



DATA SCIENCE AND MACHINE LEARNING WITH PYTHON





ABOUT ME

I'm Abdulsamod Azeez

Data Scientist and Machine Learning Engineer

Contents

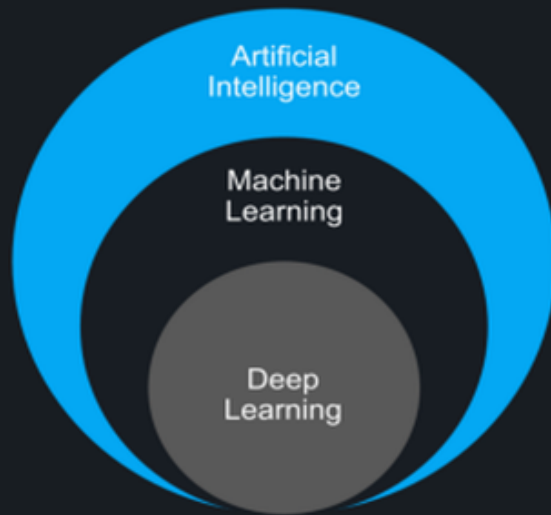
- Introduction to Artificial Intelligence (AI)
- Data Science and Machine Learning Fundamentals
- Introduction to Python Programming Language
- Predicting Penguins Species

MODULE ONE: INTRODUCTION TO ARTIFICIAL INTELLIGENCE (AI)

Introduction to AI:

Definitions of AI

- "... the science of making machines do things that would require intelligence if **done by humans**"
- Marvin Minsky
- AI is the part of computer science concerned with **designing** intelligent computer systems -E. Feigenbaum
- Systems that can demonstrate **human-like reasoning capability** to enhance the quality of life and improve business competitiveness
- Japan-S'pore AI Centre



Mimics Human Behavior

Building Trained Models of Data

- Probabilistic
- Linear Algebraic

Model built by multilayered neural networks.

History of Artificial Intelligence

1950

The time when it all started.

1955

John McCarthy coined term 'Artificial intelligence'.

1974

Computers became faster & affordable

1980

The year of Artificial Intelligence.

2000

Landmark of AI establishment achieved.

Applications of AI

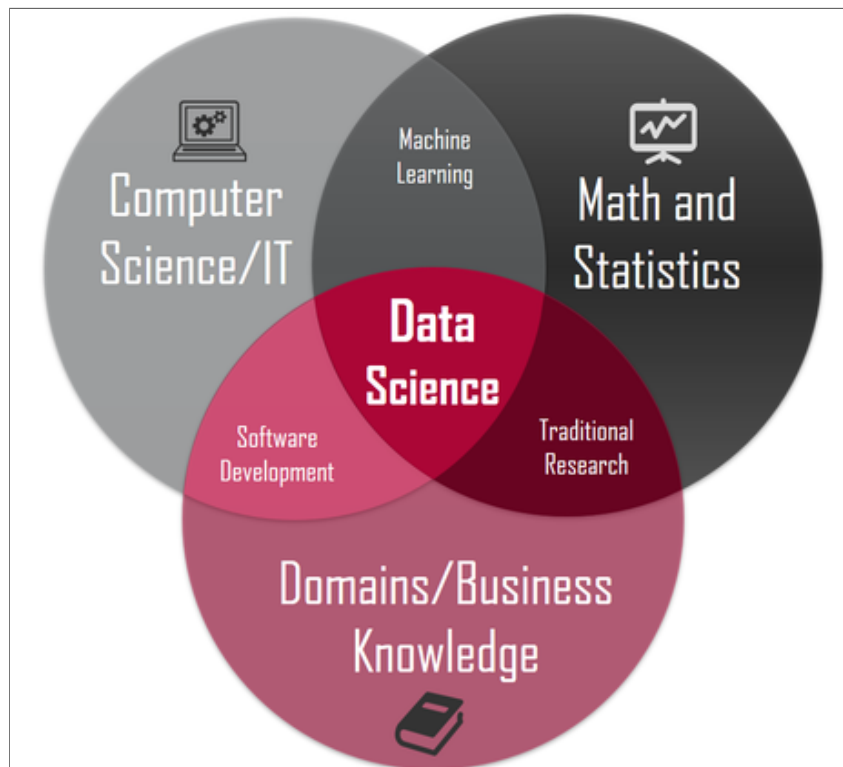
- Healthcare
- Education
- Autonomous Vehicles
- Business
- Travel Industry

In []:

MODULE TWO: INTRODUCTION TO DATA SCIENCE

DS is a "concept to unify statistics, data analysis, machine learning, domain knowledge and other related methods" in order to understand, gain insight or derive pattern from data.

The goal of every data science project is to obtain meaningful business information from data in order to make better decisions. It is the role of data scientists to define the project's objectives. This necessitates business knowledge and experience.





Data Science vs Data Analytics

	Data Science	Data Analytics
SKILLSET	<ul style="list-style-type: none">• Data Modelling• Predictive Analytics• Advanced Statistics• Engineering/Programming	<ul style="list-style-type: none">• BI Tools• Intermediate Statistics• Solid Programming Skills• Regular Expression (SQL)
SCOPE	Macro	Micro
EXPLORATION	<ul style="list-style-type: none">• Search Engine Exploration• Machine Learning• Artificial Intelligence• Big data - Often Unstructured	<ul style="list-style-type: none">• Data Visualization Techniques• Designing Principles• Big Data - Mostly Structured
GOALS	Discover New Questions to Drive Innovation	Use Existing Information to Uncover Actionable Data

What is machine learning?


Academic Definition

Machine learning is a subfield of computer science that 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.

Simple Definition

Computing systems that become smarter with learning and experience

Experience = Past data + human input



Human



I can learn everything
automatically from
experiences.
Can u learn?

Machine

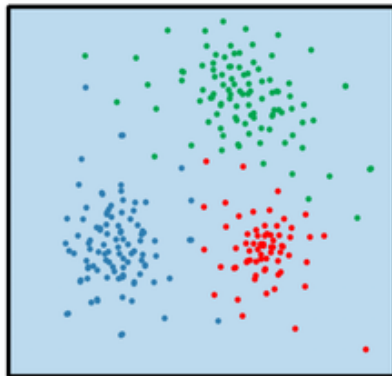


Yes, I can also learn
from past data with the
help of Machine learning

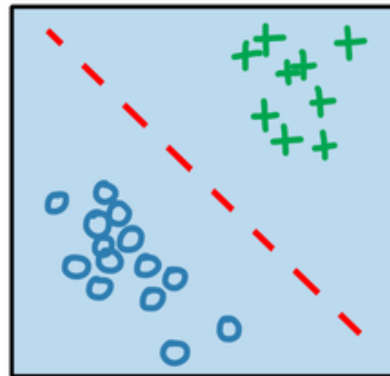
TYPES OF MACHINE LEARNING

machine learning

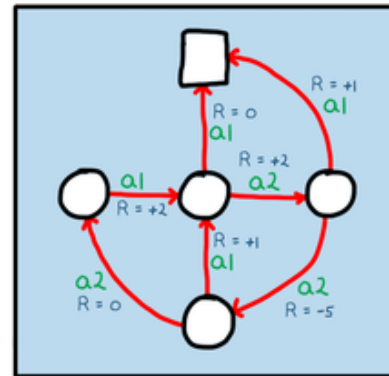
unsupervised
learning



supervised
learning



reinforcement
learning



1. Supervised Machine Learning

Supervised machine learning is based on supervision. It means that in the supervised learning technique, we train the machines using the "labelled" dataset, and the machine predicts the output based on the training. The tagged data in this case indicates that some of the inputs have already been mapped to the output. To put it another way, we first train the machine with the input and associated output, and then we ask the machine to predict the outcome using the test dataset.

Categories of Supervised Machine Learning

Supervised machine learning can be classified into two types of problems, which are given below:

- Classification
- Regression

A. Classification

Classification algorithms are used to handle classification issues using categorical output variables, such as "Yes" or "No," Male or Female, Red or Blue, and so on.

The categories in the dataset are predicted by the categorization algorithms.


Spam detection, email filtering, and other real-world examples of classification algorithms

Some popular classification algorithms are given below:

- Random Forest Algorithm
- Decision Tree Algorithm
- Logistic Regression Algorithm
- Support Vector Machine Algorithm

Target

Features



Cancel	Months since first subscription	Monthly average spent	Average number of phone calls made last month	Average number of phone calls made last quarter	Additional options
Yes	13	\$70	56	63	Yes
No	2	\$35	35	34	Yes
No	6	\$40	46	50	Yes
Yes	16	\$110	53	75	No

b. Regression

Regression algorithms are used to solve regression problems in which there is a linear relationship between input and output variables. These are used to predict continuous output variables, such as market trends, weather prediction, etc.

Some popular Regression algorithms are given below:

- Simple Linear Regression Algorithm
 - Multivariate Regression Algorithm
 - Decision Tree Algorithm
 - Lasso Regression
 - Random Forest Algorithm
-

Temperature	Pressure	Relative Humidity	Wind Direction	Wind Speed
10.69261758	986.882019	54.19337313	195.7150879	3.278597116
13.59184184	987.8729248	48.0648859	189.2951202	2.909167767
17.70494885	988.1119385	39.11965597	192.9273834	2.973036289
20.95430404	987.8500366	30.66273218	202.0752869	2.965289593
22.9278274	987.2833862	26.06723423	210.6589203	2.798230886
24.04233986	986.2907104	23.46918024	221.1188507	2.627005816
24.41475295	985.2338867	22.25082295	233.7911987	2.448749781
23.93361956	984.8914795	22.35178837	244.3504333	2.454271793
22.68800023	984.8461304	23.7538641	253.0864716	2.418341875
20.56425726	984.8380737	27.07867944	264.5071106	2.318677425
17.76400389	985.4262085	33.54900114	280.7827454	2.343950987
11.25680746	988.9386597	53.74139903	68.15406036	1.650191426
14.37810685	989.6819458	40.70884681	72.62069702	1.553469896
18.45114201	990.2960205	30.85038484	71.70604706	1.005017161
22.54895853	989.9562988	22.81738811	44.66042709	0.264133632
24.23155922	988.796875	19.74790765	318.3214111	0.329656571

2.Unsupervised Machine Learning

Unsupervised learning is different from the Supervised learning technique; as its name suggests, there is no need for supervision. It means, in unsupervised machine learning, the machine is trained using the unlabeled dataset, and the machine predicts the output without any supervision.

The main aim of the unsupervised learning algorithm is to group or categories the unsorted dataset according to the similarities, patterns, and differences. Machines are instructed to find the hidden patterns from the input dataset.

One of the example of unsupervised machine learning is:

- Clustering The clustering technique is used when we want to find the inherent groups from the data. It is a way to group the objects into a cluster such that the objects with the most similarities remain in one group and have fewer or no similarities with the objects of other groups. An example of the clustering algorithm is grouping the customers by their purchasing behaviour.

Some of the popular clustering algorithms are given below:

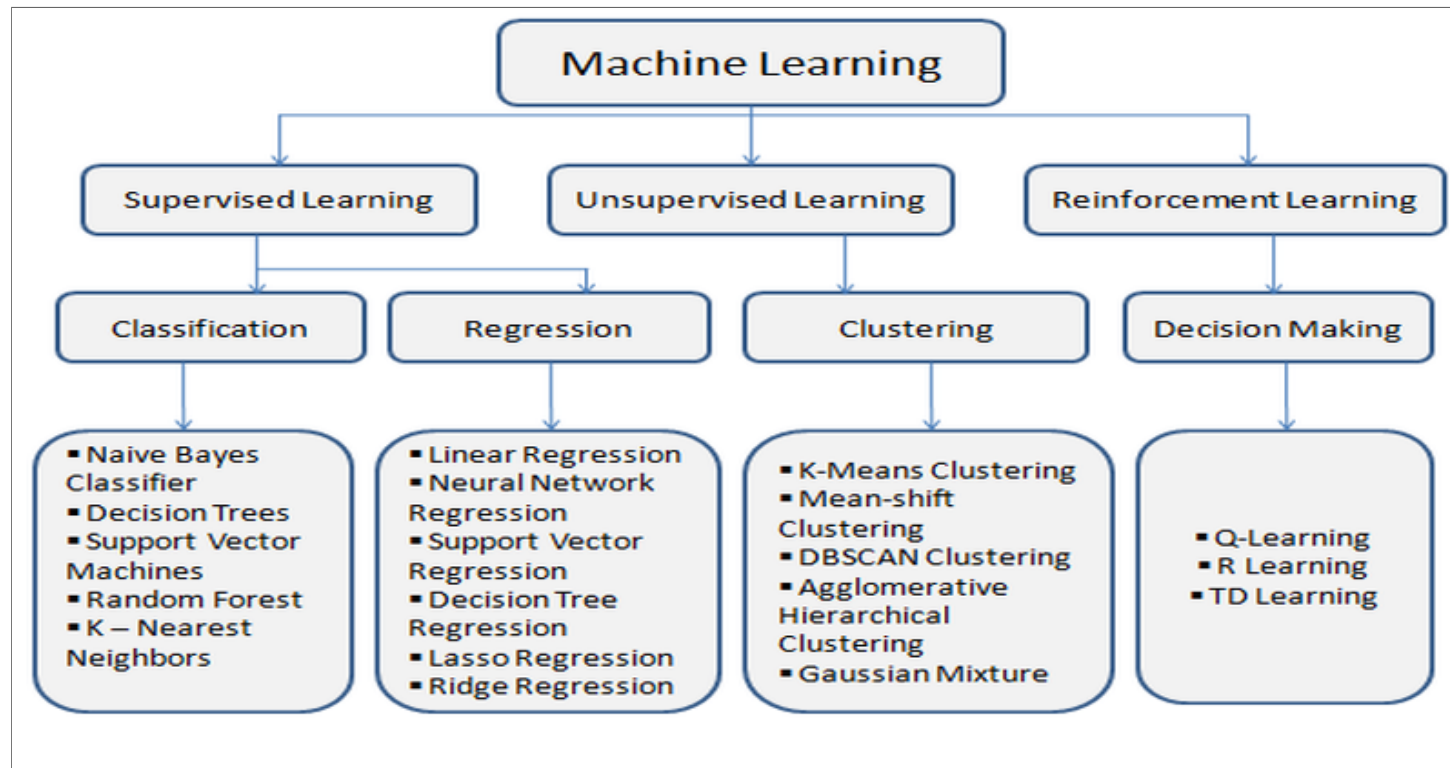
- K-Means Clustering algorithm
- Mean-shift algorithm
- DBSCAN Algorithm
- Principal Component Analysis
- Independent Component Analysis

3.Reinforcement Learning

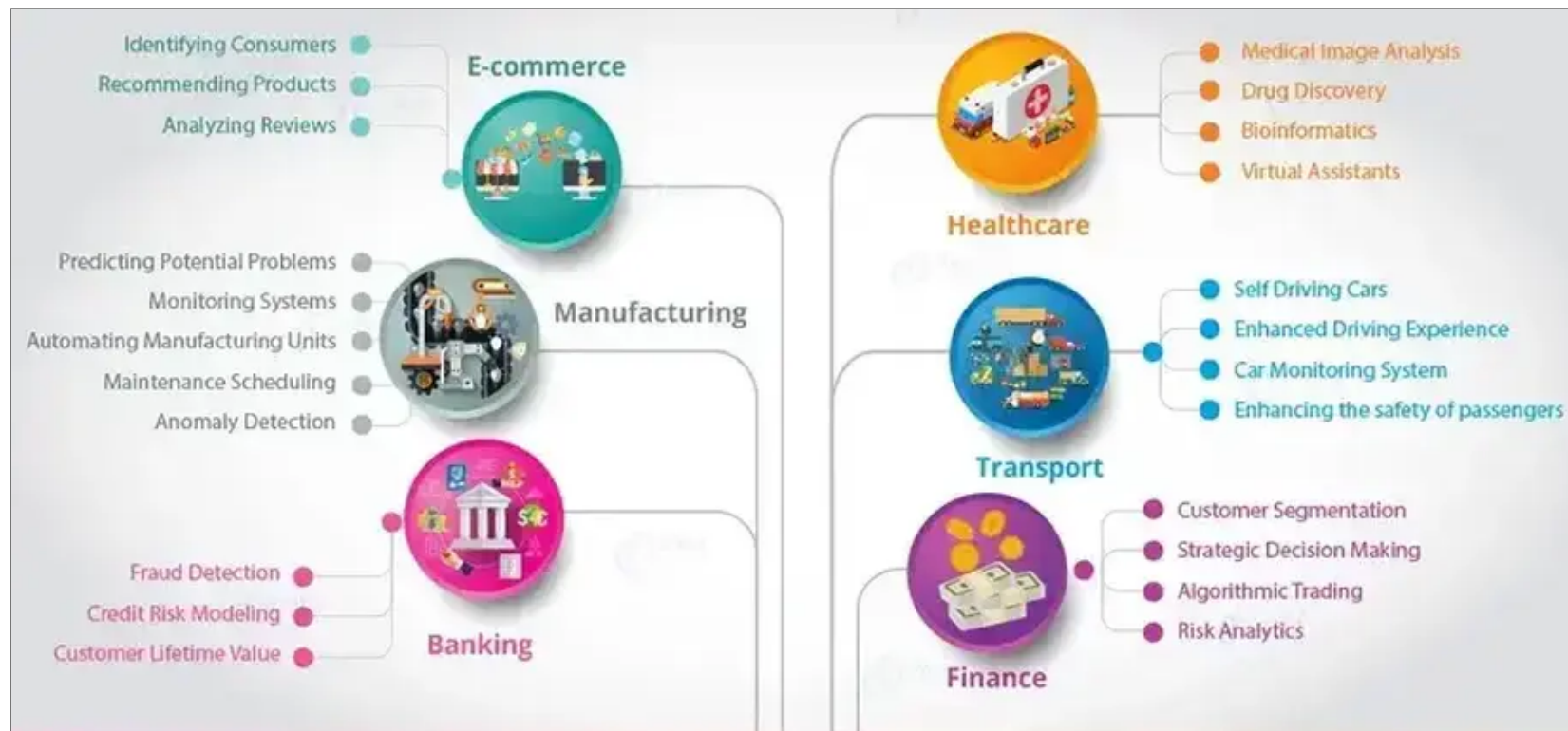
Reinforcement learning works on a feedback-based process, in which an AI agent (A software component) automatically explore its surrounding by hitting & trail, taking action, learning from experiences, and improving its performance. Agent gets rewarded for each good action and get punished for each bad action; hence the goal of reinforcement learning agent is to maximize the rewards.

In reinforcement learning, there is no labelled data like supervised learning, and agents learn from their experiences only.

Quick Summary on the Types of Machine Learning, their Categories and Alogorithm Used



Applications for Data Science



Steps in Solving Data Science Problems

Running a data science project usually involves multiple steps, including the following:

1. Defining the business problem to be solved
2. Collecting or extracting existing data
3. Analyzing, visualizing, and preparing data
4. Training a model to spot patterns in data and make predictions
5. Assessing a model's performance and making improvements
6. Communicating and presenting findings and gained insights
7. Deploying and maintaining a model

In []:

MODULE Three: INTRODUCTION TO PYTHON PROGRAMMING LANGUAGE

Python is a popular programming language. It was created by Guido van Rossum, and released in 1991.

It is used for:

- web development (server-side),
- software development,
- mathematics,
- system scripting.

What can Python do?

- Python can be used on a server to create web applications.
- Python can be used alongside software to create workflows.
- Python can connect to database systems. It can also read and modify files.
- Python can be used to handle big data and perform complex mathematics.

- Python can be used for rapid prototyping, or for production-ready software development.

Why Python?

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

Good to know

The most recent major version of Python is Python 3, which we shall be using in this workshop. However, Python 2, although not being updated with anything other than security updates, is still quite popular.

In [1]:

```
# The Print function is use to display the output  
print("Hello World")
```

Hello World

In [2]:

```
# Creating a variable
```

```
name_of_school = "Thomas Adewumi University"  
print(name_of_school)
```

```
level= 200  
print(level)
```

Thomas Adewumi University
200

Python Data Types

Example	Data Type
<code>x = "Hello world"</code>	str
<code>x = 20</code>	int
<code>x = 20.5</code>	float
<code>x = 1j</code>	complex
<code>x = ["apple", "banana", "cherry"]</code>	list
<code>x = ("apple", "banana", "cherry")</code>	tuple
<code>x = range(6)</code>	range
<code>x = {"name" : "John", "age" : 36}</code>	dict
<code>x = {"apple", "banana", "cherry"}</code>	set

Python Loops

Python has two primitive loop commands:

- while loops
- for loops

In [3]:

```
#The while loop  
#With the while loop we can execute a set of statements as long as a condition is true.  
  
i = 1  
while i < 5:  
    print(i)  
    i += 1
```

1

2

3

4

In [4]:

```
#The break statment  
i = 1  
while i < 6:  
    print(i)  
    if i == 3:  
        break  
    i += 1
```

1

2

3

In [5]:

```
#A for loop is used for iterating over a sequence (that is either a list, a tuple, a dictionary, a set, or a string).  
fruits = ["apple", "banana", "cherry"]  
for x in fruits:  
    print(x)
```

apple

banana

cherry

In [6]:

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

apple

banana

In [7]:

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

b
a
n
a
n
a

Function

A function is a block of code which only runs when it is called.

You can pass data, known as parameters, into a function.

A function can return data as a result.

Creating a Function

In Python a function is defined using the def keyword:

In [8]:

```
def my_function():  
    print("Hello World")
```

In [9]:

```
# Calling a function  
  
def my_function():  
    print("Hello World")  
  
my_function()
```

Hello World

In [10]:

```
def my_function(fname):  
    print(fname + " Adewumi")  
  
my_function("Thomas")
```

Thomas Adewumi

MODULE Four: PREDICTING PENGUINS SPECIES

AIM: Predicting penguin species

The data for this activity comes from a study of three different species of Antarctic penguin: Adelie, Chinstrap and Gentoo.

In this notebook you will explore which features can be used to predict the species of a penguin.

You will build a model to predict whether a particular penguin is either a Gentoo, Adelie or Chinstrap penguin.

The dataset consists of 7 columns.

- **species:** penguin species (Chinstrap, Adélie, or Gentoo)
- **culmen_length_mm:** culmen length (mm)
- **culmen_depth_mm:** culmen depth (mm)
- **flipper_length_mm:** flipper length (mm)
- **body_mass_g:** body mass (g)
- **island:** island name (Dream, Torgersen, or Biscoe) in the Palmer Archipelago (Antarctica)
- **sex:** penguin sex

Body Part of Penguin



DATASET

Data were collected and made available by Dr. Kristen Gorman and the Palmer Station, Antarctica LTER, a member of the Long Term Ecological Research Network.

penguins_size.csv: Simplified data from original penguin data sets. Contains variables:

- species: penguin species (Chinstrap, Adélie, or Gentoo)
- culmen_length_mm: culmen length (mm)
- culmen_depth_mm: culmen depth (mm)
- flipper_length_mm: flipper length (mm)
- body_mass_g: body mass (g)
- island: island name (Dream, Torgersen, or Biscoe) in the Palmer Archipelago (Antarctica)
- sex: penguin sex

In [11]:

```
import numpy as np
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import StandardScaler
import os

# Visualisation libraries
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
```


Importing the dataset

In [12]:

```
df = pd.read_csv('penguins_size.csv')  
df.head()
```

Out[12]:

	species	island	culmen_length_mm	culmen_depth_mm	flipper_length_mm	body_mass_g	sex
0	Adelie	Torgersen	39.1	18.7	181.0	3750.0	MALE
1	Adelie	Torgersen	39.5	17.4	186.0	3800.0	FEMALE
2	Adelie	Torgersen	40.3	18.0	195.0	3250.0	FEMALE
3	Adelie	Torgersen	NaN	NaN	NaN	NaN	NaN
4	Adelie	Torgersen	36.7	19.3	193.0	3450.0	FEMALE

Understanding the data

In [13]:

```
df.info()  
  
print(df.shape)
```

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 344 entries, 0 to 343

Data columns (total 7 columns):

#	Column	Non-Null Count	Dtype
---	-----	-----	-----
0	species	344 non-null	object
1	island	344 non-null	object
2	culmen_length_mm	342 non-null	float64
4			
3	culmen_depth_mm	342 non-null	float64
4			
4	flipper_length_mm	342 non-null	float64

```
4
  5    body_mass_g          342 non-null    float64
4
  6    sex                  334 non-null    object
dtypes: float64(4), object(3)
memory usage: 18.9+ KB
(344, 7)
```

In [14]:

```
print(df.shape)
```

(344, 7)

In [15]:

```
df.describe(include='all')
```

Out[15]:

	species	island	culmen_length_mm	culmen_depth_mm	flipper_length_mm	body_mass_g	sex
count	344	344	342.000000	342.000000	342.000000	342.000000	334
unique	3	3	NaN	NaN	NaN	NaN	3
top	Adelie	Biscoe	NaN	NaN	NaN	NaN	MALE
freq	152	168	NaN	NaN	NaN	NaN	168
mean	NaN	NaN	43.921930	17.151170	200.915205	4201.754386	NaN
std	NaN	NaN	5.459584	1.974793	14.061714	801.954536	NaN
min	NaN	NaN	32.100000	13.100000	172.000000	2700.000000	NaN
25%	NaN	NaN	39.225000	15.600000	190.000000	3550.000000	NaN
50%	NaN	NaN	44.450000	17.300000	197.000000	4050.000000	NaN
75%	NaN	NaN	48.500000	18.700000	213.000000	4750.000000	NaN
max	NaN	NaN	59.600000	21.500000	231.000000	6300.000000	NaN

In [16]:

```
# Covariance  
  
print('Covariance:')  
df.cov()
```

Covariance:

Out[16]:

	culmen_length_mm	culmen_depth_mm	flipper_length_mm	body_mass_g
culmen_length_mm	29.807054	-2.534234	50.375765	2605.591912
culmen_depth_mm	-2.534234	3.899808	-16.212950	-747.370093
flipper_length_mm	50.375765	-16.212950	197.731792	9824.416062
body_mass_g	2605.591912	-747.370093	9824.416062	643131.077327

In [17]:

```
print('Correlation:')  
df.corr()
```

Correlation:

Out[17]:

	culmen_length_mm	culmen_depth_mm	flipper_length_mm	body_mass_g
culmen_length_mm	1.000000	-0.235053	0.656181	0.595110
culmen_depth_mm	-0.235053	1.000000	-0.583851	-0.471916
flipper_length_mm	0.656181	-0.583851	1.000000	0.871202
body_mass_g	0.595110	-0.471916	0.871202	1.000000

Missing values

In [18]:

```
df.isnull().sum()
```

Out[18]:

species	0
island	0
culmen_length_mm	2
culmen_depth_mm	2
flipper_length_mm	2
body_mass_g	2
sex	10
dtype:	int64

In [19]:

```
# Handling missing values

from sklearn.impute import SimpleImputer
#setting strategy to 'most frequent' to impute by the mean
imputer = SimpleImputer(strategy='most_frequent') # strategy can also be mean or median
df.iloc[:, :] = imputer.fit_transform(df)
```

In [20]:

```
df.isnull().sum()
```

Out[20]:

species	0
island	0
culmen_length_mm	0
culmen_depth_mm	0
flipper_length_mm	0
body_mass_g	0
sex	0
dtype:	int64

Convert sex column to integer values

In [21]:

```
lb = LabelEncoder()  
df["sex"] = lb.fit_transform(df["sex"])  
df["island"] = lb.fit_transform(df["island"])  
  
df['sex'][:5]
```

Out[21]:

0	2
1	1
2	1
3	2
4	1

Name: sex, dtype: int64

Analysing the data visually

Number of Species

In [22]:

```
df['species'].value_counts()
```

Out[22]:

Adelie	152
Gentoo	124
Chinstrap	68

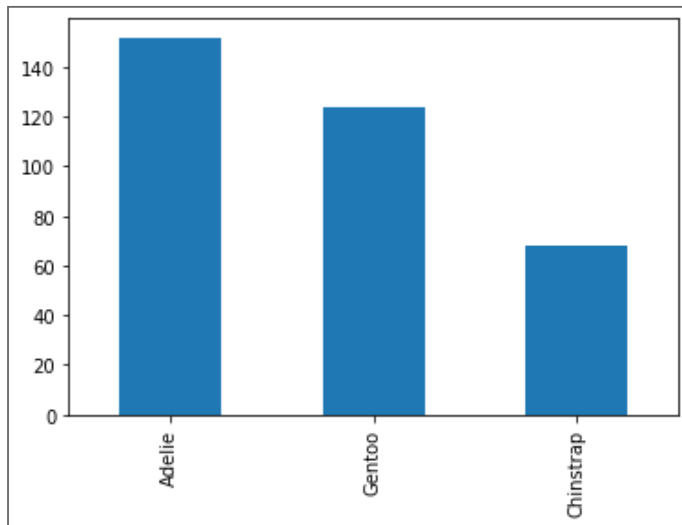
Name: species, dtype: int64

In [23]:

```
df['species'].value_counts().plot(kind='bar')  
  
#df['species'].value_counts().plot.bar()
```

Out[23]:

<AxesSubplot:>



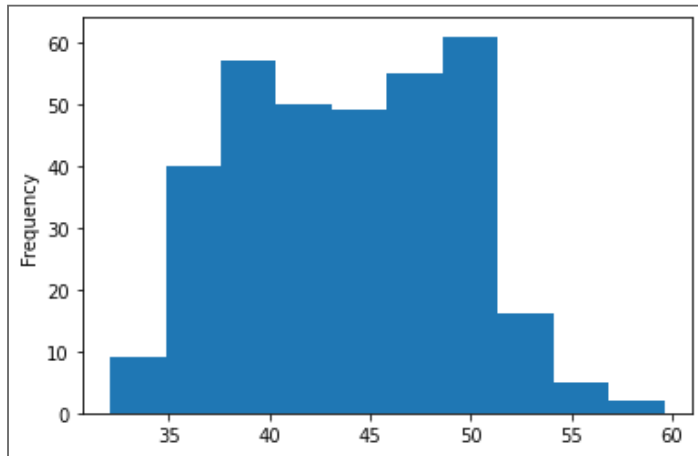
Histogram

In [24]:

```
df["culmen_length_mm"].plot(kind='hist')
```

Out[24]:

<AxesSubplot: ylabel= 'Frequency' >

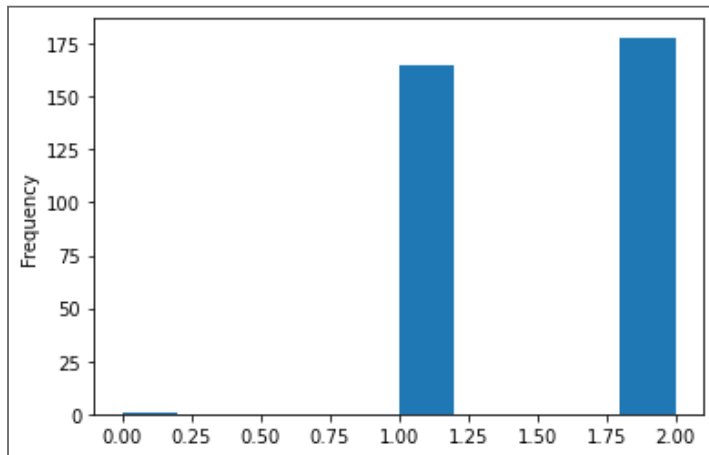


In [25]:

```
df["sex"].plot(kind='hist')
```

Out[25]:

<AxesSubplot: ylabel='Frequency'>



In [26]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
RangeIndex: 344 entries, 0 to 343
```

```
Data columns (total 7 columns):
```

#	Column	Non-Null Count	Dtype
---	-----	-----	-----
0	species	344 non-null	object
1	island	344 non-null	int64
2	culmen_length_mm	344 non-null	float64
4			
3	culmen_depth_mm	344 non-null	float64
4			
4	flipper_length_mm	344 non-null	float64
4			

5	body_mass_g	344 non-null	float64
4			
6	sex	344 non-null	int64

dtypes: float64(4), int64(2), object(1)
memory usage: 18.9+ KB

Data Preprocessing

Preparing the data for modelling

In [27]:

```
x= df.drop("species", axis=1)
y= df["species"]
```

In [28]:

```
x
```

Out[28]:

	island	culmen_length_mm	culmen_depth_mm	flipper_length_mm	body_mass_g	sex
0	2	39.1	18.7	181.0	3750.0	2
1	2	39.5	17.4	186.0	3800.0	1
2	2	40.3	18.0	195.0	3250.0	1
3	2	41.1	17.0	190.0	3800.0	2
4	2	36.7	19.3	193.0	3450.0	1
...
339	0	41.1	17.0	190.0	3800.0	2
340	0	46.8	14.3	215.0	4850.0	1
341	0	50.4	15.7	222.0	5750.0	2
342	0	45.2	14.8	212.0	5200.0	1
343	0	49.9	16.1	213.0	5400.0	2

344 rows × 6 columns

In [29]:

```
y
```

Out[29]:

0	Adelie
1	Adelie
2	Adelie
3	Adelie
4	Adelie

...

339	Gentoo
340	Gentoo
341	Gentoo
342	Gentoo
343	Gentoo

Name: species, Length: 344, dtype: object

In [30]:

```
#importing packages for modelling
```

```
from sklearn.model_selection import train_test_split  
from sklearn.linear_model import LogisticRegression  
from sklearn.ensemble import RandomForestClassifier
```

In [31]:

```
x_train, x_test, y_train, y_test= train_test_split(x, y, test_size=0.3)
```

In [32]:

```
#initialize the model  
  
model= LogisticRegression()  
  
#fit the model with the training set  
model.fit(x_train, y_train)
```

```
/home/abdulsamod/anaconda3/lib/python3.8/site-  
packages/sklearn/linear_model/_logistic.py:76  
3: ConvergenceWarning: lbfgs failed to converg  
e (status=1):  
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.
```

Increase the number of iterations (max_iter) or scale the data as shown in:

<https://scikit-learn.org/stable/modules/preprocessing.html>

Please also refer to the documentation for alt

ernative solver options:

https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
n_iter_i = _check_optimize_result(

Out[32]:

LogisticRegression()

In [33]:

```
pred= model.predict(x_test)  
pred
```

Out[33]:

```
array(['Adelie', 'Adelie', 'Adelie', 'Gentoo',  
      'Adelie', 'Adelie',  
      'Chinstrap', 'Adelie', 'Adelie', 'Adeli  
e', 'Adelie', 'Gentoo',  
      'Adelie', 'Adelie', 'Gentoo', 'Chinstra  
p', 'Gentoo', 'Adelie',  
      'Adelie', 'Gentoo', 'Adelie', 'Gentoo',  
      'Adelie', 'Adelie',  
      'Gentoo', 'Chinstrap', 'Gentoo', 'Adeli  
e', 'Gentoo', 'Gentoo',  
      'Gentoo', 'Gentoo', 'Gentoo', 'Adelie',  
      'Adelie', 'Adelie',
```

'Gentoo', 'Gentoo', 'Gentoo', 'Chinstrap', 'Adelie', 'Adelie',
 'Adelie', 'Adelie', 'Adelie', 'Adelie',
 'Adelie', 'Gentoo',
 'Gentoo', 'Chinstrap', 'Gentoo', 'Gentoo',
 'Gentoo', 'Adelie',
 'Chinstrap', 'Chinstrap', 'Gentoo', 'Adelie',
 'Gentoo', 'Adelie',
 'Chinstrap', 'Chinstrap', 'Chinstrap',
 'Gentoo', 'Gentoo',
 'Gentoo', 'Gentoo', 'Adelie', 'Chinstrap',
 'Gentoo', 'Gentoo',
 'Gentoo', 'Adelie', 'Gentoo', 'Gentoo',
 'Gentoo', 'Adelie',
 'Gentoo', 'Adelie', 'Adelie', 'Gentoo',
 'Adelie', 'Adelie',
 'Gentoo', 'Gentoo', 'Adelie', 'Chinstrap'

```
p', 'Chinstrap', 'Gentoo',  
    'Gentoo', 'Chinstrap', 'Gentoo', 'Adeli  
e', 'Gentoo', 'Gentoo',  
    'Adelie', 'Adelie', 'Gentoo', 'Gentoo',  
    'Chinstrap', 'Adelie',  
    'Adelie', 'Adelie', 'Adelie'], dtype=ob  
ject)
```

Evaluate the performance of the model

In [34]:

```
from sklearn.metrics import accuracy_score
```

In [35]:

```
accuracy= accuracy_score(pred, y_test)  
accuracy
```

Out[35]:

0.9711538461538461



Thank You

In []: