WEEK- 7: Decision trees.

Session No.4

Decision Tree

- Decision tree algorithm falls under the category of supervised learning. They can be used tosolve both regression and classification problems.
- Decision tree uses the tree representation to solve the problem.
- A decision tree is a classification and prediction tool having a tree-like structure, where eachinternal node denotes a test on an attribute, each branch represents an outcome of the test, and each leaf node (terminal node) holds a class label.

Below are some assumptions that we made while using decision tree:

- At the beginning, we consider the whole training set as the root.
- Feature values are preferred to be categorical. If the values are continuous then they arediscretized prior to building the model.
- On the basis of attribute values records are distributed recursively.
- We use statistical methods for ordering attributes as root or the internal node.

In Decision Tree the major challenge is to identification of the attribute for the root node in each level. This process is known as attribute selection. We have two popular attributeselection measures:

1. Information Gain
2. Gini Index

Attribute selection using Information Gain

Entropy:

Entropy is a measure of the randomness in the information being processed. The higherthe entropy, the harder it is to draw any conclusions from that information.

$$\mathrm{H}(X) = -\sum_{i=1}^{n} \mathrm{P}(x_i) \log_b \mathrm{P}(x_i)$$

Information Gain:

Information gain can be defined as the amount of information gained about a random variable or signal from observing another random variable. It can be considered as the difference between the entropy of parent node and weighted average entropy of child nodes.

$$IG(S,A) = H(S) - H(S,A)$$

Alternatively,

$$IG(S,A) = H(S) - \sum_{i=0}^{n} P(x) * H(x)$$

Building Decision Tree using Information Gain

- Start with all training instances associated with the root node
- Use information gain to choose which attribute to label each node with
- Note: No root-to-leaf path should contain the same discrete attribute twice
- Recursively construct each subtree on the subset of training instances that would be classified down that path in the tree.

Example:

Outlook	Temperature	Humidity	Wind	Played football(yes/no)
Sunny	Hot	High	Weak	No
Sunny	Hot	High	Strong	No
Overcast	Hot	High	Weak	Yes
Rain	Mild	High	Weak	Yes
Rain	Cool	Normal	Weak	Yes
Rain	Cool	Normal	Strong	No
Overcast	Cool	Normal	Strong	Yes
Sunny	Mild	High	Weak	No
Sunny	Cool	Normal	Weak	Yes
Rain	Mild	Normal	Weak	Yes
Sunny	Mild	Normal	Strong	Yes
Overcast	Mild	High	Strong	Yes
Overcast	Hot	Normal	Weak	Yes
Rain	Mild	High	Strong	No

Find the entropy of the class variable.

E(S) = -[(9/14)log(9/14) + (5/14)log(5/14)] = 0.94

where, S = Playing Football

From the above data for outlook, we can arrive at the following table

		play		
		yes	no	total
	sunny	3	2	5
Outlook	overcast	4	0	4
	rainy	2	3	5
				14

Now we have to calculate average weighted entropy. ie, we have found the total of weights ofeach feature multiplied by probabilities.

E(S, outlook)
= P(Sunny)* Entropy(Sunny) + P(Overcast)*Entropy(Overcast)+ P(Rain)*Entropy(Rain)
= (5/14)*E(2,3) + (4/14)*E(4,0) + (5/14)*E(3,2)
= (5/14)[-(2/5)log(2/5)-(3/5)log(3/5)] + (4/14)(0) +(5/14)[-(3/5)log(3/5)-(2/5)log(2/5)]
= 0.693

The next step is to find the information gain. It is the difference between parent entropy andaverage weighted entropy we found above.

IG(S, outlook) = 0.94 - 0.693 = 0.247

Similarly find Information gain for Temperature, Humidity, and Windy.IG(S, Temperature) = 0.940 - 0.911 = 0.029
IG(S, Humidity) = 0.940 - 0.788 = 0.152
IG(S, Windy) = 0.940 - 0.8932 = 0.048

Now select the feature having the largest entropy gain. Here it is Outlook. So it forms the firstnode (root node) of our decision tree.

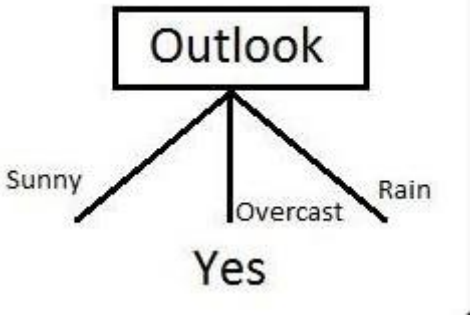
Now our data look as follows

Outlook	Temperature	Humidity	Wind	Played football(yes/no)
Sunny	Hot	High	Weak	No
Sunny	Hot	High	Strong	No
Sunny	Mild	High	Weak	No
Sunny	Cool	Normal	Weak	Yes
Sunny	Mild	Normal	Strong	Yes

Outlook	Temperature	Humidity	Wind	Played football(yes/no)
Overcast	Hot	High	Weak	Yes
Overcast	Cool	Normal	Strong	Yes
Overcast	Mild	High	Strong	Yes
Overcast	Hot	Normal	Weak	Yes

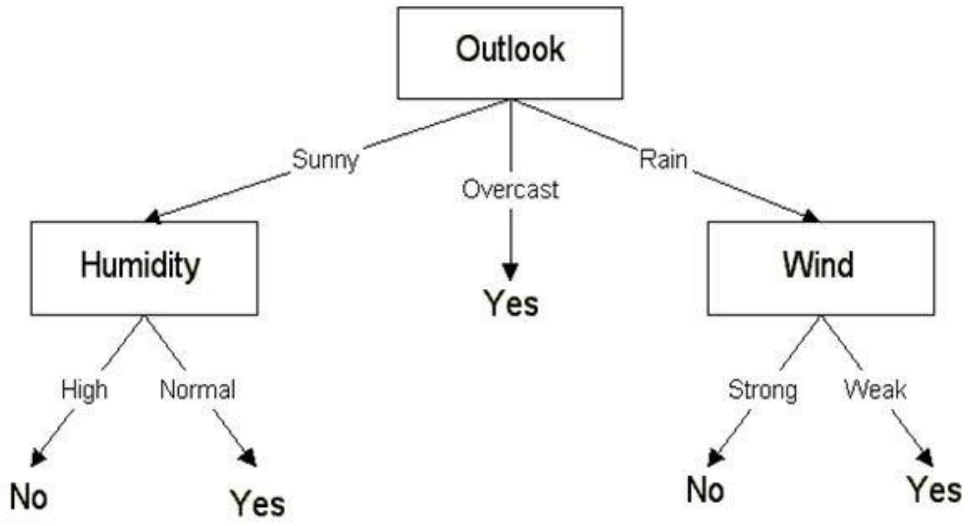
Outlook	Temperature	Humidity	Wind	Played football(yes/no)
Rain	Mild	High	Weak	Yes
Rain	Cool	Normal	Weak	Yes
Rain	Cool	Normal	Strong	No
Rain	Mild	Normal	Weak	Yes
Rain	Mild	High	Strong	No

Since overcast contains only examples of class 'Yes' we can set it as yes. That means Ifoutlook is overcast football will be played. Now our decision tree looks as follows.



Repeat the above process for each type under Outlook, i.e., Sunny and Rain to find theattribute with highest Information gain.

At the end, our decision tree will look as below:



Overfitting and Decision Trees

Decision Trees are prone to over-fitting. A decision tree will always overfit the training data ifwe allow it to grow to its max depth. A decision tree is overfit when the tree is trained to fit allsamples in the training data set perfectly.

You can tweak some parameters such as min_samples_leaf to minimize default overfitting. Thedeeper you allow your tree to grow, the more complex the sequence of decision rules becomes.Assigning a **maximum depth** to a tree can simplify it and combat overfitting.

Pruning

Pruning is a technique that is used to reduce overfitting. Pruning also simplifies a decision treeby removing the weakest rules. Pruning is often distinguished into:

- **Pre-pruning** (early stopping) stops the tree before it has completed classifying thetraining set,
- **Post-pruning** allows the tree to classify the training set perfectly and then prunes the tree.

Pre-pruning

- The pre-pruning technique of Decision Trees is tuning the hyperparameters prior to thetraining pipeline.
- It involves the heuristic known as 'early stopping' which stops the growth of the decisiontree - preventing it from reaching its full depth.
- It stops the tree-building process to avoid producing leaves with small samples.
- During each stage of the splitting of the tree, the cross-validation error will be monitored.
- If the value of the error does not decrease anymore - then we stop the growth of thedecision tree.
- The hyperparameters that can be tuned for early stopping and preventing overfitting are:max_depth, min_samples_leaf and min_samples_split

Post-pruning

- Post-pruning does the opposite of pre-pruning and allows the Decision Tree model togrow to its full depth.
- Once the model grows to its full depth, tree branches are removed to prevent the modelfrom overfitting.
- The algorithm will continue to partition data into smaller subsets until the final subsetsproduced are similar in terms of the outcome variable.
- The final subset of the tree will consist of only a few data points allowing the tree to havelearned the data to the T.
- However, when a new data point is introduced that differs from the learned data - it maynot get predicted well.
- The hyperparameter that can be tuned for post-pruning and preventing overfitting is ccp_alpha

- ccp stands for Cost Complexity Pruning and can be used as another option to control thesize of a tree. A higher value of ccp_alpha will lead to an increase in the number of nodespruned.