W4. Bagging and Boosting

Guang Cheng

University of California, Los Angeles

guangcheng@ucla.edu

Week 4

Motivation for ensemble models

- When you want to purchase a new car, will you walk up to the first car shop and purchase one based on the advice of the dealer? It's highly unlikely.

Motivation for ensemble models

- When you want to purchase a new car, will you walk up to the first car shop and purchase one based on the advice of the dealer? It's highly unlikely.
- You would likely browser a few web portals where people have posted their reviews and compare different car models, checking for their features and prices.

Motivation for ensemble models

- When you want to purchase a new car, will you walk up to the first car shop and purchase one based on the advice of the dealer? It's highly unlikely.
- You would likely browser a few web portals where people have posted their reviews and compare different car models, checking for their features and prices.
- You will also probably ask your friends and colleagues for their opinion. In short, you wouldn't directly reach a conclusion, but will instead make a decision considering the opinions of other people as well.

Motivation for ensemble models

- Ensemble models in machine learning operate on a similar idea. They combine the decisions from multiple models to improve the overall performance. This can be achieved in various ways, which you will discover in this class.

What is Ensemble Learning?

- Let's see the following example.

What is Ensemble Learning?

- Let's see the following example.
- Suppose you are a movie director and you have created a short movie on a very important and interesting topic. Now, you want to take preliminary feedback (ratings) on the movie before making it public. What are the possible ways by which you can do that?

What is Ensemble Learning?

- **A**: You may ask one of your friends to rate the movie for you. Now it's entirely possible that the person you have chosen loves you very much and doesn't want to break your heart by providing a 1-star rating to the horrible work you have created.

What is Ensemble Learning?

- **A**: You may ask one of your friends to rate the movie for you. Now it's entirely possible that the person you have chosen loves you very much and doesn't want to break your heart by providing a 1-star rating to the horrible work you have created.
- **B**: Another way could be by asking 5 colleagues of yours to rate the movie. This should provide a better idea of the movie. This method may provide honest ratings for your movie. But a problem still exists. These 5 people may not be "Subject Matter Experts" on the topic of your movie. Sure, they might understand the cinematography, the shots, or the audio, but at the same time may not be the best judges of dark humour.

What is Ensemble Learning?

- **A**: You may ask one of your friends to rate the movie for you. Now it's entirely possible that the person you have chosen loves you very much and doesn't want to break your heart by providing a 1-star rating to the horrible work you have created.
- **B**: Another way could be by asking 5 colleagues of yours to rate the movie. This should provide a better idea of the movie. This method may provide honest ratings for your movie. But a problem still exists. These 5 people may not be "Subject Matter Experts" on the topic of your movie. Sure, they might understand the cinematography, the shots, or the audio, but at the same time may not be the best judges of dark humour.
- **C**: How about asking 50 people to rate the movie? Some of which can be your friends, some of them can be your colleagues and some may even be total strangers.

What is Ensemble Learning?

- The responses, in this case, would be more generalized and diversified since now you have people with different sets of skills. And as it turns out – this is a better approach to get honest ratings than the previous cases we saw.

What is Ensemble Learning?

- The responses, in this case, would be more generalized and diversified since now you have people with different sets of skills. And as it turns out – this is a better approach to get honest ratings than the previous cases we saw.
- With these examples, you can infer that a diverse group of people are likely to make better decisions as compared to individuals. Similar is true for a diverse set of models in comparison to single models. This diversification in Machine Learning is achieved by a technique called Ensemble Learning.

What is Ensemble Learning?

- The responses, in this case, would be more generalized and diversified since now you have people with different sets of skills. And as it turns out – this is a better approach to get honest ratings than the previous cases we saw.
- With these examples, you can infer that a diverse group of people are likely to make better decisions as compared to individuals. Similar is true for a diverse set of models in comparison to single models. This diversification in Machine Learning is achieved by a technique called Ensemble Learning.
- Now that you have got a gist of what ensemble learning is – let us look at the various techniques in ensemble learning.

Simple Ensemble Techniques

- In this section, we will look at a few simple but powerful techniques, namely:

Simple Ensemble Techniques

- In this section, we will look at a few simple but powerful techniques, namely:
  - Max Voting

Simple Ensemble Techniques

- In this section, we will look at a few simple but powerful techniques, namely:
  - Max Voting
  - Averaging

Simple Ensemble Techniques

- In this section, we will look at a few simple but powerful techniques, namely:
  - Max Voting
  - Averaging
  - Weighted Averaging

Max (Majority) Voting

- The max voting method is generally used for classification problems. In this technique, multiple models are used to make predictions for each data point. The predictions by each model are considered as a 'vote'. The predictions which we get from the majority of the models are used as the final prediction.

Max (Majority) Voting

- The max voting method is generally used for classification problems. In this technique, multiple models are used to make predictions for each data point. The predictions by each model are considered as a 'vote'. The predictions which we get from the majority of the models are used as the final prediction.
- For example, when you asked 5 of your colleagues to rate your movie (out of 5); we'll assume three of them rated it as 4 while two of them gave it a 5. Since the majority gave a rating of 4, the final rating will be taken as 4. You can consider this as taking the mode of all the predictions.

Max (Majority) Voting

- The max voting method is generally used for classification problems. In this technique, multiple models are used to make predictions for each data point. The predictions by each model are considered as a 'vote'. The predictions which we get from the majority of the models are used as the final prediction.
- For example, when you asked 5 of your colleagues to rate your movie (out of 5); we'll assume three of them rated it as 4 while two of them gave it a 5. Since the majority gave a rating of 4, the final rating will be taken as 4. You can consider this as taking the mode of all the predictions.
- The result of max voting would be something like this:

Colleague 1	Colleague 2	Colleague 3	Colleague 4	Colleague 5	Final rating
5	4	5	4	4	4

Averaging

- Similar to the max voting technique, multiple predictions are made for each data point in averaging. In this method, we take an average of predictions from all the models and use it to make the final prediction. Averaging can be used for making predictions in regression problems or while calculating probabilities for classification problems.

Averaging

- Similar to the max voting technique, multiple predictions are made for each data point in averaging. In this method, we take an average of predictions from all the models and use it to make the final prediction. Averaging can be used for making predictions in regression problems or while calculating probabilities for classification problems.
- For example, in the below case, the averaging method would take the average of all the values. i.e. $(5 + 4 + 5 + 4 + 4)/5 = 4.4$

Colleague 1	Colleague 2	Colleague 3	Colleague 4	Colleague 5	Final rating
5	4	5	4	4	4.4

Weighted Average

- This is an extension of the averaging method. All models are assigned different weights defining the importance of each model for prediction. For instance, if two of your colleagues are critics, while others have no prior experience in this field, then the answers by these two friends are given more importance as compared to the other people.

Weighted Average

- This is an extension of the averaging method. All models are assigned different weights defining the importance of each model for prediction. For instance, if two of your colleagues are critics, while others have no prior experience in this field, then the answers by these two friends are given more importance as compared to the other people.
- The result is calculated as $[(5 * 0.23) + (4 * 0.23) + (5 * 0.18) + (4 * 0.18) + (4 * 0.18)] = 4.41$.

Colleague 1	Colleague 2	Colleague 3	Colleague 4	Colleague 5	Final rating	
weight	0.23	0.23	0.18	0.18	0.18	
rating	5	4	5	4	4	4.41

Weighted Average

- This is an extension of the averaging method. All models are assigned different weights defining the importance of each model for prediction. For instance, if two of your colleagues are critics, while others have no prior experience in this field, then the answers by these two friends are given more importance as compared to the other people.
- The result is calculated as $[(5 * 0.23) + (4 * 0.23) + (5 * 0.18) + (4 * 0.18) + (4 * 0.18)] = 4.41$.

Colleague 1	Colleague 2	Colleague 3	Colleague 4	Colleague 5	Final rating	
weight	0.23	0.23	0.18	0.18	0.18	
rating	5	4	5	4	4	4.41

- think about what are proper weights?

Advanced Ensemble techniques

- Now that we have covered the basic ensemble techniques, let's move on to understanding the advanced techniques.

Advanced Ensemble techniques

- Now that we have covered the basic ensemble techniques, let's move on to understanding the advanced techniques.
  - Stacking

Advanced Ensemble techniques

- Now that we have covered the basic ensemble techniques, let's move on to understanding the advanced techniques.
  - Stacking
  - Blending

Advanced Ensemble techniques

- Now that we have covered the basic ensemble techniques, let's move on to understanding the advanced techniques.
    - Stacking
    - Blending
    - Bagging

Advanced Ensemble techniques

- Now that we have covered the basic ensemble techniques, let's move on to understanding the advanced techniques.
  - Stacking
  - Blending
  - Bagging
  - Boosting

Stacking

- Stacking is an ensemble learning technique that uses predictions from multiple models (for example decision tree, knn or svm) to build a new model. This model is used for making predictions on the test set.

Stacking

- Stacking is an ensemble learning technique that uses predictions from multiple models (for example decision tree, knn or svm) to build a new model. This model is used for making predictions on the test set.
- Reminder for "ML Pipeline": training, testing and validation

Stacking

- Stacking is an ensemble learning technique that uses predictions from multiple models (for example decision tree, knn or svm) to build a new model. This model is used for making predictions on the test set.
- Reminder for "ML Pipeline": training, testing and validation
- Below is a step-wise explanation for a simple stacked ensemble:

Stacking

Step 1. You have Train Data and Test Data. Assume we are using 4-fold cross validation to train base models, the train data is then divided into 4 parts.

train_data

	x_0	x_1	x_2	x_3	y
fold_1	0.94	0.27	0.80	0.34	1
	0.02	0.22	0.17	0.84	0
fold_2	0.83	0.11	0.23	0.42	1
	0.74	0.26	0.03	0.41	0
fold_3	0.08	0.29	0.76	0.37	0
	0.71	0.76	0.43	0.95	1
fold_4	0.08	0.71	0.97	0.04	0
	0.84	0.97	0.89	0.05	1

test_data

x_0	x_1	x_2	x_3	y
0.74	0.17	0.820	0.31	1
0.04	0.27	0.13	0.80	0
0.87	0.10	0.24	0.39	1

Training data (4-fold) and Testing data

Stacking

Step 2. Using the 4-part train data, the 1st base model (assuming its a decision tree) is fitted on 3 parts and predictions are made for the 4th part. This is done for each part of the training data.

	x_0	x_1	x_2	x_3	y
fold_1	0.94	0.27	0.80	0.34	1
	0.02	0.22	0.17	0.84	0
fold_2	0.83	0.11	0.23	0.42	1
	0.74	0.26	0.03	0.41	0
fold_3	0.08	0.29	0.76	0.37	0
	0.71	0.76	0.43	0.95	1

Model_1 Train

	x_0	x_1	x_2	x_3	y
fold_4	0.08	0.71	0.97	0.04	0
	0.84	0.97	0.89	0.05	1

Model_1 Predict

pred_m1
0.19
0.91

Model 1 training and prediction using 4-fold cross validation

Stacking

Step 2. At the end, all instance from training data will have a prediction. This creates a new feature for train data, call it pred m1 (predictions model 1).

train_data

	x_0	x_1	x_2	x_3	y
fold_1	0.94	0.27	0.80	0.34	1
	0.02	0.22	0.17	0.84	0
fold_2	0.83	0.11	0.23	0.42	1
	0.74	0.26	0.03	0.41	0
fold_3	0.08	0.29	0.76	0.37	0
	0.71	0.76	0.43	0.95	1
fold_4	0.08	0.71	0.97	0.04	0
	0.84	0.97	0.89	0.05	1

new_features

pred_m1
0.96
0.03
0.90
0.12
0.03
0.77
0.19
0.91

Stacking

Step 3. Model 1 (decision tree) is then fitted on the whole training data — no folding is needed this time. The trained model will be used to predict Test Data. So test data will also have pred m1.

test_data

x_0	x_1	x_2	x_3	y
0.74	0.17	0.820	0.31	1
0.04	0.27	0.13	0.80	0
0.87	0.10	0.24	0.39	1

+ Trained M1 =

new_features

pred_m1
0.43
0.03
0.90

Stacking

Step 4. Step 2 to 3 are repeated for the 2nd model (e.g KNN) and the 3rd model (e.g. SVM). These will give both train data and test data two more features from the predictions, pred m2 and pred m3.

train_data

	x_0	x_1	x_2	x_3	y
fold_1	0.94	0.27	0.80	0.34	1
	0.02	0.22	0.17	0.84	0
fold_2	0.83	0.11	0.23	0.42	1
	0.74	0.26	0.03	0.41	0
fold_3	0.08	0.29	0.76	0.37	0
	0.71	0.76	0.43	0.95	1
fold_4	0.08	0.71	0.97	0.04	0
	0.84	0.97	0.89	0.05	1

Training m1, m2, m3

train_data_new_features

pred_m1	pred_m2	pred_m3
0.96	0.86	0.66
0.03	0.13	0.30
0.90	0.90	0.85
0.12	0.10	0.11
0.03	0.09	0.08
0.77	0.50	0.67
0.19	0.65	0.20
0.91	0.89	0.90

test_data

x_0	x_1	x_2	x_3	y
0.74	0.17	0.820	0.31	1
0.04	0.27	0.13	0.80	0
0.87	0.10	0.24	0.39	1

Predicating m1, m2, m3

test_data_new_features

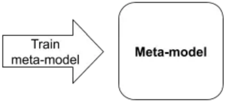
pred_m1	pred_m2	pred_m3
0.43	0.50	0.39
0.03	0.10	0.02
0.90	0.78	0.87

Stacking

Step 5. Now, to train the meta model (assume it's a logistic regression), we use only the newly added features from the base models, which are [pred m1, pred m2, pred m3]. Fit this meta model on train$_{d}ata$.

New train_data for meta-model

pred_m1	pred_m2	pred_m3	y
0.96	0.86	0.66	1
0.03	0.13	0.30	0
0.90	0.90	0.85	1
0.12	0.10	0.11	0
0.03	0.09	0.08	0
0.77	0.50	0.67	1
0.19	0.65	0.20	0
0.91	0.89	0.90	1

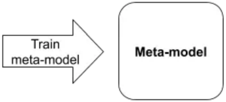


Stacking

Step 5. Now, to train the meta model (assume it's a logistic regression), we use only the newly added features from the base models, which are [pred m1, pred m2, pred m3]. Fit this meta model on train$_{d}ata$.

New train_data for meta-model

pred_m1	pred_m2	pred_m3	y
0.96	0.86	0.66	1
0.03	0.13	0.30	0
0.90	0.90	0.85	1
0.12	0.10	0.11	0
0.03	0.09	0.08	0
0.77	0.50	0.67	1
0.19	0.65	0.20	0
0.91	0.89	0.90	1



Step 6: The final prediction for test data is given by the trained meta model.

Blending

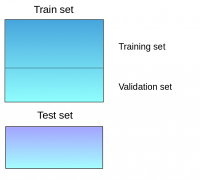
- Blending is very similar to Stacking. It also uses base models to provide base predictions as new features and a new meta model is trained on the new features that gives the final prediction.

Blending

- Blending is very similar to Stacking. It also uses base models to provide base predictions as new features and a new meta model is trained on the new features that gives the final prediction.
- The only difference is that training of the meta-model is applied on a separate holdout set (e.g 10% of train data) rather on full and folded training set.

Blending

Step 1. train data is split into base train data and holdout set.



Blending

Step 2. Base models are fitted on base train data, and predictions are made on holdout set and test data. These will create new prediction features.

train_data

	x_0	x_1	x_2	x_3	y
Base train data	0.94	0.27	0.80	0.34	1
	0.02	0.22	0.17	0.84	0
	0.83	0.11	0.23	0.42	1
	0.74	0.26	0.03	0.41	0
	0.08	0.29	0.76	0.37	0
	0.71	0.76	0.43	0.95	1
Holdout set	0.08	0.71	0.97	0.04	0
	0.84	0.97	0.89	0.05	1

train m1, m2, m3

Predict

train_data_new_features

pred_m1	pred_m2	pred_m3	y
0.19	0.65	0.20	0
0.91	0.89	0.90	1

test_data

x_0	x_1	x_2	x_3	y
0.74	0.17	0.820	0.31	1
0.04	0.27	0.13	0.80	0
0.87	0.10	0.24	0.39	1

Predicat m1, m2, m3

test_data_new_features

pred_m1	pred_m2	pred_m3
0.43	0.50	0.39
0.03	0.10	0.02
0.90	0.78	0.87

Blending

Step 3. A new meta-model is then fit on holdout set with new prediction features. Both original and meta features from holdout set will be used.

Blending

Step 3. A new meta-model is then fit on holdout set with new prediction features. Both original and meta features from holdout set will be used.

step 4. The trained meta-model is used to make final predictions on the test data using both original and new meta features.

Bagging

- Motivation: If you create all the models on the same set of data and combine it, will it be useful?

Bagging

- Motivation: If you create all the models on the same set of data and combine it, will it be useful?
- There is a high chance that these models will give the same result since they are getting the same input. So how can we solve this problem? One of the techniques is bootstrapping.

Bagging

- Motivation: If you create all the models on the same set of data and combine it, will it be useful?
- There is a high chance that these models will give the same result since they are getting the same input. So how can we solve this problem? One of the techniques is bootstrapping.
- Bootstrapping is a sampling technique in which we create subsets of observations from the original dataset, with replacement. The size of the subsets may or may not be the same as the size of the original set.

Bagging

- Motivation: If you create all the models on the same set of data and combine it, will it be useful?
- There is a high chance that these models will give the same result since they are getting the same input. So how can we solve this problem? One of the techniques is bootstrapping.
- Bootstrapping is a sampling technique in which we create subsets of observations from the original dataset, with replacement. The size of the subsets may or may not be the same as the size of the original set.
- Bagging (or Bootstrap Aggregating) technique uses these subsets (bags) to get a fair idea of the distribution (complete set). The size of subsets created for bagging may be less than the original set.

Bootstrap

- Bootstrap is a statistical technique used to estimate the sampling distribution of a statistic by resampling with replacement from the original data. It was introduced by Bradley Efron in 1979.

Bootstrap

- Bootstrap is a statistical technique used to estimate the sampling distribution of a statistic by resampling with replacement from the original data. It was introduced by Bradley Efron in 1979.
- The bootstrap method allows for the estimation of the distribution of almost any statistic, including means, medians, variances, and regression coefficients, without making strong assumptions about the underlying distribution of the data.

Bootstrap

- **Original Dataset**: Suppose you have a dataset X with n observations: $X = \{x_1, x_2, \cdots, x_n\}$.

Bootstrap

- **Original Dataset**: Suppose you have a dataset X with n observations: $X = \{x_1, x_2, \cdots, x_n\}$.
- **Resampling**: Create a large number of bootstrap samples by randomly drawing n observations with replacement from the original dataset. Each bootstrap sample is the same size as the original dataset but may contain duplicates due to the sampling with replacement.

Bootstrap

- **Original Dataset**: Suppose you have a dataset X with n observations: $X = \{x_1, x_2, \cdots, x_n\}$.
- **Resampling**: Create a large number of bootstrap samples by randomly drawing n observations with replacement from the original dataset. Each bootstrap sample is the same size as the original dataset but may contain duplicates due to the sampling with replacement.
- **Compute Statistic**: For each bootstrap sample, compute the statistic of interest (e.g., mean, median, variance).

Bootstrap

- **Original Dataset**: Suppose you have a dataset X with n observations: $X = \{x_1, x_2, \cdots, x_n\}$.
- **Resampling**: Create a large number of bootstrap samples by randomly drawing n observations with replacement from the original dataset. Each bootstrap sample is the same size as the original dataset but may contain duplicates due to the sampling with replacement.
- **Compute Statistic**: For each bootstrap sample, compute the statistic of interest (e.g., mean, median, variance).
- **Estimate Distribution**: Use the collection of bootstrap statistics to estimate the sampling distribution of the statistic. This can be used to compute confidence intervals, standard errors, and other measures of statistical uncertainty.

Bagging

- Multiple subsets are created from the original dataset, selecting observations with replacement.

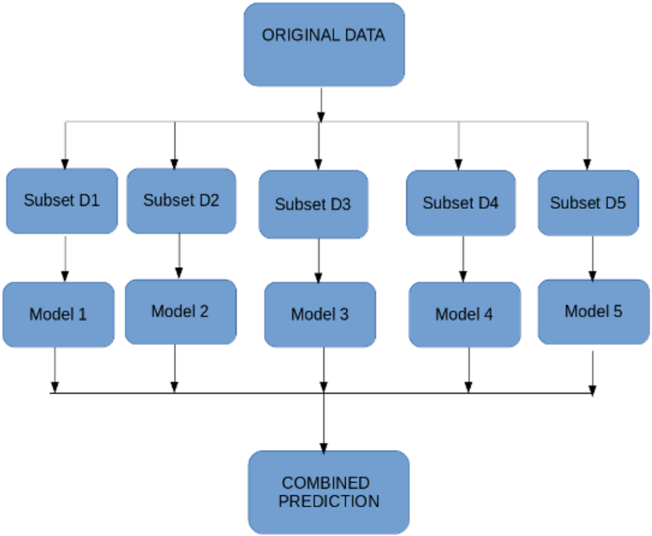
Bagging

- Multiple subsets are created from the original dataset, selecting observations with replacement.
- A base model (weak model) is created on each of these subsets.

Bagging

- Multiple subsets are created from the original dataset, selecting observations with replacement.
- A base model (weak model) is created on each of these subsets.
- The models run in parallel and are independent of each other.

Bagging


ORIGINAL DATA
Subset D1
Subset D2
Subset D3
Subset D4
Subset D5
Model 1
Model 2
Model 3
Model 4
Model 5
COMBINED PREDICTION

Boosting

- Before we go further, here's another question for you: If a data point is incorrectly predicted by the first model, and then the next (probably all models), will combining the predictions provide better results? Such situations are taken care of by boosting.

Boosting

- Before we go further, here's another question for you: If a data point is incorrectly predicted by the first model, and then the next (probably all models), will combining the predictions provide better results? Such situations are taken care of by boosting.
- Boosting is a *sequential process*, where each subsequent model attempts to correct the errors of the previous model. The succeeding models are dependent on the previous model. Let's understand the way boosting works in the below steps.

Boosting

- 1. A subset is created from the original dataset.

Boosting

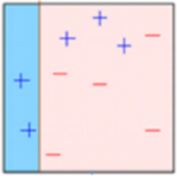
- 1. A subset is created from the original dataset.
- 2. Initially, all data points are given equal weights.

Boosting

- 1. A subset is created from the original dataset.
- 2. Initially, all data points are given equal weights.
- 3. A base model is created on this subset.

Boosting

- 1. A subset is created from the original dataset.
- 2. Initially, all data points are given equal weights.
- 3. A base model is created on this subset.
- 4. This model is used to make predictions on the whole dataset.



Boosting

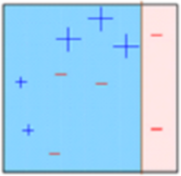
- 5. Errors are calculated using the actual values and predicted values.

Boosting

- 5. Errors are calculated using the actual values and predicted values.
- 6. The observations which are incorrectly predicted, are given higher weights. (Here, the three misclassified blue-plus points will be given higher weights)

Boosting

- 5. Errors are calculated using the actual values and predicted values.
- 6. The observations which are incorrectly predicted, are given higher weights. (Here, the three misclassified blue-plus points will be given higher weights)
- 7. Another model is created and predictions are made on the dataset (This model tries to correct the errors from the previous model)

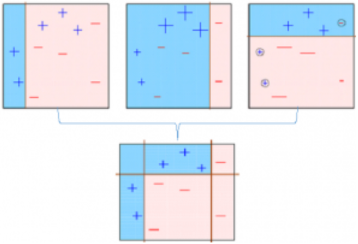


Boosting

- 8. Similarly, multiple models are created, each correcting the errors of the previous model.

Boosting

- 8. Similarly, multiple models are created, each correcting the errors of the previous model.
- 9. The final model (*strong learner*) is the weighted mean of all the models (*weak learners*). Thus, the boosting algorithm combines a number of weak learners to form a strong learner. The individual models would not perform well on the entire dataset, but they work well for some part of the dataset. Thus, each model actually boosts the performance of the ensemble.



Adaboost algorithm for creating strong classifier

- AdaBoost, short for Adaptive Boosting, is a machine learning algorithm that is used as an ensemble method for improving the performance of weak classifiers. It was introduced by Yoav Freund and Robert Schapire in 1995.

Adaboost algorithm for creating strong classifier

- AdaBoost, short for Adaptive Boosting, is a machine learning algorithm that is used as an ensemble method for improving the performance of weak classifiers. It was introduced by Yoav Freund and Robert Schapire in 1995.
- The main idea behind AdaBoost is to introduce *adaptive weights* in combining multiple weak classifiers (models that perform slightly better than random guessing) into a single strong classifier.

Adaboost algorithm for creating strong classier

- **Initialization**: Assign equal weights to all training examples. If there are N training examples, each example is assigned a weight of $1/N$.

Adaboost algorithm for creating strong classier

- **Initialization**: Assign equal weights to all training examples. If there are N training examples, each example is assigned a weight of $1/N$.
- **Iterative Training**: For each iteration t (from 1 to T, where T is the total number of iterations):

Adaboost algorithm for creating strong classier

- **Initialization**: Assign equal weights to all training examples. If there are N training examples, each example is assigned a weight of $1/N$.
- **Iterative Training**: For each iteration t (from 1 to T, where T is the total number of iterations):
  - Train a weak classifier h_t, using the weighted training examples.

Adaboost algorithm for creating strong classier

- **Initialization**: Assign equal weights to all training examples. If there are N training examples, each example is assigned a weight of $1/N$.
- **Iterative Training**: For each iteration t (from 1 to T, where T is the total number of iterations):
  - Train a weak classifier h_t, using the weighted training examples.
  - Calculate the error ε_t of h_t as the weighted sum of misclassified examples

$$\varepsilon_t = \frac{\sum_{i=1}^{N} w_i I(y_i \neq h_t(x_i))}{\sum_{i=1}^{N} w_i}$$

    where w_i is the weight of the i-th example, y_i is the true label, $h_t(x_i)$ is the predicted label by the classifier h_t, and $I(\cdot)$ is the indicator function that returns 1 if the condition is true, and 0 otherwise.

Adaboost algorithm for creating strong classier

- **Iterative Training**: For each iteration t (from 1 to T, where T is the total number of iterations):

Adaboost algorithm for creating strong classier

- **Iterative Training**: For each iteration t (from 1 to T, where T is the total number of iterations):
  - Compute the classifier's weight α_t using the formula:

$$\alpha_t = \frac{1}{2}\log\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right)$$

    The smaller the error ε_t, the larger weight α_t

Adaboost algorithm for creating strong classier

- **Iterative Training**: For each iteration t (from 1 to T, where T is the total number of iterations):
  - Compute the classifier's weight α_t using the formula:

    $$\alpha_t = \frac{1}{2}\log(\frac{1-\varepsilon_t}{\varepsilon_t})$$

    The smaller the error ε_t, the larger weight α_t
  - Update the weights of the training examples:

    $$w_i \leftarrow w_i \exp(-\alpha_t y_i h_t(x_i))$$

    and then normalize the weights so that they sum up to 1.

Adaboost algorithm for creating strong classier

- **Final Model**: After T iterations, the final model is a weighted combination of the weak classifiers:

$$H(x) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

where $\text{sign}(\cdot)$ is the sign function that return $+1$ for positive input and -1 for negative input.

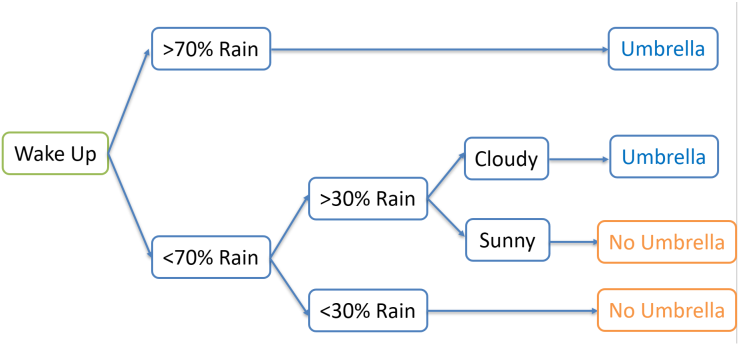
Random forest

- What is Random forest ?

Random forest

- What is Random forest ?
- A Random Forest is like a group decision-making team in machine learning. It combines the opinions of many “trees” (individual models) to make better predictions, creating a more robust and accurate overall model.

Recall: what is a decision tree ?



- Tree-logic uses a sequence of inquiries to come to a conclusion.

Random forest algorithm

- **Step 1**: In the Random forest model, a subset of data points and a subset of features is selected for constructing each decision tree. Simply put, n random records and m features are taken from the data set having k number of records.

Random forest algorithm

- **Step 1**: In the Random forest model, a subset of data points and a subset of features is selected for constructing each decision tree. Simply put, n random records and m features are taken from the data set having k number of records.
- **Step 2**: Individual decision trees are constructed for each sample.

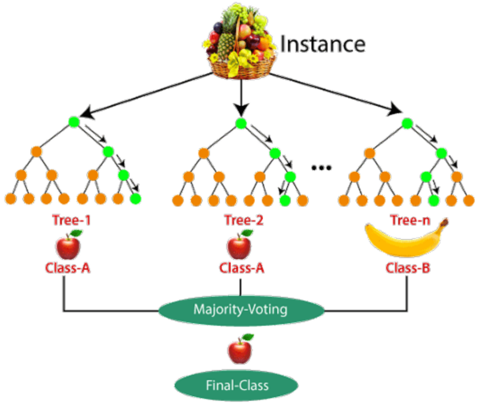
Random forest algorithm

- **Step 1**: In the Random forest model, a subset of data points and a subset of features is selected for constructing each decision tree. Simply put, n random records and m features are taken from the data set having k number of records.
- **Step 2**: Individual decision trees are constructed for each sample.
- **Step 3**: Each decision tree will generate an output.

Random forest algorithm

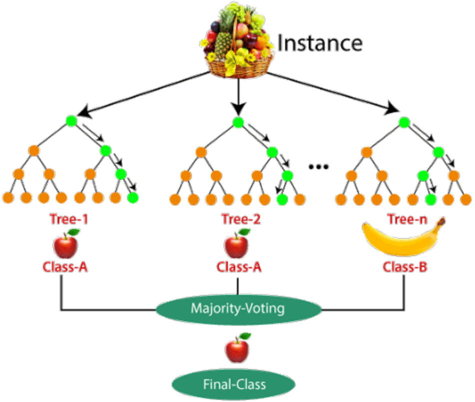
- **Step 1**: In the Random forest model, a subset of data points and a subset of features is selected for constructing each decision tree. Simply put, n random records and m features are taken from the data set having k number of records.
- **Step 2**: Individual decision trees are constructed for each sample.
- **Step 3**: Each decision tree will generate an output.
- **Step 4**: Final output is considered based on Majority Voting or Averaging for Classification and regression, respectively.

Random forest algorithm: an example



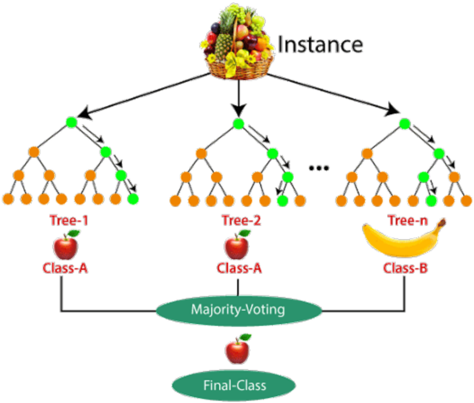
- Consider the fruit basket as the data as shown in the figure above.

Random forest algorithm: an example



- Consider the fruit basket as the data as shown in the figure above.
- Now n number of samples are taken from the fruit basket, and an individual decision tree is constructed for each sample.

Random forest algorithm: an example



- Each decision tree will generate an output, as shown in the figure. The final output is considered based on majority voting.

Exercise: Manually Constructing a Simple Random Forest

- Objective: Given a small dataset, manually construct a simple random forest model with two decision trees and make a prediction for a new observation.

Exercise: Manually Constructing a Simple Random Forest

- Objective: Given a small dataset, manually construct a simple random forest model with two decision trees and make a prediction for a new observation.
- Dataset:

Observation	Feature 1 (X1)	Feature 2 (X2)	Target (Y)
1	1	5	0
2	2	6	0
3	3	7	1
4	4	8	1

Exercise: Manually Constructing a Simple Random Forest

- **Bootstrap Sampling**:

Exercise: Manually Constructing a Simple Random Forest

- **Bootstrap Sampling**:
  - Create two bootstrap samples from the original dataset. Each sample should have the same number of observations as the original dataset, but some observations may be repeated due to sampling with replacement.

Exercise: Manually Constructing a Simple Random Forest

- **Bootstrap Sampling**:
  - Create two bootstrap samples from the original dataset. Each sample should have the same number of observations as the original dataset, but some observations may be repeated due to sampling with replacement.
  - Sample 1: [Fill in your sample]

Exercise: Manually Constructing a Simple Random Forest

- **Bootstrap Sampling**:
  - Create two bootstrap samples from the original dataset. Each sample should have the same number of observations as the original dataset, but some observations may be repeated due to sampling with replacement.
  - Sample 1: [Fill in your sample]
  - Sample 2: [Fill in your sample]

Exercise: Manually Constructing a Simple Random Forest

- **Bootstrap Sampling**:
  - Create two bootstrap samples from the original dataset. Each sample should have the same number of observations as the original dataset, but some observations may be repeated due to sampling with replacement.
  - Sample 1: [Fill in your sample]
  - Sample 2: [Fill in your sample]
- **Build Decision Trees**:

Exercise: Manually Constructing a Simple Random Forest

- **Bootstrap Sampling**:
  - Create two bootstrap samples from the original dataset. Each sample should have the same number of observations as the original dataset, but some observations may be repeated due to sampling with replacement.
  - Sample 1: [Fill in your sample]
  - Sample 2: [Fill in your sample]
- **Build Decision Trees**:
  - For each bootstrap sample, build a simple decision tree with just one split. Choose the split based on a simple criterion (e.g., the median of a feature).

Exercise: Manually Constructing a Simple Random Forest

- **Bootstrap Sampling**:
  - Create two bootstrap samples from the original dataset. Each sample should have the same number of observations as the original dataset, but some observations may be repeated due to sampling with replacement.
  - Sample 1: [Fill in your sample]
  - Sample 2: [Fill in your sample]
- **Build Decision Trees**:
  - For each bootstrap sample, build a simple decision tree with just one split. Choose the split based on a simple criterion (e.g., the median of a feature).
  - Tree 1 (from Sample 1): [Describe the split and the resulting leaf nodes]

Exercise: Manually Constructing a Simple Random Forest

- **Bootstrap Sampling**:
  - Create two bootstrap samples from the original dataset. Each sample should have the same number of observations as the original dataset, but some observations may be repeated due to sampling with replacement.
  - Sample 1: [Fill in your sample]
  - Sample 2: [Fill in your sample]
- **Build Decision Trees**:
  - For each bootstrap sample, build a simple decision tree with just one split. Choose the split based on a simple criterion (e.g., the median of a feature).
  - Tree 1 (from Sample 1): [Describe the split and the resulting leaf nodes]
  - Tree 2 (from Sample 2): [Describe the split and the resulting leaf nodes]

Exercise: Manually Constructing a Simple Random Forest

- **Make a Prediction**:

Exercise: Manually Constructing a Simple Random Forest

- **Make a Prediction**:
  - Use your Random Forest model to make a prediction for a new observation with $X_1 = 2$ and $X_2 = 7$.

Exercise: Manually Constructing a Simple Random Forest

- **Make a Prediction**:
  - Use your Random Forest model to make a prediction for a new observation with $X_1 = 2$ and $X_2 = 7$.
  - Tree 1 Prediction: [Your prediction]

- **Make a Prediction**:
  - Use your Random Forest model to make a prediction for a new observation with $X_1 = 2$ and $X_2 = 7$.
  - Tree 1 Prediction: [Your prediction]
  - Tree 2 Prediction: [Your prediction]

- **Make a Prediction**:
  - Use your Random Forest model to make a prediction for a new observation with $X_1 = 2$ and $X_2 = 7$.
  - Tree 1 Prediction: [Your prediction]
  - Tree 2 Prediction: [Your prediction]
  - Random Forest Prediction (majority vote): [Your final prediction]

Exercise: Manually Constructing a Simple Random Forest

- **Make a Prediction**:
  - Use your Random Forest model to make a prediction for a new observation with $X_1 = 2$ and $X_2 = 7$.
  - Tree 1 Prediction: [Your prediction]
  - Tree 2 Prediction: [Your prediction]
  - Random Forest Prediction (majority vote): [Your final prediction]
- **Your Task**: Fill in the blanks for each step based on the instructions provided. This exercise will help you understand the process of creating a Random Forest model and how it makes predictions.

Exercise: Manually Constructing a Simple Random Forest

- **An example of solution**:

Exercise: Manually Constructing a Simple Random Forest

- **An example of solution**:
- **Step 1: Bootstrap Sampling** Randomly sample with replacement from the dataset to create two bootstrap samples. Each sample should have the same number of observations as the original dataset.

Exercise: Manually Constructing a Simple Random Forest

- **An example of solution**:
- **Step 1: Bootstrap Sampling** Randomly sample with replacement from the dataset to create two bootstrap samples. Each sample should have the same number of observations as the original dataset.
- **Sample 1**

Observation	Feature 1 (X1)	Feature 2 (X2)	Target (Y)
1	1	5	0
2	2	6	0
3	3	7	1
4	2	6	0

Exercise: Manually Constructing a Simple Random Forest

- **An example of solution**:
- **Step 1: Bootstrap Sampling** Randomly sample with replacement from the dataset to create two bootstrap samples. Each sample should have the same number of observations as the original dataset.
- **Sample 1**

Observation	Feature 1 (X1)	Feature 2 (X2)	Target (Y)
1	1	5	0
2	2	6	0
3	3	7	1
4	2	6	0

- **Sample 2**

Observation	Feature 1 (X1)	Feature 2 (X2)	Target (Y)
1	4	8	1
2	3	7	1
3	1	5	0
4	4	8	1

Exercise: Manually Constructing a Simple Random Forest

- **Step 2: Build Decision Trees**

Exercise: Manually Constructing a Simple Random Forest

- **Step 2: Build Decision Trees**
- For simplicity, let's assume we build decision trees using only one split based on the feature that provides the best separation (e.g., using Gini impurity or information gain).

Exercise: Manually Constructing a Simple Random Forest

- **Step 2: Build Decision Trees**
- For simplicity, let's assume we build decision trees using only one split based on the feature that provides the best separation (e.g., using Gini impurity or information gain).
- Tree 1 (from Sample 1):

Exercise: Manually Constructing a Simple Random Forest

- **Step 2: Build Decision Trees**
- For simplicity, let's assume we build decision trees using only one split based on the feature that provides the best separation (e.g., using Gini impurity or information gain).
- Tree 1 (from Sample 1):
  - Split on $X_1 \leq 2.5$

Exercise: Manually Constructing a Simple Random Forest

- **Step 2: Build Decision Trees**
- For simplicity, let's assume we build decision trees using only one split based on the feature that provides the best separation (e.g., using Gini impurity or information gain).
- Tree 1 (from Sample 1):
  - Split on $X_1 \leq 2.5$
  - Left node: Majority class $= 0$

Exercise: Manually Constructing a Simple Random Forest

- **Step 2: Build Decision Trees**
- For simplicity, let's assume we build decision trees using only one split based on the feature that provides the best separation (e.g., using Gini impurity or information gain).
- Tree 1 (from Sample 1):
  - Split on $X_1 \leq 2.5$
  - Left node: Majority class $= 0$
  - Right node: Majority class $= 1$

- **Step 2: Build Decision Trees**
- For simplicity, let's assume we build decision trees using only one split based on the feature that provides the best separation (e.g., using Gini impurity or information gain).
- Tree 1 (from Sample 1):
  - Split on $X_1 \leq 2.5$
  - Left node: Majority class $= 0$
  - Right node: Majority class $= 1$
- Tree 2 (from Sample 2):

Exercise: Manually Constructing a Simple Random Forest

- **Step 2: Build Decision Trees**
- For simplicity, let's assume we build decision trees using only one split based on the feature that provides the best separation (e.g., using Gini impurity or information gain).
- Tree 1 (from Sample 1):
  - Split on $X_1 \leq 2.5$
  - Left node: Majority class $= 0$
  - Right node: Majority class $= 1$
- Tree 2 (from Sample 2):
  - Split on $X_1 \leq 3.5$

Exercise: Manually Constructing a Simple Random Forest

- **Step 2: Build Decision Trees**
- For simplicity, let's assume we build decision trees using only one split based on the feature that provides the best separation (e.g., using Gini impurity or information gain).
- Tree 1 (from Sample 1):
  - Split on $X_1 \leq 2.5$
  - Left node: Majority class $= 0$
  - Right node: Majority class $= 1$
- Tree 2 (from Sample 2):
  - Split on $X_1 \leq 3.5$
  - Left node: Majority class $= 0$

Exercise: Manually Constructing a Simple Random Forest

- **Step 2: Build Decision Trees**
- For simplicity, let's assume we build decision trees using only one split based on the feature that provides the best separation (e.g., using Gini impurity or information gain).
- Tree 1 (from Sample 1):
  - Split on $X_1 \leq 2.5$
  - Left node: Majority class $= 0$
  - Right node: Majority class $= 1$
- Tree 2 (from Sample 2):
  - Split on $X_1 \leq 3.5$
  - Left node: Majority class $= 0$
  - Right node: Majority class $= 1$

Exercise: Manually Constructing a Simple Random Forest

- **Step 3: Build Decision Trees**

Exercise: Manually Constructing a Simple Random Forest

- **Step 3: Build Decision Trees**
- To make a prediction for a new observation, pass it through each tree and take the majority vote.

Exercise: Manually Constructing a Simple Random Forest

- **Step 3: Build Decision Trees**
- To make a prediction for a new observation, pass it through each tree and take the majority vote.
- Predict Y for a new observation with $X_1 = 2$ and $X_2 = 7$.

Exercise: Manually Constructing a Simple Random Forest

- **Step 3: Build Decision Trees**
- To make a prediction for a new observation, pass it through each tree and take the majority vote.
- Predict Y for a new observation with $X_1 = 2$ and $X_2 = 7$.
  - Tree 1 prediction: $X_1 \leq 2.5 \rightarrow Y = 0$

Exercise: Manually Constructing a Simple Random Forest

- **Step 3: Build Decision Trees**
- To make a prediction for a new observation, pass it through each tree and take the majority vote.
- Predict Y for a new observation with $X_1 = 2$ and $X_2 = 7$.
  - Tree 1 prediction: $X_1 \leq 2.5 \rightarrow Y = 0$
  - Tree 2 prediction: $X_1 \leq 3.5 \rightarrow Y = 0$

Exercise: Manually Constructing a Simple Random Forest

- **Step 3: Build Decision Trees**
- To make a prediction for a new observation, pass it through each tree and take the majority vote.
- Predict Y for a new observation with $X_1 = 2$ and $X_2 = 7$.
  - Tree 1 prediction: $X_1 \leq 2.5 \rightarrow Y = 0$
  - Tree 2 prediction: $X_1 \leq 3.5 \rightarrow Y = 0$
  - Final prediction (majority vote): $Y = 0$