

# **STOCK PRICE FORECASTING**

## **A PROJECT REPORT**

*In partial fulfilment of the requirements for the award of the degree*

### **BACHELOR OF TECHNOLOGY**

**IN**

### **COMPUTER SCIENCE AND ENGINEERING**

*Under the guidance of*

**MAHENDRA DATTA**

**BY**

**SANSKRITI SHARMA**

**HARSHIT RAJPAL**



**FUTURE INSTITUTE OF ENGINEERING, KOLKATA**

**In association with**



**(ISO9001:2015)**

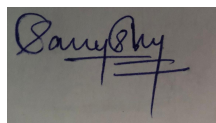
SDF Building, Module #132, Ground Floor, Salt Lake City, GP Block, Sector V, Kolkata, West Bengal 700091

1. Title of the Project: **STOCK PRICE FORECASTING**
2. Project Members: **SANSKRITI SHARMA**  
**HARSHIT RAJPAL**
3. Name of the guide: **Mr. MAHENDRA DATTA**
4. Address: Ardent Computech Pvt. Ltd  
(An ISO 9001:2015 Certified)  
SDF Building, Module #132, Ground Floor, Salt  
Lake City, GP Block, Sector V, Kolkata, West  
Bengal, 700091

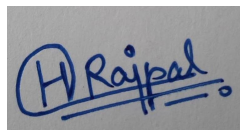
**Project Version Control History**

Version	Primary Author	Description of Version	Date Completed
Final	SANSKRITI SHARMA HARSHIT RAJPAL	Project Report	17 <sup>TH</sup> JULY ,2023

Signature of Team Member



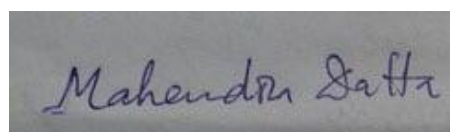
SANSKRITI SHARMA



HARSHIT RAJPAL

Date: 17-07-2023

Signature of Approver



**MR.MAHENDA DATTA**

Project Proposal Evaluator

Date: 17-07-2023

## **DECLARATION**

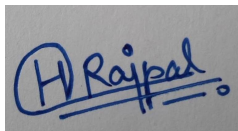
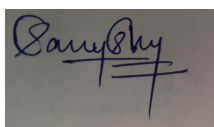
We hereby declare that the project work being presented in the project proposal entitled “**STOCK PRICE FORECASTING**” in partial fulfilment of the requirements for the award of the degree of **BACHELOR OF TECHNOLOGY** at **ARDENT COMPUTECH PVT. LTD, SALT LAKE, KOLKATA, WEST BENGAL**, is an authentic work carried out under the guidance of **MR. MAHENDRA DATTA**. The matter embodied in this project work has not been submitted elsewhere for the award of any degree of our knowledge and belief.

Date: 17-07-2023

Name of the Student: SANSKRITI SHARMA

HARSHIT RAJPAL

*Signature of the students:*



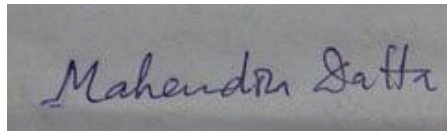
**Ardent Computech Pvt. Ltd (An ISO 9001:2015 Certified)**

SDF Building, Module #132, Ground Floor, Salt Lake City, GP Block, Sector V, Kolkata, West Bengal 700091

## **CERTIFICATE**

This is to certify that this proposal of minor project entitled “**STOCK PRICE FORECASTING**” is a record of bonafide work, carried out by **SANSKRITI SHARMA , HARSHIT RAJPAL** under my guidance at **ARDENT COMPUTECH PVT LTD**. In my opinion, the report in its present form is in partial fulfilment of the requirements for the award of the degree of **BACHELOR OF TECHNOLOGY** and as per regulations of the **ARDENT®**. To the best of my knowledge, the results embodied in this report, are original in nature and worthy of incorporation in the present version of the report.

**Guide / Supervisor**



-----  
**MR. MAHENDRA DATTA**

**Project Engineer**

**Ardent Computech Pvt. Ltd (An ISO 9001:2015 Certified)**

SDF Building, Module #132, Ground Floor, Salt Lake City, GP Block, Sector V, Kolkata, West Bengal 700091

## ACKNOWLEDGEMENT

Success of any project depends largely on the encouragement and guidelines of many others. I take this sincere opportunity to express my gratitude to the people who have been instrumental in the successful completion of this project work.

I would like to show our greatest appreciation to *Mr. MAHENDRA DATTA*, Project Engineer at Ardent, Kolkata. I always feel motivated and encouraged every time by his valuable advice and constant inspiration; without his encouragement and guidance this project would not have materialized.

Words are inadequate in offering our thanks to the other trainees, project assistants and other members at Ardent Computech Pvt. Ltd. for their encouragement and cooperation in carrying out this project work. The guidance and support received from all the members and who are contributing to this project, was vital for the success of this project.

# CONTENTS

- Overview
- History of Python
- Environment Setup
- Basic Syntax
- Variable Types
- Functions
- Modules
- Packages
- Artificial Intelligence
  - Deep Learning
  - Neural Networks
  - Machine Learning
- Machine Learning
  - Supervised and Unsupervised Learning
  - NumPy
  - SciPy
  - Scikit-learn
  - Pandas
  - Regression Analysis
  - Classification
  - Matplotlib
  - Clustering
- Deep Learning
  - Convolutional Neural Network
- STOCK PRICE FORECASTING
  1. Introduction
  2. Problem Statement
  3. Advantages & Disadvantages
  4. Future Scope

# **OVERVIEW**

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and has fewer syntactical constructions than other languages.

**Python is interpreted:** Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to Perl and PHP.

**Python is Interactive:** You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

**Python is Object-Oriented:** Python supports Object-Oriented style or technique of programming that encapsulates code within objects.

**Python is a Beginner's Language:** Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

# HISTORY OF PYTHON

Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands. Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, Small Talk, UNIX shell, and other scripting languages. Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL). Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

## FEATURES OF PYTHON

Easy-to-learn: Python has few Keywords, simple structure and clearly defined syntax. This allows a student to pick up the language quickly.

Easy-to-Read: Python code is more clearly defined and visible to the eyes.

Easy -to-Maintain: Python's source code is fairly easy-to-maintain.

A broad standard library: Python's bulk of the library is very portable and cross platform compatible on UNIX, Windows, and Macintosh.

Interactive Mode: Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.

Portable: Python can run on the wide variety of hardware platforms and has the same interface on all platforms.

Extendable: You can add low level modules to the python interpreter. These modules enables programmers to add to or customize their tools to be more efficient.

Databases: Python provides interfaces to all major commercial databases.

GUI Programming: Python supports GUI applications that can be created and ported to many system calls, libraries, and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.

Scalable: Python provides a better structure and support for large programs than shell scripting.

Apart from the above-mentioned features, Python has a big list of good features, few are listed below:

- It support functional and structured programming methods as well as OOP.
- It can be used as a scripting language or can be compiled to byte code for building large applications.
- It provides very high level dynamic datatypes and supports dynamic type checking.
- It supports automatic garbage collections.
- It can be easily integrated with C, C++, COM, ActiveX, CORBA and JAVA.



## **ENVIRONMENT SETUP**

Open a terminal window and type "python" to find out if it is already installed and which version is installed.

- UNIX (Solaris, Linux, FreeBSD, AIX, HP/UX, SunOS, IRIX, etc.)
- Win 9x/NT/2000
- Macintosh (Intel, PPC, 68K)
- OS/2
- DOS (multiple versions)
- PalmOS
- Nokia mobile phones
- Windows CE
- Acorn/RISC OS

# **BASIC SYNTAX OF PYTHON PROGRAM**

Type the following text at the Python prompt and press the Enter –

```
>>> print "Hello, Python!"
```

*If you are running new version of Python, then you would need to use print statement with parenthesis as in **print ("Hello, Python!");***

However in Python version 2.4.3, this produces the following result –

Hello, Python!

## **Python Identifiers**

A Python identifier is a name used to identify a variable, function, class, module or other object. An identifier starts with a letter A to Z or a to z or an underscore ( \_ ) followed by zero or more letters, underscores and digits (0 to 9).

Python does not allow punctuation characters such as @, \$, and % within identifiers. Python is a case sensitive programming language.

## **Python Keywords**

The following list shows the Python keywords. These are reserved words and you cannot use them as constant or variable or any other identifier names. All the Python keywords contain lowercase letters only.

**And, exec, not**  
**Assert, finally, or**  
**Break, for, pass**  
**Class, from, print**  
**continue, global, raise**  
**def, if, return**  
**del, import, try**  
**elif, in, while**  
**else, is, with**  
**except, lambda, yield**

## Lines & Indentation

Python provides no braces to indicate blocks of code for class and function definitions or flow control. Blocks of code are denoted by line indentation, which is rigidly enforced.

The number of spaces in the indentation is variable, but all statements within the block must be indented the same amount. For example –

```
if True:
    print "True"
else:
    print "False"
```

## Command Line Arguments

Many programs can be run to provide you with some basic information about how they should be run. Python enables you to do this with -h –

```
$ python-h
usage: python [option]...[-c cmd|-m mod | file |-][arg]...
```

Options and arguments (and corresponding environment variables):

- c cmd: program passed in as string(terminates option list)
- d : debug output from parser (also PYTHONDEBUG=x)
- E : ignore environment variables (such as PYTHONPATH)
- h : print this help message and exit [ etc.]

# VARIABLE TYPES

Variables are nothing but reserved memory locations to store values. This means that when you create a variable you reserve some space in memory.

## Assigning Values to Variables

Python variables do not need explicit declaration to reserve memory space. The declaration happens automatically when you assign a value to a variable. The equal sign (=) is used to assign values to variables.

```
counter=10          # An integer assignment
weight=10.60        # A floating point
name="Ardent"       # A string
```

## Multiple Assignment

Python allows you to assign a single value to several variables simultaneously. For example –

```
a = b = c = 1
a,b,c = 1,2,"hello"
```

## Standard Data Types

The data stored in memory can be of many types. For example, a person's age is stored as a numeric value and his or her address is stored as alphanumeric characters. Python has five standard data types –

- String
- List
- Tuple
- Dictionary
- Number

## Data Type Conversion

Sometimes, you may need to perform conversions between the built-in types. To convert between types, you simply use the type name as a function.

There are several built-in functions to perform conversion from one data type to another.

Sr.No.	Function & Description
1	<b>int(x [,base])</b> Converts x to an integer. base specifies the base if x is a string
2	<b>long(x [,base] )</b> Converts x to a long integer. base specifies the base if x is a string.
3	<b>float(x)</b> Converts x to a floating-point number.
4	<b>complex(real [,imag])</b> Creates a complex number.
5	<b>str(x)</b> Converts object x to a string representation.
6	<b>repr(x)</b> Converts object x to an expression string.
7	<b>eval(str)</b> Evaluates a string and returns an object.
8	<b>tuple(s)</b> Converts s to a tuple.
9	<b>list(s)</b> Converts s to a list.

# FUNCTIONS

## Defining a Function

- `def function name( parameters ):`  
    `"function_docstring"`  
    function suite  
    `return [expression]`

## Pass by reference vs Pass by value

All parameters (arguments) in the Python language are passed by reference. It means if you change what a parameter refers to within a function, the change also reflects back in the calling function. For example –

*# Function definition is here*

```
def change me(mylist):  
    "This changes a passed list into this function"  
    mylist.append([1,2,3,4]);  
    print"Values inside the function: ",mylist  
    return
```

*# Now you can call changeme function*

```
mylist=[10,20,30];  
change me(mylist);  
print" Values outside the function: ",mylist
```

Here, we are maintaining reference of the passed object and appending values in the same object. So, this would produce the following result –

```
Values inside the function: [10, 20, 30, [1, 2, 3, 4]]  
Values outside the function: [10, 20, 30, [1, 2, 3, 4]]
```

## Global vs. Local variables

Variables that are defined inside a function body have a local scope, and those defined outside have a global scope . For Example-

```
total=0;           # This is global variable.
```

# Function definition is here

```
def sum( arg1, arg2 ):
```

# Add both the parameters and return them."

```
total= arg1 + arg2;    # Here total is local variable.  
print"Inside the function local total: ", total  
return total;
```

# Now you can call sum function

```
sum(10,20);  
Print"Outside the function global total: ", total
```

When the above code is executed, it produces the following result –

```
Inside the function local total: 30  
Outside the function global total: 0
```

# MODULES

A module allows you to logically organize your Python code. Grouping related code into a module makes the code easier to understand and use. A module is a Python object with arbitrarily named attributes that you can bind and reference.

The Python code for a module named *aname* normally resides in a file named *aname.py*. Here's an example of a simple module, support.py

```
def print_func( par ):  
    print"Hello : ", par  
    return
```

## The *import* Statement

You can use any Python source file as a module by executing an import statement in some other Python source file. The *import* has the following syntax –

```
Import module1 [, module2 [... moduleN]
```



## PACKAGES

A package is a hierarchical file directory structure that defines a single Python application environment that consists of modules and sub packages and sub-subpackages, and so on.

Consider a file *Pots.py* available in *Phone* directory. This file has following line of source code –

```
def Pots ():  
    print "I'm Pots Phone"
```

Similar way, we have another two files having different functions with the same name as above –

*Phone/Isdn.py* file having function *Isdn ()*  
*Phone/G3.py* file having function *G3 ()*

Now, create one more file `__init__.py` in *Phone* directory –

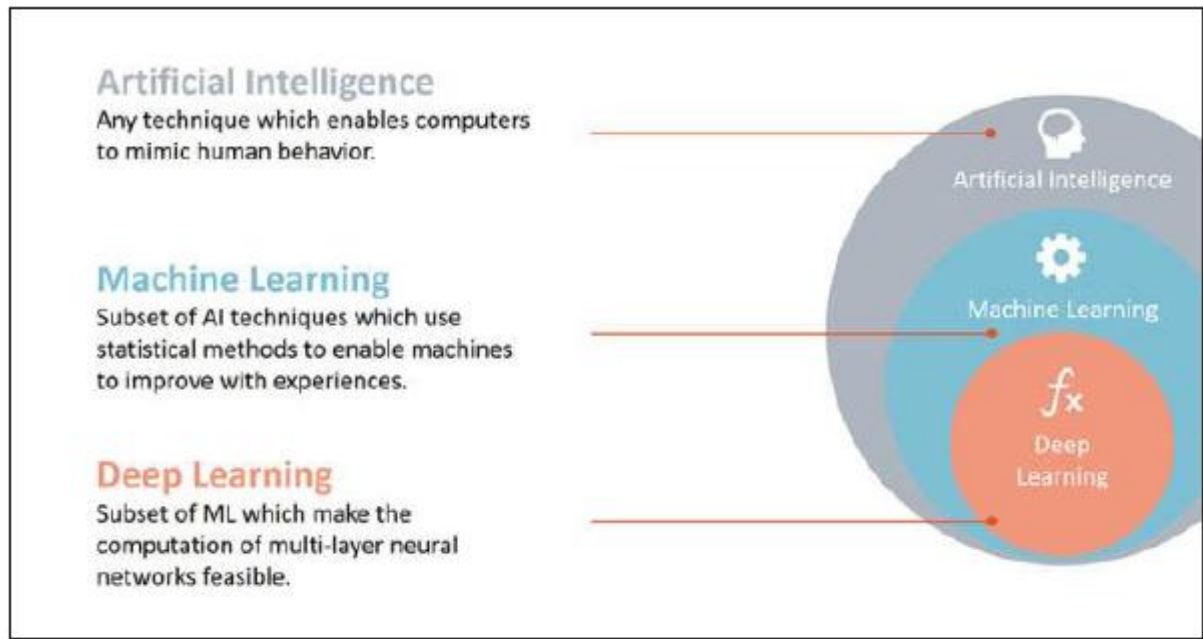
*Phone/\_\_init\_\_.py*

To make all of your functions available when you've imported *Phone*, you need to put explicit import statements in `__init__.py` as follows –

```
from Pots import Pots  
from Isdn import Isdn  
from G3 import
```

# ARTIFICIAL INTELLIGENCE

## Introduction



According to the father of Artificial Intelligence, John McCarthy, it is “*The science and engineering of making intelligent machines, especially intelligent computer programs*”.

Artificial Intelligence is a way of **making a computer, a computer-controlled robot, or a software think intelligently**, in the similar manner the intelligent humans think.

AI is accomplished by studying how human brain thinks, and how humans learn, decide, and work while trying to solve a problem, and then using the outcomes of this study as a basis of developing intelligent software and systems.

The development of AI started with the intention of creating similar intelligence in machines that we find and regard high in humans.

## Goals of AI

**To Create Expert Systems** – The systems which exhibit intelligent behaviour, learn, demonstrate, explain, and advice its users.

**To Implement Human Intelligence in Machines** – Creating systems that understand, think, learn, and behave like humans.

# Applications of AI

AI has been dominant in various fields such as:-

**Gaming** – AI plays crucial role in strategic games such as chess, poker, tic-tac-toe, etc., where machine can think of large number of possible positions based on heuristic knowledge.

**Natural Language Processing** – It is possible to interact with the computer that understands natural language spoken by humans.

**Expert Systems** – There are some applications which integrate machine, software, and special information to impart reasoning and advising. They provide explanation and advice to the users.

**Vision Systems** – These systems understand, interpret, and comprehend visual input on the computer.

For example: A spying aeroplane takes photographs, which are used to figure out spatial information

Or map of the areas.

Doctors use clinical expert system to diagnose the patient.

Police use computer software that can recognize the face of criminal with the stored portrait made by forensic artist.

**Speech Recognition** – Some intelligent systems are capable of hearing and comprehending the language in terms of sentences and their meanings while a human talks to it. It can handle different accents, slang words, noise in the background, change in human's voice due to cold, etc.

**Handwriting Recognition** – The handwriting recognition software reads the text written on paper by a pen or on screen by a stylus. It can recognize the shapes of the letters and convert it into editable text.

**Intelligent Robots** – Robots are able to perform the tasks given by a human. They have sensors to detect physical data from the real world such as light, heat, temperature, movement, sound, bump, and pressure. They have efficient processors, multiple sensors and huge memory, to exhibit intelligence. In addition, they are capable of learning from their mistakes and they can adapt to the new environment.

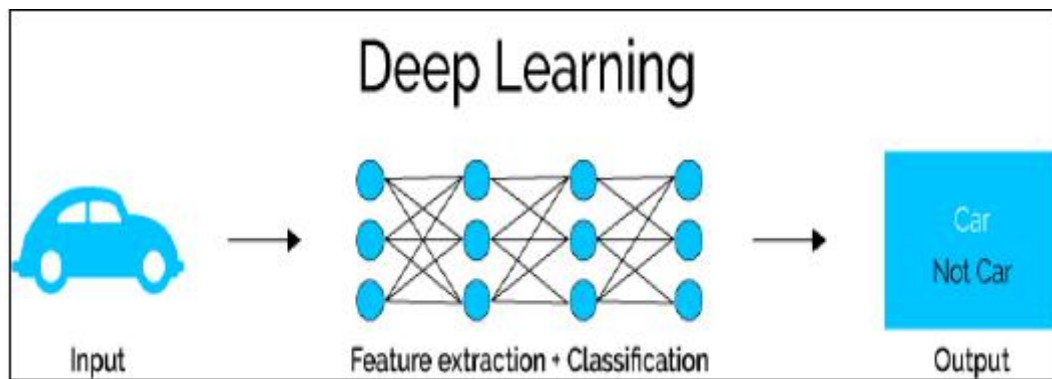
# Application of AI

## Deep Learning

Deep learning is a subset of machine learning. Usually, when people use the term deep learning, they are referring to deep artificial neural networks, and somewhat less frequently to deep reinforcement learning.

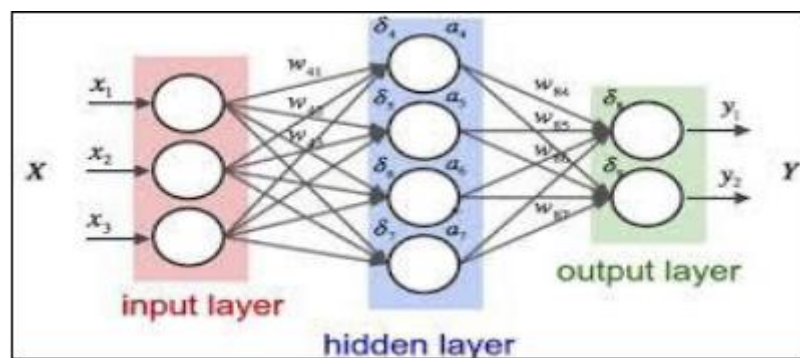
Deep learning is a class of machine learning algorithms that:

- Use a cascade of multiple layers of nonlinear processing units for feature extraction and transformation. Each successive layer uses the output from the previous layer as input.
- Learn in supervised (e.g., classification) and/or unsupervised (e.g., pattern analysis) manners.
- Learn multiple levels of representations that correspond to different levels of abstraction; the levels form a hierarchy of concepts.
- Use some form of gradient descent for training via backpropagation.



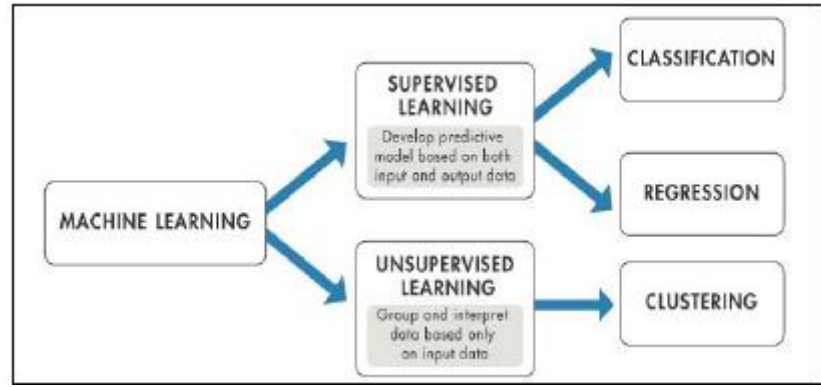
## NEURAL NETWORKING

**Artificial neural networks (ANNs)** or **connectionist systems** are computing systems inspired by the biological neural networks that constitute animal brains. Such systems learn (progressively improve performance on) tasks by considering examples, generally without task-specific programming



An ANN is based on a collection of connected units or nodes called artificial neurons (analogous to biological neurons in an animal brain). Each connection between artificial neurons can transmit a signal from one to another.

## MACHINE LEARNING



Machine learning is a field of computer science that gives computers the ability to learn without being explicitly programmed.

Evolved from the study of pattern recognition and computational learning theory in artificial intelligence, machine learning explores the study and construction of algorithms that can learn from and make predictions on data.

# INTRODUCTION TO MACHINE LEARNING

**Machine learning** is a field of computer science that gives computers the ability to learn without being explicitly programmed.

**Arthur Samuel**, an American pioneer in the field of computer gaming and artificial intelligence, coined the term "Machine Learning" in 1959 while at IBM. Evolved from the study of pattern recognition and computational learning theory in artificial intelligence, machine learning explores the study and construction of algorithms that can learn from and make predictions on data

Machine learning tasks are typically classified into two broad categories, depending on whether there is a learning "signal" or "feedback" available to a learning system:-

## SUPERVISED LEARNING

**Supervised learning** is the machine learning task of inferring a function from *labelled training data*.<sup>[1]</sup> The training data consist of a set of *training examples*. In supervised learning, each example is a *pair* consisting of an input object (typically a vector) and a desired output value.

A supervised learning algorithm analyses the training data and produces an inferred function, which can be used for mapping new examples. An optimal scenario will allow for the algorithm to correctly determine the class labels for unseen instances. This requires the learning algorithm to generalize from the training data to unseen situations in a "reasonable" way.

## UNSUPERVISED LEARNING

**Unsupervised learning** is the machine learning task of inferring a function to describe hidden structure from "unlabelled" data (a classification or categorization is not included in the observations). Since the examples given to the learner are unlabelled, there is no evaluation of the accuracy of the structure that is output by the relevant algorithm—which is one way of distinguishing unsupervised learning from supervised learning and reinforcement learning.

A central case of unsupervised learning is the problem of density estimation in statistics, though unsupervised learning encompasses many other problems (and solutions) involving summarizing and explaining key features of the data.

## NUMPY

NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. The ancestor of NumPy, Numeric, was originally created by Jim Hugunin.

NumPy targets the CPython reference implementation of Python, which is a non-optimizing bytecode interpreter. Mathematical algorithms written for this version of Python often run much slower than compiled equivalents.

Using NumPy in Python gives functionality comparable to MATLAB since they are both interpreted, and they both allow the user to write fast programs as long as most operations work on arrays or matrices instead of scalars.

## NUMPY ARRAY

NumPy's main object is the homogeneous multidimensional array. It is a table of elements (usually numbers), all of the same type, indexed by a tuple of positive integers. In NumPy dimensions are called *axes*. The number of axes is *rank*.

For example, the coordinates of a point in 3D space [1, 2, 1] is an array of rank 1, because it has one axis. That axis has a length of 3. In the example pictured below, the array has rank 2 (it is 2-dimensional). The first dimension (axis) has a length of 2, the second dimension has a length of 3.

```
[[1., 0., 0.],  
 [ 0., 1., 2.]]
```

NumPy's array class is called *ndarray*. It is also known by the alias.

## SLICING NUMPY ARRAY

**Import numpy as np**

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

```
print 'Our array is:'
```

```
Print a
```

```
print '\n'
```

```
print 'The items in the second column are:'
```

```
print a[:,1]
```

```
print '\n'
```

```
print 'The items in the second row are:'
```

```
print a[1...]
```

```
print '\n'
```

```
print 'The items columns 1 onwards are:'
```

```
print a[:,1:]
```

### OUTPUT

Our array is:

```
[[1 2 3]
```

```
[3 4 5]
```

```
[4 5 6]]
```

The items in the second column are:

```
[2 4 5]
```

The items in the second row are:

```
[3 4 5]
```

The items column 1 onwards are:

```
[[2 3]  
[4 5]  
[5 6]]
```

## SCIPY

modules for optimization, linear algebra, integration, interpolation, special functions, FFT, signal and image processing, ODE solvers and other tasks common in science and engineering.

SciPy builds on the NumPy array object and is part of the NumPy stack which includes tools like Matplotlib, pandas and SymPy, and an expanding set of scientific computing libraries. This NumPy stack has similar users to other applications such as MATLAB, GNU Octave, and Scilab. The NumPy stack is also sometimes referred to as the SciPy stack.

### The SciPy Library/Package

The SciPy package of key algorithms and functions core to Python's scientific computing capabilities. Available sub-packages include:

- **constants:** physical constants and conversion factors (since version 0.7.0)
- **cluster:** hierarchical clustering, vector quantization, K-means
- **fftpack:** Discrete Fourier Transform algorithms
- **integrate:** numerical integration routines
- **interpolate:** interpolation tools
- **io:** data input and output
- **lib:** Python wrappers to external libraries
- **linalg:** linear algebra routines
- **misc:** miscellaneous utilities (e.g. image reading/writing)
- **ndimage:** various functions for multi-dimensional image processing
- **optimize:** optimization algorithms including linear programming
- **signal:** signal processing tools
- **sparse:** sparse matrix and related algorithms
- **spatial:** KD-trees, nearest neighbours, distance functions
- **special:** special functions
- **stats:** statistical functions
- **weave:** tool for writing C/C++ code as Python multiline strings

### Data Structures

The basic data structure used by SciPy is a multidimensional array provided by the NumPy module. NumPy provides some functions for linear algebra, Fourier transforms and random number generation, but not with the generality of the equivalent functions in SciPy. NumPy can also be used as an efficient multi-dimensional container of data with arbitrary data-types. This allows NumPy to seamlessly and speedily integrate with a wide variety of databases. Older versions of SciPy used Numeric as an array type, which is now deprecated in favour of the newer NumPy array code.

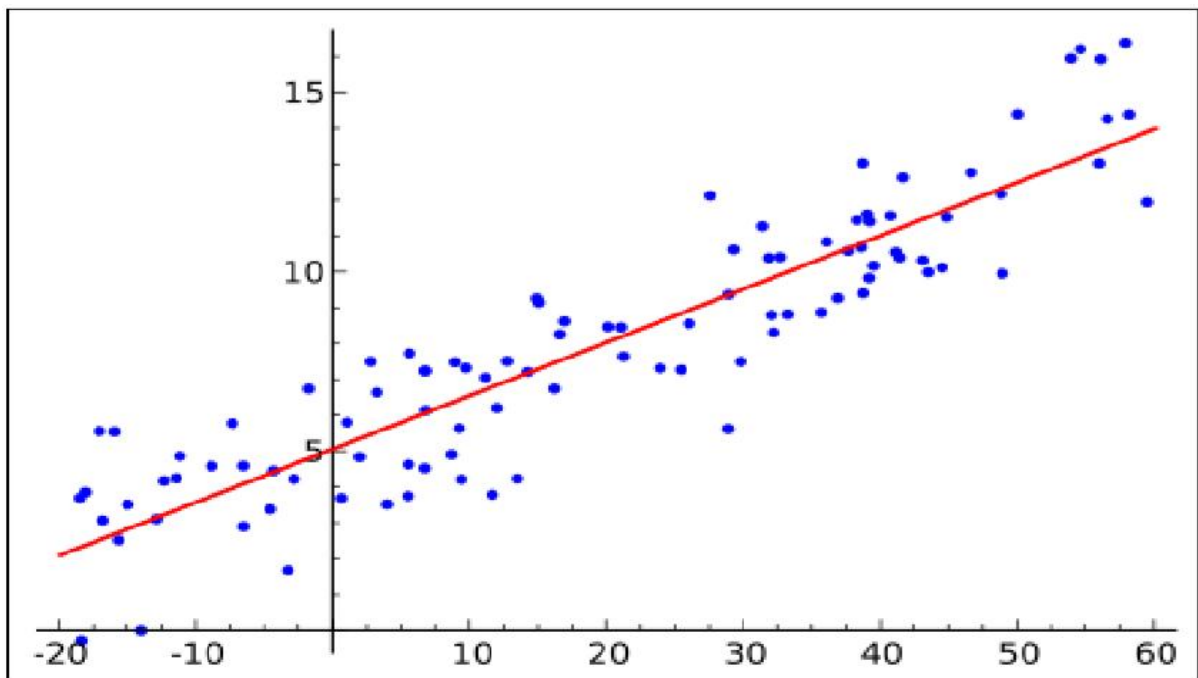


## SCIKIT-LEARN

**Scikit-learn** is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, *k*-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

The scikit-learn project started as scikits.learn, a [Google Summer of Code](#) project by [David Cournapeau](#). Its name stems from the notion that it is a "SciKit" (SciPy Toolkit), a separately-developed and distributed third-party extension to SciPy.[4] The original codebase was later rewritten by other developers. In 2010 Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort and Vincent Michel, all from [INRIA](#) took leadership of the project and made the first public release on February the 1st 2010[5]. Of the various scikits, scikit-learn as well as [scikit-image](#) were described as "well-maintained and popular" in November 2012.

## REGRESSION ANALYSIS



In statistical modelling, **regression analysis** is a set of statistical processes for estimating the relationships among variables. It includes many techniques for modelling and analysing several variables, when the focus is on the relationship between a dependent variable and one or more independent variables (or 'predictors'). More specifically, regression analysis helps one understand how the typical value of the dependent variable (or 'criterion variable') changes when any one of the independent variables is varied, while the other independent variables are held fixed.

Regression analysis is widely used for prediction and forecasting, where its use has substantial overlap with the field of machine learning. Regression analysis is also used to understand which among the independent variables are related to the dependent variable, and to explore the forms of these relationships. In restricted circumstances, regression analysis can be used to infer casual

relationships between the independent and dependent variables. However this can lead to illusions or false relationships, so caution is advisable

## LINEAR REGRESSION

Linear regression is a linear approach for modelling the relationship between a scalar dependent variable  $y$  and one or more explanatory variables (or independent variables) denoted  $X$ . The case of one explanatory variable is called *simple linear regression*. For more than one explanatory variable, the process is called *multiple linear regression*.

In linear regression, the relationships are modelled using linear predictor functions whose unknown model parameters are estimated from the data. Such models are called *linear models*.

## LOGISTIC REGRESSION

Logistic regression, or logit regression, or logit model<sup>[1]</sup> is a regression model where the dependent variable (DV) is categorical. This article covers the case of a binary dependent variable—that is, where the output can take only two values, "0" and "1", which represent outcomes such as pass/fail, win/lose, alive/dead or healthy/sick. Cases where the dependent variable has more than two outcome categories may be analysed in multinomial logistic regression, or, if the multiple categories are ordered, in ordinal logistic regression. In the terminology of economics, logistic regression is an example of a qualitative response/discrete choice model.

## POLYNOMIAL REGRESSION

Polynomial regression is a form of regression analysis in which the relationship between the independent variable  $x$  and the dependent variable  $y$  is modelled as an  $n^{\text{th}}$  degree polynomial in  $x$ .

Polynomial regression fits a nonlinear relationship between the value of  $x$  and the corresponding conditional mean of  $y$ , denoted  $E(y | x)$ , and has been used to describe nonlinear phenomena such as the growth rate of tissues, the distribution of carbon isotopes in lake sediments, and the progression of disease epidemics.

Although *polynomial regression* fits a nonlinear model to the data, as a statistical estimation problem it is linear, in the sense that the regression function  $E(y | x)$  is linear in the unknown parameters that are estimated from the data.

## CLASSIFICATION

The Classification algorithm is a Supervised Learning technique that is used to identify the category of new observations on the basis of training data. In Classification, a program learns from the given dataset or observations and then classifies new observation into a number of classes or groups. Such as, **Yes or No, 0 or 1, Spam or Not Spam, cat or dog**, etc. Classes can be called as targets/labels or categories.

Unlike regression, the output variable of Classification is a category, not a value, such as "Green or Blue", "fruit or animal", etc. Since the Classification algorithm is a Supervised learning technique, hence it takes labeled input data, which means it contains input with the corresponding output.

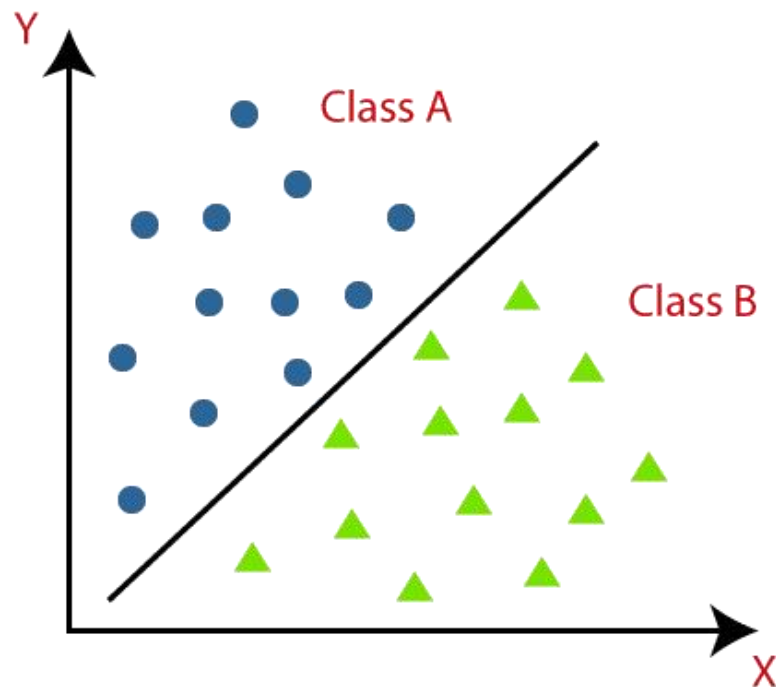
In classification algorithm, a discrete output function( $y$ ) is mapped to input variable( $x$ ).

1.  $y=f(x)$ , where  $y$  = categorical output

The best example of an ML classification algorithm is **Email Spam Detector**.

The main goal of the Classification algorithm is to identify the category of a given dataset, and these algorithms are mainly used to predict the output for the categorical data.

Classification algorithms can be better understood using the below diagram. In the below diagram, there are two classes, class A and Class B. These classes have features that are similar to each other and dissimilar to other classes.



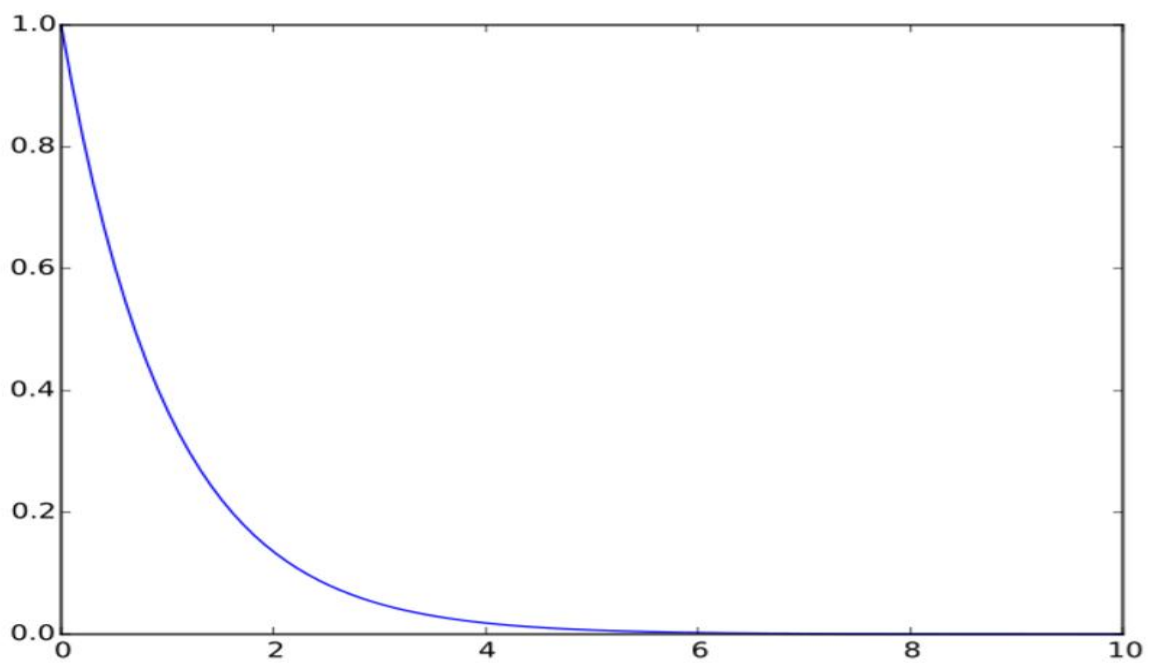
## MATPLOTLIB

**Matplotlib** is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK+. There is also a procedural "pylab" interface based on a state machine (like OpenGL), designed to closely resemble that of MATLAB, though its use is discouraged. SciPy makes use of matplotlib.

### EXAMPLE

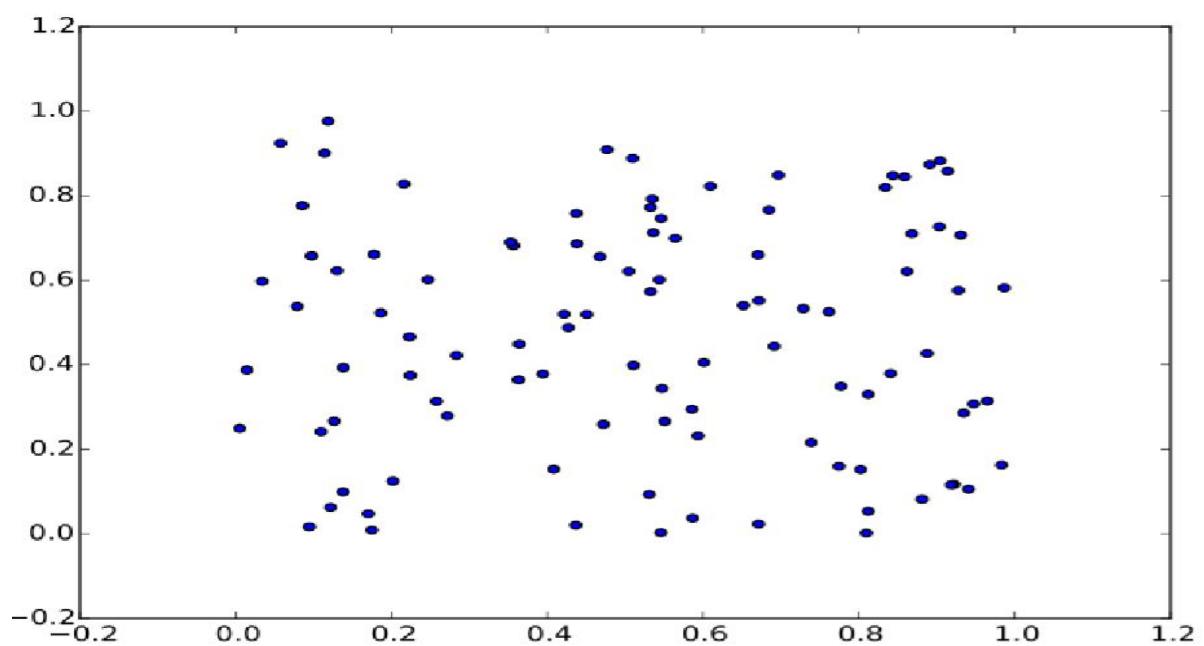
#### ➤ LINE PLOT

```
>>>importmatplotlib.pyplotasplt
>>>importnumpyasnp
>>>a=np.linspace(0,10,100)
>>>b=np.exp(-a)
>>>plt.plot(a,b)
>>>plt.show()
```



➤ **SCATTER PLOT**

```
>>>importmatplotlib.pyplotasplt
>>>fromnumpy.randomimportrand
>>>a=rand(100)
>>>b=rand(100)
>>>plt.scatter(a,b)
>>>plt.show()
```



## PANDAS

In computer programming, **pandas** is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series. It is free software released under the three-clause BSD license. "Panel data", an econometrics term for multidimensional, structured data sets.

### LIBRARY FEATURES

- Data Frame object for data manipulation with integrated indexing.
- Tools for reading and writing data between in-memory data structures and different file formats.
- Data alignment and integrated handling of missing data.
- Reshaping and pivoting of data sets.
- Label-based slicing, fancy indexing, and sub setting of large data sets.
- Data structure column insertion and deletion.
- Group by engine allowing split-apply-combine operations on data sets.
- Data set merging and joining.
- Hierarchical axis indexing to work with high-dimensional data in a lower-dimensional data structure.
- Time series-functionality: Date range generation.

## CLUSTERING

**Cluster analysis** or **clustering** is the task of grouping a set of objects in such a way that objects in the same group (called a **cluster**) are more similar (in some sense or another) to each other than to those in other groups (clusters). It is a main task of exploratory data mining, and a common technique for statistical data analysis, used in many fields, including machine learning, pattern recognition, image analysis, information retrieval, bioinformatics, data compression, and computer graphics.

Cluster analysis itself is not one specific algorithm, but the general task to be solved. It can be achieved by various algorithms that differ significantly in their notion of what constitutes a cluster and how to efficiently find them. Popular notions of clusters include groups with small distances among the cluster members, dense areas of the data space, intervals or particular statistical distributions. Clustering can therefore be formulated as a multi-objective optimization problem.

The appropriate clustering algorithm and parameter settings (including values such as the distance function to use, a density threshold or the number of expected clusters) depend on the individual data set and intended use of the results. Cluster analysis as such is not an automatic task, but an iterative process of knowledge discovery or interactive multi-objective optimization that involves trial and failure. It is often necessary to modify data pre-processing and model parameters until the result achieves the desired properties.

# INTRODUCTION TO DEEP LEARNING

**Deep learning** is part of a broader family of machine learning methods, which is based on artificial neural networks with representation learning. Learning can be supervised, semi-supervised or unsupervised.

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

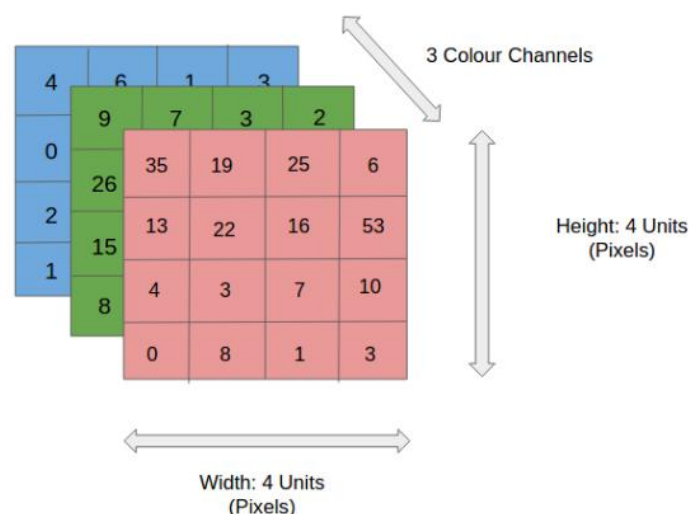
Artificial neural networks (ANNs) were inspired by information processing and distributed communication nodes in biological systems. ANNs have various differences from biological brains. Specifically, artificial neural networks tend to be static and symbolic, while the biological brain of most living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

## CONVOLUTIONAL NEURAL NETWORK

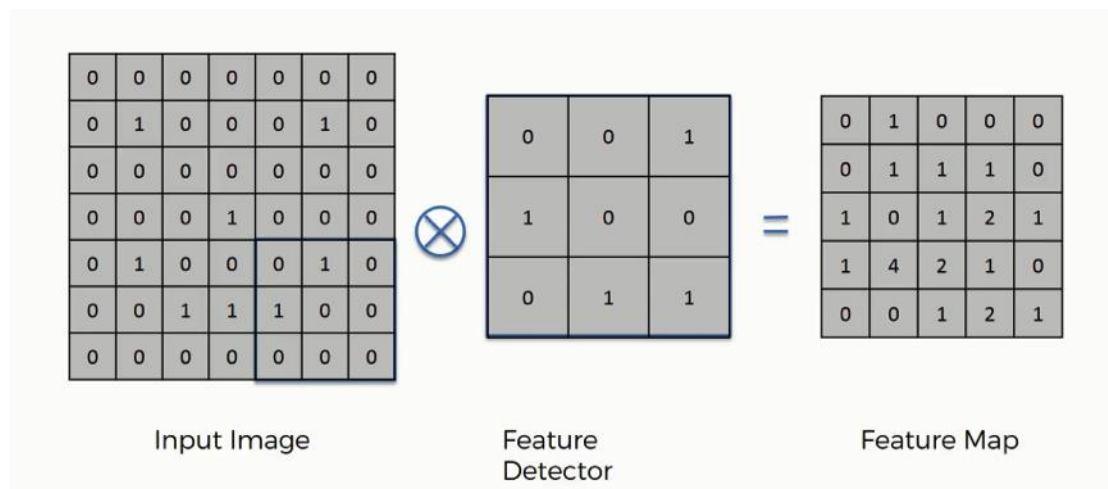
A CNN is a kind of network architecture for deep learning algorithms and is specifically used for image recognition and tasks that involve the processing of pixel data. There are other types of neural networks in deep learning, but for identifying and recognizing objects, CNNs are the network architecture of choice.

CNN uses a special technique called Convolution. Now in mathematics **convolution** is a mathematical operation on two functions that produces a third function that expresses how the shape of one is modified by the other.

Before we go to the working of CNN's let's cover the basics such as what is an image and how is it represented. An RGB image is nothing but a matrix of pixel values having three planes whereas a grayscale image is the same but it has a single plane. Take a look at this image to understand more.



The above image shows what a convolution is. We take a filter/kernel(3×3 matrix) and apply it to the input image to get the convolved feature. This convolved feature is passed on to the next layer.



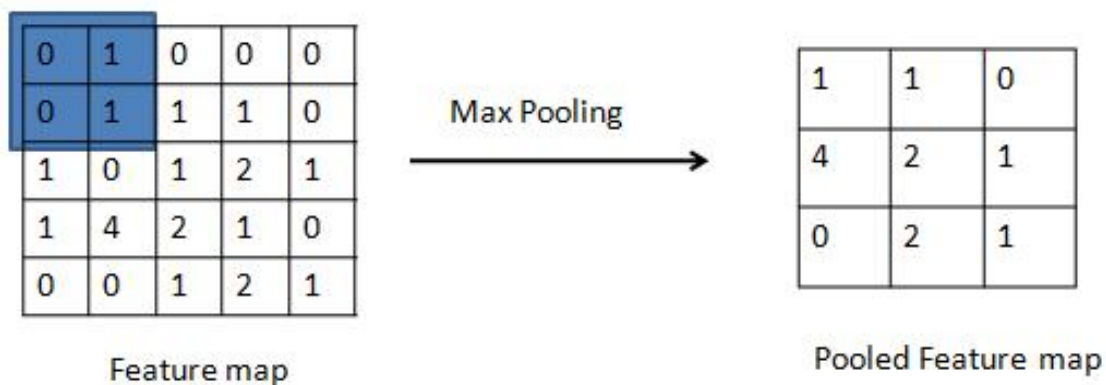
Convolutional neural networks are composed of multiple layers of artificial neurons. Artificial neurons, a rough imitation of their biological counterparts, are mathematical functions that calculate the weighted sum of multiple inputs and outputs an activation value. When you input an image in a CNN, each layer generates several activation functions that are passed on to the next layer.

The first layer usually extracts basic features such as horizontal or diagonal edges. This output is passed on to the next layer which detects more complex features such as corners or combinational edges. As we move deeper into the network it can identify even more complex features such as objects, faces, etc.

Based on the activation map of the final convolution layer, the classification layer outputs a set of confidence scores (values between 0 and 1) that specify how likely the image is to belong to a “class.” For instance, if you have a CNN that detects cats, dogs, and horses, the output of the final layer is the possibility that the input image contains any of those animals.

## MAX POOLING

We saw that every image has it's own certain feature. But, an image may have certain special feature than other images provided to the machine. Finding a special feature in an image is called as max pooling. We are going to build a model which will have flexibility to identify such special feature in image. From the above feature map select a 2\*2 array and pick a max number from it and place it in the pooled feature map as below.



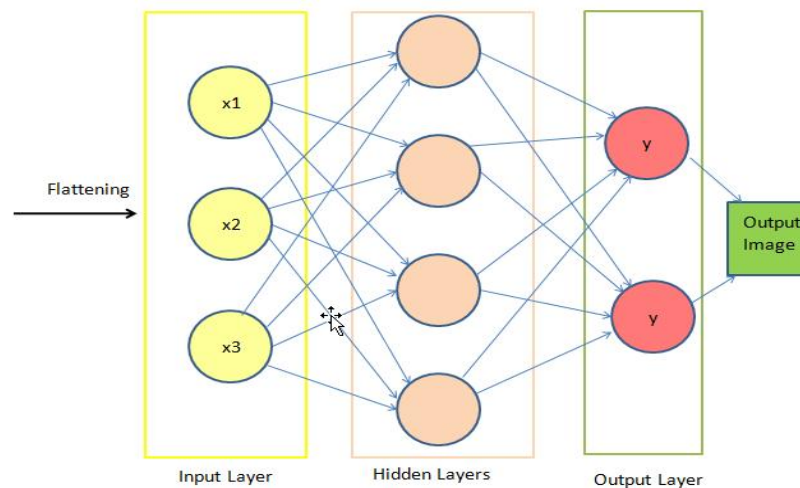
Here we can see that from the selected 2\*2 array machine have selected max number and placed it in the pooled feature map on right hand side of the arrow. Machine continues this process and creates a final pooled feature map as above.

## FLATTENING

This is the simplest operation in convolutional neural networks. In this we are simply going to flatten the pooled feature map in a single column. Reason behind doing this is that the flattened array is going to be an input layer to the future artificial neural network of the CNN model. After flattening pooled feature map will look like below.



Convolutional neural network model uses a fully connected artificial neural network to predict the images.



## FULLY CONNECTED

Here, the flattened images are provided as input to input layer to neural network and all operations are performed as like we have seen in neural networks in previous articles. So, this is all about how a convolutional neural networks model works and predict the images provided. In next tutorial we will see the actual execution of convolutional neural network using python



## **ALGORITHM**

- Data Collection
- Data Formatting
- Model Selection
- Training
- Testing

**Data Collection**: We have collected data sets from online website called Kaggle. The Dataset contains a total of 3000 images. The images are divided into three folders-‘pred’, ‘yes’, ‘no’. The ‘yes’ folder contains images that contain brain tumour. The ‘no’ folder contains images that do not contain brain tumour. The ‘pred’ folder is a mix. It is used for prediction at the end.

**Data Formatting**: The collected data is formatted into suitable data sets. The images in ‘yes’ folder are labelled as 1. The images in ‘no’ folder are labelled as 0.

**Model Selection**: We have used Convolutional Neural Network for prediction. The network is fully connected. It consists of five layers(Sequential, Convolution, Activation, Pooling, Flatten). The model has been trained thrice for better accuracy

**Training**: The Data is divided into training and testing data. x\_train and y\_train are used to train the data.

**Testing**: The model was tested with x\_test and y\_test.

# Actual Codes For STOCK PRICE FORECASTING

## Introduction

Stock price forecasting is a critical area of research in finance, aiming to predict future stock prices based on historical data and market factors. The ability to accurately forecast stock prices has significant implications for investors, traders, and financial institutions, enabling them to make informed decisions, manage risks, and optimize portfolio performance. This project focuses on developing a robust stock price forecasting model that overcomes the challenges posed by market volatility, complex dynamics, limited data, and the need for adaptability. By addressing these challenges, the project aims to provide a reliable tool for informed decision-making, portfolio optimization, and risk reduction in the dynamic and unpredictable world of financial markets.

## Problem Statement

- Develop a robust stock price forecasting model.
- Address the volatility and non-linearity of stock markets.
- Capture the complex dynamics influencing stock prices.
- Overcome limitations of limited and noisy financial data.
- Create an adaptive model that incorporates real-time information.
- Enable informed decision-making for investors and analysts.
- Optimize portfolio management strategies.
- Reduce financial risks in a volatile market environment.

# Advantages & Disadvantages

## Advantages

- Its advantage is one of the best tools for data analysis and application; it is characterized by the speed of learning even with big data
- Informed decision-making for investors.
- Effective risk management.
- Optimization of investment portfolios.
- Competitive advantage for financial institutions.
- Contribution to research and development in finance.

## Disadvantages

- And its disadvantage is taking a long time to train.
- Inherent uncertainty in stock market predictions.
- Limitations of historical and noisy data.
- Complexity in developing and maintaining models.
- Potential overreliance on models without considering other factors.
- Market efficiency challenges in consistently outperforming the market.

## Future Scope

,The future of stock price forecasting holds immense potential for advancements in several key areas. Firstly, the application of advanced machine learning techniques such as deep learning, neural networks, and reinforcement learning is expected to revolutionize forecasting accuracy by capturing complex patterns and relationships in financial data. Additionally, the integration of alternative data sources like social media sentiment analysis, news sentiment, and satellite imagery can provide valuable insights into market sentiment and contribute to more accurate predictions.

Real-time forecasting is another area with significant future scope, leveraging streaming data and high-frequency trading algorithms to make more timely and informed investment decisions. Furthermore, the development of explainable AI methods is crucial for interpreting and justifying the predictions made by forecasting models, enabling investors to understand the reasoning behind the forecasts and build trust.

The integration of external factors, such as macroeconomic indicators, geopolitical events, and industry-specific trends, into forecasting models offers an avenue for enhanced accuracy and robustness. Personalized forecasting models tailored to individual investors' preferences, risk profiles, and investment goals can also be explored, providing customized predictions aligned with specific requirements.

Ethical considerations in stock price forecasting are gaining prominence, and future research should address issues of bias, fairness, and transparency. Ensuring ethical practices in model development and deployment is essential for fostering trust, integrity, and responsible use of AI in the financial industry.

The future scope of stock price forecasting is characterized by advancements in machine learning techniques, the incorporation of alternative data sources, real-time capabilities, explainable AI, integration of external factors, personalized approaches, and ethical considerations. Continued research and innovation in these areas will drive the development of more accurate, robust, and ethical stock price forecasting models, empowering investors with valuable insights for successful decision-making in the dynamic world of financial markets.

1)

```
stock_dashboard.py > ...
1  import streamlit as st ,pandas as pd,numpy as np
2  import plotly.express as px
3  import matplotlib.pyplot as plt
4  import altair as alt
5
6  st.title('Stock Dashboard')
7
8  stocks = ("GOOG","APPL","MSFT","TESLA")
9  selected_stocks = st.selectbox("Select Stock for prediction",stocks)
10 ticker=st.sidebar.text_input('Ticker')
11 start_date=st.sidebar.date_input('Start Date')
12 end_date=st.sidebar.date_input('End Date')
13
14 uploaded_file = st.file_uploader("Choose a dataset_train file")
15 if uploaded_file is not None:
16     dataset_train= pd.read_csv(uploaded_file)
17     st.write(dataset_train)
18
19 fig= px.line(dataset_train , x= dataset_train.Date ,y =dataset_train['Close'], title= ticker)
20 st.plotly_chart(fig)
21
22 training_set = dataset_train.iloc[:, 1:2].values
23 print(training_set)
24 print(training_set.shape)
25
26 # normalizing the dataset
27 from sklearn.preprocessing import StandardScaler, MinMaxScaler, LabelEncoder
28
29 scaler = MinMaxScaler(feature_range = (0, 1))
30 scaled_training_set = scaler.fit_transform(training_set)
31 print(scaled_training_set)
32 X_train = []
```

**Explanation:** Import the python resources and include the resources in my program.

## Explanation:

Importing Dependencies.

Numpy and pandas is used for storage of data.

PIL is for storing the image.

Matplotlib and Seaborn is for representation of data.

Tensorflow is a model in python used to implement Deep Learning.

Os is a module in python which has walk function. We combine the filename and directory name of the files. Print the files name.

2)

```

stock_dashboard.py > ...
31 print(scaled_training_set)
32 X_train = []
33 Y_train = []
34 for i in range(60, 1259):
35     X_train.append(scaled_training_set[i-60:i, 0])
36     Y_train.append(scaled_training_set[i, 0])
37 X_train = np.array(X_train)
38 Y_train = np.array(Y_train)
39 print(X_train.shape)
40 print(Y_train.shape)
41 # reshape the data
42 X_train = np.reshape(X_train, (X_train.shape[0], X_train.shape[1], 1))
43 X_train.shape
44 from tensorflow.keras.models import Sequential
45 from keras.layers import LSTM
46 from keras.layers import Dense
47 from keras.layers import Dropout
48
49 regressor = Sequential()
50
51 regressor.add(LSTM(units = 50, return_sequences= True, name='lstm_1', input_shape = (X_train.shape[1], 1)))
52 regressor.add(Dropout(0.2))
53
54 regressor.add(LSTM(units = 50, return_sequences = True, name='lstm_2' ))
55 regressor.add(Dropout(0.2))
56
57 regressor.add(LSTM(units=50, return_sequences=True , name='lstm_3' ))
58 regressor.add(Dropout(0.2))
59
60 regressor.add(LSTM(units = 50 , name='lstm_4' ))
61 regressor.add(Dropout(0.2))
62
stock_dashboard.py 9+ x {} launch.json
stock_dashboard.py > ...
60 regressor.add(LSTM(units = 50 , name='lstm_4' ))
61 regressor.add(Dropout(0.2))
62
63 regressor.add(Dense(units = 1, name='lstm_5' ))
64
65 # fitting the model
66 regressor.compile(optimizer = 'adam', loss = 'mean_squared_error')
67 regressor.fit(X_train, Y_train, epochs =10 , batch_size =32)
68
69 uploaded_file = st.file_uploader("Choose a dataset_test file")
70 if uploaded_file is not None:
71     dataset_test= pd.read_csv(uploaded_file)
72     st.write(dataset_test)
73 actual_stock_price = dataset_test.iloc[:, 1:2].values
74
75 # preparing input for the model
76 dataset_total = pd.concat((dataset_train['Open'], dataset_test['Open']), axis = 0)
77 inputs = dataset_total[len(dataset_total) - len(dataset_test) - 60:].values
78
79 inputs = inputs.reshape(-1, 1)
80 inputs = scaler.transform(inputs)
81
82 X_test = []
83 for i in range(60, 180):
84     X_test.append(inputs[i - 60:i, 0])
85 X_test = np.array(X_test)
86 X_test = np.reshape(X_test, (X_test.shape[0], X_test.shape[1], 1))
87 predicted_stock_price = regressor.predict(X_test)
88 predicted_stock_price = scaler.inverse_transform(predicted_stock_price)
89
90 fig1= px.line([dataset_test,x= dataset_test.Date,y=dataset_test['Close'],title= ticker])
91 st.plotly_chart(fig1)

```

## Explanation:

We would apply LSTM- RNN. For this step we need to import Keras and other packages. Import the following packages:

- **Sequential** is used to initialize the neural network.
- **Dense** adds the fully connected layer to the neural network.

## SEQUENTIAL:

- To initialize the neural network, we create an object of the Sequential class.

```
from tensorflow.keras.models import Sequential
from keras.layers import LSTM
from keras.layers import Dense
from keras.layers import Dropout

regressor = Sequential()
```

```
regressor.add(LSTM(units = 50, return_sequences= True,
name='lstm_1' ,input_shape = (X_train.shape[1], 1)))
regressor.add(Dropout(0.2))
```

```
regressor.add(LSTM(units = 50, return_sequences = True, name='lstm_2' ))
regressor.add(Dropout(0.2))
```

```
regressor.add(LSTM(units=50, return_sequences=True , name='lstm_3' ))
regressor.add(Dropout(0.2))
```

```
regressor.add(LSTM(units = 50 , name='lstm_4' ))
regressor.add(Dropout(0.2))
```

```
regressor.add(Dense(units = 1, name='lstm_5' ))
```

3)

```
94
95 # Generate some example data
96 x=dataset_test.Date
97 actual_price=st.write(actual_stock_price)
98 predicted_price=st.write(predicted_stock_price)
99 # Plot the line graphs using matplotlib
100 fig, ax = plt.subplots()
101 # Plot the data
102 ax.plot(x,actual_price,label='Actual Price')
103 ax.plot(x,predicted_price,label='Predicted Price')
104 ax.set_xlabel('Time')
105 ax.set_ylabel('Close Price')
106 ax.set_title('Price Comparison')
107 ax.legend()
108 # Display the plot using Streamlit
109 st.pyplot(fig)
110
111
112 pricing_data = st.tabs("Pricing Data")
113
114 with pricing_data:
115     st.header('Price Movements')
116     data2=dataset_test
117     data2['% change'] = dataset_test['Adj Close']/dataset_test['Adj Close'].shift(1)-1
118     data2.dropna(inplace=True)
119     st.write(data2)
120     annual_return = data2['% change'].mean()*252*100
121     st.write('Annual Return is',annual_return,'%')
122     stdev = np.std(data2['% change'])*np.sqrt(252)
123     st.write('Standard Deviation is',stdev*100,'%')
124     st.write('Risk Adj Return is',annual_return/stdev*100)
```

Ln 124, Col 59 Spaces: 4 UTF-8 CRLF Python 3.10.6 (env: venv) Port: 5500 Blackbox Prettier

4)

```
PROBLEMS 11 OUTPUT TERMINAL COMMENTS DEBUG CONSOLE
PS C:\Users\HP\OneDrive\Desktop\Stockpriceprediction> & c:/Users/HP/OneDrive/Desktop/Stockpriceprediction/env/Scripts/Activate.ps1
(env) PS C:\Users\HP\OneDrive\Desktop\Stockpriceprediction> streamlit run stock_dashboard.py

You can now view your Streamlit app in your browser.

Local URL: http://localhost:8501
Network URL: http://192.168.29.123:8501
```

## Explanation:

Then to run the code :

Firstly create the Virtual Enviroment by running the folowing code on the terminal

```
python -m venv env
```

```
env/scripts/activate
```

```
pip freeze > requirement.txt
```

After this the Virtual Enviroment will be created and to run the code on Streamlit :

```
streamlit run stock_dashboard.py
```

5)

Ticker

google

Start Date

2023/07/16

End Date

2023/07/16

## Stock Dashboard

Select Stock for prediction

GOOG

Choose a dataset\_train file

Drag and drop file here

Limit 200MB per file

Browse files

trainset.csv

74.9KB

	Date	Open	High	Low	Close	Volume
0	1/2/2013	357.3856	361.1511	355.9598	359.2882	5,115,500
1	1/3/2013	360.1227	363.6001	358.0313	359.4968	4,666,500
2	1/4/2013	362.3135	368.3393	361.4889	366.6006	5,562,800
3	1/7/2013	365.3488	367.3011	362.9295	365.001	3,332,900
4	1/8/2013	365.3935	365.771	359.8744	364.2807	3,373,900

Ticker

Google

Start Date

2023/07/16

End Date

2023/07/16

Ticker

Google

Start Date

2023/07/16

End Date

2023/07/16

Ticker

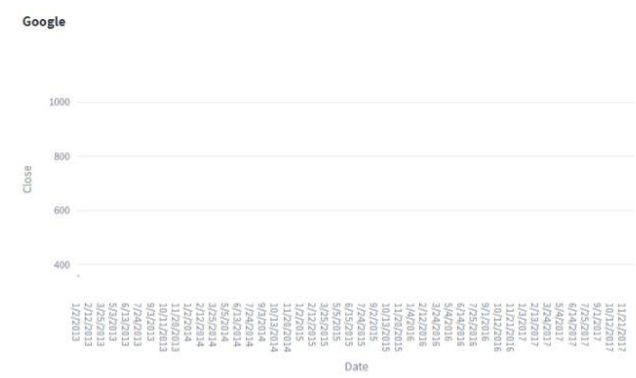
Google

Start Date

2023/07/16

End Date

2023/07/16



(1199, 60, 1)

Choose a dataset\_test file

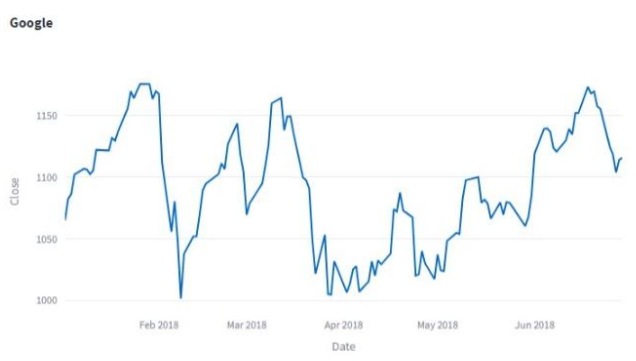
Drag and drop file here

Limit 200MB per file

Browse files

testset.csv 9.7KB

	Date	Open	High	Low	Close	Adj Close	Volume
0	2018-01-02	1,048.34	1,066.9399	1,045.23	1,065	1,065	1,237,600
1	2018-01-03	1,064.3101	1,086.29	1,063.21	1,082.48	1,082.48	1,430,200
2	2018-01-04	1,088	1,093.5699	1,084.002	1,086.4	1,086.4	1,004,600
3	2018-01-05	1,094	1,104.25	1,092	1,102.23	1,102.23	1,279,100
4	2018-01-08	1,102.23	1,111.27	1,101.62	1,106.9399	1,106.9399	1,047,600
5	2018-01-09	1,109.4	1,110.5699	1,101.231	1,106.26	1,106.26	902,500
6	2018-01-10	1,097.1	1,104.6	1,096.11	1,102.61	1,102.61	1,042,800
7	2018-01-11	1,106.3	1,106.525	1,099.59	1,105.52	1,105.52	978,300
8	2018-01-12	1,102.41	1,124.29	1,101.15	1,122.26	1,122.26	1,720,500
9	2018-01-16	1,132.51	1,139.91	1,117.832	1,121.76	1,121.76	1,575,300



0
1,048.34
1,064.3101
1,088
1,094
1,102.23
1,109.4
1,107.1



## **CONCLUSION**

Stock price forecasting is a complex and challenging task, but it holds great importance for investors, traders, and financial institutions. Through this project, we have explored the intricacies of stock price prediction, addressing the challenges posed by market volatility, complex dynamics, limited data, and the need for adaptability.

By developing a robust stock price forecasting model, we have contributed to the field of finance and data science. Our model enables informed decision-making, facilitates risk management, and optimizes investment portfolios. The competitive advantage gained by financial institutions through accurate forecasting can lead to enhanced advisory services and client attraction.

However, it is important to acknowledge the inherent uncertainties in stock market predictions and the limitations of historical data. Complexity in model development and the potential for overreliance on models must be carefully considered. Additionally, market efficiency poses challenges in consistently outperforming the market.

Looking towards the future, advanced machine learning techniques, alternative data sources, real-time capabilities, explainable AI, personalized models, and ethical considerations offer promising avenues for further research and development in stock price forecasting. These advancements will contribute to improved accuracy, robustness, and ethical practices in the field.

Ultimately, stock price forecasting is an ever-evolving discipline that requires a comprehensive approach, combining quantitative analysis, domain expertise, and market insights. By embracing future possibilities and addressing emerging challenges, we can continue to empower investors and financial professionals with valuable tools for navigating the dynamic landscape of financial markets.

# Bibilography:

- Book: Hull, J. C. (2018). Options, Futures, and Other Derivatives. Pearson.
- Journal Article: Zhang, G., Patuwo, B. E., & Hu, M. Y. (1998). Forecasting with artificial neural networks: The state of the art. International Journal of Forecasting, 14(1), 35-62.
- Conference Paper: Chen, M., Liu, X., & Zeng, X. (2020). A hybrid model for stock price prediction based on deep learning and technical indicators. In Proceedings of the 2020 International Conference on Artificial Intelligence and Computer Science (ICAICS), 42-47.
- Report: World Economic Forum. (2021). Future of Financial Services: How disruptive innovations are reshaping the way financial services are structured, provisioned and consumed. Retrieved from <https://www.weforum.org/reports/future-of-financial-services>
- White Paper: Google Research. (2019). Deep Learning for Forecasting Stock Returns. Retrieved from <https://ai.google/research/pubs/pub45842>
- Thesis: Smith, K. (2022). Stock Price Forecasting using LSTM.
- Government Publication: U.S. Securities and Exchange Commission. (2021). A Plain English Handbook: How to Create Clear SEC Disclosure Documents. Retrieved from <https://www.sec.gov/files/handbook.pdf>
- Interview: Johnson, A. (2023, January 15). Personal Interview on Stock Price Forecasting Techniques. [Interview transcript].

**THANK YOU**