MACHINE LEARNING WITH SCIKIT-LEARN

List of common machine learning algorithms

1. Linear Regression
2. Logistic Regression
3. Decision Tree
4. SVM
5. Naive Bayes
6. KNN
7. K-Means
8. Random Forest
9. Dimensionality Reduction Algorithms
10. Gradient Boosting algorithms
    1. GBM
    2. XGBoost
    3. LightGBM
    4. CatBoost

what is EDA?

Supervised learning
where we have input variable (say x) and output variable (say Y) and an algorithm to map the input to the output
That is, $Y = f(x)$

Unsupervised learning
No corresponding output variable is there.
Helps in understanding the data.

Broadly, there are 3 types of Machine learning Algorithms.

① Supervised learning (dependent variable)

* This algorithm consist of a outcome variable/target which is to be predicted from a given set of independent variables
* Using these set of variables, we generate a function that map inputs to desired outputs.
* The training continues until the model achieves a desired level of accuracy on the training data.

Example: decision tree, Random forest, KNN, logistic Regression etc.

② Unsupervised learning

* We don't have any target or outcome variable to predict/estimate.
* It is used for clustering (have similar numerical values) population in different groups, which is widely used for segmenting customers in different groups for specific intervention
* example: Apriori algorithm, K-means.

③ Reinforcement learning:

* Machine learns from past experience & tries to capture the best possible knowledge to make accurate business decisions.
* The machine is trained to make specific decisions

* The machine is exposed to an environment where it learn itself continually using trial and error.

example - Markov Decision Process

① LINEAR REGRESSION

What is Regression analysis

It is a form of predictive modeling technique in which we investigates the relationship b/w a dependent and independent variable.

Home prices in Monroe

area	price
2600	550000
3000	565000
3200	610000
3600	680000
4000	725000

Given these home prices find out prices of homes whose area is,

3300 square feet

5000 square feet

Linear vs logistic Regression

Basis	Linear Regression	Logistic Regression
Core Concept	The data is modelled using a straight line	The probability of some obtained event is represented as linear function of a combination of predictor variables
Used with	Continuous Variable	Categorical Variable

Output/Prediction	Value of the Variable	Probability of occurrence of event.
Accuracy and goodness of fit	measured by loss, R Squared, Adjusted R Squared etc.	Accuracy, Precision, Recall, F1 Score, ROC curve, Confusion Matrix etc.

Where we will use Linear Regression

① Where is linear Regression used?

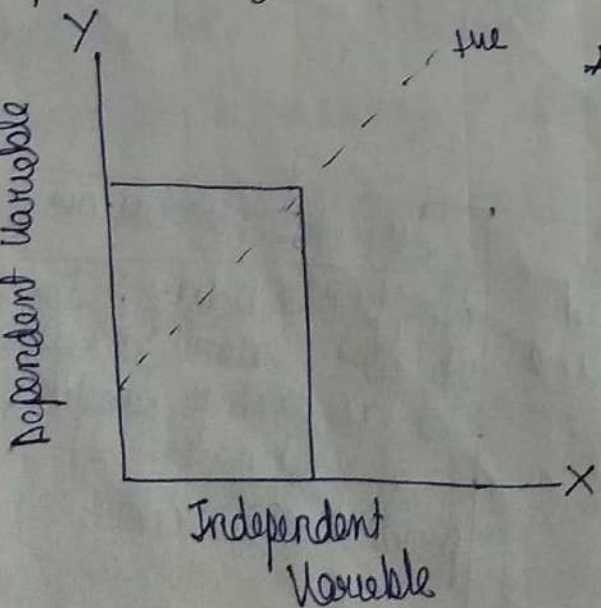
⇒ Evaluating Trends & Sales Estimates

⇒ Analyzing the impacts of price changes

⇒ Assessment of risk in financial services and insurance domain.

Understanding linear Regression Algorithm

① Positive regression types of linear regression -

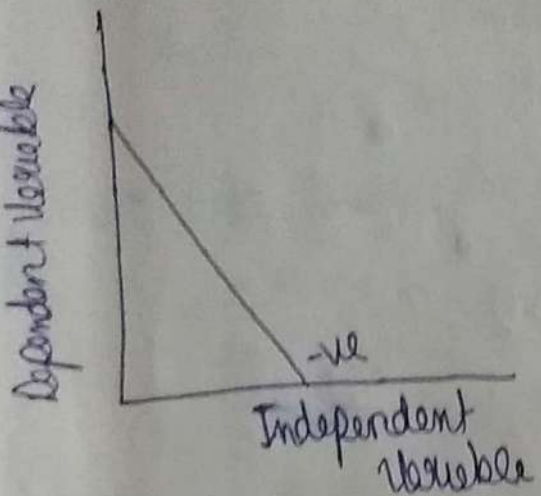


* Suppose when Independent Variable increase, dependent Variable also increase it is known as positive regression

* $Y = ax + b$

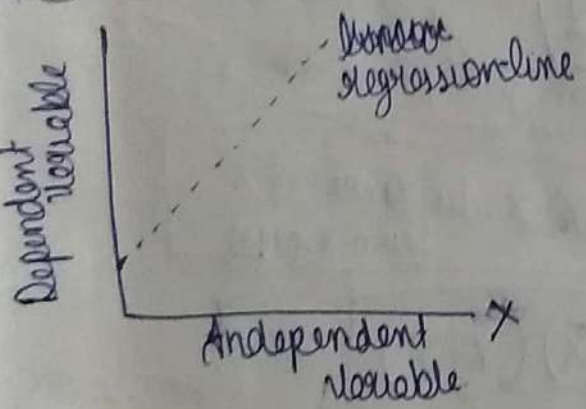
** Y = dependent Variable
x = Independent Variable
a = Slope
b = intercept

③ Negative Regression



* Suppose when Independent variable increase, dependent variable decrease it is known as negative regression
* $y = -ax + b$
  - y = dependent variable
  - x = Independent variable
  - a = Slope
  - b = intercept

Understanding linear Regression



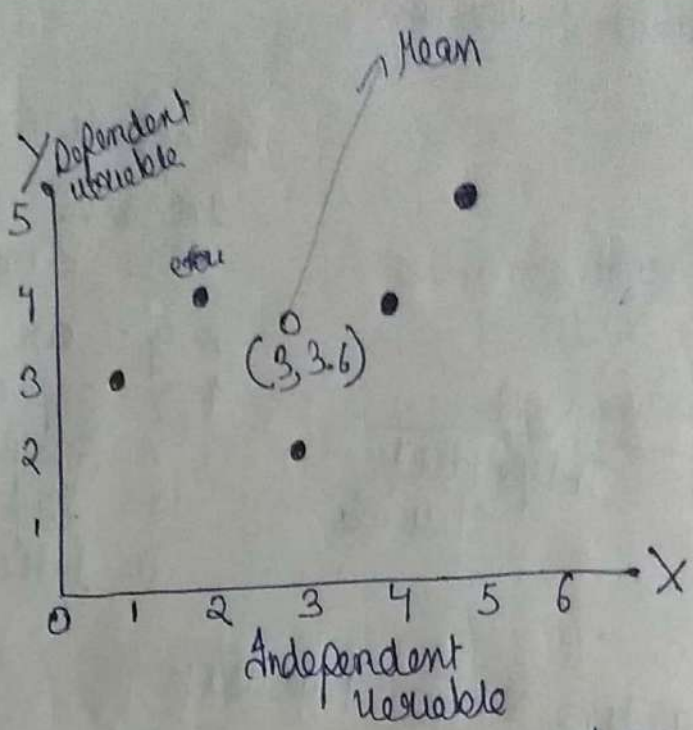
Slope: It represents the rate of changes in y as x changes because Y is dependent on x

Intercept:

Practical Example:

Independent X	Dependent Y
1	3
2	4
3	2
4	4
5	5

Lets plot this in graph



Mean of X = 3

Mean of Y = 3.6

Now plot this mean in graph for X & Y → Y is dependent on x

Slope: Rate of change in y as x changes

a = Slope

$$a = \frac{\Sigma (x-\bar{x})(y-\bar{y})}{\Sigma (x-\bar{x})^2} = \frac{4}{10}$$

$x-\bar{x}$ = distance of x from x mean

X	Y	$x-\bar{x}$	$y-\bar{y}$	$(x-\bar{x})^2$	$(x-\bar{x})(y-\bar{y})$
1	3	-2	-0.6	4	1.2
2	4	-1	0.4	1	-0.4
3	2	0	-1.6	0	0
4	4	1	0.4	1	0.4
5	5	2	1.4	4	2.8
Mean 3	3.6			$\Sigma = 10$	$\Sigma = 4$

$$a = \frac{4}{10} \text{ or } a = 0.4$$

Now, lets find the value of b (intercept)

$$Y = aX + b$$

$$3.6 = 0.4 \times 3 + b$$

$$b = 2.4$$

mean of Y = 3.6
mean of X = 3

The intercept is the expected mean value of Y when all X = 0

Finding regression line

$$y = ax + b$$

$$Y = 0.4x + 2.4$$

Now lets predict the value of Y using x values :-

$Y = 0.4 \times 1 + 2.4 = 2.8$

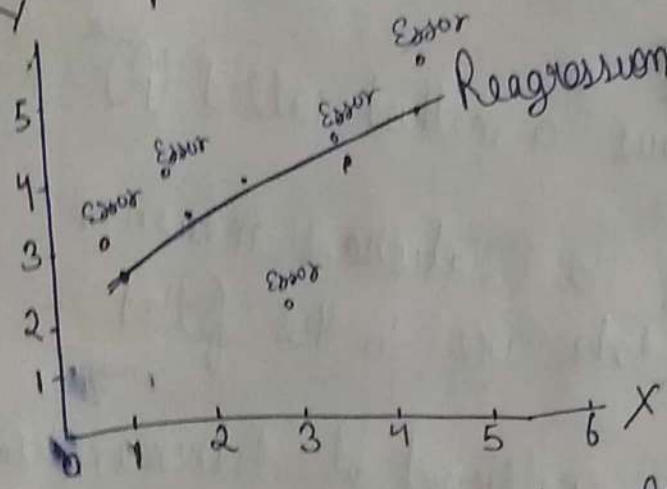
$Y = 0.4 \times 2 + 2.4 = 3.2$

$Y = 0.4 \times 3 + 2.4 = 3.6$

$Y = 0.4 \times 4 + 2.4 = 4.0$

$Y = 0.4 \times 5 + 2.4 = 4.4$

Now lets plot this regression line in graph



Now our next task is to calculate the distance b/w the actual and predicted value and our job is to reduce the distance.

Now we have to compare :-

distance actual - mean compare distance estimated - mean

actual Y	$Y - \bar{Y}$	$(Y - \bar{Y})^2$	estimated $\hat{Y}$	$\hat{Y} - \bar{Y}$	$(\hat{Y} - \bar{Y})^2$
3	$3 - 3.6 = -0.6$	0.36	2.8	$2.8 - 3.6 = -0.8$	0.64
4	$4 - 3.6 = 0.4$	0.16	3.2	$3.2 - 3.6 = -0.4$	0.16
2	$2 - 3.6 = -1.6$	2.56	3.6	$3.6 - 3.6 = 0$	0
4	$4 - 3.6 = 0.4$	0.16	4.0	$4.0 - 3.6 = 0.4$	0.16
5	$5 - 3.6 = 1.4$	1.96	4.4	$4.4 - 3.6 = 0.8$	0.64
		$\Sigma = 5.2$			$\Sigma = 1.6$

$$R^2 = \frac{1.6}{5.2} = 0.307 = \frac{\sum(\hat{Y}-\bar{Y})^2}{\sum(Y-\bar{Y})^2} \quad \frac{\text{(estimated)}}{\text{actual}}$$

* If $R^2 = 1$ it is a perfect fit
* If $R^2 = 0.2$ then there is large distance b/w actual and estimated
* If $R^2 \to 0$ then there is no relation at all

Now its time to check how good our model is performing

To check this we have a method called $(R)^2$ method

⇒ R-Squared value is a statistical measure of how close the data are to the fitted regression line

⇒ It is also known as coefficient of determination, or the coefficient of multiple determination

It is discussed above in the example

Note: Any field that attempts to predict the human behaviour typically has R^2 value lower than 0.5

Slope(a) is also called coefficient

Let's learn to code

Linear Regression Single Variable

Home prices

area	price
2600	550000
3000	565000
3200	610000
3600	680000
4000	725000

Given those home prices find out prices of homes where area is
3300 Square feet
5000 Square feet

An
```
import pandass
import numpy
import matplotlib.pyplot as plt
from SKlearn import linear_model
```

An
```
df = pd.read_csv("homeprices.csv")
df
```

from pdf

Multiple Variables

Home prices in Monroe township.

area	bedrooms	age	price
2600	3	20	550000
3000	4	15	565000
3200		18	610000
3600	3	30	595000
4000	5	8	760000

Given these home prices find out price of a home that has,

3000 sqr ft area, 3 bedrooms, 40 years old
2500 sqr ft area, 4 bedrooms, 5 year old

here price is dependent on price

Intercept

price = a_1 * area + a_2 * bedrooms + a_3 * age + b

price → dependent Variable

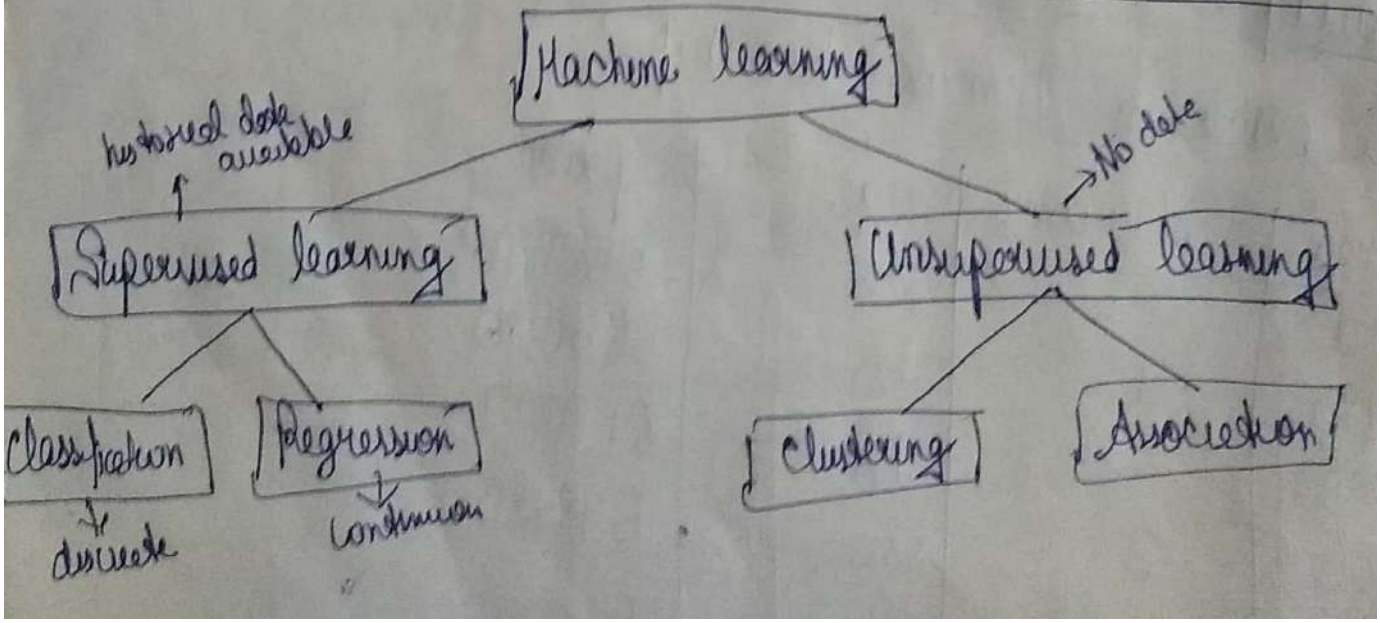
area, bedrooms, age → Independent variables (or) features

Topics

- Data preprocessing : Handling NA values
- Linear Regression Using Multiple Variables

from jupyter

File name linear_reg_single-multi



2. LOGISTIC REGRESSION

* used for classification (Supervised learning)

Solutions to classification

1. Decision Trees :
2. K-Nearest Neighbor :
3. logistic Regression :

What is logistic Regression ?

An this predicted value is categorical

For example :

1. Will customer buy life insurance?
2. Which party a person is going to vote for ?
   1. Democratic
   2. Republican
   3. Independent

Classification Types

Will customer buy life insurance ?

1. Yes
2. No

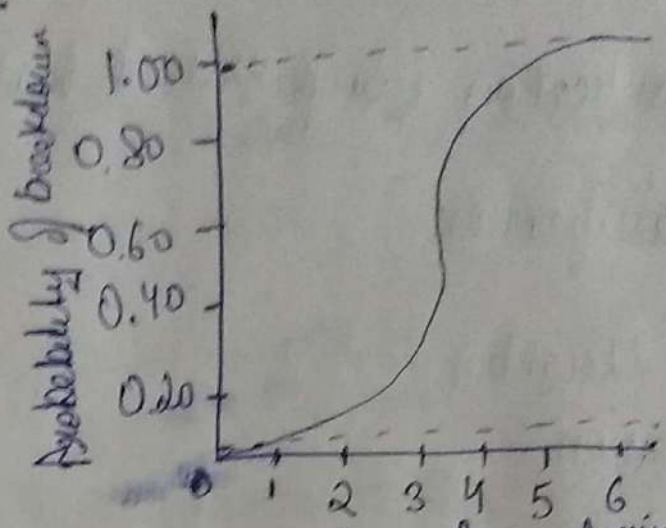
This is Binary classification

Which party a person is going to vote for ?

1. Democratic
2. Republican
3. Independent

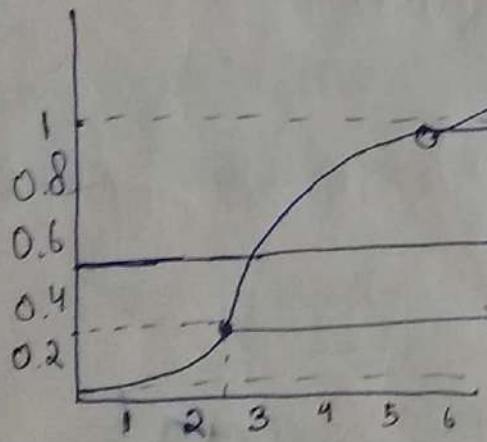
This is Multiclass classification

* How long until the car breaks down?



model created based on other users experience

* explanation



Probability > 0.50 value rounded off to 1 : the car will breakdown

Threshold Value

Probability < 0.50 value rounded off to 0 : the car will not breakdown

Threshold Value → Here, the threshold value 0.50 indicates that car is more likely to breakdown after 3.5 years of usage.

The output of 0.8 means the car will break down & the output of 0.4 means that the will not breakdown

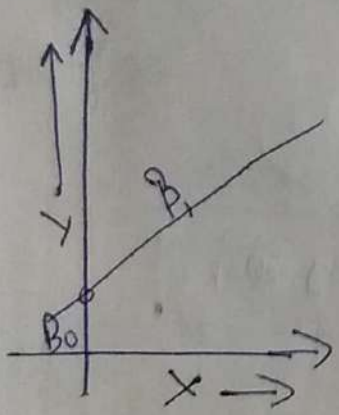
The Math behind Logistic Regression

To understand logistic Regression, let's talk about the odds of success

$$\text{Odds}(\theta) = \frac{\text{Probability of an event happening}}{\text{Probability of an event not happening}} \quad \text{or} \quad \theta = \frac{p}{1-p}$$

The values of odds range from 0 to ∞. The values of probability change from 0 to 1

* Take the equation of the straight line



The equation would be

$$y = B_0 + B_1 x$$

Here, B_0 is the y-intercept, B_1 is the slope of the line, x is the value of the co-ordinate, y is the value of the prediction

Now, we predict the odds of success

$$\log\left(\frac{P(x)}{1-P(x)}\right) = B_0 + B_1 x$$

Exponentiating both sides:

$$e^{\ln\left(\frac{P(x)}{1-P(x)}\right)} = e^{B_0 + B_1 x}$$

$$e\left(\frac{P(x)}{1-P(x)}\right) = e^{B_0 + B_1 x}$$

(equation of straight line)

let $Y = e^{B_0 + B_1 x}$

Then $\frac{P(x)}{1-P(x)} = Y$

$P(x) = Y(1-P(x))$

$$P(x) = Y - Y(P(x))$$
$$P(x) + Y(P(x)) = Y$$
$$P(x)(1+Y) = Y$$
$$P(x) = \frac{Y}{1+Y}$$
$$P(x) = \frac{e^{B_0 + B_1 x}}{1 + e^{B_0 + B_1 x}}$$

The equation of a Sigmoid function or logit function

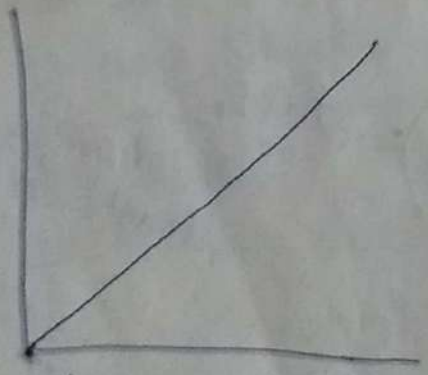
$$P(x) = \frac{e^{B_0 + B_1 x}}{1 + e^{B_0 + B_1 x}}$$

e = Euler's number ~ 2.7182

$$P(x) = \frac{1}{1 + e^{-(B_0 + B_1 x)}}$$

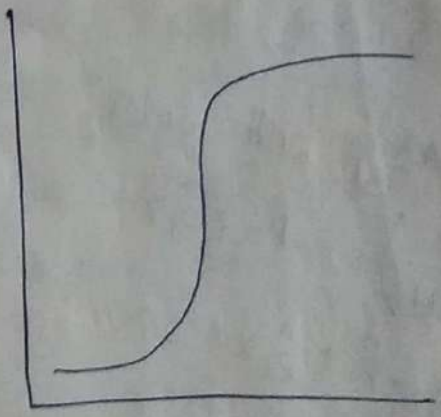
Sigmoid/logit function converts input range 0 to 1

$y = m * x + b$



Linear regression

$$y = \frac{1}{1 + e^{-(m*x+b)}}$$



Logistic Regression

Remaining on Jupyter Notebook

Continue after 3 pages

Before proceeding further we need to learn what is training and Testing Data is

TRAINING & TESTING DATA

Generally, when we have a dataset we train our model using entire dataset, But that's not a Good Strategy.

The Good Strategy will be to split our data into two parts. where one part is used for training and one part is used for testing

For example:

	Mileage	Age	Sell price
Training 80%	69000	6	18000
	35000	3	34000
	57000	5	26100
	22500	2	40000
	46000	4	315000
Test	59000	5	26750

This will Give Good idea of aquarcy of the model

An
```
import pandas as pd
df = pd.read_csv("carprices.csv")
df.head()
```

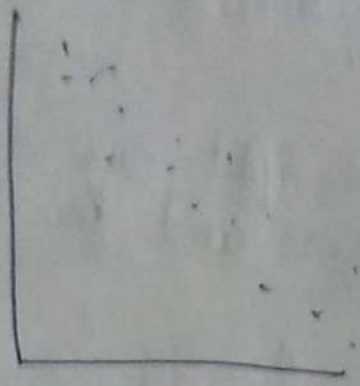
Out

	Mileage	Age	Sell price
0	690	—	—
1	—	—	—
2	—	—	—
3	—	—	—
4	—	—	—

In | import matplotlib.pyplot as plt
%

In | plt.scatter(df['mileage'], df['Sell Price($)'])

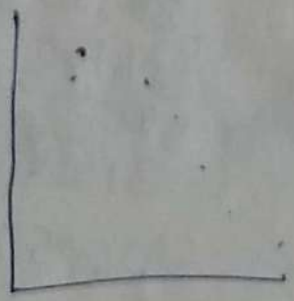
Out |



(Car Mileage Vs Sell Price)

In | plt.scatter(df['Age(yrs)'], df['Sell Price($)'])

Out |



(Car age Vs Sellprice)

In | X = df[['Mileage', 'Age(yrs)']]
y = df['Sell Price($)']

In | from SKlearn.model_selection import train_test_split

In | X_train, X_test, y_train, y_test = train_test_split(X, y
, test_size = 0.2)

Note: We get 4 arguments from this test. Test size is a size of a test that we want. Here, size of test is 20%.

In | len(X_train)

Out | 16

In | len(X_test)

Out | 4

In | X_train

Out |

	Mileage	Age(yrs)
4	—	—
19	—	—
14	—	—
⋮	⋮	⋮

Note: Everytime we run X_train it will give random rows from the dataset. It is not using first 80% of the data for model

Note: If we don't want our data to select random data everytime we can use random_state=10) do like this

x_train, x_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=10)

~~Continuation~~

Logistic Regression (multiclass classification)

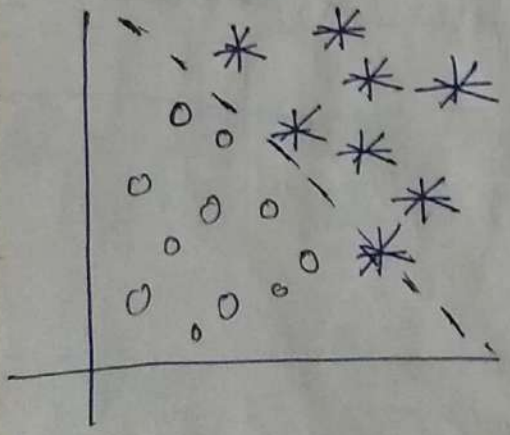
From Jupyter notebook

DECISION TREE

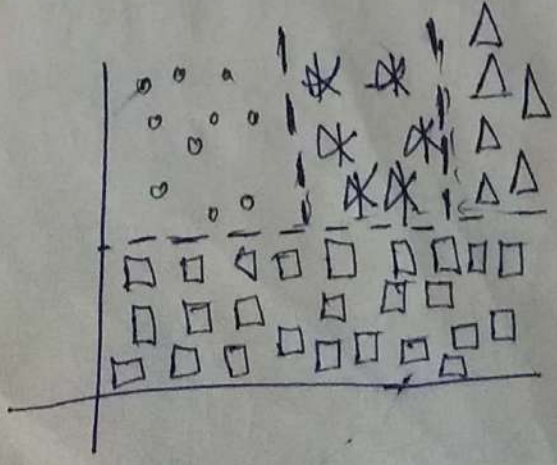
It belongs to the family of supervised learning algorithms. It can be used for classification and regression

We here will solve classification problem by using decision tree

Regression → predicting the value of target



when we have data set like this its easier to draw decision boundary using logistic regression

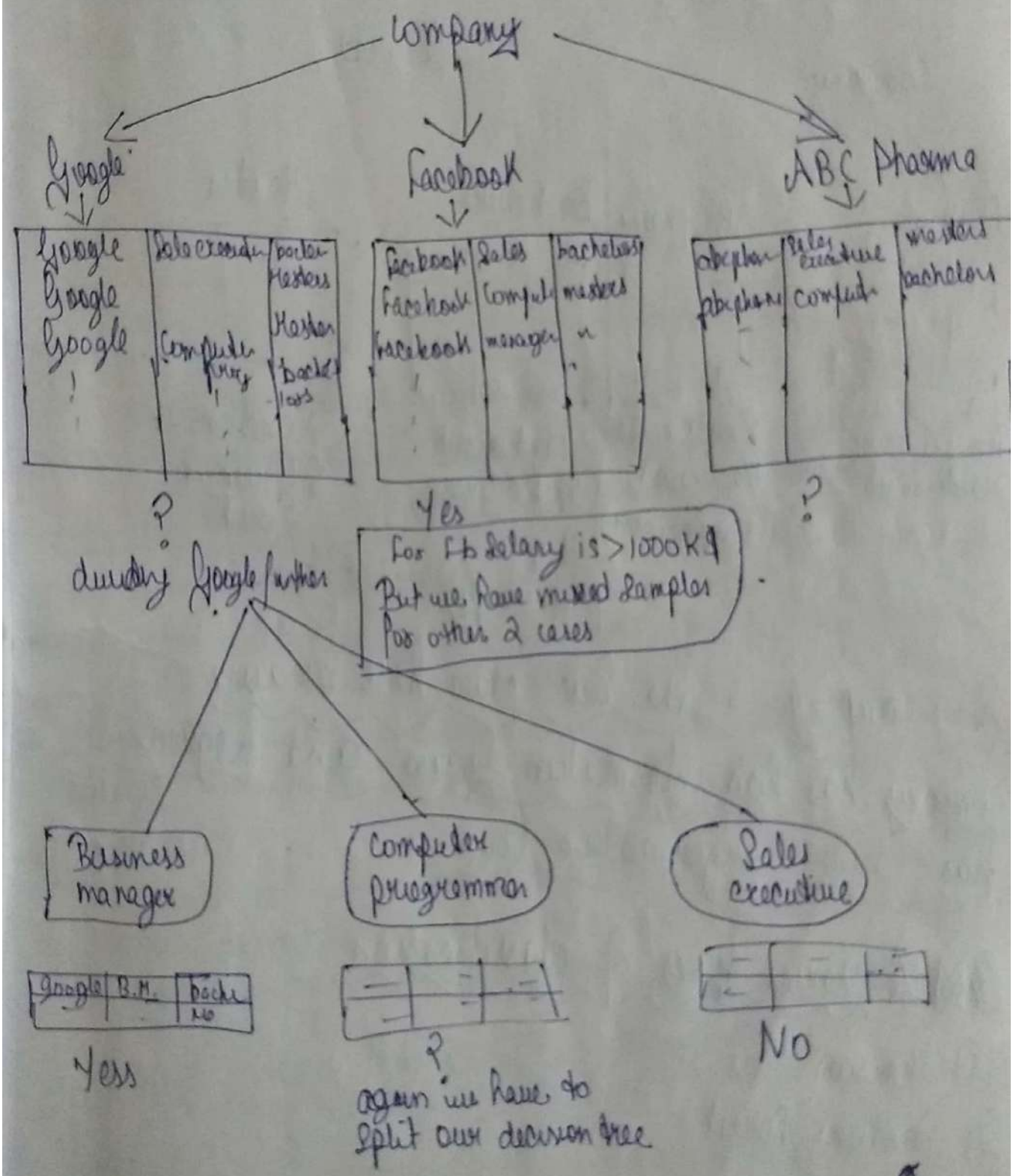


But when we have data like this we cannot just draw a single line. we might have to split our data set again and again

This is what decision tree algo does it for you.

* When you give this data set (in Jupyter Notebook) you will naturally try to build decision tree in our brain

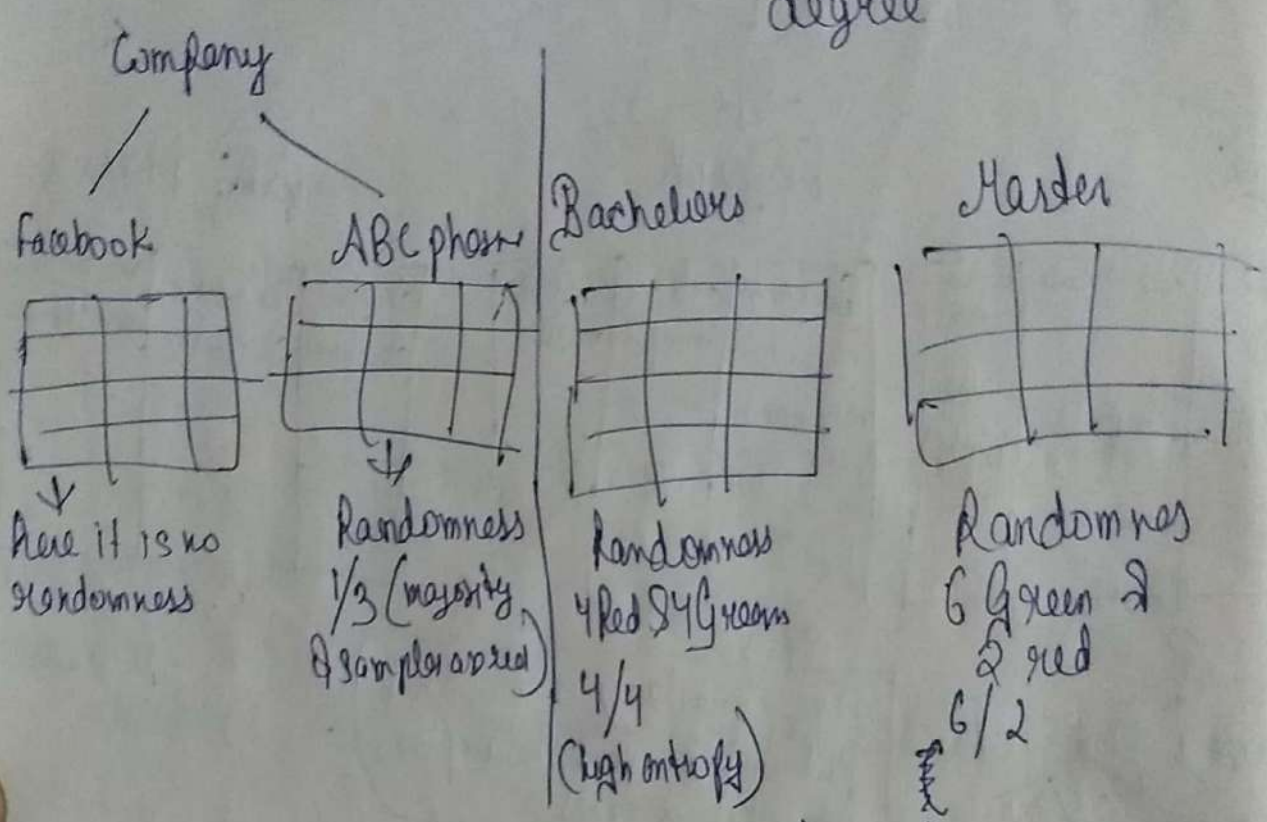
* First we'll split the data base using company like this



An which order we are going to split our dataset (like here we split on the bases of company then profile) will effect our result.

How do you select ordering of features?

An our above example we have used company first. What if we split our data only basics of degree first



So company is the best option here because company has high information gain and degree has low information gain.

Various types of classification

1. Decision Tree
2. Random Forest
3. Naive Bayes
4. KNN

Understanding a decision tree deeply

This is how our dataset looks like

open: decision tree by Rasabh rais

Colour	Diameter	label
Green	3	Mango
Yellow	3	Mango
Red	1	Grape
Red	1	Grape
Yellow	3	lemon

In order to build a decision tree we will use a decision tree algorithm called CART (Classification and Regression Tree)

Let's see how it works

* Root will receive the entire dataset
* and Nodes will receive a list of rows as input & now each row will ask True & false question.
* True & false will split the data into two different subsets. Then these subsets then become input to 2 child node
* Our goal is to ~~unmix~~ the label or produce purest possible distribution. we continue dividing the data until there are no further question to ask

node = the part of a plant stem which one or more leaves emerge

Decision Tree terminology

1. Root Node → It represents the entire population or sample and this further gets divided into two

node: a point in a network

or more homogenous sets.

(2) leaf Node → Node cannot be further segregated into further nodes

(3) Splitting → Splitting is dividing the rootnode/ Sub node into different parts on the basis of some condition.

(4) Branch or Sub tree
Formed by Splitting the tree/node

(5) Pruning
Opposite of Splitting, basically removing unwanted branches from the tree

(6) Parent and child node
Root node is the parent node and all the other nodes branched from it is known as child node

How Does A tree Decide where to Split ?

Some terminologies that you should know

ID3 → It is a Group of algorithm
- Information gain
- Entropy

(1) Gini Index : The measure of impurity used in building decision tree in CART is Gini index

(2) Information Gain - The information gain is the decrease in entropy after a dataset is split on the basis of an attribute. Constructing a

decision tree is all about finding attribute that returns the highest information gain

variance = It measures how far a data set is spread out

variance = $\frac{\Sigma(x-\bar{x})}{n}$

(3) Reduction in Variance - Reduction in Variance

Reduction in variance is an algorithm used for continuous target variables (regression problems). The split with lower variance is selected as the criteria to split the population

(4) Chi Square

It is an algorithm to find out the statistical significance b/w the differences b/w ~~the differences~~ sub nodes and parents node.

(5) Entropy ⇒ It is matrix measures the impurity of something. It is the first step to do before you solve the problem of decision tree.

lets first Understand what is impurity

* Suppose we have a basket of apple and another basket with apple labels. If one asks me to pick one item from each bowl then probability of getting a apple and its correct label is 1
So, in this case impurity is 0.

* Now, what if there is 4 different fruit and 4 different label in the bowl. Then the probability of matching the fruit with label is not 1
So, in this impurity is not 0

What is Entropy?

> Entropy is a measure of randomness in the information being processed

If number of yes = number of no ie P(S) = 0.5

$$\text{Entropy} = -P(\text{Yes}) \log_2 P(\text{Yes}) - P(\text{no}) \log_2 P(\text{no})$$

where,

- S is the total sample space
- P(Yes) is probability of yes

If number of yes = number of no. ie P(S) = 0.5

$\Rightarrow$ Entropy(S) = 1

If it contains all yes or all no ie P(S) = 1 or 0

$\Rightarrow$ Entropy(S) = 0

Note: Lets see them Mathematically

I^st Case Entropy(S) = 1 (no. of Yes = no. of No)

$$E(S) = 0.5 \log_2 0.5 - 0.5 \log_2 0.5$$

$$E(S) = 0.5 (\log_2 0.5 - \log_2 0.5)$$

$$E(S) = 1$$

II^nd Case E(S) = 0 (contains all yes or No)

$$E(S) = -P(\text{Yes}) \log_2 P(\text{Yes})$$

when P(Yes) = 1 ie Yes = Total Sample(S)

$$E(S) = 1 \log_2 1$$

$$E(S) = 0$$

$E(S) = -P(No) \log_2 P(No)$

when $P(No) = 1$ ie No = Total Sample (S)

$E(S) = 1 \log_2 1$

$E(S) = 0$

Play Golf	
Yes	No
9	5

→ Entropy (Play Golf) = Entropy (S)
= Entropy (0.36, 0.64)
= $(0.36 \log_2 0.36) - (0.64 \log_2 0.64)$
= 0.94

What is information gain?

- Measures the reduction in entropy
- Decides which attributes should be selected as the decision node

If S is our total collection,

Information gain = Entropy(S) - [(Weighted Avg) × Entropy(each feature)]

- Information gain measures how much "information" a feature tells us about the class.

Algorithm used in decision trees:

1. ID3
2. Gini Index
3. chi-square
4. Reduction in variance

ID3 = The core algorithm for building decision trees is called ID3. It uses Entropy & Information gain to construct a decision tree.

Chi-square → It can perform 2 or more splits

* Higher the value of chi-square higher the statistical significance of difference b/w sub-node & Parent node

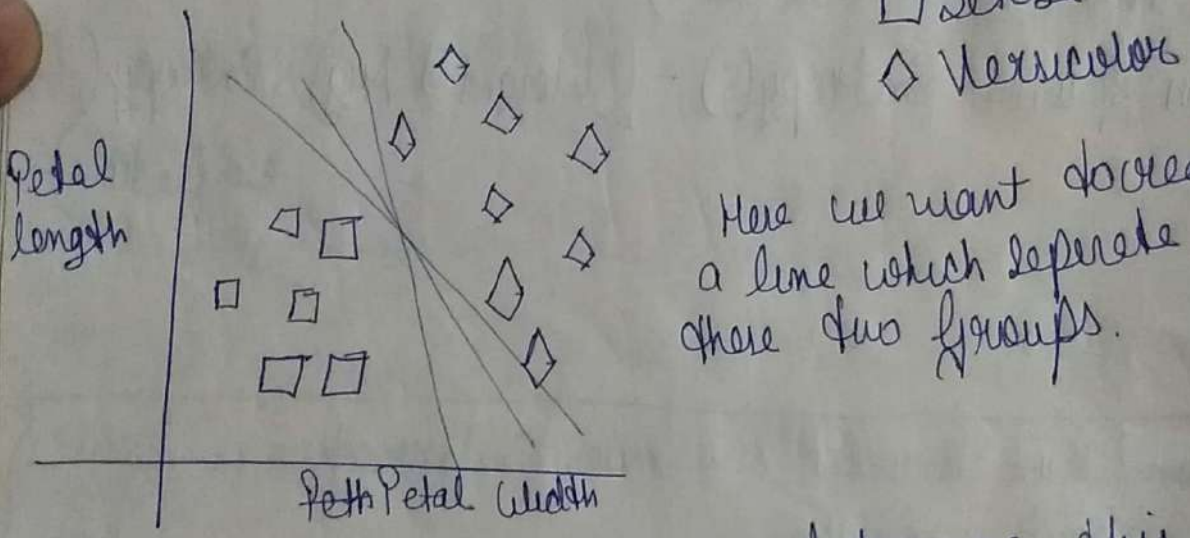
* chi-square = $\frac{(\text{Actual} - \text{Expected})^2}{\text{Expected}}$

* It generates tree called CHAID (chi-square Automatic Interaction Detector)

(4) Support Vector Machine (SVM)

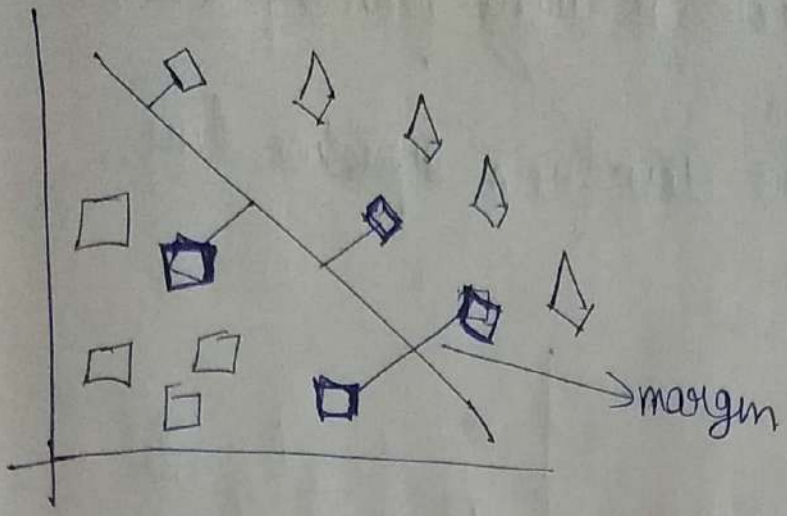
* It is very popular classification algorithms
* It is a part of supervised learning.

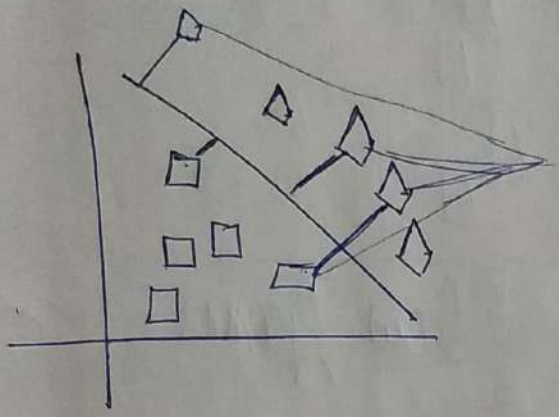
Iris flower which has four features for petal (width & height) for sepal (width & height) Based on these 4 features you can determine the species of the iris flower there are 3 different species this data set is available in Sklearn dataset module.



Here we want to create a line which seperate these two groups.

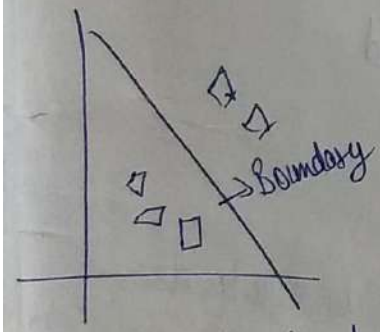
* Here we have many ways of drawing this boundary all these three are valid boundaries do. How will you decide which boundary is the best for my classification problem
* One way of looking at it is you can take nearby data points and you can measure the distance from that line to the data so best way to do this is to maximize the margin b/w the nearby data points. like this.





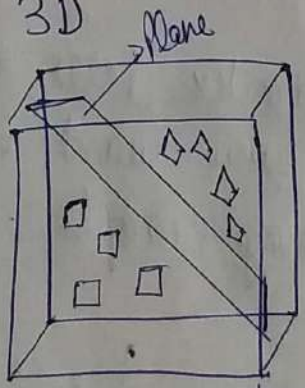
These nearby data points are called support vectors

2D



Boundary is aligned

3D



Boundary is a plane

nD

It's not possible but mathematically we called it hyper plane

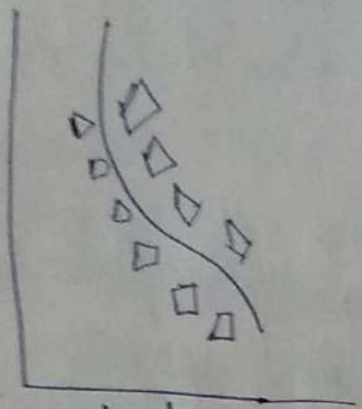
Gamma & Regularization

Gamma

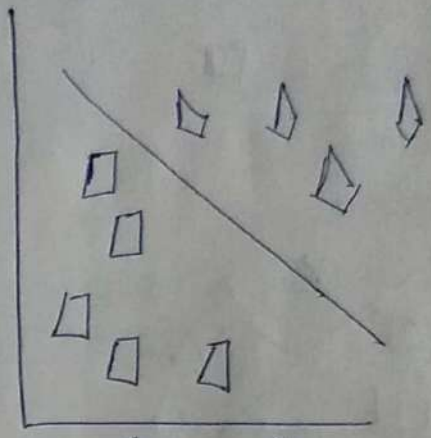
- High Gamma
- low Gamma

Low Gamma → Point are far away from Seperation line

High Gamma → Point are close from Seperation line



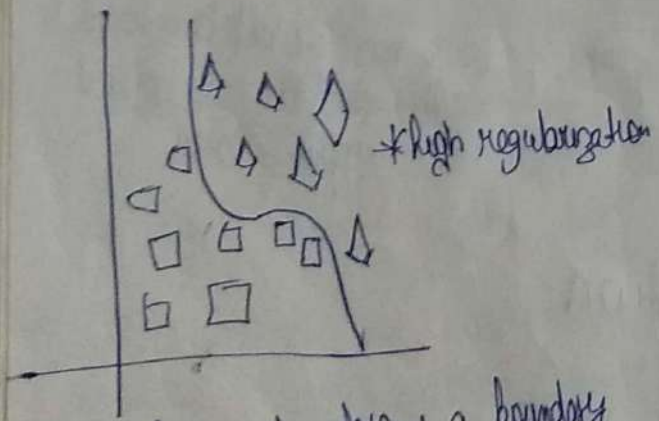
High gama



low Gamma

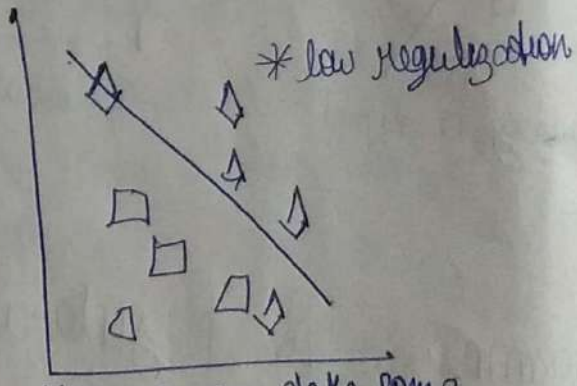
Regularization

The Regularization parameter tells the SVM optimization how much you want to avoid misclassifying each training example.



* High regularization

Here I try to draw a boundary very carefully to avoid any classification error (overfitting of model)
This line can be very zigzag

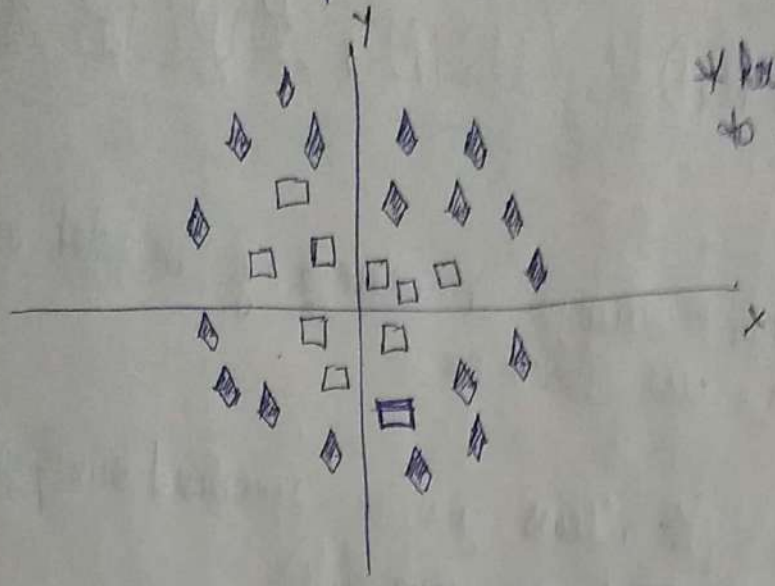


* low regularization

Here we can take some error. Our line looks more smoother

C means Regularization

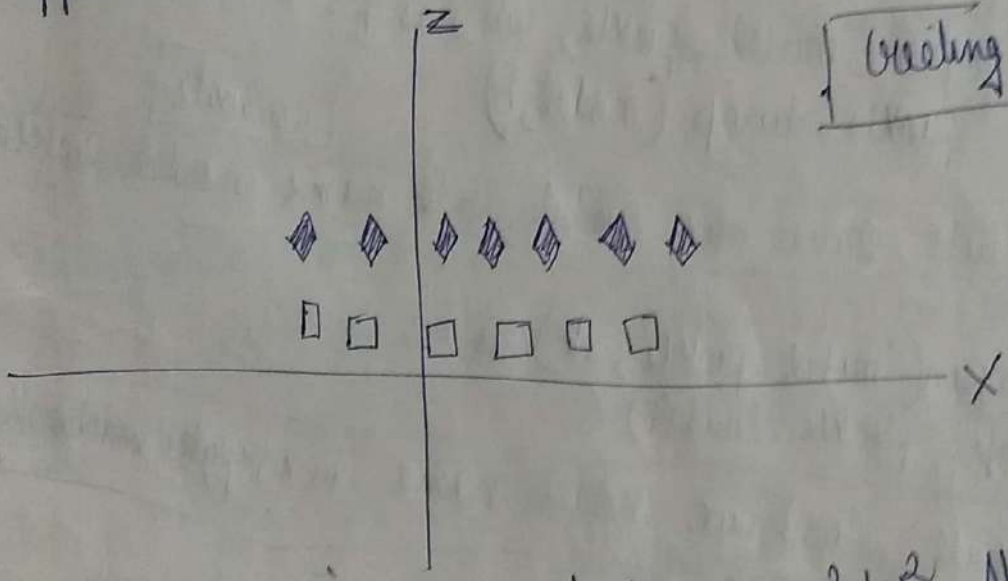
We might have a complex data set like this



* Here it is not easy to draw the boundary

One approach is to create a third dimension

Creating z-dimension



Here, we will add a new feature $z = x^2 + y^2$. Now, let's plot the data points on axis x and z

Points to be considered

- All values for z would be positive always because z is the squared sum of both x & y
- □ appear close to the origin of x & y axes
- This transformation is called Kernel.

* Kernel is used so that we can draw the decision boundary easily

Imp. SAVE MODEL USING JOBLIB AND PICKLE

It helps in saving a changing model to a file which we can use later on

Ans) import pickle (It allows you to serialized our python object into your file)

Ans | with open ('model_pickle', 'wb') as f:
pickle.dump (model, f)

wb-write

↑ This will save our model with name model_pickle

Ans | with open('model_pickle', 'rb') as f:
mp = pickle.load(f)

This will load our model name to jupyter notebook

Ans) mp.predict (5000)

Now we can predict with model mp

Using Joblib

In [ ] from sklearn.externals import joblib (importing)

In [ ] joblib.dump(model, 'model_joblib') (dumping)

In [ ] mj = joblib.load('model_joblib') (loading)

In [ ] mj.predict(5000) (predicting)

5) NAIVE BAYES CLASSIFIER

* Naive Bayes is a simple but surprisingly powerful algorithm for predictive modeling and it is a classification technique.
* It is not a single algorithm but a family of algorithms where all of them have a common principle i.e every pair of features being classified is independent to each other. Even if each other depends upon the existence of the other features. that's why it is called naive

Hypothesis: An example of a model that approximates the target function and performs mapping of inputs to outputs are called hypothesis in ML.
In ML x denotes input & Y denotes output
Relation b/w input & output is $y = f(x)$

BAYES Theorem

It describes the probability of an event based on prior knowledge of the conditions that might be related to the event. It is a way to figure out conditional probability.

Conditional probability is the probability of an event happening given that it has some relationship to one or more other events for example: Your probability of getting a parking space is connected to the time of the day you park, where you park and what conditions are you going on at that time.

Defination :-

Given hypothesis H and evidence E, Bayes theorem states that the relationship b/w the probability of the hypothesis before getting the evidence P(H) and the probability of the hypothesis after getting the evidence. P(H|E) is

$$P(H|E) = \frac{P(E|H) \cdot P(H)}{P(E)}$$

Bayes theorem Example

52 cards

probabilut of getting a King is $4/52 = 1/13$

If the evidence is provided for instance someone looks at the single ~~card~~ card is face card

Since every King is also a face card the probability of face given that it's a King is equal 0

P(Face|King) = 1

P(Face) = 12/52 = 3/13

$$= \frac{P(F|K) \cdot P(K)}{P(Face)}$$

$$= \frac{1 \cdot (1/3)}{3/13} = 1/3$$

BAYES Theorem Proof

P(A|B) = Probability of A given B

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

$$P(B|A) = \frac{P(B \cap A)}{P(A)}$$: Probability of B given A

$P(A \cap B)$ = Probability of A intersect B

$$P(A \cap B) = P(A|B) \cdot P(A) = P(B|A) \cdot P(B)$$

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

Likelihood
How probable is the evidence given that our hypothesis is true?

Prior
How probable was our hypothesis before observing the evidence

$$P(H|E) \quad \frac{P(E|H) \cdot P(H)}{P(E)}$$

Posterior
How probable is our hypothesis given that observed evidence? (Not directly computed)

Marginal
How probable is the new evidence under all possible hypotheses?

NAIVE BAYES : WORKING

Classification Steps

Data Set

Day	Outlook	Humidity	Wind	Play
D1	Sunny	High	Weak	No
D2	"	"	Strong	No
D3	Overcast	"	Weak	Yes
D4	Rain	"	"	Yes
D5	"	Normal	"	Yes
D6	"	"	Strong	No
:	:	:	:	:
D14	"	High	"	No

First of all we'll create a frequency table using each attribute of the dataset

Frequency table for Outlook

F. table		Play	
		Yes	No
Outlook	Sunny	3	2
	Overcast	4	0
	Rainy	3	2

Frequency table for Humidity

Frequency table		Play	
		Yes	No
Humidity	High	3	4
	Normal	6	1

Frequency table from Wind

Frequency table		Play	
		Yes	No
Wind	Strong	6	2
	Weak	3	3

Now for each table we'll generate a likelihood table now.

Likelihood table contains the probability of a particular day suppose we take the Sunny and we take the play as yes and no. So the probability of Sunny given that we play yes is 3/10.

$\therefore P(x|c) = P(\text{Sunny}|\text{Yes}) = 3/10 = 0.3$

Likelihood Table		Play		
		Yes	No	
Outlook	Sunny	3/10	2/4	5/14
	Overcast	4/10	0/4	4/14
	Rainy	3/10	2/4	5/14
		10/14	4/14	

$P(x) = P(\text{Sunny}) = 5/14 = 0.36$

$P(c) = P(\text{Yes}) = 10/14 = 0.71$

Likelihood of 'Yes' given Sunny is

$P(c|x) = P(\text{Yes}/\text{Sunny}) = P(\text{Sunny}|\text{Yes}) * P(\text{Yes})/P(\text{Sunny})$

$= (0.3 \times 0.71)/0.36 = 0.591$

Similarly likelihood of 'No' given Sunny is

$P(c|x) = P(\text{No}/\text{Sunny}) = P(\text{Sunny}|\text{No}) * P(\text{No})/P(\text{Sunny})$

$= (0.4 \times 0.36)/0.36 = 0.40$

Likelihood table for Humidity

Likelihood Table		Play		
		Yes	No	
Humidity	High	3/9	4/5	7/14
	Normal	6/9	1/5	7/14

$P(\text{Yes}/\text{High}) = 0.33 \times 0.6/0.5 = 0.42$

$P(\text{No}/\text{High}) = 0.8 \times 0.36/0.5 = 0.58$

Likelihood table for Wind

Likelihood Table		Play		
		Yes	No	
Wind	Weak	6/9	2/5	8/14
	Strong	3/9	3/5	6/14
		9/14	5/14	

$P(Yes/Weak) = 0.67 \times 0.64 / 0.57 = 0.75$

$P(No/Weak) = 0.4 \times 0.36 / 0.57 = 0.25$

Suppose we have a day with the following values

Outlook = Rain

Humidity = High

Wind = Weak

Play = ?

Likelihood of 'Yes' on that day = P(Outlook = Rain/Yes) * P(Humidity = High/Yes) * P(Wind = Weak/Yes) * P(Yes)

$= 2/9 * 3/9 * 6/9 * 9/14 = 0.0199$

Likelihood of 'No' on that Day = P(Outlook = Rain/No) * P(Humidity = High/Yes) * P(Wind = Weak/Yes) * P(Yes)

$= 2/5 * 4/5 * 2/5 * 5/14 = 0.0166$

$P(Yes) = 0.0199 / (0.0199 + 0.0166) = 0.55$

$P(No) = 0.0166 / (0.0199 + 0.0166) = 0.45$

Our model predict that there is 55% chance there will be game tomorrow

Applications of Naive Bayes algorithms

1. Real time prediction
2. Multi class prediction
3. Text classification/Spam Filtering/ Sentiment Analysis
4. Recommendation System

TYPES

There are three types of Naive Bayes model under Scikit learn library.

1. Gaussian: It is used in classification and it assumes that features follow a normal distribution.
2. Multinomial = It is used for discrete counts. For example we have a text classification problem. Here we can consider bernoulli trials which is one step further and instead of word occuring in the document" we have "count how often word occurs in the document", you can think of it as "number of times outcome number x-i is observed over the n trials
3. Bernoulli: The binomial model is useful if your feature vector are binary (i.e zeros and ones).

One application would be text classification with bag of words model the 1's & 0's are "word occurs in the document" and "word does not occur in the document" respectively.

```
In | from Sklearn import data datasets
     from Sklearn.naive_bayes import GaussianNB

In | dataset = datasets.load_iris()

In | model = GaussianNB()

In | model.fit(dataset.dat, dataset.target)
```

Rest is Same

⑥ KNN (Supervised machine learning)

What is KNN algorithm?

K Nearest Neighbour is a simple algorithm that stores all the available case and classifies the new data or case based on a similarity measure.

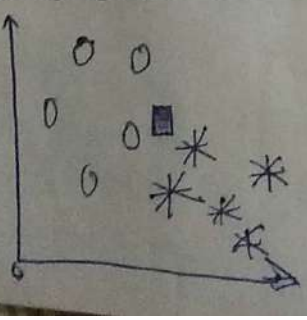
Explanation:-

* It suggest that if you are similar to your neighbours that means you are one of them
Example → If apple look more similar to banana or orange rather than monkey, rat or cat. So most likely apple belongs to the group of fruits.
This is what nearest neighbour means

So what is K in the KNN algorithm?
K denotes the number of Nearest Neighbour

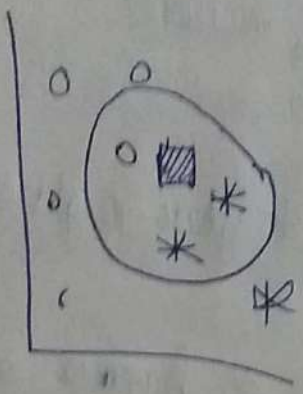
The Biggest use of KNN is Recommender System for amazon or any other company it doesn't even shown you the product but it also suggest us or find the relevent match

How does a KNN algorithm work?



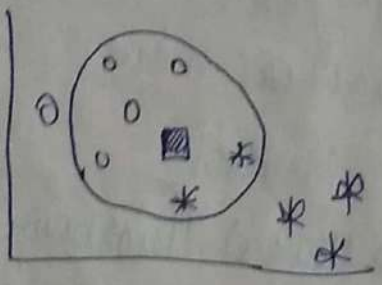
Now we have a new point (■) on our graph so how to tell that this new point belongs to which class

First we have to select the value of K
Lets take an example of K=3



So here we 3 points near to nearest neighbour
Here we have 2 * and one o
So it belongs to * class

what if we select K=6



first we have to select 6 points nearest to ■
Here we have four o and two *
So, ■ belongs to o class

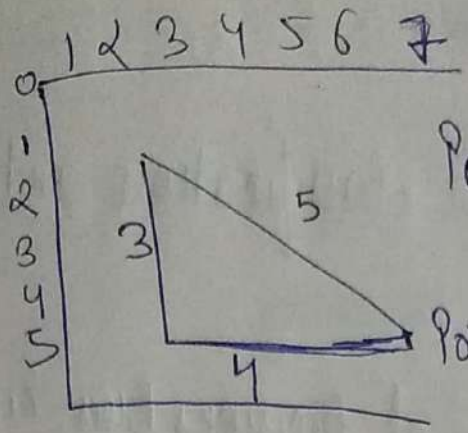
So How to choose the value of K in KNN algorithm

For 1000 n value K should be around 1 to 19 but it actually depends on dataset.

How things are predicted?

We know that we select the nearest points for new points. But how to calculate which are nearest point. Here we generally use 2 methods:-

1) Euclidean Distance



Point P1 = (1,1)

Point P(2) = (5,4)

Euclidean distance = $\sqrt{(5-1)^2+(4-1)^2} = 5$

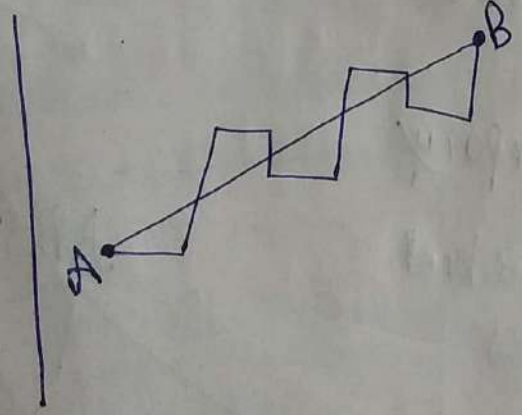
It is define as Square Root of the Sum of difference b/w new point x & existing point Y

② Manhattan distance

For the same graph

Manhattan distance = $|5-1| + |4-1| = 7$

Manhattan Distance Vs Euclidean Distance



Straight line is Euclidean distance

And zigzag line is Manhattan distance

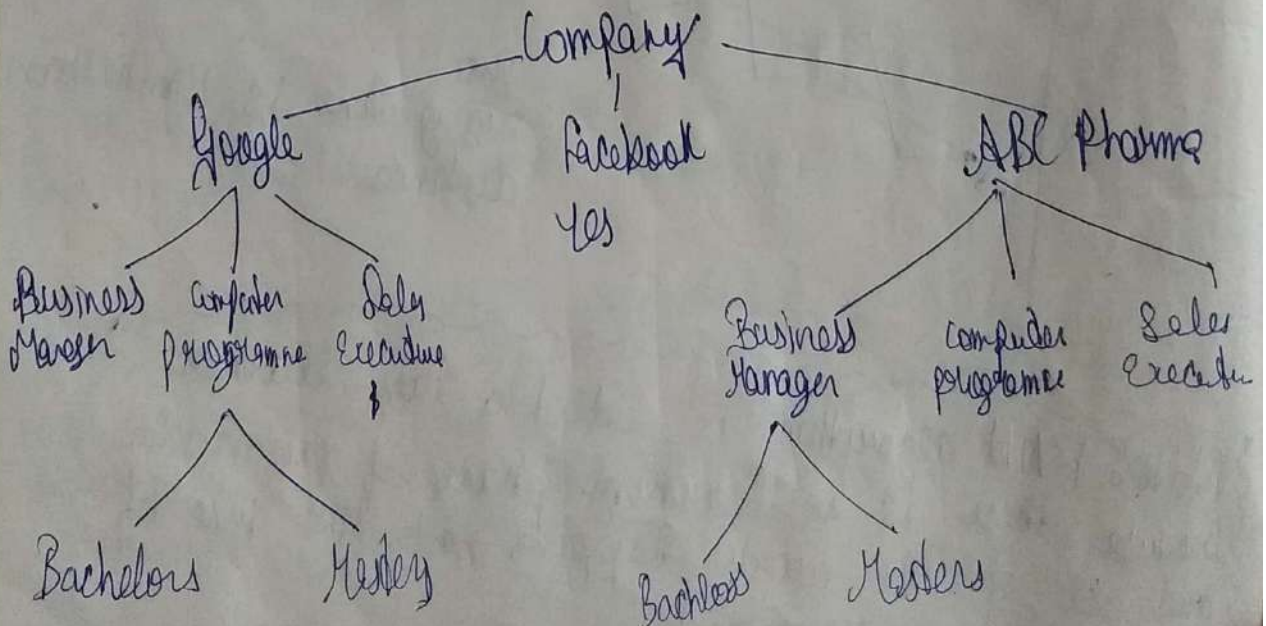
Note: KNN algorithm is called a lazy learner because there is no learning phase of the model and all of the work happens at the time of prediction is requested

7. Random Forest

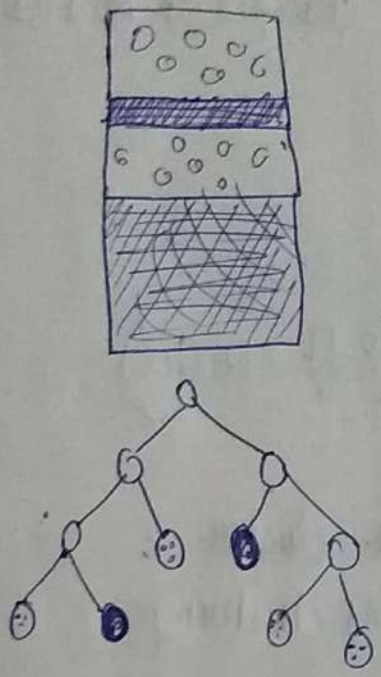
* It can used for both classification and regression tasks.
* Random Forest is a collection of Decisions Trees, but there are some differences
* The higher the number of trees in the forest gives the high accuracy results.
* Random forest classifier will handle the missing values
* When we have more trees in the forest, random forest classifier won't overfit the model

Here we are using same data-set ~~for~~ used for decision tree.

For that dataset we have decision tree like this



But we can also represent this in a very simple image.



Here An Random Forest we take multiple subset of dataset of maybe different size. They we build decision tree for each of them called Random Forest.

8. K-Means Clustering (Unsupervised)

The method of identifying similar groups of data in a dataset is called Clustering

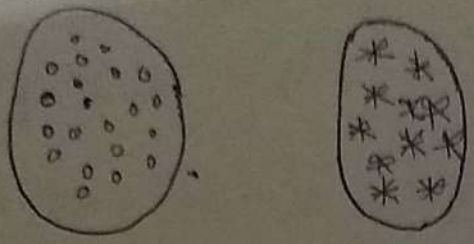
TYPES OF CLUSTERING

1. Exclusive clustering
2. Overlapping ~~Overclustering~~
3. Hierarchical clustering

① Exclusive Clustering (Hard clustering)

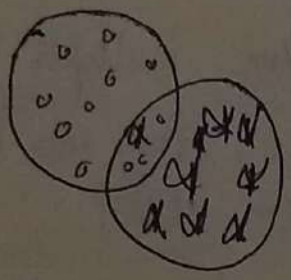
Item belongs exclusively to one cluster

example - K-Means clustering



② Overlapping Clustering (Soft Cluster)

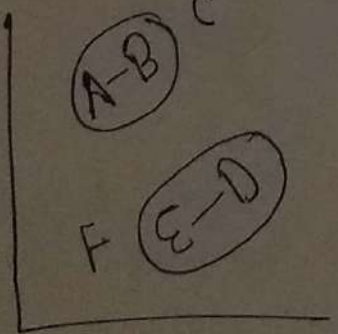
* Soft Cluster
* Item belongs to multiple cluster :
* For example : Fuzzy/C means clustering



③ Hierarchical Clustering

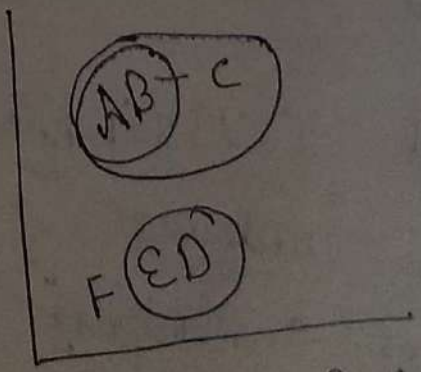
* Suppose we have 6 data points out of which A & B are similar and E & D are similar

Case 1



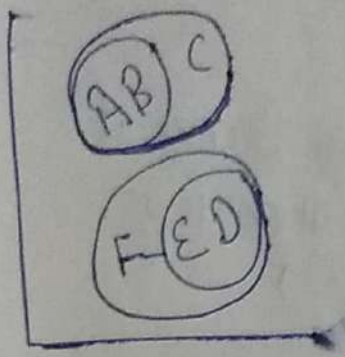
* Combine A & B based on Similarity
* Combine D & E based on Similarity.

Case 2



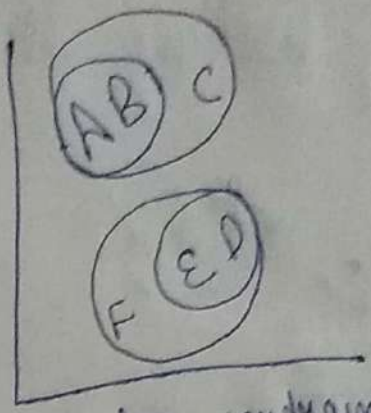
* Combination of A & B is combined with C.

Case 3



→ Combination of D & E is combined with F

Case 4



* Final tree contains all clusters combined into a single cluster.

Note: 'K' in K-Means represents the number of clusters

K means algorithm can be applied to numerical and continuous data with smaller number of dimension. It can be applied to anything where we want to predict from which group this new point is

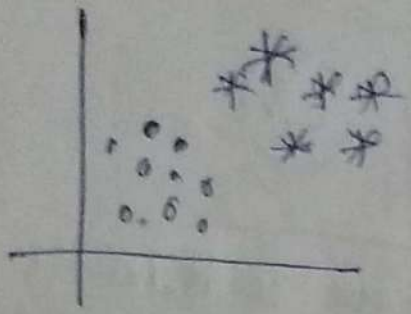
~~* K Means~~ hunt

* Here, K means number of cluster that we want. If K=3, then total number of cluster is also 3

A centroid is a data point (imaginary or real) at the center of a cluster.

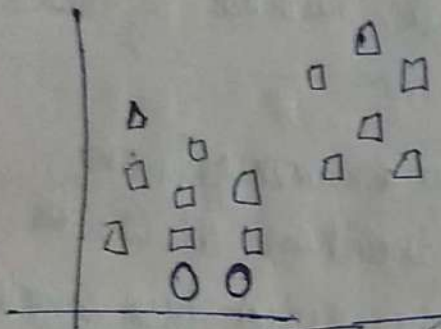
How it works?

We have tell the algorithm about the value of K



Here just by looking at it we can say that value K=2

Step 1: Start with K centroids by putting them at random place. Here K=2



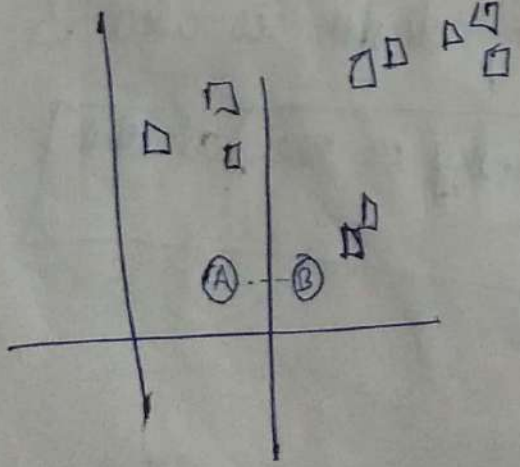
Putting 2 random points in graph

O is the random point

Random point can be place anywhere

If our K is 3 then we have to put 3 random points

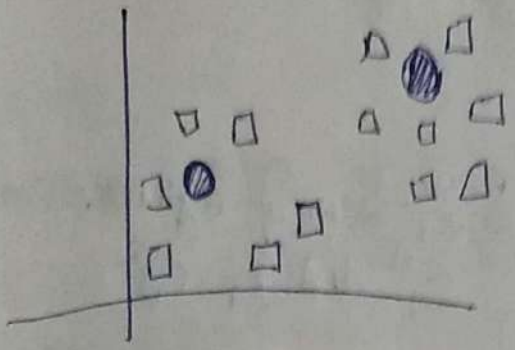
Step 2: To identify the distance of each of these data points (Random point) draw a perpendicular line on equal distance between them Left hand side will be of class A and Right hand side will be of class B



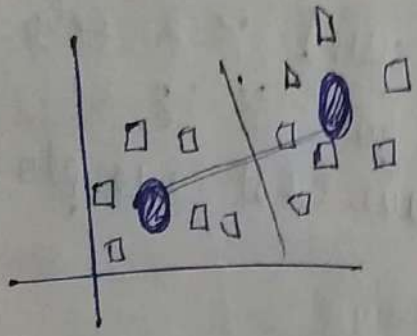
If we then we try to improve

Step 3: Adjust centroids so that they become center of gravity for given cluster

So in this step we assign our random points at the centroid of the group like this



Step 4: We repeat this process again. We draw a perpendicular line between



So left side is class A
Right side is class B

we continue this process until the point that non of the data point change its group.

What is the good number for K? (because in reality we have large number of K)

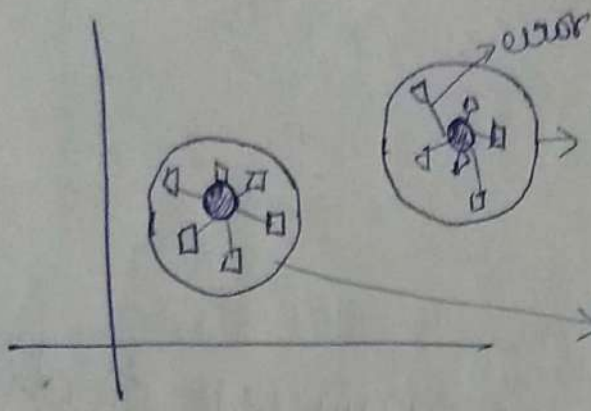
There is a technique called elbow method we have started with just 2 cluster but someone can say that No it has 4 cluster or 6.

Now our job is to find the best value of K

How elbow method works?

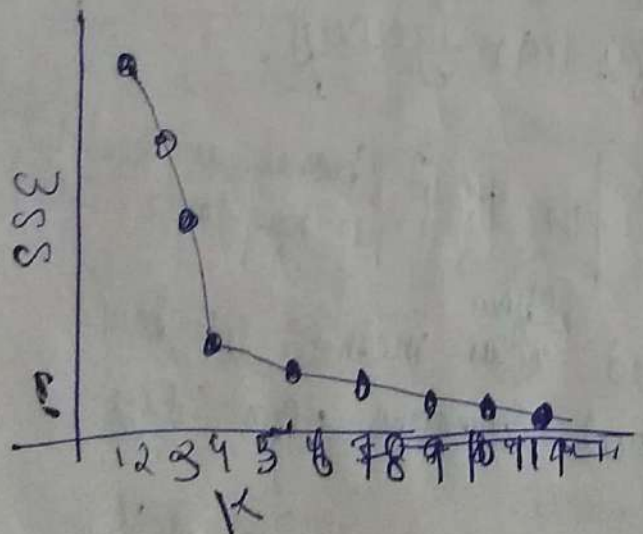
(1) Let start with some K (lets take it=2) and we try to compute some off square error

what it means is for each of the clusters you try to compute the distance of individual data points from the centroid (for all the groups) differently if I then we square it

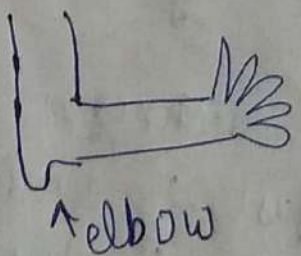


$$\text{Total Sum of Square Error (SSE)} = SSE_1 + SSE_2 + SSE_K$$

we just did this example using $K=2$ only same thing we have to do for $K=1, K=2, K=3$ -- and so on. and once have that we plot them into graph. like this



on this the chart elbow is at 4



So our good K number is 4

9. DIMENSION REDUCTION TECHNIQUES

Suppose we have a dataset which have 256 columns So, how will you compute?

Let us take case of motorbike rider in racing competitions. Today, his position and movement get measured by GPS sensor on bike, gyro meters, multiple video feeds and his smart watch Because of respective errors in recording, the data would not be exactly same. However, there is very little incremental information on position gained from putting these additional sources. Now assume than an analyst sits with all this data to analyse the racing strategy of the biker. - He/she would have a lot of variables/ dimensions which are similar and of little (or no) incremental value. This is the problem of high unwanted dimensions and needs a treatment of dimension reduction.

Dimension Reduction refers to the process of converting a set of data having vast dimensions into data with lesser dimensions ensuring that it conveys similar information concisely. These techniques are typically used while solving machine learning problems to obtain better features Classification

Benefits :-

1. It helps in data compressing and reducing the storage space.
2. It fastens the time
3. Increases the model performance

Note: Dimensionality reduction algorithm helps us along with various other algorithms like decision tree, Random forest, PCA & others

For this we use PCA. Please PCA is not a algorithm it is a technique for dimentionality Reduction

PCA (Principal Component Analysis)

From Jupyter Notebook

10. GRADIENT BOOSTING ALGORITHMS

1. AdaBoost
2. XGBoost
3. light GBM
4. CatBoost-

① Ada Boost

Where are Boosted algorithms required?

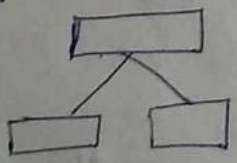
* They are used where we have plenty of data to make a prediction and where we seek exptionally high predictive power.
* It is used for reducing bias and variance. It combines multiple weak predictors to ~~bu~~ build strong predictor

> Bias: refers to tendency of a measurement process to over- or under-estimate the value of a population parameter.

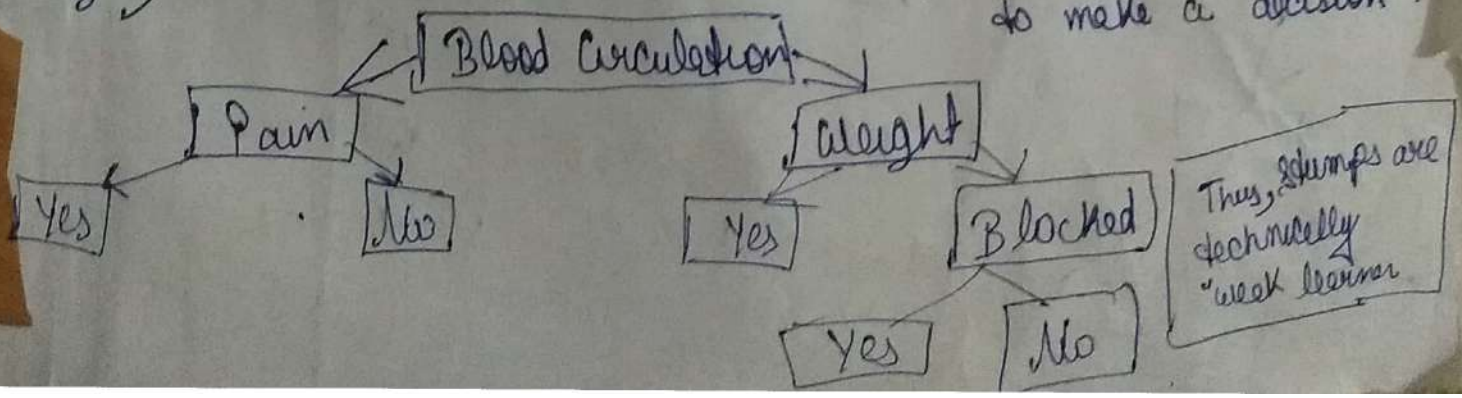
> Variance: It measures how far a data set is spread out

* AdaBoost is a Boosting done on Decision Stumps.

> Stump: A tree with just one node and two leaves is called Stump.



* Stumps are not great at making accurate ~~data~~ classification. A full sized Decision Tree would take advantage of all 4 variables that we measured (chest pain, Blood Circulation, Blocked Arteries and Weight) to make a decision but a Stump can only use one variable to make a decision.



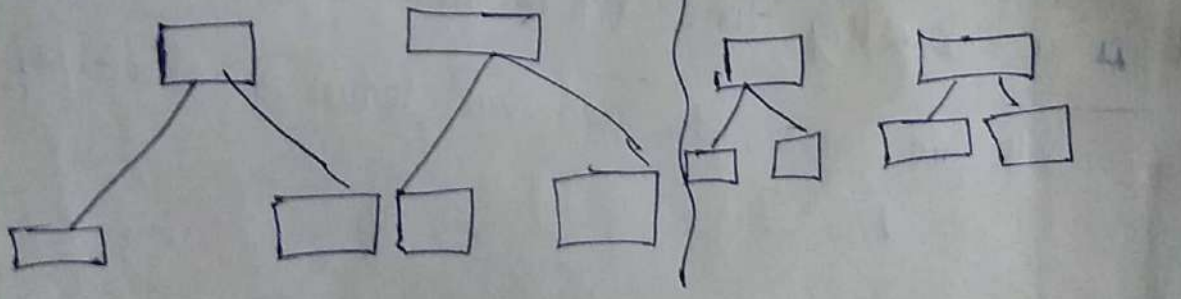
However, that's the way AdaBoost likes it, and it's one of the reasons why they are so commonly combined

Note: In Random Forest, each tree has an equal vote on the final classification.

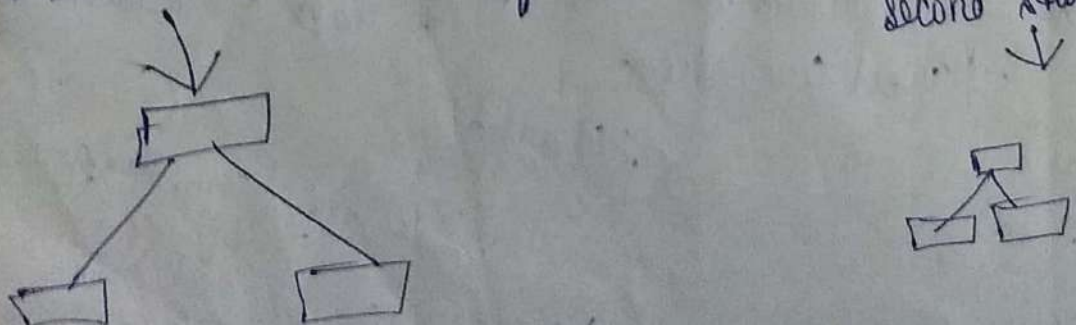
* But in a Forest of Stumps made with AdaBoost, some Stumps get some say in the final classification than others.

In this illustration, the larger Stumps get more say in the final classification than the smaller Stumps.

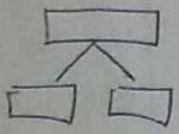
larger Stumps | Smaller Stumps

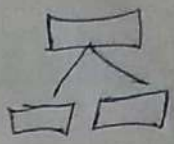


* In a Forest of Stumps made with AdaBoost order is important.
* The error that the first Stump makes influence how the second Stump is made



and the errors that the second stump makes influence how the third stump is made.





The three ideas behind Adaboost are:

① AdaBoost combines a lot of "weak learners" to make classifications. The weak learners are almost always stumps.

② Some stumps get more say in the classifications than others

③ Each stump is made by taking the previous stump's mistake into account

word meaning
nitty-gritty = the most important aspects or practical details of a subject or

* The idea of boosting came out of the idea of whether a weak learner can be modified to become better.

Now let's dive into the nitty gritty detail of how to create a Forest of Stumps using AdaBoost

Suppose we have this dataset

Chest Pain	Blocked Arteries	Patient Weight	Heart Disease	Sample Weight
Yes	Yes	205	Yes	1/8
No	Yes	180	Yes	1/8
Yes	No	210	Yes	1/8
Yes	Yes	167	Yes	1/8
No	Yes	156	No	1/8
No	Yes	125	No	1/8
Yes	No	168	No	1/8
Yes	Yes	172	No	1/8

* We create a forest of stumps with adaboost to predict if a patient has heart disease. We will make these predictions based on patient's Chest Pain and Blocked Artery Status and their weight.

* The first thing we do is give each sample a weight that indicates how important it is to be correctly classified —

* At the start, all samples get the same weight

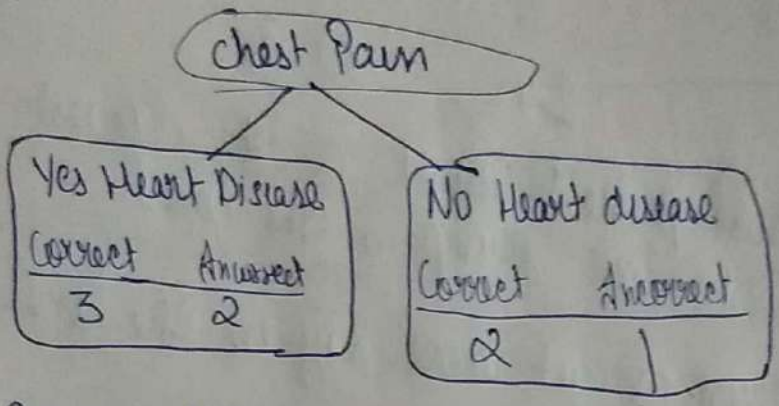
$$= \frac{1}{\text{total number of samples}} = \frac{1}{8}$$

and that makes the samples all equally important

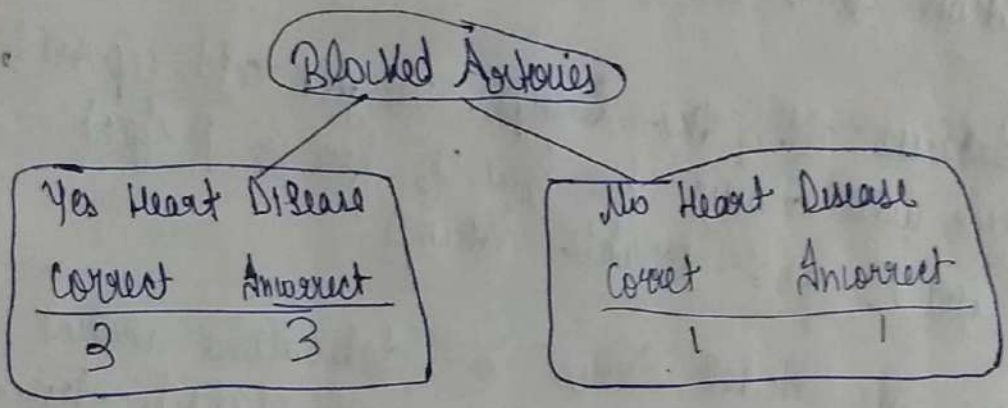
After we make the first stump, these weights will change in order to guide how the next stump is created

* Now we need do make the first stump on the tree. this is done finding the variable, chest pain, Blocked Arteries or Patient weight, that does the best job classifying the samples

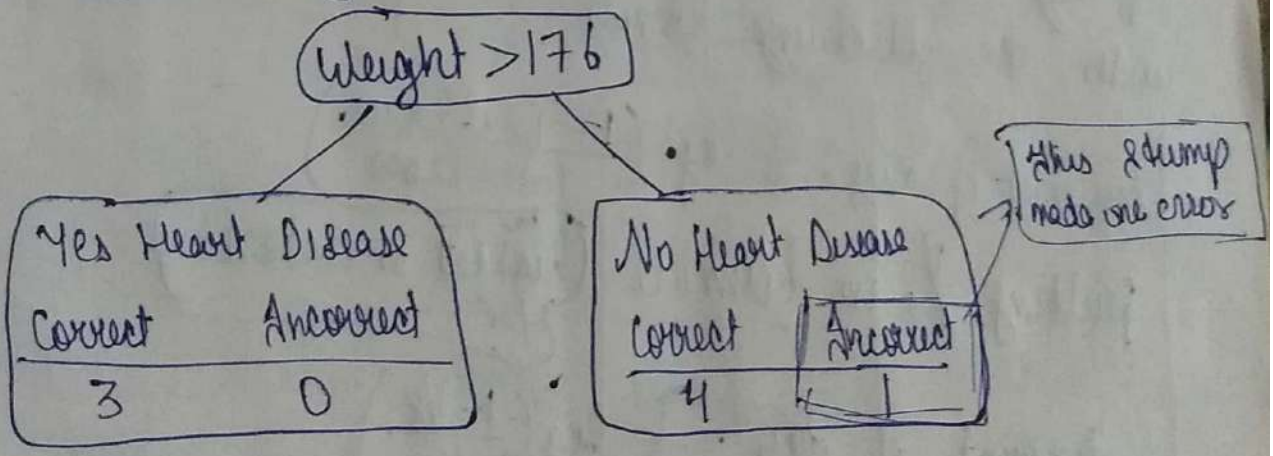
We start by seeing how well chest pain classifies the samples



For Blocked Arteries.



For Patient weight :



Note: We used the techniques described in the Decision Tree to determine that 176 was the best weight to separate the patients

* Now we calculate the Gini Index for the three Stumps.

For chest pain → gini Index → 0.47

Blocked Arteries → 0.5

Weight > 176 → 0.2

The Gini Index for patient weight is the lowest so this will be the first Stump in the forest.

Remember, some Stumps get more say in the final classification than others.

Note: Because all of the sample weights add up to 1, Total Error will always be b/w 0, for a Perfect Stump, and 1, for Horrible Stump

* We use the total Error to determine amount of say this Stump has in the final classification with the following formula

$$\text{Amount of Say} = \frac{1}{2}\log\left(\frac{1 - \text{Total Error}}{\text{Total Error}}\right)$$

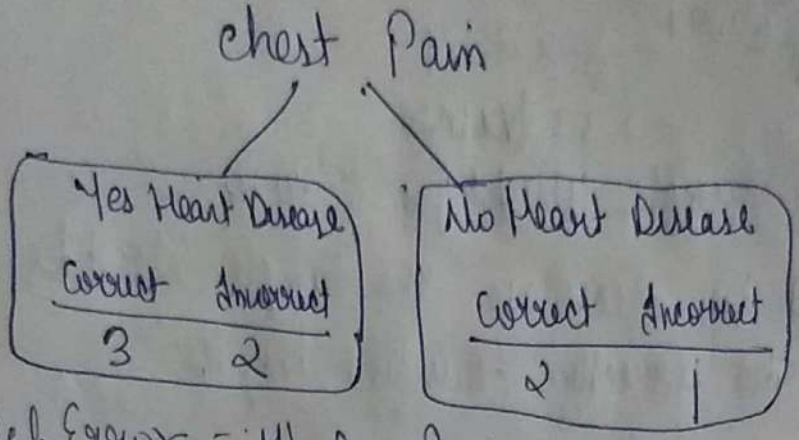
Putting 1/8 in formula (for misclassified in dataset)

$$\text{Amount of say} = \frac{1}{2}\log\left(\frac{1-1/8}{1/8}\right)$$

$$= \frac{1}{2}\log(7)$$

$$= 0.97$$

* Lets work out how much say the best pair stump would have had if it had been the best stump



Total Error = The sum of the weights for the incorrectly classified samples

Note: These incorrect value are for 2nd row, 7th row & 8th row.

Total Error = $\frac{1}{8} + \frac{1}{8} + \frac{1}{8} = \frac{3}{8}$

Amount of say = $\frac{1}{2} \log\left(\frac{1 - 3/8}{3/8}\right)$

$= \frac{1}{2} \log(7/3)$

$= 0.42$

Same for Blocked arteries stump as an exercise for the reader.

New Sample Weight = Sample weight $\times e^{\text{amount of say}}$

* Here we use this formula to increase the sample weight

* for weight

New Sample weight = Sample weight × $e^{\text{amount of say}}$

$= \frac{1}{8} e^{\text{amount of say}}$

$= \frac{1}{8} e^{0.97} = \frac{1}{8} \times 2.64$

$= 0.33$

0.33 is new Sample weight for 3rd row

* Now we need to decrease the Sample weights for all of the correctly classified samples

* This is the formula do decrease the Sample weight

= Sample weight × $e^{-\text{amount of say}}$ (-ve sign)

$= \frac{1}{8} e^{-0.97} = \frac{1}{8} \times 0.38 = 0.05$

The new Sample is 0.05 which is less than the old one 1/8 = 0.125

which is less than the old one

New weight	Normalized weight
0.05	0.07
0.05	0.07
0.05	0.07
0.33	0.49
0.05	0.07
0.05	0.07
0.05	0.07
0.05	0.07
0.05	0.07

→ Right now, if you add up the New Sample weights, you get 0.68

→ So we divide each new Sample weight by 0.68 to get the normalized values.

when we add the new sample weights, we get 1 (plus or minus a little round off error)

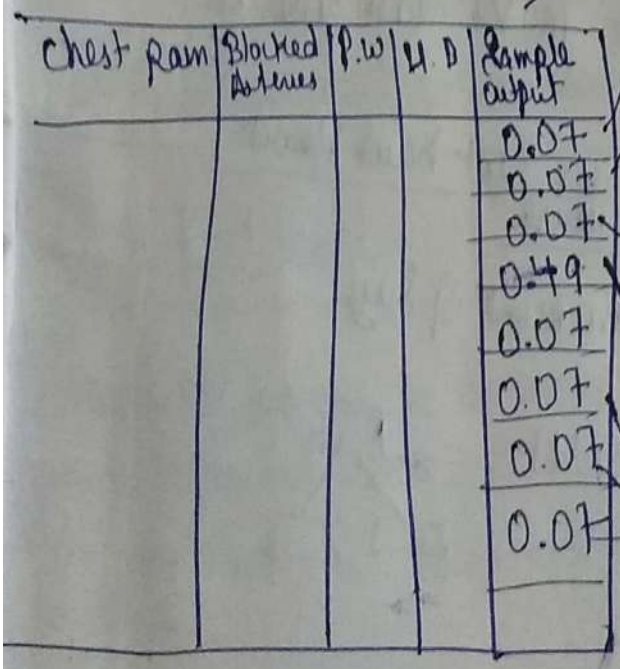
* Now we can use the modified sample weights to make the second stump in the forest.

Now, we start by making a new, but empty, dataset that is the same size as the original.

② Then we pick a random number between 0 and 1

③ Then we see where that number falls when we use the sample weights like a distribution

~~④ Imagine I pick random number 0.72 Then I'll pick 4th row where sample weight of 0.49 is in the new data set~~



If number is b/w 0 & 0.07. then we would put this sample into new dataset and if the number is b/w 0.07 & 0.14 (0.07+0.07=0.14), then we would put this sample into the new collection of samples if the no. is b/w 0.14 & 0.21 (0.14+0.07=0.21), then we'll put this if the no. is b/w 0.21 & 0.70 (0.21+0.49=0.70) then we put this sample & so on

For example, imagine the first number I picked was 0.72 then i'll put 5th row in new dataset. & then I picked 0.42 then i'll put 4th row on dataset & so on until we get the new collection is the same size as the original

A row can repeated in the new dataset

Ultimately, this sample was added to the new collection of samples 4 times, reflecting its larger sample weight

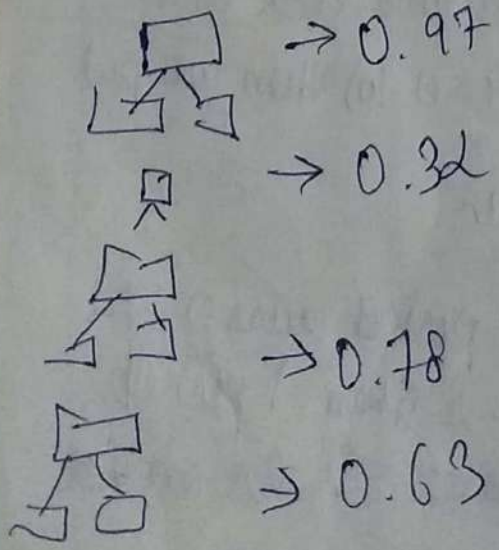
Now, we get rid of the original samples & use the new collection of samples

So that is how the errors that the first tree makes influence how the second tree is makes and so on.

Now we need to talk about how a forest of stumps created by Adaboost make classifications

Has Heart Disease

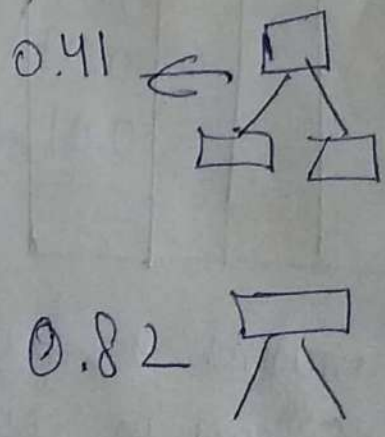
Amount of Say



Total 2.7

Does Not have Heart Disease

Amount of Say



Total = 1.23

Ultimately, the patient is classified as Has heart Disease because this is the larger sum.

Code:

An | from sklearn.ensemble import AdaBoostClassifier

An | adb = AdaBoostClassifier(DecisionTreeClassifier(), n_estimators=10, learning_rate=1)
adb.fit(x_train, y_train)

An | adb.score(x_train, y_train)

n_estimators = On 10 decisiontree
learning_rate = contribution to each individual learner to 1

It generally overfits the model

② XGBoost

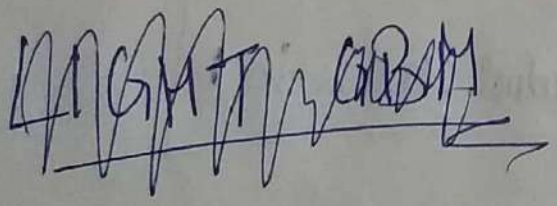
* XGBoost belongs to a family of Boosting algorithms that converts weak learners into strong learners
* It can used to solve both regression and classification problems.
* It don't overfits the model.
* XGBoost uses a few computational tricks that exploit a computer's hardware to speed up gradient descent.

* Its same as Adaboost but XGBoost has several Tuning parameters

| See parameter online |

* Code

```
In| from xgboost import XGBclassifier
In| model = XGBclassifier()
In| model.fit(x_train, y_train)
In| y_pred = model.predict(x_test)
```



IMPORTANT MACHINE LEARNING INTERVIEW QESTIONS

Q] What is Data Mining

Ans] Data Mining can be defined as the process in which we try to extract knowledge from unstructured data

Q What is Overfitting and Underfitting in Machine learning

Ans] In this will discover the concept of Generalization in machine learning and the problems of overfitting and underfitting that go along with it.

GENERALIZATION IN M.L.

Generalization refers to how well the concepts learned by a machine learning model apply to specific examples not seen by the model when it was learning. This allows us to make predictions in the future on data the model has never seen.

The goal of good Machine learning model is to Generalize well from the training data to any data from the problem domain.

There is a terminology used in ML when we talk about how well a machine learning model learns & generalizes to new data - namely overfitting and underfitting

OVERFITTING

Overfitting refers to a model that models the training data too well.

Overfitting happens when a model learns the details and noise in the training data to the extent that it negatively impacts the performance of the model on new data. This means that the noise or random fluctuation in the training data is picked up & learned as concepts by the model.

noise : Noisy data is meaningless data.
: Noisy data is data that is corrupted or distorted
Data = true Signal + noise

UNDERFITTING

An underfit machine learning model is not a suitable model and will be obvious as it will have poor performance on the training data.

A GOOD FIT IN ML.

Ideally, you want to select a model at the sweet spot between underfitting & overfitting

Over time, as the algorithm learns, the error for the model on the training data goes down and so does the error on the test dataset.

The sweet spot is the point just before the error

There are two important techniques that you can use when evaluating ML algorithm to limit overfitting

1. Resampling
2. Cross-Validation Dataset

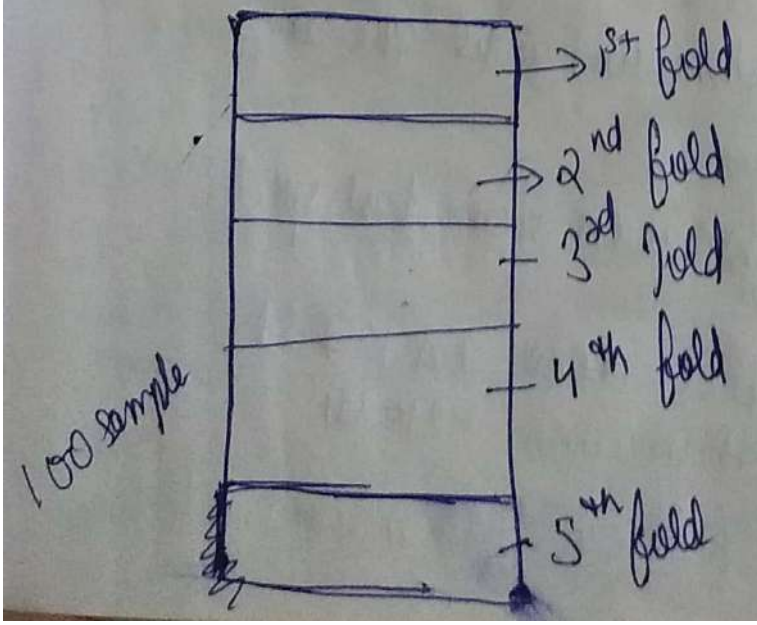
RESAMPLING

The most popular resampling technique is K-fold cross validation.

K-FOLD CROSS VALIDATION

Sometimes we get into the dilemma of which ML model should I use to solve my problem for eg. K-FOLD is a technique which allows you to answer the exact same question.

In this technique what we do is we divide our 100 samples into folds. So I have 5 folds here.



& each fold contains 20 samples then we run multiple iteration.

This technique is good because of we are giving variety of sample to you model then we are taking individual score and then averaging them out

From Jupyter Notebook

CROSS-
VALIDATION DATASET

A validation dataset is simply a subset of your training data that you hold back from your machine learning algorithm until the very end of project.

Then a differencence between train_test_split and cross-validation.

Cross validation is a test done in K-fold
It will give us more than one result
because in K-fold we fold our data n times

Using Cross Validation is a gold standard in applied M.L for estimating model accuracy on unseen data

Summary

* Overfitting :- Good performance on the training data, poor generalization to other data.
* Underfitting :- Poor performance on the training data and poor generalization to other data.

Q) What is inductive learning & deductive learning?

Ans) Inductive learning

Suppose that you are a kid and I'm your father. It my job to make you to understand that playing fire or ~~heat~~ heat can cause burn. So I will make you to understand in such a way that, I will show you some training examples of burnt people, or fire accidents or something like that with a lable 'dangerous'. So you will learn with that examples and will not try to play with fire. This method is called inductive learning

Deductive learning

The other method is ~~this~~ much easier for me to make you learn about burns but may difficult for you. The method is that I will let you to play with fire and just wait to see what happens. If you get burn, you will learn don't play with fire and when you see fire, you ~~will~~ remember it ~~with~~ with your incident and will avoid that. This is deductive learning

In General,

Deductive learning = conclusion → observation

Inductive learning = observation → conclusion

Q What are the different algorithm techniques in machine learning?

There are 3 types of Machine learning algorithm

① Supervised learning – do

② Unsupervised learning – do

③ Reinforcement learning.

- Reinforcement learning

In this machine is trained to make specific decisions. It works this way: the machine is exposed to an environment where it itself continually using trail and error.

This machine learns from past experience and tries to capture the best possible knowledge to make accurate business decision. Example: Markov Decision Process.

What is trail and error?

The trial and error method is mainly used, where we dont computers to give us a known answer, we first enter value and compare them with the correct answer they may computers can gain experience so that they avoid

previous mistakes and adjust their answer for the next value so they would improve the results gradually until they give a precise answer.

What is trial and error

The process of making repeated trials or test, improving the methods used in the light of error made, until the right result is found.

Q) What are the three stages to build the hypo-theses or model in ML?

Ans) a) Model building.
b) Model testing.
c) Applying the model.

Q) What is the standard approach to supervised learning?

Ans) The standard approch is to split the set of example into the training set and the test.

Q) What is algorithm independent machine learning?

Ans) Machine learning in where mathematical foundations is independent of any particular classifier or learning algorithm is referred as algorithm independent ML.

Q/ What is Genetic programming?

Ans/ Genetic algorithms are inspired by nature & evolution, evolution is the best general-purpose learning algorithm we've experienced, and the brain is the best general-purpose problem solver we know.

The model is based on the testing & selecting the best choice among a set of results.

Q/ What is the different b/w heuristic for rule learning & heuristic for decision tree

Sol/ A heuristic is any practical approach to solving a problem (which is not optimal)

* Heuristic is a guided search

* Heuristic is a method that might not always find the best solution But it guranted to find a good solution in reasonable time.

* Example :- To save time, I say all the boys are taller than girls. This may be true in some cases but not all the time. So how can I improve this model? I collect data from school, introduce certain variable, for eg age, ethnicity, weight, etc, then learn a ML model, tune its parameter. In this example, if you use

simple heuristic and you perform badly, it is okay because you do not lose anything.

Now imagine a retailer, an advertiser, a law enforcement personnel, these people want high performance systems. They want to sell products, maximize revenue, catch terrorists. A simple heuristic solution may lead to frequent wrong decisions, which in turn will result in losses in terms of dollars & human life. These models may further benefit from some domain knowledge. for eg a soap company advertiser would not post their products on animal websites.

Coming back to the question at the top. Every ML model for a specific dataset/problem is in some ways a heuristic, but highly specialized one.

Intuition: Intuition refers to our ability to know or understand something without reasoning or proof. It is also called gut-feeling or sixth sense.

Find answer

Q) what is perceptron in Machine learning?

Sol. Ans) A perceptron is a simple model of biological neuron in an artificial neural network. Perceptron is also the name of an early algorithm for supervised learning of binary classifiers.

The perceptron algorithm was designed to classify visual inputs, categorizing subject into one of two types & separating groups with a line.

Q) What is ensembling learning?

Ans) To solve a particular computational program, multiple models such as classifiers or experts are strategically generated and combined. This process is known as ensembling learning.

It is used to classification, prediction, function etc of a model. It is also used when you build component classifiers that are more accurate and independent from each other.

Q) What are the two paradigms of ensemble methods?

Ans) a) Sequential ensemble methods

b) Parallel " "

SEQUENTIAL ENSEMBLE METHOD

where the base learners are generated sequentially (eg AdaBoost)

* Exploit the dependence b/w the base learners

* The overall performance can be boosted by weighing previously mislabeled examples with higher weight.
* Where the base learners are generated in parallel (Eg

PARALLEL ENSEMBLE METODS

* Where the base learners are generated in parallel (Eg Random forest)
* Exploit independence b/w the base learners
* The error can be reduced dramatically by averaging.

There can be a countless number of ways you can ensemble different models. But these are some techniques that are mostly used.

TYPES OF ENSEMBLING

- Averaging :- It's defined as taking the average of predictions from models in case of regression problem or while predicting probabilities for the classification problem.

Model1	Model2	Model3	Average Prediction
45	40	65	50

② Majority vote:

It's defined as taking the prediction with maximum vote / recommendation from multiple models predictions while predicting the out outcomes of a classification problem.

Model 1	Model 2	Model 3	Voting predictions
1	0	1	1

③ Weighted average:

In this, different weights are applied to predictions from multiple models then taking the average which means giving high or low importance to specific model output.

	Model 1	Model 2	Model 3	Weight Average Prediction
Weight	0.4	0.3	0.3	
Prediction	45	40	60	48

Code:

For Majority Vote:

```
From Sklearn.ensemble import VotingClassifier
model1 = logisticRegression()
model2 = tree.DecisionTreeClassifier()
model = VotingClassifier(estimators = [('lr', model1), ('dt', model2)]
        , voting = 'hard')
model.fit(X_train, y_train)
model.score(X_test, y_test)
```

For Averaging

```
model1 = tree.DecisionTreeClassifier()
model2 = KNeighborsClassifier()
mode3 = LogisticRegression()

model1.fit(X_train, y_train)
model2.fit(X_train, y_train)
model3.fit(X_train, y_train)

pred1 = model1.predict_proba(X_test)
pred2 = model2.predict_proba(X_test)
pred3 = model3.predict_proba(X_test)
finalpred = (pred1 + pred2 + pred3)/3
```

Weighted Average

```
model1 = Same
model2 = Same
model3 = Same

model1.fit(Same)
model2.fit(Same)
Model3.fit(" )

pred1 = Same
pred2 = Same
pred3 = Same
```

finalpred = (pred1 X 0.3 + pred2 X 0.3 + pred3 X 0.4)

Advanced Ensemble techniques

Now that we have covered the basic ensemble techniques, let's move on to understanding the advanced techniques

① Stacking

* Stacking is an ensemble learning technique that uses predictions from multiple ~~model~~ models to build a new model.
* This model is used for making predictions on the test set.

Below is a step-wise explanation for a simple stacked ensemble :-

1. The train set is split into 10 parts
2. A base model (suppose a decision tree) is fitted on 9 parts and predictions are made for the 10th part. This is done for each part of the train set.
3. The base model (in this case, decision tree) is then fitted on the whole train dataset.
4. Using this model, predictions are made on the test set
5. Steps 2 to 4 are repeated for another base model (say KNN) resulting in another set of predictions for the train set & test set.
6. The predictions from the train set are used as features to build a new model.

7. This model is used to make final predictions on the test prediction set.

Code from pr not

2. Blending

- Blending follows the same approach as Stacking but uses only a holdout (validation) set from the train set to make predictions. In other words, unlike Stacking, the predictions are made on the holdout set only.
- The holdout set and the predictions are used to build a model which is run on the test set

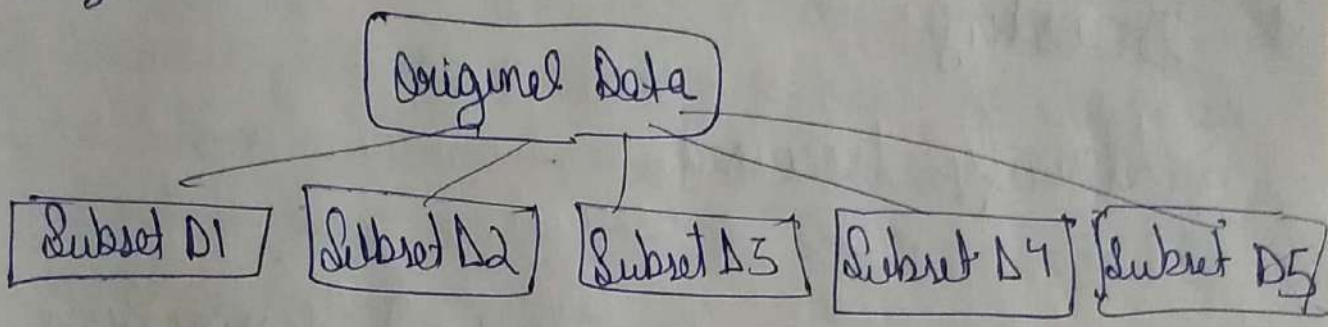
Here is a detailed explanation of this blending process:

1. The train set is split into training & validation sets.
2. Models are fitted on the training set.
3. The predictions are made on the validation set and the test set.
4. The validation set and its predictions are used as features to build a new model.
5. This model is used to make final predictions on the test & meta features.

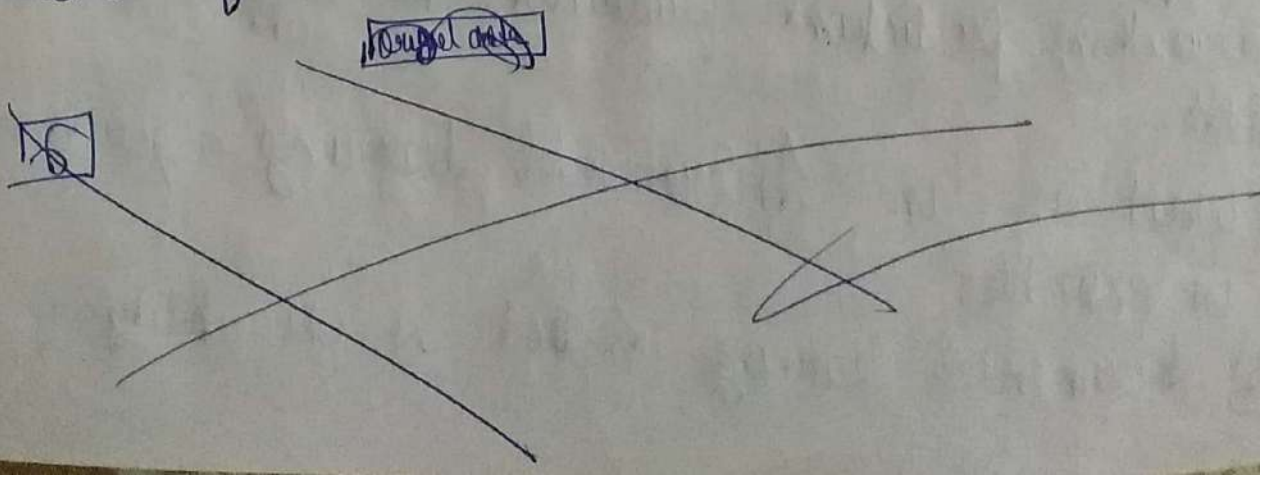
Python

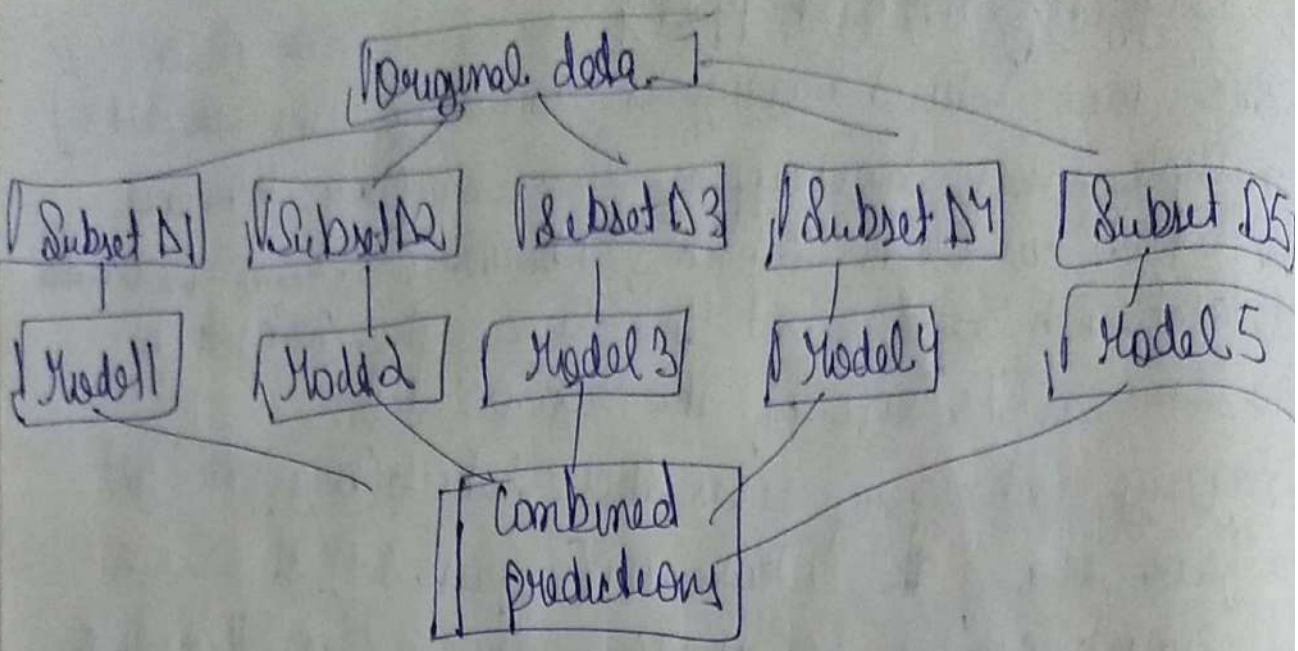
3. Bagging / Bootstrapping

* The idea behind bagging is combining the results of multiple models. Bootstrapping is a sampling technique in which we create subsets of observation from the original dataset, with replacement. The size of the subsets is the same as the size of the original set.
* Bagging technique uses these subsets (bags) to get a fair idea of the distribution (complete set). The size of subsets created for bagging may be less than the original set.



1. Multiple subsets are created from the original dataset, selecting observations with replacements.
2. A base model (weak model) is created on each of these subsets
3. The models run in parallel and are independent of each other.
4. The final predictions are determined by combining the predictions from all the models





4. Boosting

Already discussed

Q What is bias-variance decomposition of classification error in ensemble method?

Ans The expected error of a learning algorithm can be decomposed into bias & variance.
A bias term measures how closely the average classifier produced by the learning algorithm matches the target function.
The variance term measures how much the learning algorithm's prediction fluctuates for different training sets.

Q What is an Incremental learning algorithm in ensemble?

Ans Incremental learning method is the ability of

an algorithm to learn from new data that may be available after classifier has already been generated from already available dataset.

Q] What's the trade-off between bias and variance?

Ans] * Bias is error due to erroneous or overly simplistic assumptions in the learning algorithm you're using. This can lead to the model underfitting your data, making it hard for it to have high predictive accuracy.

* Variance is error due to too much complexity in the learning algorithm you're using. This leads to the algorithm being highly sensitive to high degrees of variation in your training data, which can lead your model to overfit the data.

An order to get the optimally reduced amount of error, you'll have to tradeoff bias and variance. You don't want either high bias or high variance in your model

Q3] How is KNN different from K-means clustering?

Ans] K-nearest Neighbors → supervised classification
K-means clustering → unsupervised classification

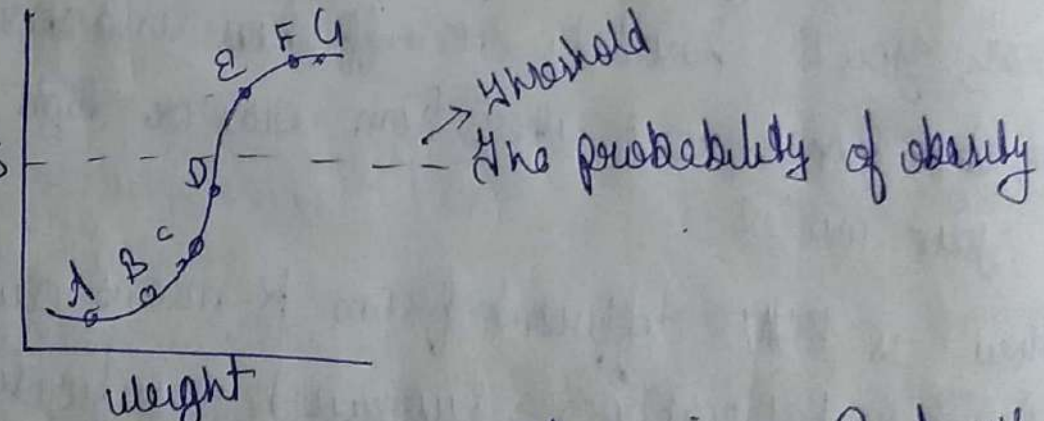
Q) Explain how a ROC curve works.

Sol:) ROC refers to Receiver operating characteristic. It is a graphical representation of the constrast between true positive rates and the false positive rate at various threshold. It's often used as a proxy for the trade-off between the sensitivity of the model (true positive) vs the fall-out or the probability it will trigger a false alarm (false positive)

* The ROC graph summarizes all of the confusion matrices that each threshold produced

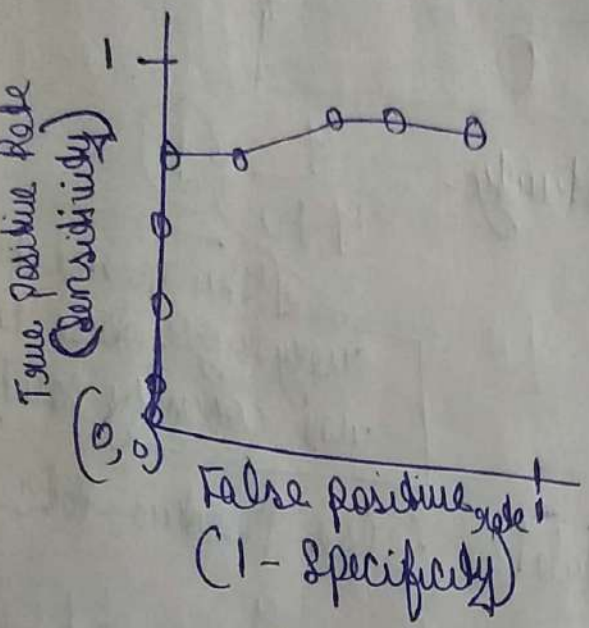
* The true-positive rate is also known as sensitivity, recall or probability of detection in machine learning. The ~~false~~ false-positive rate is also known as the fall-out or probability of false alarm and can be calculated as (1 - specificity)

*



* Here we have used logistic Regression. But we have many error in the data like point G has no obesity or point A has obesity. Here we change of threshold (probability at 0.5) from 0 to 1 then we create a confusion matrix for for each value of threshold (0 to 1)

* So instead of being overwhelmed with confusion matrices, Receiver operator characteristic (ROC) graphs provide a simple way to summarize all of the information



True positive Rate = Sensitivity = $\frac{\text{True Positives}}{\text{True positives + False negatives}}$

False positive Rate = (1 - Specificity) = $\frac{\text{False Positives}}{\text{False positives + True Negatives}}$

Confusion matrix sample

Predicted	Actual: Is obese	Actual: Is Not Obese
Is obese	True Positives	False Positives
Is Not obese	False Negatives	True Negatives

* This is how we calculate true positive rate from confusion metrics

		Actual: Is obese	Actual: Is Not obese
Predicted	Is obese	4	4
	Is not obese	0	0

True positive Rate = Sensitivity = $\frac{4}{4+0} = 1$

This means that every single obese sample was correctly classified

* So this is how we calculate False Positive Rate

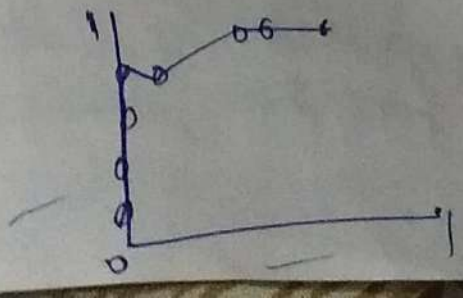
False positive Rate = 1 − Specificity = $\frac{4}{4+0} = 1$

	Is obese	Is Not obese
Is obese	4	4
Is not obese	0	0

From above we get (1,1) lets plot this on a graph

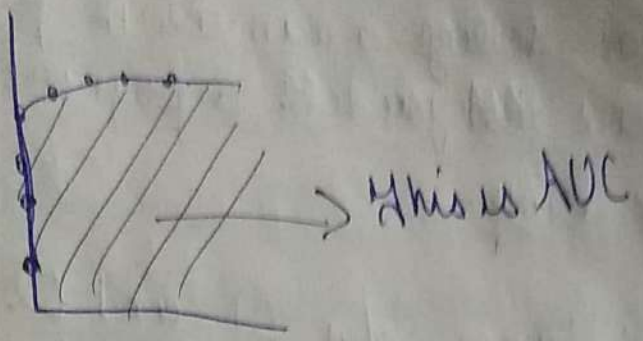
We do this for every threshold value

Then we get this type of graph



* we can say that, the ROC graph summarizes all of the confusion matrices that each threshold produced

Now lets talk about the AUC (Area under the curve)



The AUC makes it easy to compare one ROC curve to another

Note: People often replace the false positive Rate with Precision. It is the proportion of positive results that were correctly classified.

* Suppose there are 10 apples and 5 oranges in a basket You'd have perfectly recall (there are actually 10 apples and you predicted there would be 10) but 66.7% precision because out of 15 events you predicted, only 10 are correct

Q) What's the difference between Type I and Type II Error?

Sol) Type I error is a false positive, while Type II error is a false negative. ~~Boaring started, type I error~~

* Type I error means claiming something has happened when it hasn't while type II error

means that you claim nothing is happening when in fact something is.

Example:
Type I error as telling a man he is pregnant while Type II error means that you tell a pregnant woman she isn't carrying baby.

Q) Explain the difference b/w L1 and L2 regularization

Ans)

- A regression model that uses L1 regularization technique is called lasso Regression and model which uses L2 is called Ridge Regression.

L1 vs L2

L1 regularization

- When using L1 regularization, the weights for each parameter are assigned as as 0 or 1 (binary value)
- This helps perform feature selection in sparse features spaces and is good for high-dimensional data since the 0 coefficient will cause some features to not be included in the final model
- L1 can also save on computational costs since the features weighted 0 can be ignored,

however, model accuracy is often lost for this benefit
- L1 is best used in high dimensional or sparse data sets when doing classification.

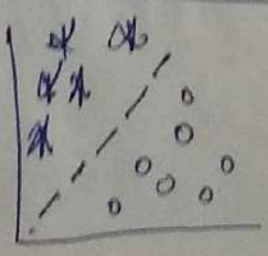
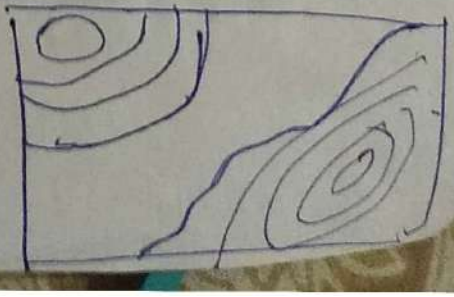
L2 Regularization

- L2 regularization spreads the error among all the features. This results in weight for every feature with the possibility that some weight are really small values close to 0.
- L2 tends to be more accurate in almost every situation however at a higher computational cost
- It is best used in non-sparse outputs, when no feature selection needs to be done, or if you need to predict a continuous output

Q] What's the difference b/w a generative and discriminative model?

Ans)

Discriminative models learn the boundary b/w classes
Generative models model the distribution of individual classes.

	Discriminative model	Generative model
Illustration		

examples | Regressions, SVMs | Naive Bayes (probability distribution)

* A generative algorithm model will learn completely from the training data and will predict the response.
* A discriminative algorithm's job is just to classify or differentiate b/w the 2 outcomes.

* Given a training set, an algorithm like logistic regression ~~or~~ tries to find a straight line - that is, a decision boundary - that separates the elephants & dogs. Then, to classify a new animal as ~~either~~ an elephant or a dog, it checks on which side of the decision boundary it falls, and makes its prediction accordingly. We call these discriminative learning algorithm.

* Here's a different approach. First looking at elephants, we can build a model of what elephants look like. Then, looking at dogs, we can build a separate model of what dogs look like. Finally to classify a new animal, we can match the new animal against the elephant model. and match the dog model, to see whether the new animal, looks more like the elephants or more like the dogs we had seen in the training set. We call these generative learning algorithm.

Q] what is a Fourier transform?

Ans] A Fourier transform converts a signal from time to frequency domain - it's a very common way to extract features from audio signals or other time series.

* At transform an aperiodic signal from the time domain to the frequency domain
* A Fourier transform is a generic method.

Q] What is more important to you - model accuracy, or model performance?

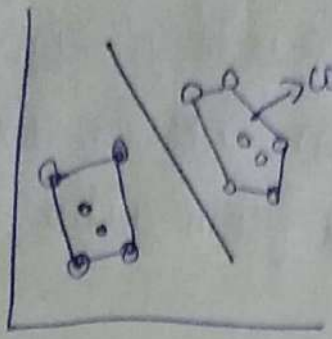
Ans] * There are model with higher accuracy that can perform worse in predictive power
* model accuracy is only a subset of model performance.

Q] What's the F1 score?

Ans] The F1 score is a measure of a model's performance. It is a weighted average of the precision and recall of a model, with results tending to 1 being the best, and those tending to 0 being the worst.

Q) What is convex hull?

Ans)



* Convex hull represents the outer boundaries of the two group of data points.
* Once convex hull is created, we get maximum margin hyperplane (MMH) as perpendicular bisector b/w two convex hulls. MMH is the line which attempts to create greatest separation b/w the two groups.

Q) Describe a Hash table.

Ans) Hashing is a technique that is used to uniquely identify a specific object from a group of ~~two~~ similar objects.

* Example:- In universities, each student is assigned a unique roll number that can be used to retrieve information about them.

Q) You are given a data set. The set has missing values which spread along 1 standard deviation from the median. What percentage of data would remain unaffected? Why

Ans) Let's assume it's a normal distribution. We know, in a normal distribution, 68%

of the data lies in 1 S.D. from mean (or mode, median), which leaves 32% of the data unaffected. Therefore, 32% of the data would remain unaffected by missing value.

Q] What is multicollinearity?

Ans] * Multicollinearity means that some of the ~~regressors~~ regressors (independent variables) are highly correlated with each other.

* It will make the estimate highly unstable. This instability will increase the variance of estimates. It means that if there is a small change in X, produces large changes in estimate.

effects -

(1) It will be difficult to find the correct predictors from the set of predictors

(2) It will be difficult to find out precise effect of each predictor.

Q] What is the difference between covariance and correlation?

Ans] Correlation is the standardized form of covariance

Q] What is Normalization?

Ans) → All feature values will be in the range 0 to 1

min max Scaler