Machine Learning

In basic terms ML is the process of training a piece of software, called a model to make useful predictions from data.

ML systems fall into 3 distinct categories

① Supervised learning
② Unsupervised learning
③ Reinforcement learning.

① Supervised learning.

→ Make predictions after seeing lots of data with the correct answers & then discovering connections b/w the elements in the data that produce correct answers.

→ The ML systems are "supervised" in the sense that human gives the ML system data with the known correct results.

Most common use case of SL

a) Regression
b) classification.

② Regression

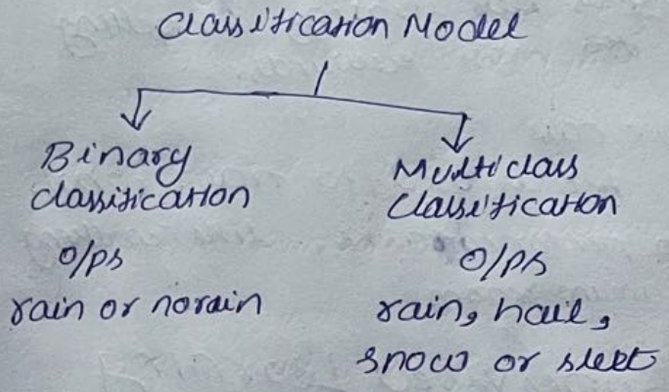
→ A regression model predicts a numeric value

Scenario	Possible i/p data	Numeric prediction
→ Future House Price	square, zip code, no of rooms, tax rate construction costs & no of homes sold in area	Price of the home
→ Future ride time		The time in minutes & seconds to arrive at dest

b) classification

classification model predict the likelihood that something belongs to a category.

→ o/ps a value that states whether or not something belongs to a particular category.



② Unsupervised learning

→ UL models make predictions by being given data that does not contain any correct answers.

→ UL model goal is to identify meaningful patterns among the data

→ model has no hints on how to categorize each piece of data but instead it must infer its own rules.

→ A common UL model employs a technique called clustering. (clustering differs from classification b/z categories aren't defined by you)

The model finds data points that demarcate natural groupings

Fig1 An ML model clustering similar data points.

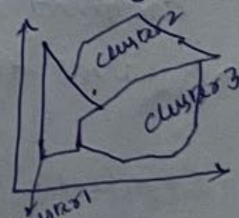


Fig2 Groups of clusters with natural demarcations

Reinforcement learning.

→ RL model make predictions by getting **rewards** or penalties based on actions performed within an environment

→ Generates a **policy** that defines best strategy for getting the most rewards.

Use case

RL is used to train robots to perform tasks, like walking around room

Supervised Learning.

→ more potential use cases than UL

→ Better utilizes historical data

SL is based on following core concepts

① Data
② Model
③ Training
④ Evaluating
⑤ Inference.

Data

→ data comes in form of coords & numbers stored in tables the values of pixels & waveforms captured in images & audio files

→ Related data is stored in form of datasets

→ Datasets made up of individual examples that contain **features** & **labels**

→ Features are values that a SL model uses to predict label

→ Label - answer or value we want to predict.

Dataset characteristics

① Size - no of examples

② Diversity - range those examples cover.

Good datasets are large & highly diverse.

③ features - variables shld be more & relavent

Eg: weather (Rain forecast)

{latitude, longitude, temperature, humidity, cloud coverage, wind direction & atmospheric pressure} → features

rainfall → label

Model

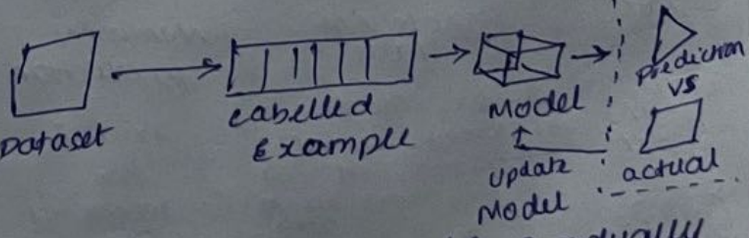
Training

→ Based on the difference b/w the predicted & the actual values defined as the loss the model gradually updates its solution

→ In other words the model learns the mathematical relationship b/w features & the label.

Eg: Predicted by model → 1.15 inch rain

Actual → 0.75 inch



→ In this way, the model gradually learns the correct relationship b/w features & the label

→ ML practitioner can make subtle adjustments to config & features of the model

→ Eg: In weather dataset
Removing & adding time_of_day feature to check if the predictions with or without is better.

Evaluating

→ Check trained model how well it learnt

→ Give the model only the features of the dataset

→ Check & compare with the obtained label values.

Inferences

→ We can use model to make predictions known as inferences

Predictive Power

→ Like told previously not every feature in a dataset has predictive power

Note: If you know beforehand the value or category you want to predict, you'd use supervised learning.

→ If you wanted to learn if your dataset contains any segmentations, groupings, patterns you'd use unsupervised learning.

Framing

→ Label usually represented by y

→ Features $\{x_1, x_2 \dots x_n\}$

- Example is a particular instance of data, x
- Labeled example $\{(x, y)\}$
  → Used to train the model
- Unlabeled Example $(x, ?)$
  → Used for making predictions on new data
- Model - maps x to y.

(A) Key ML Terminology.

① Labels
→ The thing we are predicting the 'y' variable in simple linear regression.

② Features
→ i/p variable 'x' variable in simple linear Regression.

Regression vs classification

A regression model predicts continuous values

Eg: What is the value of house in California
What is the probability user will click on this ad.

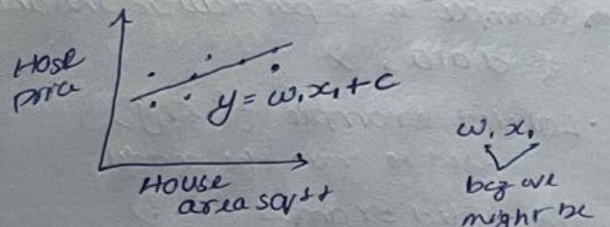
A classification model predicts discrete value

Eg In a given email message spam or not spam!

Is this an image of a dog, a cat, or a hamster?

Linear Regression

It is method for finding the straight line or hyperplane that best fits a set of points.



$w_1 x_1$ ← bcz we might be in more than 1D so 1 & 1

A Convenient Loss Function for Regression

→ sqr error

→ L_2 Loss for a given example is also called squared error

= Sqr of distance b/w prediction & label

= (observation − prediction)^2

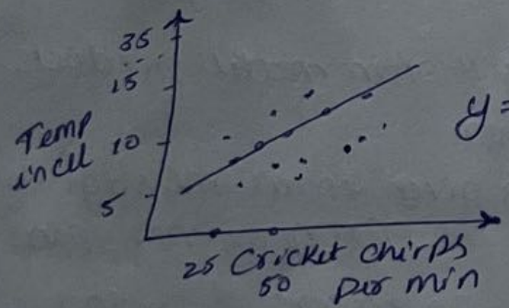
$= (y - y')^2$

$L_2 \text{ Loss} = \sum_{(x,y) \in D} (y - \text{prediction}(x))^2$

Σ → we are summing over all examples in the training set

D → sometimes useful to average so divide by $|D|$

Cricket Chirps per min VS Temp celcius



y → temp in celcius, the value we are trying to predict

m → slope of the line

x → no of chirps per minute. the value of our i/p feature

b is the y-intercept

In ML we write as

$y' = b + w_1 x_1$

y' → predicted label (a desired o/p)

b → bias (the y intercept) sometimes referred to as w_0

w_1 → weight of feature 1

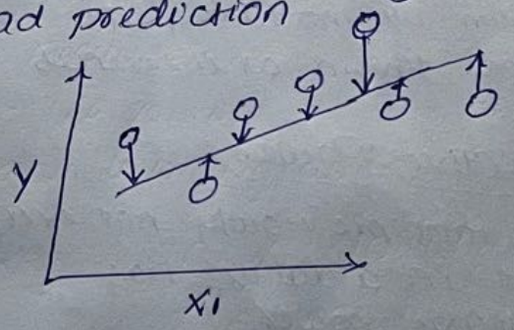
weight is same concept as slope m

x_1 → is a feature

To infer temp y'

$y' = b + w_1 x_1 + w_2 x_2 + w_3 x_3$

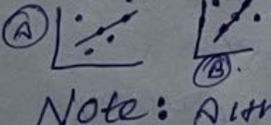
→ Loss is the penalty for bad prediction



→ Squared Loss → popular loss function.

→ Mean Square error ← Avg sqrd loss

$MSE = \frac{1}{N} \sum_{(x,y) \in D} (y - \text{prediction}(x))^2$



Note: Although only 2 points lay off the line in Eg B those pts are twice as far off the line as the outlier points in left figure squared loss amplifies those differences, so an offset of 2 incurs loss of 4 as as offset of 1

Some commonly used
ML Algorithms are

① Linear Regression
② Logistic Regression
③ Decision Tree
④ SVM (support vector machines)
⑤ Naive Bayes
⑥ KNN (K nearest neighbours)
⑦ K-Means
⑧ Random Forest
And more.

Steps to get started with Machine Learning

① Define the Problem
② Collect Data
③ Explore the Data
④ Pre-process the Data
⑤ Split the Data
⑥ Choose a Model
⑦ Train the Model
⑧ Evaluate the Model
⑨ Fine-tune the Model
⑩ Deploy the Model
⑪ Monitor the Model

Some commands

Predict species of Iris flower
Simple ML Example

```
import pandas as pd
from sklearn.model_selection
        import train_test_split
from sklearn.svm import SVC
df = pd.read_csv("iris.csv")
X = df[["sepal_length", "sepal_width",
        "petal_length", "petal_width"]]
y = df["species"]

X_train, X_test, y_train,
y_test = train_test_split
        (X, y, test_size=0.2,
        random_state=42)
# split the data into training
    & testing sets

model = SVC()
model.fit(X_train, y_train)
# Create an SVM model & train it

#Evaluate model on test data
accuracy = model.score
                (X_test, y_test)
print("Test accuracy:", accuracy)
```

O/P

0.9666666667

→ Arthur Samuel - 1959 - IBM coined the term Machine learning

There is something known as Semi Supervised Learning

→ In semi supervised learning an incomplete training signal is given

→ It is approach to ML that combines small labelled data with large amt of unlabelled data.

NOTE: ML is very close to Data Mining & Data Science.

→ The accuracy of the models predictions can be evaluated using various performance metrics such as accuracy, precision, recall & F1-score.

Under Unsupervised learning

There are 2 main types

① clustering - we know

② Dimensionality reduction:
→ This reduces the no of i/p variables in a dataset

Machine Learning	Traditional Programming	Artificial Intelligence
→ ML is a subset of AI that focus on learning from data	→ Rule based code is written by developers	→ Making m/c so capable that it can perform tasks for require human intelligence
→ depends on historical data	→ No need	→ Involves different ML Deep learning techniques
→ can find patterns & insights that's impossible for human	→ Humans find patterns & create rules	→ outcome of ML

How ML Algorithm work

① Forward Pass - Takes i/p data & produces o/p depending on model algo computes prediction

② Loss Function - Minimize the error by adjusting internal params

③ Model Optimization Process -
- Iterative process of adjusting the internal parameters
- Done using algo like gradient descent

→ This is useful for reducing complexity of dataset making it easier to visualize & analyze.

→ Popular Algos under Dimensionality reduction
- PCA (Principal Component Analysis)
- t-SNE
- Auto encoders.

Limitations of ML

→ Lack of data or diversity in dataset
→ ML model needs to have heterogeneity (diverse) to learn meaningful insight
→ If no or few variations in dataset ML model cant be created.
→ Recommended to have 20 observations per group

Under SL - Regression

Linear Regression

→ Type of regression algorithm that is used to predict a continuous output value.
→ The algorithm tries to find a linear relationship b/w the i/p features & the output variable.
→ The o/p value is predicted based on the weighted sum of i/p features.

① Linear Regression
② Logistic Regression
③ Decision Trees
④ Random Forests
⑤ Naive Bayes Classifiers
⑥ Support Vector Machine

② Logistic
→ same finds linear relationship
→ o/p is true or false

③ Decision Trees
→ Used for both regression & classification tasks
→ Tree like structure that is used model decisions & their possible consequences.
→ Each node in tree → decision
Each leaf node → outcome
→ Used to model complex relationships b/w i/p & o/p features

Random Forests

→ Made up of multiple decision trees
→ Each tree is trained on different subset of the i/p features & data
→ The final subset is made by aggregating the predictions of all trees in forest

Difference b/w SL & USL

SL	USL
→ Uses known & labelled data as i/p	→ Uses unknown data as i/p
→ Less computational complexity	→ More complex
→ No of classes are known	→ unknown
→ Accurate & Reliable Result	→ Moderately Acc & Rel
→ Training data is used to infer model	→ Not used
→ Also called classification	→ Also called clustering.
→ Eg Optical Character Recognition	→ Eg: face in an image.

Reinforcement Learning

→ Make decisions & take action in the aim of maximizing reward.
→ In SL data has answer key, in RL no answer, agent decides what to do, to perform the given task
→ Bound to learn by its experience
→ All abt learning optimal behaviour
→ Learns by trial & error
→ After each completion agent receives feedback correct, neutral or incorrect

Eg: chess game, text summarization.

Types of Reinforcement L

① Positive → event occurs due to a particular behaviour
- Maximize Performance
- Sustain change for long period of time
- Too much RL can lead to overload of states which can diminish the results.

② Negative - Strengthening of behaviour by negative condition stopped or avoided.
- Increases Behaviour
- Provide defiance to a minimum standard of Performance

Elements of RL

① Policy
- Learning agent behaviour for given time period

② Reward function
→ Goal in a RL problem
→ Provides a numerical score based on state of the env

③ Value function
→ Specify what is good in long run

④ Model of the environment
- RL has intermediate state which tells how good they are now in reaching the goal.
- decision maker (model) is called agent
- agent continuously interacts with env at each time step t to get scalar numerical reward.

Applications of RL
→ Petrol Refinery control adjustments
→ Robotics

Advantages & Disadvantages of RL

① Used to solve complex problems with conventional technique
② Handle envs that are non deterministic (useful for Envs that change frequently)
③ decision making, control & optimization.

Code example

```
import gym          ← an api for RL
import numpy as np

# Define the Q table & learning state
q_table = np.zeros((state_size, action_size))
alpha = 0.8
gamma = 0.95

# Train the Q learning algorithm
```

Classification

① Linear classifiers
- (a) Logistic Regression
- (b) Support Vector Machines having Kernel = "linear"
- (c) Singel layer Perceptron
- (d) Stochastic Gradient Descent (SGD)

② Non linear classifier
- (a) K-Nearest Neighbours
- (b) Kernel SVM
- (c) Naive Bayes
- (d) Decision Tree Classification
- (e) Ensemble learning classifiers
  - → Random Forests
  - → Ada Boost
  - → Bagging classifier
  - → Voting classifier
  - → Extra trees classifier
- (f) Multi-layer Artificial Neural Networks

Type of learners in classification Algorithm

① Lazy learners
- → They dont learn during training phase
- → They store training data & use it to classify new instances at prediction time.
- → Not effective in high dimensional spaces. Eg: KNN, case based reasoning

② Eager Learners
- → model based learners
- → use training data to learn
- → Effective in high dimensional spaces

Classification model Evaluations

① Classification Accuracy

② confusion matrix
- → To find no of true positives true negatives, false positives, false negatives over a class

③ Precision & recall
- → Precision = $\frac{\text{true positives}}{\text{predicted positives}}$
- → recall = $\frac{\text{true positives}}{\text{actual positives}}$.

To find which class is better

④ F1-score
- → $2 \times \frac{(\text{precision} \times \text{recall})}{(\text{precision} + \text{recall})}$

⑤ ROC curve & AUC
Receive operating characteristic Area under the curve.

⑥ Cross validation

How does classification work

① Understanding the Problem
- → choosing the appropriate classification algorithm

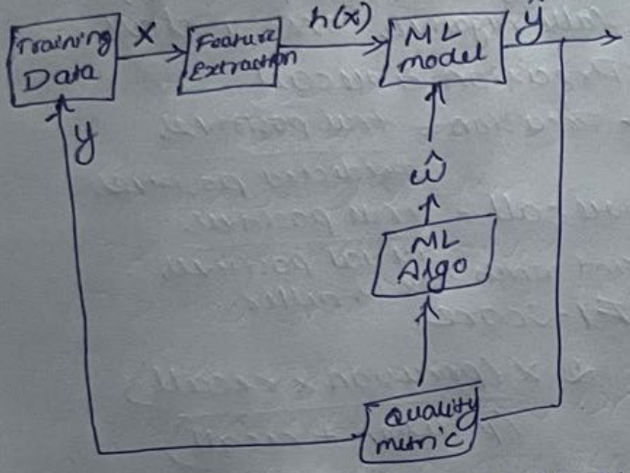
② Data preparation
collecting, preprocessing, splitting into training, validation, test sets
format data that can be used by classification algo

③ Feature Extraction.
- → selecting only required features

④ Model selection.
logistic regression, decision trees, SVM or neural networks

⑤ Model Training
- → adjusting params & minimizing error

⑥ Model Evaluations
→ Check on all those params
⑦ Fine tuning the model
⑧ Deploying the model



Applications of Classification Algo
① Email spam filtering
② Credit risk assessment
③ Sentiment analysis
④ Fraud det
⑤ Quality control.

Regression

Types of Regression Techniques

1. Linear Regression
2. Polynomial Regression
3. Stepwise "
4. Decision Tree "
5. Random Forest "
6. Support Vector "
7. Ridge
8. Lasso
9. Elastic Net
10. Bayesian

1. Linear Regression.

→ A technique used to model a linear relationship b/w single dependent variable (y) & one or more independent variables (x)

$$y = \theta x + b$$

→ In this technique there is a danger of **overfitting.**

(& are 2 types of issues in ML: underfitting)

2. Polynomial Regression

→ To model a nonlinear relationship b/w y & x

→ $y = a_0 + a_1 x + a_2 x^2 + \dots + a_n x^n$

3. Stepwise Regression.

→ Used to build a model that is parsimonious i.e smallest no of variables that can explain the data

→ Forward Selection	Backward Elimination
Add variables to model until no further improvement made	Iteratively remove variable until no further improvement made

→ Prevents overfitting

$$b_{j\,std} = b_j (s_x * s_j^{-1})$$

forward selection
backward elimination
bidirectional elimination

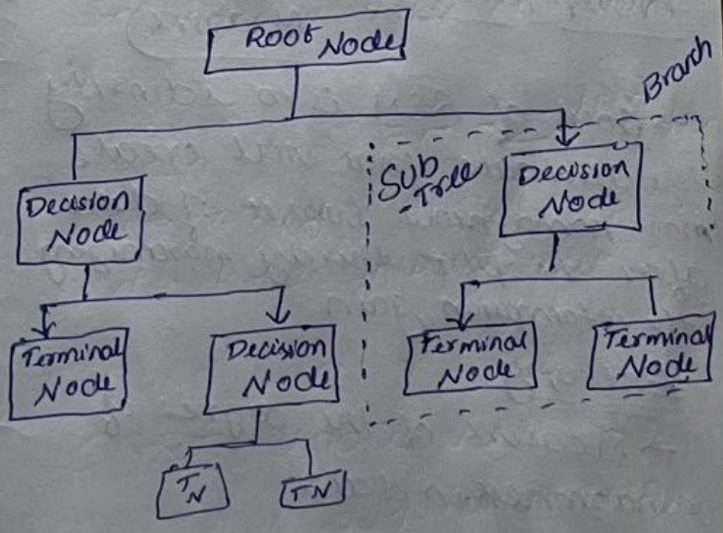
4. Decision Tree Regression

→ Used for both classification & Regression Tasks

→ Constructed by recursively splitting the training data into subsets based on the values of the attributes until stopping criteria is met like max depth of the tree or min no of samples reqr to split a node.

attributes are selected based on metric such as entropy or Gini impurity. (ASM) → separate topic

Decision Tree Terminologies



Root Node : complete dataset starting pt of decision making process

Decision/Internal : symbolize choice regarding an i/p feature

Leaf/Terminal : indicates a class label or a numerical value

⑤ Branch/Sub Tree :

Parent Node: Node that divides into one or more child nodes
child.

Impurity : measure of target variables homogeneity
→ Gini index & entropy are two commonly used impurity.

Variance : predicted ≈ target
Mean squared error
Mean absolute error
Friedman - mse
Half Poisson variance
} to measure variance

Information Gain : Measure of reduction in impurity.

Pruning : Process of removing branches that lead to overfitting.

ASM (Attribute Selection Measure)

→ Goal of ASM is to identify the attribute that will create most homogeneous subset of data after the split thereby maximizing the information gain

Entropy

→ Measure of the degree of randomness in data

→ Entropy for a subset of original dataset having K number of classes for the i^{th} node can be defined as

$$H_i = -\sum_{k \in K}^{n} P(i,k) \log_2 P(i,k)$$

$$S.P(k) = \frac{1}{n} \sum I(y=k)$$

S → dataset sample
k → particular class from K classes

Gini

Gini impurity is a score that evaluates how accurate a split is among the classified groups
→ Evaluates a score in the range b/w 0 and 1

$$\text{Gini Impurity} = 1 - \sum P_i^2$$

Information Gain

→ Measures the reduction in the entropy or variance
→ Determines the usefullness of a feature by partitioning dataset into more homogeneous subsets. w.r.t class labels or target variable.
→ The higher the information gain, more valuable the feature is

$$\text{Information Gain}(H, A) = H - \sum \frac{|H_v|}{|H|} H_v$$

$|H| \rightarrow$ Entropy of dataset sample S
$|H_v| \rightarrow$ No of Instances in subset S

To Build Decision Tree
CART Algo is used
(classification & Regression Tree)

⑤ Random Forest Regression

→ Ensemble technique capable of performing both regression & classification tasks, with the use of multiple decision trees & a technique called Bootstrap and Aggregation, commonly known as bagging.

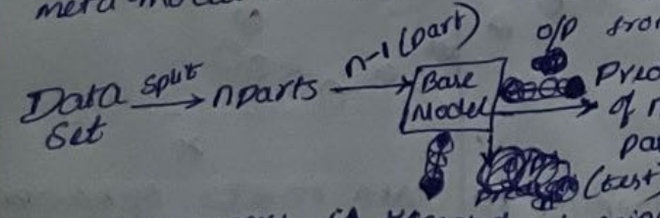
Basic Ensemble method

1) Averaging Method
Mainly used for regression problems. Building multiple models independently & returning the average of the prediction of all the models. combined o/p than individual o/p reduces variance so.

2) Max voting
Used for classification problems Building multiple models independently and getting their individual o/p called vote. The class with maximum votes is returned as o/p

Advanced Enseble methods

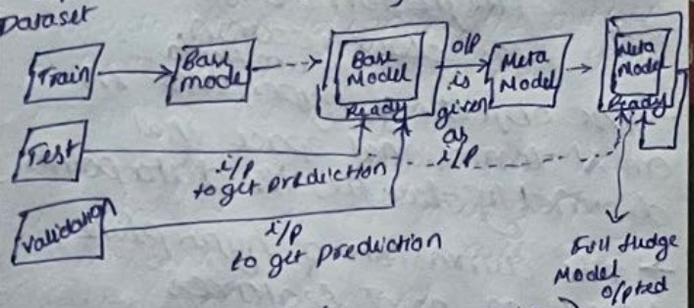
① Stacking
→ combines multiple models (regression or classification) via meta model (meta-classifier or meta-regression). The Base models are trained on complt dataset meta-model trained on features returned as o/p



The above process is replaced changing the i/ps, (1,2,3,4,5) if 5 is n, process replaced keeping 4 as n. & so on

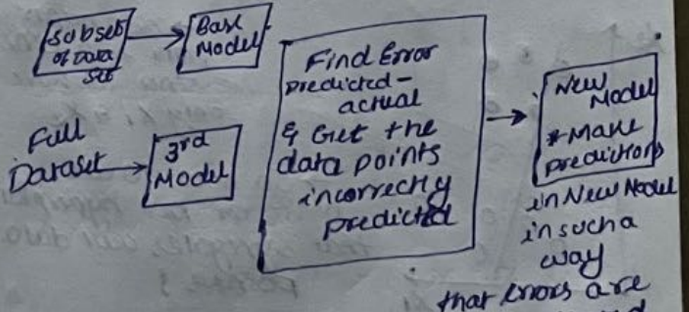
→ These predictions are used as features to build the new meta model & finally test dataset is fed to it

② Blending.
→ similar to stacking method but rather than using the whole dataset for training base-models the dataset is seperated into train, test & validation



③ Bagging - (bootstrapping)
→ Base Models are run on bags
→ Bag of subsets of whole dataset with replacement to make the size of the whole bag same as Dataset size
→ Process remains same as stacking

④ Boosting



→ The final Model is mean of all
(strong learner)
(weak learners)

⑥ Support Vector Regression

→ SVR is a type of SVM

SVM (Support Vector Machine)

→ Used for both classification & Regression. Best suited for classification.

→ The objective of SVM is to find a hyperplane in an N dimensional space that distinctly classifies the data points

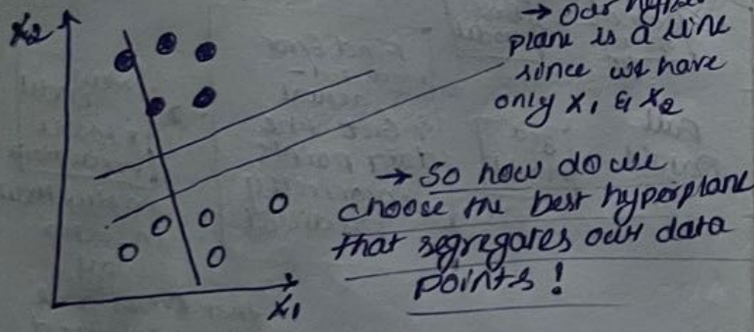
→ The dimension of hyperplane depends on no of features

2 features → then hyperplane is just a line

3 features → 2-D plane

difficult to imagine if its more than 3 features.

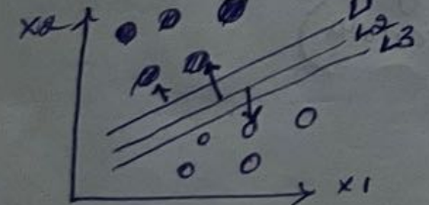
Consider 2 independent variables X_1, X_2 & 1 dependent variable which is dashed or dotted points



→ Our hyperplane is a line since we have only X_1 & X_2

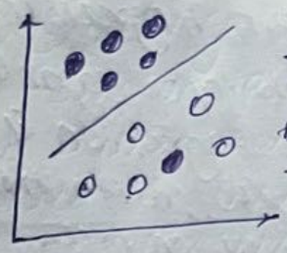
→ So how do we choose the best hyperplane that segregates our data points!

How does SVM do



→ We choose a hyperplane whose distance from it to the nearest data point on each side is max

→ If such hyperplane exist then it is called max-margin hyp / hard Margin



→ Here we have 1 blue ball in the boundary of red ball

→ How does SVM classify?

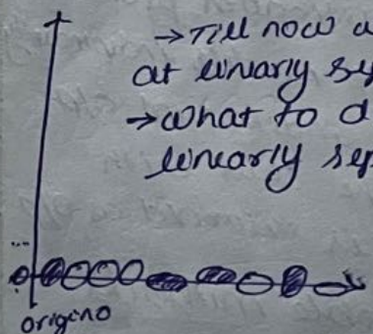
→ SVM algorithm has the characteristics to ignore the outlier & find the best hyperplane that maximizes the margin.

→ It adds a penalty each time a point crosses the margin. so margins in these cases are called soft margins

→ If no violations → no hinge loss

If violations hinge loss proportional to the distance of violation.

$$HL = 1/\text{margin} + \lambda(\Sigma \text{penalty})$$

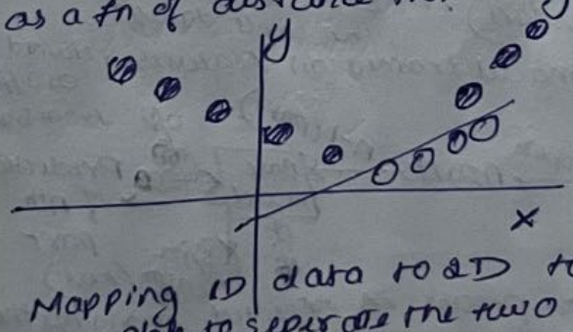


→ Till now we were looking at linearly separable data

→ What to do if data not linearly separable?

SVM solves this by creating a new variable using Kernel

→ We call a point X_i on the line & we create a new variable y_i as a fn of distance from origin o.



Mapping 1D data to 2D to become able to separate the two classes.

→ The new variable y is created as a function of distance from the origin. A non-linear function that creates a new variable is referred to as Kernel.

Use of a Kernel in the SVM Algorithm

SVM Kernel is a function that takes low-dimensional input space & transforms into higher dimensional space. it converts non-separable problems to separable problems
Useful in non linear seperation

Advantages of SVM

① Effective in high dimensional cases.
② Its memory is efficient since it uses subset of training points in the decision fn called support vectors.
③ Different Kernel fns can be specified for decision fns & its possible to specify custom Kernels.

SVM implement in Python

SVR

→ It tries to find a function that best predicts the continuous output value for a given i/p value
→ SVR can use both linear & non linear Kernals.

linear Kernel → simple dot product b/w 2 i/p vectors

non-linear Kernel → complex fn that can capture more intricate patterns in data

→ In Python → 'SVR' class
set Kernel parameter → 'linear'
→ 'RBF' (non linear)

Concepts related to SVR

① SVM
② Kernels
③ Hyperparameters - SVR has several hyperparameters that you can adjust to control the behaviour of the model.
④ Model Evaluation - Metrics like MSE, MAE to measure error.

Code examples
→ Linear Kernel
→ Polynomial Kernel
→ RBF Kernel.

⑦ Ridge Regression

→ Basically a regularized version of a Linear Regressor

→ we add a regularized term which has param 'alpha' which controls the regularization of the model

→ i.e helps in reducing the variance

$$J(\theta) = \frac{1}{m}(X\theta - Y)^2 + \alpha \frac{1}{2}(\theta)^2$$

if $\alpha = 0$ same as Linear regression

larger α stronger regularization which fits the data & helps to keep the weights lower as possible.

Code Eg: Scaling I/ps through sklearns Standard scalar is must

⑧ LASSO Regression

→ Least Absolute shrinkage & selection operator

→ The primary goal of LASSO is to find balance between model simplicity & accuracy

→ Shrinkage is where data values are shrunk towards a central point as mean.

→ LASSO encourages simple, sparse models (models with fewer parameters)

→ Uses L1 regularization technique - this is used when we have more features bcz automatic feature selection has happened.

Linear Regression Model

$$Y = B_0 + B_1 X_1 + \cdots B_n X_n$$

L1 Regularization ← adding penalty term

$$L1 = \lambda * (|B_1| + |B_2| + \cdots |B_n|)$$

So LASSO = RSS + L1

RSS → Residual sum of squares which measures the error b/w actual & predicted.

λ → tuning parameter.

Ridge Regression → soft thresholding of coefficients	LASSO Regression → hard thresholding
→ The penalty term is sum of sqrs of coefficients - L2 Regularization	→ sum of the absolute values of the coefficients L1 regularization
→ works well when there are large no of features	→ small no of features

Elastic Net Regression

→ when Linear Regression suffers from overfitting, can't deal with collinear data, too many features in dataset some of which are not relavent to predictive model.

→ This things increase inaccurate Prediction on test set (or overfitting) such a model with high varriance does not find soln for new data

→ SO ENR uses L1 & L2 regularization to get benefits of both Ridge & Lasso.

Cost fn for ENR is

$$\frac{1}{m}\left[\sum_{i=1}^{m}(y^i - h(x^i))^2 + \lambda_1 \sum_{j=1}^{n} w_j + \lambda_2 \sum_{j=1}^{n} w_j^2\right]$$

w_j → weight of i^{th} feature

n → no of features in data set

λ_1 → regularization strength for the L1 norm

λ_2 → L2 norm.

Bayesian Linear Regression

→ Based on Bayes Theorem.

Baye's Theorem

① Independent Events

② Dependent Events

③ Conditional Probablity

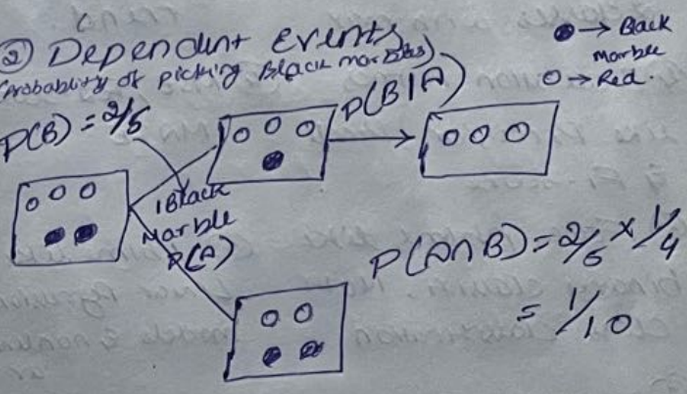
① Independent Events

Toss a coin probablity is H → 0.5, T → 0.5

Toss 2 coin simultaniously the o/p of 1 is not dependent on other

② Dependent Events

(Probablity of picking black marbles)



$P(A \cap B) = \frac{2}{5} \times \frac{1}{4} = \frac{1}{10}$

$P(B|A)$ → probablity of B given A is cmplt'd.

$P(B) = \frac{2}{5}$ $P(B|A) = \frac{1}{4}$

W.K.T ③ Conditional Probablity

$$P(B|A) = \frac{P(A \cap B)}{P(B)} = \frac{1/10}{2/5} = \frac{1}{4}$$

Lets derive Bayes Theorem

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

W.K.T $P(A \cap B) = P(B \cap A)$

$$P(A|B) * P(B) = P(B|A) * P(A)$$

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

$P(B|A)$ → Likelihood Probab

$P(A)$ → Prior

$P(A|B)$ → Posterior Probablity

$P(B)$ → Marginal Probablity.

Difference b/w classification & Regression

Classification	Regression
① Target variables are discrete	① continous
② Spam Email classify, Disease Prediction	② Housing Price Rainfall
③ In this we try to find best possible decision boundary which can separate 2 classes with max	③ we find best fit line which can represent over-all trend.
④ Evaluation Metrics like Precision, Recall & F1-score	④ MSE, R2-score MAPE
⑤ Face pblms like binary classifi, Multi class classification	⑤ Pblms like Linear Regression models & nonlinear
⑥ Spam detection image recognition sentiment analysis	⑥ Stock price prediction House price prediction. demand forecast.

SL Algos

- (1) Gradient Descent
  → Iterative Optimization ~~model~~ algo which is commonly used to train ML models & neural networks.
  → Adjusting model params to yeild smallest cost function.
  Cost function is error b/w actual & predicted.

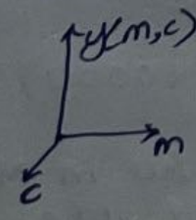
These below factors determine the future iterations

(a) Learning rate (also referred as stepsize or alpha)
→ Size of the steps that are taken to reach the minimum
High learning rates result in larger steps but risks overshooting the minimum
Low learning rates has small step size but more precision.

(b) Cost (or loss) function
→ difference b/w actual y & predicted y at its current position.

Consider a cost function for $y = mx + C$

$$J(m,C) = \sum_{i=1}^{n} (y_i - (mx_i + C))^2$$



X	Y
1	2
3	4

← Training data

initial assumptions. $C=0$, $m=1$ ← This is the ultimate goal of Gradient Descent

$$J(m,C) = [2-(C+m\cdot 1)]^2 + [4-(C+3m)]^2$$

(datapoint1, datapoint2)

$$\frac{\partial J}{\partial C} = -2[2-(C+m)] + [-2(4-(C+3m))]$$
$$= -2[2-(1)] + [-2(4-3)]$$
$$= -2[1] + [-2] = -4$$

$$C_{new} = C_{old} - LR \times (-4)$$

↓ How aggressively you want to reach 0 / step u want to take / Learning Rate

$$= 0 - (0.001 \times (-4))$$
$$= 0.004$$

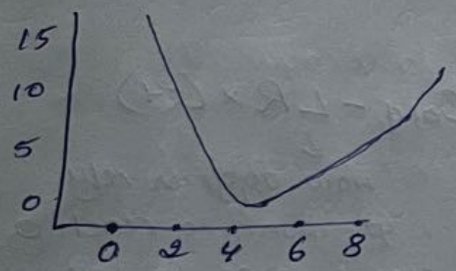
So new $C = 0.004$ & $m = -4$

How does algo knows when to stop
→ There will be a time where -4 reaches 0 after some iterations.

```
import numpy as np
import matplotlib.pyplot as plt
X = np.arange(10)
Y = (X-5)**2
Print(X, Y)

#Given a fn f(x). we want to
find the value of x that minizesf

## Visualisation
plt.style.use("seaborn")
plt.plot(X, Y)
plt.ylabel("F(X)")
plt.xlabel("X")
plt.show()
```




```
X=0
lr=0.1
error = []
#50 steps in the downhill
                direction
plt.plot(X, Y)
for i in range(50):
    grad = 2*(X-5)
    x = x - lr* grad
    y = (X-5)**2
    error.append(y)
    plt.scatter(X,y)
    print(X)
#Plot the values of error
plt.plot(error)
plt.show()
```

BGD, SBGD, MBGD

Eg:

area	bedrooms	price
2600	3	550000
3000	4	565000
3200	3	?
3600	3	~
4000	5	~
4100	6	~

ML Model

price = 1 * area + 1 * bedrooms + 1

price = 2604 ← ~~2600~~

price = 550000

error1 = (price - price)^2

error2 kyi.

⋮

error 6

So if you go through all samples then at the end is called "End of 1st Epoch"

Total Error = error1 + error2 + .. + error6

Mean Squared Error (MSE) = $\frac{\text{Total Error}}{6}$

Now adjust the weights

$w1 = w1 - \text{learning rate} * \frac{\partial(MSE)}{\partial w_1}$

$w1 = 1 - (-50) = 51$

$w2 = 1 - (-8) = 9$

$b = b - \text{learning rate} * \frac{\partial(MSE)}{\partial b}$

bias = 1 - (-20000) = 20001

Now use this in ML model & train & find 2nd Epoch allagain

price = 51 * area + 9 * bedrooms + 20001

→ Again find the error 6
→ Keep doing this until we have an error value very minimum
→ This is Batch gradient descent.
→ But what if we have 10 Million samples instead of 6
→ What if we have more than 2 features (area & bedroom)
→ Too much computation impossible to train the model

So
1. Randomly pick single data training sample.
2. Adjust weights
3. Again randomly pick a training sample
4. Again adjust weights

You adjust weights after every training sample this is called Stochastic Gradient Descent.

Batch Gradient Descent

① Use all training samples for one fwd pass & then adjust weights

② Good for small training set

Stochastic Gradient Descent

① Use one (randomly picked) sample for a forward pass & then adjust weights

② Big training set & we dont want too much computation.

Mini Batch GD

① In BGD & SGD we use 1 training sample here we use a batch of randomly picked training sample.

RL UL

ML Tutorial
SL
GD LR LogR Naive Bayes.
LR NRDR

Linear Regression

SL Algos Implimentation

Gradient Descent

① Batch Gradient Descent

Extreme supervision required for 1st 3.

② Stochastic Gradient Descent

③ Mini Batch Gradient Descent

Linear Regression

① LR impln using Python

② MULTIple Linear Reg

③ Polynomial LR

m

Logistic Regression

Naive Bayes Classifiers

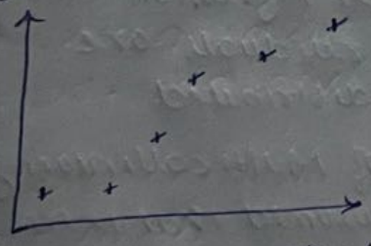
2 programs

Linear Regression

Dataset

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

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn import linear_model
df = pd.read_csv("homeprices.csv")
%matplotlib inline
plt.scatter(df.area, df.price, color='red', marker='+')
plt.xlabel('area (sqft)')
plt.ylabel('price (US $)')
```



From graph we come to know we can use linear Regression model

```
reg = linear_model.LinearRegression()
reg.fit(df[['area']], df.price)
reg.predict(3300)
```

o/p 628715.13..

We have y = mx + b = 628715.13
to get m & b seperately

m

```
reg.coef_
```

b

```
reg.intercept_
```

W.R.T price = m * area + b

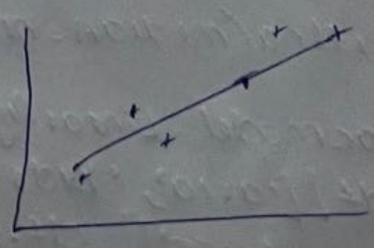
```
y = reg.coef_ * 3000 + reg.intercept_
```

o/p
628715.13

Testing - pass a csv which has only area column

```
d = pd.read_csv("areas.csv")
d.head(3)
p = reg.predict(d)
d['prices'] = p
d.to_csv("prediction.csv", index=False)
```

```
%matplotlib inline
plt.xlabel('area', fontsize=20)
plt.ylabel('price', fontsize=20)
plt.scatter(df.area, df.price, color='red', marker='+')
plt.plot(df.area, reg.predict(df[['area']]), color="blue")
```



Exercise: Predict canada per capita income in 2020

Multiple Variable Regression

Data

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

price = m_1 * area + m_2 * bedrooms + m_3 * age + b

step1
Data Preprocessing
step2
Build Model.

→ Fill the missing value with a median

```
import pandas as pd
import numpy as np
from sklearn import linear_model
df = pd.read_csv("....csv")

# calculate median of bedroom
import math
median_bedrooms =
  math.floor(df.bedrooms.
             median())

df.bedrooms = df.bedrooms.
              fillna(median_bedrooms)

reg = linear_model.LinearRegression()
reg.fit(df[['area', 'bedrooms',
            'age']], df.price)

reg.coef_
reg.intercept_
reg.predict([[3000, 3, 40]])
```

Here
reg.coef_ → gives array

Exercise
hiring.csv.

Note: Understand raw implementation of single & multivariable linear regression using python without using sci-kit linear model.

Assumption of Regression Model

① Linearity : The relationship b/w dependent & independent variables should be linear.

② Homoscedasticity
Constant variance of the errors should be maintained.

③ Multivariate Normality
Multiple Regression assumes that the residuals are normally distributed

④ Lack of Multicollinearity
It is assumed that there is little or no multicollinearity in the data