**OPS Task Report**

**Sharath Podila –** [**spodila@ufl.edu**](mailto:spodila@ufl.edu)**, 37006904**

**Task & Requirements:**

A GitHub project (https://github.com/padillacoreanolab/social\_competiton\_elo\_rating) has been developed to determine Elo ratings (a ranking score) from records of animal behavior. This Jupyter notebook reads data from an Excel sheet, including experimental results and generates Elo scores, CSV files, and graphs. Task is to clone the project and refactor and convert the file “jupyter\_notebooks/tube\_test\_processing.ipynb” such that it meets the following requirements:

1. Library that exposes functions.
2. Library should take input via function or class parameters, not via the "input" built-in function.
3. Expected outputs are the 3 CSV files and the plots for all the cages considered.

**Environment Setup: (OS – Mac)**

1) Installed Visual Studio Code.

2) Have installed necessary Python environment artifacts and plugins.

3) Embedded Jupyter notebook in VsCode so that \*.ipynb can be accessed and run directly.

4) Created a Python 3.9 virtual environment and setup the anaconda framework.

5) Installed all the python packages required for tube\_test\_processing.ipynb to run and execute for the appropriate output.

**Problem Understanding and Approach**

I have been provided with a data pipeline that monitors the competitive behavior of two mice while they are contained within a tube. The mice are classified as either winners or losers based on whether they were the ones who pushed the other mouse out of the tube or were the ones who were pushed. Ipynb is a sequential phase of Python code that collects data, builds data frames, and performs data cleaning to determine the Elo rating for various mice competing in the cages. It then merges all the data frames to produce the final rating. The final data are being written into CSV files, and the graphs that match to the number of cages are being plotted. Approach for the above problem statement and the requirements is that.

1) Apply all the OOPS concepts, such as class, objects, encapsulation, and abstraction.

2) Modularize the entire code into class member functions as per data analysis tasks such as preprocessing, cleaning, analyzing, creating data frames, merging data frames, writing CSV data, and plotting graphs.

3) Improve readability and optimize the code so that naïve users can understand the code.

4) Take the necessary inputs from command line arguments and remove usage of built-in functions!

5) Expose the modularized functions from the Python library.

**Pseudo code/Algorithm**

class DataProcessor:

def \_\_init\_\_(self):

# Initialize class variables here, such as file paths and dictionaries

def data\_preprocessing(self):

# Preprocess the data, including selecting sheets and header rows

def create\_dataframes(self):

# Create dataframes for each sheet with the appropriate header rows

def data\_cleaning(self):

# Clean the data by handling missing values, data types, and inconsistencies

def calculate\_process\_elorating(self):

# Calculate Elo ratings for each process in the dataset

def merge\_process\_eloratings(self):

# Merge Elo ratings across sheets, accounting for overlapping process data

def create\_final\_eloratings(self):

# Create a final dataframe with the Elo ratings for all processes

def make\_plots(self):

# Visualize the data by creating plots, such as bar charts and line graphs

def save\_dataframes(self):

# Save the final processed dataframes to files

# Instantiate the DataProcessor class

processor = DataProcessor()

# Execute the methods in the desired order

processor.data\_preprocessing()

processor.create\_dataframes()

processor.data\_cleaning()

processor.calculate\_process\_elorating()

processor.merge\_process\_eloratings()

processor.create\_final\_eloratings()

processor.make\_plots()

processor.save\_dataframes()

**Member function Explainations**

**data\_preprocessing()** : The main purpose of this function is to preprocess the Excel data by selecting the desired sheets and header rows, ensuring the inputted sheet names and header rows are valid and properly formatted. The main purpose of this function is to preprocess the Excel data by selecting the desired sheets and header rows, ensuring the inputted sheet names and header rows are valid and properly formatted.

**create\_dataframes()** :The main purpose of this function is to read data from the specified Excel sheets, preprocess and standardize the data, and store the resulting dataframes in a suitable format for further processing, such as data cleaning and Elo rating calculation.

**create\_dataframes()** : create\_dataframes reads data from the specified Excel sheets, preprocess and standardize the data, and store the resulting dataframes in a suitable format for further processing.

**data\_cleaning()** : The primary purpose of this function is to preprocess and clean the data by performing the following operations like identifying and setting up the appropriate columns for ties, session dividers, and cage numbers for both winners and losers, renaming certain columns to standardize their names, filling missing values using the forward-fill method, which fills missing values with the value from the previous row, calculating session number differences, mapping winner and loser IDs to their respective cage numbers if an ID-to-cage mapping is provided, extracting numeric cage numbers from the cage number columns using regular expressions, printing the processed data frame for each sheet.

The overall goal of this code is to prepare the dataset for further analysis or processing by cleaning and organizing the data.

**calculate\_process\_elorating()** : The purpose of this function is to compute Elo ratings based on the event outcomes and store them in a new dataframe along with additional relevant columns.

**merge\_process\_eloratings()** : This function's objective is to provide a single, unified dataframe that contains Elo ratings as well as any other information that may be useful for subsequent analysis or display.

**create\_final\_eloratings()** : This function's goal is to provide a summary dataframe that contains the final Elo ratings for each subject. This will make it possible to compare and analyze the findings in a more straightforward manner.

**make\_plots()** : This piece of code is intended to provide visual representations of the Elo ratings as they change over time for each subject. These representations, which can assist users in gaining insights into their performance and how it changes throughout a variety of sessions and cages, are intended to be generated by the code. The plots that are generated are stored as PNG files in the output directory that has been defined, which makes it simple to review and discuss the results.

**save\_dataframes()** : The goal of this piece of code is to perform an analysis of data pertaining to animal behavior, identify the dominance connections that exist between pairs of animals based on the results of their victories and defeats, and then store the findings in a manner that can be formatted for the purpose of conducting additional research.

Functions have been exposed from the python library using the following line of script.

# Use the \_\_all\_\_ attribute to expose the function

\_\_all\_\_ = ['main']

Use the following command to execute the Library.

<Path To EloScoreRating Script> "" "" "" "" "" "" "" "" (Command line arguments for the input inbuilt functions which is then parsed in main function and sent to member functions respectively)

Command: <Path To EloScoreRating Script> "" "" "" "" "" "" "" ""

Example: /Users/sharathpodila/social\_competiton\_elo\_rating/jupyter\_notebooks/myenv/bin/python /Users/sharathpodila/social\_competiton\_elo\_rating/jupyter\_notebooks/EloScoreRating.py "" "" "" "" "" "" "" ""