PRESIDENCY UNIVERSITY

BENGALURU

GAIN MORE KNOWLEDGE
REACH GREATER HEIGHTS

MODULE 2

CLASSIFICATION MODELS

CONTENT

- Classification models
- Decision Tree algorithms using Entropy and Gini Index as measures of node impurity,
- Model evaluation metrics for classification algorithms,
- Logistic regression.
- Multi-class classification
- Class Imbalance problem.
- Naïve Bayes Classifiers
- Naive Bayes model for sentiment classification – An Introduction

Classification Models

WHAT IS CLASSIFICATION ALGORITHM?

- Classification algorithm is a Supervised Learning technique in which a program learns from the given dataset or observations and then classifies new observation into a number of classes or groups. Such as, **Yes or No, 0 or 1, Spam or Not Spam, cat or dog,** etc.
- Classes can be called as targets/labels or categories.
- Unlike regression, the output variable of Classification is a category, not a value, such as "Green or Blue", "fruit or animal".
- Since Classification algorithm is a Supervised learning technique, it takes labeled input data, which means it contains input with the corresponding output.
- In classification algorithm, a discrete output function(y) is mapped to input variable(x), i.e., y=f(x), where y = categorical output

Types of Classifications

- The algorithm which implements the classification on a dataset is known as a **classifier**.
- There are two types of Classifications:
  - **Binary Classifier:** If the classification problem has only two possible outcomes, then it is called as Binary Classifier. **Examples:** YES or NO, MALE or FEMALE, SPAM or NOT SPAM, CAT or DOG, etc.
  - **Multi-class Classifier:** If a classification problem has more than two outcomes, then it is called as Multi-class Classifier. **Example:** Classifications of types of crops, Classification of types of music.

LEARNERS IN CLASSIFICATION PROBLEMS

1. **Lazy Learners**
   - Stores the training dataset and wait until it receives the test dataset.
   - In this case, classification is done on the basis of the most related data stored in the training dataset.
   - It takes less time in training but more time for predictions.
   - **Example:** K-NN algorithm, Case-based reasoning

2. **Eager Learners**
   - Eager Learners develop a classification model based on a training dataset before receiving a test dataset.
   - Unlike Lazy learners, Eager Learner takes more time in learning, and less time in prediction.
   - **Example:** Decision Trees, Naïve Bayes, ANN.

Types of Classification Algorithms

- **Linear Models**
  - Logistic Regression
  - Support Vector Machines

- **Non-linear Models**
  - K-Nearest Neighbours
  - Kernel SVM
  - Naïve Bayes
  - Decision Tree Classification
  - Random Forest Classification

METHODS FOR EVALUATING A CLASSIFICATION MODEL

Log Loss or Cross-Entropy Loss

- It is used for evaluating the performance of a classifier, whose output is a probability value between the 0 and 1.
- For a good binary Classification model, the value of log loss should be near to 0.
- The value of log loss increases if the predicted value deviates from the actual value.
- The lower log loss represents the higher accuracy of the model.
- For Binary classification, cross-entropy can be calculated as

  - (ylog(p)+(1-y)log(1-p))

- where y= Actual output, p= predicted output.

METHODS FOR EVALUATING A CLASSIFICATION MODEL

Confusion Matrix

- The confusion matrix provides us a matrix/table as output and describes the performance of the model.
- It is also known as the error matrix.
- The matrix consists of predictions result in a summarized form, which has a total number of correct predictions and incorrect predictions. The matrix looks like as below table:

	Actual Positive	Actual Negative
Predicted Positive	True Positive	False Positive
Predicted Negative	False Negative	True Negative

$$\text{Accuracy} = \frac{\text{TP+TN}}{\text{Total Population}}$$

METHODS FOR EVALUATING A CLASSIFICATION MODEL

AUC-ROC curve

- ROC curve stands for **Receiver Operating Characteristics Curve** and AUC stands for **Area Under the Curve**.
- It is a graph that shows the performance of the classification model at different thresholds.
- To visualize the performance of the multi-class classification model, we use the AUC-ROC Curve.
- The ROC curve is plotted with TPR and FPR, where TPR (True Positive Rate) on Y-axis and FPR(False Positive Rate) on X-axis.

USES CASES OF CLASSIFICATION ALGORITHMS

- Email Spam Detection
- Speech Recognition
- Identifications of Cancer tumor cells.
- Drugs Classification
- Biometric Identification, etc.

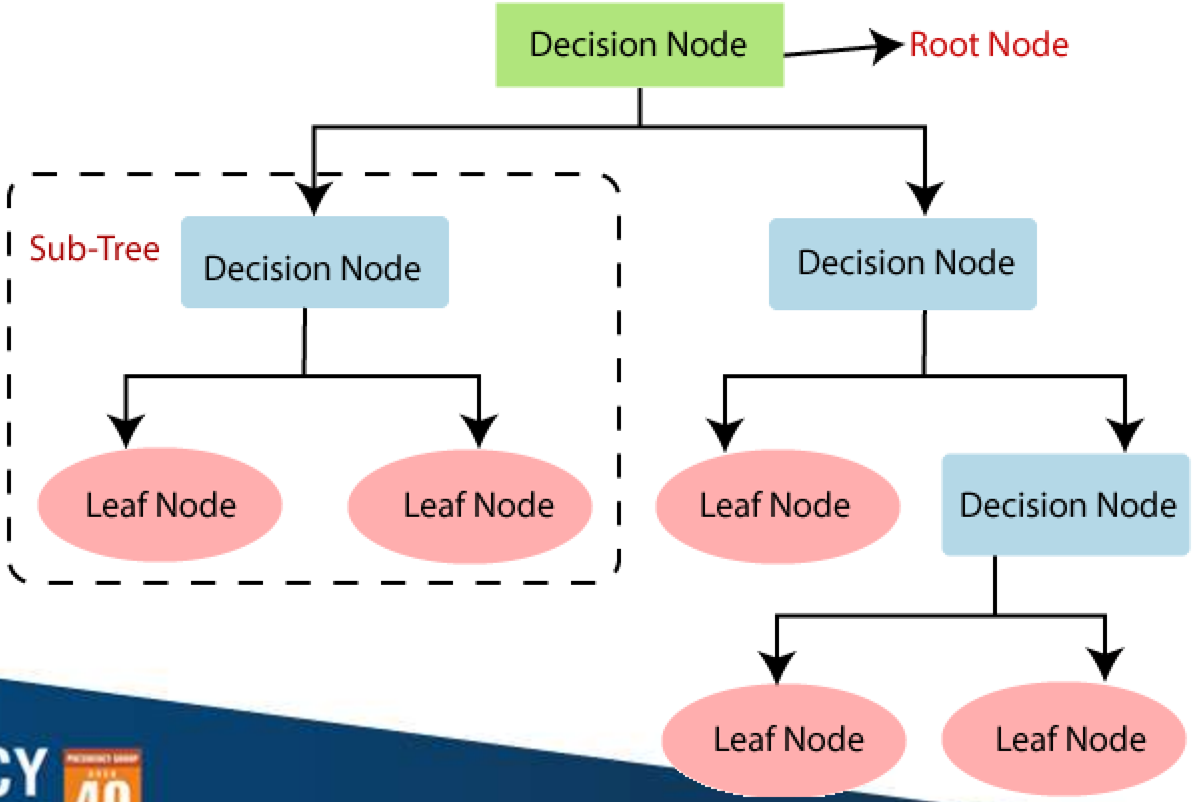
Decision Tree Algorithms using Entropy and Gini Index

What is Decision Tree?

- A **Supervised learning technique** that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems.
- Contains two nodes: **Decision Node** and **Leaf Node.**
- Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.
- It is a tree-structured classifier, where **internal nodes represent the features of a dataset, branches represent the decision rules** and **each leaf node represents the outcome.**
- The decisions or the test are performed on the basis of features of the given dataset.
- **It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.**

CONTD…

- In order to build a tree, we use the **CART algorithm,** which stands for **Classification and Regression Tree algorithm.**
- A decision tree simply asks a question, and based on the answer (Yes/No), it further split the tree into subtrees



SIGNIFICANCE OF DECISION TREE

- Decision Trees usually mimic human thinking ability while making a decision, so it is easy to understand.
- The logic behind the decision tree can be easily understood because it shows a tree-like structure.

Decision Tree Terminologies

- **Root Node:** Node from where the decision tree starts. It represents the entire dataset, which further gets divided into two or more homogeneous sets.

- **Leaf Node:** Final output node and the tree cannot be segregated further after getting a leaf node.

- **Splitting:** Process of dividing the decision node/root node into sub-nodes according to the given conditions.

- **Branch/Sub Tree:** A tree formed by splitting the tree.

- **Pruning:** Pruning is the process of removing, unwanted branches from the tree.

- **Parent & Child node:** The root node of the tree is called the parent node, and other nodes are called the child nodes.

Attribute Selection Measures: Gini Index

- Gini index is a measure of impurity or purity used while creating a decision tree in the CART(Classification and Regression Tree) algorithm.

- An attribute with the low Gini index should be preferred as compared to the high Gini index.

- It only creates binary splits, and the CART algorithm uses the Gini index to create binary splits.

•Gini index can be calculated using the below formula:

$$\text{Gini Index}= 1 - \sum_j P_j^2$$

Numerical Example –
Decision Tree (Entropy, Gini Impurity & Information Gain)

Advantages & Disadvantages of the Decision Tree

Advantages of the Decision Tree

- It is simple to understand as it follows the same process which a human follow while making any decision in real-life.
- It can be very useful for solving decision-related problems and to generate possible outcomes for a problem.
- There is less requirement of data cleaning compared to other algorithms.

Disadvantages of the Decision Tree

- The decision tree contains lots of layers, which makes it complex.
- It has overfitting issue - resolved using the Random Forest algorithm.
- For more class labels, the computational complexity of the decision tree may increase.

MULTI-CLASS CLASSIFICATION

WHAT IS MULTI-CLASS CLASSIFICATION?

- In machine learning and statistical classification, **multiclass classification** or **multinomial classification** is the problem of classifying instances into one of three or more classes (classifying instances into one of two classes is called binary classification).
- While many classification algorithms (notably multinomial logistic regression) naturally permit the use of more than two classes, some are by nature binary algorithms; these can, however, be turned into multinomial classifiers by a variety of strategies.
- Multiclass classification should not be confused with multi-label classification, where multiple labels are to be predicted for each instance.
- The existing multi-class classification techniques can be categorized into **(i) transformation to binary (ii) extension from binary and (iii) hierarchical classification**

MULTI-CLASS CLASSIFICATION TECHNIQUE

- Also called as problem transformation techniques, discusses strategies for reducing the problem of multiclass classification to multiple binary classification problems.
- Transformation to Binary : Categorized into *one vs rest* and *one vs one*.

1. One vs. rest
2. one vs. one
3. Transformation to Binary
4. Strategies of extending the existing binary classifiers to solve multi-class classification problems –
    1. neural networks
    2. decision trees
    3. k-nearest neighbors
    4. Naive Bayes.

Class Imbalance Problem

What is the Class Imbalance Problem?

- It is the problem in machine learning where the total number of a class of data (positive) is far less than the total number of another class of data (negative).
- This problem is extremely common in practice and can be observed in various disciplines including fraud detection, anomaly detection, medical diagnosis, oil spillage detection, facial recognition, etc.

Why is it a Problem?

- Most machine learning algorithms works best when the number of instances of each classes are roughly equal. When the number of instances of one class far exceeds the other, problems arise.
- Example: Given a dataset of transaction data, we would like to find out which are fraudulent and which are genuine ones. For a dataset consisting of 10000 genuine and 10 fraudulent transactions and the classifier classifies fraudulent transactions as genuine transactions.
- The reason for this can be easily explained by the numbers.
- Suppose the machine learning algorithm has two possibly outputs as follows:
  1. Model 1 classified 7 out of 10 fraudulent transactions as genuine transactions and 10 out of 10000 genuine transactions as fraudulent transactions.
  2. Model 2 classified 2 out of 10 fraudulent transactions as genuine transactions and 100 out of 10000 genuine transactions as fraudulent transactions.

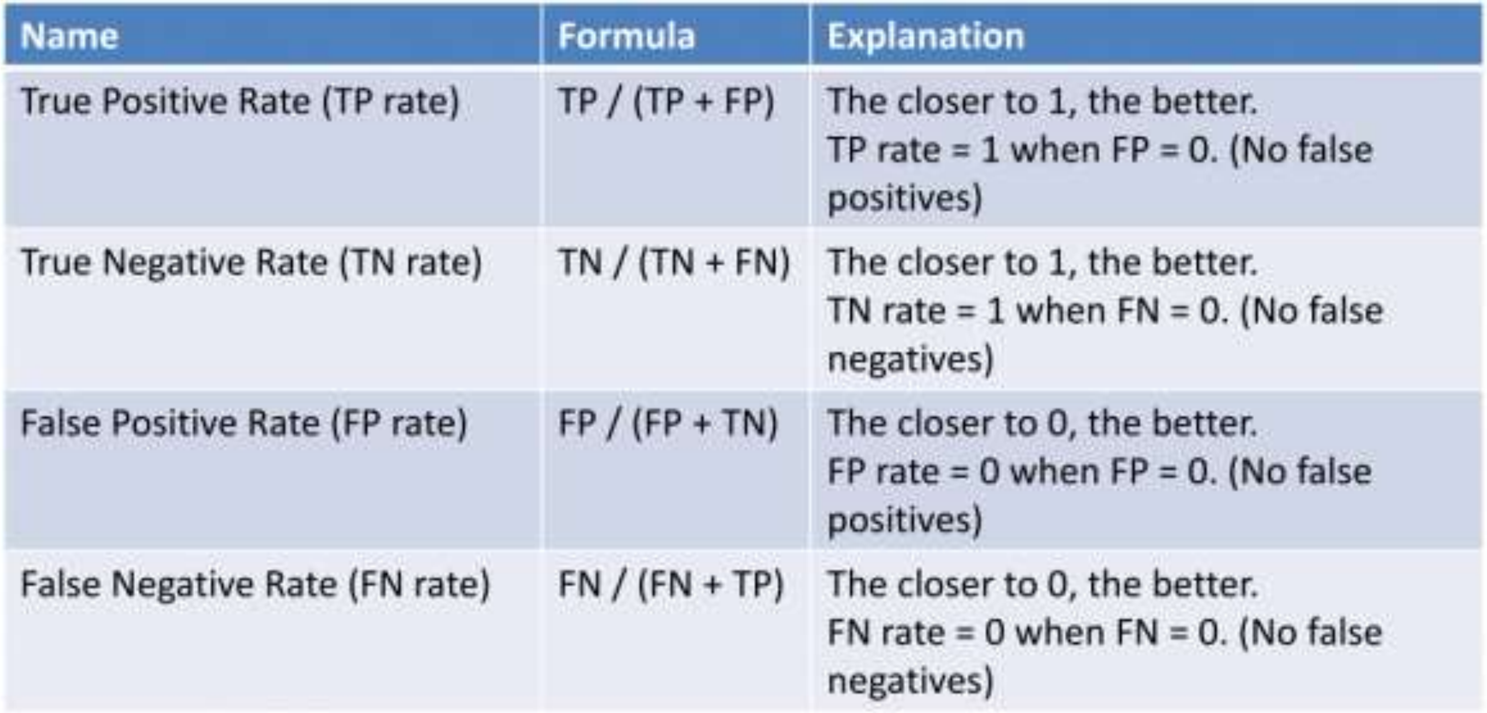
Why is it a Problem?

- If the classifier's performance is determined by the number of mistakes, then clearly Model 1 is better as it makes only a total of 17 mistakes while Model 2 made 102 mistakes.
- However, as we want to minimize the number of fraudulent transactions happening, we should pick Model 2 instead, which only made 2 mistakes classifying the fraudulent transactions.
- But, this could come at the expense of more genuine transactions being classified as fraudulent transactions, and a general machine learning algorithm will just pick Model 1 instead of Model 2, which is a problem.
- In practice, this means we will let a lot of fraudulent transactions go through although we could have stopped them by using Model 2.
- This translates to unhappy customers and money lost for the company.

How to tell a ML algorithm which is a better solution?

- To tell the machine learning algorithm (or the researcher) that Model 2 is better than Model 1, we need to show that Model 2 above is better than Model 1 above. For that, we will need better metrics than just counting the number of mistakes made.
- We introduce the concept of True Positive, True Negative, False Positive and False Negative:
  - True Positive (TP) – An example that is **positive** and is classified correctly as **positive**
  - True Negative (TN) – An example that is **negative** and is classified correctly as **negative**
  - False Positive (FP) – An example that is **negative** but is classified wrongly as **positive**
  - False Negative (FN) – An example that is **positive** but is classified wrongly as **negative**
  - Based on this above. We will have also the following of True Positive Rate, True Negative Rate, False Positive Rate, False Negative Rate:

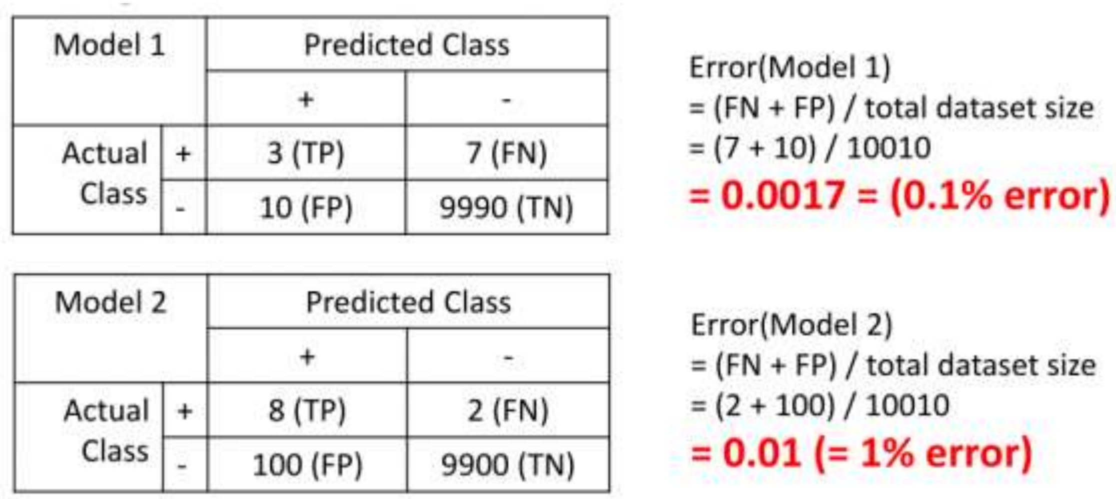
CONTD…



Name	Formula	Explanation
True Positive Rate (TP rate)	TP / (TP + FP)	The closer to 1, the better. TP rate = 1 when FP = 0. (No false positives)
True Negative Rate (TN rate)	TN / (TN + FN)	The closer to 1, the better. TN rate = 1 when FN = 0. (No false negatives)
False Positive Rate (FP rate)	FP / (FP + TN)	The closer to 0, the better. FP rate = 0 when FP = 0. (No false positives)
False Negative Rate (FN rate)	FN / (FN + TP)	The closer to 0, the better. FN rate = 0 when FN = 0. (No false negatives)

With these new metrics, let's compare it with the conventional metrics of counting the number of mistakes made with the example above. First, we will use the old metrics to calculate the number of mistakes made (error):

CONTD…



Model 1		Predicted Class	
		+	-
Actual Class	+	3 (TP)	7 (FN)
	-	10 (FP)	9990 (TN)

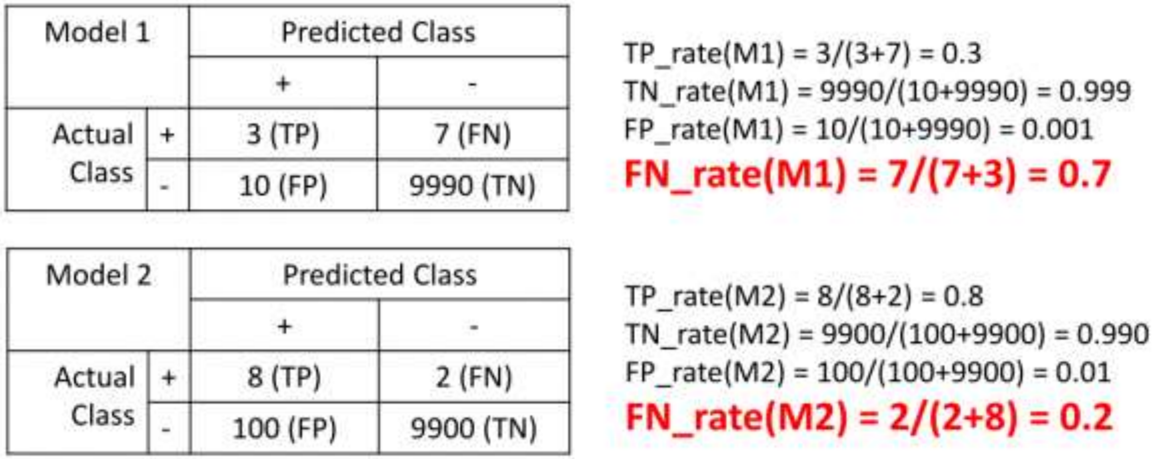
Error(Model 1)
= (FN + FP) / total dataset size
= (7 + 10) / 10010
= 0.0017 = (0.1% error)

Model 2		Predicted Class	
		+	-
Actual Class	+	8 (TP)	2 (FN)
	-	100 (FP)	9900 (TN)

Error(Model 2)
= (FN + FP) / total dataset size
= (2 + 100) / 10010
= 0.01 (= 1% error)

- As illustrated above, Model 1 looks like it has lower error (0.1% error) than Model 2 (1.0% error) but we know that Model 2 is the better one, as it makes less false negatives (FN) (maximize true positive (TP)).
- Now let's see what the performance of Model 1 and Model 2 are like with the new metrics:

CONTD…



Model 1		Predicted Class	
		+	-
Actual Class	+	3 (TP)	7 (FN)
	-	10 (FP)	9990 (TN)

TP_rate(M1) = 3/(3+7) = 0.3
TN_rate(M1) = 9990/(10+9990) = 0.999
FP_rate(M1) = 10/(10+9990) = 0.001
FN_rate(M1) = 7/(7+3) = 0.7

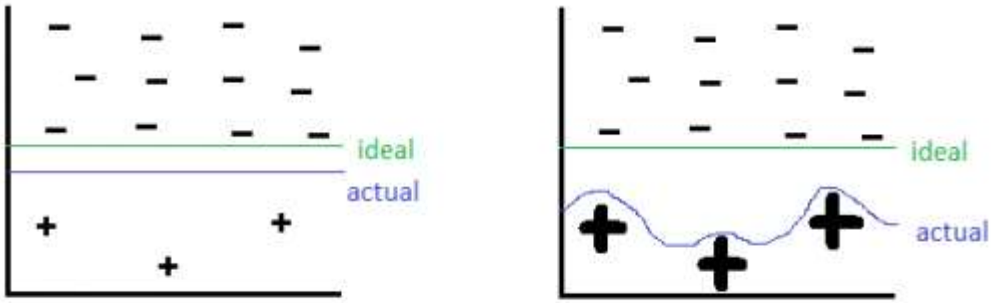
Model 2		Predicted Class	
		+	-
Actual Class	+	8 (TP)	2 (FN)
	-	100 (FP)	9900 (TN)

TP_rate(M2) = 8/(8+2) = 0.8
TN_rate(M2) = 9900/(100+9900) = 0.990
FP_rate(M2) = 100/(100+9900) = 0.01
FN_rate(M2) = 2/(2+8) = 0.2

- Now, we can see that the false negative rate of Model 1 is at 70% while the false negative rate of Model 2 is just at 20%, which is clearly a better classifier.
- This is what we should educate the machine learning algorithm (or us) to use in order to allow it to pick a better algorithm.

SAMPLING BASED APPROACH TO MITIGATE CLASS IMBALANCE PROBLEM

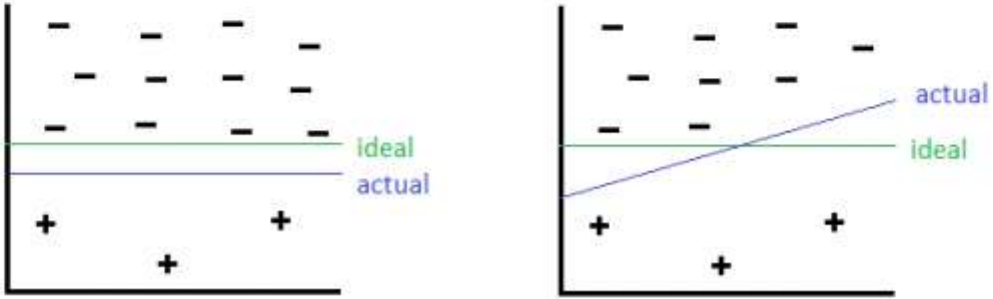
- This can be roughly classified into three categories:
  - ➢ Oversampling
  - ➢ Undersampling
  - ➢ Hybrid, a mix of oversampling and Undersampling
- **Oversampling**
  - ➢ By oversampling, just duplicating the minority classes could lead the classifier to overfitting to a few examples, which can be illustrated below:



CONTD…

- On the left hand side is before oversampling, where as on the right hand side is oversampling has been applied. On the right side, The thick positive signs indicate there are multiple repeated copies of that data instance.
- The machine learning algorithm then sees these cases many times and thus designs to overfit to these examples specifically, resulting in a blue line boundary as above.

- **Undersampling**
  - By Undersampling, we could risk removing some of the majority class instances which is more representative, thus discarding useful information. This can be illustrated as follows:



CONTD...

- Here the green line is the ideal decision boundary we would like to have, and blue is the actual result.
- On the left side is the result of just applying a general machine learning algorithm without using Undersampling.
- On the right, we undersampled the negative class but removed some informative negative class, and caused the blue decision boundary to be slanted, causing some negative class to be classified as positive class wrongly.

- **Hybrid approach**
  - By combining Undersampling and oversampling approaches, we get the advantages but also drawbacks of both approaches as illustrated above, which is still a tradeoff.

Naïve Bayes Classifiers

Naïve Bayes algorithm?

- Naïve Bayes algorithm is a supervised learning algorithm, which is based on Bayes theorem and used for solving classification problems.
- Naïve Bayes Classifier is one of the simple and most effective Classification algorithms which helps in building the fast machine learning models that can make quick predictions.
- It is a probabilistic classifier, which means it predicts on the basis of the probability of an object.
- The Naïve Bayes algorithm is comprised of two words Naïve and Bayes, Which can be described as:
  - ➢ Naïve: It is called Naïve because it assumes that the occurrence of a certain feature is independent of the occurrence of other features. Such as if the fruit is identified on the bases of color, shape, and taste, then red, spherical, and sweet fruit is recognized as an apple. Hence each feature individually contributes to identify that it is an apple without depending on each other.
  - ➢ Bayes: It is called Bayes because it depends on the principle of Bayes' Theorem.

Bayes' Theorem

- Bayes' theorem is also known as Bayes' Rule or Bayes' law, which is used to determine the probability of a hypothesis with prior knowledge. It depends on the conditional probability.
- The formula for Bayes' theorem is given as:

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

Where,

P(A|B) is Posterior probability: Probability of hypothesis A on the observed event B.

P(B|A) is Likelihood probability: Probability of the evidence given that the probability of a hypothesis is true.

P(A) is Prior Probability: Probability of hypothesis before observing the evidence.

P(B) is Marginal Probability: Probability of Evidence.

Working of Naïve Bayes' Classifier

- Working of Naïve Bayes' Classifier can be understood with the help of the below example:
- Suppose we have a dataset of weather conditions and corresponding target variable "Play". So using this dataset we need to decide that whether we should play or not on a particular day according to the weather conditions.
- So to solve this problem, we need to follow the below steps:
    1. Convert the given dataset into frequency tables.
    2. Generate Likelihood table by finding the probabilities of given features.
    3. Now, use Bayes theorem to calculate the posterior probability.

CONTD…

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

Solution: To solve this, first consider the given 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

CONTD…

Frequency table for the Weather Conditions:

Weather	Yes	No
Overcast	5	0
Rainy	2	2
Sunny	3	2
Total	10	5

CONTD…

Likelihood table weather condition:

Weather	No	Yes	
Overcast	0	5	5/14= 0.35
Rainy	2	2	4/14=0.29
Sunny	2	3	5/14=0.35
All	4/14=0.29	10/14=0.71	

APPLYING BAYES' THEOREM

- **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**

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.

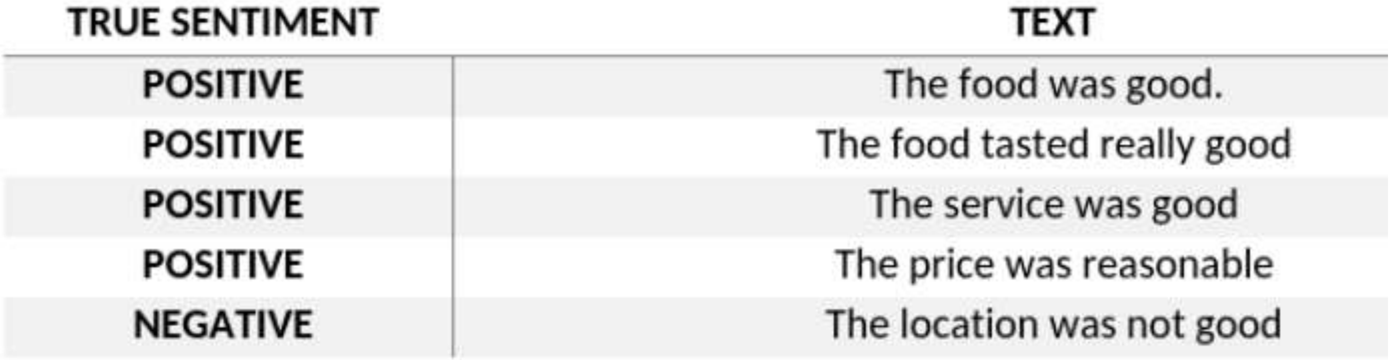
ADVANTAGES, DISADVANTAGES AND APPLICATION

- **Advantages**
  - It can be used for Binary as well as Multi-class Classifications and performs well in Multi-class predictions as compared to the other Algorithms.
  - It is widely used for text classification problems.
- **Disadvantages**: Naive Bayes assumes that all features are independent or unrelated, so it cannot learn the relationship between features.

- **Application**
  - Credit Scoring
  - Medical data classification.
  - In real-time predictions because Naïve Bayes Classifier is an eager learner.
  - Used in Text classification such as Spam filtering and Sentiment analysis.

Naive Bayes Model for Sentiment Classification – An Introduction

- In Naïve Bayes, probabilities are assigned to words or phrases, segregating them into different labels. Consider the following example:



TRUE SENTIMENT	TEXT
POSITIVE	The food was good.
POSITIVE	The food tasted really good
POSITIVE	The service was good
POSITIVE	The price was reasonable
NEGATIVE	The location was not good

- Here, the model will try to learn how these sentiments are classified using corresponding text. Example: It will see that a sentence having the word "good" has a high probability of being appositive sentiment
- Using such a probabilistic value, a total probability of a test sentiment being positive/ negative can be assigned

Thank You