BITS F464 - MACHINE LEARNING

PROJECT

SUBMITTED BY -

AMET VIKRAM - 2016A7PS0050G

JAGSIFAT SINGH MAKKAR - 2016A7PS0072G

THE PROBLEM

A Reinforcement learning model to be designed that'll balance the pole on a cart which is running on a frictionless flat-terrain with a noisy set of parameters such as action and sensors. The values for gravity and friction randomly vary at each step for the same episode. The following sub-tasks are required to be solved (for the given parameters in the code attached):
1. The cart-pole problem with random variation in gravity and friction.
2. The cart-pole problem with the previous setting's noise and noisy controls. This means that the cart's force in the desired direction may be less/more than expected.
3. The cart-pole problem with both the previous modifications and noisy sensors/sensor observations of the pole angle at any moment.

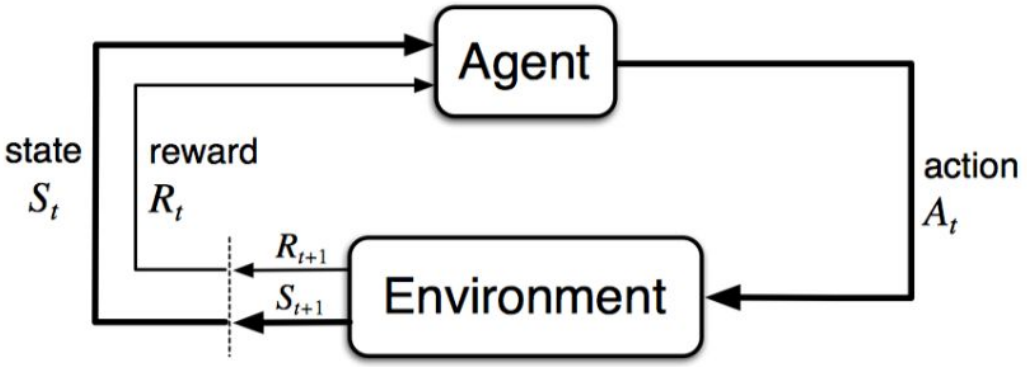
THE COMPONENTS

- **REINFORCEMENT LEARNING**

  Reinforcement learning is the training of machine learning models to make a sequence of decisions. The agent learns to achieve a goal in an uncertain, potentially complex environment. In reinforcement learning, the machine faces a game-like situation. The computer employs trial and error to come up with a solution to the problem. To get the machine to do what the programmer wants, and gets either rewards or penalties for the actions it performs. Its goal is to maximize the total reward. Although the reward policy is already set–that is, the rules of the game– no hints or suggestions are given to the model for how to solve the game. It's up to the model to figure out how to perform the task to maximize the reward, starting from totally random trials and finishing with sophisticated tactics and superhuman skills. By leveraging the power of search and many trials, reinforcement learning is currently the most effective way to hint a machine's creativity.

Characteristics of Reinforcement Learning :

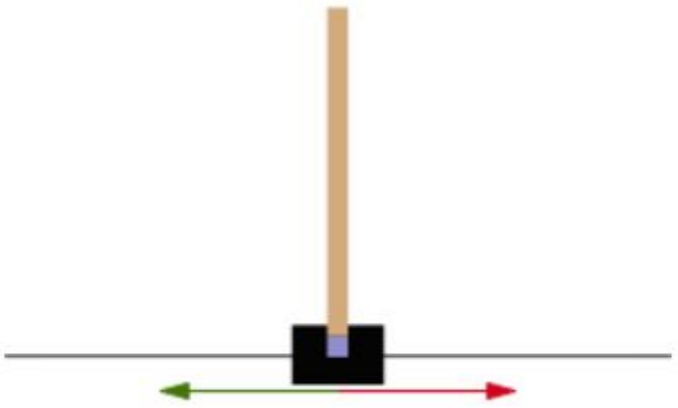
a. It is concerned with goal-directed learning and decision-making.
b. An agent learns from experiences it gains by interacting with the environment. In Supervised Learning we cannot affect the environment.
c. Rewards are often delayed in time and the agent tries to maximize a long-term goal. For example, one may need to make seemingly suboptimal moves to reach a winning position in a game.
d. An agent interacts with the environment via states, actions and rewards.



The Canonical Agent-Environment Feedback Loop

- **THE CART-POLE PROBLEM (An example of Reinforcement Learning)**

  Cartpole - known also as an "Inverted Pendulum" is a pendulum with a center of gravity above its pivot point. It's unstable, but can be controlled by moving the pivot point under the center of mass. The goal is to keep the cartpole balanced by applying appropriate forces to a pivot point.



Cartpole schematic drawing

a. Violet square indicates a pivot point.
b. Red and green arrows show possible horizontal forces that can be applied to a pivot point.

c. A pole is attached by an un-actuated joint to a cart, which moves along a frictionless track. The system is controlled by applying a force of +1 or -1 to the cart. The pendulum starts upright, and the goal is to prevent it from falling over. A reward of +1 is provided for every timestep that the pole remains upright. The episode ends when the pole is more than 15 degrees from vertical, or the cart moves more than 2.4 units from the center.

- **MARKOV-DECISION PROCESSES (MDPs)**

The agent must formally work through a theoretical framework known as a Markov Decision Process which consists of a decision (what action to take?)

$$S_0, A_0, R_1, S_1, A_1, R_2, \dots$$

to be made at each state. This gives rise to a sequence of states, actions and rewards known as a trajectory, and the objective is to maximize this set of rewards.

More formally, we look at the Markov Decision Process framework. A (Discounted) Markov Decision Process (MDP) is a tuple (S,A,R,p,γ), such that, where S_t, S_(t+1) ∈ S (state space), A_(t+1) ∈ A (action space), R_(t+1), R_t ∈ R (reward space), p defines the dynamics of the process and G_t is the discounted return.

In simple words, an MDP defines the probability of transitioning into a new state, getting some reward given the current state and the execution of an action. This framework is mathematically pleasing because it is First-Order Markov. This is just a fancy way of saying that anything that happens next is dependent only on the present and not the past. It does not matter how one arrives at the current state as long as one does. Another important part of this framework is the discount factor γ. Summing these rewards over time with a varying degree of importance to the rewards from the future leads to a notion of discounted returns. As one might expect, a higher γ leads to higher

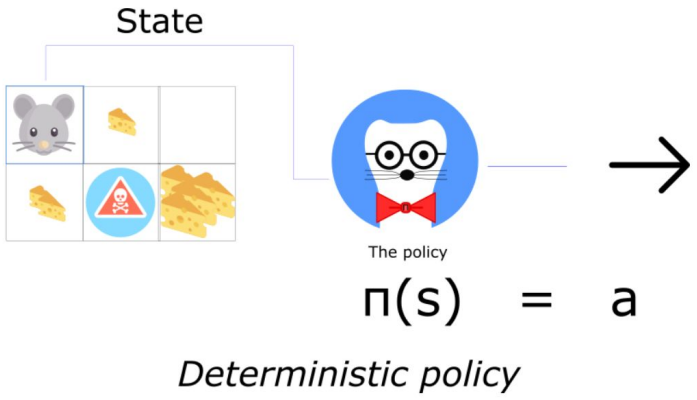
sensitivity for rewards from the future. However, the extreme case of $\gamma=0$ doesn't consider rewards from the future at all.

- **POLICY-GRADIENTS**

In policy-based methods, instead of learning a value function that tells us what is the expected sum of rewards given a state and an action, we learn directly the policy function that maps state to action (select actions without using a value function). It means that we directly try to optimize our policy function π without worrying about a value function. We'll directly parameterize π (select an action without a value function).

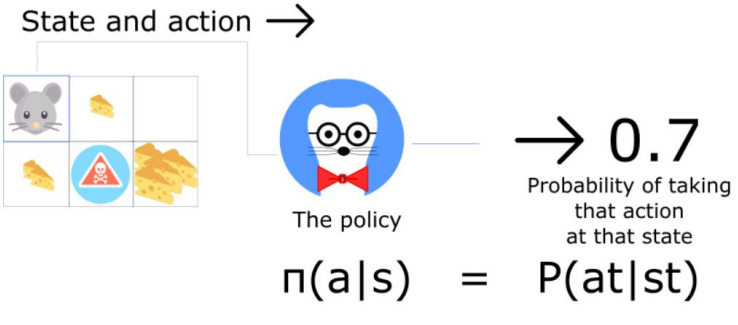
Two types of policy :

Deterministic : A deterministic policy is policy that maps state to actions. You give it a state and the function returns an action to take. Deterministic policies are used in deterministic environments. These are environments where the actions taken determine the outcome. There is no uncertainty.



Deterministic policy

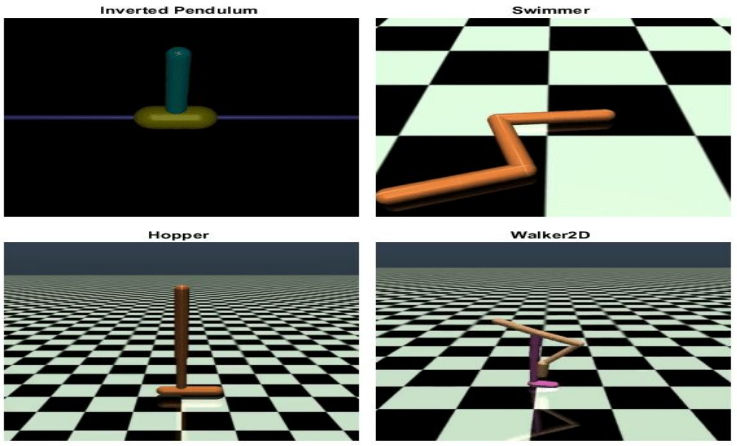
Stochastic : A stochastic policy outputs a probability distribution over actions. It means that instead of being sure of taking action *a*, there is a probability we'll take a different one. The stochastic policy is used when the environment is uncertain. We call this process a Partially Observable Markov Decision Process (POMDP).



Stochastic policy

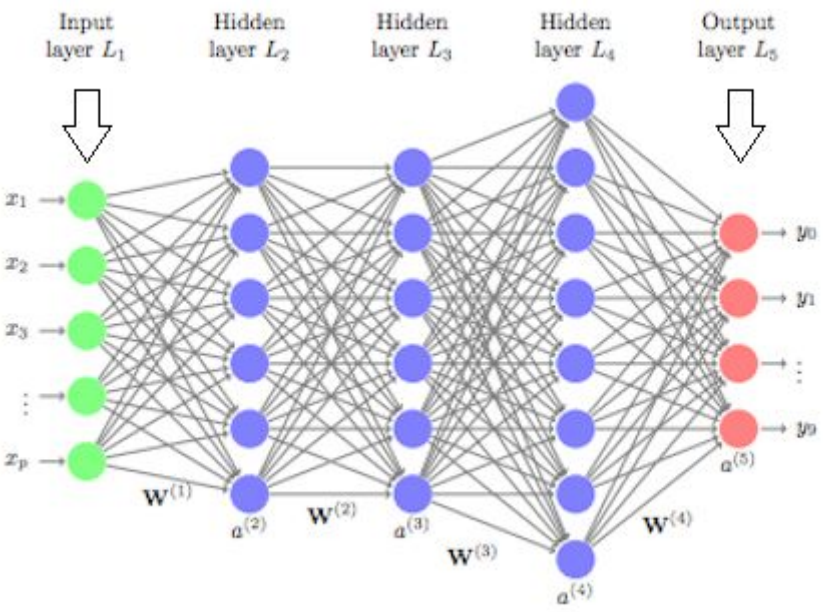
- **OpenAI-gym**

  Gym is a toolkit for developing and comparing reinforcement learning algorithms. It makes no assumptions about the structure of your agent, and is compatible with any numerical computation library, such as TensorFlow or Theano. The gym library is a collection of test problems and environments that you can use to work out your reinforcement learning algorithms. These environments have a shared interface, allowing you to write general algorithms. Gym comes with a diverse suite of environments that range from easy to difficult and involve many different kinds of data such as Classical control, Algorithmic, Atari etc. RL research is also slowed down by two factors, that is, the need for better benchmarks and the lack of standardization of environments used in publications, and gym is an attempt to fix both the problems



THE MODEL

The approach used to solve all the three tasks is **Deep Neural Network.** With various variations of different layers, number of units in layers, and extensive hyperparameter tuning.



The relevant screenshots for the model summary and training are as follows for all the tasks :-

```
PS C:\Users\Kushagra\mlproject> python.exe .\task2_solution.py                                                    Using TensorFlow backend.
2020-04-21 08:59:21.501307: W tensorflow/stream_executor/platform/default/dso_loader.cc:55] Could not load dynamic library 'cudart64_100.dll'; dlerror: cudart64_100.dll not found
2020-04-21 08:59:21.506219: I tensorflow/stream_executor/cuda/cudart_stub.cc:29] Ignore above cudart dlerror if you do not have a GPU set up on your machine.
CartPoleEnv - Version 0.2.0, Noise case: 1
C:\Users\Kushagra\.virtualenvs\mlproject-sv2LImfg\lib\site-packages\gym\logger.py:30: UserWarning: ▒[33mWARN: Box bound precision lowered by casting to float32▒[0m
  warnings.warn(colorize('%s: %s'%('WARN', msg % args), 'yellow'))
C:\Users\Kushagra\.virtualenvs\mlproject-sv2LImfg\lib\site-packages\gym\logger.py:30: UserWarning: ▒[33mWARN: Environment '<class 'gym.envs.classic_control.cartpole.CartPoleEnv'>' has deprecat
ed methods '_step' and '_reset' rather than 'step' and 'reset'. Compatibility code invoked. Set _gym_disable_underscore_compat = True to disable this behavior.▒[0m
  warnings.warn(colorize('%s: %s'%('WARN', msg % args), 'yellow'))
WARNING:tensorflow:From C:\Users\Kushagra\.virtualenvs\mlproject-sv2LImfg\lib\site-packages\tensorflow_core\python\ops\resource_variable_ops.py:1630: calling BaseResourceVariable.__init__ (fro
m tensorflow.python.ops.resource_variable_ops) with constraint is deprecated and will be removed in a future version.
Instructions for updating:
If using Keras pass *_constraint arguments to layers.
Model: "model_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         (None, 4)                 0
_________________________________________________________________
dense_1 (Dense)              (None, 200)               1000
_________________________________________________________________
dropout_1 (Dropout)          (None, 200)               0
_________________________________________________________________
dense_2 (Dense)              (None, 400)               80400
_________________________________________________________________
dropout_2 (Dropout)          (None, 400)               0
_________________________________________________________________
dense_3 (Dense)              (None, 2)                 802
=================================================================
Total params: 82,202
Trainable params: 82,202
Non-trainable params: 0
```

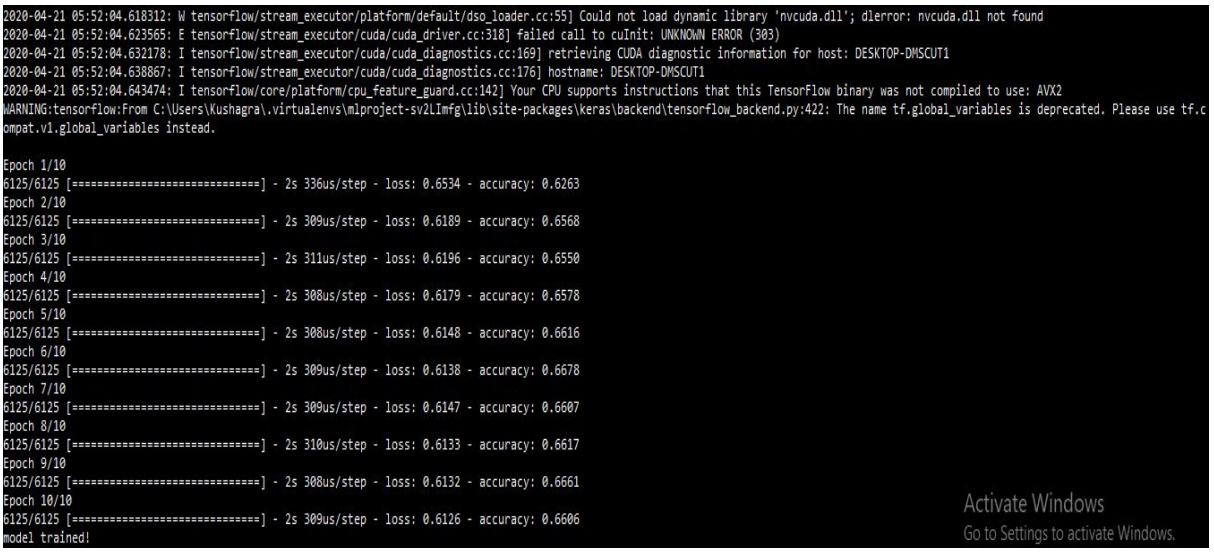
```
2020-04-21 08:59:36.679320: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:176] hostname: DESKTOP-DMSCUT1
2020-04-21 08:59:36.766573: I tensorflow/core/platform/cpu_feature_guard.cc:142] Your CPU supports instructions that this TensorFlow binary was not compiled to use: AVX2
WARNING:tensorflow:From C:\Users\Kushagra\.virtualenvs\mlproject-sv2LImfg\lib\site-packages\keras\backend\tensorflow_backend.py:422: The name tf.global_variables is deprecated. Please use tf.c
ompat.v1.global_variables instead.

Epoch 1/20
5638/5638 [==============================] - 1s 236us/step - loss: 0.6732 - accuracy: 0.6204
Epoch 2/20
5638/5638 [==============================] - 0s 25us/step - loss: 0.6663 - accuracy: 0.6520
Epoch 3/20
5638/5638 [==============================] - 0s 22us/step - loss: 0.6676 - accuracy: 0.6412
Epoch 4/20
5638/5638 [==============================] - 0s 25us/step - loss: 0.6665 - accuracy: 0.6467
Epoch 5/20
5638/5638 [==============================] - 0s 25us/step - loss: 0.6666 - accuracy: 0.6563
Epoch 6/20
5638/5638 [==============================] - 0s 22us/step - loss: 0.6636 - accuracy: 0.6541
Epoch 7/20
5638/5638 [==============================] - 0s 22us/step - loss: 0.6631 - accuracy: 0.6511
Epoch 8/20
5638/5638 [==============================] - 0s 25us/step - loss: 0.6648 - accuracy: 0.6582
Epoch 9/20
5638/5638 [==============================] - 0s 25us/step - loss: 0.6607 - accuracy: 0.6545
Epoch 10/20
5638/5638 [==============================] - 0s 25us/step - loss: 0.6624 - accuracy: 0.6543
Epoch 11/20
5638/5638 [==============================] - 0s 22us/step - loss: 0.6586 - accuracy: 0.6465
Epoch 12/20
5638/5638 [==============================] - 0s 25us/step - loss: 0.6536 - accuracy: 0.6568
Epoch 13/20
5638/5638 [==============================] - 0s 25us/step - loss: 0.6491 - accuracy: 0.6527
Epoch 14/20
5638/5638 [==============================] - 0s 25us/step - loss: 0.6447 - accuracy: 0.6541
Epoch 15/20
5638/5638 [==============================] - 0s 22us/step - loss: 0.6402 - accuracy: 0.6632
Epoch 16/20
5638/5638 [==============================] - 0s 22us/step - loss: 0.6418 - accuracy: 0.6570
Epoch 17/20
5638/5638 [==============================] - 0s 25us/step - loss: 0.6369 - accuracy: 0.6602
Epoch 18/20
5638/5638 [==============================] - 0s 25us/step - loss: 0.6358 - accuracy: 0.6566
Epoch 19/20
5638/5638 [==============================] - 0s 25us/step - loss: 0.6331 - accuracy: 0.6662
Epoch 20/20
5638/5638 [==============================] - 0s 22us/step - loss: 0.6309 - accuracy: 0.6619
model trained!
```



```
PS C:\Users\Kushagra\mlproject> python.exe .\task3_solution.py
Using TensorFlow backend.
2020-04-21 09:21:04.881362: W tensorflow/stream_executor/platform/default/dso_loader.cc:55] Could not load dynamic library 'cudart64_100.dll'; dlerror: cudart64_100.dll not found
2020-04-21 09:21:04.886529: I tensorflow/stream_executor/cuda/cudart_stub.cc:29] Ignore above cudart dlerror if you do not have a GPU set up on your machine.
CartPoleEnv - Version 0.2.0, Noise case: 1
C:\Users\Kushagra\.virtualenvs\mlproject-sv2LImfg\lib\site-packages\gym\logger.py:30: UserWarning: [33mWARN: Box bound precision lowered by casting to float32[0m
  warnings.warn(colorize('%s: %s'%('WARN', msg % args), 'yellow'))
C:\Users\Kushagra\.virtualenvs\mlproject-sv2LImfg\lib\site-packages\gym\logger.py:30: UserWarning: [33mWARN: Environment '<class 'gym.envs.classic_control.cartpole.CartPoleEnv'>' has deprecat
ed methods '_step' and '_reset' rather than 'step' and 'reset'. Compatibility code invoked. Set _gym_disable_underscore_compat = True to disable this behavior.[0m
  warnings.warn(colorize('%s: %s'%('WARN', msg % args), 'yellow'))
WARNING:tensorflow:From C:\Users\Kushagra\.virtualenvs\mlproject-sv2LImfg\lib\site-packages\tensorflow_core\python\ops\resource_variable_ops.py:1630: calling BaseResourceVariable.__init__ (fro
m tensorflow.python.ops.resource_variable_ops) with constraint is deprecated and will be removed in a future version.
Instructions for updating:
If using Keras pass *_constraint arguments to layers.
Model: "model_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         (None, 4)                 0
_________________________________________________________________
dense_1 (Dense)              (None, 200)               1000
_________________________________________________________________
dropout_1 (Dropout)          (None, 200)               0
_________________________________________________________________
dense_2 (Dense)              (None, 300)               60300
_________________________________________________________________
dropout_2 (Dropout)          (None, 300)               0
_________________________________________________________________
dense_3 (Dense)              (None, 200)               60200
_________________________________________________________________
dropout_3 (Dropout)          (None, 200)               0
_________________________________________________________________
dense_4 (Dense)              (None, 300)               60300
_________________________________________________________________
dropout_4 (Dropout)          (None, 300)               0
_________________________________________________________________
dense_5 (Dense)              (None, 2)                 602
=================================================================
Total params: 182,402
Trainable params: 182,402
Non-trainable params: 0
```

```
Epoch 1/20
8827/8827 [==============================] - 2s 183us/step - loss: 0.7517 - accuracy: 0.5753
Epoch 2/20
8827/8827 [==============================] - 0s 39us/step - loss: 0.6590 - accuracy: 0.6265
Epoch 3/20
8827/8827 [==============================] - 0s 40us/step - loss: 0.6490 - accuracy: 0.6356
Epoch 4/20
8827/8827 [==============================] - 0s 39us/step - loss: 0.6368 - accuracy: 0.6540
Epoch 5/20
8827/8827 [==============================] - 0s 40us/step - loss: 0.6289 - accuracy: 0.6514
Epoch 6/20
8827/8827 [==============================] - 0s 38us/step - loss: 0.6281 - accuracy: 0.6558
Epoch 7/20
8827/8827 [==============================] - 0s 40us/step - loss: 0.6272 - accuracy: 0.6541
Epoch 8/20
8827/8827 [==============================] - 0s 39us/step - loss: 0.6230 - accuracy: 0.6575
Epoch 9/20
8827/8827 [==============================] - 0s 40us/step - loss: 0.6244 - accuracy: 0.6591
Epoch 10/20
8827/8827 [==============================] - 0s 39us/step - loss: 0.6268 - accuracy: 0.6514
Epoch 11/20
8827/8827 [==============================] - 0s 40us/step - loss: 0.6246 - accuracy: 0.6565
Epoch 12/20
8827/8827 [==============================] - 0s 39us/step - loss: 0.6218 - accuracy: 0.6550
Epoch 13/20
8827/8827 [==============================] - 0s 40us/step - loss: 0.6229 - accuracy: 0.6575
Epoch 14/20
8827/8827 [==============================] - 0s 38us/step - loss: 0.6278 - accuracy: 0.6520
Epoch 15/20
8827/8827 [==============================] - 0s 38us/step - loss: 0.6229 - accuracy: 0.6618
Epoch 16/20
8827/8827 [==============================] - 0s 39us/step - loss: 0.6209 - accuracy: 0.6572
Epoch 17/20
8827/8827 [==============================] - 0s 39us/step - loss: 0.6251 - accuracy: 0.6523
Epoch 18/20
8827/8827 [==============================] - 0s 38us/step - loss: 0.6243 - accuracy: 0.6576
Epoch 19/20
8827/8827 [==============================] - 0s 39us/step - loss: 0.6281 - accuracy: 0.6528
Epoch 20/20
8827/8827 [==============================] - 0s 38us/step - loss: 0.6209 - accuracy: 0.6563
model trained!
```

```
PS C:\Users\Kushagra\mlproject> python.exe .\task1_solution.py
Using TensorFlow backend.
2020-04-21 05:52:01.531499: W tensorflow/stream_executor/platform/default/dso_loader.cc:55] Could not load dynamic library 'cudart64_100.dll'; dlerror: cudart64_100.dll not found
2020-04-21 05:52:01.537816: I tensorflow/stream_executor/cuda/cudart_stub.cc:29] Ignore above cudart dlerror if you do not have a GPU set up on your machine.
CartPoleEnv - Version 0.2.0, Noise case: 1
C:\Users\Kushagra\.virtualenvs\mlproject-sv2LImfg\lib\site-packages\gym\logger.py:30: UserWarning: ▒[33mWARN: Box bound precision lowered by casting to float32▒[0m
  warnings.warn(colorize('%s: %s'%('WARN', msg % args), 'yellow'))
C:\Users\Kushagra\.virtualenvs\mlproject-sv2LImfg\lib\site-packages\gym\logger.py:30: UserWarning: ▒[33mWARN: Environment '<class 'gym.envs.classic_control.cartpole.CartPoleEnv'>' has deprecated methods '_step' and '_reset' rather than 'step' and 'reset'. Compatibility code invoked. Set _gym_disable_underscore_compat = True to disable this behavior.▒[0m
  warnings.warn(colorize('%s: %s'%('WARN', msg % args), 'yellow'))
WARNING:tensorflow:From C:\Users\Kushagra\.virtualenvs\mlproject-sv2LImfg\lib\site-packages\tensorflow_core\python\ops\resource_variable_ops.py:1630: calling BaseResourceVariable.__init__ (from tensorflow.python.ops.resource_variable_ops) with constraint is deprecated and will be removed in a future version.
Instructions for updating:
If using Keras pass *_constraint arguments to layers.
Model: "model_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         (None, 4)                 0
_________________________________________________________________
dense_1 (Dense)              (None, 64)                320
_________________________________________________________________
dense_2 (Dense)              (None, 128)               8320
_________________________________________________________________
dense_3 (Dense)              (None, 256)               33024
_________________________________________________________________
dense_4 (Dense)              (None, 512)               131584
_________________________________________________________________
dense_5 (Dense)              (None, 1024)              525312
_________________________________________________________________
dense_6 (Dense)              (None, 512)               524800
_________________________________________________________________
dense_7 (Dense)              (None, 256)               131328
_________________________________________________________________
dense_8 (Dense)              (None, 128)               32896
_________________________________________________________________
dense_9 (Dense)              (None, 64)                8256
_________________________________________________________________
dense_10 (Dense)             (None, 2)                 130
=================================================================
Total params: 1,395,970
Trainable params: 1,395,970
Non-trainable params: 0
```



CONTRIBUTIONS

The major challenge in the problem was to come up with the most optimal algorithm to solve the question. We decided to divide our algorithms to study and see which ones perform the best on the problem the algorithms were divided as :-

1) Jagsifat - Deep Q learning
2) Amet - Spinning Up Algos and Deep NN

After studying the group of algos we finally decide on Deep NN to implement, which we can implement most effectively.

After this, the remaining work was divided as follows :-

1) Amet - task1_solution.py , task2_solution.py and half of the report
2) Jagsifat - task3.py and half of the report