Gradient Boosted Trees

Gradient Boosted Trees use decision trees as estimators. It can work with different loss functions (regression, classification, risk modelling etc.), evaluate it's gradient and approximates it with a simple tree (stage-wisely, that minimizes the overall error).

AdaBoost is a special case of Gradient Boosted Tree that uses exponential loss function.

The Algorithm:

- Calculate the average of the label column as initially this average shall minimise the total error.
- Calculate the pseudo residuals.

```
Pseudo residual= actual label- the predicted result (which is average in the first iteration)
```

Mathematically,

derivative of the pseudo residual=$(\frac{\delta L(y_i, f(x_i))}{\delta(f(x_i))})$

where, L is the loss function.

```
Here, the gradient of the error term is getting calculated as the goal is to minimize the error. Hence the name gradien
t boosted trees
```

- create a tree to predict the pseudo residuals instead of a tree to predict for the actual column values.
- new result= previous result+learning rate* residual

Mathematically, $F_1(x) = F_0(x) + \nu \sum \gamma$

where ν is the learning rate and γ is the residual

Repeat these steps until the residual stops decreasing

Example

For understanding this algorithm we'll use the following simple dataset for weight prediction

In [1]:
```python
import pandas as pd
weight_data= pd.read_csv('weights.csv')
weight_data
```

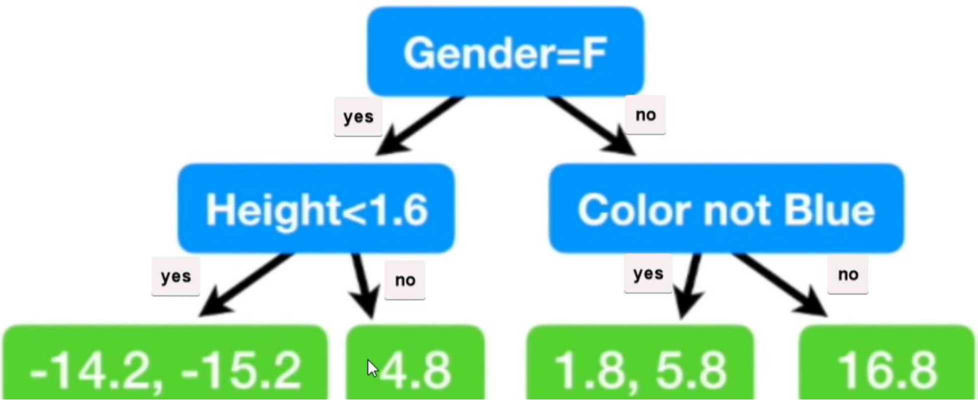
Out[1]:

	Person Height(in metres)	Person Favorite Colour	Person Gender	Person Weight (in Kg)
0	1.6	Blue	Male	88
1	1.6	Green	Female	76
2	1.5	Blue	Female	56
3	1.8	Red	Male	73
4	1.5	Green	Male	77
5	1.4	Blue	Female	57

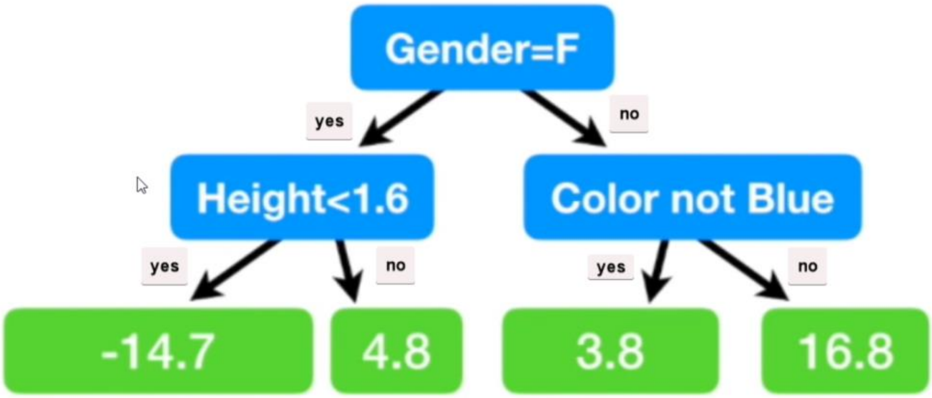
- For the first iteration, calculate the average of the target column(weight here) as it minimizes the residual initially.

Average=(88+76+56+73+77+57)/6= 71.2

- We consider this as the first prediction and then we'll calculate the residual which is the difference between the predicted and the actual value as shown below:



- Now we build a tree to predict the residuals as shown below:



- Now for prediction, we use the formula

```
New value= old value+learning rate * residual
```

If we consider the learning rate as 0.1, the result becomes.

```
New value=  71.2+0.1*16.8= 72.9 (for the first row).
```

Similarly the new predictions for all the rows is calculated.

- The above steps are repeated until there is no significant improvement in residuals.
- The final result is given by

```
Final Value= First Prediction+learning rate* 1st residual+ learning rate* 2nd residual+ and so on
```