

Code Description

The main code is reported in the **Navigation.pyynb** file, which includes information about the requirements and at the beginning runs some code examples to get environment information.

The code used to train Agent is a simple value-based training algorithm known as Deep Q-Network.

The package contains two additional files with (**model.py**, and **dqn_agent.py**) which contains the code for the model and the agent used for the training.

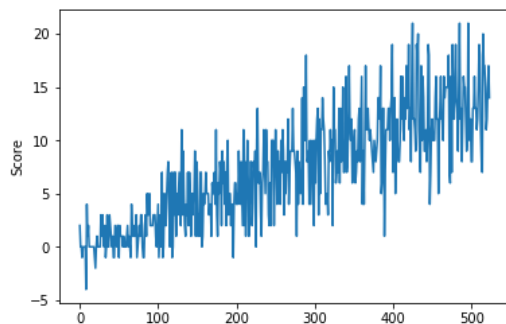
Chosen Params

state_size **37** || action_size **4** || random seed **40**

```
In [8]: import time
timer_start = time.time() #Checking on training start time
scores = dqn(n_episodes=2000, max_t=1000, eps_start=1.0, eps_end=0.01, eps_decay=0.995) #training using
dqn
print("Training Time = {:.2f} min".format((time.time()-timer_start)/60))
```

```
Episode 100    Average Score: 0.97
Episode 200    Average Score: 4.32
Episode 300    Average Score: 7.02
Episode 400    Average Score: 9.83
Episode 500    Average Score: 12.58
Episode 524    Average Score: 13.00
Environment solved in 424 episodes!    Average Score: 13.00
Training Time = 7.32 min
```

```
In [9]: # plot the scores
fig = plt.figure()
ax = fig.add_subplot(111)
plt.plot(np.arange(len(scores)), scores)
plt.ylabel('Score')
plt.xlabel('Episode #')
plt.show()
```



Future Developments

The vanilla DQN used for training the Agent works fine, but there is still room for improvements such as implementing: *a double DQN, a dueling DQN, and/or prioritized experience replay!*