PART - 3 [Classification]

- Unlike regression (continous number), we predict a category here.

- These include,
  - linear models like Logistic Regression, SVM
  - non-linear models like K-NN, Kernel SVM & Random Forests.

Classification Models:

1. Logistic Regression
2. K-Nearest Neighbors (K-NN)
3. Support Vector Machine (SVM)
4. Kernel SVM
5. Naive Bayes
6. Decision Tree Classification
7. Random Forest Classification

① Logistic Regression.

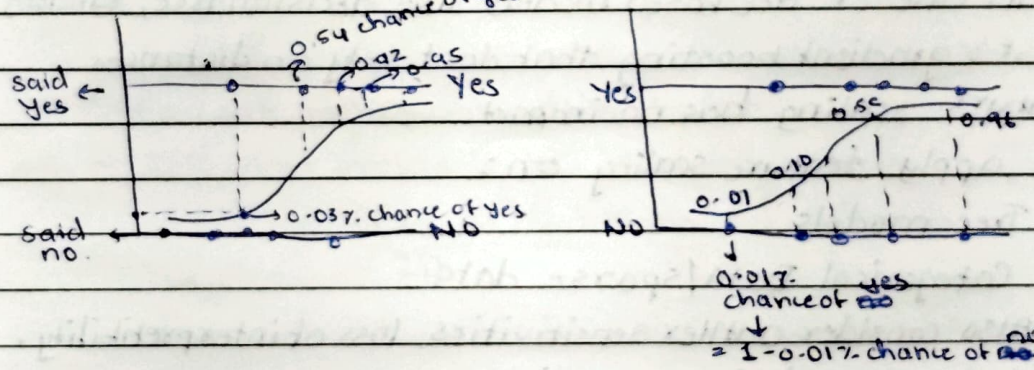
- Predict a categorical dependent variable from a no. of independent variables.
- Need to scale data here.

$$\ln \frac{P}{1-P} = b_0 + b_1 x_1 + \cdots + b_n x_n$$

- a statistical method used for binary classification.
- Predicts probability of binary outcome.
- outputs probabilities mapped to discrete classes.

* Maximum Likelihood Estimation:

- Logistic R. uses MLE to estimate coeff. of model.



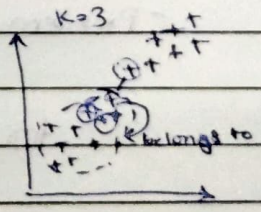
M.L. = $0.03 \times 0.54 \times \cdots \times (1-0.01) \times (1-0.04) \times \cdots$

(multiply all these numbers).

and the curve with the max. likelihood is best curve.

② K-NN.

- Find euclidean distances from all points.
- Select the K nearest ones (min. distance)
- Take the max vote to identify class.



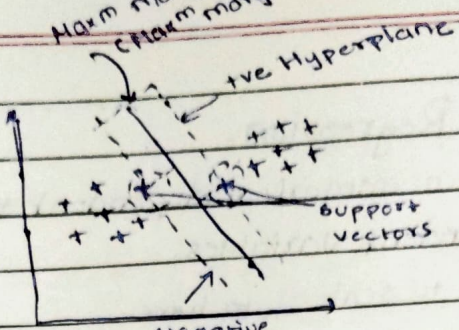
- Need to rescale data in KNN too, or a single feature will weigh too much on output.

DATE
PAGE NO.

(3) SVM.

(Linear).

coz of linearly separable data



Working Principle

- The sum of the 2 distances ~~by~~ between closest pts. i.e support vectors should be maxm.
- ~~It~~. Both support vectors are equidistant from margin.

★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★

Again we apply feature scaling. It only improves the performance. So never bad to apply it.

But in case of Tree based models like decision tree, random forest & gradient boosting that dont rely on distances feature scaling has no impact.

Dont apply feature scaling on:

1. Tree models
2. Categorical Data/sparse data.

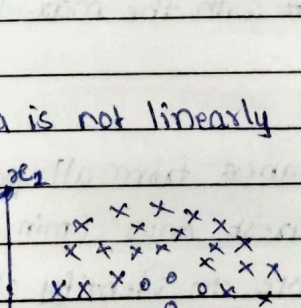
Also consider outlier sensitivities, loss of interpretibility, & algo tuned for original data like naive bayes.

★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★

(4) Kernel SVM

- Problem when data is not linearly separable.

for e.g.

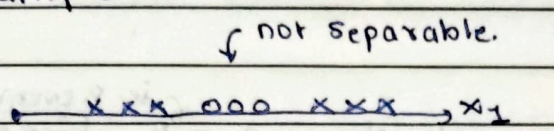


- Way to get this around is to map points in higher dimension & get linearly separable data set, then invoke SVM algo, build a decision

boundary for dataset & project all back to original dimensions.

* Mapping To Higher Dimension..

- 1D example.

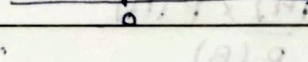


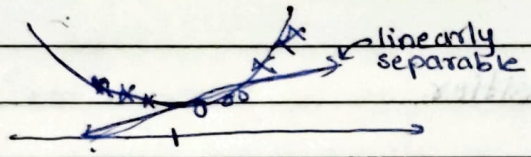
Dot is separator in 1D.
Line in 2D
Hyperplane in 3D.

- Mapping function example

$f = x - 5$

$f = (x-5)^2$





- Same with higher dimensional data.

e.g. - 2D ⟶ project in 3D applying some fn.

$\varphi(x_1, x_2) \Rightarrow (x_1, x_2, z)$ ← new dimension

2D → 3D space

Project it back to 2D

Problem : (1) Computationally ~~Expensive~~ Intensive

going to higher dimension then back to lower dimension.

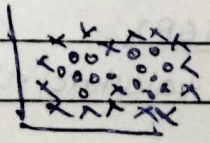
So we use Kernel Trick.

- Gaussian RBF Kernel

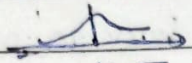
$$K(\vec{x}, \vec{l}^i) = e^{-\frac{\|\vec{x} - \vec{l}^i\|^2}{2\sigma^2}}$$

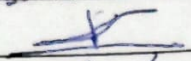
- increase σ, the circumference in 2D increases & vice versa.

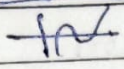
Can be added too. $K(\vec{x}, \vec{l}^1) + K(\vec{x}, \vec{l}^2) \Rightarrow$



Types of Kernel functions

(1) RBF 

(2) Sigmoid. 

(3) Polynomial 

(5) Naive Bayes.

* Bayes Theorem:

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

$P(A|B)$: Probability of event A on B (ie. B event has been occurred, now P(A) needs to happen)

* Naive Bayes classifier.

- It is a probabilistic classifier algo.
- It predicts probability of data point belonging to a class

- Calculation when adding new data point.

STEPS 1: (1) Prior Probability $P(class)$ ← for each class (No. of class / Total datapts)

(2) Marginal likelihood $P(x)$ ← x is feature set.

(3) Likelihood $P(x|class)$

(4) Posterior Probability $P(class|x)$ ← To calculate

Bayes Theorem: $P(class|x) = \frac{P(x|class) \times P(class)}{P(x)}$

where (*) $P(class)$ = Prob Total class members / Total data points.

(*) $P(x)$ = No. of observations around datapoint (radius is chosen) / Total observations.

(*) $P(x|class)$ = No. of similar obs of class around datapt. / Total no. of class observations

STEP 2:

Calculate Probability that feature X belongs to a class by bayes theorem.

STEP 3:
Similarly calculate probabilities for all classes.

STEP 4:
compare all probabilities. The maximum probability is where new data pt belongs by default.

- It works on assumptions (independent assumptions from some underlying data), thus its called Naive.

⑥ Decision Tree
- Looks for optimal splits to maximize pts.
- Same as regression but we predict class here. rather than averages.

⑦ Random Forest
① Pick random K data pts from train set.
② Build DTree on it
③ Build N trees following step ① & ② repeatedly
④ During prediction, predict with each of decision tree & take majority vote.

Model Selections.

① Confusion Matrix & Accuracy.

		Prediction: POS+	Prediction: NEG-
Actual	POS+	TP	FN
Actual	NEG-	FP	TN

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

To remember, Imagine for disease.

Type I error - False +ve — No problem, doctor checks.
Type II error - False -ve — Deadly as doctor won't check.

[Just for purpose of remember, Doesn't actually mean anything]

Evaluating Classification Models Performance:

* Accuracy paradox:
  - ~~More~~ Large data.
    - Less no. of +ve data but - TP < FP
    - More no. of -ve data and - TN >>> FN.
  - According to accuracy metric model will be good, accurate but actually for +ves its performing worse

* Better Metrics:
  * CAP Curve (Cumulative Accuracy Profile).
  * ROC (Receiver Operating Curve)

Which model to use?
- Identify problem → If → linear [Logistic, SVM e.t.c]
  → or → non-linear [KNN, Naive Bayes, Tree, Forest]
- Logistic or Naive Bayes to rank ~~probabl~~ Prediction by probability
- SVM to predict segment/group.
- Tree for clear interpretation.
- Forest for high performance on cost of less interpretation

DATE
PAGE NO.

SUMMARY

(1) Logistic Regression

- Binary classifies with output as probability. belonging to discrete class
- Uses MLE to estimate coeff
- Need to scale data before training model.

(2) K-NN

- Works on euclidean distance with K nearest neighbors
- Need to rescale data

(3) SVM (kernel = 'linear')

- Closest pts are support vectors that are equidistant from Maxm margin hyperplane & sum of distances b/w is maxm.
- Need to rescale data.

(4) Kernel SVM

- For nonlinear data, Map to higher dimension & then project back to original dimension.
- Kernel example = 'RBF'
- Need to rescale data

(5) Naive Bayes

- Probabilistic classifier predicting probability of pt. belonging to particular class
- Rescale not really needed unless theres extreme difference or range in data. (Try Avoid scaling for Bernoulli & Multinomial).

(6) Decision Trees & Random Forest

- reScaling data has little to no impact.