UNIT-II

Supervised Learning(Regression/Classification)

Basic Methods: Distance based Methods, Nearest Neighbours, Decision Trees, Naive Bayes,

Linear Models: Linear Regression, Logistic Regression, Generalized Linear Models, Support

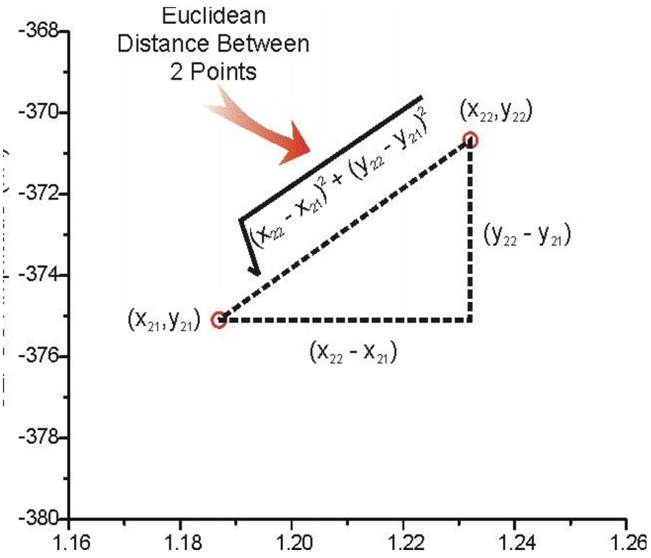
Vector MachinesBinary Classification: Multiclass/Structured outputs, MNIST, Ranking.

1. Distance based Methods

Distanced based algorithms are machine learning algorithms that classify queries by computing distances between these queries and a number of internally stored exemplars. Exemplars that are closets to the queryhave the largest influence on the classification assigned to the query.

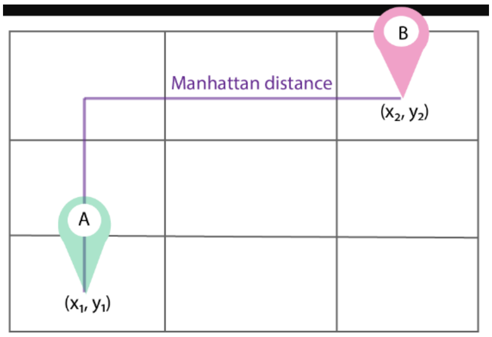
Euclidean Distance

It is the most common use of distance. In most cases when people said about distance, they will refer to Euclidean distance. Euclidean distance is also known as simply distance. When data is dense or continuous, this is the best proximity measure. The Euclidean distance between two points is the length of the path connecting them. The Pythagorean theory gives distance between two points.



Manhattan Distance

- If you want to find Manhattan distance between two different points (x1, y1) and (x2, y2) such as thefollowing, it would look like the following:
- Manhattan distance = (x2 – x1) + (y2 – y1)
- Diagrammatically, it would look like traversing the path from point A to point B while walking onthe pink straight line.



Minkowski Distance

The generalized form of the Euclidean and Manhattan Distances is the Minkowski Distance. You can expressthe Minkowski distance as

$$D(X,Y) = \left(\sum_{i=1}^{n} |x_i - y_i|^p\right)^{1/p}$$

The order of the norm is represented by p.

When an order(p) is 1, Manhattan Distance is represented, and when order(p) is 2 in the above formula,Euclidean Distance is represented.

2. Nearest Neighbors

The abbreviation **KNN** stands for "**K-Nearest Neighbor**". It is a supervised machine learning algorithm. Thealgorithm can be used to solve both classification and regression problem statements.

The number of nearest neighbors to a new unknown variable that has to be predicted or classified isdenoted by the symbol 'K'.

KNN calculates the distance from all points in the proximity of the unknown data and filters out the ones with the shortest distances to it. As a result, it's often referred to as a distance-based algorithm.

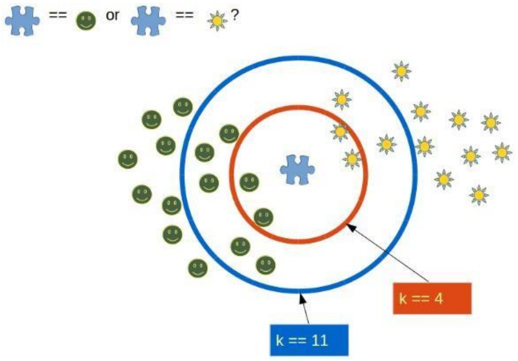
In order to correctly classify the results, we must first determine the value of K (Number of NearestNeighbors).

It is recommended to always select an odd value of K ~

When the value of K is set to even, a situation may arise in which the elements from both groups are equal.In the diagram below, elements from both groups are equal in the internal "Red" circle (k == 4).

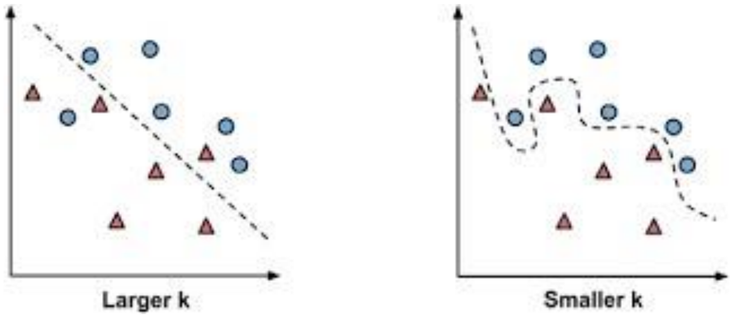
In this condition, the model would be unable to do the correct classification for you. Here the model willrandomly assign any of the two classes to this new unknown data.

Choosing an odd value for K is preferred because such a state of equality between the two classes would never occur here. Due to the fact that one of the two groups would still be in the majority, the value of K isselected as odd.



The impact of selecting a smaller or larger K value on the model

- Larger K value: The case of underfitting occurs when the value of k is increased. In this case, themodel would be unable to correctly learn on the training data.
- Smaller k value: The condition of overfitting occurs when the value of k is smaller. The model will capture all of the training data, including noise. The model will perform poorly for the test data inthis scenario.



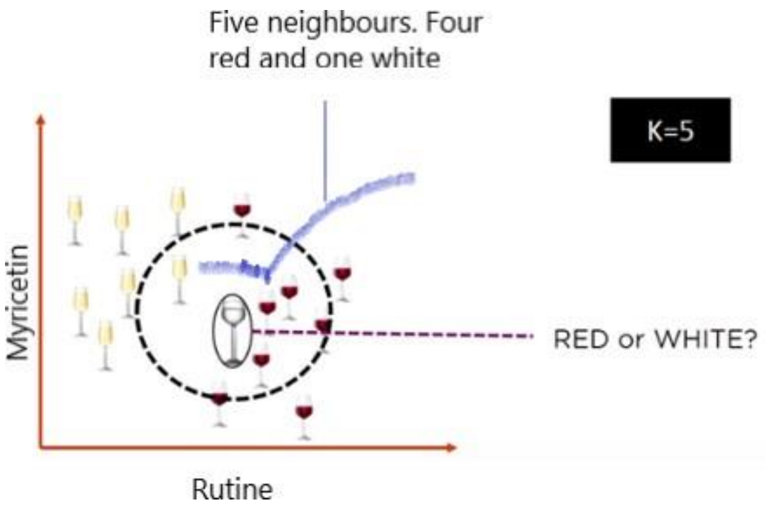
Src:
https://images.app.goo.gl/vXStNS4NeEqUC
DXn8

How does KNN work for ‘Classification’ and ‘Regression’ problem statements?

☞ **Classification**

When the problem statement is of ‘classification’ type, KNN tends to use the concept of “Majority Voting”.
Within the given range of K values, the class with the most votes is chosen.

Consider the following diagram, in which a circle is drawn within the radius of the five closest neighbours.Four of the five neighbours in this neighbourhood voted for ‘RED,’ while one voted for ‘WHITE.’ It will beclassified as a ‘RED’ wine based on the majority votes.



Real-world example:

Several parties compete in an election in a democratic country like India. Parties compete for voter supportduring election campaigns. The public votes for the candidate with whom they feel more connected.

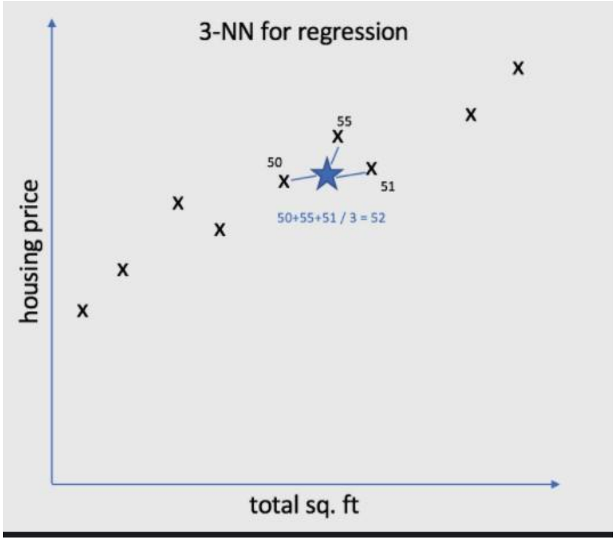
When the votes for all of the candidates have been recorded, the candidate with the most votes is declared as the election's winner.

- **Regression**

KNN employs a mean/average method for predicting the value of new data. Based on the value of K, itwould consider all of the nearest neighbours.

The algorithm attempts to calculate the mean for all the nearest neighbours' values until it has identified all
the nearest neighbours within a certain range of the K value.

Consider the diagram below, where the value of k is set to 3. It will now calculate the mean (52) based onthe values of these neighbours (50, 55, and 51) and allocate this value to the unknown data.



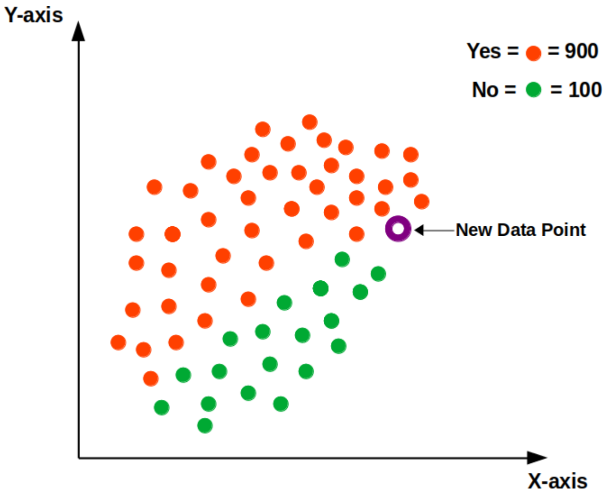
Impact of Imbalanced dataset and Outliers on KNN

- **Imbalanced dataset**

When dealing with an imbalanced data set, the model will become biased.

Consider the example shown in the diagram below, where the “Yes” class is more prominent.

As a consequence, the bulk of the closest neighbours to this new point will be from the dominant class.
Because of this, we must balance our data set using either an “Upscaling” or “Downscaling” strategy.

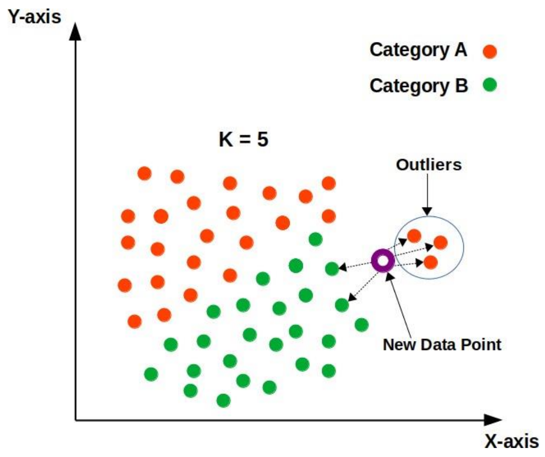


- **Outliers**

Outliers are the points that differ significantly from the rest of the data points.

The outliers will impact the classification/prediction of the model. The appropriate class for the new data point, according to the following diagram, should be “Category B” in green.

The model, however, would be unable to have the appropriate classification due to the existence of outliers.As a result, removing outliers before using KNN is recommended.



Importance of scaling down the numeric variables to the same level

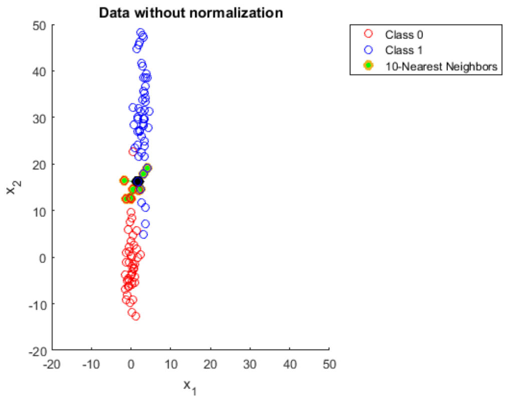
Data has 2 parts: –

1) Magnitude

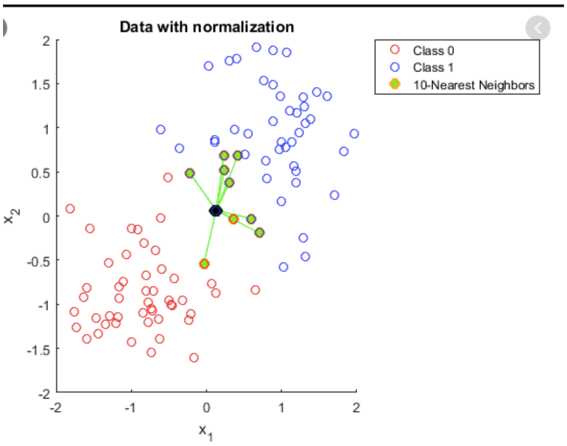
2) Unit

For instance; if we say 20 years then “20” is the magnitude here and “years” is its unit.

Since it is a distance-dependent algorithm, KNN selects the neighbours in the closest vicinity based solely onthe magnitude of the data. Have a look at the diagram below; the data is not scaled, so it can not find the closest neighbours correctly. As a consequence, the outcome will be influenced.



The data values in the previous figure have now been scaled down to the same level in the followingexample. Based on the scaled distance, all of the closest neighbours would be accurately identified.



3. Decision Trees

Like SVMs, Decision Trees are versatile Machine Learning algorithms that can perform both classification andregression tasks, and even multioutput tasks. They are very powerful algorithms, capable of fitting complex datasets.

Decision Trees are also the fundamental components of Random Forests, which are among the mostpowerful Machine Learning algorithms available today.

The concepts of Entropy and Information Gain in Decision Tree Learning

While constructing a decision tree, the very first question to be answered is, Which Attribute Is the BestClassifier?

The central choice in the ID3 algorithm is selecting which attribute to test at each node

in the tree.We would like to select the attribute that is most useful for classifying

examples.

What is a good quantitative measure of the worth of an attribute? We will define a statistical property, called*information gain*, that measures how well a given attribute separates the training examples according to their target classification.

ID3 uses this information gain measure to select among the candidate attributes at each step while growingthe tree.

ENTROPY MEASURES THE HOMOGENEITY OF EXAMPLES

Entropy characterizes the (im)purity of an arbitrary collection of examples.

Given a collection S, containing positive and negative examples of some target concept, the entropy of S relative to this boolean classification is

$$Entropy(S) \equiv -p_{\oplus} \log_2 p_{\oplus} - p_{\ominus} \log_2 p_{\ominus}$$

where p+, is the proportion of positive examples in S and p-, is the proportion of negative examples in S.In all calculations involving entropy, we define 0log0 to be .

Let, *S* is a collection of training

examples, *p*+ the proportion of

positive examples in *Sp*– the

proportion of negative examples in *S*

Examples

Entropy (*S*) = −*p*+ *log*2 *p*+ – *p*–*log*2 *p*– [0 *log*20 = 0]

Entropy ([14+, 0–]) = – 14/14 *log*2 (14/14) – 0 *log*2 (0) = 0

Entropy ([9+, 5–]) = – 9/14 *log*2 (9/14) – 5/14 *log*2 (5/14) = 0,94

Entropy ([7+, 7–]) = – 7/14 *log*2 (7/14) – 7/14 *log*2 (7/14)= 1/2 + 1/2 = 1

INFORMATION GAIN MEASURES THE EXPECTED

REDUCTION IN ENTROPY

Given entropy as a measure of the impurity in a collection of training examples, we can now define a measureof the effectiveness of an attribute in classifying the training data.

Now, the *information gain* is simply the expected reduction in entropy caused by partitioning the examplesaccording to this attribute.

More precisely, the information gain, *Gain(S, A)* of *an* attribute A, relative to a collection of examples *S,* isdefined as,

$$Gain(S, A) \equiv Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

where *Values(A)* is the set of all possible values for attribute A, and *S,* is the subset of S for which attribute Ahas value v (i.e., S_v= {s ∈ S|A(s) = v})

For example, suppose *S* is a collection of training-example days described by attributes including *Wind*, whichcan have the values *Weak* or *Strong.*

$$\begin{aligned}
Values(Wind) &= Weak, Strong \\
S &= [9+, 5-] \\
S_{Weak} &\leftarrow [6+, 2-] \\
S_{Strong} &\leftarrow [3+, 3-] \\
Gain(S, Wind) &= Entropy(S) - \sum_{v \in \{Weak, Strong\}} \frac{|S_v|}{|S|} Entropy(S_v) \\
&= Entropy(S) - (8/14) Entropy(S_{Weak}) \\
&\quad - (6/14) Entropy(S_{Strong}) \\
&= 0.940 - (8/14)0.811 - (6/14)1.00 \\
&= 0.048
\end{aligned}$$

Information gain is precisely the measure used by ID3 to select the best attribute at each step in growing the tree.

The use of information gain is to evaluate the relevance of attributes.

Q) Decision Tree construction using Entropy and information gain / Classification

- ID3 Algorithm:

Ross Quinlan developed ID# algorithm in 1987. It is greedy algorithm for decision tree construction. The ID3 algorithm is inspired from CLSs algorithm by Hunt in 1966. CLSs algorithm start with training object set, O={o1,o2,…on} from a universal where each object is described by a set of m attributes aj1,aj2,…ajk is selected and a tree node structure is formed to represent Aj.

In ID3 system, a relatively small number of training examples are randomly selected from a large set of objects O through a window. Using these a preliminary decision tree is constructed. The tree is then tested by scanning all the objects in O to see if there are any expectations to the tree. A new subset is formed using the original examples together with some of the expectations found during the scan. This process is repeated until no exceptions are found. The resulting decision tree is can be used to classify new objects.

- ID3 is the way in which the attributes are ordered for use in the classification process.
- Attributes which discriminate best are selected for the evaluation first.
- This requires computing an estimate of the expected information gain using all available

attributes andthen selecting the attribute having the largest expected gain.

- This attribute is assigned to root node.
- The attribute having the next largest gain is assigned to the next level of nodes in the tree and so onuntil the leaves of the tree have been reached.
- A decision tree is created by recursive selection of the best attribute in a top-down manner to use atthe current node in the tree.
- When a particular attribute is selected as the current node, it creates its child nodes, one for eachpossible value of the selected attribute.
- The next step is to partition the samples using the possible values of the attribute and to assign thesesubsets of examples to the appropriate child node.
- The process is repeated for every child node until we get a positive or negative result for all nodesassociated with the particular sample.

ID3 Algorithm

1. Establish a table R with classification attributes (aj1,aj2,…ajk).
2. Compute classification Shannon entropy by

$$H(x) = -\sum_{i=1}^{n} p(x_i) \log_b{}^{p(x_i)}$$

3. For each attribute in R, calculate information gain using classification attribute.

$$I_G(S, A) = I_E(S) - \Sigma^n (p(\in_S^{A_n}) \times I_E(\in_S^{A_n}))$$

$\in_S^A = \text{Subset } A \text{ of } S.$

4. Select attribute with the highest gain to be the next node in the tree.
5. Remove node attribute, creating reduced table Rs.
6. Repeat steps 3-5 until all the attributes have been used, or the same classification value remainsfor all rows in the reduced table.

Information gain for attribute A on set S is defined by taking the entropy of S and subtracting from it the summation of entropy of each subset of S, determined by the values of A multiplied by each subset's proportion of S.

Example:

Play Tennis Dataset

Day	Outlook	Temp	Humidity	Wind	Tennis?
D1	Sunny	Hot	High	Weak	NO
D2	Sunny	Hot	High	Strong	NO
D3	Cloudy	Hot	High	Weak	YES
D4	Rain	Mild	High	Weak	YES
D5	Rain	Cool	Normal	Weak	YES
D6	Rain	Cool	Normal	Strong	NO
D7	Cloudy	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	Cloudy	Mild	High	Strong	YES
D13	Cloudy	Hot	Normal	Weak	YES
D14	Rain	Mild	High	Strong	NO

→ Entropy $s[9,5] = -(9/14)\log_2(9/14) - (5/14)\log_2(5/14)$

$= 0.407 + 0.530$

$I(\overset{S_1}{Yes}, \overset{S_2}{no}) = 0.93$

→ <u>Entropy for outlook</u>

for outlook - sunny $<$ yes - 2 (s_{11})
NO 3 (s_{21})

$I(s_{11}, s_{21}) = -(2/5)\log_2(2/5) + (-3/5)\log_2(3/5)$

$= 0.5286 + 0.4414$

$= 0.970$

for outlook - cloudy $<$ yes - 4
NO 0

$I(s_{12}, s_{22}) = -(4/4)\log_2(4/4) + (-0/4)\log_2(0/4)$

$= 0$

for outlook - Rain $<$ yes - 3
no~~t~~ - 2

$I(s_{13}, s_{23}) = -(3/5)\log_2(3/5) + (-2/5)\log_2(2/5)$

$= 0.4414 + 0.5286$

$= 0.970$

$E(outlook) = ((5/14) * 0.970 + ((4/14) * 0) + ((5/14) * 0.970)$

$= 0.346 + 0 + 0.347$

$= 0.693$

$Infogain(outlook) = I(S_1, S_2) - E(outlook)$

$= 0932 - 0.693$

$= 0.238$

⇒ Entropy for temp

for temp = hot $<$ $\begin{matrix} \text{yes - 2} \\ \text{NO - 2} \end{matrix}$

$I(S_{11}, S_{21}) = -(2/4)\log 2\,(2/4) + (-2/4)\log 2\,(-2/4)$

$= 0.5 + 0.5$

$= 1$

for temp - mild $<$ $\begin{matrix} \text{yes - 4} \\ \text{NO - 2} \end{matrix}$

$I(S_{12}, S_{22}) = -(4/6)\log 2\,(4/6) + (-2/6)\log 2\,(-2/6)$

$= 0.350 + 0.4717$

$= 0.8217$

for temp - cool $<$ $\begin{matrix} \text{yes - 3} \\ \text{NO - 1} \end{matrix}$

$I(S_{13}, S_{23}) = (-3/4)\log 2\,(3/4) + (1/4)\log 2\,(-1/4)$

$= 0.302 + 0.4999$

$= 0.8019$

$E(\text{temp}) = ((4/14) * 1) + ((6/14) * 0.8217) + ((4/14) * 0.8019)$

$= 0.285 + 0.381 + 0.231$

$= 0.8977$

$\text{gain}(\text{temp}) = I(S_1, S_2) - E(\text{temp})$

$= 0.932 - 0.8977$

$= 0.0343$

→ Entropy for humidity

for humidity $<$ high - 7, Normal - 4

$I(s_{11}, s_{21}) = (-3/7)\log_2(-3/7) + (-4/7)\log_2(-4/7)$

$= 0.52 + 0.461$

$= 0.985$

$I(s_{12}, s_{22}) = (-6/7)\log_2(-6/7) + (-1/7)\log_2(-1/7)$

$= 0.1904 + 0.406$

$= 0.5916$

$E(\text{humidity}) = ((7/14) * 0.985) + ((7/14) * 0.5916)$

$= 0.78835$

$\text{Infogain}(\text{humidity}) = I(s_1, s_2) - E(\text{humidity})$

$= 0.932 - 0.78835$

$= 0.14365$

→ Entropy for wind

for wind $<$ Weak - 8, strong - 6

fair $I(s_{11}, s_{21}) = (-6/8)\log_2(-6/8) + (-2/8)\log_2(-2/8)$

$= 0.323 + 0.497$

$= 0.8080$

$I(s_{21}, s_{22}) = (-3/6)\log_2(-3/6) + (-3/6)\log_2(-3/6)$

$= 0.500 + 0.500$

$= 1$

$E(wind) = ((8/14)*0.8080) + ((6/14)*1)$

$= 0.4617 + 0.428$

$= 0.890$

$Infogain(wind) = I(s_1, s_2) - E(wind)$

$= 0.932 - 0.890$

$= 0.041$

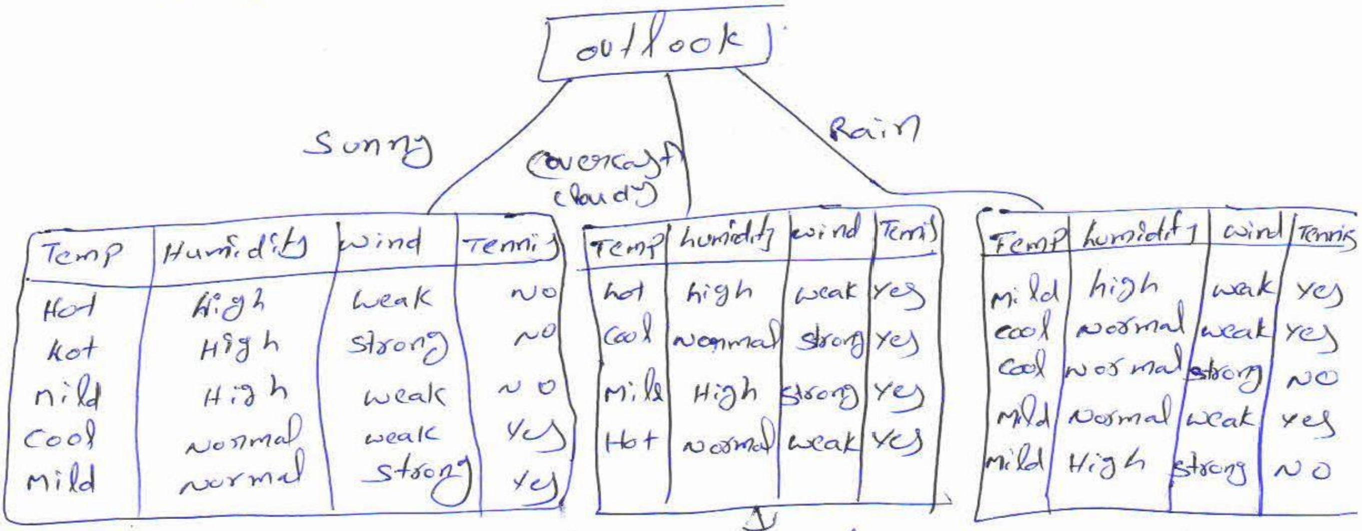
Finally infogain(S, outlook) = 0.238

Infogain(S, Temp) = 0.0343

Infogain(S, Humidity) = 0.143

Infogain(S, wind) = 0.041

Among all outlook is having highest infogain. So outlook is selected as best split. Acconding to that the decision tree is as follow.



Temp	Humidity	wind	Tennis
Hot	High	weak	NO
Hot	High	strong	NO
Mild	High	weak	NO
Cool	normal	weak	Yes
Mild	normal	strong	Yes

Temp	humidity	wind	Tennis
hot	high	weak	Yes
cool	normal	strong	Yes
Mild	High	strong	Yes
Hot	normal	weak	Yes

Temp	humidity	wind	Tennis
Mild	high	weak	yes
cool	normal	weak	yes
cool	normal	strong	NO
Mild	Normal	weak	yes
Mild	High	strong	NO

All class labels are same. so replace with leaf node for outlook sunny

$\text{Intogain}(s_1, s_2) = (-2/5)\log_2(-2/5) + (-3/5)\log_2(3/5)$

$= 0.974$

→ Entropy for temp $<$ hot - 2
mild - 2
cool - 1

for hot $<$ yes - 0
NO - 2

$I(S_{11}, S_{21}) = (-0/2)\log_2(0/2) + (-2/2)\log_2(2/2)$

$= 0$

for ~~ha~~ mild $<$ yes - 1
NO - 1

$I(S_{12}, S_{22}) = (-1/2)\log_2(-1/2) + (-1/2)\log_2(-1/2)$

$= 1$

for cool $<$ yes - 1
NO - 0

$I(S_{13}, S_{23}) = (-1/1)\log_2(-1/1) - (0/1)\log_2(-0/1)$

$= 0$

$E(\text{temp}) = (2/5 * 0) + (2/5 * 1) + (1/5 * 0)$

$= 0.4$

$\text{Infogain}(\text{temp}) = 0.974 - 0.4$

$= 0.574$

→ Entropy for humidity $<$ High - 3
normal - 2

for high $<$ yes 0
NO 3

$I(S_{11}, S_{12}) = (-0/3)\log_2(-0/3) + (-3/3)\log_2(-3/3)$

$= 0$

→ for normal < yes = 2
NO - 0

$I(S12, S22) = (-2/2) \log_2(-2/2) + (-0/2) \log_2 (0/2)$

$= 0$

E (humidity) = 0

infogain (humidity) = 0.974 - 0

= 0.974

→ for wind < weak - 3
strong - 2

for weak < yes -1
no - 2

$I(S11, S12) = (-1/3) \log_2(-1/3) + (-2/3) \log_2(-2/3)$

$= 0.525 + 0.39$

$= 0.91$

for strong < yes -1
no - 1

$I(S12, S22) = (-1/2) \log_2(-1/2) - (-1/2) \log_2 (-1/2)$

$= 1$

$E(\text{wind}) = ((3/5) * 0.918) + ((2/5) * 1)$

$= 0.946$

Infogain (wind) = 0.974 - 0.946

= 0.019

So finall for the left subtree the infogains are as follows

Infogain(S, temp) = 0.574
Infogin (S, humidity) = 0.974
Infogain(S, wind) = 0.019

Among all humidity is having highest Info gain.

→ for - outlook - rainy.

Infogain (s1, s2) = $-(3/5)\log_2(3/5) - (2/5)\log_2(2/5)$

= 0.974

entropy for temp < hot - 0
mild - 3
cool - 2

No. of tuples for hot are equal to zero

for mild < Yes - 2
no - 1

$I(s12, s22) = -(2/3)\log_2(2/3) - (1/3)\log_2(1/3)$

= 0.918

for cool < Yes - 1
NO - 1

$I(s13, s23) = -(1/2)\log_2(1/2) - (1/2)\log_2(1/2)$

= 1

E(temp) = $0 + ((3/5) * 0.918) + ((2/5) * 1)$

= 0.9508

Infogain (temp) = 0.974 - 0.9508

= 0.0232

Entropy for humidity < High - 2
normal - 3

for high < Yes 1
NO 1

$I(s11, s21) = -(1/2)\log_2(1/2) - (1/2)\log_2(1/2)$

= 1

for normal $\begin{cases} \text{yes} - 2 \\ \text{NO} - 1 \end{cases}$

$I(S12, S22) = -(2/3)\log 2(2/3) - (1/3)\log 2(1/3)$

$= 0.91$

$\text{Entropy (humidity)} = ((2/5)*1) + ((3/5)*0.91)$

$= 0.4 + 0.546$

$= 0.946$

$\text{Infogain (humidity)} = 0.974 - 0.946$

$= 0.028$

→ <u>Entropy for wind</u> $\begin{cases} \text{weak} - 3 \\ \text{strong} - 2 \end{cases}$

for weak $\begin{cases} \text{yes} - 3 \\ \text{no} - 0 \end{cases}$

$I(S11, S21) = -(3/3)\log 2(3/3) - (0/3)\log 2(0/3)$

$= 0$ for strong $\begin{cases} \text{yes} - 0 \\ \text{no} - 2 \end{cases}$

$I(S12, S22) = -(0/2)\log 2(0/2) - (2/2)\log 2(2/2)$

$= 0$

$\text{Entropy (wind)} = 0$

$\text{infogain(wind)} = 0.974 - 0$

$= 0.974$

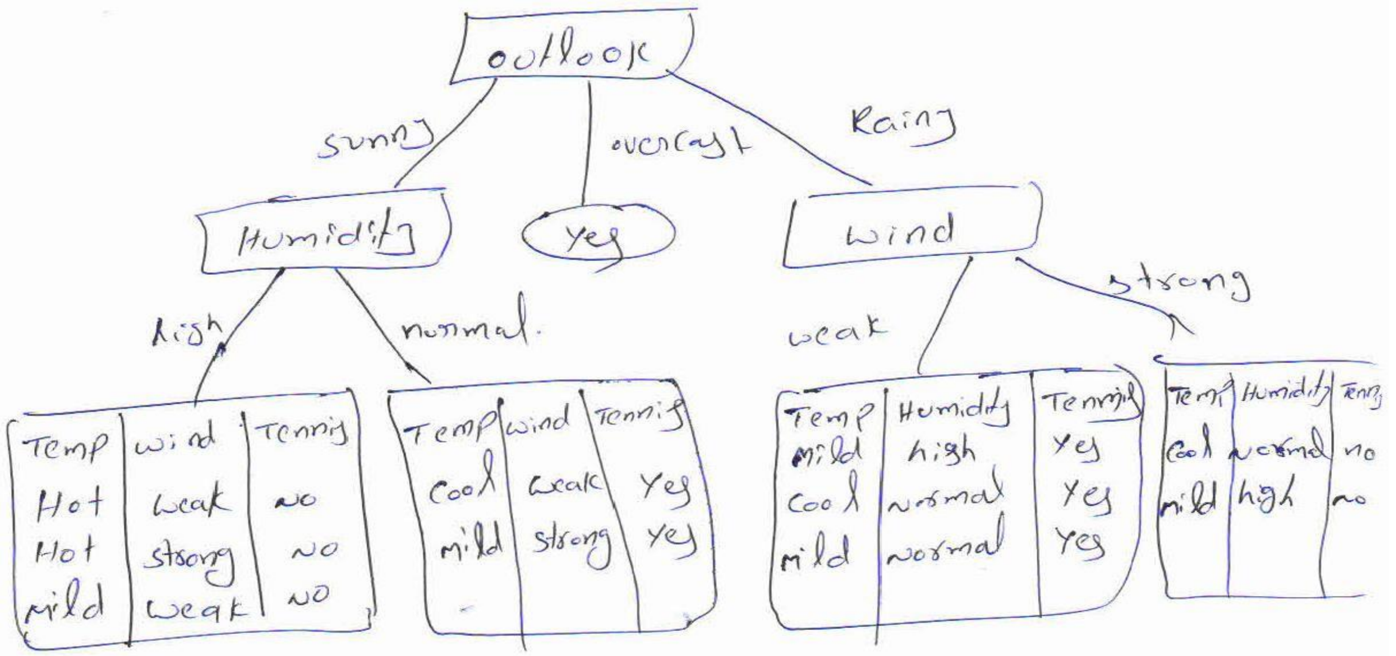
So finally for the right subtree the infogain values are as follows:

Infogain (S, temp) = 0.0232

Infogain(S, humidity) = 0.028

Infogain(S, wind) = 0.974

Among all wind is having highest infogain, so it willbe the root for right subtree. Finally the final decision tree is as follows!

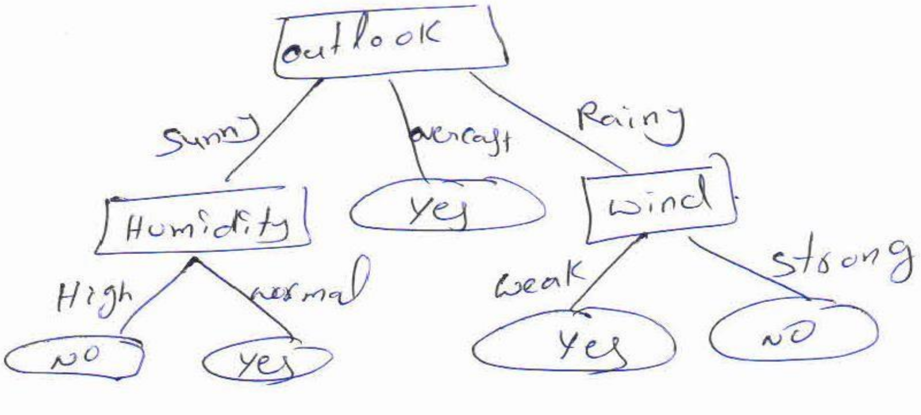


Temp	wind	Tennis
Hot	weak	no
Hot	strong	no
mild	weak	no

Temp	wind	Tennis
cool	weak	yes
mild	strong	yes

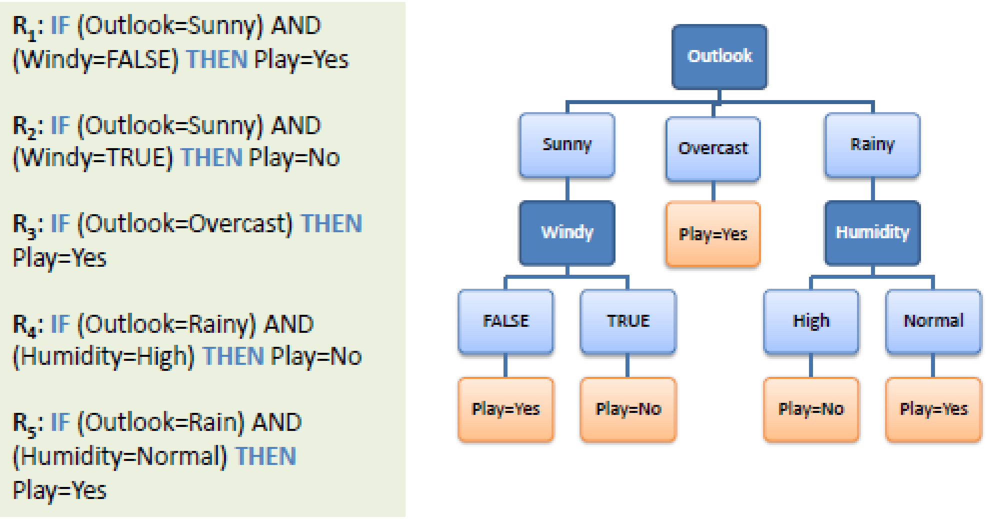
Temp	Humidity	Tennis
mild	high	yes
cool	normal	yes
mild	normal	yes

Temp	Humidity	Tennis
cool	normal	no
mild	high	no



Q) Decision Tree to Decision Rules/ Rule Extraction from Trees / Making Predictions

A decision tree can easily be transformed to a set of rules by mapping from the root node to the leaf nodesone by one.



R_1: IF (Outlook=Sunny) AND (Windy=FALSE) THEN Play=Yes

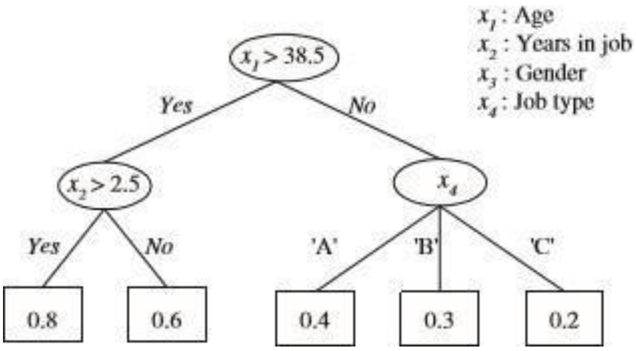
R_2: IF (Outlook=Sunny) AND (Windy=TRUE) THEN Play=No

R_3: IF (Outlook=Overcast) THEN Play=Yes

R_4: IF (Outlook=Rainy) AND (Humidity=High) THEN Play=No

R_5: IF (Outlook=Rain) AND (Humidity=Normal) THEN Play=Yes

A decision tree does its own feature extraction. The univariate tree only uses the necessary variables and after the tree is built, certain features may not be used at all. As shown in fig. x1,x2 and x4 variables are used but not x3. It is possible to use a decision tree for feature extraction: we build a tree and then take only those features used by the tree as inputs to another learning method.



Main advantage of decision tree is interpretability: the decision nodes carry conditions that are simple to understand. Each path from the root to a leaf corresponds to one conjunction of tests as all these conditions must me satisfied to reach leaf node. These paths together can be written down as asset of IF-THEN rules,called a rule base.

For example, the decision tree of fig can be written down as the following

set of rules:R1: IF (age>38.5) AND (years-in-job>2.5) THEN y=0.8

R2: IF (age>38.5) AND (years-in-job≤ 2.5)

THEN y=0.6R3: IF (age≤ 38.5) AND (job-type

=’A’) THEN y=0.4 R4: IF (age≤38.5) AND

(job-type =’B’) THEN y=0.3 R5: IF (age≤38.5)

AND (job-type =’C’) THEN y=0.2

Such a rule base allows knowledge extractor, it can be easily understood and allows experts to verify the model learned from data. For each rule, one can also calculate the percentage of training data covered by the rule namely rule support. The rules reflect the main characteristics of the dataset. They show the important features and split positions. For instance, in this example, we see that in terms of our purpose (y),

- people who are thirty –eight years old or less are different from people who are thirty-nine or moreyears old.
- And in the latter group, it is the job type that makes them different.
- In the former group no of years in job is the best discriminating characteristic.

In case of classification tree, there may be more than one leaf labelled with the same class. In such a case, these multiple conjunctive expressions corresponding to different paths can be combined as a disjunction. The class region then corresponds to a union of these multiple patches, each patch corresponding to the region definedby one leaf. For example

IF(x≤w10)OR((x1>w10) AND (x2≤w20)) Then C1

Pruning rules is possible for simplification. Pruning a subtree corresponds to pruning terms from a number of rules at the same time. It may be possible to prune a term from one rule without touching other examples. For example, in the previous rule set, for R3, if we see that all whose job-type =’A’ have outcomes close to 0.4, regardless of age, R3 can be pruned as

R3`: IF (job-type =’A’) THEN y=0.4

Once the rules are pruned we cannot write them back as a tree anymore.

Q) The CART Training Algorithm

Scikit-Learn uses the *Classification And Regression Tree* (CART) algorithm to train Decision Trees (also called “growing” trees). The idea is really quite simple: the algorithm first splits the training set in two subsets using a single feature *k* and a threshold *tk*

CART algorithm uses Gini Impurity to split the dataset into a decision tree .It does that by searching for thebest homogeneity for the sub nodes, with the help of the Gini index criterion.

Gini index/Gini impurity: The Gini index is a metric for the classification tasks in CART. It stores the sum of squared probabilities of each class. It computes the degree of probability of a specific variable that is wrongly being classified when chosen randomly and a variation of the Gini coefficient. It works on categorical variables, provides outcomes either “successful” or “failure” and hence conducts binary splitting only.

The degree of the Gini index varies from 0 to 1,

- Where 0 depicts that all the elements are allied to a certain class, or only one class exists there.
- The Gini index of value 1 signifies that all the elements are randomly distributed across various classes,and
- A value of 0.5 denotes the elements are uniformly distributed into

$$Gini = 1 - \sum_{i=1}^{n}(pi)^2$$

some classes.Mathematically, we can write Gini Impurity as follows:

where pi is the probability of an object being classified to a

particular class.Classification tree

A classification tree is an algorithm where the target variable is categorical. The algorithm is then used to identify the “Class” within which the target variable is most likely to fall. Classification trees are used when the dataset needs to be split into classes that belong to the response variable(like yes or no)

Regression tree

A Regression tree is an algorithm where the target variable is continuous and the tree is used to predict its value. Regression trees are used when the response variable is continuous. For example, if the response variable is the temperature of the day.

Advantages of CART

- Results are simplistic.
- Classification and regression trees are Nonparametric and Nonlinear.
- Classification and regression trees implicitly perform feature selection.

- Outliers have no meaningful effect on CART.
- It requires minimal supervision and produces easy-to-

understand models.Limitations of CART

- Overfitting.
- High Variance.
- low bias.
- the tree structure may be unstable.

Applications of the CART algorithm

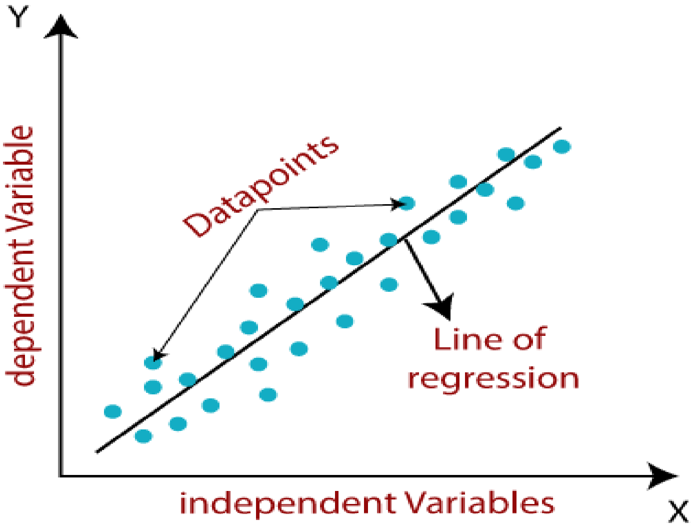
- For quick Data insights.
- In Blood Donors Classification.
- For environmental and ecological data.
- In the financial sectors.

Linear Models: Linear Regression, Logistic Regression, Generalized Linear Models, Support Vector Machines

Linear regression is one of the easiest and most popular Machine Learning algorithms. It is a statistical method that is used for predictive analysis. Linear regression makes predictions for continuous/real or numeric variables such as sales, salary, age, product price, etc.

Linear regression algorithm shows a linear relationship between a dependent (y) and one or more independent
(y) variables, hence called as linear regression. Since linear regression shows the linear relationship, which means it finds how the value of the dependent variable is changing according to the value of the independentvariable.

The linear regression model provides a sloped straight line representing the relationship between the variables. Consider the below image:



Mathematically, we can represent a linear regression as:

y= a0+a1x+ ε

Here,

Y= Dependent Variable (Target Variable)

X= Independent Variable (predictor Variable)

a0= intercept of the line (Gives an additional degree of

freedom) a1 = Linear regression coefficient (scale factor to

each input value).ε = random error

The values for x and y variables are training datasets for Linear Regression model

representation.Types of Linear Regression

Linear regression can be further divided into two types of the algorithm:

- **Simple Linear Regression:**

  If a single independent variable is used to predict the value of a numerical dependent variable, thensuch a Linear Regression algorithm is called Simple Linear Regression.

- **Multiple Linear regression:**

  If more than one independent variable is used to predict the value of a numerical dependentvariable, then such a Linear Regression algorithm is called Multiple Linear Regression.

1. Simple linear regression

Simple Linear Regression is a type of Regression algorithms that models the relationship between a dependent variable and a single independent variable. The relationship shown by a Simple Linear Regression model is linear or a sloped straight line, hence it is called Simple Linear Regression.

The key point in Simple Linear Regression is that the *dependent variable must be a continuous/real value.*However, the independent variable can be measured on continuous or categorical values.

Simple Linear regression algorithm has mainly two objectives:

- Model the relationship between the two variables. Such as the relationship between Income andexpenditure, experience and Salary, etc.
- Forecasting new observations. Such as Weather forecasting according to temperature, Revenue of acompany according to the investments in a year, etc.

Simple Linear Regression Model:

The Simple Linear Regression model can be represented using the below equation:

y= a0+a1x+ ε

Where,

a0= It is the intercept of the Regression line (can be obtained putting x=0)
a1= It is the slope of the regression line, which tells whether the line is increasing or decreasing.
ε = The error term. (For a good model it will be negligible)

Implementation of Simple Linear Regression Algorithm using

PythonProblem Statement example for Simple Linear

Regression:

Here we are taking a dataset that has two variables: salary (dependent variable) and experience (Independent variable). The goals of this problem is:

- We want to find out if there is any correlation between these two variables
- We will find the best fit line for the dataset.
- How the dependent variable is changing by changing the independent variable.

Here, we will create a Simple Linear Regression model to find out the best fitting line for representing therelationship between these two variables.

2. Multiple linear regression

Multiple Linear Regression is one of the important regression algorithms which models the linear relationship between a single dependent continuous variable and more than one independent variable.

Example:

Prediction of CO2 emission based on engine size and number of cylinders

in a car.Some key points about MLR:

- For MLR, the dependent or target variable(Y) must be the continuous/real, but the predictor orindependent variable may be of continuous or categorical form.
- Each feature variable must model the linear relationship with the dependent variable.
- MLR tries to fit a regression line through a multidimensional space of data-points.

The multiple regression equation explained above takes the following form:

y = b1x1 + b2x2 + … +

bnxn + c.Where,

Y= Output/Response variable

b0, b1, b2, b3 , bn= Coefficients of the model.

x1, x2, x3, x4,. = Various Independent/feature variable

Binary Classification: Multiclass/Structured outputs, MNIST, Ranking.

…………………………………………………………………………………………………
……………………………………………..

Binary Classification

It is a process or task of classification, in which a given data is being classified into two classes. It's basically a
kind of prediction about which of two groups the thing belongs to.

Let us suppose, two emails are sent to you, one is sent by an insurance company that keeps sending their ads,and the other is from your bank regarding your credit card bill. The email service provider will classify the two emails, the first one will be sent to the spam folder and the second one will be kept in the primary one.

This process is known as binary classification, as there are two discrete classes, one is spam and the other isprimary. So, this is a problem of binary classification.

Binary classification uses some algorithms to do the task, some of the most common algorithms used bybinary classification are .

- Logistic Regression
- k-Nearest Neighbors
- Decision Trees
- Support Vector Machine
- Naive Bayes

Multiclass

Classification

Multi-class classification is the task of classifying elements into different classes. Unlike binary, it doesn't restrict itself to any number of classes. Examples of multi-class classification are

- classification of news in different categories,
- classifying books according to the subject,
- classifying students according to their streams etc.

In these, there are different classes for the response variable to be classified in and thus according to thename, it is a Multi-class classification.

Can a classification possess both binary or multi-class?

Let us suppose we have to do sentiment analysis of a person, if the classes are just "positive" and "negative",then it will be a problem of binary class. But if the classes are "sadness", happiness", "disgusting", "depressed", then it will be called a problem of Multi-class classification.

Binary vs Multiclass Classification

Parameters	Binary classification	Multi-class classification
No. of classes	It is a classification of two groups, i.e.classifies objects in at most two classes.	There can be any number of classes in it, i.e., classifies the object into more than two classes.
Algoritms used	The most popular algorithms used by thebinary classification are- • Logistic Regression • k-Nearest Neighbors • Decision Trees • Support Vector Machine • Naive Bayes	Popular algorithms that can be used for multi-classclassification include: • k-Nearest Neighbors • Decision Trees • Naive Bayes • Random Forest. • Gradient Boosting
Examples	Examples of binary classification include- • Email spam detection (spam ornot). • Churn prediction (churn or not). • Conversion prediction (buy ornot).	Examples of multi-class classification include: • Face classification. • Plant species classification. • Optical character recognition.

Q) MNIST

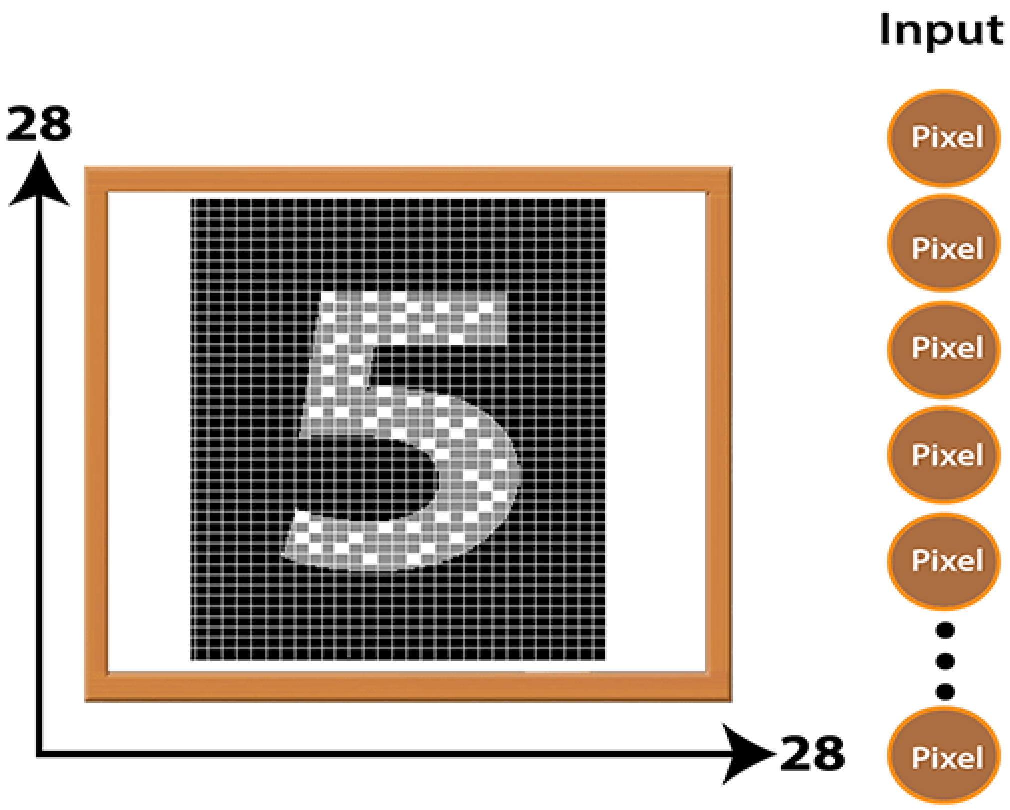
The MNIST database (Modified National Institute of Standards and Technology database) is a simple computer vision dataset. It consists of 28x28 pixel images of handwritten digits, such as:

When we look at the image, our brain and eyes work together to recognize this image as number eight. Our brain is a very powerful tool, and it's capable of categorizing this image as an eight very quickly. There are so many shapes of a number, and our mind can easily recognize these shapes and determine what number is it, but this task is not so simple for a computer to complete. There is only one way to do this, which is the use of deep neural network which allows us to train a computer to classify the handwritten digits effectively.

In MNIST dataset, a single data point comes in the form of an image. These images, contained in MNIST datasets, are typically 28*28 pixels such that 28 pixels traversing the horizontal axis and 28 pixels traversing the vertical axis. It means that a single image from the MNIST database has a total of 784 pixels which must be analyzed. There are 784 nodes in the input layer of our neural network to analyze one of these images.

The MNIST database contains 60,000 training images and 10,000 testing images. Half of the training set andhalf of the test set were taken from NIST's training dataset, while the other half of the training set and theother half of the test set were taken from NIST's testing dataset.

The MNIST dataset is a multiclass dataset which consists of 10 classes into which we can classify numbers from 0 to 9. The major difference between the datasets which we have used previously and the MNIST dataset is the method in which the MNIST data is inputted into the neural network.



Due to the additional input nodes and increased no of the classes that the numbers can be classified in 0 to 9. It is clear that our dataset is more complex than any of the datasets we analyze before. For classifying this dataset, a deep neural network is required with the effectiveness of some hidden layers.

In our deep neural network, there are 784 nodes in the input layer, a few of hidden layers which feed-forward the input values and finally ten nodes in output layer for each of the respective handwritten numbers. The values are fed through the network, and the node which outputs the highest activation value in the output layer identifies the letter or number.

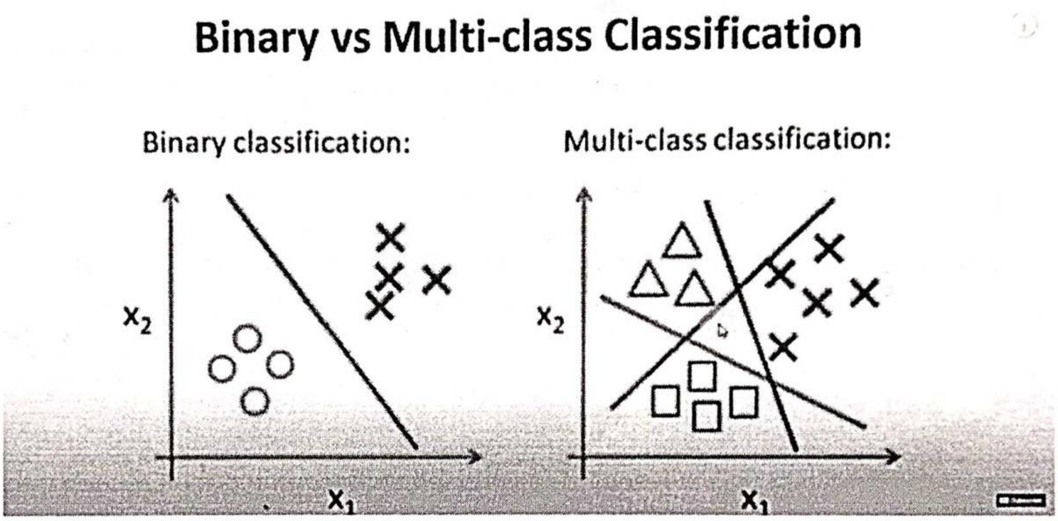
Extended MNIST (EMNIST) is a newer dataset developed and released by NIST to be the (final) successor toMNIST.MNIST included images only of handwritten digits. EMNIST includes all the images from NIST Special Database 19, which is a large database of handwritten uppercase, and lower case letters as well as digits. The images in EMNIST were converted into the same 28x28 pixel format, by the same process, as were the MNIST images. Accordingly, tools which work with the older, smaller, MNIST dataset will likely work unmodified with EMNIST.

Q) Ranking

A binary classification system involves a system that generates ratings for each occurrence, which, by ordering them, are turned into rankings, which are then compared to a threshold. Occurrences with rankings above the threshold are declared positive, and occurrences below the threshold are declared negative.

Multi Class Classification:

- When we solve a classification problem having **only two class labels**, then it becomes easy for us to filter the data, apply any classification algorithm, train the model with filtered data, and predict the outcomes.
- But when we **have more than two class instances** in input train data, then it might get complex to analyze the data, train the model and predict relatively accurate results.
- **To handle these multiple class instances, we use multi class classification.**
- Multi-class classification is the classification technique that allows us to categorize test data into multiple class labels present in trained data as a model prediction.


Binary vs Multi-class Classification
Binary classification:
Multi-class classification:
x2
x1
x2
x1

Binary Classification:

- Only two class instances are present in the dataset.
- It requires only one classifier model.
- Confusion Matrix is easy to derive and understand.
- Example:-check email is spam or not, predicting gender based on height and weight.

Multi class classification:

Multiple class labels are present in the dataset.

The number of classifier models depends on the classification technique we are applying to.

One vs. All : N class instances then N binary classier models.

One vs. One : N class instances the N*(N-1)/2 binary classifier models .

Example: Check whether the fruit is apple ,banana or orange.

1.One vs. All(One vs Rest)

In one vs All classification , for the N-class instances dataset ,we must generate the N-binary classifier models.

The no.of class labels preent in the dataset and the number of generated binary classifiers must be same.

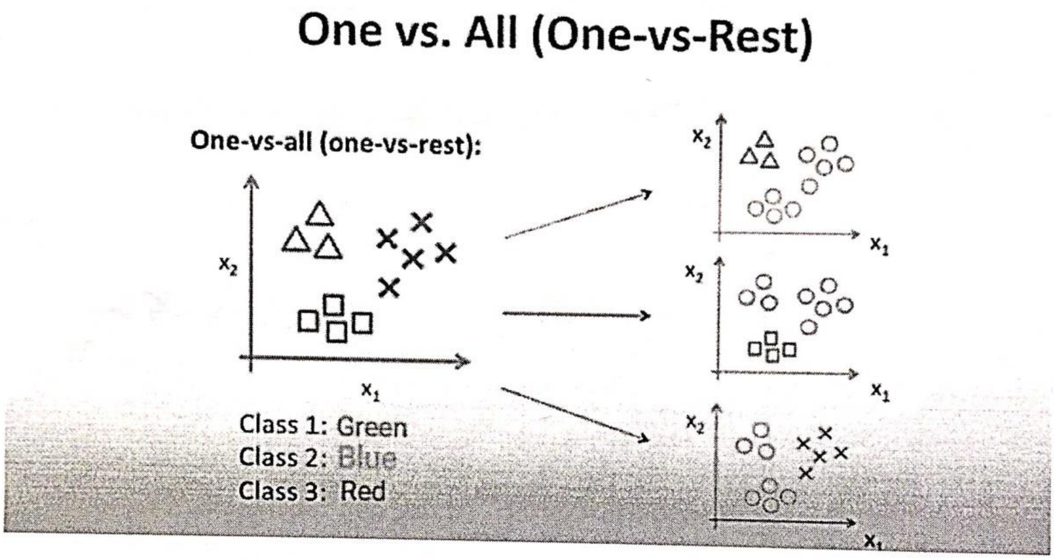
Consider we have three examples ,for example Green,Blue and Red

Now we create three classifiers here for three respective classes.

Classifier 1: [Green] vs [Red,Blue]

Classifier 2:[Blue] vs [Green,Red]

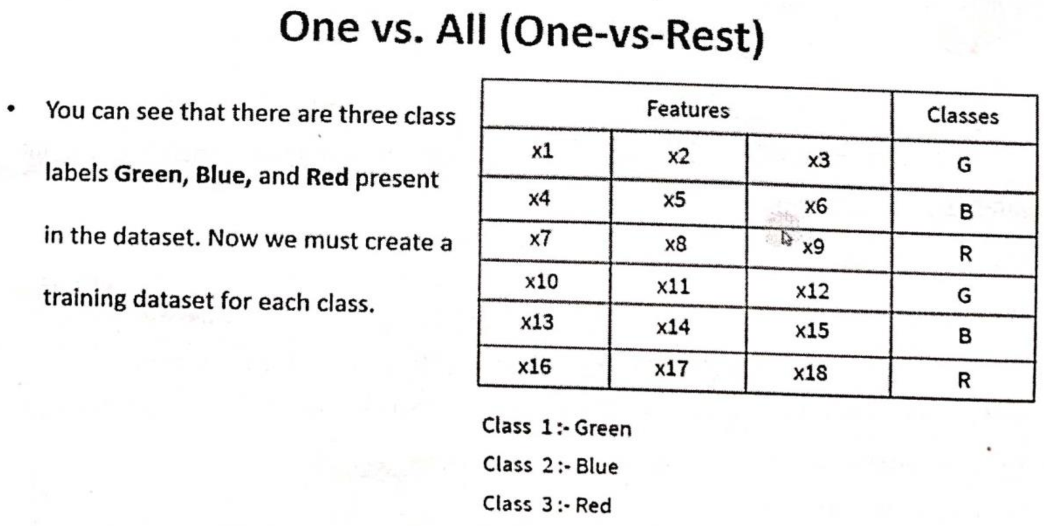
Classifier 3:[Red] vs [Blue,Green]



One vs. All (One-vs-Rest)

One-vs-all (one-vs-rest):

Class 1: Green
Class 2: Blue
Class 3: Red

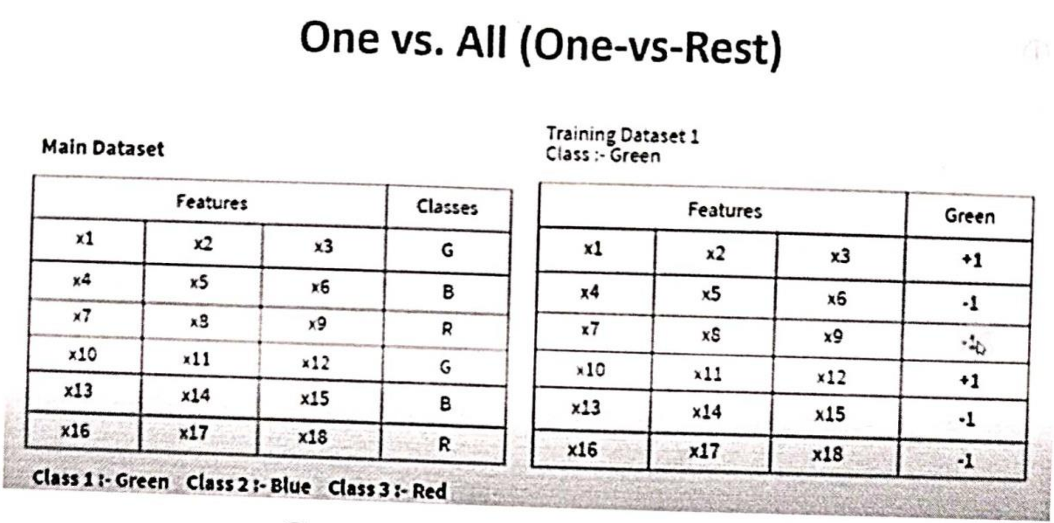


One vs. All (One-vs-Rest)

- You can see that there are three class labels **Green, Blue,** and **Red** present in the dataset. Now we must create a training dataset for each class.

Features			Classes
x1	x2	x3	G
x4	x5	x6	B
x7	x8	x9	R
x10	x11	x12	G
x13	x14	x15	B
x16	x17	x18	R

Class 1 :- Green
Class 2 :- Blue
Class 3 :- Red



One vs. All (One-vs-Rest)

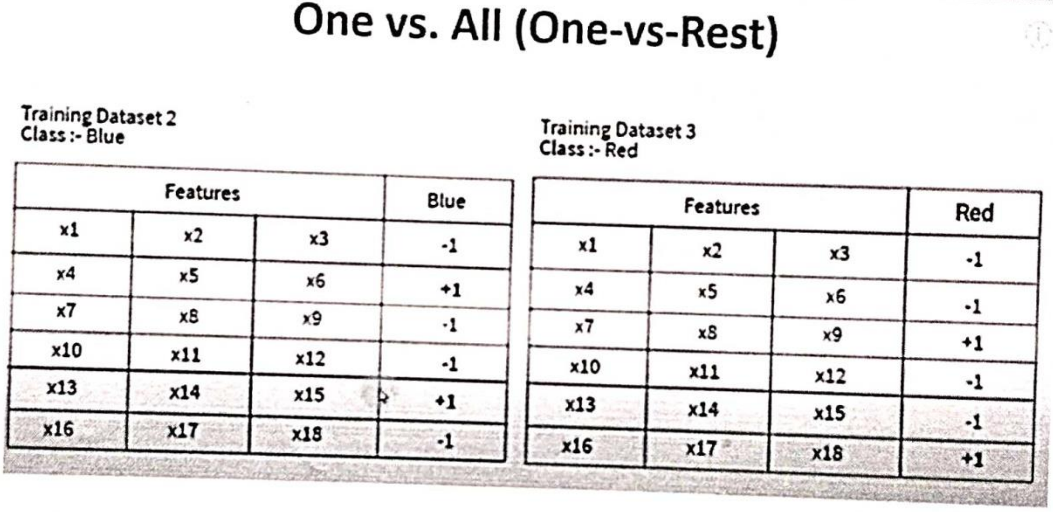
Main Dataset

Features			Classes
x1	x2	x3	G
x4	x5	x6	B
x7	x8	x9	R
x10	x11	x12	G
x13	x14	x15	B
x16	x17	x18	R

Class 1 :- Green Class 2 :- Blue Class 3 :- Red

Training Dataset 1
Class :- Green

Features			Green
x1	x2	x3	+1
x4	x5	x6	-1
x7	x8	x9	-1
x10	x11	x12	+1
x13	x14	x15	-1
x16	x17	x18	-1



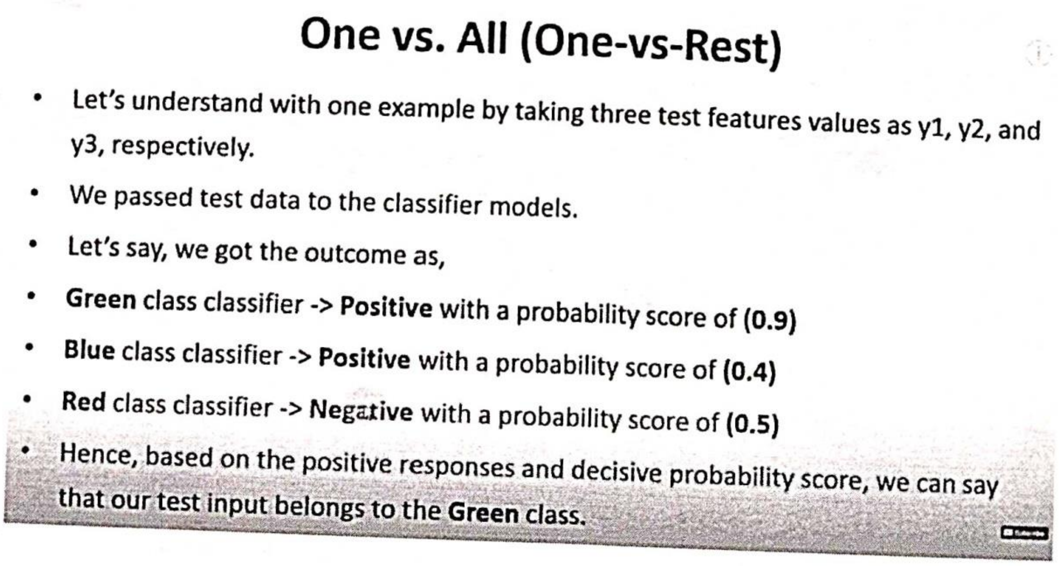
One vs. All (One-vs-Rest)

Training Dataset 2
Class :- Blue

Features			Blue
x1	x2	x3	-1
x4	x5	x6	+1
x7	x8	x9	-1
x10	x11	x12	-1
x13	x14	x15	+1
x16	x17	x18	-1

Training Dataset 3
Class :- Red

Features			Red
x1	x2	x3	-1
x4	x5	x6	-1
x7	x8	x9	+1
x10	x11	x12	-1
x13	x14	x15	-1
x16	x17	x18	+1



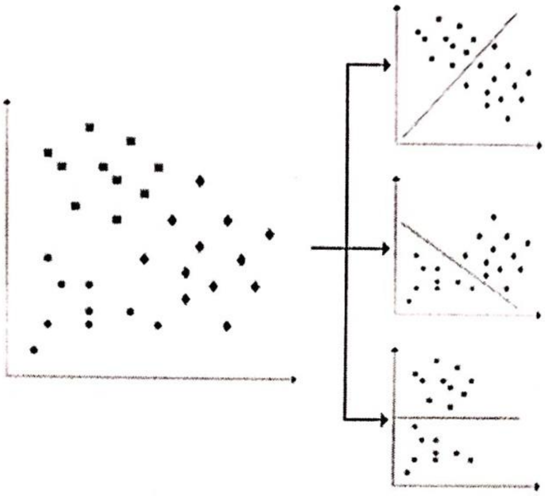
One vs. All (One-vs-Rest)

- Let's understand with one example by taking three test features values as y1, y2, and y3, respectively.
- We passed test data to the classifier models.
- Let's say, we got the outcome as,
- **Green** class classifier -> **Positive** with a probability score of **(0.9)**
- **Blue** class classifier -> **Positive** with a probability score of **(0.4)**
- **Red** class classifier -> **Negative** with a probability score of **(0.5)**
- Hence, based on the positive responses and decisive probability score, we can say that our test input belongs to the **Green** class.

2.One vs. One:

One vs. One (OvO)

- In One-vs-One classification, for the **N-class** instances dataset, we must generate the **N* (N-1)/2** binary classifier models.
- Using this classification approach, we split the primary dataset into one dataset for each class opposite to every other class.
- Taking the above example, we have a classification problem having three types: **Green, Blue,** and **Red (N=3)**



One vs. One (OvO)

- We divide this problem into **N* (N-1)/2 = 3** binary classifier problems:
  - Classifier 1: Green vs. Blue
  - Classifier 2: Green vs. Red
  - Classifier 3: Blue vs. Red
- Each binary classifier predicts one class label.
- When we input the test data to the classifier, then the model with the majority counts is concluded as a result.

SVM Regression

The SVM algorithm is quite versatile: not only does it support linear and nonlinear classification, but it also supports linear and nonlinear regression. The trick is to reverse the objective: instead of trying to fit the largest possible street between two classes while limiting margin violations, SVM Regression tries to fit as many instances as possible *on* the street while limiting margin violations(i.e., instances *off* the street).The width of the street is controlled by a hyper parameter ϵ.
Some of the key parameters used are as mentioned below:

Hyperplane:

Hyperplanes are decision boundaries that is used to predict the continuous output. The datapoints on either side of the hyperplane that are closest to the hyperplane are called SupportVectors.These are used to plot the required line that shows the predicted output of the algorithm.

Kernel:

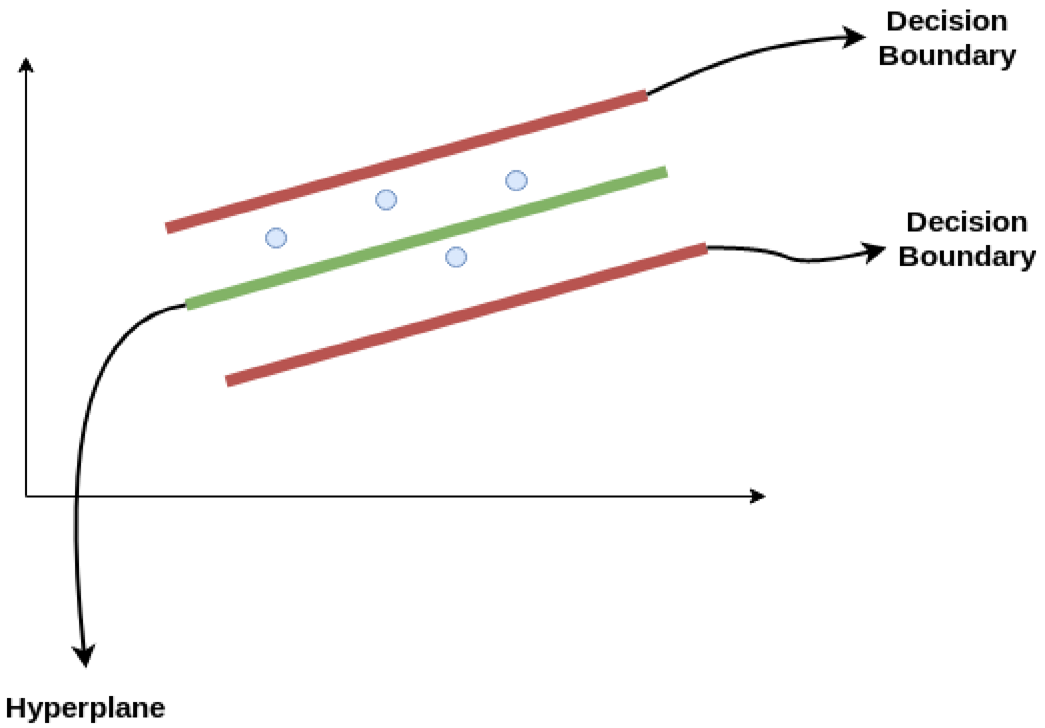
A kernel is a set of mathematical functions that takes data as input and transform it into the required form. These are generally used for finding a hyperplane in the higher dimensional space. The most widely used kernels include Linear, Non-Linear, Polynomial, Radial Basis Function (RBF) and Sigmoid. By default, RBF is used as the kernel. Each of these kernels are used depending on the dataset.

Boundary Lines:

These are the two lines that are drawn around the hyperplane at a distance of **ε**(epsilon).It is used to create a margin between the data points.

The Idea Behind Support Vector Regression

The problem of regression is to find a function that approximates mapping from an input domain to real numbers on the basis of a training sample. So let’s now dive deep and understand how SVR works actually.



Consider these two red lines as the decision boundary and the green line as the hyperplane. Our objective, when we are moving on with SVR, is to basically consider the points that are within the decision boundary line. Our best fit line is the hyperplane that has a maximum number of points.

The first thing that we'll understand is what is the decision boundary (the dangerred line above!).

Consider these lines as being at any distance, say 'a', from the hyperplane.So, these are the lines that we draw at distance '+a' and '-a' from the hyperplane. This 'a' in the text is basically referred to as epsilon.

Assuming that the equation of the hyperplane is as follows:

Y= wx+b (equationof hyperplane)

Then the equations of decision boundary become:

wx+b=+a wx+b= -a

Thus,any hyperplane that satisfies our SVR should satisfy:

-a<Y-wx+b <+a

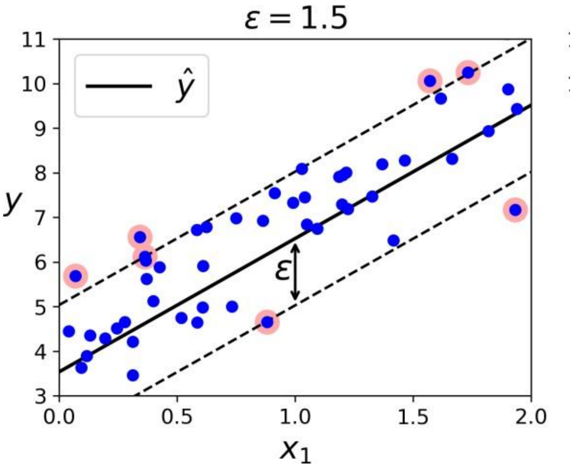
Our main aim here is to decide a decision boundary at 'a' distance from the original hyperplane such that data points closest to the hyperplane or the support vectors are within that boundary line.

Hence,wearegoingtotakeonlythosepointsthatarewithinthedecisionboundaryandhavetheleasterror rate, or are within the Margin of Tolerance. This gives us a better fitting model.

You can use Scikit-Learn’s Linear SVR class to perform linear SVMRegression

```
from sklearn.svm import LinearSVR
svm_reg=LinearSVR(epsilon=1.5) svm_reg.fit(X, y)
```

Output:


$\varepsilon = 1.5$
$\hat{y}$
ε
y
x_1