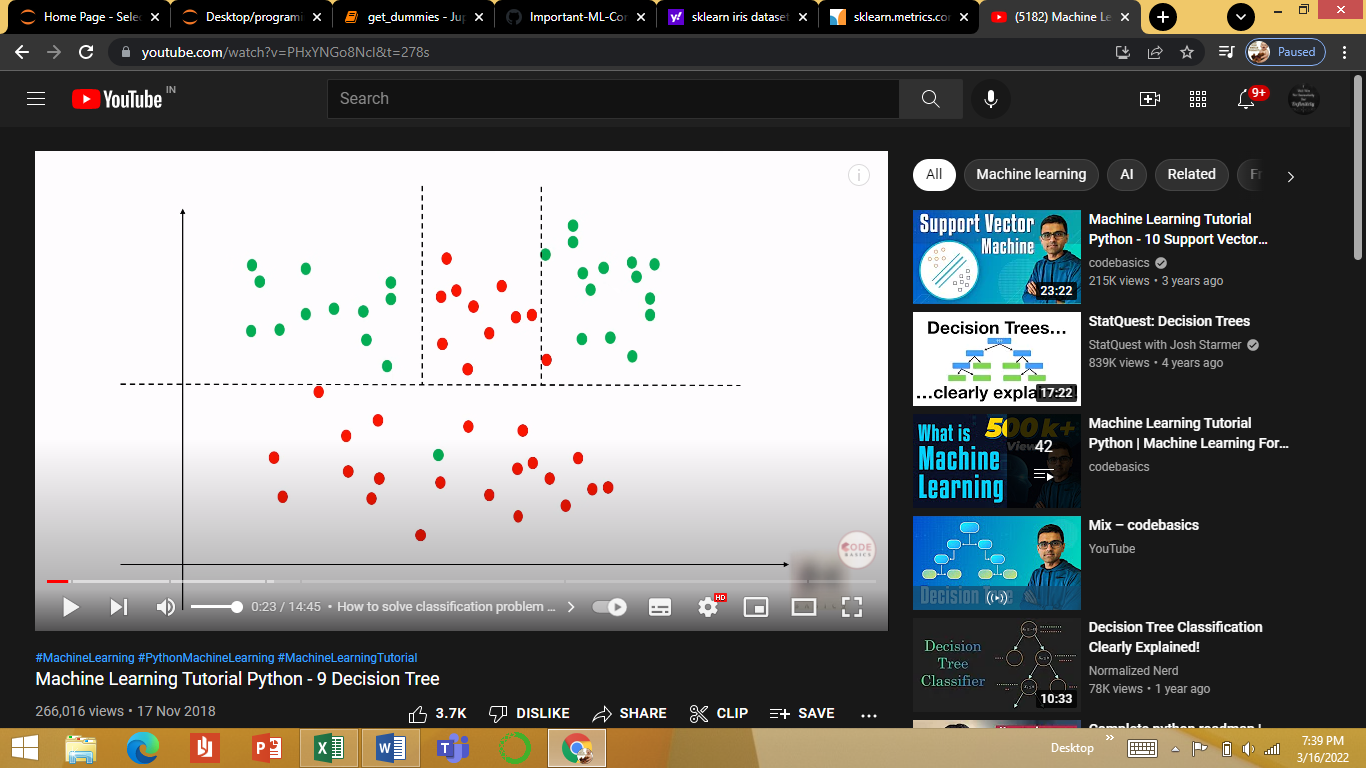
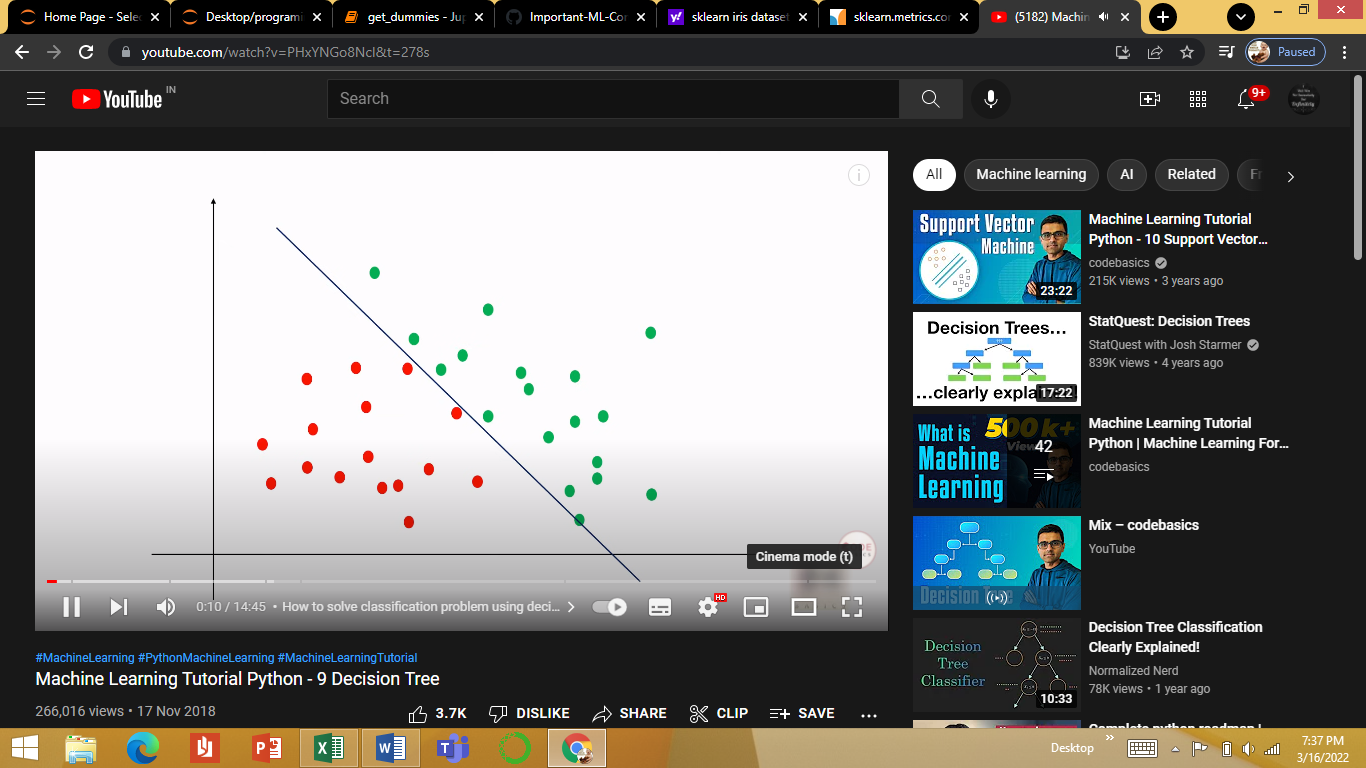
**Decision Tree**

Used for complex data sets where a clear decision boundary using logistic regression is not possible.

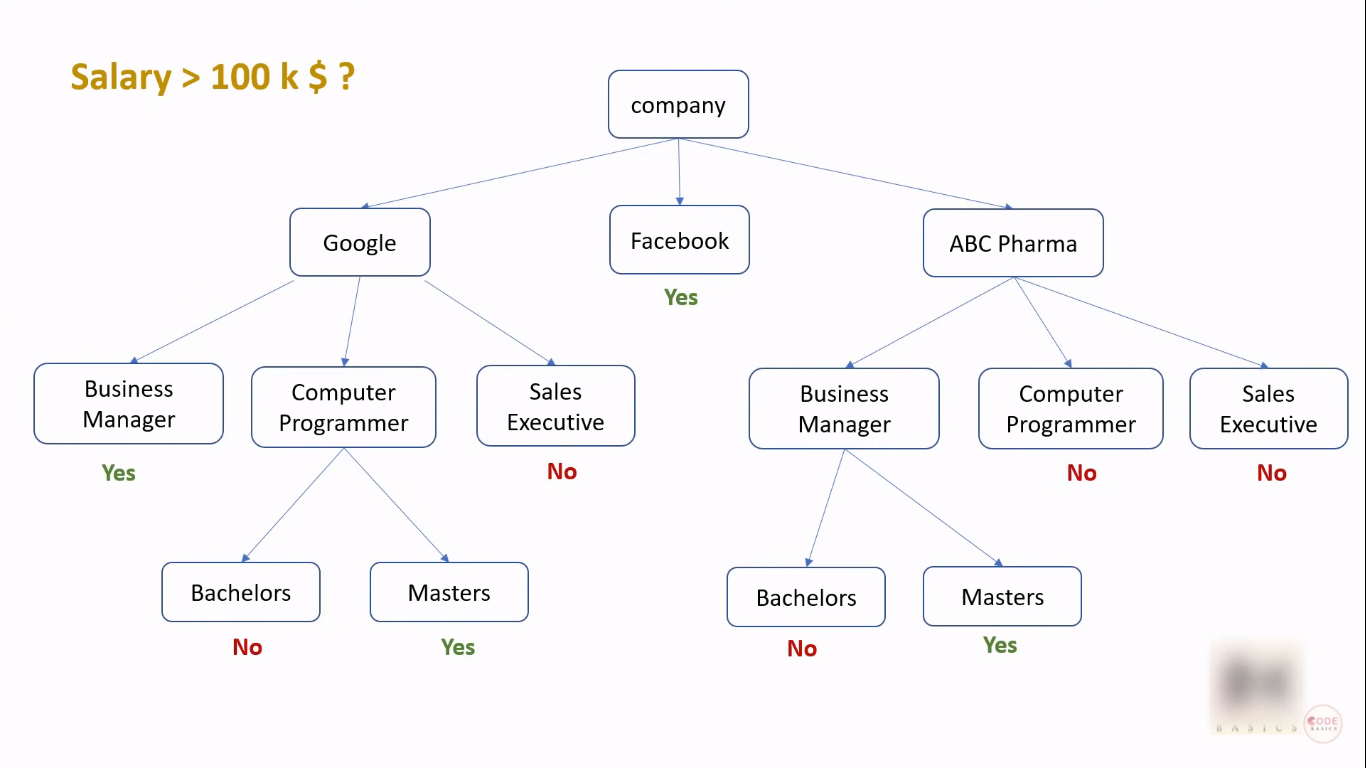
DecisionTreeRegressor : Regression Problems

DecisionTreeClassifier : Classification Problems





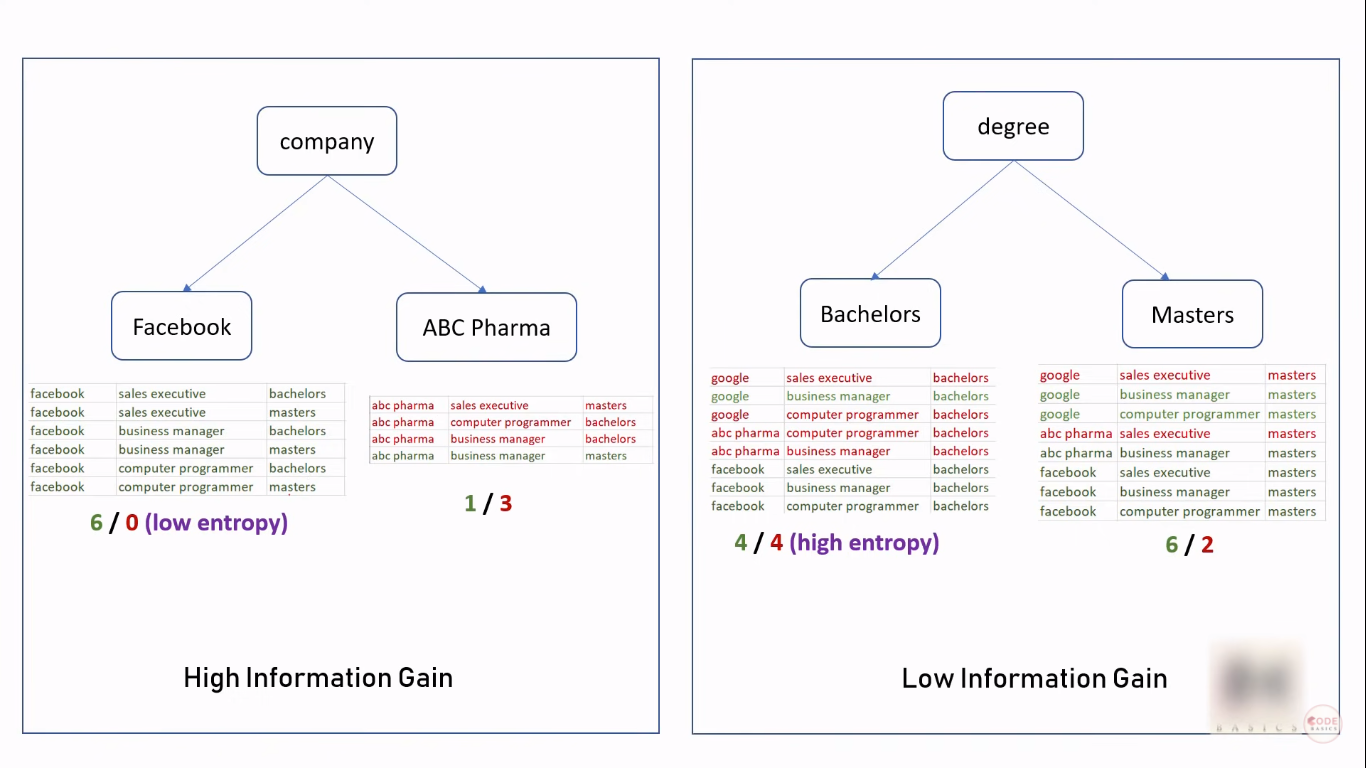
Simple Data with clear Decision boundary Complex data with unclear decision boundary

****Example Decision Tree for the salaries.csv

**Entropy:** Measure of randomess

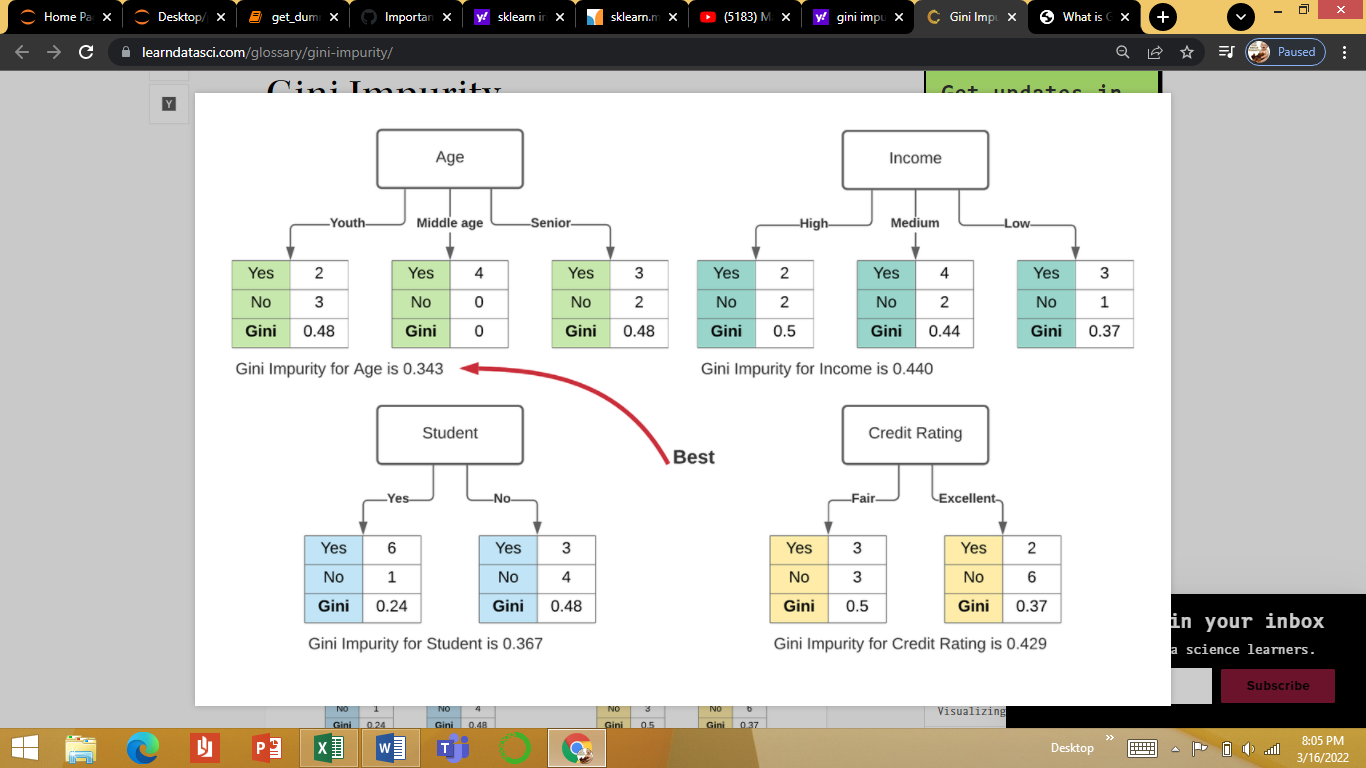
**Order of selecting the features:**

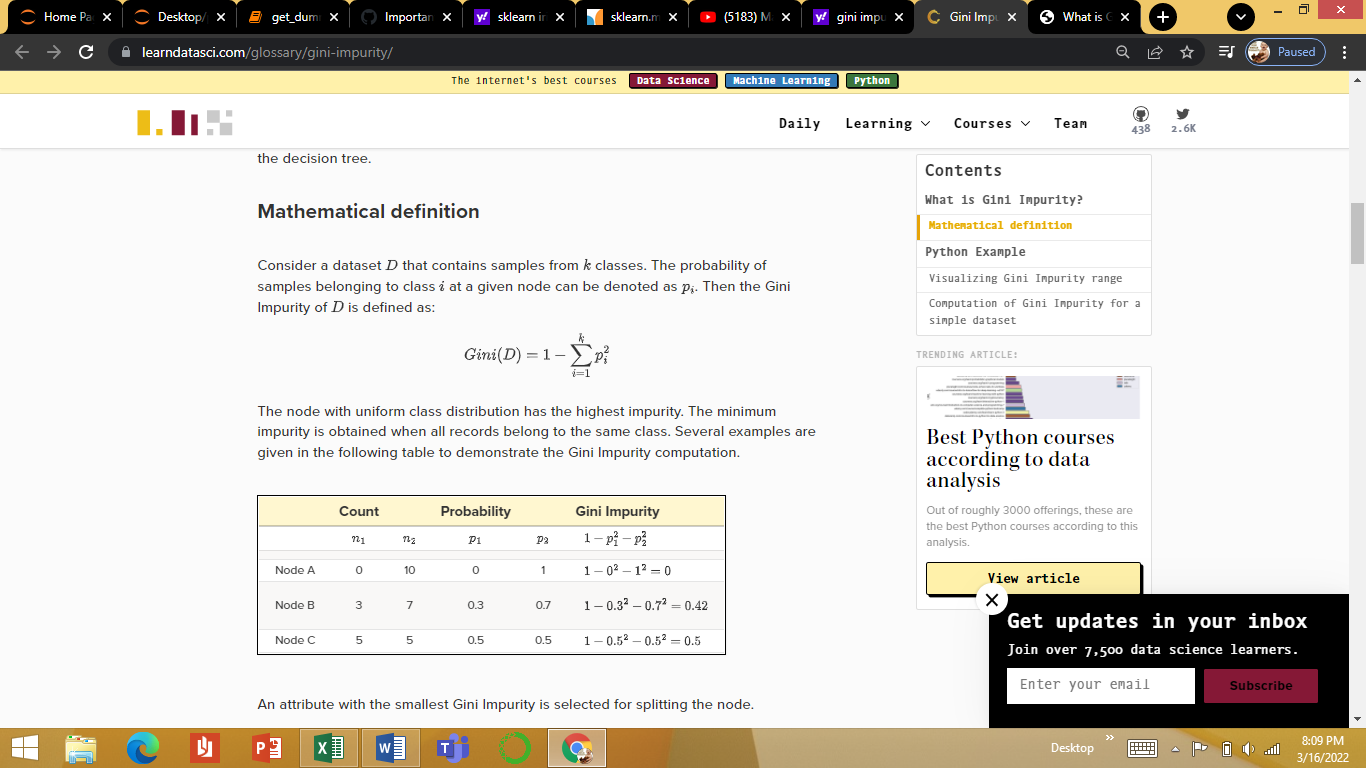
Low entropy, High information Gain (i.e. getting an almost clear subset.

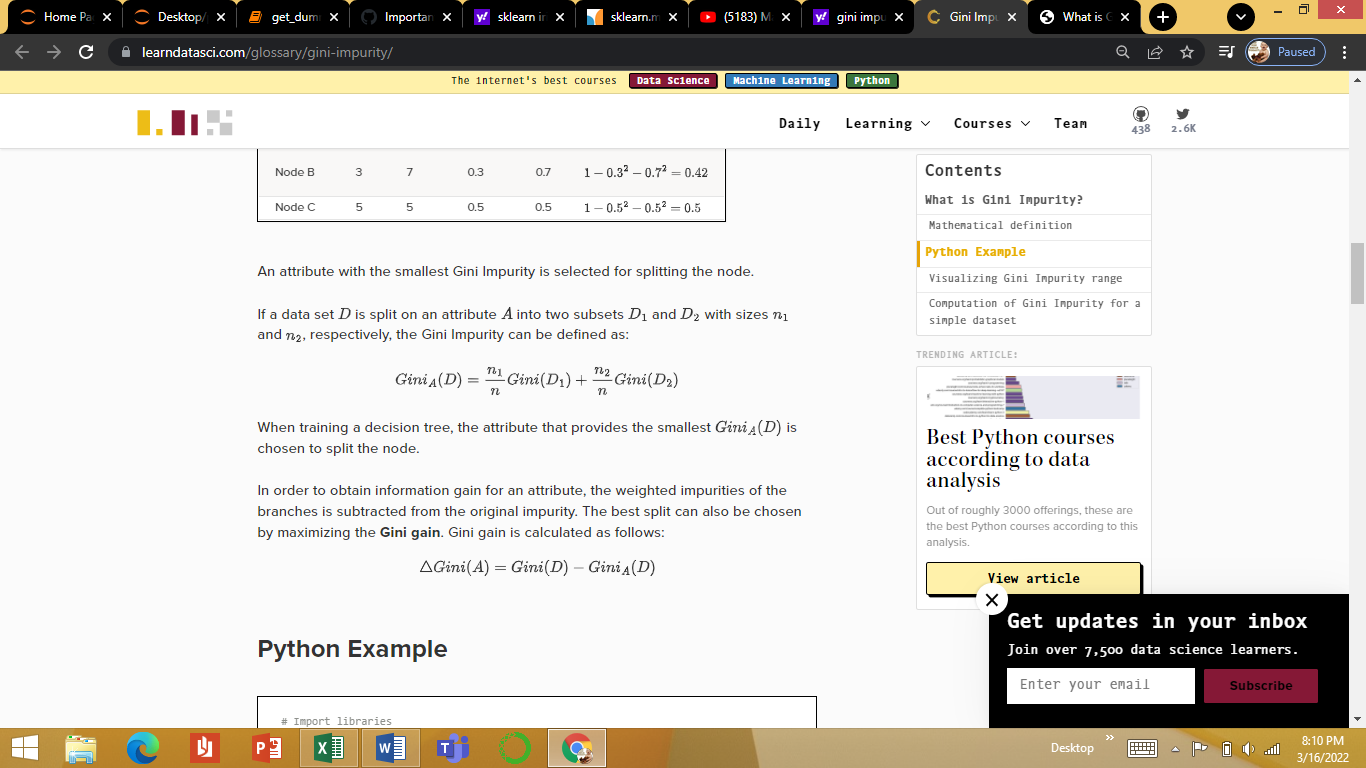


* **Gini impurity:** Gini Impurity is the probability of incorrectly classifying a randomly chosen element in the dataset if it were randomly labeled according to the class distribution in the dataset.

More precisely, the Gini Impurity of a dataset is a number between 0-0.5, which indicates the likelihood of new, random data being misclassified if it were given a random class label according to the class distribution in the dataset. An attribute with the smallest Gini Impurity is selected for splitting the node

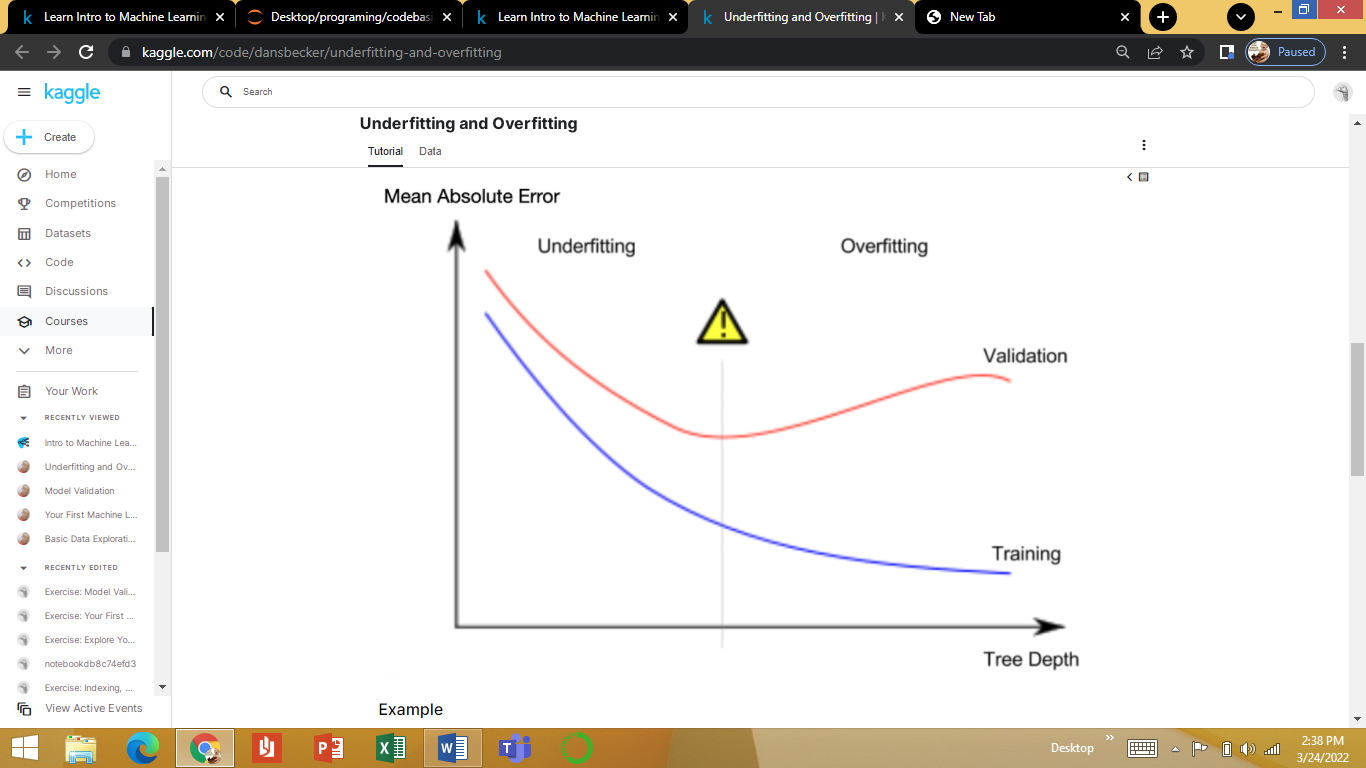
Example:



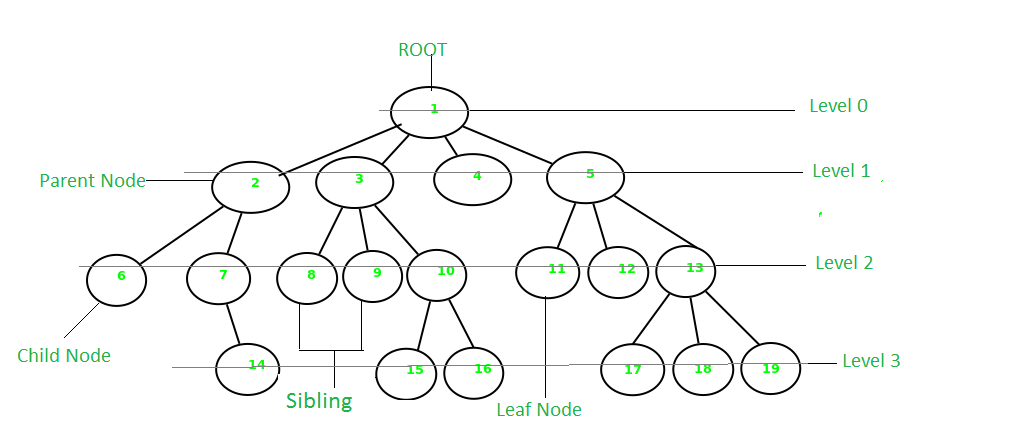


When we divide the houses amongst many leaves, we also have fewer houses in each leaf. Leaves with very few houses will make predictions that are quite close to those homes' actual values, but they may make very unreliable predictions for new data (because each prediction is based on only a few houses).

This is a phenomenon called **overfitting**, where a model matches the training data almost perfectly, but does poorly in validation and other new data.

if a tree divides houses into only 2 or 4, each group still has a wide variety of houses. Resulting predictions may be far off for most houses, even in the training data (and it will be bad in validation too for the same reason). When a model fails to capture important distinctions and patterns in the data, so it performs poorly even in training data, that is called **underfitting**.

But the max\_leaf\_nodes argument provides a very sensible way to control overfitting vs underfitting. The more leaves we allow the model to make, the more we move from the underfitting area in the above graph to the overfitting area.

**Terminology**

* **Parent Node:** The node which is a predecessor of a node is called the parent node of that node. {**2}** is the parent node of {**6, 7}**.
* **Child Node:** The node which is the immediate successor of a node is called the child node of that node. Examples: {**6, 7}** are the child nodes of {**2}**.
* **Root Node:** The topmost node of a tree or the node which does not have any parent node is called the root node. {**1}** is the root node of the tree. A non-empty tree must contain exactly one root node and exactly one path from the root to all other nodes of the tree.
* **Degree of a Node:** The total count of subtrees attached to that node is called the degree of the node. The degree of a leaf node must be **0**. The degree of a tree is the maximum degree of a node among all the nodes in the tree. The degree of the node {**3}** is **3**.
* **Leaf Node or External Node:** The nodes which do not have any child nodes are called leaf nodes. {**6, 14, 8, 9, 15, 16, 4, 11, 12, 17, 18, 19}** are the leaf nodes of the tree.
* **Ancestor of a Node:** Any predecessor nodes on the path of the root to that node are called Ancestors of that node. {**1, 2}** are the parent nodes of the node {**7}**
* **Descendant:** Any successor node on the path from the leaf node to that node. {**7, 14}** are the descendants of the node. {**2}**.
* **Sibling:** Children of the same parent node are called siblings. {**8, 9, 10}** are called siblings.
* **Depth of a node:** The count of edges from the root to the node. Depth of node {**14}** is **3**.
* **Height of a node**: The number of edges on the longest path from that node to a leaf. Height of node {**3}** is **2**.
* **Height of a tree:** The height of a tree is the height of the root node i.e the count of edges from the root to the deepest node. The height of the above tree is **3**.
* **Level of a node:** The count of edges on the path from the root node to that node. The root node has level **0**.
* **Internal node:** A node with at least one child is called Internal Node.
* **Neighbour of a Node:** Parent or child nodes of that node are called neighbors of that node.
* **Subtree**: Any node of the tree along with its descendant