# Python Programming for Data Analytics

***ATAL-FDP on Data Analytics using Python / R Programming***

***Department of Computer Science & Engineering***

***National Institute of Technology Puducherry, Karaikal***

***Date : 23-August-2021***

***Dr.P.Thiyagarajan M.S.,Ph.D.,Post-Doc***

***Assistant Professor, Department of Computer Science***

***Central University of Tamil Nadu, Thiruvarur – 05***

***Email : thiyagu@cutn.ac.in***

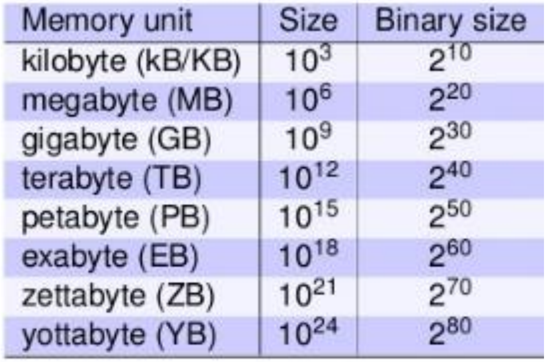
***Mobile : 0 9488584015 / 0 9841234510 (W)***

# Agenda

- Introduction to Data Analytics
- Why Python for Data Analytics?
- Python installation
- Introductory examples
- Numpy Basics
- Pandas
- Data loading, Storage and File formats
- Data Wrangling : Clean, Transform, Merge, Reshape
- Small Projects

# Big Data

- Massive sets of unstructured / semi-structured data from Web traffic, social media, sensors, etc
- Petabytes, exabytes, yottabytes of data
  - Volumes too great for typical DBMS
- Information from multiple internal and external sources:
  - Transactions
  - Social media
  - Enterprise content
  - Sensors
  - Mobile devices
- In the last minute there were …….



| Memory unit | Size | Binary size |
|---|---|---|
| kilobyte (kB/KB) | $10^3$ | $2^{10}$ |
| megabyte (MB) | $10^6$ | $2^{20}$ |
| gigabyte (GB) | $10^9$ | $2^{30}$ |
| terabyte (TB) | $10^{12}$ | $2^{40}$ |
| petabyte (PB) | $10^{15}$ | $2^{50}$ |
| exabyte (EB) | $10^{18}$ | $2^{60}$ |
| zettabyte (ZB) | $10^{21}$ | $2^{70}$ |
| yottabyte (YB) | $10^{24}$ | $2^{80}$ |

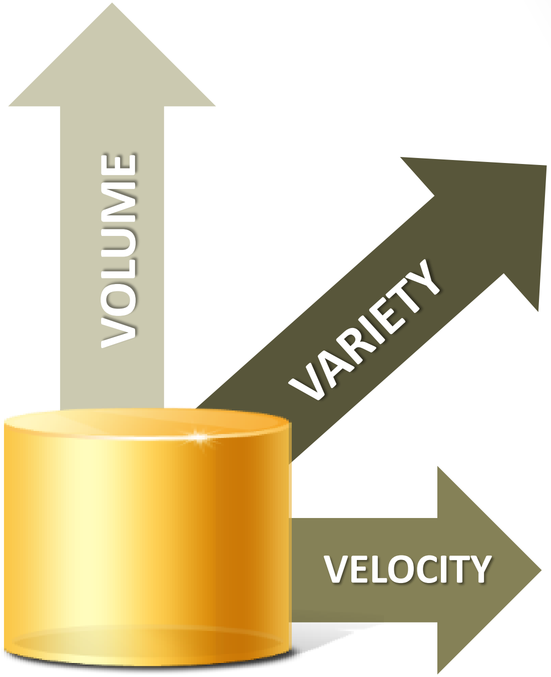
- **204 million emails sent**
- **61,000 hours of music listened to on Pandora**
- **20 million photo views**
- **100,000 tweets**
- **6 million views and 277,000 Facebook Logins**
- **2+ million Google searches**
- **3 million uploads on Flickr**

# Big Data Characteristics

- **Growing quantity of data**
- **e.g. social media, behavioral, video**

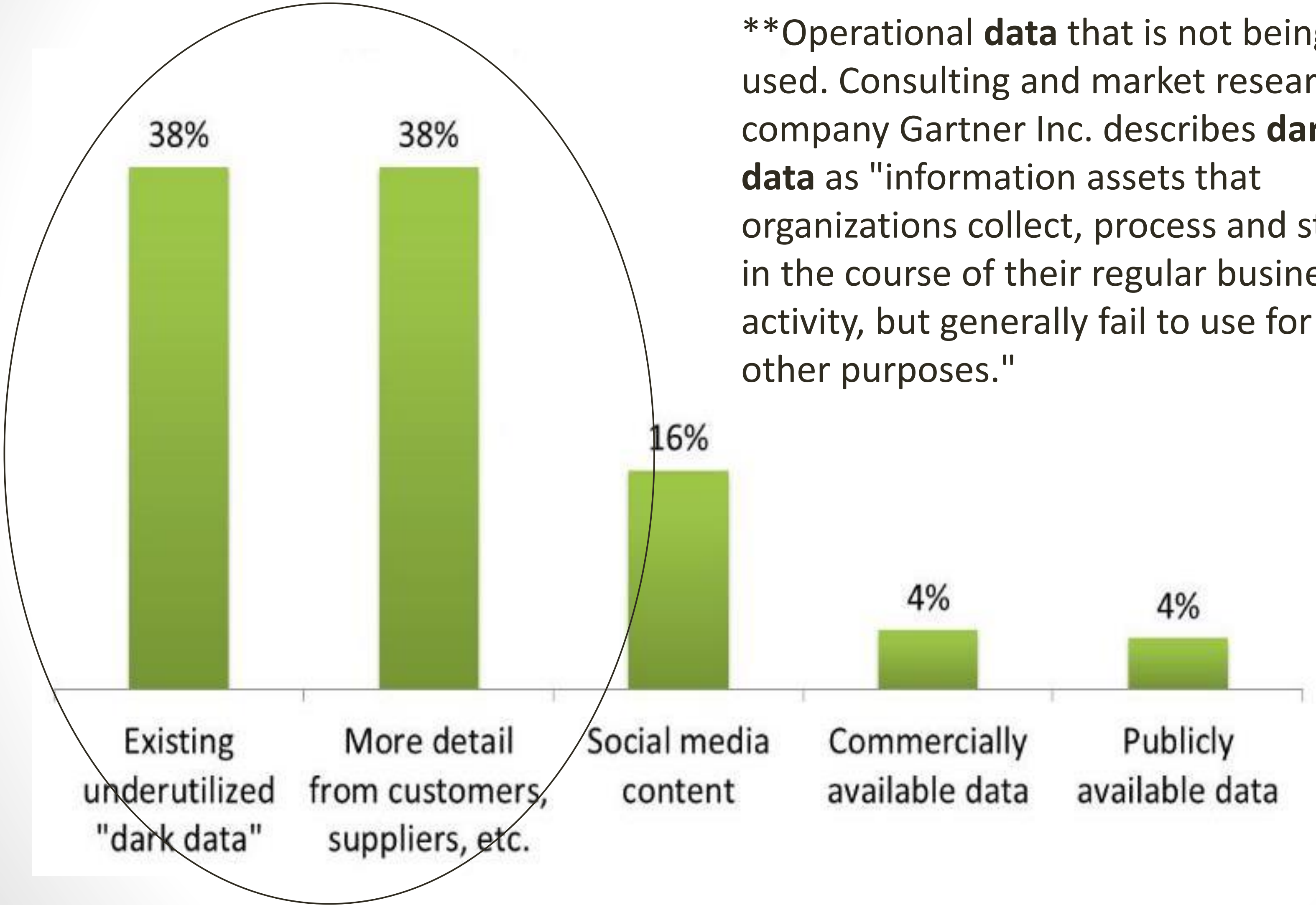
- **Quickening speed of data**
- **e.g. smart meters, process monitoring**

- **Increase in types of data**
- **e.g. app data, unstructured data**



Gartner, Feb 2001

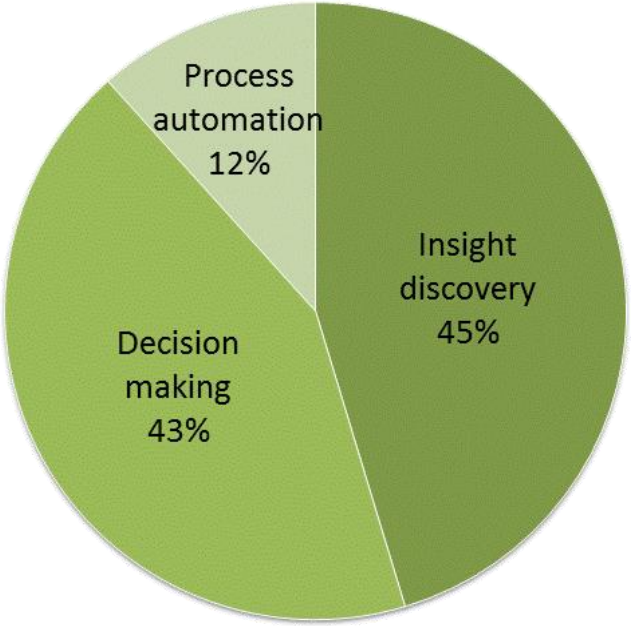
# Which source of data represents the most immediate opportunity?



**Operational **data** that is not being used. Consulting and market research company Gartner Inc. describes **dark data** as "information assets that organizations collect, process and store in the course of their regular business activity, but generally fail to use for other purposes."

*Source: Getting Value from Big Data, Gartner Webinar, May 2012*

# Which is the biggest opportunity for Big Data in your organization?



Through 2017:

85% of Fortune 500 organizations will be unable to exploit big data for competitive advantage.

Business analytics needs will drive 70% of investments in the expansion and modernization of information infrastructure.

*Source: Getting Value from Big Data, Gartner Webinar, May 2012*

# Types of Data

- When collecting or gathering data we collect data from individuals cases on particular variables.
- A *variable* is a unit of data collection whose value can vary.
- Variables can be defined into *types* according to the level of mathematical scaling that can be carried out on the data.
- There are four types of data or levels of measurement:

| | |
|---|---|
| **1. Categorical (Nominal)** | **2. Ordinal** |
| **3. Interval** | **4. Ratio** |

# Categorical (Nominal) data

- **Nominal or categorical** data is data that comprises of categories that *cannot* be rank ordered – each category is just different.

- The categories available **cannot be placed in any order** and no judgement can be made about the relative size or distance from one category to another.
  - Categories bear no quantitative relationship to one another
  - Examples:
    - customer's location (America, Europe, Asia)
    - employee classification (manager, supervisor, associate)

- What does this mean**? No mathematical operations can be performed on the data relative to each other**.

- Therefore, nominal data reflect **qualitative differences** rather than quantitative ones.

# Nominal data

Examples:

**What is your gender?** *(please tick)*

Male □

Female □

**Did you enjoy the film?** *(please tick)*

Yes □

No □

- Systems for measuring nominal data must ensure that each category is **mutually exclusive** and the system of measurement needs to be **exhaustive**.
- Exhaustive: the system of categories system should have enough categories for all the observations
- Variables that have only two responses i.e. Yes or No, are known as *dichotomies*.

# Ordinal data

Example:

**How satisfied are you with the level of service you have received?** *(please tick)*

Very satisfied □

Somewhat satisfied □

Neutral □

Somewhat dissatisfied □

Very dissatisfied □

- Ordinal data is data that **comprises of categories that *can* be rank ordered.**

- Similarly with nominal data the distance between each category cannot be calculated but the **categories can be ranked above or below each other.**

  - No fixed units of measurement
  - Examples:

    - college football rankings

    - survey responses

    (poor, average, good, very good, excellent)

- What does this mean? Can **make statistical judgements** and perform limited maths.

# Interval and ratio data

- Both interval and ratio data are examples of **scale data**.
- Scale data:
  - data is in numeric format ($50, $100, $150)
  - data that can be **measured on a continuous scale**
  - the **distance** between each can be observed and as a result **measured**
  - the data can be **placed in rank order**.

# Interval data

- Ordinal data but with constant differences between observations
- Ratios are not meaningful
- Examples:

  - **Time** – moves along a continuous measure or seconds, minutes and so on and is without a zero point of time.

  - **Temperature** – moves along a continuous measure of degrees and is without a true zero.

  - **SAT scores**

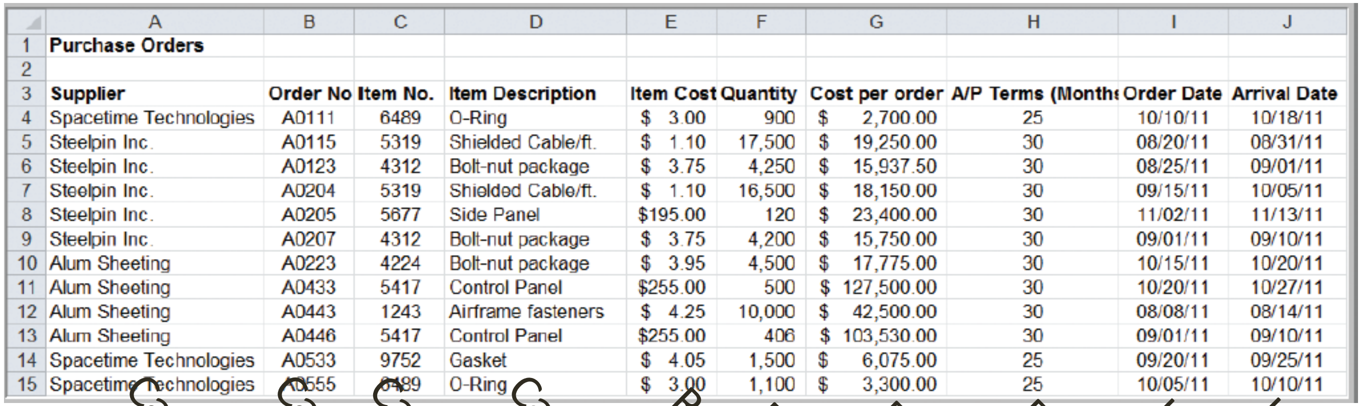
# Ratio data

- Ratio data measured on a ***continuous* scale** and ***does* have a natural zero point.**

  - Ratios are meaningful
  - Examples:
    - monthly sales
    - delivery times
    - Weight
    - Height
    - Age

# Data for Business Analytics

Classifying Data Elements in a Purchasing Database



| | A | B | C | D | E | F | G | H | I | J |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | **Purchase Orders** | | | | | | | | | |
| 2 | | | | | | | | | | |
| 3 | **Supplier** | **Order No** | **Item No.** | **Item Description** | **Item Cost** | **Quantity** | **Cost per order** | **A/P Terms (Month** | **Order Date** | **Arrival Date** |
| 4 | Spacetime Technologies | A0111 | 6489 | O-Ring | $ 3.00 | 900 | $ 2,700.00 | 25 | 10/10/11 | 10/18/11 |
| 5 | Steelpin Inc. | A0115 | 5319 | Shielded Cable/ft. | $ 1.10 | 17,500 | $ 19,250.00 | 30 | 08/20/11 | 08/31/11 |
| 6 | Steelpin Inc. | A0123 | 4312 | Bolt-nut package | $ 3.75 | 4,250 | $ 15,937.50 | 30 | 08/25/11 | 09/01/11 |
| 7 | Steelpin Inc. | A0204 | 5319 | Shielded Cable/ft. | $ 1.10 | 16,500 | $ 18,150.00 | 30 | 09/15/11 | 10/05/11 |
| 8 | Steelpin Inc. | A0205 | 5677 | Side Panel | $195.00 | 120 | $ 23,400.00 | 30 | 11/02/11 | 11/13/11 |
| 9 | Steelpin Inc. | A0207 | 4312 | Bolt-nut package | $ 3.75 | 4,200 | $ 15,750.00 | 30 | 09/01/11 | 09/10/11 |
| 10 | Alum Sheeting | A0223 | 4224 | Bolt-nut package | $ 3.95 | 4,500 | $ 17,775.00 | 30 | 10/15/11 | 10/20/11 |
| 11 | Alum Sheeting | A0433 | 5417 | Control Panel | $255.00 | 500 | $ 127,500.00 | 30 | 10/20/11 | 10/27/11 |
| 12 | Alum Sheeting | A0443 | 1243 | Airframe fasteners | $ 4.25 | 10,000 | $ 42,500.00 | 30 | 08/08/11 | 08/14/11 |
| 13 | Alum Sheeting | A0446 | 5417 | Control Panel | $255.00 | 406 | $ 103,530.00 | 30 | 09/01/11 | 09/10/11 |
| 14 | Spacetime Technologies | A0533 | 9752 | Gasket | $ 4.05 | 1,500 | $ 6,075.00 | 25 | 09/20/11 | 09/25/11 |
| 15 | Spacetime Technologies | A0555 | 6489 | O-Ring | $ 3.00 | 1,100 | $ 3,300.00 | 25 | 10/05/11 | 10/10/11 |

Figure 1.2

Categorical Categorical Categorical Categorical Ratio Ratio Ratio Ratio Interval Interval

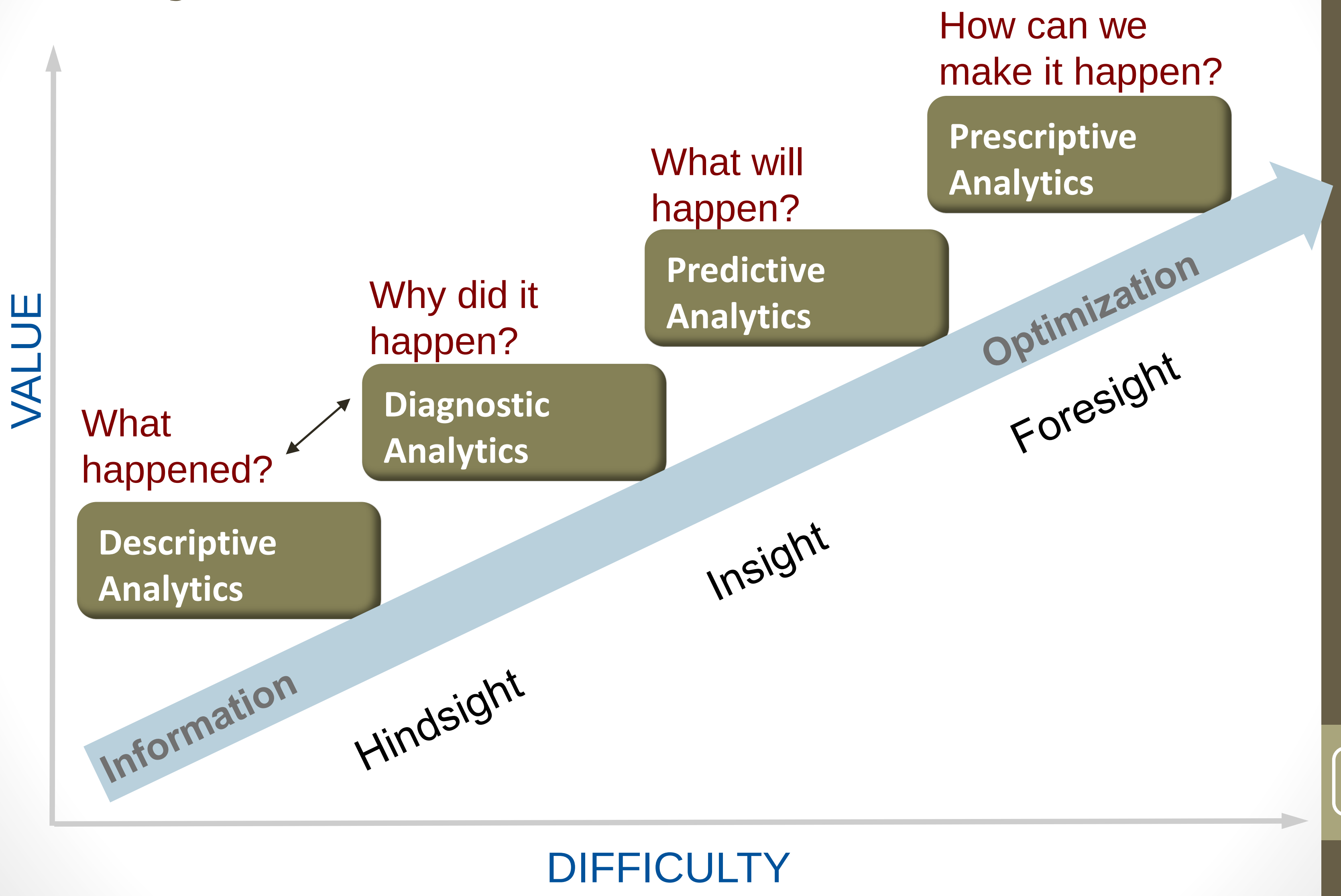
If there was field (column) for **Supplier Rating** (*Excellent, Good, Acceptable, Bad*), that data would be classified as **Ordinal**

# Business Analytics/Business Intelligence

- Business Analytics/Business intelligence (BI) is a broad category of applications, technologies, and processes for:
  - gathering,
  - storing,
  - accessing, and
  - analyzing data

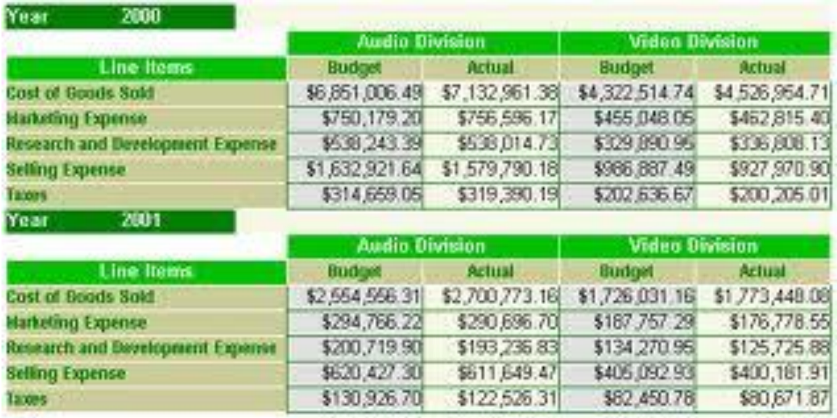
- to help business users make better decisions.

# Analytics Models



# Descriptive Analytics

- Descriptive analytics, such as reporting/OLAP, dashboards, and data visualization, have been widely used for some time.
- They are the core of traditional BI.

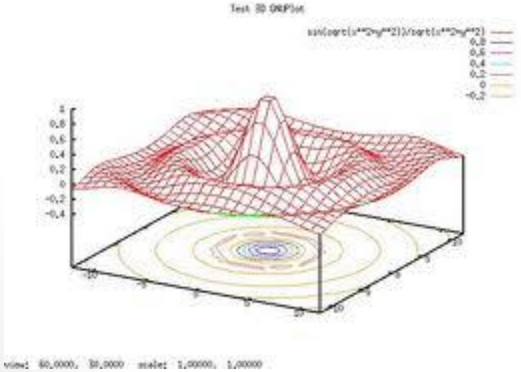


Year 2000

| Line Items | Audio Division |  | Video Division |  |
|---|---|---|---|---|
|  | Budget | Actual | Budget | Actual |
| Cost of Goods Sold | $6,851,006.49 | $7,132,961.38 | $4,322,514.74 | $4,526,954.71 |
| Marketing Expense | $750,179.20 | $756,596.17 | $455,048.05 | $462,815.40 |
| Research and Development Expense | $538,243.39 | $538,014.73 | $329,890.95 | $336,808.13 |
| Selling Expense | $1,632,921.64 | $1,579,790.18 | $986,887.49 | $927,970.90 |
| Taxes | $314,659.05 | $319,390.19 | $202,636.67 | $200,205.01 |

Year 2001

| Line Items | Audio Division |  | Video Division |  |
|---|---|---|---|---|
|  | Budget | Actual | Budget | Actual |
| Cost of Goods Sold | $2,654,556.31 | $2,700,773.16 | $1,726,031.16 | $1,773,448.08 |
| Marketing Expense | $294,766.22 | $290,696.70 | $187,757.29 | $176,778.55 |
| Research and Development Expense | $200,719.90 | $193,236.83 | $134,270.95 | $125,725.88 |
| Selling Expense | $620,427.30 | $611,649.47 | $405,092.93 | $400,181.91 |
| Taxes | $130,926.70 | $122,526.31 | $82,450.78 | $80,671.87 |

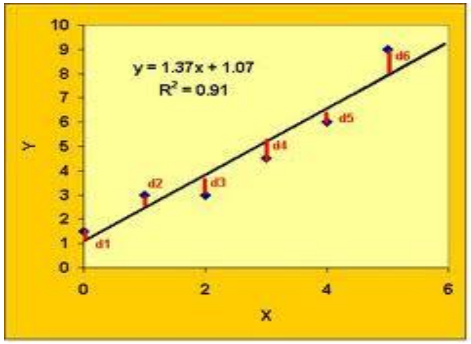

Test 3D GNUPlot
sin(sqrt(x**2+y**2))/sqrt(x**2+y**2)
0.8
0.6
0.4
0.2
0
-0.2
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
view: 60.0000, 30.0000 scale: 1.00000, 1.00000

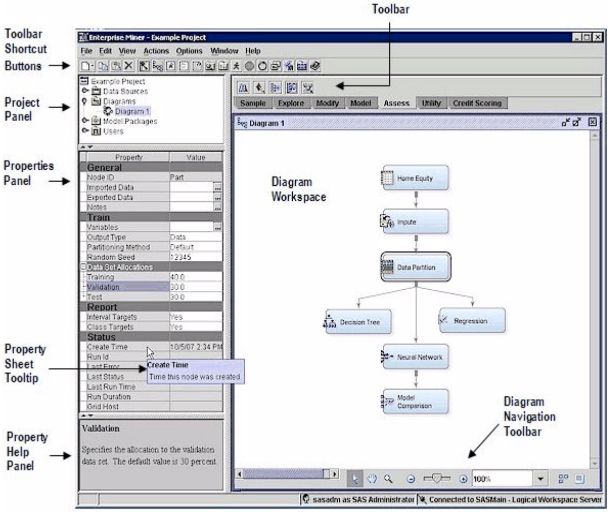
## What has occurred?

- Descriptive analytics, such as data visualization, is important in helping users interpret the output from predictive and predictive analytics.

# Predictive Analytics

- Algorithms for predictive analytics, such as regression analysis, machine learning, and neural networks, have also been around for some time.





| Property | Value |
|---|---|
| General | |
| Node ID | Part |
| Imported Data | ... |
| Exported Data | ... |
| Notes | ... |
| Train | |
| Variables | ... |
| Output Type | Data |
| Partitioning Method | Default |
| Random Seed | 12345 |
| Data Set Allocations | |
| Training | 40.0 |
| Validation | 30.0 |
| Test | 30.0 |
| Report | |
| Interval Targets | Yes |
| Class Targets | Yes |
| Status | |
| Create Time | 10/5/07 2:34 PM |
| Run Id | |
| Last Error | |
| Last Status | |
| Last Run Time | |
| Run Duration | |
| Grid Host | |

Create Time
Time this node was created

Validation

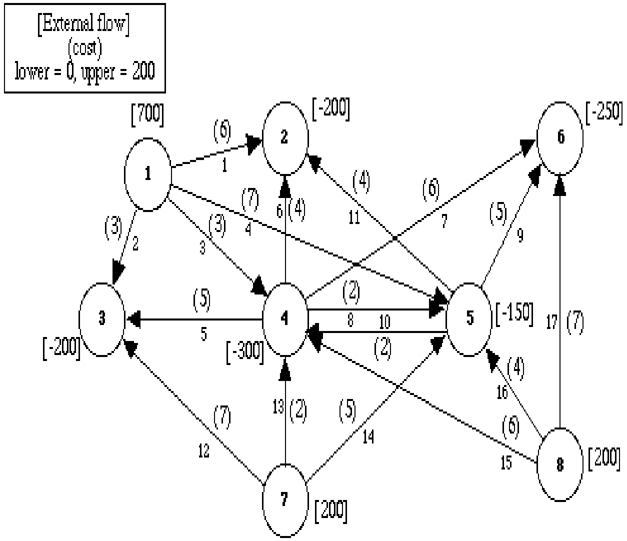
Specifies the allocation to the validation data set. The default value is 30 percent

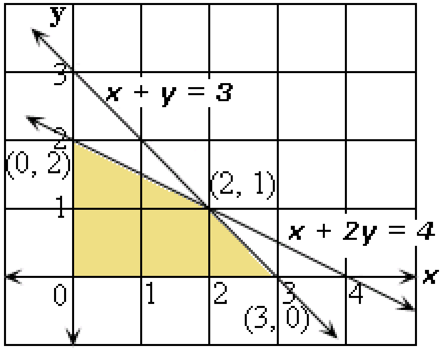
## What will occur?

- Marketing is the target for many predictive analytics applications.
- Descriptive analytics, such as data visualization, is important in helping users interpret the output from predictive and prescriptive analytics.

# Prescriptive Analytics

- Prescriptive analytics are often referred to as advanced analytics.
- Often for the allocation of scarce resources
- Optimization





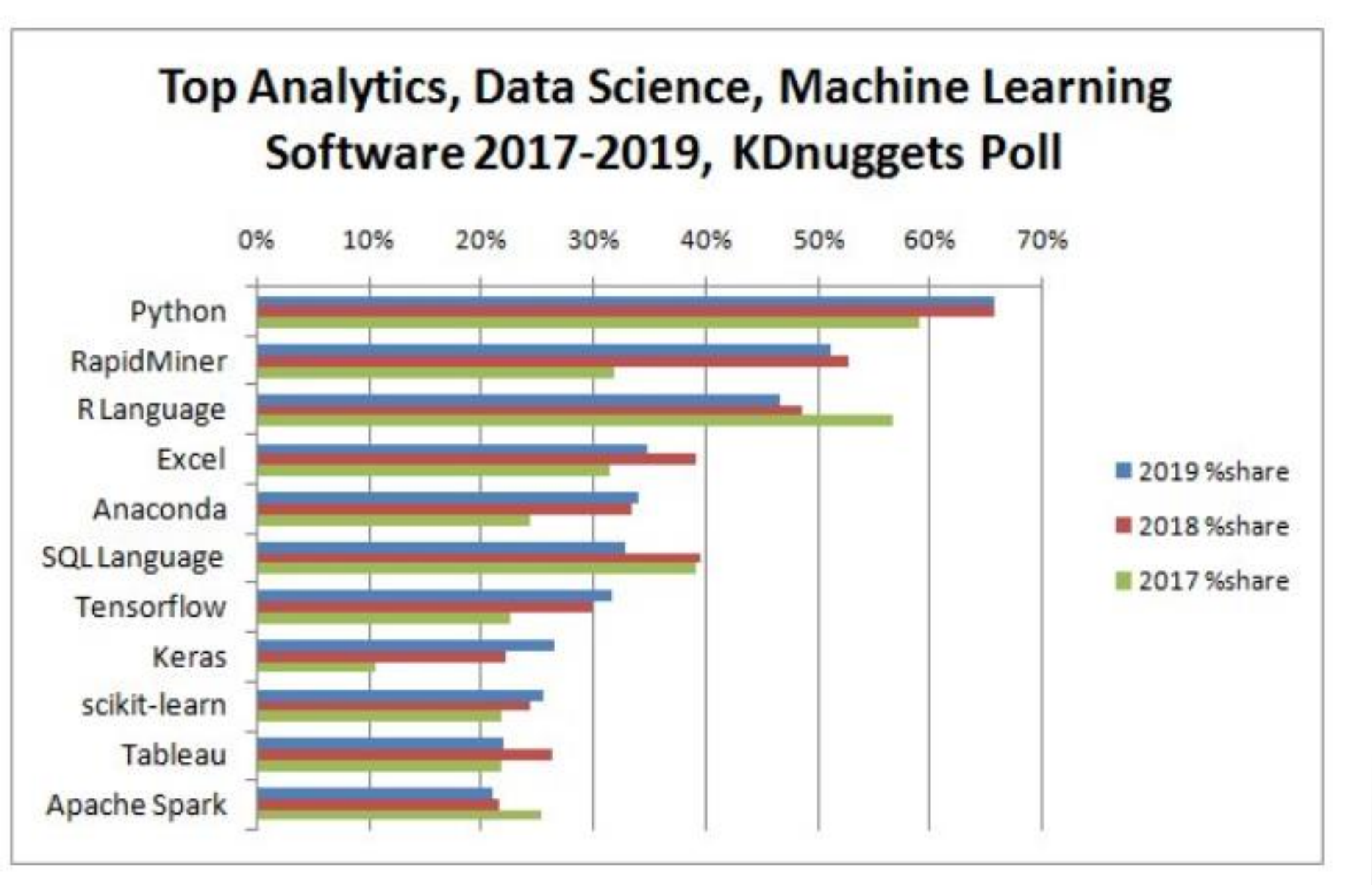
What should occur?

Prescriptive analytics can benefit healthcare strategic planning by using analytics to leverage operational and usage data combined with data of external factors such as economic data, population demographic trends and population health trends, to more accurately plan for future capital investments such as new facilities and equipment utilization as well as understand the trade-offs between adding additional beds and expanding an existing facility versus building a new one.

# Top 10 Data Analytics Tools

- Python
- R Programming
- Tableau Public
- SAS
- Apache Spark
- Excel
- RapidMiner
- Knime
- Qlikview
- Splunk

# Comparison of Data Analytics Tool



# Why Python for Data Analytics is right choice?

- Easy to learn
  - The syntax of Python is what makes it appealing. It is easy-to-write and read.
- Data Science Libraries
  - The popular data science libraries include NumPy, Pandas, SciPy, and Matplotlib.
- Scalable
  - This means that you can implement highly-scalable solutions using Python. It is crucial for projects that rely heavily on scalability and real-time data.
- Python Community
  - A great community means it is easy to find solutions. It is also easy to find mentors and coding partners.
- Easy to debug.
- Great online source material.
- Open source.

# How Python is used in Data Science?

- It is easy to get overboard with the Python programming language. If you are new, you do not need to master Python straight away. All you need to do is, **learn Python enough** to use it to solve data science problems.
- Initially, you need to **get access to data.** This will help you to do simple operations on the data using popular Python libraries such as **Pandas and Numpy.**
- To improve your analysis, you also need to **visualize the data**. To do so, simply use visualization libraries such as **Seaborn or Matplotlib**.
- With all the data collected and visualized, you now need to use **machine learning** to compute the data

# How to learn Python for Data Science?

**1. Learn Python Fundamentals**

**2. Do small Python projects**

**3. Learn Data Science Libraries**

**4. Build and Explore**

**5. Teach Data Science**

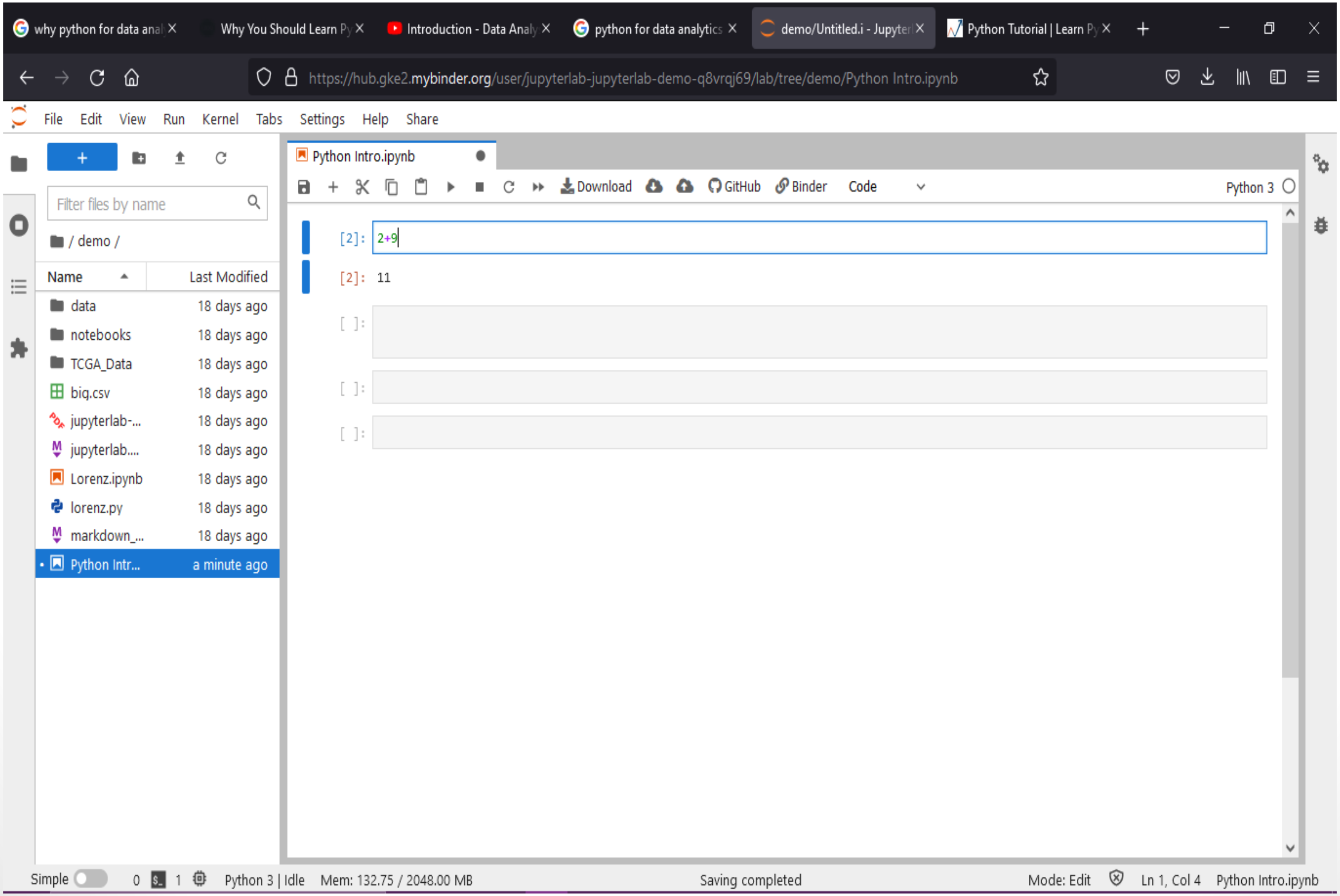
**6. Learn advanced data science techniques**

# Online Python Jupiter

- Type “Jupiter notebook python” in google
- Click the link https://jupyter.org
- Scroll down the link https://jupyter.org and click “Try it in your Browser”
- Click “Try Jupyter Lab”
- Wait for 1 or 2 seconds for the binder to gets load into the browser
- In notebook Click on “Python 3”
- Now the platform is ready to explore “Python”
- Other option is that we can install Spyder IDE / Pycharm

# Online Python Jupiter

- You can execute a code by pressing “Shift + Enter” or “ALT + Enter”, if you want to insert an additional row after.



| Name | Last Modified |
|---|---|
| data | 18 days ago |
| notebooks | 18 days ago |
| TCGA_Data | 18 days ago |
| big.csv | 18 days ago |
| jupyterlab-... | 18 days ago |
| jupyterlab.... | 18 days ago |
| Lorenz.ipynb | 18 days ago |
| lorenz.py | 18 days ago |
| markdown_... | 18 days ago |
| Python Intr... | a minute ago |

```
[2]: 2+9
[2]: 11
[ ]:
[ ]:
[ ]:
```

# Python Data Structures

Following are some data structures, which are used in Python.

- **Lists** – Lists are one of the most versatile data structure in Python.
- A list can simply be defined by writing a list of comma separated values in square brackets.
- Lists might contain items of different types, but usually the items all have the same type.
- Python lists are mutable and individual elements of a list can be changed.

- **DEMO**

# Strings

- **Strings** – Strings can simply be defined by use of single ( ‘ ), double ( ” ) or triple ( ”’ ) inverted commas.
- Strings enclosed in tripe quotes ( ”’ ) can span over multiple lines and are used frequently in docstrings (Python’s way of documenting functions).
- \ is used as an escape character. Please note that Python strings are immutable, so you can not change part of strings.

- **DEMO**

# Tuples

- **Tuples** – A tuple is represented by a number of values separated by commas.
- Tuples are immutable and the output is surrounded by parentheses so that nested tuples are processed correctly.
- Additionally, even though tuples are immutable, they can hold mutable data if needed.
- Since Tuples are immutable and can not change, they are faster in processing as compared to lists.
- Hence, if your list is unlikely to change, you should use tuples, instead of lists.

**DEMO**

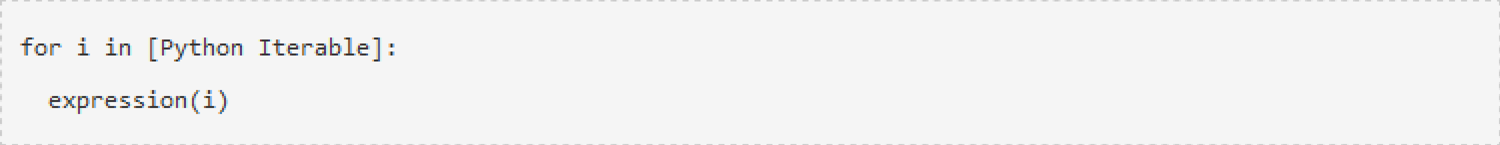
# Dictionary

- Dictionary – Dictionary is an unordered set of key: value pairs, with the requirement that the keys are unique (within one dictionary). A pair of braces creates an empty dictionary: { }.

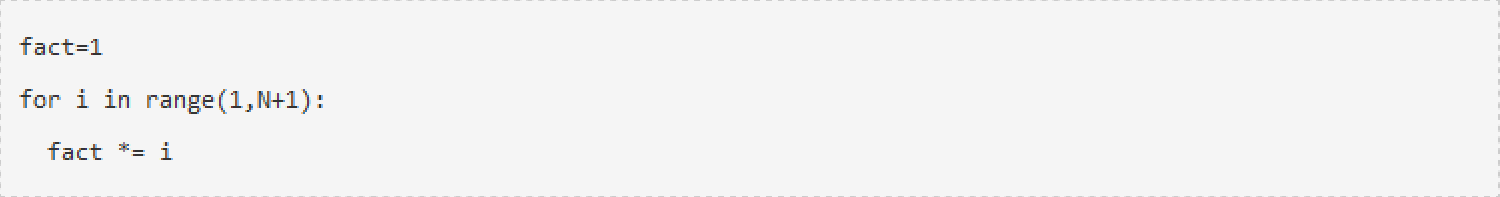
- **DEMO**

# Python – For loop

- Python  has a FOR-loop which is the most widely used method for iteration. It has a simple syntax:



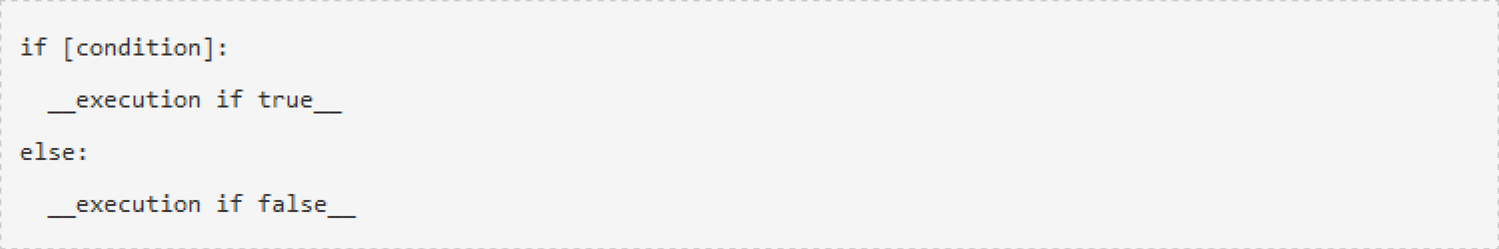
```
for i in [Python Iterable]:
  expression(i)
```



```
fact=1
for i in range(1,N+1):
  fact *= i
```

# Python –Conditional Statement

- Coming to conditional statements, these are used to execute code fragments based on a condition. The most commonly used construct is if-else, with following syntax:



```
if [condition]:
  __execution if true__
else:
  __execution if false__
```

- **DEMO**

# Python Libraries

What if you have to perform the following tasks:

1. Multiply 2 matrices
2. Plot bar charts and histograms
3. Make statistical methods

**DEMO**

# Top 10 Python Libraries – Data Scientist

**1.Pandas**

- Pandas is the most commonly used python library. It provides the users with some of the most useful set of tools to **explore, clean and analyse the data.**

- With Pandas, you can load, prepare, manipulate, and analyze all kinds of structured data. Machine learning libraries also revolve around Pandas Data Frames as an input.

**2.NumPy**

- NumPy is mainly used for **its support for N-dimensional arrays**. These multi-dimensional arrays are 50 times more robust compared to Python lists, making NumPy a favorite for data scientists. NumPy is also used by other libraries such as TensorFlow for their internal computation on tensors.

**3. Scikit-learn**

- Scikit-learn is arguably the most important library in Python for **machine learning**. After cleaning and manipulating your data with Pandas or NumPy, scikit-learn is used to build machine learning models as it has tons of tools used for predictive modelling and analysis.

# Top 10 Python Libraries – Data Scientist

**4. Gradio**

Gradio is useful for the following reasons:

- It allows for **further model validation**. Specifically, it allows you to interactively test different inputs into the model.
- It’s a good way to conduct demos.
- It’s easy to implement and distribute because the web app is accessible by anyone through a public link.

**5. TensorFlow**

- TensorFlow is one of the most popular libraries of Python for implementing **neural networks**. It uses multi-dimensional arrays, also known as tensors, which allows it to perform several operations on a particular input.

**6. Keras**

- Keras is mainly used for creating **deep learning models**, specifically neural networks. It’s built on top of TensorFlow and Theano and allows you to build neural networks very simply. Since Keras generates a computational graph using back-end infrastructure, it is relatively slow compared to other libraries.

# Top 10 Python Libraries – Data Scientist

**7. SciPy**

- As the name suggests, SciPy is mainly used for its **scientific functions and mathematical functions derived from NumPy**. Some useful functions which this library provides are stats functions, optimization functions, and signal processing functions. To solve differential equations and provide optimization, it includes functions for computing integrals numerically. Some of the applications which make SciPy important are:
  - Multi-dimensional image processing
  - Ability to solve Fourier transforms, and differential equations
  - Due to its optimized algorithms, it can do linear algebra computations very robustly and efficiently

**8. Statsmodels**

- Statsmodels is a great library for doing **hardcore statistics**. This multifunctional library is a blend of different Python libraries, taking its graphical features and functions from Matplotlib, for data handling, it uses Pandas, for handling R-like formulas, it uses Pasty, and is built on NumPy and SciPy.

- Specifically, it’s useful for creating statistical models, like OLS, and also for performing statistical tests

# Top 10 Python Libraries – Data Scientist

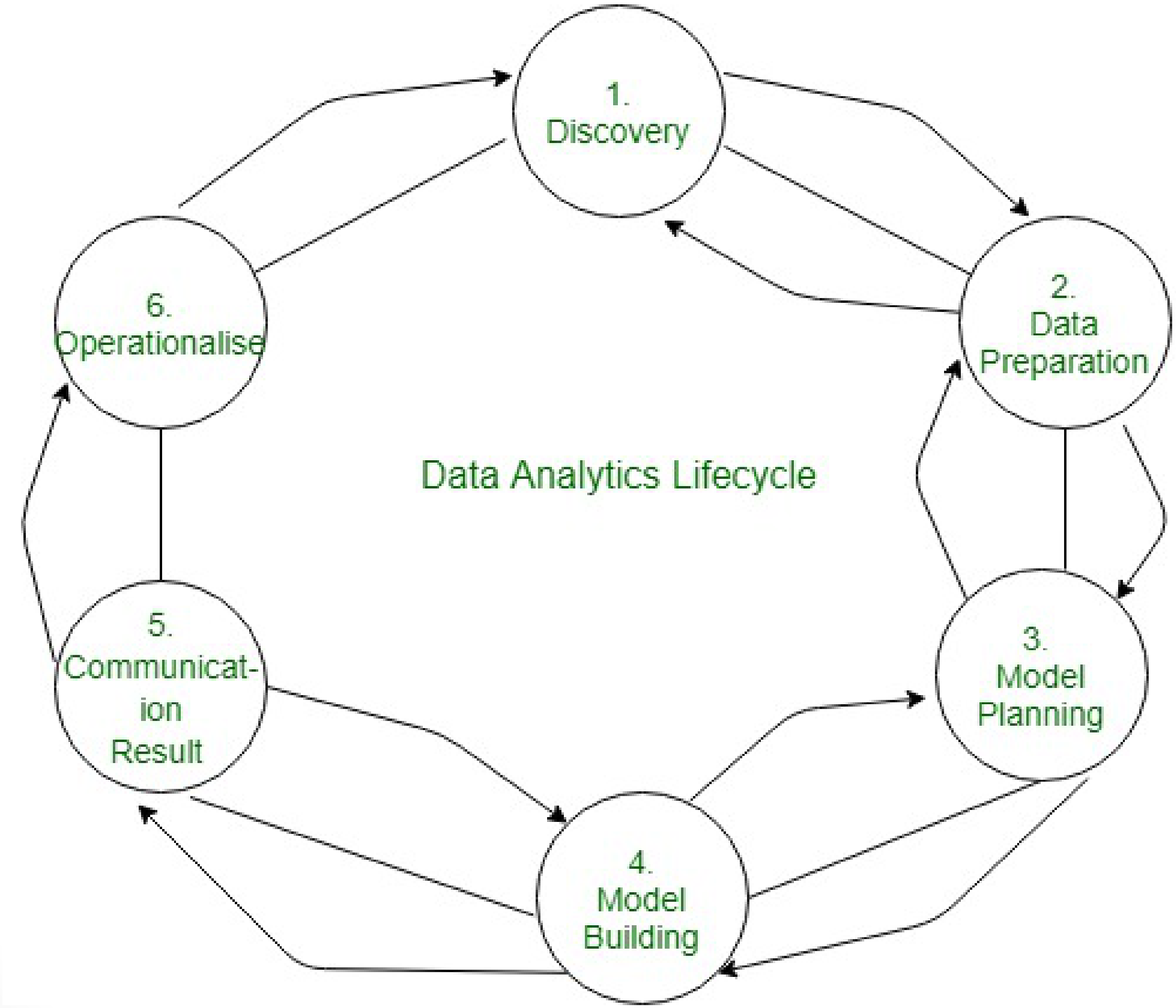
**9. Plotly**

- Plotly is definitely a must-know tool for **building visualizations** since it is extremely powerful, easy to use, and has a big benefit of being able to interact with the visualizations.
- Along with Plotly is Dash, which is a tool that allows you to build dynamic dashboards using Plotly visualizations. Dash is a web-based python interface that removes the need for JavaScript in these types of analytical web applications and allows you to run these plots online and offline.

**10. Seaborn**

- Built on the top of Matplotlib, **Seaborn is an effective library for creating different visualizations.**
- One of the most important features of Seaborn is the creation of amplified data visuals. Some of the correlations that are not obvious initially can be displayed in a visual context, allowing Data Scientists to understand the models more properly

# Data Analytics – Project Cycle



# Data Analytics – Project Cycle

- **Phase 1: Discovery –** The data science team **learn and investigate the problem.**
- Develop **context** and **understanding**.
- Come to know about data sources needed and available for the project.
- The team formulates **initial hypothesis** that can be later tested with data.

- **Phase 2: Data Preparation –** Steps to **explore, preprocess, and condition data prior to modeling and analysis**.
- It requires the presence of an analytic sandbox, the team execute, load, and transform, to get data into the sandbox.
- Data preparation tasks are likely to be performed multiple times and not in predefined order.

# Data Analytics – Project Cycle

**Phase 3: Model Planning**

- Team explores data to learn about **relationships between variables and subsequently, selects key variables and the most suitable models.**
- In this phase, data science team develop data sets for training, testing, and production purposes.
- Team builds and executes models based on the work done in the model planning phase.

**Phase 4: Model Building**

- Team develops datasets for testing, training, and production purposes.
- Team also considers whether its existing tools will suffice for running the models or if they need more robust environment for executing models.

# Data Analytics – Project Cycle

**Phase 5: Communication Results**

**After executing model team need to compare outcomes of modeling to criteria established for success and failure**.

- Team considers how best to articulate findings and outcomes to various team members and stakeholders, taking into account warning, assumptions.
- Team should identify key findings, quantify business value, and develop narrative to summarize and convey findings to stakeholders.

**Phase 6: Operationalize**

- The team **communicates benefits of project more broadly** and sets up pilot project to deploy work in controlled way before broadening the work to full enterprise of users.
- This approach enables team to learn about performance and related constraints of the model in production environment on small scale , and make adjustments before full deployment.
- The team delivers final reports, briefings, codes.

# Focus : Data Discovery

- Collect the data
- Explore the data
- Understand the data

- **DEMO**

# Focus : Data Preparation

- Forthcoming slides will emphasis more on the data preparation phase (2nd phase of data science project life cycle)
- The focus will be on the following:
  - How to handle missing values?
  - How to find the relationship between variables?
  - How to chose relevant variables for the problem chosen?

# Dealing with Missing values

There are 7 ways to handle missing values in the dataset:

1. Deleting Rows with missing values
2. Impute missing values for continuous variable
3. Impute missing values for categorical variable
4. Using Algorithms that support missing values
5. Prediction of missing values
6. Imputation using Deep Learning Library — Datawig

# Dealing with missing values

**Delete Rows with Missing Values:**

- Missing values can be handled by deleting the rows or columns having null values. If columns have more than half of the rows as null then the entire column can be dropped. The rows which are having one or more columns values as null can also be dropped.

**Pros:**

A model trained with the removal of all missing values create a robust model

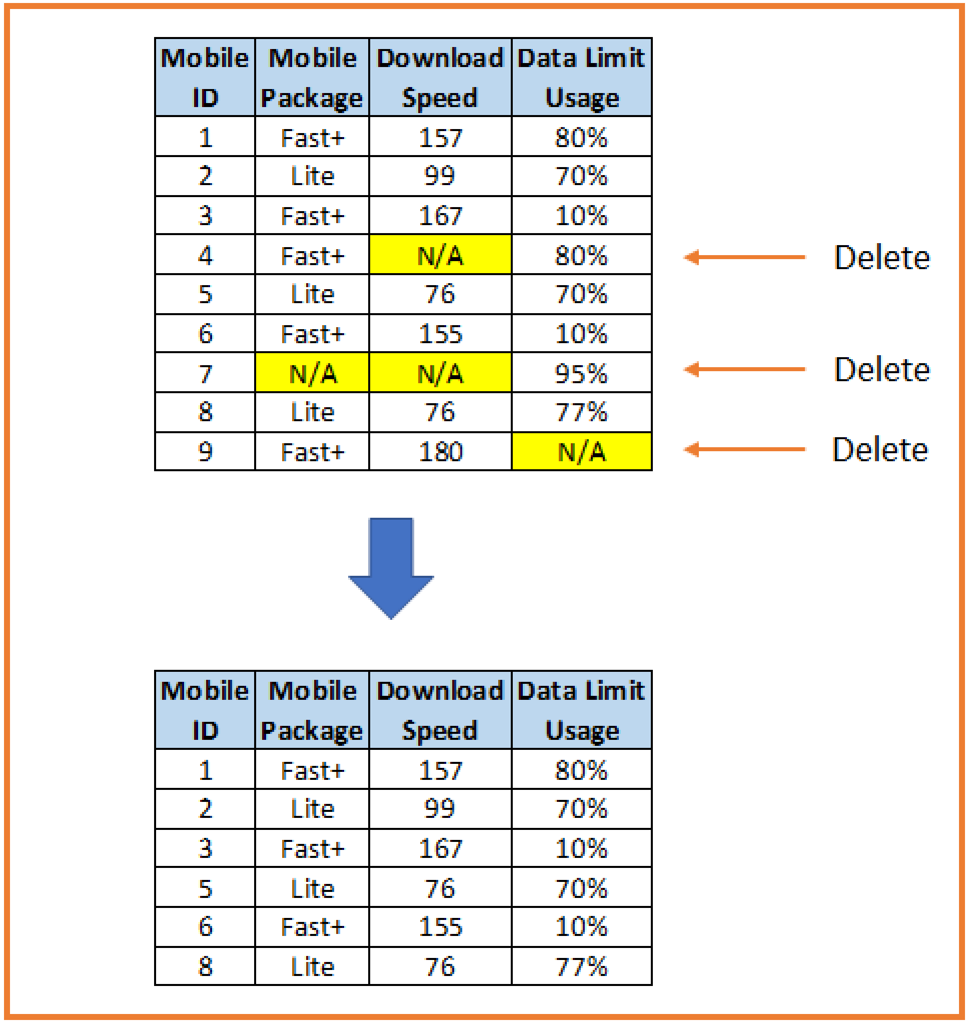
**Cons:**

Loss of lot of information

Works poorly if the percentage of missing values is excessive in comparison to the complete data set.

# Dealing with missing values

## Delete Rows with Missing Values:



| Mobile ID | Mobile Package | Download Speed | Data Limit Usage |
|---|---|---|---|
| 1 | Fast+ | 157 | 80% |
| 2 | Lite | 99 | 70% |
| 3 | Fast+ | 167 | 10% |
| 4 | Fast+ | N/A | 80% |
| 5 | Lite | 76 | 70% |
| 6 | Fast+ | 155 | 10% |
| 7 | N/A | N/A | 95% |
| 8 | Lite | 76 | 77% |
| 9 | Fast+ | 180 | N/A |

Delete
Delete
Delete

| Mobile ID | Mobile Package | Download Speed | Data Limit Usage |
|---|---|---|---|
| 1 | Fast+ | 157 | 80% |
| 2 | Lite | 99 | 70% |
| 3 | Fast+ | 167 | 10% |
| 5 | Lite | 76 | 70% |
| 6 | Fast+ | 155 | 10% |
| 8 | Lite | 76 | 77% |

# Impute missing values with Mean / Median / Mode

- Columns in the dataset which are having numeric continuous values can be replaced with mean, median or mode of remaining values in the column.
- This method can prevent the loss of data compared to earlier method.
- Replacing the above two approximation is a statistical approach to handle the missing values.
- The missing values are replaced by the mean value in the above example, in the same way, it can be replaced by the median value.

# Impute missing values with Mean / Median / Mode

- **Pros**

Prevent data loss which results in deletion of rows or columns

Works well with a small dataset and is easy to implement

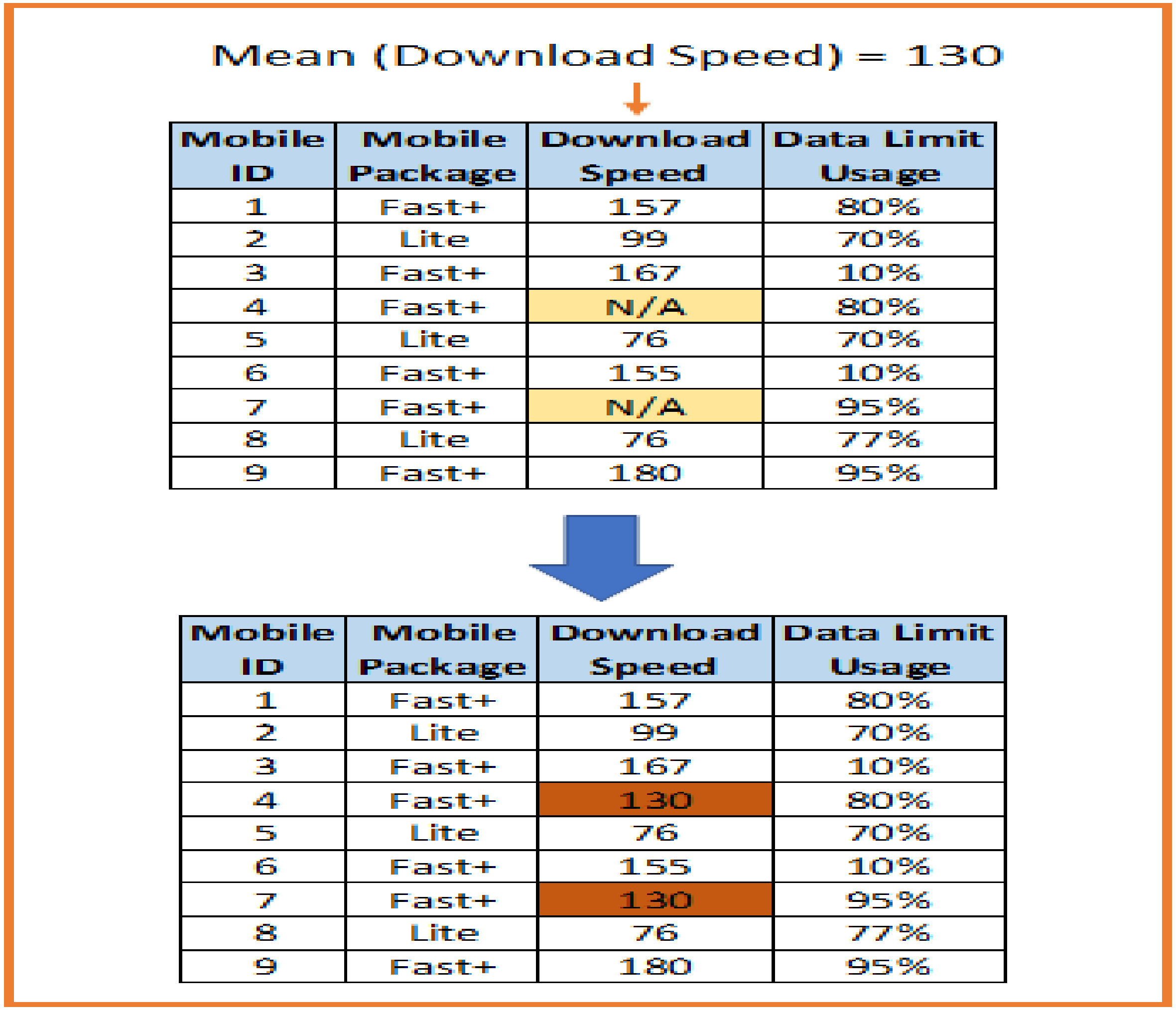
- **Cons**

Works only with numerical continuous variables

Can cause data leakage

Do not factor the covariance between features

# Impute missing values with Mean / Median / Mode

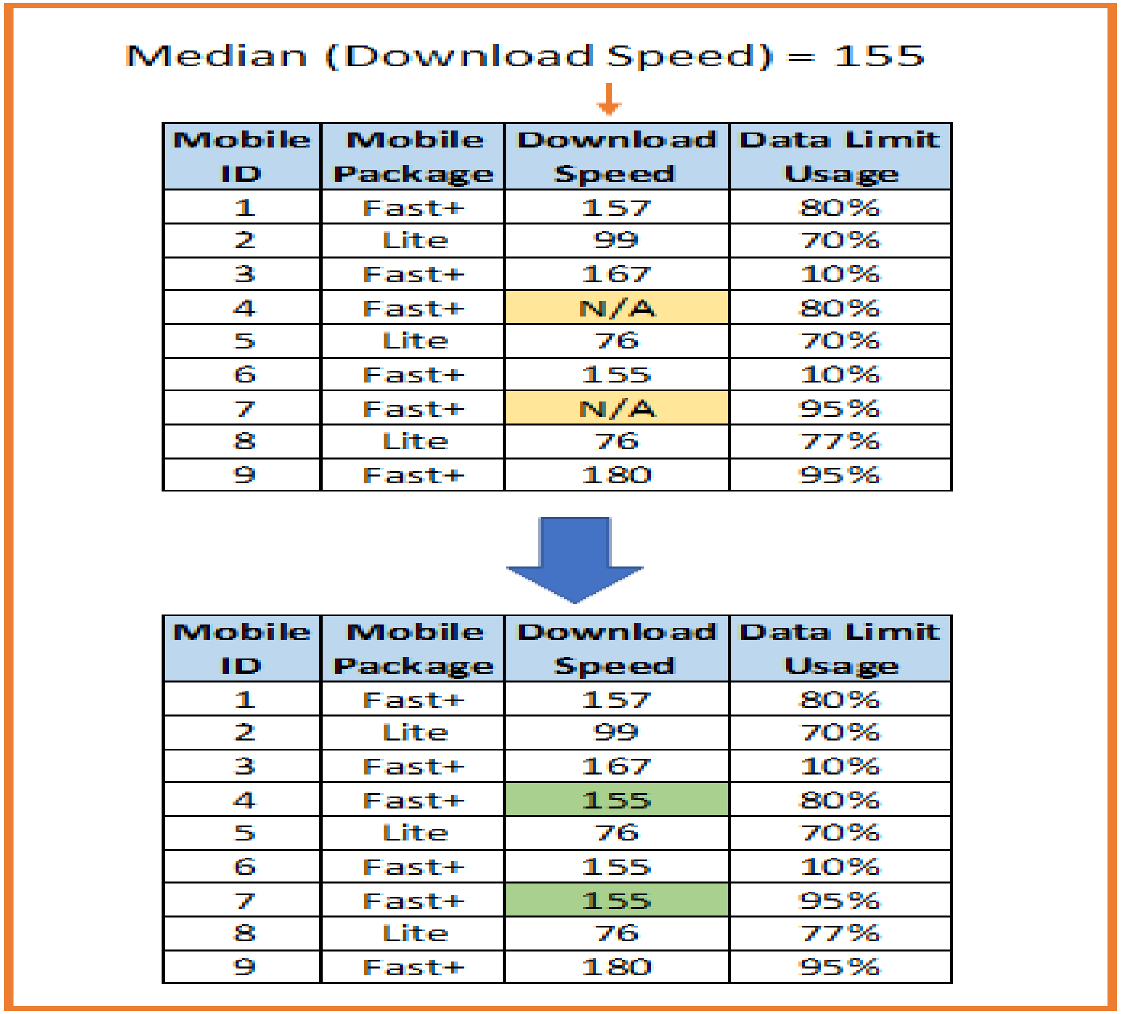


Mean (Download Speed) = 130

| Mobile ID | Mobile Package | Download Speed | Data Limit Usage |
|---|---|---|---|
| 1 | Fast+ | 157 | 80% |
| 2 | Lite | 99 | 70% |
| 3 | Fast+ | 167 | 10% |
| 4 | Fast+ | N/A | 80% |
| 5 | Lite | 76 | 70% |
| 6 | Fast+ | 155 | 10% |
| 7 | Fast+ | N/A | 95% |
| 8 | Lite | 76 | 77% |
| 9 | Fast+ | 180 | 95% |

| Mobile ID | Mobile Package | Download Speed | Data Limit Usage |
|---|---|---|---|
| 1 | Fast+ | 157 | 80% |
| 2 | Lite | 99 | 70% |
| 3 | Fast+ | 167 | 10% |
| 4 | Fast+ | 130 | 80% |
| 5 | Lite | 76 | 70% |
| 6 | Fast+ | 155 | 10% |
| 7 | Fast+ | 130 | 95% |
| 8 | Lite | 76 | 77% |
| 9 | Fast+ | 180 | 95% |

# Impute missing values with Mean / Median / Mode

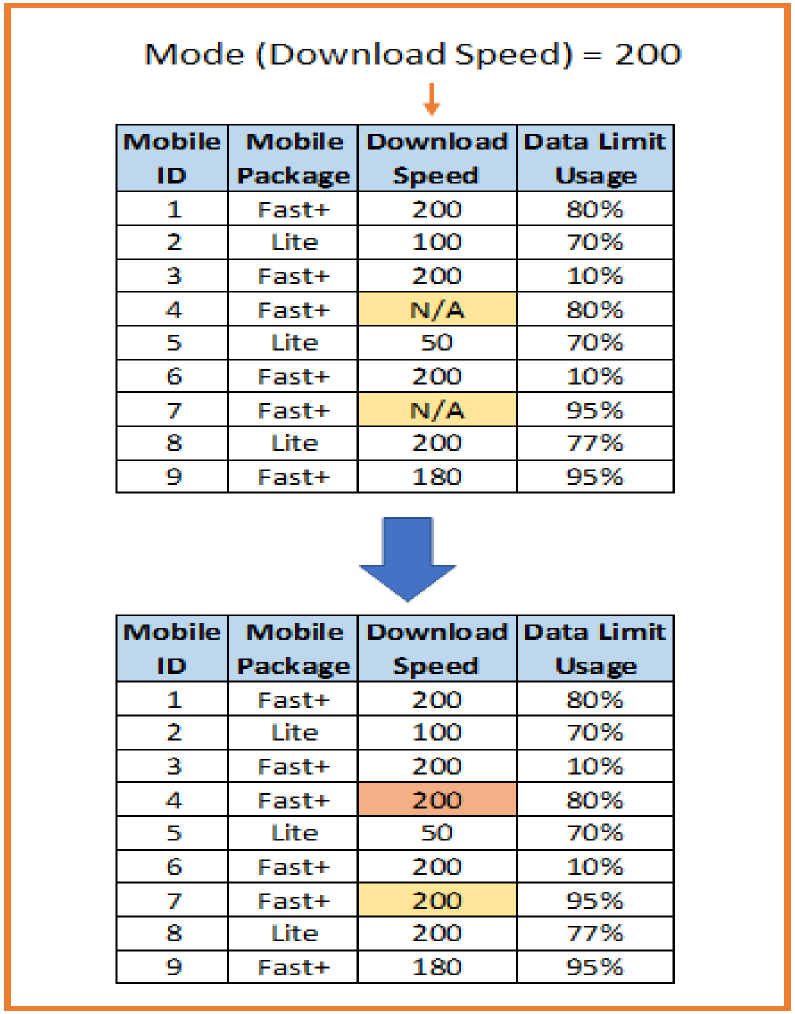


Median (Download Speed) = 155

| Mobile ID | Mobile Package | Download Speed | Data Limit Usage |
|---|---|---|---|
| 1 | Fast+ | 157 | 80% |
| 2 | Lite | 99 | 70% |
| 3 | Fast+ | 167 | 10% |
| 4 | Fast+ | N/A | 80% |
| 5 | Lite | 76 | 70% |
| 6 | Fast+ | 155 | 10% |
| 7 | Fast+ | N/A | 95% |
| 8 | Lite | 76 | 77% |
| 9 | Fast+ | 180 | 95% |

| Mobile ID | Mobile Package | Download Speed | Data Limit Usage |
|---|---|---|---|
| 1 | Fast+ | 157 | 80% |
| 2 | Lite | 99 | 70% |
| 3 | Fast+ | 167 | 10% |
| 4 | Fast+ | 155 | 80% |
| 5 | Lite | 76 | 70% |
| 6 | Fast+ | 155 | 10% |
| 7 | Fast+ | 155 | 95% |
| 8 | Lite | 76 | 77% |
| 9 | Fast+ | 180 | 95% |

# Impute missing values with Mean / Median / Mode



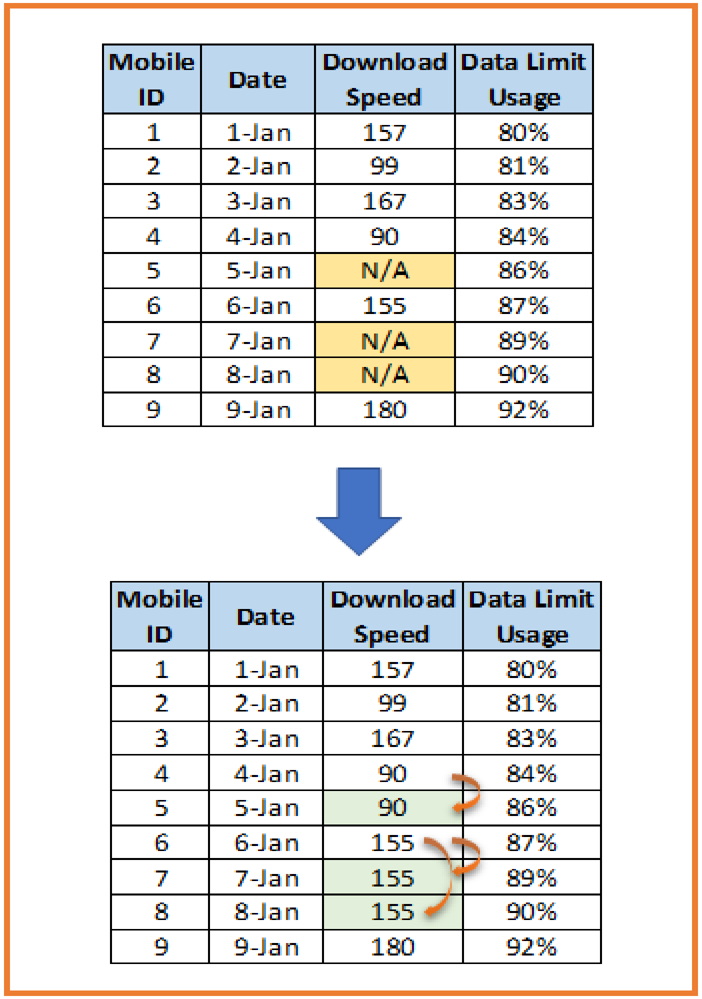
Mode (Download Speed) = 200

| Mobile ID | Mobile Package | Download Speed | Data Limit Usage |
|---|---|---|---|
| 1 | Fast+ | 200 | 80% |
| 2 | Lite | 100 | 70% |
| 3 | Fast+ | 200 | 10% |
| 4 | Fast+ | N/A | 80% |
| 5 | Lite | 50 | 70% |
| 6 | Fast+ | 200 | 10% |
| 7 | Fast+ | N/A | 95% |
| 8 | Lite | 200 | 77% |
| 9 | Fast+ | 180 | 95% |

| Mobile ID | Mobile Package | Download Speed | Data Limit Usage |
|---|---|---|---|
| 1 | Fast+ | 200 | 80% |
| 2 | Lite | 100 | 70% |
| 3 | Fast+ | 200 | 10% |
| 4 | Fast+ | 200 | 80% |
| 5 | Lite | 50 | 70% |
| 6 | Fast+ | 200 | 10% |
| 7 | Fast+ | 200 | 95% |
| 8 | Lite | 200 | 77% |
| 9 | Fast+ | 180 | 95% |

# Last Observation Carried Forward (LOCF)

- If data is time-series data, one of the most widely used imputation methods is the last observation carried forward (LOCF).

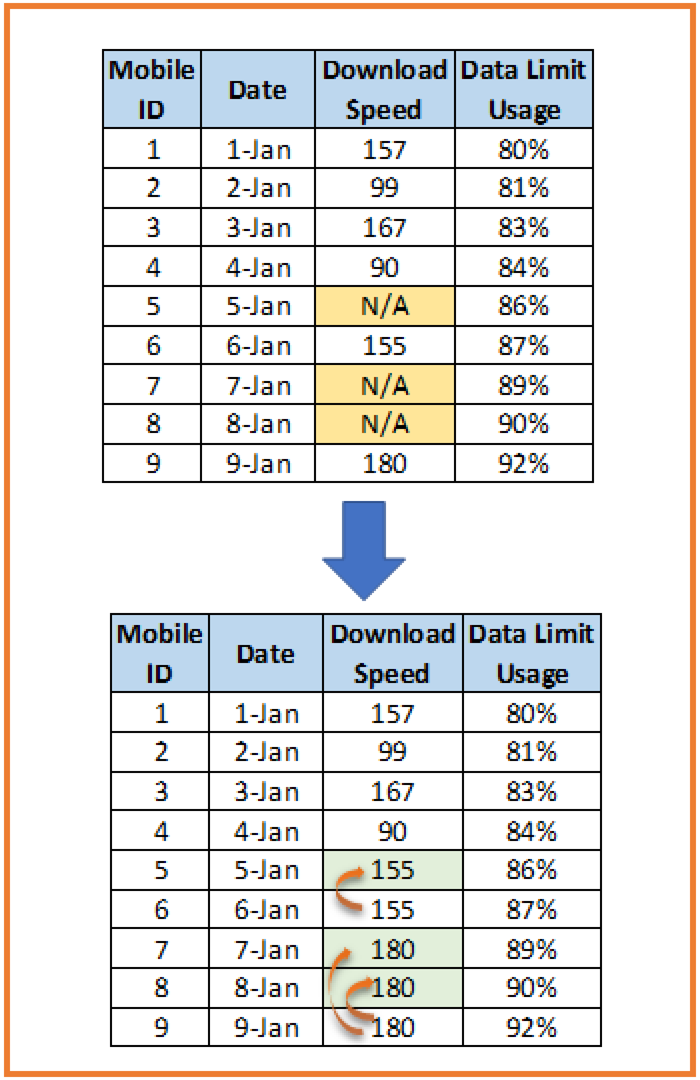


| Mobile ID | Date | Download Speed | Data Limit Usage |
|---|---|---|---|
| 1 | 1-Jan | 157 | 80% |
| 2 | 2-Jan | 99 | 81% |
| 3 | 3-Jan | 167 | 83% |
| 4 | 4-Jan | 90 | 84% |
| 5 | 5-Jan | N/A | 86% |
| 6 | 6-Jan | 155 | 87% |
| 7 | 7-Jan | N/A | 89% |
| 8 | 8-Jan | N/A | 90% |
| 9 | 9-Jan | 180 | 92% |

| Mobile ID | Date | Download Speed | Data Limit Usage |
|---|---|---|---|
| 1 | 1-Jan | 157 | 80% |
| 2 | 2-Jan | 99 | 81% |
| 3 | 3-Jan | 167 | 83% |
| 4 | 4-Jan | 90 | 84% |
| 5 | 5-Jan | 90 | 86% |
| 6 | 6-Jan | 155 | 87% |
| 7 | 7-Jan | 155 | 89% |
| 8 | 8-Jan | 155 | 90% |
| 9 | 9-Jan | 180 | 92% |

# Next Observation Carried Backward (NOCB)

A similar approach like LOCF works oppositely by taking the first observation after the missing value and carrying it backward

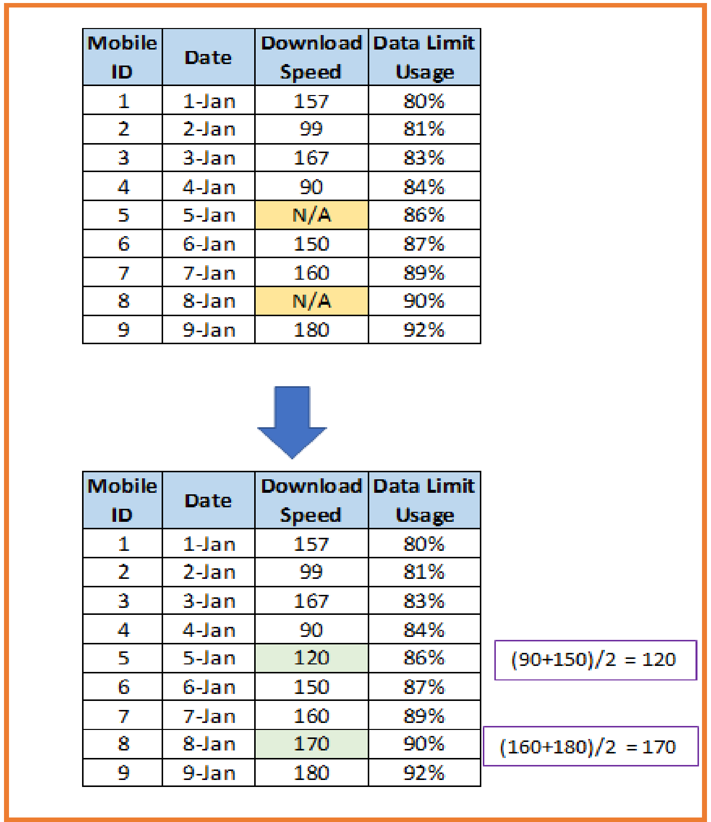


| Mobile ID | Date | Download Speed | Data Limit Usage |
|---|---|---|---|
| 1 | 1-Jan | 157 | 80% |
| 2 | 2-Jan | 99 | 81% |
| 3 | 3-Jan | 167 | 83% |
| 4 | 4-Jan | 90 | 84% |
| 5 | 5-Jan | N/A | 86% |
| 6 | 6-Jan | 155 | 87% |
| 7 | 7-Jan | N/A | 89% |
| 8 | 8-Jan | N/A | 90% |
| 9 | 9-Jan | 180 | 92% |

| Mobile ID | Date | Download Speed | Data Limit Usage |
|---|---|---|---|
| 1 | 1-Jan | 157 | 80% |
| 2 | 2-Jan | 99 | 81% |
| 3 | 3-Jan | 167 | 83% |
| 4 | 4-Jan | 90 | 84% |
| 5 | 5-Jan | 155 | 86% |
| 6 | 6-Jan | 155 | 87% |
| 7 | 7-Jan | 180 | 89% |
| 8 | 8-Jan | 180 | 90% |
| 9 | 9-Jan | 180 | 92% |

# Linear Interpolation

- Interpolation is a mathematical method that adjusts a function to data and uses this function to extrapolate the missing data. The simplest type of interpolation is linear interpolation, which means between the values before the missing data and the value.
- Just in Pandas, we have the following options like: ‘linear’, ‘time’, ‘index’, ‘values’, ‘nearest’, ‘zero’, ‘slinear’, ‘quadratic’, ‘cubic’, ‘polynomial’, ‘spline’, ‘piecewise polynomial’ and many more.



| Mobile ID | Date | Download Speed | Data Limit Usage |
|---|---|---|---|
| 1 | 1-Jan | 157 | 80% |
| 2 | 2-Jan | 99 | 81% |
| 3 | 3-Jan | 167 | 83% |
| 4 | 4-Jan | 90 | 84% |
| 5 | 5-Jan | N/A | 86% |
| 6 | 6-Jan | 150 | 87% |
| 7 | 7-Jan | 160 | 89% |
| 8 | 8-Jan | N/A | 90% |
| 9 | 9-Jan | 180 | 92% |

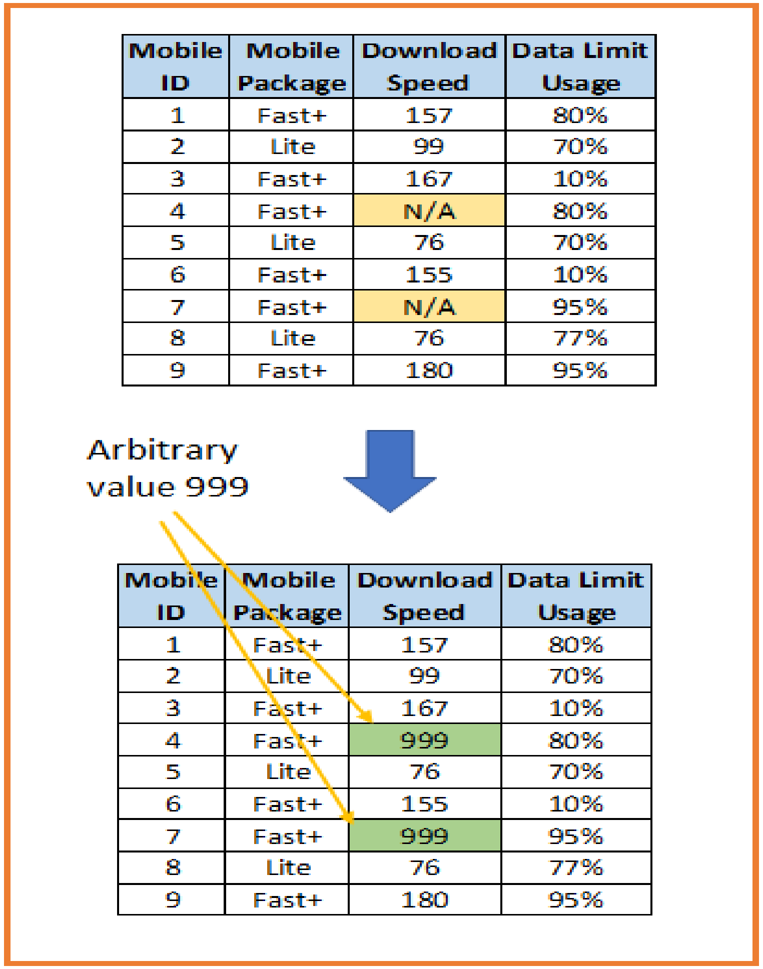
| Mobile ID | Date | Download Speed | Data Limit Usage |
|---|---|---|---|
| 1 | 1-Jan | 157 | 80% |
| 2 | 2-Jan | 99 | 81% |
| 3 | 3-Jan | 167 | 83% |
| 4 | 4-Jan | 90 | 84% |
| 5 | 5-Jan | 120 | 86% |
| 6 | 6-Jan | 150 | 87% |
| 7 | 7-Jan | 160 | 89% |
| 8 | 8-Jan | 170 | 90% |
| 9 | 9-Jan | 180 | 92% |

(90+150)/2 = 120

(160+180)/2 = 170

# Arbitrary Value Imputation

Arbitrary value imputation consists of replacing all occurrences of missing values within a variable with an arbitrary value. Ideally, the arbitrary value should be different from the median/mean/mode and not within the normal values of the variable. Typically used arbitrary values are 0, 999, -999 (or other combinations of 9's) or -1 (if the distribution is positive).



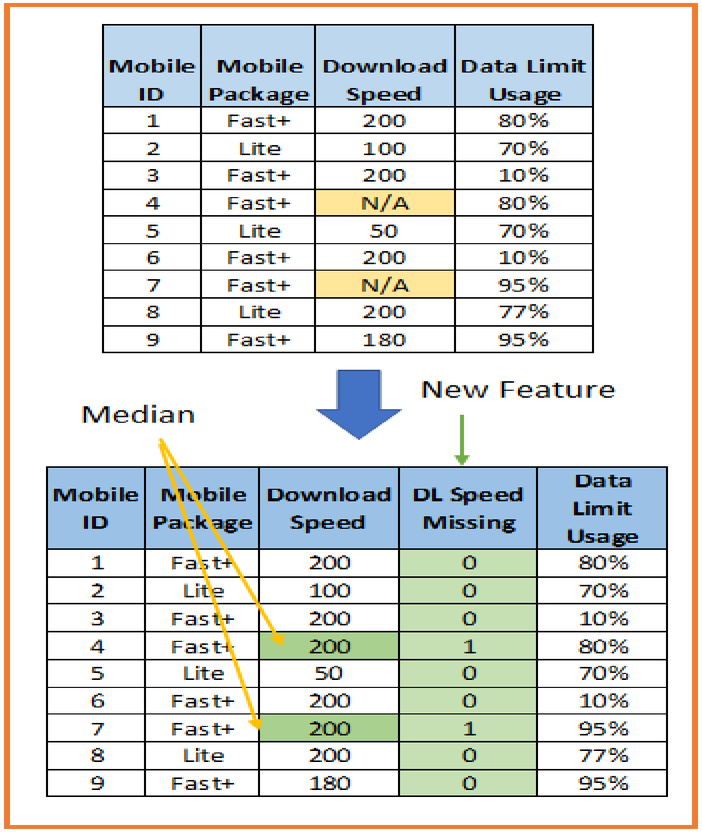
| Mobile ID | Mobile Package | Download Speed | Data Limit Usage |
|---|---|---|---|
| 1 | Fast+ | 157 | 80% |
| 2 | Lite | 99 | 70% |
| 3 | Fast+ | 167 | 10% |
| 4 | Fast+ | N/A | 80% |
| 5 | Lite | 76 | 70% |
| 6 | Fast+ | 155 | 10% |
| 7 | Fast+ | N/A | 95% |
| 8 | Lite | 76 | 77% |
| 9 | Fast+ | 180 | 95% |

Arbitrary value 999

| Mobile ID | Mobile Package | Download Speed | Data Limit Usage |
|---|---|---|---|
| 1 | Fast+ | 157 | 80% |
| 2 | Lite | 99 | 70% |
| 3 | Fast+ | 167 | 10% |
| 4 | Fast+ | 999 | 80% |
| 5 | Lite | 76 | 70% |
| 6 | Fast+ | 155 | 10% |
| 7 | Fast+ | 999 | 95% |
| 8 | Lite | 76 | 77% |
| 9 | Fast+ | 180 | 95% |

# Adding a variable to capture NA

- When data are not missing completely at random, we can capture the importance of missingness by creating an additional variable indicating whether the data was missing for that observation (1) or not (0).
- The additional variable is binary: it takes only the values 0 and 1, 0 indicating that a value was present for that observation, and 1 indicating that the value was missing.



| Mobile ID | Mobile Package | Download Speed | Data Limit Usage |
|---|---|---|---|
| 1 | Fast+ | 200 | 80% |
| 2 | Lite | 100 | 70% |
| 3 | Fast+ | 200 | 10% |
| 4 | Fast+ | N/A | 80% |
| 5 | Lite | 50 | 70% |
| 6 | Fast+ | 200 | 10% |
| 7 | Fast+ | N/A | 95% |
| 8 | Lite | 200 | 77% |
| 9 | Fast+ | 180 | 95% |

New Feature
Median

| Mobile ID | Mobile Package | Download Speed | DL Speed Missing | Data Limit Usage |
|---|---|---|---|---|
| 1 | Fast+ | 200 | 0 | 80% |
| 2 | Lite | 100 | 0 | 70% |
| 3 | Fast+ | 200 | 0 | 10% |
| 4 | Fast+ | 200 | 1 | 80% |
| 5 | Lite | 50 | 0 | 70% |
| 6 | Fast+ | 200 | 0 | 10% |
| 7 | Fast+ | 200 | 1 | 95% |
| 8 | Lite | 200 | 0 | 77% |
| 9 | Fast+ | 180 | 0 | 95% |

# Imputation method for categorical columns

- When missing values is from categorical columns (string or numerical) then the missing values can be replaced with the most frequent category.
- If the number of missing values is very large then it can be replaced with a new category.

**Pros:**

Prevent data loss which results in deletion of rows or columns

Works well with a small dataset and is easy to implement

Negates the loss of data by adding a unique category

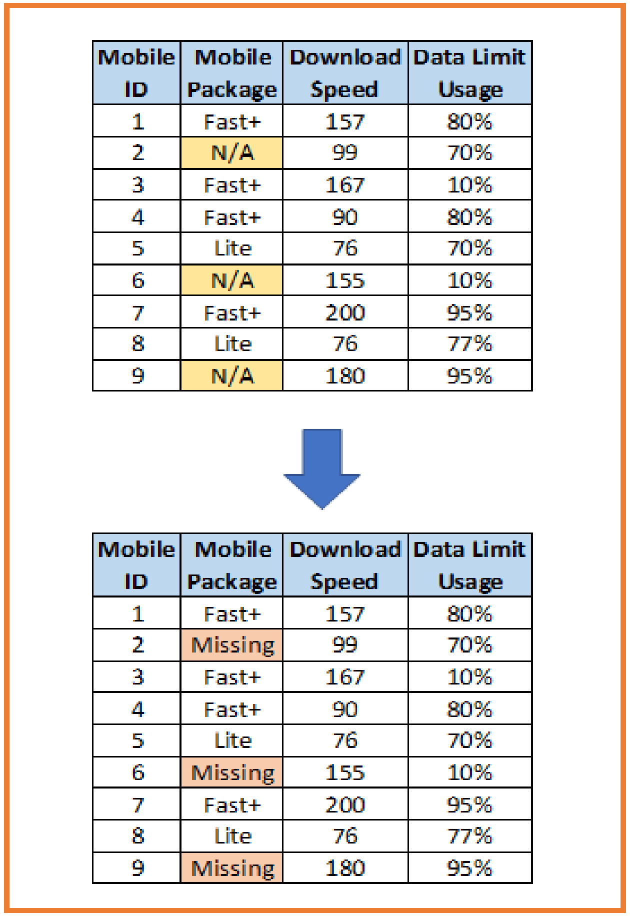
**Cons:**

Works only with categorical variables

Addition of new features to the model while encoding, which may result in poor performance.

# Adding a category to capture NA

- This is perhaps the most widely used method of missing data imputation for categorical variables.
- This method consists of treating missing data as an additional label or category of the variable. All the missing observations are grouped in the newly created label ‘Missing’

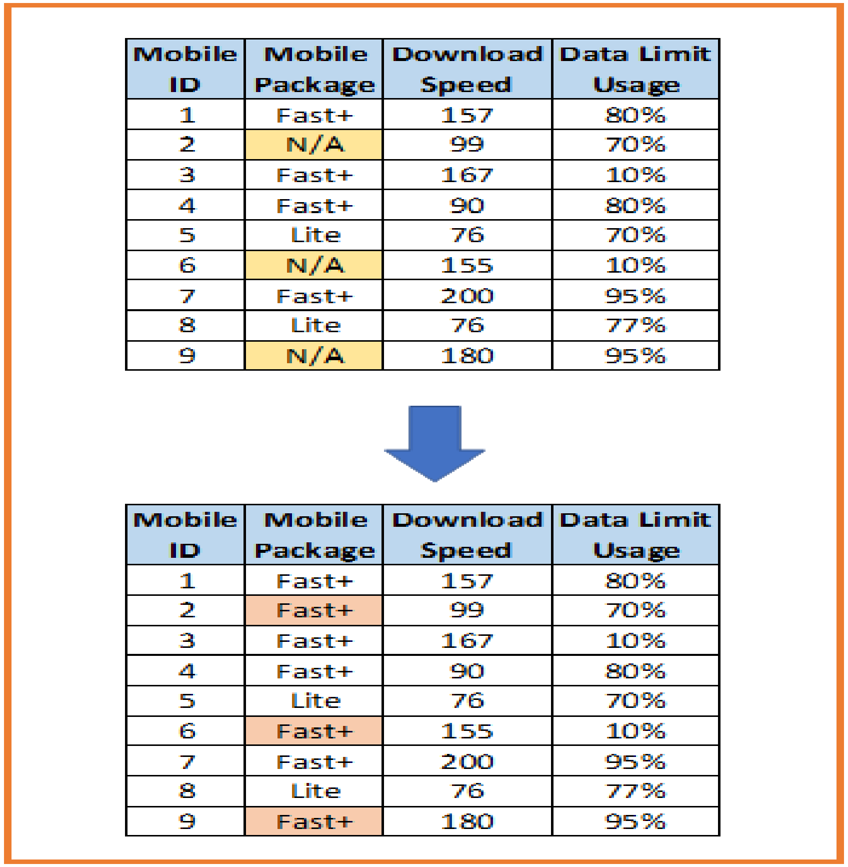


| Mobile ID | Mobile Package | Download Speed | Data Limit Usage |
|---|---|---|---|
| 1 | Fast+ | 157 | 80% |
| 2 | N/A | 99 | 70% |
| 3 | Fast+ | 167 | 10% |
| 4 | Fast+ | 90 | 80% |
| 5 | Lite | 76 | 70% |
| 6 | N/A | 155 | 10% |
| 7 | Fast+ | 200 | 95% |
| 8 | Lite | 76 | 77% |
| 9 | N/A | 180 | 95% |

| Mobile ID | Mobile Package | Download Speed | Data Limit Usage |
|---|---|---|---|
| 1 | Fast+ | 157 | 80% |
| 2 | Missing | 99 | 70% |
| 3 | Fast+ | 167 | 10% |
| 4 | Fast+ | 90 | 80% |
| 5 | Lite | 76 | 70% |
| 6 | Missing | 155 | 10% |
| 7 | Fast+ | 200 | 95% |
| 8 | Lite | 76 | 77% |
| 9 | Missing | 180 | 95% |

# Frequent category imputation

- Replacement of missing values by the most frequent category is the equivalent of mean/median imputation. It consists of replacing all occurrences of missing values within a variable with the variable's most frequent label or category.



| Mobile ID | Mobile Package | Download Speed | Data Limit Usage |
|---|---|---|---|
| 1 | Fast+ | 157 | 80% |
| 2 | N/A | 99 | 70% |
| 3 | Fast+ | 167 | 10% |
| 4 | Fast+ | 90 | 80% |
| 5 | Lite | 76 | 70% |
| 6 | N/A | 155 | 10% |
| 7 | Fast+ | 200 | 95% |
| 8 | Lite | 76 | 77% |
| 9 | N/A | 180 | 95% |

| Mobile ID | Mobile Package | Download Speed | Data Limit Usage |
|---|---|---|---|
| 1 | Fast+ | 157 | 80% |
| 2 | Fast+ | 99 | 70% |
| 3 | Fast+ | 167 | 10% |
| 4 | Fast+ | 90 | 80% |
| 5 | Lite | 76 | 70% |
| 6 | Fast+ | 155 | 10% |
| 7 | Fast+ | 200 | 95% |
| 8 | Lite | 76 | 77% |
| 9 | Fast+ | 180 | 95% |

# Random Sampling Imputation

- Random sampling imputation is in principle similar to mean/median imputation because it aims to preserve the statistical parameters of the original variable, for which data is missing.
- Random sampling consists of taking a random observation from the pool of available observations and using that randomly extracted value to fill the NA.
- In Random Sampling, one takes as many random observations as missing values are present in the variable. Random sample imputation assumes that the data are missing completely at random (MCAR).
- *If this is the case, it makes sense to substitute the missing values with values extracted from the original variable distribution.*

# Using Algorithms that support missing values

- All the machine learning algorithms don’t support missing values but some ML algorithms are robust to missing values in the dataset.
- The k-NN algorithm can be used to substitute value when data is missing.
- Naïve Bayes can also support missing values when making a prediction.
- These algorithm can be used when the dataset contains null or missing values.
- Another algorithm that can be used here in Random Forest that works well on non-linear and categorical data.
- It adapts to the data structure taking into consideration the high variance or the bias, producing better results on large datasets.

# Using Algorithms that support missing values

- **Pros**

No need to handle missing values in each column as ML algorithms will handle them efficiently.

- **Cons**

No implementation of these ML algorithms in the scikit-learn library

# Prediction of missing values:

- In the earlier methods to handle missing values, we do not use the correlation advantage of the variable containing the missing value and other variables.
- Using the other features which don’t have nulls can be used to predict missing values.
- The regression or classification model can be used for the prediction of missing values depending on the nature of the feature having missing value.

**Pros:**

Gives a better result than earlier methods

Take into account the covariance between the missing value column and other columns

**Cons:**

Considered only as a proxy for the true values.

# Imputation using Deep Learning Library – Datawig

- This method works very well with categorical, continuous and non-numerical features.
- Datawig is library that learns ML models using Deep Neural Networks to impute missing values in the datagram.
- Datawig can take a data frame and fit an imputation model for each column with missing values, with all other columns as inputs.

**Pros**

- Quite accurate compared to other methods
- It supports CPUs and GPUs

**Cons**

- Can be quite slow with large datasets

# Other phases of Data Science life cycle

- Model Planning / Model Building / Communicating Results : These phases of data science life cycle will be discussed extensively in forthcoming days of this FDP.
- Here we touched the above topics with help of demonstration of small project along with first two phases of data science life cycle.

# Summary

- Data analytics and its importance
- Why Python for data analytics?
- Demonstration of basic of python
- Data analytics life cycle
- Discussion of different phases of data analytics through small projects
- Demonstration of small data analytics project

In the afternoon session we will take sample datasets and we will work on it.

# References

- **Python Data Science Handbook**, J. VanderPlas, **O'Reilly** (2016)
- **Python for Data Analysis** by Wes McKinney (2012)
- **Introduction to Data Science , A Python Approach to Concepts, Techniques and Applications Igual**, Laura, **Seguí**, Santi (2017)
- Other web resources

**Acknowledgment**

The diagrams and the content in the presentation is/are from various sources in the Internet.

# Q&A

?

# Thank You !

You can reach me @
thiyagu@cutn.ac.in
Mobile : 0 9488584015 / 0 9841234510 (W)