**EE2211 Tutorial 9**

(Gini impurity, entropy and misclassification rate)

**Question 1:**

Compute the Gini impurity, entropy, misclassification rate for nodes A, B and C, as well as the overall metrics (Gini impurity, entropy misclassification error) at depth 1 of the decision tree shown below.

**A picture containing object, clock

Description automatically generated**

**Answer:**

Let’s assume class 1, class 2 and class 3 correspond to red triangles, orange squares and blue circles respectively.

* For node A, ,
* For node B, ,
* For node C, ,

For **Gini impurity**, recall formula is

* Node A:
* Node B:
* Node C:
* Overall Gini at depth 1:

Observe the decrease in Gini impurity from root () to depth 1 ()

For **entropy**, recall formula is

* Node A:
* Node B:
* Node C:
* Overall entropy at depth 1:

Observe the decrease in entropy from root () to depth 1 ()

For **misclassification rate**, recall formula is

* Node A: max() = 1 =
* Node B: max() = 1 =
* Node C: max() = 1 =
* Overall misclassification error rate at depth 1:
* We can also double check that at depth 1, the 4 red triangles will be classified wrongly for node B and the 1 red triangle + 2 blue circles will be classified wrongly for node C. So in total, there will be 7 wrong classifications out of 18 datapoints, which corresponds to = 0.3889
* Observe the decrease in misclassification rate from root () to depth 1 ()

(MSE of regression trees)

**Question 2:**

Calculate the overall MSE for the following data at depth 1 of a regression tree assuming a decision threshold is taken at . How does it compare with the MSE at the root?

{,}: {1, 2}, {0.8, 3}, {2, 2.5}, {2.5, 1}, {3, 2.3}, {4, 2.8}, {4.2, 1.5}, {6, 2.6}, {6.3, 3.5}, {7, 4}, {8, 3.5}, {8.2, 5}, {9, 4.5}

**Answer:**

At depth 1, when

* MSE =

At depth 1, when

* MSE =

Overall MSE at depth 1:

At the root:

* MSE =

Therefore, MSE has decreased from at the root to at depth 1

(Regression tree, Python)

**Question 3:**

Import the California Housing dataset “from sklearn.datasets import fetch\_california\_housing” and “housing = fetch\_california\_housing()”. This data set contains 8 features and 1 target variable listed below. Use “MedInc” as the input feature and “MedHouseVal” as the target output. Fit a regression tree to depth 2 and compare your results with results generated by “from sklearn.tree import DecisionTreeRegressor” using the “squared error” criterion.

Target: ['MedHouseVal']

Features:['MedInc', 'HouseAge', 'AveRooms', 'AveBedrms', 'Population', 'AveOccup', 'Latitude', 'Longitude']

**Answer:**

Please refer to Tut9\_Q3\_zhou.py. We can exactly replicate the results from scikit-learn. Note that in the plot below, the blue dots are the training datapoints. The curves from scikit-learn (black line) and our own tree (red dashed line) are on top of each other, so they might be hard to tell apart.

A diagram of a step

Description automatically generated

(Classification tree, Python)

**Question 4:**

Get the data set “from sklearn.datasets import load\_iris”. Perform the following tasks.

1. Split the database into two sets: 80% of samples for training, and 20% of samples for testing using random\_state=0
2. Train a decision tree classifier (i.e., “tree.DecisionTreeClassifier” from sklearn) using the training set with a maximum depth of 4 based on the “entropy” criterion.
3. Compute the training and test accuracies. You can use accuracy\_score from sklearn.metrics for accuracy computation
4. Plot the tree using “tree.plot\_tree”.

**Answer:**

Please refer to Tut9\_Q4\_yeo.py.

Training accuracy: 0.9917

Test accuracy: 1.0

The resulting tree looks like this: