# Naïve Bayesian Classifier

# Objective

- Probability Theory

- Bayes Theorem

- Naive Bayes Theorem

- Implementation

# Probability Theory

- The measure of the likelihood that an event will occur in a Random Experiment

- Quantified as a number between 0 and 1

  0 → Impossibility

  1 → Certainty

# Terminology

- **Random Experiment**

  A physical situation whose outcome cannot be predicted until it is observed.

- **Sample Space**

  A set of all possible outcomes of a random experiment

- **Conditional Probability P(A|B)**

  A measure of the probability of an event given that another event has already occurred.

- **Independence**

  One event it doesn't affect the probability of the other.

# Bayes Theorem

- To determine the probability of a hypothesis with prior knowledge.
- Depends on the conditional probability.

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

where,
P(A|B) → Posterior probability
P(B|A) → Likelihood probability
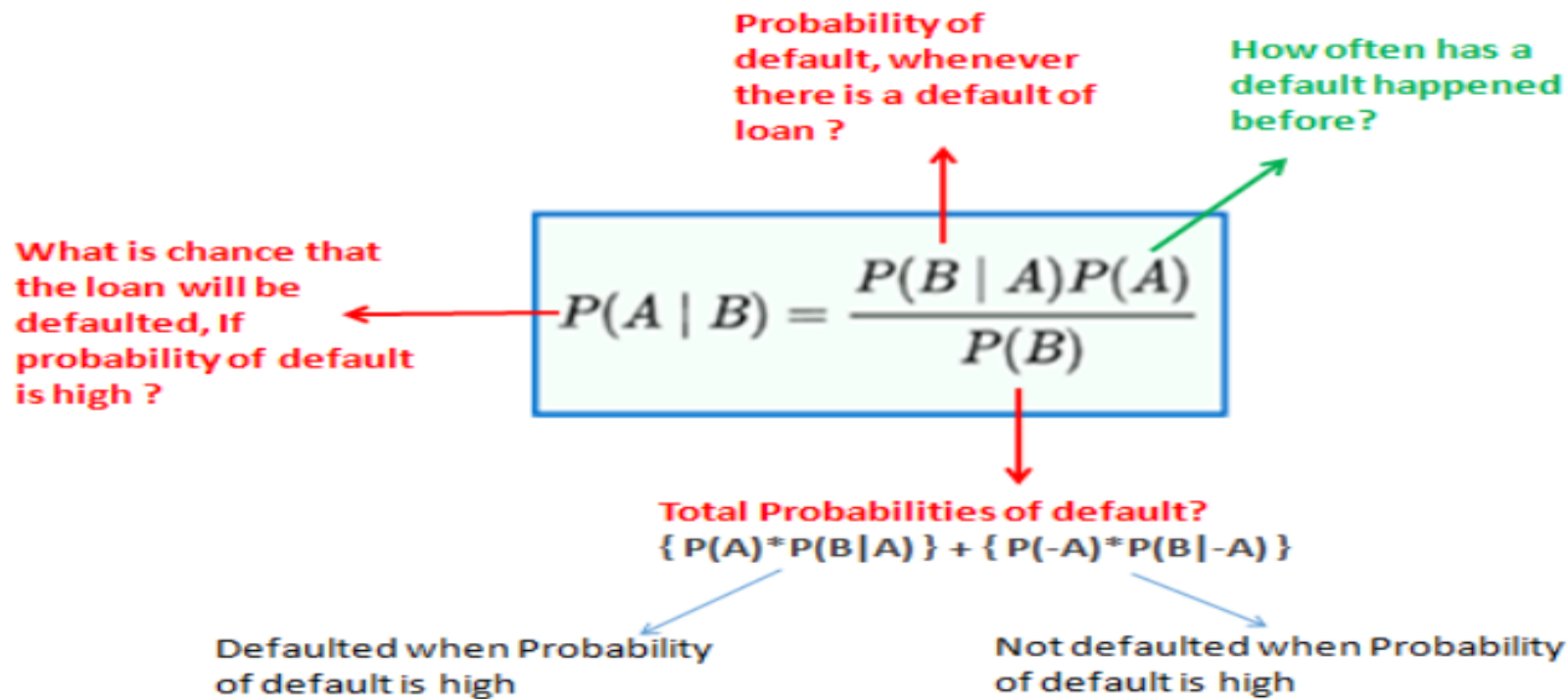P(A) → Prior Probability
P(B) → Marginal Probability

# Naive Bayes Theorem

- Supervised learning algorithm.

- Based on **Bayes theorem.**

- Helps in building the fast ML models that can make quick predictions.

- Probabilistic classifier

- **Examples:** Spam filtration, Sentimental analysis, Classifying articles etc.

# Naive Bayes Theorem Example

Bank Fraud/ Loan Default

Probability of default, whenever there is a default of loan ?

How often has a default happened before?

What is chance that the loan will be defaulted, If probability of default is high ?

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

Total Probabilities of default?
{ P(A)*P(B|A) } + { P(-A)*P(B|-A) }

Defaulted when Probability of default is high

Not defaulted when Probability of default is high

# Types of Naive Bayes Model

- Gaussian

- Multinomial

- Bernoulli

# Naive Bayes Classifier Implementation

- Building the classifier and testing the output.

```
from sklearn.naive_bayes import GaussianNB
classifier = GaussianNB()
classifier.fit(X_train, y_train)


y_pred = classifier.predict(X_test)
```

- Summary of the predictions made by the classifier

```
print(classification_report(y_test, y_pred))
print(confusion_matrix(y_test, y_pred))
# Accuracy score
from sklearn.metrics import accuracy_score
print('accuracy is',accuracy_score(y_pred,y_test))
```

# Hands On