Matteo D'Ospina s252973

Homework 2 – SVM

Dataset:

We are working with a data set named Iris. This data sets consists of 3 different types of irises' (Setosa, Versicolour, and Virginica) petal and sepal length, stored in a 150x4 numpy.ndarray. We only work with the first two features of each images.

Purpose of project:

1. **Linear SVM:** Svm is a binary classifier that aims to maximise the minum margin between two classes. Although it's a binary classifer can be adapted at a dataset with more classes adopting a logic one vs one or one vs all.
2. **RBF kernel:** The Radial basis function kernel, also called the RBF kernel, or Gaussian kernel, is a kernel that is in the form of a radial basis function (more specifically, a Gaussian function). The RBF kernel is defined as $K_{\mathrm{RBF}}(\mathbf{x}, \mathbf{x}') = \exp\left[-\gamma \|\mathbf{x} - \mathbf{x}'\|^2\right]$ where γ is a parameter that sets the "spread" of the kernel. Intuitively, the gamma parameter defines how far the influence of a single training example reaches, with low values meaning 'far' and high values meaning 'close'. The gamma parameters can be seen as the inverse of the radius of influence of samples selected by the model as support vectors.
3. **K-Fold:** In k-fold cross-validation the data is first partitioned into k equally (or nearly equally) sized segments or folds. Subsequently k iterations of training and validation are performed such that within each iteration a different fold of the data is held-out for validation while the remaining k - 1 folds are used for learning.

Linear SVM:

In the first part of the program I split the data set in 3 part: training set, validation set and test set using the proportion 5:2:3

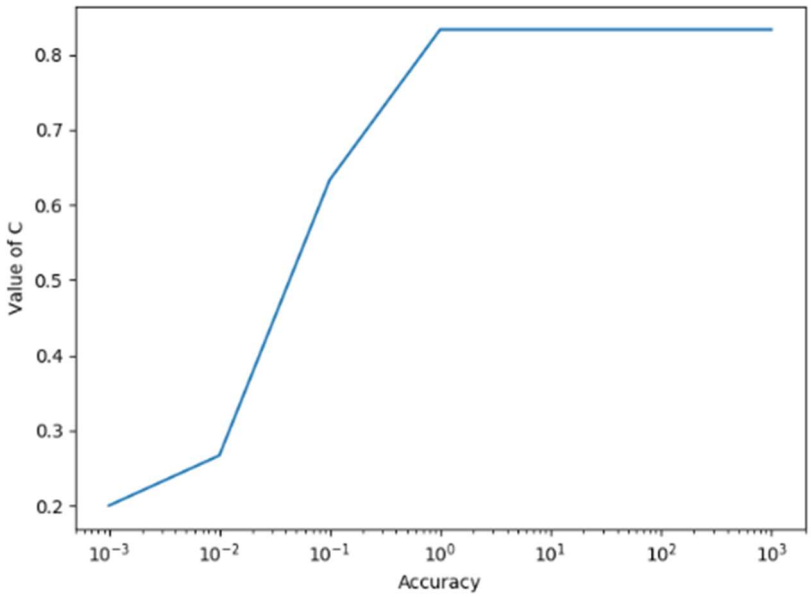
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.5)
X_test, X_val, y_test, y_val = train_test_split(X_test, y_test, test_size=0.4)
```

After that the model was trained on training set with the function "model.fit" and tested on validation set.

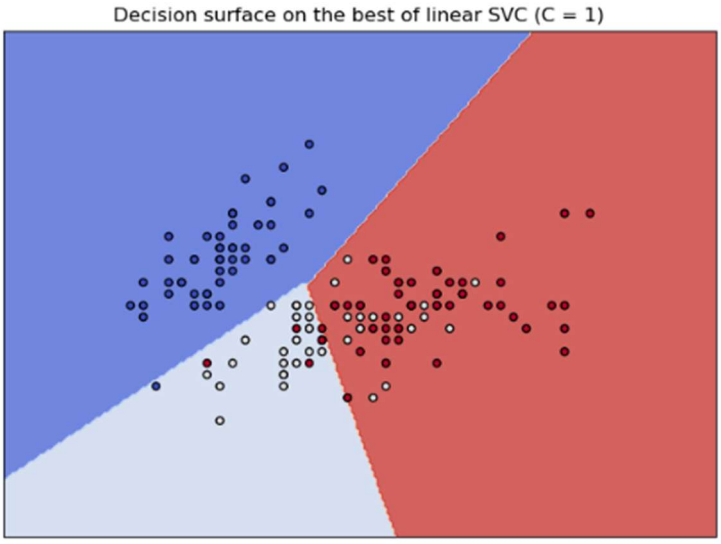
```
model = svm.SVC(kernel='linear', C=C)
clf = model.fit(X_train, y_train)
score = clf.score(X_val, y_val)
```

It was requested to change C parameter (from 0.001 to 1000) of the function in order to try to reach the best accuracy on validation set. The C parameter trades off correct classification of training examples against maximization of the decision function's margin. For larger values of C, a smaller margin will be accepted if the decision function is better at classifying all training points correctly. A lower C will encourage a larger margin, therefore a simpler decision function, at the cost of training accuracy. In other words `C` behaves as a regularization parameter in the SVM.
These are the scores I obtained. We can see that the best classification has the value of C = 0.1.

C	0.001	0.01	0.1	1	10	100	1000	Test set
Score	20%	26%	63%	83%	83%	83%	83%	75%



The accuracy of the model is also tested on the test set and we can see that the value is close to the one tested on validation set. That's because both sets come from the same data set.
This is the obtained decision boundaries.



RBF kernel:

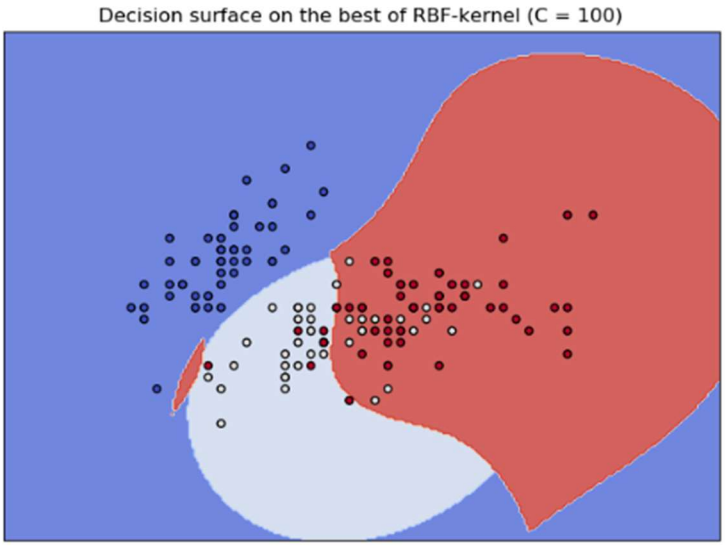
As before it was requested to find the best accuracy of the model.

```
model = svm.SVC(kernel='rbf', C=C, gamma='auto')
clf = model.fit(X_train, y_train)
score = clf.score(X_val, y_val)
```

At first it was asked just to change the value of C. These are the obtained values of accuracy.

C	0.001	0.01	0.1	1	10	100	1000	Test set
Score	20%	20%	63%	83%	83%	86%	73%	78%

Here we can see the decision boundaries of the best one.

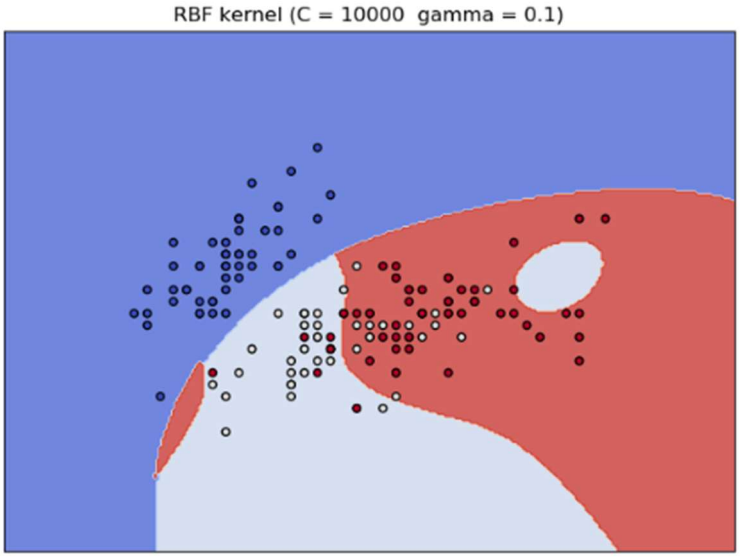


It's possible to see from the image, that it's one flower species (blue point) is linearly separable from the other two (red and white), but the other two are not linearly separable from each other. In fact thank to RBF Kernel, with the right configuration of parametes it's possible to reach a better value than linear svm.

Now it’s requied to change also the value of gamma an these are the results:

Gamma/C	0.001	0.01	0.1	1	10	100	1000	10000	100000
1e-09	20%	20%	20%	20%	20%	20%	20%	20%	20%
1e-08	20%	20%	20%	20%	20%	20%	20%	20%	20%
1e-07	20%	20%	20%	20%	20%	20%	20%	20%	60%
1e-06	20%	20%	20%	20%	20%	20%	20%	60%	70%
1e-05	20%	20%	20%	20%	20%	20%	60%	70%	83%
0.0001	20%	20%	20%	20%	20%	60%	70%	83%	83%
0.001	20%	20%	20%	20%	60%	70%	80%	83%	80%
0.01	20%	20%	20%	60%	70%	80%	80%	80%	80%
0.1	20%	20%	60%	66%	80%	80%	76%	86%	63%
1	20%	20%	63%	83%	83%	73%	60%	63%	70%
10	20%	20%	40%	80%	70%	56%	46%	46%	46%

The better configuration has a score of 86% on validation set that is very good and definitely better than a linear svm, it’s possible to see the plot:



K-Fold:

In this part it was asked to merge validation and training set, in order to have a proportion 7:3, and to use 5 folds for the algorithm. The best value of accuracy reached is 95%, with k=4 and different configuration of C and gamma, for example 100000 /0.01. The value is very good and is achieved thanks to the goodness of the k fold that allows to algorithm to train on the whole dataset.