



Tweepy

Componentes do grupo

- Cassiano Kunsch das Neves
- Hélio Schimitt



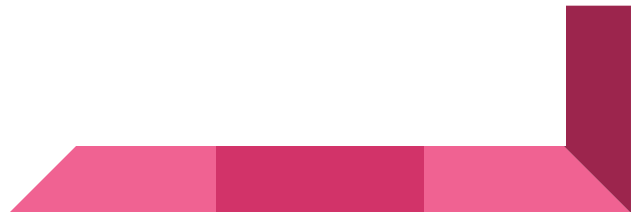
Objetivo

O trabalho terá como objetivo usar a API Tweepy, para mineração de dados da rede social Twitter. Os dados minerados são do mesmo tema do trabalho 1, esses dados serão armazenados no banco de dados para uma futura análise.



Tweepy, o que é?

Tweepy é uma API open-source, hospedado no GitHub e usa a linguagem de programação Python para se comunicar com a plataforma do Twitter e usar sua API.



Captura dos dados

- Para se capturar os twitters foi necessário criar uma conta e criar uma aplicação para poder ter acesso a obtenção dos dados;
- Após criar a conta foi criado o código para obter os dados. Neste código tem um método que fica responsável por pegar o tweet e armazenar em uma lista, após chegar na quantidade de tweet que foi setado, os tweets pegos são salvos no banco de dados (código próximo slide);



```
class StdOutListener(tweepy.StreamListener):

    def on_status(self, status):
        global cont
        global limit
        cont = cont + 1
        print(cont)
        # Captura os twitters e coloca em uma lista
        # usei utf-8 para conseguir armazenar os tweets
        p_palavras.append(status.text.encode('utf-8'))


        # quando chegar na quantidade de tweet que eu quero vai salvar no banco
        if (cont == qtd_tweet):
            salvar_bd()

    def on_error(self, status_code):
        print('Got an error with status code: ' + str(status_code))
        return True # To continue listening

    def on_timeout(self):
        print('Timeout...')
        return True # To continue listening
```

```
if __name__ == '__main__':  
  
    # minhas credenciais  
    consumer_key = **sua consumer_key**  
    consumer_secret = **sua consumer_secret**  
    access_token = **seu access_token**  
    access_token_secret = **seu access_token_secret**  
  
    # instanciando o "escutador"  
    listener = StdOutListener()  
  
    # passando as credenciais  
    auth = tweepy.OAuthHandler(consumer_key, consumer_secret)  
    auth.set_access_token(access_token, access_token_secret)  
  
    # criando a stream e passando o escutador e as credenciais  
    stream = tweepy.Stream(auth, listener)  
  
    # passando as palavras que eu quero que sejam procuradas nos tweets  
    stream.filter(track=['temperatura', 'transporte', 'caixa termica', 'orgao', 'doação', 'transplante'])
```

Nesse código é colocado a identificação da conta. Logo após instanciamos a classe responsável por fazer o controle de pegar o Twitters. Depois passamos a regra de Twitter que queremos, com determinadas palavras. Esses Twitters são guardados em uma lista para depois serem inseridos no banco.



Conexão com o banco?

Psycopg2

Para a conexão com o banco de dados, foi usada a biblioteca do Python, o Psycopg. Ele é o mais popular adaptador de banco de dados para PostgreSQL, com a linguagem de programação Python.



Gravação dos dados no banco

- Para a gravação foi feito um código em Python usando a biblioteca `psycopg2`. Neste código foi criada a tabela no banco caso ela não exista, e foi inserido cada tweet que foi capturado (código no próximo slide);



```
def salvar_bd():
    # Conecta com o BD criado no postgres
    DB = psycopg2.connect(host='localhost', dbname='SMO', user='postgres', password='Cc98576036')

    cursor = DB.cursor()

    # criando a tabela se ela não existir
    cursor.execute('CREATE TABLE IF NOT EXISTS table_twitter (id serial PRIMARY KEY, twitter varchar(250));')

    # varrendo cada tweet armazenado na lista de tweet
    for tweets in lst_tweet:
        # tirando caracter de bycode(b'')
        tweet = str(tweets).split("")
        # inserindo o tweet no banco
        cursor.execute("INSERT INTO table_twitter (twitter) VALUES ('"+tweet[1]+"');")

    # commitando os inserts
    DB.commit()
    cursor.close()

    # fechando o banco
    DB.close()

    print("Salvo com sucesso!")
    sys.exit()
```

