Machine Learning
Exercise 10

Marc Toussaint
TAs: Janik Hager, Philipp Kratzer
Machine Learning & Robotics lab, U Stuttgart
Universitätsstraße 38, 70569 Stuttgart, Germany

June 26, 2019

(DS BSc students please try to complete the full exercise this time.)

1 Method comparison: kNN regression versus Neural Networks (5 Points)

k-nearest neighbor regression is a very simple lazy learning method: Given a data set $D = \{(x_i, y_i)\}_{i=1}^n$ and query point x^*, first find the k nearest neighbors $K \subset \{1, .., n\}$. In the simplest case, the output $y = \frac{1}{K}\sum_{k \in K} y_k$ is then the average of these k nearest neighbors. In the classification case, the output is the majority vote of the neighbors.

(To make this smoother, one can weigh each nearest neighbor based on the distance $|x^* - x_k|$, and use local linear or polynomial (logistic) regression. But this is not required here.)

On the webpage there is a data set `data2ClassHastie.txt`. Your task is to compare the performance of kNN classification (with basic kNN majority voting) with a neural network classifier. (If you prefer, you can compare kNN against another classifier such as logistic regression with RBF features, instead of neural networks. The class boundaries are non-linear in x.)

As part of this exercise, discuss how a fair and rigorous comparison between two ML methods is done.

2 Gradient Boosting for classification (5 Points)

Consider the following *weak learner* for classification: Given a data set $D = \{(x_i, y_i)\}_{i=1}^n, y_i \in \{-1, +1\}$, the weak learner picks a single i^* and defines the discriminative function

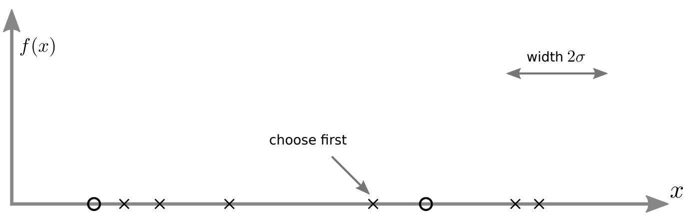
$$f(x) = \alpha e^{-(x - x_{i^*})^2 / 2\sigma^2} \ ,$$

with fixed width σ and variable parameter α. Therefore, this weak learner is parameterized only by i^* and $\alpha \in \mathbb{R}$, which are chosen to minimize the neg-log-likelihood

$$L^{\text{nll}}(f) = -\sum_{i=1}^n \log \sigma(y_i f(x_i)) \ .$$

a) Write down an explicit pseudo code for gradient boosting with this weak learner. By "pseudo code" I mean explicit equations for every step that can directly be implemented. This needs to be specific for this particular learner and loss. (3 P)

b) Here is a 1D data set, where $\circ$ are 0-class, and $\times$ 1-class data points. "Simulate" the algorithm graphically on paper. (2 P)



Extra) If we would replace the neg-log-likelihood by a hinge loss, what would be the relation to SVMs?