Classification → labelling unlabelled data points

KNN - K-nearest neighbour.

① Supervised

② non-parametric algorithm - doesn't take any assumption on underlying data.

③ Algorithm

① Select K

② Calculate euclidean distance

③ Take K nearest neighbour,

④ Count no. of data points in each category

⑤ Assign new data to maximum count.

Pros
① Simple
② effective
③ Robust to the noisy data

Cons
① High Memory Storage
② need to calculate K.

Application

① Banking System

② Calculating credit rating

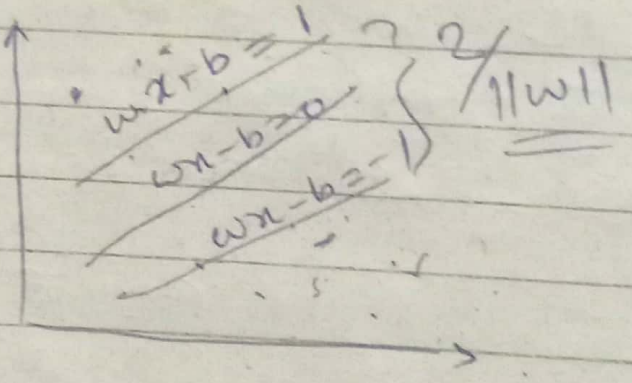
SVM - Support Vector Machine

- It uses critical boundary datapoints to create a hyperplane that differentiates the data points

Hyperplane - Decision Boundary

Space { 2D → 1D, 3D → 2D } hyperplane

Hyperplane - $wXx - b = 0$



Kernel function

non linear data → linear data

low dimension → high dimension

- Linear
- Polynomial
- Sigmoid
- Radial Basis function

Multiclass classification } more than 2 classes

techniques

(1) one vs one (OVO) - heuristic method.

multi → binary

Ex. red, Blue, green & Yellow.

- red vs Blue
- red vs green
- red vs Yellow
- Blue vs green
- blue vs yellow
- green vs Yellow.

$\frac{n(n-1)}{2}$

$\frac{4(4-1)}{2} = \frac{6}{=}$

(2) one vs Rest (OVR) (One vs All)

multi → Binary

red vs [blue, green, yellow]
Blue vs [red, green, yellow]
green vs [red, blue, yellow]
Yellow vs [red, blue, green].

Imbalanced Multi-class problem -

↳ when data points of one class is ~~to~~ much higher than other classes. Such as 100:1

Causes	Technique to handle imbalance
↳ Sampling error ↳ problem domain	(1) Random Resampling ↳ Removing sample from majority (under-Sampling) ↳ Adding sample to minority (over-Sampling) → Good choice (no info. loss)

(2) Tomek links

↳ pair of very close instances but opposite classes

→ Removing it increases the space b/w 2 classes.

→ (under Sampling)

(3) SMOTE (Synthetic Minority oversampling technique)

-) It generate artificial data for minority class

→ The Synthetic Points are added b/w the chosen points and its neighbour.

(4) Class weights

- we can add class weight to add more weightage to the minority class and less weightage to the majority class.

Ensemble Learning

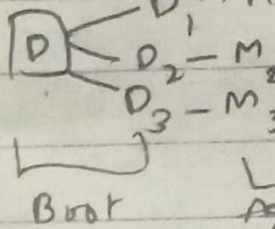
↳ Combine prediction from two or more model.

① Bagging → (Bootstrapping + Aggregation)

— ① create multiple subsets.

② Build multiple classifier

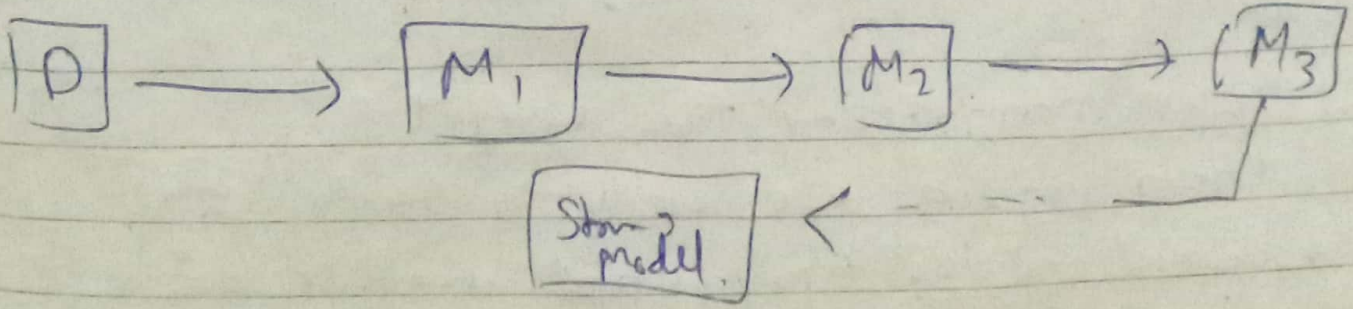
③ combine classifier



② Subagging (Subsample + Bagging)

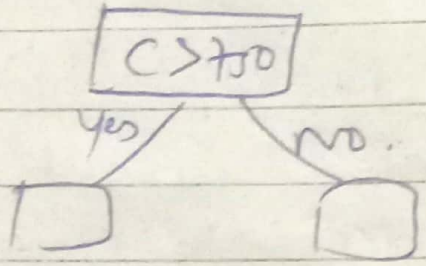
Boosting → Sequential.

→ combine weak classifier into a strong.



(3) Stumping

→ It is one level decision tree



(4) AdaBoost (Adaptive Boosting)

Steps

(1) equal weight apply to all data, $(w=\frac{1}{n})$

(2) A weak classifier is first trained.
and ϵ error is calculated.

(3) performance of classifier $\alpha = \frac{1}{2}\log_e\left(\frac{1-\epsilon}{\epsilon}\right)$

and new weight are updated

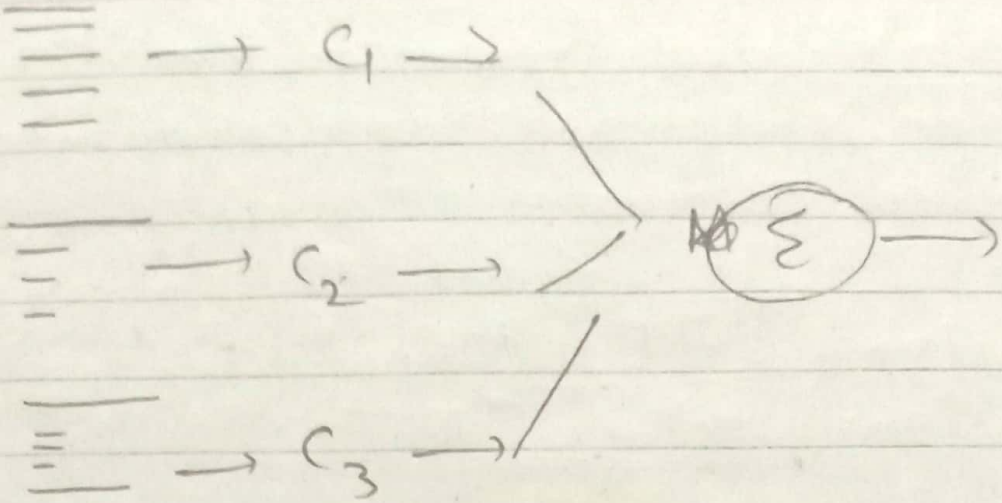
$W_{new} = W_{old} e^{-\alpha}$ (for correct data)

$W_{new} = W_{old} e^{\alpha}$ (for incorrect data)

NOW, we increase the weight for incorrect data.

Now, we normalized the weight.

(4) Then again we apply the same thing a weak classifier until we get error 0, or some user defined value.



(5) Random forest

→ Combine several decision tree to reduce errors and to build a more accurate prediction model.

→ Bagging

Steps ① n no. of random records are taken from k no. of data

② Individual decision tree is constructed for each sample

③ Each DT will generate output

④ And at last voting or averaging.

Confusion Matrix → comparison b/w the predicted classification and actual classification various Performance Measure.

		Actual	
Predicted	class1	TP	FP
	class2	fN	TN

Accuracy = $\frac{TP+TN}{TP+FP+FN+TN}$ X100.

TPR, Recall, Sensitivity $TPR = \frac{TP}{TP+FN}$

FPR $FPR = \frac{FP}{FP+TN}$.

TNR, Specificity selectivity $TNR = \frac{TN}{TN+FP}$

Precision = $= \frac{TP}{TP+FP}$.

F1 Score = $= \frac{2TP}{2TP+FP+FN}$

(2) micro Averaging Precision = $\frac{TP_{total}}{TP_{total} + FP_{total}} = \frac{TP_1 + TP_2 + TP_3}{(TP_1 + TP_2 + TP_3) + (FP_1 + FP_2 + FP_3)}$

(3) MacroAveraging Precision = $\frac{1}{3}$ (Precision1 + Precision2 + Precision3)

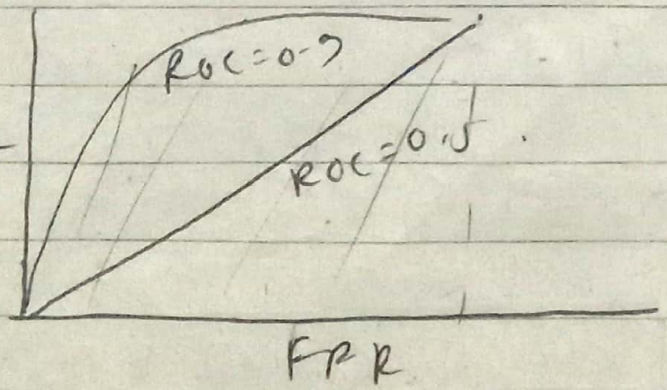
(3) micro Averaging Recall = $\frac{TP_{total}}{TP_{total} + FN_{total}}$

(3) MacroAveraging Recall = $\frac{1}{3}$ ($Recall_1 + Recall_2 + Recall_3$)

(4) Micro average f1 Score: $\frac{TP_1 + TP_2 + TP_3}{(TP_1 + TP_2 + TP_3) + \frac{1}{2}(FP_1 + FP_2 + FP_3 + FN_1 + FN_2 + FN_3)}$

(5) Macro Averaging F1 = $\frac{f_1 + f_1 + f_1}{3}$

ROC → TPR vs FPR.

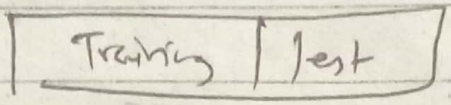


AUC - Area under curve.

- → higher area better classifier
- → low area poor classifier

Cross validation

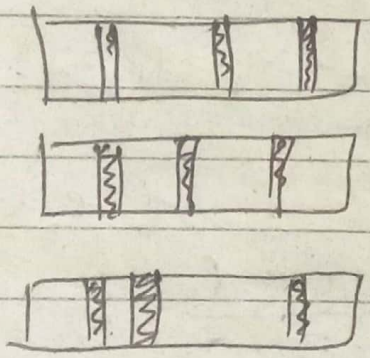
① Holdout method → dividing data into 2 parts



70:30 , 80:20, 90:10

② Repeat holdout method (iteration)

① Random Sampling → (divide testing & training set)



$$E = \frac{1}{k}\sum_{i=1}^{k} E_i$$

③ K fold cross validation

① divide sets into k part with equal size

3 fold

iteration									
①	1	2	3	4	5	6	7	8	9
②	1	2	3	4	5	6	7	8	9
③	1	2	3	4	5	6	7	8	9

test (①) · train (④⑤⑥) · train (①②③) · test (⑦⑧⑨)

Leave P-out cross validation

P no of point out from n. total.

we train (n-p) data and test p. points

You repeat this process for all possible combination