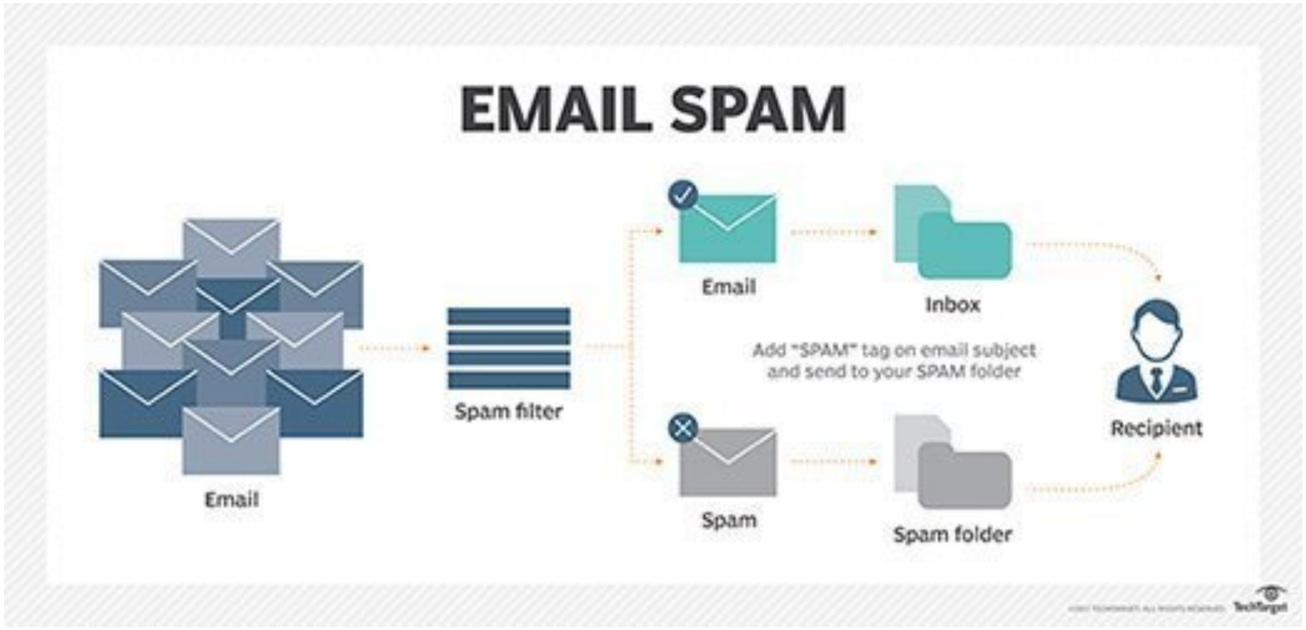
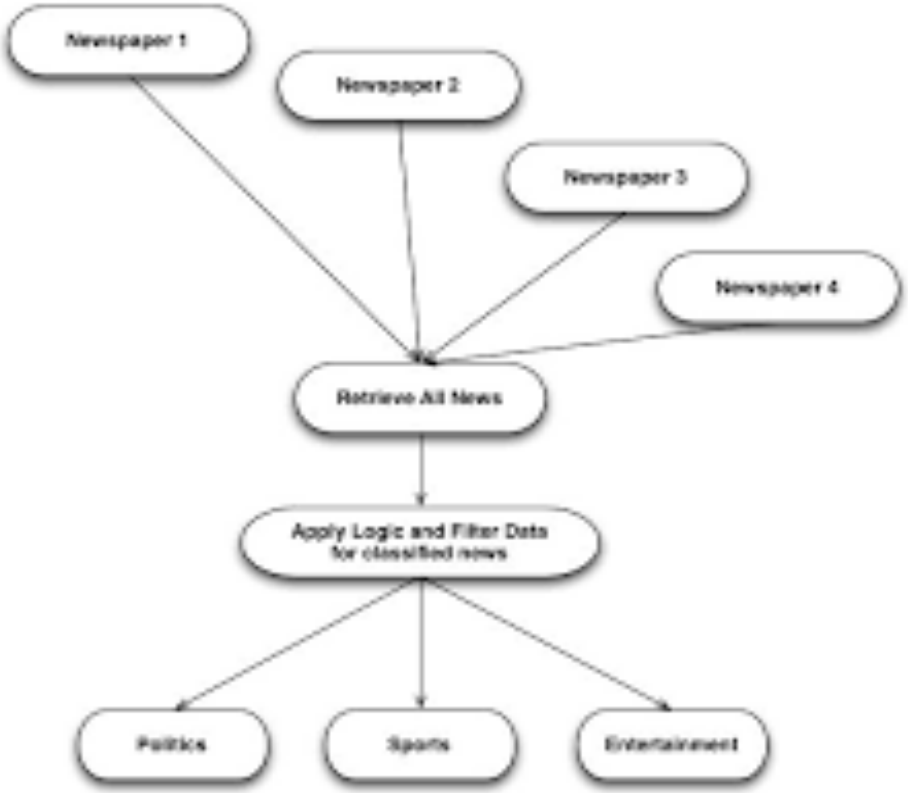
Classification: Naïve Bayes

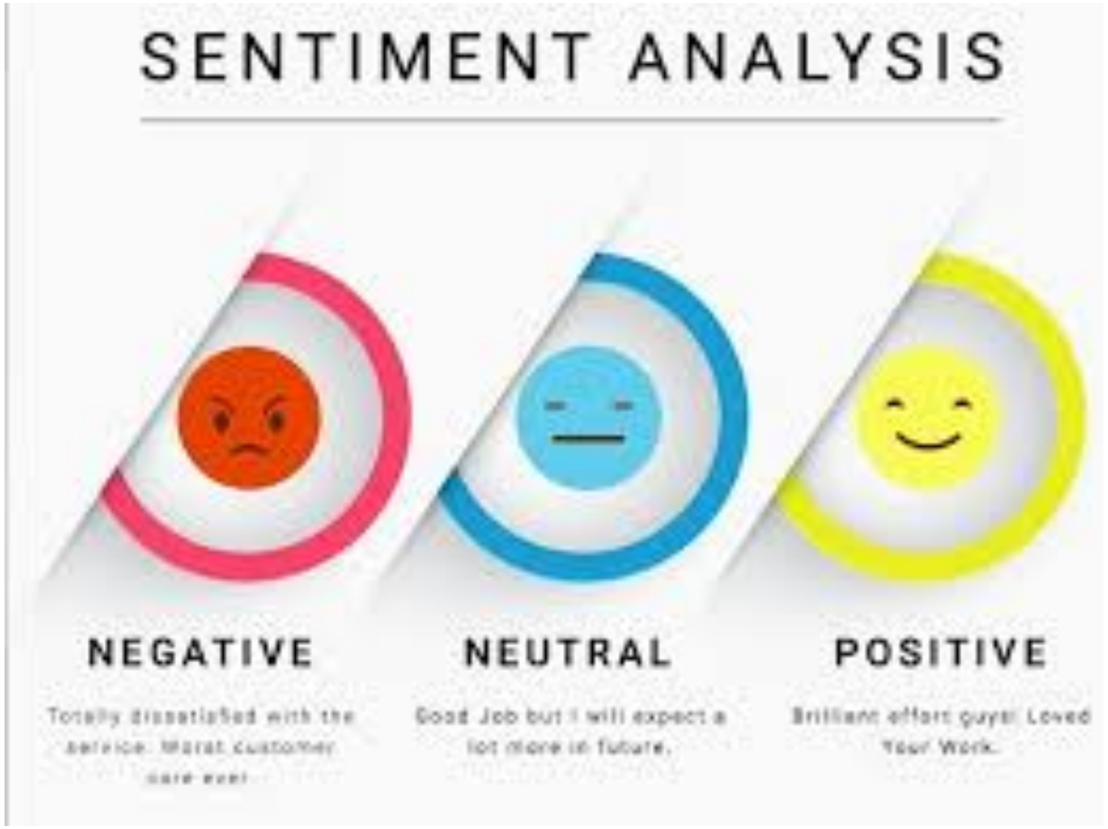
Have you ever thought how your email provider implements spam filtering?

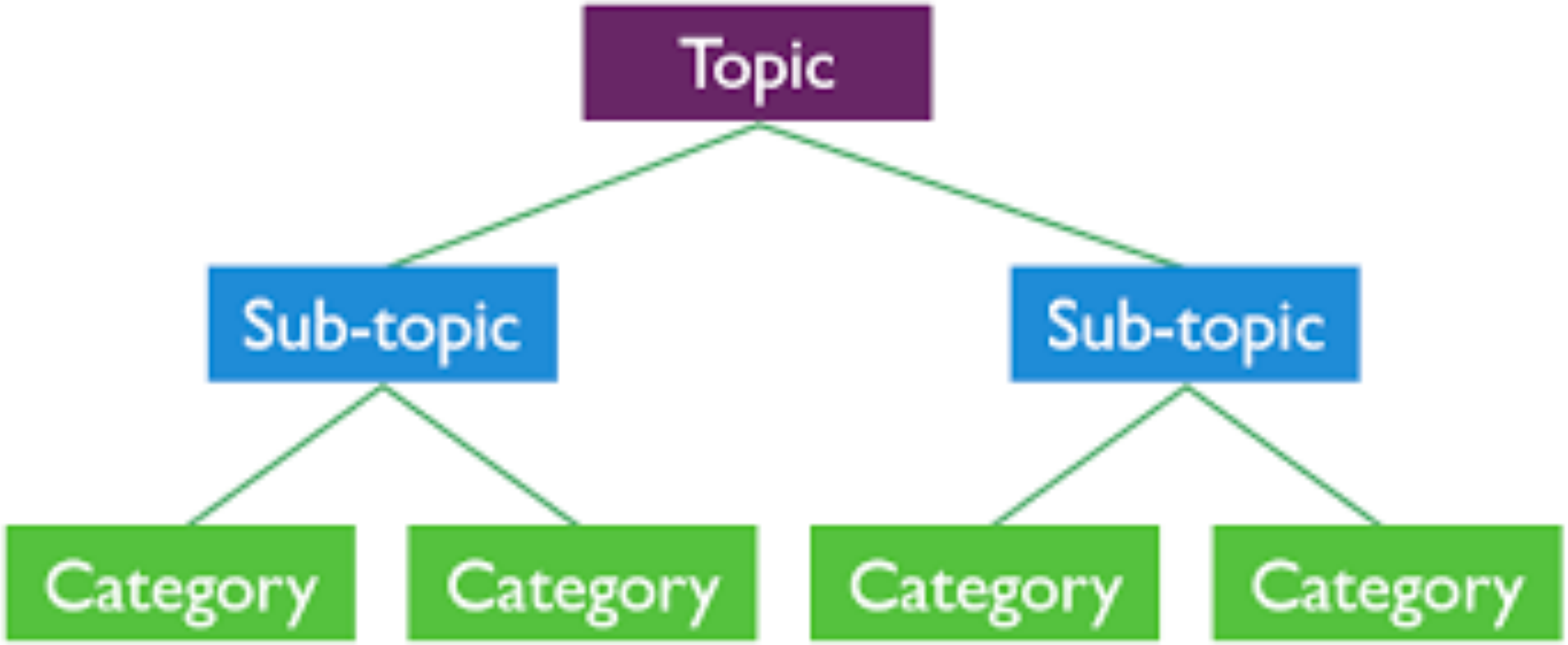


Or how Google or online channels perform news text classification?



Or how companies perform sentiment analysis of views of their audience?


SENTIMENT ANALYSIS
NEGATIVE
Totally dissatisfied with the service. Worst customer care ever.
NEUTRAL
Good Job but I will expect a lot more in future.
POSITIVE
Brilliant effort guys! Loved Your Work.


Topic
Sub-topic
Sub-topic
Category
Category
Category
Category

All the examples have the same kind of problem to classify emails into ham/spam, positive and negative sentiments.

What is classification:

We are given a data, problem is to assign a class to the new data/ unseen data.

Classification is a type of supervised learning. It specifies the class to which data elements belong to

It is used when the output has finite and discrete values

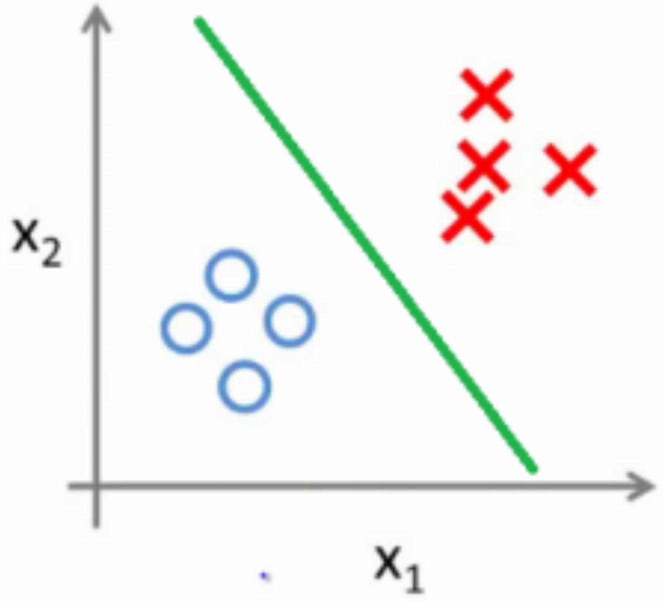
There are two types of classification: binomial and multi-class

Unlike linear regression, not all problems in the world are continuous. There are times when the solution to our problem is of a binary state or some discrete value. The model in this case helps with classifying a new data point and by specifying which category it most closely aligns. These categories can be thought of as qualitative elements. To predict qualitative responses, we use classification.

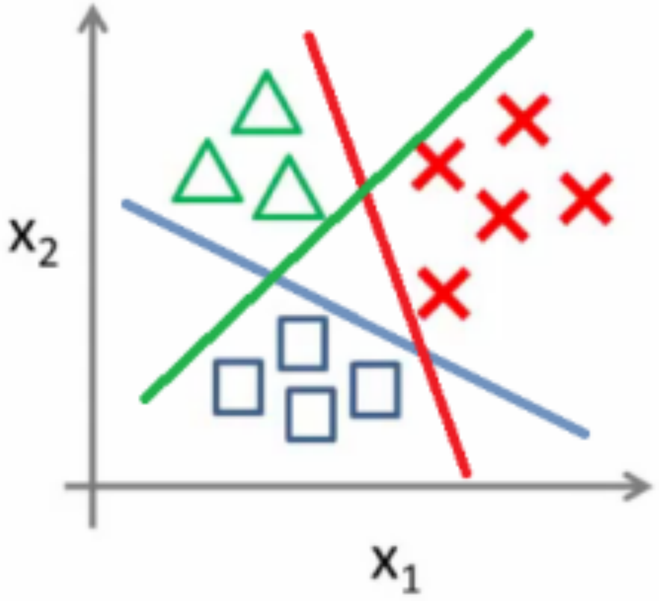
This is different from linear regression, as linear regression is concerned with the prediction of continuous value.

Classification techniques are an essential part of machine learning and data mining applications. Approximately 70-80% of problems in Data Science are classification problems. There are lots of classification problems that are available,

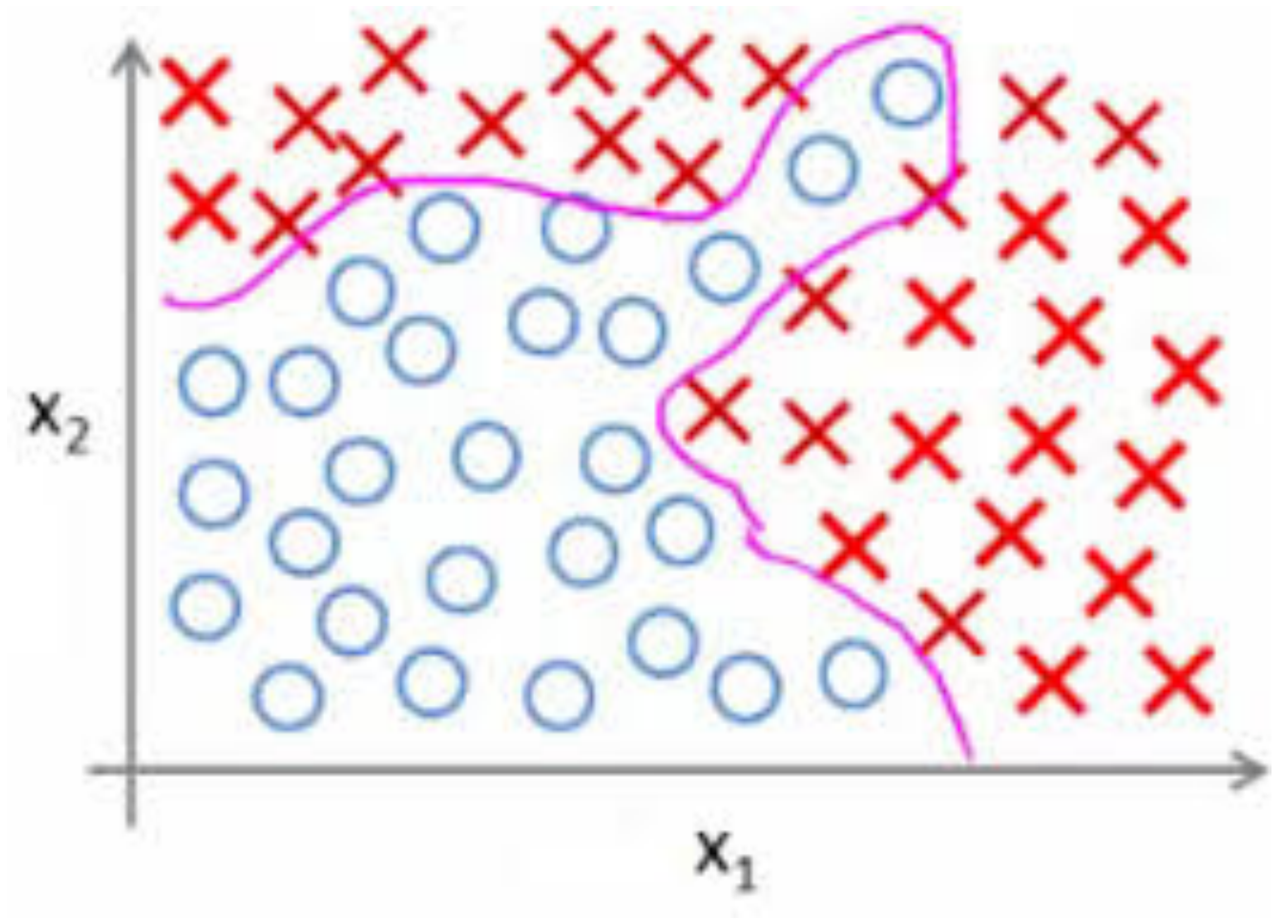
Binary classification:



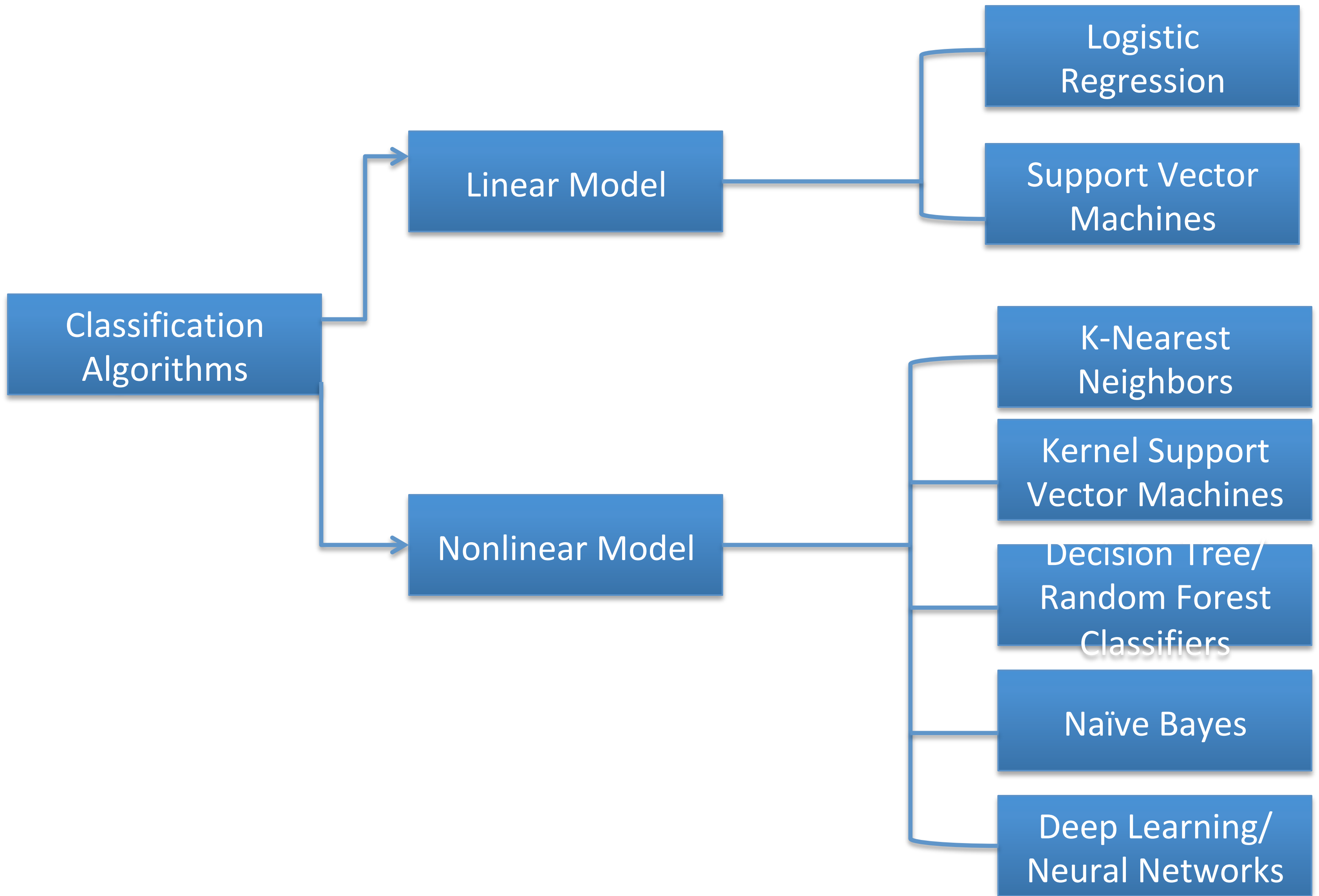
Multi-class classification:



Nonlinear Classification



Types of classification Algorithms


Classification Algorithms
Linear Model
Nonlinear Model
Logistic Regression
Support Vector Machines
K-Nearest Neighbors
Kernel Support Vector Machines
Decision Tree/ Random Forest Classifiers
Naïve Bayes
Deep Learning/ Neural Networks

Linear Classifier – Basic idea

General form of a linear classifier

General form of a linear classifier

$$w_0 + w_1x_1 + w_2x_2 + \ \dots + w_dx_d \ \underset{\text{Class 2}}{\overset{\text{Class 1}}{\gtrless}} \ 0 \quad \text{OR} \quad \sum_{j=0}^{d} w_jx_j \ \underset{\text{Class 2}}{\overset{\text{Class 1}}{\gtrless}} \ 0 \quad (x_0 \overset{\text{def}}{=} 1)$$

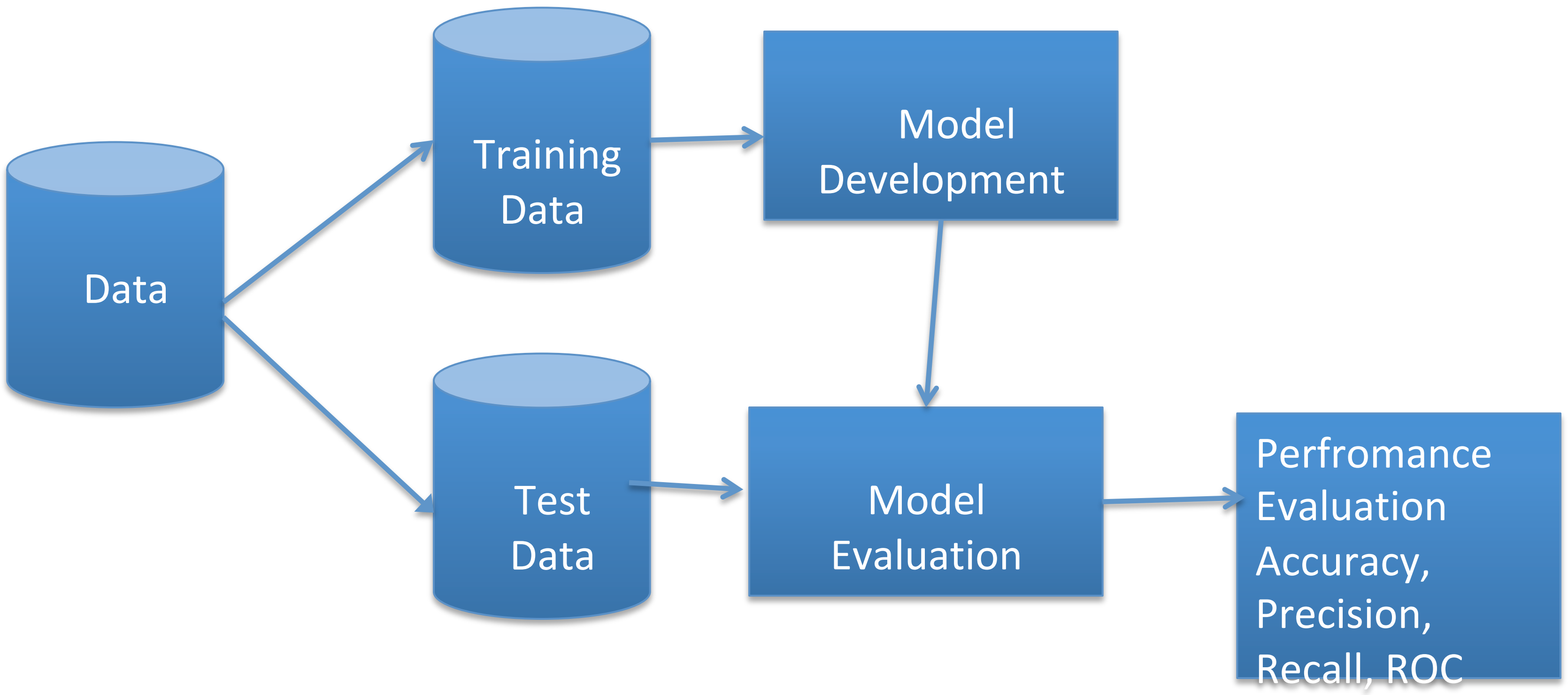
Linear classifier: Given a training set (x,y), find an optimal set of Weights (w1, w2,⋯, wd)

Step-by-step classifier design for 2 classes

Classifier Design

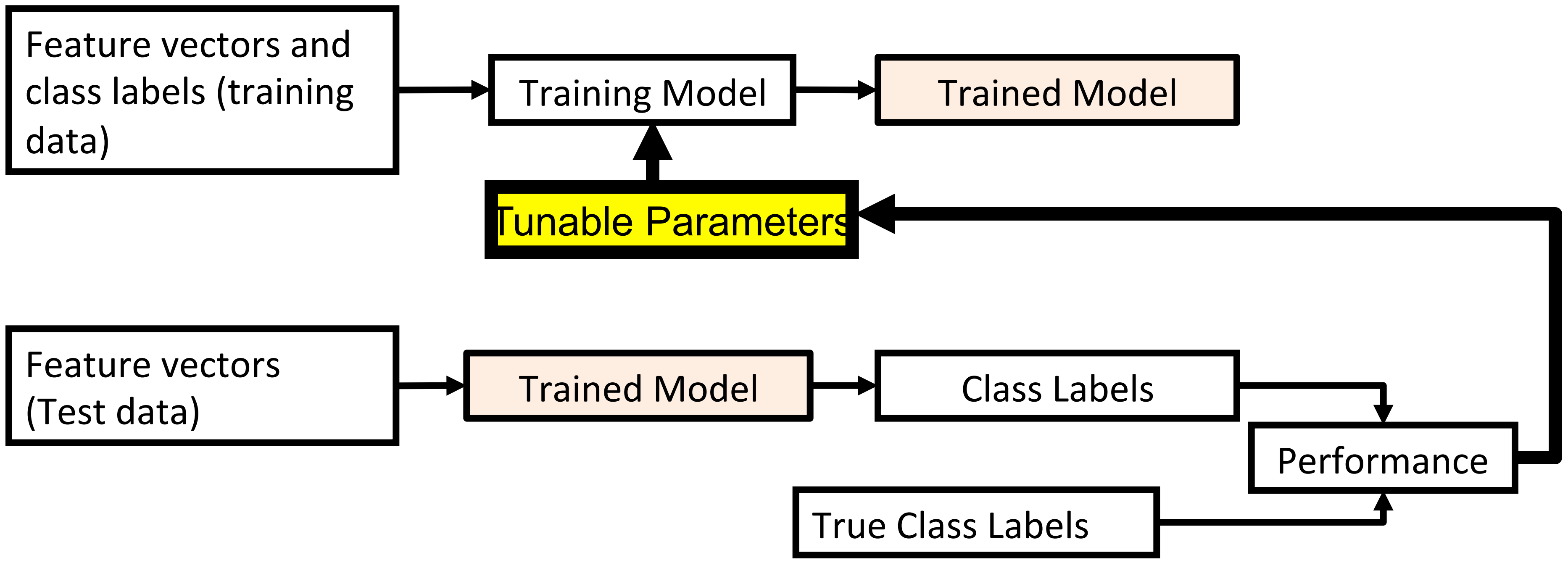
1. Collect and assemble N x d matrix of feature matrix $\mathbf{X}$**.** Create target into $\mathbf{y}$, a column vector of size $\boldsymbol{N}$ **x 1** containing class labels**.**

2. To apply linear classifier to an input vector x, simply compute the augmented feature vector X_a (by appending 1 in front of x) and classify it by computing sign(X_a y)


Data
Training Data
Test Data
Model Development
Model Evaluation
Perfromance Evaluation Accuracy, Precision, Recall, ROC

Training and Testing Subset

We divide the data into training and testing and test the performance of model on the test data.



Answers the question "How good is my classifier?"

Evaluating model performance

- How do we pick which model is the best? Or even whether the model we pick is better than a random guess?

- How reliable are the predicted results

Evaluating model performance

- After doing the usual Feature engineering, selection, implementing a model and getting some output, the next step is to find out how effective is the model based on some metric using test dataset. Different performance metrics are used to evaluate Machine Learning Algorithms.

- The metrics that you choose to evaluate your machine learning model is very important. Choice of metrics influences how the performance of machine learning algorithms is measured and compared.

Evaluating model performance

- Metrics for performance evaluation
  - How to evaluate the performance of a model?
- Methods for performance evaluation
  - How to obtain reliable estimates
- Methods for model comparison
  - How to compare the relative performance among competing model?
- Model selection
  - Which model should we prefer?

Evaluating model performance

Best guess with no model

- First of all, we need to understand the goal of our evaluation.
- Are we trying to pick the best model ?
- Are we trying to quantify the improvement of each model ?
- Regardless of our goal, it is always useful to think about what the baseline should be. Usually the baseline is what is your best guess if you don't have a model.

Evaluating model performance

- For classification problem, one approach is to do a random guess (with uniform probability) but a better approach is to guess the output class that has the largest proportion in the training samples.

- For regression problem, the best guess will be the mean of output of training samples.

Measuring Regression performance

- For regression problem, measuring the distance between the estimated output from the actual output is used to quantify the model's performance.
- Three measures are commonly used: **Root Mean Square Error**, **Relative Square Error** and **Coefficient of Determination**. Typically root mean square is used for measuring the absolute quantity of accuracy.

Mean Square Error MSE = $(1/N) * \sum(\text{yest} - \text{yactual})^2$

Root Mean Square Error RMSE = $(\text{MSE})^{1/2}$

Measuring Regression performance

To measure the accuracy with respect to the baseline, we use the ratio of MSE

Relative Square Error RSE = MSE / MSEbaseline

RSE = $\sum(\text{yactual} - \text{yest})^2 / \sum(\text{yactual} - \text{ymean})^2$

Coefficient Of Determination (also called R square) measures the variance that is explained by the model, which is the reduction of variance when using the model. R square ranges from 0 to 1 while the model has strong predictive power when it is close to 1 and is not explaining anything when it is close to 0.

$R^2 = (\text{MSEbaseline} - \text{MSE}) / \text{MSEbaseline}$

$R^2 = 1 - \text{RSE}$

Measuring Classification performance

- How well can a classifier be expected to perform on novel data?
- Choice of performance measure
- How close is the estimated performance to the true performance?
- Comparing Classifiers

Classification Measures

- Confusion Matrix:

The confusion matrix is one of the most intuitive and easiest metrics used for finding the correctness and accuracy of the classifier. Confusion matrix is a table that is often used to describe the performance of a classification model on a set of test data for which the true values are known. It allows the visualization of the performance of an algorithm.

		Actual	
		Positive(1)	Negative(0)
Predicted	Positive(1)	True positive	False positive
	Negative(0)	False negative	True negative

Classification Measures

A confusion matrix is a summary of prediction results on a classification problem.

The number of correct and incorrect predictions are summarized with count values and broken down by each class.

It gives us insight not only into the errors being made by a classifier but more importantly the types of errors that are being made.

It allows the visualization of the performance of an algorithm. It is used for classification problem where the output can be two or more types of classes. Confusion Matrix (also called a contingency table)

Classification Measures

The confusion matrix is a table with two dimensions (“Actual” and “Predicted”), and sets of classes in both dimensions. Our Actual classifications are columns Predicted ones are Rows.

The confusion matrix in itself is not a performance measure as such, but almost all of the performance metrics are based on the Confusion matrix.

Before diving into what the confusion matrix is all about and what it conveys, let us say we are solving a classification problem where we are predicting whether a person will play tennis or not. Let’s give a label to our target variable:

1: When a person plays, 0: when a person does not play

Measuring Classification performance

Terms associated with confusion matrix:

- **True positives (TP)**: True positives are the cases when the actual class of the data point was 1 (True) and the predicted is also 1 (True)
  - Ex: The case where a person plays(1) and the model classifies as play (1) comes under True positive

- **True Negatives (TN)**: True negatives are the cases when the actual class of the data point was 0(False) and the predicted is also 0(False)
  - Ex: The case where a person does not plays(0) and the model classifies his case as not play comes under True Negatives.

Measuring Classification performance

- **False positives (FP)**: False positives are the cases when the actual class of the data point was 0 (False) and the predicted is 1 (True). False is because the model has predicted incorrectly and positive because the class predicted was a positive one (1)
  - Ex: A person does not play and the model classifies as play (1) comes under False positive

- **False Negatives (FN)**: False negatives are the cases when the actual class of the data point was 1(True) and the predicted is also 0(False). False is because the model has predicted incorrectly and negative because the class predicted was a negative one.
  - Ex: The case where a person plays(1) and the model classifies his case as not play.

Classification Measures

The ideal scenario that we all want is that the model should give 0 False Positives and 0 False Negatives. But that’s not the case in real life as any model will NOT be 100% accurate most of the times.

When to minimize what?

We know that there will be some error associated with every model that we use for predicting the true class of the the target variable. This will results in False Positives and False Negatives.

There is no hard and fast rule that says what should be minimized in all the situations. It purely depends on the business needs and the context of the problem you are trying to solve. Based on that, we might want to minimize either FP or FN

Classifier Performance

- Natural performance measure for classification problems: error rate or accuracy

- Accuracy in classification problems is the number of correct predictions made by the model over all predictions made

- In the numerator, are our correct predictions (TP and TN) and in the denominator, are the kind of all predictions made by the model (right as well as wrong ones).

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Classifier Performance

- Precision refers to the accuracy of positive predictions.

$$\Pr ecision = \frac{TP}{TP + FP}$$

- Recall refers to the ratio of the positive instances that are correctly detected by the classifier (aka True positive rate)

$$\operatorname{Re} call = \frac{TP}{TP + FN}$$

Naïve Bayes

Naive Bayes is a probabilistic machine learning algorithm based on the Bayes Theorem, used in a wide variety of classification tasks. In this lecture, you will gain a clear and complete understanding of the Naive Bayes algorithm for classification.

But why is it called ‘Naive’?

The name naive is used because it assumes the features that go into the model is independent of each other. That is changing the value of one feature, does not directly influence or change the value of any of the other features used in the algorithm.

Naïve Bayes

Naive Bayes does seem to be a simple yet powerful algorithm. But why is it so popular?

That’s because there is a significant advantage with NB. Since it is a probabilistic model, the algorithm can be coded up easily and the predictions made real quick. Real-time quick. Because of this, it is easily scalable and is traditionally the algorithm of choice for real-world applications (apps) that are required to respond to user’s requests instantaneously.

Before we go into the details of the algorithm, we need to understand the concept of conditional probability.

Example

Marbles in a bag

2 blue and 3 red marbles are in a bag

What are the chances of getting a blue marble?

Answer: The chances is 2 in 5. So we can say that probability of getting blue marble is 0.4

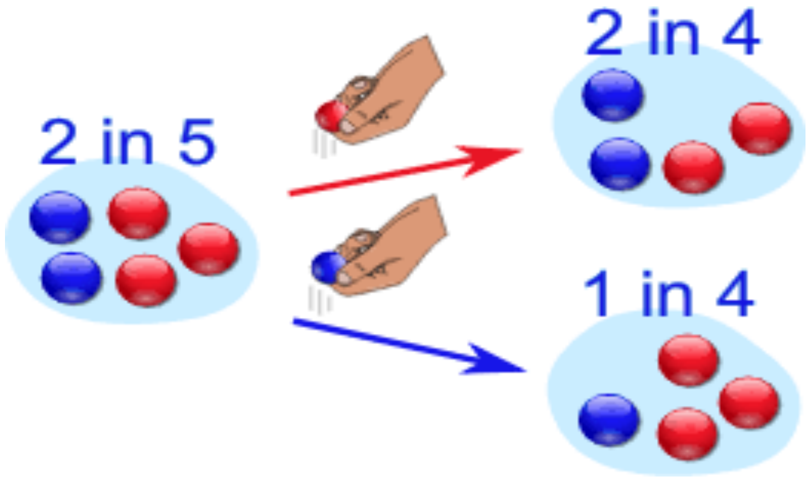
Example

But after taking one out of these chances, situation may change?

So the next time:

If we got red marble before, then the chance of a blue marble next is 2 in 4

If we got a blue marble before, then the chance of a blue marble next is 1 in 4



Example

Suppose there are three bags A,B and C and bag A has 2 red and 4 blues balls, bag B has 1 red and 2 blue balls; and bag C contains 5 red and 4 blue balls. Suppose the probabilities for selecting the bags is not the same but are

- P(A) = 1/3
- P(B) = 1/6
- P(C) = 1/2

Now, let us compute the probability that ball came from bag A if it is red

Example

We need to find out P(A/R) == ???
We will apply Bayes‘ theorem

P(A/R) = P(R/A)*P(A)/P(R)

So we need to calculate some probabilities

- Probability to select red ball P(R)
- Probability to select the bag A i.e. P(A) which is already given 1/3
- Probability to select red ball from A i.e. P(R/A)

Example

P(R) = P(A∧R) + P(**B**∧R) + P(C∧R)

Where

- P(A∧R) is probability to select bag A and red ball
- P(B∧R) is probability to select bagl B and red ball
- P(C∧R) is probability to select bag C and red ball

= P(selecting A) * P(# of red balls/total number of balls in A) +
P(selecting B) * P(# of red balls/total number of balls in B) +
P(selecting C) * P(# of red balls/total number of balls in C)

= 1/3*2/6+1/6*1/3+1/2*5/9
= 4/9

So P(R) = 4/9

Example

P(R/A)

The probability of selecting a red ball given that it Is drawn from bag A is 2/6

P(A) was given 1/3

By putting all the values in formula

P(A/R) = (2/6*1/3)/(4/9) = 0.25

So we can say that if a red ball was drawn that it will be 25% chances that it was from bag 1.

These are the classic examples of conditional probability. So, when we say the conditional probability of A given B, it denotes the probability of A occurring given that B has already occurred.

Mathematically, conditional probability of A given B can be computed as

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

The above equation can be written as:

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

This is knows as Bayes rule

Basic Probability Formulas

Bayes theorem

$$P(h \mid D) = \frac{P(D \mid h)P(h)}{P(D)}$$

Where

P(h/D): Conditional Probability of h given D: posterior probability

P(D/h): Conditional Probability of D given h : likelihood

P(h): Probability of h : prior probability

P(D): Probability of evidence

Maximum A Posterior

Based on Bayes Theorem, we can compute the Maximum A Posterior (MAP) hypothesis for the data

We are interested in the best hypothesis for some space H given observed training data D

$$h_{MAP} = \underset{h \in H}{\operatorname{argmax}} P(h \mid D)$$

$$= \underset{h \in H}{\operatorname{argmax}} \frac{P(D \mid h) P(h)}{P(D)}$$

$$= \underset{h \in H}{\operatorname{argmax}} P(D \mid h) P(h)$$

H: set of all hypothesis

Note that we can drop P(D) as the probability of the data is constant

Naïve Bayes Classifier

- Naïve Bayes is a kind of classifier which uses the Bayes Theorem
- It predicts membership probabilities for each class such as the probability that given record or data point belongs to a particular class.
- The class with the highest probability is considered as the most likely class. This is know as **Maximum A Posteriori (MAP)**

Naïve Bayes

Naïve Bayes is a conditional probability model: given a problem to be classified, represented by a vector x = (x_1, …, x_n) representing some *n* features, it assigns to this instance probabilities

$$p(C_k|x_1, \dots, x_n)$$

For each of *k* possible classes. The problem with the above formulation is that if the number of features *n* is large or if a feature can take on large number of values, then such a model based on probability is impractical. We therefore reformulate the model to make it tractable. Using Bayes' theorem, the conditional probability can be

$$p(C_k|\mathbf{x}) = \frac{p(C_k)\ p(\mathbf{x}|C_k)}{p(\mathbf{x})}.$$

Naïve Bayes

In plain words, the above equation can be written as

$$\text{posterior} = \frac{\text{prior} \times \text{likelihood}}{\text{evidence}}.$$

There is interest only in the numerator because denominator does not depend on C and the values of the features F_i are given, so that the denominator is effectively constant. The numerator is equivalent to the joint probability model

$$p(C_k, x_1, \dots, x_n)$$

Naïve Bayes

Which can be written as follows:

$$\begin{aligned}
p(C_k, x_1, \dots, x_n) &= p(C_k)\ p(x_1, \dots, x_n | C_k) \\
&= p(C_k)\ p(x_1 | C_k)\ p(x_2, \dots, x_n | C_k, x_1) \\
&= p(C_k)\ p(x_1 | C_k)\ p(x_2 | C_k, x_1)\ p(x_3, \dots, x_n | C_k, x_1, x_2) \\
&= p(C_k)\ p(x_1 | C_k)\ p(x_2 | C_k, x_1)\ \dots p(x_n | C_k, x_1, x_2, x_3, \dots, x_{n-1})
\end{aligned}$$

Now the conditional independence assumptions come into play: assume that each feature is conditionally independent of every other features F_j for $j \neq i$, given the category C. This means that

Naïve Bayes

$$p(x_i|C_k, x_j) = p(x_i|C_k)$$

$$p(x_i|C_k, x_j, x_k) = p(x_i|C_k)$$

$$p(x_i|C_k, x_j, x_k, x_l) = p(x_i|C_k)$$

and so on, for $i \neq j, k, l$ Thus the joint model can be expressed as

$$\begin{aligned} p(C_k|x_1, \dots, x_n) &\propto p(C_k, x_1, \dots, x_n) \\ &\propto p(C_k)\ p(x_1|C_k)\ p(x_2|C_k)\ p(x_3|C_k)\ \cdots \\ &\propto p(C_k) \prod_{i=1}^{n} p(x_i|C_k)\,. \end{aligned}$$

Naïve Bayes

This means that under the above independence assumptions, the conditional distribution over the class variable C is :

$$p(C_k|x_1,\dots,x_n) = \frac{1}{Z} p(C_k) \prod_{i=1}^{n} p(x_i|C_k)$$

Where the evidence $Z = p(\mathbf{x})$ is a scaling factor dependent only on $x_1, \dots, x_n$, that is a constant if the values of feature variables are known.

Naïve Bayes

Constructing a classifier from probability model:

Once probability model is available, the naïve Bayes classifier combines the model with decision rule. One common rule is to pick that is most probable; this is known as the maximum a posteriori or MAP decision rule. The corresponding classifier, is the function that assigns a class label $\hat{y} = C_k$ for some *k* as follows:

$$\hat{y} = \underset{k \in \{1, \dots, K\}}{\operatorname{argmax}} \ p(C_k) \prod_{i=1}^{n} p(x_i | C_k).$$

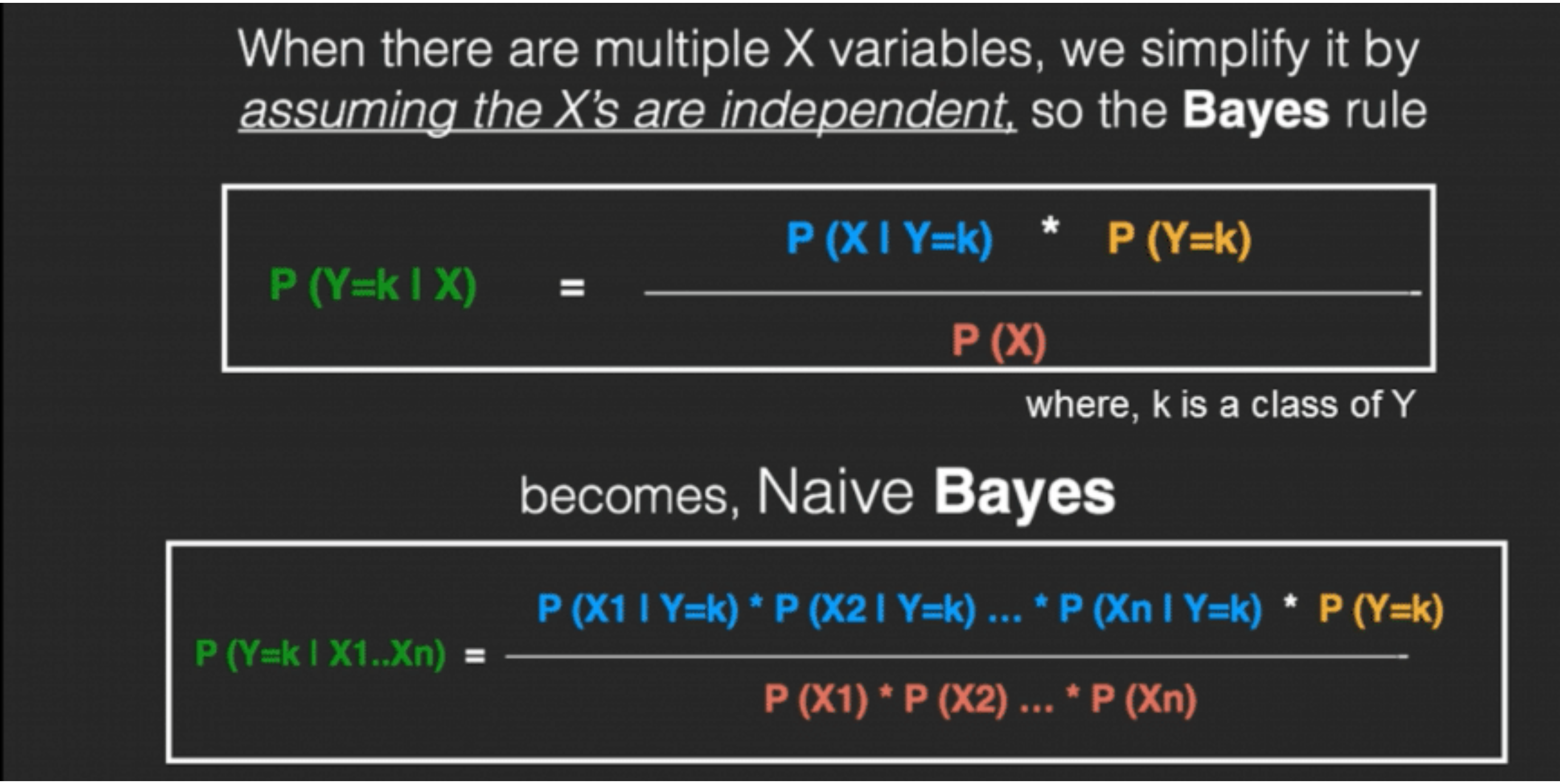
Properties

Estimating $P(x_i \mid c_j)$ instead of $P(x_1, x_2, \ldots, x_n \mid c_j)$ greatly reduces the number of parameters

The learning step in Naïve Bayes consists of estimating $P(x_i \mid c_j)$ based on the frequencies in the training data.

An unseen instance is classified by computing the class that maximizes the posterior.

When conditioned independence is satisfied, Naïve Bayes corresponds to MAP classification.



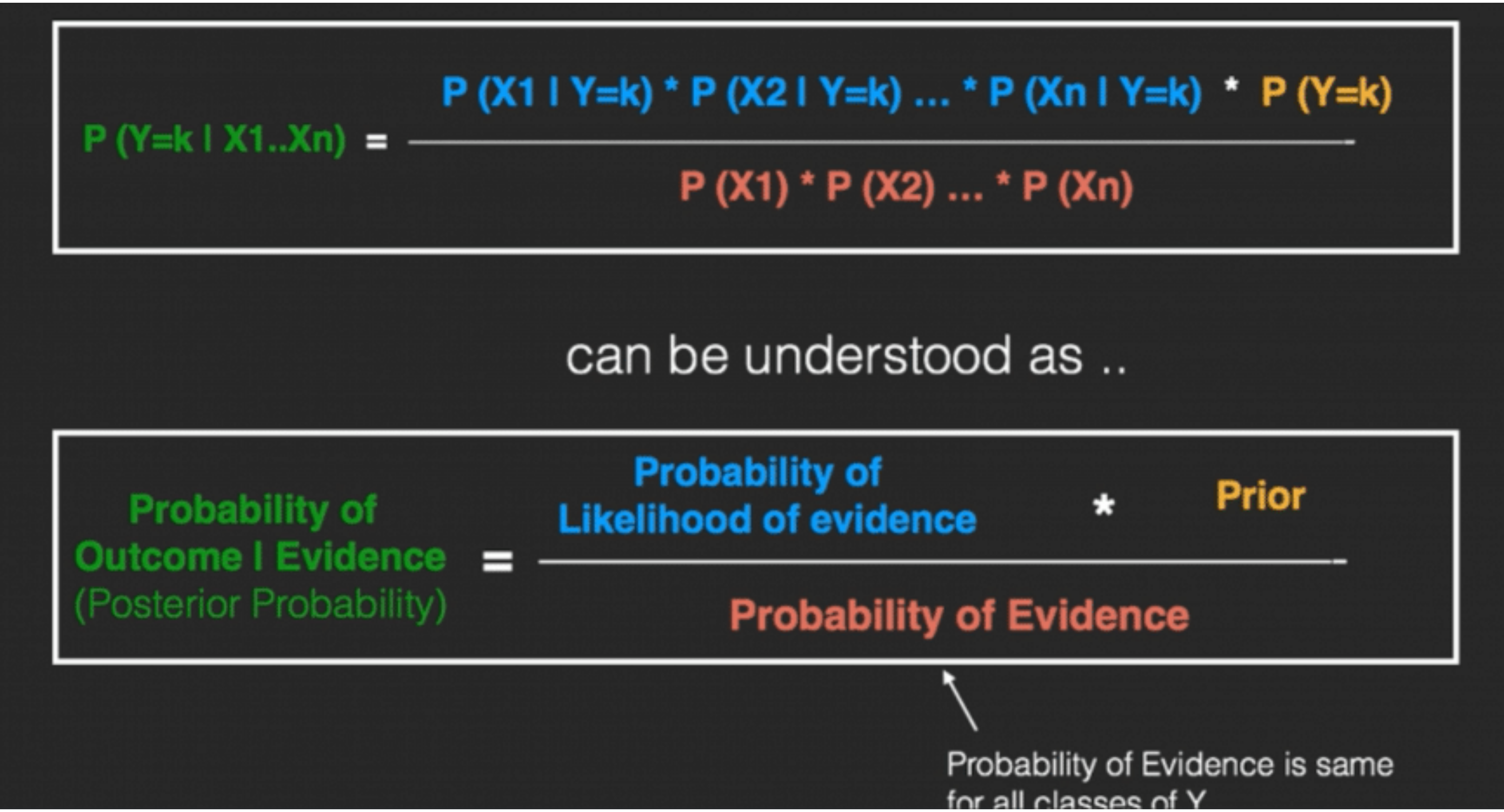
When there are multiple X variables, we simplify it by *assuming the X's are independent*, so the **Bayes** rule

$$\boxed{P(Y=k \mid X) = \frac{P(X \mid Y=k) * P(Y=k)}{P(X)}}$$

where, k is a class of Y

becomes, Naive **Bayes**

$$\boxed{P(Y=k \mid X1..Xn) = \frac{P(X1 \mid Y=k) * P(X2 \mid Y=k) \ldots * P(Xn \mid Y=k) * P(Y=k)}{P(X1) * P(X2) \ldots * P(Xn)}}$$



$$P(Y=k \mid X1..Xn) = \frac{P(X1 \mid Y=k) * P(X2 \mid Y=k) \ldots * P(Xn \mid Y=k) \;\; * \;\; P(Y=k)}{P(X1) * P(X2) \ldots * P(Xn)}$$

can be understood as ..

$$\begin{array}{c}\text{Probability of}\\ \text{Outcome | Evidence}\\ \text{(Posterior Probability)}\end{array} = \frac{\begin{array}{c}\text{Probability of}\\ \text{Likelihood of evidence}\end{array} \;\; * \;\; \text{Prior}}{\text{Probability of Evidence}}$$

Probability of Evidence is same
for all classes of Y

Types of Naïve Bayes Classifier

- **Gaussian:** It is used in classification and it assumes that features follow a normal distribution.

- **Bernoulli:** The binomial model is useful if your attributes/features are binary (i.e., zeros and ones). One application would be text classification with bag of words model where the 1s & 0s are words occur in the document and words does not occur in the document respectively.

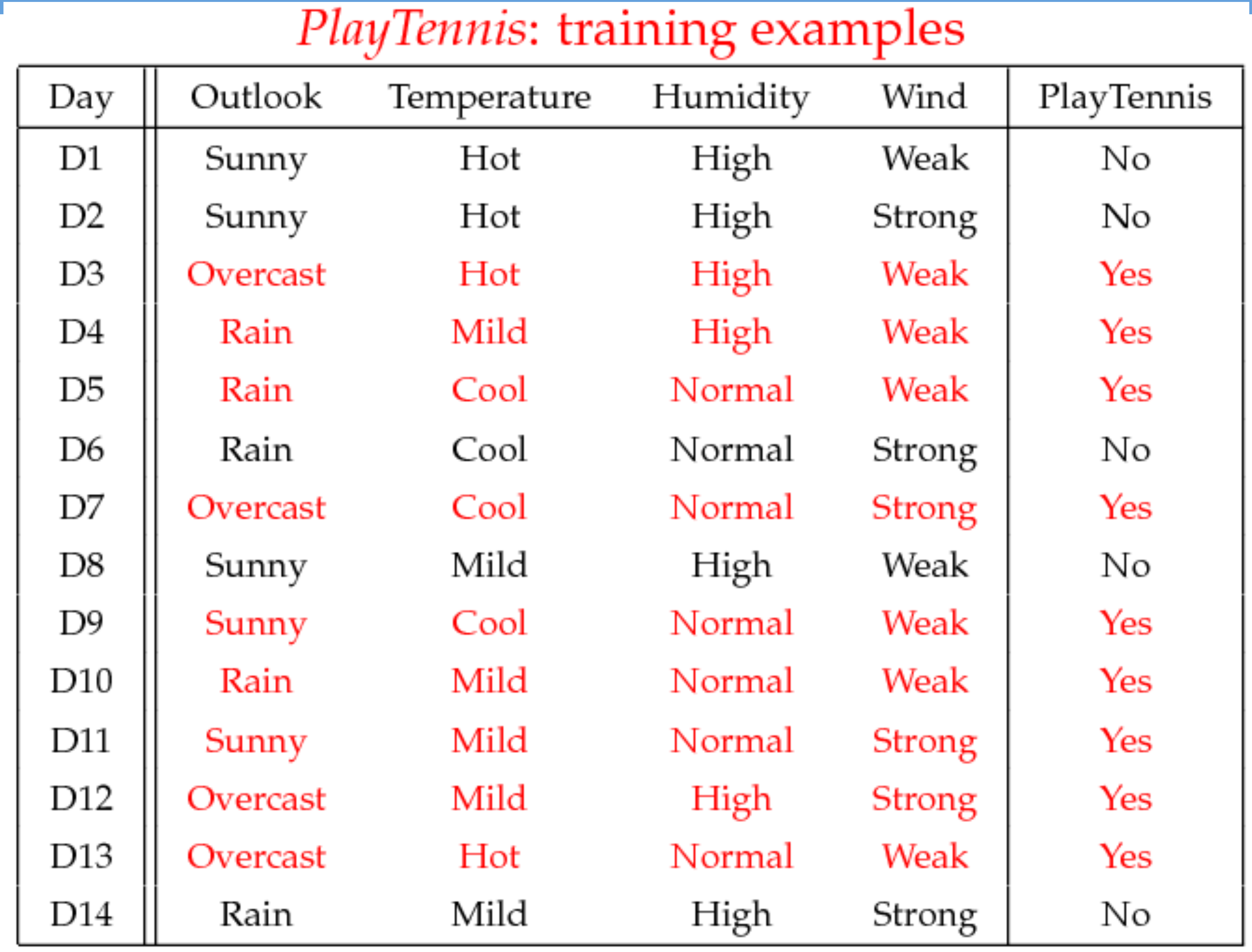
- **Multinomial:** Naïve Bayes is preferred to use on data that is multinomially distributed, e.g., we have a text classification problem. We can consider Bernoulli trials which is one step further and instead of word occurring in the document we have count how often word occurs in the document, you can think of it as number of times outcome number x_i is observed over the n trials.

How Naïve Bayes works?

Naïve Bayes calculates the probability of an event in the following steps:

1. Calculate priori probability for given class labels
2. Calculate conditional probability with each attributes for each class
3. Multiply same class conditional probability
4. Multiply prior probability with probability obtained in step 3
5. See which class has higher probability. Higher probability class belongs to given input set.

Example



PlayTennis: training examples

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Example

Outlook	Play=*Yes*	Play=*No*
Sunny	2/9	3/5
Overcast	4/9	0/5
Rain	3/9	2/5

Temperature	Play=*Yes*	Play=*No*
Hot	2/9	2/5
Mild	4/9	2/5
Cool	3/9	1/5

Humidity	Play=*Yes*	Play=No
High	3/9	4/5
Normal	6/9	1/5

Wind	Play=*Yes*	Play=*No*
Strong	3/9	3/5
Weak	6/9	2/5

P(Play = yes) = 9/14, P(play = No) = 5/14

Example

- Given a new instance,

  x' =(Outlook=*Sunny*, Temperature=*Cool*, Humidity=*High*, Wind=*Strong*)
- Look up tables

P(Outlook=*Sunny*|Play=*Yes*) = 2/9	P(Outlook=*Sunny*|Play=*No*) = 3/5
P(Temperature=*Cool*|Play=*Yes*) = 3/9	P(Temperature=*Cool*|Play==*No*) = 1/5
P(Humidity=*High*|Play=*Yes*) = 3/9	P(Humidity=*High*|Play=*No*) = 4/5
P(Wind=*Strong*|Play=*Yes*) = 3/9	P(Wind=*Strong*|Play=*No*) = 3/5
P(Play=*Yes*) = 9/14	P(Play=*No*) = 5/14

P(*Yes*|**x**'): [P(*Sunny*|*Yes*)P(*Cool*|*Yes*)P(*High*|*Yes*)P(*Strong*|*Yes*)]P(Play=*Yes*) = 0.0053

P(*No*|**x**'): [P(*Sunny*|No) P(*Cool*|No)P(*High*|*No*)P(*Strong*|*No*)]P(Play=*No*) = 0.0206

Given the fact P(*Yes*|**x**') < P(*No*|**x**'), we label **x**' to be "*No*".

Example

Classify whether a given person is a male or a female based on the measured features. The features include height, weight, and foot size

Sex	Height	Weight	Foot size
male	6	180	12
male	5.92	190	11
male	5.58	170	12
male	5.92	165	10
female	5	100	6
female	5.5	150	8
female	5.42	130	7
female	5.75	150	9

Example

Sex	Mean (height)	Variance (height)	Mean (weight)	Variance (weight)	Mean (foot size)	Variance (foot size)
male	5.855	3.5033e-02	176.25	1.2292+02	11.25	9.1667e-01
female	5.4175	9.7225e-02	132.5	5.5833+02	7.5	1.6667e+00

Let us say we have equi-probable classes so P(male) = P(female) = 0.5

Testing sample

Sex	Height	Weight	Foot size
?	6	130	8

Naïve Bayes

We wish to determine which posterior is greater, male or female. For the classification as male the posterior is given by

$$posterior(male) = \frac{P(male)\, p(height|male)\, p(weight|male)\, p(footsize|male)}{evidence}$$

For the classification as female the posterior is given by

$$posterior(female) = \frac{P(female)\, p(height|female)\, p(weight|female)\, p(footsize|female)}{evidence}$$

Example

The evidence may be calculated:

$$\begin{aligned} evidence = {} & P(male)\,p(height|male)\,p(weight|male)\,p(footsize|male) \\ & + P(female)\,p(height|female)\,p(weight|female)\,p(footsize|female) \end{aligned}$$

However, given the sample the evidence is a constant and thus scales both posteriors equally. It therefore does not affect classification and can be ignored. We now determine the probability distribution for the sex of the sample

Example

$$P(male) = 0.5$$

$$p(\text{height}|\text{male}) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{-(6-\mu)^2}{2\sigma^2}\right) \approx 1.5789$$

Where $\mu = 5.855$ and $\sigma^2 = 3.5033 \cdot 10^{-2}$ are the parameters of the normal distribution from the training set.

$$p(\text{weight}|\text{male}) = 5.9881 \cdot 10^{-6}$$

$$p(\text{foot size}|\text{male}) = 1.3112 \cdot 10^{-3}$$

$$\text{posterior numerator (male)} = \text{their product} = 6.1984 \cdot 10^{-9}$$

Example

Similarly we can calculate the posterior probability for female and it is given by

$$5.3778 \times 10^{-4}$$

Since post numerator is greater in the female case, we predict the sample is female

Naïve Bayes

Strengths

- Simple and very effective. It is easy and fast to predict class of test data set. It also performs well in multiclass prediction
- When assumption of independence holds, a Naïve Bayes classifier performs better compare to other models like logistic regression and you need less training data

- It is not among the highest quality classifier but gives good accuracy
- Does well with noisy and missing data. Small changes in the inputs lead to small changes in the probability estimates

- Requires relatively few examples for training, but also works well with very large numbers of examples

- Easy to obtain the estimated probability for prediction.

Naïve Bayes

Weaknesses

- Relies on an often-faulty assumption of equally important and independent features

- Estimated probabilities are less reliable than the predicted classes.

- Class conditional independence, therefore loss of accuracy when the assumption is seriously violated (those highly correlated sets)

Naïve Bayes

When to use

- Comparatively short running time is more important than extreme accuracy
- Very large amount of labeled text is available, so better results can be gotten from a crude algorithm over a large body of examples than a better algorithm over a data set that must be much smaller because of constraints of running time

Advantages

- It is not only a simple approach but also a fast and accurate method for prediction.
- Naive Bayes has very low computation cost.
- It can be used with multiple class prediction problems.
- When the assumption of independence holds, a Naïve Bayes classifier performs better compared to other models like logistic regression.
- Handles both continuous and discrete data
- Highly scalable with number of predictors and data points

Applications

Text classification/ Spam Filtering/ Sentiment Analysis: Naive Bayes classifiers mostly used in text classification (due to better result in multi class problems and independence rule) have higher success rate as compared to other algorithms. As a result, it is widely used in Spam filtering (identify spam e-mail) and Sentiment Analysis (in social media analysis, to identify positive and negative customer sentiments)

Recommendation System: Naive Bayes Classifier and Collaborative Filtering together builds a Recommendation System that uses machine learning and data mining techniques to filter unseen information and predict whether a user would like a given resource or not.

Weather Prediction: A Bayesian based model for weather prediction is used, where posterior probabilities are used to calculate the likelihood of each class label for input data instance and the one with maximum likelihood is considered the resulting output.

Summary

- Naïve Bayes: the conditional independence assumption
  - Training is very easy and fast; just requiring considering each attribute in each class separately
  - Test is straightforward; just looking up tables or calculating conditional probabilities with estimated distributions
- A popular generative model
  - Performance competitive to most of state-of-the-art classifiers even in presence of violating independence assumption
  - Many successful applications, e.g., spam mail filtering
  - A good candidate of a base learner in ensemble learning
  - Apart from classification, naïve Bayes can do more.

Tips to Improve the Model

- If continuous features do not have normal distribution, we should use transformation or different methods to convert it in normal distribution.
- If test data has zero frequency issue, apply smoothing techniques “Laplace Correction” to predict the class of test data set
- Remove correlated features, as the highly correlated features are voted twice in the model and it can lead to over inflating importance

Naïve Bayes Assumption and Why

Theoretically, it is not hard to find P(X|Y). However, it is much harder in reality as the number of features grows.

$X_1 = 1$	$X_2 = 1$	$Y = 1$	$P(X_1 = 1, X_2 = 1 \| Y = 1) =?$
		$Y = 0$	$P(X_1 = 1, X_2 = 1 \| Y = 0) =?$
	$X_2 = 0$	$Y = 1$	$P(X_1 = 1, X_2 = 0 \| Y = 1) =?$
		$Y = 0$	$P(X_1 = 1, X_2 = 0 \| Y = 0) =?$
$X_1 = 0$	$X_2 = 1$	$Y = 1$	$P(X_1 = 0, X_2 = 1 \| Y = 1) =?$
		$Y = 0$	$P(X_1 = 0, X_2 = 1 \| Y = 0) =?$
	$X_2 = 0$	$Y = 1$	$P(X_1 = 0, X_2 = 0 \| Y = 1) =?$
		$Y = 0$	$P(X_1 = 0, X_2 = 0 \| Y = 0) = 1 - others$

7 parameters are needed for a 2-feature binary dataset

$$P(X = (x_1, x_2, \dots, x_k) | Y = y) \text{ is called a parameter in Naive Bayes.}$$

$$If\ x_j\ \forall j\ is\ binary, there\ are\ 2^{k+1} - 1\ parameters$$

Having this amount of parameters in the model is impractical. To solve this problem, a naive assumption is made. We pretend all features are independent. What does this mean?

$$if\ X_1\ and\ X_2\ are\ indep.\ P(X_1, X_2|Y) = P(X_1|Y)P(X_2|Y)$$

Now with the help of this naive assumption (naive because features are rarely independent), we can make classification with much fewer parameters:

$$P(Y|X) = \frac{P(Y)\prod_i P(X_i|Y)}{P(X)}$$

$X_1 = 1$	$Y = 1$	$P(X_1 = 1 \| Y = 1)$
	$Y = 0$	$P(X_1 = 1 \| Y = 0)$
$X_1 = 0$	$Y = 1$	$P(X_1 = 0 \| Y = 1)$ $= 1 - P(X_1 = 0 \| Y = 1)$
	$Y = 0$	$P(X_1 = 0 \| Y = 0)$ $= 1 - P(X_1 = 1 \| Y = 0)$

$X_2 = 1$	$Y = 1$	$P(X_2 = 1 \| Y = 1)$
	$Y = 0$	$P(X_2 = 1 \| Y = 0)$
$X_2 = 0$	$Y = 1$	$P(X_2 = 0 \| Y = 1)$ $= 1 - P(X_2 = 0 \| Y = 1)$
	$Y = 0$	$P(X_2 = 0 \| Y = 0)$ $= 1 - P(X_2 = 1 \| Y = 0)$

Naive Bayes need fewer parameters (4 in this case)

$$If\ x_j\ \forall j\ is\ binary, there\ are\ 2k\ parameters$$