

Kraken

**Arthur Henrique Moreira Santos, João Lucas de Vilas Bôas Faria, João Vitor Simão e
Leonardo Souza Bahia**

Universidade Federal de São João Del Rei (UFSJ) – São João Del Rei, MG - Brasil

Introdução

Este projeto implementa um sistema de geração de encontros para RPGs de mesa, utilizando técnicas de Inteligência Artificial e Ciência de Dados para criar encontros temáticos e balanceados.

A ideia central é ler uma base de dados de monstros, limpar e transformar os dados, aplicar clusterização com K-Means para encontrar grupos de monstros semelhantes, e então gerar automaticamente encontros equilibrados conforme o nível e o tamanho do grupo de jogadores, além disso também é levado em consideração a escolha do usuário em alguns aspectos.

Objetivos

1. Criar um sistema que gera encontros automáticos de RPG coerentes com:
 - a. O nível e tamanho grupo;
 - b. O ambiente (floresta, deserto, caverna, etc.);
 - c. O tamanho e tipo dos monstros;
 - d. A dificuldade desejada(fácil, médio, difícil, impossível).
2. Aplicar técnicas de pré-processamento de dados e aprendizado não supervisionado (K-Means) para organizar e classificar os monstros em grupos (clusters).

Metodologia

O sistema foi dividido em etapas, desde a leitura dos dados até a geração final dos encontros.

Etapla 0 - Leitura e verificação do CSV

1. Lê o arquivo “input.csv” contendo os monstros e suas características.
2. Tenta automaticamente diferentes encodings ("utf-8", "utf-8-sig", "latin-1") para evitar erros de leitura.
3. Detecta automaticamente as colunas de interesse:
 - a. Name
 - b. Type
 - c. Environment
 - d. CR (Challenge Rating)
 - e. Size

Etapla 1 - Seleção das colunas essenciais

1. Cria um subconjunto do DataFrame com apenas as colunas detectadas.
2. Serve como base limpa para as próximas etapas.

Etapas 2 - Limpeza e transformação dos dados

1. Converte valores de CR para números (“¼” para “0.25”).
2. Normaliza textos (remove parênteses, espaços extras e padroniza capitalização).
3. Uniformiza ambientes (“forest, cave” para “Forest, Cave”).
4. Remove duplicatas.
5. Funções principais:
 - a. *parse_cr()* : converte CR em float
 - b. *clean_text_basic()* : limpa e formata textos
 - c. *normalize_env()* : padroniza ambientes

Etapas 3 - Transformação para Machine Learning

1. Adiciona uma coluna “MonsterID” única.
2. Cria variáveis one-hot e multi-one-hot para converter textos em números:
 - a. Type e Size sendo one-hot encoding.
 - b. Environment sendo multi-one-hot (um monstro pode ter vários ambientes de ocupação)

Os resultados são salvos como *monsters_clean.csv* (dados flexíveis) e *monsters_train.csv* (dados numéricos para treino).

Etapas 4 - Clusterização (IA com K-Means)

Primeiramente, a escolha do algoritmo K-Means deve-se à sua eficiência na identificação de padrões e agrupamentos em grandes volumes de dados numéricos, característica que se adequa bem aos objetivos deste trabalho. Por ser um método de aprendizado não supervisionado, o K-Means permite descobrir automaticamente comunidades e perfis de comportamento semelhantes dentro dos dados sem a necessidade de rótulos prévios, fornecendo uma visão mais estruturada e interpretável dos padrões de interação observados na rede social analisada.

Em relação ao código:

1. O sistema utiliza K-Means para agrupar os monstros em 4 clusters temáticos baseados em suas características (Type, Size, Environment).
2. Cada cluster representa um “grupo temático” de monstros semelhantes.

Os resultados são unidos de volta ao DataFrame, marcando o cluster de cada monstro, de forma que permite gerar encontros coerentes tematicamente, sem misturar monstros aleatórios.

Etapas 5 - Cálculo de dificuldade e geração de encontros

1. A parte principal do sistema, que combina heurísticas e IA, é composta por quatro funções:
2. *calcular_valor_grupo(niveis_jogadores)*
 - a. Recebe uma lista com os níveis dos jogadores;
 - b. Calcula um “valor de grupo” com base na soma e média dos níveis.

- c. Esse valor serve de base para balancear o encontro
- 3. *avaliar_encontro(monstros_df, valor_grupo)*
 - a. Avalia a dificuldade do encontro comparando o somatório dos CRs dos monstros com o valor do grupo
 - b. Classifica o encontro como:
 - i. Fácil
 - ii. Médio
 - iii. Difícil
 - iv. Impossível
- 4. *escolher_por_cr_alvo(df_filtrado, cr_alvo, quantidade_oponentes)*
 - a. Seleciona monstros com CR próximo ao alvo.
 - b. Usa uma busca por distância absoluta e sorteia os mais próximos.
- 5. *gerar_encontro()*
 - a. Aplica filtros de tipo, ambiente e tamanho.
 - b. Escolhe monstros de clusters coerentes.
 - c. Ajusta a dificuldade com base no nível do grupo e na quantidade de oponentes.

Resultados Esperados

O sistema desenvolvido permite ao usuário configurar parâmetros de um confronto no universo de Dungeons & Dragons, como dificuldade, ambiente, tipo e tamanho dos inimigos, número de oponentes e composição da equipe.

A partir desses parâmetros, o programa realiza o processamento da base de dados de monstros (`monsters_clean.csv`) e gera um conjunto de confrontos possíveis, exibindo:

- 1. Uma tabela com os monstros selecionados e seus atributos;
- 2. Um resumo estatístico do grupo de jogadores (níveis, valor total e nível médio);
- 3. Uma avaliação automática da dificuldade do confronto, indicando se o encontro está dentro do nível planejado.

Os resultados são exibidos diretamente no notebook, com formatação HTML para melhor legibilidade.

Resultados esperados:

- 1. Interface funcional e responsiva, com atualização dinâmica dos níveis da equipe.
- 2. Sugestões de confrontos coerentes com os parâmetros definidos.
- 3. Tempo de resposta curto para gerar cada encontro.
- 4. Dados exibidos de forma clara e organizada.

Conclusão

O projeto atendeu aos objetivos propostos de construir uma ferramenta interativa capaz de auxiliar mestres de RPG na criação de confrontos balanceados.

Através do uso de Python e bibliotecas de interface (`ipywidgets`), foi possível desenvolver uma aplicação totalmente funcional dentro do Jupyter Notebook, sem depender de ambientes externos.

O sistema combina tratamento de dados, análise de parâmetros e interface gráfica, permitindo uma experiência prática e intuitiva. Como possíveis melhorias futuras, destacam-se:

1. Adicionar um sistema de salvamento de confrontos gerados;
2. Permitir filtros adicionais (por CR médio ou por número máximo de monstros);
3. Exportar os resultados para PDF ou CSV;
4. Incorporar uma camada de aprendizado para sugerir confrontos com base em partidas anteriores.

Tecnologias e bibliotecas utilizadas

Python 3.x
Jupyter Notebook
Pandas
Ipywidgets
IPython.display
HTML
Numpy
CSV Dataset

Referências

Manual dos monstros, Dungeons & Dragons.