Probability

A ***probability model*** is a mathematical representation of a random phenomenon.

An ***event A*** is a subset of the sample space S.

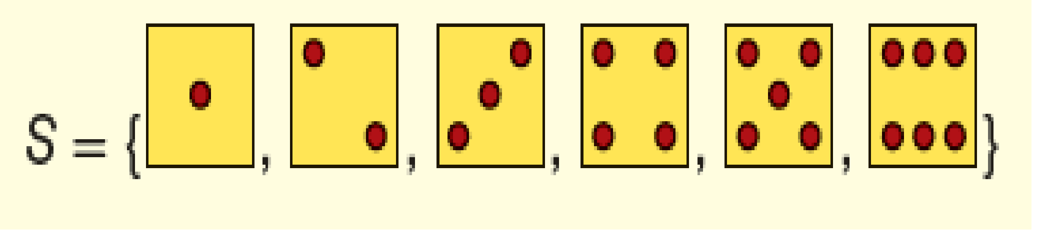
A ***probability*** is a numerical value assigned to a given event *A*. The probability of an event is written *P(A)*

The first two basic rules of probability are the following:

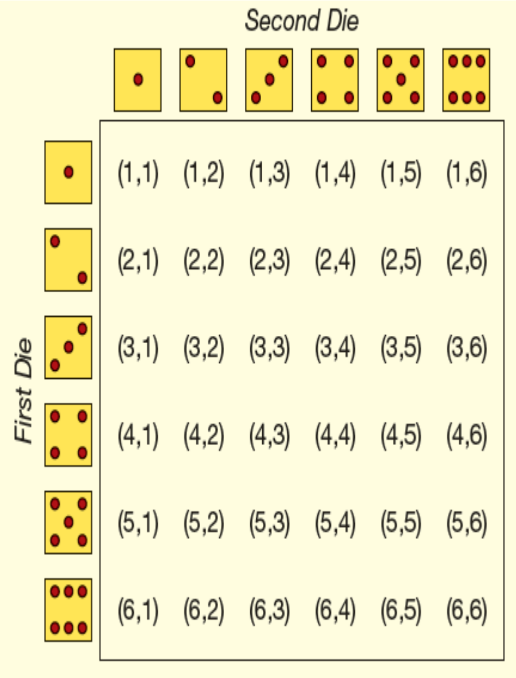
Rule 1: *Any probability P(A) is a number between 0 and 1* ($0 \leq P(A) \leq 1$).

Rule 2: *The probability of the sample space S is equal to 1* ($P(S) = 1$).

- A **random experiment** is an observational process whose results cannot be known in advance
- The sample space to describe rolling a die has six outcomes:



- Sample Space
  - When two dice are rolled, the sample space consists of 36 outcomes, each of which is a pair:



- The **probability** of an event is a number that measures the relative likelihood that the event will occur.
- The probability of an event A, denoted $P(A)$, must lie within the interval from 0 to 1:

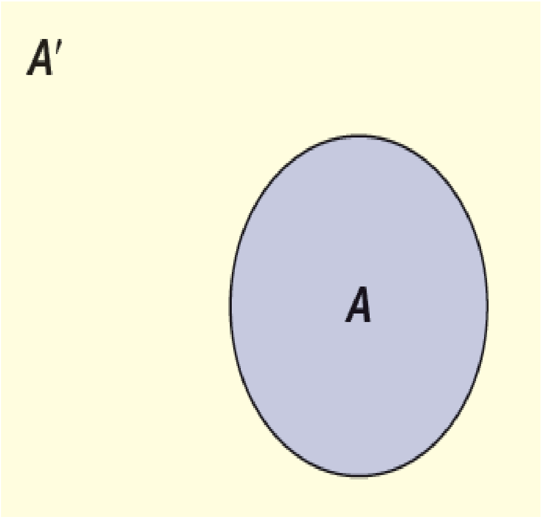
  $$0 <= P(A) <= 1$$

- In a discrete sample space, the probabilities of all simple events must sum to 1, since it is certain that one of them will occur:

  $$P(S) = P(E1) + P(E2) + \ldots + P(En) = 1$$

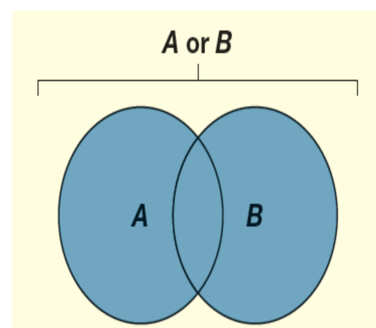
- **Complement of an Event**
- The **complement** of an event *A* is denoted *A'* and consists of everything in the sample space *S* except event *A*
- Since *A* and *A'* together comprise the sample space, their probabilities sum to 1:

$$P(A) + P(A') = 1$$
$$P(A') = 1 - P(A)$$

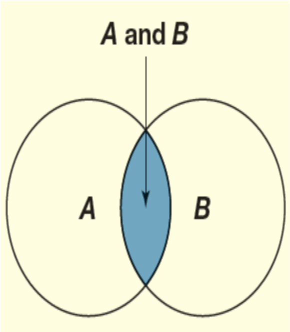


- **Complement of an Event**
  - For example, it has been reported that about 33 percent of all new small businesses fail within the first 2 years. From this we can determine that the probability that a new small business will survive at least 2 years is:

    $P(\text{survival}) = 1 - P(\text{failure}) = 1 - 0.33 = 0.67$, or 67%
- **Union of Two Events**
  - The **union** of two events consists of all outcomes in the sample space S that are contained either in event A or in event B or in both. The union of A and B is sometimes denoted $A \cup B$ or "A or B"



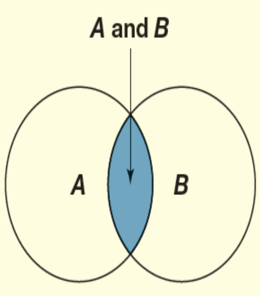
- **Intersection of Two Events**
  - The **intersection** of two events *A* and *B* is the event consisting of all outcomes in the sample space *S* that are contained in both event *A* and event *B*. The intersection of *A* and *B* is denoted $A \cap B$ *or "A and B"*
  - The probability of $A \cap B$ is called the **joint probability** and is denoted $P(A \cap B)$
  - For example, if *Q* is the event that we draw a queen and *R* is the event that we draw a red card, then, *Q* ∩ *R* is the event that we get a card that is both a queen and red. That is, the intersection of sets *Q* and *R* consists of two cards (*Q* ♥ and *Q* ♦).



General Law of Addition

- The **general law of addition** says that the probability of the union of two events *A* and *B* is the sum of their probabilities less the probability of their intersection

$$P(A \cup B) = P(A) + P(B) - P(A \cap B)$$



- If we just add the probabilities of *A* and *B*, we would count the intersection twice, so we must subtract the probability of $A \cap B$ to avoid overstating the probability of $A \cup B$.

Probability Theory – Rules of Probability

- **General Law of Addition**

$$P(A \cup B) = P(A) + P(B) - P(A \cap B)$$

  - For the card example:

    Queen: $P(Q) = 4/52$ (there are 4 queens in a deck)

    Red: $P(R) = 26/52$ (there are 26 red cards in a deck)

    Queen and Red: $P(Q \cap R) = 2/52$ (there are 2 red queens in a deck)

    Therefore,

    Queen or Red: $P(Q \cup R) = P(Q) + P(R) - P(Q \cap R)$

    $= 4/52 + 26/52 - 2/52$

    $= 28/52 = 0.5385$, or a 53.85% chance

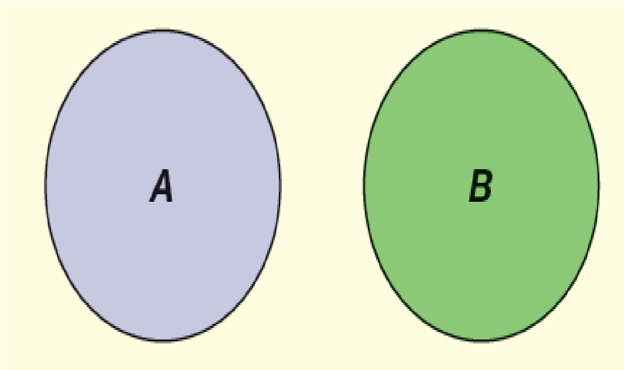
- **General Law of Addition**

A survey of introductory statistics students showed that 29.7 percent have AT&T wireless service (event A), 73.4 percent have a Visa card (event B), and 20.3 percent have both (event $A \cap B$). The probability that a student uses AT&T *or* has a Visa card is:

$$P(A \cup B) = P(A) + P(B) - P(A \cap B) = .297 + .734 - .203 = .828$$

- **Mutually Exclusive Events**
  - Events A and B are **mutually exclusive** (or **disjoint**) if their intersection is the **empty set** (a set that contains no elements). In other words, one event precludes the other from occurring.
  - The null set is denoted ϕ.

$$\text{If } A \cap B = \phi, \text{ then } P(A \cap B) = 0$$



- **Mutually Exclusive Events**

Here are examples of events that are mutually exclusive (cannot be in both categories):

- *Customer age: A* = under 21, *B* = over 65
- *Purebred dog breed: A* = border collie, *B* = golden retriever

Here are examples of events that are *not* mutually exclusive (can be in both categories):

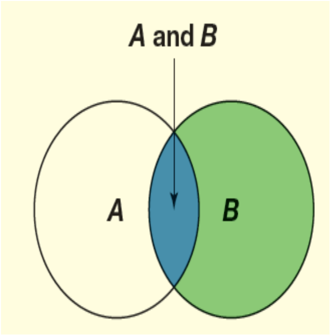
- *Student's major: A* = marketing major, *B* = economics major
- *Credit card held: A* = Visa, *B* = MasterCard, *C* = American Express

- **Special Law of Addition (Mutually Exclusive Events)**
  - If A and B are mutually exclusive events, then $P(A \cap B) = 0$ and the general addition law can be simplified to the sum of the individual probabilities for A and B, the **special law of addition**.
  - For example, if we look at a person's age, then P(under 21) 0.28 and P(over 65) = 0.12, so P(under 21 *or* over 65) = 0.28 + 0.12 = 0.40 since these events do not overlap.

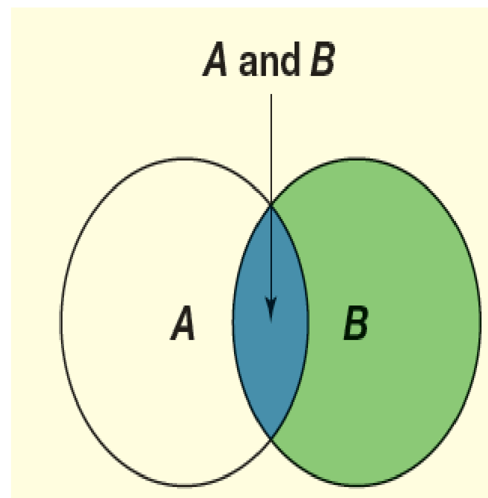
$$P(A \cup B) = P(A) + P(B) \qquad \text{(addition law for mutually exclusive events)}$$

- **Conditional Probability**
  - The probability of event *A* *given* that event *B* has occurred is a **conditional probability**, denoted $P(A \mid B)$ which is read "the probability of *A* given *B*." The vertical line is read as "given."
  - The conditional probability is the joint probability of *A* and *B* divided by the probability of *B*.

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)} \qquad \text{for } P(B) > 0$$



- **Conditional Probability**
  - The sample space is restricted to B, an event that we know has occurred (the green shaded circle). The intersection, $(A \cap B)$, is the part of B that is also in A (the blue shaded area).
  - The ratio of the relative size of set $(A \cap B)$ to set B is the conditional probability $P(A \mid B)$.



Prior Probability

P(H)

is the *a priori* probability that a specified hypothesis is true. This is often called the prior probability, or just the *prior*.

This is the **unconditional** probability, without taking any evidence into consideration.

Prior probability

Consider the random variables,

cavity={true, false}

weather={sunny, rain, cloudy, snow}

Prior or unconditional probability,

P(cavity=true)=0.1

P(weather=sunny)=0.72

Probability distribution gives values of all possible assignments:

P(weather)={0.72, 0.1, 0.08, 0.1}(normalized i.e, sums to 1)

Bayes' Rule

The condition probability of the occurrence of* A *if event B occurs

$P(A|B) = P(A \wedge B) / P(B)$

This can be written also as:

$P(A \wedge B) = P(A|B) * P(B)$

$P(A \wedge B) = P(B|A) * P(A)$

Hence $P(B|A) = \dfrac{P(A|B) * P(B)}{P(A)}$

Naïve Bayesian Classifier

Using Bayes theorem, we can find the probability of **A** happening, given that **B** has occurred. Here, **B** is the evidence and **A** is the hypothesis. The assumption made here is that the predictors/features are independent. That is presence of one particular feature does not affect the other. Hence it is called naive.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

Bayes theorem can be rewritten as:

$$P(y|X) = \frac{P(X|y)P(y)}{P(X)}$$

Towards Naïve Bayesian Classifier

- Let D be a training set of tuples and their associated class labels, and each tuple is represented by an n-D attribute vector $\mathbf{X} = (x_1, x_2, \ldots, x_n)$
- Suppose there are *m* classes $C_1, C_2, \ldots, C_m$.
- Classification is to derive the maximum posteriori, i.e., the maximal $P(C_i|\mathbf{X})$
- This can be derived from Bayes' theorem

$$P(C_i|\mathbf{X}) = \frac{P(\mathbf{X}|C_i)P(C_i)}{P(\mathbf{X})}$$

- Since P(X) is constant for all classes, only

$$P(C_i|\mathbf{X}) = P(\mathbf{X}|C_i)P(C_i)$$

needs to be maximized

Derivation of Naïve Bayes Classifier

- A simplified assumption: attributes are conditionally independent (i.e., no dependence relation between attributes):

$$P(\mathbf{X}|C_i) = \prod_{k=1}^{n} P(x_k|C_i) = P(x_1|C_i) \times P(x_2|C_i) \times \ldots \times P(x_n|C_i)$$

- This greatly reduces the computation cost: Only counts the class distribution
- If A_k is categorical, $P(x_k|C_i)$ is the # of tuples in C_i having value x_k for A_k divided by $|C_{i, D}|$ (# of tuples of C_i in D)
- If A_k is continous-valued, $P(x_k|C_i)$ is usually computed based on Gaussian distribution with a mean μ and standard deviation σ

and $P(x_k|C_i)$ is

$$g(x, \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

$$P(\mathbf{X}|C_i) = g(x_k, \mu_{C_i}, \sigma_{C_i})$$

Problem 1: If the weather is sunny, then the Player should play or not?

Solution: To solve this, first consider the below dataset:

	Outlook	Play
0	Rainy	Yes
1	Sunny	Yes
2	Overcast	Yes
3	Overcast	Yes
4	Sunny	No
5	Rainy	Yes
6	Sunny	Yes
7	Overcast	Yes
8	Rainy	No
9	Sunny	No
10	Sunny	Yes
11	Rainy	No
12	Overcast	Yes
13	Overcast	Yes

Applying Bayes'theorem:

$$P(C_i|\mathbf{X}) = \frac{P(\mathbf{X}|C_i)P(C_i)}{P(\mathbf{X})}$$

P(Yes|Sunny)= P(Sunny|Yes)*P(Yes)/P(Sunny)

P(No|Sunny)= P(Sunny|No)*P(No)/P(Sunny)

P(Yes|Sunny)= P(Sunny|Yes)*P(Yes)/P(Sunny)

P(Sunny|Yes)= 3/10= 0.3

P(Sunny)= 0.35

P(Yes)=0.71

So P(Yes|Sunny) = 0.3*0.71/0.35= **0.60**

P(No|Sunny)= P(Sunny|No)*P(No)/P(Sunny)

P(Sunny|NO)= 2/4=0.5

P(No)= 0.29

P(Sunny)= 0.35

So P(No|Sunny)= 0.5*0.29/0.35 = **0.41**

So as we can see from the above calculation that **P(Yes|Sunny)>P(No|Sunny)**

Hence on a Sunny day, Player can play the game.

Problem 2

	Outlook	Temperature	Humidity	Windy	Play Golf
0	Rainy	Hot	High	False	No
1	Rainy	Hot	High	True	No
2	Overcast	Hot	High	False	Yes
3	Sunny	Mild	High	False	Yes
4	Sunny	Cool	Normal	False	Yes
5	Sunny	Cool	Normal	True	No
6	Overcast	Cool	Normal	True	Yes
7	Rainy	Mild	High	False	No
8	Rainy	Cool	Normal	False	Yes
9	Sunny	Mild	Normal	False	Yes
10	Rainy	Mild	Normal	True	Yes
11	Overcast	Mild	High	True	Yes
12	Overcast	Hot	Normal	False	Yes
13	Sunny	Mild	High	True	No

Let us test it on a new set of features (let us call it today):
today = (Sunny, Hot, Normal, False)

Outlook

	Yes	No	P(yes)	P(no)
Sunny	2	3	2/9	3/5
Overcast	4	0	4/9	0/5
Rainy	3	2	3/9	2/5
Total	9	5	100%	100%

Temperature

	Yes	No	P(yes)	P(no)
Hot	2	2	2/9	2/5
Mild	4	2	4/9	2/5
Cool	3	1	3/9	1/5
Total	9	5	100%	100%

Humidity

	Yes	No	P(yes)	P(no)
High	3	4	3/9	4/5
Normal	6	1	6/9	1/5
Total	9	5	100%	100%

Wind

	Yes	No	P(yes)	P(no)
False	6	2	6/9	2/5
True	3	3	3/9	3/5
Total	9	5	100%	100%

Play		P(Yes)/P(No)
Yes	9	9/14
No	5	5/14
Total	14	100%

So, probability of playing golf is given by:

$$P(Yes|today) = \frac{P(SunnyOutlook|Yes)P(HotTemperature|Yes)P(NormalHumidity|Yes)P(NoWind|Yes)P(Yes)}{P(today)}$$

and probability to not play golf is given by:

$$P(No|today) = \frac{P(SunnyOutlook|No)P(HotTemperature|No)P(NormalHumidity|No)P(NoWind|No)P(No)}{P(today)}$$

P(today)=P(sunny)*P(hot)*p(normal)*p(nowind)
=(5/14)*(4/14)*(7/14)*(8/14)

P(Yes/today)=((3/9)*(2/9)*(6/9)*(6/9)*(9/14)) / P(today)

=0.0141/ P(today)

P(No/today)=((2/5)*(2/5)*(1/5)*(2/5)*(5/14)) / P(today)

=0.00457/ P(today)

Since

$$P(Yes|today) > P(No|today)$$

So, prediction that golf would be played is 'Yes'.

Naïve Bayesian Classifier: Training Dataset

Class:
C1:buys_computer = 'yes'
C2:buys_computer = 'no'

Data sample
X = (age <=30,
Income = medium,
Student = yes
Credit_rating = Fair)

age	income	student	credit_rating	_comp
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

Solution:

E= age<=30, income=medium, student=yes, credit-rating=fair
E_1 is age<=30, E2 is income=medium, student=yes, E4 is credit-rating=fair
We need to compute P(yes|E) and P(no|E) and compare them.

$$P(yes \mid E) = \frac{P(E_1 \mid yes)\, P(E_2 \mid yes)\, P(E_3 \mid yes)\, P(E_4 \mid yes)\, P(yes)}{P(E)}$$

$$P(no \mid E) = \frac{P(E_1 \mid no)\, P(E_2 \mid no)\, P(E_3 \mid no)\, P(E_4 \mid no)\, P(no)}{P(E)}$$

P(yes)=9/14=0.643 P(no)=5/14=0.357

P(E1|yes)=2/9=0.222 P(E1|no)=3/5=0.6
P(E2|yes)=4/9=0.444 P(E2|no)=2/5=0.4
P(E3|yes)=6/9=0.667 P(E3|no)=1/5=0.2
P(E4|yes)=6/9=0.667 P(E4|no)=2/5=0.4

$$P(yes \mid E) = \frac{0.222\ 0.444\ 0.667\ 0.668\ 0.443}{P(E)} = \frac{0.028}{P(E)} \qquad P(no \mid E) = \frac{0.6\ 0.4\ 0.2\ 0.4\ 0.357}{P(E)} = \frac{0.007}{P(E)}$$

Hence, the Naïve Bayes classifier predicts buys_computer=yes for the new example.

Therefore, X belongs to class ("buys_computer = yes")

Naïve Bayesian Classifier: An Example

- $P(C_i)$: P(buys_computer = “yes”) = 9/14 = 0.643
  P(buys_computer = “no”) = 5/14= 0.357

- Compute $P(X|C_i)$ for each class
  P(age = “<=30” | buys_computer = “yes”) = 2/9 = 0.222
  P(age = “<= 30” | buys_computer = “no”) = 3/5 = 0.6
  P(income = “medium” | buys_computer = “yes”) = 4/9 = 0.444
  P(income = “medium” | buys_computer = “no”) = 2/5 = 0.4
  P(student = “yes” | buys_computer = “yes) = 6/9 = 0.667
  P(student = “yes” | buys_computer = “no”) = 1/5 = 0.2
  P(credit_rating = “fair” | buys_computer = “yes”) = 6/9 = 0.667
  P(credit_rating = “fair” | buys_computer = “no”) = 2/5 = 0.4

- **X = (age <= 30 , income = medium, student = yes, credit_rating = fair)**

$P(X|C_i)$: P(X|buys_computer = “yes”) = 0.222 x 0.444 x 0.667 x 0.667 = 0.044
P(X|buys_computer = “no”) = 0.6 x 0.4 x 0.2 x 0.4 = 0.019

$P(X|C_i)*P(C_i)$: P(X|buys_computer = “yes”) * P(buys_computer = “yes”) = 0.028
P(X|buys_computer = “no”) * P(buys_computer = “no”) = 0.007

Therefore, X belongs to class (“buys_computer = yes”)

Car theft Example

- Attributes are Color , Type , Origin, and the subject, stolen can be either yes or no

Example No.	Color	Type	Origin	Stolen?
1	Red	Sports	Domestic	Yes
2	Red	Sports	Domestic	No
3	Red	Sports	Domestic	Yes
4	Yellow	Sports	Domestic	No
5	Yellow	Sports	Imported	Yes
6	Yellow	SUV	Imported	No
7	Yellow	SUV	Imported	Yes
8	Yellow	SUV	Domestic	No
9	Red	SUV	Imported	No
10	Red	Sports	Imported	Yes

Classify a Red Domestic SUV.

Random forest classifier

What is Random Forest?

- Supervised learning algorithm
- Forest - Ensemble of decision trees, usually trained with the “bagging” method.
- **Builds multiple decision trees and merges them together to get a more accurate and stable prediction.**

Ensemble learning

- *What?*
  - *Ensemble models in machine learning combine the decisions from multiple models to improve the overall performance.*
- *How?*
  - *Taking the mode of the results – majority voting*
  - *Taking weighted average of the results*

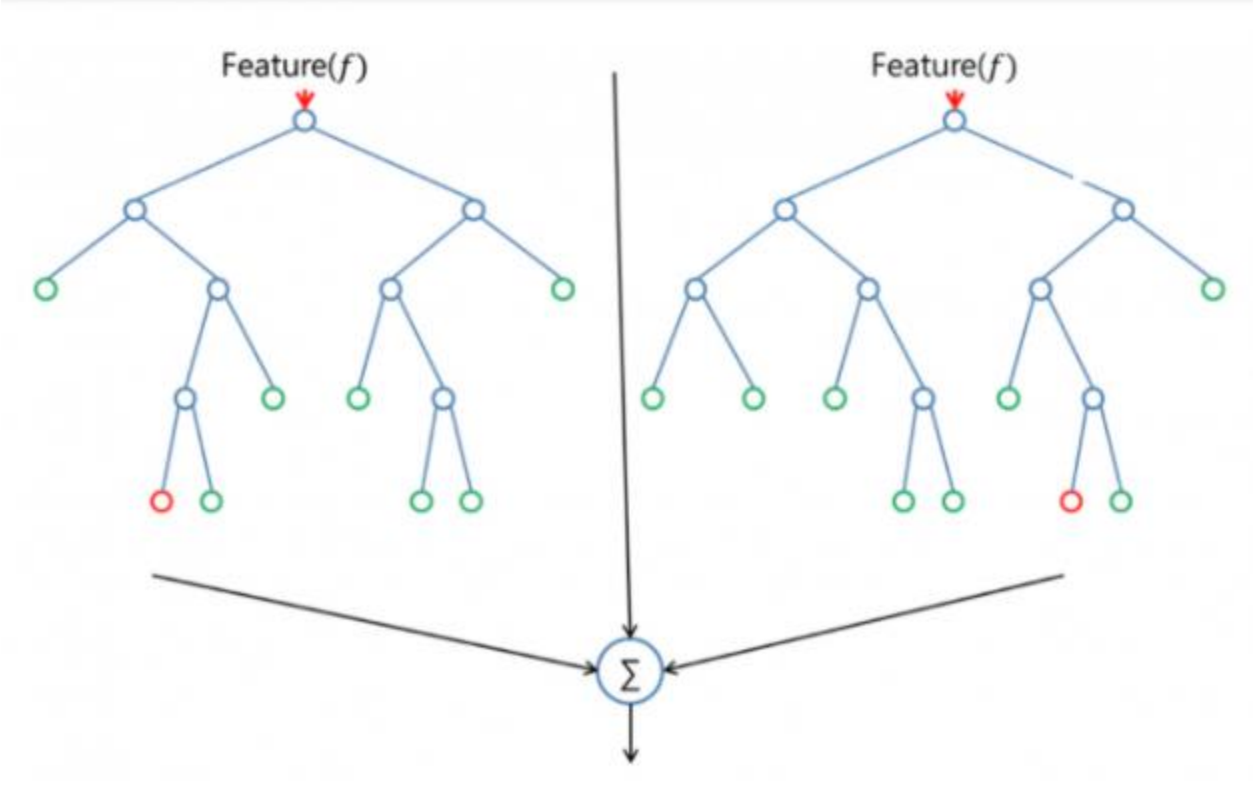
Bagging?

- Bootstrap AGGregatING
  - Create random samples of the training data set with replacement (sub sets of training data set).
  - Build a model (classifier or Decision tree) for each sample.
  - Combine the results of these multiple models using average or majority voting.

Random Forest Classifier

- Random forest adds additional randomness to the model, while growing the trees.
- Only a random subset of the features is taken into consideration by the algorithm for splitting a node.
- Randomly selects observations and features to build several decision trees and then averages the results.
- This results in a wide diversity that generally results in a better model.

Random Forest Classifier



Random Forest Algorithm

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of **ensemble learning,** which is a process of *combining multiple classifiers to solve a complex problem and to improve the performance of the model.*

As the name suggests, ***"Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset."*** Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.

The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.

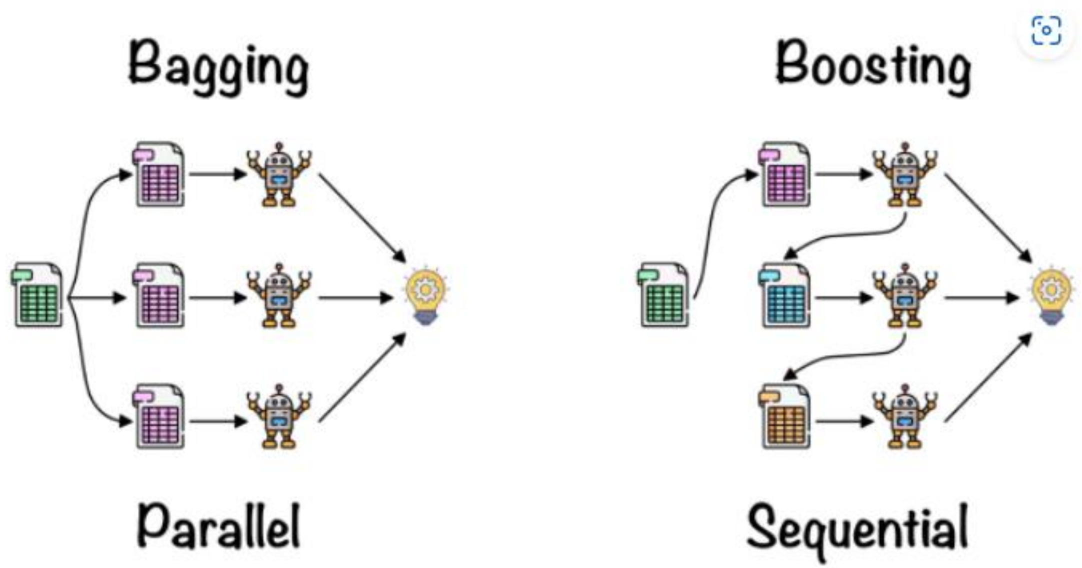
Working of Random Forest Algorithm

Ensemble simply means combining multiple models. Thus a collection of models is used to make predictions rather than an individual model.

Ensemble uses two types of methods:

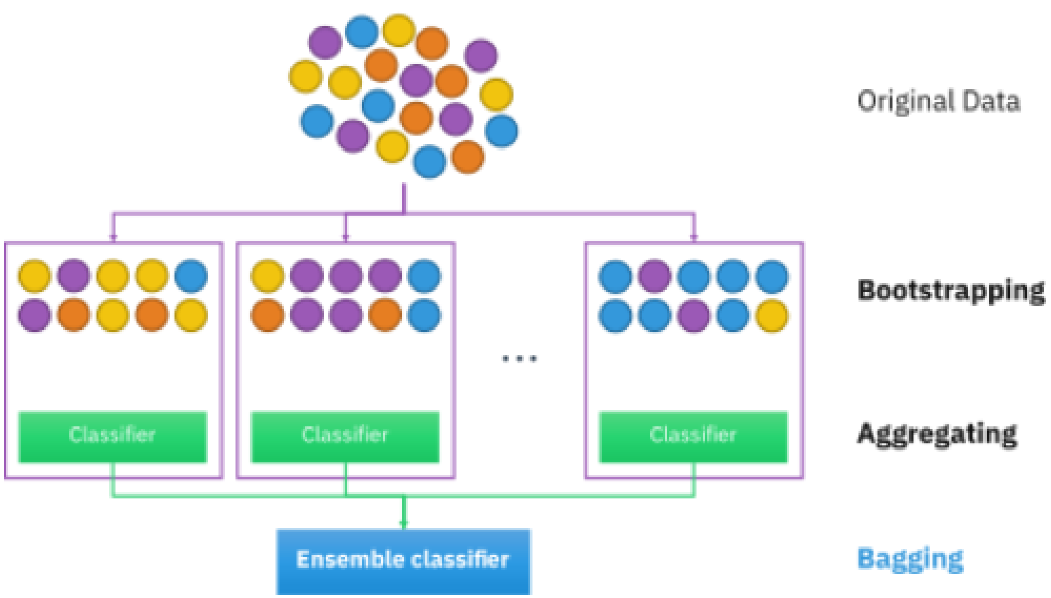
Bagging– It creates a different training subset from sample training data with replacement & the final output is based on majority voting. For example, Random Forest.

2. **Boosting**– It combines weak learners into strong learners by creating sequential models such that the final model has the highest accuracy. For example, ADA BOOST, XG BOOST.



Bagging

Bagging, also known as ***Bootstrap Aggregation,*** is the ensemble technique used by random forest. Bagging chooses a random sample/random subset from the entire data set. Hence each model is generated from the samples (Bootstrap Samples) provided by the Original Data with replacement known as ***row sampling***. This step of row sampling with replacement is called ***bootstrap***.
Now each model is trained independently, which generates results. The final output is based on majority voting after combining the results of all models. This step which involves combining all the results and generating output based on majority voting, is known as ***aggregation***.



Steps Involved in Random Forest Algorithm

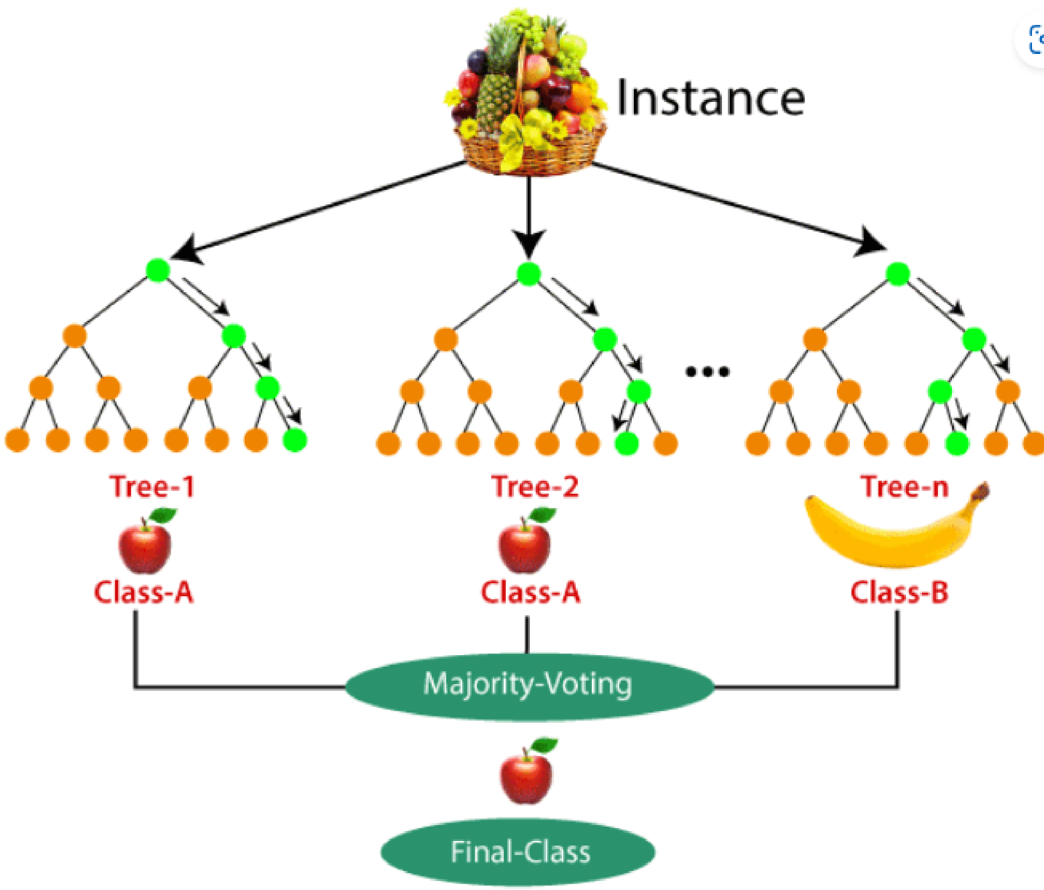
Step 1: In the Random forest model, a subset of data points and a subset of features is selected for constructing each decision tree. Simply put, n random records and m features are taken from the data set having k number of records.

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: Example



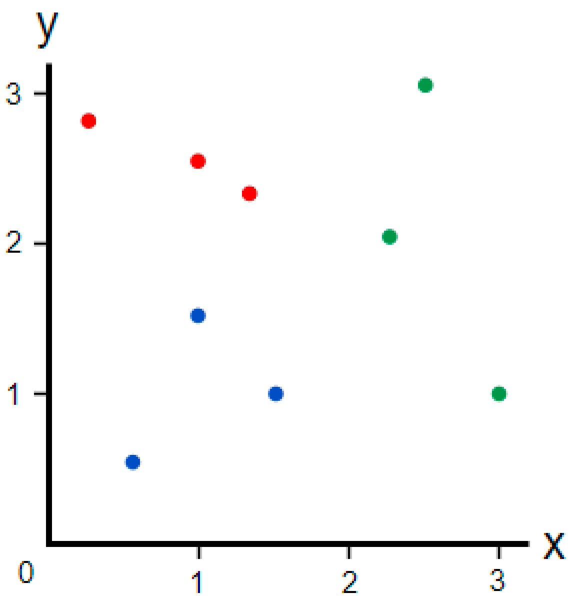
Important Features of Random Forest

- **Diversity:** Not all attributes/variables/features are considered while making an individual tree; each tree is different.
- **Immune to the curse of dimensionality:** Since each tree does not consider all the features, the feature space is reduced.
- **Parallelization:** Each tree is created independently out of different data and attributes. This means we can fully use the CPU to build random forests.
- **Train-Test split:** In a random forest, we don’t have to segregate the data for train and test as there will always be 30% of the data which is not seen by the decision tree.
- **Stability:** Stability arises because the result is based on majority voting/ averaging.

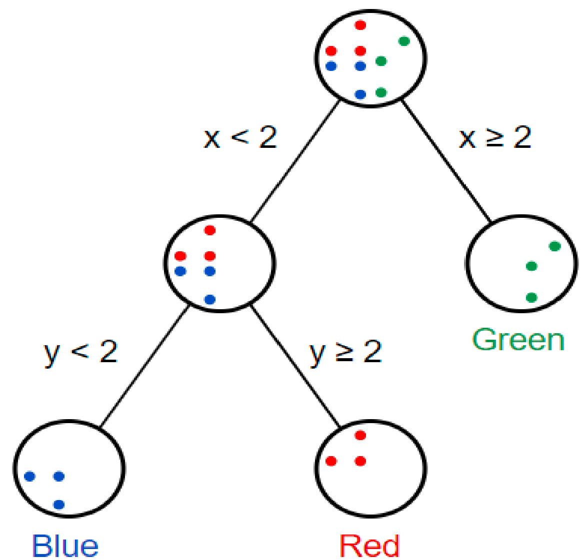
Difference Between Decision Tree and Random Forest

Decision trees	Random Forest
1. Decision trees normally suffer from the problem of overfitting if it's allowed to grow without any control.	1. Random forests are created from subsets of data, and the final output is based on average or majority ranking; hence the problem of overfitting is taken care of.
2. A single decision tree is faster in computation.	2. It is comparatively slower.
3. When a data set with features is taken as input by a decision tree, it will formulate some rules to make predictions.	3. Random forest randomly selects observations, builds a decision tree, and takes the average result. It doesn't use any set of formulas.

Decision Tree Vs. Random Forest



The Dataset v2



Decision Tree