



UNIVERSIDADE  
CATÓLICA  
PORTUGUESA

BRAGA

# Machine Learning

Session 16 - T

## Tree-Based Models – Part 2

Ciência de Dados Aplicada

2023/2024

# Why are Decision Trees poor predictors?

- Decision trees tend to have **high variance**. A small change in the training data can produce big changes in the estimated Tree.
- To improve the performance:
  - **Pruning (last session)**: grow deep trees (small bias, high variance) which then are pruned into smaller ones (reduce variance);
  - **Ensemble methods (today's session)**: combine multiple simple trees.
    - Bagging and Random Forests
    - Boosted trees

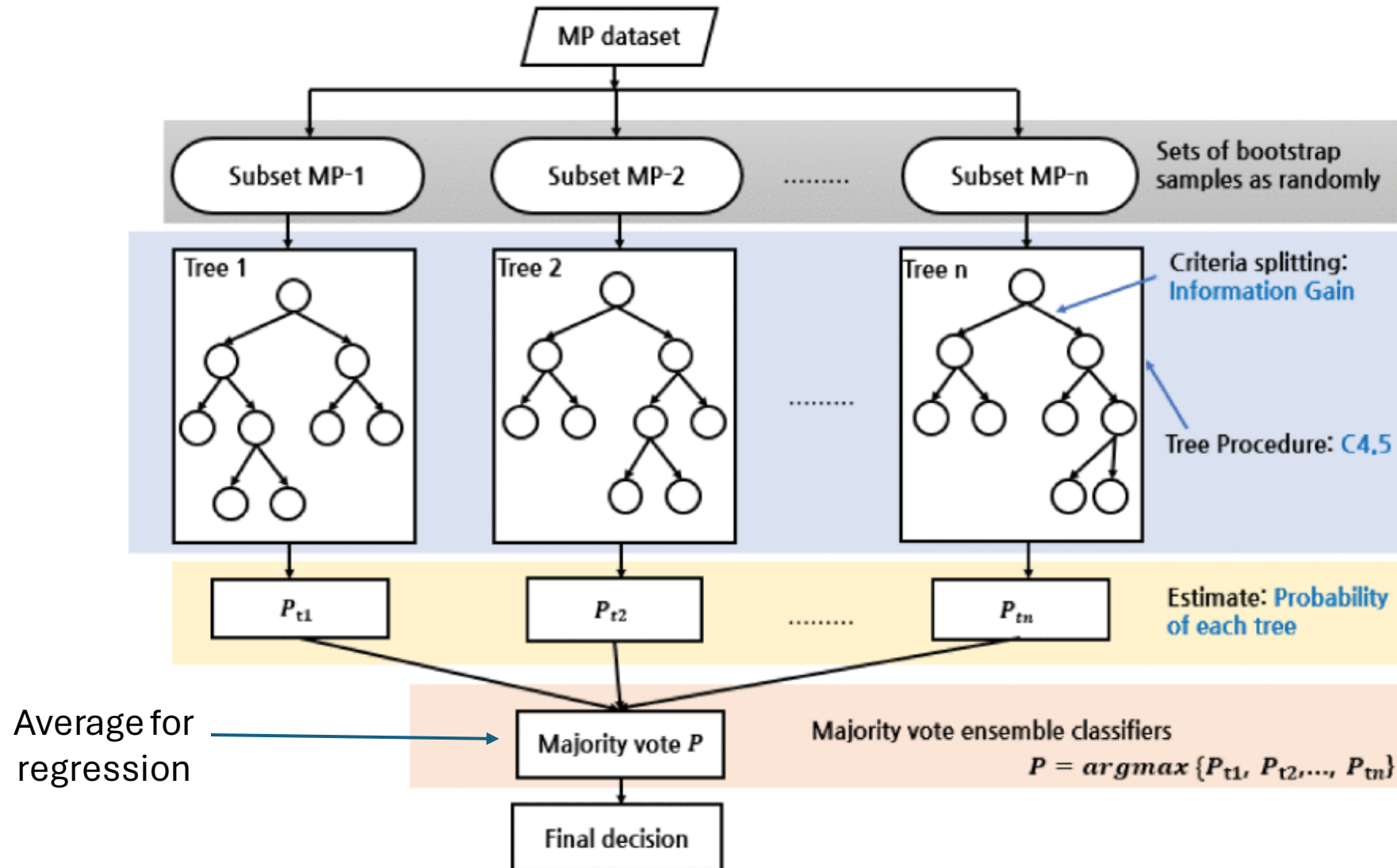
# Bagging

- Let's revisit the concept of **Cross-Validation** and why it performs better than the **Holdout** method!
- The Holdout method often overestimates the error and **provides highly different error estimates** if you alter the test split.
- On the other hand, K-fold Cross-Validation generates more **consistent error estimates by averaging across K distinct estimates of error** (one for each fold).
- **Bagging** (Bootstrap AGGregatING) operates with a similar objective: **reducing the variance** of a predictor prone to high variance by **averaging across multiple estimates**.

# Bagging

- In other words, averaging a set of observations reduces variance.
- **Obvious problem:** We only have access to one training dataset.
- **Solution:** Bootstrap the data!
  - Sample  $n$  times with replacement from the original training data;
  - Repeat  $B$  times to generate  $B$  "bootstrapped" training datasets.
- For each bootstrapped dataset train a decision tree.
- Averaging the predictions of each tree trained with different bootstrapped datasets is called bootstrap aggregation (Bagging).

# Bagging

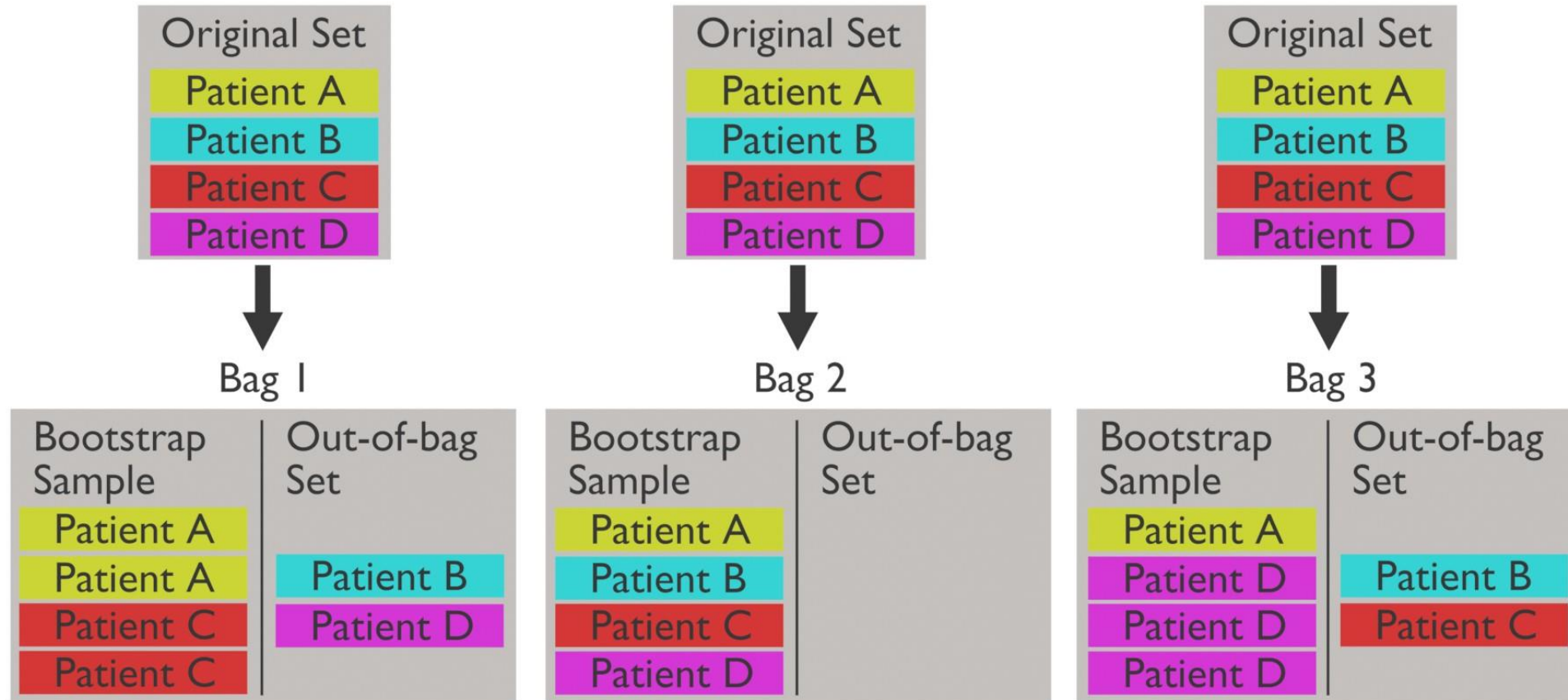


Le, T.-T.-H., Kang, H., & Kim, H. (2020). Household Appliance Classification Using Lower Odd-Numbered Harmonics and the Bagging Decision Tree. In IEEE Access (Vol. 8, pp. 55937–55952). Institute of Electrical and Electronics Engineers (IEEE). <https://doi.org/10.1109/access.2020.2981969>

# Out-of-Bag (OOB) Error Estimation

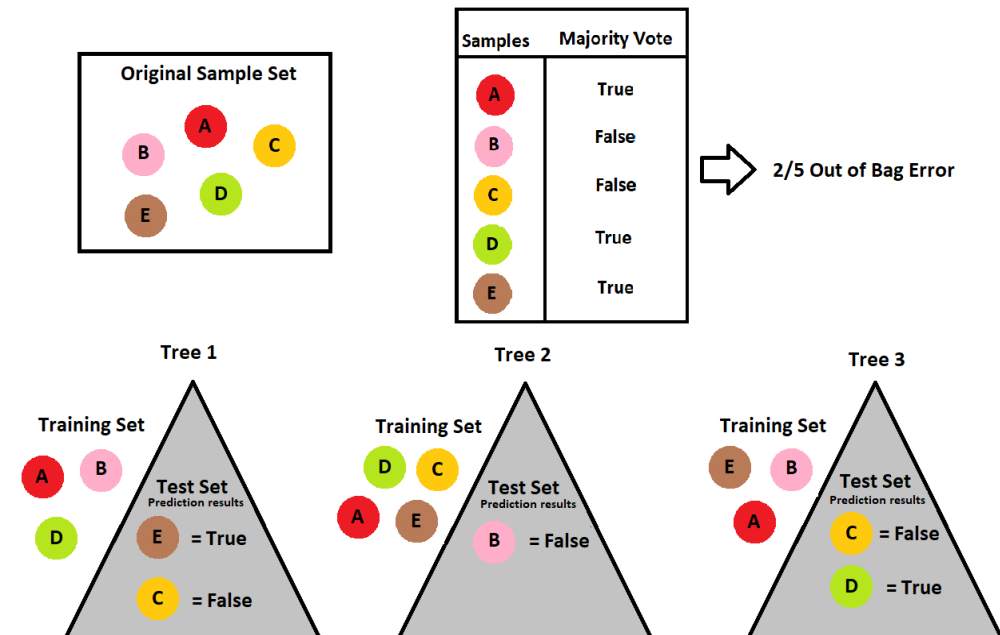
- **Not all** of the training points will appear in each sample;
- Each bootstrap sample contains roughly 63.2% of the observed data points;
- The remaining observations not used to fit a given bagged tree are called the **out-of-bag (OOB)** observations
- Another way of thinking about it: Each observation is OOB for roughly  $B/3$  of the trees. We can treat observation  $i$  as a test point each time it is OOB.

# Out-of-Bag (OOB) Error Estimation



# Out-of-Bag (OOB) Error Estimation

- To form the OOB estimate of test error:
  - Predict the response for the  $i$ th observation using each of the trees for which  $i$  was OOB. This gives us roughly  $B/3$  predictions for each observation;
  - Calculate the error of each OOB prediction;
  - Average all of the errors.

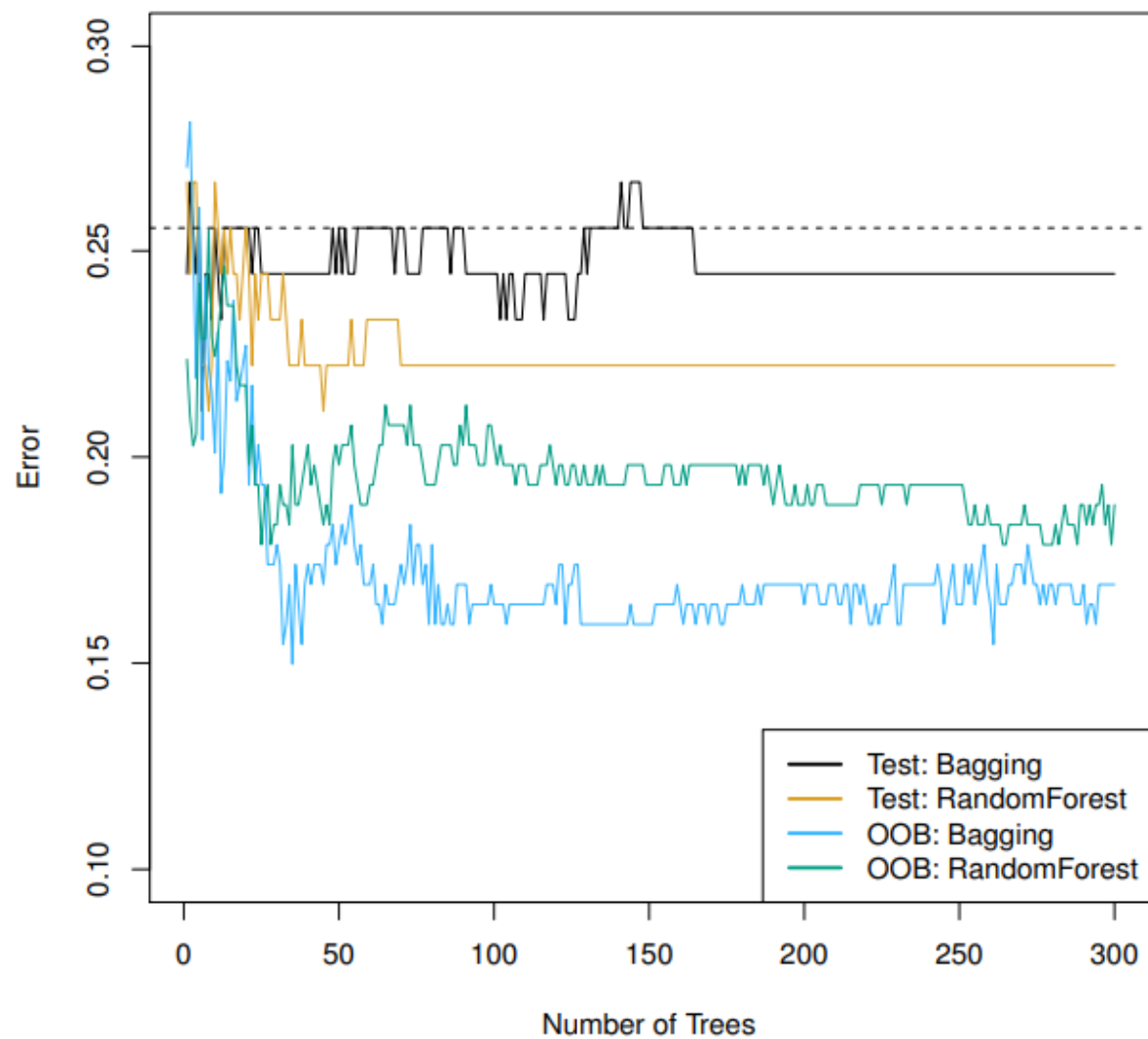




# Random Forests

- However, the **B bootstrapped dataset are correlated!**
  - the variance reduction due to averaging is diminished.
- **Random forests** provide an improvement over **bagged trees** by **de-correlate** the B trees by **randomly perturbing each tree**.
- A **random forest** is constructed by **bagging**, but for each split in each tree only a **random subset of features** are considered as splitting variables.
- Rule of thumb:  **$\sqrt{p}$  for classification trees** and  **$p/3$  for regression trees** (p is the number of features).

# Bagging vs Random Forests



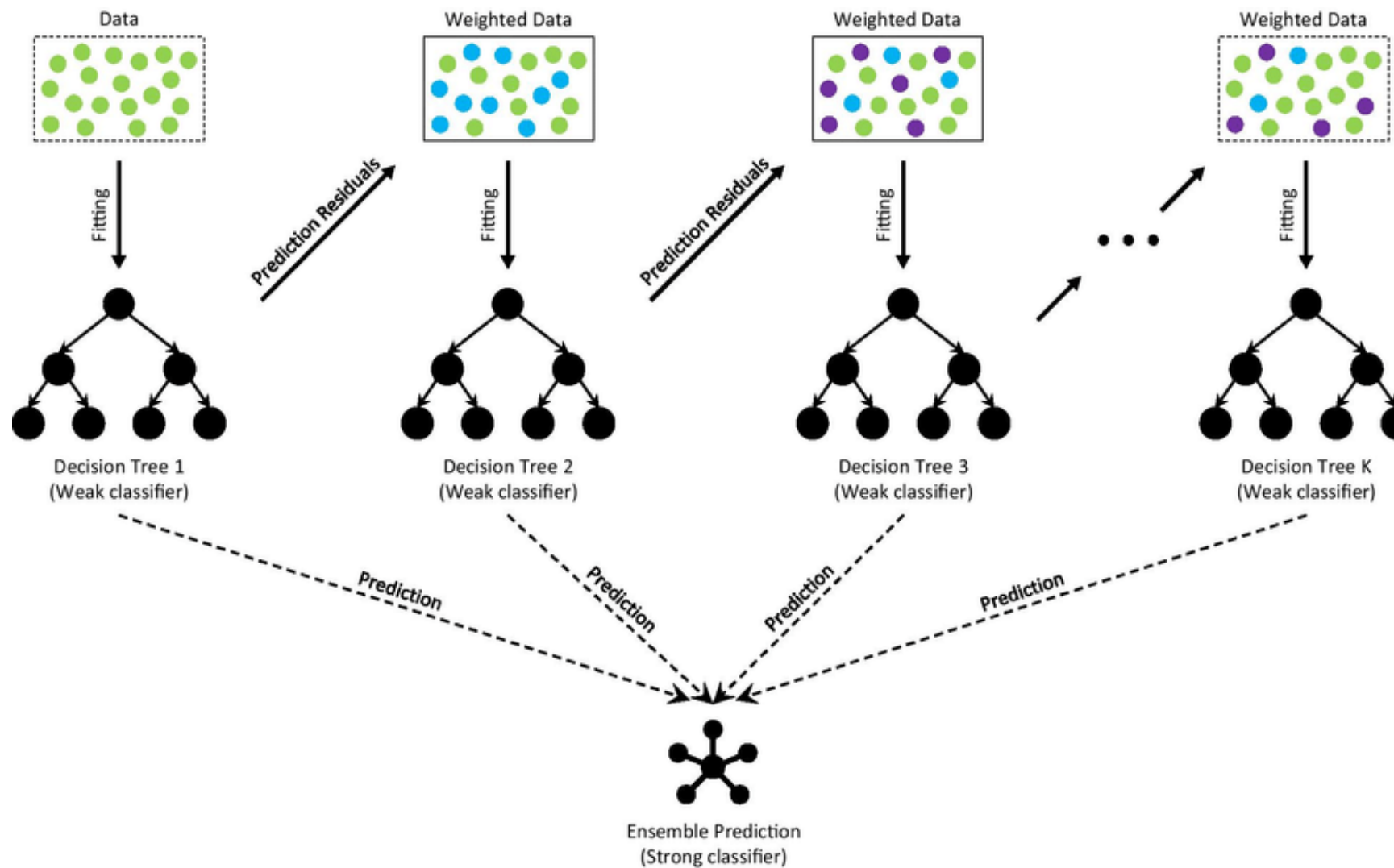
# Advantages of Random Forests

- **Reduced Overfitting:** Random forests mitigate overfitting by aggregating the predictions of multiple decision trees.
- **Improved Performance:** Random forests often provide higher performance compared to individual decision trees, especially for complex datasets.
- **Reduced Bias:** The ensemble nature of random forests reduces bias that may be present in individual decision trees.
- **Parallelization:** Training of individual decision trees within a random forest can be easily parallelized, leading to faster computation.

# Boosting





- **Boosting** works in a similar way to bagging, except that the **trees are grown sequentially**: each tree is grown using **information from previously grown trees**.
- Each subsequent tree focus on the **training examples that the previous ones got wrong**.
- To focus on specific examples, boosting uses a **weighted training set**.

# Boosting



- Given a base classifier, the key steps of **AdaBoost** are:
  - At each iteration, re-weight the training samples by assigning larger weights to samples that were classified incorrectly.
  - Train a new base classifier based on the re-weighted samples.
  - Add it to the ensemble of classifiers with an appropriate weight.
  - Repeat the process many times.
- Requirements for base classifier:
  - Needs to minimize weighted error.
  - Ensemble may get very large, so base classifier must be fast. It turns out that any so-called weak learner/classifier suffices (e.g. decision trees).
- Individually, **weak learners** may have **high bias** (underfit). By making each classifier focus on previous mistakes, AdaBoost **reduces bias**.

# Bagging and Boosting Recap

- Bagging and boosting are **ensemble learning methods** that can improve performance;
- **Bagging:** 
  - Reduces variance;
  - Bias is not changed (much);
  - Parallel; 
  - Want to minimize correlation between ensemble elements.
- **Boosting:** 
  - Reduces bias;
  - Increases variance;
  - Sequential; 
  - High dependency between ensemble elements.

# Resources

- Collins, R. (2018). Machine learning with bagging and boosting. Independently Published.
- [https://www.youtube.com/watch?v=J4Wdy0Wc\\_xQ](https://www.youtube.com/watch?v=J4Wdy0Wc_xQ)
- <https://www.youtube.com/watch?v=sQ870aTKqiM>
- <https://www.youtube.com/watch?v=Xz0x-8-cgaQ>
- <https://www.youtube.com/watch?v=tjy0yL1rRRU>