KU LEUVEN

FACULTY OF SCIENCE

Exercise Session 1

Support Vector Machines

Daniel Gerardo GIL SANCHEZ
daniel.gilsanchez@student.kuleuven.be
MSc Statistics

Supervisor: Prof. Johan Suykens

Academic year 2018-2019

1 Exercises

1.1 A simple guide: Two Gaussians

- ***Given this figure, can you make a geometric construction using lines to estimate the optimal classifier?***

  In this case is relatively easy to make a classifier by hand that minimizes the misclassification error. Figure 1 shows the corresponding graphical representation:

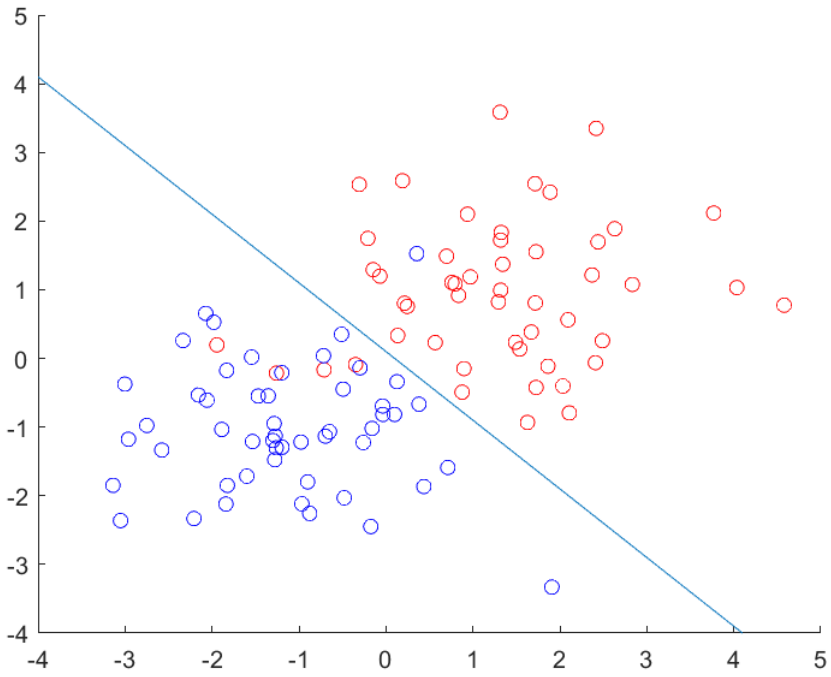


Figure 1: Graphical representation of optimal classifier

- ***Under which conditions do you think this construction is optimal/valid?***

  A couple of assumptions need to be taken into account in the development of such classifier:

  – The density of each class follows a multivariate normal distribution:

$$p(x) = \frac{1}{((2\pi)^n|\Sigma_{xx}|)^{1/2}} exp(-\frac{1}{2}(x-\mu_x)^T\Sigma_{xx}^{-1}(x-\mu_{xx}))$$

  – The variance-covariance matrix should be the same for each class. In the case of two classes:

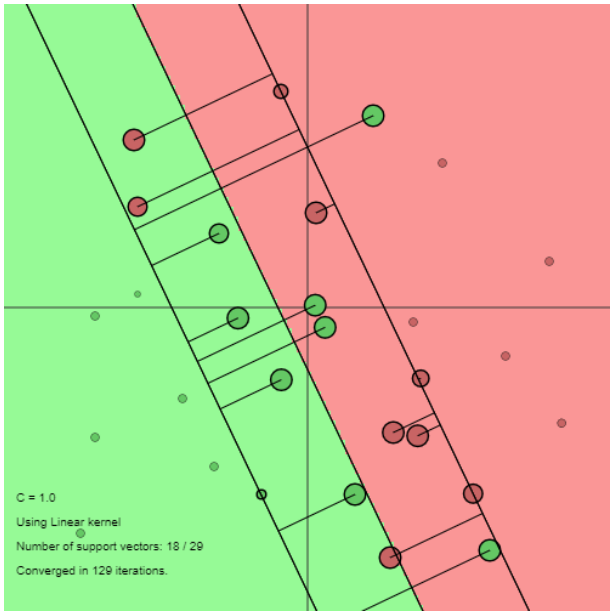
$$\Sigma_{xx1} = \Sigma_{xx2} = \Sigma_{xx}$$

1.2 Support vector machine classifier

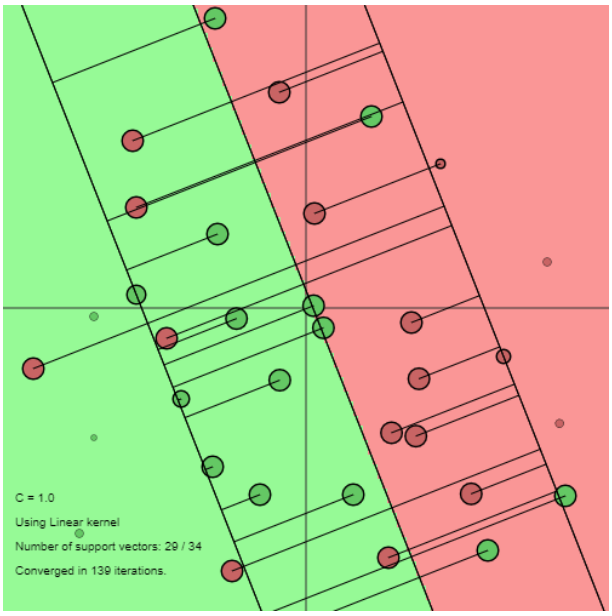
- ***What do you observe when you add more data points to the dataset - both on the right and on the wrong side of the hyperplane. How does it affect the classification hyperplane?***

  The first column of plots in Figure 2 show the initial set of data points and the corresponding classification region using a linear and an RBF kernel. The second column of the same Figure

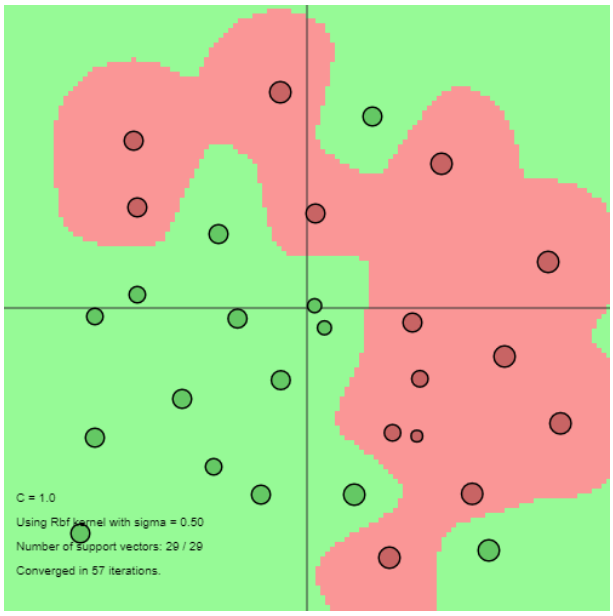
(plots 2b and 2d) shows how this region changes when new data points are added and also shows the influence of each observation in the construction of the new region.



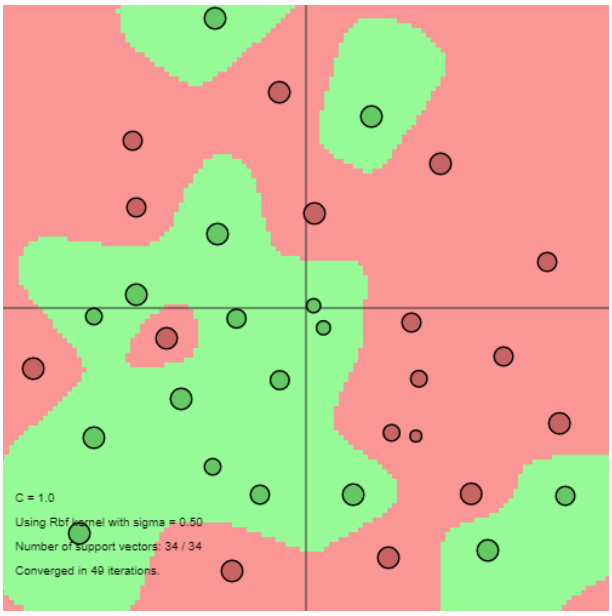
(a) Initial dataset. Linear Kernel



(b) New data points. Linear Kernel



(c) Initial dataset. RBF Kernel



(d) New data points. RBF Kernel

Figure 2: Comparison of kernels in classification of two classes

To see the difference it is important to locate first where these new points were added. There are three new red points in the third quadrant, one green point in the second quadrant (in the top) and another green point in the fourth quadrant (near the corner). In the linear kernel, the slope of the classification region seems to be the same but it has been moved a little bit to the right. The margin, which is represented as black lines parallel to the classification region, is now wider and the influence of each vector is different as well: There is a difference of 11 support vectors between the first solution and the second one.

In regard to the RBF kernel, the inclusion of these points made significant changes in the

classification region. Now the dominant class over the input space is red, not green as it was in the initial dataset. This is highly influenced by those new red points located in the third quadrant. In addition, it is important to note that in both solutions, all data points are important to build the classification region, meaning that the solution is not as sparse as in the linear kernel. However, the misclassification rate is kept at zero in both cases.

- ***Try out different values of the regularization hyperparameter C and the kernel parameter sigma. What is the role of the parameters? How do these parameters affect the classification outcome?***

  Figure 3 shows the differences in the solution using a linear kernel. By definition, the only parameter that affects the performance of the solution is the regularization constant. In this case, four different scenarios are considered: When the regularization constant is 1 (plot 3a), which is taken as a reference for comparison. When the regularization constant decreases the margin is larger and the slope is a little bit steeper (plot 3b). When this constant increases a little bit, the slope is shallower (plot 3c) and when this constant increases a lot, the slope is similar to the first scenario (plot 3d). Overall, it can be noticed that the solution is not importantly affected by the selection of the regularization constant.

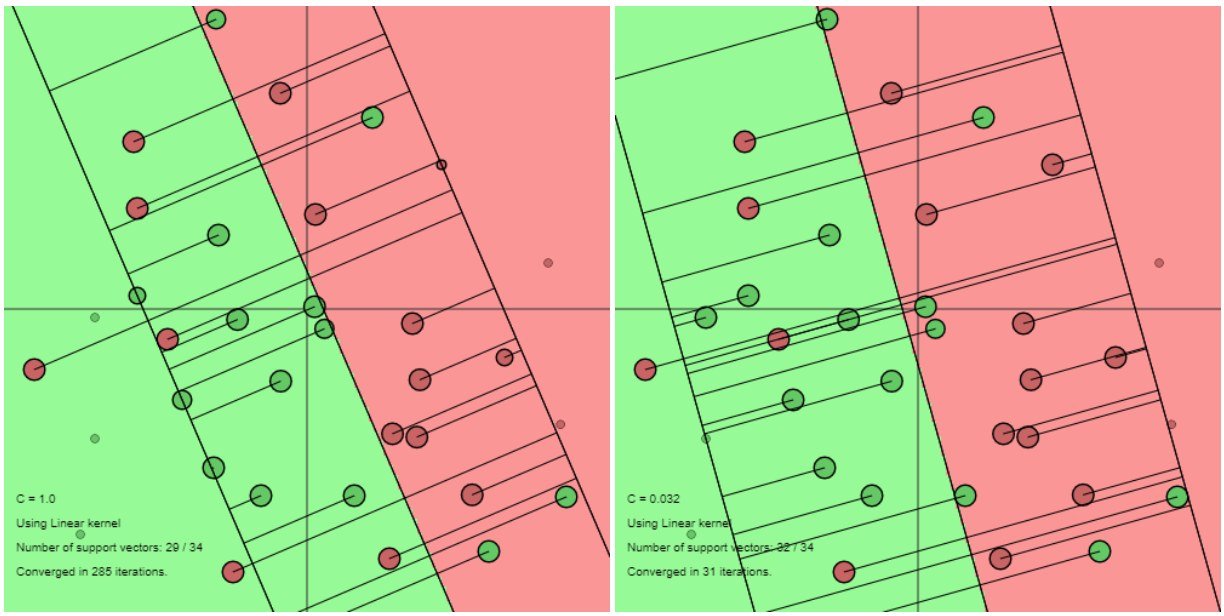
  Figure 4 shows the differences in the solution using an RBF kernel. By definition, two parameter affects the performance of the solution: the regularization constant and sigma. In this case, four different scenarios are considered: The first one is the same solution obtained in the previous item. In the second one, the regularization constant is smaller, impacting the overall solution by making the whole region green (plot 4b). In the third one, the regularization is kept at one and sigma is decreased a little, impacting the solution by making the whole region red (plot 4c). Finally, sigma is increased at 4 and the solution looks more like a polynomial kernel. Note that when sigma become larger and larger, the decision boundary tends to become linear. For this reason, the selection of the regularization constant and sigma is really important in RBF kernel because the solution is highly influenced by them.

  The role of these parameters is to make sure that the final solution is as optimal as possible. In particular, the regularization constant is a trade-off between a good solution and the complexity of the model. Usually, the more complex a model is, the less misclassification errors there are in the *training set*. This is, however, not usually the case for new datasets. for this reason is important to choose a regularization constant that improves generalization. In regard to sigma in RBF kernel, it represents the influence that a single point can have on the overall solution. The larger the sigma is, the closer other examples must to be affected. As a consequence, there are different ways to tune these parameters, such as cross-validation, Bayesian inference, among others.

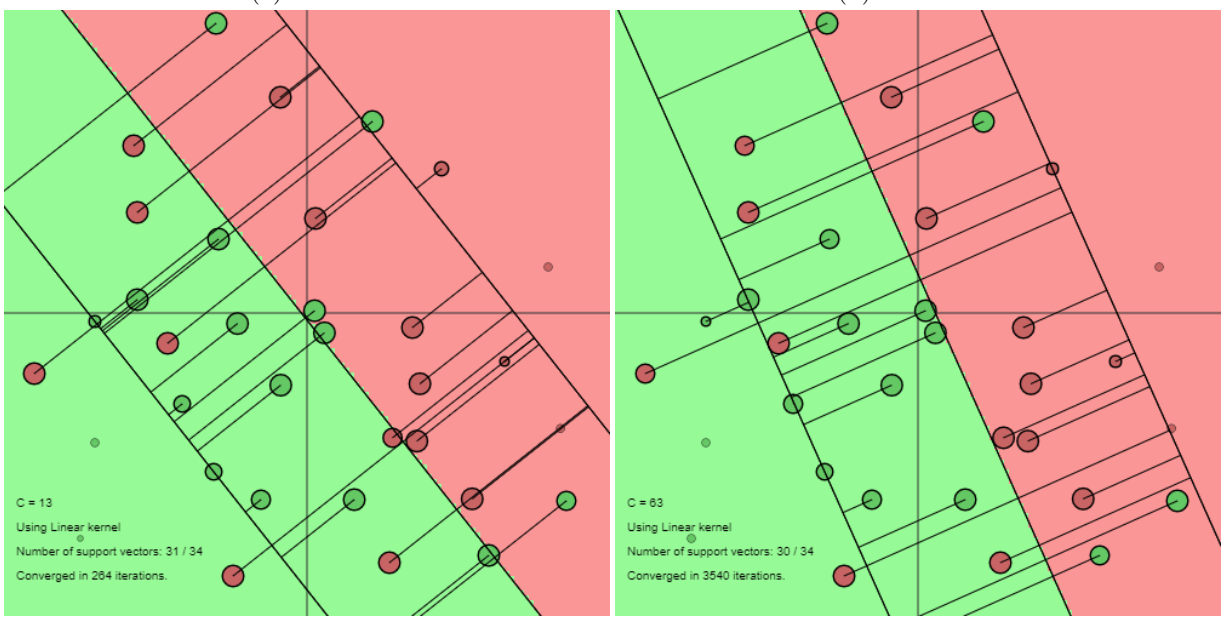
  In regard to the final question of which kernel performs better, it is easy to answer in this case because no test set is used to evaluate the performance, only in the training set. In this sense, the best solution is obtained by an RBF kernel using any combination of regularization constant and sigma because the misclassification error is zero. However, this does not mean that the model generalizes well the underlying behavior of each class, because it is possible to obtain an overfitted solution.

- ***What is a support vector? When does a particular datapoint become a support vector? When does the importance of the support vector change? Illustrate visually. Note that a support vector is indicated by a large circle with bold-lined circumference and that its importance is proportional to the size in the online application.***

  A support vector is any observation in the training set which corresponding Lagrange multiplier is different from zero. This Lagrange multiplier comes from the constraint optimization



(a) C = 1 (b) C = 0.032



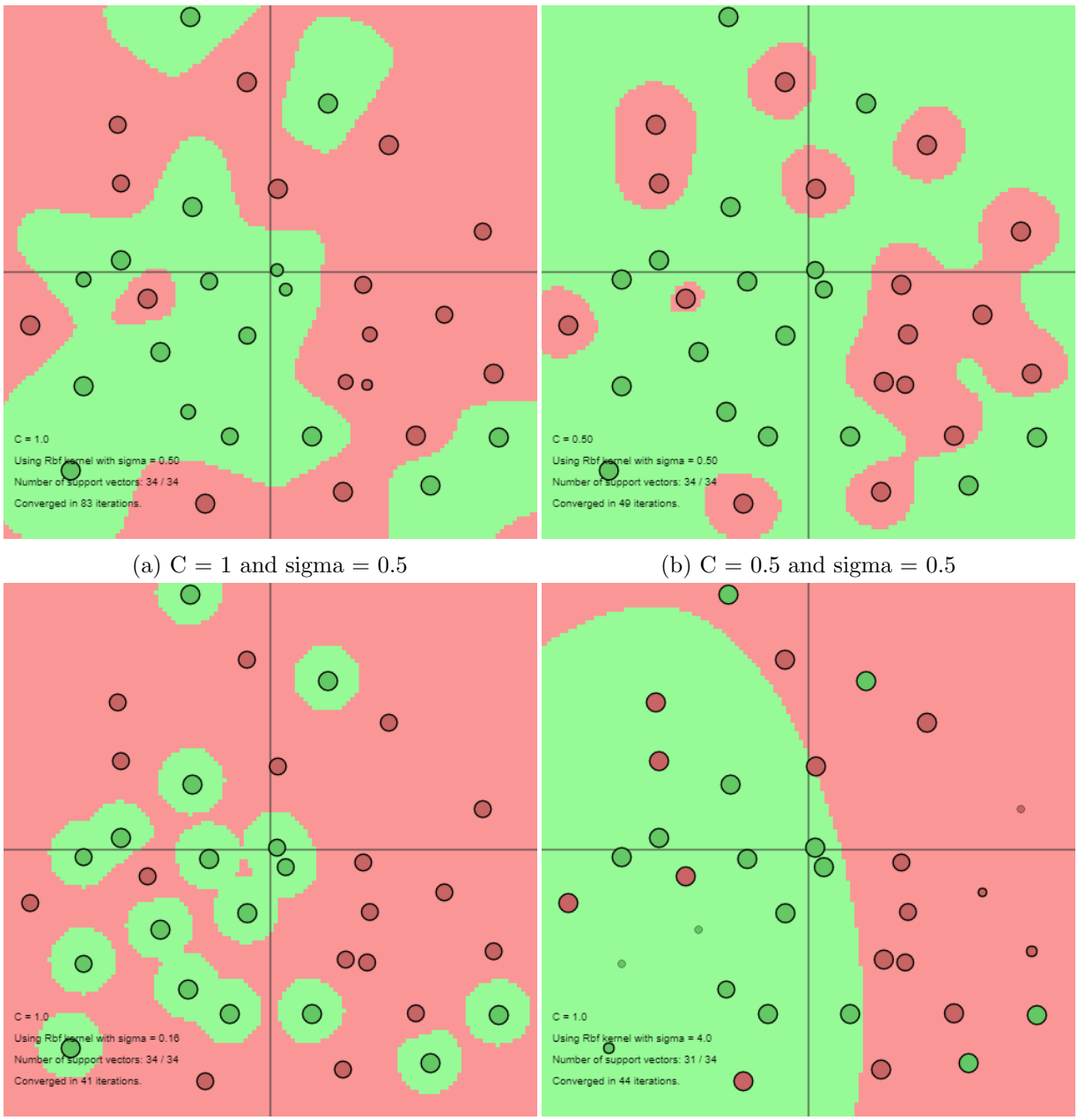
(c) C = 13 (d) C = 63

Figure 3: Comparison of linear kernel solution using different regularization constant (C)

problem encounter in the primal or dual representation of the model. The importance of a support vector may change when new observations come in the training set, or when different parameters or hyperparameters are considered, such as the regularization constant and sigma in an RBF kernel. This change can be easily seen in Figure 2, specifically in plots 2a and 2b. There, the importance of each support vector changed when new observations were added to the training set. This can be seen by the number of support vectors in each plot and the importance of some of them changed, represented as the diameter of each circle.

- ***What is the role of parameters C and sigma? What happens to the classification boundary if you change these parameters. Illustrate visually.***

  As it was mentioned before, C is the regularization constant included in the optimization pro-



(a) C = 1 and sigma = 0.5

(b) C = 0.5 and sigma = 0.5

(c) C = 1 and sigma = 0.16

(d) C = 1 and sigma = 4

Figure 4: Comparison of RBF kernel solution using different regularization constants (C) and sigma

cess of the model and it represents a trade-off between a good solution and the complexity of the model. Sigma, on the other hand, is an hyperparameter of the RBF kernel that represents the influence that a single point can have on the overall solution, also known as the bandwidth of the margin. The illustration of the influence of each parameter in both linear and RBF kernel is presented in Figures 3 and 4.

- ***What happens to the classification boundary when sigma is taken very large? Why?***

  As it was mentioned above, when sigma is very large, the decision boundary tends to become linear. Figure 4d shows an example of a large value for sigma. At this point the classification region changes a lot with respect to any other value considered. As sigma increases, the

whole region changes up to a point that all points are classified as red or green, leading to a misclassification rate of 50% (worst case scenario). This happens because sigma is used to make the transformation of the input space into a higher dimension feature space, which is then used in the dual representation of the model. This representation is as follows:

$$y(x) = sign\left[\sum_{k=1}^{N} \alpha_k y_k exp\left(\frac{-||x - x_k||_2^2}{\sigma^2}\right) + b\right]$$

Note that sigma is in the denominator, so when sigma is very large, the fraction will be small and the exponential will be close to one. For this reason, the importance of the numerator will always be diminished by the denominator, impacting the solution of the model.

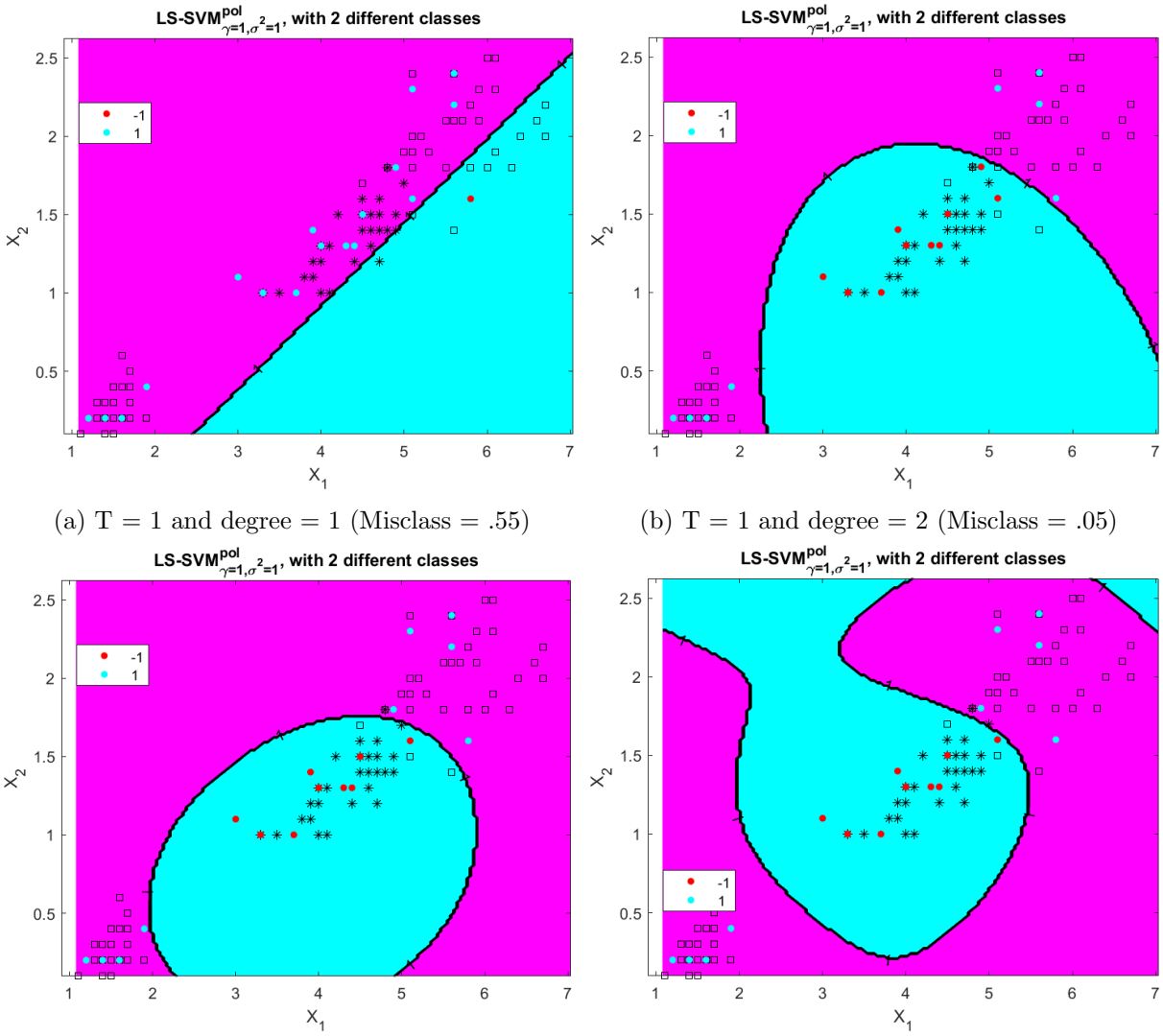
1.3 Least-squares support vector machine classifier

1.3.1 Influence of hyperparameters and kernel parameters

- ***Try out a polynomial kernel with*** $\texttt{degree} = 1, 2, 3, \ldots$ ***and*** $\mathrm{t} = 1$ ***(fix*** $\mathrm{gam} = 1$***). Assess the performance on the test set. What happens when you change the degree of the polynomial kernel?***

  Figure 5 shows the classification region of different models with polynomial kernel with degree 1, 2, 3 and 4 and their correspondent misclassification rate (Misclass). For all these models, *gamma* is fixed to 1 as well as *t*. In these figures, those data points located in the purple area are classified as 1 and those located in the blue area are classified as -1, the training points are plotted as squares or asterisks and the new points are circles in either blue or pink, as the legend states.

  In these plots is very clear how the degree of the polynomial kernel affects the classification region. The more complex the polynomial is, the better the misclassification rate obtained in the test set (in plots 5c and 5d this value is zero). It is of special interest the results from the polynomial kernel with degree 4, where the solution starts to look overfitted given the lack of smoothness when compared with the polynomial of degree 3. At this point, it would be wise to stop trying a higher degree polynomial because it is very likely that the misclassification rate on the test set deteriorates.



(a) T = 1 and degree = 1 (Misclass = .55)

(b) T = 1 and degree = 2 (Misclass = .05)

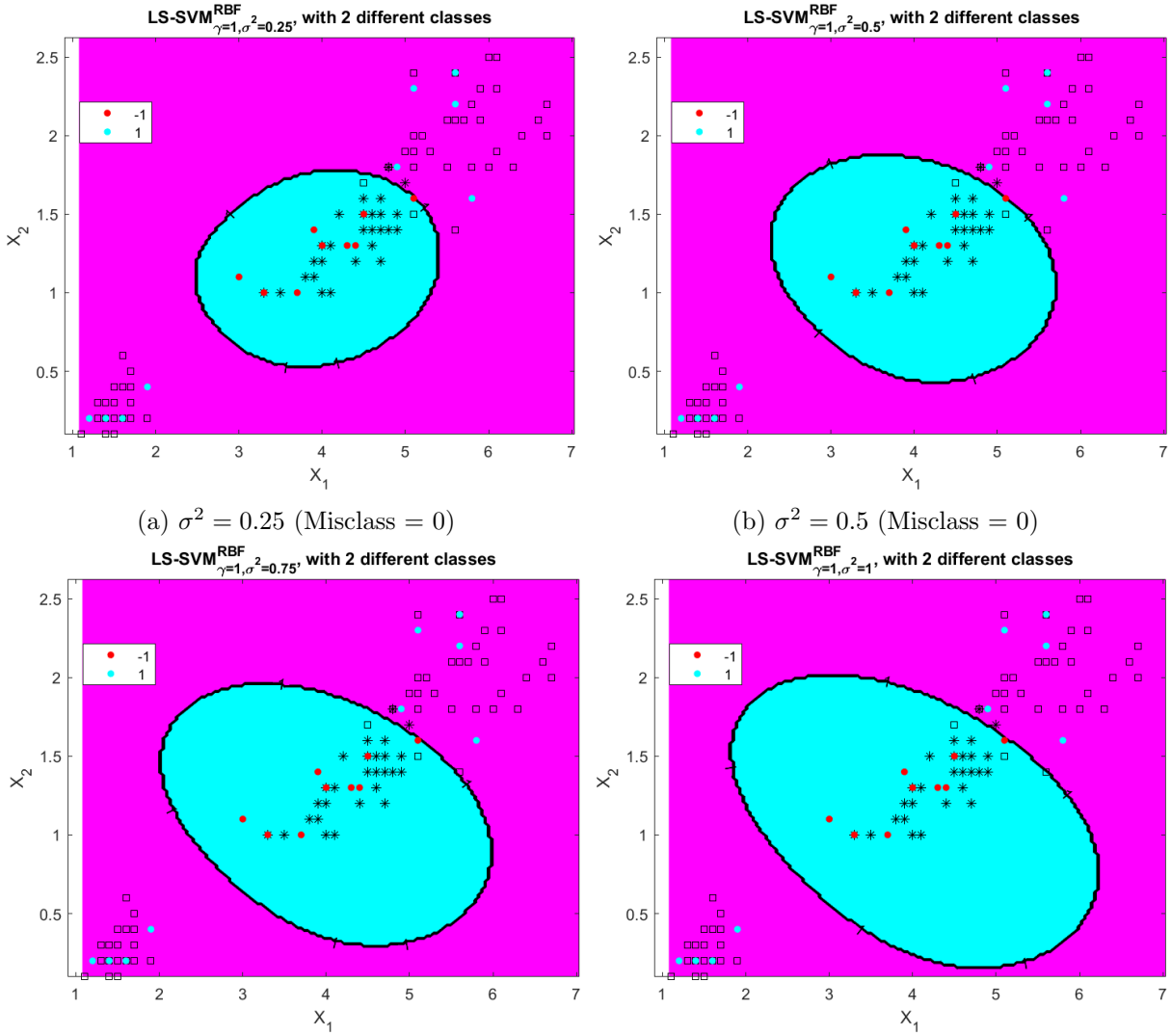
(c) T = 1 and degree = 3 (Misclass = 0)

(d) T = 1 and degree = 4 (Misclass = 0)

Figure 5: Polynomial kernel with different degree

- ***Let's now focus on the RBF kernel with squared kernel bandwidth σ^2. Try out a good range of different `sig2` values as kernel parameters (fix `gam` = 1). Assess the performance on the test set. What is a good range for `sig2`?***

  Figure 6 shows the classification region and the correspondent misclassification rate of an RBF kernel with different bandwidth σ^2. Four different scenarios were considered in this case, namely 0.25, 0.5, 0.75 and 1. In all cases the misclassification rate computed in the test set is zero, but the region of the class -1 is bigger as σ^2 increases. It would be wise to leave sigma in this range, between zero and one, because if this interval increases, it is very likely that the misclassification rate deteriorates.



(a) $\sigma^2 = 0.25$ (Misclass $= 0$)

(b) $\sigma^2 = 0.5$ (Misclass $= 0$)

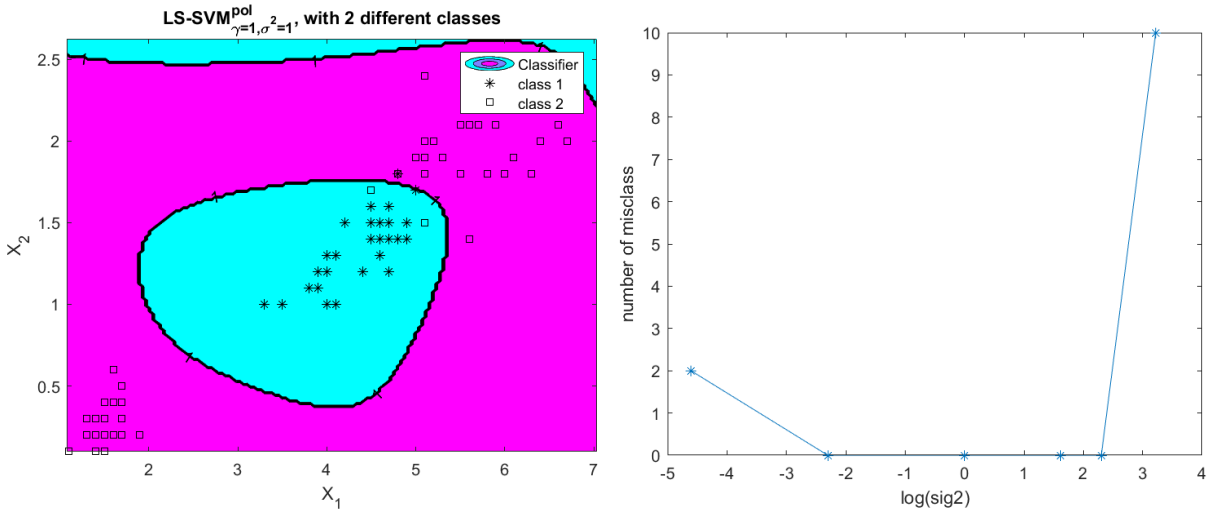
(c) $\sigma^2 = 0.75$ (Misclass $= 0$)

(d) $\sigma^2 = 1$ (Misclass $= 0$)

Figure 6: Polynomial kernel with different degree

- ***Compare your results with*** *SampleScript_iris.m****, which is available on Toledo.***

  In this script, a polynomial kernel of degree 5 was considered and different values of σ^2 were used for the RBF kernel, Figure 7 shows these results. As it was mentioned before, increasing the degree of the polynomial may produce an overfitted solution, which in this case can be assumed that if new observations came in the upper right corner of the plot (plot 7a), they would be incorrectly classified. Additionally, increasing the range of σ^2 in an RBF kernel, indeed deteriorates the misclassification rate. In this case, $\sigma^2 = 25$ produces a solution where 10 observations are missclassified (plot 7b).



(a) Polynomial kernel with degree = 5

(b) Number of misclassified observations

Figure 7: Results from `SampleScript_iris.m` script

1.3.2 Tuning parameters using validation

- ***Compute the performance for a range of*** *gam* ***and*** *sig2* ***values (e.g.,*** $\gamma, \sigma^2 = 10^{-3}, \ldots, 10^3$***). Use the random split method, 10-fold cross-validation and leave-one-out validation. Visualize the results of each method: do you observe differences? Interpret the results: which values of*** *gam* ***and*** *sig2* ***would you choose?***

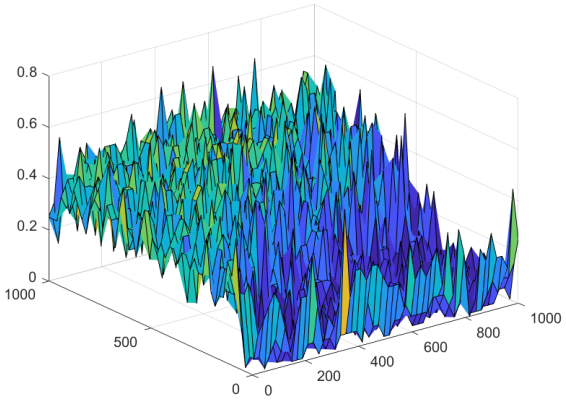
  To make a comparison between theses schemes of validation, a sequence of 50 values between 10^{-3} and 10^3 was created for each parameter (γ, σ^2). Then, for each possible pair of values the misclassification rate was computed and saved in a matrix. These results were then used to make a plot of the misclassification rate with respect to each pair of values for γ and σ^2. Finally, the pair that produces the lowest misclassification rate is chosen and compared.

  Figure 8 shows the results obtained for each validation scheme. There, it easy to see the smoothness of each surface with respect to the correspondent scheme: In random split, when the training set was divided in training and validation sets just one, the surface is more rough when compared to any other scheme. The smoothness of the surface of cross-validation scheme is improved with respect to random split but it is also more rough than leave-out-one validation. And the last scheme produces the smoothest surface with respect to the misclassification error.

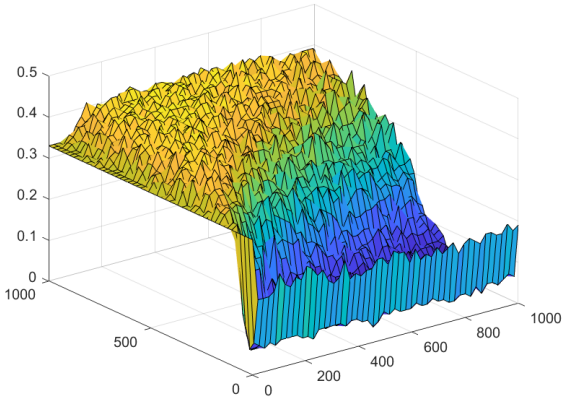
  Now, in words this means that when using random split scheme the solution will not be as stable as any other scheme. Therefore, there is a trade-off between stability and processing time with respect to the training process. On the opposite, leave-one-out validation provides the most stable solution with respect to selection of the parameters. In this case it is very likely that a good pair of parameters γ and σ^2 is found to obtain a good solution. There is, however, one problem associated to this scheme: processing time. If the dataset is large, it is relatively impossible to make an exhaustive grid search, as in this case, because it takes a lot of time to get the best estimates.

  That being said, cross-validation is usually the preferred method to get a pair of parameters that produces a good solution in terms of misclassification error. In this case, there are several pairs of parameters that provide a misclassification rate of just 3%, so one of them is chosen:

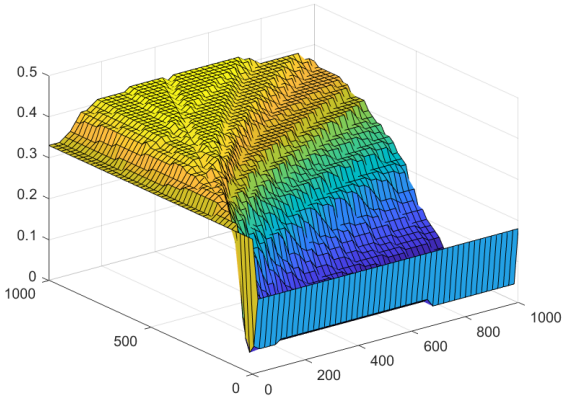
$\gamma = 469.38$ and $\sigma^2 = 20.4091$.



(a) Random split



(b) Cross-validation



(c) Leave-one-out validation

Figure 8: Comparison of validation schemes

- ***Why should one prefer cross-validation over simple validation (random split)? How to choose the value of k in k-fold crossvalidation?***

  As it is shown in Figure 8, the search for the best pair of parameter is more stable in cross-validation than in simple validation, for this reason is preferred cross-validation. Now, in terms of what value of k should be preferred, there are no rules that establish in which situations is better to choose one over the other. Five or ten-fold cross-validation are usually the schemes preferred by researches and this decision is mostly influenced by the number of observations in the training set.

1.3.3 Automatic parameter tuning

- ***Try out the different 'algorithm'. What differences do you observe? Why do the obtained hyperparameters differ a lot in different runs? What about the cost? Computational speed? Explain the results.***

  To answer these questions, each algorithm (`simplex` or `gridsearch`) was run 10 times and a comparison of hyperparameters, misclassification rates and processing times was conducted. Table 1 shows these results.

i	Simplex				Gridsearch			
	γ	σ^2	Misclass.	Time	γ	σ^2	Misclass.	Time
1	0.07	1.507	0.02	1.564	0.428	0.386	0.04	0.994
2	0.195	0.032	0.03	0.484	0.042	0.541	0.03	0.864
3	90.07	0.612	0.04	0.427	87.749	0.03	0.04	0.874
4	16609.847	2.369	0.04	0.439	0.43	0.272	0.04	1.219
5	6685.969	0.32	0.03	0.592	0.264	0.278	0.04	0.962
6	2.307	0.311	0.04	0.429	0.25	0.135	0.04	1.04
7	157.976	0.112	0.03	0.383	0.042	0.579	0.03	0.855
8	3.876	0.228	0.04	0.369	159.938	0.026	0.03	0.97
9	28.451	60.643	0.03	0.499	0.053	0.223	0.03	0.904
10	2.822	13.262	0.04	0.601	0.16	0.529	0.04	0.833

Table 1: Comparison of algorithm results over different runs (i)

The first thing that can be notice is the difference between pairs of parameters from one run to the other under the same algorithm. For instance in the `simplex` algorithm, the pair of parameters of the first run is (0.07, 1.507) and the fourth run is (16609.85, 2.37). This happens in both algorithms because the routine in Matlab do the following:

1. Use Coupled Simulated Annealing (CSA) to determine suitable starting points for every method.
2. These starting points are then given to one of the three optimization algorithms (`simplex`, `gridsearch` or `linesearch`).

According to the `LS-SVMlab Toolbox` user's guide, CSA is more effective than multistart gradient descent optimization and it uses the acceptance temperature to control the variance of the acceptance probabilities. This means that this algorithm is an improved optimization routine because it reduces the sensitivity of the algorithm to the initialization parameters. This can actually be confirmed by the misclassification error in each run. No matter what pair of parameters is chosen in each run, the misclassification rate is more or less constant.

Now in terms of computational speed there is a notable difference between algorithms. The average speed of `simplex` algorithm is 0.5787 seconds, whereas for the `gridsearch` algorithm is 0.9515 seconds. This difference does not seem to be significant in this example, but if a more complex problem is analyzed, differences in processing time can be important.

In the end, the decision of which algorithm to use depends on the particular problem and the time at hand to do the analysis. `Gridseach` will almost always be slower than `simplex`, because it makes the search of the pair of parameters in an exhaustive manner.

1.3.4 Using ROC curves

- ***In practice, we compute the ROC curve on the test set, rather than on the training set. Why?***

  Because it is always of interest to check whether the model chosen **generalizes** well or not. It would not be as useful if the ROC curve was computed on the training set because any model, specially the more complex ones, is able to obtain an area under the ROC curve equal to 1 (leading usually to overfitted solutions).

- ***Generate the ROC curve for the `iris.mat` dataset (use tuned `gam` and `sig2` values). Interpret the result.***

  Figure 9 shows the ROC curve using an RBF kernel with $gamma = 0.0425$ and $\sigma^2 = 0.5412$. The area under the ROC curve is one, meaning that the model generalized well the behaviour found in the training set. In this case, no matter what cut-off value is used to classify new observations, the accuracy of the prediction will always be 100%. Considering that this is a toy example, this results are somewhat expected.

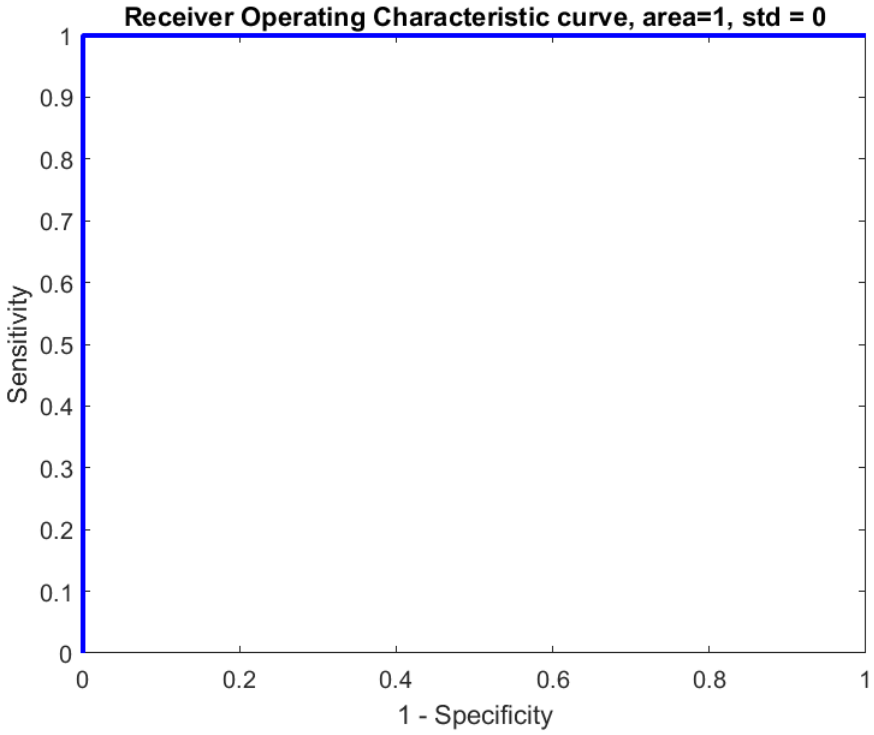


Figure 9: ROC curve

1.3.5 Bayesian framework

- ***Using the Bayesian framework, it is possible to get probability estimates. How do you interpret the colors of the plot?***

  Figure 10 shows the probability estimates for the solution of an RBF kernel with $gamma = 0.0425$ and $\sigma^2 = 0.5412$, as in the previous item. This plot can be interpreted as a heat map. If a new observation is located in the upper-right corner or the lower-left corner, it has a probability of almost one to be in the positive class. On the contrary, if a new observations is located in the middle of the plot, its probability of being in the negative class is one. These are the ranges where the model will predict with certainty.

  However, if a new observation is located in the upper-left corner or the lower-right corner, its probability of belonging to a positive class is around 0.5.

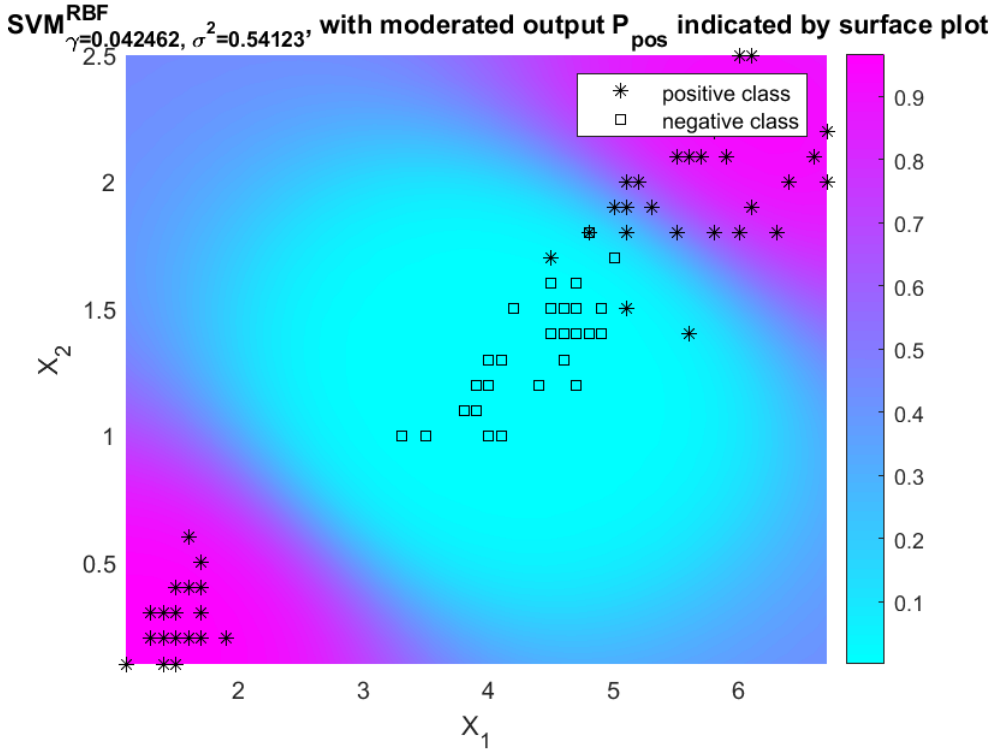


Figure 10: Probability estimates

- ***Change the values of `gam` and `sig2`. Visualize and discuss the influence of the parameters on the***

  figure.

  Figure 11 shows the probability estimates for the solution of an RBF kernel with $gamma = 0.5$ and $\sigma^2 = 1$. The range of these probabilities is similar to the previous plot. However, there are differences in the upper-left corner and the lower-right corner, where the previous model was unsure about the classification of a new point. In this case, if a new point is located in any of these corners, it will be classified as positive.

  As it was mentioned before, any change in any of these parameters will impact directly the quality of the prediction on new data points.

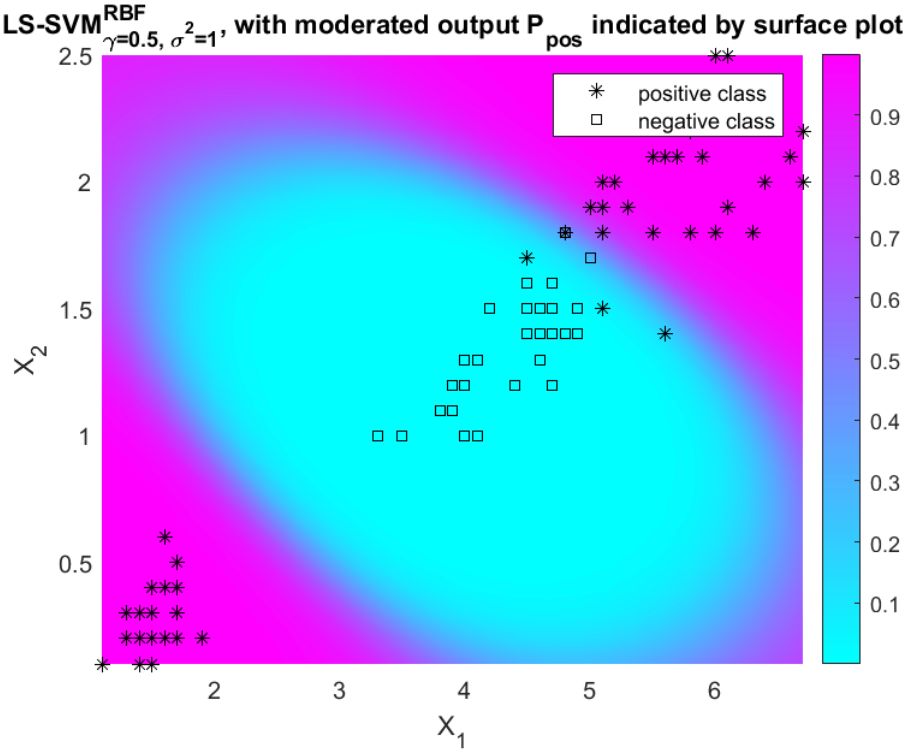


Figure 11: Probability estimates with other parameters

2 Homework problems

2.1 Ripley Dataset

This dataset consists of two classes where the data for each class have been generated by a mixture of two normal distributions. Figure 12 shows the pattern in the data, given that only two input variables are available. A polynomial kernel could be a good idea because the number of misclassifications can be reduced when comparing it to a linear kernel. Also an RBF kernel may work in this problem because the flexibility of this model can be exploited.

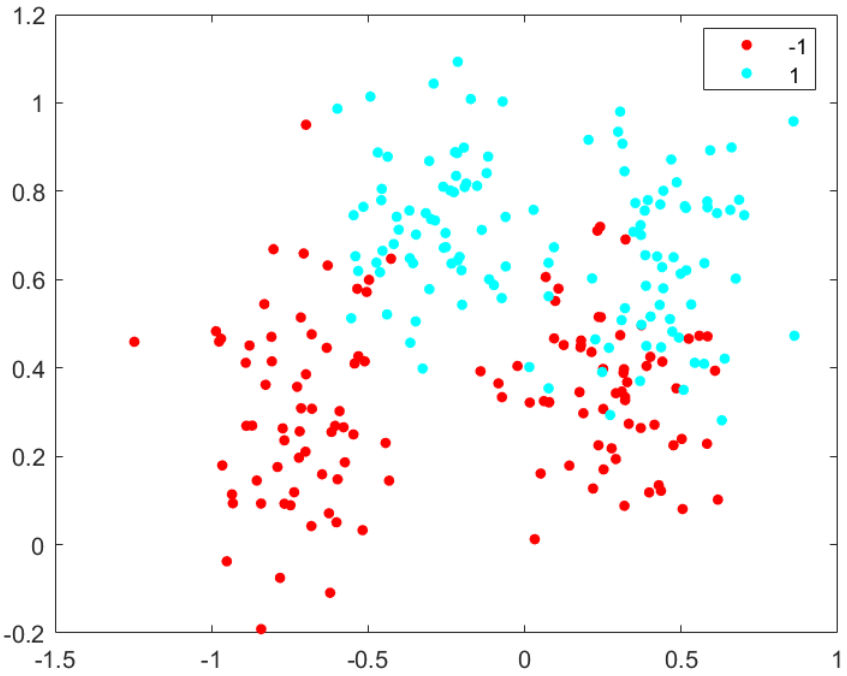


Figure 12: Visualization of Ripley dataset

These three different kernels were tried in the search for the best classification boundary. In all of them, the determination of the tuning parameters was conducted by running a combination of Couple Simulated Annealing (CSA) and a standard simplex method ten times and also running a combination of CSA and a gridsearch (linesearch in linear kernel) ten times. From this results, the combination of parameters that produced the best misclassification rate was chosen. Table 2 shows the best hyperparameters, the misclassification rate on the training set and the algorithm that produced these results for each kernel.

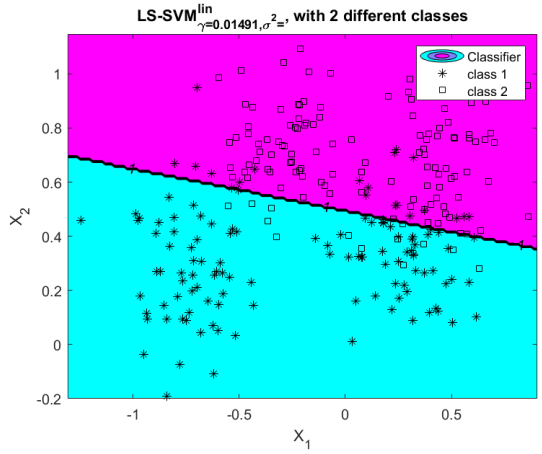
Kernel	**Hyperparameters**	**Misclassification**	**Algorithm**
Linear	$Reg = 0.0149$	0.1320	simplex
Polynomial	$Reg = 6.3019$	0.12	simplex*
	$t = 1.1712$		
	$degree = 7$		
RBF	$Reg = 2.2479$	0.1120	simplex*
	$sig2 = 0.8305$		

Table 2: Tuned hyperparameters for each kernel

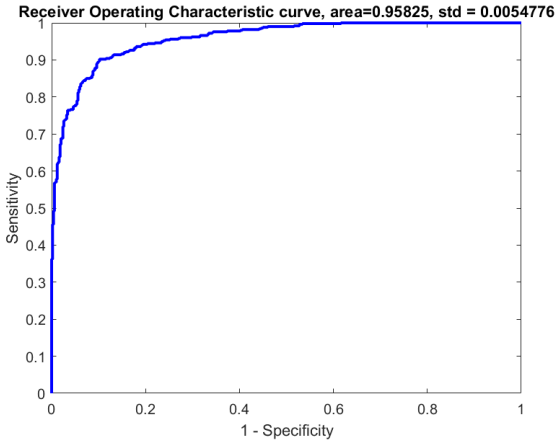
As it was expected, the more complex the model, the better the fit on the training set. The misclassification rate in the Linear kernel is about 13% and in the RBF is around 11%. It is important to mention that `gridsearch` algorithm in the polynomial and RBF kernel led to the same misclassification error that simplex algorithm, this is why there is an '*' in the end of the word.

These tuned hyperparameters were then used to train each model and predict the corresponding labels on a test set. To measure the performance of the model in a new dataset, the ROC curve is analyzed. Figure 13 presents the resulting classification region and the ROC

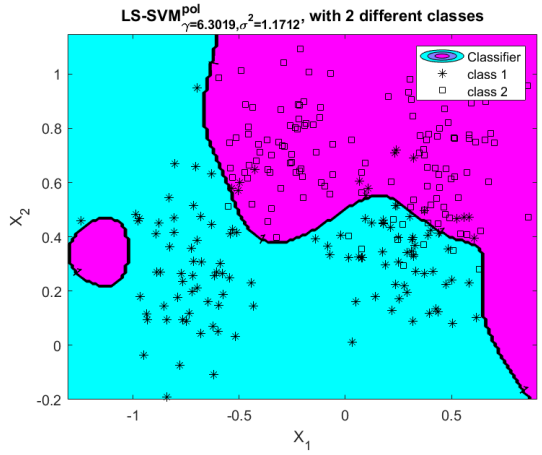
curve for each model. It can be seen that even though the linear kernel produced good results in terms of performance with an Area Under the ROC Curve (AUC) of 0.958, the RBF kernel is the chosen one because its AUC is 0.969. On the other hand, the classification region of the polynomial kernel seems to be overfitted because of the small region that appears on the left-hand side of the plot.



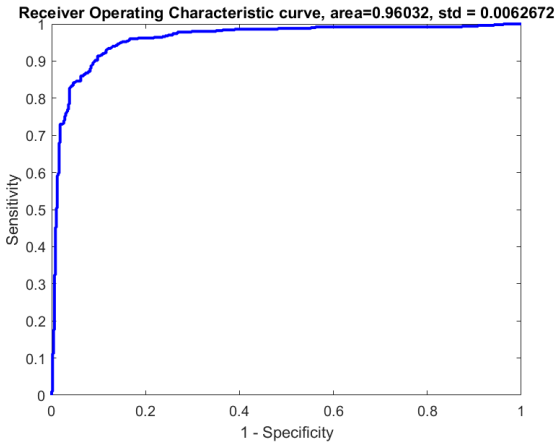
(a) Linear Kernel



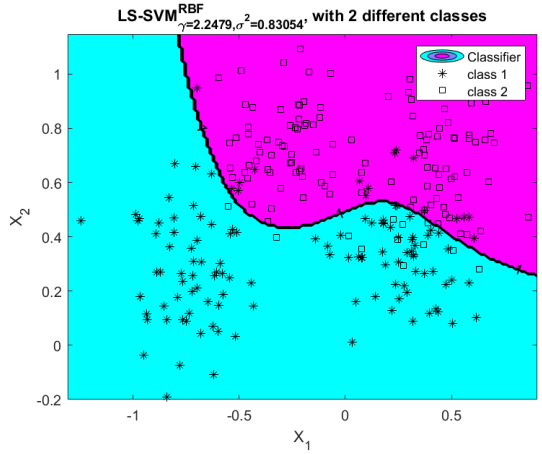
(b) Linear Kernel



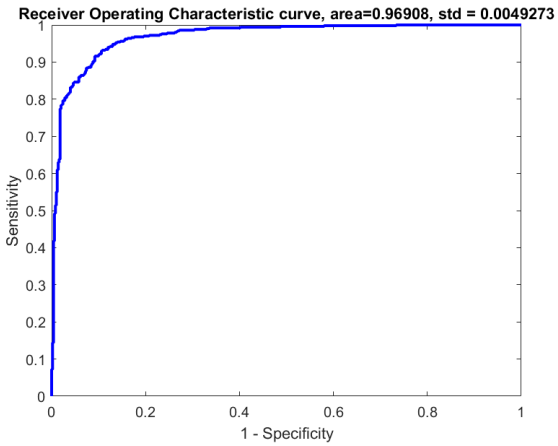
(c) Polynomial Kernel



(d) Polynomial Kernel



(e) RBF Kernel



(f) RBF Kernel

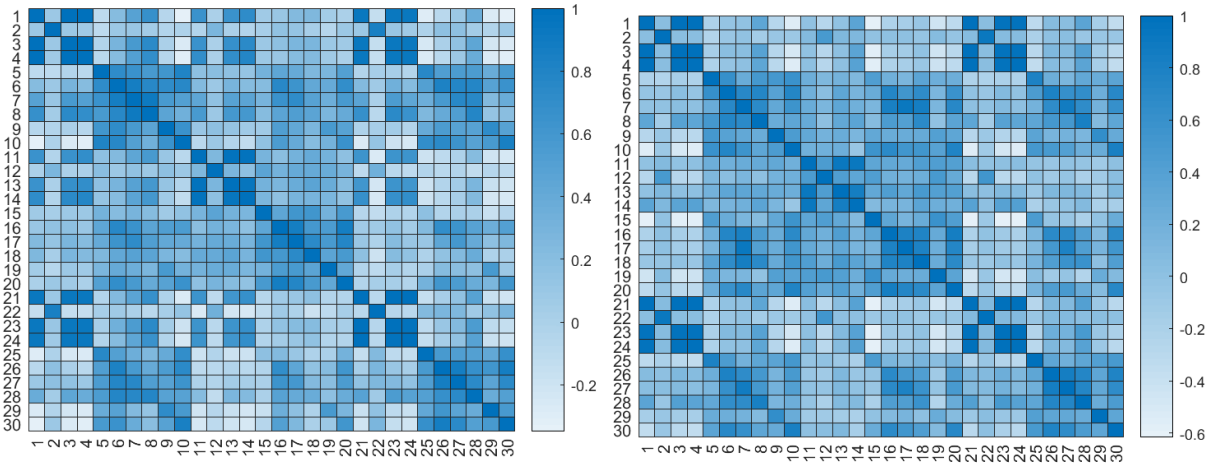
Figure 13: Classification boundary and ROC curve for each kernel

To conclude, different kernels were used to get the best classification boundary possible, namely linear, polynomial and RBF. The parameters of each model were tuned using a combination of CSA and simplex or gridsearch algorithm. The kernel that has best predictive performance in terms of area under the ROC curve is the RBF. Now, to judge whether this is the best methodology, it is important to mention that there are different ways to come up with a good classification boundary, such as generalized linear models, decision trees, random forests, among others. Since this is problem involves simulated data and therefore there is no need to explain possible relationships between the independent variables and the response, it can be concluded that SVM makes a good job.

2.2 Breast Dataset

This dataset is composed of several cell-nucleus measures taken from 569 patients with the purpose of creating a classifier that helps the practitioner to decide whether new patients are likely to have breast cancer or not. Measures such as radius, texture, perimeter, area, smoothest and other 5 characteristics about cell-nucleus are considered in this problem. The dataset was divided into training and test dataset, with 400 patients in the first set and the remain 169 on the test set. In the training set, 250 patients do not have cancer and the remaining 150 have it.

To get an insight of what is happening inside the training dataset, Figure 14 shows the correlation matrix, presented as a heat map, for each class. Unfortunately, variable labels are not present in the data, so numbers are used to reference a variable in particular. Overall, it seems that there is not any difference in correlations between patients with cancer and without it. However, there are some differences: first, the correlation between variables from 5 to 20 is stronger in patients without cancer. Second, the correlation between variables 25 to 30, with all of the remain is also stronger in patients without cancer.



(a) Breast Cancer (b) No Breast Cancer

Figure 14: Correlation matrix by class

Since it is difficult to see the pattern within each class because the high number of input variables, it is assumed that the classification region is nonlinear. For this reason, an RBF kernel may be a good idea to obtain a good classification region. However, linear and polynomial kernel can also be used to compare results and performance.

These three different kernels were tried in the search for the best classification boundary. In all of them, the determination of the tuning parameters was conducted by running a combina-

tion of Couple Simulated Annealing (CSA) and a standard simplex method ten times and also running a combination of CSA and a gridsearch (linesearch in linear kernel) ten times. From this results, the combination of parameters that produced the best misclassification rate was chosen. Table 3 shows the best hyperparameters, the misclassification rate on the training set and the algorithm that produced these results for each kernel.

Kernel	**Hyperparameters**	**Misclassification**	**Algorithm**
Linear	$Reg = 0.0795$	0.03	linesearch*
Polynomial	$Reg = 1.6 \times 10^{-6}$	0.0175	simplex
	$t = 161.7506$		
	$degree = 3$		
RBF	$Reg = 43.1866$	0.0075	simplex
	$sig2 = 28.5518$		

Table 3: Tuned hyperparameters for each kernel

As it was expected, the more complex the model, the better the fit on the training set. The misclassification rate in the Linear kernel is 3% and in the RBF is almost 1%. It is important to mention that `linesearch` algorithm in the linear kernel led to the same misclassification error that simplex algorithm, this is why there is an '*' in the end of the word.

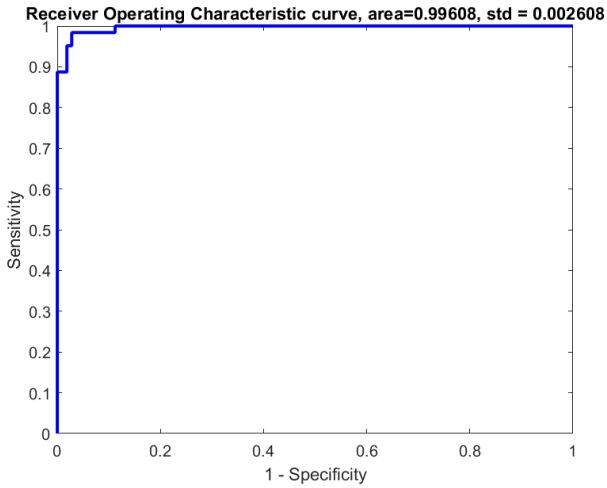
These tuned hyperparameters were then used to train each model and predict the corresponding labels on a test set. To measure the performance of the model in a new dataset, the ROC curve is analyzed. Since it is not possible to plot the classification region of each model, Figure 15 presents only the ROC curve for each model. Surprisingly, the linear kernel is the model with the best performance, having an area under the ROC curve of 0.9961. The second best model, is the RBF kernel which AUC is 0.9944. And the third model is the polynomial kernel with an AUC of 0.9818.

To conclude, different kernels were used to get the best classification boundary possible, namely linear, polynomial and RBF. The parameters of each model were tuned using a combination of CSA and simplex or gridsearch algorithm. The kernel that has best predictive performance in terms of area under the ROC curve is the linear. Now, to judge whether this is the best methodology, it is important to mention that there are different ways to come up with a good classification boundary, such as generalized linear models, decision trees, random forests, among others. Since this problem involves real life data where the interpretation of the results is important, and given that the final kernel is linear, another kind of model can be used without losing precision, such as a logit model.

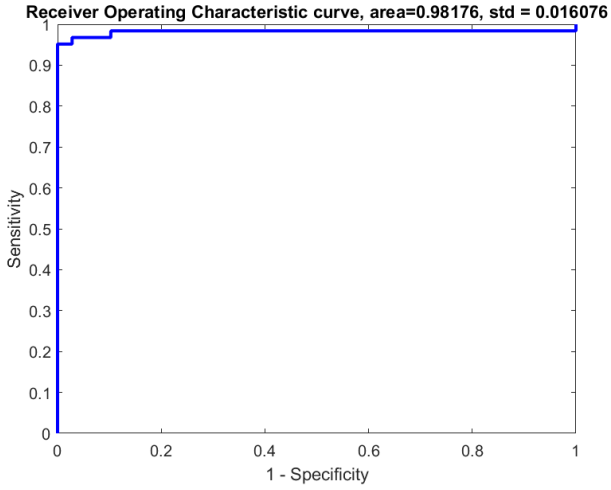
2.3 Diabetes Dataset

This dataset is composed of eight variables taken from 468 patients with the purpose of creating a classifier that helps the practitioner to decide whether new patients are likely to have diabetes or not. The dataset was divided into training and test dataset, with 300 patients in the first set and the remain 168 on the test set. In the training set, 205 patients do not have diabetes and the remaining 95 have it.

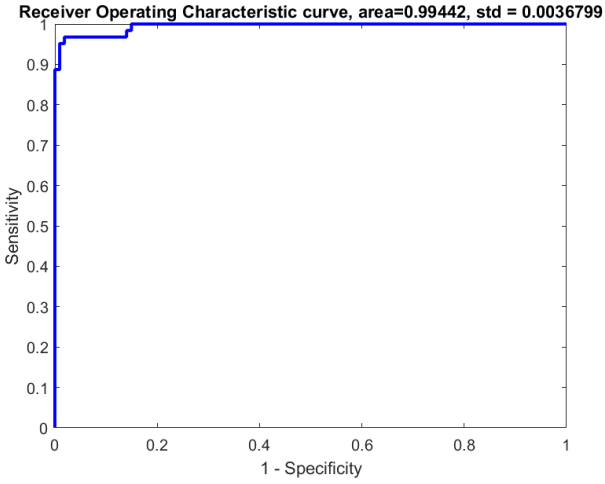
To get an insight of what is happening inside the training dataset, Figure 16 shows for each class a matrix with scatter plots between input variables and in the diagonal the corresponding histogram. Unfortunately, variable labels are not present in the data, so numbers are used to reference a variable in particular. In the diagonal, there are differences in the distribution of the second variables, where patients with diabetes tend to have higher values than those



(a) Linear Kernel



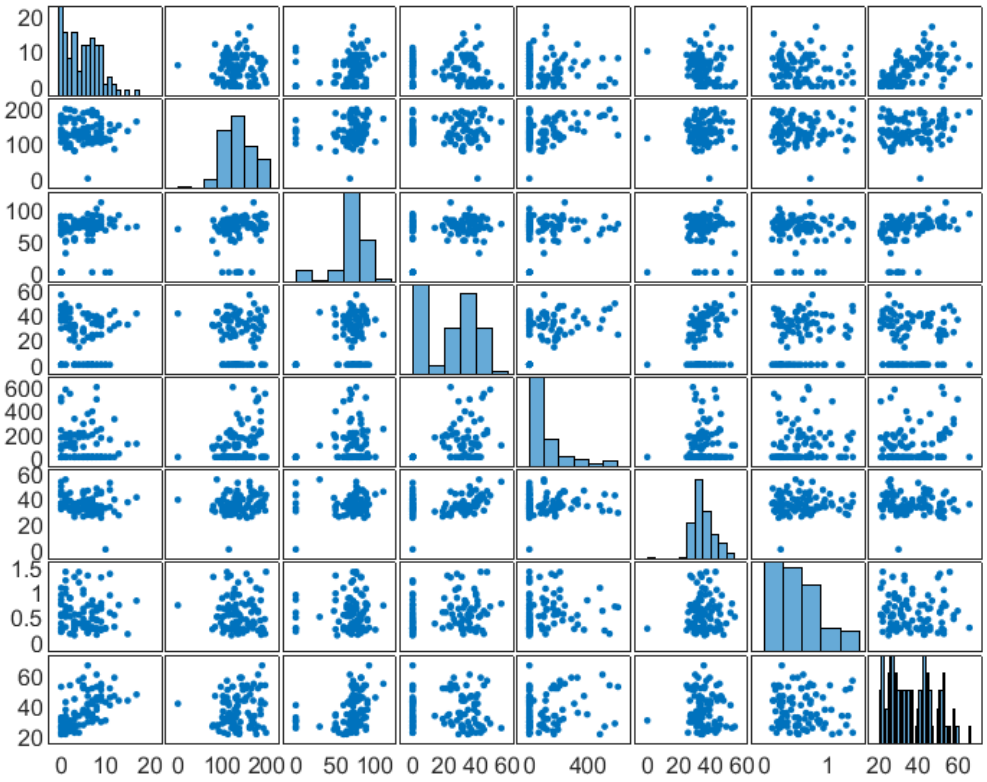
(b) Polynomial Kernel



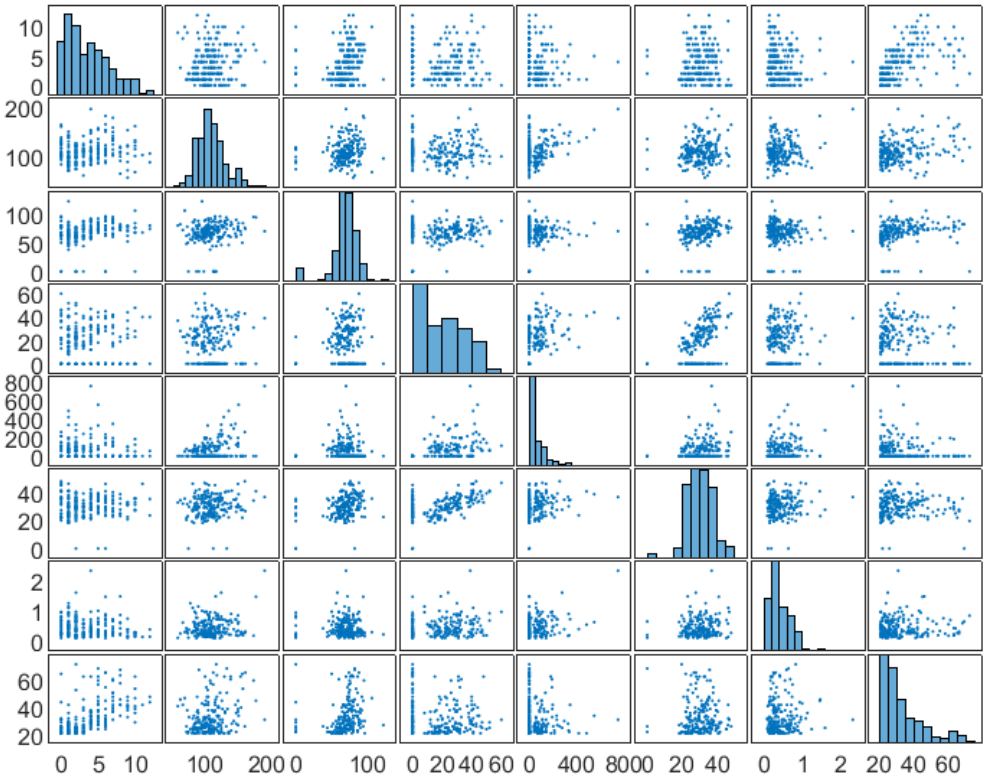
(c) RBF Kernel

Figure 15: ROC curve for each kernel

who do not have the disease, and the seventh variable, where the distribution is taller in low values. With respect to the scatter plots, there are differences between patients with diabetes and patients without it, in terms of localization of the cloud of observations. As a illustration, in the scatter plot of the first variable against the second, the scores are higher in patients with diabetes when compared to those who do not have it.



(a) Diabetes



(b) No Diabetes

Figure 16: Scatter plot by class

Since it is difficult to see the multivariate association between the input variables and the target variable, it is assumed that the classification region is nonlinear. For this reason, an RBF kernel may be a good idea to obtain a good classification region. However, linear and polynomial kernel can also be used to compare results and performance.

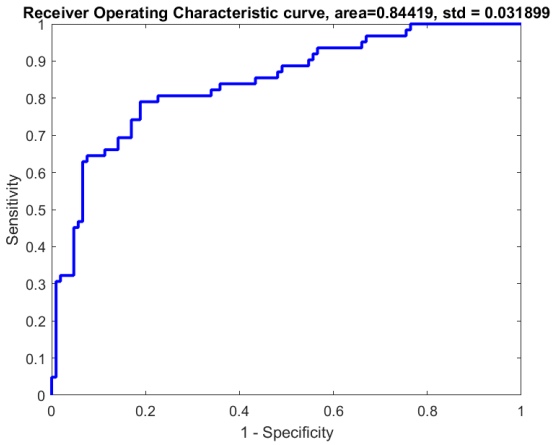
These three different kernels were tried in the search for the best classification boundary. In all of them, the determination of the tuning parameters was conducted by running a combination of Couple Simulated Annealing (CSA) and a standard simplex method ten times and also running a combination of CSA and a gridsearch (linesearch in linear kernel) ten times. From this results, the combination of parameters that produced the best misclassification rate in the training set was chosen. Table 4 shows the best hyperparameters, the misclassification rate and the algorithm that produced these results for each kernel.

Kernel	**Hyperparameters**	**Misclassification**	**Algorithm**
Linear	$Reg = 0.0475$	0.23	linesearch
Polynomial	$Reg = 2.5 \times 10^{-6}$	0.24	simplex
	$t = 314.0988$		
	$degree = 3$		
RBF	$Reg = 4.9 \times 10^{4}$	0.2233	gridsearch
	$sig2 = 2.9 \times 10^{3}$		

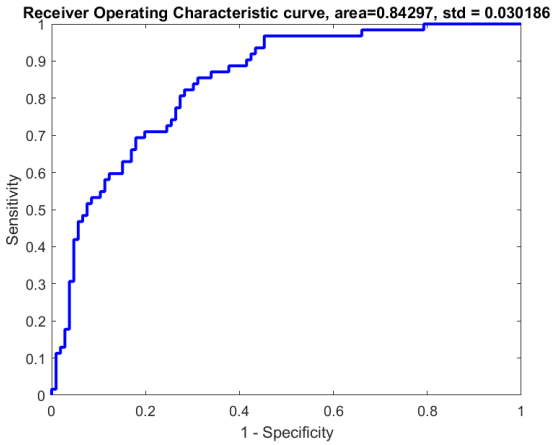
Table 4: Tuned hyperparameters for each kernel

As it was expected, the more complex the model, the better the fit on the training set. Although the differences in performance between kernels is not large: the misclassification rate in the linear kernel is 23% and in the RBF is 22.33%, a really small difference. Note that different algorithms led to best performance in each kernel, being the gridsearch (linesearch) the algorithm that performed best.

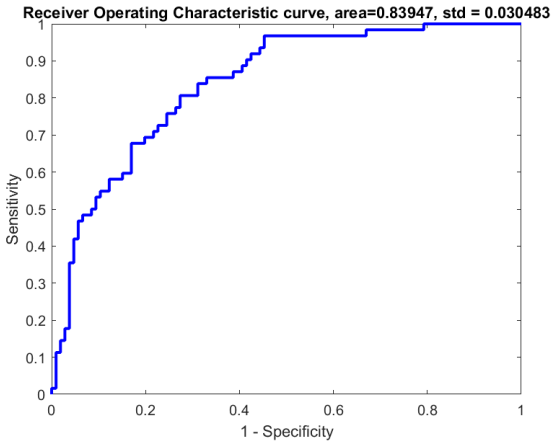
These tuned hyperparameters were then used to train each model and predict the corresponding labels on a test set. To measure the performance of the model in a new dataset, the ROC curve is analyzed. Since it is not possible to plot the classification region of each model, Figure 17 presents only the ROC curve for each model. Surprisingly, the linear kernel is the model with the best performance, having an area under the ROC curve of 0.8441. The second best model, is the polynomial kernel which AUC is 0.843. And the third model is the RBF kernel with an AUC of 0.8395.



(a) Linear Kernel



(b) Polynomial Kernel



(c) RBF Kernel

Figure 17: ROC curve for each kernel

To conclude, different kernels were used to get the best classification boundary possible, namely linear, polynomial and RBF. The parameters of each model were tuned using a combination of CSA and simplex or gridsearch algorithm. The kernel that has best predictive performance in terms of area under the ROC curve is the linear. Now, to judge whether this is the best methodology, since this problem involves real life data where the interpretation of the results is important, and given that the final kernel is linear, another kind of model can be used without losing precision, such as a logit model.