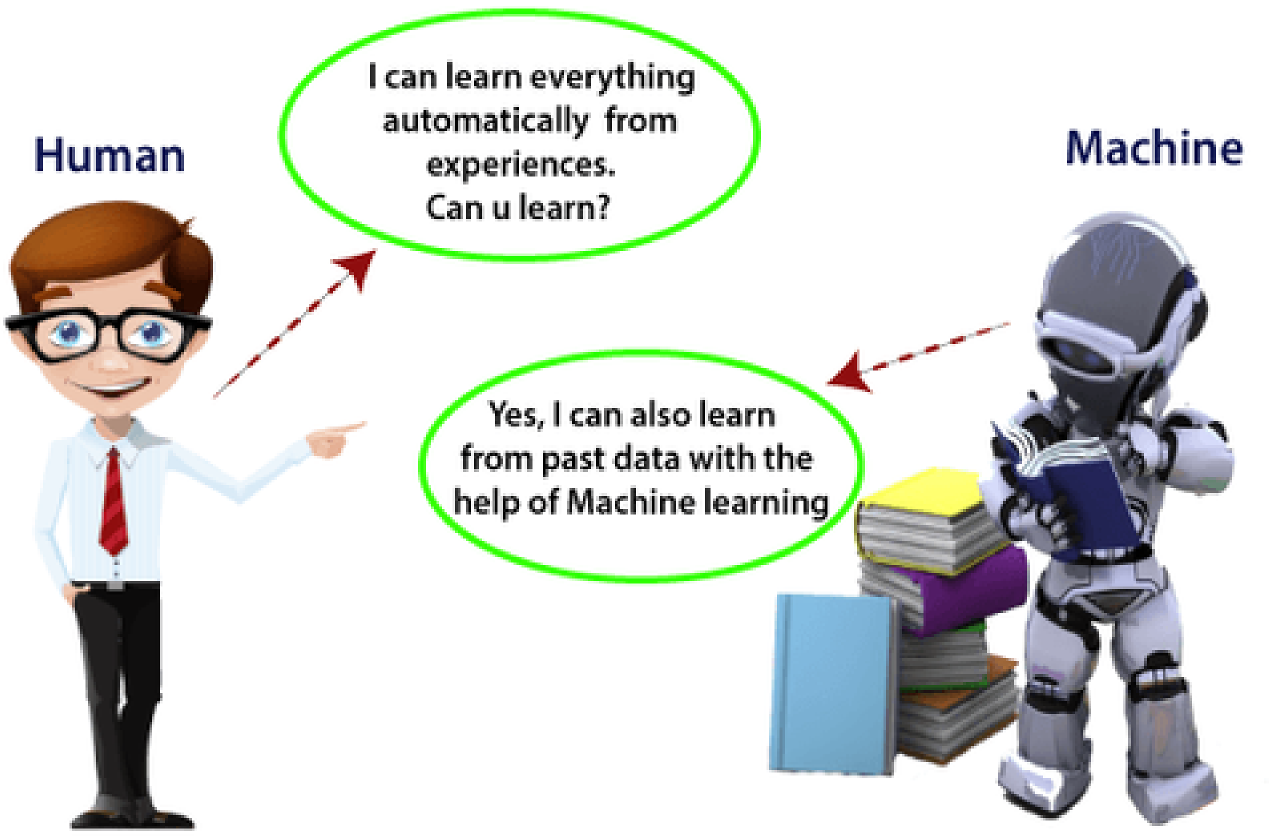
UNIT 01

Machine Learning

In the real world, we are surrounded by humans who can learn everything from their experiences with their learning capability, and we have computers or machines which work on our instructions. But can a machine also learn from experiences or past data like a human does? So here comes the role of **Machine Learning**.


Human
I can learn everything automatically from experiences. Can u learn?
Machine
Yes, I can also learn from past data with the help of Machine learning

Machine Learning is said as a subset of **artificial intelligence** that is mainly concerned with the development of algorithms which allow a computer to learn from the data and past experiences on their own. The term machine learning was first introduced by **Arthur Samuel** in **1959**. We can define it in a summarized way as:

Machine learning enables a machine to automatically learn from data, improve performance from experiences, and predict things without being explicitly programmed.

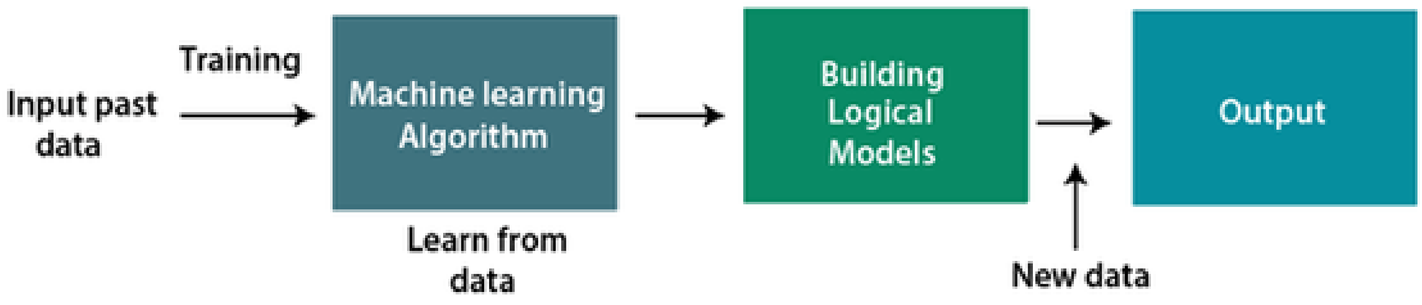
With the help of sample historical data, which is known as **training data**, machine learning algorithms build a **mathematical model** that helps in making predictions or decisions without being explicitly programmed. Machine learning brings computer science and statistics together for creating predictive models. Machine learning constructs or uses the algorithms that learn from historical data. The more we will provide the information, the higher will be the performance.

A machine has the ability to learn if it can improve its performance by gaining more data.

How does Machine Learning work

A Machine Learning system **learns from historical data, builds the prediction models, and whenever it receives new data, predicts the output for it**. The accuracy of predicted output depends upon the amount of data, as the huge amount of data helps to build a better model which predicts the output more accurately.

Suppose we have a complex problem, where we need to perform some predictions, so instead of writing a code for it, we just need to feed the data to generic algorithms, and with the help of these algorithms, machine builds the logic as per the data and predict the output. Machine learning has changed our way of thinking about the problem. The below block diagram explains the working of Machine Learning algorithm:



Features of Machine Learning:

- Machine learning uses data to detect various patterns in a given dataset.
- It can learn from past data and improve automatically.
- It is a data-driven technology.
- Machine learning is much similar to data mining as it also deals with the huge amount of the data.

Need for Machine Learning

The need for machine learning is increasing day by day. The reason behind the need for machine learning is that it is capable of doing tasks that are too complex for a person to implement directly. As a human, we have some limitations as we cannot access the huge amount of data manually, so for this, we need some computer systems and here comes the machine learning to make things easy for us.

We can train machine learning algorithms by providing them the huge amount of data and let them explore the data, construct the models, and predict the required output automatically. The performance of the machine learning algorithm depends on the amount of data, and it can be determined by the cost function. With the help of machine learning, we can save both time and money.

The importance of machine learning can be easily understood by its uses cases, Currently, machine learning is used in **self-driving cars**, **cyber fraud detection**, **face recognition**, and **friend suggestion by Facebook**, etc. Various top companies such as Netflix and Amazon have build machine learning models that are using a vast amount of data to analyze the user interest and recommend product accordingly.

Following are some key points which show the importance of Machine Learning:

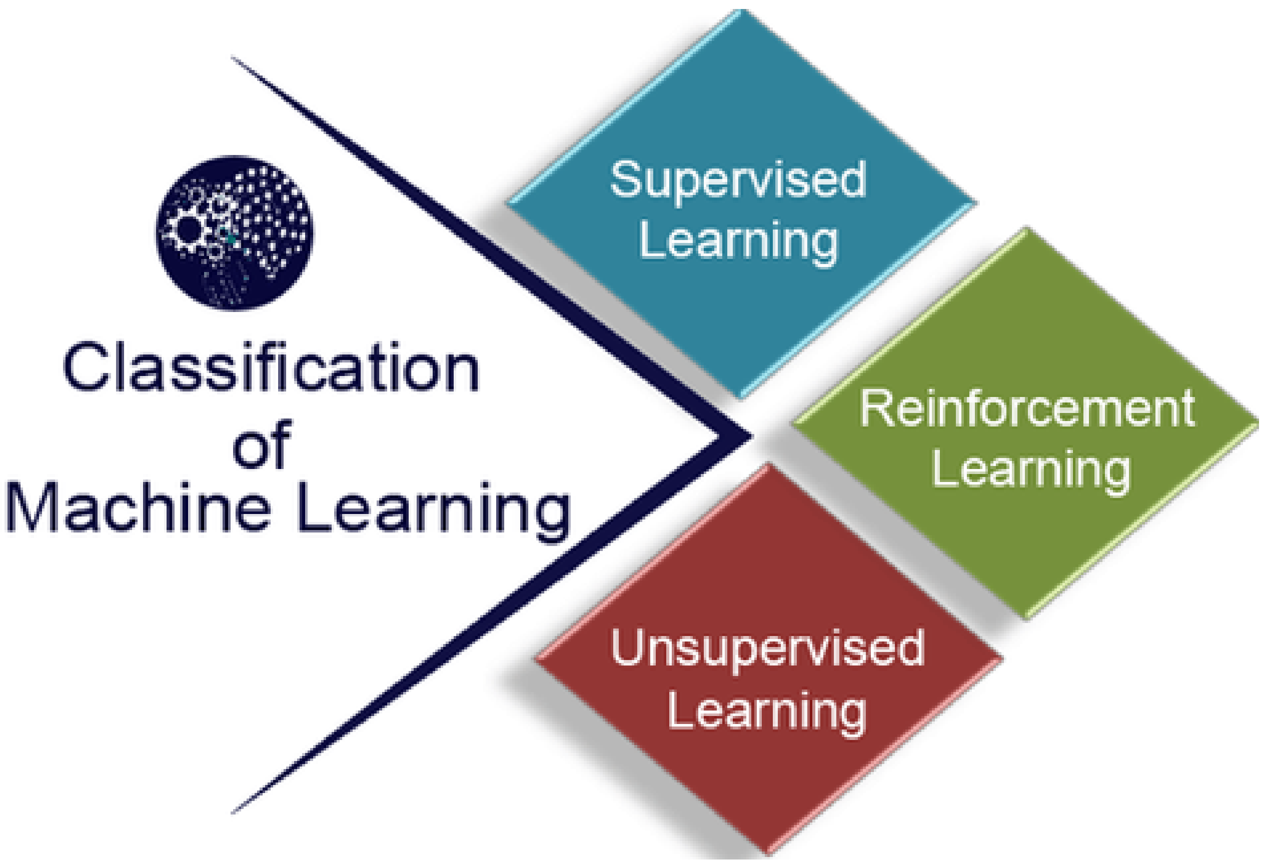
- Rapid increment in the production of data
- Solving complex problems, which are difficult for a human

- Decision making in various sector including finance
- Finding hidden patterns and extracting useful information from data.

Classification of Machine Learning

At a broad level, machine learning can be classified into three types:

1. **Supervised learning**
2. **Unsupervised learning**
3. **Reinforcement learning**



1) Supervised Learning

Supervised learning is a type of machine learning method in which we provide sample labeled data to the machine learning system in order to train it, and on that basis, it predicts the output.

The system creates a model using labeled data to understand the datasets and learn about each data, once the training and processing are done then we test the model by providing a sample data to check whether it is predicting the exact output or not.

The goal of supervised learning is to map input data with the output data. The supervised learning is based on supervision, and it is the same as when a student learns things in the supervision of the teacher. The example of supervised learning is **spam filtering**.

Supervised learning can be grouped further in two categories of algorithms:

- **Classification**
- **Regression**

2) Unsupervised Learning

Unsupervised learning is a learning method in which a machine learns without any supervision.

The training is provided to the machine with the set of data that has not been labeled, classified, or categorized, and the algorithm needs to act on that data without any supervision. The goal of unsupervised learning is to restructure the input data into new features or a group of objects with similar patterns.

In unsupervised learning, we don't have a predetermined result. The machine tries to find useful insights from the huge amount of data. It can be further classifieds into two categories of algorithms:

- **Clustering**
- **Association**

3) Reinforcement Learning

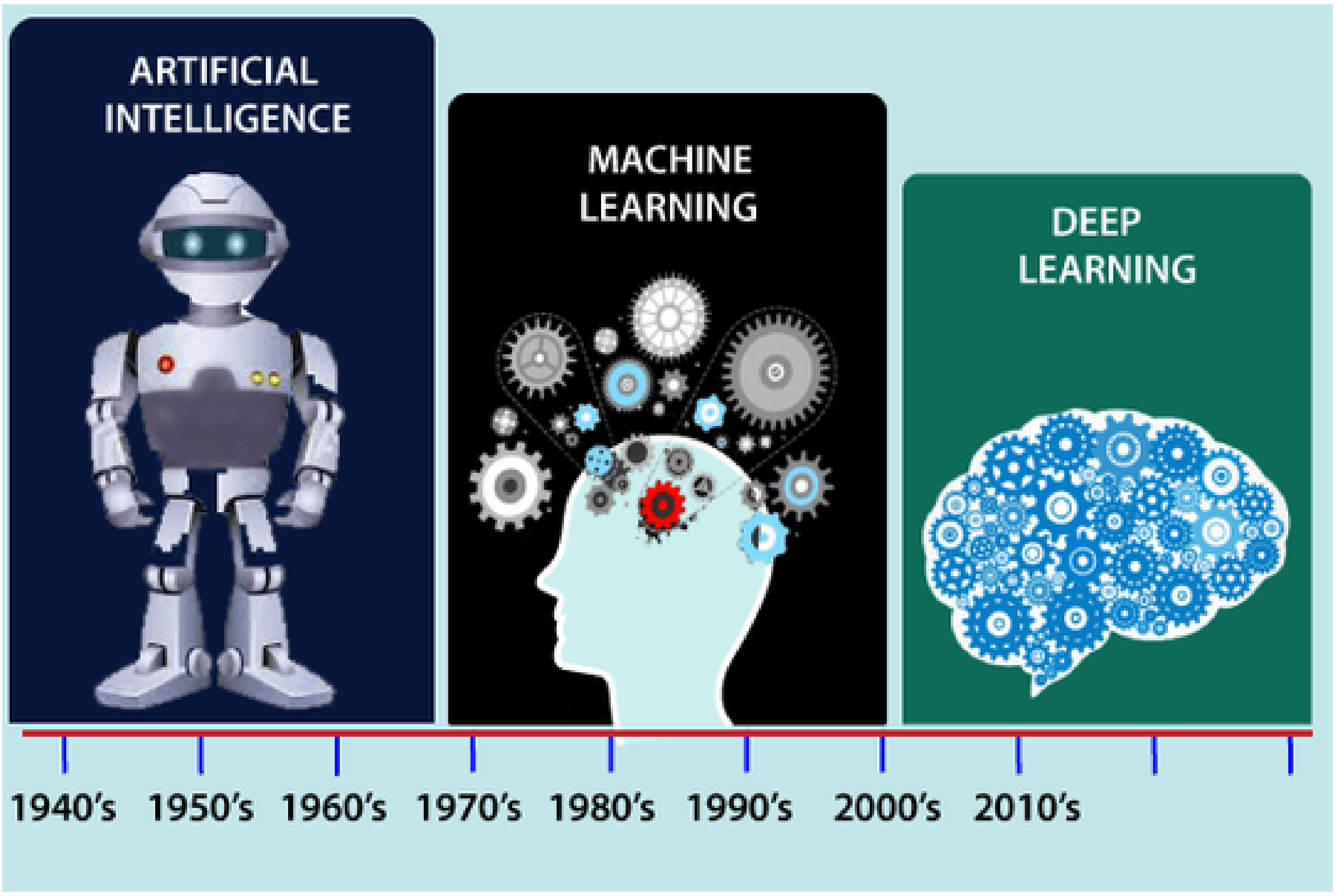
Reinforcement learning is a feedback-based learning method, in which a learning agent gets a reward for each right action and gets a penalty for each wrong action. The agent learns automatically with these feedbacks and improves its performance. In reinforcement learning, the agent interacts with the environment and explores it. The goal of an agent is to get the most reward points, and hence, it improves its performance.

The robotic dog, which automatically learns the movement of his arms, is an example of Reinforcement learning.

Note: We will learn about the above types of machine learning in detail in later chapters.

History of Machine Learning

Before some years (about 40-50 years), machine learning was science fiction, but today it is the part of our daily life. Machine learning is making our day to day life easy from **self-driving cars** to **Amazon virtual assistant "Alexa"**. However, the idea behind machine learning is so old and has a long history. Below some milestones are given which have occurred in the history of machine learning:



The early history of Machine Learning (Pre-1940):

- **1834:** In 1834, Charles Babbage, the father of the computer, conceived a device that could be programmed with punch cards. However, the machine was never built, but all modern computers rely on its logical structure.
- **1936:** In 1936, Alan Turing gave a theory that how a machine can determine and execute a set of instructions.

The era of stored program computers:

- **1940:** In 1940, the first manually operated computer, "ENIAC" was invented, which was the first electronic general-purpose computer. After that stored program computer such as EDSAC in 1949 and EDVAC in 1951 were invented.
- **1943:** In 1943, a human neural network was modeled with an electrical circuit. In 1950, the scientists started applying their idea to work and analyzed how human neurons might work.

Computer machinery and intelligence:

- **1950:** In 1950, Alan Turing published a seminal paper, "**Computer Machinery and Intelligence**," on the topic of artificial intelligence. **In his paper, he asked, "Can machines think?"**

Machine intelligence in Games:

- **1952:** Arthur Samuel, who was the pioneer of machine learning, created a program that helped an IBM computer to play a checkers game. It performed better more it played.
- **1959:** In 1959, the term "Machine Learning" was first coined by **Arthur Samuel**.

The first "AI" winter:

- The duration of 1974 to 1980 was the tough time for AI and ML researchers, and this duration was called as **AI winter**.
- In this duration, failure of machine translation occurred, and people had reduced their interest from AI, which led to reduced funding by the government to the researches.

Machine Learning from theory to reality

- **1959:** In 1959, the first neural network was applied to a real-world problem to remove echoes over phone lines using an adaptive filter.
- **1985:** In 1985, Terry Sejnowski and Charles Rosenberg invented a neural network **NETtalk**, which was able to teach itself how to correctly pronounce 20,000 words in one week.
- **1997:** The IBM's **Deep blue** intelligent computer won the chess game against the chess expert Garry Kasparov, and it became the first computer which had beaten a human chess expert.

Machine Learning at 21st century

- **2006:** In the year 2006, computer scientist Geoffrey Hinton has given a new name to neural net research as "**deep learning**," and nowadays, it has become one of the most trending technologies.
- **2012:** In 2012, Google created a deep neural network which learned to recognize the image of humans and cats in YouTube videos.
- **2014:** In 2014, the Chabot "**Eugen Goostman**" cleared the Turing Test. It was the first Chabot who convinced the 33% of human judges that it was not a machine.
- **2014: DeepFace** was a deep neural network created by Facebook, and they claimed that it could recognize a person with the same precision as a human can do.
- **2016: AlphaGo** beat the world's number second player **Lee sedol** at **Go game**. In 2017 it beat the number one player of this game **Ke Jie**.
- **2017:** In 2017, the Alphabet's Jigsaw team built an intelligent system that was able to learn the **online trolling**. It used to read millions of comments of different websites to learn to stop online trolling.

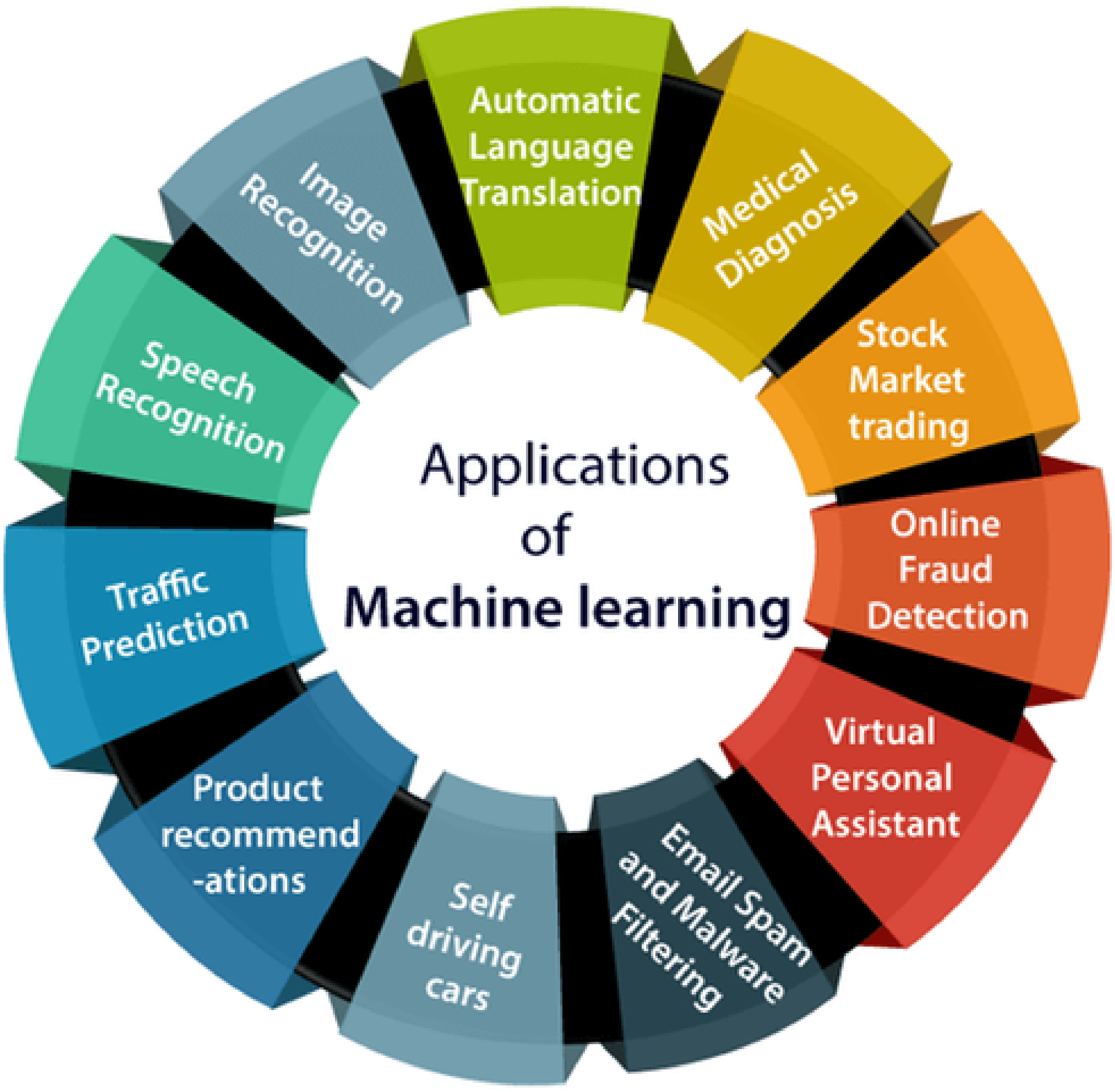
Machine Learning at present:

Now machine learning has got a great advancement in its research, and it is present everywhere around us, such as **self-driving cars**, **Amazon Alexa**, **Catboats**, **recommender system**, and many more. It includes **Supervised**, **unsupervised**, and **reinforcement learning with clustering**, **classification**, **decision tree**, **SVM algorithms**, etc.

Modern machine learning models can be used for making various predictions, including **weather prediction**, **disease prediction**, **stock market analysis**, etc.

Applications of Machine learning

Machine learning is a buzzword for today's technology, and it is growing very rapidly day by day. We are using machine learning in our daily life even without knowing it such as Google Maps, Google assistant, Alexa, etc. Below are some most trending real-world applications of Machine Learning:


Applications of Machine learning
Automatic Language Translation
Medical Diagnosis
Stock Market trading
Online Fraud Detection
Virtual Personal Assistant
Email Spam and Malware Filtering
Self driving cars
Product recommend -ations
Traffic Prediction
Speech Recognition
Image Recognition

1. Image Recognition:

Image recognition is one of the most common applications of machine learning. It is used to identify objects, persons, places, digital images, etc. The popular use case of image recognition and face detection is, Automatic friend tagging suggestion:

Facebook provides us a feature of auto friend tagging suggestion. Whenever we upload a photo with our Facebook friends, then we automatically get a tagging suggestion with name, and the technology behind this is machine learning's face detection and recognition algorithm.

It is based on the Facebook project named "Deep Face," which is responsible for face recognition and person identification in the picture.

2. Speech Recognition

While using Google, we get an option of "Search by voice," it comes under speech recognition, and it's a popular application of machine learning.

Speech recognition is a process of converting voice instructions into text, and it is also known as "Speech to text", or "Computer speech recognition." At present, machine learning algorithms are widely used by various applications of speech recognition. Google assistant, Siri, Cortana, and Alexa are using speech recognition technology to follow the voice instructions.

3. Traffic prediction:

If we want to visit a new place, we take help of Google Maps, which shows us the correct path with the shortest route and predicts the traffic conditions.

It predicts the traffic conditions such as whether traffic is cleared, slow-moving, or heavily congested with the help of two ways:

- Real Time location of the vehicle form Google Map app and sensors
- Average time has taken on past days at the same time.

Everyone who is using Google Map is helping this app to make it better. It takes information from the user and sends back to its database to improve the performance.

4. Product recommendations:

Machine learning is widely used by various e-commerce and entertainment companies such as Amazon, Netflix, etc., for product recommendation to the user. Whenever we search for some product on Amazon, then we started getting an advertisement for the same product while internet surfing on the same browser and this is because of machine learning.

Google understands the user interest using various machine learning algorithms and suggests the product as per customer interest.

As similar, when we use Netflix, we find some recommendations for entertainment series, movies, etc., and this is also done with the help of machine learning.

5. Self-driving cars:

One of the most exciting applications of machine learning is self-driving cars. Machine learning plays a significant role in self-driving cars. Tesla, the most popular car manufacturing company is working on self-driving car. It is using unsupervised learning method to train the car models to detect people and objects while driving.

6. Email Spam and Malware Filtering:

Whenever we receive a new email, it is filtered automatically as important, normal, and spam. We always receive an important mail in our inbox with the important symbol and spam emails in our spam box, and the technology behind this is Machine learning. Below are some spam filters used by Gmail:

- Content Filter
- Header filter
- General blacklists filter
- Rules-based filters
- Permission filters

Some machine learning algorithms such as Multi-Layer Perceptron, Decision tree, and Naïve Bayes classifier are used for email spam filtering and malware detection.

7. Virtual Personal Assistant:

We have various virtual personal assistants such as Google assistant, Alexa, Cortana, Siri. As the name suggests, they help us in finding the information using our voice instruction. These assistants can help us in various ways just by our voice instructions such as Play music, call someone, Open an email, Scheduling an appointment, etc.

These virtual assistants use machine learning algorithms as an important part.

These assistant record our voice instructions, send it over the server on a cloud, and decode it using ML algorithms and act accordingly.

8. Online Fraud Detection:

Machine learning is making our online transaction safe and secure by detecting fraud transaction. Whenever we perform some online transaction, there may be various ways that a fraudulent transaction can take place such as fake accounts, fake ids, and steal money in the middle of a transaction. So to detect this, Feed Forward Neural network helps us by checking whether it is a genuine transaction or a fraud transaction.

For each genuine transaction, the output is converted into some hash values, and these values become the input for the next round. For each genuine transaction, there is a specific pattern which gets change for the fraud transaction hence, it detects it and makes our online transactions more secure.

9. Stock Market trading:

Machine learning is widely used in stock market trading. In the stock market, there is always a risk of up and downs in shares, so for this machine learning's long short term memory neural network is used for the prediction of stock market trends.

10. Medical Diagnosis:

In medical science, machine learning is used for diseases diagnoses. With this, medical technology is growing very fast and able to build 3D models that can predict the exact position of lesions in the brain.

It helps in finding brain tumors and other brain-related diseases easily.

11. Automatic Language Translation:

Nowadays, if we visit a new place and we are not aware of the language then it is not a problem at all, as for this also machine learning helps us by converting the text into our known languages. Google's GNMT (Google Neural Machine Translation) provide this feature, which is a Neural Machine Learning that translates the text into our familiar language, and it called as automatic translation.

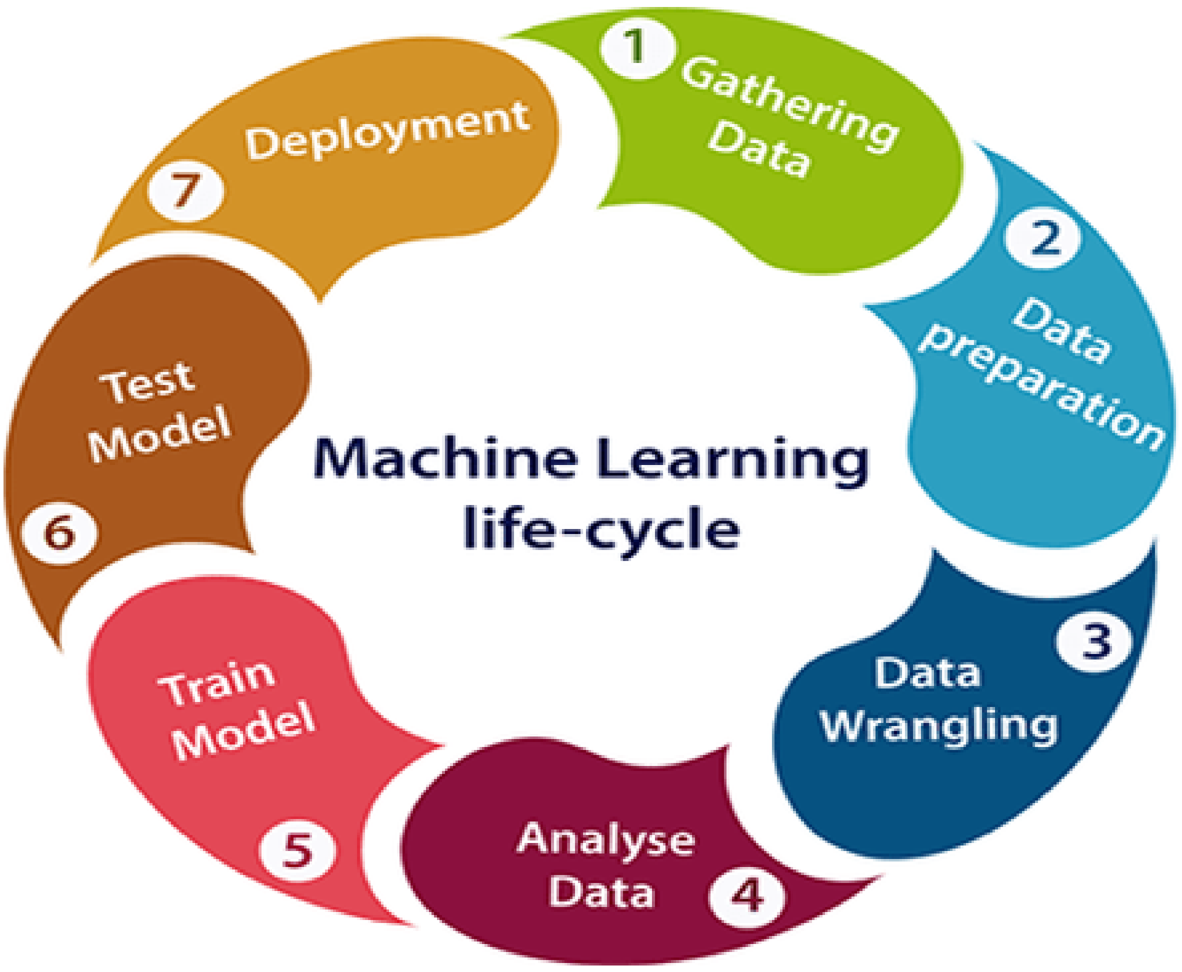
The technology behind the automatic translation is a sequence to sequence learning algorithm, which is used with image recognition and translates the text from one language to another language.

Machine learning Life cycle

Machine learning has given the computer systems the abilities to automatically learn without being explicitly programmed. But how does a machine learning system work? So, it can be described using the life cycle of machine learning. Machine learning life cycle is a cyclic process to build an efficient machine learning project. The main purpose of the life cycle is to find a solution to the problem or project.

Machine learning life cycle involves seven major steps, which are given below:

- Gathering Data
- Data preparation
- Data Wrangling
- Analyse Data
- Train the model
- Test the model
- Deployment



The most important thing in the complete process is to understand the problem and to know the purpose of the problem. Therefore, before starting the life cycle, we need to understand the problem because the good result depends on the better understanding of the problem.

In the complete life cycle process, to solve a problem, we create a machine learning system called "model", and this model is created by providing "training". But to train a model, we need data, hence, life cycle starts by collecting data.

1. Gathering Data:

Data Gathering is the first step of the machine learning life cycle. The goal of this step is to identify and obtain all data-related problems.

In this step, we need to identify the different data sources, as data can be collected from various sources such as files, database, internet, or mobile devices. It is one of the most important steps of the life cycle. The quantity and quality of the collected data will determine the efficiency of the output. The more will be the data, the more accurate will be the prediction.

This step includes the below tasks:

- Identify various data sources
- Collect data
- Integrate the data obtained from different sources

By performing the above task, we get a coherent set of data, also called as a dataset. It will be used in further steps.

2. Data preparation

After collecting the data, we need to prepare it for further steps. Data preparation is a step where we put our data into a suitable place and prepare it to use in our machine learning training.

In this step, first, we put all data together, and then randomize the ordering of data.

This step can be further divided into two processes:

- Data exploration:It is used to understand the nature of data that we have to work with. We need to understand the characteristics, format, and quality of data.
  A better understanding of data leads to an effective outcome. In this, we find Correlations, general trends, and outliers.

- Data pre-processing:Now the next step is
  preprocessing of data for its analysis.

3. Data Wrangling

Data wrangling is the process of cleaning and converting raw data into a useable format. It is the process of cleaning the data, selecting the variable to use, and transforming the data in a proper format to make it more suitable for analysis in the next step. It is one of the most important steps of the complete process. Cleaning of data is required to address the quality issues.

It is not necessary that data we have collected is always of our use as some of the data may not be useful. In real-world applications, collected data may have various issues, including:

- Missing Values
- Duplicate data
- Invalid data
- Noise

So, we use various filtering techniques to clean the data.

It is mandatory to detect and remove the above issues because it can negatively affect the quality of the outcome.

4. Data Analysis

Now the cleaned and prepared data is passed on to the analysis step. This step involves:

- Selection of analytical techniques
- Building models
- Review the result

The aim of this step is to build a machine learning model to analyze the data using various analytical techniques and review the outcome. It starts with the determination of the type of the problems, where we select the machine learning techniques such as Classification, Regression, Cluster analysis, Association, etc. then build the model using prepared data, and evaluate the model.

Hence, in this step, we take the data and use machine learning algorithms to build the model.

5. Train Model

Now the next step is to train the model, in this step we train our model to improve its performance for better outcome of the problem.

We use datasets to train the model using various machine learning algorithms. Training a model is required so that it can understand the various patterns, rules, and, features.

6. Test Model

Once our machine learning model has been trained on a given dataset, then we test the model. In this step, we check for the accuracy of our model by providing a test dataset to it.

Testing the model determines the percentage accuracy of the model as per the requirement of project or problem.

7. Deployment

The last step of machine learning life cycle is deployment, where we deploy the model in the real-world system.

If the above-prepared model is producing an accurate result as per our requirement with acceptable speed, then we deploy the model in the real system. But before deploying the project, we will check whether it is improving its performance using available data or not. The deployment phase is similar to making the final report for a project.

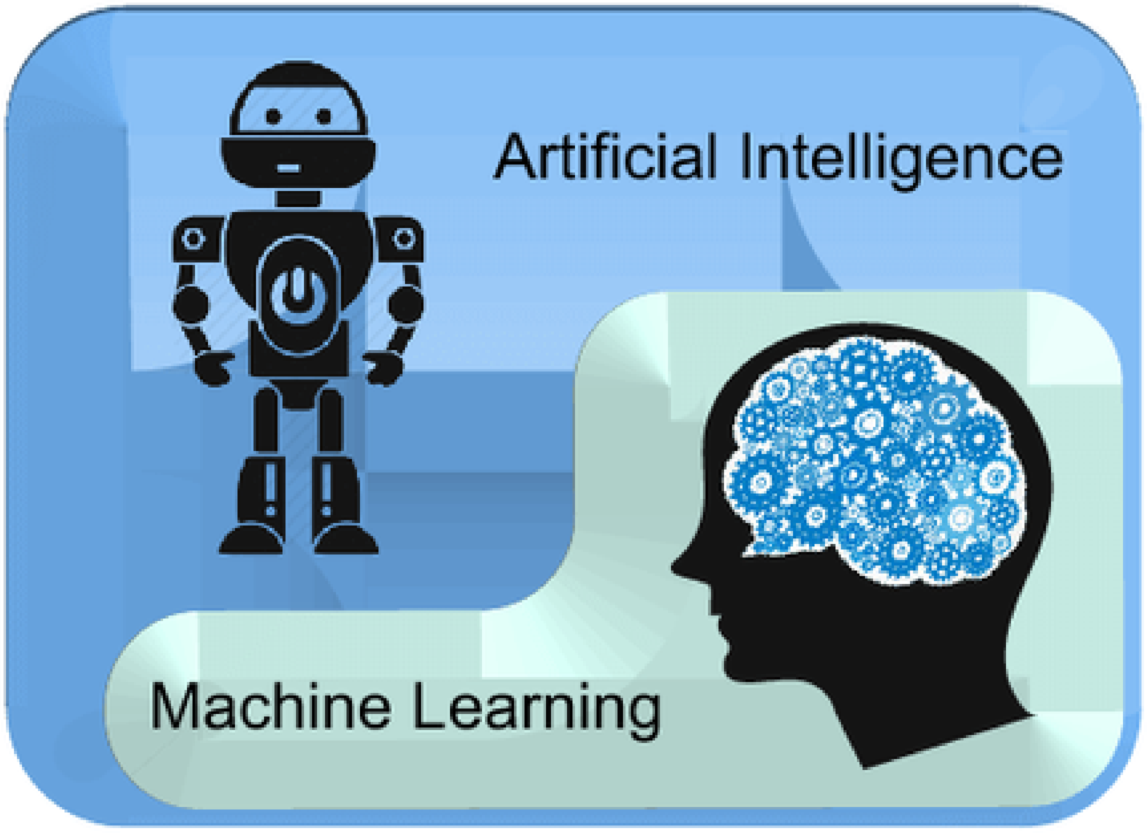
Difference between Artificial intelligence and Machine learning

Artificial intelligence and machine learning are the part of computer science that are correlated with each other. These two technologies are the most trending technologies which are used for creating intelligent systems.

Although these are two related technologies and sometimes people use them as a synonym for each other, but still both are the two different terms in various cases.

On a broad level, we can differentiate both AI and ML as:

AI is a bigger concept to create intelligent machines that can simulate human thinking capability and behavior, whereas, machine learning is an application or subset of AI that allows machines to learn from data without being programmed explicitly.



Below are some main differences between AI and machine learning along with the overview of Artificial intelligence and machine learning.

Artificial Intelligence

Artificial intelligence is a field of computer science which makes a computer system that can mimic human intelligence. It is comprised of two words "Artificial" and "intelligence", which means "a human-made thinking power." Hence we can define it as,

Artificial intelligence is a technology using which we can create intelligent systems that can simulate human intelligence.

The Artificial intelligence system does not require to be pre-programmed, instead of that, they use such algorithms which can work with their own intelligence. It involves machine learning algorithms such as Reinforcement learning algorithm and deep learning neural networks. AI is being used in multiple places such as Siri, Google?s AlphaGo, AI in Chess playing, etc.

Based on capabilities, AI can be classified into three types:

- Weak AI
- General AI
- Strong AI

Currently, we are working with weak AI and general AI. The future of AI is Strong AI for which it is said that it will be intelligent than humans.

Machine learning

Machine learning is about extracting knowledge from the data. It can be defined as, Machine learning is a subfield of artificial intelligence, which enables machines to learn from past data or experiences without being explicitly programmed.

Machine learning enables a computer system to make predictions or take some decisions using historical data without being explicitly programmed. Machine learning uses a massive amount of structured and semi-structured data so that a machine learning model can generate accurate result or give predictions based on that data.

Machine learning works on algorithm which learn by it?s own using historical data. It works only for specific domains such as if we are creating a machine learning model to detect pictures of dogs, it will only give result for dog images, but if we provide a new data like cat image then it will become unresponsive. Machine learning is being used in various places such as for online recommender system, for Google search algorithms, Email spam filter, Facebook Auto friend tagging suggestion, etc.

It can be divided into three types:

- Supervised learning
- Reinforcement learning
- Unsupervised learning

Key differences between Artificial Intelligence (AI) and Machine learning (ML):

Artificial Intelligence	Machine learning
Artificial intelligence is a technology which enables a machine to simulate human behavior.	Machine learning is a subset of AI which allows a machine to automatically learn from past data without programming explicitly.
The goal of AI is to make a smart computer system like humans to solve complex problems.	The goal of ML is to allow machines to learn from data so that they can give accurate output.
In AI, we make intelligent systems to perform any task like a human.	In ML, we teach machines with data to perform a particular task and give an accurate result.
Machine learning and deep learning are the two main subsets of AI.	Deep learning is a main subset of machine learning.
AI has a very wide range of scope.	Machine learning has a limited scope.
AI is working to create an intelligent system which can perform various complex tasks.	Machine learning is working to create machines that can perform only those specific tasks for which they are trained.
AI system is concerned about maximizing the chances of success.	Machine learning is mainly concerned about accuracy and patterns.
The main applications of AI are Siri, customer support using catboats, Expert System, Online game playing, intelligent humanoid robot, etc.	The main applications of machine learning are Online recommender system, Google search algorithms, Facebook auto friend tagging suggestions, etc.
On the basis of capabilities, AI can be divided into three types, which are, Weak AI, General AI, and Strong AI.	Machine learning can also be divided into mainly three types that are Supervised learning, Unsupervised learning, and Reinforcement learning.
It includes learning, reasoning, and self-correction.	It includes learning and self-correction when introduced with new data.
AI completely deals with Structured, semi-structured, and unstructured data.	Machine learning deals with Structured and semi-structured data.

How to get datasets for Machine Learning

The key to success in the field of machine learning or to become a great data scientist is to practice with different types of datasets. But discovering a suitable dataset for each kind of machine learning project is a difficult task. So, in this topic, we will provide the detail of the sources from where you can easily get the dataset according to your project.

Before knowing the sources of the machine learning dataset, let's discuss datasets.

What is a dataset?

A dataset is a collection of data in which data is arranged in some order. A dataset can contain any data from a series of an array to a database table. Below table shows an example of the dataset:

Country	**Age**	**Salary**	**Purchased**
India	38	48000	No
France	43	45000	Yes
Germany	30	54000	No
France	48	65000	No
Germany	40		Yes
India	35	58000	Yes

A tabular dataset can be understood as a database table or matrix, where each column corresponds to a particular variable, and each row corresponds to the fields of the dataset. The most supported file type for a tabular dataset is "Comma Separated File," or CSV. But to store a "tree-like data," we can use the JSON file more efficiently.

Types of data in datasets

- Numerical data:Such as house price, temperature, etc.
- Categorical data:Such as Yes/No, True/False, Blue/green, etc.
- Ordinal data:These data are similar to categorical data but can be measured on the basis of comparison.

Note: A real-world dataset is of huge size, which is difficult to manage and process at the initial level. Therefore, to practice machine learning algorithms, we can use any dummy dataset.

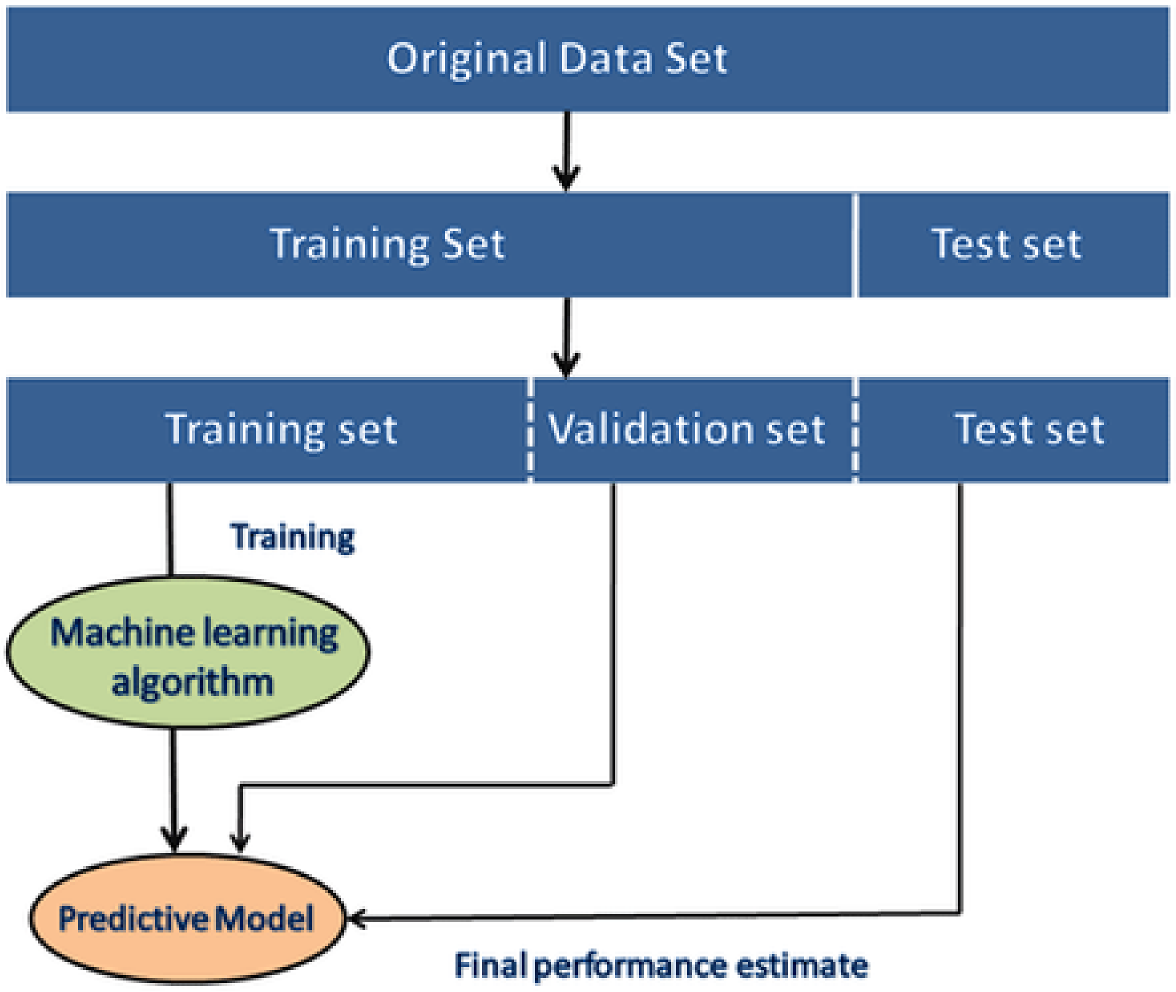
Need of Dataset

To work with machine learning projects, we need a huge amount of data, because, without the data, one cannot train ML/AI models. Collecting and preparing the dataset is one of the most crucial parts while creating an ML/AI project.

The technology applied behind any ML projects cannot work properly if the dataset is not well prepared and pre-processed.

During the development of the ML project, the developers completely rely on the datasets. In building ML applications, datasets are divided into two parts:

- Training dataset:
- Test Dataset

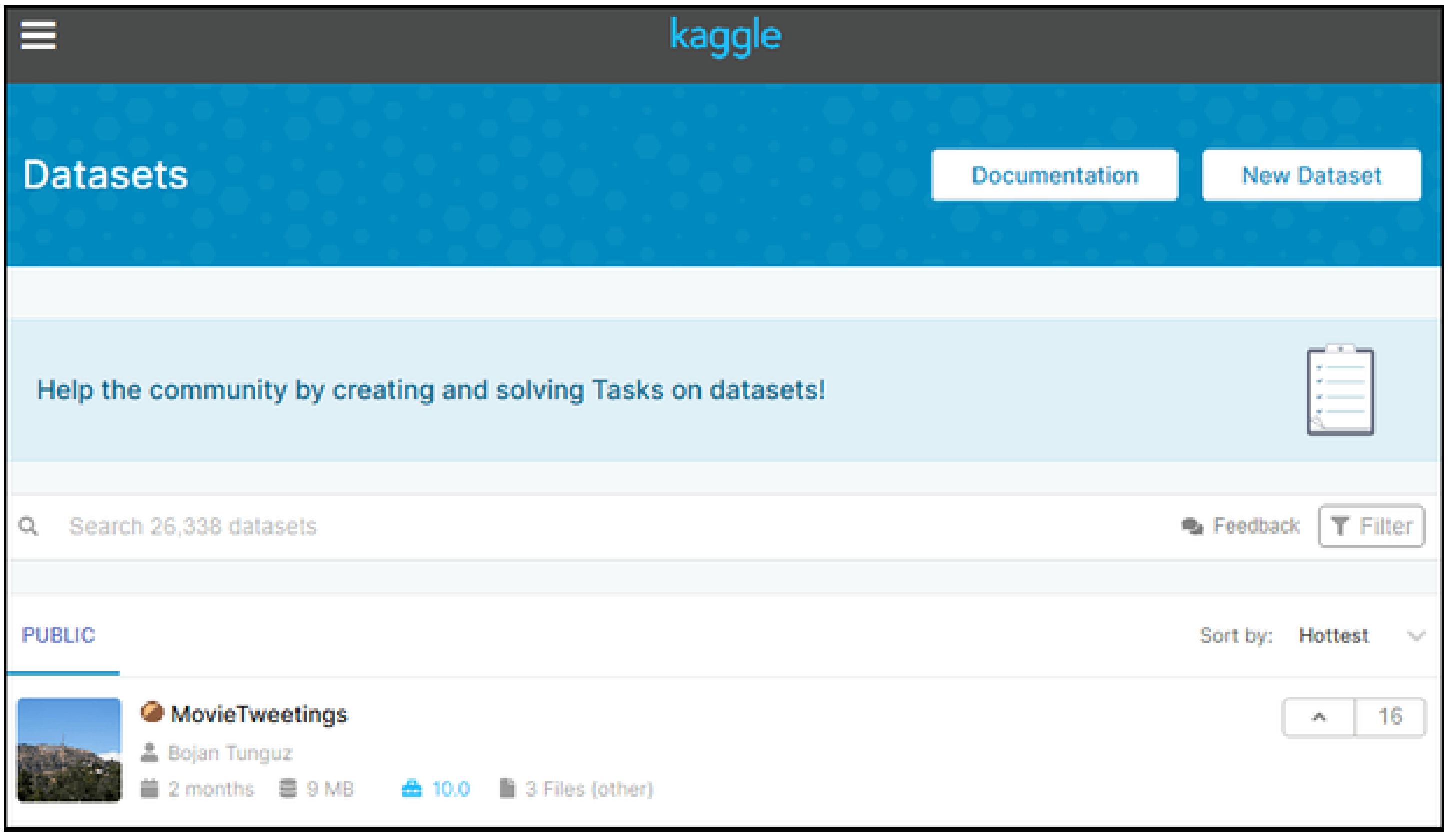


Note: The datasets are of large size, so to download these datasets, you must have fast internet on your computer.

Popular sources for Machine Learning datasets

Below is the list of datasets which are freely available for the public to work on it:

1. Kaggle Datasets

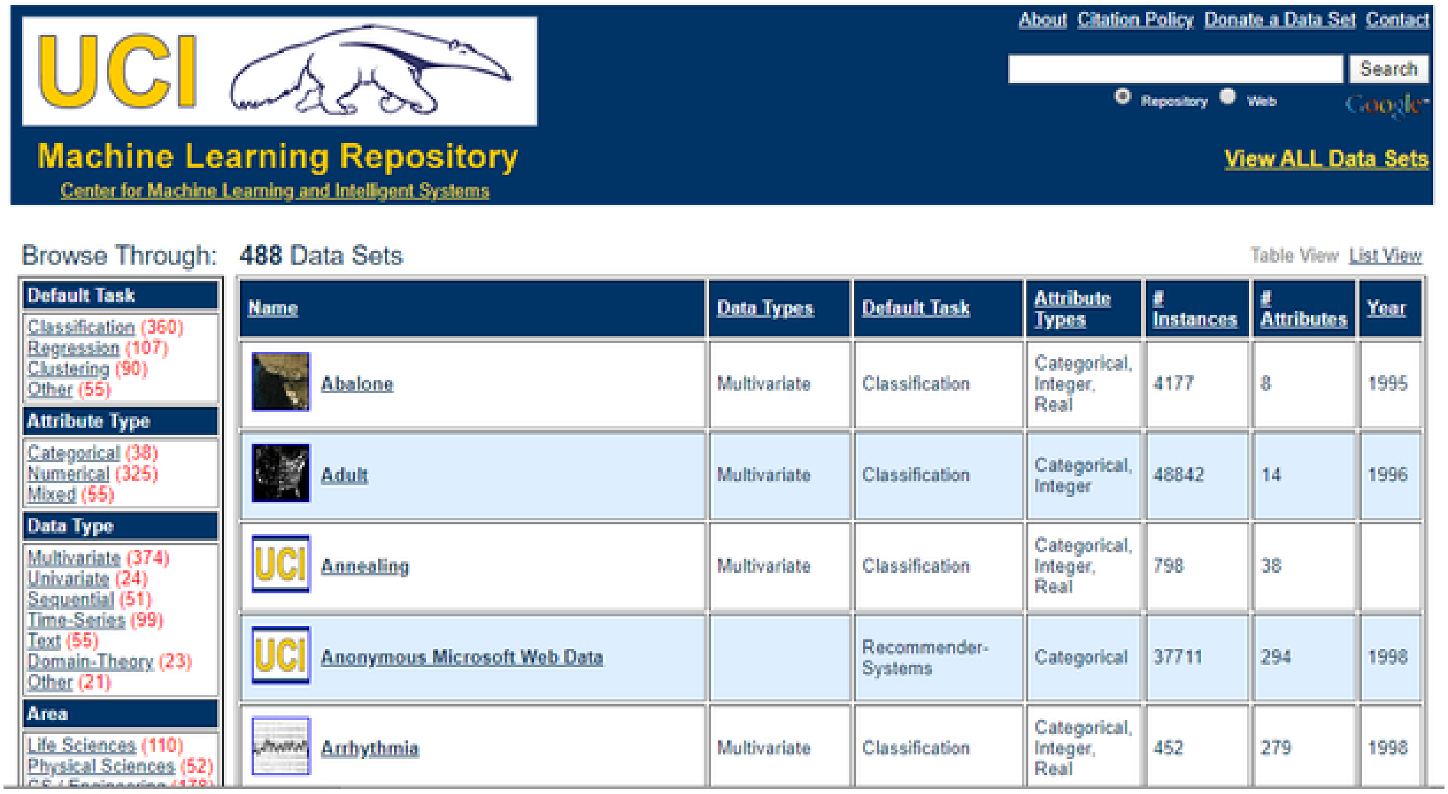


Kaggle is one of the best sources for providing datasets for Data Scientists and Machine Learners. It allows users to find, download, and publish datasets in an easy way. It also provides the opportunity to work with other machine learning engineers and solve difficult Data Science related tasks.

Kaggle provides a high-quality dataset in different formats that we can easily find and download.

The link for the Kaggle dataset is https://www.kaggle.com/datasets.

2. UCI Machine Learning Repository


UCI Machine Learning Repository

Center for Machine Learning and Intelligent Systems

About Citation Policy Donate a Data Set Contact

Search

Repository Web

View ALL Data Sets

Browse Through: 488 Data Sets

Table View List View

Default Task
Classification (360)
Regression (107)
Clustering (90)
Other (55)

Attribute Type
Categorical (38)
Numerical (325)
Mixed (55)

Data Type
Multivariate (374)
Univariate (24)
Sequential (51)
Time-Series (99)
Text (55)
Domain-Theory (23)
Other (21)

Area
Life Sciences (110)
Physical Sciences (52)

Name	Data Types	Default Task	Attribute Types	# Instances	# Attributes	Year
Abalone	Multivariate	Classification	Categorical, Integer, Real	4177	8	1995
Adult	Multivariate	Classification	Categorical, Integer	48842	14	1996
Annealing	Multivariate	Classification	Categorical, Integer, Real	798	38	
Anonymous Microsoft Web Data		Recommender-Systems	Categorical	37711	294	1998
Arrhythmia	Multivariate	Classification	Categorical, Integer, Real	452	279	1998

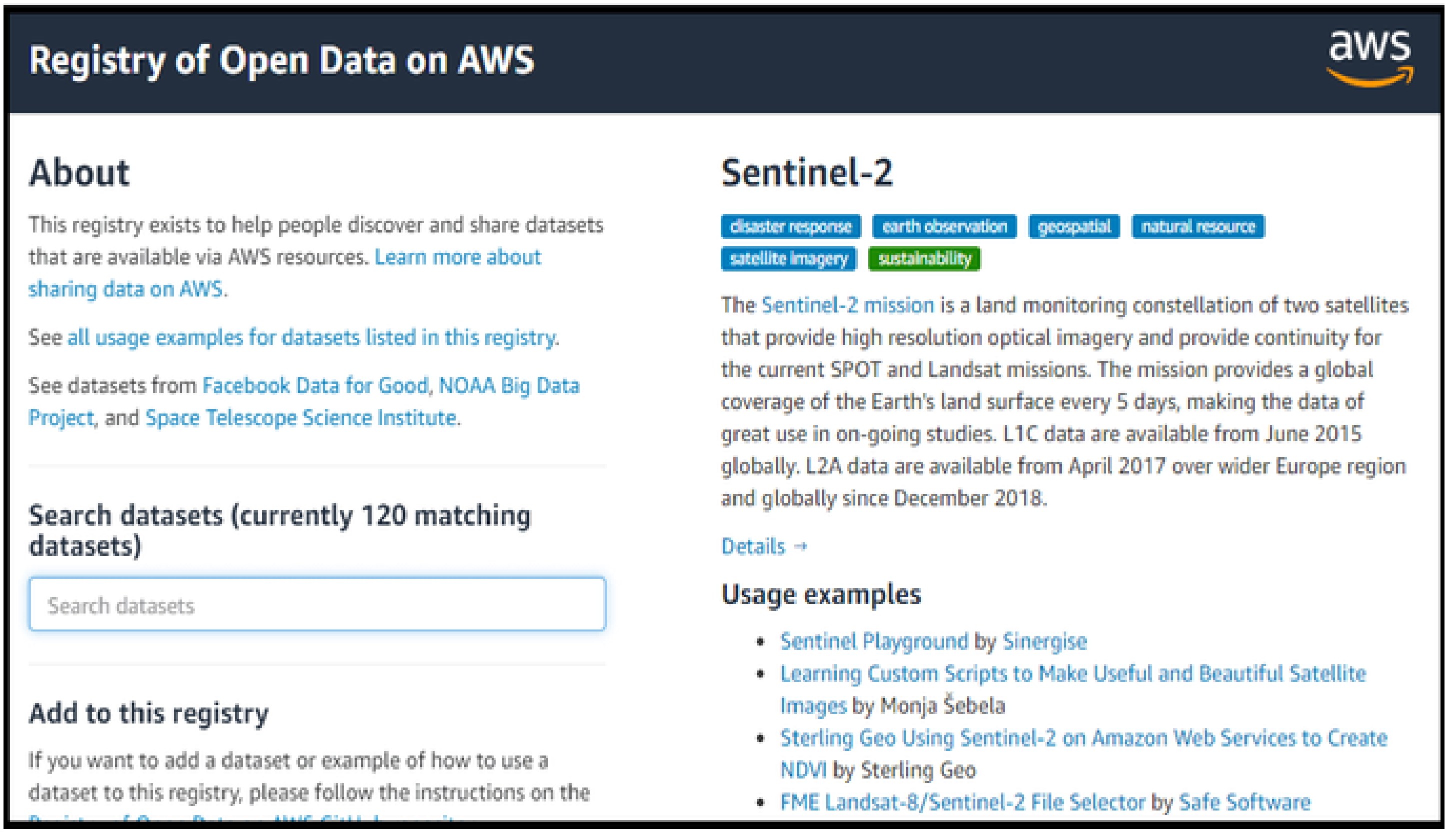
UCI Machine learning repository is one of the great sources of machine learning datasets. This repository contains databases, domain theories, and data generators that are widely used by the machine learning community for the analysis of ML algorithms.

Since the year 1987, it has been widely used by students, professors, researchers as a primary source of machine learning dataset.

It classifies the datasets as per the problems and tasks of machine learning such as Regression, Classification, Clustering, etc. It also contains some of the popular datasets such as the Iris dataset, Car Evaluation dataset, Poker Hand dataset, etc.

The link for the UCI machine learning repository is https://archive.ics.uci.edu/ml/index.php.

3. Datasets via AWS


Registry of Open Data on AWS

About

This registry exists to help people discover and share datasets that are available via AWS resources. Learn more about sharing data on AWS.

See all usage examples for datasets listed in this registry.

See datasets from Facebook Data for Good, NOAA Big Data Project, and Space Telescope Science Institute.

Search datasets (currently 120 matching datasets)

Search datasets

Add to this registry

If you want to add a dataset or example of how to use a dataset to this registry, please follow the instructions on the

Sentinel-2

disaster response | earth observation | geospatial | natural resource | satellite imagery | sustainability

The Sentinel-2 mission is a land monitoring constellation of two satellites that provide high resolution optical imagery and provide continuity for the current SPOT and Landsat missions. The mission provides a global coverage of the Earth's land surface every 5 days, making the data of great use in on-going studies. L1C data are available from June 2015 globally. L2A data are available from April 2017 over wider Europe region and globally since December 2018.

Details →

Usage examples

- Sentinel Playground by Sinergise
- Learning Custom Scripts to Make Useful and Beautiful Satellite Images by Monja Šebela
- Sterling Geo Using Sentinel-2 on Amazon Web Services to Create NDVI by Sterling Geo
- FME Landsat-8/Sentinel-2 File Selector by Safe Software

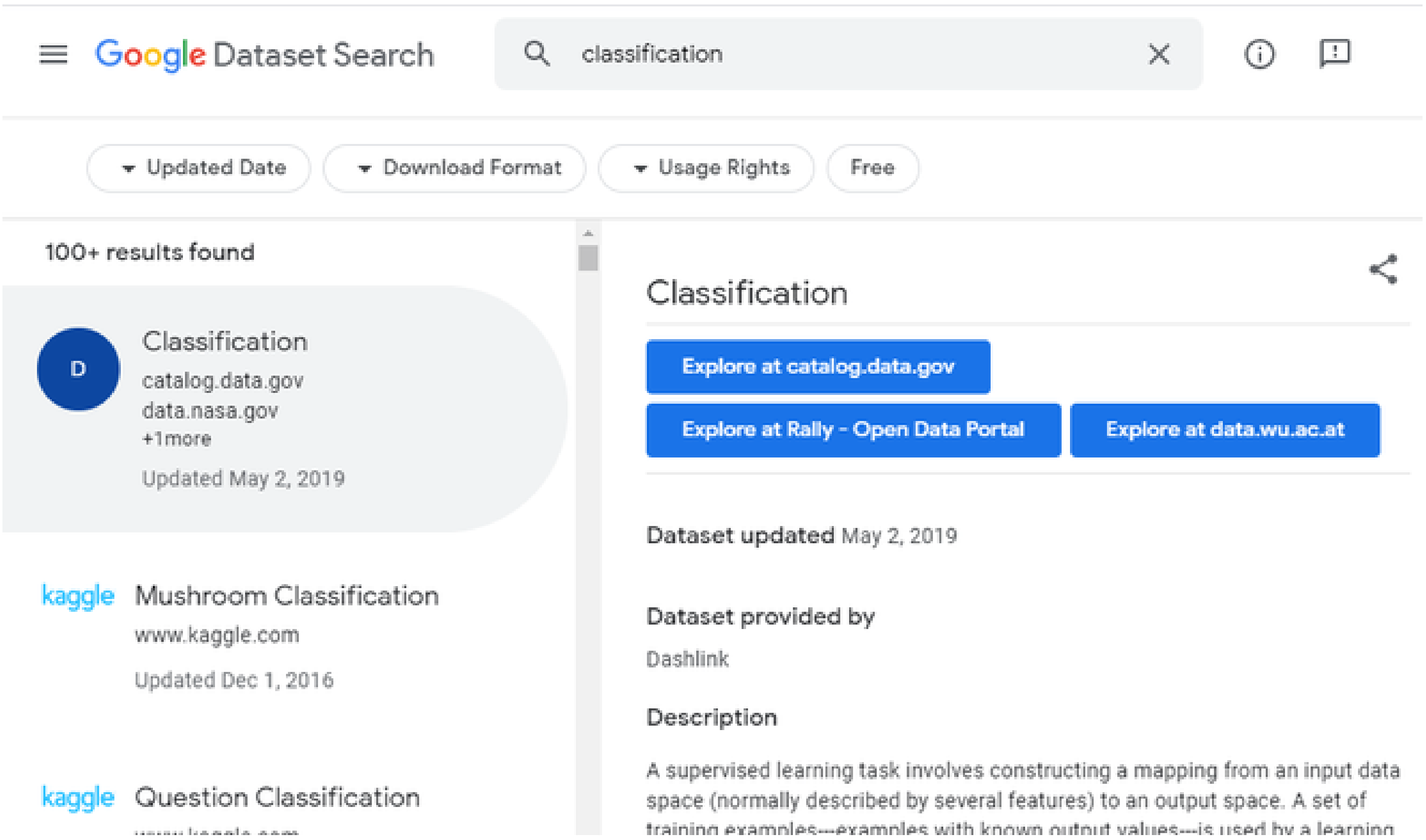
We can search, download, access, and share the datasets that are publicly available via AWS resources. These datasets can be accessed through AWS resources but provided and maintained by different government organizations, researches, businesses, or individuals.

Anyone can analyze and build various services using shared data via AWS resources. The shared dataset on cloud helps users to spend more time on data analysis rather than on acquisitions of data.

This source provides the various types of datasets with examples and ways to use the dataset. It also provides the search box using which we can search for the required dataset. Anyone can add any dataset or example to the Registry of Open Data on AWS.

The link for the resource is https://registry.opendata.aws/.

4. Google's Dataset Search Engine



100+ results found

Classification
catalog.data.gov
data.nasa.gov
+1more
Updated May 2, 2019

Mushroom Classification
www.kaggle.com
Updated Dec 1, 2016

Question Classification

Classification

Explore at catalog.data.gov
Explore at Rally - Open Data Portal
Explore at data.wu.ac.at

Dataset updated May 2, 2019

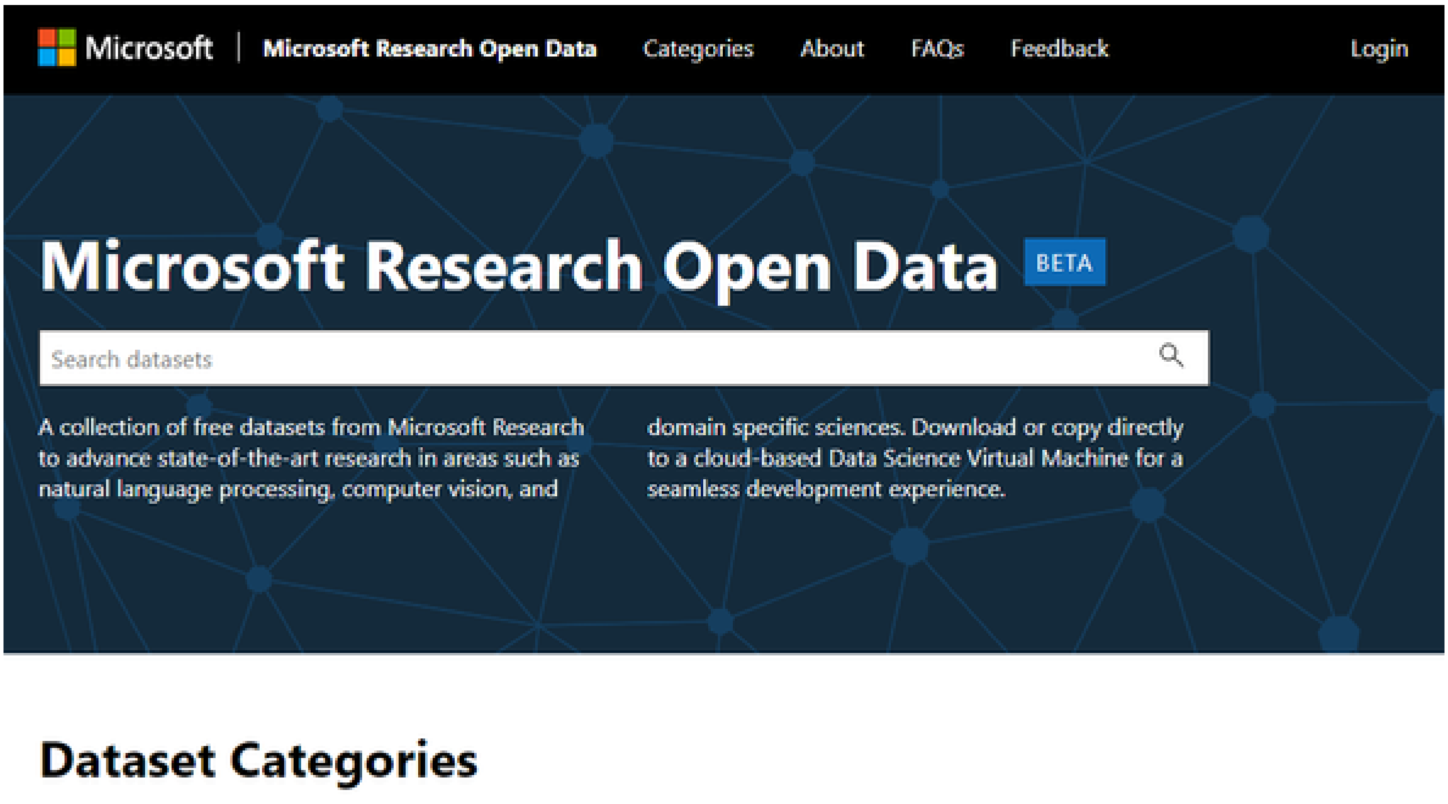
Dataset provided by
Dashlink

Description
A supervised learning task involves constructing a mapping from an input data space (normally described by several features) to an output space. A set of

Google dataset search engine is a search engine launched by Google on September 5, 2018. This source helps researchers to get online datasets that are freely available for use.

The link for the Google dataset search engine is https://toolbox.google.com/datasetsearch.

5. Microsoft Datasets



Microsoft Research Open Data BETA

A collection of free datasets from Microsoft Research to advance state-of-the-art research in areas such as natural language processing, computer vision, and domain specific sciences. Download or copy directly to a cloud-based Data Science Virtual Machine for a seamless development experience.

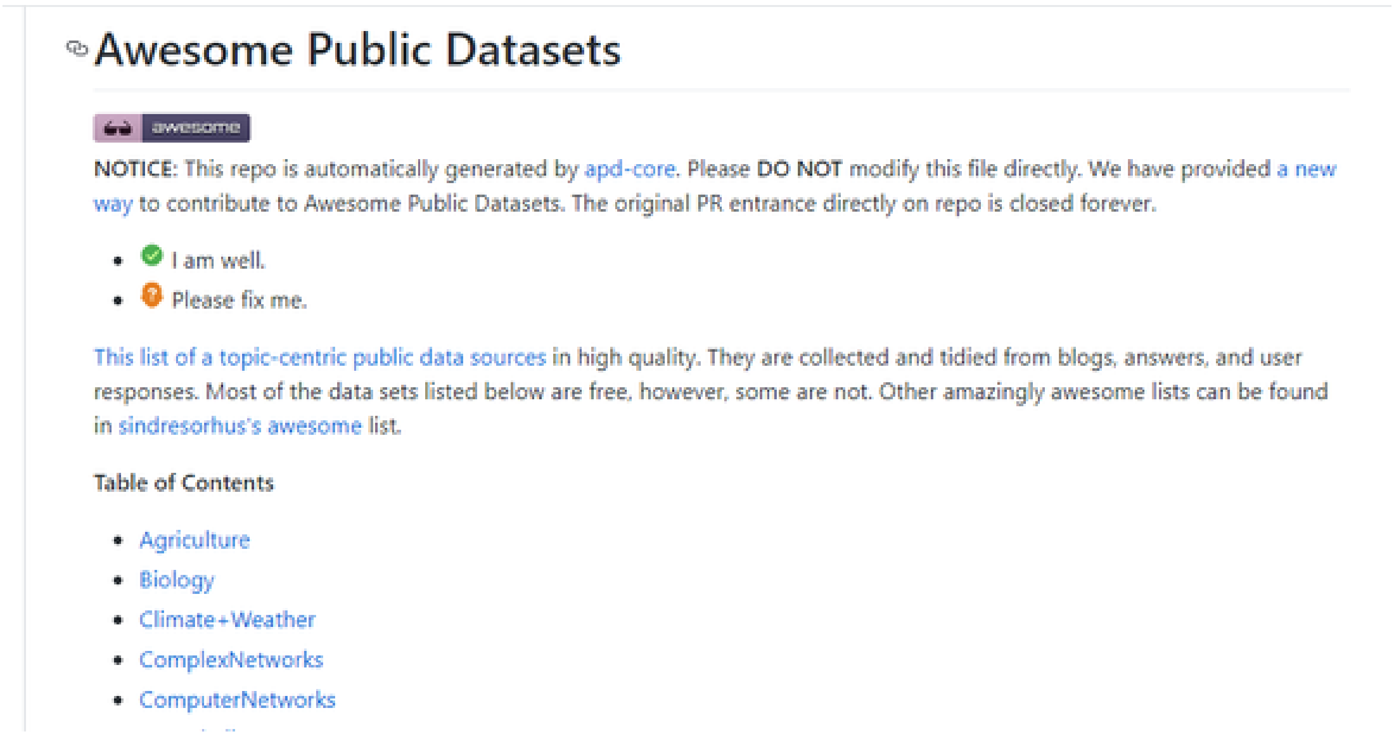
Dataset Categories

The Microsoft has launched the "Microsoft Research Open data" repository with the collection of free datasets in various areas such as natural language processing, computer vision, and domain-specific sciences.

Using this resource, we can download the datasets to use on the current device, or we can also directly use it on the cloud infrastructure.

The link to download or use the dataset from this resource is https://msropendata.com/.

6. Awesome Public Dataset Collection



Awesome Public Datasets

awesome

NOTICE: This repo is automatically generated by apd-core. Please DO NOT modify this file directly. We have provided a new way to contribute to Awesome Public Datasets. The original PR entrance directly on repo is closed forever.

- ✓ I am well.
- ? Please fix me.

This list of a topic-centric public data sources in high quality. They are collected and tidied from blogs, answers, and user responses. Most of the data sets listed below are free, however, some are not. Other amazingly awesome lists can be found in sindresorhus's awesome list.

Table of Contents

- Agriculture
- Biology
- Climate+Weather
- ComplexNetworks
- ComputerNetworks

Awesome public dataset collection provides high-quality datasets that are arranged in a well-organized manner within a list according to topics such as Agriculture, Biology, Climate, Complex networks, etc. Most of the datasets are available free, but some may not, so it is better to check the license before downloading the dataset.

The link to download the dataset from Awesome public dataset collection is https://github.com/awesomedata/awesome-public-datasets.

7. Government Datasets

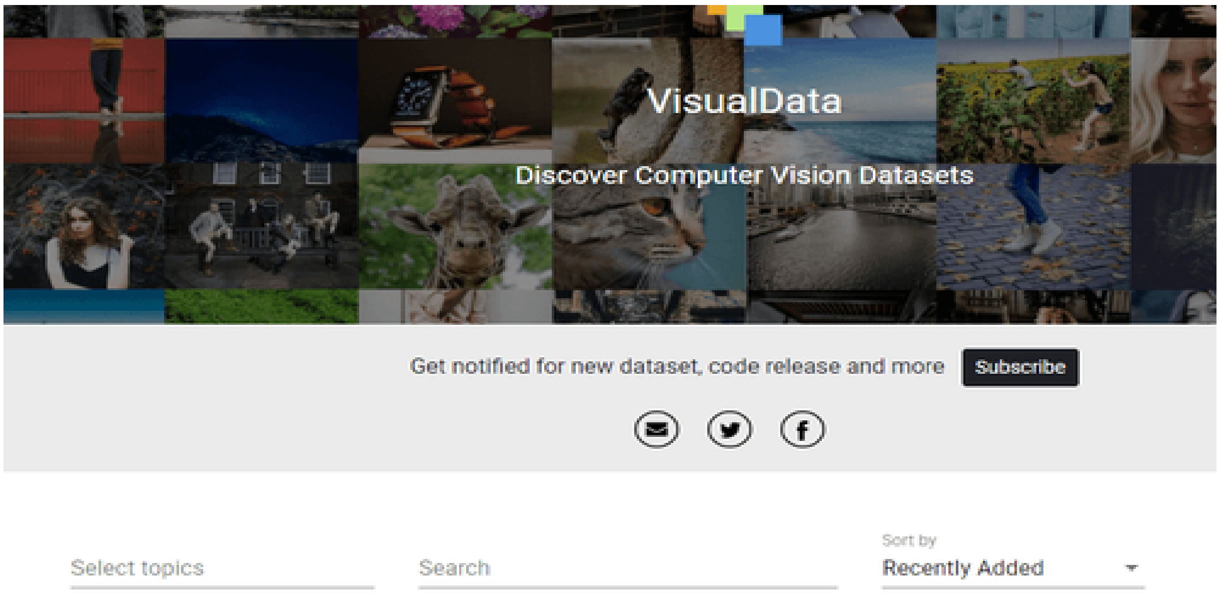
There are different sources to get government-related data. Various countries publish government data for public use collected by them from different departments.

The goal of providing these datasets is to increase transparency of government work among the people and to use the data in an innovative approach. Below are some links of government datasets:

- Indian Government dataset
- US Government Dataset
- Northern Ireland Public Sector Datasets

- European Union Open Data Portal

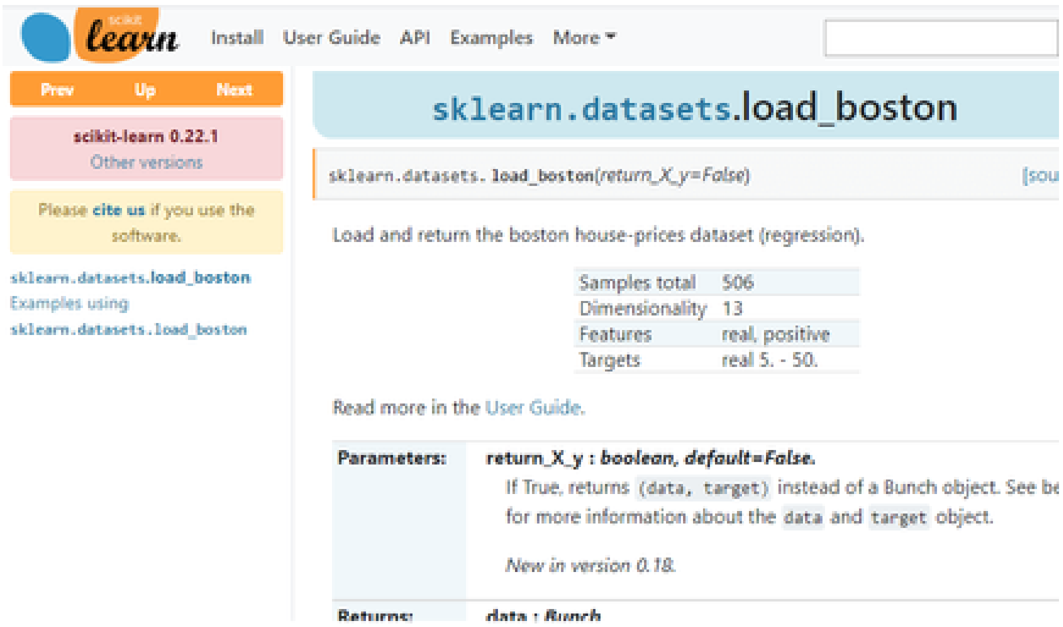
8. Computer Vision Datasets


VisualData
Discover Computer Vision Datasets
Get notified for new dataset, code release and more
Subscribe
Select topics
Search
Sort by
Recently Added

Visual data provides multiple numbers of the great dataset that are specific to computer visions such as Image Classification, Video classification, Image Segmentation, etc. Therefore, if you want to build a project on deep learning or image processing, then you can refer to this source.

The link for downloading the dataset from this source is https://www.visualdata.io/.

9. Scikit-learn dataset


scikit learn
Install
User Guide
API
Examples
More
Prev
Up
Next
scikit-learn 0.22.1
Other versions
Please cite us if you use the software.
sklearn.datasets.load_boston
Examples using sklearn.datasets.load_boston

sklearn.datasets.load_boston

`sklearn.datasets.load_boston(return_X_y=False)`

Load and return the boston house-prices dataset (regression).

Samples total	506
Dimensionality	13
Features	real, positive
Targets	real 5. - 50.

Read more in the User Guide.

Parameters: **return_X_y : boolean, default=False.**
If True, returns `(data, target)` instead of a Bunch object. See be
for more information about the `data` and `target` object.
New in version 0.18.

Returns: **data : Bunch**

Scikit-learn is a great source for machine learning enthusiasts. This source provides both toy and real-world datasets. These datasets can be obtained from sklearn.datasets package and using general dataset API.

The toy dataset available on scikit-learn can be loaded using some predefined functions such as, load_boston([return_X_y]), load_iris([return_X_y]), etc, rather than importing any file from external sources. But these datasets are not suitable for real-world projects.

The link to download datasets from this source is https://scikit-learn.org/stable/datasets/index.html.

Data Preprocessing in Machine learning

Data preprocessing is a process of preparing the raw data and making it suitable for a machine learning model. It is the first and crucial step while creating a machine learning model.

When creating a machine learning project, it is not always a case that we come across the clean and formatted data. And while doing any operation with data, it is mandatory to clean it and put in a formatted way. So for this, we use data preprocessing task.

Why do we need Data Preprocessing?

A real-world data generally contains noises, missing values, and maybe in an unusable format which cannot be directly used for machine learning models. Data preprocessing is required tasks for cleaning the data and making it suitable for a machine learning model which also increases the accuracy and efficiency of a machine learning model.

It involves below steps:

- Getting the dataset
- Importing libraries
- Importing datasets
- Finding Missing Data
- Encoding Categorical Data
- Splitting dataset into training and test set
- Feature scaling

1) Get the Dataset

To create a machine learning model, the first thing we required is a dataset as a machine learning model completely works on data. The collected data for a particular problem in a proper format is known as the dataset.

Dataset may be of different formats for different purposes, such as, if we want to create a machine learning model for business purpose, then dataset will be different with the dataset required for a liver patient. So each dataset is different from another dataset. To use the dataset in our code, we usually put it into a CSV file. However, sometimes, we may also need to use an HTML or xlsx file.

What is a CSV File?

CSV stands for "Comma-Separated Values" files; it is a file format which allows us to save the tabular data, such as spreadsheets. It is useful for huge datasets and can use these datasets in programs.

Here we will use a demo dataset for data preprocessing, and for practice, it can be downloaded from here, "https://www.superdatascience.com/pages/machine-learning. For real-world problems, we can download datasets online from various sources such as https://www.kaggle.com/uciml/datasets, https://archive.ics.uci.edu/ml/index.php etc.

We can also create our dataset by gathering data using various API with Python and put that data into a .csv file.

2) Importing Libraries

In order to perform data preprocessing using Python, we need to import some predefined Python libraries. These libraries are used to perform some specific jobs. There are three specific libraries that we will use for data preprocessing, which are:

Numpy: Numpy Python library is used for including any type of mathematical operation in the code. It is the fundamental package for scientific calculation in Python. It also supports to add large, multidimensional arrays and matrices. So, in Python, we can import it as:

1. import numpy as nm

Here we have used nm, which is a short name for Numpy, and it will be used in the whole program.

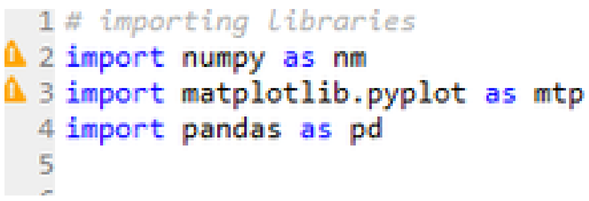
Matplotlib: The second library is matplotlib, which is a Python 2D plotting library, and with this library, we need to import a sub-library pyplot. This library is used to plot any type of charts in Python for the code. It will be imported as below:

1. import matplotlib.pyplot as mpt

Here we have used mpt as a short name for this library.

Pandas: The last library is the Pandas library, which is one of the most famous Python libraries and used for importing and managing the datasets. It is an open-source data manipulation and analysis library. It will be imported as below:

Here, we have used pd as a short name for this library. Consider the below image:



```
# importing libraries
import numpy as nm
import matplotlib.pyplot as mtp
import pandas as pd
```

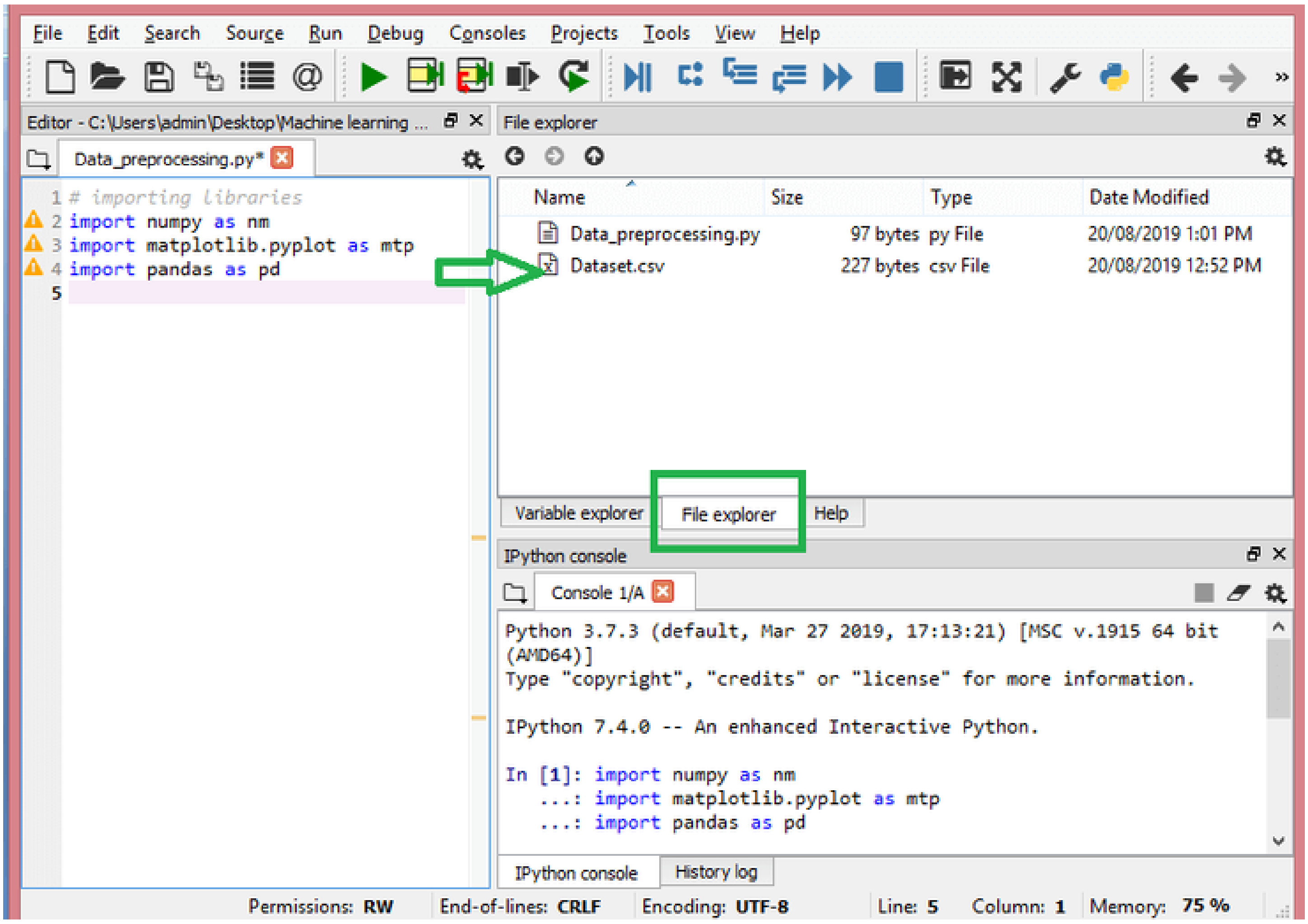
3) Importing the Datasets

Now we need to import the datasets which we have collected for our machine learning project. But before importing a dataset, we need to set the current directory as a working directory. To set a working directory in Spyder IDE, we need to follow the below steps:

1. Save your Python file in the directory which contains dataset.
2. Go to File explorer option in Spyder IDE, and select the required directory.
3. Click on F5 button or run option to execute the file.

Note: We can set any directory as a working directory, but it must contain the required dataset.

Here, in the below image, we can see the Python file along with required dataset. Now, the current folder is set as a working directory.



```
# importing libraries
import numpy as nm
import matplotlib.pyplot as mtp
import pandas as pd
```

Name	Size	Type	Date Modified
Data_preprocessing.py	97 bytes	py File	20/08/2019 1:01 PM
Dataset.csv	227 bytes	csv File	20/08/2019 12:52 PM

```
Python 3.7.3 (default, Mar 27 2019, 17:13:21) [MSC v.1915 64 bit
(AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 7.4.0 -- An enhanced Interactive Python.

In [1]: import numpy as nm
   ...: import matplotlib.pyplot as mtp
   ...: import pandas as pd
```

Permissions: RW End-of-lines: CRLF Encoding: UTF-8 Line: 5 Column: 1 Memory: 75 %

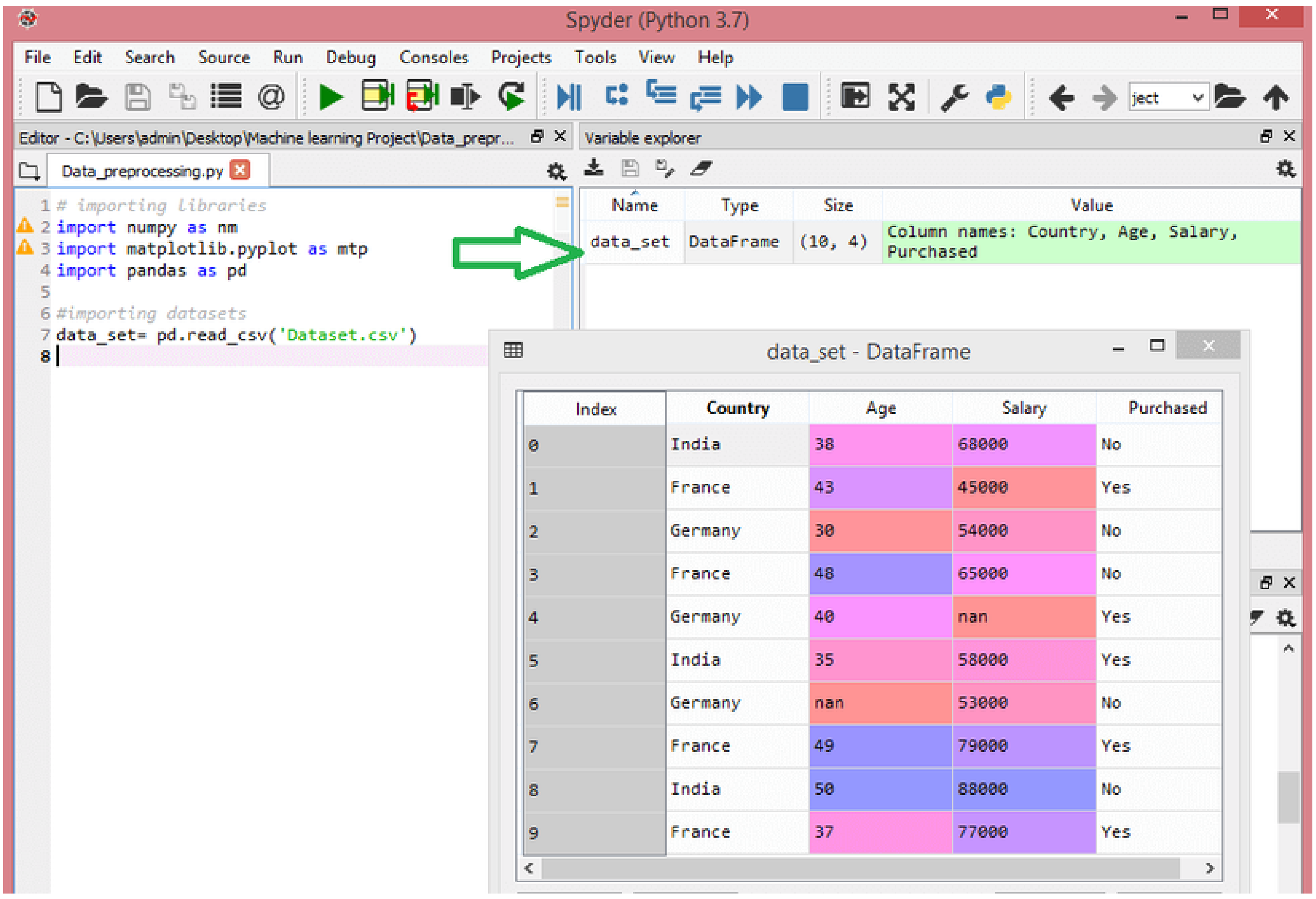
read_csv() function:

Now to import the dataset, we will use read_csv() function of pandas library, which is used to read a csv file and performs various operations on it. Using this function, we can read a csv file locally as well as through an URL.

We can use read_csv function as below:

```
1. data_set= pd.read_csv('Dataset.csv')
```

Here, data_set is a name of the variable to store our dataset, and inside the function, we have passed the name of our dataset. Once we execute the above line of code, it will successfully import the dataset in our code. We can also check the imported dataset by clicking on the section variable explorer, and then double click on data_set. Consider the below image:


Spyder (Python 3.7)
File Edit Search Source Run Debug Consoles Projects Tools View Help
Editor - C:\Users\admin\Desktop\Machine learning Project\Data_prepr...
Data_preprocessing.py
Variable explorer
data_set - DataFrame

```python
# importing libraries
import numpy as nm
import matplotlib.pyplot as mtp
import pandas as pd

#importing datasets
data_set= pd.read_csv('Dataset.csv')
```

Name	Type	Size	Value
data_set	DataFrame	(10, 4)	Column names: Country, Age, Salary, Purchased

Index	Country	Age	Salary	Purchased
0	India	38	68000	No
1	France	43	45000	Yes
2	Germany	30	54000	No
3	France	48	65000	No
4	Germany	40	nan	Yes
5	India	35	58000	Yes
6	Germany	nan	53000	No
7	France	49	79000	Yes
8	India	50	88000	No
9	France	37	77000	Yes

As in the above image, indexing is started from 0, which is the default indexing in Python. We can also change the format of our dataset by clicking on the format option.

Extracting dependent and independent variables:

In machine learning, it is important to distinguish the matrix of features (independent variables) and dependent variables from dataset. In our dataset, there are three independent variables that are Country, Age, and Salary, and one is a dependent variable which is Purchased.

Extracting independent variable:

To extract an independent variable, we will use iloc[ ] method of Pandas library. It is used to extract the required rows and columns from the dataset.

```
x= data_set.iloc[:,:-1].values
```

In the above code, the first colon(:) is used to take all the rows, and the second colon(:) is for all the columns. Here we have used :-1, because we don't want to take the last column as it contains the dependent variable. So by doing this, we will get the matrix of features.

By executing the above code, we will get output as:

```
[['India' 38.0 68000.0]
 ['France' 43.0 45000.0]
 ['Germany' 30.0 54000.0]
 ['France' 48.0 65000.0]
 ['Germany' 40.0 nan]
 ['India' 35.0 58000.0]
 ['Germany' nan 53000.0]
 ['France' 49.0 79000.0]
 ['India' 50.0 88000.0]
 ['France' 37.0 77000.0]]
```

As we can see in the above output, there are only three variables.

Extracting dependent variable:

To extract dependent variables, again, we will use Pandas .iloc[] method.

```
y= data_set.iloc[:,3].values
```

Here we have taken all the rows with the last column only. It will give the array of dependent variables.

By executing the above code, we will get output as:

Output:

```
array(['No', 'Yes', 'No', 'No', 'Yes', 'Yes', 'No', 'Yes', 'No', 'Yes'],

    dtype=object)
```

Note: If you are using Python language for machine learning, then extraction is mandatory, but for R language it is not required.

4) Handling Missing data:

The next step of data preprocessing is to handle missing data in the datasets. If our dataset contains some missing data, then it may create a huge problem for our machine learning model. Hence it is necessary to handle missing values present in the dataset.

Ways to handle missing data:

There are mainly two ways to handle missing data, which are:

By deleting the particular row: The first way is used to commonly deal with null values. In this way, we just delete the specific row or column which consists of null values. But this way is not so efficient and removing data may lead to loss of information which will not give the accurate output.

By calculating the mean: In this way, we will calculate the mean of that column or row which contains any missing value and will put it on the place of missing value. This strategy is useful for the features which have numeric data such as age, salary, year, etc. Here, we will use this approach.

To handle missing values, we will use Scikit-learn library in our code, which contains various libraries for building machine learning models. Here we will use Imputer class of sklearn.preprocessing library. Below is the code for it:

```
#handling missing data (Replacing missing data with the mean value)
from sklearn.preprocessing import Imputer
imputer= Imputer(missing_values ='NaN', strategy='mean', axis = 0)
#Fitting imputer object to the independent variables x.
```

```
imputerimputer= imputer.fit(x[:, 1:3])
#Replacing missing data with the calculated mean value
x[:, 1:3]= imputer.transform(x[:, 1:3])
```

Output:

```
array([['India', 38.0, 68000.0],

    ['France', 43.0, 45000.0],

    ['Germany', 30.0, 54000.0],

    ['France', 48.0, 65000.0],

    ['Germany', 40.0, 65222.2222222222],

    ['India', 35.0, 58000.0],

    ['Germany', 41.111111111111114, 53000.0],

    ['France', 49.0, 79000.0],

    ['India', 50.0, 88000.0],

    ['France', 37.0, 77000.0]], dtype=object
```

As we can see in the above output, the missing values have been replaced with the means of rest column values.

5) Encoding Categorical data:

Categorical data is data which has some categories such as, in our dataset; there are two categorical variable, Country, and Purchased.

Since machine learning model completely works on mathematics and numbers, but if our dataset would have a categorical variable, then it may create trouble while building the model. So it is necessary to encode these categorical variables into numbers.

For Country variable:

Firstly, we will convert the country variables into categorical data. So to do this, we will use LabelEncoder() class from preprocessing library.

```
#Catgorical data
#for Country Variable
from sklearn.preprocessing import LabelEncoder
label_encoder_x= LabelEncoder()
x[:, 0]= label_encoder_x.fit_transform(x[:, 0])
```

Output:

```
Out[15]:

 array([[2, 38.0, 68000.0],

      [0, 43.0, 45000.0],
    [1, 30.0, 54000.0],
    [0, 48.0, 65000.0],
    [1, 40.0, 65222.2222222222],
    [2, 35.0, 58000.0],
    [1, 41.1111111111114, 53000.0],
    [0, 49.0, 79000.0],
    [2, 50.0, 88000.0],
   [0, 37.0, 77000.0]], dtype=object)
```

Explanation:

In above code, we have imported LabelEncoder class of sklearn library. This class has successfully encoded the variables into digits.

But in our case, there are three country variables, and as we can see in the above output, these variables are encoded into 0, 1, and 2. By these values, the machine learning model may assume that there is some correlation between these variables which will produce the wrong output. So to remove this issue, we will use dummy encoding.

Dummy Variables:

Dummy variables are those variables which have values 0 or 1. The 1 value gives the presence of that variable in a particular column, and rest variables become 0. With dummy encoding, we will have a number of columns equal to the number of categories.

In our dataset, we have 3 categories so it will produce three columns having 0 and 1 values. For Dummy Encoding, we will use OneHotEncoder class of preprocessing library.

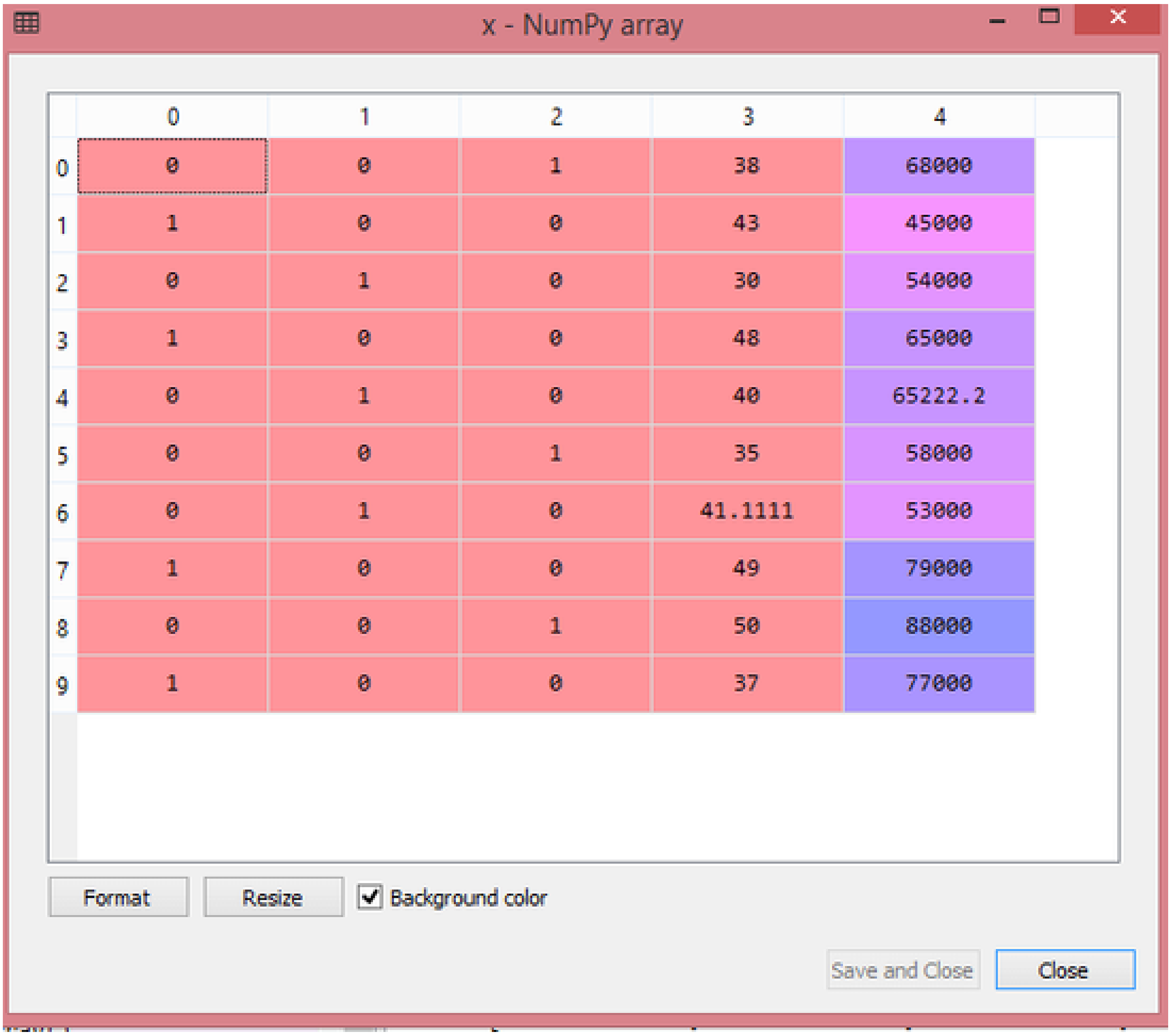
```
#for Country Variable
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
label_encoder_x= LabelEncoder()
x[:, 0]= label_encoder_x.fit_transform(x[:, 0])
#Encoding for dummy variables
onehot_encoder= OneHotEncoder(categorical_features= [0])
x= onehot_encoder.fit_transform(x).toarray()
```

Output:

```
array([[0.00000000e+00, 0.00000000e+00, 1.00000000e+00, 3.80000000e+01,
   6.80000000e+04],
  [1.00000000e+00, 0.00000000e+00, 0.00000000e+00, 4.30000000e+01,
   4.50000000e+04],
  [0.00000000e+00, 1.00000000e+00, 0.00000000e+00, 3.00000000e+01,
   5.40000000e+04],
  [1.00000000e+00, 0.00000000e+00, 0.00000000e+00, 4.80000000e+01,

   6.50000000e+04],

  [0.00000000e+00, 1.00000000e+00, 0.00000000e+00, 4.00000000e+01,

   6.52222222e+04],

  [0.00000000e+00, 0.00000000e+00, 1.00000000e+00, 3.50000000e+01,

   5.80000000e+04],

  [0.00000000e+00, 1.00000000e+00, 0.00000000e+00, 4.11111111e+01,

   5.30000000e+04],

  [1.00000000e+00, 0.00000000e+00, 0.00000000e+00, 4.90000000e+01,

   7.90000000e+04],

  [0.00000000e+00, 0.00000000e+00, 1.00000000e+00, 5.00000000e+01,

   8.80000000e+04],

  [1.00000000e+00, 0.00000000e+00, 0.00000000e+00, 3.70000000e+01,

   7.70000000e+04]])
```

As we can see in the above output, all the variables are encoded into numbers 0 and 1 and divided into three columns.

It can be seen more clearly in the variables explorer section, by clicking on x option as:


x - NumPy array

	0	1	2	3	4
0	0	0	1	38	68000
1	1	0	0	43	45000
2	0	1	0	30	54000
3	1	0	0	48	65000
4	0	1	0	40	65222.2
5	0	0	1	35	58000
6	0	1	0	41.1111	53000
7	1	0	0	49	79000
8	0	0	1	50	88000
9	1	0	0	37	77000

For Purchased Variable:

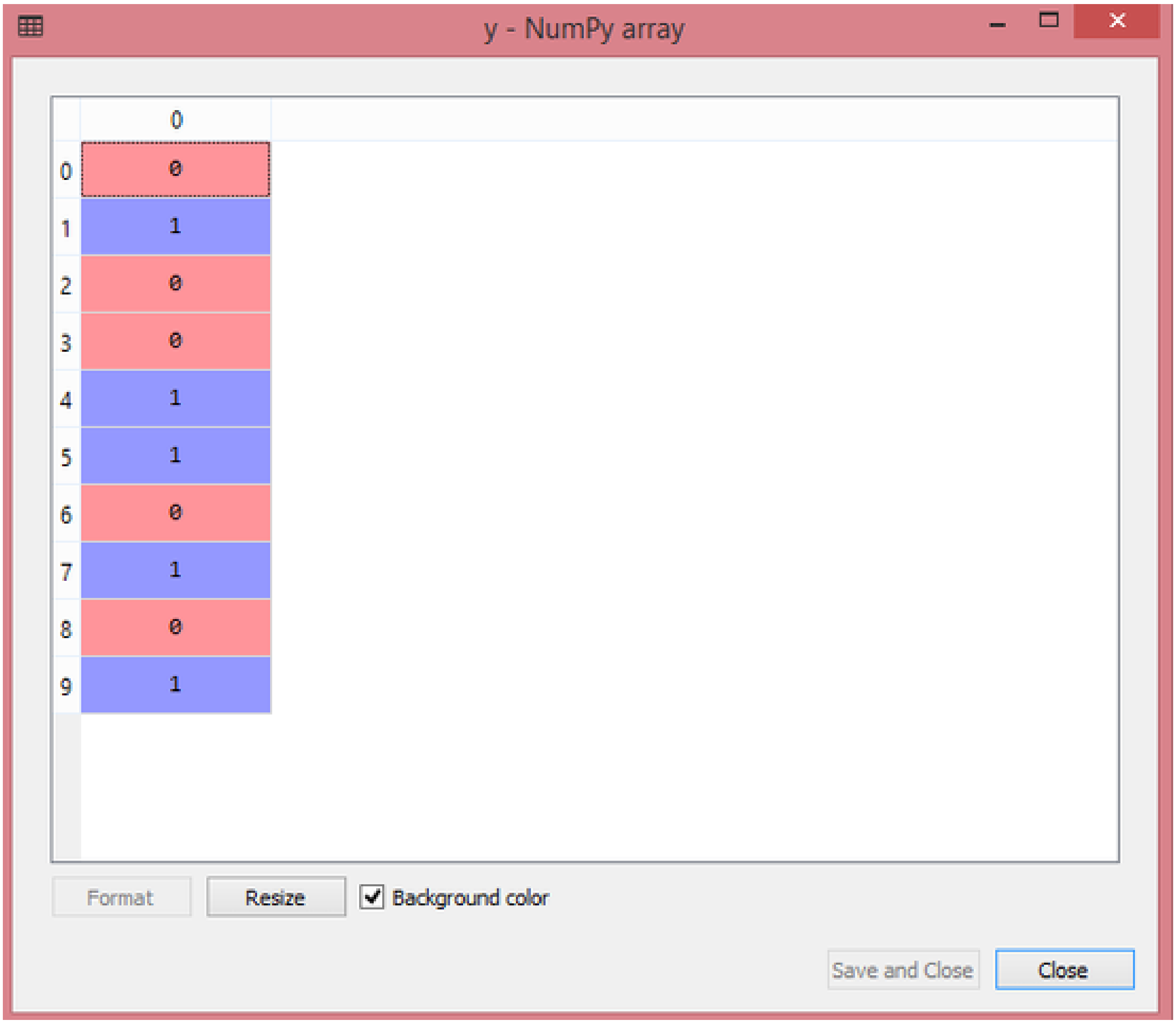
```
labelencoder_y= LabelEncoder()
y= labelencoder_y.fit_transform(y)
```

For the second categorical variable, we will only use labelencoder object of LableEncoder class. Here we are not using OneHotEncoder class because the purchased variable has only two categories yes or no, and which are automatically encoded into 0 and 1.

Output:

```
Out[17]: array([0, 1, 0, 0, 1, 1, 0, 1, 0, 1])
```

It can also be seen as:



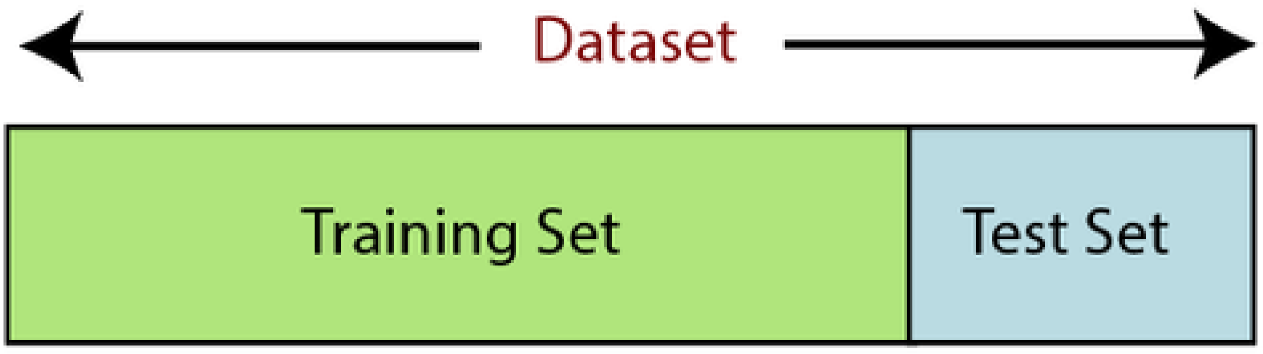
	0
0	0
1	1
2	0
3	0
4	1
5	1
6	0
7	1
8	0
9	1

6) Splitting the Dataset into the Training set and Test set

In machine learning data preprocessing, we divide our dataset into a training set and test set. This is one of the crucial steps of data preprocessing as by doing this, we can enhance the performance of our machine learning model.

Suppose, if we have given training to our machine learning model by a dataset and we test it by a completely different dataset. Then, it will create difficulties for our model to understand the correlations between the models.

If we train our model very well and its training accuracy is also very high, but we provide a new dataset to it, then it will decrease the performance. So we always try to make a machine learning model which performs well with the training set and also with the test dataset. Here, we can define these datasets as:



Training Set: A subset of dataset to train the machine learning model, and we already know the output.

Test set: A subset of dataset to test the machine learning model, and by using the test set, model predicts the output.

For splitting the dataset, we will use the below lines of code:

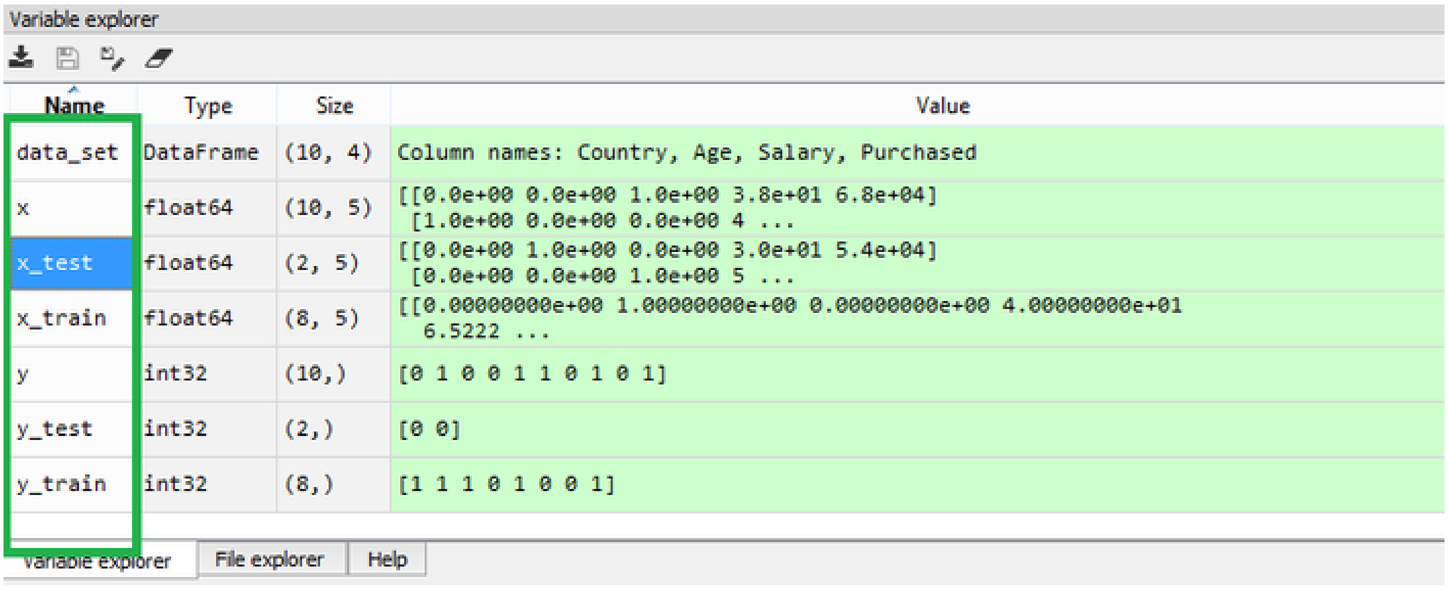
```
1. from sklearn.model_selection import train_test_split
2. x_train, x_test, y_train, y_test= train_test_split(x, y, test_size= 0.2, random_state=0)
```

Explanation:

- In the above code, the first line is used for splitting arrays of the dataset into random train and test subsets.
- In the second line, we have used four variables for our output that are
  - x_train: features for the training data
  - x_test: features for testing data
  - y_train: Dependent variables for training data
  - y_test: Independent variable for testing data
- In train_test_split() function, we have passed four parameters in which first two are for arrays of data, and test_size is for specifying the size of the test set. The test_size maybe .5, .3, or .2, which tells the dividing ratio of training and testing sets.
- The last parameter random_state is used to set a seed for a random generator so that you always get the same result, and the most used value for this is 42.

Output:

By executing the above code, we will get 4 different variables, which can be seen under the variable explorer section.


Variable explorer

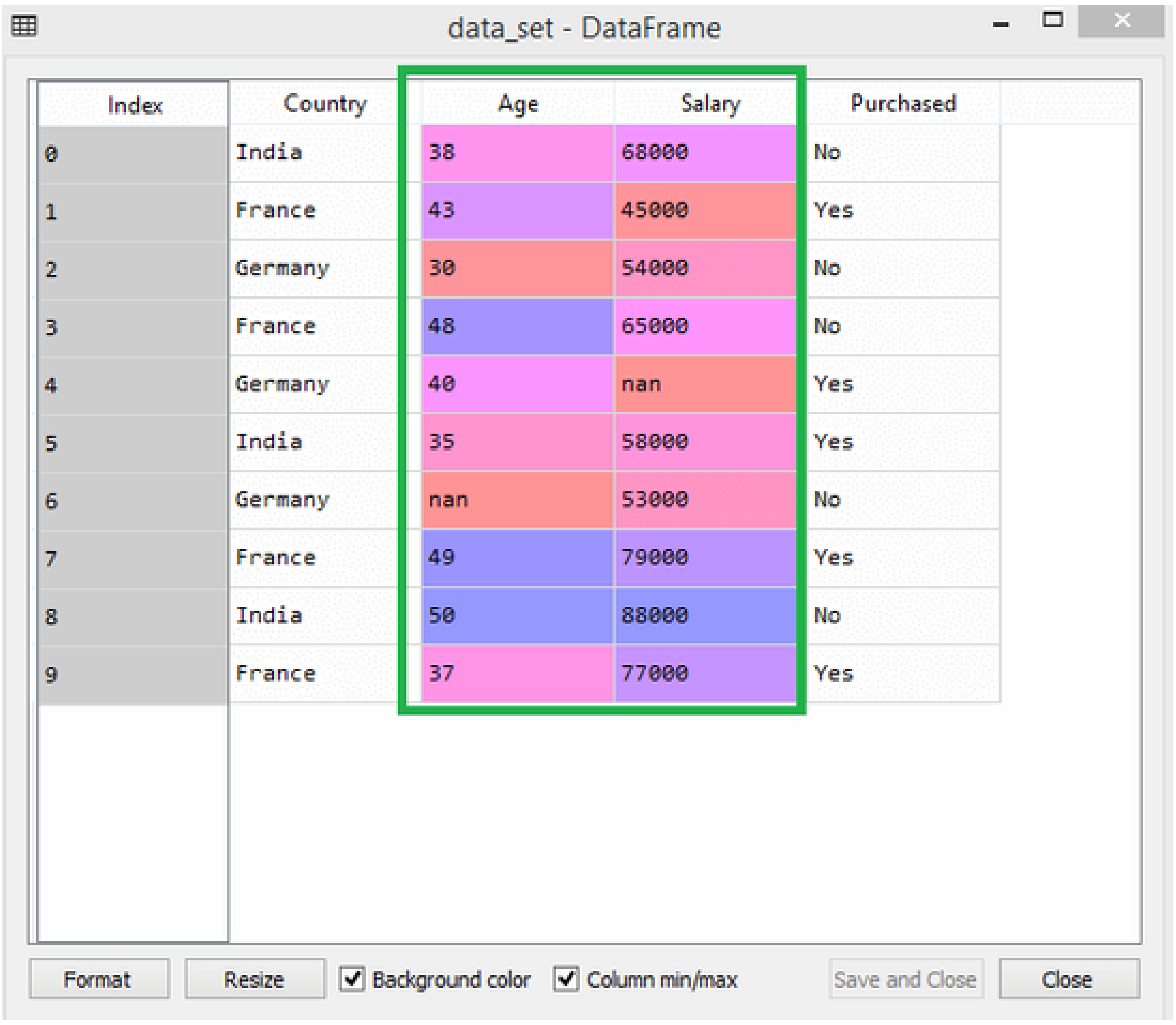
Name	Type	Size	Value
data_set	DataFrame	(10, 4)	Column names: Country, Age, Salary, Purchased
x	float64	(10, 5)	[[0.0e+00 0.0e+00 1.0e+00 3.8e+01 6.8e+04] [1.0e+00 0.0e+00 0.0e+00 4 ...
x_test	float64	(2, 5)	[[0.0e+00 1.0e+00 0.0e+00 3.0e+01 5.4e+04] [0.0e+00 0.0e+00 1.0e+00 5 ...
x_train	float64	(8, 5)	[[0.00000000e+00 1.00000000e+00 0.00000000e+00 4.00000000e+01 6.5222 ...
y	int32	(10,)	[0 1 0 0 1 1 0 1 0 1]
y_test	int32	(2,)	[0 0]
y_train	int32	(8,)	[1 1 1 0 1 0 0 1]

As we can see in the above image, the x and y variables are divided into 4 different variables with corresponding values.

7) Feature Scaling

Feature scaling is the final step of data preprocessing in machine learning. It is a technique to standardize the independent variables of the dataset in a specific range. In feature scaling, we put our variables in the same range and in the same scale so that no any variable dominate the other variable.

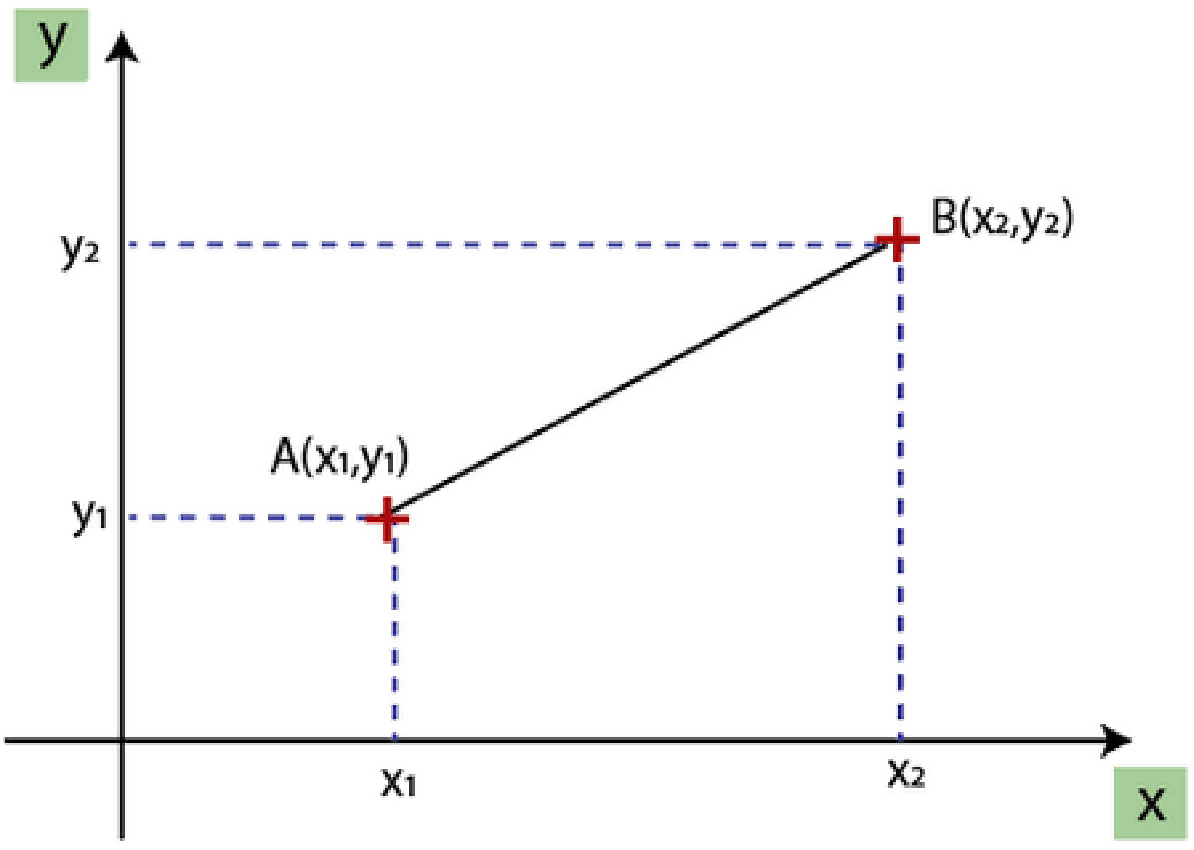
Consider the below dataset:



Index	Country	Age	Salary	Purchased
0	India	38	68000	No
1	France	43	45000	Yes
2	Germany	30	54000	No
3	France	48	65000	No
4	Germany	40	nan	Yes
5	India	35	58000	Yes
6	Germany	nan	53000	No
7	France	49	79000	Yes
8	India	50	88000	No
9	France	37	77000	Yes

As we can see, the age and salary column values are not on the same scale. A machine learning model is based on Euclidean distance, and if we do not scale the variable, then it will cause some issue in our machine learning model.

Euclidean distance is given as:

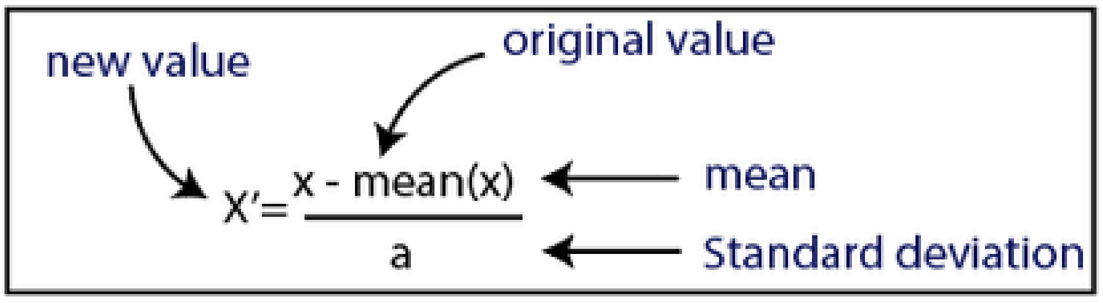


$$\text{Euclidean Distance Between A and B} = \sqrt{(x_2-x_1)^2+(y_2-y_1)^2}$$

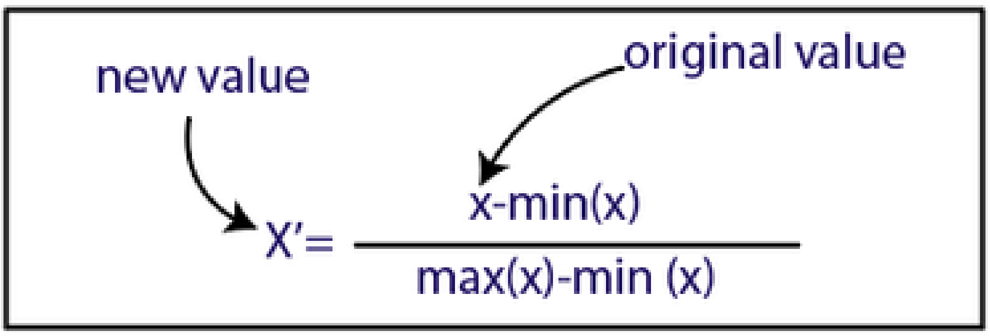
If we compute any two values from age and salary, then salary values will dominate the age values, and it will produce an incorrect result. So to remove this issue, we need to perform feature scaling for machine learning.

There are two ways to perform feature scaling in machine learning:

Standardization



Normalization



Here, we will use the standardization method for our dataset.

For feature scaling, we will import **StandardScaler** class of **sklearn.preprocessing** library as:

```
1. from sklearn.preprocessing import StandardScaler
```

Now, we will create the object of StandardScaler class for independent variables or features. And then we will fit and transform the training dataset.

```
1. st_x= StandardScaler()
2. x_train= st_x.fit_transform(x_train)
```

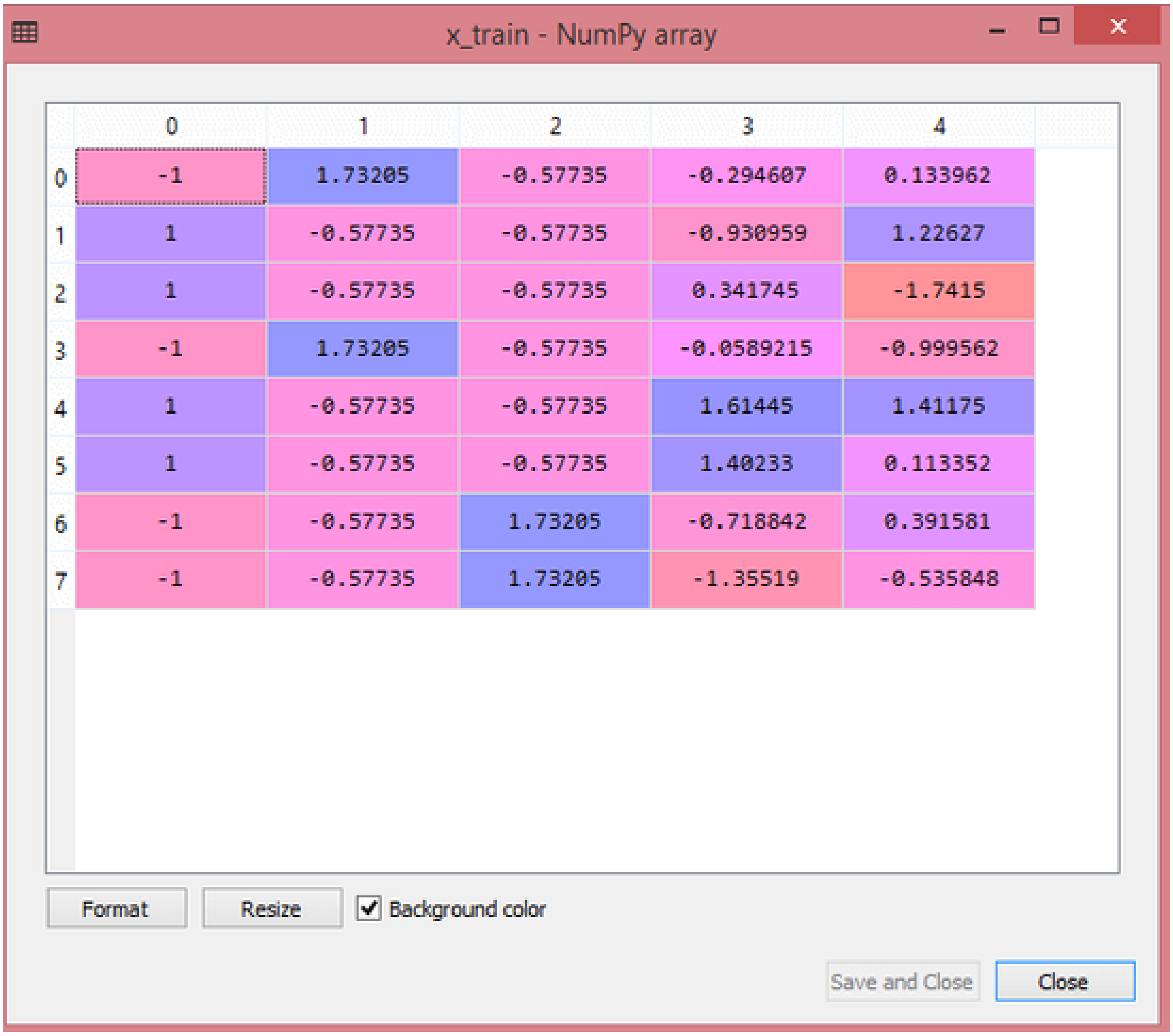
For test dataset, we will directly apply transform() function instead of fit_transform() because it is already done in training set.

```
1. x_test= st_x.transform(x_test)
```

Output:

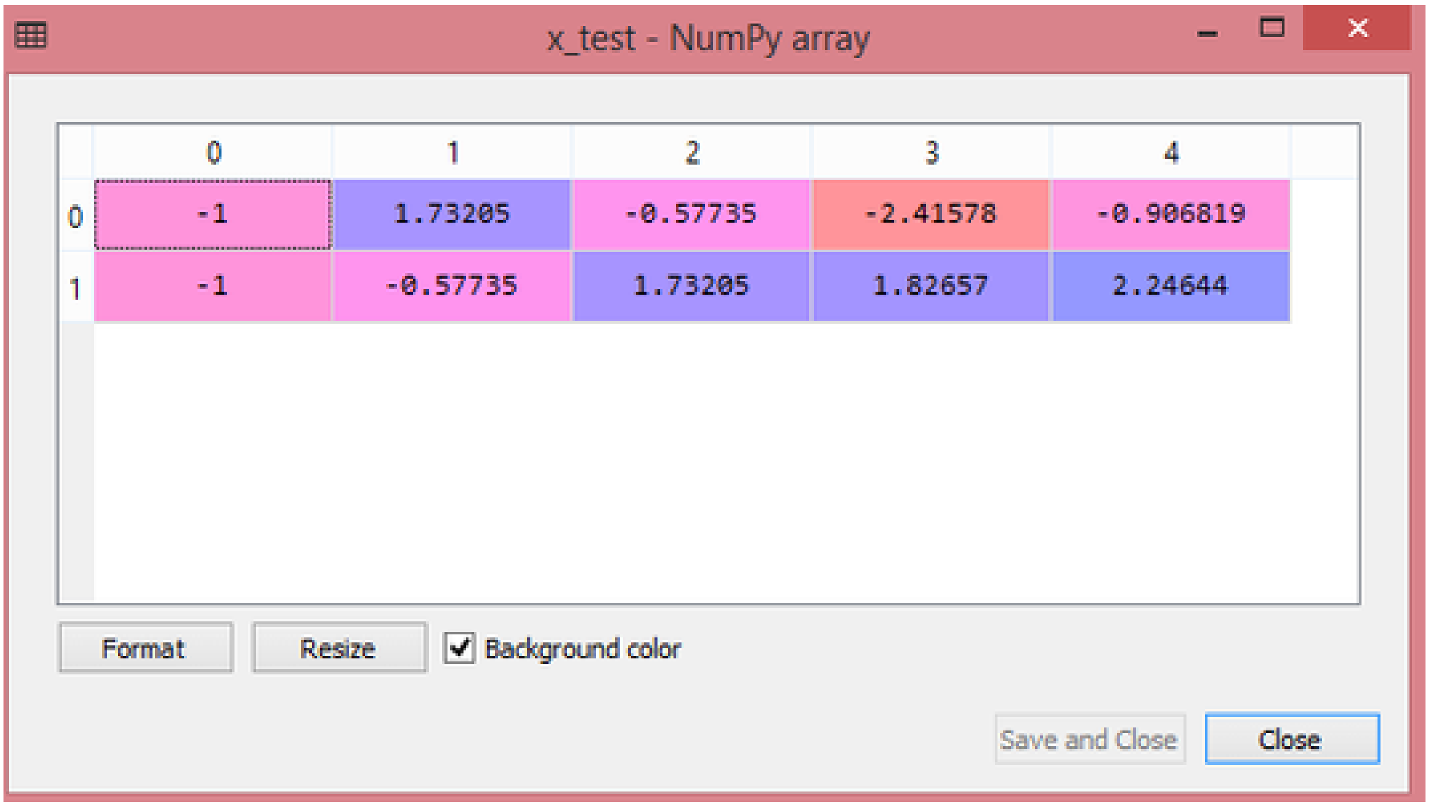
By executing the above lines of code, we will get the scaled values for x_train and x_test as:

x_train:



	0	1	2	3	4
0	-1	1.73205	-0.57735	-0.294607	0.133962
1	1	-0.57735	-0.57735	-0.930959	1.22627
2	1	-0.57735	-0.57735	0.341745	-1.7415
3	-1	1.73205	-0.57735	-0.0589215	-0.999562
4	1	-0.57735	-0.57735	1.61445	1.41175
5	1	-0.57735	-0.57735	1.40233	0.113352
6	-1	-0.57735	1.73205	-0.718842	0.391581
7	-1	-0.57735	1.73205	-1.35519	-0.535848

x_test:


x_test - NumPy array

	0	1	2	3	4
0	-1	1.73205	-0.57735	-2.41578	-0.906819
1	-1	-0.57735	1.73205	1.82657	2.24644

As we can see in the above output, all the variables are scaled between values -1 to 1.

Note: Here, we have not scaled the dependent variable because there are only two values 0 and 1. But if these variables will have more range of values, then we will also need to scale those variables.

Combining all the steps:

Now, in the end, we can combine all the steps together to make our complete code more understandable.

```
# importing libraries

import numpy as nm

import matplotlib.pyplot as mtp

import pandas as pd



#importing datasets

data_set= pd.read_csv('Dataset.csv')



#Extracting Independent Variable

x= data_set.iloc[:, :-1].values
```

```
#Extracting Dependent variable

y= data_set.iloc[:, 3].values


#handling missing data(Replacing missing data with the mean value)

from sklearn.preprocessing import Imputer

imputer= Imputer(missing_values ='NaN', strategy='mean', axis = 0)


#Fitting imputer object to the independent varibles x.

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


#Replacing missing data with the calculated mean value

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


#for Country Variable

from sklearn.preprocessing import LabelEncoder, OneHotEncoder

label_encoder_x= LabelEncoder()

x[:, 0]= label_encoder_x.fit_transform(x[:, 0])


#Encoding for dummy variables

onehot_encoder= OneHotEncoder(categorical_features= [0])

x= onehot_encoder.fit_transform(x).toarray()


#encoding for purchased variable

labelencoder_y= LabelEncoder()
```

```
y= labelencoder_y.fit_transform(y)



# Splitting the dataset into training and test set.

from sklearn.model_selection import train_test_split

x_train, x_test, y_train, y_test= train_test_split(x, y, test_size= 0.2, random_state=0)



#Feature Scaling of datasets

from sklearn.preprocessing import StandardScaler

st_x= StandardScaler()

x_train= st_x.fit_transform(x_train)

x_test= st_x.transform(x_test)
```

In the above code, we have included all the data preprocessing steps together. But there are some steps or lines of code which are not necessary for all machine learning models. So we can exclude them from our code to make it reusable for all models.

Difference between Supervised and Unsupervised Learning

Supervised and Unsupervised learning are the two techniques of machine learning. But both the techniques are used in different scenarios and with different datasets. Below the explanation of both learning methods along with their difference table is given.

Supervised Machine Learning:

Supervised learning is a machine learning method in which models are trained using labeled data. In supervised learning, models need to find the mapping function to map the input variable (X) with the output variable (Y).

Supervised learning needs supervision to train the model, which is similar to as a student learns things in the presence of a teacher. Supervised learning can be used for two types of problems: Classification and Regression.

Learn more Supervised Machine Learning

Example: Suppose we have an image of different types of fruits. The task of our supervised learning model is to identify the fruits and classify them accordingly. So to identify the image in supervised learning, we will give the input data as well as output for that, which means we will train the model by the shape, size, color, and taste of each fruit. Once the training is completed, we will test the model by giving the new set of fruit. The model will identify the fruit and predict the output using a suitable algorithm.

Unsupervised Machine Learning:

Unsupervised learning is another machine learning method in which patterns inferred from the unlabeled input data. The goal of unsupervised learning is to find the structure and patterns from the input data. Unsupervised learning does not need any supervision. Instead, it finds patterns from the data by its own.

Learn more Unsupervised Machine Learning

Unsupervised learning can be used for two types of problems: Clustering and Association.

Example: To understand the unsupervised learning, we will use the example given above. So unlike supervised learning, here we will not provide any supervision to the model. We will just provide the input dataset to the model and allow the model to find the patterns from the data. With the help of a suitable algorithm, the model will train itself and divide the fruits into different groups according to the most similar features between them.

The main differences between Supervised and Unsupervised learning are given below:

Supervised Learning	Unsupervised Learning
Supervised learning algorithms are trained using labeled data.	Unsupervised learning algorithms are trained using unlabeled data.
Supervised learning model takes direct feedback to check if it is predicting correct output or not.	Unsupervised learning model does not take any feedback.
Supervised learning model predicts the output.	Unsupervised learning model finds the hidden patterns in data.
In supervised learning, input data is provided to the model along with the output.	In unsupervised learning, only input data is provided to the model.
The goal of supervised learning is to train the model so that it can predict the output when it is given new data.	The goal of unsupervised learning is to find the hidden patterns and useful insights from the unknown dataset.
Supervised learning needs supervision to train the model.	Unsupervised learning does not need any supervision to train the model.
Supervised learning can be categorized in Classification and Regression problems.	Unsupervised Learning can be classified in Clustering and Associations problems.
Supervised learning can be used for those cases where we know the input as well as corresponding outputs.	Unsupervised learning can be used for those cases where we have only input data and no corresponding output data.
Supervised learning model produces an accurate result.	Unsupervised learning model may give less accurate result as compared to supervised learning.
Supervised learning is not close to true Artificial intelligence as in this, we first train the model for each data, and then only it can predict the correct output.	Unsupervised learning is more close to the true Artificial Intelligence as it learns similarly as a child learns daily routine things by his experiences.
It includes various algorithms such as Linear Regression, Logistic Regression, Support Vector Machine, Multi-class Classification, Decision tree, Bayesian Logic, etc.	It includes various algorithms such as Clustering and Apriori algorithm.

Overfitting and Underfitting in Machine Learning

Overfitting and Underfitting are the two main problems that occur in machine learning and degrade the performance of the machine learning models.

The main goal of each machine learning model is to generalize well. Here generalization defines the ability of an ML model to provide a suitable output by adapting the given set of unknown input. It means after providing training on the dataset, it can produce reliable and accurate output. Hence, the underfitting and overfitting are the two terms that need to be checked for the performance of the model and whether the model is generalizing well or not.

Before understanding the overfitting and underfitting, let's understand some basic term that will help to understand this topic well:

- Signal: It refers to the true underlying pattern of the data that helps the machine learning model to learn from the data.
- Noise: Noise is unnecessary and irrelevant data that reduces the performance of the model.
- Bias: Bias is a prediction error that is introduced in the model due to oversimplifying the machine learning algorithms. Or it is the difference between the predicted values and the actual values.
- Variance: If the machine learning model performs well with the training dataset, but does not perform well with the test dataset, then variance occurs.

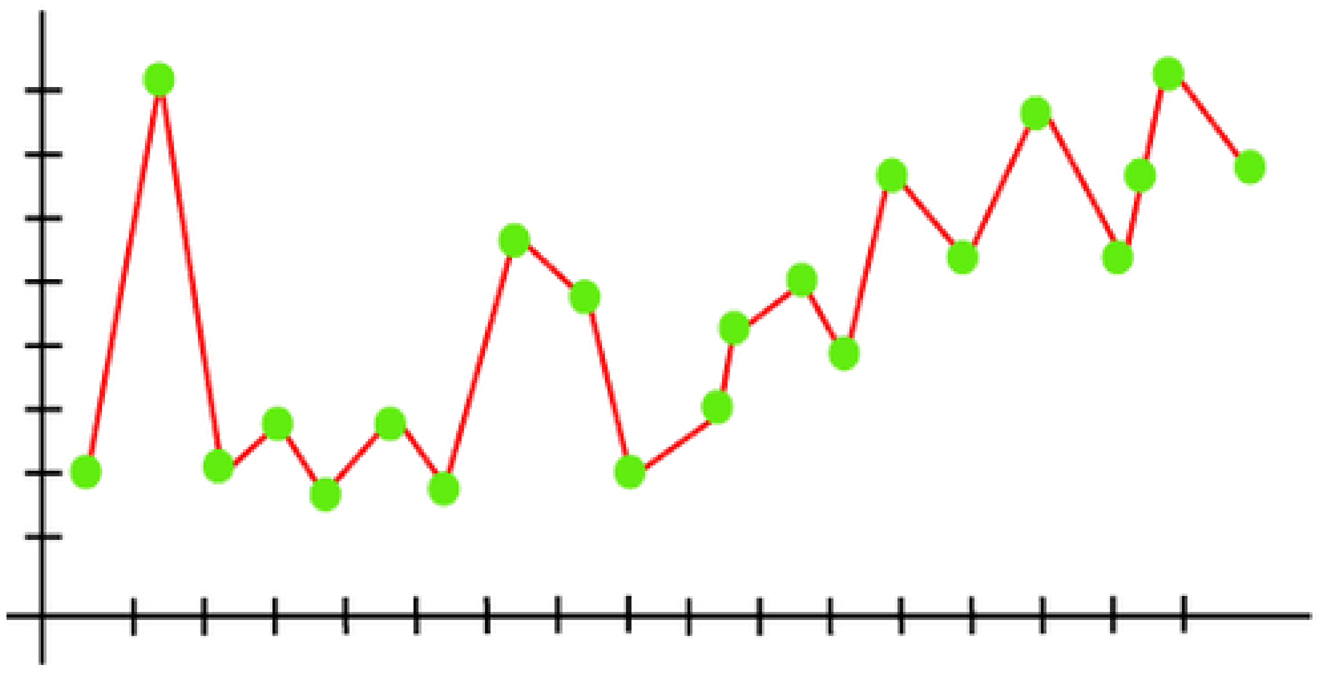
Overfitting

Overfitting occurs when our machine learning model tries to cover all the data points or more than the required data points present in the given dataset. Because of this, the model starts caching noise and inaccurate values present in the dataset, and all these factors reduce the efficiency and accuracy of the model. The overfitted model has low bias and high variance.

The chances of occurrence of overfitting increase as much we provide training to our model. It means the more we train our model, the more chances of occurring the overfitted model.

Overfitting is the main problem that occurs in supervised learning.

Example: The concept of the overfitting can be understood by the below graph of the linear regression output:



As we can see from the above graph, the model tries to cover all the data points present in the scatter plot. It may look efficient, but in reality, it is not so. Because the goal of the regression model to find the best fit line, but here we have not got any best fit, so, it will generate the prediction errors.

How to avoid the Overfitting in Model

Both overfitting and underfitting cause the degraded performance of the machine learning model. But the main cause is overfitting, so there are some ways by which we can reduce the occurrence of overfitting in our model.

- Cross-Validation
- Training with more data
- Removing features
- Early stopping the training
- Regularization
- Ensembling

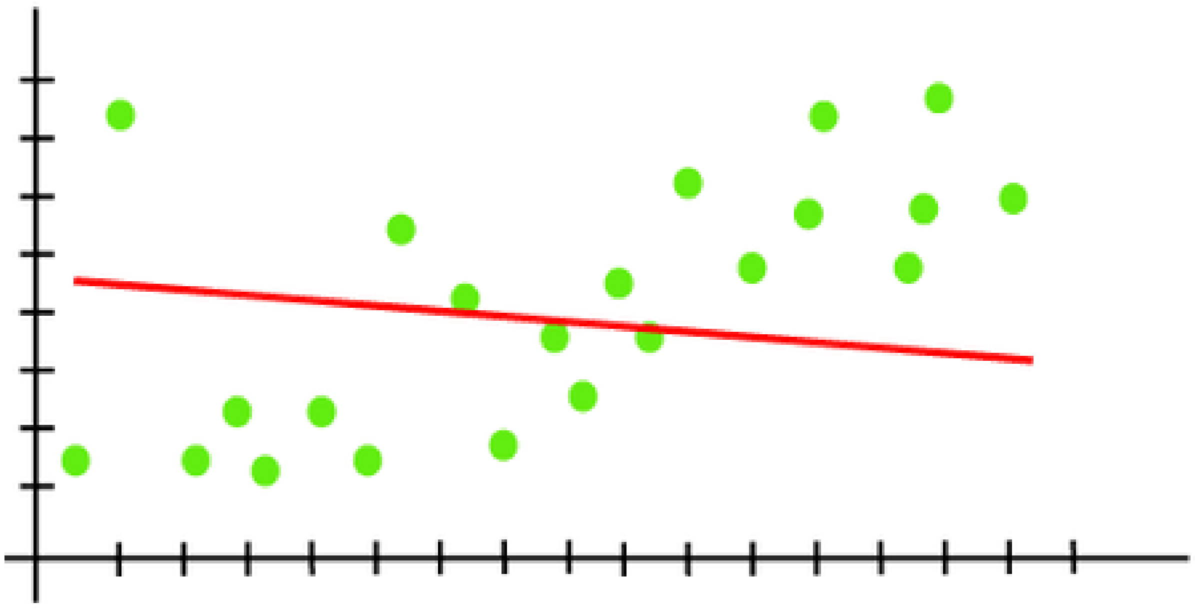
Underfitting

Underfitting occurs when our machine learning model is not able to capture the underlying trend of the data. To avoid the overfitting in the model, the fed of training data can be stopped at an early stage, due to which the model may not learn enough from the training data. As a result, it may fail to find the best fit of the dominant trend in the data.

In the case of underfitting, the model is not able to learn enough from the training data, and hence it reduces the accuracy and produces unreliable predictions.

An underfitted model has high bias and low variance.

Example: We can understand the underfitting using below output of the linear regression model:



As we can see from the above diagram, the model is unable to capture the data points present in the plot.

How to avoid underfitting:

- By increasing the training time of the model.
- By increasing the number of features.

Goodness of Fit

The "Goodness of fit" term is taken from the statistics, and the goal of the machine learning models to achieve the goodness of fit. In statistics modeling, it defines how closely the result or predicted values match the true values of the dataset.

The model with a good fit is between the underfitted and overfitted model, and ideally, it makes predictions with 0 errors, but in practice, it is difficult to achieve it.

As when we train our model for a time, the errors in the training data go down, and the same happens with test data. But if we train the model for a long duration, then the performance of the model may decrease due to the overfitting, as the model also learn the noise present in the dataset. The errors in the test dataset start increasing, so the point, just before the raising of errors, is the good point, and we can stop here for achieving a good model.

There are two other methods by which we can get a good point for our model, which are the resampling method to estimate model accuracy and validation dataset.

What Is Model Selection

Model selection is the process of selecting one final machine learning model from among a collection of candidate machine learning models for a training dataset.

Model selection is a process that can be applied both across different types of models (e.g. logistic regression, SVM, KNN, etc.) and across models of the same type configured with different model hyperparameters (e.g. different kernels in an SVM).

When we have a variety of models of different complexity (e.g., linear or logistic regression models with different degree polynomials, or KNN classifiers with different values of K), how should we pick the right one?

— Page 22, Machine Learning: A Probabilistic Perspective, 2012.

For example, we may have a dataset for which we are interested in developing a classification or regression predictive model. We do not know beforehand as to which model will perform best on this problem, as it is unknowable. Therefore, we fit and evaluate a suite of different models on the problem.

Model selection is the process of choosing one of the models as the final model that addresses the problem.

Model selection is different from **model assessment**.

For example, we evaluate or assess candidate models in order to choose the best one, and this is model selection. Whereas once a model is chosen, it can be evaluated in order to communicate how well it is expected to perform in general; this is model assessment.

The process of evaluating a model's performance is known as model assessment, whereas the process of selecting the proper level of flexibility for a model is known as model selection.

— Page 175, An Introduction to Statistical Learning: with Applications in R, 2017.

Considerations for Model Selection

Fitting models is relatively straightforward, although selecting among them is the true challenge of applied machine learning.

Firstly, we need to get over the idea of a "*best*" model.

All models have some predictive error, given the statistical noise in the data, the incompleteness of the data sample, and the limitations of each different model type. Therefore, the notion of a perfect or best model is not useful. Instead, we must seek a model that is "*good enough*."

What do we care about when choosing a final model?

The project stakeholders may have specific requirements, such as maintainability and limited model complexity. As such, a model that has lower skill but is simpler and easier to understand may be preferred.

Alternately, if model skill is prized above all other concerns, then the ability of the model to perform well on out-of-sample data will be preferred regardless of the computational complexity involved.

Therefore, a "*good enough*" model may refer to many things and is specific to your project, such as:

- A model that meets the requirements and constraints of project stakeholders.
- A model that is sufficiently skillful given the time and resources available.
- A model that is skillful as compared to naive models.
- A model that is skillful relative to other tested models.
- A model that is skillful relative to the state-of-the-art.

Next, we must consider what is being selected.

For example, we are not selecting a fit model, as all models will be discarded. This is because once we choose a model, we will fit a new final model on all available data and start using it to make predictions.

Therefore, are we choosing among algorithms used to fit the models on the training dataset?

Some algorithms require specialized data preparation in order to best expose the structure of the problem to the learning algorithm. Therefore, we must go one step further and consider **model selection as the process of selecting among model development pipelines**.

Each pipeline may take in the same raw training dataset and outputs a model that can be evaluated in the same manner but may require different or overlapping computational steps, such as:

- Data filtering.
- Data transformation.
- Feature selection.
- Feature engineering.
- And more…

The closer you look at the challenge of model selection, the more nuance you will discover.

Now that we are familiar with some considerations involved in model selection, let's review some common methods for selecting a model.

Model Selection Techniques

The best approach to model selection requires "*sufficient*" data, which may be nearly infinite depending on the complexity of the problem.

In this ideal situation, we would split the data into training, validation, and test sets, then fit candidate models on the training set, evaluate and select them on the validation set, and report the performance of the final model on the test set.

If we are in a data-rich situation, the best approach […] is to randomly divide the dataset into three parts: a training set, a validation set, and a test set. The training set is used to fit the models; the validation set is *used to estimate prediction error for model selection; the test set is used for assessment of the generalization error of the final chosen model.*

— Page 222, The Elements of Statistical Learning: Data Mining, Inference, and Prediction, 2017.

This is impractical on most predictive modeling problems given that we rarely have sufficient data, or are able to even judge what would be sufficient.

In many applications, however, the supply of data for training and testing will be limited, and in order to build good models, we wish to use as much of the available data as possible for training. However, if the validation set is small, it will give a relatively noisy estimate of predictive performance.

– Page 32, Pattern Recognition and Machine Learning, 2006.

Instead, there are two main classes of techniques to approximate the ideal case of model selection; they are:

- **Probabilistic Measures**: Choose a model via in-sample error and complexity.
- **Resampling Methods**: Choose a model via estimated out-of-sample error.

Let's take a closer look at each in turn.

Probabilistic Measures

Probabilistic measures involve analytically scoring a candidate model using both its performance on the training dataset and the complexity of the model.

It is known that training error is optimistically biased, and therefore is not a good basis for choosing a model. The performance can be penalized based on how optimistic the training error is believed to be. This is typically achieved using algorithm-specific methods, often linear, that penalize the score based on the complexity of the model.

Historically various 'information criteria' have been proposed that attempt to correct for the bias of maximum likelihood by the addition of a penalty term to compensate for the over-fitting of more complex models.

– Page 33, Pattern Recognition and Machine Learning, 2006.

A model with fewer parameters is less complex, and because of this, is preferred because it is likely to generalize better on average.

Four commonly used probabilistic model selection measures include:

- Akaike Information Criterion (AIC).
- Bayesian Information Criterion (BIC).
- Minimum Description Length (MDL).
- Structural Risk Minimization (SRM).

Probabilistic measures are appropriate when using simpler linear models like linear regression or logistic regression where the calculating of model complexity penalty (e.g. in sample bias) is known and tractable.

Resampling Methods

Resampling methods seek to estimate the performance of a model (or more precisely, the model development process) on out-of-sample data.

This is achieved by splitting the training dataset into sub train and test sets, fitting a model on the sub train set, and evaluating it on the test set. This process may then be repeated multiple times and the mean performance across each trial is reported.

It is a type of Monte Carlo estimate of model performance on out-of-sample data, although each trial is not strictly independent as depending on the resampling method chosen, the same data may appear multiple times in different training datasets, or test datasets.

Three common resampling model selection methods include:

- Random train/test splits.
- Cross-Validation (k-fold, LOOCV, etc.).
- Bootstrap.

Most of the time probabilistic measures (described in the previous section) are not available, therefore resampling methods are used.

By far the most popular is the cross-validation family of methods that includes many subtypes.

Probably the simplest and most widely used method for estimating prediction error is cross-validation.

— Page 241, The Elements of Statistical Learning: Data Mining, Inference, and Prediction, 2017.

An example is the widely used k-fold cross-validation that splits the training dataset into k folds where each example appears in a test set only once.

Another is the leave one out (LOOCV) where the test set is comprised of a single sample and each sample is given an opportunity to be the test set, requiring N (the number of samples in the training set) models to be constructed and evaluated.

Overfitting in Machine Learning

In the real world, the dataset present will never be clean and perfect. It means each dataset contains impurities, noisy data, outliers, missing data, or imbalanced data. Due to these impurities, different problems occur that affect the accuracy and the performance of the model. One of such problems is Overfitting in Machine Learning. Overfitting is a problem that a model can exhibit.

A statistical model is said to be overfitted if it can't generalize well with unseen data.

Before understanding overfitting, we need to know some basic terms, which are:

Noise: Noise is meaningless or irrelevant data present in the dataset. It affects the performance of the model if it is not removed.

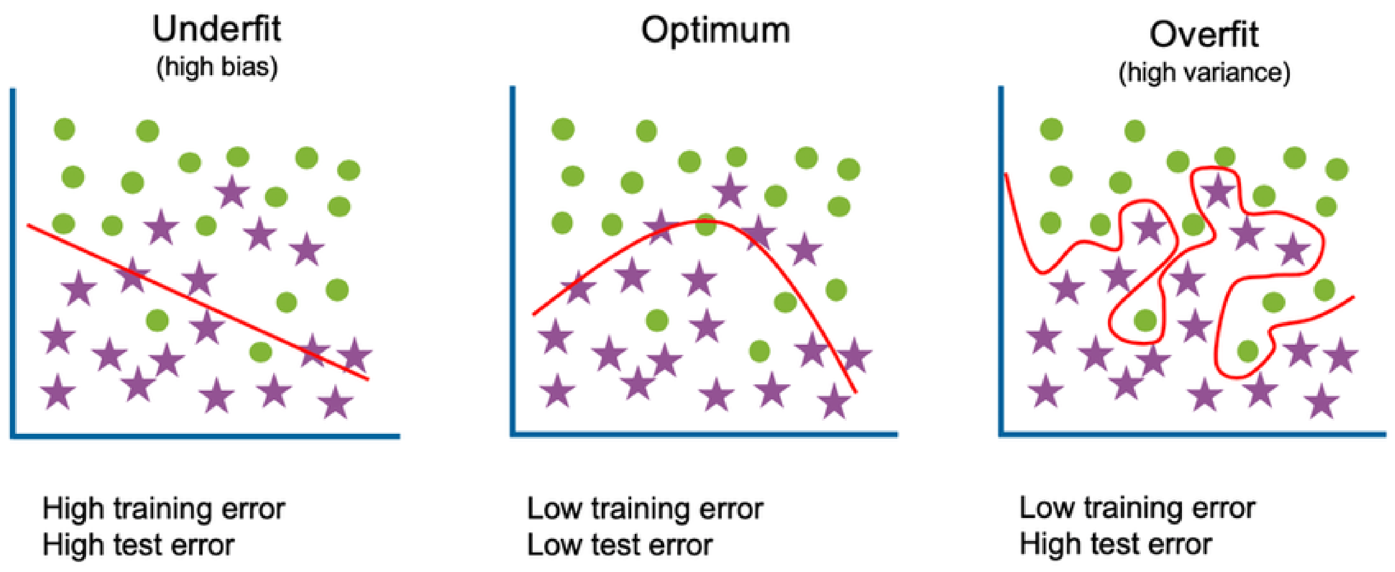
Bias: Bias is a prediction error that is introduced in the model due to oversimplifying the machine learning algorithms. Or it is the difference between the predicted values and the actual values.

Stay

Variance: If the machine learning model performs well with the training dataset, but does not perform well with the test dataset, then variance occurs.

Generalization: It shows how well a model is trained to predict unseen data.

What is Overfitting?



- Overfitting & underfitting are the two main errors/problems in the machine learning model, which cause poor performance in Machine Learning.
- Overfitting occurs when the model fits more data than required, and it tries to capture each and every datapoint fed to it. Hence it starts capturing noise and inaccurate data from the dataset, which degrades the performance of the model.
- An overfitted model doesn't perform accurately with the test/unseen dataset and can’t generalize well.
- An overfitted model is said to have low bias and high variance.

Example to Understand Overfitting

We can understand overfitting with a general example. Suppose there are three students, X, Y, and Z, and all three are preparing for an exam. X has studied only three sections of the book and left all other sections. Y has a good memory, hence memorized the whole book. And the third student, Z, has studied and practiced all the questions. So, in the exam, X will only be able to solve the questions if the exam has questions related to section 3. Student Y will only be able to solve questions if they appear exactly the same as given in the book. Student Z will be able to solve all the exam questions in a proper way.

The same happens with machine learning; if the algorithm learns from a small part of the data, it is unable to capture the required data points and hence under fitted.

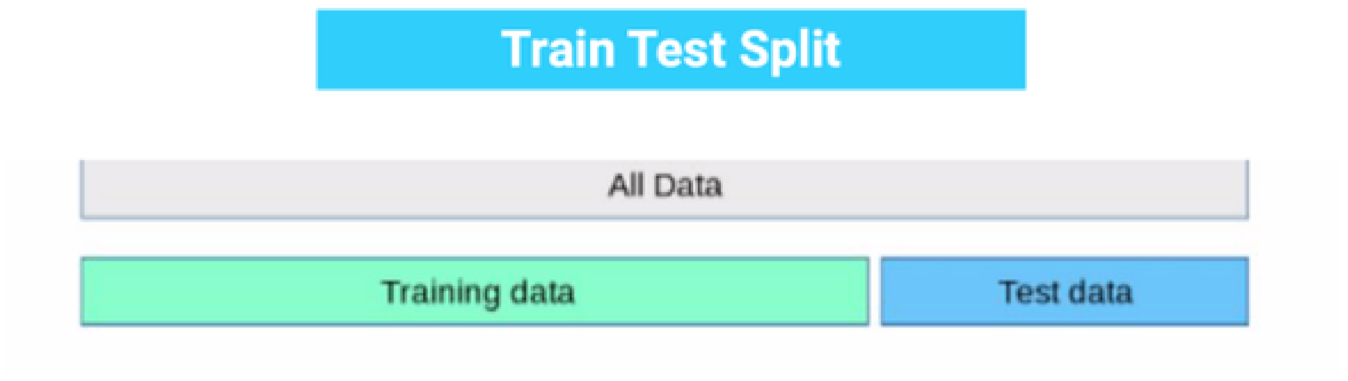
Suppose the model learns the training dataset, like the Y student. They perform very well on the seen dataset but perform badly on unseen data or unknown instances. In such cases, the model is said to be Overfitting.

And if the model performs well with the training dataset and also with the test/unseen dataset, similar to student Z, it is said to be a good fit.

How to detect Overfitting?

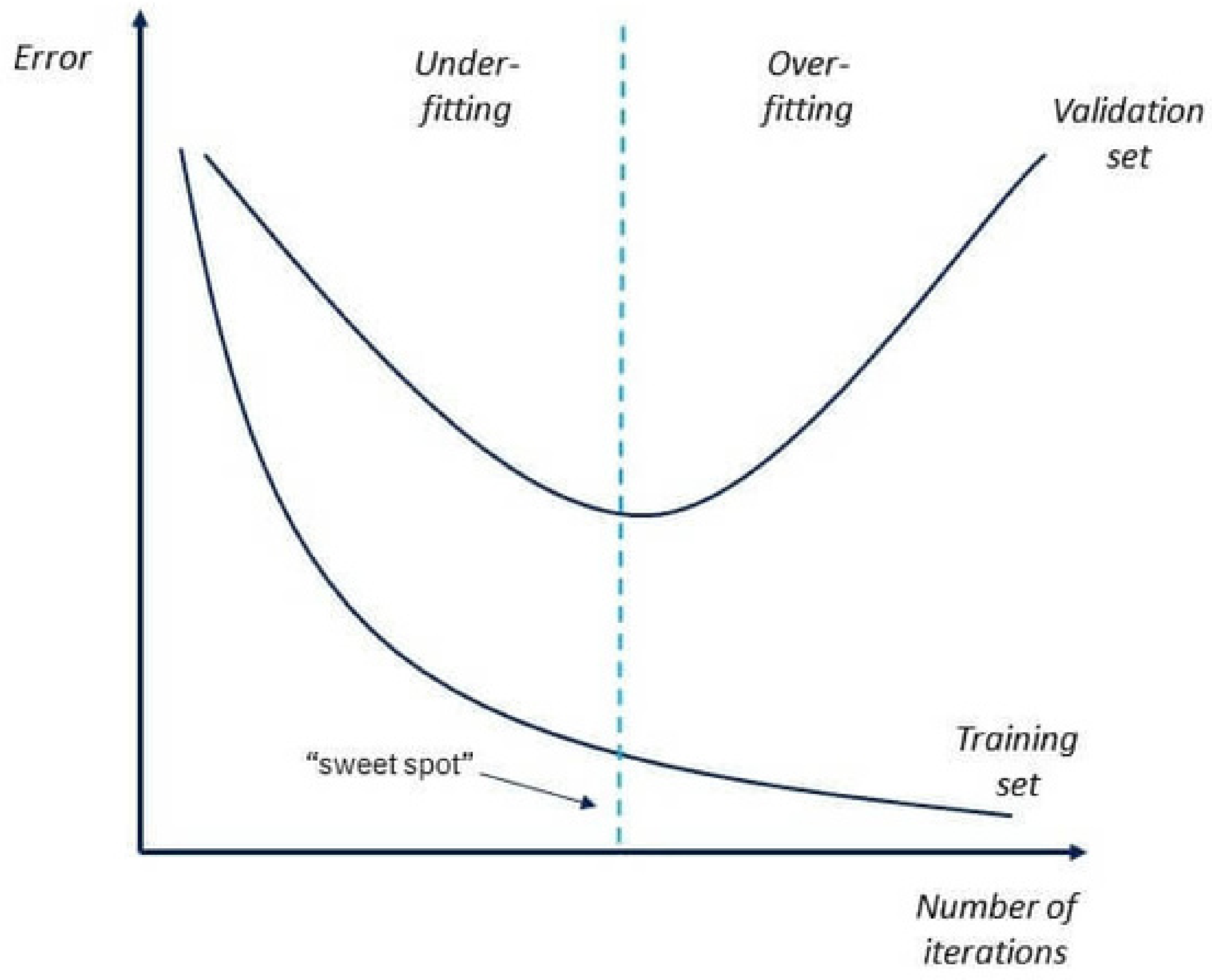
Overfitting in the model can only be detected once you test the data. To detect the issue, we can perform Train/test split.

In the train-test split of the dataset, we can divide our dataset into random test and training datasets. We train the model with a training dataset which is about 80% of the total dataset. After training the model, we test it with the test dataset, which is 20 % of the total dataset.



Now, if the model performs well with the training dataset but not with the test dataset, then it is likely to have an overfitting issue.

For example, if the model shows 85% accuracy with training data and 50% accuracy with the test dataset, it means the model is not performing well.



Ways to prevent the Overfitting

Although overfitting is an error in Machine learning which reduces the performance of the model, however, we can prevent it in several ways. With the use of the linear model, we can avoid overfitting; however, many real-world problems are non-linear ones. It is important to prevent overfitting from the models. Below are several ways that can be used to prevent overfitting:

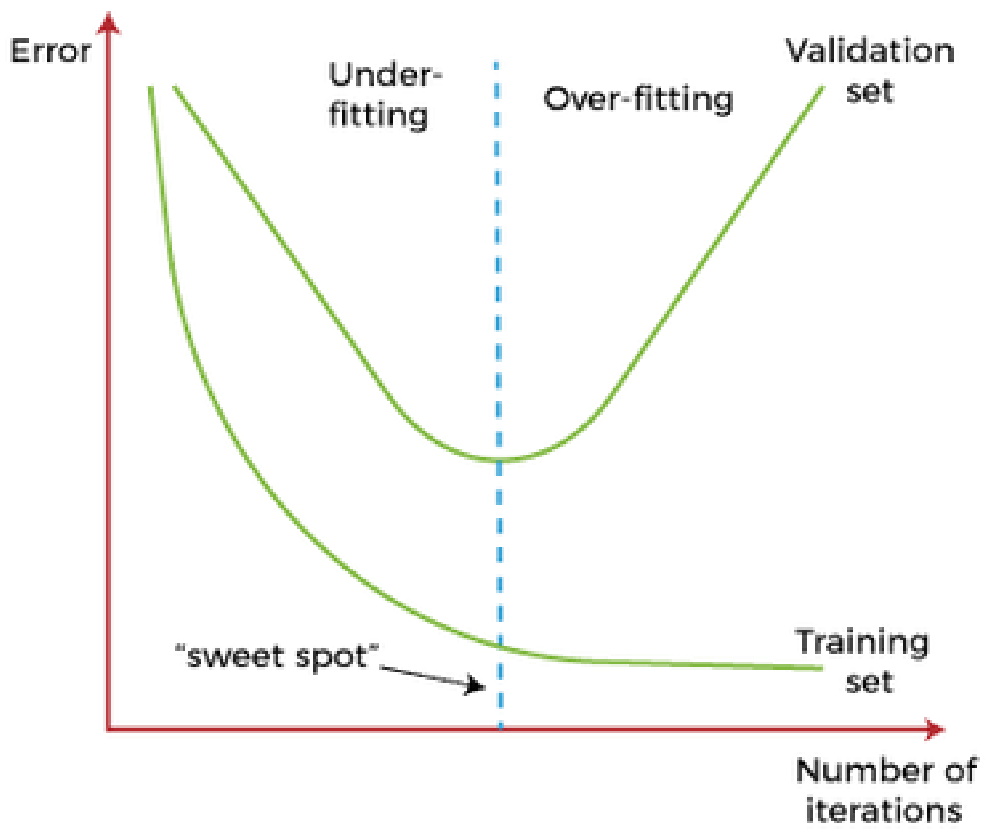
1. Early Stopping
2. Train with more data
3. Feature Selection
4. Cross-Validation
5. Data Augmentation
6. Regularization

Early Stopping

In this technique, the training is paused before the model starts learning the noise within the model. In this process, while training the model iteratively, measure the performance of the model after each iteration. Continue up to a certain number of iterations until a new iteration improves the performance of the model.

After that point, the model begins to overfit the training data; hence we need to stop the process before the learner passes that point.

Stopping the training process before the model starts capturing noise from the data is known as early stopping.



However, this technique may lead to the underfitting problem if training is paused too early. So, it is very important to find that "sweet spot" between underfitting and overfitting.

Train with More data

Increasing the training set by including more data can enhance the accuracy of the model, as it provides more chances to discover the relationship between input and output variables.

It may not always work to prevent overfitting, but this way helps the algorithm to detect the signal better to minimize the errors.

When a model is fed with more training data, it will be unable to overfit all the samples of data and forced to generalize well.

But in some cases, the additional data may add more noise to the model; hence we need to be sure that data is clean and free from in-consistencies before feeding it to the model.

Feature Selection

While building the ML model, we have a number of parameters or features that are used to predict the outcome. However, sometimes some of these features are redundant or less important for the prediction, and for this feature selection process is applied. In the feature selection process, we identify the most important features within training data, and other features are removed. Further, this process helps to simplify the model and reduces noise from the data. Some algorithms have the auto-feature selection, and if not, then we can manually perform this process.

Cross-Validation

Cross-validation is one of the powerful techniques to prevent overfitting.

In the general k-fold cross-validation technique, we divided the dataset into k-equal-sized subsets of data; these subsets are known as folds.

Data Augmentation

Data Augmentation is a data analysis technique, which is an alternative to adding more data to prevent overfitting. In this technique, instead of adding more training data, slightly modified copies of already existing data are added to the dataset.

The data augmentation technique makes it possible to appear data sample slightly different every time it is processed by the model. Hence each data set appears unique to the model and prevents overfitting.

Regularization

If overfitting occurs when a model is complex, we can reduce the number of features. However, overfitting may also occur with a simpler model, more specifically the Linear model, and for such cases, regularization techniques are much helpful.

Regularization is the most popular technique to prevent overfitting. It is a group of methods that forces the learning algorithms to make a model simpler. Applying the regularization technique may slightly increase the bias but slightly reduces the variance. In this technique, we modify the objective function by adding the penalizing term, which has a higher value with a more complex model.

The two commonly used regularization techniques are L1 Regularization and L2 Regularization.

Ensemble Methods

In ensemble methods, prediction from different machine learning models is combined to identify the most popular result.

The most commonly used ensemble methods are Bagging and Boosting.

In bagging, individual data points can be selected more than once. After the collection of several sample datasets, these models are trained independently, and depending on the type of task-i.e., regression or classification-the average of those predictions is used to predict a more accurate result. Moreover, bagging reduces the chances of overfitting in complex models.

In boosting, a large number of weak learners arranged in a sequence are trained in such a way that each learner in the sequence learns from the mistakes of the learner before it. It combines all the weak learners to come out with one strong learner. In addition, it improves the predictive flexibility of simple models

Cross-Validation in Machine Learning

Cross-validation is a technique for validating the model efficiency by training it on the subset of input data and testing on previously unseen subset of the input data. **We can also say that it is a technique to check how a statistical model generalizes to an independent dataset**.

In machine learning, there is always the need to test the stability of the model. It means based only on the training dataset; we can't fit our model on the training dataset. For this purpose, we reserve a particular sample of the dataset, which was not part of the training dataset. After that, we test our model on that sample before deployment, and this complete process comes under cross-validation. This is something different from the general train-test split.

Hence the basic steps of cross-validations are:

- Reserve a subset of the dataset as a validation set.
- Provide the training to the model using the training dataset.
- Now, evaluate model performance using the validation set. If the model performs well with the validation set, perform the further step, else check for the issues.

Methods used for Cross-Validation

There are some common methods that are used for cross-validation. These methods are given below:

1. Validation Set Approach
2. Leave-P-out cross-validation
3. Leave one out cross-validation
4. K-fold cross-validation
5. Stratified k-fold cross-validation

Validation Set Approach

We divide our input dataset into a training set and test or validation set in the validation set approach. Both the subsets are given 50% of the dataset.

But it has one of the big disadvantages that we are just using a 50% dataset to train our model, so the model may miss out to capture important information of the dataset. It also tends to give the underfitted model.

Leave-P-out cross-validation

In this approach, the p datasets are left out of the training data. It means, if there are total n datapoints in the original input dataset, then n-p data points will be used as the training dataset and the p data points as the validation set. This complete process is repeated for all the samples, and the average error is calculated to know the effectiveness of the model.

There is a disadvantage of this technique; that is, it can be computationally difficult for the large p.

Leave one out cross-validation

This method is similar to the leave-p-out cross-validation, but instead of p, we need to take 1 dataset out of training. It means, in this approach, for each learning set, only one datapoint is reserved, and the remaining dataset is used to train the model. This process repeats for each datapoint. Hence for n samples, we get n different training set and n test set. It has the following features:

- In this approach, the bias is minimum as all the data points are used.
- The process is executed for n times; hence execution time is high.
- This approach leads to high variation in testing the effectiveness of the model as we iteratively check against one data point.

K-Fold Cross-Validation

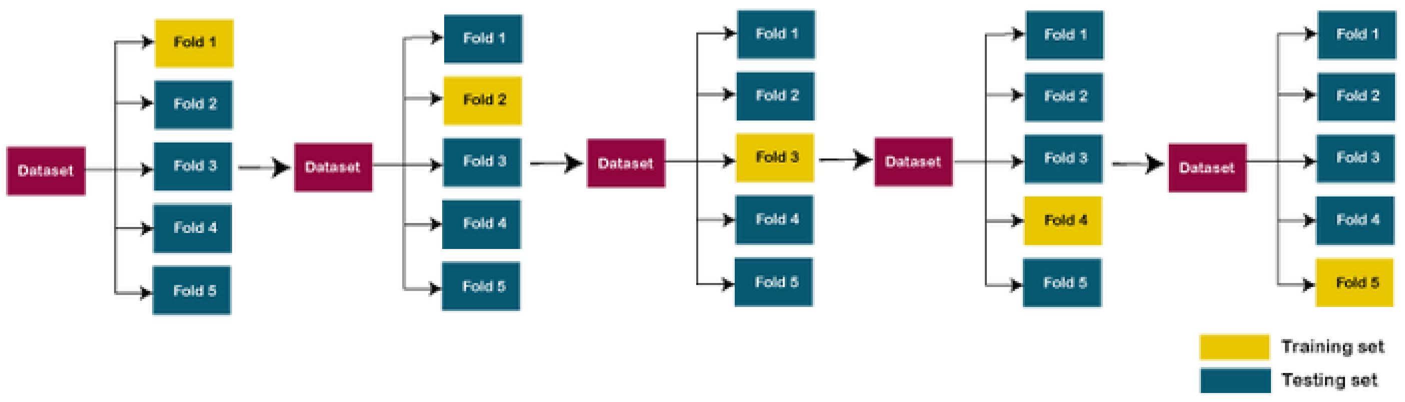
K-fold cross-validation approach divides the input dataset into K groups of samples of equal sizes. These samples are called folds. For each learning set, the prediction function uses k-1 folds, and the rest of the folds are used for the test set. This approach is a very popular CV approach because it is easy to understand, and the output is less biased than other methods.

The steps for k-fold cross-validation are:

- Split the input dataset into K groups
- For each group:
    - Take one group as the reserve or test data set.
    - Use remaining groups as the training dataset
    - Fit the model on the training set and evaluate the performance of the model using the test set.

Let's take an example of 5-folds cross-validation. So, the dataset is grouped into 5 folds. On 1st iteration, the first fold is reserved for test the model, and rest are used to train the model. On 2nd iteration, the second fold is used to test the model, and rest are used to train the model. This process will continue until each fold is not used for the test fold.

Consider the below diagram:



Stratified k-fold cross-validation

This technique is similar to k-fold cross-validation with some little changes. This approach works on stratification concept, it is a process of rearranging the data to ensure that each fold or group is a good representative of the complete dataset. To deal with the bias and variance, it is one of the best approaches.

It can be understood with an example of housing prices, such that the price of some houses can be much high than other houses. To tackle such situations, a stratified k-fold cross-validation technique is useful.

Holdout Method

This method is the simplest cross-validation technique among all. In this method, we need to remove a subset of the training data and use it to get prediction results by training it on the rest part of the dataset.

The error that occurs in this process tells how well our model will perform with the unknown dataset. Although this approach is simple to perform, it still faces the issue of high variance, and it also produces misleading results sometimes.

Comparison of Cross-validation to train/test split in Machine Learning

- Train/test split: The input data is divided into two parts, that are training set and test set on a ratio of 70:30, 80:20, etc. It provides a high variance, which is one of the biggest disadvantages.
  - Training Data: The training data is used to train the model, and the dependent variable is known.
  - Test Data: The test data is used to make the predictions from the model that is already trained on the training data. This has the same features as training data but not the part of that.
- Cross-Validation dataset: It is used to overcome the disadvantage of train/test split by splitting the dataset into groups of train/test splits, and averaging the result. It can be used if we want to optimize our model that has been trained on the training dataset for the best performance. It is more efficient as compared to train/test split as every observation is used for the training and testing both.

Limitations of Cross-Validation

There are some limitations of the cross-validation technique, which are given below:

- For the ideal conditions, it provides the optimum output. But for the inconsistent data, it may produce a drastic result. So, it is one of the big disadvantages of cross-validation, as there is no certainty of the type of data in machine learning.
- In predictive modeling, the data evolves over a period, due to which, it may face the differences between the training set and validation sets. Such as if we create a model for the prediction of stock market values, and the data is trained on the previous 5 years stock values, but the realistic future values for the next 5 years may drastically different, so it is difficult to expect the correct output for such situations.

Applications of Cross-Validation

- This technique can be used to compare the performance of different predictive modeling methods.
- It has great scope in the medical research field.
- It can also be used for the meta-analysis, as it is already being used by the data scientists in the field of medical statistics.