**Ways to ensemble:**

**Bagging – Random Forest**
**Boosting – AdaBoost, XGBoost**

**Bagging – Bootstrap Aggregation**

Let **P** be the true population where the training set **S** came from.

Assume that **P = S.**

We can know create samples (with replacements – **Bootstrap Samples**) **Z** from the new population **S**.

Let $Z_1$, …. , $Z_M$ be the bootstrap samples we created from S.

We train our model $G_m$ on $Z_m$ (m'th model on m'th bootstrap sample).

At last, we summarize the predictions of each model using the formula:
$G(m) = \sum_{m=1}^{M} \frac{G_m(x)}{M}$

**Bagging – Bias – Variance Analysis:**

$$Var(\bar{X}) = \rho\sigma^2 + \frac{1-\rho}{M}\sigma^2$$

Bootstrapping is driving down $\rho$ which is the correlation between models.

Thus, Bigger M → Less Variance.

$\rho$ is bounded so we can't decrease it down to 0.

However, Bias is slightly increased since we train our model on less data (becomes less complex).

The decrease of variance is much larger than the increase in bias.

**Boosting**

Decrease bias.
Additive.

Boosting increases the weights of our models' mistakes.
At each step reweights the sample according to the previous models.

Objective:

$$Obj^{(t)} = \sum_{i=1}^{n} L\left(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)\right) + \Omega(f_t) + constant$$

Since it is complicated to solve, we use Taylor expansion of the objective function.

$$Recall: f(x) \approx f(x) + f'(x)\Delta x + \frac{1}{2} f''(x)\Delta x^2$$

So, the Taylor expansion of the objective function is

$$Obj^{(t)} \approx \sum_{i=1}^{n} \left[L\left(y_i, \hat{y}_i^{(t-1)}\right) + g_i * f_t(x_i) + \frac{1}{2} h_i f_t''(x_i)\right] + \Omega(f_t) + constant$$
$$Where\ g_i = \partial_{\hat{y}_i^{(t-1)}} L\left(y_i, \hat{y}_i^{(t-1)}\right)\ and\ h_i = \partial^2_{\hat{y}_i^{(t-1)}} L\left(y_i, \hat{y}_i^{(t-1)}\right)$$

Removing the constant parts, we get the simplified objective function to minimize at step t:

$$\mathcal{L}^{(t)} = \sum_{i=1}^{n} \left[g_i f_t(x_i) + \frac{1}{2} h_i f_t''(x_i)\right] + \Omega(f_t)$$

Note:
$g_i$ is the **first order derivative** and $h_i$ is the **second order derivative** of the loss function with respect to previous predictions at the previous iteration.

**What is $\Omega(f_t)$?**

$$\Omega(f_t) = \gamma * T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2 ,$$

$$Where\ T\ is\ the\ number\ of\ leaves\ and\ w_j^2\ is\ L2\ normalization\ of\ leaf\ sores.$$

The larger gamma is the more conservative the algorithm is. When gamma is less than the gain value then split will happen.

The larger lambda is the more conservative the algorithm is.

Boosting is an additive process that aims to decrease bias by sequentially training a series of weak learners to correct errors made by their predecessors. The objective function incorporates the loss function, gradient, and Hessian terms, along with a regularization term, to guide model training. This leads us to the learning rate $(\eta)$, a hyperparameter that scales the contribution of each weak learner. By multiplying the gradient and Hessian terms with the learning rate, we control the step size of updates made by each weak learner.

$$\mathcal{L}^{(t)} = \sum_{i=1}^{n} \left[\eta g_i f_t(x_i) + \frac{\eta}{2} h_i f_t''(x_i)\right] + \Omega(f_t)$$

In short:

- The learning rate $\eta$ controls the step size of updates made by each weak learner.
- Regularization term $\Omega(f_t)$ controls model complexity and prevents overfitting.