Diploma in

IT, Networking and Cloud

Module 5

Business Data Analytics methods and tools

Theory Manual

Learning Outcomes

After completing this module, a student will be able to:

1. Understand business analytics and develop business intelligence.
2. Introduction to descriptive statistics and inferential statistics, measure of central tendency and spread.
3. Types of distributions-uniform, binomial, normal, log, exp
4. Sampling techniques, population
5. Introduction to business analytics and Concepts of business analytics
6. Trends in business analytics
7. Introduction to Big Data Analytics
8. Probability Theories
9. Bayes' Theorem, Maximum Likelihood
10. Hypothesis Testing
11. Central limit theorem
12. Chi-square test
13. Data Analytics using Python
14. Machine learning and its types & applications
15. Supervised machine learning techniques
16. Classification vs regression
17. Understanding Regression and types
18. Linear regression using OLS
19. Multi-Variate Linear Regression
20. Correlation concepts
21. Metrics- Loss function, MSE, RMSE, MAE, R2 Score

Introduction to business analytics and Concepts of business analytics

What are Business Analytics?

Business Analytics is "the study of data through statistical and operations analysis, the formation of predictive models, application of optimization techniques, and the communication of these results to customers, business partners, and college executives". Business Analytics requires quantitative methods and evidence-based data for business modelling and decision making; as such, Business Analytics requires the use of Big Data.

Business Analytics is a combination of Data Analytics, Business Intelligence and Computer Programming. It is the science of analysing data to find out patterns that will be helpful in developing strategies. Its usage can be found in almost every industry.

Business analytics is used by companies that are committed to making data-driven decisions.

Big Data: An Overview

SAS describes Big Data as "a term that describes the large volume of data –both structured and unstructured –that inundates a business on a day-to-day basis." What's important to keep in mind about Big Data is that the amount of data is not as important to an organization as the analytics that accompany it. When companies analyze Big Data, they are using Business Analytics to get the insights required for making better business decisions and strategic moves.

Business Analytics Process

Business Problem Framing

This is the first thing you do before you start your analysis. Even before you begin your analysis, you should understand the purpose of your analysis. Here you try to understand what the business is and what the business is trying to achieve. You formulate the business problem.

Analytics Problem Framing

Here you reformulate the business problem with respect to analytics. You develop a proposed set of factors and its relationship to output. Also, you define a metric of success of your model.

Data

Here you identify and select your data for analysis and its source. You work to clean the data and make it analysis ready. You also find relationships between data and report them.

Methodology Selection and Model Building

Once your data is worked on, you decide which method to use for your analysis. This is decided based on your data and the type of analysis you have to perform. You make multiple models and compare them based on the metrics you decided on.

Deployment

You validate your model to check if your model is giving accurate predictions. Once validated and reported, you deploy your model on the company's system which then will perform analysis on every new incoming data. When a model is deployed, it has to be constantly monitored for accuracy.

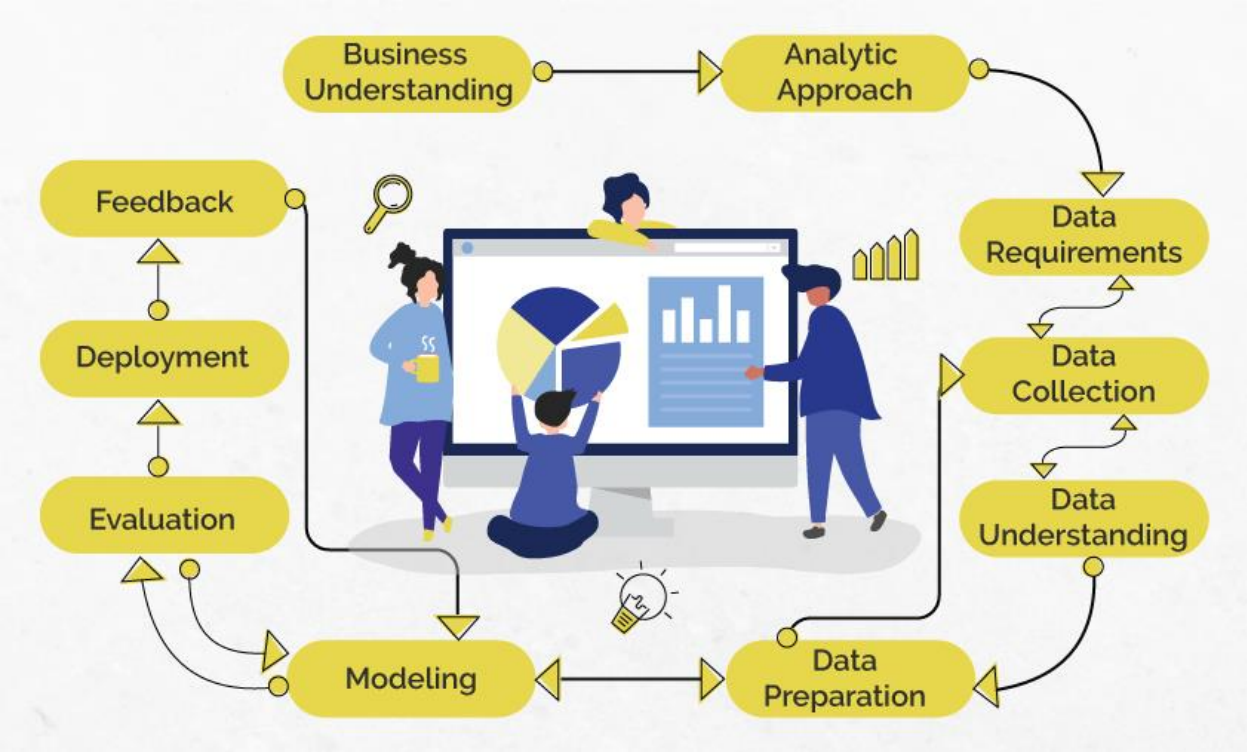


Image 1 - Business Analytical Process 2

Reference -https://www.proschoolonline.com/certification-business-analytics-course/what-is-ba

Understanding Business Analytics

Business Analytics is the procedure through which information is dissected after studying past performances and issues, to devise a successful plan for the future. Big Data or large amounts of data is used to derive solutions.

This method of going about a business or this outlook towards building and sustaining a business is vital to the economy and industries that thrive in the economy

Image 2 - Understanding Business Analytical
Reference - https://www.edureka.co/blog/what-is-business-analytics

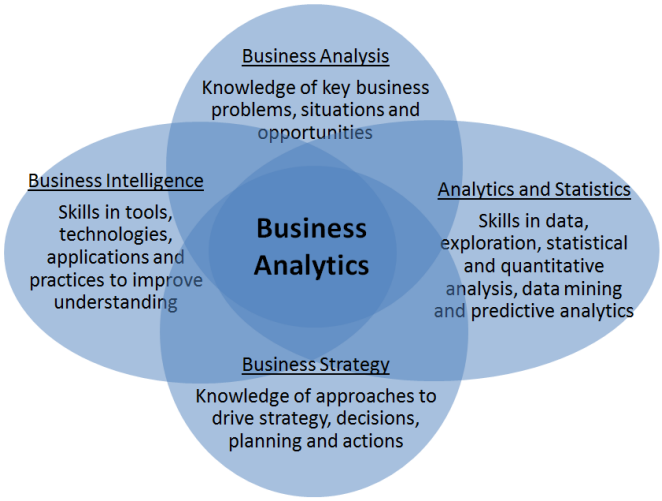


Image 3 - Data Scientist
Reference - https://www.martinsights.com/?p=1049

Components of Business Analytics

The process of business analytics starts from realizing that there is some problem or there is a scope of improvement. This is where good business acumen is required whereby understanding the inner working of the business, one can help and guide in the area of interest.

The next step is often retracing the steps and checking if the process is going in the rights direction or not. This is where business analysts must check that something that they are considering as a problem or scope of improvement actually exists or not.

Once all of this is cross-checked, then business analysts need to identify the relevant data required to do the job. This includes the identification of the data source, the type of data required, the amount of it, and the format in which it can become useful.

This is where Data Engineers also come in the scene as they are often responsible for creating the architecture to perform ETL operations that allow the Business Analysts to get hold of relevant data in a relatively short span of time.

Once the data is achieved, it is treated, cleaned, prepared, and finally is put to use. Here various methodologies are put in place that provide varied kinds of information from the data. While some are simply factual in nature, some provide predictive information.

All these aspects of the analytical process form the different components that are namely

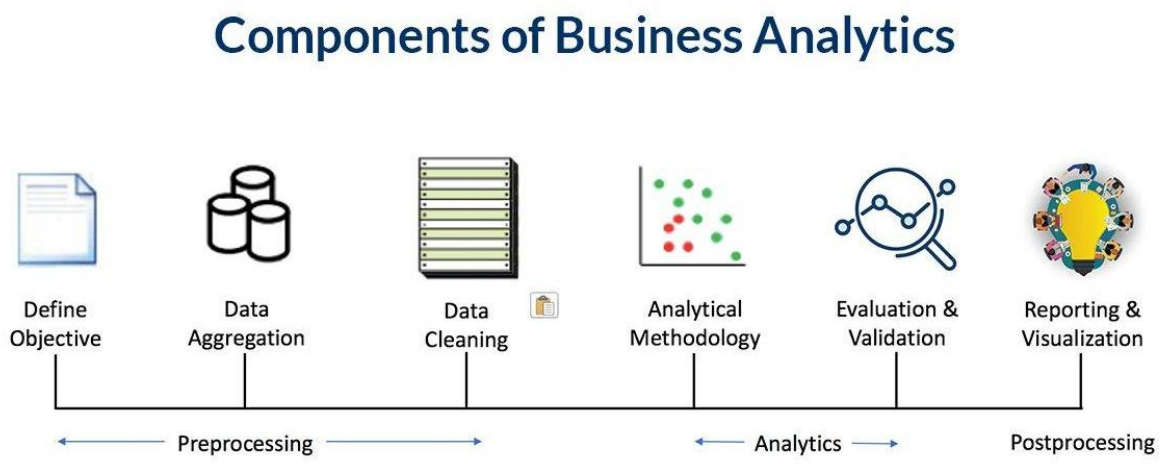


Image 4 - Components of Business Analytics
Reference-https://www.analytixlabs.co.in/blog/what-is-business-analytics

- Define Objective
- Data Aggregation
- Data Cleaning
- Analytical Methodology
- Evaluation and Validation
- Reporting and Data Visualisation

Define Objective

This is the foremost step. Without having a clear understanding of business goals, questions we need to answer, and problems we ought to solve, none of the following steps will deliver. This also helps us to translate business objectives into analytics objectives and map data requirements.

Data Aggregation

The process of having a centralized location for the data, extracting and loading the relevant data by putting relevant filters, and creating subsets of data is the core aspect of Data Aggregation. This is where the data is transformed depending upon the business requirement. Also data pertaining from various sources are combined to have one large dataset. The format of the data is also sometimes changed at this step to make it compatible with the tool being used to achieve the objectives.

Data Cleaning

Data Cleaning is an extremely important component of business analytics because the data in its raw form sometimes is not directly usable. As the other components of Business Analytics use mathematics, statistics, and computer programming, the data must be compatible with these streams of study.

For example, for applying statistics, the data mustn't have any extreme values (also known as outliers), while for mathematics, there should no blank cells or missing values (as matrix operations become difficult) while for programming, the concept of typecasting plays a role where the data is made sure to be in the right format (i.e. correct class or data type).

Also, the concepts of multicollinearity and curse of dimensionality come into play as the business analyst has to make sure that there are no implicit or explicit duplicate columns. The importance of getting rid of the unnecessary columns can only be understood once a good grasp of statistics is

there. Other aspects include the resampling of data (under-sampling, oversampling, hybrid-sampling) removal of duplicate rows, etc.

Analytical Methodology

Having a detailed understanding of the different types of analytics out there dominate this component as this is where the analysts have to identify the method with which they will go to achieve their end goal. If the end goal is to understand what is the present situation of the business then that requires a different set of methods while if there is a need to identify which has happened in the past or what can happen in the future, then a different technique is required. Here, having the know-how of various procedures, methods, and algorithms is important, and knowing what to use, when one business analyst stands apart from the other. Different types of Analytics methods and tools are explained in the latter section.

Evaluation and Validation

Once the results come out, the next task is to understand if the result stands true given a different situation or not. This is where predictive models are used and their evaluation and validation are conducted whereas, for other methods, various simulation techniques are put to use to identify the most plausible outcome, thus providing a very reliable result. Here also, the business analyst needs to learn a range of techniques to identify the shortcoming in their method, work on it, improve it, and make their insights stable and valuable.

Reporting and Data Visualisation

Perhaps the most important and often overlooked component of the discipline is the aspect of communicating the results in an easy to understand way. This requires innovation and creativity and is the reason that this field is open to all and not only to mathematicians, statisticians, or computer programmers. To quickly make people understand the complex insights discovered over weeks or even months, reports or presentations are created that have simple tables, bullet points, etc. On top of all this, visualizing the data plays a major role here as it allows the people in the leadership positions to quickly view where the organization is coming from and perhaps where they are headed. Business Analytics ought to know the various ways of visualizing the data, the transformation required to be done on the data to make it possible, and finding innovative ways to string together different information ina smooth storytelling method.

Types of Business Analytics Methods

Big data analytics cannot be assumed as a one-size-fits-all strategy. In fact, what differentiates a finest data expert or data analyst from others, is their capability to categorize the type of analytics that can be used to benefit the business at the optimum level.

Fundamentally there are Four types of analytics.

1. **Descriptive analytics**-Describing or summarising the existing data using existing business intelligence tools to better understand what is going on or what has happened.
2. **Diagnostic analytics**-Focus on past performance to determine what happened and why. The result of the analysis is often an analytic dashboard.
3. **Predictive analytics**-Emphasizes on predicting the possible outcome using statistical models and machine learning techniques.
4. **Prescriptive analytics**-It is a type of predictive analytics that is used to recommend one or more courses of action on analyzing the data.

Image 5 -Types of Business Analytics Methods
Reference-https://www.analyticsinsight.net/four-types-of-business-analytics-to-know/

Descriptive Analytics

Descriptive analytics manipulates figures from multiple data sources to provide valuable information about the past. So, developers can predict important trends and signal the necessity of preventative or stimulative actions.

These describe what has already happened.

- For example, a retailer can learn the number of clients, the average bill, the most popular goods, etc.
- A medical company can evaluate the most common illnesses and susceptibility to disease.
- Governments can stimulate population growth by organising additional social support for families with several children if the overall number of pregnant women has been decreasing over several consistent years.

With the help of descriptive analysis, any company is able to group its customers by social factors, behaviour and other features, as well as monitoring peak activities according to seasonal or local factors.

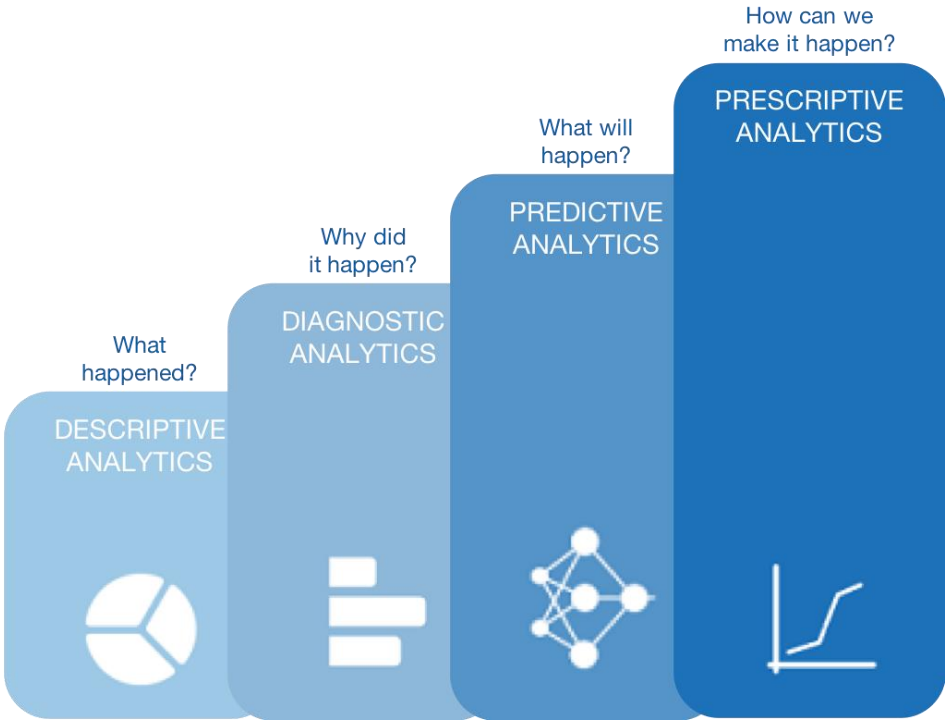


Image 6 - Descriptive Analytics
Reference -https://magora-systems.com/types-of-data-analytics/

Diagnostic analytics

This type doesn't simply state but aims to find the reason behind why something happened. We use it to identify the patterns and consequences of our policies and actions. It provides a deeper understanding of any given problem.

In IT, for this purpose, we use business intelligence (BI), to be more specific -it's machine learning techniques again. These are computer-based methods of analysing and reporting valuable information. For example, using ML in healthcare, medics can diagnose a person's susceptibility to cancer-based on medical screenings.

Predictive analytics

Predictive analytics forecasts the possibility of future events using statistical models and machine learning techniques.
Predictive analytics address what might happen.

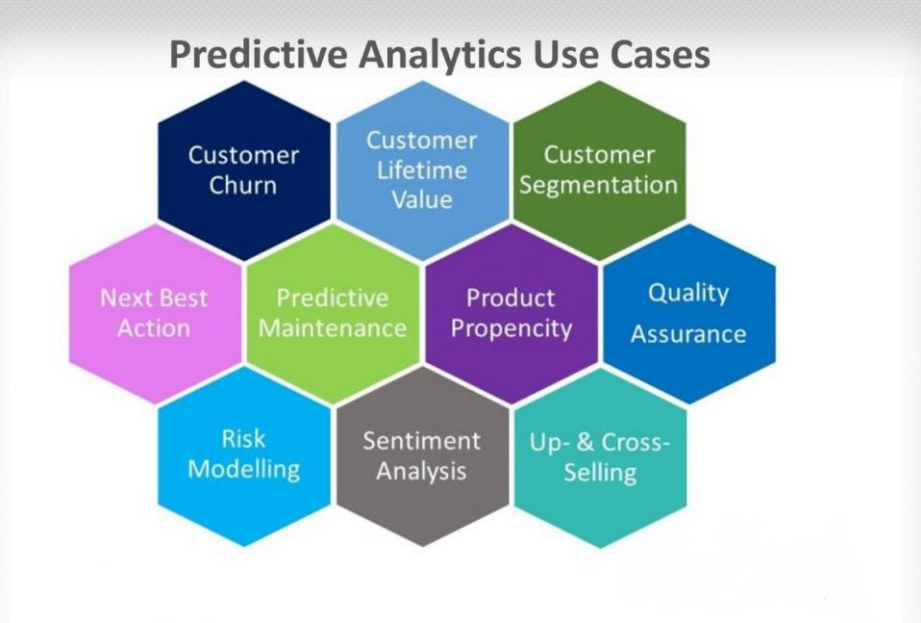


Image 7 -Predictive Analytics
Reference -https://magora-systems.com/types-of-data-analytics/

To be able to predict trends and see into the future, this type of analytics uses the results of the previous two –i.e. it bases its results on true facts of the past.

- With the help of predictive analysis, an entrepreneur can optimise the raw material storage and the warehouse stock. Computer systems predict stock exchanges, market fluctuations and currency exchange rates. This is specifically useful in finance, production, logistics and banking.

However, it's important to understand that all the results this type of analytics provides you with are approximate. The accuracy of data and the stability of the situation have a significant influence on the result. It requires careful processing and constant optimisation.

Combining the approaches gives the best, most relevant results.

Prescriptive analytics

Prescriptive analysis is based on mathematical modelling. Its mission is to show the consequences of certain actions based on possible changes to data and conditions.

Image 8 - Prescriptive Analytics
Reference -https://magora-systems.com/types-of-data-analytics/

When is it time to use prescriptive analysis?

- Prescriptive analytics is a branch with a high degree of responsibility: it utilises top-notch tools and technologies, such as machine learning and data mining.
- It helps in decision-making by constructing a potential future and estimating the probability and extent of any given factor's influence. (Remember "Back to the Future" –they were realising the prescriptive analysis in a series of events).

- It's complicated and expensive. If you are working with vital factors, it lets you save much more than you spend. But for some cases, it's worth using mathematical modelling and statistical analysis to get value for money solution.

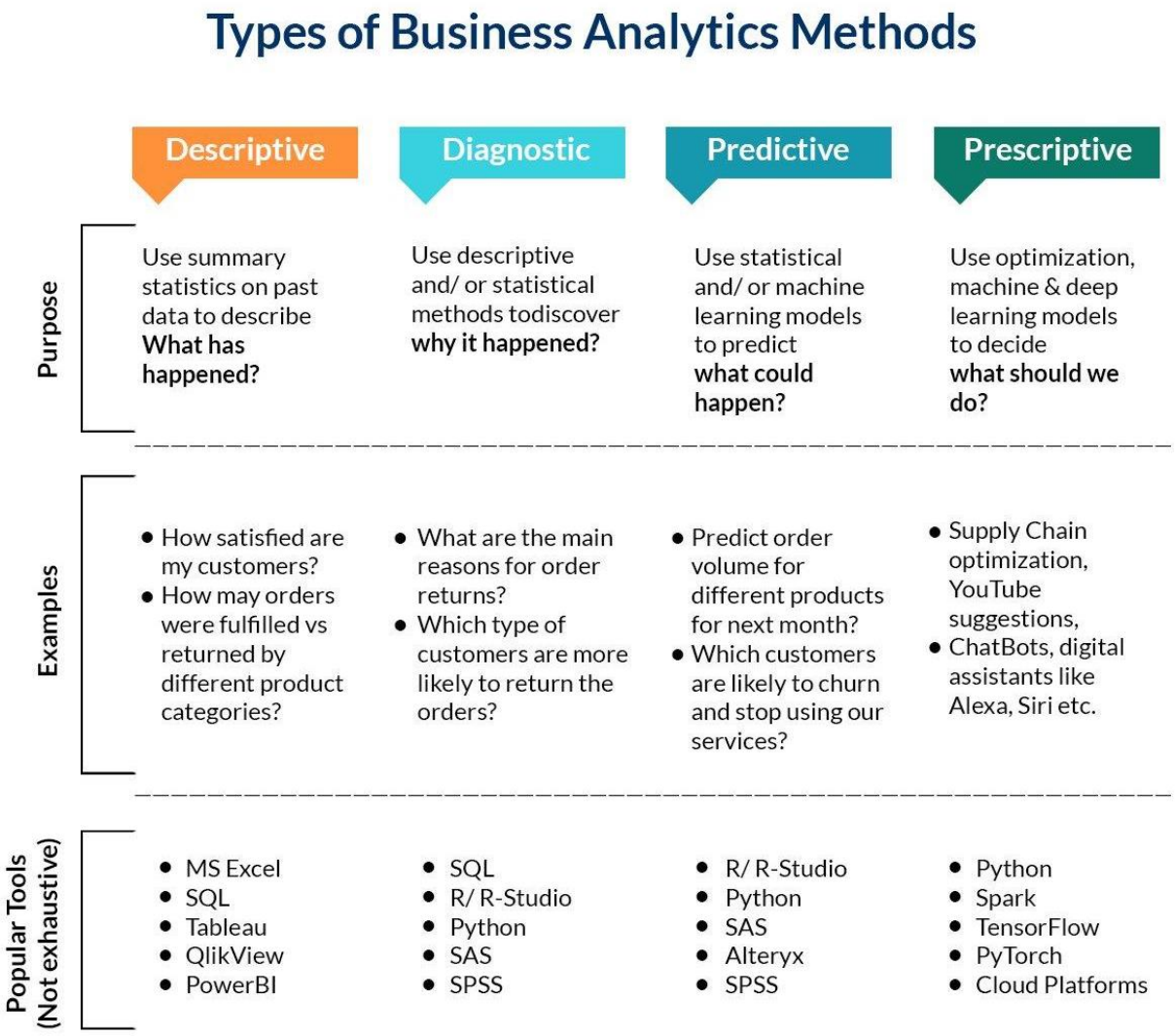


Image 9 - Types of Business Analytics Methods
Reference-https://www.analytixlabs.co.in/blog/what-is-business-analytics/

Uses and Benefits of Business Analytics

Business Analytics is adopted by companies to facilitate data-driven decisions. The insight that is arrived at with the help of business analytics enables companies to optimize their various processes to provide even better results and thus achieve a competitive advantage over others.

- To carry out data mining and exploring new data to find new patterns and relationships.

- To carry out statistical and quantitative analysis to provide explanations for certain occurrences.
- Test previous decisions are taken with the help of A/B testing and multivariate testing.
- Deploy predictive modeling to predict future outcomes.

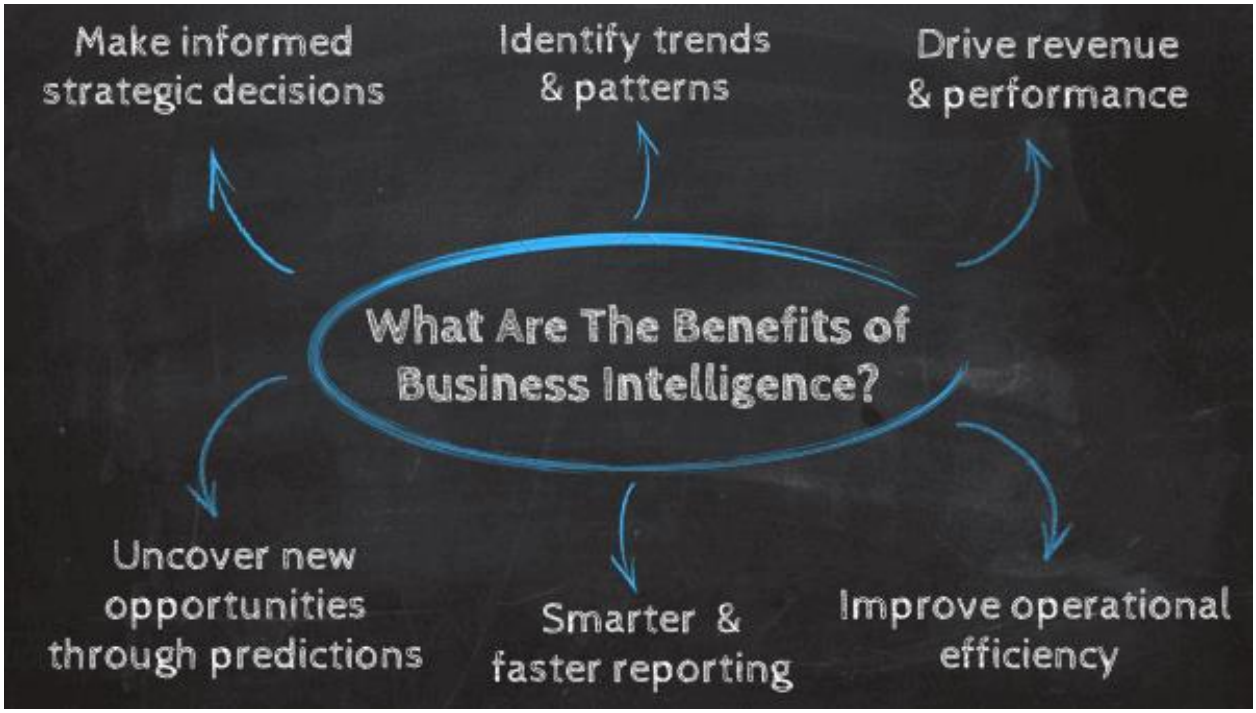


Image 10 - Uses and benefits of Business Analytics
Reference-https://www.datapine.com/blog/benefits-of-business-intelligence-and-business-analytics

Business Analytics Tools

What are business analytics can also be answered by understanding the tools used in this field. As the discipline of business analytics is of a unique kind as it covers various types of analytics and this is the reason that there is a range of business analytics tools to accomplish the objectives.

As business analysts grapple with a number of problems that range from sourcing data to create predictive models, tools that specialize in different fields are put to use and a good business analyst must know how to use all of them in harmony. Some of the common Business Analytics related tools include:

- SQL
- Tableau/ QlikView/ Power BI
- Birt
- Python

- R
- MS Excel
- Sisense
- Clear Analytics
- Pentaho BI
- MicroStrategy

SQL

It is among the most important tools as SQL queries allow the user to easily filter out and create subsets of an otherwise large dataset. By having the relevant amount of data, the analyst can quickly start working on the cleaning of the data and then creating models out of it. SQL sometimes is used with other tools such as Omni Sci or Zeppelin by Apache where PostgreSQL is used.

Tableau/ QlikView/ Power BI

The most important tool for report generation through the means of visualization. Tableau allows the user to quickly create interesting, complex, and detailed graphs that can magnify the impact of a report. The good aspect of this tool is that it is easy to use and requires less data preparation in order to get the desired output.

Birt

Another useful report-based tool allows us to create graphs and dashboards, however, it is relatively more complex than tableau as the user needs to have a decent knowledge of Java to make the most out of it.

Python

One of the most advanced tools, python allows the user to perform multiple things. Python can be used to perform basic steps such as data cleaning to a complex aspect of analytics that includes the development of various kinds of models. The development of highly complex machine learning and deep learning models is particularly effective through this tool. Python also allows us to create reports and has libraries for visualization but it is up to the user to use them or use dedicated visualization tools.

R

This statistical tool created “by the statisticians for the statisticians”, allows a business analyst to perform all the descriptive and inferential statistics along with the development of statistical models. If compared to python it has a bit of a steep learning curve but this eventually pays off as it has a large community of users and is respected in the world of corporate as well as academia.

MS Excel

One of the most basic yet widely used and effective tools. The importance of MS Excel in the field of Business Analytics can be understood from realizing the difference between a sword and a needle. While performing the complex operation of data extraction, model development, and report generation, there are several heavyweight tools. MS Excel on the other hand sometimes is used at the very end or sometimes in the very beginning to provide an easy, user interactive experience to gain quick insights regarding the data or the final output and this is the reason that still many times the final output is in the form of an Excel.

Sisense

Sisense is an agile business intelligence (BI) solution that provides advanced tools to manage and support business data with analytics, visuals and reporting.

Clear Analytics

Clear Analytics is a data analytics solution that enables small to midsize business users to perform a variety of self-service analytics within an Excel-based environment. ... Clear Analytics can publish data to Microsoft's Power BI Cloud Portal to make dashboards available online or on mobile devices.

Pentaho BI

Pentaho is a business intelligence system designed to help companies make data-driven decisions, with a platform for data integration and analytics. The platform includes extract, transform, and load (ETL), big data analytics, visualizations, dashboards, reporting, data mining, and predictive analytics.

MicroStrategy

MicroStrategy is a Business Intelligence software, which offers a wide range of data analytics capabilities. As a suite of applications, it offers Data Discovery, Advanced Analytics, Data Visualizations, Embedded BI, and Banded Reports and Statements.

Applications of Business Analytics

Marketing –Business Analytics helps in the marketing field as it correctly studies consumer behavior and market trends. In addition, companies can thus base their strategies on this vital information and identify their target audience as well as identify new markets to penetrate.
Finance –In the field of finance, business analytics tools can help companies to uncover vital insights on stock performances by processing vast amounts of data.
Human Resources –HR professionals are turning to business analytics tools to conduct background checks on candidates and get relevant valuable information.
Manufacturing –The data that is collected can be subjected to these tools to provide information regarding inventory management, supply chain management, performance insight, and risk mitigation methods. Companies can also work on their operations skills through these tools.

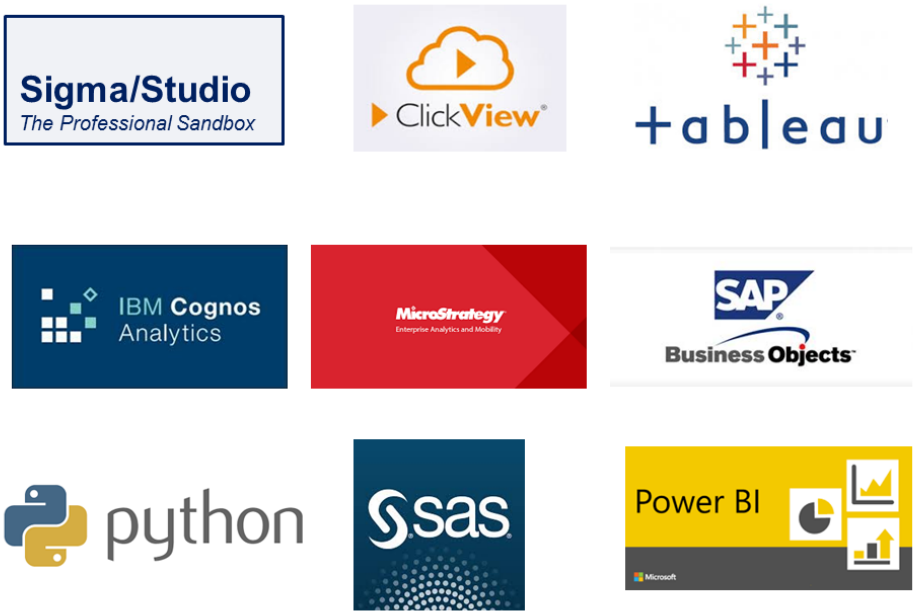


Image 11 - Business Analytics Tools
Reference-https://sigma4sap.com/?page_id=466

Trends in business analytics

Business Analytics Trends For 2021

Data Quality Management (DQM)

The analytics trends in data quality grew greatly this past year. The development of business intelligence to analyze and extract value from the countless sources of data that we gather at a high scale, brought alongside a bunch of errors and low-quality reports: the disparity of data sources and data types added some more complexity to the data integration process.

Data Discovery/Visualization

Data discovery has increased its impact in the last year. The already mentioned survey conducted by the Business Application Research Center listed data discovery in the top 3 business intelligence trends by the importance hierarchy. BI practitioners steadily show that the empowerment of business users is a strong and consistent trend.

Artificial Intelligence

Artificial intelligence is already widely used in business applications, including automation, data analytics, and natural language processing. Across industries, these three fields of AI are streamlining operations and improving efficiencies. Automation alleviates repetitive or even dangerous tasks.

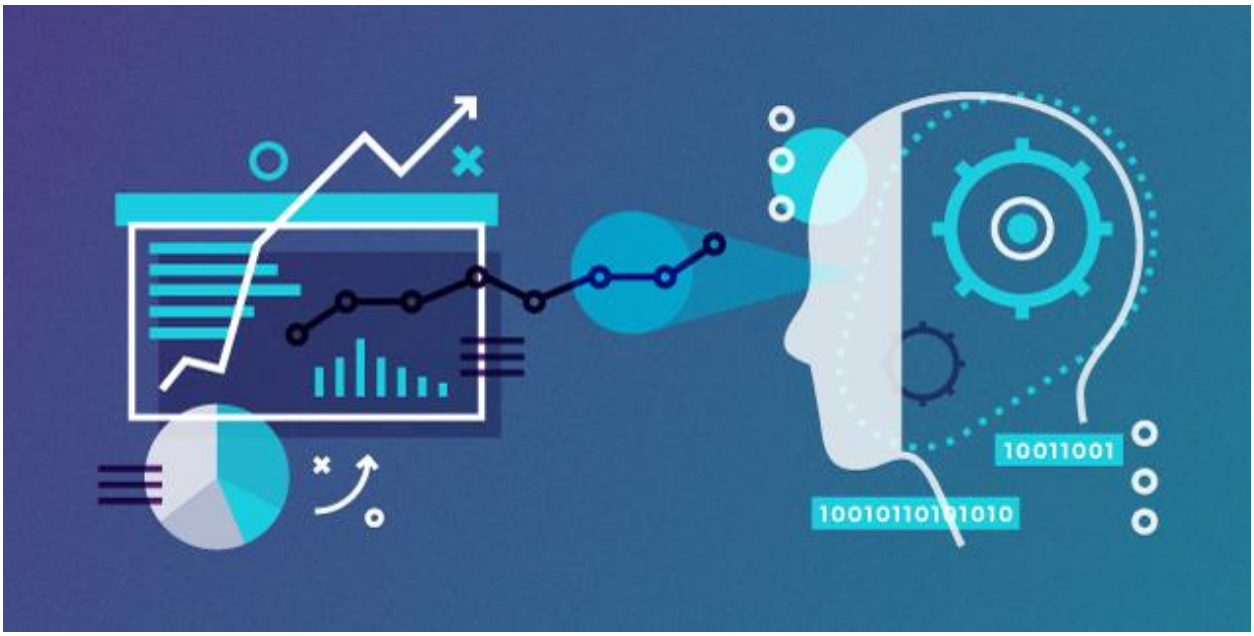


Image 12 - ArtificialIntelligence
Reference-https://feedsportal.com/artificial-intelligence-is-the-future-of-business-analytics/

Predictive And Prescriptive Analytics Tools

Predictive analytics is the practice of extracting information from existing data sets in order to forecast future probabilities. It's an extension of data mining which refers only to past data. Predictive analytics includes estimated future data and therefore always includes the possibility of errors from its definition, although those errors steadily decrease as software that manages large volumes of data today becomes smarter and more efficient. Predictive analytics indicates what might happen in the future with an acceptable level of reliability, including a few alternative scenarios and risk assessment. Applied to business, predictive analytics is used to analyze current data and historical facts in order to better understand customers, products, and partners and to identify potential risks and opportunities for a company.

Prescriptive analytics goes a step further into the future. It examines data or content to determine what decisions should be made and which steps taken to achieve an intended goal. It is characterized by techniques such as graph analysis, simulation, complex event processing, neural networks, recommendation engines, heuristics, and machine learning. Prescriptive analytics tries to see what the effect of future decisions will be in order to adjust the decisions before they are actually made. This improves decision-making a lot, as future outcomes are taken into consideration in the prediction. Prescriptive analytics can help you optimize scheduling, production, inventory, and supply chain design to deliver what your customers want in the most optimized way.

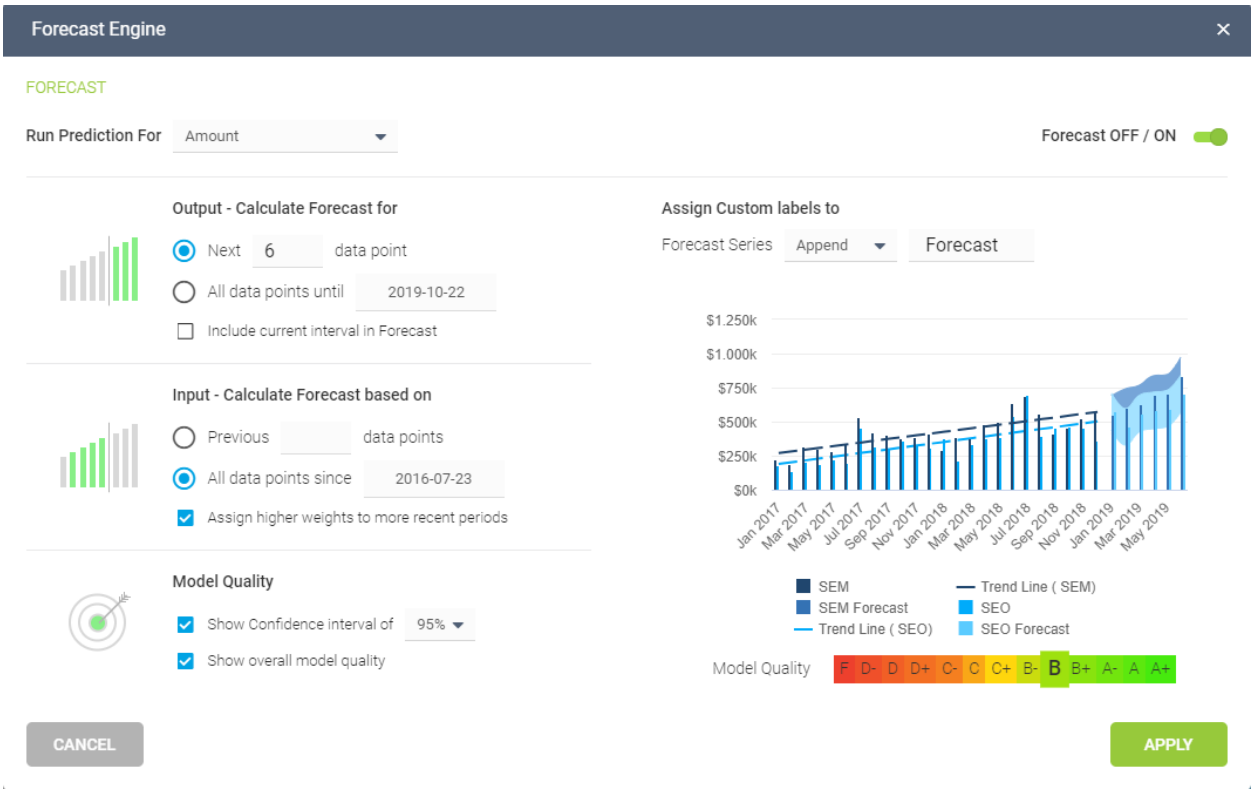


Image 13 - Predictive and Prescriptive Analytics Tools
Reference-https://www.datapine.com/blog/business-intelligence-trends/

Collaborative Business Intelligence

Collaborative BI (collaborative business intelligence) is the merging of business intelligence software with collaboration tools, including social and Web 2.0 technologies, to support improved data-driven decision making.

Data-driven Culture

The concept of a data-driven culture treats data as the main resource for leveraging insights in every department of the organization. While companies have always been interested in their numbers, the extent of data use is exercised at a higher level within a data-driven culture.

Augmented Analytics

Augmented analytics is the use of statistical and linguistic technologies to improve data management performance, from data analysis to data sharing and business intelligence. It is somehow connected to the ability to transform big data into smaller, more usable, datasets.

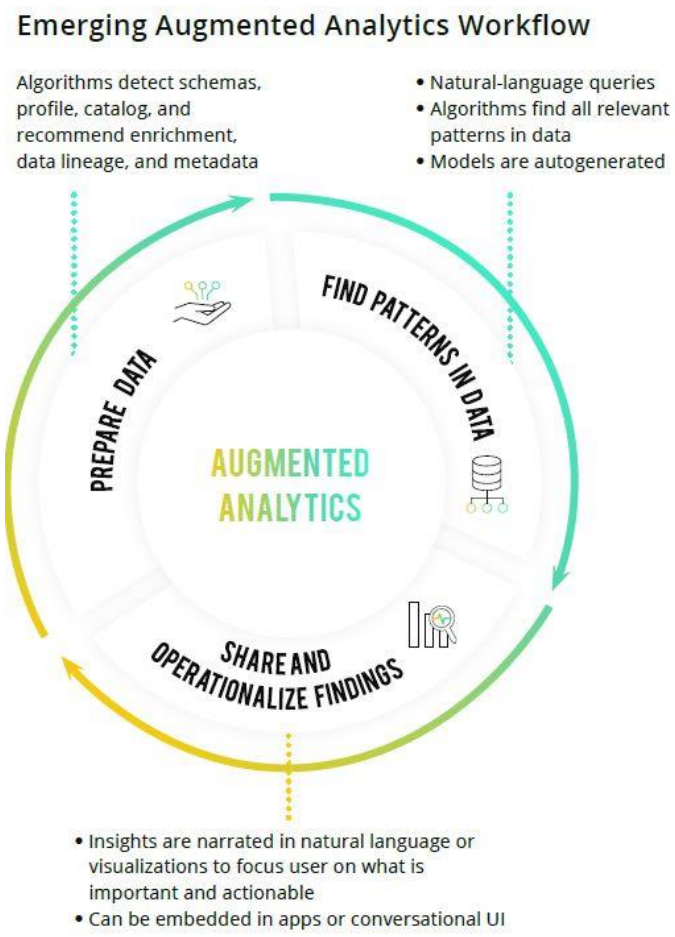


Image 14 - Augmented Analytics
Reference-https://www.sisense.com/whitepapers/augmented-analytics-the-future-of-business-intelligence/

Mobile BI

Mobile BI refers to the recent trend of business users accessing their data and dashboards on mobile and tablet devices. This change is forcing designers to re-evaluate the user experience (UX) and replicate the same relevant content usually seen solely on desktop devices on mobile.

Image 15 - Mobile BI
https://matillion.com/wp-content/uploads/2015/09/mobile-business-intelligence-solution-needs-to-be.jpg

Data Automation

Business intelligence topics wouldn't be complete without data (analysis) automation. In the last decade, we saw so much data produced, stored, and ready to process that companies and organizations were seriously looking for modern data automation solutions to tackle massive volumes of information that has been collected. Gartner predicts that next year more than 40% of data science tasks will be automated, hence, this is one of the trends in business intelligence that we need to keep an eye on.

Embedded analytics

Embedded analytics is the integration of analytic content and capabilities within business process applications. It provides relevant information and analytical tools designed so users can work smarter and more efficiently in the applications they use every day.

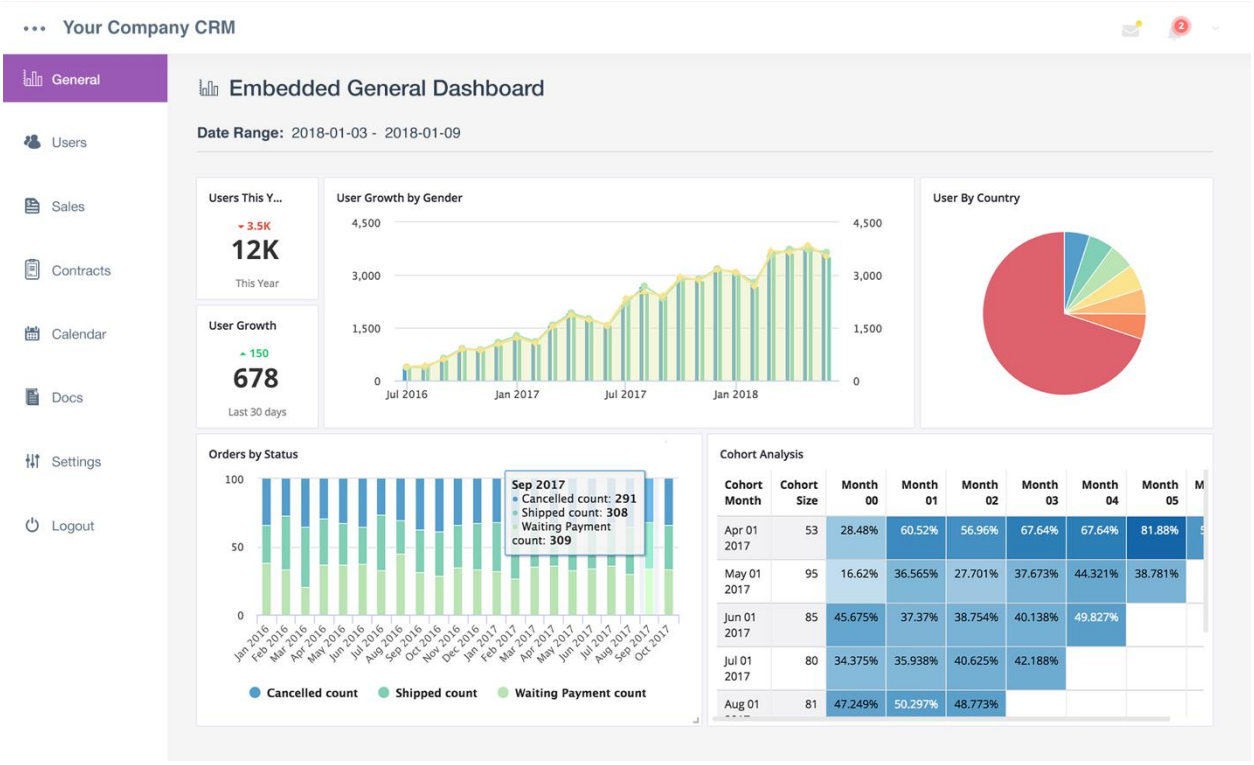


Image 16 - Embedded analytics
Reference-https://www.holistics.io/features/embedded-analytics/

Natural language processing

Natural language processing (NLP) is a subfield of linguistics, computer science, information engineering, and artificial intelligence concerned with the interactions between computers and human (natural) languages, in particular how to program computers to process and analyze large amounts of natural language data.

Speech Recognition: Converting spoken words into data a computer can understand. This is the NLP technology you use every time you ask Siri, Cortana, Echo or Google Voice a question.
Machine Translation: Translating text from one language to another. This is the tech that underlies translation apps like Google Translate.
Natural Language Generation: Outputting information as a human language. This is the tech you use every time Siri or Cortana answers your question.

Semantic Search: Closely linked to speech recognition, as above, this allows you to ask natural questions of an app like Siri, rather than having to formulate your question in a particular, unnatural way.
Machine Learning: Machine learning is a whole other topic, but essentially, it uses the data that NLP interprets to "teach" itself about future actions.

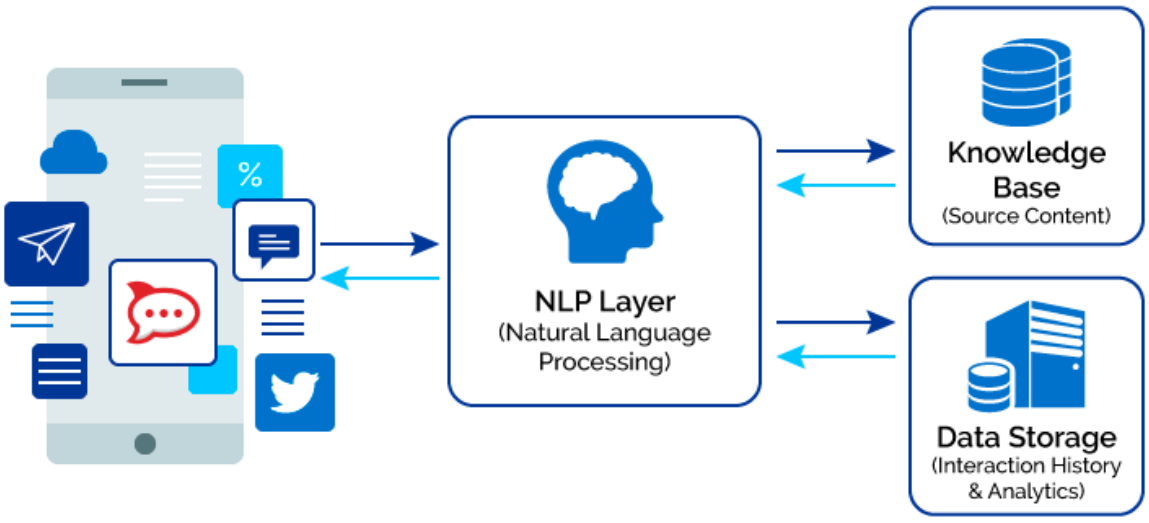


Image 17 - NLP
Reference-https://marutitech.com/what-nlp-reasons-everyone-retail-use-it/

Descriptive analytics

What are Descriptive Analytics?

Descriptive analytics is the interpretation of historical data to better understand changes that have occurred in a business. Descriptive analytics describes the use of a range of historic data to draw comparisons.

Descriptive analytics is a statistical method that is used to search and summarize historical data in order to identify patterns or meaning.

For learning analytics, this is a reflective analysis of learner data and is meant to provide insight into historical patterns of behaviors and performance in online learning environments.

For example, in an online learning course with a discussion board, descriptive analytics could determine how many students participated in the discussion, or how many times a particular student posted in the discussion forum.

Descriptive approach is considered to be the foundation of research. Its logical design is based on statistics of the research analysis. Since this analysis doesn't explain the cause of result, hence it

can’t take into account the validity of research results. Here are the common methods used in descriptive analytics:

Observation Method: In this method data is observed in both natural and artificial ways in order to draw meaningful conclusions. It is an effective way of inferring from natural observation since we can obtain original results of the research. On the other hand, in the artificial method results would depend on the quantity of the data provided for observation.

Case study Method: It involves a deep study on all the problems discussed. It makes us understand a particular situation more closely.

Survey Method: In this method questionnaires are prepared and given to the participants. After receiving the answers the research is preceded and results are concluded.

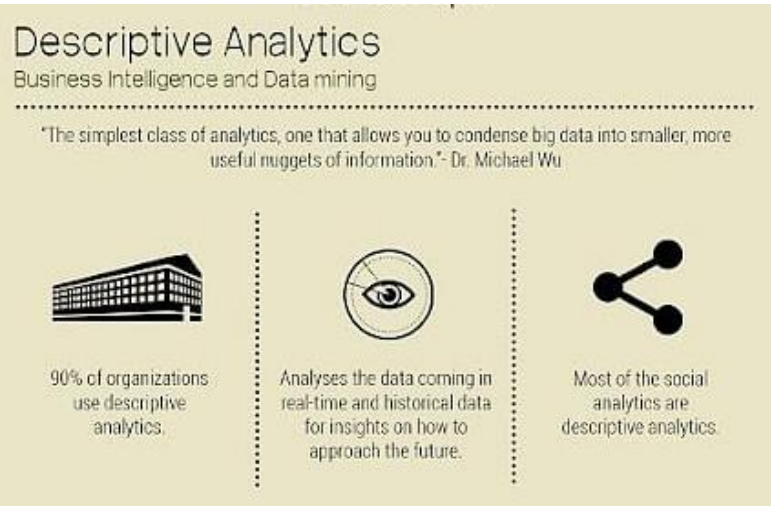


Image 18 - Descriptive Analytics
Reference-https://www.dezyre.com/article/types-of-analytics-descriptive-predictive-prescriptive-analytics/209

How does descriptive analytics work?

Data aggregation and data mining are two techniques used in descriptive analytics to discover historical data. Data is first gathered and sorted by data aggregation in order to make the datasets more manageable by analysts.

Data mining describes the next step of the analysis and involves a search of the data to identify patterns and meaning. Identified patterns are analyzed to discover the specific ways that learners interacted with the learning content and within the learning environment.

What can descriptive analytics tell us?

Image 19 - Descriptive Analytics
Reference-https://www.valamis.com/hub/descriptive-analytics

The kind of information that descriptive analytics can provide depends on the learning analytic capability of the learning management system (LMS) being used and what the system is reporting on specifically.

Some common indicators that can be identified include learner engagement and learner performance. With learner engagement, analysts can detect the participation level of learners in the course and how and when course resources were accessed.

Performance data provides analysts with insight into how well learners succeeded on the course; this information could come from data taken from assessments or assignments. It's important to note that insights learned from descriptive analysis are not used for making inferences or predictions about a learner's future performance.

The analytical method is meant to provide strategic insight into where learners, or a specific learner, may have needed more support. It can also help course designers improve the design of learning by providing insight into what went well and what did not go well on the course.

Examples of descriptive analytics

Many LMS platforms and learning systems offer descriptive analytical reporting with the aim of helping businesses and institutions measure learner performance to ensure that training goals and targets are met.

The findings from descriptive analytics can quickly identify areas that require improvement - whether that be improving learner engagement or the effectiveness of course delivery.

Here are some examples of how descriptive analytics is being used in the field of learning analytics:

- Tracking course enrolments, course compliance rates,
- Recording which learning resources are accessed and how often
- Summarizing the number of times, a learner posts in a discussion board
- Tracking assignment and assessment grades
- Comparing pre-test and post-test assessments
- Analysing course completion rates by learner or by course
- Collating course survey results
- Identifying length of time that learners took to complete a course

Image 20 - Example of Descriptive Analytics
Reference-https://www.sisense.com/glossary/descriptive-analytics/

The functions delivered by descriptive analytics solutions fall broadly into five categories:
State business metrics: Determine which metrics are important for evaluating performance against business goals. Some goal examples would be to increase revenue, reduce costs, improve operational efficiency, and measure productivity. Each goal must have associated KPIs to help monitor achievement.

Identify data required: Business data is located in many different sources within the enterprise, including systems of record, databases, desktops, and shadow IT repositories. To measure accurately against KPIs, companies must catalog and prepare the correct data sources to extract the needed data and calculate metrics based on the current state of the business.

Extract and prepare data: Data must be prepared for analysis. Deduplication, transformation, and cleansing are a few examples of the data preparation steps that need to take place prior to analysis. Often, this is the most time-consuming and labor-intensive step, requiring up to 80% of an analyst's time, but it is critical for ensuring accuracy.

Analyze data: Data analysts can create models and run analyses such as summary statistics, clustering, and regression analysis on the data to determine patterns and measure performance. Key metrics are calculated and compared with stated business goals to evaluate performance based on historical results. Data scientists often use open source tools like R and Python to programmatically analyze and visualize data.

Present data: Results of the analytics are usually presented to stakeholders in the form of charts and graphs. This is where the data visualization mentioned earlier comes into play. BI tools give users the ability to present data visually in a way that non-data analysts can understand. Many self service data visualization tools also enable business users to create their own visualizations and manipulate the output.

Advantages of descriptive analytics

When learners engage in online learning, they leave a digital trace behind with every interaction they have in the learning environment.

This means that descriptive analytics in online learning can gain insight into behaviours and performance indicators that would otherwise not be known.

Here are some advantages to utilizing this information:

- Quickly and easily report on the Return on Investment (ROI) by showing how performance achieved business or target goals.
- Identify gaps and performance issues early - before they become problems.
- Identify specific learners who require additional support, regardless of how many students or employees there are.

- Identify successful learners in order to offer positive feedback or additional resources.
- Analyze the value and impact of course design and learning resources.

Introduction to Big Data Analytics

What is Data?

The quantities, characters, or symbols on which operations are performed by a computer, which may be stored and transmitted in the form of electrical signals and recorded on magnetic, optical, or mechanical recording media.

What is Big Data?

Big Data is a collection of data that is huge in volume, yet growing exponentially with time. It is a data with so large size and complexity that none of traditional data management tools can store it or process it efficiently. Big data is also a data but with huge size.
In this Big Data analytics, you will learn,

- What is Data?
- What is Big Data?
- What is an Example of Big Data?
- Types Of Big Data
- Characteristics Of Big Data
- Advantages Of Big Data Processing

Image 21 - What is Big Data?
Reference: https://www.guru99.com/images/Big_Data/061114_0759_WhatIsBigDa1.jpg

What is an Example of Big Data?

Following are some of the Big Data examples-

The **New York Stock Exchange** is an example of Big Data that generates about ***one terabyte*** of new trade data per day.

Image 22
Reference: https://www.guru99.com/images/Big_Data/061114_0759_WhatIsBigDa2.jpg

Social Media

The statistic shows that ***500+terabytes*** of new data get ingested into the databases of social media site **Facebook**, every day. This data is mainly generated in terms of photo and video uploads, message exchanges, putting comments etc.

Image 23
Reference: https://www.guru99.com/images/Big_Data/061114_0759_WhatIsBigDa3.jpg

A single **Jet engine** can generate ***10+terabytes*** of data in ***30 minutes*** of flight time. With many thousand flights per day, generation of data reaches up to many ***Petabytes.***

Image 24
Reference: https://www.guru99.com/images/Big_Data/061114_0759_WhatIsBigDa4.jpg

Types Of Big Data

Following are the types of Big Data:

1. **Structured**
2. **Unstructured**
3. **Semi-structured**

Structured

Any data that can be stored, accessed and processed in the form of fixed format is termed as a 'structured' data. Over the period of time, talent in computer science has achieved greater success in developing techniques for working with such kind of data (where the format is well known in advance) and also deriving value out of it. However, nowadays, we are foreseeing issues when a size of such data grows to a huge extent, typical sizes are being in the rage of multiple zettabytes.

Do you know? 10^{21} bytes equal to ***1 zettabyte*** or ***one billion terabytes*** forms ***a zettabyte***.

Looking at these figures one can easily understand why the name Big Data is given and imagine the challenges involved in its storage and processing.

Do you know? Data stored in a relational database management system is one example of a **'structured'** data.

Examples Of Structured Data

An 'Employee' table in a database is an example of Structured Data

Employee_ID	Employee_Name	Gender	Department	Salary_In_lacs
2365	Rajesh Kulkarni	Male	Finance	650000
3398	Pratibha Joshi	Female	Admin	650000
7465	Shushil Roy	Male	Admin	500000
7500	Shubhojit Das	Male	Finance	500000
7699	Priya Sane	Female	Finance	550000

Unstructured

Any data with unknown form or the structure is classified as unstructured data. In addition to the size being huge, un-structured data poses multiple challenges in terms of its processing for deriving value out of it. A typical example of unstructured data is a heterogeneous data source containing a combination of simple text files, images, videos etc. Now day organizations have wealth of data available with them but unfortunately, they don't know how to derive value out of it since this data is in its raw form or unstructured format.

Examples Of Un-Structured Data

The output returned by 'Google Search'

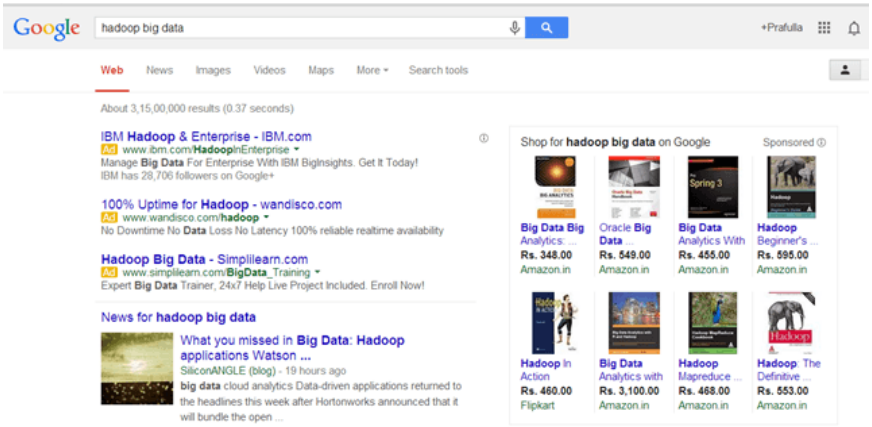


Image 25 - Example of Un-structured Data
Reference: https://www.guru99.com/images/Big_Data/061114_0759_WhatIsBigDa5.png

Semi-structured

Semi-structured data can contain both the forms of data. We can see semi-structured data as a structured in form but it is actually not defined with e.g., a table definition in relational DBMS. Example of semi-structured data is a data represented in an XML file.

Examples Of Semi-Structured Data

Personal data stored in an XML file-

<rec><name>Prashant Rao</name><sex>Male</sex><age>35</age></rec>
<rec><name>Seema R.</name><sex>Female</sex><age>41</age></rec>
<rec><name>Satish Mane</name><sex>Male</sex><age>29</age></rec>
<rec><name>Subrato Roy</name><sex>Male</sex><age>26</age></rec>
<rec><name>Jeremiah J.</name><sex>Male</sex><age>35</age></rec>

Data Growth over the years

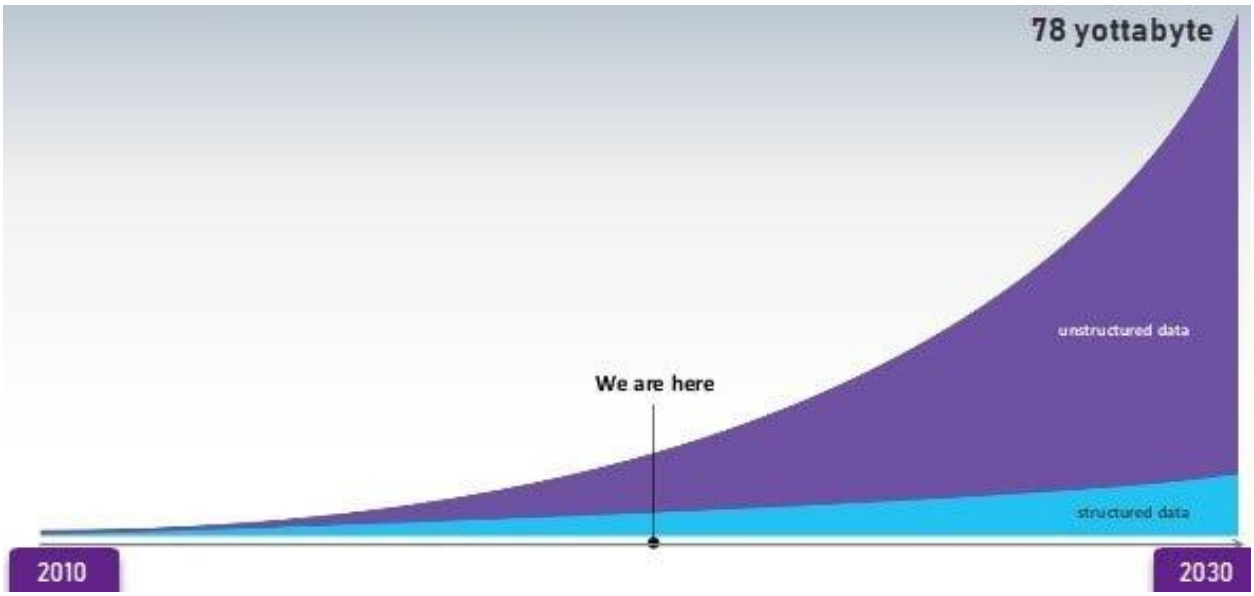


Image 26 - Data Growth over the years
Reference: https://www.guru99.com/images/1/big-data-growth.jpg

Please note that web application data, which is unstructured, consists of log files, transaction history files etc. OLTP systems are built to work with structured data wherein data is stored in relations (tables).

Characteristics Of Big Data

Big data can be described by the following characteristics:

- Volume
- Variety
- Velocity
- Variability

***(i) Volume* –** The name Big Data itself is related to a size which is enormous. Size of data plays a very crucial role in determining value out of data. Also, whether a particular data can actually be considered as a Big Data or not, is dependent upon the volume of data. Hence, **'Volume'** is one characteristic which needs to be considered while dealing with Big Data solutions.

***(ii) Variety* –** The next aspect of Big Data is its **variety**.

Variety refers to heterogeneous sources and the nature of data, both structured and unstructured. During earlier days, spreadsheets and databases were the only sources of data considered by most of the applications. Nowadays, data in the form of emails, photos, videos, monitoring devices, PDFs, audio, etc. are also being considered in the analysis applications. This variety of unstructured data poses certain issues for storage, mining and analyzing data.

***(iii) Velocity* –** The term **'velocity'** refers to the speed of generation of data. How fast the data is generated and processed to meet the demands, determines real potential in the data.

Big Data Velocity deals with the speed at which data flows in from sources like business processes, application logs, networks, and social media sites, sensors, Mobile devices, etc. The flow of data is massive and continuous.

***(iv) Variability* –** This refers to the inconsistency which can be shown by the data at times, thus hampering the process of being able to handle and manage the data effectively.

Advantages Of Big Data Processing

Ability to process Big Data in DBMS brings in multiple benefits, such as-

- Businesses can utilize outside intelligence while taking decisions

Access to social data from search engines and sites like facebook, twitter are enabling organizations to fine tune their business strategies.

- Improved customer service

Traditional customer feedback systems are getting replaced by new systems designed with Big Data technologies. In these new systems, Big Data and natural language processing technologies are being used to read and evaluate consumer responses.

- Early identification of risk to the product/services, if any
- Better operational efficiency

Big Data technologies can be used for creating a staging area or landing zone for new data before identifying what data should be moved to the data warehouse. In addition, such integration of Big Data technologies and data warehouse helps an organization to offload infrequently accessed data.

Summary

- Big Data definition: Big Data meaning a data that is huge in size. Bigdata is a term used to describe a collection of data that is huge in size and yet growing exponentially with time.
- Big Data analytics examples includes stock exchanges, social media sites, jet engines, etc.
- Big Data could be 1) Structured, 2) Unstructured, 3) Semi-structured
- Volume, Variety, Velocity, and Variability are few Big Data characteristics
- Improved customer service, better operational efficiency, Better Decision Making are few advantages of Bigdata

Introduction to Descriptive statistics

What are descriptive statistics?

In Descriptive statistics you are describing, presenting, summarizing, and organizing your data, either through numerical calculations or graphs or tables.

Descriptive statistics are brief descriptive coefficients that summarize a given data set, which can be either a representation of the entire population or a sample of a population. Descriptive statistics are broken down into measures of central tendency and measures of variability (spread). Measures of central tendency include the mean, median, and mode, while measures of variability include standard deviation, variance, minimum and maximum variables, kurtosis, and skewness.

Understanding Descriptive Statistics

Descriptive statistics, in short, help describe and understand the features of a specific data set by giving short summaries about the sample and measures of the data. The most recognized types of descriptive statistics are measures of center: the mean, median, and mode, which are used at almost all levels of math and statistics. The mean, or the average, is calculated by adding all the figures within the data set and then dividing by the number of figures within the set.

For example, the sum of the following data set is 20: (2, 3, 4, 5, 6). The mean is 4 (20/5). The mode of a data set is the value appearing most often, and the median is the figure situated in the middle of the data set. It is the figure separating the higher figures from the lower figures within a data set. However, there are fewer common types of descriptive statistics that are still very important.

People use descriptive statistics to repurpose hard-to-understand quantitative insights across a large data set into bite-sized descriptions. A student's grade point average (GPA), for example, provides a good understanding of descriptive statistics. The idea of a GPA is that it takes data points from a wide range of exams, classes, and grades, and averages them together to provide a general understanding of a student's overall academic performance. A student's personal GPA reflects their mean academic performance.

Types of Descriptive Statistics

All descriptive statistics are either measures of central tendency or measures of variability, also known as measures of dispersion.

Measure of Central Tendency

Measures of central tendency focus on the average or middle values of data sets, whereas measures of variability focus on the dispersion of data. These two measures use graphs, tables and general discussions to help people understand the meaning of the analyzed data.

Measures of central tendency describe the center position of a distribution for a data set. A person analyses the frequency of each data point in the distribution and describes it using the mean, median, or mode, which measures the most common patterns of the analyzed data set.
There are three main measures of central tendency:

1. **Mean:** It is the sum of the observation divided by the sample size. It is not a robust statistic as it is affected by extreme values. So, very large or very low value (i.e., Outliers) can distort the answer.

$$Mean = Sum\ of\ all\ data\ values / number\ of\ data\ values$$

2. **Median:** It is the middle value of data. It splits the data in half and also called 50th percentile. It is much less affected by the outliers and skewed data than mean. If the no. of elements in the dataset is odd, the middle most element is the median. If the no. of elements in the dataset is even, the median would be the average of two central elements.

$$Median = (n+1)/2$$

3. **Mode:** It is the value that occurs more frequently in a dataset. Therefore, a dataset has no mode, if no category is the same and also possible that a dataset has more than one mode. It is the only measure of central tendency that can be used for categorical variables.

$$Mode\ = L\ +\ (fm - f1)h\ /(fm - f1) + (fm - f2)$$

Were,
L = Lower limit Mode of modal class
fm = Frequency of modal class
f1 = Frequency of class preceding the modal class
f2= Frequency of class succeeding the modal class
h = Size of class interval

Measures of Variability

Measures of variability (or the measures of spread) aid in analyzing how dispersed the distribution is for a set of data. For example, while the measures of central tendency may give a person the average of a data set, it does not describe how the data is distributed within the set.
So, while the average of the data maybe 65 out of 100, there can still be data points at both 1 and 100. Measures of variability help communicate this by describing the shape and spread of the data set. Range, quartiles, absolute deviation, and variance are all examples of measures of variability. Consider the following data set: 5, 19, 24, 62, 91, 100. The range of that data set is 95, which is calculated by subtracting the lowest number (5) in the data set from the highest (100).

Why do we need statistics that simply describe data?

Descriptive statistics are used to describe or summarize the characteristics of a sample or data set, such as a variable's mean, standard deviation, or frequency. Inferential statistics can help us understand the collective properties of the elements of a data sample. Knowing the sample mean, variance, and distribution of a variable can help us understand the world around us.

What are mean and standard deviation?

These are two commonly employed descriptive statistics. Mean is the average level observed in some piece of data, while standard deviation describes the variance, or how dispersed the data observed in that variable is distributed around its mean.

Can descriptive statistics be used to make inference or prediction?

No. While these descriptive helps understand data attributes, inferential statistical techniques—a separate branch of statistics—are required to understand how variables interact with one another in a data set.

Measures of Spread

Introduction

A measure of spread, sometimes also called a measure of dispersion, is used to describe the variability in a sample or population. It is usually used in conjunction with a measure of central tendency, such as the mean or median, to provide an overall description of a set of data.

When can we measure spread?

The spread of the values can be measured for quantitative data, as the variables are numeric and can be arranged into a logical order with a low-end value and a high end value.

Why is it important to measure the spread of data?

There are many reasons why the measure of the spread of data values is important, but one of the main reasons regards its relationship with measures of central tendency. A measure of spread gives us an idea of how well the mean, for example, represents the data. If the spread of values in the data set is large, the mean is not as representative of the data as if the spread of data is small. This is because a large spread indicates that there are probably large differences between individual scores. The most common are:

1. The range (including the interquartile range and the interdecile range),
2. The standard deviation,
3. The variance,
4. Quartiles.

Calculating the Range

Dataset A - 4, 5, 5, 5, 6, 6, 6, 6, 7, 7, 7, 8

The range is 4, the difference between the highest value (8) and the lowest value (4).

Dataset B - 1, 2, 3, 4, 5, 6, 6, 7, 8, 9, 10, 11

The range is 10, the difference between the highest value (11) and the lowest value (1).

Dataset A

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

Dataset B

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

On a number line, you can see that the range of values for Dataset B is larger than Dataset A. Quartiles divide an ordered dataset into four equal parts, and refer to the values of the point between the quarters. A dataset may also be divided into quintiles (five equal parts) or deciles (ten equal parts).

Quartiles

25% of values Q1 25% of values Q2 25% of values Q3 25% of values

The lower quartile (Q1) is the point between the lowest 25% of values and the highest 75% of values. It is also called the 25th percentile.

The second quartile (Q2) is the middle of the data set. It is also called the 50th percentile, or the median.

The upper quartile (Q3) is the point between the lowest 75% and highest 25% of values. It is also called the 75th percentile.

Calculating Quartiles

Dataset A

4 5 5 Q1 5 6 6 Q2 6 6 7 Q3 7 7 8

As the quartile point falls between two values, the mean (average) of those values is the quartile value:

Q1 = (5+5) / 2 = 5

Q2 = (6+6) / 2 = 6

Q3 = (7+7) / 2 = 7

Dataset B

1 2 3 Q1 4 5 6 Q2 6 7 8 Q3 9 10 11

As the quartile point falls between two values, the mean (average) of those values is the quartile value:

Q1 = (3+4) / 2 = 3.5

Q2 = (6+6) / 2 = 6

Q3 = (8+9) / 2 = 8.5

The interquartile range (IQR) is the difference between the upper (Q3) and lower (Q1) quartiles, and describes the middle 50% of values when ordered from lowest to highest. The IQR is often seen as a better measure of spread than the range as it is not affected by outliers.

Interquartile Range

25% of values Q1 25% of values Q2 25% of values Q3 25% of values

Calculating the Interquartile Range

The IQR for Dataset A is = 2

IQR = Q3 - Q1 = 7 – 5 = 2

The IQR for Dataset B is = 5

IQR = Q3 - Q1 = 8.5 - 3.5 = 5

The variance and the standard deviation are measures of the spread of the data around the mean. They summaries how close each observed data value is to the mean value.

In datasets with a small spread all values are very close to the mean, resulting in a small variance and standard deviation. Where a dataset is more dispersed, values are spread further away from the mean, leading to a larger variance and standard deviation.

The smaller the variance and standard deviation, the more the mean value is indicative of the whole dataset. Therefore, if all values of a dataset are the same, the standard deviation and variance are zero.

The standard deviation of a normal distribution enables us to calculate confidence intervals. In a normal distribution, about 68% of the values are within one standard deviation either side of the mean and about 95% of the scores are within two standard deviations of the mean.

The population Variance σ2 (pronounced sigma squared) of a discrete set of numbers is expressed by the following formula:

$$\sigma^2 = \frac{\sum_{i=1}^{N}(X_i - \mu)^2}{N}$$

where:

Xi represents the ith unit, starting from the first observation to the last

μ represents the population mean

N represents the number of units in the population

The Variance of a sample s2 (pronounced s squared) is expressed by a slightly different formula:

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

where:

xi represents the ith unit, starting from the first observation to the last

$\bar{x}$ represents the sample mean

n represents the number of units in the sample

The standard deviation is the square root of the variance. The standard deviation for a population is represented by σ, and the standard deviation for a sample is represented by s.

Calculating the Population Variance σ2 and Standard Deviation σ

Dataset A

Calculate the population mean (μ) of Dataset A.

(4 + 5 + 5 + 5 + 6 + 6 + 6 + 6 + 7 + 7 + 7 + 8) / 12

mean (μ) = 6

Calculate the deviation of the individual values from the mean by subtracting the mean from each value in the dataset $X_i - \mu$ = -2, -1, -1, -1, 0, 0, 0, 0, 1, 1, 1, 2

Square each individual deviation value $(X_i - \mu)^2$ = 4, 1, 1, 1, 0, 0, 0, 0, 1,1,1, 4

Calculate the mean of the squared deviation values

$$\frac{\sum_{i=1}^{N}(X_i - \mu)^2}{N} = (4 + 1 + 1 + 1 + 0 + 0 + 0 + 0 + 1 + 1 + 1 + 4) / 12$$

Variance σ2= 1.17

Calculate the square root of the variance

Standard deviation σ = 1.08

Dataset B

Calculate the population mean (μ) of Dataset B.

(1 + 2 + 3 + 4 + 5 + 6 + 6 + 7 + 8 + 9 + 10 + 11) / 12

mean (μ) = 6

Calculate the deviation of the individual values from the mean by subtracting the mean from each value in the dataset $X_i - \mu$ = -5, -4, -3, -2, -1, 0, 0, 1, 2, 3, 4, 5,

Square each individual deviation value

$(X_i - \mu)^2$ = 25, 16, 9, 4, 1, 0, 0, 1, 4, 9, 16, 25

Calculate the mean of the squared deviation values

$$\frac{\sum_{i=1}^{N}(X_i - \mu)^2}{N} = (25 + 16 + 9 + 4 + 1 + 0 + 0 + 1 + 4 + 9 + 16 + 25) / 12$$

Variance σ2 = 9.17

Calculate the square root of the variance

Standard deviation σ = 3.03

The larger Variance and Standard Deviation in Dataset B further demonstrates that Dataset B is more dispersed than Dataset A.

Inferential Statistics basics

What is Inferential Statistic?

If you see based on the language, inferential means can be concluded.
In general, inferential statistics are a type of statistics that focus on processing sample data so that they can make decisions or conclusions on the population.
Inferential statistics focus on analyzing sample data to infer the population.
The flow of using inferential statistics is the sampling method, data analysis, and decision making for the entire population.
Inferential statistics are used by many people (especially scientist and researcher) because they are able to produce accurate estimates at a relatively affordable cost.

Advantages of using inferential statistics

Inferential statistics have different benefits and advantages.

1. A precise tool for estimating population - The main purpose of using inferential statistics is to estimate population values. With the use of this method, of course, we expect accurate and precise measurement results and are able to describe the actual conditions.

2. Highly structured analytical methods - Inferential statistics have a very neat formula and structure. The method used is tested mathematically and can be regarded as an unbiased estimator.

Knowing and using inferential statistics

Inferential Statistics Examples

There are lots of examples of applications and the application of inferential statistics in life. However, in general, the inferential statistics that are often used are:

1. Regression Analysis - Regression analysis is one of the most popular analysis tools. Regression analysis is used to predict the relationship between independent variables and the dependent variable.
Using this analysis, we can determine which variables have a significant effect in a study.

For example, you want to know what factors can influence the decline in poverty. You use variables such as road length, economic growth, electrification ratio, number of teachers, number of medical personnel, etc.
After analysis, you will find which variables have an influence in reducing the poverty rate.
2. Hypothesis test - Hypothesis testing is a statistical test where we want to know the truth of an assumption or opinion that is common in society. Usually, this test is used to find out about the truth of a claim circulating in the community.
Hypothesis testing also helps us to prove whether the opinions or things we believe are true or false.
For example, we often hear the assumption that female students tend to have higher mathematical values than men. Is that right?
To prove this, you can take a representative sample and analyze the mathematical values of the samples taken.
By using a hypothesis test, you can draw conclusions about the actual conditions.

Can you use the entire data on the overall mathematics value of students and analyze the data? Certainly, very allowed.
But, of course, you will need a longer time in reaching conclusions because the data collection process also requires substantial time.
3. Confidence Interval - Confidence interval or confidence level is a statistical test used to estimate the population by using samples. With this level of trust, we can estimate with a greater probability what the actual population value is.
When using confidence intervals, we will find the upper and lower limits of a statistical test that we believe there is a population value we estimate.
When we use 95 percent confidence intervals, it means we believe that the test statistics we use are within the range of values we have obtained based on the formula.
For example, we want to estimate what the average expenditure is for everyone in city X. Therefore, research is conducted by taking a number of samples. The results of this study certainly vary.
Therefore, we must determine the estimated range of the actual expenditure of each person. The hope is, of course, the actual average value will fall in the range of values that we have calculated before.
At the last part of this article, I will show you how confidence interval works as inferential statistics examples.
4. Time series analysis - As you know, one type of data based on time is time series data. Sometimes, often a data occurs repeatedly or has special and common patterns so it is very interesting to study more deeply.

Time series analysis is one type of statistical analysis that tries to predict an event in the future based on pre-existing data. With this method, we can estimate how predictions a value or event that appears in the future.
Example: every year, policymakers always estimate economic growth, both quarterly and yearly. By using time series analysis, we can use data from 20 to 30 years to estimate how economic growth will be in the future.

Procedure for using inferential statistics

1. Determine the population data that we want to examine
2. Determine the number of samples that are representative of the population
3. Select an analysis that matches the purpose and type of data we have
4. Make conclusions on the results of the analysis

Differences in Inferential Statistics and Descriptive Statistics
Inferential statistics and descriptive statistics have very basic differences in the analysis process. In general, these two types of statistics also have different objectives.

1. Descriptive statistics aim to describe the characteristics of the data. While statistical inferencing aims to draw conclusions for the population by analyzing the sample.
2. Descriptive statistics are usually only presented in the form of tables and graphs. The test statistics used are fairly simple, such as averages, variances, etc. While inferential statistics, the statistics used are classified as very complicated. Not everyone is able to use inferential statistics so special seriousness and learning are needed before using it.

Therefore, we cannot use any analytical tools available in descriptive analysis to infer the overall data.

How to make inferential statistics as a stronger tool?
Probably, the analyst knows several things that can influence inferential statistics in order to produce accurate estimates. The main key is good sampling.
Samples taken must be random or random. That is, there should not be certain trends in taking who, what, and how the condition of the sample.
The selected sample must also meet the minimum sample requirements. Actually, there is no specific requirement for the number of samples that must be used to be able to represent the population. However, many experts agree that the number of samples used must be at least 30 units.
Samples must also be able to meet certain distributions. Usually, the commonly used sample distribution is a normal distribution. Although sometimes, there are cases where other distributions are indeed more suitable.

Make sure the above three conditions are met so that your analysis results don't disappoint later.

Why we should use inferential statistics?

Case Study of Inferential Statistics

There are several types of inferential statistics examples that you can use. But in this case, I will just give an example using statistical confidence intervals.

Suppose a regional head claims that the poverty rate in his area is very low. To prove this, he conducted a household income and expenditure survey that was theoretically able to produce poverty.

Considering the survey period and budget, 10,000 household samples were selected from a total of 100,000 households in the district.

Based on the survey results, it was found that there were still 5,000 poor people. Of course, this number is not entirely true considering the survey always has errors.

Therefore, confidence intervals were made to strengthen the results of this survey.

Based on the results of calculations, with a confidence level of 95 percent and the standard deviation is 500, it can be concluded that the number of poor people in the city ranges from 4,990 to 5010 people.

Types of Distributions

Bernoulli Distribution

Let's start with the easiest distribution that is Bernoulli Distribution. It is actually easier to understand than it sounds!

All you cricket junkies out there! At the beginning of any cricket match, how do you decide who is going to bat or ball? A toss! It all depends on whether you win or lose the toss, right? Let's say if the toss results in a head, you win. Else, you lose. There's no midway.

A **Bernoulli distribution** has only two possible outcomes, namely 1 (success) and 0 (failure), and a single trial. So, the random variable X which has a Bernoulli distribution can take value 1 with the probability of success, say p, and the value 0 with the probability of failure, say q or 1-p.

Here, the occurrence of a head denotes success, and the occurrence of a tail denotes failure.

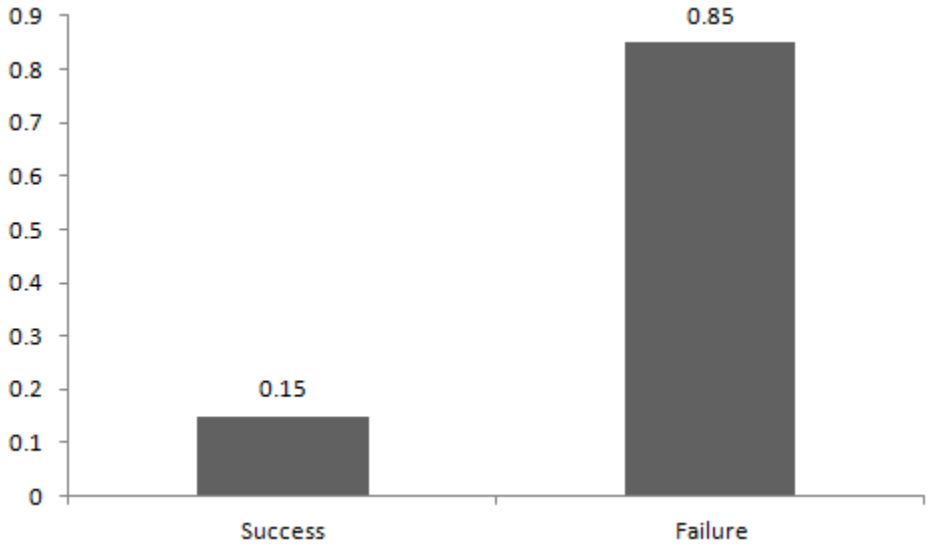
Probability of getting a head = 0.5 = Probability of getting a tail since there are only two possible outcomes.

The probability mass function is given by: px(1-p)1-x where x € (0, 1).

It can also be written as

$$P(x) = \begin{cases} 1-p, & x = 0 \\ p, & x = 1 \end{cases}$$

The probabilities of success and failure need not be equally likely, like the result of a fight between me and Undertaker. He is pretty much certain to win. So in this case probability of my success is 0.15 while my failure is 0.85



Here, the probability of success(p) is not same as the probability of failure. So, the chart below shows the Bernoulli Distribution of our fight.

Here, the probability of success = 0.15 and probability of failure = 0.85. The expected value is exactly what it sounds. If I punch you, I may expect you to punch me back. Basically expected value of any distribution is the mean of the distribution. The expected value of a random variable X from a Bernoulli distribution is found as follows:

$E(X) = 1*p + 0*(1-p) = p$

The variance of a random variable from a bernoulli distribution is:

$V(X) = E(X^2) – [E(X)]^2 = p – p^2 = p(1-p)$

There are many examples of Bernoulli distribution such as whether it's going to rain tomorrow or not where rain denotes success and no rain denotes failure and Winning (success) or losing (failure) the game.

Uniform Distribution

When you roll a fair die, the outcomes are 1 to 6. The probabilities of getting these outcomes are equally likely and that is the basis of a uniform distribution. Unlike Bernoulli Distribution, all the n number of possible outcomes of a uniform distribution are equally likely.

A variable X is said to be uniformly distributed if the density function is:

$$f(x) = \frac{1}{b-a} \qquad for\ -\infty < a \leq x \leq b < \infty$$

The graph of a uniform distribution curve looks like

You can see that the shape of the Uniform distribution curve is rectangular, the reason why Uniform distribution is called rectangular distribution.

For a Uniform Distribution, a and b are the parameters.

The number of bouquets sold daily at a flower shop is uniformly distributed with a maximum of 40 and a minimum of 10.

Let's try calculating the probability that the daily sales will fall between 15 and 30.

The probability that daily sales will fall between 15 and 30 is (30-15) *(1/ (40-10)) = 0.5

Similarly, the probability that daily sales are greater than 20 is = 0.667

The mean and variance of X following a uniform distribution is:

Mean -> E(X) = (a+b)/2

Variance -> V(X) = $(b-a)^2/12$

The standard uniform density has parameters a = 0 and b = 1, so the PDF for standard uniform density is given by:

$$f(x) = \begin{cases} 1, & 0 \leq x \leq 1 \\ 0, & otherwise \end{cases}$$

Binomial Distribution

Let's get back to cricket. Suppose that you won the toss today and this indicates a successful event. You toss again but you lost this time. If you win a toss today, this does not necessitate that you will win the toss tomorrow. Let's assign a random variable, say X, to the number of times you won the toss. What can be the possible value of X? It can be any number depending on the number of times you tossed a coin.

There are only two possible outcomes. Head denoting success and tail denoting failure. Therefore, probability of getting a head = 0.5 and the probability of failure can be easily computed as: q = 1- p = 0.5.

A distribution where only two outcomes are possible, such as success or failure, gain or loss, win or lose and where the probability of success and failure is same for all the trials is called a Binomial Distribution.

The outcomes need not be equally likely. Remember the example of a fight between me and Undertaker? So, if the probability of success in an experiment is 0.2 then the probability of failure can be easily computed as q = 1 – 0.2 = 0.8.

Each trial is independent since the outcome of the previous toss doesn't determine or affect the outcome of the current toss. An experiment with only two possible outcomes repeated n number of times is called binomial. The parameters of a binomial distribution are n and p where n is the total number of trials and p is the probability of success in each trial.

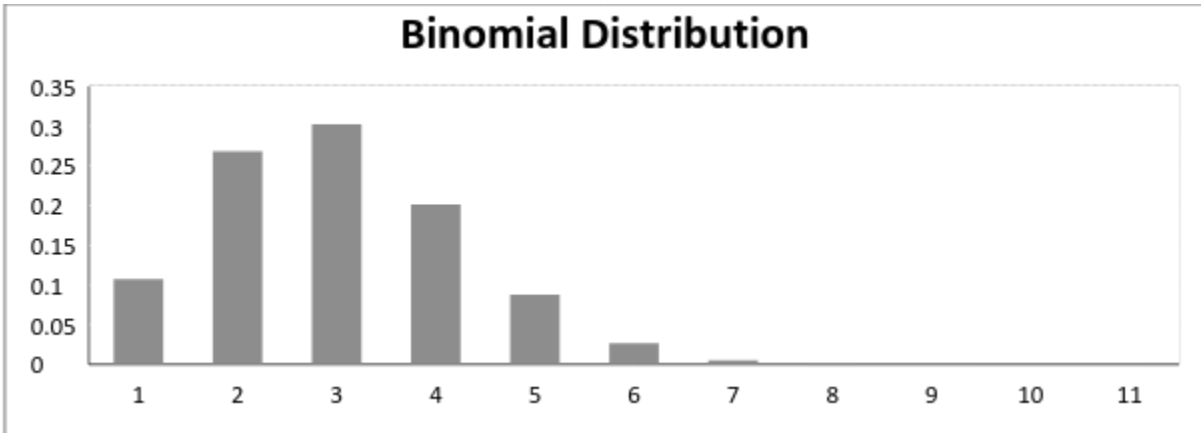
On the basis of the above explanation, the properties of a Binomial Distribution are

1. Each trial is independent.
2. There are only two possible outcomes in a trial- either a success or a failure.
3. A total number of n identical trials are conducted.
4. The probability of success and failure is same for all trials. (Trials are identical.)

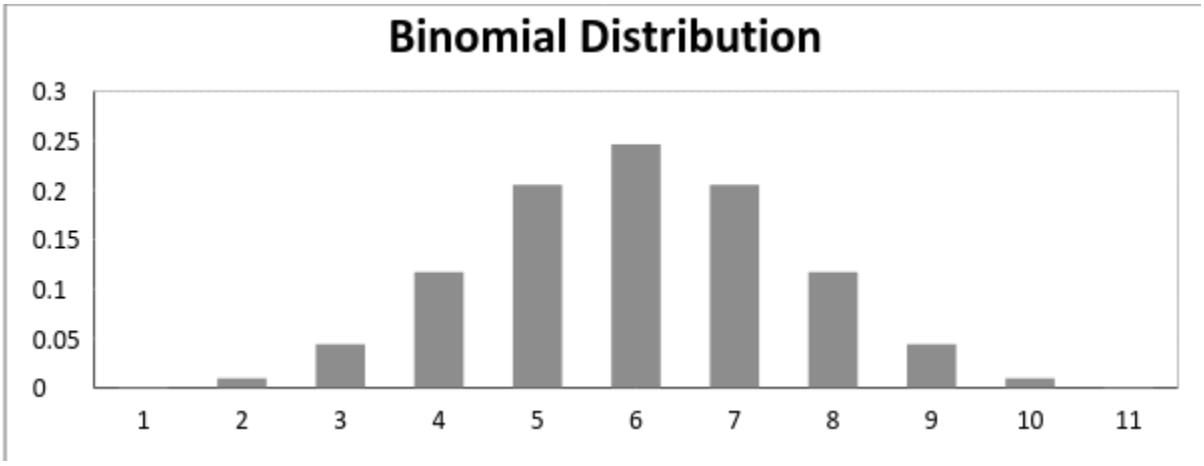
The mathematical representation of binomial distribution is given by:

$$P(x) = \frac{n!}{(n-x)!x!} p^x q^{n-x}$$

A binomial distribution graph where the probability of success does not equal the probability of failure looks like



Now, when probability of success = probability of failure, in such a situation the graph of binomial distribution looks like



The mean and variance of a binomial distribution are given by:

Mean -> $\mu = n*p$

Variance -> $Var(X) = n*p*q$

Normal Distribution

Normal distribution represents the behavior of most of the situations in the universe (That is why it's called a "normal" distribution. I guess!). The large sum of (small) random variables often turns out to be normally distributed, contributing to its widespread application. Any distribution is known as Normal distribution if it has the following characteristics:

1. The mean, median and mode of the distribution coincide.
2. The curve of the distribution is bell-shaped and symmetrical about the line $x=\mu$.
3. The total area under the curve is 1.
4. Exactly half of the values are to the left of the center and the other half to the right.

A normal distribution is highly different from Binomial Distribution. However, if the number of trials approaches infinity, then the shapes will be quite similar.

The PDF of a random variable X following a normal distribution is given by:

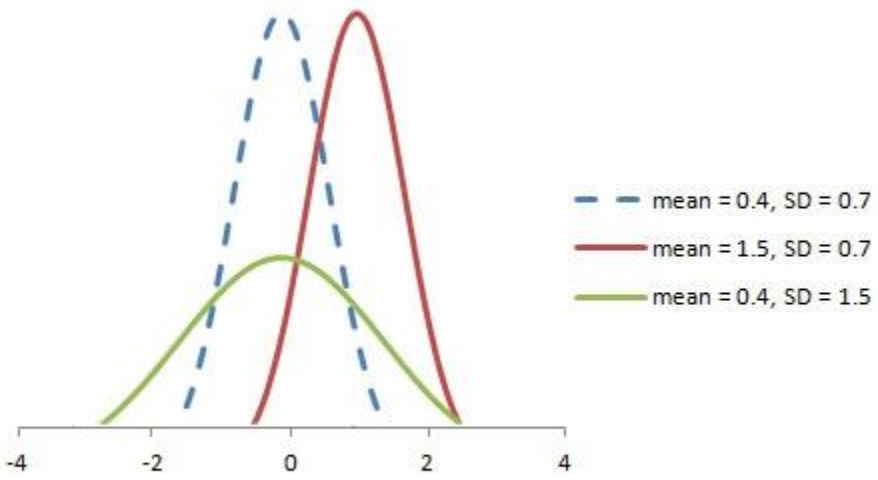
$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{\{-\frac{1}{2}(\frac{x-u}{\sigma})^2\}} \qquad for -\infty < x < \infty.$$

The mean and variance of a random variable X which is said to be normally distributed is given by:

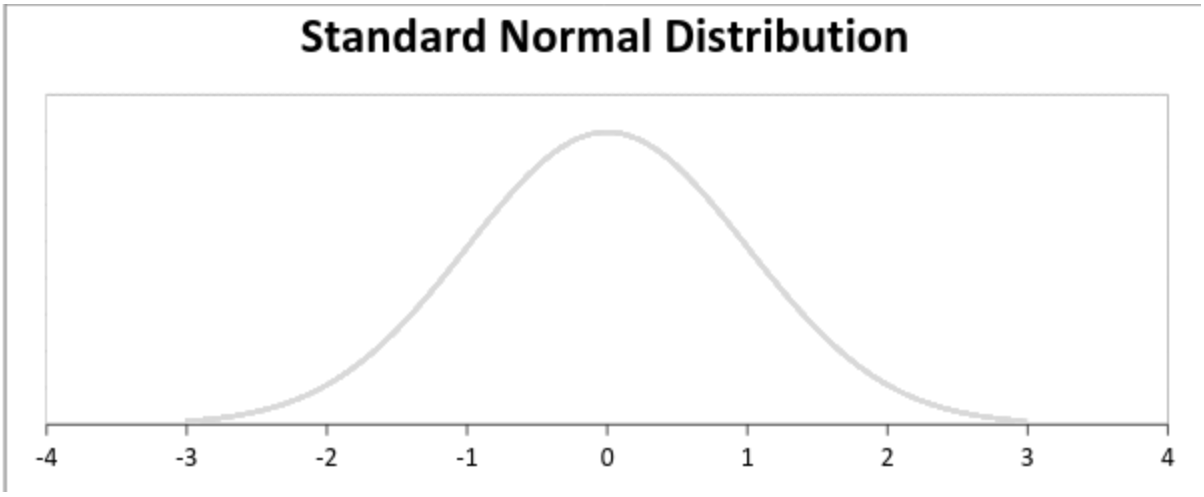
Mean -> E(X) = μ

Variance -> Var(X) = σ^2

Here, μ (mean) and σ (standard deviation) are the parameters.
The graph of a random variable X ~ N (μ, σ) is shown below.



A standard normal distribution is defined as the distribution with mean 0 and standard deviation 1. For such a case, the PDF becomes:



Poisson Distribution

Suppose you work at a call center; approximately how many calls do you get in a day? It can be any number. Now, the entire number of calls at a call center in a day is modeled by Poisson distribution. Some more examples are

1. The number of emergency calls recorded at a hospital in a day.
2. The number of thefts reported in an area on a day.
3. The number of customers arriving at a salon in an hour.
4. The number of suicides reported in a particular city.
5. The number of printing errors at each page of the book.

You can now think of many examples following the same course. Poisson Distribution is applicable in situations where events occur at random points of time and space wherein our interest lies only in the number of occurrences of the event.

A distribution is called Poisson distribution when the following assumptions are valid:

1. Any successful event should not influence the outcome of another successful event.
2. The probability of success over a short interval must equal the probability of success over a longer interval.
3. The probability of success in an interval approaches zero as the interval becomes smaller.

Now, if any distribution validates the above assumptions then it is a Poisson distribution. Some notations used in Poisson distribution are:

- λ is the rate at which an event occurs,
- t is the length of a time interval,
- And X is the number of events in that time interval.

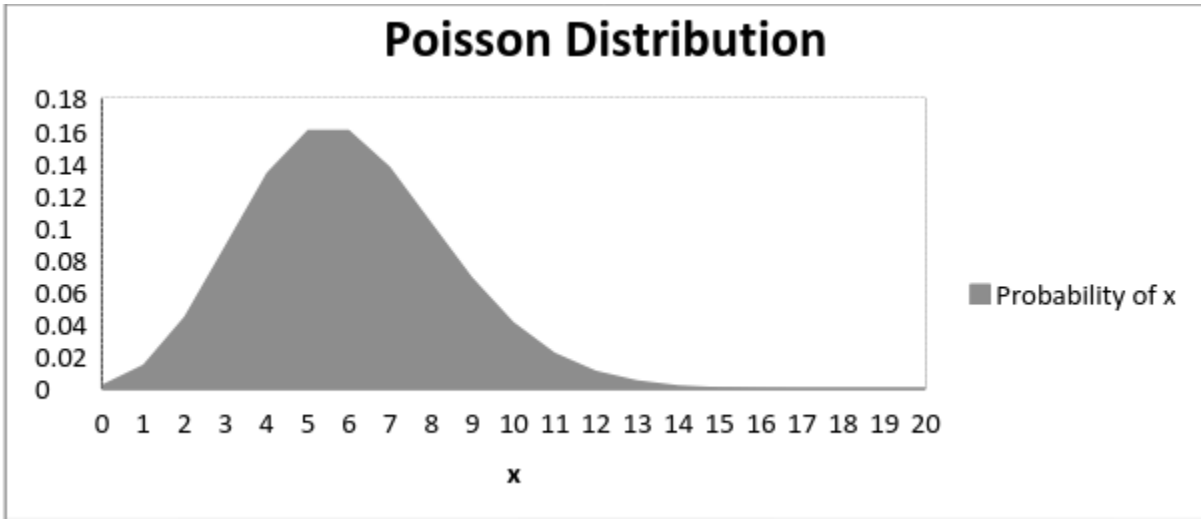
Here, X is called a Poisson Random Variable and the probability distribution of X is called Poisson distribution.

Let μ denote the mean number of events in an interval of length t. Then, μ = λ*t.

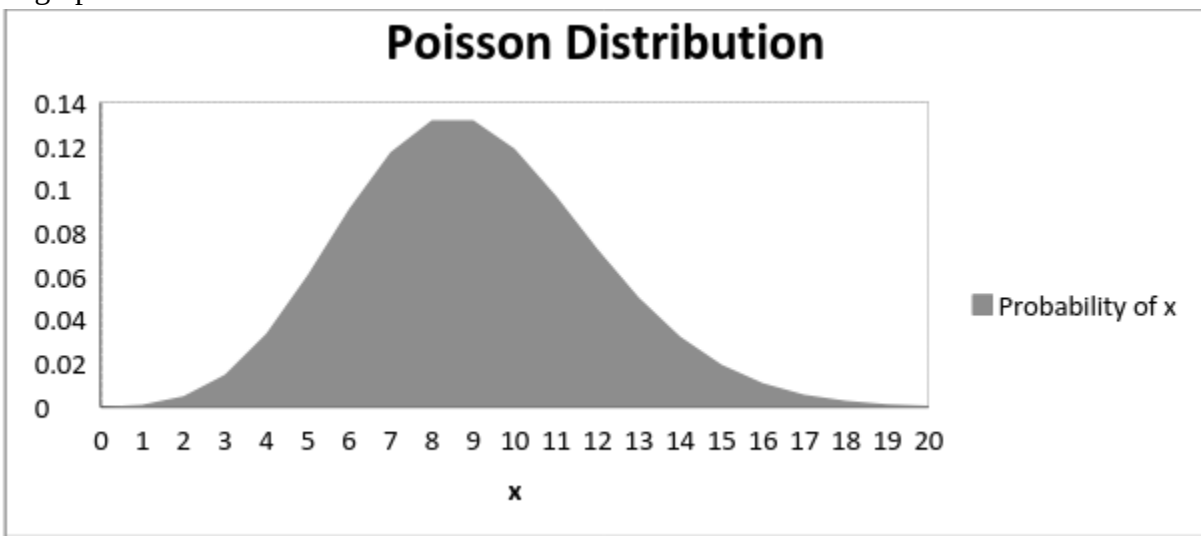
The PMF of X following a Poisson distribution is given by:

$$P(X = x) = e^{-\mu}\frac{\mu^x}{x!} \qquad for\ x = 0, 1, 2, \ldots..$$

The mean μ is the parameter of this distribution. μ is also defined as the λ times length of that interval. The graph of a Poisson distribution is shown below:



The graph shown below illustrates the shift in the curve due to increase in mean.



It is perceptible that as the mean increases, the curve shifts to the right.
The mean and variance of X following a Poisson distribution:

$$\text{Mean} \rightarrow E(X) = \mu$$

$$\text{Variance} \rightarrow Var(X) = \mu$$

Exponential Distribution

Let's consider the call centre example one more time. What about the interval of time between the calls? Here, exponential distribution comes to our rescue. Exponential distribution models the interval of time between the calls.
Other examples are:

1. Length of time between metro arrivals,

2. Length of time between arrivals at a gas station
3. The life of an Air Conditioner

Exponential distribution is widely used for survival analysis. From the expected life of a machine to the expected life of a human, exponential distribution successfully delivers the result.

A random variable X is said to have an exponential distribution with PDF:

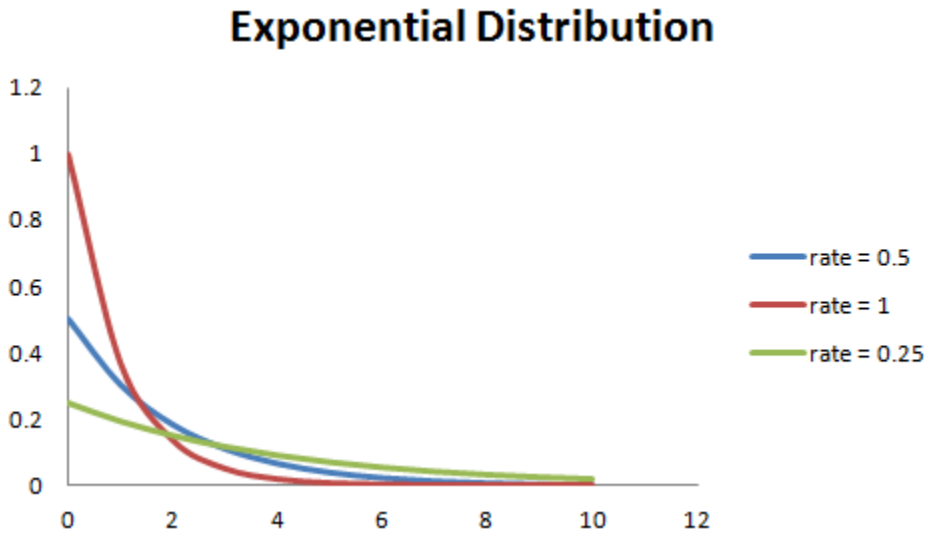
$$f(x) = \{ \lambda e^{-\lambda x}, \quad x \geq 0$$

and parameter $\lambda > 0$ which is also called the rate.
For survival analysis, λ is called the failure rate of a device at any time t, given that it has survived up to t.
Mean and Variance of a random variable X following an exponential distribution:

$$\text{Mean} \rightarrow E(X) = 1/\lambda$$

$$\text{Variance} \rightarrow Var(X) = (1/\lambda)^2$$

Also, the greater the rate, the faster the curve drops and the lower the rate, flatter the curve. This is explained better with the graph shown below.



To ease the computation, there are some formulas given below.
$P\{X \leq x\} = 1 - e^{-\lambda x}$, corresponds to the area under the density curve to the left of x.
$P\{X > x\} = e^{-\lambda x}$, corresponds to the area under the density curve to the right of x.
$P\{x1 < X \leq x2\} = e^{-\lambda x1} - e^{-\lambda x2}$, corresponds to the area under the density curve between x1 and x2.

What is sampling?
Sampling is a technique of selecting individual members or a subset of the population to make statistical inferences from them and estimate characteristics of the whole population. Different

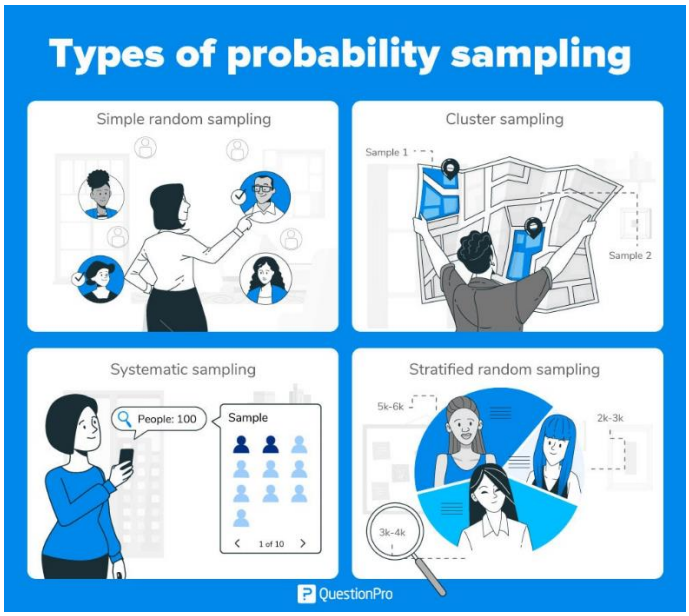
sampling methods are widely used by researchers in market research so that they do not need to research the entire population to collect actionable insights.
It is also a time-convenient and a cost-effective method and hence forms the basis of any research design. Sampling techniques can be used in a research survey software for optimum derivation.
For example, if a drug manufacturer would like to research the adverse side effects of a drug on the country's population, it is almost impossible to conduct a research study that involves everyone. In this case, the researcher decides a sample of people from each demographic and then researches them, giving him/her indicative feedback on the drug's behavior.

Types of sampling: sampling methods
Sampling in market research is of two types – probability sampling and non-probability sampling. Let's take a closer look at these two methods of sampling.

Probability sampling: Probability sampling is a sampling technique where a researcher sets a selection of a few criteria and chooses members of a population randomly. All the members have an equal opportunity to be a part of the sample with this selection parameter.

Non-probability sampling: In non-probability sampling, the researcher chooses members for research at random. This sampling method is not a fixed or predefined selection process. This makes it difficult for all elements of a population to have equal opportunities to be included in a sample.

Types of probability sampling with examples:
Probability sampling is a sampling technique in which researchers choose samples from a larger population using a method based on the theory of probability. This sampling method considers every member of the population and forms samples based on a fixed process.
For example, in a population of 1000 members, every member will have a 1/1000 chance of being selected to be a part of a sample. Probability sampling eliminates bias in the population and gives all members a fair chance to be included in the sample.
There are four types of probability sampling techniques:



Types of probability sampling:

Simple random sampling: One of the best probability sampling techniques that helps in saving time and resources, is the Simple Random Sampling method. It is a reliable method of obtaining information where every single member of a population is chosen randomly, merely by chance. Each individual has the same probability of being chosen to be a part of a sample.

For example, in an organization of 500 employees, if the HR team decides on conducting team building activities, it is highly likely that they would prefer picking chits out of a bowl. In this case, each of the 500 employees has an equal opportunity of being selected.

Cluster sampling: Cluster sampling is a method where the researchers divide the entire population into sections or clusters that represent a population. Clusters are identified and included in a sample based on demographic parameters like age, sex, location, etc. This makes it very simple for a survey creator to derive effective inference from the feedback.

For example, if the United States government wishes to evaluate the number of immigrants living in the mainland US, they can divide it into clusters based on states such as California, Texas, Florida, Massachusetts, Colorado, Hawaii, etc. This way of conducting a survey will be more effective as the results will be organized into states and provide insightful immigration data.

Systematic sampling: Researchers use the systematic sampling method to choose the sample members of a population at regular intervals. It requires the selection of a starting point for the sample and sample size that can be repeated at regular intervals. This type of sampling method has a predefined range, and hence this sampling technique is the least time-consuming.

For example, a researcher intends to collect a systematic sample of 500 people in a population of 5000. He/she numbers each element of the population from 1-5000 and will choose every 10th individual to be a part of the sample (Total population/ Sample Size = 5000/500 = 10).

Stratified random sampling: Stratified random sampling is a method in which the researcher divides the population into smaller groups that don't overlap but represent the entire population. While sampling, these groups can be organized and then draw a sample from each group separately.

For example, a researcher looking to analyze the characteristics of people belonging to different annual income divisions will create strata (groups) according to the annual family income. Eg – less than $20,000, $21,000 – $30,000, $31,000 to $40,000, $41,000 to $50,000, etc. By doing this, the researcher concludes the characteristics of people belonging to different income groups. Marketers can analyze which income groups to target and which ones to eliminate to create a roadmap that would bear fruitful results.

Uses of probability sampling

There are multiple uses of probability sampling:

Reduce Sample Bias: Using the probability sampling method, the bias in the sample derived from a population is negligible to non-existent. The selection of the sample mainly depicts the understanding and the inference of the researcher. Probability sampling leads to higher quality data collection as the sample appropriately represents the population.

Diverse Population: When the population is vast and diverse, it is essential to have adequate representation so that the data is not skewed towards one demographic. For example, if Square would like to understand the people that could make their point-of-sale devices, a survey conducted from a sample of people across the US from different industries and socio-economic backgrounds helps.
Create an Accurate Sample: Probability sampling helps the researchers plan and create an accurate sample. This helps to obtain well-defined data.

Types of non-probability sampling with examples

The non-probability method is a sampling method that involves a collection of feedback based on a researcher or statistician's sample selection capabilities and not on a fixed selection process. In most situations, the output of a survey conducted with a non-probable sample leads to skewed results, which may not represent the desired target population. But, there are situations such as the preliminary stages of research or cost constraints for conducting research, where non-probability sampling will be much more useful than the other type.
Four types of non-probability sampling explain the purpose of this sampling method in a better manner:

Convenience sampling: This method is dependent on the ease of access to subjects such as surveying customers at a mall or passers-by on a busy street. It is usually termed as convenience sampling, because of the researcher's ease of carrying it out and getting in touch with the subjects. Researchers have nearly no authority to select the sample elements, and it's purely done based on proximity and not representativeness. This non-probability sampling method is used when there are time and cost limitations in collecting feedback. In situations where there are resource limitations such as the initial stages of research, convenience sampling is used.
For example, startups and NGOs usually conduct convenience sampling at a mall to distribute leaflets of upcoming events or promotion of a cause – they do that by standing at the mall entrance and giving out pamphlets randomly.

Judgmental or purposive sampling: Judgemental or purposive samples are formed by the discretion of the researcher. Researchers purely consider the purpose of the study, along with the understanding of the target audience. For instance, when researchers want to understand the thought process of people interested in studying for their master's degree. The selection criteria will be: "Are you interested in doing your masters in …?" and those who respond with a "No" are excluded from the sample.

Snowball sampling: Snowball sampling is a sampling method that researchers apply when the subjects are difficult to trace. For example, it will be extremely challenging to survey shelterless people or illegal immigrants. In such cases, using the snowball theory, researchers can track a few categories to interview and derive results. Researchers also implement this sampling method in situations where the topic is highly sensitive and not openly discussed—for example, surveys to gather information about HIV Aids. Not many victims will readily respond to the questions. Still, researchers can contact people they might know or volunteers associated with the cause to get in touch with the victims and collect information.

Quota sampling: In Quota sampling, the selection of members in this sampling technique happens based on a pre-set standard. In this case, as a sample is formed based on specific attributes, the created sample will have the same qualities found in the total population. It is a rapid method of collecting samples.

Uses of non-probability sampling

Non-probability sampling is used for the following:
Create a hypothesis: Researchers use the non-probability sampling method to create an assumption when limited to no prior information is available. This method helps with the immediate return of data and builds a base for further research.
Exploratory research: Researchers use this sampling technique widely when conducting qualitative research, pilot studies, or exploratory research.
Budget and time constraints: The non-probability method when there are budget and time constraints, and some preliminary data must be collected. Since the survey design is not rigid, it is easier to pick respondents at random and have them take the survey or questionnaire.

How do you decide on the type of sampling to use?
For any research, it is essential to choose a sampling method accurately to meet the goals of your study. The effectiveness of your sampling relies on various factors. Here are some steps expert researchers follow to decide the best sampling method.

- Jot down the research goals. Generally, it must be a combination of cost, precision, or accuracy.
- Identify the effective sampling techniques that might potentially achieve the research goals.
- Test each of these methods and examine whether they help in achieving your goal.
- Select the method that works best for the research.

Difference between probability sampling and non-probability sampling methods

We have looked at the different types of sampling methods above and their subtypes. To encapsulate the whole discussion, though, the significant differences between probability sampling methods and non-probability sampling methods are as below:

	Probability Sampling Methods	Non-Probability Sampling Methods
Definition	Probability Sampling is a sampling technique in which samples from a larger population are chosen using a method based on the theory of probability.	Non-probability sampling is a sampling technique in which the researcher selects samples based on the researcher's subjective judgment rather than random selection.
Alternatively Known as	Random sampling method.	Non-random sampling method
Population selection	The population is selected randomly.	The population is selected arbitrarily.
Nature	The research is conclusive.	The research is exploratory.
Sample	Since there is a method for deciding the sample, the population demographics are conclusively represented.	Since the sampling method is arbitrary, the population demographics representation is almost always skewed.
Time Taken	Takes longer to conduct since the research design defines the selection parameters before the market research study begins.	This type of sampling method is quick since neither the sample or selection criteria of the sample are undefined.
Results	This type of sampling is entirely unbiased and hence the results are unbiased too and conclusive.	This type of sampling is entirely biased and hence the results are biased too, rendering the research speculative.

Hypothesis	In probability sampling, there is an underlying hypothesis before the study begins and the objective of this method is to prove the hypothesis.	In non-probability sampling, the hypothesis is derived after conducting the research study.

Probability Theories

What Is Probability Theory?

Probability theory is a branch of mathematics focusing on the analysis of random phenomena. It is an important skill for data scientists using data affected by chance.

With randomness existing everywhere, the use of probability theory allows for the analysis of chance events. The aim is to determine the likelihood of an event occurring, often using a numerical scale of between 0 and 1, with the number "0" indicating impossibility and "1" indicating certainty.

A classic example of this is a coin toss, where there can be two possible options: heads or tails. Here the possibility of flipping a head or a tail on a single toss is 50%. When conducting your own experiment, you may find that the outcomes can vary. But if you continue flipping the coin, the outcome grows closer to 50/50.

Probability plays a vital role in many areas of scientific research. Researchers can integrate uncertainty into their research models as a way of describing their findings. This allows for a predictive distribution of findings tied to what may have been observed in the past.

Randomness and uncertainty are popular themes tied to probability. In Nassim Taleb's bestselling books The Black Swan and Fooled By Randomness, the claim is made that rare events typically hold more importance than common ones because their effect size is not as restricted. Also, because of their rarity, results are unlikely to be determined.

Taleb popularized what he calls a "black swan" event, one that is rare, has a catastrophic impact when it does occur and can be explained in hindsight in a way that leads many to believe that it was actually predictable.

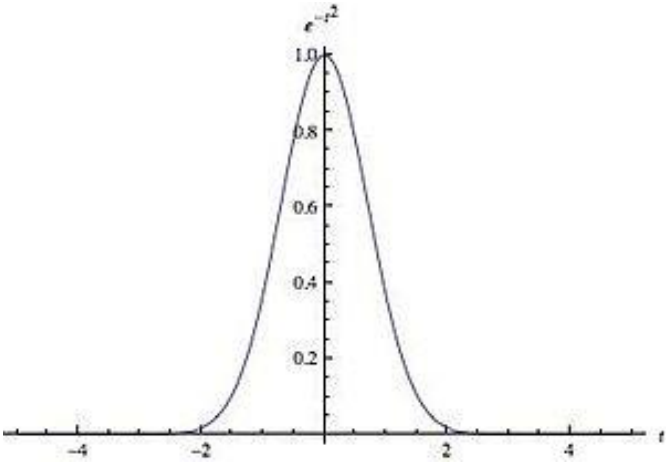


Image 27 - Probability Theory

Reference: https://upload.wikimedia.org/wikipedia/commons/thumb/d/d2/Gaussian_distribution_2.jpg/300px-Gaussian_distribution_2.jpg

Practical Uses for Probability Theory

Probability is commonly used by data scientists to model situations where experiments, conducted during similar circumstances, yield different results (as in the case of throwing dice or a coin).

It also has many practical uses in the business world. Take for example the insurance industry, where actuarial records chart life expectancy of individuals of a certain age. Instead of predicting what will happen to any one individual, the aim is to capture a collective result encompassing a large number of people.

Similar approaches have been taken in genetic science, where assessing the likelihood of a genetic disease is tied to frequency of occurrence as opposed to predictions about a specific individual.

Another common application of probability is also commonly applied in clinical trials where new disease treatments, drugs or surgical treatments are being sought. In assessing whether a treatment can be deemed a success or failure, the clinical trial aims to determine whether the new treatment is more successful than a prevailing treatment standard.

An example here is testing the efficacy of a new vaccine, such as the poliomyelitis testing done for the Salk vaccine in 1954 involving almost two million children. Organized by the U.S. Public Health Service, the vaccine nearly eliminated polio as a health problem in the industrialized world.

What Are the 3 Types of Probability?

There are three types of probability commonly used to gather statistical inference data. These are:

i. Classical
ii. Relative Frequency
iii. Subjective Probability

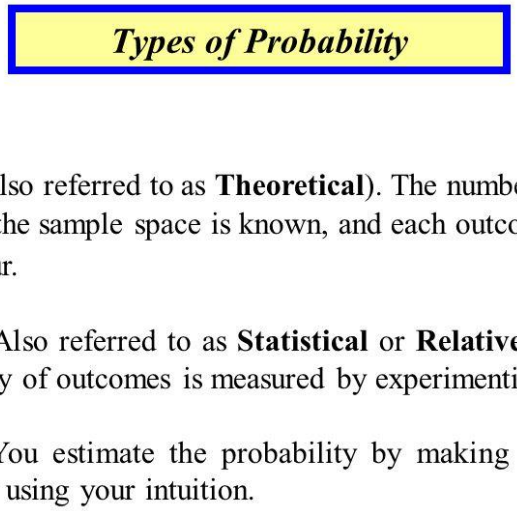


Image 28 - Types Of Probability
Reference: https://slideplayer.com/slide/6379961/22/images/3/Types+of+Probability.jpg

i. **Classical**

Also known as the axiomatic method, this type of probability involves a set of axioms (rules) attached to it. For example, you could have a rule that the probability must be greater than 0.5% in order for it to be valid.

ii. **Relative Frequency**

This involves looking at the occurrence ratio of a singular event in comparison to the total number of outcomes. This type of probability is often used after data from an experiment has been gathered to compare a subset of data to the total amount of collected data.

iii. **Subjective Probability**

When using the subjective approach, probability is the likelihood of something happening based on one's experiences or personal judgment. Here there are no formal calculations for subjective probability for it is based on one's beliefs, judgment and personal reasoning.
By way of example, during a sporting event, fans of one team share who they are rooting for. This is based on facts or opinions they personally hold regarding the game, the two teams playing and the odds of the team winning.

Probability Theory Examples

Probability theory is a tool employed by researchers, businesses, investment analysts and countless others for risk management and scenario analysis.

Epidemiology

Take epidemiology, which is the science of disease distribution. Researchers in this field study disease frequency, assessing how the probability differs across groups of people. A present-day example of this is the use of probability by epidemiologists to assess the cause-effect relationship between exposure and illness to the coronavirus.
Probability theory is often used to unlock key factors denoting the relationship between exposures and health risks. The aim here is to quantify uncertainty. This knowledge can fuel a course of action based on best outcomes for those affected by various diseases.

Insurance

The actuaries who are often employed in the insurance industry make primary use of probability, statistics and other data science tools to calculate the probability of uncertain future events occurring over a period of time. They then apply other data concepts to determine the amount of money that needs to be set aside to pay for future losses.

Small Business

Then there's the small-business world where owners cannot always turn to their hunches and instincts to run a successful company. In today's competitive business environment, probability analysis can provide entrepreneurs with key metrics pointing the way to the most profitable and productive paths. This analysis offers a controlled way to anticipate potential results.

For example, if a business enterprise expects to receive between $500,000 and $750,000 in revenue each month, the graph will begin with $500,000 at the low end and $750,000 at the high end. For a typical probability distribution, the graph will resemble a bell curve, where the least likely outcomes fall nearer the extreme ends of the range and the most likely nearer to the midpoint of the extremes.

Meteorology

A weather forecast serves as another example of probability theory. The probability for precipitation or severe weather is tied to a specific geographic location. As a result, forecasting can be viewed as the combination of the chance of a weather occurrence and the coverage of that event. According to an information statement of the American Meteorological Society:

"A probability forecast includes a numerical expression of uncertainty about the quantity or event being forecast. Ideally, all elements (temperature, wind, precipitation, etc.) of a weather forecast would include information that accurately quantifies the inherent uncertainty. Surveys have consistently indicated that users desire information about uncertainty or confidence of weather forecasts. The widespread dissemination and effective communication of forecast uncertainty information is likely to yield substantial economic and social benefits, because users can make decisions that explicitly account for this uncertainty."

Advantages and Disadvantages of Probability Theory

For data scientists, there are a number of advantages and disadvantages with probability that need to be considered.

i. Classical

The classical method of probability is used when all probable outcomes have an equal likelihood of happening and every outcome is known in advance. The coin toss example above uses the classical approach to probability. The classical approach offers a simple approach to real-world examples that is easy to digest for those not possessing a math or science background.

With respect to limitations, the classical approach is unable to handle projects where an infinite number of possible outcomes exist. It's also ineffective in scenarios where each outcome is not equally likely, as in the case of tossing a weighted die. These limitations affect the ability of this approach to handling more complicated tasks.

ii. Relative Frequency

Unlike the classical approach, relative frequency offers the advantage of being able to handle scenarios where outcomes have different theoretical probability (or likelihood) of

occurring. This approach can also manage a probability situation where possible outcomes are unknown.

Although you can use relative frequency probability in more diverse situations and settings than classical probability, it has several limitations. The first limitation to relative frequency involves the problem of "infinite repetitions." This is where experiments possessing an infinite number of times cannot be analyzed with this theory. So while a large number of trials can be conducted, that number can't be infinite.

iii. Subjective

Problems that benefit from subjective probability are those that require some level of belief to make possible. For example, a candidate who may be down in the polls may use subjective probability to make a case for staying in the race.

Subjective probability also benefits from what is known as the reference class problem. In a reference class problem, assigning a probability to a certain event might require that event to be classified. That classification can be subjective, and thus changing the classification can change the probability of the event.

For example, if you want to determine the probability of a person contracting an infectious disease like COVID-19, we need to begin with assessing which classes of people are relevant to the problem. It's here where various reference classes can be established. A broad class such as "all U.S. residents" could be used. Or it could be narrowed down to, say, "all residents of the states of X, Y and Z, where 80% of the deaths are occurring." In other words, depending on the reference class chosen, different probabilities will emerge.

What is the probability formula?

The formula for probability states that the possibility of an event happening, or P(E), equals the ratio of the number of favourable outcomes to the number of total outcomes. Mathematically, it looks like this:

P(E) = favourable outcomes/total outcomes

How Data Scientists Use Probability Theory

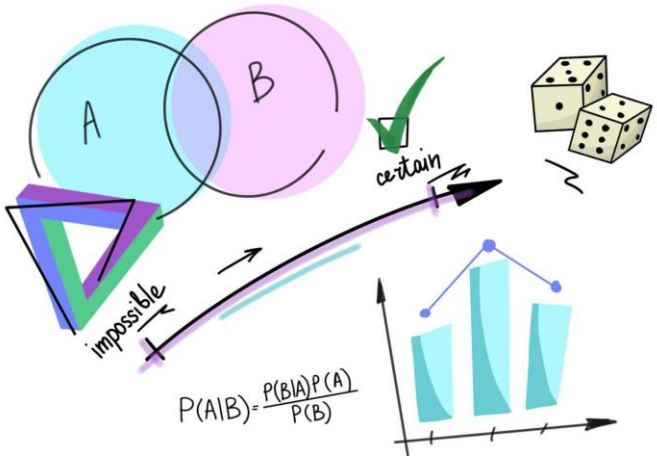


Image 29 - Data Science. Probability
References: https://luminousmen.com/media/data-science-probability_2.jpg

Probability allows data scientists to assess the certainty of outcomes of a particular study or experiment. An experiment is a planned study that is executed under controlled conditions. When a result is not already predetermined, the experiment is referred to as a chance experiment. Conducting a coin toss twice is an example of a chance experiment.

Today's data scientists need to have an understanding of the foundational concepts of probability theory including key concepts involving probability distribution, statistical significance, hypothesis testing and regression. Learn more statistics concepts that data scientists use regularly; probability distribution is only one of them.

Bayes' Theorem, Maximum Likelihood

Bayes' Theorem

What is the Bayes' Theorem?

In statistics and probability theory, the Bayes' theorem (also known as the Bayes' rule) is a mathematical formula used to determine the conditional probability of events. Essentially, the Bayes' theorem describes the probability of an event based on prior knowledge of the conditions that might be relevant to the event.

The theorem is named after English statistician, Thomas Bayes, who discovered the formula in 1763. It is considered the foundation of the special statistical inference approach called the Bayes' inference.

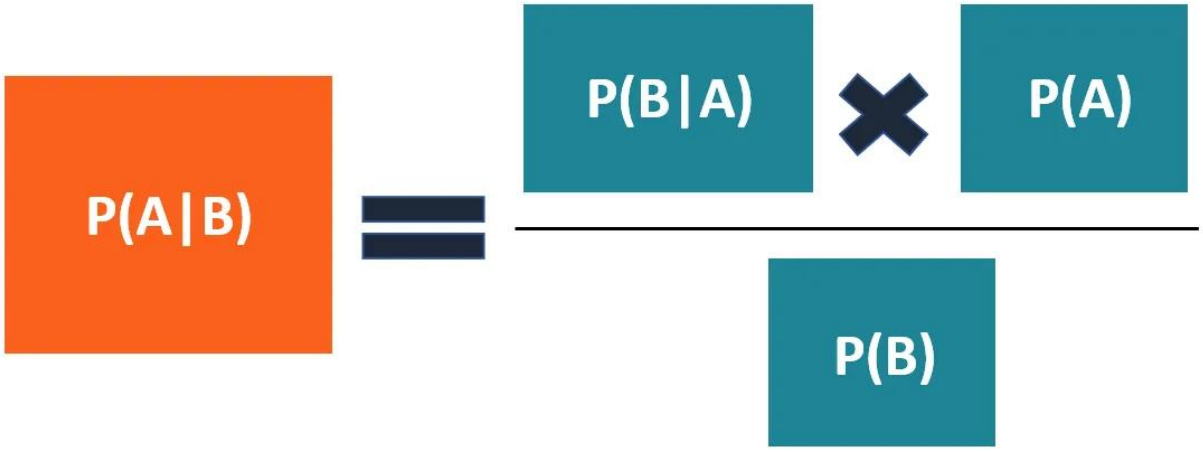


Image 30 - Bayes' Theorem
Reference: https://cdn.corporatefinanceinstitute.com/assets/bayes-theorem.png

Besides statistics, the Bayes' theorem is also used in various disciplines, with medicine and pharmacology as the most notable examples. In addition, the theorem is commonly employed in different fields of finance. Some of the applications include but are not limited to, modeling the risk of lending money to borrowers or forecasting the probability of the success of an investment.

Formula for Bayes' Theorem

The Bayes' theorem is expressed in the following formula:

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

Image 31 - Bayes' Theorem – Formula
Reference: https://cdn.corporatefinanceinstitute.com/assets/bayes-theorem1.png

Where:

- P(A|B) – the probability of event A occurring, given event B has occurred
- P(B|A) – the probability of event B occurring, given event A has occurred
- P(A) – the probability of event A
- P(B) – the probability of event B

Note that events A and B are independent events (i.e., the probability of the outcome of event A does not depend on the probability of the outcome of event B).

A special case of the Bayes' theorem is when event A is a binary variable. In such a case, the theorem is expressed in the following way:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B|A^-)P(A^-) + P(B|A^+)P(A^+)}$$

Image 32 - Special Case
Reference: https://cdn.corporatefinanceinstitute.com/assets/bayes-theorem2-600x97.png

here:

- P(B|A–) – the probability of event B occurring given that event A– has occurred
- P(B|A+) – the probability of event B occurring given that event A+ has occurred

In the special case above, events A– and A+ are mutually exclusive outcomes of event A.

Example of Bayes' Theorem

Imagine you are a financial analyst at an investment bank. According to your research of publicly-traded companies, 60% of the companies that increased their share price by more than 5% in the last three years replaced their CEOs during the period.

At the same time, only 35% of the companies that did not increase their share price by more than 5% in the same period replaced their CEOs. Knowing that the probability that the stock prices grow by more than 5% is 4%, find the probability that the shares of a company that fires its CEO will increase by more than 5%.

Before finding the probabilities, you must first define the notation of the probabilities.

- P(A) – the probability that the stock price increases by 5%
- P(B) – the probability that the CEO is replaced
- P(A|B) – the probability of the stock price increases by 5% given that the CEO has been replaced
- P(B|A) – the probability of the CEO replacement given the stock price has increased by 5%.

Using the Bayes' theorem, we can find the required probability:

$$P(A|B) = \frac{0.60 \times 0.04}{0.60 \times 0.04 + 0.35 \times (1 - 0.04)} = 0.067 \text{ or } 6.67\%$$

Image 33 - Sample Calculation
Reference: https://cdn.corporatefinanceinstitute.com/assets/bayes-theorem3.png

Thus, the probability that the shares of a company that replaces its CEO will grow by more than 5% is 6.67%.

Maximum likelihood

What is Maximum likelihood

Maximum likelihood is a widely used technique for estimation with applications in many areas including time series modelling, panel data, discrete data, and even machine learning.
Here we cover the fundamentals of maximum likelihood estimation.
In particular, we discuss:

1. The basic theory of maximum likelihood.
2. The advantages and disadvantages of maximum likelihood estimation.
3. The log-likelihood function.
4. Modelling applications.

In addition, we consider a simple application of maximum likelihood estimation to a linear regression model.

What is Maximum Likelihood Estimation?

Maximum likelihood estimation is a statistical method for estimating the parameters of a model. In maximum likelihood estimation, the parameters are chosen to maximize the likelihood that the assumed model results in the observed data.
This implies that in order to implement maximum likelihood estimation we must:

1. Assume a model, also known as a data generating process, for our data.
2. Be able to derive the likelihood function for our data, given our assumed model

Once the likelihood function is derived, maximum likelihood estimation is nothing more than a simple optimization problem.

What are the Advantages and Disadvantages of Maximum Likelihood Estimation?

At this point, you may be wondering why you should pick maximum likelihood estimation over other methods such as least squares regression or the generalized method of moments. The reality is that we shouldn't always choose maximum likelihood estimation. Like any estimation technique, maximum likelihood estimation has advantages and disadvantages.

Advantages of Maximum Likelihood Estimation

There are many advantages of maximum likelihood estimation:

- If the model is correctly assumed, the maximum likelihood estimator is the most efficient estimator.

- It provides a consistent but flexible approach which makes it suitable for a wide variety of applications, including cases where assumptions of other models are violated.
- It results in unbiased estimates in larger samples.

Efficiency is one measure of the quality of an estimator. An efficient estimator is one that has a small variance or mean squared error.

Disadvantages of Maximum Likelihood Estimation

- It relies on the assumption of a model and the derivation of the likelihood function which is not always easy.
- Like other optimization problems, maximum likelihood estimation can be sensitive to the choice of starting values.
- Depending on the complexity of the likelihood function, the numerical estimation can be computationally expensive.
- Estimates can be biased in small samples.

What is the Likelihood Function?

Maximum likelihood estimation hinges on the derivation of the likelihood function. For this reason, it is important to have a good understanding of what the likelihood function is and where it comes from.
Let's start with the very simple case where we have one series with 10 independent observations: 5, 0, 1, 1, 0, 3, 2, 3, 4, 1.

The Probability Density

The first step in maximum likelihood estimation is to assume a probability distribution for the data. A probability density function measures the probability of observing the data given a set of underlying model parameters.
In this case, we will assume that our data has an underlying Poisson distribution which is a common assumption, particularly for data that is nonnegative count data.
The Poisson probability density function for an individual observation, is given by

$$f(y_i|\theta) = \frac{e^{-\theta}\theta^{y_i}}{y_i!}$$

Because the observations in our sample are independent, the probability density of our observed sample can be found by taking the product of the probability of the individual observations: $f(y_1, y_2, \ldots, y_{10}|\theta) = \prod_{i=1}^{10} \frac{e^{-\theta}\theta^{y_i}}{y_i!} = \frac{e^{-10\theta}\theta^{\sum_{i=1}^{10} y_i}}{\prod_{i=1}^{10} y_i!}$

We can use the probability density to answer the question of how likely it is that our data occurs given specific parameters.

The Likelihood Function

The differences between the likelihood function and the probability density function are nuanced but important.

- A probability density function expresses the probability of observing our data given the underlying distribution parameters. It assumes that the parameters are known.
- The likelihood function expresses the likelihood of parameter values occurring given the observed data. It assumes that the parameters are unknown.

Mathematically the likelihood function looks similar to the probability density:

$$L(\theta|y_1, y_2, \ldots, y_{10}) = f(y_1, y_2, \ldots, y_{10}|\theta)$$

For our Poisson example, we can fairly easily derive the likelihood function

$$L(\theta|y_1, y_2, \ldots, y_{10}) = \frac{e^{-10\theta}\theta^{\sum_{i=1}^{10} y_i}}{\prod_{i=1}^{10} y_i!} = \frac{e^{-10\theta}\theta^{20}}{207,360}$$

The maximum likelihood estimate of the unknown parameter, , is the value that maximizes this likelihood.

The Log-Likelihood Function

In practice, the joint distribution function can be difficult to work with and the of the likelihood function is used instead. In the case of our Poisson dataset the log-likelihood function is:

$$\ln(L(\theta|y)) = -n\theta + \ln\sum_{i=1}^{n} y_i - \ln\theta\sum_{i=1}^{n} y_i! = -10\theta + 20\ln(\theta) - \ln(207,360)$$

The log-likelihood is usually easier to optimize than the likelihood function.

The Maximum Likelihood Estimator

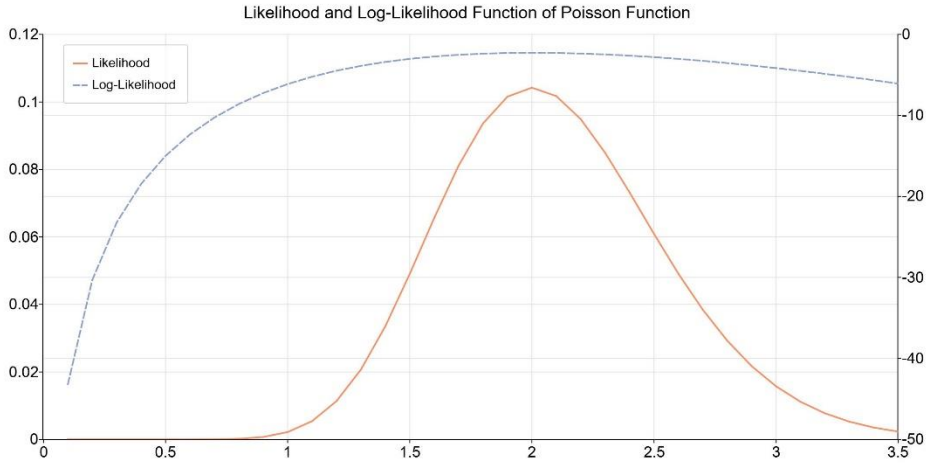


Image 34 - The Maximum Likelihood Estimator
Reference: https://www.aptech.com/wp-content/uploads/2020/09/poisson-likelihood-function.jpeg

A graph of the likelihood and log-likelihood for our dataset shows that the maximum likelihood occurs when θ= 2. This means that our maximum likelihood estimator, $\hat{\theta}_{MLE} = 2$.

Maximum Likelihood Estimation (MLE) vs Bayesian Estimation.

	MLE	Bayesian Estimation
Predictions	We make predictions utilizing the latent variables in the density function to compute a probability.	We make predictions using the posterior distribution and the parameters which are considered as the random variables.
Situations to working with	Data with minimal values and the knowledge of prior is low. We can use MLE.	Data with sparse value and knowledge about the reliability of priors is high. We can use Bayesian estimation.
Complexity	MLE is less complex because we require to compute only the likelihood function	Bayesian estimation is more complex because the computation requires the likelihood function, evidence, and prior.

Hypothesis Testing

What Is Hypothesis Testing?

Hypothesis testing is an act in statistics whereby an analyst tests an assumption regarding a population parameter. The methodology employed by the analyst depends on the nature of the data used and the reason for the analysis.

Hypothesis testing is used to assess the plausibility of a hypothesis by using sample data. Such data may come from a larger population, or from a data-generating process. The word "population" will be used for both of these cases in the following descriptions.

- Hypothesis testing is used to assess the plausibility of a hypothesis by using sample data.
- The test provides evidence concerning the plausibility of the hypothesis, given the data.
- Statistical analysts test a hypothesis by measuring and examining a random sample of the population being analysed.

How Hypothesis Testing Works

In hypothesis testing, an analyst tests a statistical sample, with the goal of providing evidence on the plausibility of the null hypothesis.

Statistical analysts test a hypothesis by measuring and examining a random sample of the population being analyzed. All analysts use a random population sample to test two different hypotheses: the null hypothesis and the alternative hypothesis.

The null hypothesis is usually a hypothesis of equality between population parameters; e.g., a null hypothesis may state that the population mean return is equal to zero. The alternative hypothesis is effectively the opposite of a null hypothesis (e.g., the population mean return is not equal to zero). Thus, they are mutually exclusive, and only one can be true. However, one of the two hypotheses will always be true.

Types of Hypothesis Testing

There are two types of Hypothesis Testing

i. Null hypothesis
ii. Alternative hypothesis

i. Null Hypothesis: It is denoted by H0. A null hypothesis is the one in which sample observations result purely from chance. This means that the observations are not influenced by some non-random cause.

ii. **Alternative Hypothesis:** It is denoted by Ha or H1. An alternative hypothesis is the one in which sample observations are influenced by some non-random cause. A hypothesis test concludes whether to reject the null hypothesis and accept the alternative hypothesis or to fail to reject the null hypothesis. The decision is based on the value of X and R.

Steps in Hypothesis Testing

Econometricians follow a formal process to test a hypothesis and determine whether it is to be rejected. The steps include:

1. **Stating the Hypotheses**
   The first step involves positioning the null and alternative hypotheses. Remember, that these are mutually exclusive. If one hypothesis states a fact, the other must reject it.
2. **Making Statistical Assumptions**
   Consider statistical assumptions – such as independence of observations from each other, normality of observations, random errors and probability distribution of random errors, randomization during sampling, etc.
3. **Formulating an Analysis Plan**
   This includes deciding the test which is to be carried out to test the hypothesis. At the same time, we need to decide how sample data will be used to test the null hypothesis.
4. **Investigating Sample Data**
   At this stage, sample data is examined. It's when we find scores – mean values, normal distribution, t distribution, z score, etc.
5. **Interpreting Results**
   This stage involves making decision to either reject the null hypothesis in favor of alternative hypothesis or not to reject the null hypothesis.

Accepting or Rejecting Null Hypothesis

This is an extension of the last step - interpreting results in the process of hypothesis testing. A null hypothesis is accepted or rejected basis P value and the region of acceptance.

P value – it is a function of the observed sample results. A threshold value is chosen before the test is conducted and is called the significance level, which is represented as α. If the calculated value of $P \leq \alpha$, it suggests the inconsistency between the observed data and the assumption that the null hypothesis is true. This suggests that the null hypothesis must be rejected. However, this doesn't mean that alternative hypothesis can be accepted as true. This is when Type I error occurs.

Real-World Example of Hypothesis Testing

If, for example, a person wants to test that a penny has exactly a 50% chance of landing on heads, the null hypothesis would be that 50% is correct, and the alternative hypothesis would be that 50% is not correct.

Mathematically, the null hypothesis would be represented as Ho: $P = 0.5$. The alternative hypothesis would be denoted as "Ha" and be identical to the null hypothesis, except with the equal sign struck-through, meaning that it does not equal 50%.

A random sample of 100 coin flips is taken, and the null hypothesis is then tested. If it is found that the 100 coin flips were distributed as 40 heads and 60 tails, the analyst would assume that a

penny does not have a 50% chance of landing on heads and would reject the null hypothesis and accept the alternative hypothesis.
If, on the other hand, there were 48 heads and 52 tails, then it is plausible that the coin could be fair and still produce such a result. In cases such as this where the null hypothesis is "accepted," the analyst states that the difference between the expected results (50 heads and 50 tails) and the observed results (48 heads and 52 tails) is "explainable by chance alone."

Central limit theorem

What is the Central Limit Theorem (CLT)?

The Central Limit Theorem (CLT) is a statistical concept that states that the sample mean distribution of a random variable will assume a near-normal or normal distribution if the sample size is large enough. In simple terms, the theorem states that the sampling distribution of the mean approaches a normal distribution as the size of the sample increases, regardless of the shape of the original population distribution.

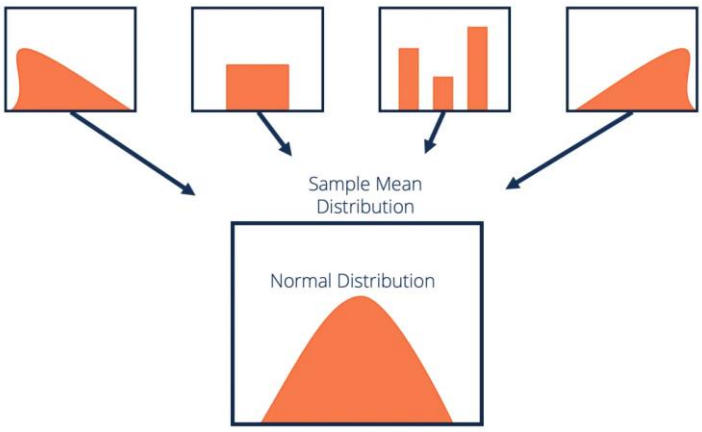


Image 35 - Central Limit Theorem (CLT) Diagram showing Convergence to Normal Distribution
Reference: https://cdn.corporatefinanceinstitute.com/assets/Central-Limit-Theorem-CLT-Diagram-1200x734.png

As the user increases the number of samples to 30, 40, 50, etc., the graph of the sample means will move towards a normal distribution. The sample size must be 30 or higher for the central limit theorem to hold.
One of the most important components of the theorem is that the mean of the sample will be the mean of the entire population. If you calculate the mean of multiple samples of the population, add them up, and find their average, the result will be the estimate of the population mean.
The same applies when using standard deviation. If you calculate the standard deviation of all the samples in the population, add them up, and find the average, the result will be the standard deviation of the entire population.

How Does the Central Limit Theorem Work?

The central limit theorem forms the basis of the probability distribution. It makes it easy to understand how population estimates behave when subjected to repeated sampling. When plotted on a graph, the theorem shows the shape of the distribution formed by means of repeated population samples.

As the sample sizes get bigger, the distribution of the means from the repeated samples tends to normalize and resemble a normal distribution. The result remains the same regardless of what the original shape of the distribution was. It can be illustrated in the figure below:

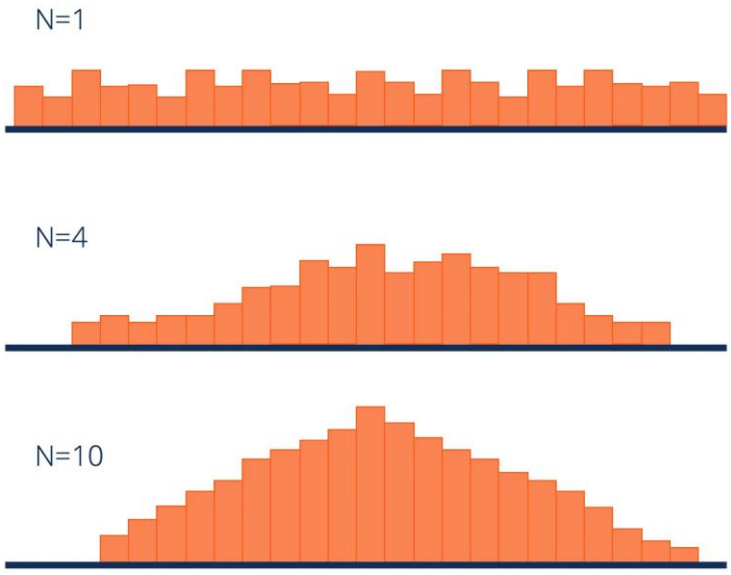


Image 36 - Central Limit Theorem (CLT) - How it arises
Reference: https://cdn.corporatefinanceinstitute.com/assets/Central-Limit-Theorem-CLT-How-it-works-and-how-it-arises.png

From the figure above, we can deduce that despite the fact that the original shape of the distribution was uniform, it tends towards a normal distribution as the value of n (sample size) increases.

Apart from showing the shape that the sample means will take, the central limit theorem also gives an overview of the mean and variance of the distribution. The sample mean of the distribution is the actual population mean from which the samples were taken.

The variance of the sample distribution, on the other hand, is the variance of the population divided by n. Therefore, the larger the sample size of the distribution, the smaller the variance of the sample mean.

Example of Central Limit Theorem

An investor is interested in estimating the return of ABC stock market index that is comprised of 100,000 stocks. Due to the large size of the index, the investor is unable to analyze each stock

independently and instead chooses to use random sampling to get an estimate of the overall return of the index.
The investor picks random samples of the stocks, with each sample comprising at least 30 stocks. The samples must be random, and any previously selected samples must be replaced in subsequent samples to avoid bias.
If the first sample produces an average return of 7.5%, the next sample may produce an average return of 7.8%. With the nature of randomized sampling, each sample will produce a different result. As you increase the size of the sample size with each sample you pick, the sample means will start forming their own distributions.
The distribution of the sample means will move toward normal as the value of n increases. The average return of the stocks in the sample index estimates the return of the whole index of 100,000 stocks, and the average return is normally distributed.

History of the Central Limit Theorem
The initial version of the central limit theorem was coined by Abraham De Moivre, a French-born mathematician. In an article published in 1733, De Moivre used the normal distribution to find the number of heads resulting from multiple tosses of a coin. The concept was unpopular at the time, and it was forgotten quickly.
However, in 1812, the concept was reintroduced by Pierre-Simon Laplace, another famous French mathematician. Laplace re-introduced the normal distribution concept in his work titled "Théorie Analytique des Probabilités," where he attempted to approximate binomial distribution with the normal distribution.
The mathematician found that the average of independent random variables, when increased in number, tends to follow a normal distribution. At that time, Laplace's findings on the central limit theorem attracted attention from other theorists and academicians.

Later in 1901, the central limit theorem was expanded by Aleksandr Lyapunov, a Russian mathematician. Lyapunov went a step ahead to define the concept in general terms and prove how the concept worked mathematically. The characteristic functions that he used to provide the theorem were adopted in modern probability theory.

Chi-square test

What Is a Chi-Square Statistic?
A chi-square ($\chi 2$) statistic is a test that measures how a model compares to actual observed data. The data used in calculating a chi-square statistic must be random, raw, mutually exclusive, drawn from independent variables, and drawn from a large enough sample. For example, the results of tossing a fair coin meet these criteria.

Chi-square tests are often used in hypothesis testing. The chi-square statistic compares the size of any discrepancies between the expected results and the actual results, given the size of the sample and the number of variables in the relationship.
For these tests, degrees of freedom are utilized to determine if a certain null hypothesis can be rejected based on the total number of variables and samples within the experiment. As with any statistic, the larger the sample size, the more reliable the result.

- A chi-square ($\chi 2$) statistic is a measure of the difference between the observed and expected frequencies of the outcomes of a set of events or variables.
- Chi-square is useful for analyzing such differences in categorical variables, especially those nominal in nature.
- $\chi 2$ depends on the size of the difference between actual and observed values, the degrees of freedom, and the samples size.
- $\chi 2$ can be used to test whether two variables are related or independent from one another.
- It can also be used to test the goodness-of-fit between an observed distribution and a theoretical distribution of frequencies.

The Formula for Chi-Square Is

$$\chi_c^2 = \sum \frac{(O_i - E_i)^2}{E_i}$$

where:

$c =$ Degrees of freedom

$O =$ Observed value(s)

$E =$ Expected value(s)

What Does a Chi-Square Statistic Tell You?

There are two main kinds of chi-square tests: the test of independence, which asks a question of relationship, such as, "Is there a relationship between student sex and course choice?"; and the goodness-of-fit test, which asks something like "How well does the coin in my hand match a theoretically fair coin?"
Chi-square analysis is applied to categorical variables and is especially useful when those variables are nominal (where order doesn't matter, like marital status or gender).

Independence

When considering student sex and course choice, a $\chi 2$ test for independence could be used. To do this test, the researcher would collect data on the two chosen variables (sex and courses picked) and then compare the frequencies at which male and female students select among the offered classes using the formula given above and a $\chi 2$ statistical table.

If there is no relationship between sex and course selection (that is, if they are independent), then the actual frequencies at which male and female students select each offered course should be expected to be approximately equal, or conversely, the proportion of male and female students in any selected course should be approximately equal to the proportion of male and female students in the sample.
A $\chi 2$ test for independence can tell us how likely it is that random chance can explain any observed difference between the actual frequencies in the data and these theoretical expectations.

Goodness-of-Fit
$\chi 2$ provides a way to test how well a sample of data matches the (known or assumed) characteristics of the larger population that the sample is intended to represent. This is known as goodness of fit. If the sample data do not fit the expected properties of the population that we are interested in, then we would not want to use this sample to draw conclusions about the larger population.

Example
For example, consider an imaginary coin with exactly a 50/50 chance of landing heads or tails and a real coin that you toss 100 times. If this coin is fair, then it will also have an equal probability of landing on either side, and the expected result of tossing the coin 100 times is that heads will come up 50 times and tails will come up 50 times.\
In this case, $\chi 2$ can tell us how well the actual results of 100 coin flips compare to the theoretical model that a fair coin will give 50/50 results. The actual toss could come up 50/50, or 60/40, or even 90/10. The farther away the actual results of the 100 tosses is from 50/50, the less good the fit of this set of tosses is to the theoretical expectation of 50/50, and the more likely we might conclude that this coin is not actually a fair coin.

When to Use a Chi-Square Test
A chi-square test is used to help determine if observed results are in line with expected results, and to rule out that observations are due to chance. A chi-square test is appropriate for this when the data being analyzed is from a random sample, and when the variable in question is a categorical variable. A categorical variable is one that consists of selections such as type of car, race, educational attainment, male vs. female, how much somebody likes a political candidate (from very much to very little), etc.
These types of data are often collected via survey responses or questionnaires. Therefore, chi-square analysis is often most useful in analyzing this type of data.

What is a chi-square test used for?

Chi-square is a statistical test used to examine the differences between categorical variables from a random sample in order to judge goodness of fit between expected and observed results.

Who uses chi-square analysis?

Since chi-square applies to categorical variables, it is most used by researchers who are studying survey response data. This type of research can range from demography to consumer and marketing research to political science and economics.

Is chi-square analysis used when the independent variable is nominal or ordinal?

A nominal variable is a categorical variable that differs by quality, but whose numerical order could be irrelevant. For instance, asking somebody their favorite color would produce a nominal variable. Asking somebody's age, on the other hand, would produce an ordinal set of data. Chi-square can be best applied to nominal data.

Data mining, wrangling, data manipulation techniques

Data mining

Data mining is the process of sorting through large data sets to identify patterns and relationships that can help solve business problems through data analysis. Data mining techniques and tools enable enterprises to predict future trends and make more-informed business decisions.

Data mining is a key part of data analytics overall and one of the core disciplines in data science, which uses advanced analytics techniques to find useful information in data sets. At a more granular level, data mining is a step in the knowledge discovery in databases (KDD) process, a data science methodology for gathering, processing and analyzing data. Data mining and KDD are sometimes referred to interchangeably, but they're more commonly seen as distinct things.

Why is data mining important?

Data mining is a crucial component of successful analytics initiatives in organizations. The information it generates can be used in business intelligence (BI) and advanced analytics applications that involve analysis of historical data, as well as real-time analytics applications that examine streaming data as it's created or collected.

Effective data mining aids in various aspects of planning business strategies and managing operations. That includes customer-facing functions such as marketing, advertising, sales and customer support, plus manufacturing, supply chain management, finance and HR. Data mining supports fraud detection, risk management, cybersecurity planning and many other critical business use cases. It also plays an important role in healthcare, government, scientific research, mathematics, sports and more.

Data Mining Steps

When asking “what is data mining,” let’s break it down into the steps data scientists and analysts take when tackling a data mining project.

1. **Understand Business**

What is the company's current situation, the project's objectives, and what defines success?

2. **Understand the Data**

Figure out what kind of data is needed to solve the issue, and then collect it from the proper sources.

3. **Prepare the Data**

Resolve data quality problems like duplicate, missing, or corrupted data, then prepare the data in a format suitable to resolve the business problem.

4. **Model the Data**

Employ algorithms to ascertain data patterns. Data scientists create, test, and evaluate the model.

5. **Evaluate the Data**

Decide whether and how effective the results delivered by a particular model will help meet the business goal or remedy the problem. Sometimes there's an iterative phase for finding the best algorithm, especially if the data scientists don't get it quite right the first time. There may be some data mining algorithms shopping around.

6. **Deploy the Solution**

Give the results of the project to the people in charge of making decisions.

Benefits of Data Mining

Data mining provides us with the means of resolving problems and issues in this challenging information age.
Data mining benefits include:

- It helps companies gather reliable information
- It's an efficient, cost-effective solution compared to other data applications
- It helps businesses make profitable production and operational adjustments
- Data mining uses both new and legacy systems
- It helps businesses make informed decisions
- It helps detect credit risks and fraud
- It helps data scientists easily analyze enormous amounts of data quickly
- Data scientists can use the information to detect fraud, build risk models, and improve product safety
- It helps data scientists quickly initiate automated predictions of behaviors and trends and discover hidden patterns.

Data Mining tools

Artificial Intelligence

AI systems perform analytical functions that mimic human intelligence, such as learning, planning, problem-solving, and reasoning.

Association Rule Learning

This toolset, also called market basket analysis, searches for relationships among dataset variables. For example, association rule learning can determine which products are frequently purchased together (e.g., a smartphone and a protective case).

Clustering

This process partitions datasets into a set of meaningful sub-classes, known as clusters. The process helps users understand the natural structure or grouping within the data.

Classification

This technique assigns particular items in a dataset to different target categories or classes. The goal is to develop accurate predictions within the target class for each case in the data.

Data Analytics

The data analytics process enables professionals to evaluate digital information and turn it into useful business intelligence.

Data Cleansing and Preparation

This technique transforms the data into a form optimal for further analysis and processing. Preparation includes activities such as identifying and removing errors and missing or duplicate data.

Data Warehousing

Data warehousing consists of an extensive collection of business data that businesses use to help them make decisions. Warehousing is a fundamental and necessary component of most large-scale data mining efforts.

Machine Learning

Related to the AI technique mentioned earlier, machine learning is a computer programming technique that employs statistical probabilities to provide computers with the ability to learn without human intervention or being manually programmed.

Regression

The regression technique predicts a range of numeric values in categories such as sales, stock prices, or even temperature. The ranges are based on the information found in a particular data set.

Two specific tools need mentioning.

R. This language is an open-source tool used for graphics and statistical computing. It provides analysts with a wide selection of statistical tests, classification and graphical techniques, and time-series analysis.

Oracle Data Mining (ODM). This tool is a module of the Oracle Advanced Analytics Database. It helps data analysts make predictions and generate detailed insights. Analysts use ODM to predict customer behaviour, develop customer profiles, and identify cross-selling opportunities.

Data Mining Applications

Retail
Online retailers mine customer data and internet clickstream records to help them target marketing campaigns, ads and promotional offers to individual shoppers. Data mining and predictive modelling also power the recommendation engines that suggest possible purchases to website visitors, as well as inventory and supply chain management activities.

Financial services
Banks and credit card companies use data mining tools to build financial risk models, detect fraudulent transactions and vet loan and credit applications. Data mining also plays a key role in marketing and in identifying potential upselling opportunities with existing customers.

Insurance
Insurers rely on data mining to aid in pricing insurance policies and deciding whether to approve policy applications, including risk modeling and management for prospective customers.
Manufacturing
Data mining applications for manufacturers include efforts to improve uptime and operational efficiency in production plants, supply chain performance and product safety.

Entertainment
Streaming services do data mining to analyze what users are watching or listening to and to make personalized recommendations based on people's viewing and listening habits.

Healthcare
Data mining helps doctors diagnose medical conditions, treat patients and analyze X-rays and other medical imaging results. Medical research also depends heavily on data mining, machine learning and other forms of analytics.

Data Wrangling
Data Wrangling is the process of gathering, collecting, and transforming Raw data into another format for better understanding, decision-making, accessing, and analysis in less time. Data Wrangling is also known as Data Munging.

Example
Books selling Website want to show top-selling books of different domains, according to user preference. For example, a new user search for motivational books, then they want to show those motivational books which sell the most or having a high rating, etc.
But on their website, there are plenty of raw data from different users. Here the concept of Data Munging or Data Wrangling is used. As we know Data is not Wrangled by System. This process

is done by Data Scientists. So, the data Scientist will wrangle data in such a way that they will sort that motivational book that are sold more or have high ratings or user buy this book with these packages of Books, etc. On the basis of that, the new user will make choice. This will explain the importance of Data wrangling.

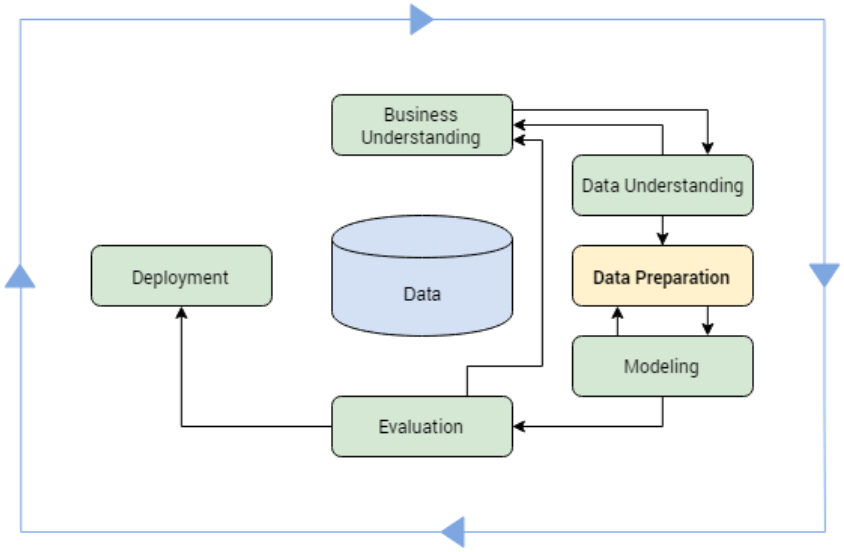


Image 37: Data Wrangling
Reference: https://theappsolutions.com/blog/development/data-wrangling-guide-to-data-preparation/

What is "raw data"?
It is any repository data (texts, images, database records) that is documented but yet to be processed and fully integrated into the system.

The process of wrangling can be described as "digesting" data (often referred to as "munging" thus the alternative term "data munging") and making it useful (aka usable) for the system. It can be described as a preparation stage for every other data-related operation.
Data Wrangling is usually accompanied by Mapping. The term "Data Mapping" refers to the element of the wrangling process that involves identifying source data fields to their respective target data fields. While Wrangling is dedicated to transforming data, Mapping is about connecting the dots between different elements.

What is the Purpose of Data Wrangling?
The primary purpose of data wrangling can be described as getting data in coherent shape. In other words, it is making raw data usable. It provides the substance for further proceedings.
As such, Data Wrangling acts as a preparation stage for the data-mining operation. Process-wise these two operations are coupled together as you can't do one without another.
Overall, data wrangling covers the following processes:

- Getting data from the various source into one place
- Piecing the data together according to the determined setting
- Cleaning the data from the noise or erroneous, missing elements

Data Wrangling Steps

Data Discovery:
This is an all-encompassing term that describes understanding what your data is all about. In this first step, you get familiar with your data

Data Structuring:
When you collect raw data, it initially is in all shapes and sizes, and has no definite structure. Such data needs to be restructured to suit the analytical model that your enterprise plans to deploy

Data Cleaning:
Raw data comes with some errors that need to be fixed before data is passed on to the next stage. Cleaning involves the tackling of outliers, making corrections, or deleting bad data completely

Data Enriching:
By this stage, you have kind of become familiar with the data in hand. Now is the time to ask yourself this question – do you need to embellish the raw data? Do you want to augment it with other data?

Data Validating:
This activity surfaces data quality issues, and they have to be addressed with the necessary transformations. The rules of validation rules require repetitive programming steps to check the authenticity and the quality of your data

Data Publishing:
Once all the above steps are completed, the final output of your data wrangling efforts are pushed downstream for your analytics needs

Data wrangling is a core iterative process that throws up the cleanest, most useful data possible before you start your actual analysis.

Image 38: Data Wrangling Steps
Reference: https://theappsolutions.com/blog/development/data-wrangling-guide-to-data-preparation/

Data Wrangling Tools

- Excel Power Query / Spreadsheets: it is the basic manual structure wrangling tool.

- Open Refine: more sophisticated solutions, requires programming.
- Google Data Prep: for data exploration, cleaning, and feature engineering.
- Tabula: it's suitable for all kinds of data.
- Data Wrangler: used for data cleaning and transformation.
- CSVKit: used for converting data.

Data Wrangling in Python

Numpy (aka Numerical Python)
the most basic package. Lots of features for operations on n-arrays and matrices in Python. The library provides vectorization of mathematical operations on the NumPy array type, which improves performance and accordingly speeds up the execution.
Pandas
designed for fast and easy data analysis operations. Useful for data structures with labeled axes. Explicit data alignment prevents common errors that result from misaligned data coming in from different sources.
Matplotlib
Python visualization module. Good for line graphs, pie charts, histograms, and other professional grade figures.
Plotly
for interactive, publication-quality graphs. Excellent for line plots, scatter plots, area charts, bar charts, error bars, box plots, histograms, heatmaps, subplots, multiple-axis, polar graphs, and bubble charts.

Theano
library for numerical computation similar to Numpy. This library is designed to define, optimize, and evaluate mathematical expressions involving multi-dimensional arrays efficiently.

Data manipulation

What are Data Manipulations?
Data manipulation with python is defined as a process in the python programming language that enables users in data organization in order to make reading or interpreting the insights from the data more structured and comprises of having better design.
For example, arranging the employee's names in alphabetical order will enable quicker searching of a particular employee by their name. The key feature of data manipulation is enabling faster business operations and also emphasize optimization in the process. Through proper manipulated data one can analyze trends, interpret insights from financial data, analyze consumer behavior or pattern, etc. Not only the analyzing, but it also enables users to neglect any unnecessary data in the set so that one can save space and only fill the limited space with important and necessary data.

Pandas module is basically an open-source Python module**. It has a wide scope of use in the field of computing, data analysis, statistics, etc.**
Pandas' module uses the basic functionalities of the **NumPy module**.

There are various ways to install the Python Pandas module.
One of the easiest ways is to install using **Python package installer** i.e., **PIP**.
Type the following command in your Command-prompt:

pip install pandas

In order to add the Pandas and NumPy module to your code, we need to import these modules in our code.

```
import pandas
import numpy
```

Python Pandas Module – Data Structures
Panda's work around the following data structures:

- **Series**
- **Data Frame**
- **Panel**

These data structures are faster as compared to the NumPy arrays.
1. Series
Pandas Series is a 1-dimensional structure resembling arrays containing homogeneous data in it. It is a linear data structure and stores elements in a single dimension.

Note: The **size of the Series Data Structure in Pandas is immutable** i.e. once set, it cannot be changed dynamically. While the **values/elements in the Series can be changed or manipulated**.

Syntax:

pandas.Series(input_data, index, data_type, copy)

input_data: Takes input in vivid forms such as list, constants, NumPy arrays, Dict, etc.
index: Index values passed to the data.
data_type: Recognizes the data type.
copy: Copies Data. The default value is False.

Example:

```
import pandas
import numpy
input = numpy.array(['John','Bran','Sam','Peter'])
series_data = pandas.Series(input,index=[10,11,12,13])
print(series_data)
```

In the above code snippet, we have provided the input using NumPy arrays and set the index values to the input data.

Output:

```
10    John
11    Bran
12    Sam
13    Peter
dtype: object
```

2. DataFrame

Python Pandas module provides DataFrame that is a 2-dimensional structure, resembling the 2-D arrays. Here, the input data is framed in the form of rows and columns.

Note: The **size of the DataFrame Data Structure in Pandas is mutable**.

Syntax:

pandas.DataFrame(input_data, index_value, columns, data_type, copy)

input_data: Takes input in vivid forms such as list, series, NumPy arrays, Dict, another DataFrame, etc.
index values: Index values being passed to the data.
data_type: Recognizes the data type of each column.
copy: Copy Data. The default value is False.
columns: Labels provided the data of the columns.

Example:

```
import pandas
input = [['John','Pune'],['Bran','Mumbai'],['Peter','Delhi']]
data_frame = pandas.DataFrame(input,columns=['Name','City'],index=[1,2,3])
print(data_frame)
```

In the above code, we have provided the input using lists, have added labels: ‘Name’ and ‘City’ to the columns and have set the index values for the same.

Output:

```
   Name    City
```

1 John Pune
2 Bran Mumbai
3 Peter Delhi

3. Panel

Python Pandas module offers a Panel that is a 3-dimensional data structure and contains 3 axes to serve the following functions:

items: (axis 0) Every item of it corresponds to a DataFrame in it.
major_axis: (axis 1) It corresponds to the rows of each DataFrame.
minor_axis: (axis 2) It corresponds to the columns of each DataFrame.

Syntax:

pandas.Panel(input_data, items, major_axis, minor_axis, data_type, copy)

9 Effective Pandas Techniques in Python for Data Manipulation

Pandas is an open-source python library that implements easy, high-performance data structures and data analysis tools. The name comes from the term 'panel data', which relates to multidimensional data sets found in statistics and econometrics.

Installation

To install pandas, just run pip install pandas inside Python environment. Then we can import pandas as pd.

```
1 import pandas as pd
```

One of the most familiar things that pandas is used for is reading in CSV files, utilising pd.read_csv. It is often the starting point for practising pandas.

```
csv_data = pd.read_csv("iris_training.csv")
```

pd.read_csv loads this data into a DataFrame. This can be considered as essentially a table or spreadsheet. Once loaded we can take a quick glimpse of the dataset by calling head() on the data frame.

```
1 csv_data.head()
```

	sepal_len	sepal_width	pedal_len	pedal_width
6.4	2.8	5.6	2.2	2
5.0	2.3	3.3	1.0	1
4.9	2.5	4.5	1.7	2
4.9	3.1	1.5	0.1	0
5.7	3.8	1.7	0.3	0

1.Pivot Table

Pandas can be practised to produce MS Excel style pivot tables. For example, in a table, a key column which has missing values. We can impute it using mean amount of other groups.

```
In [4]: import pandas as pd
        df = pd.read_csv("weather.csv")
        df
```

Out[4]:

	date	city	temperature	humidity
0	5/1/2017	new york	65	56
1	5/2/2017	new york	66	58
2	5/3/2017	new york	68	60
3	5/1/2017	mumbai	75	80
4	5/2/2017	mumbai	78	83
5	5/3/2017	mumbai	82	85
6	5/1/2017	beijing	80	26
7	5/2/2017	beijing	77	30
8	5/3/2017	beijing	79	35

```
In [8]: df.pivot(index="date",columns="city")
```

Out[8]:

	temperature			humidity		
city	beijing	mumbai	new york	beijing	mumbai	new york
date						
5/1/2017	80	75	65	26	80	56
5/2/2017	77	78	66	30	83	58
5/3/2017	79	82	68	35	85	60

2.Boolean Indexing

Boolean Indexing is used if user wants to filter the values of a column based on conditions from another set of columns. For instance, we want a list of all students who are not scholars and got a loan. Boolean indexing can support here.

```
In [ ]: # In Python, 0 is False and 1 is True
        # Also, True is 1 and False is 0

In [8]: 0 == False
Out[8]: True

In [11]: c = 10

In [12]: (c > 1) + (c < 20) + (c == 12)
Out[12]: 2

In [ ]: # a boolean test can be used as an index for an array or tuple

In [13]: state = True

In [50]: state = (True, False)[state]

In [52]: state

In [1]: bp = [120, 117, 131, 118, 133, 122,134,116]
```

3.Apply Function

It is one of the regularly used functions for working with data and building new variables. Apply gains some value after passing each row/column of a data frame with some function. The function can be either default or user-defined.

```
import pandas as pd

def sublst(row):
    return lst[row['J1']:row['J2']]

df = pd.DataFrame({'ID':['1','2','3'], 'J1': [0,2,3], 'J2':[1,4,5]})
print df
lst = ['a','b','c','d','e','f']

df['J3'] = df.apply(sublst,axis=1)
print df
```

Output:

```
  ID  J1  J2
0  1   0   1
1  2   2   4
2  3   3   5
  ID  J1  J2      J3
0  1   0   1     [a]
1  2   2   4  [c, d]
2  3   3   5  [d, e]
```

4.Crosstab

This function is used to get an original view of the data. The function provides scope to validate some fundamental hypothesis. For instance, one column is expected to affect the other column.

```
In [1]: import pandas as pd
In [2]: df = pd.read_csv('train.csv')
In [3]: df.head()
```

Out[3]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

```
In [4]: pd.crosstab(df.Survived, df.Pclass)
```

Out[4]:

Pclass	1	2	3
Survived			
0	80	97	372
1	136	87	119

5.Merge DataFrames

Merging data frames is vital when a user has data coming from various sources to be related.

```
In [1]: import pandas as pd
df1 = pd.DataFrame({
    "city": ["new york","chicago","orlando"],
    "temperature": [21,14,35],
})
df1
```

Out[1]:

	city	temperature
0	new york	21
1	chicago	14
2	orlando	35

```
In [2]: df2 = pd.DataFrame({
    "city": ["chicago","new york","orlando"],
    "humidity": [65,68,75],
})
df2
```

Out[2]:

	city	humidity
0	chicago	65
1	new york	68
2	orlando	75

```
In [ ]: df3=pd.merge(df1,df2,on="city")
```

```
In [3]: df3=pd.merge(df1,df2,on="city")
df3
```

Out[3]:

	city	temperature	humidity
0	new york	21	68
1	chicago	14	65
2	orlando	35	75

6.Sorting DataFrames

When we want to sort Pandas data frame in a particular way. When a user wants to sort pandas data frame based on the values of one or more columns or sort based on the contents of row index or row names of the panda's data frame. Pandas data frame has two useful functions

- sort_values(): this command is used to sort pandas data frame by one or more columns
- sort_index(): this command is used to sort pandas data frame by row index

The above functions come with various options, like sorting the data frame in a specific order, place, sorting with missing values, sorting by a specific algorithm and many more.

```
sort_by_life = gapminder.sort_values('lifeExp')
```

```
print(sort_by_life.head(n=3))
```

```
          country  year        pop continent  lifeExp   gdpPercap
1292       Rwanda  1992  7290203.0    Africa   23.599  737.068595
0     Afghanistan  1952  8425333.0      Asia   28.801  779.445314
552        Gambia  1952   284320.0    Africa   30.000  485.230659
```

7.Plotting

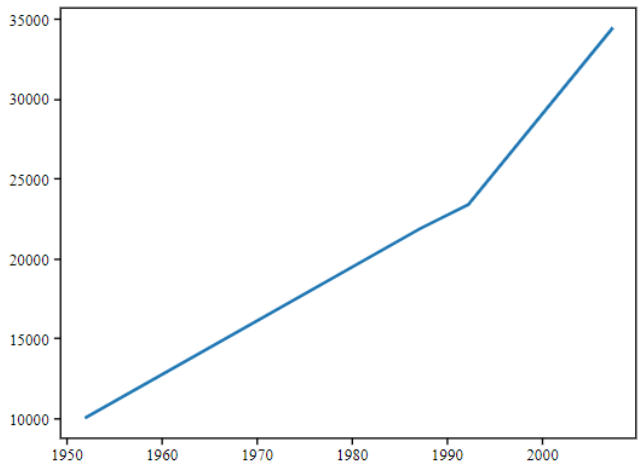
Analysing data from different columns can be very illuminating. Pandas make doing so simple with multi-column DataFrames. By default, calling df.plot() will make pandas to over-plot all column data, with each column as a single line.

```
import pandas

data = pandas.read_csv('data/gapminder_gdp_oceania.csv', index_col='country')

# Extract year from last 4 characters of each column name
years = data.columns.str.strip('gdpPercap_')
# Convert year values to integers, saving results back to dataframe
data.columns = years.astype(int)

data.loc['Australia'].plot()
```



8.Cut function

Seldom numerical values make more sense if grouped together. For illustration, if we're examining to model aeroplanes (#planes flying) with the time of the day (minutes). The specific minute of an hour might not be that appropriate for predicting air traffic as analysed to the actual period of the day like "Morning", "Afternoon", "Evening", "Night", "Late Night". Modelling air traffic this way will be more intuitive and will bypass overfitting.

```
In [1]: import pandas as pd

In [2]: raw_data = {
            "city": ["Tripoli", "Sydney", "Tripoli", "Rome", "Rome", "Tripoli","Rome", "Sydney",
            "rank": ["1st", "2nd", "1st", "2nd", "1st", "2nd","1st", "2nd", "1st"],
            "score1": [44, 48, 39, 41, 38, 44, 34, 54, 61],
            "score2": [67, 63, 55, 70, 64, 77, 45, 66, 72]
           }

        df = pd.DataFrame(raw_data,
                          index = pd.Index(['A','B','C','D','E','F','G','H','I'], name='letter'),
                          columns = pd.Index(['city', 'rank','score1','score2'],name='attributes'))
        #df

In [3]: #Define bins as 0 - 25, 25 - 50, 50 - 75, 75 - 100
        bins = [0, 25, 50, 75, 100]

In [7]: #names for the four groups
        group_names = ['low', 'average', 'good', 'brilliant']

In [12]: df['grade'] = pd.cut(df['score2'], bins, labels=group_names)
```

```
In [3]: #Define bins as 0 to 25, 25 - 50, 50 - 75, 75 - 100
        bins = [0, 25, 50, 75, 100]

In [7]: #names for the four groups
        group_names = ['low', 'average', 'good', 'brilliant']

In [16]: df['grade'] = pd.cut(df['score2'], bins, labels=group_names)

In [17]: df
```

Out[17]:

attributes	city	rank	score1	score2	grade
letter					
A	Tripoli	1st	44	67	good
B	Sydney	2nd	48	63	good
C	Tripoli	1st	39	55	good
D	Rome	2nd	41	70	good
E	Rome	1st	38	64	good
F	Tripoli	2nd	44	77	brilliant

9.Impute Missing Values

Imputing relates to applying a model to restore missing values.

There are several options users can consider while replacing a missing value, for example:

- A fixed value that has meaning within the domain, such as 0, distinct from all other values.
- A value from another randomly chosen from the record.
- A mean, median or mode value replaced for the column.
- A value determined by another predictive model.

Any imputing conducted on the training dataset will have to be performed on new data in the future when predictions are required from the finalized model. This needs to be taken into factor when choosing how to impute the missing values.

For example, if one chooses to impute with mean column values, the mean column values will need to be stored to file for later exercise new data that has missing values.

Pandas provide the fillna() function for returning values with a specific value.

```
import pandas as pd
import numpy as np
import pandas_datareader.data as web
import datetime as dt

''' retrieve data from Yahoo Finance'''
start = dt.datetime(1986,1,1)
end = dt.datetime.today()

df = web.DataReader('^HSI','yahoo',start,end)

df1 = df.fillna(method = 'ffill')


print(df.head())
print(df1.head())
```

Data cleaning and pre-processing techniques

Data Cleaning

Data Cleaning means the process of identifying the incorrect, incomplete, inaccurate, irrelevant or missing part of the data and then modifying, replacing or deleting them according to the necessity. Data cleaning is considered a foundational element of the basic data science.

For example, if you conduct a survey and ask people for their phone numbers, people may enter their numbers in different formats.

When working with multiple data sources, there are many chances for data to be incorrect, duplicated, or mislabeled. If data is wrong, outcomes and algorithms are unreliable, even though they may look correct. Data cleaning is the process of changing or eliminating garbage, incorrect, duplicate, corrupted, or incomplete data in a dataset. There's no such absolute way to describe the precise steps in the data cleaning process because the processes may vary from dataset to dataset. Data cleansing, data cleansing, or data scrub is that the initiative among the general data preparation process. Data cleaning plays an important part in developing reliable answers and within the analytical process and is observed to be a basic feature of the info science basics. The motive of data cleaning services is to construct uniform and standardized data sets that enable data analytical tools and business intelligence easy access and perceive accurate data for each problem.

Why data cleaning is essential?

Data cleaning is the most important task that should be done as a data science professional. Having wrong or bad quality data can be detrimental to processes and analysis. Having clean data will ultimately increase overall productivity and permit the very best quality information in your decision-making. Following are some reasons why data cleaning is essential:

Image 39: Data Cleaning
Reference: https://www.analyticsvidhya.com/blog/2021/06/data-cleaning-using-pandas/

1. Error-Free Data

When multiple sources of data are combined there may be chances of so much error. Through Data Cleaning, errors can be removed from data. Having clean data which is free from wrong and garbage values can help in performing analysis faster as well as efficiently. By doing this task our

considerable amount of time is saved. If we use data containing garbage values, the results won't be accurate. When we don't use accurate data, surely, we will make mistakes. Monitoring errors and good reporting helps to find where errors are coming from, and also makes it easier to fix incorrect or corrupt data for future applications.

2. Data Quality:

The quality of the data is the degree to which it follows the rules of particular requirements. For example, if we have imported phone numbers data of different customers, and in some places, we have added email addresses of customers in the data. But because our needs were straightforward for phone numbers, then the email addresses would be invalid data. Here some pieces of data follow a specific format. Some types of numbers have to be in a specific range. Some data cells might require a selected quite data like numeric, Boolean, etc. In every scenario, there are some mandatory constraints our data should follow. Certain conditions affect multiple fields of data in a particular form. Particular types of data have unique restrictions. If the data isn't in the required format, it would always be invalid. Data cleaning will help us simplify this process and avoid useless data values.

3. Accurate and Efficient:

Ensuring the data is close to the correct values. We know that most of the data in a dataset are valid, and we should focus on establishing its accuracy. Even if the data is authentic and correct, it doesn't mean the data is accurate. Determining accuracy helps to figure out the data entered is accurate or not. For example, the address of a customer is stored in the specified format, maybe it doesn't need to be in the right one. The email has an additional character or value that makes it incorrect or invalid. Another example is the phone number of a customer. This means that we have to rely on data sources, to cross-check the data to figure out if it's accurate or not. Depending on the kind of data we are using, we might be able to find various resources that could help us in this regard for cleaning.

4. Complete Data:

Completeness is the degree to which we should know all the required values. Completeness is a little more challenging to achieve than accuracy or quality. Because it's nearly impossible to have all the info we need. Only known facts can be entered. We can try to complete data by redoing the data gathering activities like approaching the clients again, re-interviewing people, etc. For example, we might need to enter every customer's contact information. But a number of them might not have email addresses. In this case, we have to leave those columns empty. If we have a system that requires us to fill all columns, we can try to enter missing or unknown there. But entering such values does not mean that the data is complete. It would be still being referred to as incomplete.

5. Maintains Data Consistency:

To ensure the data is consistent within the same dataset or across multiple datasets, we can measure consistency by comparing two similar systems. We can also check the data values within the same dataset to see if they are consistent or not. Consistency can be relational. For example, a customer's age might be 25, which is a valid value and also accurate, but it is also stated as a senior citizen in the same system. In such cases, we have to cross-check the data, similar to measuring accuracy, and see which value is true. Is the client a 25-year-old? Or the client is a senior citizen? Only one of these values can be true.

There are multiple ways to for your data consistent.

- By checking in different systems.
- By checking the source.
- By checking the latest data.

8 effective data cleaning techniques

- Remove duplicates
- Remove irrelevant data
- Standardize capitalization
- Convert data type
- Clear formatting
- Fix errors
- Language translation
- Handle missing values

1. Remove Duplicates

When you collect your data from a range of different places, or scrape your data, it's likely that you will have duplicated entries. These duplicates could originate from human error where the person inputting the data or filling out a form made a mistake.

Duplicates will inevitably skew your data and/or confuse your results. They can also just make the data hard to read when you want to visualize it, so it's best to remove them right away.

2. Remove Irrelevant Data

Irrelevant data will slow down and confuse any analysis that you want to do. So, deciphering what is relevant and what is not is necessary before you begin your data cleaning. For instance, if you are analyzing the age range of your customers, you don't need to include their email addresses.

Other elements you'll need to remove as they add nothing to your data include:

- Personal identifiable (PII) data
- URLs
- HTML tags

- Boilerplate text (for ex. in emails)
- Tracking codes
- Excessive blank space between text

3. Standardize Capitalization

Within your data, you need to make sure that the text is consistent. If you have a mixture of capitalization, this could lead to different erroneous categories being created.

It could also cause problems when you need to translate before processing as capitalization can change the meaning. For instance, Bill is a person's name whereas a bill or to bill is something else entirely.

If, in addition to data cleaning, you are text cleaning in order to process your data with a computer model, it's much simpler to put everything in lowercase.

4. Convert Data Types

Numbers are the most common data type that you will need to convert when cleaning your data. Often numbers are imputed as text, however, in order to be processed, they need to appear as numerals.

If they are appearing as text, they are classed as a string and your analysis algorithms cannot perform mathematical equations on them.

The same is true for dates that are stored as text. These should all be changed to numerals. For example, if you have an entry that reads September 24th 2021, you'll need to change that to read 09/24/2021.

5. Clear Formatting

Machine learning models can't process your information if it is heavily formatted. If you are taking data from a range of sources, it's likely that there are a number of different document formats. This can make your data confusing and incorrect.

You should remove any kind of formatting that has been applied to your documents, so you can start from zero. This is normally not a difficult process, both excel and google sheets, for example, have a simple standardization function to do this.

6. Fix Errors

It probably goes without saying that you'll need to carefully remove any errors from your data. Errors as avoidable as typos could lead to you missing out on key findings from your data. Some of these can be avoided with something as simple as a quick spell-check.

Spelling mistakes or extra punctuation in data like an email address could mean you miss out on communicating with your customers. It could also lead to you sending unwanted emails to people who didn't sign up for them.

Other errors can include inconsistencies in formatting. For example, if you have a column of US dollar amounts, you'll have to convert any other currency type into US dollars so as to preserve a consistent standard currency. The same is true of any other form of measurement such as grams, ounces, etc.

7. Language Translation

To have consistent data, you'll want everything in the same language.

The Natural Language Processing (NLP) models behind software used to analyze data are also predominantly monolingual, meaning they are not capable of processing multiple languages. So, you'll need to translate everything into one language.

8. Handle Missing Values

When it comes to missing values, you have two options:

- Remove the observations that have this missing value
- Input the missing data

What you choose to do will depend on your analysis goals and what you want to do next with your data.

Removing the missing value completely might remove useful insights from your data. After all, there was a reason that you wanted to pull this information in the first place.

Therefore, it might be better to input the missing data by researching what should go in that field. If you don't know what it is, you could replace it with the word missing. If it is numerical, you can place a zero in the missing field.

However, if there are so many missing values that there isn't enough data to use, then you should remove the whole section.

6 Steps to Manipulate and Cleanse Data with Python

1. **Implementing missing values imputation** – This is a standard statistical imputing constant, using KNN imputation.
   Outlier/Anomaly Detection is carried out using: Isolation Forest, One Class SVM, Local Outlier Factor, and/or outlier detection algorithms.
2. **Carrying out outlier/anomaly detection**- You can accomplish this by using Isolation Forest, One-Class SVM, and/or Local Outlier Factor outlier detection algorithms.
3. **Utilizing cleaning techniques from the X-Variable family**- In this instance, you want to apply custom functions, remove duplicates, as well as replace crucial values.

4. **Using Cleaning Techniques of the Y-Variable sort** – Here it is important to do label encoding, one-hot encoding, as well as dictionary mapping.
5. **'DataFrames' need to be merged**- This step includes concatenating, merging, and joining.
6. **The last step consists of 'parsing dates'**- Here you need to use auto-format detecting strings to accomplish 'DateTime' converting, including changing 'DateTime' objects to numbers.

Let's go into detail for each step:

ONE: Imputing Missing Values

One of the most common issues you may come across in raw extracted data sets is missing values. As long as there are not too many of them, they can easily be imputed at this stage.

- Simple Imputation Methods, like mean, median, mode can be used to fill in missing values (NaN) with the statistical measure of each column. The parameter can be replaced with 'mean', 'median', 'most_frequent' or mode, or 'constant' which is a manual value.
- KNN Imputing is a more complex method for imputing missing values. The KNN algorithm is used to find different data points like the ones missing values within the datasets.

It is important to note that in order to use KNN imputation, the data needs to be normalized to remove differences in scale. To use KNN imputation you will need to:

- Normalize the data
- KNN impute to fill in missing values
- Inverse scale/normalize the data again

TWO: Outlier and Anomaly Detection

- Isolation Forest is an algorithm used to return the anomaly score of the datasets. The algorithm selects a feature and isolates observations by randomly choosing a split value. Paths are then created by representing the value's normality. The shorter the paths reveal anomalies. 'Trees' of shorter paths for these samples make up a 'forest' likely to reveal the anomalies.
- One Class SVM is another method for finding outliers. This is suited for instances where Isolation Forest cannot be applied due to excessive variance.
- Local Outlier Factor is the third method used to detect anomalies. The Local Outlier Factor measures the deviation of density in each dataset in comparison to the other. Samples that display lower density than their neighbors are likely to be outliers. This algorithm is

distance based, meaning you will need to normalize the data before you can use it. This method is a high variance alternative to Isolation Forest.

It is important when using any of these three methods to be sure that the anomalies are not simply data clusters. You can use PCA visualization to double-check.

THREE: X-Variable Cleaning Methods

- Applying customs functions is necessary when cleaning cannot be done via the built in functions. In this case, you may need to write your own functions but you can try to use an external built-in function first.
- Removing duplicates is an important part of data cleansing. This can be done with data.drop_duplicates(), which removes rows of identical value. You must be careful to check that the duplicate rows are not errors, especially in smaller datasets.
- Sampling data points is important for large datasets. This allows you to sample random data points and can be done with data.sample(number_of_samples).
- Renaming columns is done with .rename, where the key is the original column name and the value is the renamed value.
- Replacing values can be done with data,replace(), which takes two values from the dataframe that you will replace with other values. This is useful for imputing missing values so that imputing algorithms can work effectively.

FOUR: Y-Variable Cleaning Methods

- Label Encoding is necessary for categorical y-variables. If your data has two classes, they need to be converted into 0 and 1, because machine learning algorithms can only operate with mathematical characters. You can do this by using the .map() function, which converts a dictionary or original names and replaces the values with numbers. If there are too many classes to manually map, you can use sklearn's automated method. This method is beneficial because the data can easily be reverted to the original format by using encoder.inverse_transform(array).
- One-Hot Encoding may be preferred in specific cases when you have many classes and you do not want to place quantitative measures on the data. With one-hot encoding, every y-value is a vector the length of the number of each class, with a '1' marking an index within the vector and the rest of the values are marked with '0's. Pandas has a built in function called get_dummies, that can automatically take forms and output the one-hot encoded dataframe.

FIVE: Merging DataFrames

- Concatenation is the top-down method of joining DataFrames

- Merging is the left to right process of merging two DataFrames
- Joining is for other types of merging. Merging only combines rows where there is a common keyword in both data frames. Joining includes left outer joining, where all keywords in the left DataFrame are included, while rows in the right DataFrame are only included if their keywords exist in the left one

SIX: Parsing Dates

- Dates can be very difficult sets of data, but are also some of the most important. This is why it is so important for you to understand how to correctly work with this type of data
- Auto-format detecting string to datetime converting is a crucial skill, as datasets rarely come with datetime objects that can be readily accessed. You can use dateutil to automatically determine the location of days, months and years.
- Converting dates to numbers is necessary for models to be able to understand the concept of time. Datetime objects are converted to numbers. In other words, each date represents the number of days passed since the earliest date in your dataset. The function is applied to the date column using .apply()

Data Pre-processing

Data pre-processing is a step in the data mining and data analysis process that takes raw data and transforms it into a format that can be understood and analyzed by computers and machine learning.

Raw, real-world data in the form of text, images, video, etc., is messy. Not only may it contain errors and inconsistencies, but it is often incomplete, and doesn't have a regular, uniform design.

Machines like to process nice and tidy information – they read data as 1s and 0s. So calculating structured data, like whole numbers and percentages is easy. However, unstructured data, in the form of text and images must first be cleaned and formatted before analysis.

Data Pre-processing Importance

When using data sets to train machine learning models, you'll often hear the phrase **"garbage in, garbage out"** This means that if you use bad or "dirty" data to train your model, you'll end up with a bad, improperly trained model that won't actually be relevant to your analysis.

Good, pre-processed data is even more important than the most powerful algorithms, to the point that machine learning models trained with bad data could actually be harmful to the analysis you're trying to do – giving you "garbage" results.

Image 40: Data Pre-processing
Reference: https://monkeylearn.com/blog/data-preprocessing/

Depending on your data gathering techniques and sources, you may end up with data that's out of range or includes an incorrect feature, like household income below zero or an image from a set of "zoo animals" that is actually a tree. Your set could have missing values or fields. Or text data, for example, will often have misspelled words and irrelevant symbols, URLs, etc.
When you properly preprocess and clean your data, you'll set yourself up for much more accurate downstream processes. We often hear about the importance of "data-driven decision making," but if these decisions are driven by bad data, they're simply bad decisions.

Data Pre-processing Steps

- Data quality assessment
- Data cleaning
- Data transformation
- Data reduction

1. Data quality assessment

Image 41: Data Sources
Reference: https://medium.com/easyread/basics-of-data-preprocessing-71c314bc7188

Take a good look at your data and get an idea of its overall quality, relevance to your project, and consistency. There are a number of data anomalies and inherent problems to look out for in almost any data set, for example:

- **Mismatched data types**: When you collect data from many different sources, it may come to you in different formats. While the ultimate goal of this entire process is to reformat your data for machines, you still need to begin with similarly formatted data. For example, if part of your analysis involves family income from multiple countries, you'll have to convert each income amount into a single currency.
- **Mixed data values**: Perhaps different sources use different descriptors for features – for example, man or male. These value descriptors should all be made uniform.
- **Data outliers**: Outliers can have a huge impact on data analysis results. For example if you're averaging test scores for a class, and one student didn't respond to any of the questions, their 0% could greatly skew the results.
- **Missing data**: Take a look for missing data fields, blank spaces in text, or unanswered survey questions. This could be due to human error or incomplete data. To take care of missing data, you'll have to perform data cleaning.

2. Data cleaning

Data cleaning is the process of adding missing data and correcting, repairing, or removing incorrect or irrelevant data from a data set. Dating cleaning is the most important step of preprocessing because it will ensure that your data is ready to go for your downstream needs.

Image 42: Data Cleaning
Reference: https://medium.com/easyread/basics-of-data-preprocessing-71c314bc7188

Data cleaning will correct all of the inconsistent data you uncovered in your data quality assessment. Depending on the kind of data you're working with, there are a number of possible cleaners you'll need to run your data through.

Missing data

There are a number of ways to correct for missing data, but the two most common are:

- **Ignore the tuples**: A tuple is an ordered list or sequence of numbers or entities. If multiple values are missing within tuples, you may simply discard the tuples with that missing information. This is only recommended for large data sets, when a few ignored tuples won't harm further analysis.
- **Manually fill in missing data**: This can be tedious, but is definitely necessary when working with smaller data sets.

Noisy data

Data cleaning also includes fixing "noisy" data. This is data that includes unnecessary data points, irrelevant data, and data that's more difficult to group together.

Binning: Binning sorts data of a wide data set into smaller groups of more similar data. It's often used when analysing demographics. Income, for example, could be grouped: $35,000-$50,000, $50,000-$75,000, etc.

Regression: Regression is used to decide which variables will actually apply to your analysis. Regression analysis is used to smooth large amounts of data. This will help you get a handle on your data, so you're not overburdened with unnecessary data.

Clustering: Clustering algorithms are used to properly group data, so that it can be analyzed with like data. They're generally used in unsupervised learning, when not a lot is known about the relationships within your data.

If you're working with text data, for example, some things you should consider when cleaning your data are:

- Remove URLs, symbols, emojis, etc., that aren't relevant to your analysis
- Translate all text into the language you'll be working in
- Remove HTML tags
- Remove boilerplate email text
- Remove unnecessary blank text between words
- Remove duplicate data

After data cleaning, you may realize you have insufficient data for the task at hand. At this point you can also perform data wrangling or data enrichment to add new data sets and run them through quality assessment and cleaning again before adding them to your original data.

3. Data transformation

With data cleaning, we've already begun to modify our data, but data transformation will begin the process of turning the data into the proper format(s) you'll need for analysis and other downstream processes.

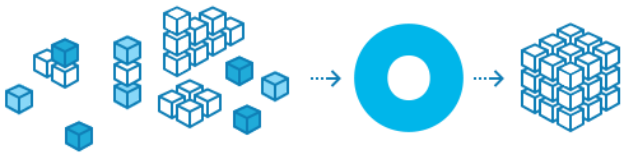


Image 43: Data Transformation
Reference: https://medium.com/easyread/basics-of-data-preprocessing-71c314bc7188

This generally happens in one or more of the below:

1. Aggregation
2. Normalization
3. Feature selection
4. Discreditization
5. Concept hierarchy generation

- **Aggregation:** Data aggregation combines all of your data together in a uniform format.
- **Normalization**: Normalization scales your data into a regularized range so that you can compare it more accurately. For example, if you're comparing employee loss or gain within a number of companies (some with just a dozen employees and some with 200+), you'll have to scale them within a specified range, like -1.0 to 1.0 or 0.0 to 1.0.
- **Feature selection:** Feature selection is the process of deciding which variables (features, characteristics, categories, etc.) are most important to your analysis. These features will be used to train ML models. It's important to remember, that the more features you choose to use, the longer the training process and, sometimes, the less accurate your results, because some feature characteristics may overlap or be less present in the data.

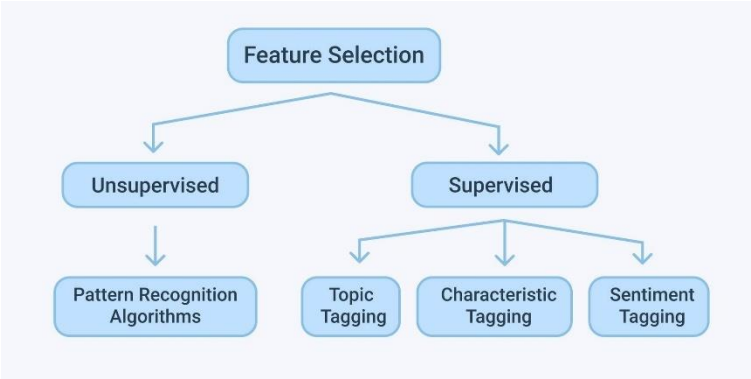


Image 44: Feature Selection
Reference: https://medium.com/easyread/basics-of-data-preprocessing-71c314bc7188

- **Discreditization:** Discreditization pools data into smaller intervals. It's somewhat similar to binning, but usually happens after data has been cleaned. For example, when calculating average daily exercise, rather than using the exact minutes and seconds, you could join together data to fall into 0-15 minutes, 15-30, etc.
- **Concept hierarchy generation:** Concept hierarchy generation can add a hierarchy within and between your features that wasn't present in the original data. If your analysis contains wolves and coyotes, for example, you could add the hierarchy for their genus: canis.

4. Data reduction

The more data you're working with, the harder it will be to analyze, even after cleaning and transforming it. Depending on your task at hand, you may actually have more data than you need. Especially when working with text analysis, much of regular human speech is superfluous or irrelevant to the needs of the researcher. Data reduction not only makes the analysis easier and more accurate, but cuts down on data storage.

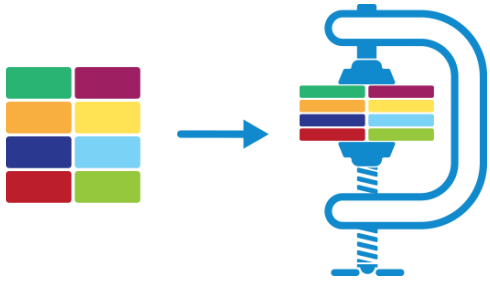


Image 45: Data reduction
Reference: https://medium.com/easyread/basics-of-data-preprocessing-71c314bc7188

It will also help identify the most important features to the process at hand.

- **Attribute selection:** Similar to discreditization, attribute selection can fit your data into smaller pools. It, essentially, combines tags or features, so that tags like male/female and professor could be combined into male professor/female professor.
- **Numerosity reduction:** This will help with data storage and transmission. You can use a regression model, for example, to use only the data and variables that are relevant to your analysis.
- **Dimensionality reduction:** This, again, reduces the amount of data used to help facilitate analysis and downstream processes. Algorithms like K-nearest neighbors use pattern recognition to combine similar data and make it more manageable.

Steps involved in data pre-processing:

- Importing the required Libraries
- Importing the data set
- Handling the Missing Data.
- Encoding Categorical Data.
- Splitting the data set into test set and training set.
- Feature Scaling.

Step 1: Importing the required Libraries

To follow along you will need to download this dataset: **Data.csv**

Every time we make a new model, we will require to import Numpy and Pandas. Numpy is a Library which contains Mathematical functions and is used for scientific computing while Pandas is used to import and manage the data sets.

```
import pandas as pd
import numpy as np
```

Here we are importing the pandas and Numpy library and assigning a shortcut “pd” and “np” respectively.

Step 2: Importing the Dataset

Data sets are available in .csv format. A CSV file stores tabular data in plain text. Each line of the file is a data record. We use the read_csv method of the pandas library to read a local CSV file as a **dataframe**.

```
dataset = pd.read_csv('Data.csv')
```

After carefully inspecting our dataset, we are going to create a matrix of features in our dataset (X) and create a dependent vector (Y) with their respective observations. To read the columns, we will use iloc of pandas (used to fix the indexes for selection) which takes two parameters — [row selection, column selection].

X = dataset.iloc[:,:-1].values
y = dataset.iloc[:, 3].values

Step 3: Handling the Missing Data

```
df
```

	A	B	C	D
0	1.0	2.0	3.0	4.0
1	5.0	6.0	NaN	8.0
2	0.0	11.0	12.0	NaN

```
from sklearn.preprocessing import Imputer
imputer = Imputer(missing_values='NaN', strategy='mean', axis=0)
imputer = imputer.fit(df)
imputed_data = imputer.transform(df.values)
imputed_data
```

```
array([[  1. ,   2. ,   3. ,   4. ],
       [  5. ,   6. ,   7.5,   8. ],
       [  0. ,  11. ,  12. ,   6. ]])
```

The data we get is rarely homogenous. Sometimes data can be missing and it needs to be handled so that it does not reduce the performance of our machine learning model.
To do this we need to replace the missing data by the Mean or Median of the entire column. For this we will be using the sklearn. preprocessing Library which contains a class called Imputer which will help us in taking care of our missing data.
From sklearn.preprocessing import Imputer
imputer = Imputer(missing_values = "NaN", strategy = "mean", axis = 0)

Our object name is **imputer.** The Imputer class can take parameters like:

1. **missing_values** : It is the placeholder for the missing values. All occurrences of missing_values will be imputed. We can give it an integer or “NaN” for it to find missing values.
2. **strategy** : It is the imputation strategy — If “mean”, then replace missing values using the mean along the axis (Column). Other strategies include “median” and “most_frequent”.
3. **axis** : It can be assigned 0 or 1, 0 to impute along columns and 1 to impute along rows.

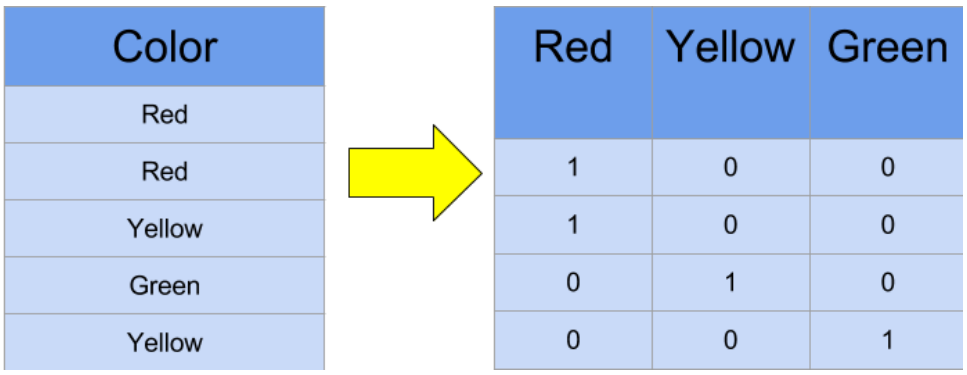
Now we fit the imputer object to our data.

imputer = imputer.fit(X[:, 1:3])

Now replacing the missing values with the mean of the column by using transform method.

X[:, 1:3] = imputer.transform(X[:, 1:3])

Step 4: Encoding categorical data



Color
Red
Red
Yellow
Green
Yellow

Red	Yellow	Green
1	0	0
1	0	0
0	1	0
0	0	1

Converting Categorical data into dummy variables

Any variable that is not quantitative is categorical. Examples include Hair color, gender, field of study, college attended, political affiliation, status of disease infection.

But why encoding?

We cannot use values like “Male” and “Female” in mathematical equations of the model so we need to encode these variables into numbers.

To do this we import “LabelEncoder” class from “sklearn.preprocessing” library and create an object labelencoder_X of the LabelEncoder class. After that we use the fit_transform method on the categorical features.

After Encoding it is necessary to distinguish between between the variables in the same column, for this we will use OneHotEncoder class from sklearn.preprocessing library.

One-Hot Encoding

One hot encoding transforms categorical features to a format that works better with classification and regression algorithms.

```
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
labelencoder_X = LabelEncoder()
X[:, 0] = labelencoder_X.fit_transform(X[:, 0])onehotencoder =
OneHotEncoder(categorical_features = [0])
X = onehotencoder.fit_transform(X).toarray()labelencoder_y = LabelEncoder()
y = labelencoder_y.fit_transform(y)
```

Step 5: Splitting the Data set into Training set and Test Set

Now we divide our data into two sets, one for training our model called the **training set** and the other for testing the performance of our model called the **test set**. The split is generally 80/20. To do this we import the "train_test_split" method of "sklearn.model_selection" library.

```
from sklearn.model_selection import train_test_split
```

Now to build our training and test sets, we will create 4 sets —

1. **X_train** (training part of the matrix of features),
2. **X_test** (test part of the matrix of features),
3. **Y_train** (training part of the dependent variables associated with the X train sets, and therefore also the same indices) ,
4. **Y_test** (test part of the dependent variables associated with the X test sets, and therefore also the same indices).

We will assign to them the test_train_split, which takes the parameters — arrays (X and Y), test_size (Specifies the ratio in which to split the data set).

```
X_train, X_test, Y_train, Y_test = train_test_split( X , Y , test_size = 0.2, random_state = 0)
```

Step 6: Feature Scaling

Most of the machine learning algorithms use the **Euclidean distance** between two data points in their computations. Because of this, **high magnitudes features will weigh more** in the distance calculations **than features with low magnitudes**. To avoid this Feature standardization or Z-score normalization is used. This is done by using "StandardScaler" class of "sklearn.preprocessing".

```
from sklearn.preprocessing import StandardScaler
sc_X = StandardScaler()
```

Further we will transform our X_test set while we will need to fit as well as transform our X_train set.The transform function will transform all the data to a same standardized scale.

```
X_train = sc_X.fit_transform(X_train)
X_test = sc_X.transform(X_test)
```

Data analytics project lifecycle

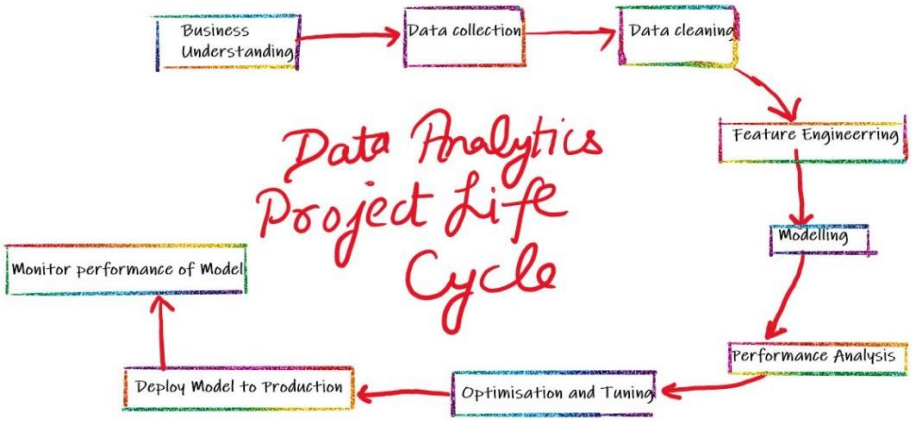


Image 46: Data Analytics Project Life Cycle
Reference: https://medium.com/easyread/basics-of-data-preprocessing-71c314bc7188

A Data Analytics Project has the below steps to be followed from start to end, these steps are performed in sequence but sometime after analyzing the situation there can be need to getting back to the previous step followed instead of moving ahead.

Domain/Business Understanding:
This initial phase focuses on understanding the project objectives and requirements from a business perspective, and then converting this knowledge into a data mining.
Understand the Business Process, beliefs, data generation process and storage.
It's very important to spend a good amount of time in this step so that the future once become more relevant to perform and there are less chances of errors.

Data collection/Data Exploration:
The data understanding phase starts with an initial data collection and proceeds with activities in order to get familiar with the data, to identify data quality problems, to discover first insights into the data, or to detect interesting subsets to form hypotheses for hidden information.

Data Cleaning:
It basically includes filling the missing values, dropping irrelevant and duplicate data. Handling outliers and unnatural values.

Feature Engineering:
Feature engineering is about creating new input features from your existing ones.
In general, you can think of data cleaning as a process of subtraction and feature engineering as a process of addition.
Getting Qualitative and quantitative information from the data, Feature selection using Data Visualization.

Modelling:
In this phase, various modelling techniques are selected and applied and their parameters are calibrated to optimal values. Typically, there are several techniques for the same data mining problem type. Some techniques have specific requirements on the form of data. Therefore, it is often required to step back to the data preparation phase.

Evaluation:
At this stage in the project, you have built a model (or models) that appears to have high quality, from a data analysis perspective. Before proceeding to final deployment of the model, it is important to evaluate the model thoroughly and review the steps executed to construct the model, to be certain it properly achieves the business objectives.

Optimization and Tuning:
Even after evaluation of Model there can be certain requirements which can be raised at time of evaluation, Analyzing the same and integrating in the model.

Deploy the Model to production:
Finally, after all the UAT and implementing changes during UAT, the model is finally put to production. i.e the same goes live.

Monitor the performance during Production:
It takes some time to settle the model in live environment and familiarize the audience with the new features, so a continuous monitor is required so that any discrepancies can be solved straight away.

Numerical Computing using NumPy Library

What is Numerical Computing?

Numerical computing is an approach for solving complex mathematical problems using only simple arithmetic operations

There is a frequent need for processing large amounts of data in computational science applications. Storing data in lists and traversing lists with plain Python for loops leads to slow code, especially when compared with similar code in compiled languages such as Fortran, C, or C++. Fortunately, there is an extension of Python, commonly called Numerical Python, or abbreviated NumPy, which offers efficient array computations. **Numerical Python** has a fixed-size, homogeneous (fixed-type), multi-dimensional array type and lots of functions for various array operations. The result is a dynamically typed environment for array computing similar to basic Matlab. Usually, the speed of NumPy operations is quite close to what is obtained in pure Fortran, C, or C++.

What is NumPy in Python?

NumPy is an open-source library available in Python, which helps in mathematical, scientific, engineering, and data science programming. It is a very useful library to perform mathematical and statistical operations in Python. It works perfectly for multi-dimensional arrays and matrix multiplication. It is easy to integrate with C/C++ and Fortran.

For any scientific project, NumPy is the tool to know. It has been built to work with the N-dimensional array, linear algebra, random number, Fourier transform, etc.

NumPy is a programming language that deals with multi-dimensional arrays and matrices. On top of the arrays and matrices, NumPy supports a large number of mathematical operations.

Why use NumPy?

NumPy is memory efficiency, meaning it can handle the vast amount of data more accessible than any other library. Besides, NumPy is very convenient to work with, especially for matrix multiplication and reshaping. On top of that, NumPy is fast. In fact, TensorFlow and Scikit learn to use NumPy array to compute the matrix multiplication in the back end.

pip install numpy

#import it in your applications by adding the import keyword:

import numpy

```
import numpy as np
arr = np.array([1, 2, 3, 4, 5])
print(arr)
```

Checking NumPy Version

The version string is stored under __version__ attribute.

```
import numpy as np
print(np.__version__)
```

Creating Arrays

NumPy is used to work with arrays. The array object in NumPy is called ndarray. We can create a NumPy ndarray object by using the array() function.

```
import numpy as np
arr = np.array([1, 2, 3, 4, 5])
print(arr)
print(type(arr))
```

Dimensions in Arrays

A dimension in arrays is one level of array depth (nested arrays).

0-D Arrays

0-D arrays, or Scalars, are the elements in an array. Each value in an array is a 0-D array.

```
arr = np.array(42)
```

1-D Arrays

An array that has 0-D arrays as its elements is called uni-dimensional or 1-D array.

```
import numpy as np
arr = np.array([1, 2, 3, 4, 5])
print(arr)
```

2-D Arrays

An array that has 1-D arrays as its elements is called a 2-D array.

These are often used to represent matrix or 2nd order tensors.

```
arr = np.array([[1, 2, 3], [4, 5, 6]])
```

3-D arrays

An array that has 2-D arrays (matrices) as its elements is called 3-D array.

These are often used to represent a 3rd order tensor.

```
arr = np.array([[[1, 2, 3], [4, 5, 6]], [[1, 2, 3], [4, 5, 6]]])
```

Random Numbers using NumPy

To generate random numbers for Gaussian distribution, use:

```
numpy.random.normal(loc, scale, size)
```

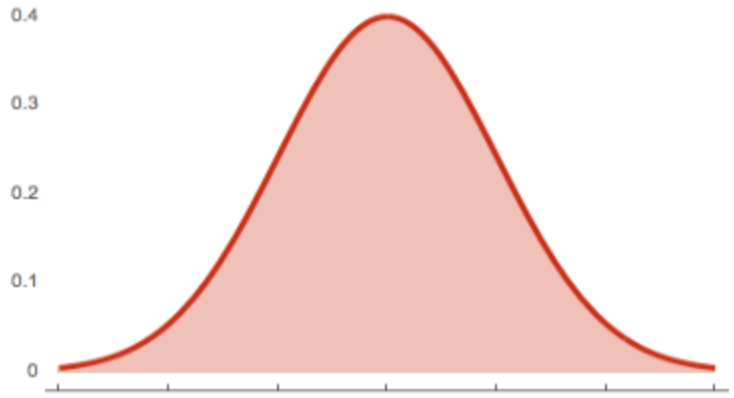
Here,

- **Loc**: the mean. The center of distribution
- **Scale**: standard deviation.
- **Size**: number of returns

Example:

```
## Generate random nmber from normal distribution
normal_array = np.random.normal(5, 0.5, 10)
print(normal_array)
[5.56171852 4.84233558 4.65392767 4.946659   4.85165567 5.61211317 4.46704244
5.22675736 4.49888936 4.68731125]
```

If plotted the distribution will be similar to following plot



Indexing and Slicing in Python

Slicing data is trivial with numpy. We will slice the matrice “e”. Note that, in Python, you need to use the brackets to return the rows or columns.

Example:

```
## Slice
import numpy as np
e  = np.array([(1,2,3), (4,5,6)])
print(e)
[[1 2 3]
 [4 5 6]]
```

Remember with numpy the first array/column starts at 0.

```
## First column
print('First row:', e[0])
```

```
## Second col
print('Second row:', e[1])
```

Output:

```
First row: [1 2 3]
Second row: [4 5 6]
```

In Python, like many other languages,

- The values before the comma stand for the rows
- The value on the rights stands for the columns.
- If you want to select a column, you need to add : before the column index.
- : means you want all the rows from the selected column.

```
print('Second column:', e[:,1])
Second column: [2 5]
```

To return the first two values of the second row. You use : to select all columns up to the second

```
## Second Row, two values
  print(e[1, :2])
  [4 5]
```

Shape of an Array

The shape of an array is the number of elements in each dimension.

```
import numpy as np
arr = np.array([[1, 2, 3, 4], [5, 6, 7, 8]])
print(arr.shape)
```

Output

(2,4)

What is numpy.zeros()?

numpy.zeros() or np.zeros Python function is used to create a matrix full of zeroes. numpy.zeros() in Python can be used when you initialize the weights during the first iteration in TensorFlow and other statistic tasks.

Syntax

```
numpy.zeros(shape, dtype=float, order='C')
```

Python numpy.zeros() Parameters

Here,

- **Shape**: is the shape of the numpy zero array
- **Dtype**: is the datatype in numpy zeros. It is optional. The default value is float64
- **Order**: Default is C which is an essential row style for numpy.zeros() in Python.

Example

```
import numpy as np
np.zeros((2,2))
```

Output:

```
array([[0., 0.],
       [0., 0.]])
```

Example of numpy zero with Datatype

```
import numpy as np
np.zeros((2,2), dtype=np.int16)
```

Output:

```
array([[0, 0],
       [0, 0]], dtype=int16)
```

What is numpy.ones()?

np.ones() function is used to create a matrix full of ones. numpy.ones() in Python can be used when you initialize the weights during the first iteration in TensorFlow and other statistic tasks.

Syntax

```
numpy.ones(shape, dtype=float, order='C')
```

Python numpy.ones() Parameters

Here,

- **Shape**: is the shape of the np.ones Python Array
- **Dtype**: is the datatype in numpy ones. It is optional. The default value is float64
- **Order**: Default is C which is an essential row style.

Python numpy.ones() 2D Array with Datatype Example

```
import numpy as np
np.ones((1,2,3), dtype=np.int16)
```

Output:

```
array([[[1, 1, 1],
```

```
    [1, 1, 1]]], dtype=int16)
```

numpy.reshape() function in Python

Python NumPy Reshape function is used to shape an array without changing its data. In some occasions, you may need to reshape the data from wide to long.

Syntax

```
numpy.reshape(a, newShape, order='C')
```

Here,

- **a**: Array that you want to reshape
- **newShape**: The new desires shape
- **Order**: Default is C which is an essential row style.

Example

```
import numpy as np
e  = np.array([(1,2,3), (4,5,6)])
print(e)
e.reshape(3,2)
```

Output:

```
 // Before reshape
[[1 2 3]
 [4 5 6]]
//After Reshape
array([[1, 2],
       [3, 4],
       [5, 6]])
```

numpy.flatten() in Python

Python NumPy Flatten function is used to return a copy of the array in one-dimension. When you deal with some neural network like convnet, you need to flatten the array.

Syntax

```
numpy.flatten(order='C')
```

Here,

Order: Default is C which is an essential row style.

Example

```
e.flatten()
```

Output:
array([1, 2, 3, 4, 5, 6])

Statistical Functions in Python

NumPy has quite a few useful statistical functions for finding minimum, maximum, percentile standard deviation and variance, etc from the given elements in the array. The functions are explained as follows
Numpy is equipped with the robust statistical function as listed below
Consider the following Array:
Example:

```
import numpy as np
normal_array = np.random.normal(5, 0.5, 10)
print(normal_array)
```

Output:

```
[5.56171852 4.84233558 4.65392767 4.946659 4.85165567 5.61211317 4.46704244
5.22675736 4.49888936 4.68731125]
```

Example of NumPy Statistical function

```
### Min
print(np.min(normal_array))

### Max
print(np.max(normal_array))

### Mean
print(np.mean(normal_array))

### Median
print(np.median(normal_array))

### Sd
```

print(np.std(normal_array))

Output:

```
4.467042435266913
5.612113171990201
4.934841002270593
4.846995625786663
0.3875019367395316
```

What is numpy dot product?

Numpy.dot product is a powerful library for matrix computation. For instance, you can compute the dot product with np.dot. Numpy.dot product is the dot product of a and b. numpy.dot () in Python handles the 2D arrays and perform matrix multiplications.

Syntax:

```
numpy.dot (x, y, out=None)
```

Parameters
Here,

- **x,y**: Input arrays. x and y both should be 1-D or 2-D for the np.dot() function to work
- **out**: This is the output argument for 1-D array scalar to be returned. Otherwise ndarray should be returned.

Returns
The function numpy.dot() in Python returns a Dot product of two arrays x and y. The dot() function returns a scalar if both x and y are 1-D; otherwise, it returns an array. If 'out' is given then it is returned.
Raises
Dot product in Python raises a ValueError exception if the last dimension of x does not have the same size as the second last dimension of y.

Example:

```
## Linear algebra
### Dot product: product of two arrays
f = np.array([1,2])
```

```
g = np.array([4,5])
### 1*4+2*5
np.dot(f, g)
```

Output:

14

Matrix Multiplication in Python

The Numpy matmul() function is used to return the matrix product of 2 arrays. Here is how it works

1. 2-D arrays, it returns normal product
2. Dimensions > 2, the product is treated as a stack of matrix
3. 1-D array is first promoted to a matrix, and then the product is calculated

Syntax:

numpy.matmul(x, y, out=None)

Here,

```
x,y: Input arrays. scalars not allowed
out: This is optional parameter. Usually output is stored in ndarray
```

Example:

In the same way, you can compute matrices multiplication with np.matmul

```
### Matmul: matruc product of two arrays
h = [[1,2],[3,4]]
i = [[5,6],[7,8]]
### 1*5+2*7 = 19
np.matmul(h, i)
```

Output:

```
array([[19, 22],
       [43, 50]])
```

Determinant

Last but not least, if you need to compute the determinant, you can use np.linalg.det(). Note that numpy takes care of the dimension.

Example:

```
## Determinant 2*2 matrix
### 5*8-7*6np.linalg.det(i)
```

Output:

-2.000000000000005

Multidimensional data handling using Pandas Library

What are PANDAS

PANDAS (PANel Data) is a high-level data manipulation tool used for analysis data. It is very easy to import and export data using the Pandas library which has a very rich set of functions. It gives us a single, convenient place to do most of our data analysis and visualization work.
Pandas have three important data structures, namely- Series, DataFrame, and Panel to make the process of analyzing data organized, effective and efficient.

- It is a package useful for data analysis and manipulation.
- Pandas provide an easy way to create, manipulate and wrangle the data.
- Pandas provide powerful and easy-to-use data structures, as well as the means to quickly perform operations on these structures.

Data scientists use Pandas for its following advantages:

- Easily handles missing data.
- It uses Series for one-dimensional data structure and DataFrame for multi-dimensional data structure.
- It provides an efficient way to slice the data.
- It provides a flexible way to merge, concatenate or reshape the data.

Image 47 - What is PANDAS
Reference: https://cdn.educba.com/academy/wp-content/uploads/2019/04/What-is-Pandas-1.jpg

Data Structure in Pandas

A data structure is a collection of data values and operations that can be applied to that data. It enables efficient storage, retrieval and modification to the data.
Pandas deals with 3 data structure

1. Series
2. Data Frame
3. Panel

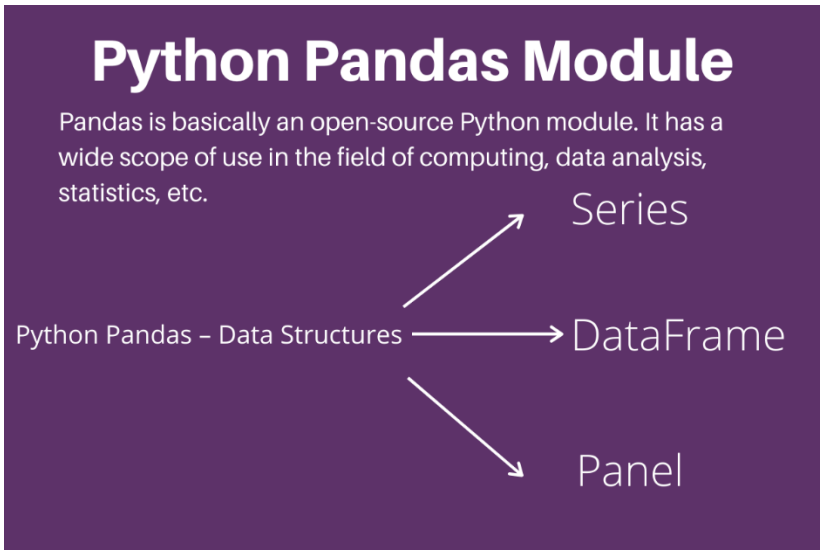


Image 48 - Data Structure in PANDAS
Reference: https://www.askpython.com/wp-content/uploads/2020/02/Python-Pandas-Module.png

Series

Series is a one-dimensional array like structure with homogeneous data, which can be used to handle and manipulate data. What makes it special is its index attribute, which has incredible functionality and is heavily mutable.

It has two parts

1. Data part (An array of actual data)
2. Associated index with data (associated array of indexes or data labels)

- We can say that Series is a labeled one-dimensional array which can hold any type of data.
- Data of Series is always mutable, means it can be changed.
- But the size of Data of Series is always immutable, means it cannot be changed.
- Series may be considered as a Data Structure with two arrays out which one array works as Index (Labels) and the second array works as original Data.
- Row Labels in Series are called Index.

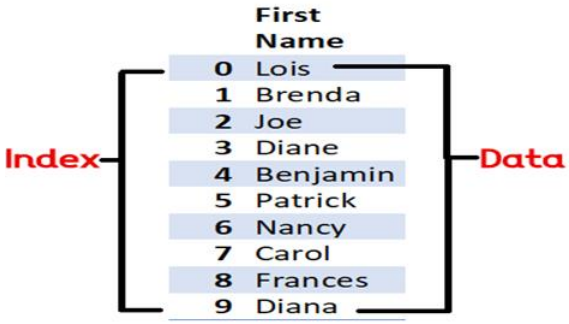


Image 49 - Series
Reference: https://1.bp.blogspot.com/-mYFbQb6dNWo/Xyq3MiVrWMI/AAAAAAAAsB8/ayRGqtckDEIyctuow7M65ezNtMIdqeH7ACPcBGAYYCw/s437/Pandas%2Bseries.png

Example of a series containing names of students is given below:
Index Value
0 Arnab
1 Samridhi
2 Ramit
3 Divyam
4 Kritika

Creation of Series

There are different ways in which a series can be created in Pandas. To create or use series, we first need to import the Pandas library.
Creation of Series from Scalar Values
A Series can be created using scalar values as shown in
the example below:

```
>>> import pandas as pd #import Pandas with alias pd
>>> series1 = pd.Series([10,20,30]) #create a Series
>>> print(series1) #Display the series
```

Output:

```
0 10
1 20
2 30
dtype: int64
```

Creation of Series from NumPy Arrays

We can create a series from a one-dimensional (1D) NumPy array, as shown below:

```
>>> import numpy as np # import NumPy with alias np
```

```
>>> import pandas as pd
>>> array1 = np.array([1,2,3,4])
>>> series3 = pd.Series(array1)
>>> print(series3)
```

Output:

```
0 1
1 2
2 3
3 4
dtype: int32
```

Creation of Series from Dictionary

Recall that Python dictionary has key: value pairs and a value can be quickly retrieved when its key is known.

Dictionary keys can be used to construct an index for a Series, as shown in the following example. Here, keys of the dictionary dict1 become indices in the series.

```
>>> dict1 = {'India': 'NewDelhi', 'UK': 'London', 'Japan': 'Tokyo'}
>>> print(dict1) #Display the dictionary {'India': 'NewDelhi', 'UK': 'London', 'Japan': 'Tokyo'}
>>> series8 = pd.Series(dict1)
>>> print(series8) #Display the series
India NewDelhi
UK London
Japan Tokyo
dtype: object
```

Accessing Elements of a Series

There are two common ways for accessing the elements of a series: Indexing and Slicing.

Indexing

Indexing in Series is similar to that for NumPy arrays, and is used to access elements in a series. Indexes are of two types: positional index and labelled index. Positional index takes an integer value that corresponds to its position in the series starting from 0, whereas labelled index takes any user-defined label as index.Following example shows usage of the positional index for accessing a value from a Series.

```
>>> seriesNum = pd.Series([10,20,30])
>>> seriesNum[2]
30
```

Here, the value 30 is displayed for the positional index 2.

When labels are specified, we can use labels as indices while selecting values from a Series, as shown below. Here, the value 3 is displayed for the labelled index Mar.

```
>>> seriesMnths = pd.Series([2,3,4],index=["Feb ","Mar","Apr"])
>>> seriesMnths["Mar"]
3
```

Slicing

Sometimes, we may need to extract a part of a series. This can be done through slicing. This is similar to slicing used with NumPy arrays. We can define which part of the series is to be sliced by specifying the start and end parameters [start :end] with the series name. When we use positional indices for slicing, the value at the endindex position is excluded, i.e., only (end - start) number of data values of the series are extracted.

Consider the following series seriesCapCntry:

```
>>> seriesCapCntry = pd.Series(['NewDelhi', 'WashingtonDC', 'London', 'Paris'], index=['India', 'USA', 'UK', 'France'])
 >>> seriesCapCntry[1:3] #excludes the value at index position 3
USA WashingtonDC
UK London
dtype: objectPage Break
```

Data Visualization using Matplotlib

Data Visualization is an important part of business activities as organizations nowadays collect a huge amount of data. Sensors all over the world are collecting climate data, user data through clicks, car data for prediction of steering wheels etc. All of these data collected hold key insights for businesses and visualizations make these insights easy to interpret.

Image 50- Data visualization using Matplotlib
Reference: https://i.ytimg.com/vi/VyhLRJVoIrI/maxresdefault.jpg

Why are visualizations important?

Visualizations are the easiest way to analyze and absorb information. Visuals help to easily understand the complex problem. They help in identifying patterns, relationships, and outliers in data. It helps in understanding business problems better and quickly. It helps to build a compelling story based on visuals. Insights gathered from the visuals help in building strategies for businesses. It is also a precursor to many high-level data analysis for Exploratory Data Analysis(EDA) and Machine Learning(ML).
Data visualizations in python can be done via many packages. We'll be discussing of matplotlib package. It can be used in Python scripts, Jupyter notebook, and web application servers.

Matplotlib

Matplotlib is a 2-D plotting library that helps in visualizing figures. Matplotlib emulates Matlab like graphs and visualizations. Matlab is not free, is difficult to scale and as a programming language is tedious. So, matplotlib in Python is used as it is a robust, free and easy library for data visualization.
Anatomy of Matplotlib Figure

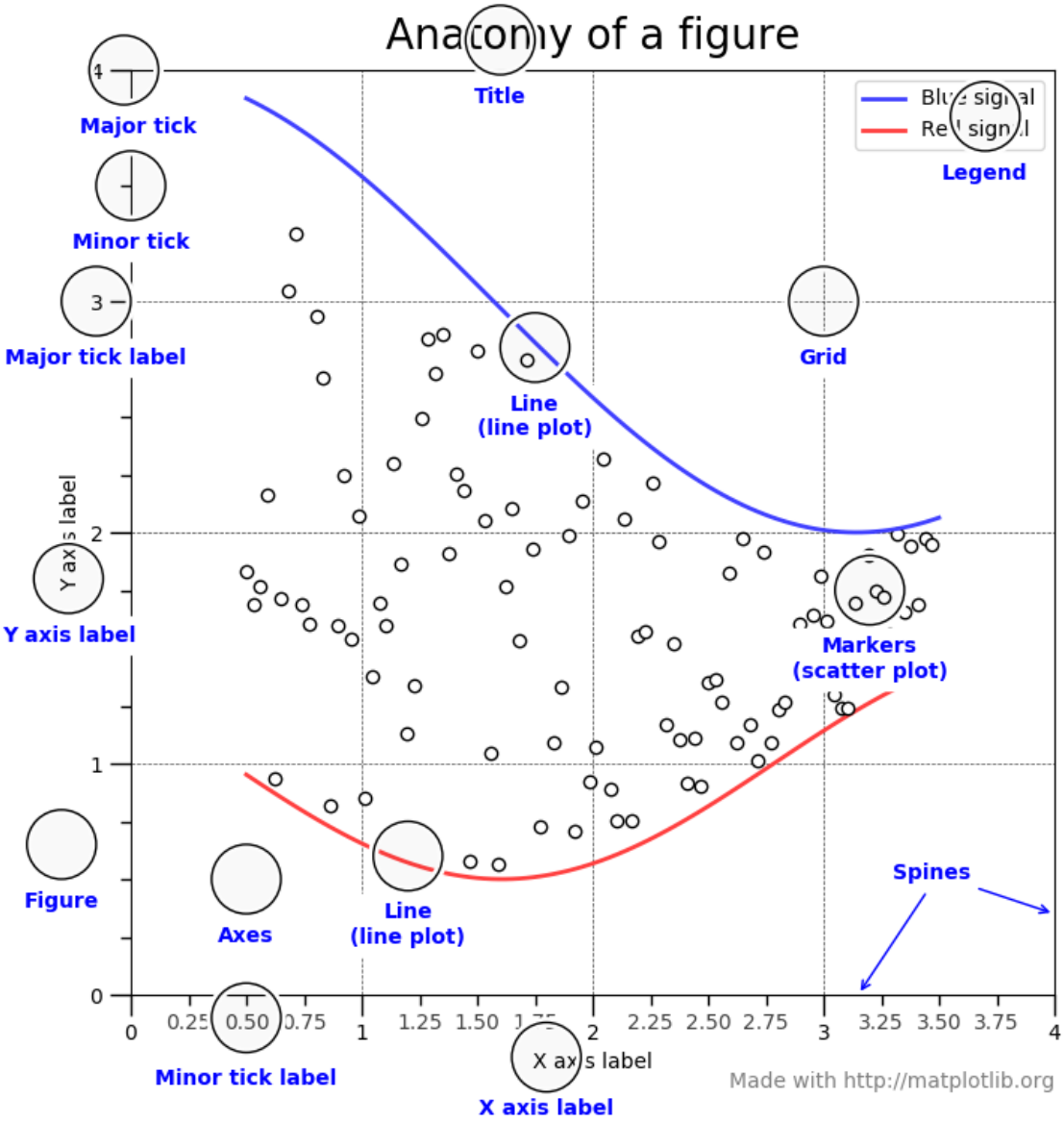


Image 51- Anatomy of Matplotlib figure
Reference: https://matplotlib.org/3.1.1/_images/sphx_glr_anatomy_001.png

The figure contains the overall window where plotting happens, contained within the figure are where actual graphs are plotted. Every Axes has an x-axis and y-axis for plotting. And contained within the axes are titles, ticks, labels associated with each axis. An essential figure of matplotlib is that we can more than axes in a figure which helps in building multiple plots, as shown below. In matplotlib, pyplot is used to create figures and change the characteristics of figures.

Installing Matplotlib

Type !pip install matplotlib in the Jupyter Notebook or if it doesn't work in cmd type conda install -c conda-forge matplotlib . This should work in most cases.

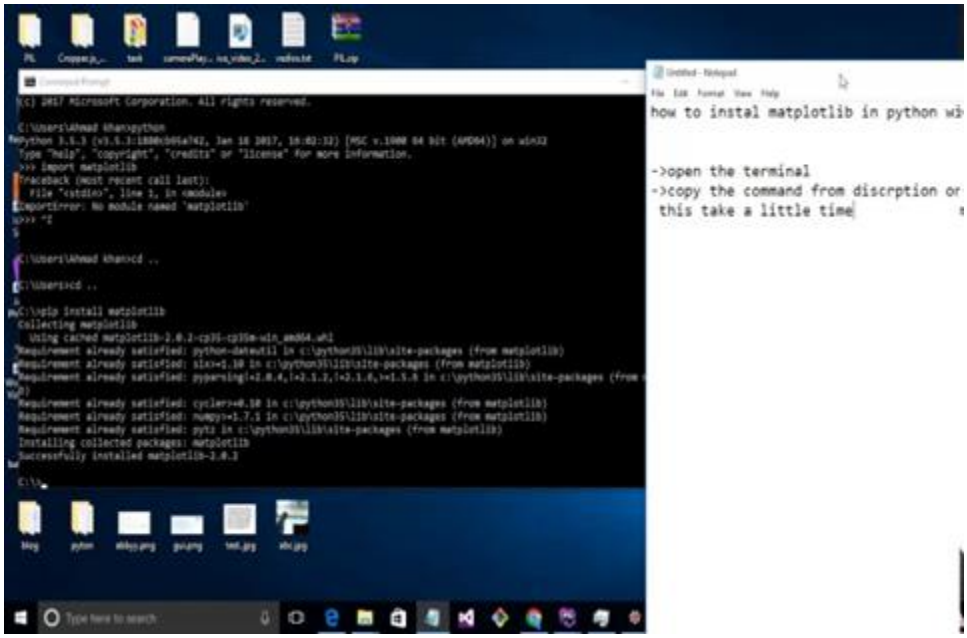


Image 52 -Installing Matplotlib
Reference: https://i.ytimg.com/vi/Iq9f2bQJOPg/maxresdefault.jpg

Things to follow

Plotting of Matplotlib is quite easy. Generally, while plotting they follow the same steps in each and every plot. Matplotlib has a module called pyplot which aids in plotting figure. The Jupyter notebook is used for running the plots. We import matplotlib.pyplot as plt for making it call the package module.

- Importing required libraries and dataset to plot using Pandas pd.read_csv()
- Extracting important parts for plots using conditions on Pandas Dataframes.
- plt.plot()for plotting line chart similarly in place of plot other functions are used for plotting. All plotting functions require data and it is provided in the function through parameters.
- plot.xlabel , plt.ylabel for labeling x and y-axis respectively.
- plt.xticks , plt.yticks for labeling x and y-axis observation tick points respectively.
- plt.legend() for signifying the observation variables.
- plt.title() for setting the title of the plot.
- plot.show() for displaying the plot.

Histogram

A histogram takes in a series of data and divides the data into a number of bins. It then plots the frequency data points in each bin (i.e. the interval of points). It is useful in understanding the count of data ranges.
When to use: We should use histogram when we need the count of the variable in a plot.
eg: Number of particular games sold in a store.

Histogram

```
In [3]:
plt.hist(np_data['GrandCanyon'],
         facecolor='peru',
         edgecolor='blue',
         bins=10)
plt.show()
```

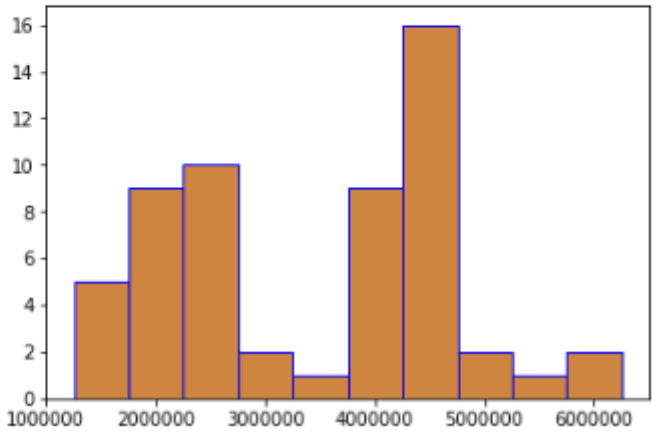


Image 53-Histogram
Reference: https://miro.medium.com/max/820/1*_1qvF8jD_1amVM0SMJ3cjA.png

From above we can see the histogram for GrandCanyon visitors in years. plt.hist() takes the first argument as numeric data in the horizontal axis i.e GrandCanyon visitor.bins=10 is used to create 10 bins between values of visitors in GrandCanyon.

Components of a histogram

- **n:** Contains the frequency of each bin
- **bins:** Represents the middle value of each bin
- **patches:** The Patch object for the rectangle shape representing each bar

```
In [4]:
n, bins, patches = plt.hist(np_data['GrandCanyon'],
                            facecolor='peru',
                            edgecolor='blue',
                            bins=10)

print('n: ', n)
print('bins: ', bins)
print('patches: ', patches)
```

```
n:  [ 5.  9. 10.  2.  1.  9. 16.  2.  1.  2.]
bins:  [1253000.  1753123.8 2253247.6 2753371.4 3253495.2 3753619.  4253742.8
 4753866.6 5253990.4 5754114.2 6254238. ]
patches:  <a list of 10 Patch objects>
```

Image 54 - Histogram
Reference: https://miro.medium.com/max/1224/1*5a17l7yG4VLKjGLVouyycw.png

From above, we can see the components that make a histogram, n as the max values in each bin of histogram i.e 5,9, and so on.

The cumulative property

If True, then a histogram is computed where each bin gives the counts in that bin plus all bins for smaller values. The last bin gives the total number of datapoints.

```
In [5]:
plt.hist(np_data['GrandCanyon'],
         facecolor='peru',
         edgecolor='blue',
         bins=10,
         cumulative=True)
plt.show()
```

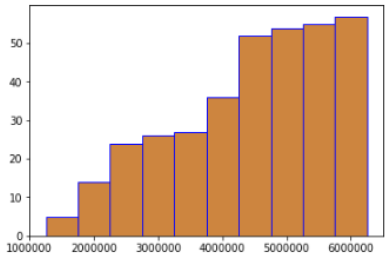


Image 55 - Histogram
Reference: https://miro.medium.com/max/1400/1*TVUmVjh_BVjQe2wRwwnOzw.png

The cumulative property gives us the end added value and helps us understand the increase in value at each bin.

Check the histogram to a range of values

We only look at the data points within the range 2M-5M. This realigns the bins in the histogram

In [6]:
```
plt.hist(np_data['GrandCanyon'],
         facecolor='peru',
         edgecolor='blue',
         bins=10,
         range=(2000000, 5000000))

plt.show()
```

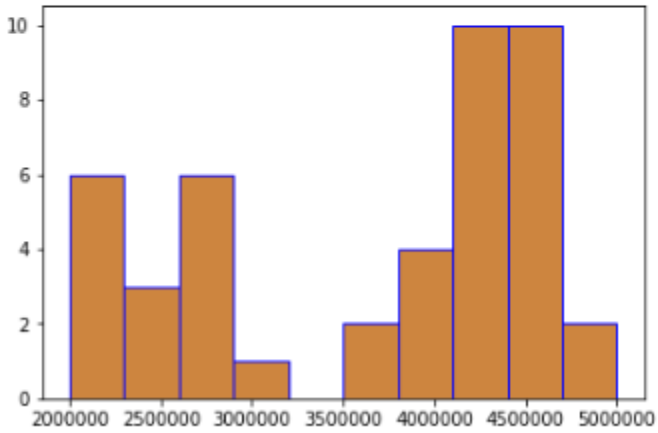


*Image 56 - **Histogram***
Reference: https://miro.medium.com/max/1234/1*RryYV9wgjBdscZR_8OhRdg.png

Range helps us in understanding value distribution between specified values.

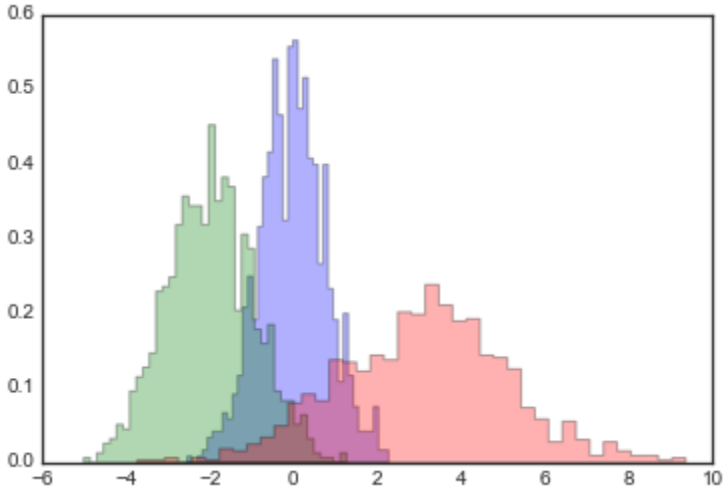


Image 57-Histogram
Reference: https://i.stack.imgur.com/Ji8I6.png

Multiple histograms are useful in understanding the distribution between 2 entity variables. We can see that GrandCanyon has comparably more visitors than BryceCanyon.

Implementation: Histogram

Pie Chart

It is a circular plot which is divided into slices to illustrate numerical proportion. The slice of a pie chart is to show the proportion of parts out of a whole.

When to use: Pie chart should be used seldom used as it is difficult to compare sections of the chart. Bar plot is used instead as comparing sections is easy.

e.g.: Market share in Films.

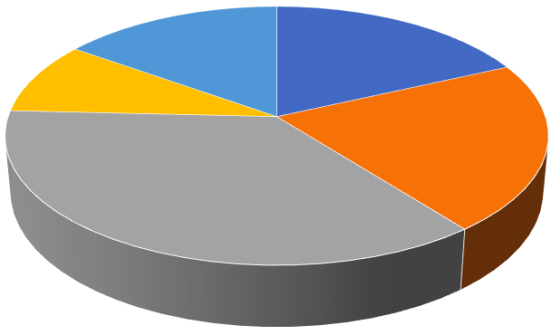


Image 58 - Histogram

Reference: https://miro.medium.com/max/629/1*rzNBRBt1RCr4NsN4Lb7V1Q.png

Above, plt.pie() takes the numeric data as 1st argument i.e Percentage and labels to display as second argument i.e Sector. Ultimately, it shows the distribution of data in proportion to the pie.

Pie chart components

- **wedges:** A list of Patch objects representing each wedge
- **texts:** List of Text objects representing the labels
- **autotexts:** List of Text objects for the numeric values - this is only availabe if the autopct value for the pie chart is not None

In [4]:

```
wedges, texts, autotexts = plt.pie(t_mov['Percentage'],
                                   labels=t_mov['Sector'],
                                   autopct='%.2f')

plt.axis('equal')

print('Wedges: ', wedges)
print('Texts: ', texts)
print('Autotexts: ', autotexts)
```

```
Wedges:  [<matplotlib.patches.Wedge object at 0x000001F40DD48BE0>, <matplotlib.patches.Wedge object at 0x00
tplotlib.patches.Wedge object at 0x000001F40DD51B00>, <matplotlib.patches.Wedge object at 0x000001F40DD5B2E
hes.Wedge object at 0x000001F40DD5BA90>]
Texts:  [Text(0.889919,0.646564,'Sci-Fi'), Text(-0.777817,0.777817,'Drama'), Text(-0.49939,-0.980107,'Actio
0.980107,'Comedy'), Text(0.980107,-0.49939,'Romance')]
Autotexts:  [Text(0.48541,0.352671,'20.00'), Text(-0.424264,0.424264,'35.00'), Text(-0.272394,-0.534604,'25
4,-0.534604,'5.00'), Text(0.534604,-0.272394,'15.00')]
```

Image 59 - Histogram

Reference: https://miro.medium.com/max/1400/1*yrThnLG3Ub5By5IwzaC1yQ.png

From above we can the components that make a pie chart and it returns wedge object, text in labels and so on.

Pie chart customizations

We set the values for the colors and autopct properties. The latter sets the format for the values to be displayed.

In [5]:
```
colors = ['darkorange', 'sandybrown', 'darksalmon', 'orangered','chocolate']

plt.pie(t_mov['Percentage'],
        labels=t_mov['Sector'],
        colors=colors,
        autopct='%.2f')

plt.axis('equal')

plt.show()
```

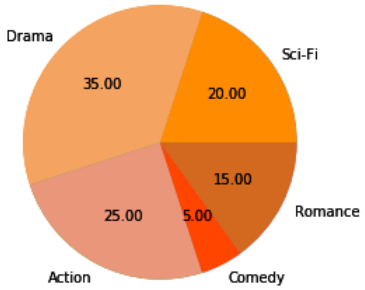


Image 60 - Histogram
Reference: https://miro.medium.com/max/1400/1*vR0jlrqSeVkBf8gmHi79eA.png

A pie chart can be easily customized and from above color and label values are formatted.

The explode property

To highlight a particular wedge of the pie chart, we use explode to separate it from the rest of the chart.

The value for "explode" represents the fraction of the radius with which to offset each wedge.

In [6]:
```
explode = (0, 0.1, 0, 0, 0.3)

plt.pie(t_mov['Percentage'],
        labels=t_mov['Sector'],
        colors=colors,
        autopct='%.2f',
        explode=explode)

plt.axis('equal')

plt.show()
```

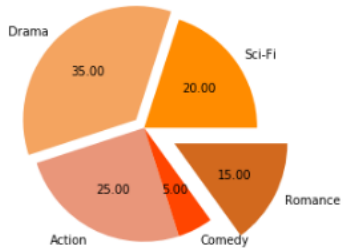


Image 61 - Histogram
Reference: https://miro.medium.com/max/1298/1*5qWJfIY-cBCBgZlTzerXsA.png

From above explode is used to separate out points from the pie. Similar to a pizza piece being cut.

Time Series by line plot

Time series is a line plot and it is basically connecting data points with a straight line. It is useful in understanding the trend over time. It can explain the correlation between points by the trend. An upward trend means positive correlation and downward trend means a negative correlation. It mostly used in forecasting, monitoring models.

When to use: Time Series should be used when single or multiple variables are to be plotted over time.

e.g.: Stock Market Analysis of Companies, Weather Forecasting.

```
In [2]: 1 stock_data.head()
```

Out[2]:

	Date	AAPL	ADBE	CVX	GOOG	IBM	MDLZ	MSFT	NFLX	ORCL	SBUX
0	3-Jan-07	11.107141	38.869999	50.777351	251.001007	79.242500	17.519524	24.118483	3.258571	15.696321	15.752188
1	1-Feb-07	10.962033	39.250000	48.082939	224.949951	74.503204	16.019426	22.092464	3.218571	15.028588	13.930813
2	1-Mar-07	12.037377	41.700001	51.900383	229.309311	75.561348	16.009354	21.857189	3.312857	16.583584	14.138198
3	2-Apr-07	12.930043	41.560001	54.588032	235.925919	81.934280	16.924608	23.480597	3.167143	17.196436	13.984914
4	1-May-07	15.701322	44.060001	57.598267	249.204208	85.786057	17.111704	24.146753	3.128572	17.726965	12.988567

```
In [3]: 1 stock_data['Date'] = pd.to_datetime(stock_data['Date'])
        2 stock_data.head()
```

Out[3]:

	Date	AAPL	ADBE	CVX	GOOG	IBM	MDLZ	MSFT	NFLX	ORCL	SBUX
0	2007-01-03	11.107141	38.869999	50.777351	251.001007	79.242500	17.519524	24.118483	3.258571	15.696321	15.752188
1	2007-02-01	10.962033	39.250000	48.082939	224.949951	74.503204	16.019426	22.092464	3.218571	15.028588	13.930813
2	2007-03-01	12.037377	41.700001	51.900383	229.309311	75.561348	16.009354	21.857189	3.312857	16.583584	14.138198
3	2007-04-02	12.930043	41.560001	54.588032	235.925919	81.934280	16.924608	23.480597	3.167143	17.196436	13.984914
4	2007-05-01	15.701322	44.060001	57.598267	249.204208	85.786057	17.111704	24.146753	3.128572	17.726965	12.988567

Image 62 - Time Series by line plot
Reference: https://miro.medium.com/max/1400/1*3HWpcekG8MUwpQAtmW1Oow.png

First, Convert Date to pandas DateTime for easier plotting of data.

Compare Stock side-by-side

```
In [4]:
1  fig = plt.figure(figsize=(10,6))
2
3  ax1 = fig.add_axes([0, 0, 1, 1])
4  ax2 = fig.add_axes([0.05, 0.65, 0.5, 0.3])
5
6  ax1.set_title('AAPL vs IBM(inset)')
7
8  ax1.plot(stock_data['Date'],
9           stock_data['AAPL'],
10          color='green')
11
12 ax2.plot(stock_data['Date'],
13          stock_data['IBM'],
14          color='blue')
15 plt.show()
16
```

Image 63 - Time Series by line plot
Reference: https://miro.medium.com/max/822/1*X3q-_Cl_eMuncoVG-DoQCQ.png

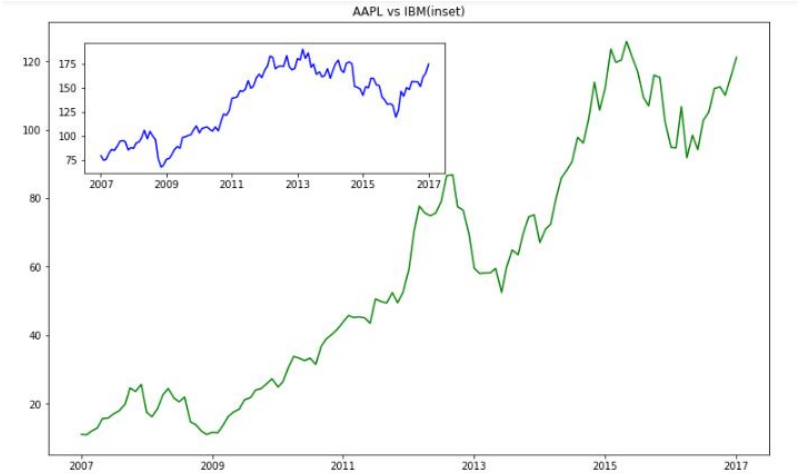


Image 64 - Time Series by line plot
Reference: https://miro.medium.com/max/1400/1*aYgw-zGwdRn_RE3h5KcIHQ.png

From above, fig.add_axes is used for plotting the canvas. Check this What are the differences between add_axes and add_subplot? to understand axes and subplots. plt.plot() takes the 1st argument as numeric data i.e Date and 2nd argument is to numeric stock data. AAPL Stock is considered as ax1 which is the outer figure and on ax2 IBM Stock is considered for plotting which is inset.

In [15]:
```
# Figsize for width and height of plot
fig = plt.figure(figsize=(15,7))

fig.suptitle('Stock price comparison 2007-2017',
             fontsize=20)

ax1 = fig.add_subplot(231)
ax1.set_title('MSFT')

ax1.plot(stock_data['Date'],
         stock_data['MSFT'],
         color='green')

ax2 = fig.add_subplot(232)
ax2.set_title('GOOG')

ax2.plot(stock_data['Date'],
         stock_data['GOOG'],
         color='purple')

ax3 = fig.add_subplot(233)
ax3.set_title('SBUX')

ax3.plot(stock_data['Date'],
         stock_data['SBUX'],
         color='magenta')

ax4 = fig.add_subplot(234)
ax4.set_title('ADBE')

ax4.plot(stock_data['Date'],
         stock_data['ADBE'],
         color='orange')

ax4 = fig.add_subplot(235)
ax4.set_title('NFLX')

ax4.plot(stock_data['Date'],
         stock_data['NFLX'],
         color='chocolate')

ax4 = fig.add_subplot(236)
ax4.set_title('ORCL')

ax4.plot(stock_data['Date'],
         stock_data['ORCL'],
         color='teal')

```

Image 65 - Time Series by line plot

Reference: https://miro.medium.com/max/786/1*klsAZMknOQT6B2yd-My-xA.png

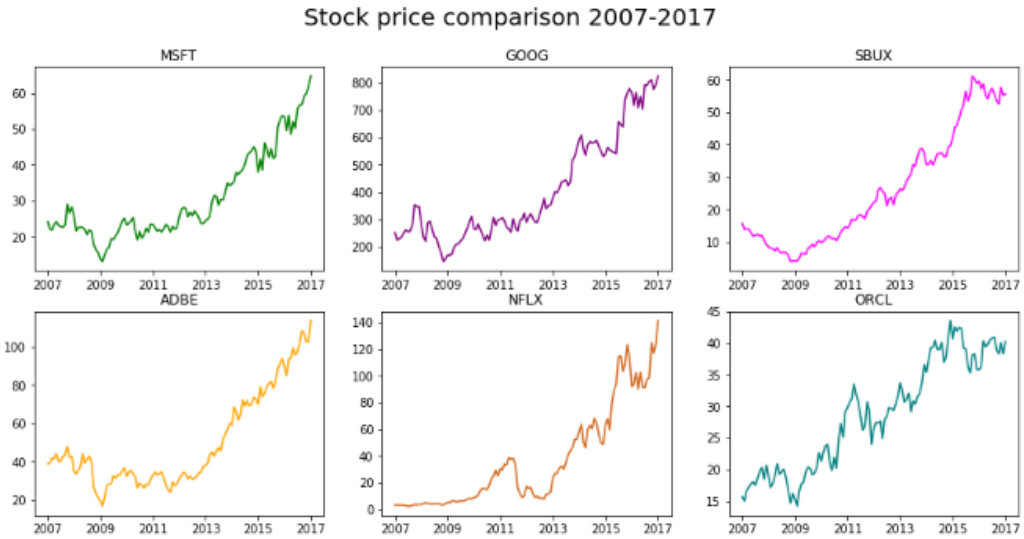


Image 66 - Time Series by line plot
Reference: https://miro.medium.com/max/1378/1*26SwjgcLr_j4_xY6EJ2Jfw.png

In the earlier figure,add_axes is used to used to add an axes to a figure whereas from above add_subplot adds multiple subplots to a figure. fig.add_subplot(237) cannot be done as there are only 6 subplots possible.
We can see that the tech company stocks are following an upward trend showing positive results for traders to invest in stocks.

Boxplot and Violin plot

Boxplot
Boxplot gives a nice summary of the data. It helps in understanding our distribution better.
When to use: It should be used when we require to use the overall statistical information on the distribution of the data. It can be used to detect outliers in the data.
eg: Credit Score of Customer. We can get the max, min and much more information about the mark.

Understanding Boxplot

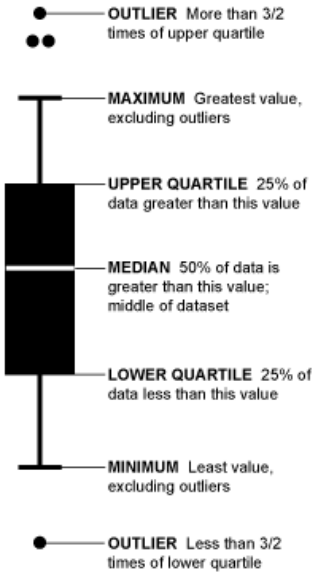


Image 67 - Boxplot
Reference: https://miro.medium.com/max/482/1*fCE_5juz235c6cmaOP_PDQ.png

From the above diagram, the line that divides the box into 2 parts represents the median of the data. The end of the box shows the upper quartile (75%) and the start of the box represents the lower quartile (25%). Upper Quartile is also called 3rd quartile and similarly, Lower Quartile is also called as 1st quartile. The region between lower quartile and the upper quartile is called as Inter Quartile Range (IQR) and it is used to approximate the 50% spread in the middle data (75–25=50%). The maximum is the highest value in data, similarly minimum is the lowest value in data, it is also called as caps. The points outside the boxes and between the maximum and maximum are called as whiskers, they show the range of values in data. The extreme points are outliers to the data. A commonly used rule is that a value is an outlier if it's less than lower quartile-1.5 * IQR or high than the upper quartile + 1.5* IQR.

In [3]:
```
exam_scores = exam_data[['math score', 'reading score', 'writing score']]
exam_scores.head()
```

Out[3]:

	math score	reading score	writing score
0	69	61	58
1	47	65	69
2	66	52	53
3	88	89	82
4	62	82	76

In [4]:
```
exam_scores.describe()
```

Out[4]:

	math score	reading score	writing score
count	100.000000	100.000000	100.000000
mean	67.150000	69.180000	67.780000
std	12.797885	13.832607	14.874954
min	36.000000	34.000000	33.000000
25%	56.000000	60.000000	57.750000
50%	68.000000	69.000000	68.500000
75%	76.000000	80.000000	77.250000
max	95.000000	99.000000	100.000000

In [5]:
```
# For plotting in boxplot, we convert it to an array
exam_scores_array = np.array(exam_scores)
exam_scores_array
```

Out[5]:
```
array([[ 69,  61,  58],
       [ 47,  65,  69],
       [ 66,  52,  53],
       [ 88,  89,  82],
       [ 62,  82,  76],
       [ 47,  69,  60],
       [ 71,  66,  74],
       [ 57,  62,  60],
```

Image 68 - Boxplot

Reference: https://miro.medium.com/max/1400/1*ad0xiuRs1X-CvlO-XSCKoA.png

Box plots

In [6]:
```
colors = ['blue', 'grey', 'lawngreen']

bp = plt.boxplot(exam_scores_array,
                 patch_artist=True,
                 notch=True)

for i in range(len(bp['boxes'])):

    bp['boxes'][i].set(facecolor=colors[i])

    bp['caps'][2*i + 1].set(color=colors[i])

plt.xticks([1, 2, 3], ['Math', 'Reading', 'Writing'])

plt.show()
```

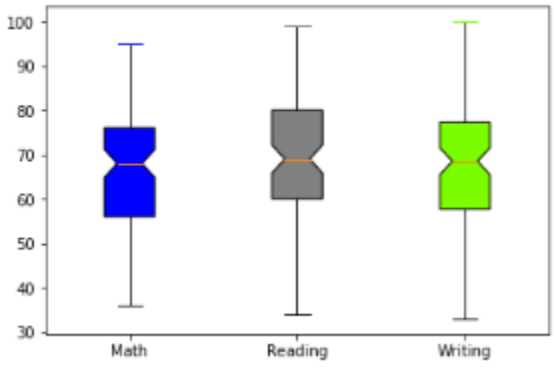


Image 69 - Boxplot

Reference: https://miro.medium.com/max/824/1*g12e9wIZOtKe8IHWGDNVBw.png

bp contains the boxplot components like boxes, whiskers, medians, caps. Seaborn another plotting library makes it easier to build custom plots than matplotlib. patch_artist makes the customization possible. notchmakes the median look more prominent.

A caveat of using boxplot is the number of observations in the unique value is not defined, Jitter Plot in Seaborn can overcome this caveat or Violinplot is also useful

Violin plot

Violin plot is a better chart than boxplot as it gives a much broader understanding of the distribution. It resembles a violin and dense areas point the more distribution of data otherwise hidden by box plots

When to use: It's an extension to boxplot. It should be used when we require a better intuitive understanding of data.

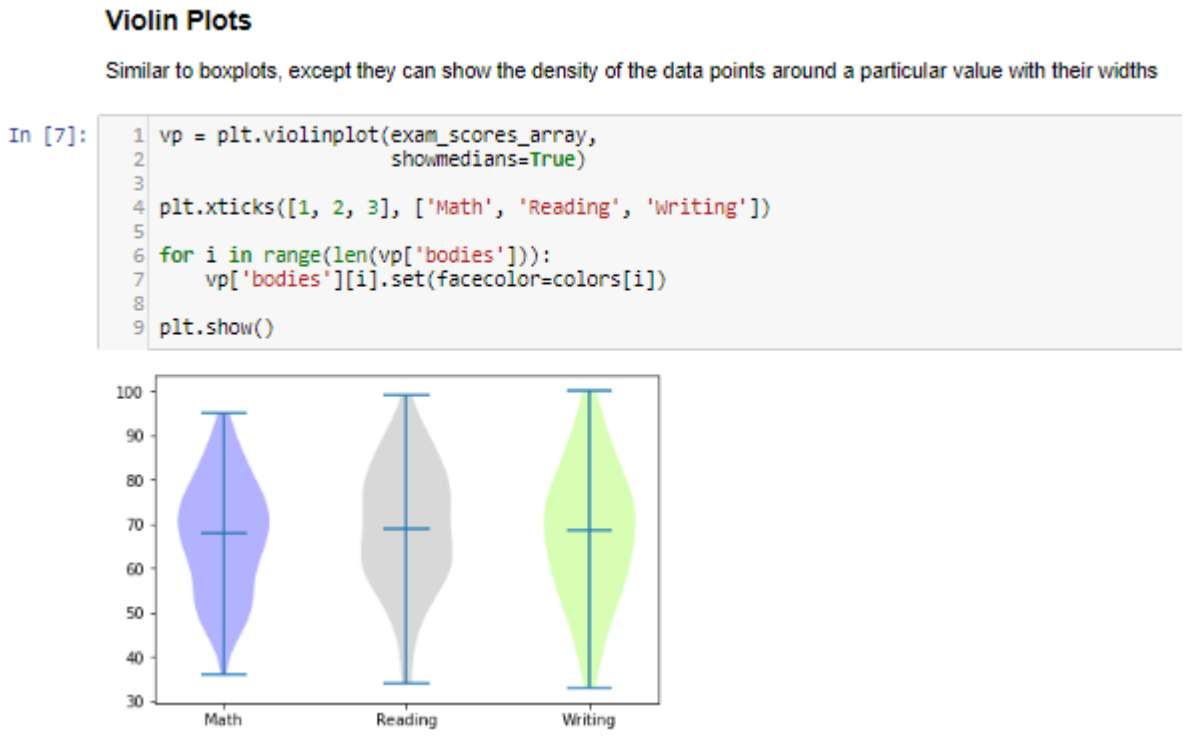


Image 70 – Violin Plot

Reference: https://miro.medium.com/max/1204/1*J9OnuX8f5BjlB3XZiyHkVA.png

The density of points in the middle seems more as students tend to score around average mostly in the subjects.

TwinAxis

TwinAxis helps in visualizing plotting 2 plots w.r.t to the y-axis and same x-axis.

When to use: It should when we require 2 plots or grouped data in the same direction.
Eg: Population, GDP data in the same x-axis (Date).
Plotting 2 Plots w.r.t the y-axis and same x-axis

```
In [2]: 1 austin_weather.head()
```

Out[2]:

	Date	TempHighF	TempAvgF	TempLowF	DewPointHighF	DewPointAvgF	DewPointLowF	HumidityHighPercent	HumidityAvgPercent	HumidityLowPercent
0	2013-12-21	74	60	45	67	49	43	93	75	57
1	2013-12-22	56	48	39	43	36	28	93	68	43
2	2013-12-23	58	45	32	31	27	23	76	52	27
3	2013-12-24	61	46	31	36	28	21	89	56	22
4	2013-12-25	58	50	41	44	40	36	86	71	56

5 rows × 21 columns

Extracting Date,Avg Temperature and Avg Wind Speed columns

```
In [3]: 1 austin_weather = austin_weather[['Date', 'TempAvgF', 'WindAvgMPH']].head(30)
        2 austin_weather
```

Out[3]:

	Date	TempAvgF	WindAvgMPH
0	2013-12-21	60	4
1	2013-12-22	48	6

Image 71 - TwinAxis
Reference: https://miro.medium.com/max/1400/1*dHwnU6ySte5aqNZYezQAgQ.png

Extracting important details i.e., Date for the x-axis, TempAvgF, and WindAvgMPH for the different y-axis.

```
In [4]:
 # Subplot for Plotting Figure
fig, ax_tempF = plt.subplots()

# Similar to fig=plt.figure(figsize=(12,6))
fig.set_figwidth(12)
fig.set_figheight(6)

# set x label which is common
ax_tempF.set_xlabel('Date')

# bottom= false disables ticks and labelbottom disables x-axis labels
ax_tempF.tick_params(axis = 'x',
                     bottom=False,
                     labelbottom=False)

# set left y-axis label
ax_tempF.set_ylabel('Temp (F)',
                     color='red',
                     size='x-large')

# set labelcolor and labelsize to the left Y-axis
ax_tempF.tick_params(axis='y',
                      labelcolor='red',
                      labelsize='large')

# plot AvgTemp on Y-axis to the left
ax_tempF.plot(austin_weather['Date'],
              austin_weather['TempAvgF'],
              color='red')

#ax_tempF.legend(0)

# twinx sets the same x-axis for both plots
ax_precip = ax_tempF.twinx()

#set Right y-axis label
ax_precip.set_ylabel('Avg Wind Speed (MPH)',
                      color='blue',
                      size='x-large')

# set labelcolor and labelsize to the Right Y-axis
ax_precip.tick_params(axis='y',
```

Image 72 - TwinAxis

Reference: https://miro.medium.com/max/1124/1*d0qOyBrqqoXjgybphi_RLQ.png

As we can there is only 1 axis,twinx() is used for twinning the x-axis and left y-axis is used for Temp and the right y-axis is used for WindMPH.

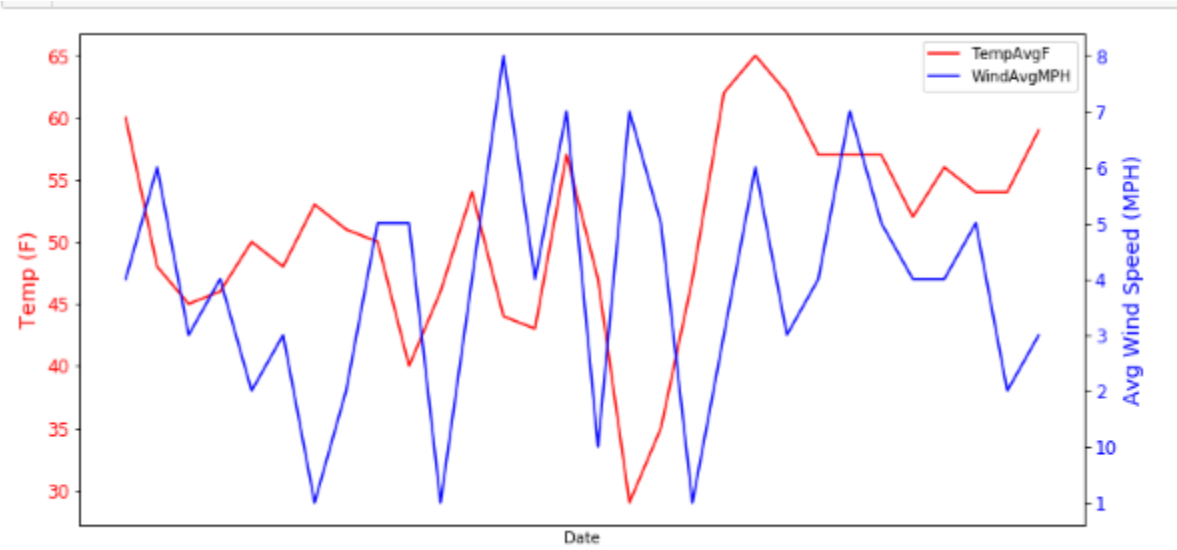


Image 73 - TwinAxis

Reference: https://miro.medium.com/max/1208/1*r8oHw5OPQhMcl4aUcYA2-w.png

Plotting the same data in different units and the same x-axis

Function to convert from fahrenheit to celsius

In [5]:
```
def fahrenheit2celsius(f):
    return (f - 32)*5/9
```

In [6]:
```
 # Subplot for Plotting Figure
fig, ax_tempF = plt.subplots()

# Similar to fig=plt.figure(figsize=(12,6))
fig.set_figwidth(12)
fig.set_figheight(6)

# set x label which is common
ax_tempF.set_xlabel('Date')

# bottom= false disables ticks and labelbottom disables x-axis labels
ax_tempF.tick_params(axis = 'x',
                     bottom=False,
                     labelbottom=False)

# set left y-axis label
ax_tempF.set_ylabel('Temp (F)',
                    color='red',
                    size='x-large')

# set labelcolor and labelsize to the left Y-axis
ax_tempF.tick_params(axis='y',
                     labelcolor='red',
                     labelsize='large')

# plot AvgTemp on Y-axis to the left
ax_tempF.plot(austin_weather['Date'],
              austin_weather['TempAvgF'],
              color='green')

# twinx sets the same x-axis for both plots
ax_tempC = ax_tempF.twinx()
```

Image 74 - Plotting the same data in different units and the same x-axis
Reference: https://miro.medium.com/max/1020/1*5ScqG3lYngJ1-eHUWaNbhA.png

The function is defined for calculating different unit of data i.e convert from Fahrenheit to Celsius.

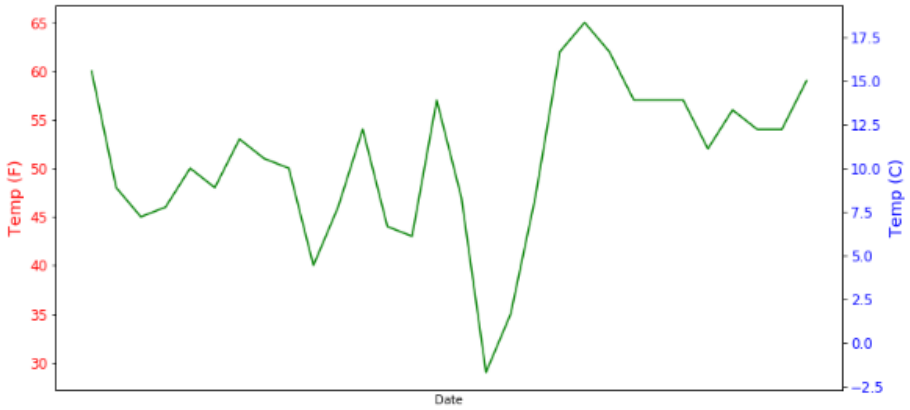


Image 75 - Plotting the same data in different units and the same x-axis
Reference: https://miro.medium.com/max/1222/1*lbNAVBW3_4x2aEsmcgem0A.png

We can see that to the left y-axis Temp in Fahrenheit is plotted and to the right x-axis Temp in Celsius is plotted.

Stack Plot and Stem Plot

Stack plot visualizes data in stacks and shows the distribution of data over time.
When to use: It is used for checking multiple variable area plots in a single plot.
Eg: It is useful in understanding the change of distribution in multiple variables over an interval.

```
In [2]:  1 np_data.head()
```

Out[2]:

	Year	Badlands	GrandCanyon	BryceCanyon
0	1961	833300	1253000	264800
1	1962	1044800	1447400	251000
2	1963	1074000	1539500	289500
3	1964	1079800	1576600	300300
4	1965	1091300	1689200	366800

```
In [3]:  1 x = np_data['Year']
```

```
In [4]:  1 y = np.vstack([np_data['Badlands'],
         2                np_data['GrandCanyon'],
         3                np_data['BryceCanyon']])
```

Image 76 - Stack Plot
Reference: https://miro.medium.com/max/662/1*BxpiEY02pFavicN3H4-VIg.png

As stack plot requires stacking, it is done in using np.vstack()

Stack Plots

It is used stack plots and visualize data overtime.

In [5]:
```
# Labels for each stack
labels = ['Badlands',
          'GrandCanyon',
          'BryceCanyon']

# Colors for each stack
colors = ['sandybrown',
          'tomato',
          'skyblue']

#  Similar to pandas df.plot.area()
plt.stackplot(x, y,
              labels=labels,
              colors=colors,
              edgecolor='black')

# Plots legend to the upperleft of Figure
plt.legend(loc=2)

plt.show()
```

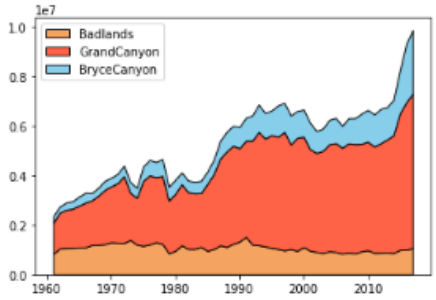


Image 77 - Stack Plot
Reference: https://miro.medium.com/max/684/1*Vco9sE3BfPzN8Wm4G4hojg.png

plt.stackplot takes in 1st argument numeric data i.e year and 2nd argument the vertically stacked data i.e the Nationalparks.

Percentage Stacked plot

Similar to stack plot but each data is converted into a percentage of distribution it holds.

Percentage Stacked Area Chart

Stack plots based on percentage in distribution.

```
In [6]:
plt.figure(figsize=(10,7))

data_perc = np_data.divide(np_data.sum(axis=1), axis=0)

plt.stackplot(x,
              data_perc["Badlands"],data_perc["GrandCanyon"],data_perc["BryceCanyon"],
              edgecolor='black',
              colors=colors,
              labels=labels)

plt.legend(loc='center left', bbox_to_anchor=(1, 0.5))

plt.show()
```

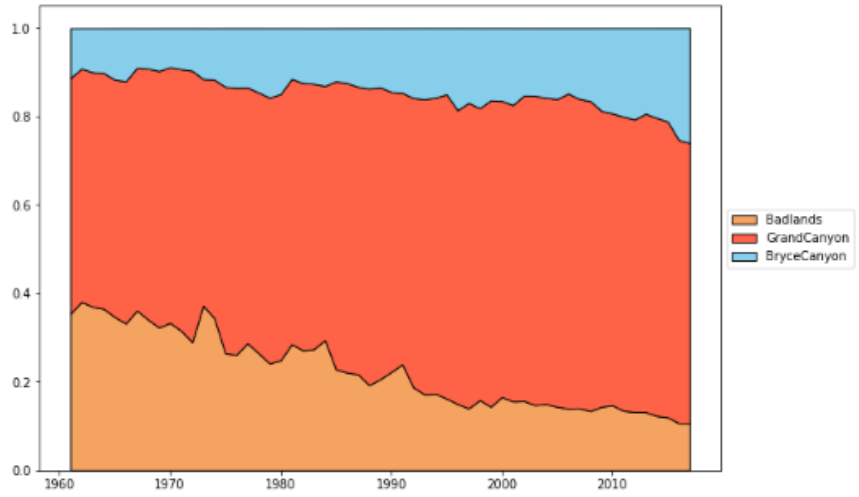


Image 78 - Stack Plot

Reference: https://miro.medium.com/max/1252/1*-7K1U1plketjCnpk8fVTXw.png

data_prec is used to divide the overall percentage into individual percentage distributions. s= np_data.sum(axis=1) calculates sum along columns, np_data.divide(s,axis=0) divides data along rows.

Stem Plot

Stemplot even takes negative values, so the difference is taken of data and is plotted over time.

When to use:

It is similar to a stack plot but the difference helps in comparing the data points.

Stem Plots

Similar to Stack Plots but it helps us view negative numbers, i.e the difference in data over time.

In [7]:
```
np_stem= np_data.copy()
```

In [8]:
```
np_stem[['Badlands',
         'GrandCanyon',
         'BryceCanyon']] = np_data[['Badlands',
                                    'GrandCanyon',
                                    'BryceCanyon']].diff()

np_stem.head()
```

Out[8]:

	Year	Badlands	GrandCanyon	BryceCanyon
0	1961	NaN	NaN	NaN
1	1962	211500.0	194400.0	-13800.0
2	1963	29200.0	92100.0	38500.0
3	1964	5800.0	37100.0	10800.0
4	1965	11500.0	112800.0	66500.0

Image 79 - Stem Plot
Reference: https://miro.medium.com/max/1040/1*hKw-F9uSY-5oPhW7ofdamg.png

diff() is used to find the difference between previous data and is stored in another copy of the data. The first data point is NaN (Not a Number) as it doesn't contain any previous data for calculating the difference.

In [9]:
```
plt.figure(figsize=(15,11))

plt.suptitle('Change in Number of Visitors', y=0.94)

plt.subplot(311)
plt.stem(np_stem['Year'],
         np_stem['Badlands'],
         markerfmt = 'b_',
         linefmt = 'r--',
         basefmt = 'g:')
plt.title('GrandCanyon')

plt.subplot(312)
plt.stem(np_stem['Year'],
         np_stem['GrandCanyon'],
         markerfmt = 'b_',
         linefmt = 'r--',
         basefmt = 'g:')
plt.title('Badlands')

plt.subplot(313)
plt.stem(np_stem['Year'],
         np_stem['BryceCanyon'],
         markerfmt = 'b_',
         linefmt = 'r--',
         basefmt = 'g:')
plt.title('BryceCanyon')

plt.show()
```

Image 80 - Stem Plot
Reference: https://miro.medium.com/max/1400/1*SQsTcZvyGCu3nWhoReBWyw.png

(31n)Subplots are created to accommodate 3 rows 1 column subplots in the figure. plt.stem() takes the 1st argument as numeric data i.e year and 2nd argument as numeric data of the National Park visitors.

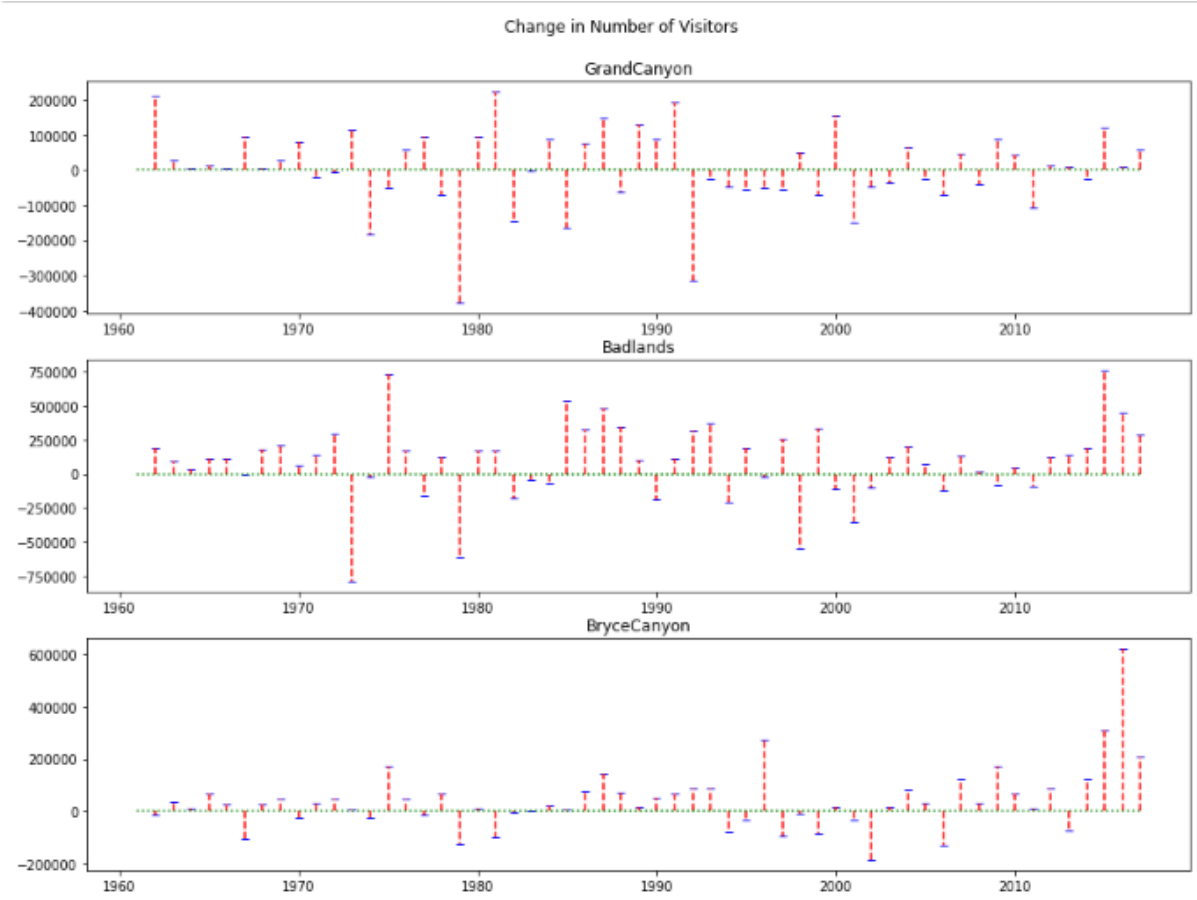


Image 81 - Stem Plot
Reference: https://miro.medium.com/max/1400/1*AlIkWpXoCSdzut5xwJ4jEw.png

Bar Plot

Bar Plot shows the distribution of data over several groups. It is commonly confused with a histogram which only takes numerical data for plotting. It helps in comparing multiple numeric values.

When to use:

It is used when to compare between several groups.

Eg: Student marks in an exam.

```
In [5]:
plt.figure(figsize=(12,6))

x=range(10)
plt.bar(x,top_10_p['population']/10**9)
plt.xticks(x,top_10_p['country'])
plt.xlabel('Countries')
plt.ylabel('Population in Billions')
plt.title('10 Most Populous Countries in 2007')
plt.show()
```

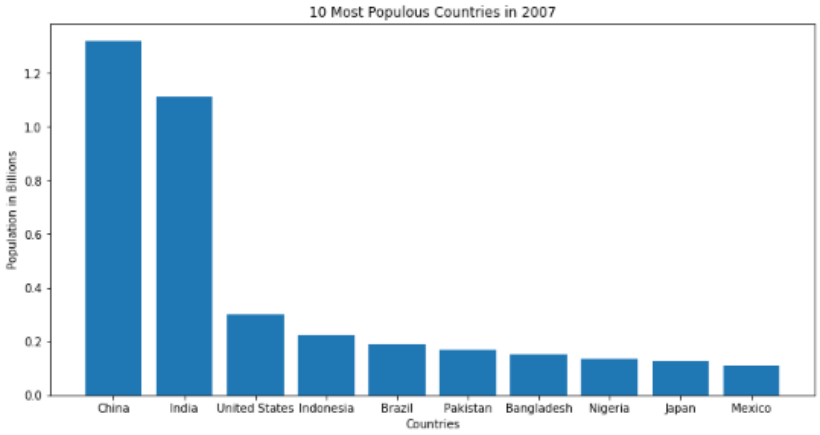


Image 82 - Bar Plot

Reference: https://miro.medium.com/max/1296/1*xKfvb2z9GY3MS0Tp9yrytg.png

plt.bar() takes the 1st argument as labels in numeric format and 2nd argument for the value it represents w.r.t to the plots.

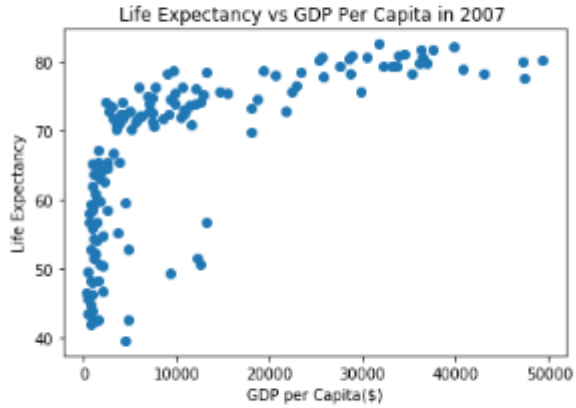
Scatter Plot

Scatter plot helps in visualizing 2 numeric variables. It helps in identifying the relationship of the data with each variable i.e correlation or trend patterns. It also helps in detecting outliers in the plot.

When to use:

It is used in Machine learning concepts like regression, where x and y are continuous variables. It is also used in clustering scatters or outlier detection.

```
In [4]:
plt.scatter(data_2007['gdpPerCapita'],data_2007['lifeExpectancy'])
plt.title('Life Expectancy vs GDP Per Capita in 2007 ')
plt.xlabel('GDP per Capita($)')
plt.ylabel('Life Expectancy')
plt.show()
```




```
In [5]:
data_2007['gdpPerCapita'].corr(data_2007['lifeExpectancy'])
Out[5]: 0.6786623986777587
```

Image 83 - Scatter Plot
Reference: https://miro.medium.com/max/974/1*u2qXX6T1h4oRppTe_y0d4g.png

plt.scatter() takes 2 numeric arguments for scattering data points in the plot. It is similar to line plot except without the connected straight lines. By corr we mean correlation and it means that how correlated GDP is with life expectancy, as we can see that it is positive it means as GDP of a country increases, life expectancy too increases.

```
In [6]:
plt.scatter(np.log10(data_2007['gdpPerCapita']),data_2007['lifeExpectancy'])
plt.title('Life Expectancy vs GDP Per Capita in 2007 ')
plt.xlabel('GDP per Capita($)')
plt.ylabel('Life Expectancy')
plt.show()
```


```
In [7]:
np.log10(data_2007['gdpPerCapita']).corr(data_2007['lifeExpectancy'])
Out[7]: 0.8089802514849209
```

Image 84 - Scatter Plot

Reference: https://miro.medium.com/max/1086/1*o8g6tfbhafhcq333IG1vIw.png

By taking the log of GDP, we can there is a much better correlation as we can fit points better, it converts GDP in log scale i.e log($1000)=3.

3D Scatterplot

3D Scatterplot helps in visualizing 3 numerical variables in a three- dimensional plot.

```
In [8]:
from mpl_toolkits.mplot3d import Axes3D
fig = plt.figure(figsize=(10,6))
ax = fig.add_subplot(111, projection='3d')
ax.scatter(countries['year'],countries['gdpPerCapita'],countries['lifeExpectancy'], s=30)
plt.show()
```

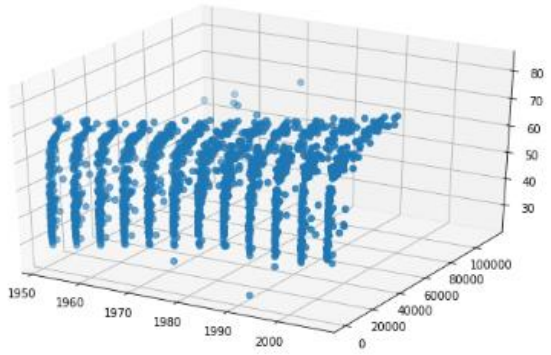


Image 85 - 3D Scatterplot
Reference: https://miro.medium.com/max/1320/1*1vqIWOwAZXcl9RJ6Iutp6g.png

It is similar to scatter except we add 3 numerical variables this time. By looking at the plot we can make an inference that as the year and GDP increases, life expectancy too increases.

Advanced data visualization using seaborn

Data visualization occupies a special place at the heart of all data-related professions. Nothing is more satisfying for a data scientist than to take a large set of random numbers and turn it into a beautiful visual.

The majority of data visuals created by data scientists are created with Python and its twin visualization libraries: Matplotlib and Seaborn. Matplotlib and Seaborn are widely used to create graphs that enable individuals and companies to make sense of terabytes of data.

What is Seaborn?

So, what are these two libraries, exactly?

Matplotlib is the king of Python data visualization libraries and makes it a breeze to explore tabular data visually.

Seaborn is another Python data visualization library built on top of Matplotlib that introduces some features that weren't previously available, and, in this tutorial, we'll use Seaborn.

To follow along with this project, you'll also need to know about Pandas, a powerful library that manipulates and analyzes tabular data.

In this blog post, we'll learn how to perform data analysis through visualizations created with Seaborn. You will be introduced to histograms, KDEs, bar charts, and more. By the end, you'll have a solid understanding of how to visualize data.

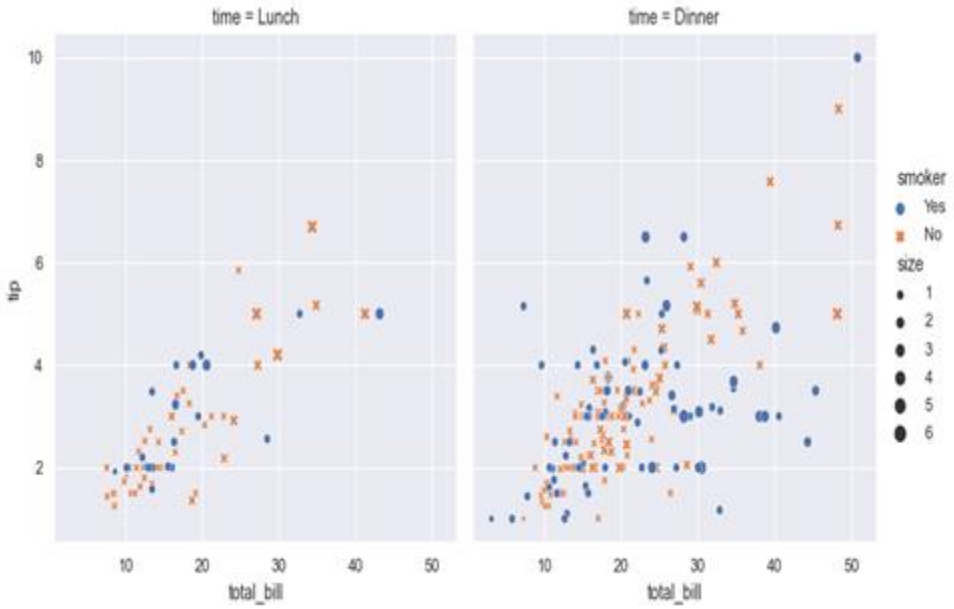


Image 86 - What is Seaborn?
Reference: https://seaborn.pydata.org/_images/introduction_1_0.png

Installing the libraries and loading the data

We will start by installing the libraries and importing our data. Running the below command will install the Pandas, Matplotlib, and Seaborn libraries for data visualization:

pip install pandas matplotlib seaborn

Now, let's import the libraries under their standard aliases:

```
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
```

Next, load in the data to be analyzed. The dataset contains physical measurements of 54,000 diamonds and their prices. You can download the original dataset as a CSV file from here on Kaggle, but we will be using a shortcut:

```
diamonds = sns.load_dataset("diamonds")
```

Because the dataset is already built into Seaborn, we can load it as pandas.DataFrame using the load_dataset function.

```
>>> type(diamonds)
```

Output:
pandas.core.frame.DataFrame

Exploring the dataset
Before we dive head-first into visuals, let's ensure we have a high-level understanding of our dataset:
>>> diamonds.head()

	carat	cut	color	clarity	depth	table	price	x	y	z
0	0.23	Ideal	E	SI2	61.5	55.0	326	3.95	3.98	2.43
1	0.21	Premium	E	SI1	59.8	61.0	326	3.89	3.84	2.31
2	0.23	Good	E	VS1	56.9	65.0	327	4.05	4.07	2.31
3	0.29	Premium	I	VS2	62.4	58.0	334	4.20	4.23	2.63
4	0.31	Good	J	SI2	63.3	58.0	335	4.34	4.35	2.75

Image 87 - Installing the libraries and loading the data
Reference: https://blog.logrocket.com/wp-content/uploads/2021/11/dataset-graph.png

We have used the handy head function of Pandas that prints out the first five rows of the data frame. head should be the first function you use when you load a dataset into your environment for the first time.
Notice the dataset has ten variables— three categorical and seven numeric.

- Carat: weight of a diamond
- Cut: the cut quality with five possible values in increasing order: Fair, Good, Very Good, Premium, Ideal
- Color: the color of a diamond with color codes from D (the best) to J (the worst)
- Clarity: the clarity of a diamond with eight clarity codes
- X: length of a diamond (mm)
- Y: the height of a diamond (mm)
- Z: depth of a diamond (mm)
- Depth: total depth percentage calculated as Z / average(X, Y)
- Table: the ratio of the height of a diamond to its widest point
- Price: diamond price in dollars

Instead of counting all variables one by one, we can use the shape attribute of the data frame:
>>> **diamonds.shape**
(53940, 10)

There are 53,940 diamonds recorded, along with their ten different features. Now, let's print a five-number summary of the dataset:

>>> **diamonds.describe()**

	carat	depth	table	price	x	y	z
count	53940.000000	53940.000000	53940.000000	53940.000000	53940.000000	53940.000000	53940.000000
mean	0.797940	61.749405	57.457184	3932.799722	5.731157	5.734526	3.538734
std	0.474011	1.432621	2.234491	3989.439738	1.121761	1.142135	0.705699
min	0.200000	43.000000	43.000000	326.000000	0.000000	0.000000	0.000000
25%	0.400000	61.000000	56.000000	950.000000	4.710000	4.720000	2.910000
50%	0.700000	61.800000	57.000000	2401.000000	5.700000	5.710000	3.530000
75%	1.040000	62.500000	59.000000	5324.250000	6.540000	6.540000	4.040000
max	5.010000	79.000000	95.000000	18823.000000	10.740000	58.900000	31.800000

Image 88 - Exploring the Dataset
Reference: https://blog.logrocket.com/wp-content/uploads/2021/11/dataset-summary.png

The describe function displays some critical metrics of each numeric variable in a data frame. Here are some observations from the above output:

- The cheapest diamond in the dataset costs $326, while the most expensive costs almost 60 times more,$18,823
- The minimum weight of a diamond is 0.2 carats, while the max is 5.01. The average weight is ~0.8
- Looking at the mean of X and Y features, we see that diamonds, on average, have the same height and width

Now that we are comfortable with the features in our dataset, we can start plotting them to uncover more insights.

Performing univariate analysis with Seaborn

In the previous section, we started something called "Exploratory Data Analysis" (EDA), which is the basis for any data-related project.

The goal of EDA is simple — get to know your dataset at the deepest level possible. Becoming intimate with the data and learning its relationships between its variables is an absolute must.

Completing a successful and thorough EDA lays the groundwork for future stages of your data project.

We have already performed the first stage of EDA, which was a simple "get acquainted" step. Now, let's go deeper, starting with univariate analysis.

As the name suggests, we'll explore variables one at a time, not the relationships between them just yet. Before we start plotting, we take a small dataset sample because 54,000 is more than we need, and we can learn about the data set pretty well with just 3,000 and to prevent overplotting.

sample = diamonds.sample(3000)
To take a sample, we use the sample function of pandas, passing in the number of random data points to include in a sample.

Creating histograms in Seaborn
Now, we create our first plot, which is a histogram:
sns.histplot(x=sample["price"])

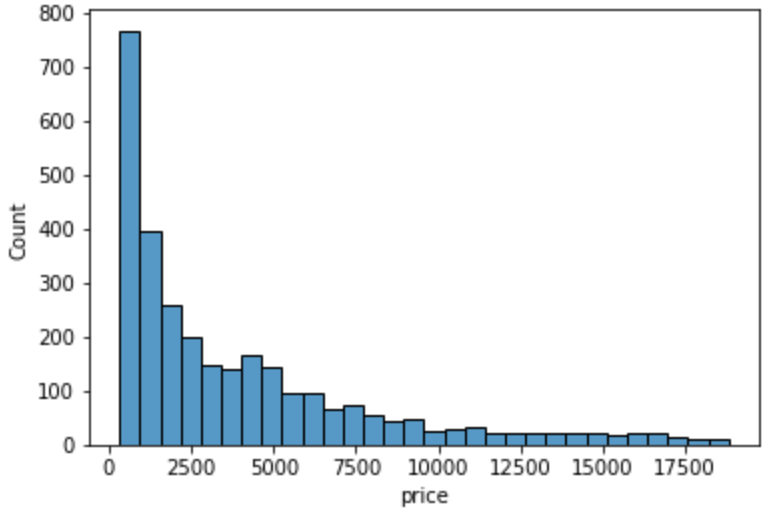


Image 89 - Creating histograms in Seaborn
Reference: https://blog.logrocket.com/wp-content/uploads/2021/11/histogram-count.png

Histograms only work on numeric variables. They divide the data into an arbitrary number of equal-sized bins and display how many diamonds go into each bin. Here, we can approximate that nearly 800 diamonds are priced between 0 and 1000.
Each bin contains the count of diamonds. Instead, we might want to see what percentage of the diamonds falls into each bin. For that, we will set the stat argument of the histplot function to percent:
>>> **sns.histplot(sample["price"], stat="percent")**

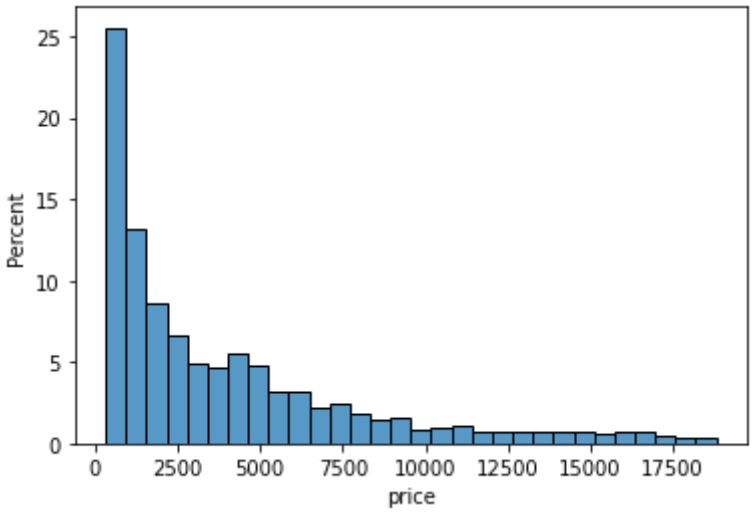


Image 90 - Creating histograms in Seaborn
Reference: https://blog.logrocket.com/wp-content/uploads/2021/11/histogram.png

Now, the height of each bar/bin shows the percentage of the diamonds. Let's do the same for the carat of the diamonds:

```
sns.histplot(sample["carat"], stat="percent")
```

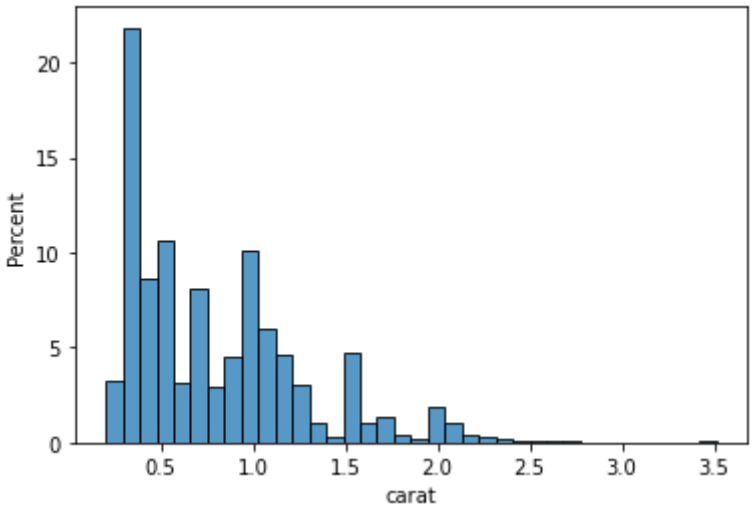


Image 91 - Creating histograms in Seaborn
Reference: https://blog.logrocket.com/wp-content/uploads/2021/11/diamod-histogram.png

Looking at the first few bars, we can conclude that the majority of the diamonds weigh less than 0.5 carats. Histograms aim to take a numeric variable and show what its shape generally looks like. Statisticians look at the distribution of a variable.

However, histograms aren't the only plots that do the job. There is also a plot called KDE Plot (Kernel Density Estimate), which uses some fancy math under the hood to draw curves like this:

```
sns.kdeplot(sample["table"])
```

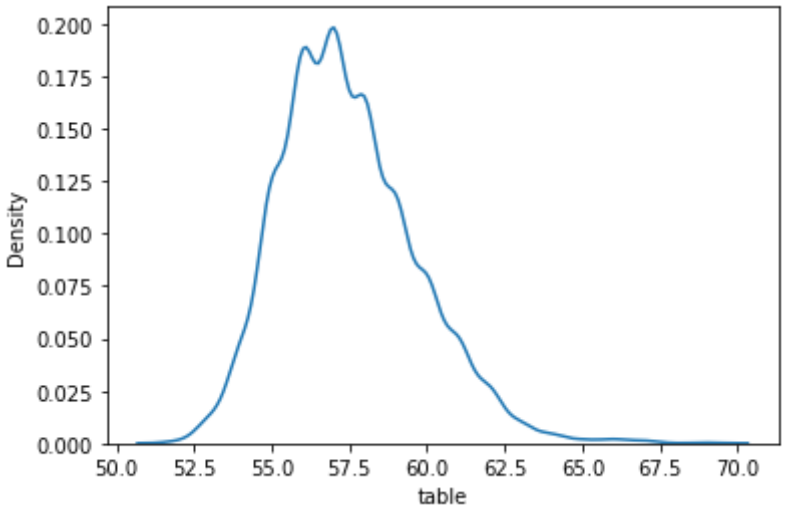


Image 92 - Creating histograms in Seaborn
Reference: https://blog.logrocket.com/wp-content/uploads/2021/11/density-table.png

Creating the KDE plot of the table variable shows us that the majority of diamonds measure between 55.0 and 60.0. At this point, I will leave it to you to plot the KDEs and histograms of other numeric variables because we have to move on to categorical features.

Creating count plots in Seaborn

The most common plot for categorical features is a countplot. Passing the name of a categorical feature in our dataset to Seaborn's countplot draws a bar chart, with each bar height representing the number of diamonds in each category. Below is a countplot of diamond cuts:

sns.countplot(sample["cut"])

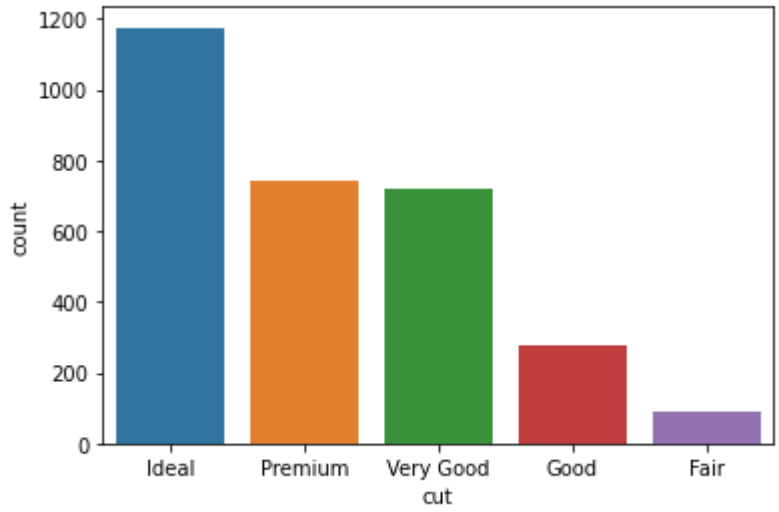


Image 93 - Creating count plots in Seaborn
Reference: https://blog.logrocket.com/wp-content/uploads/2021/11/dataset-color.png

We can see that our dataset consists of much more ideal diamonds than premium or very good diamonds. Here is a countplot of colors for the interested:

sns.countplot(sample["color"])

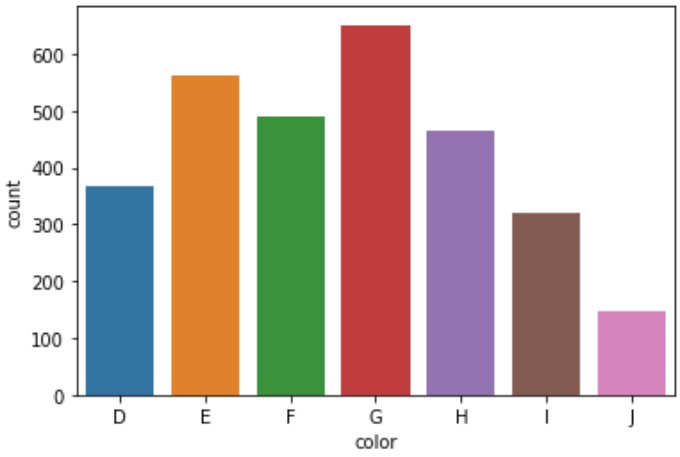


Image 94 - Creating count plots in Seaborn
Reference: https://blog.logrocket.com/wp-content/uploads/2021/11/countplot-colors.png

This concludes the univariate analysis section of the EDA.

Performing bivariate analysis with Seaborn

Now, let's look at the relationships between two variables at a time. Let's start with the connection between diamond carats and price.

Creating scatterplots

We already know that diamonds with higher carats cost more. Let's see if we can visually capture this trend:

sns.scatterplot(x=sample["carat"], y=sample["price"])

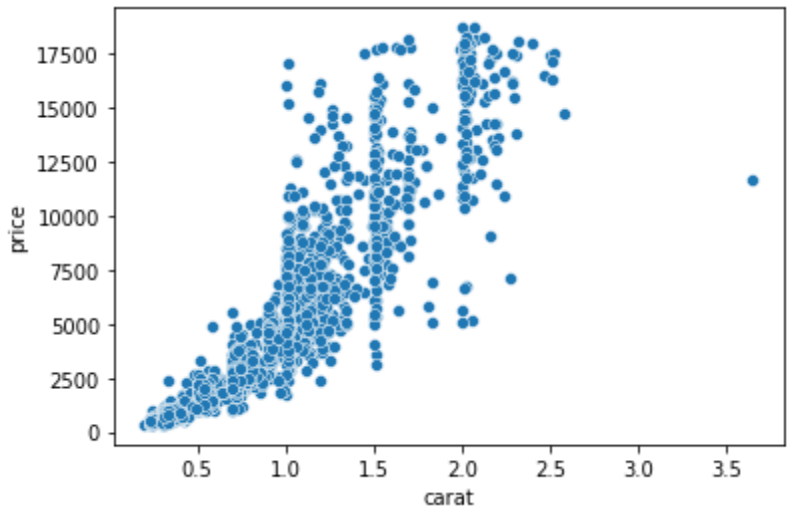


Image 95 - Creating scatterplots
Reference: https://blog.logrocket.com/wp-content/uploads/2021/11/carat-count.png

Here, we are using another Seaborn function that plots a scatter plot. Scatterplots are one of the most widely-used charts because they accurately show the relationships between two variables by using a cloud of dots.

Above, each dot represents a single diamond. The dots' positions are determined by their carat and price measurements, which we passed to X and Y parameters of the scatterplot function.

The plot confirms our assumptions — heavier diamonds tend to be more expensive. We are drawing this conclusion based on the curvy upward trend of the dots.

sns.scatterplot(x=sample["depth"], y=sample["table"])

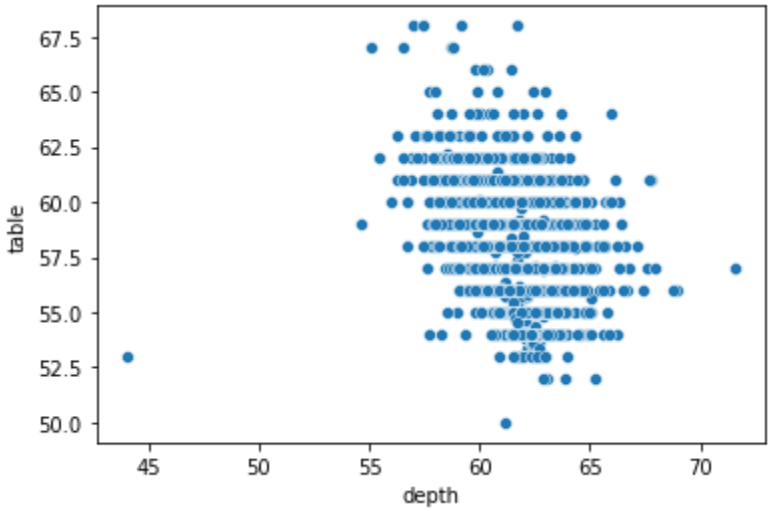


Image 96- Creating scatterplots

Reference: https://blog.logrocket.com/wp-content/uploads/2021/11/scatterplot.png

Let's try plotting depth against the table. Frankly, this scatterplot is disappointing because we can't draw a tangible conclusion as we did with the previous one.

Building boxplots

Another typical bivariate plot is a boxplot, which plots the distribution of a variable against another based on their five-number summary:

sns.boxplot(x=sample["color"], y=sample["price"])

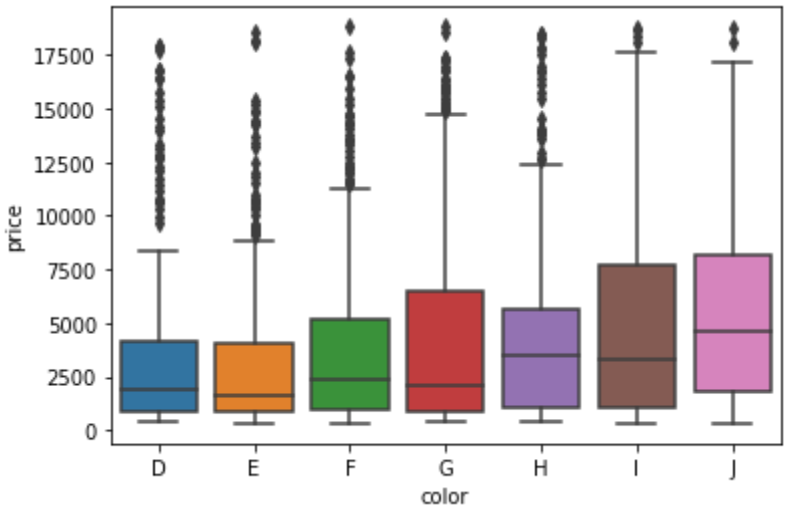


Image 97 - Creating scatterplots
Reference: https://blog.logrocket.com/wp-content/uploads/2021/11/boxplaot.png

The boxplot above shows the relationship between each color category and their respective prices. The horizontal vertices at the bottom and top of each vertical line of a box represent that category's minimum and maximum values. The edges of the boxes, specifically the bottom and top edges, represent the 25th and 75th percentiles.

In other words, the bottom edge of the first box tells us that 25% of D-colored diamonds cost less than about $1,250, while the top edge says that 75% of diamonds cost less than about $4,500. The little horizontal line in the middle denotes the median, the 50% mark.

The dark dots above are outliers. Let's plot a boxplot of diamond clarities and their relationship with carat:

sns.boxplot(diamonds["clarity"], diamonds["carat"])

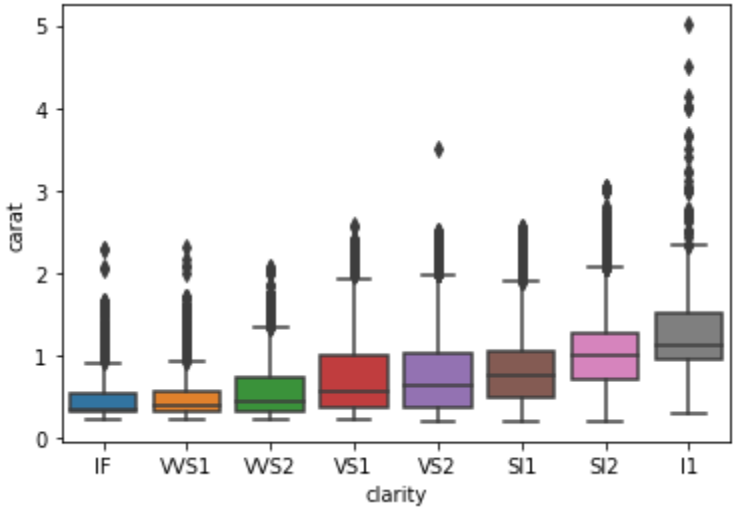


Image 98 - Building boxplots
Reference: https://blog.logrocket.com/wp-content/uploads/2021/11/diamond-clarity.png

Pandas profiling for report generation

What is Pandas Profiling

Pandas Profiling is an open-source python library, which allows you to do your EDA very quickly. By the way, it also generates an interactive HTML report, which you can show to anyone. Imagine going to your boss, who doesn't code, with an interactive description of the company's data. Great for your branding, right?
These are some of the things you get in your report:

- Type inference: detect the types of columns in a Data Frame.
- Essentials: type, unique values, missing values.
- Quantile statistics like minimum value, Q1, median, Q3, maximum, range, interquartile range.
- Descriptive statistics like mean, mode, standard deviation, sum, median absolute deviation, coefficient of variation, kurtosis, skewness.
- Most frequent values.
- Histogram.
- Correlations highlighting of highly correlated variables, Spearman, Pearson and Kendall matrices.
- Missing values matrix, count, heat-map and dendrogram of missing values.
- Text analysis learns about categories (Uppercase, Space), scripts (Latin, Cyrillic) and blocks (ASCII) of text data.
- File and Image analysis extract file sizes, creation dates and dimensions and scan for truncated images or those containing EXIF information.

Given this, let's get going.

How to install Pandas Profiling

First of all, you need to install the package.
#installing Pandas Profiling
!pip install https://github.com/pandas-profiling/pandas-profiling/archive/master.zip -q
Now, let's import both pandas and panda_profiling.
#importing modules
from pandas_profiling import ProfileReport
import pandas as pd
We will be using the Titanic dataset to complete our analysis, let's import it:
#linking df to our dataset
df = pd.read_csv("https://raw.githubusercontent.com/datasciencedojo/

datasets/master/titanic.csv")

After you import it, you should always take a look at your dataset and then merely link report to it:

report = ProfileReport(df)

Now you simply have “to tell” Pandas Profiling to make a report out of your dataset.

report.to_notebook_iframe()

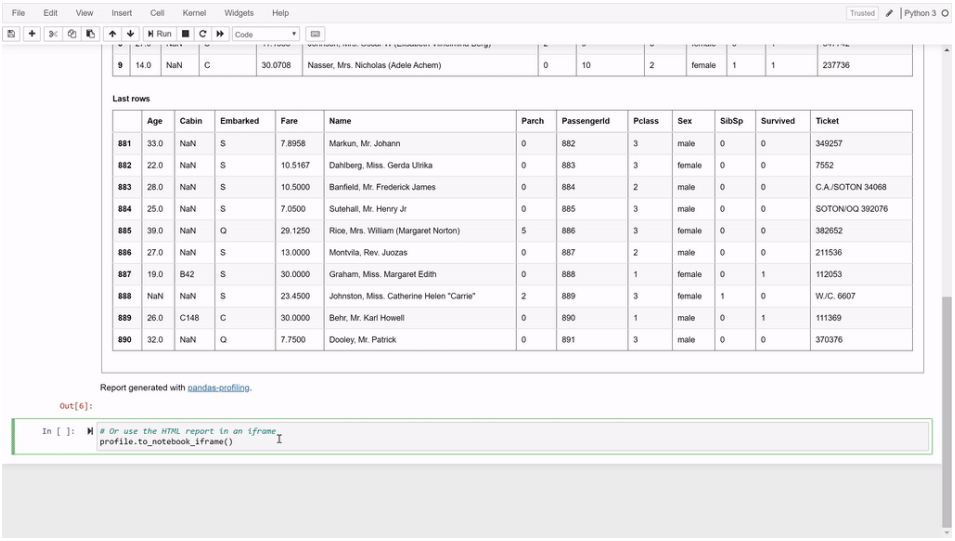


Image 99 - How to install Pandas Profiling
Reference: https://miro.medium.com/max/1400/0*LtaSDCg7z7BMMkJ5

If you use a Jupyter Notebook, your report is embedded in it. However, you may want to use it in other places, and Pandas Profiling also allows you to do that. Just type this to save your report as an HTML file:

report.to_file('file_name')

If you want the HTML source “code” (don’t kill me for calling it code), which would be quite rare, however possible, just type:

report.to_html()

It will return the whole HTML source code.

Image 100 - How to install Pandas Profiling
Reference: https://miro.medium.com/max/1400/1*qeXNHD0ZQA1IYJMF1tTbxw.png

You can even save it as a JSON file:

As a string

```
json_data = profile.to_json()
# As a file
profile.to_file("your_report.json")
```

Need for data visualization

What is Data Visualization?

With so much information being collected through data analysis in the business world today, we must have a way to paint a picture of that data so we can interpret it. Data visualization gives us a clear idea of what the information means by giving it visual context through maps or graphs. This makes the data more natural for the human mind to comprehend and therefore makes it easier to identify trends, patterns, and outliers within large data sets.

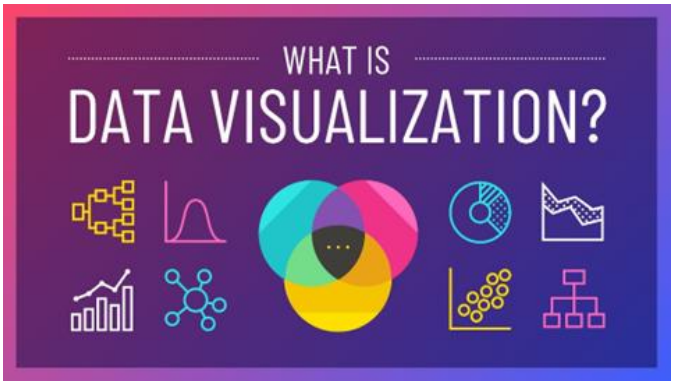


Image 101- What is Data Visualization?
Reference: https://venngage-wordpress.s3.amazonaws.com/uploads/2020/06/What-is-Data-Visualization-Blog-Header.jpg

Why is Data Visualization Important?

No matter what business or career you've chosen, data visualization can help by delivering data in the most efficient way possible. As one of the essential steps in the business intelligence process, data visualization takes the raw data, models it, and delivers the data so that conclusions can be reached. In advanced analytics, data scientists are creating machine learning algorithms to better compile essential data into visualizations that are easier to understand and interpret.

Specifically, data visualization uses visual data to communicate information in a manner that is universal, fast, and effective. This practice can help companies identify which areas need to be improved, which factors affect customer satisfaction and dissatisfaction, and what to do with specific products (where should they go and who should they be sold to). Visualized data gives stakeholders, business owners, and decision-makers a better prediction of sales volumes and future growth.

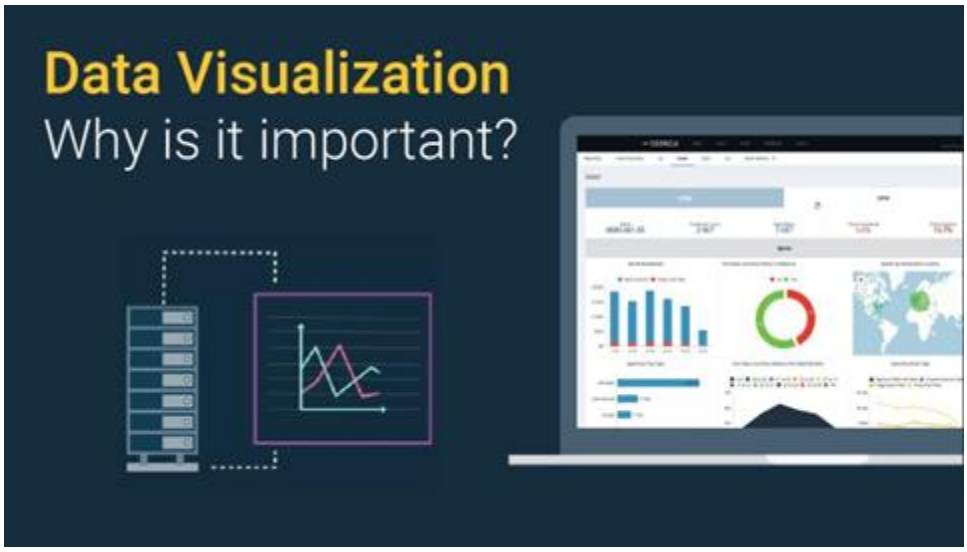


Image 101 - Why is Data Visualization Important?
Reference: https://i.ytimg.com/vi/VyhLRJVoIrI/maxresdefault.jpg

What Are the Benefits of Data Visualization?

Data visualization positively affects an organization's decision-making process with interactive visual representations of data. Businesses can now recognize patterns more quickly because they can interpret data in graphical or pictorial forms. Here are some more specific ways that data visualization can benefit an organization:

- Correlations in Relationships: Without data visualization, it is challenging to identify the correlations between the relationship of independent variables. By making sense of those independent variables, we can make better business decisions.
- Trends Over Time: While this seems like an obvious use of data visualization, it is also one of the most valuable applications. It's impossible to make predictions without having the necessary information from the past and present. Trends over time tell us where we were and where we can potentially go.
- Frequency: Closely related to trends over time is frequency. By examining the rate, or how often, customers purchase and when they buy gives us a better feel for how potential new customers might act and react to different marketing and customer acquisition strategies.
- Examining the Market: Data visualization takes the information from different markets to give you insights into which audiences to focus your attention on and which ones to stay away from. We get a clearer picture of the opportunities within those markets by displaying this data on various charts and graphs.
- Risk and Reward: Looking at value and risk metrics requires expertise because, without data visualization, we must interpret complicated spreadsheets and numbers. Once information is visualized, we can then pinpoint areas that may or may not require action.

- Reacting to the Market: The ability to obtain information quickly and easily with data displayed clearly on a functional dashboard allows businesses to act and respond to findings swiftly and helps to avoid making mistakes.

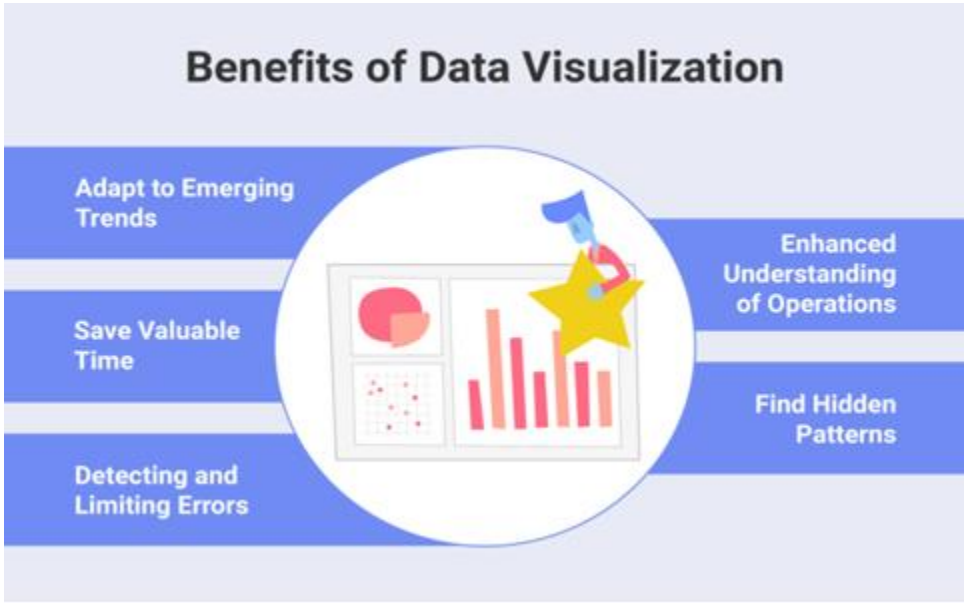


Image 102 - Benefits of Data Visualization
Reference: https://cdn.slingshotapp.io/wp-content/uploads/2021/09/slingshot-Benefits-of-Data-Visualization-768x505.png

Which Data Visualization Techniques are Used?

There are many different methods of putting together information in a way that the data can be visualized. Depending on the data being modeled, and what its intended purpose is, a variety of different graphs and tables may be utilized to create an easy to interpret dashboard. Some visualizations are manually created, while others are automated. Either way, there are many types to meet your visualization needs.

- Infographics: Unlike a single data visualization, infographics take an extensive collection of information and gives you a comprehensive visual representation. An infographic is excellent for exploring complex and highly-subjective topics.
- Heatmap Visualization: This method uses a graph with numerical data points highlighted in light or warm colors to indicate whether the data is a high-value or a low-value point. Psychologically, this data visualization method helps the viewer to identify the information because studies have shown that humans interpret colors much better than numbers and letters.
- Fever Charts: A fever chart shows changing data over a period of time. As a marketing tool, we could take the performance from the previous year and compare that to the prior

year to get an accurate projection of next year. This can help decision-makers easily interpret wide and varying data sources.

- Area Chart (or Graph): Area charts are excellent for visualizing the data's time-series relationship. Whether you're looking at the earnings for individual departments on a month-to-month basis or the popularity of a product since the 1980s, area charts can visualize this relationship.
- Histogram: Rather than looking at the trends over time, histograms are measuring frequencies instead. These graphs show the distribution of numerical data using an automated data visualization formula to display a range of values that can be easily interpreted.

Who Uses Data Visualization?

Data visualization is used across all industries to increase sales with existing customers and target new markets and demographics for potential customers. The World Advertising and Research Center (WARC) predicts that in 2020 half of the world's advertising dollars will be spent online, which means companies everywhere have discovered the importance of web data. As a crucial step in data analytics, data visualization gives companies critical insights into untapped information and messages that would otherwise be lost. The days of scouring through thousands of rows of spreadsheets are over, as now we have a visual summary of data to identify trends and patterns.

Machine Learning

Machine Learning is the field of study that gives computers the capability to learn without being explicitly programmed. ML is one of the most exciting technologies that one would have ever come across. As it is evident from the name, it gives the computer that makes it more similar to humans: *The ability to learn*.

Machine learning is a growing technology which enables computers to learn automatically from past data. Machine learning uses various algorithms for building mathematical models and making predictions using historical data or information. Currently, it is being used for various tasks such as image recognition, speech recognition, email filtering, Facebook auto-tagging, recommender system, and many more.

How does Machine Learning Work?

A Machine Learning system learns from historical data, builds the prediction models, and whenever it receives new data, predicts the output for it. The accuracy of predicted output depends upon the amount of data, as the huge amount of data helps to build a better model which predicts the output more accurately.

Suppose we have a complex problem, where we need to perform some predictions, so instead of writing a code for it, we just need to feed the data to generic algorithms, and with the help of these algorithms, machine builds the logic as per the data and predict the output. Machine learning has changed our way of thinking about the problem. The below block diagram explains the working of Machine Learning algorithm:

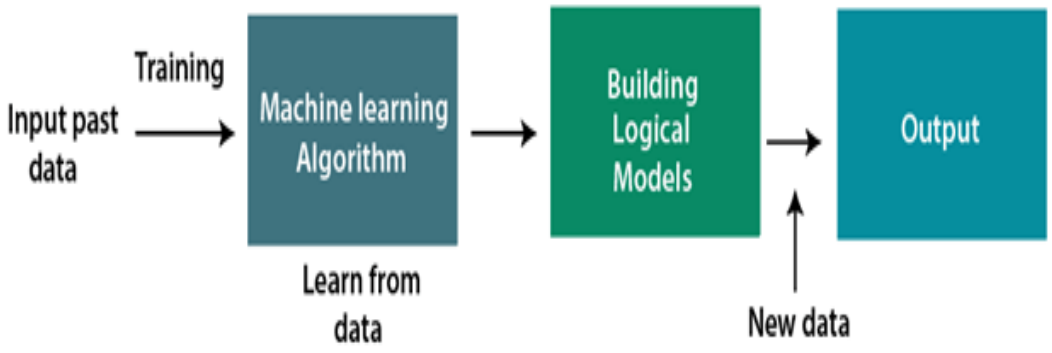


Image 103 - Machine Learning
Reference:https://static.javatpoint.com/tutorial/machine-learning/images/introduction-to-machine-learning2.png

Applications of Machine learning

Machine learning is a buzzword for today's technology, and it is growing very rapidly day by day. We are using machine learning in our daily life even without knowing it such as Google Maps, Google assistant, Alexa, etc. Below are some most trending real-world applications of machine learning:

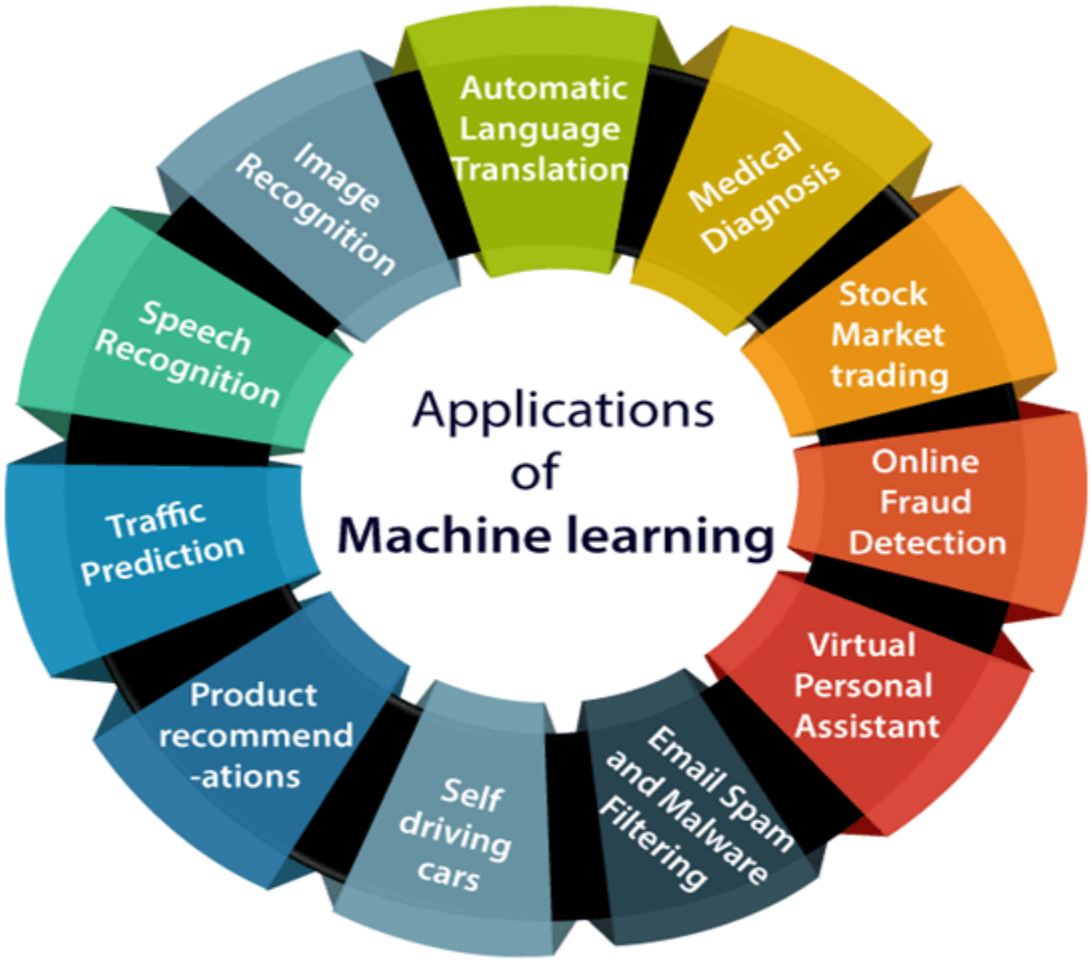


Image 104-Machine Learning Application
Reference: https://static.javatpoint.com/tutorial/machine-learning/images/applications-of-machine-learning.png

- **Image Recognition:**

  Image recognition is one of the most common applications of machine learning. It is used to identify objects, persons, places, digital images, etc. The popular use case of image recognition and face detection is, **Automatic friend tagging suggestion**:

Facebook provides us a feature of auto friend tagging suggestion. Whenever we upload a photo with our Facebook friends, then we automatically get a tagging suggestion with name, and the technology behind this is machine learning's face detection and recognition algorithm. It is based on the Facebook project named "Deep Face," which is responsible for face recognition and person identification in the picture.

- **Speech Recognition:**

While using Google, we get an option of "**Search by voice**," it comes under speech recognition, and it's a popular application of machine learning.

Speech recognition is a process of converting voice instructions into text, and it is also known as "**Speech to text**", or "**Computer speech recognition**." At present, machine learning algorithms are widely used by various applications of speech recognition. **Google assistant**, **Siri**, **Cortana**, and **Alexa** are using speech recognition technology to follow the voice instructions.

- **Traffic prediction:**

If we want to visit a new place, we take help of Google Maps, which shows us the correct path with the shortest route and predicts the traffic conditions. It predicts the traffic conditions such as whether traffic is cleared, slow-moving, or heavily congested with the help of two ways:

Real Time location of the vehicle form Google Map app and sensors

Average time has taken on past days at the same time.

Everyone who is using Google Map is helping this app to make it better. It takes information from the user and sends back to its database to improve the performance.

- **Product recommendations:**

Machine learning is widely used by various e-commerce and entertainment companies such as **Amazon**, **Netflix**, etc., for product recommendation to the user. Whenever we search for some product on Amazon, then we started getting an advertisement for the same product while internet surfing on the same browser and this is because of machine learning. Google understands the user interest using various machine learning algorithms and suggests the product as per customer interest.

As similar, when we use Netflix, we find some recommendations for entertainment series, movies, etc., and this is also done with the help of machine learning.

- **Self-driving cars:**

  One of the most exciting applications of machine learning is self-driving cars. Machine learning plays a significant role in self-driving cars. Tesla, the most popular car manufacturing company is working on self-driving car. It is using unsupervised learning method to train the car models to detect people and objects while driving.

- **Email Spam and Malware Filtering:**

  Whenever we receive a new email, it is filtered automatically as important, normal, and spam. We always receive an important mail in our inbox with the important symbol and spam emails in our spam box, and the technology behind this is Machine learning. Below are some spam filters used by Gmail:

  i. Content Filter
  ii. Header filter
  iii. General blacklists filter
  iv. Rules-based filters
  v. Permission filters

  Some machine learning algorithms such as **Multi-Layer Perceptron**, **Decision tree**, and **Naïve Bayes classifier** are used for email spam filtering and malware detection.

- **Virtual Personal Assistant:**

  We have various virtual personal assistants such as Google assistant, Alexa, Cortana, Siri. As the name suggests, they help us in finding the information using our voice instruction. These assistants can help us in various ways just by our voice instructions such as Play music, call someone, Open an email, Scheduling an appointment, etc. These virtual assistants use machine learning algorithms as an important part. These assistant record our voice instructions, send it over the server on a cloud, and decode it using ML algorithms and act accordingly.

- **Online Fraud Detection:**

  Machine learning is making our online transaction safe and secure by detecting fraud transaction. Whenever we perform some online transaction, there may be various ways that a fraudulent transaction can take place such as **fake accounts**, **fake ids**, and **steal money** in the middle of a transaction. So to detect this, **Feed Forward Neural network** helps us by checking whether it is a genuine transaction or a fraud transaction.

  For each genuine transaction, the output is converted into some hash values, and these values become the input for the next round. For each genuine transaction, there is a specific

pattern which gets change for the fraud transaction hence, it detects it and makes our online transactions more secure.

- **Stock Market trading:**

  Machine learning is widely used in stock market trading. In the stock market, there is always a risk of up and downs in shares, so for this machine learning's **long short term memory neural network** is used for the prediction of stock market trends.

- **Medical Diagnosis:**

  In medical science, machine learning is used for diseases diagnoses. With this, medical technology is growing very fast and able to build 3D models that can predict the exact position of lesions in the brain. It helps in finding brain tumors and other brain-related diseases easily.

- **Automatic Language Translation:**

  Nowadays, if we visit a new place and we are not aware of the language then it is not a problem at all, as for this also machine learning helps us by converting the text into our known languages. Google's GNMT (Google Neural Machine Translation) provide this feature, which is a Neural Machine Learning that translates the text into our familiar language, and it called as automatic translation.

  The technology behind the automatic translation is a sequence-to-sequence learning algorithm, which is used with image recognition and translates the text from one language to another language.

Types of Machine Learning

There are different ways to train machine learning algorithms, each with their own advantages and disadvantages. To understand the pros and cons of each type of machine learning, we must first look at what kind of data they ingest. In ML, there are two kinds of data — labeled data and unlabeled data.

Labeled data has both the input and output parameters in a completely machine-readable pattern, but requires a lot of human labor to label the data, to begin with. Unlabeled data only has one or none of the parameters in a machine-readable form. This negates the need for human labor but requires more complex solutions.

There are also some types of machine learning algorithms that are used in very specific use-cases, but three main methods are used today.

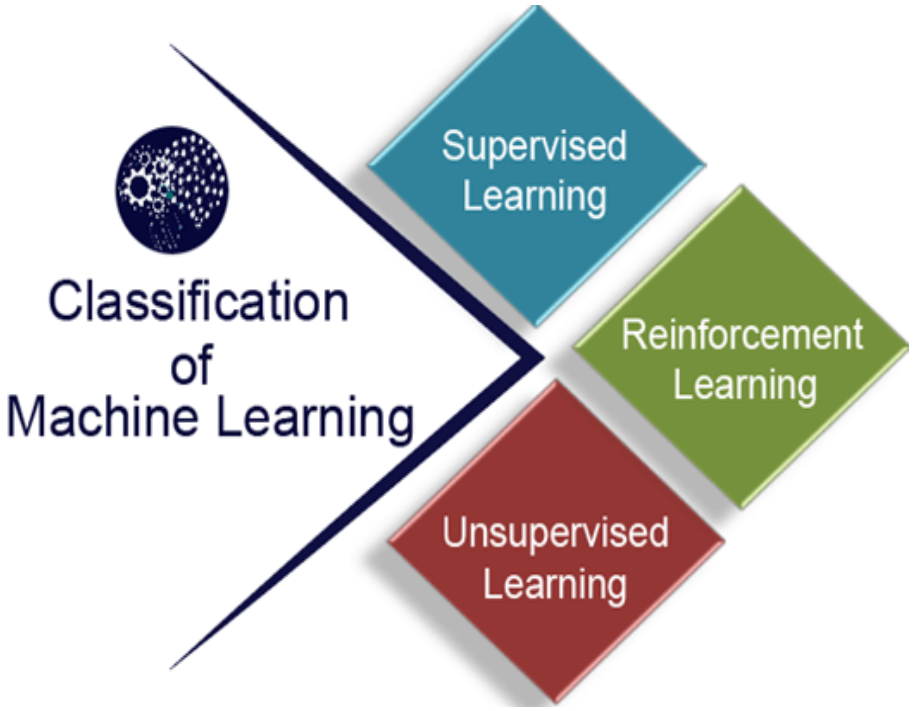


Image 105-Machine Learning Types
Reference: https://static.javatpoint.com/tutorial/machine-learning/images/classification-of-machine-learning.png

- **Supervised Learning:**

  Supervised learning is when the model is getting trained on a labelled dataset. A labelled dataset is one that has both input and output parameters. In this type of learning both training and validation, datasets are labelled as shown in the figures below.

User ID	Gender	Age	Salary	Purchased
15624510	Male	19	19000	0
15810944	Male	35	20000	1
15668575	Female	26	43000	0
15603246	Female	27	57000	0
15804002	Male	19	76000	1
15728773	Male	27	58000	1
15598044	Female	27	84000	0
15694829	Female	32	150000	1
15600575	Male	25	33000	1
15727311	Female	35	65000	0
15570769	Female	26	80000	1
15606274	Female	26	52000	0
15746139	Male	20	86000	1
15704987	Male	32	18000	0
15628972	Male	18	82000	0
15697686	Male	29	80000	0
15733883	Male	47	25000	1

Figure A: CLASSIFICATION

Temperature	Pressure	Relative Humidity	Wind Direction	Wind Speed
10.69261758	986.882019	54.19337313	195.7150879	3.278597116
13.59184184	987.8729248	48.0648859	189.2951202	2.909167767
17.70494885	988.1119385	39.11965597	192.9273834	2.973036289
20.95430404	987.8500366	30.66273218	202.0752869	2.965289593
22.9278274	987.2833862	26.06723423	210.6589203	2.798230886
24.04233986	986.2907104	23.46918024	221.1188507	2.627005816
24.41475295	985.2338867	22.25082295	233.7911987	2.448749781
23.93361956	984.8914795	22.35178837	244.3504333	2.454271793
22.68800023	984.8461304	23.7538641	253.0864716	2.418341875
20.56425726	984.8380737	27.07867944	264.5071106	2.318677425
17.76400389	985.4262085	33.54900114	280.7827454	2.343950987
11.25680746	988.9386597	53.74139903	68.15406036	1.650191426
14.37810685	989.6819458	40.70884681	72.62069702	1.553469896
18.45114201	990.2960205	30.85038484	71.70604706	1.005017161
22.54895853	989.9562988	22.81738811	44.66042709	0.264133632
24.23155922	988.796875	19.74790765	318.3214111	0.329656571

Figure B: REGRESSION

Image 106 -Supervised Learning Approach
Reference: https://media.geeksforgeeks.org/wp-content/uploads/supervised-data.png

Figure A: It is a dataset of a shopping store that is useful in predicting whether a customer will purchase a particular product under consideration or not based on his/ her gender, age, and salary.
Input: Gender, Age and Salary
Output: Purchased i.e., 0 or 1; 1 means yes, the customer will purchase and 0 means that the customer won't purchase it.

Figure B: It is a Meteorological dataset that serves the purpose of predicting wind speed based on different parameters.
Input: Dew Point, Temperature, Pressure, Relative Humidity, Wind Direction
Output: Wind Speed

- **Unsupervised Learning**

Unsupervised learning is a learning method in which a machine learns without any supervision. The training is provided to the machine with the set of data that has not been labeled, classified, or categorized, and the algorithm needs to act on that data without any supervision. The goal of unsupervised learning is to restructure the input data into new features or a group of objects with similar patterns. In unsupervised learning, we don't have a predetermined result. The machine tries to find useful insights from the huge amount of data. It can be further classifieds into two categories of algorithms:

i) Clustering
ii) Association

- **Reinforcement Learning**

Reinforcement learning is a feedback-based learning method, in which a learning agent gets a reward for each right action and gets a penalty for each wrong action. The agent learns automatically with these feedbacks and improves its performance. In reinforcement learning, the agent interacts with the environment and explores it. The goal of an agent is to get the most reward points, and hence, it improves its performance.

The robotic dog, which automatically learns the movement of his arms, is an example of Reinforcement learning.

Types of Supervised Learning:

- **Classification:** It is a Supervised Learning task where output is having defined labels (discrete value). For example, in above Figure A, Output – Purchased has defined labels

i.e., 0 or 1; 1 means the customer will purchase and 0 means that customer won't purchase. The goal here is to predict discrete values belonging to a particular class and evaluate them on the basis of accuracy. It can be either binary or multi-class classification. In binary classification, the model predicts either 0 or 1; yes or no but in the case of multi-class classification, the model predicts more than one class. Example: Gmail classifies mails in more than one class like social, promotions, updates, forums.

Classification is a process of finding a function which helps in dividing the dataset into classes based on different parameters. In Classification, a computer program is trained on the training dataset and based on that training, it categorizes the data into different classes. The task of the classification algorithm is to find the mapping function to map the input(x) to the discrete output(y).

Example: The best example to understand the Classification problem is Email Spam Detection. The model is trained on the basis of millions of emails on different parameters, and whenever it receives a new email, it identifies whether the email is spam or not. If the email is spam, then it is moved to the Spam folder.

Classification Algorithms can be further divided into the following types:

- Logistic Regression
- K-Nearest Neighbours
- Support Vector Machines
- Kernel SVM
- Naïve Bayes
- Decision Tree Classification
- Random Forest Classification

- **Regression:** It is a Supervised Learning task where output is having continuous value. Example in above Figure B, Output – Wind Speed is not having any discrete value but is continuous in the particular range. The goal here is to predict a value as much closer to the actual output value as our model can and then evaluation is done by calculating the error value. The smaller the error the greater the accuracy of our regression model.

  Regression is a process of finding the correlations between dependent and independent variables. It helps in predicting the continuous variables such as prediction of **Market Trends**, prediction of House prices, etc. The task of the Regression algorithm is to find the mapping function to map the input variable(x) to the continuous output variable(y).

Example: Suppose we want to do weather forecasting, so for this, we will use the Regression algorithm. In weather prediction, the model is trained on the past data, and once the training is completed, it can easily predict the weather for future days.

Regression vs. Classification in Machine Learning

Regression and Classification algorithms are Supervised Learning algorithms. Both the algorithms are used for prediction in Machine learning and work with the labeled datasets. But the difference between both is how they are used for different machine learning problems.

The main difference between Regression and Classification algorithms that Regression algorithms are used to **predict the continuous** values such as price, salary, age, etc. and Classification algorithms are used to **predict/Classify the discrete values** such as Male or Female, True or False, Spam or Not Spam, etc.

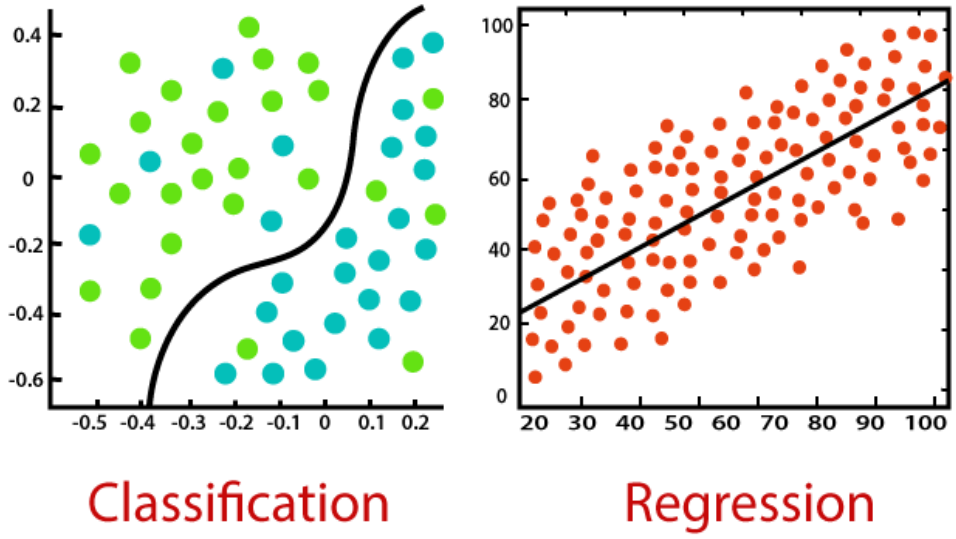


Image 107-Regression vs. Classification
Reference: https://static.javatpoint.com/tutorial/machine-learning/images/regression-vs-classification-in-machine-learning.png

Regression Algorithm	Classification Algorithm
In Regression, the output variable must be of continuous nature or real value.	In Classification, the output variable must be a discrete value.
The task of the regression algorithm is to map the input value (x) with the continuous output variable(y).	The task of the classification algorithm is to map the input value(x) with the discrete output variable(y).

Regression Algorithms are used with continuous data.	Classification Algorithms are used with discrete data.
In Regression, we try to find the best fit line, which can predict the output more accurately.	In Classification, we try to find the decision boundary, which can divide the dataset into different classes.
Regression algorithms can be used to solve the regression problems such as Weather Prediction, House price prediction, etc.	Classification Algorithms can be used to solve classification problems such as Identification of spam emails, Speech Recognition, Identification of cancer cells, etc.
The regression Algorithm can be further divided into Linear and Non-linear Regression.	The Classification algorithms can be divided into Binary Classifier and Multi-class Classifier.

Understanding Regression and types

Regression analysis is a statistical method to model the relationship between a dependent (target) and independent (predictor) variables with one or more independent variables. More specifically, Regression analysis helps us to understand how the value of the dependent variable is changing corresponding to an independent variable when other independent variables are held fixed. It predicts continuous/real values such as **temperature, age, salary, price,** etc.

Example: Suppose there is a marketing company A, who does various advertisement every year and get sales on that. The below list shows the advertisement made by the company in the last 5 years and the corresponding sales:

Advertisement	Sales
$90	$1000
$120	$1300
$150	$1800
$100	$1200
$130	$1380
$200	??

Image 108-Regression Analysis
Reference: https://static.javatpoint.com/tutorial/machine-learning/images/regression-analysis-in-machine-learning.png

Now, the company wants to do the advertisement of $200 in the year 2019 **and wants to know the prediction about the sales for this year**. So, to solve such type of prediction problems in machine learning, we need regression analysis.

Regression is a supervised learning technique which helps in finding the correlation between variables and enables us to predict the continuous output variable based on the one or more predictor variables. It is mainly used for prediction, forecasting, time series modeling, and determining the causal-effect relationship between variables.

In Regression, we plot a graph between the variables which best fits the given data points, using this plot, the machine learning model can make predictions about the data. In simple words, "Regression shows a line or curve that passes through all the datapoints on target-predictor graph in such a way that the vertical distance between the data points and the regression line is minimum." The distance between data points and line tells whether a model has captured a strong relationship or not.

Regression analysis helps in the prediction of a continuous variable. There are various scenarios in the real world where we need some future predictions such as weather condition, sales prediction, marketing trends, etc., for such case we need some technology which can make predictions more accurately. So, for such case we need Regression analysis which is a statistical method and used in machine learning and data science. Below are some other reasons for using Regression analysis:

- Regression estimates the relationship between the target and the independent variable.
- It is used to find the trends in data.
- It helps to predict real/continuous values.
- By performing the regression, we can confidently determine the most important factor, the least important factor, and how each factor is affecting the other factors.

Types of Regression

There are various types of regressions which are used in data science and machine learning. Each type has its own importance on different scenarios, but at the core, all the regression methods analyze the effect of the independent variable on dependent variables. Here we are discussing some important types of regression which are given below:

- Linear Regression
- Logistic Regression
- Polynomial Regression
- Support Vector Regression
- Decision Tree Regression

- Random Forest Regression
- Ridge Regression
- Lasso Regression:

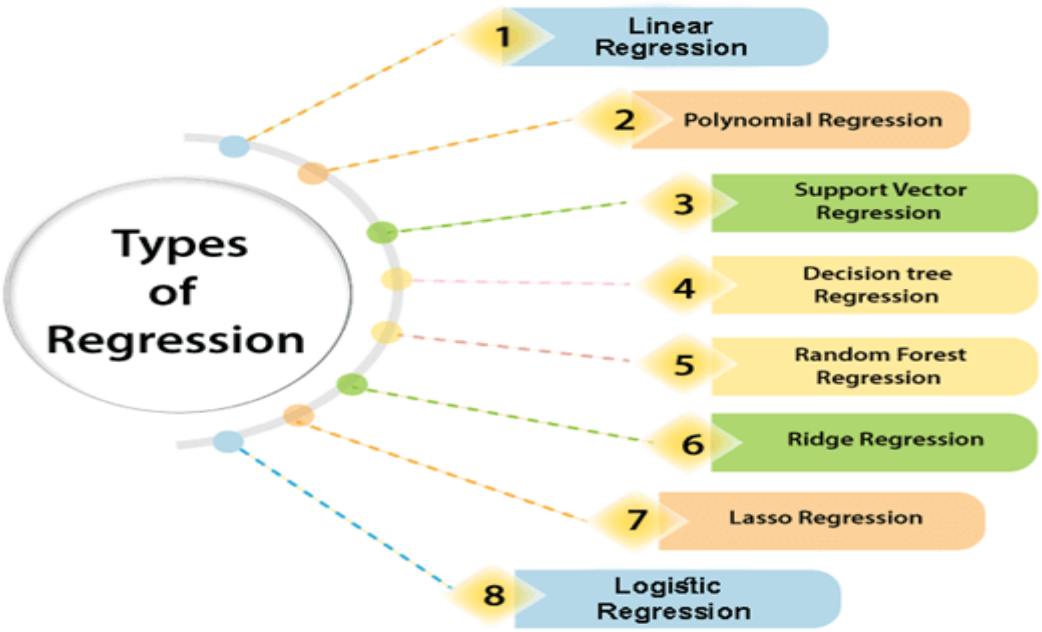


Image 109-Types of Regression
Reference: https://static.javatpoint.com/tutorial/machine-learning/images/types-of-regression.png

Linear Regression:

- Linear regression is a statistical regression method which is used for predictive analysis.
- It is one of the very simple and easy algorithms which works on regression and shows the relationship between the continuous variables.
- It is used for solving the regression problem in machine learning.
- Linear regression shows the linear relationship between the independent variable (X-axis) and the dependent variable (Y-axis), hence called linear regression.
- If there is only one input variable (x), then such linear regression is called simple linear regression. And if there is more than one input variable, then such linear regression is called multiple linear regression.

- The relationship between variables in the linear regression model can be explained using the below image. Here we are predicting the salary of an employee on the basis of the year of experience.

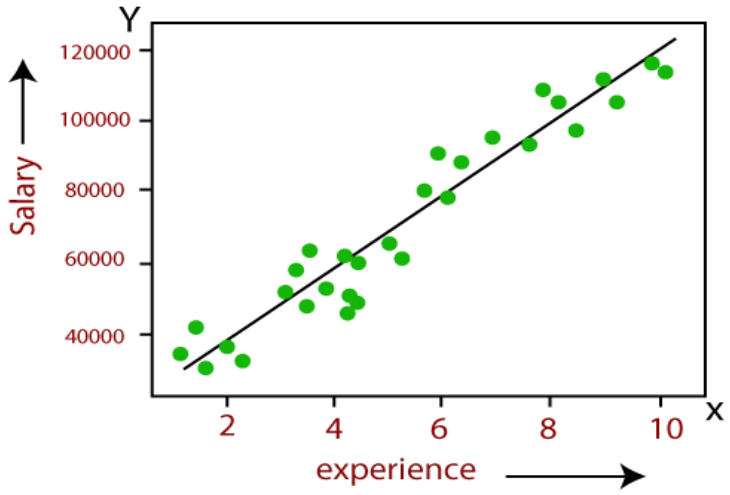


Image 200 - Dependent (Salary) and Independent (Experience) Variable
Reference: https://static.javatpoint.com/tutorial/machine-learning/images/types-of-regression2.png

The mathematical equation for Linear regression:

$$\mathbf{Y= aX+b}$$

Here, Y = dependent variables (target variables), X= Independent variables (predictor variables), a and b are the linear coefficients

Linear regression using Ordinary Least Squares (OLS)

Ordinary least squares, or linear least squares, estimates the parameters in a regression model by minimizing the sum of the squared residuals. This method draws a line through the data points that minimizes the sum of the squared differences between the observed values and the corresponding fitted values.

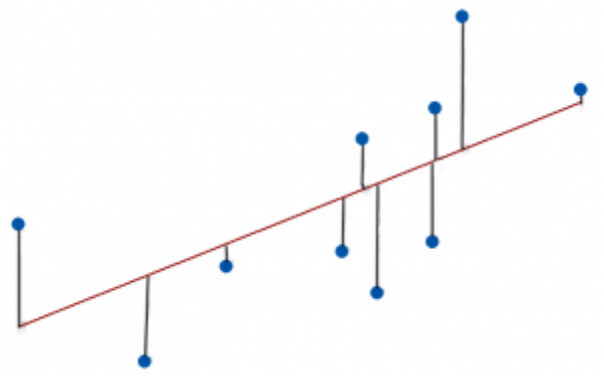


Image 201 -Ordinary least squares line
Reference: https://statisticsbyjim.com/glossary/ordinary-least-squares/#:~:text=Ordinary%20least%20squares%2C%20or%20linear,and%20the%20corresponding%20fitted%20values

- The Ordinary Least Squares procedure seeks to minimize the sum of the squared residuals. This means that given a regression line through the data we calculate the distance from each data point to the regression line, square it, and sum all of the squared errors together. This is the quantity that ordinary least squares seeks to minimize.
- This approach treats the data as a matrix and uses linear algebra operations to estimate the optimal values for the coefficients. It means that all of the data must be available and you must have enough memory to fit the data and perform matrix operations.
- Ordinary Least Squares is a form of statistical regression used as a way to predict unknown values from an existing set of data. An example of a scenario in which one may use Ordinary Least Squares, or OLS, is in predicting shoe size from a data set that includes height and shoe size. Given the data, one can use the ordinary least squares formula to create a rate of change and predict shoe size, given a subject's height. In short, OLS takes an input, the independent variable, and produces an output, the dependent variable.

OLS method equation:

$$m = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sum(x_i - \bar{x})^2}$$
$$b = \bar{y} - m * \bar{x}$$

$x = independent\ variables$
$\bar{x} = average\ of independent\ variables$
$y = dependent\ variables$
$\bar{y} = average\ of dependent\ variables$

Ordinary Least Squares method works for both univariate dataset which means single independent variables and single dependent variables and multi-variate dataset which contains a single

independent variable set and multiple dependent variables sets. Ordinary Least Squares method requires a machine learning algorithm called “Gradient Descent”.

Multi-Variate Linear Regression

Multiple Linear Regression is one of the important regression algorithms which models the linear relationship between a single dependent continuous variable and more than one independent variable.

- For MLR, the dependent or target variable (Y) must be the continuous/real, but the predictor or independent variable may be of continuous or categorical form.
- Each feature variable must model the linear relationship with the dependent variable.
- MLR tries to fit a regression line through a multidimensional space of data-points.

$$y_i=\beta_0+\beta_1 x_{i1} + \beta_2 x_{i2}+...+\beta_p x_{ip}+\epsilon$$

where, for i=n observations:

y_i=dependent variable

x_i=explanatory variables

β_0=y-intercept (constant term)

β_p=slope coefficients for each explanatory variable

ϵ=the model’s error term (also known as the residuals)

Correlation concepts

A correlation is a statistical measure of the relationship between two variables. The measure is best used in variables that demonstrate a linear relationship between each other. The fit of the data can be visually represented in a scatter plot. Using a scatter plot, we can generally assess the relationship between the variables and determine whether they are correlated or not.

The correlation coefficient is a value that indicates the strength of the relationship between variables. The coefficient can take any values from -1 to 1. The interpretations of the values are:

- -1: Perfect negative correlation. The variables tend to move in opposite directions (i.e., when one variable increases, the other variable decreases).
- 0: No correlation. The variables do not have a relationship with each other.
- 1: Perfect positive correlation. The variables tend to move in the same direction (i.e., when one variable increases, the other variable also increases).

The correlation coefficient that indicates the strength of the relationship between two variables can be found using the following formula:

$$r_{xy} = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum(x_i - \bar{x})^2 \sum(y_i - \bar{y})^2}}$$

Image 202 - correlation coefficient
Reference: https://cdn.corporatefinanceinstitute.com/assets/correlation1.png

Where:

- r_{xy} – the correlation coefficient of the linear relationship between the variables x and y
- x_i – the values of the x-variable in a sample
- $\bar{x}$ – the mean of the values of the x-variable
- y_i – the values of the y-variable in a sample
- $\bar{y}$ – the mean of the values of the y-variable

Example:

John is an investor. His portfolio primarily tracks the performance of the S&P 500 and John wants to add the stock of Apple Inc. Before adding Apple to his portfolio, he wants to assess the correlation between the stock and the S&P 500 to ensure that adding the stock won't increase the systematic risk of his portfolio.

To find the coefficient, John gathers the following prices for the last five years (**Step 1**):

	S&P 500	Apple
2013	1691.75	68.96
2014	1977.80	100.11
2015	1884.09	109.06
2016	2151.13	112.18
2017	2519.36	154.12

Using the formula above, John can determine the correlation between the prices of the S&P 500 Index and Apple Inc. First, John calculates the average prices of each security for the given periods (**Step 2**):

	S&P 500	Apple
2013	1691.75	68.96
2014	1977.80	100.11
2015	1884.09	109.06
2016	2151.13	112.18
2017	2519.36	154.12
Mean	2044.83	108.89

After the calculation of the average prices, we can find the other values. A summary of the calculations are given in the table below:

			Step 3		Step 4	Step 5	
	S&P 500	Apple	a	b	a x b	a^2	b^2
2013	1691.75	68.96	- 353.08	- 39.93	14,096.91	124,662.66	1,594.09
2014	1977.80	100.11	- 67.03	- 8.78	588.22	4,492.48	77.02
2015	1884.09	109.06	- 160.74	0.17	- 27.97	25,836.07	0.03
2016	2151.13	112.18	106.30	3.29	350.16	11,300.52	10.85
2017	2519.36	154.12	474.53	45.23	21,465.08	225,182.62	2,046.11
Mean	2044.83	108.89	Sums		36,472.40	391,474.35	3,728.10

Using the obtained numbers, John can calculate the coefficient:

$$r_{xy} = \frac{36{,}272.40}{\sqrt{391{,}474.35 \times 3{,}728.10}} = 0.95$$

The coefficient indicates that the prices of the S&P 500 and Apple Inc. have a high positive correlation. This means that their respective prices tend to move in the same direction. Therefore, adding Apple to his portfolio would, in fact, increase the level of systematic risk.

Metrics

Regression models are another family of machine learning and statistical models, which are used to predict a continuous target values[3]. They have a wide range of applications, from house price prediction, E-commerce pricing systems, weather forecasting, stock market prediction, to image super resolution, feature learning via auto-encoders, and image compression. Models such as linear regression, random forest, XGboost, convolutional neural network, recurrent neural network are some of the most popular regression models. Metrics used to evaluate these models should be able to work on a set of continuous values (with infinite cardinality), and are therefore slightly different from classification metrics. A function that calculates loss for 1 data point is called the loss function.

$$\text{Squared Error} = (y_i - \hat{y}_i)^2$$

- **Mean Squared Error (MSE) / Mean Squared Deviation (MSD)**

  The Mean Squared Error measures the average of the errors squared. It basically calculates the difference between the estimated and the actual value, squares these results and then computes their average. Because the errors are squared, MSE can only

assume non-negative values. Due to the intrinsic randomness and noise associated with most processes, MSE is usually positive and not zero.

Let's assume we have a regression model which predicts the price of houses in Seattle area (show them with $\hat{y}_i$), and let's say for each house we also have the actual price the house was sold for (denoted with y_i). Then the MSE can be calculated as:

$$\mathbf{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - y'_i)^2$$

Like the variance, MSE has the same units of measurement as the square of the quantity being estimated. Similarly, to the Variance, one major disadvantage of Mean Squared Error is that it is not robust to outliers. In case a sample has a "y" and associated error which is way larger than the other samples, the square of the error will be even larger. This, paired to the fact that MSE calculates the average of errors, makes MSE prone to outliers.

- **Root Mean Squared Error (RMSE) / Root Mean Squared Deviation (RMSD)**

Similarly, to the Mean Squared Error, RMSE calculates the average of the squared errors across all samples but, in addition, takes the square root of the result, effectively taking the square root of MSE. By doing so, RMSE provides an error measure in the same unit as the target variable. For instance, if our target y is next year's sales in dollars, RMSE will give the error in dollars, while MSE would be in dollars squared, which is much less interpretable.

$$\mathbf{RMSD} = \sqrt{\frac{\sum_{i=1}^{N}(x_i - x'_i)^2}{N}}$$

- **Mean Absolute Error (MAE)**

The Mean Absolute Error does not take the square of the errors. Instead, it simply calculates the absolute value of the errors and then takes the average of these values. The MAE takes the absolute value as we are not interested in the direction in which the estimated and actual target values differ (estimated > actual or vice-versa) but on the absolute distance. This also avoids errors to cancel each other out when calculating the MAE.

$$\mathbf{MAE} = \frac{\sum_{i=1}^{n}|y_i - x_i|}{n}$$

Differently from MSE, MAE does not penalize larger errors more than smaller ones, because the formula for MAE does not apply the square to errors. Another advantage is that MAE does not square the units, similarly to RMSE, making the results more interpretable.

- **R Squared (R^2) / Coefficient of Determination**

  R Squared (R^2) represents the proportion of the variance for the dependent variable y that's explained by the independent variables X. R^2 explains to what extent the variance of one variable explains the variance of the second variable. So, if the R^2 of a model is 0.75, then approximately 75% of the observed variation can be explained by the model's features.

  R^2 is calculated by taking one minus the sum of squares of residuals divided by the total sum of squares.

$$R^2 = 1 - \frac{SS_{RES}}{SS_{TOT}} = 1 - \frac{\sum_i (y_i - y'_i)^2}{\sum_i (y_i - y^c)^2}$$

  R^2 compares the fit of the chosen model with that of a horizontal line, which acts as a baseline. If the chosen model fits worse than a horizontal line, the R^2 is negative. Because of the formula of R^2, even though the "square" is involved, it can have a negative value without violating any rules of math. R^2 is negative only when the model does not follow the trend of the data and fits worse than a horizontal line.

Residuals in Regression

When you perform simple linear regression (or any other type of regression analysis), you get a line of best fit. The data points usually don't fall exactly on this regression equation line; they are scattered around. A residual is the vertical distance between a data point and the regression line. Each data point has one residual. They are:

- Positive if they are above the regression line,
- Negative if they are below the regression line,
- Zero if the regression line actually passes through the point,

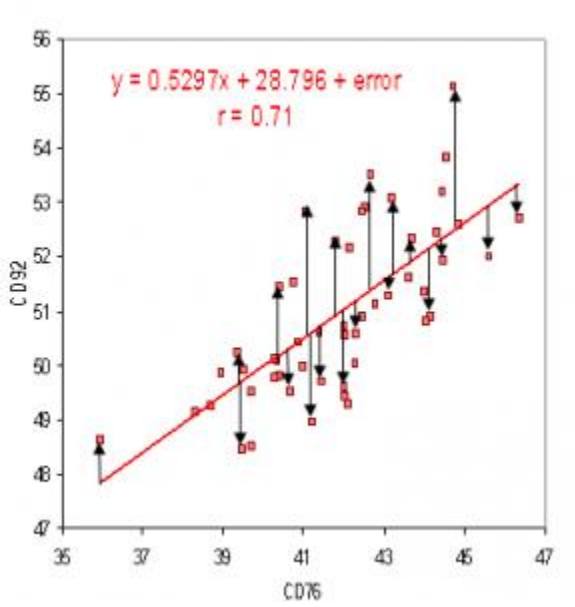


Image 203-Residuals on a scatter plot.
Reference: nws.noaa.gov

As residuals are the difference between any data point and the regression line, they are sometimes called "errors." Error in this context doesn't mean that there's something wrong with the analysis; it just means that there is some unexplained difference. In other words, the residual is the error that isn't explained by the regression line.

The residual(e) can also be expressed with an equation. The e is the difference between the predicted value ($\hat{y}$) and the observed value. The scatter plot is a set of data points that are observed, while the regression line is the prediction.

Residual = Observed value – predicted value

$$\mathbf{e = y - \hat{y}}$$

The Sum and Mean of Residuals

The sum of the residuals always equals zero (assuming that your line is actually the line of "best fit." The mean of residuals is also equal to zero, as the mean = the sum of the residuals / the number of items. The sum is zero, so 0/n will always equal zero.

Polynomial Features

Polynomial features are those features created by raising existing features to an exponent.

For example, if a dataset had one input feature X, then a polynomial feature would be the addition of a new feature (column) where values were calculated by squaring the values in X, e.g., X^2. This process can be repeated for each input variable in the dataset, creating a transformed version of each.

As such, polynomial features are a type of feature engineering, e.g., the creation of new input features based on the existing features.
The “degree” of the polynomial is used to control the number of features added, e.g., a degree of 3 will add two new variables for each input variable. Typically, a small degree is used such as 2 or 3.

Classification techniques

What Is Classification?
Classification is the process of recognizing, understanding, and grouping ideas and objects into preset categories or “sub-populations.” Using pre-categorized training datasets, machine learning programs use a variety of algorithms to classify future datasets into categories.

Classification algorithms in machine learning use input training data to predict the likelihood that subsequent data will fall into one of the predetermined categories. One of the most common uses of classification is filtering emails into “spam” or “non-spam.”
In short, classification is a form of “pattern recognition,” with classification algorithms applied to the training data to find the same pattern (similar words or sentiments, number sequences, etc.) in future sets of data.
Popular Classification Algorithms:

- Logistic Regression
- Naive Bayes
- K-Nearest Neighbors
- Decision Tree
- Support Vector Machines

Logistic Regression

Logistic regression is a calculation used to predict a binary outcome: either something happens, or does not. This can be exhibited as Yes/No, Pass/Fail, Alive/Dead, etc.
Independent variables are analysed to determine the binary outcome with the results falling into one of two categories. The independent variables can be categorical or numeric, but the dependent variable is always categorical. Written like this:

$$P(Y=1|X) \text{ or } P(Y=0|X)$$

It calculates the probability of dependent variable Y, given independent variable X.
This can be used to calculate the probability of a word having a positive or negative connotation (0, 1, or on a scale between). Or it can be used to determine the object contained in a photo (tree, flower, grass, etc.), with each object given a probability between 0 and 1.

Naive Bayes

Naive Bayes calculates the possibility of whether a data point belongs within a certain category or does not. In text analysis, it can be used to categorize words or phrases as belonging to a preset "tag" (classification) or not. For example:

Text	Tag
"A great game"	Sports
"The election was over"	Not sports
"Very clean match"	Sports
"A clean but forgettable game"	Sports
"It was a close election"	Not sports

To decide whether or not a phrase should be tagged as "sports," you need to calculate:

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

Naive Bayes formula.
Or… the probability of A, if B is true, is equal to the probability of B, if A is true, times the probability of A being true, divided by the probability of B being true.

K-nearest Neighbors

K-nearest neighbors (k-NN) is a pattern recognition algorithm that uses training datasets to find the k closest relatives.

When k-NN is used in classification, you calculate to place data within the category of its nearest neighbor. If k = 1, then it would be placed in the class nearest 1. K is classified by a plurality poll of its neighbors.

Decision Tree

A decision tree is a supervised learning algorithm that is perfect for classification problems, as it's able to order classes on a precise level. It works like a flow chart, separating data points into two similar categories at a time from the "tree trunk" to "branches," to "leaves," where the categories become more finitely similar. This creates categories within categories, allowing for organic classification with limited human supervision.
To continue with the sports example, this is how the decision tree works:

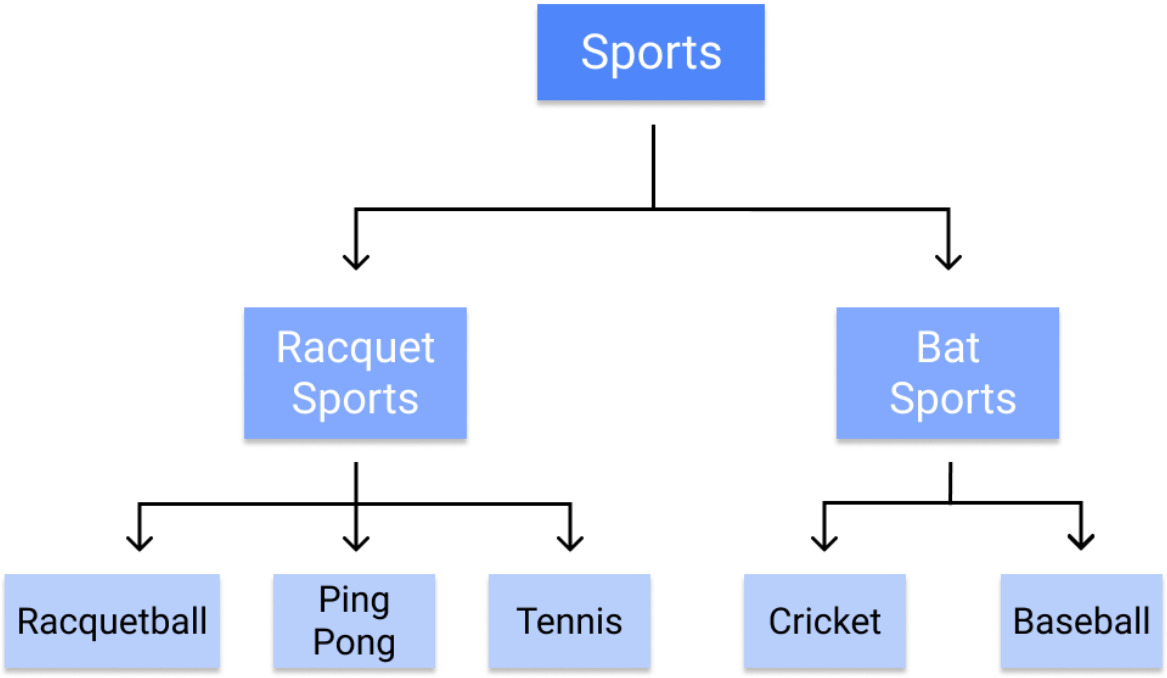


Image 204-Working of a Decision Tree.
Reference: nws.noaa.gov

Random Forest

The random forest algorithm is an expansion of decision tree, in that you first construct a multitude of decision trees with training data, then fit your new data within one of the trees as a "random forest."
It, essentially, averages your data to connect it to the nearest tree on the data scale. Random forest models are helpful as they remedy for the decision tree's problem of "forcing" data points within a category unnecessarily.

Support Vector Machines

A support vector machine (SVM) uses algorithms to train and classify data within degrees of polarity, taking it to a degree beyond X/Y prediction.

For a simple visual explanation, we'll use two tags: red and blue, with two data features: X and Y, then train our classifier to output an X/Y coordinate as either red or blue.

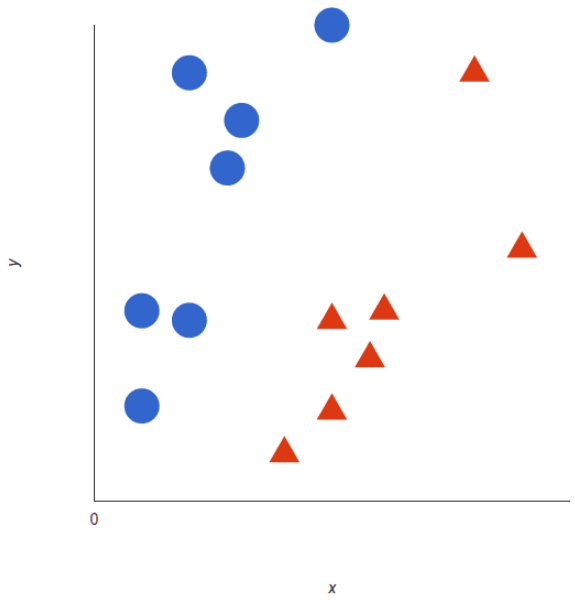


Image 205-Working of a Decision Tree.
Reference: https://monkeylearn.com/blog/classification-algorithms/

The SVM then assigns a hyperplane that best separates the tags. In two dimensions this is simply a line. Anything on one side of the line is red and anything on the other side is blue. In sentiment analysis, for example, this would be positive and negative.

In order to maximize machine learning, the best hyperplane is the one with the largest distance between each tag:

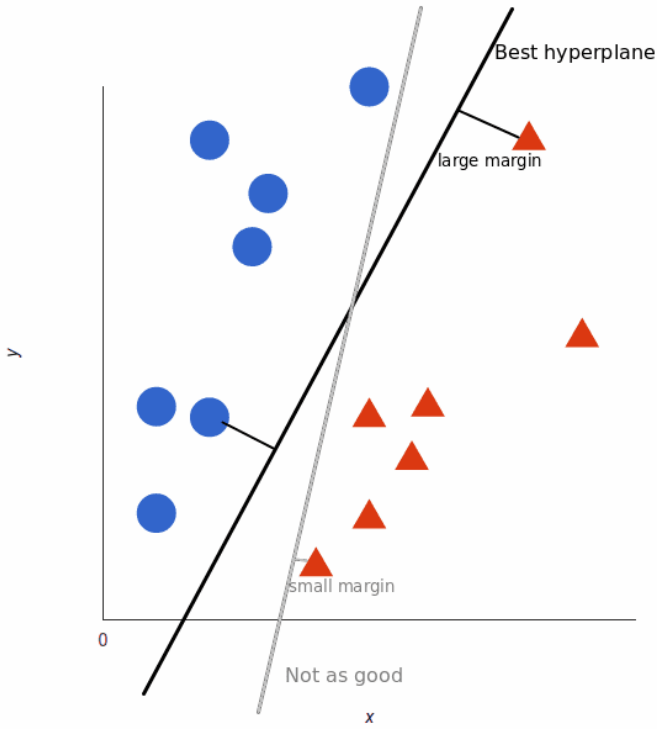


Image 206-SVM
Reference: https://monkeylearn.com/blog/classification-algorithms/

Using SVM, the more complex the data, the more accurate the predictor will become.

Types of distance metrics

Distance measures play an important role in machine learning.
They provide the foundation for many popular and effective machine learning algorithms like k-nearest neighbors for supervised learning and k-means clustering for unsupervised learning.
Different distance measures must be chosen and used depending on the types of the data. As such, it is important to know how to implement and calculate a range of different popular distance measures and the intuitions for the resulting scores.

Role of Distance Measures

A distance measure is an objective score that summarizes the relative difference between two objects in a problem domain.
Most commonly, the two objects are rows of data that describe a subject (such as a person, car, or house), or an event (such as a purchase, a claim, or a diagnosis).

Perhaps the most likely way you will encounter distance measures is when you are using a specific machine learning algorithm that uses distance measures at its core. The most famous algorithm of this type is the k-nearest neighbors algorithm, or KNN.

Following are the 4 most commonly used distance measures in machine learning:

1. Hamming Distance
2. Euclidean Distance
3. Manhattan Distance
4. Minkowski Distance

Hamming Distance

Hamming distance calculates the distance between two binary vectors, also referred to as binary strings or bitstrings for short.
You are most likely going to encounter bitstrings when you one-hot encode categorical columns of data.
For example, if a column had the categories ‘red,’ ‘green,’ and ‘blue,’ you might one hot encode each example as a bitstring with one bit for each column.
red = [1, 0, 0]
green = [0, 1, 0]
blue = [0, 0, 1]
The distance between red and green could be calculated as the sum or the average number of bit differences between the two bitstrings. This is the Hamming distance.

Euclidean Distance

Euclidean distance calculates the distance between two real-valued vectors.
You are most likely to use Euclidean distance when calculating the distance between two rows of data that have numerical values, such a floating point or integer values.
If columns have values with differing scales, it is common to normalize or standardize the numerical values across all columns prior to calculating the Euclidean distance. Otherwise, columns that have large values will dominate the distance measure.
Euclidean distance is calculated as the square root of the sum of the squared differences between the two vectors.

EuclideanDistance = sqrt(sum for i to N (v1[i] – v2[i])^2)

If the distance calculation is to be performed thousands or millions of times, it is common to remove the square root operation in an effort to speed up the calculation. The resulting scores will have the same relative proportions after this modification and can still be used effectively within a machine learning algorithm for finding the most similar examples.

EuclideanDistance = sum for i to N (v1[i] – v2[i])^2

Manhattan Distance (Taxicab or City Block Distance)

The Manhattan distance, also called the Taxicab distance or the City Block distance, calculates the distance between two real-valued vectors.
It is perhaps more useful to vectors that describe objects on a uniform grid, like a chessboard or city blocks. The taxicab name for the measure refers to the intuition for what the measure calculates: the shortest path that a taxicab would take between city blocks (coordinates on the grid).
It might make sense to calculate Manhattan distance instead of Euclidean distance for two vectors in an integer feature space.
Manhattan distance is calculated as the sum of the absolute differences between the two vectors.

ManhattanDistance = sum for i to N sum |v1[i] – v2[i]

Minkowski Distance

Minkowski distance calculates the distance between two real-valued vectors.
It is a generalization of the Euclidean and Manhattan distance measures and adds a parameter, called the "order" or "p", that allows different distance measures to be calculated.
The Minkowski distance measure is calculated as follows:
EuclideanDistance = (sum for i to N (abs(v1[i] – v2[i]))^p)^(1/p)
Where "p" is the order parameter.
When p is set to 1, the calculation is the same as the Manhattan distance. When p is set to 2, it is the same as the Euclidean distance.
p=1: Manhattan distance.
p=2: Euclidean distance.
Intermediate values provide a controlled balance between the two measures.
It is common to use Minkowski distance when implementing a machine learning algorithm that uses distance measures as it gives control over the type of distance measure used for real-valued vectors via a hyperparameter "p" that can be tuned.

KNN Classification

K-Nearest Neighbour is one of the simplest Machine Learning algorithms based on Supervised Learning technique.
K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.
K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm.
K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for the Classification problems.

K-NN is a non-parametric algorithm, which means it does not make any assumption on underlying data.
It is also called a lazy learner algorithm because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset.
KNN algorithm at the training phase just stores the dataset and when it gets new data, then it classifies that data into a category that is much similar to the new data.
Example: Suppose, we have an image of a creature that looks similar to cat and dog, but we want to know either it is a cat or dog. So for this identification, we can use the KNN algorithm, as it works on a similarity measure. Our KNN model will find the similar features of the new data set to the cats and dogs images and based on the most similar features it will put it in either cat or dog category.

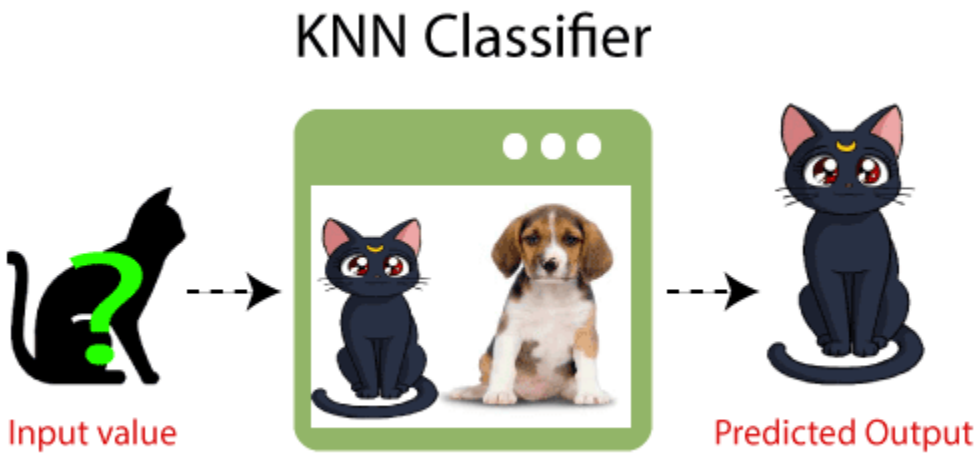


Image 207-KNN Classifier
Reference: https://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning

Why do we need a K-NN Algorithm?

Suppose there are two categories, i.e., Category A and Category B, and we have a new data point x1, so this data point will lie in which of these categories. To solve this type of problem, we need a K-NN algorithm. With the help of K-NN, we can easily identify the category or class of a particular dataset. Consider the below diagram:

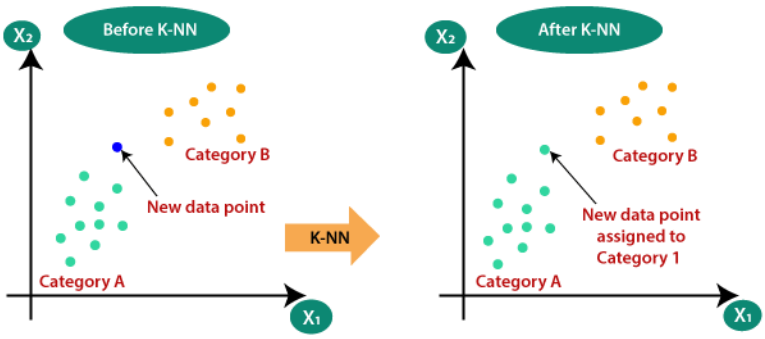


Image 208-KNN Classifier
Reference: https://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning

How does K-NN work?

The K-NN working can be explained on the basis of the below algorithm:
Step-1: Select the number K of the neighbors
Step-2: Calculate the Euclidean distance of K number of neighbors
Step-3: Take the K nearest neighbors as per the calculated Euclidean distance.
Step-4: Among these k neighbors, count the number of the data points in each category.
Step-5: Assign the new data points to that category for which the number of the neighbor is maximum.
Step-6: Our model is ready.
Suppose we have a new data point and we need to put it in the required category. Consider the below image:

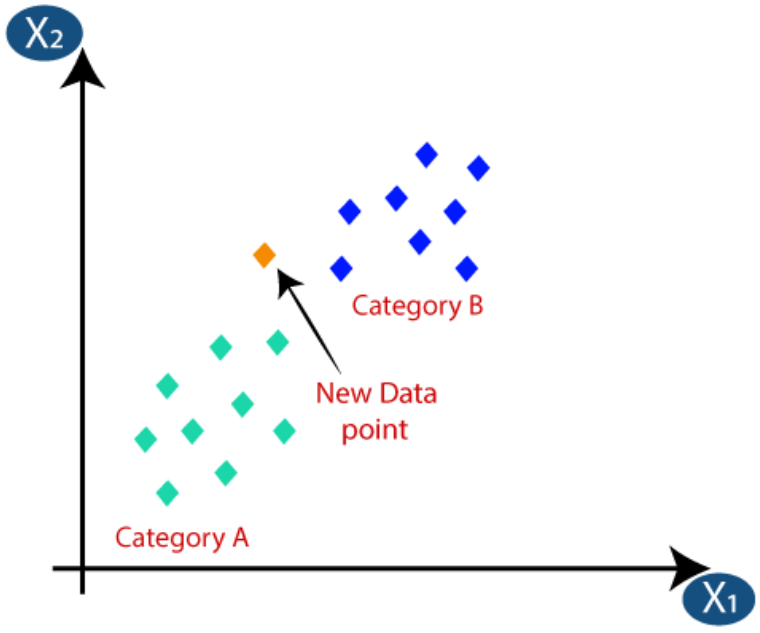


Image 209-KNN Classifier
Reference: https://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning

Firstly, we will choose the number of neighbors, so we will choose the k=5.
Next, we will calculate the Euclidean distance between the data points. The Euclidean distance is the distance between two points, which we have already studied in geometry. It can be calculated as:

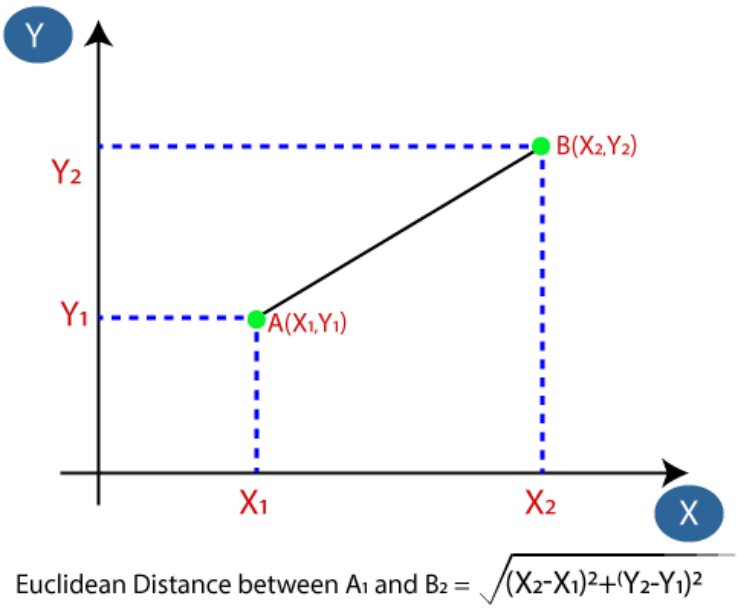


Image 210-KNN Classifier
Reference: https://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning

By calculating the Euclidean distance, we got the nearest neighbors, as three nearest neighbors in category A and two nearest neighbors in category B. Consider the below image:

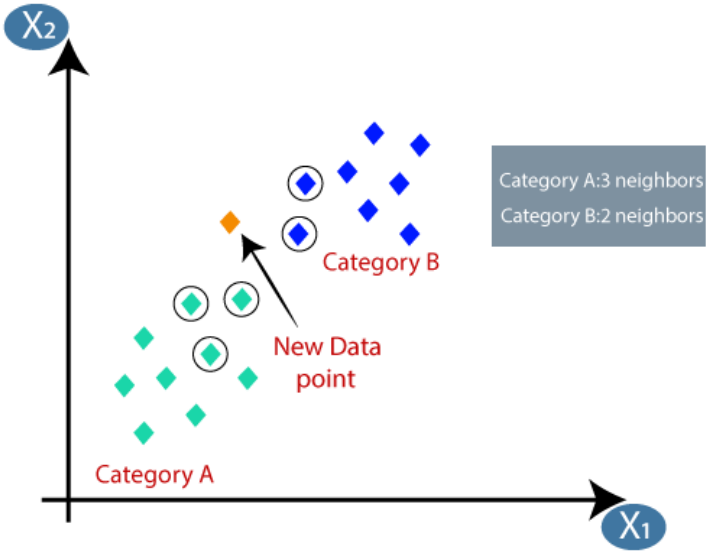


Image 211-KNN Classifier
Reference: https://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning

As we can see the 3 nearest neighbors are from category A, hence this new data point must belong to category A.

Gradient Descent

Gradient descent (GD) is an iterative first-order optimisation algorithm used to find a local minimum/maximum of a given function. This method is commonly used in machine learning (ML) and deep learning (DL) to minimise a cost/loss function (e.g. in a linear regression). Due to its importance and ease of implementation, this algorithm is usually taught at the beginning of almost all machine learning courses.
However, its use is not limited to ML/DL only, it's being widely used also in areas like:

- control engineering (robotics, chemical, etc.)
- computer games
- mechanical engineering

What is Gradient?

In the case of a univariate function, it is simply the first derivative at a selected point. In the case of a multivariate function, it is a vector of derivatives in each main direction (along variable axes). Because we are interested only in a slope along one axis and we don't care about others these derivatives are called partial derivatives.
A gradient for an n-dimensional function f(x) at a given point p is defined as follows:

$$\nabla f(p) = \begin{bmatrix} \frac{\partial f}{\partial x_1}(p) \\ \vdots \\ \frac{\partial f}{\partial x_n}(p) \end{bmatrix}$$

The upside-down triangle is a so-called nabla symbol and you read it "del". To better understand how to calculate it let's do a hand calculation for an exemplary 2-dimensional function below.

$$f(x) = 0.5x^2 + y^2$$

How does Gradient Descent work?

Before starting the working principle of gradient descent, we should know some basic concepts to find out the slope of a line from linear regression. The equation for simple linear regression is given as:

Y=mX+c

Where 'm' represents the slope of the line, and 'c' represents the intercepts on the y-axis.

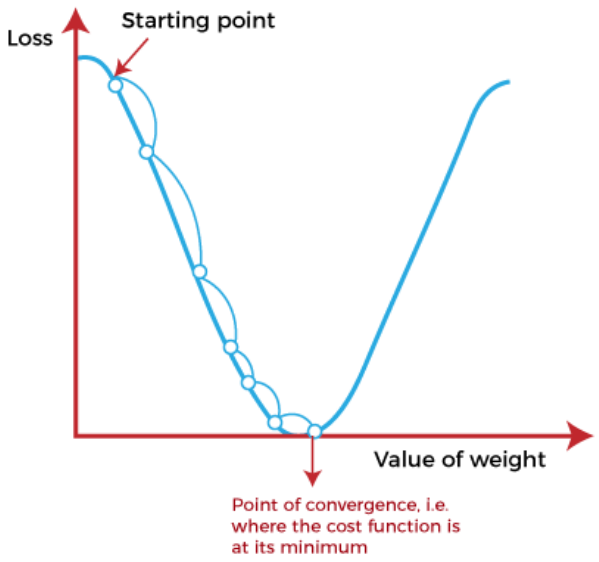


Image 212-Gradient Descent
Reference: https://www.javatpoint.com/gradient-descent-in-machine-learning

The starting point is used to evaluate the performance as it is considered just as an arbitrary point. At this starting point, we will derive the first derivative or slope and then use a tangent line to calculate the steepness of this slope. Further, this slope will inform the updates to the parameters (weights and bias).
The slope becomes steeper at the starting point or arbitrary point, but whenever new parameters are generated, then steepness gradually reduces, and at the lowest point, it approaches the lowest point, which is called a point of convergence.
The main objective of gradient descent is to minimize the cost function or the error between expected and actual. To minimize the cost function, two data points are required:

Direction & Learning Rate

These two factors are used to determine the partial derivative calculation of future iteration and allow it to the point of convergence or local minimum or global minimum. Let's discuss learning rate factors in brief;

Learning Rate:

It is defined as the step size taken to reach the minimum or lowest point. This is typically a small value that is evaluated and updated based on the behavior of the cost function. If the learning rate is high, it results in larger steps but also leads to risks of overshooting the minimum. At the same time, a low learning rate shows the small step sizes, which compromises overall efficiency but gives the advantage of more precision.

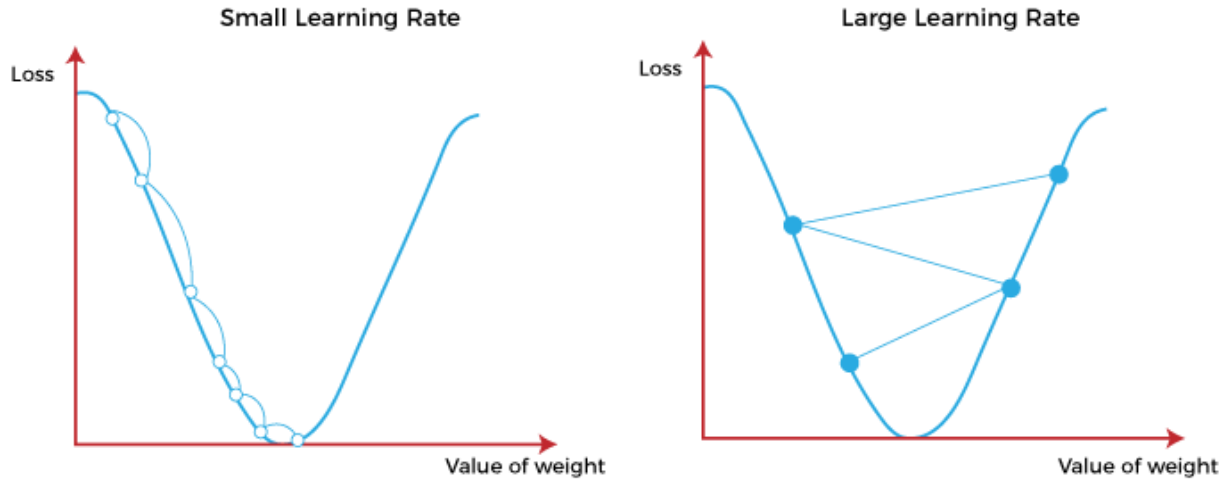


Image 213-Learning rate
Reference: https://www.javatpoint.com/gradient-descent-in-machine-learning

Types of Gradient Descent

Based on the error in various training models, the Gradient Descent learning algorithm can be divided into:

1. Batch gradient descent,
2. stochastic gradient descent, and
3. mini-batch gradient descent.

Batch Gradient Descent:

Batch gradient descent (BGD) is used to find the error for each point in the training set and update the model after evaluating all training examples. This procedure is known as the training epoch.

Advantages of Batch gradient descent:

- It produces less noise in comparison to other gradient descent.

- It produces stable gradient descent convergence.
- It is Computationally efficient as all resources are used for all training samples.

Stochastic gradient descent

Stochastic gradient descent (SGD) is a type of gradient descent that runs one training example per iteration. Or in other words, it processes a training epoch for each example within a dataset and updates each training example's parameters one at a time. As it requires only one training example at a time, hence it is easier to store in allocated memory. However, it shows some computational efficiency losses in comparison to batch gradient systems as it shows frequent updates that require more detail and speed. Further, due to frequent updates, it is also treated as a noisy gradient. However, sometimes it can be helpful in finding the global minimum and also escaping the local minimum.

Advantages of Stochastic gradient descent:

In Stochastic gradient descent (SGD), learning happens on every example, and it consists of a few advantages over other gradient descent.

- It is easier to allocate in desired memory.
- It is relatively fast to compute than batch gradient descent.
- It is more efficient for large datasets.

MiniBatch Gradient Descent:

Mini Batch gradient descent is the combination of both batch gradient descent and stochastic gradient descent. It divides the training datasets into small batch sizes then performs the updates on those batches separately. Splitting training datasets into smaller batches make a balance to maintain the computational efficiency of batch gradient descent and speed of stochastic gradient descent. Hence, we can achieve a special type of gradient descent with higher computational efficiency and less noisy gradient descent.

Advantages of Mini Batch gradient descent:

- It is easier to fit in allocated memory.
- It is computationally efficient.

- It produces stable gradient descent convergence.

Gradient Descent Algorithm

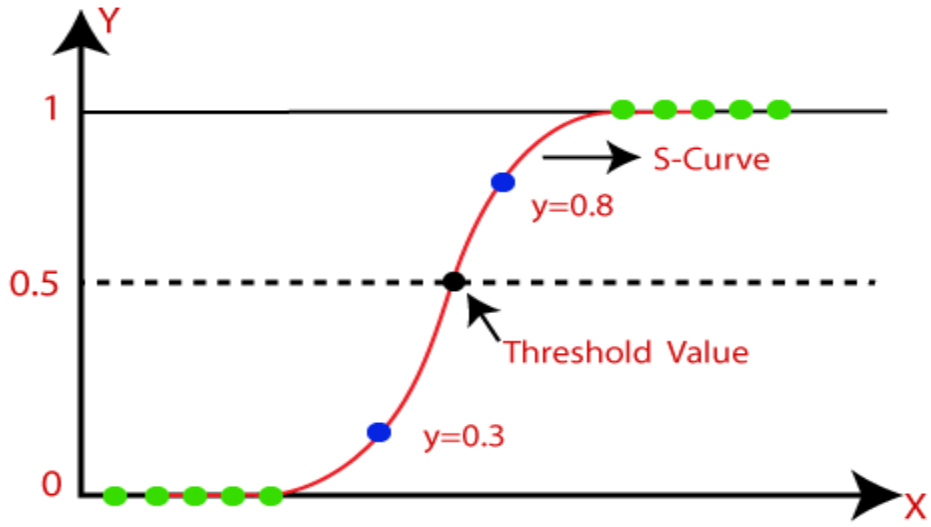
Gradient Descent Algorithm iteratively calculates the next point using gradient at the current position, then scales it (by a learning rate) and subtracts obtained value from the current position (makes a step). It subtracts the value because we want to minimise the function (to maximise it would be adding). This process can be written as:
There's an important parameter η which scales the gradient and thus controls the step size. In machine learning, it is called learning rate and have a strong influence on performance.
The smaller learning rate the longer GD converges, or may reach maximum iteration before reaching the optimum point
If learning rate is too big the algorithm may not converge to the optimal point (jump around) or even to diverge completely.
In summary, Gradient Descent method's steps are:

- choose a starting point (initialisation)
- calculate gradient at this point
- make a scaled step in the opposite direction to the gradient (objective: minimise)
- repeat points 2 and 3 until one of the criteria is met:
- maximum number of iterations reached
- step size is smaller than the tolerance.

Logistic Regression

- Logistic regression is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a given set of independent variables.
- Logistic regression predicts the output of a categorical dependent variable. Therefore the outcome must be a categorical or discrete value. It can be either Yes or No, 0 or 1, true or False, etc. but instead of giving the exact value as 0 and 1, **it gives the probabilistic values which lie between 0 and 1**.
- Logistic Regression is much similar to the Linear Regression except that how they are used. Linear Regression is used for solving Regression problems, whereas **Logistic regression is used for solving the classification problems**.
- In Logistic regression, instead of fitting a regression line, we fit an "S" shaped logistic function, which predicts two maximum values (0 or 1).
- The curve from the logistic function indicates the likelihood of something such as whether the cells are cancerous or not, a mouse is obese or not based on its weight, etc.

- Logistic Regression is a significant machine learning algorithm because it has the ability to provide probabilities and classify new data using continuous and discrete datasets.
- Logistic Regression can be used to classify the observations using different types of data and can easily determine the most effective variables used for the classification. The below image is showing the logistic function:



Ref: https://www.javatpoint.com/logistic-regression-in-machine-learning

Logistic Function (Sigmoid Function):

- The sigmoid function is a mathematical function used to map the predicted values to probabilities.
- It maps any real value into another value within a range of 0 and 1.
- The value of the logistic regression must be between 0 and 1, which cannot go beyond this limit, so it forms a curve like the "S" form. The S-form curve is called the Sigmoid function or the logistic function.
- In logistic regression, we use the concept of the threshold value, which defines the probability of either 0 or 1. Such as values above the threshold value tends to 1, and a value below the threshold values tends to 0.

Assumptions for Logistic Regression:

- The dependent variable must be categorical in nature.
- The independent variable should not have multi-collinearity.

Logistic Regression Equation:

The Logistic regression equation can be obtained from the Linear Regression equation. The mathematical steps to get Logistic Regression equations are given below:

- We know the equation of the straight line can be written as:

$$y= b_0 + b_1x_1 + b_2x_2 + b_3x_3 + \cdots + b_nx_n$$

- In Logistic Regression y can be between 0 and 1 only, so for this let's divide the above equation by (1-y):

$$\frac{y}{1-y}; \text{ 0 for y= 0, and infinity for y=1}$$

- But we need range between -[infinity] to +[infinity], then take logarithm of the equation it will become:

$$log\left[\frac{y}{1-y}\right] = b_0 + b_1x_1 + b_2x_2 + b_3x_3 + \cdots + b_nx_n$$

The above equation is the final equation for Logistic Regression.

Type of Logistic Regression:

On the basis of the categories, Logistic Regression can be classified into three types:

- Binomial: In binomial Logistic regression, there can be only two possible types of the dependent variables, such as 0 or 1, Pass or Fail, etc.
- Multinomial: In multinomial Logistic regression, there can be 3 or more possible unordered types of the dependent variable, such as "cat", "dogs", or "sheep"
- Ordinal: In ordinal Logistic regression, there can be 3 or more possible ordered types of dependent variables, such as "low", "Medium", or "High".

Python Implementation of Logistic Regression (Binomial)

To understand the implementation of Logistic Regression in Python, we will use the below example:

Example: There is a dataset given which contains the information of various users obtained from the social networking sites. There is a car making company that has recently launched a new SUV car. So, the company wanted to check how many users from the dataset, wants to purchase the car.

For this problem, we will build a Machine Learning model using the Logistic regression algorithm. The dataset is shown in the below image. In this problem, we will predict the **purchased variable (Dependent Variable)** by using **age and salary (Independent variables)**.

User ID	Gender	Age	EstimatedSalary	Purchased
15624510	Male	19	19000	0
15810944	Male	35	20000	0
15668575	Female	26	43000	0
15603246	Female	27	57000	0
15804002	Male	19	76000	0
15728773	Male	27	58000	0
15598044	Female	27	84000	0
15694829	Female	32	150000	1
15600575	Male	25	33000	0
15727311	Female	35	65000	0
15570769	Female	26	80000	0
15606274	Female	26	52000	0
15746139	Male	20	86000	0
15704987	Male	32	18000	0
15628972	Male	18	82000	0
15697686	Male	29	80000	0
15733883	Male	47	25000	1
15617482	Male	45	26000	1
15704583	Male	46	28000	1
15621083	Female	48	29000	1
15649487	Male	45	22000	1
15736760	Female	47	49000	1

Image 214-Dataset
Reference: https://www.javatpoint.com/logistic-regression-in-machine-learning

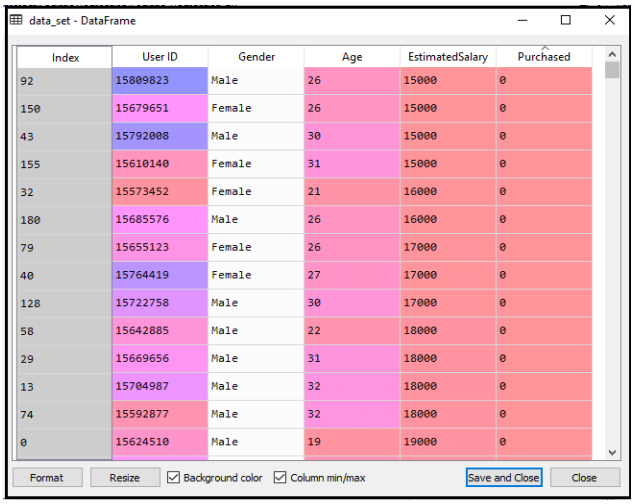
Steps in Logistic Regression: To implement the Logistic Regression using Python, we will use the same steps as we have done in previous topics of Regression. Below are the steps:

- Data Pre-processing step
- Fitting Logistic Regression to the Training set
- Predicting the test result
- Test accuracy of the result (Creation of Confusion matrix)
- Visualizing the test set result.

1. Data Pre-processing step: In this step, we will pre-process/prepare the data so that we can use it in our code efficiently. It will be the same as we have done in Data pre-processing topic. The code for this is given below:

```
#Data Pre-procesing Step
# importing libraries
import numpy as nm
import matplotlib.pyplot as mtp
import pandas as pd
#importing datasets
data_set= pd.read_csv('user_data.csv')
```

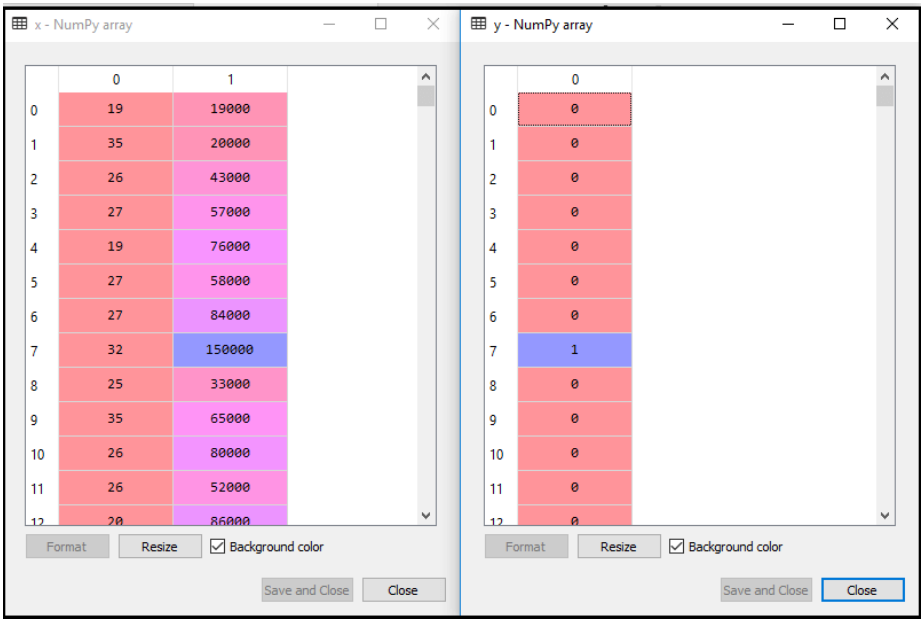
By executing the above lines of code, we will get the dataset as the output. Consider the given image:



Now, we will extract the dependent and independent variables from the given dataset. Below is the code for it:

In the above code, we have taken [2, 3] for x because our independent variables are age and salary, which are at index 2, 3. And we have taken 4 for y variable because our dependent variable is at index 4. The output will be:

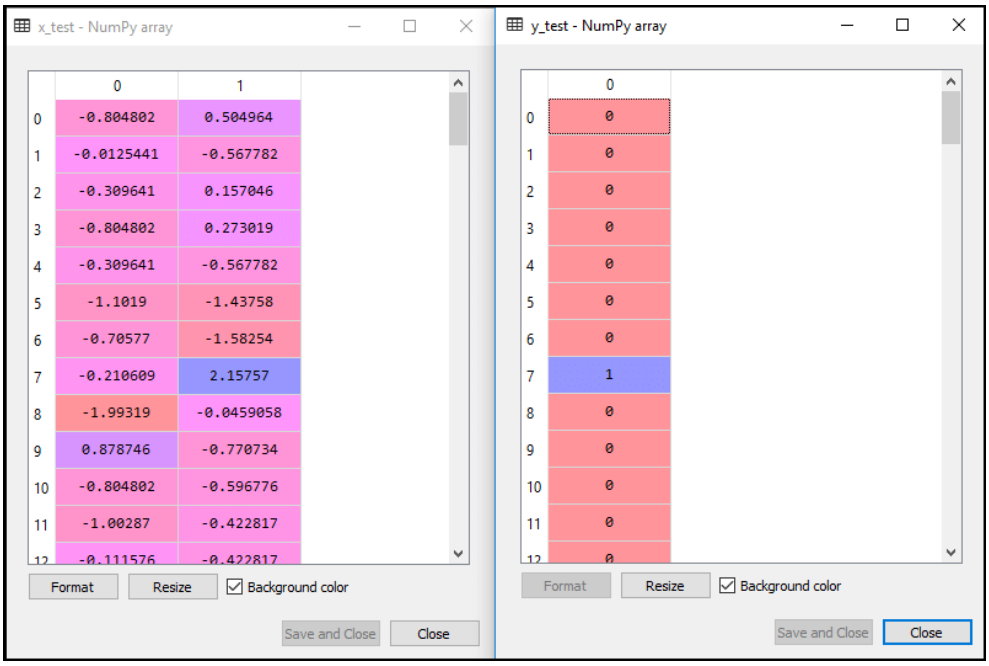
```
#Extracting Independent and dependent Variable
x= data_set.iloc[:, [2,3]].values
y= data_set.iloc[:, 4].values
```



Now we will split the dataset into a training set and test set. Below is the code for it:

```
# Splitting the dataset into training and test set.
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test= train_test_split(x, y, test_size= 0.25, random_state=0)
```

Output for test set:



In logistic regression, we will do feature scaling because we want accurate result of predictions. Here we will only scale the independent variable because dependent variable has only 0 and 1 values. Below is the code for it:

```
#feature Scaling
from sklearn.preprocessing import StandardScaler
st_x= StandardScaler()
x_train= st_x.fit_transform(x_train)
x_test= st_x.transform(x_test)
```

The scaled output is given below:

2. Fitting Logistic Regression to the Training set:

We have well prepared our dataset, and now we will train the dataset using the training set. For providing training or fitting the model to the training set, we will import the **LogisticRegression** class of the **sklearn** library.

After importing the class, we will create a classifier object and use it to fit the model to the logistic regression. Below is the code for it:

```
#Fitting Logistic Regression to the training set
from sklearn.linear_model import LogisticRegression
classifier= LogisticRegression(random_state=0)
classifier.fit(x_train, y_train)
```

Output: By executing the above code, we will get the below output:

```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
                   intercept_scaling=1, l1_ratio=None, max_iter=100,
                   multi_class='warn', n_jobs=None, penalty='l2',
                   random_state=0, solver='warn', tol=0.0001, verbose=0,
                   warm_start=False)
```

Hence our model is well fitted to the training set.

3. Predicting the Test Result

Our model is well trained on the training set, so we will now predict the result by using test set data. Below is the code for it:

```
#Predicting the test set result
y_pred= classifier.predict(x_test)
```

In the above code, we have created a y_pred vector to predict the test set result.

Output: By executing the above code, a new vector (y_pred) will be created under the variable explorer option. It can be seen as:

The above output image shows the corresponding predicted users who want to purchase or not purchase the car.

4. Test Accuracy of the result

Now we will create the confusion matrix here to check the accuracy of the classification. To create it, we need to import the **confusion_matrix** function of the sklearn library. After importing the function, we will call it using a new variable **cm**. The function takes two parameters, mainly **y_true**(the actual values) and **y_pred** (the targeted value return by the classifier). Below is the code for it:

```

```

```
#Creating the Confusion matrix
from sklearn.metrics import confusion_matrix
cm= confusion_matrix()
```

Output:

By executing the above code, a new confusion matrix will be created. Consider the below image:

We can find the accuracy of the predicted result by interpreting the confusion matrix. By above output, we can interpret that 65+24= 89 (Correct Output) and 8+3= 11(Incorrect Output).

5. Visualizing the training set result

Finally, we will visualize the training set result. To visualize the result, we will use **ListedColormap** class of matplotlib library. Below is the code for it:

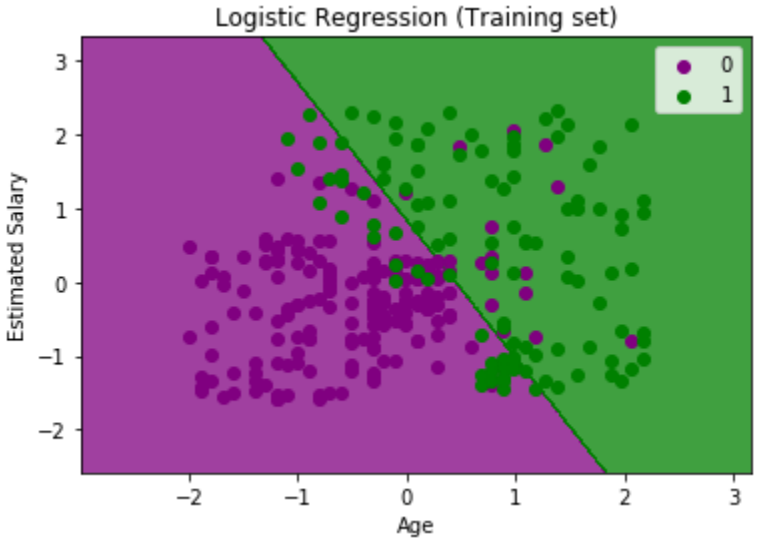
```
#Visualizing the training set result
from matplotlib.colors import ListedColormap
x_set, y_set = x_train, y_train
x1, x2 = nm.meshgrid(nm.arange(start = x_set[:, 0].min() - 1, stop = x_set[:, 0].max() + 1, step  =
0.01),
nm.arange(start = x_set[:, 1].min() - 1, stop = x_set[:, 1].max() + 1, step = 0.01))
```

```
mtp.contourf(x1, x2, classifier.predict(nm.array([x1.ravel(), x2.ravel()]).T).reshape(x1.shape),
alpha = 0.75, cmap = ListedColormap(('purple','green' )))
mtp.xlim(x1.min(), x1.max())
mtp.ylim(x2.min(), x2.max())
for i, j in enumerate(nm.unique(y_set)):
    mtp.scatter(x_set[y_set == j, 0], x_set[y_set == j, 1],
        c = ListedColormap(('purple', 'green'))(i), label = j)
mtp.title('Logistic Regression (Training set)')
mtp.xlabel('Age')
mtp.ylabel('Estimated Salary')
mtp.legend()
mtp.show()
```

In the above code, we have imported the **ListedColormap** class of Matplotlib library to create the colormap for visualizing the result. We have created two new variables **x_set** and **y_set** to replace **x_train** and **y_train**. After that, we have used the **nm.meshgrid** command to create a rectangular grid, which has a range of -1(minimum) to 1 (maximum). The pixel points we have taken are of 0.01 resolution.

To create a filled contour, we have used **mtp.contourf** command, it will create regions of provided colors (purple and green). In this function, we have passed the **classifier.predict** to show the predicted data points predicted by the classifier.

Output: By executing the above code, we will get the below output:



The graph can be explained in the below points:

- In the above graph, we can see that there are some **Green points** within the green region and **Purple points** within the purple region.
- All these data points are the observation points from the training set, which shows the result for purchased variables.
- This graph is made by using two independent variables i.e., **Age on the x-axis** and **Estimated salary on the y-axis**.
- The **purple point observations** are for which purchased (dependent variable) is probably 0, i.e., users who did not purchase the SUV car.
- The **green point observations** are for which purchased (dependent variable) is probably 1 means user who purchased the SUV car.
- We can also estimate from the graph that the users who are younger with low salary, did not purchase the car, whereas older users with high estimated salary purchased the car.
- But there are some purple points in the green region (Buying the car) and some green points in the purple region(Not buying the car). So we can say that younger users with a high estimated salary purchased the car, whereas an older user with a low estimated salary did not purchase the car.

The goal of the classifier:

We have successfully visualized the training set result for the logistic regression, and our goal for this classification is to divide the users who purchased the SUV car and who did not purchase the car. So, from the output graph, we can clearly see the two regions (Purple and Green) with the

observation points. The Purple region is for those users who didn't buy the car, and Green Region is for those users who purchased the car.

Linear Classifier:

As we can see from the graph, the classifier is a Straight line or linear in nature as we have used the Linear model for Logistic Regression. In further topics, we will learn for non-linear Classifiers.

Visualizing the test set result:

Our model is well trained using the training dataset. Now, we will visualize the result for new observations (Test set). The code for the test set will remain same as above except that here we will use **x_test and y_test** instead of **x_train and y_train**. Below is the code for it:

```
#Visulaizing the test set result
from matplotlib.colors import ListedColormap
x_set, y_set = x_test, y_test
x1, x2 = nm.meshgrid(nm.arange(start = x_set[:, 0].min() - 1, stop = x_set[:, 0].max() + 1, step  = 0.01),
nm.arange(start = x_set[:, 1].min() - 1, stop = x_set[:, 1].max() + 1, step = 0.01))
mtp.contourf(x1, x2, classifier.predict(nm.array([x1.ravel(), x2.ravel()]).T).reshape(x1.shape),
alpha = 0.75, cmap = ListedColormap(('purple','green' )))
mtp.xlim(x1.min(), x1.max())
mtp.ylim(x2.min(), x2.max())
for i, j in enumerate(nm.unique(y_set)):
    mtp.scatter(x_set[y_set == j, 0], x_set[y_set == j, 1],
        c = ListedColormap(('purple', 'green'))(i), label = j)
mtp.title('Logistic Regression (Test set)')
mtp.xlabel('Age')
mtp.ylabel('Estimated Salary')
mtp.legend()
mtp.show()
```

The above graph shows the test set result. As we can see, the graph is divided into two regions (Purple and Green). And Green observations are in the green region, and Purple observations are in the purple region. So we can say it is a good prediction and model. Some of the green and purple data points are in different regions, which can be ignored as we have already calculated this error using the confusion matrix (11 Incorrect output).

Hence our model is pretty good and ready to make new predictions for this classification problem.

Evaluation- Confusion Matrix, Precision, Recall, F1 Score, Accuracy model performance metrics that can be used to assess the model performance of a classification model.

Following Matrices will be used to assess the model performance of logistic regression classification model.

1. Confusion matrix
2. Precision
3. Recall
4. F1 Score
5. Accuracy

Let’s understand by example:

We’ll be using an example of a dataset having yes and no labels to be used to train a logistic regression model. This use case can be of any classification problem — spam detection, cancer prediction, attrition rate prediction, campaign target predictions, etc. We’ll be referring to special

use-cases as and when required in this post. For now, we will take into consideration a simple logistic model which has to predict yes or no.

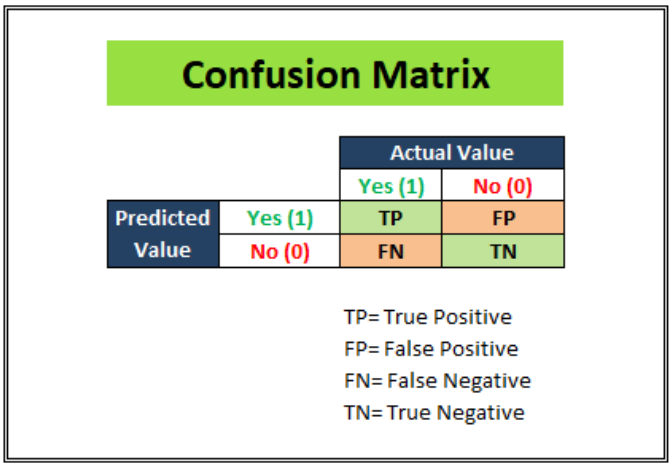
First things first, a logistic model can give two kinds of outputs:

1. It gives out class labels as output values (yes/no, 1/0, malignant/benign, attrited/retained, spam/not spam etc.)

2. It gives probability values between 0 to 1 as output values to signify how likely or how unlikely an event is for a particular observation.

The class labels scenario can be further segmented into the cases of balanced or imbalanced datasets, both of these cannot be judged/should not be judged basis on similar metrics. Some metrics are more suited for but not another and vice-versa. Similarly, the Probabilities scenario has different model performance metrics than the class labels one.

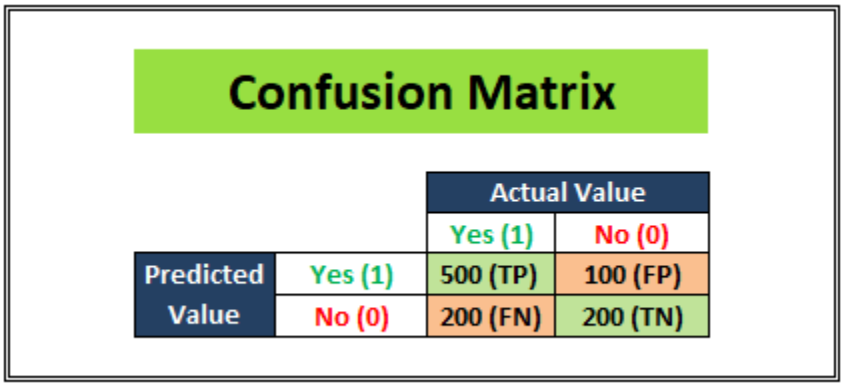
1. Confusion Matrix



Reference: https://towardsdatascience.com/top-10-model-evaluation-metrics-for-classification-ml-models-a0a0f1d51b9

We start with a development dataset while building any statistical or ML model. Divide that dataset into 2 parts: Training and Test. Keep aside the test dataset and train the model using the training dataset. Once the model is ready to predict, we try making predictions on the test dataset. And once we segment the results into a matrix similar to as shown in the above figure, we can see how much our model is able to predict right and how much of its predictions are wrong.

We populate the following 4 cells with the numbers from our test dataset(having 1000 observations for instance).



Reference: https://towardsdatascience.com/top-10-model-evaluation-metrics-for-classification-ml-models-a0a0f1d51b9

1. **TP (True-positives):** Where the actual label for that column was "Yes" in the test dataset and our logistic regression model also predicted "Yes". (500 observations)

2. **TN (True-negatives):** Where the actual label for that column was "No" in the test dataset and our logistic regression model also predicted "No". (200 observations)

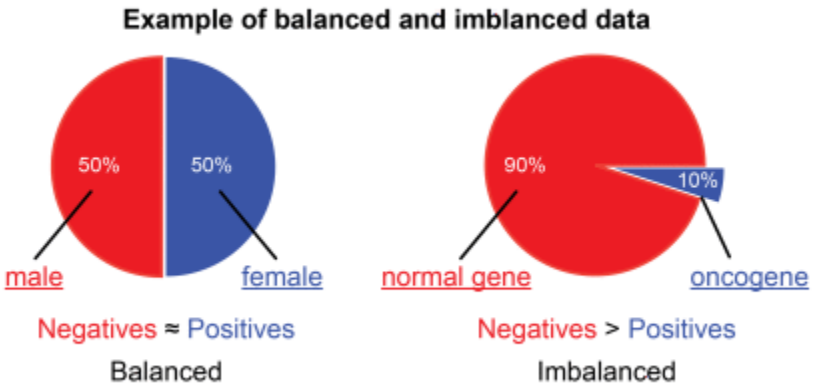
3. **FP (False-positives):** Where the actual label for that column was "No" in the test dataset but our logistic regression model predicted "Yes". (100 observations)

4. **FN (False-negatives):** Where the actual label for that column was "Yes" in the test dataset but our logistic regression model predicted "No". (200 observations)

A confusion matrix is a table that is often used to **describe the performance of a classification model** (or "classifier") on a set of test data for which the true values are known.

2. Accuracy

Accuracy is a metric that is best used for a balanced dataset



Reference: https://medium.com/analytics-vidhya/what-is-balance-and-imbalance-dataset-89e8d7f46bc5

As you can see, a balanced dataset is one where the 1's and 0's, yes's and no's, positive and negatives are equally represented by the training data. On the other hand, if the ratio of the two class-labels is skewed then our model will get biased towards one category.

Assuming we have a balanced dataset, let's learn what is Accuracy.

Accuracy

		Actual Value	
		Yes (1)	No (0)
Predicted Value	Yes (1)	500 (TP)	100 (FP)
	No (0)	200 (FN)	200 (TN)

Accuracy= (TP+TN)/(TP+FP+FN+TN)
=(500+200)/(500+100+200+200)
70%

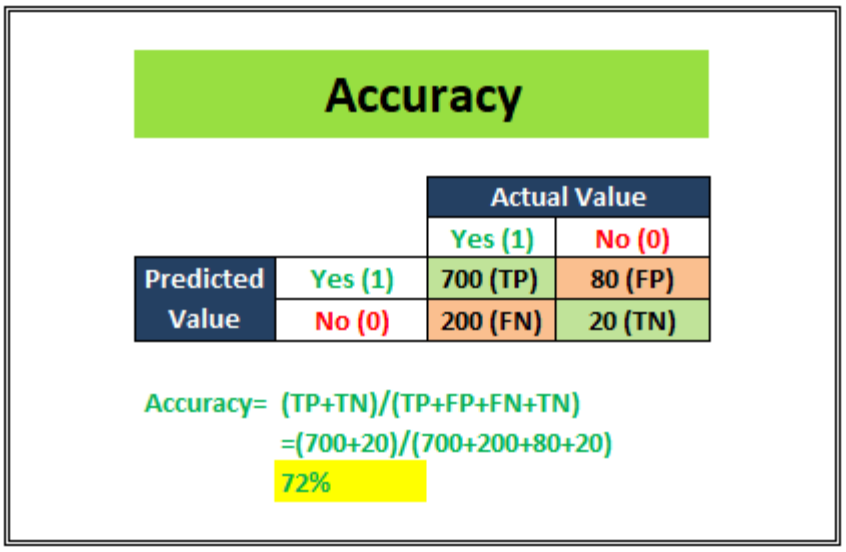
Accuracy is the proximity of measurement results to the true value. It tell us how accurate our classification model is able to predict the class labels given in the problem statement.

For example: Let's suppose that our classification model is trying to predict for customer attrition scenario. In the image above, Out of the total 700 actually attrited customers (TP+FN) , the model was correctly able to classify 500 attrite customers correctly (TP). Similarly, out of the total 300 retained customers (FP+TN), the model was correctly able to classify 200 retained customers correctly (TN).

Accuracy= (TP+TN)/Total customers

In the above scenario, we see that the accuracy of the model on the test dataset of 1000 customers is 70%.

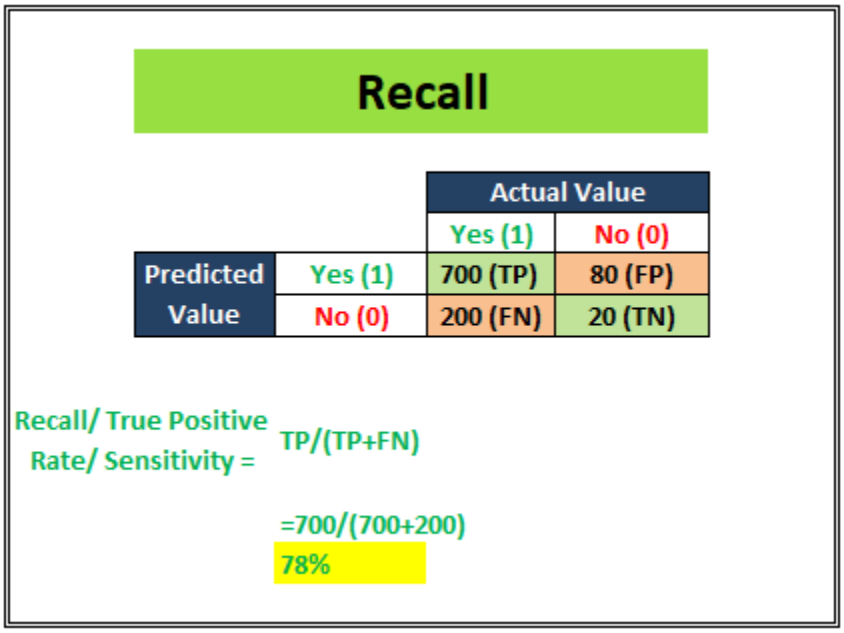
Now, we learned that Accuracy is a metric that should be used only for a balanced dataset. Why is that so? Let's look at an example to understand that.



In this example, this model was trained on an imbalanced dataset and even the test dataset is imbalanced. The Accuracy metric has a score of 72% which might give us the impression that our model is doing a good job at the classification. But, look closer, this model is doing a terrible job out of predicting the Negative class labels. It only predicted 20 correct outcomes out of 100 total negative label observations. This is why the Accuracy metric should not be used if you have an imbalanced dataset.

The next question is, then what is to be used if you have an imbalanced dataset? The answer is Recall and Precision. Let's learn more about these.

3. Recall/ Sensitivity/ TPR



Recall/ Sensitivity/ TPR (True Positive Rate) attempts to answer the following question:

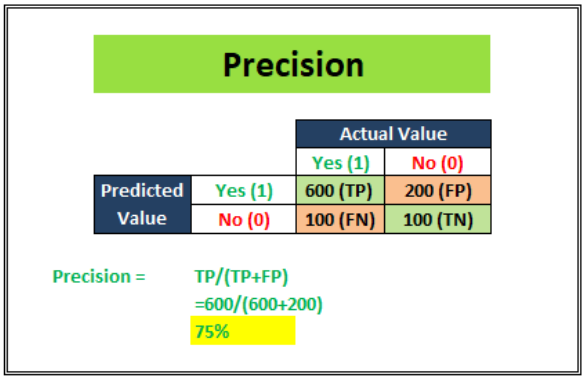
What proportion of actual positives was identified correctly?

sensitivity, recall, hit rate, or true positive rate (TPR)

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{P}} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} = 1 - \mathrm{FNR}$$

This metric gives us 78% as the Recall score in the above image. **Recall is generally used in use cases where the truth-detection is of utmost importance.** For example: The cancer prediction, the stock market classification, etc. over here the problem statement requires that the False negatives be minimized which implies Recall/Sensitivity be maximized.

4. Precision



Precision attempts to answer the following question:

What proportion of positive identifications was actually correct?

precision or positive predictive value (PPV)

$$\mathrm{PPV} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} = 1 - \mathrm{FDR}$$

The example shown in the above image shows us that the Precision score is 75%. Precision is generally used in cases where it's of utmost importance not to have a high number of False positives. For example: In spam detection cases, as we discussed above, a false positive would be an observation that was not spam but was classified as Spam by our classification model. Too many of the false positives can defeat the purpose of a spam classifier model. Thus, Precision comes handy here to judge the model performance in this scenario.

5. F1 Score

We talked about Recall and Precision in points numbers 6 and 7 respectively. We understand that there are some problem statements where a higher Recall takes precedence over a higher Precision and vice-versa.

But there are some use-cases, where the distinction is not very clear and as developers, we want to give importance to both Recall and Precision. In this case, there is another metric- F1 Score that can be used. It is dependent on both Precision and Recall.

In a statistical analysis of binary classification, the **F1 score** (also **F-score** or **F-measure**) is a measure of a test's accuracy. It considers both the precision p and the recall r of the test to compute the score

The traditional F-measure or balanced F-score (**F_1 score**) is the harmonic mean of precision and recall:

$$F_1 = \left(\frac{2}{\mathrm{recall}^{-1} + \mathrm{precision}^{-1}} \right) = 2 \cdot \frac{\mathrm{precision} \cdot \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}}.$$

Python Library: Sci-Kit Learn

What is Scikit-Learn?

Open-source ML library for Python. Built on NumPy, SciPy, and Matplotlib.

Scikit-learn is a library in Python that provides many unsupervised and supervised learning algorithms. It's built upon some of the technology you might already be familiar with, like NumPy, pandas, and Matplotlib!
The functionality that scikit-learn provides include:

- **Regression**, including Linear and Logistic Regression
- **Classification**, including K-Nearest Neighbors
- **Clustering**, including K-Means and K-Means++
- **Model selection**
- **Pre-processing**, including Min-Max Normalization

Prerequisites

Before we start using scikit-learn latest release, we require the following −

- Python (>=3.5)
- NumPy (>= 1.11.0)
- Scipy (>= 0.17.0)li
- Joblib (>= 0.11)
- Matplotlib (>= 1.5.1) is required for Sklearn plotting capabilities.
- Pandas (>= 0.18.0) is required for some of the scikit-learn examples using data structure and analysis.

Installation

If you already installed NumPy and Scipy, following are the two easiest ways to install scikit-learn −

Using pip

Following command can be used to install scikit-learn via pip −

pip install -U scikit-learn

Using conda

Following command can be used to install scikit-learn via conda −

conda install scikit-learn

On the other hand, if NumPy and Scipy is not yet installed on your Python workstation then, you can install them by using either **pip** or **conda**.
Another option to use scikit-learn is to use Python distributions like ***Canopy*** and ***Anaconda*** because they both ship the latest version of scikit-learn.

Features of scikit-learn

Rather than focusing on loading, manipulating and summarising data, Scikit-learn library is focused on modelling the data. Some of the most popular groups of models provided by Sklearn are as follows –

Supervised Learning algorithms − Almost all the popular supervised learning algorithms, like Linear Regression, Support Vector Machine (SVM), Decision Tree etc., are the part of scikit-learn.

Unsupervised Learning algorithms − On the other hand, it also has all the popular unsupervised learning algorithms from clustering, factor analysis, PCA (Principal Component Analysis) to unsupervised neural networks.

Clustering − This model is used for grouping unlabelled data.

Cross Validation − It is used to check the accuracy of supervised models on unseen data.

Dimensionality Reduction − It is used for reducing the number of attributes in data which can be further used for summarisation, visualisation and feature selection.

Ensemble methods − As name suggest, it is used for combining the predictions of multiple supervised models.

Feature extraction − It is used to extract the features from data to define the attributes in image and text data.

Feature selection − It is used to identify useful attributes to create supervised models.

Open Source − It is open-source library and also commercially usable under BSD license.

Dataset Loading

A collection of data is called dataset. It is having the following two components −
Features − The variables of data are called its features. They are also known as predictors, inputs or attributes.

- **Feature matrix** − It is the collection of features, in case there are more than one.
- **Feature Names** − It is the list of all the names of the features.

Response − It is the output variable that basically depends upon the feature variables. They are also known as target, label or output.

- **Response Vector** − It is used to represent response column. Generally, we have just one response column.
- **Target Names** − It represent the possible values taken by a response vector.

Scikit-learn have few example datasets like **iris** and **digits** for classification and the **Boston house prices** for regression.

Example

Following is an example to load **iris** dataset −

```
from sklearn.datasets import load_iris
iris = load_iris()
X = iris.data
y = iris.target
feature_names = iris.feature_names
target_names = iris.target_names
print("Feature names:", feature_names)
print("Target names:", target_names)
print("\nFirst 10 rows of X:\n", X[:10])
```

Output:

```
Feature names: ['sepal length (cm)', 'sepal width (cm)', 'petal length (cm)', 'petal width (cm)']
Target names: ['setosa' 'versicolor' 'virginica']
First 10 rows of X:
[
  [5.1 3.5 1.4 0.2]
  [4.9 3. 1.4 0.2]
  [4.7 3.2 1.3 0.2]
  [4.6 3.1 1.5 0.2]
  [5. 3.6 1.4 0.2]
  [5.4 3.9 1.7 0.4]
  [4.6 3.4 1.4 0.3]
  [5. 3.4 1.5 0.2]
  [4.4 2.9 1.4 0.2]
  [4.9 3.1 1.5 0.1]
]
```

Splitting the dataset

To check the accuracy of our model, we can split the dataset into two pieces-**a training set** and **a testing set**. Use the training set to train the model and testing set to test the model. After that, we can evaluate how well our model did.

Example

The following example will split the data into 70:30 ratio, i.e. 70% data will be used as training data and 30% will be used as testing data. The dataset is iris dataset as in above example.

```
from sklearn.datasets import load_iris
iris = load_iris()

X = iris.data
y = iris.target
```

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(
   X, y, test_size = 0.3, random_state = 1
)
```

```
print(X_train.shape)
print(X_test.shape)

print(y_train.shape)
print(y_test.shape)
```

Output:

```
(105, 4)
(45, 4)
(105,)
(45,)
```

As seen in the example above, it uses **train_test_split()** function of scikit-learn to split the dataset. This function has the following arguments −

- **X, y** − Here, **X** is the **feature matrix** and y is the **response vector**, which need to be split.
- **test_size** − This represents the ratio of test data to the total given data. As in the above example, we are setting **test_data = 0.3** for 150 rows of X. It will produce test data of 150*0.3 = 45 rows.
- **random_size** − It is used to guarantee that the split will always be the same. This is useful in the situations where you want reproducible results.

Train the Model

Next, we can use our dataset to train some prediction-model. As discussed, scikit-learn has wide range of **Machine Learning (ML) algorithms** which have a consistent interface for fitting, predicting accuracy, recall etc.

Example

In the example below, we are going to use KNN (K nearest neighbors) classifier. Don't go into the details of KNN algorithms, as there will be a separate chapter for that. This example is used to make you understand the implementation part only.

```
from sklearn.datasets import load_iris
iris = load_iris()
X = iris.data
y = iris.target
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(
   X, y, test_size = 0.4, random_state=1
)
from sklearn.neighbors import KNeighborsClassifier
from sklearn import metrics
classifier_knn = KNeighborsClassifier(n_neighbors = 3)
classifier_knn.fit(X_train, y_train)
y_pred = classifier_knn.predict(X_test)
# Finding accuracy by comparing actual response values(y_test)with predicted response value(y_pred)
print("Accuracy:", metrics.accuracy_score(y_test, y_pred))
# Providing sample data and the model will make prediction out of that data

sample = [[5, 5, 3, 2], [2, 4, 3, 5]]
preds = classifier_knn.predict(sample)
pred_species = [iris.target_names[p] for p in preds] print("Predictions:", pred_species)
```

Output:

```
Accuracy: 0.9833333333333333
Predictions: ['versicolor', 'virginica']
```

There will be many more function in scikit-learn to explore more Machine learning algorithms. Following some important features, you can use for machine learning model.

Linear Regression

This supervised ML model is used when the output variable is continuous and it follows linear relation with dependent variables. It can be used to forecast sales in the coming months by analyzing the sales data for previous months.

With the help of sklearn, we can easily implement the Linear Regression model as follows:

```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
regression_model = LinearRegression()
regression_model.fit(x_train, y_train)
y_predicted = regression_model.predict(x_test)
rmse = mean_squared_error(y_test, y_predicted)
r2 = r2_score(y_test, y_predicted)
```

LinerRegression() creates an object of linear regression. Then we fit the model on the training set. Finally, we predicted the model on the test dataset. “rmse” and “r_score” can be used to check the accuracy of the model.

More function you can explore with machine learning examples.

References

1. https://www.investopedia.com/terms/d/descriptive_statistics.asp
2. https://www.abs.gov.au/websitedbs/D3310114.nsf/home/statistical+language+-+measures+of+spread
3. https://statistics.laerd.com/statistical-guides/measures-of-spread-range-quartiles.php
4. https://worldsustainable.org/inferential-statistics-examples/
5. https://www.analyticsvidhya.com/blog/2017/09/6-probability-distributions-data-science/
6. https://www.questionpro.com/blog/types-of-sampling-for-social-research/
7. https://www.mastersindatascience.org/learning/statistics-data-science/probability-theory/#:~:text=Probability%20theory%20is%20a%20branch,the%20analysis%20of%20chance%20events.
8. https://corporatefinanceinstitute.com/resources/knowledge/other/bayes-theorem/
9. https://www.aptech.com/blog/beginners-guide-to-maximum-likelihood-estimation-in-gauss/
10. https://www.investopedia.com/articles/active-trading/092214/hypothesis-testing-finance-concept-examples.asp
11. https://www.managementstudyguide.com/hypothesis-testing.htm
12. https://www.investopedia.com/terms/h/hypothesistesting.asp
13. https://corporatefinanceinstitute.com/resources/knowledge/other/central-limit-theorem/
14. https://www.investopedia.com/terms/c/chi-square-statistic.asp
15. https://www.geeksforgeeks.org/machine-learning/
16. https://towardsdatascience.com/performance-metrics-in-machine-learning-part-2-regression-c60608f3ef6a
17. https://corporatefinanceinstitute.com/resources/knowledge/finance/correlation/
18. https://www.i2tutorials.com/tag/ordinary-least-square-method-in-machine-learning/
19. https://statisticsbyjim.com/glossary/ordinary-least-squares/

20. https://www.potentiaco.com/what-is-machine-learning-definition-types-applications-and-examples
21. https://www.investopedia.com/terms/m/mlr.asp#:~:text=Key%20Takeaways-,Multiple%20linear%20regression
22. https://mycstutorial.in/data-handling-using-pandas-i/
23. http://python.mykvs.in/presentation/presentation2021/class%20xii/informatics%20practices/Python%20Pandas.pdf
24. https://towardsdatascience.com/data-handling-using-pandas-cleaning-and-processing-3aa657dc9418
25. https://matplotlib.org/
26. https://www.mygreatlearning.com/blog/matplotlib-tutorial-for-data-visualisation/
27. https://datacarpentry.org/python-socialsci/13-matplotlib/index.html
28. https://www.geeksforgeeks.org/data-visualization-with-python-seaborn/
29. https://datalore.jetbrains.com/view/notebook/v8mLoENq8XTfmStTCLNMV6
30. https://www.edureka.co/blog/python-seaborn-tutorial/
31. https://towardsdatascience.com/exploratory-data-analysis-with-pandas-profiling-de3aae2ddff3
32. https://www.thiscodeworks.com/generate-pandas-profiling-report-and-save-it-to-html-python/61f0b8384faf03001546b098
33. https://www.geeksforgeeks.org/what-is-data-visualization-and-why-is-it-important/
34. https://splashbi.com/importance-purpose-benefit-of-data-visualization-tools/
35. https://www.edureka.co/blog/needs-and-benefits-of-data-visualization/
36. https://piktochart.com/blog/what-is-data-visualization/
37. https://www.simplilearn.com/what-is-data-mining-article
38. https://expressanalytics.com/blog/what-is-data-wrangling-what-are-the-steps-in-data-wrangling/
39. https://analyticsindiamag.com/9-effective-pandas-techniques-in-python-for-data-manipulation/
40. https://monkeylearn.com/blog/data-preprocessing/
41. https://www.guru99.com/numpy-tutorial.html
42. https://www.statisticshowto.com/residual/
43. https://machinelearningmastery.com/polynomial-features-transforms-for-machine-learning/#:~:text=Polynomial%20features%20are%20those%20features,X%2C%20e.g.%20X%5E2.
44. https://monkeylearn.com/blog/classification-algorithms/

45. https://machinelearningmastery.com/distance-measures-for-machine-learning/#:~:text=Perhaps%20four%20of%20the%20most,Manhattan%20Distance

46. https://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning

47. https://towardsdatascience.com/gradient-descent-algorithm-a-deep-dive-cf04e8115f21

48. https://www.javatpoint.com/gradient-descent-in-machine-learning
49. https://www.javatpoint.com/logistic-regression-in-machine-learning

50. https://towardsdatascience.com/top-10-model-evaluation-metrics-for-classification-ml-models-a0a0f1d51b9

51. https://www.tutorialspoint.com/scikit_learn/scikit_learn_introduction.htm

52. https://scikit-learn.org/stable/tutorial/basic/tutorial.html

53. https://www.analyticsvidhya.com/blog/2021/07/15-most-important-features-of-scikit-learn/#:~:text=Through%20scikit%2Dlearn%2C%20we%20can,ensemble%20techniques%2C%20and%20inbuilt%20datasets.