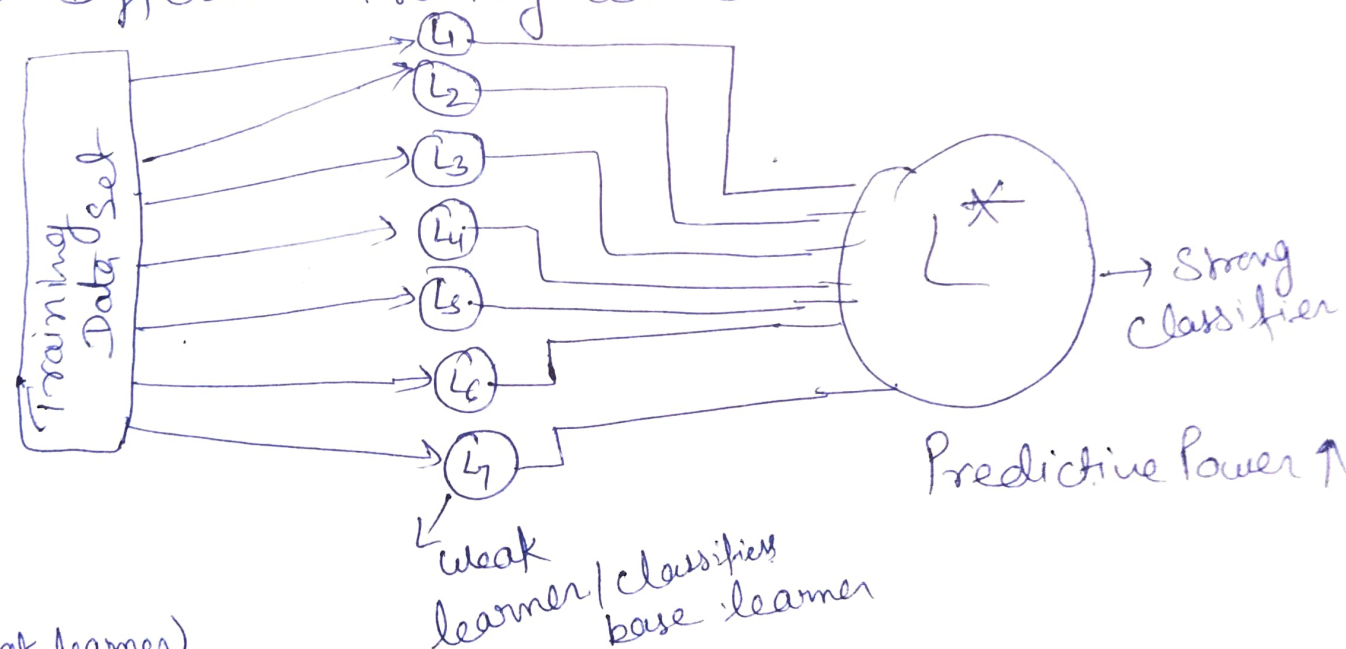
Ensemble learning →

In Ensemble learning we will Combine the output of multiple classifier in order to get better Prediction and or classification accuracy.

And under Conditions, Where the classifier output are independent of Each other and make Errors in an independent Manner, it is possible that by Combining the output of Several classifier. we get a resulting classifier which is better than any of the Constituant classifier.

→ Different algorithm

→ Different Training data set



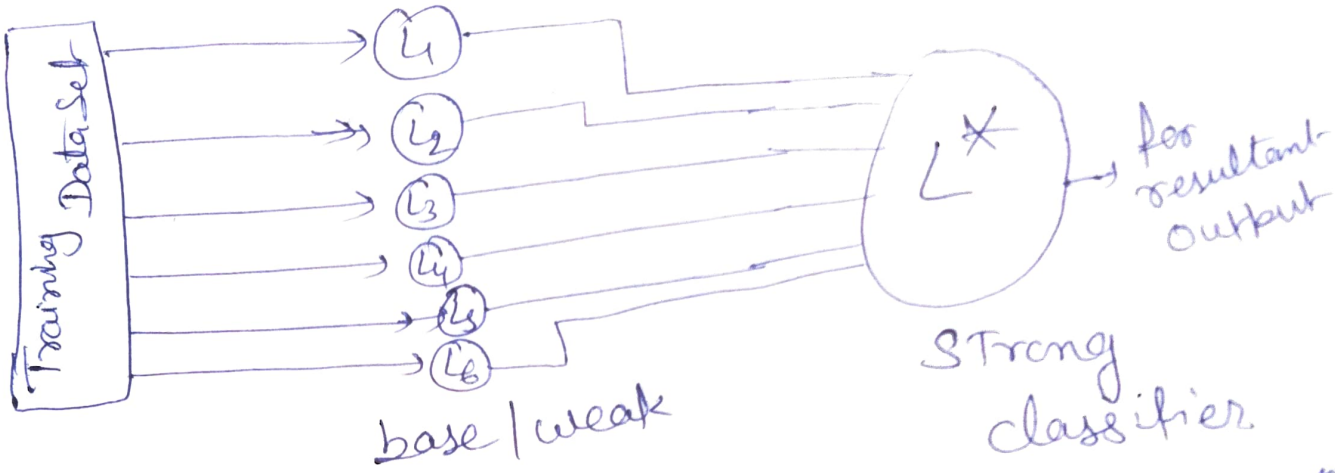
(weak learner)

they Can use different algorithm to maintain the diversity of their output.

this is called Heterogeneous Ensemble because all base learnes are using different - diffenent algorithm.

if all base learners use same algorithm
diversity of output will be low.

To get diversity we can use different data set for each base learner



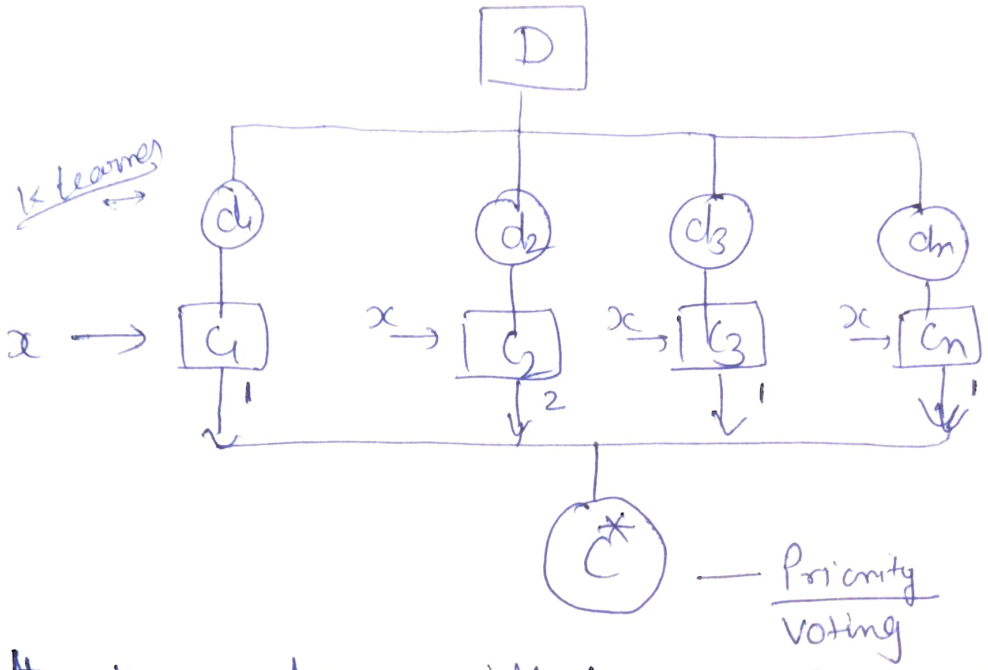
Predictive Power ↑
Accracy ↑
Precision ↑
Error Rate ↓

Two Different Ensemble learning Methods

1) Bagging 2) Boosting.

Bagging → It is also known as Bootstrap aggregation.

· first of all the samples are generated such that the samples are different from Eachother we sample D_1 D_2 D_3 -- D_k these are different Data Subset which is sampled from Original Data. They may have same in Common



but they also will have different values.

To get di Draw Random sample with replacement..

for Example

if we desire each of these Di we will have m Examples we randomly draw m samples from D with replacement ~~me~~ that is we can have the same instance repeated several times and some instances may not appear in a particular Di and so these Di can be different.

we use bagging technique for these k learners and train a base learner with each and combine the output by voting.

Boosting → When we try to get D_i we have to sample from D. But In this we assign a Probability with Each instance. So Every instance is assigned a Probability.

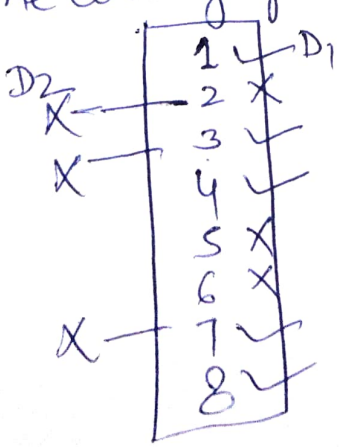
Suppose, to start with there are n instances and initially we may have ~~are~~ may give Equal Probability to all the instances

So; ~~Each of none of them has~~

In the first iteration, we choose D_1 based on this uniform Probability distribution; and ~~we~~ get D_1 by Sampling. Now on D_i we train a Model (M_1) for classification. and this Model we apply to all these instances.

Now this Model may have 100% accuracy on these instances. that is the case then it is a very gread Model & we no need to really Proceed. So it will have When all of them it will do it Correctly.

But In Most of time this Model will label Some of these instances Correctly and Some wrongly in D.



D_1 labels Some Correctly and Some wrongly.

Now we want the next Model to have independent Errors. and we want the next Model to do well on those instances where the Current Model has Not done well.

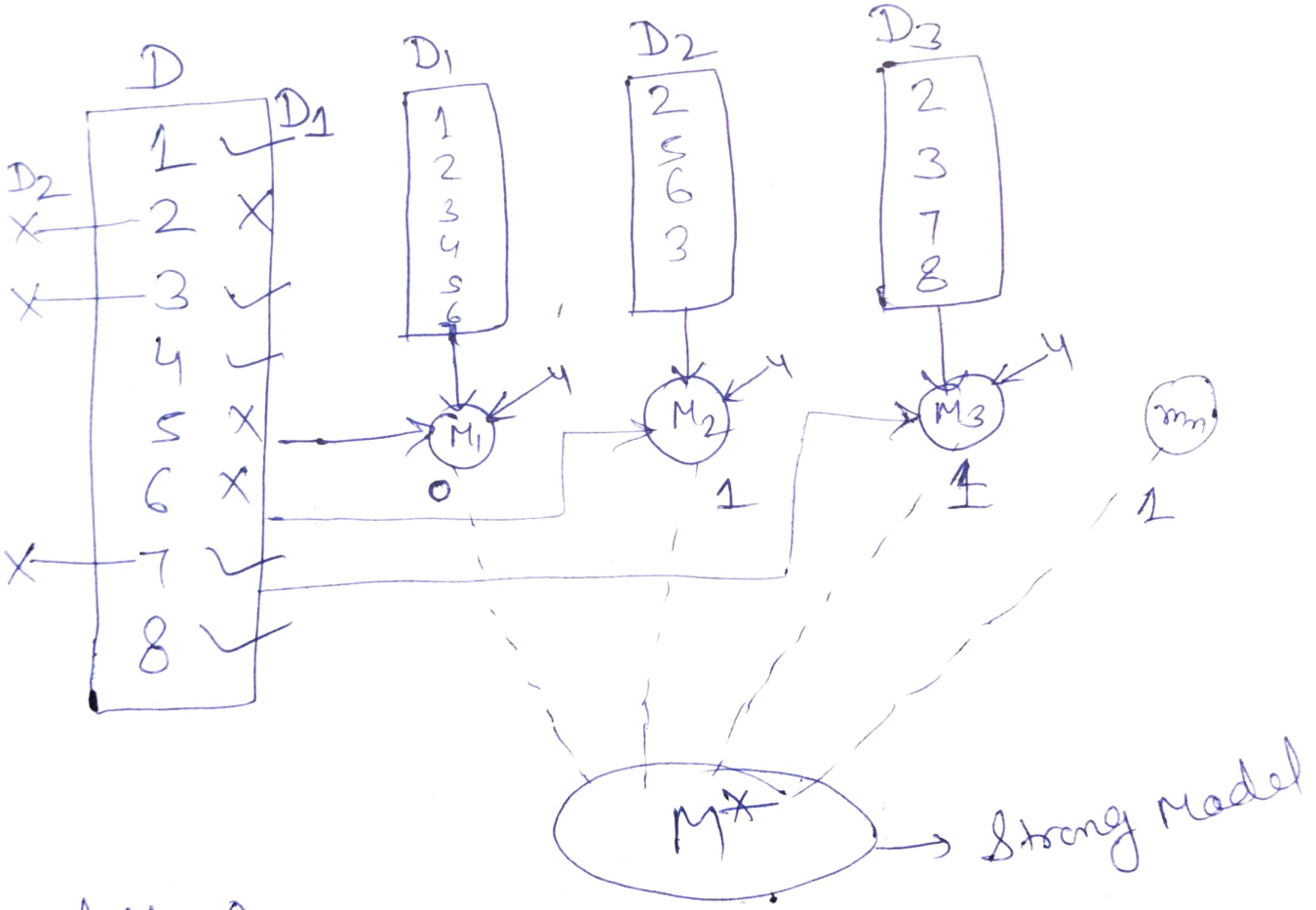
So, now we will change the Probability distribution. And more specifically we identify those instances which were wrongly labeled by Model one, and we will increase the Probability of these instances. those which were Correctly labeled we will Proportionately reduce the Probability.

we want to hide the Probability of those instance which we were not Correctly classified by Model M_1. So we will get the same dataset, but now the Probability distribution will change. Now we will now Sample D_2 according to this Probability distribution. and we will get a Sub Sample and this Sub Sample is more likely to Contain 2, 5, 6 then 1, 3, 4, 7 8, , So we get D_2 and on D_2 we apply Model we apply learning algorithm and get the Model M_2. Now again we apply M_2 on these training instances and which ones D_2 does Error on

Suppose D_2 wrongly classifies 2, 7, 3 and rest it does correctly So, So we will increase the Probability of 2, 3, 7

beyond. What was the current Probability. and reduce the Probability of the rest to keep the sum of the Probability Equal to 1.

So based on that we will find D_3 on which we will train Model M_3 like this we will go on., we will get the different Models a different data and the different Models and then we will Combine these different outputs of different Models by voting and kkk.



class → 0
→ 1 voting ↑
1

$y = 1$

Random Forest Algorithm

* Random Sampling of Training data when building the trees. this first Randomness.

II Randomness → Random A Subset of input features When Splitting at Nodes.

for Example

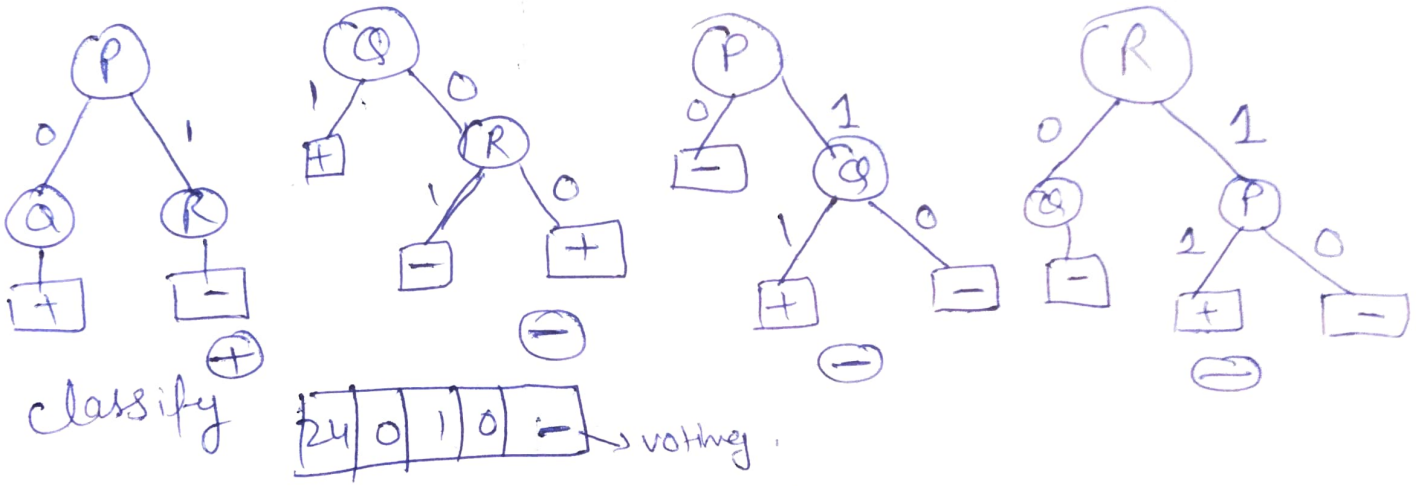
Data set

ID	P	Q	R	Class
20	0	0	1	+
21	1	0	0	−
22	2	0	0	−
23	1	1	2	+

→

Boot Strap Sample

ID	P	Q	R	Class
20	0	0	1	+
22	2	0	0	−
20	0	0	1	+
22	2	0	0	−



classify

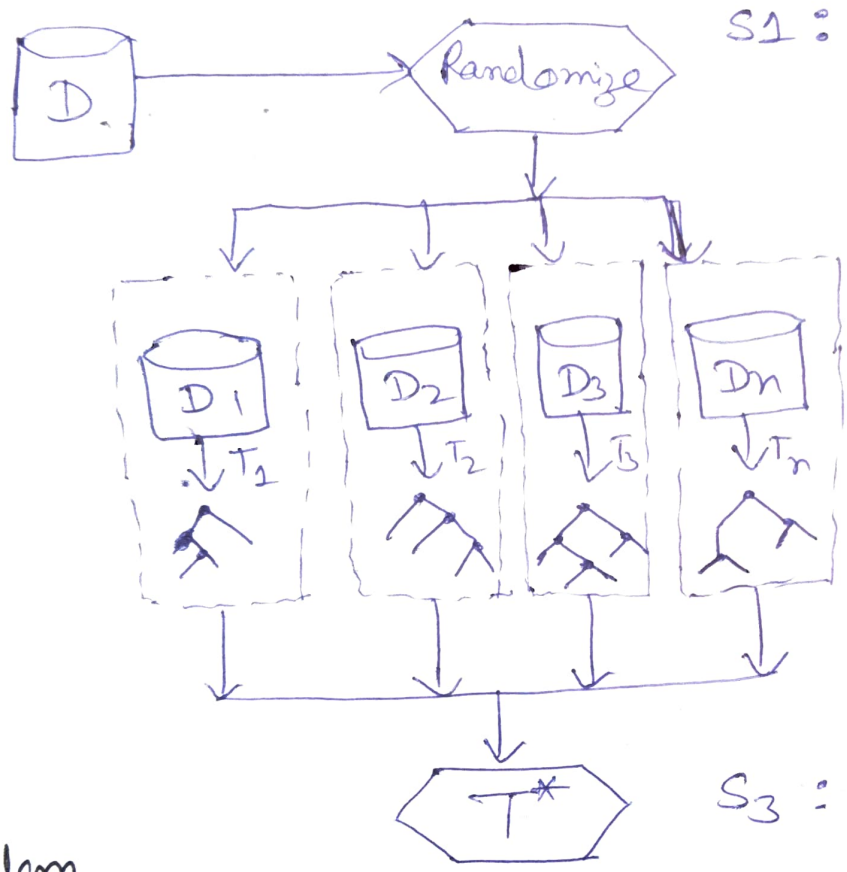
24	0	1	0	−

→ voting.

for Creating a boot Strap Sample we can have duplicate Entries or duplicate observation. So this we have random Sampling of the data Means we have taken randomly. In some random fashion. Second the random Subset of input feature Selection When Splitting at the Nodes now. P, Q, R are the instances or attributes which we have for this data set or boot Strap Sample. So this we Create one instance which have highest Information Gain as you read-read.

And rest two will become the root nodes are node which is intermediate nodes.

So we can create n number of bootstrap sample. or n number of Decision tree. and Prediction are combined using this voting mechanism.



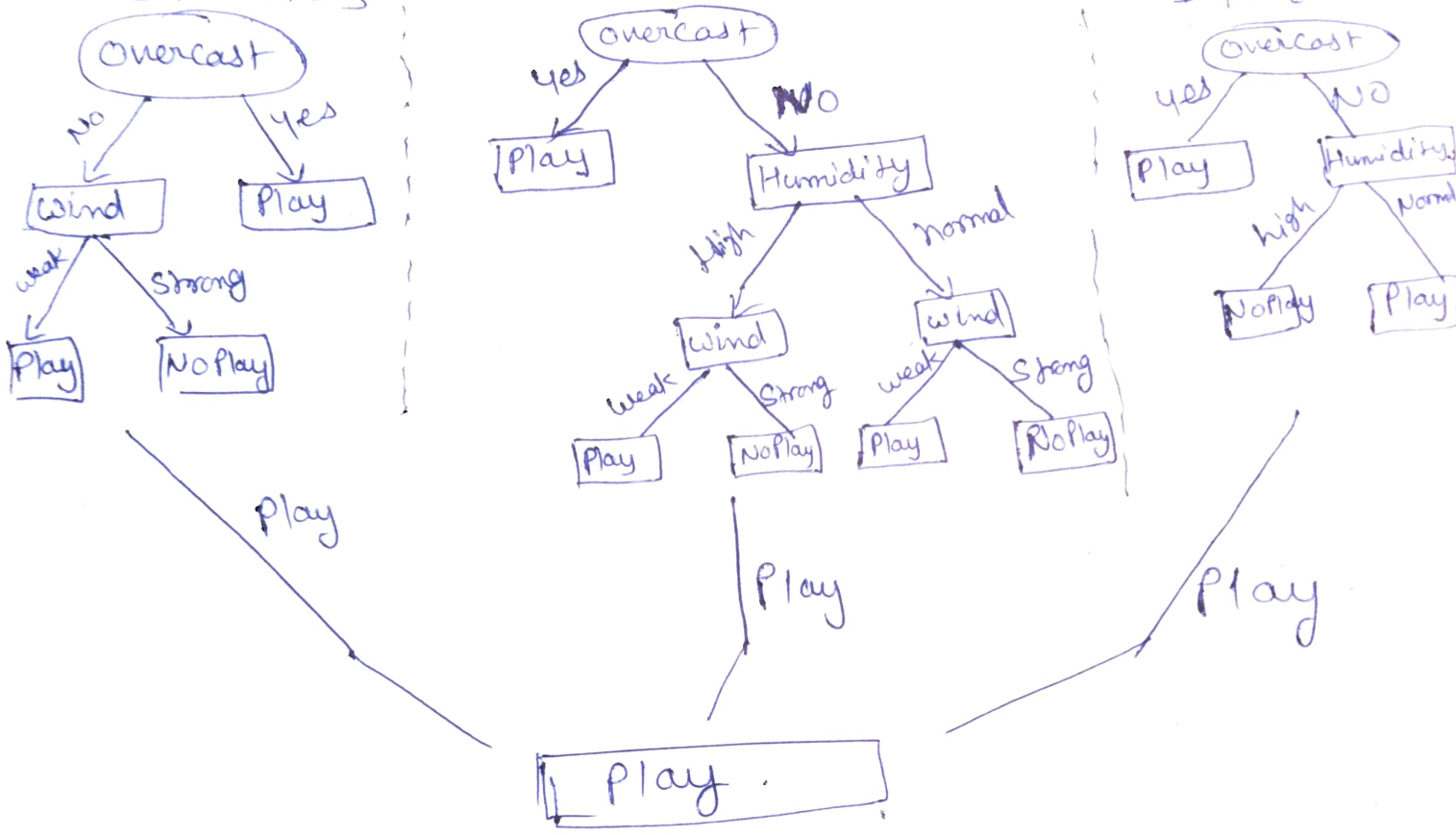
Problem

Day	Outlook	Humidity	Wind	Play
D1	Sunny	High	Wind	NO
D2	Sunny	High	Strong	NO
D3	overcast	High	Wind	yes
D4	Rain	High	Wind	Yes
D5	Rain	Normal	Wind	Yes
D6	Rain	Normal	Strong	NO
D7	Overcast	Normal	Strong	Yes
D8	Sunny	High	Wind	NO
D9	Sunny	Normal	Wind	yes
D10	Rain	Normal	Wind	yes
D11	Sunny	Normal	Strong	yes
D12	Overcast	High	Strong	yes
D13	Overcast	Normal	Wind	yes
D14	Rain	High	Strong	NO

OutHook – Rain
Humidity = High
Wind – weak
Play = ?

Now using Random forest we need to Predict whether the game will happen if the the weather Condition is above mentioned.

* We divide the data into smaller subset.
* Every subset may or may not be overlapped.



features → Most accurate learning algorithm
→ Runs efficiently on large database
→ Perform implicit feature selection (automatically) Collect some random feature.
→ Can be easily grown in parallel so it
→ ~~Re~~ reduce Computation time.

Support Vector Machine (SVM) →

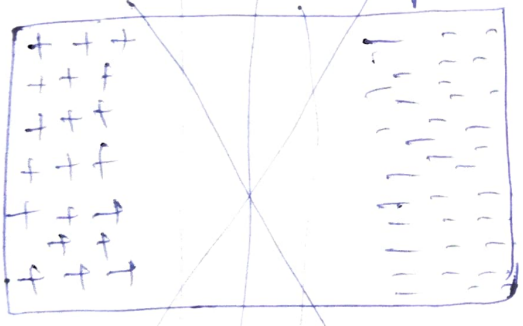
Applications:

1. Hand written · character / digit · recognition
2. it Perform well on high Dimentional Data and Solve the Problem of high Dimentionality.

Support vector: → it means Examples · or traing data

↙

Subset of trainig data and used to represent decision boundary.



→ These are hyperplane.

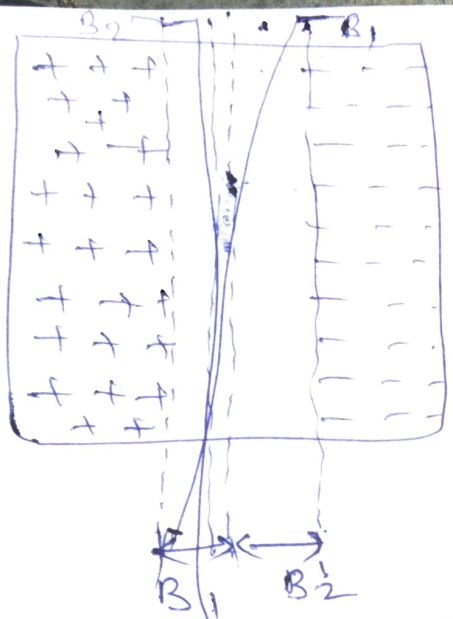
These ~~can be~~ infinite in number

* we can choose most optimal one.
* Training Error on this 0

The chosen hyperplan may not perform in future its not Equally Performed well on any unseen Examples.

Suppose we choose one hyperplan and we use it another day for classification then it can not Equally Seprate all the unseen & future instances. So this is the drawback of this.

for choosing the hyplane we use maximal margin hyperplane.

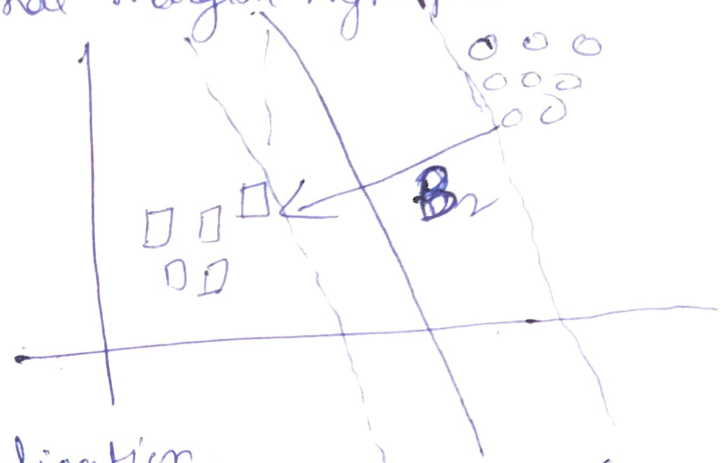


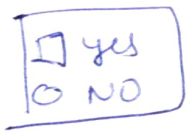
Margin ↑ → better Control on generalization Error

Margin ↓ → Model will fall. victim of overfitting

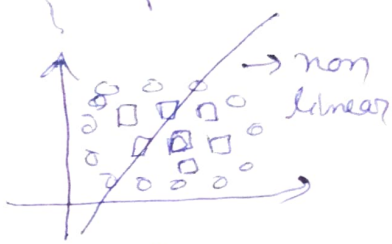
$B_2 > B_1$

Maximal margin hyperplane





Linear classification.



Maximal Margin gives better future Prediction.

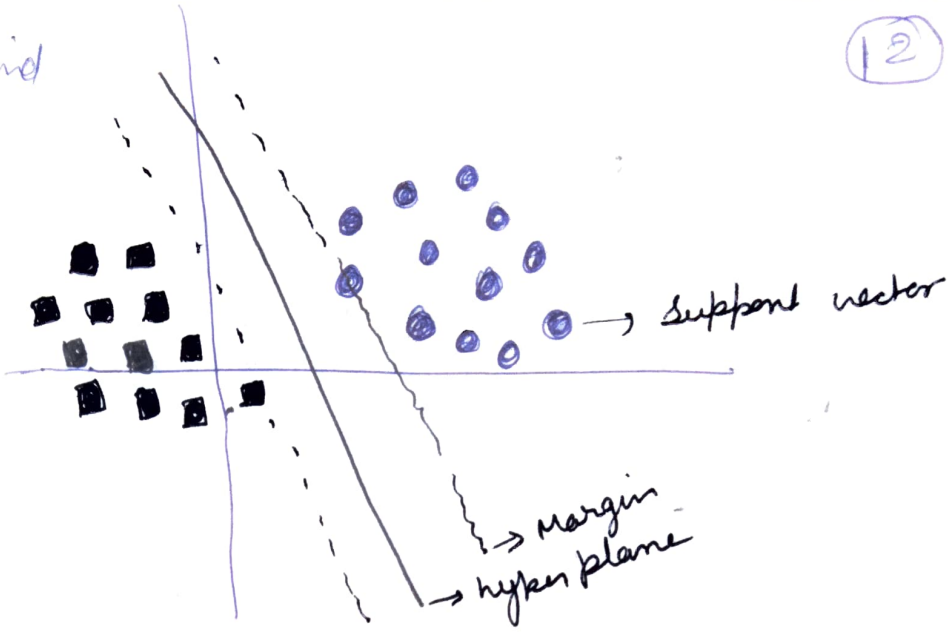
Margin ↑ Misclassification ↓ Accuracy ↑ Error ↓

Accuracy ≤ 50%

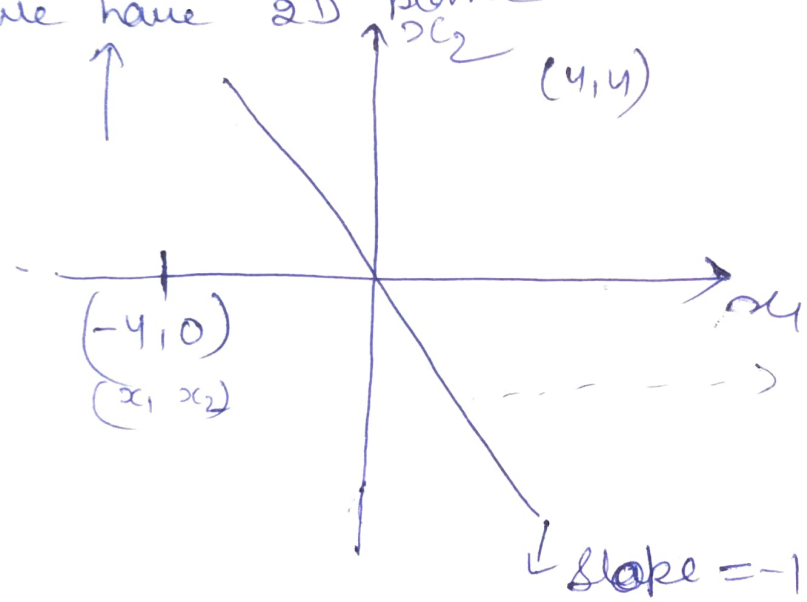
Binary class → line
3D → plane
4-6 → hyper plane

SVM kernal is used to ~~Convert~~ increase dimensional of plane. So it Convert 2D to 3D, 4D

We have to find out Plane.



for Example we have 2D plane



Equation of line

$y = mx + c$

$m = -1$

$c = 0$

$y = mx + c$

if we we Convert this line Equation in to vector. then

$$y = W^T X + c$$

Here
$X = (x_1, x_2)$
W^T = W trasporse of x

and if it is hyperplane then

$$W^T x + b = 0$$

i.e if any point exits in the hyperplane then value of part is zero.

W = Vector

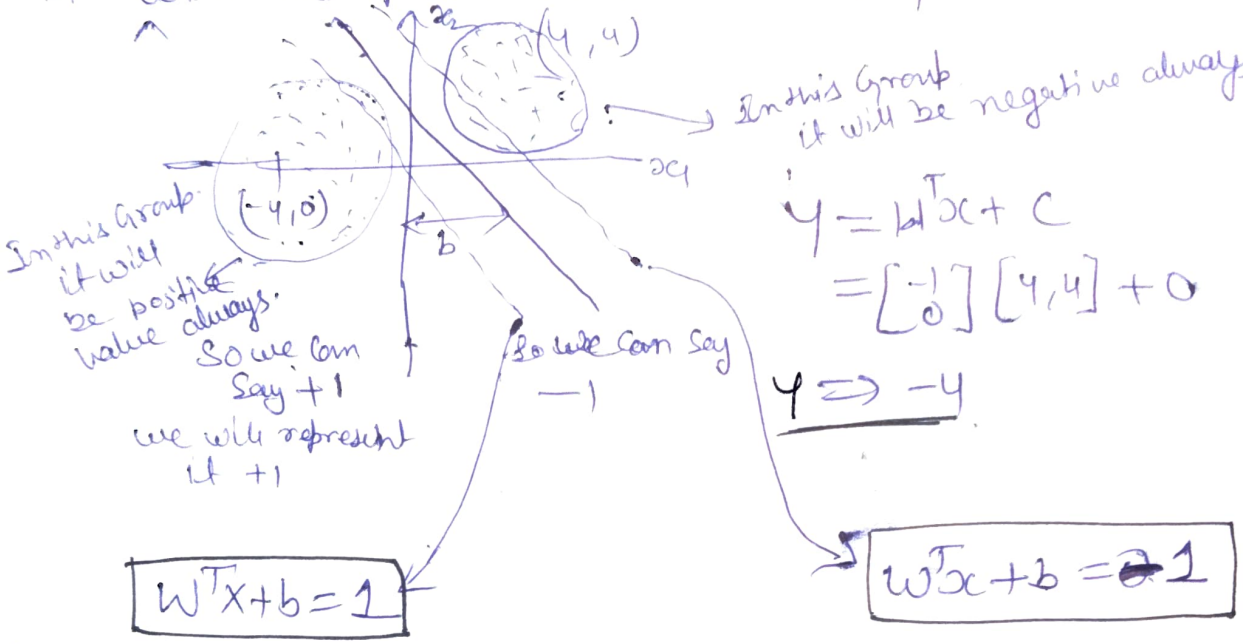
Now $y = w^T x + c$ $\quad m = -1$, $c = 0$

$y = \begin{bmatrix} -1 \\ 0 \end{bmatrix} [-4\ 0] + 0$ $\quad x = [x_1\ x_2]$

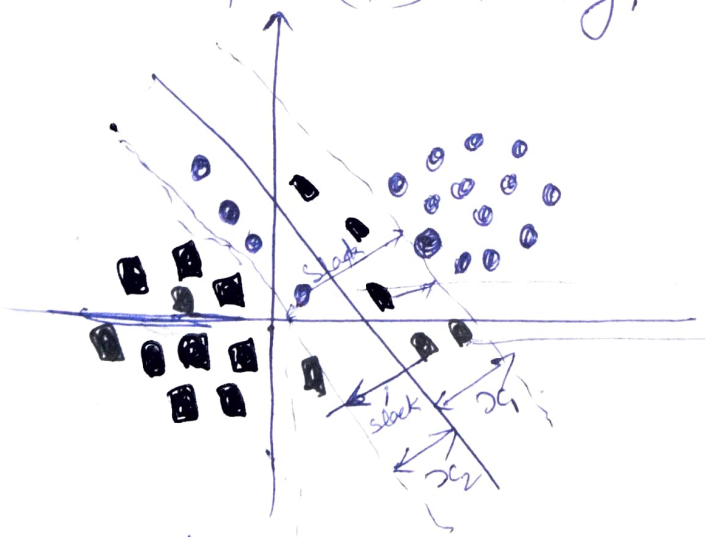
$w^T \begin{bmatrix} -1 \\ 0 \end{bmatrix} \begin{bmatrix} m \\ c \end{bmatrix}$

$y = y$

if we will put any value in this equation so it will be positive in the left side.



b is distance from (0,0) to hyperplane



In this situation misclassification of these point so model will be overfit. we have to select optimized hyperplane

we have to find out $x_2 - x_1$ because points are in opposite direction

$w^T x_1 + b = -1$

$-w^T x_2 - b = -1$

$w^T(-x_2 + x_1) = -2$ $\Rightarrow -w^T(x_2 - x_1) = -2$

$\Rightarrow w^T(x_2 - x_1) = 2$

$$w^T(x_2 - x_1) = 2$$

we have to find out $x_2 - x_1$ so we have to replace w^T for that

$$\frac{w^T}{||w||}(x_2 - x_1) = \frac{2}{||w||}$$

$\frac{w^T}{||w||} = 1$ Norm of w

$$(x_2 - x_1) = \frac{2}{||w||}$$ ↑↑↑

$(x_2 - x_1)$ so we have to increased this for finding optimized hyperplane.

for all w & b

$$(w^*, b^*) \max \frac{2}{||w||}$$ → this is optimized function.

Such that

$$y_i = \begin{matrix} +1 & w^T x + b \geq 1 \\ -1 & w^T x + b \leq 1 \end{matrix}$$

$$y_i * w^T x + b \geq 1$$

↘ it is always true. if it is not true then misclassification.

and if we have to find min of any maximum value then we have to do reciprocal of Max.

$$(w^* b^*) \min \frac{||w||}{2} + C_i \sum_{i=1}^{n} \zeta$$ → hyperfunction

ζ slack variable

No of point in Misclassification to allow.

How many Error you want

final Equation.

this will be regularization to avoid overfit.

Slack variable = value of Error it is a distance between point to margin.