Universidade Paulista - UNIP

Gabriel Leite

Implementação de Big Data com Hadoop: Um Guia Prático

Limeira

2020

Universidade Paulista - UNIP

Gabriel Leite

Implementação de Big Data com Hadoop: Um Guia Prático

Trabalho de conclusão de curso apresentado à banca examinadora da Faculdade UNIP, como requisito parcial à obtenção do Bacharelado em Ciência da Computação sob a orientação do professor Me. Sergio Eduardo Nunes.

Limeira

2020

Gabriel Leite

Implementação de Big Data com Hadoop: Um Guia Prático

Trabalho de conclusão de curso apresentado à banca examinadora da Faculdade UNIP, como requisito parcial à obtenção do Bacharelado em Ciência da Computação sob a orientação do professor Me. Sergio Eduardo Nunes.

Aprovada em XX de XXXXX de 202X.

BANCA EXAMINADORA

_____

Prof. Dr. Nome completo

_____

Prof. Me. Nome completo

_____

Prof. Esp. Nome completo

DEDICATÓRIA

Dedico esse trabalho a todos que acreditaram em mim e me ajudaram nessa incrível jornada que foi não apenas o TCC mas também a graduação, a todas as incríveis pessoas que conheci durante esse período e a minha família que sempre esteve ao meu lado, mesmo na dificuldade.

“Sem dados você é apenas outra pessoa com uma opinião”.

(Willian Edwards)

RESUMO

Esse trabalho, de maneira extremamente resumida, tem como sua função, de certa maneira ensinar alguns princípios básicos do campo de Data Science, fornecendo uma base onde o leitor, a partir desse ponto, possa de maneira independente aprender e adquirir conhecimento, pois uma das principais dificuldades hoje, nessa área que tem crescido muito na última década, é a acessibilidade devido a pouca documentação e material de estudo em PT-BR, além disso, existe a falta de um “mapa” ao qual a pessoa deve seguir para não ficar perdida sem saber qual é o próximo passo na jornada de aprendizado. E por isso, no documento, é traçado um caminho que vai desde o início do processo de Data Science com a mineração de dados até o processamento desses dados usando a ferramenta Hadoop, incluindo linguagens utilizadas, algoritmos, bibliotecas, comandos na linha de comando, configuração e por fim, a instalação. Isso tudo de maneira que seja de fácil compreensão.

Palavra-Chave: Hadoop; Data Science; Mapa; Guia.

ABSTRACT

This paper, at an extremely short form, has the function of somehow teach some basic principles of data science, to provide a foundation of the subject to the reader, so he can, at that point onwards, start to learn and to acquire knowledge in and independent form, because one of the worst difficulties today, is the lack of data science documentation and study material in Portuguese with easy access, also, not only that, but the lack of a guide to provide the user a path so he can follow to now get lost on that journey of learning. And because of that, on this document, a path is laid down, from the very start of the data science process to the very end, using tools and frameworks like Apache Hadoop and also programming languages, APIs, command lines and even the installation. All that meant to be easily understandable.

Key Words: Hadoop; Data Science; Roadmap; Guide.

LISTA DE FIGURAS

Figura 1 - Hadoop Yarn 20
Figura 2 - Atualizando o pip 23
Figura 3 - Instalando o Tweepy 23
Figura 4 - Instalando o dotenv 24
Figura 5 - Criando o app na API do Twitter 24
Figura 6 - Chaves da API do Twitter 25
Figura 7 - Gerando o token de acesso 26
Figura 8 - Tokens de acesso 26
Figura 9 - Importando as bibliotecas no python 27
Figura 10 - Stopwords.txt 28
Figura 11 - arquivo .env 28
Figura 12 - Abrindo o stopwords.txt e separando as palavras 29
Figura 13 - Carregando o .env 29
Figura 14 - Adquirindo as chaves do .Env 29
Figura 15 - Adiciona as chaves no objeto de autenticação do tweepy 30
Figura 16 - Classe básica de streamListener 30
Figura 17 - Chama a classe e o método de streaming 30
Figura 18 - Função concluída 31
Figura 19 - Script rodando e coletando tweets em tempo real 32
Figura 20 – Exemplo dos resultados 32
Figura 21 - Pesquisa pelo editor de variaveis do sistema 33
Figura 22 - Janela de propriedades do sistema 34
Figura 23 - Criando uma variável para o Java 35
Figura 24 - Adicionando o Hadoop e o Java a variavel de caminho 35
Figura 25 - Verificando a funcionalidade do Java 36
Figura 26 - Instalando os utilitários do Windows 36
Figura 27 - Diretório dos arquivos de configuração 37
Figura 28 - Core-site.xml 37
Figura 29 - hdfs-site.xml 37
Figura 30 - mapred.xml 38
Figura 31 - Yarn-site.xml 38
Figura 32 - Formatando o namenode 38

Figura 33 - Alterando o jar problemático por um funcional 39
Figura 34 - Formatação concluída com sucesso 39
Figura 35 - Selecionando o arquivo 39
Figura 36 - Colando no diretório correto 40
Figura 37- Iniciando o serviço 40
Figura 38 - Namenode e datanode funcionando 41
Figura 39 - Comando para iniciar o Yarn 41
Figura 40 - start-all em ação 41
Figura 41 - painel de controle do Hadoop 42
Figura 42 - Painel do cluster 42
Figura 43 - Código do map 43
Figura 44 - Código do reducer 44
Figura 45 - mkdir no hadoop 44
Figura 46 - copyFromLocal 44
Figura 47 - ls mostrando o conteudo da pasta data 45
Figura 48 - Executando o mapreduce 45
Figura 49 - Listando os serviços 45
Figura 50 - "Matando" um serviço 46
Figura 51 - Resultados no painel de controle 46
Figura 52 - Comando para exibir o resultado 46
Figura 53 - resultado 47

LISTA DE ABREVIATURAS

HDFS – Hadoop Distributed File System

CSV – Comma Separated Values (Valores separados por ponto e vírgula)

JDK – Java SE Development Kit

VM – Virtual Machine (máquina virtual)

API – Application programming interface (interface de programação de aplicativos)

SUMÁRIO

1. INTRODUÇÃO 13

1.1. OBJETIVO 14

1.2. JUSTIFICATIVA 14

1.3. METODOLOGIA 15

2. ENGENHARIA DE DADOS 16

2.1. PYTHON 16

2.1.1. CARACTERÍSTICAS 16

2.2. MINERAÇÃO DE DADOS 17

2.3. TWEEPY 17

2.4. StreamListener 17

3. CIÊNCIA DE DADOS 18

3.1. HADOOP 18

3.1.1. MAPREDUCE 21

4. PROCESSOS EXPERIMENTAIS 22

4.1. DATA MINING 22

4.1.1. PYTHON 22

4.1.2. TWITTER API 24

4.1.3. TWEEPY 27

4.2. DATA SCIENCE 33

4.2.1. HADOOP 33

4.2.1.1. REQUISITOS 33

4.2.1.2. CONFIGURAÇÃO 36

4.2.2. MAPREDUCE 43

4.2.2.1. MAP 43

4.2.2.2. REDUCE 43

4.2.3. COMANDOS ÚTEIS 44
4.3. RESULTADOS 46
CONCLUSÃO 48
REFERÊNCIAS BIBLIOGRÁFICAS 49

1. INTRODUÇÃO

Com a tecnologia moderna, a quantidade de dados e informações que passaram ser possíveis armazenar aumentou drasticamente, não se limitando ao meio físico e sim ao digital, e com o surgimento da internet, o número de aplicações que gera e coleta informações também teve um grande aumento, pois, nos dias atuais, tudo gera dados, desde formulários até notas fiscais, e esses dados podem ser se vitais para mostrar o desempenho da empresa.

Entretanto, a coleta desses dados, a preparação, armazenamento e processamento para informação, não é tarefa fácil, requerendo muita habilidade por parte do engenheiro de dados para coletar esses dados sem perder nem alterar nada, e também na parte de processamento, onde o cientista de dados, é responsável por processar esses dados preparados pelo engenheiro e transformá-los em algo apresentável. Dito isso, o ramo de ciência de dados, desde a primeira década do século 21, vem ascendendo a ponto de hoje, em 2020, o engenheiro de dados, ser um dos campos com mais vagas disponíveis no mercado de trabalho, basta uma rápida pesquisa em sites que oferecem vagas de emprego para ver a ampla variedade de empresas contratando profissionais da área.

Essa alta demanda pode ser ligada a colossal quantidade de dados gerados diariamente na internet, o Big Data, que hoje, é essencial para grandes empresas tentando sobreviver nesse ecossistema da era da informação, tendo em vista que tudo acontece muito rapidamente e então necessita-se de alguém que possa se responsabilizar por adquirir tais dados e os transformar em conjuntos de dados, e também um profissional que possa usar tais conjuntos para gerar uma informação tangível e útil que podem ser desde relatório de como a publicidade ou até mesmo as vendas estão indo naquele momento.

Porém, mesmo com a alta demanda, ainda faltam profissionais da área da ciência de dados, e a razão disso, pode ser a escassez de documentação de qualidade e de fácil acesso, e principalmente, em português, e por isso, esse trabalho procura mostrar um caminho, oferecendo uma base onde a pessoa pode aprender o básico da ciência de dados como um todo, desde a mineração de dados, até o processamento final.

1.1. OBJETIVO

De maneira ampla, o objetivo desenvolver um material que pode servir como um guia acerca de Data Science, iniciando desde o mais básico, que é a instalação e configuração do Python e as bibliotecas necessárias, e se entendendo até o processamento de arquivos minerados na internet os processando.

Como dito anteriormente nessa sessão, o objetivo do trabalho não é se aprofundar em nenhuma tecnologia em específico, mas sim, uma visão ampla da área, para facilitar o aprendizado e preencher essa falta de documentação em português.

1.2. JUSTIFICATIVA

A razão a qual esse trabalho foi escrito, como citado acima, é para fornecer uma base de conhecimento ao qual o leitor possa a partir daí, de maneira independente, seguir o seu próprio caminho, estudando tecnologias que se relacionam a partes mais especificas do campo de ciência de dados, ou também se aprofundar no que foi oferecido pelo trabalho.

Pois, existe uma falta de documentação a respeito desses assuntos na internet, especialmente em português, tendo em vista que boa parte da população brasileira não é falante nem possui capacidade de compreender o inglês, o processo de aprendizado pode ser intimidador para alguém novo, pois há também a falta de um “mapa” ao qual o estudante possa seguir, podendo facilmente não saber o que fazer ou qual tecnologia usar para progredir.

Portanto, esse trabalho foi feito na tentativa de ajudar a amenizar esse problema de aprendizado, que leva a falta de profissionais na área, devido a escassa, se não, inexistente documentação em português e também a ausência de um material de referência para auxiliar na progressão e prover um caminho ao qual o usuário pode seguir.

1.3. METODOLOGIA

A primeira etapa do trabalho, consiste na introdução do leitor as tecnologias as quais serão utilizadas durante todo o progresso do aprendizado, a razão disso, é para apresentar o funcionamento das ferramentas em sua parte técnica, o que pode facilitar o entendimento de seu funcionamento também na parte prática.

Em seguida, se tem início a segunda etapa, onde é demonstrado o processo de engenharia de dados, a qual dados são coletados da rede social Twitter, além de arquivos externos, com o objetivo de se introduzir o leitor a várias maneiras de se coletar dados, e também em seguida tratá-los para remover o que não é de utilidade e armazenar o que é útil para o cientista de dados.

E então a etapa final, onde é apresentado o Hadoop, que hoje serve de base para muitos dos sistemas de processamento de Big Bata, como Apache Spark, Hive, Pig e vários outros presentes no mercado. O processo visa ensinar a fazer todo o procedimento de instalação, desde os requerimentos, a definição das variáveis até o final do processo, que é o momento em que se gera uma informação útil a partir de dados que foram coletados na segunda etapa.

2. ENGENHARIA DE DADOS

Engenharia de Dados, em 2020, é uma das profissões que mais possui vagas abertas no mercado de trabalho, pois a ciência de dados é algo vital para uma empresa que quer sobreviver na era da informação, tendo em vista que toda pessoa, toda parte de uma companhia gera dados, os mesmos mostram de tudo, desde as funções de uma empresa, o lucro, opinião pública, publicidade, parcerias, vendas e bens, mas se esses dados não são vistos, não há como obter informações.

O Engenheiro de Dados é o responsável por acessar todos esses dados, desde vários arquivos em PDF, posts em redes sociais, planilhas no Excel, banco de dados relacionais e não relacionais e diversas outras fontes, além de também movê-los e armazená-los sem adulteração e de maneira eficiente.

2.1. PYTHON

O Python por sua vez, foi inicialmente concebido na década de 80 por Guido van Rossun no '*Centrun Wiskunde & Informatica'* na Holanda, com sua primeira implementação acontecendo em 1989, Van Rossun, foi o desenvolvedor líder, até 12 de Julho de 2018, quando ele anunciou as suas "férias permanentes" como "Ditador benevolente" da linguagem.

Hoje, no final de 2020, Python é uma das linguagens mais populares no mercado, chegando até a passar o Java em questão de popularidade, e apenas atrás de Java Script.

2.1.1. CARACTERÍSTICAS

O Python, é uma linguagem de alto nível e multiparadigma, ou seja, suportando vários meios para se programar, além disso, é amplamente utilizada na área acadêmica, Machine Learning e Data Science, a grande vantagem de Python é sua acessibilidade devido a sua sintaxe extremamente simples e tipagem forte, além da vasta quantidade de bibliotecas disponíveis e também uma gigante comunidade.

Os princípios de Python, como dito pelo engenheiro de software por Tim Peters em ‘Zen of Python’, podem se resumir a:

> “Beautiful is better than ugly.
> Explicit is better than implicit.
> Simple is better than complex.
> Complex is better than complicated.
> Readability counts.”

Isso tudo culmina em uma linguagem perfeita para a ciência de dados, pois ela permite a integração de APIs e sistemas com muita facilidade, devido a sua simplicidade, mesmo ficando um pouco atrás em performance comparado a linguagens compiladas como C# e Java.

2.2. MINERAÇÃO DE DADOS

A mineração de dados é uma das principais tarefas do engenheiro de dados, que como dito anteriormente, é o responsável por pegar dados de várias fontes, desde PDFs, Planilhas, Relatórios até feeds de redes sociais como Twitter. Para minerar, são desenvolvidos scripts, normalmente em Python devido a sua simplicidade e facilidade com bibliotecas e APIs integradas, como por exemplo o Tweepy, que é uma biblioteca que faz a ligação com a API do Twitter, e com ela é possível coletar posts feitos por usuários, retweets (compartilhamentos) e comentários em postagens.

2.3. TWEEPY

Como citado anteriormente, com o Tweepy, é possível fazer a integração da API do Twitter no script Python, sendo assim possível até mesmo pegar as postagens em tempo real com base em palavras chaves definidas pelo Engenheiro de Dados, e após coletá-las, as modelar da forma que for melhor para o Cientista de Dados, como por exemplo, usando um algoritmo para pegar as hashtags do post e salvá-los para um documento ou banco de dados externo, que a partir daí, pode ser usado pelo cientista de dados.

2.4. StreamListener

O StreamListener, é responsável pela mineração em tempo real dos dados, ao criar uma classe de StreamListener, é possível criar um algoritmo que, por

exemplo, seja possível pegar apenas as hashtags no tweet, além disso, ao chamar a classe, é possível utilizar vários filtros, que podem filtrar por postagens de um usuário especifico ou palavras chaves, mas esses filtros também se aplicam para outras funções da API.

E diferente de sua alternativa, o REST API que faz uma requisição para pegar os dados, o StreamListener do Tweepy cria uma sessão persistente que permite a coleta de mais dados em tempo real de que a REST jamais conseguiria**.**

3. CIÊNCIA DE DADOS

A ciência de dados, é um vasto campo interdisciplinar, pois são vários profissionais que podem atuar na área, como por exemplo, um cientista de Machine Learning, que usa das informações para alimentar uma inteligência artificial com Machine Learning, ou seja, alimentando conhecimento para a IA, temos também o Consultor de Dados, que fica com a tarefa de identificar a melhor maneira de utilizar as informações geradas pelo analista de dados, que é o profissional que trabalha com os datasets fornecidos pelo Engenheiro de Dados para gerar uma informação.

E nos dias de hoje, um Analista de Dados, junto ao Engenheiro de Dados, é um dos profissionais mais procurados, pois ele é essencial para as grandes empresas, pois ele vai ser o responsável por desenvolver soluções para processar os dados e gerar relatórios essenciais.

3.1. HADOOP

O Hadoop é uma biblioteca de software, que permite ao Analista de Dados processar enormes quantidades de informações, podendo fazer isso não apenas na máquina local, mas também em um cluster de computadores.

Além disso, o Hadoop, possui vários outros projetos que o utilizam, como por exemplo o Spark, que é muito utilizado quando se trata de aplicações com Machine Learning, tendo também a interação com bases de dados não relacionais como Hbase que foi feita pela própria Apache Software Foundation para uso em conjunto ao Hadoop, Cassandra, MongoDB e vários outros bancos

de dados não relacionais. É importante também dizer, que o Hadoop é desenvolvido em Java.

O Hadoop pode ser dividido em 4 partes ou módulos:

- **Hadoop common**

A base do software, que é usado como requisito para os outros módulos

- **Hadoop Distributed File System (HDFS)**

Sistema distribuído de dados, que possui alta capacidade de volume de dados da aplicação e embora possua algumas similaridades com alguns outros sistemas de arquivo distribuído, o HDFS possui uma grande diferença. O sistema é muito tolerante a falhas, e é desenvolvido para possibilitar a aplicação em máquinas de baixo custo, além de ser extremamente portátil podendo facilmente ser transferido de uma plataforma a outra.

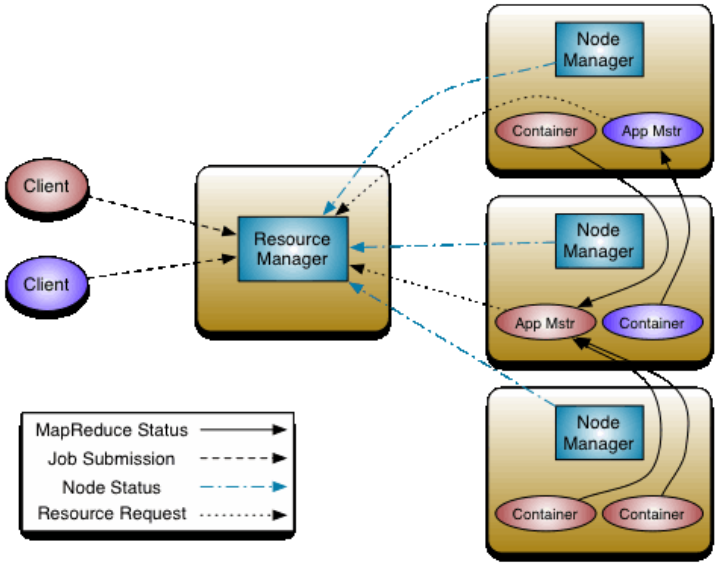
A arquitetura HDFS é composta por uma estrutura master e slave, possuindo um Namenode como o mestre e responsável por regular todo o acesso dos arquivos aos clientes, além também do Namespace, existem também os Datanodes, que normalmente é apenas um por nó no Cluster, ele é o responsável por administrar o armazenamento nos nós aos quais eles estão atribuídos, em adição a isso, também administra blocos.

De maneira geral, o HDFS permite o usuário a armazenar os dados em arquivos, e esses arquivos, são por sua vez, divididos em blocos são armazenados nos Datanodes.

- **Hadoop YARN**

O Framework YARN tem como principal ideia, dividir todo o gerenciamento de recursos e Jobs em daemons separados, a ideia é existir um gerenciador global, e um local, para cada aplicação.

Figura 1 - Hadoop Yarn



Fonte: Website do Hadoop

De maneira ampla, o YARN virtualiza todas a aplicações e funciona de maneira similar a aplicações de virtualização como por exemplo o Docker.

- **Hadoop Mapreduce**

Um Sistema baseado em Yarn para o processamento de altas quantidades de dados de maneira paralela e em grandes clusters, isso tudo com grande tolerância a falha.

O Mapreduce pode ser dividido em 2, o input dos dados, onde tais dados são divididos em partes independentes, que por sua vez são processadas pelo Map, o output do processamento, é então enviado para o Reduce, onde ele realiza diversas operações dependendo do que o usuário programou. De maneira geral, o Map vai separar os dados e o Reduce tornar esses dados intermediários em algo tangível.

3.1.1. MAPREDUCE

Como dito na sessão anterior, o Mapreduce é o responsável pelo processamento de todo dado pelo Hadoop, e ele permite fazer isso de maneira paralela, tolerante a falha, em grandes quantidades com Datasets de vários terabytes e em grandes clusters com centenas de Datanodes.

É importante dizer que mesmo o Hadoop sendo uma aplicação feita em Java, o Mapreduce não necessita ser feito em Java, mas também podendo ser escritas em Ruby, C++ e Python.

Como dito, o Mapreduce é dividido em duas partes, o Map e o Reduce, eles têm funções diferentes, o Map, inicialmente divide o Dataset em várias partes que são então enviadas para suas tarefas de Map individuais, após isso, é então criado o output de dados intermediários de acordo com a função do map. De certo modo é como se o Map recebesse uma lista, ele pode separar essa lista, e em cada tarefa onde esses blocos separados estão, é executado a função do Map, sendo assim, a maneira que o Map foi utilizada no projeto feito para esse documento, para separar as palavras e as deixar independentes, com isso, o Map então, cria o output e então entra na parte do Reduce.

O Reduce por sua vez, é o que vai agregar esses dados individuais e de maneira similar ao Map, vai executar uma função para isso, por exemplo, contando a quantidade de vezes ao qual a palavra se repete, ou sortindo a lista por ordem alfabética ou numérica.

4. PROCESSOS EXPERIMENTAIS

Nessa etapa, vai ser explicado os procedimentos de instalação, configuração e uso de cada uma das tecnologias utilizadas no processo de Data Science, com objetivo de identificar quais as hashtags no Twitter mais populares relacionadas aos consoles de vídeo game de nova geração, para isso, vai ser utilizado um script feito em Python para minerar dados da plataforma, que em seguida, vão ser preparados e armazenados e enviados para o Hadoop, onde será realizada a função de contagem de palavras.

4.1. DATA MINING

Como citado acima, o nessa parte, vai ser realizada a parte prática do trabalho, e para realizar o processo de Data Science, é necessário primeiramente obter os dados necessários, essa é a parte de Engenharia de Dados, e tais dados, podem ser retirados de inúmeros lugares diferentes, entretanto, vai ser realizado um Web Scraping, que nada mais é, uma técnica para se coletar dados de um site, que nesse caso, é a rede social Twitter.

O que vai ser buscado, são postagens a respeito dos consoles de vídeo game da nova geração, o Xbox series x e series s, e também o Playstation 5, para isso vamos usar da API da plataforma, juntamente com uma biblioteca no Python e com isso, através de palavras chaves definidas em um arquivo separado, vamos filtrar e coletar todas as postagens contendo aquelas palavras.

Após isso, vamos separar as hashtags e armazená-los de um modo que seja possível o seu eventual processamento, com o objetivo final, observar qual das duas plataformas foi mais discutida durante o período observado.

4.1.1. PYTHON

Primeiramente, é necessário a instalação do Python, essa parte não tem muitos segredos, bastando visitar o website oficial e logo na página inicial já é possível encontrar o botão para fazer o download da versão mais recente. Já no Linux, em boa parte das distribuições, o Python já vem instalado por padrão.

Feito isso, agora é necessário a instalação de uma IDE para o desenvolvimento, para quem não é familiar, uma IDE ou Integrated Development

Environment, ou no português, Ambiente de desenvolvimento integrado, é, de maneira simples, um software que reúne em um único lugar um conjunto de ferramentas com o intuito de agilizar o processo de desenvolvimento.

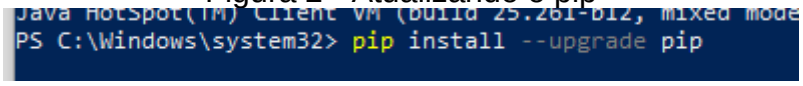
A escolha pessoal para o trabalho foi a IDE PyCharm, desenvolvida pela JetBrains, mas isso pode ser escolhido de maneira subjetiva, sempre procurando o programa mais familiar.

Com a IDE de preferência instalada, agora vem uma parte importante, a instalação das bibliotecas necessárias, não são muitas, e nem um processo complicado, e para isso, vai ser utilizado o pip, o instalador de pacotes do Python, ele já vem instalado por padrão nas versões mais recentes do Python.

Para realizar a instalação de tais bibliotecas, primeiro é necessário a execução da linha de comando, no caso do Linux, seu terminal, e no caso do Windows, foi utilizado o Shell, para abrir o Shell, é bem simples, basta utilizar a barra de pesquisa do sistema, que apenas pressionando a tecla Windows, já permite a digitação, após isso, basta procurar “Power Shell” e o executar em modo administrador.

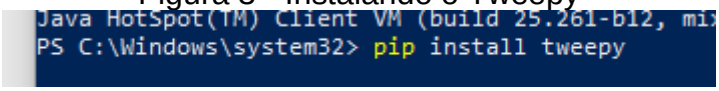
Feito isso, a primeira coisa a ser feita, é a atualização do pip, para isso, basta executar o seguinte comando no prompt:

Figura 2 - Atualizando o pip



A razão disso, é garantir que o instalador vai possuir todas as bibliotecas necessárias e atualizadas, agora pode-se instalar o necessário, iniciando com o Tweepy**:**

Figura 3 - Instalando o Tweepy



O Tweepy, vai ser o responsável por interagir com a API do Twitter, que é de onde vão ser retirados os dados. Em seguida, dotenv:

Figura 4 - Instalando o dotenv

```
Java HotSpot(TM) Client VM (build 25.261-b12, mixed
PS C:\Windows\system32> pip install python-dotenv
```

O dotenv, é utilizado para armazenar as chaves de acesso a API, é uma ótima maneira de organizar múltiplas chaves e deixar o código organizado, limpo, além de permitir o compartilhamento sem a necessidade de remover nada.

4.1.2. TWITTER API

Com tudo instalado, agora é necessário adquirir as chaves de acesso da API, isso é relativamente fácil de fazer, basta entrar no web site de desenvolvedores do Twitter. Disponível em: https://developer.twitter.com/ Acesso em 14 de set. de 2020.

Para conseguir o acesso, basta entrar em sua conta do Twitter, e se cadastrar na plataforma de desenvolvedores. Após o cadastro, vai ser necessário a criação de um app, caso seja o primeiro acesso, o site já vai dar início a criação.

Figura 5 - Criando o app na API do Twitter

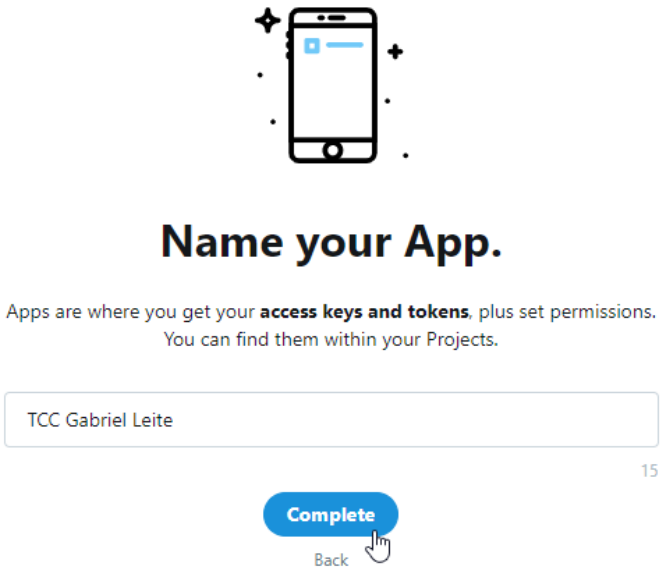
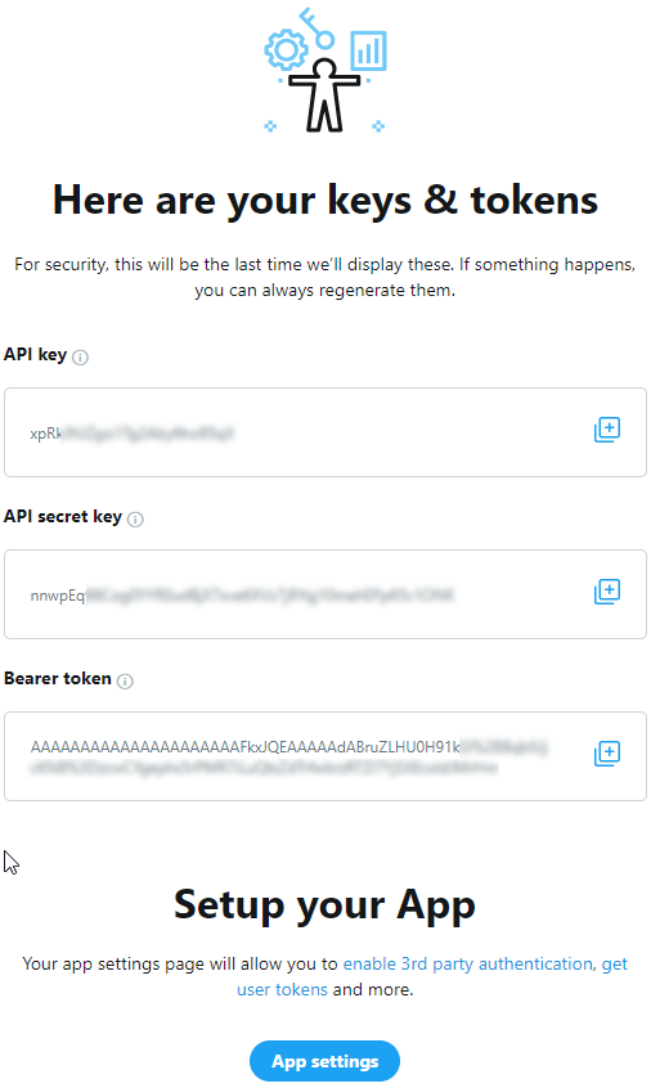


Figura 6 - Chaves da API do Twitter



Feito isso, as chaves da API vão ser geradas, mas ainda faltam duas chaves, e para consegui-las, é bem simples, basta clicar em “App settings”, e na tela do app, ir na aba “Keys and Tokens”. Nessa tela, vai ser possível gerar as chaves de acesso que são necessárias. Clicando em “Generate” para o “Access Token & Secret”, as chaves restantes vão ser geradas e uma janela com elas vai ser aberta.

Figura 7 - Gerando o token de acesso

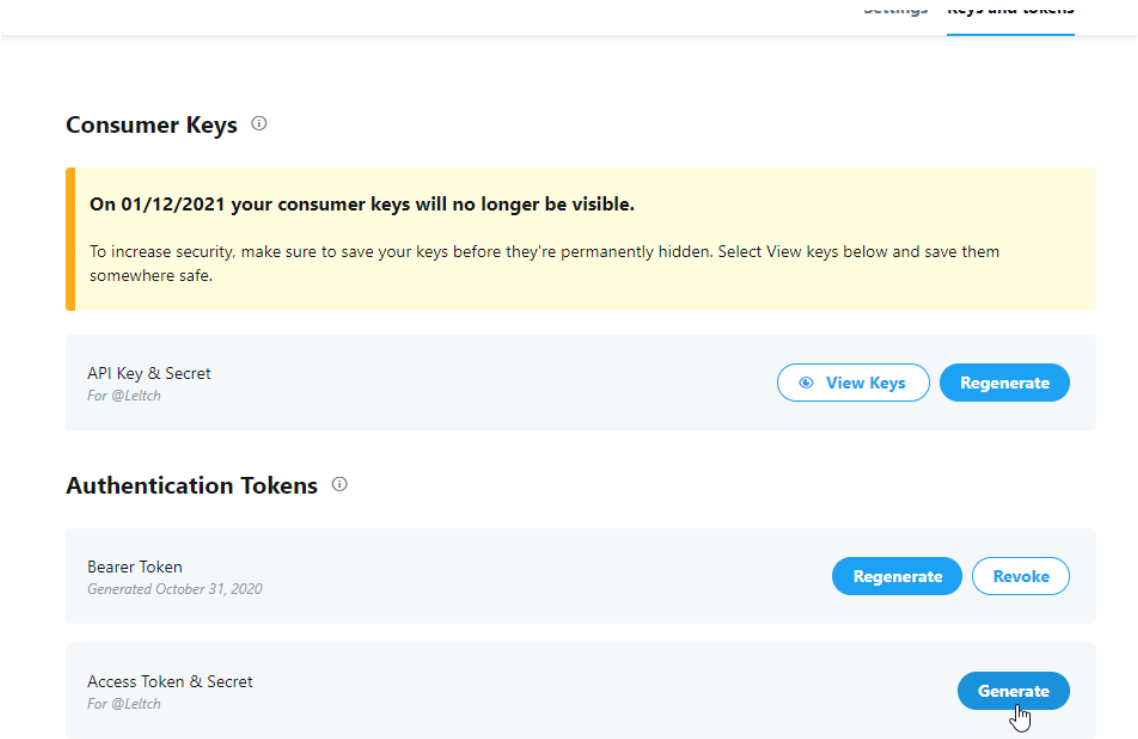
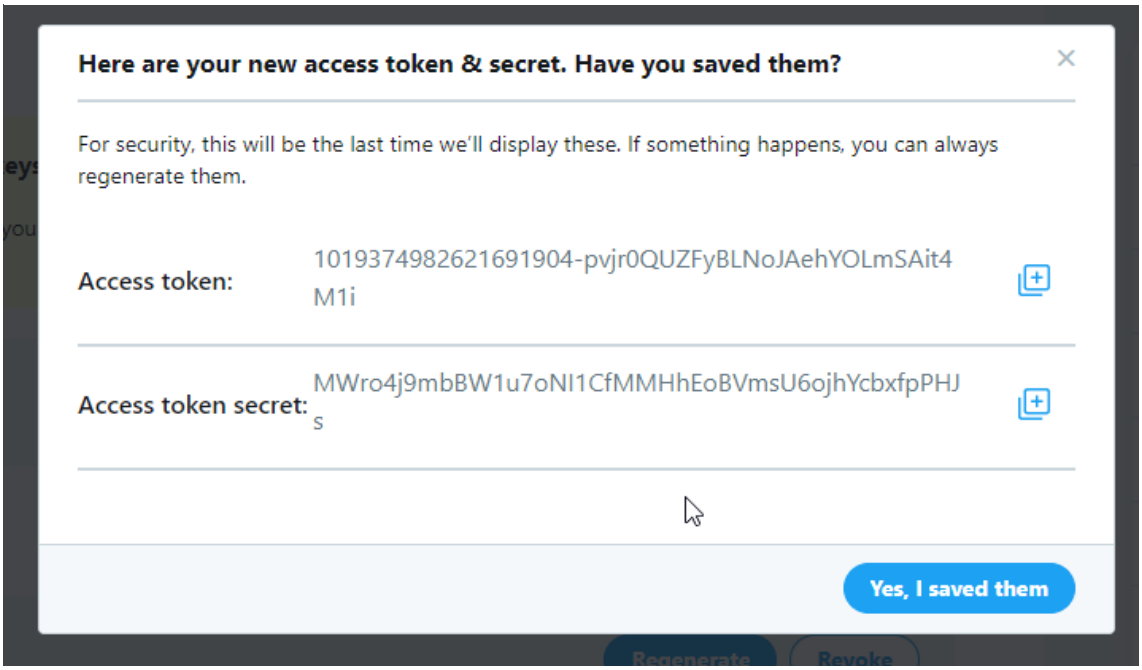


Figura 8 - Tokens de acesso



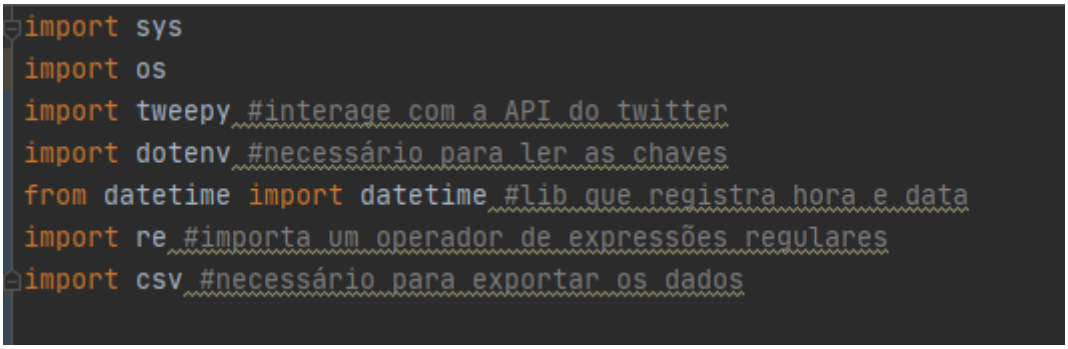
Com as chaves em mãos, agora é possível se autenticar no Twitter através de sua API e até mesmo publicar tweets através de scripts programados.

4.1.3. TWEEPY

E por fim, essa é a hora que se é colocado a mão na massa, com tudo preparado, basta criar um projeto na IDE, lembrando que é de extrema importância que a instância correta do Python seja selecionada na criação.

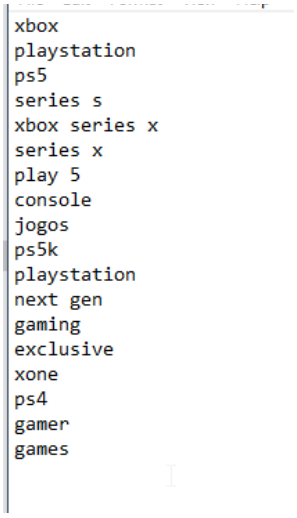
Com o projeto criado, a primeira coisa a ser feita, sem dúvida é *print "hello world!"*. Após verificar que o Python está de fato funcionando e já ter dado as boas-vindas ao ambiente, pode ser iniciar a importação das bibliotecas.

Figura 9 - Importando as bibliotecas no python



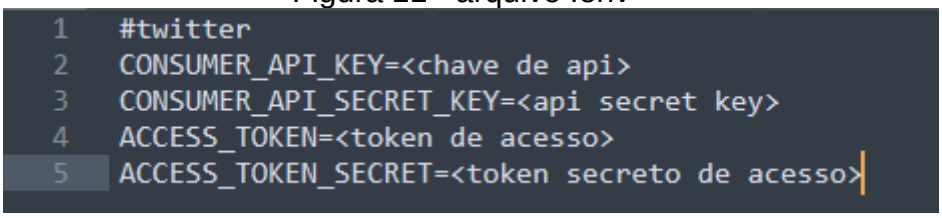
Com as bibliotecas importadas, a primeira coisa a ser feita, é na pasta raiz do projeto, um arquivo de texto do bloco de notas, ou txt, onde as palavras chaves vão ser inseridas, essas palavras, preferencialmente tem relação ao assunto ao qual se deseja minerar sobre, esse arquivo, deve se chamar "stopwords.txt".

Figura 10 - Stopwords.txt



Após a criação do arquivo de texto, agora é necessário criar um arquivo chamado “.env”, que vai ser onde as chaves vão ser armazenadas, para isso, basta abrir o bloco de notas, e em seu interior, são criados os campos onde as chaves são inseridas.

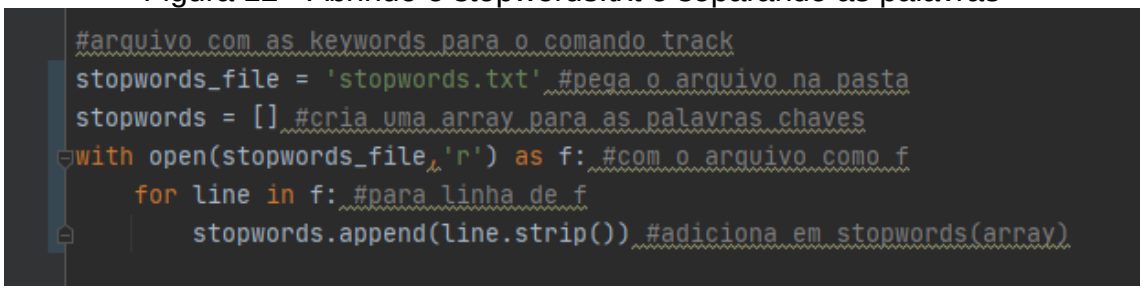
Figura 11 - arquivo .env



Na figura acima, é demonstrado um exemplo de como foi estruturado o arquivo, lembrando que o nome definido para cada chave é o mesmo que vai ser utilizado mais tarde ao realizar a autenticação, e entre maior e menor, são onde as chaves serão colocadas.

Feito isso, agora é possível começar a escrever o script, a primeira coisa a ser feita após importar todas as dependências, é abrir o arquivo ‘stopwords.txt’ no script e pegar as palavras chaves para serem usadas na busca. Para isso, o arquivo vai ser separado linha por linha, e cada linha é destinada a uma posição de um array.

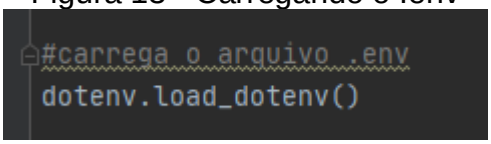
Figura 12 - Abrindo o stopwords.txt e separando as palavras



Após isso, o programa vai ser capaz de pegar tudo do arquivo de texto e usar as palavras no momento da busca pelos tweets.

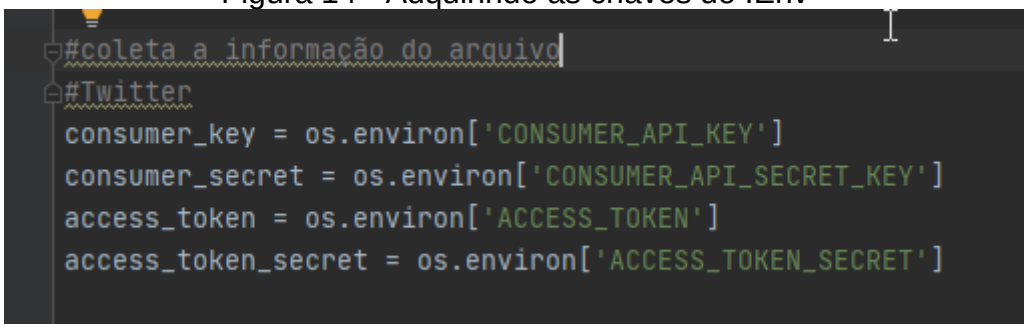
A próxima parte, é realizar a autenticação da API do Twitter, para isso, é preciso abrir o arquivo dotenv no script e coletar as informações dentro, o que não é difícil, basta apenas uma linha de código para abrir o arquivo.

Figura 13 - Carregando o .env



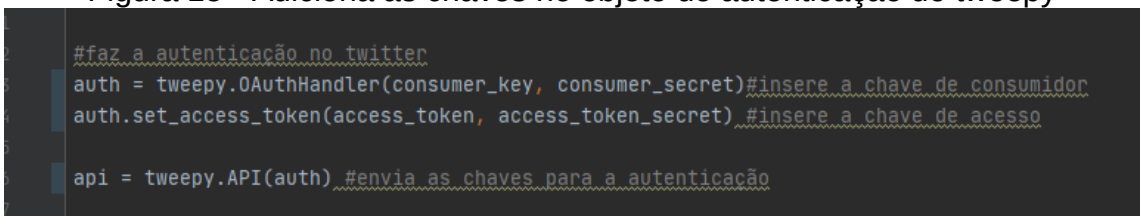
Com o arquivo carregado, agora é possível pegar as informações de autenticação. Como citado a pouco, vai ser utilizado o nome das chaves de acordo com o que foi definido no arquivo '.env', ou seja, caso sua chave de API foi escrita como "API_KEY", no código deve ser chamada com esse mesmo nome.

Figura 14 - Adquirindo as chaves do .Env



Na imagem acima, os nomes são iguais aos nomes definidos no arquivo .env, e assim vão coletar as chaves corretamente, e em seguida inserir dentro de uma variável, que vai ser inserida na variável 'auth' do método 'api' que eventualmente vai ser usado para poder ganhar acesso ao servidor para coletar dados.

Figura 15 - Adiciona as chaves no objeto de autenticação do tweepy



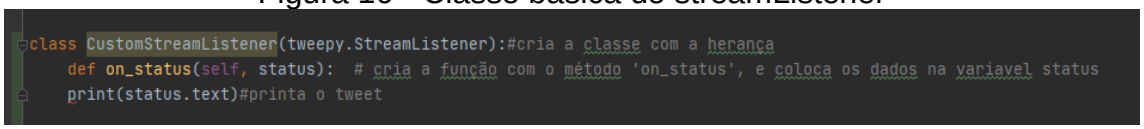
```
#faz a autenticação no twitter
auth = tweepy.OAuthHandler(consumer_key, consumer_secret)#insere a chave de consumidor
auth.set_access_token(access_token, access_token_secret) #insere a chave de acesso

api = tweepy.API(auth) #envia as chaves para a autenticação
```

Com isso feito, agora pode-se dar início a criação da função de coleta de tweets, é possível coletar posts de seu feed, coletar informações do usuário e pegar em tempo real os tweets da plataforma. Nesse caso, serão pegos os posts em tempo real, e dentro da função receberão tratamento e em seguida, serão armazenados dentro de um arquivo de extensão csv.

Agora entra a parte mais complexa, o momento em que é criado a classe que contém a função, essa função cria um StreamListener, usando alguns métodos do Tweepy. O primeiro passo é criar a classe com a herança do StreamListener, e dentro dela, a função com o método ‘on_status’, que vai receber os dados do listener.

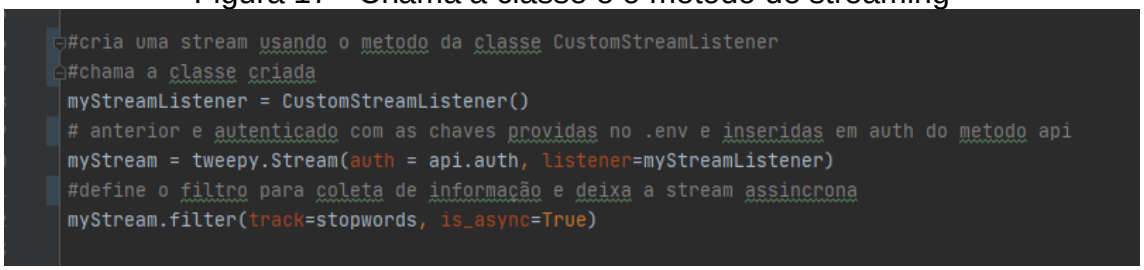
Figura 16 - Classe básica de streamListener



```
class CustomStreamListener(tweepy.StreamListener):#cria a classe com a herança
    def on_status(self, status):  # cria a função com o método 'on_status', e coloca os dados na variavel status
    print(status.text)#printa o tweet
```

Com a classe feita, é necessário criar a stream, para isso, é feito uma variável que vai requisitar a classe e em seguida é realizado a autenticação e se tem início a stream. Entretanto, se for executado dessa maneira, nada vai ser coletado, a razão disso é que é necessário aplicar o filtro, o que não é difícil, bastando apenas usar o método ‘filter’.

Figura 17 - Chama a classe e o método de streaming



```
#cria uma stream usando o metodo da classe CustomStreamListener
#chama a classe criada
myStreamListener = CustomStreamListener()
# anterior e autenticado com as chaves providas no .env e inseridas em auth do metodo api
myStream = tweepy.Stream(auth = api.auth, listener=myStreamListener)
#define o filtro para coleta de informação e deixa a stream assincrona
myStream.filter(track=stopwords, is_async=True)
```

Se executado, o script vai printar em tempo real todos os tweets que contêm as palavras chaves que estão dentro do arquivo ‘stopwords’, entretanto, ainda é

necessário tratar esses tweets e armazená-los de maneira que facilite o processamento desses dados eventualmente.

O primeiro passo para o tratamento é definir que dado vai ser necessário se obter, nesse caso, o objetivo vai ser coletar hashtags relacionadas aos console, o Xbox e Playstation 5, o filtro já está coletando os tweets corretos, agora basta tratar e armazenar.

Dentro do método ‘on_status’ na classe, é trocado as quebras de linha por espaço, e em seguida, as strings são separadas em palavras e são identificadas quais palavras se iniciam com o símbolo da hashtag(#), e em seguida é normalizado tudo para que se possa ser armazenado dentro de um arquivo CSV. E após isso tudo, ainda é necessário criar o tratamento de erros da stream, para caso de desconexão ou problema no serviço do twitter.

Figura 18 - Função concluída

```
class CustomStreamListener(tweepy.StreamListener):#cria a classe com a herança
    def on_status(self, status):  # cria a função com o método 'on_status', e coloca os dados na variavel status
        words = status.text.replace('\n', ' ')  # troca as quebras de linha por um espaço em branco
        words = words.split(' ')  # realiza a separação dessas linhas
        hashtags = [w for w in words if '#' in w]  # verifica se tem # no inicio da palavra
        hashtags = [re.sub(r'[^\w\s]', '', h) for h in hashtags]  # pega quais tem # e normaliza
        for h in hashtags:#para todas as hashtags
            # vai abrir o arquivo console.csv, definir o encoder dele pra utf-8
            with open('consoles.csv', 'a', encoding='utf-8', errors='replace') as xx:
                writer = csv.writer(xx)#define o arquivo como csv para o writer
                writer.writerow([h.encode('utf-8')])#insere no arquivo a hashtag
            print(status.text)#printa o tweet no console ou terminal

        def on_error(self, status_code):#em caso de erro
            print(status_code)
            return True  # Don't kill the stream

        def on_timeout(self):#em caso de problema de conexão
            print('timeout')
            return True  # Don't kill the stream

        def on_exception(self, exception):#demais erros
            print(exception)
```

Com tudo isso concluído, agora se executado, o programa vai coletar todas as hashtags que existirem dentro dos tweets que forem pegos e os armazenar dentro do arquivo “consoles.csv”.

Figura 19 - Script rodando e coletando tweets em tempo real

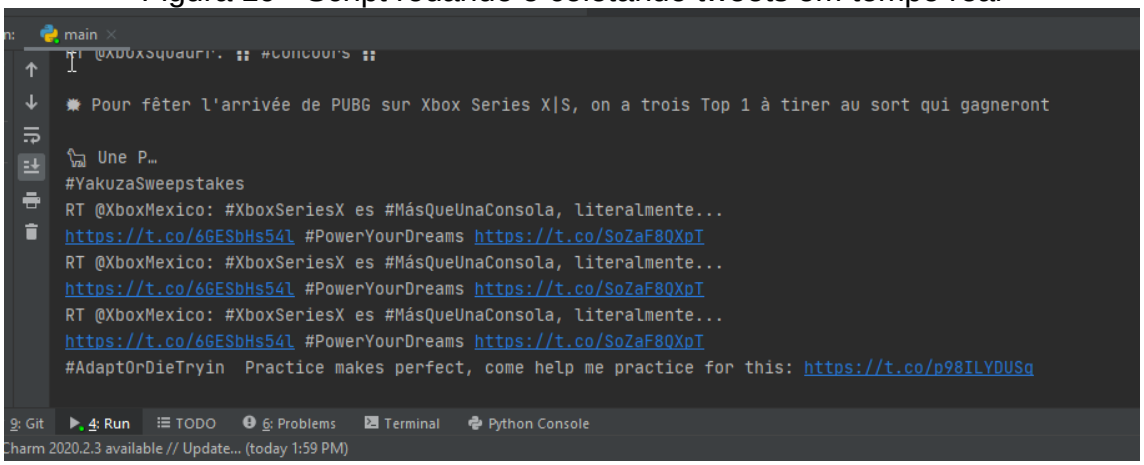


Figura 20 – Exemplo dos resultados

499135	b'GhostOfTsushima'
499136	
499137	b'Apex'
499138	
499139	b'Apexlegend'
499140	
499141	b'Apexlegendps4'
499142	
499143	b'PS4'
499144	
499145	b'PS4share'
499146	
499147	b'PS4share'
499148	
499149	b'PS4live'
499150	
499151	b'GhostOfTsushima'
499152	
499153	b'IndieGame'
499154	
499155	b'PS4live'
499156	

4.2. DATA SCIENCE

Com os dados tratados e armazenados, agora é possível partir para o processamento desses dados, para isso, vai ser utilizado o Hadoop.

4.2.1. HADOOP

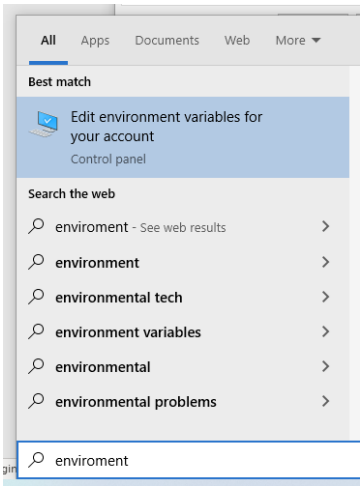
Primeiramente, é necessário fazer o download do haddoop, ele se encontra disponível em: https://hadoop.apache.org/releases.html. Acesso em 13 de set. 2020.

Após o download, é necessário descomprimir o programa, isso pode ser feito em qualquer diretório, mas para a simplicidade, é recomendado ser descomprimido em um lugar de fácil acesso. Ao finalizar o processo de descompressão, alguns erros podem aparecer, mas eles podem ser ignorados, pois não afetam os arquivos que o haddoop precisa para funcionar.

4.2.1.1. REQUISITOS

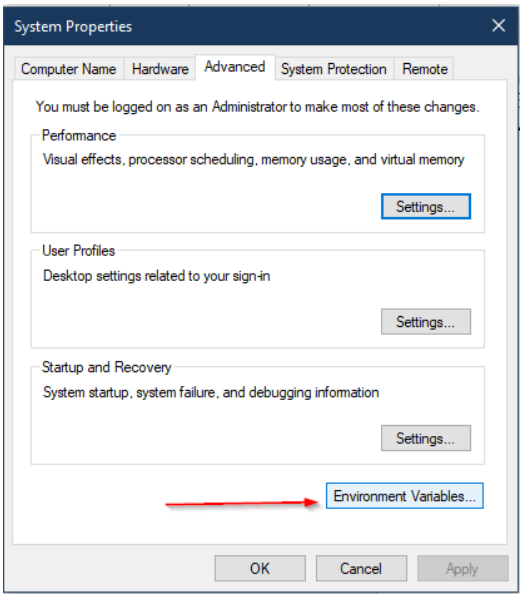
Para o utilizar o Hadoop, vai ser necessário primeiro, instalar alguns requerimentos, dentre esses requerimentos, a primeira dependência que deve ser instalada é o Java SE Development Kit, ou mais popularmente conhecido como JDK, para isso, basta ir no website da Oracle, e realizar o download normalmente, após a instalação, é preciso a criação de uma variável de ambiente para o Java. Vá na barra de pesquisa, e digite ‘environment’ ou ‘variavel’.

Figura 21 - Pesquisa pelo editor de variaveis do sistema



Após abrir o editor, uma janela das propriedades do sistema será aberta já na aba avançada, onde é possível acessar o local onde se adiciona variáveis de ambiente.

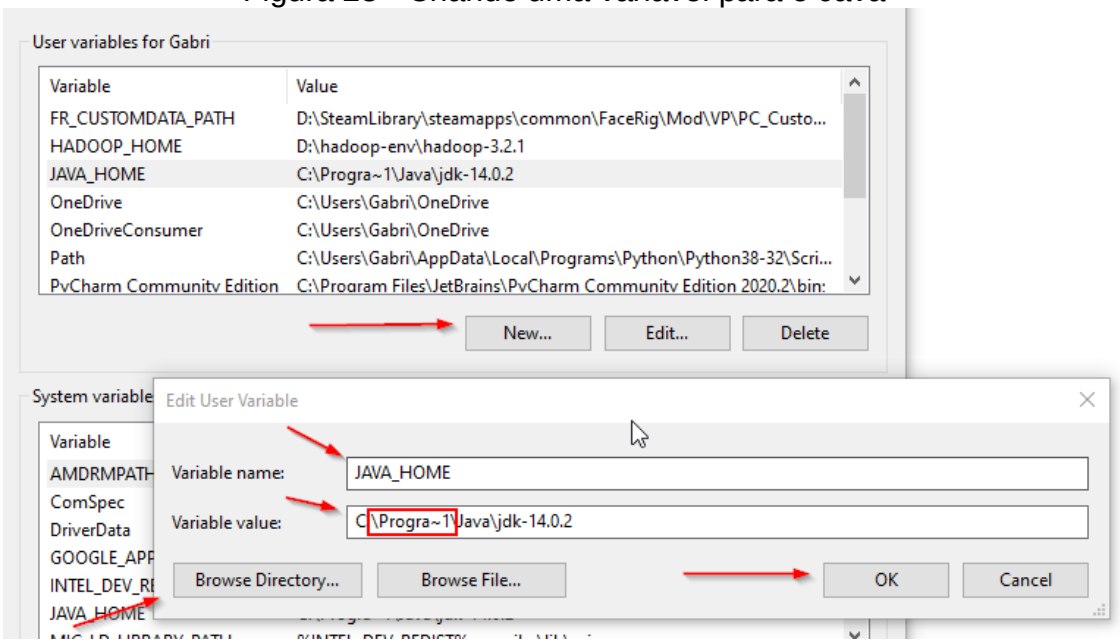
Figura 22 - Janela de propriedades do sistema



Com o menu aberto, é preciso criar uma variável de sistema ou de usuário, no campo nome, deve ser colocado como “JAVA_PATH” e no campo de baixo, o diretório onde está localizada a sua instalação do JDK. Como está sendo feito para ser usado por apenas um usuário, não há a necessidade de usar a variável de sistema, embora não haja diferença além disso, portanto, fica a critério do usuário qual variável usar. Outra coisa importante a ser lembrada, é que o sistema não reconhece espaços em branco, portanto, alguns erros vão ser encontrados durante a execução do Hadoop caso o diretório não for inserido usando o short name do Windows, que por sua vez é uma forma de descrever caminhos ao qual o sistema reconheça o diretório independentemente de quais caracteres forem usados nos nomes de pastas.

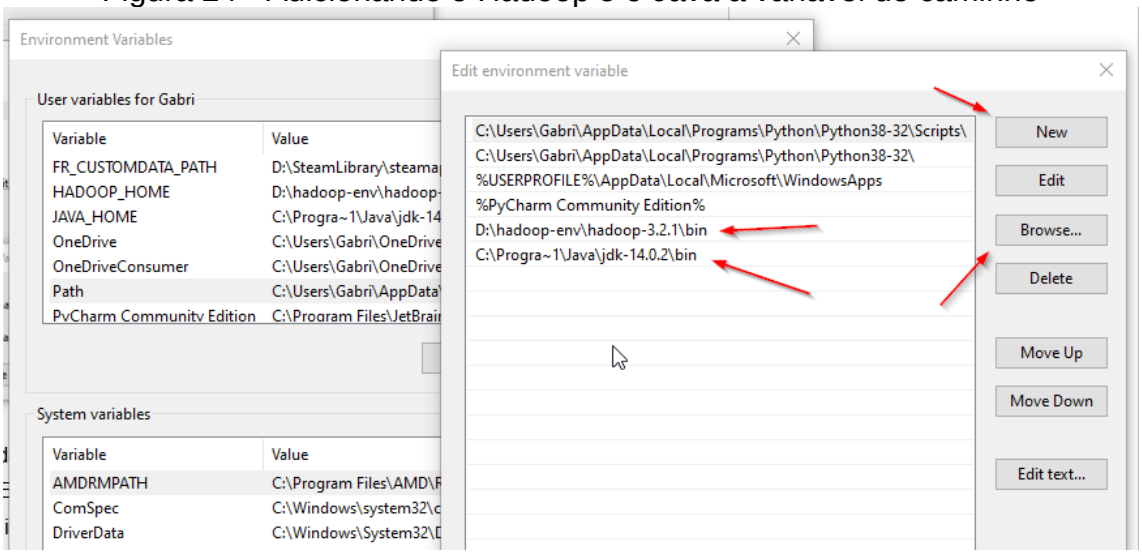
No caso da variável do Java, ele é localizado por padrão na pasta Arquivos de programas ou Program Files em inglês, e já que existe um espaço que pode causar problemas, a solução para isso é substituir “program files” para ‘progra~1’, que para o Windows, é reconhecido como o caminho correto.

Figura 23 - Criando uma variável para o Java



Em seguida, crie também uma variável para o Hadoop, chamado “HADOOP_HOME”, e como seu diretório, selecione o local onde foi instalado o Hadoop. E com isso, agora, na variável Path, selecione-a, e vá em editar, fazendo isso, uma nova janela vai ser aberta, ela é por onde se é possível adicionar caminhos para a variável.

Figura 24 - Adicionando o Hadoop e o Java a variavel de caminho



Após a criação das variáveis é recomendado a reinicialização do computador, pois o sistema precisa inicializar novamente as variáveis para que sejam reconhecidas. Com o OS reiniciado, agora é possível verificar a

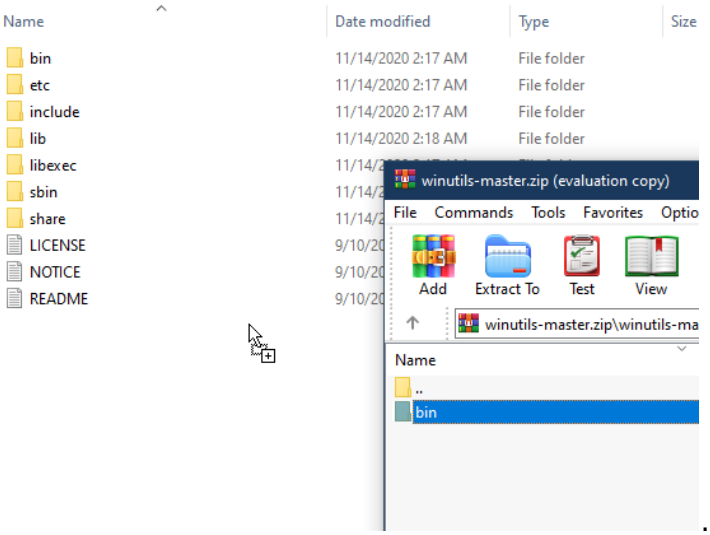
funcionalidade do Java no Hadoop, que caso configurado corretamente, vai exibir a versão do Java ao ser inserido o comando “Hadoop -version” no terminal.

Figura 25 - Verificando a funcionalidade do Java

```
PS D:\> hadoop -version
java version "14.0.2" 2020-07-14
Java(TM) SE Runtime Environment (build 14.0.2+12-46)
Java HotSpot(TM) 64-Bit Server VM (build 14.0.2+12-46, mixed mode, sharing)
PS D:\>
```

Feito isso, agora vem uma parte de extrema importância, os utilitários do Windows, que são algumas bibliotecas nativas que o Hadoop usa e são necessárias pois garantem algumas permissões e funcionalidades necessárias. A instalação delas são simples, bastando soltar os arquivos em suas pastas corretas no diretório do Hadoop. Esses arquivos estão disponíveis em: https://github.com/cdarlint/winutils. Acesso em 20 de out. 2020.

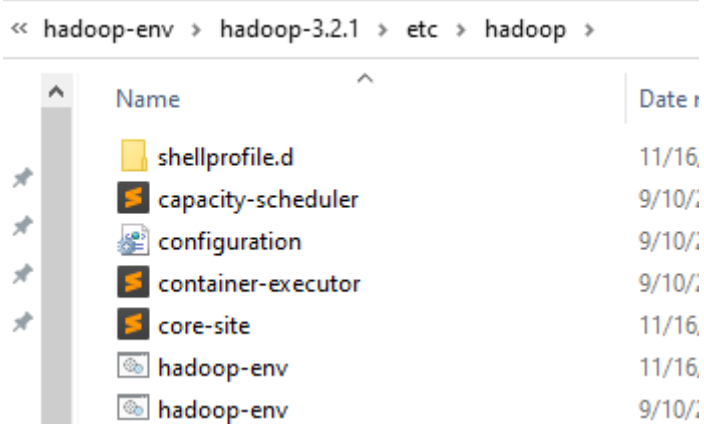
Figura 26 - Instalando os utilitários do Windows



4.2.1.2. CONFIGURAÇÃO

Após tudo instalado, é necessário configurar, isso é feito através de vários arquivos localizados dentro do local onde o Hadoop foi instalado. Esses arquivos vão ser editados com um editor de texto, que embora o notepad padrão do Windows seja totalmente funcional para isso, é recomendado usar outra ferramenta como sublime text ou notepad++, ambos possuem funcionalidades que ajudam no processo de edição. Tais arquivos, se localizam no seguinte diretório.

Figura 27 - Diretório dos arquivos de configuração



O primeiro arquivo a ser editado, será o ‘core-site.xml’, esse arquivo é responsável por definir o endereço onde vai ser possível acessar o painel de controle do Hadoop.

Figura 28 - Core-site.xml

```
19  <configuration>
20      <property>
21          <name>fs.default.name</name>
22          <value>hdfs://localhost:9000</value>
23      </property>
24  </configuration>
```

Em seguida, ‘hdfs-site.xml’, que por sua vez, é onde se define o diretório do datanode e namenode, além disso a quantidade de nós, que nesse caso, é apenas um.

Figura 29 - hdfs-site.xml

```
19  <configuration>
20      <property>
21          <name>dfs.replication</name>
22          <value>1</value>
23      </property>
24      <property>
25          <name>dfs.namenode.name.dir</name>
26          <value>file:///E:/hadoop-env/hadoop-3.2.1/data/dfs/namenode</value>
27      </property>
28      <property>
29          <name>dfs.datanode.data.dir</name>
30          <value>file:///E:/hadoop-env/hadoop-3.2.1/data/dfs/datanode</value>
31      </property>
32  </configuration>
33
```

Após isso, o próximo arquivo é o ‘mapred-site.xml’

Figura 30 - mapred.xml

```
<configuration>
    <property>
        <name>mapreduce.framework.name</name>
        <value>yarn</value>
        <description>MapReduce framework name</description>
    </property>
</configuration>
```

Agora, ‘yarn-site.xml’

Figura 31 - Yarn-site.xml

```
-->
<configuration>
<property>
  <name>yarn.nodemanager.aux-services</name>
  <value>mapreduce_shuffle</value>
</property>
<!-- Site specific YARN configuration properties -->

</configuration>
```

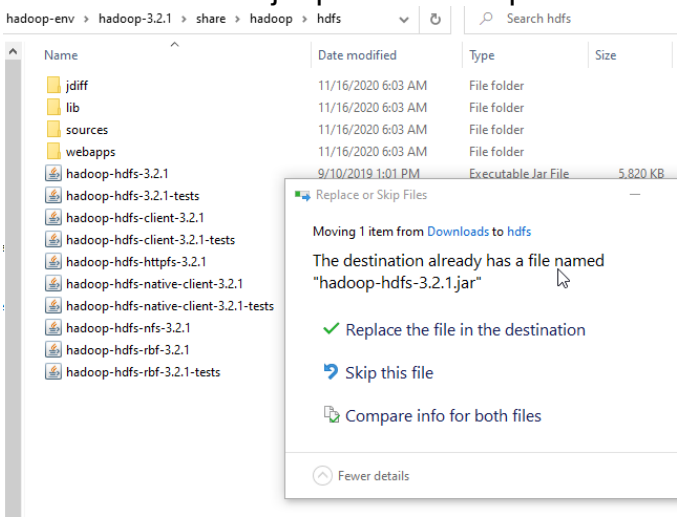
Com isso feito, agora é preciso formatar os namenodes do Hadoop, isso é feito de maneira bem simples, bastando apenas executar um comando dentro do diretório haddoop/bin.

Figura 32 - Formatando o namenode

```
************************************************************/
PS C:\TCC\hadoop-3.1.0\bin> .\hdfs namenode -format
2020-11-14 23:53:13,176 INFO namenode.NameNode: STARTUP_MSG:
```

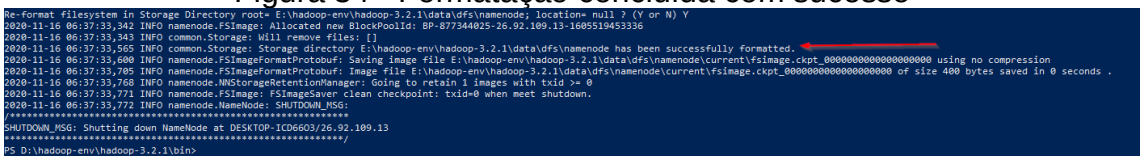
Ao formatar, talvez possa ocorrer, de uma mensagem de erro aparecer e não completar o processo por completo, isso acontece devido a um bug da versão 3.2.1, e pode ser corrigida através de um arquivo de extensão jar do hadoop diferente. Disponível em: https://github.com/FahaoTang/big-data/blob/master/hadoop-hdfs-3.2.1.jar. Acesso em 20 de out. 2020.

Figura 33 - Alterando o jar problemático por um funcional



Após a alteração, o comando pode ser executado novamente que dessa vez não vai aparecer nenhuma mensagem de erro, e ao invés disso, o processo vai ser concluído com sucesso.

Figura 34 - Formatação concluída com sucesso



E antes de finalmente iniciar o Hadoop, ainda tem mais uma coisa a ser feita, mover um arquivo essencial do serviço de yarn para a pasta correta.

Figura 35 - Selecionando o arquivo

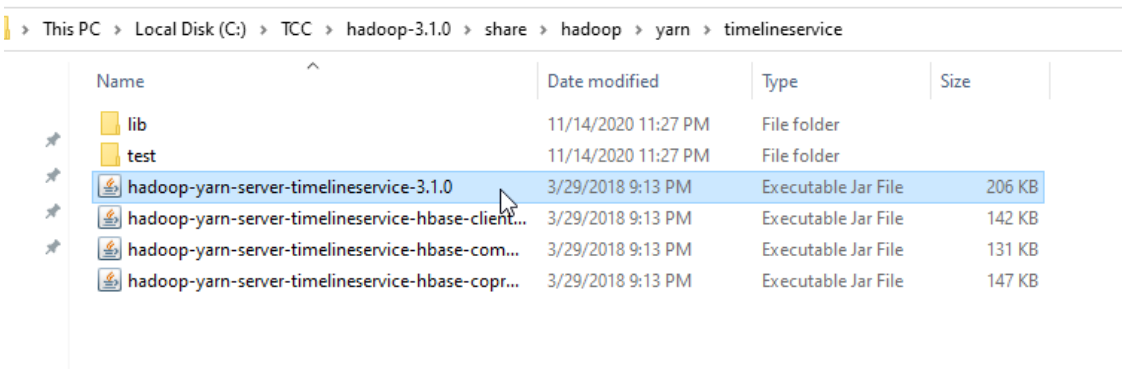
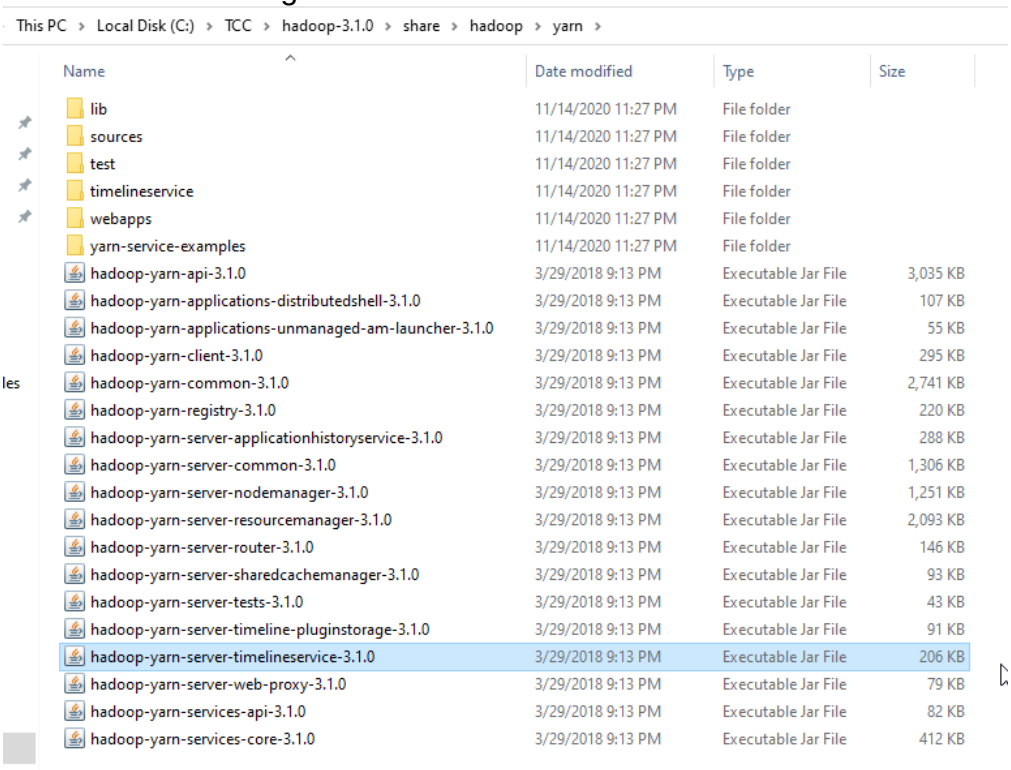


Figura 36 - Colando no diretório correto



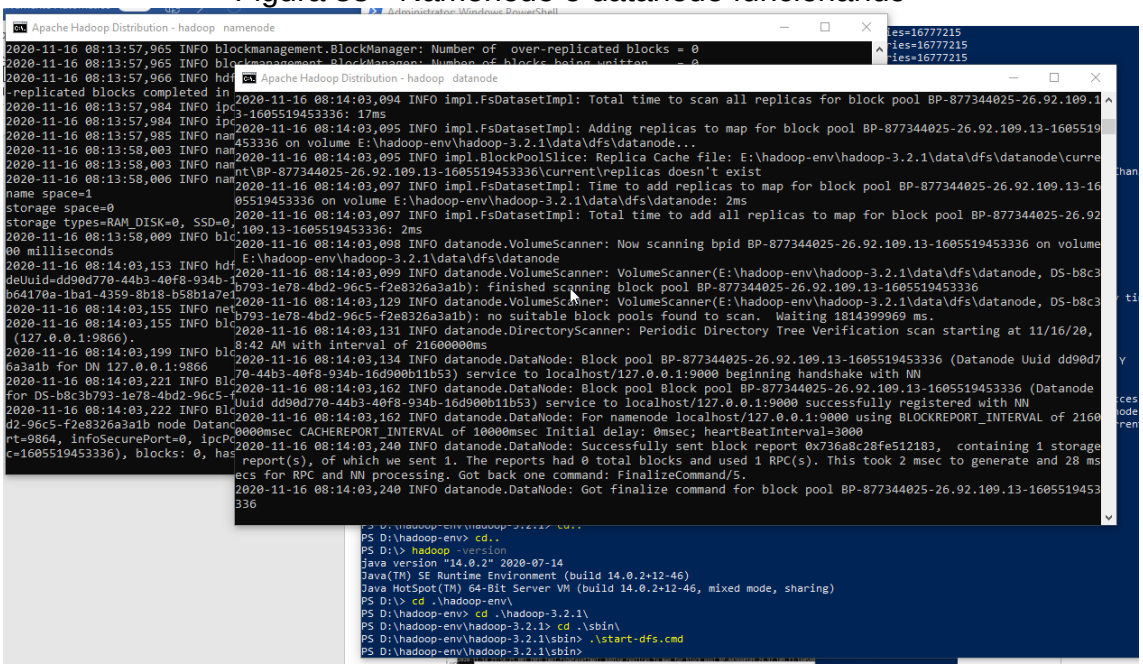
Com tudo configurado, agora é possível executar o Hadoop, isso pode ser feito de maneira simples, bastando voltar para a pasta anterior do Hadoop e agora acessando a pasta sbin. Primeiramente é iniciado o sistema de arquivos da Hadoop com o seguinte comando.

Figura 37- Iniciando o serviço

```
PS D:\> cd .\hadoop-env\
PS D:\hadoop-env> cd .\hadoop-3.2.1\
PS D:\hadoop-env\hadoop-3.2.1> cd .\sbin\
PS D:\hadoop-env\hadoop-3.2.1\sbin> .\start-dfs.cmd
```

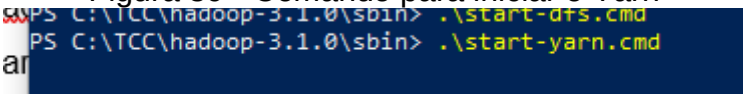
Feito isso, duas janelas vão ser abertas com o serviço de namenode e o datanode funcionando.

Figura 38 - Namenode e datanode funcionando



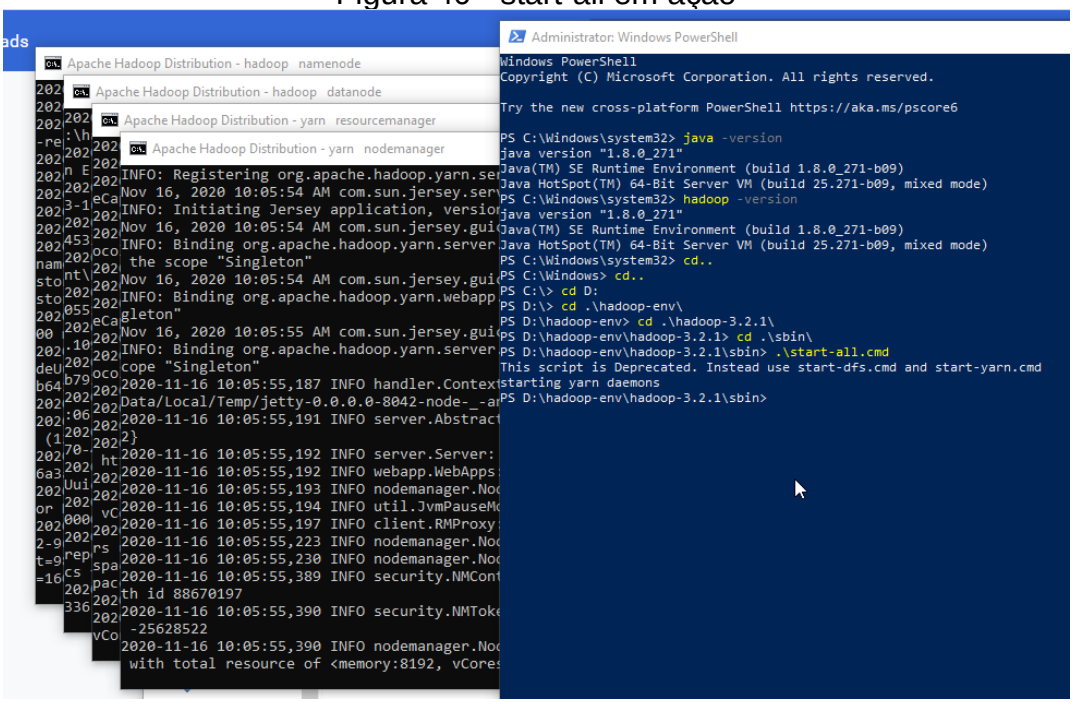
Com os serviços funcionando, agora é a vez do Yarn.

Figura 39 - Comando para iniciar o Yarn



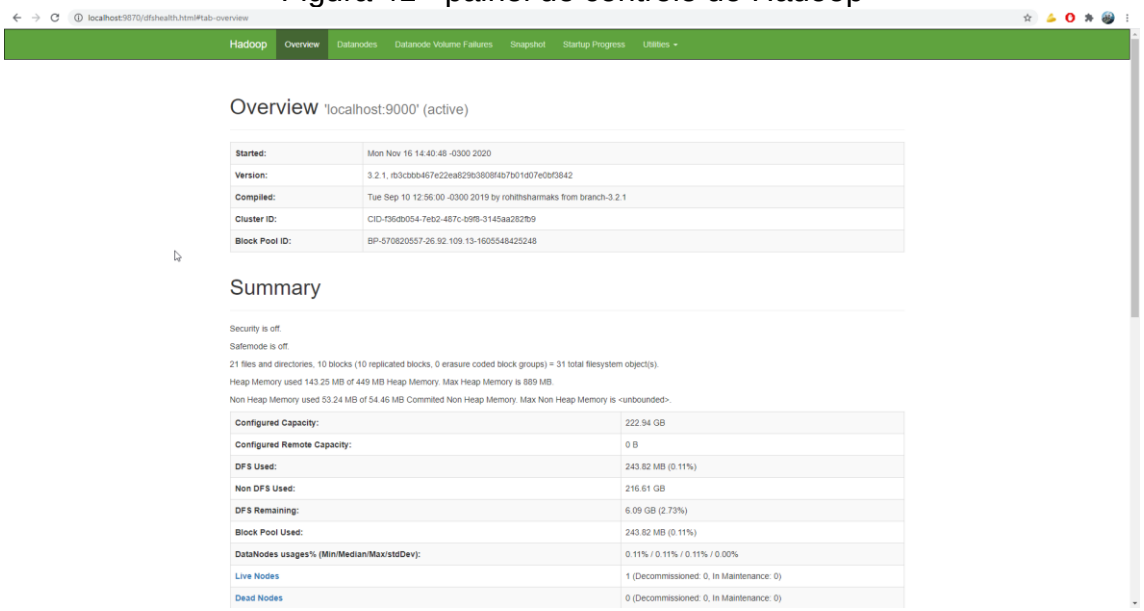
Alternativamente, também existe o comando que inicializa tudo já de uma vez, embora descontinuando, ainda é funcional.

Figura 40 - start-all em ação



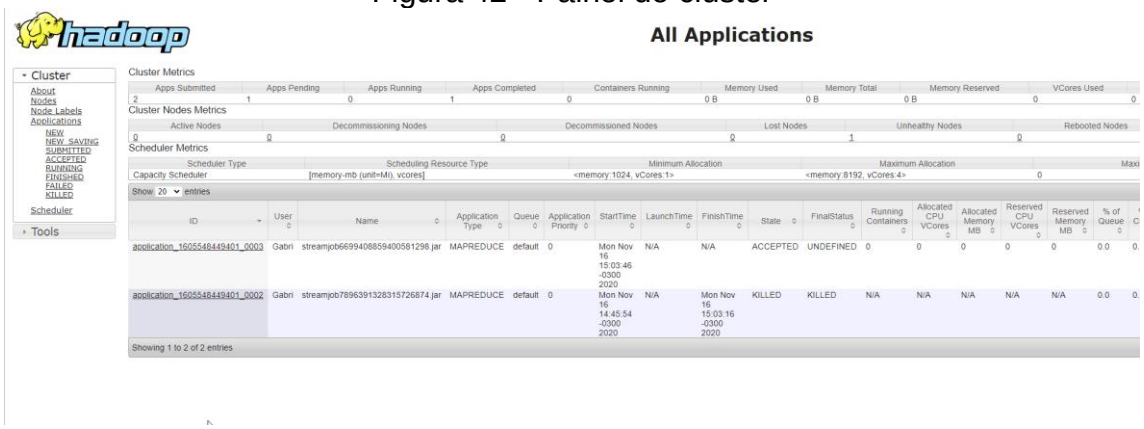
E assim todos os 4 serviços são inicializados, o namenode, datanode, gerenciador de recursos e o gerenciador de nós, e caso nenhum receba uma mensagem de shutdown, significa que estão funcionando normalmente e prontos para uso, além de também ser possível acessar o painel de controle do Hadoop diretamente do navegador.

Figura 41 - painel de controle do Hadoop



A partir desse painel, várias informações são disponibilizadas para o usuário, como quantidade de nós, porcentagem de armazenamento disponível, número de núcleos, e memória disponível. Existe também uma página onde se encontra informações do cluster, tais como a situação dos nós e os serviços a serem processados e seus status.

Figura 42 - Painel do cluster



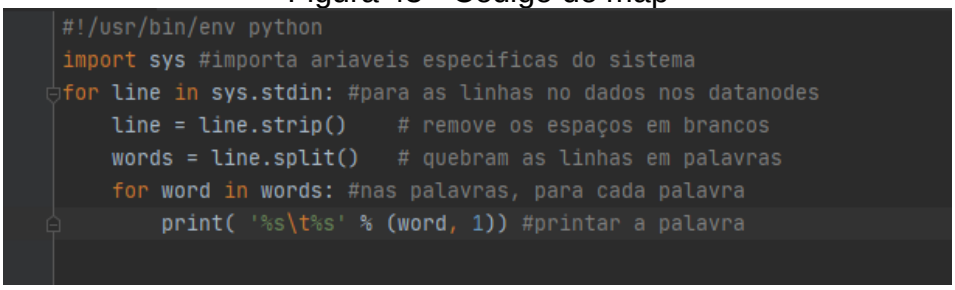
4.2.2. MAPREDUCE

Com o sistema funcionado, e testado, agora o próximo passo, é criar um script de Map e um Reduce, isso pode ser feito tanto em Java quanto em Python, ou até mesmo em outras linguagens como Ruby e C++, mas para o processo experimental foi realizado em Python.

4.2.2.1. MAP

O primeiro passo é criar o map.

Figura 43 - Código do map



Como o objetivo é contabilizar a quantidades de tweets que foram feitos durante o período ao qual o script de mineração foi executado, o que está sendo feito é um wordcounter, que já existe dentro dos exemplos que o haddoop oferece ao usuário ao instalá-lo, mas mesmo com um exemplo pronto, o mapper é relativamente simples de ser feito, bastando apenas algumas poucas linhas de código para ser criado em Python. A funcionalidade dele se baseia na separação das linhas em palavras para que em seguida o reducer faça a redução desses dados e junto disso a contagem das palavras.

4.2.2.2. REDUCE

Com o mapper já feito, então é possível criar o reducer. O reducer por sua vez, é um código mais longo e que vai ser o responsável por contar a quantidades de vezes que a palavra se repete, ou seja, um contador.

Figura 44 - Código do reducer

```python
#!/usr/bin/env python
from operator import itemgetter #importar o operador
import sys #importa o sys
current_word = None #palavras atualmente = nenhuma
current_count = 0 #atual contagem = 0
word = None #palavra = nenhuma
for line in sys.stdin: #para linha
    line = line.strip() #limpa a linha
    word, count = line.split('\t', 1) #quebra a palavra onde está escrito count
    try: #onde vai pegar o valor que está em int(1, no caso do mapper criado)
        count = int(count) #e vai inserir na variavel count
    except ValueError: #em caso de erro, o programa vai prosseguir normalmente
        continue #continua
    if current_word == word: #caso a palavra anterior for igual a palavra anterior
        current_count += count #adiciona mais um na contagem
    else: #se não
        if current_word: #caso palavra anterior
            print( '%s\t%s' % (current_word, current_count)) #printar a palavra atual e a contagem atual
        current_count = count #vai inserir a contagem da palavra atual na variavel count
        current_word = word #a palavra atual
if current_word == word: #caso a palavra atual for igual a palavra anterior
    print( '%s\t%s' % (current_word, current_count)) #printar palavra e contagem atual
```

4.2.3. COMANDOS ÚTEIS

Com o mapper e o reducer feitos, agora vem a parte onde se executam os códigos feitos a mão para testar as funcionalidades deles. Para executá-los, é necessário digitar um comando no terminal ou powershell.

A primeira coisa a se considerar, é que é possível navegar dentro do Hadoop, que funciona como uma VM, bastando ir no diretório raiz da sua instalação do Hadoop, e usar o ‘hdfs dfs’ e em seguida o comando, como por exemplo o mkdir para criar um diretório para os datasets.

Figura 45 - mkdir no hadoop

```
PS C:\Users\Gabri\OneDrive\Documentos\hadoop-3.2.1> hdfs dfs -mkdir /user
PS C:\Users\Gabri\OneDrive\Documentos\hadoop-3.2.1> hdfs dfs -mkdir /user/Gabri
PS C:\Users\Gabri\OneDrive\Documentos\hadoop-3.2.1> hdfs dfs -mkdir /user/Gabri/data
```

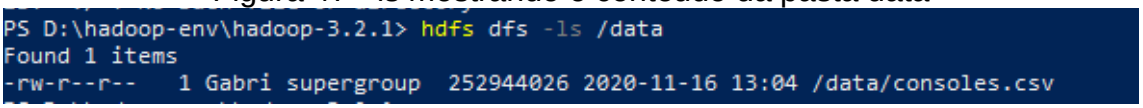
Outra função existente é a de copiar, o que faz possível, transferir arquivos para o ambiente do Hadoop, assim permitindo o processamento desse dataset.

Figura 46 - copyFromLocal

```
PS C:\Users\Gabri\OneDrive\Documentos\hadoop-3.2.1> hdfs dfs -copyFromLocal 'C:\Users\Gabri\
OneDrive\Area de Trabalho\TCC_UNIP\consoles.csv' /user/Gabri/data
2020-11-16 14:45:42,288 INFO sasl.SaslDataTransferClient: SASL encryption trust check: local
HostTrusted = false, remoteHostTrusted = false
2020-11-16 14:45:42,561 INFO sasl.SaslDataTransferClient: SASL encryption trust check: local
HostTrusted = false, remoteHostTrusted = false
```

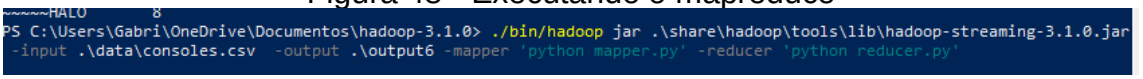
É possível também listar os diretórios e arquivos usando o ‘-ls’ com o mesmo prefixo, ‘hdfs dfs’.

Figura 47 - ls mostrando o conteudo da pasta data



Agora, o próximo passo, é executar os scripts de mapreduce, o primeiro passo é declarar que estamos usando o hadoop, e ler o arquivo de streaming, que vai ser o responsável por executar o comando de mapreduce e verificar o progresso, após isso, é informado a entrada, ou seja, o dataset, que foi movido para um diretório dentro do ambiente do Hadoop, e a sua frente, o local de saída do processamento, que necessita estar completamente vazio, caso contrário o processo vai falhar, e após isso, pode se colocar quais arquivos vão ser usados como o mapper e o reducer, e em caso dos arquivos não se localizem na pasta raiz do hadoop, vai ser necessário primeiro usar '-files' para apontar o local do arquivo, sendo importante declarar esses scripts como sendo feitos em Python, ou o programa não vai ser capaz de lê-los.

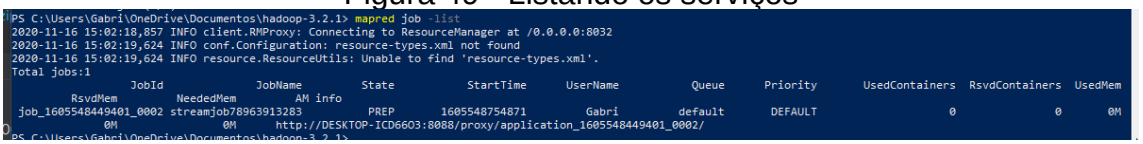
Figura 48 - Executando o mapreduce



Com todo o comando digitado, basta pressionar a tecla enter e enviar o comando, que caso esteja correto, vai iniciar primeiro o map e em seguida o reduce.

É importante deixar claro que por padrão, o hadoop precisa de 10% de espaço de disco disponível para executar um serviço, caso o espaço for menor que 10%, o nó vai ser considerado não saudável e não vai ser capaz de processar mais dados até que ou espaço seja liberado ou o valor seja diminuído manualmente nos arquivos de configuração.

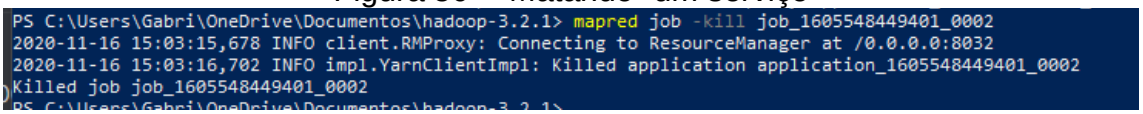
Figura 49 - Listando os serviços



Existe também o comando para listar todos os trabalhos de mapreduce do cluster, sendo mostrado várias informações úteis para o usuário, tais como o estado, a quantidade de nós usados, memória, a prioridade, e várias outras

informações. Além disso, há também a possibilidade de cancelar serviços pelo terminal, bastando apenas saber o código do serviço para poder cancelar.

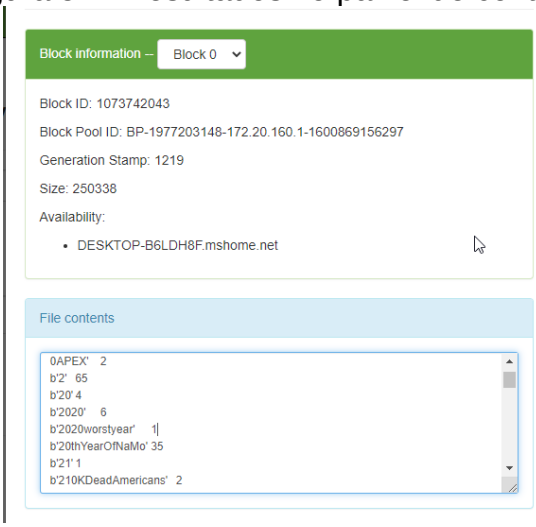
Figura 50 - "Matando" um serviço



4.3. RESULTADOS

Com tudo já feito, e o serviço de mapreduce concluído, agora basta exibir os resultados, sendo possível os ver de duas maneiras, a primeira é através do painel de controle do Hadoop no navegador, onde existe uma aba a qual se pode gerenciar os arquivos do ambiente. Através de lá, o diretório da saída é de fácil acesso, podendo ser aberto e visto pelo navegador.

Figura 51 - Resultados no painel de controle



Outra alternativa, é através do próprio terminal usando de um comando para converter em texto a saída.

Figura 52 - Comando para exibir o resultado

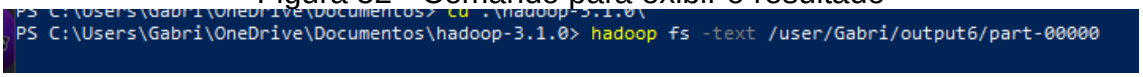
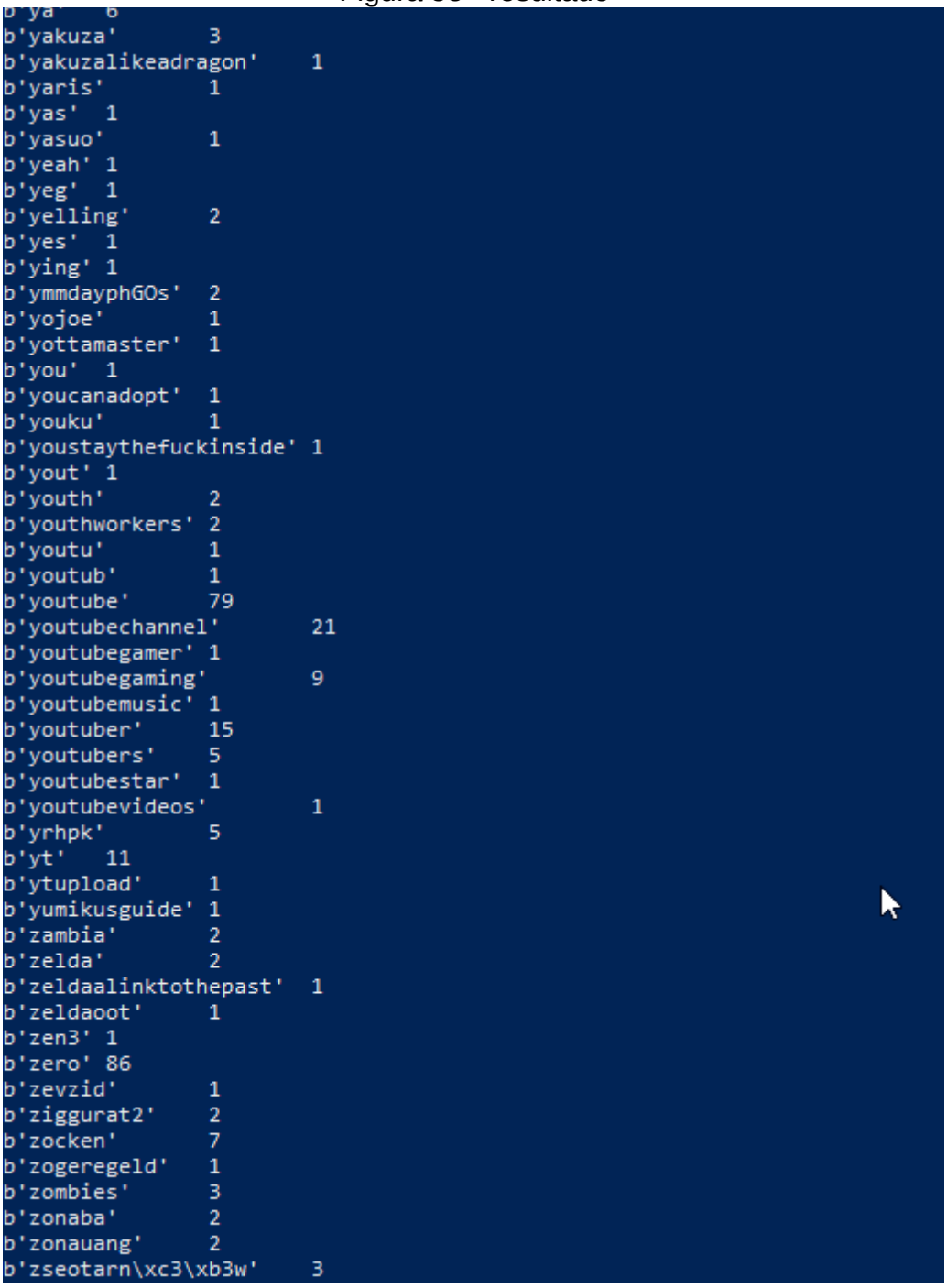


Figura 53 - resultado



Ainda é possível organizar por ordem decrescente, mas para isso é necessário a criação de outro mapper e outro reducer, dessa vez com uma complexidade maior, além de um conversor da saída para um arquivo sequencial.

CONCLUSÃO

Em conclusão, pode-se dizer que a área de Data Science, tem um grande potencial, entretanto, não é fácil de se tornar um profissional, pois, além de ser extremamente complexo o processo de instalação e configuração, existem diversos problemas que podem acontecer durante o percurso de aprendizado e uso.

E tendo isso em mente o trabalho procurou trazer soluções para os problemas mais comuns de se acontecer na operação do software, desde problemas específicos de versão ou até mesmo relacionados ao sistema operacional.

Muitas coisas poderiam ter sido encaixadas, como outras ferramentas e APIs, mas que fugiam do escopo e do objetivo geral do trabalho, além de não trazer uma carga de conhecimento que faria uma diferença significativa ou até mesmo não trazendo benefício algum.

O trabalho consegue trazer o prometido, uma fundação básica e ampla, para que o leitor possa usar como ponto de partida para explorar assuntos mais complexos do campo da ciência de dados, além de facilitar o progresso do aprendizado devido ao caminho bem definido já traçado desde o início.

REFERÊNCIAS BIBLIOGRÁFICAS

FERRANTE, R. **Data Engineer Interview Questions with Python**. Disponível em: <https://realpython.com/data-engineer-interview-questions-python/>. Acessado em 16 de outubro de 2020.

PYTHON SOFTWARE FOUNDATION. **Data Engineer Interview Questions with Python**. Disponível em:<https://www.python.org/>. Acessado em 17 de outubro de 2020.

LAYTON, D. **Data Engineering: What is it?** Disponível em:<https://towardsdatascience.com/data-engineering-what-is-it-ebd8e32df589>. Acessado em 16 de outubro de 2020.

ORACLE. **O que é Ciência de dados**. Disponível em: <https://www.oracle.com/br/data-science/what-is-data-science.html>. Acessado em 17 de outubro de 2020.

VENNERS, B. **Making of Python: A conversation with Guido Van Rossum**. Disponível em: <https://www.artima.com/intv/python.html>. Acessado em 21 de outubro de 2020.

PETERS, T. **Zen of Python**. Disponível em: <https://www.artima.com/intv/python.html>. Acessado em 21 de outubro de 2020.

PETER, P. **The impacts of big data that you may not have heard of**. Disponível em: <https://www.forbes.com/sites/peterpham/2015/08/28/the-impacts-of-big-data-that-you-may-not-have-heard-of/#2771fbb86429>. Acessado em 21 de outubro de 2020.

APACHE SOFTWARE FOUNDATION. **Apache Hadoop**. Disponível em: <https://hadoop.apache.org/docs/current/index.html>. Acessado em 25 de outubro de 2020.

TWEEPY. **Tweepy Documentation**. Disponível em: <http://docs.tweepy.org/en/latest/>. Acessado em 10 de setembro de 2020.

SANTINO, R. **Python passa a ser a segunda linguagem de programação mais popular**. Disponível em: <https://olhardigital.com.br/noticia/phyton-passa-a-ser-segunda-linguagem-de-programacao-mais-popular/104574>. Acessado em 7 de novembro de 2020.