# **Machine Learning**

## **What is Machine Learning?**

The use and development of computer systems that can learn and adapt without following explicit instructions, by using algorithms and statistical models to analyze and draw inferences from patterns in data.

## **When we use Machine Learning?**

* When you have a problem that requires many long lists of rules to find the solution. In this case, machine-learning techniques can simplify your code and improve performance.
* Very complex problems for which there is no solution with a traditional approach.
* Non- stable environments’: machine-learning software can adapt to new data.

## **Data Types from A Machine Learning Perspective**

### **Numerical Data**

* Numerical data is any data where data points are exact numbers. Statisticians also might call numerical data, quantitative data.
* Numerical data can be characterized by continuous or discrete data. Continuous data can assume any value within a range whereas discrete data has distinct values.

Diagram

Description automatically generated

### **Categorical Data**

* Categorical data represents characteristics, such as a hockey player’s position, team, hometown. Categorical data can take numerical values. For example, maybe we would use 1 for red and 2 for blue. But these numbers don’t have a mathematical meaning. That is, we can’t add them together or take the average.
* In the context of super classification, categorical data would be the class label. This would also be something like if a person is a man or woman, or property is residential or commercial.

### **Time-Series Data**

* Time series data is a sequence of numbers collected at regular intervals over some period. It is very important, especially in particular fields like finance. Time series data has a temporal value attached to it, so this would be something like a date or a timestamp that you can look for trends in time.

Chart, line chart

Description automatically generated

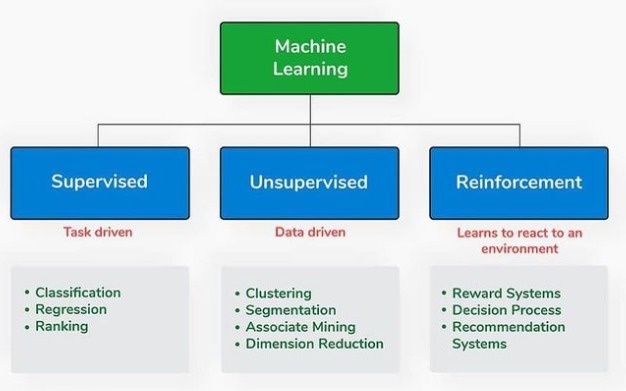
### **Text**

* Text data is basically just words. A lot of the time the first thing that you do with text is you turn it into numbers using some interesting functions like the bag of words formulation.

A picture containing text, accessory

Description automatically generated

## **Types of Systems of Machine Learning**



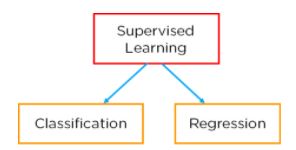
There are different types of machine-learning systems. We can divide them into

categories, depending on whether:

* They have been trained with humans or not
  + Supervised
  + Unsupervised
  + Semi-supervised
  + Reinforcement Learning
* If they can learn incrementally
* If they work simply by comparing new data points to find data points, or can detect new patterns in the data, and then will build a model.

## **Supervised & Unsupervised Learning**

### **Supervised Learning**



In this type of machine-learning system, the data that you feed into the algorithm, with the desired solution, are referred to as “**labels**.”

#### **Regression**

##### Regression Model

– relationship between dependent and independent variable and the output is continuous. (Predict a number)

* Linear regression-
* Decision Tree-
* Random Forest-
* Neural Network-

#### **Classification**

Classification is defined as the process of recognition, understanding, and grouping of objects and ideas into preset categories a.k.a “sub-populations.” With the help of these pre-categorized training datasets, classification in machine learning programs leverage a wide range of algorithms to classify future datasets into respective and relevant categories.

Classification algorithms used in machine learning utilize input training data for the purpose of predicting the likelihood or probability that the data that follows will fall into one of the predetermined categories. One of the most common applications of classification is for filtering emails into “spam” or “non-spam”, as used by today’s top email service providers.

In short, classification is a form of “pattern recognition,”. Here, classification algorithms applied to the training data find the same pattern (similar number sequences, words or sentiments, and the like) in future data sets.

We will explore classification algorithms in detail and discover how a text analysis software can perform actions like sentiment analysis - used for categorizing unstructured text by opinion polarity (positive, negative, neutral, and the like).

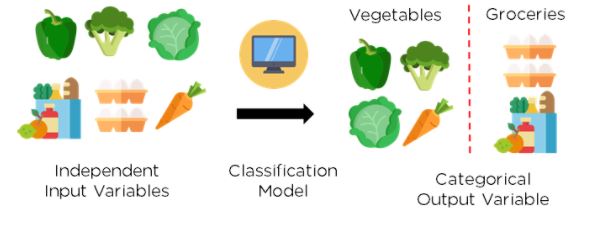


Figure 2: Classification of vegetables and groceries

##### **What is Classification Model/Algorithm?**

Based on training data, the Classification algorithm is a Supervised Learning technique used to categorize new observations. In classification, a program uses the dataset or observations provided to learn how to categorize new observations into various classes or groups. For instance, 0 or 1, red or blue, yes or no, spam or not spam, etc. Targets, labels, or categories can all be used to describe classes. The Classification algorithm uses labeled input data because it is a supervised learning technique and comprises input and output information. A discrete output function (y) is transferred to an input variable in the classification process (x).

In simple words, classification is a type of pattern recognition in which classification algorithms are performed on training data to discover the same pattern in new data sets.

Classification defined as the process of recognition, understanding, and grouping of objects and ideas into preset categories a.k.a “sub-populations.” With the help of these pre-categorized training datasets, classification in machine learning programs leverage a wide range of algorithms to classify future datasets into respective and relevant categories.

Classification Model/algorithm is a Supervised Learning technique used to categorize new observations. In classification, a program uses the dataset or observations provided to learn how to categorize new observations into various classes or groups. For instance, 0 or 1, red or blue, yes or no, spam or not spam, etc.

– the output is discrete. Predict a category

* + Logistics Regression – The output values can only be between 0 and 1
  + Support Vector Machine (SVM) – it is a supervise classification technique that

Chart, scatter chart

Description automatically generated

* + Naïve Bayes –
  + Decision Tree-
  + Random Forest-
  + Neural Network-
  + K Nearest Neighbors –

##### 4 different types of Classification Tasks:

* Binary Classification
* Multi-Class Classification
* Multi-Label Classification
* Imbalanced Classification

###### Binary Classification

Graphical user interface, application

Description automatically generated

Those classification jobs with only two class labels are referred to as binary classification.

Examples comprise -

* Prediction of conversion (buy or not).
* Churn forecast (churn or not).
* Detection of spam email (spam or not).

Binary classification problems often require two classes, one representing the normal state and the other representing the aberrant state.

For instance, the normal condition is "not spam," while the abnormal state is "spam." Another illustration is when a task involving a medical test has a normal condition of "cancer not identified" and an abnormal state of "cancer detected."

Class label 0 is given to the class in the normal state, whereas class label 1 is given to the class in the abnormal condition.

###### Multi-Class Classification

Application, table

Description automatically generated

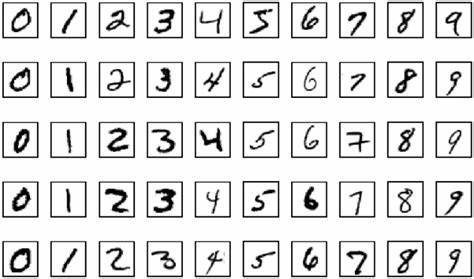
Multi-class labels are used in classification tasks referred to as multi-class classification.

Examples comprise -

* Categorization of faces.
* Classifying plant species.
* Character recognition using optical.

The multi-class classification does not have the idea of normal and abnormal outcomes, in contrast to binary classification. Instead, instances are grouped into one of several well-known classes.

In some cases, the number of class labels could be rather high. In a facial recognition system, for instance, a model might predict that a shot belongs to one of thousands or tens of thousands of faces.



Text translation models and other problems involving word prediction could be categorized as a particular case of multi-class classification. Each word in the sequence of words to be predicted requires a multi-class classification, where the vocabulary size determines the number of possible classes that may be predicted and may range from tens of thousands to hundreds of thousands of words.

###### Multi-Label Classification

Table

Description automatically generated

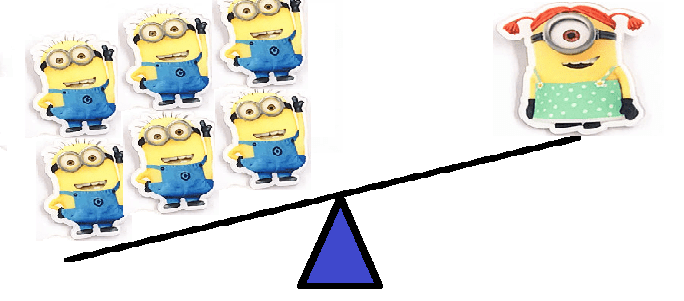
Multi-label classification problems are those that feature two or more class labels and allow for the prediction of one or more class labels for each example.

Think about the photo classification example. Here a model can predict the existence of many known things in a photo, such as “person”, “apple”, "bicycle," etc. A particular photo may have multiple objects in the scene.

This greatly contrasts with multi-class classification and binary classification, which anticipate a single class label for each occurrence.

###### Imbalanced Classification

Application

Description automatically generated

The term "imbalanced classification" describes classification jobs where the distribution of examples within each class is not equal.

Most of the training dataset's instances belong to the normal class, while a minority belong to the abnormal class, making imbalanced classification tasks binary classification tasks in general.

Examples comprise -

* Clinical diagnostic procedures
* Detection of outliers
* Fraud investigation

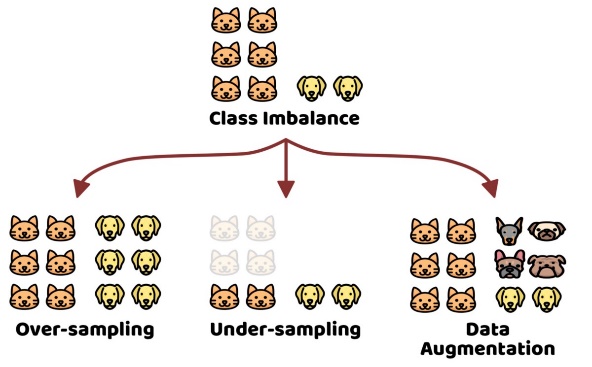
Although they could need unique methods, these issues are modeled as binary classification jobs.

By oversampling the minority class or undersampling the majority class, specialized strategies can be employed to alter the sample composition in the training dataset.

Examples comprise -

* SMOTE Oversampling
* Random Under sampling

It is possible to utilize specialized modeling techniques, like the cost-sensitive machine learning algorithms, that give the minority class more consideration when fitting the model to the training dataset.



### **Unsupervised Learning**

In this type of machine-learning system, you can guess that the data is unlabeled.

* Patterns from input data without references to labeled outcomes
  1. Clustering – Divide by similarity
     + K-means
     + Hierarchical
     + Mean shift
     + Density-based
  2. Association – Identify sequence.
  3. Dimensionality Reduction – a process of reducing the dimension of your feature set. Find hidden dependencies.
     + Feature elimination
     + Feature extraction

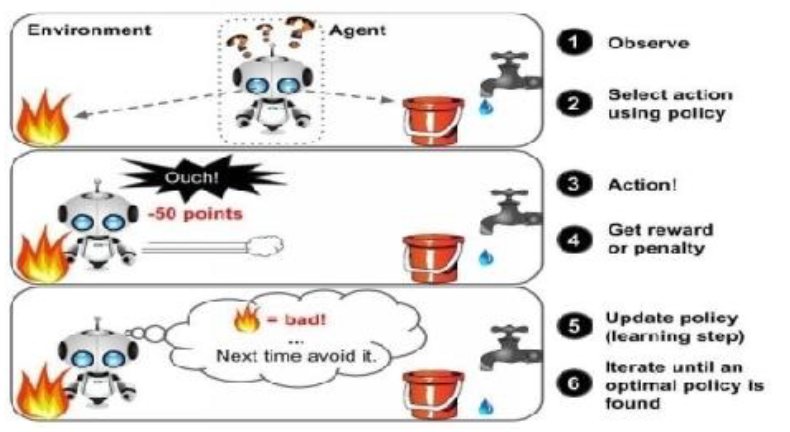
PRINCIPAL COMPONENT ANALYSIS (PCA) – popular method of Dimensionality Reduction

### Note

Supervised machine learning models make specific predictions or classifications based on labeled training data, while unsupervised machine learning models seek to cluster or otherwise find patterns in unlabeled data.

## **Reinforcement Learning**

Reinforcement learning is another type of machine-learning system. An agent “AI system” will observe the environment, perform given actions, and then receive t rewards in return. With this type, the agent must learn by itself. Ties are called a policy.



You can find this type of learning type in many robotics applications that learn how to walk.

### Reinforcement Learning Algorithms

1. **Q-Learning**

## **Auto ML**

Automated machine learning (AutoML) is the process of automating machine learning, from a raw dataset to a deployable model, for easily solving real-world problems. With AutoML, you can use machine learning without having to become an expert in the field.

## **Other Types of Machine Learning**

### **Batch Learning**

In this kind of machine-learning system, the system can’t learn incrementally: the system must obtain all the needed data. That means it will require many resources and a huge amount of time, so it’s always done offline. So, to work with this type of learning, the first thing to do is to train the system, and then launch it without any learning.

### **Online Learning**

This kind of learning is the opposite of batch learning. I mean that, here, the system can learn incrementally by providing the system with all the available data as instances (groups or individually), and then the system can learn on the fly.

### **Instance based Learning**

This is the simplest type of learning that you should learn by heart. By using this type of learning in our email program, it will flag all the emails that were flagged by users.

### **Model-based Learning**

There is another type of learning in which learning from examples allows construction to make predictions

## **Things to consider in Machine Learning Development**

### **Bad and Insufficient Quantity of Training Data**

Machine-learning systems are not like children, who can distinguish apples and oranges in all sorts of colors and shapes, but they require lot of data to work effectively, whether you're working with very simple programs and problems, or complex applications like image processing and speech recognition.

### **Poor-Quality Data**

If you're working with training data that is full of errors and outliers, this will make it very hard for the system to detect patterns, so it won't work properly. So, if you want your program to work well, you must spend more time cleaning up your training data.

### **Irrelevant Features**

The system will only be able to learn if the training data contains enough features and data that aren’t too irrelevant. The most important part of any ML project is to develop good features “of feature engineering”.

#### **Feature Engineering**

The process of feature engineering goes like this:

* Selection of features: selecting the most useful features.
* Extraction of features: combining existing features to provide more useful features.
* Creation of new features: creation of new features, based on data.

### **Testing**

If you'd like to make sure that your model is working well and that model can generalize with new cases, you can try out new cases with it by putting the model in the environment and then monitoring how it will perform. This is a good method, but if your model is inadequate, the user will complain.

You should divide your data into two sets, one set for training and the second one for testing, so that you can train your model using the first one and test it using the second. The generalization error is the rate of error by evaluation of your model on the test set. The value you get will tell you if your model is good enough, and if it will work properly.

If the error rate is low, the model is good and will perform properly. In contrast, if your rate is high, this means your model will perform badly and not work properly. My advice to you is to use 80% of the data for training and 20% for testing purposes, so that it’s very simple to test or evaluate a model.

### **Overfitting the Data**

If you're in a foreign country and someone steals something of yours, you might say that everyone is a thief. This is an overgeneralization, and, in machine learning, is called “overfitting”. This means that machines do the same thing: they can perform well when they're working with the training data, but they can't generalize them properly.

When does this occur?

Overfitting occurs when the model is very complex for the amount of training data given.

#### Solutions

To solve the overfitting problem, you should do the following:

* Gather more data for “training data”
* Reduce the noise level
* Select one with fewer parameters

### **Underfitting the Data**

From its name, underfitting is the opposite of overfitting, and you'll encounter this when the model is very simple to learn. For example, using the example of quality of life, real life is more complex than your model, so the predictions won't yield the same, even in the training examples.

#### Sulutions

To fix this problem:

* Select the most powerful model, which has many parameters.
* Feed the best features into your algorithm. Here, I'm referring to feature engineering.
* Reduce the constraints on your model.

## **Machine Learning lifecycle architecture diagram**

Figure 2 lists the ML lifecycle phases with data processing phase expanded into data collection and data preparation phases. These phases will be discussed in more detail in this section.


      Figure 4 includes the ML lifecycle from Figure 3 and expands its data processing phase 
        into sub-phases of collect data, and prepare data phases. Tje
        prepare data phase is further expanded into pre-process data, and engineer feature.
    

Figure 2: ML lifecycle with data processing sub-phases included

Figure 3 illustrates the ML lifecycle phases beyond problem framing phase in a zoomed-in version. The architecture diagrams in these figures show detail with expanded components that enable best practices that will be discussed in this paper. The data processing phase expands into data collection and data preparation. Data preparation expands into data preprocessing and feature engineering. Model development includes training, tuning, and evaluation. Deploy phase includes the staging environment for model validation for security and robustness. Monitoring is key in timely detection and mitigation of drifts. Feedback loops across the ML lifecycle phases are key enablers for monitoring. Feature stores (online/offline) provide consistent and reusable features across model development and deployment phases. The model registry enables the version control and lineage tracking for model and data components. This figure also emphasizes on the lineage tracking and its components that are discussed in this section in more detail.

The cloud agnostic architecture diagrams in this paper provide high-level best practices with the following assumptions:

* All presented concepts are cloud and technology agnostic.
* Solid black lines are indicative of process flow.
* Dashed color lines are indicative of input and output flow.
* Architecture diagram components are color coded for ease of communication across this document.


      Figure 5 includes a more detailed version of the ML lifecycle architecture diagram 
        and illustrates processes, technologies, and components that support many of the best practices in 
        this whitepaper.
    

Figure 3: ML lifecycle with detailed phases and expanded components

**ML lifecycle as shown in Figure 5 includes the following components:**

* Online/Offline feature store — Feature store reduces duplication and rerun of feature engineering code across teams and projects. An online store with low-latency retrieval capabilities is ideal for real-time inference. The offline store should maintain a history of feature values and is suited for training and batch scoring.
* Model registry — Model registry is a repository for storing ML model artifacts including trained model and related metadata (data, code, model). It enables lineage for ML models as it can act as a version control system.
* Performance feedback loop — Automates model performance evaluation tasks initiated from the model development to data processing phase.
* Model drift feedback loop — Automates model update re-training tasks initiated from the production deployment to data processing phase.
* Alarm manager — Alarm manager receives the alerts from the model monitoring system. It then runs actions by publishing notifications to services that can deliver alerts to target applications to handle them. The model update re-training pipeline is one such target application.
* Scheduler — A scheduler can initiate re-training at business defined intervals.
* Lineage tracker — The machine learning lineage tracking enables reproducible machine learning experiences. It enables re-creating the ML environment at a specific point-in-time, reflecting the versions of all resources and environments at that time.

### **Data Processing**

#### Data Collection

…

#### Pre-Processing

##### Data Cleaning

Reference: [Data Cleaning Steps. | Data Science and Machine Learning | Kaggle](https://www.kaggle.com/getting-started/250322)

1. Find the Dirt - determining what is wrong with your data
   * Are there rows with empty values? Entire columns with no data? Which data is missing and why?
   * Keep an eye out for the weird: are there impossible values? Like “date of birth: male”, “address: -1234”.
   * Is your data consistent? Why are the same product names written in uppercase and other times in camelCase?
2. Scrub the Dirt | Removing corrupted records/ unwanted values

Depending on the type of data dirt you’re facing, you’ll need different cleaning techniques.

Scrub the Dirt is broken down into eight parts:

* + 2.1 Missing Data | Handling missing values
* Sometimes you will have rows with missing values. Sometimes, almost entire columns will be empty.
* Start by spotting all the different disguises missing data wears. It appears in values such as 0, “0”, empty strings, “Not Applicable”, “NA”, “#NA”, None, NaN, NULL or Inf. Programmers before you might have put default values instead of missing data (“email@company.com”).

There are 3 main approaches to cleaning missing data:

* Drop rows and/or columns with missing data.
* Recode missing data into a different format. Numerical computations can break down with missing data. Recoding missing values into a different column saves the day. For example, the column “payment\_date” with empty rows can be recoded into a column “payed\_yet” with 0 for “no” and 1 for “yes”.
* Fill in missing values with “best guesses.” Use moving averages and backfilling to estimate the most probable values of data at that point. This is especially crucial for time-series analyses, where missing data can distort your conclusions.
  + 2.2 Outliers Contaminated
  + 2.3 Data Inconsistent
  + 2.4 Data Invalid
  + 2.5 Data Duplicate
  + 2.6 Data Type Issues
  + 2.7 Structural Errors

1. Rinse and Repeat

- Once cleaned, repeat steps 1 and 2.

##### Data Transform

* A chosen data is converted into a format that is ready to process

#### **Feature Selection / Feature Engineering**

1. Correlation Analysis
   1. Pearson’s correlation coefficient
   2. Spearman’s correlation coefficient
   3. F-Test
2. Chi-Square analysis
3. T-Test
4. ANOVA-Test
5. Information gain Analysis
6. Gain ratio

* Heat map visualization
* Hierarchical clustering methods
* Genetic Algorithm Method
* Cross-tabulation method

### **Model Development**

#### Train

--

#### Testing

--

#### Validation

-- Evaluation Criteria for Machine Learning Models

https://medium.com/analytics-vidhya/evaluation-criteria-for-machine-learning-models-ce04692cc3e3

https://riskspan.com/evaluating-supervised-and-unsupervised-learning-models/#:~:text=Model%20evaluation%20%28including%20evaluating%20supervised%20and%20unsupervised%20learning,or%20appropriately%20flagging%20credit%20card%20transactions%20as%20fraud.

## SUMMARY

* Machine learning: ML refers to making machines work better at some task, using given data.
* Machine learning comes in many types, such as supervised, batch, unsupervised, and online learning.
* To perform an ML project, you need to gather data in a training set, and then feed that set to a learning algorithm to get an output, “predictions”.
* If you want to get the right output, your system should use clear data, which is not too small and which does not have irrelevant features.

# **Deep Learning**

Deep learning is largely considered a *subset* of Machine Learning, and it is built on traditional artificial neural networks.

Image analysis or image processing is the most researched topic in deep learning. Image processing simply means extracting information from images and learning from images. This is an important aspect of computer vision. In deep learning, image processing is largely done using convolutional neural networks (CNNs) and can be generally divided into image classification, object detection, image segmentation, and so on.

Illustrates the differences between image classification, object detection, and image segmentation. Image classification is about identifying the image or classifying the image. Image classification can be typically done by using deep learning neural networks such as AlexNet, GoogLeNet, VGG, ResNet, MobileNet, and so on. Object detection is about identifying a particular object in the image. Object detection can be typically done by using deep learning neural networks such as region-based convolutional neural networks (R-CNNs), you only look once (YOLO), and so on. Image segmentation means dividing the image into different segments according to the content. Image segmentation can be typically done by using Detectron, Gluon, PixelLib libraries, and so on.



Figure 5.1: The differences between image classification, object detection, and image segmentation

## **Neural Networks**

* ANN
* CNN
* KNN

## **Image Classification**

Image classification is the simplest and most used image processing technique. Image classification allows you to identify the content of an image, for example, whether an image is a dog or a cat, a type of flower, cancer or noncancer, and so on. Image classification is done by two steps called training and inference, as shown in Figure 5.2. During training, you feed the deep learning neural networks with training images and targets (also called labels) and adjust the weights of the neural network until it maximizes the recognition rate, called accuracy. The more training images and the more complex the neural networks, the higher the training accuracy. After training, you can feed a query image to the network, and it will predict the result. The process of using a trained deep learning model to make predictions against previously unseen data is called inference.

For image classification, you can either use pre- trained models or custom trained models. Pre-trained models are trained on a data source such as the ImageNet dataset (http://www.image-net.org/) or the CIFAR-10 dataset (https://www.cs.toronto.edu/~kriz/cifar.html). ImageNet has 1,000 classes, and CIFAR-10 consists of 10 classes; the pre-trained model will therefore only recognize the 1,000 classes or 10 classes. You can also re-train the pre-trained models with your own datasets, which is called transfer learning.

Chart, scatter chart

Description automatically generated

Figure 5.2: Training and inference in image classification

Image classification has countless real-life applications. For example, take a picture of a skin mole; image classification can tell you whether it is benign or malignant. Or, take a picture of a flower or a plant; it can tell you what type of flower or plant it is. For tourists, it can tell you what building or what attraction is in your photo. For fashionistas, it can tell what brand of cloth or shoes the photo is about. In healthcare, it can also classify X-ray images, CT images, and MRI images. The potential is really endless!

### **Classification with Pre-trained Models**

There are several deep learning neural networks available. The easiest way to use them for image classification is through the TensorFlow and Keras libraries.

### **Classification with Custom Trained Models: Transfer Learning**

Although classifying images with a pre-trained model is useful, it can only classify the images that have been previously specified. For example, if a model has  
been trained on ImageNet, then the model can classify only the 1,000 types of  
images defined by ImageNet. If a model has been trained on CIFAR-10, then it  
can classify only 10 classes of images defined by CIFAR-10.

Therefore, if you want to classify new types of images, you can use the pre-  
trained deep learning neural network model as the starting point for a model on  
the second task of interest, for example, on your own image data. This is called  
transfer learning. Transfer learning is popular because it can train deep learning  
neural networks with comparatively little data. The potential of transfer learning is enormous and can really bring image classification to life.

# **ADVANCED PYTHON LIBRARY**

## Libraries for Getting Data

Data science starts with data. To do data analysis or modeling with Python, you need to first import your data. Data can be stored in different formats, but luckily the Python community has developed many packages for getting input data. Let’s see which Python libraries are the most popular for importing and preparing data.

### [csv](https://docs.python.org/3/library/csv.html)

CSV (Comma Separated Values) is a common format for storing tabular data as well as importing and exporting data. To **handle CSV files**, Python has a built-in csv module. For example, if you need to read data from a CSV file, you can use the csv.reader() function, which basically iterates through the rows of the CSV file. If you want to export data to a CSV format, the csv.writer() function can handle this.

### [json](https://docs.python.org/3/library/json.html)

JSON, or JavaScript Object Notation, is a standard format for storing and exchanging text data. Even though it was inspired by a subset of the JavaScript programming language, JSON is language-agnostic – you don’t need to know JavaScript to work with JSON files.

To **encode and decode JSON data**, Python has a built-in module called json. After importing the json module, you’ll be able to read JSON documents with the json.load() method or convert your data into JSON files with the json.dump() method.

### [openpyxl](https://openpyxl.readthedocs.io/en/stable/)

If your data is primarily stored in Excel, you’ll find the openpyxl library very helpful. It was born to **read and write Excel 2010 docs**. The library supports xlsx, xlsm, xltx, and xltm files. In contrast to the above packages, openpyxl is not built into Python; you’ll need to install it before you use it.

This library allows you to read Excel spreadsheets, import specific data from a particular sheet, append data to the existing spreadsheet, and create new spreadsheets with formulas, images, and charts.

### [Scrapy](https://scrapy.org/)

If the data you want to use is on the web, Python has several packages that’ll get it in a fast and simple way. Scrapy is a popular open-source library for **crawling web sites and extracting structured data**.

With Scrapy you can, for example, [SCRAPE TWITTER](https://towardsdatascience.com/hands-on-web-scraping-building-your-own-twitter-dataset-with-python-and-scrapy-8823fb7d0598) for tweets from a particular account or with specified hashtags. The result may include lots of information beyond the tweet itself; you may get a table with usernames, tweet times and texts, the number of likes, retweets, and replies, etc. Other than web scraping, Scrapy can also be used to extract data using APIs.

Its speed and flexibility make Scrapy a great tool for extracting structured data that can be further processed and used in various data science projects.

### [Beautiful Soup](https://www.crummy.com/software/BeautifulSoup/bs4/doc/)

Beautiful Soup is another popular library for getting data from the web. It was created to **extract useful information from HTML and XML files**, including those with invalid syntax and structure. The unusual name of this Python library refers to the fact that such poorly-marked-up pages are often called ‘tag soup’**.**

When you run an HTML document through Beautiful Soup, you get a BeautifulSoup object that represents the document as a nested data structure. Then you can easily navigate that data structure to get what you need, e.g. the page’s text, link URLs, specific headings, etc.

The flexibility of the Beautiful Soup library is remarkable. Check it out if you need to work with web data.

## **Libraries for Processing and Modeling Data**

After getting your data, you’ll need to clean and prepare it for analysis and modeling. Let’s review Python libraries that assist data scientists in preparing data and building and training machine learning models.

### NumPy

– is a well-known general-purpose array-processing package. An extensive collection of high complexity mathematical functions makes NumPy powerful to process large multi-dimensional arrays and matrices. NumPy is very useful for handling linear algebra, Fourier transforms, and random numbers. Other libraries like TensorFlow uses NumPy at the backend for manipulating tensors. With NumPy, you can define arbitrary data types and easily integrate with most databases. NumPy can also serve as an efficient multi-dimensional container for any generic data that is in any datatype.

NumPy is a fundamental Python library for data science. It is designed to perform numerical operations with n-dimensional arrays. Arrays store values of the same data type. The NumPy vectorization of arrays significantly enhances performance and accelerates the speed of computing operations.

With NumPy, you can do basic and advanced array operations (e.g. add, multiply, slice, reshape, index), generate random numbers, and perform linear algebra routines, Fourier transforms, and more.

### SciPy

- SciPy library offers modules for linear algebra, image optimization, integration interpolation, special functions, Fast Fourier transform, signal and image processing, Ordinary Differential Equation (ODE) solving, and other computational tasks in science and analytics. The underlying data structure used by SciPy is a multi-dimensional array provided by the NumPy module. SciPy depends on NumPy for the array manipulation subroutines. The SciPy library was built to work with NumPy arrays along with providing user-friendly and efficient numerical functions.

SciPy is a fundamental library for scientific computing. It’s built upon NumPy and leverages many of that library’s benefits for working with arrays.

With SciPy, you can perform scientific programming tasks such as calculus, ordinary differential equations, numerical integration, interpolation, optimization, linear algebra, and statistical computations.

### Scikit-learn

– Skikit-learn was built on top of two Python libraries – NumPy and SciPy and has become the most popular Python machine learning library for developing machine learning algorithms. Scikit-learn has a wide range of supervised and unsupervised learning algorithms that works on a consistent interface in Python. The library can also be used for data mining and data analysis. The main machine learning functions that the Scikit-learn library can handle are classification, regression, clustering, dimensionality reduction, model selection, and preprocessing.

**Feature selection** (<https://scikit-learn.org/stable/modules/feature_selection.html>)

A fundamental Python library for machine learning, scikit-learn focuses on modeling data after it has been cleaned and prepared (using libraries like NumPy and pandas). This is a very efficient tool for predictive data analysis. Furthermore, it is beginner-friendly, making machine learning with Python accessible to everybody.

With just a few lines of code, scikit-learn allows you to build and train machine learning models for regression, classification, clustering, dimensionality reduction, and more. It supports algorithms such as support vector machines (SVM), random forests, k-means, gradient boosting, and many others.

### Theano

- Theano is a python machine learning library that can act as an optimizing compiler for evaluating and manipulating mathematical expressions and matrix calculations. Built on NumPy, Theano exhibits a tight integration with NumPy and has a very similar interface. Theano can work on Graphics Processing Unit (GPU) and CPU. Working on GPU architecture yields faster results. Theano can perform data-intensive computations up to 140x faster on GPU than on a CPU. Theano can automatically avoid errors and bugs when dealing with logarithmic and exponential functions. Theano has built-in tools for unit-testing and validation, thereby avoiding bugs and problems.

### TensorFlow

- TensorFlow was developed for Google’s internal use by the Google Brain team. Its first release came in November 2015 under Apache License 2.0. TensorFlow is a popular computational framework for creating machine learning models. TensorFlow supports a variety of different toolkits for constructing models at varying levels of abstraction.

TensorFlow operates on multidimensional arrays or *tensors* represented as [tf.Tensor](https://www.tensorflow.org/api_docs/python/tf/Tensor) objects. TensorFlow is an end-to-end platform for machine learning. It supports the following:

* Multidimensional array based numeric computation (similar to [NumPy](https://numpy.org/).)
* GPU and distributed processing
* Automatic differentiation
* Model construction, training, and export
* And more

TensorFlow is another open-source library for developing and training machine learning models. Built by the Google Brain team, TensorFlow is a major competitor to PyTorch in the development of deep learning applications.

TensorFlow and PyTorch used to have some major differences, but they have now adopted many good features from each other. They are both excellent frameworks for building deep learning models. When you hear about breakthrough neural network architectures for object detection, facial recognition, language generation, or chatbots, they are very likely coded using either PyTorch or Tensorflow libraries.

### Keras

– Keras is an open-source library used for neural networks and machine learning. Keras can run on top of TensorFlow, Theano, Microsoft Cognitive Toolkit, R, or PlaidML. Keras also can run efficiently on CPU and GPU. Keras works with neural-network building blocks like layers, objectives, activation functions, and optimizers. Keras also have a bunch of features to work on images and text images that comes handy when writing Deep Neural Network code. Apart from the standard neural network, Keras supports convolutional and recurrent neural networks.

### PyTorch

- PyTorch has a range of tools and libraries that support computer vision, machine learning, and natural language processing. The PyTorch library is open-source and is based on the Torch library. The most significant advantage of PyTorch library is it’s ease of learning and using.

PyTorch can smoothly integrate with the python data science stack, including NumPy. You will hardly make out a difference between NumPy and PyTorch. PyTorch also allows developers to perform computations on Tensors. PyTorch has a robust framework to build computational graphs on the go and even change them in runtime. Other advantages of PyTorch include multi GPU support, simplified preprocessors, and custom data loaders.

PyTorch is an open-source deep learning framework built by Facebook’s AI Research lab. It was created to implement advanced neural networks and cutting-edge research ideas in industry and academia.

Like scikit-learn, PyTorch focuses on data modeling. However, it is intended for advanced users who work primarily with deep neural networks. PyTorch is a great tool to use when you need a production-ready machine learning model that is fast, efficient, scalable, and can work with a distributed environment.

### Pandas

– used to import and manage the datasets

Pandas are turning up to be the most popular Python library that is used for data analysis with support for fast, flexible, and expressive data structures designed to work on both “relational” or “labeled” data. Pandas today is an inevitable library for solving practical, real-world data analysis in Python. Pandas is highly stable, providing highly optimized performance. The backend code is purely written in C or Python.

The two main types of data structures used by pandas are :

* Series (1-dimensional)
* DataFrame (2-dimensional)

These two put together can handle a vast majority of data requirements and use cases from most sectors like science, statistics, social, finance, and of course, analytics and other areas of engineering.

Pandas support and perform well with different kinds of data including the below :

* Tabular data with columns of heterogeneous data. For instance, consider the data coming from the SQL table or Excel spreadsheet.
* Ordered and unordered time series data. The frequency of time series need not be fixed, unlike other libraries and tools. Pandas is exceptionally robust in handling uneven time-series data
* Arbitrary matrix data with the homogeneous or heterogeneous type of data in the rows and columns
* Any other form of statistical or observational data sets. The data need not be labeled at all. Pandas data structure can process it even without labeling.

For those working with tabular data in Python, pandas is the first choice for data analysis and manipulation. One of its key features is the data frame, a dedicated data structure for two-dimensional data. Data frame objects have rows and columns just like tables in Excel.

The pandas library has a huge set of tools for data cleaning, manipulation, analysis, and visualization. With pandas, you can:

* Add, delete, and update data frame columns.
* Handle missing values.
* Index, rename, sort, and merge data frames.
* Plot data distribution, etc.

## **Libraries for Visualizing Data (Plotting libraries)**

In addition to data analysis and modeling, Python is also a great tool for visualizing data. Here are some of the most popular Python libraries that can help you create meaningful, informative, interactive, and appealing data visualizations.

### Matplotlib

– Matplotlib is a data visualization library that is used for 2D plotting to produce publication-quality image plots and figures in a variety of formats. The library helps to generate histograms, plots, error charts, scatter plots, bar charts with just a few lines of code.

It provides a MATLAB-like interface and is exceptionally user-friendly. It works by using standard GUI toolkits like GTK+, wxPython, Tkinter, or Qt to provide an object-oriented API that helps programmers to embed graphs and plots into their applications.

This is a standard library for generating data visualizations in Python. It supports building basic two-dimensional graphs like line plots, histograms, scatter plots, bar charts, and pie charts, as well as more complex animated and interactive visualizations.

The matplotlib library is also flexible with regards to formatting and styling plots; you can choose how to display labels, grids, legends, etc. However, one major disadvantage to matplotlib is that it requires data scientists to write lots of code to create complex and visually appealing plots.

#### Seaborn

- Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.

Although it was built upon matplotlib, the seaborn library has a high-level interface that enables users to draw attractive and informative statistical graphs in just a few lines of code – or only one line of code! Its concise syntax and advanced features make it my favorite visualization tool.

Thanks to an expansive collection of visualizations and a set of built-in themes, you can create professional plots even if you are very new to coding data visualizations. Leverage seaborn’s extensive features to create heatmaps, violin plots, joint plots, multi-plot grids, and more.

### Bokeh

Bokeh is a great tool for creating interactive visualizations inside browsers. Like seaborn, it allows you to build complex plots using simple commands. However, its main focus is on interactivity.

With Bokeh, you can link plots, display relevant data while hovering over specific data points, embed different widgets, etc. Its extensive interactive abilities make Bokeh a perfect tool for building dashboards, network graphs, and other complex visualizations.

### Plotly

- is another browser-based visualization library. It offers many useful out-of-the-box graphics, including:

* Basic plots (e.g. scatterplots, line plots, bar charts, pie charts, bubble charts)
* Statistical plots (e.g., error bars, box plots, histograms).
* Scientific plots (e.g. contour plots, heatmaps).
* Financial charts (e.g. time series and candlestick charts).
* Maps (e.g. adding lines, filled areas, bubbles, and heatmaps to geographic maps).
* 3D plots (e.g. scatterplots, surface plots).

Consider using Plotly if you want to build interactive and publication-quality graphs.

# **References**

<https://docs.aws.amazon.com/wellarchitected/latest/machine-learning-lens/machine-learning-lens.html>

<https://riskspan.com/evaluating-supervised-and-unsupervised-learning-models/#:~:text=Model%20evaluation%20%28including%20evaluating%20supervised%20and%20unsupervised%20learning,or%20appropriately%20flagging%20credit%20card%20transactions%20as%20fraud>.

<https://medium.com/analytics-vidhya/evaluation-criteria-for-machine-learning-models-ce04692cc3e3>

<https://www.datasciencecentral.com/profiles/blogs/9-python-analytics-libraries-1>

<https://towardsdatascience.com/a-feature-selection-tool-for-machine-learning-in-python-b64dd23710f0>

<https://www.simplilearn.com/tutorials/machine-learning-tutorial/classification-in-machine-learning>

<https://learnpython.com/blog/python-libraries-for-data-science/#top>

**Book:**

Step-by-Step Guide To Implement Machine Learning Algorithms with Python, Author: Rudolph Russell

Perry Xiao - Artificial Intelligence Programming with Python from Zero to Hero-John WIley (2022)