(Day 24/21)

Day-27.

K-Nearest Neighbours.

- classification Technique
- classify records with help of euclidean distance.

find closeness of 2 points

Features.

- All instances correspond to pts. in n-dimensional Euclidean space
- Classification is delayed till new instance arrives.
- Target f^n may be discrete or real valued.
- Classification done by comparing feature vectors of diff. points.

It is instance Based Learning

→ Based on characteristics of record we classify the O/P.

Euclidean distance →

Find closeness of 2 records.

1st response

2nd response

$$\sqrt{(x_1-\mu_1)^2 + (x_2-\mu_2)^2 + \dots + (x_p-\mu_p)^2}$$

or

$$D(X,Y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

record 1

Ex Jay:
Age = 35
Salary = 95k
CD = 3

record 2.

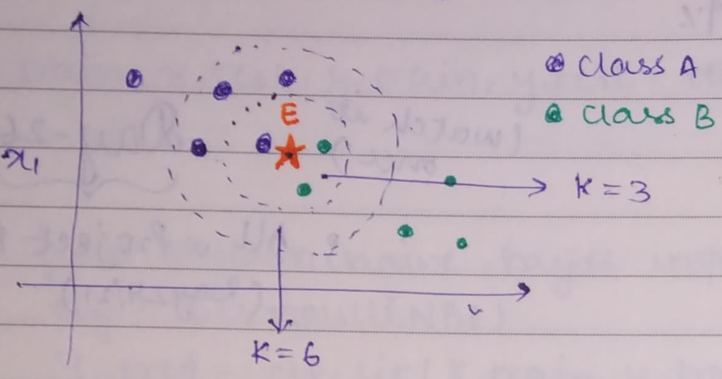
Riya:
Age = 41
Salary = 215k
CD = 2

Ex:

euclidean dis. $= \sqrt{\underbrace{(35-41)^2}_{\text{Age}} + \underbrace{(95000-215000)^2}_{\text{Salary}} - \underbrace{(3-2)^2}_{\text{CD}}}$

Note choosing K: k=5 (default), choose that valle of K which has lowest error rate in validation data.

Knn → find how much target variable is closer to predicted variable.



E(★) needs to be predicted, from which class it belongs,

- for K=3, 3 nearest neighbours are taken & from majority from 3, it is predicted.
- for K=6, 6 neighbours were taken for prediction

. Prediction.

K=3 → class B

K=6 → class A

sadhan.

Ex. If Both are same : 2 class A, 2 Class B, then we need to ↑ the K value. & we need to do iteration.

Strength

- Simple to implement & use.
- easy to explain pred^{n}
- Robust to noisy data by avg. knn

Weakness.

- Need lot of space to store all eg (example)
- Takes more time to classify a new ex. than with a model.

Advantage.

- can be applied to data from any distribution
- Good classification if no. of sample is large enough.

Disadvantage.

- chooking best K maybe difficult
- Need large no. of samples for accuracy.
- can never fix without assuming parametric distribution.

{ DV → categorical
{ IDV → " + contn.

Practical Implementation.

Dataset used → Titanic.

· import all essential lib.

```
df = pd.read_csv('train.csv')
df.columns

le = preprocessing.LabelEncoder()
df['sex'] = le.fit_transform(df['Sex'])

from sklearn import neighbors
    y = df['PClass']
    x = ~~df~~[['Pclass','Passengerid'], axis=1)
        df.drop
```

· DV → categorical

```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.3,
                                                    random_state=0)

knn = neighbors.KNeighboursClassifier(n_neighbors=3)
knn.fit(x_train, y_train).score(x_test, y_test)
```

↳ 85.39% (model accuracy).

```
ypred = knn.predict(xtest)
confusion_matrix(ytest, ypred)
```

		Traing Pclass		
		1	2	3
Test Pclass	1	60	6	4
	2	7	27	15
	3	3	4	141

Total = 267
Correct = 228
Incorr = 39

228/267 = 85.39%.

(Here DV is const. & change k)

DV	IDV	K	Score
Pclass	All.	3	85.39%.
"	"	4	83.89%.
"	"	2	85.39%.

hw. create fn.

K = 1 to 167.
plt.plot ~~hist~~.

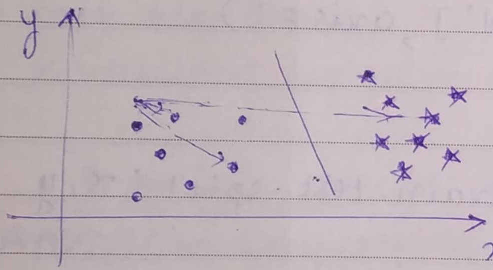
SVM (support Vector Machine).

↳ classify record with help of hyperplane.

SVM classification
- → Linear SVM (HyperPlane)
- → Non-Linear SVM (Kernel Trick)

- SVM is used to classify record for over-dimensional data.
  - Used for both Regression & Classification problems.
  - mostly used in classification problem.

(we plot each data item as pt. in n-dimensional space), wher n is no. of features.



hyperplane is a line which divide the 2 groups.

Rules for Hyperplane

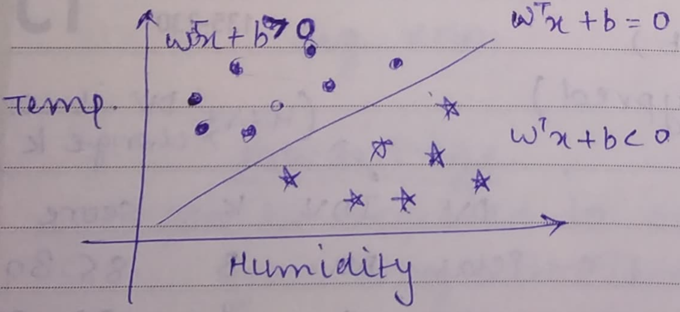
① Hyperplane should divide the 2 groups.

it should be equidistant to both groups.

② whichever pt. is closer to hyperplane that pt. is called support vector.

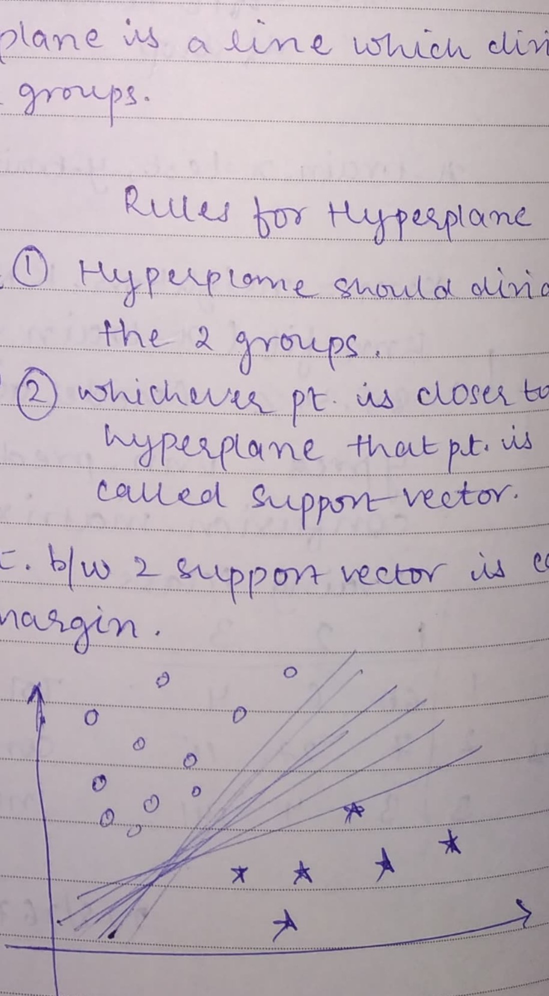
dist. b/w 2 support vector is called margin.

- Linear SVM.



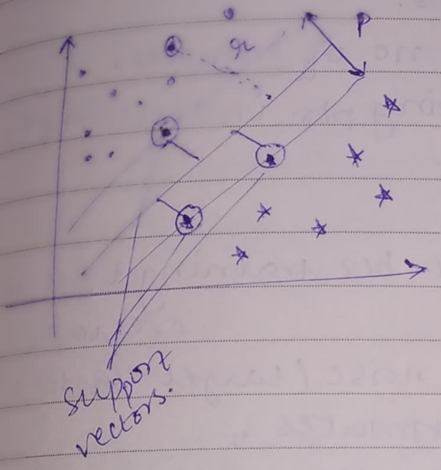
$$f(x) = \text{sign}\,(\underbrace{w^T x + b}_{})$$

eqⁿ of hyperplane from algebra.



n hyperplanes.

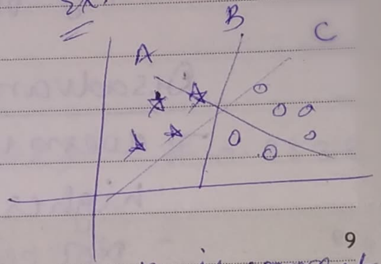
we can draw n-no. of hyperplanes which divide 2 groups.
But AIM is to find such hyperplane that correctly clasify data.



Good Hyperplane-

① } Both rules
② should be followed.

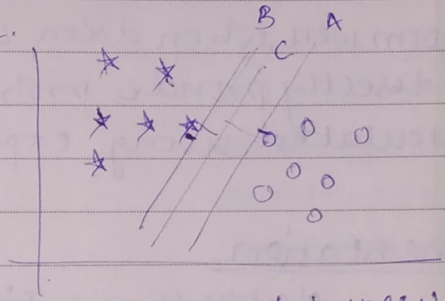
Ex.



9

B is correct 10
hyperplane 11

Ex2.



12

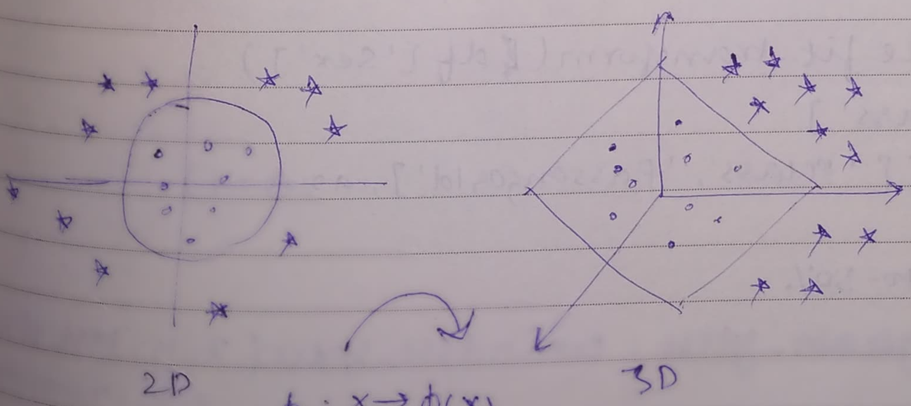
1

2

∴ C is correct hyperplane.
(A/B ko agr lenge to ek to supportvector k pas hoga 3
lekin dusra vector dur hojayega.) 4
Margin difference (they are not equidistant) 5

6

Non-Linear.
we convert 2D to multidimensional. is called Kernel-trick.



$\phi : x \to \phi(x)$

Advantage

- works really well with clear margin of separation
- effective in high dimensional spaces.
- effective where no. of dimension > no. of samples.
- M/ry efficient (uses subset of training pts. in decision f^n).

Disadvantage

- doesn't perform well with large dataset bcz training time is higher.
- Not perform well, when data has noise (target classes overlaps.)
- doesn't directly provide prob. estimates, calculated using expensive 5 fold cross-valid^n.

Python Implementation

dataset → Titanic

import necessary libraries.

```
from sklearn import svm
df = pd.read_csv('train.csv')

df['sex'] = le.fit_transform(df['sex'])
y = df ['Pclass']
x = df.drop(['Pclass', 'PassengerId'], axis=1)
```

Split data into 70-30%.

```
x_train, x_test, ytrain, ytest = train_test_split(x, y, test_size=
                                      0.3, random_
                                      state=0).
```

clf = svm.SVC(gamma=0.01, C=100)

↓ 99.9% this record is going to execute accurately.

↳ complexity factor ie 100 times it perform iteration

clf.fit(xtrain, ytrain)
ypred = clf.predict(x-test)
accuracy-score(ytest, ypred,normalize=True) # 0.8838
confusion matrix(ytest, ypred).

9

Training Pclass 10

	1	2	3
1	61	6	3
2	4	35	10
3	2	6	140

11 12 1 2

Total = 267
correct = 236
Incorrect = 31

} 236/267 = 88.38%

3

HW

	DV	IDV	ACC.
✓	Pclass	All	~~88.38%~~ 89.55
	Survived	"	76.11
	Gender	"	75%
	Embarked	"	74.62%
	Parch	"	82.83%
	Sibsp	"	74.25%

4 5 6

which comb^n is giving high accuracy?

. Create fn.