Classification Algorithms

- Logistic Regression
- Naïve Bayes
- Stochastic Gradient Descent
- K-Nearest Neighbours
- Decision Tree
- Random Forest
- Support Vector Machine

WEKA IMPLEMENTATIONS

• Inferring rudimentary rules: *OneR, zeroR and HyperPipes* (learns one rule per class)
• Statistical modeling:
NaïveBayes and many variants, including *NaiveBayesMultinomial*
• Decision trees: *Id3*
• Decision rules: *Prism*
• Association rules (see Section 11.7 and Table 11.8): *apriori*
• Linear models:
  • SimpleLinearRegression, LinearRegression, Logistic (regression)
  • VotedPerceptron, Winnow
• Instance-based learning: *IB1*, *VFI* (voting feature intervals)
• Clustering (see Section 11.6 and Table 11.7): *SimpleKMeans*
• Multi-instance learning: *SimpleMI*, *MIWrapper*

Table 9.1 Top 10 Algorithms in Data Mining

	Algorithm	Category	Book Section
1	C4.5	Classification	4.3, 6.2
2	*k*-means	Clustering	4.8
3	SVM	Statistical learning	6.4
4	Apriori	Association analysis	4.5, 6.3
5	EM	Statistical learning	6.8
6	PageRank	Link mining	9.6
7	Adaboost	Ensemble learning	8.4
8	kNN	Classification	4.7, 6.5
9	Naïve Bayes	Classification	4.2
10	CART	Classification	6.1

Note: *The information here was obtained during a 2006 poll conducted by the International Data Mining Conference.*

TOP MACHINE LEARNING ALGORITHMS YOU SHOULD KNOW

- Linear Regression
- Logistic Regression
- Linear Discriminant Analysis
- Classification and Regression Trees
- Naive Bayes
- K-Nearest Neighbors (KNN)
- Learning Vector Quantization (LVQ)
- Support Vector Machines (SVM)
- Random Forest
- Boosting
- AdaBoost

The [datatype] can be any of the four types supported by Weka:

- numeric.
- integer is treated as numeric.
- real is treated as numeric.
- [nominal-specification]
- string.
- date [date-format]
- relational for multi-instance data (for future use)

Basic Statistics

What is Kappa and How Does It Measure Inter-rater Reliability?

The Kappa Statistic or Cohen's* Kappa is a statistical measure of inter-rater reliability for categorical variables. In fact, it's almost synonymous with inter-rater reliability.

Kappa is used when two raters both apply a criterion based on a tool to assess whether or not some condition occurs. Examples include:

— Two doctors rate whether or not each of 20 patients has diabetes based on symptoms.
— Two swallowing experts assess whether each of 30 stroke patients has aspirated while drinking based on their cough reaction.
— Two education researchers both assess whether each of 50 children reads proficiently based on a reading evaluation.

If the two-raters can reliably use the criterion to make the **same assessment on the same targets,** then their agreement will be very high and provide evidence we have reliable ratings. If they can't, then either the criterion tool isn't useful or the raters are not well enough trained.

Why not just use percent agreement? The Kappa statistic corrects for chance agreement and percent agreement does not.

Here is a classic example: Two raters rating subjects on a diagnosis of Diabetes.

		Rater A Diabetes	
		Yes	**No**
Rater B Diabetes	**Yes**	35	20
	No	15	40

35 times they agree on yes. 40 times they agree on no.

20 disagreements come from Rater B choosing Yes and Rater A choosing no. 15 disagreements come from Rater A choosing Yes and Rater B choosing no.

This table would result in a Kappa of .51 (I won't go into the formulas).

But how do you know if you have a high level of agreement?

An often-heard Rule of Thumb for the Kappa statistic is:

"A Kappa Value of .70 Indicates good reliability"

Where did this originate? Cohen suggested the Kappa statistic be interpreted as:

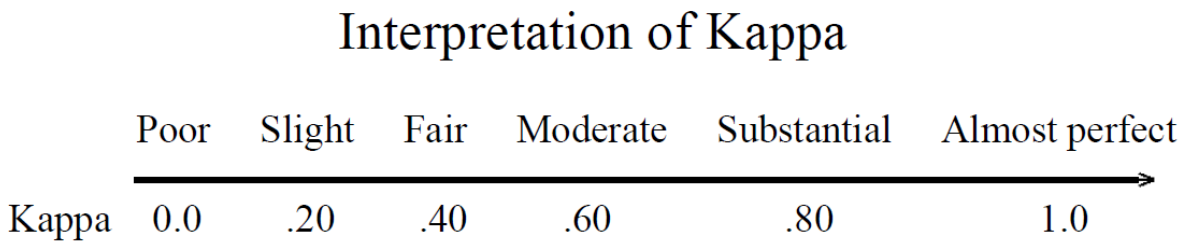
values ≤ 0	no agreement
0.01-0.20	as none to slight,
0.21-0.40	as fair,
0.41- 0.60	as moderate,
0.61-0.80	as substantial
0.81-1.00	as almost perfect agreement.

The emphasis is on SUGGESTED.
Several benchmarks have been proposed by other authors:

Landis and Koch	.81 to 1	Almost Perfect
Fleiss	>.75	Excellent
Altman	.81 to 1	Very Good

It's very similar to correlation coefficients. There isn't a clear cut off for what you consider strong, moderate, or weak.

Why can't we use these rules of thumb as clear cut offs? As with everyt



Kappa	Agreement
< 0	Less than chance agreement
0.01–0.20	Slight agreement
0.21– 0.40	Fair agreement
0.41–0.60	Moderate agreement
0.61–0.80	Substantial agreement
0.81–0.99	Almost perfect agreement

What is Absolute Error?

Absolute Error is the amount of error in your measurements. It is the difference between the measured value and "true" value. For example, if a scale states 90 pounds but you know your true weight is 89 pounds, then the scale has an absolute error of 90 lbs – 89 lbs = 1 lb

Formula

The formula for the absolute error (Δx) is:

$$(\Delta x) = x_i - x,$$

$$\mathbf{MAE} = \frac{1}{n}\sum_{i=1}^{n} |\boldsymbol{x}_i - \boldsymbol{x}|$$

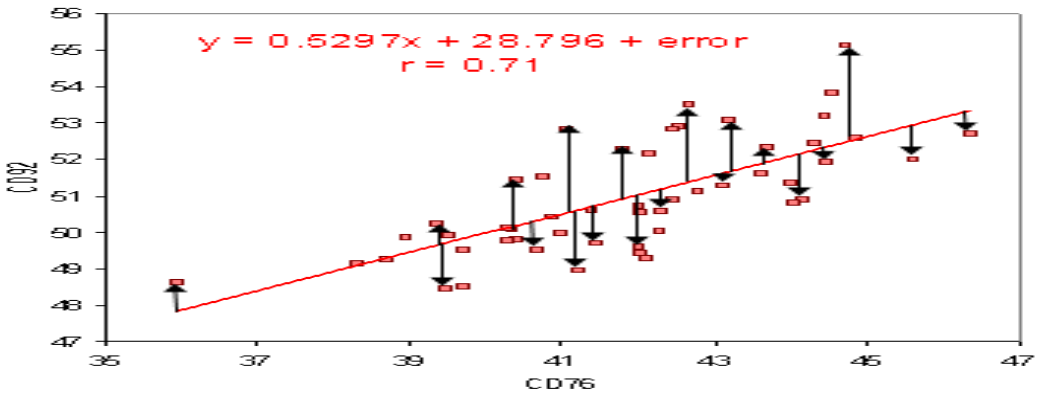
- n = the number of errors,
- Σ = summation symbol (which means "add them all up"),
- $|x_i - x|$ = the absolute errors.

The formula may look a little daunting, but the steps are easy:

1. Find all of your absolute errors, $x_i - x$.
2. Add them all up.
3. Divide by the number of errors. For example, if you had 10 measurements, divide by 10.
4.

Root Mean Square Error (RMSE

Root Mean Square Error (RMSE) is **the standard deviation of the residuals (prediction errors)**. Residuals are a measure of how far from the regression line data points are; RMSE is a measure of how spread out these residuals are. In other words, it tells you how concentrated the data is around the line of best fit.



If the RMSE for the test set is much higher than that of the training set, it is likely that you'**ve badly over fit the data**, i.e. you've created a model that tests well in sample, but has little predictive value when tested out of sample.

SUMMA OF

Algo Name	**Sup/Usup/reinfo**	**Dataset Name**	**Instances// TD/RESTDAT/SPLIT/CROSS**	**F.Mes**
accuracy	**Conf.Metrix**	**Accept.**		
OneR				
ZeroR				
NB				

Attribute-Relation File Format (ARFF)

An ARFF (Attribute-Relation File Format) file is an ASCII text file that describes a list of instances sharing a set of attributes. ARFF files were developed by the Machine Learning Project at the Department of Computer Science of The University of Waikato for use with the Weka machine learning software. This document descibes the version of ARFF used with Weka versions 3.2 to 3.3; this is an extension of the ARFF format as described in the data mining book written by Ian H. Witten and Eibe Frank (the new additions are string attributes, date attributes, and sparse instances).

This explanation was cobbled together by Gordon Paynter (gordon.paynter at ucr.edu) from the Weka 2.1 ARFF description, email from Len Trigg (lenbok at myrealbox.com) and Eibe Frank (eibe at cs.waikato.ac.nz), and some datasets. It has been edited by Richard Kirkby (rkirkby at cs.waikato.ac.nz). Contact Len if you're interested in seeing the ARFF 3 proposal.

Overview

ARFF files have two distinct sections. The first section is the **Header** information, which is followed the **Data** information.

The **Header** of the ARFF file contains the name of the relation, a list of the attributes (the columns in the data), and their types. An example header on the standard IRIS dataset looks like this:

```
% 1. Title: Iris Plants Database
%
% 2. Sources:
%      (a) Creator: R.A. Fisher
%      (b) Donor: Michael Marshall (MARSHALL%PLU@io.arc.nasa.gov)
%      (c) Date: July, 1988
%
@RELATION
iris

@ATTRIBUTE sepallength  NUMERIC
@ATTRIBUTE sepalwidth   NUMERIC
@ATTRIBUTE petallength  NUMERIC
@ATTRIBUTE petalwidth   NUMERIC
@ATTRIBUTE class        {Iris-setosa,Iris-versicolor,Iris-virginica}
```

The **Data** of the ARFF file looks like the following:

```
@DATA
5.1,3.5,1.4,0.2,Iris-setosa
4.9,3.0,1.4,0.2,Iris-setosa
4.7,3.2,1.3,0.2,Iris-setosa
4.6,3.1,1.5,0.2,Iris-setosa
5.0,3.6,1.4,0.2,Iris-setosa
5.4,3.9,1.7,0.4,Iris-setosa
4.6,3.4,1.4,0.3,Iris-setosa
5.0,3.4,1.5,0.2,Iris-setosa
4.4,2.9,1.4,0.2,Iris-setosa
4.9,3.1,1.5,0.1,Iris-setosa
```

Lines that begin with a % are comments. The **@RELATION**, **@ATTRIBUTE** and **@DATA** declarations are case insensitive.

Examples

Several well-known machine learning datasets are distributed with Weka in the $WEKAHOME/data directory as ARFF files.

The ARFF Header Section

The ARFF Header section of the file contains the relation declaration and attribute declarations.

The @relation Declaration

The relation name is defined as the first line in the ARFF file. The format is:

@relation <relation-name>

where <relation-name> is a string. The string must be quoted if the name includes spaces.

The @attribute Declarations

Attribute declarations take the form of an orderd sequence of **@attribute** statements. Each attribute in the data set has its own **@attribute** statement which uniquely defines the name of that attribute and it's data type. The order the attributes are declared indicates the column position in the data section of the file. For example, if an attribute is the third one declared then Weka expects that all that attributes values will be found in the third comma delimited column.

The format for the **@attribute** statement is:

@attribute <attribute-name> <datatype>

where the <*attribute-name*> must start with an alphabetic character. If spaces are to be included in the name then the entire name must be quoted.

The <*datatype*> can be any of the four types currently (version 3.2.1) supported by Weka:

- numeric
- <nominal-specification>
- string
- date [<date-format>]

where <nominal-specification> and <date-format> are defined below. The keywords **numeric**, **string** and **date** are case insensitive.

Numeric attributes

Numeric attributes can be real or integer numbers.

Nominal attributes

Nominal values are defined by providing an <nominal-specification> listing the possible values: {<nominal-name1>, <nominal-name2>, <nominal-name3>, ...}

For example, the class value of the Iris dataset can be defined as follows:

@ATTRIBUTE class {Iris-setosa,Iris-versicolor,Iris-virginica}

Values that contain spaces must be quoted.

String attributes

String attributes allow us to create attributes containing arbitrary textual values. This is very useful in text-mining applications, as we can create datasets with string attributes, then write Weka Filters to manipulate strings (like StringToWordVectorFilter). String attributes are declared as follows:

```
@ATTRIBUTE LCC    string
```

Date attributes

Date attribute declarations take the form:

```
@attribute <name> date [<date-format>]
```

where <name> is the name for the attribute and <date-format> is an optional string specifying how date values should be parsed and printed (this is the same format used by SimpleDateFormat). The default format string accepts the ISO-8601 combined date and time format: "yyyy-MM-dd'T'HH:mm:ss".

Dates must be specified in the data section as the corresponding string representations of the date/time (see example below).

ARFF Data Section

The ARFF Data section of the file contains the data declaration line and the actual instance lines.

The @data Declaration

The **@data** declaration is a single line denoting the start of the data segment in the file. The format is:

```
@data
```

The instance data

Each instance is represented on a single line, with carriage returns denoting the end of the instance.

Attribute values for each instance are delimited by commas. They must appear in the order that they were declared in the header section (i.e. the data corresponding to the nth **@attribute** declaration is always the nth field of the attribute).

Missing values are represented by a single question mark, as in:

```
@data
4.4,?,1.5,?,Iris-setosa
```

Values of string and nominal attributes are case sensitive, and any that contain space must be quoted, as follows:

```
@relation LCCvsLCSH

@attribute LCC string
@attribute LCSH string

@data
AG5,   'Encyclopedias and dictionaries.;Twentieth century.'
AS262, 'Science -- Soviet Union -- History.'
AE5,   'Encyclopedias and dictionaries.'
```

```
AS281, 'Astronomy, Assyro-Babylonian.;Moon -- Phases.'
AS281, 'Astronomy, Assyro-Babylonian.;Moon -- Tables.'
```

Dates must be specified in the data section using the string representation specified in the attribute declaration. For example:

```
@RELATION Timestamps

@ATTRIBUTE timestamp DATE "yyyy-MM-dd HH:mm:ss"

@DATA
"2001-04-03 12:12:12"
"2001-05-03 12:59:55"
```

Sparse ARFF files

Sparse ARFF files are very similar to ARFF files, but data with value 0 are not be explicitly represented.

Sparse ARFF files have the same header (i.e **@relation** and **@attribute** tags) but the data section is different. Instead of representing each value in order, like this:

```
@data
0, X, 0, Y, "class A"
0, 0, W, 0, "class B"
```

the non-zero attributes are explicitly identified by attribute number and their value stated, like this:

```
@data
{1 X, 3 Y, 4 "class A"}
{2 W, 4 "class B"}
```

Each instance is surrounded by curly braces, and the format for each entry is: <index> <space> <value> where index is the attribute index (starting from 0).

Note that the omitted values in a sparse instance are **0**, they are not "missing" values! If a value is unknown, you must explicitly represent it with a question mark (?).

Warning: There is a known problem saving SparseInstance objects from datasets that have string attributes. In Weka, string and nominal data values are stored as numbers; these numbers act as indexes into an array of possible attribute values (this is very efficient). However, the first string value is assigned index 0: this means that, internally, this value is stored as a 0. When a SparseInstance is written, string instances with internal value 0 are not output, so their string value is lost (and when the arff file is read again, the default value 0 is the index of a different string value, so the attribute value appears to change). To get around this problem, add a dummy string value at index 0 that is never used whenever you declare string attributes that are likely to be used in SparseInstance objects and saved as Sparse ARFF files.

BASIC TERMINOLOGY OF ML

TP Rate: rate of true positives (instances correctly classified as a given class)

FP Rate: rate of false positives (instances falsely classified as a given class)

Precision: proportion of instances that are truly of a class divided by the total instances classified as that class

Recall: proportion of instances classified as a given class divided by the actual total in that class (equivalent to TP rate)

F-Measure: A combined measure for precision and recall calculated as 2 * Precision * Recall / (Precision + Recall)

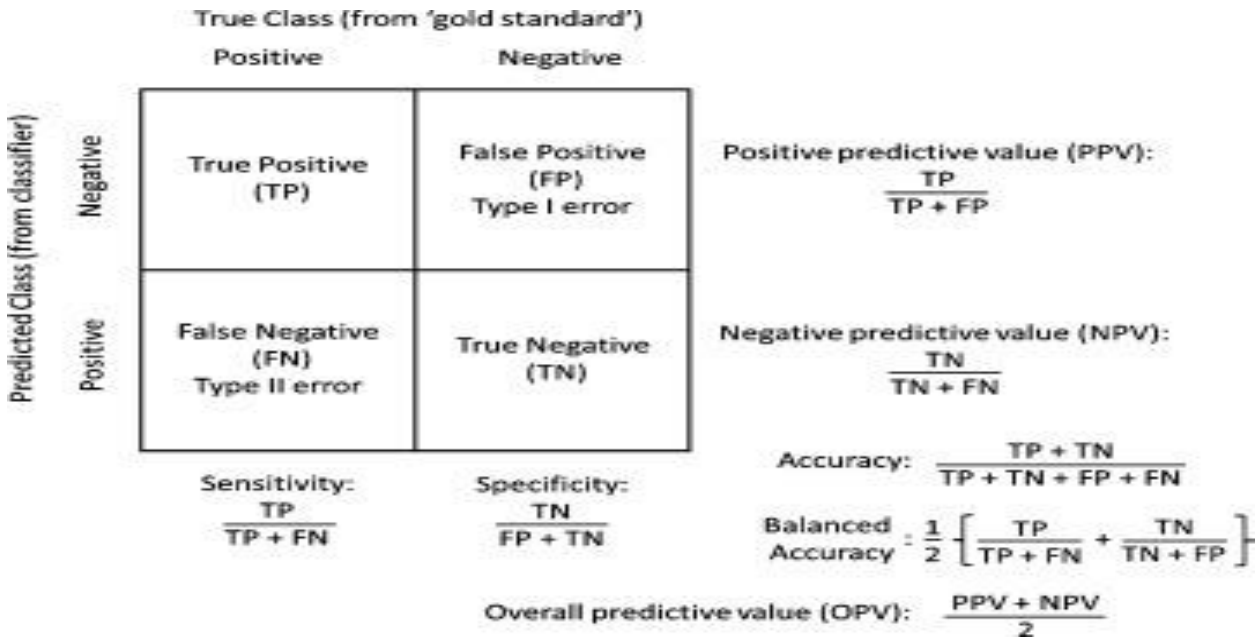
MCC is used in machine learning as a measure of the quality of binary (two-class) classifications. It takes into account true and false positives and negatives and is generally regarded as a balanced measure which can be used even if the classes are of very different sizes

ROC(Receiver Operating Characteristics) area measurement: One of the most important values output by Weka. They give you an idea of how the classifiers are performing in general

PRC(Precision Recall) area

A confusion matrix is **a technique for summarizing the performance of a classification algorithm**. Classification accuracy alone can be misleading if you have an unequal number of observations in each class or if you have more than two classes in your dataset.

In the field of machine learning and specifically the problem of statistical classification, a confusion matrix, also known as an **error matrix**, is a specific table layout that allows visualization of the performance of an algorithm, typically a supervised learning one.



Predictive Model: Evaluation

$$\text{Accuracy} = \frac{tp + tn}{tp + tn + fp + fn}$$

predictive result / classification		actual result / classification: yes	actual result / classification: no
	yes	**tp** (true positive)	**fp** (false positive) ← Type 1 error
	no	**fn** (false negative)	**tn** (true negative)

$$\text{Precision} = \frac{tp}{tp + fp} \qquad \text{Recall} = \frac{tp}{tp + fn}$$

$$F = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \qquad \text{True Negative Rate} = \frac{tn}{tn + fp}$$

Training/Validation and Test data

Lesson 2.3: Repeated training and testing

Evaluate J48 on segment-challenge

Sample mean $\bar{x} = \frac{\sum x_i}{n}$

Variance $\sigma^2 = \frac{\sum (x_i - \bar{x})^2}{n-1}$

Standard deviation σ

0.967
0.940
0.940
0.967
0.953
0.967
0.920
0.947
0.933
0.947

$\bar{x} = 0.949,\ \sigma = 0.018$

Supervised learning

Supervised learning, also known as supervised machine learning, is a subcategory of machine learning and artificial intelligence. It is defined by its use of **labeled datasets to train algorithms that to classify data or predict outcomes accurately**. As input data is fed into the model, it adjusts its weights until the model has been fitted appropriately, which occurs as part of the cross validation process. Supervised learning helps organizations solve for a variety of real-world problems at scale, such as classifying spam in a separate folder from your inbox.

Supervised learning can be separated into two types of problems when data mining—classification and regression:

- **Classification** uses an algorithm to accurately assign test data into specific categories. It recognizes specific entities within the dataset and attempts to draw some conclusions on how those entities should be labeled or defined. Common classification algorithms are linear classifiers, support vector machines (SVM), decision trees, k-nearest neighbor, and random forest, which are described in more detail below.
- **Regression** is used to **understand the relations**hip between dependent and independent variables. It is commonly used to make projections, such as for sales revenue for a given business. Linear regression, logistical regression, and polynomial regression are popular regression algorithms.

Supervised learning algorithms

Neural networks

Primarily leveraged for deep learning algorithms, neural networks process training data by mimicking the interconnectivity of the human brain through layers of nodes. Each node is made up of inputs, weights, a bias (or threshold), and an output. If that output value exceeds a given threshold, it “fires” or activates the node, passing data to the next layer in the network. Neural networks learn this mapping function through supervised learning, adjusting based on the loss function through the process of gradient descent. When the cost function is at or near zero, we can be confident in the model’s accuracy to yield the correct answer.

Naive Bayes

Naive Bayes is classification approach that adopts the **principle of class conditional** independence from the Bayes Theorem. This means that the presence of one feature does not impact the presence of another in the probability of a given outcome, and each predictor has an equal effect on that result. There are three types of Naïve Bayes classifiers: Multinomial Naïve Bayes, Bernoulli Naïve Bayes, and Gaussian Naïve Bayes. This technique is primarily used in text classification, spam identification, and recommendation systems.

Linear regression

Linear regression is used to identify the relationship between a dependent variable and one or more independent variables and is typically leveraged to make predictions about future outcomes. When there is only one independent variable and one dependent variable, it is known as simple linear regression. As the number of independent variables increases, it is referred to as multiple linear regression. For each type of linear regression, it seeks to plot a line of best fit, which is calculated through the method of least squares. However, unlike other regression models, this line is straight when plotted on a graph.

Logistic regression

While linear regression is leveraged when dependent variables are continuous, logistical regression is selected when the dependent variable is categorical, meaning they have binary outputs, such as "true" and "false" or "yes" and "no." While both regression models seek to understand relationships between data inputs, logistic regression is mainly used to solve binary classification problems, such as spam identification.

Support vector machine (SVM)

A support vector machine is a popular supervised learning model developed by Vladimir Vapnik, used for both data classification and regression. That said, it is typically leveraged for classification problems, constructing a hyperplane where the distance between two classes of data points is at its maximum. This hyperplane is known as the decision boundary, separating the classes of data points (e.g., oranges vs. apples) on either side of the plane.

K-nearest neighbor

K-nearest neighbor, also known as the KNN algorithm, is a non-parametric algorithm that classifies data points based on their proximity and association to other available data. This algorithm assumes that similar data points can be found near each other. As a result, it seeks to calculate the distance between data points, usually through Euclidean distance, and then it assigns a category based on the most frequent category or average.

Its ease of use and low calculation time make it a preferred algorithm by data scientists, but as the test dataset grows, the processing time lengthens, making it less appealing for classification tasks. KNN is typically used for recommendation engines and image recognition.

Random forest

Random forest is another flexible supervised machine learning algorithm used for both classification and regression purposes. The "forest" references a collection of uncorrelated decision trees, which are then merged together to reduce variance and create more accurate data predictions.

Unsupervised

Unsupervised machine learning and supervised machine learning are frequently discussed together. Unlike supervised learning, unsupervised learning uses unlabeled data. From that data, it discovers patterns that help solve for clustering or association problems. This is particularly useful when subject matter experts are unsure of common properties within a data set. Common clustering algorithms are hierarchical, k-means, and Gaussian mixture models.

Semi-supervised learning occurs when only part of the given input data has been labeled. Unsupervised and semi-supervised learning can be more appealing alternatives as it can be time-consuming and costly to rely on domain expertise to label data appropriately for supervised learning.

For a deep dive into the differences between these approaches, check out "Supervised vs. Unsupervised Learning: What's the Difference?"

Unsupervised learning examples

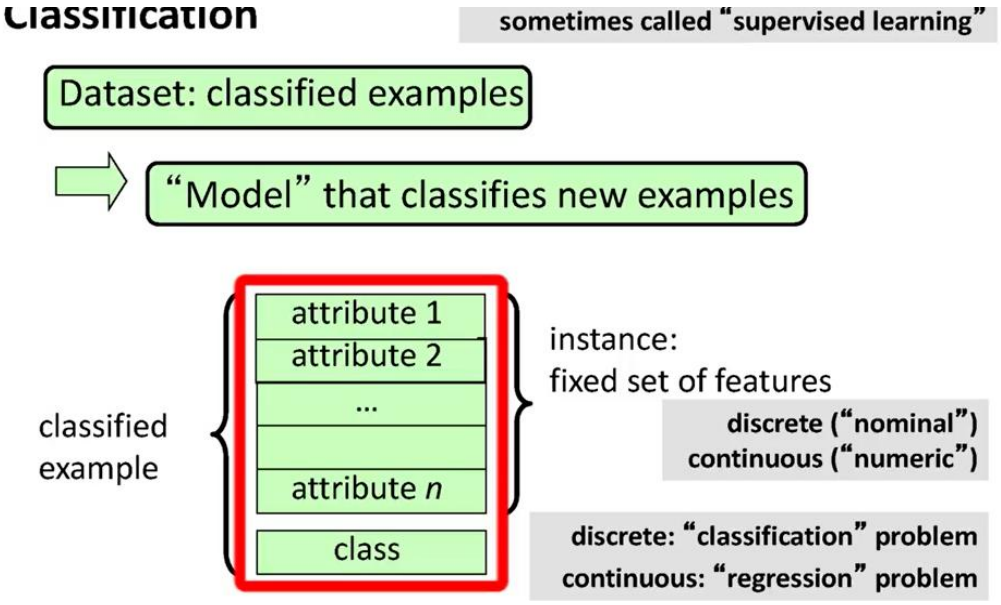
Supervised learning models can be used to build and advance a number of business applications, including the following:

- **Image- and object-recognition:** Supervised learning algorithms can be used to locate, isolate, and categorize objects out of videos or images, making them useful when applied to various computer vision techniques and imagery analysis.
- **Predictive analytics:** A widespread use case for supervised learning models is in creating predictive analytics systems to provide deep insights into various business data points. This allows enterprises to anticipate certain results based on a given output variable, helping business leaders justify decisions or pivot for the benefit of the organization.
- **Customer sentiment analysis:** Using supervised machine learning algorithms, organizations can extract and classify important pieces of information from large volumes of data—including context, emotion, and intent—with very little human intervention. This can be incredibly useful when gaining a better understanding of customer interactions and can be used to improve brand engagement efforts.
- **Spam detection:** Spam detection is another example of a supervised learning model. Using supervised classification algorithms, organizations can train databases to recognize patterns or anomalies in new data to organize spam and non-spam-related correspondences effectively.

Challenges Of Supervised Learning:

Although supervised learning can offer businesses advantages, such as deep data insights and improved automation, there are some challenges when building sustainable supervised learning models. The following are some of these challenges:

- Supervised learning models can require certain levels of expertise to structure accurately.
- Training supervised learning models can be very time intensive.
- Datasets can have a higher likelihood of human error, resulting in algorithms learning incorrectly.
- Unlike unsupervised learning models, supervised learning cannot cluster or classify data on its own.



BASE LINE ACCURACY

Classification	Clustering
Uses labelled data as the input	Uses unlabelled data as the input
The output is known	The output is unknown
Uses supervised machine learning	Uses unsupervised machine learning
A training data set is provided and used to produce classifications	A training data set is not provided and used to produce clusters
Examples of algorithms: Decision-trees, Bayesian Classifiers and Support Vector Machines (SVM)	Examples of algorithms: Partition-based clustering (k-means), Hierarchical clustering (agglomerative & divisive) and DBSCAN
Can be more compex than clustering	Can be less compex than classification
Does not specify areas for improvement	Specifies areas for improvement
Two-phase	Single-phase
Boundary conditions must be specified	Boundary conditions do not always need to be specified

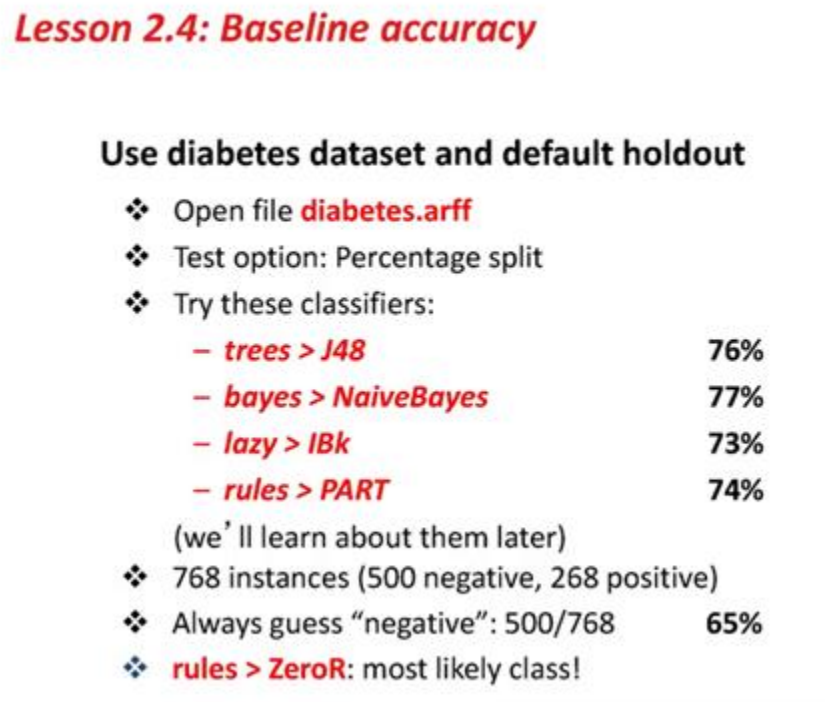
Association Rule

Association rule mining finds interesting associations and relationships among large sets of data items. This rule shows how frequently a item set occurs in a transaction. A typical example is Market Based Analysis.

Market Based Analysis is one of the key techniques used by large relations to show associations between items. It allows retailers to identify relationships between the items that people buy together frequently.

Given a set of transactions, we can find rules that will predict the occurrence of an item based on the occurrences of other items in the transaction.

Select Attributes



Lesson 2.4: Baseline accuracy

Sometimes baseline is best!

- Open **supermarket.arff** and blindly apply

rules > ZeroR	64%
trees > J48	63%
bayes > NaiveBayes	63%
lazy > IBk	38% (!!)
rules > PART	63%

- Attributes are not informative
- Don't just apply Weka to a dataset:
  you need to understand what's going on!

Lesson 2.4: Baseline accuracy

- Consider whether differences are likely to be significant
- Always try a simple baseline,
  e.g. rules > ZeroR

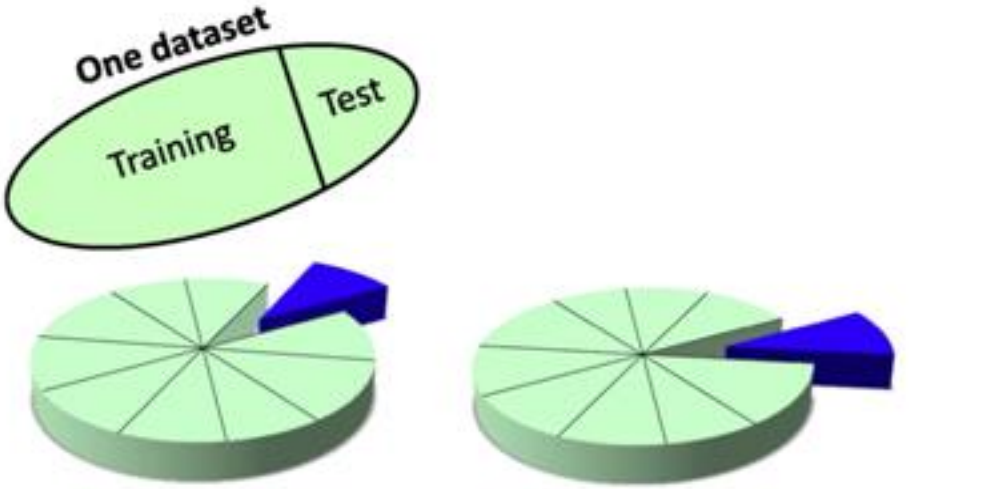
- Look at the dataset
- Don't blindly apply Weka:
  try to understand what's going on!

Lesson 2.5: Cross-validation

- Can we improve upon repeated holdout?
  (i.e. reduce variance)
- Cross-validation
- Stratified cross-validation

Lesson 2.5: Cross-validation

- Repeated holdout
  (in Lesson 2.3, hold out 10% for testing, repeat 10 times)

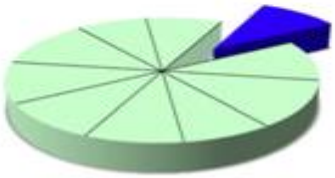


(repeat 10 times)

Lesson 2.5: Cross-validation

10-fold cross-validation

- Divide dataset into 10 parts (folds)
- Hold out each part in turn
- Average the results
- Each data point used once for testing, 9 times for training

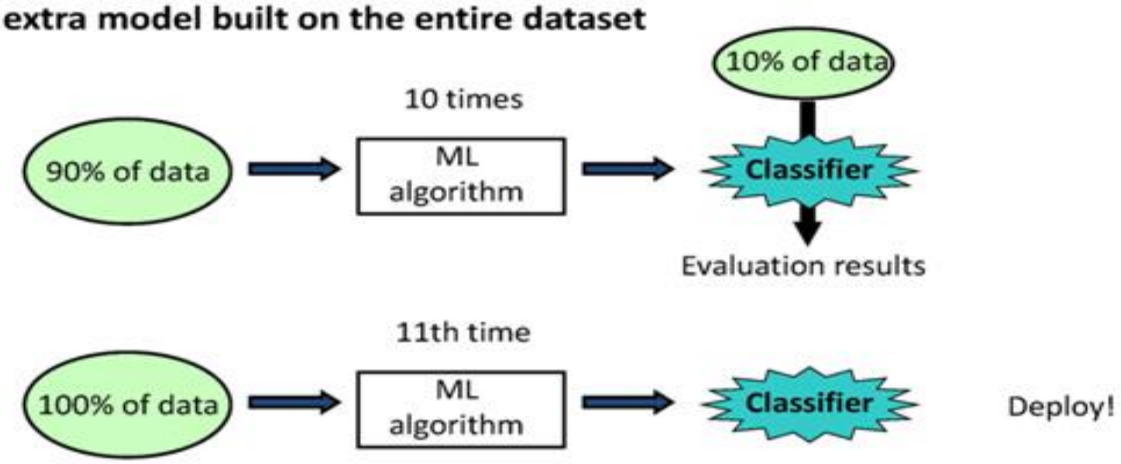


Stratified cross-validation

- Ensure that each fold has the right proportion of each class value

Lesson 2.5: Cross-validation

After cross-validation, Weka outputs an extra model built on the entire dataset



Lesson 2.5: Cross-validation

- Cross-validation better than repeated holdout
- Stratified is even better
- With 10-fold cross-validation, Weka invokes the learning algorithm 11 times
- **Practical rule of thumb:**
- Lots of data? – use percentage split
- Else stratified 10-fold cross-validation

Course text

- Section 5.3 *Cross-validation*

Lesson 2.6: Cross-validation results

Is cross-validation really better than repeated holdout?

- **Diabetes** dataset
- Baseline accuracy (**rules > ZeroR**): 65.1%
- **trees > J48**
- 10-fold cross-validation 73.8%
- ... with different random number seed

1	2	3	4	5	6	7	8	9	10
73.8	75.0	75.5	75.5	74.4	75.6	73.6	74.0	74.5	73.0

Lesson 2.6: Cross-validation results

Is cross-validation really better than repeated holdout?

- **Diabetes** dataset
- Baseline accuracy (**rules > ZeroR**): 65.1%
- **trees > J48**
- 10-fold cross-validation 73.8%
- ... with different random number seed

1	2	3	4	5	6	7	8	9	10
73.8	75.0	75.5	75.5	74.4	75.6	73.6	74.0	74.5	73.0

Lesson 2.6: Cross-validation results

Is cross-validation really better than repeated holdout?

- **Diabetes** dataset
- Baseline accuracy (**rules > ZeroR**): 65.1%
- **trees > J48**
- 10-fold cross-validation 73.8%
- ... with different random number seed

1	2	3	4	5	6	7	8	9	10
73.8	75.0	75.5	75.5	74.4	75.6	73.6	74.0	74.5	73.0

Lesson 2.6: Cross-validation results

- Why 10-fold? E.g. 20-fold: 75.1%

- Cross-validation really is better than repeated holdout
- It reduces the variance of the estimate

ZeroR

ZeroR is the simplest classification method which relies on the target and ignores all predictors. ZeroR classifier predicts the **majority category** (class). Although there is no predictability power in ZeroR, it is useful for determi baseline performance as a benchmark for other classification methods.

Algorithm

Construct a frequency table for the target and select its most frequent value.

Example:

"Play Golf = Yes" is the ZeroR model for the following dataset with an accuracy of 0.64.

Predictors | Target

Outlook	Temp	Humidity	Windy	Play Golf
Rainy	Hot	High	False	No
Rainy	Hot	High	True	No
Overcast	Hot	High	False	Yes
Sunny	Mild	High	False	Yes
Sunny	Cool	Normal	False	Yes
Sunny	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Rainy	Mild	High	False	No
Rainy	Cool	Normal	False	Yes
Sunny	Mild	Normal	False	Yes
Rainy	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Sunny	Mild	High	True	No

Play Golf	
Yes	No
9	5

Predictors Contribution

There is nothing to be said about the predictors contribution to the model because ZeroR does not use any of them.

Model Evaluation

The following confusion matrix shows that ZeroR only predicts the majority class correctly. As mentioned before, ZeroR is only useful for determining a baseline performance for other classification methods.

Confusion Matrix		Play Golf			
		Yes	No		
ZeroR	Yes	9	5	*Positive Predictive Value*	0.64
	No	0	0	*Negative Predictive Value*	0.00
		Sensitivity	*Specificity*	Accuracy = 0.64	
		1.00	0.00		

OneR

OneR, short for "One Rule", is a simple, yet accurate, classification algorithm that generates one rule for each predictor in the data, then selects the rule with the smallest total error as its "one rule". To create a rule for a predictor, we construct a frequency table for each predictor against the target. It has been shown that OneR produces rules only slightly less accurate than state-of-the-art classification algorithms while producing rules that are simple for humans to interpret.

OneR Algorithm

For each predictor,
- For each value of that predictor, make a rule as follows;
  - Count how often each value of target (class) appears
  - Find the most frequent class
  - Make the rule assign that class to this value of the predictor
- Calculate the total error of the rules of each predictor

Choose the predictor with the smallest total error.

Example:

Finding the best predictor with the smallest total error using OneR algorithm based on related frequency tables.

Which one is the best predictor ?

Outlook	Temp	Humidity	Windy	Play Golf
Rainy	Hot	High	False	No
Rainy	Hot	High	True	No
Overcast	Hot	High	False	Yes
Sunny	Mild	High	False	Yes
Sunny	Cool	Normal	False	Yes
Sunny	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Rainy	Mild	High	False	No
Rainy	Cool	Normal	False	Yes
Sunny	Mild	Normal	False	Yes
Rainy	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Sunny	Mild	High	True	No

Frequency Tables

★		Play Golf	
		Yes	No
Outlook	Sunny	3	2
	Overcast	4	0
	Rainy	2	3

		Play Golf	
		Yes	No
Temp.	Hot	2	2
	Mild	4	2
	Cool	3	1

		Play Golf	
		Yes	No
Humidity	High	3	4
	Normal	6	1

		Play Golf	
		Yes	No
Windy	False	6	2
	True	3	3

The best predictor is:

★		Play Golf	
		Yes	No
Outlook	Sunny	3	2
	Overcast	4	0
	Rainy	2	3

IF Outlook = Sunny THEN PlayGolf = Yes
IF Outlook = Overcast THEN PlayGolf = Yes
IF Outlook = Rainy THEN PlayGolf = No

Predictors Contribution

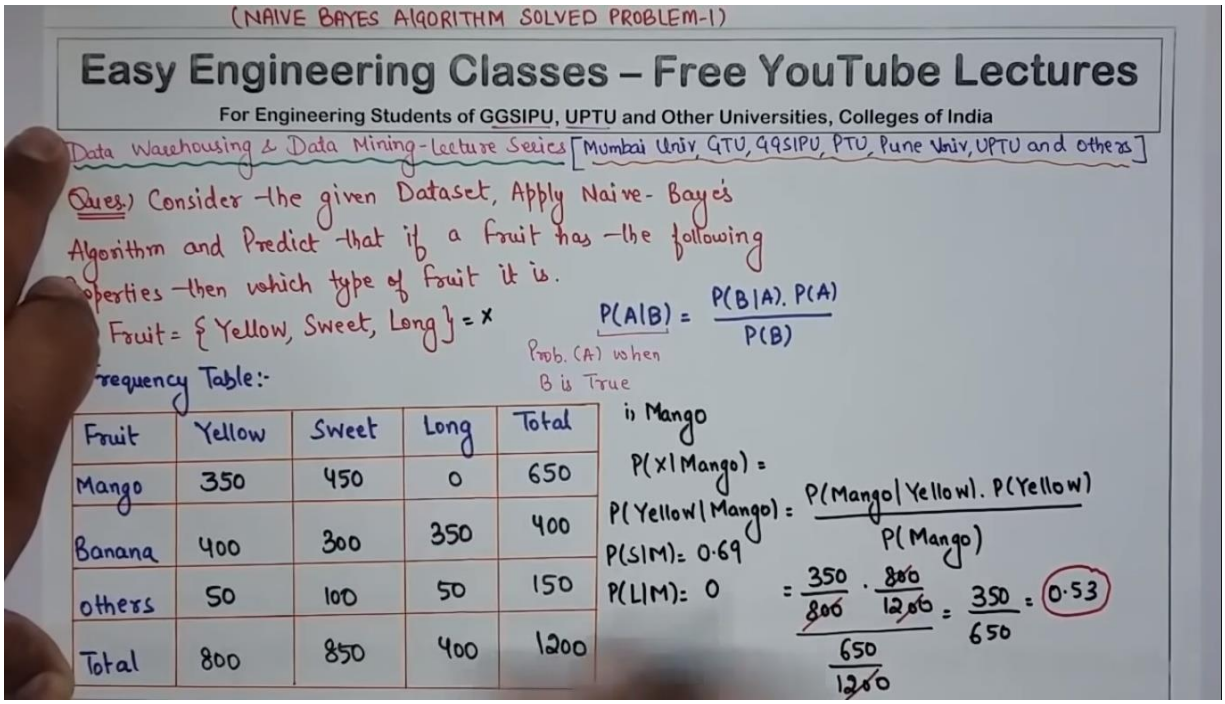
Simply, the total error calculated from the frequency tables is the measure of each predictor contribution. A low total error means a higher contribution to the predictability of the model.

Model Evaluation

The following confusion matrix shows significant predictability power. OneR does not generate score or probability, which means evaluation charts (Gain, Lift, K-S and ROC) are not applicable.

Confusion Matrix		Play Golf			
		Yes	No		
OneR	Yes	7	2	*Positive Predictive Value*	0.78
	No	2	3	*Negative Predictive Value*	0.60
		Sensitivity	*Specificity*	Accuracy = 0.71	
		0.78	0.60		

NAÏVE BAYES ALGORITHMS WITH AN EXAMPLE



(NAIVE BAYES ALGORITHM SOLVED PROBLEM-1)

Easy Engineering Classes – Free YouTube Lectures

For Engineering Students of GGSIPU, UPTU and Other Universities, Colleges of India

Data Warehousing & Data Mining-Lecture Series [Mumbai Univ, GTU, GGSIPU, PTU, Pune Univ, UPTU and others]

Ques.) Consider the given Dataset, Apply Naive-Bayes Algorithm and Predict that if a fruit has the following properties then which type of fruit it is.

Fruit = { Yellow, Sweet, Long } = X

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

Prob. (A) when B is True

Frequency Table:-

Fruit	Yellow	Sweet	Long	Total
Mango	350	450	0	650
Banana	400	300	350	400
others	50	100	50	150
Total	800	850	400	1200

i) Mango X

$P(X|Mango) = P(Y|M).P(S|M).P(L|M) = 0$

$P(Yellow|Mango) = \frac{P(Mango|Yellow).P(Yellow)}{P(Mango)}$

$= \frac{\frac{350}{800} \cdot \frac{800}{1200}}{\frac{650}{1200}} = \frac{350}{650} = 0.53$

$P(S|M) = 0.69$

$P(L|M) = 0$

ii) Banana

$P(Y|B) = 1$

$P(S|B) = 0.75$

$P(L|B) = 0.875$

$P(X|B) \rightarrow 0.65$

iii) Others

$P(Y|O) = 0.33$

$P(S|O) = 0.66$

$P(L|O) = 0.33$

$P(X|O) \rightarrow 0.072$

(NAIVE BAYES ALGORITHM SOLVED PROBLEM-1)

Easy Engineering Classes – Free YouTube Lectures

For Engineering Students of GGSIPU, UPTU and Other Universities, Colleges of India

Data Warehousing & Data Mining-Lecture Series [Mumbai Univ, GTU, GGSIPU, PTU, Pune Univ, UPTU and others]

Ques.) Consider the given Dataset, Apply Naive-Bayes Algorithm and Predict that if a fruit has the following properties then which type of fruit it is.

Fruit = { Yellow, Sweet, Long } = X

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

Prob. (A) when B is True

Frequency Table:-

Fruit	Yellow	Sweet	Long	Total
Mango	350	450	0	650
Banana	400	300	350	400
others	50	100	50	150
Total	800	850	400	1200

i) Mango X

$P(X|Mango) = P(Y|M).P(S|M).P(L|M) = 0$

$P(Yellow|Mango) = \frac{P(Mango|Yellow).P(Yellow)}{P(Mango)}$

$= \frac{\frac{350}{800} \cdot \frac{800}{1200}}{\frac{650}{1200}} = \frac{350}{650} = 0.53$

$P(S|M) = 0.69$

$P(L|M) = 0$

ii) Banana

$P(Y|B) = 1$

$P(S|B) = 0.75$

$P(L|B) = 0.875$

$P(X|B) \rightarrow 0.65$

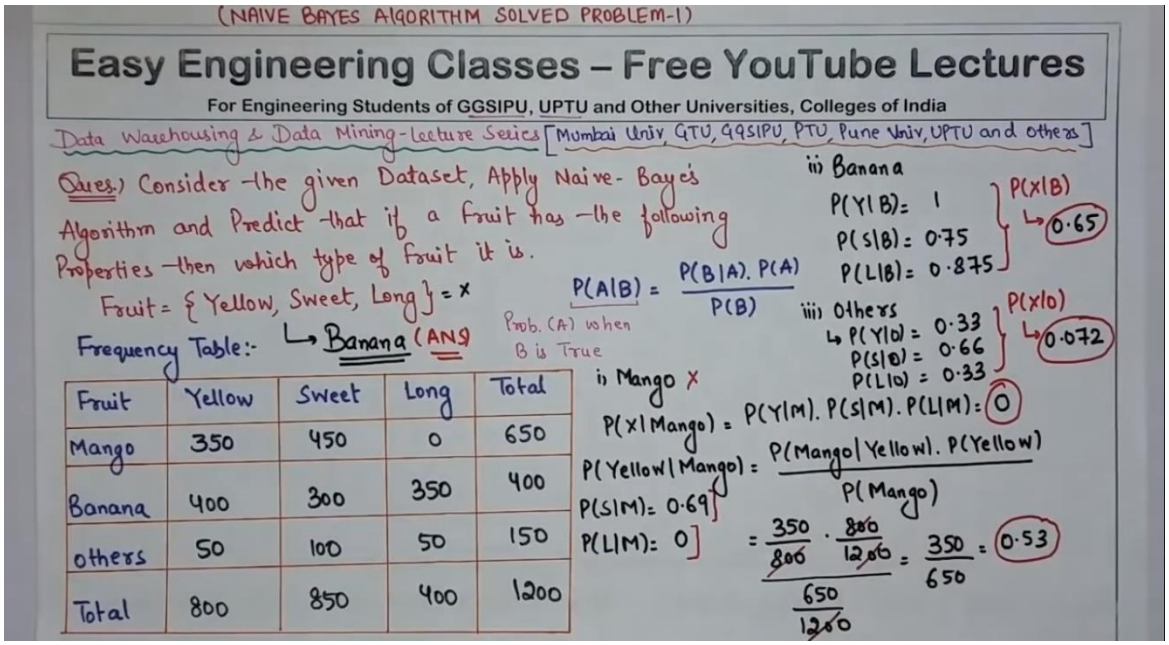
iii) Others

$P(Y|O) = 0.33$

$P(S|O) = 0.66$

$P(L|O) = 0.33$

$P(X|O) \rightarrow 0.072$



J48 Algorithms

Algorithm of J48 (*D*)
Input: a dataset *D*
begin
  Tree = {}
  If (D is "pure") ||(other stopping criteria met)then terminate;
  For all attribute a $a \in D$ D do
    Compute criteria of impurity function if we split on *a*;
  a_{best}= Best attribute according to above computed criteria
  Tree = Create a decision node that tests a_{best} in the root
  D_v= Induced sub-datasets from *D* based on a_{best}
  For all D_v do
    begin
      Tree $_v$= J48(D_v)
      Attach Tree $_v$ to the corresponding branch of Tree
    end
  return Tree
end

What is Hypothesis?

Hypothesis is **an assumption that** is made on the basis of some evidence. This is the initial point of any investigation that translates the research questions into a prediction. It includes components like variables, population and the relation between the variables. A research hypothesis is a hypothesis that is used to test the relationship between two or more variables.

Characteristics of Hypothesis

Following are the characteristics of hypothesis:

- The hypothesis should be clear and precise to consider it to be reliable.
- If the hypothesis is a relational hypothesis, then it should be stating the relationship between variables.
- The hypothesis must be specific and should have scope for conducting more tests.
- The way of explanation of the hypothesis must be very simple and it should also be understood that the simplicity of the hypothesis is not related to its significance.

Sources of Hypothesis

Following are the sources of hypothesis:

- The resemblance between the phenomenons.
- Observations from past studies, present-day experiences and from the competitors.
- Scientific theories.
- General patterns that influence the thinking process of people.

Types of Hypothesis

There are six forms of hypothesis and they are:

- Simple hypothesis
- Complex hypothesis
- Directional hypothesis
- Non-directional hypothesis
- Null hypothesis
- Associative and casual hypothesis

Simple Hypothesis

It shows a relationship between one dependent variable and a single independent variable. For example – If you eat more vegetables, you will lose weight faster. Here, eating more vegetables is an independent variable, while losing weight is the dependent variable.

Complex Hypothesis

It shows the relationship between two or more dependent variables and two or more independent variables. Eating more vegetables and fruits leads to weight loss, glowing skin, reduces the risk of many diseases such as heart disease, high blood pressure and some cancers.

Directional Hypothesis

It shows how a researcher is intellectual and committed to a particular outcome. The relationship between the variables can also predict its nature. For example- children aged four years eating proper food over a five-year period are having higher IQ levels than children not having a proper meal. This shows the effect and direction of effect.

Non-directional Hypothesis

It is used when there is no theory involved. It is a statement that a relationship exists between two variables, without predicting the exact nature (direction) of the relationship.

Null Hypothesis

It provides the statement which is contrary to the hypothesis. It's a negative statement, and there is no relationship between independent and dependent variables. The symbol is denoted by "HO".

Associative and Causal Hypothesis

Associative hypothesis occurs when there is a change in one variable resulting in a change in the other variable. Whereas, causal hypothesis proposes a cause and effect interaction between two or more variables.

Examples of Hypothesis

Following are the examples of hypothesis based on their types:

- Consumption of sugary drinks every day leads to obesity is an example of a simple hypothesis.
- All lilies have the same number of petals is an example of a null hypothesis.
- If a person gets 7 hours of sleep, then he will feel less fatigue than if he sleeps less.

Functions of Hypothesis

Following are the functions performed by the hypothesis:

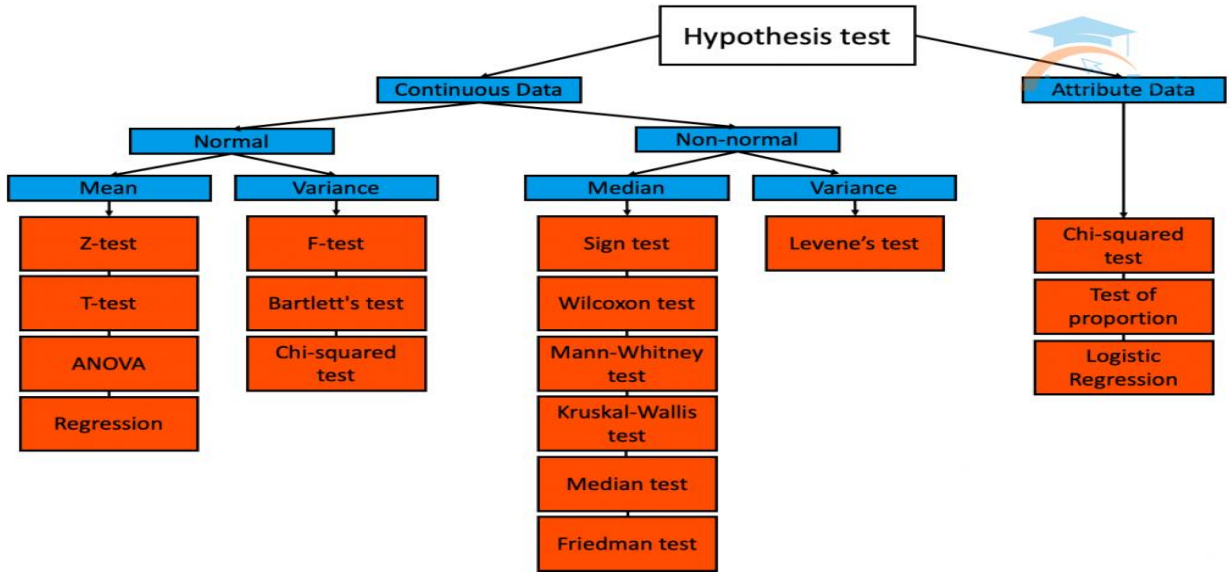
- Hypothesis helps in making an observation and experiments possible.
- It becomes the start point for the investigation.
- Hypothesis helps in verifying the observations.

- It helps in directing the inquiries in the right directions.
-

How will Hypothesis help in Scientific Method?

Researchers use hypothesis to put down their thoughts directing how the experiment would take place. Following are the steps that are involved in the scientific method:

- Formation of question
- Doing background research
- Creation of hypothesis
- Designing an experiment
- Collection of data
- Result analysis
- Summarizing the experiment
- Communicating the results



Association Rule

Association rule mining finds interesting associations and relationships among large sets of data items. This rule shows how frequently a item set occurs in a transaction. A typical example is Market Based Analysis.

Market Based Analysis is one of the key techniques used by large relations to show associations between items. It allows retailers to identify relationships between the items that people buy together frequently.

Given a set of transactions, we can find rules that will predict the occurrence of an item based on the occurrences of other items in the transaction.

TID	Items
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Before we start defining the rule, let us first see the basic definitions.

Support Count( ) – Frequency of occurrence of a itemset.
Here ({Milk, Bread, Diaper})=2

Frequent Itemset – An itemset whose support is greater than or equal to minsup threshold.

Association Rule – An implication expression of the form X -> Y, where X and Y are any 2 itemsets.
Example: {Milk, Diaper}->{Beer}

Rule Evaluation Metrics –

- **Support(s) –**
  The number of transactions that include items in the {X} and {Y} parts of the rule as a percentage of the total number of transaction.It is a measure of how frequently the collection of items occur together as a percentage of all transactions.
  Support = (X+Y) total –
  It is interpreted as fraction of transactions that contain both X and Y.
- **Confidence(c) –**

  It is the ratio of the no of transactions that includes all items in {B} as well as the no of

transactions that includes all items in {A} to the no of transactions that includes all items in {A}.

- **Conf(X=>Y) = Supp(X Y) Supp(X) –** It measures how often each item in Y appears in transactions that contains items in X also.

- **Lift(l) –** The lift of the rule X=>Y is the confidence of the rule divided by the expected confidence, assuming that the itemsets X and Y are independent of each other.The expected confidence is the confidence divided by the frequency of {Y}.
- **Lift(X=>Y) = Conf(X=>Y) Supp(Y) –**

  Lift value near 1 indicates X and Y almost often appear together as expected, greater than 1 means they appear together more than expected and less than 1 means they appear less than expected.Greater lift values indicate stronger association.

Example – From the above table, {Milk, Diaper}=>{Beer}

```
s= σ({Milk, Diaper, Beer}) ÷ |T|
= 2/5
= 0.4

c= σ(Milk, Diaper, Beer) ÷ σ(Milk, Diaper)
= 2/3
= 0.67

l= Supp({Milk, Diaper, Beer}) ÷ Supp({Milk, Diaper})*Supp({Beer})
= 0.4/(0.6*0.6)
= 1.11
```

The Association rule is very useful in analyzing datasets. The data is collected using bar-code scanners in supermarkets. Such databases consists of a large number of transaction records which list all items bought by a customer on a single purchase. So the manager could know if certain groups of items are consistently purchased together and use this data for adjusting store layouts, cross-selling, promotions based on statistics.

Lesson 3.3: Association rules

- **Support:** number of instances that satisfy a rule
- **Confidence:** proportion of instances that satisfy the left-hand side for which the right-hand side also holds
- Specify minimum confidence, seek the rules with greatest support??

			support	confidence
1. outlook = overcast	==>	play = yes	4	100%
2. temperature = cool	==>	humidity = normal	4	100%
3. humidity = normal & windy = false	==>	play = yes	4	100%
4. outlook = sunny & play = no	==>	humidity = high	3	100%
5. outlook = sunny & humidity = high	==>	play = no	3	100%
6. outlook = rainy & play = yes	==>	windy = false	3	100%
7. outlook = rainy & windy = false	==>	play = yes	3	100%
8. temperature = cool & play = yes	==>	humidity = normal	3	100%
9. outlook = sunny & temperature = hot	==>	humidity = high	2	100%
10. temperature = hot & play = no	==>	outlook = sunny	2	100%

EXAMPLE OF ASSOCIATION RULES

Assume there are 5 customers

3 of them bought milk, 2 bought potato chip and 2 bought both of them

Transaction 1: Frozen pizza, cola, milk
Transaction 2: Milk, potato chips
Transaction 3: Cola, frozen pizza
Transaction 4: Milk, potato chips
Transaction 5: Cola, pretzels

milk → potato chip

How about Potato chip → Milk ?

support milk = P(milk) = 3/5 = 0.6

support potato chip = P(potato chip) = 2/5 = 0.4

support = P(milk & potato chip) = 2/5 = 0.4

confidence

= support (milk & potato chip)/support(milk)

= 0.4/0.6

= 0.67

CONFIDENCE= P(Milk & potato chip) /P(Milk)

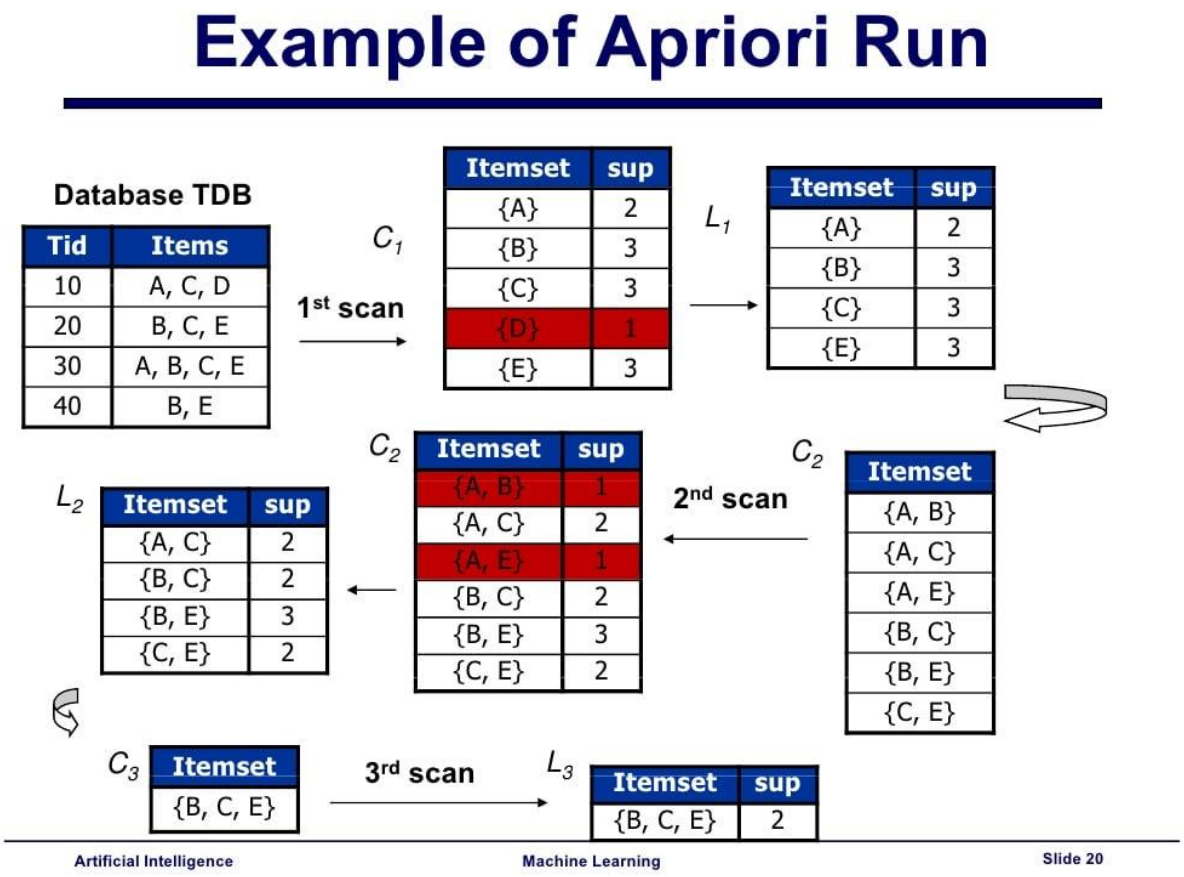
lift = confidence/support(potato chip) = 0.67/0.40 = 1.67

LIFT = [P(Milk & Potato chip)/P(milk)] /P(Potato chip)

Any rule with a lift < 1 does not indicate a cross-selling opportunity

Association Rules	Support Count of Antecedent	Support Count of Rule	Support of Rule	Confidence of Rule
Diaper → Beer	4	3	$3/5 = 0.6$	$3/4 = 0.75$
{Milk, Diaper} → Beer	3	2	$2/5 = 0.4$	$2/3 = 0.67$
Bread → Milk	4	3	$3/5 = 0.6$	$3/4 = 0.75$
{Bread, Milk} → Diaper	3	2	$3/5 = 0.6$	$2/3 = 0.67$
{Bread, Milk} → Coke	3	1	$3/5 = 0.6$	$1/3 = 0.33$

Apriori Association Algorithms



Association Rule Mining - Apriori Algorithm

Association Rule is one of the very important concepts of machine learning being used in market basket analysis.

Market Basket Analysis is the study of customer transaction databases to determine dependencies between the various items they purchase at different times .

Association rule learning is a rule-based machine learning method for discovering interesting relations between variables in large databases. It identifies frequent if-then associations called association rules which consists of an antecedent (if) and a consequent (then).

For example: *"If tea and milk, then sugar" ("If tea and milk are purchased, then sugar would also be bought by the customer")*

– **Antecedent**: Tea and Milk

– **Consequent**: Sugar.

There are three common metrics to measure association:

Support is an indication of how frequently the items appear in the data. Mathematically, support is the fraction of the total number of transactions in which the item set occurs.

$$Support(\{X\} \rightarrow \{Y\}) = \frac{Transactions\ containing\ both\ X\ and\ Y}{Total\ number\ of\ transactions}$$

Confidence indicates the number of times the if-then statements are found true. Confidence is the conditional probability of occurrence of consequent given the antecedent.

$$Confidence(\{X\} \rightarrow \{Y\}) = \frac{Transactions\ containing\ both\ X\ and\ Y}{Transactions\ containing\ X}$$

Lift can be used to compare confidence with expected confidence. This says how likely item Y is purchased when item X is purchased, while controlling for how popular item Y is. Mathematically,

$$Lift(\{X\} \rightarrow \{Y\}) = \frac{(Transactions\ containing\ both\ X\ and\ Y)/(Transactions\ containing\ X)}{Fraction\ of\ transactions\ containing\ Y}$$

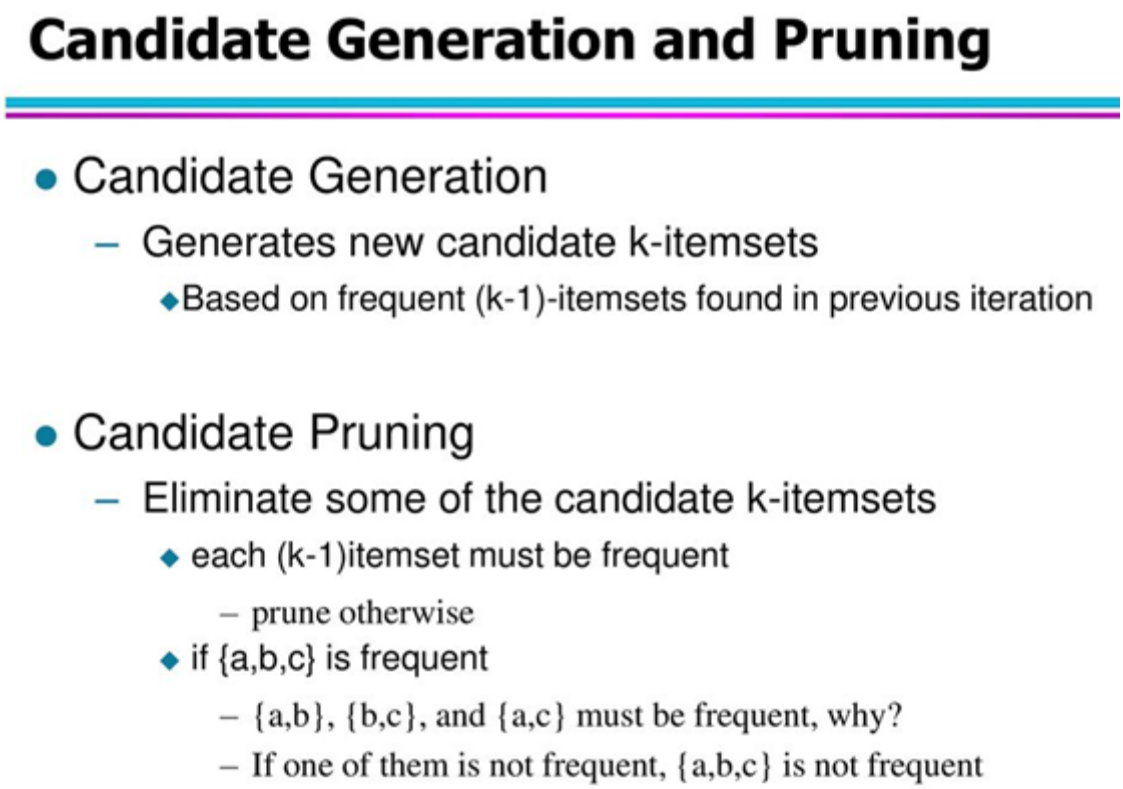
Candidate Generation Overview

Candidate generation is the first stage of recommendation. Given a query, the system generates a set of relevant candidates. The following table shows two common candidate generation approaches:

Type	Definition	Example
content-based filtering	Uses *similarity between items* to recommend items similar to what the user likes.	If user A watches two cute cat videos, then the system can recommend cute animal videos to that user.
collaborative filtering	Uses *similarities between queries and items simultaneously* to provide recommendations.	If user A is similar to user B, and user B likes video 1, then the system can recommend video 1 to user A (even if user A hasn't seen any videos similar to video 1).

Candidate Pruning

Candidate Pruning: **goes hand in hand with Candidate Generation** and is responsible for eliminating some of the candidate k-itemsets that are infrequent. Support Counting: this step is responsible for determining the frequency of occurrence for every candidate itemset that remains after the candidate pruning stage



FP-Growth (frequent-pattern growth) algorithm

Association rules mining is an important technology in data mining. FP-Growth (frequent-pattern growth) algorithm is a classical algorithm in association rules mining. But the FP-Growth algorithm in mining **needs two times to scan database**, which reduces the efficiency of algorithm.

Shortcomings Of Apriori Algorithm

1. Using Apriori needs a generation of candidate itemsets. These itemsets may be large in number if the itemset in the database is huge.
2. Apriori needs multiple scans of the database to check the support of each itemset generated and this leads to high costs.

These shortcomings can be overcome using the FP growth algorithm.

Frequent Pattern Growth Algorithm

This algorithm is an improvement to the Apriori method. A frequent pattern is generated without the need for candidate generation. FP growth algorithm represents the database in the form of a tree called a frequent pattern tree or FP tree.

This tree structure will maintain the association between the itemsets. The database is fragmented using one frequent item. This fragmented part is called “pattern fragment”. The itemsets of these fragmented patterns are analyzed. Thus with this method, the search for frequent itemsets is reduced comparatively.

FP Tree

Frequent Pattern Tree is a tree-like structure that is made with the initial itemsets of the database. The purpose of the FP tree is to mine the most frequent pattern. Each node of the FP tree represents an item of the itemset.

The root node represents **null while** the lower nodes represent the itemsets. The association of the nodes with the lower nodes that is the itemsets with the other itemsets are maintained while forming the tree.

Frequent Pattern Algorithm Steps

The frequent pattern growth method lets us find the frequent pattern without candidate generation.

Let us see the steps followed to mine the frequent pattern using frequent pattern growth algorithm:

#1) The first step is to scan the database to find the occurrences of the itemsets in the database. This step is the same as the first step of Apriori. The count of 1-itemsets in the database is called support count or frequency of 1-itemset.

#2) The second step is to construct the FP tree. For this, create the root of the tree. The root is represented by null.

#3) The next step is to scan the database again and examine the transactions. Examine the first transaction and find out the itemset in it. The itemset with the max count is taken at the top, the next itemset with lower count and so on. It means that the branch of the tree is constructed with transaction itemsets in descending order of count.

#4) The next transaction in the database is examined. The itemsets are ordered in descending order of count. If any itemset of this transaction is already present in another branch (for example in the 1st transaction), then this transaction branch would share a common prefix to the root. This means that the common itemset is linked to the new node of another itemset in this transaction.

#5) Also, the count of the itemset is incremented as it occurs in the transactions. Both the common node and new node count is increased by 1 as they are created and linked according to transactions.

#6) The next step is to mine the created FP Tree. For this, the lowest node is examined first along with the links of the lowest nodes. The lowest node represents the frequency pattern length 1.

From this, traverse the path in the FP Tree. This path or paths are called a conditional pattern base.
Conditional pattern base is a sub-database consisting of prefix paths in the FP tree occurring with the lowest node (suffix).

#7) Construct a Conditional FP Tree, which is formed by a count of itemsets in the path. The itemsets meeting the threshold support are considered in the Conditional FP Tree.
#8) Frequent Patterns are generated from the Conditional FP Tree.

Example Of FP-Growth Algorithm

Support threshold=50%, Confidence= 60%
Table 1

Transaction	List of items
T1	I1,I2,I3
T2	I2,I3,I4
T3	I4,I5
T4	I1,I2,I4
T5	I1,I2,I3,I5
T6	I1,I2,I3,I4

Solution:
Support threshold=50% => 0.5*6= 3 => min_sup=3

1. Count of each item
Table 2

Item	Count
I1	4
I2	5
I3	4
I4	4
I5	2

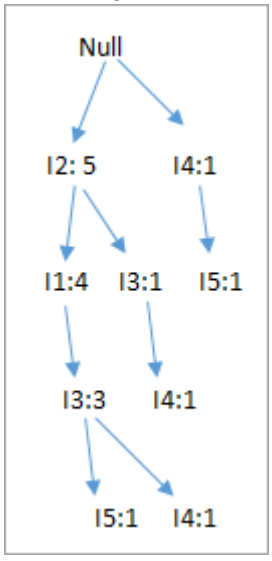
2. Sort the itemset in descending order.
Table 3

Item	Count

Item	Count
I2	5
I1	4
I3	4
I4	4

3. Build FP Tree

1. Considering the root node null.
2. The first scan of Transaction T1: I1, I2, I3 contains three items {I1:1}, {I2:1}, {I3:1}, where I2 is linked as a child to root, I1 is linked to I2 and I3 is linked to I1.
3. T2: I2, I3, I4 contains I2, I3, and I4, where I2 is linked to root, I3 is linked to I2 and I4 is linked to I3. But this branch would share I2 node as common as it is already used in T1.
4. Increment the count of I2 by 1 and I3 is linked as a child to I2, I4 is linked as a child to I3. The count is {I2:2}, {I3:1}, {I4:1}.
5. T3: I4, I5. Similarly, a new branch with I5 is linked to I4 as a child is created.
6. T4: I1, I2, I4. The sequence will be I2, I1, and I4. I2 is already linked to the root node, hence it will be incremented by 1. Similarly I1 will be incremented by 1 as it is already linked with I2 in T1, thus {I2:3}, {I1:2}, {I4:1}.
7. T5:I1, I2, I3, I5. The sequence will be I2, I1, I3, and I5. Thus {I2:4}, {I1:3}, {I3:2}, {I5:1}.
8. T6: I1, I2, I3, I4. The sequence will be I2, I1, I3, and I4. Thus {I2:5}, {I1:4}, {I3:3}, {I4 1}.



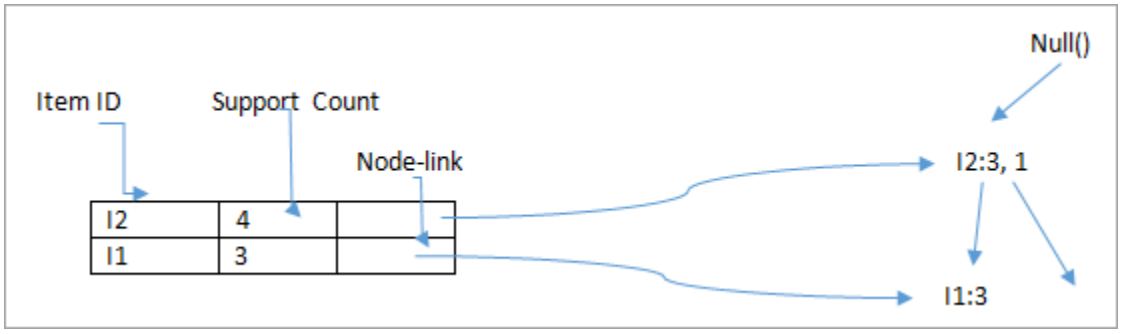
4. Mining of FP-tree is summarized below:

1. The lowest node item I5 is not considered as it does not have a min support count, hence it is deleted.
2. The next lower node is I4. I4 occurs in 2 branches , {I2,I1,I3:,I41},{I2,I3,I4:1}. Therefore considering I4 as suffix the prefix paths will be {I2, I1, I3:1}, {I2, I3: 1}. This forms the conditional pattern base.

3. The conditional pattern base is considered a transaction database, an FP-tree is constructed. This will contain {I2:2, I3:2}, I1 is not considered as it does not meet the min support count.
4. This path will generate all combinations of frequent patterns : {I2,I4:2},{I3,I4:2},{I2,I3,I4:2}
5. For I3, the prefix path would be: {I2,I1:3},{I2:1}, this will generate a 2 node FP-tree : {I2:4, I1:3} and frequent patterns are generated: {I2,I3:4}, {I1:I3:3}, {I2,I1,I3:3}.
6. For I1, the prefix path would be: {I2:4} this will generate a single node FP-tree: {I2:4} and frequent patterns are generated: {I2, I1:4}.

Item	Conditional Pattern Base	Conditional FP-tree	Frequent Patterns Generated
I4	{I2,I1,I3:1},{I2,I3:1}	{I2:2, I3:2}	{I2,I4:2},{I3,I4:2},{I2,I3,I4:2}
I3	{I2,I1:3},{I2:1}	{I2:4, I1:3}	{I2,I3:4}, {I1:I3:3}, {I2,I1,I3:3}
I1	{I2:4}	{I2:4}	{I2,I1:4}

The diagram given below depicts the conditional FP tree associated with the conditional node I3.



Advantages Of FP Growth Algorithm

1. This algorithm needs to scan the database only twice when compared to Apriori which scans the transactions for each iteration.
2. The pairing of items is not done in this algorithm and this makes it faster.
3. The database is stored in a compact version in memory.
4. It is efficient and scalable for mining both long and short frequent patterns.

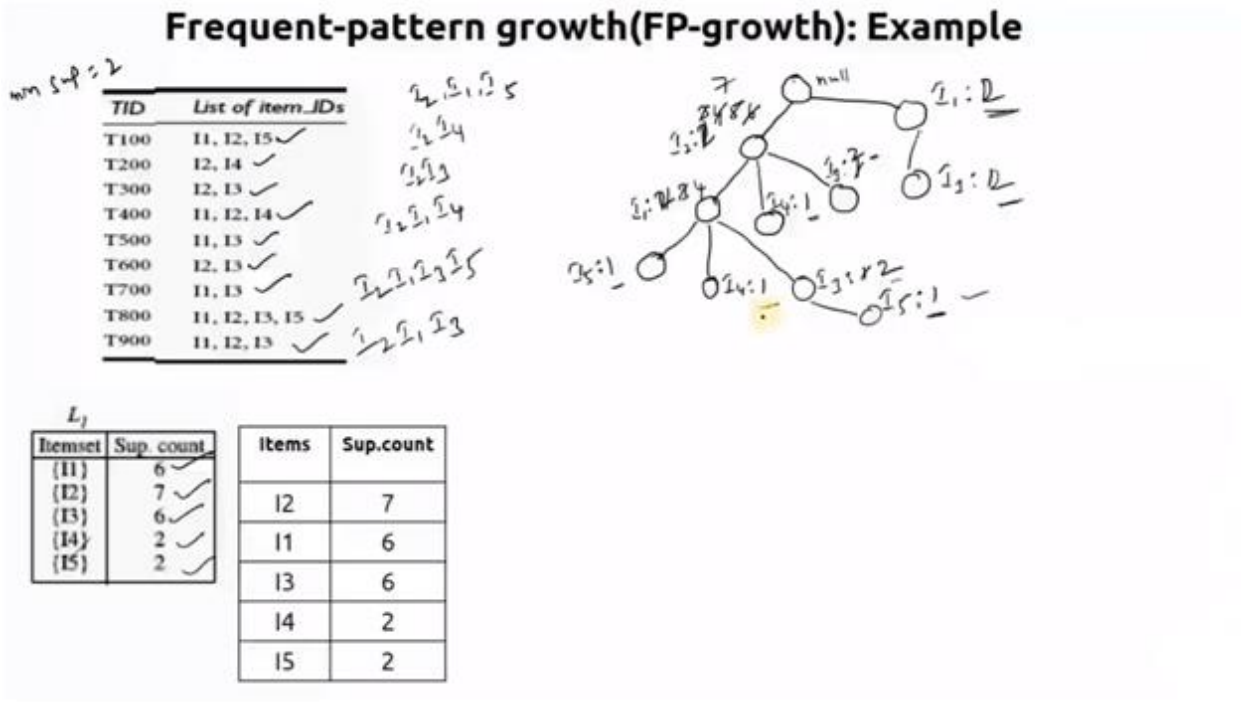
Disadvantages Of FP-Growth Algorithm

1. FP Tree is more cumbersome and difficult to build than Apriori.
2. It may be expensive.
3. When the database is large, the algorithm may not fit in the shared memory.

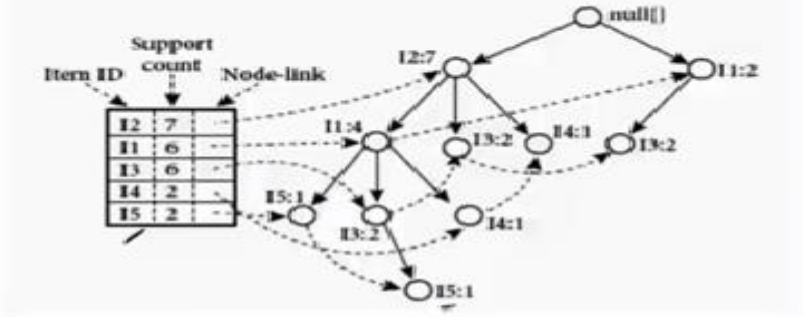
FP Growth vs Apriori

FP Growth	Apriori
Pattern Generation	
FP growth generates pattern by constructing a FP tree	Apriori generates pattern by pairing the items into singletons, pairs and triplets.
Candidate Generation	

FP Growth	Apriori
There is no candidate generation	Apriori uses candidate generation
Process	
The process is faster as compared to Apriori. The runtime of process increases linearly with increase in number of itemsets.	The process is comparatively slower than FP Growth, the runtime increases exponentially with increase in number of itemsets
Memory Usage	
A compact version of database is saved	The candidates combinations are saved in memory



Frequent-pattern growth(FP-growth): Example



Item	Conditional Pattern Base	Conditional FP-tree	Frequent Patterns Generated
I5	{{I2, I1: 1}, {I2, I1, I3: 1}}	⟨I2: 2, I1: 2⟩	{I2, I5: 2}, {I1, I5: 2}, {I2, I1, I5: 2}
I4	{{I2, I1: 1}, {I2: 1}}	⟨I2: 2⟩	{I2, I4: 2}
I3	{{I2, I1: 2}, {I2: 2}, {I1: 2}}	⟨I2: 4, I1: 2⟩, ⟨I1: 2⟩	{I2, I3: 4}, {I1, I3: 4}, {I2, I1, I3: 2}
I1	{{I2: 4}}	⟨I2: 4⟩	{I2, I1: 4}

CLASSIFICATION ALGORITHMS

PREDICT PROBABILITY –

ON THE BASES OF PROBABILITY WHICH IS IN TERM OF VALUE.

WE ->ON THE BASES OF VALUES – CALCULATE THE CLASSIFICATION –LIKE PASS/FAIL

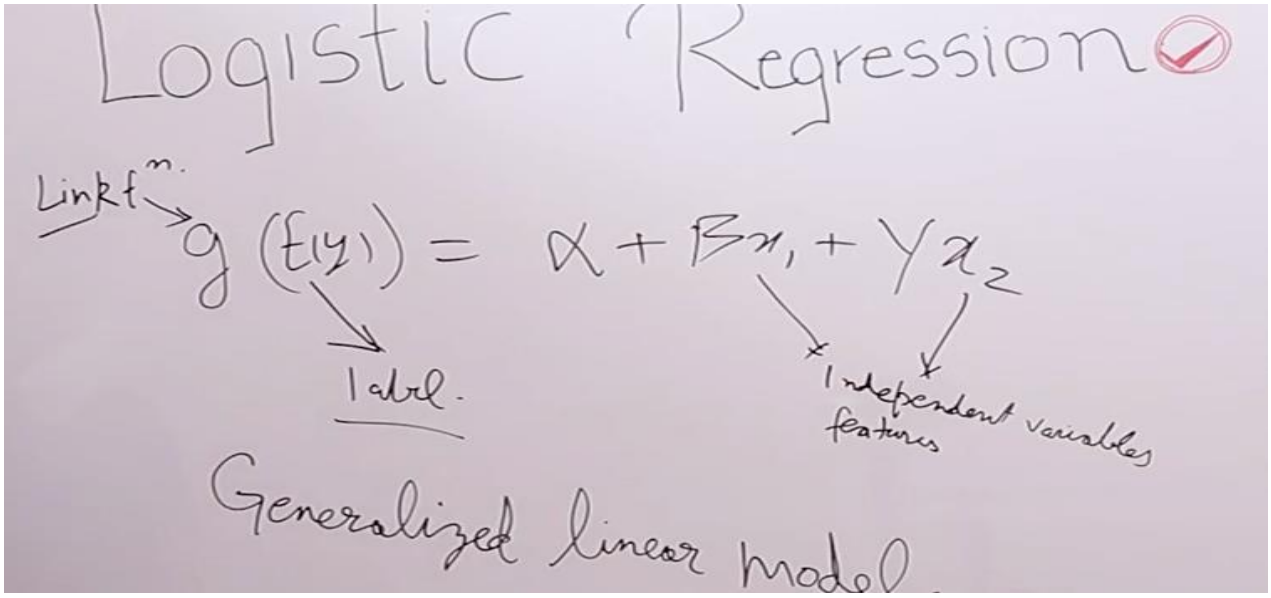
INTIAL ITS FOR BINARY PREDICATION VALUE

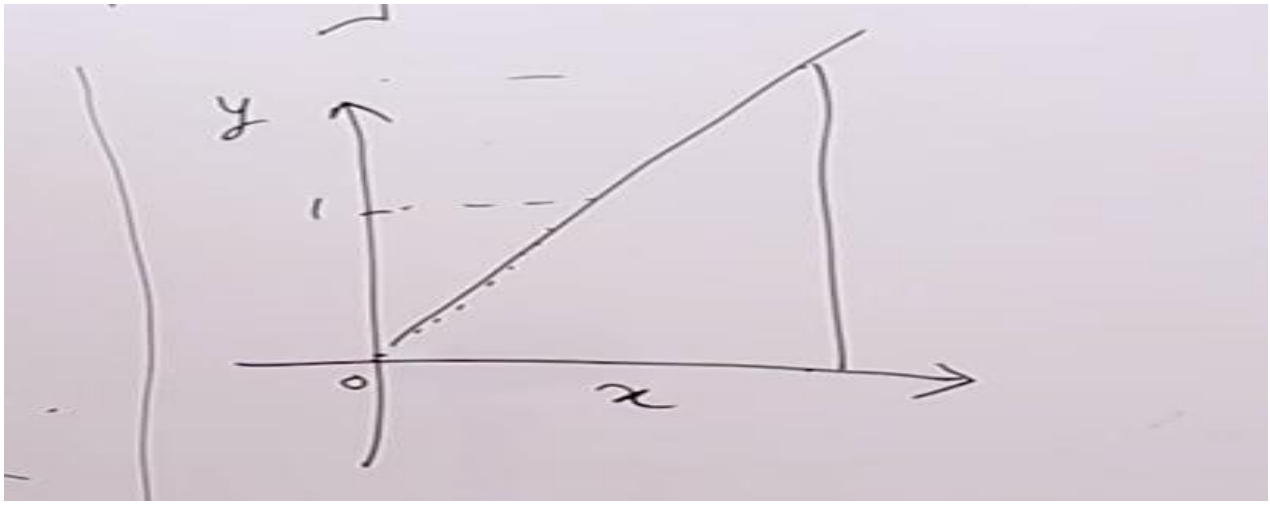
ASSUME THAT WE HAVE FEATURES VALUES—0,1,2,3,4 ->/ AND ASSUMED THAT PROBABILITY OF THESE FEATURES ARE 0.5 AND THUS WE FIND THE PROBABILITY BELOW 0.5 IS FAIL

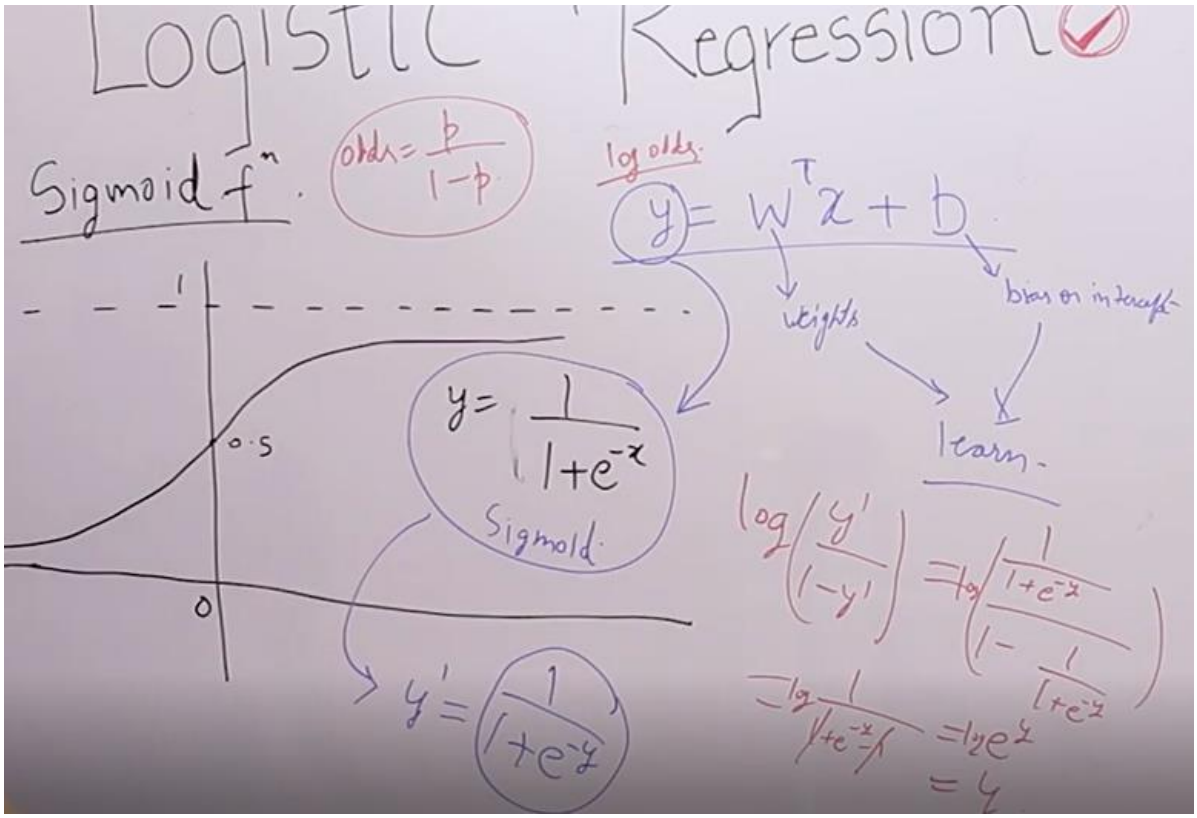
IT IS CALLED REGRESSION BECAUSE – IT PREDICT THE VALUE (PROBABILITY IN NUMBER)

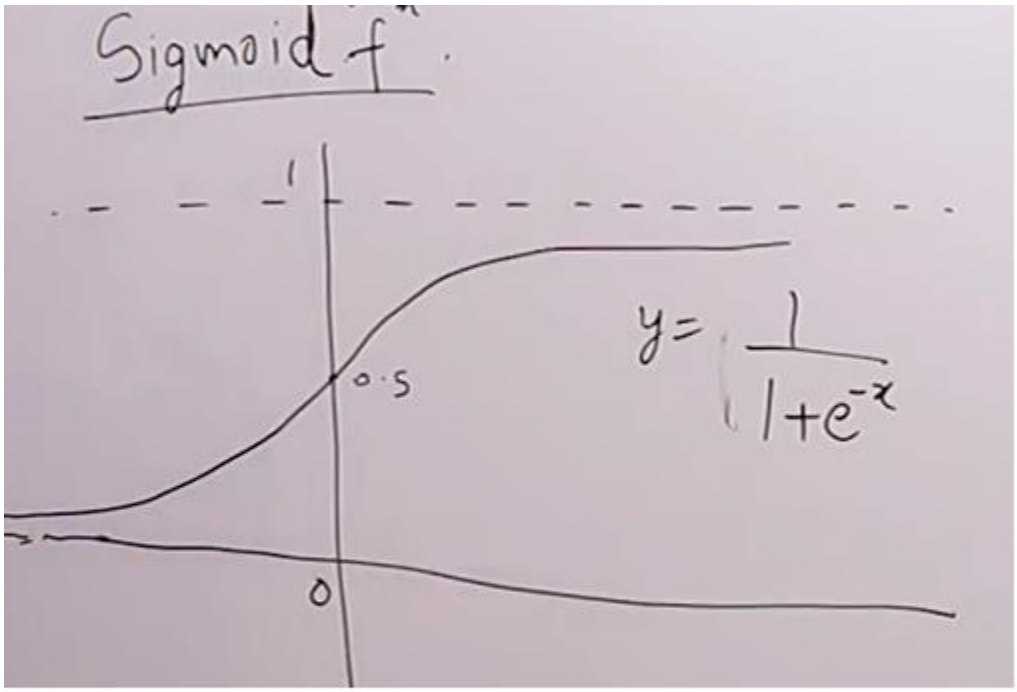
THIS IS GENERALIZED LINEAR CLASS ALGORITHMS

GENERALIZED LINEAR CLASS ALGORITHMS






Logistic Regression
Sigmoid fn.
odds = p / 1-p
log odds:
y = W^T x + b
weights
bias or intercept
learn.
1
0.5
0
y = 1 / 1+e^-z
Sigmoid.
y' = 1 / 1+e^-y
log(y' / 1-y') = log(1/1+e^-z / 1 - 1/1+e^-z)
= log 1 / 1+e^-z - 1 = log e^z
= z


Sigmoid f^n.
1
0.5
0
$y = \frac{1}{1+e^{-x}}$

Sigmoid f^n.

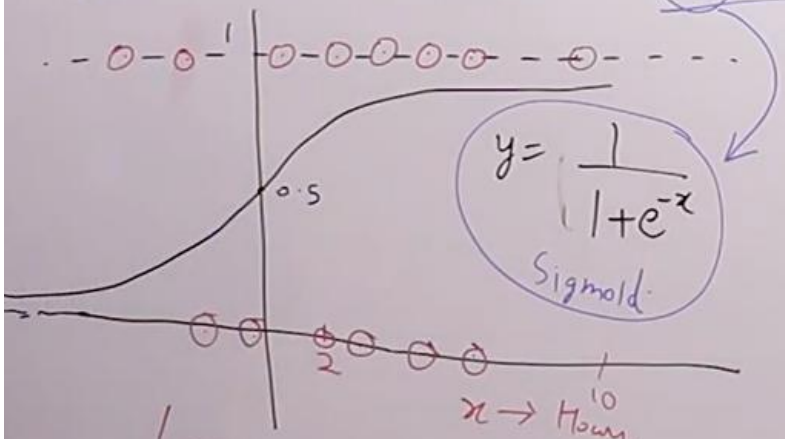
odds = $\frac{p}{1-p}$

log odds.

$y = W^T x + b$

weights

bias or intercept



$y = \frac{1}{1+e^{-x}}$

Sigmoid.

Hours	Pass/fail = 0
10	1
2	0
1	0
7	1
11	0

$Loss = -y \log y' \quad y = 1$

$Loss = -(1-y) \log(1-y') \quad y = 0$

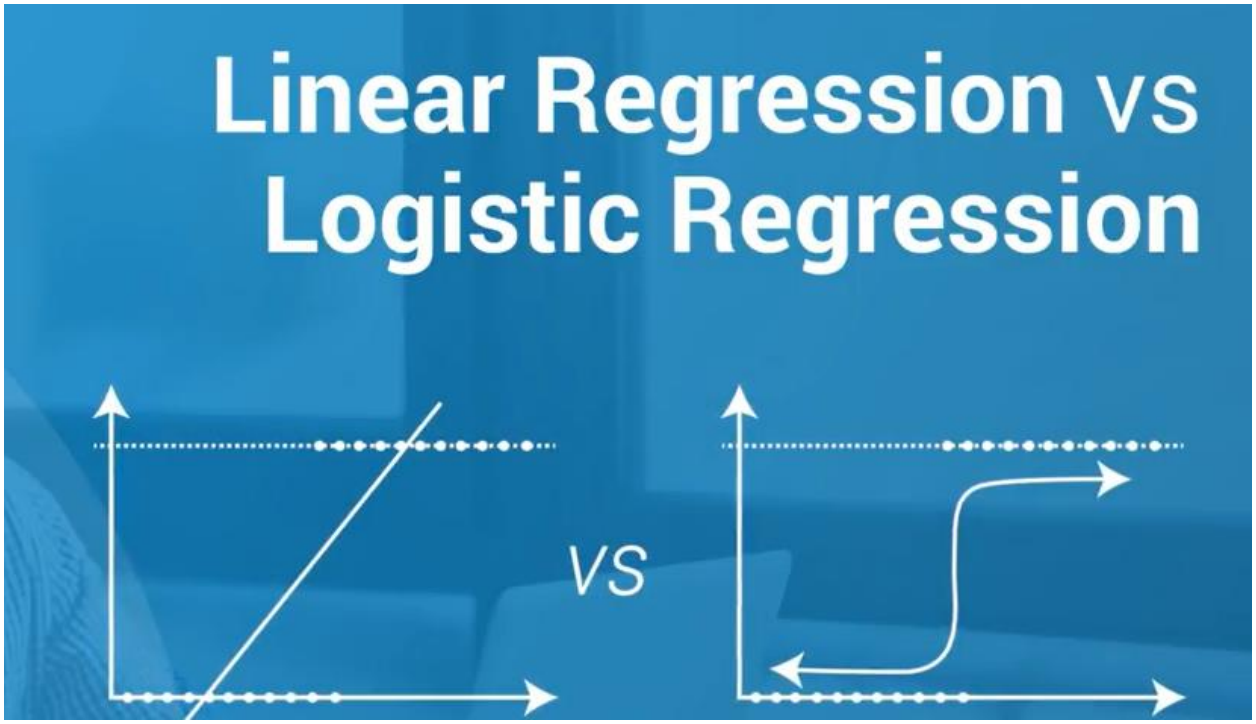
$Loss = -y \log y' - (1-y) \log(1-y')$

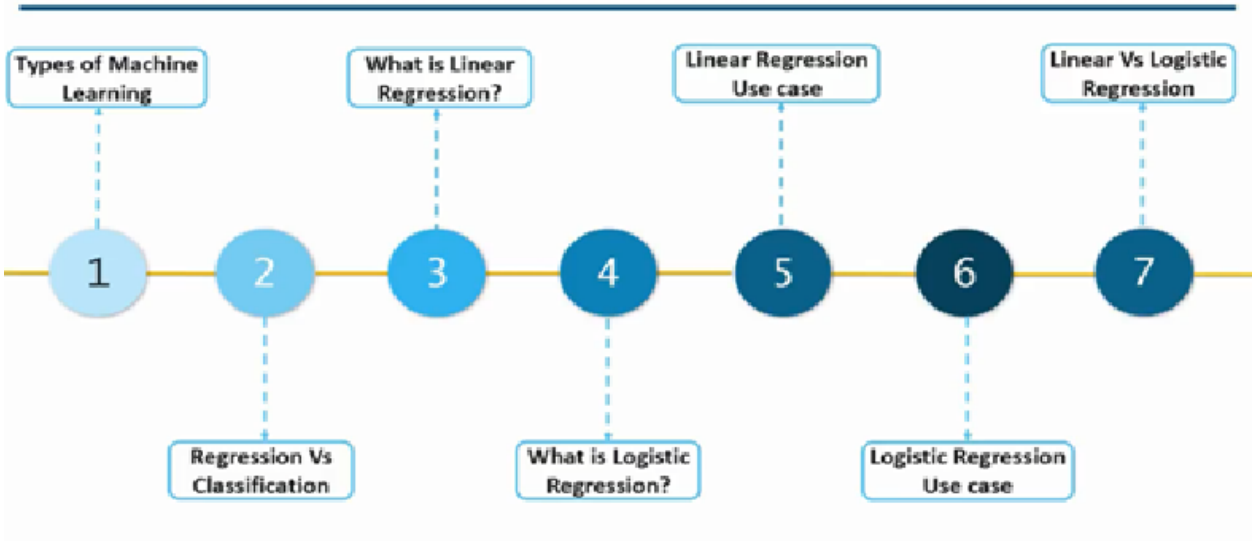
GD

$w_1, w_2 \ldots$

b

$y' = \frac{1}{1+e^{-(w_1 x_1 + b)}} = 0.22$


Linear Regression vs
Logistic Regression
VS


Types of Machine Learning
What is Linear Regression?
Linear Regression Use case
Linear Vs Logistic Regression
1
2
3
4
5
6
7
Regression Vs Classification
What is Logistic Regression?
Logistic Regression Use case

Types Of Machine Learning

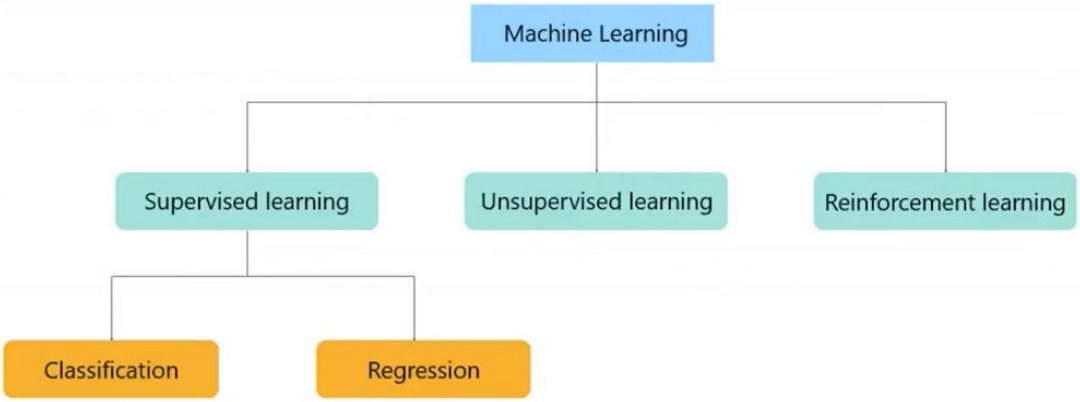
Types Of Machine Learning

Supervised Learning

Unsupervised Learning

Reinforcement Learning

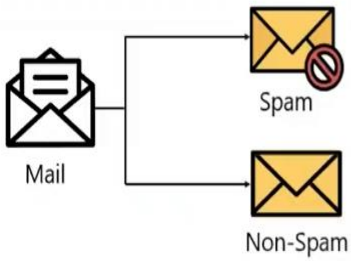
Regression And Classification



Classification

Classification is the task of predicting a discrete class label

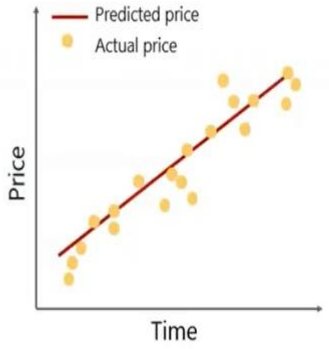
- In a classification problem data is classified into one of two or more classes
- A classification problem with two classes is called binary, more than two classes is called a multi-class classification



Regression

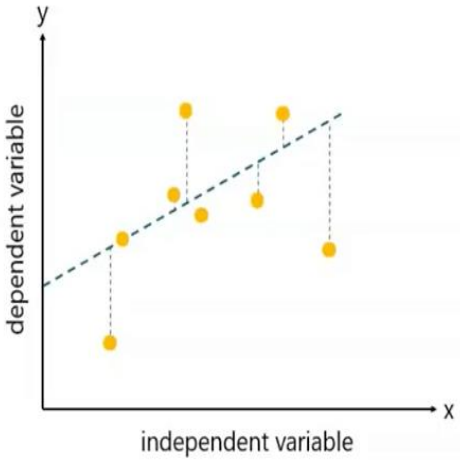
Regression is the task of predicting a continuous quantity

- A regression problem requires the prediction of a quantity
- A regression problem with multiple input variables is called a multivariate regression problem



What Is Linear Regression?

Linear Regression is a method to predict dependent variable (Y) based on values of independent variables (X). It can be used for the cases where we want to predict some continuous quantity.



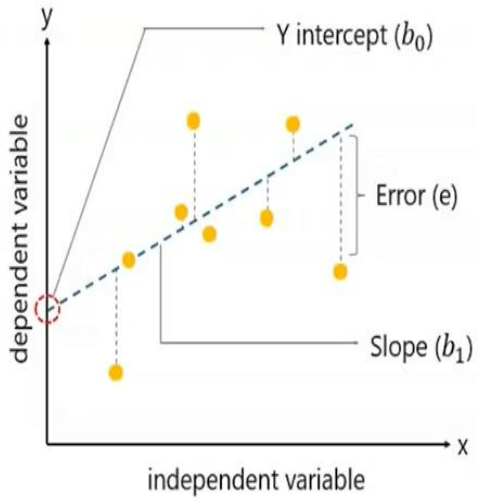
- *Dependent variable (Y):*
The response variable who's value needs to be predicted.

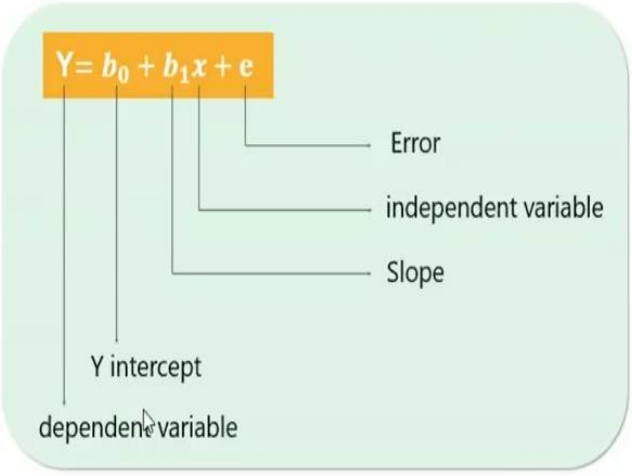
- *Independent variable (X):*
The predictor variable used to predict the response variable.

The following equation is used to represent a linear regression model:

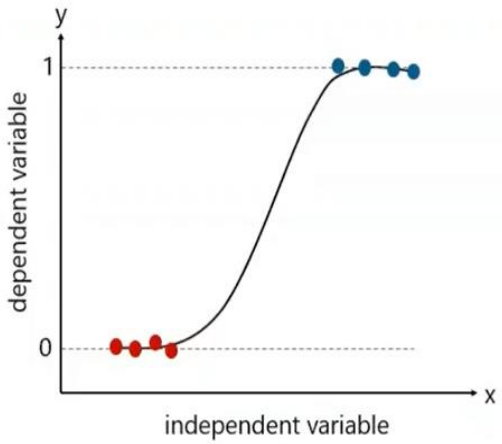
$$Y = b_0 + b_1 x + e$$

Linear Regression is a method to predict dependent variable (Y) based on values of independent variables (X). It can be used for the cases where we want to predict some continuous quantity.





Logistic Regression is a method used to predict a dependent variable, given a set of independent variables, such that the dependent variable is categorical.



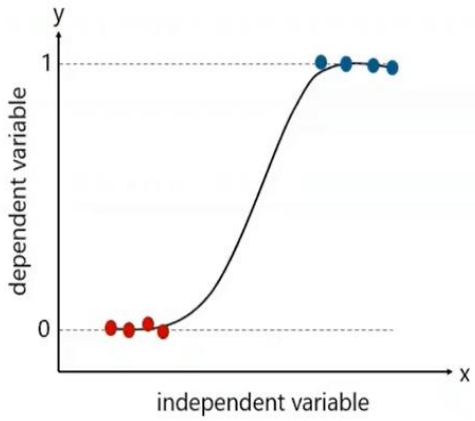
- *Dependent variable (Y):*
The response binary variable holding values like 0 or 1, Yes or No, A, B or C

- *Independent variable (X):*
The predictor variable used to predict the response variable.

The following equation is used to represent a linear regression model:

$$\log\left(\frac{Y}{1-Y}\right) = C + B1X1 + B2X2 + \ldots.$$

Logistic Regression is a method used to predict a dependent variable, given a set of independent variables, such that the dependent variable is categorical.



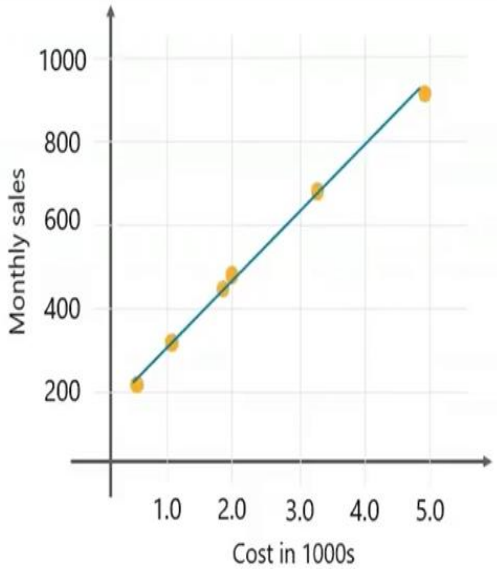
$$\log\left(\frac{Y}{1-Y}\right) = C + B1X1 + B2X2 + \ldots.$$

- Y is the probability of an event to happen which you are trying to predict
- x1, x2 are the independent variables which determine the occurrence of an event i.e. Y
- C is the constant term which will be the probability of the event happening when no other factors are considered

Linear Regression Use Case

To forecast monthly sales by studying the relationship between the monthly e-commerce sales and the online advertising costs.

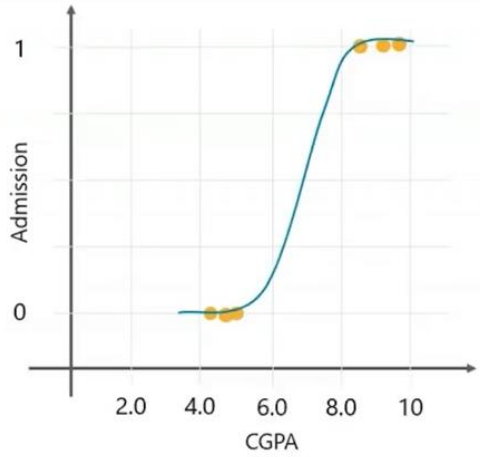
Monthly sales	Advertising cost In 1000s
200	0.5
900	5
450	1.9
680	3.2
490	2.0
300	1.0



Logistic Regression Use Case

To predict if a student will get admitted to a school based on his CGPA.

Admission	CGPA
0	4.2
0	5.1
0	5.5
1	8.2
1	9.0
1	9.1



Linear Regression Vs Logistic Regression

	Linear Regression	Logistic Regression
1 Definition	To predict a continuous dependent variable based on values of independent variables	*To predict a categorical dependent variable based on values of independent variables*
2 Variable Type	Continuous dependent variable	Categorical dependent variable
3 Estimation method	Least square estimation	Maximum like-hood estimation
4 Equation	$Y = b_0 + b_1 x + e$	$\log\left(\frac{Y}{1-Y}\right)$ = C + B1X1 + B2X2 + ...
5 Best fit line	Straight line	Curve
6 Relationship between DV & IV	Linear relationship between the dependent and independent variable	Linear relationship is not mandatory
7 Output	Predicted integer value	Predicted binary value (0 or 1)
8 Applications	Business domain, forecasting sales	Classification problems, cybersecurity, image processing

least squares method

- The least squares method is a statistical procedure to find the best fit for a set of data points by minimizing the sum of the offsets or residuals of points from the plotted curve.
- Least squares regression is used to predict the behavior of dependent variables.

Least Squares Fit of a Straight Line: Example

Fit a straight line to the x and y values in the following Table:

x_i	y_i	$x_i y_i$	x_i^2
1	0.5	0.5	1
2	2.5	5	4
3	2	6	9
4	4	16	16
5	3.5	17.5	25
6	6	36	36
7	5.5	38.5	49
28	**24**	**119.5**	**140**

$$\sum x_i = 28 \quad \sum y_i = 24.0$$

$$\sum x_i^2 = 140 \quad \sum x_i y_i = 1195$$

$$\bar{x} = \frac{28}{7} = 4$$

$$\bar{y} = \frac{24}{7} = 3.428571$$

Maximum Like hood estimation

Logistic regression (5) (maximum likelihood estimation)

- Let y_i be the observed output (1 or 0) for the corresponding input x_i
- let $Pr(y = 1 | x_i)$ be the probability obtained by the model.

the probability of obtaining the measured output value y_i is

- $Pr(y = 1 | x_i)$ if $y_i = 1$,
- $Pr(y = 0 | x_i) = 1 - Pr(y = 1 | x_i)$ if $y_i = 0$.

K-nearest neighbour classification

query ⇒ x = (Maths = 6, CS = 8), (K=3)

Pass

	maths	Cs	Result
1)	4	3	Fail
2)	6	7	Pass
3)	7	8	Pass
4)	5	5	Fail
5)	8	8	Pass

I $\sqrt{(6-4)^2+(8-3)^2} = \sqrt{29} = 5.38$

II $\sqrt{(6-6)^2+(8-7)^2} = 1$

III $\sqrt{(6-7)^2+(8-8)^2} = 1$

IV $\sqrt{(6-5)^2+(8-5)^2} = \sqrt{10} = 3.16$

V $\sqrt{(6-8)^2+(8-8)^2} = 2$

Euclidean distance :-

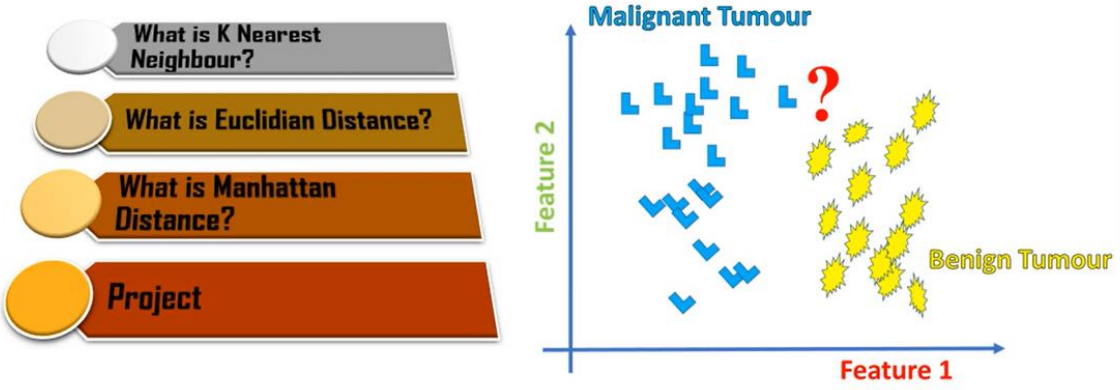
Euclidean distance :-

V $\sqrt{(6-8)^2+(8-8)^2} = 2$

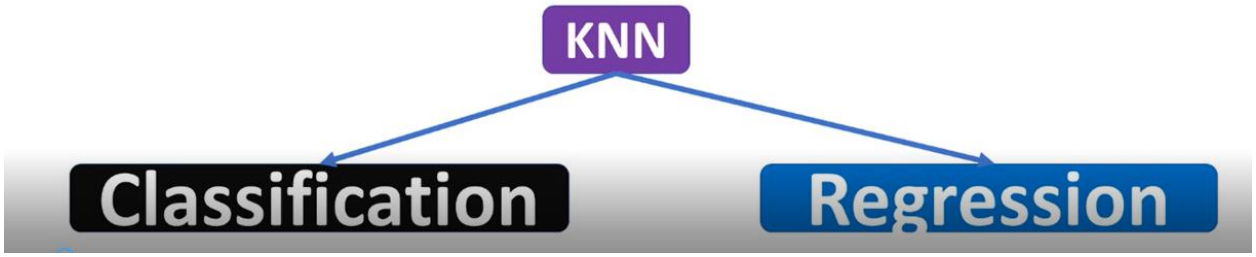
$$d = \sqrt{|x_{01}-x_{A1}|^2 + |x_{02}-x_{A2}|^2}$$

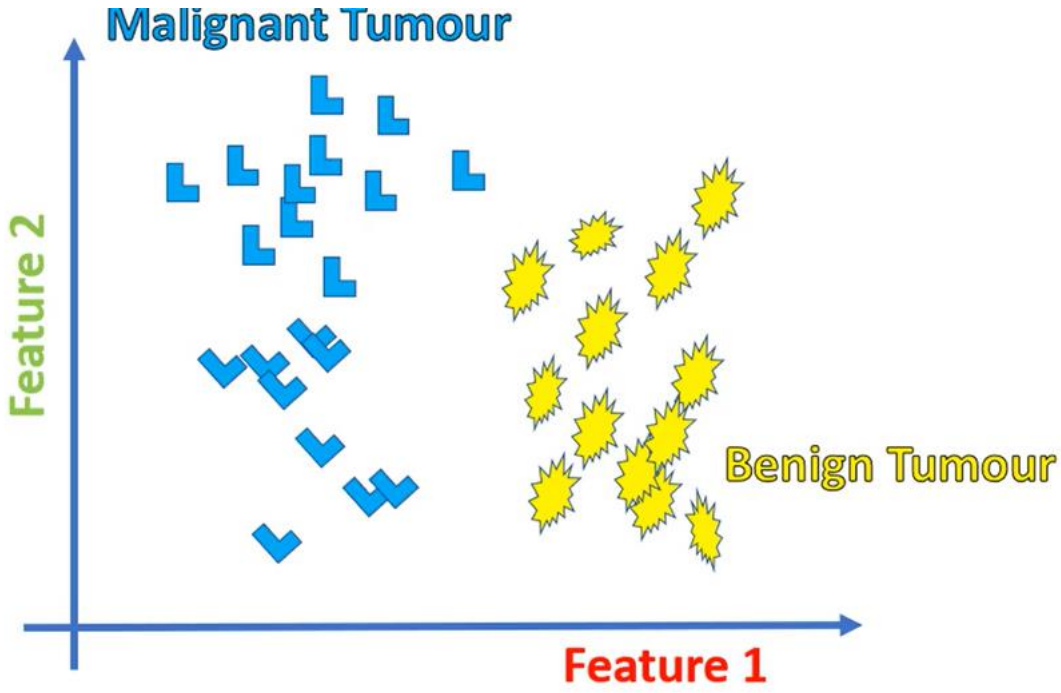
P P P = 3P

3 > 0 ≠ OF



- K Nearest Neighbor is supervised Machine Learning algorithms that can perform both classification and regression tasks using numbers(K) of neighbors(Instances).

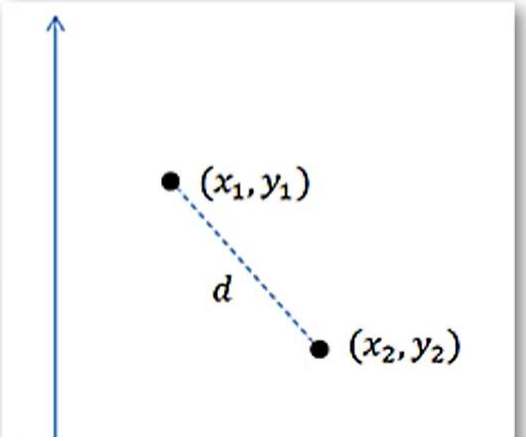




Steps of K Nearest Neighbor (KNN)

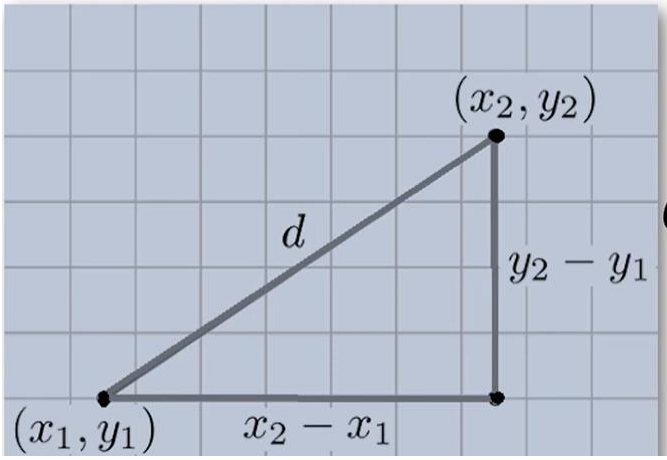
1. Get Data
2. Define K Neighbors
3. Calculate the Neighbors distance
4. Assign New instance to Majority of Neighbors

What is Euclidian Distance?



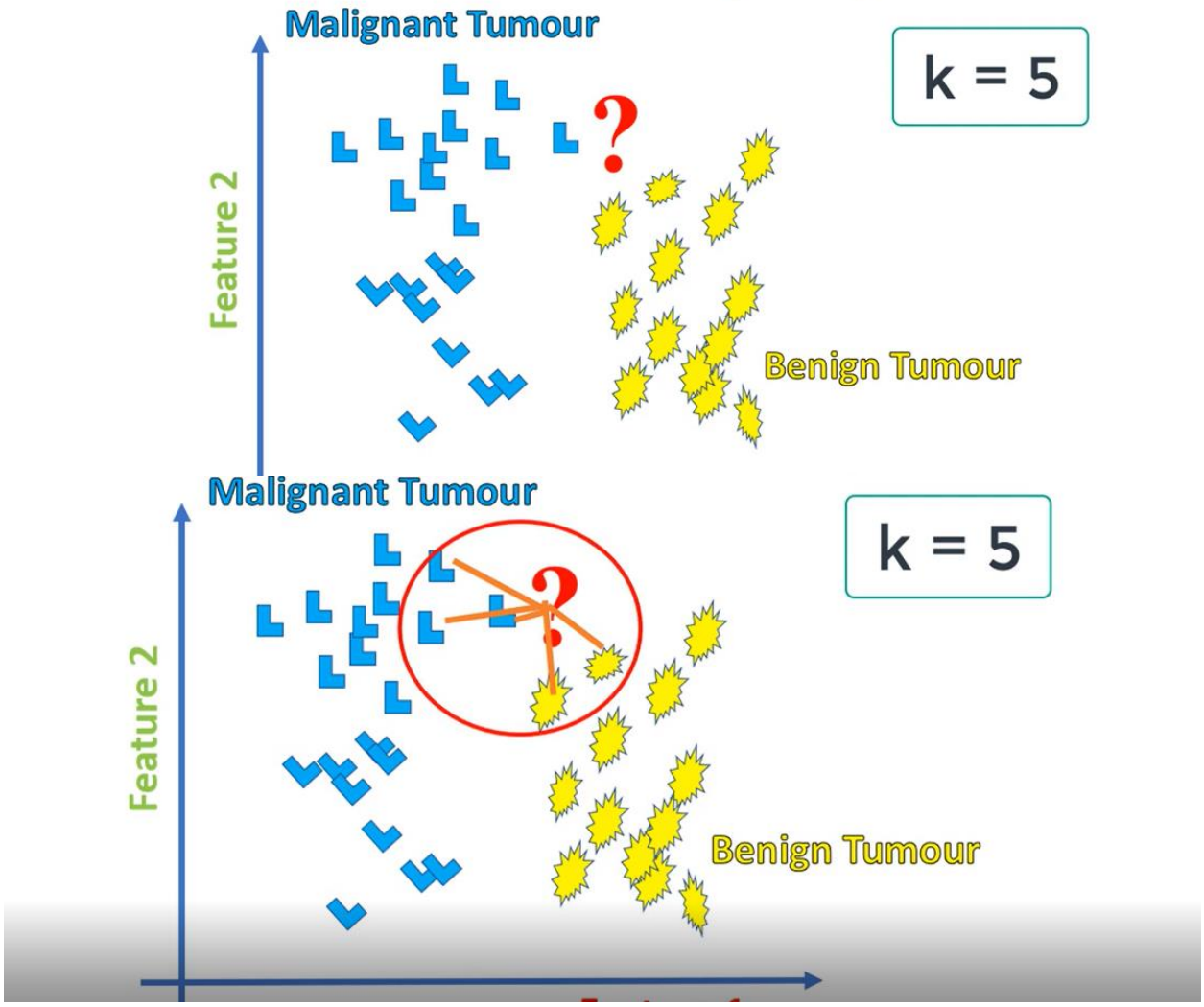
$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

What is Manhattan Distance?



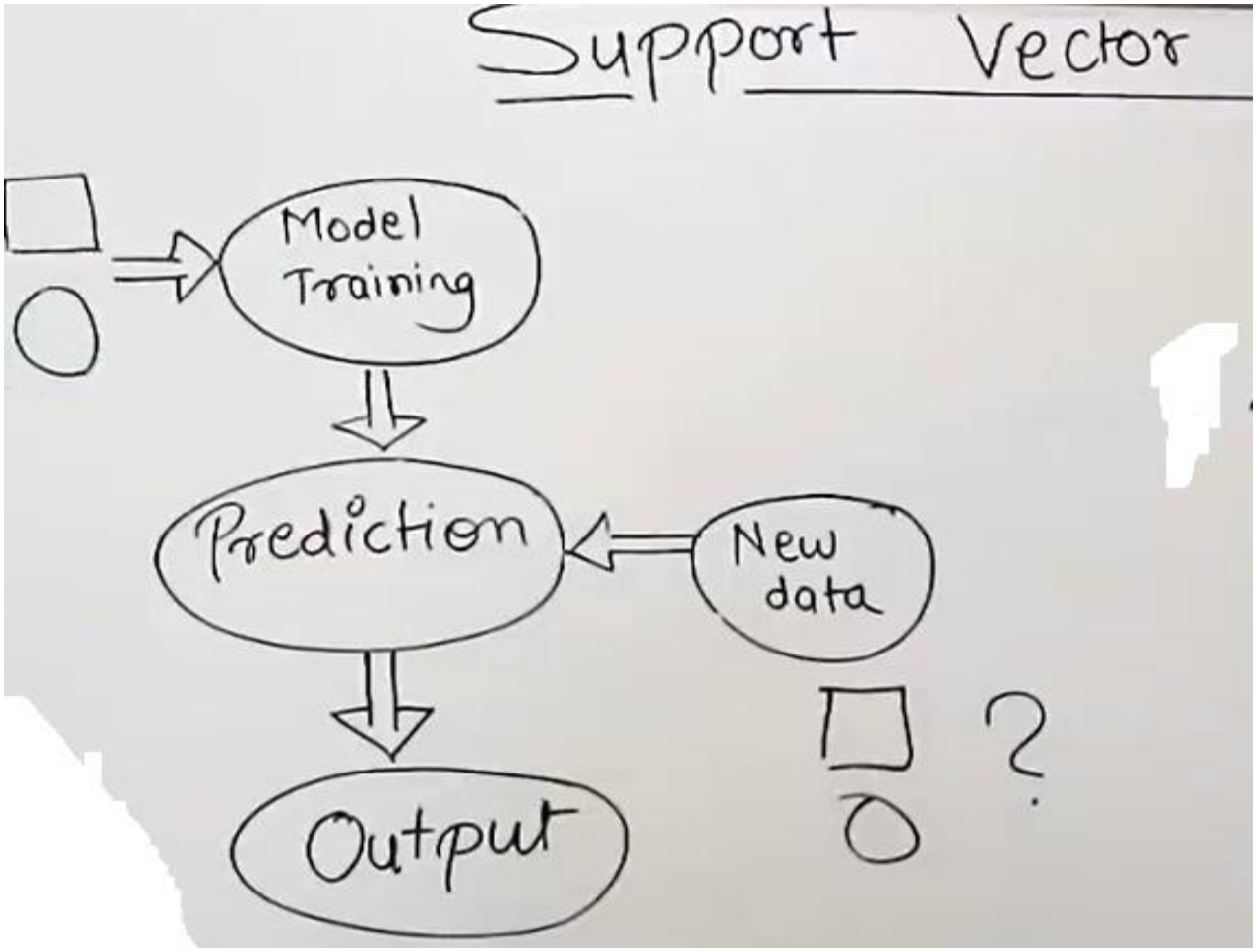
$$d = |x_2 - x_1| + |y_2 - y_1|$$

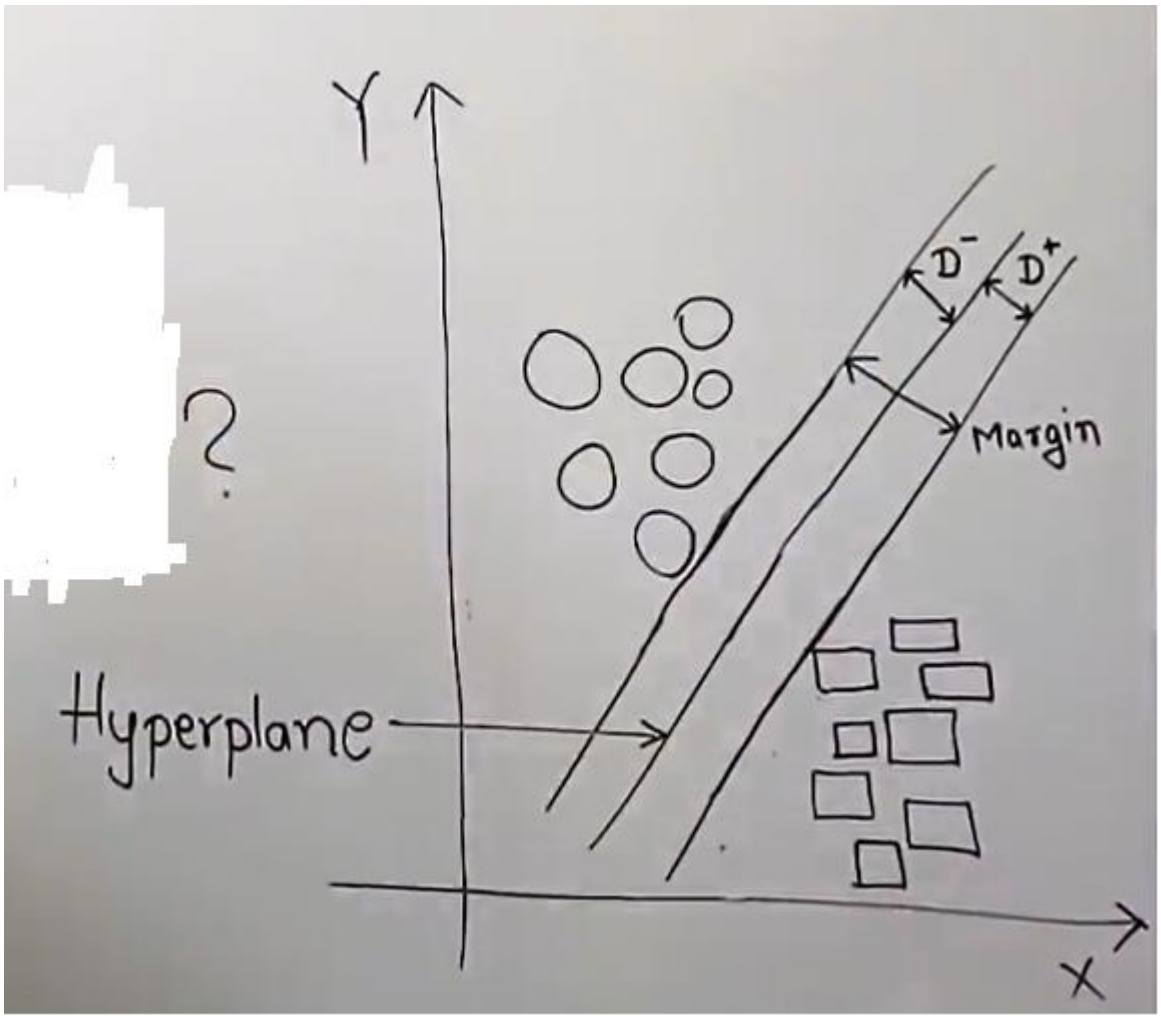
What is K Nearest Neighbor (KNN)?

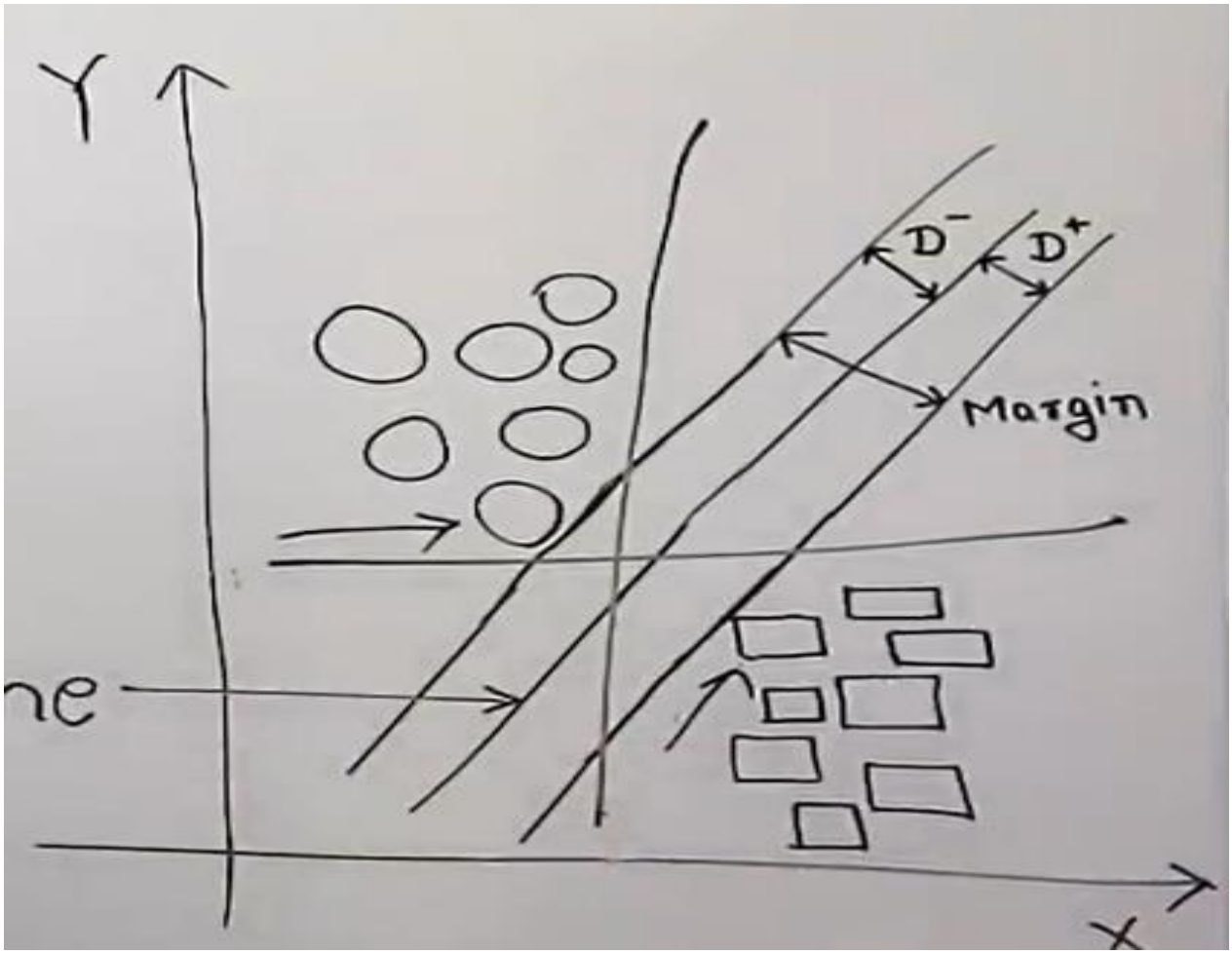


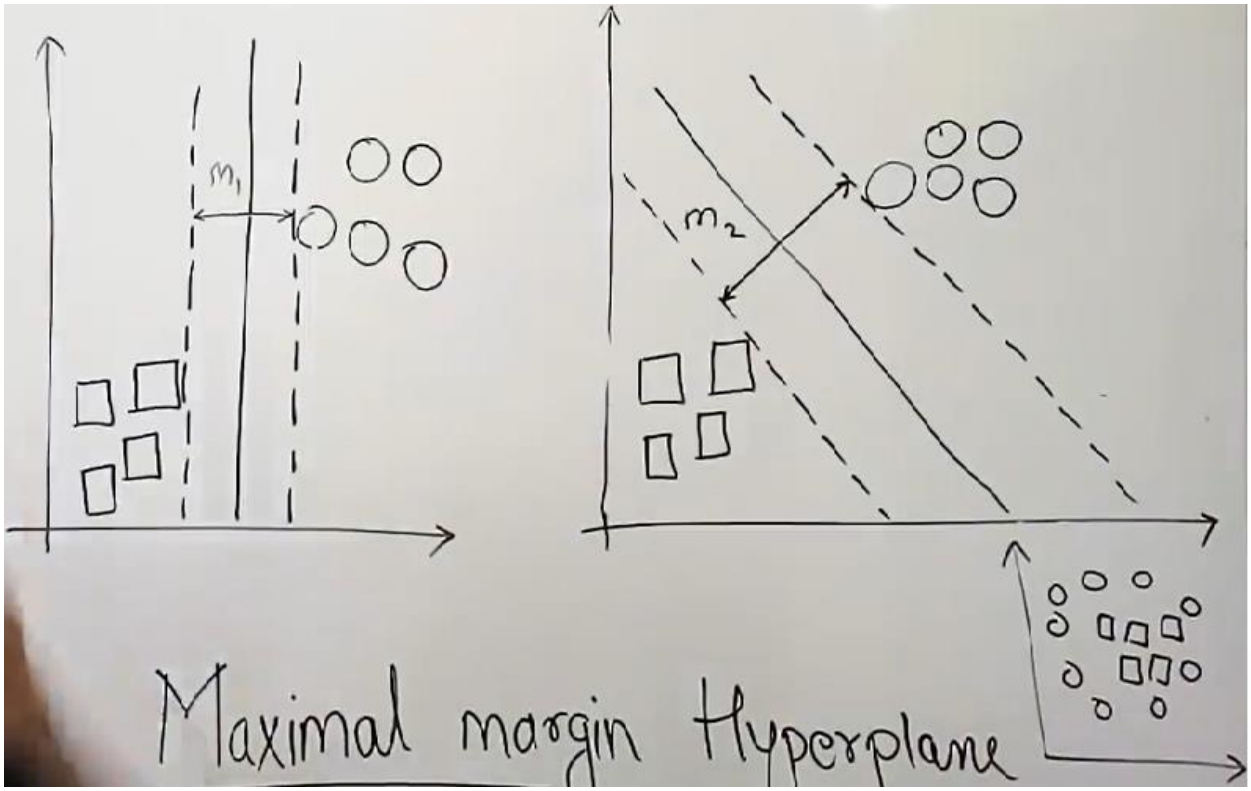
SV Machine is supervising learning algorithms. It solves the classification problems. Questions related to svm.

1. Why we call it support vector?
2. What is margin? Explain the way to decide the margin.
3. What is Maximal Margin Hyper plane (MMH) and it's Role?
4. What are Linearly Separable data and Non-Linearly Separable data? Explain with graph.

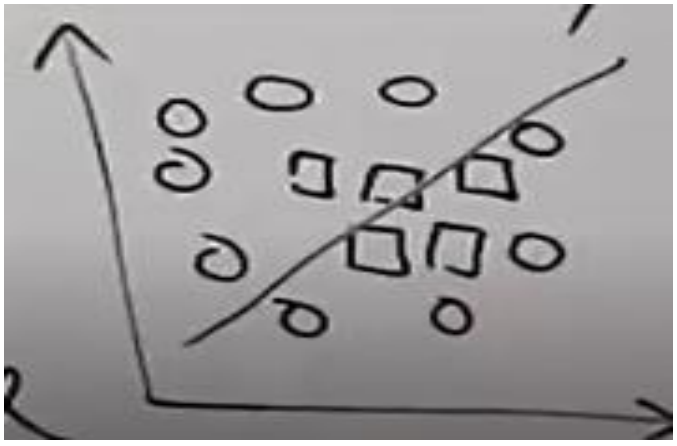



Y
?
D⁻
D⁺
Margin
Hyperplane
X

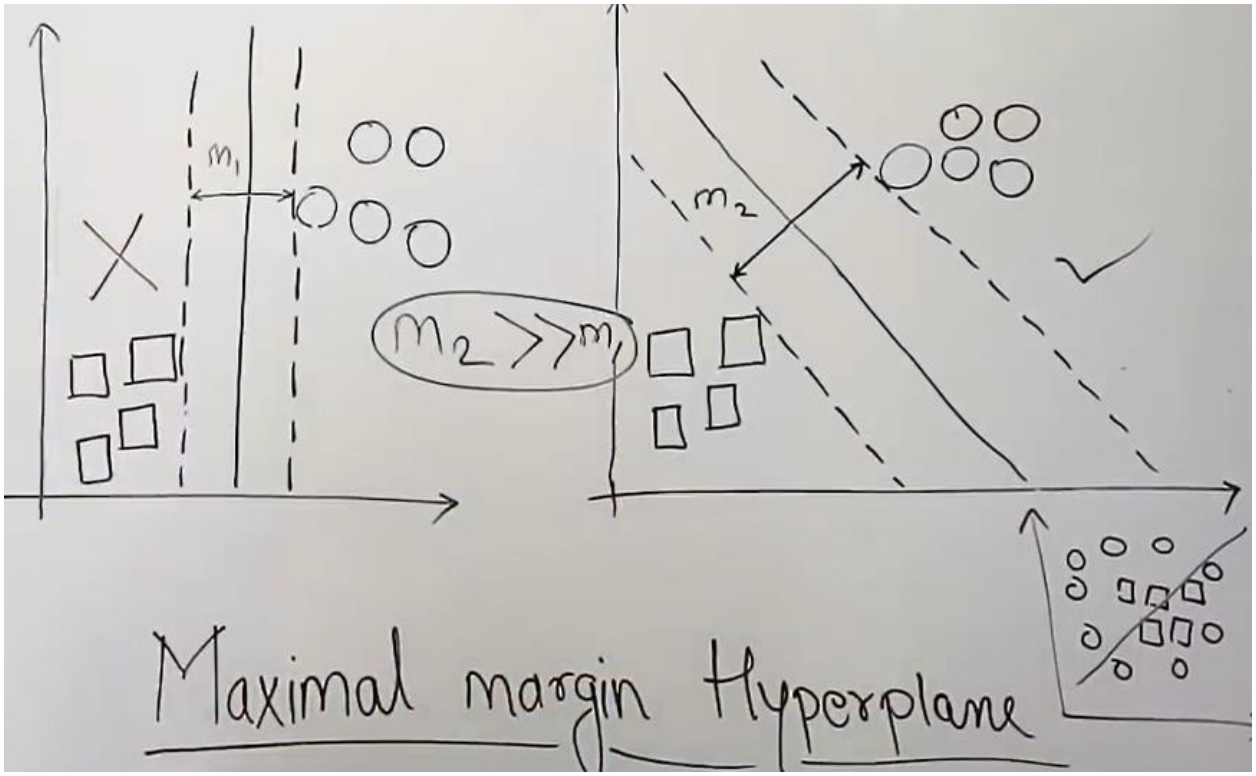

Y
D⁻
D⁺
Margin
ne
X



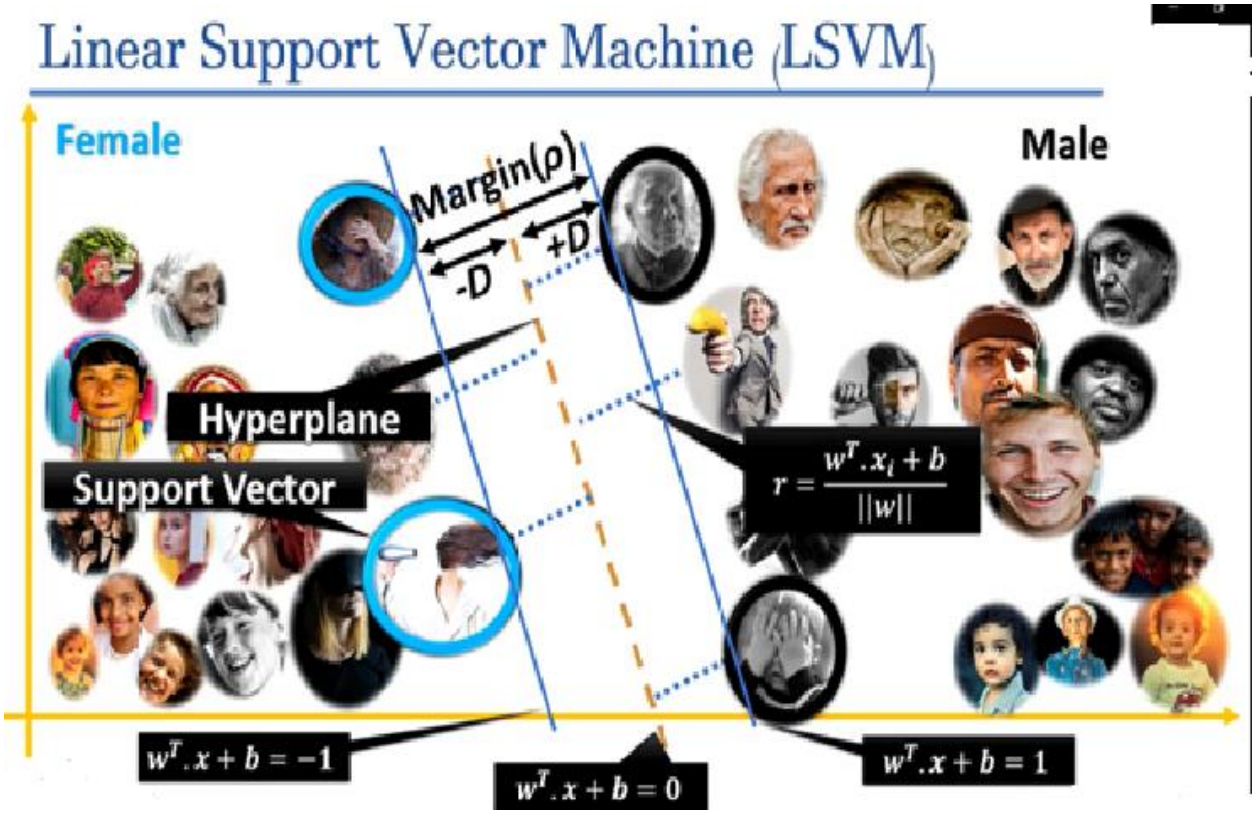
Linearly Separable data

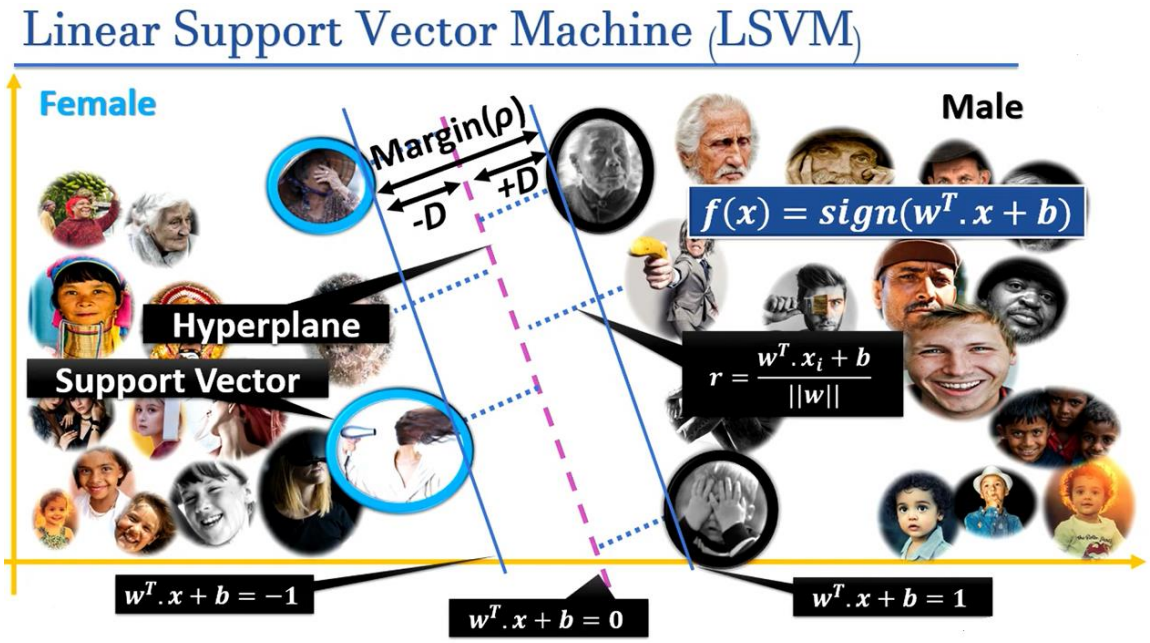


Non-Linearly Separable data

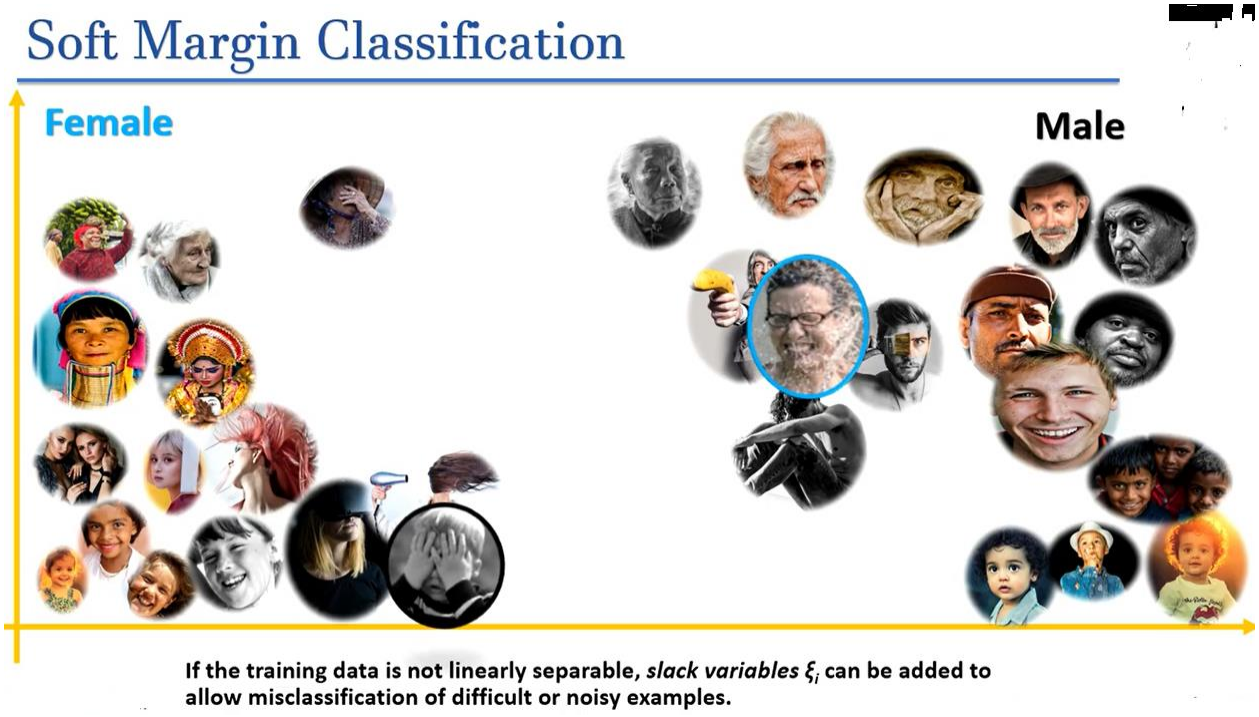


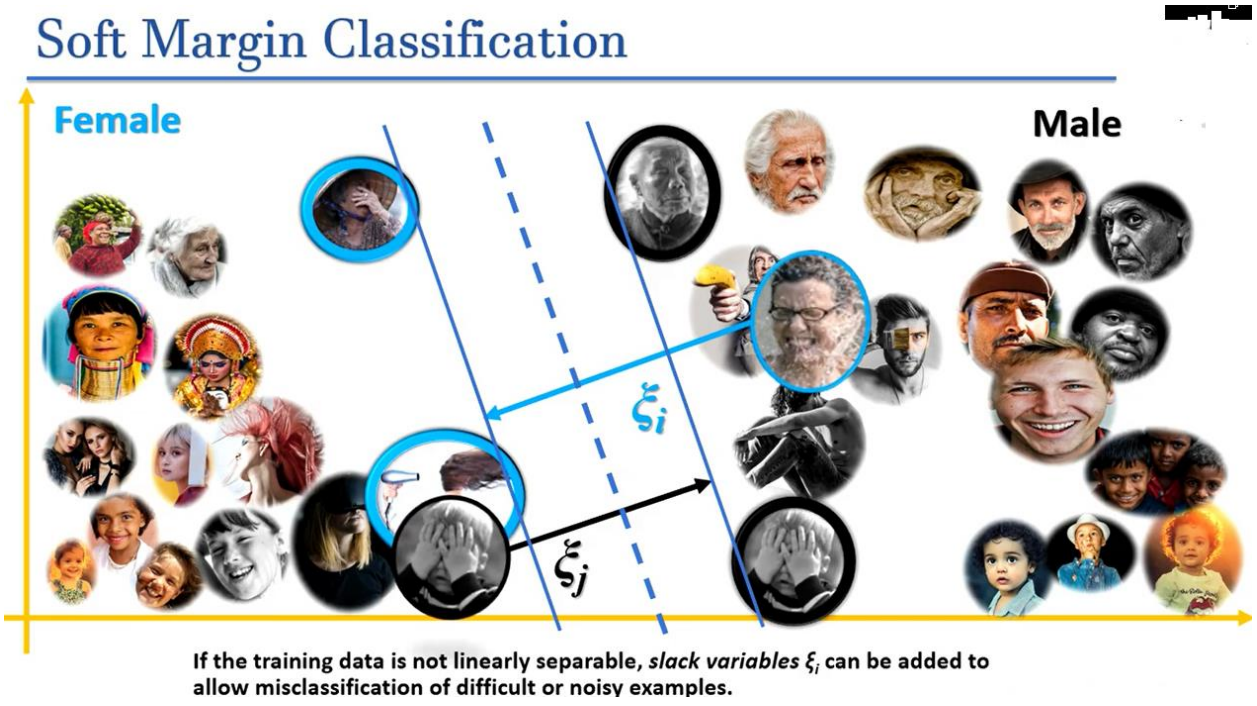
Maximum margin consider more miss classification data as compare to minimum margin. So MMH should be selected because it is good for future predication purpose. The reason behind this is that it contain minimum error rate so ultimately the accuracy level is high which good for prediction model.





LSVM with svm equation





**Calculate the female data distance from hyper plane of h1 and same for male data distance from h2 hype plane line and this is known as slack variable or soft Margin Classification.

Psodocode of sv

MACHINE LEARNING

Example 1

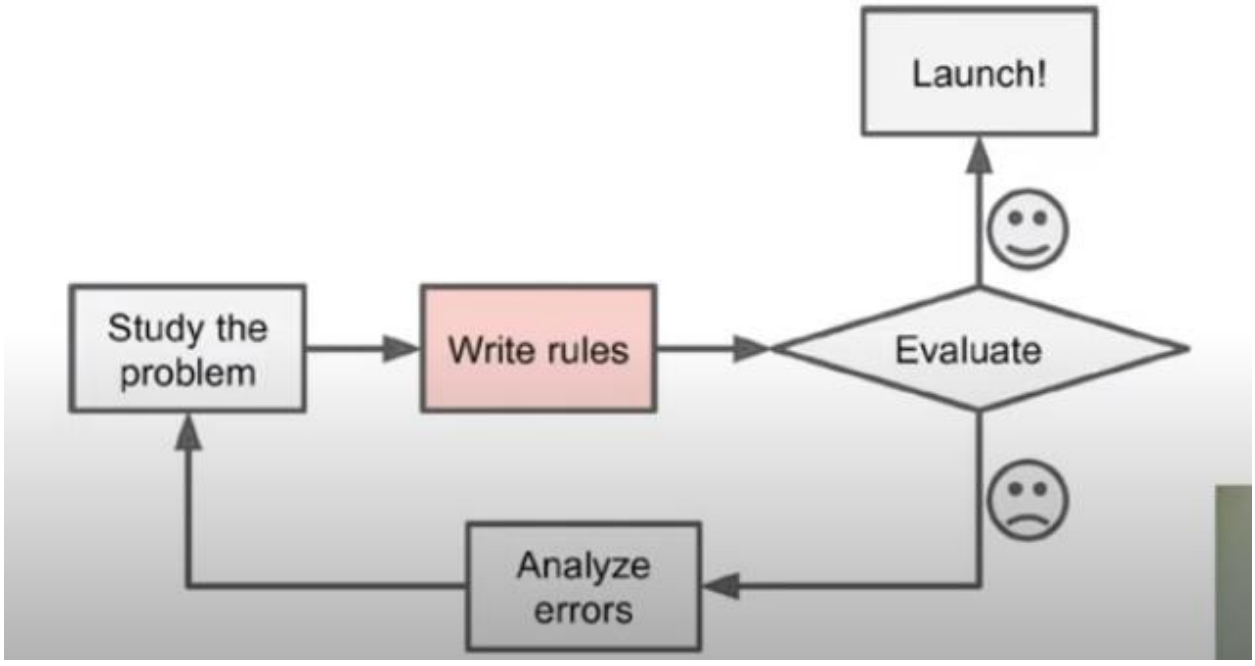
You want to write a spam filter using traditional programming techniques

- ❑ Find patterns , example may be "4U" , "Credit Card" , "Free" etc.
- ❑ Write code to detect such pattern
- ❑ Test Program, if not good then repeat step 1 and 2

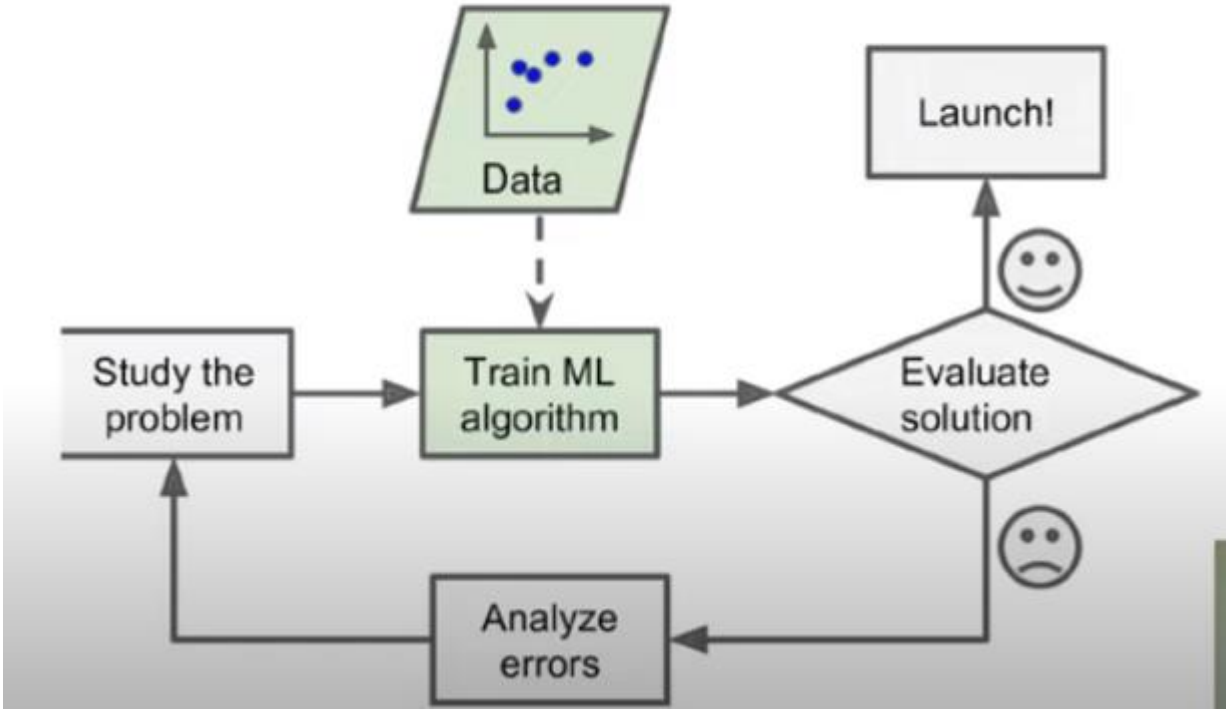
Issues:

- Program is long and hard to maintain.
- Spammers will try to outsmart you.

Workflow without ML



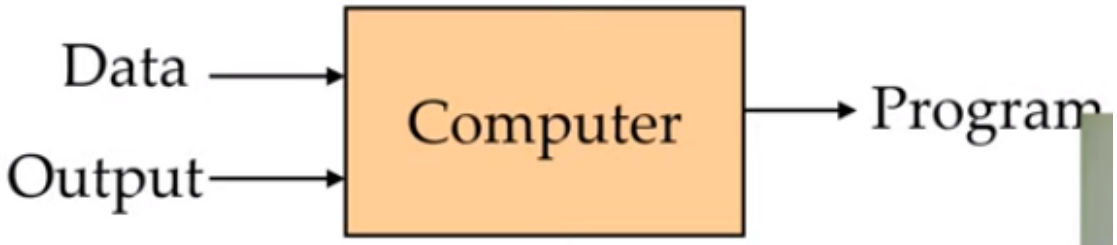
Workflow with ML



Another Example

Traditional Programming

Machine Learning



Definition

" Machine learning is the field of study which gives the computers the ability to learn without being explicitly programmed" - Arther Samuels 1959

"A computer program is said to learn from experience **E** with respect to some class of tasks **T** and performance measure **P**, if its performance at tasks in **T**, as measured by P, improves with experience **E**." - Tom Mitchells 1997

Identify P , E and T

- Identifying Spam Emails

Task **T** : Classification Email
Performance Measure **P**: percentage of emails correctly classified
Training Experience E: Database of emails with labels

- Handwriting recognition learning problem

Task **T** : Recognizing and classifying handwritten words within images.
Performance Measure **P**: percentage of words correctly classified.
Training Experience E: A database of handwritten words with given classifications.