7/01/2019

Machine Learning

→ Books

Understanding Machine Learning from theory to Algorithms
— By Shalev Shawartz and Shai Ben-David

Pattern Recognition and Machine Learning.
— By Christopher M. Bishop.

Machine Learning by Tom Mitchell

Introduction to Machine Learning by Alpaydin.

→ Learning:

A child learning alphabets

Monkey and Banana

pigeon Superstition

Bait Synonyms Shyness.

Generalisation
— comes from experience.
Extrapolating.

Machine Learning

→ Learning is the process of counting experience to knowledge.

→ Machine learning is about a program that they can learn from the available inputs to them and to convert it into knowledge and perform the same task.

→ Unseen data — The data not used for training.

prior knowledge or Inductive Bias.
- The knowledge of features of human and animal to differentiate.

features or attributes / - define the entity.
characteristics/properties.
- Algorithm must be
  - adaptive.

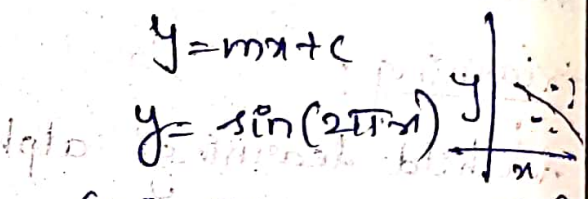
Types of learning:

⇒ Supervised learning. (guidance exists) labels, classes, benchmark.
- Unsupervised learning. -

$f(x_i) = y_i$.

$y = mx + c$

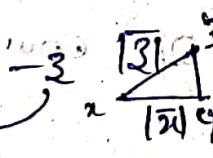
$y = \sin(2\pi x)$



x_i can be a vector $- (x_{i1}, x_{i2}, \ldots x_{im})$
where m is the number of featu

14/01/2020

Euclidean - to measure similarity / dissimilarity
Manhattan.
Pythagorean - $x - y - z$



Point similar

$d(x, x) = 0$, $d(x, y) = d(y, x)$

$d(x, z) < d(x, y) + d(y, z)$

Metric: A measure which satisfies these three properties is called a metric

Euclidean distance is L_2 norm. $= \sqrt{\sum (x_i - y_i)^2}$

→ If the points are similar then similarity will be or metric is low.

→ While trying to group the objects we need to consider similarity and dissimilarity.

(2)

→ Mapping $y = f(x)$

Choosing correct function. f_1, f_2, f_3, f_4 } among these.

A formal model:

domain Set - {Input you expect}

An arbitary set $\mathcal{X} = \{x_1, x_2, x_3 \ldots x_n\}$.

Label Set - {Target that we need to find}.

$\mathcal{Y}$ is Label set $\mathcal{Y} = \{0, 1\}$ for two class problems

$= \{+1, -1\}$

Training data - $\{(x_1, y_1), (x_2, y_2) \ldots\ldots (x_p, y_p)\}$.

You shouldn't use 100% for training.

We have to have some data for testing of the trained model.

Controlled Environment - Already have knowledge for which x we have which y.

(x_i, y_i)

Training (f) Testing

→ Learners output:

The learner is also called predictor (learning mapping of x and y.) provide x value from 'x' data and see 'y' value

hypotheses or a classifier.

The learner is respected to produce y_i $y = f(x)$

a production rule $y_i' = f(x_i)$

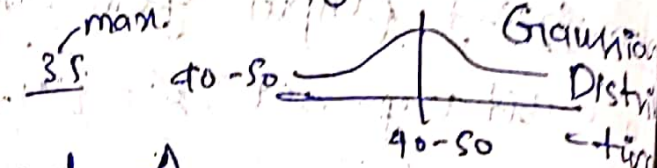
$h: \mathcal{X} \to \mathcal{Y}$.

Error $\Rightarrow y_i' - y_i$

→ We use A(S) to denote the hypothesis that a learning Algorithm A, returns upon recieving the training Sequence S.

→ Measure of Success:

We define the error of a classifier to be the probability that it does predict the correct label on a random data points guaranteed by the underlying distribution.



Formally, given a domain subset, $A \subset X$, the probability Distribution D, assigns a number, $D(A)$, which determines how likely it is to observe a point $x \in A$. We refer to A as an event and express it using a function $\Pi : X \to \{0,1\}$ namely, $A = \{x \in X : \Pi(x) = 1\}$.

{from TB Shalev and Shai Ben}.

// If $x \in$ to the class. then $\Pi(x) = 1$ or else the belongs $\Pi(x) = 0$.

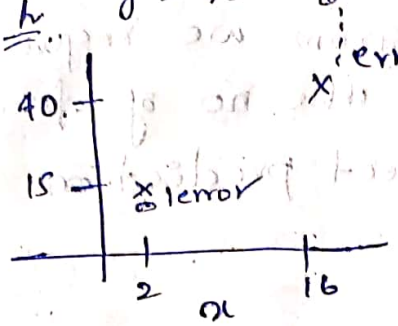
→ We define the error of prediction rule. $h : X \to y$ to be

$$L_{D,f}(h) = \underset{x \to D}{P}\left[\underbrace{h(x)}_{\text{hypothesis}} \neq \underbrace{f(x)}_{\text{actual}}\right] = D\{x : h(x) \neq f(x)\}$$

// The actual mapping verses the mapping which we derive — the difference b/w them is the error.

Ex: If $y = 2x + 9$ $x = 2$ $y = 13$, $y' = 14$ error = 1

$y' = 10x - 6$ $x = 16$ $y = 41$ $y' = 154$ error = high



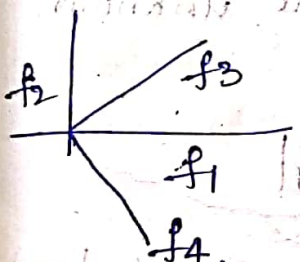
o → y
x → y' } Representation.

(5)

→ $\alpha_{D,f}(h)$ → has many synonyms such as

- Generalisation error
- The risk
- The true error of h.

Feature space: $x: f_1\ f_2 \ldots f_n$



0 0
0 1
1 0
1 1

- 2^n possibilities in the Data Set.

If data generated from restricted space, then it learns about all the similar data but the disadvantage here, is the model cannot learn about dissimilar data. (1)

If we want to represent all the real world data, we have to choose the data/samples such that it not only has the restricted space.

If Generalisation not happening correctly because of taking only the restricted area, then it may leads to Generalisation error

Curse of Dimensionality:

Dimension - size features/
As the number of dimensions grow, we require extension of size of samples or the no of features should grow up. for the correct prediction.

Optimisation:

Machine Learning Algorithms are Optimisation Algorithms.

optimisation in this case is $(y'-y)^2$, to make it positive.

Emperical Risk Minimisation: (ERM).

The goal of the algorithm is to find hs that maximises the error with respect to the unknown D and f.

Training error, $L_S(h) = \frac{|\{i \in [m] : h(x_i) \neq y_i\}|}{m}$

hypothesis. expected out

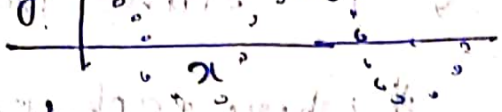
where $[m] = \{1, 2, 3, \ldots m\}$.

The learning Algorithm that comes up a predicto 'h' that minimises $L_S(h)$ is called Emperical risk minimisation or ERM.

Overfitting:

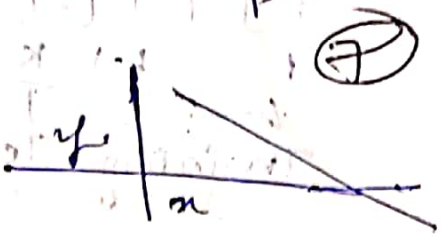
If we are given training data, instead of learning it generalises (Remembering) (Child learning tables) $\Rightarrow L_S(h)$ if value not there in S error huge.

Ex:- $y = \sin(2\pi x)$ (T.B. Bishop)



$y = w_0 + w_1 x + w_2 x^2 + \dots$

exponent value = 2 (quadratic)



If $y = w_0 + w_1 x$ only this taken then it becomes linear. (there by error hige.)

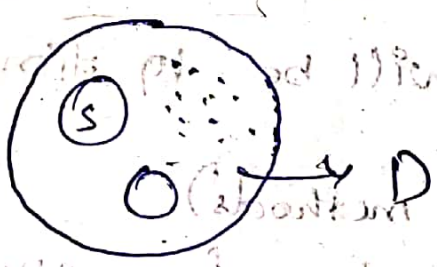
if value = 8 (almost fits)

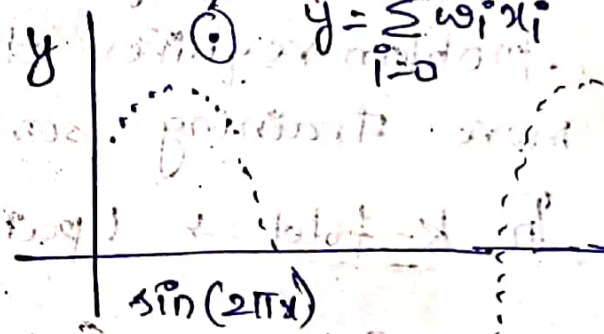
if exponent value increases more then there will be many oscillations [Because it tries to remember].

16/01/2020

$\epsilon_{Def}(h)$ Generalisation error.

$\epsilon_S(h)$ Emperical Risk.





Minimum Emperical Risk is 0

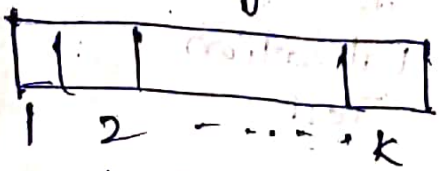
→ k-fold Cross-Validation:

Set

| Training | Test |

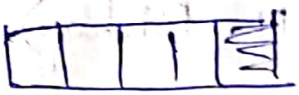
Create k-sets or partition k-sets and Randomly pick the training set and the testing set.

→



randomly pick from this k-divided set.

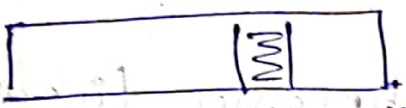
some partitions for training and some partitions for testing

→  → 1st time Training.

1 k-1 k

Training Testing.

* Where each partition will become test set.

 → 2nd time Training.

k-1 remaining train.

Test.

* We do the Cross fold validations in order to
  - decrease the error.
  - For unbiased learning.

More amount of time for training.

→ Prior Knowledge; (set of features to say the problem requires that).

more training some times creates confusion.

In k-fold → 1 part will be k times trained.

Sample Selection: (Sampling methods)

Random selection of samples from every area.

Random Sampling - Good enough for sampling.

→ Instead of k-fold Cross validation one we can use Validation Set method

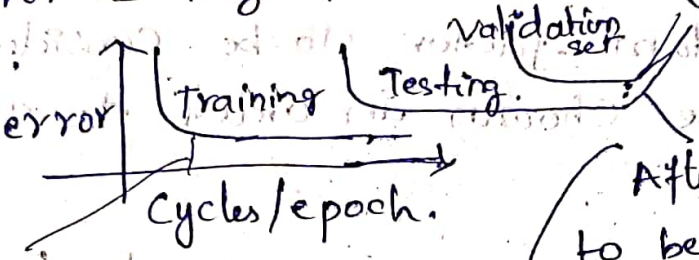
1 k-2 k-1 k → Test.

Training Validation Test

→ Test can only be tested after the learning Completed.

→ So, while the training is going on, $\frac{d_S(h)}{d} = 0$.

The Hypothesis is tested on the validation set.
So, In order to decrease the generalisation error (This set is used for testing generalisation and if error is huge, while training itself we get to know):



Learned in training test that on test data

After this point training to be stopped since model overlearns therefore, only Rembers.

→ for this purpose, Validation set is used.
(At that point error is checked and we stop training there).

→ Rote Learning (child learning tables). (Just Rembers)
(Rote)

→ To minimise the Emperical Risk
we go for Inductive Biasing.

→ A set of predictors called class of Hypothesis. is denoted by ff $h \in ff$ is a function mapping $X \to y$.

we have to pick one in order to minimise the Emperical Risk.

$$ERM_{ff}(S) \in \arg\min_{h \in ff} d_S(h)$$

$\sin(2\pi x)$

$h_1 = ax + b$

$h_2 = cx^2 + dx + e$

$h_3 = f + gx + hx^2 + \cdots +$

$H = \{h_1, h_2, h_3, \cdots h_n\}$

We have to choose among them which gives the best.

* which gives the minimum Error to be chosen

→ Q) Another important factor to be considered is which hypothesis to be chosen in order to avoid overfitting.

→ how many hypotheses to be considered, how many can a data set have? Can there be any upperbound?

*** Assumptions: in statistical theory.

The i.i.d assumption. (Independently, Identically Distributed according to Distribution D)

21|01|2020

→ Non Representative Set: Take samples from Distribution

→ Some Missing Regions: How many number of Samples are sufficient? (Probabilistic and approximation)

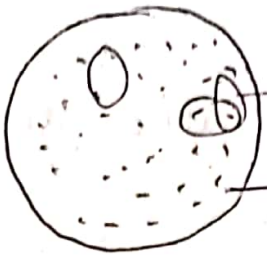
Samples (Independent and Identical)

→ The Training Samples are Independently and Identically distributed according to a Distribution D.

$S \sim D^m$ where 'm' is the size of S, and D^m denotes the probability of m-tuples indeed by applying

D to pick each element of the sample independently. of the other members of the tuple.

→ Q.) What is m?



→ sets of bad instances.

→ A set Consisting of 'm' number of samples

→ Confidence parameter } 2 parameters to estimate
→ Accuracy

What is the Confidence that, that Hypothesis provid, certain accuracy.

→ The Probability of getting a non representative Sample is δ and call $(1-\delta)$ the confidence parameter of our prediction.

Since, we cannot guarantee a perfect label, we choose another parameter for the quality of prediction, the accuracy parameter, ϵ

→ $\mathcal{L}_{(D,f)}^{(h)} < \epsilon$ for the Correct predictor

True Error

$f: X \to Y$

$h: S \to Y$.

training.

→ Let $S|_x = \{x_1, x_2, \ldots \ldots x_m\}$ is the instances 'm', the training set, if there are bad samples and if we find the upperbound for probability, then we get $\mathcal{L}_{(D,f)}^{h}$ high leads to $\mathcal{L}_{(D,f)}^{(h)} > \epsilon$ to, we can avoid them}

$\rightarrow D^m(\{S|_x | L_{(D,f)}^{(hs)} > \epsilon\})$

Let H_s be the set of bad hypothesis, that is

$H_B = \{h \in H : L_{(D,f)}^{(h)} > \epsilon\}$

Let $M = \{S|_x : \exists h \in H_B, L_S(h) = 0\}$ be the set of misleading samples.

$L_S(h)$ — Training error = 0

Leads to overfitting.

(ERM) – Called as

If this happens True error increases

$S|_x : L_{(D,f)}^{(hs)} > \epsilon\} \subseteq M$

(Subset of M)

M = Union of Bad set of examples.

$$M = \bigcup_{h \in H_B} \{S|_x : L_S(h) = 0\}$$

Hence, $D^m(\{S|_x : L_{(D,f)}^{(h)} > \epsilon\}) \leq D^m(M) = D^m\left(\bigcup_{h \in H_B} (S|_x : L_S(h) = 0\right)$

→ Lemma: For any two sets A and B and a distribution D' $D(A \cup B) \leq D(A) + D(B)$

$$D^m(\{S|_x : L_{(D,f)}^{(h)} > \epsilon\}) \leq D^m(M) = D^m\left(\bigcup_{h \in H_B} (S|_x : L_S(h) = 0\}\right)$$

$$= \sum_{h \in H_B} D^m(\{S|_x : L_S(h) = 0\})$$

$h \in H_B$ Bad Hypothesis.

$$\rightarrow D^m(\{S|_x : L_S(h) = 0\}) = D^m[S|_x : \forall\, h(x_i) = f(x_i)\}$$

$$= \prod_{i=1}^{m} D(x_i : h(x_i) = f(x_i))$$

$$\rightarrow = \sum_{h \in H_B} D^m(\{S|_x : L_S(h) = 0\}) = 1 - L_{(D,f)}^{(h)} \leq 1 - \epsilon$$

$$D(x_i : h(x_i) = y) = 1 - L_{(D,f)} \leq 1 - \epsilon$$

$$\therefore = \sum D^m(\{S|_x : L_S(h) = 0\}) = (1 - L_{(D,f)}^{h'})^m \leq (1-\epsilon)^m.$$

$$\rightarrow D^m(\{S|_x : L_S(h) = 0\}) \leq (1-\epsilon)^m \leq e^{-\epsilon m}. \quad \text{—(2.7)}$$

$$\rightarrow D^m(\{S|_x : L_{(D,f)}^{h_S} > \epsilon) \leq \sum_{h \in H_B} D^m(\{S|_x : h_S(h) = 0) \quad \text{— (2.9).}$$

$$\leq |H_B| e^{-\epsilon m}. \rightarrow \text{If log applied on b.s. we get}$$

→ Corollary : Let H be a finite hypothesis class. Let $\delta \in (0,1)$ and $\epsilon > 0$ and let m be an integer that satisfies $m \geq \frac{\log(|H|/\delta)}{\epsilon}$

→ Suppose, the hypothesis space H is the set of Conjunction of literals of n Boolean variables. In this 3 states for Boolean variable.

1. Negated.
2. Unnegated
3. It does not appear

$|H| = 3^{(n)} + 1$ ← adding conjunctions. (no of boolean variables)

3 states;

$+1$ is needed to represent false.

Which is the conjunction of any atom and its negation.

$m \geq \frac{\log(1/\delta)}{\epsilon}$

→ $\epsilon = 5\%$ confidence, δ 99% (δ)
(error less than 5% - accuracy tells)

→ If we want to guarantee at most 5% error and 99% of the time

$n = 30$
$\epsilon = 5\% = 1/20$
$\delta = 99\% = \frac{1}{100}$

The Sample complexity
$= 20 \times (n \ln 3 + \ln(100))$
$\approx = 752$

Using no of samples 752 we can generate error less than 5% and with confidence 99%.

$752 \leq < 2^{30} = 1,073,741,824$

→ And, the number of hypothesis
$= 3^{30} + 1 = 205,891,132,094,650$

Hypothesis Space = 2^n.

$\left(m = \frac{1}{\epsilon}(2^n \log^{\ln} + \log)\right)$

$$\boxed{m = \frac{1}{\epsilon}\left(2^n \ln 2 + \ln\left(\tfrac{1}{\delta}\right)\right)}$$

$n = 20$
$\epsilon = \frac{1}{20}$
$\delta = \frac{1}{100}$

$m = 20 \times (2^{30} \ln 2 + \ln 100) \approx 14,885,222,452$ samples.
14 Billion samples

→ To choose the hypothesis to do extrapolation we require prior knowledge (Inductive Bias)

Biased variance - trade off

→ For a finite hypothesis class, if the ERM rule with respect to that class is applied on a sufficiently large sample (whose size is independent of underlying distribution or labeling function) then the output hypothesis will be probably approximately correct.

(PAC learnability / Algorithm)

PAC Learning model: Probably Approximately Correct.

def: A hypothesis class H is PAC learnable if there exists a function $m_H : (0,1)^2 \to N$ and a learning algorithm with the learning property: for every $\epsilon, \delta \in (0,1)$, (belong to) for every Distribution D over X, and for every labeling function $f : X \to (0,1)$, if the realizable assumption holds with respect to H, D, f, then.

When running the learning algorithm on $m \geq m_H(\epsilon, \delta)$ i.i.d examples generated by D and labelled by f, the algorithm returns a hypothesis h such that, with probability of atleast $(1-\delta)$ (over the choice of examples), $L_{(D,f)}(h) < \epsilon$

22/01/2020. IT Lab – 2 ML (16)

→ Numpy python Matplotlib pandas python object c-structure in python

Introduction to Numpy: (Numerical python)
arrays homogeneous (same type)

np.<tab> – gives documentation of numpy methods

Understanding data types in python. obj-ref obj-type obj-size. obj_digit

np.array([1,2,3,4]) import array.
L=list(range(10))
np.array([1,3,3,4]), A=array.array('i',4)
A.

fancy indexing
masking
Bradcasting.

np.empty(3)

→ Creating arrays from scratch

arange(0,20,2) → 0,2 ---- 18

linspace(0,1,5) → 0, 0.125, 0.25, 0.75)

np.zeros(dtype=)

* Basics of Numpy Arrays

→

x – 2 dimension
b – 1 dimension
w – 1x2

$$sigmoid = \frac{1}{1+e^{-z}}$$

$$z = +(wx+b)$$

23/01/2020 → $h^* \in H$ which provides.

$\epsilon_p(h^*) = 0$

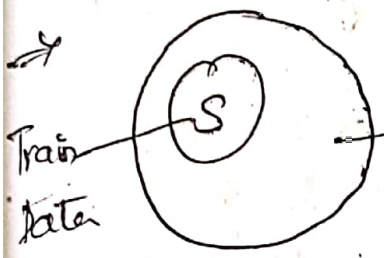
Realizability

Sample Complexity ← $m \geq \frac{(\ln|H|/\delta)}{\epsilon}$

Come up with a hypothesis that probably.

VC → dimension – we need to measure.

Prediction – 2 Levels.



Level 1 Yes / No (predicting class label)

Level 2 y value (predicting the value), (may be dependent on x) (Regression)

Realizability – Considering the finite hypothesis and choosing the correct hypothesis such that the generalisation error is minimised.

(depends on prior knowledge) – too much leads to over fitting,

→ Relaxing Realizability Condition – If finite hypothesis will be called agnostic PAC learning. (no assumption of H)

Instead of $x \to y$ will be mapping to the

Distribution $x \times y \to$ Distribution.

→ D is considered as Probability Distribution over

$x \times y$

→ The Emperical and True Error

$\mathcal{L}_D(h) \equiv \mathbb{P}$ (probability not a number)

$\mathcal{L}_D(h) = \mathbb{P}_{(x,y)\in D}(h(x) \neq y)$ { Emperical error same but the true error differs }

$\mathcal{L}_S(h) \equiv \frac{|\{x \in [m] : h(x_i) \neq y_i\}|}{m}$.

→ There must be some bench mark to take the Hypothesis (There will be 50% probability of success if classes are 2 (minimum performance))
min – 60%/50% in Agnostic PAC

→ Generalised Loss Functions: (0-1 loss function)

$x \to y$ some of the features doesnot give or help in finding y.

$x_1 - - - x_m$.

$x_i = (x_{i1}, - - x_{id})$

$\mathcal{L}_D(h) = \mathcal{L}_D(h, (x,y)) = \begin{cases} 1 & \text{if } h(x) = y. \\ 0 & \text{if } h(x) \neq y. \end{cases}$

This is called 0-1 loss function.

Square loss: $\mathcal{L}_D(h, (x,y)) = (h(x) - y)^2$.

Def: A hypothesis class H is agnostic PAC learnable with respect to a set 'Z' and a loss function. $l : H \times Z \to R_+$, if there exist a function $m_H : (0,1)^2$ and a learning algorithm with the following property:

For every $\epsilon, \delta \in (0,1)$ and for distribution D over $\mathcal{Z}$, when running the learning algorithm on $m \geq m_H(\epsilon, \delta)$ i.i.d examples generated by D, the algorithm return $h \in H$ such that with probability of atleast $1-\delta$,

$$L_D(h) \leq \min_{h' \in H} L_D(h') + \epsilon \qquad L_D(h) = 0.$$

28/01/2020.

Model Complexity: $\{h_1, h_2, h_3 \ldots h_n\}$ — set of hypothesis
$\{D_1, D_2, D_3 \ldots D_n\}$ — Data sets

h_1 can be used for solving D_1 Data set.

- There is no Universal learner.

Theorem: No free Lunch Theorem.

Let 'A' be any learning Algorithm for the task of binary classification with respect to the 0-1 loss over a domain X. Let 'm' be the any member smaller than $\frac{|X|}{2}$, representing, a training set size. Then there exists Distribution D' over $X \times \{0,1\}$ such that:

1. There exists a function $f: X \to \{0,1\}$ with $L_D(f) = 0$.
2. With the probability of atleast $1/7$ over the choice of $S \sim D^m$ we have that $L_D(A(S)) \geq 1/8$

Characteristics that we need to choose for the selection of model is crucial.

Bias Variance Trade Off :(↓↑)

Bias: [Comes from the fact that choosing a model.]

[Too much prior knowledge or restricting of choosing the hypothesis doesn't benifit]

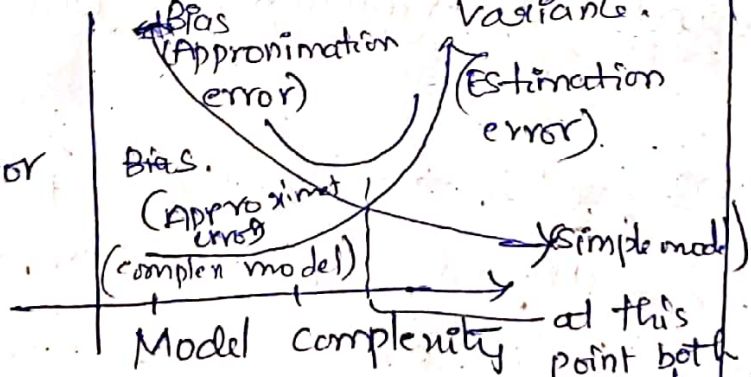
→ 2 factors that influence the model

- Bias
- Variance.

E = Appximation Error + Estimation Error

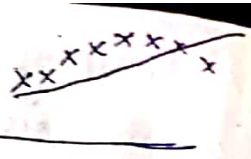
Appximation Error is Bias.
Estimation Error is Variance.

- what model
- how much it is able to fet in the model.



In Simple model Bias (decreases) that is approximation Error increases and In Complex model Variance. increases.

Lot of Error

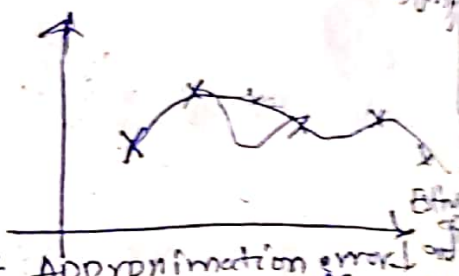


$\Theta = \Theta_0 + \Theta_1 x$ Underfitting happens.

model - Simple
- Rigid.

(19)

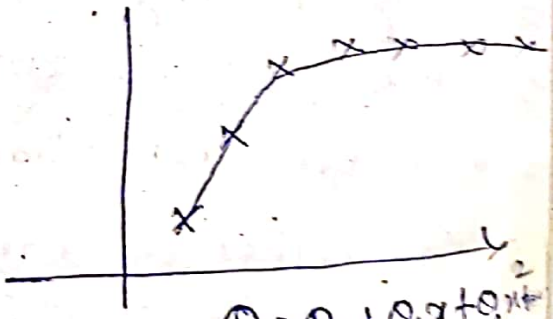
$S = \{(x_1, y_1), (x_2, y_2) \ldots (x_m, y_m)\}$



In this Approximation error

$\Theta = \Theta_0 + \Theta_1 x + \Theta_2 x^2 + \ldots + \Theta_m x^m$ (Overfit)

In this case Empeical Error decreases but the Generalisation or true er Increases.



$\Theta = \Theta_0 + \Theta_1 x + \Theta_2 x^2$

In this case it is bet -een overfitting and unde -itting (In this

→ NP- Non Deterministic Polynomial.

Problem: Verify the solution in the polynomial time- problem solution and verification in polynomial time.
non polynomial time

Np complete — Ans yes/no (Decidable problem)

Optimisation → Decision problem.

So, optimisation Based on constraint.
cannot be Np-complete
they are Np-Hard.

→ x → y mapping model such that true error at least error = 0.

→ Bias:

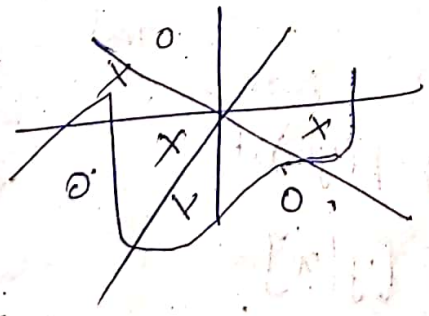
Bias are the simplifying assumptions made by a model to make the target function easier to learn.

Low Bias - Suggests that less assumptions about the form of the target function. Ex: Decision tree, knn, SVM.

High Bias - Suggests more assumptions about the form of target function. Ex: Linear Regression, Linear Discriminant analysis, Logistic Regression, etc.

→

Bias



→ Variance:-

Variance is the amount that the estimate of a target function will change if different training.

data is used.

Zero-Variance:- Suggests small changes to the estimate of the target function with changes to the training data Ex: Linear Regression, LDA, Logistic Regression.

High Variance:- Suggests large changes to the estimate of the target function with changes to the training data set Ex: Decision Tree, KNN, Support Vector Machine.

→ Loss function:- Cd

Loss Function (L) for Regression.

$$E(L) = \iint L(y, h(x))\, p(x,y)\, dx\, dy.$$

(y: actual Target, $h(x)$: predicted Target)

Sum Squared Error:

$$L(y, h(x)) = (y - h(x))^2.$$

$$E(L) = \iint (y - h(x))^2\, p(x,y)\, dx\, dy.$$

→ $$\frac{\delta E(L)}{\delta h(x)} = 2\int (h(x) - y)\, p(x,y)\, dy = 0$$

(for finding minimal value differentiate and equate it to 0 and)

Solving for $y(x)$

$$y(x) = \frac{\int y\, p(x,y)\, dy}{p(x)} = \int y\, p(y|x)\, dy.$$

$$= E[y|x]$$

Expected y given 'x'

$(h(x)-y)^2 = \{h(x) - E[y|x] + E[y|x] - y\}^2$

add and subtract.

$= \{h(x) - E[y|x]\}^2 + \{E[y|x] - y\}^2 + 2(h(x) - E[y|x])(E[y|x] - y)$.

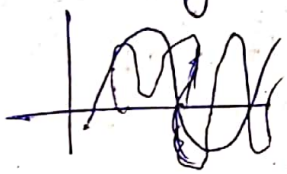
$E[L] = \int \{h(x) - E[y|x]\}^2 p(x)\,dx + \int \{E[y|x] - y\}^2 p(x)\,dx$.

approximation Error → Estimation Error

Training data set that defines y. not $h(x)$

$y = \sin(2\pi(x))$ — When tried for serial expansion, we try to control the oscillations using the term called Regularisation term.

$$y = \theta_0 + \theta_1 x + \theta_2 x^2 + \cdots + \theta_K x^K + \frac{1}{C}$$ Regularisation Term.



30/01/2020 Learnable Tactible problems

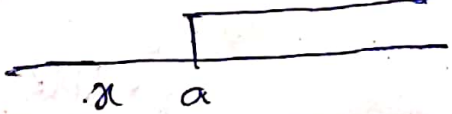
Theory proposed by Vapnik – Chervonenkis (VC):-

The PAC agnostic PACS etc has finite hypothesis,. But, this theory suggested that even though there are infinite hypothesis the problem is learnable.

Theory: (VC)

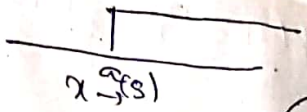
Even if the hypothesis class is infinite, if the VC dimension of the class is finite, then that particular model is learnable.

Shattering.: There may be threshold function (ex: step function below which is 0 above which is 1)



ha is a step function.

above 'a' is '1' below 'a' is 0



2 point class $C = \{0,1\}$: 2 classes

If one point is used, either '0' or '1' can be assigned as the target.

If two points are (0,0), (0,1), (1,0), (1,1)

x	0	0	1	1	4 types of class labels
S	0	1	0	1	

If the hypothesis is able to get such type of class labels then it is good to take such hypothesis.

→ ha - is the a - is the threshold.

$ha(x) = 1$ if $x < a$

$ha(x) = 0$ if $x \geq a$

Two points $c_1 \leq c_2$ $\quad a = c_1 + 1$

$ha(c_1) = 1$ $\quad c_1 < c_1 + 1$ ✓ true.

$a = c_1 - 1$ — The next threshold function

$ha(c_1) = 0$ since $c_1 < c_1 - 1$

$c_1 < a$

Consider another point c_2

$a = c_1 + 1$

$c_1 \leq c_2$

$c_2 < a$

@ $ha(c_2) = 1$

$ha(c_1) = 1$

possible

possible

No threshold can be choosen in this.

$ha(c_1) = 0$ } Not possible we cannot

$ha(c_2) = 1$ } choose such threshold.

$c_1 > a$

$ha(c_1) = 0$

$ha(c_2) = 0$

possible

$c_1 \leq c_2$

$c_1 < a < c_2$; $c_1 < a$

$ha(c_1) = 1$ $\quad ha(c_1) = 1$ $c_2 > a$

$ha(c_2) = 0$. $\quad c_2 = 0$

possible

→ Def: Let $\mathcal{H}$ be a class function from X to $\{0,1\}$. and let $C = \{c_1, c_2 \ldots, c_n\} \subseteq X$ The restriction of $\mathcal{H}$ to C is the set of functions from C to $\{0,1\}$ that can be derived from $\mathcal{H}$, that is (24)

$$\mathcal{H}_C = \{h(c_1), h(c_2) \ldots h(c_m)), h \in \mathcal{H}.$$

Where we represent each function from C to $\{0,1\}$. as a vector in $\{0,1\}^{|C|}$.

Def (Shattering): A Hypothesis class $\mathcal{H}$ Shatters a finite set $C \subseteq X$ if the restriction of $\mathcal{H}$ to C is the set of the all function from C to $\{0,1\}$.

That is $|\mathcal{H}_C| = 2^{|C|}$

Def: A dictionary of a set S is a partition of S into two disjoint subsets.

$$S = \{x_1, x_2, \ldots, x_m\}.$$

$$S^+ = \{x_1, x_6, x_8 \ldots x\}.$$

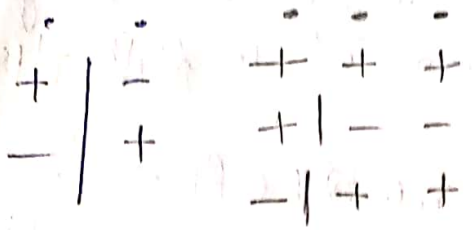
$$S^- = \{x_2, x_3, x_4, x_5$$

Def: A set of instances S is Shattered by hypothesis space $\mathcal{H}$ if and only if for every dichotony of S there exists some hypothesis in $\mathcal{H}$ Consistent with this dichotony.

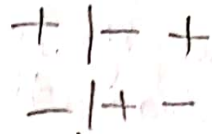
$\exists h \in H$, classifies all S^+ as positive.
all S^- as negative.

Ex: 1-dimensional
A Linear classifier in 1-dimensional

A linear classifier in 1 dimensional

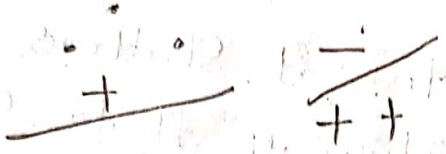


→ not satisfies for 3 cases.

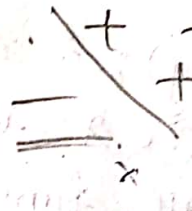


→ A linear classifier for 2 dimensional (2D)

→



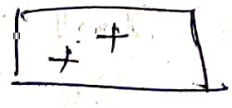
not satisfies for



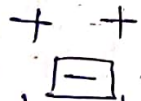
→ The Vc dimension of a hypothesis class 'H' denoted Vc dim(H) is the maximal size of a set $C \subset X$ that can be shattered by H. (dimension is d we can shatter d+1)

4/02/2020

→ If we take a



+ − * → 4 points $2^4 = 16$ labels



$2^5 = 32$ combinations – should be able to shatter those 32 combinations

Linear Predictors, Linear Regression:

→ Set of points and if there exists linear Relationship b/w them then they can be classified as Linear Regression. under.

Variable as one

$$y = mx + c$$

Multi variable as

$$y = w_0 + w_1(x) + w_2(x) + \cdots\cdots$$

$y = w_0 + w_1\phi_1(x) + w_2\phi_2(x) + \cdots + w_n\phi_{n-1}(x_{M-1})$

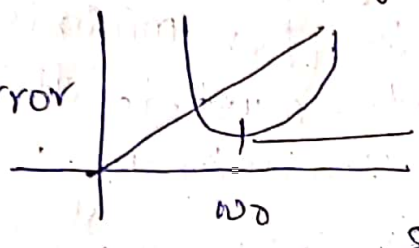
→ Linear combination of non-linear functions.

$\phi_j(x_j)$ can be non linear.

$$\phi_0(x_0) = 1$$

$$y = \sum w_i \phi_i(x_i)$$

fitting data is like optimisation problem.



we have to choose such that we can reduce error w.r.t weights so, it is optimisation problem.

→ $\varphi_j(x) = \exp\left\{-\frac{(x-\mu)^2}{2s^2}\right\}$ – Gaussian.

$\varphi_j(x) = \sigma\left(\frac{x-\mu_j}{s}\right)$ – sigmoid

$\sigma(a) = \frac{1}{1+\exp(-a)}$ → logistic

→ Since, we know the w's values, if new data comes it could be able to predict the y value.

$y = w_0 + w_1 x$

Functional form error:

$X = \{x_1 \cdots x_N\}$ $\quad x_j = \{x_1, x_2 \cdots x_m\}$

Sample (x_1) – 1 – – – – M

Sample (x_2) – 1 – – – – M

1 to M attributes

Sample set

$S = \{(x_1, t_1), (x_2, t_2), \cdots (x_N, t_N)\}$

class labels

$t = \{t_1, t_2, \cdots, t_N\}$:

t = target value.

y = Predicted value.

$y = wx + \epsilon$

$y = wx$

Least Mean Square → we need to find 'w' value.

(mean Square error) $E_D(w) = \sum_{i=1}^{N} (t_i - y_i)^2$ — Error.

to find appropriate 'w' value.

t_i → actual value; y_i → predicted value.

$$\frac{\partial E_D}{\partial w} = \frac{\partial}{\partial w} \sum_{i=1}^{N} (t_i - wx_i)^2$$ (For minimisation we must do the differentiation)

$$= 2 \sum_{i=1}^{N} (t - wx_i) \frac{\partial}{\partial w}(wx_i) = -2 \sum_{i=1}^{N} (t - wx_i) x_i = 0$$

$\sum_{i=1}^{N} t - wx_i = 0$ OR $x_i = 0$.

$t = wx_i$ $\quad \boxed{w = t/x_i}$ $\quad \sum_{i=1}^{N} (t - wx_i)\, x_i = 0$

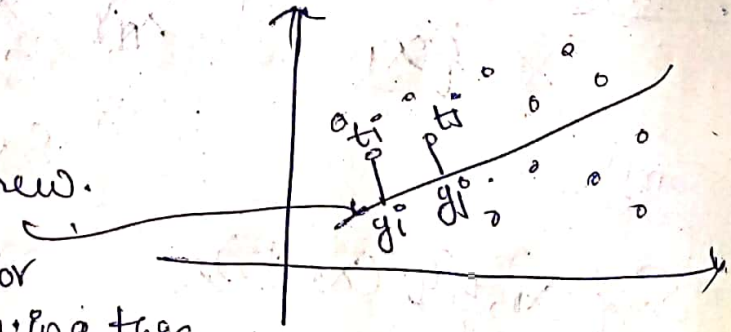
$\sum t x_i - w x_i^2 = 0$

$$w = \frac{\sum t x_i}{x_i^2} = \frac{\text{(Covariance } (x_i \text{ and } t))}{\text{Variance } (x)}$$

→ We wanted to get the 'w' value in order to minimise the error.

$y_{new} = w \times x_{new}$.

$y_{new} = \left(\frac{x_i \cdot t}{x_i^2}\right) x_{new}$.



for drawing this line

→ Least Means Square Method.

→ Gradient Descendent Method.

$$E_D(w) = \sum_{n=1}^{N} (t_n - y_n)^2$$

$$= \sum_{n=1}^{N} (t_n - (w_j \phi_j(x_j)))^2$$

$$y = \sum_{j=0}^{M-1} w_j x_j = \sum w_j \phi_j(x_j) = w^T \phi_j(x_j)$$

weight vector, Input vector

$$w = (w_0, w_1, \ldots w_{M-1})^T.$$

(In order to do matrix multiplication we have to do transpose)

$$\frac{d E_D(w)}{d w_j} = \frac{d}{d w_j} \sum_{n=1}^{N} (t_n - \sum w^T \phi(x_j))^2$$

$$= -2 \sum_{n=1}^{N} (t_n - \sum w^T \phi_j(x_j)) \phi_j(x_j)^T$$

$$\Rightarrow = -2 \sum_{n=1}^{N} (t_n \phi_j(x_j)^T - \sum w^T \phi_j(x_j) \phi_j(x_j)^T) = 0.$$

→ After Rearranging the terms

$$w_{ML} = (\Phi^T \Phi)^{-1} \Phi^T . t$$

Maximum Likelihood.

$\Phi_{n \times n}$ - matrix

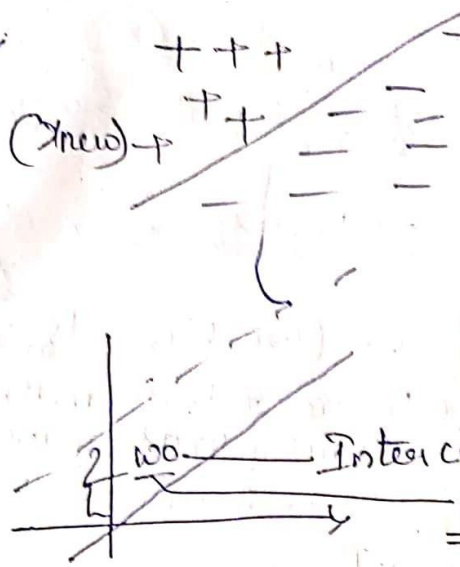
→ These are known as the normal equations for the least Square Problem.

Φ is an $N \times M$ matrix called the design matrix. whose elements are given by $\Phi_{nj} = \phi_j(x_n)$

$$\Phi = \begin{pmatrix} \phi_0(x_1) & \phi_1(x_1) & \cdots & \phi_{M-1}(x_1) \\ \phi_0(x_2) & \phi_1(x_2) & \cdots & \phi_{M-1}(x_2) \\ \vdots & & & \\ \phi_0(x_N) & \phi_1(x_N) & \cdots & \phi_{M-1}(x_N) \end{pmatrix}$$

→ $\Phi^\dagger = (\Phi^T \Phi)^{-1} \Phi^T$ is known as Moore pseudo inverse matrix.

Purose.

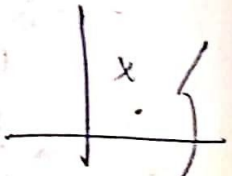


with w values we will be able to draw a line.

If we have the Binary classes such that will rain or will not rain then, we will be able to determine the x_{new} where it belongs

$\Phi_0(x_0) = 1$

Gradient Descent for Linear Regression:
(Slope)



where grad — nt highest we move at that path. assuming that we may reach our target soon.

Since least Mean Square involves inverse calculation we don't wanted to do such complex Calculations we Chook Gradient method.

$$y_j = \sum w_j \phi_j(x_j)$$

$$E_D(w) = \sum (t_n - y_j)^2$$

$$\frac{d}{dw} E_D(w) = \frac{d}{dw_j} \sum (t_n - y_j)^2$$

$$= -2 \left(\sum t_n - y_j\right) \frac{dy_j}{dw_j} = \underbrace{-2 \sum (t - y_j) \phi_j(x_j)}_{\text{factor}}$$

Types of Learning.
- Batch Learning
- Sequential or online Learning
  - as when samples come make changes to the system

→ Gradient Descent Algorithm for linear Regression

Initialise weights to 0 $w(1)=0$

For $i=1$ to ... until convergence (epochs)

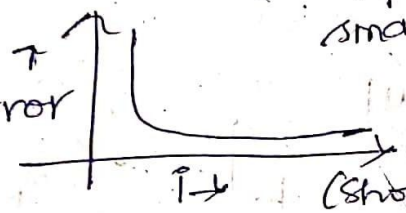
→ predict for each sample x_j using $y_k = \sum_{j=0}^{N} w_j \phi_j(x_k)$

→ Compute the gradient Descent of the loss as $\nabla E = \sum (t_j - y_j)\phi_j(x_j)$ not a matrix simple Calculation

→ This is a vector g.

→ update $w^{(\tau+1)} = w^{(\tau)} - \eta g$

η small quantity (learning parameter) 0 to 1



(should make sure not to overfit)

→ Regularized Least Squares: (In order to reduce the overfitting effect)

$E_D(w) + \lambda E_D(w)$

Total error $\frac{1}{2}\sum_{n=1}^{N} t_n - w^T\phi(x_n)^2 + \lambda E_f(w)$

λ is the Regularisation coefficient

Simplest form of Regularisation:

is $E_D(w) = \frac{1}{2} w^T w$ — weight decay.

$E_D(w) = \frac{1}{2}\sum_{n=1}^{N} (t_n - w^T \phi_n(x_n))^2 + \frac{1}{2} w^T w.$

$w = (\lambda I + \phi^T\phi)^{-1}\phi^T t$

Bernoulli. Gaussian.

1.) Bernoulli

$$p(x;\phi) = \phi^x (1-\phi)^{1-x}$$ DL

(31)

6/02/2020

2.) Gaussian

$$p(x;\mu;\sigma) = \sqrt{\frac{1}{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

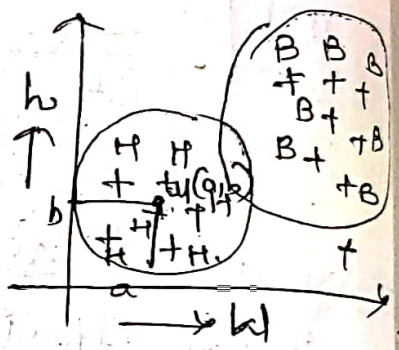
→

$$p(x;\vec{\mu};\Sigma) = \sqrt{\frac{1}{(2\pi)^n |\Sigma|}} e^{-\frac{1}{2}[x-\mu]^T \Sigma^{-1} [x-\mu]}$$

↑ 2×2 matrix

3.) Exponential $p(x,\lambda) = \lambda^x e^{-\lambda x}$

4.) Laplace. $p(x;\mu,\gamma) = \frac{1}{2\gamma} e^{-\frac{|x-\mu|}{\gamma}}$



μ – a, b Σ – function of r

in Gaussian

when x = a, b it gives highest or maximum probability

$$\Sigma = \begin{bmatrix} \sigma_w^2 & 0 \\ 0 & \sigma_h^2 \end{bmatrix}$$

variance in weight

Both are same for simplified Case.

variance in height

2×2

Joint variance in height and weight called as covariance

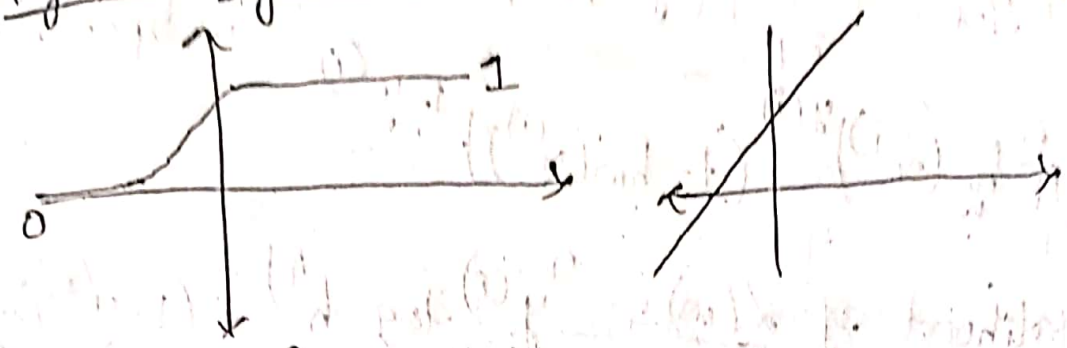
'w' and 'h' are independence. which means covariance is 0

6/02/2020 →

$\frac{d}{dw}$ =

assumption : discrimination line is not a straight line

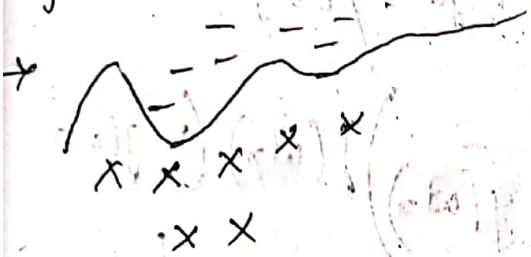
Logistic Regression :



In logistic function, the value will be b/w 0 and 1

Logistic function = $\frac{1}{1+\exp(-x)}$

$f: x \rightarrow (0,1)$



How do you learn to differentiate b/w those points?

→ If $g(z) = \frac{1}{1+e^{-z}}$ is the logistic function or sigmoid function.

$$g'(z) = \frac{d}{dz}\left(\frac{1}{1+e^{-z}}\right) = \frac{1}{1+e^{-z}}(e^{-z}) = \frac{1}{1+e^{-z}}\left(1 - \frac{1}{1+e^{-z}}\right)$$

$$= g(z)(1-g(z))$$

→ If we assume $P(y=1|x,w) = h_w(x)$

$$P(y=0|x,w) = 1 - h_w(x)$$

This can be written as $(h_w(x))^y (1-h_w(x))^{1-y}$.
(Binomial distribution)

Assuming n samples are i.i.d, we can write (38)

$$L(w) = P(y/x;w) = \prod_{i=1}^{n} P(y^{(i)}/x^{(i)}, w)$$ mutually independent

$$= \prod_{i=1}^{n} h_w(x^{(i)})^{y^{(i)}} \{1-h_w(x^{(i)})\}^{1-y^{(i)}}$$

Log likelihood of $L(w) = \sum_{i=1}^{n} y^{(i)} \log h^{(i)} + (1-y^{(i)}) \log (1-h(x^{(i)}))$

Maximisation $y = \sum w_i x_i$

$$h_w(x) = g(w^T x) = \frac{1}{1+\exp(-x)}$$

→ To maximise likelihood, use stocastic gradient rule:

t = target
y is target value
$h(x)$
$y = \sum w_i x_i$

$$\frac{\partial}{\partial w_j} L(w) = \left(y \frac{1}{g(w^T x)} - (1-y) \frac{1}{1-g(w^T x)} \right) \frac{\partial}{\partial w_j} g(w^T x)$$

$$= \left(y \frac{1}{g(w^T x)} - (1-y) \frac{1}{1-g(w^T x)} \right) g(w^T x)(1-g(w^T x)) \frac{\partial}{\partial w_j}(w^T x)$$

$$= y(1-g(w^T x)) - (1-y) g(w^T x) x_j$$

$$= (y - h_w(x)) x_j$$

(same like $\frac{t}{=} - y$)

Classification: To decide whether unseen sample belongs to which particular class based on the learning that has happened on the training data. (24)

may be binary or multi class classification.

(1) $C_1, C_2, \ldots C_k$

(2) k number of classes for example.

How multiclass classification happens?

Two methods—

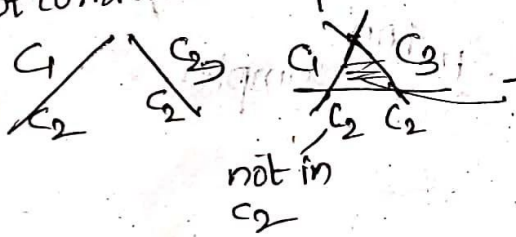
1. One Verses All (making Binary - C_1 / one other class: $C_2 \ldots C_k$

Disadv: What is common b/w $C_2 \ldots C_k$ may not be good enough.

2. Pair wise learning of multiple classes

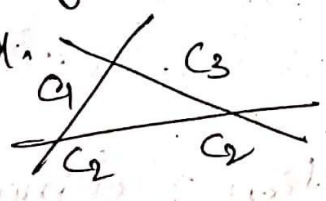
$(C_1, C_2), (C_1, C_3) \ldots, (C_1, C_k)$

not common keep in other class



this type of gray area may not be classified by any of the classifiers.

By using the binary classifier we may get contradictions.



If x belongs to C_2 and C_1 and C_3 again we get a contradiction.

11/02/2020 — If there is a Binary classifier.

◇ — Class 1 x — class 2

(35)

→ By adding extra feature for seperation of their Classes.

1d 10 — samples

2d 10^2 —

:

kd 10^d samples

K no of features

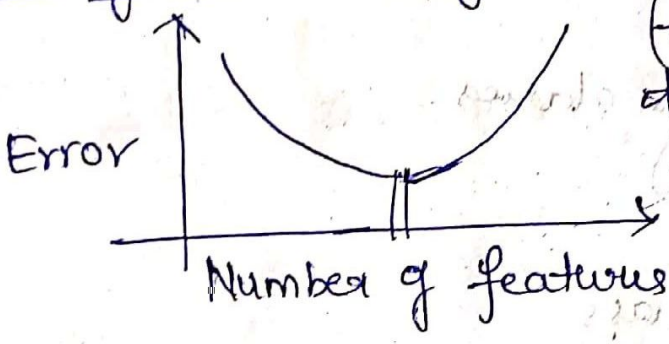
10^2 samples — in 2d — density requ

If we add extra feature density also increases (we want density to be same)

When the features grow the samples shouldn't grow

(Most of time we use Euclidean distance for similarity or dissimilarity)

Curse of Dimensionality:



(As the Dimensionality increases the samples may grow) ~~but~~

20. [ ] so, we may need 10^{400} samples.

NX400 20

P_1 — — — — — 400

P_2 — — — 400

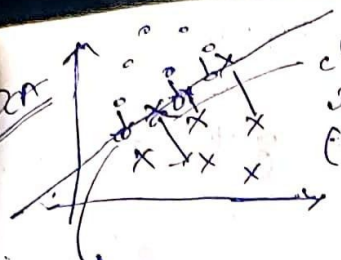
:

P_{10} — — — 400

Feature Selection - one way of reduction of dimensionality since we select only the particular features

Dimensionality Reduction: PCA, FDA

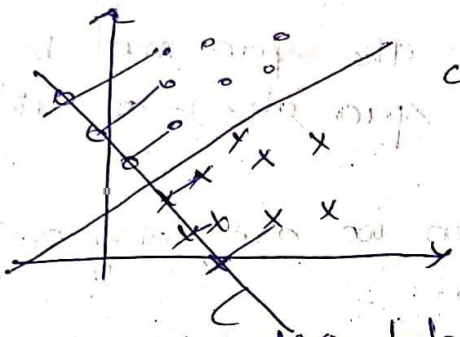
PCA



choose such an axis for which the variation in the samples is highest (less no of features to project on to the line).

No clear way of doing that.

(36)



In PCA can reduce into any number of features (can fail in some cases)

may be this helps. (Hence, we can differentiate b/w these classes)

we need to get the Direction of line such that it will be helpful to differentiate b/w those classes. what is the slope/bias needed to be identified this is FDA (Feature Dis Fischels Discriminant Analysis)

Fishels Linear DisGriminant: What is projection you need

kd (k-1) dimensions

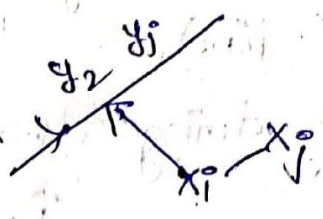
Main Idea: Find projection to a line such that samples from different classes are well seperated.

Suppose we have 2 classes and 'd' number of dimensions where n_1 samples are from class 1 and n_2 samples are from class 2 $n_1+n_2=N$ $X=\{x_1, x_2 - - - x_n\}$

Let the line lie in the direction given by the unit vector w

Scalar $w^t x_i$ $\forall i \leq N$ is the distance of the projection from the origin.

Let $y_i = w^t x_i$

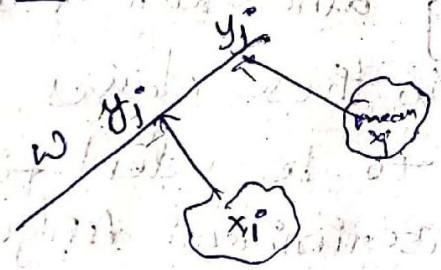


(Whoever is my neighbours in the space must be also my neighbours in the projected space also) — as said by

---

This we need to take care when we are doing projection in the lower dimensions also.

To measure the seperations between projections of classes 1 and 2 (we can make use of mean)



Does it require mean only or something else?

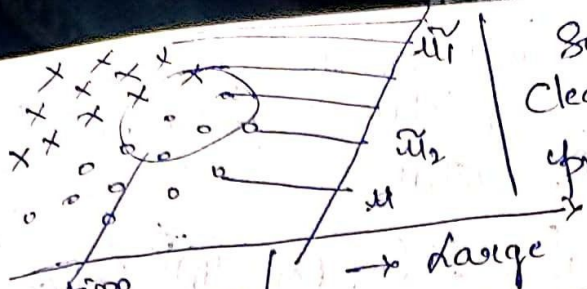
Let μ_1 and μ_2 be the means of classes C_1 and C_2 and $\tilde{\mu}_1$ and $\tilde{\mu}_2$ are the means of the projections

(If they are well seperated then points are also we seperated — assumption)

$$\tilde{\mu}_1 = \frac{1}{n_1} \sum_{x_i \in C_1}^{n_1} w^t x_i = w^t \sum_{x_i \in C_1} x_i = w^t \mu_1$$

$$\tilde{\mu}_2 = w^t \mu_2 = \frac{1}{n_2} \sum_{x_i \in C_2}^{n_2} w^t x_i = w^t \sum_{x_i \in C_2}^{n_2} x_i .$$

Projections along which we wanted to do projection



So, seperation can happen clearly by choosing the correct projection line.

seperation may not happen clearly here.

→ large variation in the data and seperation $\tilde{\mu}_1$ and $\tilde{\mu}_2$ is always large.

(38)

this line can be choosen (so, choosing the line is Linear Discriminant Analysis).

may be

Not only mean, we should also consider the variances.

The $\tilde{S}_1$ and $\tilde{S}_2$ FDA.

The Fisher's Criterion $J(w) = \dfrac{(\tilde{\mu}_1 - \tilde{\mu}_2)^2}{\tilde{S}_1^2 + \tilde{S}_2^2}$

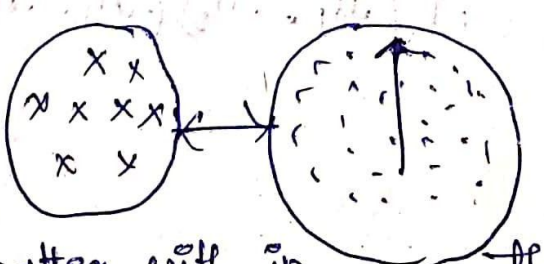
$$\tilde{S}_1^2 = \sum_{y_i \in C_1} (y_i - \tilde{\mu}_1)^2$$

$$\tilde{S}_2^2 = \sum_{y_i \in C_2} (y_i - \tilde{\mu}_2)^2$$

→ $\tilde{S}_1$, $\tilde{S}_2$ – scatter/variance.

Scatter in class – must be small.

Scatter accross the class – must be as large as possible



S_W = Scatter with in the class.

S_B = Scatter between the classes.

$S_w = S_1 + S_2$ (Scatter with in the class is sum of scatters of class 1 and class 2)

Using $y_i = w^t x_i$ $\quad \tilde{\mu}_1 = w^t \mu_1$ (37)

$$\tilde{S}_1^2 = \sum_{y_i \in C_1} (w^t x_i - w^t \mu_1)^2 = \sum_{y_i \in C_1} (w^t (x_i - \mu_1))^t (w^t (x_i - \mu_1))$$

Var in projected space

$$= \sum_{y_i \in C_1} w^t \underbrace{(x_i - \mu_1)(x_i - \mu_1)^t}_{S_1} w$$

$$= w^t S_1 w.$$

$$\tilde{S}_2^2 = w^t S_2 w.$$

$$S_1 = \sum_{x_i \in C_1} (x_i - \mu_1)(x_i - \mu_1)^t \quad \text{scatter}$$

$$S_2 = \sum_{x_i \in C_2} (x_i - \mu_2)(x_i - \mu_2)^t$$

$$\tilde{S}_1^2 + \tilde{S}_2^2 = w^t S_1 w + w^t S_2 w = w^t (S_1 + S_2) w = w^t S_w w.$$

Define $S_B = (\mu_1 - \mu_2)(\mu_1 - \mu_2)^t$

$$\tilde{\mu}_1^2 - \tilde{\mu}_2^2 = (w^t \mu_1 - w^t \mu_2)^2 = w^t (\mu_1 - \mu_2)(\mu_1 - \mu_2)^t w.$$

$$= w^t S_B w.$$

$$J(w) = \frac{w^t S_w w}{w^t S_B w}.$$

$$\frac{d}{dw}J(w) = \frac{\frac{d}{dw}(w^t S_B w)\, w^t S_w w - \frac{d}{dw}(w^t S_w w)\, w^t S_B w}{(w^t S_B w)^2} \quad (40)$$

$$= \frac{(2 S_B w)\, w^t S_w w - (2 S_w w)\, w^t S_B w}{(w^t S_B w)^2} = 0$$

$$w^t S_w w\,(S_B w) - w^t S_B w\,(S_w w) = 0$$

$$\frac{w^t S_w w\,(S_B w)}{w^t S_w w} - \boxed{\frac{w^t S_B w\,(S_w w)}{w^t S_w w}} = 0 \Rightarrow S_B w - \lambda S_w w = 0$$

Consider it as 1

$$S_B w = \lambda S_w w$$

$$\underbrace{S^{-1} S_B w}_{Ax} = \underbrace{\lambda w}_{\lambda x}$$

Converting it into Eigen value problem.

Eigen value problem $Ax = \lambda x$

$$S_w^{-1} S_B w = \lambda w$$

$S_B x$ for any vector x, points in the same direction as $(\mu_1 - \mu_2)$

$$S_B x = (\mu_1 - \mu_2)(\mu_1 - \mu_2)^t x = (\mu_1 - \mu_2)\frac{(\mu_1 - \mu_2)^t x}{\alpha}$$

$$= \alpha(\mu_1 - \mu_2)$$

$$S_w^{-1} S_B w = \lambda w$$

$$S_w^{-1} S_B (S_w$$

$$S_B = (\mu_1 - \mu_2)(\mu_1 - \mu_2)^t$$

$$S_w = \sum_{n \in C_1} (x_n - \mu_1)(x_n - \mu_1)^t + \sum_{n \in C_2} (x_n - \mu_2)(x_n - \mu_2)^t$$

(41)

$$J(w) = \frac{w^t S_B w}{w^t S_w w}$$

$$\underbrace{(w^T S_B w)}_{\text{Scalar term}} S_w w = \underbrace{(w^T S_w w)}_{\text{Scalar term}} S_B w$$

$$S_w w = S_B w$$

$$(S_w^{-1} S_w) w = S_w^{-1} S_B w$$

$$w \propto S_w^{-1}(\mu_1 - \mu_2)$$

(Aim: To find out the direction of the line required for the projection)

$$S_w^{-1} S_B w = \lambda w$$

$$S_w^{-1} S_B \underbrace{(S_w^{-1}(\mu_1 - \mu_2))}_{w} = S_w^{-1} \propto (\mu_1 - \mu_2)$$

$$= \frac{\propto}{\lambda} \underbrace{(S_w^{-1}(\mu_1 - \mu_2))}_{w}$$

↳ If we find out this project all the points

Eigen value problem → $Ax = \lambda x$ form.

$C_1 = [(1,2)(2,3)(3,3)(4,5)(5,5)]$

$C_2 = [(1,0)(2,1)(3,1)(3,2)(5,3)(6,5)]$

$$C_1 = \begin{bmatrix} 1 & 2 \\ 2 & 3 \\ 3 & 3 \\ 4 & 5 \\ 5 & 5 \end{bmatrix} \qquad C_2 = \begin{bmatrix} 1 & 0 \\ 2 & 1 \\ 3 & 1 \\ 3 & 5 \\ 6 & 5 \end{bmatrix}$$

(U2)

① μ_1 = mean $C_1 = [3 \quad 3.6]$

μ_2 = mean $C_2 = [3.3 \quad 2]$

② Compute the Scatter Matrices

$$S_1 = 4 \times \text{cov}(C_1) = \begin{bmatrix} 10 & 8.0 \\ 8.0 & 7.2 \end{bmatrix}$$

$$S_2 = 5 \times \text{cov}(C_2) = \begin{bmatrix} 17.3 & 16 \\ 16 & 16 \end{bmatrix}$$

③ Within the class scatter.

$$S_W = S_1 + S_2 = \begin{bmatrix} 27.3 & 24 \\ 24 & 23.2 \end{bmatrix}$$

④ The inverse of $S_W = S_W^{-1} = \begin{bmatrix} 0.39 & -0.41 \\ -0.04 & 0.47 \end{bmatrix}$

⑤ Optimal line direction $w = S_W^{-1}(\mu_1 - \mu_2) = \begin{bmatrix} -0.79 \\ 0.89 \end{bmatrix}$

6. Final step is the actual projection of x_i onto line

$$y_1 = w^t x_1 = w^t c_1^t = [-0.79 \quad 0.89] \begin{bmatrix} 1 & 5 \\ \vdots & \vdots \\ 2 & 5 \end{bmatrix} = [0.8, \ldots, 0.4]$$

$$y_2 = w^t x_2^t = [-0.79 \quad 0.89] \begin{bmatrix} 1 & 6 \\ \vdots & \vdots \\ 0 & 5 \end{bmatrix} = [-0.65, \ldots, -0.25]$$

43