# POLITECNICO DI TORINO

*MSc in Data Science and Engineering*

Machine learning and Deep learning

**Homework 1**

*Professor*:
Barbara Caputo

*Student*:
Paola Privitera

Academic Year 2019-2020

# Index

## Introduction

The aim of the project is to perform a classification of Italian wines contained in the Wine dataset through three different algorithms and discuss the outcomes of each method, starting from K-Nearest Neighbors, proceeding with Linear SVM and concluding with RBF Kernel.

The general procedure to determine a model's goodness is to partition the dataset into specific subsets: training, validation and test. The former is necessary to allow the model to understand and "learn" from the underlying data, derive their properties and be able to establish criteria for the classification. The validation set is necessary to test whether the training phase has been successful and determine which are the best model parameters to achieve the maximum accuracy. Finally, the test set (composed by "fresh data") aims at estimating the actual goodness of the model.

For each classification algorithm, we followed the aforementioned practice and provided a visual representation of the results for each analysis phase.

The selected methods present a different model structure to one another, which allows to classify the wines by different approaches. On one hand, the KNN algorithm exploits the prevalent cultivar in the neighborhood of a specific datapoint to determine the corresponding producer, and ultimately the size of the neighborhood has a significant impact on the accuracy results. The linear SVM exploits the identification of entire 2-dimensional planes to determine the cultivar, therefore the classification is no longer point-wise as in the KNN, but rather more generalized. Finally, the RBF Kernel is a variation of the classical SVM allowing for non-linear separators and thus non-linear hyperplanes. As the RBF Kernel relies onto two main parameters, $C$, the cost of miss-classification and $\gamma$, which is a measure of dispersion of the decision region, we performed the optimization of the algorithm initially by $C$, keeping $\gamma$ constant, and then through a grid search using both parameters.

We finally executed a study of the behavior of the RBF Kernel model (optimizing cross-parameter with the grid-search) on different configurations of the training and validation sets. To do so, we deployed the K-Fold technique which allows to shift the validation set across the training set. By moving the validation set at each fold, the model "learns" from a different dataset at each fold in round, leading to an understanding of the model results' consistency.

## 1 Dataset

The dataset contains 178 Italian wines produced by three different cultivars. Each wine is characterized by 13 measurements, resulting from a chemical analysis and the information about the belonging producer.

In order to optimize the visual representation of decision boundaries in a 2D space, we selected the first two features (“alcohol” and “malic_acid”) amongst the 13 attributes.

We split the dataset into train, validation and test sets adopting a random partitioning approach, keeping the 5:2:3 proportion. In particular, we used a customized partitioning algorithm which shuffles the data to obtain a well-diversified dataset and, through the stratification, it preserves the original labels’ distribution.

## 2 Classification methodologies

To serve the purpose of the project, we applied three different classification methods: K-Nearest Neighbors (KNN), Linear Support Vector Machine (LSVM), and RBF Kernel (RBF).

### 2.1 K-Nearest Neighbors

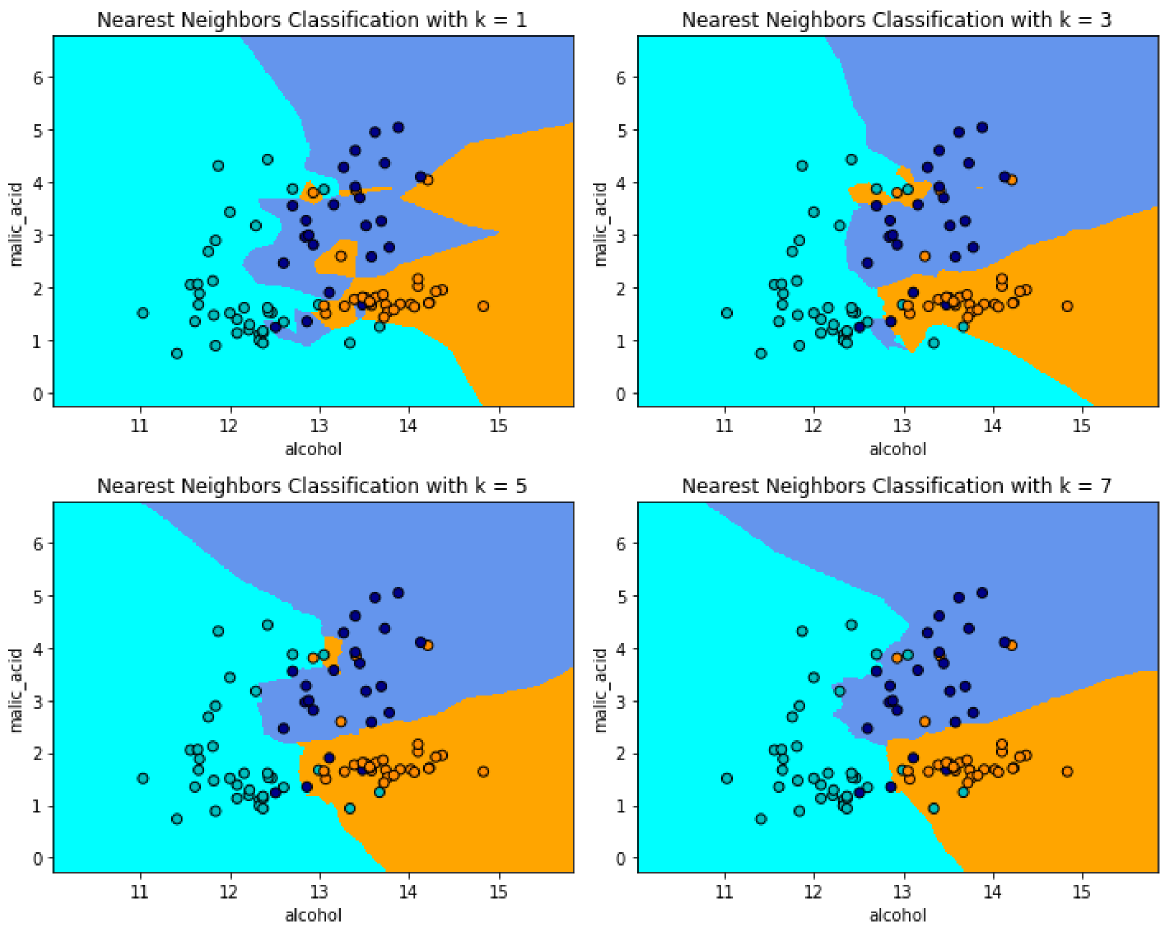
The KNN classifier computes the distance (in this case the Euclidean distance) of the object that needs to be classified from all the other objects in the training set. It identifies a number of *k* nearest neighbors and the object is classified by the majority vote of class labels in the neighborhood.

The selection of the value of *k* can be a difficult issue:

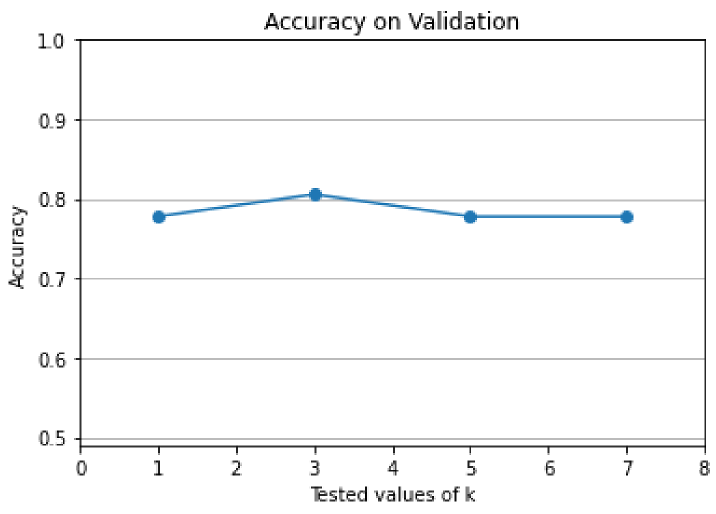
- if *k* is too small, the problem becomes sensitive to noisy points,
- if *k* is too large, the neighborhood may include points from other classes.

We tried the algorithm with four different values of the parameter *k* (respectively 1, 3, 5 and 7).

The results of the training phase are shown in the four graphs below. The individual dots represent the training items, and the background colors represent the cultivar class the wines are supposed to belong to according to their attributes’ values. The graphs suggest that, as *k* increases, the boundary lines (cultivar class region) become more homogenous. Indeed, with *k* equal to 1 the classification relies exclusively on the class of a single neighbor, which results in a more fragmented surface. On the contrary, with *k* equal to 7, the boundary lines smooth, because the class is attributed by the predominant cultivar among the neighbors. As mentioned, with a smaller value of *k* the algorithm is more sensitive to outliers (bad generalization) which means it is prone to overfitting.



To select the best model to apply on the test set, we performed the KNN algorithm on the validation set and calculated the corresponding accuracies for each value of *k*. The accuracy is computed as the number of correctly classified objects over the total classified items. As it can be seen in the following chart, *k* equal to 3 provided the most satisfactorily result (highest accuracy across the values of *k*) with an accuracy of 81%.



Therefore, we proceeded with the evaluation of the model (with $k$ = 3 as best value) on the test set, registering an accuracy of 74%. The decrease in accuracy, with respect to the validation round, might be attributed to a possible overfitting phenomenon coming from a relatively low number of classifying neighbors.

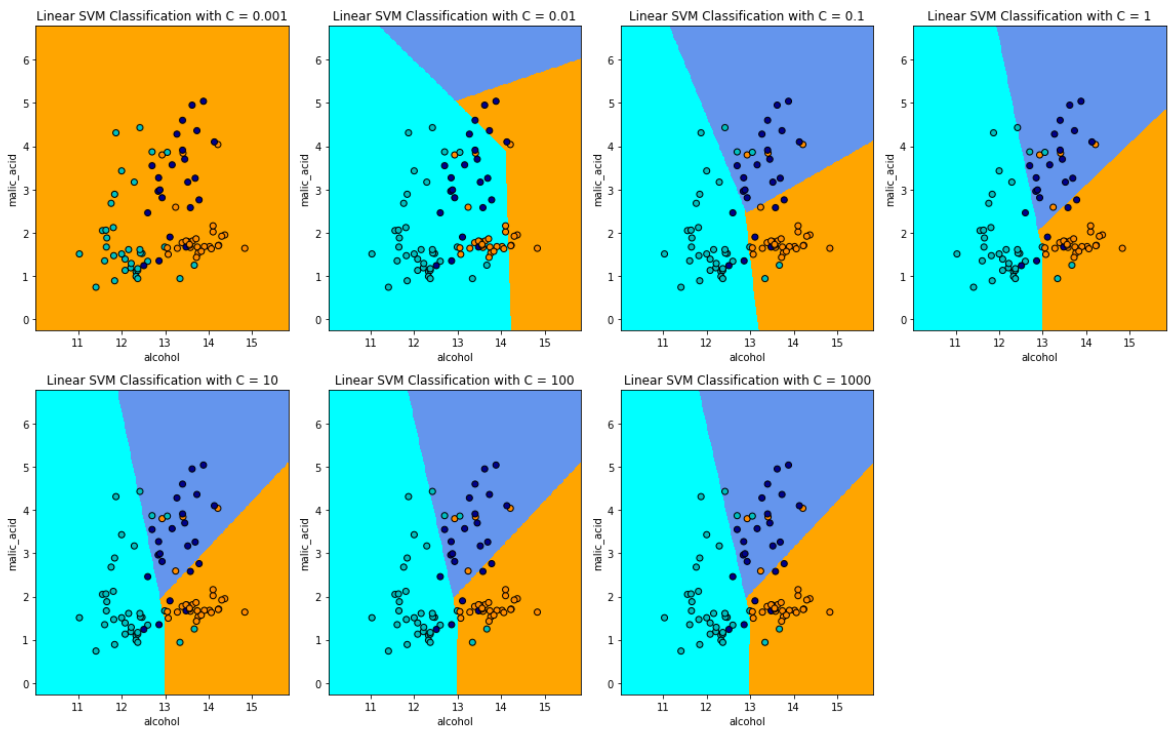
## 2.2 Linear SVM

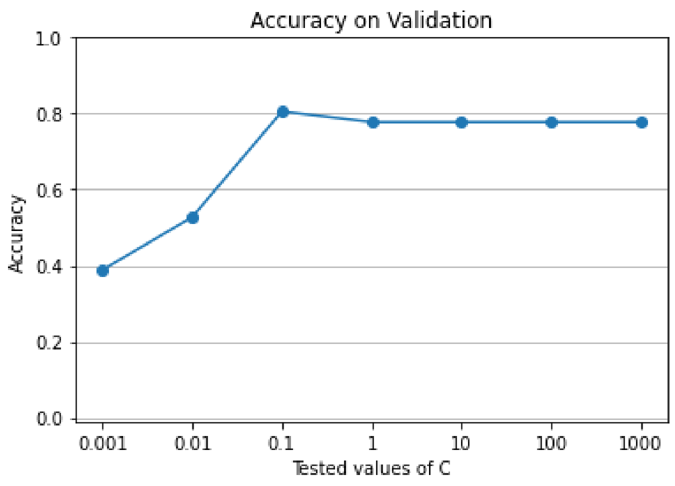
The aim of the Linear SVM is to identify linear hyperplanes that separate the items within the dataset, with the ultimate goal of maximizing the distance between each couple of nearest objects belonging to distinct decision boundaries and correctly classify as many instances as possible.
In case of noisy data, it is impossible to find a linear separator that correctly classifies all the instances; thus, we considered the addition of several slack variables as necessary. The number of the slack variables depends on a constant $C$ (cost of misclassification); however, the choice of the $C$ value presents what is called the "s*oft-margin problem*" for which, the higher the value of the parameter, the higher is the margin of error of the classification. Nonetheless, a higher value of $C$ might resolve, or at least constraint the possibility of miss-classification due to the presence of outliers in the dataset (from this the need of *softening the margins*). Hence, the choice of $C$ comes from the trade-off between a higher accuracy at the expenses of the likelihood of miss-classification.

For this project, we considered the following values of $C$:

$$C = [\ 0.001\ ,\ 0.01\ ,\ 0.1\ ,\ 1\ ,\ 10\ ,\ 100\ ,\ 1000\ ]$$

The charts below provide a visual representation of the classification for the different values of *C* and the corresponding accuracies on the validation set. As one might expect, for *C* equal to 0.001 the model delivered a rather poor accuracy of 39% (which basically means that only the wines belonging to the only cultivar identified have been correctly classified). Therefore, we observed that the best combination of cost of miss-classification and accuracy is reached for *C* equal to 0.1 resulting in an accuracy of 81%. Obviously, in case we obtained the same value of accuracy for two distinct values of *C*, we would have gone for the smaller cost of miss-classification.





Finally, we performed the evaluation of the model (with *C* = 0.1 as best value) on the test set, registering an accuracy of 76%. By comparing the result with the first methodology, the linear SVM improved the overall classification accuracy on the test set by 2 percentage points.

## 2.3 Kernel RBF

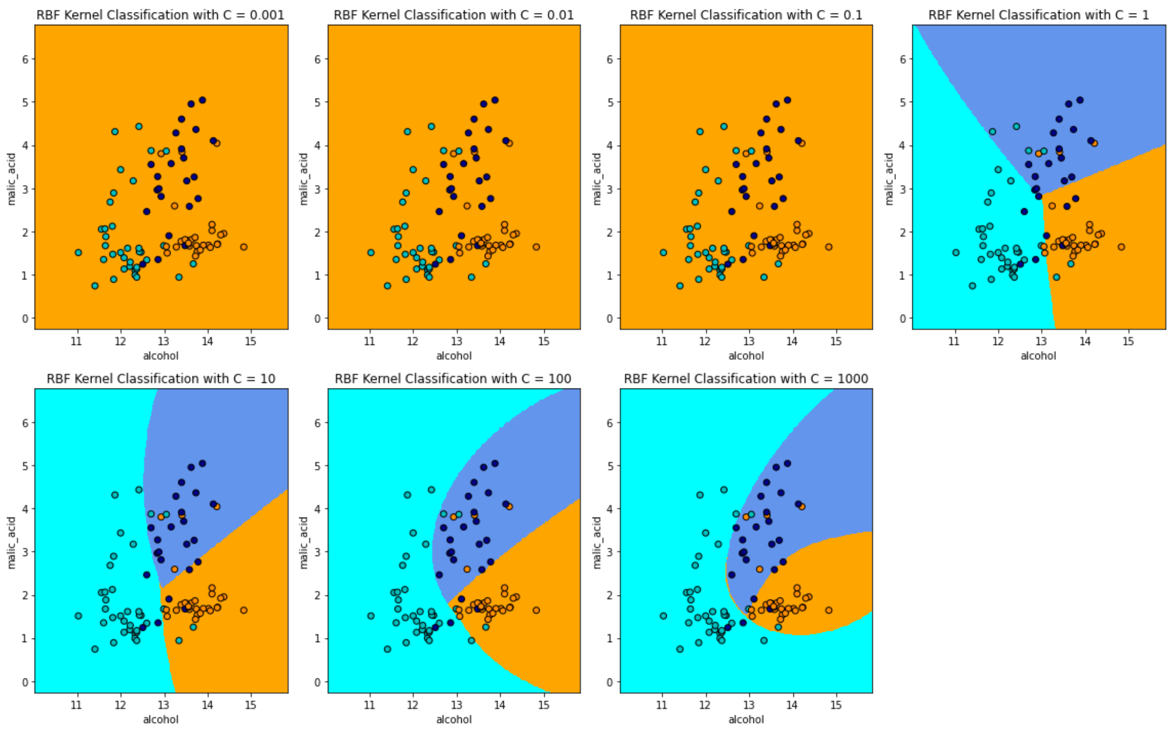
The last methodology is the SVM algorithm which exploits the Kernel *trick* (Radial Basis Function), which evolves the classical linear separator into a non-linear separator since each solution can be expressed as a function of scalar products. The objective is to separate each data point by projecting it into a higher dimensional space.

RBF Kernel relies on two main parameters: *C*, the cost of miss-classification (as in the classical SVM), and γ, which is a measure of dispersion of the decision region – i.e. how spread the region is across the dimensional space.
Analogously to the linear SVM, we performed a cross-parameter tuning - firstly on *C*, keeping γ constant to a default value determined as,

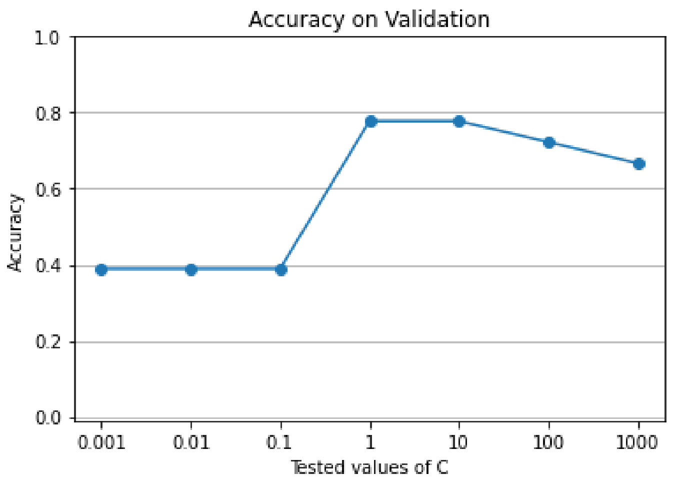
$$\gamma = \frac{1}{n\_features * X.var()}$$

and then by performing a grid search to find the optimal couple (C, γ) which maximize the model accuracy.

The chosen values of *C* are the same as in the linear SVM model, and the results are shown in the following collection of charts.



The key difference with respect to the first SVM model is that, as $C$ increases, the boundaries are no longer linear, indeed the shape is rather similar to a curve.



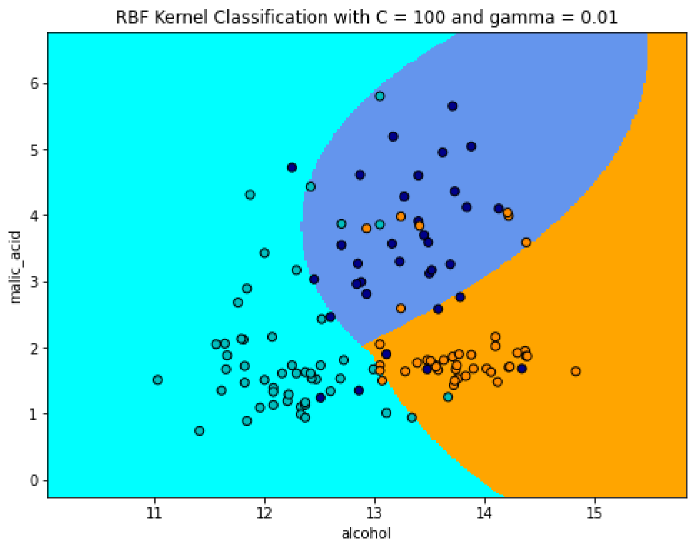
By optimizing exclusively on the cost of miss-classification (with constant $\gamma$), with $C = 1$ the highest accuracy of 78%, on the validation set, is reached. While, on the test set the accuracy is 74%.

We then performed the tuning on both parameters through a grid search of $C$ and $\gamma$. For both factors we identified a range of values, as it follows:

$$C = [\ 0.01\ ,\ 0.1\ ,\ 1\ ,\ 10\ ,\ 100\ ]$$

$$\gamma = [\ 0.01\ ,\ 0.1\ ,\ 1\ ,\ 10\ ,\ 100\ ]$$

As in the previous cases, we selected the couple ($C$, $\gamma$) which maximizes the accuracy on the validation set, to score the test set. The resulting accuracy is 80%. The following graph shows the visual representation of the decision boundaries derived using the best parameters (100, 0.01).



So far, we divided the dataset into three different sets (train, validation and test) which were fixed *ex-ante*. Due to the lack of data (only 178 observations), by removing a portion of instances it would lead to either an unavoidable reduction of the instances in the training set, impacting the model ability to correctly classify the objects, or in the test set, thus producing biased estimates of the model's performance.

To serve the purpose, the original dataset is no longer divided into three sets, but only in two, where the validation set is nested in the training set and no longer fixed, but it assumes a different configuration basing on the folding round. Indeed, the K-Fold validation technique allows to *fold* the validation set across the larger training set (while still keeping the overall proportion to 5:2:3). In this case, the number of folding rounds is five.

For each couple ($C$, $\gamma$), we executed the K-Fold algorithm, and computed the final accuracy as the average of the five accuracies obtained from each folding round. We selected the highest accuracy of 82% for the couple (10, 0.1). The accuracy on the test set using the abovementioned best hyperparameters is 80%.

As it turned out, the K-Fold algorithm provided with the best accuracy on the test set, compared to the other two methodologies (KNN and SVM), not necessarily because the former present a better model structure, but rather because by varying the underlying composition of the train and validation sets, it is not surprising the final results might vary across the board.

## 3 Other pairs of attributes

In conclusion, we selected a different pair of attributes from the 13 measurements of each wine. In particular, we chose the "color_intensity" and "hue" features, in order to classify the wines based on their colors.

We performed once again all the classification models introduced so far, and the main evidence is that the accuracy on the validation set is equal to approximately 85%, while it usually drops nearly 10 percentage points on the test set.
The only exception is for the K-Fold algorithm where the accuracy on the test set slightly improved compared to the validation one.