NOVA IMS Information Management School

Final Project

Introduction to Computational Thinking and Data Science

Algorithm implementation

May 7th - May 31st

NOVA IMS

2023

Contents

1 Project Overview

This project aims to evaluate your knowledge acquired during both the practical and theoretical classes of Introduction to Computational Thinking and Data Science (ICT & DS) while also assessing your researching capabilities and your capability of solving problems in a creative and technical way.

The groups are build by 4 to 5 students and are chosen by yourself. You must Submit your groups on the moodle link (that will be made available to you) from ***May 8th*** to ***May 13th, 23:59h***. Groups who enrol after the enrollment deadline (or deliver the project without enrolling) will be given a 10% penalty on their final grade. If you do not have a group, a specific section on moodle will be available to you where you can sign up individually and have a group created for you by the professors.

The deadline for the delivery of this project is ***May 31st, 23:59h***. Groups who will not respect this deadline will suffer a penalty of two values for each day of delay (up until 5 days, afterwards the project will not be accepted and the students will receive a project evaluation value of 0).

All projects will be defended orally on ***June 2nd*** at a time and place that will be shared with you shortly.

In the project defense, you will be asked questions regarding your implementation of the project as well as questions about the theoretical concepts.

All members of the group will be asked questions and students who will not be able to provide satisfactory answers will be discounted.

Projects will be delivered through Moodle (using the given delivery link).

A specific list of the elements of the project delivery and their corresponding rules are provided in Section 4.

Make sure you read every section carefully!

2 Project Description

Your goal in this project is to test your ability to follow pseudo-code and instructions in order to successfully implement an algorithm. Furthermore, your ability to plan, organize and implement certain choices will also be put to the test.

More specifically, you will be tasked with implementing the proposed algorithm, performing some simple data preparation and running the implemented algorithm on your data in order to solve a problem proposed to you.

You will be given all the information you need with regarding to the problem that you need to solve as well as all the information regarding the algorithm you need to implement and the treatment you should do to the data in order to do it. However, a small number of decisions depend on your choices and what you opt to do so make sure you research your options and justify your decisions.

The specific requisites for the project (and their further explanation) are found below. Make sure you read the rules carefully!

3 Project Rules

Overall you are requested to:

- Implement the algorithm proposed in 3.3.
- Perform all requested data transformations referenced in 3.2.2 while not forgetting to justify any decision taken.
- Write a report where you visually present your results as well as explain your implementations. This report will be in the format of a jupyter notebook.
- Additionally, you are not allowed to use the Pandas library to solve this problem.

3.1 Project Evaluation

Your final grade will be given in the following way:

- **Project Report - 30%**. In the report you must describe the methodology used while presenting and analysing your obtained results. Adding visualizations will really help your storytelling! Deliver the report in the format of Jupyter notebook where you can properly explain your thinking between each cells.
- **Implementation - 70%**. This part of the evaluation corresponds to your ability to correctly implement the algorithm and treat the data, referring to both the quality and efficiency of your implementations.

3.2 The Problem

Imagine that a famous YouTuber contacts you and requests for your help in optimizing his strategy in a famous game that they are streaming. The game in question is day based, which makes it so that there are a limited number of things that can be done and certain things that only occur on specific days. Therefore, gaining a lot of money in the game can highly depend on rightly choosing what main activity to invest each day in. For the sake of this project we will use the famous game Stardew Valley as inspiration. However, keep in mind that any game with a similar day based game logic can be considered. So, the famous Youtuber wants to use the knowledge obtained from the statistics of different days in order to choose what activity they should focus their attention on at the beginning of a specific day. Naturally, the data that you will be given is entirely made up and not based on actual game statistics.

In summary (and for the sake of this project) there are five main activities that a player can focus on: Foraging, Farming, Mining, Animal Care or Fishing.

Figure 1: The five main activities present in our data.

The objective is to decide which one of these five activities the player should focus on, based on the characteristics of the day.

3.2.1 Data characteristics

Overall there are four seasons in the year, each season with 28 days. Different seasons are characterized by having different activities, crops and available fish. Every day the player can check what is the mood of the spirits. Better mood indicates better luck which means better fish and ores that can be mined.

Each case (dictionary that corresponds to a day and is inside the data list) is characterized by having the following information:

- **season, string** : represents the season of the game
- **day, int** : represents the day in the season of the game. Ranges from 1 to 28.
- **spirits_mood, string** : represents the luck level associated with that day. Can be very displeased, displeased, neutral, pleased or very pleased
- **plot_fertilized, bool** represents whether the farming plot currently has fertilizer or not.
- **fish_category, string** : represents the group of fish available during the day. Can be A, A1, B, B1, C, C1, D or D1.
- **mining_level, int** : represents the mining level that the player has currently unlocked. Ranges from 0 to 100
- **desert_unlocked, bool** : represents whether the player unlocked the desert or not. Unlocking the deserts allows more mining possiblity.
- **special_crops, bool** : represents whether the player unlocked has special crops that can be planted or not.
- **forage_category, string** : represents the group of forage items that can be found around the world during the day. Can be A, B, C, or D.
- **is_raining, bool** : represents whether it is raining. Certain fish only appear when its raining. Also, crops do not need to be watered.
- **tomorrow_rain, bool** : represents whether tomorrow will be a rainy day.
- **current_money, int** : represents how much money the player currently has.
- **available_quest, String** : represents the type of activity associated with the current quest that is available for the player. Can be Foraging, Farming, Mining, Animal Care or Fishing.
- **quest_income, int** : represents how much gold the players will earn if they choose to (and are able to) complete the quest.

Additionally, each case has the information regarding **5 more variables** called profit_foraging, profit_farming, profit_mining, profit_animal Care and profit_fishing. These variables refer to the profit that the players would have if they have dedicated to each one of the different activities.

3.2.2 Data Transformations

You are requested to prepare the data before giving it to your algorithm. This preparation is a very small and simple step that is, however, very relevant for the success of your implementation.

You are requested to do the following:

- Remove all of the 5 final variables (profit_foraging, profit_farming, etc...) and create one variable called best_option that refers to which one of the five activities is the most profitable for that day.
- Remove (or treat) the days that have None values.
- Correctly transform the non numeric data as required by the algorithm (presented in 3.3).
- As you can see in 4, you are not allowed to use Pandas library. However, turning this list of dictionaries into a list of numpy arrays (or numpy array of numpy arrays) might be necessary in order to facilitade your implementations.

3.3 The Algorithm

As previously mentioned, the Youtuber who contacted you wants you to use data regarding days and profits in order to correctly assess what is the best activity for them to invest their time in. You decided to brainstorm together and try to come up with an algorithm that would allow you to achieve your goal. At the end, you want to figure out how well your algorithm performs and so you also choose to implement a relevant accuracy measure to assess the quality of your algorithm's suggestions.

In order to do this you chose to randomly divide the data you have into two arrays. One sub-array (called helper) will be used as a base for your algorithm as you try to correctly assess what the most profitable activity is in the other, smaller array (called assessor). At the end, you will calculate the accuracy of your algorithm's suggestions by comparing what suggestions your algorithm gave to the cases in the assessor array in comparison to their actual right answer. You decided to use 65% of the data for the helper array and 35% for the assessor array.
In a brainstorming session you two came up with the following idea:

- For each case in the assessor sub-dataset, Xi:
  - calculate the Euclidean distance between Xi and all the cases in the helper sub-dataset. Note that in order to successfully calculate the euclidean distance between two vectors that have categorical data certain transformations need to occur. Research and implement the strategy you considered best for this transformation.
  - choose the 6 more similar cases
  - classify Xi's best_option as the most common best_option among the 6 most similar cases.

At the end, you will calculate how well your algorithm is able to make suggestions, considering the best main activity your YouTuber friend should focus on. For that purpose, you will use and calculate a success score (SCS) for each one of the main activities (indicating how successful your algorithm is in understanding when each activity is best to spend time on).

$$SCS = 2 \times \frac{S1 \times S2}{S1 + S2}$$

Where,

$$S1 = \frac{RA}{RA + WS} \quad (1) \qquad\qquad S2 = \frac{RA}{RA + MS} \quad (2)$$

and RA corresponds to the Right Answer (i.e., number of times your algorithm correctly suggested this activity as the most profitable activity), WS corresponds to the Wrong Suggestion (i.e., number of times the algorithm suggested the corresponding activity, when it wasn't the most profitable one) and MS corresponds to the Missed Suggestions (i.e., number of times where the algorithm suggested another activity where the corresponding activity would be the most profitable one).

3.3.1 Methodology specification

Running your algorithm once is not enough! You might obtained terrible or great results simply due to the randomness aspects of your data split. Therefore, you should repeat your algorithm in an iterative fashion at least 10 times (dividing the data into the helper and assessor arrays randomly at each time) and storing your quality measure results. At the end, you should calculate the 25th percentile, the median, the 75th percentile, the mean and standard deviation of your SCS for each main activity (over the 10 iterations) and present those values as your final results. You may also try a different number of similar cases (rather than just 6), though that is not mandatory.

4 Delivery

The delivery of this project **must** contain the following:

1. Your code. You must **comment** your code well! You must document your functions using the numpy format docstrings.

2. A report, as indicated in Section 3.1. Do not forget to use proper citations if relevant, analyse your results and justify any decision you made and tell the store

of your methodology, strategies chosen and the final model. Note that your code and the report are the same file!

You must deliver a jupyter notebook as both your code and the report on moodle, using the link that will be provided to you at the time.

Good work! ☺