CS6140 Machine Learning

HW7 - Local Methods and Clustering

Make sure you check the syllabus for the due date.

PROBLEM 1 [50 points] k-Nearest Neighbors (kNN)

Implement the kNN classification algorithm such that it can work with different distance functions (or kernels as similarities) and with different values of ‘k’= "the number of closest neighbors used", i.e. k=1,k=3, and k=7.

A) Spambase dataset. Try k=1,3,7 with Euclidian distance.

B) Digits Dataset, with extracted features. Try k=1,3,7 and the following similarities/distances:

- Cosine distance
- Gaussian Kernel
- Plynomial degree-2 Kernel

PROBLEM 2 [50 points] Neighbors in a range/window

Instead of using the closest k Neighbors, now we are going to use all neighbors within a window. It should be clear that a testpoint z_1 might have many neighbors (a dense area) in its window, while another testpoint z_2 might have only (a sparse area) or none at all.

A) Fixed Window. Fix an appropriate window size around the test datapoint by defining a windows "radius" R , or the maximum distance allowed. Predict the label as the majority or average of training neighbors within the window.

- Run on Spambase dataset with Euclidian distance.
- Run on Digits dataset with cosine distance.

B) Kernel density estimation. Separately for each label class, estimate P(z|C) using the density estimator given by the kernel K restricted to the training points form class C. There is no need for physical windows, as the kernel is weighting each point by similarity:

$$P(z|C) = \frac{1}{m_C} \sum_{y_i=C} k(z,x)$$

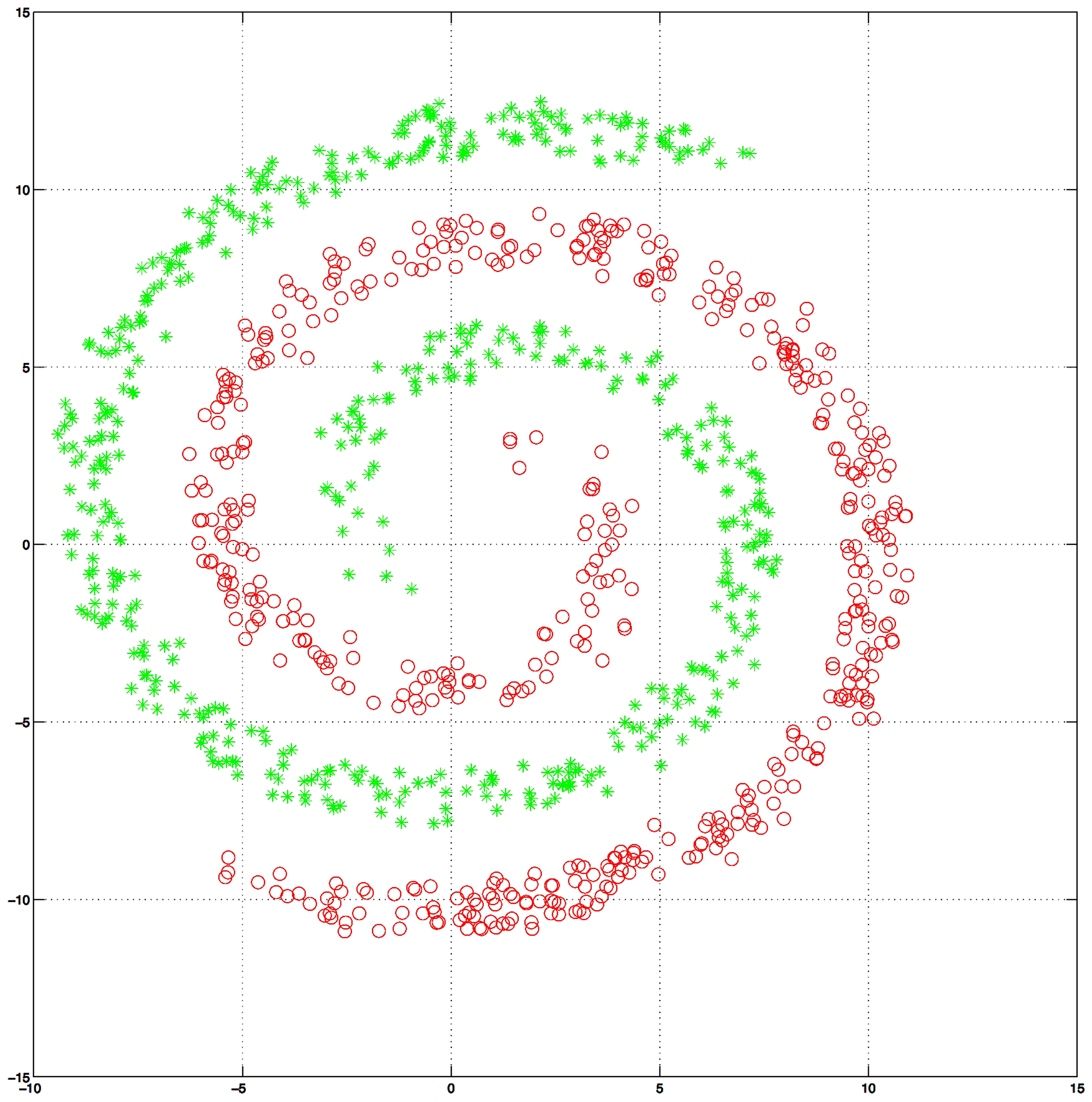
where m_C is the number of points in the training set with label C. Then predict the class by largest Bayesian probabilities P(C|z) = P(C) * P(z|C) /P(z).

- Run on Spambase dataset with Gaussian kernel.
- Run on Digits dataset with Gaussian kernel
- Run on Digits dataset with polynomial kernel.

PROBLEM 3 [50 points] Dual Perceptron with kernels

A) Implement the dual version of the perceptron, keeping in mind that you have to make it work with kernels. Apply the dual perceptron to the artificial dataset we used on HW2, in order to make sure your dual is equivalent to the primal (same solution).
Note: in our implementation of the dual perceptron, we did not pre-process convert all X to positive labels, as we did with the primal perceptron. Such conversion would imply that the kernel implicit mapping satisfies -ϕ(x) = ϕ(-x) which is not necessarily true.

B) We have created another artificial binary dataset "two spirals", this time linearly highly-nonseparable on purpose. Try to run the dual perceptron (with dot product) and conclude that the perceptron does not work. Then run the dual perceptron with the Gaussian kernel and conclude the data is now separable.
Expected accuracy dot product kernel : 50% average oscillating between 66% and 33%
Expected accuracy with RBF kernel : 99.9% (depends on sigma)



PROBLEM 4 [Extra Credit] Dual Perceptron from Representer Theorem

A) What is the perceptron primal loss? Write the primal perceptron as an optimization problem.

B) Apply the Representer theorem and substitute w as a linear combination of the training set in the objective, to obtain the dual form

PROBLEM 5 [Extra Credit] Feature Selection with kNN

Implement the RELIEF algorithm for feature selection: independently assess the quality W of each feature by adjusting the weight with each instance. For feature j, the update on point x is

$$W^j_{new} = W^j_{old} - (x^j - z^j_{same})^2 + (x^j - z^j_{opp})^2$$

where z_{same} is the closest-to-x from the same class as x, and z_{opp} is the closest-to-x from the opposite class to class of x.
Run on Spambase dataset (training) and select the top 5 features by W. Then run the algorithm from PB1

again only with selected features.

PROBLEM 6 [Extra Credit] Dual Regression via Representer Theorem

A) What is the loss function that linear regression minimizes? (Hint: answer is in the notes)

B) If we use regularized linear regression, how can we write the final weight vector as Equation-2. What are the alphas in this case, and is linear regression kernelizable?

C) Compare your developed loss function from part (b) with the Hinge and Log loss. Can we consider linear regression as a max-margin learning algorithm? (Hint: plot the different loss functions, and see what happens when we try to maximize the margin for a single input data instance)