BIG DATA

LABDATA

Introdução ao Big Data

Tema da Aula: **Twitter Streaming API**

Prof.: **Dino Magri**

21 de Novembro de 2018

Coordenação:

Prof. Dr. Adolpho Walter Pimazzi Canton

Profa. Dra. Alessandra de Ávila Montini

LABDATA

Coordenação:

Prof. Dr. Adolpho Walter Pimazzi Canton

Profa. Dra. Alessandra de Ávila Montini

- Contatos:
  - E-mail: professor.dinomagri@gmail.com
  - Twitter: https://twitter.com/prof_dinomagri
  - LinkedIn:http://www.linkedin.com/in/dinomagri
  - Site: http://www.dinomagri.com

Currículo

- **(2014-Presente)** – Professor no curso de Extensão, Pós e MBA na Fundação Instituto de Administração (FIA) – www.fia.com.br
- **(2013-Presente)** – Pesquisa e Desenvolvimento no Laboratório de Arquitetura e Redes de Computadores (LARC) na Universidade de São Paulo – www.larc.usp.br
- **(2013)** – Professor no MBA em Desenvolvimento de Inovações Tecnológicas para WEB na IMED Passo Fundo – RS – www.imed.edu.br
- **(2012)** – Bacharel em Ciência da Computação pela Universidade do Estado de Santa Catarina (UDESC) – www.cct.udesc.br
- **(2009/2010)** – Pesquisador e Desenvolvedor no Centro de Computação Gráfica – Guimarães – Portugal – www.ccg.pt
- **Lattes:** http://lattes.cnpq.br/5673884504184733

Material das aulas

- Material das aulas estão disponíveis em:
  - https://urls.dinomagri.com/posmba-turma9
    - **Senha**: `turma9`
    - **Data Expiração:** `31-12-2018`

Material das aulas

- Caso esteja utilizando seu próprio computador, realize o download de todos os arquivos e salve na **Área de Trabalho** para facilitar o acesso.
  - Lembre-se de instalar os softwares necessários conforme descrito no documento de Instalação (**`InstalaçãoPython3v1.1.pdf`**).

- Nos computadores da FIA os arquivos já estão disponíveis, bem como a instalação dos softwares necessários.

Conteúdo da Aula

- Objetivo
- Twitter Streaming API
- Exercícios

Conteúdo da Aula

- **Objetivo**
- Twitter Streaming API
- Exercícios

Objetivo

- Objetivo dessa aula é introduzir os conceitos sobre as **APIs do Twitter** e como utilizá-las.

Conteúdo da Aula

- Objetivo
- **Twitter Streaming API**
- Exercícios

Twitter Streaming API

- Como vimos o módulo Tweepy facilita o uso das APIs do Twitter, pois tem métodos que lidam com a autenticação, conexão, criação e destruição de sessões, leitura de mensagens de entrada, entre outros.
- A API de **Streaming do Twitter** é utilizada para realizar o download das mensagens em **tempo real**.

</> Abra o arquivo **"aula5-parte1-demo.ipynb"**

O que iremos fazer?

- Desafio 1 - Recuperar dados do Twitter e imprimir na tela
- Desafio 2 - Recuperar dados do Twitter e salvar em um arquivo
- Desafio 3 - Recuperar dados do Twitter durante 5 minutos e salvar em um arquivo
- Desafio 4 - Realizar o tratamento dos dados salvos

O que iremos fazer?

- **Desafio 1 - Recuperar dados do Twitter e imprimir na tela**
- Desafio 2 - Recuperar dados do Twitter e salvar em um arquivo
- Desafio 3 - Recuperar dados do Twitter durante 5 minutos e salvar em um arquivo
- Desafio 4 - Realizar o tratamento dos dados salvos

</> Abra o arquivo **"aula5-parte2-desafio1.ipynb"**

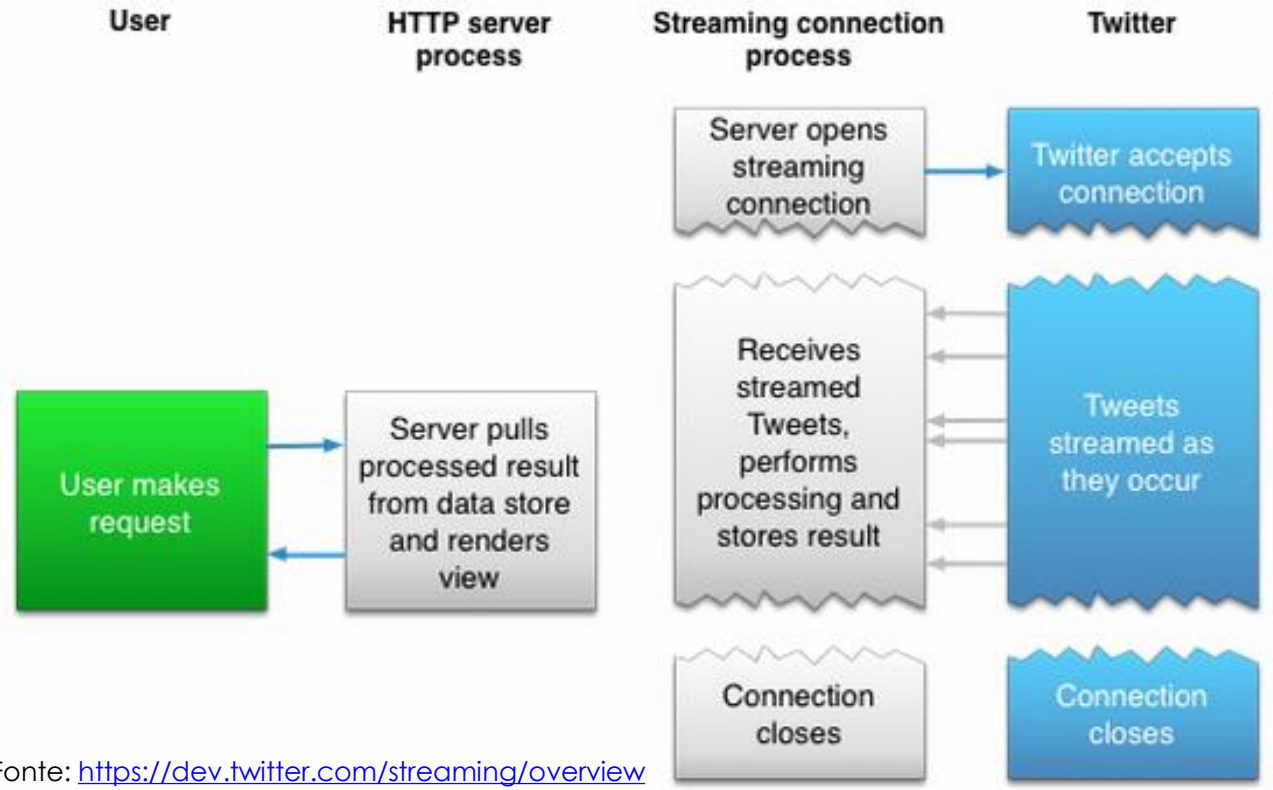
O que iremos fazer?

- **Desafio 1** - Recuperar dados do Twitter e imprimir na tela

O que precisamos aprender!

1. **Entender a API de Streaming do Twitter**
2. Criar uma classe herdando os atributos da classe StreamListener
3. Instanciar essa classe e utilizá-la para se conectar a API através de um objeto de Stream.
4. Iniciar um fluxo (Stream)

1. Entender a API de Streaming do Twitter



Fonte: https://dev.twitter.com/streaming/overview

Podemos ter acesso continuo de novas respostas para as mesmas requisições das consultas realizadas pela REST API através de conexões HTTP utilizando a **Streaming API**.

1. Entender a API de Streaming do Twitter

- Podemos utilizar diversos pontos de fluxo (streaming), cada um customizado para um fim.

Tipo	Descrição
Fluxos públicos (Public Stream)	Acesso ao fluxo publico de dados através do Twitter. Perfeito para seguir um usuário ou tópico específico, além de data mining.
Fluxos do usuário (User Stream)	Fluxo de único-usuário contém todos os dados correspondente com uma visão única do usuário do Twitter.
Fluxos de site (Site Stream)	A versão multiusuário dos fluxos de usuário. É destinado para servidores que devem se conectar ao Twitter em benefício de vários usuários.

1. Entender a API de Streaming do Twitter

- API de Streaming é diferente da API REST pois a API REST é utilizada para puxar dados do Twitter, enquanto que a API de Streaming empurra as mensagens para uma sessão persistente.

- Isso permite que a API de Streaming consiga realizar o download dos dados em tempo real.

https://dev.twitter.com/streaming/overview

1. Entender a API de Streaming do Twitter

- No Tweepy, iremos utilizar uma instância da classe **`tweepy.Stream`**.
- Essa classe estabelece uma sessão de streaming e encaminha as mensagens para a instância **`StreamListener`**.

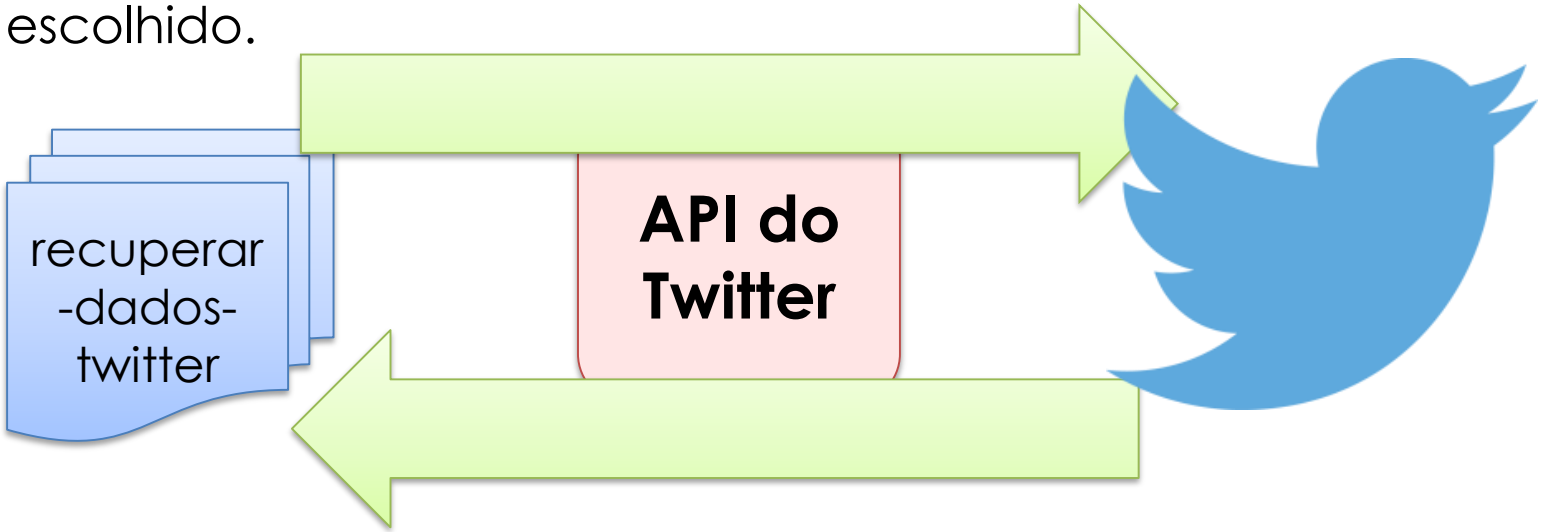
Lembre-se da aula 2 sobre **classes** e **objetos**.

1. Entender a API de Streaming do Twitter

- Na classe StreamListener o método **`on_data`** irá receber todas as mensagens e chamadas da função de acordo com o tipo da mensagem.
- Além disso o método on_data do StreamListener convenientemente passa os dados do status para o método **`on_status`**.
- Isso facilita o acesso as mensagens!

1. Entender a API de Streaming do Twitter

- Solicitar para a API os tweets públicos, aplicando o filtro escolhido.



O Twitter irá devolver os dados solicitados, **em tempo real**

1. Entender a API de Streaming do Twitter

- Para estabelecer essa sessão de streaming, temos que:
  1. **Criar** uma **classe herdando** os atributos da classe **StreamListener**
  2. Utilizar a classe para **criar** um objeto **Stream**
  3. **Conectar** à API **utilizando o Stream**
- Além disso, temos que ter as credenciais de acesso (consumer key, consumer secret, access token, access token secret).

O que precisamos aprender!

1. ~~Entender a API de Streaming do Twitter~~
2. **Criar uma classe herdando os atributos da classe StreamListener**
3. **Instanciar essa classe e utilizá-la para se conectar a API através de um objeto de Stream.**
4. Iniciar um fluxo (Stream)

Vamos ao código!

O que precisamos aprender!

1. ~~Entender a API de Streaming do Twitter~~
2. ~~Criar uma classe herdando os atributos da classe StreamListener~~
3. ~~Instanciar essa classe e utilizá-la para se conectar a API através de um objeto de Stream.~~
4. **Iniciar um fluxo (Stream)**

5. Iniciar um fluxo (Stream)

- Podemos realizar o Stream através do *filter*, *user_stream* ou *sitestream*, conforme explicado anteriormente.
- Neste exemplo iremos utilizar o **`filter`** para capturar todos os tweets contendo a palavra "Big Data".
- O parâmetro **`track`** é uma lista de termos a serem pesquisados e recuperados.

```
fluxo.filter(track=['Big Data'])
```

5. Iniciar um fluxo (Stream)

- Sobre o parâmetro **track**: https://dev.twitter.com/streaming/overview/request-parameters#track

Parâmetro	Irá corresponder		Não irá corresponder
Twitter	TWITTER	twitter	TwitterTracker #newtwitter
	"Twitter"	twitter.	
	#twitter	@twitter	
	http://twitter.com		
Twitter's	I like Twitter's new design		Someday I'd like to visit @Twitter's office
twitter api,twitter streaming	The Twitter API is awesome The twitter streaming service is fast Twitter has a streaming API		I'm new to Twitter
example.com	Someday I will visit example.com		There is no example.com/foobarbaz
example.com/foobarbaz	example.com/foobarbaz www.example.com/foobarbaz		example.com

5. Iniciar um fluxo (Stream)

- Ainda existem outros parâmetros que podem ser utilizados na método `filter`. Por exemplo, podemos recuperar os tweets que sejam de um determinado idioma, utilizando o parâmetro `languages=['pt', 'en']`.

5. Iniciar um fluxo (Stream)

- Outros parâmetros podem ser visualizados em:

  https://dev.twitter.com/streaming/overview/request-parameters

Vamos ao código!

O que precisamos aprender!

1. ~~Entender a API de Streaming do Twitter~~
2. ~~Criar uma classe herdando os atributos da classe StreamListener~~
3. ~~Instanciar essa classe e utilizá-la para se conectar a API através de um objeto de Stream.~~
4. ~~Iniciar um fluxo (Stream)~~

O que iremos fazer?

- Desafio 1 - Recuperar dados do Twitter e imprimir na tela
- **Desafio 2 - Recuperar dados do Twitter e salvar em um arquivo**
- Desafio 3 - Recuperar dados do Twitter durante 5 minutos e salvar em um arquivo
- Desafio 4 - Realizar o tratamento dos dados salvos

</> Abra o arquivo **"aula5-parte3-desafio2.ipynb"**

O que iremos fazer?

- **Desafio 2** - Recuperar dados do Twitter e salvar em um arquivo

O que precisamos aprender!

1. **Entender a diferença entre os métodos on_status e on_data**
2. Modificar nossa classe para utilizar o método on_data
3. Revisar a função embutida open para salvar os arquivos

1. Diferenças entre os métodos on_status e on_data

- O método `on_status` lida apenas com dados do status.
- Enquanto que o método `on_data` recebe todas as mensagens e chamadas das funções de acordo com o tipo da mensagem.
- Ou seja, podemos acessar dados como:
  - Respostas para status, deletar, eventos, mensagens diretas, amigos, entre outros.

O que precisamos aprender!

1. ~~Entender a diferença entre os métodos on_status e on_data~~
2. **Modificar nossa classe para utilizar o método on_data**
3. Revisar a função embutida open para salvar os arquivos

2. Modificar nossa classe para utilizar o método on_data

- Antes de modificar nossa classe, temos que entender o formato em que os dados são retornados do Twitter
- O formato é o JSON – JavaScript Object Notation
  - Os arquivos salvos nesse formato têm a extensão .json
  - É utilizado para serialização e transmissão de informações pela rede:
    - Devido ao seu formato, trafega uma quantidade menor de dados.
    - Além de facilitar a leitura dessas informações

2. Modificar nossa classe para utilizar o método on_data

- A sintaxe do JSON:
  - Chaves são utilizadas para envolver objetos {}
  - Colchetes são utilizados para envolver listas []
  - Os dados são colocados em pares de nome/valor
  - Os dados são separados por vírgula

2. Modificar nossa classe para utilizar o método on_data

```
{
        "pessoa" : {
                "nome" : "Dino Magri",
                "idade" : 28,
                "formacao" : "BS em Ciência da Computação",
                "habilidades" : [
                        "python", "big data",
                        "machine learning", "spark"
                ]
        }
}
```

2. Modificar nossa classe para utilizar o método on_data

- E como podemos ler esses dados no Python?
- Simples, vamos utilizar a biblioteca JSON que já vem com a instalação do Python 3.
- Essa biblioteca permite codificar e decodificar de maneira simples, rápida e completa os dados que estão nesse formato.

</> Abra o arquivo **"aula5-parte3-json.ipynb"**

Vamos ao código!

O que precisamos aprender!

1. ~~Entender a diferença entre os métodos on_status e on_data~~
2. ~~Modificar nossa classe para utilizar o método on_data~~
3. **Revisar a função embutida open para salvar os arquivos**

3. Revisar a função embutida `open` para salvar os arquivos

- A função `open` abre um arquivo e retorna o objeto correspondente.

```
arquivo = open(arq, mode='r')
```

- O `arq` é o nome e o caminho do arquivo que será aberto.

Modo	Significado
`'r'`	Abre para leitura (padrão)
`'w'`	Abre para escrita, truncando o arquivo primeiro
`'x'`	Abre para criação exclusiva, falha se o arquivo já existe
`'a'`	Abre para a escrita, acrescentando ao fim do arquivo, se existir
`'b'`	Modo binário
`'+'`	Abre um arquivo em disco para atualização (leitura e escrita)

Vamos ao código!

O que precisamos aprender!

1. ~~Entender a diferença entre os métodos on_status e on_data~~
2. ~~Modificar nossa classe para utilizar o método on_data~~
3. ~~Revisar a função embutida open para salvar os arquivos~~

O que iremos fazer?

- Desafio 1 - Recuperar dados do Twitter e imprimir na tela
- Desafio 2 - Recuperar dados do Twitter e salvar em um arquivo
- **Desafio 3 - Recuperar dados do Twitter durante 5 minutos e salvar em um arquivo**
- Desafio 4 - Realizar o tratamento dos dados salvos

</> Abra o arquivo **"aula5-parte4-desafio3.ipynb"**

O que iremos fazer?

- **Desafio 3** - Recuperar dados do Twitter durante 5 minutos e salvar em um arquivo

O que precisamos aprender!

1. **Entender como funcionar o módulo time**
2. Modificar nossa classe para ter duração de 5 minutos

1. Entender como funcionar o módulo time

- O módulo time, fornece diversas funções relacionadas ao tempo.
- Como por exemplo:
  - `sleep(s)` – Para a execução do código por **s** segundos.

```
>>> from time import sleep
>>> sleep(5)
>>> print("Dormiu por 5 segundos")
```

  - `time()` – Retorna o tempo em segundos desde a Época.
    - Época é o inicio onde o tempo inicia. Para saber o inicio, utilize a função `gmtime`

```
>>> from time import gmtime
>>> print(gmtime(0))
```

1. Entender como funcionar o módulo time

- Com isso, podemos por exemplo:
  - Definir o tempo inicial da execução através do time()
  - Definir um tempo limite de execução - 5 minutos
  - Verificar se o tempo atual – tempo inicial é menor que o tempo limite
    - Se for igual, salva em arquivo e continua executando.
    - Se não for igual, fechar o arquivo e finaliza a execução.

O que precisamos aprender!

1. ~~Entender como funcionar o módulo time~~
2. **Modificar nossa classe para ter duração de 5 minutos**

Vamos ao código!

O que precisamos aprender!

1. ~~Entender como funcionar o módulo time~~
2. ~~Modificar nossa classe para ter duração de 5 minutos~~

O que iremos fazer?

- Desafio 1 - Recuperar dados do Twitter e imprimir na tela
- Desafio 2 - Recuperar dados do Twitter e salvar em um arquivo
- Desafio 3 - Recuperar dados do Twitter durante 5 minutos e salvar em um arquivo
- **Desafio 4 - Realizar o tratamento dos dados salvos**

</> Abra o arquivo **"aula5-parte5-desafio4.ipynb"**

O que iremos fazer?

- **Desafio 4** - Realizar o tratamento dos dados salvos

O que precisamos aprender!

1. ~~Introdução ao Pandas – Já visto nas aulas anteriores!~~
2. **Carregar o arquivo JSON**
3. Definir o que iremos salvar!
4. Entender o módulo Geopy e criar função que retorna a latitude e longitude
5. Preparar os dados para a visualização
6. Salvar os dados em CSV

2. Carregar o arquivo JSON

- No desafio anterior criamos um arquivo chamado `'tweets_5minutos.json'`, que contém os tweets que salvamos.
- Vamos salvar cada objeto JSON para ser uma linha no DataFrame do Pandas.
- Para carregar o arquivo, vamos utilizar a função open, junto com o comando with.
- O with é uma estrutura de controle de fluxo.

2. Carregar o arquivo JSON

- O with deixa o código mais simples se comparado a forma que era feita antes (via bloco **try...finally**) para garantir que o código seja executado.
- Seu funcionamento básico:

```
>>> with open('arquivo.json', 'r') as arquivo:
        for linha in arquivo:
            print(linha)
```

Vamos ao código!

O que precisamos aprender!

1. ~~Introdução ao Pandas – Já visto nas aulas anteriores!~~
2. ~~Carregar o arquivo JSON~~
3. **Definir o que iremos salvar!**
4. Entender o módulo Geopy e criar função que retorna a latitude e longitude
5. Preparar os dados para a visualização
6. Salvar os dados em CSV

3. Definir o que iremos salvar!

- Legal, salvamos e carregamos muitos dados! Será que temos que utilizar todos?
- Quais dados queremos analisar?
  - Informações do tweet
  - Informações do usuário
  - Informações do perfil do Usuário
- Abra o arquivo exemplo.json que está na pasta da disciplina.

3. Definir o que iremos salvar!

- **Informações do Tweet**
  - Texto do tweet (`text`)
  - Data da criação (`created_at`)
  - Quantidade de vezes que foi favoritado (`favorite_count`)
  - Idioma do tweet (`lang`)
  - Local do tweet (`coordinates`)
  - Quantidade de retweets (`retweet_count`)
  - URL de origem (`source`)

3. Definir o que iremos salvar!

- **Informações do Usuário**
  - Nome do usuário (`'user' -> 'screen_name'`)
  - Data da criação (`'user' -> 'created_at'`)
  - Idioma do usuário (`'user' -> 'lang'`)
  - Quantidade de amigos do usuário (`'user' -> 'friends_count'`)
  - Quantidade de seguidores (`'user' -> followers_count '`)

3. Definir o que iremos salvar!

- **Informações do perfil do Usuário**
  - Cor de fundo do perfil (`'user' -> 'profile_background_color'`)
  - URL da imagem de fundo (`'user' -> 'profile_background_image_url'`)
  - URL da imagem do perfil (`'user' -> 'profile_image_url'`)

Vamos ao código!

O que precisamos aprender!

1. ~~Introdução ao Pandas – Já visto nas aulas anteriores!~~
2. ~~Carregar o arquivo JSON~~
3. ~~Definir o que iremos salvar!~~
4. **Entender o módulo Geopy e criar função que retorna a latitude e longitude**
5. Preparar os dados para a visualização
6. Salvar os dados em CSV

4. O módulo Geopy

- É um serviço web para codificação geográfica diversa.
- Permite localizar coordenadas de endereços, cidades, países e pontos de referência.
- Geopy funciona para Python 2 e 3.
- Maiores informações em: http://geopy.readthedocs.io/en/latest/
- **Limites**: 1 requisição por segundo:
  - https://operations.osmfoundation.org/policies/nominatim/

4. O módulo Geopy

- A biblioteca `geopy` contêm uma classe chamada **`Nominatim`** que utiliza o nome de um local para gerar os códigos de latitude e longitude.
- Essa classe está dentro do **módulo** geocoders que abstrai os serviços das APIs que são utilizadas.
- Essa classe utiliza o OpenStreetMap para recuperar as informações de geolocalização (http://nominatim.openstreetmap.org/).

4. O módulo Geopy

- Uso básico:

```
>>> from geopy.geocoders import Nominatim

>>> geolocalizador = Nominatim()

>>> localizacao = geolocalizador.geocode("Av. Paulista, 302,
SP")

>>> print(localizacao.address)
Edifício José Martins Borges, 302, Avenida Paulista, Bela
Vista, SP, Microrregião de São Paulo, RMSP, Mesorregião
Metropolitana de São Paulo, SP, Região Sudeste, 01310-000,
Brasil
```

4. O módulo Geopy

- Uso básico:

```
>>> print((localizacao.latitude, localizacao.longitude))

(-23.5691802, -46.646645)


>>> print(localizacao.raw)
{'place_id': '62189735', 'licence': 'Data © OpenStreetMap contributors, ODbL 1.0. http://www.openstreetmap.org/copyright', 'osm_type': 'node', 'osm_id': '4962188947', 'boundingbox': ['-23.5692302', '-23.5691302', '-46.646695', '-46.646595'], 'lat': '-23.5691802', 'lon': '-46.646645', 'display_name': 'Edifício José Martins Borges, 302, Avenida Paulista, Bela Vista, SP, Microrregião de São Paulo, RMSP, Mesorregião Metropolitana de São Paulo, SP, Região Sudeste, 01310-000, Brasil', 'class': 'place', 'type': 'house', 'importance': 0.421}
```

Vamos ao código!

O que precisamos aprender!

1. ~~Introdução ao Pandas – Já visto nas aulas anteriores!~~
2. ~~Carregar o arquivo JSON~~
3. ~~Definir o que iremos salvar!~~
4. ~~Entender o módulo Geopy e criar função que retorna a latitude e longitude~~
5. **Preparar os dados para a visualização**
6. Salvar os dados em CSV

5. Preparar os dados para a visualização

- Já carregamos os dados no formato JSON.
- Criamos o DataFrame com as colunas necessárias.
- Criamos uma função para pegar a latitude e longitude.
- **O que falta fazer?**

5. Preparar os dados para a visualização

- Criar uma coluna com as **hashtags** do texto.
- Adicionar as colunas de latitude, longitude e hashtags em nosso DataFrame.
- Repetir esse processo para todos os tweets salvos.
- É importante lembrar que um dos nossos objetivos é plotar em um mapa as localizações dos usuários.
  - **O que não pode faltar?**

Vamos ao código!

O que precisamos aprender!

1. ~~Introdução ao Pandas – Já visto nas aulas anteriores!~~
2. ~~Carregar o arquivo JSON~~
3. ~~Definir o que iremos salvar!~~
4. ~~Entender o módulo Geopy e criar função que retorna a latitude e longitude~~
5. ~~Preparar os dados para a visualização~~
6. **Salvar os dados em CSV**

Vamos ao código!

O que precisamos aprender!

1. ~~Introdução ao Pandas – Já visto nas aulas anteriores!~~
2. ~~Carregar o arquivo JSON~~
3. ~~Definir o que iremos salvar!~~
4. ~~Entender o módulo Geopy e criar função que retorna a latitude e longitude~~
5. ~~Preparar os dados para a visualização~~
6. ~~Salvar os dados em CSV~~

Conteúdo da Aula

- Objetivo
- Twitter Streaming API
- **Exercícios**

Exercícios

- **Exercício 1** – Modifique o código da Streaming API para recuperar os tweets do Brasil que tenham os seguintes termos:
  - Big Data, Hadoop, Spark, Python, Data Science
- O que podemos fazer com os dados recuperados?
- Quais elementos podemos salvar em um DataFrame para posterior análise?

Exercícios

- **Exercício 2** – Crie uma proposta para uso das APIs do Twitter (REST e Streaming).
  - Qual é o principal objetivo da proposta?
  - Quais informações devemos salvar?
  - Será necessário realizar algum tratamento dos dados?
  - Onde podemos armazenar esses dados?
- Adicione outras informações que jugarem necessárias.

Referências Bibliográficas

- **21 Recipes for Mining Twitter** – Matthew A. Russell – USA: O'Reilly, 2011.
- **Mastering pandas** – Femi Anthony – Packt Publishing, 2015.
- **Data Science from Scratch** – Joel Grus – O'Reilly, 2015.
- **Python for Data Analysis** – Wes McKinney – USA: O'Reilly, 2013.
- Referência da API do Tweepy - http://docs.tweepy.org/en/latest/api.html

Referências Bibliográficas

- **Python for kids – A playful Introduction to programming** – Jason R. Briggs – San Francisco – CA: No Starch Press, 2013.

- **Python Cookbook** – David Beazley & Brian K. Jones – O'Reilly, 3th Edition, 2013.

- As referências de links utilizados podem ser visualizados em http://urls.dinomagri.com/refs