Sistema de Ranking Esportivo (Elo Rating)

Projeto Final de Introdução à Ciência da Computação I (ICC I)

Desenvolvido por:

- **Pedro Paulo Silva Santos** (N° USP: 16993206)
- **Vinicius Gonzalez** (N° USP: 16903897)
- **Julia Lopes Lamarchi** (N° USP: 15676110)

Docente: Matheus Machado dos Santos

Sobre o Projeto

Este projeto consiste no desenvolvimento de um sistema de análise esportiva capaz de calcular ratings dinâmicos para diferentes equipes com base em resultados reais de partidas. Apesar do nome “Elo Rating”, o sistema utiliza **um modelo matemático próprio**, implementado por nós, que incorpora:

- o rating atual do time,
- o fator casa (vantagem do mandante),
- e a sequência dos últimos três jogos de cada equipe.

Esse modelo foi construído para refletir melhor o comportamento esportivo observado em futebol e basquete. Ele não utiliza o Elo clássico; em vez disso, adota uma lógica adaptada para este projeto.

O sistema foi desenvolvido em Python e estruturado utilizando o paradigma de **Programação Orientada a Objetos (POO)** para gerenciar o estado e o histórico de desempenho de cada equipe.

A motivação principal surgiu do desejo de integrar os conceitos de programação aprendidos em ICC I com Estatística Aplicada (Probabilidade I), fugindo de escopos triviais para abordar um problema real de *Sports Analytics*. O software permite processar bases de dados históricas, calcular o rating atual de cada time, gerar gráficos de evolução e prever resultados de partidas futuras.

Diferenciais

- **Dual Mode:** O sistema pode ser executado tanto via **Terminal (CLI)** quanto através de uma **Interface Gráfica (GUI)** moderna.
- **Flexibilidade:** Suporte para diferentes esportes (futebol e basquete) com ajustes nos algoritmos de previsão.
- **Visualização de Dados:** Geração automática de gráficos de evolução de desempenho.

Tecnologias e Bibliotecas Utilizadas

Para garantir a robustez e a eficiência na manipulação de dados, utilizamos as seguintes ferramentas:

- **Linguagem:** Python 3
- **Pandas:** Para leitura, manipulação e estruturação eficiente dos arquivos de dados (`.csv`).
- **Matplotlib:** Para a plotagem dos gráficos de evolução do Elo e comparação visual entre times.
- **CustomTkinter:** Para a construção da Interface Gráfica (GUI), oferecendo uma experiência de usuário visualmente superior ao Tkinter padrão.
- **Pillow (PIL):** Para manipulação de imagens dentro da interface.

Instruções de Instalação e Execução

Para que o sistema funcione corretamente, siga os passos abaixo.

1. Pré-requisitos

Certifique-se de ter o **Python** instalado. Em seguida, instale as bibliotecas necessárias executando o comando abaixo no seu terminal:

pip install pandas matplotlib customtkinter pillow

2. Organização dos Arquivos

É fundamental que o arquivo de base de dados (planilha `.csv`) esteja na **mesma pasta** que os arquivos de código (`main.py`, `interface.py`, `simulador.py`, etc.). Isso garante que o programa encontre os dados sem erros de diretório.

Se os arquivos estiverem fora da pasta, o programa não conseguirá localizá-los.

3. Formatação da Base de Dados (`.csv`)

O sistema foi projetado para ler arquivos `.csv`. Para garantir a leitura correta, a sua planilha deve conter **obrigatoriamente** as seguintes colunas (a ordem não importa, mas os nomes devem conter essas palavras-chave):

- **TIME A**: Nome do time mandante.
- **TIME B**: Nome do time visitante.
- **PLACAR A**: Pontuação/Gols do Time A.
- **PLACAR B**: Pontuação/Gols do Time B.

Além disso, o sistema normaliza automaticamente o nome das colunas, ou seja:

- espaços são ignorados,
- caixa alta/baixa não importa,
- contanto que as palavras-chave estejam presentes.

Se sua planilha tiver nomes diferentes (“Home Team”, “Gols Time 1”, etc.), basta **renomear manualmente** as colunas antes da execução.

Colunas extras não atrapalham, elas são simplesmente ignoradas pelo simulador.

---

4. Como Rodar o Projeto

Você pode interagir com o sistema de duas formas:

Opção A: Modo Interface Gráfica

Para uma experiência visual completa:

1. Abra o terminal na pasta do projeto.
2. Execute o comando:
   python interface.py

3. Uma janela se abrirá. Clique em início, escolha o esporte desejado e utilize o menu lateral para carregar seu arquivo `.csv`, visualizar o ranking, analisar estatísticas individuais de times e gerar gráficos.

Opção B: Modo Terminal (Texto)

Para uma execução direta e rápida, ideal para testes ou servidores:

1. Abra o terminal na pasta do projeto.
2. Execute o comando:
   python main.py

3. Siga as instruções interativas no console:
    - Escolha o esporte (Futebol ou Basquete).
    - Selecione se deseja processar um arquivo de jogos (Gerar Elo) ou simular partidas.
    - O programa exibirá os resultados e prints diretamente na tela.

Funcionalidades do Sistema

1. **Cálculo Dinâmico de Elo:** Processa partida por partida, atualizando o rating dos times com base no resultado e na força do oponente.
2. **Histórico e Estatísticas:**
    - Armazena o maior e menor Elo atingido por cada time.
    - Contabiliza vitórias, derrotas e empates.
    - Identifica a sequência atual (Vitória/Derrota).
3. **Visualização Gráfica:** Plota a "Linha do Tempo" do rating de um time, permitindo visualizar ascensões e quedas de desempenho.
4. **Simulador de Partidas:** Com base nos ratings atuais, o sistema calcula a probabilidade matemática de vitória para um confronto hipotético entre dois times.
5. **Datasets de Exemplo:** Para facilitar a validação das funcionalidades do sistema, incluímos na pasta do projeto arquivos .csv de exemplo que já estão formatados corretamente: “jogos_brasileirao”, “jogos_basquete_olimpiadas”, “jogos_premier_league” e “jogos_nbb”. Esses arquivos foram obtidos à partir de Datasets públicos e foram editados para obedecer as regras de formatação das colunas.

Conceitos de Computação Aplicados

Este projeto foi construído para atender rigorosamente aos critérios de avaliação da disciplina de ICC I, aplicando os seguintes pilares:

- **Programação Orientada a Objetos (POO):**
    - Utilizamos a classe `Equipe` (`equipe.py`) para encapsular os atributos (nome, rating, histórico) e comportamentos de cada time.
    - A classe `SimuladorElo` (`simulador.py`) gerencia a lógica de negócio, manipulando o dicionário de objetos `Equipe`.
    - A classe `InterfaceApp` (`interface.py`) herda de `ctk.CTk`, demonstrando uso de herança para criar a GUI.
- **Estruturas de Dados:**
    - Uso extensivo de **Dicionários** (`self.times = {}`) para acesso rápido aos objetos das equipes usando o nome como chave.

    - **Listas** para armazenar o histórico de ratings e permitir a plotagem de gráficos.
- **Manipulação de Arquivos e Tratamento de Exceções:**
    - Leitura de arquivos `.csv` com tratamento de erros (`try/except`) para arquivos inexistentes ou colunas mal formatadas, garantindo a robustez do sistema (item 7 e 8 dos critérios de avaliação).
- **Modularização:** O código está dividido em múltiplos arquivos (`main.py`, `equipe.py`, `simulador.py`, `interface.py`), facilitando a manutenção e a legibilidade, conforme as boas práticas exigidas.

Conclusão e Trabalhos Futuros

Embora o sistema atenda plenamente aos requisitos de complexidade esperados para a disciplina de Introdução à Ciência da Computação I, reconhecemos que o desenvolvimento de software é um processo contínuo. Identificamos oportunidades de evolução técnica que, devido às restrições temporais do semestre e ao foco pedagógico na consolidação de estruturas fundamentais (como POO e manipulação de arquivos locais), foram mapeadas para implementações futuras:

1. **Automação de Coleta (Web Scraping):** Para eliminar a dependência de arquivos `.csv` manuais, a evolução natural do projeto seria a implementação de um módulo de *Web Scraping*. Isso permitiria ao sistema buscar resultados em tempo real diretamente de portais esportivos, automatizando a atualização da base de dados.
2. **Aprofundamento Estatístico:** Atualmente, utilizamos um modelo Elo customizado. Com mais tempo de desenvolvimento, poderíamos incorporar métricas de dispersão, como **desvio padrão** e **média móvel**, para refinar a precisão das previsões e reduzir a volatilidade do ranking em início de temporada, por exemplo.
3. **Visualização Avançada:** Também seria possível expandir a biblioteca gráfica para incluir histogramas de distribuição de força e gráficos de dispersão (*scatter plots*), permitindo correlacionar visualmente o saldo de gols com o rating acumulado.

Projeto desenvolvido para a disciplina SSC0801 - Introdução à Ciência da Computação I - ICMC/USP.