李昕旸(2017202105)实验二报告

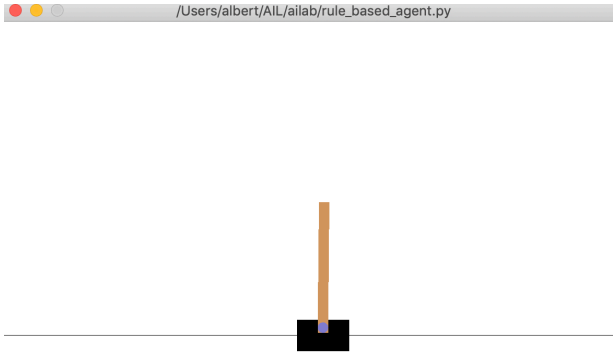
选题： open ai gym

实验环境： Mac OS X Python 3.7

实验概述：

如图所示，本实验装置由一个可以移动的黑色小车和一个细长的木棍组成，我们的主要任务是左右移动小车使得木棍在长时间内不倾倒且小车不撞上左右边缘。本实验在每个状态给我们提供一个长度为4的向量，分别代表：小车位置、小车速度、木棒角度正弦值、木棒角度变化率，上面四个值的负数

表示向左、正数表示向右；而我们需要返回一个决策给系统，返回0代表往左走，返回1代表往右走。



实验过程：

1. 本次实验我希望使用env返回的四维向量拟合一个线性方程，根据上述方程的符号来做出决策，具体如下：

```python
@autojit
def __get_action(self, observation, factors: numpy.ndarray) -> int:
    """
    用线性方程 y = ax + by + cz + dw + e 计算结果
    根据 y 的符号做出决策
    :param factors: 为一个五个浮点数的数组 分别对应上述方程的 a b c d e
    :param observation: 即 env 返回的四维向量 分别对应 小车位置 小车速度 木棍角度正弦值 木棍角度变化率
    :return: 做出的决策 0 或 1
    """

    y = factors[0] * observation[0] + factors[1] * observation[1] + factors[2] * observation[2] \
        + factors[3] * observation[3] + factors[4]
    if y >= 0.0:
        return 1
    else:
        return 0
```

2. 后续就是通过训练不断的优化上述方程的五个参数，我使用爬山算法来训练，每次给拟合的五个参数一点随机偏移，如果偏移后的效果（十次测试步数的均值）更好了就更换为新的参数，具体代码如下图：

```python
def __random_walk(self):
    """
    给现有的参数加上一个随机偏移
    :return: 返回 随机游走后的参数数组
    """
    return self.factors + np.random.normal(0, 0.2, 5)

def __hill_climbing(self):
    cur_factors = self.__random_walk()
    cur_sum_reward = self.__get_avg_reward_by_factors(cur_factors)

    if cur_sum_reward > self.best_reward:
        self.best_reward = cur_sum_reward
        self.factors = cur_factors
        print(self.best_reward)
```

实验结果：
经过训练，三次测试游戏的得分均达到了500的最大值：

```
500.0
500.0
500.0
```

源代码可见附件 src/2017202105.py