**MUSTAFA CAN ÇALIŞKAN**

**150200097**

**BLG454E HW2**

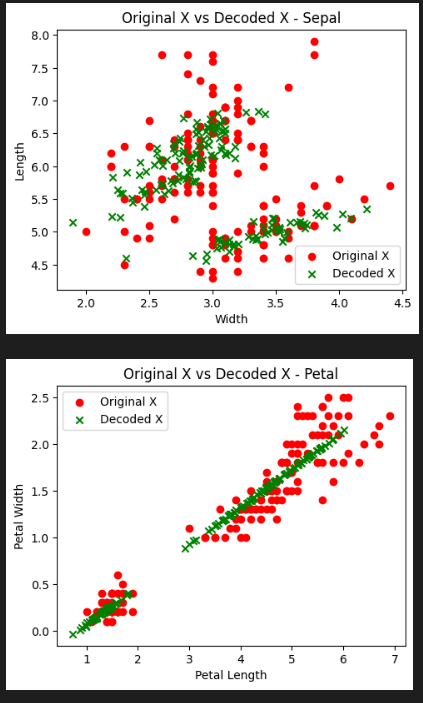
**1)**

The decision tree algorithm, being a greedy approach, tends to produce local optima rather than a global optimum. Consequently, resulting trees can grow to sizes that discourage practical use. To mitigate this inefficiency, several techniques have been developed, including pruning, impurity constraints, and limitations on sample counts within nodes. I have applied two such methods: restricting the number of samples within nodes and constraining the depth of the tree.

I've utilized entropy as the measure for impurity calculation. Yet, it's essential to note that in practical scenarios, the assumption of log(0) equating to 0 for impurity calculation differs from computer implementations. Consequently, to address this discrepancy, I've employed log(1.0001-p) instead of log(1-p) in the calculation to ensure computational accuracy.

**2)**

The "normalize" and "denormalize" methods serve to rescale input data to the [0, 1] range and reverse this process, respectively. Within the network's hidden layer, the tanh function was employed as the activation function due to its differentiability and output range of [-1, 1], aiding in the regulation of neuron outputs. Empirically, the sequence of normalizing followed by utilizing tanh as the activation function for this layer has demonstrated improved outcomes. A visual representation of this outcome is depicted in the following graph.



**Important note:** Interpretation errors can occur in Google Colab due to the version difference, specifically with Numpy 1.23.5. However, the code runs smoothly without any issues when version 1.26.1 is used locally.