Lecture#9

Ensemble Learning

Ensemble Learning

- People are good at different things
- This also applies to machine learning algorithms, they also have their pros and cons
- Instead of trying to find one perfect algorithm for our classification task, why not try several?
- We usually do that in an experiment step, then select the one that performs best
- Another option is to use Ensemble Learning

Ensemble Learning

- The idea is that you select a group of classifiers, which could be the same or different algorithms
- Each classifier is trained on a random subset of the training data
- To predict a new instance, you classify it in all of your trained classifiers and return the most frequent result (voting)

Ensemble Learning

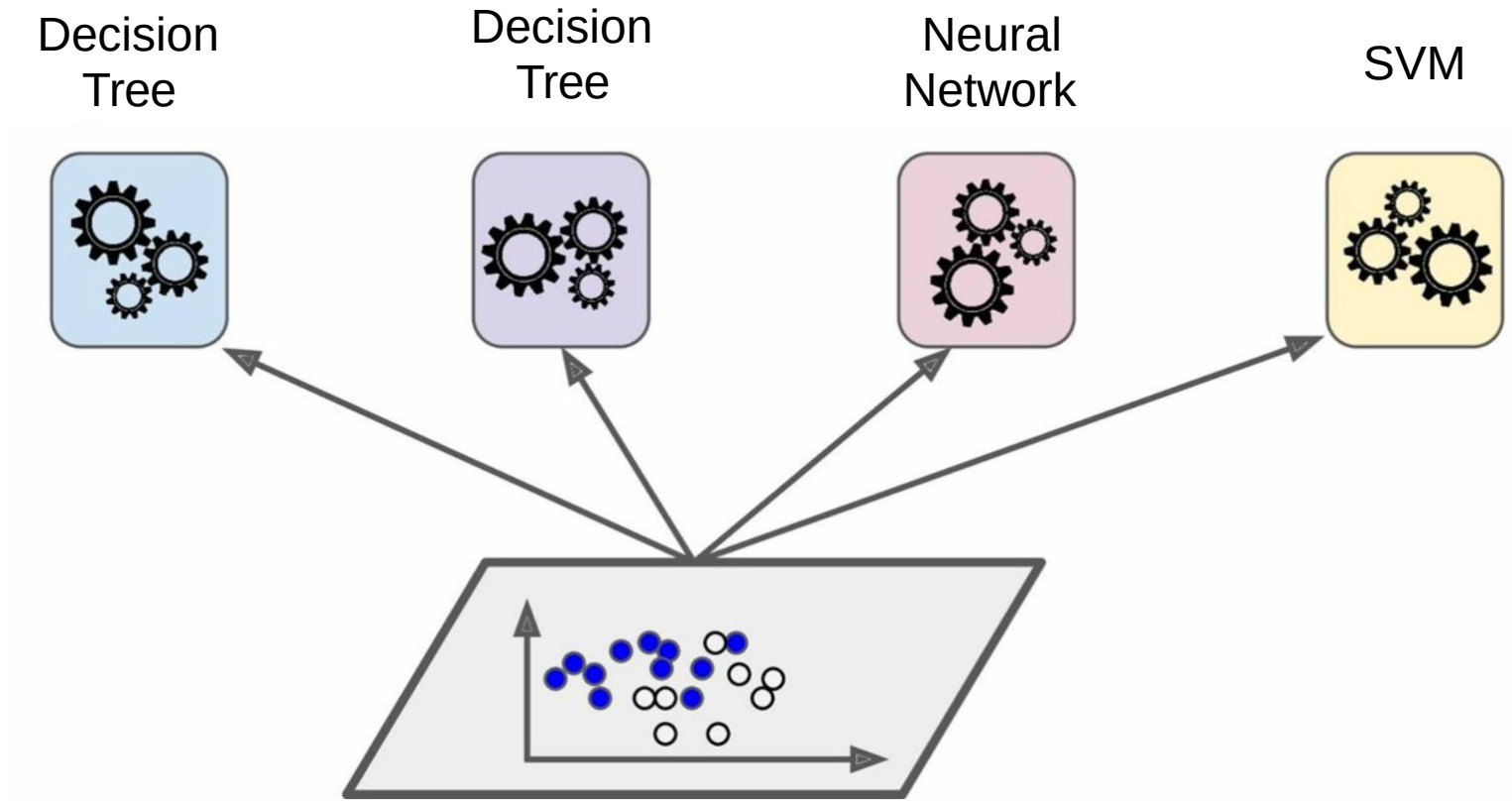


Figure from Hands-On Machine Learning with Scikit-learn and Tensorflow (Géron)

Prediction

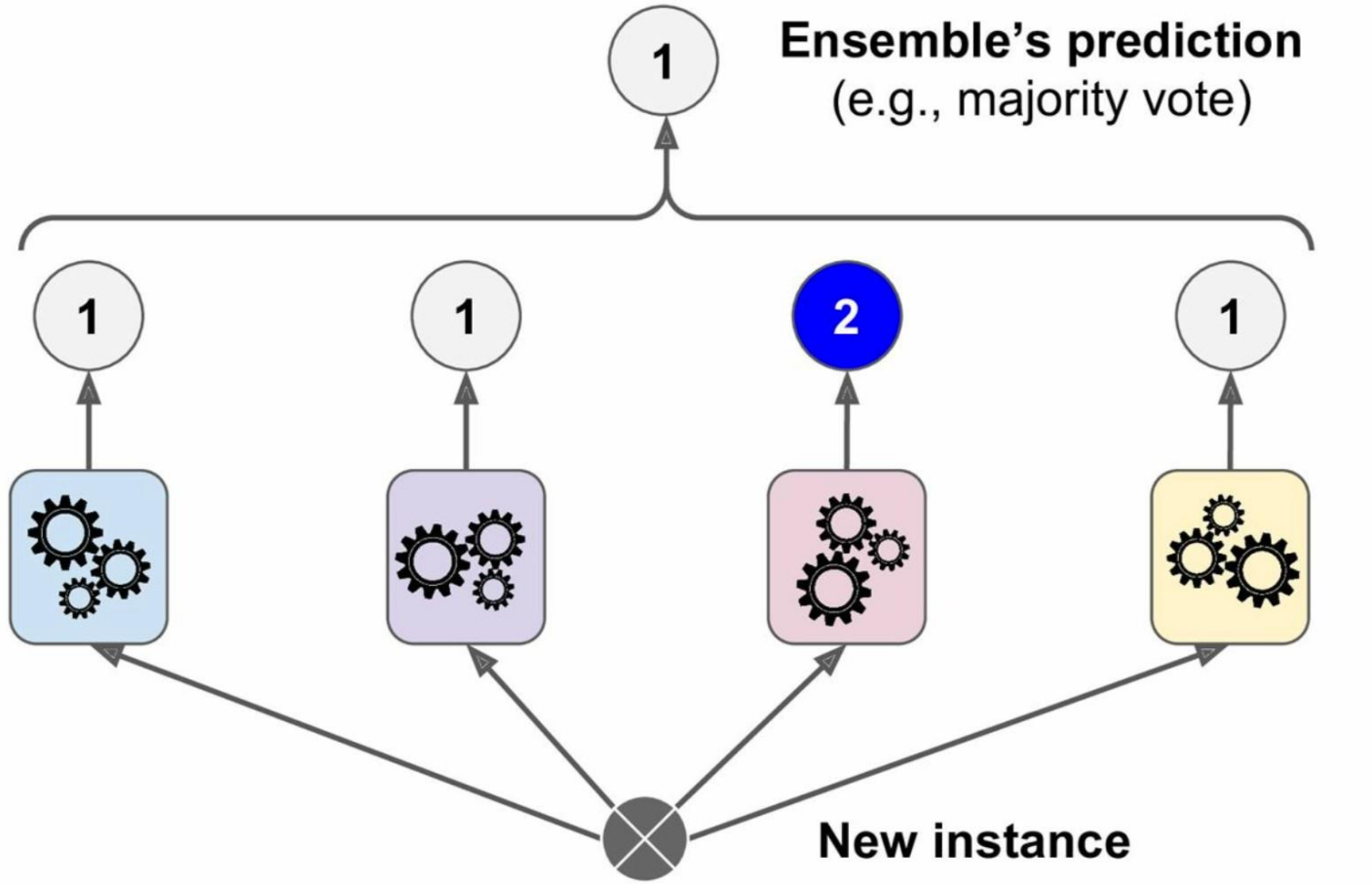


Figure from Hands-On Machine Learning with Scikit-learn and Tensorflow (Géron)

Ensemble Learning

- The ensemble often has a higher accuracy than the individual classifiers
- This is often the case even if the individuals are *weak classifiers*, meaning that they are not complex enough to model the dataset
- Ensembles are also often good at reducing overfitting and improving generalization

Bagging and Pasting

- Each individual are trained on a random subset of the training data
- This is necessary, otherwise they would likely do the same errors since they are trained on the same data
- The random selection can be done with replacement, i.e. an instance can show up more than once in the training set for an individual
- This is the most common option, and is called bagging (short for bootstrap aggregating)

Bagging and Pasting

- If the selection is performed without replacement it is called pasting
- Bagging results in more diversity in the dataset each individual is trained on
- A higher diversity is often preferred, and therefore bagging is often preferred

Random subset selection

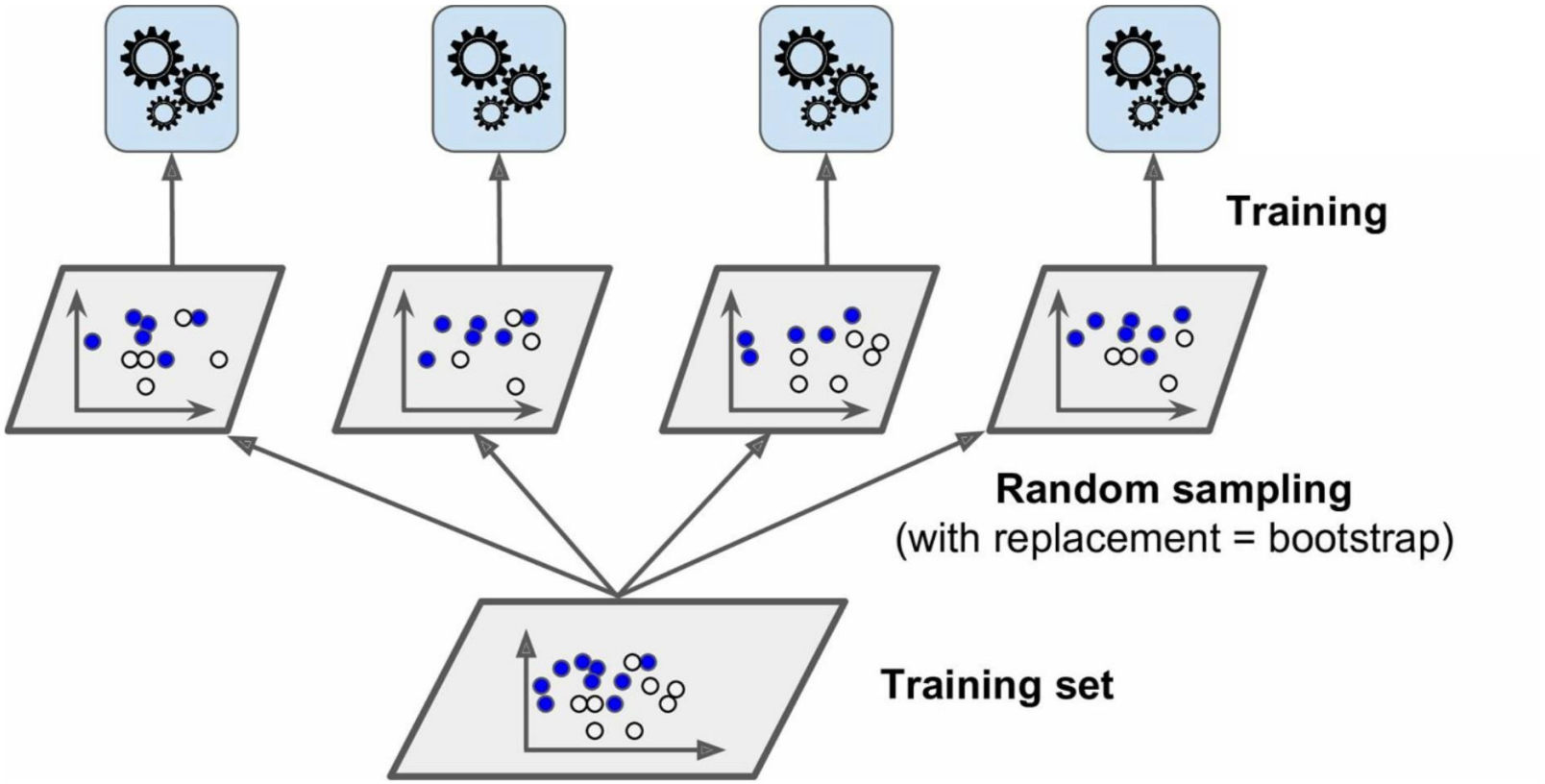


Figure from Hands-On Machine Learning with Scikit-learn and Tensorflow (Géron)

Voting

- Voting is typically done by selecting the most frequent category among the individual predictions
- This is called hard voting
- Some classifiers can output a probability that an instance belongs to each category
- If we only use such classifiers, we can apply soft voting
- We then sum and normalize the probabilities for the instance belonging to each category, and select the category with the highest probability
- Soft voting often gives better result than hard voting

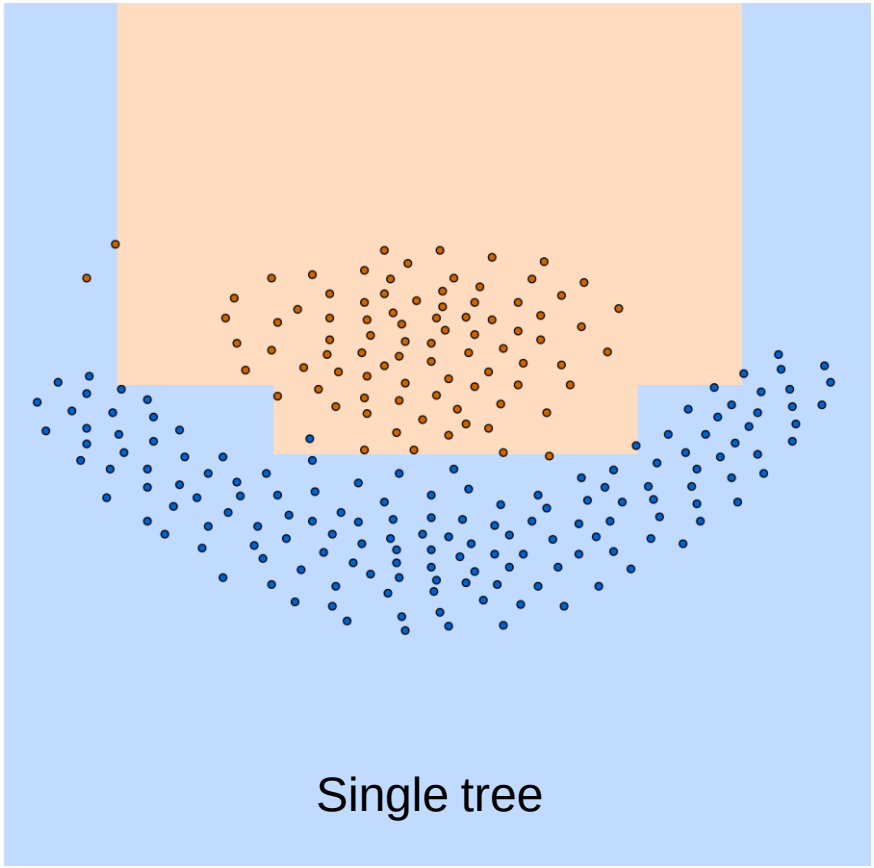
Random Forest

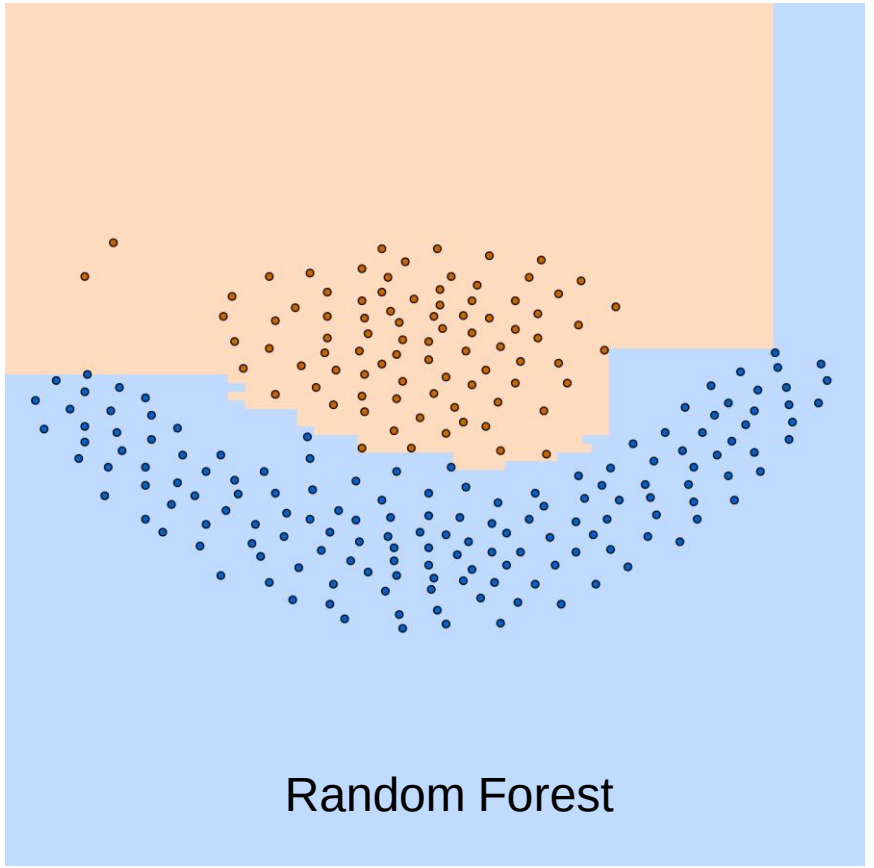
- Random Forest is an ensemble learner where we only use CART Decision Trees
- To make the trees more diverse, we only include a random subset of all attributes at each node split
- Random Forest and variants of it, such as Xgboost, is often very accurate at small to medium sized datasets

Random Forest

- To reduce overfitting, there are two techniques we can use:
  - Don’t allow a tree to grow deeper than a defined max depth limit
  - Don’t split the data at a node if there are fewer instances than a defined limit
- These two techniques are also useful for single tree classifiers

RF vs. single tree





Boosting

- Boosting is an ensemble that combines several *weak learners* into a *strong learner*
- The idea is that we first train an individual classifier on the whole dataset
- We then take the instances that the individual has problem classifying correctly
- We then train a new individual classifier on only the problematic instances from the previous individual
- We can continue with yet another individual that is trained on the problematic instances from the second classifier…

AdaBoost

- AdaBoost is the most popular boosting method
- The first classifier is trained on the whole dataset
- The subsequent classifiers are also trained on the whole dataset, but each instance is weighted on how accurate it was prediced by the previous classifier
- After each individual is trained, the weights are updated

AdaBoost

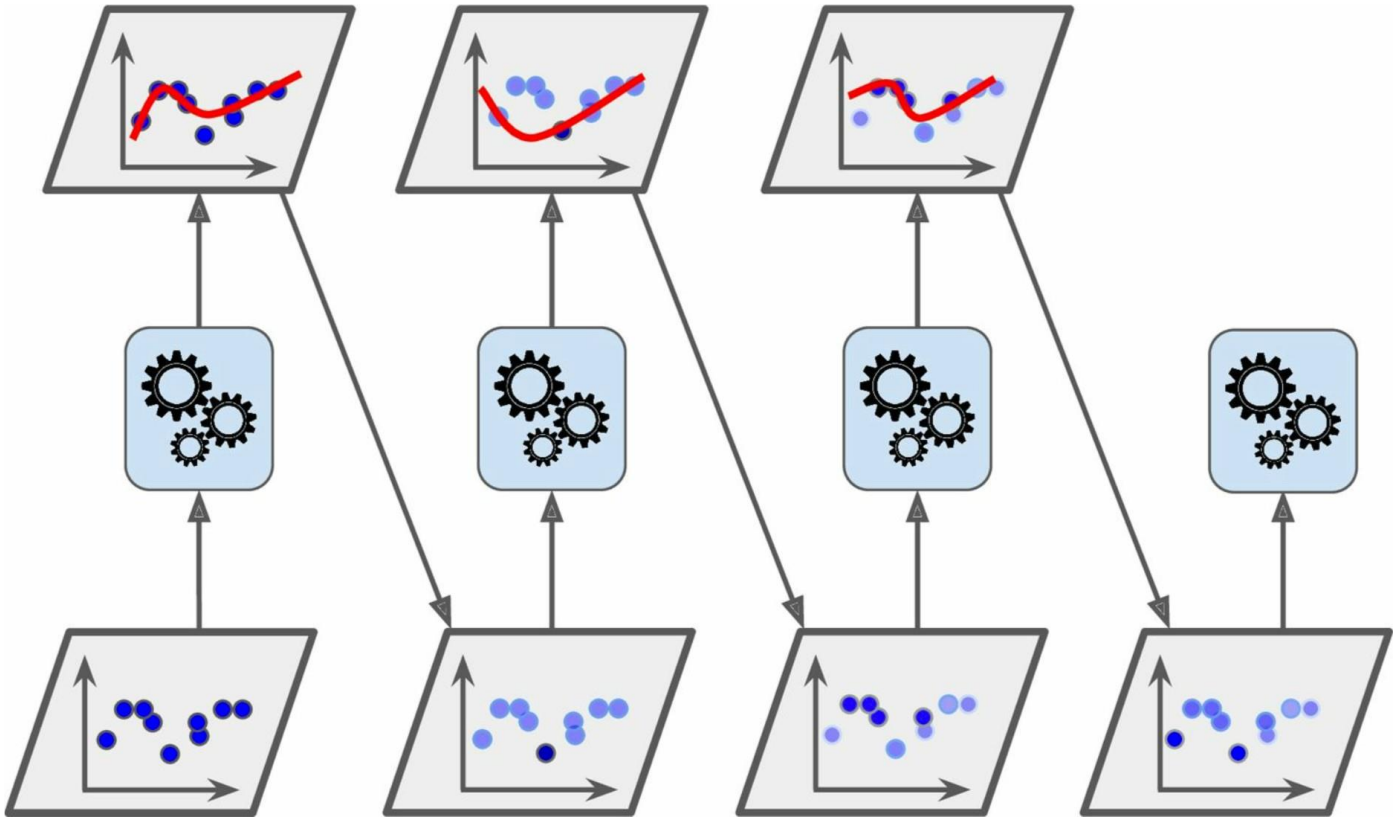


Figure from Hands-On Machine Learning with Scikit-learn and Tensorflow (Géron)

Boosting

- A benefit of bagging is that we can train the individual classifiers in parallel
- In boosting, we have to train the individuals sequentially
- Boosting typically uses fast and weak individual learners, so the sequential training phase might not be a problem
- Boosting can also be effective on some datasets
- You have to test and evaluate both to be sure which is the best for your specific task

XGBoost

- Another popular boosting method is Gradient Boosting
- Instead of weighting the instances as in AdaBoost, the subsequent individual classifiers are trained on the *residual errors* made by the previous individual
- XGBoost is a variant of Gradient Boosted Trees that have received a lot of attention lately
- It is fast and often very accurate on many datasets
- Scitkit-learn has an interface to the XGBoost library

Gradient Boosting

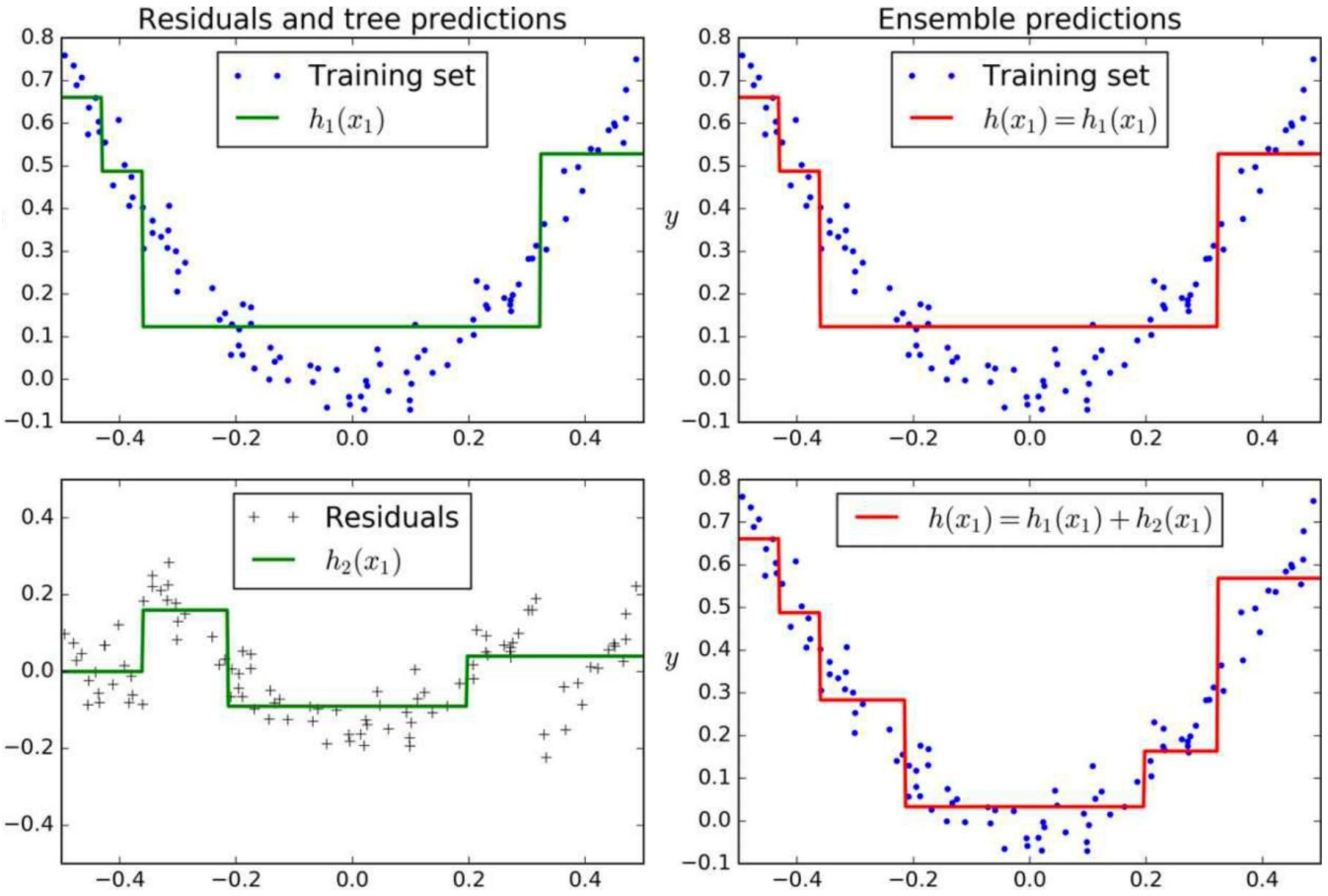


Figure from Hands-On Machine Learning with Scikit-learn and Tensorflow (Géron)

Stacking

- In the previous ensemble methods, we used voting when classifying an instance
- The idea behind stacking is to replace the simple voting with a classifier
- The classifier is trained on the outputs from the individuals
- The final “voting” classifier is usually called *blender* or *meta learner*

Stacking

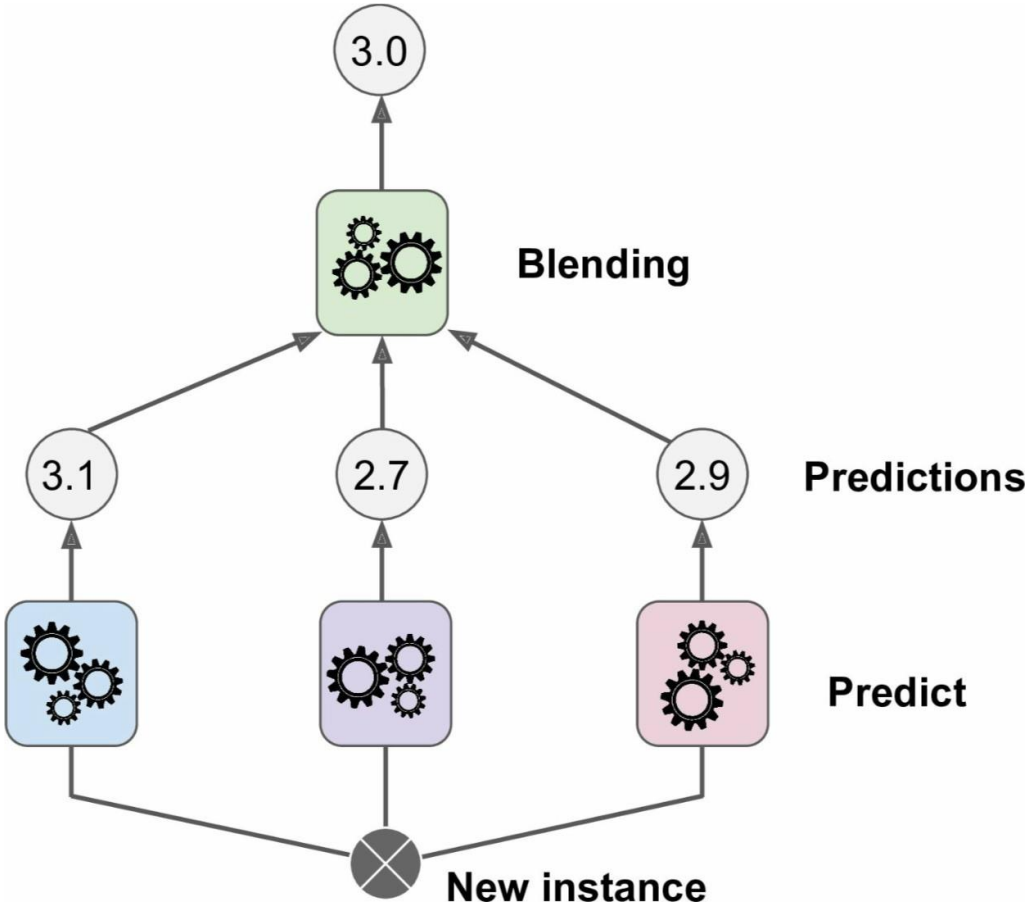


Figure from Hands-On Machine Learning with Scikit-learn and Tensorflow (Géron)