Machine Learning:
Assignment Sheet #4

Due on March 8, 2022 at 10:00

Group HB
Henri Sota, Enis Mustafaj

Problem 4.1

In this task, we revise Examples 3.2 and 3.3 from the lecture notes for a different setup:
Let $X : \Omega \to \mathbb{R}$ be an input variable and $Y : \Omega \to \mathbb{R}$ be an output variable. For the input variable we assume that it follows the uniform distribution as $X \sim \mathcal{U}[-1, 1]$. (Note the other range!) Moreover, we make the very strong assumption to know the "true" dependency between X and Y. Specifically we define Y via

$$Y := g(X), \quad \text{with} \quad g(x) = x^4$$

Now we look for a function f that shall approximate that (usually unknown) relationship between X and Y. We claim that

$$f(x) = x^3$$

is a good approximation to $Y = g(X)$.

a) Calculate the expected (squared) prediction error for f. Why is the error that large compared to the lecture example?

$$EPE(f) = E[L_2(y, f(x)] = \int_R \int_R ((y - f(x))^2 p(x, y)\, dxdy$$

Based on Knowledge 3.1, it is possible to substitute the joint density with the product of the marginal density of X and a Dirac Delta function, and then use the Dirac Delta function to integrate with regards to y.

$$\begin{aligned} EPE(f) &= \int_R \int_R (y - x^3)^2 p_X(x) \delta(y - x^4)\, dxdy \\ &= \int_R \int_R (y - x^3)^2 p_X(x) \delta(y - x^4)\, dydx \\ &= \int_R p_X(x) \int_R (y - x^3)^2 \delta(y - x^4)\, dydx \\ &= \int_R p_X(x)(x^4 - x^3)^2\, dx \end{aligned}$$

Since we know that X is uniformly distributed in the range $[-1, 1]$, the marginal density will be equal to 0.5 in that interval and 0 everywhere else.

$$\begin{aligned} EPE(f) &= \frac{1}{2} \int_{-1}^{1} (x^4 - x^3)^2\, dx \\ &= \frac{1}{2} \int_{-1}^{1} (x^8 - 2x^7 + x^6)\, dx \\ &= \frac{1}{2} \left[\frac{x^9}{9} - \frac{x^8}{4} + \frac{x^7}{7} \right]_{-1}^{1} \\ &= \frac{1}{2} \left[\left(\frac{1}{9} - \frac{1}{4} + \frac{1}{7} \right) - \left(-\frac{1}{9} - \frac{1}{4} - \frac{1}{7} \right) \right] \\ &= \frac{1}{2} \left(\frac{2}{9} + \frac{2}{7} \right) \\ &= \frac{16}{63} \\ &\approx 0.254 \end{aligned}$$

The error is large compared to the lecture example due to the extended range $[-1, 1]$ and due to the difference between an odd (x^3) and even function (x^4) on the negative domain.

b) Find the regressor for $Y = g(X)$.

Using Knowledge 3.1, we can retrieve the dependency of Y on X by using the conditional density of Y given X.

$$p(y|x) = \frac{p(x,y)}{p_X(x)} = \frac{\delta(y - x^4)p_X(x)}{p_X(x)} = \delta(y - x^4)$$

Evaluating the regressor using the Dirac Delta function to integrate with regards to y:

$$\begin{aligned} E[Y|X] &= \int_R yp(y|x)\, dy \\ &= \int_R y\delta(y - x^4)\, dy \\ &= y \\ &= x^4 \end{aligned}$$

Problem 4.2

In the lecture slides, you have seen that the expected (squared) prediction error $EPE(f)$ is given by

$$EPE(f) = E(L_2(Y, f(X))$$

Theorem 3.1 then states that the function f that minimizes the expected (squared) prediction error $EPE(f)$ is given by

$$f(x) = E(Y|X = x)$$

Note that in the middle of the proof, we encounter an equation

$$EPE(f) = E_X E_{Y|X}[(f(X) - E[Y|X])^2 + 2(f(X) - E[Y|X])(E[Y|X] - Y) + (E[Y|X] - Y)^2|X]$$

where the second term can be dropped. Now, prove that this is possible, i.e. that it holds

$$2E_X E_{Y|X}[(f(X) - E[Y|X])(E[Y|X] - Y)|X] = 0$$

From the above statement we can derive that:

$$E_X E_{Y|X}[(f(X) - E[Y|X])(E[Y|X] - Y)|X] = 0$$

By expanding the statement inside the first expected value we get:

$$E_X E_{Y|X}[E[Y|X] \cdot f(x) - Y \cdot f(x) - E[Y|X]^2 + Y \cdot E[Y|X]|X] = 0$$

In order for the equation to be 0, the term:

$$E_{Y|X}[E[Y|X] \cdot f(x) - Y \cdot f(x) - E[Y|X]^2 + Y \cdot E[Y|X]|X]$$

needs to be 0. Now we expand the expected value with regard to the terms inside:

$$E[[E[Y|X] \cdot f(x)|X] - E[Y \cdot f(x)|X] - E[E[Y|X]^2|X] + E[Y \cdot E[Y|X]|X] = 0$$

By using the property of the expected value that if 2 events A and B are independent of each other, then:

$$E[A \cdot B] = E[A] \cdot E[B]$$

we get:

$$E[[E[Y|X]|X] \cdot E[f(x)|X] - E[Y|X] \cdot E[f(x)|X] - E[E[Y|X]|X] \cdot E[E[Y|X]|X] + E[Y|X] \cdot E[E[Y|X]|X] = 0$$

By simplifying:

$$E[E[Y|X]|X] = E[Y|X]$$

we get:

$$E[Y|X] \cdot E[f(x)|X] - E[Y|X] \cdot E[f(x)|X] - E[Y|X] \cdot E[Y|X] + E[Y|X] \cdot E[Y|X] = 0$$

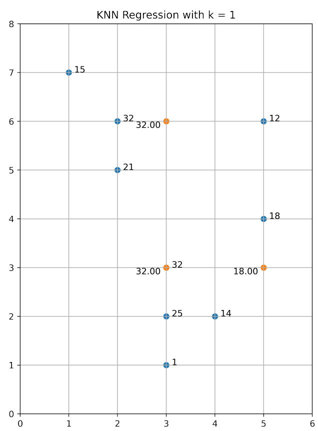
From the equation we can see that the terms cancel each other and the result will be 0.

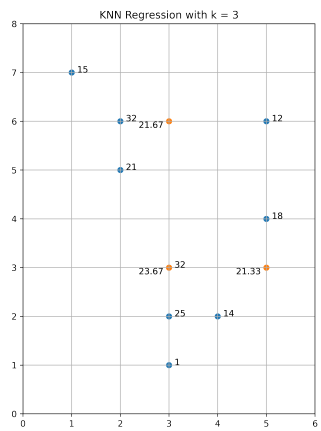
Problem 4.3

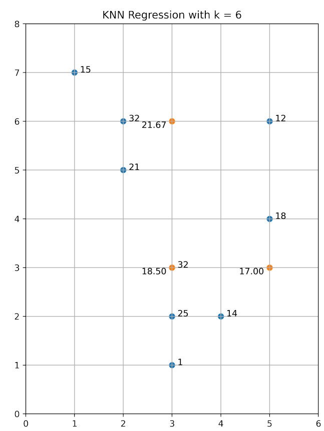
You are given the following training data:

$$\mathcal{T} = \{((1,7)^\top, 15), ((2,5)^\top, 21), ((2,6)^\top, 32), ((3,3)^\top, 32), ((3,2)^\top, 25), ((3,1)^\top, 1), ((4,2)^\top, 14), ((5,4)^\top, 18), ((5,6)^\top, 12)\}$$

Manually carry a kNN regression prediction for $\mathbf{x}_1 = (3,3)^\top$, $\mathbf{x}_2 = (3,6)^\top$, $\mathbf{x}_3 = (5,3)^\top$, and $k = 1, k = 3, k = 6$. As part of the task, you have to draw the points in a scatter plot (on paper) and mark the respective neighborhoods that contribute to the final result.







- k = 1
  - The nearest neighbour of the point $x_1 = (3,3)^T$ is the point $(3,3)^T$ itself so the result of KNN regression will be 32
  - The nearest neighbour of the point $x_2 = (3,6)^T$ is the point $(2,6)^T$. The result of KNN regression will be 32
  - The nearest neighbour of the point $x_3 = (5,3)^T$ is the point $(5,4)^T$. The result of KNN regression will be 18
- k = 3
  - The 3 nearest neighbours of the point $x_1 = (3,3)^T$ are the points $(3,3)^T, (3,2)^T, (4,2)^T$. The the result of KNN regression will be $\frac{32+25+14}{3} = 23.67$
  - The 3 nearest neighbours of the point $x_2 = (3,6)^T$ are the points $(2,5)^T, (2,6)^T, (5,6)^T$. The the result of KNN regression will be $\frac{32+21+12}{3} = 21.67$
  - The 3 nearest neighbours of the point $x_3 = (5,3)^T$ are the points $(5,6)^T, (4,2)^T, (3,3)^T$. The the result of KNN regression will be $\frac{32+18+14}{3} = 21.33$

- k = 6
  - The 6 nearest neighbour of the point $x_1 = (3,3)^T$ are the points $(3,3)^T, (3,2)^T, (4,2)^T, (3,1)^T$, $(5,4)^T, (2,5)^T$ itself so the result of KNN regression will be $\frac{32+25+1+18+14+21}{6} = 18.5$
  - The 6 nearest neighbour of the point $x_2 = (3,6)^T$ are the points $(3,3)^T, (2,6)^T, (2,5)^T, (5,6)^T$, $(1,7)^T, (5,4)^T$ itself so the result of KNN regression will be $\frac{32+18+12+32+21+15}{6} = 21.67$
  - The 6 nearest neighbour of the point $x_1 = (5,3)^T$ are the points $(3,3)^T, (3,2)^T, (3,1)^T, (4,2)^T$, $(5,4)^T, (5,6)^T$ itself so the result of KNN regression will be $\frac{32+25+1+14+18+12}{6} = 17$

Programming Problem 4.1

Consider the Examples 3.4 and 3.5 from the lecture, for which you also have access to the source code. Complete the following tasks:

a) (Re-)implement Example 3.4. This time, however, you need to implement the kNN regression by yourself, without a machine learning library and without a kNN search library. (If you implement in Python, just start from the available Jupyter notebook) Verify the correctness of your implementation by cross-checking it with Example 3.4.

b) Apply your implementation to the Energy efficiency Data Set from the UCI Machine Learning Repository. Build the predictor for the required heating load on the full data set and predict the load on the first three samples of the data set and $k = 1$, $k = 3$, $k = 10$.

c) (Re-)implement Example 3.5. This time, however you need to implement the kNN classification by yourself, without a machine learning library and without a kNN search library. (If you implement in Python, just start from the available Jupyter notebook) Verify the correctness of your implementation by cross-checking it with Example 3.5.

d) Now, we would like to use kNN classification for SPAM classification. Either you collect some e-mails (both SPAM and no SPAM) from your Inbox, label them manually and create the representation using your implementation from last week or you use the Spambase Data Set from the UCI Machine Learning Repository. Apply your implementation to one of these data sets and evaluate the predictor for three random samples in the data set and values $k = 1$, $k = 3$, $k = 10$. Compare these results to the training data.

The implementation of the kNN regressor and classifier can be found in the file `programming_exercises.ipynb`. Both have been implemented as classes that can be instantiated given the `n_neighbors` argument which is the number of neighbors used when searching for the neighbors of data point. The classes offer two methods, `fit` and `predict`, which add the data to the model and predict the expected regression value or class respectively.