Movella Coding Assessment

Please create a git repository and use commits to act as a chronological record of your implementation. If you have any questions, please let us know. When finished, please email the GitHub link for review.

Hockey Aggregates
Using the Professional Hockey dataset available on Kaggle, complete the assessment stated below.

Source:
https://www.kaggle.com/open-source-sports/professional-hockey-database?select=Goalies.csv
Data on hockey players, teams, and coaches from 1909 to 2011

That dataset has many .csv files. For this assessment use Goalies.csv. (Goaltending statistics)
Descriptions of the fields that are relevant to this exercise are listed here:

- playerID: Player ID
- year: Year (2005-06 listed as "2005")
- tmID: Team ID
- GP: Games played
- Min: Minutes
- W: Wins
- L: Losses
- T/OL: Ties / overtime losses
- ENG: Empty net goals
- SHO: Shutouts
- GA: Goals against
- SA: Shots against

Assessment Task:

Program and display these aggregates as output

1. tmID: string
2. year: Year
3. Wins_agg: total wins / total players
4. Losses_agg: total losses / total players
5. GP_agg: total games played / total players
6. Mins_over_GA_agg: total minutes played / total goals against
7. GA_over_SA_agg: total goals against / total shots against
8. avg_percentage_wins: calculate the percentage of games won for each player, then take the mean at team level
9. most_goals_stopped: {‘playerID’: playerID, ‘goals_stopped’: goals_stopped}
    - Description: calculate goals stopped per player, then take the player with the max goals stopped and put the details in the dictionary
10. most_efficient_player: {‘playerID’: playerID, ‘efficiency’: goals_stopped / minutes played}
    - Description: calculate the goals stopped per minute of play for each player, then take the player with the max efficiency just calculated and put the details in the dictionary

What we value:

- We look for a well-structured code with sensible test coverage through robust unit test cases
- Descriptive function and variable names are appreciated, as is isolating your business logic from the rest of your code
- Include documentation about your implementation and any assumptions made
- Use frequent commits to act as chronological records for your implementation

Program in Python3 and push the .py files to git repository. Use any numerical/data packages you may think are useful, such as NumPy, SciPy, Scikit-Learn, Pandas, etc.