AU 332 Artificial Intelligence: Principles and Techniques

By: Yangqian Wu (517021911095)

HW#: 0

September 14, 2019

I. INTRODUCTION

A. Purpose

Homework 0 is about python, which is a popular language for data science. This homework guides us through basics of Python for the Deep Learning course and provides some useful references. Python enjoys a large number of useful add-on packages developed by its active community. For this tutorial, we will use the following packages:

- NumPy: provides multi-dimensional arrays to store and manipulate data;
- SciPy: built on top of NumPy that provides a large number of functions operating on Numpy arrays and useful for scientific applications;
- Pandas: built on top of NumPy that provides additional higher level data manipulation tools that make working with tabular data even more convenient;
- Scikit-learn: provides the off-the-shelf implementation of various machine learning algorithms;
- Matplotlib and Seaborn: provides various plotting functions for data and model visualization;
- Tensorflow: for deep learning.

B. Equipment

There is a minimal amount of equipment to be used in this homework. The few requirements are listed below:

- Python 3.6
- Sublime Text
- Anaconda

C. Procedure

1. Revise Scientific Python in homework
2. Start Sublime Text
3. Accomplish the homework 0 according to guide

II. HOMEWORK DESCRIPTION

The homework is focused on basic Python programming. Its description is as below:

Question 1: Addition

Open `addition.py` and look at the definition of `add`:

```
def add(a, b):
    "Return the sum of a and b"
    "*** YOUR CODE HERE ***"
    return 0
```

Please modify this definition to return the sum of a and b.

(a) Homework 1's description

Question 2: buyLotsOfFruit function

Add a `buyLotsOfFruit(orderList)` function to `buyLotsOfFruit.py` which takes a list of `(fruit,pound)` tuples and returns the cost of your list. If there is some `fruit` in the list which doesn't appear in `fruitPrices` it should print an error message and return `None`. Please do not change the `fruitPrices` variable.

(b) Homework 2's description

Question 3: shopSmart function

Fill in the function `shopSmart(orders, shops)` in `shopSmart.py`, which takes an `orderList` (like the kind passed in to `FruitShop.getPriceOfOrder`) and a list of `FruitShop` and returns the `FruitShop` where your order costs the least amount in total. Don't change the file name or variable names, please. Note that we will provide the `shop.py` implementation as a "support" file, so you don't need to submit yours.

(c) Homework 3's description

FIG. 1: The homework's description.

III. HOMEWORK'S RESULT

This section will consist of the final result of each homework.

A. Homework 1

```
# addition.py
# ------------
# Licensing Information:  You are free to use or extend these projects for
# educational purposes provided that (1) you do not distribute or publish
# solutions, (2) you retain this notice, and (3) you provide clear
# attribution to UC Berkeley, including a link to http://ai.berkeley.edu.
#
# Attribution Information: The Pacman AI projects were developed at UC Berkeley.
# The core projects and autograders were primarily created by John DeNero
# (denero@cs.berkeley.edu) and Dan Klein (klein@cs.berkeley.edu).
# Student side autograding was added by Brad Miller, Nick Hay, and
# Pieter Abbeel (pabbeel@cs.berkeley.edu).


def add(a, b):
    "print the sum of a and b"
    print(a + b)                    #calculate a + b
    return 0
```

B. Homework 2

```
# buyLotsOfFruit.py
# -----------------
# Licensing Information:  You are free to use or extend these projects for
# educational purposes provided that (1) you do not distribute or publish
# solutions, (2) you retain this notice, and (3) you provide clear
# attribution to UC Berkeley, including a link to http://ai.berkeley.edu.
#
# Attribution Information: The Pacman AI projects were developed at UC Berkeley.
# The core projects and autograders were primarily created by John DeNero
# (denero@cs.berkeley.edu) and Dan Klein (klein@cs.berkeley.edu).
# Student side autograding was added by Brad Miller, Nick Hay, and
# Pieter Abbeel (pabbeel@cs.berkeley.edu).


"""
To run this script, type

  python buyLotsOfFruit.py

Once you have correctly implemented the buyLotsOfFruit function,
the script should produce the output:

Cost of [('apples', 2.0), ('pears', 3.0), ('limes', 4.0)] is 12.25
"""

fruitPrices = {'apples':2.00, 'oranges': 1.50, 'pears': 1.75,
               'limes':0.75, 'strawberries':1.00}

def buyLotsOfFruit(orderList):
    """
        orderList: List of (fruit, numPounds) tuples

    Returns cost of order
    """
    totalCost = 0.0                 #to initialize
    for fruit in orderList:         #count the cost
      if fruit[0] in fruitPrices:
        totalCost += fruit[1] * fruitPrices[fruit[0]]
      else:
        return None
    return totalCost                #return the cost


# Main Method
if __name__ == '__main__':
    "This code runs when you invoke the script from the command line"
    orderList = [ ('apples', 2.0), ('pears', 3.0), ('limes', 4.0) ]
    print('Cost of', orderList, 'is', buyLotsOfFruit(orderList))
```

C. Homework 3

```
# shopSmart.py
# ------------
# Licensing Information:  You are free to use or extend these projects for
# educational purposes provided that (1) you do not distribute or publish
# solutions, (2) you retain this notice, and (3) you provide clear
# attribution to UC Berkeley, including a link to http://ai.berkeley.edu.
#
# Attribution Information: The Pacman AI projects were developed at UC Berkeley.
# The core projects and autograders were primarily created by John DeNero
# (denero@cs.berkeley.edu) and Dan Klein (klein@cs.berkeley.edu).
# Student side autograding was added by Brad Miller, Nick Hay, and
# Pieter Abbeel (pabbeel@cs.berkeley.edu).


"""
Here's the intended output of this script, once you fill it in:

Welcome to shop1 fruit shop
Welcome to shop2 fruit shop
For orders:  [('apples', 1.0), ('oranges', 3.0)] best shop is shop1
For orders:  [('apples', 3.0)] best shop is shop2
"""

import shop

def shopSmart(orderList, fruitShops):
    """
    orderList: List of (fruit, numPound) tuples
    fruitShops: List of FruitShops
    """
    cost1 = cost2 = 0.0        #initialize each cost

    for fruit in orders:       #to judge the cost of each shop
        cost1 += fruit[1] * dir1[fruit[0]]
        cost2 += fruit[1] * dir2[fruit[0]]

    if cost1 < cost2:          #to make the decision based on each cost
      return shop1
    else:
      return shop2

if __name__ == '__main__':
  "This code runs when you invoke the script from the command line"
  orders = [('apples',1.0), ('oranges',3.0)]
  dir1 = {'apples': 2.0, 'oranges':1.0}
  shop1 =  shop.FruitShop('shop1',dir1)
  dir2 = {'apples': 1.0, 'oranges': 5.0}
  shop2 = shop.FruitShop('shop2',dir2)
  shops = [shop1, shop2]
  print("For orders ", orders, ", the best shop is", shopSmart(orders, shops).getName())
  orders = [('apples',3.0)]
  print("For orders: ", orders, ", the best shop is", shopSmart(orders, shops).getName())
```

IV. DISCUSSION & CONCLUSION

The goal of this homework was to revise and get familiar with Python programming. Since I had a long time not using Python, I get to realize that I have forgot lots of grammatical principles and techniques.Therefore, I still need some time to revise it and keep learning along the way.

Thanks for teacher and assistant's help.

Fighting!