What is Entropy and why Information gain matter in Decision Trees?

Nasir Islam Sujan Follow
Jun 29, 2018 · 5 min read

> *According to [Wikipedia](), **Entropy** refers to disorder or uncertainty.*

> ***Definition**: **Entropy** is the measures of **impurity**, **disorder** or **uncertainty** in a bunch of examples.*

What an Entropy basically does?

*Entropy controls how a Decision Tree decides to **split** the data. It actually effects how a **Decision Tree** draws its boundaries.*

The Equation of Entropy:

$$Entropy = -\sum p(X) \log p(X)$$

here p(x) is a fraction of examples in a given class

Equation of Entropy

· · ·

What is Information gain and why it is matter in Decision Tree?

> ***Definition: Information gain (IG)*** *measures how much "information" a feature gives us about the class.*

Why it matter ?

- **Information gain** is the main key that is used by **Decision Tree Algorithms** to construct a Decision Tree.
- **Decision Trees** algorithm will always tries to maximize **Information gain**.
- An **attribute** with highest **Information gain** will tested/split first.

The Equation of Information gain:

$$\text{Information gain} = \text{entropy (parent)} - [\text{weightes average}] * \text{entropy (children)}$$

Equation of Information gain

. . .

To understand Entropy and Information gain, lets draw a simple table with some features and labels.

This example taken from Udacity (Introduction to Machine Learning) course

Here in this `table` ,

- `Grade` , `Bumpiness` and `Speed Limit` are the features and **`Speed`** is label.
- Total four observation.

. . .

First, lets work with **`Grade`** ***feature***

In the `Grade` column there are four values and correspond that values there are four labels.

Lets consider all the labels as a parent `node`.

```
SSFF => parent node
```

> ***So, what is the entropy of this parent node ?***

Lets find out,

firstly we need to find out the *fraction of examples* that are present in the parent node. There are 2 types*(slow and fast)* of example present in the parent node, and parent node contains total 4 examples.

```
1. P(slow) => fraction of slow examples in parent node
2. P(fast) => fraction of fast examples in parent node
```

lets find out `P(slow)`,

> *p(slow) = no. of slow examples in parent node / total number of examples*

$$p_{slow} = \frac{2}{4} = 0.5$$

fraction of P(slow) examples

Similarly the fraction of fast examples `P(fast)` will be,

$$p_{fast} = \frac{2}{4} = 0.5$$

fraction of P(fast) examples

So, the **entropy** of parent node:

$$Entropy_{parent} = -\sum P_{slow}log_2(P_{slow}) + P_{fast}log_2(P_{fast})$$

entropy of parent node

```
Entropy(parent) = - {0.5 log2(0.5) + 0.5 log2(0.5)}
                = - {-0.5 + (-0.5)}
                = 1
```

So the *entropy* of parent node is `1` .

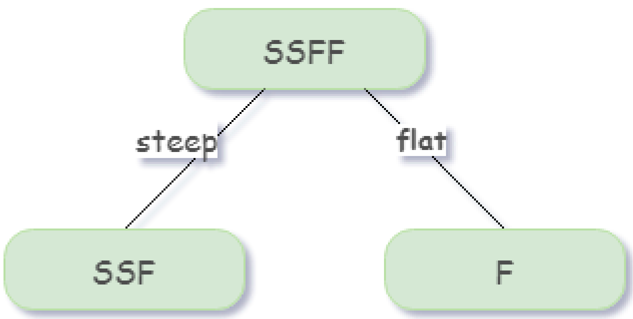
. . .

> *Now, lets explore how a* ***Decision Tree Algorithm*** *construct a* ***Decision Tree*** *based on* ***Information gain***

First lets check whether the parent node split by `Grade` or not.

If the ***Information gain*** from `Grade` feature is greater than all other features then the parent node can be split by `Grade` .

To find out ***Information gain of*** `Grade` feature, we need to virtually split the parent node by `Grade` feature.



Virtually split by Grade

Now, we need to find out the entropy both of this child nodes.

Entropy of the right side child node `(F)` is `0` *, because all of the examples in this node belongs to the same class*.

Lets find out ***Entropy*** of the left side node `SSF` :

In this node `SSF` there are two type of examples present, so we need to find out the ***fraction of slow and fast example*** separately for this node.

```
P(slow) = 2/3 = 0.667
P(fast) = 1/3 = 0.334
```

So,

```
Entropy(SSF) = - {0.667 log2(0.667) + 0.334 log2(0.334)}
             = - {-0.38 + (-0.52)}
             = 0.9
```

we can also find out the *Entropy* by using `scipy` library.

Now, we need to find out `Entropy(children)` with weighted average.

```
Total number of examples in parent node: 4
  "      "    "     "     "   left child node: 3
  "      "    "     "     "   right child node: 1
```

Formula of Entropy(children) with weighted avg. :

```
[Weighted avg]Entropy(children) =
(no. of examples in left child node) / (total no. of
examples in parent node) * (entropy of left node)
+
(no. of examples in right child node)/ (total no. of
examples in parent node) * (entropy of right node)
```

$$[weighted_{avg}](children) = \frac{3}{4} * 0.9 + \frac{1}{4} * 0$$

Entropy(children) with weighted avg. is = **0.675**

So,

Information gain = entropy (parent) – [weightes average] * entropy (children)

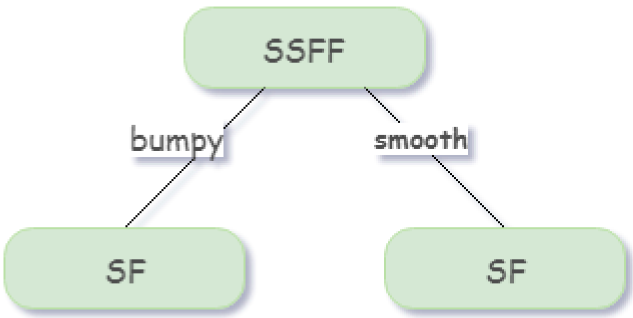
Equation of Information gain

```
Information gain(Grade) = 1 - 0.675
                        = 0.325
```

Information gain from `Grade` feature is `0.325` .

Decision Tree Algorithm choose the highest Information gain to *split/construct* a **Decision Tree.** So we need to check all the feature in order to split the Tree.

> *Information gain from* `Bumpiness`



virtually split by Bumpyness

The ***entropy*** of left and right child nodes are same because they contains same classes.

entropy(bumpy) and ***entropy(smooth)*** both equals to `1` .

So, ***entropy (children)*** with weighted avg. for `Bumpiness` :

```
[weighted avg.]entropy(children) = 2/4 * 1 + 2/4 * 1
                                 = 1
```

Hence,

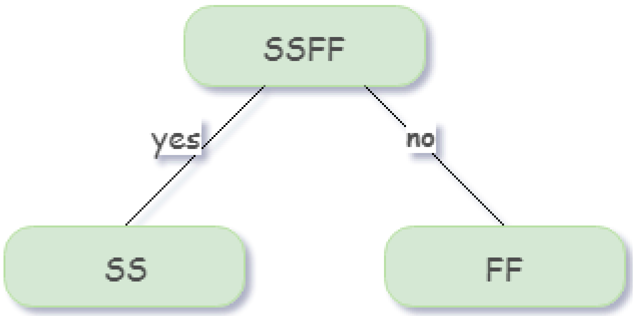
```
Information gain(Bumpiness) = 1 - 1
                            = 0
```

Till now we have to ***Information gain***:

```
IG(Grade) => 0.325
IG(Bumpiness) => 0
```

. . .

> *Information gain from* `SpeedLimit`



virtually split by SpeedLimit

The ***entropy*** of left side child node will be `0` , *because all of the examples in this node belongs to the same class.*

Similarly, ***entropy*** of right side node is `0` .

Hence, ***Entropy(children)*** with weighted avg. for `SpeedLimit` :

```
[weighted avg.] entropy(children) = 2/4 *0 + 2/4 *0
                                  = 0
```

So, ***Information gain*** from `SpeedLimit` :

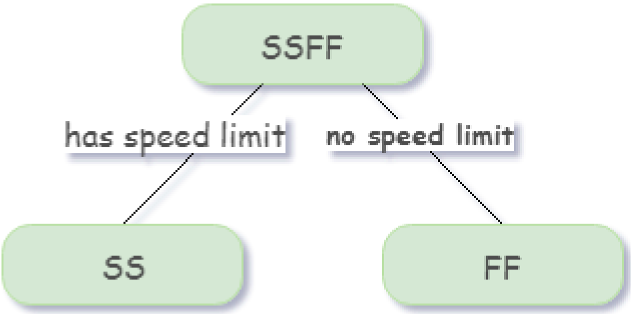
```
Information gain(SpeedLimit) = 1 - 0
                             = 1
```

Final Information gain from all the features:

```
IG(Grade) => 0.325
IG(Bumpiness) => 0
IG(SpeedLimit) => 1
```

> *As we know that,* ***Decision Tree Algorithm*** *construct* ***Decision Tree*** *based on features that have highest* ***Information gain***

So, here we can see that `SpeedLimit` has highest ***Information gain***. So the final **Decision Tree** for this datasets will be look like this:



Final Decision Tree