```python
import numpy as np
import gym
import matplotlib.pyplot as plt
import keras
from keras.models import Sequential
from keras.layers import Dense
from collections import deque
import random
import tensorflow as tf

env = gym.make("CartPole-v1")
state = env.reset()


model = Sequential()
model.add(Dense(units = 50, input_dim=4, activation='relu'))
model.add(Dense(units = 50, activation = "relu"))
model.add(Dense(units = 2, activation = "linear"))

opt = tf.keras.optimizers.Adam(learning_rate=0.001)
#opt = tf.keras.optimizers.SGD(learning_rate=0.001)

model.compile(loss='MSE',optimizer=opt)
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 50)                250

 dense_1 (Dense)             (None, 50)                2550

 dense_2 (Dense)             (None, 2)                 102

=================================================================
Total params: 2,902
Trainable params: 2,902
Non-trainable params: 0
_________________________________________________________________
```

```
train_episodes = 200
epsilon = 1
gamma = 0.99
max_steps = 150
state = env.reset()


Loss = []
Rewards = []

for e in range(1, train_episodes+1):
  epsilon = epsilon - (1/train_episodes)
  total_reward = 0
  t = 0

  state = env.reset()
  state = np.reshape(state, [1, 4])

  done = False
  while t < max_steps and done == False:

    Qs = model.predict(state)[0]

    if np.random.rand()<epsilon:
      action = env.action_space.sample()
    else:
      action = np.argmax(Qs)

    next_state, reward, done, _ = env.step(action)
    next_state = np.reshape(next_state, [1, 4])

    total_reward += reward


    if done:
      y = reward
    else:
      y = reward + gamma*np.max(model.predict(next_state)[0])

    Q_target = model.predict(state)
    Q_target[0][action] = y

    h = model.fit(state,Q_target,epochs=1,verbose=0)

    loss = h.history['loss'][0]

    state = next_state
    t+=1

  print(e," R=",total_reward," L=",loss)
  Rewards.append(total_reward)
  Loss.append(loss)
```

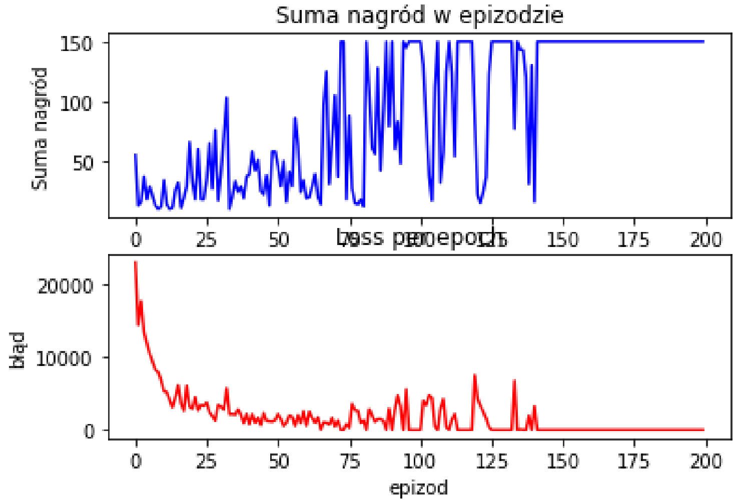
```
143  R= 150.0  L= 0.05605342611670494
144  R= 150.0  L= 0.03562774881720543
145  R= 150.0  L= 0.10835572332143784
146  R= 150.0  L= 0.04605530947446823
147  R= 150.0  L= 0.9776216745376587
148  R= 150.0  L= 0.7532294392585754
149  R= 150.0  L= 0.5138581395149231
150  R= 150.0  L= 0.08586925268173218
151  R= 150.0  L= 0.0009202086366713047
152  R= 150.0  L= 0.1275496482849121
153  R= 150.0  L= 0.06793259084224701
154  R= 150.0  L= 0.06405257433652878
155  R= 150.0  L= 0.00045708639663644135
156  R= 150.0  L= 0.08655671775341034
157  R= 150.0  L= 0.10646744072437286
158  R= 150.0  L= 0.1870243102312088

159  R= 150.0  L= 0.00875283032655716
160  R= 150.0  L= 0.03806300461292267
161  R= 150.0  L= 0.5337464809417725
162  R= 150.0  L= 0.03217417001724243
163  R= 150.0  L= 0.0027559103909879923
164  R= 150.0  L= 0.018773671239614487
165  R= 150.0  L= 0.2345951497554779
166  R= 150.0  L= 0.13159649074077606
167  R= 150.0  L= 0.016582690179347992
168  R= 150.0  L= 0.19022955000400543
169  R= 150.0  L= 0.15670038759708405
```

```
    169  R= 150.0  L= 0.13070030739708105
    170  R= 150.0  L= 0.07229465991258621
    171  R= 150.0  L= 0.02350209653377533
    172  R= 150.0  L= 0.03356293588876724
    173  R= 150.0  L= 0.05501856282353401
    174  R= 150.0  L= 0.02514421008527279
    175  R= 150.0  L= 0.1591486781835556
    176  R= 150.0  L= 0.03967103362083435
    177  R= 150.0  L= 4.55475237686187e-05
    178  R= 150.0  L= 0.007744935806840658
    179  R= 150.0  L= 0.010163173079490662
    180  R= 150.0  L= 0.012025322765111923
    181  R= 150.0  L= 0.010693103075027466
    182  R= 150.0  L= 0.2475358247756958
    183  R= 150.0  L= 0.012818903662264347
    184  R= 150.0  L= 0.032611094415187836
    185  R= 150.0  L= 0.02377747744321823
    186  R= 150.0  L= 0.12229868769645691
    187  R= 150.0  L= 0.02826005034148693
    188  R= 150.0  L= 0.0005776244215667248
    189  R= 150.0  L= 0.03762223571538925
    190  R= 150.0  L= 0.07234399020671844
    191  R= 150.0  L= 0.046266261488199234
    192  R= 150.0  L= 0.016728900372982025
    193  R= 150.0  L= 0.0012360771652311087
    194  R= 150.0  L= 1.6370904631912708e-07
    195  R= 150.0  L= 0.001105438219383359
    196  R= 150.0  L= 0.0013418059097602963
    197  R= 150.0  L= 8.116767276078463e-07
    198  R= 150.0  L= 0.006019055377691984
    199  R= 150.0  L= 0.0017094686627388
    200  R= 150.0  L= 0.002776907756924629
```

```
plt.subplot(211)
plt.ylabel('Suma nagród')
plt.title('Suma nagród w epizodzie')
plt.plot(list(range(train_episodes)),Rewards,"b")

plt.subplot(212)
plt.xlabel('epizod')
plt.ylabel('błąd')
plt.title('Loss per epoch')
plt.plot(list(range(train_episodes)),Loss,"r")
plt.show()
```


Suma nagród w epizodzie
Suma nagród
150
100
50
0
25
50
75
100
125
150
175
200
Loss per epoch
błąd
20000
10000
0
0
25
50
75
100
125
150
175
200
epizod