VGP337 - Neural Network & Machine Learning

Instructor: Peter Chan

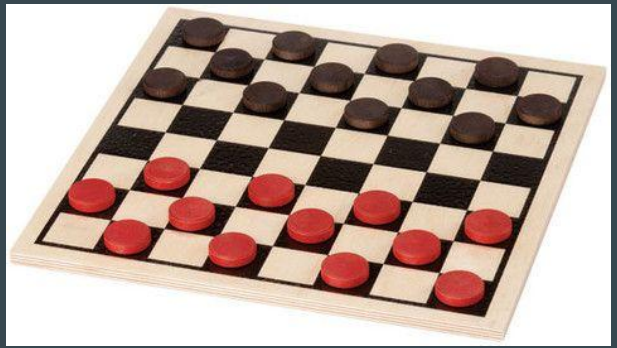
What is Machine Learning?

- A subfield of Artificial Intelligence.
- Study of computer algorithms that improve automatically through experience.
- The term Machine Learning was popularized in 1959 by Arthur Samuel, an American pioneer in the field of computer gaming and AI.
- He stated that “It gives computers the ability to learn without being explicitly programmed”.

A more concise definition

- In 1997, Tom Mitchell gave a more mathematical definition: “A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E”.

Learning Problems

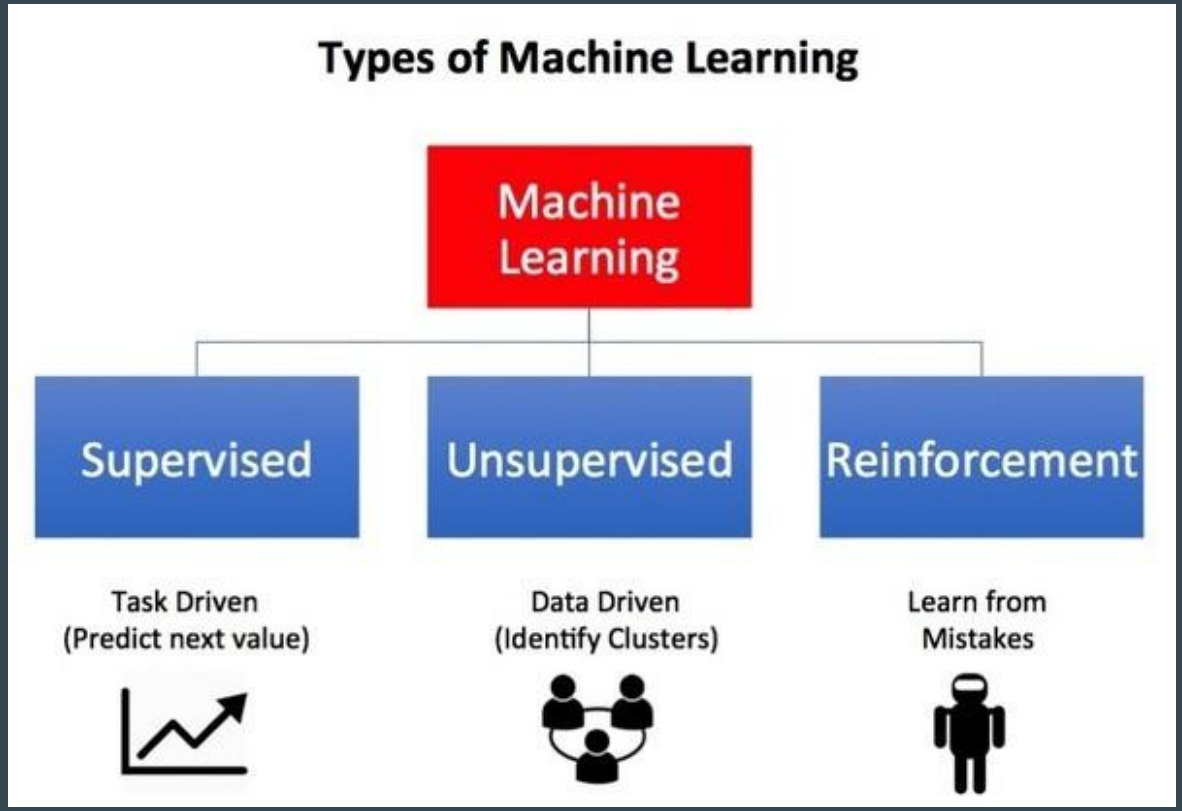


- Example A:
  - A checkers learning problem:
    - Task T: playing checkers
    - Performance measure P: percent of games won against opponents
    - Training experience E: playing practice games against itself
- Example B:
  - A handwriting recognition learning problem:
    - Task T: recognize and classifying handwritten words within images
    - Performance measure P: percent of words correctly classified
    - Training experience E: database of handwritten words with given classifications

Types of learning algorithms

- There are many different types of machine learning algorithms. They are categorized largely based on:
  - The approach to improve performance
  - The type of input and output data
  - The type of task or problem they can solve
- At the high level, there are three main categories:
  - Supervised Learning
  - Unsupervised Learning
  - Reinforcement Learning

Types of learning algorithms

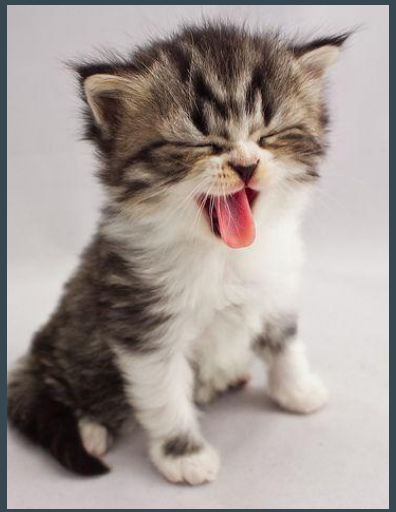


Supervised Learning

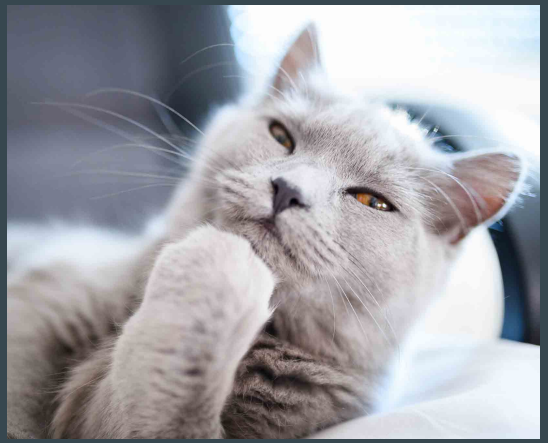
- In supervised learning, we are given a set of data containing both the input and output of a system that we are trying to model
- In other words, there is a teacher that tells the system what the answer should be given a set of attributes
- The idea is that once the algorithm have seen enough examples, it can accurately predict the output when presented data that it has not seen before

Supervised Learning

This is a cat



This is also a cat



Hmm ...

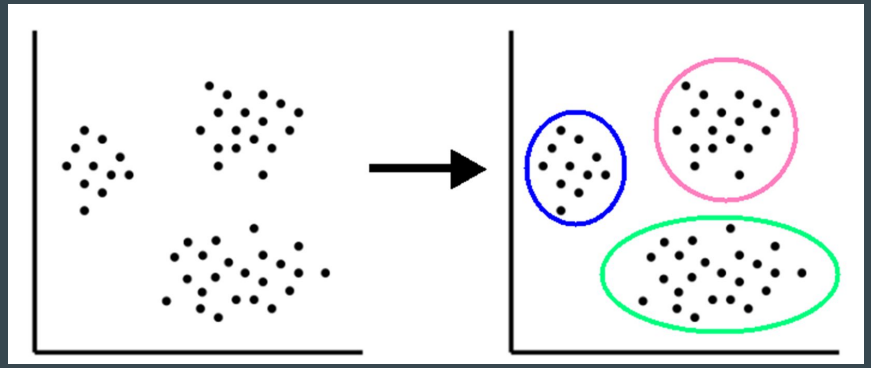
Supervised Learning Examples

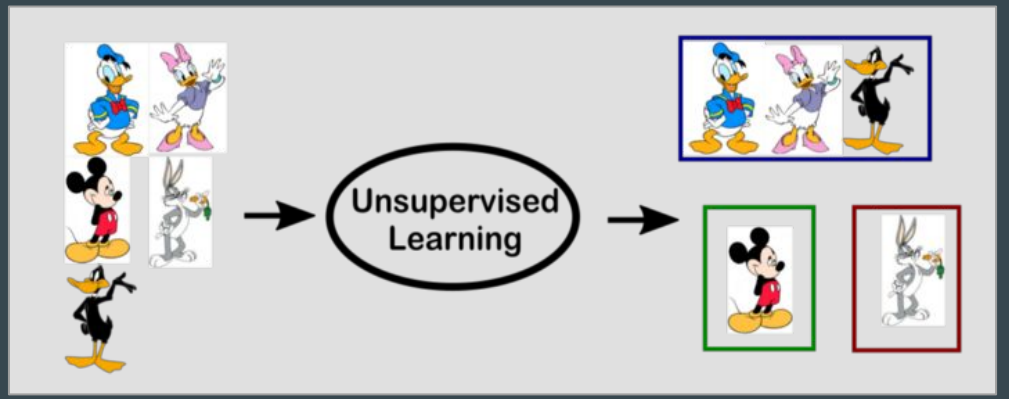
- Applications of supervised learning algorithms include:
  - Speech Recognition
    - Google Assistant
    - Siri
    - Alexa
  - Spam Email Filter
  - Object and Face Recognition
    - Windows Hello
    - Face ID
    - Video Surveillance
    - Fingerprint scanner

Unsupervised Learning

- On the other hand, data for an unsupervised learning algorithm is not labeled
- This means that no one knows what the answer should be, nor how many answers are there
- In unsupervised learning, the goal is for the algorithm to observe and identify any potential patterns or similarities in the data
- Sometimes referred as clustering or classification

Unsupervised Learning





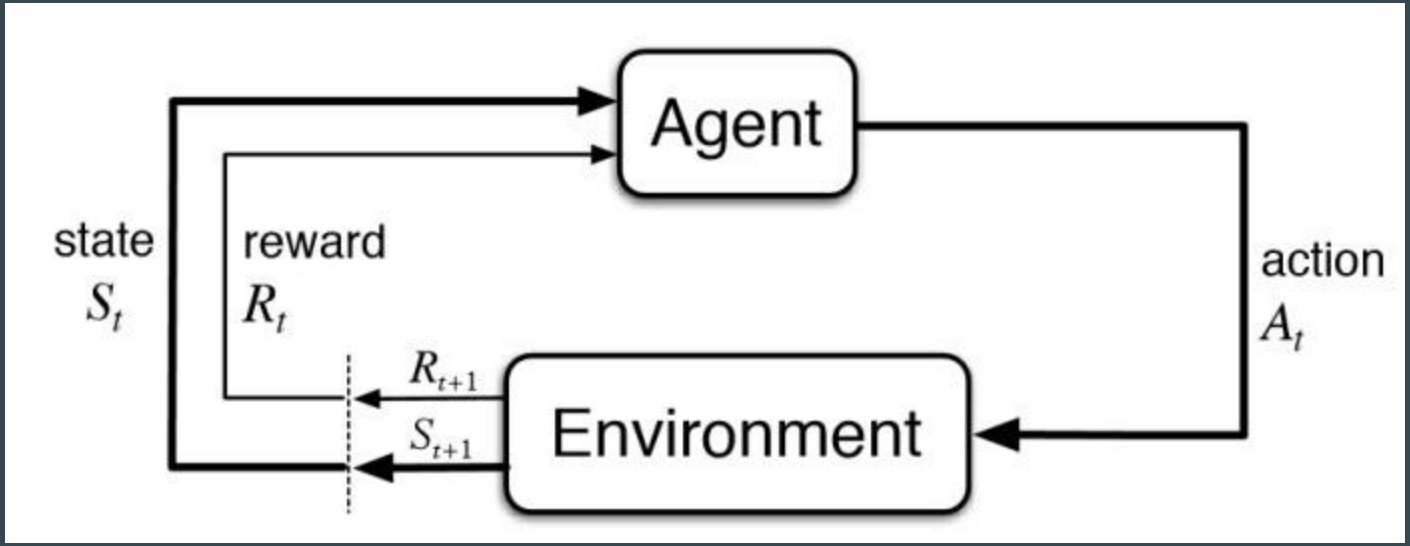
Unsupervised Learning Examples

- Applications of unsupervised learning algorithms include:
  - Market segmentation for targeting appropriate customers
  - Anomaly/fraud detection
  - Google Photos automatically group pictures

Reinforcement Learning

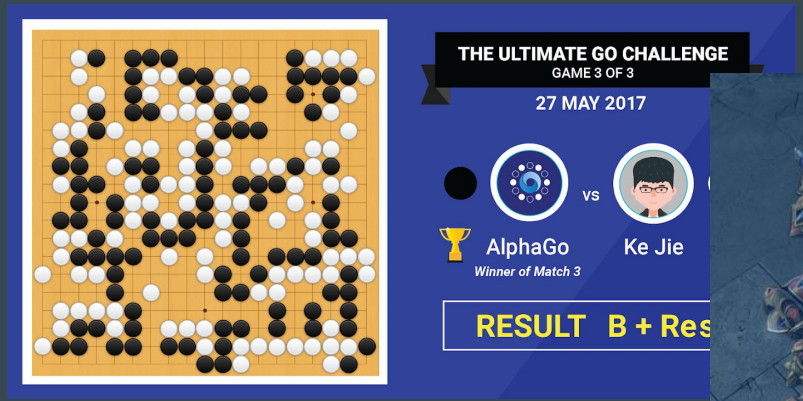
- Reinforcement learning is an area of machine learning focused on taking actions that will maximize the total, delayed reward
- It differs from supervised learning in not requiring labelled data or whether the action taken is the best
- Instead the focus is on balancing between exploration (in hope of gaining new information) and exploitation (cashing in on learned knowledge)

Reinforcement Learning



Reinforcement Learning Examples

- Applications of reinforcement learning algorithms include:
  - Robotics for industrial automation
  - Business strategy and trading
  - Games
    - AlphaGo Zero (https://deepmind.com/blog/article/alphago-zero-starting-scratch)
    - AlphaStar (https://deepmind.com/blog/article/alphastar-mastering-real-time-strategy-game-starcraft-ii)



Python

Python

- Python is an interpreted, object-oriented, high-level programming language
- It has high-level data structures, dynamic typing, dynamic binding, plus a large repository of high quality, off-the-shelf packages that can be easily added to any software project
- Due to its popularity, Python is used for a wide variety of applications
- In particular, it is the language of choice for a lot of data science and machine learning libraries including Google's TensorFlow and Facebook's PyTorch

Python Crash Course

- Comments
- Types and Variables
- Strings
- Data Structures
- Functions
- Conditionals
- Loops
- Modules
- Classes

Python - Comments

- Single line comments in Python starts with the character '#'

```
# This is a comment in Python
```

- For multiline comments (sometimes called docstrings), you can use 3 single quotes or double quotes

```
'''
This is a
multiline comment.
'''
```

Python - Variables

- Python is dynamically typed, meaning that you do not need to specify the type of a variable.
- Variable names are case sensitive (count and COUNT are different variables)
- The name must start with a letter or an underscore
- Numbers can be used only after the first letter

```
x = 1                       # int
y = 2.5                     # float
name = ‘Peter’              # str, can use ‘abc’ or “abc”
is_alive = True             # bool, can be True or False
a, b, c = (1, 2.3, True)    # tuple assignment
```

Python - print()

- When in doubt, you can always use the print() function to verify that you have the correct type or values

```
x = 3.14                    # int variable
print(type(x), x)           # <class 'float'> 3.14
x = int(x)                  # casting
print(type(x), x)           # <class 'int'> 3
x = str(x + x)              # casting, arithmetics
print(type(x), x)           # <class 'str'> 6
x = (x == x)                # comparison
print(type(x), x)           # <class 'bool'> True
```

Python - Strings

- String concatenation and formatting are pretty straightforward

```
name = ‘Peter’
age = 39
print(‘My name is ‘ + name + ‘ and I am ‘ + str(age))
print(‘My name is {n} and I am {a}’.format(n=name, a=age))
print(f’My name is {name} and I am {age}’)
```

- There are also a bunch of useful string methods you can read more about here: https://docs.python.org/3/library/string.html

Python - Lists

- A list is a collection which is ordered and mutable, it also allows duplicates
- Lists are a part of the core Python language and does not need any libraries

```
numbers = [1, 2, 3, 4, 5]     # list of int
weapons = [‘sword’, ‘axe’]    # list of str
print(weapons[1])             # axe, 0 based indexing
print(len(weapons))           # 2
weapons.append(‘hammer’)      # insert to end of list
print(weapons)                # ['sword', 'axe', 'hammer']
```

- For more, see https://docs.python.org/3/tutorial/datastructures.html#more-on-lists

Python - Other Data Structures

- Most of the time, you will be using Python list, however there are other built-in containers that may be useful:
  - List - ordered, mutable, allows duplicates
  - Tuple - ordered, immutable, allows duplicates
  - Set - unordered, unindexed, no duplicates
  - Dictionary - unordered, mutable and indexed, no duplicates
- More info here: https://docs.python.org/3/tutorial/datastructures.html#

Python - Functions

- In Python, we do not use curly braces, instead we use tabs and colons
- *Note* code block and scope in Python is determined by indentation!

```
def sayHello(name):
    print(f'Hello {name}')

def getSum(num1, num2):
    total = num1 + num2
    return total
```

Python - Conditionals

- If-else statements in Python is pretty similar other languages

```
x, y = (10, 5)
if x > y:                # can also use ==, !=, >, <, >=, <=
    print(f‘{x} is greater than {y}’)
elif x < y:
    print(f’{x} is less than {y}’)
else
    print(f’{x} is equal to {y}’)
```

- You can also use `and`, `or`, and `not` to combine conditions

Python - For Loops

- A for loop is used for iterating over a sequence, which can be a list, a tuple, a dictionary, a set, or a string

```
students = [‘Jacky’, ‘Joe’, ‘John’]
for person in students:
    print(person)

for i in range(len(students)):      # or use range(0, 3)
    print(students[i])
```

- You can also use `break` or `continue` to exit the loop or skip an element

Python - While Loops

- A while loop runs until a condition is no longer true

```
count = 0
while count <= 10:
    print(str(count))
    count += 1
```

Python - Modules

- A module is basically like a header in C++ containing a set of features to be used in your application
- There are core Python modules, custom modules you can implement yourself, as well as external modules you can install using the pip package manager

```
import datetime as dt
from time import time

today = dt.date.today()
now = time()
print(f'{today} at {now}')
```

Python - Class

- Python supports class and allows you to use object-oriented programming
- You can define methods and add member variables as usual, interestingly, all member variables are public in Python

```
class Player:
    def __init__(self, name):    # reserved name for ctor
        self.name = name         # self is same as this ptr
    def getName(self):
        return self.name
```

- You can also use inheritance as well