Basketball Player Performance Analysis

My name is Adaora Onwumelu and this project is part of my Applied Algorithms class with Professor Ghosh. The goal of this project is to analyze basketball player performance using various algorithms such as clustering, classification, and regression.

Overview

This project aims to develop a comprehensive system for analyzing basketball player performance. The system incorporates various algorithms and techniques to predict player performance, categorize players into distinct classes, and cluster players based on their playing profiles. Additionally, the system includes modules for computing various performance metrics and visualizing the results to aid decision-making for coaches, scouts, and team management.The implemented algorithms include:

Features:

- **Regression Algorithm:** Utilizes regression analysis to predict player performance metrics, such as points per game (PTS), based on relevant features.
- **Classification Algorithm**: Categorizes players into distinct classes (e.g., Star Player, Average Player, Bench Player) based on their performance metrics using classification techniques.
- **Clustering Algorithm**: Groups players into clusters based on their performance metrics, such as minutes played (MIN) and points (PTS), to identify different player profiles or playing styles.
- **Performance Analysis Module**: Computes various performance metrics for each player, including PTS, rebounds (REB), assists (AST), steals (STL), blocks (BLK), and shooting percentages.

Implementation

The project is implemented in C programming language. Each algorithm has its own source code file:

- `clustering_algorithm.c`
- `classification_algorithm.c`
- `regression_algorithm.c`
- `player_performance_analysis.c`

Each source code file loads input data from a file provided by the user, performs the respective analysis, and prints the results to the screen.

Usage

To run each algorithm:

1. Compile the source code file using a C compiler.

gcc classification_algorithm.c -o classification_algorithm
gcc clustering_algorithm.c -o clustering_algorithm
gcc regression_algorithm.c -o regression_algorithm
gcc player_performance_analysis.c -o player_performance_analysis

2. Execute the compiled program.

./classification_algorithm
./clustering_algorithm
./regression_algorithm
./player_performance_analysis

3. Follow the prompts to enter the name of the input data file.
Player_data.txt

Data Collection: Gather basketball player data from various sources, including publicly available datasets and proprietary team databases.
Data Preprocessing: Clean and preprocess the data to handle missing values, normalize features, and encode categorical variables.
Algorithm Implementation:

Implement regression models for predicting player performance metrics.
Develop classification algorithms for categorizing players into performance classes.
Design clustering algorithms to group players based on their playing profiles.

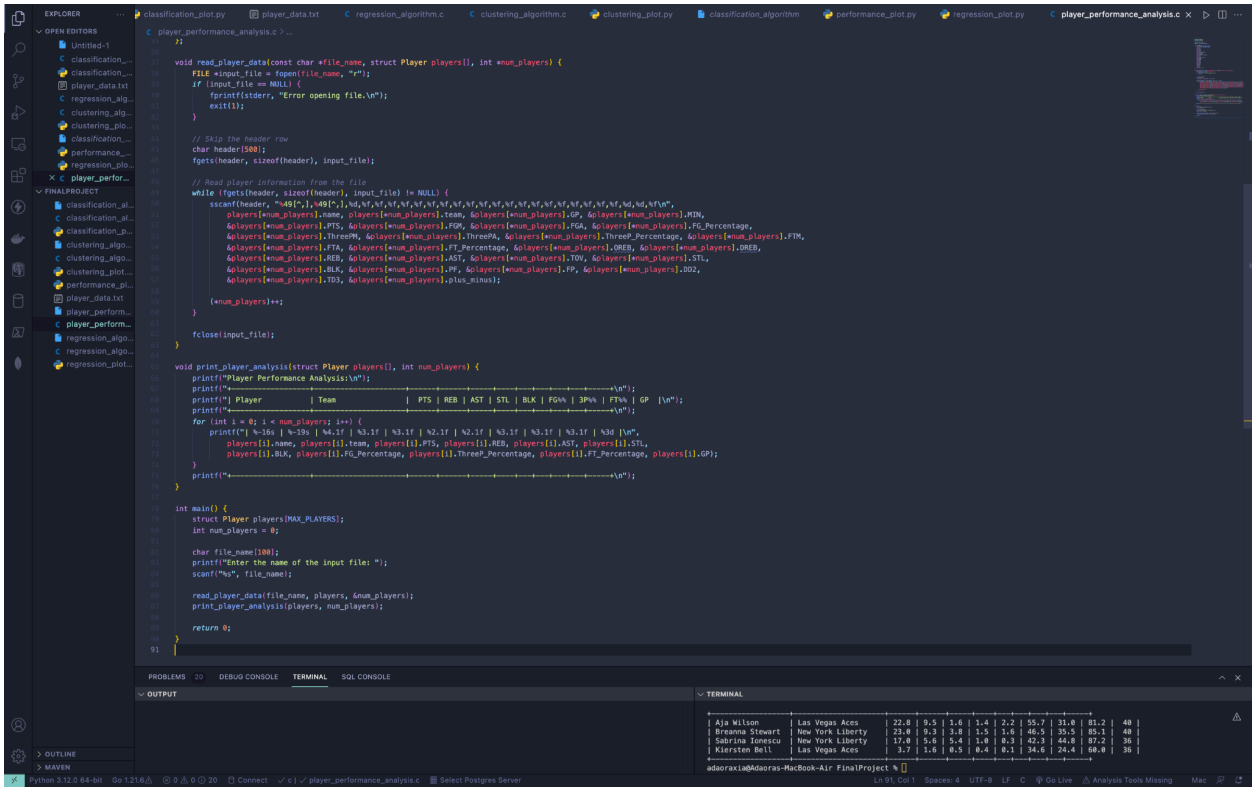
Performance Analysis:

Compute performance metrics for each player using the analysis module.
Visualize the results using plots and charts to facilitate understanding and decision-making.

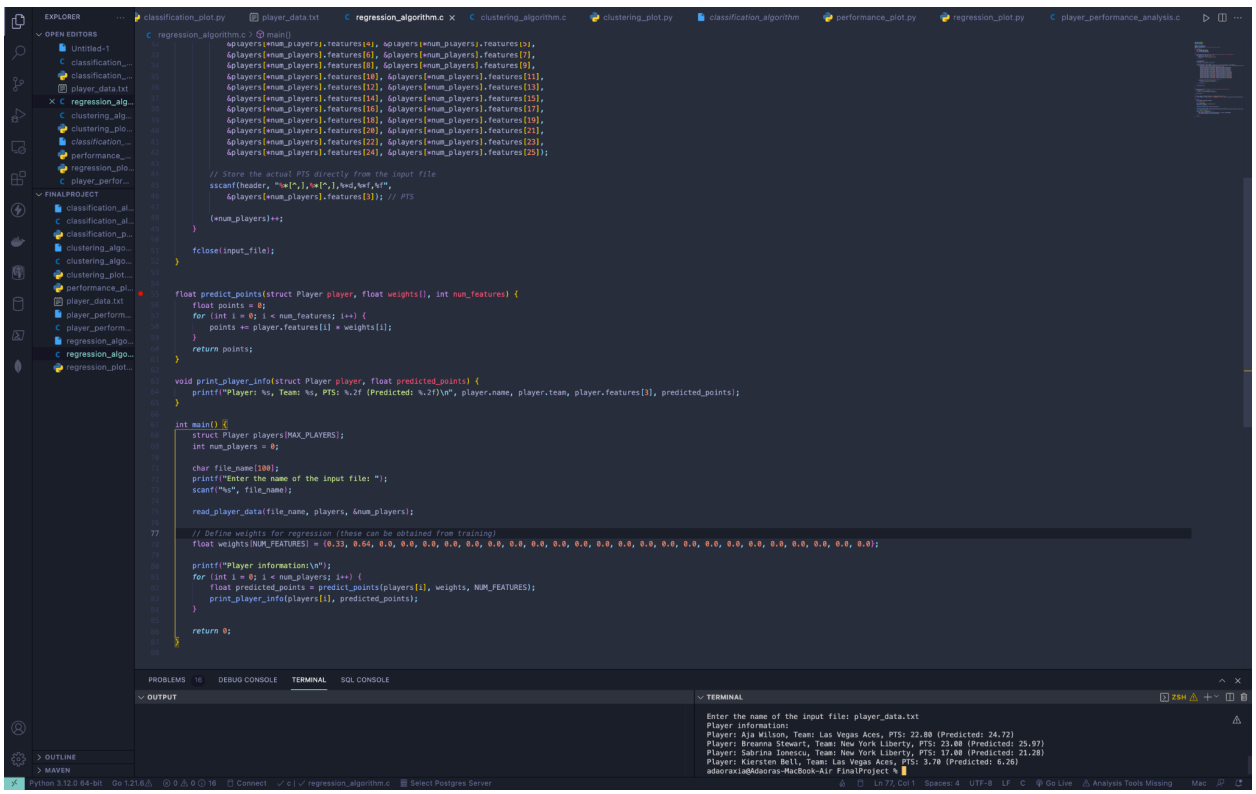
Efficiency Evaluation:

Conduct experiments using both synthetic minimal data and real WNBA data to evaluate the efficiency and scalability of the algorithms.
Measure runtime performance, monitor memory usage, and analyze scalability across varying problem sizes.

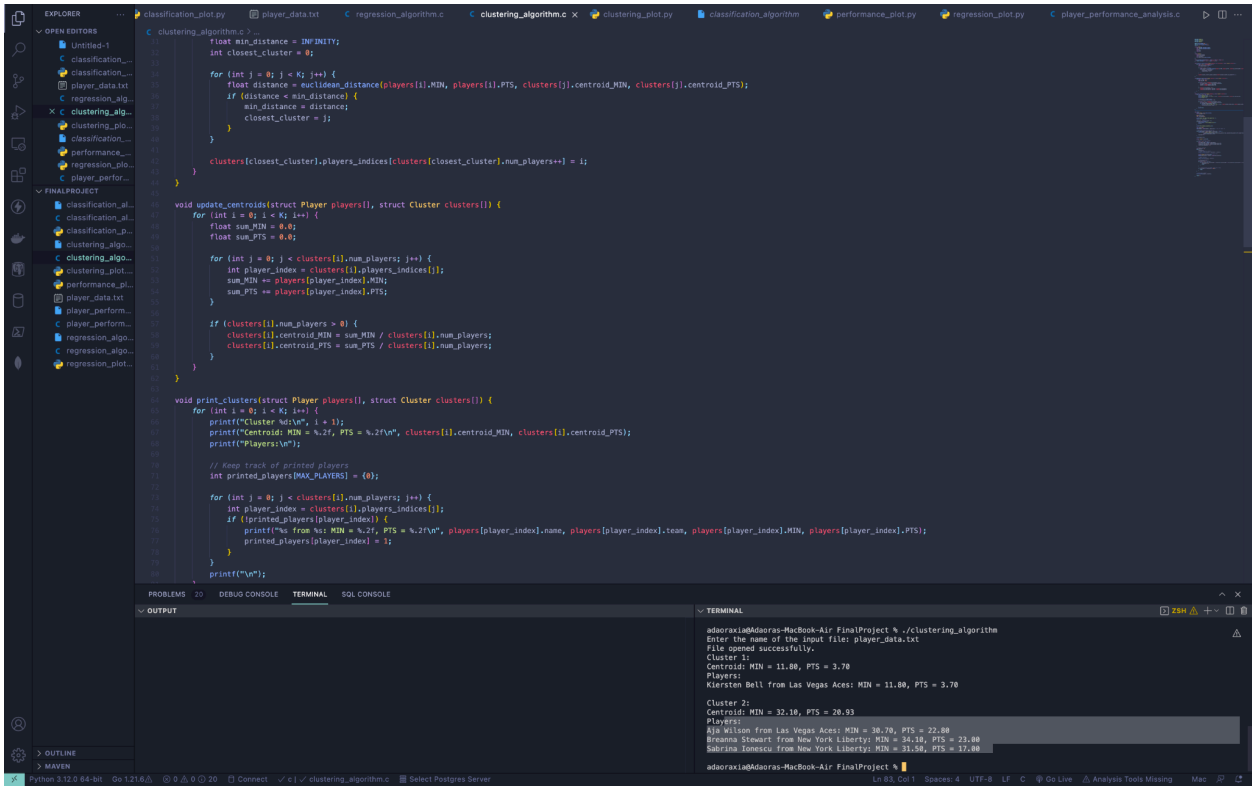
Input Data Screenshots

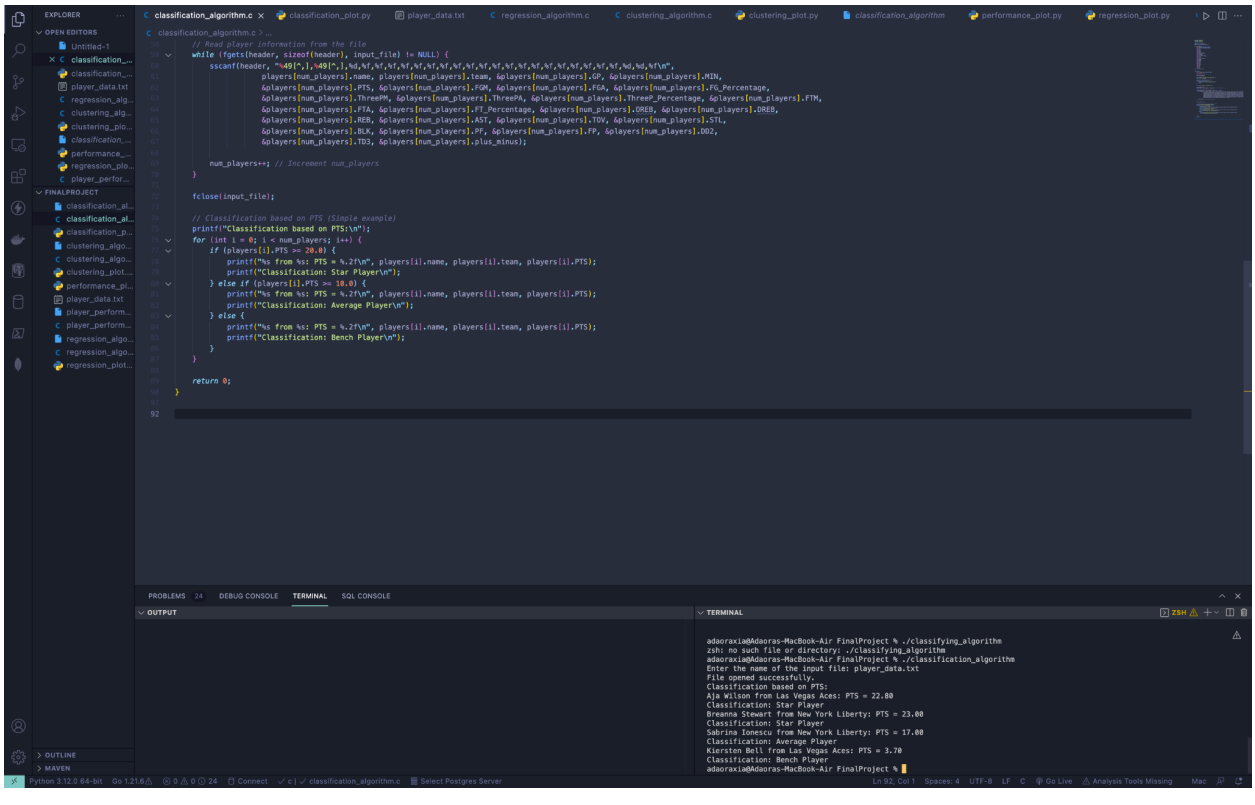


```
| Aja Wilson     | Las Vegas Aces      | 22.8 | 9.5 | 1.6 | 1.4 | 2.2 | 55.7 | 31.0 | 81.2 |  40 |
| Breanna Stewart | New York Liberty   | 23.0 | 9.3 | 3.8 | 1.5 | 1.6 | 46.5 | 35.5 | 85.1 |  40 |
| Sabrina Ionescu | New York Liberty   | 17.0 | 5.6 | 5.4 | 1.0 | 0.3 | 42.3 | 44.0 | 87.2 |  36 |
| Kiersten Bell  | Las Vegas Aces      |  3.7 | 1.6 | 0.5 | 0.4 | 0.1 | 34.6 | 24.4 | 60.0 |  36 |
adaoraxia@Adaoras-MacBook-Air FinalProject %
```



```
Enter the name of the input file: player_data.txt
Player information:
Player: Aja Wilson, Team: Las Vegas Aces, PTS: 22.80 (Predicted: 24.72)
Player: Breanna Stewart, Team: New York Liberty, PTS: 23.00 (Predicted: 25.97)
Player: Sabrina Ionescu, Team: New York Liberty, PTS: 17.00 (Predicted: 21.28)
Player: Kiersten Bell, Team: Las Vegas Aces, PTS: 3.70 (Predicted: 6.26)
adaoraxia@Adaoras-MacBook-Air FinalProject %
```


clustering_algorithm.c
float min_distance = INFINITY;
int closest_cluster = 0;
for (int j = 0; j < K; j++) {
float distance = euclidean_distance(players[i].MIN, players[i].PTS, clusters[j].centroid_MIN, clusters[j].centroid_PTS);
if (distance < min_distance) {
min_distance = distance;
closest_cluster = j;
}
}
clusters[closest_cluster].players_indices[clusters[closest_cluster].num_players++] = i;
}
}
void update_centroids(struct Player players[], struct Cluster clusters[]) {
for (int i = 0; i < K; i++) {
float sum_MIN = 0.0;
float sum_PTS = 0.0;
for (int j = 0; j < clusters[i].num_players; j++) {
int player_index = clusters[i].players_indices[j];
sum_MIN += players[player_index].MIN;
sum_PTS += players[player_index].PTS;
}
if (clusters[i].num_players > 0) {
clusters[i].centroid_MIN = sum_MIN / clusters[i].num_players;
clusters[i].centroid_PTS = sum_PTS / clusters[i].num_players;
}
}
}
void print_clusters(struct Player players[], struct Cluster clusters[]) {
for (int i = 0; i < K; i++) {
printf("Cluster %d:\n", i + 1);
printf("Centroid: MIN = %.2f, PTS = %.2f\n", clusters[i].centroid_MIN, clusters[i].centroid_PTS);
printf("Players:\n");
// Keep track of printed players
int printed_players[MAX_PLAYERS] = {0};
for (int j = 0; j < clusters[i].num_players; j++) {
int player_index = clusters[i].players_indices[j];
if (!printed_players[player_index]) {
printf("%s from %s: MIN = %.2f, PTS = %.2f\n", players[player_index].name, players[player_index].team, players[player_index].MIN, players[player_index].PTS);
printed_players[player_index] = 1;
}
}
printf("\n");
PROBLEMS 20
DEBUG CONSOLE
TERMINAL
SQL CONSOLE
OUTPUT
TERMINAL
adaoraxia@Adaoras-MacBook-Air FinalProject % ./clustering_algorithm
Enter the name of the input file: player_data.txt
File opened successfully.
Cluster 1:
Centroid: MIN = 11.80, PTS = 3.70
Players:
Kiersten Bell from Las Vegas Aces: MIN = 11.80, PTS = 3.70
Cluster 2:
Centroid: MIN = 32.10, PTS = 20.93
Players:
Aja Wilson from Las Vegas Aces: MIN = 30.70, PTS = 22.80
Breanna Stewart from New York Liberty: MIN = 34.10, PTS = 23.80
Sabrina Ionescu from New York Liberty: MIN = 31.50, PTS = 17.00
adaoraxia@Adaoras-MacBook-Air FinalProject %


classification_algorithm.c
// Read player information from the file
while (fgets(header, sizeof(header), input_file) != NULL) {
sscanf(header, "%49[^,],%49[^,],%d,%f,%f,%f,%f,%f,%f,%f,%f,%f,%f,%f,%f,%f,%f,%f,%f,%f,%f,%f,%f,%f,%f,%f,%f,%f,%d,%d,%f\n",
players[num_players].name, players[num_players].team, &players[num_players].GP, &players[num_players].MIN,
&players[num_players].PTS, &players[num_players].FGM, &players[num_players].FGA, &players[num_players].FG_Percentage,
&players[num_players].ThreePM, &players[num_players].ThreePA, &players[num_players].ThreeP_Percentage, &players[num_players].FTM,
&players[num_players].FTA, &players[num_players].FT_Percentage, &players[num_players].OREB, &players[num_players].DREB,
&players[num_players].REB, &players[num_players].AST, &players[num_players].TOV, &players[num_players].STL,
&players[num_players].BLK, &players[num_players].PF, &players[num_players].FP, &players[num_players].DD2,
&players[num_players].TD3, &players[num_players].plus_minus);
num_players++; // Increment num_players
}
fclose(input_file);
// Classification based on PTS (Simple example)
printf("Classification based on PTS:\n");
for (int i = 0; i < num_players; i++) {
if (players[i].PTS >= 20.0) {
printf("%s from %s: PTS = %.2f\n", players[i].name, players[i].team, players[i].PTS);
printf("Classification: Star Player\n");
} else if (players[i].PTS >= 10.0) {
printf("%s from %s: PTS = %.2f\n", players[i].name, players[i].team, players[i].PTS);
printf("Classification: Average Player\n");
} else {
printf("%s from %s: PTS = %.2f\n", players[i].name, players[i].team, players[i].PTS);
printf("Classification: Bench Player\n");
}
}
return 0;
}
PROBLEMS 24
DEBUG CONSOLE
TERMINAL
SQL CONSOLE
OUTPUT
TERMINAL
adaoraxia@Adaoras-MacBook-Air FinalProject % ./classifying_algorithm
zsh: no such file or directory: ./classifying_algorithm
adaoraxia@Adaoras-MacBook-Air FinalProject % ./classification_algorithm
Enter the name of the input file: player_data.txt
File opened successfully.
Classification based on PTS:
Aja Wilson from Las Vegas Aces: PTS = 22.80
Classification: Star Player
Breanna Stewart from New York Liberty: PTS = 23.80
Classification: Star Player
Sabrina Ionescu from New York Liberty: PTS = 17.00
Classification: Average Player
Kiersten Bell from Las Vegas Aces: PTS = 3.70
Classification: Bench Player
adaoraxia@Adaoras-MacBook-Air FinalProject %