1. Bagging: apply classifier C on random samples of the data followed by majority vote. If C is linear then bagging also gives a linear classifier. For bagging to be effective we want a non-linear classifier like decision trees. Thus random forests are effective because they use decisions trees as the base classifier. For example if we have three classifiers then the output of bagging is sign(⅓*C1 + ⅓*C2 + ⅓*C3)
2. Boosting: linearly combine different classifiers. Boosting reduces to bagging if we give same weight to all classifiers. In boosting we assign different weights. For example for three classifier the output would be sign(a*C1 + b*C2 + c*C3)
3. Stacking is another method to combine classifiers. It is highly effective but theoretically not understood. In stacking we make new features from the base classifiers and then apply a classifier to the new feature representation. This is a two-stage method. Suppose we have three classifiers C1, C2, and C3. We predict the train and test datasets with C1, C2, and C3. In this case we will have three outputs for each datapoint. These outputs can either be the sign function or the probability. So we now have a new representation of the data, in this case a three dimensional representation. We then apply a final classifier to the new representation of the data.

Stacking vs neural networks:

Neural network	Stacking
Optimization is on one global objective	We optimize each node independently of others. The final classifier is also applied independently of the first level classifiers.
Hidden nodes are the same classifier, usually least squares for purposes of solving it	There are no hidden nodes. Each node here is a classifier and different from the classifier in the other nodes.
We train the network on all training data	In the first level we train on part of the training data (say half) and then predict on the other half and the full test. We then switch training to the other half and repeat. Thus if we have three classifiers this would give us 6 new features.
Hard to implement, can overfit, and lots of parameters	Very easy to implement, has excellent empirical question. (Research question: can it outperform neural networks?)