

FACULDADE ANGLO-AMERICANO DE FOZ DO IGUAÇU

THIAGO MEDEIROS DE SOUZA

**MINERAÇÃO DE DADOS APLICADO À REDE SOCIAL *TWITTER*
UTILIZANDO A LINGUAGEM DE PROGRAMAÇÃO PYTHON**

FOZ DO IGUAÇU

2016

THIAGO MEDEIROS DE SOUZA

**MINERAÇÃO DE DADOS APLICADO À REDE SOCIAL *TWITTER*
UTILIZANDO A LINGUAGEM DE PROGRAMAÇÃO PYTHON**

Trabalho de conclusão de curso apresentado
como requisito obrigatório para obtenção do tí-
tulo de Bacharel em Ciência da Computação da
Faculdade Anglo-Americano de Foz do Iguaçu.

Orientador: Prof. Msc. Valmei Abreu Júnior

Coorientador: Prof. Esp. João Paulo de Lima
Barbosa

FOZ DO IGUAÇU

2016

TERMO DE APROVAÇÃO

THIAGO MEDEIROS DE SOUZA

MINERAÇÃO DE DADOS APLICADO À REDE SOCIAL *TWITTER* UTILIZANDO A LINGUAGEM DE PROGRAMAÇÃO PYTHON

Trabalho de conclusão de curso apresentado como requisito obrigatório para obtenção do título de Bacharel em Ciência da Computação da Faculdade Anglo-Americano de Foz do Iguaçu, pela seguinte banca examinadora:

Prof. Msc. Valmei Abreu Júnior
Faculdade Anglo-Americano
(Orientador)

Prof. Banca 2
Faculdade Anglo-Americano

Prof. Banca 3
Faculdade Anglo-Americano

Foz do Iguaçu, 26 de abril de 2016

*Dedico este trabalho a meus pais,
Valmir M. de Souza e Maria Emilia M. de Souza
que, com muito amor, me ensinaram os valores da vida.*

AGRADECIMENTOS

Primeiramente agradeço a Deus por sua graça e salvação.

À minha família, por terem me proporcionado oportunidades únicas e as melhores condições de estudo.

À Elyn Hsu, por me mostrar o caminho da disciplina e amor e também pela sua beleza e seu sorriso que me incentivaram durante esta jornada.

Aos meus grandes amigos, Daniel Gonzalez Maciel e Jann Claude Mousquer, por me acompanharem no caminho da vida profissional.

A todos os professores que fizeram parte desta importante etapa da minha vida.

Aos meus orientadores, Valmei Abreu Júnior e João Paulo de Lima Barbosa, por toda a disponibilidade e orientação.

*"Diggin deep for eternal treasure
Stay away from quicksand and false pleasure."
(Matthew Paul Miller - Matisyahu)*

RESUMO

Este trabalho tem como objetivo analisar e minerar os dados provenientes da rede social *Twitter*, com a finalidade de encontrar padrões em perfis de profissionais de Tecnologia da Informação. Para a realização desta atividade será utilizada a linguagem de programação Python como ferramenta principal para o estudo e implementação prática de métodos de aprendizado de máquina para o reconhecimento e apresentação dos dados.

Palavras-chaves: Dados. Data Mining. Twitter. Python.

ABSTRACT

This paper aims to analyze and mine the data from the social network *Twitter*, in order to find patterns in Information Technology professional's profiles. For this activity will be used Python as the main programming language as well as a tool for the study and practical implementation of machine learning methods for data recognition and presentation.

Keywords: Data. Data Mining. Twitter. Python.

LISTA DE ILUSTRAÇÕES

| | |
|--|----|
| FIGURA 1 – Etapas do processo de KDD | 23 |
| FIGURA 2 – Exemplo de uma <i>Series</i> | 32 |
| FIGURA 3 – Criação de um <i>DataFrame</i> | 33 |
| FIGURA 4 – Conteúdo de um <i>DataFrame</i> pelo interpretador <i>IPython</i> | 33 |
| FIGURA 5 – Exemplo de um gráfico gerado pelo <i>matplotlib</i> | 34 |
| FIGURA 6 – Exemplo de uma página <i>web</i> do <i>IPython Notebook</i> | 36 |
| FIGURA 7 – Execução do <i>script</i> para coleta de dados | 43 |
| FIGURA 8 – <i>Dirty Data</i> presente no arquivo coletado | 48 |
| FIGURA 9 – Utilizando o comando <i>grep</i> para gerar um novo arquivo sem <i>dirty data</i> | 48 |
| FIGURA 10 – Línguas que mais realizaram <i>tweets</i> | 50 |
| FIGURA 11 – Países que mais realizaram <i>tweets</i> | 51 |

LISTA DE TABELAS

| | |
|---|----|
| TABELA 1 – Cronograma de execução | 18 |
|---|----|

LISTA DE ABREVIATURAS

| | |
|-------|--|
| API | <i>Application Programming Interface</i> - Interface de Programação de Aplicação |
| BMP | <i>Windows Bitmap</i> |
| CGI | <i>Common Gateway Interface</i> - Interface Comum de Entrada ¹ |
| CSV | <i>Comma-Separated Values</i> - Valores Separados Por Vírgula ¹ |
| DBA | <i>Database Administrator</i> - Administrador de Banco de Dados |
| FTP | <i>File Transfer Protocol</i> - Protocolo de Transferência de Arquivos |
| GIF | <i>Graphics Interchange Format</i> - Formato Para Intercâmbio de Gráficos ¹ |
| GUI | <i>Graphical User Interface</i> - Interface Gráfica do Usuário |
| HTTP | <i>Hypertext Transfer Protocol</i> - Protocolo de Transferência de Hipertexto |
| HTTPS | <i>Hyper Text Transfer Protocol Secure</i> - Protocolo de Transferência de Hipertexto Seguro |
| IETF | <i>Internet Engineering Task Force</i> |
| IMAP | <i>Internet Message Access Protocol</i> - Protocolo de Acesso a Mensagem da Internet |
| IP | <i>Internet Protocol</i> - Protocolo de Internet |
| JPG | <i>Joint Photographic Experts Group</i> |
| JSON | <i>JavaScript Object Notation</i> - Notação de Objeto JavaScript ¹ |
| KDD | <i>Knowledge Discovery From Data</i> - Descoberta de Conhecimento por Dados |
| PDF | <i>Portable Document Format</i> - Formato de Documento Portátil ¹ |
| PNG | <i>Portable Network Graphics</i> - Rede Portável de Gráficos ¹ |
| POP | <i>Post Office Protocol</i> - Protocolo dos Correios |

¹ Tradução do Autor

| | |
|-------|---|
| RFC | <i>Request for Comments</i> - Pedido Para Comentários |
| RPC | <i>Remote Procedure Call</i> - Chamada Remota de Procedimento ² |
| SMTP | <i>Simple Mail Transfer Protocol</i> - Protocolo de Transferência de Correio Simples |
| SSL | <i>Secure Sockets Layer</i> - Camada Segura de Sockets |
| SVG | <i>Scalable Vector Graphics</i> - Gráficos Vetoriais Escaláveis |
| TCP | <i>Transmission Control Protocol</i> - Protocolo de Controle de Transmissão |
| URI | <i>Uniform Resource Identifier</i> - Identificador Uniforme de Recursos |
| XHTML | <i>eXtensible Hypertext Markup Language</i> - Linguagem de Marcação de Hipertexto Extensiva |
| XML | <i>eXtensible Markup Language</i> - Linguagem de Marcação Extensiva |
| YML | <i>Yet Another Markup Language</i> - Uma Outra Linguagem de Marcação ² |

² Tradução do Autor

SUMÁRIO

| | | |
|----------|--|-----------|
| 1 | INTRODUÇÃO | 14 |
| 1.1 | JUSTIFICATIVA [FAZENDO] | 16 |
| 1.2 | OBJETIVOS | 17 |
| 1.2.1 | Objetivo Geral | 17 |
| 1.2.2 | Objetivos Específicos | 17 |
| 1.3 | CRONOGRAMA DE ATIVIDADES | 17 |
| 1.4 | ORGANIZAÇÃO DO TRABALHO | 18 |
| 2 | REVISÃO BIBLIOGRÁFICA | 20 |
| 3 | FUNDAMENTAÇÃO TEÓRICA | 22 |
| 3.1 | DESCOBERTA DE CONHECIMENTO EM BASE DE DADOS E <i>DATA MINING</i> | 22 |
| 3.2 | <i>MACHINE LEARNING</i> | 26 |
| 3.3 | LINGUAGEM PYTHON | 28 |
| 4 | MATERIAIS E MÉTODOS | 30 |
| 4.1 | TECNOLOGIAS E FERRAMENTAS | 30 |
| 4.1.1 | Bibliotecas da Linguagem Python | 30 |
| 4.1.1.1 | <i>Biblioteca NumPy</i> | 30 |
| 4.1.1.2 | <i>Biblioteca pandas</i> | 32 |
| 4.1.1.3 | <i>Biblioteca matplotlib</i> | 33 |
| 4.1.1.4 | <i>Biblioteca SciPy</i> | 34 |
| 4.1.1.5 | Interpretador <i>IPython</i> | 35 |
| 4.1.1.6 | <i>Biblioteca tweepy</i> | 36 |
| 4.1.2 | Interface de Programação de Aplicações - API | 36 |
| 4.1.2.1 | Arquitetura REST | 36 |
| 4.1.3 | Protocolo de Autenticação - OAuth | 37 |
| 4.1.3.1 | Protocolo OAuth 1.0a | 38 |
| 4.1.3.2 | Protocolo OAuth 2.0 | 39 |
| 4.1.4 | Rede Social <i>Twitter</i> | 39 |
| 4.1.4.1 | API <i>Twitter</i> | 39 |
| 4.2 | METODOLOGIA E DESENVOLVIMENTO | 40 |
| 4.2.1 | Iterativo e Incremental | 40 |
| 4.2.2 | Etapas Para a Mineração de Dados do <i>LinkedIn</i> | 40 |
| 4.2.2.1 | <i>Clustering</i> | 40 |
| 4.2.2.2 | Normalização de Dados | 40 |
| 4.2.2.3 | Computação de Similaridade | 41 |

| | | |
|----------|--|-----------|
| 5 | IMPLEMENTAÇÃO DAS TÉCNICAS E ANÁLISE DOS RESULTADOS | 42 |
| 5.1 | INTRODUÇÃO | 42 |
| 5.2 | COLETA DE DADOS | 42 |
| 5.3 | ANÁLISE DE DADOS | 47 |
| 6 | CONCLUSÕES E SUGESTÕES PARA FUTUROS TRABALHOS . . | 52 |
| 6.1 | CONCLUSÕES | 52 |
| 6.2 | SUGESTÕES PARA FUTUROS TRABALHOS | 52 |
| | REFERÊNCIAS | 53 |

1 INTRODUÇÃO

Redes sociais se tornaram um termo comum e uma chave fundamental para o estilo de vida moderno. Hoje em dia, a maioria das pessoas, independente de idade, sexo, crença, utilizam uma ou mais redes sociais. A princípio, esses ambientes *on-line* focavam-se na comunicação, por exemplo; a possibilidade de se comunicar com alguém distante e tornar esse diálogo pessoal, seguro e, de alguma forma, próximo, ajudando na popularização desse tipo de tecnologia. No decorrer dos anos e com o avanço tecnológico, diferentes tipos de redes sociais surgiram com ideias semelhantes ou extremamente diferentes, não sendo apenas para a comunicação, mas para outros fins como o compartilhamento de mídias, localização, críticas, *mini-blogs*, perguntas e respostas, negócios, profissão, música, artes, venda e troca de produtos, entre outros.

Facebook, *Twitter*, *LinkedIn*, *Google+* e, muito comum entre desenvolvedores, o *GitHub* são exemplos populares de redes sociais. Logo, possuem grande número de usuários e diversas interações que estes realizam a cada momento, gerando uma quantidade gigantesca de dados. Esses dados são informações sobre pessoas, comportamentos, gostos, marcas e vários outros tipos de conteúdo. Devido a diversidade e a vasta quantidade desse tipo de informação, algumas redes sociais as utilizam para o aprimoramento de conteúdo ou, então, para o comércio de dados para empresas, por exemplo; de publicidade e marketing, que fazem a mineração desses dados para encontrar padrões de seus usuários e, assim, conseguir aumentar suas vendas, reduzir riscos e, até mesmo, gerar novas tendências.

Dados é um termo, deliberadamente vago, que agrega várias formas comuns de informações, como por exemplo matrizes (vetores multidimensionais), tabelas ou planilhas, onde cada coluna pode ter um tipo diferente de informação (caracteres, numéricos, data, entre outros). Essas tabelas podem se relacionar através de colunas chaves apresentada no modelo relacional de Codd (1969 apud JANSSENS, 2014).

Certamente que todos esses exemplos citados não demonstram a totalidade e nem toda a abordagem para a palavra dados. Não é sempre que grande percentual de um conjunto de dados pode ser transformados em uma forma estruturada, onde é possível serem analisados e modelados.

Cientistas de dados precisam visualizá-los com o objetivo de produzir resultados claros e serem capazes de informar ao seus mantenedores sobre a situação atual e a qualquer momento. Este é o verdadeiro valor que um cientista nessa área precisa prover.

A mineração de dados, também conhecida como *data mining*, é o processo de

analisar dados em diferentes perspectivas e transformar em informação útil. Hoje em dia, o *data mining* é usado por companhias com grande foco em varejo, finanças, comunicação e marketing, para conseguirem determinar as relações de fatores internos como preço, posição de produto, ou habilidade de recurso humano, e fatores externos como indicadores econômicos, competições e população demográfica de clientes (RUSSELL, 2013).

Essa análise de dados consiste em visualizar informações em diferentes maneiras e formas, plotando gráficos e planilhas. Com isso, novas informações aparecerão permitindo alguma previsão ou predição desse conteúdo. As observações levarão a uma reflexão que resultará em possibilidades ou probabilidades concretas para se exercer uma atividade. No primeiro momento, essas informações são amorfas e, após a análise, se transformará em ideias (HAN et al., 2012).

Para que essas ideias se tornem um trabalho futuro é preciso capturá-las e interpretá-las através de um modelo de extração de conhecimento. Esse modelo, geralmente, é um processo que apresenta etapas que vão do armazenamento dos dados em estudos até processos matemáticos, estatísticos e computacionais, com o objetivo de extrair informações úteis. Um modelo então, é muito mais que apenas a descrição dos dados, incorpora o entendimento de todo o processo da origem dos dados até a competência deles. Logo, ele consegue fazer previsões sobre os conhecimentos analisados (HAN et al., 2012).

Para conseguir fazer melhores previsões é preciso desenvolver métodos mais sofisticados antes de formular um modelo relevante. Com isso, a dificuldade aumenta e, então, é necessário implementar um modelo computacional que consiga obter possíveis resultados através do reconhecimento desses dados.

Para a análise e a interação de dados, computação exploratória e visualização de dados, a linguagem de programação Python vai, inevitavelmente, ser comparada a muitas outras, tanto no domínio de software livre, como também, com linguagens e ferramentas comerciais, como R, MATLAB, SAS, Stata e outros. Atualmente, o Python possui bibliotecas que se tornaram fortes alternativas para a tarefa de manipulação de dados. Combinado com o poder de programação que a linguagem tem, é uma excelente escolha como linguagem para a construção de aplicações centradas em dados (MCKINNEY, 2013).

Em muitas organizações, é comum realizar pesquisas, prototipar e testar novas ideias utilizando mais de um domínio específico de linguagem computacional, como MATLAB ou R e, posteriormente, estas ideias viram parte de um sistema de produção maior, escrito, por exemplo, em Java, C#, ou C++. Kaldero (2015), afirma que Python não é somente uma linguagem adequada para a pesquisa e prototipagem, mas também para o desenvolvimento de sistemas.

Devido a esta solução de apenas uma única linguagem, as organizações podem se beneficiar, tendo cientistas e tecnólogos usando o mesmo conjunto de ferramentas programáticas. Portanto, Python é a ferramenta escolhida pela maioria desses profissionais. Essa escolha se deve, não somente a alta produtividade que a linguagem fornece, mas também por ela ser uma ferramenta comum a diferentes times e organizações (KALDERO, 2015).

Python é uma linguagem de programação livre e multiplataforma, possui uma excelente documentação e está sobre cuidado de uma enorme comunidade, onde é possível obter ajuda e melhores soluções para problemas durante a codificação. Tem como grande vantagem a facilidade de aprendizado, porque foi desenvolvida para ser simples e descomplicada. É uma linguagem interpretada, dinamicamente tipada, com grande precisão e sintaxe eficiente. Tem grande popularidade para analisar dados devido ao enorme poder que suas bibliotecas possuem (*NumPy*, *SciPy*, *pandas*, *matplotlib*, *IPython*). A linguagem apresenta alta produtividade para prototipação, desenvolvimento de sistemas menores e reaproveitáveis.

A mineração de dados busca então, extrair dos dados o conhecimento útil para algum objetivo específico. Entretanto, a tarefa de extração de conhecimento é complexa devido a multidisciplinaridade envolvida no seu processo de extração e, também, por não ter um modelo de mineração genérico para a busca de informação útil. Para isso, é necessário o uso de ferramentas que viabilizam essas tarefas. Logo, Python dispõe de um conjunto de bibliotecas para a análise e mineração de dados extremamente poderosas e com uma curva de aprendizado curta, graças a sintaxe clara e descomplicada que a linguagem fornece.

1.1 JUSTIFICATIVA [FAZENDO]

A rede social *Twitter* é uma fonte rica de dados e um ponto de partida inicial para minerar redes sociais, graças a sua abertura para o consumo público, API bem documentada, e vasta quantidade de informações devido a interação dos usuários a todo instante. Os dados do *Twitter* são particularmente interessante porque *tweets*, frases postadas na rede social, segundo Russell (2013), acontecem na "velocidade do pensamento" e logo estão disponíveis para o consumo.

A necessidade de compartilhar ideias e experiências permite que pessoas se relacionem, são ouvidas, e se sentem importantes e parte de um meio.

Para que os dados se tornem uma informação útil é preciso saber como coletar, processar, modelar e visualizar. Portanto, este estudo é tratado como uma área interdisciplinar que depende de uma arquitetura de *software* apropriada, técnicas de processamento massivo de dados, algoritmos de redução de dimensionalidade, mo-

delagem estatística e computacional, visualização de dados, entre outros.

1.2 OBJETIVOS

1.2.1 Objetivo Geral

Este trabalho tem como objetivo principal utilizar técnicas e algoritmos de *data mining*, como o aprendizado de máquina (*machine learning*), para a análise e mineração de dados provenientes da rede social *Twitter*, utilizando os recursos e bibliotecas que a linguagem de programação Python possui.

1.2.2 Objetivos Específicos

- Identificar os conceitos sobre KDD e *data mining*;
- Descrever as técnicas de *data mining*;
- Explorar as funcionalidades das bibliotecas de mineração e visualização da linguagem Python;
- Examinar e utilizar a API da rede social *Twitter*;
- Encontrar padrões utilizando técnicas e algoritmos de aprendizado de máquina (*machine learning*);
- Compreender e aplicar técnicas de geolocalização para melhor apresentação de resultados;
- Apresentar testes e resultados obtidos da análise e mineração dos dados.

1.3 CRONOGRAMA DE ATIVIDADES

As atividades a serem executadas no decorrer do projeto visando o êxito do mesmo, estão listados a seguir e especificados em meses na Tabela 1:

- Estudo e Pesquisa: aquisição dos conhecimentos pertinentes e necessários para o desenvolvimento do projeto;
- Análise de Requisitos: levantamento dos requisitos do projeto;
- Geração do Documento: desenvolvimento das documentações para especificação do projeto;
- Implementação: desenvolvimento dos códigos para a análise de dados;

- Testes: execução dos testes que irão garantir a qualidade das informações a serem geradas;
- Elaboração de Artigos: parte do tempo destinado ao projeto será para desenvolver artigos visando a publicação em eventos da área;
- Apresentação de Resultados: etapas destinadas à apresentação dos resultados parciais e finais.

TABELA 1: Cronograma de execução

| Mês - Ano | 08/15 | 09/15 | 10/15 | 11/15 | 12/15 | 02/16 | 03/16 | 04/16 | 05/16 |
|----------------------------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| Estudo e Pesquisa | X | X | X | X | X | X | X | X | |
| Análise de Requisitos | X | X | X | X | X | X | X | X | |
| Geração do Documento | X | X | X | X | X | X | X | X | X |
| Implementação | | | | X | X | X | X | X | X |
| Testes | | | | X | X | X | X | X | X |
| Elaboração de Artigos | | | X | X | X | | | X | X |
| Apresentação de Resultados | | | | | X | | | | X |

FONTE: Elaborado pelo autor

1.4 ORGANIZAÇÃO DO TRABALHO

Além deste capítulo introdutório, este trabalho é composto de mais seis capítulos.

O Capítulo 2 apresenta os trabalhos que são referências para este estudo.

Os fundamentos teóricos, como os conceitos de *data mining*, e base para o entendimento do tema proposto estão descritos no Capítulo 3.

As bibliotecas da linguagem Python utilizadas para a mineração de dados são expostas no Capítulo 4. Também, são apresentadas neste capítulo a API do *Twitter*, as etapas de *data mining* e outros materiais e metodologias utilizados para execução deste trabalho.

[CUIDAR]

No Capítulo 5 são demonstradas as fases de desenvolvimento de algoritmos para a implementação do *data mining* e *machine learning*.

Os resultados obtidos e a apresentação de planilhas e gráficos das soluções desenvolvidas são apresentados no Capítulo 6.

Por fim, a conclusão deste trabalho se dá no Capítulo 7, onde são abordadas e analisadas as dificuldades, além de determinar as possibilidades para trabalhos futuros.

2 REVISÃO BIBLIOGRÁFICA

Alguns trabalhos serviram como ajuda e inspiração para este estudo. Porém durante o período de busca por bibliografias a respeito de *data mining*, pouco material foi encontrado quanto a mineração em redes sociais. Ainda sim, alguns estudos possuem um destaque e é necessário citá-los, devido a utilização de ferramentas específicas em *data mining* e, também, aos resultados obtidos neste processo.

De acordo com Lemos (2003), um dado se transforma em informação quando ganha um significado para seu utilizador, caso contrário, continua sendo simplesmente um dado.

Em seu estudo, Lemos (2003) aborda duas técnicas de *data mining*: Árvores de Decisão e Redes Neurais, para realizar a análise de crédito bancário. Estas técnicas permitem fazer o reconhecimento de padrões e também diagnosticar novos casos. Através deste modo, os analistas têm condições de diagnosticar os novos clientes, quanto ao merecimento de crédito ou não. Estas técnicas de mineração de dados são ferramentas que podem ser utilizadas pelo especialista para auxiliá-lo nas tomadas de decisão, nunca porém poderão por si só, substituir a figura do especialista no contexto da análise de crédito.

Semelhante as técnicas abordadas para a análise de crédito bancário, Steiner et al. (2004) realizaram um estudo na área médica, onde a posse e uso de ferramentas que auxiliem na tarefa de classificação de pacientes em prováveis ictericos com câncer ou ictericos com cálculo, pode ser crucial. Em uma tentativa de otimizar todo o processo do diagnóstico, minimizando riscos e custos e, por outro lado, maximizando a eficácia nos resultados, utilizaram técnicas de *data mining* como um processo para extração de informações valiosas. O trabalho consistiu na análise de 118 históricos de pacientes utilizando Árvores de Decisão e Regras de Classificação como ferramenta. Os autores afirmam que os métodos de *data mining* apresentam a vantagem de deixar claro ao usuário quais são os atributos que estão discriminando os padrões e de que forma a mesma está ocorrendo, isso é uma característica altamente desejável em qualquer técnica de reconhecimento de padrões.

O reconhecimento de padrões permite a obtenção de conhecimento da frequência com que determinadas seções de uma página *web* são acessadas e quais são os serviços mais procurados. Isso possibilita que empresas consigam descobrir o perfil de seus usuários e, com base nesse acontecimento, ofertar serviços e atendimento personalizado. Essa afirmação foi resultado do estudo de Silva, Boscarioli e Peres (2003), onde realizaram a mineração de dados em *logs* de acesso a servidores

web. Para o desenvolvimento do trabalho utilizaram regras de associação para a descoberta e representação de padrões frequentes em conjuntos de dados. Isso propiciou a identificação de padrões de comportamento de usuários da Internet ao navegarem por *websites*. Também concluem que outras ferramentas de mineração podem ser aplicadas, visando aumentar a flexibilidade de manipulação dos atributos específicos para o ambiente *Web*.

Como visto, o tema sobre *data mining* possui uma abordagem comum a setores e áreas diferentes, o que caracteriza a enorme quantidade de dados que ainda não se tem informação útil para seus utilizadores. Por mais que as técnicas utilizadas dentre essa diversidade de áreas sejam semelhantes, o desenvolvimento e aplicação das técnicas são particulares para cada caso. Neste contexto, esta dissertação modela, de uma forma mais simplificada, o reconhecimento de padrões para a predição dos dados extraídos da rede social *Twitter*.

3 FUNDAMENTAÇÃO TEÓRICA

A mineração de dados é um assunto totalmente interdisciplinar, podendo ser definido de diversas maneiras. Até mesmo o termo *data mining* não representa realmente todos os componentes desta área. Han et al. (2012) exemplificam esta questão comentando sobre a mineração de ouro através da extração de rocha e areia, que é chamado de mineração de ouro e não mineração de rochas ou mineração de areia. Analogamente, a mineração de dados deveria se chamar "mineração de conhecimento através de dados" que, infelizmente, é um termo um tanto longo. Entretanto, uma referência mais curta, como "mineração de conhecimento", pode não enfatizar a mineração de uma enorme quantidade de dados. Apesar disso, a mineração é um termo que caracteriza o processo de encontrar uma pequena quantidade de uma preciosa pepita em uma grande quantidade de matéria bruta. Nesse sentido, um termo impróprio contendo ambos "*data*" e "*mining*" se tornou popular e, como consequência, muitos outros nomes similares surgiram: *knowledge mining from data*, *knowledge extraction*, *data/pattern analysis*, *data archaeology* e *data dredging*.

Na seção 3.1 será abordado os conceitos de descoberta de conhecimento em base de dados e a sua diferença em relação ao *data mining*. Logo após a explicação desta distinção, será apresentado os métodos e a concepção de *data mining*. Alguns desses métodos tem como prática o uso de aprendizado de máquina, que será definido na seção 3.2. A seção 3.3 irá caracterizar a linguagem de programação Python e qual a sua vantagem em utilizá-la para a mineração de dados.

3.1 DESCOBERTA DE CONHECIMENTO EM BASE DE DADOS E *DATA MINING*

Muitas pessoas tratam a mineração de dados como um sinônimo para outro termo muito popular, descoberta de conhecimento em base de dados (*knowledge discovery from data*) - KDD, enquanto outros referenciam *data mining* como apenas uma etapa no processo de descoberta de conhecimento em base de dados. O processo de KDD é demonstrado através da Figura 1 e, posteriormente, listada como uma sequência interativa e iterativa dos seguintes passos:

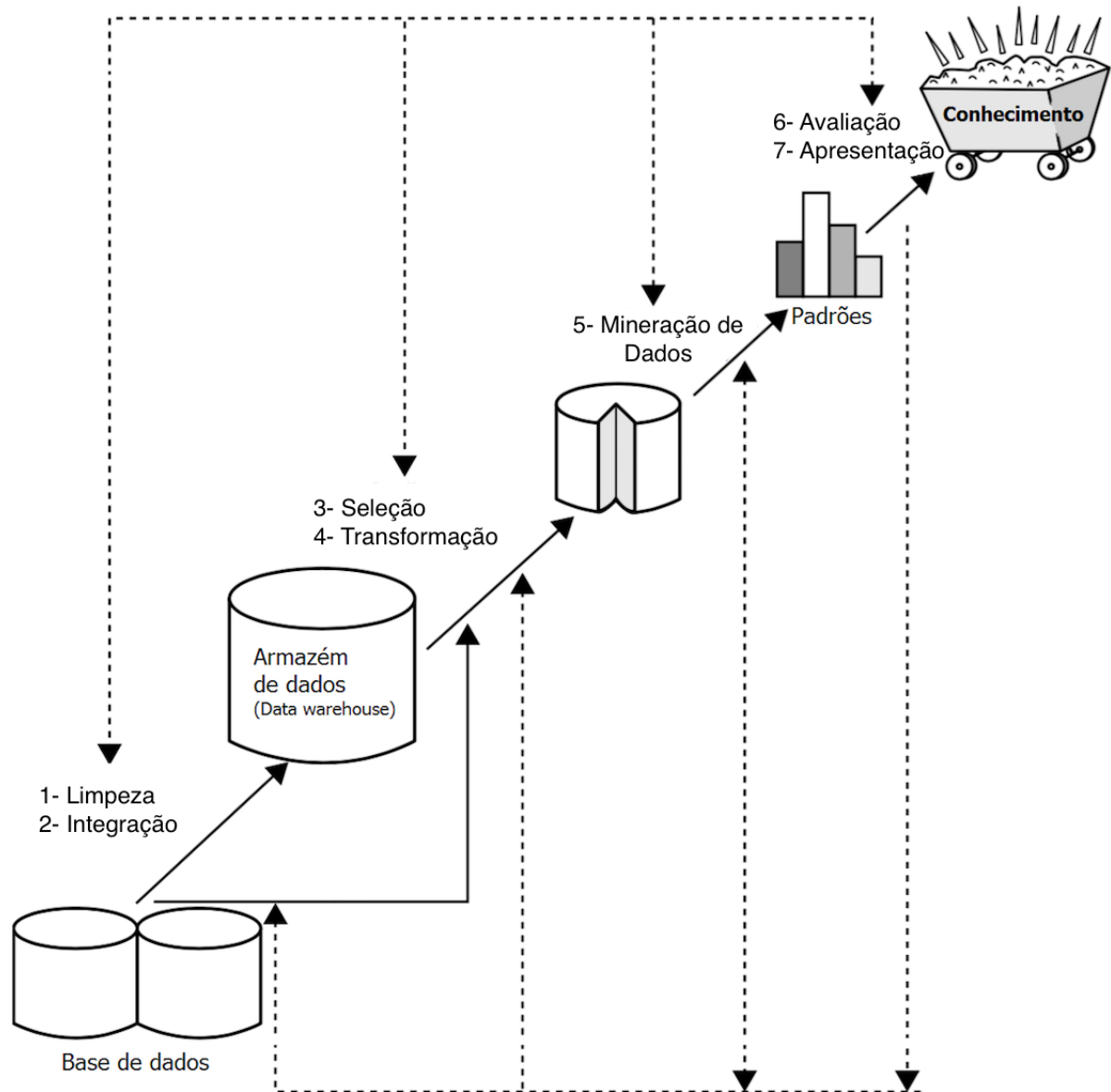


FIGURA 1: Etapas do processo de KDD

FONTE: Adaptado de Han et al. (2012)

1. *Data cleaning* (Limpeza de dados);
2. *Data integration* (Integração de dados);
3. *Data selection* (Seleção de dados);
4. *Data transformation* (Transformação de dados);
5. *Data mining* (Mineração de dados);
6. *Pattern evaluation* (Avaliação de padrões);
7. *Knowledge presentation* (Apresentação de conhecimento).

É importante notar que algum dos processos acontecem na mesma etapa: Limpeza e integração; Seleção e transformação; Avaliação e apresentação.

De acordo com Brachman et al. (1996 apud FAYYAD et al., 1996-b), as etapas são interativas porque envolvem a cooperação da pessoa responsável pela análise de dados, cujo conhecimento sobre o domínio orientará a execução do processo. Por sua vez, a interação deve-se ao fato de que, com frequência, esse processo não é executado de forma sequencial, mas envolve repetidas seleções de parâmetros e conjuntos de dados, aplicações das técnicas de *data mining* e posterior análise dos resultados obtidos, a fim de refinar os conhecimentos extraídos.

KDD refere-se ao processo global de descobrimento de conhecimento útil em bases de dados. *Data mining* é um passo particular neste processo de aplicação de algoritmos específicos para extrair padrões (modelos) de dados. Os passos adicionais no processo KDD, como integração de dados, limpeza, seleção e transformação dos dados, assim como a interpretação e a apresentação dos resultados, asseguram que o conhecimento útil (informação) foi descoberto provenientes da etapa de mineração (HAN et al., 2012). A aplicação cega de métodos de *data mining*, conforme alertado por Navega (2002), pode ser uma atividade perigosa que conduz a descoberta de padrões sem sentido.

O KDD evoluiu e continua evoluindo da interseção de pesquisas em campos como bancos de dados, aprendizado de máquinas (*machine learning*), reconhecimento de padrões, estatísticas, inteligência artificial, aquisição de conhecimento para sistemas especialistas, visualização de dados, descoberta científica, recuperação de informação e computação de alto-desempenho. Aplicações de KDD incorporam teorias, algoritmos e métodos de todos estes campos (LEMOS, 2003).

Apesar do conceito de *data mining*, na maioria das vezes, ser utilizado pelas indústrias, mídias e centros de pesquisa para se referir ao processo de descoberta de conhecimento considerado em sua globalidade, o termo *data mining* pode ser usado

também para indicar o quinto estágio do KDD, sendo um processo essencial na descoberta e extração de padrões de dados. Han et al. (2012), adotam uma visão mais abrangente para a funcionalidade de mineração de dados: *data mining* é o processo de descoberta de padrões interessantes e conhecimentos de um vasto conjunto de dados. A fonte dos dados pode ser banco de dados, *data warehouses*, a Internet, outros repositórios de informações, ou dados correntes em sistemas dinâmicos.

Uma das definições, talvez, mais importante de *data mining* foi elaborada por Fayyad et al. (1996-a) "...o processo não-trivial de identificar, em dados, padrões válidos, novos, potencialmente úteis e ultimamente compreensíveis".

Data mining ou mineração de dados, pode ser entendido então, como o processo de extrair informação, ou conhecimento útil, de algum conjunto de dados e utilizar este conhecimento adquirido para tomada de decisões ou descrever características e padrões descobertos (SFERRA; CORREA, 2003).

Diversos métodos são usados em *data mining* para encontrar respostas ou extrair conhecimento interessante. Esses podem ser obtidos através dos seguintes métodos:

- Classificação: associa ou classifica um item a uma ou várias classes. Os objetivos dessa técnica envolvem a descrição gráfica ou algébrica das características diferenciais das observações de várias populações. A ideia principal é derivar uma regra que possa ser usada para classificar, de forma otimizada, uma nova observação a uma classe já rotulada;
- Modelos de Relacionamento entre Variáveis: associa um item a uma ou mais variáveis de predição de valores reais, conhecidas como variáveis independentes ou exploratórias. Nesta etapa se destacam algumas técnicas estatísticas como regressão linear simples, múltipla e modelos lineares por transformações, com o objetivo de verificar o relacionamento funcional entre duas variáveis quantitativas, ou seja, constatar se há uma relação funcional entre X e Y;
- Análise de Agrupamento (*Cluster*): associa um item a uma ou várias classes (ou *clusters*). Os *clusters* são definidos por meio do agrupamento de dados baseados em modelos probabilísticos ou medidas de similaridade. Analisar *clusters* é uma técnica com o objetivo de detectar a existência de diferentes grupos dentro de um determinado conjunto de dados e, caso exista, determinar quais são eles;
- Sumarização: determina uma descrição compacta para um determinado subconjunto, por exemplos; medidas de posição e variabilidade. Nesta etapa se aplica

algumas funções mais sofisticadas envolvendo técnicas de visualização e a determinação de relações funcionais entre variáveis. Estas funções são usadas para a geração automatizada de relatórios, sendo responsáveis pela descrição compacta de um conjunto de dados;

- **Modelo de Dependência:** descreve dependências significativas entre variáveis. Estes modelos existem em dois níveis: estruturado e quantitativo. O nível estruturado demonstra, através de gráficos, quais variáveis são localmente dependentes. O nível quantitativo especifica o grau de dependência utilizando alguma escala numérica;
- **Regras de Associação:** determinam relações entre campos de um banco de dados. Esta relação é a derivação de correlações multivariadas que permitam auxiliar as tomadas de decisão. Medidas estatísticas, como correlação e testes de hipóteses apropriados, revelam a frequência de uma regra no universo dos dados minerados;
- **Análise de Séries Temporais:** determina características sequenciais, como dados com dependência no tempo. Tem como objetivo modelar o estado do processo extraíndo e registrando desvios e tendências no tempo. As séries são compostas por quatro padrões: tendência, variações cíclicas, variações sazonais e variações irregulares. Existem vários modelos estatísticos que podem ser aplicados a essas situações.

A maioria destes métodos são baseados em técnicas de aprendizado de máquina (*machine learning*), reconhecimento de padrões e estatística. Essas técnicas vão desde estatística multivariada, como análise de agrupamentos e regressões, até modelos mais atuais de aprendizagem, como redes neurais, lógica difusa e algoritmos genéticos (SFERRA; CORREA, 2003).

Devido aos vários métodos estatísticos que são aplicados no processo de *data mining*, Fayyad et al. (1996-a) mostram uma relevância da estatística para o processo de extração de conhecimentos ao afirmar que essa ciência provê uma linguagem e uma estrutura para quantificar a incerteza resultante quando se tenta deduzir padrões de uma amostra a partir de uma população.

3.2 MACHINE LEARNING

Abstratamente, pode-se pensar em *machine learning*, ou aprendizado de máquina, como um conjunto de ferramentas e métodos que tentam inferir padrões e extrair *insights* de uma porção daquilo que se é observado no mundo. Por exemplo, ao

tentar ensinar um computador a reconhecer os códigos postais escritos nos envelopes, os dados podem consistir em fotografias dos envelopes, além de um registro do código postal a que cada envelope estava endereçado, ou seja, dentro de um contexto, é possível selecionar um registro de ações de certos objetos, aprender com este registro e, em seguida, criar um modelo dessas atividades que irão informar a compreensão deste contexto futuramente (CONWAY; WHITE, 2012).

Na prática, isto requer dados e, em aplicações atuais, isso, muitas vezes, significa uma grande quantidade de dados (talvez vários *terabytes*). A maioria das técnicas de aprendizagem automática considera a disponibilidade de tais dados como algo inquestionável, o que significa novas oportunidades para a sua aplicação, em função da quantidade de dados que são produzidos como um produto de administrar companhias modernas.

Machine learning é a intersecção entre ciência da computação, engenharia, estatística e outras disciplinas. É possível ser aplicada em várias áreas, desde políticas a geociência. É uma ferramenta que pode ser utilizada para a solução de vários problemas. Qualquer campo que precisa interpretar e agir sobre dados pode se beneficiar do uso de técnicas de *machine learning* (CONWAY; WHITE, 2012).

A prática de engenharia está em utilizar a ciência para resolver um problema. Em engenharia, é comum resolver um problema determinista, em que a solução dada por humanos sempre resolve o problema. Se desenvolver um software para controlar uma máquina de venda automática é melhor que esta trabalhe sempre, independentemente do dinheiro depositado ou dos botões pressionados. Muitos problemas existem quando a solução não é determinista, isto é, ou não se sabe o suficiente sobre o problema ou não se tem poder computacional suficiente para delineá-lo adequadamente. Para esses problemas, precisa-se de estatísticas.

Uma das tarefas de *machine learning* é a classificação. Na classificação, o trabalho é prever em que classe deve uma porção de dados ser enquadrada. Outra tarefa é a regressão que é a previsão de um valor numérico. Classificação e regressão são exemplos de aprendizado supervisionado, que são conhecidos como supervisionado, por dizer o que o algoritmo deve prever (HARRINGTON, 2012).

O oposto de aprendizagem supervisionada é um conjunto de tarefas conhecidas como aprendizado não supervisionado, onde não há nenhum rótulo ou valor alvo dado para os dados. Uma solução para este tipo de aprendizagem é o agrupamento de itens semelhantes denominado de *clustering*. Na aprendizagem não supervisionada, também pode-se querer encontrar valores estatísticos que descrevem os dados. Isso é conhecido como estimativa da densidade. Outra tarefa do aprendizado não supervisionado está em reduzir dados com várias funcionalidades até se chegar a um número reduzido, em que seja possível visualizá-lo em duas ou três dimensões (HARRING-

TON, 2012).

3.3 LINGUAGEM PYTHON

Python é uma linguagem de programação orientada a objetos, interpretada e interativa. Incorpora módulos, exceções e de tipagem dinâmica alta. Possui uma sintaxe clara e simples, o que facilita o aprendizado para novos desenvolvedores, assim como a rápida leitura e interpretação para usuários mais experientes. Dispõe de interfaces para várias chamadas de sistemas (*system calls*) e bibliotecas, também para vários sistemas de janelas, e é extensível a outras linguagens de programação como C ou C++. É também usada como uma linguagem de extensão para aplicações que precisam de uma interface programática (PYTHON, 2015).

Outra característica da linguagem Python é a portabilidade, podendo ser utilizada em diversos sistemas operacionais como variantes do Unix, em sistemas Mac e também em PCs sob MS-DOS, Windows, Windows-NT, e OS/2.

É uma linguagem de programação de alto-nível que pode ser aplicada em soluções para diversas classes diferentes. Possui uma vasta quantidade de bibliotecas que atende a áreas como o processamento de *strings* (expressões regulares, Unicode, cálculo de diferença entre arquivos), protocolos de Internet (HTTP, FTP, SMTP, XML-RPC, POP, IMAP, CGI *programming*), engenharia de software (testes unitários, registro de logs, *profiling*, análise de código Python), e interfaces para sistemas operacionais (*system calls*, sistemas de arquivos, TCP/IP *sockets*) (PYTHON, 2015).

A sintaxe bastante expressiva e a abundância de suas bibliotecas tornam Python uma ótima linguagem para se obter resultados em várias questões. Algumas de suas utilidades são apresentadas conforme a seguinte lista:

- Escrita de *scripts*: Python é uma ótima linguagem para a criação de *scripts*. É possível usar *scripts* para analisar arquivos de texto, gerar amostra de entradas para testar programas, coletar conteúdos de páginas *web* utilizando a biblioteca *Beautiful Soup*, dentre outras atividades;
- Desenvolvimento *backend* para aplicações *web*: É possível criar APIs (*Application Programming Interface*, apresentado no capítulo 4) e interagir com banco de dados. *Frameworks* mais utilizados inclui *Django*, *Flask* e *Pyramid*;
- Análise e visualização de dados: Conforme o foco deste trabalho, bibliotecas como *pandas*, *NumPy* e recursos semelhantes a outras ferramentas como R e MATLAB estão dispostas através da biblioteca *SciPy*;
- *matplotlib* e *Seaborn*: são mecanismos que possibilitam a visualização dos dados.

A utilização dessa linguagem como ferramenta principal para este trabalho se justifica na utilização dos pacotes que facilitam a análise e interpretação de dados. Como exemplo, *NumPy*, *SciPy*, *pandas*, *matplotlib* e *IPython* são os mecanismos indispensáveis para a mineração de informações juntamente com as estruturas de dados já presentes em Python como os *dictionaries* (estruturas similares ao JSON), que permitem ordenar dados através de um modelo chave-valor. Devido então a sintaxe intuitiva que a linguagem possui e seu excelente ecossistema de bibliotecas, é possível acessar APIs e manipular dados com mais facilidade (RUSSELL, 2013).

4 MATERIAIS E MÉTODOS

Após a revisão bibliográfica de outros estudos e os fundamentos teóricos necessários para a mineração de dados utilizando Python, torna-se importante definir as ferramentas, tecnologias e procedimentos necessários para o desenvolvimento do projeto.

Este capítulo apresenta os materiais e métodos utilizados para a realização do processo de *data mining*, onde, na seção 4.1 são apresentadas as tecnologias e ferramentas que serão utilizadas durante o estudo. Serão abordados quais as bibliotecas que a linguagem Python disponibiliza para a análise e mineração de dados e, também, como acessar a API do *LinkedIn*. Esta será esclarecida também neste capítulo, após a explicação do conceito de API e o protocolo OAuth.

A seção 4.2 irá concluir o capítulo apresentando as etapas de *data mining* com o intuito de evidenciar o processo para a obtenção de conhecimento útil.

4.1 TECNOLOGIAS E FERRAMENTAS

Tecnologias e ferramentas para a criação e prototipagem dos algoritmos.

4.1.1 Bibliotecas da Linguagem Python

Um dos grandes diferenciais da linguagem Python é o seu enorme conjunto de bibliotecas para soluções de diversos problemas.

A seguir serão apresentadas as bibliotecas necessárias para a mineração de dados, através das quais é possível coletar, limpar, transformar, realizar operações e apresentar resultados proveniente dos dados da rede social *LinkedIn*. Para evitar repetições da palavra "biblioteca", o termo "pacote" também será utilizado com o mesmo significado no restante desta dissertação.

4.1.1.1 Biblioteca NumPy

NumPy é o pacote fundamental para computação científica em Python. É o acrônimo para *Numerical Python*. Esta biblioteca provê:

- *ndarray* que é um objeto de matriz multidimensional;
- Funções que permitem realizar operações vetoriais ou operações matemáticas entre matrizes sem a necessidade de programar *loops*;
- Ferramentas para a leitura e escrita em conjuntos de dados matriciais;

- Operações de álgebra linear, transformada de Fourier e geração de números aleatórios;
- Ferramentas para a integração em outras linguagens de programação como C, C++ e Fortran.

Além da capacidade de rápido processamento em matrizes que o *NumPy* oferece ao Python, um dos principais objetivos em relação a análise de dados é que serve como um "container" para os dados serem passado por algoritmos. Para dados numéricos, as matrizes de *NumPy* são muito mais eficientes para a ordenação e manipulação de dados do que qualquer outra estrutura embutida em Python. Igualmente, bibliotecas escritas em linguagens de baixo nível, como C ou Fortran, podem operar dados gravados em matrizes da *NumPy* sem precisar da cópia de qualquer dado (MCKINNEY, 2013).

A biblioteca *NumPy* por si só, não provê uma funcionalidade de alto-nível para a análise de dados. Tendo um conhecimento sobre as matrizes de *NumPy* e matrizes orientadas a computação (*array-oriented computing*) irá facilitar o uso de outras ferramentas, como *pandas*, com mais efetividade.

Para aplicações voltadas para a análise de dados, esta biblioteca possui grande funcionalidade em setores como:

- Criação rápida de matrizes para a interação e limpeza de dados, separação e filtragem, transformação e outros tipos de operações computacionais;
- Algoritmos comuns para matrizes como ordenação, operações únicas e definidas;
- Eficiente descrição estatística e agregação/sumarização de dados;
- Alinhamento de dados e manipulação de dados relacionais para operações de junção e imerção (*join* e *merge*) de conjuntos de dados heterogeneos;
- Expressar lógicas de condições através de expressões matriciais ao invés de laços de repetições e condições como *while*, *for*, *if-elif-else*;
- Agrupamento de manipulação de dados (agregação, transformação, aplicação de funções).

Enquanto *NumPy* oferece o fundamento computacional para essas operações, é preferível utilizar a biblioteca *pandas* como base para a mineração de dados (especialmente de dados estruturados ou dados tabulados), devido a sua interface rica e de alto-nível no qual permite as tarefas com dados mais concisas e simples.

4.1.1.2 Biblioteca *pandas*

A biblioteca *pandas* atrai o maior interesse de cientistas quanto a mineração de dados. Ela possui estruturas de dados de alto-nível e ferramentas de manipulação desenvolvidas para facilitar e agilizar a análise de dados em Python. *pandas* é desenvolvida sob a biblioteca *NumPy* e viabiliza o uso em aplicações centradas nesta. A seguir são expostas algumas soluções que a biblioteca disponibiliza (MCKINNEY, 2013):

- Estrutura de dados com eixos rotulados suportam o alinhamento de dados automáticos ou explícitos. Isso evita erros comuns resultantes de dados desalinhados e dados indexados de formas diferentes provenientes de outras fontes de dados;
- A mesma estrutura de dados consegue manusear tanto dados de séries temporais como dados não-temporais;
- Operações e reduções aritméticas é passado para metadados (eixos rotulados);
- Manipulação flexível de dados em falta;
- *Merge* (fundir) e outras operações relacionais encontradas em bancos de dados relacional.

Esta biblioteca possui duas estrutura de dados principais: *Series* e *DataFrame*. Estas estruturas não são uma solução universal para todos os problemas, mas provê uma base sólida e de fácil manipulação para a maioria das aplicações com mineração de dados.

Uma *Series* é um tipo de *array* ou uma matriz unidimensional, similar a um *array* que possui uma matriz de dados (qualquer tipo de dado da biblioteca *NumPy*) e um outro vetor associado a dados rotulados, chamados de *index* (índice). Uma simples *Series* é formado por uma única matriz de dados, conforme a Figura 2.

```
[In [5]: obj = Series([4, 7, -5, 3])  
  
[In [6]: obj  
Out[6]:  
0      4  
1      7  
2     -5  
3      3  
dtype: int64
```

FIGURA 2: Exemplo de uma *Series*

FONTE: McKinney (2013)

DataFrame representa uma tabela, uma estrutura de dados do tipo planilha, que possui uma coleção ordenada de colunas, onde cada uma delas pode ter um tipo de valor diferente (numérico, *string*, *boolean*, etc.). O *DataFrame* possui um índice para linhas e também para colunas. Pode ser interpretado como um dicionário de *Series*. De uma maneira geral, o dado é armazenado como um ou mais blocos bi-dimensionais ao invés de uma lista, dicionário, ou outro tipo de coleção de matriz unidimensional (MCKINNEY, 2013).

Existem várias maneiras diferentes de se criar um *DataFrame*, entretanto uma forma comum é um dicionário de dimensões iguais, conforme a Figura 3 e a Figura 4, ou uma matriz *NumPy*.

```
data = {'state': ['Ohio', 'Ohio', 'Ohio', 'Nevada', 'Nevada'],
        'year': [2000, 2001, 2002, 2001, 2002],
        'pop': [1.5, 1.7, 3.6, 2.4, 2.9]}

frame = DataFrame(data)
```

FIGURA 3: Criação de um *DataFrame*

FONTE: McKinney (2013)

```
[In [9]: frame
Out[9]:
```

| | pop | state | year |
|---|-----|--------|------|
| 0 | 1.5 | Ohio | 2000 |
| 1 | 1.7 | Ohio | 2001 |
| 2 | 3.6 | Ohio | 2002 |
| 3 | 2.4 | Nevada | 2001 |
| 4 | 2.9 | Nevada | 2002 |

FIGURA 4: Conteúdo de um *DataFrame* pelo interpretador *IPython*

FONTE: McKinney (2013)

4.1.1.3 Biblioteca *matplotlib*

O pacote *matplotlib* é desenvolvido para a geração de gráficos bidimensionais a partir de *arrays*. Gráficos comuns podem ser criados com alta qualidade a partir de simples comandos, inspirados nos comandos gráficos do MATLAB, exemplo ilustrado na Figura 5.

Quando usado em conjunto com ferramentas GUI (*IPython*, por exemplo), esta biblioteca possui recursos interativos como zoom e visão panorâmica. Além disto, suporta várias ferramentas GUI *backend*, nos diversos sistemas operacionais suporta-

dos pelo Python, e permitem exportar gráficos em diversos formatos: PDF, SVG, JPG, PNG, BMP, GIF, etc.

matplotlib também possui várias ferramentas adicionais, como o *mplot3d* para plotar gráficos tridimensionais, e o *basemap* para mapeamentos e projeções.

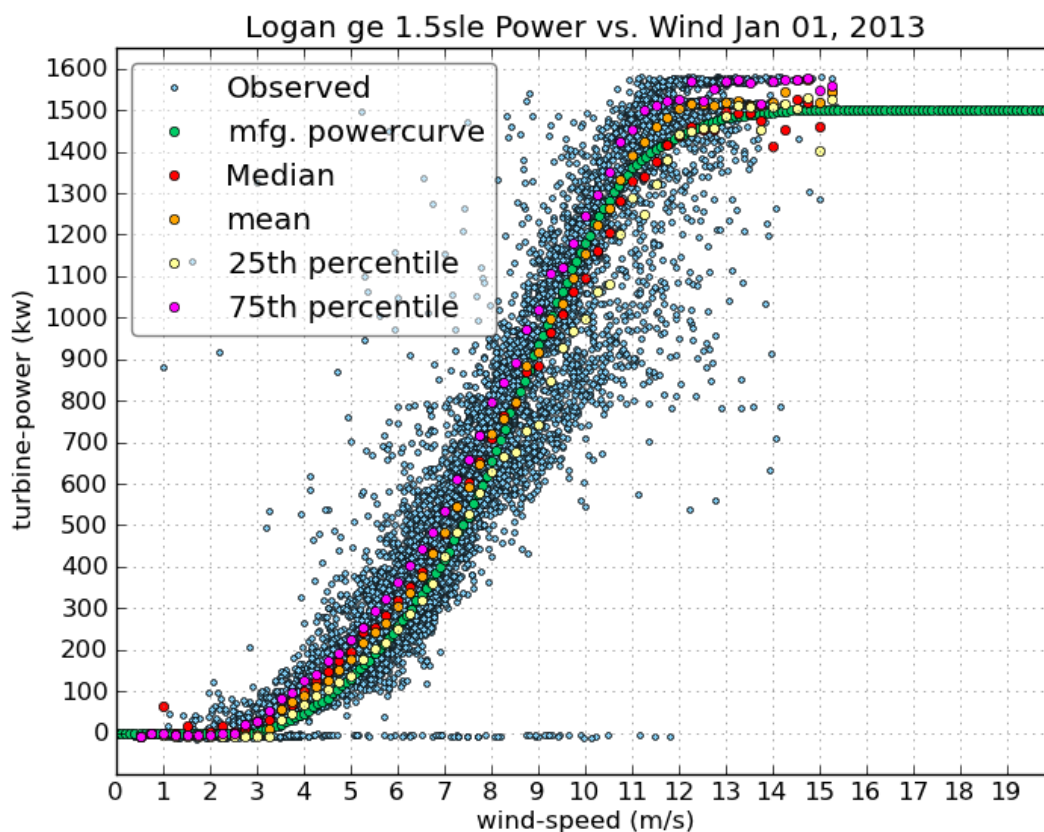


FIGURA 5: Exemplo de um gráfico gerado pelo *matplotlib*

FONTE: Wiener (2014)

4.1.1.4 Biblioteca *SciPy*

SciPy é uma coleção de pacotes que abordam uma série de soluções para diferentes domínios na computação científica. Na lista a seguir são apresentados exemplos desses pacotes (MCKINNEY, 2013):

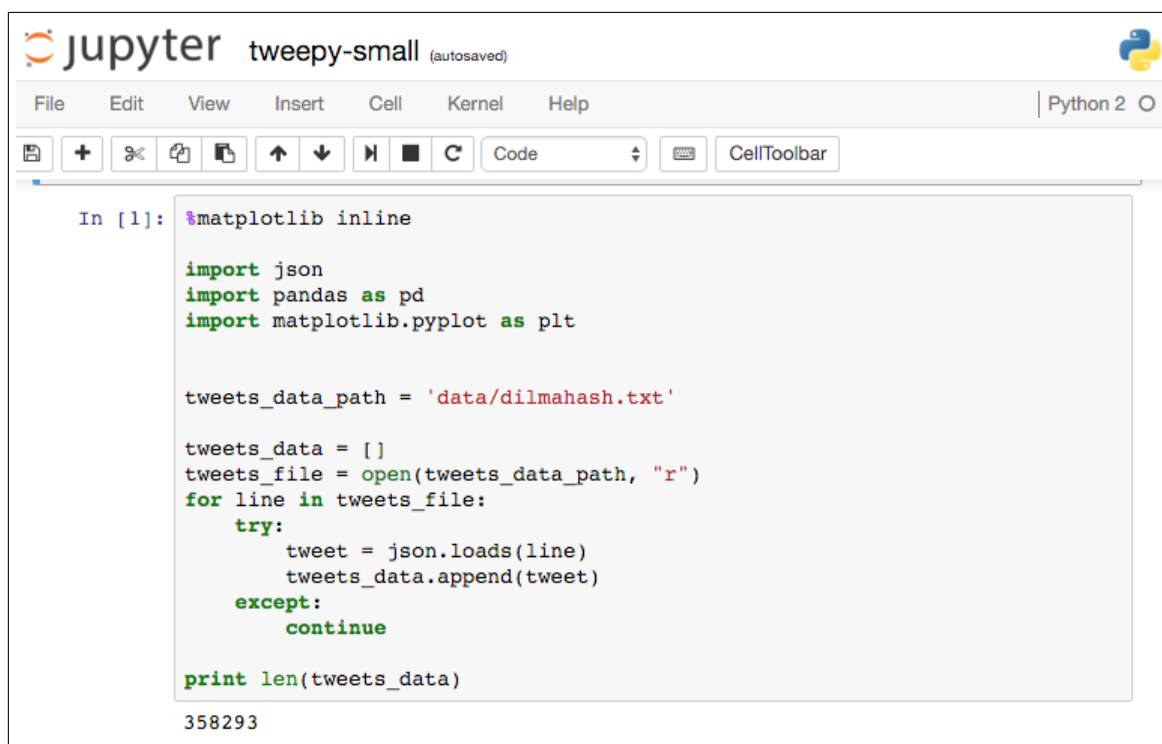
- *scipy.integrate*: rotinas de integração numéricas e soluções de equações diferenciais;
- *scipy.linalg*: rotinas de álgebra linear e decomposição de matrizes;
- *scipy.optimize*: funções otimizadoras (minimizadoras) e algoritmos de busca em raiz;

- *scipy.signal*: ferramentas para processamento de sinais;
- *scipy.sparse*: matrizes esparsas e soluções de sistemas lineares esparsos;
- *scipy.special*: agregador do *SPECFUN*, uma biblioteca do Fortran que implementa várias funções matemáticas, como exemplo, a função gama;
- *scipy.stats*: funções estatísticas, variáveis contínuas e discretas, testes estatísticos e outros modelos estatísticos;
- *scipy.weave*: ferramenta para usar códigos *inline* de C++ para acelerar a computação de matrizes.

4.1.1.5 Interpretador *IPython*

O interpretador *IPython* teve seu desenvolvimento iniciado em 2001, com o intuito de ser um interpretador interativo para a linguagem Python. Desde a sua criação o *IPython* evoluiu grandemente, ao ponto de ser considerada uma das mais importantes ferramentas para computação científica em Python. Essa biblioteca não oferece nenhuma ferramenta para análise de dados ou análise computacional em si, sendo designada para maximizar a produtividade, tanto na interação computacional como no desenvolvimento de softwares. Oferece um fluxo de visualização de um modo *execute-explore* ao invés do típico modelo *edit-compile-run* de muitas outras linguagens de programação. Ela também provê uma pequena integração com o *shell* e sistemas de arquivos. Como a maior parte da programação focada na mineração de dados envolve exploração, tentativa, erro e iteração, *IPython*, em quase todos os casos, irá facilitar este tipo de trabalho (MCKINNEY, 2013).

Hoje, o projeto *IPython*, mantido pela empresa *Jupyter*, engloba muito mais do que apenas um interpretador *shell* para Python. Ele também inclui um console gráfico interativo, o *IPython Notebook*, que provê ao usuário uma experiência de caderno (*notebook-like*) através de um navegador *web*, conforme Figura 6, e dispõe de um mecanismo de processamento paralelo. Assim como muitas outras ferramentas desenvolvidas para programadores, é extremamente customizável (RUSSELL, 2013).



The screenshot shows a Jupyter Notebook window titled "tweepy-small (autosaved)". The interface includes a menu bar (File, Edit, View, Insert, Cell, Kernel, Help) and a toolbar with icons for file operations and execution. The code cell contains the following Python code:

```
In [1]: %matplotlib inline

import json
import pandas as pd
import matplotlib.pyplot as plt

tweets_data_path = 'data/dilmahash.txt'

tweets_data = []
tweets_file = open(tweets_data_path, "r")
for line in tweets_file:
    try:
        tweet = json.loads(line)
        tweets_data.append(tweet)
    except:
        continue

print len(tweets_data)
```

The output of the code cell is the number 358293.

FIGURA 6: Exemplo de uma página web do *IPython Notebook*

FONTE: Elaborado pelo autor

4.1.1.6 Biblioteca *tweepy*

Esta biblioteca provê ao Python a API do *Twitter*. Através da utilização do protocolo OAuth 1.0a, é possível acessar diversos campos como *Profile*, *Group*, *Company*, *Jobs*, *Search*, *Share*, *Network* e requisições REST APIs (TWEETPY, 2009).

4.1.2 Interface de Programação de Aplicações - API

API é uma sigla para *Application Programming Interface* e basicamente é uma tecnologia que permite um pedaço de *software* se comunicar com outro pedaço de *software*. Existem vários tipos de API e é comumente referenciado a outras tecnologias. Por exemplo, para o desenvolvimento deste trabalho será utilizado a API do *LinkedIn*.

Uma API é composta por uma série de funções acessíveis somente por programação, e que permitem utilizar características do *software* menos evidentes ao utilizador tradicional.

4.1.2.1 Arquitetura REST

Abreviação para Transferência de Estado Representacional (REST), é um estilo arquitetural baseado em recursos e nas representações desses recursos. Enfatiza a escalabilidade na interação entre componentes, a generalidade de interfaces, a im-

plantação independente dos componentes de um sistema, o uso de componentes intermediários visando a redução na latência de interações, o reforço na segurança e o encapsulamento de sistemas legados. A REST ignora os detalhes da implementação de componente e a sintaxe de protocolo com o objetivo de focar nos papéis dos componentes, nas restrições sobre sua interação com outros componentes e na sua interpretação de elementos de dados significantes (FIELDING, 2000).

REST foi um termo criado por Fielding (2000), onde ele modela um estilo de arquitetura para a construção de serviços *web* consistentes e coesos. O estilo da arquitetura REST é baseado em recursos e nos estados desses recursos.

A funcionalidade de uma REST API é similar ao funcionamento de uma página *web*, onde o usuário efetua uma requisição a um servidor *web*, utilizando o protocolo HTTP, e recebe dados como resposta.

Um recurso é qualquer conteúdo ou informação que é exposto na Internet, podendo ser um documento, vídeo clip, até processos de negócio ou dispositivos. Para utilizar um recurso é necessário ser capaz de identificá-lo na rede e de ter meios para manipulá-lo. Tem-se então, o *Uniform Resource Identifiers* (URI) para este propósito. Um URI unicamente identifica um recurso e, ao mesmo tempo, o torna endereçável ou capaz de ser manipulado utilizando um protocolo, como o HTTP. O URI de um recurso se distingue dos de qualquer outro recurso e é através do próprio URI que ocorrem as interações com o recurso (WEBBER; PARASTATIDIS; ROBINSON, 2010).

Recursos devem possuir pelo menos um identificador para ser endereçável, e cada identificador é associado com uma ou mais representações, que é uma transformação ou uma visão do estado do recurso em um instante de tempo. Essa visão é codificada em um ou mais formatos transferíveis, tal como XHTML, Atom, texto simples, XML, YAML, JSON, JPG, MP3, entre outros (WEBBER; PARASTATIDIS; ROBINSON, 2010).

Os recursos provêm o conteúdo ou objeto com o qual se quer interagir e para atuar sobre eles é utilizado os métodos de HTTP. Os métodos HTTP na arquitetura REST podem ser referenciados como Verbos, uma vez que representam ações sobre os recursos (WEBBER; PARASTATIDIS; ROBINSON, 2010).

4.1.3 Protocolo de Autenticação - OAuth

Protocolos de autenticação são capazes de, simplesmente, autenticar a parte que está se conectando, ou ainda de autenticar a parte que está conectando, assim como se autenticar para ele.

Neste trabalho será utilizado apenas o protocolo OAuth 2.0 para o acesso aos dados do *LinkedIn*. É possível também, realizar a autenticação utilizando a versão

mais antiga, OAuth 1.0a, mas será apenas referenciado, neste trabalho, para a melhor compreensão do funcionamento do protocolo.

OAuth é uma sigla para "*open authorization*", ou autorização aberta, e provê um meio para que usuários autorizem uma aplicação acessar dados, com alguma finalidade, através de uma API, sem que os usuários precisem passar credenciais como nome de usuário e senha. De um modo geral, usuários são capazes de controlar o nível de acesso para estas aplicações e revogar este controle a qualquer momento (RUSSELL, 2013).

4.1.3.1 Protocolo OAuth 1.0a

OAuth 1.0a é um protocolo que permite que um cliente (*client*) *web* tenha acesso a um recurso protegido pelo seu dono em um servidor. Esta definição se dá através da RFC 5849. Que são documentos técnicos desenvolvidos e mantidos pelo Internet Engineering Task Force (IETF), instituição que especifica os padrões que serão implementados e utilizados em toda a Internet.

A razão para a existência dessa tecnologia é para evitar problemas de usuários (donos dos recursos) compartilhar suas senhas com aplicações *web*.

A versão OAuth 1.0a não permite que credenciais sejam trocadas utilizando uma conexão *Secure Socket Layer* (SSL) através de um protocolo HTTPS. Por esse motivo, muitos desenvolvedores achavam tedioso o trabalho devido aos vários detalhes envolvidos em encriptação.

SSL é um padrão global para tecnologia de segurança. Tem como função principal criar um canal criptografado entre um servidor *web* e um navegador (*browser*) para garantir que todos os dados transmitidos sejam seguros e sigilosos.

Uma aplicação que está requerindo acesso é conhecida como *client*, em alguns momentos chamado de *consumer*, a rede social ou o serviço que contém os recursos protegidos é nomeado como *server* (também chamado de *provider*) e o usuário que concede o acesso é o *resource owner* (dono do recurso, tradução livre). Com estes elementos, as três participações que envolvem o processo e a interação que estes elementos possuem é conhecida como "*three-legged-flow*" ou de uma maneira mais coloquial, a *OAuth dance*. Estas são as etapas fundamentais que envolvem a *OAuth dance* que, como resultado, permite ao *client* o acesso a recursos protegidos, conforme listado a seguir (RUSSELL, 2013):

1. O *client* obtém um *token* de requisição do servidor de serviço (aplicação);
2. O dono do recurso autoriza o *token* de requisição;
3. O *client* troca o *token* de requisição por um *token* de acesso;

4. O *client* usa o *token* de acesso para acessar os recursos protegidos com a consideração do dono do recurso.

Para credenciais particulares, um *client* começa com uma *consumer key* e um *consumer secret* e no fim do processo de *OAuth dance*, termina com um *token* de acesso e *token* de acesso secreto que pode ser usado para acessar recursos protegidos.

4.1.3.2 Protocolo OAuth 2.0

Enquanto o protocolo OAuth 1.0a permite uma autorização útil para o acesso a aplicações *web*, o OAuth 2.0 foi originalmente destinado a simplificar, significativamente, a implementação detalhada para desenvolvedores de aplicações *web*, baseando-se completamente no SSL para aspectos de segurança e para satisfazer uma vasta quantidade de casos de uso. Esses casos de uso variaram desde suporte para dispositivos móveis à necessidades empresariais e, conseqüentemente, às necessidades de um termo mais futuro, da "Internet das Coisas" (RUSSELL, 2013).

Diferentemente da implementação OAuth 1.0a, que consiste de um rígido conjunto de etapas, a implementação do OAuth 2.0, definido através do RFC 6749, pode variar de acordo com a particularidade do caso de uso. Um decorrer típico da execução do OAuth 2.0 tem a vantagem do SSL e, essencialmente, contém apenas poucos redirecionamentos que, acompanhada de em alto-nível, não possui tanta diferença em relação ao processo anterior que envolvem um ciclo do OAuth 1.0a.

4.1.4 Rede Social *Twitter*

Na subseção 4.1.4.1 é apresentado a API do *Twitter* e como se obter dados através dela.

4.1.4.1 API *Twitter*

O acesso a API se dá através da criação de uma aplicação pela página *web* de desenvolvimento do *Twitter*. Após a criação da aplicação é fornecido ao usuário informações para o acesso utilizando o protocolo OAuth, será informado uma chave da API da aplicação, uma chave secreta, o *token* de usuário OAuth e credenciais secretas do usuário OAuth.

Dispondo de todas as credenciais do protocolo OAuth o acesso à API acontece utilizando a biblioteca *tweepy*, especificada na subseção 4.1.1.

4.2 METODOLOGIA E DESENVOLVIMENTO

O processo de desenvolvimento da solução segue uma série de princípios de conjunto de boas práticas e etapas do *data mining*, para melhor estruturar e obter, não só o resultado esperado, mas também para que todo o processo ocorra de forma coerente e padronizada.

Nesta seção são apresentadas as principais metodologias utilizadas neste trabalho. Na seção XX ...

4.2.1 Iterativo e Incremental

4.2.2 Etapas Para a Mineração de Dados do *LinkedIn*

Abordado previamente, o acesso aos dados do *LinkedIn* acontece através da utilização de sua API e a interpretação dos dados se dá pela aplicação da técnica de *clustering*, normalização de dados e computação de similaridade.

4.2.2.1 *Clustering*

A técnica de *clustering* é um aprendizado não supervisionado que está presente em várias ferramentas de *data mining*. Consiste na adoção de uma coleção de itens e, após particioná-los em pequenas coleções, conhecidos como *cluster*, baseando-se em alguma heurística que será usada para comparar com outros itens da coleção.

Técnicas para *clustering* são partes fundamentais para um processo de mineração de dados. Uma implementação simples de *clustering* pode criar experiências de usuário incrivelmente convincentes para alcançar resultados. Também pode ser aplicado a bases de texto utilizando algoritmos de *text mining*, onde o algoritmo procura agrupar textos que falem sobre o mesmo assunto e separar textos de conteúdo diferentes.

Por exemplo, caso queira considerar a localização geográfica dos contatos na rede *LinkedIn*, é necessário realizar um *cluster* nas conexões através de um determinado número de regiões com o intuito da melhor compreensão de oportunidades econômicas disponíveis.

4.2.2.2 Normalização de Dados

Quando se recupera dados provenientes de uma API, é muito comum que esses não estarão no formato desejado para a sua análise. No caso do *LinkedIn*, pode ser que usuários procurem por vagas de emprego escrevendo os nomes das vagas de alguma outra maneira sem ser o nome correto da vaga. Como exemplo, uma vaga

para administrador de banco de dados pode ser buscado através da sigla "DBA", que em inglês significa *Database Administrator*.

A normalização de dados, procura então resolver esses problemas padronizando situações específicas que irá facilitar a análise posterior dos dados.

4.2.2.3 Computação de Similaridade

Após a normalização dos itens, é preciso verificar a similaridade entre eles. Podendo ser vagas de emprego, nomes de empresas, interesses profissionais, indicação geográfica, ou qualquer outro campo digitado na busca como um texto livre. Para isso é necessário definir uma heurística que conseguirá aproximar a similaridade entre dois valores quaisquer. Em algumas situações a similaridade heurística será um tanto óbvia, porém em outros casos será complicada. Por exemplo, comparar o tempo de carreira entre duas pessoas pode ser simples como uma operação de soma ou subtração. Mas comparar um elemento profissional, como "atitude de liderança" de uma maneira automatizada pode ser um desafio.

5 IMPLEMENTAÇÃO DAS TÉCNICAS E ANÁLISE DOS RESULTADOS

5.1 INTRODUÇÃO

Este capítulo tem como finalidade apresentar com um maior nível de detalhamento as técnicas utilizadas neste trabalho, com o objetivo de se atingir as metas propostas já descritas no Capítulo 1, no item 1.2.

5.2 COLETA DE DADOS

Uma característica comentada anteriormente sobre a rede social *Twitter* é a disponibilidade de informações de eventos em tempo real. Os *tweets* podem ser postados com o intuito de comentar sobre a final de um campeonato de futebol, datas importantes, acontecimentos internacionais e políticos, entre outros.

Para este trabalho foi aproveitado o dia 17 de abril de 2016, onde foi realizado a votação do Congresso Brasileiro pela continuação do processo de Impeachment do cargo da presidenta Dilma Rouseff. Neste dia milhares de *tweets* foram publicados utilizando a *hashtag* "#ImpeachmentDay" com o objetivo de comentar sobre o evento de votação e, também, a atual situação política do Brasil.

Com o objetivo de coletar todos os dados que contenham a *hashtag* "#ImpeachmentDay", foi desenvolvido um *script* que utiliza o serviço de *Stream* do *Twitter* também conhecido como *FireHose*.

As primeiras linhas do *script* servem para importar as bibliotecas e objetos necessários para a utilização da API do *Twitter*, assim como as informações das chaves de acesso e *tokens* para o protocolo OAuth, que são fundamentais para a comunicação com o serviço de *Stream*.

```

1 from tweepy.streaming import StreamListener
2 from tweepy import OAuthHandler
3 from tweepy import Stream
4
5 access_token = "131556934-LrYRiXzAL3QcRyFN0fdN53EDWhNGfZFnVX59NCnT"
6 access_token_secret = "JraMtps51B98d8XoelAF71KHn8ZQ4nshdoSKiFlTz60Hd"
7 consumer_key = "P4XZ2GUkeqdhIlQM0redBuW05"
8 consumer_secret = "r5TPb2UcM8bzxq7t5zf1RPMHUrCfwNG4GRuVPXypowrpHhTmue"
9
10
11 class StdOutListener(StreamListener):
12

```

```
13     def on_data(self, data):
14         print data
15         return True
16
17     def on_error(self, status):
18         print status
19
20
21 if __name__ == '__main__':
22
23     l = StdOutListener()
24     auth = OAuthHandler(consumer_key, consumer_secret)
25     auth.set_access_token(access_token, access_token_secret)
26     stream = Stream(auth, l)
27
28     stream.filter(track=['ImpeachmentDay'])
```

Na linha 11 é criada uma classe que irá instanciar um serviço de escuta para este *Stream*. O serviço de escuta é responsável por observar todos os *tweets* que são publicados e após identificar no *tweet* alguma palavra ou a *hashtag* semelhante a palavra declarada na linha 28, é então capturado este *tweet* e impresso na tela de execução do *script*.

Quando este *script* está em execução é mostrado na tela todos os *tweets* que estão sendo coletados, já filtrados, porém não está sendo persistido em nenhum arquivo ou banco de dados. Com o intuito de salvar todos estes dados foi utilizado o comando ">", conhecido como *stdout*, presente em sistemas operacionais de base *Unix*. O comando ">" permite redirecionar a saída do código anterior, no caso a execução do *script* "coletar-hashtags.py", para um novo arquivo ou um arquivo já existente conforme ilustrado pela Figura 7.

```
scripts git:(master) x
> python coletar-hashtags.py > ../data/coleta-impeachment.json
```

FIGURA 7: Execução do *script* para coleta de dados

FONTE: Elaborado pelo autor

O *script* de coleta de dados permaneceu em execução durante 14 horas (das 08:30 às 22:30) do dia 17 de abril, concluindo em um arquivo de 2.6 gigabytes. O formato deste arquivo é um JSON onde apresenta todas as informações presentes em um *tweet*, demonstrado através do seguinte código:

```

1 {
2     "created_at": "Sun Apr 17 15:15:21 +0000 2016",
3     "id": 721718699418202112,
4     "id_str": "721718699418202112",
5     "text": "RT @GarotaCiume: N\u00e3o sou petista, s\u00f3 n\u00e3o sou
6     cega. Impeachment sem crime de responsabilidade \u00e9 golpe, e voc\u
7     \u00eas est\u00e3o traindo a democracia. #\u2026",
8     "source": "\u003ca href=\"http://twitter.com/download/android\"
9     rel=\"nofollow\"\u003eTwitter for Android\u003c/a\u003e",
10    "truncated": false,
11    "in_reply_to_status_id": null,
12    "in_reply_to_status_id_str": null,
13    "in_reply_to_user_id": null,
14    "in_reply_to_user_id_str": null,
15    "in_reply_to_screen_name": null,
16    "user": {
17        "id": 1876843824,
18        "id_str": "1876843824",
19        "name": "Millena Souza",
20        "screen_name": "Souzaa_mih1",
21        "location": "Brasil, Vit\u00f3ria- ES",
22        "url": "https://www.facebook.com/myllena.souzaa",
23        "description": "Brasileira 15 , paulista e capixaba, feminista,
24        Taurina e n\u00e3o sou obrigada a nada \u2764\ufe0f BORN TO DIE !!",
25        "protected": false,
26        "verified": false,
27        "followers_count": 433,
28        "friends_count": 431,
29        "listed_count": 0,
30        "favourites_count": 613,
31        "statuses_count": 28370,
32        "created_at": "Tue Sep 17 20:52:04 +0000 2013",
33        "utc_offset": -10800,
34        "time_zone": "Brasilia",
35        "geo_enabled": true,
36        "lang": "pt",
37        "contributors_enabled": false,
38        "is_translator": false,
39        "profile_background_color": "611994",
40        "profile_background_image_url": "http://pbs.twimg.com/pro
41        file_background_images/378800000076733988/60
42        df12d5296d8d8081e19c306c15c859.jpeg",
43        "profile_background_image_url_https": "https://pbs.twimg.com/pro
44        file_background_images/378800000076733988/60
45        df12d5296d8d8081e19c306c15c859.jpeg",
46        "profile_background_tile": true,
47        "profile_link_color": "6B25C7",

```

```

40     "profile_sidebar_border_color": "FFFFFF",
41     "profile_sidebar_fill_color": "7AC3EE",
42     "profile_text_color": "3D1957",
43     "profile_use_background_image": true,
44     "profile_image_url": "http://pbs.twimg.com/profile_images
    \699081483546443776/uCBDXAf0_normal.jpg",
45     "profile_image_url_https": "https://pbs.twimg.com/
    profile_images/699081483546443776/uCBDXAf0_normal.jpg",
46     "profile_banner_url": "https://pbs.twimg.com/profile_banners
    \1876843824/1455509034",
47     "default_profile": false,
48     "default_profile_image": false,
49     "following": null,
50     "follow_request_sent": null,
51     "notifications": null
52 },
53 "geo": null,
54 "coordinates": null,
55 "place": null,
56 "contributors": null,
57 "retweeted_status": {
58     "created_at": "Sun Apr 17 15:07:13 +0000 2016",
59     "id": 721716654569349121,
60     "id_str": "721716654569349121",
61     "text": "N\u00e3o sou petista, s\u00f3 n\u00e3o sou cega.
    Impeachment sem crime de responsabilidade \u00e9 golpe, e voc\u00eas
    est\u00e3o traindo a democracia. #ImpeachmentDay",
62     "source": "\u003ca href=\"http://twitter.com/download/iphone
    \" rel=\"nofollow\"\u003eTwitter for iPhone\u003c/a\u003e",
63     "truncated": false,
64     "in_reply_to_status_id": null,
65     "in_reply_to_status_id_str": null,
66     "in_reply_to_user_id": null,
67     "in_reply_to_user_id_str": null,
68     "in_reply_to_screen_name": null,
69     "user": {
70         "id": 206181302,
71         "id_str": "206181302",
72         "name": "Garota Ci\u00fame",
73         "screen_name": "GarotaCiume",
74         "location": "snapchat \/\ leuncosta \u2728",
75         "url": "http://Instagram.com/leuncosta",
76         "description": "Ciume \u00e9 igual bater o dedinho do p\u
    \u00e9 no sof\u00e1: coisa boba, mas d\u00e9 demais. \u2022 contato:
    contatogarotaciume@gmail.com",
77         "protected": false,
78         "verified": false,

```

```

79         "followers_count": 1202807,
80         "friends_count": 139,
81         "listed_count": 383,
82         "favourites_count": 40277,
83         "statuses_count": 25882,
84         "created_at": "Fri Oct 22 12:52:10 +0000 2010",
85         "utc_offset": -10800,
86         "time_zone": "Brasilia",
87         "geo_enabled": true,
88         "lang": "pt",
89         "contributors_enabled": false,
90         "is_translator": false,
91         "profile_background_color": "FFFFFF",
92         "profile_background_image_url": "http://pbs.twimg.com/
profile_background_images/879103282/2454
b176f3fffc4b100fb2bc64b1f2b5.png",
93         "profile_background_image_url_https": "https://pbs.twimg.
com/profile_background_images/879103282/2454
b176f3fffc4b100fb2bc64b1f2b5.png",
94         "profile_background_tile": false,
95         "profile_link_color": "383838",
96         "profile_sidebar_border_color": "FFFFFF",
97         "profile_sidebar_fill_color": "F6F6F6",
98         "profile_text_color": "000000",
99         "profile_use_background_image": true,
100        "profile_image_url": "http://pbs.twimg.com/profile_images
/3719404815/b868778f39b4b087561e3444b3093e20_normal.gif",
101        "profile_image_url_https": "https://pbs.twimg.com/
profile_images/3719404815/b868778f39b4b087561e3444b3093e20_normal.
gif",
102        "profile_banner_url": "https://pbs.twimg.com/
profile_banners/206181302/1403658723",
103        "default_profile": false,
104        "default_profile_image": false,
105        "following": null,
106        "follow_request_sent": null,
107        "notifications": null
108    },
109    "geo": null,
110    "coordinates": null,
111    "place": null,
112    "contributors": null,
113    "is_quote_status": false,
114    "retweet_count": 153,
115    "favorite_count": 130,
116    "entities": {
117        "hashtags": [{

```

```
118         "text": "ImpeachmentDay",
119         "indices": [121, 136]
120     ]],
121     "urls": [],
122     "user_mentions": [],
123     "symbols": []
124 },
125     "favorited": false,
126     "retweeted": false,
127     "filter_level": "low",
128     "lang": "pt"
129 },
130 "is_quote_status": false,
131 "retweet_count": 0,
132 "favorite_count": 0,
133 "entities": {
134     "hashtags": [{
135         "text": "ImpeachmentDay",
136         "indices": [139, 140]
137     }],
138     "urls": [],
139     "user_mentions": [{
140         "screen_name": "GarotaCiume",
141         "name": "Garota Ci\u00fame",
142         "id": 206181302,
143         "id_str": "206181302",
144         "indices": [3, 15]
145     }],
146     "symbols": []
147 },
148 "favorited": false,
149 "retweeted": false,
150 "filter_level": "low",
151 "lang": "pt",
152 "timestamp_ms": "1460906121483"
153 }
```

5.3 ANÁLISE DE DADOS

Após a coleta de dados foi gerado, então, um arquivo JSON de 2.6 gigabytes. Portanto a próxima etapa será analisar estes dados para extrair informações úteis.

Foi realizado testes no arquivo JSON coletado para verificar se existe algum tipo de *dirty data*, que são informações quebradas, dados irrelevantes ou códigos que impedem a execução de *scripts* de mineração (PIPINO; KOPCSO, 2004). A Figura 8 demonstra o único padrão de *dirty data* encontrado dentro do arquivo.


```
{"created_at":"Mon Apr 18 02:46:16 +0000 2016","id":721892  
{"created_at":"Mon Apr 18 02:46:16 +0000 2016","id":721892  
{"created_at":"Mon Apr 18 02:46:17 +0000 2016","id":721892  
{"limit":{"track":11138,"timestamp_ms":"1460947577319"}}  
{"created_at":"Mon Apr 18 02:46:17 +0000 2016","id":721892  
{"created_at":"Mon Apr 18 02:46:17 +0000 2016","id":721892  
{"created_at":"Mon Apr 18 02:46:17 +0000 2016","id":721892
```

FIGURA 8: *Dirty Data* presente no arquivo coletado

FONTE: Elaborado pelo autor

Dentro das milhares linhas do arquivo existem algumas linhas contendo este mesmo padrão de `{"limit":` que foram removidos utilizando outra ferramenta de sistemas baseados em Unix, o *grep*.

A ferramenta *grep* é um canivete suíço para o uso de expressões regulares e possui uma funcionalidade que permite realizar uma consulta inversa, ou seja, o usuário passa um padrão ou uma palavra que se deseja encontrar dentro de um determinado arquivo, e após encontrar todas as ocorrências o *grep* seleciona tudo o que não contém este padrão ou palavra.

Combinado com o comando de *stdout* (`>`), é possível redirecionar todos os dados que não possui este tipo de *dirty data* para um novo arquivo conforme a Figura 9.

```
data git:(master) x  
> grep --invert-match '{"limit":' coleta-impeachment.json > coleta-sem-dirty-data.json
```

FIGURA 9: Utilizando o comando *grep* para gerar um novo arquivo sem *dirty data*

FONTE: Elaborado pelo autor

Tendo então um arquivo JSON sem a presença de *dirty data* é possível iniciar o processo de extração de conhecimento utilizando o interpretador *IPython*.

Nesta primeira etapa importa-se as bibliotecas necessárias para se trabalhar com arquivos do tipo JSON e também para a geração de gráficos, conforme o código

a seguir:

```
1 %matplotlib inline
2
3 import json
4 import pandas as pd
5 import matplotlib.pyplot as plt
6
7
8 tweets_data_path = 'data/coleta-sem-dirty-data.json'
9
10 tweets_data = []
11 tweets_file = open(tweets_data_path, "r")
12 for line in tweets_file:
13     try:
14         tweet = json.loads(line)
15         tweets_data.append(tweet)
16     except:
17         continue
18
19 tweets = pd.DataFrame()
20 print len(tweets_data)
```

A primeira linha deste código permite a visualização de gráficos e planilhas no *IPython*. Na linha 8 em diante está sendo realizado a construção de um *DataFrame*, que é definido no Capítulo 4, seção 4.1.1.2, através leitura do arquivo "coleta-sem-dirty-data.json".

A execução deste código permite, então, gerar um *DataFrame* e informar o número de *tweets* que ele contém. Esses *DataFrames* possuem uma arquitetura semelhante ao formato de tabelas (linhas e colunas) em que é possível adicionar uma coluna ou uma tabela a uma variável em Python. Através do mapeamento de uma condição em um *DataFrame*, apresentado no código seguinte para mapear o texto, a língua e o país em que o *tweet* foi publicado.

```
1 tweets['text'] = map(lambda tweet: tweet['text'], tweets_data)
2 tweets['lang'] = map(lambda tweet: tweet['lang'], tweets_data)
3 tweets['country'] = map(lambda tweet: tweet['place']['country'],
4     if tweet['place'] != None else None, tweets_data)
5
6 tweets_by_lang = tweets['lang'].value_counts()
7
8 fig, ax = plt.subplots(figsize=(20,10))
9 ax.tick_params(axis='x', labels=20)
10 ax.tick_params(axis='y', labels=15)
```

```
11 ax.set_xlabel('Línguas'.decode('utf-8'), fontsize=20)
12 ax.set_ylabel('Número de tweets'.decode('utf-8'), fontsize=20)
13 ax.set_title('Top 4 Línguas'.decode('utf-8'),
14             fontsize=20, fontweight='bold')
15 tweets_by_lang[:4].plot(ax=ax, kind='bar', color='mediumspringgreen')
16 plt.grid()
```

Após o mapeamento das informações do *DataFrame* foi feito um gráfico para medir a quantidade de *tweets* em relação as 04 línguas mais *tweetadas*. A Figura 10 apresenta o gráfico gerado.

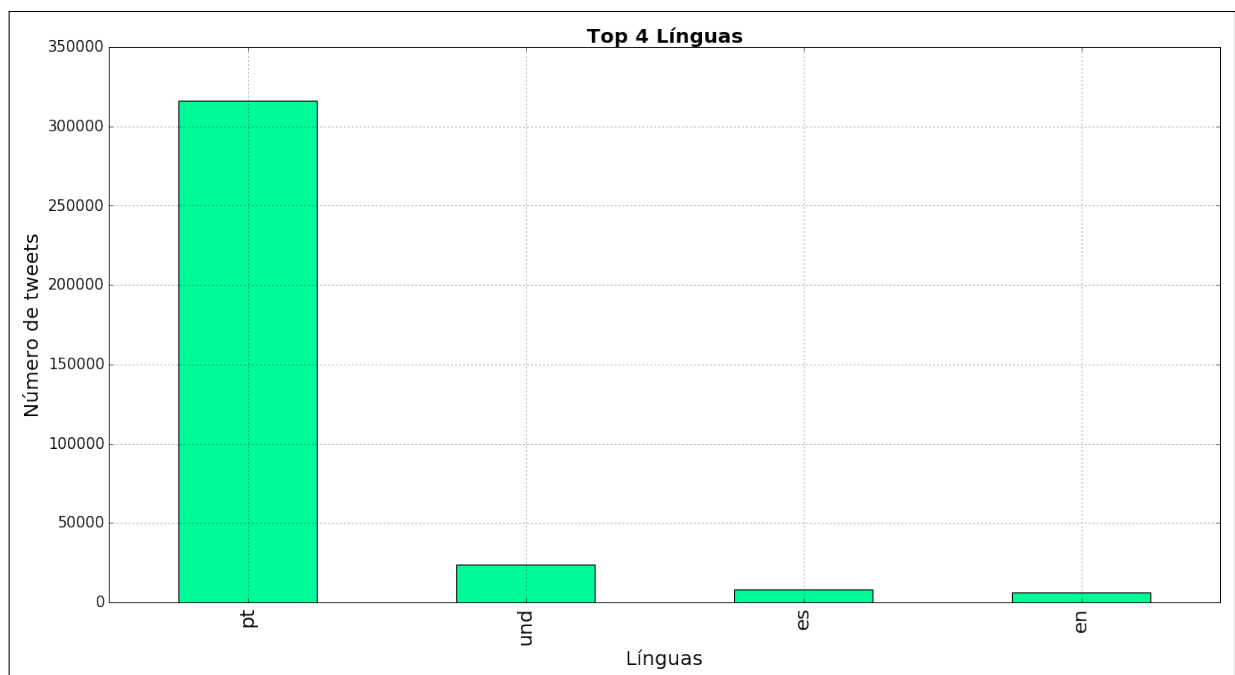


FIGURA 10: Línguas que mais realizaram *tweets*

FONTE: Elaborado pelo autor

É possível verificar que existe uma barra com o nome de *und* para especificar a segunda língua que realizou mais *tweets*. Essa é uma condição em que não foi identificada a língua de origem e então a API do *Twitter* classifica como *undefined*, ou indefinido no português. Essa linguagem indefinida ocorre devido ao *software* em que o usuário está realizando o *tweet*, por exemplo um navegador *web*, um aplicativo *mobile* do *Twitter* ou algum aplicativo de terceiro que permite realizar ações no *Twitter*. Caso a linguagem não está definida nestes ambientes a API o classifica como *undefined*.

Semelhante ao código anterior é possível verificar quais os países que estão comentando sobre a *hashtag* "#ImpeachmentDay".

```
1 tweets_by_country = tweets['country'].value_counts()
2
3 fig, ax = plt.subplots(figsize=(20,10))
4 ax.tick_params(axis='x', labels=15)
5 ax.tick_params(axis='y', labels=15)
6 ax.set_xlabel('Países'.decode('utf-8'), fontsize=20)
7 ax.set_ylabel('Numero de tweets'.decode('utf-8'), fontsize=20)
8 ax.set_title('Top 5 Países'.decode('utf-8'), fontsize=20, fontweight='
    bold')
9 tweets_by_country[:5].plot(ax=ax, kind='bar', color='lightskyblue')
10 plt.grid()
```

O código acima verifica quais são os 05 países que mais realizaram *tweets* através do mapeamento do *DataFrame* e então gera o gráfico representado na Figura 11.

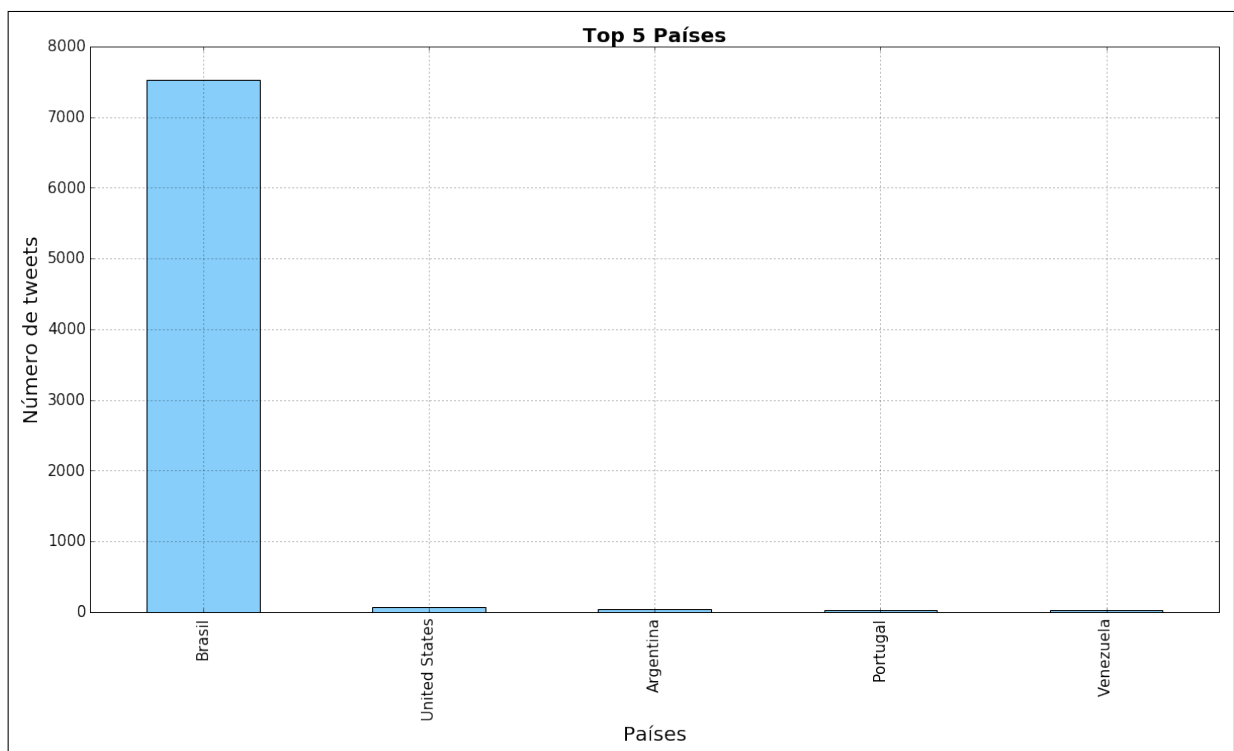


FIGURA 11: Países que mais realizaram *tweets*

FONTE: Elaborado pelo autor

Este gráfico demonstra claramente que a maior parte dos *tweets* foram publicados do Brasil e apenas uma pequena quantia deles foram realizados nos Estados Unidos, Argentina, Portugal e Venezuela. É importante notar que nem todos os *tweets* gerados possuem um país de origem semelhante a situação anterior onde a API do *Twitter* os classificam como *undefined*, porém não é apresentado neste gráfico.

6 CONCLUSÕES E SUGESTÕES PARA FUTUROS TRABALHOS

6.1 CONCLUSÕES

A utilização das bibliotecas que Python oferece para a mineração de dados permite que o desenvolvimento das soluções se tornem mais rápidos e efetivos. Assim como a utilização da API do *LinkedIn* para a obtenção dos dados e as etapas do processo de *data mining*. Boas práticas, protocolos e tecnologias são aproveitadas no andamento deste trabalho. Considerando os conceitos até aqui abordados, é possível compreender de forma clara os tópicos seguintes.

6.2 SUGESTÕES PARA FUTUROS TRABALHOS

REFERÊNCIAS

- BRACHMAN, R. J. et al. The process of knowledge discovery in databases. 1996. Acesso em 23 de outubro de 2015. Disponível em: <<https://www.aaai.org/Papers/Workshops/1994/WS-94-03/WS94-03-001.pdf>>.
- CODD, E. F. A relational model of data for large shared data banks. 1969. Acesso em 04 de outubro de 2015. Disponível em: <<https://www.seas.upenn.edu/~zives/03f/cis550/codd.pdf>>.
- CONWAY, D.; WHITE, J. M. **Machine Learning For Hackers**. [S.l.]: O'Reilly Media, 2012. ISBN 978-1-449-30371-6.
- FAYYAD, U. et al. From data mining to knowledge discovery in databases. 1996–a. Acesso em 23 de outubro de 2015. Disponível em: <<http://www.csd.uwo.ca/faculty/ling/cs435/fayyad.pdf>>.
- FAYYAD, U. et al. Advances in knowledge discovery in data mining. 1996–b.
- FIELDING, R. T. Architectural styles and the design of network-based software architectures. 2000.
- HAN, J. et al. **Data Mining: Concepts and Techniques**. [S.l.]: Elsevier, 2012.
- HARRINGTON, P. **Machine Learning in Action**. [S.l.]: Manning Publications, 2012. ISBN 9781617290183.
- JANSSENS, J. **Data Science at the Command Line**. [S.l.]: O'Reilly Media, 2014. ISBN 978-1-4919-4779-1.
- KALDERO, N. **Why is Python a language of choice for data scientists?** 2015. Acesso em 27 de outubro de 2015. Disponível em: <<http://qr.ae/RkleiB>>.
- LE MOS, E. P. **Análise de crédito bancário com o uso de data mining: redes neurais e árvores de decisão**. Tese (Doutorado) — Universidade Federal do Paraná, 2003.
- MCKINNEY, W. **Python for Data Analysis**. [S.l.]: O'Reilly, 2013.
- NAVEGA, S. Princípios essenciais do data mining. 2002.
- PIPINO, L.; KOPCSO, D. Data mining, dirty data, and costs. In: **IQ**. [S.l.: s.n.], 2004. p. 164–169.
- PYTHON. **Python 3.4.3 Documentation**. 2015. Acesso em 21 de novembro de 2015. Disponível em: <<https://docs.python.org/3.4/>>.
- RUSSELL, M. A. **Mining the Social Web**. [S.l.]: O'Reilly Media, 2013. ISBN 978-1-449-36761-9.
- SFERRA, H. H.; CORREA, A. M. C. J. Conceitos e aplicações de data mining. 2003.

SILVA, M. P. da; BOSCARIOLI, C.; PERES, S. M. Análise de logs da web por meio de técnicas de data mining. 2003.

STEINER, M. T. A. et al. Data-mining como suporte à tomada de decisões-uma aplicação no diagnóstico médico. **XXXVI SIMPÓSIO BRASILEIRO DE PESQUISA OPERACIONAL, "O IMPACTO DA PESQUISA OPERACIONAL NAS NOVAS TENDÊNCIAS MULTIDISCIPLINARES**, v. 23, p. 96–107, 2004.

TWEEPY, D. **Biblioteca tweepy**. 2009. Acesso em 03 de abril de 2016. Disponível em: <<http://tweepy.readthedocs.org/en/v3.5.0/index.html>>.

WEBBER, J.; PARASTATIDIS, S.; ROBINSON, I. **REST in Practice - Hypermedia and Systems Architecture**. [S.l.]: O'Reilly, 2010.

WIENER, E. Matplotlib tools. 2014. Acesso em 27 de novembro de 2015. Disponível em: <<https://wiki.ucar.edu/display/ral/Matplotlib+Tools>>.