Antonio Norelli

Machine learning *a.a 2016/2017* *prof. Barbara Caputo*

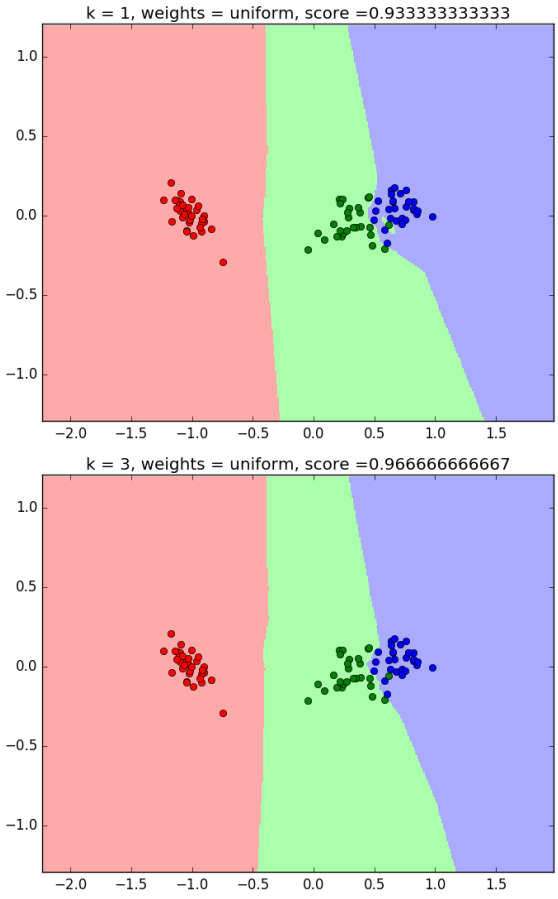
Homework 3: K-nearest neighbors

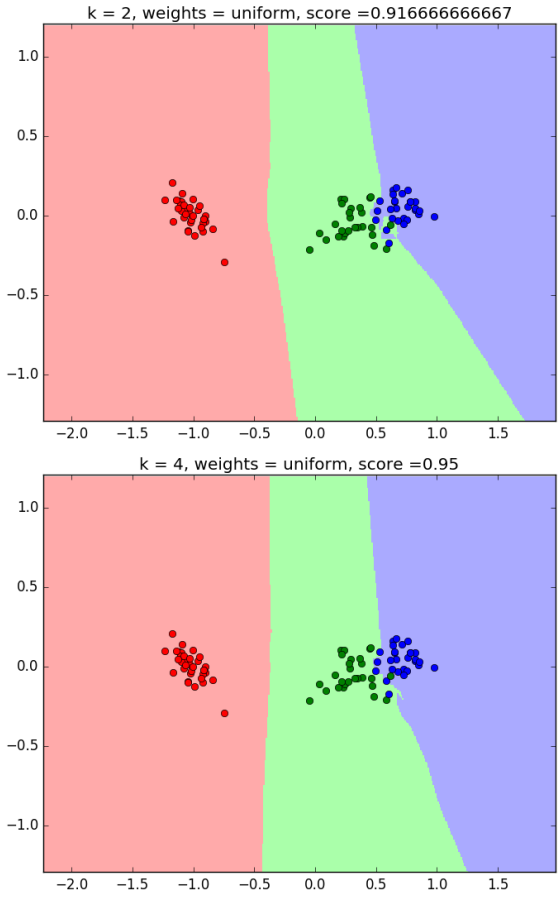
K-nearest neighbors (KNN)

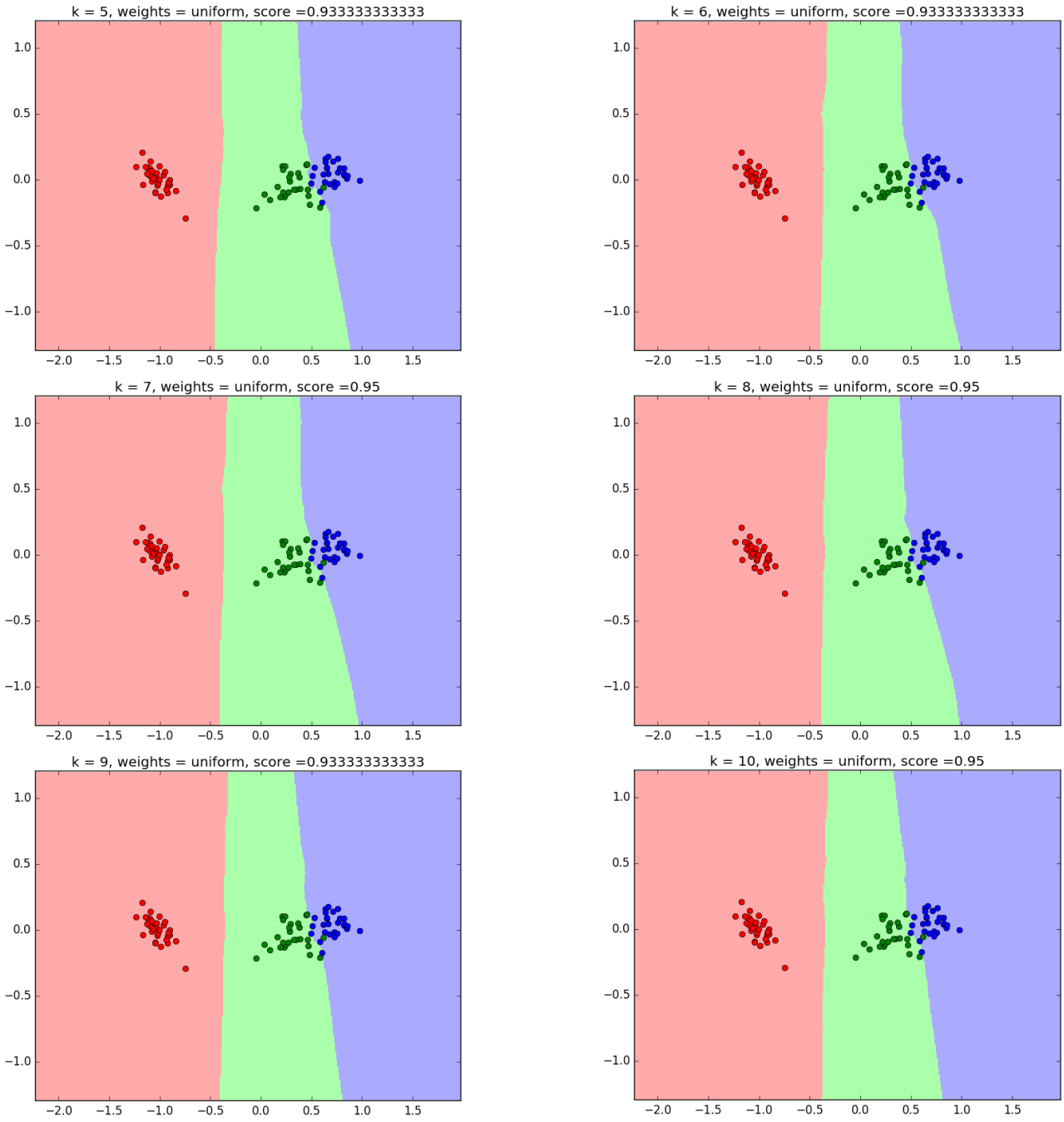
The dataset are the first two principal components of the Iris dataset available in the sklearn's standard dataset library.

We have three classes; the data is split into train and test set in proportion 3:2.

We start using the standard uniform weight function, we want to see the results varying the number of k from 1 to 10. In each plot is reported the KNN classification (decision boundaries), the true classification (colour of the points) and the accuracy score in the title.





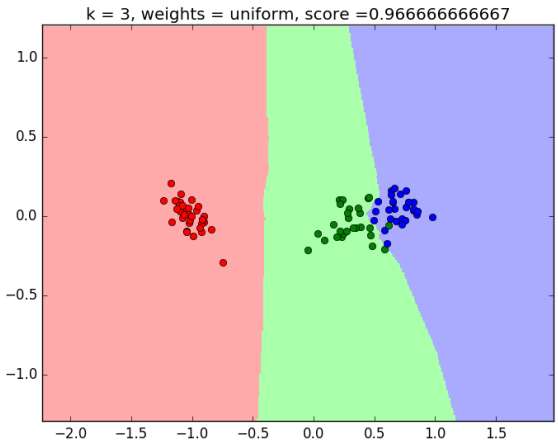


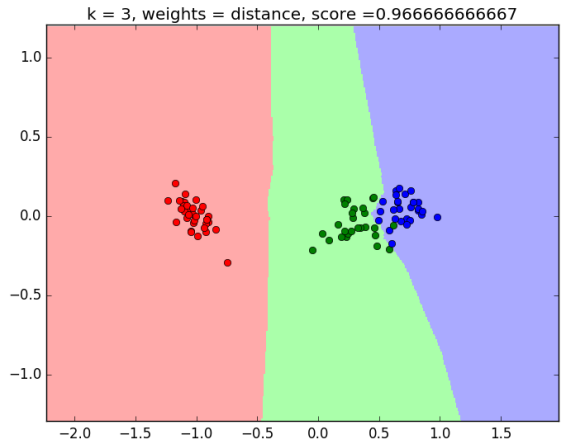
How do the boundaries change? Why?

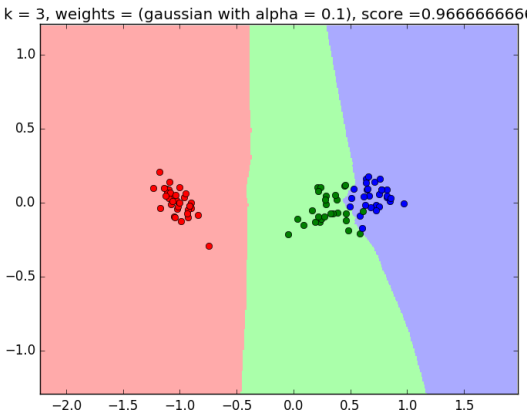
Increasing the number of neighbours considered, the boundaries become smoother because of the individual statistical fluctuation is compensated by the other neighbours. It follows that the higher the number of neighbours considered the lower the probability of overfitting.

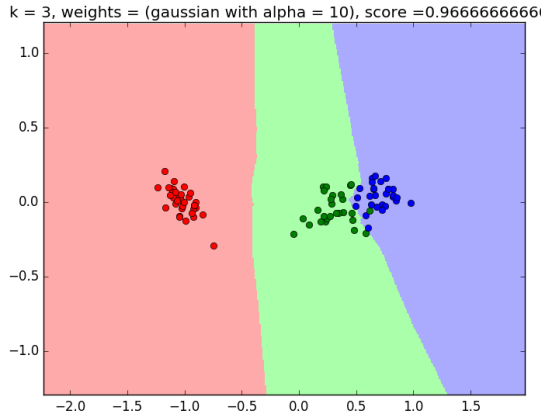
Different weight functions

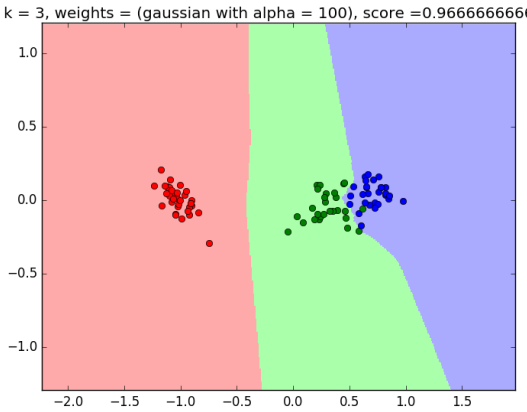
Now the number of neighbours is set at 3 while the weight function varies. The weight function used is reported in the title of each plot.

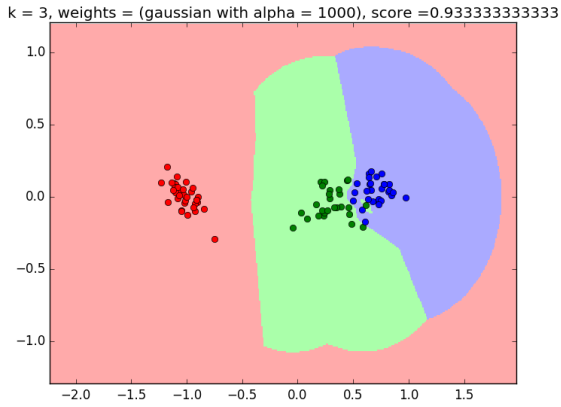










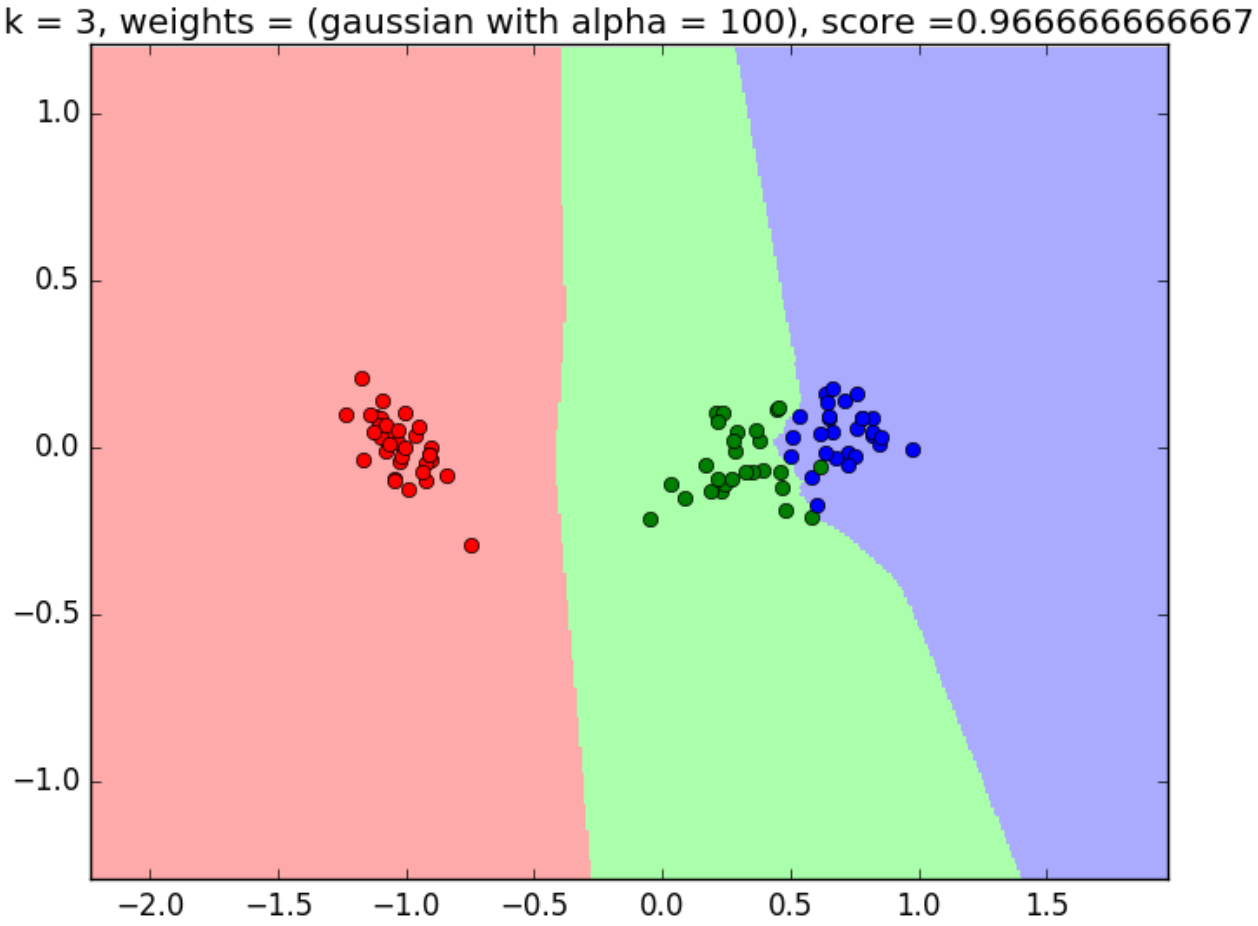


How do the boundaries change? Why?

Now the main difference is in the shape of the boundaries instead of the smoothness. A different weight function means a different importance of the neighbours considered. In the last plot we see a strange behaviour, the red areas in the right should not be there; I guess it is simply an error of approximation due to the sklearn implementation, the neighbours of those points have a weight $w = e^{-\alpha d^2} \cong e^{-1000*1^2} < 10^{-300} \sim 0$, so each class results zero and the points are arbitrarily assigned at the red one.

Grid search

k\weight functions	uniform	distance	gauss. α=0.1	gauss. α=10	gauss. α=100	gauss. α=1000
1	0.93	0.93	0.93	0.93	0.93	0.93
2	0.92	0.93	0.93	0.93	0.93	0.93
3	0.97	0.97	0.97	0.97	0.97	0.93
4	0.95	0.97	0.97	0.97	0.97	0.93
5	0.93	0.95	0.93	0.93	0.95	0.93
6	0.93	0.95	0.95	0.95	0.95	0.93
7	0.95	0.95	0.95	0.95	0.95	0.93
8	0.95	0.95	0.95	0.95	0.95	0.93
9	0.93	0.95	0.93	0.93	0.95	0.93
10	0.93	0.93	0.93	0.93	0.93	0.93



We have found the same best score for the majority of the instances with k=3 nearest neighbours. We choice the solution with a Gaussian (alpha=100) weight function because it is the best also in the other cases. Borders looks good, they are smooth and without isolate islands, that in this case could be a symptom of overfitting.