## Information gain

Before exploring this criterion, be sure you are comfortable with entropy. It is not possible to discuss information gain without entropy.

Information gain is the difference between before and after a split on a given attribute. It measures how much information a feature provides about a target.

Constructing a decision tree is solely about finding a feature that returns the highest information gain. The feature with the highest information gain produces the best split, classifying the training dataset better according to the target variable.

Information gain has the following formula:

A black background with a black square

Description automatically generated with medium confidenceInformation gain formula: Image by author

**Entropy (I)**

* Measures homogeneity of the sets
* Tells us how pure / impure a set is
* e.g. In a binary classification dataset, if **S** is the dataset having + and – classes, then Entropy (***I***nformation) is measured as:

**E(S) = -p(+)log2 p(+) – p(-)log2p(-)**

where

**p(+)** = % of positive class

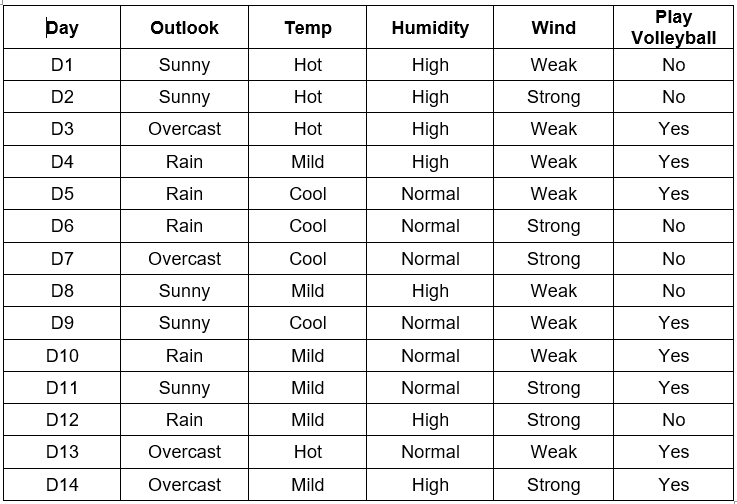
**p(-)**  = % of negative class

* Interpretation of Entropy
  + 0 <= I <= 1
  + Number of bits that is needed to identify if an item in the given dataset is + or –
  + For a pure subset, number of bits = 0
  + For a tie, number of bits = 1

Where:

* a is the specific attribute or class label.
* Entropy(S) is the entropy of dataset S.
* |Sv| / |S| is the **proportion**of the values in S to the number of values in the dataset S.

Suppose we had the following dataset:

Image by author

From the above example dataset, we are required to construct a decision tree to help us decide whether we should play volleyball based on the weather conditions.

To construct a decision tree, we need to pick the features that will best guide us to make a viable decision on whether we should play or not play volleyball.

We can't randomly select a feature from the dataset to build the tree, so **Entropy**and **information gain**are good criteria for this problem.

To begin with, we have four features we need to consider:

* Outlook
* Temp
* Humidity
* Wind

A decision tree has various parts, the root node, internal nodes, and leaf nodes. Read more on our [Decision Tree and Random Forests](https://www.machinelearningnuggets.com/decision-trees-and-random-forests/) article.

### Finding the root node feature

Since we can not just pick one of the features to start our decision tree, we need to make calculations to get the feature with the highest information gain from which we start splitting.

#### Calculate the entropy of the entire dataset(Entropy(S))

A table of weather forecasts

Description automatically generatedImage by author

We can see that we have **5 Noes**(or negatives) and **9 Yeses**(or positives). The total number of entries is **14**.

The entropy of the whole dataset is:

Entropy of Dataset: Image by author

import math

S = -(9/14) \* math.log2(9/14) - (5/14) \* math.log2(5/14)

S

# 0.9402859586706311

#### Calculate the information gain for the Outlook feature

Outlook has 3 attributes:

* Sunny
* Overcast
* Rain.

So we will calculate the entropy of each of these attributes(Sv) as follows:

We have 5 Sunny attributes for Outlook:

* 3 negative Sunny Outlooks(When Play Volleyball is No).
* 2 positive Sunny Outlooks(When Play Volleyball is Yes).

Let's calculate:

Image by author

import math

entropy\_S\_sunny = -(2/5) \* math.log2(2/5) - (3/5) \* math.log2(3/5)

entropy\_S\_sunny

# 0.9709505944546686

We have 4 Overcast attributes for Outlook:

* 0 negative Overcast Outlooks.
* 4 positive Overcast Outlooks.

Let's calculate:

Image by author

We have 5 Rain attributes for Outlook:

* 2 negative Rain attributes.
* 3 positive Rain attributes.

Let's calculate:

Image by author

import math

entropy\_S\_rain = -(3/5) \* math.log2(3/5) - (2/5) \* math.log2(2/5)

entropy\_S\_rain

# 0.9709505944546686

#### The information gain for Outlook

We have the following Entropies:

* Entropy(S) = 0.94
* Entropy(SSunny) = 0.97
* Entropy(SOvercast) = 0
* Entropy(SRain) = 0.97

We use the formula for information gain to calculate the gain.

A black background with a black square

Description automatically generated with medium confidenceImage by author

So:

A black background with a black square

Description automatically generated with medium confidenceImage by authorImage by author

**Information gain for Outlook is 0.24.**

Similarly, we have to calculate the information gain for the other features.

#### Calculate the information gain for the Temp feature

Temp has 3 attributes:

* Hot
* Mild
* Cool

Since we already have the entropy for the entire dataset(**Entropy(S)**), we will calculate the entropy of each attribute(Entropy (Sv)) of Temp, just as we did with Outlook.

The entropy of Hot:

A black background with a black square

Description automatically generated with medium confidenceImage by author

import math

entropy\_S\_hot = -(2/4) \* math.log2(2/4) - (2/4) \* math.log2(2/4)

entropy\_S\_hot

# 1.0

The entropy of Mild:

A black background with a black square

Description automatically generated with medium confidenceImage by author

import math

entropy\_S\_mild = -(4/6) \* math.log2(4/6) - (2/6) \* math.log2(2/6)

entropy\_S\_mild

# 0.9182958340544896

The entropy of Cool:

A black background with a black square

Description automatically generated with medium confidenceImage by author

import math

entropy\_S\_cool = -(3/4) \* math.log2(3/4) - (1/4) \* math.log2(1/4)

entropy\_S\_cool

# 0.8112781244591328

Calculate the information gain for the Temp feature:

Image by author

**Information gain for Temp is 0.03.**

Similarly, calculate the information gain for Humidity and Wind. All information gain values will be:

* Gain(S, Outlook) = 0.24
* Gain(S, Temp) = 0.03
* Gain(S, Humidity) =  0.15
* Gain(S, Wind) = 0.04

💡

Outlook gives the highest information about our target variable from the information gain values. It will act as the **root node** of our tree from where the splitting will begin.

A diagram of a diagram

Description automatically generatedImage by author

**Note,**for Sunny and Rain branches, we can not easily conclude a yes or a no since we have events where Play Volleyball is yes and Play volleyball is no. That means that their entropy is more than zero and hence impure. So we need to split them.

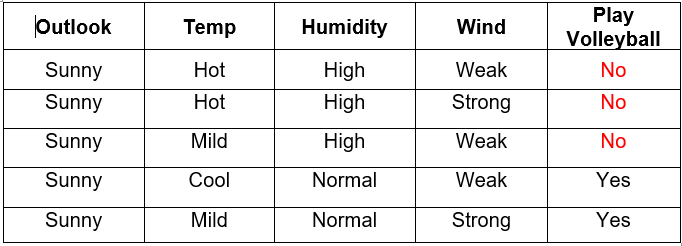
💡

Overcast is a branch with zero entropy since it has all events as Play volleyball(Yes), so it automatically becomes a leaf node.

### Finding the internal nodes

We will calculate information gain for the rest of the features when the Outlook is **Sunny**and when the Outlook is **Rain**:

#### Splitting on the Sunny attribute

Image by author

##### Calculate the information gain for Temp

Values(Temp) = Hot, Mild, Cool.

The entropy for Hot:

A black background with white text

Description automatically generatedImage by author

The entropy for Mild:

A close up of numbers

Description automatically generatedImage by author

The entropy for Cool:

A close up of a number

Description automatically generatedImage by author

**The Information gain for Temp:**

A black background with white letters

Description automatically generatedImage by authorImage by author

#### Calculate the information gain for Humidity

Values(Humidity) = High, Normal.

The entropy for Sunny:

Entropy(SSunny) = 0.97

The entropy for High:

Entropy(SHigh) = 0

Then entropy for Normal:

Entropy(SNormal) = 0

**The Information gain for Humidity:**

A black background with white text

Description automatically generatedImage by authorImage by author

##### Calculate the information gain for Wind

Values(Wind) = Strong, Weak.

The entropy for Sunny:

Entropy(SSunny) = 0.97

The entropy for Strong:

Entropy(SStrong) = 1.0

The entropy for Weak:

A close-up of a clock

Description automatically generatedImage by author

**The Information gain for Wind:**

A black background with white text

Description automatically generatedImage by authorImage by author

**Humidity**gives the **highest**information gain value(0.97). So far, our Tree will look like this:

A diagram of weather forecast

Description automatically generatedImage by author

#### Splitting on the Rain attribute

A table with text on it

Description automatically generatedImage by author

##### Calculate the information gain for Temp

Values(Temp) = Mild, Cool

The entropy for Mild:

A close up of numbers

Description automatically generated

The entropy for Cool:

Entropy(SCool) = 1.0

**The information gain for Temp:**



##### Calculate the information gain for Humidity

Values(Humidity) = High, Normal.

The entropy for High:

Entropy(SHigh) = 1.0

The Entropy for Normal:

A black and white image of a logo

Description automatically generated

**The Information gain for Humidity:**



##### Calculate the information gain for Wind

Values(Wind) = Strong, Weak.

The entropy for Strong:

Entropy(SStrong) = 0

The entropy for Weak:

Entropy(SWeak) = 0

**The information gain for Wind:**



**Wind**gives the **highest**information gain value(0.97). Now we can complete our Decision Tree.

## A complete decision tree with Entropy and Information gain criteria

A diagram of weather forecast

Description automatically generatedDecision tree (splits criteria: Entropy and Information gain): Image by author

# **Classification using the ID3 algorithm**

Consider whether a dataset based on which we will determine whether to play football or not.

A table of weather forecasts

Description automatically generated with medium confidence

Here There are for independent variables to determine the dependent variable. The independent variables are Outlook, Temperature, Humidity, and Wind. The dependent variable is whether to play football or not.

As the first step, we have to find the parent node for our decision tree. For that follow the steps:

**Find the entropy of the class variable.**

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

note: Here typically we will take log to base 2.Here total there are 14 yes/no. Out of which 9 yes and 5 no.Based on it we calculated probability above.

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

A white sheet with black text and red letters

Description automatically generated

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

E(S, outlook) = (5/14)\*E(3,2) + (4/14)\*E(4,0) + (5/14)\*E(2,3) = (5/14)(-(3/5)log(3/5)-(2/5)log(2/5))+ (4/14)(0) + (5/14)((2/5)log(2/5)-(3/5)log(3/5)) = 0.693

**The next step is to find the information gain**. It is the difference between parent entropy and average 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 first node(root node) of our decision tree.

Now our data look as follows

A table with words on it

Description automatically generated

A white table with black text

Description automatically generated

A table with words on it

Description automatically generated

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

A diagram of a diagram

Description automatically generated with medium confidence

The next step is to find the next node in our decision tree. Now we will find one under sunny. We have to determine which of the following Temperature, Humidity or Wind has higher information gain.

A white table with black text

Description automatically generated

Calculate parent entropy E(sunny)

E(sunny) = (-(3/5)log(3/5)-(2/5)log(2/5)) = 0.971.

Now Calculate the information gain of Temperature. IG(sunny, Temperature)

A screenshot of a computer

Description automatically generated

E(sunny, Temperature) = (2/5)\*E(0,2) + (2/5)\*E(1,1) + (1/5)\*E(1,0)=2/5=0.4

Now calculate information gain.

IG(sunny, Temperature) = 0.971–0.4 =0.571

Similarly we get

IG(sunny, Humidity) = 0.971

IG(sunny, Windy) = 0.020

Here IG(sunny, Humidity) is the largest value. So Humidity is the node that comes under sunny.

A white sheet with red text

Description automatically generated

For humidity from the above table, we can say that play will occur if humidity is normal and will not occur if it is high. Similarly, find the nodes under rainy.

**Note: A branch with entropy more than 0 needs further splitting.**

Finally, our decision tree will look as below:

A diagram of a weather forecast

Description automatically generated

**Formula of Gini Index**

The formula of the Gini Index is as follows:

Gini=1−∑(pi)2

where,  
‘pi’ is the probability of an object being classified to a particular class.

While building the [decision tree](https://quantra.quantinsti.com/course/decision-trees-analysis-trading-ernest-chan), we would prefer to choose the attribute/feature with the least Gini Index as the root node.

**Example of Gini Index**

Let us now see the example of the Gini Index for trading. We will make the decision tree model be given a particular set of data that is [readable for the machine](https://blog.quantinsti.com/data-preprocessing/).

Now, let us calculate Gini Index for past trend, open interest, trading volume and return in the following manner with the example data:

|  |  |  |  |
| --- | --- | --- | --- |
| Past Trend | Open Interest | Trading Volume | Return |
| Positive | Low | High | Up |
| Negative | High | Low | Down |
| Positive | Low | High | Up |
| Positive | High | High | Up |
| Negative | Low | High | Down |
| Positive | Low | Low | Down |
| Negative | High | High | Down |
| Negative | Low | High | Down |
| Positive | Low | Low | Down |
| Positive | High | High | Up |

**Calculation of Gini Index**

We will now calculate the Gini Index with the following -

* Calculating the Gini Index for past trend
* Calculating the Gini Index for open interest

**Calculating the Gini Index for past trend**

Since the past trend is positive 6 number of times out of 10 and negative 4 number of times, the calculation will be as follows:

P(Past Trend=Positive): 6/10

P(Past Trend=Negative): 4/10

* If (Past Trend = Positive & Return = Up), probability = 4/6
* If (Past Trend = Positive & Return = Down), probability = 2/6

Gini Index = 1 - ((4/6)^2 + (2/6)^2) = 0.45

* If (Past Trend = Negative & Return = Up), probability = 0
* If (Past Trend = Negative & Return = Down), probability = 4/4

Gini Index = 1 - ((0)^2 + (4/4)^2) = 0

* Weighted sum of the Gini Indices can be calculated as follows:

Gini Index for Past Trend = (6/10)0.45 + (4/10)0 = 0.27

[**Decision Trees in Financial Markets ›**](https://quantra.quantinsti.com/course/decision-trees-analysis-trading-ernest-chan)

**Create a machine learning trading strategy using Decision Trees and ensemble methods**

**Calculating the Gini Index for open interest**

Coming to open interest, the open interest is high 4 times and low 6 times out of total 10 times and is calculated as follows:

P(Open Interest=High): 4/10

P(Open Interest=Low): 6/10

* If (Open Interest = High & Return = Up), probability = 2/4
* If (Open Interest = High & Return = Down), probability = 2/4

Gini Index = 1 - ((2/4)^2 + (2/4)^2) = 0.5

* If (Open Interest = Low & Return = Up), probability = 2/6
* If (Open Interest = Low & Return = Down), probability = 4/6

Gini Index = 1 - ((2/6)^2 + (4/6)^2) = 0.45

* Weighted sum of the Gini Indices can be calculated as follows:

Gini Index for Open Interest = (4/10)0.5 + (6/10)0.45 = 0.47

**Calculating the Gini Index for trading volume**

Trading volume is 7 times high and 3 times low and is calculated as follows:

P(Trading Volume=High): 7/10

P(Trading Volume=Low): 3/10

* If (Trading Volume = High & Return = Up), probability = 4/7
* If (Trading Volume = High & Return = Down), probability = 3/7

Gini Index = 1 - ((4/7)^2 + (3/7)^2) = 0.49

* If (Trading Volume = Low & Return = Up), probability = 0
* If (Trading Volume = Low & Return = Down), probability = 3/3

Gini Index = 1 - ((0)^2 + (1)^2) = 0

* Weighted sum of the Gini Indices can be calculated as follows:

Gini Index for Trading Volume = (7/10)0.49 + (3/10)0 = 0.34

**Gini Index attributes or features**

|  |  |
| --- | --- |
| Attributes/Features | Gini Index |
| Past Trend | 0.27 |
| Open Interest | 0.47 |
| Trading Volume | 0.34 |

From the above table, we observe that ‘past trend’ has the lowest Gini Index and hence, it will be chosen as the root node for how the decision tree works.

**Determining the sub nodes or the branches (features going down) of the decision tree**

We will repeat the same procedure to determine the sub-nodes or branches of the decision tree.

We will calculate the Gini Index for the ‘positive’ branch of past trend as follows:

|  |  |  |  |
| --- | --- | --- | --- |
| Past Trend | Open Interest | Trading Volume | Return |
| Positive | Low | High | Up |
| Positive | Low | High | Up |
| Positive | High | High | Up |
| Positive | Low | Low | Down |
| Positive | Low | Low | Down |
| Positive | High | High | Up |

**Calculating Gini Index of open interest for positive past trend**

Open interest for positive past trend is high 2 times out of 6 and low 4 times out of 6 and the Gini Index of open interest for positive past trend is calculated as follows:

P(Open Interest=High): 2/6

P(Open Interest=Low): 4/6

* If (Open Interest = High & Return = Up), probability = 2/2
* If (Open Interest = High & Return = Down), probability = 0

Gini Index = 1 - (sq(2/2) + sq(0)) = 0

* If (Open Interest = Low & Return = Up), probability = 2/4
* If (Open Interest = Low & Return = Down), probability = 2/4

Gini Index = 1 - (sq(0) + sq(2/4)) = 0.50

* Weighted sum of the Gini Indices can be calculated as follows:

Gini Index for Open Interest = (2/6)0 + (4/6)0.50 = 0.33

**Calculating Gini Index for trading volume**

The trading volume is high 4 out of 6 times and low 2 out of 6 times and is calculated as follows:

P(Trading Volume=High): 4/6

P(Trading Volume=Low): 2/6

* If (Trading Volume = High & Return = Up), probability = 4/4
* If (Trading Volume = High & Return = Down), probability = 0

Gini Index = 1 - (sq(4/4) + sq(0)) = 0

* If (Trading Volume = Low & Return = Up), probability = 0
* If (Trading Volume = Low & Return = Down), probability = 2/2

Gini Index = 1 - (sq(0) + sq(2/2)) = 0

* Weighted sum of the Gini Indices can be calculated as follows:

Gini Index for Trading Volume = (4/6)0 + (2/6)0 = 0

**Gini Index attributes or features**

|  |  |
| --- | --- |
| Attributes/Features | Gini Index |
| Open interest | 0.33 |
| Trading volume | 0 |

We will split the node further using the ‘Trading Volume’ feature, as it has the minimum Gini Index.

**Conclusion**

Gini Index is a powerful measure of the randomness or the impurity or entropy in the values of a dataset. Gini Index aims to decrease the impurities from the root nodes (at the top of decision tree) to the leaf nodes (vertical branches down the decision tree) of a decision tree model.