CS 3352 - Foundation of Data Science

Unit I - Introduction

Data science: benefits and uses - facets of data
Data science process: overview - Defining Research Goals
Retrieving Data - Data preparation - Exploratory Data
Analysis - Build the model - Presenting findings and
building applications - Data mining - Data warehousing
Basic statistical descriptions of data.

Big data:
It is a blanket term for any collection of data sets so large or complex that it becomes difficult to process them using traditional data management techniques such as RDBMS

Data science:
→ It involves using methods to analyze massive amounts of data and extract the knowledge it contains.
→ Relationship between big data and data science is like the relationship between crude oil & oil refinery
→ Data science & big data evolved from statistics and traditional data management

The characteristics of big data:
i) Volume - how much data?
ii) Variety - how diverse are different types of data?
iii) Velocity - At what speed is new data generated?

Additional characteristics:
iv) Veracity (or) variability - how accurate?
v) Value - usefulness of Data.
→ It often referred to as three Vs.

Benefits and uses of Data Science & big data.

⇒ Data Science & big data are used everywhere in both Commercial & non Commercial purpose.

⇒ Commercial companies use data Science and big data to understand in deep about their customers, process, staff and products.

⇒ To offer customers a better user experience & cross-sell-upsell products.

Eg: Google Adsense. Which collects data from Internet users and recommends ads based on it history search and recommends ads based on it history search maxpoint eg of real time personalized advertising.

People Analytics. HR professionals use people analytics data mining to screen candidates, monitor the mood of employees and study informal networks among coworkers

Financial Institutions. Use data Science & big data to predict stock markets, risks of lending money & learn to attract new clients.

Government Organizations rely on data Science to discover valuable information.

eg: Data.gov. It is the home of the US Govt open data.

Non government organizations use it saves their funds. Used to increase the effectiveness of their fund raising efforts

Online learning Platforms.

Edx, coursera, Udacity, Udemy & edx, moocs Massive Open Online courses.

Facets of data: Different types of data (or) main categories of data

1) Structured
2) Unstructured
3) Natural language
4) Machine Generated
5) Graph based
6) Audio, Video and Image
7) Streaming

1. Structured data:
Structured data is data that depends on a data model & resides in a fixed field within a record. It is easy to store & manage data in tables within data base or Excel file.
eg: An excel table.

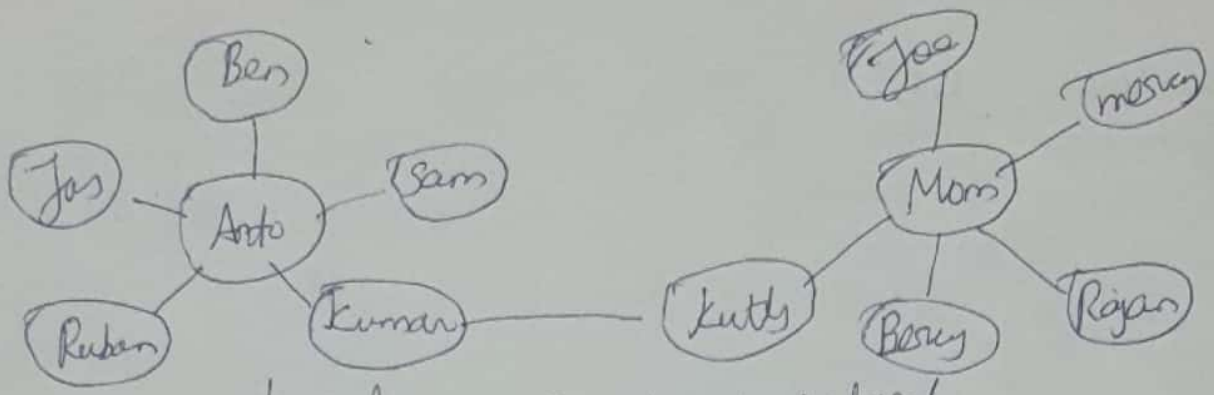
S.No	Register No	Student name	mobile No	email Id.

2. unstructured data
It is not easy to fit into a data model because the content is context specific or varying.
eg: Email.

3. Natural Language:
It is a special type of unstructured data
It is difficult to process because it requires knowledge of specific data science techniques and linguistics
eg: Email, word documents.
Natural Language processing techniques are topic recognition, summarization, text completion, sentiment Analysis

4. machine generated Data:
It is information that is automatically created by a computer, process, application or other machine without human intervention
eg: Web server logs, call detail records, network event logs.
CSIPERF: TxCommit=313236 2014-11-28 11:36:13
Info CSI 000000153 Creating NT transaction
Analysis of machine data relies on highly scalable tools due to its high volume & speed.

5. Graph based (or) Network data.
It data that focuses on relationships between objects. It uses graph structure with nodes & edges to represent objects and their relationships.
eg: friends in social networks.



friends on a social network.

⑥ Audio, Image & Video:
multimedia data in the form of audio, video, images become an integral part of everyday life. Object recognition: challenging for computers
Deepmind: developed an algorithm which is
- capable of learning how to play video games
- able to interpret everything on the video screen via deep learning.

⑦ Streaming data:
It takes almost any of the previous forms but it means the data flows into the system when an event happens instead of being loaded into a data store in a batch.
Eg: whats trending on twitter, live sporting or music events.

1.3 The Data Science Process:
The data science process consists of 6 steps they are
① Setting the research goal.
② Retrieving data
③ Data preparation
④ Data Exploration
⑤ Data modeling
⑥ presentation and automation.

1. Setting the research goal
The purpose of this step is making sure all the stakeholders understand the what how and why of the project.

2. Retrieving data
This step includes finding suitable data & getting access to the data from the data owner. The result is data in its raw form.

3 Data preparation:
This includes transforming the data from a raw form into usable data. This involves detecting & correcting different kinds of error in the data, combining data from different data sources & transform it..

4. Data Exploration:
The step involves finding patterns correlation & deviations based on visual & descriptive techniques to gain a deep understanding of the data.

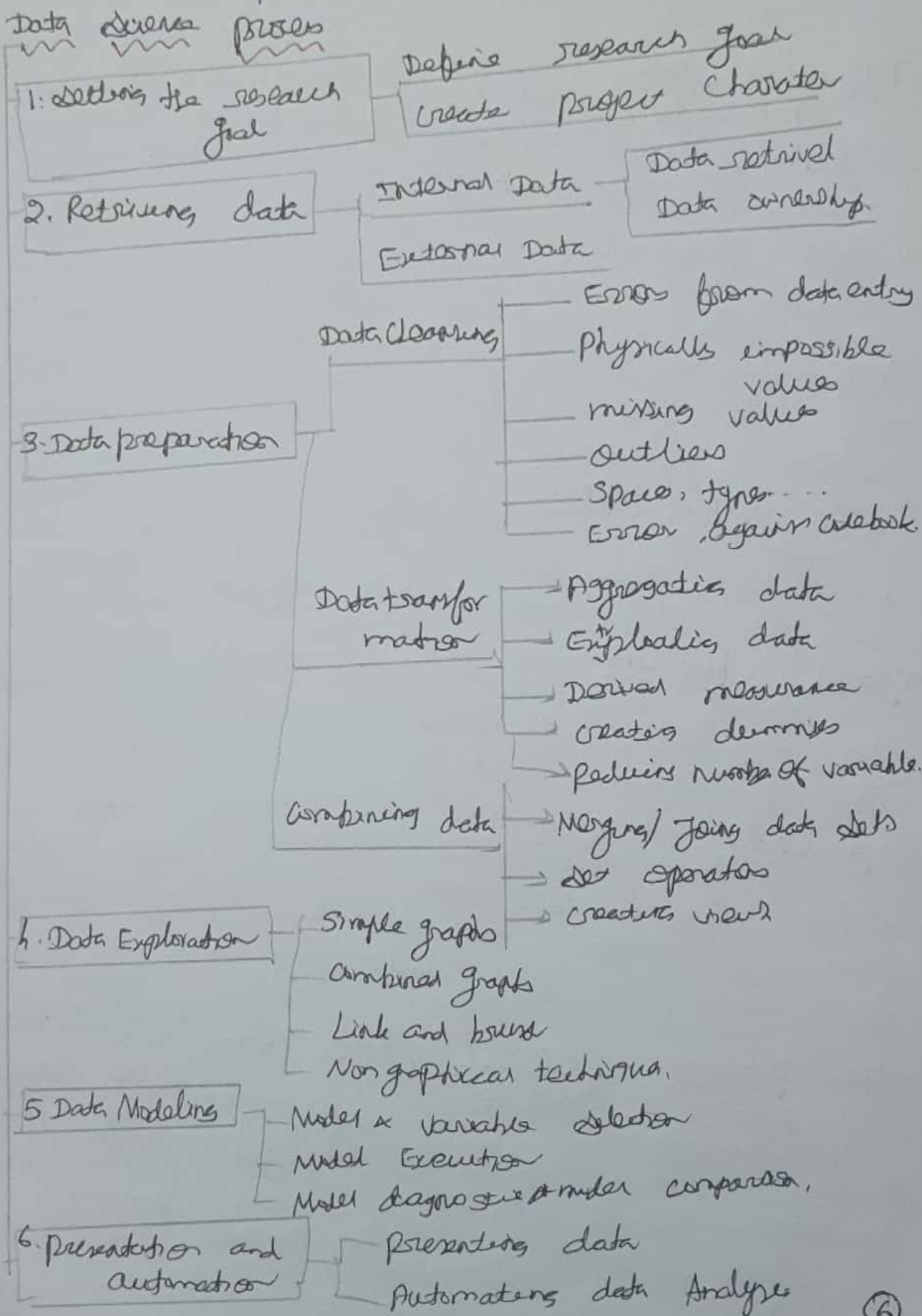
5. Model building:
Select the variables to build the model & also a modeling technique. building models with a goal of making predictions classifying objects.

6 Presentation & automation:
Finally present the result to the business Results can take many forms ranging from presentation to research reports. Automate the execution of the process if needed.

Step: Defining research goals and creating a project charater.
understanding the what, why & how of the project answering questions is the goal of first phase. outcome should be a clear research goal good understanding of the context well defined deliverables plan of action with a time table.

Overview of the data science process

The following figure summarizes the data science process.



Creating a proper Character:

→ A clear research goal
→ the project mission and context
→ how do you perform the analysis
→ what resources are expected to use
→ Proof of concepts
→ Deliverables and a measure of success A timetable

A client can use this information to estimate the project cost, data and people required for completion of project.

Step 2: Retrieving data

→ Second step of data science process is retrieving the required data.
→ many companies have already collected & stored the data for use internally
→ many organizations are making high quality data freely available for public & commercial use
→ Data can be stored in offline data repositories such as databases, data marts, data warehouses & data lakes
→ Getting access to data is difficult task. Do organizations have policies which controls the access of data everyone in that organization [ie Access rights - who can access what data]

A list of open data providers.

① Data.gov - the home of the US government's open data
② Open-data.europa.eu - the home of European commission's open data
③ Data.worldbank.org - open data initiative from the World Bank.

Step 3: Cleansing, Integrating and Transforming.

The data received from the data retrieval phase is like a diamond in the rough. It must be sanitized & prepared for use in modeling & reporting phase.

Cleansing data:

Data cleansing is a subprocess of data science process that focuss on removing errors in the data to make the data become true & consistant one.

An overview of common errors.

Error Description	possible solution:
→ mistake during data entry	→ manual overrules
→ Redundant white spaces	Use string functions
→ Impossible values	manual overrules
→ Missing values	Remove observation (or) value
→ outliers	Validate, if error treat as missing.
→ Deviation from a code book	match on keys (or) else use manual overrules
→ Different units of measurements	Recalculate
→ Different levels of aggregation	Bring to same level of measurment.

possible errors with examples:

① Data entry errors - Entry as "Godd" instead of "Good"

② Redudant white Spaces - white spaces at the end of a string causes errors. Very hard to detect.

③ Impossible values - people taller than 3 meters (or) people with an age of 299 year.

④ Outliers - is a data that lies abnormally far away from other values in a data set

15, 25, 30, 20, 35 ⇒ 98 outlier.

⑤ Deviation from a code book - Detecting errors in large data sets again a code book.

eg Salary / week and Salary / month week

A good practice is to correct errors as early as possible Week

combining data from different data sources.

different ways to combine data.

→ Joining

→ Appending table.

① Joining:
Enriching an observation from one table with information from another table

eg/

Client	Item	month
John	Pencil	January
Jane	pen	January

Client	Region
John	TamilNadu
Jane	Karnataka

Result of joining two tables

Client	Item	month	Region
John	pencil	January	TamilNadu
Jane	pen	January	Karnataka

② Appending tables: Adding the observation of one table to those of another table

Client	Item	month
John	coco	January
Jane	Paper	January

Client	Item	month
Jack	fruit	February
Ben	Juice	March

Client	Item	month
John	coco	January
Jane	paper	January
Jack	fruit	February
Ben	Juice	February

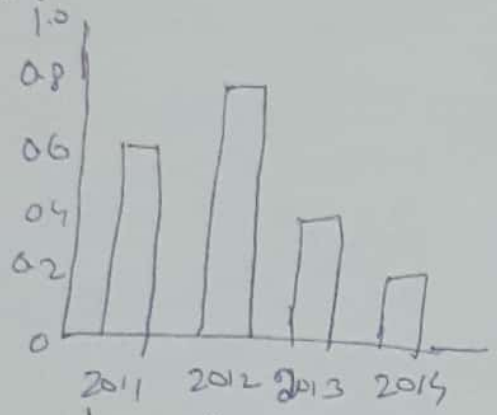
→ To avoid duplication of data, one can virtually combine data with views.
→ Also data reorganization can be done by adding calculated information

Steps: Transforming Data:
→ It makes the data suitable for Data modeling.
eg: Transforming x to log x
→ Reducing the no of variables (Principles Component Analysis)
→ Turning variables with dummies. Dummy Variables can only take two values true (1) or false (0)

Step: Exploratory Data Analysis.

→ Helps to gain deep understanding of data & the interactions between variables.

→ A bar chart, a line chart, & histograms used in exploratory analysis.

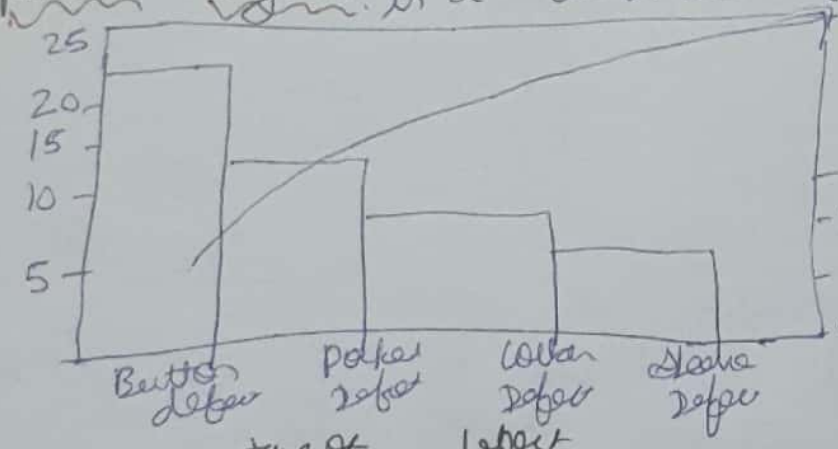


bar chart

Line Chart

Combined graph - pareto diagram

pareto diagram: is a combination of a bar graph & line graph

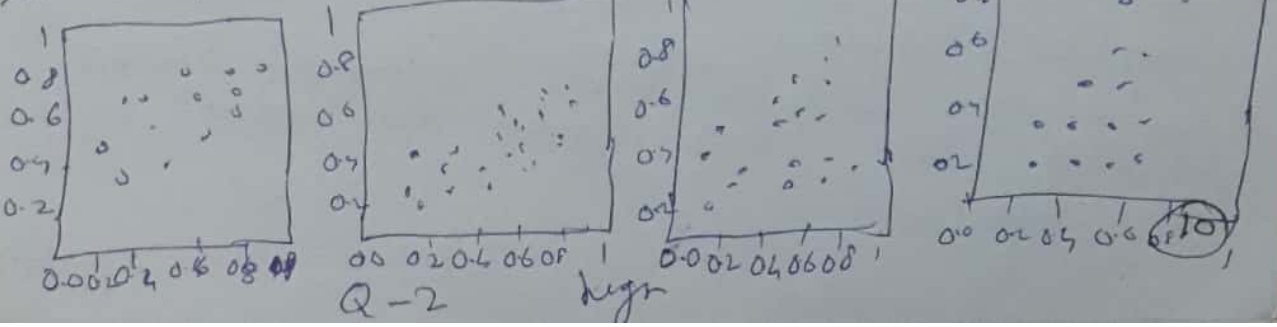


pareto charts helps in showing

Link and brush.

→ helps to combine & link different graphs & tables so changes in one graph are automatically transferred to the other graphs.

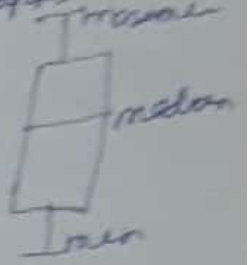
⇒ the following shows the average score per country for question



It shows the correlation between answers & also to see the outlier points in the graph.

Boxplot: shows the maximum, minimum median & other characterizing measures

→ Other are tabulation, clustering & other modeling techniques.



Steps build the model:

→ building models with the goal of making better predictions. Classifying objects.

→ builds a model in an iterative process

→ It depends on statistics & machine learning techniques

→ The components of data modeling

a) model & variable selection:

→ many modelling techniques are available.

→ Choosing the right model based on factors such as model performance. Projects requirements, easy to implement, difficulty in maintenance. & easy to explain.

b) model execution:

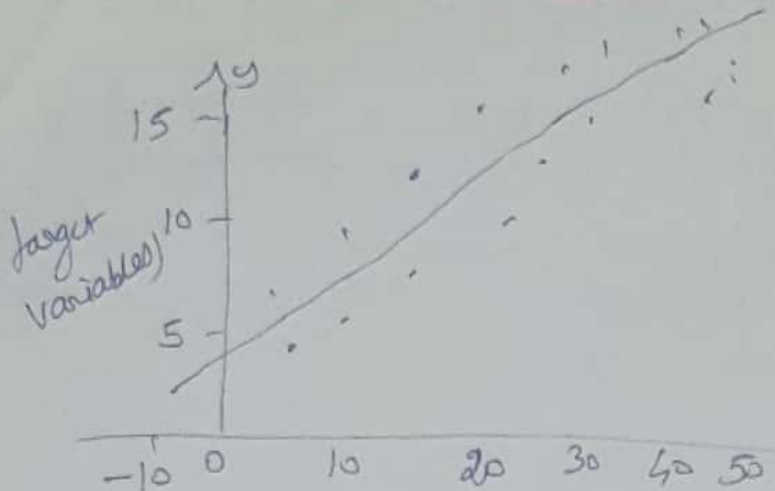
→ Implementation of chosen model in code

→ Python has libraries such as Stats Models(e)

→ Scikit-learn to implement models.

Implementation of Linear Regression Model:

```
import statsmodels.api as sm // import necessary python
import numpy as np                    module
predictors = np.random.random(1000).reshape(500)
target = predictors.dot(np.array([0.4, 0.6])) +
                    np.random.random(500)
LinRegmodel = sm.OLS(target, predictors) # to fit Linear
result = lm.RegModel.fit()              Regression on
result.summary() → Shows model Statistics   data
```



Linear regression tries to fit a line while maintaining the distance to each point. Linear equation is $y = 0.7858x + 1.1252$

Some important output parts of linear regression model

a) model fit: R-squared or adjusted R-squared is used → It is an indication of the amount of variation in the data that gets captured by the model.

b) predictor variables have a coefficient

c) predictor significance: how significant the predictors.

model diagnostics and model comparison.

1) Choose the best model based on multiple criteria

▲ multiple error measures are available one is mean square error (MSE)

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2$$

▲ It checks for each prediction how far it is from the actual.

▲ Choose the model with the lowest error

▲ verification of most out of model assumptions is called model diagnostics.

step 6: Presenting findings and building applications on top of them:

presentations { Presenting data
Automating data Analysis

presenting the findings to be world through powerpoint presentations.

Automate the models in order to do the tasks repetedly.

Data Mining:

Data is raw fact (or) disconnected fact
Information is the processed data
Knowledge - is derived from information by applying rules to it

Data ⟹ Information ⟹ Knowledge

Data mining: is the process of extracting hidden, valid and potentially useful patterns in huge datasets.

Steps in data mining.

1. Data cleaning: Remove noise & inconsistent data
2. Data Integration: where multiple data sources may be combined
3. Data selection: where data relevant to the analysis task are retrieved from the database.
4. Data transformation: where data are transformed consolidated into forms appropriate for mining by performing summary or aggregation operations.
5. Data Mining: An essential process where int methods are applied to extract data patterns
6. Pattern evaluation: to identify the truly interesting patterns representing knowledge
7. Knowledge Presentation: where visualization & knowledge representation techniques are applied to present mined knowledge to user

Type of data for mining:

1. Flat files: Transactions data, Time series Data
   Scientific measurements.
2. Data base data: Relational database.
3. Data warehouse data: is a repository of information collected from multiple sources, stored under a unified schema.
4. Transactional data: A transaction includes a unique transaction ID and a list of items making up the transaction

Data Warehousing:
It is the process of compiling and transforming data and making it available to users in a timely manner

Data Warehouse:
A single complete and consistent store of data obtained from a variety of different sources made available to end users in a way understand and use in a business context.
eg: Walmart - 24 TB (Tera Bytes)
National medical records - 10^{15} bytes in order of Exabytes

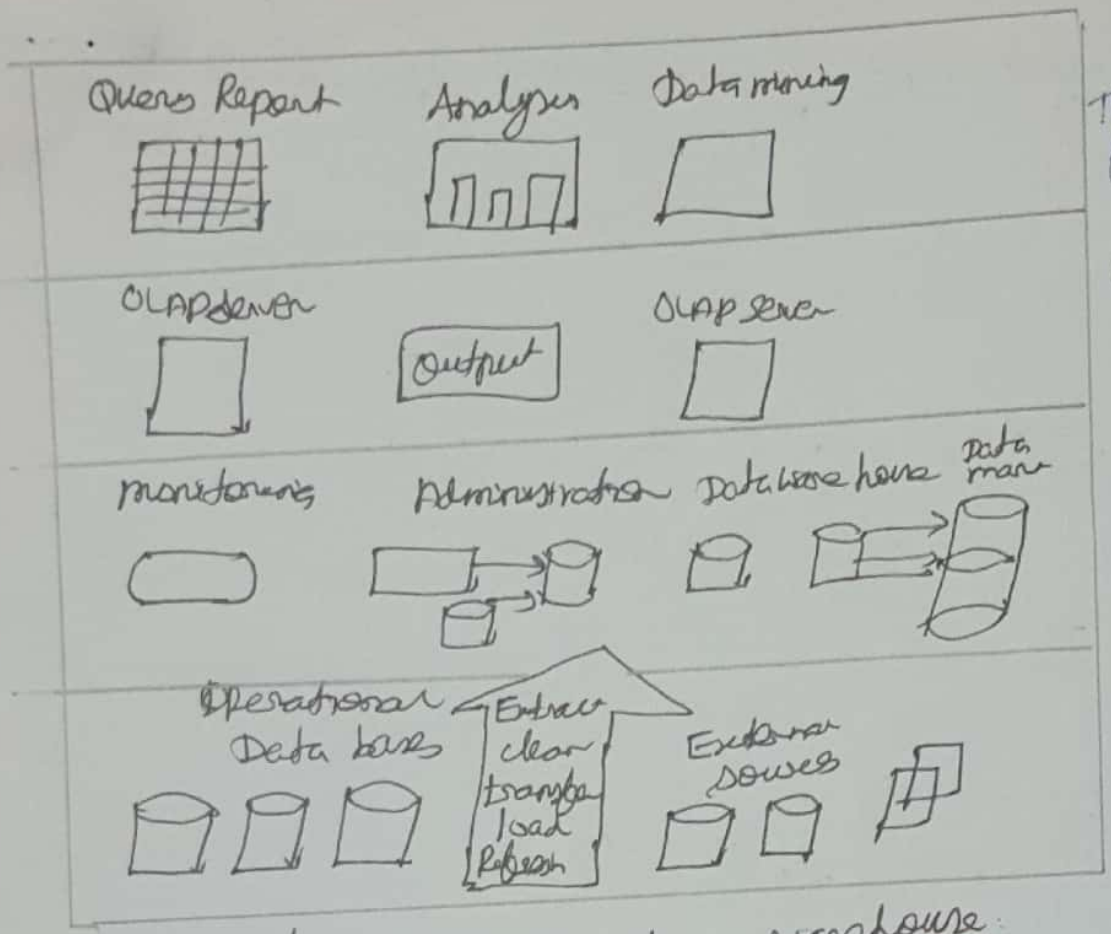
Key Characteristics of Data Warehouse
→ A data warehouse is a subject oriented - provides topwise information eg: [Sales - Inventory]
→ Integrated - Data from varied sources in a consistent format
→ Time varying - Data with time
→ non-volatile - Data remain unchanged collection of
→ data that is used primarily in organizational decision making.

Data Warehousing - Architecture.
Generally a data warehouse adopts a three-tier architecture.

Bottom Tier: It is the relational database server. It is the data base system. back end tools and utilities are used to feed data into the bottom tier. they perform extract, clean load & refresh functions

Middle Tier: Here OLAP server can be implemented in either Relational OLAP or multidimensional OLAP

Top Tier: the is front end client layer. this layer holds the query tools and reporting tools analysis tools and data mining tools



Process flow in Data warehouse:

There are 4 main process

1. Extract and load the data
2. Cleaning and transforming the data
3. Backup & archive the data
4. Managing queries and directing them to the appropriate data source.

Extract & Load process - [load manager]

This process extract data from operational data bases & External sources & load it into the data warehouse in Consistent form.

Clean & Transform process:

There are 3 steps in cleaning & transforming process clean & transform data with a structure partition the data - partition each table into multiple partitions.

Aggregation - It is required to speed up common queries.

Backup & Archive the Data - [warehouse manager]

In order to recover the data in the event of data loss, software failure or hardware failure it is necessary to keep regular backups.

Query Management process [covers managers]

The process performs the following functions

- manages the queries
- speed up the execution time of queries.
- directs the query to their most effective data sources
- monitors the actual query profile.

Online Analytical Processing [OLAP]

→ OLAP allows managers and analysts to get an insight of the information through fast consistent and interactive access to information.

→ It enables end users to perform better analysis of data in multiple dimensions.

OLAP Applications.

a) Finance and accounting
   - → budgeting
   - → Financial performance analysis
   - → Financial modeling.

b) Sales and marketing.
   - → Sales analysis and forecasting
   - → market research analysis
   - → customer analysis.

c) Production
   - → Production planning
   - → Defect analysis

→ OLAP Pre-calculates most of the queries namely aggregation / Joining and grouping

Thank you !

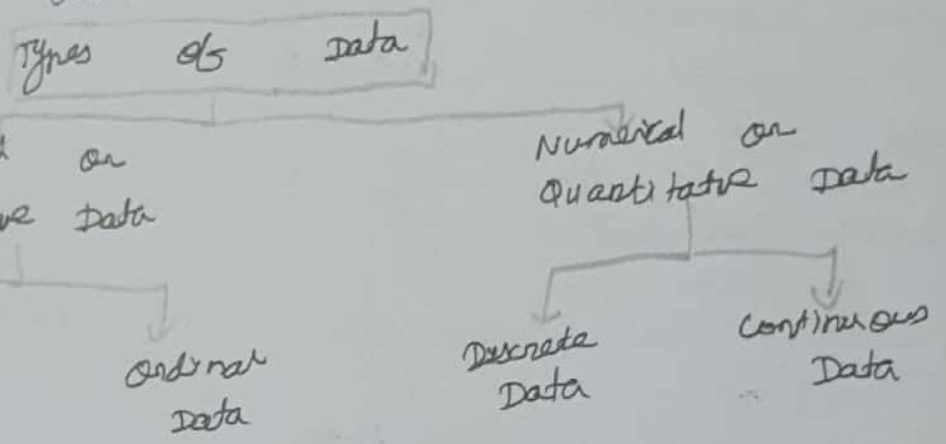
Unit - II

Types of Data - Types of Variables - Describing Data with Tables and Graphs - Describing Data with Average - Describing Variability - Normal Distributions and Standard (z) scores

Types of Data:

Data is the collection of actual observations or) scores in a survey or experiment whose can be used for statistical Analysis.

ex Structured and Stored data base. etc in table format - containing Numeric or text values.



Qualative (or) Categorical Data:

→ Qualative Data also known as Categorical Data.
→ Describes the Data that fits into the Categories
→ Qualative data are not Numerical.
Or → Categorical variables describes features such as
→ persons, gender, home, town, etc.
→ Categorical Data are birthdate, favourite sport, school.
post use Qualitative Data consist of word (yes or No)
Letters (Y or N)
Numerical code (0 or 1)

Ex Do you like Cartoons. Reply from four students

Students	Result
1	Y
2	N
3	Y
4	N

Nominal Data
Which helps to label the variable without providing the numerical value. but the data can be qualitative and quantitative.
eg: Nominal data are letters, Symbols, words, gender.

Ordinal Data:
Ordinal data is a qualitative data that. variable is type of data that follows a natural order. (group the variable into ordered categories. Ex/ this variable is found in surveys, finance, economics, questionaries. the ordinal data is commonly represented using a bar chart. these data are investigated and interpret.

Ranked Data:
→ Ranking is the data transformation in which numerical (or) ordinal values are replaced by their rank when the data are sorted.
→ arranging in ascending order and making from low to high.

2. Quantitative (or) Numerical Data
Quantitative Data represents the numerical value like how, much, how, often, how may.
eg: Numerical data are height, size, weight.
eg: Height of 5 students

Students	Height (inches)
1	160
2	165
3	170

Discrete Data
Discrete information contains only a finite number of possible values (those values cannot be subdivide meaningfully). It can be counted in whole numbers.

Discrete data Key characteristics.
You can count the data it usually units in whole numbers.
the values cannot be divide into smaller pieces and add additional Meaning

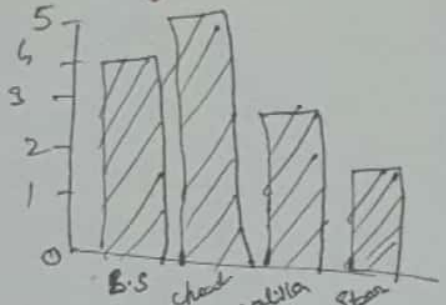
Discrete Data can be represented by
→ Bar Graph
→ Frequency table
→ Line Plot (Number Line)

Bar Graph is the most suitable way to represent discrete data. as finite values can be represented clearly through vertical bars.

Example:
In a survey with 14 children on their favourite ice cream flavour, It was found that 4 children like butterscotch flavour, 5 children like chocolate flavour, 3 children like vanilla flavour & 2 children like strawberry flavour.

→ It is an eg of discrete data we can count the number of children who like a particular flavour of ice cream.

bar graph:

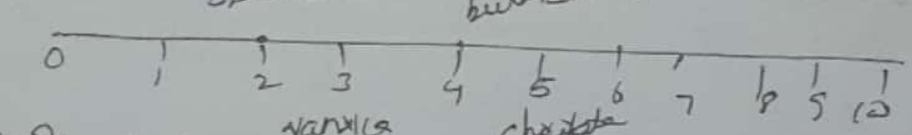


frequency table:

flavour	Tally marks	Number of children
butterscotch	IIII	4
Chocolate	HHI	5
Vanilla	III	3
Strawberry	II	2

Values are represented by tally marks.

Number Line: we mark the value of each variable on the number line.



2.2 Continuous data:

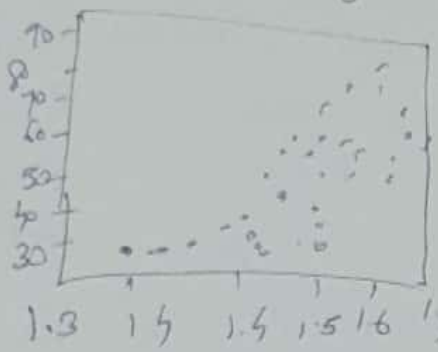
→ Continuous data is data that can be calculated. It has an infinite number of probable value that can be divided within a given specific range.

eg: Temperature Range.

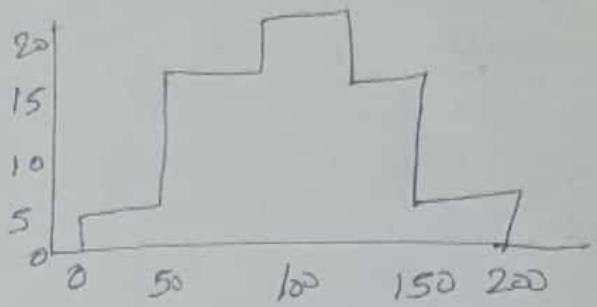
→ the amount of time required to complete a project

the height of Children

Histogram & scatterplot is given continous data those graphs are designed to handle values



height & weight are continous variables



Annual Revenue.

Difference between Discrete & continous data.

Discrete Data	Continous data.
① Values can take only specific values	① Variable can take any value with in a range.
② We can count the values in a discrete data series	② We cannot count the values in a continous data series. Instead we measure the values
③ Whole numbers are used to write discrete data values	③ Real numbers are used to write continous data value.

Types of variables..

there are four types of variables.

i) Numerical ii) categorical iii) Date time iv) Mixed

Numerical Variable..

This category of variables deals with the Numbers only. It is subdivided into..

i) Discrete ii) Continous.

Categorical Variables.

This category deals with two categories.

i) Ordinal ii) Nominal

Date & Time variables

The category of variable deals with the date & time aspects. this category can obtain types of value.

⇒ only having date

⇒ only having Time

⇒ Having both date & time.

Mixed Variables.

It deals with the collection of multiple values for the multiple observation of a specific variable.

i) Number or real / String of different observation
ex performance of a student (GPA, percentage, or grade)
ii) Numeric data label / Strings for the same observation.
ex Caller number, Voter Registration Number.

Independent Variable.

It is a singular characteristic that the other variables in your experiment cannot change. ex Age.

Dependent Variable.

It reads on and can be changed by other components.
ex Grade on exam. Time you spend studies (dependent)

2.3. Describing data with Tables and Graphs.

There are 4 ways to represent the data presented.

① Tables ② Pictographs ③ Line graphs ④ Bar Graph

Table are the simplest way to represent data

A Pictograph uses images to represent a certain number of items.

A Line graph Plots individual data points as dots and connect them with Lines.

Bar graphs use bars of different heights to represent data

Table (Frequency Distributions)

Organization of Data.

⇒ Statistics refers to the collection, organization, distribution & interpretation of data on a set of observation.

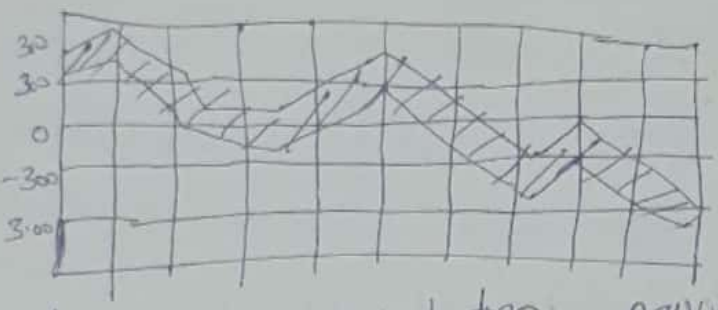
⇒ A frequency distribution is a collection of Observation Produced by sorting Observation into class and showing their frequency (f) of occurrence in each class

⇒ there are five frequency Distribution they are.

Types of frequency distribution.

i) ungrouped frequency distribution
ii) grouped frequency distribution
iii) Cumulative frequency Distribution
iv) Relative frequency distribution
v) Relative Cumulative frequency distribution

ungrouped frequency distribution



⇒ frequency distribution provides with raw data on random values. In this case, we may have to make either for individual observations or class intervals.

⇒ Let the test score of all 20 Students be as follows 23, 26, 11, 18, 09, 21, 23, 30, 32, 11, 20, 11, 13, 23, 11, 29, 25, 21, 26.

Marks obtained in the test	tally marks	No of Students (Frequency)
09	I	1
11	IIII	4
13	I	1
18	I	1
20	I	1
21	II	2
22	I	1
23	III	3
25	I	1
26	III	3
29	I	1
30	I	1
total		20

Grouped Frequency Distribution

→ It shows the scores by grouping the observations into intervals and then lists these intervals in the frequency distribution table. the intervals in grouped frequency distribution are called class intervals or limits.

→ Grouped of data in the form of class intervals to tally the frequency for the data that belongs to that particular class interval. the lowest number in a class interval is called lower limit. the higher number is upper limit

Marks obtained in the test (class-interval)	No. of students (frequency)
0-5	3
5-10	11
10-15	38
15-20	34
20-25	9
25-30	5
Total	100

Cumulative frequency Distribution.

The cumulative frequency is the total of frequencies in which the frequency of first class interval is added to the frequency of the second class interval and then the sum is added to the frequency,

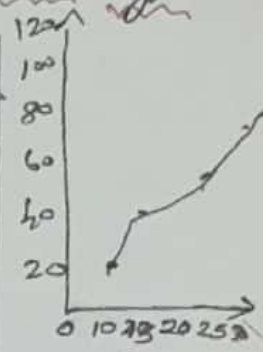
Steps to construct less than C.F.C.

① Mark the upper limits on the horizontal axis as x axis.
② mark the C.F on the vertical axis as y axis.
③ Plot the point (x, y) in the coordinate plane where x represents the upper limit value

(4) Finally, Join the points and draw the Smooth curve.
(5) The curve so obtained gives a cumulative frequency distribution graph of less than type.
(6) To draw the frequency distribution. if less than type.

table cumulative frequency distribution table of more than type.

Level of Exam	Age group class boundary	Age Groups	Number of Participate	cumulative frequency
Level 1	10-15	Less than 15	30	20
Level 2	15-20	Less than 20	32	52
Level 3	20-25	Less than 25	16	70
Level 4	25-30	Less than 30	30	100



Then is added to the frequency of the third class interval and so on.

types i) less than cumulative frequency
ii) Greater than cumulative frequency.

The cumulative total frequencies starting from the highest class to the lowest class.

Relative frequency distribution
It shows the proportion of the total number of observations associated with each value or class of values and is related to a probability distribution.

Height	frequency	Rel. frequency.
57 or less	1	0.025
57.1 to 58.6	1	0.025
58.6 to 60.1	3	0.075
60.1 to 61.7	6	0.15
61.7 to 63.3	8	0.2
63.3 to 64.8	11	0.275
64.8 to 66.4	3	0.075
66.4 to 68.0	7	0.175
Total	40	1

Cumulative Relative frequency.

It can be found by dividing the cumulative frequency of each interval by the total number of observations.

$$C.R.F.D = \frac{\text{cumulative frequency of each interval}}{\text{total number of observations.}}$$

eg) following datasets.

{1, 1, 1, 1, 1, 1, 3, 3, 3, 3, 3, 5, 5, 5, 5, 5, 11, 11, 11, 11, 11, 11, 11, 11, 7}

first are → find frequency table then we find its cumulative Relative frequency

Count	frequency	Cumulative frequency	Cumulative Relative frequency.
1	5	5	5/25 = 0.2 20%
3	5	10	10/25 = 0.4 40%
5	6	16	16/25 = 0.64 64%
7	1	17	17/25 = 0.6 68%
11	8	25	25/25 = 1.0 100%

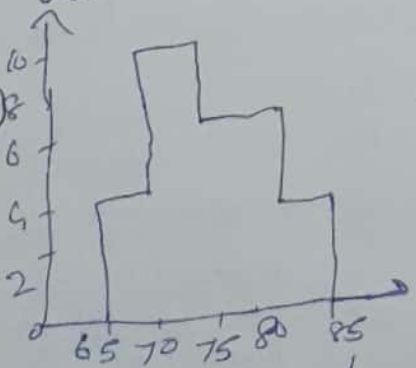
Qualitative data from frequency table are presented by histogram.

Histogram:

It is used to visualize the distribution of data among different intervals as a series of vertical bars

eg) Construct a histogram for the following frequency distribution table distributes the frequency of weight of 25 Student in a class.

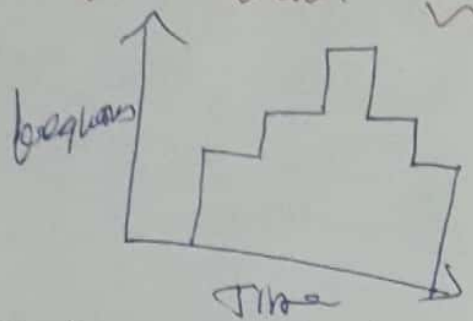
weight	frequency (No. of students)
65-70	4
70-75	10
75-80	8
80-85	4



horizontal axis → Start from 65 not from 0
1 unit = 15

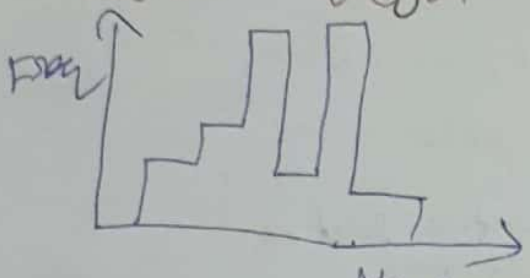
Vertical axis: frequency varying from 4 to 10 1 unit = 2

1) Bell shaped histogram



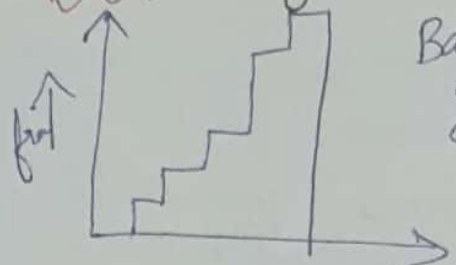
→ It has a single peak
→ has only one peak at the time.

② Bimodal histogram



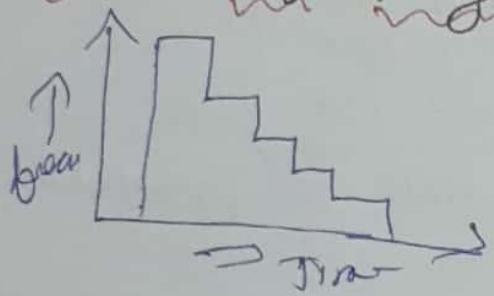
It has two peaks

③ Skewed Right Histogram

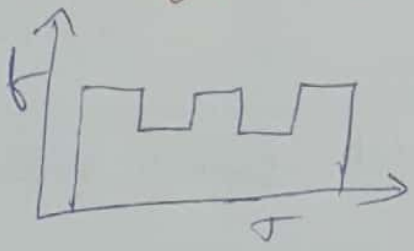


Bars of the histogram are skewed to Right.

④ Skewed Left histogram



⑤ Uniform histogram



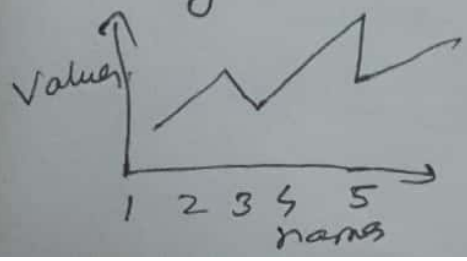
All the bars are more or less the same height

Pictograph:

Pictorial representation of data using images like or symbols.

Flavours	Number of children	
Chocolate	○○△	Key: ○ Represents ½ children
butterscotch	○○○○△	$2 \times \frac{1}{2} + \frac{1}{2} \times \frac{1}{2} = 9$
Vanilla	○○○	$8 + 1 = 9$
Strawberry	○○	

Line Graph

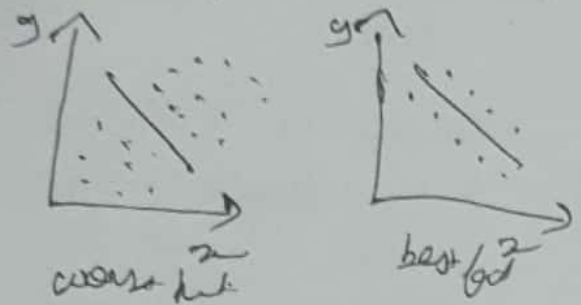


Marks	1	2	3	4	5
Runs	10	30	50	20	40

x axis → marks
y axis → values

Scatter Graph

A scatter xy plot has points that show the relationship between two sets of data



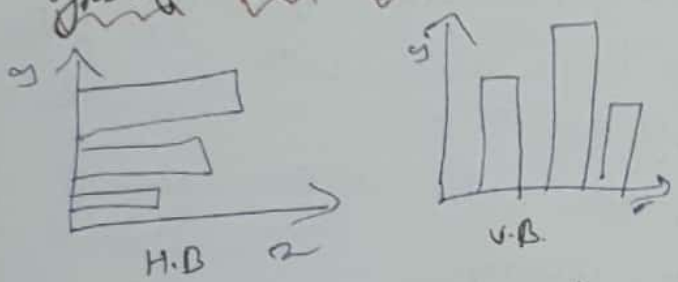
the line is drawn which is nearest to almost all the points.

Bar Graph:

It is the pictorial representation of data in the form of vertical & horizontal bar

types of Bar Graph:

i) horizontal bar chart
ii) vertical bar chart



2. Describing Data with Averages:

Different measures can be computed to describe how data are distributed. There are classified into 3 categories.

i) Measures of location
ii) Measures of variation
iii) Measures of shape.

→ location measure is understood as the "center" of the data eg) Average price of a new product middle of a new product, or the most frequent price of a new product.

→ Most widely used measures of the data "center" are the ⇒ Mean (average)
⇒ Median
⇒ Mode.

Average:

Average represents the whole value and it lies between the minimum & maximum value of the data. Hence it is the representative figure of the entire data.

Definition:

Clark and Sekkade defines as "average is an attempt to find one single figure to describe whole of figures.

Objective of an Average.

→ The representative value explains the character of the whole data. One value may represent even thousand and millions of values.

→ policy decision may be taken with the help of one representative figure.

→ It helps to compare with other data

Characteristics of an average

- Average can easily understand the single representative value.
- It is very simple to calculate.
- It should Represent the entire data.

Types of Average.

1) Arithmetic Mean → Simple Arithmetic Mean
   → Weighted Arithmetic Mean
2) Median
3) Mode
4) Geometric Mean
5) Harmonic Mean

Arithmetic Mean

The Mean of data indicates an average of the given collection of data.

It is equal to the sum of all the values in the graph data divided by the total number of values $\bar{x} = \frac{x_1 + x_2 + x_3 \cdots x_n}{n}$ $= \frac{\sum_{i=1}^{n} x_i}{n}$

Calculating the Mean when the frequencies of the observations is given

$$\bar{x} = \frac{f_1x_1 + f_2x_2 + f_3x_3 + \cdots + f_nx_n}{f_1 + f_2 + f_3 + \cdots f_n} = \frac{\sum f_i x_i}{\sum f_i}$$

In a class 30 students, marks obtained by students in mathematics out of 50 is tabulated below. Calculate the Mean of the data

marks obtained	Number of Student	Class mark	$f_i x_i$
10-20	5	15	75
20-30	5	25	125
30-40	8	35	280
40-50	12	45	540
total	$\sum f_i = 30$		$\sum f_i x_i = 1020$

$$\bar{x} = \frac{\sum f_i x_i}{\sum f_i} = \frac{1020}{30} = 34$$

The Mean of the given data is 34

Median: Median is the middle most value in a given dataset after it is arranged in ascending order.

If the number of items in the dataset is odd $= \left[\frac{n+1}{2}\right]^{th}$ term

even $= \left[\frac{\left[\frac{n}{2}\right]^{th} \text{term} + \left[\frac{n}{2}+1\right]^{th} \text{term}}{2}\right]$

n → total number of observation

find the Median of the following data 48, 29, 50, 67, 73
Arranged in ascending order no of observations = 5

⇒ Median of odd data $= [n+1/2] = [5+1/2] = 6/2 = 3$

Median is 3 observation | Median = 50 |

find the median of the above grouped data, to find median. We need cumulative frequency.

class intervals	No of girls	cumulative frequency
120-130	2	2
130-140	8	2+8 = 10
140-150	12	10+12 = 22
150-160	20	20+22 = 42
160-170	8	42+8 = 50

n = sum of cf N = 50 $n/2 = 25$

N: median class = 150 − 160

here $l = 150$, $c = 22$, $f = 20$, $h = 10$

$$\text{Median} = l + [(n/2 - c)/f] * h$$

$$= 150 + [(50/2 - 22)/20] \times 10 = 150 + 1.5$$

Median = 151.5

Ans:

Weekly Expenditure	0-1000	1000-2000	2000-3000	3000-4000	4000-5000	Total
No of families	34	12	43	60	51	200

Mode:

It is the value which has the maximum frequency. It is possible to have more than one value which has the same maximum frequency.

find the mode of ungrouped data.

Wickets taken by a bowler in 10 Cricket matches are

2, 6, 4, 5, 0, 9, 1, 3, 2, 3

to find the mode of ungrouped data.

Number of wickets	0	1	2	3	4	5	6
Number of matches	1	1	3	2	1	1	1

maximum no of matches = 3

Mode of the given data = 2

It is a ungrouped data.

$$\text{Mode} = l + \left(\frac{f_1 - f_0}{2f_1 - f_0 - f_2}\right) \times h$$

Where,

→ f_1 is the frequency of the model class

→ f_0 is the frequency of the class preceding the model class

→ f_2 is the frequency of the class succeeding the model class

→ h is the size of the class interval

→ l is the lower limit of the model class

mode of grouped data.

eg

Size of family	1-3	3-5	5-7	7-9	9-11
No of families	7 (f_0)	8 (f_1)	2 (f_2)	2	1

maximum class frequency = 8

class interval = 3-5

lower Limit of the model class = 1 = 3

class size (or) size of the interval $h = 2$

frequency of the modal class $f_1 = 8$

⇒frequency of the class preceding to modal class $f_0 = 7$

⇒frequency of the class succeeding of modal class $f_2 = 2$

Know that the formula to find the mode of the grouped data.

$$\text{Mode} = l + \left(\frac{f_1 - f_0}{2f_1 - f_0 - f_2}\right) \times h$$

$$\text{Mode} = 3 + \left(\frac{8-7}{2(8)-7-2}\right) \times 2 \Rightarrow 3 + \left(\frac{1}{16-7-2}\right) \times 2$$

$$= 3 + \left(\frac{1}{7} \times 2\right) \Rightarrow 3 + \frac{2}{7} = \frac{21+2}{7}$$

$$= 23/7$$

3.5 Describing variability:

⇒Variability (also called spread or dispersion) refers to how spread out a set of data is.

⇒Variability gives you a way to describe how much data sets vary and allows you to use statistics to compare your data to other set of data. the four main ways to describe variability.

1. Range
2. Interquartile Range
3. Variance
4. Standard deviation

Range:

Range is the simplest method of studying dispersion it is defined as the difference between the value of the smallest item value of the largest item included in the distribution

$$\boxed{\text{Range} = L - S}$$

L → Largest Item

S → Smaller item

the relative measure of Corresponding to the Range is called its coefficient of Range.

Coefficient of range $= \frac{L-S}{L+S}$

eg) the following are the prices of shares of AB company Ltd, from monday to saturday. Monday, Tuesday, Wednesday, Thursday, Friday, Saturday.

Price: 200, 210, 208, 160, 220, 250.

Calculate Range and its coefficient

Range = L − S

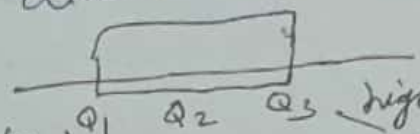
Range $= 250 - 160 = 90$

Coefficient of Range $= \frac{L-S}{L+S} = \frac{250-160}{250+160} = \frac{90}{410} = 0.22$

Inter qualitive Range (IQR)

⇒ It is the range for the middle 50 percent of the scores. they are value that divide the dataset into 4 equal parts.

$IQR = Q_3 - Q_1$



⇒ Inter qualite range gives you the spread of the middle of your distribution.

⇒ The interqualitive range is the third quarite (Q_3) minus the first quartile (Q_1). It will provide the range of the middle half of a dataset.

Quartile deviation $QD = \frac{Q_3 - Q_1}{2}$

Coefficient $QD = \frac{Q_3 - Q_1}{Q_3 + Q_1}$

⇒ Lowest value
⇒ Q_1 : 25th percentile
⇒ Q_2 : the median
⇒ Q_3 : 75th percentile
⇒ higher value (Q_4)

eg) find out the value of QD and its coefficient from the following data.

Roll No: 1, 2, 3, 4, 5, 6, 7

marks: 20, 28, 40, 12, 30, 15, 50

$n = 7$

Calculating Quartile Deviation

Marks arranged in ascending data 12, 15, 20, 28, 30, 40, 50

$Q_1 = \frac{1}{4}(N+1)^{th}$ term $= \frac{1}{4}(7+1) = 3/4$

odd = n + 1 term
even = n th term.

$Q_1 = 2$th term is (15)

$Q_3 = 3/4\,(N+1)$th $= 3/4 \times 18 = 6$ th term $=$ (40)

$OD = 40 - 15/2 = 25/2 = 12.5$.

Coefficient of $QD = \frac{Q_3 - Q_1}{Q_3 + Q_1} = \frac{40-15}{40+15} = \frac{25}{55} = 0.45$

Standared deviation.

The Std deviation tells you how tightly your data is clustered around the mean (the avrag)

A small SD indicates that your data is tightly clustered

A larg SD tells you that your data is more spread a part.

Std Dist $= \sqrt{\text{variance}}$

Stardard Deviation formula for Population & sample.

formula	Explanation
$\sigma = \sqrt{\frac{\Sigma(X-u)^2}{N}}$	σ - population Std deviation Σ - sum of X - each value u - population mean N - Number of value in the population

Stardard deviation formula for Samples.

$$s = \sqrt{\frac{\Sigma(x-\bar{x})^2}{n-1}}$$

s - sample Stadard deviation
Σ - Sum of
x - each value
$\bar{x}$ - Sample mean
n - number of values in the sample.

Variance:

It is the square root of standard deviation

Variance of a data set is how spread out your data

A small no of variance means: dataset is tightly clustered together

A large no for the variance means: The values in the dataset is more spread apart

$\sigma^2 = \frac{S^2}{n}$ = $\frac{\text{Sum of squares}}{N}$, Stddeviation = $\sqrt{\text{variance}}$

Variance formula for populations → entire or complete set

$\sigma^2 = \frac{\Sigma (x-\mu)^2}{N}$

σ^2 : population variance
Σ - Sum of each value
μ - population mean
n - no of values in the population

Variance formula for samples: → subset of population

$s^2 = \frac{\Sigma (x-\bar{x})^2}{n-1}$

s^2 - sample variance
Σ → Sum of each value.
$\bar{x}$ = sample mean
n - number of values in the sample.

Ex: data values are arranged in assending order

7, 11, 11, 15, 20, 20, 28

x	$\bar{x}$	$x-\bar{x}$	$(x-\bar{x})^2$
7	16	-9	81
11	16	-5	25
11	16	-5	25
15	16	-1	1
20	16	4	16
20	16	4	16
28	10	12	144

$\Sigma (x-\bar{x})^2 = 308$

Mean = $\bar{x} = \frac{\Sigma x}{n}$

$\bar{x} = 7+11+11+15+20+20+28 / 7$

$\bar{x} = 112/7 = 16$

Variance formula for sample

$s^2 = \frac{\Sigma (x-\bar{x})^2}{n-1} = \frac{308}{7-1} = \frac{308}{6}$

$s^2 = 51.3333$

Standard Deviation $\sigma = \sqrt{\frac{\Sigma (x-\bar{x})^2}{n-1}}$

$\sigma = \sqrt{\frac{308}{6}} = \sqrt{51.333}$

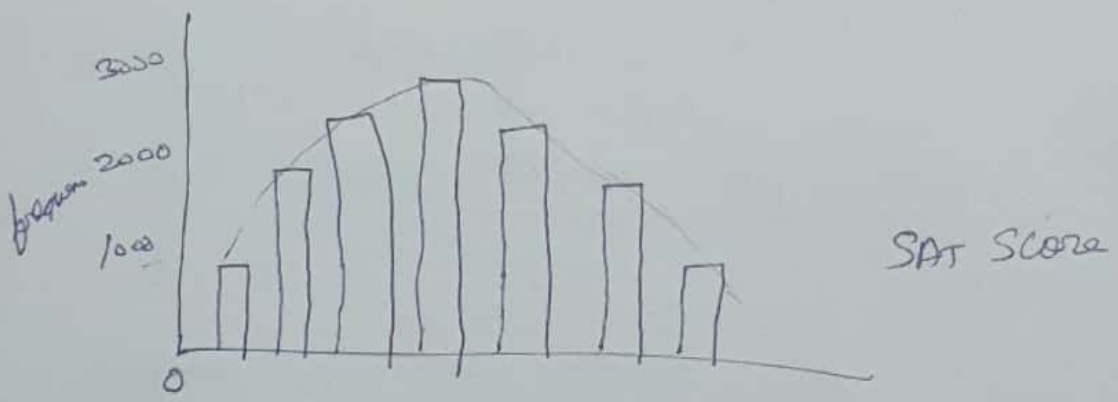
$\sigma = 7.165$

2.6. Normal Distributions and Standard (z) scores.

⇒ Normal distribution also known as Gaussian distribution, is a probability distribution that is symmetric about the mean, showing the data near the mean are more frequent in occurrence than data far from the mean.

→ In graphical form the normal distribution appears as a "bell curve".

→ In a Normal distribution, data is symmetrically distributed with no skew. When plotted on a graph the data follows a bell shape, with most values clustering around a central region.

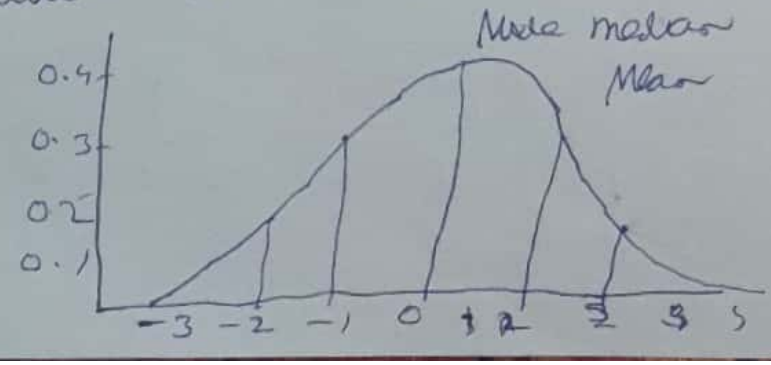


Properties of Normal distribution.

⇒ the mean, median, mode are exactly the same

⇒ the distribution is symmetric about the mean. half the values fall below the mean and half above the mean

⇒ the distribution can be described by two values the mean and its Standard deviation.



examples.

① Movie rating reflect ordinal measurement because they can be ordered from most to least restrictive NC17, R, PG13 and G, the ratings of some films shown as follows.

PG	PG	PG	PG13	G
G	PG13	R	PG	PG
R	PG	R	PG	R
NC17	NC17	PG	G	PG13

a) construct freq distribution
b) convert to relative frequency, exp as %
c) construct a cumulative freq. dist

d) find the appropriate percentile rank for those films with a PG rating.

sol

Rating	(a) frequency	(b) Rel. frequency	(c) CF	CF%
NC17	2	$2/20 \times 100 = 10$	$18+2=20$	$20/20 \times 100 = 100$
R	4	$4/20 \times 100 = 20$	$14+4=18$	$18/20 \times 100 = 90$
PG13	3	$3/20 \times 100 = 15$	$11+3=14$	$14/20 \times 100 = 70$
PG	8	$8/20 \times 100 = 40$	$3+8=11$	$11/20 \times 100 = 55$
G	3	$3/20 \times 100 = 15$	3	$3/20 \times 100 = 15$
	$\Sigma f = 20$			

② 2, 17, 5, 5, 3, 28, 7, 5, 8, 5, 6, 2, 12, 10, 4, 3

1) find Mean, median mode.
2) without constructing a freq distribution or graph would you characterize the shape of the dist as balanced, tails skewed or -Ve skewed

sol Sort the list -

2, 2, 3, 3, 4, 5, 5, 5, 6, 7, 8, 10, 12, 17, 28

$$\text{Mean} : \frac{\Sigma x}{n} = \frac{2+2+3+3+4+5+5+5+6+7+8+10+12+17+28}{15}$$

$$\text{Mean} = 7.8$$

(1) Mode = 5

Median → odd ⇒ $\left(\frac{n+1}{2}\right)$

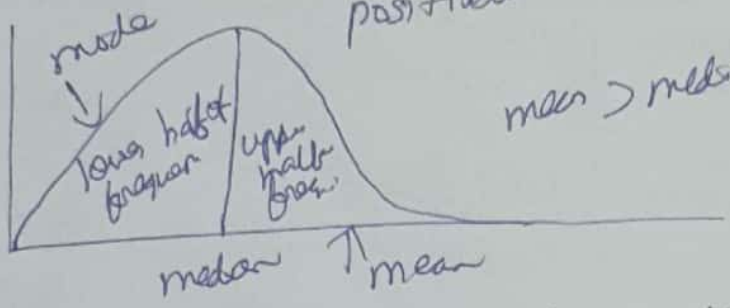
$= \left(\frac{15+1}{2}\right) = 16/2 = 8$th (5)

Median = 5

(2) Mean > median

7.8 > 5 +ve skewed.

positively skewed distribution.



mean > median

(3) construct Steam & Leaf display for the following 10 scores obtained from a group of 4-yr old Children.

120	98	118	117	79	111
126	85	88	124	104	113
106	141	123	137	78	96
102	132	109	106	143	

Separate unit digit.

Remains digit / unit digit

Steam Leaf

Sol

120 → 12-0
98 → 9-8
118 → 11-8
117 → 11-7
79 → 7-9
126 → 12-6

85 → 8-5
88 → 8-8
124 → 12-4
104 → 10-4
113 → 11-3
108 → 10-8
141 → 14-1

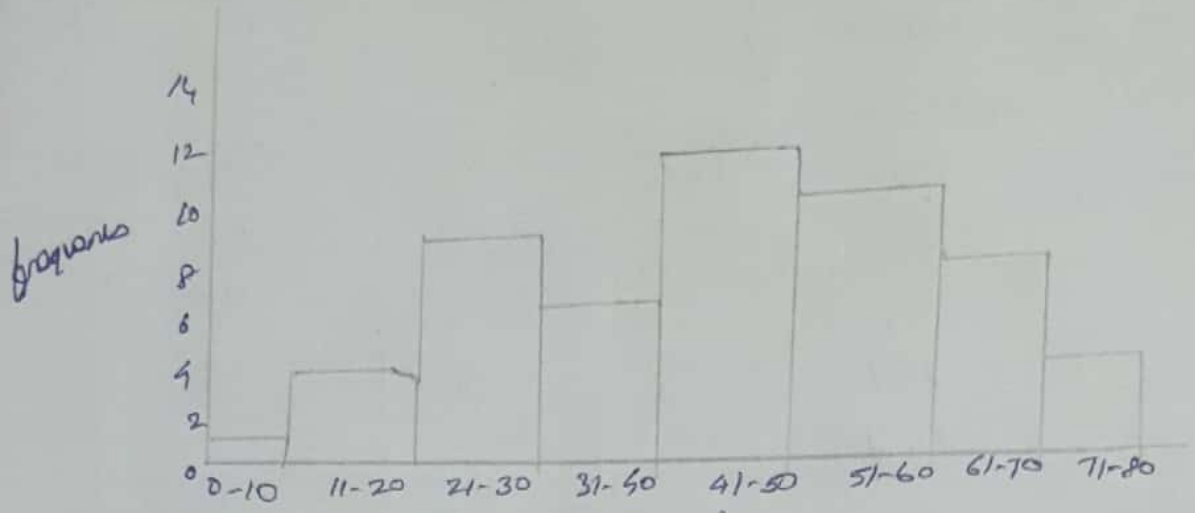
123 → 12-3
137 - 13-7
78 → 7-8
96 → 9-6
102 → 10-2
132 → 13-2
109 → 10-9

106 → 10-6
143 - 14-3

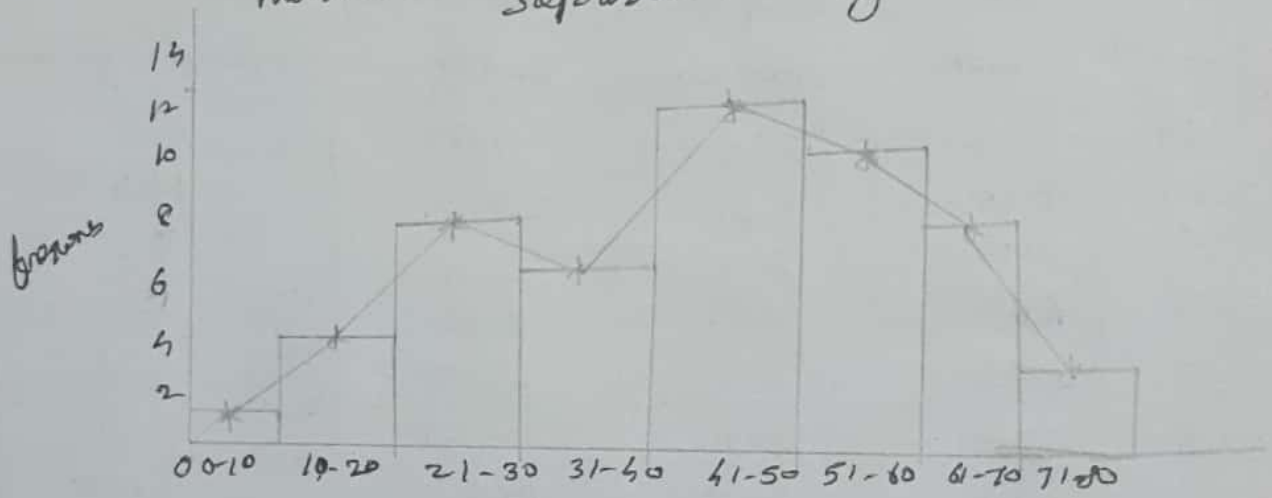
Steam	Leaf
7	8
8	5 8
9	8 9 6
10	8 2 9 6 4
11	8 7 1 3
12	0 6 3 4
13	2 7
14	1 3

(4) Construct a histogram for the following data.

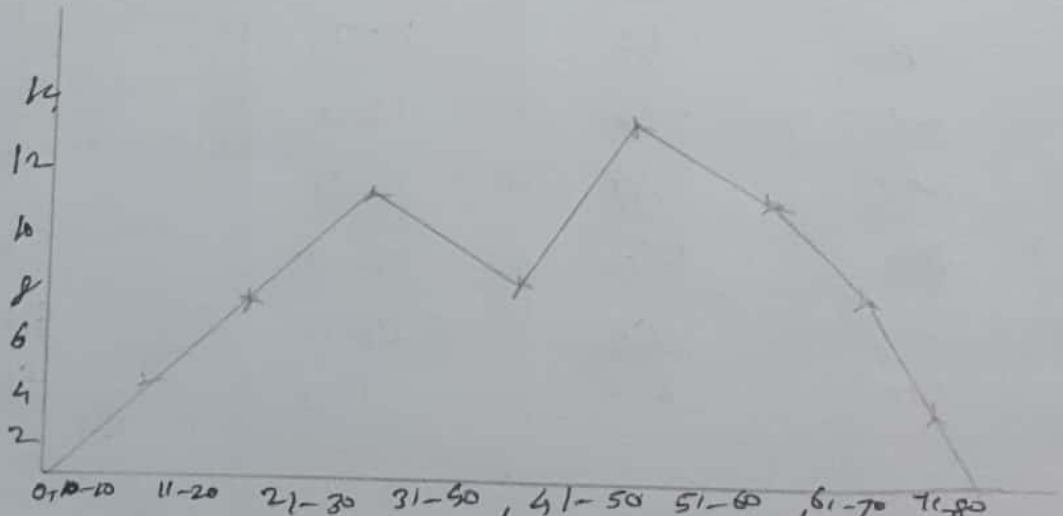
Age	0-10	11-20	21-30	31-40	41-50	51-60	61-70	71-80
frequency	1	4	8	6	12	10	7	3



Histogram representing age



Connecting midpoints.



Constructing frequency polygon.

(5) ex computation of population variance.

No of items (N)	Observation (x)	μ	$x-\mu$	$(x-\mu)^2$
1	105		4	16
2	100		-1	1
3	102		1	1
4	95	101	-6	36
5	100		-1	1
6	98		-3	9
7	107		6	36
σ^2	14.29			

(6) computation of Mean, sum of squares, population variance and sample variance.

Year	Month	New Use	x-mean	Standard Difference
2019	Jan	242	9.0	81
2019	Feb	221	-12.0	144
2019	Mar	225	-8.0	64
2019	Apr	233	0.0	0
2019	May	211	-22.0	484
2019	Jun	224	-9.0	81
2019	Jul	243	10.0	100
2019	Aug	213	-20.0	400
2019	Sep	244	11.0	121
2019	Oct	248	15.0	225
2019	Nov	249	16.0	256
2019	Dec	247	14.0	196
2020	Jan	246	13.0	169
2020	Feb	216	-17.0	289

Mean = 333.000

sum of squares : 2610.00

Variance of population : 186.429 $\sigma^2 = \sum_{i=1}^{n} \frac{(x_i - \mu)^2}{N}$

Variance of sample : 200.769 $S^2 = \sum_{i=1}^{n} \frac{(x_i - \bar{x})^2}{n-1}$

(1) Computation of Standard Deviation.

No	Returns	Return - Mean	(Return - Mean)^2
1	72	12	144
2	45	-15	225
3	58	-2	4
4	84	24	576
5	60	0	0
6	10	-50	2500
7	91	31	961
8	65	5	25
9	55	-5	25
10	60	0	0
Total	600		4460
Mean	60		
Standard Deviation	22.26		

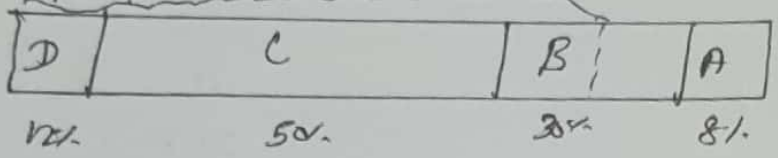
Standard deviation = $\sqrt{\text{variance}}$

Standard deviation is a rough measure of the average (or standard) amount of how much scores deviate on either side of the mean.

(8) In a test 12% got D grade. 50% got C grade. 30% got B grade and 8% got A grade.

Grade	Frequency	cumulative frequency	Frequency percentage
A	16	200	8
B	60	184	30
C	100	124	50
D	24	24	12
Total	200		

Percentile Rank: Percentile Rank B: 77%



12% got D

50% got C

Half of the 30% got B

for the total percentile Rank of

12% + 50% + 15% = 77%

using the formula.

$$RR = \frac{CF - (0.5 \times F)}{N} \times 100$$

the frequency of B = 60. the cumulative frequency of B is 184. the N value is 200 substituting these values in the above formula

$$RR = \frac{184 - (0.5 \times 60)}{200} \times 100$$

RR = 77%

unit - III

Describing Relationships

correlation - Scatter plots - correlation coefficient for quantitative - computational formal for correlation coefficient - regression - regression Line least squares regression Line - Standard error of estimate - interpretation of STE - multiple regression equations - regression toward the mean

correlation: Definition:

→ According to simple, correlation is the relationship that exists between two variables, when one variable changes, the other variable changes as well (decreases or increases)

→ Mathematically we can say a function has a purpose to predict a value, by converting input (x) to output (f(x))

Types of correlation: -1 to +1

i) positive +1 and negative -1 correlation
ii) simple correlation and multiple correlation
iii) partial and Total correlation
iv) Linear and non Linear correlation. no correlation (0)

positive and negative

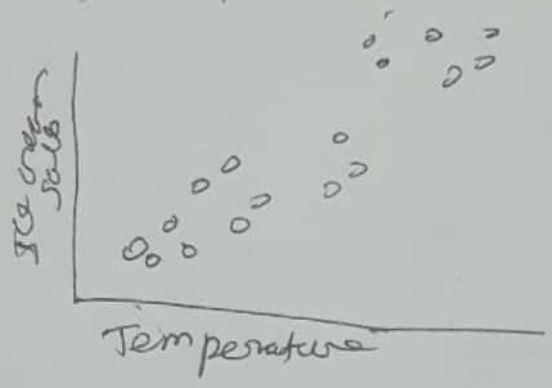
→ In positive correlation, increasing or decreasing the values of one variable necessarily leads to increasing (or decreasing) the values of another variables. → In the case of negative correlation increasing (or decreasing) the values of one variable certainly leads to decreasing (or increasing) the values of another variable.

Positive Correlation: two variables changes in the same direction

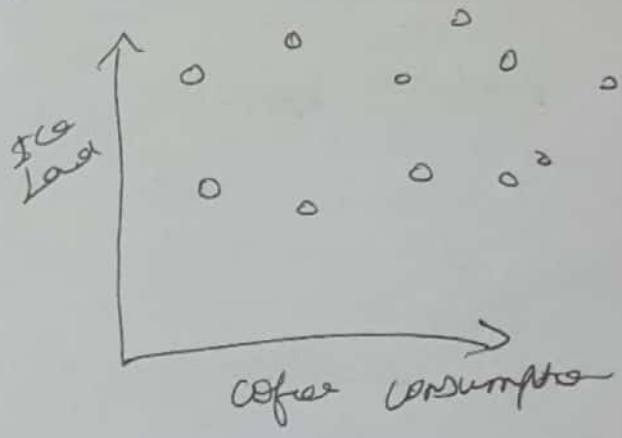
Negative correlation: two variables change in opposite direction

No Correlation
There is no association or relevant relationship between the two variable.

Positive correlation . eg.
1) height VS weight
2) Temperature VS Ice cream sales



Negative correlation



Amount of coffee and Ice Level has a correlation of Zero

Simple and multiple correlation
Simple: When we studs the relationship between only two variable, It is called simple correlat
eg: price & Demand
multiple: When we studs the relationship between more than two variable, it is called multiple correlation eg price / Demand & supply of a produc

Partial and total Correlation
When we studs the relationship between only two variables by eliminating other variable is called Partial correlation.
eg: studying price & Demand Eliminating supply of a Product
In total correlation: all the facts are taken into account.

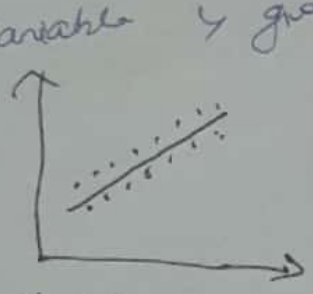
Linear and non Linear correlation.

Linear - When the ratio of changes between two variables is uniform it is called Linear

non Linear - If the amount of change in one variable is not the same like the other, it is called Non Linear correlation.

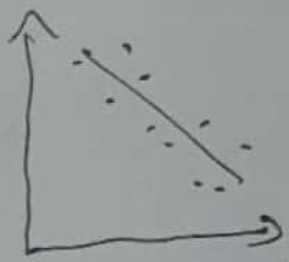
Scatter plots

A scatter diagram is a diagram that shows the values of two variables X and Y, along with the way in which these two variables relate to each other. The values of variable x are given along the horizontal axis and with the values of variable y given on the vertical axes



Strong positive correlation

Weak positive correlation



Strong negative correlation.

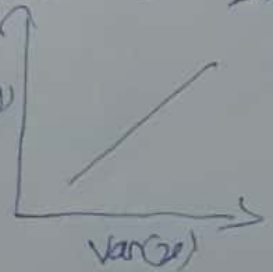
Weak negative correlation

NO correlation

Linear Relationship

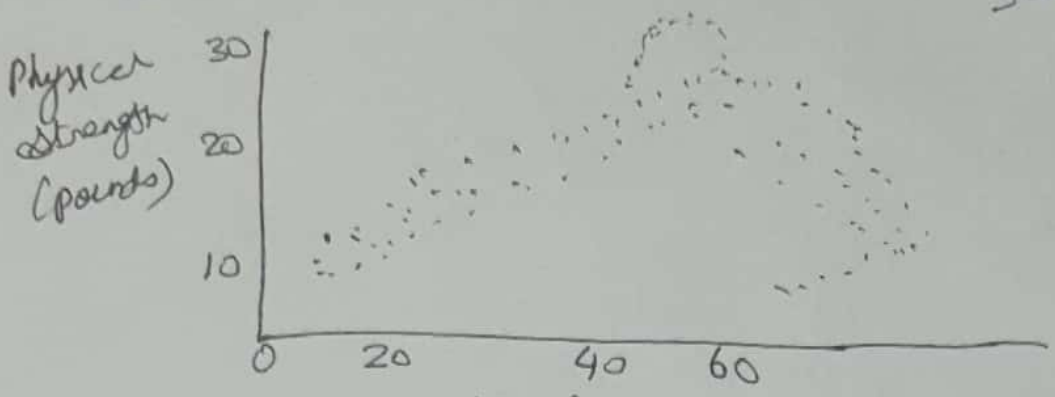
If the dot cluster approximates a straight line and it reflects a linear relationship between x & y

x	y
1	10
2	20
3	30



Curvilinear Relationship

If the data cluster approximates a bent or curved line and therfore reflects a curvilinear relationship



Correlation Coefficient for quantitative Data.

⇒ Quantitative or Numerical Data can be Discrete or Continous

⇒ In Statistics, correlation coefficients are a quantitative assessment that measures both the direction and strength of the tendency to vary together

→ correlation coefficient can be Calculated using "pearson correlation coefficient" formula. both variable are quantitative and normally distributed with no outliers.

⇒ Pearson's correlation Coefficient is represented by the Greek letter 'rho' (ρ) → population parameter
r → sample Statistic

⇒ Correlation coefficient is a single number that measure both the strength and direction of the Linear relationship between two continous variables value can range from −1 to +1

Strength:

Greater the absolute value of the pearson correlation coefficient → stronger relationship

Direction:

The sign of the pearson correlation coefficient represents the direction of the relationship

Positive coefficients:

→ It indicate that when the value of one variable increases, the value of the other variable also tends to increase.

→ positive relationships produce an upward slope on a scatterplot.

Negative coefficients:

It represent cases when the value of one variable increases, the value of the other variable tends to decrease.

negative relationships produce a downward slope.

Pearson correlation coefficient Formula:

$$r = \frac{n(\Sigma xy) - (\Sigma x)(\Sigma y)}{\sqrt{[n\Sigma x^2 - (\Sigma x)^2][n\Sigma y^2 - (\Sigma y)^2]}}$$

When n = Quantity of Information

Σx = total of the first variable value

Σy = total of the second variable value

Σxy = sum of the product of first & second value

Σx^2 = sum of the squares of the first value

Σy^2 = sum of the squares of the second value

Correlation coefficient measures the relationship between two variables.

1 → there is a perfect Linear relationship between variable

0 → there is a NO Linear relationship between variable.

−1 → there is a perfect negative Linear relationship between variables.

Find the Karl pearsons correlation coefficient

x	x^2	y	y^2	xy
6	36	9	81	54
2	4	11	121	22
10	100	5	25	50
4	16	8	64	32
8	64	7	49	56
30	220	40	340	214

$x = 6, 2, 10, 4, 8$

$y = 9, 11, 5, 8, 7$

$$r = \frac{N\Sigma xy - (\Sigma x)(\Sigma y)}{\sqrt{N\Sigma x^2 - (\Sigma x)^2}\sqrt{N\Sigma y^2 - (\Sigma y)^2}}$$

$$= \frac{5 \times 214 - 30 \times 40}{\sqrt{5 \times 220 - (30)^2}\sqrt{5 \times 340 - (40)^2}} = \frac{1070 - 1200}{\sqrt{1100 - 900}\sqrt{1700 - 1600}}$$

$$= \frac{-310}{\sqrt{200}\sqrt{100}} = \frac{-130}{14.14(10)} = \frac{-130}{141.4}$$

$$= -0.919$$

3) Computational formula for correlation coefficient

The four types of correlation coefficient are given by

i) Pearson correlation coefficient
ii) Linear correlation coefficient
iii) Sample correlation coefficient
iv) Population correlation coefficient

⇒ correlation shows its relation between two variables correlation coefficient shows the measure of correlation to compare two datasets, we can use the correlation formulas.

Pearson correlation coefficient formula.

The common formula is the pearson correlation coefficient used for Linear dependency between that dataset. the value of the coefficient lies between -1 to +1. when the coefficient goes down to zero. the data is considered as not related

If we get the value of +1 → then the data are positively correlated

-1 → Negatively correlated.

$$r = \frac{n\sum(xy) - (\sum x)(\sum y)}{\sqrt{[n\sum x^2 - (\sum x^2)][n\sum y^2 - (\sum y^2)]}}$$

Where n = Quantity of Information

$\sum x$ → total of the first variable value

$\sum y$ → total of the second variable value

$\sum xy$ → sum of the product of first & second value

$\sum x^2$ - sum of the squares of the first value

$\sum y^2$ → sum of the squares second value.

Linear correlation coefficient formula.

The formula for the Linear correlation coefficient is given by.

$$r_{xy} = \frac{n\sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} x_i \sum_{i=1}^{n} y_i}{\sqrt{n\sum_{i=1}^{n} x_i^2 - (\sum_{i=1}^{n} x_i)^2}\sqrt{n\sum_{i=1}^{n} y_i^2 - (\sum_{i=1}^{n} y_i)^2}}$$

Sample correlation coefficient formula.

The formula is given by

$$r_{xy} = \frac{S_{xy}}{S_x S_y}$$ where S_x & S_y are sample standard deviations

S_{xy} - is the sample covariance.

Population correlation coefficient formula

The population correlation coefficient uses σ_x and σ_y as the population standard deviations and σ_{xy} as the population covariance.

$$\rho_{xy} = \sigma_{xy} / \sigma_x \sigma_y$$

Rank correlation: Charles edward spearman developed the method of Rank correlation coefficient

This method can be applied where the population is not known

$$R = 1 - 6\Sigma D^2 / N^3 - N$$

eg:3 two judges in the beauty contest rank the 12 entries as follows. (1)

x: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12

y: 12, 9, 6, 10, 3, 5, 4, 7, 8, 2, 11, 1

Compute the rank correlation between the two coefficient.

Soln: Calculation of Rank correlation

Rank x	Rank y	$D = R(x) - R(y)$	D^2
1	12	-11	121
2	9	-7	49
3	6	-3	9
4	10	-6	36
5	3	2	4
6	5	-1	1
7	4	3	9
8	7	1	1
9	8	1	1
10	2	8	64
11	11	0	0
12	1	1	121
N = 12			$\Sigma D^2 = 416$

$$R = 1 - \frac{6\Sigma D^2}{N^3 - N}$$

$$= 1 - \frac{6 \times 416}{12^3 - 12}$$

$$= 1 - \frac{2496}{1728 - 12}$$

$$= 1 - \frac{2496}{1716}$$

$$= 1 - 1.455$$

$$= -0.455$$

$$R = -0.455$$

(8)

Regression.

→ Regression is defined as a statistical method that helps us to analyze and understand the relationship between two or more variables of interest.

→ In regression, we normally have one dependent variable and more or more independent variables.

Regression Analysis:

Regression Analysis is a branch of statistics that is widely used in almost all the scientific disciplines.

Correlation	Regression
Relationship between two or more variables	It is a mathematical measure showing the average relationship between variables.
x and y are random variable.	x is a Random variable y is a fixed variable
It gives limited information regarding the relationship between the variables.	It is used for the prediction of one value, in relationship to other given value.
The range of relationship between the variables.	It Regression value is absolute figure.
It studies linear relationship between the variables.	It studies linear and non linear relationship between variables
If the coefficient of correlation is positive then the two variables are positively correlated and vice versa.	The regression are the one explains that the decrease in one variable is associated with the increase in another variable.

Dependent variable: This is the variable that we are trying to understand or forecast

Independent Variable:

→ These are factors that influence the analysis of target variable and provide us with information regarding the relationship of the variable with the target variable.

Application of Regression Analysis:

→ The regression analysis method of forecasting generally involves five basic applications.

i) Predictive Analysis
ii) operation Efficiency
iii) Supporting decision
iv) Correcting errors
v) New Insights.

Regression Line:

→ The Regression Line is a straight line rather than a curved line because of the linear relationship between the two variables.

→ the regression line is often referred to as the least squares regression line.

→ If we take two variables x & y we have 2 regression lines.

Regression equation:

→ The algebraic equation of the two regression lines are called regression equation.

→ Regression of x on y $\boxed{x = a + by}$

→ the values of a and b can be calculated by solving the two normal equations.

$$\Sigma x = Na + b\Sigma y$$

$$\Sigma xy = a\Sigma y + b\Sigma y^2$$

Regression of y on x

$y = a + bx$

the values of a and b can be calculated by solving the two normal equations.

$\Sigma y = Na + b\Sigma x$ $\quad \Sigma xy = a\Sigma x + b\Sigma x^2$

Find regression line: x: 3, 5, 6, 8, 9, 11, y = 2, 3, 4, 6, 5, 10

x	$x = x - \bar{x}$	x^2	y	$y = y - \bar{y}$	y^2	xy
3	-4	16	2	-3	9	12
5	-2	4	3	-2	4	4
6	-1	1	4	-1	1	1
8	1	1	6	1	1	1
9	2	4	5	0	0	0
11	4	16	10	5	25	20
$\Sigma x = 42$	$\Sigma x = 0$	$\Sigma x^2 = 42$	$\Sigma y = 30$	$\Sigma y = 0$	$\Sigma y^2 = 40$	$\Sigma xy = 38$

$\bar{x} = \frac{\Sigma x}{n} = \frac{42}{6} = 7$ $\quad \boxed{\bar{x} = 7}$

$\bar{y} = \frac{\Sigma y}{n} = \frac{30}{6} = 5$ $\quad \boxed{\bar{y} = 5}$

Regression equation of x on y $(X - \bar{x}) = r\frac{\sigma x}{\sigma y}(y - \bar{y})$

$r \times \frac{\sigma x}{\sigma y} = \frac{\Sigma xy}{\Sigma y^2}$ $\quad \frac{\Sigma xy}{\Sigma y^2} = 0.95$

$(x - 7) = 0.95(y - 5)$

$x - 7 = 0.95y - 4.75$

$x = 0.95y - 4.75 + 7$

$\boxed{x = 0.95y + 2.25}$

Regression equation of y on x $y - \bar{y} = r\frac{\sigma y}{\sigma x}(x - \bar{x})$

$r\frac{\sigma y}{\sigma x} = \frac{\Sigma xy}{\Sigma x^2}$ $\quad \frac{\Sigma xy}{\Sigma x^2} = 0.904$

$y - 5 = 0.904(x - 7)$

$y = 0.904x - 6.328 + 5$

$\boxed{y = 0.904x - 1.328}$ (11)

3.7 Least Squares Regression Equation:

→ An Equation pinpoints the exact least squares regression line for any scatter plot.

Where ⇒ y represents the Predicted Value
⇒ x represents the known value
⇒ b & a represents the Calculated from the original correlation analysis.

→ Using linear regression we can find the line the best "fits" our data. the line is known as the least squares regression line and it can be used to help us understand the relationship between weight and height. line of best fit is the formula for the written as $y = b + ax$

y → Predicted value of the response variable
b₀ → intercept
a → Regression coefficient
x → value of the predictor variable.

Notice how our data points are scattered closely around the line. That's because this least squares regression line is the best fitting line for our data

Fitting least squares Regression Lines.

Eg. Predictor value: 140, 155, 179, 192, 200, 212
Response value: 60, 62, 67, 70, 71, 72, 75

Find a Regression line using the Calculator also answer
a) for a person who weights 176 pounds, how tall would we expect them to be?
b) for a person who weights 150 pounds, how tall would we expect that to be?

Linear Regression equation:

$\hat{y} = 32.7830 + (0.2001) * x$ the calculator automatically finds the Least Squares

Regression Line:

$y = 32.7830 + 0.2001x$

How to use the Least Squares Regression Line
Using the Least Squares regression line, we can answer the questions we

a) to answers this, we can simply substitute $x = 170$ into our regression line for x and solve for y

$y = 32.7830 + 0.2001(170) = 66.8$ inches

b) to answer this we can substitute for $x = 150$ into our regression line for x and solve for y.

Standard Error of estimate.

→ is the measure of variation of an observation made around the computed regression line. Simply it is used to check the accuracy of predictions made with the regression line.

→ the regression equations helps us to predict the values of y for values of x or

→ the value of x for values of y

→ the deviation of can not from the regression line is symbolized as $y - y_c$. Thus the square root of the mean of the squared deviation.

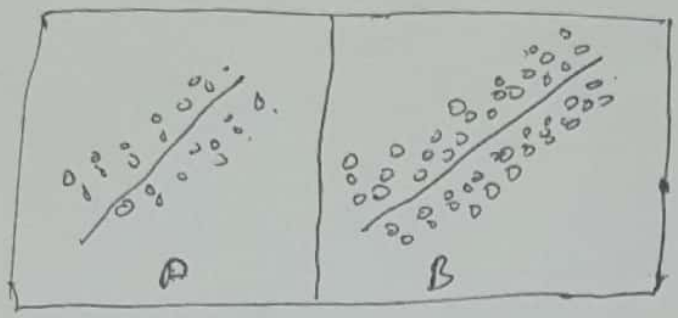
$$S_y = \sqrt{\frac{\Sigma(y-y_c)^2}{N}} \qquad S_x = \sqrt{\frac{\Sigma(x-x_c)^2}{N}}$$ Similarly

→ In four $(x - x_c)$ and $(y - y_c)$ represent the unexplained variation in x and y terms.

$$S_y = \sqrt{\frac{\text{unexplained variation in } y}{N}} \qquad S_x = \sqrt{\frac{\text{unexplained variation in } x}{N}}$$

→ The computation of standard error of estimate by the above formula is quite tedious. More convenient formula.

$$\sigma_x = \sqrt{\frac{\Sigma x_2^2 - a\Sigma x - b\Sigma xy}{N}} \qquad \sigma_y = \sqrt{\frac{\Sigma y^2 - a\Sigma y - b\Sigma xy}{N}}$$



Regression differing in accuracy of prediction

→ You can see that group A, the points are closer to the line than they are in Graph B,

→ The prediction in Graph A are more accurate than in Graph B.

→ the standard error of the estimate is a measure of the accuracy of predictions.

→ the regression line is the line that minimizes the sum of squared deviations of predictions (also called the sum of squares error)

$$\sigma_{est} = \sqrt{\frac{\Sigma (y-y')^2}{N}}$$

σ_{est} is the standard error of the estimate
y is an actual score
y' is a predicted score
N is the number pair of scores.
$(y-y')$ - is the error of prediction

$$\sigma_{est} = \sqrt{\frac{\Sigma (y-y')^2}{N}} \text{ for population}$$

Similar formula are used when the standard error of the estimate is computed from a sample rather than a population

$$s_{est} = \sqrt{\frac{\Sigma (y-y')^2}{N-2}}$$

Interpretation of r^2 multiple regression

R-squared:

R-squared (R^2 or the coefficient of determination) is the statistical measure in a regression model that determines the proportion of variance in the dependent variable that can be explained by the independent variable. In other words, r-squared shows how well the data fit the regression model

0-100%

(error)

Regression statistics

Multiple R	0.939440536
R square	0.88254859
Adjusted R square	0.87981709
Standard Error	207.205402

0% = Line no represent

100 = all show

larger (R²) > line fit

have person.

R-squared can take any value between 0 and 1

In addition, it doesnot indicate the correctness of the regression model. therefore the user should always draw conclusions about the model by analyzing r-squared together with the other variable in statical model.

how to calculate R-squared.

the formula for calculating R-squared is

$$R\text{-squared} = \frac{SS_{regression}}{SS_{total}}$$

SS regression is the sum of square due to regression (Explained sum of squares)

SS total is the total sum of squares.

Multiple Regression:

It is the statistical techniques that can be used to analyze the relationship between a single dependent variable and several independent variables.

Independent variable ⇒ If value is independent of other variables called.

Dependent variable ⇒ If its value depends on changes in the independent variable.

Multiple Regression analysis is used whenever we wish to model the relationship between one response variable and more than one regressor variables.

Multiple Linear Regression:

formula for multiple Linear Regression

$$y = b_0 + b_1 * x_1 + b_2 * x_2 + \ldots + b_k * x_k$$

The variables in this equation are

y is the predicted or expected value of the dependent variable.

x_1, x_2, x_p are three independent or predictor variables.

b_0 is the value of y when all the independent variables

$b_1, b_2 \& b_k$ are the estimated regression coefficients

each regression coefficient represents the change in y relative to a one-unit change in the respective independent variable.

Public health:

If we want predict the future spread of these illness based upon current known infectious multiple Independent variables can affect number of future infections including population size, population density, air temperature, asymptomatic carriers

Difference between Simple Linear and multiple Linie Regression.

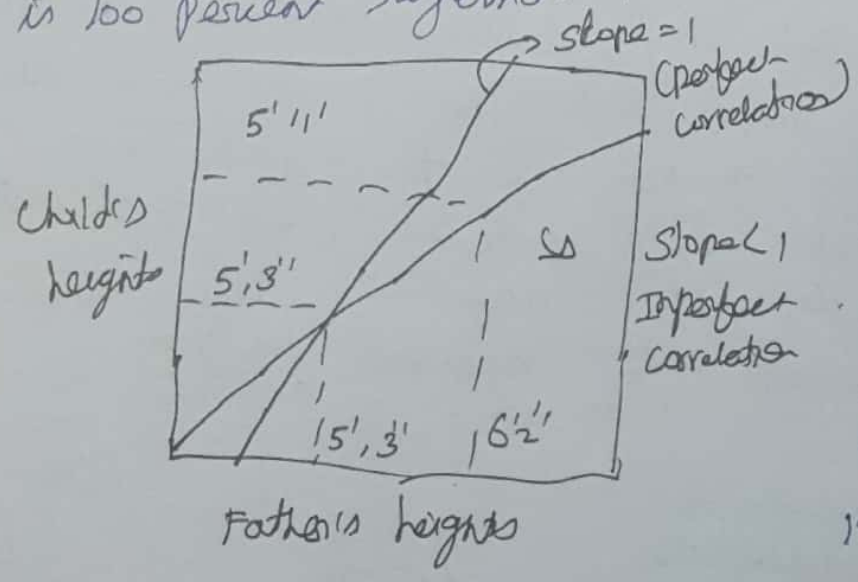
Simple Linear regression has only one x and one y variable. Multiple Linear regression has one y and two more x variable.

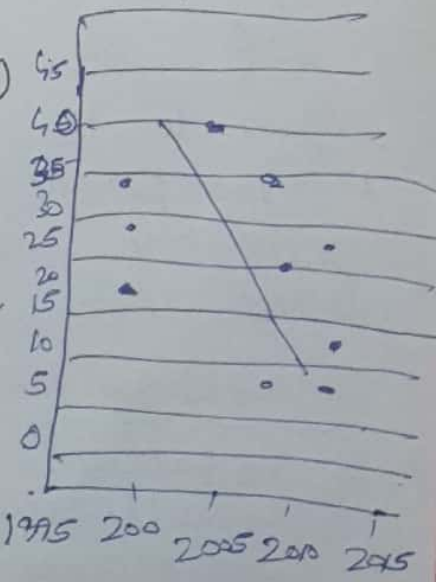
Advantages of multiple regression.

→ more accurate
→ Price understanding of the association of each individual factor with the outcome.

Regression to the Mean.

→ RTM is a statistical phenomenon that can make natural variation in repeated data look like real change. It happens when unusually large or small measurements tend to be followed by measurements that are closer to the mean
→ mathematically, the strength of the "Regression" effect is dependant on whether or not all of the random variable are drawn from the same distribution.
→ If data has perfect correlation, it will race regression to the mean. with an r of zero, there is 100 percent regression to the mean.





Regression analysis
Linear amount (dollars)

The fallacy:

The general statistical rule is that whenever the correlation between two variables is imperfect there will be regression to the mean.

Importance of Regression to the mean:

It is important to minimize instances of bad judgement and address the weak spot in reasoning

Regression to the mean fallacy:

The regression (or) regressive fallacy is an informal fallacy. It assumes that something returned to normal because of cognitive actions that fails to account for natural fluctuations

Why Regression to the mean happen:

Regression to the mean usually happens because of sampling error. A good sampling technique is to randomly sample from the population.

formula for the percent of regression to the mean

Percent of Regression to the mean = $100(1-r)$

⇒ correlation coefficient & regression to Mean:

▷ r - correlation coefficient If $r=1$ → perfect correlation then $1-1=0$ and the regression to the mean is zero

▷ If your data has perfect correlation, it will regress to the mean. with an r of zero. then it is 100 percent regression to the mean.

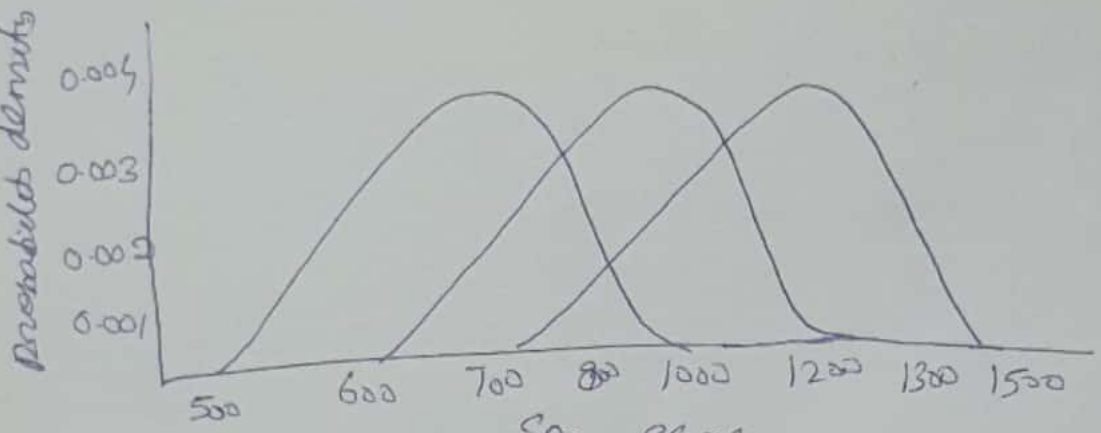
★ In other words, data with an r of zero will always regress to the mean.

Terminologies related to regression analysis.

- → Outliers
- → multicollinearity
- → Heteroscedasticity
- → underfitting and overfitting.

Normal distributions Important:

All kinds of variables in natural and social sciences are normally or approximately normally distributed height, birth weight, Reading ability Job satisfaction or Entrance scores.



Normal distributions with different means.

(1) Outliers:

An unusual high or low piece of data shown by outlier.

Start your data:-

To identify outlier, to see your data, which allows you to see as unusual data points within your information.

for eg) 3, 6, 7, 10 and 54 arranged in ascending order you can see 54 is blue larger than the rest of the data point.

2. Graph your data:-

To find outliers, graph your data in scatter plots or histograms.

useful for visualizing outlier, because one dot is far away from the other dots.

Histogram:

It displays data in groups called "bins"

most of your data points, are on the Right side of the graph, and one bin of data is on the left side of the graph.

Left bin → outlier.

3. Calculate the Z-score:

To Calculate Z-score, substract the mean from the raw measurment and divide it by the standard deviation

$$z = \frac{x - \mu}{\sigma}$$

x - raw measurment

μ - mean

σ - Standard deviation.

Thank you :)

UNIT IV PYTHON Libraries for Data Wrangling

Basic of Numpy arrays - aggregations - computations on arrays - comparisons, masks, boolean logic - fancy indexing - structural arrays - Data manipulation with Pandas - data indexing and selection - operating on data - missing data - Hierarchical indexing - combining datasets - aggregation and grouping - Pivot tables.

Basic of Numpy Arrays:

Numpy - stands for Numerical python.

→ It is a python Library used for working with an array. Numpy is a powerful N-dimensional array object & has functions for working in domain of linear algebra, fourier transform & matrices.

→ In python lists serve the purpose of arrays but lists are slow to process.

→ Numpy provide an array object called as ndarray which is 50x faster than traditional python lists.

Why is numpy faster than lists?

→ Numpy arrays are stored at one continuous place in memory. Unlike lists, so process can access & manipulate them very efficiently.

→ Numpy Library is partially written in C or C++ & partically written in python.

→ Data manipulation in python is done with Numpy array manipulation.

→ data manipulation includes access data & subarrays split, reshape and join the arrays.

Few categories of basic array manipulations.

① Attributes of arrays: Determining the size shape, memory consumption & data type of array.

② Indexing of arrays: Getting & setting the values of individual array elements.

③ Slicing of arrays: Creating & deleting smaller subarrays within a larger array

④ Reshaping of arrays: Changing the shape of a given array.

⑤ Joining & splitting of arrays: Combining multiple arrays into one, and splitting one array into many.

1. Numpy Array Attributes:
Determining the size, shape, dimensions, memory consumption & data types of arrays.
Each array has the following attributes.

1. ndim - number of dimensions eg- one dimension, 2D, 3D
2. shape - size of each dimension eg (3x3) or (2x3) or (3x3x5)
3. size - total size of the array ie no of elements in the array
4. dtype - data type of the array
5. itemsize - size of each item in bytes.
6. nbytes - total size of the array in bytes.

eg: Sample python program.

```
import numpy as np
x1 = np.random.randint(10, size=6)  # One dimensional array
x2 = np.random.randint(10, size=(3,4))  # create 3x4 2D Array with random values
x3 = np.random.randint(10, size=(3,4,5))  # create 3D array with in the range 10
print("x3 ndim:", x3.ndim)
print("x3 shape:", x3.shape)
print("x3 size:", x3.size)
print("x3 dtype:", x3.dtype)
print("x3 itemsize:", x3.itemsize, "in bytes")
print("x3 total size:", x3.nbytes, "in bytes")
```

Output
x3 ndim: 3
x3 shape: (3,4,5)
x3 size: 60
x3 dtype: int64
x3 itemsize: 8 in bytes
x3 total size: 480 in bytes

II: Array Indexing:
→ Accessing single element.
→ the indexing specifies the desired index in square bracket.
→ the indexes in Numpy arrays starts with '0'
→ negative indexes are used to index from the end of the array.
→ Creating & getting the value of array element indexing is used.

ex Creating a values:

```
import numpy as np
a1 = np.array([1,2,3,4]) # Create 1D array [1,2,3,4]
a11 = np.array([[1,2,3,4],[6,7,8,9,10]])
print(a1[0])   # print the first item in a array   output 1
print(a11[0,1]) # print the item on 1st row & 2nd column   2
print(a1[-1]) # print the last item in a1 array   4
print(a11[1,-1]) # print the last item in a11 array   10
```

ex Setting a value:

```
a1[0]=5   # modify (or) set the first item in a1 array as 5
a11[0,1]=10 # set the item in 1st row & 2nd column as 10
a1[-1]=20 # set to last item in a1 array as 20
```

output
print(a1[0]) 5
print(a11[0,1]) 10
print(a1[-1]) 20

III Array Slicing:
Accessing subarray.

Syntax arrayname[start : stop : step]
Start - is assumed as '0' if it is unspecified
Stop - is assumed as 'size of dimension' if it is unspecified.
Step - is assumed as '1' if it is unspecified.
Note: If the step value is -ve, default values of start and stop are swapped.
eg: print(a[::-1]) # print the values of array a in reverse order

Example

```
import numpy as np
a = np.array([1,2,3,4,5])  # create 1D array
b = np.array([[1,3,4,5],[5,6,7,8,9],[9,10,11,12]])  # create 2D array
print(a[:3])  # print the first 3 items of array a starts with index 0 i.e. [1,2,3]
```

O/P => [1,2,3]

```
print(a[2:7]
```

O/P => [3,4,5]

```
print(a[1:3])
```

O/P => [2,3]

```
print(b[:2, :3])  # print the items in first 2 rows & first 3 columns of array b
```

O/P => [[1,2,3],[5,6,7]]

Fancy Indexing:

Access and modify the portions of arrays by using simple indices, slices, Boolean mask then are arrays indexing.

Fancy indexing is like a simple indexing but we pass the arrays of indices in place of single scalars. this allows us to very quickly access and modify complicated subsets of an array values.

Exploring Fancy Indexing:

It means passing an array of indices to access multiple array element at once.

```
In [1]: import numpy as np
        rand = np.random.RandomState(42)
        x = rand.randint(100, size=10)
        print(x)
```

[51, 92, 14, 71, 60, 20, 82, 86, 74, 74]

Suppose we want to access three different elements

```
In [2]: [x[3], x[7], x[2]]
```

Out[2]: [71, 86, 14]

Alternatively we Pass a single list or array of indices to obtain the same result.

```
In [3]: ind = [3, 7, 4]
        x[ind]
Out[3]: array([7, 86, 60])
```

With fancy indexing. The shape of the result reflects the shape of the index array.

```
In [4]: ind = np.array([[3, 7], [4, 5]])
        x[ind]
Out[4]: array([[7, 86], [60, 20]])
```

Structured Array: Array with some compound data types. Here the structure provides efficient storage for compound heterogenous data.

Name (string)	Age int 32	height float 64
Alice	23	173.4
Bob	35	165.5
Anto	18	175.8
Duke	28	179.4

```
dt = np.dtype([('Name', 'U10'), ('Age', 'i4'),
               ('height', 'f8')])
data = np.array([('Alice', 23, 178.4), ('Bob', 35, 165.5),
                 ('Anto', 18, 175.8), ('Duke', 28, 179.4)])
print(data.dtype) → 'name' 'U10' 'Age' 'i4' 'height' 'f8'
without dtype => print data
print(data['Name'])
print(np.sort(data, order='Age'))
data = np.zeros(4, dtype={'name': ('Name', 'age', 'height'),
                          'format': ('U10', 'i4', 'f8')})
```

Create a Structured Array with fields such as employee, Qualification, designation and store then printed details in the structured array normal order as well as sorted orders.

Name (String)	Qualification	Salary	Designation
Jeno	BE	20000	Employee
Anu	BE	30000	Assistant
Safrin	ME	50000	Manager
Sabi	MSE	25000	Employee 2
Radha	ME	30000	Employee 3

```
dt = np.dtype([('Name', 'U10'), ('Salary', 'f8'),
               ('Qualification', 'U10'), ('Designation', U10)])
data = np.array([('Jeno', 20000, BE, Employee), ('Anu', 30000, BE,
        Assistant), ('Safrin', 50000, Manager), ('Sabi', 25000, MSC,
        Employee 2), ('Radha', 30000, ME, Employee 3)])
print (data)
```

The Pandas Index Object:

We have seen both the series and Dataframe object contain explicit index → used to reference & modify data.

Indexed Object → contain interesting data structure itself

It may be either Immutable array ordered.

ordered set → Technically multiset, Index object may contain repeated values.

Some operations available on Index object.

eg) Lets construct an index from a list of integers

```
In [30]: ind = pd.Index([2,3,5,7,11])
Out [30]: Int 64 Index [2,3,5,7,11], dtype='int64')
```

Index as Immutable Array:

Index object in many ways operates like an array. eg) we can use std. Python Indexing notation to retrieve the values or slices.

```
In [31]: ind[1]
Out [31]: 3
In [32]: ind[::2]
Out [32]: int64 Index ([2,5,11], dtype='int64')
```

Index Object also have many of the attributes familiar from Numpy Arrays.

In [33]: print (ind.size, ind.shape, ind.ndim, ind.dtype)

5 (5,) 1 int 64

Difference between Index Object and Numpy arrays

→ Index are immutable

→ so it cannot be modified in a normal way

In [34]: ind[1]=0

Type Error: Index doesnot support mutable operations.

Index as ordered set:

→ Pandas Object are designed to facilitate operations such as Join across datasets. It depends on many aspect of set arithmetic.

→ Index object are used in python built in set data structure. Such as unions, intersections, differences and other combinations.

In [35]: ind A = pd.Index([1, 3, 5, 7, 9])

ind B = pd.Index ([2, 3, 5, 7, 11])

In [36]: ind A & ind B #intersection

Out [36] = Int 64 Index ([3, 5, 7], dtype = 'int 64')

In [37]: ind A | ind B #union

Out [37]: Int 64 Index ([1, 2, 3, 5, 7, 9, 11], dtype='int64')

In [38]: ind A ^ ind B #symmetric difference

Out [38] = Int 64 Index ([1, 2, 9, 11], dtype = 'int 64')

These operations may also be accessed via Object method Eg... ind A. Intersection (ind B)

Data Indexing and selection.

Data selection in Series:

1) Series Object act like ↔ one dimensional Numpy Array / Standard Python Dictionary

11) Series object Provides Dictionary style Data selection mechanism

1) Series as Dictionary:

Like a dictionary Series object provides a mapping from a collection of keys to a collection of values.

```
In[]: import pandas as pd
data = pd.Series([0.25, 0.5, 0.75, 1.0],
                 index = ['a', 'b', 'c', 'd'])

data
Out[]: a  0.25
       b  0.5
       c  0.75
       d  1.0
       dtype: float64
```

Var [Start Index : end Index]

Value Included — Value Excluded

::2 → Add index +2

```
In[2]: data['b']
Out[2]: 0.5
```

→ We can also use dictionary like Python expressions and methods to examine the keys/indices and values.

```
In[3]: 'a' in data
Out[3]: True
In[4]: data.keys()
Out[4]: Index(['a', 'b', 'c', 'd'], dtype='object')
In[5]: list(data.items())
Out[5]: [('a', 0.25), ('b', 0.5), ('c', 0.75), ('d', 1.0)]
```

Series Objects:

It can be modified with a dictionary like syntax. Dictionary is extended by new key in the same way series can be extended by assigning a new index value.

```
In[6]: data['e'] = 1.25
       data
Out[6]: a  0.25
        b  0.50
        c  0.75
        d  1.00
        e  1.25
        data: float64
```

iii) Series as one dimensional Array:

Series provides Array style item selection mechanisms i.e) * slices * masking
* fancy Indexing

In [7]: # slicing by Explicit Index
data ['a' : 'c']

Out [7]: a 0.95
b 0.50
c 0.75
datatype : float 64

In [8]: # slicing by Implicit integer index
data [0:2]

Out [8]: a 025 datatype : float 64
b 050

In [9]: # masking
data [(data > 0.3) & (data < 0.8)]

Out [9]: b 050
c 075 dtype : float 64

In [10]: # fancy Indexing
data [['a', 'e']]

Out [10]: a 0.25
e 1.25 dtype : float 64

When you are slicing with an explicit index (data ['a' : 'c']), the final index is included in the slice.

When you are slicing with an Implicit index data [0:2], the final index is excluded from the slice.

Indexers, loc, iloc and ix

Loc = location & row Attributes It return one or more specified rows.

While a indexing operator data [] will be use explicit index.

While a slicing operator like data [1:3] will use implicit index.

In Python code → "Explicit is better than Implicit"
the explicit nature loc and iloc make very useful in maintaining clean Readable code.

Data Selection in DataFrame:
Data frame acts like a two dimensional or structured Arrays or in other way like a dictionary.

i) data frame as a dictionary:
It related series acts [Let's consider the eg]

```
In [8]: area = pd.Series({'California': 4329, 'India':
        : 628653, 'New York': 1597, 'Florida': 17032})
pop = pd.Series({'California': 38332521, 'India': 2644893,
      'newyork': 1965127, 'Florida': 19552868})
data = pd.DataFrame({'area': area, 'pop': pop})
data
```

Out[18]:

	area	pop
California	4329	38332521
Florida	17032	19552868
India	628653	2644893
newyork	4597	1965127.

the individual series that makes up the columns of the data frame can be accessed via dictionary style indexing of the column name.

Name : area, dtype : int 64

equivalently we can use attribute style access with column name.

```
In [20] data.area
```

Out [20]:

California	42396
Florida	13032
India	628653
Newyork	16127

Name : area, dtype : int 64

→ Attribute style column access have the same object as the dictionary style access.

In [2]: data.area is data["area"]

Out [2]: True

→ Attribute style access is not possible, when the column names are not strings, column name conflicts with the method of the data frame.

In [2]: data.pop is data['pop']

Out [22]: False.

Like series object; Dictionary style syntax add a new column

In [23] * data['density'] = data['pop'] / data['area'] data

Out [23]:

	area	pop	density
California	433987	3833252	904/3
Florida	17032	19552650	114.804
India	62650	12882135	85.8376
New York	141297	19651 27	139.0187

ii) Dataframe As two-dimensional Arrays

In [24]: data.values

Out [25]: Array ([[4.23 3.83 9.05],
[1.70 1.955 1.14],
[1.49 1.28 8.58],
[1.41 1.96 1.39]])

Arrays like Observations on Data frame:

Transpose for data frame.

↳ swap rows and columns

In [25]: data.T

Out [25]:

	California	Florida	India	Neyyar
Area	4.23	1.70	1.49	1.41
pop	3.83	1.95	1.28	1.96
density	9.05	1.14	8.58	1.39

Indexing of columns to access a row in a array.

In [26]: data.values[0]
Out[26]: array([423, 3.83, 9.04])

Passing a single 'index' to a Dataframe accesses a column.

In [27]: data['area']
Out[27]: California 423
Florida 170
India 149
Newyork 141
Name: area, dtype: int 64

Array style Indexing
uses loc, iloc, ix
iloc indexer uses implicit Python, like style indexing

In[28]: data.iloc[:3, :2]

	area	pop
California	423	383
Florida	170	195
India	149	128

In [29]: data.loc[:'India', :'pop']

	area	pop
California	423	383
Florida	170	195
India	149	128

ix indexer allows a hybrid of two approaches.

In [30]: data.ix[:3, :'pop']

	area	pop
California	423	383
Florida	170	195
India	149	128

Numpy style data access patterns also used in indexers. The indexer constructs masking and fancy indexing.

In [31]: data.loc[data.density > 1, ['pop', 'density']]

	pop	density
Florida	195	1.14
Newyork	196	1.39

Additional Indexing conventions:

In [33]: data ['florida', 'India']

Out [33]:	area	pop	density
Florida	170	195	1.14
India	143	128	0.8

Pandas while ordering refers to columns.
Slicing refers to rows.

→ Slicing refers to row by number rather than by Index.

In [34]: data [1:3]

Out [34] :	area	pop	density
Florida	170	195	1.14
India	143	128	0.8

Operations on Data In pandas.

Data manipulation with Pandas:

→ Pandas is a high level data manipulation tool developed by Wes McKinney. It is built on the numpy package and its key data structure is called its DataFrame.

→ DataFrame allows you to store and manipulate tabular data in rows of observations and columns of variables.

→ pandas is built on the top of the Numpy package, meaning a lot of structure of Numpy is used for replicated in pands. Data in pandas

Operations:

→ Numpy has ability to perform basic arithmetic operations (addition, substraction, Multiplication), and with more sophisticated operations trigonometric functions, Exponential, Logarithmic functions.

→ Pandas inherits the functionality from numpy and use "Computation on Numpy Arrays: Universal Functions"

Pandas Includes:

Binary operations: such as addition, multiplication
Pandas will automatically align index when
Parsing the object to
using operations. its ufunc.
like negation and trigonometric functions

Pandas preserve index and column labels.

1) Ufuncs: Index Preservation.

→ Pandas is designed to work with
Numpy: Numpy ufunc will work on
Pandas Series and Dataframe objects.

→ Lets us start by defining a simple
Series and Data frame on which to
demonstrate the

In[1]: import pandas as pd

→ import numpy as np

In[2]: rng = np.random.RandomState(42) random no generator.

ser = pd.Series(rng.randint(0, 10, 5)) create the 5 random ints

ser

Out[2]:
0 6
1 3
2 7
3 4
dtype: int64

Pandas is the libraries for data manipulation and analysis, usually it is the starting point for your data science tasks. It allows you to read/write data from to multiple source. Prior the messy data align your data reshape it, merge and join it with other data cleaning data process it, etc.

Missing Data:

→ Data can have missing value for a number of reason such as observations that were not recorded and data corruption. Handling missing data is important as many machine learning algorithms do not support data with missing values.

→ You can load the dataset as a Pandas Dataframe and print summaries of attributes.

```
# load and summarize
from pandas import read_csv
# load the dataset
dataset = read_csv('csv file name', header=None)
#summarize the dataset
print(dataset.describe())
```

In Python, specifically Pandas, Numpy and Scikit-learn, we mark missing values as NaN values. Then these values are ignored by operations like sum, count, etc.

Encoding Missingness:

The fillna() function is used to fill NA/NaN values using the specified method.

Syntax:

```
DataFrame.fillna(value=None, method=None, axis=None, inplace=False, limit=None, downcast=None, **kwargs)
```

Where

1. value: It is a value that is used to fill the null value
2. method: used to fill the null values.
3. axis: It takes int or string value for row/column.
4. inplace: It is true, it fill values at an empty place.
5. Limit: It is an Integer value that specifies what to be done as like ffill bfill

Hierarchical Indexing:

→ Is a method to creating structured group relationships in data.

→ A multi Index or hierarchical Index comes in when our DataFrame has more than two dimensions.

→ A multi Index adds at Least one more dimension to the data. A hierarchical index as the name suggest is ordering more than one item of their ranking.

To create DataFrame with Player rating of a few Players from the Fifa 19 dataset.

In [1]: import Pandas as Pd

In [2]: data = {'position': ['GK', 'GK', 'GK', 'DF', 'DF', 'DF', 'DF', 'MF', 'MF', 'MF', 'CF', 'CF', 'CF'],
'Name': ['De Gea', 'Courtois', 'Allis', 'Varane', 'Ram', 'Godin', 'Hazard', 'Kane', 'De', 'Ronald', 'Messi', 'Neymar'],
'Overall': [91, 88, 90, 80, 96, 93, 92, 95, 93, 94, 93, 92],
'Rank': ['1st', '3rd', '2nd', '3rd', '1st', '2nd', '3rd', '5th', '1st', '2nd', '3rd', '4th']}

In [3]: fifa 19 = Pd.DataFrame(data, columns = ['position', 'Name', 'Overall', 'Rank'])

In [4]: fifa19

Out [5]

	Position	Name	Overall	Rank
0	GK	De Gea	91	1st
1	GK	Courtois	88	3rd
2	GK	Allis	90	2nd
3	DF	Varane	80	3rd
4	DF	Ram	96	1st
5	DF	Godin	93	2nd
6	DF	Hazard	92	3rd
7	MF	Kane	93	5th
8	MF	De	95	1st
9	CF	Ronan	94	2nd
10	CF	Messi	93	3rd
11	CF	Neam	92	4th

Combining Dataset. concat and append.

→ Data comes from different Data sources.

→ Operation involve ⇒ combination of two different data set. are more Complicated.

→ Pandas includes functions and methods ⇒ that make sort of data wrangling fair and straightforward. Simple concatenation of Series and Data frame with pd. concat function. we begin with the standard import.

In[1]: import Pandas as pd
import numpy as np

In[2]: def make_df (cols, ind): '''quickly make a Dataframe '''
data = { c: [str(c) + str(i) for i in ind]
for c in cols }
return pd. DataFrame (data, ind)

#example Dataframe
make_df ('ABC', range (3))

Out[3]:

	A	B	C
	A_0	B_0	B_0
1	A_1	B_1	C_1
2	A_2	B_2	C_2

Recall: Concatenation of Numpy arrays.
Concatenation of Series & Dataframe objects similar to the Numpy Array, done by np. Concatenate function the contents of two or more arrays into a single Array

In [4] x = [1, 2, 3]
y = [4, 5, 6]
z = [7, 8, 9]
np. concatenate ([x, y, z])

out [4] array
([1, 2, 3, 4, 5, 6, 7, 8, 9])

In [5]: x = [1,2], [3,5]]

np. concatenate ([x,x], axis=1)

Out [5]: array([[1,2,1,2], [3,5,3,5]])

Simple Concatenation with pd. concat.

Pandas has a function, pd. concat(). Similar to np. concatenate but pandas contain several options with them.

Signature in pandas V0.18

pd.concat(objs, axis=0, join='outer', join_axes=None, ignore_index=False, keys=None, levels=None, names=None, verify_integrity=False, copy=True)

pd.concat() → It can be used for a simple concatenation of series or Dataframe objects.

np.concatenate() → It can be used for simple concatenation of arrays

In [6]: ser1 = pd.Series(['A','B','C'], index=[1,2,3])

ser2 = pd.Series(['D','E','F'], index=[4,5,6])

pd.concat([ser1, ser2])

Out [6]

1	A
2	B
3	C
4	D
5	E
6	F

dtype: object

Append() method:

Simply call df1.append(df2):

Append() method in pandas does not modify the original object. It creates a new object with the combined data.

Combining Data Set: Merge and Join:

Pandas provide high performance in memory join and merge operation

pd.merge() function

join() function

Categories: → one to one
→ many to one
→ many to many

Aggregation and Grouping

In large data set, the data are summarized or aggregated for efficiency.

aggregation → it will return single number for a large dataset, some aggregations → sum(), mean(), median(), max, min().

Simple Aggregation In Pandas:

For a Pandas series the aggregates returns single value

```
In[4]: rng = np.random.RandomState(42)
       ser = pd.Series(rng.rand(5)) ser
Out[4]: 0  0.3
        1  0.9
        2  0.7
        3  0.5
        4  0.2
        dtype: float64
In[5]: ser.sum()
Out[5]: 2.5
Out[6]: ser.mean()
Out[6]: 0.5
```

for a Dataframe, by default the aggregation can be taken in each column.

```
In[7]: df = pd.DataFrame({'A': rng.rand(5),
                          'B': rng.rand(5)})
df
Out[7]:
```

	A	B
0	0.1	0.01
1	0.05	0.9
2	0.08	0.8
3	0.6	0.2
4	0.1	0.1

```
In[8]: df.mean()
Out[8]:
```

A	0.45
B	0.4

dtype: float64

```
In[9]: df.mean(axis='columns')
Out[5]:
```

0	0.05
1	0.5
2	0.8
3	0.4
4	0.4

dtype: float64

Some of the built in Pandas Aggregations.

Aggregation	Description.
count()	Total number of items
first(), last()	First and last item
mean(), median()	Mean and Median
min(), max()	Minimum & maximum
std(), var()	Standard deviation and variance
mad()	Mean absolute deviation.
prod()	Product of all items
sum()	Sum of all items

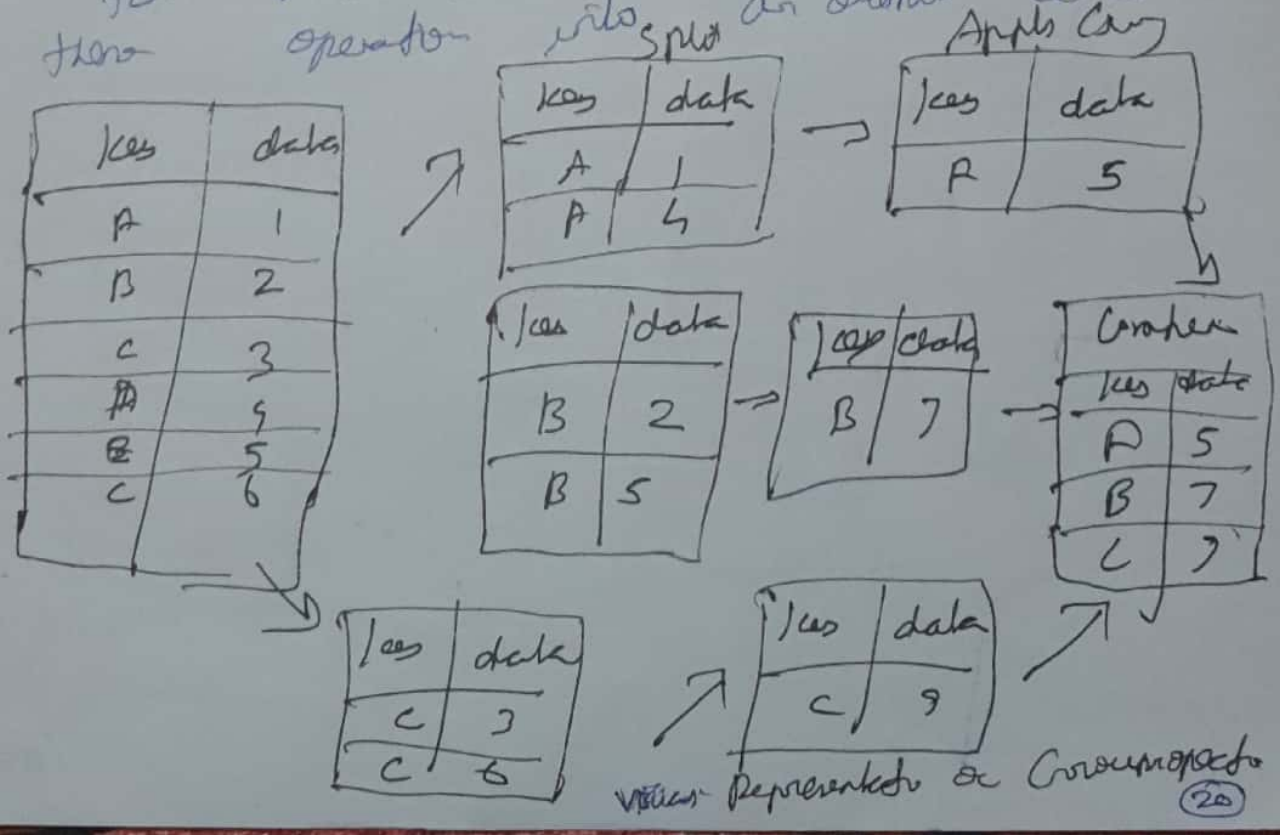
Groupby, Split, Apply, Combine

Groupby is accomplished by the steps.

Split step involve breaking up and grouping a data frame depending on the value of the specified key.

Apply step involve computing some function, usually an: 1) Aggregate 2) transformation 3) filtering within the individual groups.

Combine step merges the result of these operation into an output array.



Visual Representation of Groupby steps

Data wrangling:

→ Data wrangling is the process of transforming data from its original "raw" form into a more digestible format and organizing sets from various sources into a singular coherent whole for further processing.

→ Data wrangling is also called as data munging. The primary purpose of data wrangling can be described as getting data in coherent shape. In other words. It is making raw data usable.

Data wrangling processes:

i) Getting the data from the various source into one place.

ii) Piecing the data together according to the determined setting.

iii) Cleaning the data from the noise or errorneous, missing elements

Data wrangling is the process of cleaning & structuring and enriching raw data into a desired format for better decision making in less time.

Six iterative steps that make wrangling process

(1) Discovering:

Before you can dive deeply, you must better understand what is in your data. What will inform how you want to analyze it. How you wrangle customer data may be informed by where they are located, what they bought, or what promotions they received.

(2) Structuring:
→ This means organizing the data, which necessary because raw data comes in many different shapes and sizes.
→ A large column may turn into several rows for easier analysis. One column may become two. Movement of data is made for easier computation and analysis.

(3) Cleaning:
→ What happen when errors and outliers skew your data you clean the data.
→ What happens when State data is entered as AP or Andra pradesh or Arunachal Pradesh. you clean the data. Null values are changed & standard formatting implemented, ultimately increases data quality.

(4) Enriching: here you take stock in your data strategize about how other additional data might augument it. Questions asked during wrangling step might be: what new types of data can be derived from what I already have or what other information

(5) Validating: Validation rules are repetative programming sequences that verify data consistency, quality & security. eg of validation include ensuring uniform distribution of attributes that should be distributed normally (eg birth date) or confirming accuracy of fields through a check across data.

(6) Publishing: Analyze Prepare the wrangled data for use downstream, whether a particular use or software document any particular steps taken or logic to used wrangle data. understand the implementation of insight relies upon the care with which it can be accessed and utilized by others.

Pivot Tables:

→ A pivot table is an interactive way to quickly summarize huge amounts of data.

→ It can be used to analyze numerical data in detail and answer unanticipated questions about the data.

→ A pivot table is especially designed for querying large amount of data in many user friendly ways. In pandas pivot table() is used to calculate, aggregate and summarize your data. It is defined as a powerful tool that aggregates data with Calculations such as sum, count, average, Max & Min. It also allows to sort and filter data when the pivot table has been created.

Parameters:

data : A dataframe

values : It is an optional parameter & column to aggregate.

Index : It refer to the column, Groups & arrays.

Syntax:

Pandas.pivot_table(data, value = None, index = None, columns = None, aggfunc = 'mean', fill_value = None, margins = False, dropna = True, margins_name = 'All')

Data : Dataframe

values : Column to aggregate, optional

Index : Column, Groups, arrays, or list of the previous

Columns : Column, Groups, arrays or list of the previous.

aggfunc : function, list of functions, dict, default numpy.mean

fill_value [scalar, default None] : value to replace missing value with

margins [boolean, default, False] : Add all row/columns

dropna [boolean, default, True] : Do not include columns and all NaN

margins_name [string default 'All'] : Name of the row/column that will contain the total when margins is true.

Consider a data frame:

```
#create a simple data frame
#importing pandas as pd
import Pandas as pd
import numpy as np
#creating a dataframe.
df = pd.DataFrame({'A': ['Alex', 'Bays', 'Dunn', 'Taylor', 'Nicks'],
'B': ['master', 'Graduate', 'Graduate', 'Masters', 'Graduate'],
'C': [27, 23, 21, 23, 24]})
```

df

OUT:

	A	B	C
0	Alex	Masters	27
1	Bayes	Graduate	23
2	Dunit	Graduate	21
3	Taylor	Master	23
4	Nicks	Graduate	24

```
#Simplest Pivot table must have a dataframe
# and an index /ar of index
table = pd.pivot_table(df, index = ['A', 'B'])
table
```

OT:

A	B	C
Alex	Masters	27
Bayes	Graduate	23
Nicks	Graduate	24
Dunit	Graduate	21
Taylor	Master	23

Applications of pivot tables:

- summarizing data
- finding its average sales for each region for each product from a product data table.
- Listing unique values in an item of a table.
- creating a pivot report with sub-totals and
- Custom formats
- Filtering, sorting, drilling, down data in the reports.

Unit - V

Data Visualization

Importing Matplotlib - Line plots - Scatter plots - Visualizing errors - density and contour plots - Histograms Legends - colors - subplots - text and annotation - customization - three dimensional plotting Geographic data with Basemap - Visualization with seaborn.

Introduction : Matplotlib:

→ Matplotlib tool for visualization in Python Matplotlib is a multiplatform data visualization library built on Numpy arrays, and designed to work with the broader stack. Main important features is ability to play well with many operating systems and graphics backends.

→ The cross platform everything to everyone approach has been great strength of Matplotlib

5:1 Importing Matplotlib:

→ We use np shorthand for Numpy and the plt shorthand for pands.

→ we will use some standard shorthand for Matplotlib imports.

```
In [ ]: import matplotlib as mpl
        import matplotlib.pyplot as plt
```

meg offensive

Setting Styles

We will use the plt style directive to choose appropriate aesthetic style for our figures. we will set it and use the classic matplotlib style

```
In [ ]: plt.style.use('classic')
```

We will adjust the style as needed from the matplotlib version 1.5 ⇒ customizing matplotlib configurations and style sheets.

show() or No show()?

Plotting from a script

If you are using Matplotlib from within a script, the function plt.show() opens one or more interactive windows and displays your figure or figures.

```
# ---- file: myplot.py
import matplotlib.pyplot as plt import numpy as np
x = np.linspace(0, 10, 100)
plt.plot(x, np.sin(x))
plt.plot(x, np.cos(x))
plt.show()
```

When you are run the script, what will result is a window opening with figure displayed.

Plotting from an IPython shell:

It can be very convenient to use Matplotlib interactively within an IPython shell. IPython is built to work with Matplotlib if you specify matplotlib mode ⇒ to enable this mode you can use %matplotlib magic command after starting ipython.

```
In [1]: %matplotlib
Using matplotlib backend: TkAgg
In [2]: import matplotlib.pyplot as plt
```

In the point- and plot command will cause a figure window to be open and further commands can be run to update the plot. Some changes (such as modifying properties of lines that are already drawn) will not draw automatically. To force an update, use plt.draw(). Using plt.show() in matplotlib lib mode is not required.

Plotting from an IPython Notebook

→ IPython interactive data Notebook is a browser-based computational environment. Combine descriptive code, graphics, HTML elements, and much more into a single executable document. Plotting interactively within an IPython Notebook can be done with the %matplotlib command.

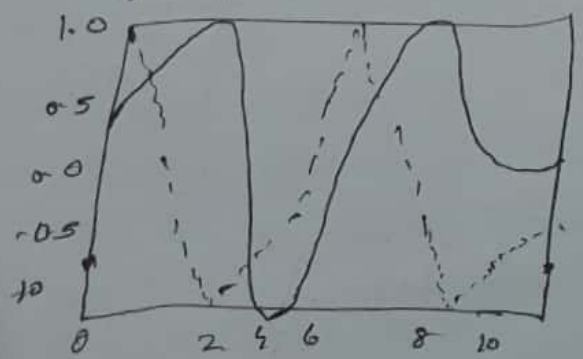
→ In IPython Notebook, two possible options of embedding graphics.

→ %matplotlib Notebook will lead to interactive plot embedded within the Notebook

→ %matplotlib inline will lead to static images of your plot embedded in the Notebook

→ In[3]: %matplotlib inline

→ After you run this command (it only needs to be run once per kernel/session), every plot you create will be embedded as a PNG image in the notebook, right after the code cell that generated it.




```
In [5]: import numpy as np
x = np.linspace(0, 10, 100)
fig = plt.figure().plt.plot(x, np.sin(x), '-')
plt.plot(x, np.cos(x), '--');
```

Saving figure to file:

One nice feature of matplotlib is the ability to save figures in a wide variety of formats. You can save the figure using the savefig() command.

for eg) To save the previous figure as a PNG file you can run this:

```
In [5]: fig.savefig('my-figure.png')
```

we now have a file called my-figure.png in the current working directory.

```
In [6]: !ls -lh my-fig.png
-rw-r--r-- 1 Jake 197121 16K Aug 11 05:59 my-fig.png
```

Let's use IPython Image object to display the contents of this file:

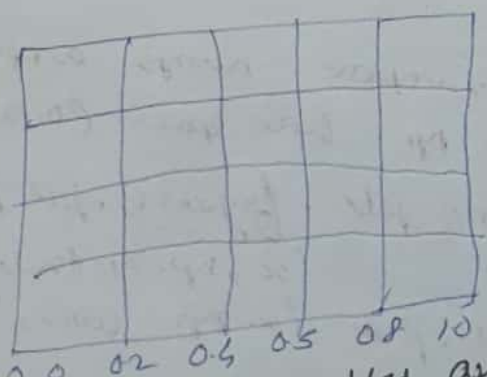
Simple Line plots:

The simplest of all plots is the visualization of a single function $y = f(x)$ for creating a simple plot of this type. (Notebook for plotting and importing functions)

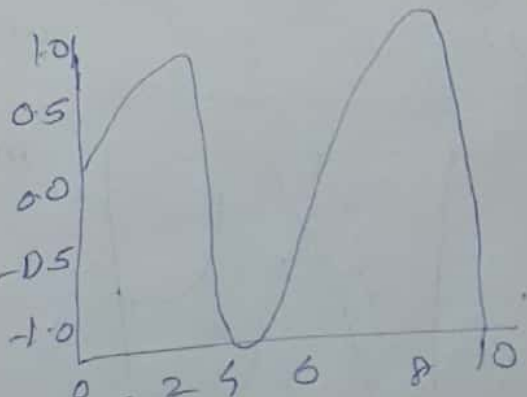
```
In [1]: %matplotlib inline
import matplotlib.pyplot as plt
import numpy as np
```

we start by creating a figure and axes.

```
In [2]: fig = plt.figure()
        ax = plt.axes()
```

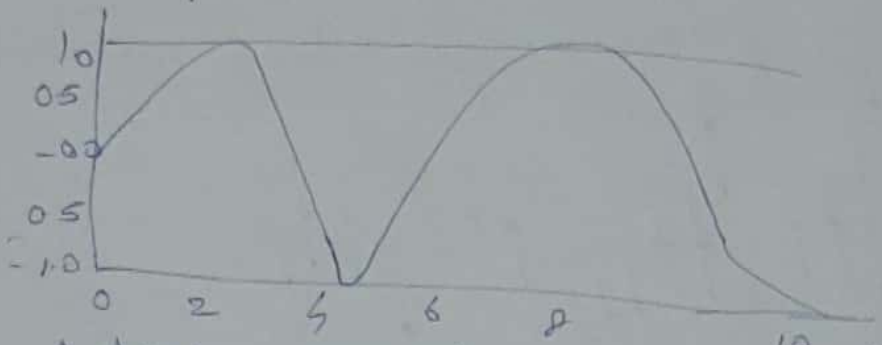


An empty gridded axes.



In [3]: fig = plt.figure() ax = plt.axes()
x = np.linspace(0, 10, 100) ax.plot(x, np.sin(x));

In [4]: plt.plot(x, np.sin(x))
plt.plot(x, np.cos(x));



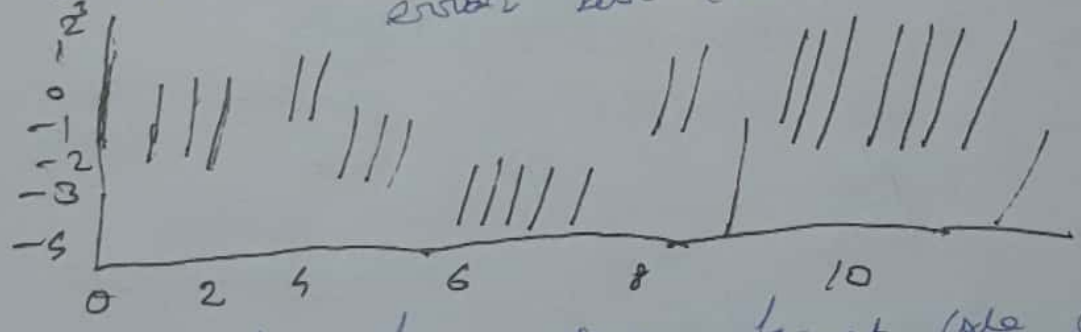
A simple sinusoid via 10 object oriented interface

Visualizing Errors:

Showing the Errors of data and results effectively can make a plot convey much more complete information

Basic Errorbars.

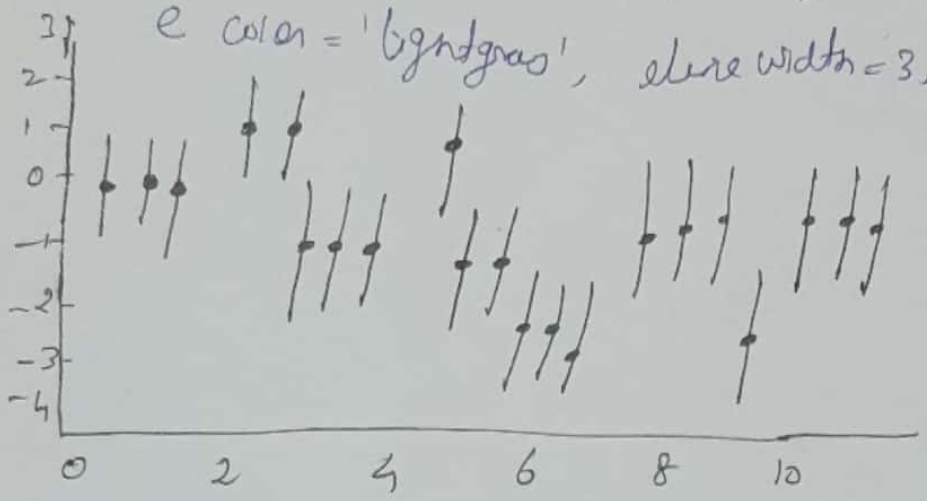
In [1]: % matplotlib inline
import matplotlib.pyplot as plt plt.style.use('seaborn-whitegrid')
import numpy as np
In [2]: x = np.linspace(0, 10, 50)
dy = 0.8
y = np.sin(x) + dy * np.random.randn(50) plt.errorbar(x, y, yerr = dy, fmt = 'k');



Here the fmt is a format code controlling the appearance of lines and points, and has the same syntax as the shorthand used in plt.plot outlined in "Simple Line plots".

The errorbar function has many options to fine-tune the outputs.

```
In[3]: plt.errorbar(x, y, yerr=dy, fmt='o',
                    color='black',
                    ecolor='lightgray', elinewidth=3, capsize=0);
```



In addition there options you can specify horizontal errorbars (xerr), one-sided errorbars and many other variants for more information on the options available, refer to the docstring of plt.errorbar

Density and contour plots:

→ Sometimes it is useful to display three-dimensional data in two dimensions using contours or color-coded Regions.

→ there are Matplotlib functions that can be helpful for this task.

plt.contour for contour plots.
plt.contourf for filled contour plots.
plt.imshow for showing images

Notebook for plotting and importing functions we will use.

In [1]: %matplotlib inline
import matplotlib.pyplot as plt plt.style.use('seaborn-white')
import numpy as np

Visualizing a three dimensional function.

→ We will start by demonstrating a contour plot using a function $z = f(x, y)$ using the following particular choice.

In [2]: def f(x,y) return np.sin(x) ^ np.cos(y)
→ return np.sin(x) ** 10 + np.cos(10 + y * x) * np.cos(x)

A contour plot can be created with the plt.contour function. It takes three arguments

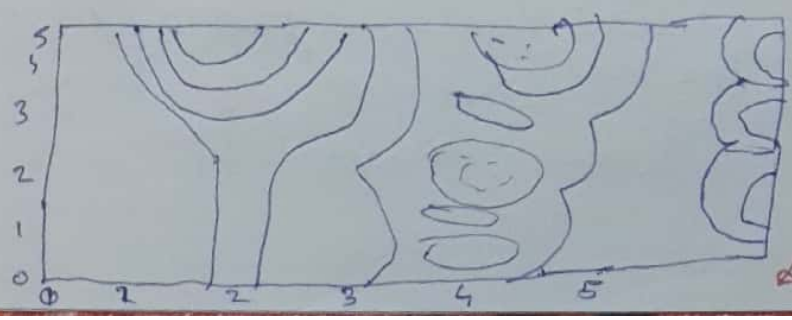
A grid of x values
A grid of y values
A grid of z values

→ the x and y values represent the positions on the plot.
z values represented by the contour levels.
np.meshgrid function → which builds two-dimensional grids from one-dimensional arrays.

In [3]: x = np.linspace(0, 5, 50)
y = np.linspace(0, 5, 40)
x, y = np.meshgrid(x, y) z = f(x, y)

x, y = np.meshgrid(x, y)
z = f(x, y)

In [4]: plt.contour(X, Y, Z, colors = 'black');



Visualizing three dimensional data with contours

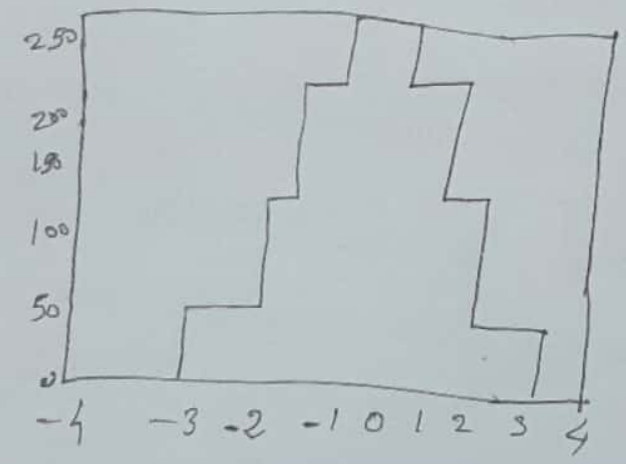
Diagram or map.

Histograms : Binning and Density:

A simple Histogram can be great first step in understanding dataset earlier we saw a preview of Matplotlibs' histogram function which creates a basic histogram in one line once the normal boiler plate imports are done.

```
In [1]: % matplotlib inline
        import numpy as np
        import matplotlib.pyplot as plt
        plt.style.use('seaborn-white')
        data = np.random.randn(1000)
```

```
In [2]: plt.hist(data);
```




```
x1 = np.random.normal(0, 0.8, 1000)
plt.hist(x1);
```

```
# contour can used to plot
plt.contour(x, y, z, 20,
            cmap = 'RdGy');
plt.colorbar();
contourf(for ➡)
plt.imshow(z, extent=
    [0,5, 0,5], origin='lower',
    cmap = 'RdGy')
plt.colorbar()
plt.axis(aspect='image');
```

the hist() function has many options to tune both the calculation and the display

```
In [3]: plt.hist(data, bins = 30, normed = True,
            alpha = 0.5, histtype = 'stepfilled', color =
            'steelblue', edgecolor = 'none');
```

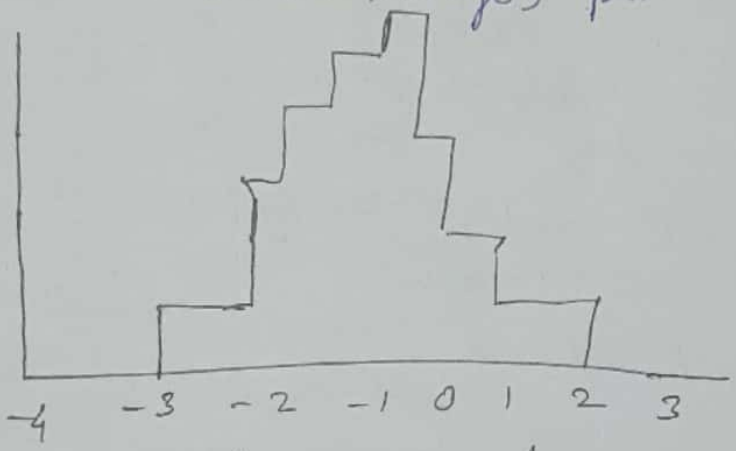
I find the combination of histtype='stepfilled' along with some transparency

In [4]: x_1 = np.random.normal(0, 0.8, 1000)

x_2 = np.random.normal(-2, 1, 1000)

x_3 = np.random.normal(3, 2, 1000)

kwargs = dict(histtype='stepfilled', alpha=0.3, normed=True, bins=40) plt.hist(x_1, **kwargs) plt.hist(x_2, **kwargs) plt.hist(x_3, **kwargs);



A customized histogram.

visualization with seaborn:

→ Matplotlib has proven to be incredibly useful and prolific visualization tool, but even avid users will admit it often leaves much to be desired. There are several valid complaints about Matplotlib that often come up

→ Prior to version 2.0, matplotlib's default are not exactly the best choice. It was based off of matlab circa 1999,

→ Matplotlib's API is relatively low level. Doing sophisticated statistical visualization is possible, but often requires a lot of boilerplate code.

→ Matplotlib predated pandas by more than a decade, and thus is not

designed for use with pandas DataFrame, you must extract each series and often concatenate them together into a right format. It would be nice to have a plotting library that can intelligently use the DataFrame labels in a plot.

Seaborn versus Matplotlib

Here is an example of a simple random walk plot in Matplotlib, using its classic plot formatting and colors

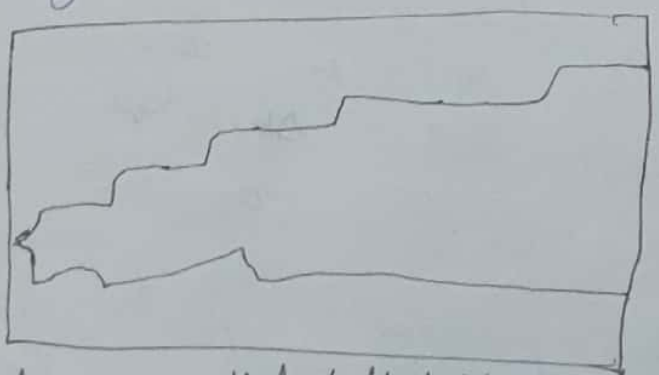
```
In [1]: import matplotlib.pyplot as plt
        plt.style.use('classic')
        %matplotlib inline
        import numpy as np
        import pandas as pd
```

Now we create some random walk data

```
In [2]: # Create some data
        rng = np.random.RandomState(0)
        x = np.linspace(0, 10, 500)
        y = np.cumsum(rng.randn(500, 6), 0)
```

And do a simple plot.

```
In [3]: plt.plot(x, y)
        plt.legend('ABCDEF', ncol=2, loc='upper left');
```



Data in Matplotlib's default style.

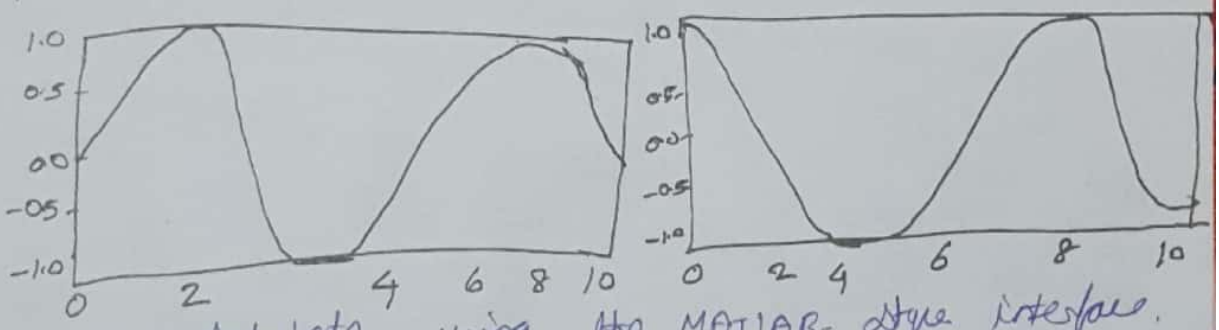
Two Interfaces Of Matplotlib.

i) Matlab-style state-based interface
ii) Object-oriented Interface.

MATLAB-style Interface.

Matplotlib was originally written as a Python alternative for MATLAB users.
The MATLAB-style tools are contained in the pyplot (plt) interface.

```
plt.figure() # create a plot figure
# create the first of two panels and set current axis
plt.subplot(2, 1, 1) # (rows, columns, panel number)
plt.plot(x, np.sin(x))
# create the second panel and set current axis.
plt.subplot(2, 1, 2)
plt.plot(x, np.cos(x));
```



⇒ Subplots using the MATLAB-style interface.

This interface is stateful
Keeps track of the "current" figure and axes.
Reference to these can be obtained using the plt.gcf() (get current figure) and plt.gca() (get current axes) routines.
Stateful interface is fast and convenient for simple plots.
Adding something to the first panel after creating the second panel cannot be done.

Object-oriented interface:

The object-oriented interface is available for more complicated situations and provide more control over the figure.

plotting functions are figure and axes objects and explicit.

Object-oriented interface are more readable can structure the code with split plotting. Labbling and other tasks into the code blocks.

Example:

```
# first create a grid of plots.
# ax will be an array of two Axes objects
fig, ax = plt.subplots(2)
# Call plot() method on the appropriate object
ax[0].plot(x, np.sin(x))
ax[1].plot(x, np.cos(x));
```

Line plots:

→ The Pyplot a sublibrary of matplotlib, is a collection of functions that helps in creating a variety of charts.

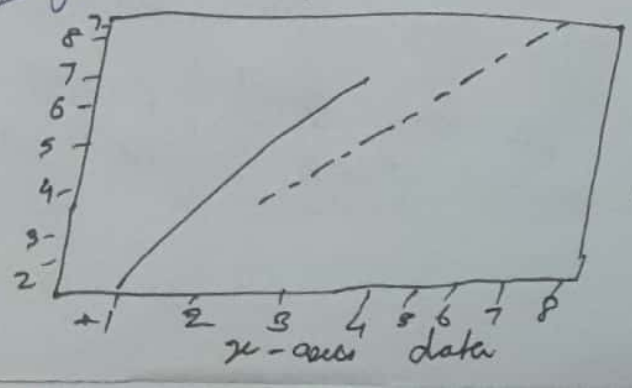
→ Line plots are used to represent the readers between two data x and y on a different axes

→ Import Matplotlib.Pyplot library for plotting functions.

Import the numpy library as per requirements.

Define Data values x and y.

Multiple plots

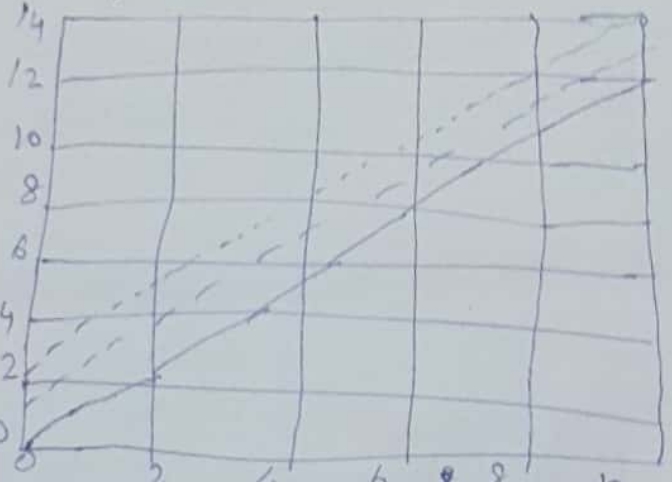


Multiple plots.

Setting Line colours and styles

The plt.plot() function takes additional arguments to specify line colors and styles. To adjust the color, color keyword is used, which accepts a string argument representing virtually any imaginable color. The color can be specified in a variety of ways

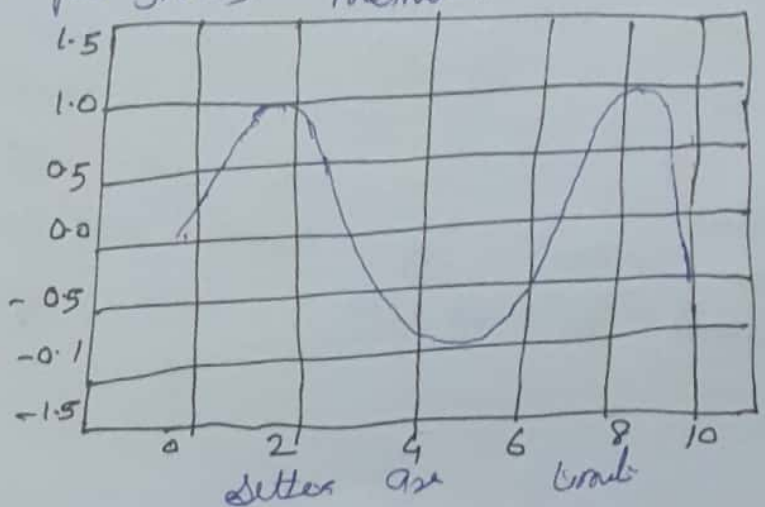
```
In[]: plt.plot(x, x+0, '-g') # solid green
      plt.plot(x, x+1, '--c') # dashed cyan
      plt.plot(x, x+2, '-.k') # dashdot black
      plt.plot(x, x+3, ':r'); # dotted red
```



controlling colors and styles with the shorthand syntax.

Setting the axes limits of plots

Default axes limits are chosen for the plot automatically. It can also be controlled by adjusting the axis limits using plt.xlim() and plt.ylim() methods.

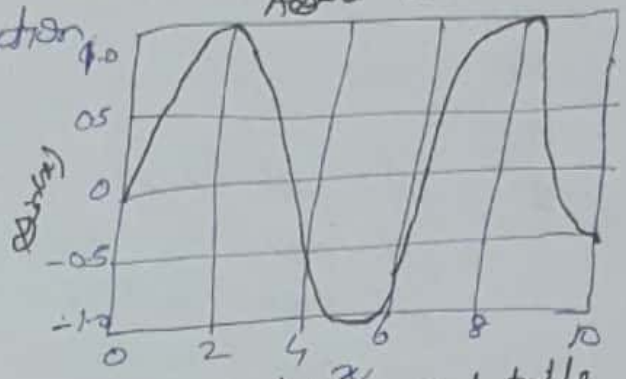


Setting axis limits

Labeling plots:

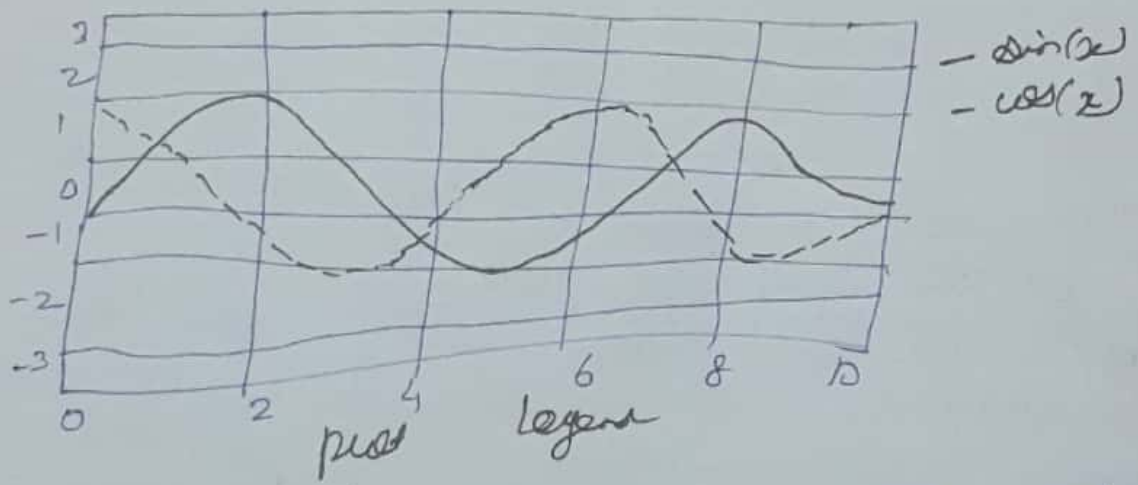
Labelling plots with the titles, axis labels and simple legends can be done with the pyplot as below. The position, size, and style of these labels can be altered using optional arguments to the function.

```
In[]: plt.plot(x, np.sin(x))
      plt.title("A sine curve")
      plt.xlabel('x')
      plt.ylabel("sin(x)");
```



Axes labels and title

```
In[]: plt.plot(x, np.sin(x), '-g', label='sin(x)')
      plt.plot(x, np.cos(x), ':b', label='cos(x)')
      plt.axis('equal')
      plt.legend()
```



plot legend

Scatter plots:

Scatter plots are used to observe relationships between variables. Scatter plot is a type of plot in which the points are represented individually with a dot, circle, or other shape instead of points being joined by line segments as in the case of a line plot.

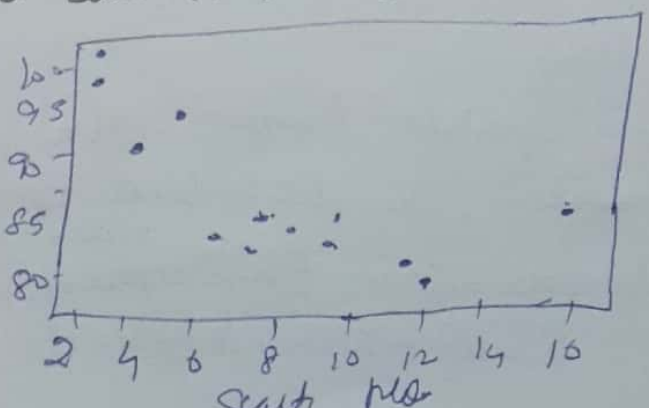
The scatter() method in the matplotlib library is used to draw a scatter plot. Scatter plots are used to visualize the relation among variables and how change in one affects the other variable.

Syntax: the syntax for scatter() method is given below:

matplotlib.pyplot.scatter(x_axis_data, y_axis_data, s=None, c=None, marker=None, cmap=None, vmin=None, vmax=None, alpha=None, linewidths=None, edgecolors=None)

where x_axis_data - An array containing x axis data
y_axis_data - An array containing y axis data.
s - marker size (can be scalar or array of size equal to size of x & y)
c - color of sequence of colors for markers
marker - marker style
cmap - cmap name
linewidths - width of marker borders
edgecolor - marker border color
alpha - blending value between 0 (transparent) and 1 (opaque)

```
# Simple scatter plot
import matplotlib.pyplot as plt
x = [5,7,8,7,2,17,2,9,4,11,12,9,6]  y = [99,86.87,88,100,86,103,87,94,78,77,85,86]
plt.scatter(x, y, c="blue")  # to show the plot plt.show()
```



Legends:

Plot Legends give meaning to a visualization, assigning labels to the various plot elements. The simplest Legend can be created with the plt.legend() command. When automatically creates a Legend for any labeled plot elements.

```
In [1]: import matplotlib.pyplot as plt
        plt.style.use('classic')
In [2]: %matplotlib inline
        import numpy as np
In [3]: x = np.linspace(0, 10, 1000)
        fig, ax = plt.subplots()
        ax.plot(x, np.sin(x), '-b', label='sine')
        ax.plot(x, np.cos(x), '--r', label='cosine')
        ax.axis('equal')
        leg = ax.legend();
```

Default plot legend.

```
In [4]: ax.legend(loc='upper left', frameon=False)
In [5]: ax.legend(frameon=False, loc='lower center',
                  ncol=2)
In [6]: ax.legend(fancybox=True, framealpha=1,
                  shadow=True, borderpad=1)
```

Choosing Elements for the Legend:

The Legend includes all labeled elements by default. the labels appear in the Legend by using the objects returned by plot commands.

The plt.plot() commands is able to create multiple lines at once, and returns a list of created line instance. Passing one offthese to plt.Legend() will end it which to identify, along with the labels.

```
In [7]: y = np.sin(x[:, np.newaxis] + np.pi * np.arange(0, 2, 2.05))
        Lines = plt.plot(x, y)
# Lines is a list of plt.Line 2D instances
plt.legend(Lines[:2], ['first', 'second']);
```

COLORS:

Plot legends identifies discrete labels of discrete points. for continuous labels based on the color of points, lines, or regions, a labeled colorbar. Can be used. In matplotlib a colorbar is a separate axes that can provide a key for the meaning of colours in a plot.

for plotting and importing the functions:

```
In [1]: import matplotlib.pyplot as plt
        plt.style.use('classic')
In [2]: %matplotlib inline
        import numpy as np
```

The simplest colorbar can becreated with the plt.colorbar function.

```
In [3]: x = np.linspace(0, 10, 1000)
        I = np.sin(x) * np.cos(x[:, np.newaxis])
        plt.imshow(I)
        plt.colorbar();
```

Choosing the colormap:

The three categories of colormaps:

I) Sequential colormaps: these consist of one continuous sequence of colors (eg: binary or Viridis).

II) Divergent colormaps: these usually contain two distinct colors, which show positive & negative deviation from a mean (eg: RdBu or PuOr)

III) Qualitative colormaps: these mix colors with no particular sequence (eg: rainbow or Jet)

The Jet colormap, which was the default in matplotlib prior to version 2.0 is an example of a qualitative colormap. Its status as the default was quite unfortunate, because qualitative maps are often poor choice for representing quantitate data.

Color limits and extensions:

Matplotlib allows for a large range of colorbar customization. The colorbar is an instance of plt.Axes, so all of the axes and tick formatting are applicable.

Discrete color Bars:

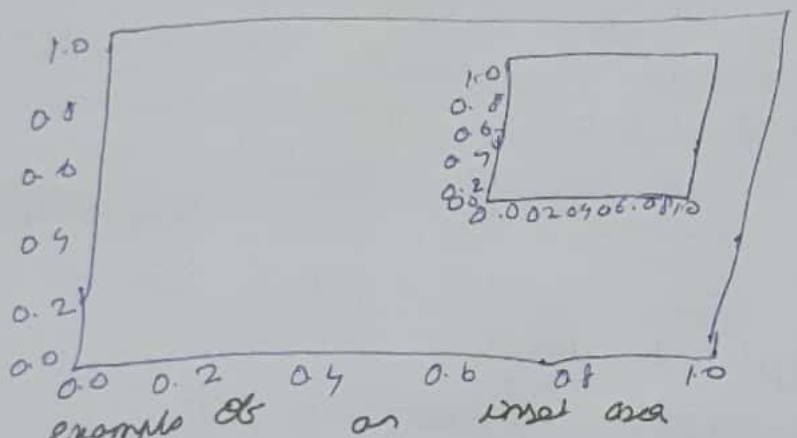
Colormaps are by default continuous but can be represented with discrete values. The easier way to do this is to use the plt.cm.get_cmap() function and pass the name of a suitable colormap along with the number of desired bins:

```
In []: plt.imshow(I, cmap = plt.cm.get_cmap
                  ('Blues', 6))
       plt.colorbar()
       plt.clim(-1, 1);
```

Subplots:

It is necessary to compare different views of data side by side. Matplotlib has the concept of subplots: group of smaller axes that can exist together within a single figure.

```
In [1]: %matplotlib inline
        import matplotlib.pyplot as plt
        plt.style.use('seaborn-white')
        import numpy as np
In [2]: ax1 = plt.axes()  #standard axes
        ax2 = plt.axes([0.65, 0.65, 0.2, 0.2])
```

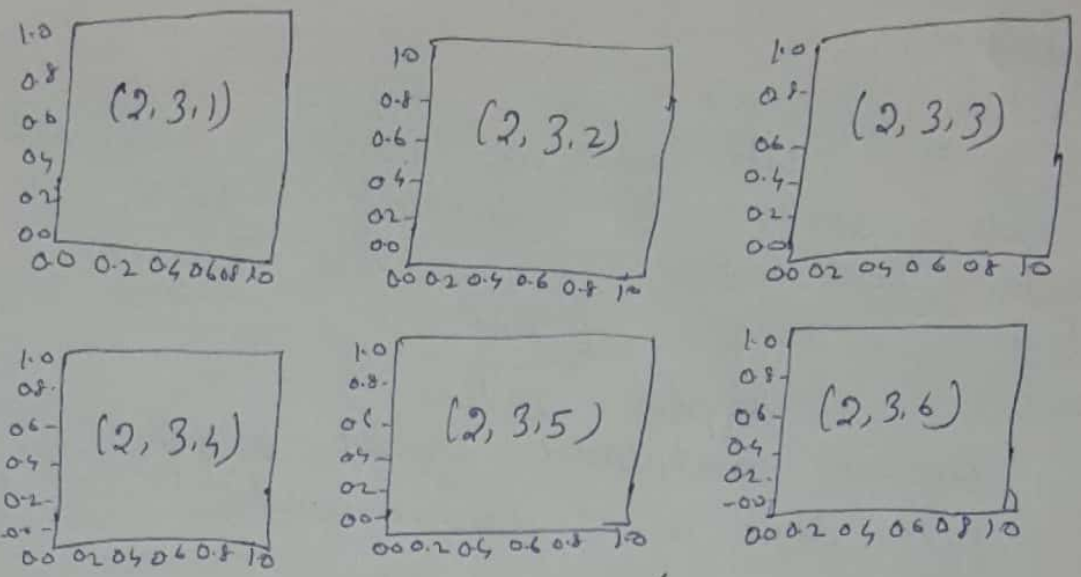


example of an inset axes

plt.subplot: Simple Grids of subplots.

=> plt.subplot() creates a single subplot within a grid. This command takes three integer arguments— the number of rows, the number of columns, and the index of the plot to be created in this scheme, which runs from the upper left to the bottom right.

```
In [3]: for i in range(1,7):
            plt.subplot(2,3,i);
            plt.text(0.5, 0.5, str((2,3,i)),
            fontsize=18, ha='center')
fig, ax = plt.subplots(3,4), sharex='col', sharey='row')
```



plt.subplot() example

Text and Annotation

Good visualization guides the user so that the figure conveys a story. In some cases, this story can be expressed as an entirely visual manner. without the need for added text, but in others, small textual cues and labels are necessary. The basic type of annotations are axes labels and titles

In [1]: %matplotlib inline.

```
import matplotlib.pyplot as plt
import matplotlib as mpl
plt.style.use('seaborn-whitegrid')
import numpy as np
import Pandas as pd.
```

Transforms and Text position:

ax.transData : Transform associated with data coordinates

ax.transAxes: Transform associated with the axes (in units of axes dimensions)

fig.transFigure: Transform associated with the figure (in units of figure dimensions)

Arrows and Annotation

Drawing arrows in Matplotlib is much harder. plt.arrow() function: the arrows it creates are SVG objects that will be subject to the varying aspect ratio of the plots.

plt.annotate() function: this function creates some text and an arrow, and the arrows are flexible.

The arrow style is controlled through the arrowprops dictionary

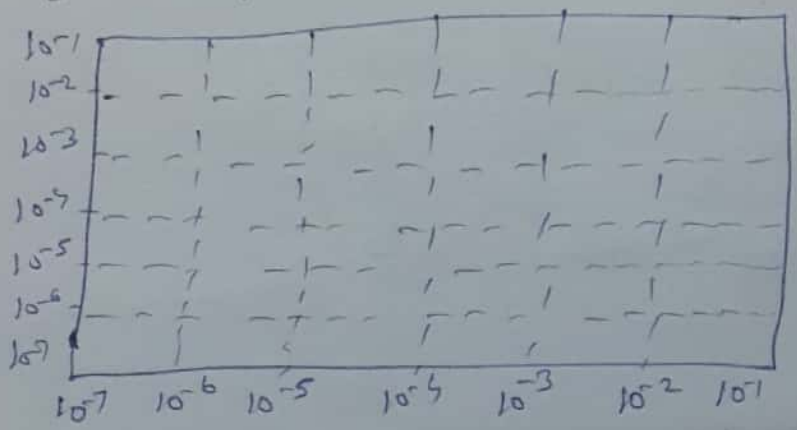
CUSTOMIZATION:

Customizing Ticks: matplotlib's default tick locators and formatters are designed to be generally sufficient in many common situations. but are in no way optimal for every plot.

Major and minor Ticks:

within each axes, there is the concept of a major tick mark and a minor tick mark. As the names would imply, major ticks are usually bigger or more pronounced while minor ticks are usually smaller. by default, Matplotlib rarely makes use of minor ticks, but one place you can see them is within logarithmic plots.

```
In [1]: % matplotlib inline
        import matplotlib.pyplot as plt
        plt.style.use('seaborn-whitegrid')
        import numpy as np
In [2]: ax = plt.axes(xscale='log', yscale='log')
```



Geographic Data with Basemap

A common type of visualization in data science is that of geographic data. Matplotlib's main tool for this type of visualization is the Basemap toolkit.

This is several Matplotlib toolkits that are under the mpl_toolkits namespace, Basemap is a useful tool for python users to have in their visual toolbelts.

Installation of Basemap. pip install basemap

$ conda install basemap

```
In[1]: % matplotlib inline
       import numpy as np
       import matplotlib.pyplot as plt
       from mpl_toolkits.basemap import Basemap
In[2]: plt.figure(figsize=(8,8))
       m = Basemap(projection='ortho', resolution=None,
                   lat_0=50, lon_0=-100)
       m.bluemarble(scale=0.5)
```

27.77

Map Projections:

It is used to project a spherical map like that of the earth, onto a flat surface without distortions it or breaking its continuity.

Depending on the intended use of the map projection, there are certain map features

```
In[]:
from itertools import chain
def draw_map(m, scale=0.2):
#from a shaded_relief image
m.shadedrelief(scale=scale)
```

```
# lats and longs are returned as a dictionary
lats = m.drawparallels(np.linspace(-90, 90, 13))
lons = m.drawmeridians(np.linspace(-180, 180, 13))
# keys contain the plt.Line2D instances.
lat_lines = chain(*(tup[1][0] for tup in lats.items()))
lon_lines = chain(*(tup[1][0] for tup in lons.items()))
all_lines = chain(lat_lines, lon_lines)
# cycle through these lines and set the desired style.
for line in all_lines:
    line.set(linestyle='-', alpha=0.3, color='w')
```

Cylindrical projections:

The simplest of map projections are cylindrical projections. in which lines of constant latitude and longitude are mapped to horizontal and vertical lines respectively.

```
fig = plt.figure(figsize=(8,6), edgecolor='w')
m = Basemap(projection='cyl', resolution=None,
            llcrnrlat=-90, urcrnrlat=90,
            llcrnrlon=-180, urcrnrlon=180)
draw_map(m)
```

Pseudo-cylindrical projections:

It gives better properties near the poles of the projection.

The extra arguments to Basemap refers to the central latitude (lat_0) and longitude (lon_0) for the desired map

orthographic projection

```
In [ ]: fig = plt.figure(figsize=(8,8))
        m = Basemap(projection='ortho', resolution=None,
                    lat_0=50, lon_0=0)
        draw_map(m);
```

Drawing a Map Background

The Basemap package contains functions for drawing borders of physical features like continents oceans, lakes and rivers, as well as political boundaries.

The following are some of its available drawing functions using IPython's help features:

Physical boundaries and bodies of water

drawcoastlines(): Draw continental coast lines.

drawlsmask(): Draw a mask between the land and sea, for use with projecting images on one.

drawrivers(): Draw rivers on the map

political boundaries

drawcountries(): Draw country boundaries

drawstates(): Draw US state boundaries

drawcounties(): Draw US county boundaries.

Map features drawgreatcircle(): Draw a great circle between two points

drawparallels(): Draw lines of constant latitude

drawmeridians(): Draw lines of constant longitude.

drawmapscale(): Draw a linear scale on the map.

whole-globe images:

bluemarble(): Project NASA's blue marble image onto the map

shadedrelief(): Project a shaded relief image onto the map

etopo(): Draw an etopo relief image onto the map

warpimage(): Project a user-provided image onto a map.

Thank you.