Machine Learning Worksheet 2

Decision Trees and k-Nearest Neighbors

1 Dataset

You are free to do the following exercises completely by hand or use a computer to help speed things up (python, MATLAB, R, Excel, …). You should, however, show the basic steps of your work and implement your own "helpers" instead of blindly using code. Using a machine learning toolbox and copying the result will not help you understand.

The table below gives you a feature matrix $\boldsymbol{X}$ together with the output z_i for every row i of the feature matrix. This data is also available in Piazza as `02_homework_dataset.csv`. The column with names for each datapoint i can help you reference specific data points.

i	$x_{i,1}$	$x_{i,2}$	$x_{i,3}$	z_i
A	5.5	0.5	4.5	2
B	7.4	1.1	3.6	0
C	5.9	0.2	3.4	2
D	9.9	0.1	0.8	0
E	6.9	-0.1	0.6	2
F	6.8	-0.3	5.1	2
G	4.1	0.3	5.1	1
H	1.3	-0.2	1.8	1
I	4.5	0.4	2.0	0
J	0.5	0.0	2.3	1
K	5.9	-0.1	4.4	0
L	9.3	-0.2	3.2	0
M	1.0	0.1	2.8	1
N	0.4	0.1	4.3	1
O	2.7	-0.5	4.2	1

2 Decision Trees

Problem 1: Build a decision tree T for your data $\boldsymbol{X}$. Consider all possible feature tests and use the Gini index to build your tree. Build the tree only to a depth of two! Provide at least the value of the final Gini index at each node and the distribution of classes at each leaf.

Let $T^1, T^2, \ldots$ denote the "versions" of the whole tree generated by iteratively adding nodes and T_i denote the subtree of T that has node i as root node. Furthermore, let $i(T_i)$ be the Gini index of the distribution $\boldsymbol{n_i}$ at a node i. We write $\text{Gini}(n_0, n_1, n_2)$ as a shorthand for individual class occurrences.

$$i(T_i) = \text{Gini}(\boldsymbol{n}_{\boldsymbol{i}0}, \boldsymbol{n}_{\boldsymbol{i}1}, \boldsymbol{n}_{\boldsymbol{i}2}) = 1 - \left(\frac{\boldsymbol{n}_{\boldsymbol{i}0}}{\boldsymbol{n}_{\boldsymbol{i}0}+\boldsymbol{n}_{\boldsymbol{i}1}+\boldsymbol{n}_{\boldsymbol{i}2}}\right)^2 - \left(\frac{\boldsymbol{n}_{\boldsymbol{i}1}}{\boldsymbol{n}_{\boldsymbol{i}0}+\boldsymbol{n}_{\boldsymbol{i}1}+\boldsymbol{n}_{\boldsymbol{i}2}}\right)^2 - \left(\frac{\boldsymbol{n}_{\boldsymbol{i}2}}{\boldsymbol{n}_{\boldsymbol{i}0}+\boldsymbol{n}_{\boldsymbol{i}1}+\boldsymbol{n}_{\boldsymbol{i}2}}\right)^2$$

For each feature, we use the values that occur in the data, but exclude the maximum as a potential splitting value e.g. x_1 can be split on $0.4, 0.5, \dots, 7.4, 9.3$.

Before splitting: $\boldsymbol{n}_0 = (5, 6, 4)$ $\quad i(T_0) = \text{Gini}(5, 6, 4) \approx .658$

After splitting at $x_1 \leq 4.1$:

Left $\quad \boldsymbol{n}_L = (0, 6, 0) \quad i(T_L) = \text{Gini}(0, 6, 0) \approx .000 \quad p_L = \frac{6}{15}$

Right $\quad \boldsymbol{n}_R = (5, 0, 4) \quad i(T_R) = \text{Gini}(5, 0, 4) \approx .494 \quad p_R = \frac{9}{15}$

$$\Rightarrow \Delta i(x_1 \leq 4.1, T^1) = i(T_0) - p_L i(T_L) - p_R i(T_R) \approx .296$$

Node L is pure. $\quad i(T_L) = .00$

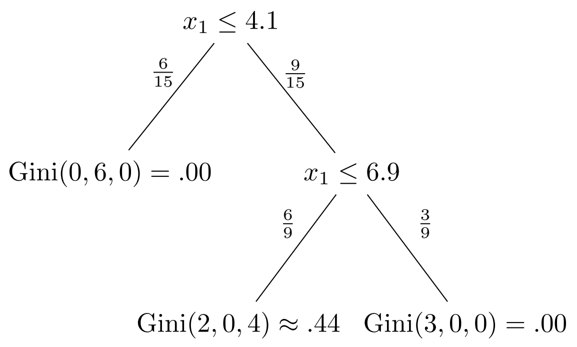
Split node R at $x_1 \leq 6.9$:

Left $\quad \boldsymbol{n}_{RL} = (2, 0, 4) \quad i(T_{RL}) = \text{Gini}(2, 0, 4) \approx .444 \quad p_{RL} = \frac{6}{9}$

Right $\quad \boldsymbol{n}_{RR} = (3, 0, 0) \quad i(T_{RR}) = \text{Gini}(3, 0, 0) = .000 \quad p_{RR} = \frac{3}{9}$

$$\Rightarrow \Delta i(x_1 \leq 6.9, T^2) = i(T_R) - p_{RL} i(T_{RL}) - p_{RR} i(T_{RR}) \approx .296$$

This leads to the final tree T^3 or T:


$x_1 \leq 4.1$
$\frac{6}{15}$
$\frac{9}{15}$
$\text{Gini}(0,6,0) = .00$
$x_1 \leq 6.9$
$\frac{6}{9}$
$\frac{3}{9}$
$\text{Gini}(2,0,4) \approx .44$
$\text{Gini}(3,0,0) = .00$

Problem 2: Use the final tree T from the previous problem to classify the vectors $\boldsymbol{x}_a = (4.1, -0.1, 2.2)^T$ and $\boldsymbol{x}_b = (6.1, 0.4, 1.3)^T$. Provide both your classification y_a and y_b and their respective probabilities $p(c = y_a \mid \boldsymbol{x}_a, T)$ and $p(c = y_b \mid \boldsymbol{x}_b, T)$

To classify $\boldsymbol{x}_a$ we use the path $0 - R$ to reach leaf R, similarly we reach leaf node LR by following $0 - L - LR$ for $\boldsymbol{x}_b$.

$$p(c = 1 \mid \boldsymbol{x}_a, T) = 1 \Rightarrow y_a = 1$$

$$p(c = 2 \mid \boldsymbol{x}_b, T) = \frac{2}{3} \Rightarrow y_b = 2$$

3 k-Nearest Neighbor

Problem 3: Load the notebook `02_homework_kNN.ipynb` from piazza. Fill in the missing code and run the notebook. Convert the evaluated notebook to pdf and add it to the printout of your homework.

Note: For information on IPython notebooks and how to convert them to other formats, consult the Jupyter documentation and nbconvert documentation.

See the solution-notebook `02_homework_kNN_solution.ipynb`.

Problem 4: Classify the two vectors $\boldsymbol{x}_a$ and $\boldsymbol{x}_b$ given in Problem 2 with the k-nearest neighbors algorithm. Use $k = 3$ and Euclidean distance.

Let $\boldsymbol{x}_A \dots \boldsymbol{x}_O$ denote the training sample points in the order given. (The index is given in the table as well.)

We compute the distances of all training points to the given vectors with

$$d(\boldsymbol{u}, \boldsymbol{v}) = \textstyle\sum_i |u_i - v_i|$$

$$\text{Nearest neighbors of } \boldsymbol{x}_a = (4.1, -0.1, 2.2)^T: \quad \begin{array}{ccc} \text{I} & 0.671 & 0 \\ \text{C} & 2.184 & 2 \\ \text{O} & 2.474 & 1 \\ \text{A} & 2.759 & 2 \\ \dots & & \end{array} \quad \Rightarrow \mathcal{N}_3(\boldsymbol{x}_a) = \{C, I, O\}$$

$\mathcal{N}_3(\boldsymbol{x}_a)$ produces a tie.
Tie-breaking: randomly assign a class, use the next neighbor as a tie-breaker, ... With the latter: $y_a = 2$.
Note: In tasks where you prefer not to make a prediction if the certainty of your prediction is low, you could also return “unassigned” if $p(y \mid \boldsymbol{x}, \mathcal{M}) < t$ (the probability of your prediction given your model $\mathcal{M}$ is below a certain threshold.)

$$\text{Nearest neighbors of } \boldsymbol{x}_b = (6.1, 0.4, 1.3)^T: \quad \begin{array}{ccc} \text{E} & 1.175 & 2 \\ \text{I} & 1.746 & 0 \\ \text{C} & 2.119 & 2 \\ \text{B} & 2.733 & 0 \\ \dots & & \end{array} \quad \Rightarrow \mathcal{N}_3(\boldsymbol{x}_b) = \{C, E, I\}$$

$$y_b = 2,\ p(y_b \mid \boldsymbol{X}, k = 3) = \frac{2}{3}$$

Problem 5: Now, consider z_i to be real-valued labels rather than classes. Perform 3-NN regression to label the vectors from Problem 2.

$$y_a = \frac{1}{\mathcal{C}} \sum_{i \in \mathcal{N}_3(\boldsymbol{x}_a)} \frac{z_i}{\mathrm{d}(\boldsymbol{x}_a, \boldsymbol{x}_i)} = \frac{1}{\frac{1}{0.671} + \frac{1}{2.184} + \frac{1}{2.474}} \times \left(\frac{0}{0.671} + \frac{2}{2.184} + \frac{1}{2.474} \right) \approx 0.561$$

$$y_b \approx 1.396$$

Problem 6: Look at the data. Which problem do you see w.r.t. building a Euclidean distance-based k-NN model on $\boldsymbol{X}$? How can you compensate for this problem? Does this problem also arise when training a decision tree?

The standard deviations of features are $\sigma_1 \approx 2.99, \sigma_2 \approx .37, \sigma_3 \approx 1.41$. The scale of the second feature is very low and has little influence on the overall result. When using k-NN, useful class-specific information in this feature will be lost.
This can be compensated by rescaling all features with their respective scale either by standardizing the features or using Mahalanobis distance.
Decision trees are scale-invariant.