Machine Learning and Deep Learning
Politecnico di Torino

Homework 1
Silvia Giammarinaro

Student ID: 269991

June 15, 2020

1 Introduction

The aim of this homework is to create and tune models using K-Nearest Neighbors and support vector machines, both classification algorithms taken from the sci-kit learn library. The dataset used is the Wine dataset, the data is a chemical analysis of wines grown in the same region in Italy but derived from three different cultivars. We have three different classes and we consider the following feautures: alcohol and malic acid. The dataset is divided in train, validation and test set (5:3:2).
After a brief analysis, we can see that the number of samples per classes is quiet balanced. Furthermore, plotting the two selected features, we don't have three well separated classes, there are might be some outliers. An outlier is a data point which does not reflect the distribution of the data.

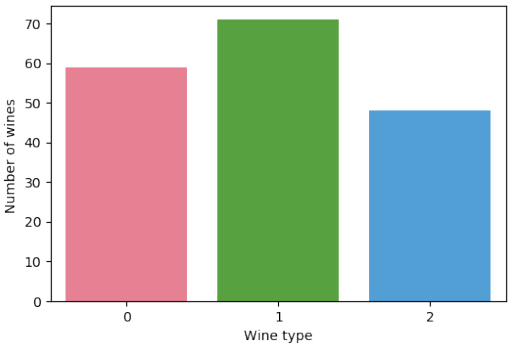


Figure 1: Dataset distribution

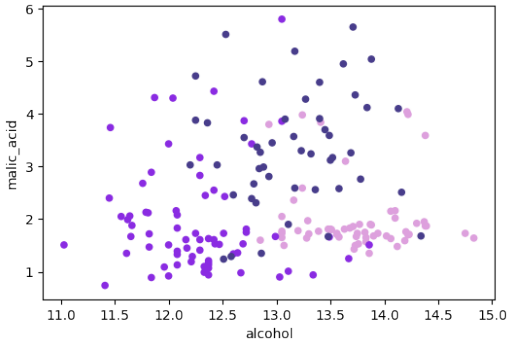
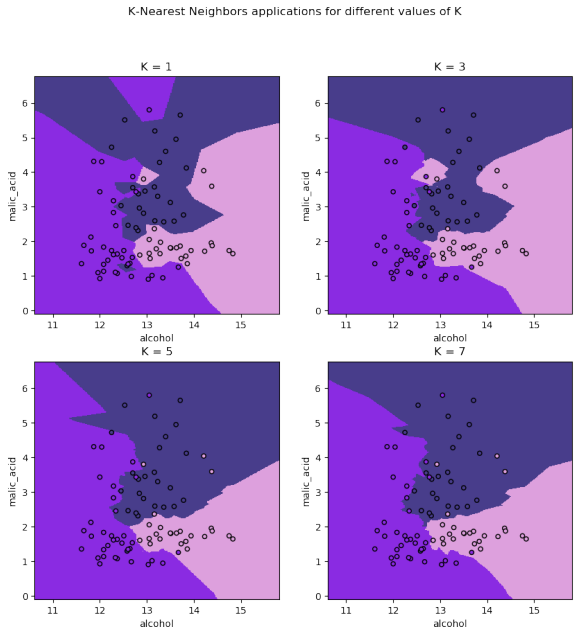


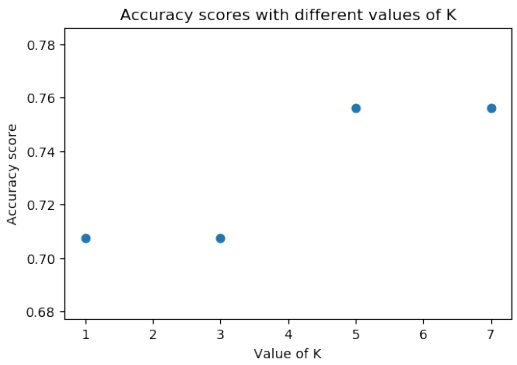
Figure 2: Plotting the samples with relative class

2 K-Nearest Neighbors

The first model we want to analyze is K-Nearest Neighbors. The hyperparameter we want to tune is K, the number of the neighbor on which the classification task is based. After defining a vector containing the desired K values, we apply the algorithm and plot all the data and the decision boundaries for the train set. Then we evaluate every model obtained in the validation set and pick the best for the test set. As the best model, we consider the one with the highest accuracy score, a metric commonly used to evaluate classification algorithms.



```
K  Accuracy
1  0.707317
3  0.707317
5  0.756098
7  0.756098
```

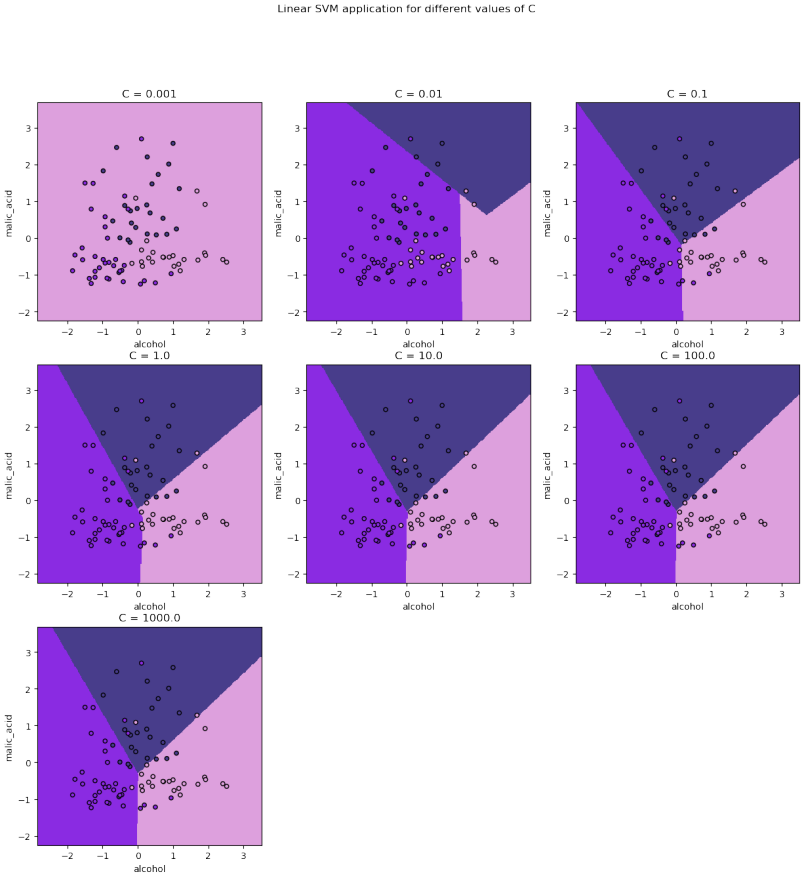



```
Accuracy score for the best value K = 5: 0.814
```

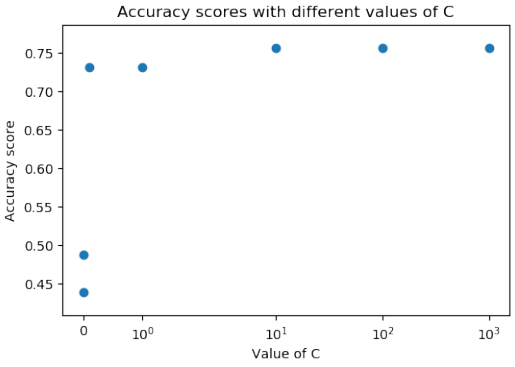
After this simulation, we can see how KNN works. KNN can be applied to linear or non-linear distributed data, but it is sensitive to outliers for a low K value. In the decision boundaries generated with K=1, there are two points isolated from the three macro groups and they are correctly classified. When we increase K, these two little areas disappear. Lastly consider we have taken two random features, maybe with a feature selection the classification improves.
Taking the best model from the validation set, the accuracy on the test set is 0.814, greater than the one obtained on the validation set.

3 Support vector machine with linear kernel

The second algorithm we want to analyze is Support vector machines. In this case, we consider a linear kernel and the hyperparameter to tune is C, which is inversely proportional to the width of the margin located between the classes. So if we have a low C, we get an hard margin classification, instead with an huge C we get soft margin. SVMs are sensitive to the feauture scales, so we use the method StandardScaler to standardize the data. In this problem we have three different classes, so the classifier applies a one vs one classification.



C	Accuracy
0.001	0.439024
0.010	0.487805
0.100	0.731707
1	0.731707
10	0.756098
100	0.756098
1000	0.756098

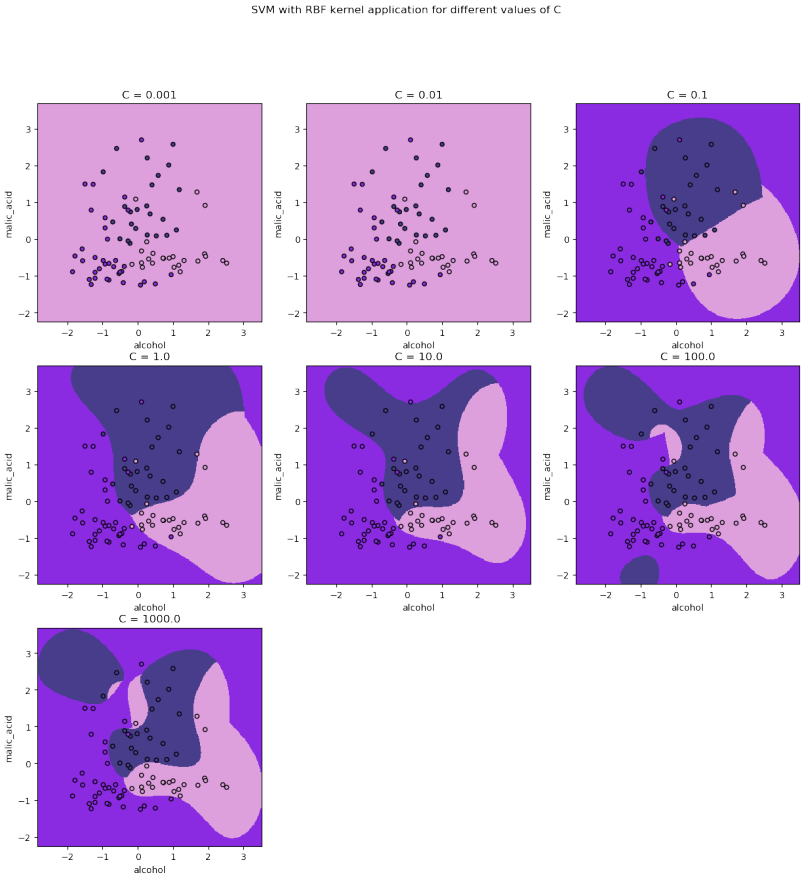


```
Accuracy score for the best value C = 10.0: 0.851
```

Plotting the decision boundaries, with C=0.001 a margin between the classes cannot be identified, a lower C requires a larger margin, which is not possible to create in this case.

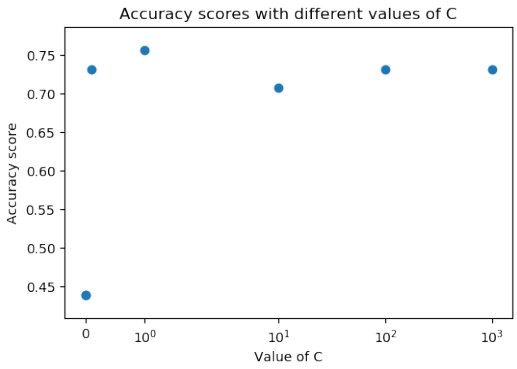
4 Support vector machine with RBF kernel

We apply the same analysis with a support vector machine with RBF Kernel. The RBF kernel applies the kernel trick, which is a mathematical technique used for mapping the data in a higher dimensional space. Note that with a high polynomial degree the model can deal with more complex datasets. We first analyze the model tuning the values of C.




```
Accuracy score for the best value C = 1.0: 0.833
```

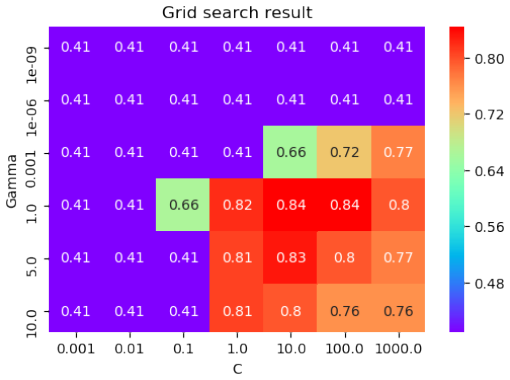
C	Accuracy
0.001	0.439024
0.010	0.439024
0.100	0.731707
1.000	0.756098
10.000	0.707317
100.000	0.731707
1000.000	0.731707

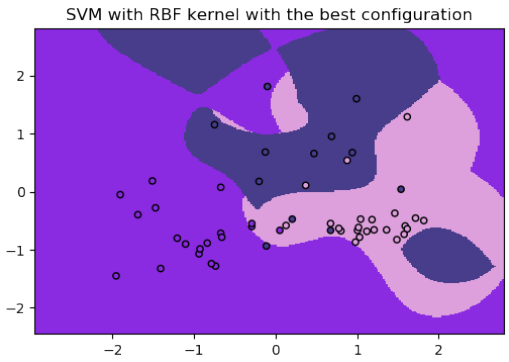


As we can see, the RBF kernel creates irregular curves to divide the data, instead of lines. The linear case requires the data to be linearly separable, RBF kernel instead can handle non-linear distributed data. Note that choosing a more higher dimensional space can cause overfitting because we are strongly relying on the training set.

5 Improving the model with grid search and k-Fold cross validation

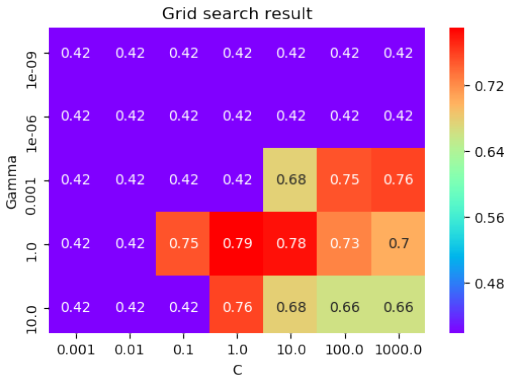
To find the best hyperparameters for the models, we are going to use the grid search, which evaluates the model for every combination of parameters indicated by the user. In this case, we introduce the tuning of the parameter γ. Increasing γ makes the curves narrower and the boundaries become more irregular. We can say that gamma is a regularization hyperparameter: if the model is overfitting, you should reduce it and vice versa. With grid search, k-fold cross validation is also applied. The k-fold cross validation is used to divide the training set into k distinct subsets. Then at each iteration, the k-subset is used as the validation set, meanwhile the other subsets are used as the training set. This is done for the better training of the classification task. Taking the best parameters from the grid search, we fit the model using the training set and we predict the target of the test set.

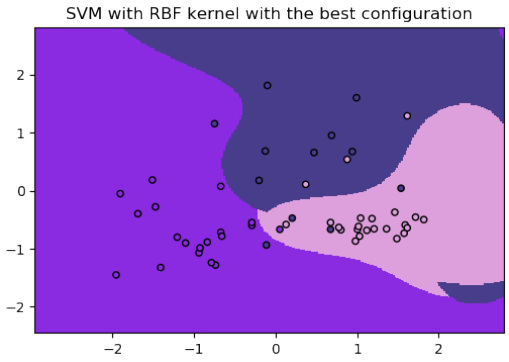




`Accuracy score for the best configuration {'C': 100, 'gamma': 1}: 0.814`

In the second passage, we consider both the train and the validation set to fit the models created by the grid search.





`Accuracy score for the best configuration {'C': 1, 'gamma': 1}: 0.833`

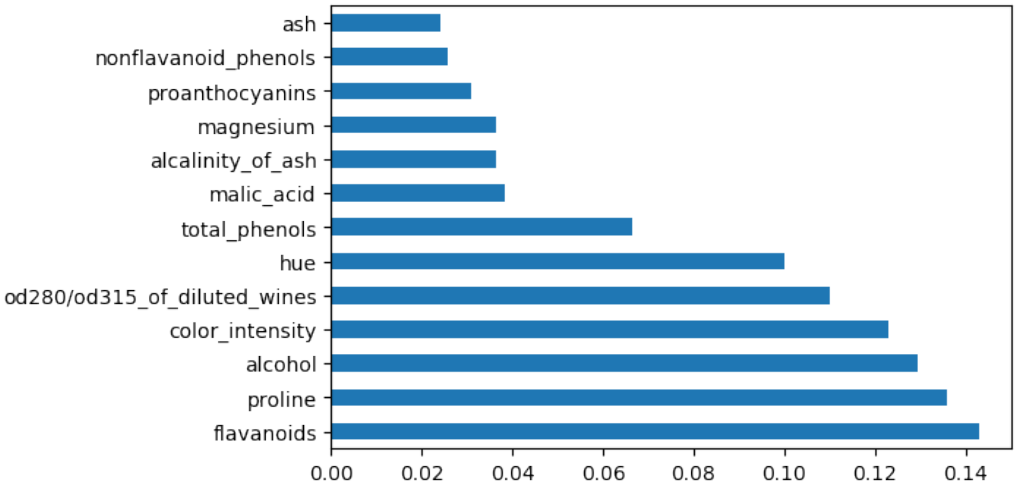
We can note that the accuracy score improves when considering also the validation set for fitting the model, but this is a dangerous approach. One should consider the validation set for tuning the hyperparameters and evaluate the performance. The test set is the last resource to use because this could lead to overfit. Even if the accuracy is greater in the second case, when deploying the model we could see a reduced performance. The aim of classification is to find a generalization of the problem, not to obtain an higher accuracy on the test set.

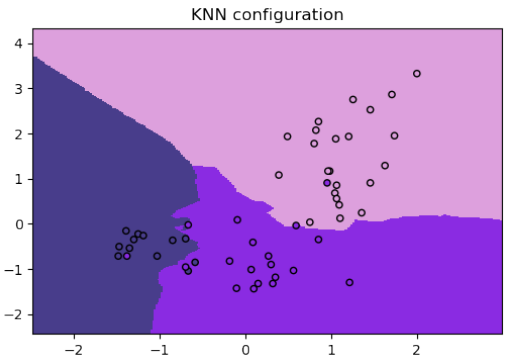
6 KNN vs SVM

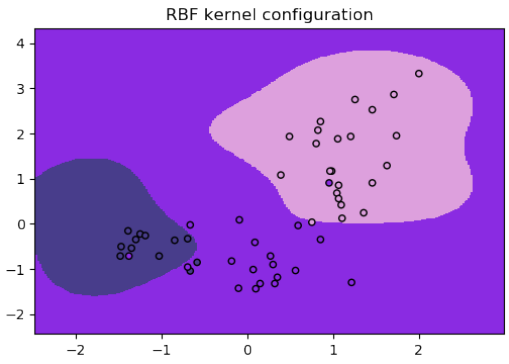
After applying both the algorithms, we can make some comparisons. KNN can handle both linear or non-linear distributed data, although tuning is a crucial part. This algorithm is also sensitive to outliers and a good selection of features is needed.
SVM can be used with linear or non-linear dataset using a proper kernel. It can handle outliers because it considers only relevant points to find the margin (support vectors). In the end this model performs well also in high dimensional space.

7 Features selection

We want to analyse if our predictions improve applying features selection. To extract the more relevant feature from the dataset we use the extra trees classifier. As we can see the two most relevant are flavanoids and proline, so we decide to pick these. As before we standardize the data with the standard scaler.
We compare two different configurations with the KNN algorithm and the SVM with RBF kernel using the best parameters obtained with the grid search. As we can see the accuracy improves so we have proven features selection is a crucial passage to obtain a better classification model.








```
Accuracy score for KNN {'K': 5}: 0.888
```

```
Accuracy score for RBF kernel {'C': 10, 'gamma': 1}: 0.907
```

8 References

Hands-On Machine Learning with Scikit-Learn, Keras, and Tensorflow: Concepts, Tools, and Techniques to Build Intelligent Systems - Aurélien Géron