FAA - FACULDADE ANGLO-AMERICANO

THIAGO MEDEIROS DE SOUZA

MINERAÇÃO DE DADOS DA REDE SOCIAL *LINKEDIN* UTILIZANDO A LINGUAGEM DE PROGRAMAÇÃO PYTHON E BIBLIOTECAS PARA ANÁLISE E MINERAÇÃO DE DADOS

FOZ DO IGUAÇU

2015

THIAGO MEDEIROS DE SOUZA

MINERAÇÃO DE DADOS DA REDE SOCIAL *LINKEDIN* UTILIZANDO A LINGUAGEM DE PROGRAMAÇÃO PYTHON E BIBLIOTECAS PARA ANÁLISE E MINERAÇÃO DE DADOS

Trabalho de conclusão de curso apresentado como requisito obrigatório para obtenção do título de Bacharel em Ciência da Computação da Faculdade Anglo-Americano - FAA.

Orientador: Prof. Msc. Valmei Abreu Júnior

Coorientador: Prof. Esp. João Paulo de Lima Barbosa

FOZ DO IGUAÇU

2015

TERMO DE APROVAÇÃO

THIAGO MEDEIROS DE SOUZA

MINERAÇÃO DE DADOS DA REDE SOCIAL *LINKEDIN* UTILIZANDO A LINGUAGEM DE PROGRAMAÇÃO PYTHON E BIBLIOTECAS PARA ANÁLISE E MINERAÇÃO DE DADOS

Trabalho de conclusão de curso apresentado como requisito obrigatório para obtenção do título de Bacharel em Ciência da Computação da Faculdade Anglo-Americano - FAA, 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, 23 de novembro de 2015

A todos os que tentam a mais de meia década se graduar,
Eu consegui!

AGRADECIMENTOS

Agradecimentos para serem agradecidos...

"Apenas que... Busquem conhecimento."
(E.T. Bilu)

RESUMO

Resumo do meu trabalho...

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

ABSTRACT

Here goes my abstract....

Keywords: Data Mining. Python. Linkedin. Data.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

KDD	*Knowledge Discovery From Data* - Descoberta de Conhecimento por Dados

Sumário

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, ajudou 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.

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 *data mining* é usado por companhias com grande foco em varejo, finanças, comunicação e marketing. Empresas conseguem 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.

A análise de dados consiste em visualizar informações em diferentes maneiras e formas, plotando gráficos e planilhas. Neste momento, novas informações aparecerão permitindo alguma previsão ou predição desse conteúdo.

Essas 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.

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 desde o armazenamento dos dados em estudos e passa por processos matemáticos, estatísticos e computacionais, com o objetivo de extrair informação útil. Um modelo, então, é muito mais que apenas a descrição dos dados, modelo incorpora o entendimento de todo o processo da origem dos dados até a competência deles. Logo esse modelo consegue fazer previsões sobre os conhecimentos analisados.

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 conseguirá obter possíveis resultados através do reconhecimento desses dados.

Dados é um termo, deliberadamente vago, que agrega várias formas comuns de dados, 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 ser analisados e modelados.

Cientistas de dados precisam visualizar os dados 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.

Para a análise e a interação de dados, computação exploratória e visualização de dados, a linguagem de programação Python vai, inevitavalmente, 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.

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++. O que se percebe é 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, dinâmicamente 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

LinkedIn é uma rede social que possui foco em relacionamento profissional e negócios. De um modo ilustrativo, *LinkedIn* se parece com um evento privado, onde os participantes possuem normas para vestimenta e estão extremamente bem vestidos, tentando demonstrar os valores específicos e expertises que possuem para trazer ao mercado profissional. Usuários desta rede estão principalmente interessados em oportunidades de negócios que este meio provê como uma forma arbitrária de socialização e irá, necessariamente, trazer detalhes sobre relações, históricos profissionais e outros.

Hoje, o *LinkedIn* possui mais de 400 milhões de usuários e um crescimento de 40%, a cada ano, de novos integrantes. Existem mais de 3 milhões de páginas de companhias e empresas, permitindo a conexão de 225 milhões de profissionais em

sua rede. A cada dia, 200 conversas acontecem por minuto. Essas informações foram realizadas à dois anos atrás e demonstram um pouco do tamanho de informações que esta rede social possui (LINKEDIN, 2015).

A mineração de dados do *LinkedIn* ajuda no conhecimento de fortes conexões, novas habilidades a serem adquiridas, na comparação entre perfis e a busca para um cargo em específico, assim como a busca a uma companhia específica.

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

1.2 OBJETIVOS

1.2.1 Objetivo Geral

Este trabalho tem como objetivo principal desenvolver soluções e algoritmos utilizando a linguagem Python e algumas de suas bibliotecas para analisar a rede social *LinkedIn*, tendo como foco a predição de um perfil de um profissional de tecnologia da informação, a verificação de fortes conexões e construção de um perfil mais influente na rede.

1.2.2 Objetivos Específicos

- Apresentar a API (interface de programação de aplicações) do *LinkedIn*;
- Apresentar a linguagem Python e as bibliotecas que ela possui para a análise de dados;
- Estudo e manipulação de APIs de redes sociais e seus métodos de autenticação;
- Implementar algoritmos para a mineração de dados;
- Gerar gráficos e planilhas para a interpretação dos dados;
- Apresentar os testes e resultados obtidos com o desenvolvimento de algoritmos.

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 semanas 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 de Documentação: 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.

Semanas	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
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	X	X
Testes		X	X	X	X	X	X	X	X
Elaboração de Artigos					X			X	X
Apresentação de Resultados					X				X

Tabela 1: Cronograma de execução

1.4 ORGANIZAÇÃO DO TRABALHO

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

No capítulo 2 é apresentado as citações de trabalhos que foram referências bibliográficas 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 de Python utilizadas para a mineração de dados é exposta no capítulo 4. Também é apresentado a API do *LinkedIn*, etapas de *data mining* e outros materiais e metodologias utilizados para execução deste trabalho.

No capítulo 5 é demonstrado 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 do trabalho se dá no capítulo 7, sendo nesse abordado e analisado as dificuldades, além de determinadas as possibilidade para trabalhos posteriores.

2 Revisão Bibliográfica

2.1

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 quantia 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, na Seção 3.2, é 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 3.3. A Seção 3.4 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 como uma sequência interativa e iterativa dos seguintes passos:

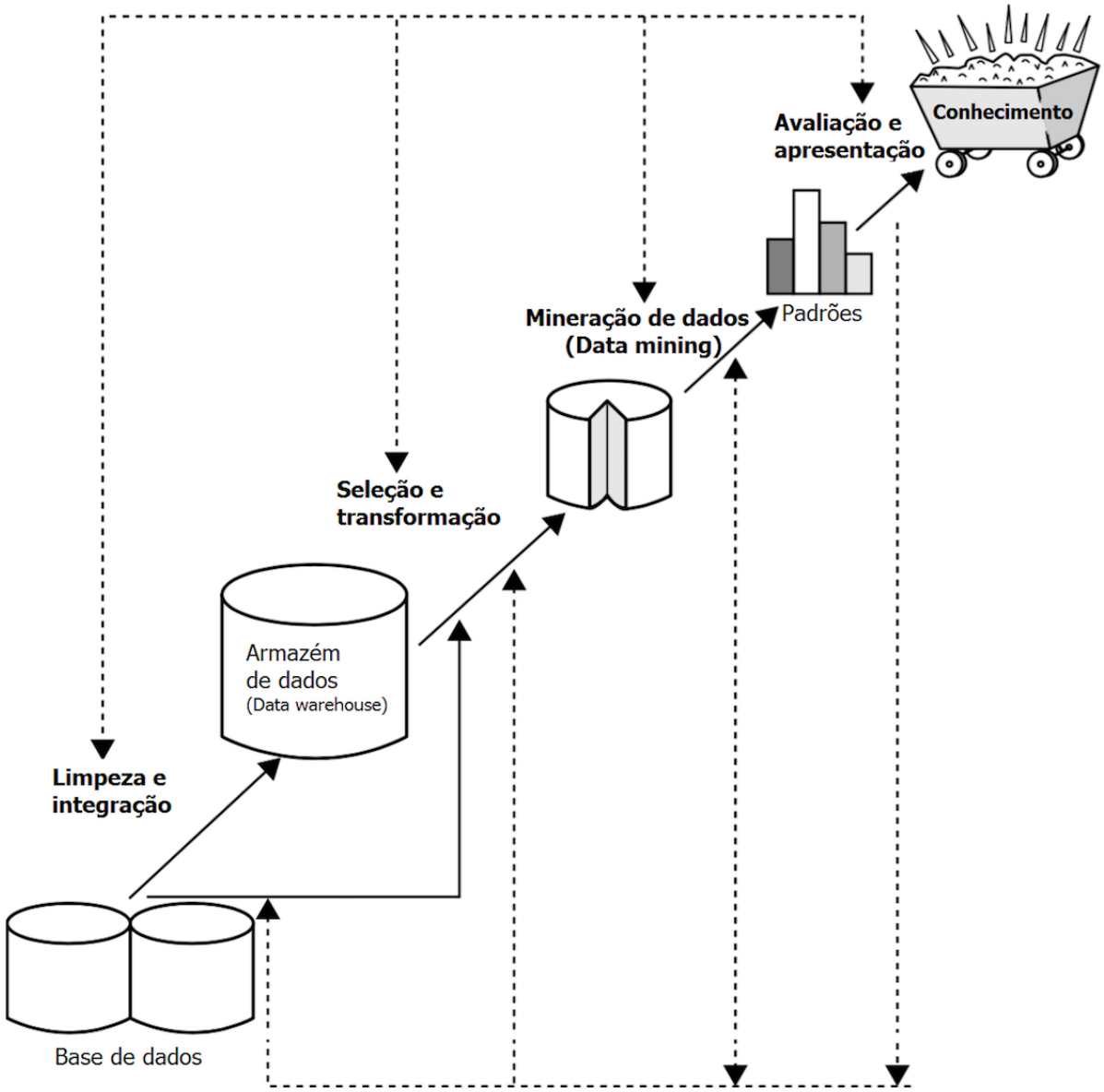

Figura 1: Etapas do processo de KDD

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).

De acordo com Brachman et al. (1996 apud FAYYAD et al., 1996a), 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 aplicação de algoritmos específicos para extrair padrões (modelos) de dados. Os passos adicionais no processo KDD, como integração de dados, limpeza de dados, seleção de dados, incorporação de conhecimento anterior apropriado e interpretação formal dos resultados de mineração assegura aquele conhecimento útil que é derivado dos dados. A aplicação cega de métodos de *data mining* pode ser uma atividade perigosa que conduz a descoberta de padrões sem sentido (NAVEGA, 2002).

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).

3.2 *Data Mining*

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* poderá 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. (1996b) "...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 extração de informações, sem conhecimento prévio, de algum conjunto de dados e seu uso para tomada de decisões. A mineração de dados se define através de processos automatizados de captura e análise deste conjunto de dados com a finalidade de extrair algum significado, podendo descrever características do passado, como também para predizer futuras tendências (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 sequênciais, como dados com dependência no tempo. Tem como objetivo modelar o estado do processo extraindo 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., 1996b) 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.

3.3 *Machine Learning*

3.4 Python

Python é uma linguagem de programação orientada a objetos, interpretada, interativa. Incorpora módulos, excessões, tipagem dinâmica alta. Possuindo uma sintaxe

clara e simples, o que facilita o aprendizado para novos desenvolvedores, assim como a rápida leitura e interpretação para usuários mais experientes. A linguagem 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 de 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.

Python é uma linguagem de programação de alto-nível que pode ser aplicada em soluções para diversas classes diferentes. A linguagem 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 é apresentada 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*, *SciPy* trazem recursos de outras ferramentas como MATLAB e R. *matplotlib* e *Seaborn* são mecanismos que possibilitam a visualização dos dados.

Dictionaries em Python são estruturas de dados 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 é apresentado as ferramentas e tecnologias que serão utilizadas durante o estudo. Serão abordados quais as bibliotecas que o Python disponibiliza para a análise e mineração de dados e também como acessar a API do *LinkedIn*, e essa 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 do Python

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

Em seguida, 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*.

4.1.1.1 *NumPy*

NumPy é o pacote fundamental para computação científica. É o acrônico 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 *pandas*

pandas é a biblioteca de maior interesse para 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 *Serie* 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 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 *matplotlib*

matplotlib é uma biblioteca desenvolvida 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), *matplotlib* possui recursos interativos como zoom e visão panorâmica. Esta biblioteca su-

porta várias ferramentas GUI *backend*, nos diversos sistemas operacionais suportados 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 em tridimensionais e *basemap* para mapeamentos e projeções.

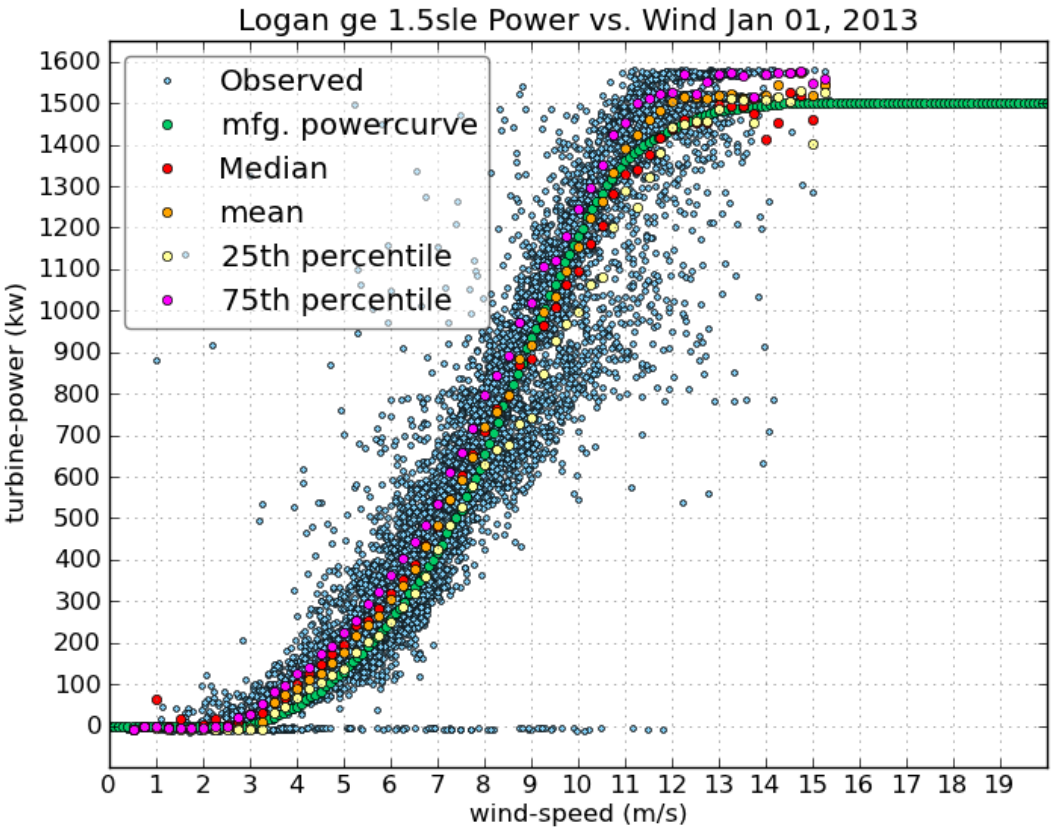

Figura 5: Exemplo de um gráfico gerado pelo *matplotlib*

FONTE: Wiener (2014)

4.1.1.4 *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 raíz;
- *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 *IPython*

IPython foi desenvolvido com o intuito de ser um interpretador interativo para o Python que teve início em 2001. 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 produtivadade 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 o sistema de arquivos de sistema operacional. Como a maior parte da programação focada na mineração de dados envolve exploração, tentativa e erro, e iteração, *IPython*, em quase todos os casos, irá facilitar este tipo de trabalho (MCKINNEY, 2013).

Hoje, o projeto *IPython* 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).

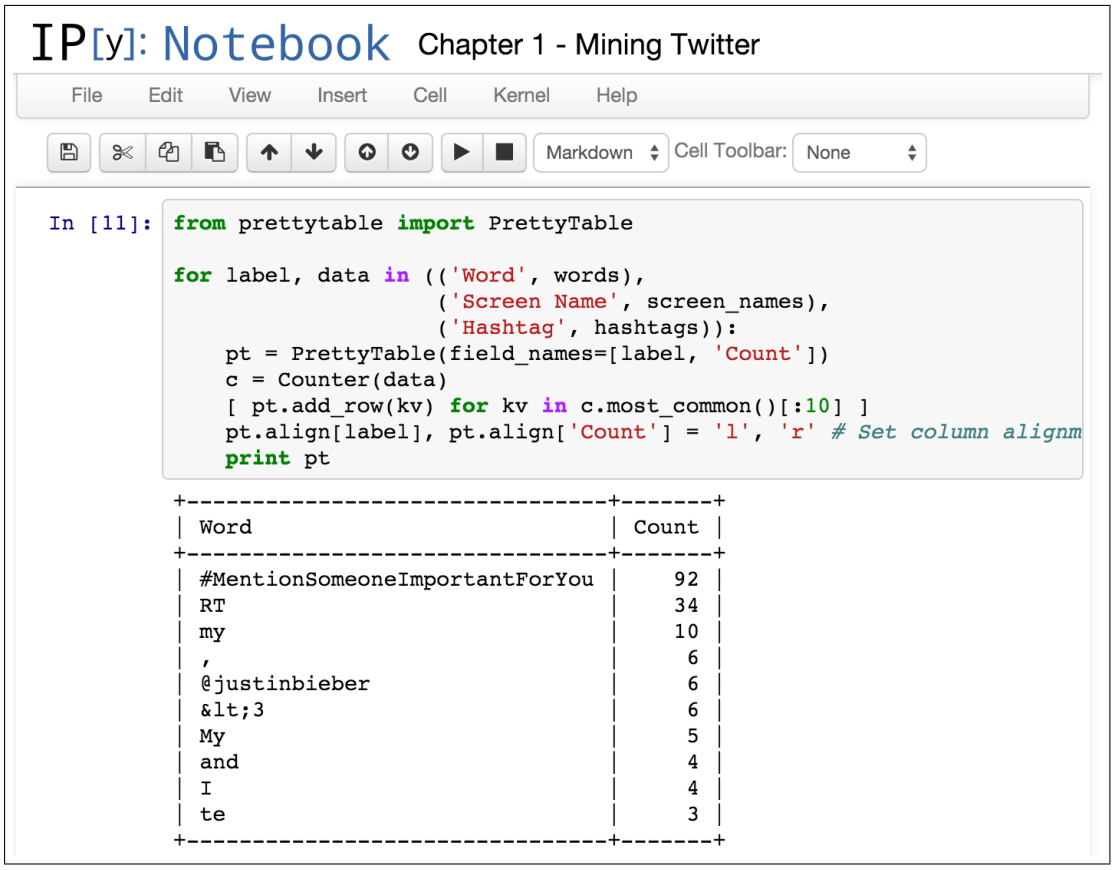

Figura 6: Exemplo de uma página *web* do *IPython Notebook*

FONTE: McKinney (2013)

4.1.1.6 *python-linkedin*

Esta biblioteca provê ao Python a API do *LinkedIn*. Através da utilização do protocolo OAuth 2.0, é possível acessar diveros campos como *Profile*, *Group*, *Company*, *Jobs*, *Search*, *Share*, *Network* e requisições REST APIs (VATANSEVER, 2015).

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

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.

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 implantação 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. O 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).

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, torna ele 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. Uma representação 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, YML, 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 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. RFCs são documentos técnicos desenvolvidos e mantidos pelo Internet Enginnering 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 SSL através de um protocolo HTTPS, por esse motivo muitos desenvolvedores achavam tedioso o trabalho devido aos vários detalhes envolvidos em encriptação.

SSL vem de *Secure Socket Layer*, e 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 protegidas 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, três participações 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* obtêm 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 um 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 OAuth 2.0

Enquanto o protocolo OAuth 1.0a permite uma autorização útil para o acesso a aplicações *web*, OAuth 2.0 foi originalmente destinado a simplificar significantemente a implementação detalhada para desenvolvedores de aplicações *web*, baseando-se completamente no SSL para aspectos de segurança e satisfazer uma vasta quantidade de casos de uso. Esses casos de uso variaram desde suporte para dispositivos móveis à necessidades empresariais e, consequentemente as 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 Linkedin

4.1.4.1 API LinkedIn

4.2 Metodologia de 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.

4.2.1 Etapas do Data Mining

4.3 Considerações Finais

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>.

FAYYAD, U. et al. Advances in knowledge discovery in data mining. 1996.

FAYYAD, U. et al. From data mining to knowledge discovery in databases. 1996. Acesso em 23 de outubro de 2015. Disponível em: <http://www.csd.uwo.ca/faculty/ling/cs435/fayyad.pdf>.

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.

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/RkIeiB>.

LEMOS, E. P. Análise de crédito bancário com o uso de data mining: Redes neurais e Árvores de decisão. 2003.

LINKEDIN. *About LinkedIn*. 2015. Acesso em 21 de novembro de 2015. Disponível em: <https://press.linkedin.com/about-linkedin>.

MCKINNEY, W. *Python for Data Analysis*. [S.l.]: O'Reilly, 2013.

NAVEGA, S. Princípios essenciais do data mining. 2002.

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.

VATANSEVER, O. *Biblioteca python-linkedin*. 2015. Acesso em 18 de novembro de 2015. Disponível em: <https://github.com/ozgur/python-linkedin>.

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>.