9/3/2020

Vaxxmisinfo machine learning framework

Standard Operating Procedures

Muhammad Farhan Riaz & Matthew Vassov
UNIVERSITY OF TORONTO

Table of Contents

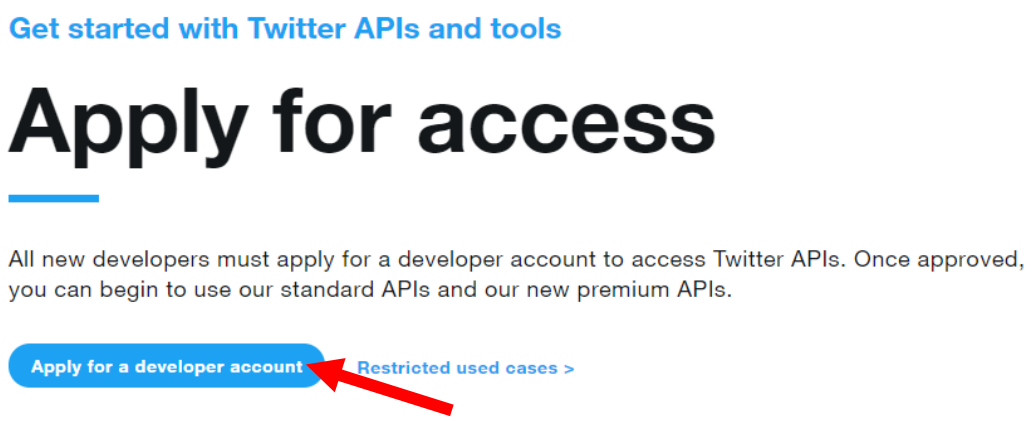
Introduction

The purpose of these Standard Operating Procedures (SOP's) is to guide the user with the knowledge required to execute the machine learning framework that has been built for the analysis. For simplicity, these SOP's will assume the user is executing code in the *Jupyter Notebook IDE*.

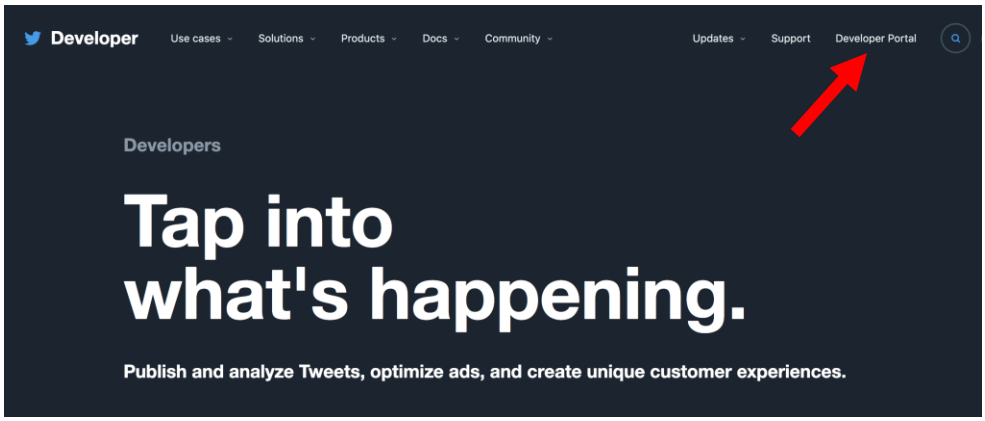
The following scripts have been written in Python and exported to a *Python Notebook (.ipynb)* filetype. It is recommended that *Jupyter Notebook* be used to run each concurrent script. It is also assumed that the user has a basic understanding of Python programming to seamlessly execute each module.

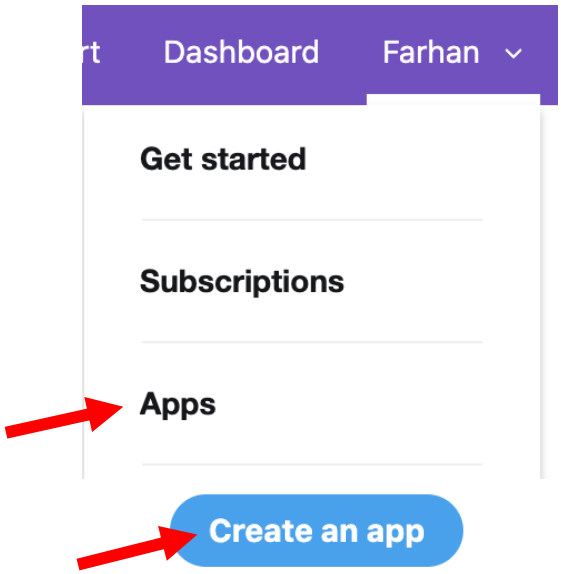
1. Signing Up for Twitter Developer Access

- The Twython API requires users to have their own Twitter account along with Twitter Developer Access (TDA) key credentials.
- First, visit the following URL and sign up for TDA by clicking on the **Apply for a developer account** button: https://developer.twitter.com/en/apply-for-access

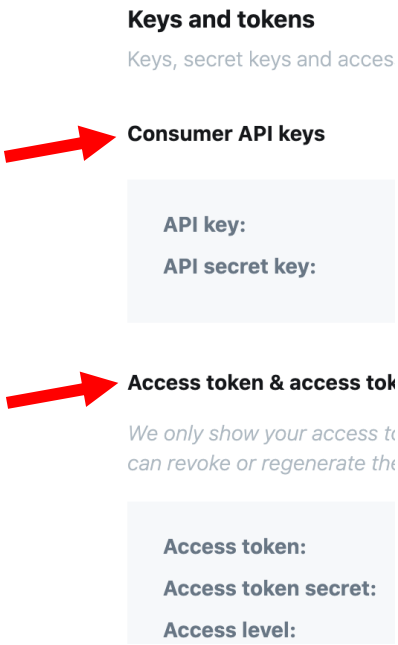


- Follow the steps and answer the questions. It may take some time for a Twitter representative to approve your application.
- Once you have been approved, you will have access to the Twitter Developer tools and will be able to apply for an App.





- Once the App has been created, you will receive four unique codes after logging in to your developer account:
  - **CONSUMER KEY**
  - **CONSUMER SECRET**
  - **ACCESS TOKEN**
  - **ACCESS SECRET**



- You will need these codes to run Twython, which can be accessed from GitHub: https://github.com/ryanmcgrath/twython
- Use Section 4 of these SOP's to understand how to upload the code to Jupyter.

2. End-to-End Project Workflow

The workflow for this project has been broken down into five key components that make up the overall framework. Below is a brief synopsis of what each component does and can be quickly references for future use. The five components are as follows:

1. Downloading Tweets

This requires background research and a list of all relevant ID's (in a CSV file) that can undergo Twython's hydration process. It requires an active Twitter account along with the account's consumer information (discussed in part 2 of these SOP's). The CSV file with the Tweet IDs must be one column only, containing the full ID of the Tweet (see screenshot below of sample)

1178824945238982656
1178829212104347648
1178830007965278208
1178831234857877510

2. Data Pre-Processing

This is where the first program downloaded from GitHub will come in to play. The user must have downloaded both the *01_vaxxmisinfo_data_preprocessing.ipynb* and *02_vaxxmisinfo_ml_training.ipynb.* The first program will be run in order to export the data into an Excel template that can be annotated by human annotators.

3. Data Annotating

Here, a large sample of Tweets need to be annotated from the exported data. There should be at least two annotators labelling the same data, and a tie breaker where the two annotators disagree. Ideally, all five classes (or however many are chosen for your variation of the project) should have the same number of samples.

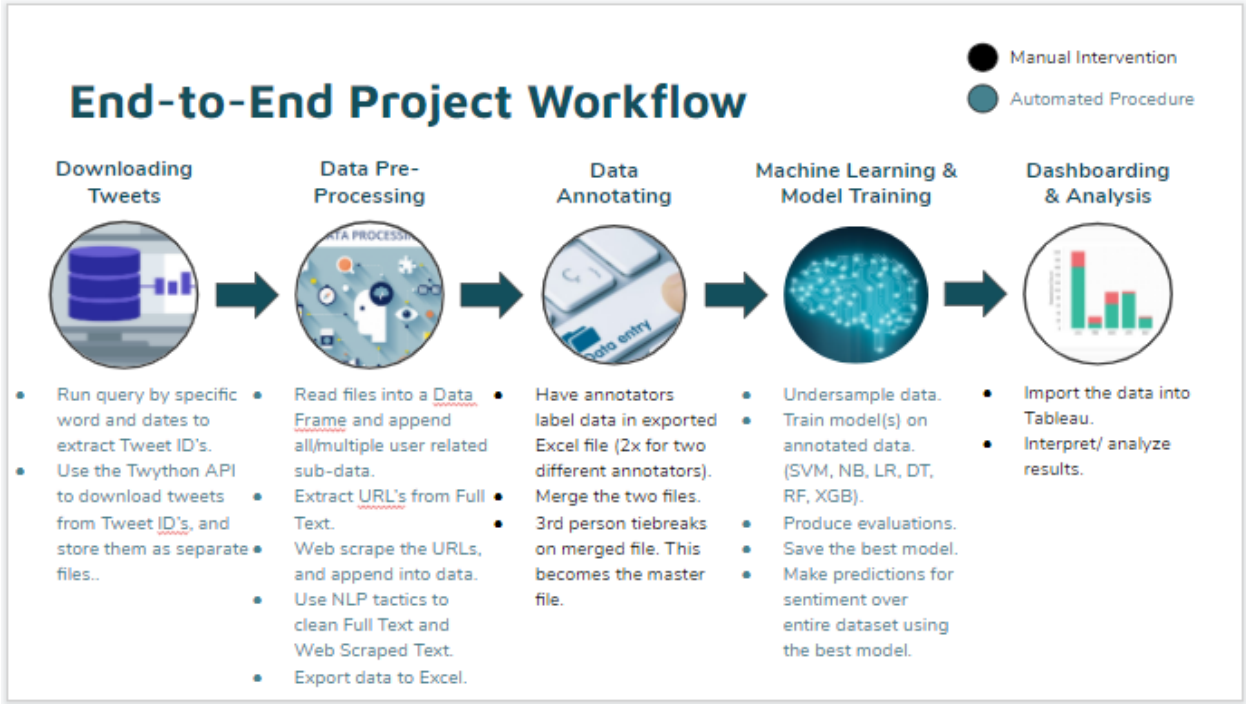
4. Machine Learning & Model Training

The second program will be run here. Once the data has been annotated and tie-broken, this newly labelled information must be fed as training data to the classifier. The program utilizes a grid-search which outputs the best word vectorizer, machine learning model and hyperparameters.

Each model will export a pickle file that can be used to re-train the data. The best model will be used to make the actual predictions over the rest of the dataset.

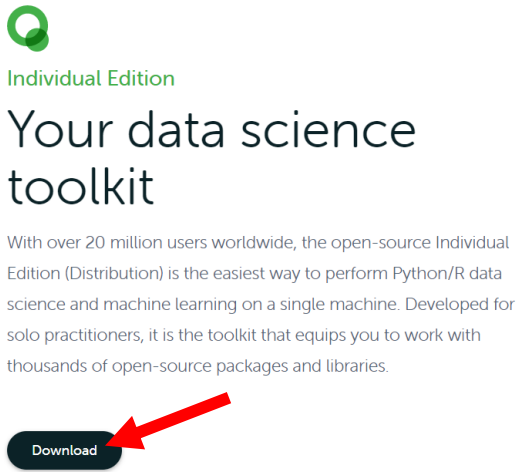
5. Dashboarding & Analysis

Finally, the predicted dataset can be uploaded to Tableau and used for analysis. This will be displayed in the form of an interactive dashboard.

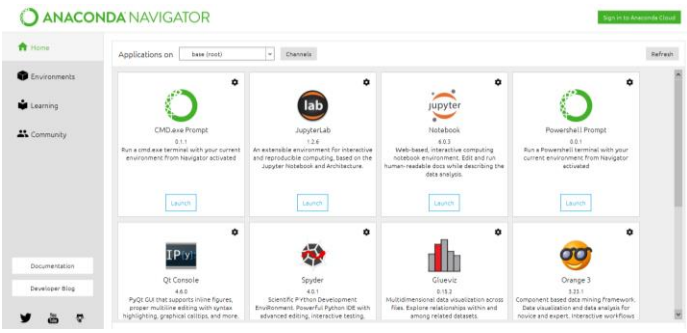


3. Downloading Anaconda

- To begin, Anaconda must be installed. Anaconda is a free, open-source data science platform which contains various IDEs for Python and R programming.
- Visit the following URL and click the Download button: https://www.anaconda.com/products/individual

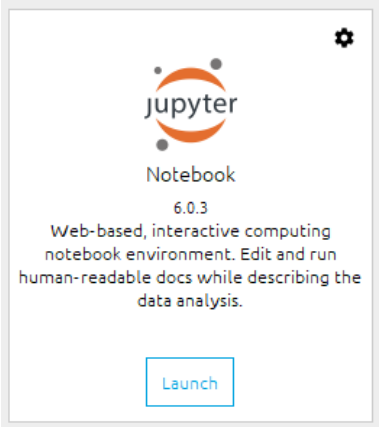


- This will prompt for a Windows, MacOS, or Linux installation. Select the appropriate operating system for your device.
- Once Anaconda has finished installing, open the program. This will bring you to the Anaconda Navigator page.

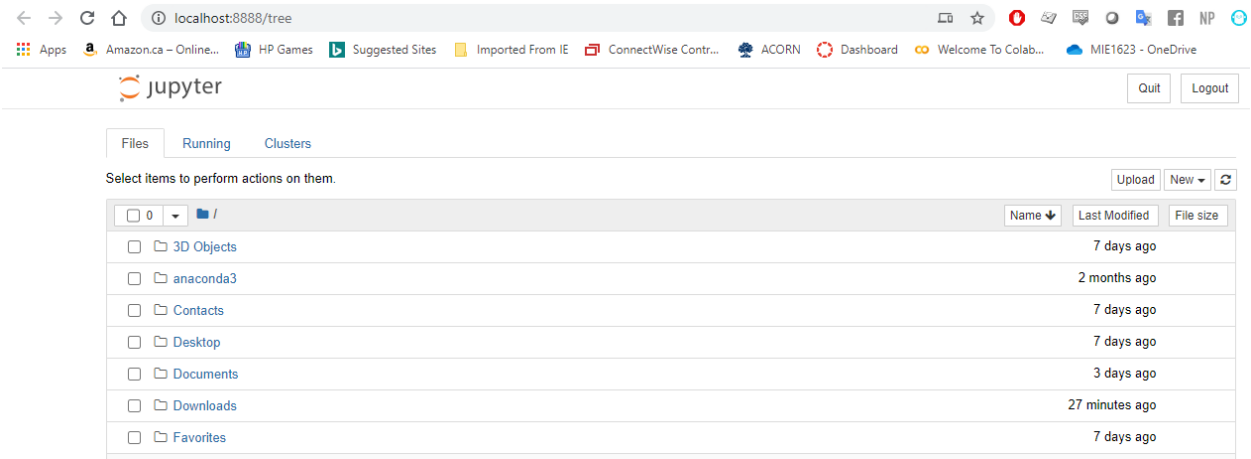


4. Uploading the Modules to Jupyter Notebook

- From the Anaconda Navigator, click the Launch button for Jupyter Notebook.



- When it opens, this will run a locally hosted server in your web browser.

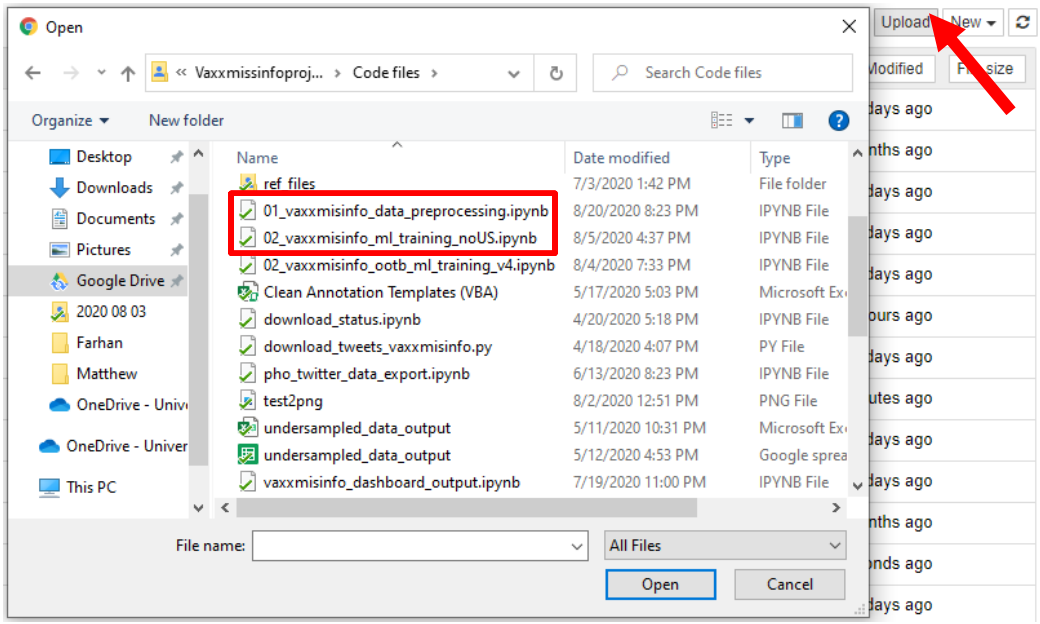


- Next, the must now be either uploaded to Jupyter, OR go to the directory where your code files/project files are located.

To upload:

- Click the Upload button and select the following two files:

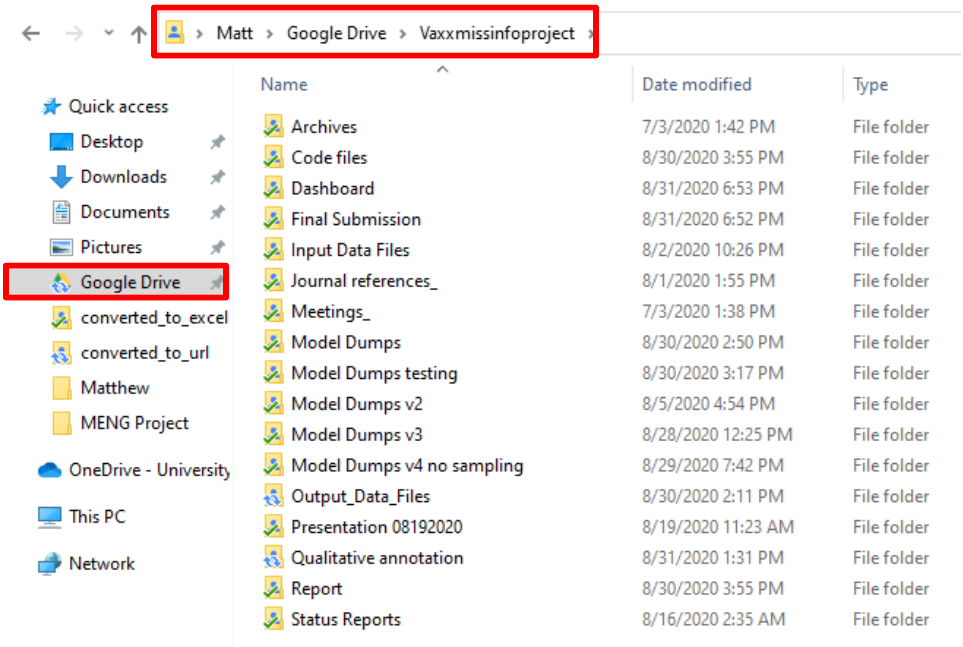
- *01_vaxxmisinfo_data_preprocessing.ipynb*
- *02_vaxxmisinfo_ml_training.ipynb*



- Both Python notebooks should now be available in Jupyter.

5. Running the Data Pre-Processing Code

- Open the *01_vaxxmisinfo_data_preprocessing* notebook.
- Before running the program, it is extremely important that your folders are set up to ***read from*** and ***write to***.
- If you are collaborating on the framework with others, it is highly recommended that you create these folders in **Google Drive** and set them up via **Backup & Sync**. This will allow you to **access your folders locally**.
- More on Google Backup & Sync can be found here: https://www.google.com/intl/en-GB_ALL/drive/download/backup-and-sync/



- There are **three primary folders** that the data pre-processing program will need to *read from* and *write to*. They are as follows:
  - A *read from* repository for all stored JSONL's from Twython.
  - A *write to* folder location to deposit all converted Excel files and one consolidated file.
  - A *write to* folder location to deposit the extracted URL Excel files and final consolidated master file.
- Once you have established these folder trees, change the name of the folder path under section 2 in the notebook. A set of instructions has also been provided in the code.
- The folder path names are all conveniently located at the beginning of the code to easily change.

2. Changing Folder Paths

The data-preprocessing program has been constructed in such a way where it constantly reads and exports data to your local folder paths. It is recommended that Google Drive Backup & Sync (GDBS) is installed on your device if you are collaborating with multiple people while using this framework. GDBS allows for your folders on Google Drive to be accessed locally by the program.

Please change you folder path names here. Each folder path can be described in detail below:

project_folder: root folder name for all relevant files.
json_path: folder to hold all exported JSON files from Twython.
converted_path: deposit initial Excel files.
converted_url_path: deposit final Excel files.

```
In [4]: project_folder = r'C:\Users\Matt\Google Drive\Vaxxmissinfoproject' # <-- change root folder name

        json_path = project_folder +'\Output_Data_Files\downloaded_json_files' # <-- change folder location for your stored JSON files
        converted_path = project_folder + '\Output_Data_Files\converted_to_excel' # <-- change folder location for depositing your Excel

        converted_url_path = project_folder + '\Output_Data_Files\converted_to_url' # <-- change folder location for depositing your Exce
```

- Next, section 3 will allow you to enter negative and positive keywords that may help identify the Tweet sentiment. A set of instructions has also been provided in the code.

- A list of positive and negative words has already been provided, but you may enter any additional terms in either list.

3. Keywords Filter List

Please enter a list of negative and positive keywords here. These will be used to identify potential negative and positive vaccine sentiments in the database when you are ready to label the training data. A "keyword_filter" column will appear at the end of the database which may speed up the labelling process and help to determine class targets.

A list of both sets of keywords has been identified below. Please add any additional words that may be relevant.

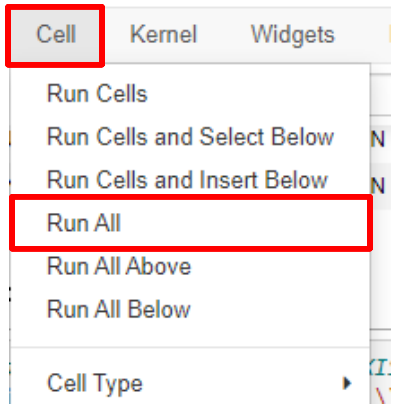
```
In [5]: neg_keywords = ["hearus","FDA", "poisoning", "ugly", "truth", "aborted", "tissue", "fetal", "population", "VAERS",
"africa", "fetal", "cells", "population", "control", "vaxxed", "fetus", "profit", "vaxxed", "force", "SIDS",
"victim", "agenda", "fraud", "sterilization", "victims", "allergy", "free", "choice", "theft", "wake up", "aluminium",
"freedom", "thimerosal", "wakefield", "aluminum", "Gates", "tissue", "warfare", "Bill", "gilead", "toxic", "weapon",
"black", "greed", "kill", "whistleblower", "chip", "hidden", "liars", "witnessed", "choice", "hoax", "mandate",
"compensation", "injure", "manmade", "control", "injured", "man-made", "corrupt", "injuries", "manufacture", "damage",
"injury", "merck", "damage", "insert", "mercury", "diabetes", "patent", "monetize", "disclose", "Poison", "natural", "engineer",
"Patent", "truth"]

        pos_keywords = ["flufighters", "flujab", "vaccineswork", "clinic", "public health", "idweek19", "flu shot", "flushot",
                        "fluvaccine", "flu jab", "flu vaccine", "get your flu shot", "fluchampions", "flu war", "flu season"]
```

- Finally, your Twitter consumer codes (from section 1 of these SOP's) must be entered.
- In the code, scroll down to section 6 (**Webscraping URLs & Appending to Master Dataframe**), sub-section **Twitter Extraction (url2tweetID > id2tweet)** and enter all 4 passcodes indicated by an XXXXXXX. A set of instructions has also been provided in the code.

Twitter Extraction (url2tweetID > id2tweet)

```
In [35]: tweets and title databases DO NOT EXIST in directory then create
         s.path.exists(converted_url_path + '\\twitter_place_url_extracted_tweets.xlsx') != True and os.path.exists(converted_url_path + ''

         '''url2tweetID'''
         extracted_tweets = {'unique_url':id_unique_url,'full_url':id_url_converted_list,'twitter_id':id_list}

         df_extracted_tweets = pd.DataFrame(extracted_tweets)
         df_extracted_tweets.rename(columns = {"twitter_id": "id"}, inplace = True)
         df_extracted_tweets.id = df_extracted_tweets.id.astype(int64)

         df_id_list = df_extracted_tweets.drop(columns = ['unique_url', 'full_url'])
         df_id_list.set_index("id",inplace = True)
         df_id_list.to_csv('extracted_tweet_ids.csv',encoding='utf-8-sig')

         '''ID2tweet'''
         '''Please put in your own credentials where XXXXXXX is indicated below'''
         !python download_tweets_vaxxmisinfo.py -i extracted_tweet_ids.csv -o extracted_tweet_ids_downloaded.jsonl --consumerkey XXXXXXX -
```

- Once all changes have been made in the code, you can finally run the program.
- To do this, click *Cell* > *Run All* from the top menu in Jupyter.

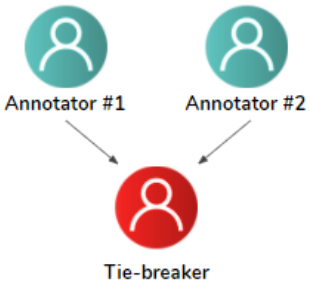


- ****Warning**** This may take a **long** time to finish processing. Several checkpoints have been implemented within the code to allow for automation and easy access should the code be run more than once.
    - However, if you are re-running with the intention of over-writing your previously exported files, please delete the files from that directory before running the code again.
- It is **highly recommended** that your computer utilizes a high-processing, multi-threaded CPU. It is also recommended that you monitor your core temperatures so as to not overheat the CPU. This program performs many complicated calculations which may bring your CPU to 100% utilization.

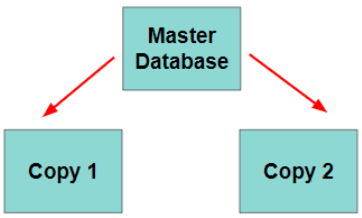
6. Data Annotation & Labelling

i. Preparation & Recommendations

- Data annotating requires **at least** two persons in charge of labelling the data and **a tiebreaker** (however, the more people involved in this process, the better). The reason for this is because people may have different opinions of how a Tweet should be labelled.



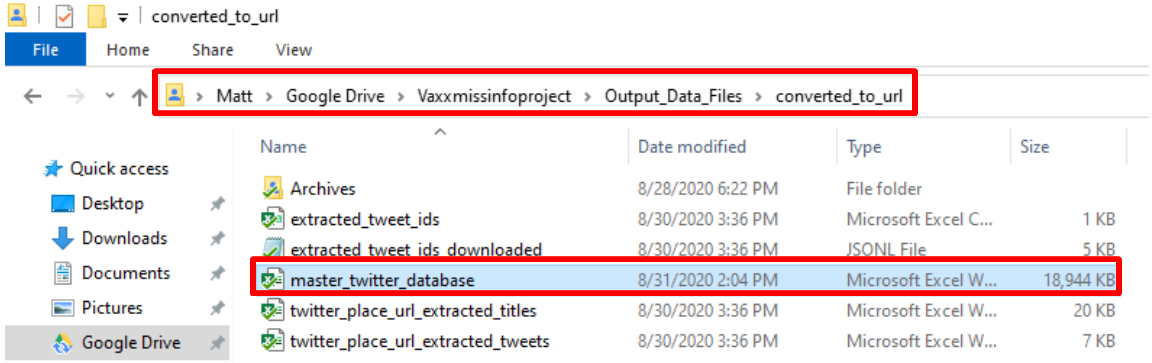
- The two (or more) annotators must label the **same Tweets** in **their own master file copies**. This will help **eliminate any biases** when it comes to labelling.



- When a sufficient sample of Tweets has been labelled, the tiebreaker must compare both sets of labels. Any differences between each set must be **resolved by the tiebreaker(s)**.
- When labelling the data, try to keep in mind **class balance.** All classes (0-4) should have an approximately equal number of labels (i.e. each class has ~2,500 labels).
- In order to increase the machine learning accuracy, a **reasonable number of Tweets** must be labelled! While this number may vary, try to aim between 12,500 – 50,000 labels (and potentially more!).

ii. Labelling the Master Excel File

- To begin, navigate to your defined folder path outputs.
- From your specified folder path, open the *master_twitter_database* Excel file.



- Here, the *target_label* column must be labelled from classes 0-4 based on the text written in the *full_text* column.

	D	E	F	G
1	full_text	clean_t	truncat	lay_tex
2	Stigmabase — Hope for HIV Vaccine Being Tested in South Africa: A	stigmabas	FALSE	[0, 241]
3	Stigmabase — New research reveals over half of people in Ireland h	stigmabas	FALSE	[0, 278]
4	Thoughts? San Francisco:: Tennessee raccoons to get vaccinated for r	thought sa	FALSE	[0, 97]
5	We ♡ vaccination 💉 @ Bristol, United Kingdom https://t.co/dMul	vaccin bris	FALSE	[0, 69]
6	My daily whatsyouroya is seeing all these Galloway Ridge employee	daili what	FALSE	[0, 236]
7	Flu shot clinic #2 for gallowayridge_fearrington employees today. 1	flu shot cl	FALSE	[0, 232]
8	🐕🐈 ATTENTION PET OWNERS! Humane Rescue Alliance Walk-In V	attent pet	FALSE	[0, 108]
9	I have got flu 🤧 vaccine shot @ Busch Campus of Rutgers University	got flu vac	FALSE	[0, 90]
10	Today is an amazing day! Today we released "Che". After several day	today ama	FALSE	[0, 239]
11	Your first line of protection from the flu season is getting your flu sho	first line p	FALSE	[0, 240]
12	Fighting Stigma: Ciara Kelly calls for Medical Council reform after 've	fight stign	FALSE	[0, 278]

AK	AL	AM	AN	AO
place_country	target_label	nique_	ed_title	d_twee
South Africa		https://t.co/1a916WBsA6		
Ireland		https://t.co/L0UCON7g78		
United States		https://t.co/oYlzZrdwE0		
United Kingdom		https://t.c	share photo instagra	
United States		https://t.c	han su instagram da	
United States		https://t.c	han su instagram flu	
United States		https://t.co/E5uAeHTSuh		
United States		https://t.c		
United Arab Emirates		https://t.c	leonardo lo pez inst	
United States		https://t.c	expressdoc urgent c	
Canada		https://t.co/ZYaiy5piHN		
United States		https://t.c		

- The following table categorizes the classes based on their misinformation type. The *target_label* column must be manually labelled while keeping these label classifications in mind:

Class Label	Categorization	Sample Tweets
0	• Pro-vaccine Tweets. • No misinformation.	"Just got my flu vaccine today, yay #vaccines"
1	• Neutral sentiment. • No misinformation.	"Vaccine supply shortage in India."
2	• Natural immunity is better than artificial. • Vaccines increase risk of diseases & disorders. • Vaccines make people sick.	"My baby experience seizures after she received 6 shots in 2 days."
3	• Pharmaceutical companies are making money off of sick people • Vaccination is genocide.	"Bill Gates is killing millions of innocent African children with the polio vaccine."
4	• Other	Irrelevant Tweets having nothing to do with vaccinations.
N/A	• Unavailable	Tweets no longer available, non-English or account suspended.

- You can also filter out Tweets by using the *keywords* column, which may help identify Tweet sentiment.

AM	AN	AO	AP
unique_	ed_title	d_twee	keywords
https://t.co/1a916WBsA6			none
https://t.co/L0UCON7g78			positive
https://t.co/oYlzZrdwE0			none
https://t.c	share photo instagr		none
https://t.c	han su instagram da		none
https://t.c	han su instagram flu		positive
https://t.co/E5uAeHTSuh			none
https://t.c			none
https://t.c	leonardo lo pez inst		none
https://t.c	expressdoc urgent c		positive
https://t.co/ZYaiy5piHN			none
https://t.c			none
https://t.c	instagram fun sick g		positive

7. Machine Learning & Model Training

- Open the *02_vaxxmisinfo_ML_training* notebook.
- Similar to section 6 of these SOP's, you will need to change the folder paths in order for the program to ***read from*** and ***write to***.
- To begin, in the code under section 1 (**Importing Modules**) sub-section **File Paths**, change the folder paths to match your local directories. A set of instructions has also been provided in the code.

File paths

```
In [ ]: project_folder = r'C:\Users\Farhan\Google Drive\Vaxxmissinfoproject'

        #This is the path where all your annotated data files are
        data_path = project_folder + '\Qualitative annotation\combined\Master' #<-Modify

        #The path where the .sav files will be dumped as the models get trained
        model_path = project_folder + '\Model Dumps testing' #<-Modify

        #The path where the the final files with predictions for the dashboard are outputted.
        dashboard_file_path = project_folder + '\Dashboard\pred_data'  #<-Modify

        #This is the path to the annotated data
        annotated_data_path = project_folder +'\Output_Data_Files\converted_to_url\master_twitter_database.xlsx'  #<-Modify

        #This is the path to the best selected model for final model deployment.
        #This will need to be updated based on your model's results.
        #Default is M3's Logistic Regression using TFIDF (based on best F1 Score)
        best_model = model_path + '\m3\model_logistic_regression_tfidf_f1_macro.sav'  #<-Modify
```

- The rest of the code can be defined based on user preference. Various machine learning classifiers have been specified along with their hyperparameters. It is up to you if you would like to change hyperparameter values or attributes.
- To do this, a dictionary *model_params* has been created. So far, *model_params* incorporates the following supervised models: SVM's, Logistic Regression, Naïve Bayes, XGBoost, Decision Trees, Random Forests, ADABoost, SGD's and KNN (not all shown below).

```
model_params = {
    'svm': {
        'model': svm.SVC(gamma='auto'),
        'params' : {
            'C': [1,10,20,30],
            'kernel': ['rbf','linear']
        }
    },
    'logistic_regression' : {
        'model': LogisticRegression(solver='lbfgs',multi_class='multinomial'),
        'params': {
            'C': [1,5,10]
        }
    },
    'naive_bayes' : {
        'model': MultinomialNB(),
        'params': {
            'alpha': [0,1]
        }
    },
    'xgboost' : {
        'model': xgb.XGBClassifier(),
        'params' : {
            'max_depth ':[10,20,50,100],
            'min_child_weight ':[5,10,15]

        }
```

- ****Note**** Please be aware that any additional details added to *model_params* may have a significant increase in program run-time.

VERY IMPORTANT: The model training code is set up in such a way as to utilize all cores and threads of your computer's CPU. This is done so by changing the *n_jobs* parameter in the **best_model_search()** function. *n_jobs* = -1 which utilizes all the cores / threads of your computer's CPU. Should you wish to not allocate the whole CPU towards model training (which is recommended), please adjust the *n_jobs* to be equal to the number of threads that you can sufficiently allocate. for example, if you

have a CPU with 4 threads, you may want to allocate only half of these to the training process, hence you would set *n_jobs* = 2.

```
# Modify n_jobs if needed

def best_model_search(X, y,
```

```
n_jobs =-1)
```

Similar to the data pre-processing code, the ML training code may take several hours to run – potentially longer if your CPU is slower. It is important to measure your CPU's core temperatures. For reference: On a Ryzen 7 3800X CPU, with all 16 threads being used, the training time with under sampled data was about 4 hours!

8. Dashboard

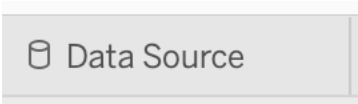
i. Connecting to data

Important note: This is not an exhaustive tutorial on Tableau. It is expected that the user has a general / basic understanding of using Tableau before attempting to use it in the context of this framework.

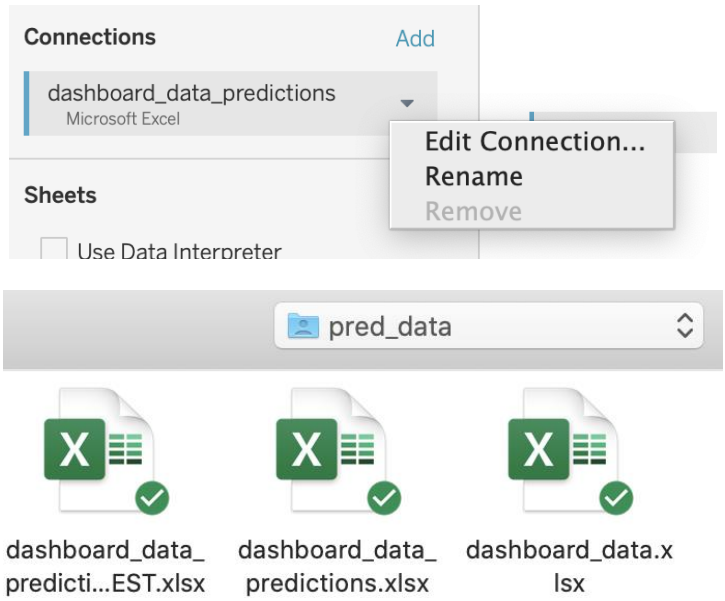
When you open the dashboard file for the first time, you may be prompted to 'connect to the dataset' or a similar error. This indicates that the data source that was originally used to create the dashboard cannot be located. It should then give you the option to select the file you wish to use (by opening an Explorer or Finder window, depending on your OS). You may simply select the "**dashboard_data_predictions.xslx"** which gets created at the end of the ML training code in order to connect the correct file.

ii. Troubleshooting instructions to connect data

In case the above option is not viable, the following method can be used to connect the dashboard to your dataset. To connect to your newly trained data, open the dashboard file that accompanies this project and go to the tab on the bottom left titled "Data Source"



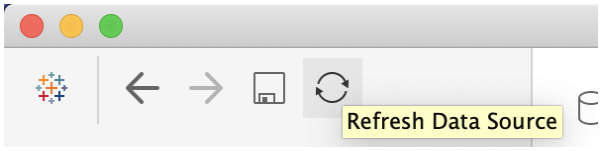
Once here, you will see the list of 'Connections' on the top left pane of the window. By default the dashboard should already have a sheet connected to it. This connected will need to be updated. To do so, click on the dropdown arrow and click on 'Edit Connections'



This will take you to an Explorer Window (for Windows) or Finder (for MacOS). Select the file you would like to use as input into this dashboard. If you have followed along in the process above (with the ML training), the file should be labelled as "**dashboard_data_predictions.xslx".**

iii. Refreshing data

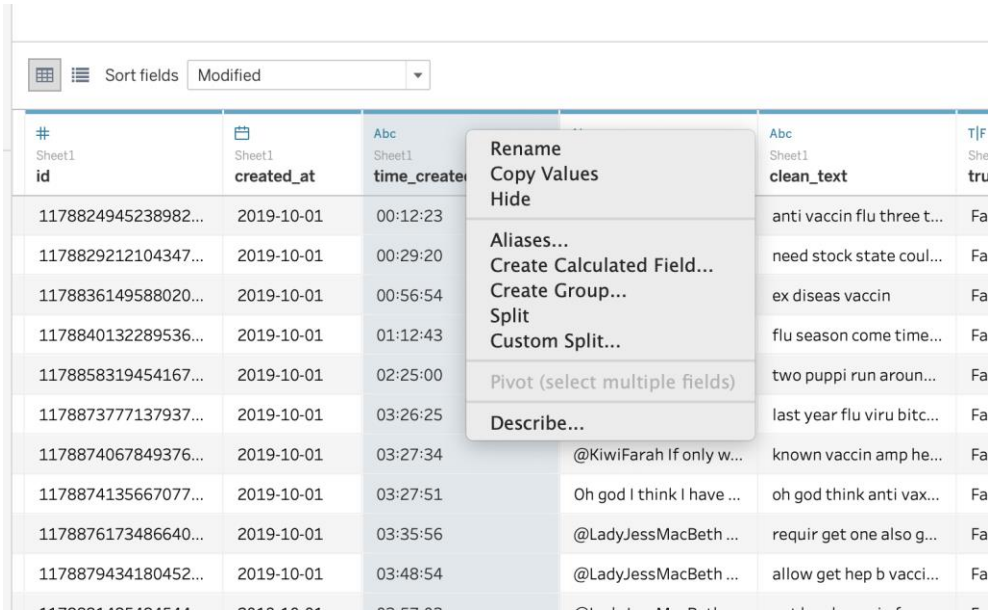
Make sure to refresh your data connection every time you create a new predictions file, thereby allowing for the dashboard to be as updated as possible. This can be done by clicking on the 'Refresh Data' button.



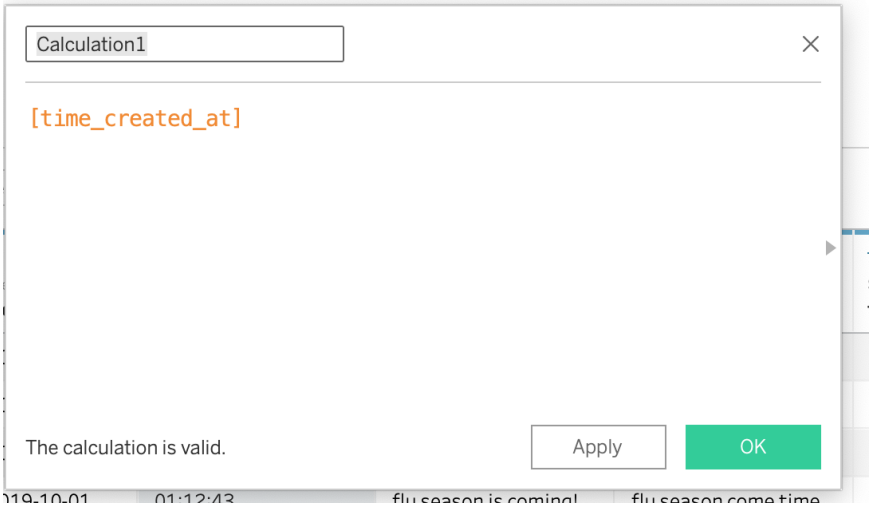
iv. New Calculated Fields in Data Source

Two new calculated fields were created for this dashboard. The intent of these fields is to provide ease in navigation and slicing the data. These should already be pre-existing if you are using the dashboard file which accompanies this project, however if for whatever reason they need to be recreated, the parameters and instructions are provided below:

Instructions to create a new calculated field in the data source: Right click anywhere in the preview section of the Data Sources tab. Then click on "Create Calculated Field…" in the menu.



This will open a new window which will allow you to enter a new field (column) name, along with an area where the formula can be entered.



The table below provides the two new field names (column names) and formulae, which can be used to recreate the calculated fields.

New Calculated Data Field	Calculation

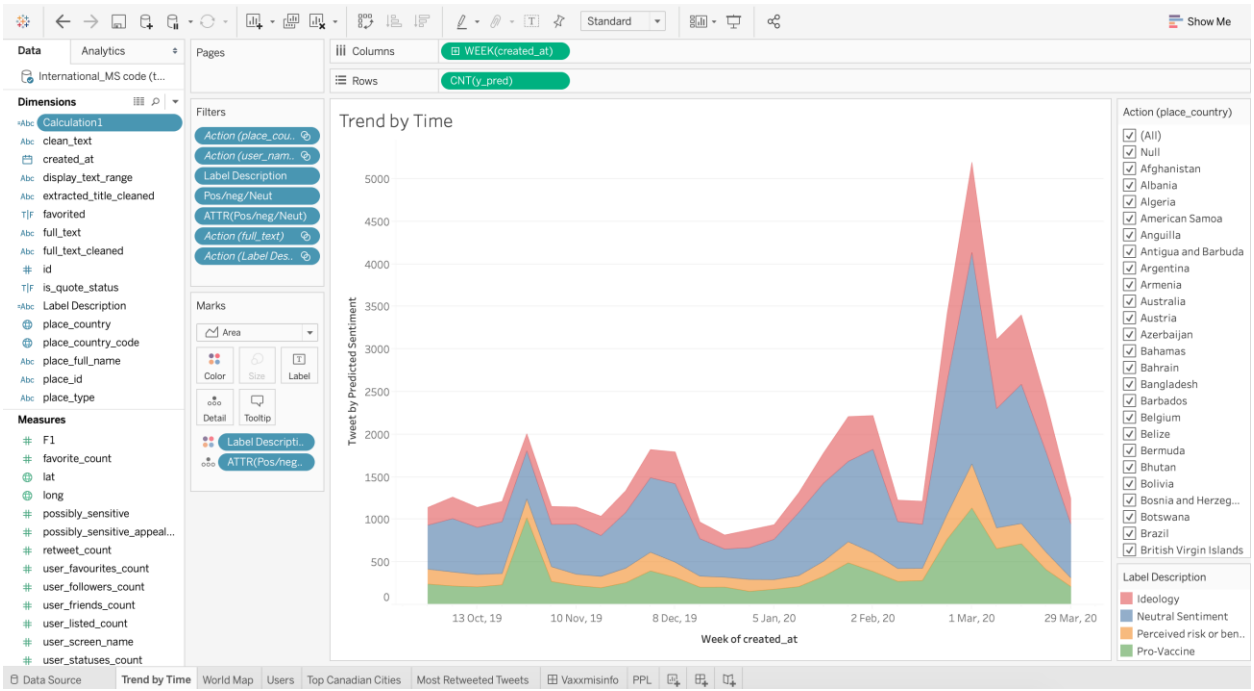
Pos/neg/Neut	`IF`
	`[y_pred] = 0 THEN 'Positive'`
	`ELSEIF`
	`[y_pred] = 1 THEN 'Neutral'`
	`ELSEIF`
	`[y_pred] = 2 THEN 'Negative'`

	ELSEIF [y_pred] = 3 THEN 'Negative' ELSEIF [y_pred] = 4 THEN 'Neutral' END
Label Description	IF [y_pred] = 0 THEN 'Pro-Vaccine' ELSEIF [y_pred] = 1 THEN 'Neutral Sentiment' ELSEIF [y_pred] = 2 THEN 'Perceived risk or benefit of getting vaccinated ' ELSEIF [y_pred] = 3 THEN 'Ideology' ELSEIF [y_pred] = 4 THEN 'Other' END

v. Sheets: Trends by time

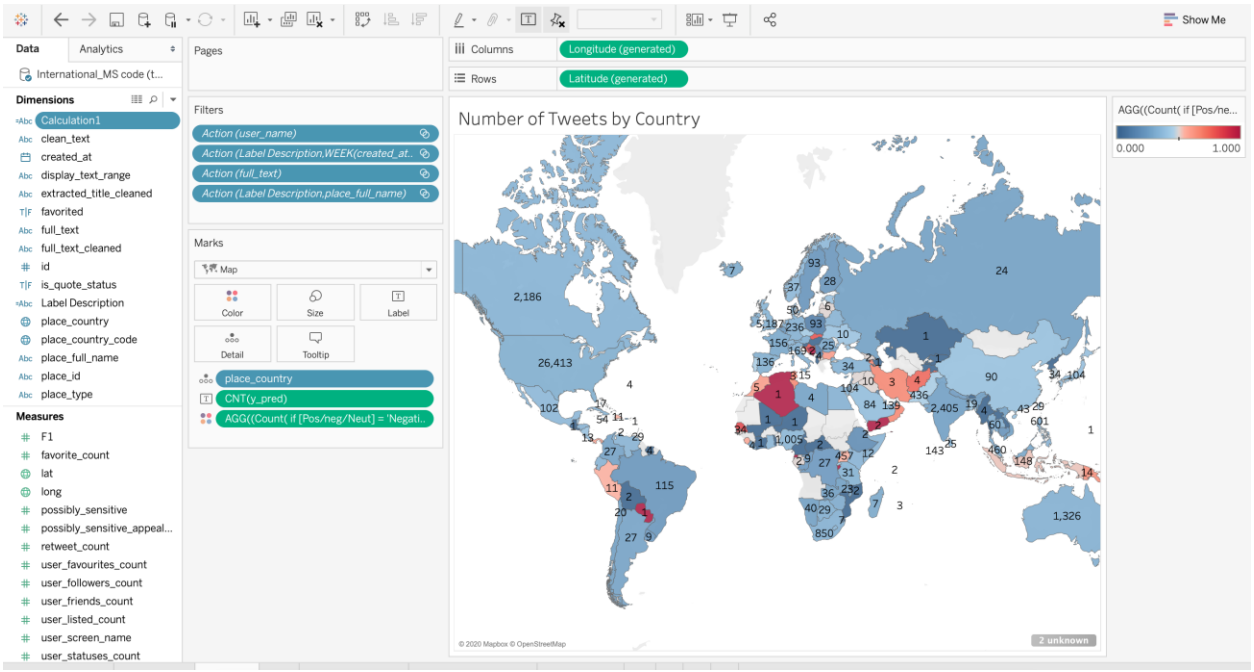
The dashboard provided as part of this project already has this pre-built, however for whatever reason if it has to be recreated, the parameters listed in the table below and depicted in the screenshot can be used to recreate it. You may ignore the 'Action' filters as those are created within the final dashboard itself

vi. Sheets: Trends by time



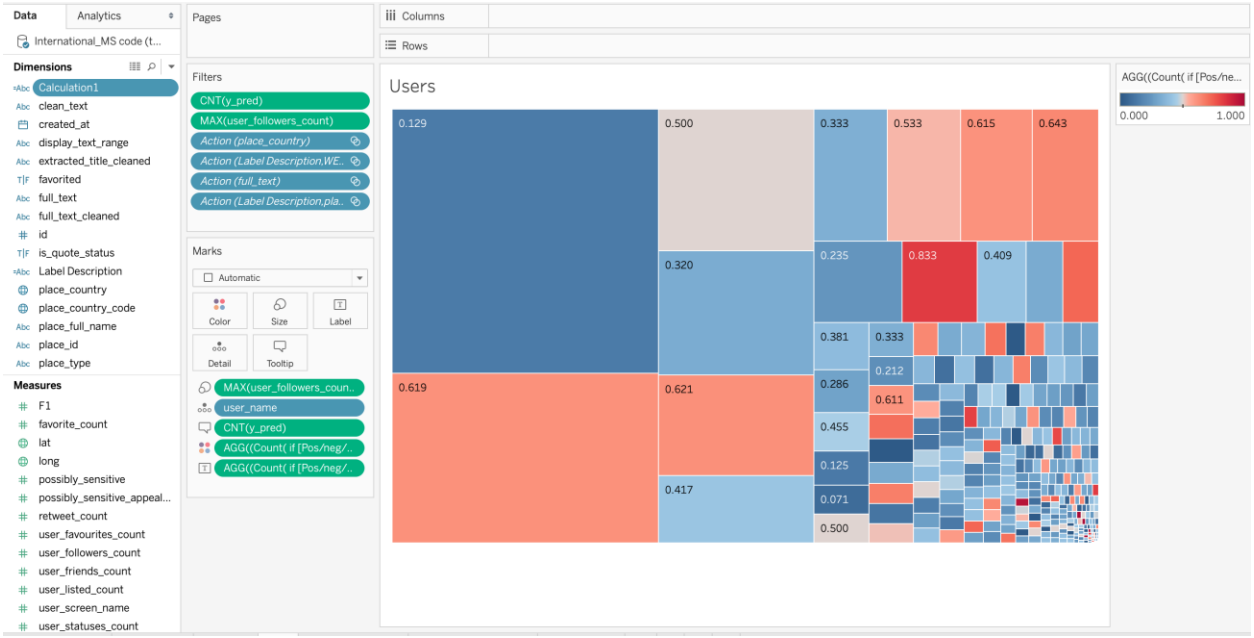
Field	Description
Columns	WEEK(*created_at*)
Rows	CNT(y_pred)
Filters	- Label Description - Pos/neg/Neut - ATTR([Pos/neg/Neut])
Marks	- **Type**: Area - **Colour**: Label Description - **Detail**: ATTR([Pos/neg/Neut])

vii. Sheets: Number of Tweets by Country



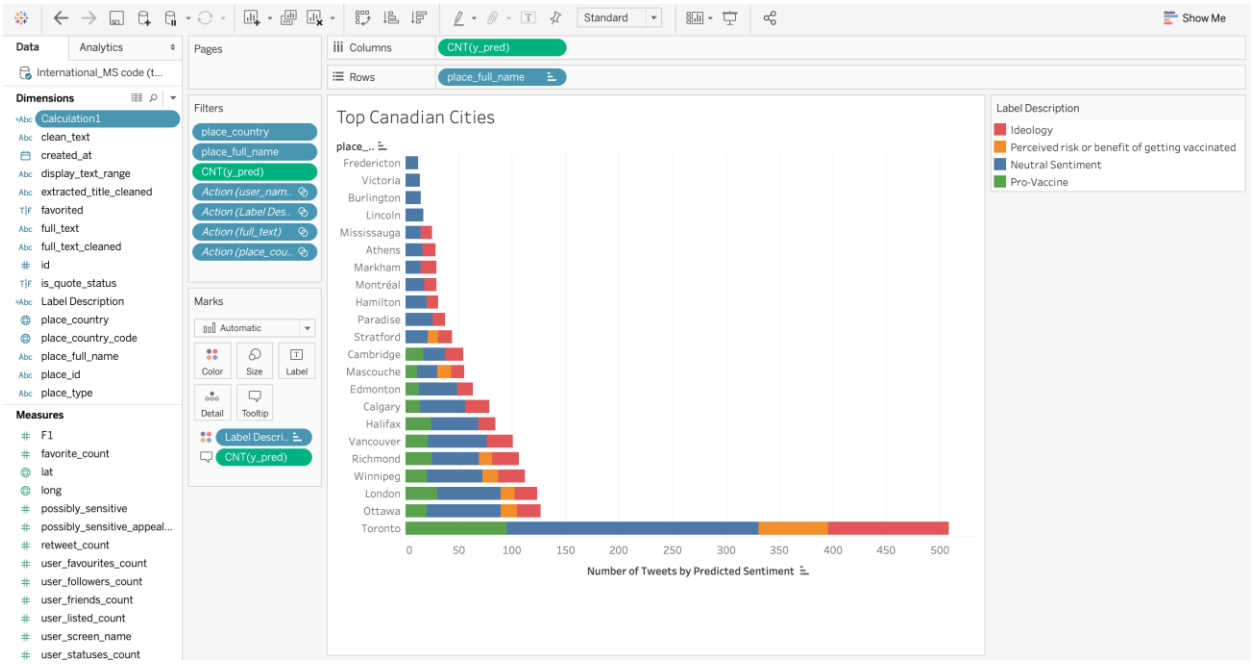
Field	Description
Columns	Longitude (generated)
Rows	Latitude (generated)
Filters	N/A
Marks	- **Type: Map** - **Label**: CNT(y_pred) - **Colour**: AGG((Count(if [Pos/neg/Neut] = 'Negative' then [y_pred] end) / count(y_pred)))

viii. Sheets: Users



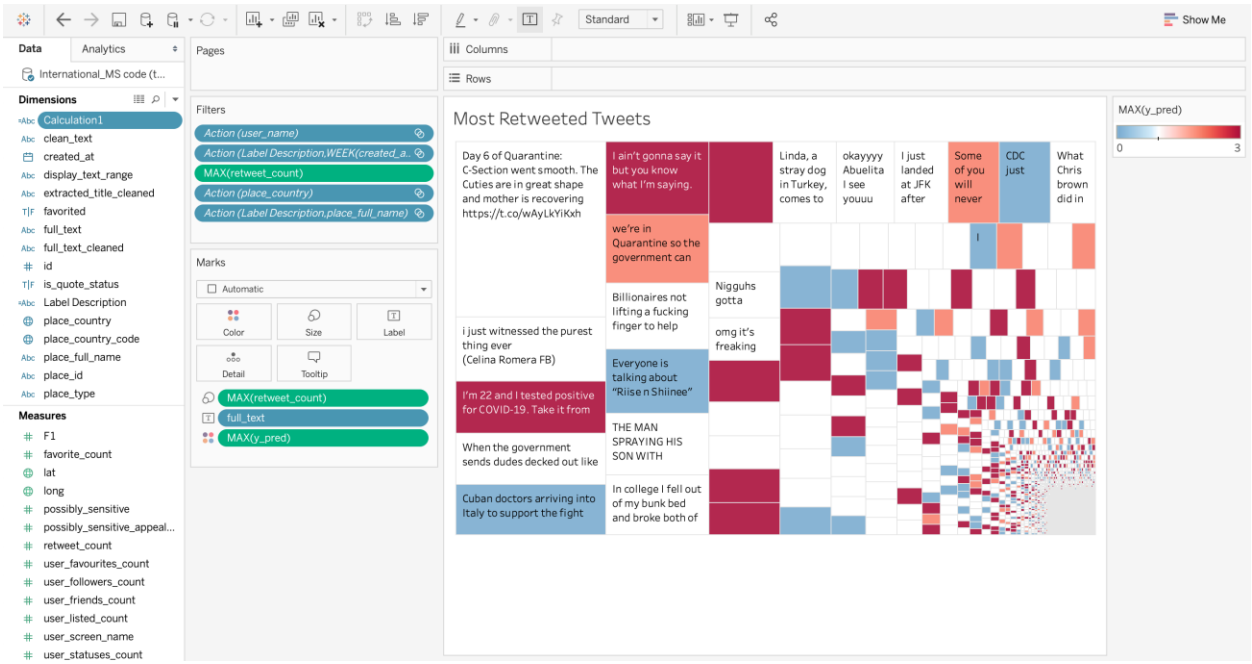
Field	Description
Columns	N/A
Rows	N/A
Filters	- CNT(y_pred) >= 10 - MAX(user_followers_count)>= 5
Marks	- **Type**: Automatic - **Size**: MAX(user_followers_count) - **Detail**: user_name (hidden / not displayed) - **Colour**: AGG((Count(if [Pos/neg/Neut] = 'Negative' then [y_pred] end) / count(y_pred))) - **Label**: AGG((Count(if [Pos/neg/Neut] = 'Negative' then [y_pred] end) / count(y_pred)))

ix. Sheet: Most Retweeted Tweets



Field	Description
Columns	N/A
Rows	N/A
Filters	- MAX(retweet_count) >=1
Marks	- **Size**: MAX(retweet_count) - **Label**: full_text - **Colour**: Max(y_pred)

x. Sheet: Top Canadian Tweets



Field	Description
Columns	CNT(y_pred)
Rows	Place_full_name
Filters	- Place_country = Canada - Place_full_name - CNT(y_pred) >=10
Marks	- Type: Automatic