NDFDS2 – Projeto de Data Wrangling

Autor: Márcio Souza de Oliveira

E-mail: emaildomso@gmail.com

Relatório de Wrangling

Conforme solicitado no projeto, os dados para análise deveriam ser coletados de 3 fontes:

- **twitter-archive-enhanced** – Dados de tweets do WeHateDogs já disponibilizado com o projeto (acesso via URL);
- **image-predictions.tsv** – Dados de previsões de raças de diversos tweets do WeHateDogs com base nas imagens dos mesmo usando uma rede neural. Esse arquivo precisava ser obtido via programação em Python através da URL fornecida
- **tweet-json.txt** – Arquivo de dados complementares ao primeiro que precisava ser gerado por este projeto com base nos id's do primeiro arquivo usando a API do Twitter.

Coleta

Para os 2 primeiros arquivos, utilizei o mesmo método em Python para fazer o download com o auxílio da biblioteca **"requests"**

Para o terceiro arquivo, foi utilizada a biblioteca **"tweepy"** para acessar a API do Twitter. Os dados de autenticação na API são lidos de um arquivo TXT externo.

Inicialmente a estratégia seria percorrer cada id encontrado no arquivo **"twitter-archive-enhanced.csv"**, importando os dados para um DataFrame da biblioteca Pandas, e obter o JSON retornado pela API para cada id gravando um por linha no arquivo **"tweet-json.txt"**

Devido à dificuldade inicial de coletar os dados (usando conexão 3G) e às limitações de tráfego impostas pela API, decidi mudar um pouco a estratégia adiantando uma parte da limpeza desses dados. Como percebi na **"motivação do projeto"** que deveríamos trabalhar apenas com tweets originais (e não retweets), optei por já filtrar o arquivo **"twitter-archive-enhanced.csv"** para eliminar os retweets (considerei como retweet os registros em que a coluna **"retweeted_status_id"** não eram nulos). Com isso houve uma redução considerável de id's para serem processados na API, mas mesmo assim levou um tempo razoável (entre 30 a 40 minutos) para concluir.

Com isso, conseguir gerar um arquivo final **"tweet-json.txt"** com os JSON's de cada id de um tweet original.

Análise

Fiz a análise separadamente em cada um dos 3 arquivos coletados:

- **twitter-archive-enhanced**

Numa análise visual inicial detectei os seguintes problemas:

  - A coluna **"source"** continha URL's envoltas em tags HTML de link (<A>);
  - A coluna **"nome"** continha grande parte dos valores como "None" e os que começavam com letra minúscula não aparentavam ser nomes válidos ("a", "actually", "all", "na", "by", etc);
  - A coluna **"rating_denominator"** também apresentava valores suspeitos. Fazendo uma ordenação dos valores, percebi valores muito baixos como zero, 2 e 7 e muitos altos como 50, 110, 170, etc. A grande maioria era 10. Então fiz uma filtragem por valores diferentes de 10 e comparei o texto do tweet (coluna **"text"**) desses id's com as classificações registradas no csv e percebi que muitos estavam erradas, pois a programação que extraiu essa informação falhou por conta de alguns textos apresentarem uma data que deve ter confundido o algoritmo de extração;
  - Nas colunas de "doggo", "floofer", etc, percebi que vários id's possuíam mais de um valor preenchido. Ao filtrar por esses registros, também constatei ter sido falha no mesmo algoritmo de extração desses valores.

Numa análise mais programática, foi possível avaliar que algumas colunas não estavam com um datatype correto.

- **image-predictions.tsv**

Numa análise visual inicial detectei os seguintes problemas:

  - Nem todos os id's receberam uma raça de cachorro válida da rede neural (ambas as colunas **"p1_dog"**, **"p2_dog"** e **"p3_dog"** com valor FALSE) ou apenas algumas eram válidas (nem sempre a primeira (**"p1"**);
  - Algumas raças com *underscore* no nome;
  - Menos registros no total em comparação com o arquivo **"twitter-archive-enhanced.csv".**

- **Tweet-json.txt**

Foram considerados nos diversos JSON's apenas as contagens de favoritos e retweets. Não havia problemas com esses dados.

Limpeza

Optei por gerar no final do trabalho um DataFrame unificado para simplificar a análise, visto que não havia muitas observações na tabela e diversas colunas foram descartadas após a limpeza. Como a motivação do projeto também indicava que a análise deveria ser feita em tweets com imagens, realizei "inner join" com a tabela de processamento de imagens, com isso os id's de **"twitter-archive-enhanced.csv"** que não possuíam correspondência com **"image-predictions.tsv"** foram descartados.

Grande parte dos problemas de qualidade levantados (Ex: classificações inválidas, underscores, tags HTML, etc.) obtiveram uma correção simples. Outros foram necessários recorrer a expressão regulares.

Os maiores desafios vieram com a arrumação dos dados:

- Criar colunas finais únicas (**"breed"** e **"breed_conf"**) com a raça (e respectiva confiança) obtida da rede neural. A estratégia foi considerar preferencialmente dados das colunas **"p1_*"** que possuíam maior confiança (desde que válidas), caso contrário considerar as colunas **"p2_*"** e, por fim, as colunas **"p3_*"**. Se nenhuma fosse válida, desconsiderar o registro.
- Pivotear colunas **"doggo"**, **"floofer"**, **"pupper"** e **"puppo"** numa única coluna de estágio do cachorro (**"stage"**). Caso duas ou mais colunas possuíssem valores, ambas seriam descartadas por ser impossível determinar o correto.