COVID-19 Vaccines Related Tweets Sentiment Analysis and Relationship with Vaccination Levels

Fantastic Four:
Sharlene Chen
Jing Yang
Luqi Cai
Xueying Hu

April 28th, 2022

Agenda

1. Introduction
2. Data Preprocessing
3. Sentiment Analysis
4. EDA Findings
5. ML Modeling
6. Conclusion

Agenda

1 Introduction

2 Data Preprocessing

3 Sentiment Analysis

4 EDA Findings

5 ML Modeling

6 Conclusion

Introduction: Objective and Rationale

Objective:

Explore the sentiments of the tweets and the relationships between sentiments and vaccination rates

Rationale:

DataSet1: Vaccination Tweets

- Source: Kaggle
- Numbers: 200k+ records with 16 attributes
- Contents: **vaccine related tweets** and user information

WordCloud

Sentiment Analysis

- **Analyze the sentiment of tweets** in each country and month based on the NLP techniques

Simple Sentiment Analysis

NRC Sentiment Analysis

DataSet2: Vaccination Progress

- Source: Kaggle
- Numbers: 80k+ records with 16 attributes
- Contents: **daily vaccination progress** at the country level

Merged Dataset

Machine Learning Model

Linear Regression

Random Forest

KNN

- **Link tweet sentiments to the progress of vaccination**
- Whether the sentiment informs anything about progress of vaccinations in different countries

Agenda

Data Preprocessing: Dataset1 for Vaccination Tweets

Initial Cleaning

1. Key attributes **extraction**: 6 attributes

user_location	date	text	hashtag	retweets	favorites
LaCrescenta-Montrose, CA	2020-12-20 06:06:44	They promote their Vaccines...	['PfizerBioNTech']	0	0

2. User_locations standardization: **Google API**

Emoji / Non-English languages / Non-location words

user_location
Punjab, Pakistan
On a boat
world
กำแพงมีหู ประตูมีตา
🏡 🇺🇸 ✈️ 🇭🇰 🇬🇧

location
Pakistan
NaN
NaN
India
NaN

3. Time standardization: **Timestamp**

date
2020-12-20 06:06:44

year_month
(2020, 12)

Text Preprocessing

1. Remove emoji
2. Remove html tags / url
3. Fix misspelled words
4. Remove contractions
5. Tokenize each tweet
6. Remove stopwords
7. Remove punctuation
8. Lemmatization

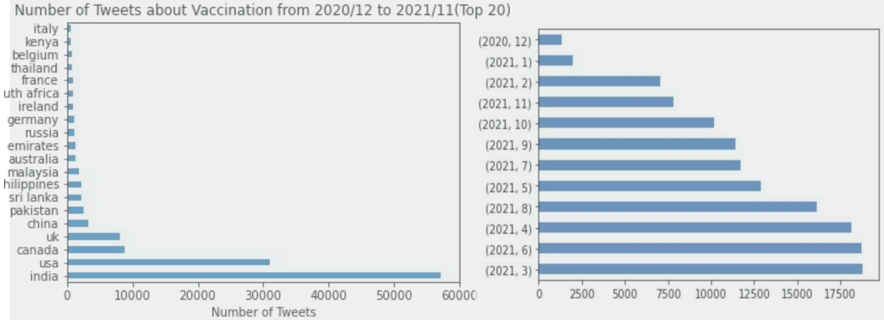
text
“"Facts are immutable, Senator, even when you're not ethically sturdy enough to acknowledge them. (1) You were born i_”

cleaned_text
['fact','immutable','senator','even','ethical','sturdy','enough','acknowledge','born']

Dataset1 Description and WordCloud

Dataset1 Description

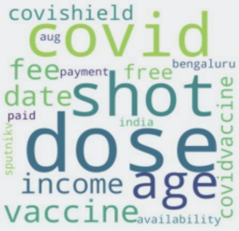
- **136k+** pieces of data after preprocessing
- Location: 87 countries in total, **42% India, 23% US**
- Year_month: 2020.12-2021.11, **67%** in 2021.3-5
  - Increased with time



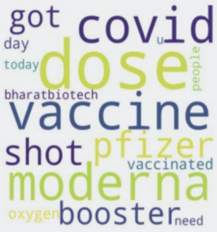
- Text: for further sentimental analysis
- Hashtag:
  [Moderna] &[Covaxin] & [Sinopharm] & [Pfizer]: 84%
- Retweets & Favorites: 0 make up 75%, for further **weighted** analysis

WordCloud for Topwords

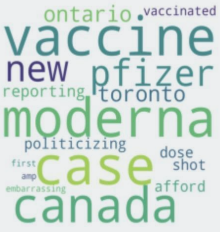
- For top **5** countries in terms of the number of tweets
  - Tweets all focus on the covid and vaccinations
  - Developing countries: focused on more **fee** and **availability**
  - Developed country: focused more on **different brands of vaccinations**



India

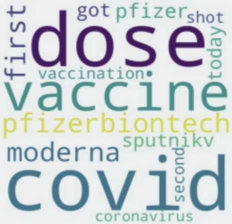


The U.S.

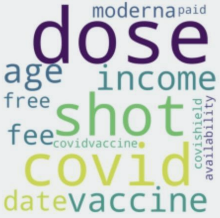


Canada

- For **2** different periods of time



2020.12-2021.6



2021.7-2021.11

- People focused more on the **age**

Weighted Method for Retweets & Favorites

Weighted Method for Sentiment Analysis

1. Using 'retweets' and dropping 'favorites'

- retweets and favorites have **similar** meanings:
- other users agree with the original tweets
- retweets might have **stronger emotional connotations** compared to favorites

2. Proportional vs **Logarithm:**

- If **proportional**

- The tweets **with more number of retweets** (more than several thousands) will dominate the sentiment and **exaggerate the effects**
- Having many retweets may be because those users have many followers

3. Specific Formula

Weight = log(n+1+1)= log(n+2)

- **n** is the **number of retweets**
- **1** represents the user who **posted the tweet**
- 1 is the number, for **avoiding the situation** that if there is no retweet, then the weight would be **log(1) = 0**
- Combined the texts with their weight for same country and year_month

Quick Overview for Weighted_text

	country	year_month	weighted_text
0	Albania	(2021, 1)	[['vaccine', 0.6931471805599453], ['refrigerat...
1	Albania	(2021, 2)	[['serbia', 0.6931471805599453], ['donate', 0....
2	Albania	(2021, 3)	[['pfizerbiontech', 0.6931471805599453], ['pos...
3	Albania	(2021, 4)	[['butantan', 0.6931471805599453], ['institute...
4	Albania	(2021, 5)	[['prime', 1.6094379124341003], ['minister', 1...
...	...	...	...
894	Zimbabwe	(2021, 5)	[['vaccine', 0.6931471805599453], ['must', 0.6...
895	Zimbabwe	(2021, 6)	[['sinovac', 0.6931471805599453], ['vaccine', ...
896	Zimbabwe	(2021, 7)	[['vaccine', 0.6931471805599453], ['people', 0...
897	Zimbabwe	(2021, 8)	[['sight', 1.6094379124341003], ['best', 1.609...
898	Zimbabwe	(2021, 9)	[['zimbabwe', 2.302585092994046], ['cocain', 2...

```
"[['vaccine', 0.6931471805599453], ['refrigerator', 0.6931471805599453], ['health', 0.69314
71805599453], ['minister', 0.6931471805599453], ['gmanastirliu', 0.6931471805599453], ['sto
re', 0.6931471805599453], ['pfizerbiontech', 0.6931471805599453], ['jaw', 0.693147180559945
3]]"
```

Data Preprocessing: Dataset 2 for Vaccinations Progress

Initial Cleaning

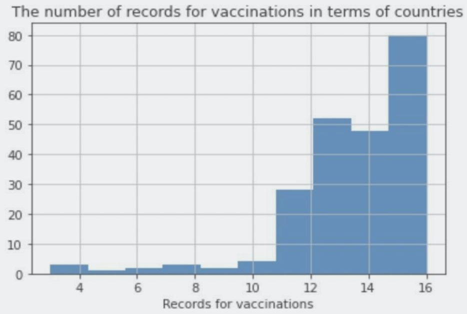
1. Key attributes extraction: 6 attributes
2. Same time standardization: Timestamp
3. **Aggregation** for both country and date:

country	year_month	monthly_vaccinations	monthly_vaccinations_per_million
Afghanistan	(2021, 2)	8202.0	204.0
	(2021, 3)	85894.0	2154.0
	(2021, 4)	219606.0	5511.0
	(2021, 5)	285838.0	7171.0
	(2021, 6)	242899.0	6097.0

Merge 2 Dataset

Dataset2 Description

- **3018** pieces of data after preprocessing
- Country: 223 countries in total, **95% countries** have more than **10 months of records for vaccinations**



- Year_month: 2020.12-2022.3
- Monthly_vaccinations/ per_million: 3018 non-null
- People_vaccinated /_per_hundred: 2834 non-null

Agenda

1 Introduction

2 Data Preprocessing

3 **Sentiment Analysis**

4 EDA Findings

5 ML Modeling

6 Conclusion

Simple Sentiment Analysis

Steps for simple sentiment analysis

- Hu and Liu's sentiment analysis lexicon
- + weight of each word
- positive_emotion_percentage = pos/(pos+neg)
- negative_emotion_percentage = neg/(pos+neg)
- positive_emotion = pos/number of all words
- negative_emotion = neg/number of all words

```
pos_per = []
neg_per = []
for row in total.weighted_text:
    pos = 0
    neg = 0
    if type(row) == float:
        pos_per.append(0)
        neg_per.append(0)
        continue
    for each in ast.literal_eval(row):
        if each[0] in positive_words:
            pos += each[1]
        elif each[0] in negative_words:
            neg += each[1]
    all = neg + pos
```

Result for simple sentiment analysis

```
"[['vaccine', 0.6931471805599453], ['refrigerator', 0.6931471805599453], ['health', 0.69314
71805599453], ['minister', 0.6931471805599453], ['gmanastirliu', 0.6931471805599453], ['sto
re', 0.6931471805599453], ['pfizerbiontech', 0.6931471805599453], ['jaw', 0.693147180559945
3]]"
```

positive_emotion_percentage	negative_emotion_percentage	positive_emotion	negative_emotion
54.536643	45.463357	2.749019	2.291664
69.897000	30.103000	1.477365	0.636266
48.960058	51.039942	3.139462	3.272830
53.724357	46.275643	8.166725	7.034434
0.000000	100.000000	0.000000	5.882353

NRC Sentiment Analysis

Steps for NRC sentiment analysis

- NRC Emotion Lexicon v0.92
- As each word's relation to each emotion corresponds to a binary result, we calculated the emotions for weighted texts using following steps:
  - For each word in the texts, calculate **word weight * emotion binary value / total text weight**
  - Append each emotion's percentage in texts into dictionary
  - Create table and merge with the original dataset

Results for NRC sentiment analysis

"[['vaccine', 0.6931471805599453], ['refrigerator', 0.6931471805599453], ['health', 0.6931471805599453], ['minister', 0.6931471805599453], ['gmanastirliu', 0.6931471805599453], ['store', 0.6931471805599453], ['pfizerbiontech', 0.6931471805599453], ['jaw', 0.6931471805599453]]"

```python
for row in total.weighted_text:
    new = emotion_count.copy()
    if type(row) == float:
        emo.append(new)
        continue
    to = get_2nd_weight(ast.literal_eval(row))
    for word in ast.literal_eval(row):
        if emotion_dict.get(word[0]):
            for emotion in emotion_dict.get(word[0]):
                new[emotion] += word[1]/to*100
    emo.append(new)
emo
```

	disgust	surprise	fear	positive	joy	anger	negative	sadness	anticipation	trust
0	0.000000	0.000000	0.000000	25.000000	0.000000	0.000000	0.000000	0.000000	12.500000	0.000000
1	0.000000	0.000000	0.000000	13.513514	0.000000	0.000000	0.000000	0.000000	0.000000	5.405405
2	3.144267	1.886560	7.914098	17.975754	0.628853	4.401974	8.542951	4.401974	8.542951	5.659681
3	3.508772	0.000000	5.263158	17.543860	0.000000	3.508772	5.263158	5.263158	5.263158	8.771930
4	0.000000	0.000000	0.000000	22.222222	0.000000	0.000000	11.111111	0.000000	0.000000	0.000000

Vader Sentiment Analysis

Steps for Vader sentiment analysis

- vaderSentiment library
- Vader looks at each sentence's emotion with the ability to identify inner logic, so we constructed weight for each sentence
  - For each sentence in the texts, calculate:
  - **sentence weight * emotion value / (length of sentence * total text weight)**
  - Sum each sentence emotions' for each month and country, append value into dictionary
  - Create table and merge with the original dataset

```
[['#Serbia donates 4.860 #PfizerBiontech jabs to #NorthMacedonia \n\n_~_https://t.co/rTF9ct
pPUg https://t.co/WryQIOQB4C',
  0.6931471805599453],
 ['#Jerusalem reportedly agreed to buy #Russian #SputnikV vaccine to free jailed Israeli in
#Syria 占쏙옙 占쏙옙 占쏙옙https://t.co/IeRC6iB5An',
  1.3862943611198906],
 ['#Hungary on Wednesday started using #COVID-19 vaccines produced by the Chinese laborator
y #Sinopharm, becoming the_ https://t.co/Udc85Wuzq6',
  0.6931471805599453]]
```

Results for Vader sentiment analysis

```
for i in range(len(texts)):
    sentences = sent_tokenize(texts[i][0])
    text_weight = texts[i][1]

    vs = analyzer.polarity_scores(sentences)
    pos+=vs['pos']*text_weight/(len(sentences))
    compound+=vs['compound']*text_weight/(len(sentences))
    neu+=vs['neu']*text_weight/(len(sentences))
    neg+=vs['neg']*text_weight/(len(sentences))
return pos,neg,neu,compound
```

Vader pos	Vader neg	Vader neu	Vader com
0.000000	0.000000	1.000000	0.000000
0.119500	0.071000	0.809500	0.148000
0.043696	0.017581	0.842476	0.060754
0.036143	0.068214	0.752786	-0.064843
0.174000	0.000000	0.826000	0.378600

Simple Sentiment Analysis

- Hu and Liu's sentiment analysis lexicon
- \+ weight of each word
- positive_emotion_percentage = pos/(pos+neg)
- negative_emotion_percentage = neg/(pos+neg)
- positive_emotion = pos/number of all words
- negative_emotion = neg/number of all words

"[['vaccine', 0.6931471805599453], ['refrigerator', 0.6931471805599453], ['health', 0.6931471805599453], ['minister', 0.6931471805599453], ['gmanastirliu', 0.6931471805599453], ['store', 0.6931471805599453], ['pfizerbiontech', 0.6931471805599453], ['jaw', 0.6931471805599453]]"

positive_emotion_percentage	negative_emotion_percentage	positive_emotion	negative_emotion
54.536643	45.463357	2.749019	2.291664
69.897000	30.103000	1.477365	0.636266
48.960058	51.039942	3.139462	3.272830
53.724357	46.275643	8.166725	7.034434
0.000000	100.000000	0.000000	5.882353

```python
pos_per = []
neg_per = []
for row in total.weighted_text:
    pos = 0
    neg = 0
    if type(row) == float:
        pos_per.append(0)
        neg_per.append(0)
        continue
    for each in ast.literal_eval(row):
        if each[0] in positive_words:
            pos += each[1]
        elif each[0] in negative_words:
            neg += each[1]
    all = neg + pos
```

NRC Sentiment Analysis

- NRC Emotion Lexicon v0.92
- As each word’s relation with a kind of emotion corresponds to a binary results, so for weighted texts, we calculated the emotions using following steps:
    - For each word in the texts, calculate **word weight * emotion binary value / total text weight**
    - Append each emotion’s percentage in texts into dictionary
    - Create table and merge with the original dataset

"[['vaccine', 0.6931471805599453], ['refrigerator', 0.6931471805599453], ['health', 0.6931471805599453], ['minister', 0.6931471805599453], ['gmanastirliu', 0.6931471805599453], ['store', 0.6931471805599453], ['pfizerbiontech', 0.6931471805599453], ['jaw', 0.6931471805599453]]"

```
for row in total.weighted_text:
    new = emotion_count.copy()
    if type(row) == float:
        emo.append(new)
        continue
    to = get_2nd_weight(ast.literal_eval(row))
    for word in ast.literal_eval(row):
        if emotion_dict.get(word[0]):
            for emotion in emotion_dict.get(word[0]):
                new[emotion] += word[1]/to*100
    emo.append(new)
emo
```

	disgust	surprise	fear	positive	joy	anger	negative	sadness	anticipation	trust
0	0.000000	0.000000	0.000000	25.000000	0.000000	0.000000	0.000000	0.000000	12.500000	0.000000
1	0.000000	0.000000	0.000000	13.513514	0.000000	0.000000	0.000000	0.000000	0.000000	5.405405
2	3.144267	1.886560	7.914098	17.975754	0.628853	4.401974	8.542951	4.401974	8.542951	5.659681
3	3.508772	0.000000	5.263158	17.543860	0.000000	3.508772	5.263158	5.263158	5.263158	8.771930
4	0.000000	0.000000	0.000000	22.222222	0.000000	0.000000	11.111111	0.000000	0.000000	0.000000

Agenda

1 Introduction

2 Data Preprocessing

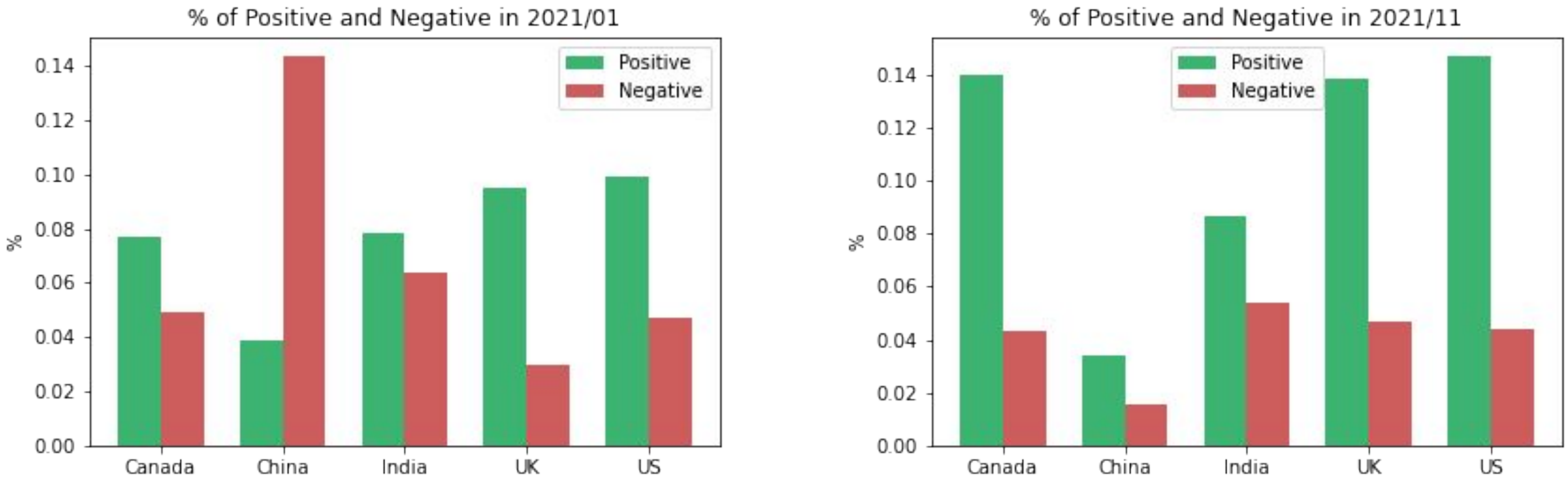
3 Sentiment Analysis

4 **EDA Findings**

5 ML Modeling

6 Conclusion

Bar Charts for positive and negative emotions in top 5 countries



Selected Top 5 countries to show the change of positive and negative emotions on tweets from 2021/01 to 2021/11

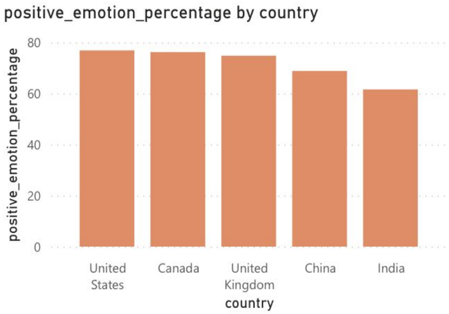
Implications:

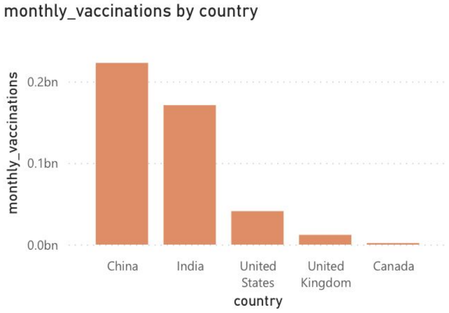
Comparing across the timeline, we can find that:

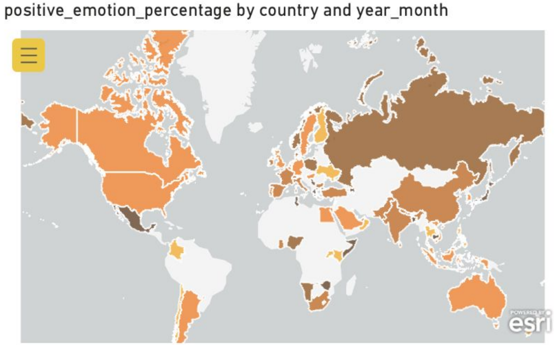
- The weights of **positive emotions** were **increasing**
- The weight of positive emotions in China stayed the same but negative emotions decreased sharply

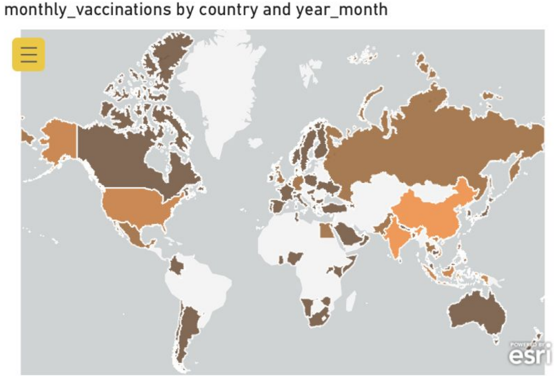
Heatmap for positive emotions and vaccinations through countries

Utilized Power Bi to compare the relationship between emotions derived from tweets and monthly vaccination counts in each country









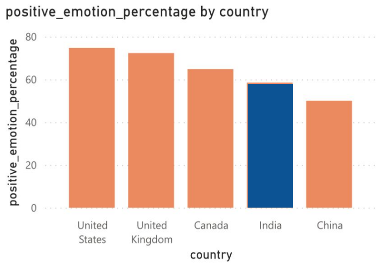
Upper graph: Positive emotion in tweets

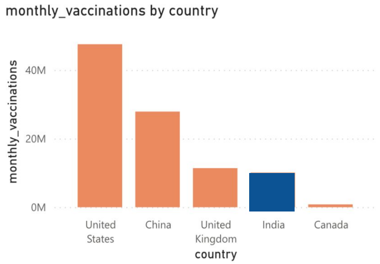
Lower graph: Monthly vaccination counts

Implications:

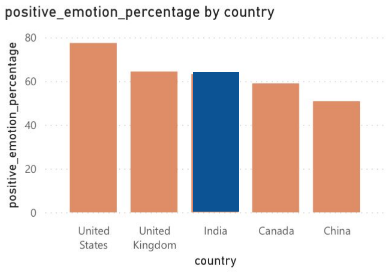
- No obvious relationship between monthly vaccination counts and positive emotion percentage
- Geographical differences in positive emotion percentage

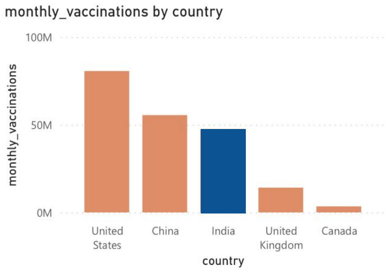
Heatmap for positive emotions and vaccinations for top 5 countries



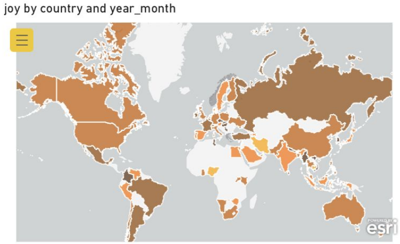


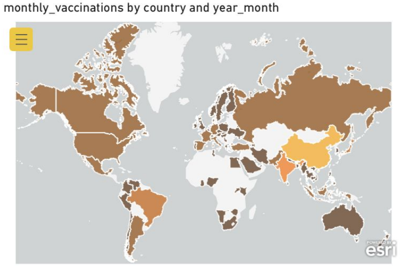
2021/02





2021/03



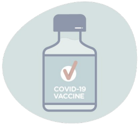


Joy Heatmap and Monthly Vaccination Number Heatmap

Implications:

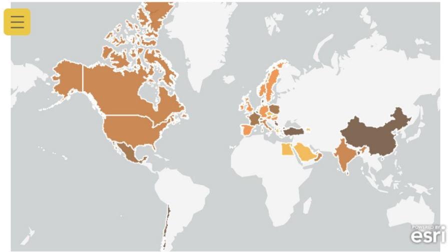
Comparing across the timeline, we can find:

- Relation between monthly vaccination number surge within one month and positive emotion percentage
- Relation between joy and monthly vaccination number
- Will verify this relation using Machine Learning

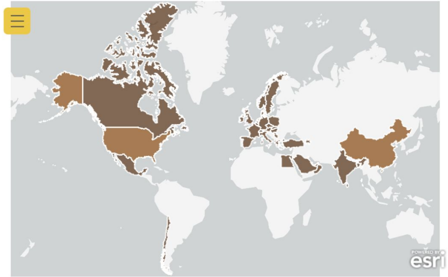


Heatmap for positive emotions and vaccinations through timeline

positive_emotion_percentage by country and year_month

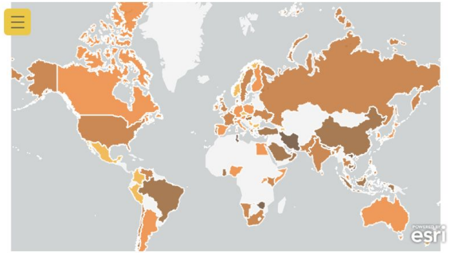


monthly_vaccinations by country and year_month

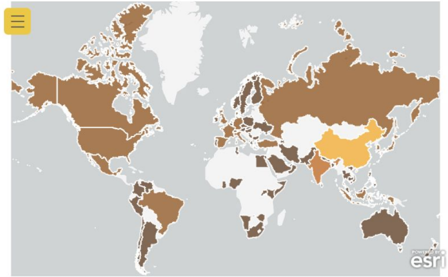


2021/01

positive_emotion_percentage by country and year_month

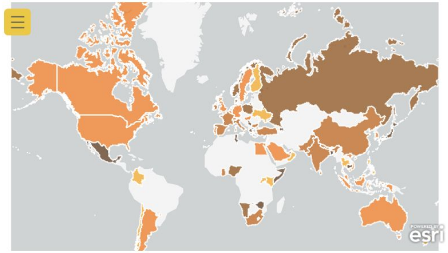


monthly_vaccinations by country and year_month

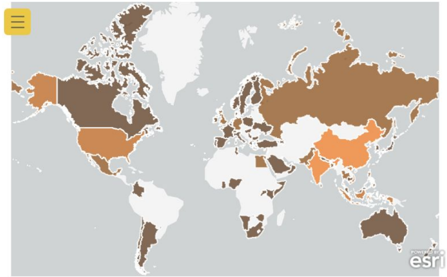


2021/06

positive_emotion_percentage by country and year_month



monthly_vaccinations by country and year_month



2021/11

Implications:

Comparing across the timeline, we can find that:

- Positive emotion is rising across countries

Agenda

1 Introduction

2 Data Preprocessing

3 Sentiment Analysis

4 EDA Findings

5 **ML Modeling**

6 Conclusion

Method 1: Linear Regression

Steps to Create Regression

- Independent variables: the **emotions**
- Dependent variables: **monthly vaccination levels** in percentage (not accumulated)
- Regression will be indifferent to country and time
- Splitted into a **train** and a **test** set

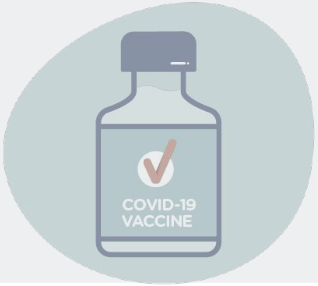
2-Emotion Results

Dep. Variable:	daily_vaccinations_per	**R-squared:**	0.043
Model:	OLS	**Adj. R-squared:**	0.040
Method:	Least Squares	**F-statistic:**	16.53
Date:	Fri, 22 Apr 2022	**Prob (F-statistic):**	9.45e-08
Time:	15:46:28	**Log-Likelihood:**	-2718.7
No. Observations:	741	**AIC:**	5443.
Df Residuals:	738	**BIC:**	5457.
Df Model:	2		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	7.8091	0.693	11.271	0.000	6.449	9.169
positive_emotion	0.3211	0.059	5.438	0.000	0.205	0.437
negative_emotion	0.1404	0.109	1.285	0.199	-0.074	0.355

Omnibus:	82.344	**Durbin-Watson:**	0.633
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	108.402
Skew:	0.928	**Prob(JB):**	2.89e-24
Kurtosis:	3.255	**Cond. No.**	21.7

- **Positive emotion** has a positive coefficient and is statistically significant
- (also matches with previous observations!)



8-Emotion Results

Dep. Variable:	daily_vaccinations_per	**R-squared:**	0.030
Model:	OLS	**Adj. R-squared:**	0.019
Method:	Least Squares	**F-statistic:**	2.809
Date:	Mon, 25 Apr 2022	**Prob (F-statistic):**	0.00451
Time:	21:51:20	**Log-Likelihood:**	-2723.7
No. Observations:	741	**AIC:**	5465.
Df Residuals:	732	**BIC:**	5507.
Df Model:	8		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	10.4806	0.777	13.494	0.000	8.956	12.005
disgust	0.6124	0.210	2.921	0.004	0.201	1.024
fear	0.1752	0.190	0.922	0.357	-0.198	0.548
joy	0.6315	0.195	3.238	0.001	0.249	1.014
surprise	-0.1087	0.200	-0.544	0.586	-0.501	0.283
anger	-0.0082	0.228	-0.036	0.971	-0.455	0.439
trust	-0.2016	0.125	-1.614	0.107	-0.447	0.044
sadness	-0.1101	0.221	-0.498	0.618	-0.544	0.324
anticipation	-0.2063	0.133	-1.551	0.121	-0.467	0.055

Omnibus:	77.459	**Durbin-Watson:**	0.615
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	100.766
Skew:	0.900	**Prob(JB):**	1.32e-22
Kurtosis:	3.152	**Cond. No.**	23.5

Model with all emotions fitted

Implications:

- Overall, the **R-squared score is rather low**
- Many other factors are also into play with vaccination levels

Joy

Rather intuitive, the more joy, the more people accept vaccinations.

Disgust

If we also **consider the impacts of policies** that could generate disgust, The more disgust may mean more restrictions but could lead to higher vaccinations rates).

Method 1: Linear Regression

Vader Sentiment Results

Dep. Variable:	monthly_vaccinations_per	R-squared:	0.004
Model:	OLS	Adj. R-squared:	-0.001
Method:	Least Squares	F-statistic:	0.8386
Date:	Fri, 29 Apr 2022	Prob (F-statistic):	0.501
Time:	15:09:19	Log-Likelihood:	-3007.6
No. Observations:	812	AIC:	6025.
Df Residuals:	807	BIC:	6049.
Df Model:	4		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	14.4921	2.108	6.874	0.000	10.354	18.630
vader_pos	-9.6429	22.553	-0.428	0.669	-53.913	34.627
vader_neg	-1.4961	26.900	-0.056	0.956	-54.299	51.307
vader_neu	-4.1600	2.631	-1.581	0.114	-9.324	1.004
vader_com	4.0555	9.345	0.434	0.664	-14.287	22.398

Omnibus:	84.546	Durbin-Watson:	0.585
Prob(Omnibus):	0.000	Jarque-Bera (JB):	110.788
Skew:	0.904	Prob(JB):	8.77e-25
Kurtosis:	3.048	Cond. No.	124.

	Simple Sentiments	NRC Sentiments	Vader Sentiments
Root Mean Squared Error	48.81	53.64	46.54
Most significant variables	Positive	Disgust, Fear, Joy	Vader Neutral

Implications:

- Vader Sentiment results **doesn't** give statistically significant variables, the most significant only has p-value of 0.11.
- **High multicollinearity** is probably present, because the scores depend on each other.
- More data and other methods of fitting is needed to confirm the relationship between sentiments and vaccination levels.

Method 2: Random Forest

Steps to create the algorithm

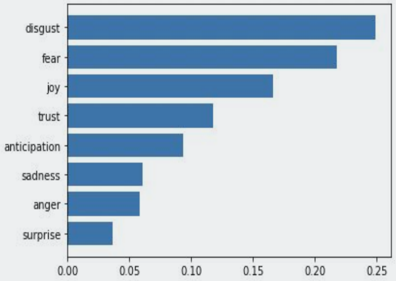
- Same dataset that from linear regression
- Using the **random forest algorithm** to fit the model
- Using Grid Search CV to test out different combinations of parameters to tune the model.

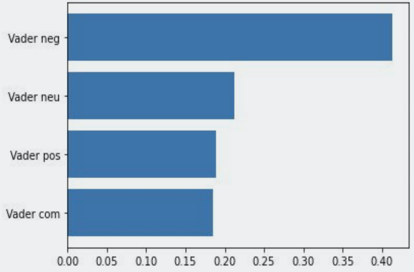
```
import numpy as np
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import GridSearchCV
parameters = {
      'n_estimators':(4, 6, 8), #the number of trees
      'max_depth':(3,4,5,6,10),
      'min_samples_split': (10, 50, 100), # minimum number of samples required to split an internal node
      'min_samples_leaf': (10,50,100) # the minimum number of samples required to be at a leaf node.
}

model = GridSearchCV(RandomForestRegressor(),parameters,cv=3)
train_np = np.array(train[['vader_pos','vader_neg','vader_neu','vader_com']])
train_y = np.array( train['monthly_vaccinations_per'])
model.fit(train_np, np.ravel(train_y))
# sk_m.r2_score(test["daily_vaccinations_per"], preds)
model.best_score_, model.best_params_
```

```
(0.008348253214933784,
 {'max_depth': 6,
  'min_samples_leaf': 50,
  'min_samples_split': 50,
  'n_estimators': 6})
```

With GridCV, we can see hypertuned parameters for Vader Sentiment random forest:
Max depth: 6
Min sample leaf: 50
Min sample split: 50
N Estimators: 6

Preliminary Results





- Using the same parameters, we saw that the best **RMSE is about 46.48** from Vader Sentiments, though it varies a bit when ran multiple times.
- The important features for NRC sentiments include **disgust, fear and joy which match results from linear regression**.
- The important features for Vader Sentiments are negative emotions.

Method 3: k-Nearest Neighbors

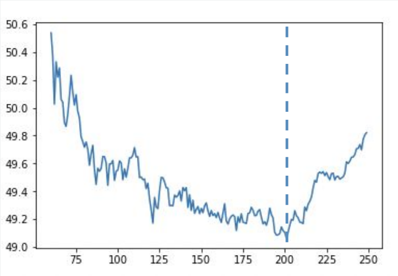
Steps to create the algorithm

- Same dataset that from linear regression
- Using K-nearest neighbors (specifically K Neighbors Regressor to fit the model.

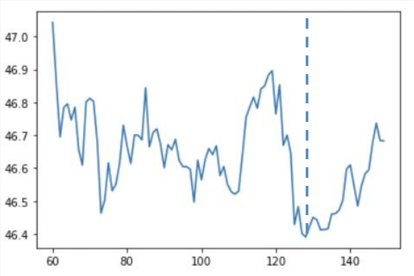
```python
import sklearn.neighbors as sk_n
import sklearn.metrics as sk_m

scores = []
for n in range(60,150):
# Fit a k-NN model on the training set
    knn = sk_n.KNeighborsRegressor(n_neighbors=n)
    knn.fit(train[emotions], train['daily_vaccinations_per'])

# Make predictions on the test set
    preds = knn.predict(test[emotions])

# Find the R-squared on the test set
    score = sk_m.r2_score(test["daily_vaccinations_per"], preds)
    scores.append(score)
```

Preliminary Results

Then, we would **test out all possible k values** to pick the best k, as shown in the graph on the right



The **best k in this case is 201**, and the RMSE is 49.06.



The **best k in this case is 128**, and the RMSE is 46.39.

Comparison Between 3 Methods

	Linear Regression	Random Forest	kNN
Best Setup	Vader Sentiments Regression	Vader Sentiments Random Forest max depth: 6; min sample leaf: 50 min sample split: 50; n estimators: 6	Vader Sentiments k=128
Best Root Mean Squared Error	46.54	46.48	46.39
Most Significant Variables	2 sentiment: Positive 8 sentiment: Disgust and Joy	8 sentiment: Disgust, Fear, Joy Vader: Negative	N/A

- **Vader Sentiments** gives the best results in all 3 methods.
- **Linear regression** can best tell us what emotions are more significant, but **kNN** performs best.
- **R-squared in general is quite low**, showing that there are many other factors impacting the dependent variable (vaccinated rates).
- The most significant emotions are **positive emotions** for 2 emotions case, or within the 8 emotions context, the most significant ones are **disgust and joy**.
- More evidences is needed to prove the relationship since results don't all align well.

Agenda

1 Introduction

2 Data Preprocessing

3 Sentiment Analysis

4 EDA Findings

5 ML Modelling

6 **Conclusion**

Conclusion

- Analysis show that **sentiments would not necessarily impact** vaccination rates
- Further data on policies within certain time period / country is needed to prove the relationship
- **Improvements**: include **topic analyses** to further identify certain topics that may help increase vaccination rates

Thanks for Listening!