2021

Artificial Intelligence

Disclaimer: The content is curated for educational purposes only.
© Edunet Foundation. All rights reserved.

Table of Contents

Table of Contents 2
Learning Outcome 5
Chapter 1: What is AI and ML? 6
1.1 Introduction to Artificial Intelligence 6
1.2 AI Career Opportunities 12
1.3 Introduction to Machine Learning 13
1.4 PRACTICAL : Image Recognition on Web Browser 17
1.5 What is Data Analysis? 25
1.6 Anaconda Software and Introduction to Python 31
1.7 Variables in Python 35
1.8 Advanced Datatypes in Python 36
1.9 Condition & Loop in Python 38
1.10 Python Libraries 46
1.11 Numpy Library 47
1.12 Matrix Multiplication in Numpy 51
1.13 NumPy Statistical Functions 52
1.14 Percentile & Interquartile in Numpy 56
1.15 Array Broadcasting in Numpy 58
1.16 What is Data Visualization? 59
1.17 Graph Plotting in Python using Matplotlib 63
1.18 Line Plot, Bar Plot, Scatter Plot 64
1.19 Three-Dimensional Plotting in Matplotlib 66
1.20 Visualizing with NumPy 69
DIY Activity 71
Chapter 2: Exploratory Data Analysis and Supervised Learning 72
2.1 Data Manipulation with Pandas 72
2.2 Introducing Pandas Objects 76
2.3 Creating Series from simple datatypes 79
2.4 Data Storage Formats in Pandas 82
2.5 CSV file and JSON file 82
2.6 Reading data from files 83
2.7 Groupby Methods 84

© Edunet Foundation. All rights reserved.

2.8 Pivot Tables 87
2.9 Pandas Plotting 89
2.10 Machine Learning Basics 95
2.11 Techniques of ML 99
2.12 Scikit Learn library overview 102
2.12 Regression vs Classification 103
2.13 Least Square Method 105
2.14 Azure ML No Code Platform 112
2.15 Setting up an account in Azure ML Studio and Creating Workspace 113
2.16 Training a ML model in Azure Studio 117
2.17 Create a Regression model with Azure ML Studio 129
Do it Yourself Activity 142
Chapter 3 : Text Analytics and Building Machine Learning Models 142
3.1 Probability Theory 142
3.2 Naïve Bayes Classifier Algorithm 143
3.3 Bag of Words Approach 145
3.4 Lexicons for Sentiment Analysis 147
Do it Yourself Activity 148
Chapter 4: Building Deep Learning Models 149
4.1 Deep Learning Basics 149
4.2 Concept of Neural Networks 151
4.3 Overfitting in Deep Learning 156
4.4 Computer Vision Basics 163
4.5 Image Processing 167
4.6 Facial Detection 171
Do it Yourself 176
Reference 176

© Edunet Foundation. All rights reserved.

This course booklet has been designed by Edunet Foundation for the Tech-Saksham programme in partnership with Microsoft and SAP

© Edunet Foundation. All rights reserved.

Learning Outcome

After completing this handbook, learner will be able to

- Demonstrate fundamental understanding of the history of artificial intelligence and its foundations.
- Apply the basic principles, models, and algorithms of AI to recognize, model, and solve problems in the analysis and design of information systems.
- Analyze the structures and algorithms of a selection of techniques related to machine learning and Artificial Intelligence.
- Able to design and implement various machine learning algorithms in a range of real-world applications.
- Appreciate the underlying mathematical relationships within and across Machine Learning algorithms and the paradigms of supervised learning.
- Be able to identify new application requirements in the field of computer vision using Deep Learning.

© Edunet Foundation. All rights reserved.

Chapter 1: What is AI and ML?

Learning Outcomes:

- Understand the AI market trends, investments and career opportunities
- Understand fundamental concepts of Artificial Intelligence and Machine Learning
- Create a Machine Learning model to differentiate between Images
- Explore the fundamentals of Data Analysis
- Understand key concepts of Python Programming

1.1 Introduction to Artificial Intelligence

Definitions

Artificial intelligence (**AI**) is the intelligence exhibited by machines or software. It is also the name of the academic field of study which studies how to create computers and computer software that are capable of intelligent behaviour. It is an area of computer science that emphasizes the creation of intelligent machines that work and react like humans. Some of the activities computers with artificial intelligence are designed for include: Speech recognition, Learning, Planning, Problem solving. In this topic we shall discus the following subjects; Deep learning, Machine learning, Computer Programming, Medical field.

It has become an essential part of the technology industry. Research associated with artificial intelligence is highly technical and specialized. The core problems of artificial intelligence include programming computers for certain traits such as: Knowledge, Reasoning, Problem solving, Perception, Learning, Planning, and Ability to manipulate and move objects. Knowledge engineering is a core part of AI research.

Machines can often act and react like humans only if they have abundant information relating to the world. Artificial intelligence must have access to objects, categories, properties and relations between all of them to implement knowledge engineering. Initiating common sense, reasoning and problem-solving power in machines is a difficult and tedious approach. Machine learning is another core part of AI.

Learning without any kind of supervision requires an ability to identify patterns in streams of inputs, whereas learning with adequate supervision involves classification and numerical regressions. Classification determines the category an object belongs to and regression deals with obtaining a set of numerical input or output examples,

© Edunet Foundation. All rights reserved.

thereby discovering functions enabling the generation of suitable outputs from respective inputs.

Mathematical analysis of machine learning algorithms and their performance is a well-defined branch of theoretical computer science often referred to as computational learning theory. Machine perception deals with the capability to use sensory inputs to deduce the different aspects of the world, while computer vision is the power to analyse visual inputs with a few sub- problems such as facial, object and gesture recognition. Robotics is also a major field related to AI. Robots require intelligence to handle tasks such as object manipulation and navigation, along with sub-problems of localization, motion planning and mapping.

Classification of AI

Artificial Intelligence, also called **(AI)**, is an imitation of the human intelligence system, and the processes involved are carried out by a computing device (a computer, robot, smartphone or any such device). It includes the processes of **acquiring information**, **validating assigned rules**, and **debugging itself** by the intelligent software and **executing actions** by the hardware installed with it.

Let's simplify the definition of AI; a machine, robot or android that has an intelligent system like a human brain which can **sense**, **reason**, **act** & **adapt** according to the operational instructions. It works based on the data, stored as well as collected, and configures itself with real-time instructions.

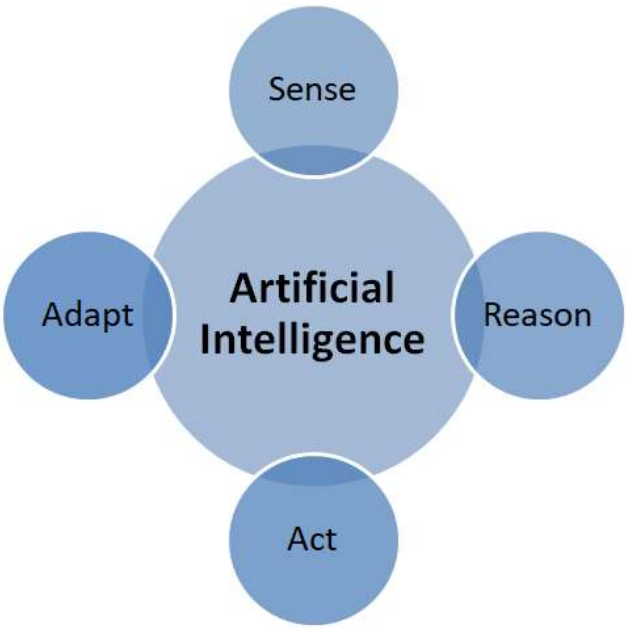


Image: Basic Block diagram of AI

- Sense: It's one of the properties of AI. It not only identifies worthy materials & objects but also recognizes real-time operational activities. The sensors or

© Edunet Foundation. All rights reserved.

sensing devices can trace out & quickly differentiate between wrong & correct objects.

- Reason: This property works like a human brain does to complete a task successfully. It understands, judges and prepares to execute. Reason enables AI to deal with internal & external properties of resources like condition, time-frame, behavior, other parameters of entities involved during conversion of the task.

- Act: This is a decisive property that enables to execute an action or send instructions to others to execute the action instead. The act is the part where the functionality & operational activities are directly proportionate.

- Adapt: It's the property that works with the highest intelligence. Like the way, a human brain remembers the result of any past event. Its re-trains, debugs and even find out uncovered properties to put the operation more accurate. It remembers the last events & manages the functionalities accordingly.

For AI data is an essential element that underpins AI's underlying logic. Without data, data processing for AI will not be possible. With data mining's cleaning, integration, reduction and other pre-treatment means, AI could have adequate data for learning. As AI technologies iterate, the production, collection, storage, calculation, transmission and application of data will all be completed by machines.

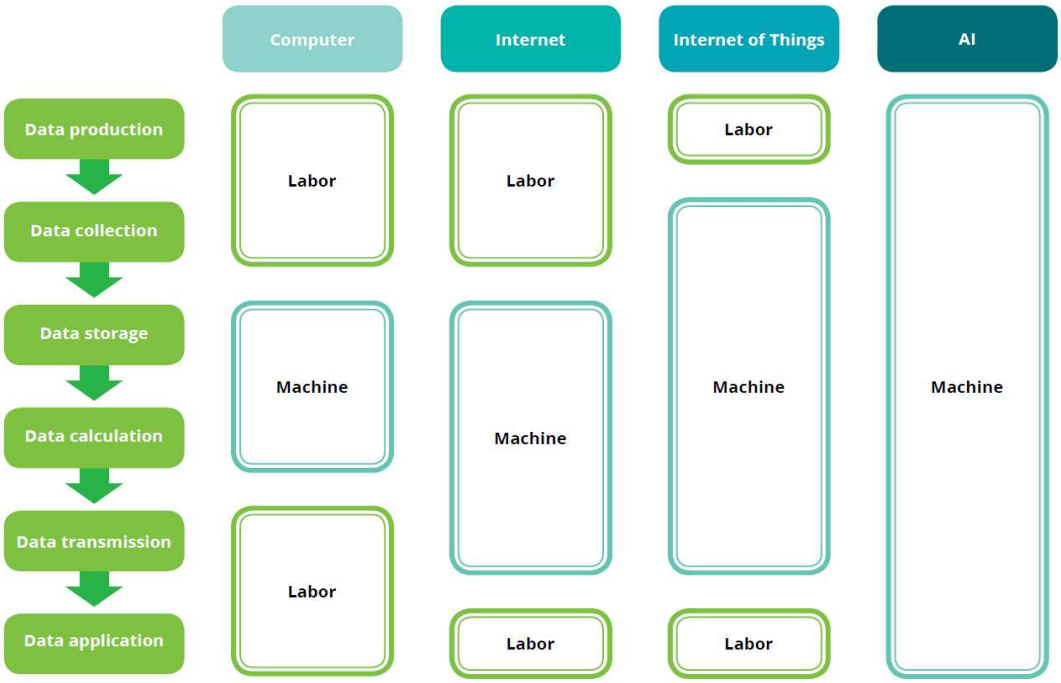


Image: Development stages of data processing
Reference: Public information, Deloitte Research

© Edunet Foundation. All rights reserved.

Applications of AI

Here we have some of the Artificial Intelligence Applications in real world.

Image: Applications of Artificial Intelligence

Reference: https://techvidvan.com/tutorials/artificial-intelligence-applications/

1.Healthcare

One of the foremost deep-lying impacts which AI has created is within the Healthcare space. A device, as common as a Fitbit or an iWatch, collects a lot of data like the sleep patterns of the individual, the calories burnt by him, heart rate and a lot more which can help with early detection, personalization, even disease diagnosis.

This device, when powered with AI can easily monitor and notify abnormal trends. This can even schedule a visit to the closest Doctor by itself and therefore, it's also of great help to the doctors who can get help in making decisions and research with AI. It has been used to predict ICU transfers, improve clinical workflows and even pinpoint a patient's risk of hospital-acquired infections.

2. Automobile

At this stage where automobiles changing from an engine with a chassis around it to a software-controlled intelligent machine, the role of AI cannot be underestimated.

The goal of self-driving cars, during which Autopilot by Tesla has been the frontrunner, takes up data from all the Tesla's running on the road and uses it in machine learning algorithms. The assessment of both chips is later matched by the system and followed if the input from both is the same.

3. Banking and Finance

One of the early adopters of Artificial Intelligence is the Banking and Finance Industry. Features like AI bots, digital payment advisers and biometric fraud detection mechanisms cause higher quality of services to a wider customer base.

© Edunet Foundation. All rights reserved.

The adoption of AI in banking is constant to rework companies within the industry, provide greater levels useful and more personalized experiences to their customers, reduce risks as well as increase opportunities involving financial engines of our modern economy.

4. Surveillance

AI has made it possible to develop face recognition Tools which may be used for surveillance and security purposes. As a result, this empowers the systems to monitor the footage in real-time and can be a pathbreaking development in regards to public safety.

Manual monitoring of a CCTV camera requires constant human intervention so they're prone to errors and fatigue.AI-based surveillance is automated and works 24/7, providing real-time insights. According to a report by the Carnegie Endowment for International Peace, a minimum of 75 out of the 176 countries are using AI tools for surveillance purposes.

Across the country, 400 million CCTV cameras are already in situ, powered by AI technologies, primarily face recognition.

5. Social Media

Social Media is not just a platform for networking and expressing oneself. It subconsciously shapes our choices, ideologies, and temperament.

This is due to the synthetic Intelligence tools which work silently within the background, showing us posts that we "might" like and advertising products that "might" be useful based on our search and browsing history.

This helps with social media advertising because of its unprecedented ability to run paid ads to platform users based on highly granular demographic and behavioural targeting.

6. Entertainment

The show business, with the arrival of online streaming services like Netflix and Amazon Prime, relies heavily on the info collected by the users.

This helps with recommendations based upon the previously viewed content. This is done not only to deliver accurate suggestions but also to create content that would be liked by a majority of the viewers.

With new contents being created every minute, it is very difficult to classify them and making them easier to search.AI tools analyse the contents of videos frame by frame and identify objects to feature appropriate tags.AI is additionally helping media companies to form strategic decisions.

© Edunet Foundation. All rights reserved.

7. Education

In the education sector also, there are a number of problems which will be solved by the implementation of AI.

A few of them being automated marking software, content retention techniques and suggesting improvements that are required. This can help the teachers monitor not just the academic but also the psychological, mental and physical well-being of the students but also their all-round development.

This would also help in extending the reach of education to areas where quality educators can't be present physically.

8. Space Exploration

AI systems are being developed to scale back the danger of human life that venture into the vast realms of the undiscovered and unravelled universe which is a very risky task that the astronauts need to take up.

As a result, unmanned space exploration missions just like the Mars Rover are possible due to the utilization of AI. It has helped us discover numerous exoplanets, stars, galaxies, and more recently, two new planets in our very own system.

NASA is also working with AI applications for space exploration to automate image analysis and to develop autonomous spacecraft that would avoid space debris without human intervention, create communication networks more efficient and distortion-free by using an AI-based device.

9. Gaming

In the gaming industry also, computer game Systems powered by AI is ushering us into a replacement era of immersive experience in gaming. It serves to enhance the game-player experience instead of machine learning or deciding. AI has also been playing a huge role in creating video games and making it more tailored to players' preferences.

10. Robotics

With increasing developments within the field of AI, robots are becoming more efficient in performing tasks that earlier were too complex.

AI in robotics helps the robots to learn the processes and perform the tasks with complete autonomy, without any human intervention. This is because robots are designed to perform repetitive tasks with utmost precision and increased speed.

AI has been introducing flexibility and learning capabilities in previously rigid applications of robots. These benefits are expected to reinforce the market growth.

© Edunet Foundation. All rights reserved.

11. Agriculture

Artificial Intelligence is changing the way we do one among our most primitive and basic professions which are farming. The use of AI in agriculture are often attributed to agriculture robots, predictive analysis, and crop and soil monitoring.
In addition, drones are also used for spraying insecticides and detecting weed formation in large farms. This is getting to help firms like Blue River Technologies, better manage the farms.
AI has also enhanced crop production and improved real-time monitoring, harvesting, processing and marketing.

12. E-Commerce

This is one of the Artificial Intelligence Applications that's found to be widely used.
Different departments of E-commerce including logistics, predicting demand, intelligent marketing, better personalization, use of chatbots, etc. are being disrupted by AI. The E-Commerce industry, a prominent player being Amazon is one among the primary industries to embrace AI. This may experience a good use of AI with time.
E-commerce retailers are increasingly turning towards chatbots or digital assistants to supply 24×7 support to their online buyers.

1.2 AI Career Opportunities

The World Economic Forum's "The Future of Jobs 2018" aims to base this debate on facts rather than speculation. By tracking the acceleration of technological change as it gives rise to new job roles, occupations and industries, the report evaluates the changing contours of work in the Fourth Industrial Revolution.

One of the primary drivers of change identified is the role of emerging technologies, such as artificial intelligence (AI) and automation. The report seeks to shed more light on the role of new technologies in the labour market, and to bring more clarity to the debate about how AI could both create and limit economic opportunity. With 575 million members globally, LinkedIn's platform provides a unique vantage point into global labour-market developments, enabling us to support the Forum's examination of the trends that will shape the future of work.

Our analysis uncovered two concurrent trends: the continued rise of tech jobs and skills, and, in parallel, a growth in what we call "human-centric" jobs and skills. That is, those that depend on intrinsically human qualities.

© Edunet Foundation. All rights reserved.

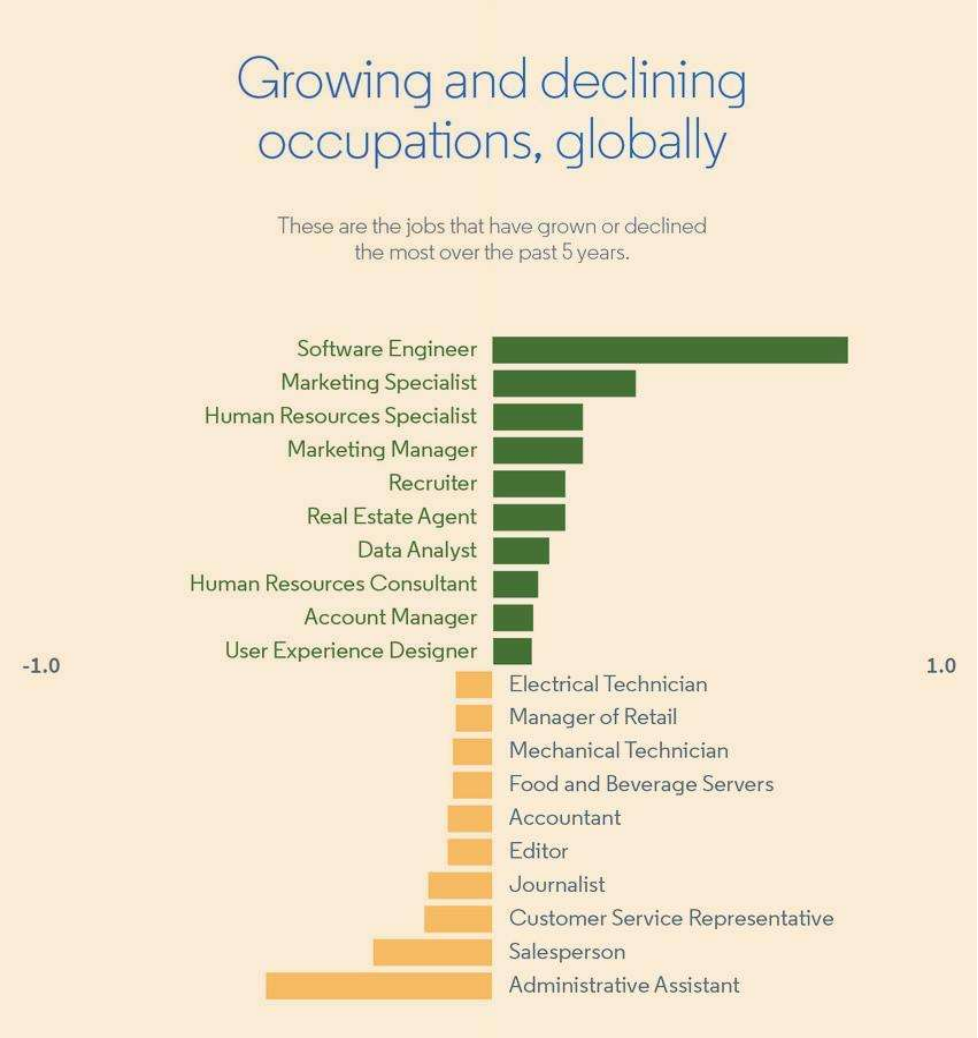


Image: career opportunities in AI
Reference: https://www.weforum.org/agenda/2018/09/artificial-intelligence-shaking-up-job-market/

Tech jobs like software engineers and data analysts, along with technical skills such as cloud computing, mobile application development, software testing and AI, are on the rise in most industries and across all regions. But several highly "automatable" jobs fall into the top 10 most declining occupations – i.e., jobs that have seen the largest decreases in share of hiring over the past five years. These occupations include administrative assistants, customer service representatives, accountants, and electrical/mechanical technicians, many of which depend on more repetitive tasks.

1.3 Introduction to Machine Learning

Machine learning is the study of computer algorithms that improve automatically through experience and has been central to AI research since the field's inception. Unsupervised learning is the ability to find patterns in a stream of input. Supervised learning includes both classification and numerical regression.

© Edunet Foundation. All rights reserved.

Classification is used to determine what category something belongs in, after seeing a number of examples of things from several categories. Regression is the attempt to produce a function that describes the relationship between inputs and
outputs and predicts how the outputs should change as the inputs change. In reinforcement learning the agent is rewarded for good responses and punished for bad ones.

Introduction of ML

Machine learning pipeline helps to automate ML Workflow and enable the sequence data to be transformed and correlated together in a model to analyzed and achieve outputs. ML pipeline is constructed to allow the flow of data from raw data format to some valuable information. It provides a mechanism to build a Multi-ML parallel pipeline system to examine different ML methods' outcomes.
The Objective of the Machine learning pipeline is to exercise control over the ML model. A well-planned pipeline helps to makes the implementation more flexible. It is like having an overview of a code to pick the fault and replace them with the correct code.
A pipeline consists of several stages. Each stage of a pipeline is fed with the data processed from its preceding stage, i.e., the output of a processing unit supplied as an input to the next step. Machine Learning Pipeline consists of four main stages as Pre-processing, Learning, Evaluation, and Prediction.

1. Pre-processing

Data preprocessing is a Data Mining technique that involves transferring raw data into an understandable format. Real-world data is usually incomplete, inconsistent, and lacks certain behaviors or trends, most likely to contain many inaccuracies. The process of getting usable data for a Machine Learning algorithm follows steps such as Feature Extraction and Scaling, Feature Selection, Dimensionality reduction, and sampling. The product of Data Pre-processing is the final dataset used for training the model and testing purposes.

2. Learning

A learning algorithm is used to process understandable data to extract patterns appropriate for application in a new situation. In particular, the aim is to utilize a system for a specific input-output transformation task. For this, choose the best-performing model from a set of models produced by different hyperparameter settings, metrics, and cross-validation techniques.

3. Evaluation

To Evaluate the Machine Learning model's performance, fit a model to the training data, and predict the labels of the test set. Further, count the number of wrong predictions on the test dataset to compute the model's prediction accuracy.

© Edunet Foundation. All rights reserved.

4. Prediction

The model's performance to determine the outcomes of the test data set was not used for any training or cross-validation activities.

Machine Learning Applications

Machine learning 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 which makes it more similar to humans: The ability to learn. Machine learning is actively being used today, perhaps in many more places than one would expect. We probably use a learning algorithm dozen of time without even knowing it. Applications of Machine Learning include:

a. ***Web Search Engine***

One of the reasons why search engines like google, Bing etc. work so well is because the system has learnt how to rank pages through a complex learning algorithm.

b. ***Photo tagging Applications***

Be it Facebook or any other photo tagging application, the ability to tag friends makes it even more happening. It is all possible because of a face recognition algorithm that runs behind the application.

c. ***Spam Detector***

Our mail agent like Gmail or Hotmail does a lot of hard work for us in classifying the mails and moving the spam mails to spam folder. This is again achieved by a spam classifier running in the back end of mail application.

The key difference between AI and ML are:

ARTIFICIAL INTELLIGENCE	MACHINE LEARNING
AI stands for Artificial intelligence, where intelligence is defined acquisition of knowledge intelligence is defined as an ability to acquire and apply knowledge.	ML stands for Machine Learning which is defined as the acquisition of knowledge or skill
The aim is to increase chance of success and not accuracy.	The aim is to increase accuracy, but it does not care about success
It works as a computer program that does smart work	It is a simple concept machine takes data and learn from data.

© Edunet Foundation. All rights reserved.

The goal is to simulate natural intelligence to solve complex problem	The goal is to learn from data on certain task to maximize the performance of machine on this task.
AI is decision making.	ML allows system to learn new things from data.
It leads to develop a system to mimic human to respond behave in a circumstance.	It involves in creating self-learning algorithms.
AI will go for finding the optimal solution.	ML will go for only solution for that whether it is optimal or not.
AI leads to intelligence or wisdom.	ML leads to knowledge.

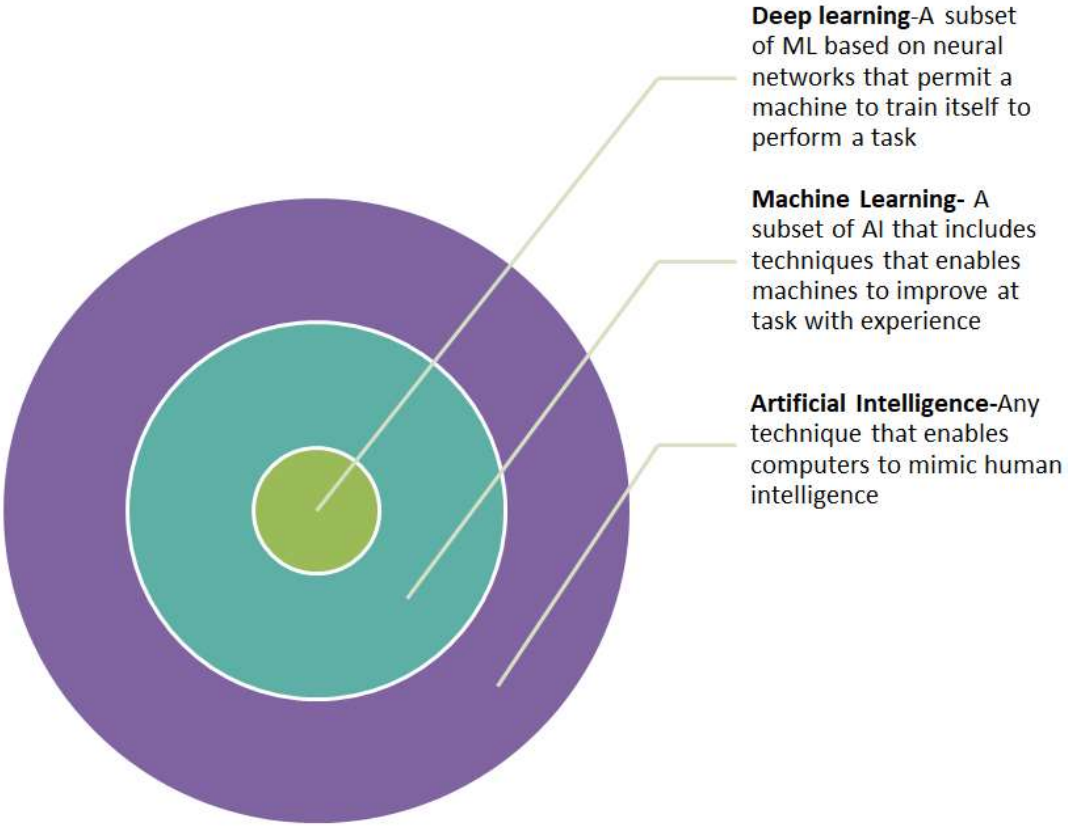


Image: Artificial Intelligence

Artificial intelligence

Artificial intelligence (AI) is the capability of a computer to imitate intelligent human behaviour. Through AI, machines can analyse images, comprehend speech, interact in natural ways, and make predictions using data.

© Edunet Foundation. All rights reserved.

AI concepts Algorithm

An *algorithm* is a sequence of calculations and rules used to solve a problem or analyse a set of data. It is like a flow chart, with step-by-step instructions for questions to ask, but written in math and programming code. An algorithm may describe how to determine whether a pet is a cat, dog, fish, bird, or lizard. Another far more complicated algorithm may describe how to identify a written or spoken language, analyse its words, translate them into a different language, and then check the translation for accuracy.

Machine learning

Machine learning (ML) is an AI technique that uses mathematical algorithms to create predictive models. An algorithm is used to parse data fields and to "learn" from that data by using patterns found within it to generate models. Those models are then used to make informed predictions or decisions about new data. The predictive models are validated against known data, measured by performance metrics selected for specific business scenarios, and then adjusted as needed. This process of learning and validation is called *training*. Through periodic retraining, ML models are improved over time.

Deep learning

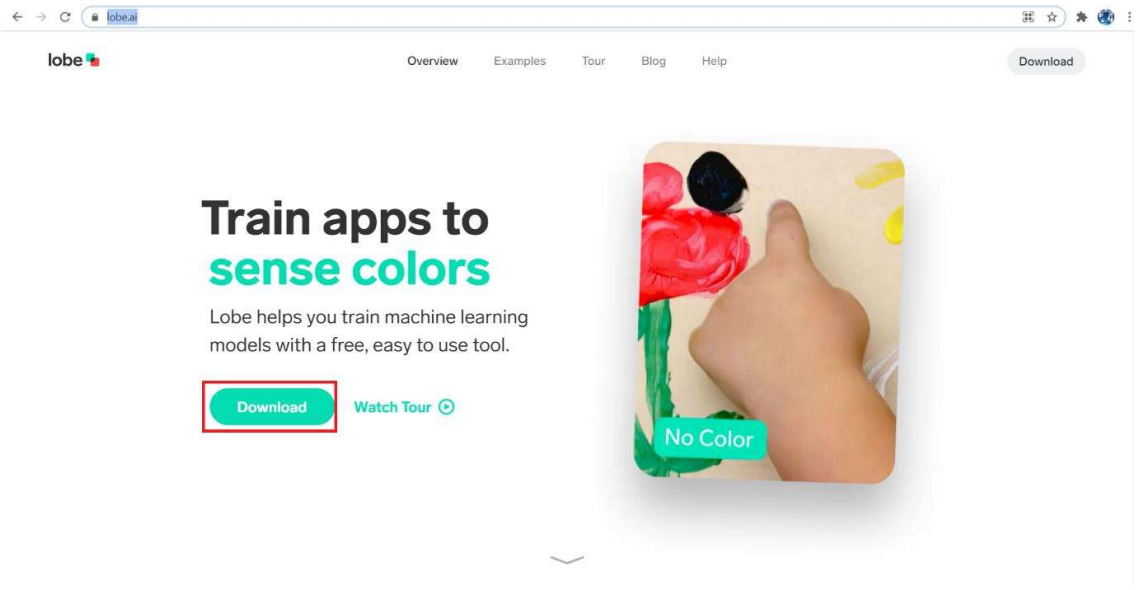
Deep learning is a type of ML that can determine for itself whether its predictions are accurate. It also uses algorithms to analyze data, but it does so on a larger scale than ML. Deep learning uses artificial neural networks, which consist of multiple layers of algorithms. Each layer looks at the incoming data, performs its own specialized analysis, and produces an output that other layers can understand. This output is then passed to the next layer, where a different algorithm does its own analysis, and so on.

With many layers in each neural network-and sometimes using multiple neural networks-a machine can learn through its own data processing. This requires much more data and much more computing power than ML.

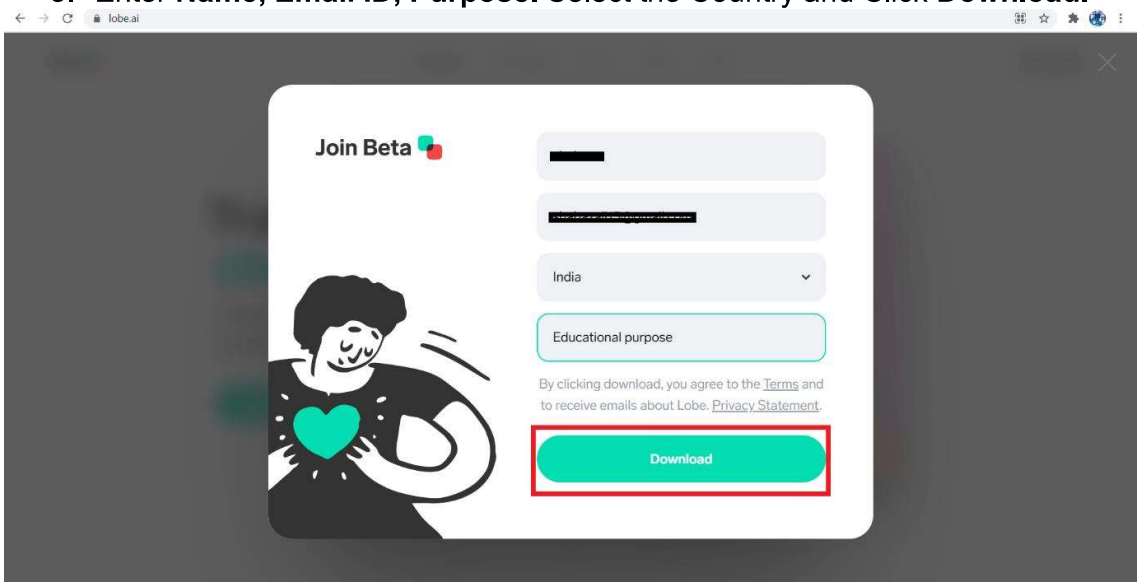
1.4 PRACTICAL : Image Recognition on Web Browser

1. Visit https://lobe.ai/ Website
2. Click on **Download**

© Edunet Foundation. All rights reserved.

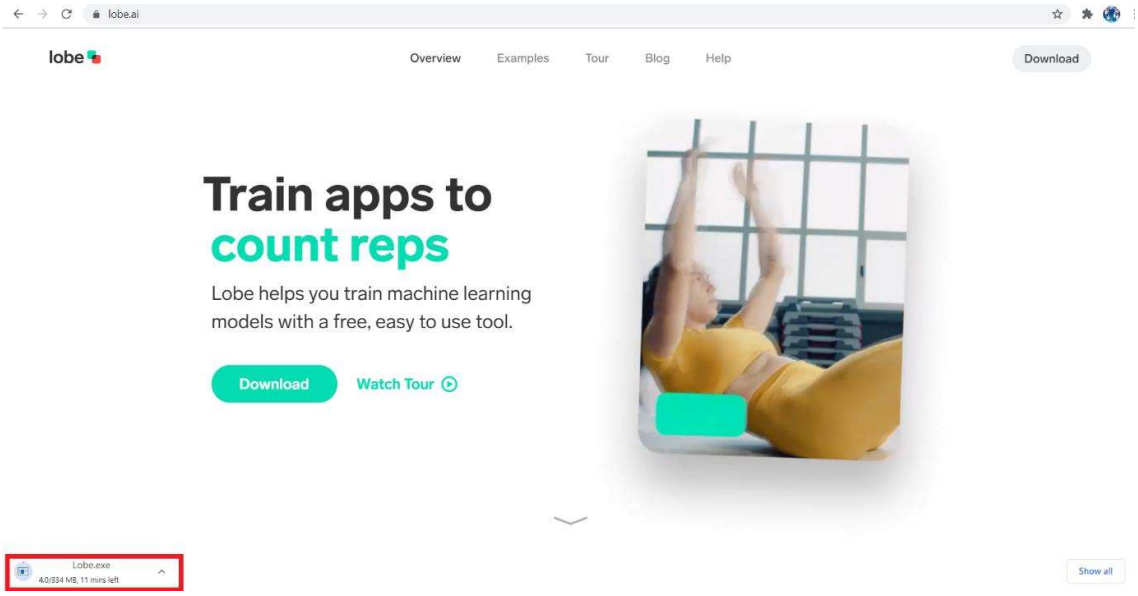


3. Enter **Name, Email ID, Purpose.** Select the Country and Click **Download.**

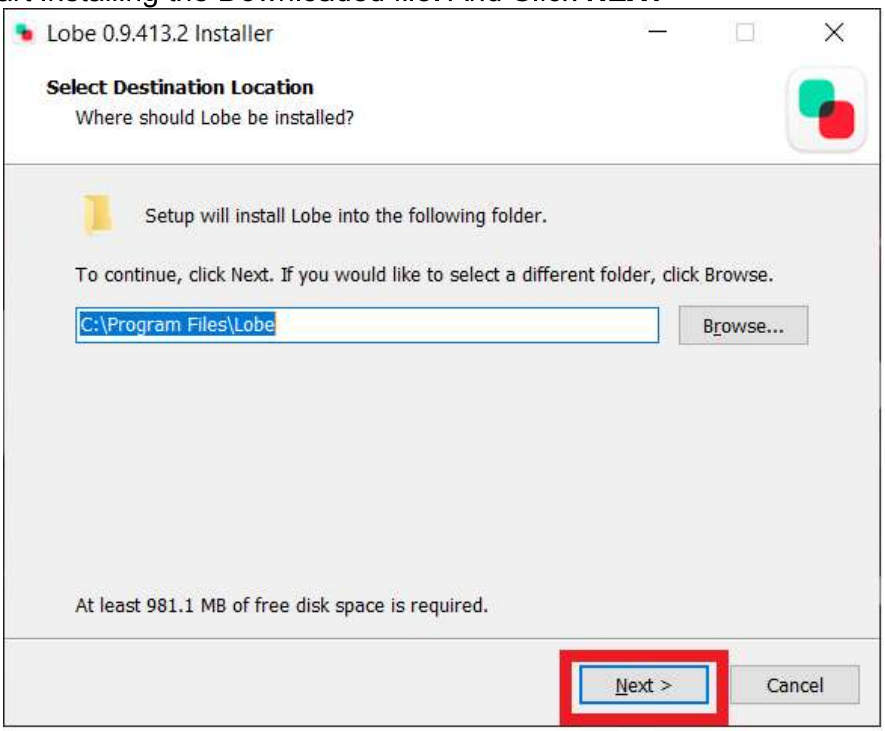


4. File is being **Downloaded**

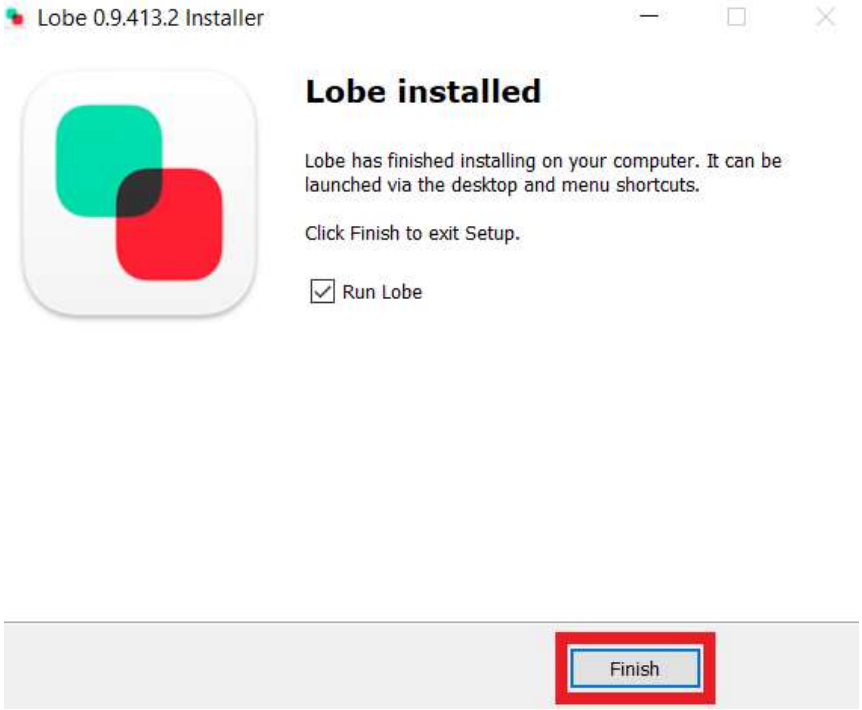
© Edunet Foundation. All rights reserved.



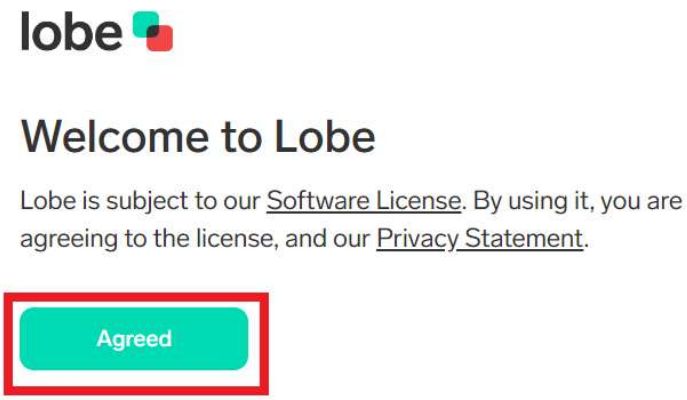
5. Start Installing the Downloaded file. And Click **NEXT**



6. Installation will take place. After that Click on **Finish**

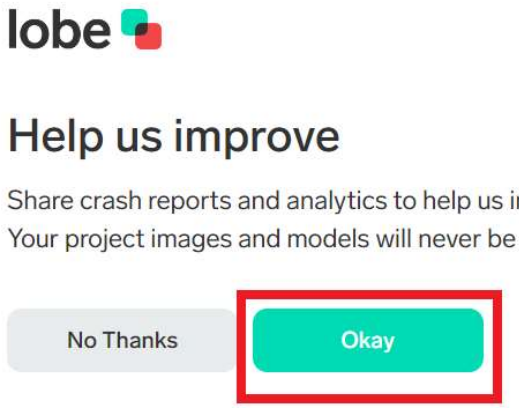


7. Click on **Agreed**

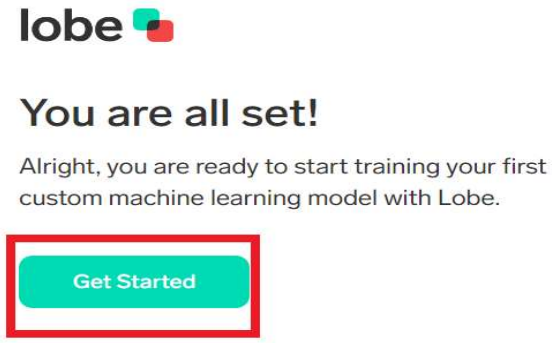


© Edunet Foundation. All rights reserved.

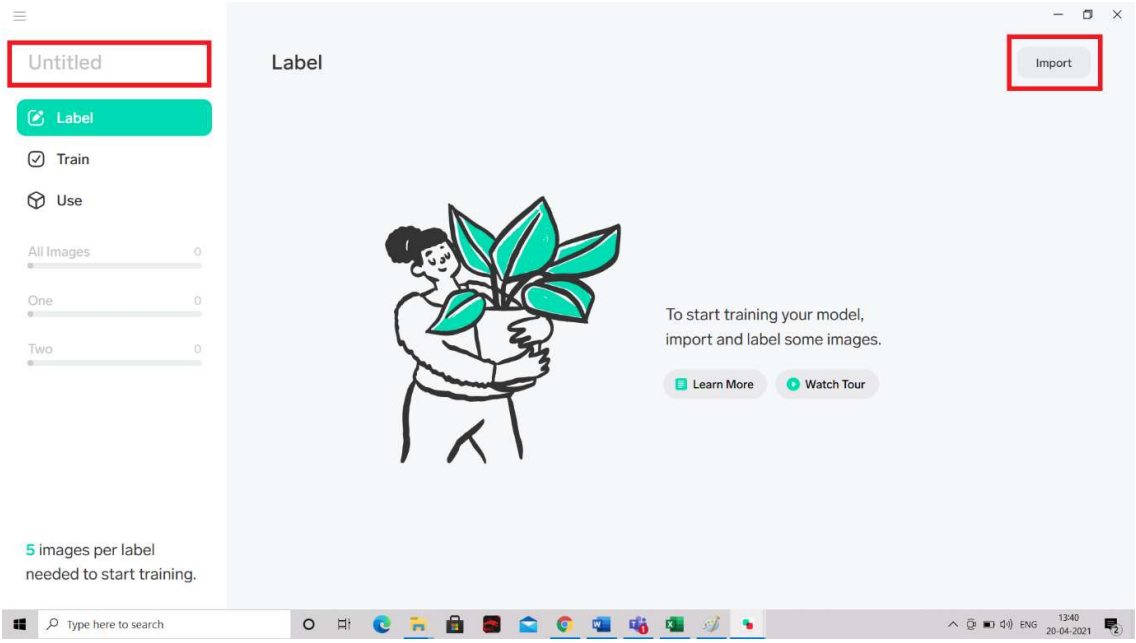
8. Click on **Okay**



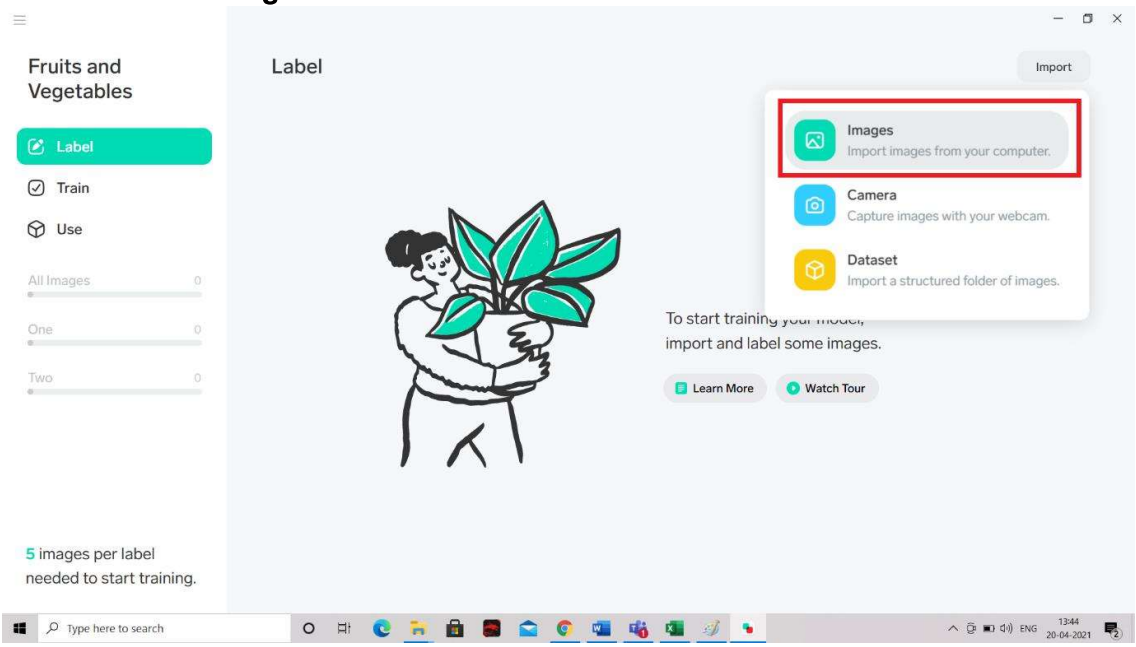
9. Click on **Get Started**



10. Name the Project. And Click on **Import.** To Upload Images from the System. (NOTE: My project is to train my system Difference between Fruits and Vegetables)

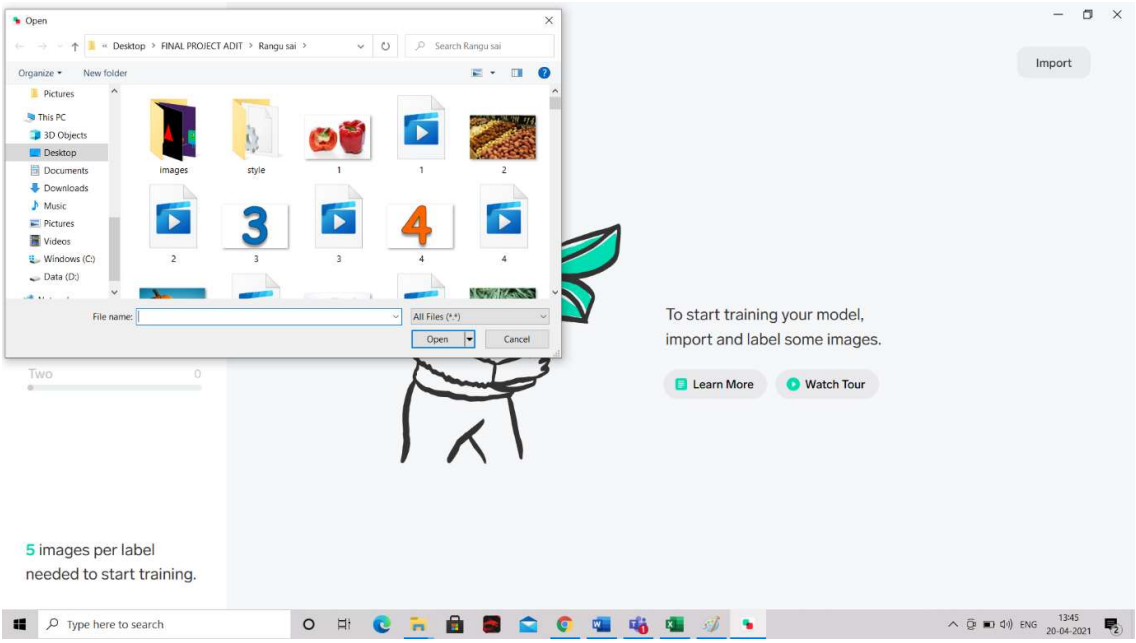


11. Click on **Images**

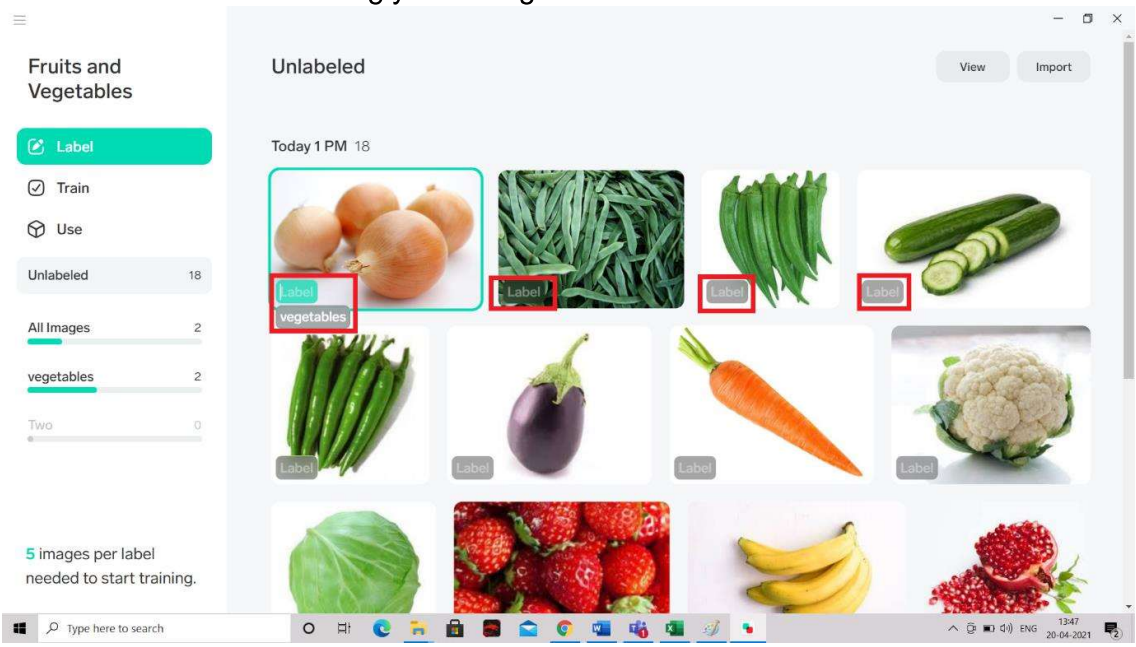


12. Select all you Project Images

© Edunet Foundation. All rights reserved.

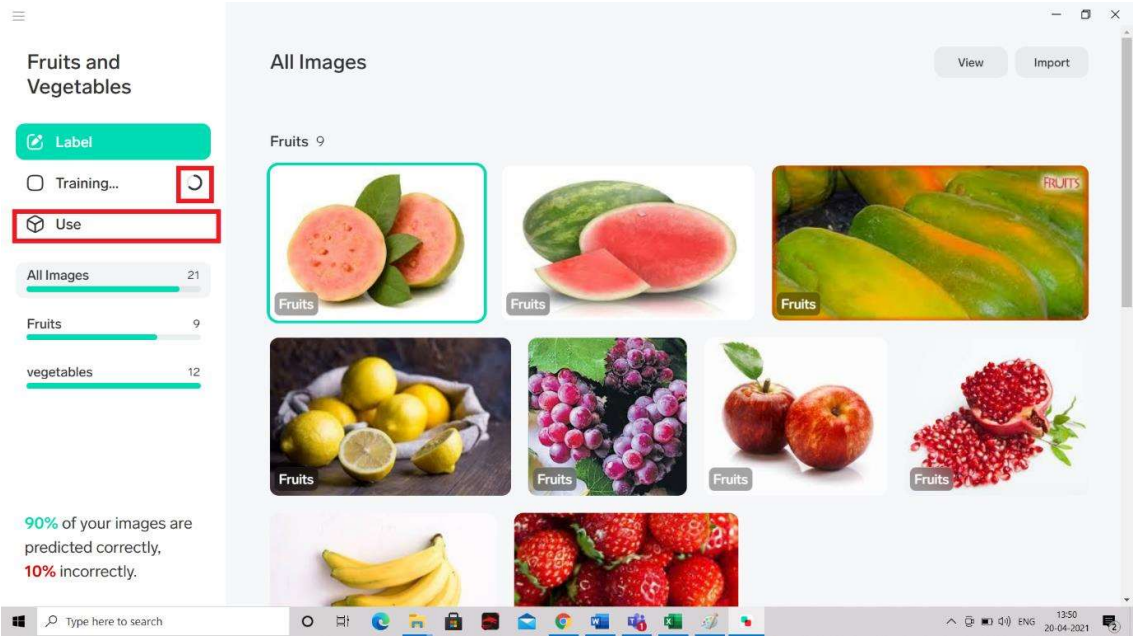


13. Label them Accordingly EX:- Vegetables and Fruits

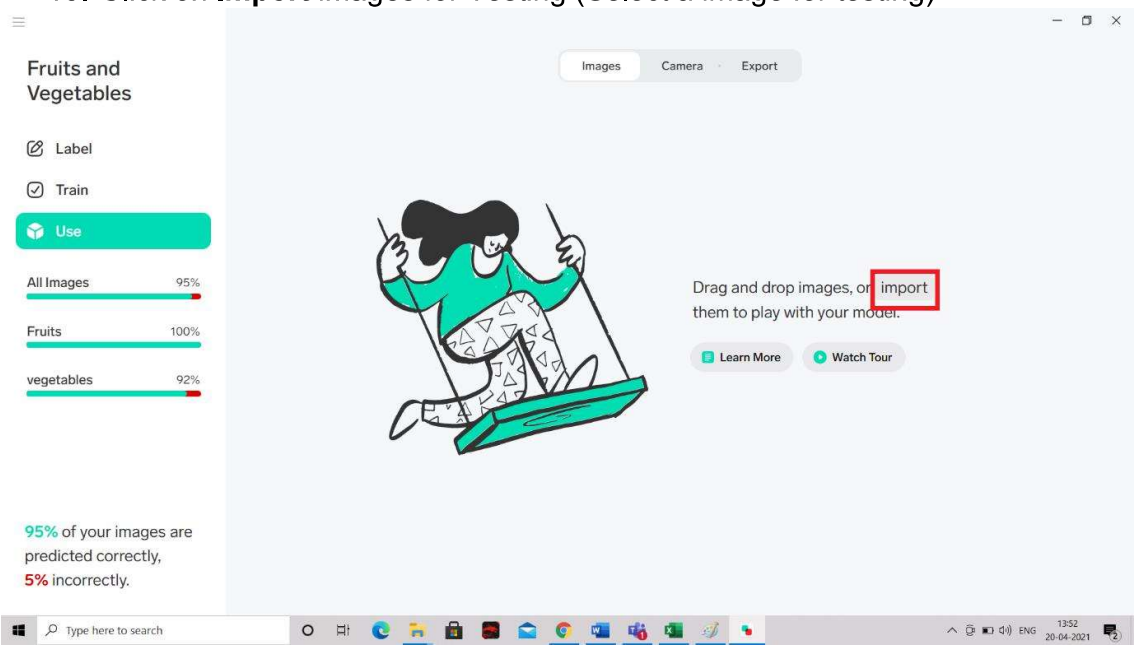


14. Training Will be done Automatically and Click on **USE**

© Edunet Foundation. All rights reserved.

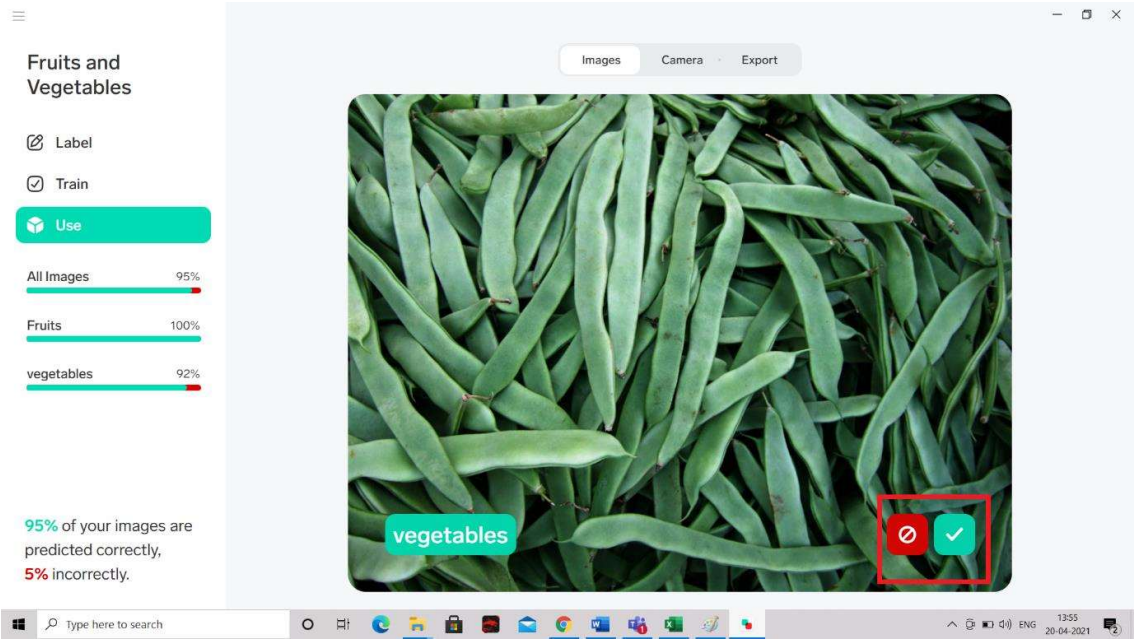


15. Click on **Import** images for Testing (Select a image for testing)



16.The Result will be Shown Down. If it is Correct Click on Right. If it is wrong Click on Wrong Which will help to learn more to the system

© Edunet Foundation. All rights reserved.

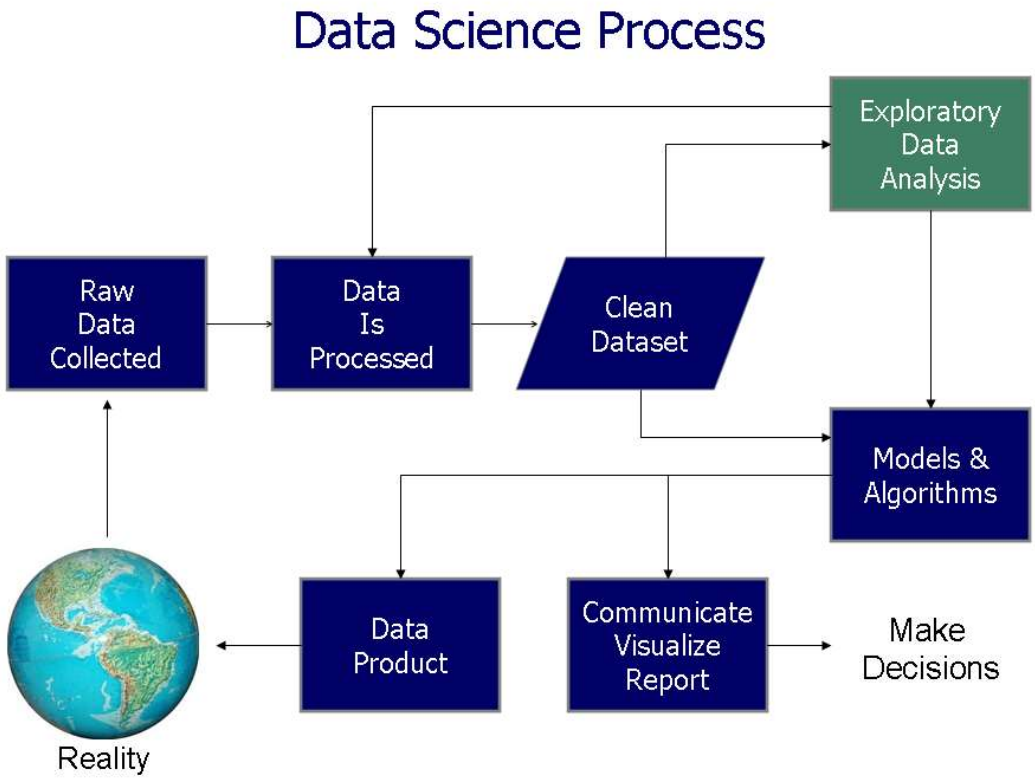


So, we have seen the history, market trends and opportunities in the very exciting field of Artificial Intelligence and the fundamental concepts in AI and machine learning. We also explored practically some of the applications of AI and Machine Learning like sentiment analysis, Natural Language Processing, and Image classification. We explored an overall understanding of what is AI, ML and its real-life applications.
Next, we will be discussing about basics of Data Analysis, Anaconda Software and key concepts in Python Programming language.

1.5 What is Data Analysis?

Data analysis is a process of inspecting, cleansing, transforming, and modelling data with the goal of discovering useful information, informing conclusions, and supporting decision-making. Data analysis has multiple facets and approaches, encompassing diverse techniques under a variety of names, and is used in different business, science, and social science domains. In today's business world, data analysis plays a role in making decisions more scientific and helping businesses operate more effectively.
Systematically applying statistical and/or logical techniques to describe and illustrate, condense and recap, and evaluate data.

© Edunet Foundation. All rights reserved.



Data science process flowchart
Ref : https://en.wikipedia.org/wiki/Data_analysis

The data analytics encompasses six phases that are data discovery, data aggregation, planning of the data models, data model execution, communication of the results, and operationalization.

© Edunet Foundation. All rights reserved.

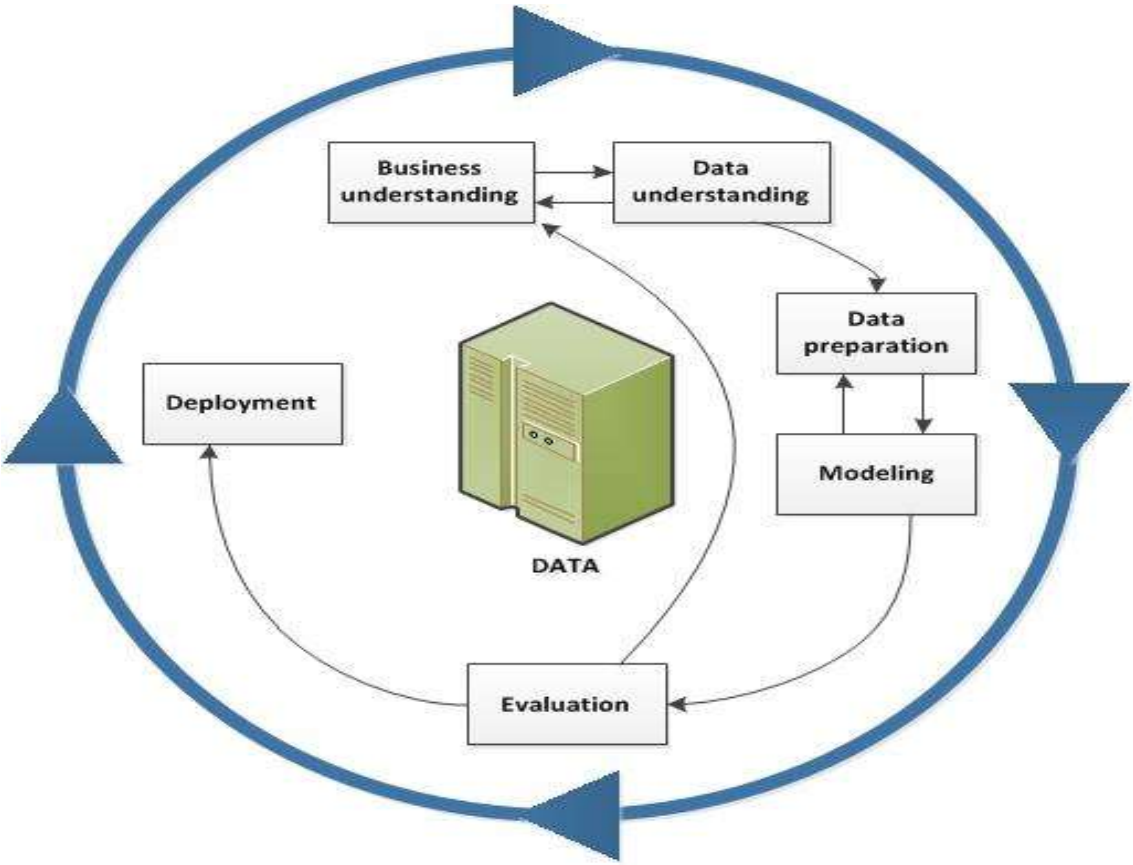


Fig: Data Analysis Process
Ref : https://www.ibm.com/support/knowledgecenter/

Let's understand the basic phases in Data Science Project

1. Business Understanding

- Data mining goals are defined.
- The fundamental requirement is to understand client and business objectives.
- Current data mining scenario factors in resources, constraints and assumptions should also be taken into the consideration.

2. Data Understanding

- In this stage, a sanity check is conducted to understand whether it is appropriate for data mining goals.
- The data is collected from various sources within the organization.
- It is a highly complex process since data and process from various sources are unlikely to match easily.

3. Data Preparation

- The data is production ready in this stage.

© Edunet Foundation. All rights reserved.

- The data from diverse sources should be nominated, cleaned, transformed, formatted, anonymized, and created.
- Data cleaning is a process to "clean" the data by smoothing noisy data and satisfying in missing values.

4. Modelling

- In this stage, mathematical models are used to determine the data patterns.
- Suitable modelling techniques need to be chosen for the prepared data set.
- After that, create a scenario to validate the model. Then run the model on the prepared data set.

5. Evaluation

- In this stage, patterns recognized are examined against business objectives.
- A go or no-go decision should be taken to move the model in the deployment phase.

6. Deployment

- In this stage, ship your data mining.
- Discoveries (model/reports) to every business operation.
- A thorough deployment plan, for shipping, maintenance, and monitoring of data mining discoveries is created. – Knowledge Management.

Let us understand the difference between Descriptive and Predictive analytics

Descriptive vs Predictive Analytics

- ***Descriptive Analytics:*** Describing or summarising the existing data using existing business intelligence tools to better understand what is going on or what has happened.

- ***Predictive Analytics:*** Emphasizes on predicting the possible outcome using statistical models and machine learning techniques.

© Edunet Foundation. All rights reserved.

Basis for Comparison	Descriptive Analytics	Predictive Analytics
Describes	What happened in the past? By using the stored data.	What might happen in the future? By using the past data and analyzing it.
Process Involved	Involves Data Aggregation and Data Mining.	Involves Statistics and forecast techniques.
Definition	The process of finding useful and important information by analyzing the huge data.	This process involves in forecasting the future of the company, which are very useful.
Data Volume	It involves in processing huge data that are stored in data warehouses. Limited to past data.	It involves analyzing large past data and then predicts the future using advance techniques.
Examples	Sales report, revenue of a company, performance analysis, etc.	Sentimental analysis, credit score analysis, forecast reports for a company, etc.
Accuracy	It provides accurate data in the reports using past data.	Results are not accurate, it will not tell you exactly what will happen but it will tell you what might happen in the future.
Approach	It allows the reactive approach	While this a proactive approach

Descriptive Analytics

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.

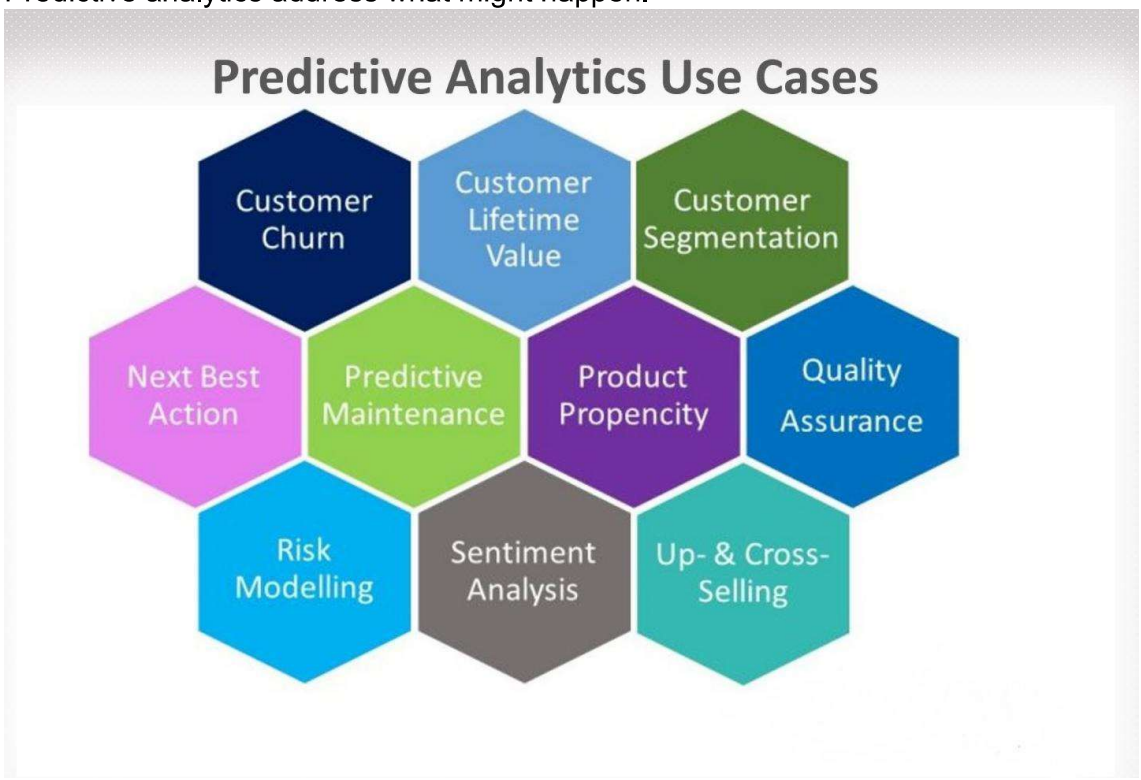
© Edunet Foundation. All rights reserved.

- 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.
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.

Predictive analytics

Predictive analytics address what might happen.



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 the 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.

© Edunet Foundation. All rights reserved.

Tools for Data Analysis

Many tools are available in the market, which make it easier for us :

1) To process and manipulate data

2) Analyze the relationships and correlations between data sets,

3) Helps to identify patterns and trends for interpretation

Some of them are:

- Python
- Tableau
- R Programming
- Power BI

1.6 Anaconda Software and Introduction to Python

Anaconda is a free open-source data science tool that focusses on the distribution of R and Python programming languages for data science and machine learning tasks. Anaconda aims at simplifying the data management and deployment of the same.
Anaconda is popular because it brings many of the tools used in data science and machine learning with just one install, so it's great for having short and simple setup. Anaconda also uses the concept of creating environments so as to isolate different libraries and versions.

Anaconda is a powerful data science platform for data scientists. The package manager of Anaconda is the conda which manages the package versions.
Anaconda is a tool that offers all the required package involved in data science at once. The programmers choose Anaconda for its ease of use.

Anaconda is written in Python, and the worthy information on Conda is unlike pip in Python, this package manager checks for the requirement of the dependencies and installs it if it is required. More importantly, warning signs are given if the dependencies already exist.

Conda very quickly installs the dependencies along with frequent updates. It facilitates creation and loading with equal speed along with easy environment switching. Anaconda is pre-built with more than 1500 Python or R data science packages. Anaconda has specific tools to collect data using Machine learning and Artificial Intelligence. Anaconda is indeed a tool used for developing, testing and training in one single system. The tool can be managed with any project as the environment is easily manageable.

© Edunet Foundation. All rights reserved.

The installation of Anaconda is very easy and most preferred by non-programmers who are data scientists.

Comparison Table Between Anaconda and Python

Parameter of Comparison	Anaconda	Python
Definition	Anaconda is the enterprise data science platform which distributes R and Python for machine learning and data science	Python is a high-level general-purpose programming language used for machine learning and data science
Category	Anaconda belongs to Data Science Tools	Python belongs to Computer Languages
Package Manager	Anaconda has conda has its package manager	Python has pip as the package manager
User Applications	Anaconda is primarily developed to support data science and machine learning tasks	Python is not only used in data science and machine learning but also a variety of applications in embedded systems, web development, and networking program
Package Management	Package manager conda allows Python as well as Non-Python library dependencies to install.	Package manager pip allows all the Python dependencies to install

Anaconda is a new distribution of the Python and R data science package. It was formerly known as Continuum Analytics. Anaconda has more than 100 new packages. This work environment, Anaconda is used for scientific computing, data science, statistical analysis, and machine learning.

Anaconda is a distribution of the Python and R programming languages for scientific computing (data science, machine learning applications, large-scale data processing, predictive analytics, etc.), that aims to simplify package management and deployment. The distribution includes data-science packages suitable for Windows, Linux, and macOS.

Anaconda Individual Edition contains conda and Anaconda Navigator, as well as Python and hundreds of scientific packages. When you installed Anaconda, you installed all these too.

© Edunet Foundation. All rights reserved.

Conda works on your command line interface such as Anaconda Prompt on Windows and terminal on macOS and Linux.

Navigator is a desktop graphical user interface that allows you to launch applications and easily manage conda packages, environments, and channels without using command-line commands.

Anaconda Navigator

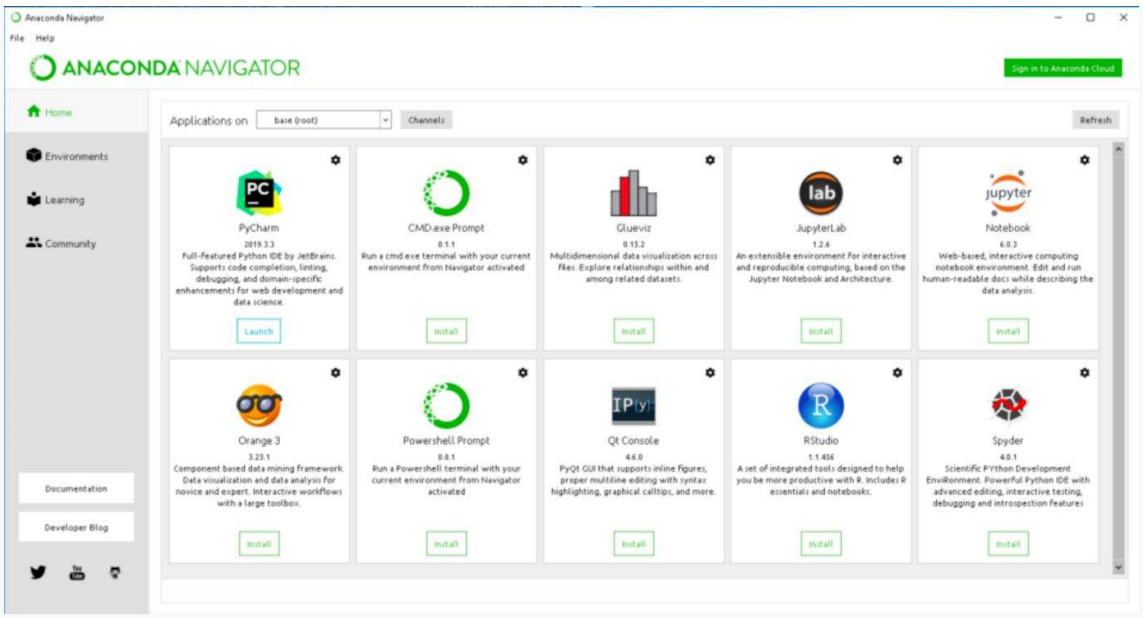


Fig : Anaconda Navigator Interface

Anaconda Navigator is a desktop graphical user interface (GUI) included in Anaconda distribution that allows users to launch applications and manage conda packages, environments and channels without using command-line commands. Navigator can search for packages on Anaconda Cloud or in a local Anaconda Repository, install them in an environment, run the packages and update them. It is available for Windows, macOS and Linux.

The following applications are available by default in Navigator;

- JupyterLab
- Jupyter Notebook
- QtConsole
- Spyder
- Glue
- Orange
- RStudio
- Visual Studio Code

© Edunet Foundation. All rights reserved.

Conda

Conda is an open source, cross-platform, language-agnostic package manager and environment management system that installs, runs, and updates packages and their dependencies. It was created for Python programs, but it can package and distribute software for any language (e.g., R), including multi-language projects. The conda package and environment manager is included in all versions of Anaconda, Miniconda, and Anaconda Repository.

Anaconda Cloud

Anaconda Cloud is a package management service by Anaconda where users can find, access, store and share public and private notebooks, environments, and conda and PyPI packages. Cloud hosts useful Python packages, notebooks and environments for a wide variety of applications. Users do not need to log in or to have a Cloud account, to search for public packages, download and install them.
Users can build new packages using the Anaconda Client command line interface (CLI), then manually or automatically upload the packages to Cloud.

Now let's get started with very popular programming language used in data science and variety of other tasks like website building, server-side programming etc. Named Python.

Introduction to Python

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built-in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse.
Released in 1991, Python is used for;

- Web development (server-side),
- Software development,
- Mathematics,
- System scripting.

© Edunet Foundation. All rights reserved.

What can Python do?

- Python can be used on a server to create web applications.
- Python can be used alongside software to create workflows.
- Python can connect to database systems. It can also read and modify files.
- Python can be used to handle big data and perform complex mathematics.
- Python can be used for rapid prototyping, or for production-ready software development.

Why Python?

- Python works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc).
- Python has a simple syntax like the English language.
- Python has syntax that allows developers to write programs with fewer lines than some other programming languages.
- Python runs on an interpreter system, meaning that code can be executed as soon as it is written. This means that prototyping can be very quick.
- Python can be treated in a procedural way, an object-oriented way or a functional way.

Python Syntax compared to other programming languages

- Python was designed for readability and has some similarities to the English language with influence from mathematics.
- Python uses new lines to complete a command, as opposed to other programming languages which often use semicolons or parentheses.
- Python relies on indentation, using whitespace, to define scope; such as the scope of loops, functions and classes. Other programming languages often use curly-brackets for this purpose.

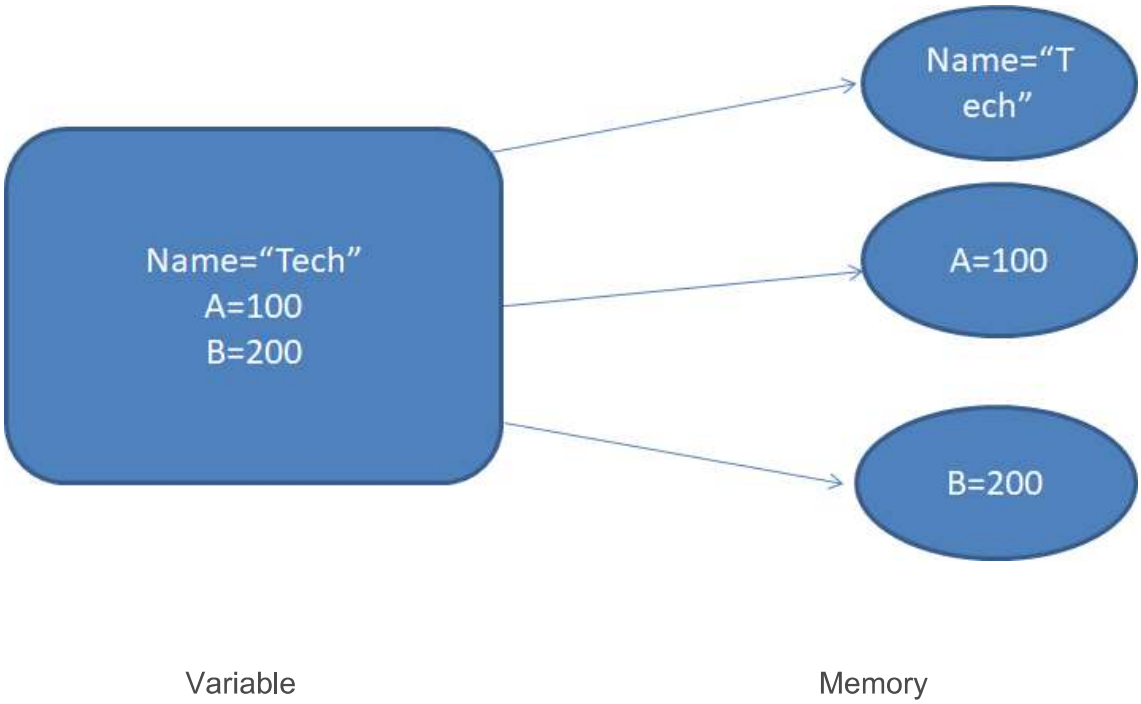
1.7 Variables in Python

Variables are containers for storing data values.

What Are Variables in Python?

Variables and data types in python as the name suggests are the values that vary. In a programming language, a variable is a memory location where you store a value. The value that you have stored may change in the future according to the specifications.

© Edunet Foundation. All rights reserved.



Variable in python is created as soon as a value is assigned to it. It does not need any additional commands to declare a variable in python.

After introducing to python lets install anaconda on Linux and windows operating systems to perform some data analysis and machine learning stuff.

1.8 Advanced Datatypes in Python

Data type defines the format, sets the upper & lower bounds of the data so that a program could use it appropriately.
In Python, we don't need to declare a variable with explicitly mentioning the data type. This feature is famously known as dynamic typing.
Python determines the type of a literal directly from the syntax at runtime. For example – the quotes mark the declaration of a string value, square brackets represent a list

© Edunet Foundation. All rights reserved.

and curly brackets for a dictionary. Also, the non-decimal numbers will get assigned to Integer type whereas the ones with a decimal point will be a float.

Below is the list of important data types that are commonly used in Python.

- Booleans
- Numbers
- Strings
- Bytes
- Lists
- Tuples
- Sets
- Dictionaries

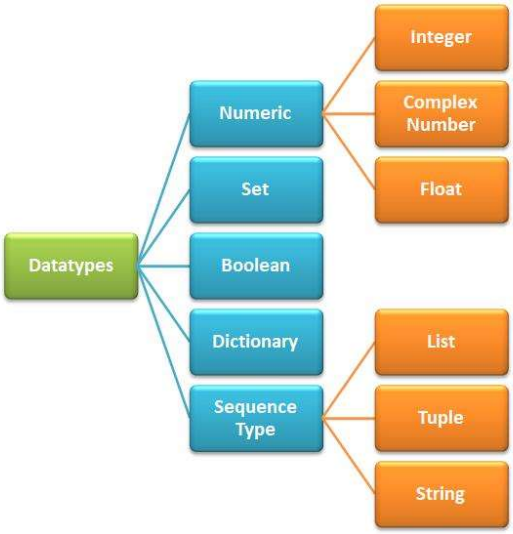


Fig: Datatypes in Python

In Python, numeric data type represents the data which has numeric value. Numeric value can be integer, floating number or even complex numbers. These values are defined as int, float and complex class in Python.

- **Integers –** This value is represented by int class. It contains positive or negative whole numbers (without fraction or decimal). In Python there is no limit to how long an integer value can be.
- **Float –** This value is represented by float class. It is a real number with floating point representation. It is specified by a decimal point. Optionally, the character e or E followed by a positive or negative integer may be appended to specify scientific notation.

© Edunet Foundation. All rights reserved.

- **Complex Numbers –** Complex number is represented by complex class. It is specified as (real part) + (imaginary part)j. For example – 2+3j

```
a = 5
print("Type of a: ", type(a))

b = 5.0
print("\nType of b: ", type(b))

c = 2 + 4j
print("\nType of c: ", type(c))
```

```
Type of a:  <class 'int'>

Type of b:  <class 'float'>

Type of c:  <class 'complex'>
```

1.9 Condition & Loop in Python

Condition

Python supports the usual logical conditions from mathematics:

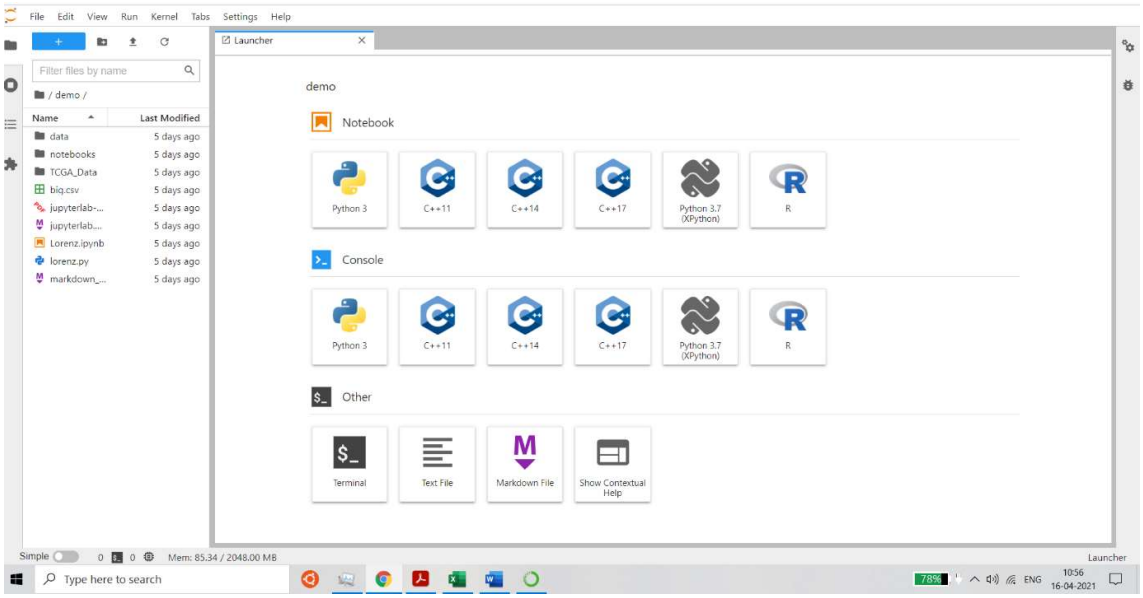
- Equals: a == b
- Not Equals: a != b
- Less than: a < b
- Less than or equal to: a <= b
- Greater than: a > b
- Greater than or equal to: a >= b

These conditions can be used in several ways, most commonly in "if statements" and loops.

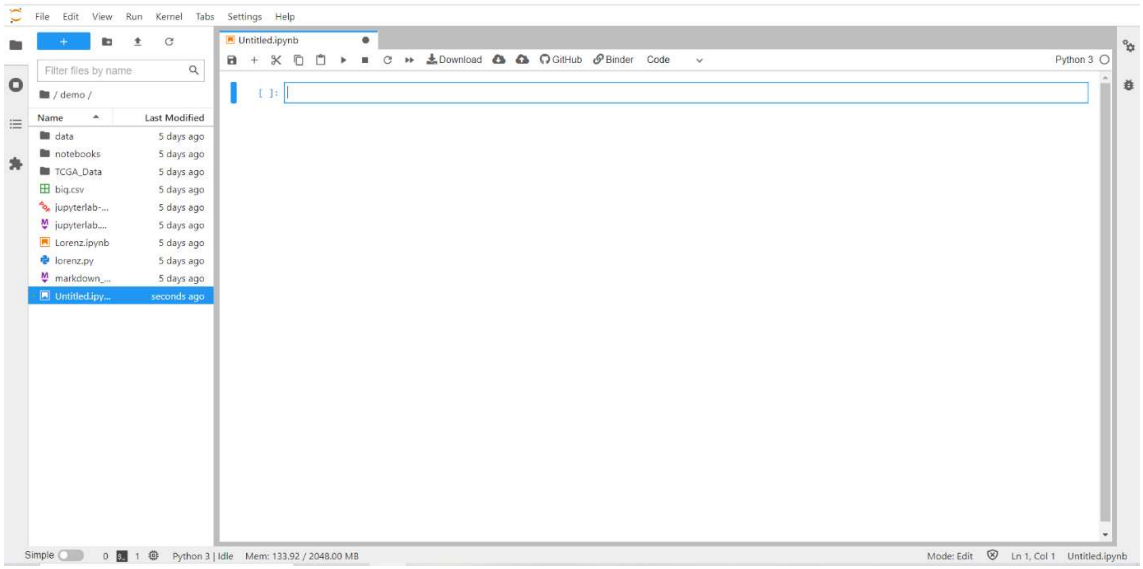
An "if statement" is written by using the if keyword.

1. Open Jupyter Notebook

© Edunet Foundation. All rights reserved.



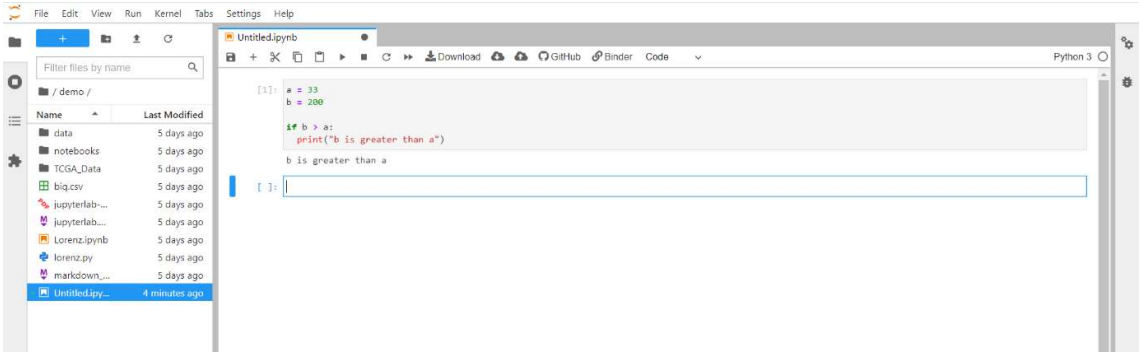
2. Go for Python



3. Try the code for Condition.

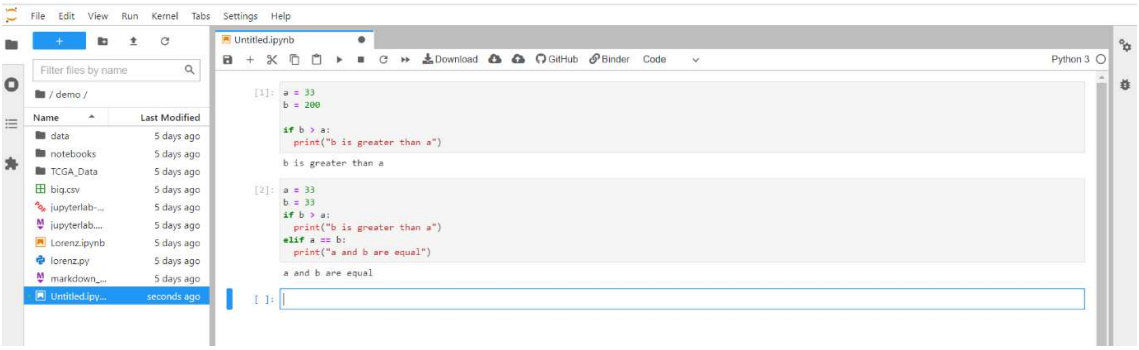
(Take care of the indentation (whitespace at the beginning of a line) to define scope in the code)

© Edunet Foundation. All rights reserved.



Elif

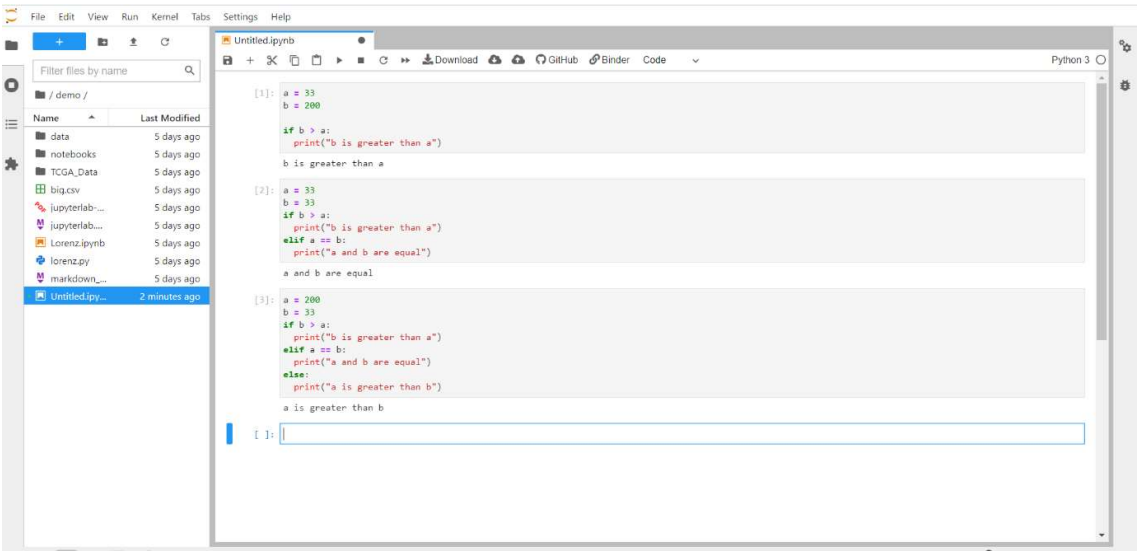
The elif keyword is python's way of saying "if the previous conditions were not true, then try this condition".



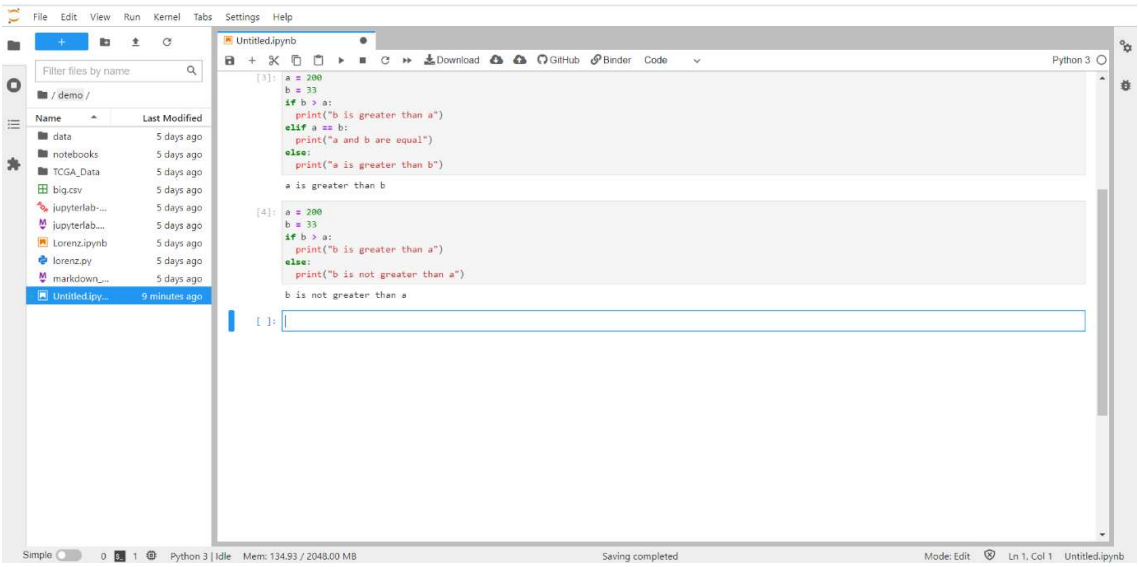
Else

The else keyword catches anything which isn't caught by the preceding conditions.

© Edunet Foundation. All rights reserved.



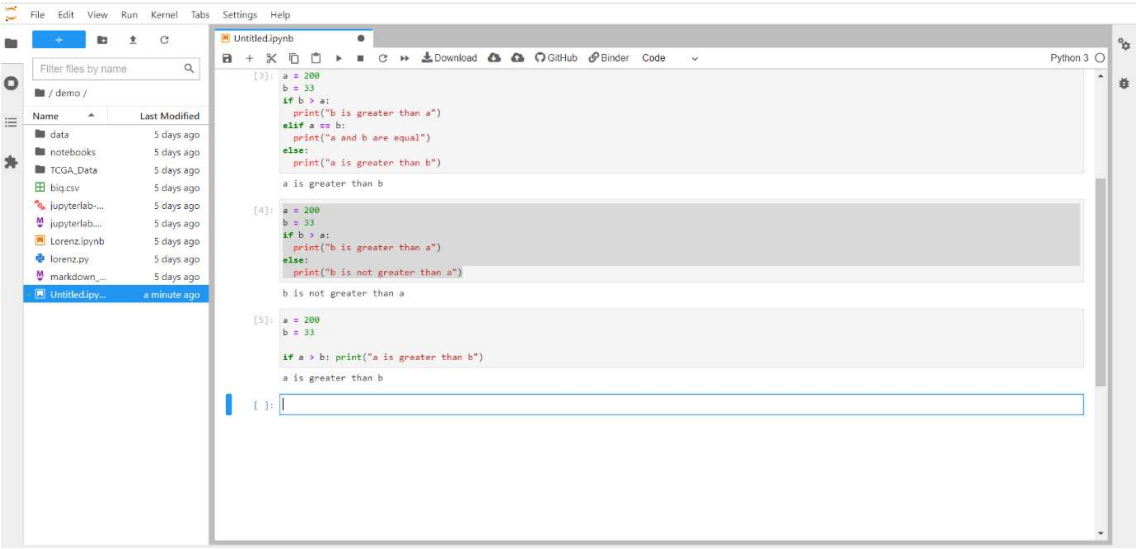
else without the elif



If

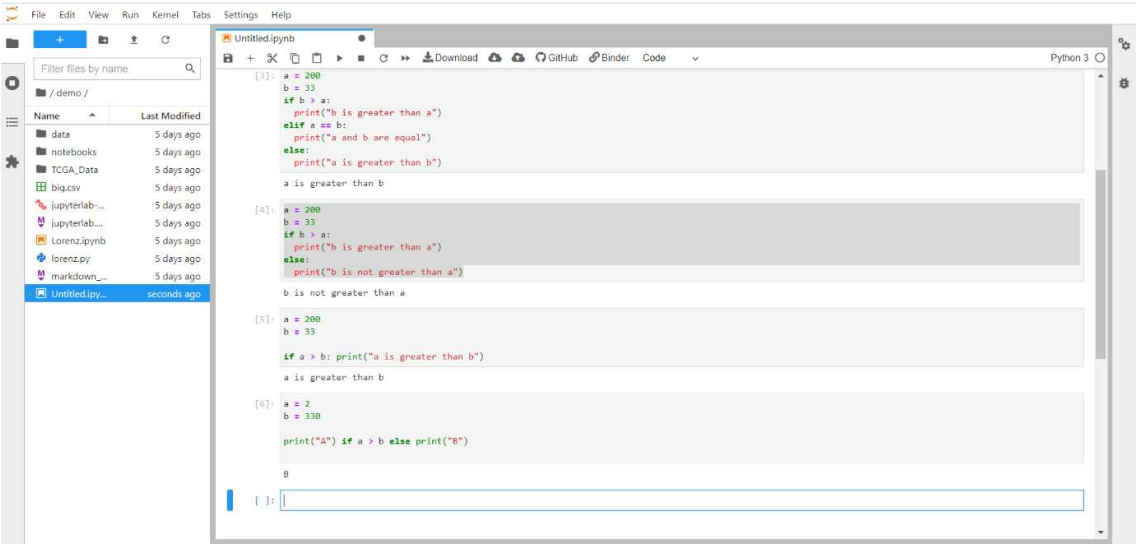
If you have only one statement to execute, you can put it on the same line as the if statement.

© Edunet Foundation. All rights reserved.



If ... Else

If you have only one statement to execute, one for if, and one for else, you can put it all on the same line:

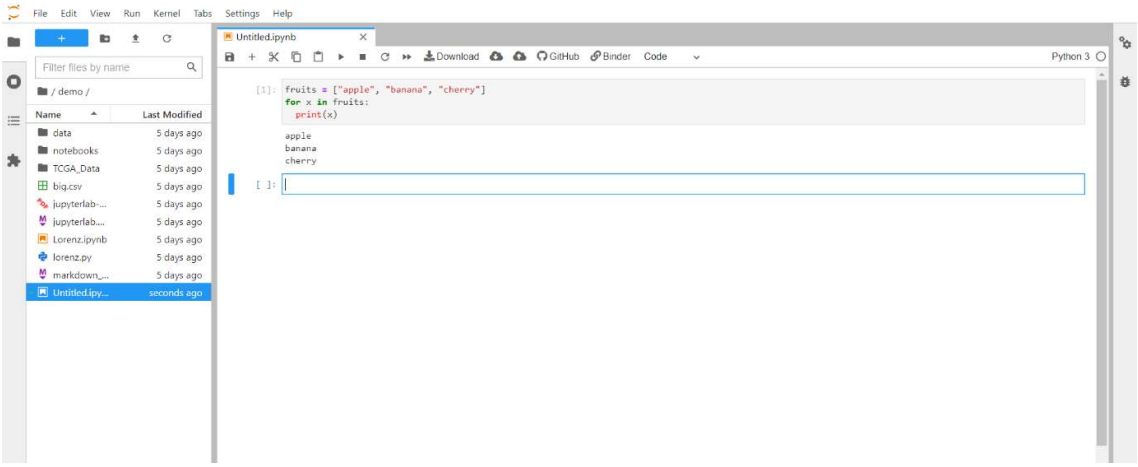


Loops

A for loop is used for iterating over a sequence (that is either a list, a tuple, a dictionary, a set, or a string). This is less like the for keyword in other programming languages, and works more like an iterator method as found in other object-orientated programming languages.

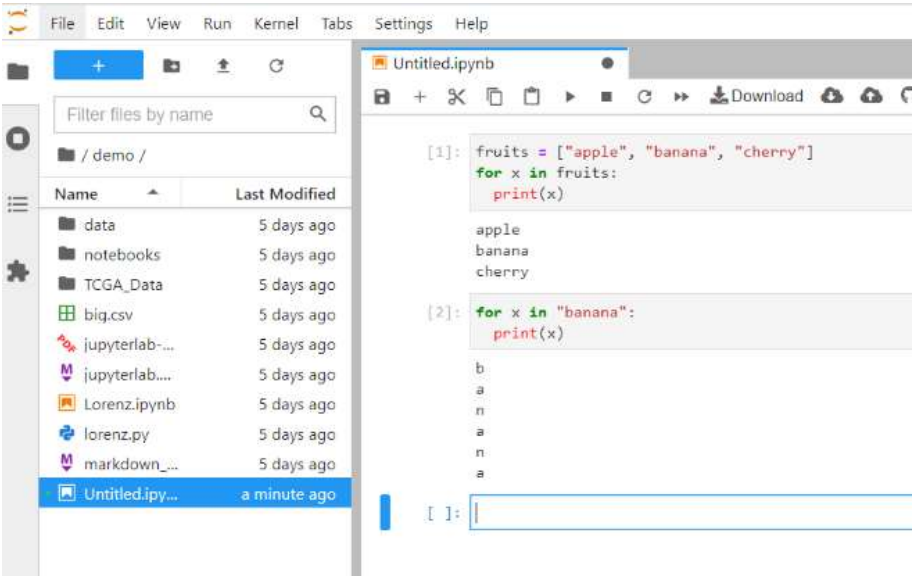
© Edunet Foundation. All rights reserved.

With the for loop we can execute a set of statements, once for each item in a list, tuple, set etc.



Looping Through a String

Even strings are iterable objects, they contain a sequence of characters:



break Statement

With the break statement we can stop the loop before it has looped through all the items:

© Edunet Foundation. All rights reserved.

```
]: fruits = ["apple", "banana", "cherry"]
for x in fruits:
  print(x)

apple
banana
cherry

]: for x in "banana":
  print(x)

b
a
n
a
n
a

]: fruits = ["apple", "banana", "cherry"]
for x in fruits:
  print(x)
  if x == "banana":
    break

apple
banana

]:
```

continue Statement

With the continue statement we can stop the current iteration of the loop, and continue with the next:

```
[4]: fruits = ["apple", "banana", "cherry"]
for x in fruits:
  if x == "banana":
    continue
  print(x)

apple
cherry

[ ]:
```

© Edunet Foundation. All rights reserved.

range() Function

To loop through a set of code a specified number of times, we can use the range() function.

The range() function returns a sequence of numbers, starting from 0 by default, and increments by 1 (by default), and ends at a specified number.

```
[5]: for x in range(6):
       print(x)

     0
     1
     2
     3
     4
     5

[ ]:
```

Else in For Loop

The `else` keyword in a `for` loop specifies a block of code to be executed when the loop is finished:

```
[6]: for x in range(6):
       print(x)
     else:
       print("Finally finished!")

     0
     1
     2
     3
     4
     5
     Finally finished!
```

Nested Loops

A nested loop is a loop inside a loop.

The "inner loop" will be executed one time for each iteration of the "outer loop":

© Edunet Foundation. All rights reserved.

```
[7]: adj = ["red", "big", "tasty"]
     fruits = ["apple", "banana", "cherry"]

     for x in adj:
       for y in fruits:
         print(x, y)

     red apple
     red banana
     red cherry
     big apple
     big banana
     big cherry
     tasty apple
     tasty banana
     tasty cherry
```

pass Statement

`for` loops cannot be empty, but if you for some reason have a `for` loop with no content, put in the `pass` statement to avoid getting an error.

```
[8]: for x in [0, 1, 2]:
       pass

[ ]:
```

1.10 Python Libraries

Python Libraries are a set of useful functions that eliminate the need for writing codes from scratch.
There are over 137,000 python libraries present today.
Python libraries play a vital role in developing machine learning, data science, data visualization, image and data manipulation applications and more.

What is a Library?

A library is a collection of pre-combined codes that can be used iteratively to reduce the time required to code. They are particularly useful for accessing the pre-written frequently used codes, instead of writing them from scratch every single time. Similar to the physical libraries, these are a collection of reusable resources, which means every library has a root source. This is the foundation behind the numerous open-source libraries available in Python.

© Edunet Foundation. All rights reserved.

Some basic Python Libraries

1. Scikit- learn

It is a free software machine learning library for the Python programming language and can be effectively used for a variety of applications which include classification, regression, clustering, model selection, naive Bayes', grade boosting, K-means, and pre-processing.

2. NumPy

NumPy is one of the fundamental packages for Python providing support for large multidimensional arrays and matrices along with a collection of high-level mathematical functions to execute these functions swiftly. NumPy relies on BLAS and LAPACK for efficient linear algebra computations. NumPy can also be used as an efficient multi-dimensional container of generic data.

3. TensorFlow

The most popular deep learning framework, TensorFlow is an open-source software library for high-performance numerical computation. It is an iconic math library and is also used for machine learning and deep learning algorithms. Tensorflow was developed by the researchers at the Google Brain team within Google AI organisation, and today it is being used by researchers for machine learning algorithms, and by physicists for complex mathematical computations.

4. Keras

It is an open-source neural network library written in Python designed to enable fast experimentation with deep neural networks. With deep learning becoming ubiquitous, Keras becomes the ideal choice as it is API designed for humans and not machines according to the creators.

5. Pandas

It is an open-source, BSD licensed library. Pandas enable the provision of easy data structure and quicker data analysis for Python. For operations like data analysis and modelling, Pandas makes it possible to carry these out without needing to switch to more domain-specific language like R.

6. OpenCV Python

Open Source Computer Vision or OpenCV is used for image processing. It is a Python package that monitors overall functions focused on instant computer vision. OpenCV provides several inbuilt functions, with the help of this you can learn Computer Vision. It allows both read and write images at the same time. Objects such as faces, trees, etc., can be diagnosed in any video or image.

1.11 Numpy Library

NumPy is an open source Python library used for working with arrays. It also has functions for working in domain of linear algebra, fourier transform, and matrices. NumPy stands for Numerical Python.

© Edunet Foundation. All rights reserved.

Why NumPy?

In Python we have lists that serve the purpose of arrays, but they are slow to process. NumPy aims to provide an array object that is up to 50x faster than traditional Python lists. The array object in NumPy is called `ndarray`, it provides a lot of supporting functions that make working with `ndarray` very easy. Arrays are very frequently used in data science, where speed and resources are very important.

Numpy arrays are stored in a single contiguous (continuous) block of memory. There are two key concepts relating to memory: dimensions and **strides**.

Firstly, many Numpy functions use strides to make things fast. Examples include integer slicing (e.g. X[1,0:2]) and broadcasting. Understanding strides helps us better understand how Numpy operates.

Secondly, we can directly use strides to make our own code faster. This can be particularly useful for data pre-processing in machine learning.

NumPy is a Python library and is written partially in Python, but most of the parts that require fast computation are written in C or C++.

Installing Numpy Module

1. To install numpy library,

   pip install numpy

2. To verify the libraries already installed,

   pip list

Create a NumPy ndarray Object

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.

1. Create a NumPy ndarray Object

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.

© Edunet Foundation. All rights reserved.

```
In [3]: import numpy as np

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

        print(arr)

        print(type(arr))
```

```
[1 2 3 4 5]
<class 'numpy.ndarray'>
```

Min, Max & Everything in between

```
In [4]: # import numpy library
        import numpy

        # creating a numpy array of integers
        arr = numpy.array([1, 5, 4, 8, 3, 7])

        # finding the maximum and
        # minimum element in the array
        max_element = numpy.max(arr)
        min_element = numpy.min(arr)

        # printing the result
        print('maximum element in the array is: ',
            max_element)
        print('minimum element in the array is: ',
            min_element)
```

```
maximum element in the array is:  8
minimum element in the array is:  1
```

Random Number Generator

1. Generate Random Number

NumPy offers the random module to work with random numbers.

Ex: Generate a random integer from 0 to 100 :

© Edunet Foundation. All rights reserved.

```
In [5]: from numpy import random
        x = random.randint(100)
        print(x)
```

```
89
```

```
In [6]: from numpy import random
        x = random.randint(100)
        print(x)
```

```
45
```

2. Generate Random Float

The random module's rand() method returns a random float between 0 and 1.

Ex: Generate a random float from 0 to 1 :

```
In [7]: from numpy import random
        x = random.rand()
        print(x)
```

```
0.43203375825754287
```

3. Generate Random Array

In NumPy we work with arrays, and you can use the two methods from the above examples to make random arrays.

Integers

The randint() method takes a size parameter where you can specify the shape of an array.

Ex: Generate a 1-D array containing 5 random integers from 0 to 100

© Edunet Foundation. All rights reserved.

```
In [8]: from numpy import random

        x=random.randint(100, size=(5))

        print(x)

        [71 53 93 89 47]
```

Ex: Generate a 2-D array with 3 rows, each row containing 5 random integers from 0 to 100

```
In [9]: from numpy import random

        x = random.randint(100, size=(3, 5))

        print(x)

        [[75  8 37 45  4]
         [39 17 51 75 29]
         [66 61 48 52 83]]
```

1.12 Matrix Multiplication in Numpy

Matrix multiplication is an operation that takes two matrices as input and produces single matrix by multiplying rows of the first matrix to the column of the second matrix.In matrix multiplication make sure that the number of rows of the first matrix should be equal to the number of columns of the second matrix.

Ex;

```
Input:matrix1 = ([1, 2, 3],
                 [3, 4, 5],
                 [7, 6, 4])
      matrix2 = ([5, 2, 6],
                 [5, 6, 7],
                 [7, 6, 4])

Output : [[36 32 32]
          [70 60 66]
          [93 74 100]]
```

© Edunet Foundation. All rights reserved.

```
In [14]: import numpy as np

         # input two matrices
         mat1 = ([1, 6, 5],[3 ,4, 8],[2, 12, 3])
         mat2 = ([3, 4, 6],[5, 6, 7],[6,56, 7])

         # This will return dot product
         res = np.dot(mat1,mat2)


         # print resulted matrix
         print(res)
```

```
[[ 63 320  83]
 [ 77 484 102]
 [ 84 248 117]]
```

1.13 NumPy Statistical Functions

Statistics is concerned with collecting and then analyzing that data. It includes methods for collecting the samples, describing the data, and then concluding that data. NumPy is the fundamental package for scientific calculations and hence goes hand-in-hand for NumPy statistical Functions.

NumPy contains various statistical functions that are used to perform statistical data analysis. These statistical functions are useful when finding a maximum or minimum of elements. It is also used to find basic statistical concepts like standard deviation, variance, etc.

© Edunet Foundation. All rights reserved.

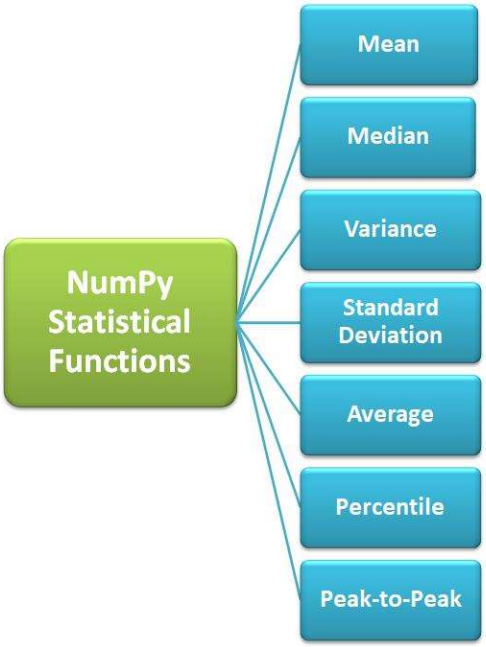


Fig: Numpy Statistical Functions

NumPy Statistical Functions

NumPy is equipped with the following statistical functions, Let us see one by one:

1. np.amin()- This function determines the minimum value of the element along a specified axis.
2. np.amax()- This function determines the maximum value of the element along a specified axis.
3. np.mean()- It determines the mean value of the data set.
4. np.median()- It determines the median value of the data set.
5. np.std()- It determines the standard deviation
6. np.var() – It determines the variance.
7. np.ptp()- It returns a range of values along an axis.
8. np.average()- It determines the weighted average
9. np.percentile()- It determines the nth percentile of data along the specified axis.

1. Finding maximum and minimum of array in NumPy

NumPy **np.amin()**and **np.amax()**functions are useful to determine the minimum and maximum value of array elements along a specified axis.

© Edunet Foundation. All rights reserved.

```
In [17]: import numpy as np
         arr= np.array([[1,23,78],[98,60,75],[79,25,48]])
         print(arr)
         #Minimum Function
         print(np.amin(arr))
         #Maximum Function
         print(np.amax(arr))
```

```
[[ 1 23 78]
 [98 60 75]
 [79 25 48]]
1
98
```

2. Finding Mean, Median, Standard Deviation and Variance in NumPy

Mean

Mean is the sum of the elements divided by the number of elements and is given by the following formula:

$$\bar{x} = \frac{1}{n}(x_1 + x_2 + \ldots + x_n)$$

It calculates the mean by adding all the items of the arrays and then divides it by the number of elements. We can also mention the axis along which the mean can be calculated.

```
In [18]: import numpy as np
         a = np.array([5,6,7])
         print(a)
         print(np.mean(a))
```

```
[5 6 7]
6.0
```

Median

Median is the middle element of the array. The formula differs for odd and even sets. The median for a given set of data with n elements (observations) is given by

$$\text{Odd data} = \left(\frac{n+1}{2}\right)^{\text{th}} \text{ observation}, \quad \text{Even data} = \frac{\frac{n}{2}^{\text{th}}\text{obs.} + \left(\frac{n}{2}+1\right)^{\text{th}}\text{obs.}}{2}$$

© Edunet Foundation. All rights reserved.

It can calculate the median for both one-dimensional and multi-dimensional arrays. Median separates the higher and lower range of data values.

```
In [19]: import numpy as np
         a = np.array([5,6,7])
         print(a)
         print(np.median(a))

         [5 6 7]
         6.0
```

Standard Deviation

Standard deviation is the square root of the average of square deviations from mean. The formula for standard deviation is:

$$\mathrm{SD} = \sqrt{\frac{\sum |x - \bar{x}|^2}{n}}$$

```
In [20]: import numpy as np
         a = np.array([5,6,7])
         print(a)
         print(np.std(a))

         [5 6 7]
         0.816496580927726
```

Variance

Variance is the average of the squared differences from the mean. Following is the formula for the same:

$$\sigma^2 = \frac{\sum (x_r - \overline{x})^2}{n}$$

© Edunet Foundation. All rights reserved.

```
In [21]: import numpy as np
         a = np.array([5,6,7])
         print(a)
         print(np.var(a))

         [5 6 7]
         0.6666666666666666
```

1.14 Percentile & Interquartile in Numpy

Quartiles :

A quartile is a type of quantile. The first quartile (Q1), is defined as the middle number between the smallest number and the median of the data set, the second quartile (Q2) – median of the given data set while the third quartile (Q3), is the middle number between the median and the largest value of the data set.

Algorithm to find Quartiles:

Quartiles are calculated by the help of the median. If the number of entries is an even number i.e. of the form 2n, then, first quartile (Q1) is equal to the median of the **n** smallest entries and the third quartile (Q3) is equal to the median of the **n** largest entries.

If the number of entries is an odd number i.e. of the form (2n + 1), then

- the first quartile (Q1) is equal to the median of the **n** smallest entries
- the third quartile (Q3) is equal to the median of the **n** largest entries
- the second quartile(Q2) is the same as the ordinary median.

Range: It is the difference between the largest value and the smallest value in the given data set.

Interquartile Range : The interquartile range (IQR), also called as midspread or middle 50%, or technically H-spread is the difference between the third quartile (Q3) and the first quartile (Q1). It covers the center of the distribution and contains 50% of the observations. IQR = Q3 – Q1

Uses;

- The interquartile range has a breakdown point of 25% due to which it is often preferred over the total range.
- The IQR is used to build box plots, simple graphical representations of a probability distribution.
- The IQR can also be used to identify the outliers in the given data set.
- The IQR gives the central tendency of the data.

© Edunet Foundation. All rights reserved.

Decision Making

- The data set having a higher value of interquartile range (IQR) has more variability.
- The data set having a lower value of interquartile range (IQR) is preferable.

Interquartile range using numpy.median

```
In [22]: import numpy as np

         data = [32, 36, 46, 47, 56, 69, 75, 79, 79, 88, 89, 91, 92, 93, 96, 97,
                 101, 105, 112, 116]

         # First quartile (Q1)
         Q1 = np.median(data[:10])

         # Third quartile (Q3)
         Q3 = np.median(data[10:])

         # Interquartile range (IQR)
         IQR = Q3 - Q1

         print(IQR)

         34.0
```

Interquartile range using numpy.percentile

```
In [23]: # Import numpy library
         import numpy as np

         data = [32, 36, 46, 47, 56, 69, 75, 79, 79, 88, 89, 91, 92, 93, 96, 97,
                 101, 105, 112, 116]

         # First quartile (Q1)
         Q1 = np.percentile(data, 25, interpolation = 'midpoint')

         # Third quartile (Q3)
         Q3 = np.percentile(data, 75, interpolation = 'midpoint')

         # Interquaritle range (IQR)
         IQR = Q3 - Q1

         print(IQR)

         34.0
```

After knowing the various statistical functions in NumPy which are used in descriptive analytics, let us see some other interesting functionalities of NumPy.

© Edunet Foundation. All rights reserved.

1.15 Array Broadcasting in Numpy

The term broadcasting describes how numpy treats arrays with different shapes during arithmetic operations. Subject to certain constraints, the smaller array is “broadcast” across the larger array so that they have compatible shapes.

```
In [27]: a = np.array([1.0, 2.0, 3.0])
         b = np.array([2.0, 2.0, 2.0])
         a * b

Out[27]: array([2., 4., 6.])
```

```
In [28]: from numpy import array
         a = array([1.0, 2.0, 3.0])
         b = array([2.0, 2.0, 2.0])
         a * b

Out[28]: array([2., 4., 6.])
```

```
In [29]: from numpy import array
         a = array([1.0,2.0,3.0])
         b = 2.0
         a * b

Out[29]: array([2., 4., 6.])
```

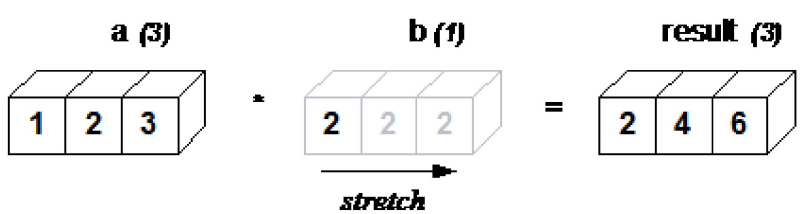


Figure 1: Broadcast Array
Ref: https://numpy.org/devdocs/user/theory.broadcasting.html#figure-1

In the simplest example of broadcasting, the scalar ``b`` is stretched to become an array of same shape as ``a`` so the shapes are compatible for element-by-element multiplication.

The result is equivalent to the previous example where b was an array. We can think of the scalar b being stretched during the arithmetic operation into an array with the

© Edunet Foundation. All rights reserved.

same shape as a. The new elements in b, as shown in Figure 1, are simply copies of the original scalar.

Sorting Arrays

numpy.sort() : This function returns a sorted copy of an array.

Parameters;

arr : Array to be sorted.

axis : Axis along which we need array to be started.

order : This argument specifies which fields to compare first.

kind : [‘quicksort’{default}, ‘mergesort’, ‘heapsort’]Sorting algorithm.

```
In [30]: import numpy as np

         # sort along the first axis
         a = np.array([[12, 15], [10, 1]])
         arr1 = np.sort(a, axis = 0)
         print ("Along first axis : \n", arr1)


         # sort along the last axis
         a = np.array([[10, 15], [12, 1]])
         arr2 = np.sort(a, axis = -1)
         print ("\nAlong first axis : \n", arr2)


         a = np.array([[12, 15], [10, 1]])
         arr1 = np.sort(a, axis = None)
         print ("\nAlong none axis : \n", arr1)
```

```
Along first axis :
 [[10  1]
 [12 15]]

Along first axis :
 [[10 15]
 [ 1 12]]

Along none axis :
 [ 1 10 12 15]
```

1.16 What is Data Visualization?

Data visualization is the graphical representation of information and data. By using visual elements like charts, graphs, and maps, data visualization tools provide an accessible way to see and understand trends, outliers, and patterns in data.

© Edunet Foundation. All rights reserved.

In the world of Big Data, data visualization tools and technologies are essential to analyze massive amounts of information and make data-driven decisions.

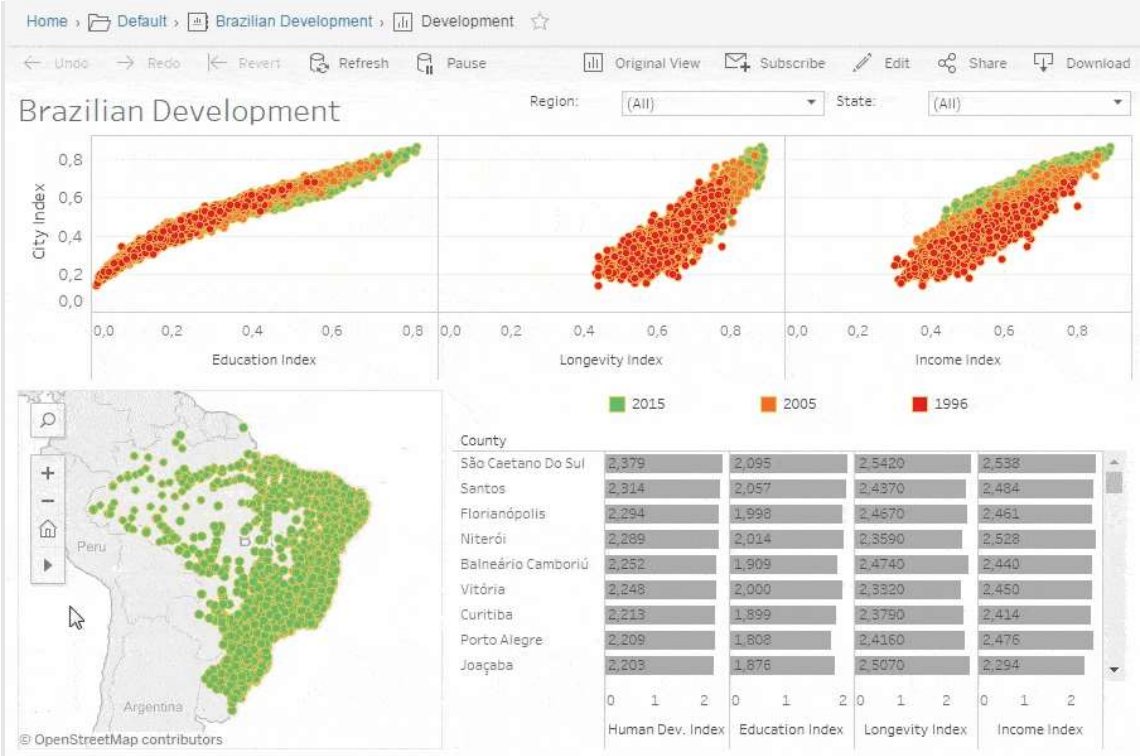


Image: sample Data visualization
Reference: https://cdnl.tblsft.com/sites/default/files/pages/_data_visualization_definition.gif

Our eyes are drawn to colors and patterns. We can quickly identify red from blue, square from circle. Our culture is visual, including everything from art and advertisements to TV and movies.

Data visualization is another form of visual art that grabs our interest and keeps our eyes on the message. When we see a chart, we quickly see trends and outliers. If we can see something, we internalize it quickly. It's storytelling with a purpose. If you've ever stared at a massive spreadsheet of data and couldn't see a trend, you know how much more effective a visualization can be.

Data visualization helps transform your data into an engaging story with details and patterns.

- Better Analysis
- Speed up decision making process
- Quick action
- Identifying patterns
- Story telling is more engaging
- Grasping the latest trends
- Finding errors

© Edunet Foundation. All rights reserved.

Families of Visualizations

Data Visualization's Family speaks to its nature. This allows for hierarchical classification between major groups: Charts, Geospatial Visualizations, and Tables.

Charts

A chart is a representation of data in the form of a graph, diagram, map, or tabular format. This could make the other two families, Geospatial and Tables, subfamilies of it.

Later, we will discuss about different types of charts in detail.

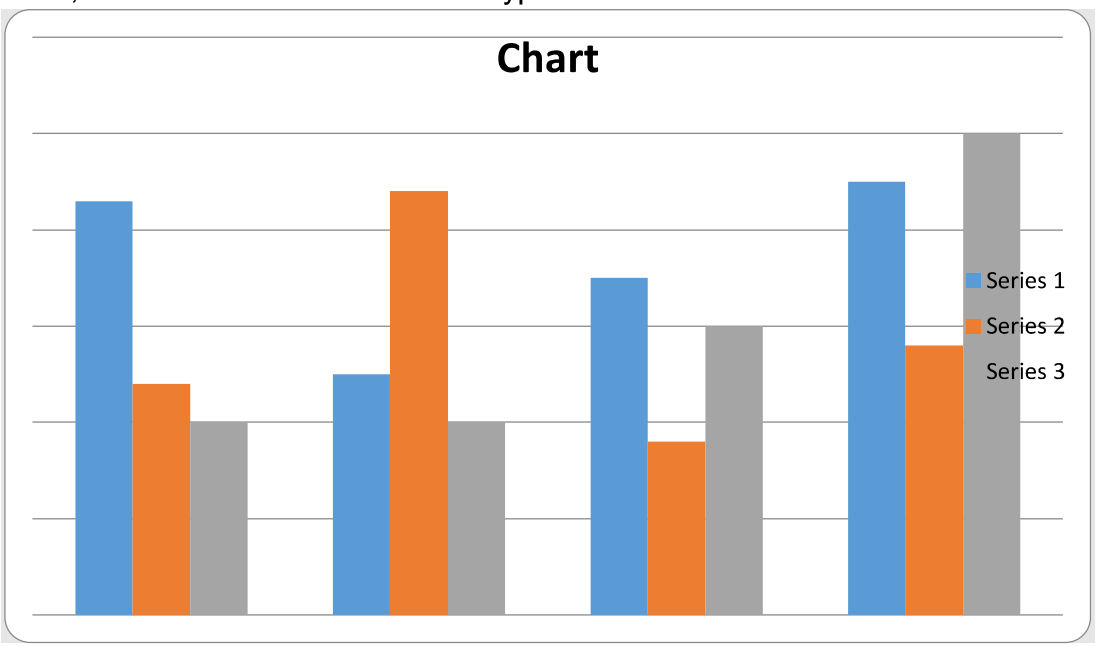


Image: Sample Chart

Tables

Tables, also known as "crosstabs" or "spreadsheets", are one of the most common ways to present data. They may not be visualizations in themselves, but they can be a powerful tool for visual analysis

	1234	678
	368	8034
	2620	2559
	971	322

Image: sample tables visualization

Reference: https://www.tableau.com/learn/articles/data-visualization/glossary

© Edunet Foundation. All rights reserved.

Plotting and Visualization

Plotting is a chart or map showing the movements or progress of an object.
A plot is a graphical technique for representing a data set, usually as a graph showing the relationship between two or more variables.
Python offers multiple great graphing libraries that come packed with lots of different features.

To get a little overview here are a few popular plotting libraries:

- Matplotlib: low level, provides lots of freedom
- Pandas Visualization: easy to use interface, built on Matplotlib
- Seaborn: high-level interface, great default styles
- ggplot: based on R's ggplot2, uses Grammar of Graphics
- Plotly: can create interactive plots

Let us explore the Matplotlib library.

Matplotlib

Here we will learn how to create basic plots using Matplotlib.

Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python.

How to install matplotlib in python?

```
pip install matplotlib
or
conda install matplotlib
```

```
import matplotlib.pyplot as plt
```

matplotlib.pyplot is a collection of command style functions that make matplotlib work like MATLAB.

© Edunet Foundation. All rights reserved.

1.17 Graph Plotting in Python using Matplotlib

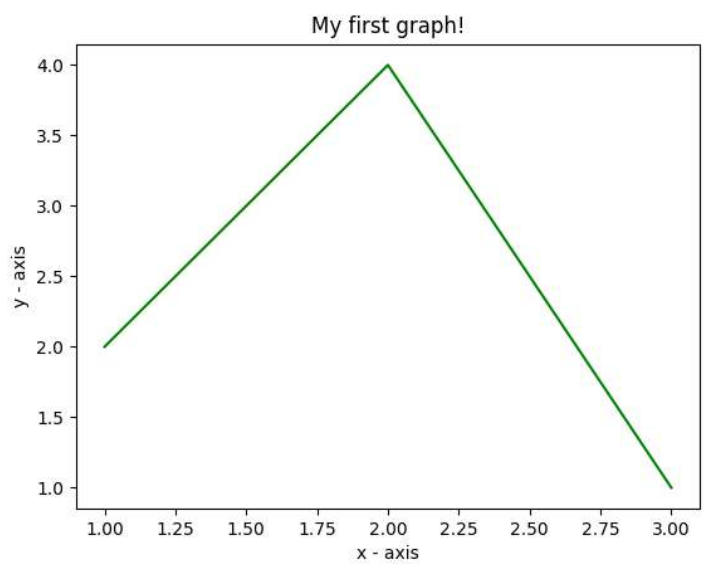
File Edit View Insert Cell Kernel Widgets Help

Run Code

In [1]:
```
# importing the required module
import matplotlib.pyplot as plt
# x axis values
x = [1,2,3]
# corresponding y axis values
y = [2,4,1]
# plotting the points
plt.plot(x, y)
# naming the x axis
plt.xlabel('x - axis')
# naming the y axis
plt.ylabel('y - axis')
# giving a title to my graph
plt.title('My first graph!')
# function to show the plot
plt.show()
```

Code: Graph Plotting in Python using matplotlib.ipynb

Output:



The code seems self explanatory. Following steps were followed:

- Define the x-axis and corresponding y-axis values as lists.

© Edunet Foundation. All rights reserved.

- Plot them on canvas using .plot() function.
- Give a name to x-axis and y-axis using .xlabel() and .ylabel() functions.
- Give a title to your plot using .title() function.
- Finally, to view your plot, we use .show() function.

1.18 Line Plot, Bar Plot, Scatter Plot

Line Plot

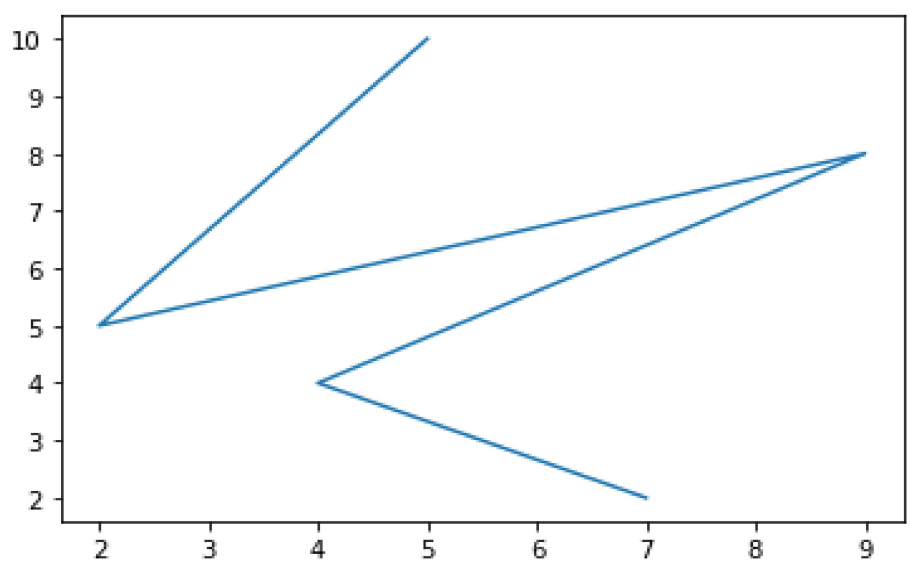
Write below code in **jupyter notebook** for **Line chart** and click on **Run**

```
#Line plot using matplotlib librabry, here we use pyplot function of matplotlib
# importing matplotlib module
from matplotlib import pyplot as plt

# x-axis values
x = [5, 2, 9, 4, 7]

# Y-axis values
y = [10, 5, 8, 4, 2]

# Function to plot
plt.plot(x, y)

# function to show the plot
plt.show()
```

Code: **Line Plot.ipynb**

Result:

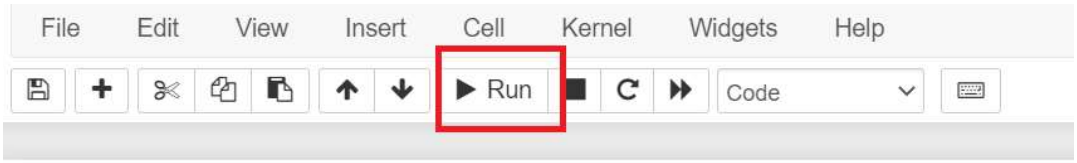
When you run the above code in jupyter notebook you will get the output shown below.



Bar Plot

Write the code given below in **jupyter notebook** for **Bar chart** and click on **Run**

© Edunet Foundation. All rights reserved.

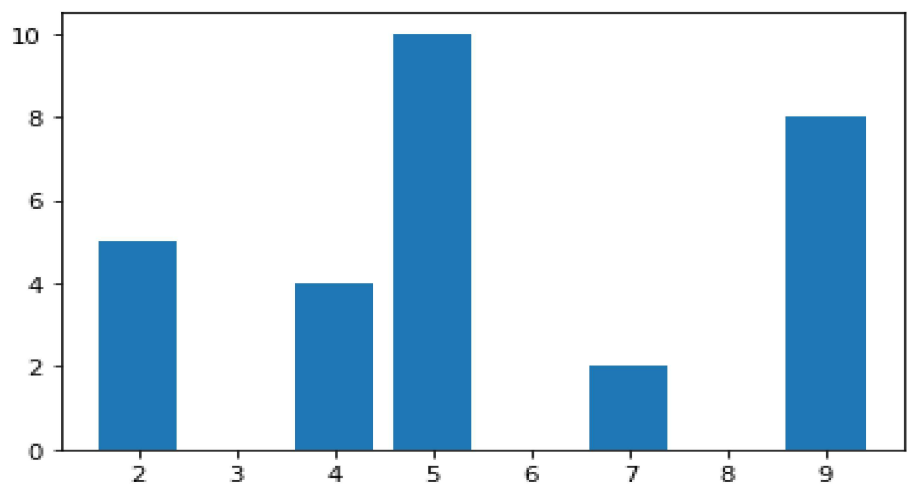



```
In [ ]: # Bar Plot

        # importing matplotlib module
        from matplotlib import pyplot as plt

        # x-axis values
        x = [5, 2, 9, 4, 7]

        # Y-axis values
        y = [10, 5, 8, 4, 2]

        # Function to plot
        plt.bar(x, y)

        # function to show the plot
        plt.show()
```

Code: Bar Plot.ipynb

Output:

When you run the above code in jupyter notebook you will get the output given below.



Scatter Plot

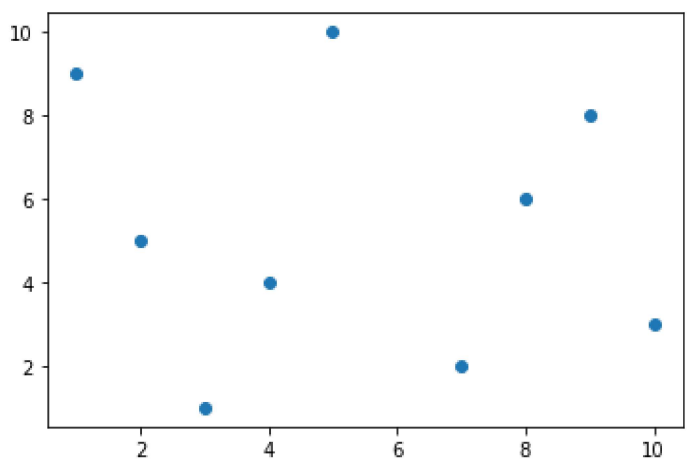
Write the code given below in **jupyter notebook** for **Scatter plot** and click on **Run**

© Edunet Foundation. All rights reserved.

```
In [2]: #Scatter Plot
        # importing matplotlib module
        from matplotlib import pyplot as plt

        # x-axis values
        x = [5, 2, 9, 4, 7, 3, 8, 10, 1]

        # Y-axis values
        y = [10, 5, 8, 4, 2, 1, 6, 3, 9]

        # Function to plot scatter
        plt.scatter(x, y)

        # function to show the plot
        plt.show()
```

Code: **Scatter Plot.ipynb**

Result:
When you run the above code in jupyter notebook you will get the following output.



1.19 Three-Dimensional Plotting in Matplotlib

Three-dimensional plots are enabled by importing the mplot3d toolkit, included with the main Matplotlib installation:

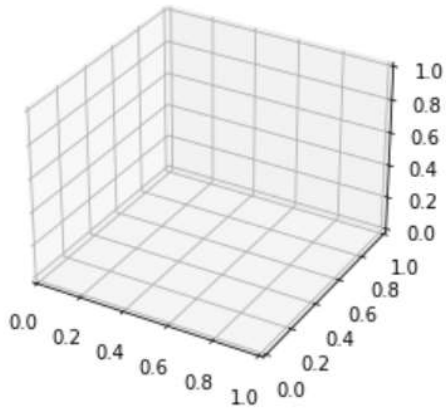
Step 1:

```
from mpl_toolkits import mplot3d
```

© Edunet Foundation. All rights reserved.

Once this submodule is imported, three-dimensional axes can be created by passing the keyword projection='3d' to any of the normal axes creation routines:
Step 2: create simple 3D graph

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


        fig = plt.figure()
        ax = plt.axes(projection ='3d')
```

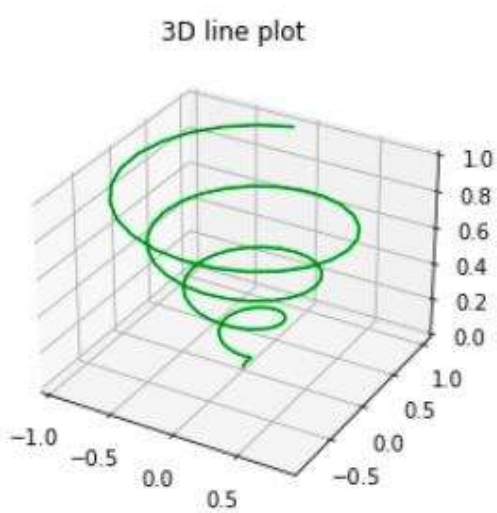


Plotting 3-D Lines and Points

Graph with lines and point are the simplest 3 dimensional graph. **ax.plot3d and ax.scatter** are the function to plot line and point graph respectively.
Write down the code given below in jupyter notebook and click on Run

```
In [1]: # importing mplot3d toolkits, numpy and matplotlib
        from mpl_toolkits import mplot3d
        import numpy as np
        import matplotlib.pyplot as plt

        fig = plt.figure()

        # syntax for 3-D projection
        ax = plt.axes(projection ='3d')

        # defining all 3 axes
        z = np.linspace(0, 1, 100)
        x = z * np.sin(25 * z)
        y = z * np.cos(25 * z)

        # plotting
        ax.plot3D(x, y, z, 'green')
        ax.set_title('3D line plot')
        plt.show()
```

Output:



Plot a pie chart in Python using Matplotlib

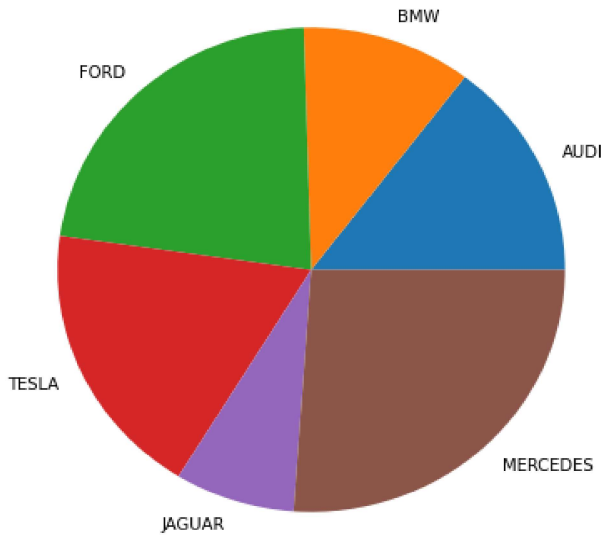
Matplotlib API has `pie()` function in its `pyplot` module which creates a pie chart representing the data in an array.

Let's create a simple pie chart using the `pie()` function:

© Edunet Foundation. All rights reserved.

```
In [1]: # Import libraries
        from matplotlib import pyplot as plt
        import numpy as np


        # Creating dataset
        cars = ['AUDI', 'BMW', 'FORD',
                'TESLA', 'JAGUAR', 'MERCEDES']

        data = [23, 17, 35, 29, 12, 41]

        # Creating plot
        fig = plt.figure(figsize =(10, 7))
        plt.pie(data, labels = cars)

        # show plot
        plt.show()
```

Output:



1.20 Visualizing with NumPy

Create plots using NumPy Array

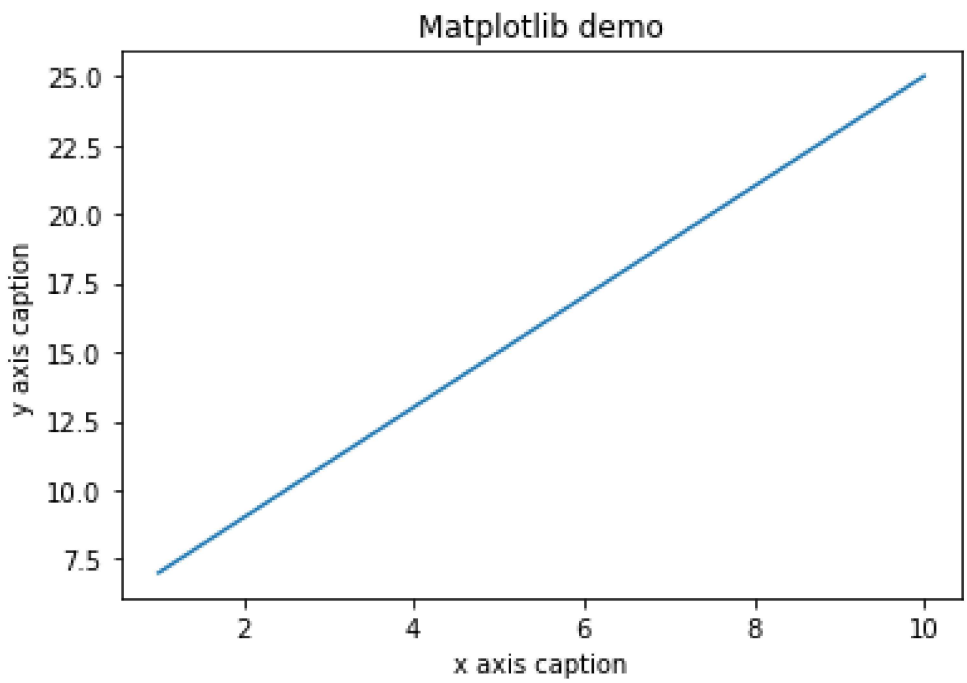
NumPy is a Python library used for working with arrays.
Step 1: write the program given below in jupyter notebook and click on Run.

© Edunet Foundation. All rights reserved.

```
In [1]: #import numpy
        import numpy as np
        from matplotlib import pyplot as plt

        x = np.arange(1,11)
        y = 2 * x + 5
        plt.title("Matplotlib demo")
        plt.xlabel("x axis caption")
        plt.ylabel("y axis caption")
        plt.plot(x,y)
        plt.show()
```

Code: Numpy excercise.ipynb

Output:



Draw Sine Wave Plot using Numpy

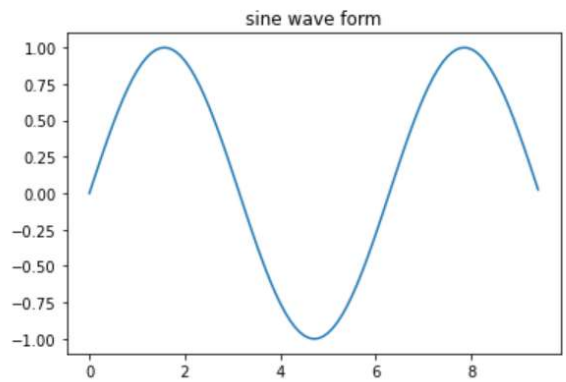
Write down the code given below in Jupyter notebook and click on Run

© Edunet Foundation. All rights reserved.

File Edit View Insert Cell Kernel Widgets Help

Run Code

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

# Compute the x and y coordinates for points on a sine curve
x = np.arange(0, 3 * np.pi, 0.1)
y = np.sin(x)
plt.title("sine wave form")

# Plot the points using matplotlib
plt.plot(x, y)
plt.show()
```



Code: sine_wave_numpy.ipynb

We have learned and explored the various functions in Matplotlib library which are used to visualize the data in various format and make it available to help in decision making process. Now, let us see another powerful library called Pandas which is very popular in data analytics.

DIY Activity

Activity: Data Visualization using Matplotlib

Matplotlib is a Python plotting library that produces high-quality charts and figures, which helps us visualize extensive data to understand better. The data set named 'company_sales_data' is given to you. Analyze the data and plot the graphs for the following questions.

1. Read Total profit of all months and show it using a line plot
2. Read all product sales data and show it using a multiline plot
3. Read toothpaste sales data of each month and show it using a scatter plot
4. Read face cream and face-wash product sales data and show it using the bar chart. Save this plot to your hard disk
5. Calculate total sale data for last year for each product and show it using a Pie chart

Self-paced Courses –
https://docs.microsoft.com/en-us/learn/modules/explore-analyze-data-with-python/

© Edunet Foundation. All rights reserved.

Chapter 2: Exploratory Data Analysis and Supervised Learning

Learning Outcomes:

- Analyse different types of data using Pandas
- Understand the use of data analytics in improved decision making
- Create Machine Learning model using various Algorithms
- Demonstrate working of ML model without coding using ML Studio

2.1 Data Manipulation with Pandas

Pandas is a fast, powerful, flexible and easy to use open source data analysis and manipulation tool. Pandas is a newer package built on top of NumPy, and provides an efficient implementation of **a DataFrame**.

DataFrames are essentially multidimensional arrays with attached row and column labels, and often with heterogeneous types and/or missing data. Apart from offering a convenient storage interface for labelled data, Pandas also implements a number of powerful data operations familiar to users of both database frameworks and spreadsheet programs.

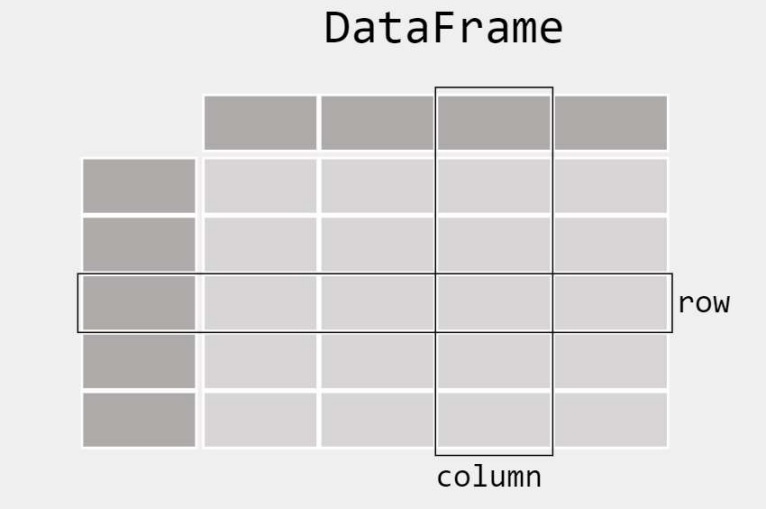


Image: dataframe

Reference: https://pandas.pydata.org/docs/getting_started/index.html

© Edunet Foundation. All rights reserved.

How to install Pandas?

pip install pandas

How to load Pandas?

import pandas as pd

To load the pandas package and start working with it, import the package

For e.g store passengers data of the Titanic. For a number of passengers, I know the name (characters), age (integers) and sex (male/female) data.

```
In [7]: import pandas as pd
        df = pd.DataFrame(
                {
                     "Name": [
                         "Mr. Pravinkumar Prajapati",
                         "Miss. Priyanka Paul",
                        "Miss. Elizabeth",
                    ],
                     "Age": [22, 30, 58],
                     "Sex": ["male", "female", "female"],
                 }
            )
```

Code: panda data manipulation.ipynb

Output:

```
In [8]: df
```

Out[8]:

	Name	Age	Sex
0	Mr. Pravinkumar Prajapati	22	male
1	Miss. Priyanka Paul	30	female
2	Miss. Elizabeth	58	female

To manually store data in a table, create a DataFrame. When using a Python dictionary of lists, the dictionary keys will be used as column headers and the values in each list as columns of the DataFrame.

Now we can perform more operation to get more insights into data. Let us explore more command

© Edunet Foundation. All rights reserved.

When selecting a single column of a pandas **DataFrame**, the result is a pandas **Series**. To select the column, use the column label in between square brackets **[]**.

```
In [6]: df["Age"]

Out[6]: 0    22
        1    35
        2    58
        Name: Age, dtype: int64
```

Find maximum Age of the passengers.

```
In [9]: df["Age"].max()

Out[9]: 58
```

The describe() method provides a quick overview of the numerical data in a DataFrame. As the Name and Sex columns are textual data, these are by default not taken into account by the describe() method

```
In [10]: df.describe()
Out[10]:
```

	Age
count	3.000000
mean	36.666667
std	18.903263
min	22.000000
25%	26.000000
50%	30.000000
75%	44.000000
max	58.000000

.

How to manipulate textual data?

Here we use the Titanic data set, stored as CSV. The data consists of the following data columns:

- PassengerId: Id of every passenger.
- Survived: These features have value 0 and 1. 0 for not survived and 1 for survived.
- Pclass: There are 3 classes: Class 1, Class 2 and Class 3.

© Edunet Foundation. All rights reserved.

- Name: Name of passenger.
- Sex: Gender of passenger.
- Age: Age of passenger.
- SibSp: Indication that passenger have siblings and spouse.
- Parch: Whether a passenger is alone or have family.
- Ticket: Ticket number of passenger.
- Fare: Indicating the fare.
- Cabin: The cabin of passenger.
- Embarked: The embarked category.

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

```
In [3]: titanic = pd.read_csv("titanic.csv")
```

```
In [4]: titanic.head()
```

Out[4]:

	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

Code: pandas data manipulation.ipynb

Task 1: Make all name characters lowercase.

```
In [5]: titanic["Name"].str.lower()
```

```
Out[5]: 0                                braund, mr. owen harris
        1      cumings, mrs. john bradley (florence briggs th...
        2                               heikkinen, miss. laina
        3           futrelle, mrs. jacques heath (lily may peel)
        4                             allen, mr. william henry
                                   ...
        886                             montvila, rev. juozas
        887                       graham, miss. margaret edith
        888            johnston, miss. catherine helen "carrie"
        889                             behr, mr. karl howell
        890                               dooley, mr. patrick
        Name: Name, Length: 891, dtype: object
```

Code: pandas data manipulation.ipynb

Task 2: Create a new column Surname that contains the surname of the passengers by extracting the part before the comma.

© Edunet Foundation. All rights reserved.

```
In [6]: titanic["Name"].str.split(",")
```

```
Out[6]: 0                              [Braund,  Mr. Owen Harris]
        1      [Cumings,  Mrs. John Bradley (Florence Briggs ...
        2                               [Heikkinen,  Miss. Laina]
        3        [Futrelle,  Mrs. Jacques Heath (Lily May Peel)]
        4                             [Allen,  Mr. William Henry]
                                      ...
        886                              [Montvila,  Rev. Juozas]
        887                       [Graham,  Miss. Margaret Edith]
        888          [Johnston,  Miss. Catherine Helen "Carrie"]
        889                              [Behr,  Mr. Karl Howell]
        890                                [Dooley,  Mr. Patrick]
        Name: Name, Length: 891, dtype: object
```

Code: pandas data manipulation.ipynb

Using the series.str.split() method, each of the values is returned as a list of 2 elements. The first element is the part before the comma and the second element is the part after the comma.

```
In [7]: titanic["Surname"] = titanic["Name"].str.split(",").str.get(0)
```

```
In [8]: titanic["Surname"]
```

```
Out[8]: 0         Braund
        1        Cumings
        2      Heikkinen
        3       Futrelle
        4          Allen
                 ...
        886     Montvila
        887       Graham
        888     Johnston
        889         Behr
        890       Dooley
        Name: Surname, Length: 891, dtype: object
```

Code: pandas data manipulation.ipynb

By this method and various functions you can perform data manipulation using pandas.

2.2 Introducing Pandas Objects

Pandas objects can be thought of as enhanced versions of NumPy structured arrays in which the rows and columns are identified with labels rather than simple integer indices

There are fundamental Pandas data structures:

1. Series,

© Edunet Foundation. All rights reserved.

2. DataFrame

The Pandas Series Object

A Pandas Series is a one-dimensional array of indexed data. It can be created from a list or array as follows:

```
In [1]: #Pandas Series Object
        import numpy as np
        import pandas as pd
        data = pd.Series([0.25, 0.5, 0.75, 1.0])
        data

Out[1]: 0    0.25
        1    0.50
        2    0.75
        3    1.00
        dtype: float64
```

```
In [ ]:
```

Code: **panda object.ipynb**

As we see in the output, the `Series` wraps both a sequence of values and a sequence of indices, which we can access with the `values` and `index` attributes. The `values` are simply a familiar NumPy array:

```
In [2]: data.values

Out[2]: array([0.25, 0.5 , 0.75, 1.  ])
```

Like with a NumPy array, data can be accessed by the associated index via the familiar Python square-bracket notation:

```
In [3]: data[1]

Out[3]: 0.5

In [4]: data[1:3]

Out[4]: 1    0.50
        2    0.75
        dtype: float64
```

The Pandas DataFrame Object

The next fundamental structure in Pandas is the `DataFrame`. Like the `Series` object discussed in the previous section, the `DataFrame` can be thought of either as a generalization of a NumPy array, or as a specialization of a Python dictionary.

© Edunet Foundation. All rights reserved.

DataFrame as a generalized NumPy array

If a `Series` is an analog of a one-dimensional array with flexible indices, a `DataFrame` is an analog of a two-dimensional array with both flexible row indices and flexible column names.

Just as you might think of a two-dimensional array as an ordered sequence of aligned one-dimensional columns, you can think of a `DataFrame` as a sequence of aligned `Series` objects. Here, by "aligned" we mean that they share the same index.

To demonstrate this, let's first construct two new Series listing of the area and the population of say five states of USA.:

```
In [5]: area_dict = {'California': 423967, 'Texas': 695662, 'New York': 141297,
                     'Florida': 170312, 'Illinois': 149995}
        area = pd.Series(area_dict)
        area

Out[5]: California    423967
        Texas         695662
        New York      141297
        Florida       170312
        Illinois      149995
        dtype: int64
```

```
In [7]: population_dict = {'California': 38332521,'Texas': 26448193,'New York': 19651127,
                           'Florida': 19552860,'Illinois': 12882135}
        population = pd.Series(population_dict)
        population

Out[7]: California    38332521
        Texas         26448193
        New York      19651127
        Florida       19552860
        Illinois      12882135
        dtype: int64
```

Code: **panda object.ipynb**

Now that we have the `area` and the `population` Series from before, we can use a dictionary to construct a single two-dimensional object containing this information:

```
In [8]: states = pd.DataFrame({'population': population,
                               'area': area})
        states
```

Out[8]:

	population	area
California	38332521	423967
Texas	26448193	695662
New York	19651127	141297
Florida	19552860	170312
Illinois	12882135	149995

© Edunet Foundation. All rights reserved.

2.3 Creating Series from simple datatypes

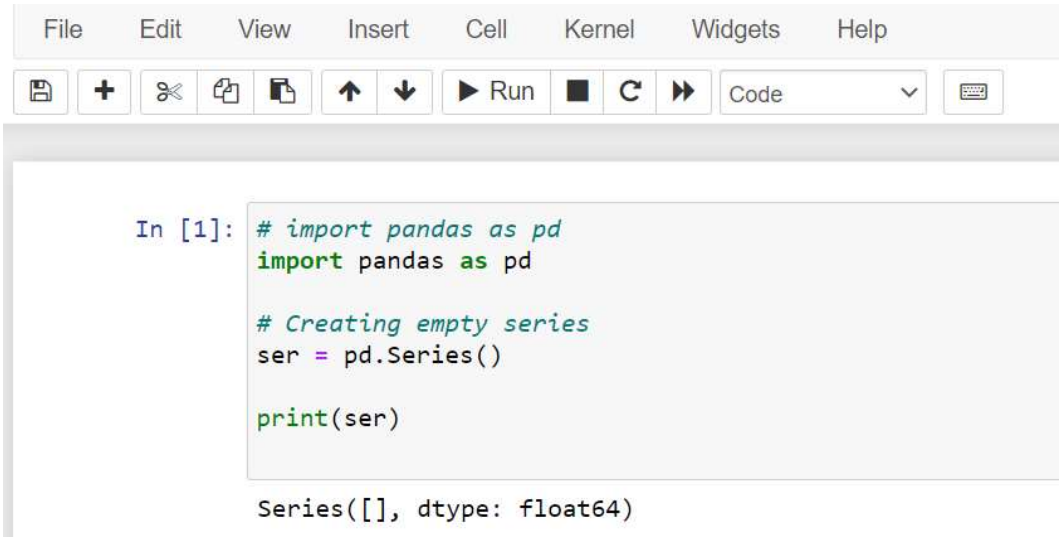
Creating a Pandas Series

The Pandas Series can be defined as a one-dimensional array that is capable of storing various data types. We can easily convert the list, tuple, and dictionary into series using "**series**' method. The row labels of series are called the index. A Series cannot contain multiple columns. It has the following parameter:

- **data:** It can be any list, dictionary, or scalar value.
- **index:** The value of the index should be unique and hashable. It must be of the same length as data. If we do not pass any index, default **np.arrange(n)** will be used.
- **dtype:** It refers to the data type of series.
- **copy:** It is used for copying the data.

1. ***Create an Empty Series***:

We can easily create an empty series in Pandas which means it will not have any value.



Code: **Creating Series from simple datatypes.ipynb**

2. *Creating a series from array:*

In order to create a series from array, we have to import a numpy module and have to use array() function.

```
In [2]: # import pandas as pd
        import pandas as pd

        # import numpy as np
        import numpy as np

        # simple array
        data = np.array(['m', 'i', 'c', 'r', 'o', 's', 'o', 'f', 't'])

        ser = pd.Series(data)
        print(ser)
```

OUTPUT----->

```
0    m
1    i
2    c
3    r
4    o
5    s
6    o
7    f
8    t
dtype: object
```

Code: Creating Series from simple datatypes.ipynb

3. Creating a series from Lists:

In order to create a series from list, we have to first create a list after that we can create a series from list.

© Edunet Foundation. All rights reserved.

```
In [4]: import pandas as pd

        # a simple list
        list = ['E', 'd', 't', 'e', 'c', 'H']

        # create series form a list
        ser = pd.Series(list)
        print(ser)

0    E
1    d
2    t
3    e
4    c
5    H
dtype: object
```

```
In [ ]:
```

Code: Creating Series from simple datatypes.ipynb

4. Creating a series from Dictionary:

In order to create a series from dictionary, we have to first create a dictionary after that we can make a series using dictionary. Dictionary key are used to construct a index.

```
In [5]: import pandas as pd

        # a simple dictionary
        dict = {'Python' : 10,
                'for' : 20,
                'Artificial' : 30,

                'Intelligence' : 50}

        # create series from dictionary
        ser = pd.Series(dict)

        print(ser)

Python          10
for             20
Artificial      30
Intelligence    50
dtype: int64
```

Code: Creating Series from simple datatypes.ipynb

© Edunet Foundation. All rights reserved.

2.4 Data Storage Formats in Pandas

The different data storage formats available to be manipulated by Pandas library are text, binary and SQL. Below is a table containing available 'readers' and 'writers' functions of the pandas I/O API set with data format and description. ***[6.5]***

Format Type	Data Description	Reader	Writer
text	CSV	read_csv	to_csv
text	Fixed-Width Text File	read_fwf	
text	JSON	read_json	to_json
text	HTML	read_html	to_html
text	Local clipboard	read_clipboard	to_clipboard
binary	MS Excel	read_excel	to_excel
binary	OpenDocument	read_excel	
binary	HDF5 Format	read_hdf	to_hdf
binary	Feather Format	read_feather	to_feather
binary	Parquet Format	read_parquet	to_parquet
binary	ORC Format	read_orc	
binary	Msgpack	read_msgpack	to_msgpack
binary	Stata	read_stata	to_stata
binary	SAS	read_sas	
binary	SPSS	read_spss	
binary	Python Pickle Format	read_pickle	to_pickle
SQL	SQL	read_sql	to_sql
SQL	Google BigQuery	read_gbq	to_gbq

Figure: "readers" and "writers" functions in pandas
Reference: https://pandas.pydata.org/pandas-docs/stable/user_guide/io.html

2.5 CSV file and JSON file

What is CSV file?

A CSV is a comma-separated values file, which allows data to be saved in a tabular format. CSV files can be used with many spreadsheet program, such as Microsoft Excel or Google Spreadsheets. They differ from other spreadsheet file types because you can only have a single sheet in a file, they cannot save cell, column, or row. Also, you cannot save formulas in this format.

© Edunet Foundation. All rights reserved.

Why are .CSV files used?

These files serve a number of different business purposes. Take, for instance, they help companies export a high volume of data to a more concentrated database. They also serve two other primary business functions:

- CSV files are plain-text files, making them easier for the website developer to create
- Since they're plain text, they're easier to import into a spreadsheet or another storage database, regardless of the specific software you're using.
- To better organize large amounts of data.

How do I save CSV files?

Saving CSV files is relatively easy, you just need to know where to change the file type. Under the "File name" section in the "Save As" tab, you can select "Save as type" and change it to "CSV (Comma delimited) (*.csv). Once that option is selected, you are on your way to quicker and easier data organization. This should be the same for both Apple and Microsoft operating systems.

What is a JSON file?

A JSON file is a file that stores simple data structures and objects in JavaScript Object Notation (JSON) format, which is a standard data interchange format. It is primarily used for transmitting data between a web application and a server. JSON files are lightweight, text-based, human-readable, and can be edited using a text editor.

How do I open a JSON file?

Because JSON files are plain text files, you can open them in any text editor, including:

- Microsoft Notepad (Windows)
- Apple TextEdit (Mac)
- Vim (Linux)
- GitHub Atom (cross-platform)

You can also open a JSON file in the Google Chrome and Mozilla Firefox web browsers by dragging and dropping the file into your browser window.

2.6 Reading data from files

Load CSV files to Python Pandas

© Edunet Foundation. All rights reserved.

The basic process of loading data from a CSV file into a Pandas DataFrame is achieved using the “read_csv” function in Pandas:

```
# Load the Pandas libraries with alias 'pd'
import pandas as pd

# Read data from file 'filename.csv'
# (in the same directory that your python process is based)
# Control delimiters, rows, column names with read_csv (see later)
data = pd.read_csv("filename.csv")

# Preview the first 5 lines of the loaded data
data.head();
```

While this code seems simple, an understanding of three fundamental concepts is required to fully grasp and debug the operation of the data loading procedure if you run into issues:

Load JSON files to Python Pandas

```
import pandas as pd
df = pd.read_json (r'C:\Users\Ron\Desktop\data.json')
print (df)
```

Run the code in Python (adjusted to your path), and you’ll get the following DataFrame:

```
             Product  Price
0  Desktop Computer    700
1            Tablet    250
2            iPhone    800
3            Laptop   1200
```

Figure: Output

2.7 Groupby Methods

Before applying groupby function to the dataset, let’s go over a visual example. Assume we have two features. One is color which is a categorical feature and the other one is a numerical feature, values. We want to group values by color and calculate the mean (or any other aggregation) of values for different colors. Then finally sort the colors based on average values. The following figure shows the steps of this process.

© Edunet Foundation. All rights reserved.

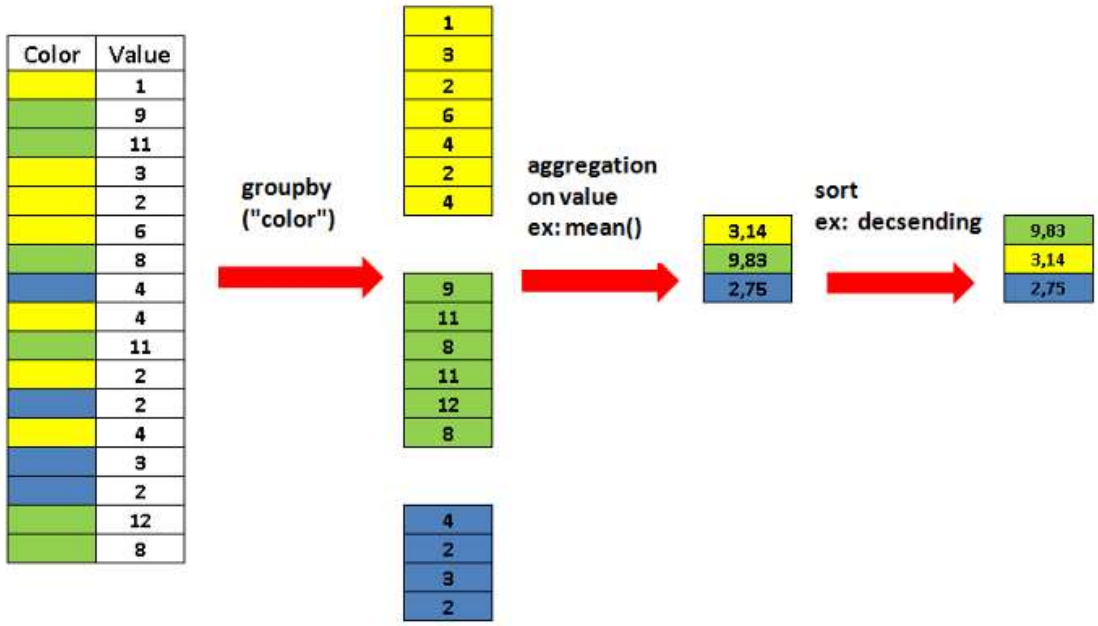


Figure: groupby in Pandas
Reference:https://towardsdatascience.com/pandas-groupby-explained-453692519d0

A Sample DataFrame

In order to demonstrate the effectiveness and simplicity of the grouping commands, we will need some data. The dataset contains 830 entries from my mobile phone log spanning a total time of 5 months. The CSV file can be loaded into a pandas DataFrame using the pandas.DataFrame.from_csv() function, and looks like this:

	date	duration	item	month	network	network_type
0	15/10/14 06:58	34.429	data	2014-11	data	data
1	15/10/14 06:58	13.000	call	2014-11	Vodafone	mobile
2	15/10/14 14:46	23.000	call	2014-11	Meteor	mobile
3	15/10/14 14:48	4.000	call	2014-11	Tesco	mobile
4	15/10/14 17:27	4.000	call	2014-11	Tesco	mobile
5	15/10/14 18:55	4.000	call	2014-11	Tesco	mobile
6	16/10/14 06:58	34.429	data	2014-11	data	data
7	16/10/14 15:01	602.000	call	2014-11	Three	mobile
8	16/10/14 15:12	1050.000	call	2014-11	Three	mobile

© Edunet Foundation. All rights reserved.

9	16/10/14 15:30	19.000	call	2014-11	voicemail	voicemail
10	16/10/14 16:21	1183.000	call	2014-11	Three	mobile
11	16/10/14 22:18	1.000	sms	2014-11	Meteor	mobile
…	…	…	…	…	…	…

The main columns in the file are:

1. date: The date and time of the entry
2. duration: The duration (in seconds) for each call, the amount of data (in MB) for each data entry, and the number of texts sent (usually 1) for each sms entry.
3. item: A description of the event occurring – can be one of call, sms, or data.
4. month: The billing month that each entry belongs to – of form 'YYYY-MM'.
5. network: The mobile network that was called/texted for each entry.
6. network_type: Whether the number being called was a mobile, international ('world'), voicemail, landline, or other ('special') number.

The date column can be parsed using the **dateutil** library.

```
import pandas as pd
import dateutil

# Load data from csv file
data = pd.DataFrame.from_csv('phone_data.csv')
# Convert date from string to date times
data['date'] = data['date'].apply(dateutil.parser.parse, dayfirst=True)
```

Summarizing the DataFrame

Once the data has been loaded into Python, Pandas makes the calculation of different statistics very simple. For example, mean, max, min, standard deviations and more for columns are easily calculable:

```
# How many rows the dataset
data['item'].count()
```

Out[38]: 830

```
# What was the longest phone call / data entry?
data['duration'].max()
```

Out[39]: 10528.0

```
# How many seconds of phone calls are recorded in total?
data['duration'][data['item'] == 'call'].sum()
```

Out[40]: 92321.0

```
# How many entries are there for each month?
data['month'].value_counts()
```

© Edunet Foundation. All rights reserved.

Out[41]:
2014-11 230
2015-01 205
2014-12 157
2015-02 137
2015-03 101
dtype: int64

```
# Number of non-null unique network entries
data['network'].nunique()
```

Out[42]: 9

The .describe() function is a useful summarisation tool that will quickly display statistics for any variable or group it is applied to. The describe() output varies depending on whether you apply it to a numeric or character column.

2.8 Pivot Tables

You may be familiar with pivot tables in Excel to generate easy insights into your data. The function is quite similar to the group by function available in Pandas.

How to Build a Pivot Table in Python

It's a table of statistics that helps summarize the data of a larger table by "pivoting" that data. In Pandas, we can construct a pivot table using the following syntax,

pandas.pivot_table(data)

We'll use Pandas to import the data into a dataframe called df. We'll also print out the first five rows using the .head() function:
import pandas as pd df = pd.read_excel('https://github.com/datagy/pivot_table_pandas/raw/master/sample_pivot.xlsx', parse_dates=['Date']) print(df.head())

```
Out:
  Date       Region Type                 Units   Sales
0 2020-07-11 East   Children's Clothing  18      306
1 2020-09-23 North  Children's Clothing  14      448
2 2020-04-02 South  Women's Clothing     17      425
3 2020-02-28 East   Children's Clothing  26      832
4 2020-03-19 West   Women's Clothing     3       33
```

© Edunet Foundation. All rights reserved.

Creating a Pivot Table in Pandas

We'll begin by aggregating the Sales values by the Region the sale took place in:
sales_by_region = pd.pivot_table(df, index = 'Region', values = 'Sales')
print(sales_by_region)
This returns the following output:

```
Sales   Region
East    408.182482
North   438.924051
South   432.956204
West    452.029412
```

This gave us a summary of the Sales field by Region. The default parameter for **aggfunc** is mean. Because of this, the Sales field in the resulting dataframe is the average of Sales per Region.
If we wanted to change the type of function used, we could use the **aggfunc** parameter. For example, if we wanted to return the sum of all Sales across a region, we could write:
total_by_region = pd.pivot_table(df, index = 'Region', values = 'Sales', aggfunc='sum') print(total_by_region)
This returns:

```
Sales    Region
East     167763
North    138700
South    59315
West     61476
```

Filtering Python Pivot Tables

Let's create a dataframe that generates the **mean Sale** price by **Region**:
avg_region_price = pd.pivot_table(df, index = 'Region', values = 'Sales')
The values in this dataframe are:

```
Sales    Region
East     408.182482
North    438.924051
South    432.956204
West     452.029412
```

Now, say we wanted to filter the dataframe to only include Regions where the average sale price was over 450, we could write:

© Edunet Foundation. All rights reserved.

avg_region_price[avg_region_price['Sales'] > 450]

```
Sales   Region
West    452.029412
```

We can also apply multiple conditions, such as filtering to show only sales greater than 450 or less than 430.
avg_region_price[(avg_region_price['Sales'] > 450) **|** *(avg_region_price['Sales'] < 430)]*
We have wrapped each condition in brackets and separated the conditions by a **pipe** (**|**) symbol. This returns the following:

```
Sales   Region
East    408.182482
West    452.029412
```

2.9 Pandas Plotting

Plot a Scatter Diagram using Pandas

Scatter plots are used to depict a relationship between two variables. Here are the steps to plot a scatter diagram using Pandas.

Step 1: Prepare the data

To start, prepare the data for your scatter diagram.
For example, the following data will be used to create the scatter diagram. This data captures the relationship between two variables related to an economy:

Unemployment_Rate	Stock_Index_Price
6.1	1500
5.8	1520
5.7	1525
5.7	1523
5.8	1515
5.6	1540
5.5	1545
5.3	1560
5.2	1555
5.2	1565

Step 2: Create the DataFrame

Once you have your data ready, you can proceed to create the DataFrame in Python. For our example, the DataFrame would look like this:
import pandas as pd

© Edunet Foundation. All rights reserved.

```
data = {'Unemployment_Rate': [6.1,5.8,5.7,5.7,5.8,5.6,5.5,5.3,5.2,5.2],
        'Stock_Index_Price': [1500,1520,1525,1523,1515,1540,1545,1560,1555,1565]
      }

df = pd.DataFrame(data,columns=['Unemployment_Rate','Stock_Index_Price'])
print (df)
```

Run the code in Python, and you'll get the following DataFrame:

	Unemployment_Rate	Stock_Index_Price
0	6.1	1500
1	5.8	1520
2	5.7	1525
3	5.7	1523
4	5.8	1515
5	5.6	1540
6	5.5	1545
7	5.3	1560
8	5.2	1555
9	5.2	1565

Step 3: Plot the DataFrame using Pandas

Finally, you can plot the DataFrame by adding the following syntax:
df.plot(x ='Unemployment_Rate', y='Stock_Index_Price', kind = 'scatter')

Notice that you can specify the type of chart by setting **kind = 'scatter'**
You'll also need to add the Matplotlib syntax to show the plot (ensure that the Matplotlib package is install in Python):

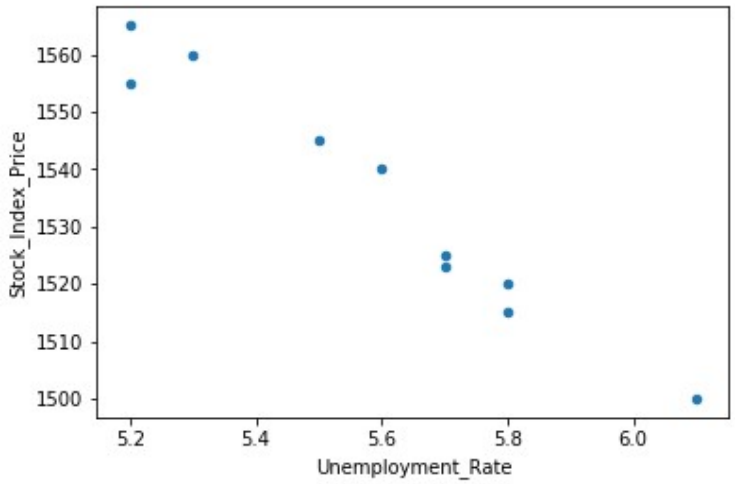
- *import matplotlib.pyplot as plt*
- *plt.show()*

Putting everything together:

```
import pandas as pd
import matplotlib.pyplot as plt

data = {'Unemployment_Rate': [6.1,5.8,5.7,5.7,5.8,5.6,5.5,5.3,5.2,5.2],
        'Stock_Index_Price': [1500,1520,1525,1523,1515,1540,1545,1560,1555,1565]
      }

df = pd.DataFrame(data,columns=['Unemployment_Rate','Stock_Index_Price'])
df.plot(x ='Unemployment_Rate', y='Stock_Index_Price', kind = 'scatter')
plt.show();
```

Once you run the above code, you'll get the following scatter diagram:

© Edunet Foundation. All rights reserved.



Plot a Line Chart using Pandas

Line charts are often used to display trends overtime. Let's now see the steps to plot a line chart using Pandas.

Step 1: Prepare the data

To start, prepare your data for the line chart. Here is an example of a dataset that captures the unemployment rate over time:

Year	Unemployment_Rate
1920	9.8
1930	12
1940	8
1950	7.2
1960	6.9
1970	7
1980	6.5
1990	6.2
2000	5.5
2010	6.3

Step 2: Create the DataFrame

Now create the DataFrame based on the above data:

```
import pandas as pd
data = {'Year': [1920,1930,1940,1950,1960,1970,1980,1990,2000,2010],
        'Unemployment_Rate': [9.8,12,8,7.2,6.9,7,6.5,6.2,5.5,6.3] }
df = pd.DataFrame(data,columns=['Year','Unemployment_Rate'])
print (df);
```

This is how the DataFrame would look like:

© Edunet Foundation. All rights reserved.

	Year	Unemployment_Rate
0	1920	9.8
1	1930	12.0
2	1940	8.0
3	1950	7.2
4	1960	6.9
5	1970	7.0
6	1980	6.5
7	1990	6.2
8	2000	5.5
9	2010	6.3

Step 3: Plot the DataFrame using Pandas

Finally, plot the DataFrame by adding the following syntax:

df.plot(x ='Year', y='Unemployment_Rate', kind = 'line')

You'll notice that the *kind* is now set to **'line'** in order to plot the line chart.

```
import pandas as pd
import matplotlib.pyplot as plt
 data = {'Year': [1920,1930,1940,1950,1960,1970,1980,1990,2000,2010],
       'Unemployment_Rate': [9.8,12,8,7.2,6.9,7,6.5,6.2,5.5,6.3]}
df = pd.DataFrame(data,columns=['Year','Unemployment_Rate'])
df.plot(x ='Year', y='Unemployment_Rate', kind = 'line')
plt.show();
```

Plot a Bar Chart using Pandas

Bar charts are used to display categorical data. Let's now see how to plot a bar chart using Pandas.

© Edunet Foundation. All rights reserved.

Step 1: Prepare your data

As before, you'll need to prepare your data. Here, the following dataset will be used to create the bar chart:

Country	GDP_Per_Capita
USA	45000
Canada	42000
Germany	52000
UK	49000
France	47000

Step 2: Create the DataFrame

Create the DataFrame as follows:

```
import pandas as pd
 data = {'Country': ['USA','Canada','Germany','UK','France'],
       'GDP_Per_Capita': [45000,42000,52000,49000,47000] }
df = pd.DataFrame(data,columns=['Country','GDP_Per_Capita'])
print (df);
```

You'll then get this DataFrame:

```
   Country  GDP_Per_Capita
0      USA           45000
1   Canada           42000
2  Germany           52000
3       UK           49000
4   France           47000
```

Step 3: Plot the DataFrame using Pandas

Finally, add the following syntax to the Python code:

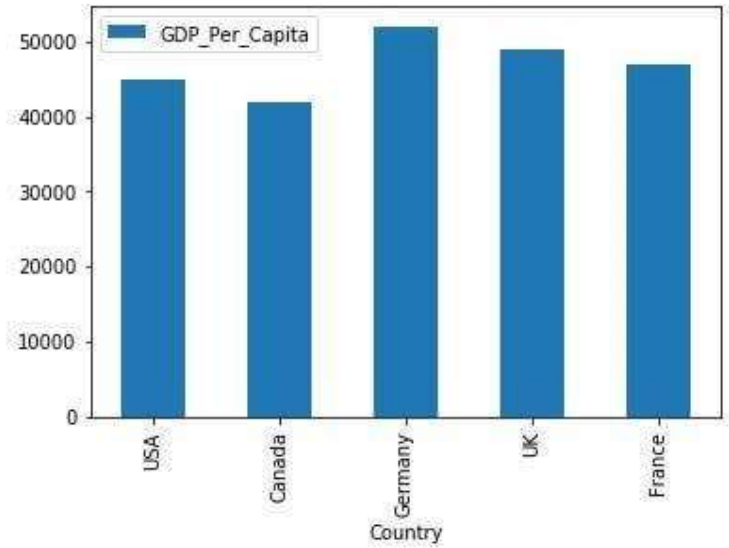
df.plot(x ='Country', y='GDP_Per_Capita', kind = 'bar')

In this case, set the **kind = 'bar'** to plot the bar chart.

```
import pandas as pd
import matplotlib.pyplot as plt
  data = {'Country': ['USA','Canada','Germany','UK','France'],
       'GDP_Per_Capita': [45000,42000,52000,49000,47000]
      }
 df = pd.DataFrame(data,columns=['Country','GDP_Per_Capita'])
df.plot(x ='Country', y='GDP_Per_Capita', kind = 'bar')
plt.show();
```

Run the code and you'll get this bar chart:

© Edunet Foundation. All rights reserved.



Plot a Pie Chart using Pandas

Step 1: Prepare your data

For demonstration purposes, the following data about the status of *tasks* was prepared:

Tasks Pending	300
Tasks Ongoing	500
Tasks Completed	700

The goal is to create a pie chart based on the above data.

Step 2: Create the DataFrame

You can then create the DataFrame using this code:

```
import pandas as pd
data = {'Tasks': [300,500,700]}
df = pd.DataFrame(data,columns=['Tasks'],index = ['Tasks Pending','Tasks Ongoing','Tasks Completed'])
print (df);
```

You'll now see this DataFrame:

```
                 Tasks
Tasks Pending      300
Tasks Ongoing      500
Tasks Completed    700
```

Step 3: Plot the DataFrame using Pandas

Finally, plot the DataFrame by adding the following syntax:

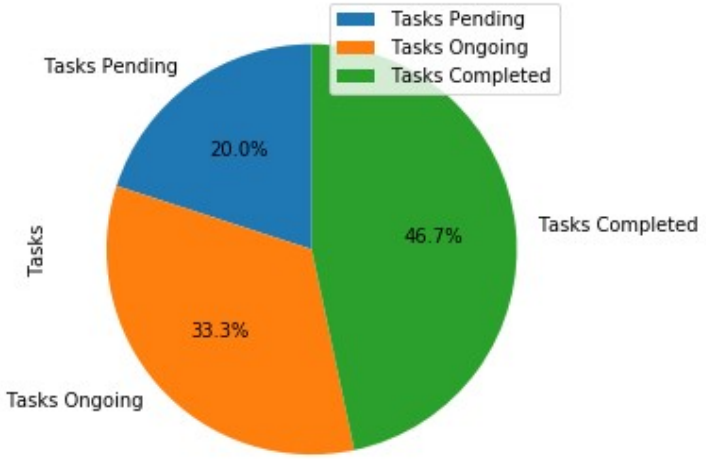
df.plot.pie(y='Tasks',figsize=(5, 5),autopct='%1.1f%%', startangle=90)

The complete code is as follows:

© Edunet Foundation. All rights reserved.

```
import pandas as pd
import matplotlib.pyplot as plt
data = {'Tasks': [300,500,700]}
df = pd.DataFrame(data,columns=['Tasks'],index = ['Tasks Pending','Tasks
Ongoing','Tasks Completed'])
df.plot.pie(y='Tasks',figsize=(5, 5),autopct='%1.1f%%', startangle=90)
plt.show();
```

Once you run the code, you'll get this pie chart: ***[6.15]***



So, in this chapter we have explored about various important concepts of Data analytics and the libraries which are used in data analysis like NumPy, Pandas and Matplotlib. Using these libraries, we can analyse our data and make sense out of data.

Please refer this link [Click Here] for all the codes/practical in this chapter.

2.10 Machine Learning Basics

Machine Learning is undeniably one of the most influential and powerful technologies in today's world. Machine learning is a tool for turning information into knowledge. In the past 50 years, there has been an explosion of data. This mass of data is useless unless we analyze it and find the patterns hidden within. Machine learning techniques are used to automatically find the valuable underlying patterns within complex data that we would otherwise struggle to discover. The hidden patterns and knowledge about a problem can be used to predict future events and perform all kinds of complex decision making. Machine learning is the scientific study of algorithms and statistical models that computer systems use to perform a specific task without using explicit instructions, relying on patterns and inference instead.

In laymen's terms, Machine Learning is about making predictions (answering questions) and classifications based on data. The more data you have, the easier it

© Edunet Foundation. All rights reserved.

will be to recognize patterns and inferences. The data is used to train a machine learning model, then the model is used to make predictions and answer questions.

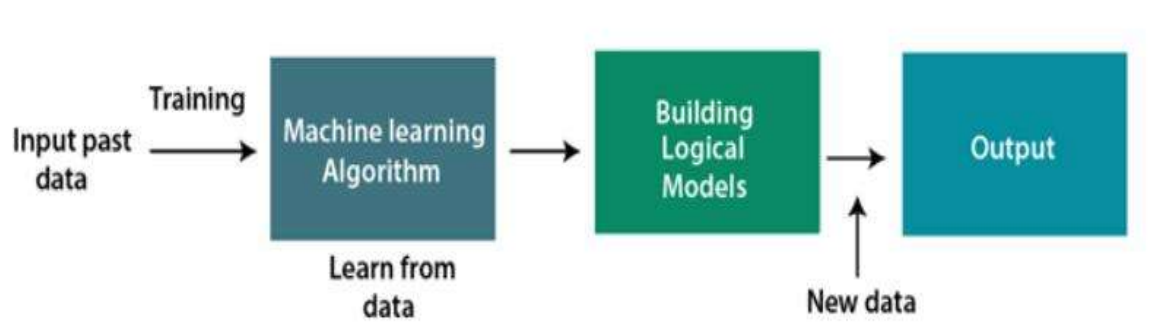


Image: ML working concept
Reference: https://static.javatpoint.com/tutorial/machine-learning/images/introduction-to-machine-learning2.png

Terminology of ML

Dataset: A set of data examples, that contain features important to solving the problem.

Features: Important pieces of data that help us understand a problem. These are fed in to a Machine Learning algorithm to help it learn.

Model: The representation (internal model) of a phenomenon that a Machine Learning algorithm has learnt. It learns this from the data it is shown during training. The model is the output you get after training an algorithm. For example, a decision tree algorithm would be trained and produce a decision tree model.

Process

Data Collection: Collect the data that the algorithm will learn from.

Data Preparation: Format and engineer the data into the optimal format, extracting important features and performing dimensionality reduction.

Training: Also known as the fitting stage, this is where the Machine Learning algorithm actually learns by showing it the data that has been collected and prepared.

Evaluation: Test the model to see how well it performs.

Tuning: Fine tune the model to maximize its performance.

Let us see some real life applications of machine learning in our day-to-day life.

© Edunet Foundation. All rights reserved.

Real time Application of Machine Learning

Machine learning is relevant in many fields, industries, and has the capability to grow over time. Here are six real-life examples of how machine learning is being used.

Image: Applications of ML

1. Image recognition

Image recognition is a well-known and widespread example of machine learning in the real world. It can identify an object as a digital image, based on the intensity of the pixels in black and white images or colour images.

- Real-world examples of image recognition:
- Label an x-ray as cancerous or not
- Assign a name to a photographed face (aka “tagging” on social media)
- Recognise handwriting by segmenting a single letter into smaller images
- Machine learning is also frequently used for facial recognition within an image. Using a database of people, the system can identify commonalities and match them to faces. This is often used in law enforcement.

2. Speech recognition

Machine learning can translate speech into text. Certain software applications can convert live voice and recorded speech into a text file. The speech can be segmented by intensities on time-frequency bands as well.
Real-world examples of speech recognition:

- Voice search
- Voice dialling

© Edunet Foundation. All rights reserved.

- Appliance control

Some of the most common uses of speech recognition software are devices like Google Home or Amazon Alexa.

3. Medical diagnosis

Machine learning can help with the diagnosis of diseases. Many physicians use chatbots with speech recognition capabilities to discern patterns in symptoms.

Real-world examples for medical diagnosis:

- Assisting in formulating a diagnosis or recommends a treatment option
- Oncology and pathology use machine learning to recognise cancerous tissue
- Analyse bodily fluids
- In the case of rare diseases, the joint use of facial recognition software and machine learning helps scan patient photos and identify phenotypes that correlate with rare genetic diseases.

4. Statistical arbitrage

Arbitrage is an automated trading strategy that's used in finance to manage a large volume of securities. The strategy uses a trading algorithm to analyze a set of securities using economic variables and correlations.

Real-world examples of statistical arbitrage:

- Algorithmic trading which analyses a market microstructure
- Analyse large data sets
- Identify real-time arbitrage opportunities

Machine learning optimizes the arbitrage strategy to enhance results.

5. Predictive analytics

Machine learning can classify available data into groups, which are then defined by rules set by analysts. When the classification is complete, the analysts can calculate the probability of a fault.

Real-world examples of predictive analytics:

- Predicting whether a transaction is fraudulent or legitimate
- Improve prediction systems to calculate the possibility of fault

Predictive analytics is one of the most promising examples of machine learning. It's applicable for everything; from product development to real estate pricing.

© Edunet Foundation. All rights reserved.

6. Extraction

Machine learning can extract structured information from unstructured data. Organizations amass huge volumes of data from customers. A machine learning algorithm automates the process of annotating datasets for predictive analytics tools.

Real-world examples of extraction:

- Generate a model to predict vocal cord disorders
- Develop methods to prevent, diagnose, and treat the disorders
- Help physicians diagnose and treat problems quickly

Typically, these processes are tedious. But machine learning can track and extract information to obtain billions of data samples.

Let us see the different techniques in machine learning.

2.11 Techniques of ML

The two main categories of machine learning techniques are supervised learning and unsupervised learning.

Supervised Machine Learning

Supervised learning is the types of machine learning in which machines are trained using well "labelled" training data, and on basis of that data, machines predict the output. The labelled data means some input data is already tagged with the correct output. In supervised learning, the training data provided to the machines work as the supervisor that teaches the machines to predict the output correctly. It applies the same concept as a student learns in the supervision of the teacher.
Supervised learning is a process of providing input data as well as correct output data to the machine learning model. The aim of a supervised learning algorithm is to find a mapping function to map the input variable (x) with the output variable (y).
In the real-world, supervised learning can be used for Risk Assessment, Image classification, Fraud Detection, spam filtering, etc.

© Edunet Foundation. All rights reserved.

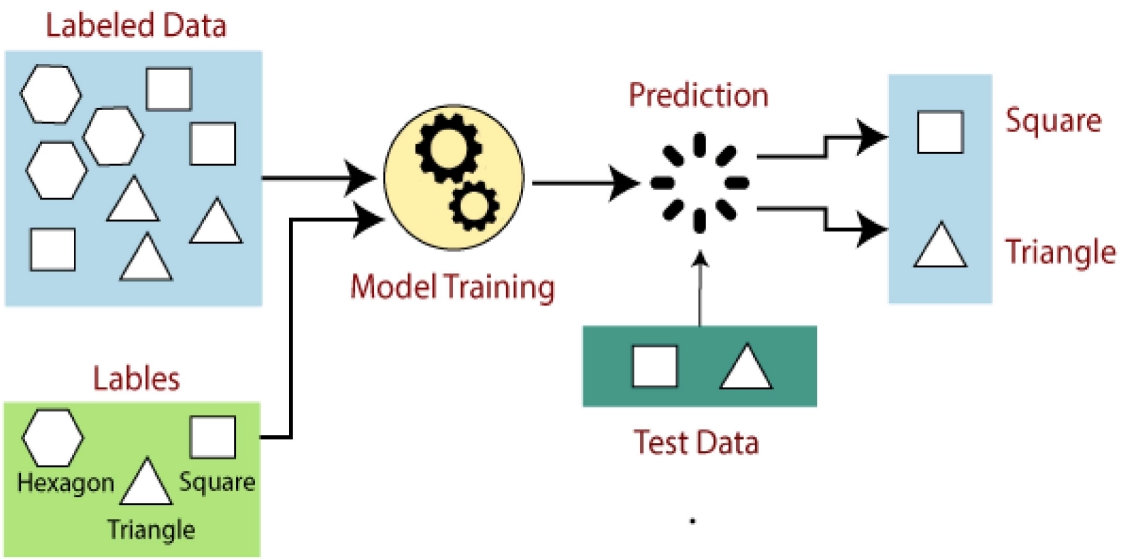


Image: Working of Supervised learning
Reference: https://www.javatpoint.com/supervised-machine-learning

Types of supervised Machine learning Algorithms:

Supervised learning can be further divided into two types of problems:

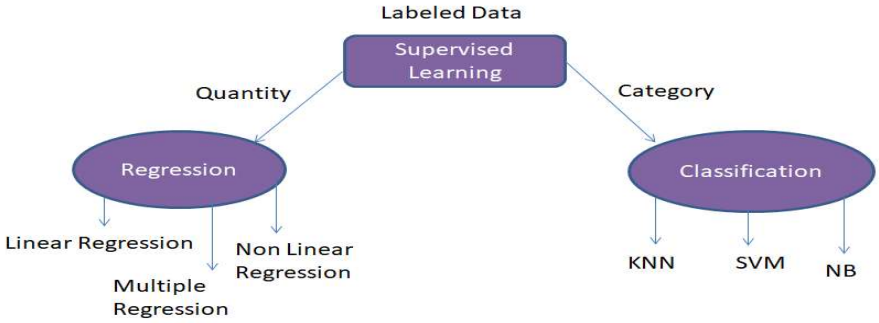


Image: Categories of supervised learning

Unsupervised Machine Learning

Unsupervised learning is a machine learning technique in which models are not supervised using training dataset. Instead, models itself find the hidden patterns and insights from the given data. It can be compared to learning which takes place in the human brain while learning new things.

Unsupervised learning cannot be directly applied to a regression or classification problem because unlike supervised learning, we have the input data but no corresponding output data. The goal of unsupervised learning is to find the underlying

© Edunet Foundation. All rights reserved.

structure of dataset, group that data according to similarities, and represent that dataset in a compressed format.

Example: Suppose the unsupervised learning algorithm is given an input dataset containing images of different types of cats and dogs. The algorithm is never trained upon the given dataset, which means it does not have any idea about the features of the dataset. The task of the unsupervised learning algorithm is to identify the image features on their own. Unsupervised learning algorithm will perform this task by clustering the image dataset into the groups according to similarities between images.

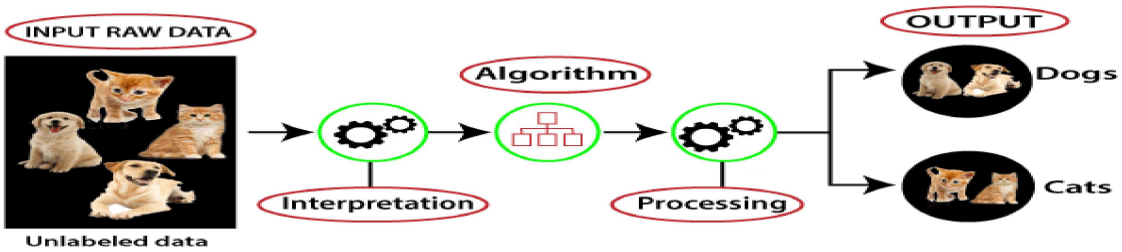


Image: Working of unsupervised learning
Reference: https://www.javatpoint.com/unsupervised-machine-learning

Types of Unsupervised Learning Algorithm:

The unsupervised learning algorithm can be further categorized into two types of problems:

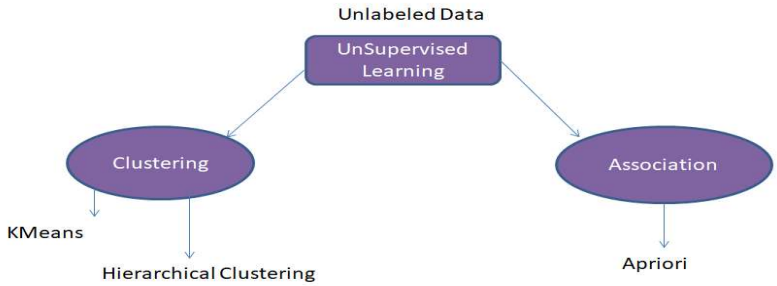


Image: Categories of unsupervised learning

Reinforcement Machine Learning

In this type of machine learning, during the training of algorithm it uses system of rewards and penalty. The learning system, called agent in this context, learns with an interactive environment. The agent selects and performs actions and receives rewards by performing correctly and penalties for performing incorrectly. In reinforcement learning the agent learns by itself, without the intervention from a human. For example, Self-Driving Cars, robots etc.

© Edunet Foundation. All rights reserved.

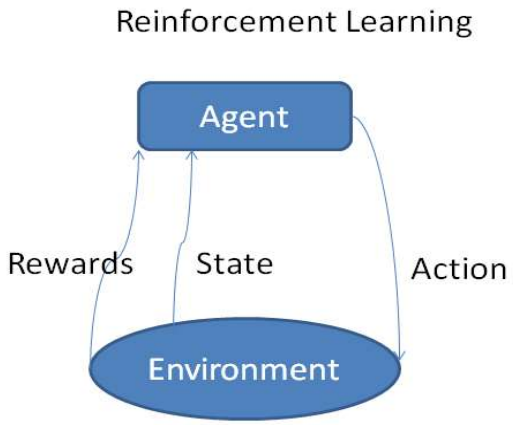


Image: Reinforcement Machine Learning

Now, after understanding the brief overview of machine learning and its types, let us explore a very popular library in machine learning which is used by many machine learning engineers and data scientists to perform various data science and AI projects, named as Scikit-learn. Let us get started with Scikit learn library.

2.12 Scikit Learn library overview

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use.

Scikit Learn is built on top of several common data and math Python libraries. Such a design makes it super easy to integrate between them all. You can pass numpy arrays and pandas data frames directly to the ML algorithms of Scikit! It uses the following libraries:
NumPy: For any work with matrices, especially math operations
SciPy: Scientific and technical computing
Matplotlib: Data visualization
IPython: Interactive console for Python
Sympy: Symbolic mathematics
Pandas: Data handling, manipulation, and analysis
Scikit Learn is focused on Machine Learning, e.g data modelling. It is not concerned with the loading, handling, manipulating, and visualizing of data. Thus, it is natural and common practice to use the above libraries, especially NumPy, for those extra steps; they are made for each other!

© Edunet Foundation. All rights reserved.

Scikit's robust set of algorithm offerings include:

- Regression: Fitting linear and non-linear models
- Clustering: Unsupervised classification
- Decision Trees: Tree induction and pruning for both classification and regression tasks
- Neural Networks: End-to-end training for both classification and regression. Layers can be easily defined in a tuple
- SVMs: for learning decision boundaries
- Naive Bayes: Direct probabilistic modelling
- Even beyond that, it has some very convenient and advanced functions not commonly offered by other libraries:
- Ensemble Methods: Boosting, Bagging, Random Forest, Model voting and averaging
- Feature Manipulation: Dimensionality reduction, feature selection, feature analysis
- Outlier Detection: For detecting outliers and rejecting noise
- Model selection and validation: Cross-validation, Hyperparamter tuning, and metrics.

Now we have discussed about the machine learning library sklearn , let us start with the basic algorithms in machine learning.

2.12 Regression vs Classification

Regression and Classification algorithms are Supervised Learning algorithms. Both the algorithms are used for prediction in Machine learning and work with the labelled datasets. But the difference between both is how they are used for different machine learning problems.

The main difference between Regression and Classification algorithms is that Regression algorithms are used to predict the continuous values such as price, salary, age, etc. whereas Classification algorithms are used to predict/Classify the discrete values such as Male or Female, True or False, Spam or Not Spam, etc.

© Edunet Foundation. All rights reserved.

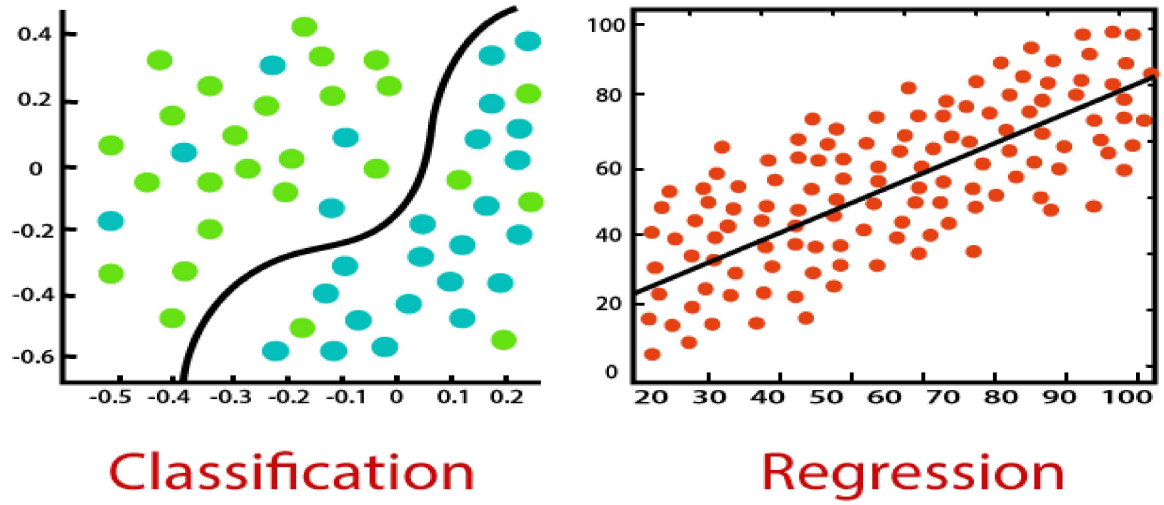


Image: Classification vs Regression

Reference: https://www.javatpoint.com/regression-vs-classification-in-machine-learning

Classification

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.

Types of ML Classification Algorithms:

Classification Algorithms can be further divided into the following types:

i) Logistic Regression
ii) K-Nearest Neighbours
iii) Support Vector Machines
iv) Kernel SVM
v) Naïve Bayes
vi) Decision Tree Classification
vii) Random Forest Classification

© Edunet Foundation. All rights reserved.

Regression:

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.

Types of Regression Algorithm:

i) Simple Linear Regression
ii) Multiple Linear Regression
iii) Polynomial Regression
iv) Support Vector Regression
v) Decision Tree Regression
vi) Random Forest Regression

Before starting the linear regression algorithm, let us get started with least square method which is used as a backbone of regression analysis.

2.13 Least Square Method

The least-squares regression method is a technique commonly used in Regression Analysis. It is a mathematical method used to find the best fit line that represents the relationship between an independent and dependent variable.

Line of Best Fit

Line of best fit is drawn to represent the relationship between 2 or more variables. To be more specific, the best fit line is drawn across a scatter plot of data points in order to represent a relationship between those data points.

Regression analysis makes use of mathematical methods such as least squares to obtain a definite relationship between the predictor variable (s) and the target variable. The least-squares method is one of the most effective ways used to draw the line of best fit. It is based on the idea that the square of the errors obtained must be minimized to the most possible extent and hence the name least squares method.

Least Squares Regression Example

© Edunet Foundation. All rights reserved.

Consider an example. Tom who is the owner of a retail shop, found the price of different T-shirts vs the number of T-shirts sold at his shop over a period of one week.

He tabulated this like:

Price of T-shirts in dollars (x)	# of T-shirts sold (y)
2	4
3	5
5	7
7	10
9	15

Let us use the concept of least squares regression to find the line of best fit for the above data.

Step 1: Calculate the slope 'm' by using the following formula, where n is the number of observations:

$$m = \frac{n\sum xy - (\Sigma x)(\Sigma y)}{n\Sigma x^2 - (\Sigma x)^2}$$

After you substitute the respective values, m = 1.518 approximately.

Step 2: Compute the y-intercept value

$$\mathbf{c} = \frac{(\sum \mathbf{y} \sum \mathbf{x^2}) - \sum \mathbf{x} \sum \mathbf{xy}}{\mathbf{n}(\sum \mathbf{x^2}) - (\sum \mathbf{x})^2}$$

After you substitute the respective values, c = 0.305 approximately.

Step 3: Substitute the values in the final equation

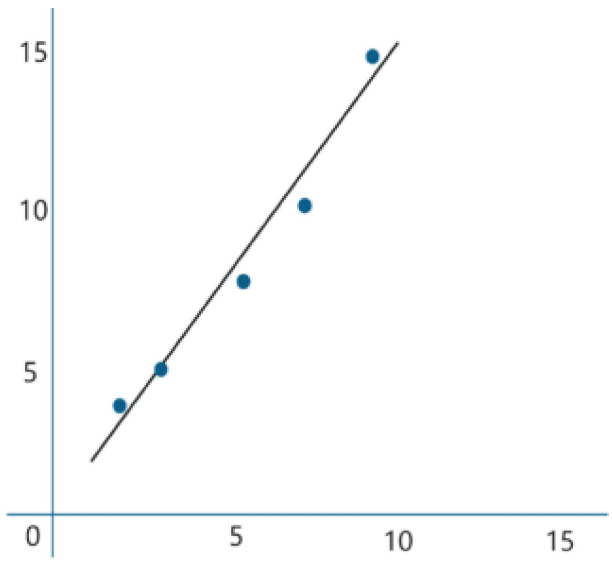
© Edunet Foundation. All rights reserved.

$$y = mx + c$$

Once you substitute the values, it should look something like this:

Price of T-shirts in dollars (x)	# of T-shirts sold (y)	Y= mx + c	Error (Y-y)
2	4	3.341	-0.659
3	5	4.859	-0.141
5	7	7.895	0.895
7	10	10.931	0.931
9	15	13.967	-1.033

Let's construct a graph that represents the y=mx + c line of best fit:



Now, Tom can use the above equation to estimate how many T-shirts of price $8 can he sell at the retail shop.

y = 1.518 x 8 + 0.305 = 12.45 T-shirts

The least squares regression method works by minimizing the sum of the square of the errors as small as possible, hence the name least squares. Basically, the distance between the line of best fit and the error must be minimized as much as possible.

A few things to keep in mind before implementing the least squares regression method is:

© Edunet Foundation. All rights reserved.

- The data must be free of outliers because they might lead to a biased and wrongful line of best fit.
- The line of best fit can be drawn iteratively until you get a line with the minimum possible squares of errors.
- This method works well even with non-linear data.
- Technically, the difference between the actual value of 'y' and the predicted value of 'y' is called the Residual (denotes the error).

Linear Regression

Linear regression is one of the easiest and most popular Machine Learning algorithms. It is a statistical method that is used for predictive analysis. Linear regression makes predictions for continuous/real or numeric variables such as sales, salary, age, product price, etc.

Linear regression algorithm shows a linear relationship between a dependent (y) and one or more independent (x) variables, hence called as linear regression. Since linear regression shows the linear relationship, which means it finds how the value of the dependent variable is changing according to the value of the independent variable.

The linear regression model provides a sloped straight line representing the relationship between the variables.

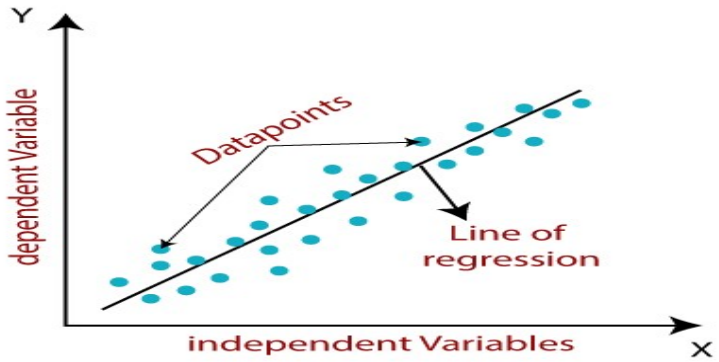


Image: Relationship between the variables in linear regression
Reference:https://www.javatpoint.com/linear-regression-in-machine-learning

Mathematically, we can represent a linear regression as:

$$y = a0 + a1x + \varepsilon$$

Here,
y= Dependent Variable (Target Variable)
x= Independent Variable (predictor Variable)
a0= intercept of the line (Gives an additional degree of freedom)
a1 = Linear regression coefficient (scale factor to each input value).

© Edunet Foundation. All rights reserved.

ε = random error

The values for x and y variables are training datasets for Linear Regression model representation.

Types of Linear Regression

Linear regression can be further divided into two types of the algorithm:

1. Simple Linear Regression:

If a single independent variable is used to predict the value of a numerical dependent variable, then such a Linear Regression algorithm is called Simple Linear Regression.

2. Multiple Linear regression:

If more than one independent variable is used to predict the value of a numerical dependent variable, then such a Linear Regression algorithm is called Multiple Linear Regression.

Linear Regression Line

A linear line showing the relationship between the dependent and independent variables is called a regression line. A regression line can show two types of relationship:

- Positive Linear Relationship:

If the dependent variable increases on the Y-axis and independent variable increases on X-axis, then such a relationship is termed as a Positive linear relationship.

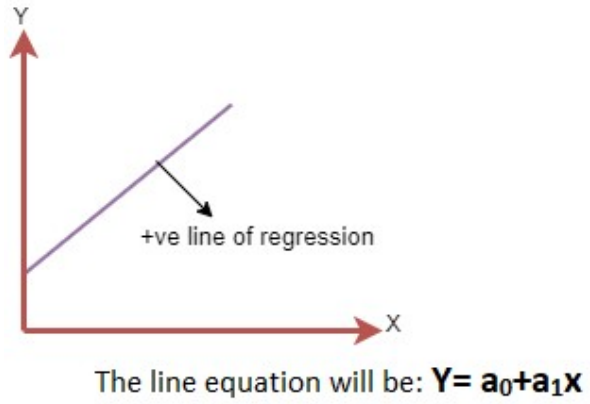


Image:Positive linear relationship

Reference: https://www.javatpoint.com/linear-regression-in-machine-learning

- Negative Linear Relationship:

If the dependent variable decreases on the Y-axis and independent variable increases on the X-axis, then such a relationship is called a negative linear relationship.

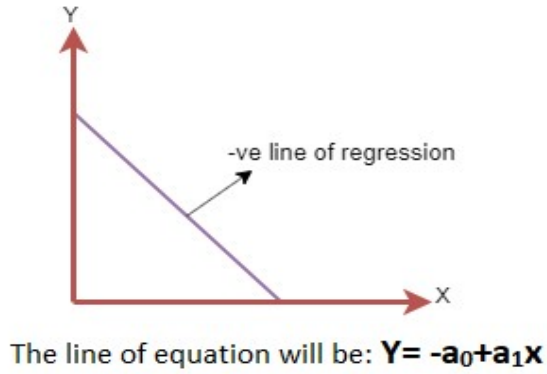


Image:Negative linear relationship
Reference: https://www.javatpoint.com/linear-regression-in-machine-learning

Ordinary Least Square Method

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.

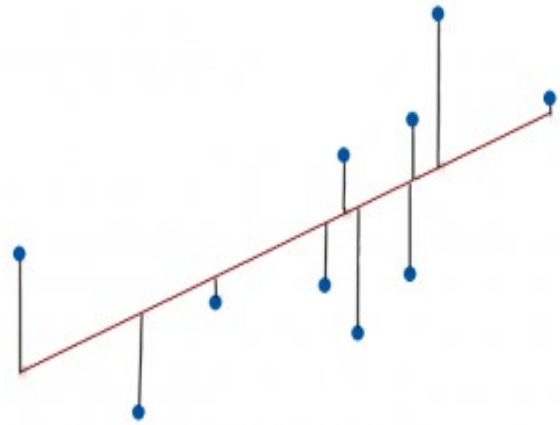


Image: Ordinary least squares line
Reference: https://statisticsbyjim.com/glossary/ordinary-least-squares/#:~:text=Ordinary%20least%20squares%2C%20or%20linear,and%20the%20corresponding%20fitted%20values

- 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.

Project : Linear Regression in Sklearn

Let us see how to build a simple linear machine learning model for the Diabetes dataset. This dataset consists of Ten baseline variables, age, sex, body mass index, average blood pressure, and six blood serum measurements were obtained for each of n = 442 diabetes patients, as well as the response of interest, a quantitative measure of disease progression one year after baseline. We will apply Linear Regression for this considering one input feature and output target.

Link for Project :

https://github.com/Edunet-Foundation/Tech-Saksham/blob/main/Tech%20Saksham%20AI%20Practical%20-%20Notebooks/Chapter%204/1.Linear%20Regression.ipynb

© Edunet Foundation. All rights reserved.

We have discussed about regression, ordinary least square method, and metrics to measure the performance of regression. Now, let's get started with the cloud platform where we can train, test and deploy our machine learning model.

2.14 Azure ML No Code Platform

Machine Learning is the foundation for most artificial intelligence solutions, and the creation of an intelligent solution often begins with the use of machine learning to train a predictive model using historic data that you have collected. Azure Machine Learning is a cloud service that you can use to train and manage machine learning models.

Azure Machine Learning is a cloud service that allows building no-code machine learning models through a drag and drop visual interface. It's designed to help data scientists and machine learning engineers to leverage their existing data processing and model development skills & frameworks.

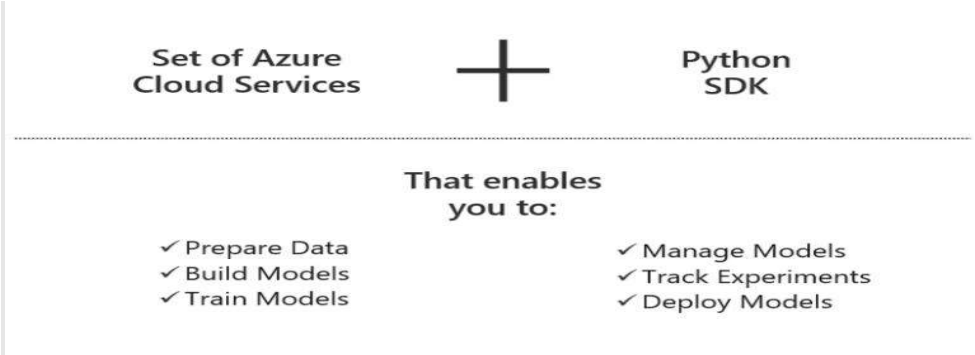


Image: Azure Machine Learning

Reference: https://docs.microsoft.com/en-us/azure/machine-learning/overview-what-is-azure-ml?WT.mc_id=aiml-0000-abornst

Azure ML Studio Briefing

Azure Machine Learning Studio is web-based integrated development environment (IDE) for developing data experiments. It is closely knit with the rest of Azure's cloud services and that simplifies development and deployment of machine learning models and services.

Creating the Experiment

There are five basic steps to creating a machine learning example.

OBTAINING THE DATA

Gathering data is one of the most important step in this process. Relevance and clarity of the data are the basis for creating good prediction models. Azure Machine Learning Studio provides a number of sample data sets.

© Edunet Foundation. All rights reserved.

- After collecting the data, we need to upload it to the Studio through their simple data upload mechanism:
- Our next step is to create a new experiment by dragging and dropping modules from the panel on the left into the working area.

PREPROCESSING DATA

Preprocessing available data involves adjusting the available data to your needs. The first module that we will use here is “Descriptive Statistics”. It computes statistical data from the available data. Besides “Descriptive Statistics” module, one of the commonly used modules is “Clean Missing Data”. The aim of this step is to give meaning to missing (null) values by replacing it with some other value or by removing them entirely.

DEFINING FEATURES

This module determines the features of the dataset that are most relevant to the results that we want to predict.

CHOOSING AND APPLYING AN ALGORITHM

Next step is to split the available data using the “Split” module. The first part of the data will be used to train the model and the rest is used to score the trained model.

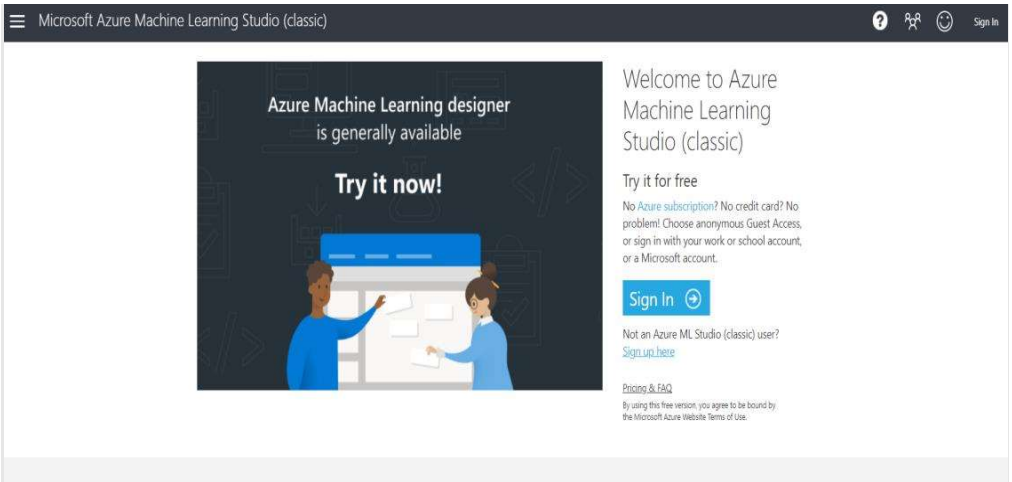
- The following steps are the most important steps in the entire Azure machine learning process. The module “Train Model” accepts two input parameters. First is the raw training data, and the other is the learning algorithm.
- Evaluate Model module gives us an evaluation of the trained model expressed in statistical values.

PREDICTING NEW DATA

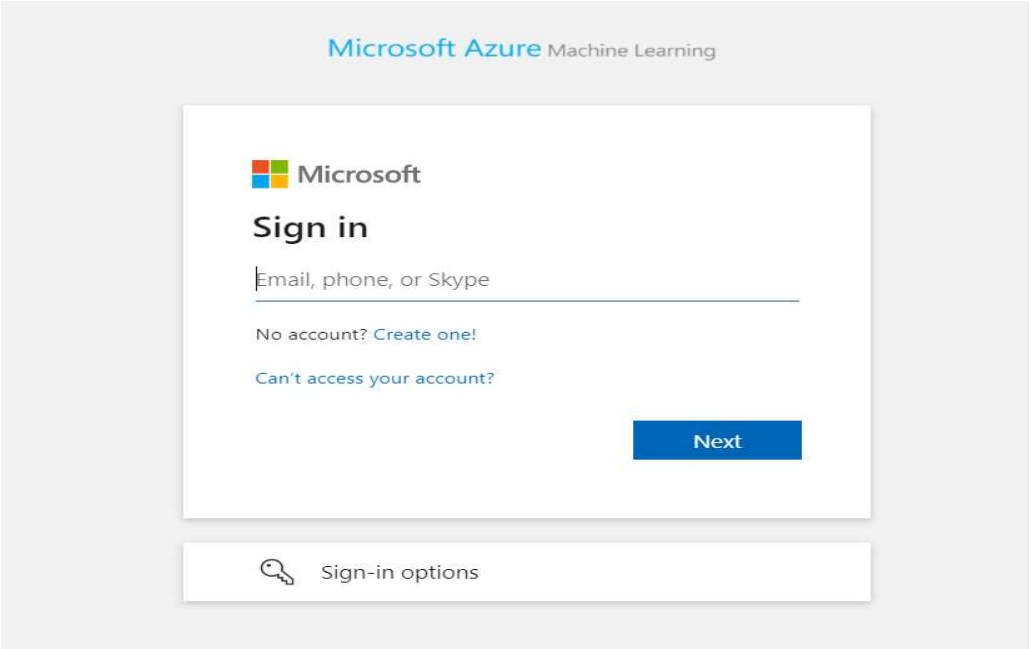
Finally, we can test our prediction web service using a simple test form.

2.15 Setting up an account in Azure ML Studio and Creating Workspace

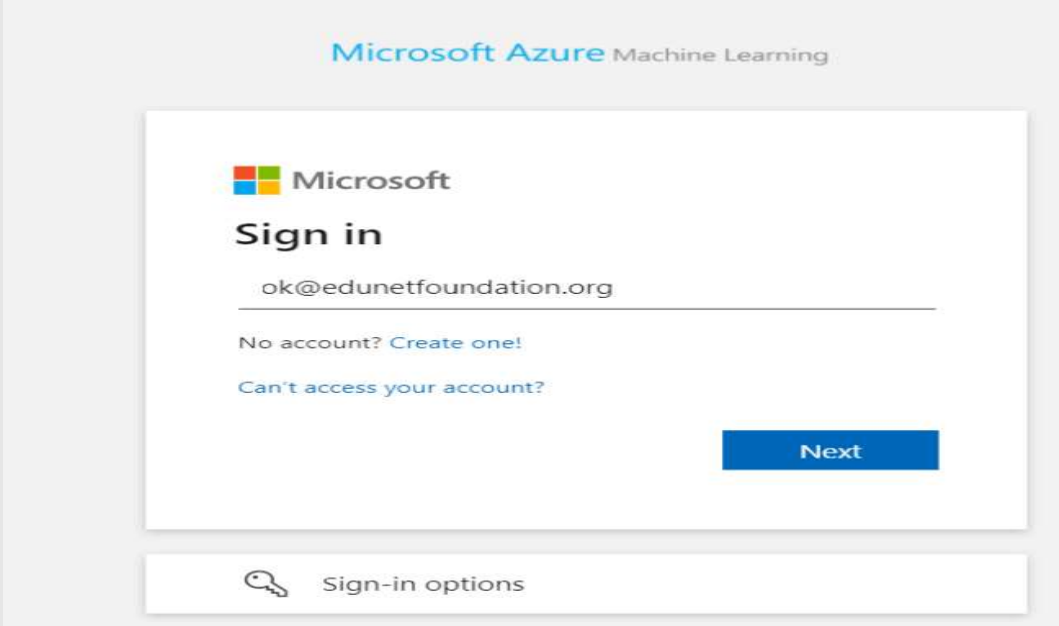
- Open the following web link on the web browser.
  https://studio.azureml.net/



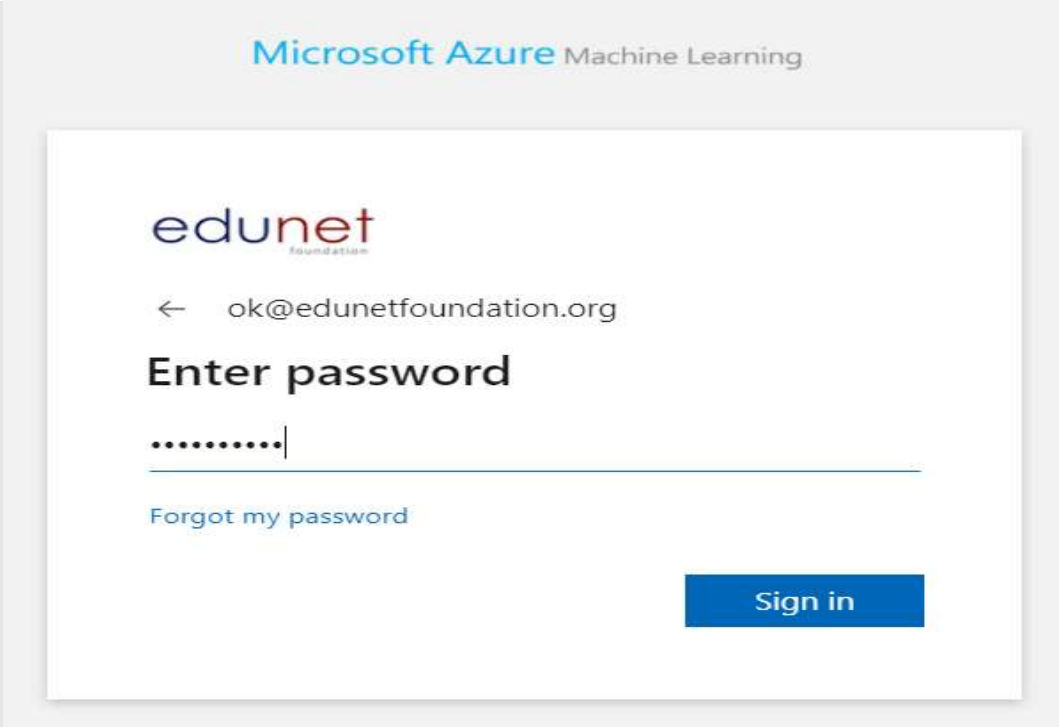
- Click on the Sign In.



- Type your Microsoft email ID and then press next. If you do not have the Microsoft account then create first using **create one** option (https://signup.live.com/).

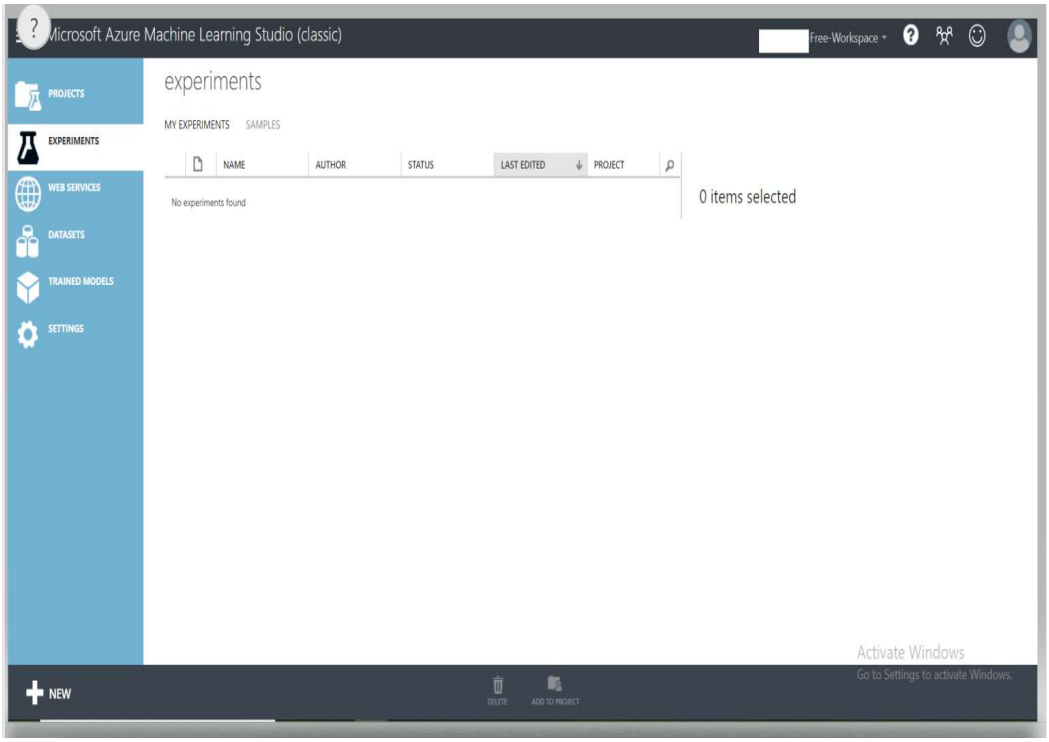


- Type the password for your Microsoft account and press sign in.



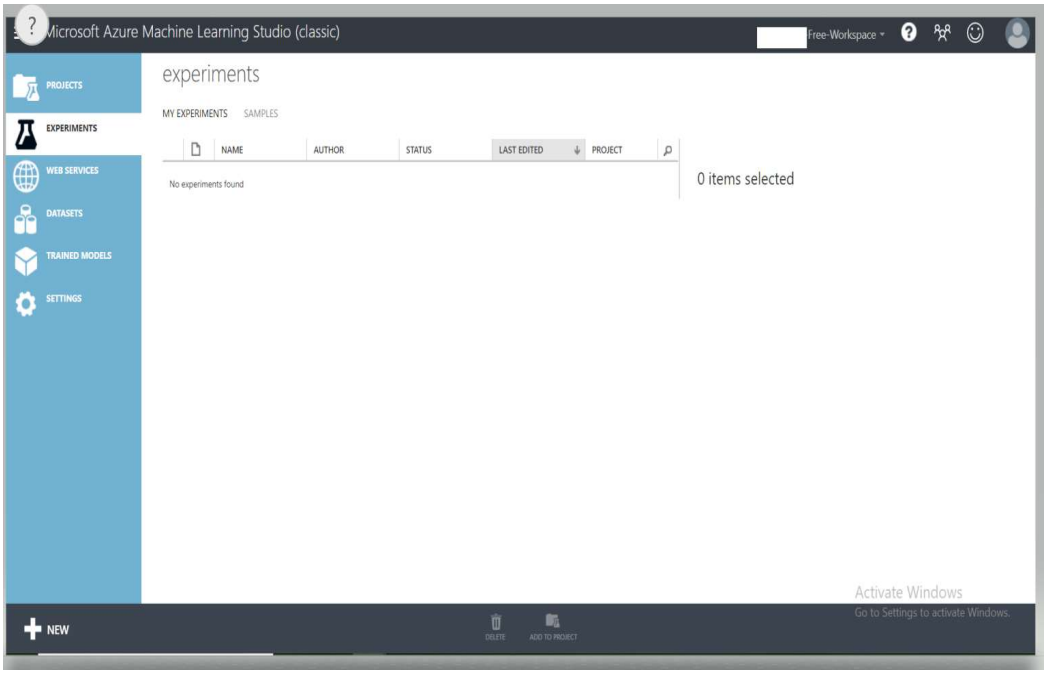
- Press Yes to stay sign in.

- Now, You will be redirected into following Microsoft Azure Machine Learning Studio (Classic) and your free workspace will be created as below:

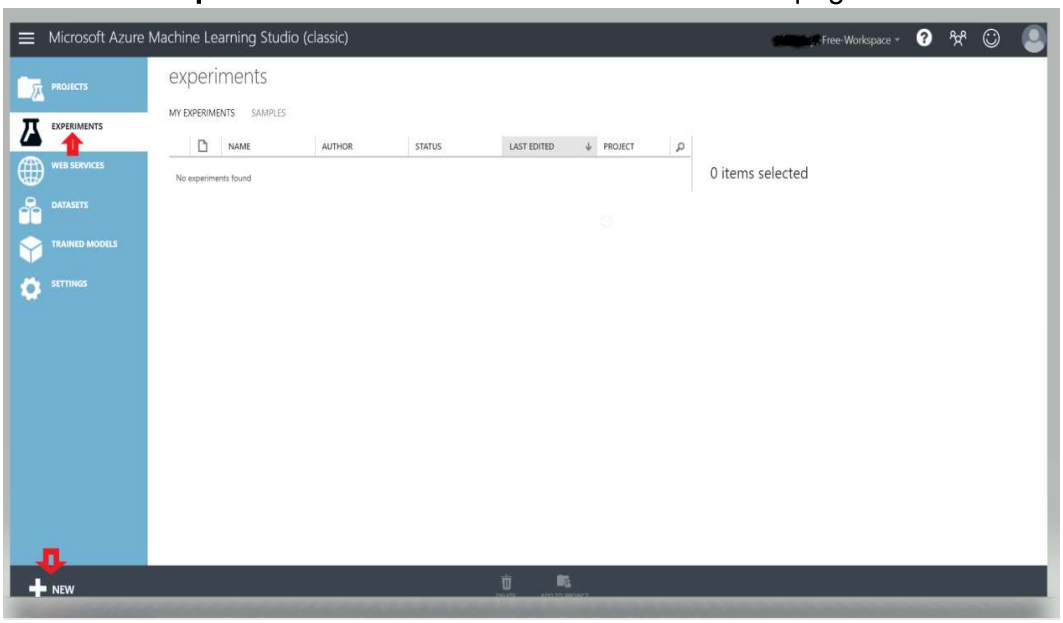


2.16 Training a ML model in Azure Studio

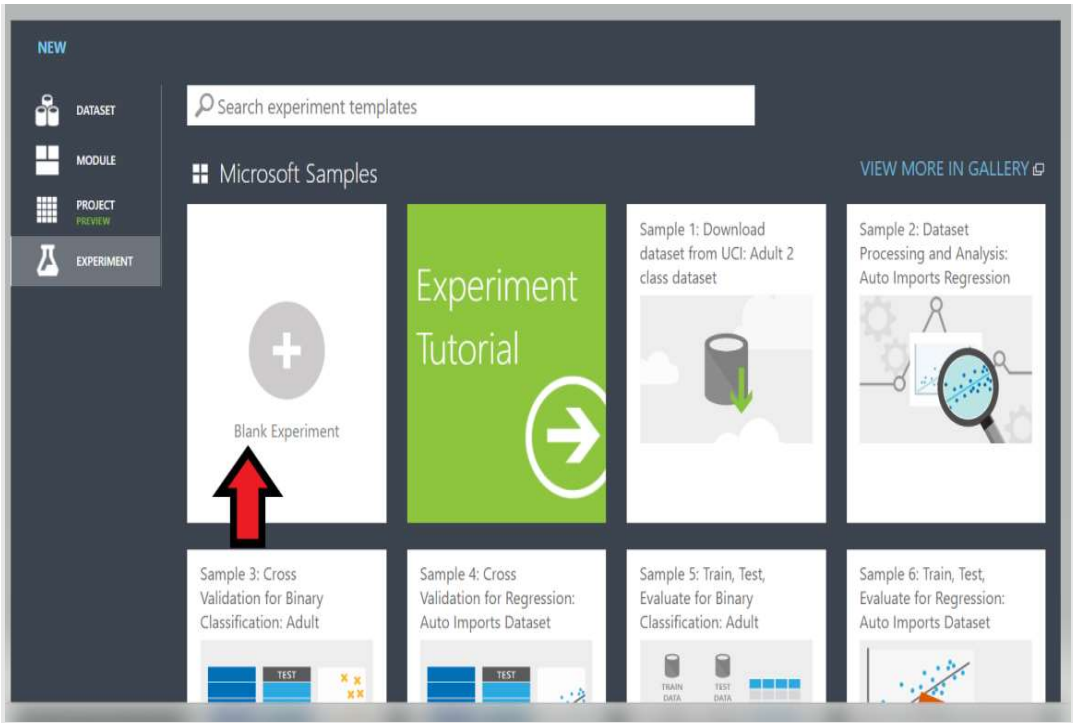
- Sign in into Microsoft Azure Machine Learning Studio (classic) and create workspace.



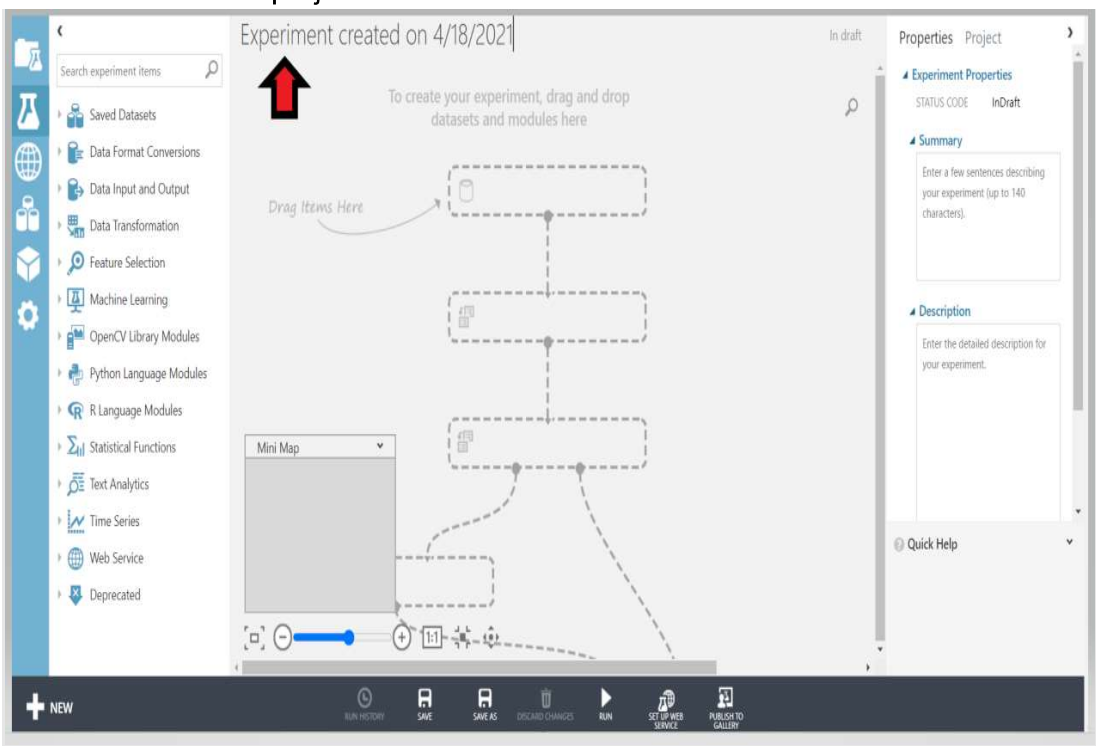
- First select **Experiment** and then **New** at the bottom of the page.



- Select Blank Experiment.

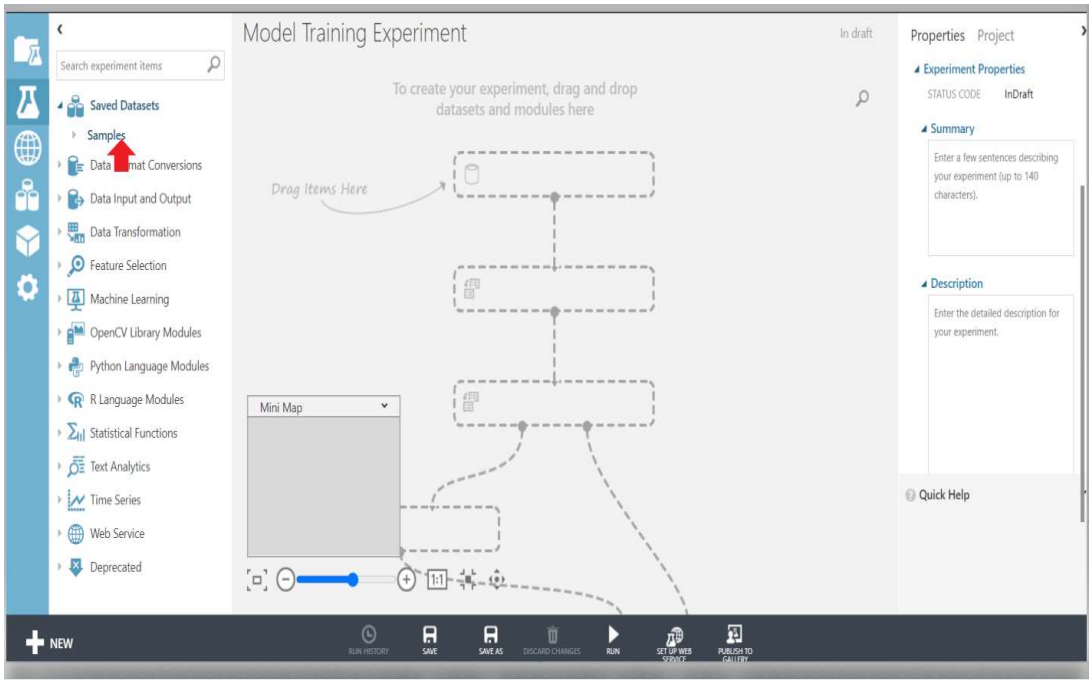


- Give the title for the project.

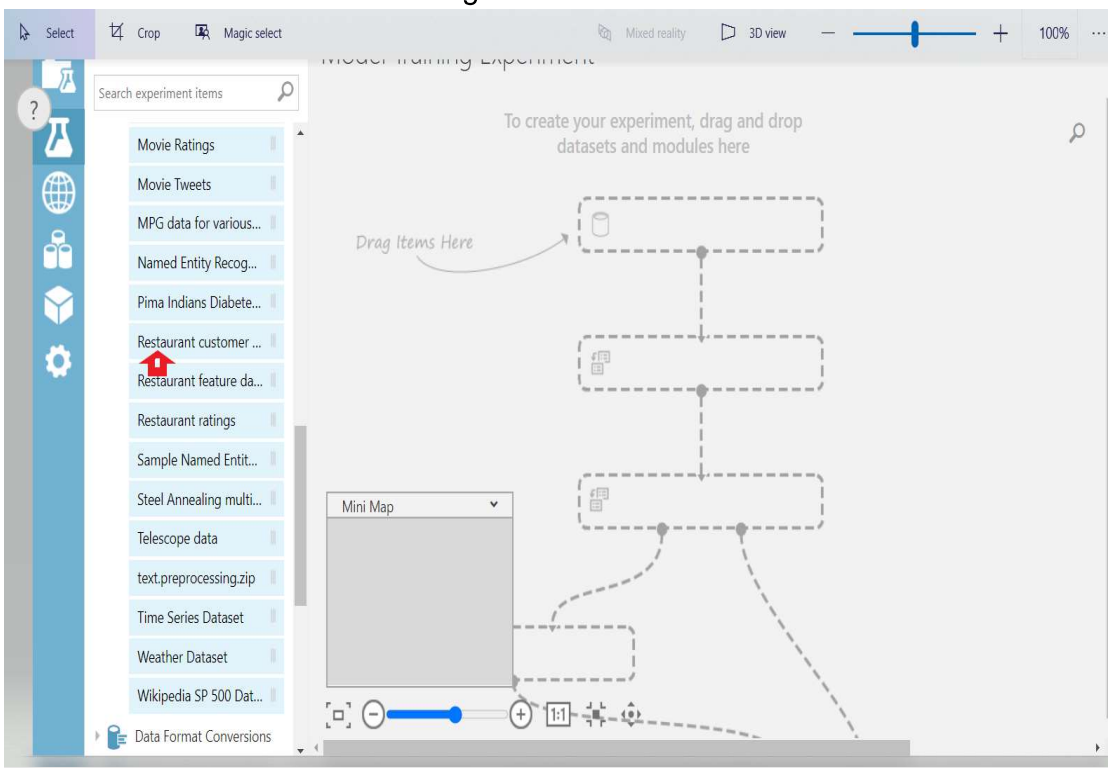


- Select **Sample** option from the **Saved Dataset.**

© Edunet Foundation. All rights reserved.

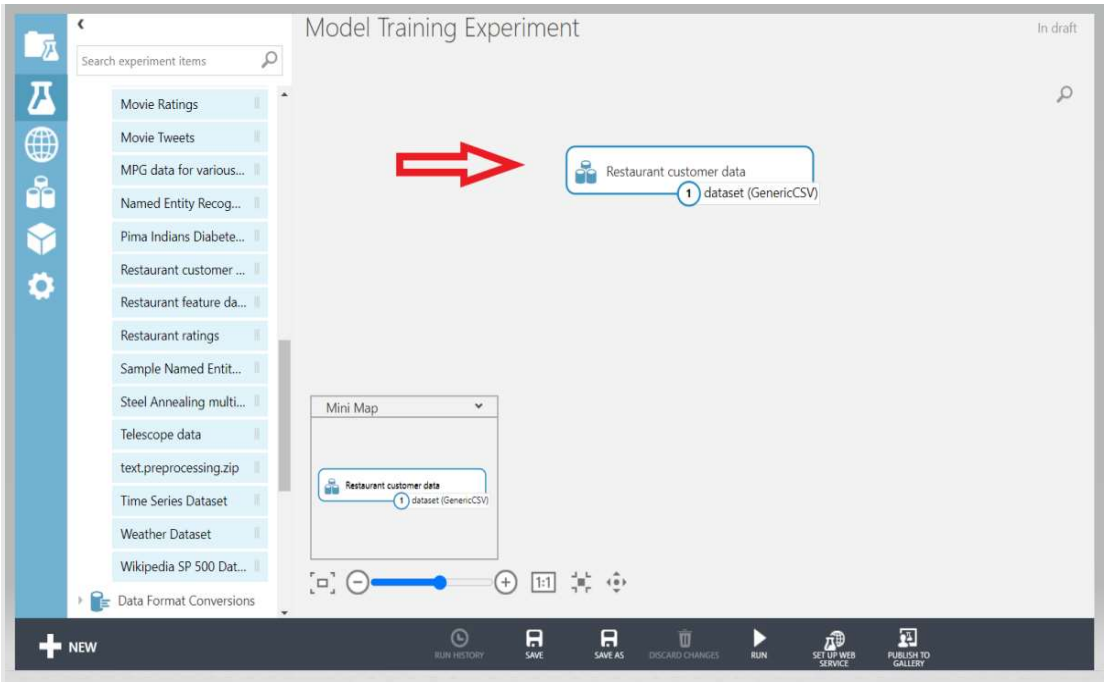


- Select Restaurant Customer Rating Dataset

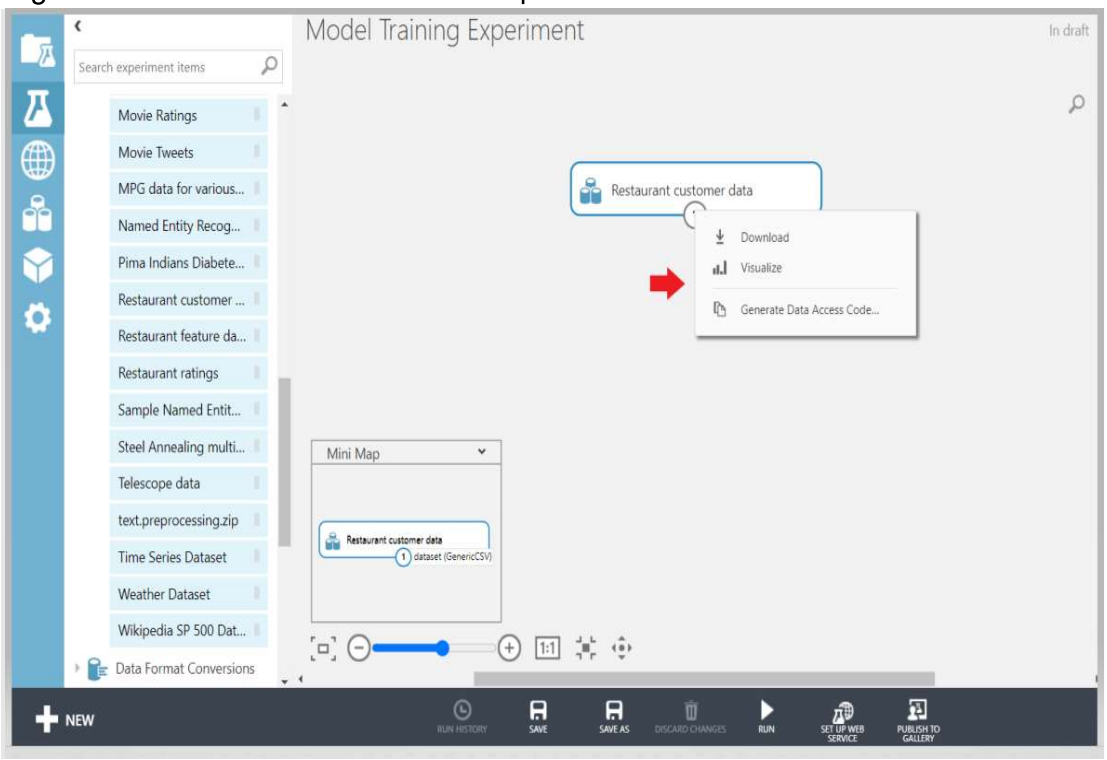


- Drag selected dataset on Panel.

© Edunet Foundation. All rights reserved.

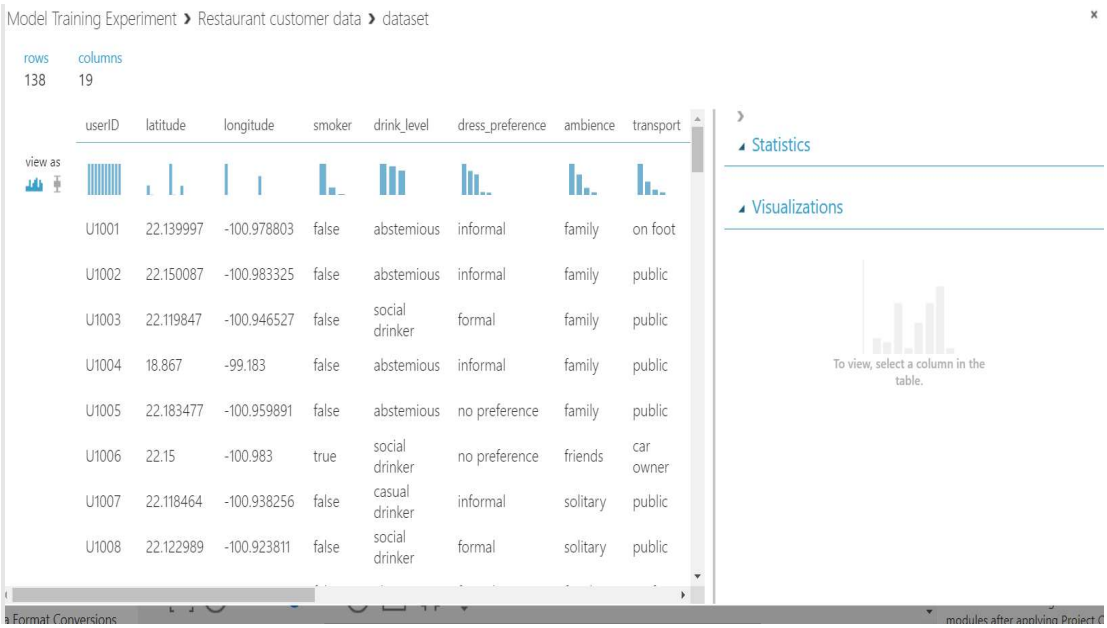


- Right click on 1 and choose visualize option.



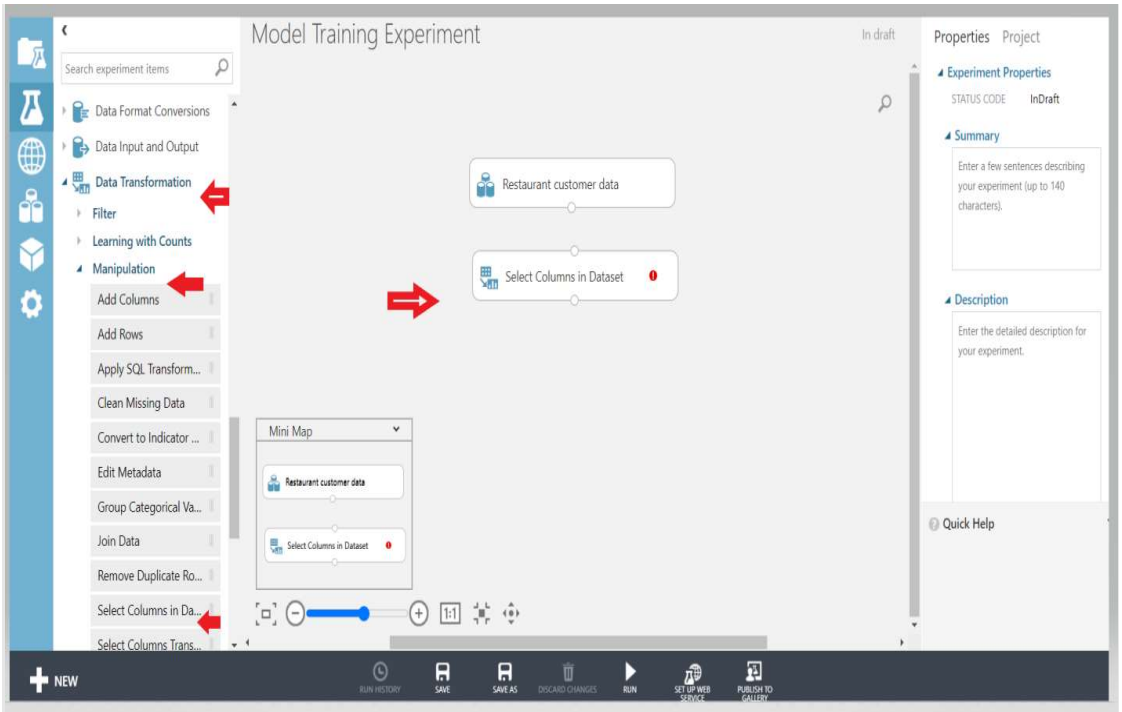
- Visualize the Dataset and then close it:

© Edunet Foundation. All rights reserved.

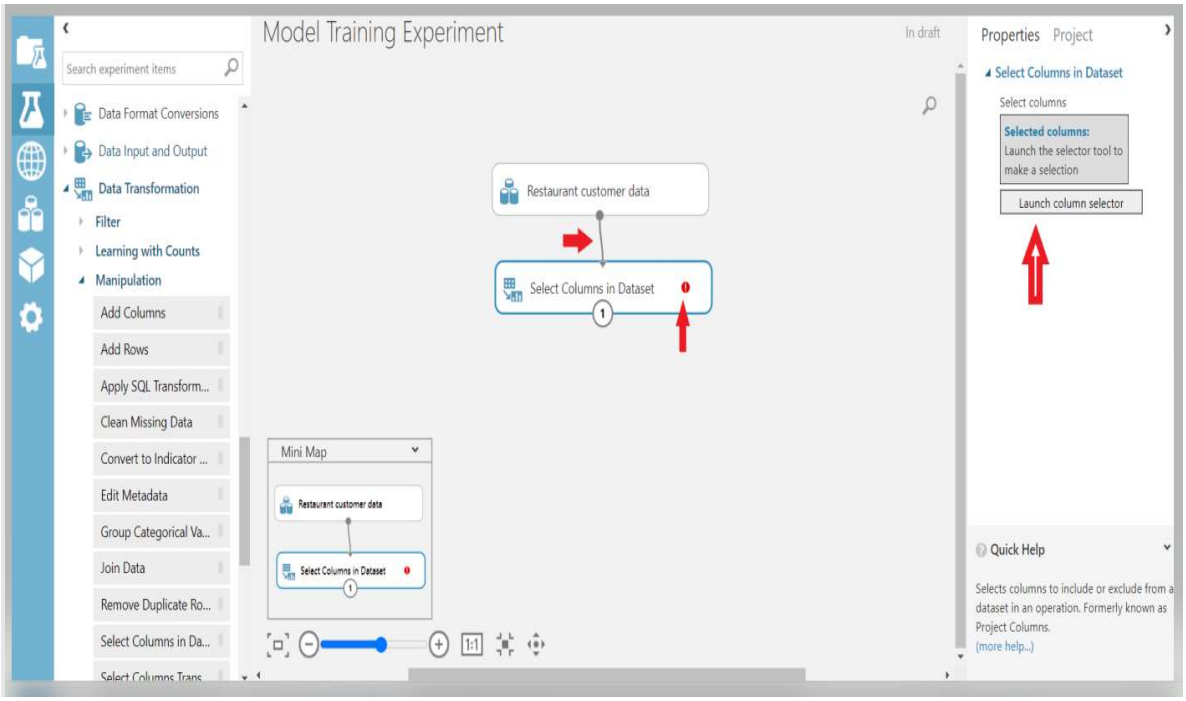


userID	latitude	longitude	smoker	drink_level	dress_preference	ambience	transport
U1001	22.139997	-100.978803	false	abstemious	informal	family	on foot
U1002	22.150087	-100.983325	false	abstemious	informal	family	public
U1003	22.119847	-100.946527	false	social drinker	formal	family	public
U1004	18.867	-99.183	false	abstemious	informal	family	public
U1005	22.183477	-100.959891	false	abstemious	no preference	family	public
U1006	22.15	-100.983	true	social drinker	no preference	friends	car owner
U1007	22.118464	-100.938256	false	casual drinker	informal	solitary	public
U1008	22.122989	-100.923811	false	social drinker	formal	solitary	public

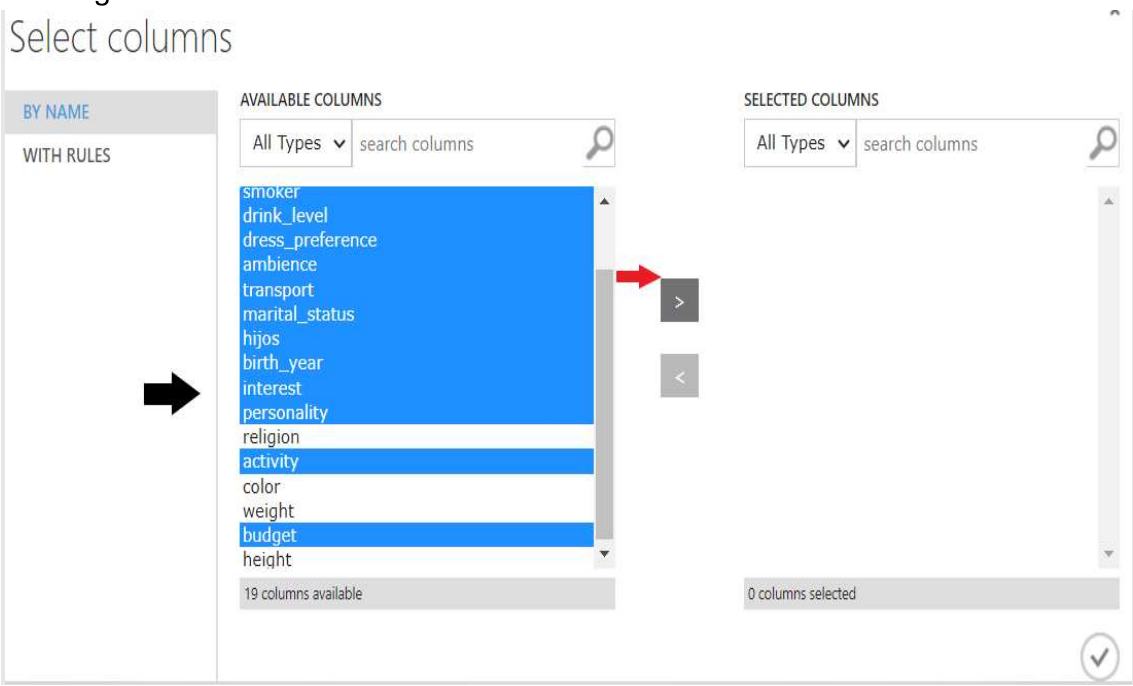
- Select the Data Transformation → Manipulation → Select Columns in Dataset and drag it into Panel.



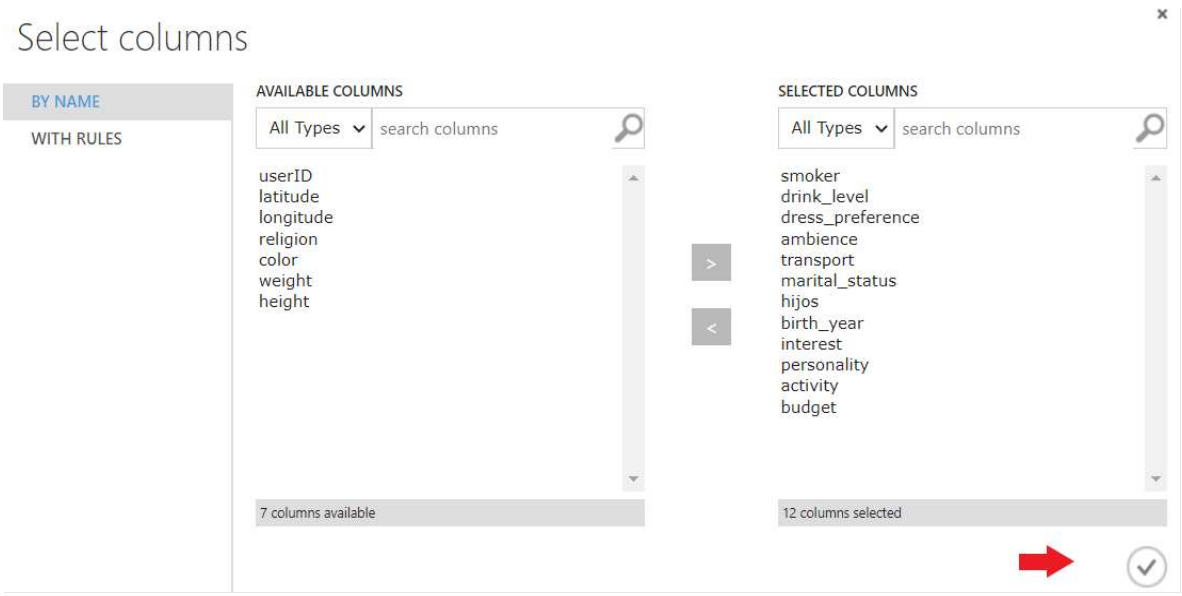
- Make connection between dragged item, press on red sign and then Launch column selector to choose the relevant column.



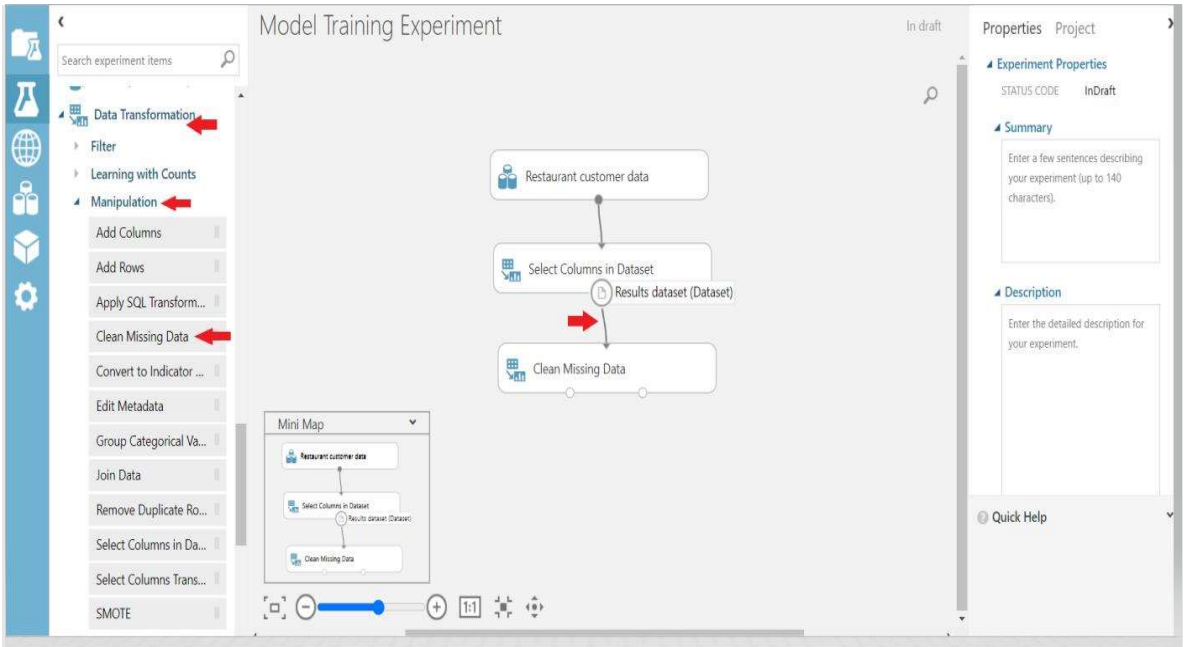
- Select all the relevant column by using **Ctrl Key** then use **>** option to move them right.



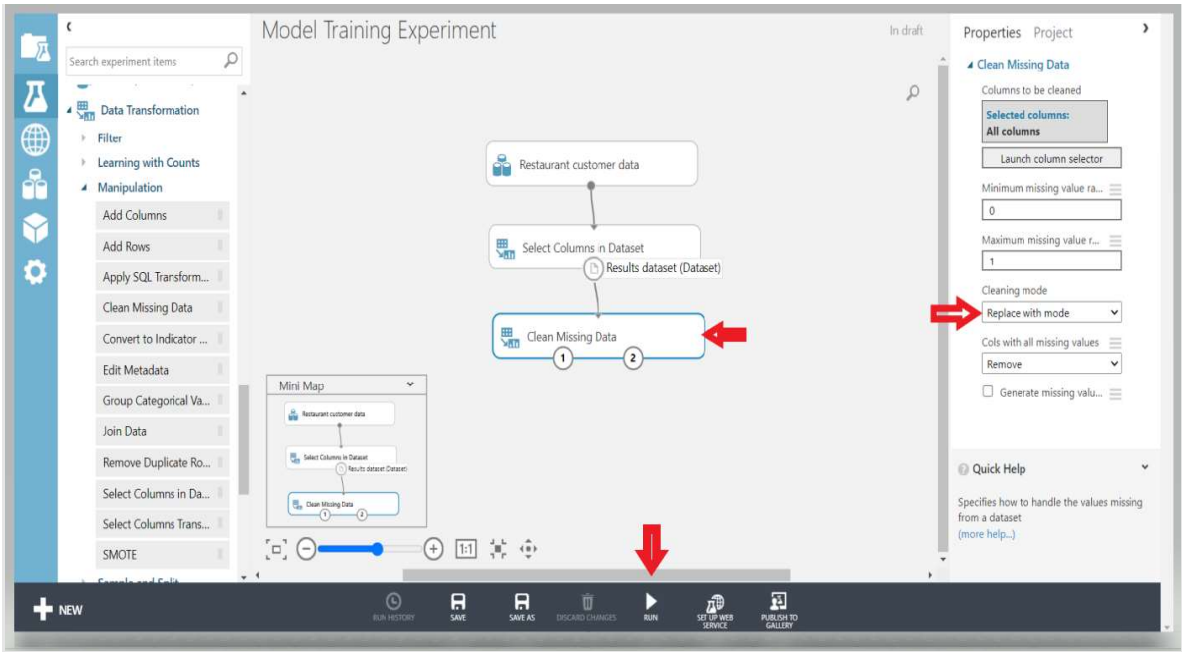
- Now click on Tick sign at the bottom of the page.



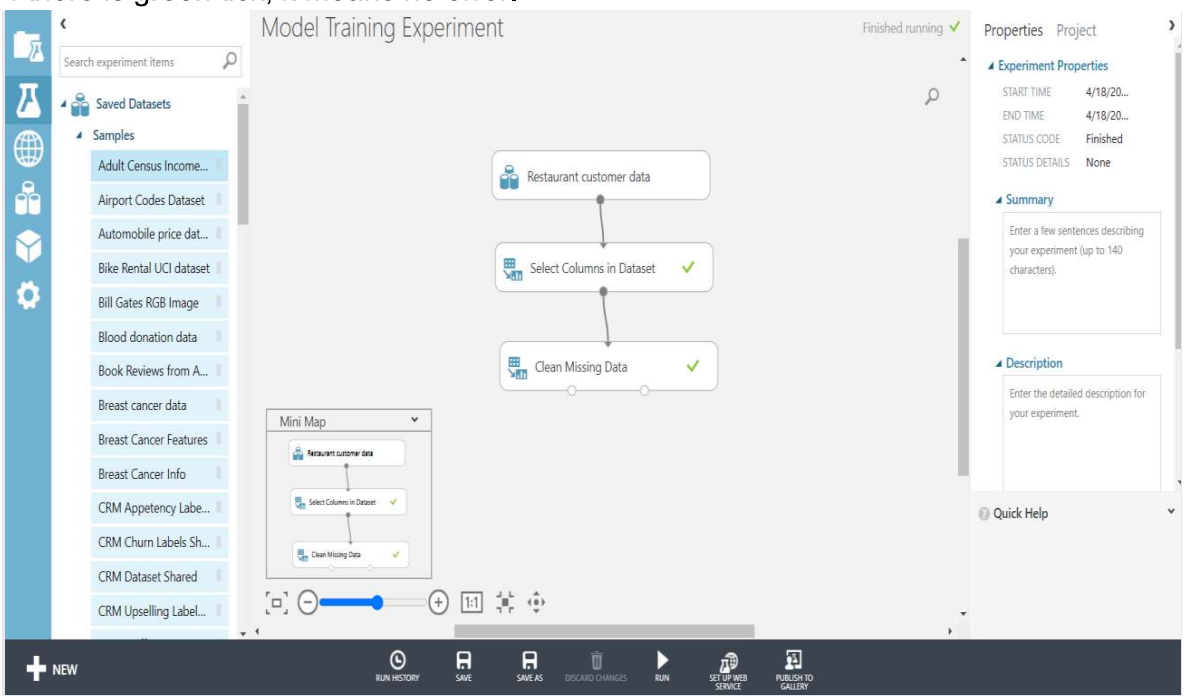
- Select the Data Transformation → Manipulation → Clean missing data, drag it into Panel and make connection.



- Click on Clean missing data, set Replace with mode in cleaning mode and press Run.

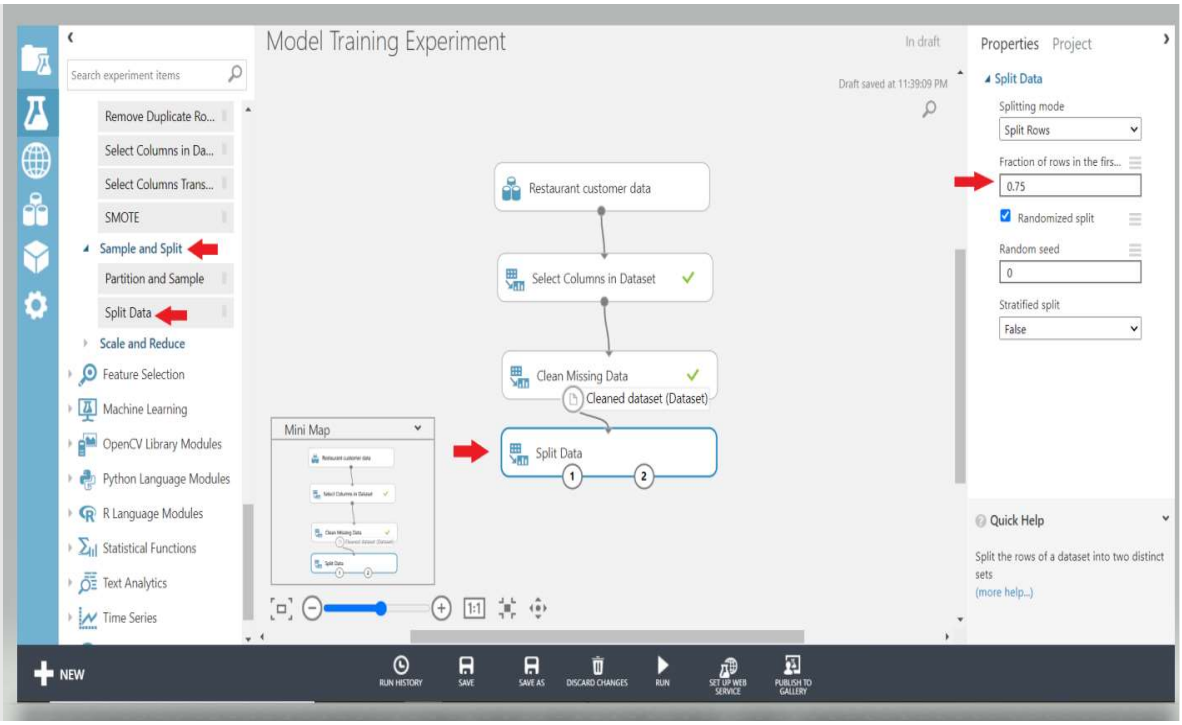


- If there is green tick, it means no error.

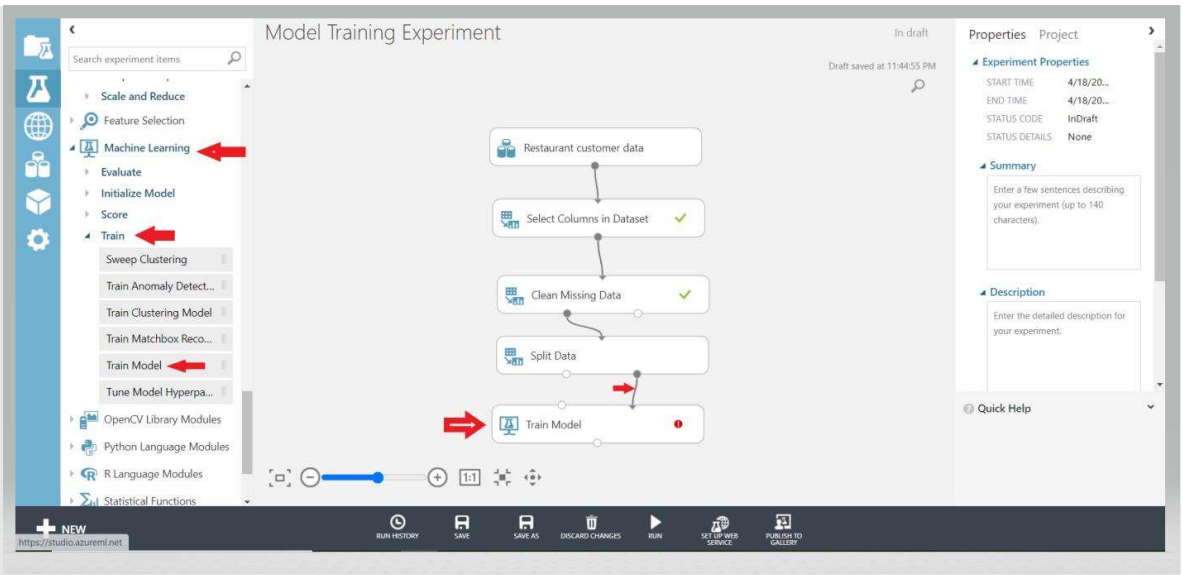


- Select Data Transformation → Sample and Split→ Split data and drag it into Panel. Select the value 0.75 to split the data.

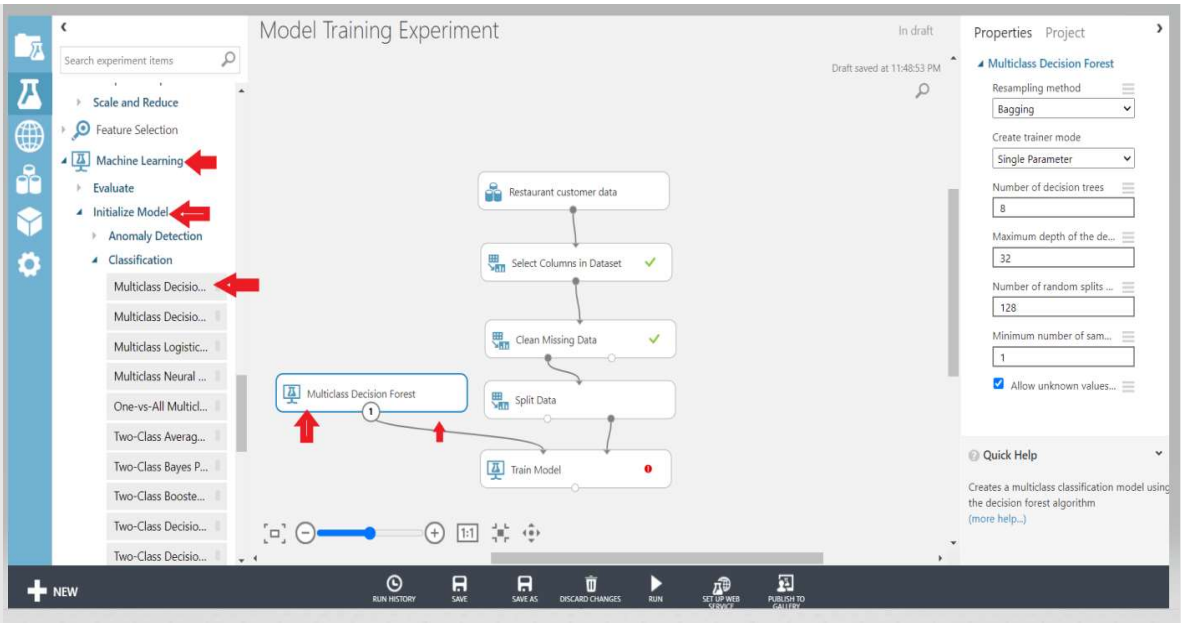
© Edunet Foundation. All rights reserved.



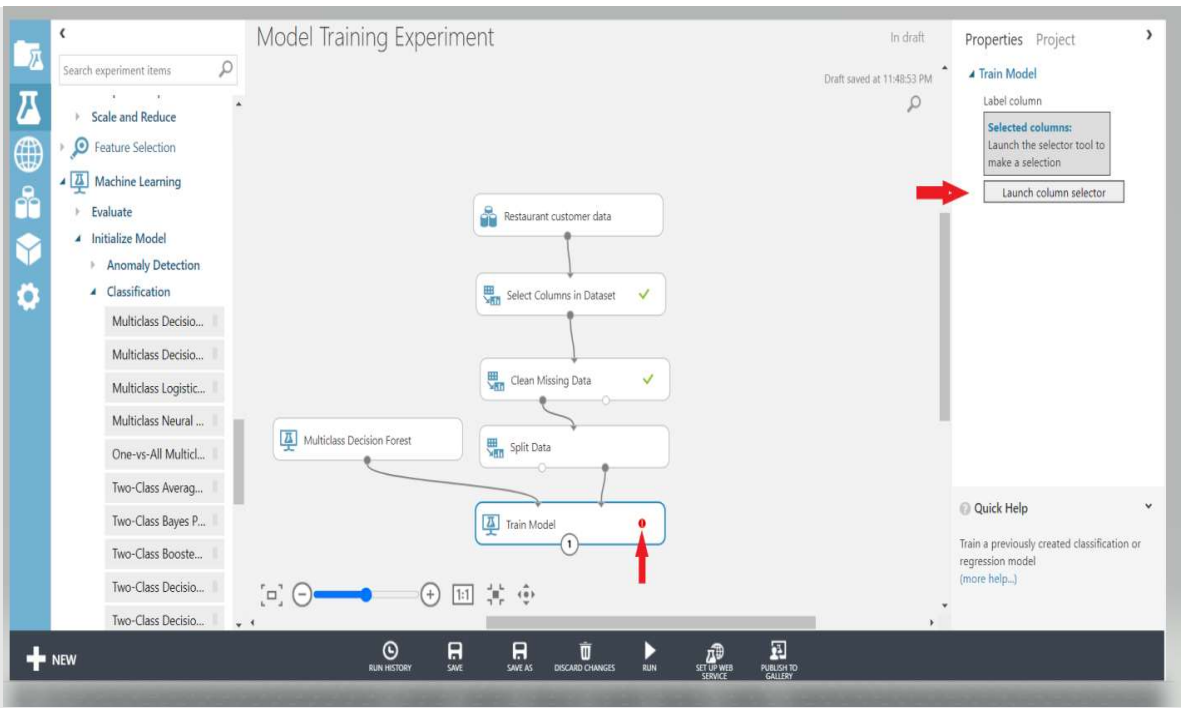
- Select Machine Learning → Train → Train Model and drag it into the Panel and connect it.



- Select the machine learning Algorithm from Machine Learning → Initialize Model→ Classification →Multiclass Decision Forest, drag it into panel and make connection.

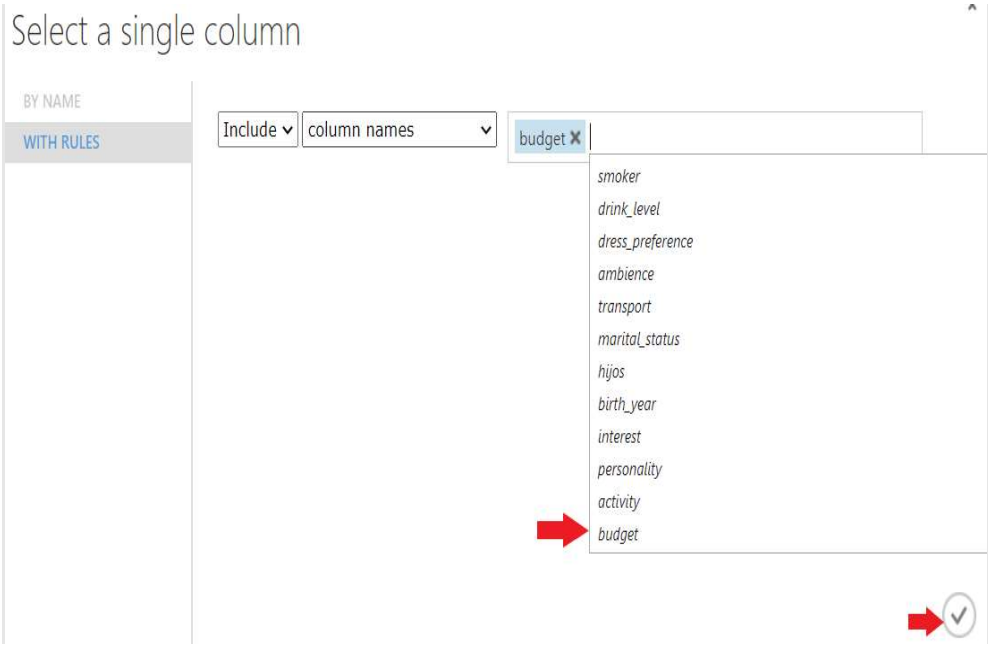


- Click on Train Model and Launch Column Selector

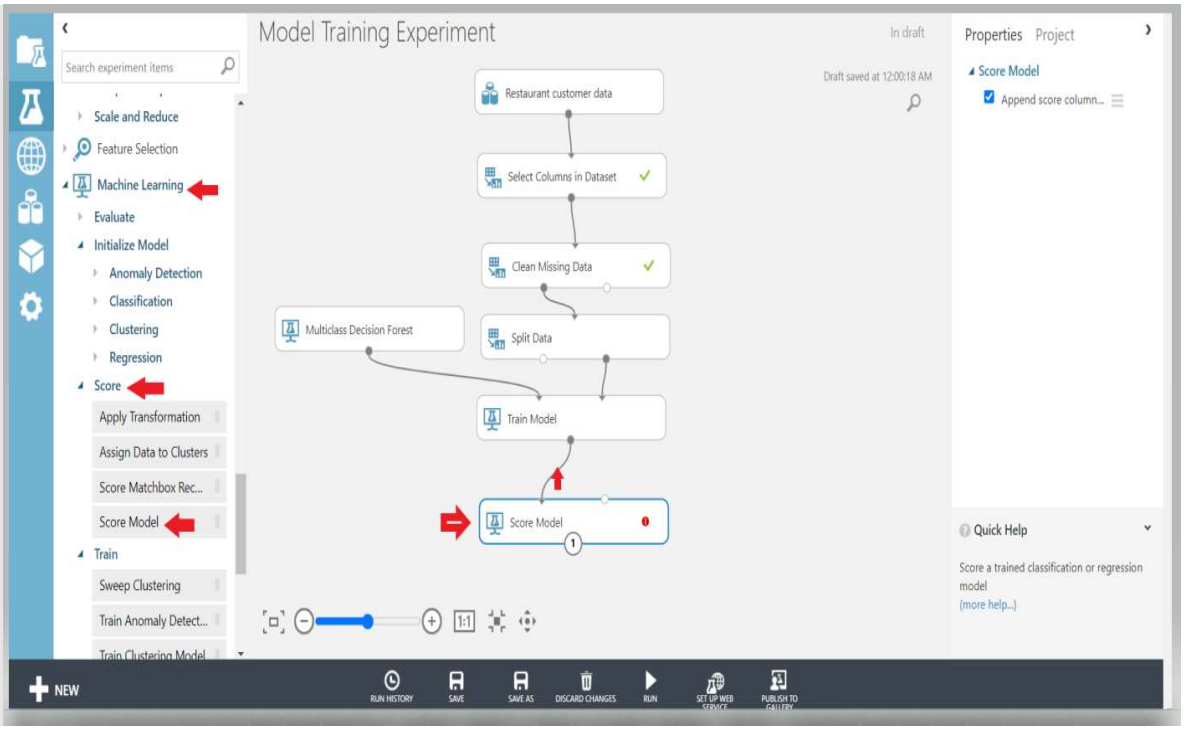


- Now Select Budget as output for the prediction and press the tick mark.

© Edunet Foundation. All rights reserved.

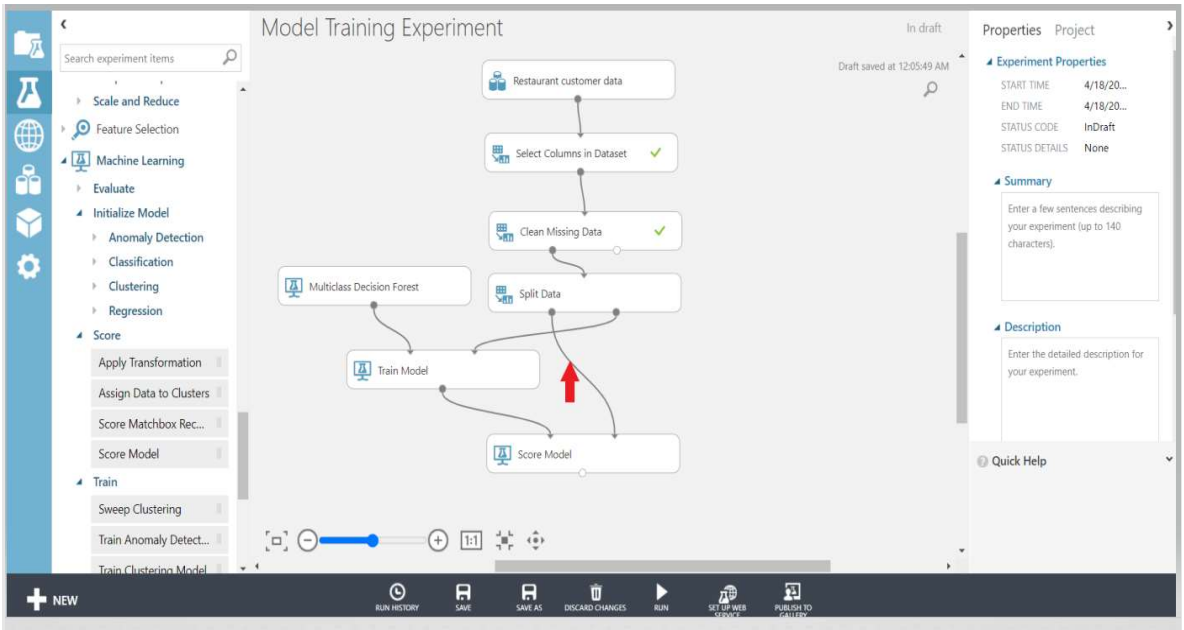


- For Score calculation, Choose Machine Learning → Score → Score Model and drag it into panel and Connect it.

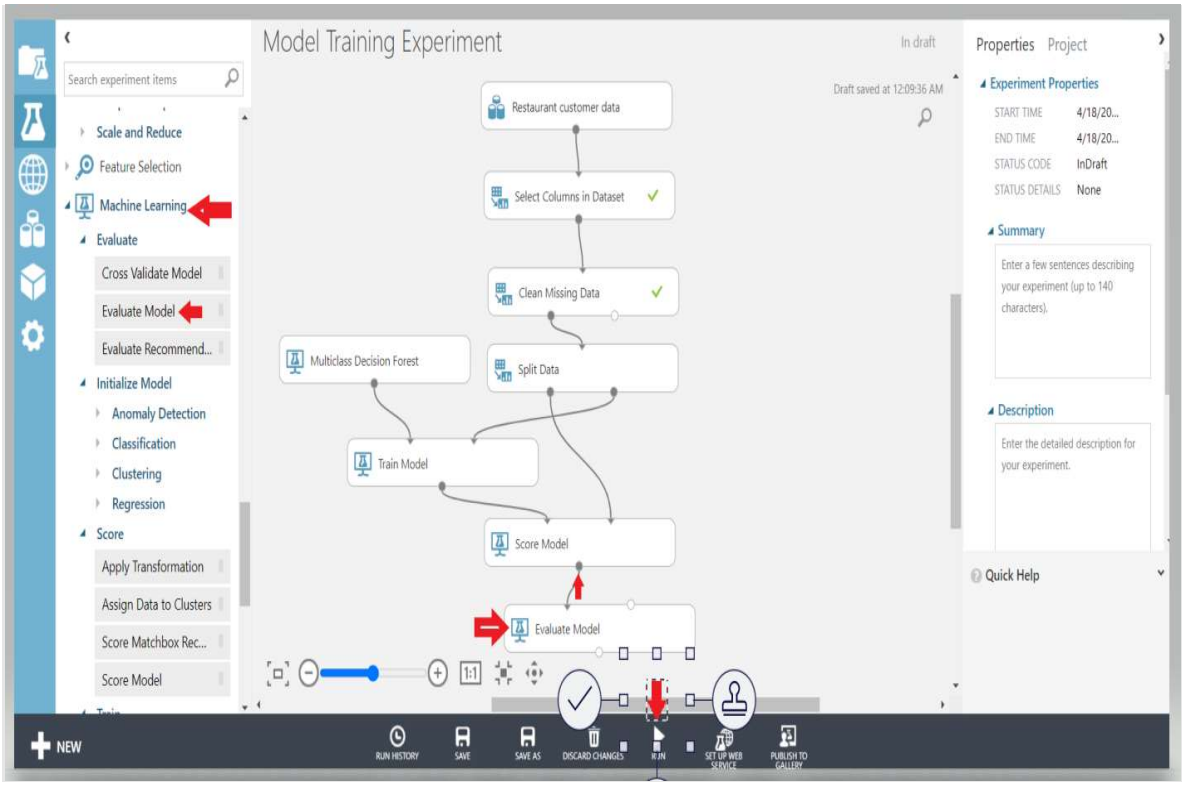


- Connect Split Data with Score Model for Testing.

© Edunet Foundation. All rights reserved.

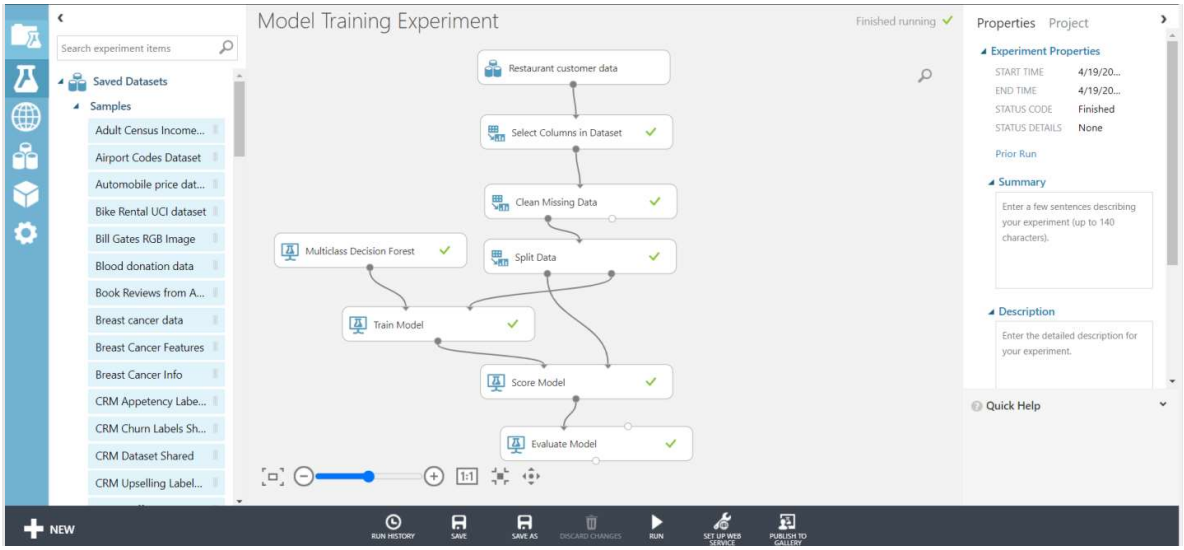


- For Evaluation of the model choose, Machine Learning → Evaluate → Evaluate Model, drag it into panel and connect.

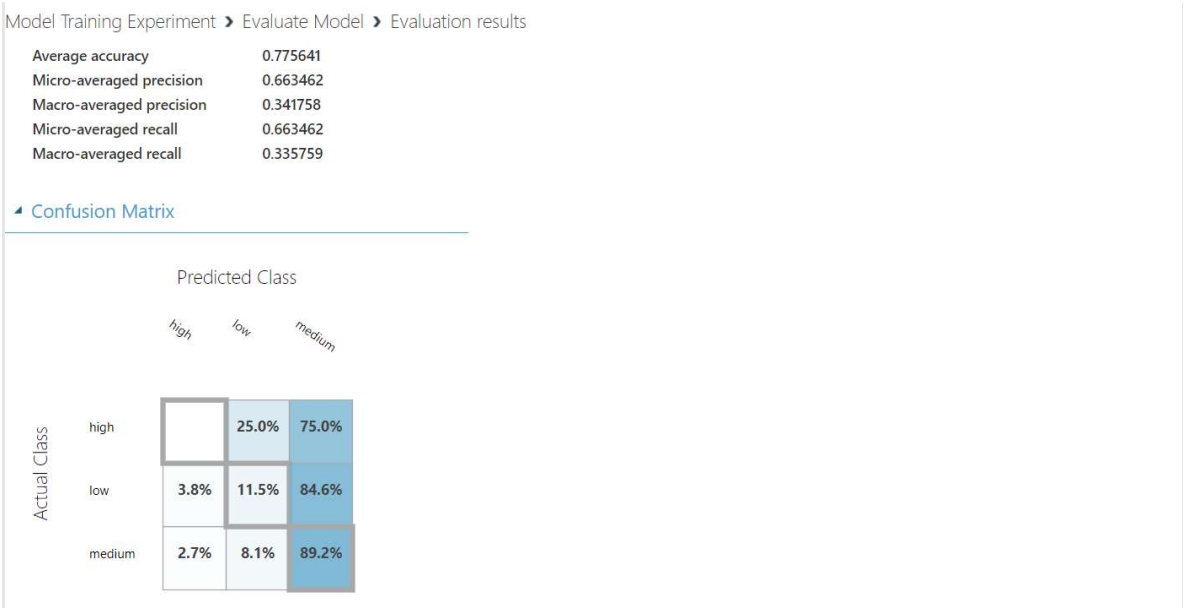


- Now Run the model. Green tick shows that model has run successfully.

© Edunet Foundation. All rights reserved.

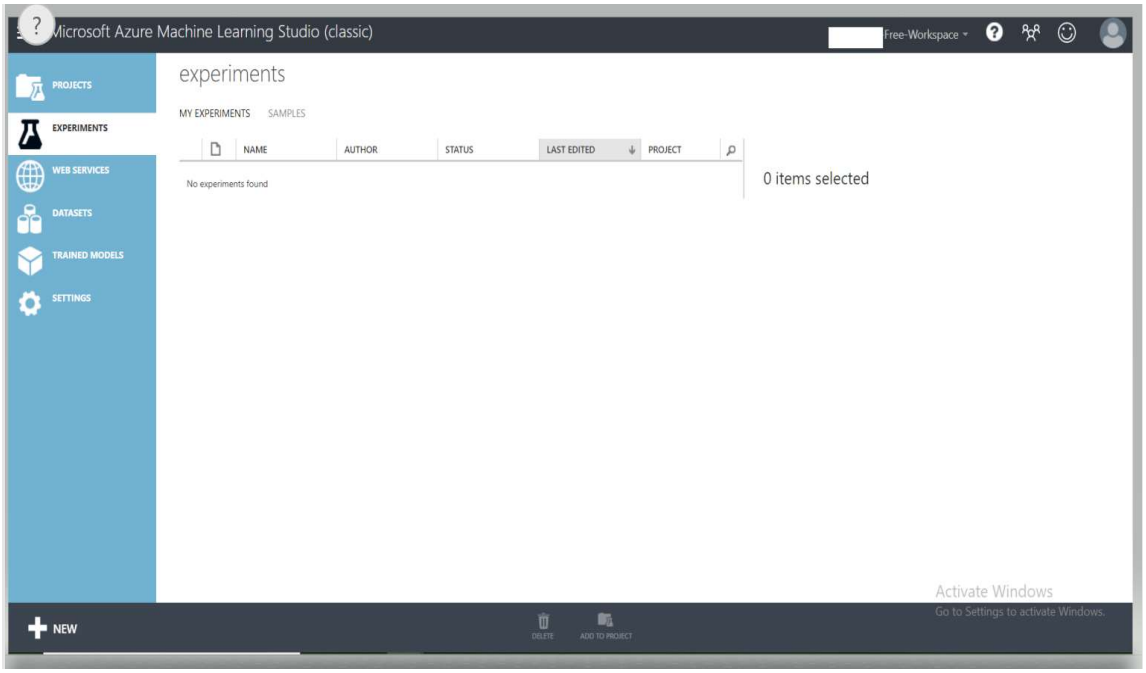


- To see the visualization Result, right click on Evaluate model → Evaluation Result→ Visualize

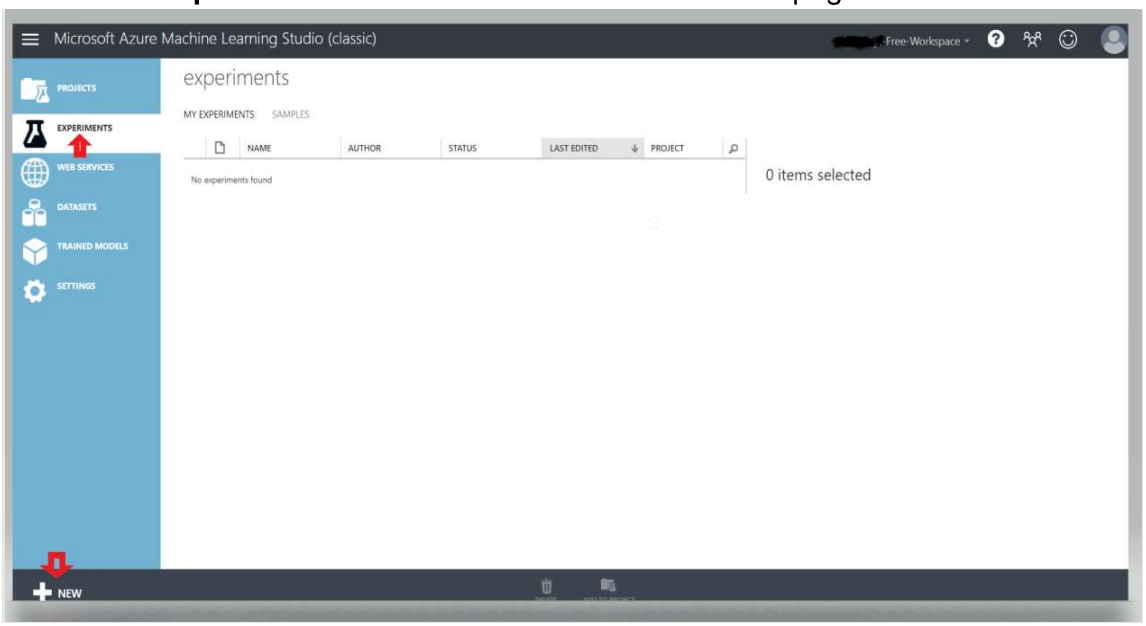


2.17 Create a Regression model with Azure ML Studio

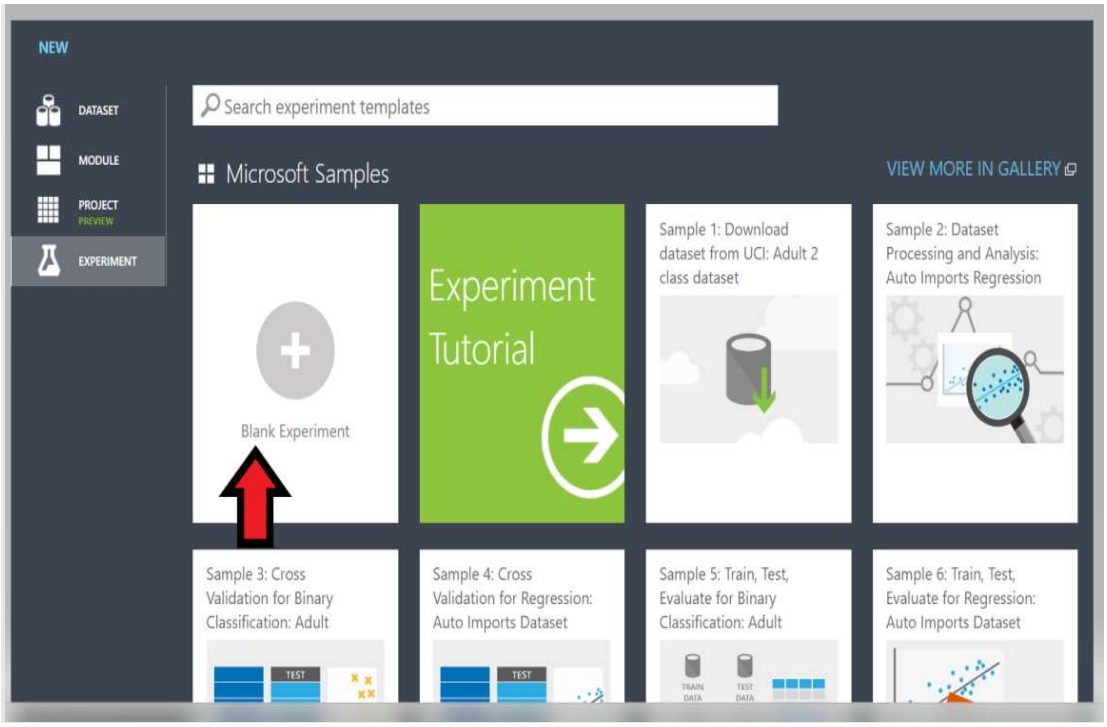
- Sign in into Microsoft Azure Machine Learning Studio (classic) and create workspace.



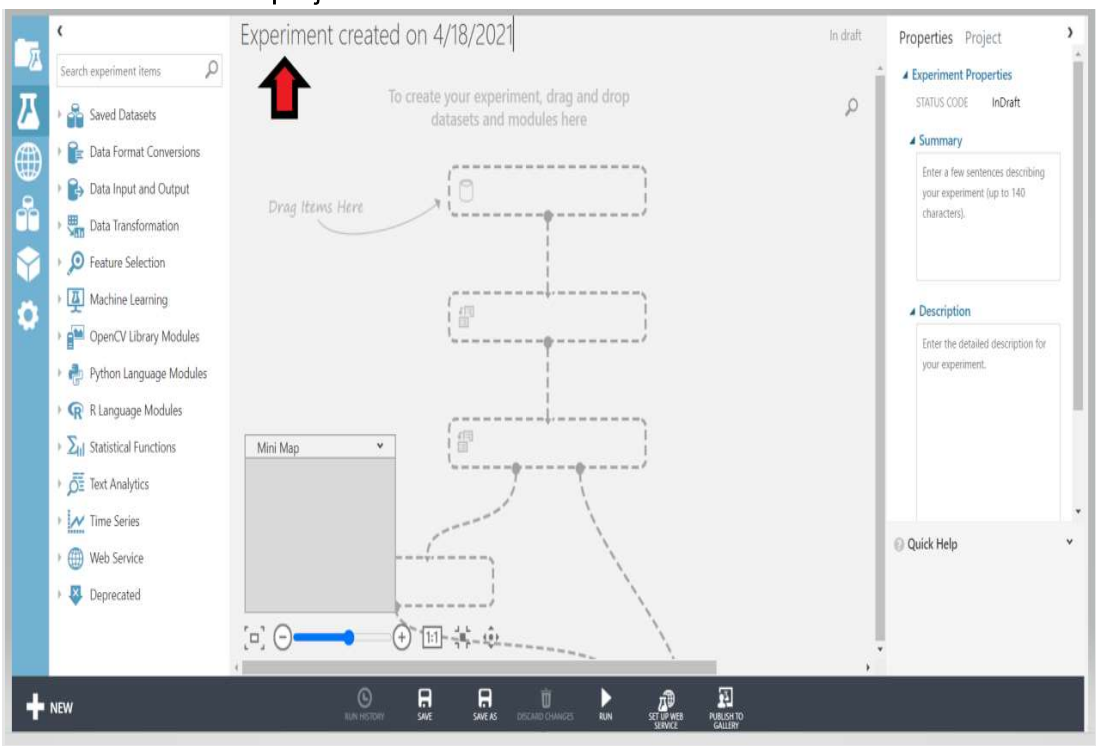
- First select **Experiment** and then **New** at the bottom of the page.



- Select Blank Experiment.

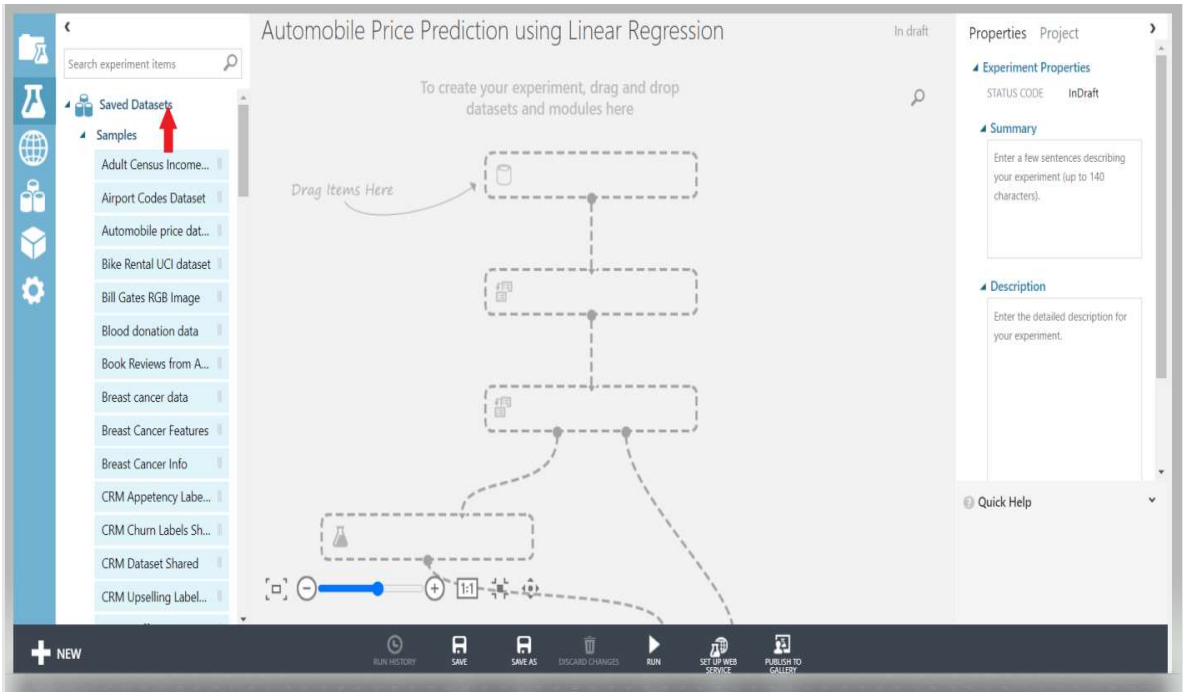


- Give the title for the project.

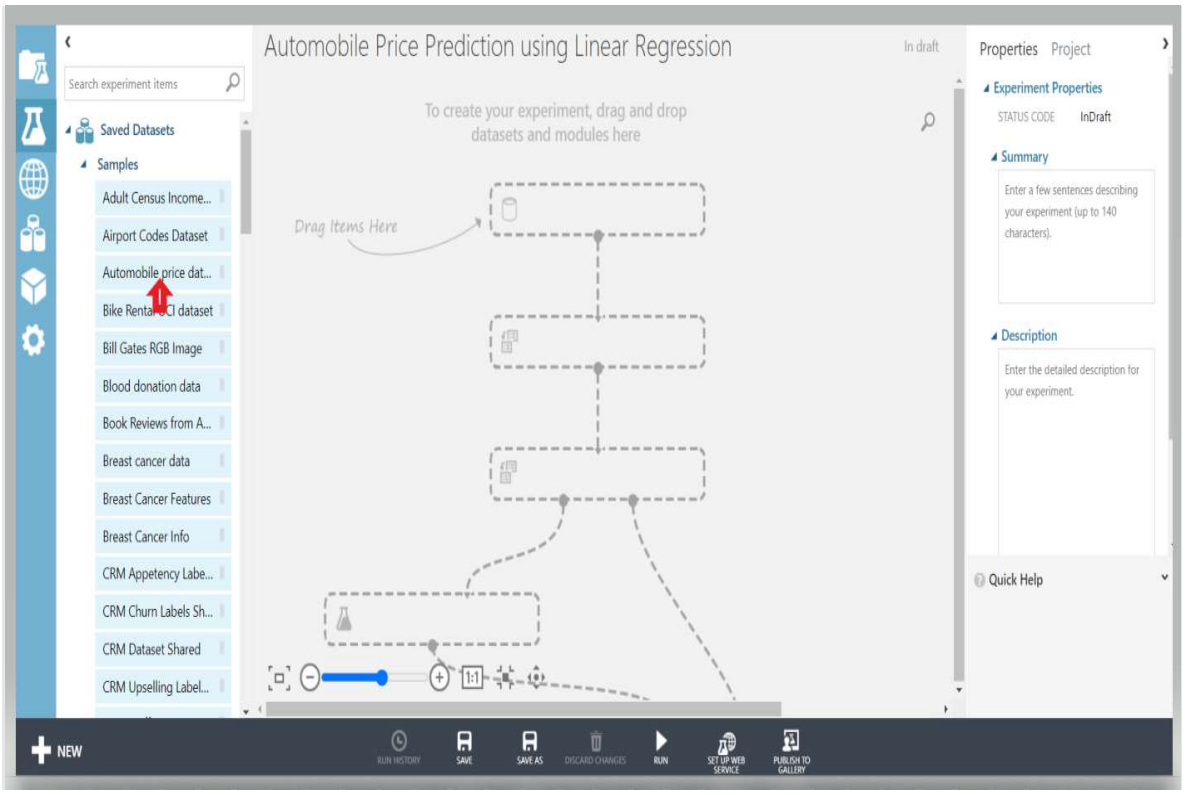


- Select **Sample** option from the **Saved Dataset.**

© Edunet Foundation. All rights reserved.

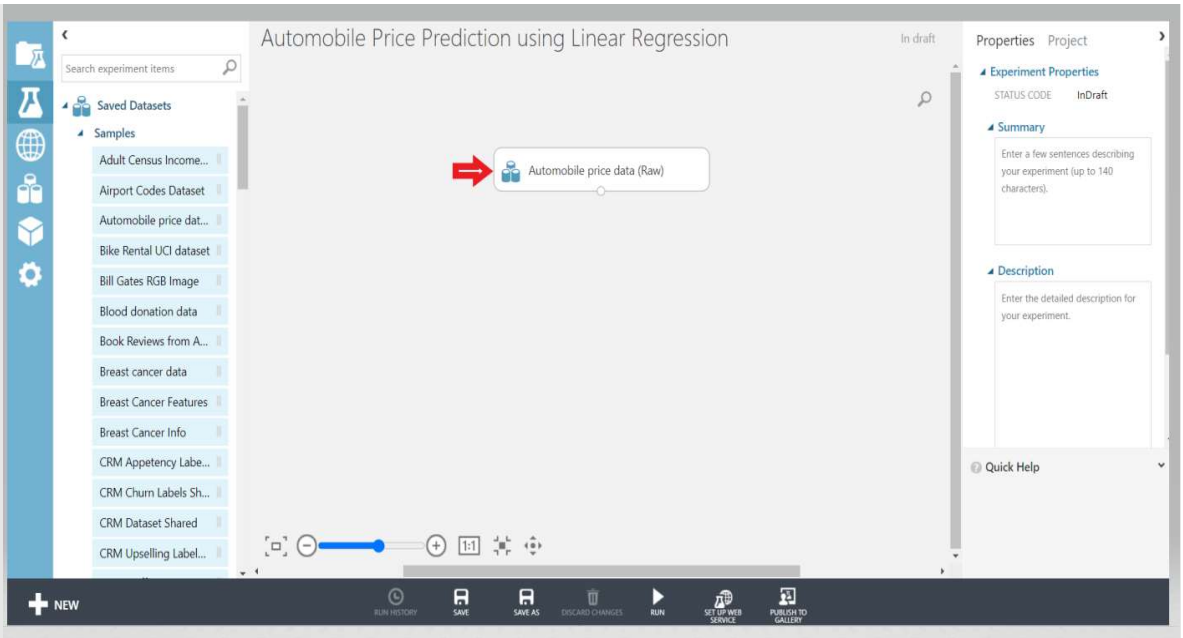


- Select Automobile Price Data Dataset

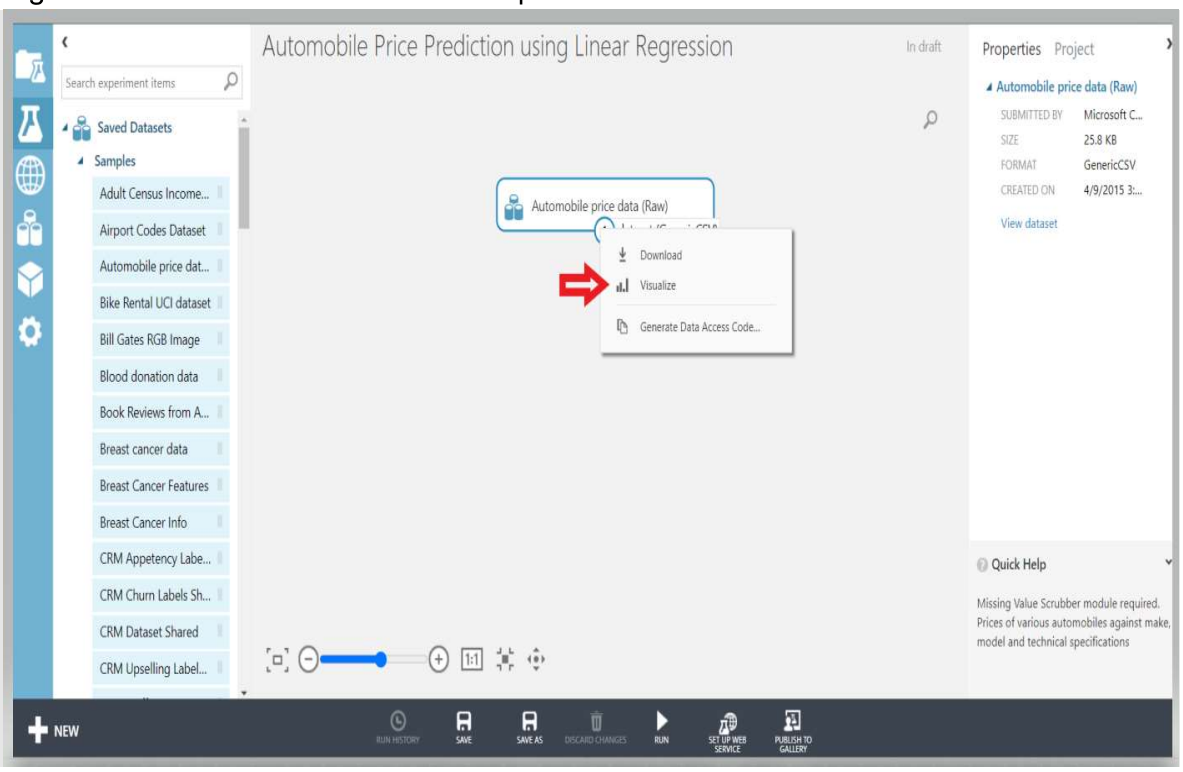


- Drag selected dataset on Panel.

© Edunet Foundation. All rights reserved.

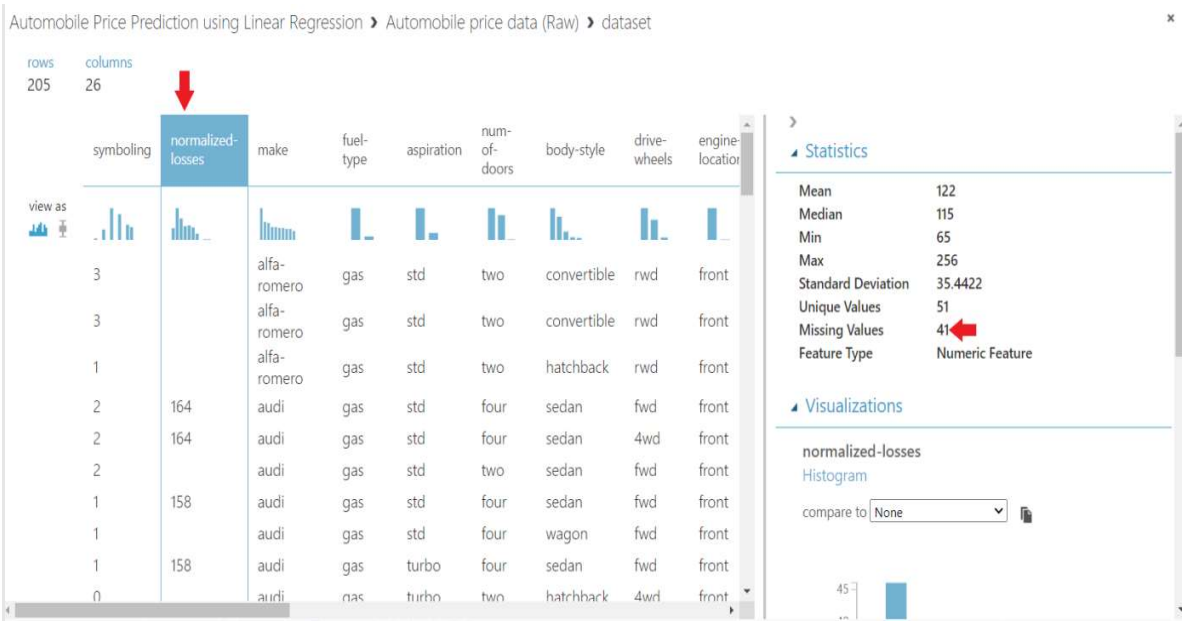


- Right click on 1 and choose visualize option.

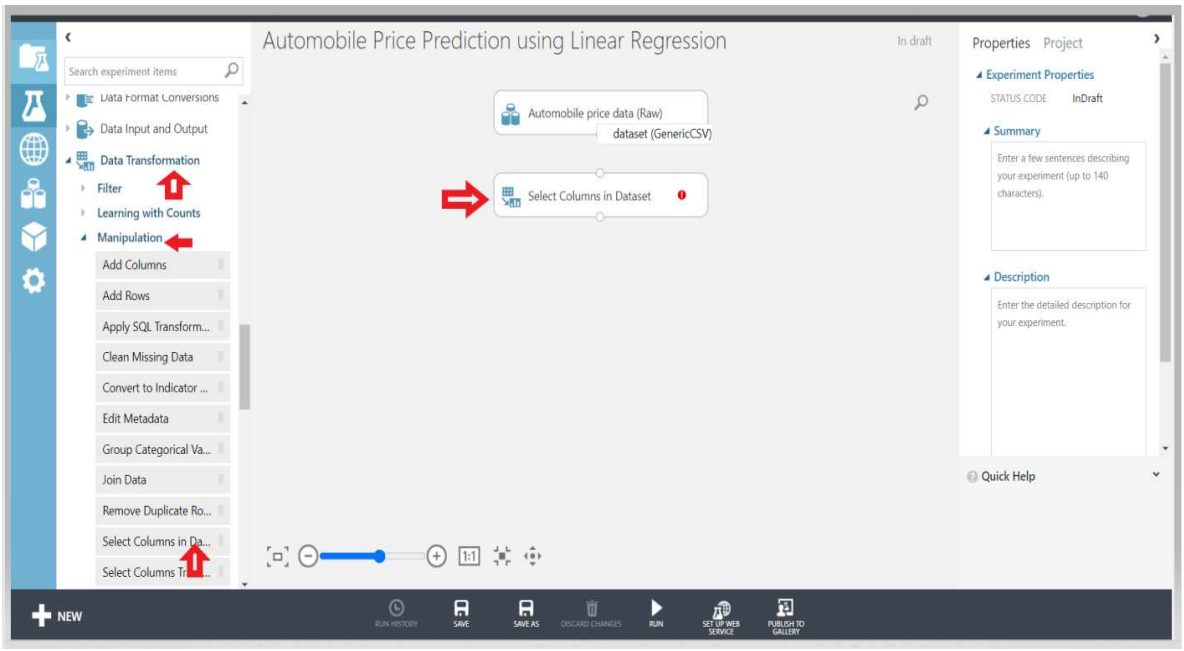


- Visualize the Dataset and then close it.
  For example: Normalized losses has the 41 missing value which is the maximum..

© Edunet Foundation. All rights reserved.

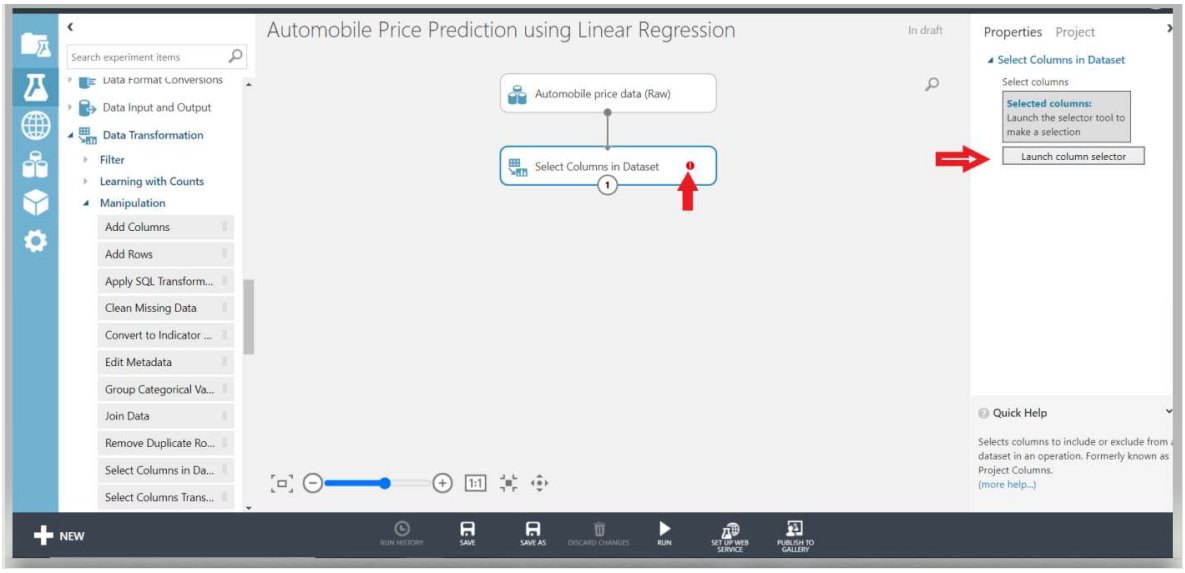


- Select the Data Transformation → Manipulation → Select Columns in Dataset and drag it into Panel.

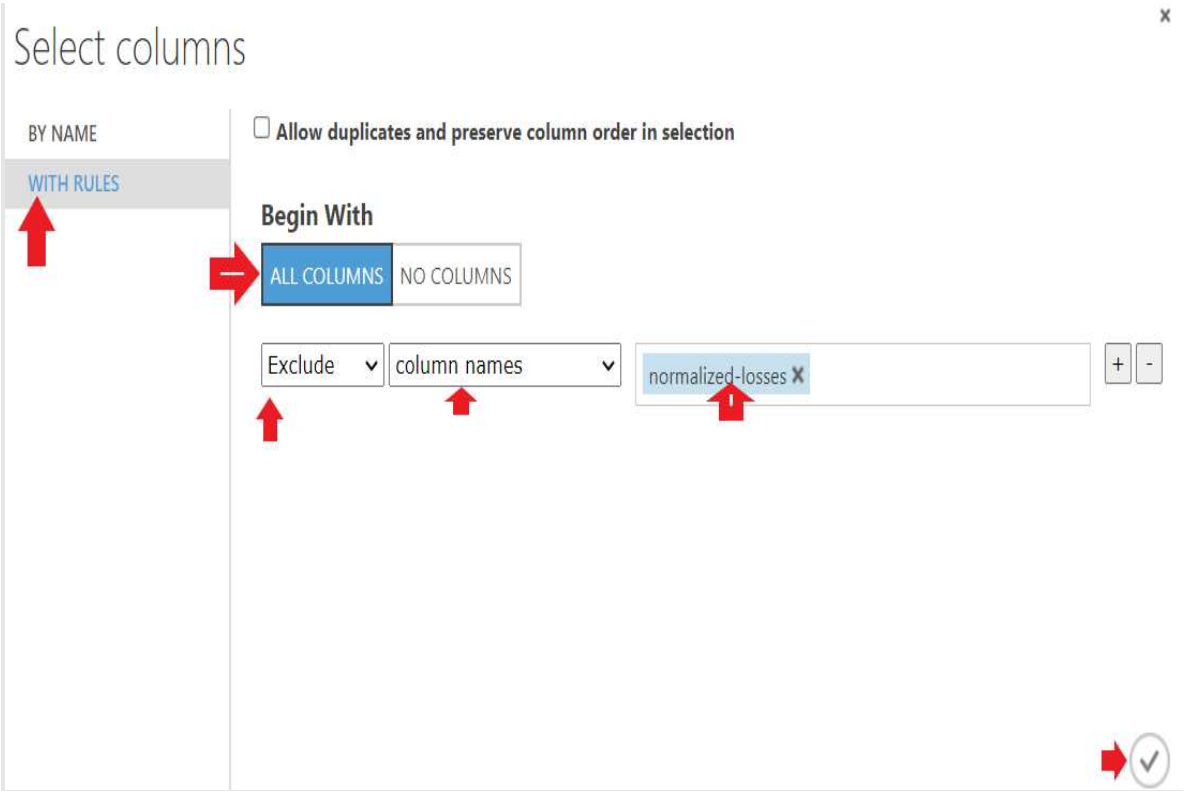


- Make connection between dragged item, press on red sign and then Launch column selector to choose the relevant column.

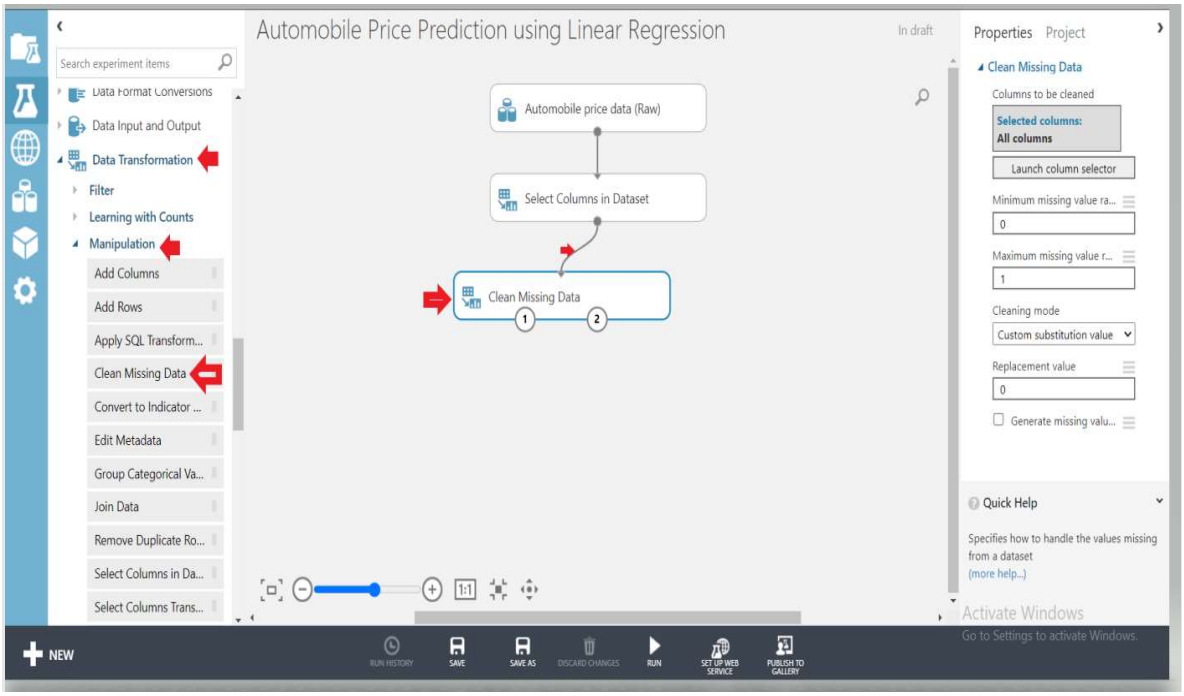
© Edunet Foundation. All rights reserved.



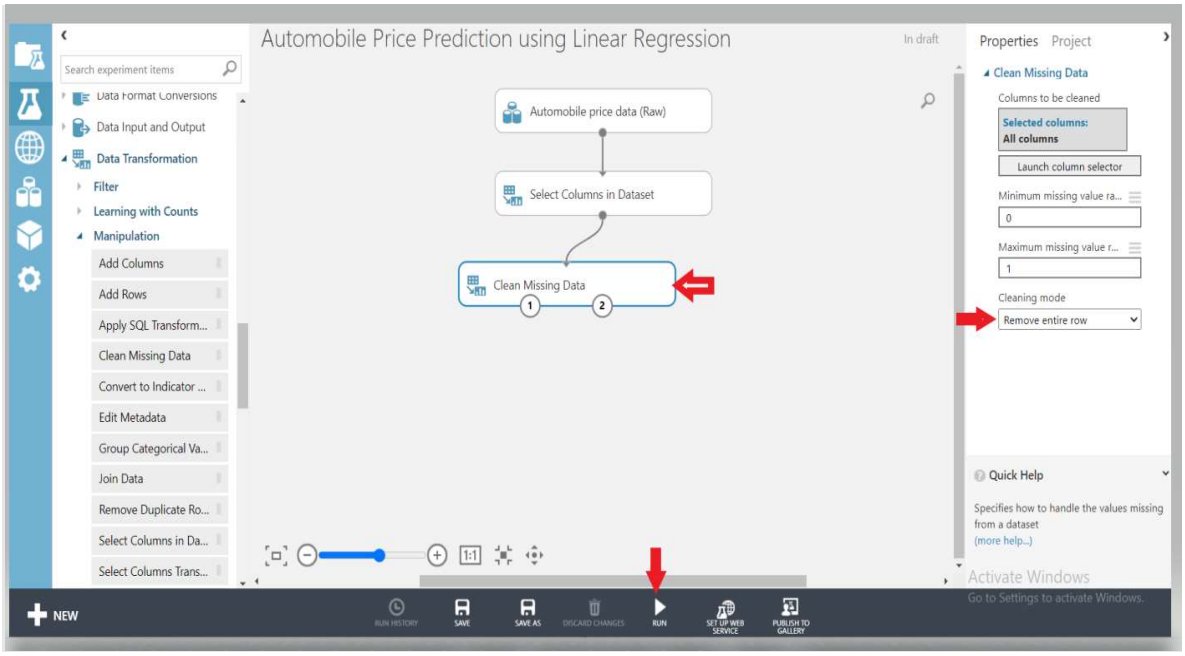
- Remove the column which have the missing value by selecting With Rules → All Columns → Exclude → column names →normalized –losses and click on tick.



- Select the Data Transformation → Manipulation → Clean missing data, drag it into Panel and make connection.

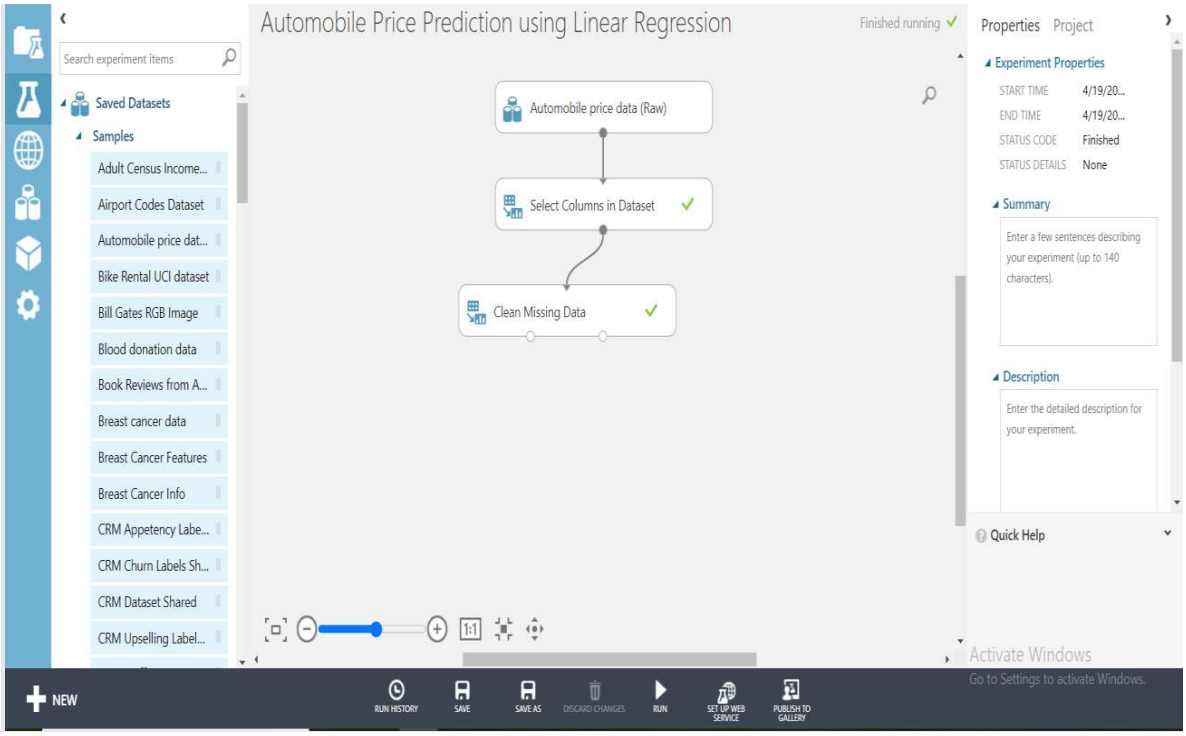


- Click on Clean missing data, set Remove entire row in cleaning mode and press Run.

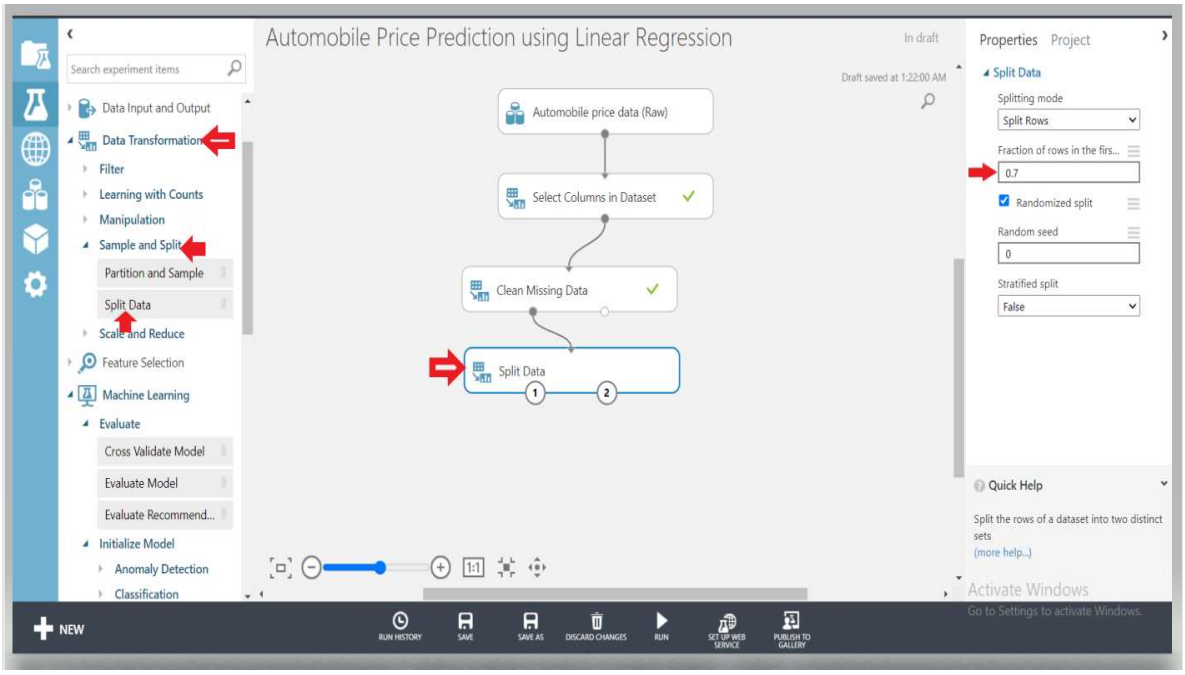


- If there is green tick, it means no error.

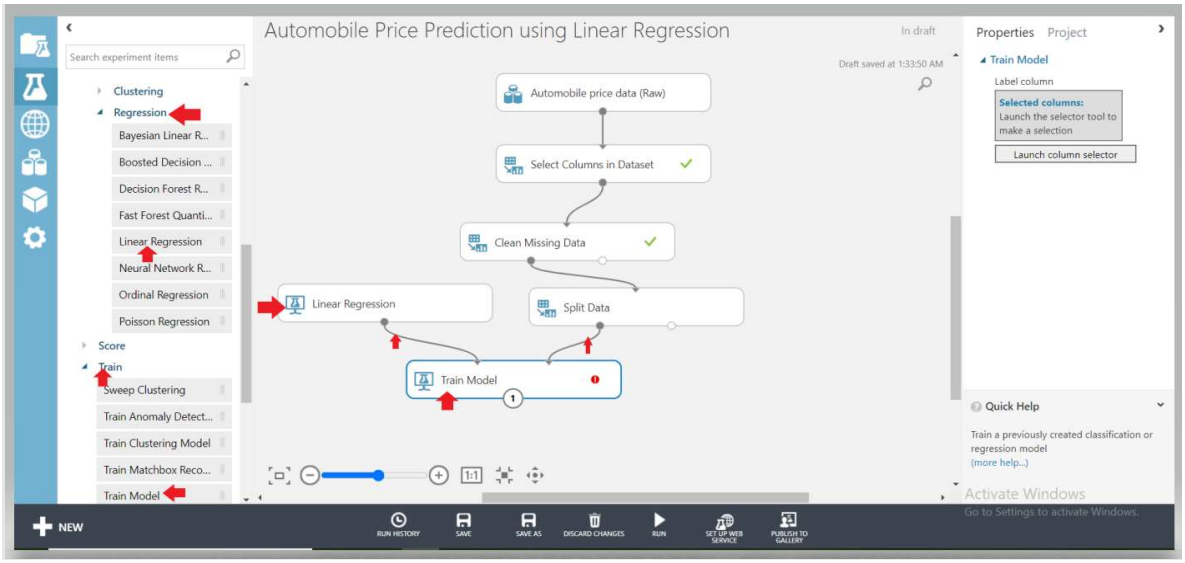
© Edunet Foundation. All rights reserved.



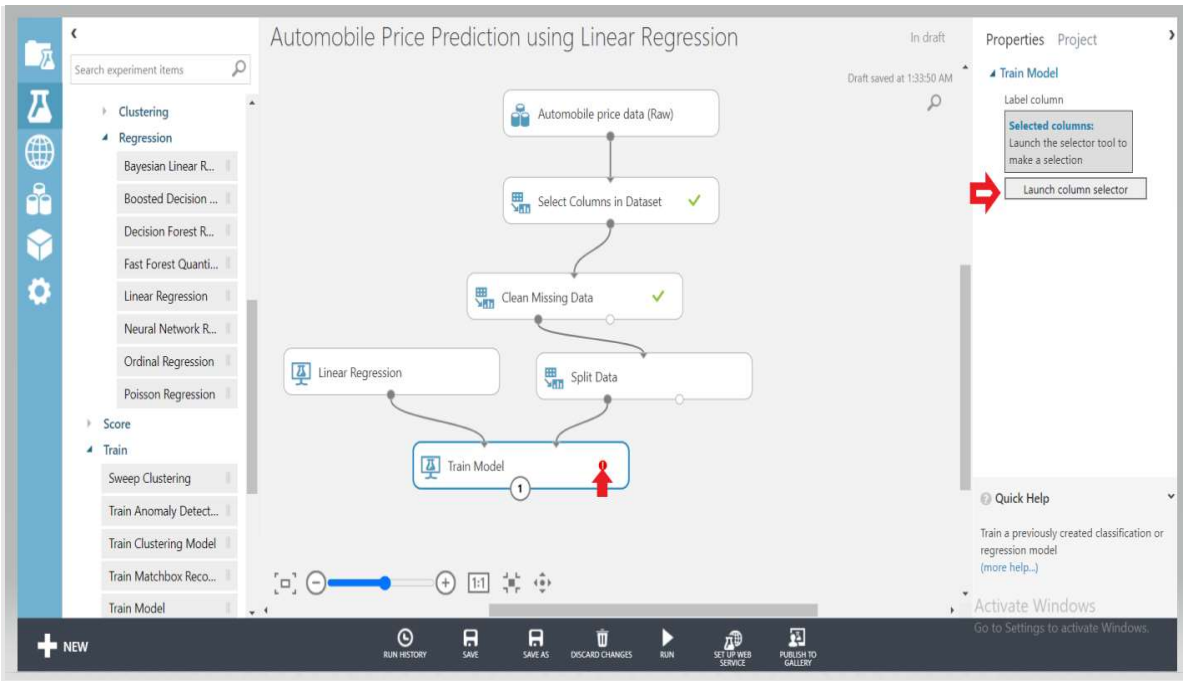
- Select Data Transformation → Sample and Split→ Split data and drag it into Panel. Select the value 0.75 to split the data.



- Select Machine Learning → Train → Train Model and drag it into the Panel and connect it.
  Select the machine learning Algorithm from Machine Learning → Initialize Model →Regression → Linear Regression, drag it into panel and make connection.

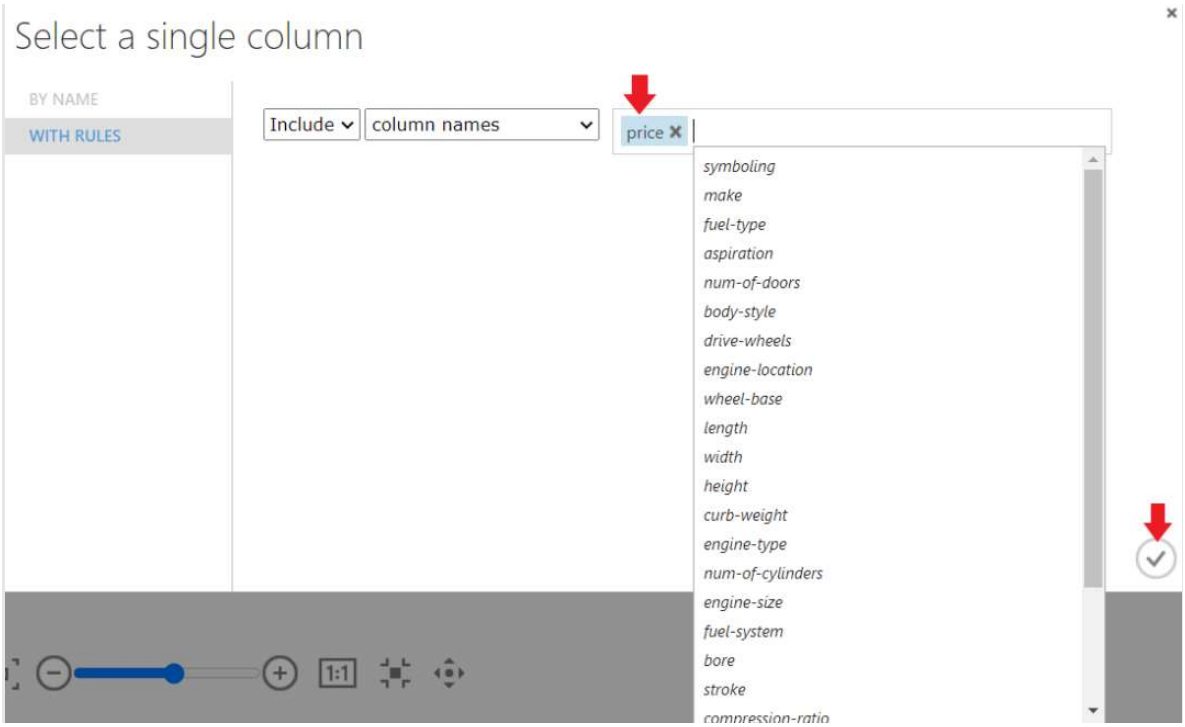


- Click on Train Model and Launch Column Selector

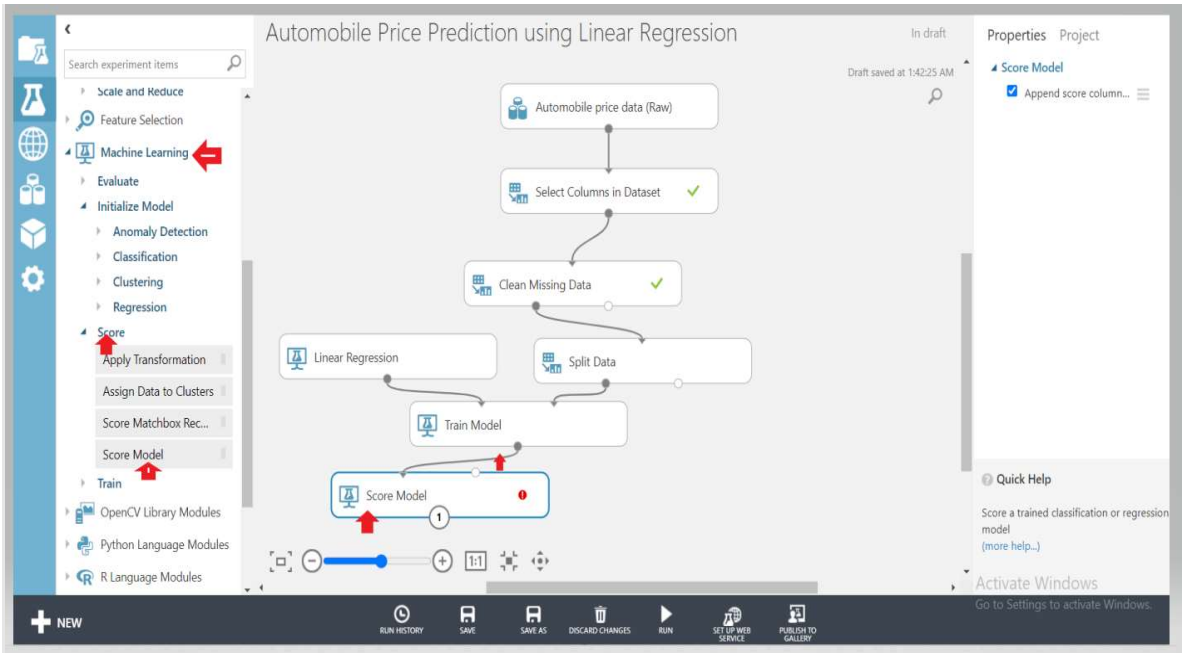


© Edunet Foundation. All rights reserved.

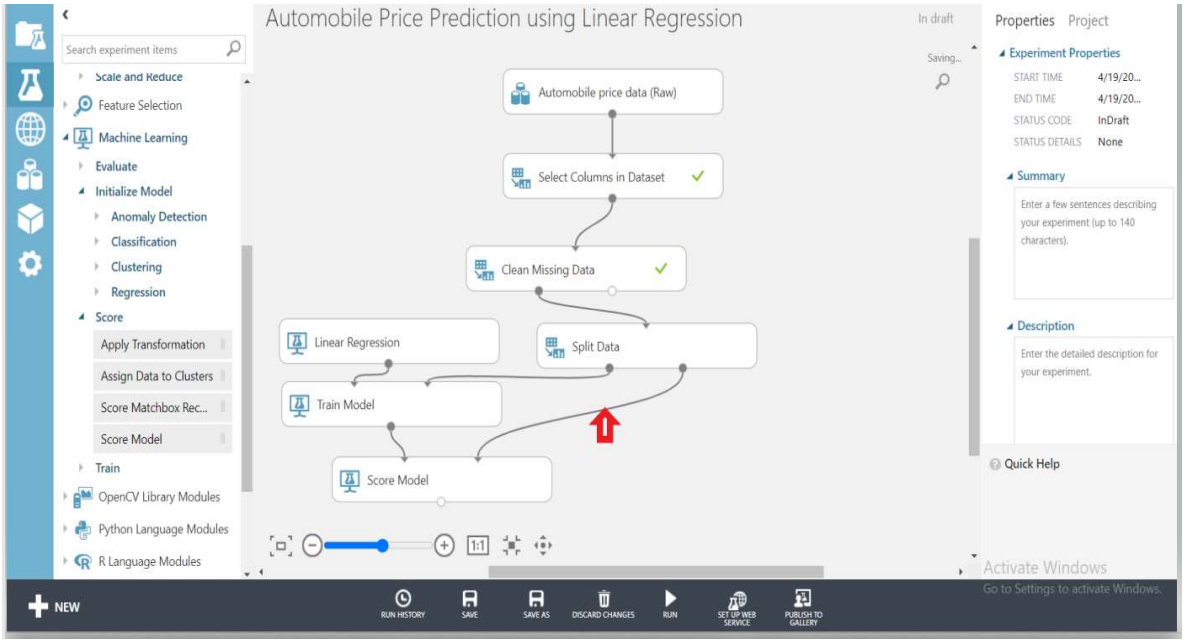
- Now Select price as output for the prediction and press the tick mark.



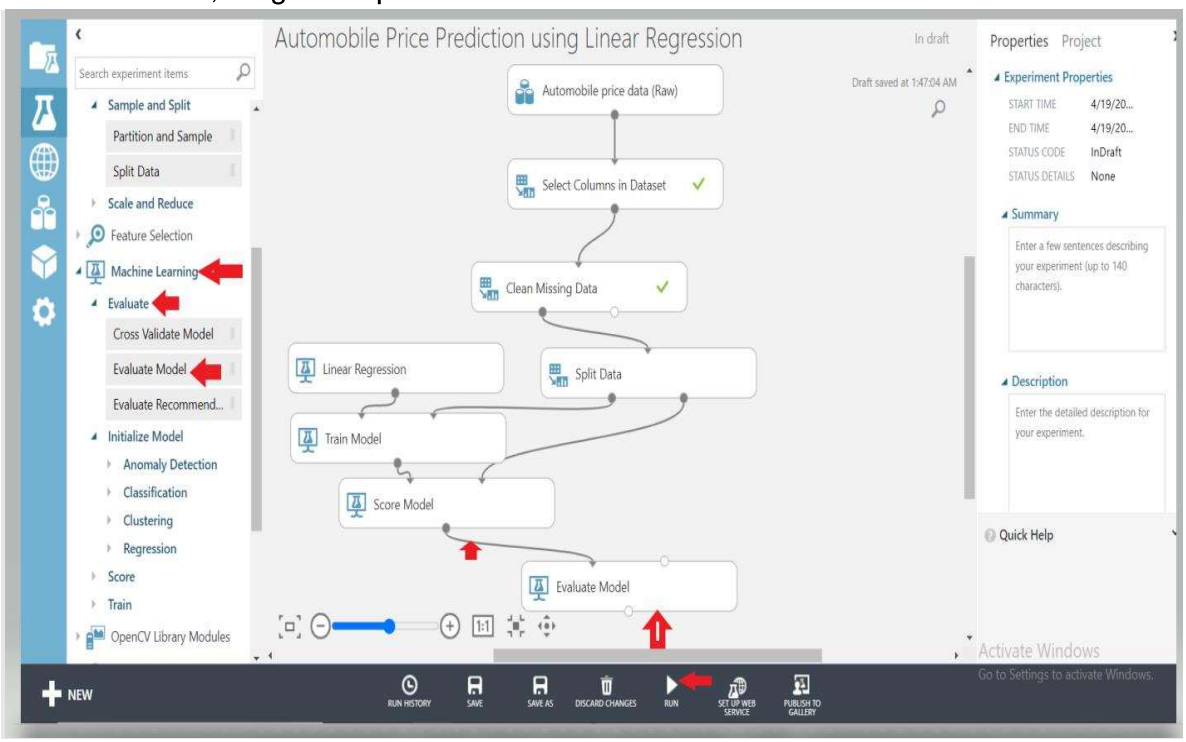
- For Score calculation, Choose Machine Learning → Score → Score Model and drag it into panel and connect it.



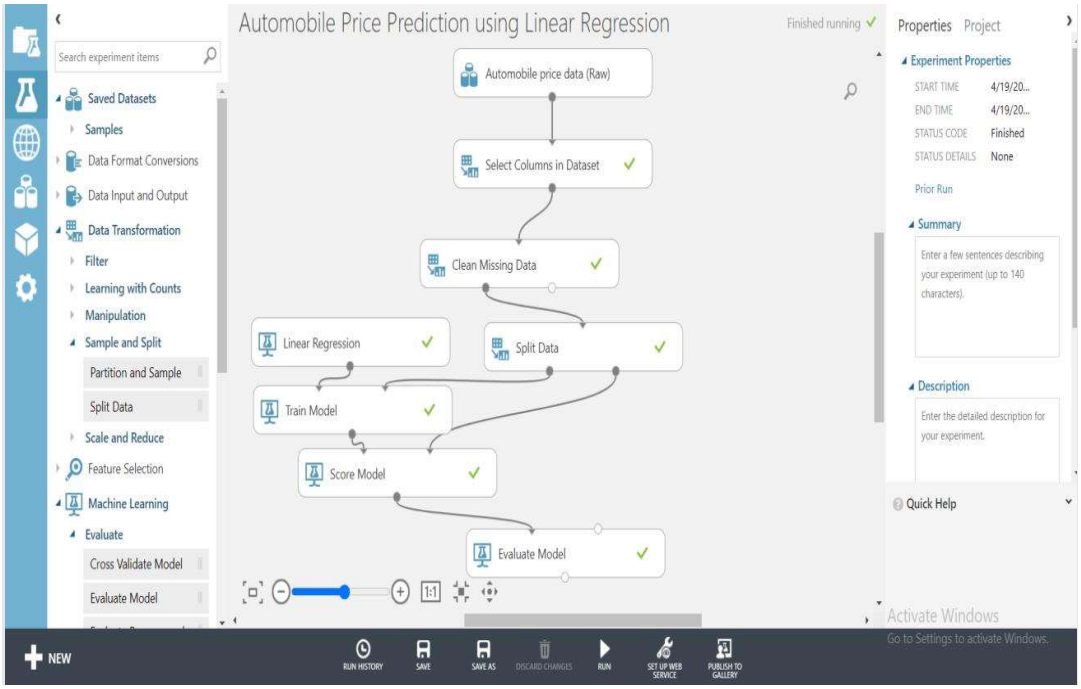
- Connect Split Data with Score Model for Testing.



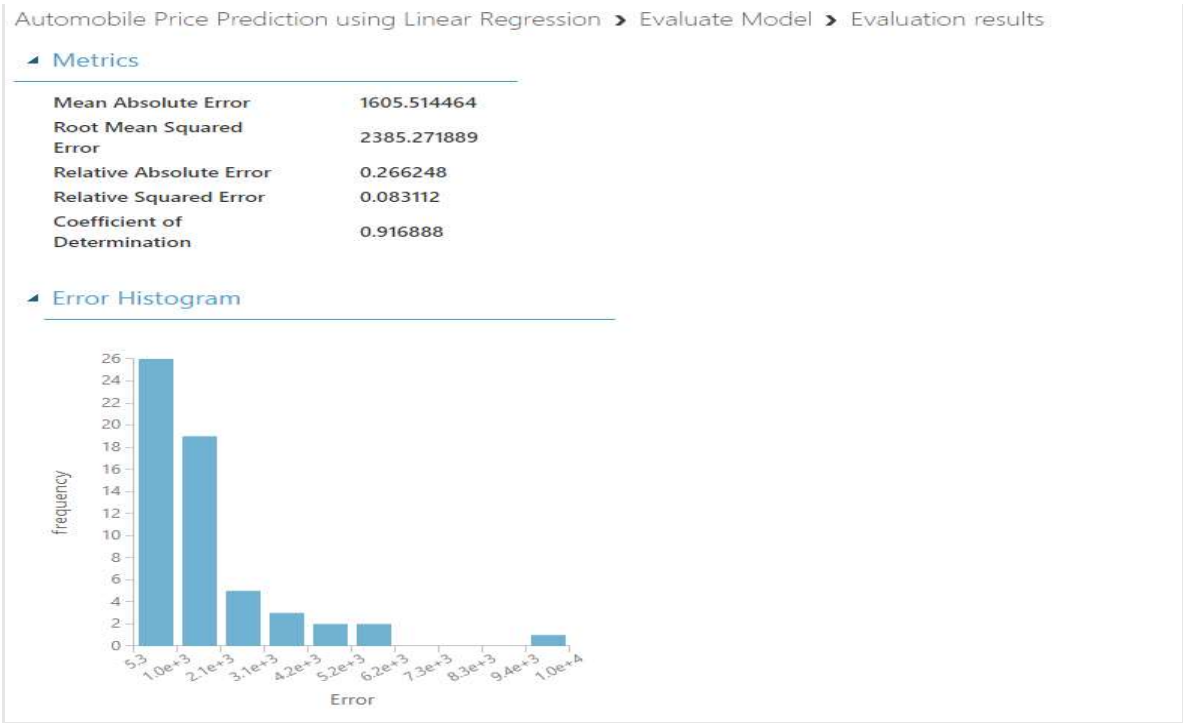
- For Evaluation of the model choose, Machine Learning → Evaluate → Evaluate Model, drag it into panel and connect. Run the model.



- Green tick shows that model has run successfully. Entire model would be display as below:



- To see the visualization Result, right click on Evaluate model → Evaluation Result→ Visualize



We just explored the Microsoft Azure machine learning studio which is GUI based integrated development environment for constructing and operationalizing machine learning workflow and we also constructed linear

regression model. Now let us get started with logistic regression algorithm which is mainly used for classification.

Do it Yourself Activity

Create Multi Linear Regression Model

To calculate an individual's home loan approval limit, we not only need his age but also his credit rating and other features are also considered such as age, credit rating, number of children's, etc. The data set named 'loan-approval-dataset' is given to you.

Implement linear regression with multiple features (also known as multiple linear regressions) to predict loan approval limit.

Self-paced Courses – https://docs.microsoft.com/en-us/learn/modules/understand-regression-machine-learning/

Chapter 3 : Text Analytics and Building Machine Learning Models

Learning Outcomes:

- Understand the basics of machine learning , its types and applications
- Create Machine Learning model using various Algorithms
- Demonstrate working of ML model without coding
- Able to differentiate between Supervised and Unsupervised Learning
- Able to identify and apply specific ML algorithm to solve real life problems

3.1 Probability Theory

Probability is the measure of the likelihood that an event will occur in a Random Experiment. Probability is quantified as a number between 0 and 1, where, loosely speaking, 0 indicates impossibility and 1 indicates certainty. The higher the probability of an event, the more likely it is that the event will occur.

Example:-

© Edunet Foundation. All rights reserved.

A simple example is the tossing of a fair (unbiased) coin. Since the coin is fair, the two outcomes ("heads" and "tails") are both equally probable; the probability of "heads" equals the probability of "tails"; and since no other outcomes are possible, the probability of either "heads" or "tails" is 1/2 (which could also be written as 0.5 or 50%).

3.2 Naïve Bayes Classifier Algorithm

- Naïve Bayes algorithm is a supervised learning algorithm, which is based on Bayes theorem and used for solving classification problems.
- It is mainly used in text classification that includes a high-dimensional training dataset.
- Naïve Bayes Classifier is one of the simple and most effective Classification algorithms which helps in building the fast machine learning models that can make quick predictions.
- It is a probabilistic classifier, which means it predicts on the basis of the probability of an object.
- Some popular examples of Naïve Bayes Algorithm are spam filtration, Sentimental analysis, and classifying articles.

Why is it called Naïve Bayes?

The Naïve Bayes algorithm is comprised of two words Naïve and Bayes, which can be described as:

- Naïve: It is called Naïve because it assumes that the occurrence of a certain feature is independent of the occurrence of other features. Such as if the fruit is identified on the bases of color, shape, and taste, then red, spherical, and sweet fruit is recognized as an apple. Hence each feature individually contributes to identify that it is an apple without depending on each other.
- Bayes: It is called Bayes because it depends on the principle of Bayes' Theorem.

Bayes' Theorem

Bayes' theorem is also known as Bayes' Rule or Bayes' law, which is used to determine the probability of a hypothesis with prior knowledge. It depends on the conditional probability.

The formula for Bayes' theorem is given as:

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

where,

© Edunet Foundation. All rights reserved.

- P(A|B) is Posterior probability: Probability of hypothesis A on the observed event B.
- P(B|A) is Likelihood probability: Probability of the evidence given that the probability of a hypothesis is true.
- P(A) is Prior Probability: Probability of hypothesis before observing the evidence.
- P(B) is Marginal Probability: Probability of Evidence.

Advantages of Naïve Bayes Classifier

- Naïve Bayes is one of the fast and easy ML algorithms to predict a class of datasets.
- It can be used for Binary as well as Multi-class Classifications.
- It performs well in Multi-class predictions as compared to the other Algorithms.
- It is the most popular choice for text classification problems.

Disadvantages of Naïve Bayes Classifier

- Naive Bayes assumes that all features are independent or unrelated, so it cannot learn the relationship between features.

Applications of Naïve Bayes Classifier

- It is used for Credit Scoring.
- It is used in medical data classification.
- It can be used in real-time predictions because Naïve Bayes Classifier is an eager learner.
- It is used in Text classification such as Spam filtering and Sentiment analysis.

Types of Naïve Bayes Model

There are three types of Naive Bayes Model, which are given below:

1. Gaussian: The Gaussian model assumes that features follow a normal distribution. This means if predictors take continuous values instead of discrete, then the model assumes that these values are sampled from the Gaussian distribution.
2. Multinomial: The Multinomial Naïve Bayes classifier is used when the data is multinomial distributed. It is primarily used for document classification problems, it means a particular document belongs to which category such as Sports, Politics, education, etc.
3. Bernoulli: The Bernoulli classifier works similar to the Multinomial classifier, but the predictor variables are the independent Booleans variables. Such as if a particular word is present or not in a document. This model is also famous for document classification tasks.

© Edunet Foundation. All rights reserved.

Project : Naïve Bayes using Sklearn

Let us consider a dataset about sports. We have 2 features telling us about weather and humidity. The target what we need to find is whether a person will play or not . Let us use GaussianNB to predict the output.

Link for the project : https://github.com/Edunet-Foundation/Tech-Saksham/blob/main/Tech%20Saksham%20AI%20Practical%20-%20Notebooks/Chapter%204/7.Naive_Bayes.py

Project: Text Analytics using Pandas

Text Analytics is the process of drawing meaning out of written communication. Let us see how to perform Text Analytics using Pandas library

https://github.com/Edunet-Foundation/Tech-Saksham/blob/main/Tech%20Saksham%20AI%20Practical%20-%20Notebooks/Chapter%204/8.Text_Analytics_with_Pandas.ipynb

3.3 Bag of Words Approach

The bag-of-words model is a way of representing text data when modeling text with machine learning algorithms. The bag-of-words model is simple to understand and implement and has seen great success in problems such as language modeling and document classification.

A bag-of-words model, or BoW for short, is a way of extracting features from text for use in modeling, such as with machine learning algorithms. The approach is very simple and flexible, and can be used in a myriad of ways for extracting features from documents. A bag-of-words is a representation of text that describes the occurrence of words within a document. It involves two things:

- A vocabulary of known words.
- A measure of the presence of known words.

It is called a "*bag*" of words, because any information about the order or structure of words in the document is discarded. The model is only concerned with whether known words occur in the document, not where in the document.

Here's a sample of reviews about a particular horror movie.

© Edunet Foundation. All rights reserved.

- Review 1: This movie is very scary and long
- Review 2: This movie is not scary and is slow
- Review 3: This movie is spooky and good

We can see that there are some contrasting reviews about the movie as well as the length and pace of the movie. Imagine looking at a thousand reviews like these. Clearly, there is a lot of interesting insights we can draw from them and build upon them to gauge how well the movie performed.

However, as we saw above, we cannot simply give these sentences to a machine learning model and ask it to tell us whether a review was positive or negative. We need to perform certain text preprocessing steps.

We will first build a vocabulary from all the unique words in the above three reviews. The vocabulary consists of these 11 words: 'This', 'movie', 'is', 'very', 'scary', 'and', 'long', 'not', 'slow', 'spooky', 'good'. We can now take each of these words and mark their occurrence in the three movie reviews above with 1s and 0s. This will give us 3 vectors for 3 reviews:

	1 This	2 movie	3 is	4 very	5 scary	6 and	7 long	8 not	9 slow	10 spooky	11 good	Length of the review(in words)
Review 1	1	1	1	1	1	1	1	0	0	0	0	7
Review 2	1	1	2	0	0	1	1	0	1	0	0	8
Review 3	1	1	1	0	0	0	1	0	0	1	1	6

Image: Bag of Words Approach
Reference: https://cdn.analyticsvidhya.com/wp-content/uploads/2020/02/BoWBag-of-Words-model-2.png

Vector of Review 1: [1 1 1 1 1 1 1 0 0 0 0]

Vector of Review 2: [1 1 2 0 0 1 1 0 1 0 0]

Vector of Review 3: [1 1 1 0 0 0 1 0 0 1 1]

And that's the core idea behind a Bag of Words (BoW) model.

Project on Spam Ham Using using BOW concept:

Spam Ham classification is one of the classical application of Machine Learning. There are two types of data present in this repository, which is **ham** (non-spam) and **spam** data. Furthermore, in the ham data, there are easy and hard, which means

© Edunet Foundation. All rights reserved.

there is some non-spam data that has a very high similarity with spam data. This might pose a difficulty for our system to make a decision. Let us use the concept of Bag of Words to convert Text to Numbers and then apply Classification algorithm on it.

Link for the project:

https://github.com/Edunet-Foundation/Tech-Saksham/blob/main/Tech%20Saksham%20AI%20Practical%20-%20Notebooks/Chapter%204/10.%20Spam_Ham_Demonstration.ipynb

3.4 Lexicons for Sentiment Analysis

Sentiment analysis is a process by which information is analyzed through the use of natural language processing (NLP). Lexicons calculate the sentiment from the semantic orientation of word or phrases that occur in a text.

The types of lexicons are:

- Affin
- Textblob
- VADER

Afinn

It is the simplest yet popular lexicons used for sentiment analysis developed by Finn Årup Nielsen. It contains 3300+ words with a polarity score associated with each word. In python, there is an in-built function for this lexicon. [10.6]

Textblob

It is a simple python library that offers API access to different NLP tasks such as sentiment analysis, spelling correction, etc. Textblob sentiment analyzer returns two properties for a given input sentence:

- Polarity is a float that lies between [-1,1], -1 indicates negative sentiment and +1 indicates positive sentiments.

- Subjectivity is also a float which lies in the range of [0,1]. Subjective sentences generally refer to personal opinion, emotion, or judgment.

VADER sentiment

Valence aware dictionary for sentiment reasoning (VADER) is another popular rule-based sentiment analyzer. It uses a list of lexical features (e.g. word) which are labeled as positive or negative according to their semantic orientation to calculate the text sentiment. Vader sentiment returns the probability of a given input sentence to be Positive, negative, and neutral.

© Edunet Foundation. All rights reserved.

For example:

"The food was great!"
Positive: 99%
Negative :1%
Neutral: 0%

These three probabilities will add up to 100%.

Project : Sentiment Analysis using Lexicons

We will use all the 3 lexicons mentioned in the following project

https://github.com/Edunet-Foundation/Tech-Saksham/blob/main/Tech%20Saksham%20AI%20Practical%20-%20Notebooks/Chapter%204/11.Sentiment%20Analysis%20using%20Affin%20Lexicon.ipynb

https://github.com/Edunet-Foundation/Tech-Saksham/blob/main/Tech%20Saksham%20AI%20Practical%20-%20Notebooks/Chapter%204/12.Sentiment%20Analysis%20using%20Textblob%20Lexicon.ipynb

https://github.com/Edunet-Foundation/Tech-Saksham/blob/main/Tech%20Saksham%20AI%20Practical%20-%20Notebooks/Chapter%204/13.Sentiment%20Analysis%20using%20Vader%20Lexicon.ipynb

Do it Yourself Activity

Create a Logistic Regression model on Azure ML Studio

The dataset named 'diabetes' is given to you. Build logistic regression model to predict whether patient is diabetic or not in azure Machine learning studio. Also perform required data preprocessing activities on dataset, if required.
https://docs.microsoft.com/en-us/learn/modules/intro-to-azure-machine-learning-service/
https://docs.microsoft.com/en-us/learn/modules/create-classification-model-azure-machine-learning-designer/

So, in this chapter we explored the various algorithms which falls under the category of supervised machine learning. But there are some drawbacks or limitations on these algorithms. So, in the next chapter we are going to discuss the subset of machine learning named deep learning and how it helps to overcome the limitations of machine learning. So let us get started with deep learning.

© Edunet Foundation. All rights reserved.

Chapter 4: Building Deep Learning Models

Learning Outcomes:

- Understand basic key concepts of Deep Learning
- Learn about Image Processing
- Understand the process of automatic feature extraction using Artificial Neural Networks

4.1 Deep Learning Basics

Deep learning (also known as deep structured learning) is a part of a broader family of machine learning methods based on artificial neural networks with representation learning. Learning can be supervised, semi-supervised or unsupervised. Deep learning models are not meant to be trained with an algorithm. Instead, they make learning a step further. Deep learning models works directly with audio, images and video data to get real time analysis. The data being fed to the deep learning model does not need any external intervention. You can feed raw data to the model and receive actionable insights.

Deep-learning architectures such as deep neural networks, deep belief networks, recurrent neural networks and convolutional neural networks have been applied to fields including computer vision, machine vision, speech recognition, natural language processing, audio recognition, social network filtering, machine translation, bioinformatics, drug design, medical image analysis, material inspection and board game programs, where they have produced results comparable to and in some cases surpassing human expert performance.

Artificial neural networks (ANNs) were inspired by information processing and distributed communication nodes in biological systems. ANNs have various differences from biological brains. Specifically, neural networks tend to be static and symbolic, while the biological brain of most living organisms is dynamic (plastic) and analogue.

The adjective "deep" in deep learning refers to the use of multiple layers in the network. Early work showed that a linear perceptron cannot be a universal classifier, and then that a network with a nonpolynomial activation function with one hidden layer of unbounded width can on the other hand so be. Deep learning is a modern variation which is concerned with an unbounded number of layers of bounded size, which

© Edunet Foundation. All rights reserved.

permits practical application and optimized implementation, while retaining theoretical universality under mild conditions. In deep learning the layers are also permitted to be heterogeneous and to deviate widely from biologically informed connectionist models, for the sake of efficiency, trainability and understandability, whence the "structured" part.

In deep learning, each level learns to transform its input data into a slightly more abstract and composite representation. In an image recognition application, the raw input may be a matrix of pixels; the first representational layer may abstract the pixels and encode edges; the second layer may compose and encode arrangements of edges; the third layer may encode a nose and eyes; and the fourth layer may recognize that the image contains a face. Importantly, a deep learning process can learn which features to optimally place in which level *on its own*. (Of course, this does not completely eliminate the need for hand-tuning; for example, varying numbers of layers and layer sizes can provide different degrees of abstraction.)

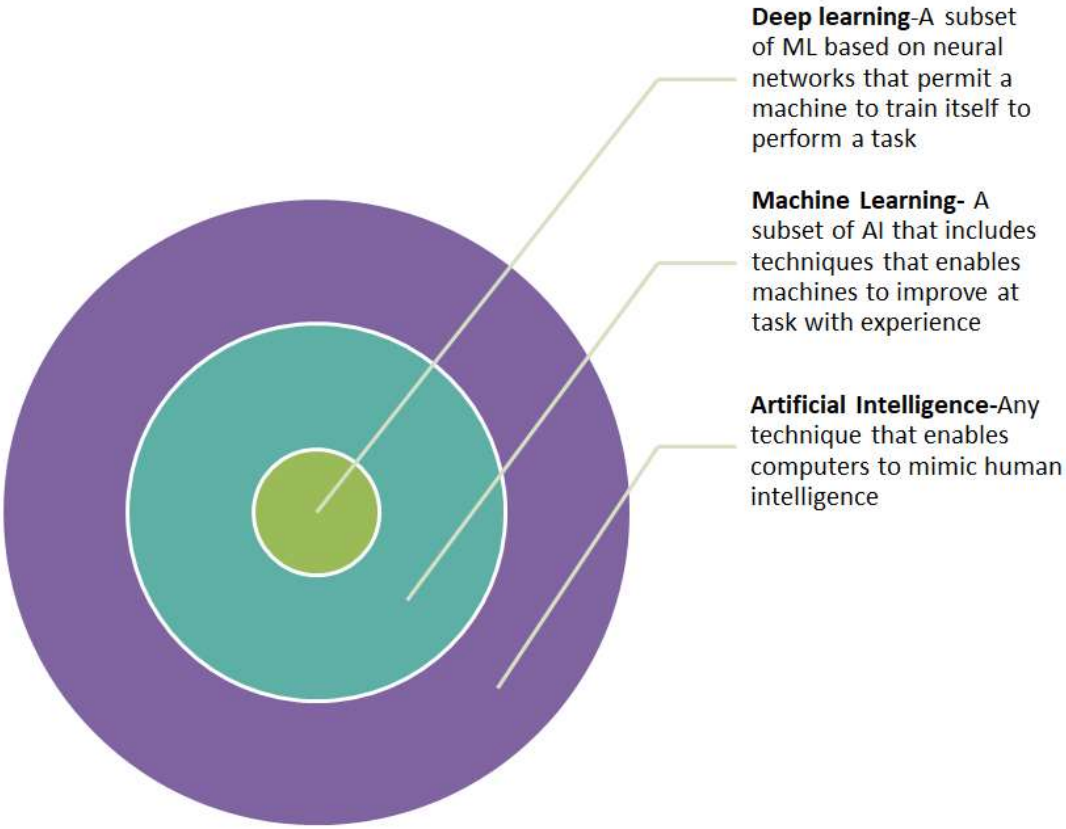


Fig : Relation between Deep Learning , Machine Learning and Artificial Intelligence

© Edunet Foundation. All rights reserved.

Differences between Machine Learning and Deep Learning

Machine Learning	Deep Learning
Outputs: Numerical Value, like classification of score	Anything from numerical values to free-form elements, such as free text and sound.
Uses various types of automated algorithms that turn to model functions and predict future action from data.	Uses neural network that passes data through processing layers to the interpret data features and relations.
Algorithms are detected by data analysts to examine specific variables in data sets.	Algorithms are largely self-depicted on data analysis once they're put into production.

Now let's explore the building blocks of neural networks which are mainly inspired by working of human brain.

4.2 Concept of Neural Networks

Artificial neural networks (ANNs), usually simply called neural networks (NNs), are computing systems vaguely inspired by the biological neural networks that constitute animal brains.

An ANN is based on a collection of connected units or nodes called artificial neurons, which loosely model the neurons in a biological brain. Each connection, like the synapses in a biological brain, can transmit a signal to other neurons. An artificial neuron that receives a signal then processes it and can signal neurons connected to it. The "signal" at a connection is a real number, and the output of each neuron is computed by some non-linear function of the sum of its inputs. The connections are called *edges*. Neurons and edges typically have a *weight* that adjusts as learning proceeds. The weight increases or decreases the strength of the signal at a connection. Neurons may have a threshold such that a signal is sent only if the aggregate signal crosses that threshold. Typically, neurons are aggregated into layers. Different layers may perform different transformations on their inputs. Signals travel from the first layer (the input layer), to the last layer (the output layer), possibly after traversing the layers multiple times.

Neural networks learn (or are trained) by processing examples, each of which contains a known "input" and "result," forming probability-weighted associations between the two, which are stored within the data structure of the net itself. The training of a neural network from a given example is usually conducted by determining the difference between the processed output of the network (often a prediction) and a target output. This is the error. The network then adjusts its weighted associations according to a

© Edunet Foundation. All rights reserved.

learning rule and using the error value. Successive adjustments will cause the neural network to produce output which is increasingly similar to the target output. After a sufficient number of these adjustments the training can be terminated based upon certain criteria. This is known as supervised learning.

Such systems "learn" to perform tasks by considering examples, generally without being programmed with task-specific rules. For example, in image recognition, they might learn to identify images that contain cats by analyzing example images that have been manually labeled as "cat" or "no cat" and using the results to identify cats in other images. They do this without any prior knowledge of cats, for example, that the cats have fur, tails, whiskers, and cat-like faces. Instead, they automatically generate identifying characteristics from the examples that they process.

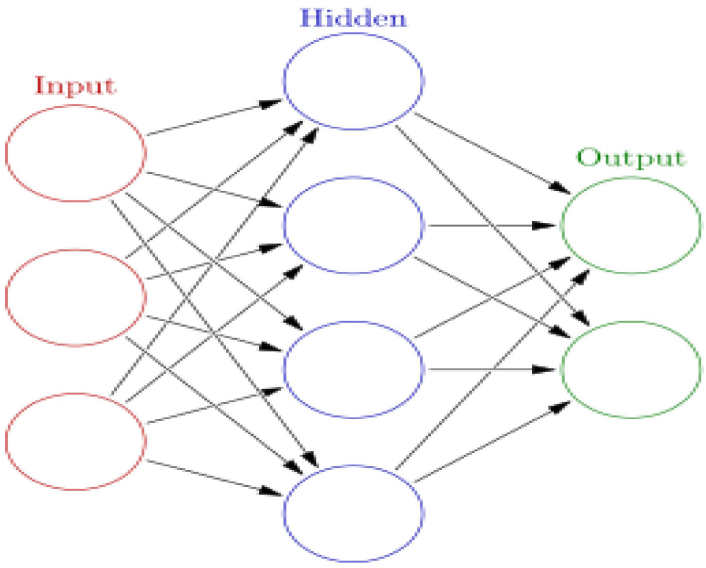


Fig : Neural Network
Reference - Glosser.ca, CC BY-SA 3.0 <https://creativecommons.org/licenses/by-sa/3.0>, via Wikimedia Commons

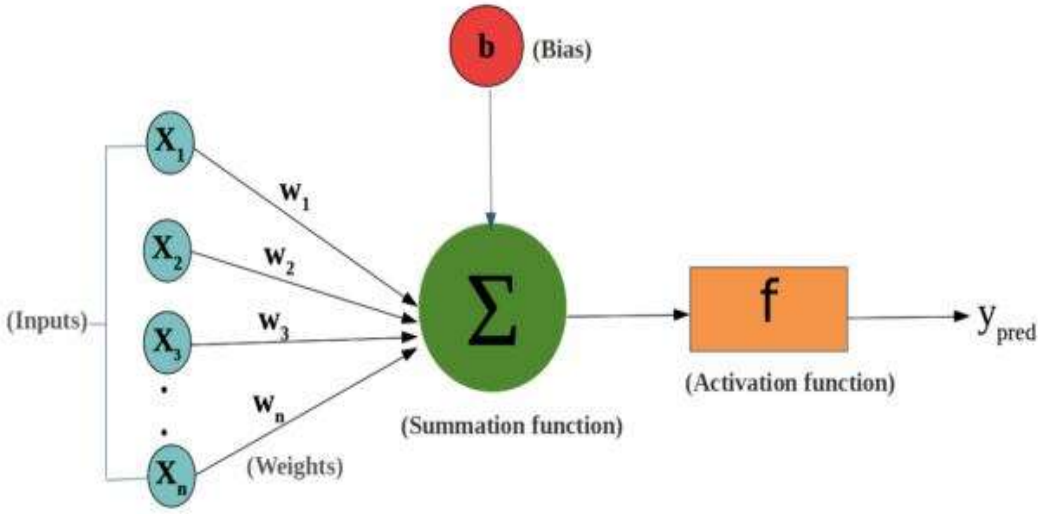


Fig : Basic building blocks of Neural Networks
Reference - https://towardsdatascience.com/whats-the-role-of-weights-and-bias-in-a-neural-network-4cf7e9888a0f

Let us understand the core part of artificial neural networks which is Neuron.

Neurons

ANNs are composed of artificial neurons which are conceptually derived from biological neurons. Each artificial neuron has inputs and produces a single output which can be sent to multiple other neurons. The inputs can be the feature values of a sample of external data, such as images or documents, or they can be the outputs of other neurons. The outputs of the final *output neurons* of the neural net accomplish the task, such as recognizing an object in an image.

To find the output of the neuron, first we take the weighted sum of all the inputs, weighted by the *weights* of the *connections* from the inputs to the neuron. We add a *bias* term to this sum. This weighted sum is sometimes called the *activation*. This weighted sum is then passed through a (usually nonlinear) activation function to produce the output. The initial inputs are external data, such as images and documents. The ultimate outputs accomplish the task, such as recognizing an object in an image.

© Edunet Foundation. All rights reserved.

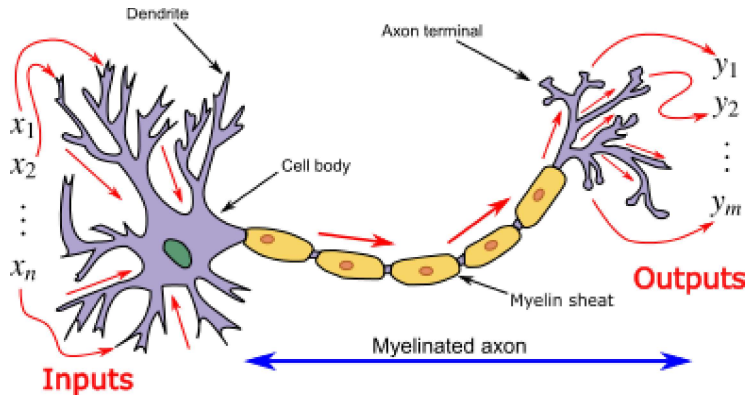


Fig : Neuron
Reference - Egm4313.s12 at English Wikipedia, CC BY-SA 3.0 <https://creativecommons.org/licenses/by-sa/3.0>, via Wikimedia Commons

The activation function of a node defines the output of that node given an input or set of inputs.

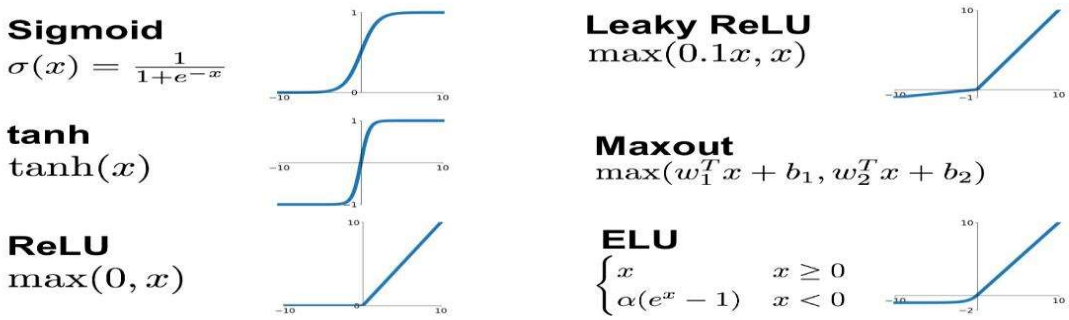


Fig : Different Activation functions
Reference - https://hsf-training.github.io/hsf-training-ml-webpage/03-nn/index.html

Weights

Weights are the real values that are associated with each feature which tells the importance of that feature in predicting the final value.

What do the weights in a Neuron convey to us?

- Importance of the feature

  Weights associated with each feature, convey the importance of that feature in predicting the output value. Features with weights that are close to zero said to have lesser importance in the prediction process compared to the features with weights having a larger value.

- Tells the relationship between a particular feature in the dataset and the target value.

Bias

Bias is used for shifting the activation function towards left or right, it can be referred to as a y-intercept in the line equation.

Forward Propagation

Forward propagation is how neural networks make predictions. Input data is “forward propagated” through the network layer by layer to the final layer which outputs a prediction.

Backpropagation

In machine learning, backpropagation is a widely used algorithm for training feedforward neural networks. Generalizations of backpropagation exist for other artificial neural networks (ANNs), and for functions generally. These classes of algorithms are all referred to generically as "backpropagation". In fitting a neural network, backpropagation computes the gradient of the loss function with respect to the weights of the network for a single input–output example, and does so efficiently, unlike a naive direct computation of the gradient with respect to each weight individually.

This efficiency makes it feasible to use gradient methods for training multilayer networks, updating weights to minimize loss; gradient descent, or variants such as stochastic gradient descent, are commonly used. The backpropagation algorithm works by computing the gradient of the loss function with respect to each weight by the chain rule, computing the gradient one layer at a time, iterating backward from the last layer to avoid redundant calculations of intermediate terms in the chain rule; this is an example of dynamic programming.

© Edunet Foundation. All rights reserved.

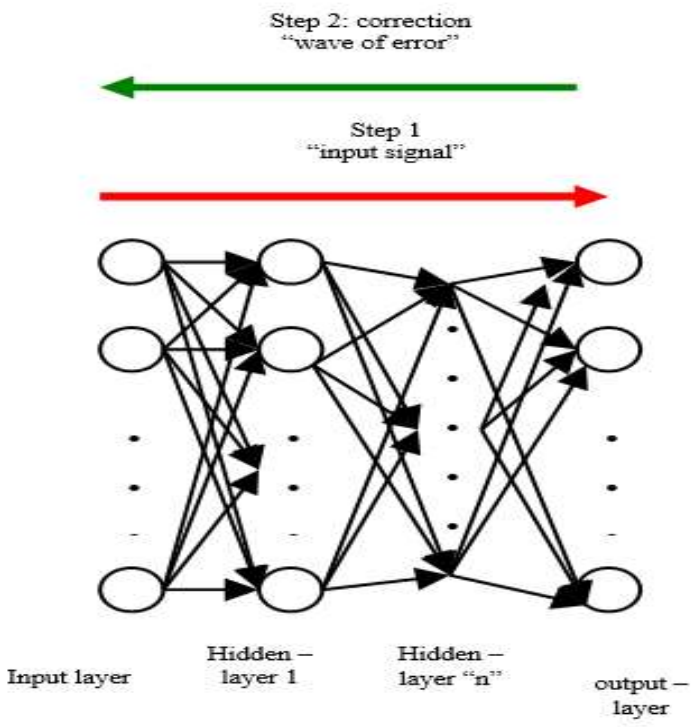


Fig : Forward and backward propagation in Neural Networks
Reference - Jorge Guerra Pires, CC BY-SA 3.0 <https://creativecommons.org/licenses/by-sa/3.0>, via Wikimedia Commons

After understanding the concepts like neuron, bias, weights, forward propagation, backward propagation which plays a vital role in predicting final output. Let's explore some problems like overfitting and underfitting of a model.

Project on Gradient Descent Implementation

We have understood how Gradient Descent works theoretically. Let us implement this practically in Python

Link

https://github.com/Edunet-Foundation/Tech-Saksham/blob/main/Tech%20Saksham%20AI%20Practical%20-%20Notebooks/Chapter%205/3.Gradient_Descent/Pract_Gradient_Descent.ipynb

4.3 Overfitting in Deep Learning

Overfitting is "the production of an analysis that corresponds too closely or exactly to a particular set of data, and may therefore fail to fit additional data or predict future observations reliably". An overfitted model is a statistical model that contains more parameters than can be justified by the data. The essence of overfitting is to have

unknowingly extracted some of the residual variation (i.e. the noise) as if that variation represented underlying model structure.

Underfitting occurs when a statistical model cannot adequately capture the underlying structure of the data. An under-fitted model is a model where some parameters or terms that would appear in a correctly specified model are missing. Under-fitting would occur, for example, when fitting a linear model to non-linear data. Such a model will tend to have poor predictive performance.

The possibility of over-fitting exists because the criterion used for selecting the model is not the same as the criterion used to judge the suitability of a model. For example, a model might be selected by maximizing its performance on some set of training data, and yet its suitability might be determined by its ability to perform well on unseen data; then overfitting occurs when a model begins to "memorize" training data rather than "learning" to generalize from a trend.

As an extreme example, if the number of parameters is the same as or greater than the number of observations, then a model can perfectly predict the training data simply by memorizing the data in its entirety. Such a model, though, will typically fail severely when making predictions.

The potential for overfitting depends not only on the number of parameters and data but also the conformability of the model structure with the data shape, and the magnitude of model error compared to the expected level of noise or error in the data. Even when the fitted model does not have an excessive number of parameters, it is to be expected that the fitted relationship will appear to perform less well on a new data set than on the data set used for fitting (a phenomenon sometimes known as *shrinkage*). In particular, the value of the coefficient of determination will shrink relative to the original data.

To lessen the chance of, or amount of, overfitting, several techniques are available (e.g. model comparison, cross-validation, regularization, early stopping, pruning, Bayesian priors, or dropout). The basis of some techniques is either (1) to explicitly penalize overly complex models or (2) to test the model's ability to generalize by evaluating its performance on a set of data not used for training, which is assumed to approximate the typical unseen data that a model will encounter.

© Edunet Foundation. All rights reserved.

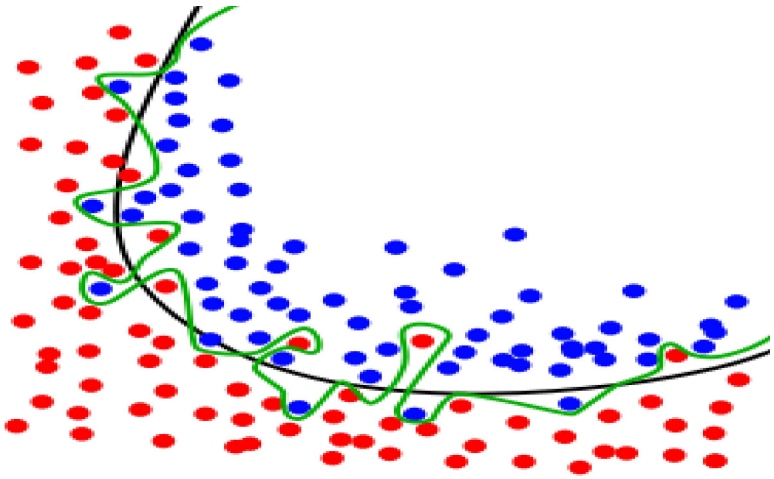


Fig : Green line shows Overfitting
Reference - Chabacano, CC BY-SA 4.0 <https://creativecommons.org/licenses/by-sa/4.0>, via Wikimedia Commons

Measures to prevent overfitting

Reducing network complexity

Deep neural networks like CNN are prone to overfitting because of the millions or billions of parameters it encloses. A model with these many parameters can overfit on the training data because it has sufficient capacity to do so.

By removing certain layers or decreasing the number of neurons (filters in CNN) the network becomes less prone to overfitting as the neurons contributing to overfitting are removed or deactivated. The network also has a reduced number of parameters because of which it cannot memorize all the data points & will be forced to generalize.

There is no general rule as to how many layers are to be removed or how many neurons must be in a layer before the network can overfit. The popular approach for reducing the network complexity is

- Grid search can be applied to find out the number of neurons and/or layers to reduce or remove overfitting.
- The overfit model can be pruned (trimmed) by removing nodes or connections until it reaches suitable performance on test data.

Data Augmentation

One of the best strategies to avoid overfitting is to increase the size of the training dataset. As discussed, when the size of the training data is small the network tends to have greater control over the training data. But in real-world scenarios gathering of large amounts of data is a tedious & time-consuming task, hence the collection of new data is not a viable option.

Data augmentation provides techniques to increase the size of existing training data without any external addition. If our training data consists of images, image augmentation techniques like rotation, horizontal & vertical flipping, translation, increasing or decreasing the brightness or adding noise, cutouts etc can be applied to the existing training images to increase the number of instances.

By applying the above-mentioned data augmentation strategies, the network is trained on multiple instances of the same class of object in different perspectives.

The concept behind data augmentation is that by increasing the size of the training dataset, the network is unable to overfit on all the input samples (original images + augmented images) & thus is forced to generalize. The overall training loss increases because the network doesn't predict accurately on the augmented images thus increasing the training loss & the optimizer (optimization algorithm) tunes the network to capture the generalized trend in the training data.

Weight Regularization

Weight regularization is a technique which aims to stabilize an overfitted network by penalizing the large value of weights in the network. An overfitted network usually presents with problems with a large value of weights as a small change in the input can lead to large changes in the output. For instance, when the network is given new or test data, it results in incorrect predictions.

Weight regularization penalizes the network's large weights & forcing the optimization algorithm to reduce the larger weight values to smaller weights, and this leads to stability of the network & presents good performance. In weight regularization, the network configuration remains unchanged only modifying the value of weights.

Dropouts

Dropout is a regularization strategy that prevents deep neural networks from overfitting. While L1 & L2 regularization reduces overfitting by modifying the loss function, dropouts, on the other hand, deactivate a certain number of neurons at a layer from firing during training.

At each iteration different set of neurons are deactivated & this results in a different set of results. Many deep learning frameworks implement dropouts as a layer which receives inputs from the previous layer, the dropout layer randomly selects neurons which are not fired to the next layer. By deactivating certain neurons which might contribute to overfitting the performance of the network on test data improves.

Dropouts reduce overfitting in a variety of problems like image classification, image segmentation, word embedding etc.

© Edunet Foundation. All rights reserved.

Importance of validation split

The definitions of training, validation, and test sets can be fairly nuanced, and the terms are sometimes inconsistently used. In the deep learning community, “test-time inference” is often used to refer to evaluation on data in production, which is not the technical definition of a test set.

One of the most likely culprits for this disconnect between results in development vs results in production is a poorly chosen validation set (or even worse, no validation set at all). Depending on the nature of your data, choosing a validation set can be the most important step. Although sklearn offers a train_test_split method, this method takes a random subset of the data, which is a poor choice for many real-world problems.

When creating a machine learning model, the ultimate goal is for it to be accurate on new data, not just the data you are using to build it. Consider the below example of 3 different models for a set of data:

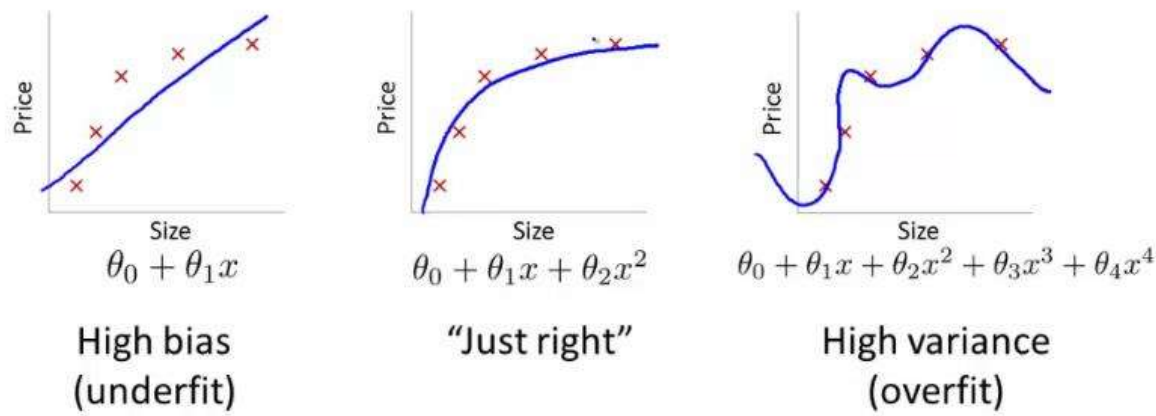


Fig : Bias Variance Tradeoff
Reference - https://www.quora.com/What-is-an-intuitive-explanation-for-bias-variance-tradeoff

The error for the pictured data points is lowest for the model on the far right (the blue curve passes through the red points almost perfectly), yet it’s not the best choice. Why is that? If you were to gather some new data points, they most likely would not be on that curve in the graph on the right, but would be closer to the curve in the middle graph.

The underlying idea is that:

- the training set is used to train a given model
- the validation set is used to choose between models (for instance, does a random forest or a neural net work better for your problem? do you want a random forest with 40 trees or 50 trees?)

© Edunet Foundation. All rights reserved.

- the test set tells you how you've done. If you've tried out a lot of different models, you may get one that does well on your validation set just by chance, and having a test set helps make sure that is not the case.

A key property of the validation and test sets is that they must be representative of the new data you will see in the future. This may sound like an impossible order! By definition, you haven't seen this data yet. But there are still a few things you know about it.[13.3]

We frequently take insights from validation errors to tune our models. So we are implicitly leaking information from our validation data to our model. Advanced validation methods have obscured the importance of single split validation data. K-fold cross-validation is quite robust and probably the current industry standard for model performance validation and parameter tuning. So if you are using cross-validation techniques in your analysis, you may ignore the validation data split.

The primary objective of test data is to give an unbiased estimate of model accuracy. It should be used at the very end and only for a couple of times. If you tune your model after looking at the test accuracies, you are technically leaking information and hence cheating.

For the very same reason as above *(leakage of information)*, in spite of the programming convenience we should not combine train-validation-test dataset to make common preprocessing flow. Some might argue that according to the base hypothesis train-validation-test data come from the same population distribution and hence there should be no harm in combining them for a common preprocessing flow. This is true in idealistic scenarios, but real life if far from it as you never know when your real-time production system start getting evolving data (*whose distribution is slightly different from the training data)*. As a good Data Scientist you should strive to make a model flow that is generalizable and performs well *(without any additional changes) irrespective* of the uncertainties in future data.

We are supposed to develop 2 separate pre-processing pipelines. (A) for training data and (B) for validation and test data. However, it should be noted that these pipelines aren't completely independent. You learn the transformation features *(mean/range/standard-deviation)* from training data and use it to transform your validation and test data. [13.4]

Practical Code – Loss Function (MSE and Cross Entropy)

In this section, we will implement Loss Functions from scratch . For Regression we will use Mean Square Error and for classification we will use Cross Entropy.

https://github.com/Edunet-Foundation/Tech-Saksham/blob/main/Tech%20Saksham%20AI%20Practical%20-

© Edunet Foundation. All rights reserved.

%20Notebooks/Chapter%205/4.Cross_Entropy_Vs_MSE/Pract_Cross_Entropy___MSE.ipynb

Practical Code – Tensors in Deep Learning

Tensors form the base for creating Neural Networks. Let us see some of the popular ways of creating tensors and playing around with them.

https://github.com/Edunet-Foundation/Tech-Saksham/blob/main/Tech%20Saksham%20AI%20Practical%20-%20Notebooks/Chapter%205/5.Tensors/Pract_Tensors.ipynb

Practical Code – Different Types of Layers in Keras

There are many layers which are available in Keras. We have Flatten layer for converting the values from 2D to 1 D. Next, we have Dense Layer which are regularly densely connected layers

https://github.com/Edunet-Foundation/Tech-Saksham/blob/main/Tech%20Saksham%20AI%20Practical%20-%20Notebooks/Chapter%205/6.Dense_Flatten_Layers/Pract_Dense_Flatten_Layers.ipynb

Practical Code – Regularization in Deep Learning

The simplest and perhaps most common regularization method is to add a penalty to the loss function in proportion to the size of the weights in the model. A callback is an object that can perform actions at various stages of training (e.g. at the start or end of an epoch, before or after a single batch, etc).

Practical Code:

https://github.com/Edunet-Foundation/Tech-Saksham/tree/main/Tech%20Saksham%20AI%20Practical%20-%20Notebooks/Chapter%205/8.Regularization_Techniques

https://github.com/Edunet-Foundation/Tech-Saksham/blob/main/Tech%20Saksham%20AI%20Practical%20-%20Notebooks/Chapter%205/9.Custom_Callbacks/Pract_Custom_Callbacks.ipynb

Now let us start with the basics of computer vision.

© Edunet Foundation. All rights reserved.

4.4 Computer Vision Basics

What is Computer Vision?

Computer Vision is a field of study that helps computer understand the content of digital images such as videos and Photographs. Smartphones have cameras, and taking a photo or video and sharing it has never been easier, resulting in the incredible growth of modern social networks like Instagram.

The internet is comprised of text and images. It is relatively straightforward to index and search text, but in order to index and search images, algorithms need to know what the images contain. To get the most out of image data, we need computers to "*see*" an image and understand the content.

It is a multidisciplinary field that could broadly be called a subfield of artificial intelligence and machine learning, which may involve the use of specialized methods and make use of general learning algorithms. The goal of computer vision is to extract useful information from images.

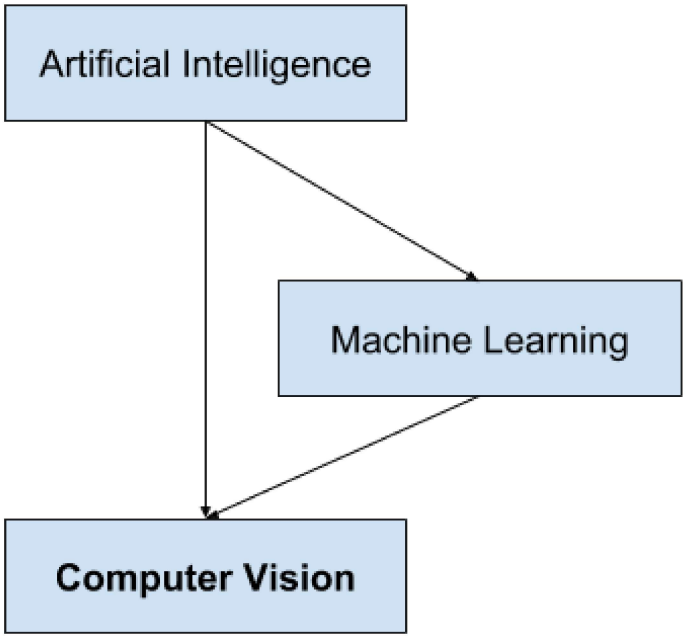


Fig : Overview of the Relationship of Artificial Intelligence and Computer Vision
https://machinelearningmastery.com/wp-content/uploads/2019/03/Computer-Vision.png

Many popular computer vision applications involve trying to recognize some high-level problems; for example:

1. Image Classification
2. Object Detection
3. Optical Character Recognition
4. Image Segmentation

© Edunet Foundation. All rights reserved.

Python Libraries for Computer Vision

- **OpenCV** (Open Source Computer Vision Library: http://opencv.org) is an open-source BSD-licensed library that includes several hundreds of computer vision algorithms.
- **Scikit-Image** - A collection of algorithms for image processing in Python.
- **SimpleCV** - An open source computer vision framework that gives access to several high-powered computer vision libraries, such as OpenCV. Written on Python and runs on Mac, Windows, and Ubuntu Linux.
- **face_recognition** - Face recognition library that recognize and manipulate faces from Python or from the command line.
- **pytessarct** - Python-tesseract is an optical character recognition (OCR) tool for python. That is, it will recognize and "read" the text embedded in images.Python-tesseract is a wrapper for Google's Tesseract-OCR **Engine.**

What are Pixels?

Pixel or Picture Element is smallest addressable element in an image. Each pixel is a sample of an original image; more samples typically provide more accurate representations of the original. The intensity of each pixel is variable. *Pixels* can be used as a unit of measure such as: 2400 pixels per inch, 640 pixels per line, or spaced 10 pixels apart.

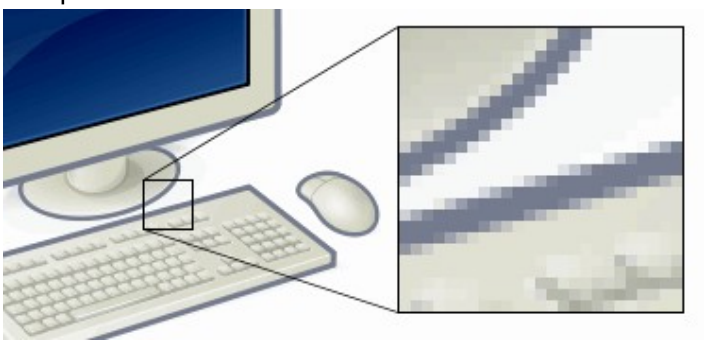


Fig : Image enlarged showing the pixels, rendered as small squares
https://upload.wikimedia.org/wikipedia/commons/2/2b/Pixel-example.png

The more pixels used to represent an image, the closer the result can resemble the original. The number of pixels in an image is sometimes called the resolution, though resolution has a more specific definition. Pixel counts can be expressed as a single number, as in a "three-megapixel" digital camera, which has a nominal three million pixels, or as a pair of numbers, as in a "640 by 480 display", which has 640 pixels from side to side and 480 from top to bottom (as in a VGA display) and therefore has a total number of 640 × 480 = 307,200 pixels, or 0.3 megapixels.

© Edunet Foundation. All rights reserved.

Image as Matrix

Images are represented in rows and columns. For example digital grayscale image is presented in the computer by pixels matrix. Each pixel of such image is presented by one matrix element – integer from the set . The numeric values in pixel presentation are uniformly changed from zero (black pixels) to 255 (white pixels).

0	2	15	0	0	11	10	0	0	0	0	9	9	0	0	0
0	0	0	4	60	157	236	255	255	177	95	61	32	0	0	29
0	10	16	119	238	255	244	245	243	250	249	255	222	103	10	0
0	14	170	255	255	244	254	255	253	245	255	249	253	251	124	1
2	98	255	228	255	251	254	211	141	116	122	215	251	238	255	49
13	217	243	255	155	33	226	52	2	0	10	13	232	255	255	36
16	229	252	254	49	12	0	0	7	7	0	70	237	252	235	62
6	141	245	255	212	25	11	9	3	0	115	236	243	255	137	0
0	87	252	250	248	215	60	0	1	121	252	255	248	144	6	0
0	13	113	255	255	245	255	182	181	248	252	242	208	36	0	19
1	0	5	117	251	255	241	255	247	255	241	162	17	0	7	0
0	0	0	4	58	251	255	246	254	253	255	120	11	0	1	0
0	0	4	97	255	255	255	248	252	255	244	255	182	10	0	4
0	22	206	252	246	251	241	100	24	113	255	245	255	194	9	0
0	111	255	242	255	158	24	0	0	6	39	255	232	230	56	0
0	218	251	250	137	7	11	0	0	0	2	62	255	250	125	3
0	173	255	255	101	9	20	0	13	3	13	182	251	245	61	0
0	107	251	241	255	230	98	55	19	118	217	248	253	255	52	4
0	18	146	250	255	247	255	255	255	249	255	240	255	129	0	5
0	0	23	113	215	255	250	248	255	255	248	248	118	14	12	0
0	0	6	1	0	52	153	233	255	252	147	37	0	0	4	1
0	0	5	5	0	0	0	0	0	14	1	0	6	6	0	0

0	2	15	0	0	11	10	0	0	0	0	9	9	0	0	0
0	0	0	4	60	157	236	255	255	177	95	61	32	0	0	29
0	10	16	119	238	255	244	245	243	250	249	255	222	103	10	0
0	14	170	255	255	244	254	255	253	245	255	249	253	251	124	1
2	98	255	228	255	251	254	211	141	116	122	215	251	238	255	49
13	217	243	255	155	33	226	52	2	0	10	13	232	255	255	36
16	229	252	254	49	12	0	0	7	7	0	70	237	252	235	62
6	141	245	255	212	25	11	9	3	0	115	236	243	255	137	0
0	87	252	250	248	215	60	0	1	121	252	255	248	144	6	0
0	13	113	255	255	245	255	182	181	248	252	242	208	36	0	19
1	0	5	117	251	255	241	255	247	255	241	162	17	0	7	0
0	0	0	4	58	251	255	246	254	253	255	120	11	0	1	0
0	0	4	97	255	255	255	248	252	255	244	255	182	10	0	4
0	22	206	252	246	251	241	100	24	113	255	245	255	194	9	0
0	111	255	242	255	158	24	0	0	6	39	255	232	230	56	0
0	218	251	250	137	7	11	0	0	0	2	62	255	250	125	3
0	173	255	255	101	9	20	0	13	3	13	182	251	245	61	0
0	107	251	241	255	230	98	55	19	118	217	248	253	255	52	4
0	18	146	250	255	247	255	255	255	249	255	240	255	129	0	5
0	0	23	113	215	255	250	248	255	255	248	248	118	14	12	0
0	0	6	1	0	52	153	233	255	252	147	37	0	0	4	1
0	0	5	5	0	0	0	0	0	14	1	0	6	6	0	0

Fig : Image with Pixels
https://mozanunal.com/images/pixel.png

Types of Images

Images can be divided in 3 different categories

1. Binary Images

It is the simplest type of image. It takes only two values i.e, Black and White or 0 and 1. The binary image consists of a 1-bit image and it takes only 1 binary digit to represent a pixel. Binary images are mostly used for general shape or outline.

For Example: Optical Character Recognition (OCR).

Binary images are generated using threshold operation. When a pixel is above the threshold value, then it is turned white('1') and which are below the threshold value then they are turned black('0')

© Edunet Foundation. All rights reserved.

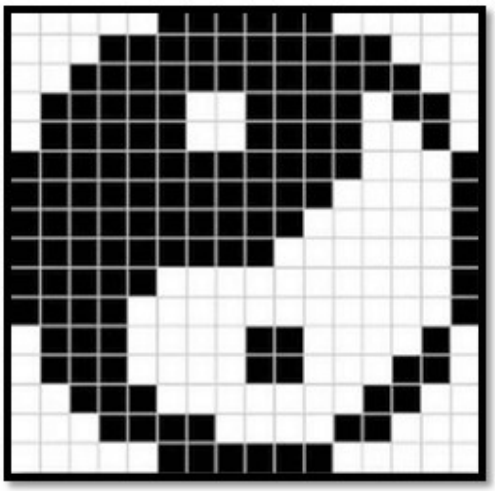


Fig. Binary image

https://static.javatpoint.com/tutorial/dip/images/binary-images.png

2. Gray-scale images

Grayscale images are monochrome images, Means they have only one color. Grayscale images do not contain any information about color. Each pixel determines available different grey levels.

A normal grayscale image contains 8 bits/pixel data, which has 256 different grey levels. In medical images and astronomy, 12 or 16 bits/pixel images are used.

Fig. Gray-scale image

https://static.javatpoint.com/tutorial/dip/images/gray-scale-images.png

3. Colour images

Colour images are three band monochrome images in which, each band contains a different color and the actual information is stored in the digital image. The color images contain gray level information in each spectral band.

The images are represented as red, green and blue (RGB images). And each color image has 24 bits/pixel means 8 bits for each of the three color band(RGB).

Fig. Colour image

https://static.javatpoint.com/tutorial/dip/images/colour-images.png

After understanding the basics of image and pixels, let us get started with exploring the concepts of Image processing.

4.5 Image Processing

What is Image Processing?

Image processing is a method to perform some operations on an image, in order to get an enhanced image or to extract some useful information from it. It is a type of signal processing in which input is an image and output may be an image or characteristics/features associated with that image.

Why do we need it?

The data that we collect or generate is mostly raw data, i.e. it is not fit to be used in applications directly due to a number of possible reasons. Therefore, we need to analyze it first, perform the necessary pre-processing, and then use it.

For instance, let's assume that we were trying to build a cat classifier. Our program would take an image as input and then tell us whether the image contains a cat or not. The first step for building this classifier would be to collect hundreds of cat pictures. One common issue is that all the pictures we have scraped would not be of the same size/dimensions, so before feeding them to the model for training, we would need to

© Edunet Foundation. All rights reserved.

resize/pre-process them all to a standard size and apply filters on them to extract useful information.

This is just one of many reasons why image processing is essential to any computer vision

Image processing mainly include the following steps:

1.Importing the image via image acquisition tools;
2. Analysing and manipulating the image;
3.Output in which result can be altered image or a report which is based on analysing that image.

Convolution

Convolution is the process of adding each element of the image to its local neighbors, weighted by the kernel. This is related to a form of mathematical convolution. The matrix operation being performed—convolution—is not traditional matrix multiplication, despite being similarly denoted by *.

For example, if we have two three-by-three matrices, the first a kernel, and the second an image piece, convolution is the process of flipping both the rows and columns of the kernel and multiplying locally similar entries and summing. The element at coordinates [2, 2] (that is, the central element) of the resulting image would be a weighted combination of all the entries of the image matrix, with weights given by the kernel:

$$\left(\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} * \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}\right)[2,2] = (i \cdot 1) + (h \cdot 2) + (g \cdot 3) + (f \cdot 4) + (e \cdot 5) + (d \cdot 6) + (c \cdot 7) + (b \cdot 8) + (a \cdot 9).$$

Fig : Convolution Formula

https://wikimedia.org/api/rest_v1/media/math/render/svg/570600fdeed436d98626278f22bf034ff5ab5162

The other entries would be similarly weighted, where we position the center of the kernel on each of the boundary points of the image and compute a weighted sum.

Filters/Kernels in Image Processing

In image processing, a kernel, convolution matrix, or mask is a small matrix used to apply effects like the ones you might find in Photoshop or Gimp. It is used for blurring, sharpening, embossing, edge detection, and more. This is accomplished by doing a convolution between a kernel and an image.

© Edunet Foundation. All rights reserved.

Depending on the element values, a kernel can cause a wide range of effects.

Fig : Unmodified (left) and the same image processed with a blur kernel.
https://www.naturefocused.com/articles/kernel-blur-mask.jpg

Some popular types of kernels

Edge detection: this kernel detects edges within an image. A 3x3 example:

0	-1	0
-1	4	-1
0	-1	0

Fig : 3*3 Filter

Notice that if all pixel values are comparable, the the resultant pixel value will be close to 0. However, edges - locations with extreme differences in pixel values - will result in values far from zero.

Gaussian Blur: This kernel is similar to the blur kernel presented above, but is different in that it is dependent upon the Gaussian function - a function which creates a distribution of values around the center point. This results in a kernel in which pixels near the center contribute more towards the new pixel value than those further away.

Sharpening: This kernel sharpens an image - accentuating the edges of the image. Sharpening an image add contrast to edges, and a 3x3 version of this mask is similar

© Edunet Foundation. All rights reserved.

to the edge detection kernel with a center value of 5. This adds contrast around an edge by accentuating bright and dark areas.

Unsharp Mask: Used to sharpen an image, this technique is based upon first creating a gaussian blurred copy of the image. This blurred copy is then subtracted from the original - pixels above a given threshold are sharpened by enhancing light and dark pixels.

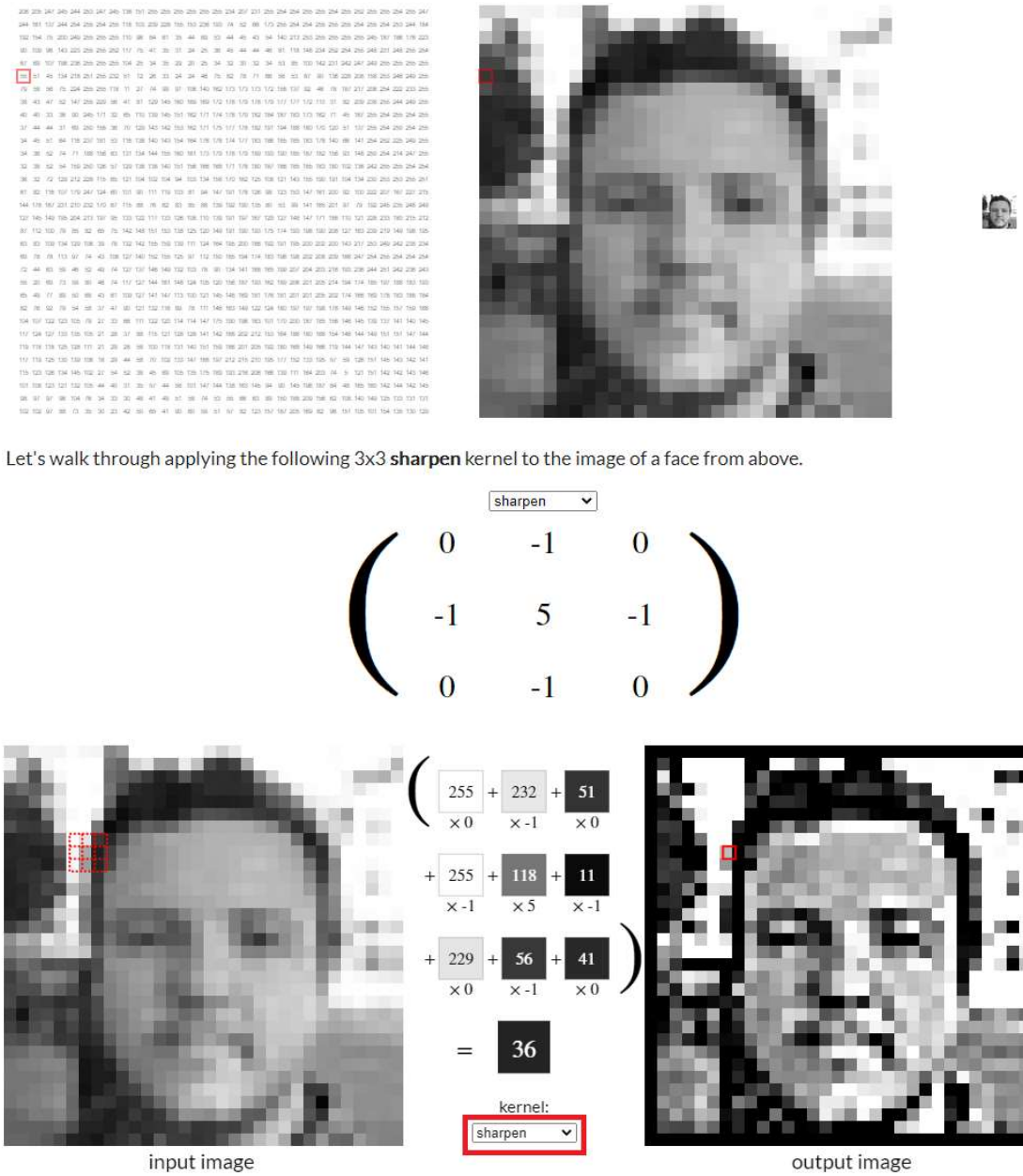


Fig : Applying Kernel on Image
https://setosa.io/ev/image-kernels/

Check the above link for Image Kernels demo

© Edunet Foundation. All rights reserved.

4.6 Facial Detection

Face detection is a computer technology being used in a variety of applications that identifies human faces in digital images. It is the action of locating human faces in an image and optionally returning different kinds of face-related data. At a minimum, each detected face corresponds to a faceRectangle field in the response. This set of pixel coordinates for the left, top, width, and height mark the located face. Using these coordinates, you can get the location of the face and its size.

Try the demo from this link
https://azure.microsoft.com/en-us/services/cognitive-services/face/#demo

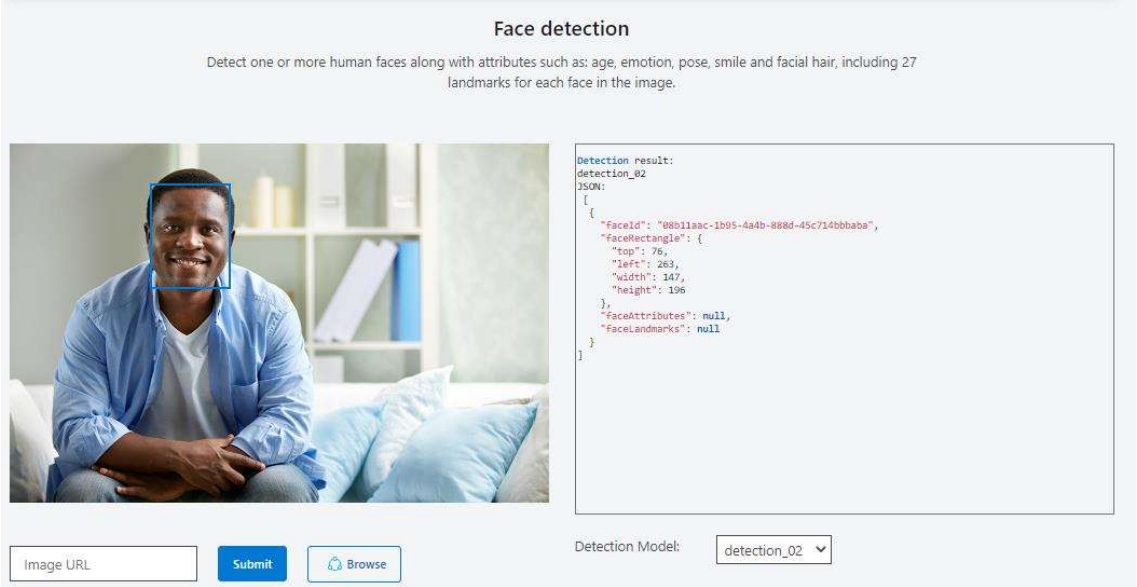


Fig : Face Detection Demo
https://azure.microsoft.com/en-us/services/cognitive-services/face/#demo

Face landmarks

Face landmarks are a set of easy-to-find points on a face, such as the pupils or the tip of the nose. By default, there are 27 predefined landmark points. The following figure shows all 27 points:

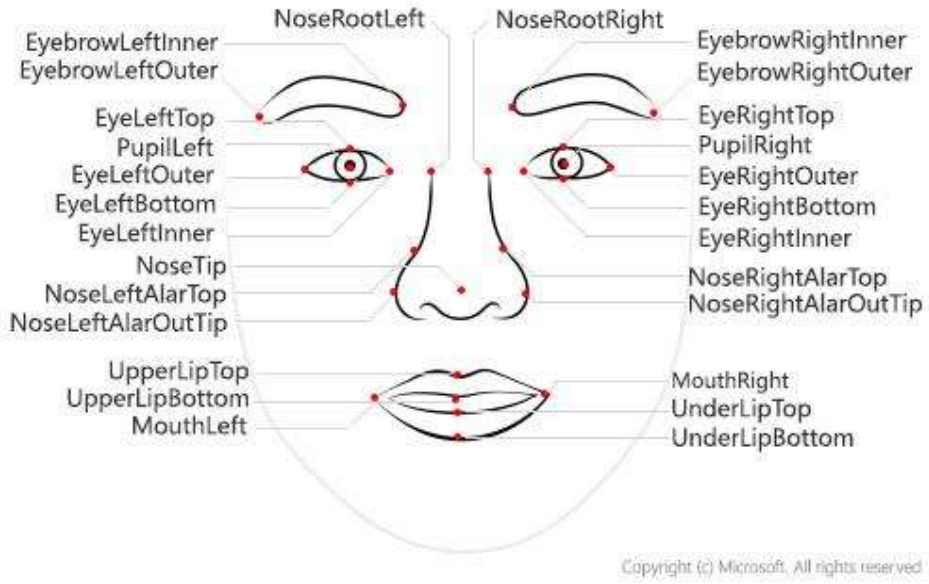


Fig : Face Landmarks
https://docs.microsoft.com/en-us/azure/cognitive-services/face/images/landmarks.1.jpg

Face Detection With OpenCV

Feature-based face detection algorithms are fast and effective and have been used successfully for decades. Perhaps the most successful example is a technique called cascade classifiers first described by Paul Viola and Michael Jones and their 2001 paper titled "Rapid Object Detection using a Boosted Cascade of Simple Features." In the paper, effective features are learned using the AdaBoost algorithm, although importantly, multiple models are organized into a hierarchy or "*cascade*."

Their detector, called detector cascade, consists of a sequence of simple-to-complex face classifiers and has attracted extensive research efforts. Moreover, detector cascade has been deployed in many commercial products such as smartphones and digital cameras.

A modern implementation of the Classifier Cascade face detection algorithm is provided in the OpenCV library. This is a C++ computer vision library that provides a python interface. The benefit of this implementation is that it provides pre-trained face detection models, and provides an interface to train a model on your own dataset.

OpenCV can be installed by the package manager system on your platform, or via pip; for example:

```
pip install opencv-python
```

OpenCV provides the CascadeClassifier class that can be used to create a cascade classifier for face detection. The constructor can take a filename as an argument that specifies the XML file for a pre-trained model.

OpenCV provides a number of pre-trained models as part of the installation. These are available on your system and are also available on the OpenCV GitHub project. Download a pre-trained model for frontal face detection from the OpenCV GitHub project and place it in your current working directory with the filename '*haarcascade_frontalface_default.xml*'.

Here we will work with face detection. Initially, the algorithm needs a lot of positive images (images of faces) and negative images (images without faces) to train the classifier. Then we need to extract features from it. For this, Haar features shown in the below image are used. They are just like our convolutional kernel. Each feature is a single value obtained by subtracting sum of pixels under the white rectangle from sum of pixels under the black rectangle.

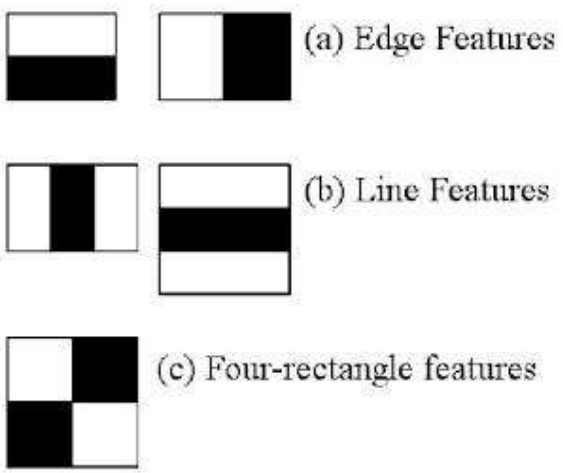


Fig : Features of an Image
https://docs.opencv.org/3.4.3/haar_features.jpg

Now, all possible sizes and locations of each kernel are used to calculate lots of features. (Just imagine how much computation it needs? Even a 24x24 window results over 160000 features). For each feature calculation, we need to find the sum of the pixels under white and black rectangles. To solve this, they introduced the integral image. However large your image, it reduces the calculations for a given pixel to an operation involving just four pixels. Nice, isn't it? It makes things super-fast.

But among all these features we calculated, most of them are irrelevant. For example, consider the image below. The top row shows two good features. The first feature

© Edunet Foundation. All rights reserved.

selected seems to focus on the property that the region of the eyes is often darker than the region of the nose and cheeks. The second feature selected relies on the property that the eyes are darker than the bridge of the nose. But the same windows applied to cheeks or any other place is irrelevant. So how do we select the best features out of 160000+ features? It is achieved by **Adaboost**.

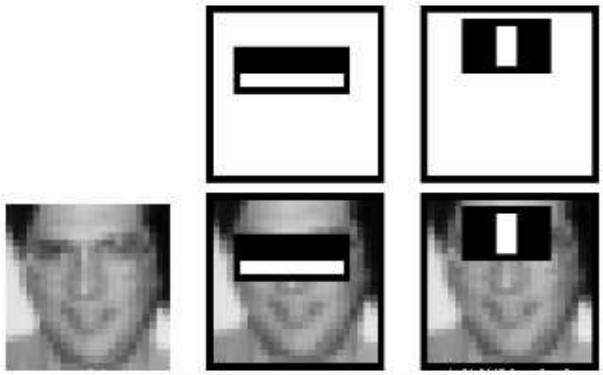


Fig : Applying Haarcascade on Faces
https://docs.opencv.org/3.4.3/haar.png

For this, we apply each and every feature on all the training images. For each feature, it finds the best threshold which will classify the faces to positive and negative. Obviously, there will be errors or misclassifications. We select the features with minimum error rate, which means they are the features that most accurately classify the face and non-face images. (The process is not as simple as this. Each image is given an equal weight in the beginning. After each classification, weights of misclassified images are increased. Then the same process is done. New error rates are calculated. Also new weights. The process is continued until the required accuracy or error rate is achieved or the required number of features are found).

The final classifier is a weighted sum of these weak classifiers. It is called weak because it alone can't classify the image, but together with others forms a strong classifier. The paper says even 200 features provide detection with 95% accuracy. Their final setup had around 6000 features. (Imagine a reduction from 160000+ features to 6000 features. That is a big gain).

So now you take an image. Take each 24x24 window. Apply 6000 features to it. Check if it is face or not. Wow.. Isn't it a little inefficient and time consuming? Yes, it is. The authors have a good solution for that.

In an image, most of the image is non-face region. So it is a better idea to have a simple method to check if a window is not a face region. If it is not, discard it in a single shot, and don't process it again. Instead, focus on regions where there can be a face. This way, we spend more time checking possible face regions.

© Edunet Foundation. All rights reserved.

For this they introduced the concept of **Cascade of Classifiers**. Instead of applying all 6000 features on a window, the features are grouped into different stages of classifiers and applied one-by-one. (Normally the first few stages will contain very many fewer features). If a window fails the first stage, discard it. We don't consider the remaining features on it. If it passes, apply the second stage of features and continue the process. The window which passes all stages is a face region.

Implementation of Face Detection in Code

```
faces = face_classifier.detectMultiScale(gray, 1.1, 6)
```

In this piece of code what we are trying to do is, using the **face_classifier** which is an object loaded with **haarcascade_frontalface_default.xml,** we are using an inbuilt function with it called the **detectMultiScale.**
This function will help us to find the features/locations of the new image. The way it does is, it will use all the features from the **face_classifier** object to detect the features of the new image.

The parameters that we will pass to this function are:

1. The gray scale variable — gray in our case
2. **scaleFactor** — Parameter specifying how much the image size is reduced at each image scale. Basically, the scale factor is used to create your scale pyramid. More explanation, your model has a fixed size defined during training, which is visible in the XML. This means that this size of the face is detected in the image if present. However, by rescaling the input image, you can resize a larger face to a smaller one, making it detectable by the algorithm. 1.05 is a good possible value for this, which means you use a small step for resizing, i.e. reduce the size by 5%, you increase the chance of a matching size with the model for detection is found.
3. **minNeighbors** — Parameter specifying how many neighbors each candidate rectangle should have to retain it. This parameter will affect the quality of the detected faces. Higher value results in fewer detections but with higher quality. 3~6 is a good value for it.

Practical Code for CV Basics and Image Processing:

https://github.com/Edunet-Foundation/Tech-Saksham/blob/main/Tech%20Saksham%20AI%20Practical%20-%20Notebooks/Chapter%205/10.%20OpenCV%20Basics.ipynb

So, in this chapter we explored the concepts of deep learning like artificial neural networks, image processing. Happy Learning.

Do it Yourself

Create a simple CNN model for Image Classification
In this exercise, Create a simple CNN-based image classifier for images of simple geometric shapes. The dataset named 'simple-shapes' is given to you which contains 300 train images and 200 test images of circle, square, triangle, rectangle, ellipse etc.
Self-paced Courses –
https://docs.microsoft.com/en-us/learn/modules/train-evaluate-deep-learn-models/

Please refers below mentioned Links to Codes/ Practical Notebooks of all the chapters

1. Chapter 2
2. Chapter 3
3. Chapter 4
4. Chapter 5

Reference

1. https://en.wikiversity.org/wiki/Artificial_intelligence/Introduction
2. https://docs.microsoft.com/en-us/azure/architecture/data-guide/big-data/ai-overview
3. https://www.javatpoint.com/history-of-artificial-intelligence
4. https://aidemos.microsoft.com/
5. https://lobe.ai/
6. https://en.wikipedia.org/wiki/Anaconda_(Python_distribution)
7. https://en.wikipedia.org/wiki/Python_(programming_language)
8. https://www.python.org/downloads/release/python-394/
9. https://www.datacamp.com/community/tutorials/functions-python-tutorial
10. https://www.tutorialspoint.com/difference-between-method-and-function-in-python
11. https://www.w3schools.com/python/python_conditions.asp
12. https://www.geeksforgeeks.org/python-data-types/
13. https://www.tutorialspoint.com/python/python_strings.htm
14. https://en.wikipedia.org/wiki/Data_analysis

15. https://techdifferences.com/difference-between-descriptive-and-predictive-data-mining.html
16. https://www.tutorialspoint.com/numpy
17. https://numpy.org/
18. https://www.tutorialspoint.com/python_data_science
19. https://www.w3schools.in/python-tutorial/decision-making/
20. https://www.geeksforgeeks.org/python-broadcasting-with-numpy-arrays/
21. https://realpython.com/
22. https://www.tableau.com/learn/articles/data-visualization#:~:text=Data%20visualization%20is%20the%20graphical,outliers%2C%20and%20patterns%20in%20data
23. https://www.geeksforgeeks.org/python-matplotlib-pyplot-ticks/
24. https://www.geeksforgeeks.org/matplotlib-pyplot-legend-in-python/
25. https://jakevdp.github.io/PythonDataScienceHandbook/04.06-customizing-legends.html
26. https://jakevdp.github.io/PythonDataScienceHandbook/04.12-three-dimensional-plotting.html
27. https://www.geeksforgeeks.org/three-dimensional-plotting-in-python-using-matplotlib/
28. https://pythontic.com/visualization/charts/piechart
29. https://www.geeksforgeeks.org/plot-a-pie-chart-in-python-using-matplotlib/
30. https://www.tutorialspoint.com/numpy/numpy_matplotlib.htm:
31. https://www.edureka.co/blog/python-numpy-tutorial/
32. https://pandas.pydata.org/docs/getting_started/index.html
33. https://datatofish.com/install-package-python-using-pip/
34. https://jakevdp.github.io/PythonDataScienceHandbook/03.01-introducing-pandas-objects.html
35. https://pandas.pydata.org/pandas-docs/stable/user_guide/io.html
36. https://www.bigcommerce.com/ecommerce-answers/what-csv-file-and-what-does-it-mean-my-ecommerce-business/#:~:text=A%20CSV%20is%20a%20comma,Microsoft%20Excel%20or%20Google%20Spreadsheets.
37. https://www.w3resource.com/JSON/structures.php
38. https://www.shanelynn.ie/python-pandas-read-csv-load-data-from-csv-files/
39. https://datatofish.com/load-json-pandas-dataframe/
40. https://www.shanelynn.ie/summarising-aggregation-and-grouping-data-in-python-pandas
41. https://towardsdatascience.com/pandas-groupby-explained-453692519d0
42. https://towardsdatascience.com/a-step-by-step-guide-to-pandas-pivot-tables-e0641d0c6c70
43. https://pandas.pydata.org/docs/reference/api/pandas.pivot_table.html
44. https://datatofish.com/plot-dataframe-pandas
45. https://towardsdatascience.com/an-introduction-to-scikit-learn-the-gold-standard-of-python-machine-learning-e2b9238a98ab

© Edunet Foundation. All rights reserved.

46. https://www.javatpoint.com/regression-vs-classification-in-machine-learning
47. https://towardsdatascience.com/mathematics-for-machine-learning-linear-regression-least-square-regression-de09cf53757c
48. https://www.javatpoint.com/linear-regression-in-machine-learning
49. https://www.i2tutorials.com/tag/ordinary-least-square-method-in-machine-learning/
50. https://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning
51. https://towardsdatascience.com/basic-probability-theory-and-statistics-3105ab637213
52. https://www.javatpoint.com/machine-learning-naive-bayes-classifier
53. Andreas C. Müller and Sarah Guido , Introduction to Machine learning with Python , O'reilly , October 2016.
54. Chire, CC BY-SA 4.0 <https://creativecommons.org/licenses/by-sa/4.0>, via Wikimedia Commons\
55. https://en.wikipedia.org/wiki/Deep_learning
56. https://www.geeksforgeeks.org/difference-between-machine-learning-and-deep-learning/
57. https://en.wikipedia.org/wiki/Artificial_neural_network
58. https://towardsdatascience.com/whats-the-role-of-weights-and-bias-in-a-neural-network-4cf7e9888a0f
59. https://ml-cheatsheet.readthedocs.io/en/latest/forwardpropagation.html
60. https://en.wikipedia.org/wiki/Overfitting
61. https://medium.com/analytics-vidhya/the-perfect-fit-for-a-dnn-596954c9ea39
62. https://www.kdnuggets.com/2017/11/create-good-validation-set.html
63. https://medium.datadriveninvestor.com/data-science-essentials-why-train-validation-test-data-b7f7d472dc1f
64. https://machinelearningmastery.com/what-is-computer-vision/
65. https://en.wikipedia.org/wiki/Pixel
66. https://www.javatpoint.com/dip-types-of-images
67. https://stackabuse.com/introduction-to-image-processing-in-python-with-opencv/
68. https://www.naturefocused.com/articles/photography-image-processing-kernel.html
69. https://setosa.io/ev/image-kernels/

© Edunet Foundation. All rights reserved.