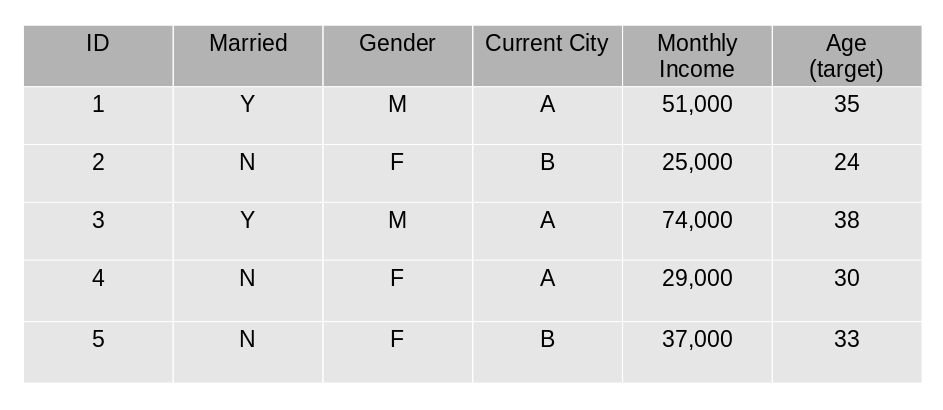
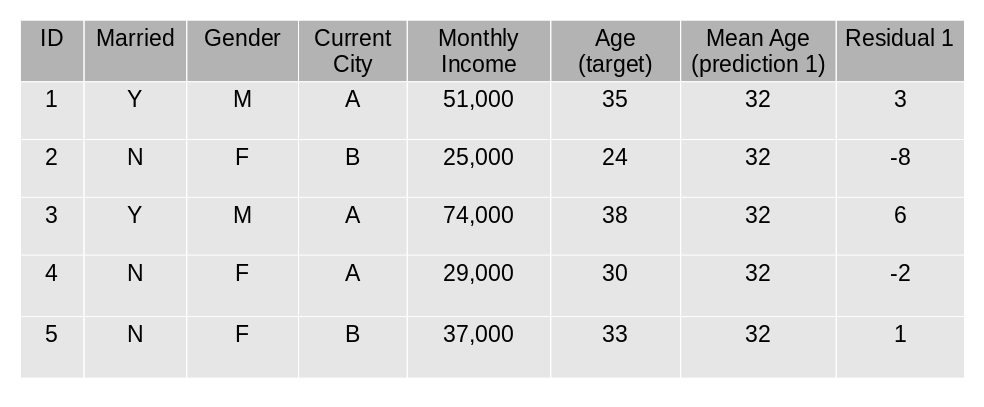
**Gradient Boosting**

Gradient Boosting or GBM is another ensemble machine learning algorithm that works for both regression and classification problems. GBM uses the boosting technique, combining a number of weak learners to form a strong learner. Regression trees used as a base learner, each subsequent tree in series is built on the errors calculated by the previous tree.

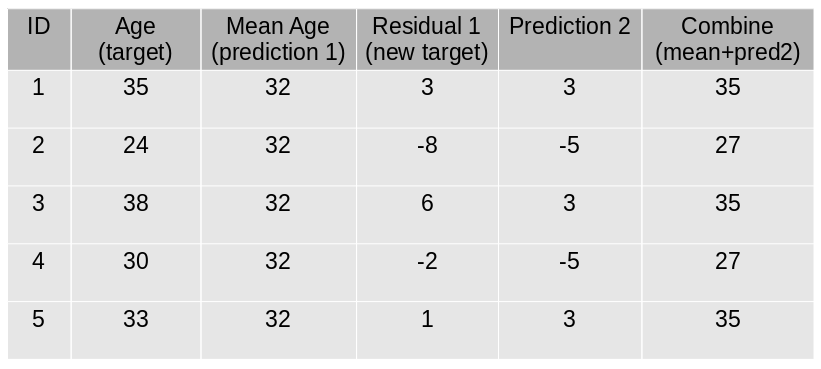
We will use a simple example to understand the GBM algorithm. We have to predict the age of a group of people using the below data:



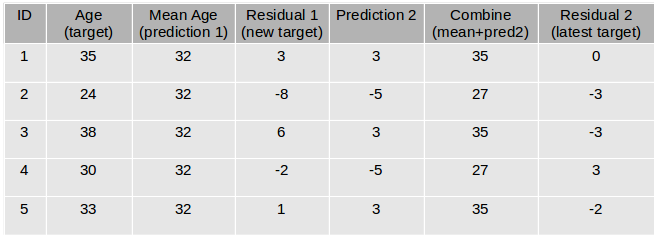
1. The mean age is assumed to be the predicted value for all observations in the dataset.
2. The errors are calculated using this mean prediction and actual values of age.



1. A tree model is created using the errors calculated above as target variable. Our objective is to find the best split to minimize the error.
2. The predictions by this model are combined with the predictions 1

.

1. This value calculated above is the new prediction.
2. New errors are calculated using this predicted value and actual value.



1. Steps 2 to 6 are repeated till the maximum number of iterations is reached (or error function does not change).

**Parameters**

* **min\_samples\_split**
  + Defines the minimum number of samples (or observations) which are required in a node to be considered for splitting.
  + Used to control over-fitting. Higher values prevent a model from learning relations which might be highly specific to the particular sample selected for a tree.
* **min\_samples\_leaf**
  + Defines the minimum samples required in a terminal or leaf node.
  + Generally, lower values should be chosen for imbalanced class problems because the regions in which the minority class will be in the majority will be very small.
* **min\_weight\_fraction\_leaf**
  + Similar to min\_samples\_leaf but defined as a fraction of the total number of observations instead of an integer.
* **max\_depth**
  + The maximum depth of a tree.
  + Used to control over-fitting as higher depth will allow the model to learn relations very specific to a particular sample.
  + Should be tuned using CV.
* **max\_leaf\_nodes**
  + The maximum number of terminal nodes or leaves in a tree.
  + Can be defined in place of max\_depth. Since binary trees are created, a depth of ‘n’ would produce a maximum of 2^n leaves.
  + If this is defined, GBM will ignore max\_depth.
* **max\_features**
  + The number of features to consider while searching for the best split. These will be randomly selected.
  + As a thumb-rule, the square root of the total number of features works great but we should check up to 30-40% of the total number of features.
  + Higher values can lead to over-fitting but it generally depends on a case to case scenario.