Decision Tree is supervised Machine Learning Algorthim, which can used for both classification and Regression.

In [1]:
```python
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

In [2]:
```python
df=pd.read_csv(r"C:\Users\USER\Downloads\PlayTennis.csv")
```

In [3]:
```python
df.head()
```

Out[3]:

	outlook	temp	humidity	windy	play
0	sunny	hot	high	False	no
1	sunny	hot	high	True	no
2	overcast	hot	high	False	yes
3	rainy	mild	high	False	yes
4	rainy	cool	normal	False	yes

Pseudocode:

Pseudo code

- Begin with your training dataset, which should have some feature variables and classification or regression output.
- Determine the “best feature” in the dataset to split the data on; more on how we define “best feature” later
- Split the data into subsets that contain the correct values for this best feature. This splitting basically defines a node on the tree i.e each node is a splitting point based on a certain feature from our data.
- Recursively generate new tree nodes by using the subset of data created from step 3.

Decision tree algorthim mainly depend on 2 concept: Entropy and Information gain.

Entropy, in simple term, it is a measure of disorder or measure of purity/impurity.

Actually, If we know our dataset or system better, entropy is less: Example:

Salary	Age	Purchase
20000	21	Yes
10000	45	No
60000	27	Yes
15000	31	No
12000	18	No

$$H(d) = -P_y \log_2(P_y) - P_n \log_2(P_n)$$

$$H(d) = -2/5 \log_2(2/5) - 3/5 \log_2(3/5)$$

$$H(d) = 0.97$$

Salary	Age	Purchase
34000	31	No
15000	25	No
69000	57	Yes
25000	21	No
32000	28	No

$$H(d) = -P_y \log_2(P_y) - P_n \log_2(P_n)$$

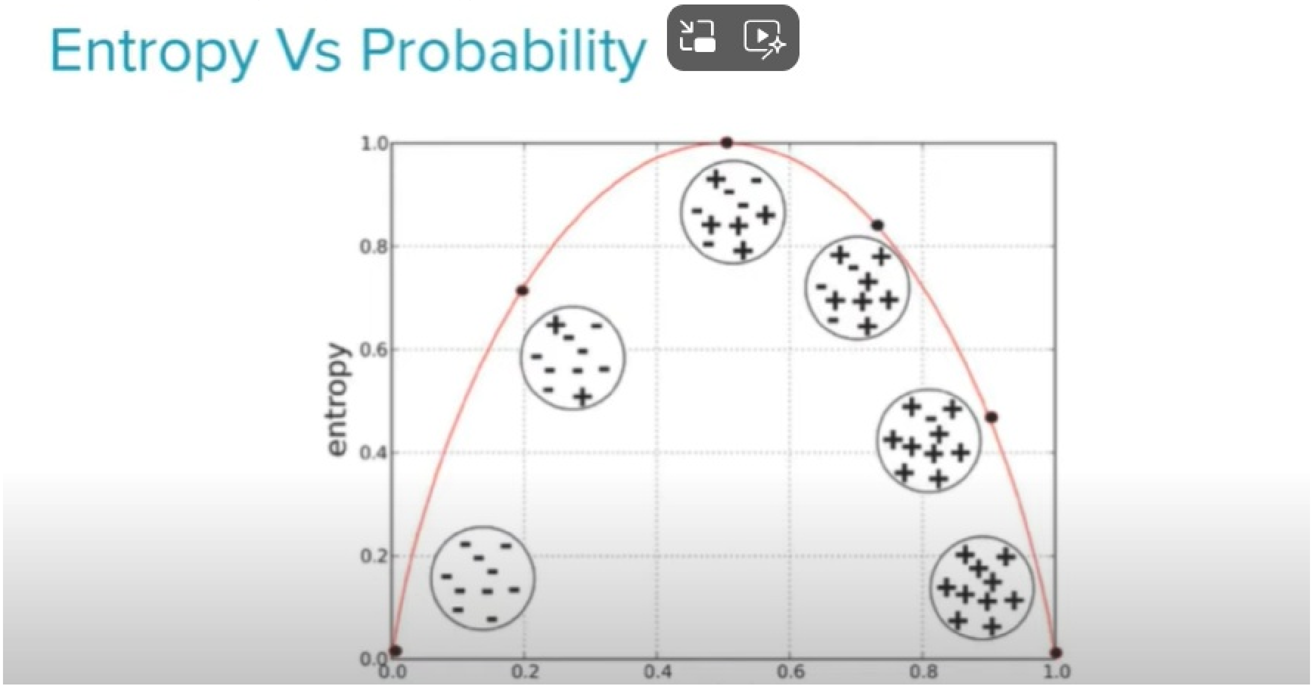
$$H(d) = -1/5 \log_2(1/5) - 4/5 \log_2(4/5)$$

$$H(d) = 0.72$$

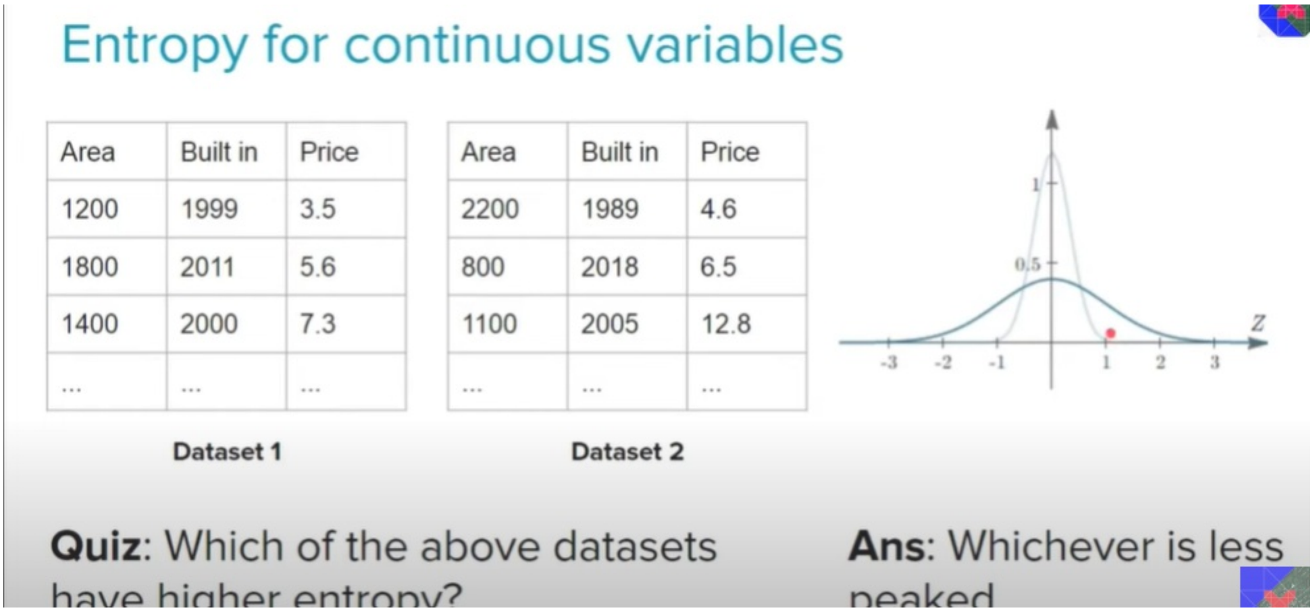
Observation:

- More uncertainity (like if we do not know more about our datasets/system) more is entropy
- For two class like above example(Y/N): minimum value for entropy is 0 and max is 1
- But, for three class, (if there is Y/N/MAYBE), minimum value is 0 but for max, it can be greater than 1.
- for formula we can use any log2 or loge

Graphical Representation (Entropy vs Probability)



For Continous variable:



So, what we graphed? We graphed price.

Gini impurity also calcualte same as entropy.

Lets take Example of Decision Tree

In [1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
```

Basic Example or data just to understand ML

In [2]:
```
data = {"Credit_Score": [650, 720, 580, 800, 690, 750, 600, 670, 710, 680], "Loan_Approval": [1, 1, 0, 1, 0, 1,
df = pd.DataFrame(data)
print(df.head())
```

```
   Credit_Score  Loan_Approval
0           650              1
1           720              1
2           580              0
3           800              1
4           690              0
```

Here we are not evaluating machine learning model, we are just Practicing thatswhy no need of spliting the dataset into test and train

In [3]:
```python
X=df[['Credit_Score']]
y=df['Loan_Approval']
```

In [4]:
```python
clf = DecisionTreeClassifier(random_state=42)
clf.fit(X, y)
```

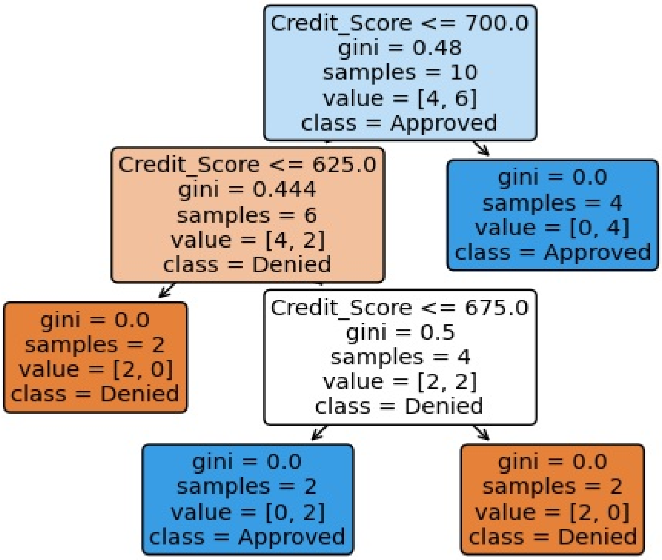
Out[4]:
```
▾          DecisionTreeClassifier
DecisionTreeClassifier(random_state=42)
```

Visualization of Decision tree

In [5]:
```python
from sklearn.tree import plot_tree
```

In [6]:
```python
plot_tree(clf, filled=True, rounded=True,
          feature_names=['Credit_Score'],
          class_names=['Denied', 'Approved'])
```

Out[6]:
```
[Text(0.6, 0.875, 'Credit_Score <= 700.0\ngini = 0.48\nsamples = 10\nvalue = [4, 6]\nclass = Approved'),
 Text(0.4, 0.625, 'Credit_Score <= 625.0\ngini = 0.444\nsamples = 6\nvalue = [4, 2]\nclass = Denied'),
 Text(0.2, 0.375, 'gini = 0.0\nsamples = 2\nvalue = [2, 0]\nclass = Denied'),
 Text(0.6, 0.375, 'Credit_Score <= 675.0\ngini = 0.5\nsamples = 4\nvalue = [2, 2]\nclass = Denied'),
 Text(0.4, 0.125, 'gini = 0.0\nsamples = 2\nvalue = [0, 2]\nclass = Approved'),
 Text(0.8, 0.125, 'gini = 0.0\nsamples = 2\nvalue = [2, 0]\nclass = Denied'),
 Text(0.8, 0.625, 'gini = 0.0\nsamples = 4\nvalue = [0, 4]\nclass = Approved')]
```



Prediction

In [7]:
```python
user_credit_score = float(input("Enter your credit score: "))

prediction = clf.predict([[user_credit_score]])

if prediction[0] == 1:
    print("Congratulations! Your loan application is likely to be approved.")
else:
    print("We regret to inform you that your loan application is likely to be denied.")
```

```
Enter your credit score: 500
We regret to inform you that your loan application is likely to be denied.
```

```
C:\Users\USER\anaconda3\Lib\site-packages\sklearn\base.py:464: UserWarning: X does not have valid feature names
, but DecisionTreeClassifier was fitted with feature names
  warnings.warn(
```

How to find Entropy and Information gain

- Entropy: A measure of impurity or disorder in a dataset.
- Information Gain: The reduction in entropy after splitting a dataset based on a feature. It measures how well a given feature separates the classes in the dataset.

In [2]:
```python
# Example dataset is above dataset
```

```
#calculating entropy for target
```

Step 1:

For our dataset, let's calculate the entropy for the target variable "PlayTennis".

- Number of instances: 14
- Number of "Yes": 9
- Number of "No": 5

$$p_{\text{Yes}} = \frac{9}{14}, \quad p_{\text{No}} = \frac{5}{14}$$

$$H(S) = -\left(\frac{9}{14}\log_2\left(\frac{9}{14}\right) + \frac{5}{14}\log_2\left(\frac{5}{14}\right)\right)$$

After this in similar manner calculating entropy for splits like summer, rainy

Let's calculate the entropy for the "Outlook" feature.

Sunny:

- Instances: 5 (3 No, 2 Yes)
- Entropy:

$$H(S_{\text{Sunny}}) = -\left(\frac{3}{5}\log_2\left(\frac{3}{5}\right) + \frac{2}{5}\log_2\left(\frac{2}{5}\right)\right)$$

$$H(S_{\text{Sunny}}) \approx -(0.6 \times -0.737 + 0.4 \times -1.322)$$

Similarly for overcast and rainy:

Overcast:

- Instances: 4 (4 Yes)
- Entropy:

$$H(S_{\text{Overcast}}) = -\left(\frac{4}{4}\log_2\left(\frac{4}{4}\right)\right) = 0$$

Rain:

- Instances: 5 (2 No, 3 Yes)
- Entropy:

$$H(S_{\text{Rain}}) = -\left(\frac{2}{5}\log_2\left(\frac{2}{5}\right) + \frac{3}{5}\log_2\left(\frac{3}{5}\right)\right)$$

Now calculating information gain of outlook feature:

For "Outlook":

$$IG(S, \text{Outlook}) = H(S) - \left(\frac{5}{14}H(S_{\text{Sunny}}) + \frac{4}{14}H(S_{\text{Overcast}}) + \frac{5}{14}H(S_{\text{Rain}})\right)$$

$$IG(S, \text{Outlook}) = 0.617 - \left(\frac{5}{14} \times 0.971 + \frac{4}{14} \times 0 + \frac{5}{14} \times 0.971\right)$$

$$IG(S, \text{Outlook}) = 0.617 - (0.3475 + 0 + 0.3475)$$

$$IG(S, \text{Outlook}) = 0.617 - 0.695 = -0.078$$

Since information gain should always be positive, lets recheck some calculation may be wrong:

Information Gain for "Outlook":

$$IG(S, \text{Outlook}) = 0.940 - \left(\frac{5}{14} \times 0.971 + \frac{4}{14} \times 0 + \frac{5}{14} \times 0.971\right)$$

$$IG(S, \text{Outlook}) = 0.940 - \left(\frac{5}{14} \times 0.971 + \frac{4}{14} \times 0 + \frac{5}{14} \times 0.971\right) \approx 0.247$$

Similarly, we can calculate information for other features too like other column. "Outlook" has an information gain of 0.247. By comparing the information gain for other features, the feature with the highest information gain is selected for the first split.