Anamoly Detection Notes

- Darshan R M

"Learn, Share, and Collaborate"

Week - 20

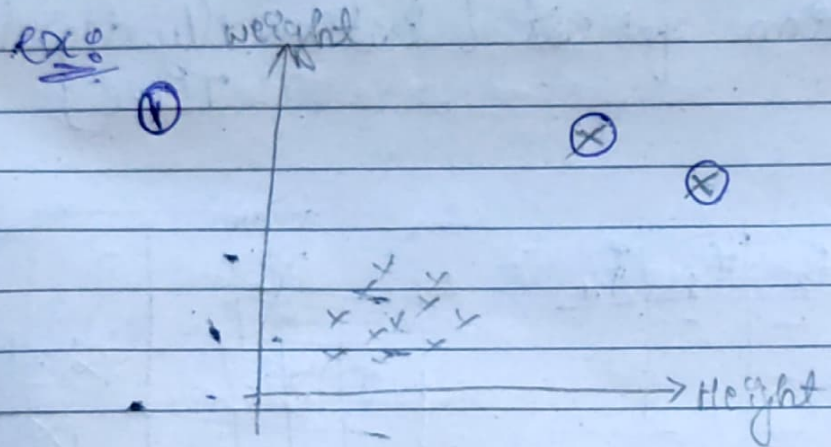
Anomaly detection and Time series

* Anomaly detection

  [To detect outliers] → play a imp role.

  → used when if outlier is the important parameter for solving a problem.

ex:-



(2) if person will have a cancer or not it is a very important outlier

↓

In this type of problem statements where outliers play major role we use "anamoly detection".

(3) IP address if hacker hacking from different location we can detect

④ IPL

over	runs/over
1	15
2	10
3	12
4	100

→ outlier as run > 36

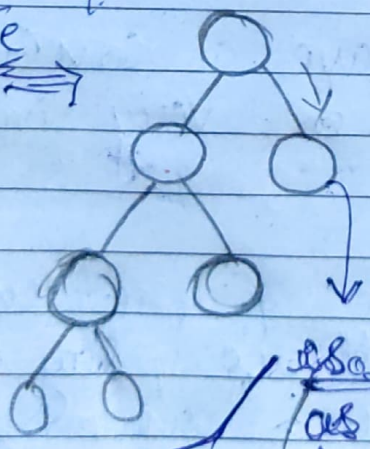
* Types of anomaly detection

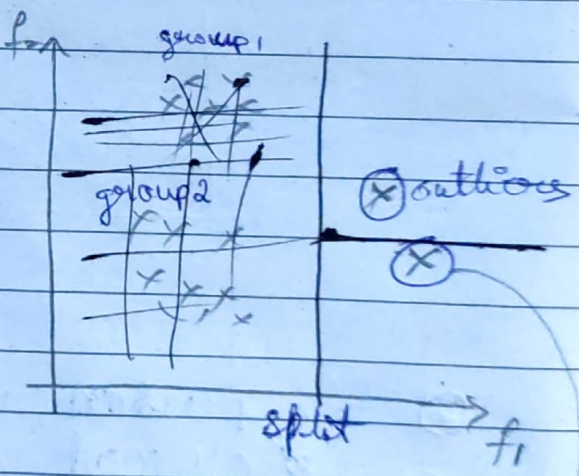
① Isolation forest [Internally decision Tree]

→ Consider,

f_1	f_2	f_3	f_4
-	-	-	-
-	-	-	-
-	-	-	-

DT called as Isolated forest ⇒





→ Create as DT for each point. until we get leaf node.

we make this point completely isolated

isolated as a leaf node.

because can isolate a data point quickly.

how quickly we are able to isolate pt. → outliers

* we Calculate a "Anomaly score"

↓

so that we get to know why we are Considering particular point as a anamaly / isolated point / outlier.

if cross particular threshold then we Consider

*

Maths Formula for Computation of anamaly score for a data point

$$s(x, m) = 2^{-\frac{E[h(x)]}{c(m)}}$$

where,

$m \Rightarrow$ no of data points

$x \Rightarrow$ Data point

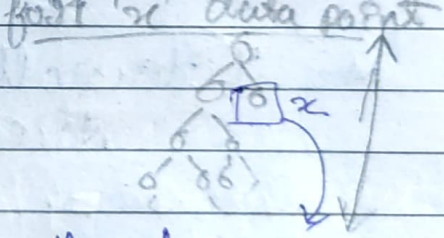
$E[h(x)] \Rightarrow$ Average search depth for x from the isolated tree.

$c(m) \Rightarrow$ Average value of $h(x)$ (or) depth

Here, we are not Considering for 'x' data point as average we Considers for all points

Whats is the average search depth of the 'x' from all different isolation tree.

Computing only for 'x' data point



we Can Consider multiple feature and splot the tree so we will get multiple different isolated trees.

* If we have an outlier then

For outlier,

$E[h(x)] \ll c(m) \Rightarrow s(x,m) \approx 1$

as $\frac{E[h(x)]}{c(m)} \approx 0 \Rightarrow 2^{-0} \approx 1$

Outliers ⇐ anomaly score

For outlier,

Threshold ≥ 0.5 ⇐ we can keep a threshold and then categorize as outlier.

* $E[h(x)] >> c(m) \Rightarrow s(x,m) \approx 0.5$

⇓

Normal points

* Internally, we are creating isolated tree.

* Difference, DT and Isolated tree

⇒ we are creating isolated tree for each and every data points.

* Implementation

① Import data.
* visualize outlier

② . Anamaly detection

> from sklearn.ensemble import IsolationForest

> clf. = IsolationForest (contamination=0.2)

> clf. fit (df)

> prediction = clf. predict (df)

— —

③ plot

> index = np.where (prediction < 0)

> plt.scatter (df.iloc[:,0], df.iloc[:,1])

> plt.scatter (x[index,0], x[index,1], edgecolors = 'red')

— —

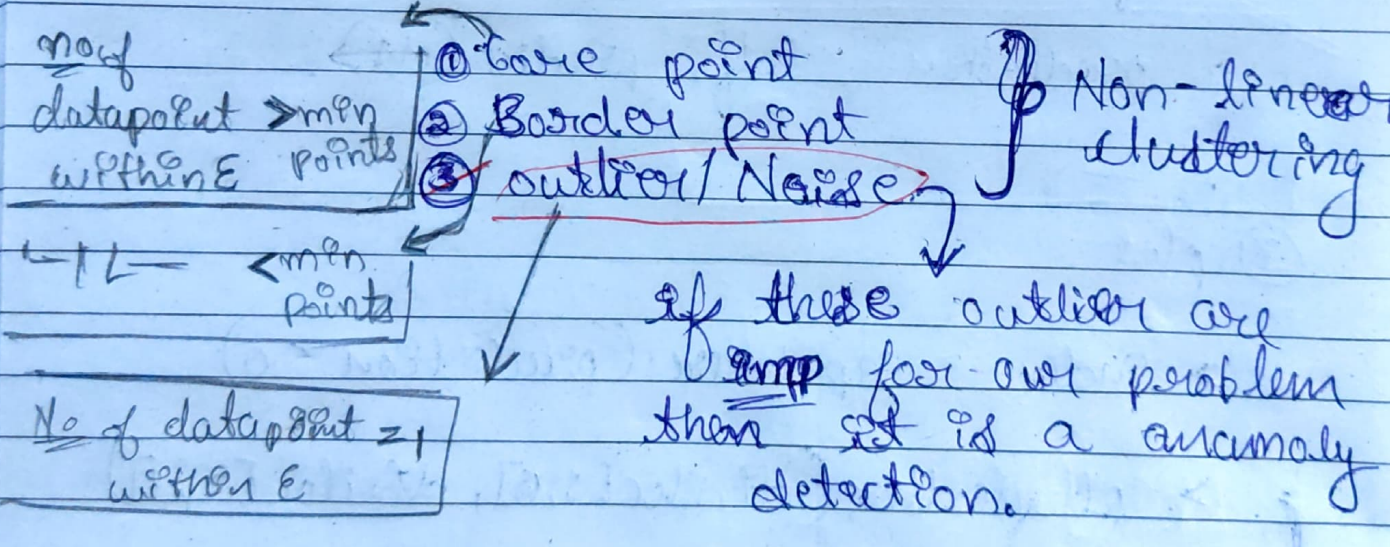
* Anamoly detection using (DBSCAN)

Can do non-linear clustering

we detect a outlier here

If those outlier were important for our problem statement then it's a kind of anamoly detection.

* 3 points

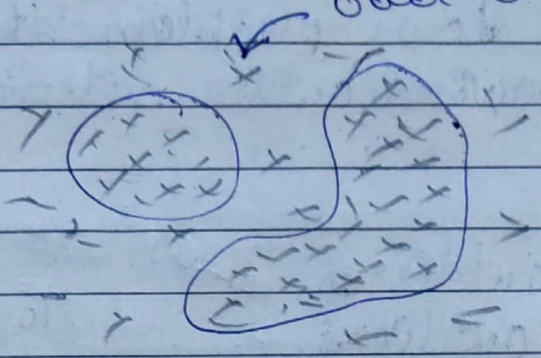


* 2 hyper parameters

① minimum points.
② Epsilon [ε] - radius.

* DBSCAN is robust to outliers.

* ex:



* Implementation

> from sklearn.datasets import make_moons, make_circles.

> from sklearn.cluster import DBSCAN

> x, y = make_circles(n_samples = 750, factor = 0.3, noise = 0.1)

> plt.scatter(x[:, 0], x[:, 1])

> dbscan = DBSCAN(eps = 0.1)

> dbscan.predict(x)

> dbscan.labels_

LOF

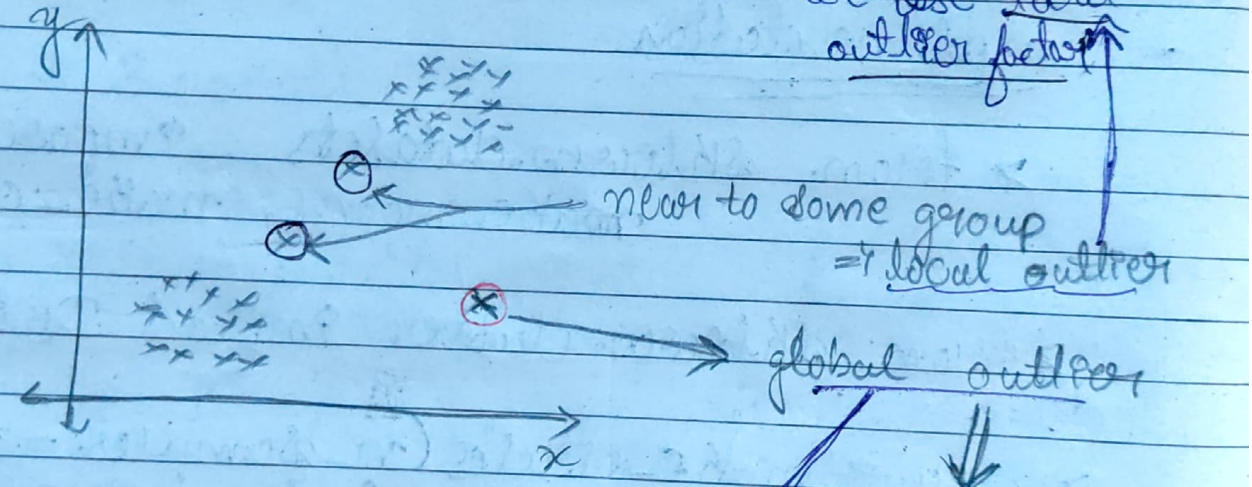
* "Local outlier factor" anamoly detection.
  - Finding outlier that are imp for our problem statement
  - Unsupervised outlier detection using LOF.

* 2 terms:

① local outliers. } LOF score
② Global outliers. }

To find these types of outlier we use "local outlier factor"

⇒



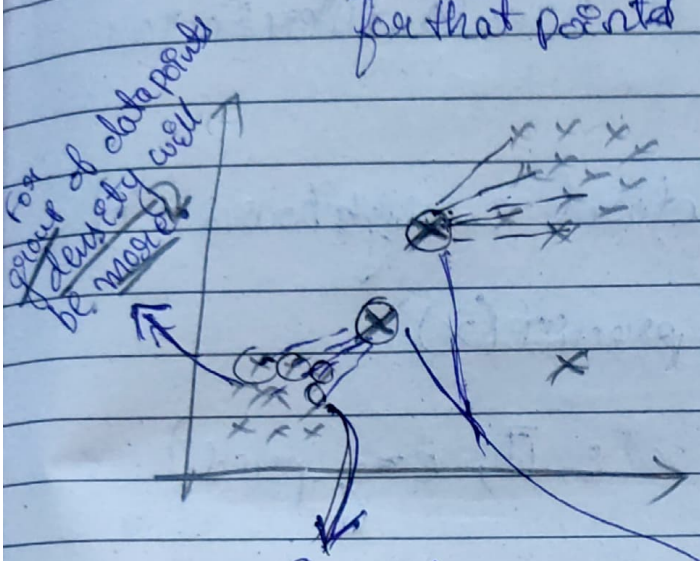
not near to any of the groups.

easy to find this type of outliers using DBSCAN / Isolation Forest.

* Internally it uses K-Nearest neighbours
  ⇓
  "local ~~outliers~~ density"

* Suppose, To check those local outlier are outliers (or) not.

K=5 ← for that points



Calculate average distance

we consider those points and then we observe what is the density w.r.t neighbour points

if density is less, we can consider points as a outlier local

* If the average distance is more then density is less.
  - means points are far.

* Here, it compares local density of samples with local density of its neighbors.
  - Can identify samples that have a lower density than their neighbors.

outliers ⇐

* Implementation

> from sklearn.neighbours import LocalOutlierFactor

> lof = LocalOutlierFactor(n_neighbours = 7)

> ypred = lof.fit_predict(x)

> plt.scatter(x[:,0], x[:,1], c=y_pred)

---

* Time Series

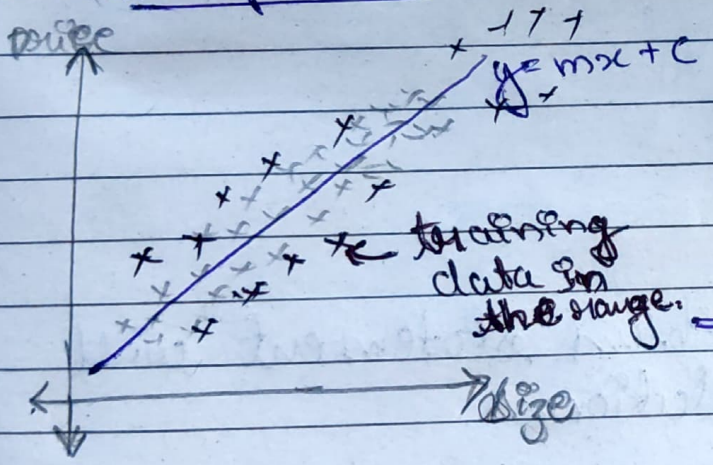
Non time series	Time series
* Doesn't have any time related column.	* At one column will be time
Ex: House price prediction	Ex: Sales data
Size, location, bedroom, Price	Day1 50K Day2 60K Day3 70K Day4 40K Day5 60K
No time representing column	time stamp
regression problem.	

* Time stamp can be hours, min, sec, day, month

Q) Can we solve time-series problem with linear regression

⇒ Yes, but not recommended. because of more cons.

* Interpolation ⇒ To find out the value within the range.



⇒ Supervised learning [regression]

⇒ test data will be with range.. as of training data range.

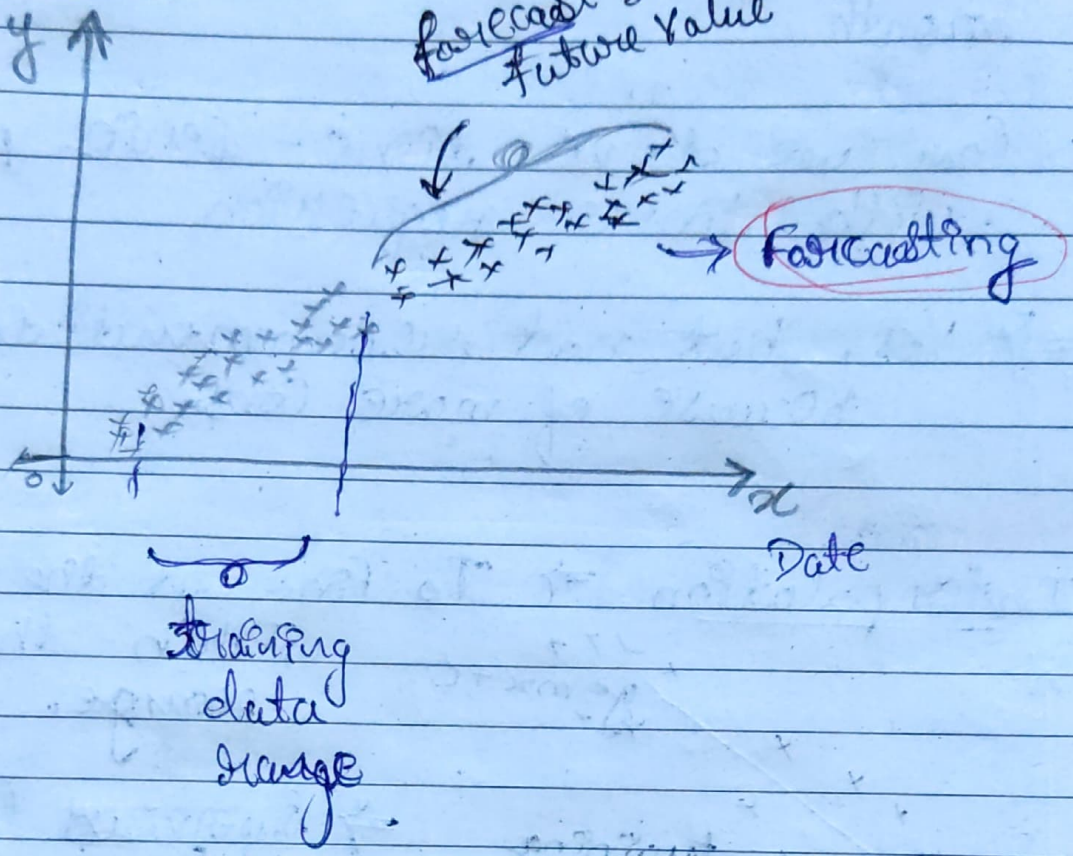
* Extrapolation ⇒ To find out the value out the range.

if training data range is 0-100

if test data comes ⇒ 60, 70, 40, 20, 200, 1000,

lead to wrong prediction.

Extrapolation



* Time series problem statement will be a extrapolation.

⇒ Based on previous history forecast the future values

* If we use linear regression for the time series it may lead to the wrong prediction.

*Cons of regression on Time series data

① Because of extrapolation.
② Because of outlier.
③ There should be a linear relationship.
↓
but in time series data will not be a linear.
⇒ Data will be complicated.

④ Non-time series data → no effect of → previous time

⑤ but time series data → will have effect of → previous time

⇒ Non time series data

size	location	price
1000	Delhi	55L
2000	Mumbai	1cr

Independent of previous data

⇒ Time series data

	sales
Day1	50K
Day2	60K

dependent on previous data.

Time dependency

* Example of time series

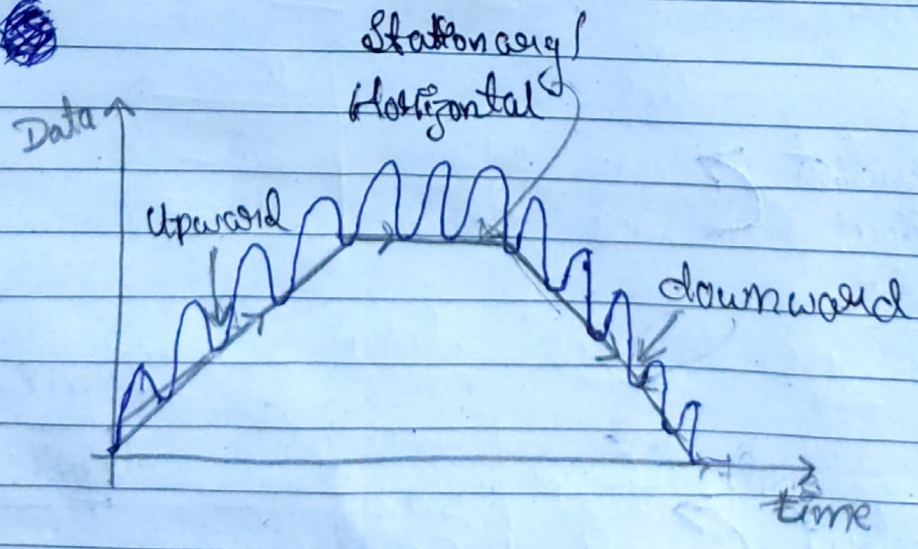
1. Economics forCasting. ⇒ GDP, interest,
2. Finance ⇒ Sales, bond price
3. weather Forcasting ⇒ weather, pattern
4. medical. ⇒ based on previous hist future prediction.

* Time series. —Can be applied to→ where time dependencies is present.

* Time series components

1. Trend.
2. Season.
3. Cycle.
4. Noise.

① Trend



Trend:

- ① Upward.
- ② Horizontal / Stationary.
- ③ Downward.

② Season — based on daily, yearly, monthly, hourly.

→ it is a frequent repetation

ex:- ① Sales of ice cream is summer,

② Traffic at 5pm in my area.

③ Tourist in goa at the end of year. [Dec/Jan]

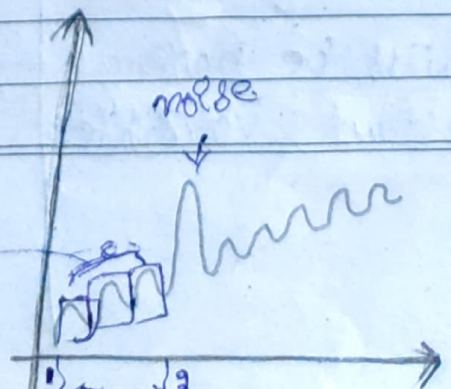
③ Cycle

→ (Cyclic pattern)

→ time series behaviour over the long time.

Cycle ⇒ Season + noise
[fluctuation]

Ex:- Stock price



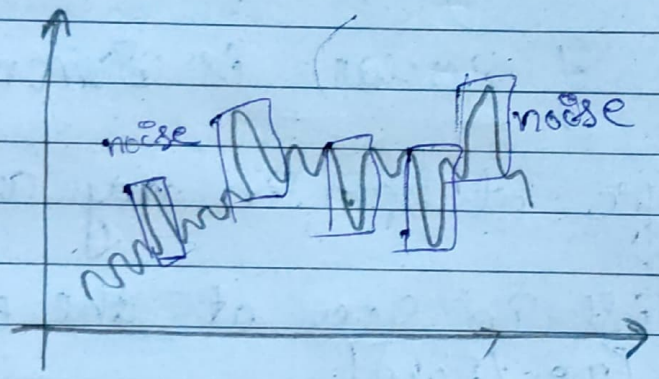
④ Noise

↳ Some uncertainty / randomness in data because of unpredictable reason.

Ex: pandmic, war, news, ---

↓ ↓ ↓

Corona palestan/ Adani Apan



*

trend season noise

$Y_t = T \times S \times N$ ⇒ multiplicative TS

$Y_t = T + S + N$ ⇒ additive TS

Additive	Multiplicative
* It will be linear over time.	* Non-linear
* It will be having constant variable.	* Non-Constant Variable.

* Moving average

Types

① SMA [Simple moving average]
② CMA [Cummulative moving average]
③ EMA [Exponential moving average]
(or) EWMA

a) why we calculate Moving average?
↳ Smoothing of time series data.

① Simple moving average:

~~Average~~ :

* data = [10, 12, 15, 13, 11]

⇒ average = $\frac{\text{sum of all values}}{\text{No of values}}$

$= \frac{10+12+15+13+11}{5}$

average = 12.2

moving ⇒ move over the time axis in specific window time.

① window size.
② average calculating.

Consider,

time	sales	
D1	10	1
D2	12	2
D3	15	3
D4	13	4
D5	14	5
D6	16	
D7	17	

window size = 3

1^{st} avg $= \frac{D1 + D2 + D3}{3} = \frac{10 + 12 + 15}{3}$

$= 12.33$

2^{nd} avg $= 13.33$

$= 14$

$= 14.33$

5^{th} avg $= 15.66$

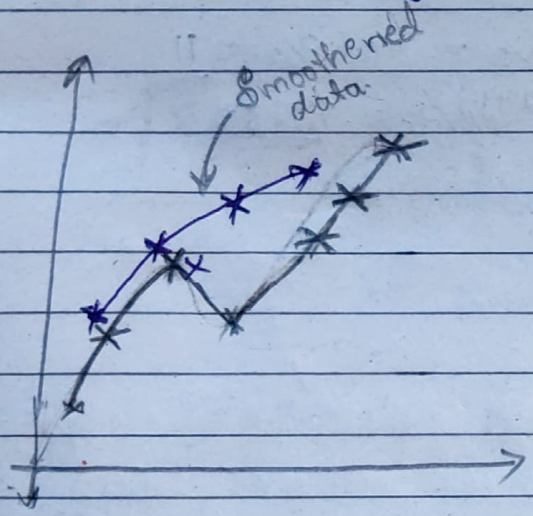
Calculating due to smoothing of data.

⇓

To remove all effect from the data we can perform smoothing with the help of moving average.

Advantage of smoothing

1. Pattern recognition from the data.
2. You can analysis the trend of the data.
3. You can reduce the effect of outlier.
4. Enhancing the visualization of data.



* Window size effect the smoothing.

If window size is (less) → then it will be more specific to point. (not good smoothing)

If window size is (more) → correct smoothing

② Cumulative moving average: [CMA]

↳ Find out average of all the data point upto given time stamp.

Consider

D_1	10
D_2	12
D_3	15
D_4	19
D_5	16
D_6	17
D_7	1

$D_1 = 10$

$D_1 + D_2/2 = 10 + 12/2 = 11$

$10 + 12 + 15/3 = 12.33$

* used for long time period.
* give exponential trend.

* Here, also we do smoothing of data

③ Exponential moving average [EMA]:

↳ we give more weight to recent datapoint/timestamp.

Consider, Dataset

weightage

D1 → dess
D2
D3
D4
D5
D6 → movie

$$V_t = \beta V_{t-1} + (1-\beta)\theta_t$$

V_t = EMA at time t,

β → $0 < \beta < 1$ → 0.9 (Value may be)

amount of smoothing

V_{t-1} ⇒ EMA at previous time stamp.

θ_t ⇒ Data at time stamp t.

Consider,

V_0 → D1	25
V_1 → D2	13.
V_2 → D3	17
D4	31
D5.	43

$N_0 = 0$ / $V_0 = 25$

$V_1 = \beta \times V_0 + (1-\beta)\theta$

$= 0.9 \times 0 + (1-0.9)\,13$

$= 0 + 0.1 \times 13 = 1.3$

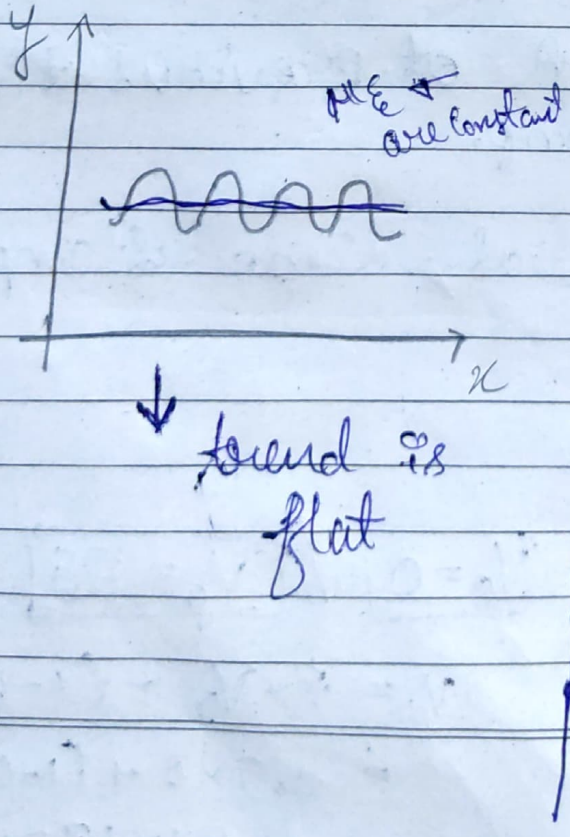
~~* Time series~~

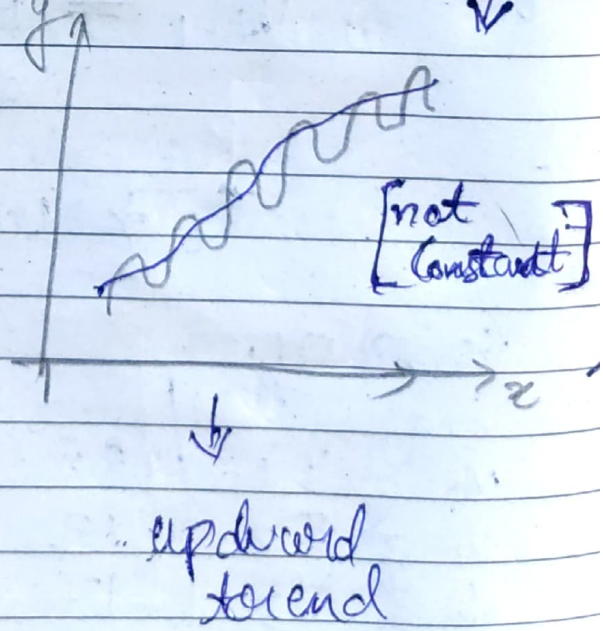
* Stationary and Non-stationary time series

* Non-stationary time series ⇒ mean, variance will not be constant.

* Stationary time series ⇒ mean, variance are constant.

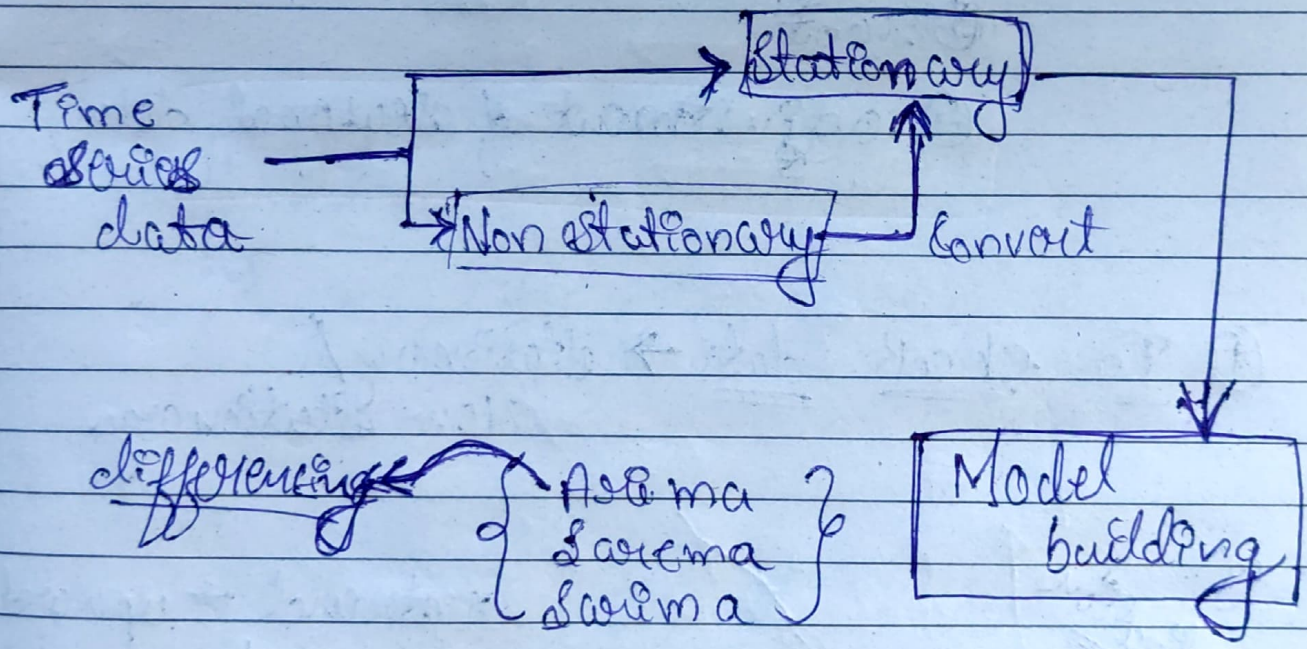
* Constant ⇒ values are not varying about time axis.





Machine learning

1. Data ingestion.
2. EDA.
3. Feature engineering / Preprocessing.
4. Model building.
5. Model evaluation.



* For checking our Time series data stationary / Non-stationary time series

1. Visualization.
2. Stats based test.

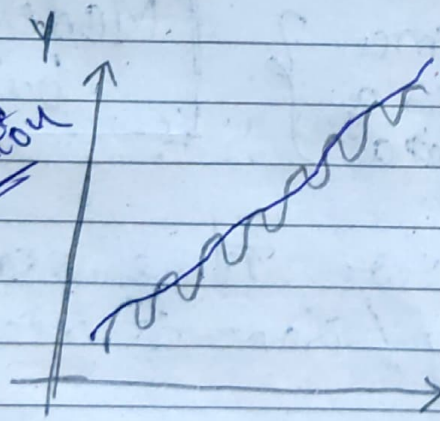
* If Non-stationary Time series data is present then to convert it into stationary Time series data.

⇓

① Differencing.
② log
③ root
④ adjustment of seasonal data.

① To check data → stationary / Non-stationary.

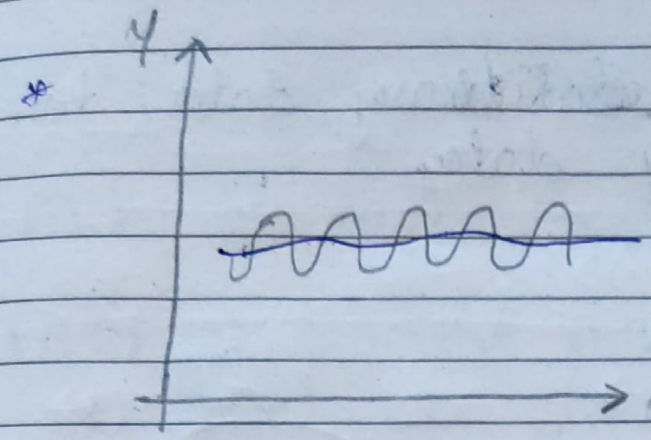
(i) Visualization



* trend ⇒ upward
+
* moving average → increasing over time

⇓

Non-stationary



* Trend ⇒ flat
+
* moving average ⇒ Constant over time

⇓

Stationary

(ii) Stats test

↳ we use ADF test.
[Augmented dickey Fuller Test]

① Statss test.
② P-value → observing P-value we can tell Is data is Stationary / Not
③ Critical value

* h_0 ⇒ data is Non-stationary.
h_1 ⇒ data is stationary.

Stationary ⇐ $P \leq 0.05$ → reject (h_0) null hypothesis

$P > 0.05$ → fail to reject null hypothesis

Non-stationary

② Convert Non-stationary data to stationary data.

① Differencing

Consider,

		First differencing	Second differencing
D_1	5	NA	
D_2	10	5	
D_3	6	-4	
D_4	8	2	
D_5	15	7	
D_6	7	8	

If Non-stationary

↓

test TS data

↓

if stationary / Non-stationary

* difference the data until we get a stationary data.

* ACF, PACF, Auto regression

ACF → Auto Correlation Function

PACF → Partial Auto Correlation function

In ACF ⇒ AUTO + CORRELATION

AUTO → Correlation within same feature for particular time interval.

CORRELATION → relationship b/w 2 features

∴ Types / methods
1. Pearson.
2. Spearman rank.
3. Kendall.

$$\rho_{(x,y)} = \frac{Cov(x,y)}{\sigma_x \cdot \sigma_y}$$

$$\rho_{(R(x),R(y))} = \frac{Cov[R(x),R(y)]}{\sigma_{R(x)}\ \sigma_{R(y)}}$$

→ ACF measure the correlation between time series and its lag value.

Corr (Y_t, Y_{t-1})

Corr (Y_t, Y_{t-2})

	time series	1st lag	2nd lag
D1	10	N/A	N/A
D2	25	$D_1 \rightarrow 10$	N/A
D3	14	$D_2 \rightarrow 25$	$D_1 \rightarrow 10$
D4	16	$D_3 \rightarrow 14$	$D_2 \rightarrow 25$
D5	20	$D_4 \rightarrow 16$	$D_3 \rightarrow 14$
D6	32	$D_5 \rightarrow 20$	$D_4 \rightarrow 16$
	Y_t	Y_{t-1}	Y_{t-2}

relation

Auto Correlation

$\Rightarrow Corr(Y_t, Y_{t-1})$
$\Rightarrow Corr(Y_t, Y_{t-1})$
$\Rightarrow Corr(Y_t, Y_{t-1})$

* **PACF** [Partial Auto Correlation Function]

direct

Ex: $t-2$ $\longleftrightarrow$ t

(i) $t-1$

not Consider

Remove $t-1$ from finding Correlation

(ii) Monday Tuesday Wednesday

Sunny → ~~Storm~~ → Rain

if we want to find relationship b/w monday and wednesday
⇒ In PACF we ignore/ donot consider Tuesday
bc we assume that there will be no storm b/w monday and wednesday.

* PACF ⇒ we remove the intermidiate effect.

---

* Auto regression

Auto + regression

Auto ↙ regression itself within the variable.

regression ⇓ relationship

x ↔ y

simple linear regression

$h_\theta(x) = \theta_0 + \theta_1 x$

x → independent
y → dependent

* formula

$$Y_t = \psi Y_{t-1} + \psi_2 Y_{t-2} + \cdots + \psi_n Y_{t-n} + C$$

where,

Y_t ⇒ Value at current timestamp

ψ ⇒ Coefficient term

C ⇒ Constant

ε ⇒ error

epsilon

* To know until which term we have to consider we know in ARIMA.

* ARIMA (p, d, q)

→ Auto regression, integrated, moving average — 3 parts

Machine learning

Data → Model

linear regression

SVM

KNN

Time series Data → model

ARIMA
SARIMA, SARIMAX
[Deep learning] DL: RNN, Attention, Transformer.

ARIMA

AR
[Auto regression]

P

value can be: [0, 1, 2 → n]

lag value

→ PACF
[Correlogram graph]

I
[Integration]

d

[0, 1, 2 — n]

lag value

→ differencing
[stationary & non stationary]

MA
[Moving average model]

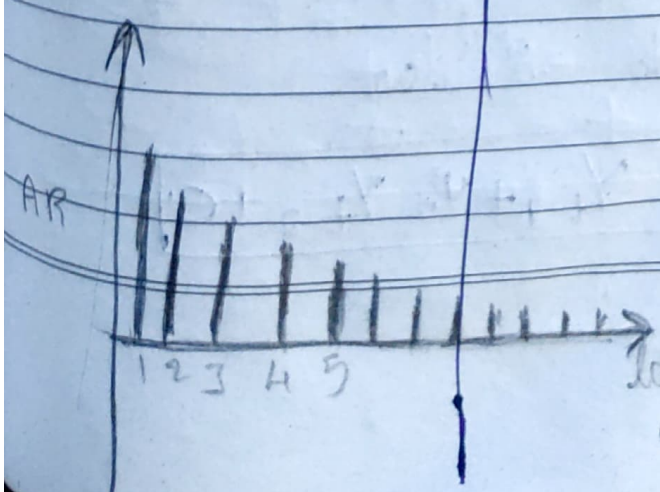
we analyze moving average & calculate error from current timestamp to previous timestamp

q

[0, 1, 2 — n]

lag value

→ ACF
[Correlogram]



$$Y_t = \psi_1 Y_{t-1} + \psi_2 Y_{t-2} + \cdots + \psi_n Y_{t-n} + C$$

(*)

* AR(1) ⇒ upto 1 lag

Current value ← previous

$$Y_t = \psi_1 Y_{t-1} + C$$

lag = 1
AR = 1
P = 1

Consider,

TS Data	Y_t	Y_{t-1} lag 1 TS
D_1	10	NAN
D_2	20	$D_1 = 10$
D_3	30	$D_2 = 20$
D_4	40	$D_3 = 30$
D_5	50	$D_4 = 40$

⇒

Previous Error	
Y_t	
20	10
30	20
40	30
50	40
22	45
22	2

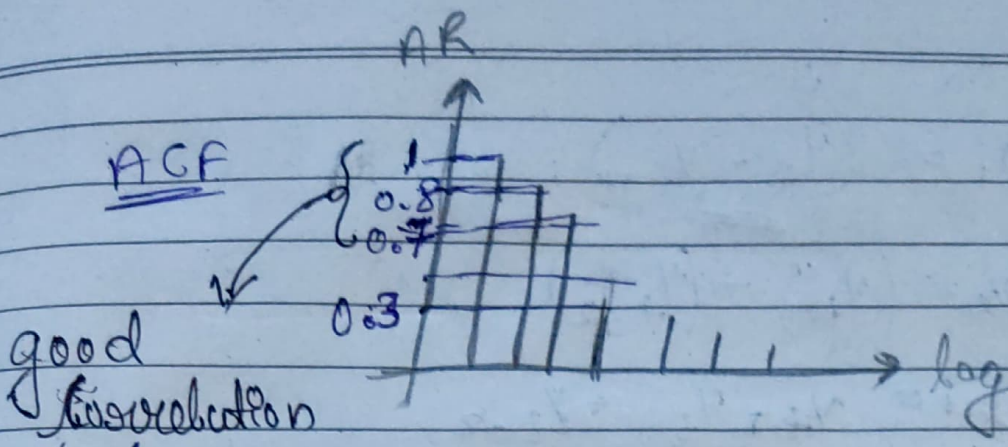
Future Forecasting

$$Y_t = \psi_1 Y_{t-1} + C$$

Y_t Y_{t-1}

* AR(2) ⇒ upto 2nd lag

$$Y_t = \psi_1 Y_{t-1} + \psi_2 Y_{t-2} + C$$



↳ so we take AR(3)

Here, usually we don't take (ACF)

because we don't consider intermediate effect also"

D1	~~16~~		$(Y_t)-(Y_{t-2})$
D2	20		
D3	~~30~~	40-20	
D4	40	20	⇒ we don't Consider the intermediate effect.
D5	~~50~~	60-40	
D6	60	20	

---

* I ⇒ Integration ⇒ difference

Count	D=0	D=1
~~Y_{t-4}~~	D_1	
Y_{t-3}	D_2	D_2-D_1
Y_{t-2}	D_3	D_3-D_2
Y_{t-1}	D_4	D_4-D_3
Y_t	D_5	D_5-D_4

$D_0 = Y_t$

$D_1 = (Y_t - Y_{t-1})$

$D_2 = (Y_{t-1}) - (Y_{t-2})$

$D = Y_t - Y_{t-1} \div Y_{t-2}$

Moving average

↳ model build with the error.

$$Y_t = \varepsilon_{t-1} \Psi + c$$

(error → Coefficient term; c → Constant)

Moving average model

$$Y_t = \varepsilon_{t-1} \Psi_{t-1} + \varepsilon_{t-2} \Psi_{t-2} + \dots + \varepsilon_{t-n} \Psi_{t-n} + c$$

general eqn

built on error term

ARIMA

$Y_t = (Y_{t-1} \Psi_{t-1} + Y_{t-2} \Psi_{t-2} + \dots + Y_{t-n} \Psi_{t-n} + c)$

$Y_t = (Y_{t-1} \Psi_{t-1} + Y_{t-2} \Psi_{t-2} + \dots + Y_{t-n} \Psi_{t-n} + \epsilon) +$
$(Y_{t-1} - Y_t - Y_{t-2} = - - Y_{tn}) +$
$(\varepsilon_1 \Psi_{t-1} + \varepsilon_2 \Psi_{t-2} + - - + \varepsilon_n \Psi_{tn} + c)$

this arima eqn is used to forecast the fiture values.

if

AR(1) , p = 1

I(2) , d = 2 ⇒

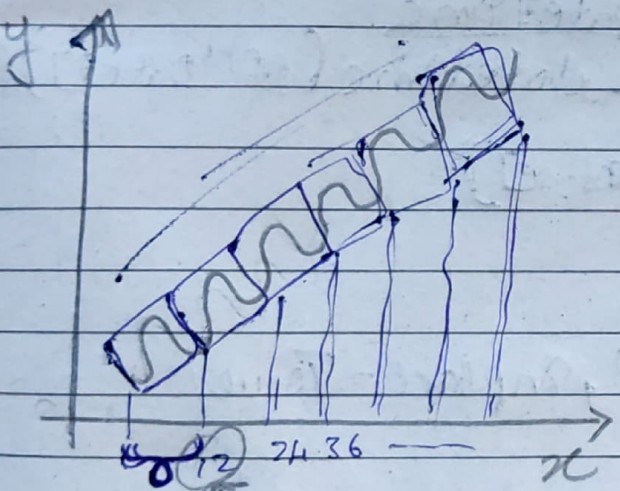
MA(1) , q = 1 ⇒

$$Y_t = (\Psi_{t-1} \Psi_{t-1} + c) + (Y_t - Y_{t-1} - Y_{t-2}) + (\varepsilon_{t-1} \Psi_{t-1} + c)$$

⊛ Variants of ARIMA:

SARIMA

↳ Seasonal



for

pdq ⇒ Full

(PDQ)s ⇒ for seasonal

(p,d,q)(P,D,Q)s 12 month

⇒ (1,1,2)(1,0,2)12

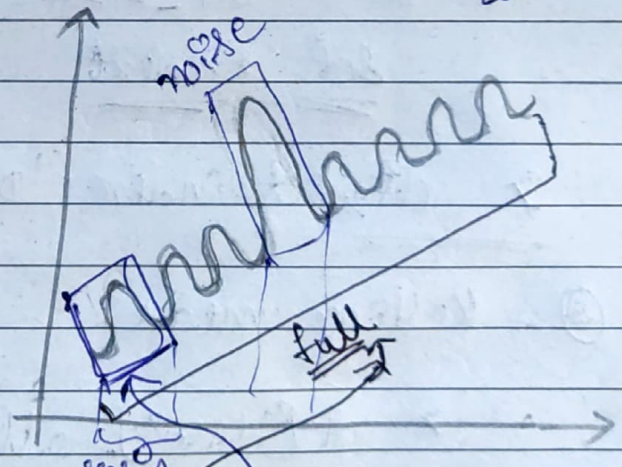
AR I MA / AR I MA

SARIMX

↳ exogenous

(outlier)

[extra effect]



season

(p,d,q) (P,D,Q)

⇒ (0,1,2)(1,1,1)

* Implementation

steps

1. Data ingestion.
2. EDA.
3. processing of the data.
4. Model building.
5. Model evaluation.

(1) * Convert object to Datetime

> df['Date'] = pd.to_datetime(df['Date'])

> df = df[['Date', 'Close']]

Set Index

> df.set_index('Date', inplace = True)

(2) Rolling mean

> df['close'].rolling(12)

> df['close'].rolling(window = 12).mean()

(3) To convert stationary to Non-stationary

↓

differencing

> dfc = df['close']

> dfc = dfc.diff()

> dfc.dropna()

⑤ ——

> import statsmodel.api as sm
> from statsmodel.tsa.arima.model
import ARIMA
> from sklearn.metrics import
mean_squared_error.

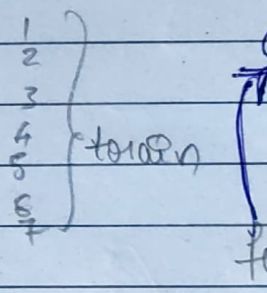
——

> model = ARIMA(history, order(1,1,1))

> model_fit = model.fit()

> model_fit.summary()

> model.forecast()



predicts next value