Esforços necessários ao fazer *Data Wrangling* para o projeto

Com o intuito de efetuar o processo completo de Data Wrangling para o projeto “Preparar e analusar Dados” foi necessário fazer uso de várias bibliotecas não-triviais do Python (json, requests, tweepy) e usar programação iterativa fortemente. Inicialmente foram lidos três DataFrames que foram importados para o Jupyter Notebook de diferentes maneiras; o mais interessante foi baixá-los programaticamente através da API tweepy, em que foi necessário criar uma conta de desenvolvedor no Twitter e usar suas próprias chaves de acesso.

Ao analisar os três DataFrames foram encontrados vários erros neles, tais como erros de formatação (data em um formato errado) que foram consertados desmembrando as componentes da data em dia, mês e ano, e colunas extras se necessário. Também foi observado que algumas colunas estavam como strings enquanto deveriam ser inteiros ou pontos flutuantes, o que é simples de ser consertado utilizando a função *astype* do *Pandas*. Mesmo no caso em que a coluna deveria ser do tipo string, havia um caso que ela vinha com um problema: aspas extras; isso foi consertado programaticamente iterando sobre a coluna, depois adicionando-a ao DataFrame e removendo a antiga pelo método *drop*. Também houve a necessidade de renomear as colunas em alguns casos, para que assim haja compatibilidade entre DataFrames, usando então a função *rename* do *Pandas*. Ao longo de todo o processo de programação, para atender o caso em que os dados estão (parcialmente) corrompidos ou indisponíveis foi muito útil utilizar blocos *try-except*.

Com o intuito de criar os masters DataFrames (os DataFrames finais que estão limpos e condensados) foi necessário usar programação programática e usar função como *merge* do Pandas para unir dataframes tendo como base a coluna *tweet-id*. Foi optado por unir o primeiro e segundo DataFrames, criando um novo, e também unir o segundo e terceiro DataFrames, criando um novo.
Temos, então, dois DataFrames Masters. A partir dos dados limpos, pode-se partir para a fase de visualização e interpretação dos dados.

Aqui abaixo estão, sumarizados, os esforços para fazer limpeza nos DataFrames:

1) csv-file:

Problemas de Qualidade

1) timestamp do jeito que está escrito é um problema.
vamos separa-la em month, day e year!

2) ‘in_reply_to_status_id’ e ‘in_reply_to_user_id’ não significam nada,
podemos dar um drop nelas.

3) 'retweeted_status_user_id' e 'retweeted_status_timestamp' não significam nada,
podemos dar um drop nelas.

Problemas de Arrumação

1) a maneira que foi mostrada ‘rating_numerator’ e ‘rating_denominator’ para fazer concordância com DF posteriores, criaremos uma columa denominada score_rate = rating_numerator/rating_denominator

2) TSV-file:

Quality Issues
Em suma: nesse DF achamos três erros de qualidade que são:

1) na coluna ‘p3_dog’ os valores True e False estão como True',True" e False';

2) na coluna tweet_id os valores devem ser inteiros, não strings!

3) tem raça classificada (coluna 'p1') como "website, limousine, fountain, revolver,military_uniform,seatbelt, etc.

3) Arquivo que deve ser baixado pela API tweepy

Podemos ver os seguintes problemas no DF acima:

Qualidade (Quality)

1) 'full_text' que deve ser desmembrado em ‘gender’, ‘name’ e em ‘rate_score’ (somente nessa parte são três problemas de qualidade);

Estruturais (Tidyness)

1) novamente, não há sentido na ‘timestamp’ que foi apresentada, ele teve que ser reestrurada convenientemente!