|  |
| --- |
| Relatório – Intro CD  DATA: 12/12/2022 |
| **PROJETO:** SPEEDRUN  Equipe: Ciro Gomes, Daniel Santos, Epitácio Neto  **Objetivo:** Usamos os dados da API do site “speedrun.com”, para fazer a análise de dados e retirar algumas informações sobre os jogos.  **Tecnologias usadas:** Python, Colab  **Libs:** json, requests, pandas, seaborn, matplotlib.pyplot, numpy, pandas.io.json,  Missingno, time.  Antes de começar a análise do dataset, importamos algumas bibliotecas que vamos usar no decorrer da analise, para buscar com mais eficiencia oque estamos procurando nesse dataset.    Primeiramente, fazemos um request para a API, de forma que consigamos receber 200 jogos da base de dados do Speedrun.com    Após isto, vamos obter alguns dados de plataformas e especificidades que estão codificadas no JSON que foi retornado dos jogos.    Depois fizemos algumas normalizações para facilitar o momento em que os IDs de plataforma serão mapeados para os nomes da plataforma. |
| Como podemos ver, temos muitas colunas que não possuem utilidade ou não possuem tratamento de dados claro, portanto, vamos remover a maioria das colunas que se caracterizam como "sem tratamento de dados claro", usando um threshold de 10 valores nulos. |

# Resultado após fazer oo drop

# 

# A partir daqui, retiraremos as colunas que estavam com o NaN disfarçado, bem como as colunas que não representam valor na análise de dados.

# 

# Resultado

# 

# Agora aplicando a normalização de nomes dos jogos ao invés de IDs.

# 

# Trocando os ids discretos por nomes das plataformas

# 

# Resultado da troca dos ids para o nome da plataforma

# Extração das Runs

# A partir de aqui, dentro da coluna 'links', estão armazenados as requests que correspondem às runs, de cada jogo. Portanto, vamos extrair esses links e armazená-los em uma lista.

# 

# Dentro da documentação da api, temos a seguinte situação:

# O limite de requests da api são de 100

# Após extrair as 100 requests, é preciso de 1 minuto para as próximas requests

# Outra coisa que é interessante dizer é sobre o offset que, basicamente, desloca onde iremos iniciar a request, portanto, a lista 'offsets' foi feita de forma com que possamos passar por todas as requests presentes na lista 'runs'.

# Colocamos estas requests em uma lista, assim como fizemos para os jogos e, para as requests não falharem, pela regra da api, colocamos um delay de 65s (esses 5s de segurança para garantir sucesso nas requests, encontrado após os testes).

# 

# Vemos, pelo gráfico, que este dataframe também possui vários valores NaN, alguns deles que não conseguimos tratar, portanto, iremos retirá-los.

# 

# Resultado final

# Análise de Centralidade

# Como podemos ver, os dados numéricos, por enquanto, estão nos mostrando a mesma informação, pelo que o describe está nos mostrando. Infelizmente, não estão armazenadas, para esse conjunto de dados, o tempo com as telas de loading removidas, que podiam servir para uma análise mais profunda desses dados. A mediana e a média estão com valores bem próximos, que é um sinal da base estar bem balanceada, nesse quesito, implicando que os tempos das runs feitas pelos jogadores são bastantes próximos, indicando que o cenário, para este jogo, é relativamente competitivo.

# 

# Análise por Gráficos de Dispersão

# Visualmente falando, conseguimos observar que os tempos primary e realtime são praticamente iguais. Como foi confirmado na análise da centralidade, conseguimos atestar, novamente, que os dados estão bem distribuídos, com exceção de uma ocorrência acima da média, de valor maior que 1000 segundos.

# 

# Análise do Boxplot

# Mais uma vez, o boxplot condiz com as análises anteriores, no entanto, vemos alguns outliers menos "gritantes" que os presentes em 1000 segundos para frente, que não conseguimos enxergar, anteriormente, devido à quantidade de bins escolhidas para os gráficos de dispersão.

# 

# Análise de gráficos de linha

# 

# 

# Análise de dados das runs em relação

# aos anos.

# Obs: este foi um teste de

# Classificação por data.

# Carregando o arquivo do dataset gerado no processo de consulta e tratamento dos dados recebidos da api que se referem as runs.

# 

# Tratar valores nulos(nan, none) e remover a coluna com índices únicos em referência aos jogos das runs.

# 

# Removendo colunas não essenciais e runs rejeitadas e não verificadas.

# 

# Análise de centralidade

# 

# Dispersão por categoria em alguns jogos

# 

# Alguns gráficos de dispersão

# 

# Boxplots referentes a alguns jogos da série '.hack'

# 

# 