Pandemia Covid-19 e o Mundo do Trabalho

Ana Luísa Carneiro[1*]

[1*]Departamento de Informática, Universidade do Minho, Braga, Portugal.

Corresponding author(s). E-mail(s): pg46983@alunos.uminho.pt;

Abstract

Desde de 2020 que a pandemia Covid-19 tem tido uma grande influência em todos os fatores da sociedade. Desta forma, este projeto visa mostrar como a pandemia Covid-19 afetou o espaço de trabalho de vários profissionais. Além disso, como o projeto está enquadrado na UC de Análise Inteligente em Sistemas de *Big Data* tem como grande objetivo sensibilizar para o uso de ferramentas de *big data* e como estas são aplicadas em casos práticos. Para determinar um pipeline que consiga resolver o caso de uso em questão foram utilizadas métodos e funcionalidades do Apache PySpark, MongoDB e PowerBi que permitiram processar, armazenar e visualizar os dados, respetivamente

Keywords: *big data*, *pipeline*, *dataset*, Covid-19, Python, Apache Spark, PowerBi, MongoDB

1 Introdução

A pandemia Covid-19 veio afetar a vida de muitas pessoas em vários níveis. A nível profissional, com a pandemia, muitos profissionais tiveram de realizar o seu trabalho remotamente e muitos deles foram demitidos devido às restrições *stay-at-home* provocadas pelo aumento do número de contaminados. Com estas restrições muitos locais de trabalho foram obrigados ou recomendados a fechar ao público, dando a permitir aos trabalhadores realizar o seu trabalho em casa. Contudo, esta solução não foi viável para certos estabelecimentos tais como restaurantes, cafés e lojas que ao não terem público que mantivesse o estabelecimento ativo, foram obrigados a demitir trabalhadores.

O caso de uso apresentado neste projeto visa analisar métricas que investiguem o impacto que o aparecimento do Covid-19 teve nos estabelecimentos profissionais e na vida dos trabalhadores. Desta forma, pretende-se avaliar como é que o número de contaminados e de mortes por Covid-19 afetou as regras aplicadas a estabelecimentos profissionais e como é que a evolução do processo de vacinação influenciou nestes mesmos estabelecimentos. Também foi avaliado como é que a evolução da pandemia e da vacinação influenciaram no apoio financeiro a pessoas que perderam o seu trabalho, devido ao fecho dos seus locais de trabalho.

2 Metodologias

Inicialmente pesquisou-se diversos *websites* de forma a encontrar *datasets* que fossem facilmente concatenados com o *dataset* fornecido pelos docentes, onde é indicado para cada dia (desde de janeiro de 2020 até março de 2022) números representativos da evolução da pandemia em cada país. Desta forma é necessário que os *datasets* a serem concatenados com este tenham parâmetros como o dia e o país associados a cada observação. Assim, pesquisa dos *datasets* tomou uma abordagem *bottom-up*, isto é, a partir de *datasets* que faziam sentido concatenar com o fornecido chegou-se ao caso de uso apresentado.

Para tentar responder ao *use case* serão utilizados mais três *datsets* para além do fornecido pelos docentes. Um dos *datasets* representa a evolução diária da vacinação por país[1], outro mostra a evolução das restrições aplicadas aos locais de trabalho[2] e o terceiro mostra a evolução dos apoios financeiros a pessoas que perderam o seu trabalho durante a pandemia[3].

A pesquisa das ferramentas de *big data* a serem utilizadas para, posteriormente, responder ao *use case* foi dividida em quatro categorias: ferramentas para junção dos *datasets*, para o armazenamento dos dados, para o processamento e tratamento do *dataset* e para visualização dos dados. Nesta pesquisa foram encontradas cerca de 10 ferramentas, duas para cada uma destas categorias.

3 Estado da Arte

As ferramentas encontradas para a junção dos *datasets* foram o ***package*** **Pandas** e o ***package*** **Numpy** da linguagem Python. Para o armazenamento do *dataset* concatenado, as ferramenta encontradas foram o **HDFS** e o **MongoDB**. Para o tratamento e processamento dos dados as ferramentas pesquisadas foram o **Apache Spark** e o **MapReduce**. Finalmente para a visualização dos dados processados pode-se utilizar as ferramentas **PowerBi** e o **Tableau**.

3.1 Vantagens e Desvantagens das Ferramentas

De seguida, apresenta-se as vantagens e desvantagens de cada uma das ferramentas pesquisadas.

3.1.1 Junção dos *datasets*

- **Biblioteca Pandas:** É uma biblioteca da linguagem Python muito usada para o tratamento e armazenamento de *datasets*. A biblioteca Pandas possui diversos métodos e funcionalidades para representação de dados em larga escala, utilizando a menor quantidade de código possível. Apresenta ainda um vasto conjunto de funcionalidades personalizáveis para a análise e processamentos de qualquer conjunto de dados. Contudo, esta biblioteca apresenta uma sintaxe diferente e complexa da apresentada pelo Python e tem uma documentação difícil de entender.[4]
- **Biblioteca Numpy:** A biblioteca Numpy da linguagem Python, semelhante ao Pandas, permite ao utilizador processar e armazenar os dados em forma de um *array* de alta performance multidimensional designado por *numpy array*. Estes *arrays* são fáceis de utilizar, rápidos e permitem ao utilizador realizar cálculos sobre os dados desse *array* utilizando pouca memória. No entanto, esta biblioteca tem um armazenamento limitado pois permite armazenar no *numpy array* valores de um único tipo e está direcionado para armazenamento de valores numéricos. Além disso, também há necessidade de alocação continuada de memória devido a utilização de *arrays*.[5]

3.1.2 Armazenamento do *dataset* concatenado

- **HDFS:** O *Hadoop Distrubuted File System* é um sistema de ficheiros distribuído que foi criado com o propósito de armazenar grandes quantidades de dados a serem utilizados no processo de tratamento. Esta ferramenta tem um modelo coerente, simples, robusto e escalável permitindo facilidade no acesso a informação. Além disso, possui conectividade com diversas linguagens de programação e ferramentas e possui técnicas rápidas de recuperação dos dados. No entanto, esta ferramenta não é viável no caso da máquina local for abaixo sendo por isso necessário que os utilizadores mantenham cópias nas suas máquinas antes de utilizar o HDFS.[6]
- **MongoDB:** É uma base de dados NoSQL com um modelo documental fácil de utilizar e entender que fornece suporte a várias tecnologias e plataformas. Além disso, como se trata de uma base de dados documental não há necessidade de criar esquemas para o armazenamentos dados, pois basta desenvolver documentos com formatos facilmente manipuláveis e inserir-los na base de dados criada. No entanto, estes tipo modelos têm limitações no que toca aos métodos de análise dos dados e poderá ser lento para certos problemas.[7]

3.1.3 Tratamento e processamento dos dados

- **Apache Spark:** É um *software open source* usado principalmente para o processamentos de dados, considerado por muitos o futuro das plataformas de *big data*. O Spark utiliza um conjunto de benefícios que poucas plataformas conseguem oferecer. Este *software* é 100x mais rápido que o Hadoop

para o tratamento de grandes quantidades de dados, sendo por isso conhecido pela sua rapidez de processamento e oferece um conjunto de API's com diversos algoritmos de análise fáceis de entender e utilizar. Além disso, é uma poderosa ferramenta que suporta o uso de diversas linguagens para a escrita de código como Python, Java, Scala, entre outros e opera eficazmente algoritmos iterativos e complexos. Contudo, este *software* apresenta algumas falhas tais como na falta de otimização do código sendo por isso necessário que o utilizador otimize o programa implementado e apresenta poucos algoritmos de *machine learning*. Além disso, este *software* não utiliza nenhum sistema de ficheiros interno sendo necessário o uso de uma ferramenta externa e limita o número de ficheiros grandes a serem utilizados pelo *software*.[8]

- **MapReduce:** É uma ferramenta, do ecossistema do Hadoop, que permite dividir os dados de forma a distribuir o processamento por nodos diferentes. Devido a esta distribuição do tratamento é possível obter resultados mais rapidamente, não necessitando de muita memória para processar os dados. Além disso, a cada iteração do *pipeline* esta ferramenta consegue devolver o resultado pretendido possibilitando ao utilizador controlar melhor todas as execuções do *pipeline*. Contudo, é uma ferramenta que possibilita um único fluxo de *pipeline* e necessita de muito código manual para realizar o processamento.[9] Além disso, esta ferramenta é muito utilizada para análises e processamento simples, sendo que em análises mais avançadas ou pedidos mais interactivos e complexos esta ferramenta deixa de ser eficiente.[10]

3.1.4 Visualização dos dados

- **PowerBi:** É uma ferramenta da Microsoft usada para a conversão de dados brutos em informação significativa através do uso de tabelas e gráficos para visualização dos dados. Esta ferramenta oferece um conjunto de gráficos customizáveis e interativos que melhor se enquadram no problema com fácil integração e conectividade com vários tipos de formatos de dados (XML, JSON, entre outros). Além disso, tem uma interface intuitiva e de fácil uso, para utilizadores não familiarizados com este tipo de ferramentas. No entanto, esta ferramenta não oferece muitas opções de configuração de gráficos, não é muito eficiente a lidar com grandes quantidades de dados e é mais usada para dados que não sejam obtidos em tempo real.[11]
- **Tableau:** É um software que apresenta uma variedade de produtos com o objetivo de visualizar, explorar e entender os dados de grandes ou pequenos *datasets*. Este software apresenta um conjunto vasto de funcionalidades e opções de visualização que melhor se adequam ao problema, além de ser um software que não necessita de programação para implementar *data queries*. Esta ferramenta apresenta ainda algumas desvantagens que estão relacionadas com a falta de opções na migração dos dados entre os vários servidores do Tableau[12] e certas funcionalidades mais complexas são difíceis de entender e utilizar, tornando a ferramenta menos intuitiva. [13]

4 Resultados

Nesta secção apresenta-se a constituição e descrição de cada um dos *datasets* encontrados assim como a descrição e fundamentação da arquitectura proposta para responder ao caso de uso.

4.1 *Dataset*

Para responder ao caso de uso foram utilizados 4 *datasets*, sendo que 1 deles foi fornecidos pelos docentes e os restantes foram encontrados nos *websites* listados na secção referências. É de notar que todos os *datasets* estão em formato CSV exceto o *dataset* da evolução da vacinação que se encontra em formato JSON e que apesar das observações serem realizadas em períodos de tempo distintos para cada *dataset*, nem sempre os países têm dados associados a todos os dias dentro desse período.

***Dataset - Who Covid*:** *Dataset* fornecido pelos docentes que representa os dados globais associados à pandemia Covid-19, desde 3 de janeiro de 2020 até 1 de março de 2022, em cada um dos 236 países. Estes dados fazem de base para os restantes *datasets*. Para além dos atributos que representam o país, o código do país e a data, este *dataset* tem os seguintes atributos:

- ***New_cases***: número de novos casos de Covid-19, por dia.
- ***Cumalative_cases***: número casos de Covid-19 acumulados ao longo dos dias.
- ***New_deaths***: número de novas mortes por Covid-19, por dia.
- ***Cumalative_deaths***: número de mortes por Covid-19 acumulados ao longo dos dias.

***Dataset - Vaccination*[1]:** Este *dataset* representa a evolução do processo de vacinação desde 22 de janeiro de 2021 até 27 de março de 2022 em 235 países. De forma a representar esse processo de vacinação por país foram utilizadas algumas métricas e atributos, contudo é de notar que nem todas as observações possuem valores para todos estes parâmetros. Neste *dataset* existem cerca de 14 atributos representativos do processo de vacinação, no entanto a evolução da vacinação pode ser analisada através das seguintes métricas, sendo que as restantes servem para completar cada observação.

- ***country***: país.
- ***iso_code***: código do país.
- ***date***: dia da observação.
- ***total_vaccinations***: número total de doses administradas. Para vacinas que requerem doses múltiplas, cada dose individual é contada.
- ***daily_vaccinations***: novas doses administradas por dia.
- ***people_vaccinated***: número total de pessoas que receberam pelo menos uma dose de vacina.
- ***people_fully_vaccinated***: número total de pessoas que receberam todas as doses prescritas pelo protocolo de vacinação inicial, isto é, 2 ou 1 dose dependendo da vacina administrada.

- ***total_boosters***: número total de doses de reforço administradas.
- ***daily_people_vaccinado***: número diário de pessoas que receberam a primeira dose da vacina.

Dataset - Workplace closures*[2]:** Neste *dataset* encontra-se associada a cada dia (1 de janeiro de 2020 a 21 de março de 2022) e a cada um dos 186 países, as medidas aplicadas aos locais de trabalho. Desta forma, o *dataset* é constituído por 4 colunas onde a primeira representa o país, a segunda representa o código do país, a terceira o dia e a quarta as restrições aplicadas (workplace_closures***). Estas restrições são caracterizadas por valores de 0 a 3, sendo que 0 representa a inexistência de medidas, 1 equivale à recomendação de fecho, 2 reflete o fecho do local para uma categoria de trabalhadores e 3 implica o fecho do local exceto se este prestar serviços essências como farmácias, mercearias, entre outros. Nas restrições de 2 e 3, para além do fecho, é obrigatório que os estabelecimentos permitam a adoção do regime de teletrabalho por parte dos trabalhadores sempre que for possível.

Dataset - Income Support*[3]:** Neste *dataset* encontra-se associada a cada dia (1 de janeiro de 2020 a 21 de março de 2022) e a cada um dos 186 país, uma métrica que simboliza se o governo está a cobrir os salários ou a fornecer pagamentos em dinheiro, a pessoas que perdem os seus empregos ou não podem trabalhar. Desta forma, o *dataset* é constituído por 4 colunas onde a primeira representa o país, a segunda representa o código do país, a terceira o dia e a quarta a métrica aplicada (income_support***). Esta métrica é caracterizada por valores de 0 a 2, sendo que 0 representa a inexistência deste tipo de apoio, 1 simboliza que governo apoia estas pessoas com menos de 50% do salário perdido e o 2 implica que governo apoia com 50% ou mais do salário perdido.

4.2 Arquitetura

Tal como já foi mencionado anteriormente, o objetivo principal deste projeto é aplicar diversas ferramentas de *big data* num caso prático de forma a construir um *pipeline* que consiga responder ao *use case* estabelecido. Desta forma decidiu-se desenvolver uma arquitetura em 4 passos, onde na última etapa consigamos ter meios para analisar e visualizar os dados e assim explorar como é que pandemia Covid-19 influenciou o mundo do trabalho. As etapas e respetivas ferramentas estabelecidas para o *pipeline* a ser implementado são:

- **Etapa 1 - Junção dos *datasets*:** Nesta fase vamos utilizar a **biblioteca Pandas** que, no contexto do problema, será usada para a junção dos *datasets* pois, há familiaridade com os métodos que utiliza e como é um *package* da linguagem Python são mais fáceis de usar juntamente com esta linguagem. Comparativamente com o Numpy, a biblioteca Pandas é mais direcionada para a representação de tabelas com vários tipos de dados através do uso de *dataframes* permitindo ao utilizador maior facilidade na manipulação destes objetos.
- **Etapa 2 - Armazenamento do *dataset* concatenado:** Nesta etapa utilizamos a base de dados NoSQL, **MongoDB**, que será utilizada no *pipeline*

para armazenar o *dataset* após concatenação. Esta base de dados foi escolhida pois, é uma ferramenta fácil de usar que utiliza diversos conectores e métodos que permitem a comunicação com outras ferramentas na arquitetura. Comparativamente como o HDFS, o MongoDB é uma ferramenta mais familiar que armazena documentos do tipo JSON ou CSV de forma mais robusta e flexível. Além disso, tem uma linguagem estilo *query* que permite retirar informação do documento original de forma eficiente e rápida.[10]

- **Etapa 3 - Processamento e tratamento dos dados:** Nesta fase de ETL (*Extract, transform, load*) será utilizada a ferramenta **Apache Spark** pois, oferece aos utilizadores a API **PySpark** que aplica os métodos e funcionalidades desta ferramenta em ambiente Python, mantendo desta forma consistência com as bibliotecas de Python que também serão utilizadas. Comparativamente como o MapReduce, este software é mais eficaz na implementação de métodos iterativos e complexos.
- **Etapa 4 - Visualização dos dados:** Finalmente, nesta fase usamos o *software* PowerBi pois, apresenta uma variedade de gráficos e meios de visualização que se enquadram no problema e que serão usados para responder ao caso de uso apresentado. Comparativamente com o Tableau, o PowerBI é muito mais intuitivo e fácil de utilizar e os dados que serão utilizados para visualização não são obtidos em tempo real, não havendo a necessidade de utilizar uma ferramenta tão complexa como o Tableau.

Na figura seguinte encontra-se um diagrama representativo da arquitetura a ser desenvolvida assim como as ferramentas e linguagens utilizadas em cada etapa.

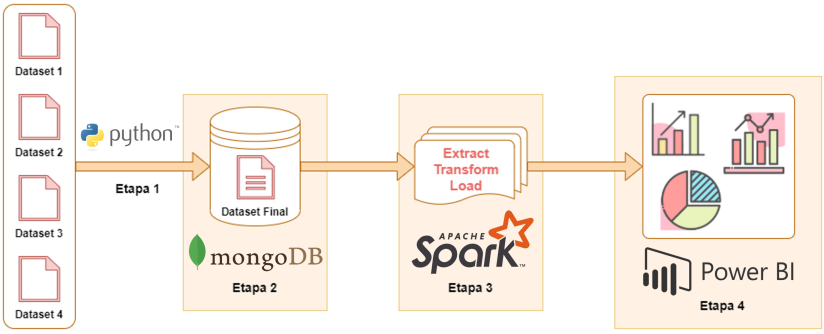

Fig. 1 Pipeline a ser aplicado ao problema

4.2.1 Etapa 1 - Junção dos *datasets*

Esta etapa do *pipeline* tem como objetivo juntar os 4 *datasets* de forma a formar um *dataset* em formato JSON com toda a informação necessária para responder ao use case.

Para realizar este processo começamos por transformar todos o *datasets* em *dataframes* do Pandas através dos métodos ***read_csv*** e ***read_json*** que lê ficheiros CSV e JSON, respetivamente. Para o *dataset* com formato JSON será necessário um processamento extra utilizando métodos do Python, de forma a conseguir transformar corretamente o ficheiro JSON num *dataframe*.

De seguida vamos juntar os 3 *datasets* encontrados utilizando o método ***merge*** do Pandas. Este método permite juntar os *datasets* segundo o nome das colunas, colocando a *null* os valores das observações que estejam em falta. Como o *dataset* de base (*Who Covid*) tem os códigos e nomes dos países diferentes dos *datasets* encontrados, então vamos utilizar a *package* **pycountry** que vai permitir associar os códigos dos países que estão no *dataset* base com os códigos que se encontram nos *datasets* encontrados. Desta forma criamos uma coluna nos *datasets* para que torne possível a concatenação dos dados. Esta concatenação vai ser realizada utilizando novamente o método *merge*.

Assim, é criado um *dataframe* final cujas as observações encontram-se sobre um período de tempo de 3 de janeiro de 2020 até 1 de março de 2022 para cada um dos 236 países presentes no *dataset* de base. Finalmente, utilizando o método ***to_json*** do *package* Pandas conseguimos passar o *dataset* final para o formato JSON.

4.2.2 Etapa 2: Armazenamento do *dataset* concatenado

Esta etapa do *pipeline* tem como objetivo armazenar o *dataset* obtido na etapa anterior na base dados MongoDB. Decidiu-se armazenar o ficheiro original com valores em falta, isto é, valores não tratados, pois caso haja necessidade de obter o *dataset* original sem dados sintéticos gerados, conseguimos obtê-lo com esta fase. Além disso, com o armazenamento deste ficheiro em bruto conseguimos identificar facilmente os dados sintéticos que serão gerados em fases futuras.

Inicialmente será necessário instalar na nossa máquina local o MongoDB e o MongoDB Express de forma a que estas ferramentas estejam disponíveis nas *ports* 8081 e 27017, respetivamente. De seguida será necessário criar uma base de dados e uma coleção utilizando o MongoDB Express onde se vai armazenar o documento JSON com o *dataset* concatenado.

Finalmente, utilizamos o método ***MongoClient*** da biblioteca Pymongo da linguagem Python de forma a criar uma conexão com a coleção e base dados onde vamos armazenar o ficheiro JSON. Utilizando o método ***insert_one*** desta biblioteca conseguimos armazenar o documento na coleção pretendida.

4.2.3 Etapa 3: Processamento e tratamento dos dados

Nesta etapa avançamos com o processo de tratamentos dos dados do *dataset* gerado, ou seja, será nesta fase que se vai implementar o ETL. Para isso utiliza-se a ferramenta Apache Spark, mais precisamente a API PySpark.

Inicia-se o processo com a conexão à base de dados MongoDB a onde vamos retirar o *dataset* armazenado na fase anterior. Para isso utilizam-se diversos métodos do PySpark como o ***SparkContext*** e o ***SparkConf*** que vão

utilizar um conector entre o MongoDB e o Spark de forma a estabelecer uma comunicação entre estas duas ferramentas. De seguida é utilizado o método ***SQLContext*** que irá se conectar à *port* onde corre o MongoDB e através de *queries* SQL retirar a informação presente numa determinada coleção de base de dados.

Como forma a realizar todo o tratamento necessário ao *dataset* é preciso criar uma *spark session* utilizando o método ***SparkSession***. Será nesta sessão que se irá gerar valores sintéticos para preencher os campos em falta no *dataset*, assim como processar e tratar todos os dados de forma a serem utilizados para análise e visualização na fase seguinte.

Inicialmente, nesta etapa de ETL, podemos analisar que atributos do *dataset* serão necessários para conseguirmos responder ao caso e uso apresentado. Caso uma coluna não seja considerada importante para avaliação dos dados pode ser retirada do *dataset* através do método ***drop*** do PySpark e assim não há necessidade de processar os dados presentes nessa coluna.

Tal como vimos na secção anterior os *datasets* encontrados tem informação correspondente a períodos de tempo distintos da pandemia. No entanto, como o *dataset* de base está sobre um período de 3 de janeiro de 2020 até 1 de março de 2022, este será o período de análise do caso de uso apresentado. Desta forma, será necessário gerar um conjunto de valores sintéticos para assim preencher as observações que se encontram em falta. As observações associadas ao processo de vacinação foram iniciadas a 22 de janeiro de 2021, não havendo informação sobre a vacinação desde de 3 janeiro de 2020 até esta data. Isto acontece pois, o processo de vacinação na maioria dos países só se iniciou em janeiro de 2021. Desta forma podemos assumir que os dados associados à vacinação em 2020 até 21 de janeiro de 2021 sejam iguais a 0. Para este processo utiliza-se o método ***withColumn*** do PySpark que preenche valores de uma coluna através de certas condições.

Ao longo do período determinado para análise (3 de janeiro de 2020 até 1 de março de 2022) poderá não haver informação de certos atributos para determinados países, sendo o valor desses campos iguais a *null*. Para preencher esses valores em falta será inicialmente realizada uma análise através da biblioteca ***sql.functions*** do PySpark que irá avaliar a percentagem de valores a *null* que existe num dado atributo quando associado a um país. Quando essa percentagem ultrapassar os 60%, então os valores gerados não serão representativos pois, prevê-se que não existem valores reais suficientes para conseguir gerar valores sintéticos que sejam próximos do real. Caso isso aconteça então esse país, assim como todas as observações a ele associadas, vão ser eliminadas do *dataset* através do método ***drop***. Caso haja valores não nulos suficientes para gerar valores sintéticos, então o preenchimento dos valores em falta será realizado utilizando o método ***withColumn*** e a função ***Last*** que preenche os valores a *null* utilizando o valor não nulo anterior. Utilizou-se este método pois, a probabilidade dos valores destas duas colunas não alterarem de um dia para o outro é grande.

4.2.4 Etapa 4: Visualização dos dados

Nesta última etapa do *pipeline* vamos implementar diversos gráficos e *dasboards* de forma a analisar a influência que a pandemia Covid-19 teve no mundo do trabalho. Para isso vamos utilizar a ferramenta PowerBi que permite a criação de gráficos que se enquadram no problema de forma a conseguimos obter uma resposta para o caso de uso. Após armazenar o ficheiro JSON processado e tratado, este será importado para o PowerBi onde vamos conseguir associar as variáveis e atributos do ficheiro de forma a criar um gráfico representativo do problema.

Iniciamos a análise com a criação de um gráfico temporal que avalie ao longo do período de tempo estabelecido o número de mortes e/ou número de contaminados por Covid-19 com as restrições de fecho de locais de trabalho. Desta forma podemos analisar como é que estes números influenciaram nas restrições aplicadas aos locais de trabalho. A este gráfico também pode ser acrescentado a evolução do número de doses da vacina que foram administradas como forma de assim avaliar qual a influência que a vacina teve nas restrições aplicadas.

Também pode ser criado um gráfico que permita avaliar como é que as restrições aplicadas aos locais de trabalho influenciaram no apoio do governo a pessoas que perderam os seus trabalhos. Neste gráfico também pode ser incluído o número de mortes e/ou o número de contaminados por Covid-19 e o número de vacinas administradas como forma de avaliar a influência que estes parâmetros possam ter no apoio do governo a pessoas necessitadas.

Neste estes dois gráficos podemos utilizar como filtro os países e assim avaliar qual a influência que a pandemia Covid-19 assim como o processo de vacinação tiveram no mundo do trabalho em diversos países.

5 Conclusão

Dada por concluído o projeto, faz sentido apresentar uma visão crítica do trabalho realizado. No espetro positivo, considera-se que o *pipeline* e arquitetura projetada assim como as ferramentas escolhidas conseguem responder de forma eficiente e coerente ao caso de uso apresentado. Além disso através do documento apresentado consegue-se ter uma visão ampla e justificada dos passos tomados na escolha das ferramentas e no *pipeline*. Contudo, sentiu-se dificuldades na descrição dos métodos que serão usados nas etapas da arquitetura, não estando totalmente de acordo com todos os passos a serem aplicados na prática. Em suma, considera-se que o balanço do trabalho é positivo, as dificuldades foram superadas e os requisitos propostos foram cumpridos.

References

[1] in Data, O.W.: Vaccinations - Covid-19 Data (Github). Github - https://github.com/owid/covid-19-data/blob/master/public/data/vaccinations/vaccinations.json (2022)

[2] in Data, O.W.: Workplace closures during the COVID-19 pandemic. Link - https://ourworldindata.org/grapher/workplace-closures-covid (2022)

[3] in Data, O.W.: Income support during the COVID-19 pandemic. Link - https://ourworldindata.org/grapher/income-support-covid (2022)

[4] Tonight, S.: Pros and Cons of using Pandas. Link - https://www.studytonight.com/pandas/pros-and-cons-of-using-pandas (2022)

[5] Pedamkar, P.: Introduction to NumPy. Link - https://www.educba.com/introduction-to-numpy/ (2022)

[6] Campus, T.: HDFS Overview. Link - https://www.tutorialscampus.com/hadoop/hdfs-overview.htm (2022)

[7] Help, S.T.: Top 15 Big Data Tools (Big Data Analytics Tools) In 2022. Link - https://www.softwaretestinghelp.com/big-data-tools/#9_MongoDB (2022)

[8] KnowledgeHut: Apache Spark Pros and Cons. Link - https://www.knowledgehut.com/blog/big-data/apache-spark-advantages-disadvantages (2021)

[9] Rodriguez, P.J.: Advantages and Disadvantages of MapReduce. Link - https://vikram-bajaj.gitbook.io/cs-gy-9223-d-programming-for-big-data/hadoop/advantages-and-disadvantages-of-mapreduce (2021)

[10] Simplilearn: Hadoop Vs. MongoDB: What Should You Use for Big Data? Link - https://www.simplilearn.com/hadoop-vs-mongodb-article (2021)

[11] Flair, D.: Pros and Cons of Power BI – The Bright the Dull side of visualization suite. Link - https://data-flair.training/blogs/power-bi-advantages-and-disadvantages/ (2022)

[12] Designers: A Complete Overview of the Best Data Visualization Tools. Link - https://www.toptal.com/designers/data-visualization/data-visualization-tools (2022)

[13] Biswal, A.: Power BI Vs Tableau: Difference and Comparison. Link - https://www.simplilearn.com/tutorials/power-bi-tutorial/power-bi-vs-tableau (2022)