NAME OF THE COURSE	MADSC202 DEEP LEARNING AND AI
NAME OF THE FACULTY	Dr. ATHANASIOS RIZOS
NAME OF THE STUDENTS	KUMAR GAURAV – 22026384
TITLE OF ASSIGNMENT	FINAL ASSIGNMENT
DATE	13.03.2023
PLACE	MÜNCHEN

eu Business School

SIMILARITIES BETWEEN (UN)SUPERVISED, REINFORCEMENT AND ACTIVE LEARNING

- All are subsets of machine learning and tries to find a solution or an outcome for a problem
- Supervised learning and active learning use labelled data for training
- Similarities between Active learning and unsupervised learning are they use unlabeled data for training
- Unsupervised learning and reinforcement learning requires no supervision during training

DIFFERENCES BETWEEN (UN)SUPERVISED, REINFORCEMENT, AND ACTIVE LEARNING

	Supervised Learning	Unsupervised Learning	Reinforcement learning	Active learning
Definition	Model learns labelled dataset with guidance. Applied in faces recognition, object recognition. Models: Regression, Random forest	Machine is trained with unlabeled data without guidance. Clustering DNA patterns Models: K–Means, PCA	Agent interacts with the environment by performing actions and learns from error or rewards. Applied in self driving cars	Model interactively queries the user to label new data points with desired output. Applied in image recognition
Types of problems	Regression– Eg. When output is continuous i.e stock taking Classification — Target is Categorical; disease or no disease	Association– Eg.people that purchase A tend to buy B Clustering — grouping cars based on some features	Reward based – Low error, high reward; high error, low reward	Medical imaging
Type of data	Use labelled data	Use unlabeled data	Data is not predefined	Use both labelled and unlabeled data for training
Training	Requires external supervision	No supervision is needed	No supervision is needed	Supervised and unsupervised
Approach	Maps labelled input to the known output	Understands pattern and discover the output	Follows trial and error method – Low error high reward Vs	Algorithm queries the user in form of unlabeled data instances and the request is to a human to label the instance

ACTIVE LEARNING CATEGORIES

Active learning is part of machine learning where the algorithm interacts with the user actively to label the data in a collection of unlabeled data. It queries the user to label data.

Active learning has 3 categories namely:–

- Stream based selective sampling
- Pool based sampling approach
- Membership query synthesis approach

STREAM BASED SELECTIVE SAMPLING

- Algorithm assess unlabeled data points one–by–one,
- When the algorithm comes across a data point while in training, it immediately decides whether to query the label or not
- This immediate decision making makes this category to often exceed the allocated budget

POOL BASED SAMPLING

- This category tries to go through the entire data and evaluate it before deciding which is the best part to query for a label
- It is first trained on a labelled data and then it is used to decide which part would be beneficial to query about
- Then the beneficial part is introduced into the training data of the next iteration loop
- But this category requires a lot of memory as it requires to these loops

MEMBERSHIP QUERY SYNTHESIS

- Active learner algorithm generates its own hypothetical data points
- This is possible only when there is smaller data
- For example, when the data consists of animals and humans, it can send a part of the image to query it for confirming if it belongs to humans or animals.

EXAMPLES WHERE DL/ML IS UNFAIR

Unfairness due to the algorithms

- Nickon's facial recognition algorithm in the camera trained to identify blinking eyes when taking a picture, identified Asian people as blinking at higher rate compared to other people. This may be due to the unfair training data that did not cover all types demographics and ethnicity.

Bias in data that leads to life impacting decision

- Machine learning algorithms are used in courts to determining whether a criminal will commit the crime again.
- In such cases if the data is biased then the algorithm which is trained on that will also be biased and may give a result that could be false and also impact the human life on a larger scale.

MARKOV DECISION PROCESS

- This is type of reinforcement learning where an agent decides an action based on current state and this step is repeated for every current state and is called Markov Decision Process model
- It is a decision making process that uses a frame work of mathematics to model a decision making system and MDP is mainly used when the results are random
- For example predicting an efficient financial market for the future
- The MDP has the following elements

  States – This represents every state that an agent can be in

  Actions – All possible actions that can be taken in the state

  Transition models – The effect of the action taken in the state

  Reward – The rewards that an agent is awarded for taking action when being in a state and arriving at a desired state

  Policies – It is a map that contains the actions to be taken when being present in a state

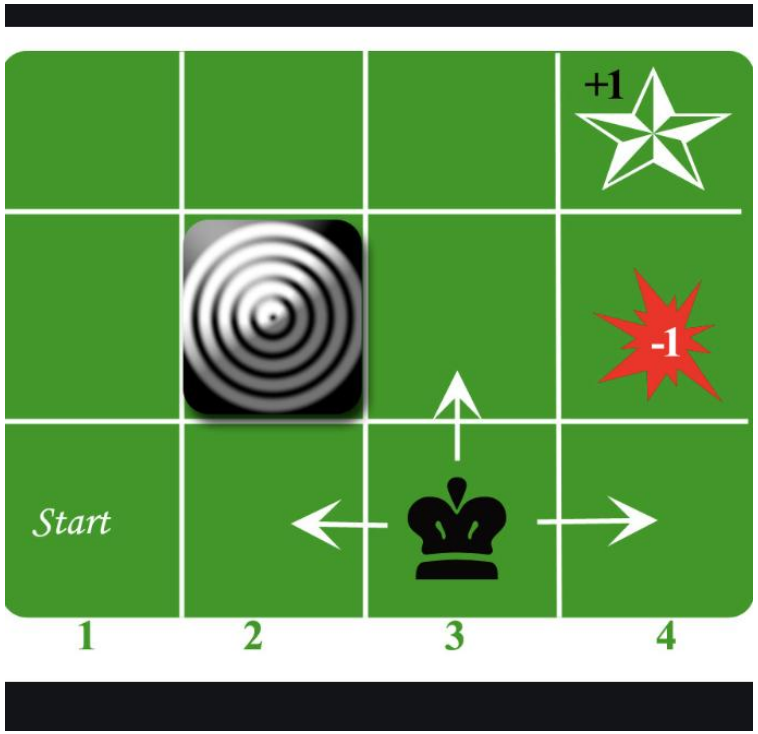
MARKOV DECISION PROCESS

- An agent is the one responsible for taking the decisions and

  It is rewarded based on the outcomes
- In the example to the right the final goal is to reach the white star
- The agent operates within the grid, and it starts from the 'start'

  position
- It has the policy map to take actions like

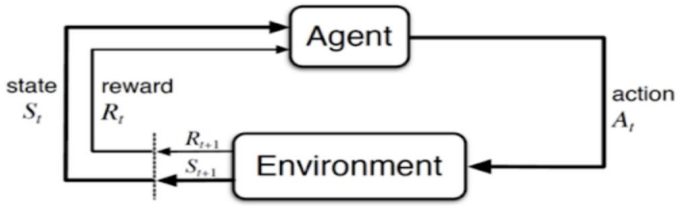
  Up, Down, Right, Left required when in a state
- It should avoid the red star (4,2) and it has a block at (2,2)
- The agent is rewarded on each steps that may be small rewards
- Big rewards are obtained when the final state is achieved, and the goal is to reach maximum rewards and reach the final state



(Figure 1: Markov decision process, geeksforgeeks.org)

MARKOV DECISION PROCESS

- The below figure represents the MDP model



(Figure 2: Key elements of MDP, spiceworks.com)

- The MDP uses Markov property and the it can be given by the below equation

$$P[St+1|St] = P[St+1\ |S1,S2,S3.....St]$$

(Figure 3: Equation for evaluating Markov property, spiceworks.com)

- Here the equation states that MDP only considers current state to evaluate next state without considering previous states

VARIABLE CATEGORIES IN PYTHON

- There are four categories of variables in python and they are: **Int, Float, Complex, String**

Category 1 – Int:

- Integer variable also known as Int is a type of variable which stores whole numbers (0,1,2,3,4 etc) as its value. It also includes negative numbers (–1,–2—3,–4 etc)
- For example:

```
lucky_number=6
Print("The variable type is:",type(lucky_number))
```

- Here we assign a whole number to a variable named lucky_number and hence the value is a whole number the variable becomes an Integer (Int)
- We can also use two Int variables in another to perform mathematical operations
- For example

```
a=3
b=2
c=a/b
print(c)
```

VARIABLE CATEGORIES IN PYTHON

- **Category 2 – Float:**
- When the variable is assigned a number which is not a whole number, then python takes it as a float number
- Float number consists of decimal points
- For example: 2.4

```
float_num=2.4
Print("The variable type is:",type(float_num))
```

- The output for this would be <class 'float'>
- Another example

```
float_num1=6.4
num2=2
num3=float_num1/num2
print("The variable type is:",type(num3))
```

- The result would be 3.2 and it would print it as <class 'float'>

VARIABLE CATEGORIES IN PYTHON

- **Category 3 – Complex:**
- Complex numbers are when a real part and an imaginary number is present in the values
- For example: ex=5+6j, where 5 is the real part and 6j is the imaginary part
- This takes an algebraic form and python handles it as it is in the mathematical operation
- For example,

```
complex1=23028194j
complex2=5+6j
normal=2023
complex=complex1+normal
print("The variable type is:",type(complex))
```

- Here it is not necessary that we need to have a real and imaginary part to obtain the result as complex number
- Python will consider the variable as complex even if we just give the imaginary part
- The output of the code even if we consider only the variable 'complex1' without real part, the output will be <class 'complex'>

VARIABLE CATEGORIES IN PYTHON

- **Category 4 – String:**
- A string variable is a collection of characters which are treated as text
- A sequence of characters is put into a string variable
- For example name=’sekar’ is a string variable containing sequence of characters ‘s’, ‘e’, ‘k’, ‘a’, ‘r’
- String variables are defined with single and double quotes
- For example

```
name1="அரவிந்  “
print(“The variable type is:”,type(name1))
```

- Above is Aravind in ‘Tamil’ language
- Python can detect any language texts and consider it as character variable

GETTING STUDENT ID AND CALCULATING THEIR SUM

- We can get the student Id from the user using input function
- Then a simple addition is done to calculate the sum of the student Id
- Int function is used since the student number that we get from the user are integer
- For example,

```
#getting student  Ids
id_1=int(input(“Enter the Student ID of member 1: ”))
id_2=int(input(“Enter the Student ID of member 2: ”))
id_3=int(input(“Enter the Student ID of member 3: ”))
id_4=int(input(“Enter the Student ID of member 4: ”))

# calculating sum of student IDs and printing the sum
student_id_sum=id1+id2+id3+id4
print(“The sum of all the student Id of the members is: “,student_id_sum)
```

GETTING SURNAMES FROM USER AND USING + AND * OPERATORS

- We can use the input function to get the surnames from the user
- We need to use the quotes since it is a character variable
- For example,

```
#collecting surnames from users
surname1=input("Enter the surname of member 1: ")
surname2=input("Enter the surname of member 2: ")
surname3=input("Enter the surname of member 3: ")
surname4=input("Enter the surname of member 4: ")

#using + and * operators
combination1=surname1*3+surname2
combination2=surname2+surname3
combination3=surname3+surname4*2
```

GETTING SURNAMES FROM USER AND USING + AND * OPERATORS

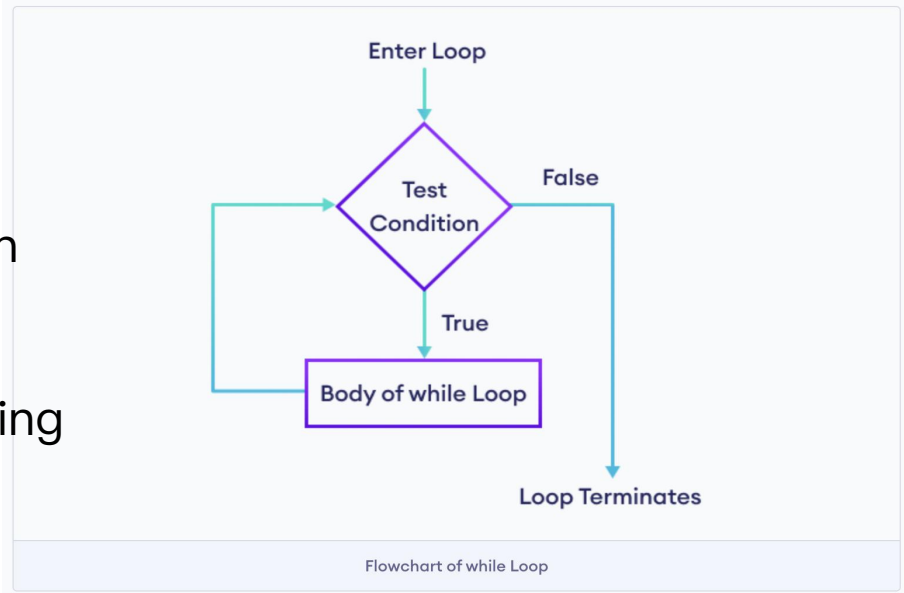
```
#printing the variables
print(combination1)
print(combination2)
print(combination3)
```

- The output would be :

```
Enter the surname of member 1: palanisamy
Enter the surname of member 2: gaurav
Enter the surname of member 3: onyango
Enter the surname of member 4: ibrahim
palanisamypalanisamypalanisamygaurav
gauravonyango
onyangoibrahimibrahim
```

WHILE LOOP

- While loops are used when we want to execute a certain code repeatedly until the given condition is satisfied
- The flowchart explains the flow of the While loop
- First the while loop checks the condition and if it is True, it will execute the code
- Again after executing the code it will check for the condition and if it is true it will execute the code again
- It will run the code until the condition is True
- When the condition becomes false the loop will stop executing



(Figure 4: Flowchart of While loop, programiz.com)

- For example,

```
speed=55
max_speed=60

while speed<=max_speed:
        print("You are driving at a safe speed: ",speed)
        speed=speed+1
```

WHILE LOOP

- The output of the code would be :

```
You are driving at safe speed:  55
You are driving at safe speed:  56
You are driving at safe speed:  57
You are driving at safe speed:  58
You are driving at safe speed:  59
You are driving at safe speed:  60
```

- So we can see that the loop will execute the code inside the loop until the given condition is met
- The while loop will be running infinitely if the condition is always true

FOR LOOP

- For loop is used when there is a need to iterate the code for a sequence like a list or tuple or a range
- For example,

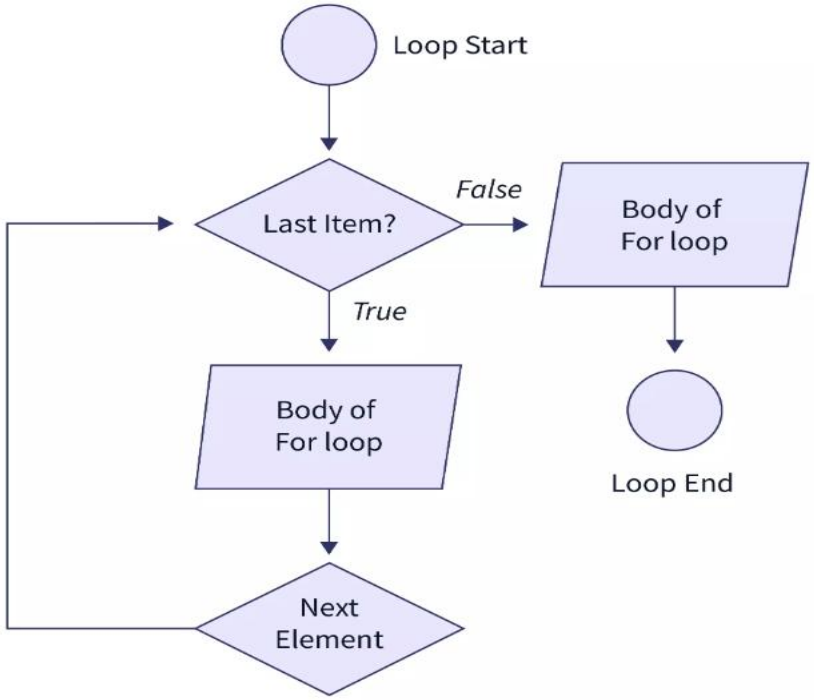
```
cars=[“ferrari, “ford”,”lamborgini”]
for x in cars:
    print(x)
```

The output of the program would be

```
Ferrari
ford
lamborgini
```

- For loop can also be looped for a string

```
for x in “ferrari”
    print(x)
```



(Figure 5: Flowchart of Forloop, scaler.com)

FOR LOOP

- The loop can be used within a specific range using range function

```
for x in range(5,50,5):
    print(x)
```

The output of the program would be:

```
5
10
15
20
25
30
35
40
45
```

- A third value can be specified in the range according to the need. Here an increment factor is used.

WHILE LOOP VS FOR LOOP KEY DIFFERENCES

COMPARING POINT	WHILE LOOP	FOR LOOP
Iterations	While loop is used when the desired iterations are unknown	For loop is used when the desired iterations are known
Initialization nature	It can be repeated for every iterations done	It cannot be repeated when it is done
Function	There are no functions like range to iterate	For iterating range function can be used in the loop
Absence of Conditions	When there is no condition then it will run infinitely	When there is absence of condition it will throw an compiler error
Speed of the loop	This loop is slower than the for loop	For loop is faster than the while loop

BREAK STATEMENT

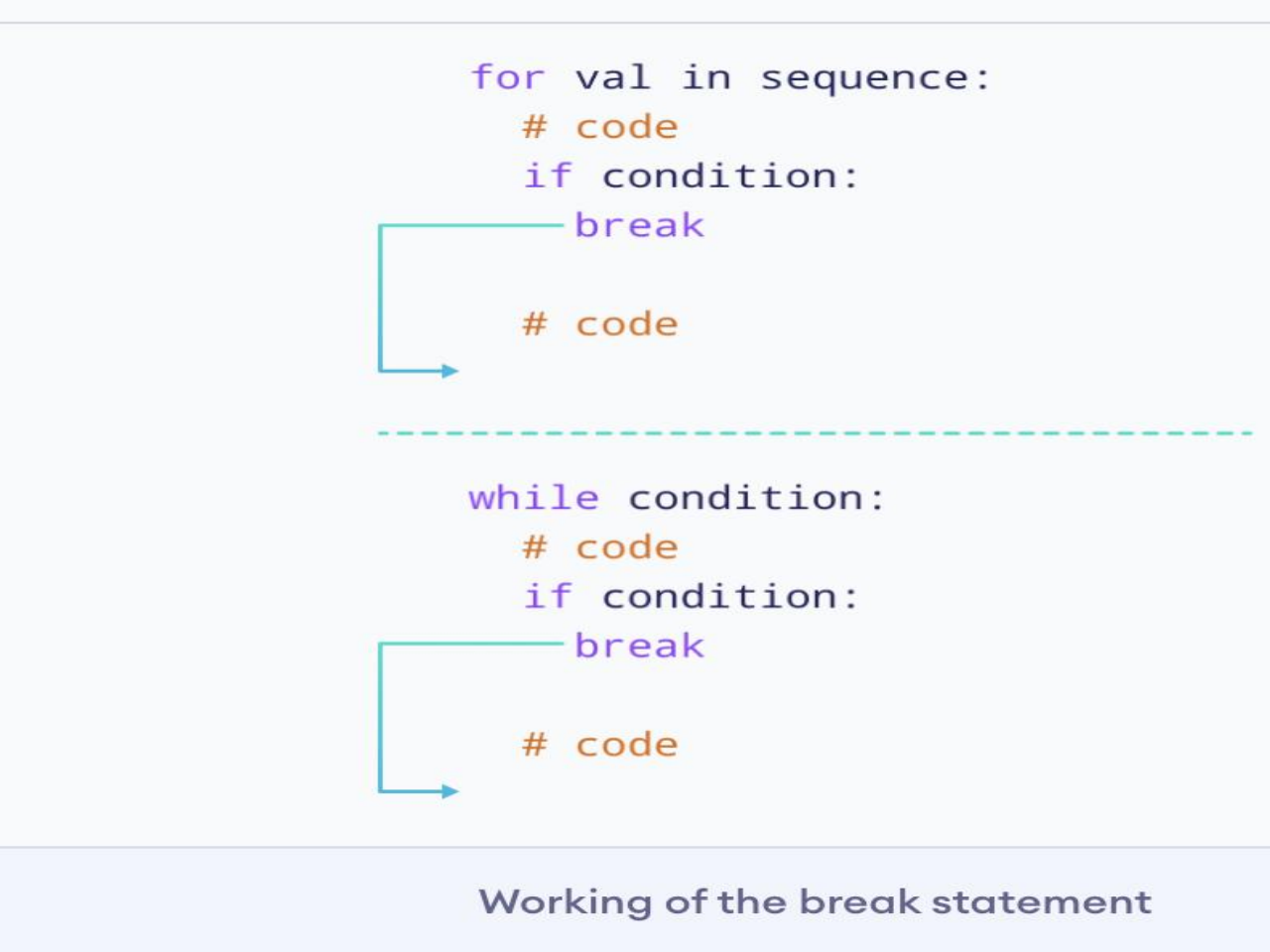
- The break statement is used in the loop to immediately end the loop
- Break statement can be used in both For and While loops
- The loops ends when it faces the break statement
- The flowchart shows the working of the break statement
- For example,

Break statement with for loop

```
for i in range(6):
    if i==5:
        break
    print(i)
```

The output would be: 0 1 2 3 4

- Since we have used Break when i==5, the loop will iterate till i==4 and when it encounters i==5, it immediately terminates and exits the loop



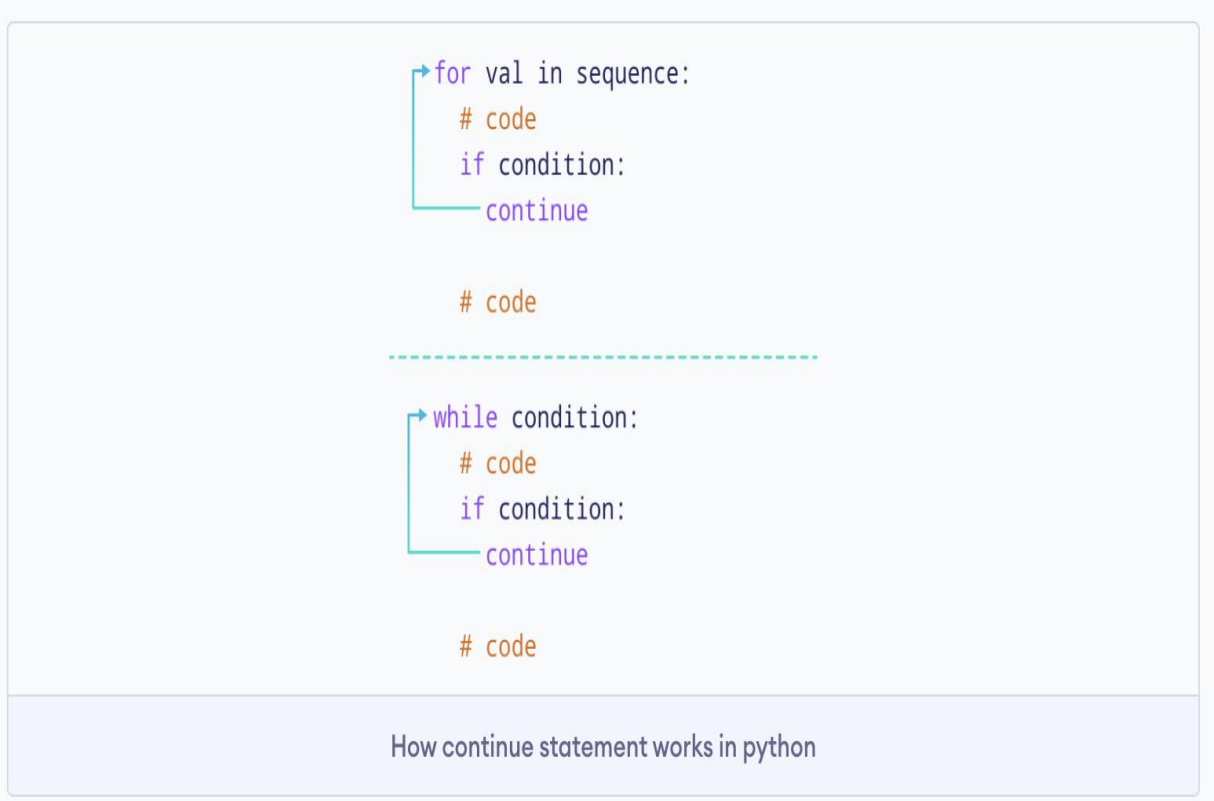
(Figure 6: Working of the break statement, programiz.com)

CONTINUE STATEMENT

- Continue statement is used I n the loop when we just want to skip the current iteration and continue to the next iteration
- The flowchart shows the working of the continue statement
- Continue statement can also be used in for and while loops
- For example,

  Continue statement with while loop

```
person=5
while person<5:
    person+=1
    if person==3:
        continue
    print(person)
```



(Figure 7: Working of continue statement, programiz.com)

- The output would be : 0 1 2 3 4
- Here since we used to continue if the person ==3, the loop will skip the iteration if the person==3 and goes to the next iteration without printing the value 3

ALPHANUMERICAL FUNCTIONS

str.isdigit() :

- This function searches if the given string contains only digits
- For example,

```
var="23028194"
v=var.isdigit()
print(v)
```

- The output would be True as the string contains only numbers
- It gives output as False if there is are no numbers in the string variable

ALPHANUMERICAL FUNCTIONS

str.isupper() :

- This function searches if the given string contains only capital letters
- For example,

```
char="FORD VS FERRARI"
c=char.isupper()
print(c)
```

- The output would be True as the string contains letters which are all in capital
- It gives output as False if the letters are not capital in the string variable

ALPHANUMERICAL FUNCTIONS

str.find() :

- This function finds if the given string contains the letter or a word specified in the function
- It goes from one position to another and gives the position of the specified letter or word if it is present
- For example,

```
name="vettel"
name1=name.find(l)
print(name1)
```

- The output would be 6 as the string contains letter 'l' in the 6th position
- It gives output as –1 if the letter or word specified in the function is not present in the string variable

ALPHANUMERICAL FUNCTIONS

str.count() :

- This function counts how many times a letter or a word specified in the function has appeared in the given string
- It return the number of times the letter or word has appeared
- For example,

```
myname="messi"
count=myname.count("s")
print(count)
```

- The output would be 2 as the letter 's' has appeared 2 times in the given string "messi"
- We can also specify the starting and ending positions, so that the function will count only within that range

```
example: count=myname.count("s",1,2)
```

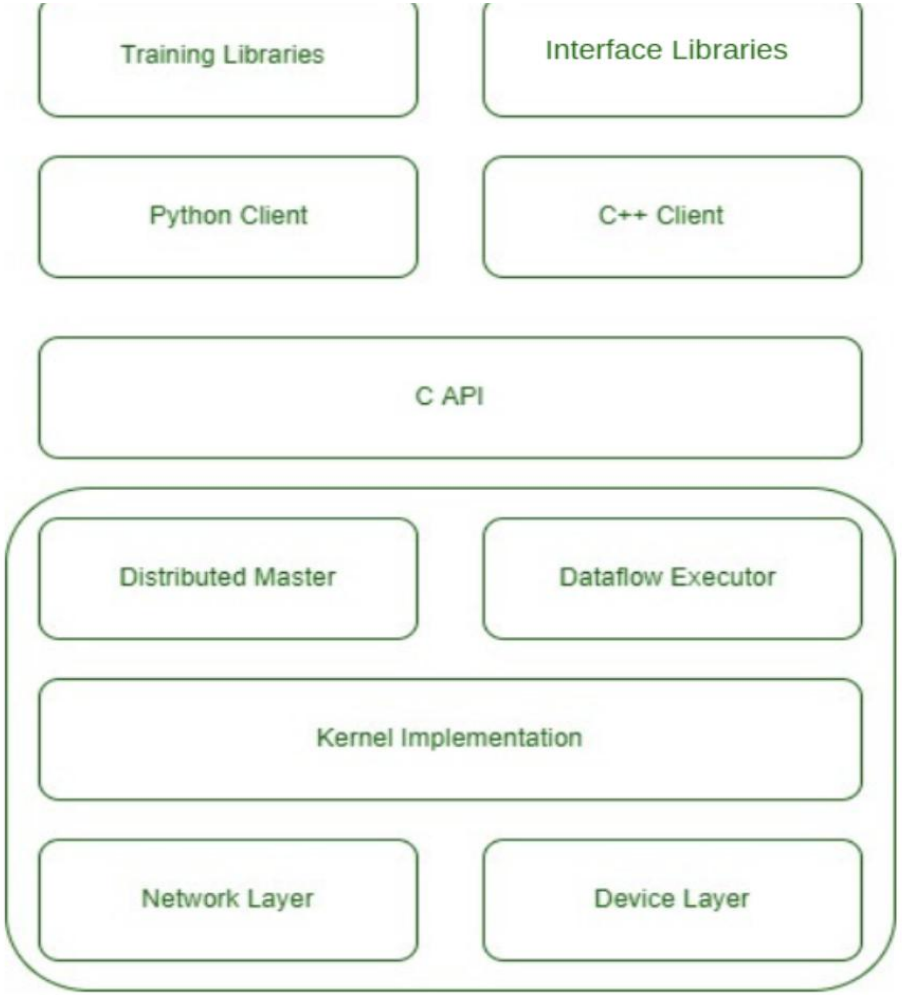
TENSORFLOW

- Tensorflow is an open source library mainly used for large numerical calculation and also for huge machine learning tasks
- It was first created by Google brain team, Google in the year 2015
- Tensorflow uses python or java script in the front end for easeness and executes the given tasks with high performance c++ at the backend
- Tensorflow works on the basis of graph database
- It makes the user to create dataflow graphs which describes on series of nodes
- This can be run on a local machine, CPU, GPU, IOS, Android or as a cluster in a cloud

TENSORFLOW

Architecture

- The diagram shows the architecture of the tensorflow
- The device layer is the first layer, and It allows to connect to devices like CPU, GPU or TPU
- The network layer enables to connect to other machines when on a distributed network
- The kernel implementation layer is the second layer and it contains applications related to machine learning
- The distributed master and data flow executor are the third layer
- The distributed master distributes the workload among the devices and data flow executor executes data flow graphs efficiently
- The fourth layer consists of functionalities in API which is implemented C because it is fast and can run on any system



(Figure 8: Tensorflow high-level Architecture, geeksforgeeks.org

- The next layer acts as a support system for python and c++ clients and last layer libraries implemented in python and c++

T E N S O R F L O W

How it can be used in python:

- Tensorflow supports latest versions of python 3 and, also it may run on the older versions of python
- Tensorflow works on the basis of nodes and tensors which are objects in python and hence the applications in tensorflow are python applications
- The nodes and layers of the tensorflow are linked through the keras library in python
- The libraries for tranformations in tensorflow are available in high performance c++ binaries and python acts as bridge between that and the high-level programming parts that we use generally
- The keras library in python allows us to code a model with three tensorflow layers in a few lines of codes

Use cases:

Tensorflow is mainly used for

- Image classification
- Natural language Processing (NLP) — translation, sentimental analysis
- Time series analysis — stock market, weather analysis
- Reinforcement learning — Games, Robots
- Anomaly detection — identifying pattern in security related tasks

REFERENCES

- Atitude (2020) *supervised vs unsupervised vs reinforcement.* Available at: https://www.aitude.com/supervised-vs-unsupervised-vs-reinforcement/ (Accessed 10 March 2023)
- DataRobot (2020) *Active learning machine learning: what is it and how it works.* Available at: https://www.datarobot.com/blog/active-learning-machine-learning/#:~:text=What%20is%20active%20learning%3F,the%20pool%20of%20unlabeled%20data (Accessed 10 March 2023)
- TheSoceityPages (2009) *Nikon Camera Says Asians: People Are Always Blinking.* Available at: https://thesocietypages.org/socimages/2009/05/29/nikon-camera-says-asians-are-always-blinking/ (Accessed 10 March 2023)
- Geeksforgeeks (2021) *Markov Decision Process.* Available at: https://www.geeksforgeeks.org/markov-decision-process/ (Accessed 11 March 2023)
- Spiceworks (2022) *what is Markov Decision Process? Definition, Working, and Examples* . Available at: https://www.spiceworks.com/tech/artificial-intelligence/articles/what-is-markov-decision-process/#:~:text=A%20Markov%20decision%20process%20(MDP)%20refers%20to%20a%20stochastic%20decision,makes%20sequential%20decisions%20over%20time (Accessed 11 March 2023)
- Programiz (2022) *Python while loop.* Available at: https://www.programiz.com/python-programming/while-loop (Accessed 11 March 2023)

REFERENCES

- Scaler (2022) *What is the difference between For and While loop in Python?.* Available at: https://www.scaler.com/topics/difference-between-for-and-while-loop-in-python/ (Accessed 11 March 2023)
- Programiz (2022) *Python Break and Continue.* Available at: https://www.programiz.com/python-programming/break-continue (Accessed 11 March 2023)
- InfoWorld (2022) *What is Tensorflow? The Machine learning library explained.* Available at: https://www.infoworld.com/article/3278008/what-is-tensorflow-the-machine-learning-library-explained.html (Accessed 12 March 2023)
- Geeksforgeeks (2023) *Architecture of Tensorflow.* Available at: https://www.geeksforgeeks.org/architecture-of-tensorflow/ (Accessed 12 March 2023)