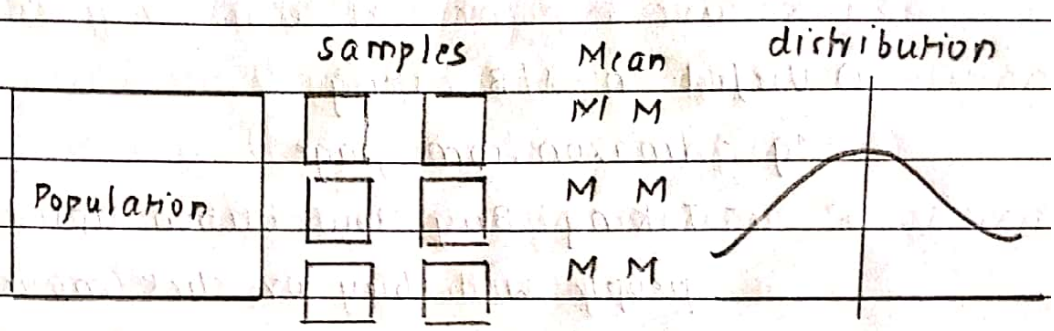
Date 01-12-19
Page 1

1. Central Limit Theorem

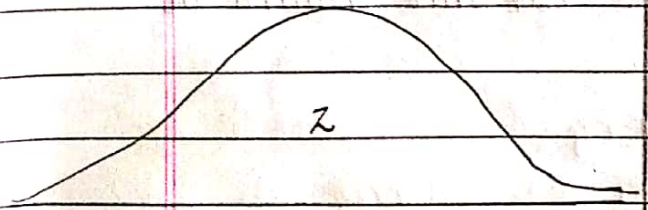
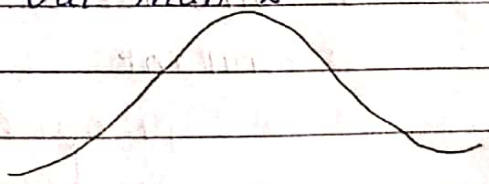


- Mean (sample) ~ mean (population)
- Sample error = SD of sampling distribution

$$= \frac{\sigma}{\sqrt{n}}$$

- $SD = \frac{\sigma}{\sqrt{n}}$
- If σ is very large it causes uncertainty or huge spread of data from population itself.

2.

Z-test	T-test
	- Slightly more spread out than Z 
	- Use when we don't know about SD of population

3. Hypothesis testing

1) Useful in A/B testing

eg:- Amazon.com page

- Taking Buy Now button upside so people will buy or click more

0.2 0.20001

Apply hypothesis

2) At what point sample size is large enough to have correct test.

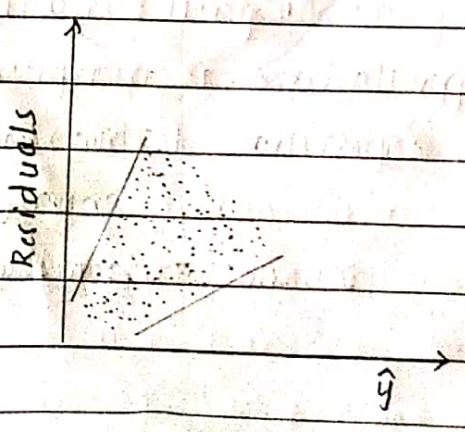
4. Linear Regression

ASSUMPTIONS:

1) No auto-correlation: values should not be dependent on past values

2) No SLR on TSA because TSA has auto correlation (TSA: Time Series Analysis)

3) Homo-skedastic: variance should be conitant



4) Residuals: difference between actual and predicted values, should be normally distributed

5) Hypothesis function:

$h(x) = \theta^T x$

Minimize error (actual - predicted value)

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} [y_i - \theta^T x_i]^2$$

6) $\sqrt{(x)^2} = \pm 5$, $(\sqrt{x})^2 = x \quad \forall \; x \in \mathbb{R}$

$\sqrt{25} = \sqrt{-5} \times \sqrt{-5}$

$= \sqrt{5i} \times \sqrt{5i}$

$= -1 \times 5$

$= -5$

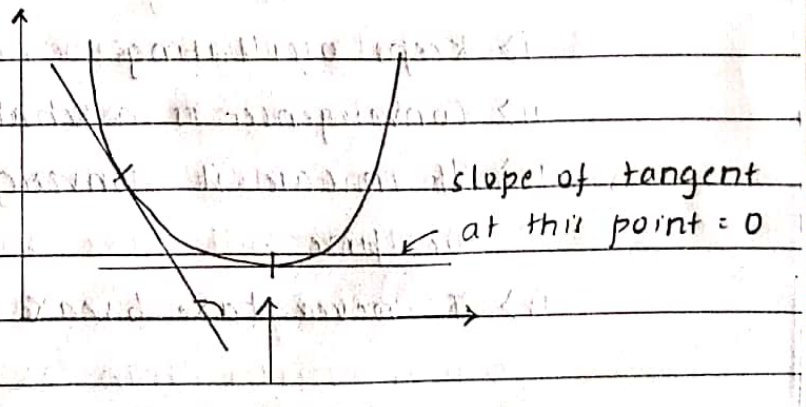
5. **Gradient Descent**: optimization algo used to find parameters (coefficients) of function

1) $\theta_i = \theta_i - \alpha \frac{\partial J(\theta)}{\partial \theta_i}$ that minimizes cost function.

α: Learning Rate

$\frac{\partial J(\theta)}{\partial \theta_i}$: Gradient / slope (Partial derivative)

2)

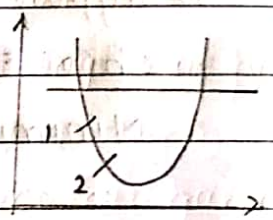


3) Determine slope of curve?

→ Take a point, draw a tangent

1. slope of line = $\frac{y_2 - y_1}{x_2 - x_1}$

4) Since the line is constant, so taking 2 points is of no use



5) Slope of tangent = slope of curve at that point

6) Slope = $\frac{\partial J(\theta)}{\partial \theta_i}$ ← direction
← magnitude

7) α : control of jump of choosing points

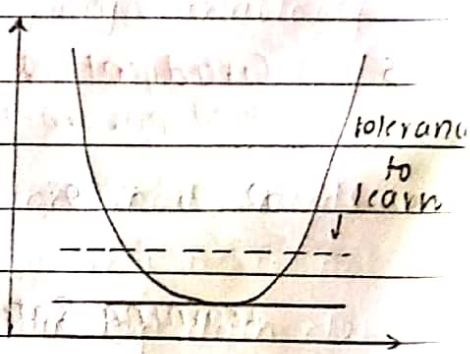
6. α and tolerance

1) LαHT :

i) No high parameter

ii) Accuracy is low

iii) T stops too soon



2) HαLT :

i) Keeps oscillating

ii) Convergence is a challenge

iii) If in case it converges, accuracy will be fine

iv) T: never take breaks

3) LαLT :

i) Extremely long time to converge

4) H2 HT:

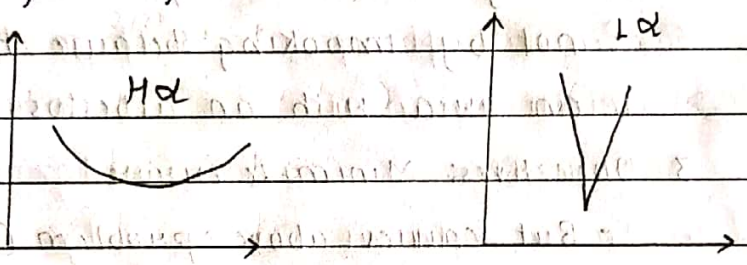
i) Very likely that it will converge and accuracy is low

7. Scaling

Q. Why to do scaling in LLR?

→ 1) Impact speed

- Std^n
- Norm^n
- Min-Max
- Unit vector



3) We cannot have same α for both, but we can use only 1

4) ∴ Both will have different αs

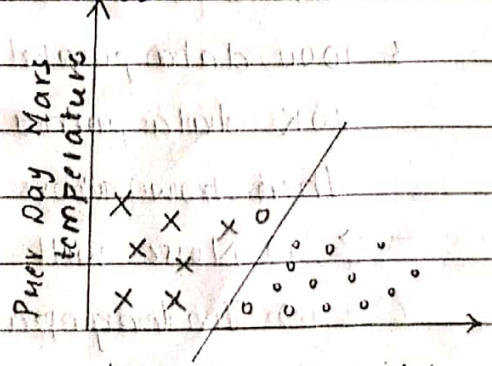
8. Overfitting

1) Won't predict well on unseen data given model has performed well on training

2) What stops model from overfitting?

→ i) Predict rain in Mumbai

ii) We have 1 variable that has nothing to do with our model



iii) Another thing that Prev Day Humidity allows to make our model overfit is number of features.

iv) Degree of polynomial

Overfitting: when a model learns detail & noise in training data to the extent that is -vely impact test

Date
Page 6

9. NOISE

1. Data that is unexplainable by given variable
2. What is the source of noise?
   → i) Eg: Data IP / Tracking error
   - having cigarettes per day but lying about it
   ii) Features of cigarette smoker having lung cancer but 1 person has cancer not by smooking because he lives in an area with an asbestos factory nearby
3. Objective: Minimize error
   - But causes above problem (i.e. noise in data) because we have trained our model to consider everything so it will take each and every variable

10. Complexity of Model

1. The amount of data the model needs to store to save itself
2. Increase the size of training data to avoid overfitting
3. 1000 data points, 200 noisy points
   10K data points, 2K noisy points
   Then how above statements holds true?
   → i) Noise will take random to the model
   ii) Finding pattern in 200 noise points is easier than 2K noise points
4. Overfit: going away from signal

Prevent Overfitting: Cross Validation, Remove features, Ensembling, Regularization

Date
Page 7

5. Reduce complexity
   - Reduce features
   - Reduce degree
   - Reduce hidden nodes
   - Reduce tree depth
   (i) Whatever helps you to reduce model complexity will reduce noise
6. Increase training data
7. Update objective function (do that in Regularization; Don't just minimize error but minimum error + something)

* Underfitting:
   - Increase model complexity
   - True for 5 only

II. Regularization

1) Lasso cost function:

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} [h(x_i) - y_i]^2 + \lambda \Sigma |\theta|$$

↑ small difference; ↑ value increases

2) Ridge cost function:

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} [h(x_i) - y_i]^2 + \lambda \Sigma \theta^2$$

Underfitting High bias

- Increase model complex.

Date
Page 8

3) Sparse : fairly spread out

4)

LASSO (sparse data lot of features)	RIDGE
1. No. of observations < no. of features	1. Don't have any info about features
2. Feature selection	2. Asymptotically to 0
3. Removes all of multi-collinear variables.	3. If multi-collinearity is encountered use ridge, it will keep it
4. Sparse data as it contains many zero features.	4. Ridge also does zero but rarely
5. Speed is much slower compared to ridge	5. Speed is relatively higher

5) Lasso removes features so why it is not faster? OR Why ridge is faster than lasso?

→ i) Ridge is a continuous function so we can use differential function

ii) Lasso cannot, we have to use conjugate differential equation

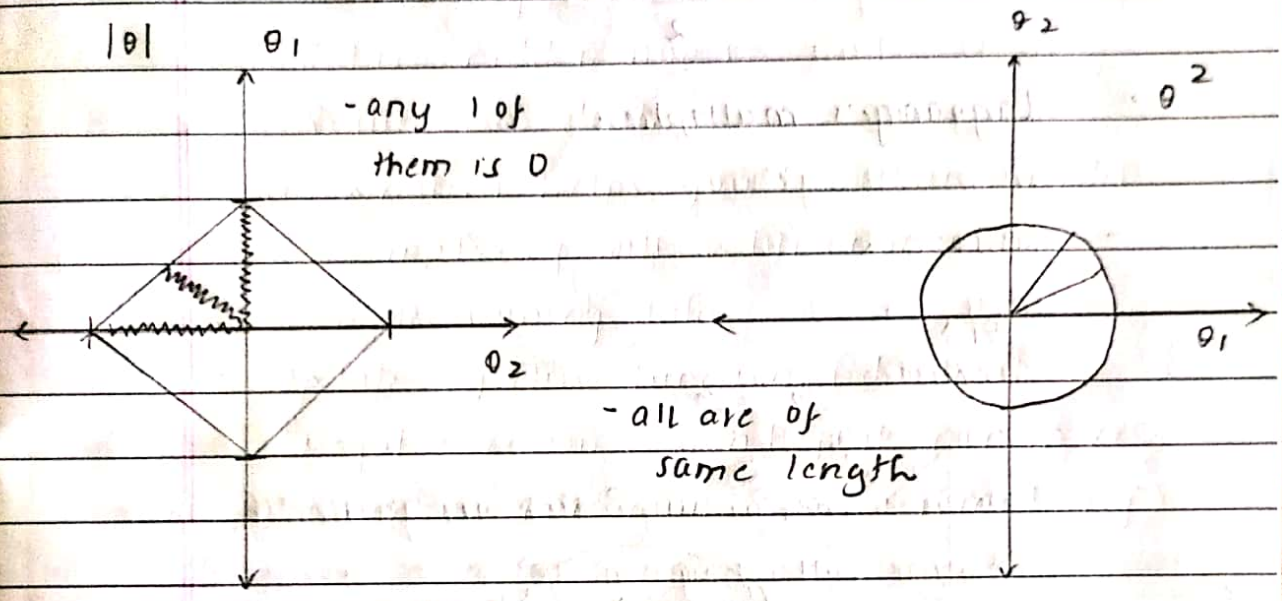
6) Kernel function: Increases dimension; sparses the data it easy to seperate them

High Variance:
- more data
- Reduce model complex.

Page 9

Q How does lasso removes features while ridge doesn't?

→



→ Elastic Net:

i) 1 extra parameter than ridge to tune (to bridge lasso and ridge)

ii) Sparse

iii) $J(\theta) = \frac{1}{m} \sum_{i=1}^{m} [h(x_i) - y]^2 + \lambda_1 \Sigma|\theta| + \lambda_2 \Sigma\theta^2$

→ Ridge:

i) less tuning

ii) Speed↑

→ overfit: difficult to seperate

Underfitting: model is to weak to fit our dataset

Date 08-12-2019
Page. 10

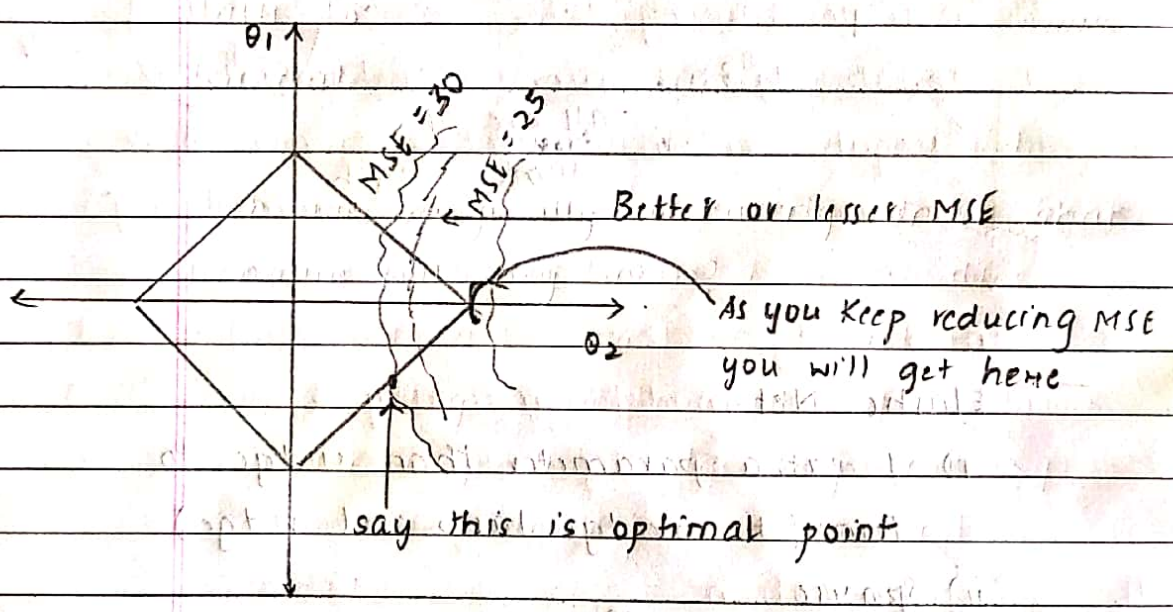
- Use dataset or features or model complex

- Ridge $[J(\theta)] = \frac{1}{2} MSE + \lambda \Sigma \theta^2$

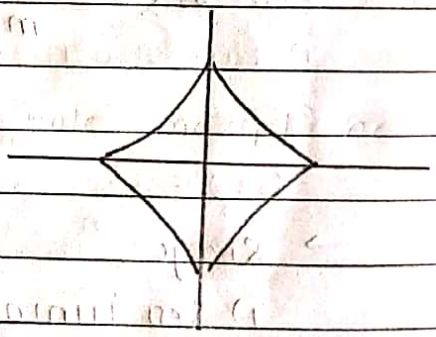
- Lasso $[J(\theta)] = \frac{1}{2} MSE + \lambda \Sigma |\theta|$

- Lagrange's multiplier

- Lasso E

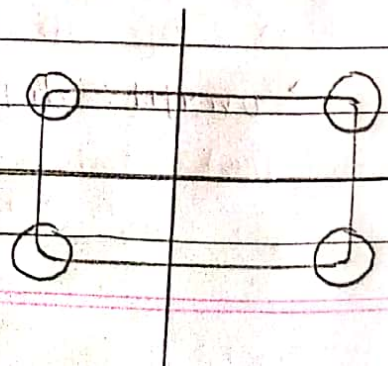


- $L_{0.5} = \lambda \Sigma |\theta|$
$= |\sqrt{\theta_1}| + |\sqrt{\theta_2}|$



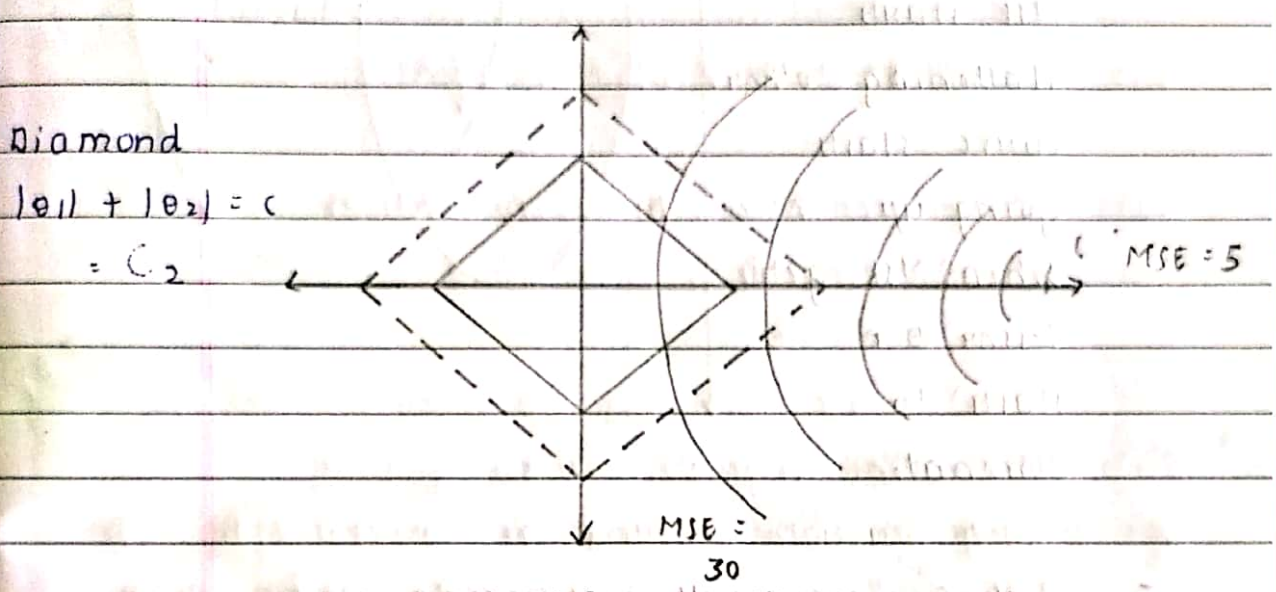
- $L_4 = \lambda \Sigma \theta^4$

Minimum MSE at circled points



← Model is not complex enough to accurately capture relationships between features & target variable

- Lasso keeps them within check



- Regularization doesn't mean reducing θ at every iteration
- It means finding best way to have minimum error while keeping the model simple

- λ(|θ1| + |θ2| = 20) , λ(|θ1| + |θ2| = 10)

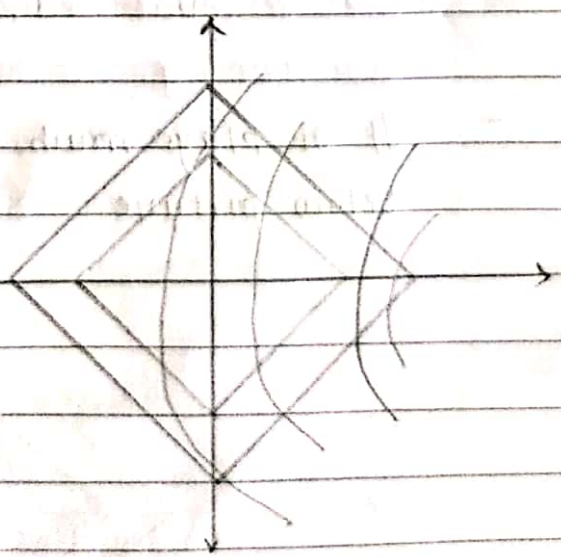
100 0.01

Ridge = $\theta_1^2 + \theta_2^2$

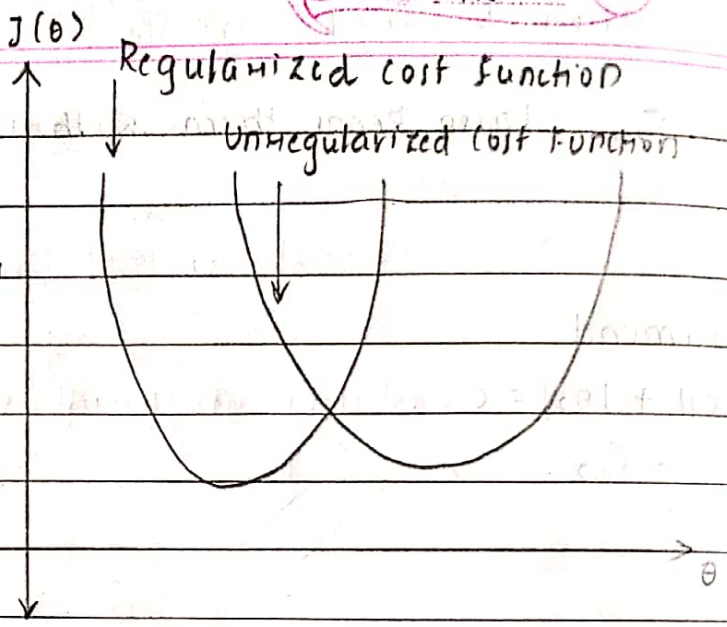
= 10000 0.01

Lasso = |θ1| + |θ2|

= 100 0.1

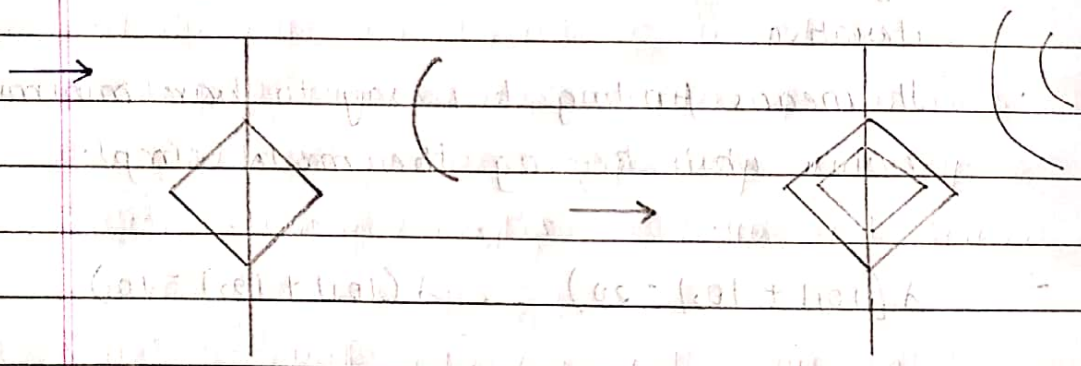


So ridge is more likely to reduce θ1 = 10000 than lasso which is more likely to reduce both

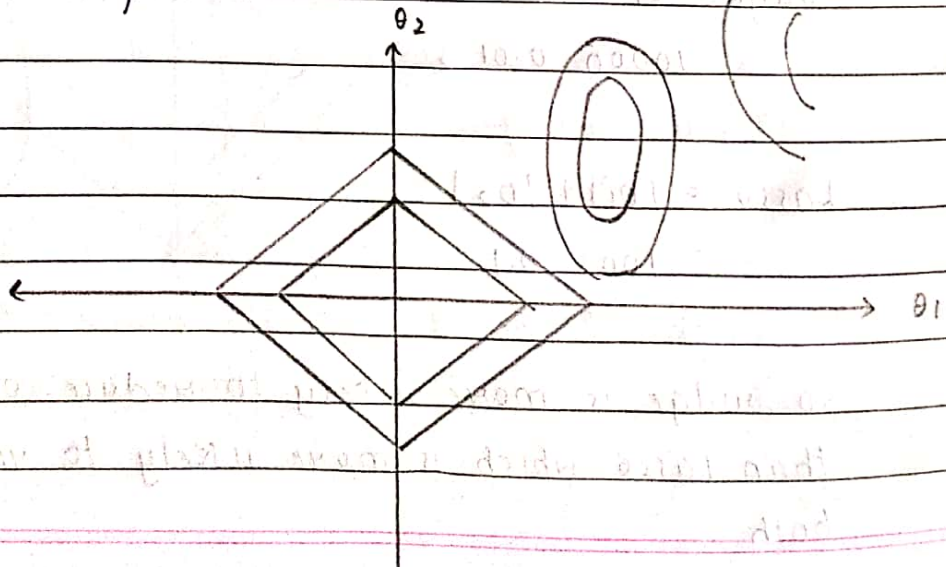


- As reduction in MSE starts flattening, other curve starts going up
- When MSE gets lesser θ starts to dominate

- Both are marginally increasing. Under what scenario will they meet?



- If θ_1 plays much bigger role, we get the below picture



- Lasso: In most cases we are likely to hit edge
  Ridge: - Less likely that θ_s becomes 0
  - Less likely to hit edge

- Regularization sees if it can reduce θ without impacting error

Role of λ:

1. How much diamond will expand in context to contour.
2. If λ is small, diamond will expand more, much more compared to contour.
3. If diamond is static, θ_s are zero.
4. λ controls rate of expansion
5. If $\lambda = 0$, focus on MSE, no effect from θ
6. λ is very high, optimal point will be very close to zero.
7. If λ is very low, minimize MSE
8. If λ is very large, the error is high, so error contour will expand fast, so diamond won't expand, optimal point will be diamond close to (0,0)
9. Decide λ using cross validation.

Elastic Net

1. $\frac{1}{2}$ MSE $+ \lambda_1 \Sigma|\theta| + \lambda_2 \Sigma \theta^2$

2. There are three things to consider:
   i) diamond
   ii) contour
   iii) circle

3. Minimize large θ values

4. When multiple features are correlated to each other, lasso will keep 1 and remove the rest

5. Ridge will try and keep these variables

6. Elastic net will depend on λ_1 and λ_2

7. E.N. will try to make cluster of variables; it will either go the ridge way and try to preserve the variables or lasso and remove them all.

8. Use manual training and cross validation for tuning.

AIC and BIC

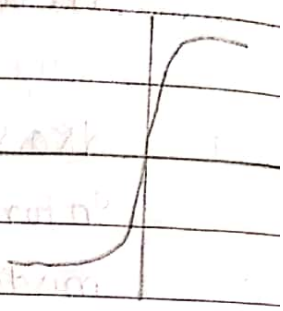
1. Akaike Information Criteria and Bayesian Information Criteria both help to compare models.
2. BIC sort of assumes 1 out of say 10 models is absolute true model and assign values accordingly.
3. AIC is not as strict as much as BIC on parameters.
   ∴ BIC is more likely to pick a simple model.
   AIC might choose complex model.
4. AIC → $2K$ + $\boxed{\ln RSS}$ → nMSE
   BIC → $K \cdot \ln(n)$ + $\boxed{\ln RSS}$ → nMSE

   K: number of parameters
   n: number of datapoints (observations)

   If $n > e^2$, BIC > AIC (BIC has or puts higher weightage on K than AIC)

5. Overfitting → simpler model (observations increase, overfitting decreases)
   Underfitting → complex model
6. If sample size is small, it is shown AIC tends to pick better model.
7. $n < K^2$: always go for AIC (recommended)

Logistic Regression:



1) Sigmoid function: $\hat{y} = \frac{1}{1+e^{-\theta^T x}}$

2) Error: mis-classification

3) Output is either zero or 1.

$\theta^T x = -\infty, \hat{y} = 0$

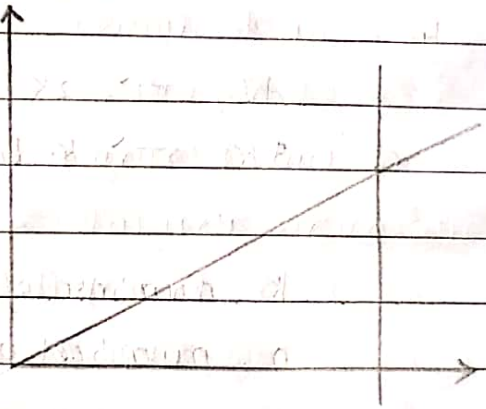
$\theta^T x = \infty, \hat{y} = 1$

4) Dependent variable is categorical.

Q. Why we use sigmoid in Logistic Regression?

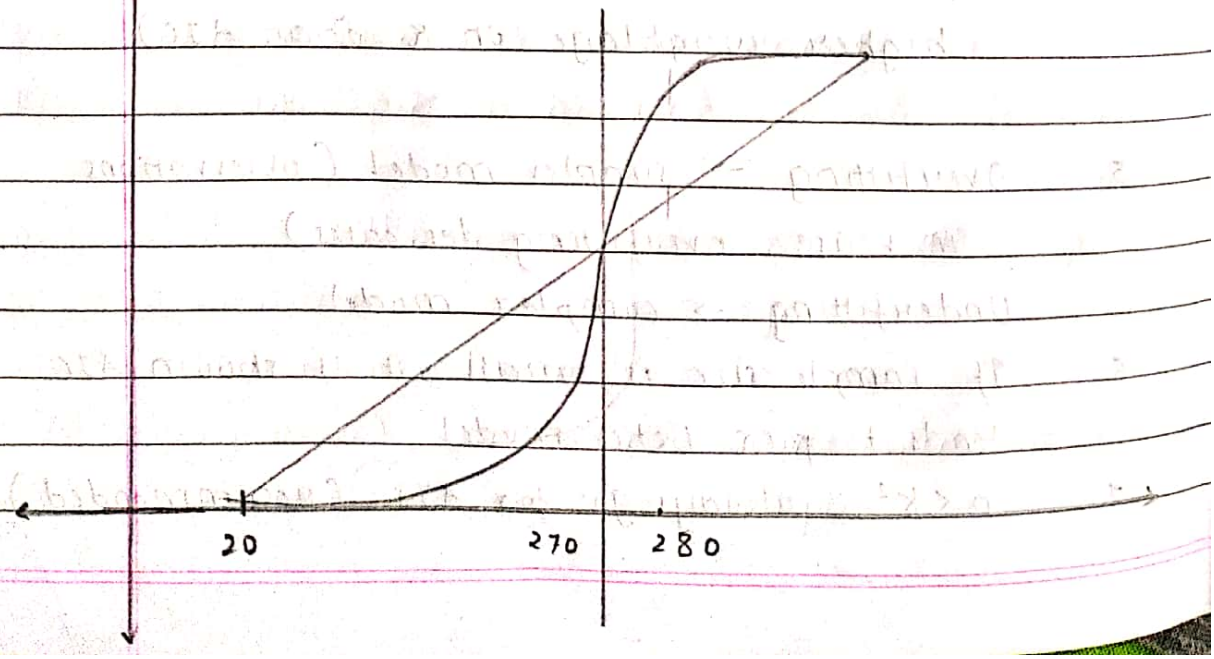
OR

Q. Why we cannot use below graph?



→ sigmoid: anything on other side of boundary, wrong side point get penalized

5) $\hat{y} = 0.002x - 0.04$

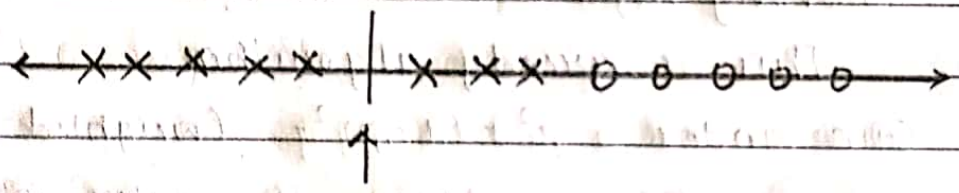


- Let's say x = 280 (predicted value)
- As per linear R. : 0.52 / 0.66
  sigmoid : 0.66 / 0.85

6) Sigmoid induces bigger errors for mis-classification and allows it to learn much faster.

7) Sigmoid gives much error for points close to boundary.

8) Sigmoid pushes values either to 0 or 1, so very few in between values

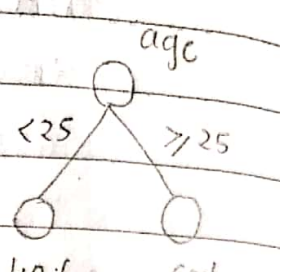


sigmoid will move this line much faster than linear because of high error

9) There is no arbitrary change in function for continuous function

- not differentiable
- cannot use GD and other

Decision Trees

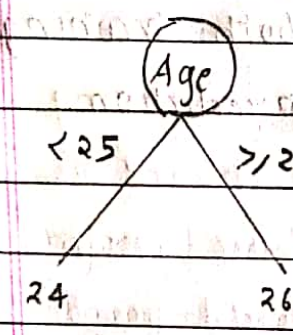


1. It keeps splitting till all datapoints belong to same class.
2. Goal is to make homogeneous.
3. How to split?
   → We consider bunch of matrices
   → Try to optimize at every step with the assumption that it is optimized at every step.
   This is greedy algorithm
4. Gini index: $p^2 + (1-p)^2$ (weighted gini, Entropy: $-p \cdot \log p$ ratio, etc.)
5. You decide split using variance reduction.
6. DT cannot be used for 2 datasets which are identical but belong to different classes because we have nothing to split.
   So there is a hit with accuracy
7. DT almost always overfits.
8. Gain is minimal so we cannot split.
9. Overall depth of the tree.

10.

Categorical	Continuous
- Gender	- Age
- states (Resides in India)	- Income
- shortlisted	- Time spent on a page

D.T



- Here, we are not able to differentiate between adult or middle aged person because of such a split

Logistic

0.5 x states RI

26 27 55

↓ ↓ ↓

12 13 28

- Here, we are able to tell apart

II.	Decision Trees	Logistic
1)	Works better with categorical data	1) Works better with continuous data
Relative → 2)	Can build almost any type of non-linear boundary / fit very complex function	2) less likely to overfit.
3)	Robust to outlier	3) Not robust
4)	OK with missing values (doesn't consider values for that feature but for others)	4)
5)	Better with class skew (class imbalance)	5) Cannot handle much class skew.
6)	Matters in ability to minimize error	6)

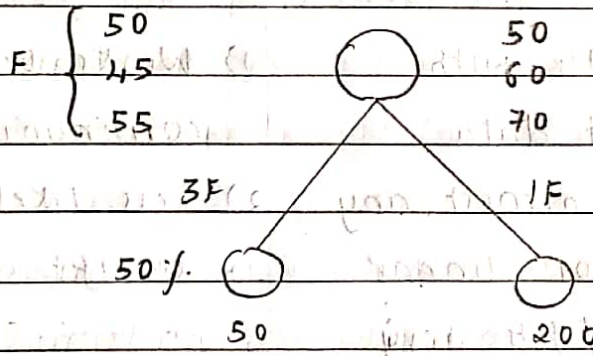
7)	Class skew will be handled at lower levels, so it increases depth (reduce depth for outliers.)	7) Faster than DT at both training and prediction.

12. How we use DT when target variable is continuous?

OR

How will you decide split?

→ 1) Gini index or entropy cannot be used.



2) So we use Reduction in variance for continuous target variable
- checks by how much variance has been reduced
- As much as variance has been reduced better it is

13. Suppose we want to check whether someone will buy the product or not.

→ 1) Check time spent on a page

2) Now the question arises whether where will we put DT boundary?

Gini index (default)	Entropy
- Faster because we take square	- log takes time

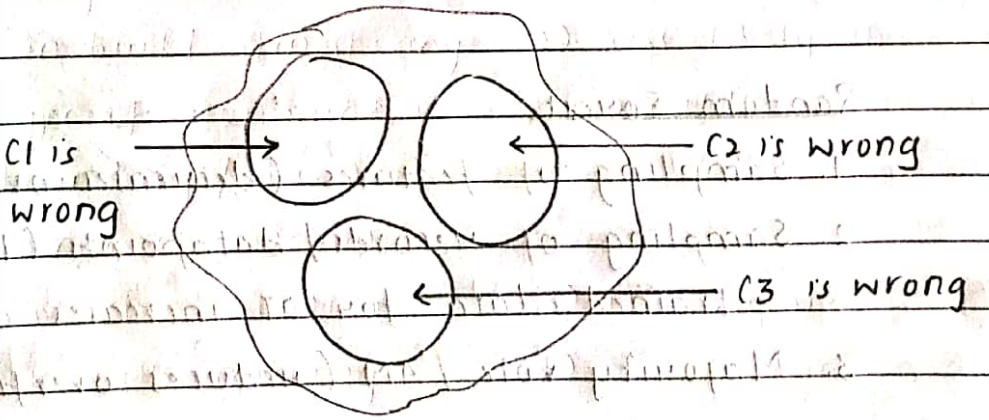
Ensembling

1. For binary classification, weak classifier is

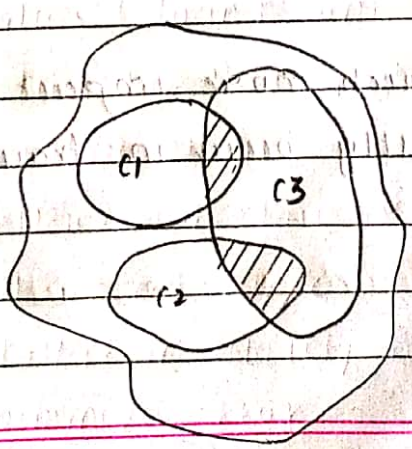
Accuracy

0 — 0.5 — 1

2.



3.



- This get better and better when we have more classifiers
- For overlapping if 2 classifiers vote incorrect then the classifiers are wrong.

4. It would be correct only in shaded region else is incorrect.
So we are at disadvantage by taking ensemble.

5. For ensemble, the classifiers should be accurate by more than 50%. otherwise ensemble is useless i.e. each classifier should be better than random.

Bagging

1. Bootstrap aggregation
   - sampling with replacement
2. How would bagging with DT work?
   What type of model will we get if combine BA and DT?
   ⟶ Decision Tree

Random Forest

1. Sampling of features (decreases overfitting)
2. Sampling of records / datapoints (less trained data for DT increases overfitting)
3. Majority vote / AM (reduces overfitting)

* Features have much more impact than data
* Noisy labels mostly occur in fraud detection

	Random Forest	Decision Trees
1.	less likely to overfit	1. Interpretable
2.	Handles noisy labels	2. Speed (especially prediction speed)
3.	Tells feature importance closer to root are imp	3. Less likely to under-fit

Boosting:

1. $h_1, h_2, h_3, \ldots$ → +1 / -1

2. Boost misclassified datapoints in next classifier

$\sum w_i = 1$, $E = \sum_{\text{wrong}} w_i$

Boost stronger classifiers

$h(x) = \text{sign}[\alpha_1 h_1(x) + \alpha_2 h_2(x) + \alpha_3 h_3(x) + \ldots]$

Start with, $w_i = \frac{1}{N}$ → Pick h_t (classifier) that minimizes E_t (Error at that time)

ADA BOOST ✓

Calculate α_t ← Pick α_t (weight you want to give to classifier)

3. $w_i^{t+1} = \frac{w_i^t}{z} e^{-\alpha^t h^t(x_i) y_i}$ {Misclassified than -ve}

$$\alpha^t = \frac{1}{2} \ln \frac{1 - E_t}{E_t}$$

If correct, $w_i^{t+1} = \frac{w_i^t}{z} e^{\ln \sqrt{\frac{E_t}{1-E_t}}}$

If wrong, $\frac{w_i^t}{z} e^{\ln \sqrt{\frac{1-E_t}{E_t}}}$

$$\frac{1}{z}\left(\sqrt{\frac{E_t}{1-E_t}} \sum_{\text{correct}} w_i^t \sqrt{\frac{1-E_t}{E_t}} \sum_{\text{wrong}} w_i^t\right) = 1$$

$$z = 2\sqrt{E_t(1-E_t)}$$

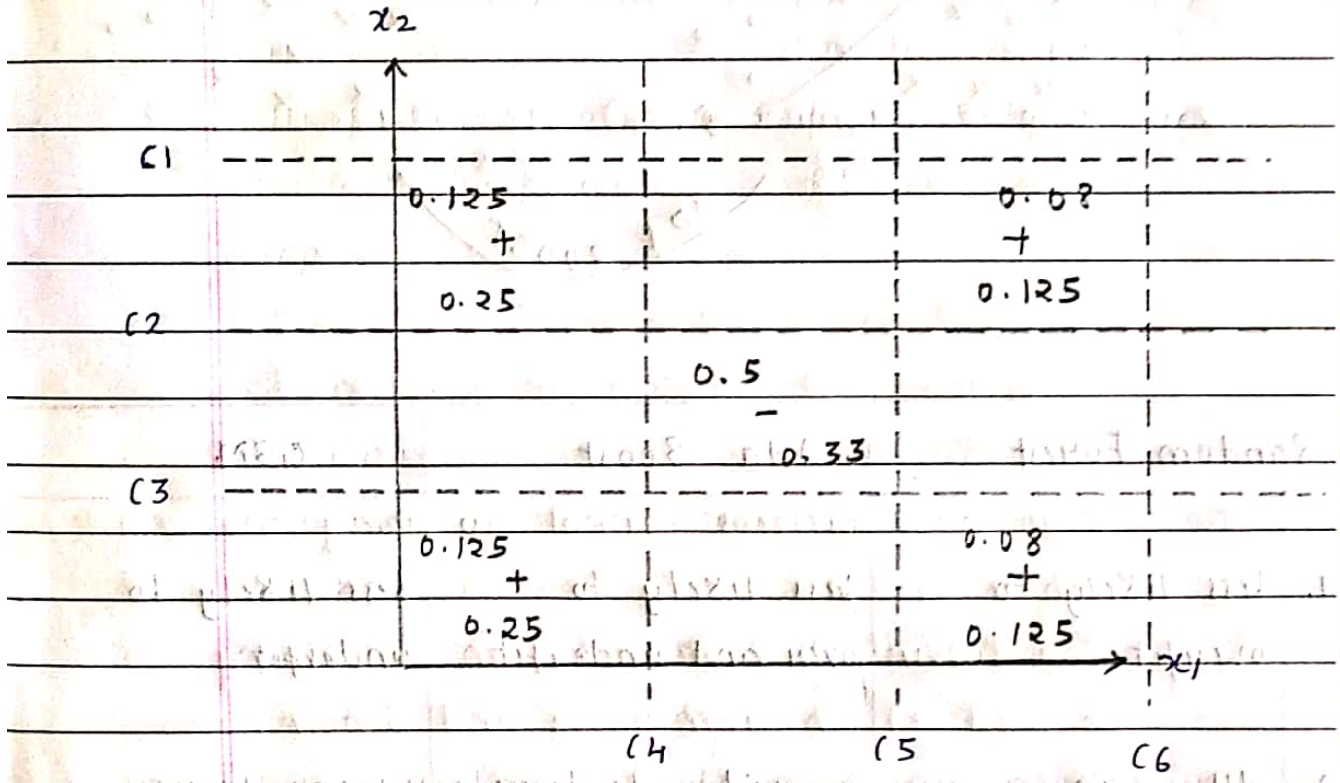
$$\sum_{\text{correct}} w_i^t = \frac{1}{z}\sqrt{\frac{E_t}{1-E_t}}$$

$$w_i^{t+1} = \frac{w_i^t}{z}\sqrt{\frac{E_t}{1-E_t}} = \frac{w_i^t}{2} \cdot \frac{1}{1-E_t}$$

$$\sum_{\text{correct}} w_i^{t+1} = \frac{1}{2}$$

All correct and incorrect points get weightage of $\frac{1}{2}$

If 80% of your points are correct, its collective weight will be 1/2 and remaining 20% of incorrect points will be 1/2

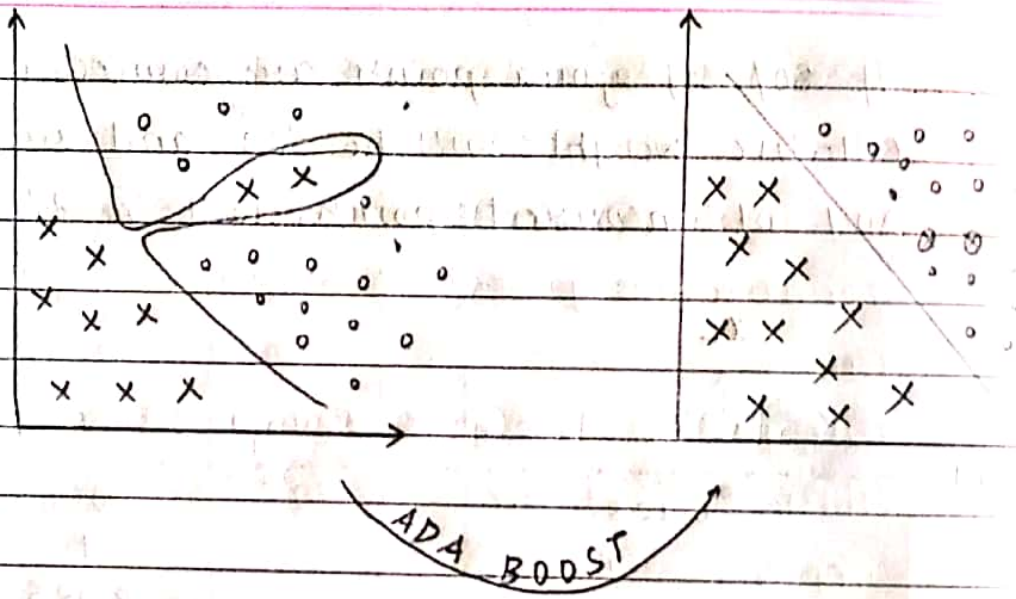


Gradient Boosting:

- learn error in previous step

$y = h(x_i) + \text{error}$	$y_i = h^t(x_i)$
$err^t = h^{t+1}(x_i) + err^{t+1}$	$y_i = h^t(x_i) + h^{t+1}(x_i) +$
$err^{t+1} = h^{t+2}(x_i) + err^{t+2}$	$h^{t+2}(x_i)$
⋮	

Till error becomes very small



	Random Forest	Ada Boost	GBM
1.	less likely to overfit	Less likely to over and underfit	less likely to underfit
2	All can	provide feature	importance
3.	Works with noisy labels	Better with class skew.	Better with class skew
4.	Training speed when parallel	Prediction speed	Prediction speed
5.	Longer to predict	Relatively fast	Relatively fast
6.	Missing values, outliers	Lot of hyper parameters	Lot of hyper parameters.

XGBoost:

1. GBM + regularization + speed
2. Faster for parallel

21-12-2019

Naive Bayes (Generative Model)

1. $P(C, d)$ read as joint probability of C and D
2. In generative model what you learn is joint probability.
3. $P(C, d) = P(d|C) \cdot P(C) = P(C|d) \cdot P(d)$
4. $P(C|d) = \frac{P(d|C) \cdot P(C)}{P(d)}$

   For 1 datapoint $P(d)$ is constant because $P(d)$ is fixed.
5. $\text{Max}[P(C|d)] = \text{Max}[P(d|C) \cdot P(C)]$
6. $P(d|C) = P(x_1, x_2, x_3, x_4, x_5 | C)$

   (Joint probability of all features | class)

   $= P(x_1, x_2, x_3, x_4 | C \cdot x_5)$

   $= P(x_1, x_2, x_3 | C \cdot x_5 \cdot x_4)$
7. In NB all these features are independent of each other.

   $= P(x_1|C) \cdot P(x_2|C) \cdots P(x_5|C)$
8. NB chooses class with highest probability.

Multinomial NB Classifier:

1. $C_{map} = \underset{c \in C}{\text{argmax}}\ P(x_1, x_2, x_3, \ldots, x_n | c) \cdot P(c)$

2. $C_{NB} = \underset{c \in C}{\text{argmax}}\ P(c_j) \prod_{x_i \in X} P(x|c)$

Problems with Maximum likelihood:

1. Zero probabilities cannot go away, no matter other evidence.
2. For solving this we perform smoothing.
   - Add +1 in numerator and denominator
   - So our data never have zero. This is called Laplace smoothing.

$$P(w_i|c) = \frac{count(w) + 1}{count(c) + 1}$$

Other types of smoothing:
- (i) Good tuning
- (ii) Mason A

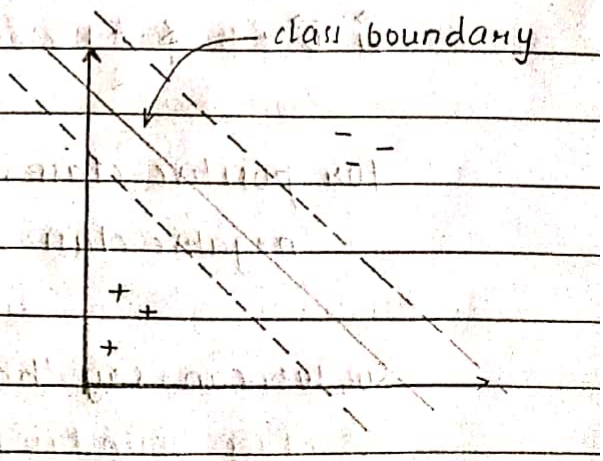
Advantages of NB

1. Very fast.
2. Training is also quick.
3. Low storage requirements.

4. Extremely robust to irrelevant features (IF)
IF cancel out each other without affecting features.
5. Least likely to overfit.
6. Very good in domains with many equally important features. (Decision trees suffers form fragmentation in such cases, when data is little)
7. NB takes probability and multiplies by 2 if 2 features are same.
8. Works really well on exceptionally high dimensional dataset.
9. Works good for text classification.

Q. Why don't we use NB before logistic, linear...?
→ Discriminative classifiers (less likely to underfit) works really well than generative model.

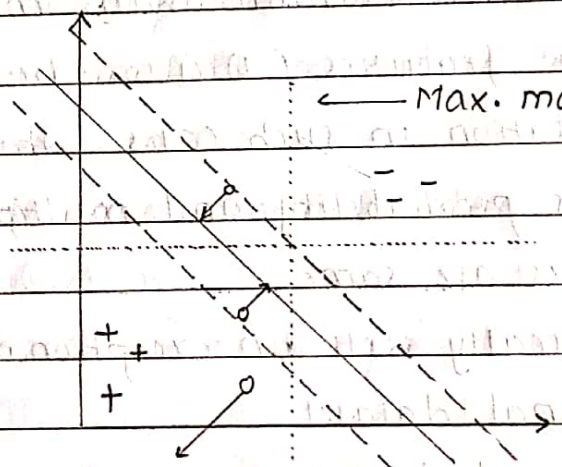
Support Vector Machines (SVMs)



1. Less likely to start overfitting
2. SVM can split these +ve and -ve classes or points in many ways.
3. Logick puts boundary in such a way that all points are correctly classified.

4. SVM nuances further : The distance between boundary and closest points are high as possible.

5. Separation is as high as possible.



w: vector that allows to determine distance between margin and boundary

$\vec{w} \cdot \vec{u} = c$, $\vec{w} \cdot \vec{u} - c \geq 0$

$\vec{w} \cdot \vec{v} = c$ $\vec{w} \cdot \vec{v} + b \geq 0$

$\vec{w} \cdot \vec{u} + b \geq +1$, y: class

$\vec{w} \cdot \vec{v} + b \leq -1$

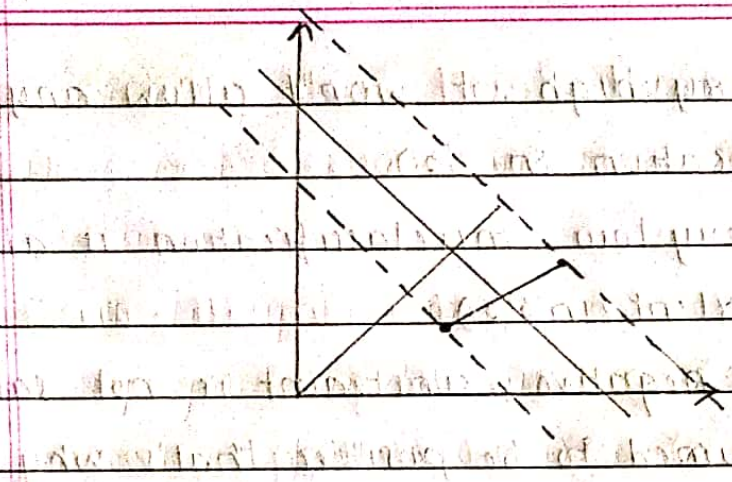
For positive class, $y = 1$

negative class, $y = -1$

So, we can say that, $y(\vec{w} \cdot \vec{u} + b) \geq +1$

$y(\vec{w} \cdot \vec{u} + b) - 1 \geq 0$

Date
Page 31



6. Objective:

i) Maximize width as much as possible

ii) Misclassified points

7. $\overrightarrow{x_p} = (1-b)\frac{\overrightarrow{w}}{\|w\|}$, $\overrightarrow{x_n} = (-1-b)\frac{\overrightarrow{w}}{\|w\|}$

$\overrightarrow{x_p} - \overrightarrow{x_n} = \frac{2 \cdot \overrightarrow{w}}{\|w\|}$ = And this is what needs to be maximized.

Minimize $\frac{1}{2}\|w\|^2$ with $y_i(\vec{w}\vec{v} + b) - 1 = 0$ as constraint.

8. Lagrange's multiplier: Using some multiplier we will minimize entire term.

$$L = \underbrace{\frac{1}{2}\|w_1\|^2}_{1} - \alpha \underbrace{[\Sigma y(\vec{w}\vec{v} + b) - 1]}_{2}$$

α: how much importance to (2) w.r.t. (1)

9. If α is very high, it won't allow any misclassification in (2)
10. If α is very low, misclassification is allowed to some extent in (2)
11. If (2) is negative, constraint is not satisfied.
12. α is assumed to be positive that's why we take -ve sign.
13. We will have minimal value when derivative becomes zero.

$$\frac{\partial L}{\partial w} = \vec{w} - \sum \alpha y \vec{w} = 0$$

$$w = \sum \alpha y \vec{v}$$

$$\frac{\partial L}{\partial b} = \sum \alpha y = 0$$

$$L = \frac{1}{2} \alpha_i y_i \vec{v_i} \cdot \sum \alpha_j y_j \vec{v_j} - \sum_i \alpha_i y_i \cdot \sum \alpha_j y_j \vec{v_j} \vec{v}$$

$$- \sum \alpha_i y_i b + \sum \alpha_i$$

$$= \sum \alpha - \frac{1}{2} \sum \sum_i \alpha_j y_i y_i \boxed{\vec{v_i} \vec{v_j}}$$

↑ In optimal scenario it resolves to this

- Allows for special scenario:
  - Class boundary is linear boundary
  - To build Kernel function transforms to high dimension.

Dot product of every combination,

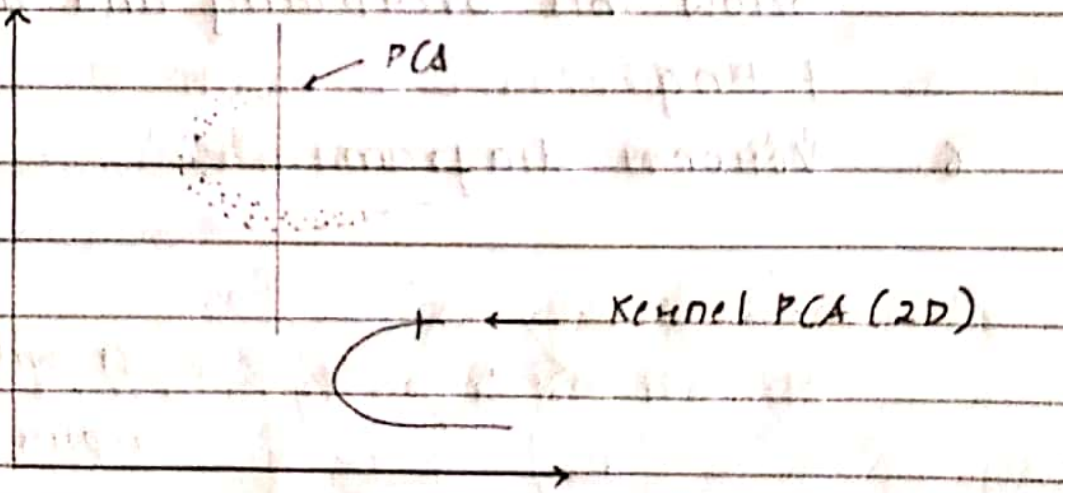
$K \rightarrow (\vec{u} \cdot \vec{v} + 1)$

14. SVMs are the only ones which can tolerate higher dimensions using Kernel.

15. Kernels are also used in PCA, but why?

→ PCA builds linear combinations of features

→ But PCA cannot build non-linear combinations, therefore we use Kernels.



Advantages:

1) Less likely to overfit.

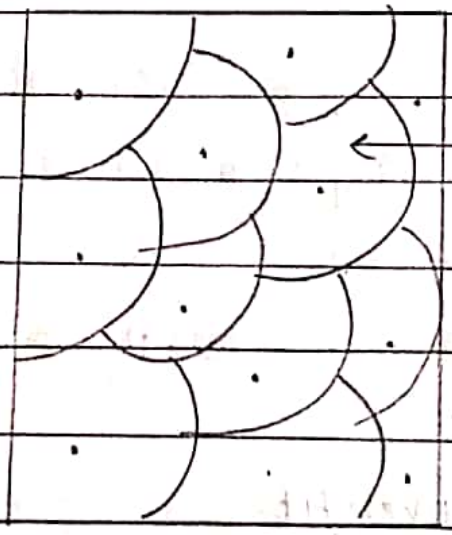
2) High dimensionality: Works well with text

Disadvantages:

1) Slow

Neighbourhood Based Predictions:
KNN (K Nearest Neighbours)

1. Distances of each point from all points.
2. LK (Low K) → Overfitting
3. HK (High K) → Underfitting
4. Good with non-linear data.
5. Value of K = 2, 3, 4, 5 are pretty good classifiers
   Below this overfitting and above that under fitting.
6. Voronoi diagram:



It will have representative points in each section.