MACHINE LEARNING USING R

Class 5

Topics

- Meta Learning
  - Bagging
  - Boosting
  - Deep dive into Random Forests
- Evaluating Model Performance
  - Out of Bag error
  - Confusing Matrix
  - Accuracy & Error Rate
  - Sensitivity & Specificity
  - Precision & Recall
  - F-measure
- Visualizing performance tradeoffs
  - ROC & AUC

Recap

- When we build Decision Trees if we split the training data into two parts at random, and fit a decision tree to both halves, the results that we get could be quite different (High variance!)
- If new data is added, it is possible that our decision tree will change to fit to this new data (Overfitting!)

We need to find solutions to be able to overcome these challenges.

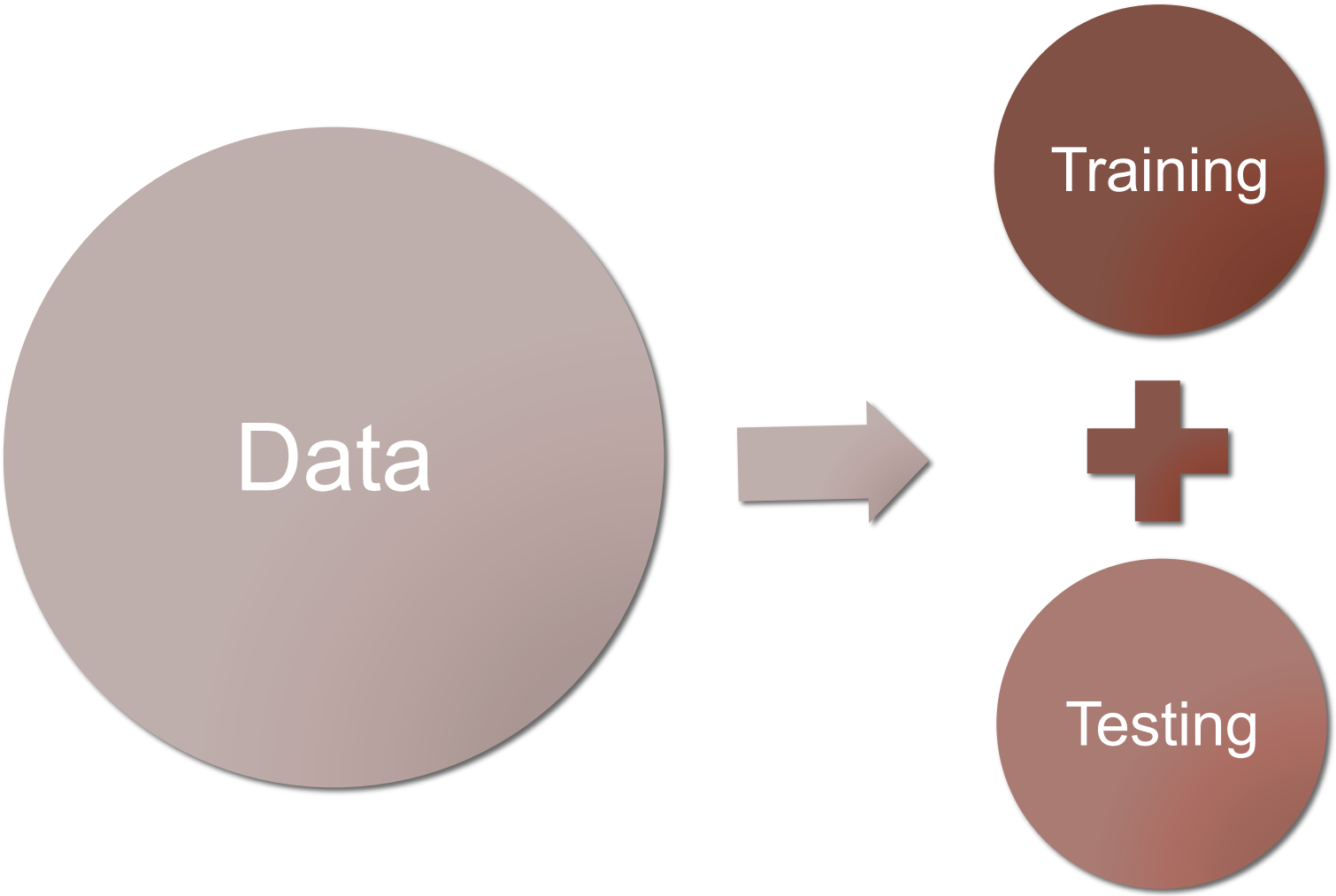
Meta-Learning

- The **allocation function** dictates whether each model receives the full training dataset or merely a sample. Since the ideal ensemble includes a diverse set of models, the allocation function could increase diversity by artificially varying the input data to train a variety of learners.
- The **combination function** governs how disagreements among the predictions are reconciled. For example, the ensemble might use a majority vote to determine the final prediction, or it could use a more complex strategy such as weighting each model's votes based on its prior performance.
- This process of using the predictions of several models to train a final arbiter model is known as **stacking**.

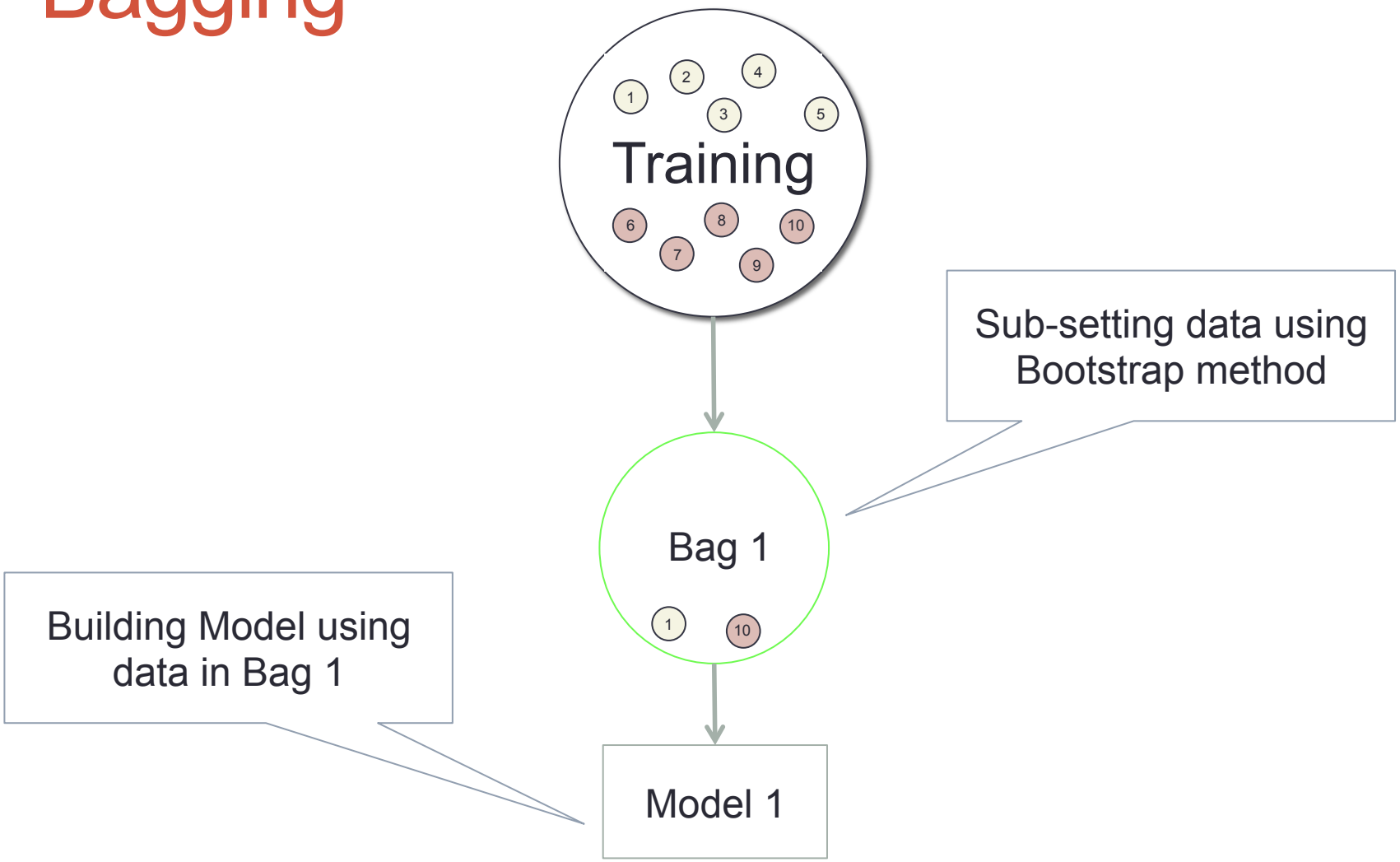
Bagging

- Also known as **B**ootstrap **Agg**regat**ing**
- Takes multiple unstable learners and creates a strong learner
- Usually applied to Decision Trees but can be used for any other method
- Trees are grown in parallel
- Bagging improves prediction accuracy at the expense of interpretability.

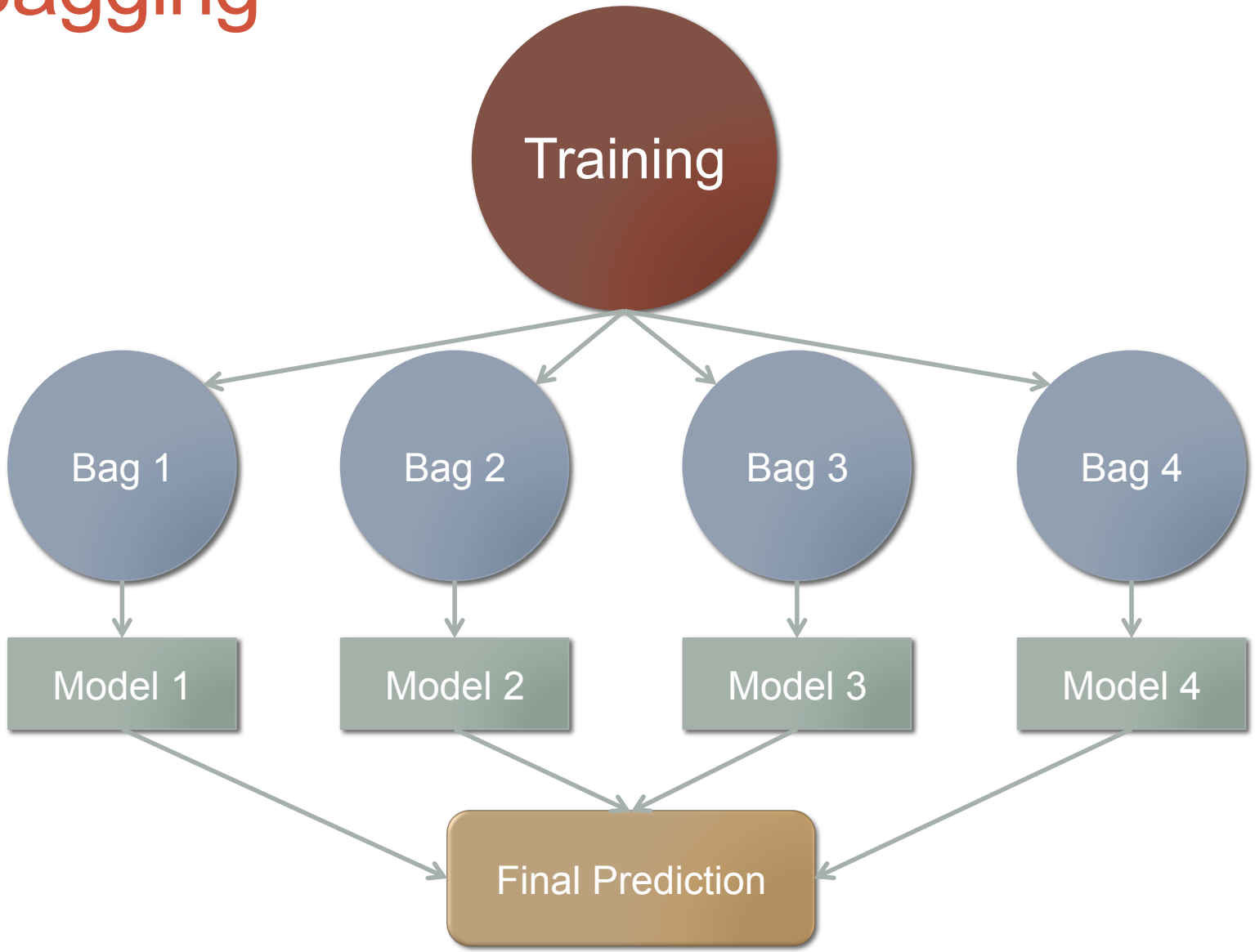
Bagging



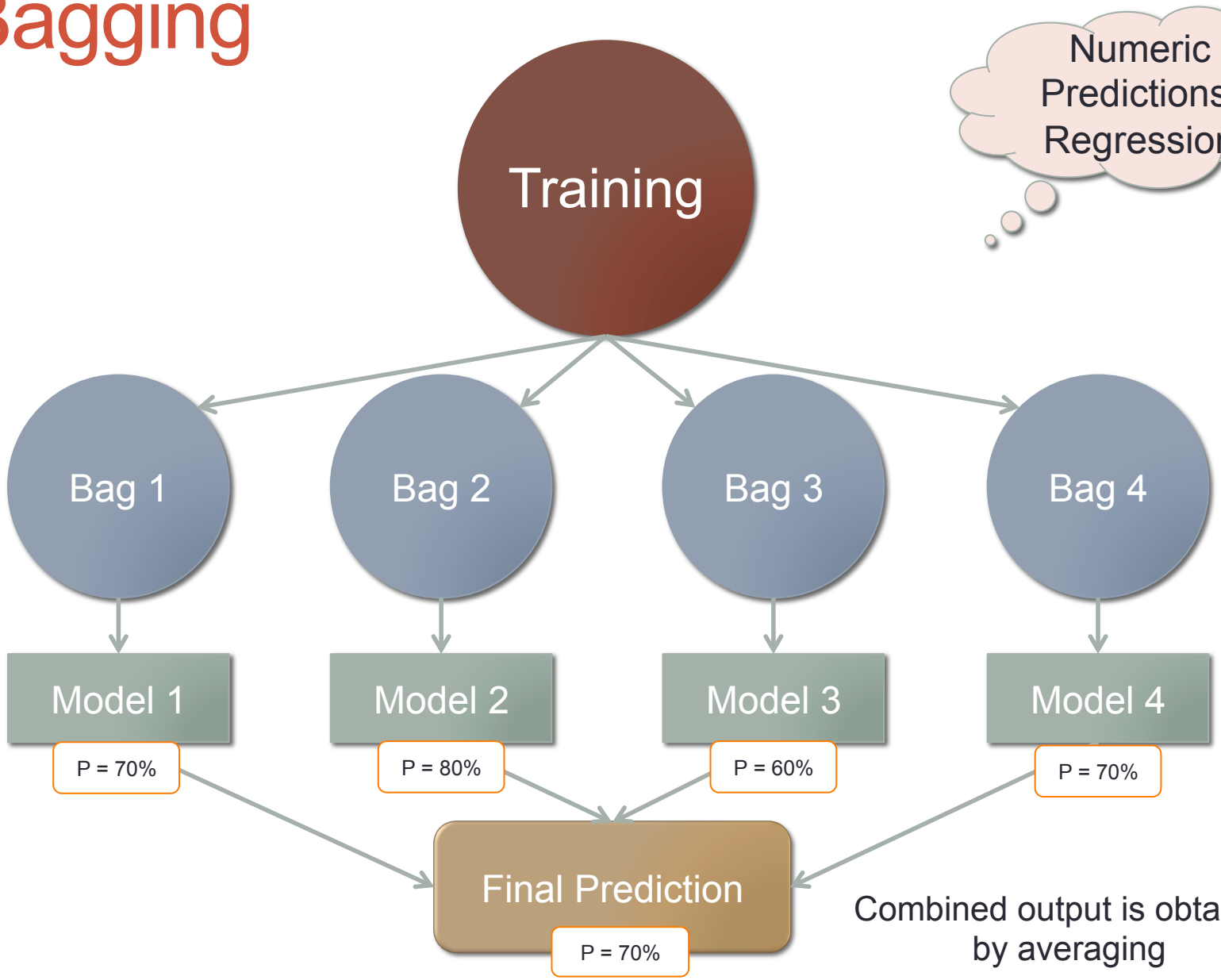
Bagging


1
2
3
4
5
Training
6
7
8
9
10
Sub-setting data using Bootstrap method
Bag 1
1
10
Building Model using data in Bag 1
Model 1

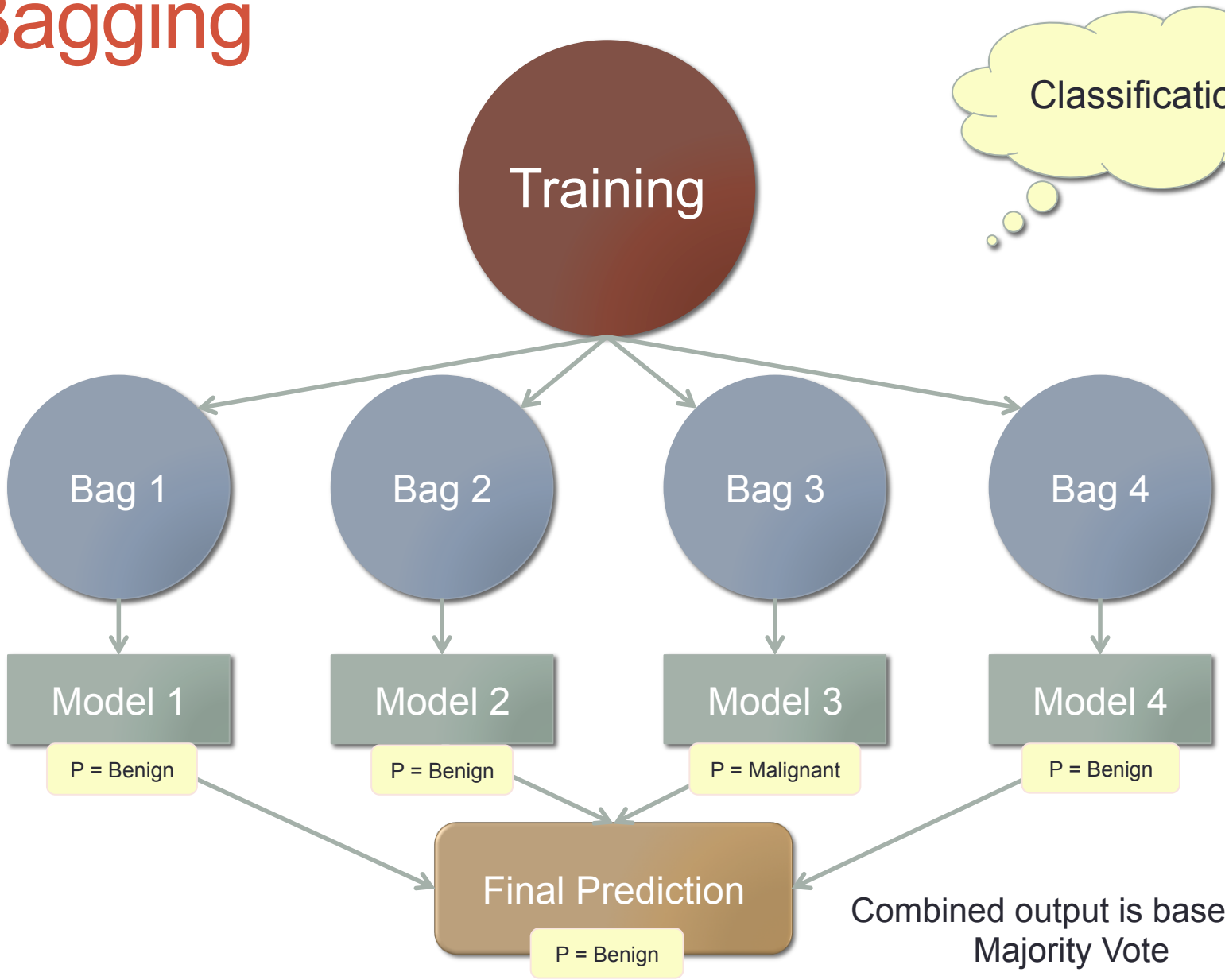
Bagging


Training
Bag 1
Bag 2
Bag 3
Bag 4
Model 1
Model 2
Model 3
Model 4
Final Prediction

Bagging


Numeric Predictions/ Regression
Training
Bag 1
Bag 2
Bag 3
Bag 4
Model 1
Model 2
Model 3
Model 4
P = 70%
P = 80%
P = 60%
P = 70%
Final Prediction
P = 70%
Combined output is obtained by averaging

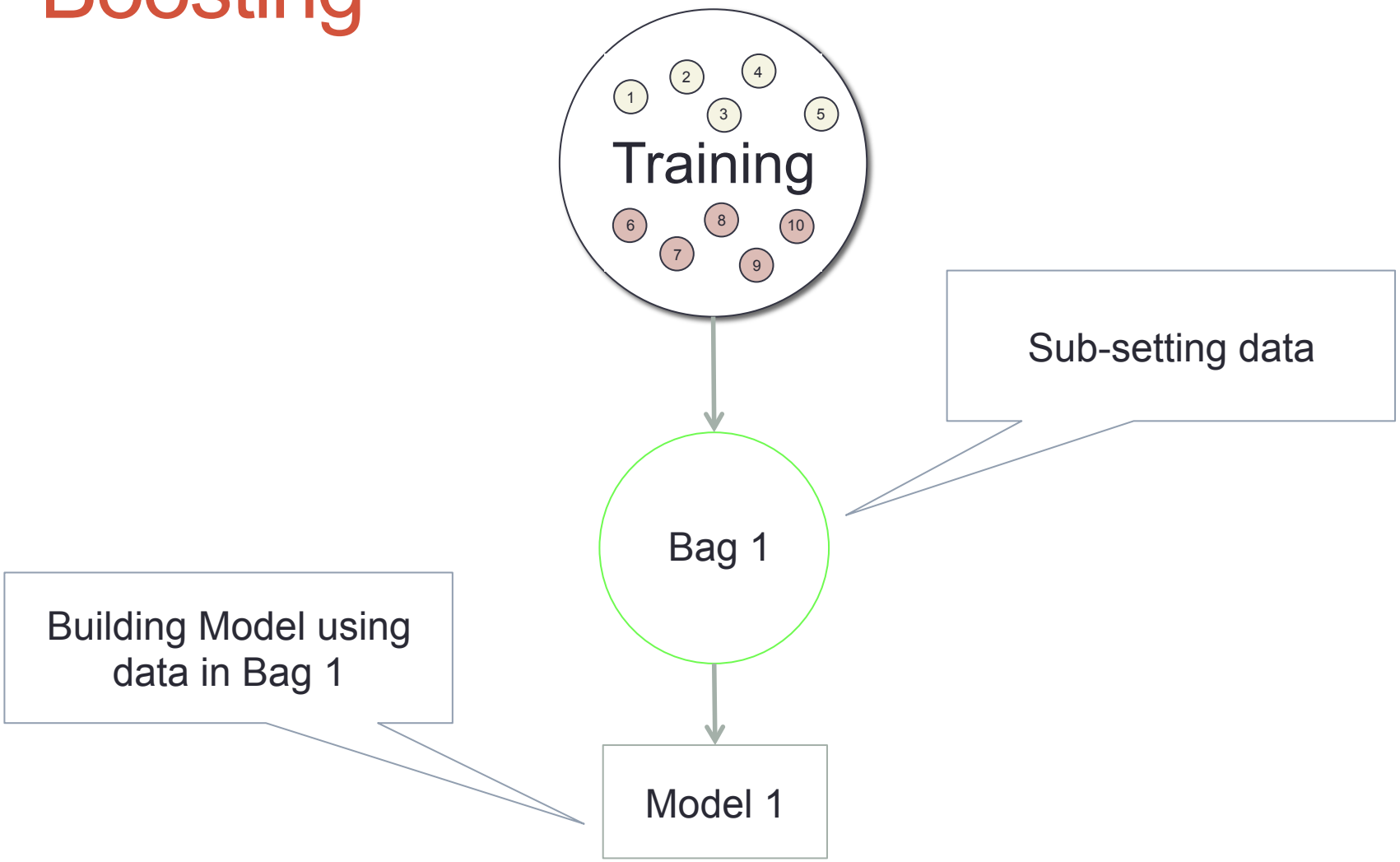
Bagging


Classification
Training
Bag 1
Bag 2
Bag 3
Bag 4
Model 1
Model 2
Model 3
Model 4
P = Benign
P = Benign
P = Malignant
P = Benign
Final Prediction
P = Benign
Combined output is based on Majority Vote

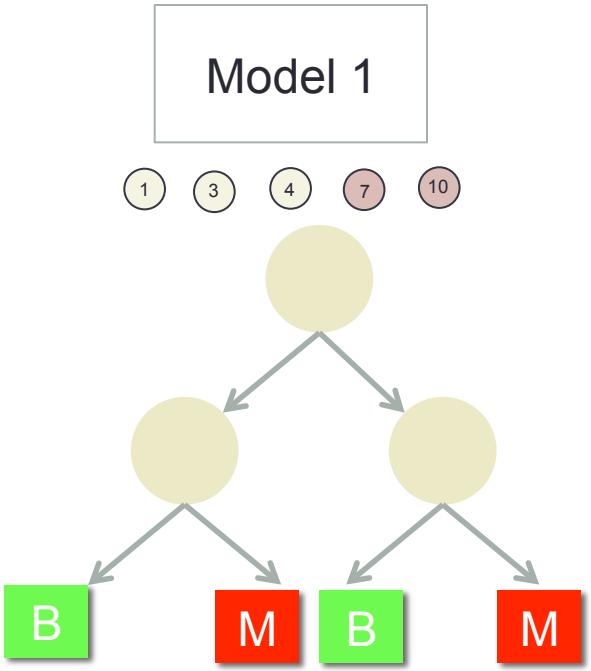
Boosting

- Takes multiple weak learners and creates a strong learner
- Does not use Bootstrap sampling
- Usually applied to Decision Trees but can be used for any other method
- Trees are grown sequentially based on the previous trees

Boosting


1
2
3
4
5
Training
6
7
8
9
10
Sub-setting data
Bag 1
Building Model using data in Bag 1
Model 1

Boosting

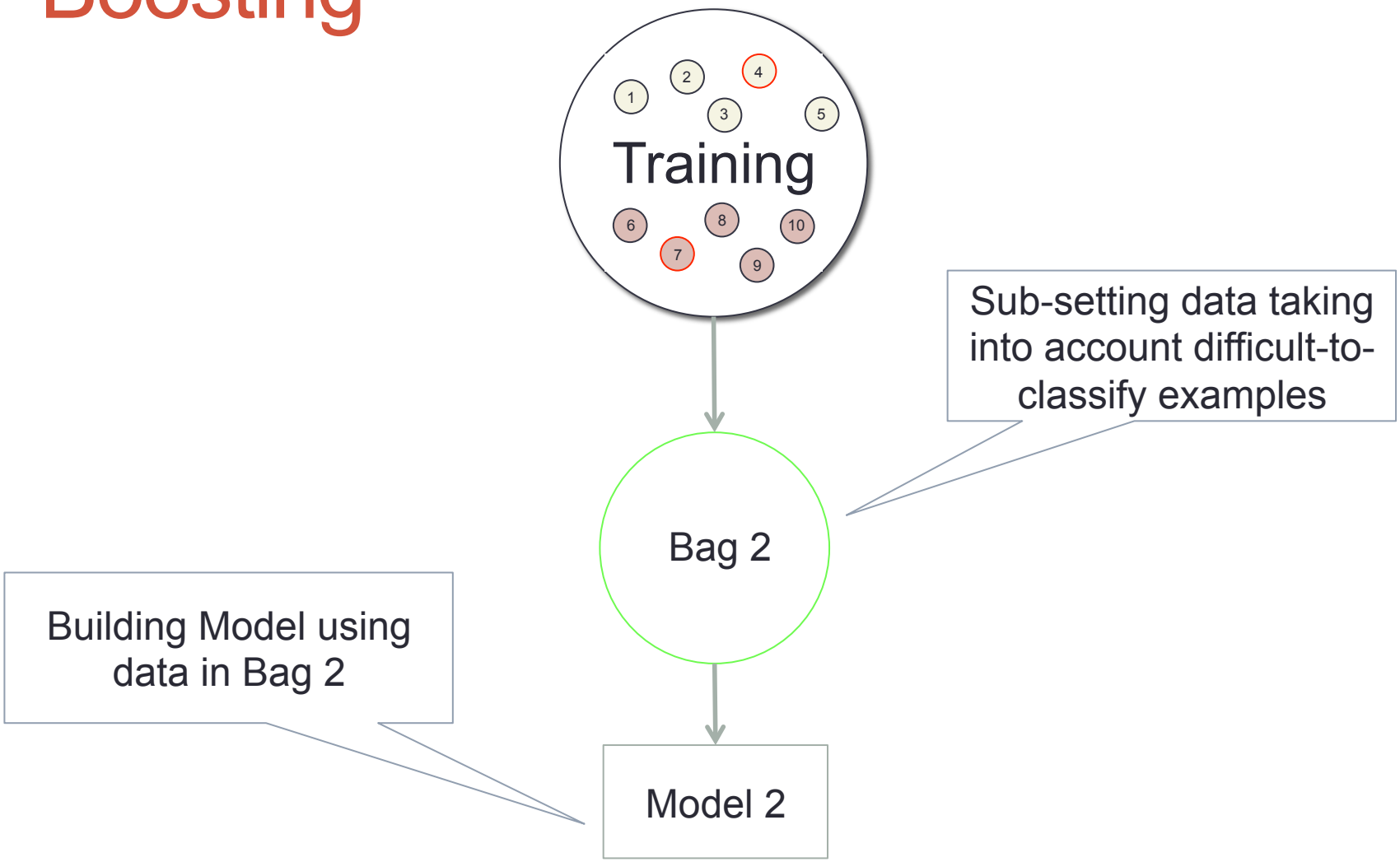


Beginning from an unweighted dataset, the first classifier attempts to model the outcome.

Examples that the classifier predicted correctly will be less likely to appear in the training dataset for the following classifier, and conversely, the difficult-to-classify examples will appear more frequently.

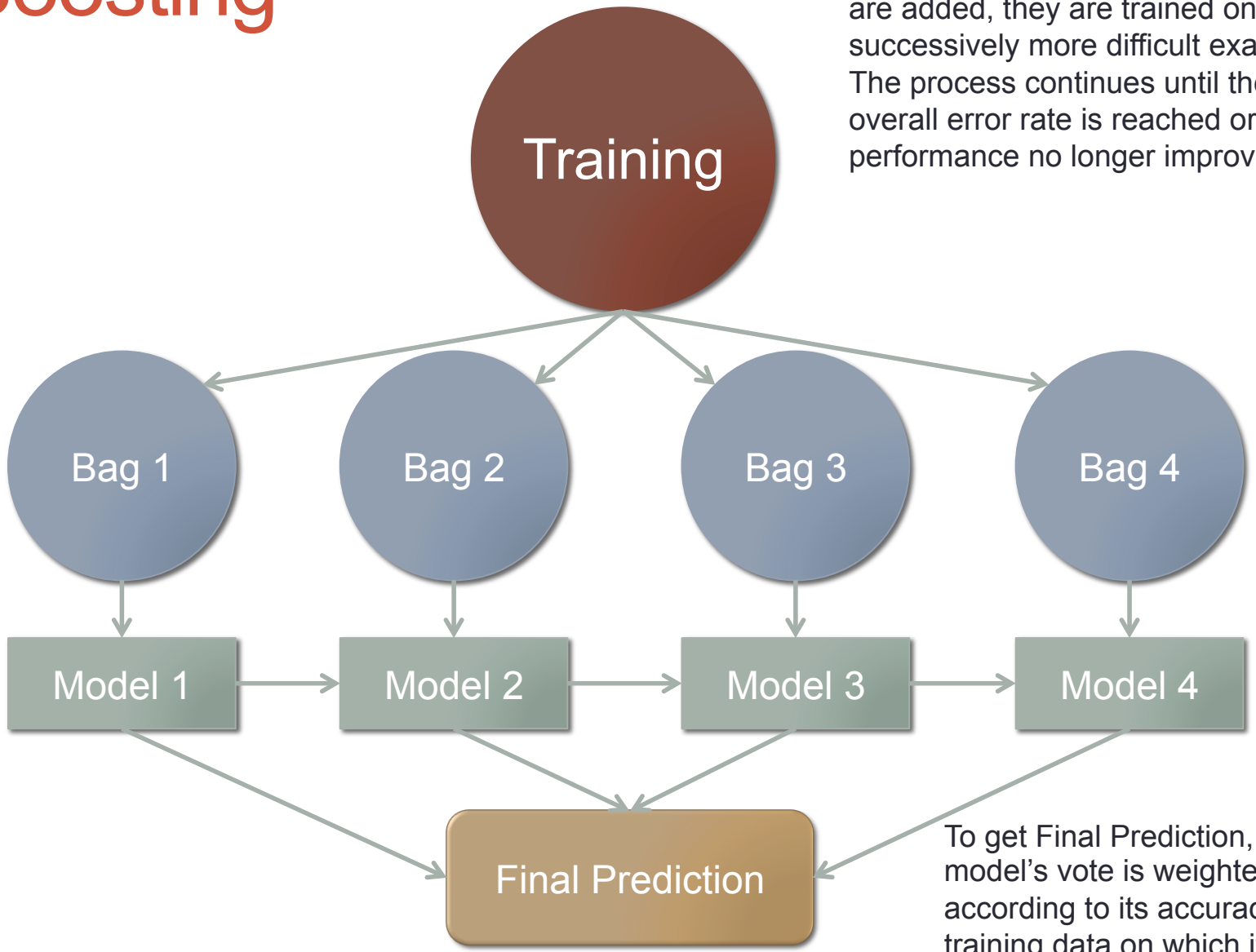
Examples 4 and 7 are likely to appear in the training dataset for Model 2.

Boosting


1
2
3
4
5
Training
6
7
8
9
10
Sub-setting data taking into account difficult-to-classify examples
Bag 2
Building Model using data in Bag 2
Model 2

Boosting

As additional rounds of weak learners are added, they are trained on data with successively more difficult examples. The process continues until the desired overall error rate is reached or performance no longer improves.

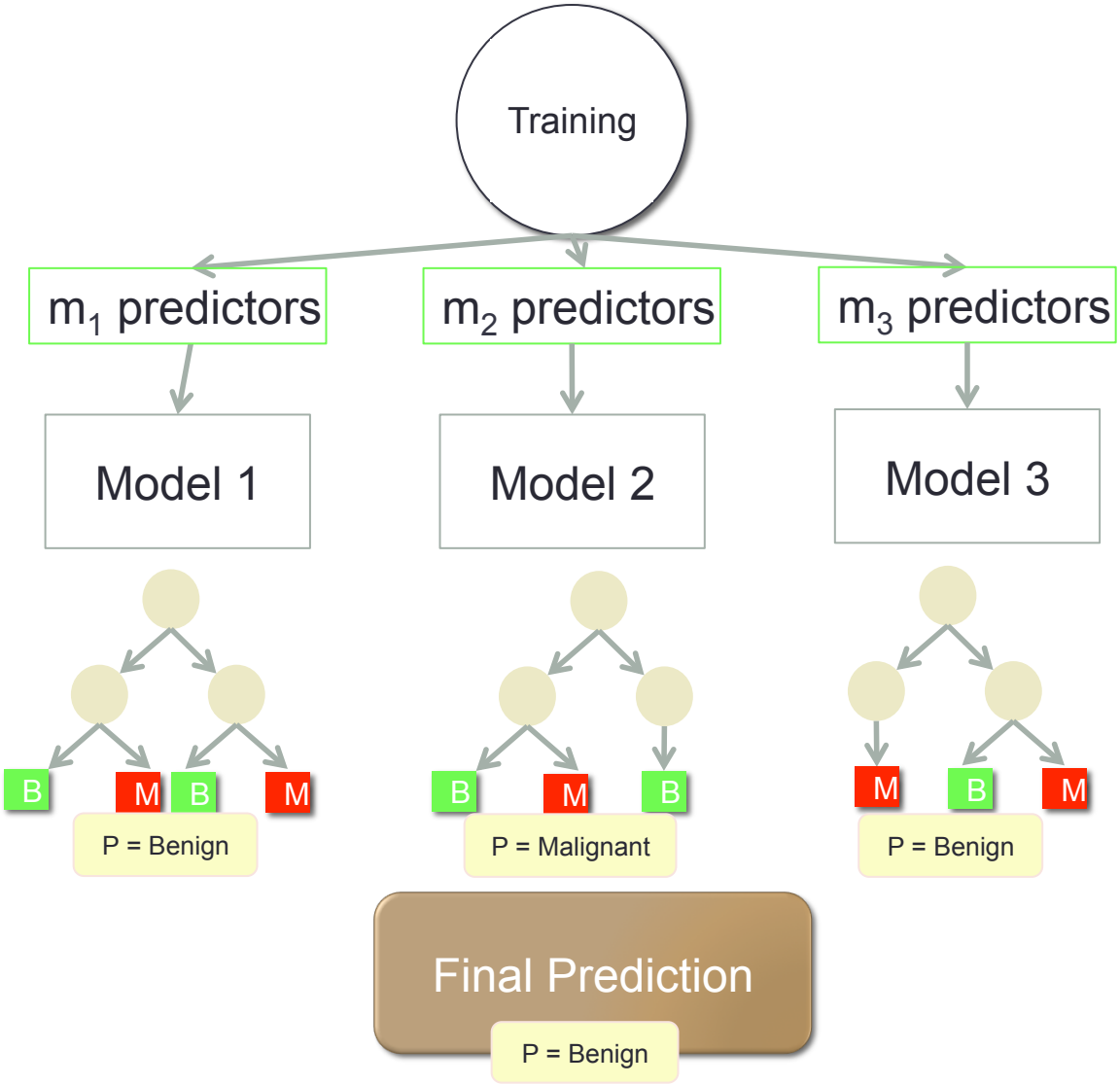


To get Final Prediction, each model's vote is weighted according to its accuracy on the training data on which it was built.

Random Forests

- We create decorrelated trees
- At each split, a random sample of predictors is chosen as split candidates from the full set of predictors
- Number of predictors considered at each split (m) is approximately equal to the square root of the total number of predictors (p), $m = \sqrt{p}$.
- This way strong predictors do not dictate the trees that get generated

Random Forest



Random Forest

```
> rf
Call:
       randomForest( formula = default ~ ., data = credit)
                Type of random forest: classification
                Number of trees: 500
No. of variables tried at each split: 4
                OOB estimate of error
Confusion matrix:
       no     yes    class.error
no     640    60     0.08571429
yes    178    122    0.59333333
```

Formula used to create the forest

Type:

Number of tress

Number of predictors evaluated. This can be specified in formula

Random Forests: Advantages

- An all-purpose model that performs well on most problems
- Can handle noisy or missing data; categorical or continuous features
- Selects only the most important features
- Can be used on data with an extremely large number of features or examples

[Lantz, Brett. Machine Learning with R. Packt Publishing]

Random Forests: Disadvantages

- Unlike a decision tree, the model is not easily interpretable
- May require some work to tune the model to the data

[Lantz, Brett. Machine Learning with R. Packt Publishing]

EVALUATING MODEL PERFORMANCE

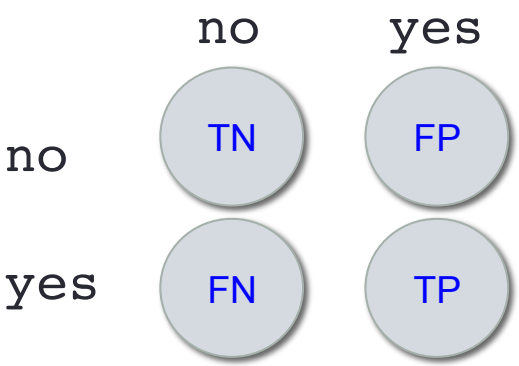
Out-of Bag Error

- Unbiased estimate of test error of an example using only the trees that were not fit using that example.
- The confusion matrix reflects the out-of-bag error rate.
- Any example not selected for a single tree's bootstrap sample can be used as a way to test the model's performance on unseen data. At the end of the forest construction, the predictions for each example each time it was held out are tallied, and a vote is taken to determine the final prediction for the example.
- The total error rate of such predictions becomes the out-of-bag error rate.

Confusion Matrix

The relationship between positive class and negative class predictions can be depicted as a 2 x 2 confusion matrix that tabulates whether predictions fall into one of four categories:

```
Confusion matrix:
```



- True Positive (TP): Correctly classified as the class of interest
- True Negative (TN): Correctly classified as not the class of interest
- False Positive (FP): Incorrectly classified as the class of interest
- False Negative (FN): Incorrectly classified as not the class of interest

Accuracy

Also knows as Success Rate

$$accuracy = \frac{TP+TN}{TP+TN+FP+FN} = \frac{122+640}{122+640+60+178} = 0.7635$$

Error Rate

Also *1 - accuracy*

$$errorrate = \frac{FP+FN}{TP+TN+FP+FN} = \frac{60+178}{122+640+60+178} = 0.2385$$

Sensitivity and Specificity

- Classification often involves a balance between being overly conservative and overly aggressive in decision making. This tradeoff is captured by a pair of measures: sensitivity and specificity.
- Sensitivity and specificity range from 0 to 1, with values close to 1 being more desirable. Of course, it is important to find an appropriate balance between the two — a task that is often quite context-specific.

Sensitivity

The sensitivity of a model (also called the true positive rate), measures the proportion of positive examples that were correctly classified.

$$sensitivity = \frac{TP}{TP + FN} = \frac{122}{122 + 178} = 0.4067$$

Specificity

The specificity of a model (also called the true negative rate), measures the proportion of negative examples that were correctly classified

$$specificity = \frac{TN}{TN+FP} = \frac{640}{640+60} = 0.9143$$

Precision and Recall

- Used primarily in the context of information retrieval, these statistics are intended to provide an indication of how interesting and relevant a model's results are, or whether the predictions are diluted by meaningless noise.

Precision

- The precision (also known as the positive predictive value) is defined as the proportion of positive examples that are truly positive; in other words, when a model predicts the positive class, how often is it correct?

$$precision = \frac{TP}{TP + FP} = \frac{122}{122 + 60} = 0.6703$$

Recall

- Measure of how complete the results are.
- The number of true positives over the total number of positives. This is the same as sensitivity, only the interpretation differs.

$$recall = \frac{TP}{TP + FN} = \frac{122}{122 + 178} = 0.4067$$

- A model with high recall captures a large portion of the positive examples, meaning that it has wide breadth. For example, a search engine with high recall returns a large number of documents pertinent to the search query.

F-measure

- Also known as F1 score or F-score
- The F-measure combines precision and recall using the harmonic mean.
- The harmonic mean is used rather than the more common arithmetic mean since both precision and recall are expressed as proportions between zero and one.
- The following is the formula for F-measure:

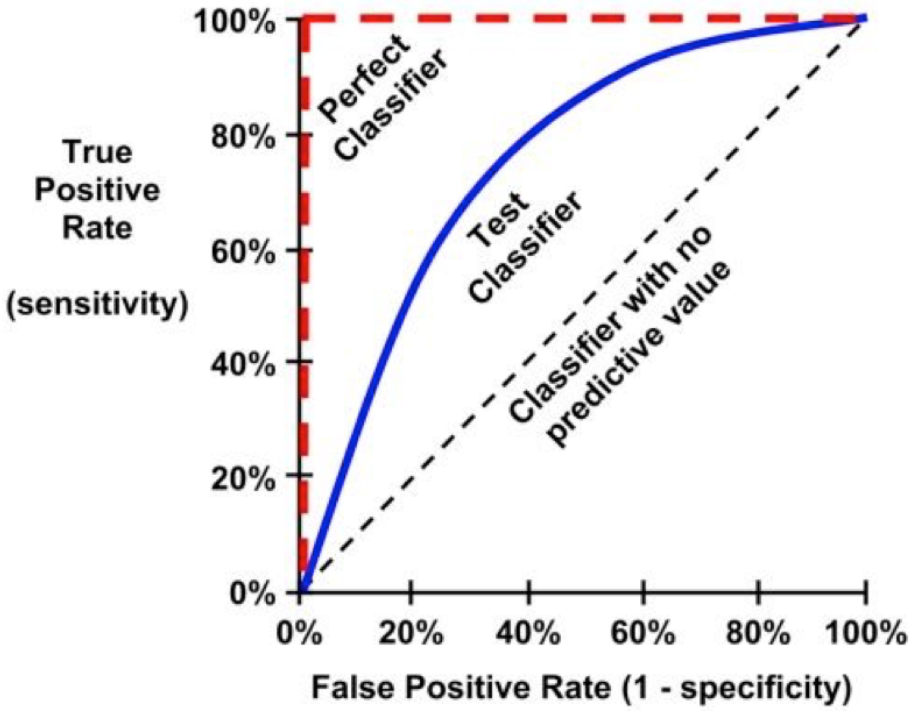
$$F-measure = \frac{2 \times precision \times recall}{recall + precision} = \frac{2 \times TP}{2 \times TP + FP + FN} = 0.5701$$

- Allows to compare models side by side provided precision and recall are assigned equal weights.
- A better practice is to use measures such as the F-score in combination with methods that consider a model's strengths and weaknesses more globally

VISUALIZING PERFORMANCE TRADEOFFS

ROC Curve

- The ROC curve (Receiver Operating Characteristic) or sensitivity/ specificity plot is used to examine the tradeoff between the detection of true positives, while avoiding the false positives.



Area under ROC Curve (AUC)

- Total area under the ROC curve
- AUC ranges from 0.5 (for a classifier with no predictive value), to 1.0 (for a perfect classifier).
- Convention for interpreting AUC scores uses a system similar to academic letter grades:
  - 0.9 – 1.0 = A (outstanding)
  - 0.8 – 0.9 = B (excellent/ good)
  - 0.7 – 0.8 = C (acceptable/ fair)
  - 0.6 – 0.7 = D (poor)
  - 0.5 – 0.6 = F (no discrimination)
- Two ROC curves may be shaped very differently, yet have identical AUC. For this reason, AUC can be extremely misleading. The best practice is to use AUC in combination with qualitative examination of the ROC curve.