cross validation versus random split (simple validation)

Looking at figure 35 and 37 its clear that 10 runs of cross-validating on 10-fold validation data compared to just one run on a randomly split data, gives a lower performance error.
This is because cross-validation divides the dataset into 'k' (10 in our example) parts and trains the model using k-1 subsets as training set and the remaining one subset as the validation set.
This process is repeated k number of times so that the model is tested on all k folds used as validation set.
Cross validation is better than random split method as it ensures that the model does not overfit on just one validation set and is optimized on whole dataset.
When choosing a value for cross validation there is a trade-off between bias and variance. As 'k' gets larger, the difference in size of training set and the resampling subsets and the bias gets smaller to avoid overfitting on just one validation set.
Usually k = 5 or k = 10 are used as it is proven experimentally, to give lowest test error to avoid high bias and variance in model.

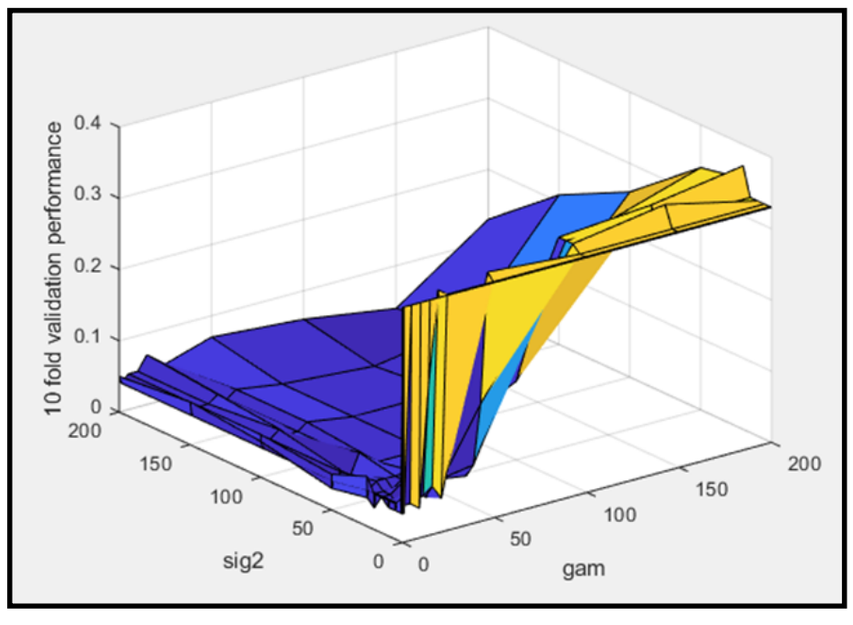


fig 36

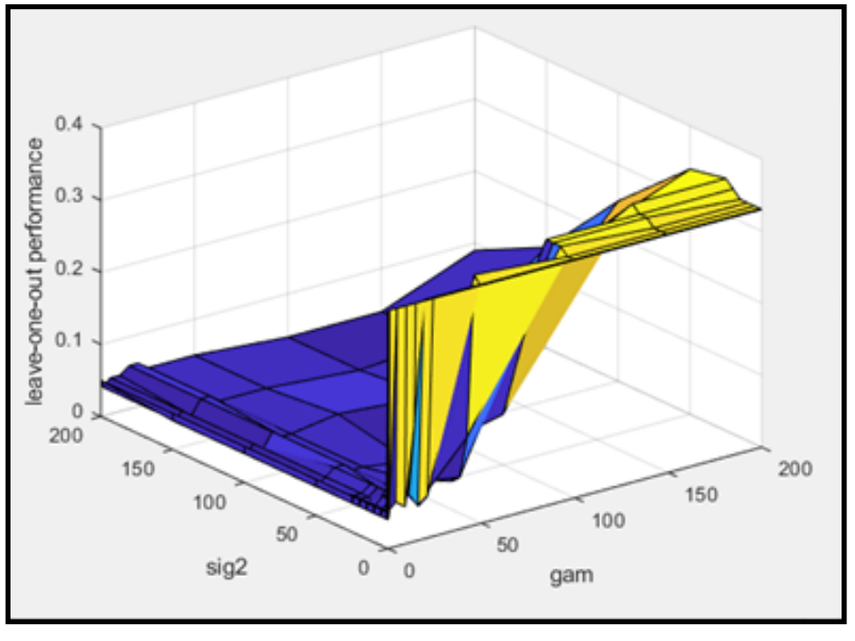


fig 37

1.3.3 Automatic parameters tuning

1.3.3.1 Hyper-parameters using 'simplex' and 'grid search' algorithms and the tunelssvm function

Table 2:
gam-sig^2 pairs generated by the tunelssvm function for Simplex and Grid-Search optimization algorithms;
A starter function is chosen to be Coupled Simulation Annealing (CSA) by default.
The values in blue show the highest pair values obtained while the ones in red are the lowest parameter values obtained in 10 runs for each combination of starter and optimization functions.

Hyper-parameter gamma and sigma values can differ widely even if other parameters are kept unchanged, and this is in part due to the starter function CSA, which first finds a suitable starting pair and passes it to the simplex and grid-search methods to come up with better performing values.
The better or worse the initial estimation by the starter functions is, the smaller or higher the final outcome we'll be, accordingly.

Simplex algorithm does not rely on computing gradients. However, it is a type of greedy algorithm, meaning that it takes the optimal solution for that particular step without regard to a global perspective. So, convergence can be slow around local maxima due to small steps around the extremum.

Grid search method tests every single possible combination and selects the best combination. This solution is then compared to the data using cost function. This method is extremely time intensive and computationally expensive.

In conlusion, Grid search is impractical for problems with large number of hyper-parameters, and Simplex works better on functions that are not locally smooth such as experimental data points.

The cost is the crossvalidatelssvm is 0.0400 both with simplex and grid search methods.

Optimization Algorith	Staring hyperparameters [gam sig^2]	Obtained Hyperparameters [gam sig^2]
Simplex	[26.38 0.04]	[87.60 0.04]
	[7.99 0.02]	[26.54 0.02]
	[6.84 x 10^4 0.55]	[6.84 x 10^4 0.55]
	[1.17 x 10^6 7.95]	[1.17 x 10^6 7.95]
	[1.88 0.27]	[1.88 0.27]
	[322.23 0.05]	[322.23 0.05]
	[34.18 0.01]	[34.18 0.01]
	[1.20 x 10^4 0.65]	[4.0 x 10^4 0.65]
	[15.88 0.68]	[52.74 0.68]
Grid Search	[0.004 0.053]	[0.33 0.26]
	[0.15 0.005]	[0.42 0.19]
	[10.94 0.011]	[899.70 0.098]
	[0.003 0.040]	[0.087 1.53]
	[0.030 0.001]	[1.05 0.017]
	[947.32 0.027]	[5.76 x 10^4 0.47]
	[0.22 0.004]	[1.86 0.0323]
	[0.15 0.012]	[0.58 0.056]
	[168.61 0.044]	[1.25 x 10^3 0.252]
	[0.18 0.034]	[0.85 0.034]

Table 2

1.3.4 Using ROC curves

1.3.4.1 Why computer ROC curve on the test set rather than on the training set

ROC curve helps in the interpretation of probabilistic forecast for binary classification problems. It assesses the fit between the model and the data. If we overfit training data, we will get an overly optimistic estimate of the performance of the model when we assess it with the training data itself (resubstitution).

1.3.4.2 ROC curve for the iris.mat dataset (use tuned gam and sig2 values)

Figure 38 and 39. show ROC curves for linear and RBF kernels.
higher sensitivity of 1.0 (true positive rate) in the RBF kernel;
lower sensitivity of 0.6 in a linear kernel.
Specificity (true negative rate) is the same in both.

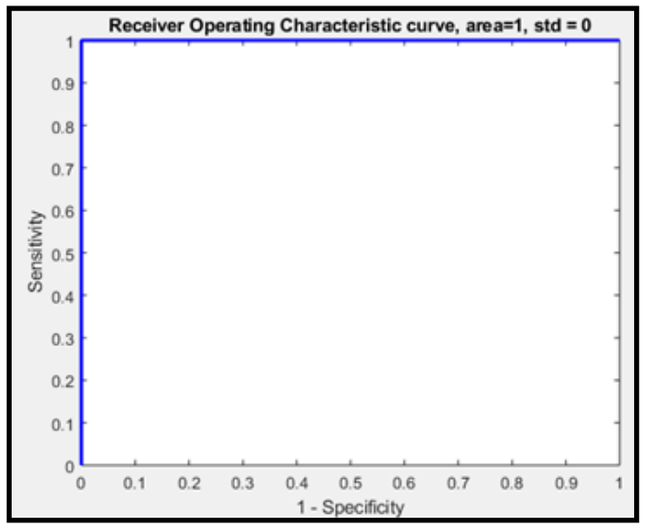


fig 38

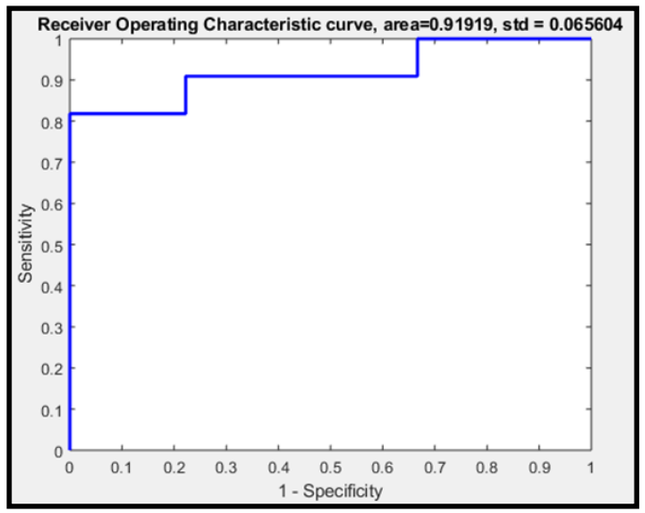


fig 39

1.3.5 Bayesian framework

Using Bayesian framework, we can find class probabilities for classification. This can be seen in figure 40.

1.3.5.1 Interpretation of colors

The different colors represent the probabilities of the spread of the two classes of the dataset. The blue region represents negative class while the surrounding violet region represents the positive class. The more spread the distribution of the dataset is, the more hue of colors we have in the plots. Also, if there are more datapoints with equal probability of classified a positive negative class, we see a more diffused region between violet and blue. Scale of probability 0-1 is given in left vertical bars, more defined and darker purple region defines a high probability of the positive class in the area.

1.3.5.1 Influence of gamma and sig2

The plots in the figure 40 show how gam and sig2 parameters influence the spread and depth of the colors representing the two classes. Using a low sig2 value causes the colored regions to be more defined and their overlapping boundaries shrink, showing high probability of the class spread in the respective regions. Whereas using a low value of gam also shrinks the boundaries but at the same time it enlarges the region of the negative class. High values of gam and sig2 parameters make the boundaries to overlap more, thus giving a low probability of the class boundaries.

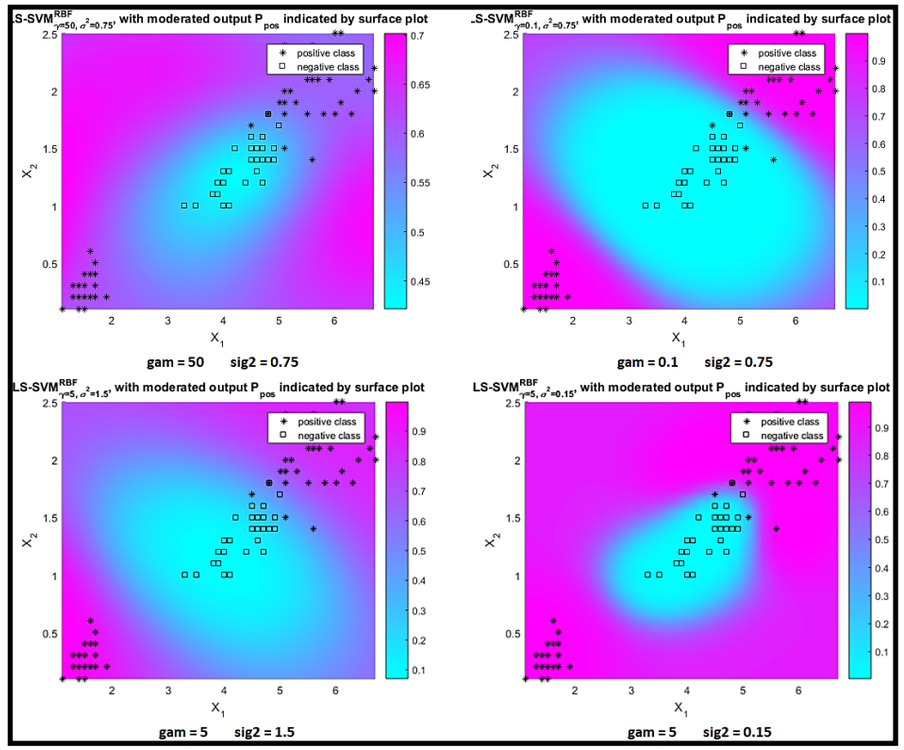


fig 40

2 Homework problems

2.1 the Ripley dataset

2.1.1 Properties of ripley dataset

The Ripley dataset plot is shown in Figure 41. It consists of 2-dimensional data set with 1000 datapoints divided into two classes. The test set contains 250 datapoints. Due to class overlap of many datapoints it cannot be separable with linear kernel. In the experiments in the following sections we will try to come up with the best way to classify this dataset by using SVMs.

2.1.2 Using a linear model

The plot of classification of the Ripley dataset with a linear SVM is shown in Figure 42. This shows a high classification error which remains which remains unchanged at 14.0% even when we try with very wide range of gamma parameter values are used.

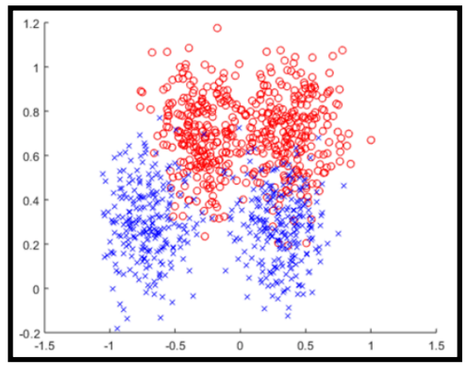


fig 41

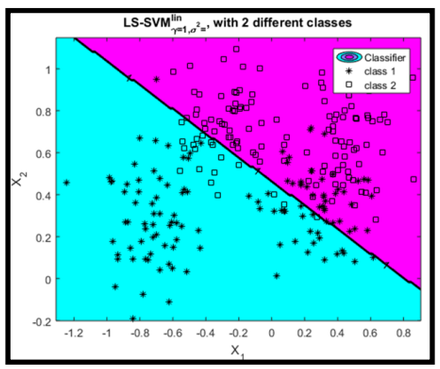


fig 42

2.1.2 Using a linear model

The dataset is classified by using the RBF kernel of the SVM, and a 10-fold cross-validation methos is used to fine tune the model.

FIg 43: Results of the 10-fold cross-validation with varying values of gam and sig^2.
FIg 44: The fine-tuned trained model. It uses the values of 0.12 and 0.27 for gam and sig^2 respectively achieving a lower error rate of 13.4% compared to the error rate of the linear kernel used earlier.
We see almost perfect separation of the two classes by the decision boundary.

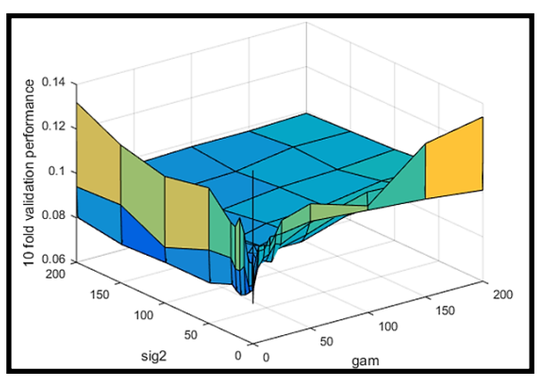


fig 43

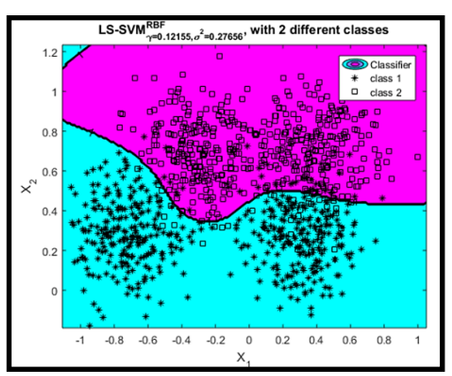


fig 44

Figure 45 shows the classification error using the leave-one-out validation method with various values of gam and sig². Figure 45 shows the achieved classification. This model has a 13.6% error which shows there is not much improvement on the previous set of parameter tuning.

Finally, the tunelssvm function is used multiple times to get an optimal set of parameters for the RBF kernel. With each run a new set of parameters is obtained that is significantly different from the previous one. On the other hand, the error rates of these sets of parameter values remain consistent. We choose the best of the obtained values and use them to train our model, results are shown in Figure 46. This model has an error rate of 13.2%.

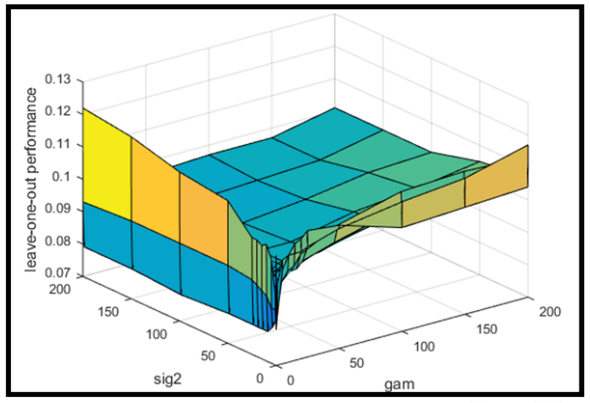


fig 45

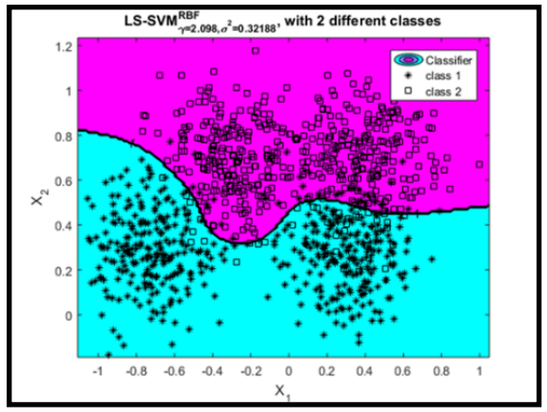


fig 46

2.1.3 ROC curves of models and their performance

Comparing the ROC curves of the linear kernel with the fine-tuned RBF kernel in figure 47 shows that RBF kernel outperforms the linear kernel by a slight margin only.

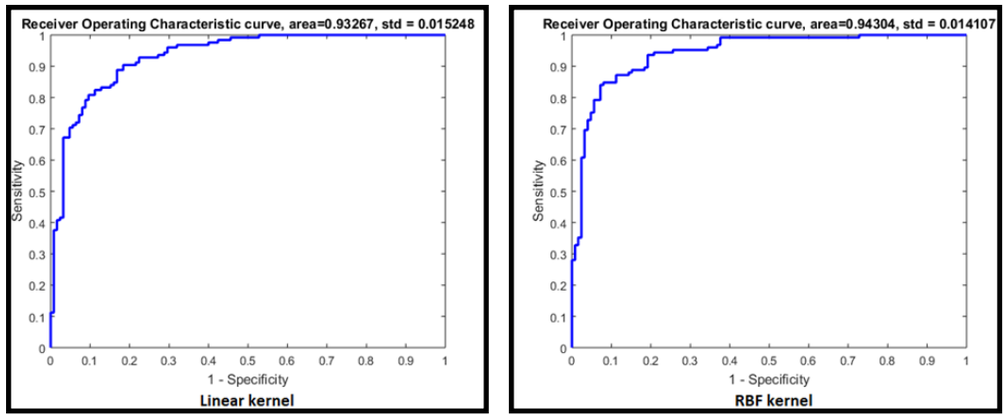


fig 47

2.1.4 final model's suitability for the dataset

Different models with varying hyperparameter values, kernel type and validation method were tested on Ripley dataset. Due to large dataset with many overlapping class distributions, parameter tuning was done by different methods, but despite this the maximum possible performance that can be achieved is easy to reach but relatively harder to perform better than this. However, the use of multiple validation schemes ensure that our model is trained to achieve the best possible results without suffering from any biases.

2.2. Breast cancer dataset

2.2.1 Properties of breast cancer dataset

This is a dataset collected from breast cancer patients to be classified into two groups; depending on if patients are diagnosed with malignant stage of cancer or not. The dataset consists of training set of 400 datapoints and test set of 169 datapoints with 30 dimensions. Since the dimensionality of the dataset is too large, the dataset cannot be visualized on a graph.

2.2.2 Using a linear model

When using a linear kernel, the only parameter that must be tuned is the y parameter. Testing for a wide range of values gives the same classification error (4.73%) for every run despite the high dimensionality of the dataset.

2.2.3 Using RBF kernel model

The dataset is classified by using the RBF kernel of the SVM. 10-fold cross-validation is used to fine tune the model.

FIg 48: Results of 10-fold cross-validation with varying values of gam and sig².
Due to high dimensionality of the dataset, the fine-tuned trained model cannot be visualized.
Gam and Sig² values are set at 25.
The model gives an error of 2.36% which is significantly less than the error of the linear kernel.
Using leave-one-out validation makes no significant change in results.

Tunelssvm function is run multiple times to get an optimal set of parameters for the RBF kernel.
With each run a new set of parameters obtained are significantly different from the previous one.
The error rates on these sets of parameter values also vary a lot from the previous ones, with maximum error reaching at 14.4%.
We choose the best obtained values (gam 18.06 and sig² 871.34) to train our model; giving an error of 3.65%.
The set of values obtained from 10-fold cross-validation give better results and hence we use these values for final optimization of RBF kernel.

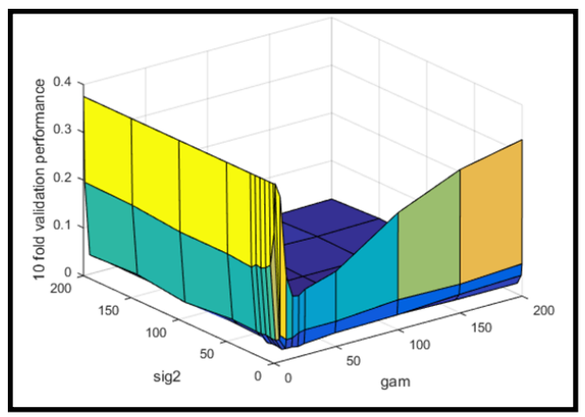


fig 48

2.2.4 ROC curves of models and their performance

Figure 49 shows the ROC curves of linear and the optimized RBF kernel. Although both these classifications perform nearly perfect, the optimized RBF outperforms the linear kernel.

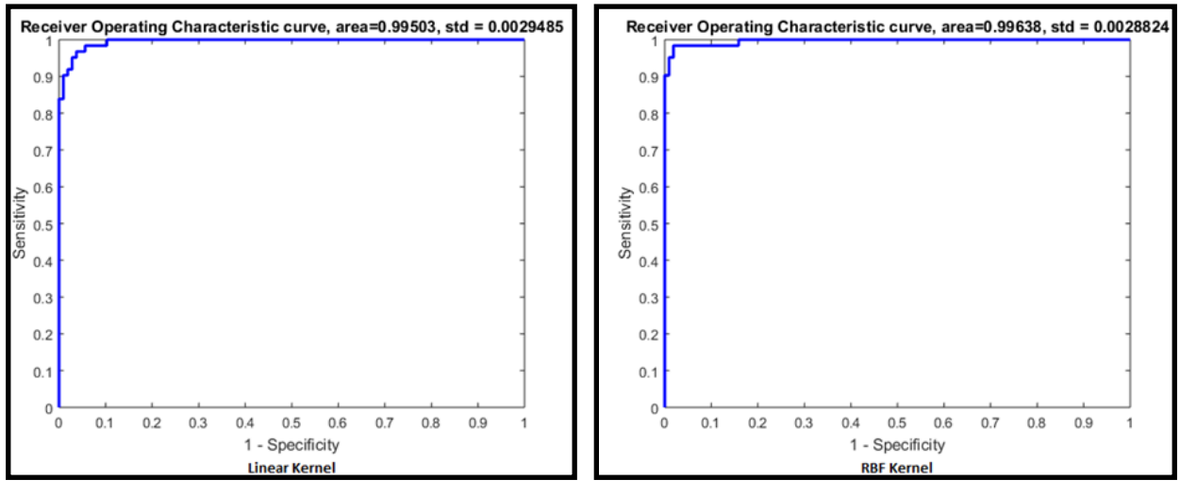


fig 49

2.2.5 final model's suitability for the dataset

The methodology used is the same for this dataset as that on Ripley's dataset. Due to high dimension of dataset, it could not be visualized for observing the distribution and classification.
Validation and optimization of the parameters helped achieve high accuracy results that shows the effectiveness of the methodology and model used.

2.3. The diabetes dataset

2.3.1 Properties of diabetes dataset

The diabetes dataset consists of 2 classes with a training set of 300 datapoints and 8 dimensions, and a test set of 168 datapoints.
The dataset cannot be visualized due to its high dimensionality.

2.3.2 Using a linear kernel

Fig 50: Testing for a wide range of values of 'y' we get the lowest classification error of 20.45% at y=0.1.
The high error can be due to high dimensionality of dataset.

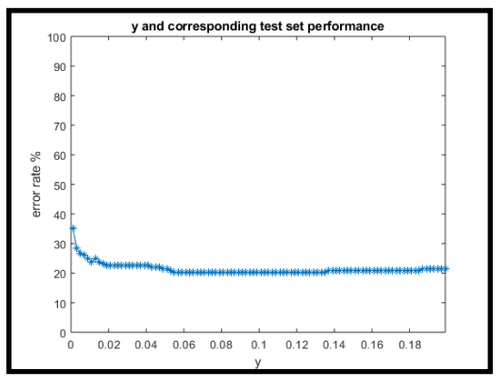


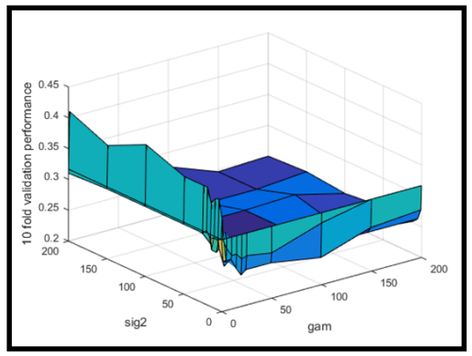
fig 50

2.3.2 Using a Radial basis function kernel

Fig 52: error rates corresponding to different combination of gam and sig2 values, tested to classify dataset.
10-fold cross-validation is used with different values of gam and sig2 to fine tune the model.

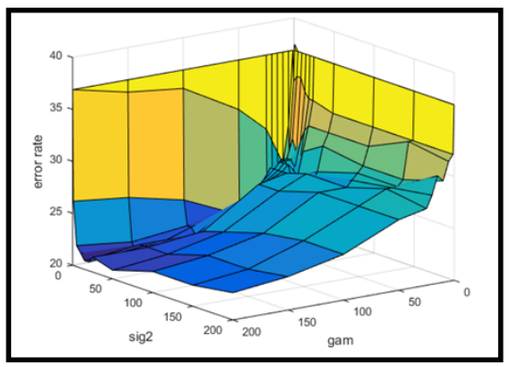
Fig 51: Results of the 10-fold crossvalidation with varying values of gam and sig2.
From two test plots we get the parameter values of 10 and 150 for gam and sig2 y.
The model's error of 20.40 % is equal to the error of the linear kernel.

Due to high dimensionality of dataset, the fine-tuned trained model cannot be visualized.



results of the RBG, 10-fold cross-validation tuned, corresponding to

fig 51



error rate of the RBG corresponding to different combination of gam and sig2

fig 52

2.3.3 ROC curves of models and their performance

Fig 53: the ROC curves of the linear kernel and the optimized RBF kernel. These curves show that the optimized RBF kernel is slightly better in performance than the linear kernel on this dataset.

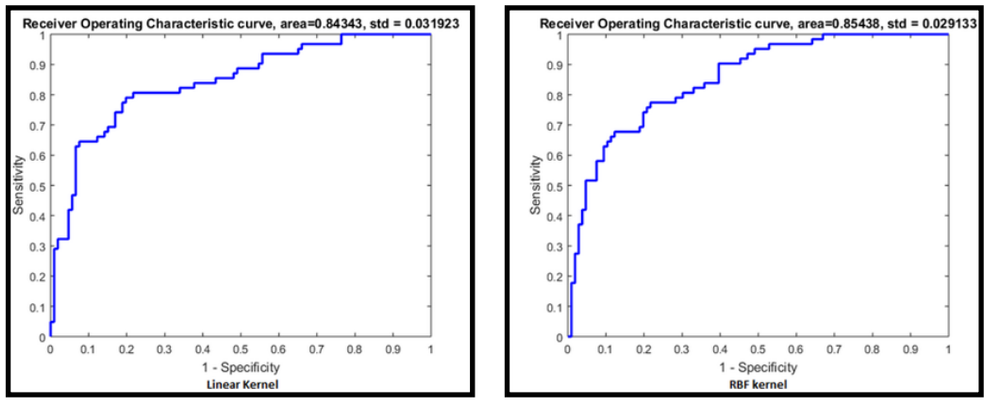


fig 53

2.3.4 final model's suitability for the dataset

The methodology used for classification of this dataset is also the same as the previous ones.
Just like the Breast Cancer dataset, this dataset has a high dimensionality and it cannot be visualized.
Unlike the previous dataset, this dataset had a high error rate on both linear and RBF kernels and different optimization parameters.
In general, the linear kernel seems to perform well for this dataset as the RBF kernel requires vigorous tuning to perform equally well.
Overall the results are exhaustive and very little, if any, improvements can be made to this dataset's classification.