Coleta – Gather

Utilizei **pandas** para ler o arquivo csv disponibilizado no site na Udacity. Para ler o arquivo tsv disponibilizado via link pela Udacity, utilizei as bibliotecas **request** e **os.** Como não queria salvar o arquivo localmente, fiz o **request** do link para uma variável e utilizei a biblioteca **os** para decodificar o arquivo. Para a API com Twitter utilizei o **tweepy** e a biblioteca **json**. Através da documentação, à página disponibilizada pela udacity e alguns links do stackoverflow baixei json do twitter. Como não sabia como armezenar os arquivos json, pesquisei e através desse link, https://stackabuse.com/reading-and-writing-json-to-a-file-in-python/, me orientei para escrever no arquivos **txt**.

Avaliação – Assess

Para avaliação, que é a parte mais fácil, utilizei os métodos **head(), info() e value_counts()** do pandas. Precisei também para uma melhor avaliação utilizar a configuração: **pd.set_option('display.max_colwidth', -1),** para poder olhar o texto sem nenhum impedimento de tamanho. Foram encontras 12 problemas referentes a qualidade dos dataframes e 5 problemas referentes à arrumação dos dados. Um desses problemas é compartilhado. São colunas irrelevantes pois contém muitos valores nulos. Mesmo identificando que existem outros erros, procurei ataca-los visando uma maior velocidade na resolução dos problemas. Creio que em outra oportunidade, tentarei resolve-los visto que tem uma maior complexidade na resolução.

Limpeza – Clean

Eis a parte mais difícil, optei por trabalhar inicialmente com os três dataframes juntos visto que isso pouparia tempo. Essa foi a primeira ação, utilizei o método **merge do pandas** para unir os dataframes que estavam separados. Em relação a poluição dos textos, optei por utilizar a biblioteca **re**, quebrei um pouco a cabeça para encontrar o padrão, mas através de dois links do stackeroverflow consegui dar prosseguimento. Para transformar a coluna timestamp de string para datetime, utilizei mais uma vez um método do pandas **pd.to_datetime.** A variação das raças dos cachorros entre maiúsculas e minúsculas foi utilizada o médoto **lower do str.** Para uma melhor análise resolvi unir as 3 colunas da **confidence prediction** em uma só, assim a confiança da predição poderia ser melhor vista. Renomeie a **p1 para dog_breed** para um melhor entendimento também. Considerei que todo tweet com **p1_dog, p2_dog e p3_dog iguais à False** que os mesmos não era cachorros e os eliminei do dataframe. Retirei as colunas cujo não achei necessária a utilização, em sua maioria pela falta de informação relevante. Uni as colunas **"doggo", "pupper", "floofler" e "puppo"** em uma única coluna chamada **"Dog_stage"** utilizando o método **melt do pandas.** Para os **outliers do denominador e numerador,** peguei o número mais utilizado e o somei a sua metade para descobrir os outliers superiores, e os inferiores, novamente peguei o número mais utilizado e diminui a sua metade, para descobrir os inferiores. Considerei somente os que estavam nessa faixa, os demais um o retirei do dataframe. Os nomes cujo eram artigos: **A, An e The os transformei para None**. E por final, transformei o Tweet ID em um objeto, visto que estava como um inteiro.

Armazenamento - Storing

Armezenei o dataframe com uma csv, seguindo o pedido da Udacity para este projeto através de um método do Pandas.