ISYE 6740 – Spring 2021

Final Report

Sentiment Analysis of COVID-19 Vaccination Tweets

Jocelyn Wuici Chai (GT ID: 903656829)

Xiyue Lin (GT ID: 903662002)

1. Problem Statement

Coronavirus disease 2019 (COVID-19) is the disease caused by a new coronavirus called SARS-CoV-2. The World Health Organization (WHO) first learned of this new virus on December 31, 2019, following a report of a cluster of cases of 'viral pneumonia' in Wuhan, People's Republic of China. On March 11, 2020, the WHO officially declared it as a global pandemic. As of March 29, 2021, more than 127 million cases have been confirmed, with more than 2.78 million deaths, making it one of the deadliest pandemics in history [1]. Preventive measures to reduce the infection rate ("flatten the curve") include social distancing, wearing face coverings in public, ventilating indoor spaces, and most importantly, vaccination.

The first COVID-19 vaccine was granted approval by the United Kingdom's Medicines & Healthcare Products Regulatory (MHRA), later by the United States Food & Drug Administration (FDA) and in several other countries. As of now, several vaccines are being developed and distributed, including Pfizer-BioNTech, Moderna, Oxford-AstraZeneca, Johnson & Johnson and others. However, the public's attitude towards vaccination is extremely divided, especially in the United States. Hence, we will build a sentiment analysis prediction model to get a clearer understanding of people's opinions concerning vaccination in this country. We will be relying on a pre-trained Bidirectional Encoder Representations from Transformers (BERT) model to predict the sentiment of 'tweets', namely, discussions on one of the biggest social media platform, Twitter. Subsequently, we will explore the relationship between the vaccination progress and sentiments about vaccines as reflected in tweets throughout the different states. We will also undertake a detailed exploratory analysis by comparing the sentiments in each state towards different vaccines.

2. Data Source

We will be using a combination of three different datasets in our project. Firstly, we will use the 'Sentiment140' dataset, which contains 1.6 million tweets that has been pre-extracted for us using the Twitter API [2]. The dataset contains six columns, namely, `target`, `ids`, `date`, `flag`,

`user` and `text`. We will be using the column `target`, which represents the sentiment of the tweet, and `text`, which contains the actual tweet, to train our model. The sentiment of the tweet has two unique values where '0' represents a negative sentiment and '4' represents a positive sentiment.

The second dataset that we will be using is the 'All COVID-19 Vaccines Tweets' dataset from Kaggle [3]. While this dataset is updated regularly (almost daily), we will be using the Version 46 (last updated on March 29, 2021). The data was collected by filtering through tweets that have frequently used terms for the Pfizer/BioNTech, Sinopharm, Sinovac, Mordena, Oxford/Astra-Zeneca, Covaxin and Sputnik V vaccines respectively from December 12, 2020 until March 29, 2021. Out of the 16 columns, we will be only utilizing the `user_location`, `date` and `text` column. The `user_location` column specifies the user's location, the `date` column specifies the date of tweet and the `text` column specifies the actual content of the tweet. There are a total of 31036 tweets from all around the globe.

Lastly, we will also be making use of the 'United States Vaccination' dataset from Our World in Data's GitHub page [4]. The dataset relies on data that is updated daily by the United States Centers for Disease Control and Prevention. We will be using the version of this dataset that was last committed on March 29, 2021. It contains data from January 1, 2021 to March 29, 2021. Of the 14 columns, we are interested in the `date`, `location` and `total_vaccinations` columns accordingly. The *location* column contains the name of the state or federal entity, the *date* column contains the date of the observation and the `people_vaccinated` column contains data about the total number of people who received at least one vaccine dose. Note that this does not consider the people that have received their second dose.

3. Methodology

3.1 Data Pre-processing

The dataset we used, 'Sentiment140' dataset, contains 1.6 million tweets. In order to clean the dataset, we need to remove Twitter handles (*Fig.1*), URLs (*Fig.2*) along with punctuations, numbers, and special characters (*Fig.3*). While the hash key (i.e. #) was removed from the hashtags, word that came after the hash should be kept as it may have some sentimental value, especially after it has been separated appropriately if it was made up of one or more words. Then the tweets were all converted to lowercase (*Fig.4*).

```
Example of a Tweet:
happy #charitytuesday @theNSPCC @SparksCharity @SpeakingUpH4H

Twitter handles:
['@theNSPCC', '@SparksCharity', '@SpeakingUpH4H']

Results:
'happy #charitytuesday'
```

Fig 1

```
Example of a Tweet:
@switchfoot http://twitpic.com/2y1zl - Awww, that's a bummer.  You shoulda got David Carr of Third Day to do it. ;D

URL:
['http://twitpic.com/2y1zl']

Results:
'- Awww, that's a bummer.  You shoulda got David Carr of Third Day to do it. ;D'
```

Fig 2

```
Example of a Tweet:
@switchfoot http://twitpic.com/2y1zl - Awww, that's a bummer.  You shoulda got David Carr of Third Day to do it. ;D
happy #charitytuesday @theNSPCC @SparksCharity @SpeakingUpH4H

Punctuations, numbers, special characters:
['@', ':', '/', '/', '.', '/', '2', '1', '-', ',', "'", '.', '.', ';']
['#', '@', '@', '@', '4']

Results:
Awww thats a bummer You shoulda got David Carr of Third Day to do it D
happy charitytuesday
```

Fig 3

```
Example of a Tweet:
@switchfoot http://twitpic.com/2y1zl - Awww, that's a bummer.  You shoulda got David Carr of Third Day to do it. ;D

Results:
awww thats a bummer you shoulda got david carr of third day to do it d
```

Fig 4

Below we have examples of the before cleaning (*Fig.5*) and after cleaning (*Fig.6*).

```
Tweet 0: @switchfoot http://twitpic.com/2y1zl - Awww, that's a bummer.  You shoulda got David Carr of Third Day to do it. ;D
Tweet 1: is upset that he can't update his Facebook by texting it... and might cry as a result  School today also. Blah!
Tweet 2: @Kenichan I dived many times for the ball. Managed to save 50%  The rest go out of bounds
Tweet 3: my whole body feels itchy and like its on fire
Tweet 4: @nationwideclass no, it's not behaving at all. i'm mad. why am i here? because I can't see you all over there.
Tweet 5: @Kwesidei not the whole crew
Tweet 6: Need a hug
Tweet 7: @LOLTrish hey  long time no see! Yes.. Rains a bit ,only a bit  LOL , I'm fine thanks , how's you ?
Tweet 8: @Tatiana_K nope they didn't have it
Tweet 9: @twittera que me muera ?
Tweet 10: spring break in plain city... it's snowing
Tweet 11: I just re-pierced my ears
Tweet 12: @caregiving I couldn't bear to watch it.  And I thought the UA loss was embarrassing . . . . .
Tweet 13: @octolinz16 It it counts, idk why I did either. you never talk to me anymore
Tweet 14: @smarrison i would've been the first, but i didn't have a gun.    not really though, zac snyder's just a doucheclown.
Tweet 15: @iamjazzyfizzle I wish I got to watch it with you!! I miss you and @iamlilnicki  how was the premiere?!
Tweet 16: Hollis' death scene will hurt me severely to watch on film  wry is directors cut not out now?
Tweet 17: about to file taxes
Tweet 18: @LettyA ahh ive always wanted to see rent  love the soundtrack!!
Tweet 19: @FakerPattyPattz Oh dear. Were you drinking out of the forgotten table drinks?
```

Fig 5

```
Tweets after cleaning:

Tweet 0: awww thats a bummer you shoulda got david carr of third day to do it d
Tweet 1: is upset that he cant update his facebook by texting it and might cry as a result school today also blah
Tweet 2: i dived many times for the ball managed to save the rest go out of bounds
Tweet 3: my whole body feels itchy and like its on fire
Tweet 4: no its not behaving at all im mad why am i here because i cant see you all over there
Tweet 5: not the whole crew
Tweet 6: need a hug
Tweet 7: hey long time no see yes rains a bit only a bit lol im fine thanks hows you
Tweet 8: nope they didnt have it
Tweet 9: que me muera
Tweet 10: spring break in plain city its snowing
Tweet 11: i just repierced my ears
Tweet 12: i couldnt bear to watch it and i thought the ua loss was embarrassing
Tweet 13: it it counts idk why i did either you never talk to me anymore
Tweet 14: i wouldve been the first but i didnt have a gun not really though zac snyders just a doucheclown
Tweet 15: i wish i got to watch it with you i miss you and how was the premiere
Tweet 16: hollis death scene will hurt me severely to watch on film wry is directors cut not out now
Tweet 17: about to file taxes
Tweet 18: ahh ive always wanted to see rent love the soundtrack
Tweet 19: oh dear were you drinking out of the forgotten table drinks
```

Fig 6

The text should then be tokenized (split into an array of words) and stemmed (normalized to its root word). Although the removal of stop words such as ‘a’, ‘the’ and ‘of’, is commonly done in Natural Language Processing (NLP) tasks, this might not be the best approach when dealing with sentiment analysis. For instance, Python’s Natural Language Toolkit (NLTK) considers ‘not’ a stop word. If we remove stop words from our tweets, a sentence such as ‘I did not like the COVID-19 vaccine’ would then become ‘like COVID-19 vaccine’. In other words, a tweet with a negative sentiment could be turned into a positive one instead, which is not ideal. Then, we will vectorize the text into a feature matrix as vectorization transforms text data into numeric representations that can be easily recognized and processed by the machine.

Once the data is ready to be used, we would normally need to split our data into a training and testing set. Conveniently, the ‘Sentiment140’ dataset already has a separate training and testing set prepared for us. We also found out that the training set is balanced (*Fig.7*). Hence, we further split 80% from the training set for training and leave 20% for validation. Now, the training set consists of 1,280,000 rows, the validation set consists of 320,000 and the test set has 359 rows. Thereafter, we just need to feed this into the BERT model for fine-tuning.

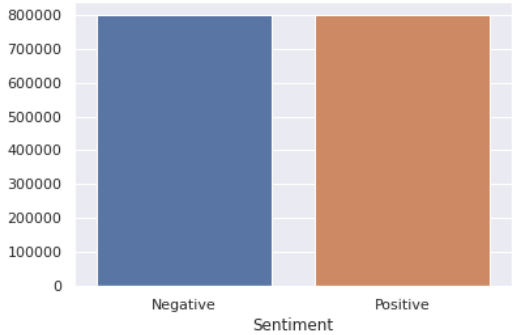


Fig 7

3.2 Fine-Tuning BERT

3.2.1 Introduction to BERT

As mentioned previously, we will use 'Sentiment140' dataset to fine-tune the pre-trained bert-base-uncased model (12-layer, 768-hidden, 12-heads, 110M parameters neural network architecture) by performing supervised training on the dataset. [5]

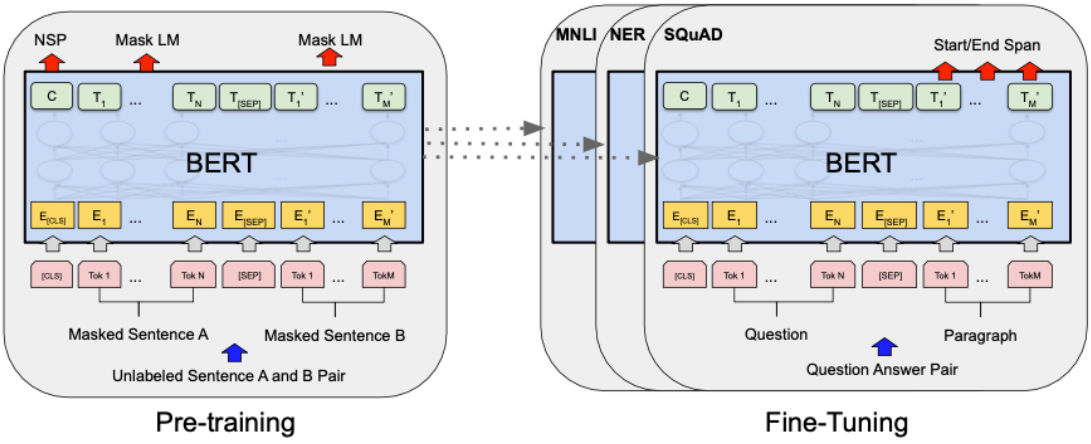


Fig 8

3.2.2 Fine-Tuning Strategies

BERT-base model contains an encoder with a hidden size of 768, 12 Transformer blocks, and 12 self-attention heads. BERT takes an input of a sequence of no more than 512 tokens and outputs the representation of the sequence. The sequence has one or two segments that the first token of the sequence is always `[CLS]` which contains the special classification embedding and another special token `[SEP]` is used for separating segments. [6]

To better adapt to our sentiment classification task, there are few factors to be considered:

1. The pre-processing of long text as the maximum sequence length of BERT is only 512 and all sequences must be padded or truncated to a single, fixed length.
2. Selection of the most effective layers. As BERT-base model consists of an embedding layer, a 12-layer encoder, and a pooling layer. Therefore, correct selection of the most effective layers is highly needed.
3. Avoid overfitting problem, which means a better optimizer with an appropriate learning rate is desired.

3.2.3 Fine-Tuning with PyTorch

We explored two different methods to fine-tune the BERT model. We experimented with using Pytorch and Keras/ktrain respectively. For fine-tuning, we relied on the 1 Tesla V100 GPU offered by Google Colab Pro.

The HuggingFace library is one of the most powerful PyTorch interface working with BERT, the library includes multiple pre-trained transformer models, and for our binary text classification task at hand, we chose `BertForSequenceClassification` model.

After loading the pre-processed data to our model, we first import `BertTokenizer` from the HuggingFace library and convert our input sequences into the tokens that required by BERT. One example is illustrated as below:

```
Original:  i dived many times for the ball managed to save the rest go out of
bounds

Tokenized:  ['i', 'dive', '##d', 'many', 'times', 'for', 'the', 'ball',
'managed', 'to', 'save', 'the', 'rest', 'go', 'out', 'of', 'bounds']

Token IDs:  [1045, 11529, 2094, 2116, 2335, 2005, 1996, 3608, 3266, 2000,
3828, 1996, 2717, 2175, 2041, 1997, 19202]
```

Before encoding the texts, we need to pad and truncate the input texts such that they are all of the same length. We identified the maximum length of the input sequence in the Sentiment 140 dataset is 72. To achieve better accuracy, we first set the maximum length as 256. However, the maximum length has a heavy impact on training speed. For example, with a Tesla V100 GPU, we can see that setting `MAX_LEN` to a larger value took noticeably longer time.

```
MAX_LEN = 256 --> Each training epoch took ~8:30:42

MAX_LEN = 128 --> Each training epoch took ~4:11:38

MAX_LEN = 64 --> Each training epoch took ~1:11:26
```

Next we can tokenize the texts by utilizing `tokenizer.encode_plus` function, which combines multiple steps, it split the input sequence into tokens, add the special token `[CLS]` and `[SEP]` to the start and end respectively, map the tokens to its IDs, then pad and truncation the sequence to `MAX_LEN`, lastly, it create attention masks for `[PAD]` tokens. For batch size, we opt for 32 to fully utilize the GPU and to avoid memory overload issues.

Then, we will choose the optimizer and learning rate scheduler. We followed the GLUE benchmark. Here, we adopted Adam optimizer with weight decay (`AdamW`) and we start with setting `learning_rate=2e-5, epochs=2`. We also create a learning rate scheduler by utilizing `get_linear_schedule_with_warmup` from HuggingFace library. It employs a learning rate scheduler that firstly warms up from 0 and then decays to 0.

3.2.4 Fine-Tuning with Keras/ktrain

The `ktrain` library is a low-code Python library that can be used with machine learning models that are implemented in TensorFlow Keras and is a wrapper to many other libraries including the `Transformers` library. Therefore, we chose to utilize `ktrain` as it is designed to make complex machine learning models simpler to apply and more accessible. In contrast to automatic machine learning (AutoML) solutions that emphasizes on automating subsets of the model-building process, `ktrain` focuses on either partially or fully automating other aspects of the machine learning workflow. `ktrain` assists in the whole machine learning process from preprocessing to training, tuning, troubleshooting and applying models but still gives users the flexibility to customize their models to the way that best fits each user's specific application requirements [7].

The first step in our workflow is to load and preprocess the data. Here, we used our cleaned dataset as an input to the `text.texts_from_df()` function in `ktrain`. In this step, we had to decide on the maximum length of words that will be used. Again, we opted for the maximum sequence length to be 128, which means that the recommended batch size that we will use is 32.

Next, we build our text classification model using the `text.text_classifier()` function. We specified to use the BERT model as our text classifier. Then, we create a learner instance using the function `ktrain.get_learner()` that can be used to tune and train Keras models using a batch size of 32. Our next task is to find the optimal learning rate for our model. To do this, we use `learner.lr_find()`. This function simulates training for different learning rates and plots a graph (*Fig.9*) of loss as learning rate increases when it has completed. Due to time and computing power limitations, we only ran this for 1 epoch, which took approximately 4 hours. From this graph, we choose the highest learning rate corresponding to a still falling loss.

```
simulating training for different learning rates... this may take a few moments...
40000/40000 [==============================] - 14416s 360ms/step - loss: nan - accuracy: 0.6897

done.
Visually inspect loss plot and select learning rate associated with falling loss
```

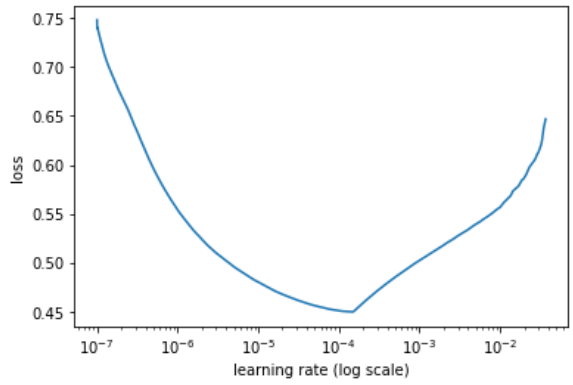


Fig 9

We need to be very careful when choosing a good learning rate. In neural networks, the aim of training is to minimize the loss function. If the learning rate is too low, training will be more dependable but since the step size is small, optimization will take longer. Conversely, if the learning rate is too high, training might not even converge meaning that loss will not be minimized. Here, we choose 1×10^{-4} as the maximum learning rate.

Thereafter, we train our model using the triangular learning rate policy, 1cycle policy and SGDR policy with the chosen maximum learning rate to see which one performed the best.

3.3 Sentiment Prediction of COVID-19 Vaccine Tweets

With our chosen fine-tuned BERT model, we will use it to predict the sentiments of the COVID-19 Vaccines Tweet dataset. However, before we do so, we need to first clean and filter the dataset to extract relevant tweets. Here, we only want tweets from each state in the United States. Initially we tried filtering the `user_location` column by all the cities in a particular state and the state name but it returned numerous irrelevant results. Therefore, we tried to filter by state name, state abbreviation and its capital instead, which returned mostly useable data. For states such as Alabama with abbreviation 'AL', Delaware with abbreviation 'DE', Indiana with abbreviation 'IN', Louisiana with abbreviation 'LA', Maine with capital 'Augusta' and abbreviation 'ME', Montana with abbreviation 'MT', Nebraska with abbreviation 'NE' and capital 'Lincoln', South Carolina with capital 'Columbia', we had to further filter the dataset due to the common use of their abbreviations and/or capital name. After extracting 6904 relevant ones from the initial 31036 tweets, the resulting number of tweets for each state is as follows.

State	No. of Tweets	State	No. of Tweets
Alabama	53	Montana	13
Alaska	24	Nebraska	14
Arizona	127	Nevada	55
Arkansas	29	New Hampshire	15
California	1170	New Jersey	156
Colorado	132	New Mexico	34
Connecticut	90	New York	808
Delaware	18	North Carolina	159
Florida	403	North Dakota	8
Georgia	202	Ohio	191
Hawaii	38	Oklahoma	71
Idaho	24	Oregon	125
Illinois	206	Pennsylvania	225
Indiana	47	Rhode Island	23
Iowa	50	South Carolina	54
Kansas	62	South Dakota	10
Kentucky	66	Tennessee	114
Louisiana	68	Texas	560
Maine	28	Utah	42
Maryland	120	Vermont	25
Massachusetts	244	Virginia	158
Michigan	134	Washington	392
Minnesota	102	West Virginia	38
Mississippi	23	Wisconsin	60
Missouri	86	Wyoming	8

As illustrated above, we notice that states with larger cities like California and New York have a larger number of tweets as these cities are probably more populated with younger people that are well-versed with social media platforms (i.e. Twitter). From the `explain()` function from `ktrain`, we can also see an explanation of how each text classification was made as an example shown below.

The words highlighted in green contributes to a positive prediction and the ones in red contribute to a negative prediction.

After predicting the sentiments, we can then analyze how the sentiments towards vaccines change as shown from the tweets and find out its correlation with the COVID-19 vaccination progress.

4. Evaluation

To evaluate the reliability of our model, we will be looking at its accuracy when predicting whether the sentiment of a tweet is positive or negative on our 'Sentiment140' validation set.

4.1 BERT Hyperparameter Tuning

There are a few hyperparameters to be considered when it comes to optimizing our fine-tuning process. Per Appendix A.3 of the BERT paper [8], the authors recommended the following range of possible values to work well across all tasks. Therefore, we will use this as a starting point and further explore. It is also noted that for large datasets, such as the "Sentiment140" we have at hand, it is far less sensitive to hyperparameter choice than small datasets.

1. Learning rate (Adam): 5e-5, 3e-5, 2e-5
2. Batch size: 16, 32
3. Number of epochs: 2, 3, 4

4.1.1 Learning Rate

It is known that the learning rate is the most important hyperparameter to tune for training deep neural networks. We divide this section into the following two parts. First, we follow the authors' suggestions and choose the optimal learning rates. Then, we will do some experiments on three learning rate policies using `ktrain`.

A. Optimal Learning Rate Values with AdamW optimizer

As previously discussed in 3.2.3, following the GLUE benchmark presented in the paper, we adopted Adam optimizer with weight decay (`AdamW`). The default `learning_rate` value is 5e-5. After experimentation, we decided to use the base `learning_rate` 2e-5 as we find that a lower learning rate is necessary to help BERT overcome forgetting problem. With the default learning rate 5e-5 , the training set fails to converge.

B. Experimenting with Learning Rate Policies with *ktrain*

Since varying the learning rate cyclically during training has proven to be effective, *ktrain* provides three different learning rate policies to be utilized during training. The learning rate policies include the triangular learning rate policy via the `autofit()` method, the 1cycle policy via the `fit_onecycle()` method and the Stochastic Gradient Descent with Restart (SGDR) schedule via the `fit()` method.

An example of how their learning rate schedule differs is shown in the figures below.

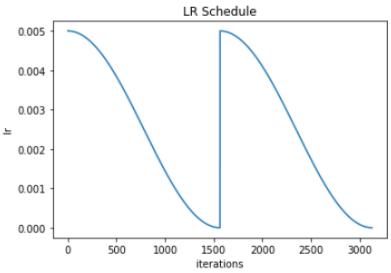


Fig 12: SGDR

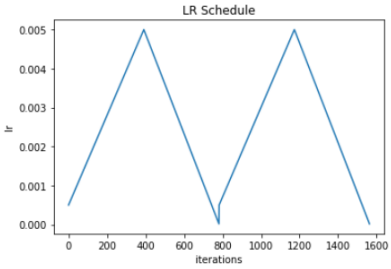


Fig 13:
Triangular Learning Rate Policy

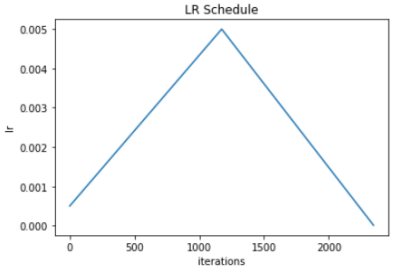


Fig 14: 1cycle Policy

We will test all three policies and chose the policy best suited for our model (i.e. the one that performs the best).

The 1cycle Policy

It is recommended to do a cycle with two steps of equal lengths, one going from a lower learning rate to a higher one then going back to the minimum. The maximum should be the value picked with the learning rate finder and the lower one can be ten times lower. Applying this policy also allows us to pick larger maximum learning rates. [9]

We tested 1cycle policy with the maximum learning rate of 1×10^{-4} and ran it for 1 epoch, which took about 4 hours and 18 minutes.

```
# Train the model
learner.fit_onecycle(lr = 1e-4, epochs = 1)
```

```
begin training using onecycle policy with max lr of 0.0001...
40000/40000 [==============================] - 15513s 387ms/step - loss: 0.3815 - accuracy: 0.8297 - val_loss: 0.3466 - val_accuracy: 0.8478
<tensorflow.python.keras.callbacks.History at 0x7f466584a850>
```

This produced a training accuracy of 83% and validation accuracy of 84.78%. Since a validation accuracy of 84.78% is impressive for only 1 epoch, we attempted to repeat this for 2 epochs to see if the accuracy will further improve. This took approximately 8 hours and 19 minutes to train. Interestingly, the accuracy drops significantly in the second epoch. This might be due to overfitting.

```
begin training using onecycle policy with max lr of 0.0001...
Epoch 1/2
40000/40000 [==============================] - 14980s 375ms/step - loss: 0.3787 - accuracy:
0.8312 - val_loss: 0.3989 - val_accuracy: 0.8178
Epoch 2/2
40000/40000 [==============================] - 14961s 374ms/step - loss: 0.6867 - accuracy:
0.5162 - val_loss: 0.7050 - val_accuracy: 0.5006
```

The Triangular Learning Rate Policy

First, the triangular learning rate policy linearly increases the learning rate and then linearly decays it according the the following update schedule.

$$
\begin{aligned}
cut &= \lfloor T \cdot cut\_frac \rfloor \\
p &= \begin{cases} t/cut, & \text{if } t < cut \\ 1 - \frac{t-cut}{cut \cdot (1/cut\_frac - 1)}, & \text{otherwise} \end{cases} \\
\eta_t &= \eta_{\max} \cdot \frac{1 + p \cdot (\text{ ratio } - 1)}{\text{ratio}}
\end{aligned}
$$

Here, T is the number of training iterations, cut_frac is the fraction of itereations we increase the learning rate, cut is the iteration when we switch from increasing to decreasing the learning rate, p is the fraction of the number of iterations we have increased or will decrease the learning rate respectively, $ratio$ specifies how much smaller the lowest learning rate is from the maximum learning rate $\eta_{\max}$, and η_t is the learning rate at iteration t. [10]

Since training with 2 epochs decreases validation accuracy for the 1cycle policy in addition to computing power and quota limitations, we will be training using the triangular learning rate policy next with 1 epoch. This took approximately 6 hours and 38 minutes to complete.

```
begin training using triangular learning rate policy with max lr of 0.0001...
40000/40000 [==============================] - 23932s 598ms/step - loss: 0.3906 - accuracy
0.8237 - val_loss: 0.3429 - val_accuracy: 0.8498
```

Comparatively, the results are very similar to the 1cycle policy but the validation accuracy is slightly higher at 84.98%.

The SGDR Policy

This is a simple warm restart technique for stochastic gradient descent to improve its anytime performance while training deep neural networks. As gradients and loss values can vary widely

from one batch of the data to another, the incoming information should be denoised – by considering averaged gradients and losses. [11]

```
40000/40000 [==============================] - 23492s 587ms/step - loss: 0.5818 - accuracy: 0.6407 - val_loss:
0.6936 - val_accuracy: 0.4994
```

The SGDR policy took approximately 6 hours and 31 minutes to run and ended up with a low validation accuracy of 49.94%.

The Best Learning Rate Policy in *ktrain*

Based on the results, it seems that the triangular learning rate policy is best suited for our model as it performed the best among the three models.

4.1.2 Batch Size

Due to GPU memory limitations and for the purpose of better generalization, we set our batch size at 32.

4.1.3 Epoch

An epoch refers to one cycle through the full training set. As BERT was pre-trained on a huge amount of data and already encoded with a lot of information, there is no need for running lots of epochs.

In `ktrain`, the models performed the best when trained with only 1 epoch, reaching a validation accuracy of 85%.

When using the `Hugging Face` library, we empirically set the max number of the epoch to 4 and saved the best model, as suggested by the authors of BERT paper. Nevertheless, we realized that 2 epochs provided us with the best validation accuracy of 88%, and the validation accuracy started to drop from the 3rd epochs due to overfitting. Thus, we opted for 2 epochs.

4.2 Model Selection

Since the model from the `Hugging Face` library performed the best, we selected it as our final model with the following hyperparameter values:

1. Learning rate: 2e-5
2. Max sequence length: 128
3. Batch size: 32
4. Epochs: 2

The performance statistics for training and validation is as follows:

epoch	Training Loss	Val_Loss	Val_Accuracy	Training Time	Validation Time
1	0.33	0.3	0.87	4:11:38	0:20:11
2	0.26	0.3	0.88	4:10:42	0:19:59

Since we have chosen the best model, we will run it against the test data to see its final performance. The resulting accuracy for the test set is 85%.

5. Final Results

Now that we have our prediction results from our classification of COVID-19 vaccines-related tweet, we will examine the relationship between the vaccination progress and sentiments about vaccines as reflected in tweets throughout different states by visualizing it as a percentage stacked bar chart and percentage bar chart. Although initially we wanted to represent both as a line chart, we felt that a percentage bar chart would be a better presentation.

For each state, as shown below on the left column, we will plot a percentage stacked bar chart with month on the x-axis and percentage of positive/negative tweets on the y-axis. The percentage of tweets that are negative is represented by the red portion while the percentage of positive tweets is represented by the green portion.

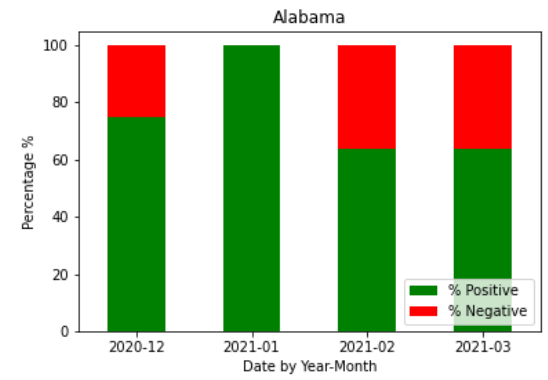
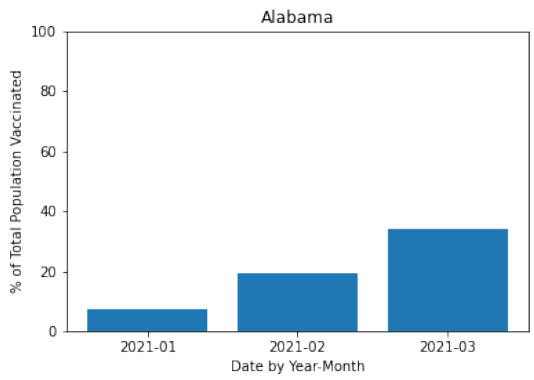
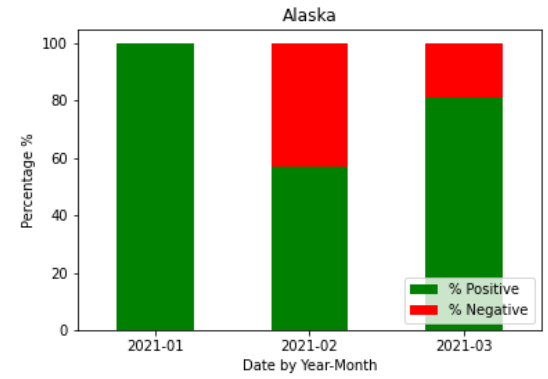
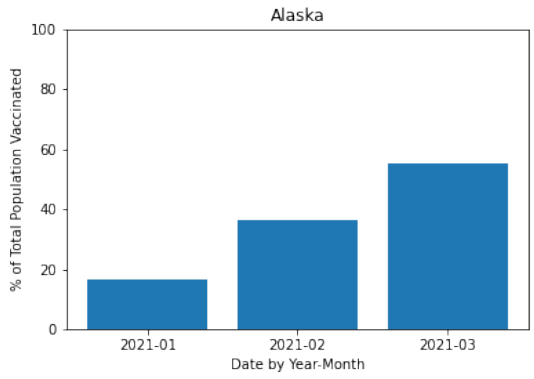
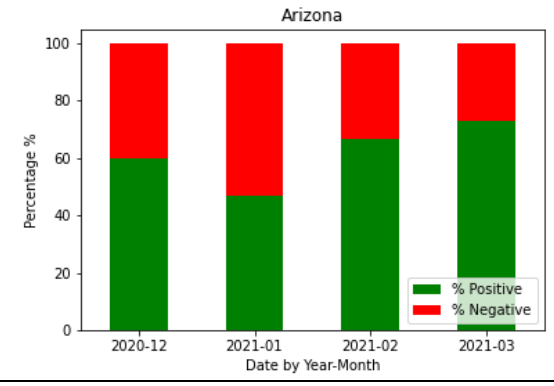
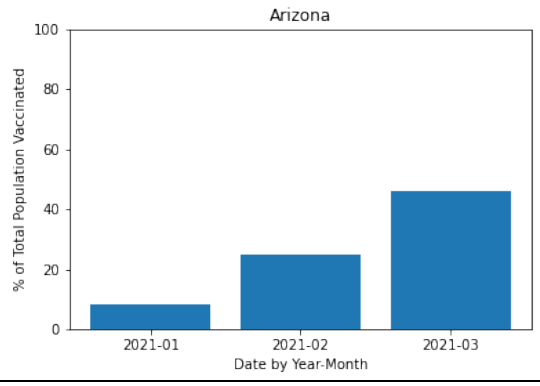
On the right column, we will present another bar chart with month on the x-axis and percentage of each state's population that is vaccinated on the y-axis. Then, we will do a comparison to determine their relationship. Even though we originally wanted to further explore the individual vaccines for each state as the vaccination progresses, we realized that this might not be the best choice with the amount of data that we have for each state.

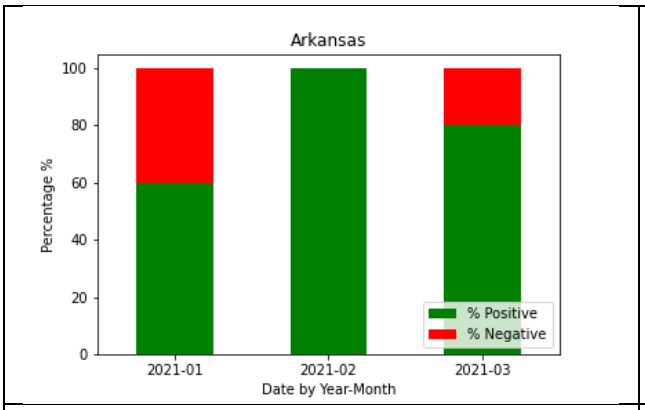
Based on the bar charts below, we made a couple of observations:

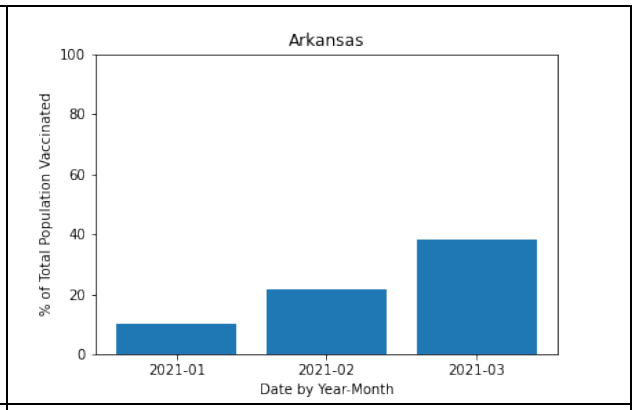
1. By March, the percentage of the population that is vaccinated in most states is at least 50% except for Georgia, Indiana, Louisiana, Mississippi, Missouri, South Carolina, Tennessee, Texas and Utah. The percentage of the population that is vaccinated in those other states is approximately 40%.
2. Since the vaccination progress has a similar trend overall in the different states (i.e. increasing from January to March), we will focus on how the sentiments change from December to March. Here are the states that stood out.
   a) *Alaska*: Started off by being completely positive (100%) in January but dropped to 60% positive in February. It later increased to being 80% positive in March. *Arizona*: On average, Arizona maintained a sentiment of approximately 60% positive throughout the 4 months.
   b) *California*: California was similar to Arizona but maintained a positive sentiment of 70% on average.
   c) *Delaware*: There was a very drastic change in sentiment over the months. It was 100% positive in January but 100% negative in February. It was approximately 55% positive in

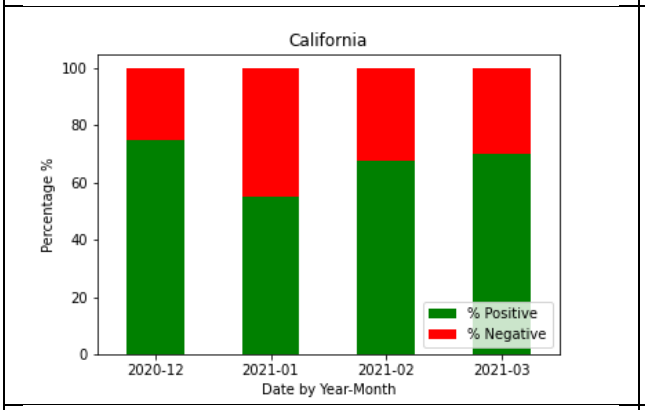
March. This major fluctuation might be due to the fact that there are only 18 Tweets for this state.

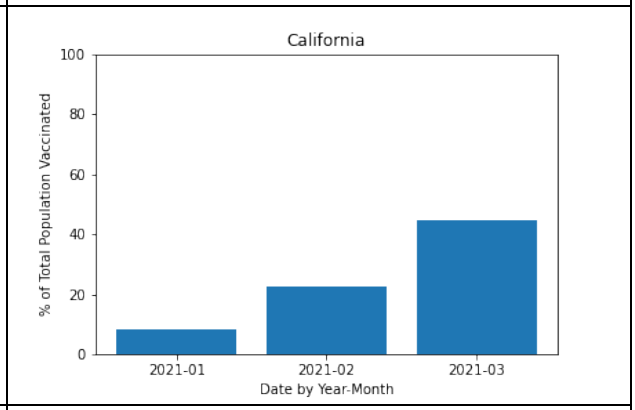
d) *Haiwaii* – Hawaii was pretty positive in the first two months (100%) but dropped to approximately 65% in February and increased again to 80% in the March.
e) *Kansas* – Initially uncertain (50%) but was completely positive (100%) in January and maintained around 80% in the next two months.
f) *Kentucky* – Initially entirely negative in December but entirely positive in January. Then was approximately 70% positive in the next two months.
g) *Mississippi* – Entirely positive (100%) for the first 3 months and around 75% positive in March.
h) *North Dakota* – Entirely positive (100%) from January to March.

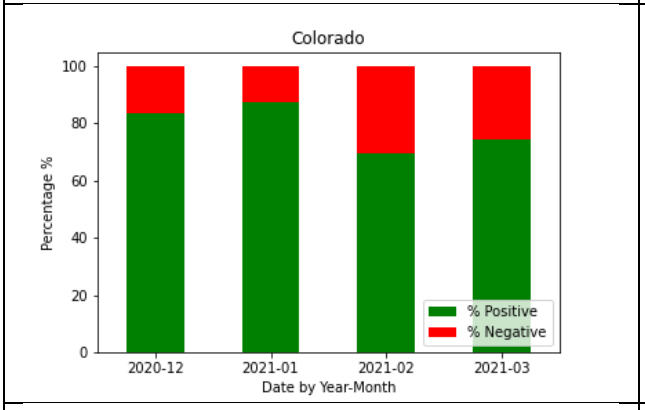
Percentage Stacked Bar Chart	Vaccination Progress
	
	
	

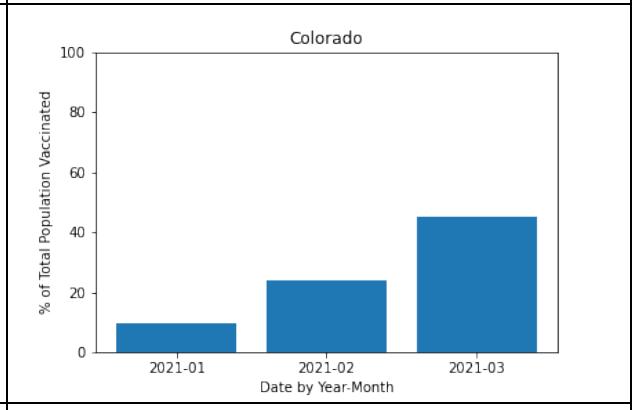

Arkansas
Percentage %
Date by Year-Month
% Positive
% Negative
2021-01
2021-02
2021-03

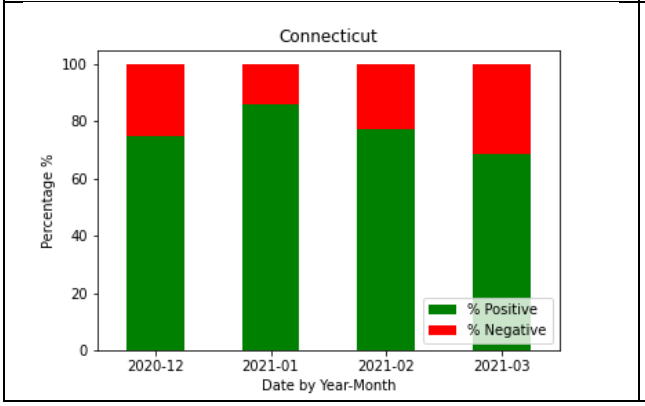

Arkansas
% of Total Population Vaccinated
Date by Year-Month
2021-01
2021-02
2021-03

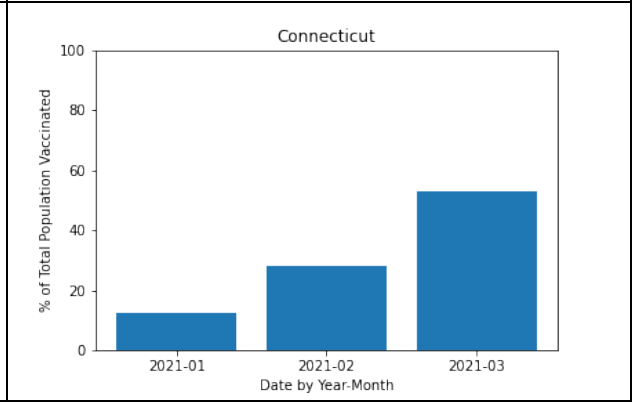

California
Percentage %
Date by Year-Month
% Positive
% Negative
2020-12
2021-01
2021-02
2021-03

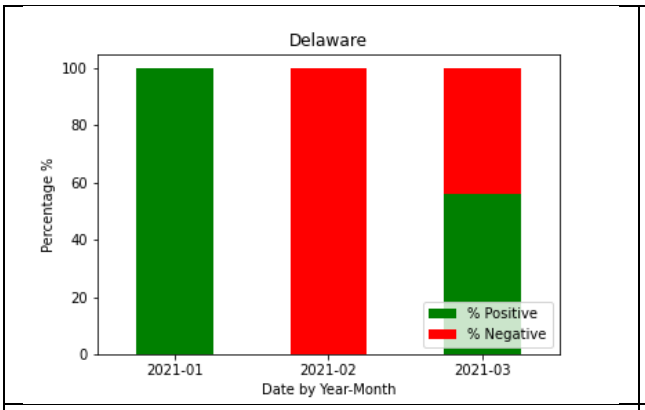

California
% of Total Population Vaccinated
Date by Year-Month
2021-01
2021-02
2021-03

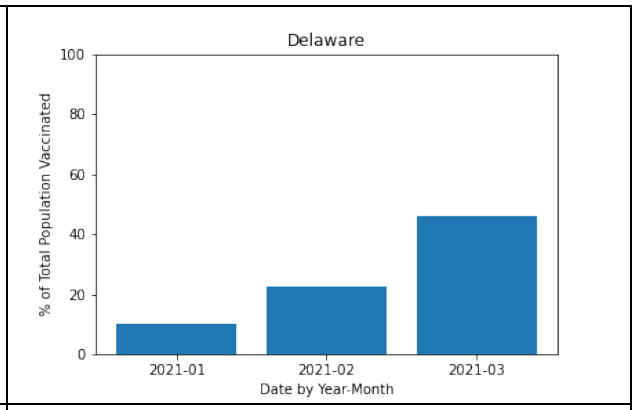

Colorado
Percentage %
Date by Year-Month
% Positive
% Negative
2020-12
2021-01
2021-02
2021-03

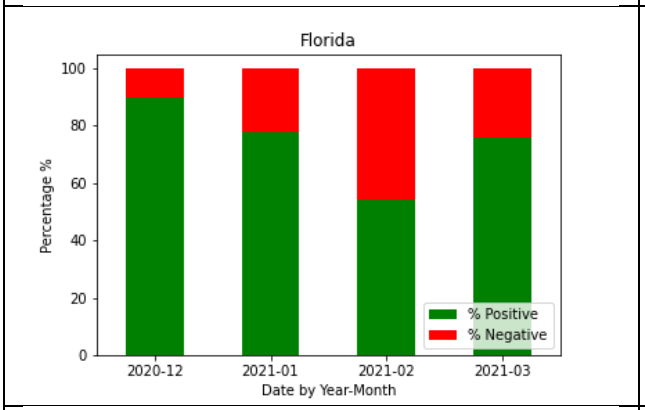

Colorado
% of Total Population Vaccinated
Date by Year-Month
2021-01
2021-02
2021-03

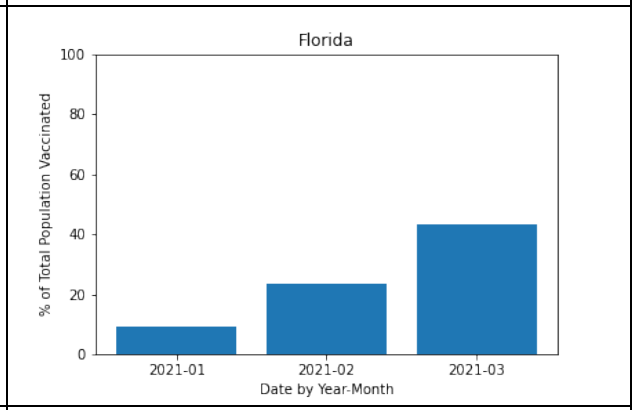

Connecticut
Percentage %
Date by Year-Month
% Positive
% Negative
2020-12
2021-01
2021-02
2021-03

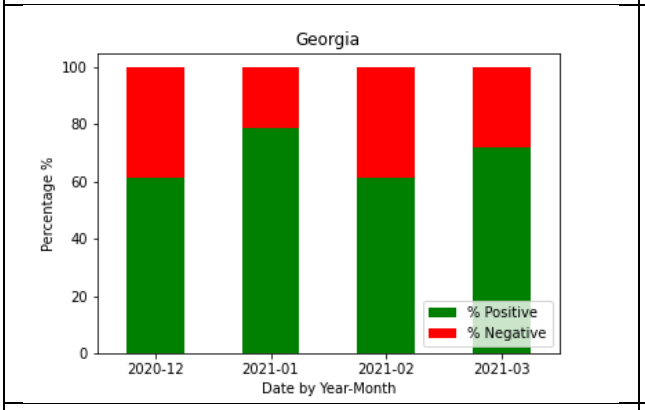

Connecticut
% of Total Population Vaccinated
Date by Year-Month
2021-01
2021-02
2021-03

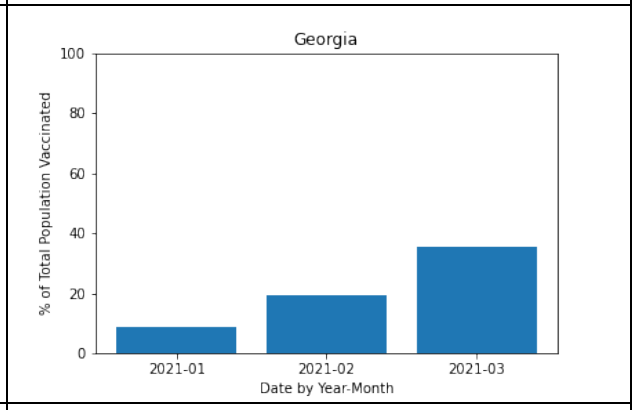

Delaware
Percentage %
Date by Year-Month
% Positive
% Negative
2021-01
2021-02
2021-03

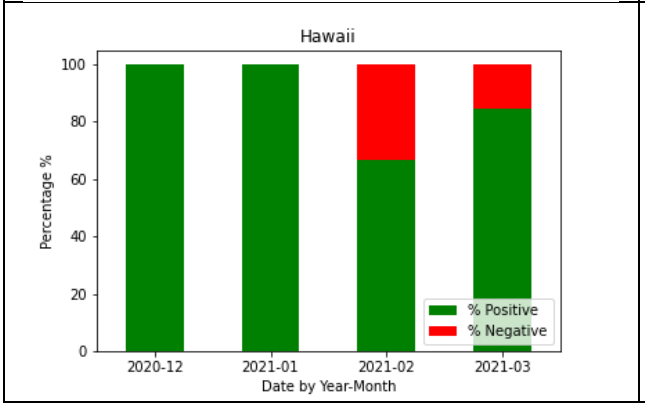

Delaware
% of Total Population Vaccinated
Date by Year-Month
2021-01
2021-02
2021-03

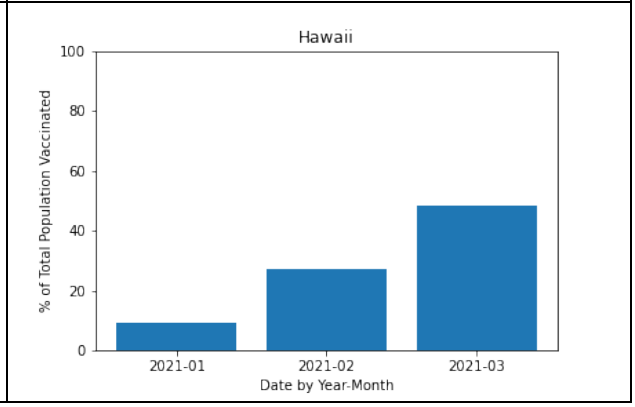

Florida
Percentage %
Date by Year-Month
% Positive
% Negative
2020-12
2021-01
2021-02
2021-03

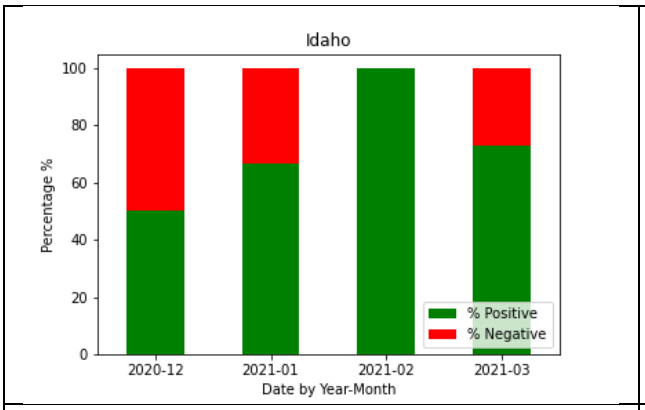

Florida
% of Total Population Vaccinated
Date by Year-Month
2021-01
2021-02
2021-03

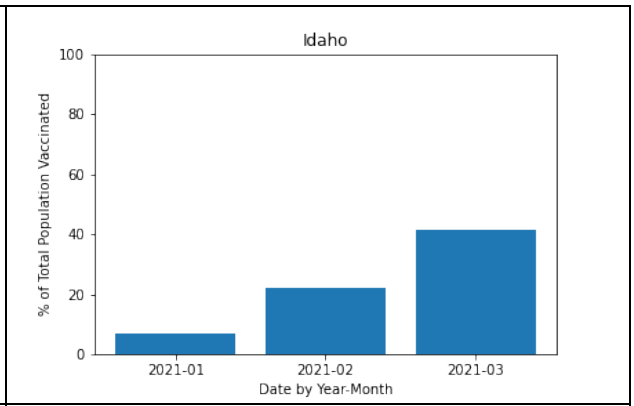

Georgia
Percentage %
Date by Year-Month
% Positive
% Negative
2020-12
2021-01
2021-02
2021-03

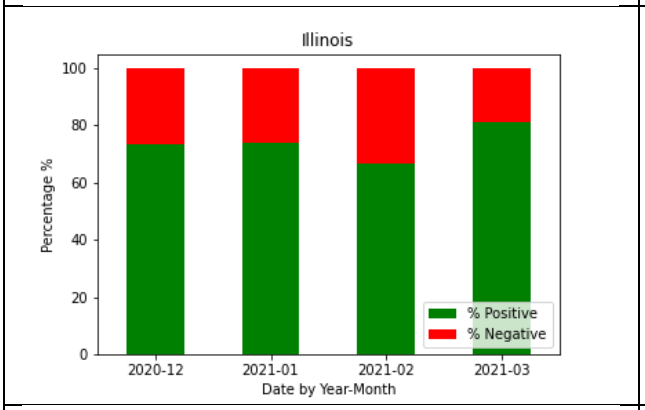

Georgia
% of Total Population Vaccinated
Date by Year-Month
2021-01
2021-02
2021-03

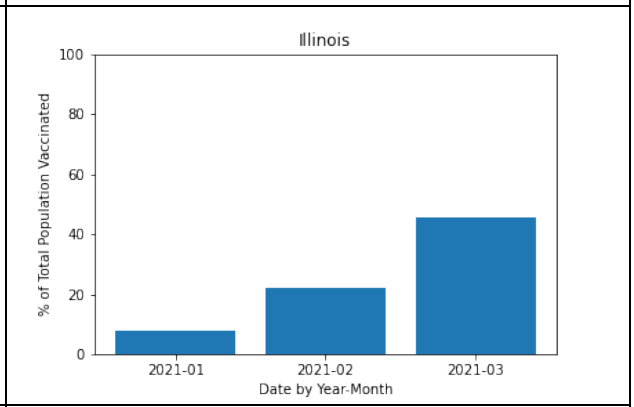

Hawaii
Percentage %
Date by Year-Month
% Positive
% Negative
2020-12
2021-01
2021-02
2021-03

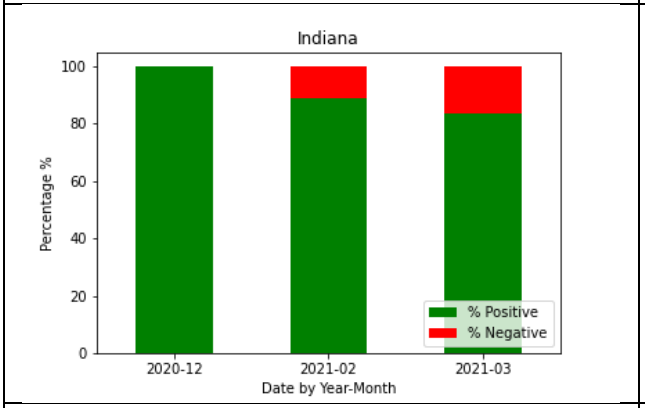

Hawaii
% of Total Population Vaccinated
Date by Year-Month
2021-01
2021-02
2021-03

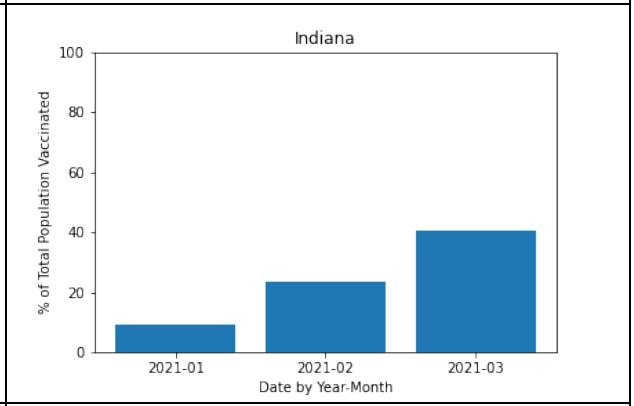

Idaho
Percentage %
Date by Year-Month
% Positive
% Negative
2020-12
2021-01
2021-02
2021-03

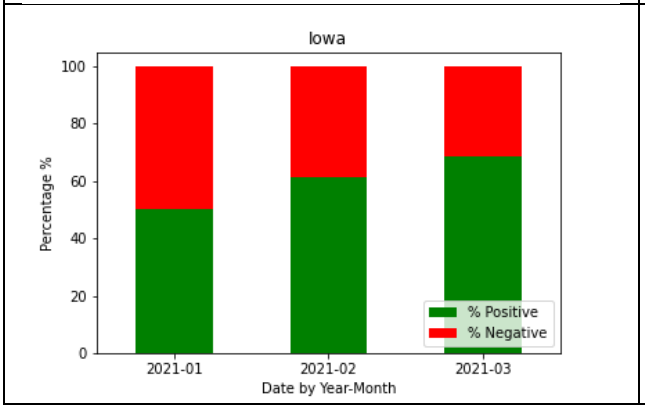

Idaho
% of Total Population Vaccinated
Date by Year-Month
2021-01
2021-02
2021-03

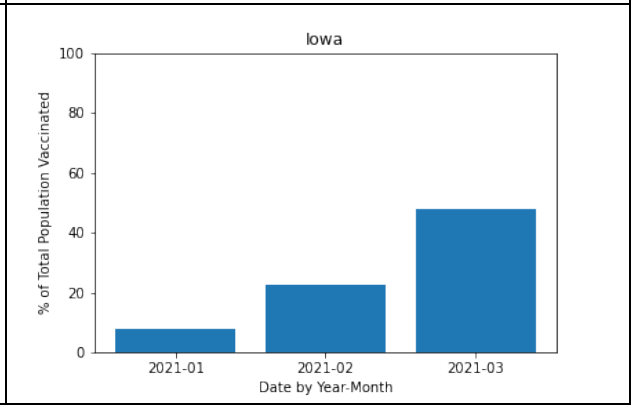

Illinois
Percentage %
Date by Year-Month
% Positive
% Negative
2020-12
2021-01
2021-02
2021-03

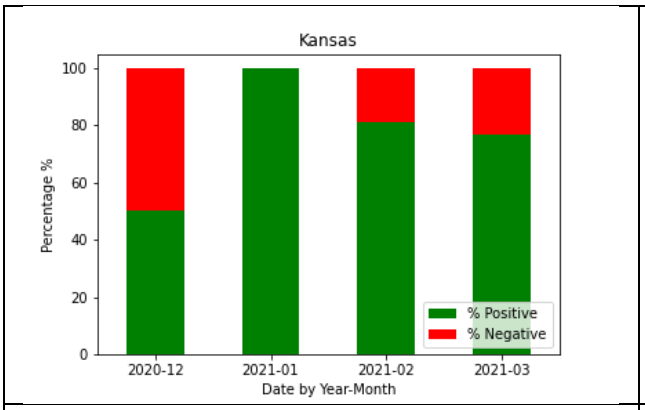

Illinois
% of Total Population Vaccinated
Date by Year-Month
2021-01
2021-02
2021-03

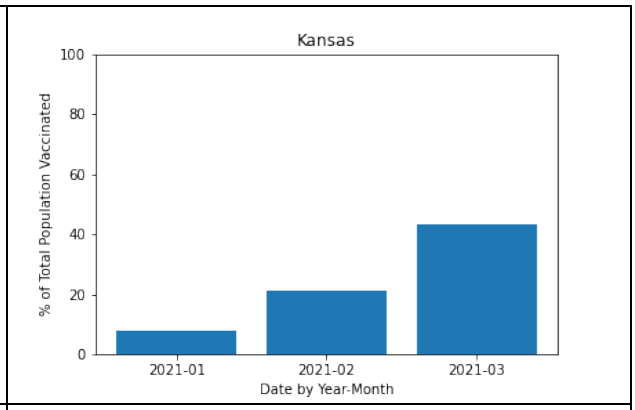

Indiana
Percentage %
Date by Year-Month
% Positive
% Negative
2020-12
2021-02
2021-03

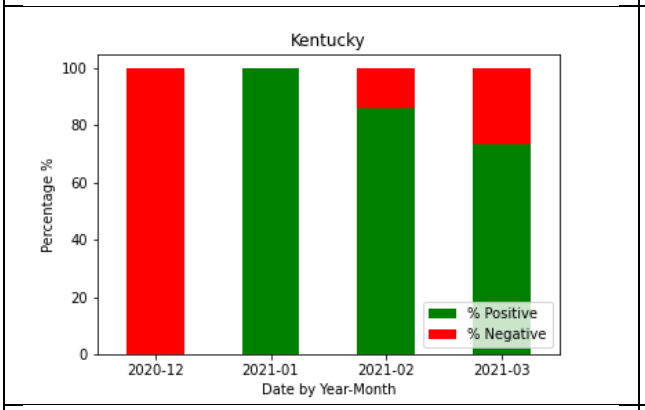

Indiana
% of Total Population Vaccinated
Date by Year-Month
2021-01
2021-02
2021-03

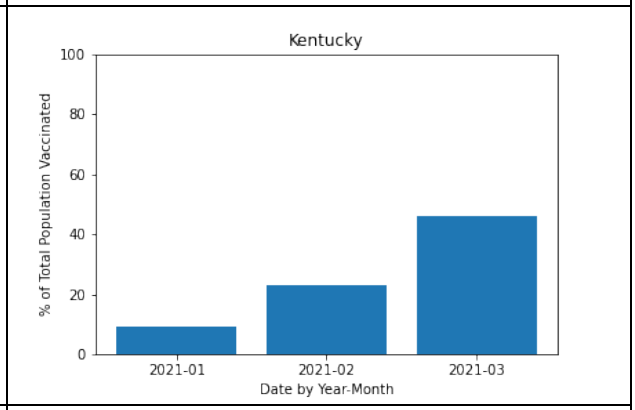

Iowa
Percentage %
Date by Year-Month
% Positive
% Negative
2021-01
2021-02
2021-03

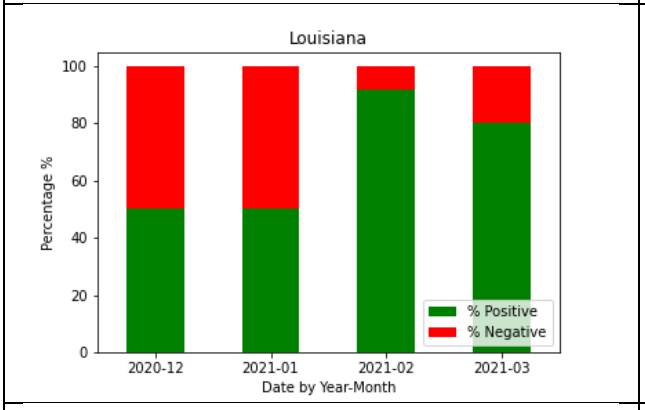

Iowa
% of Total Population Vaccinated
Date by Year-Month
2021-01
2021-02
2021-03

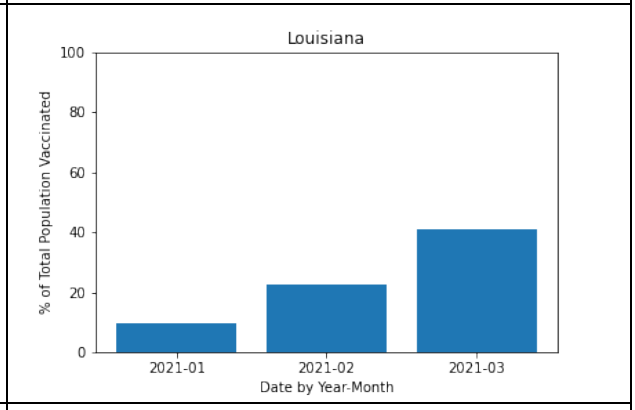

Kansas
Percentage %
Date by Year-Month
2020-12
2021-01
2021-02
2021-03
% Positive
% Negative

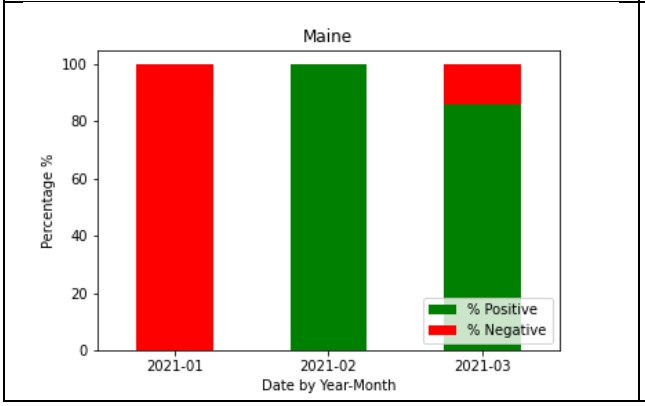

Kansas
% of Total Population Vaccinated
Date by Year-Month
2021-01
2021-02
2021-03

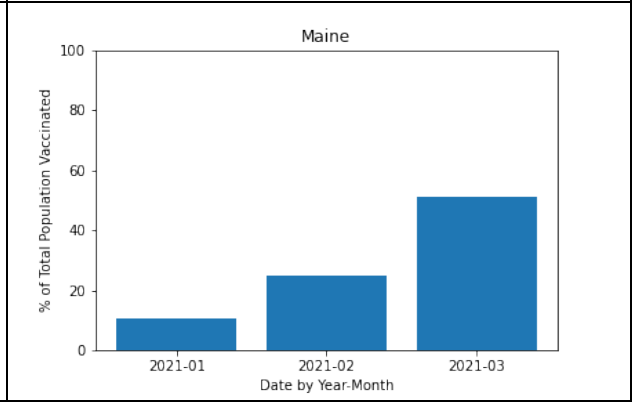

Kentucky
Percentage %
Date by Year-Month
2020-12
2021-01
2021-02
2021-03
% Positive
% Negative

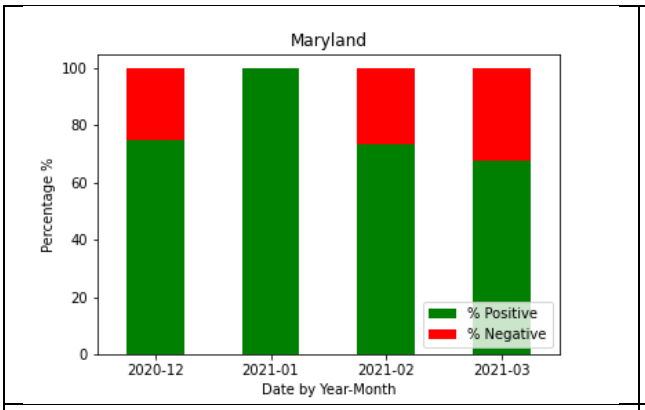

Kentucky
% of Total Population Vaccinated
Date by Year-Month
2021-01
2021-02
2021-03

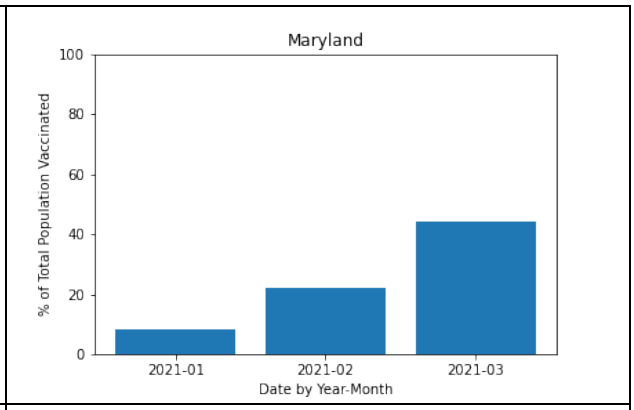

Louisiana
Percentage %
Date by Year-Month
2020-12
2021-01
2021-02
2021-03
% Positive
% Negative

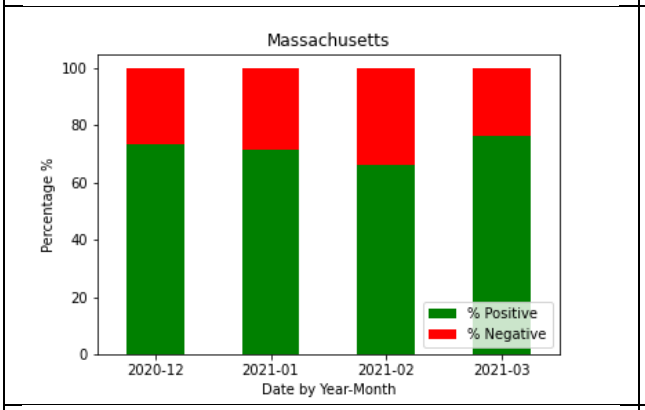

Louisiana
% of Total Population Vaccinated
Date by Year-Month
2021-01
2021-02
2021-03

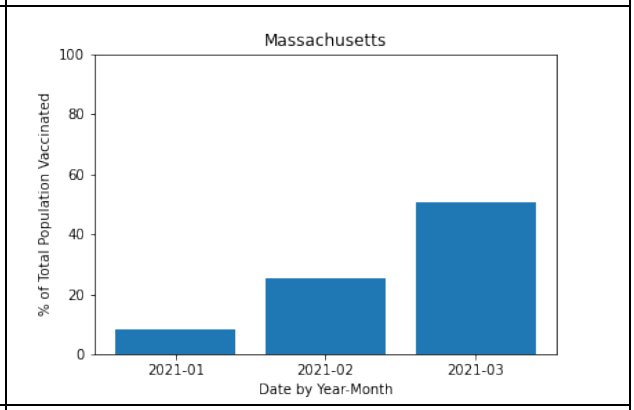

Maine
Percentage %
Date by Year-Month
2021-01
2021-02
2021-03
% Positive
% Negative

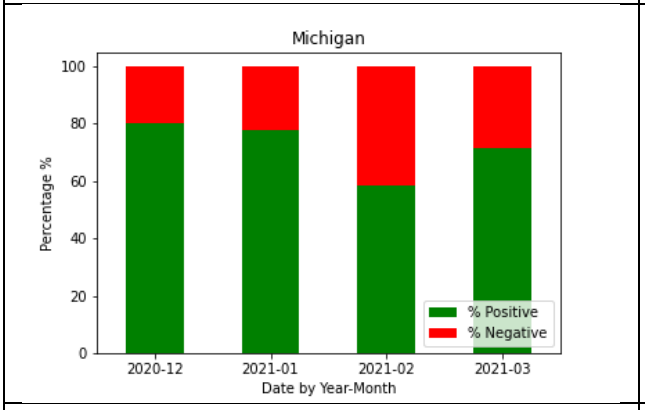

Maine
% of Total Population Vaccinated
Date by Year-Month
2021-01
2021-02
2021-03

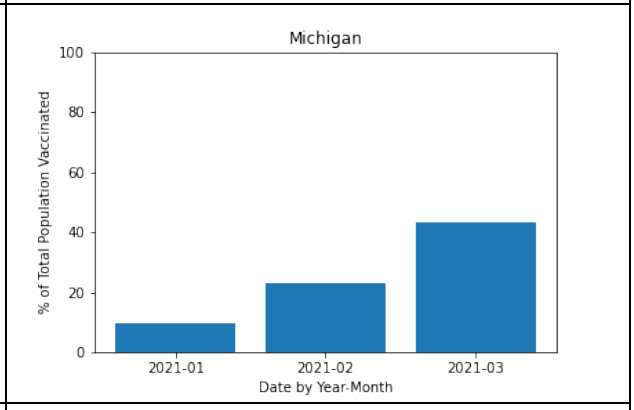

Maryland
Percentage %
Date by Year-Month
% Positive
% Negative
2020-12
2021-01
2021-02
2021-03

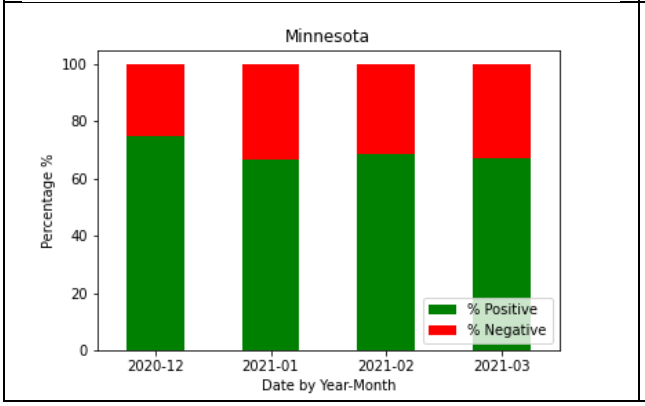

Maryland
% of Total Population Vaccinated
Date by Year-Month
2021-01
2021-02
2021-03

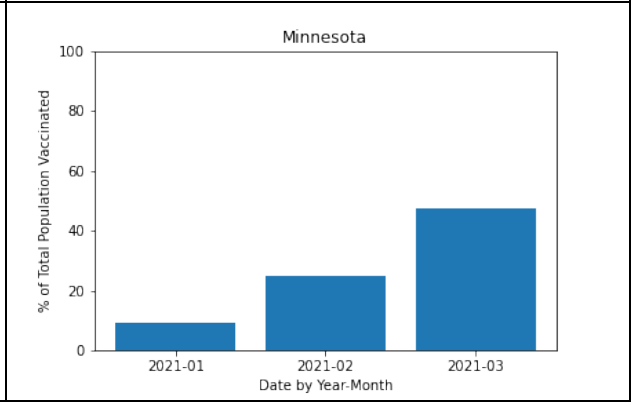

Massachusetts
Percentage %
Date by Year-Month
% Positive
% Negative
2020-12
2021-01
2021-02
2021-03

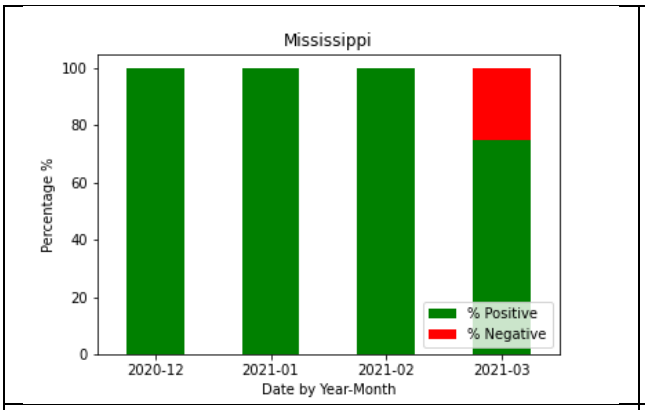

Massachusetts
% of Total Population Vaccinated
Date by Year-Month
2021-01
2021-02
2021-03

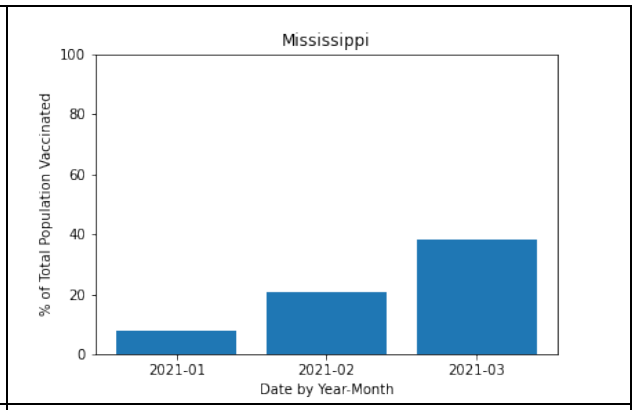

Michigan
Percentage %
Date by Year-Month
% Positive
% Negative
2020-12
2021-01
2021-02
2021-03

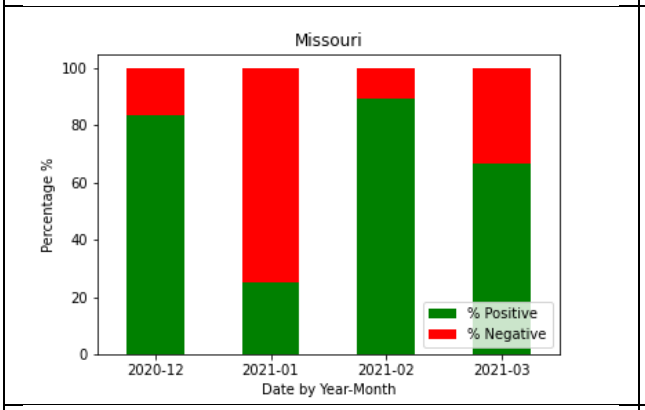

Michigan
% of Total Population Vaccinated
Date by Year-Month
2021-01
2021-02
2021-03

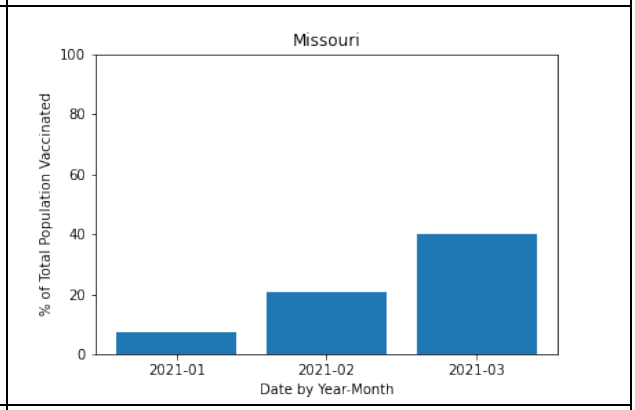

Minnesota
Percentage %
Date by Year-Month
% Positive
% Negative
2020-12
2021-01
2021-02
2021-03

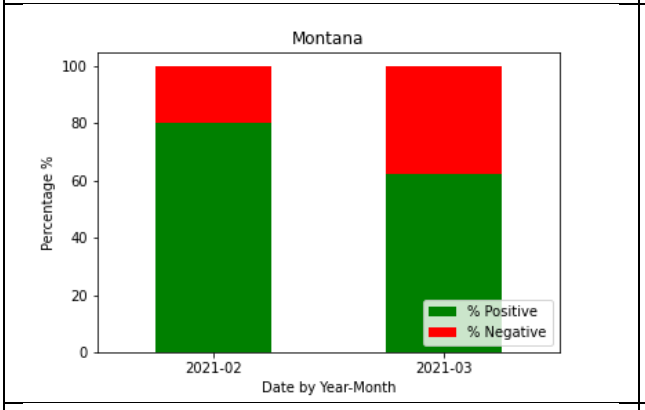

Minnesota
% of Total Population Vaccinated
Date by Year-Month
2021-01
2021-02
2021-03

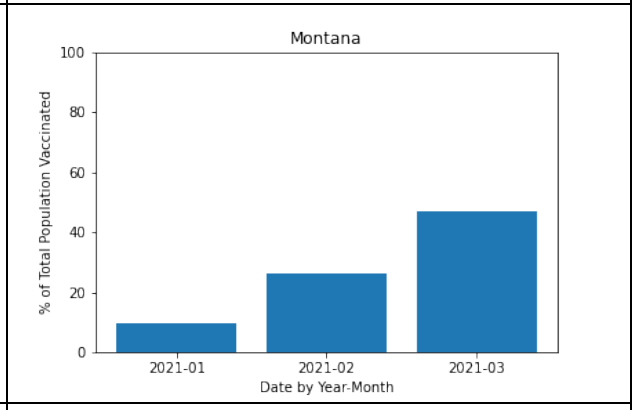

Mississippi
Percentage %
Date by Year-Month
2020-12
2021-01
2021-02
2021-03
% Positive
% Negative

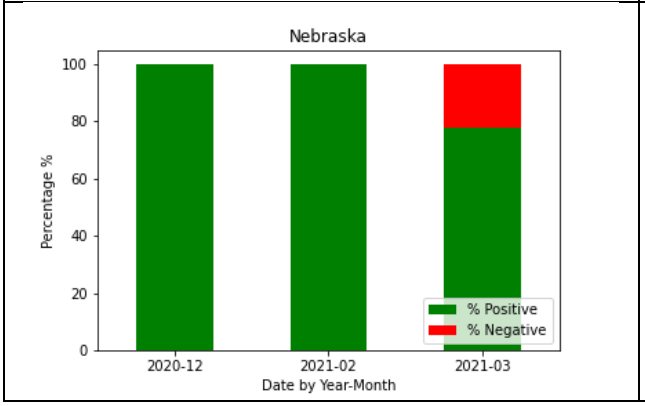

Mississippi
% of Total Population Vaccinated
Date by Year-Month
2021-01
2021-02
2021-03

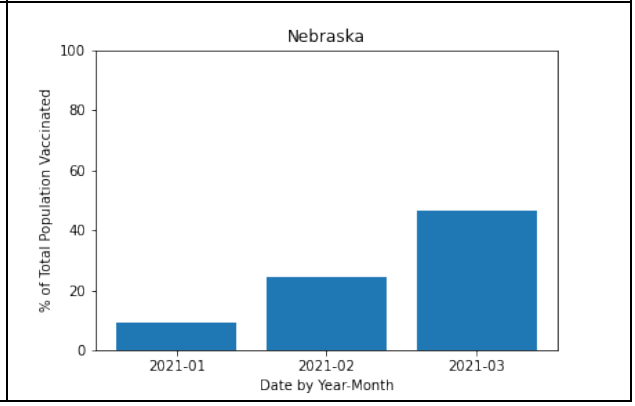

Missouri
Percentage %
Date by Year-Month
2020-12
2021-01
2021-02
2021-03
% Positive
% Negative

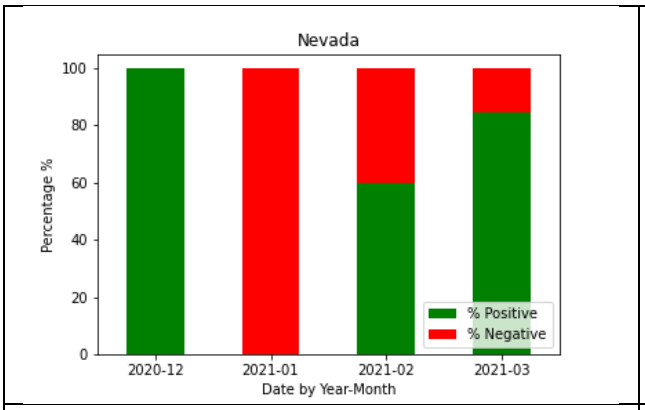

Missouri
% of Total Population Vaccinated
Date by Year-Month
2021-01
2021-02
2021-03

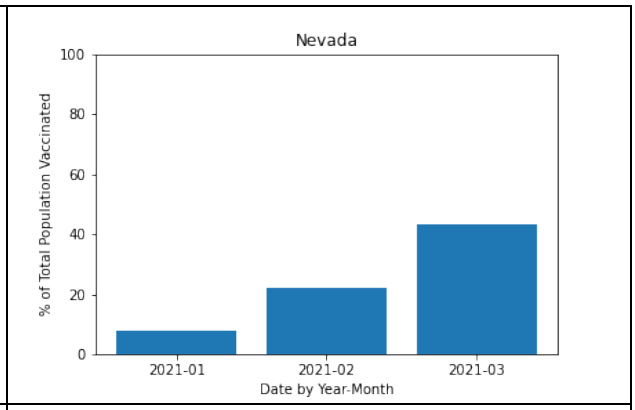

Montana
Percentage %
Date by Year-Month
2021-02
2021-03
% Positive
% Negative

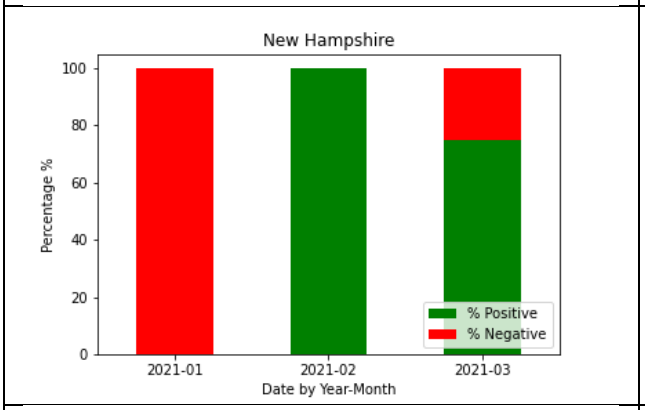

Montana
% of Total Population Vaccinated
Date by Year-Month
2021-01
2021-02
2021-03

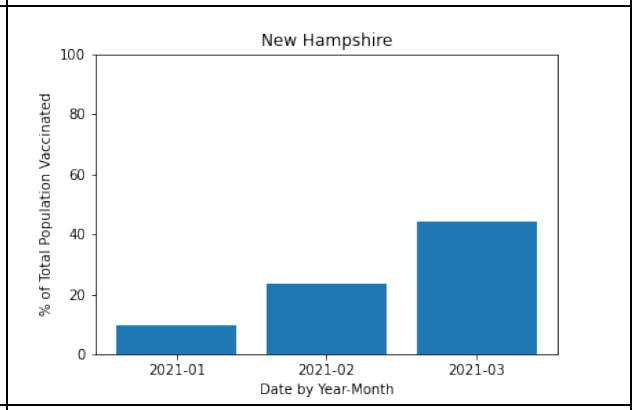

Nebraska
Percentage %
Date by Year-Month
2020-12
2021-02
2021-03
% Positive
% Negative

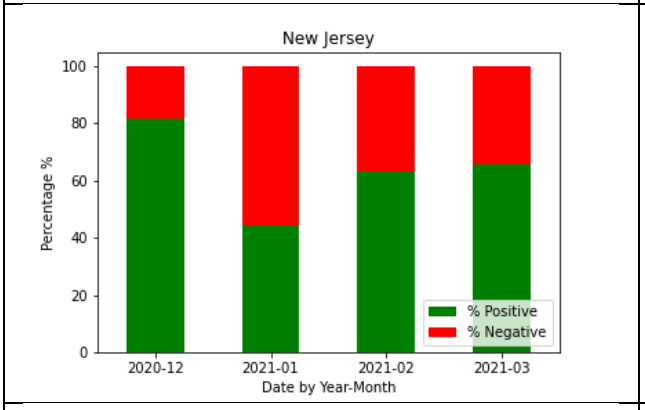

Nebraska
% of Total Population Vaccinated
Date by Year-Month
2021-01
2021-02
2021-03

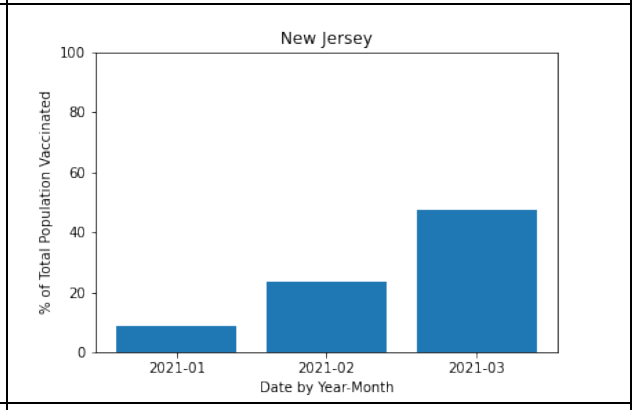

Nevada
Percentage %
Date by Year-Month
% Positive
% Negative
2020-12
2021-01
2021-02
2021-03

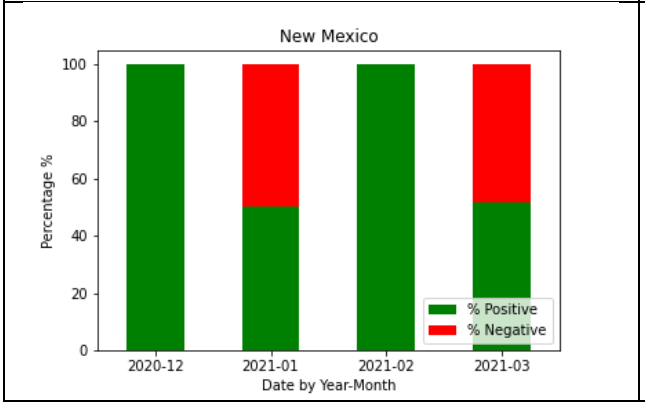

Nevada
% of Total Population Vaccinated
Date by Year-Month
2021-01
2021-02
2021-03

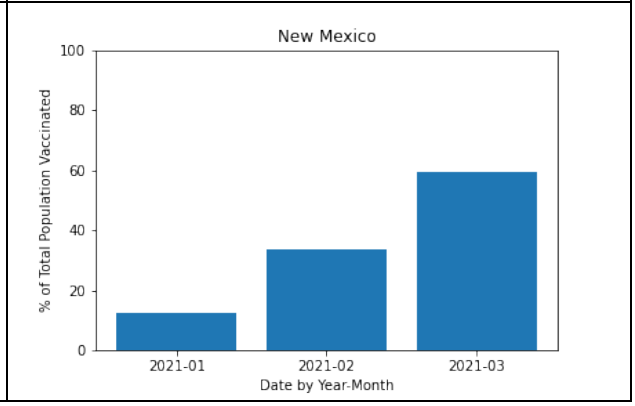

New Hampshire
Percentage %
Date by Year-Month
% Positive
% Negative
2021-01
2021-02
2021-03

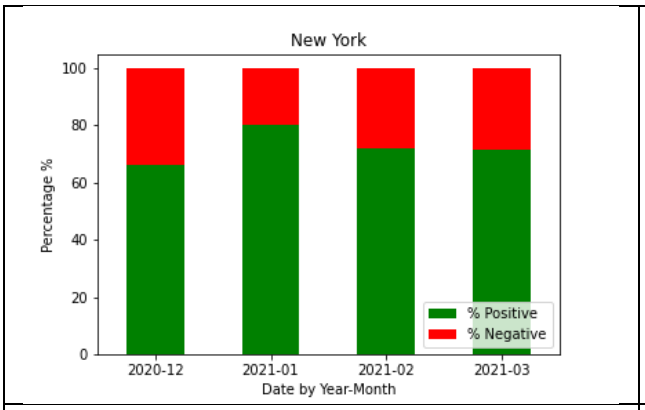

New Hampshire
% of Total Population Vaccinated
Date by Year-Month
2021-01
2021-02
2021-03

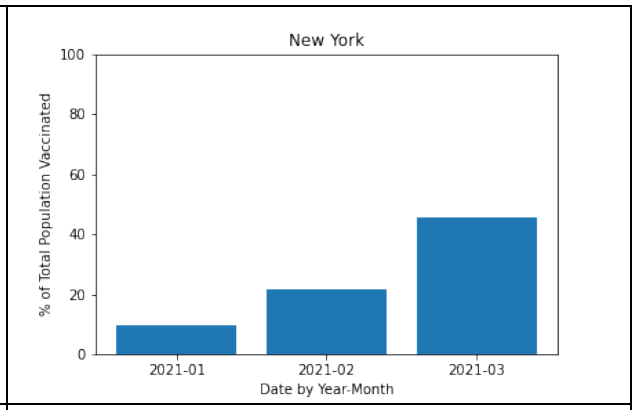

New Jersey
Percentage %
Date by Year-Month
% Positive
% Negative
2020-12
2021-01
2021-02
2021-03

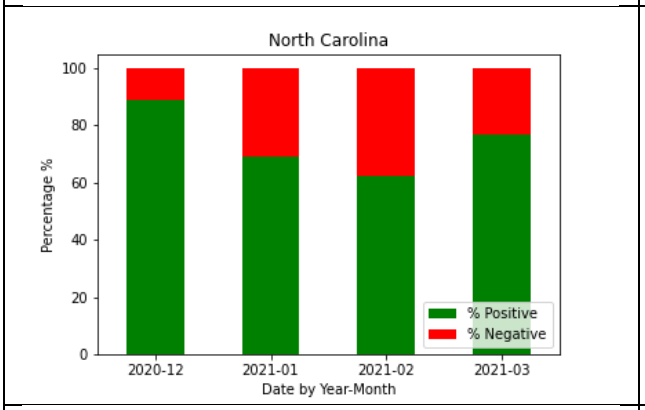

New Jersey
% of Total Population Vaccinated
Date by Year-Month
2021-01
2021-02
2021-03

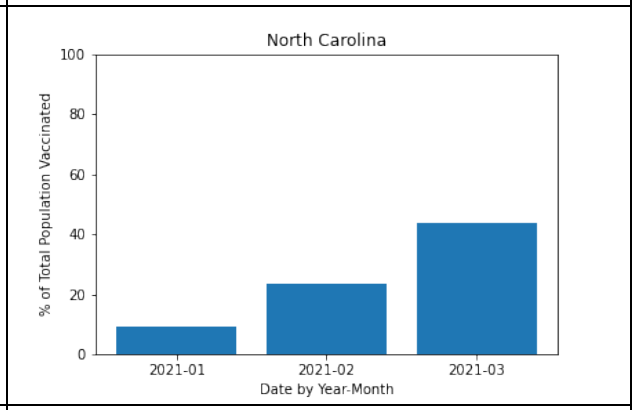

New Mexico
Percentage %
Date by Year-Month
% Positive
% Negative
2020-12
2021-01
2021-02
2021-03

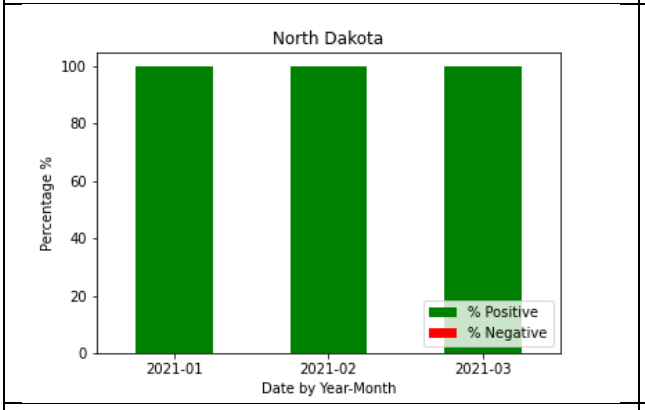

New Mexico
% of Total Population Vaccinated
Date by Year-Month
2021-01
2021-02
2021-03

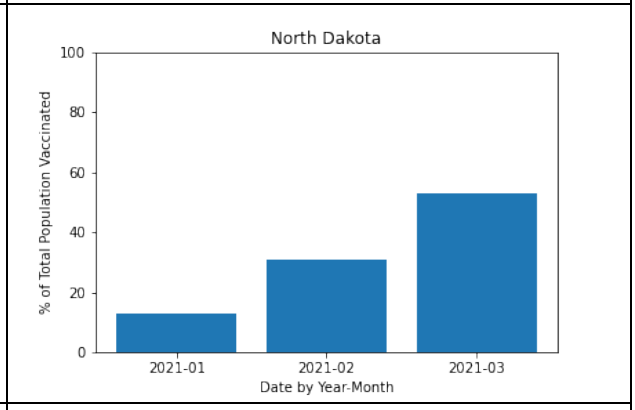

New York
Percentage %
Date by Year-Month
2020-12
2021-01
2021-02
2021-03
% Positive
% Negative

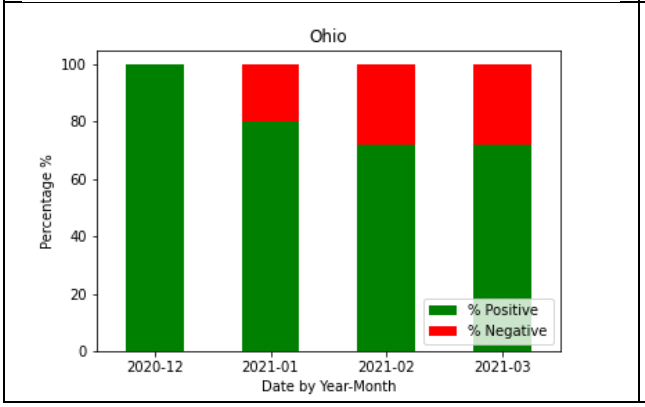

New York
% of Total Population Vaccinated
Date by Year-Month
2021-01
2021-02
2021-03

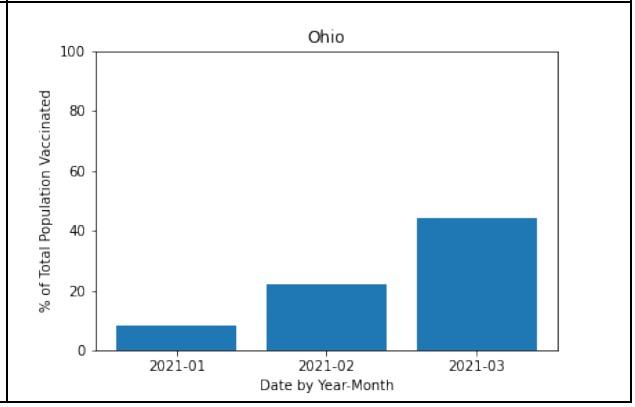

North Carolina
Percentage %
Date by Year-Month
2020-12
2021-01
2021-02
2021-03
% Positive
% Negative

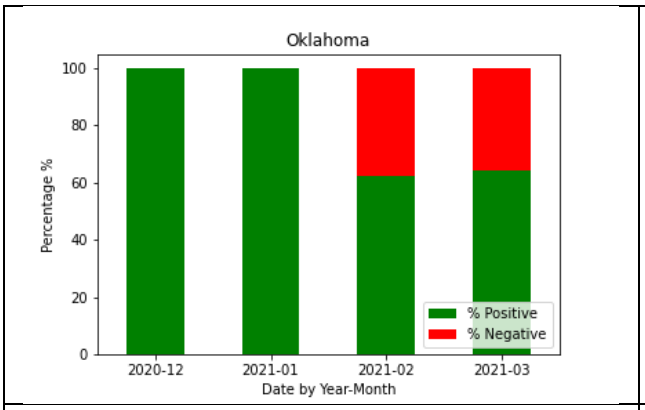

North Carolina
% of Total Population Vaccinated
Date by Year-Month
2021-01
2021-02
2021-03

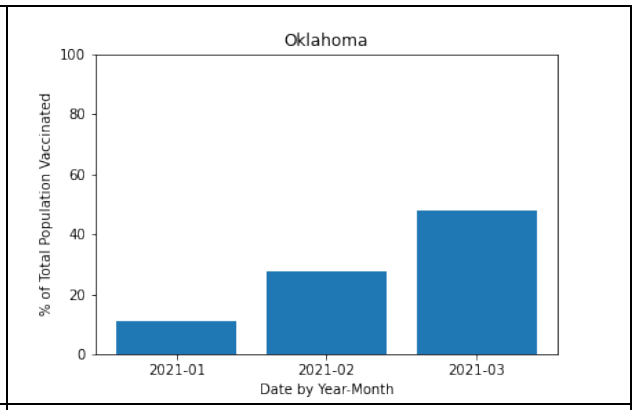

North Dakota
Percentage %
Date by Year-Month
2021-01
2021-02
2021-03
% Positive
% Negative

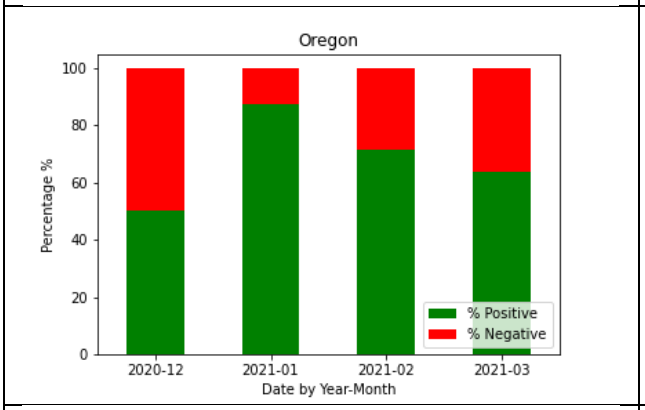

North Dakota
% of Total Population Vaccinated
Date by Year-Month
2021-01
2021-02
2021-03

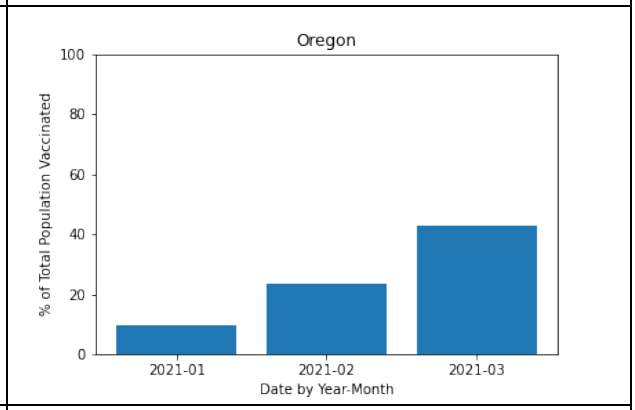

Ohio
Percentage %
Date by Year-Month
2020-12
2021-01
2021-02
2021-03
% Positive
% Negative

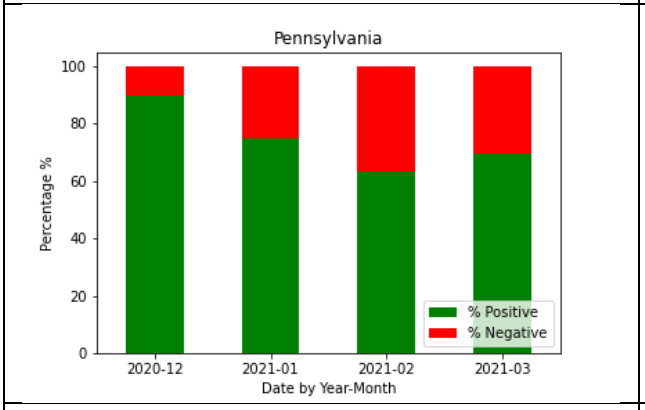

Ohio
% of Total Population Vaccinated
Date by Year-Month
2021-01
2021-02
2021-03

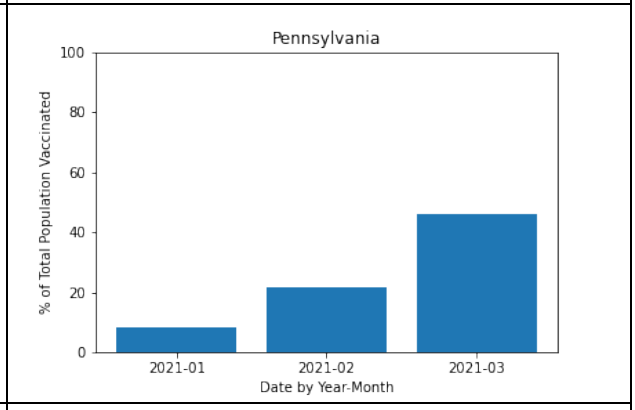

Oklahoma
Percentage %
Date by Year-Month
2020-12
2021-01
2021-02
2021-03
% Positive
% Negative

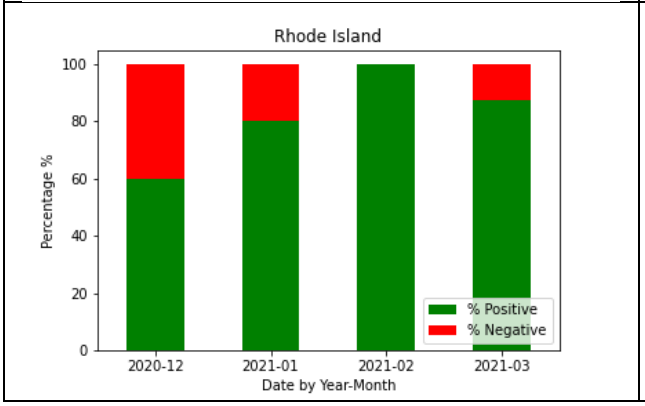

Oklahoma
% of Total Population Vaccinated
Date by Year-Month
2021-01
2021-02
2021-03

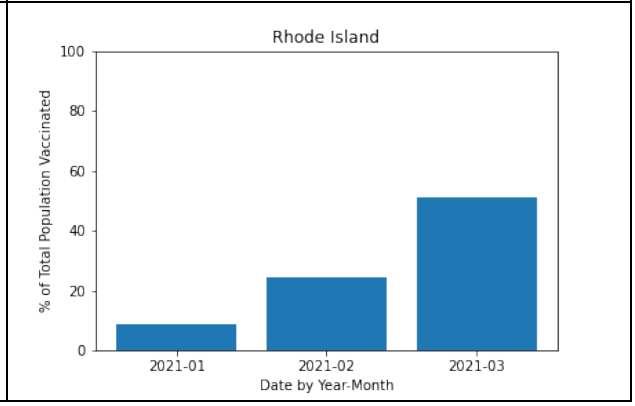

Oregon
Percentage %
Date by Year-Month
2020-12
2021-01
2021-02
2021-03
% Positive
% Negative

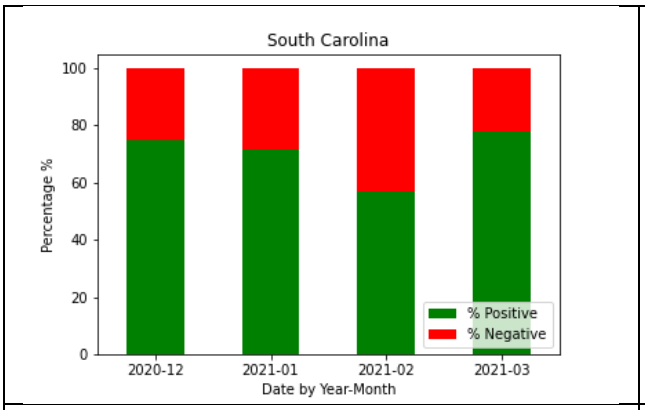

Oregon
% of Total Population Vaccinated
Date by Year-Month
2021-01
2021-02
2021-03

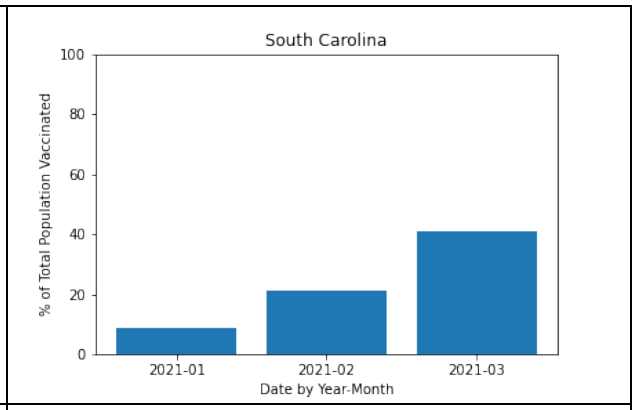

Pennsylvania
Percentage %
Date by Year-Month
2020-12
2021-01
2021-02
2021-03
% Positive
% Negative

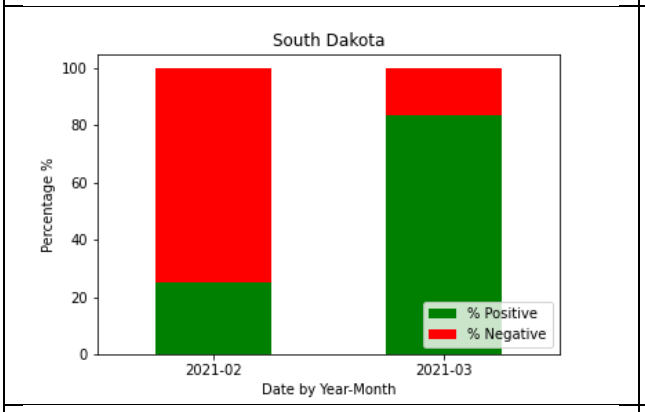

Pennsylvania
% of Total Population Vaccinated
Date by Year-Month
2021-01
2021-02
2021-03

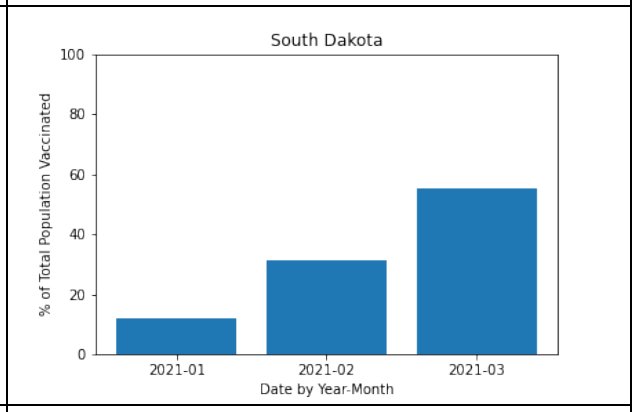

Rhode Island
Percentage %
Date by Year-Month
2020-12
2021-01
2021-02
2021-03
% Positive
% Negative

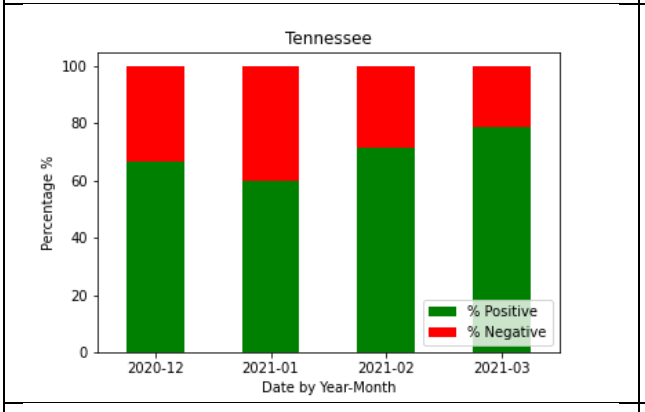

Rhode Island
% of Total Population Vaccinated
Date by Year-Month
2021-01
2021-02
2021-03

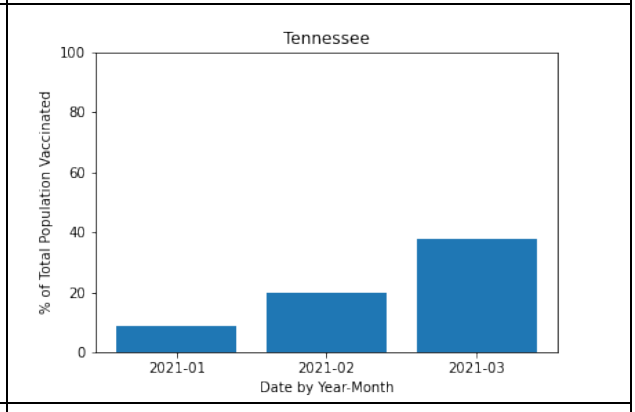

South Carolina
Percentage %
Date by Year-Month
% Positive
% Negative
2020-12
2021-01
2021-02
2021-03

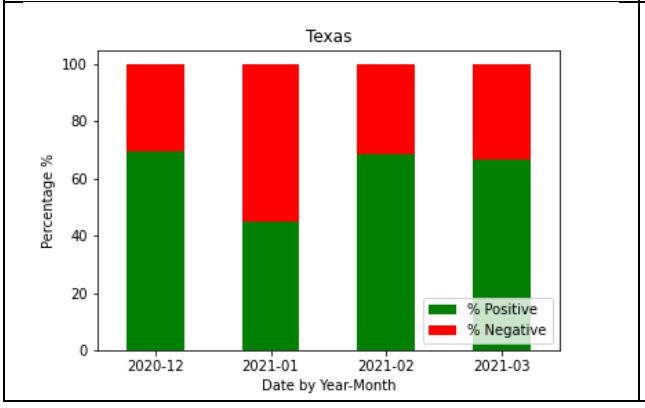

South Carolina
% of Total Population Vaccinated
Date by Year-Month
2021-01
2021-02
2021-03

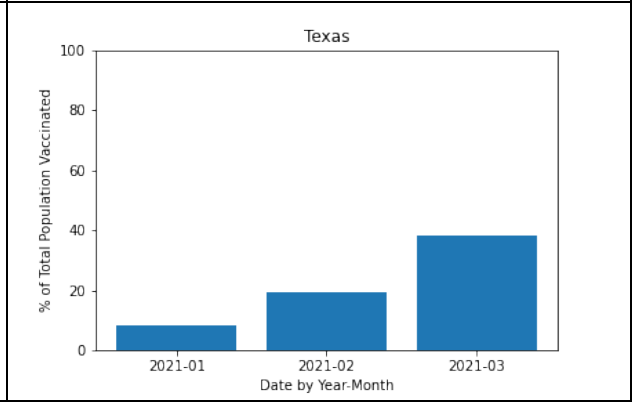

South Dakota
Percentage %
Date by Year-Month
% Positive
% Negative
2021-02
2021-03

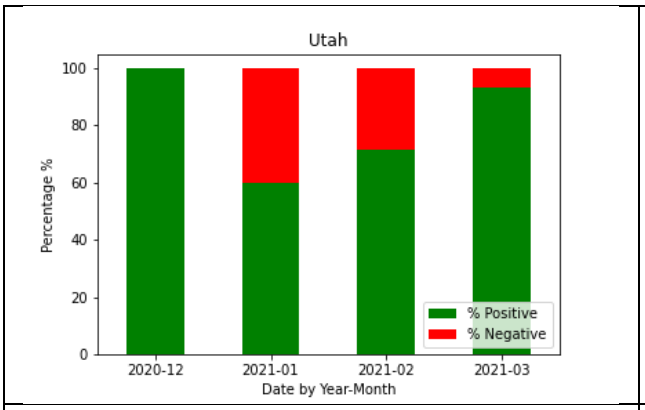

South Dakota
% of Total Population Vaccinated
Date by Year-Month
2021-01
2021-02
2021-03

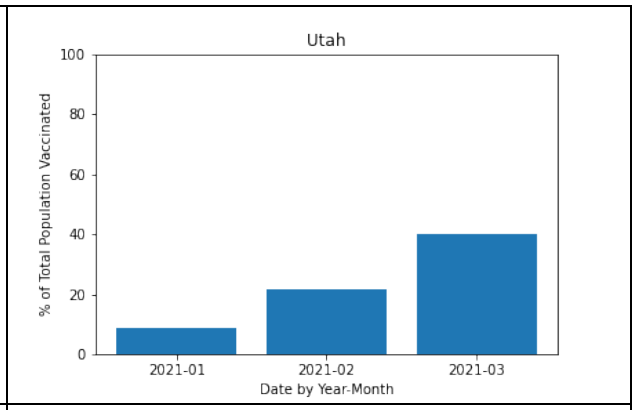

Tennessee
Percentage %
Date by Year-Month
% Positive
% Negative
2020-12
2021-01
2021-02
2021-03

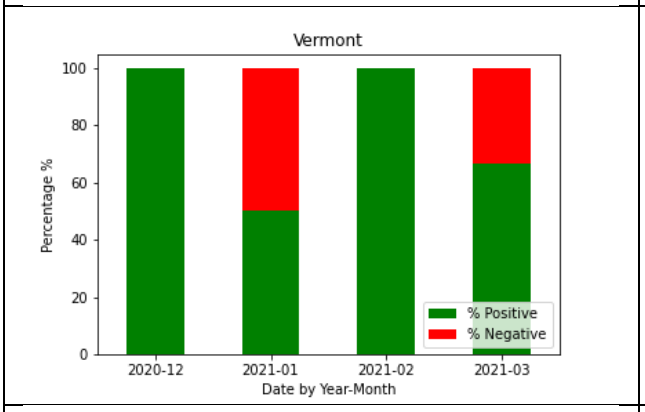

Tennessee
% of Total Population Vaccinated
Date by Year-Month
2021-01
2021-02
2021-03

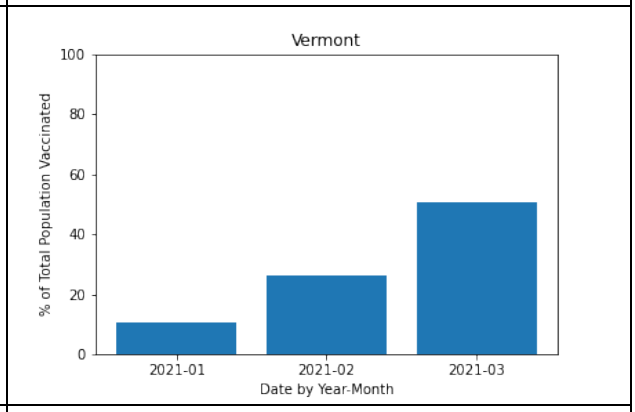

Texas
Percentage %
Date by Year-Month
% Positive
% Negative
2020-12
2021-01
2021-02
2021-03

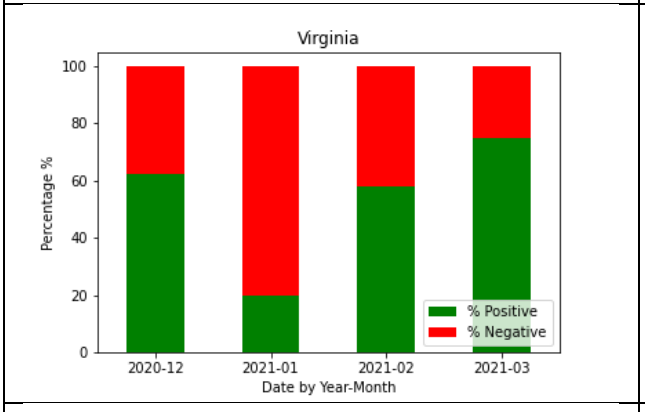

Texas
% of Total Population Vaccinated
Date by Year-Month
2021-01
2021-02
2021-03

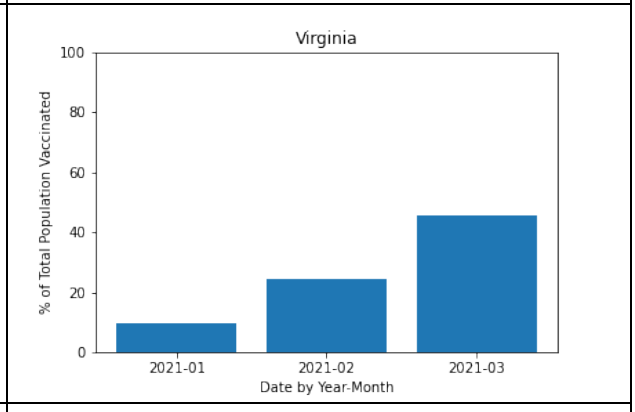

Utah
Percentage %
Date by Year-Month
% Positive
% Negative
2020-12
2021-01
2021-02
2021-03

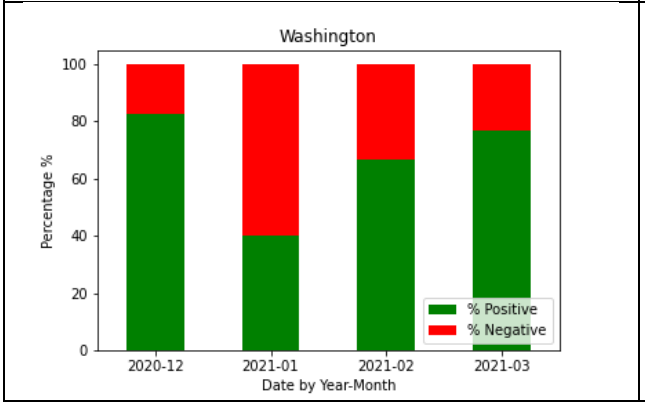

Utah
% of Total Population Vaccinated
Date by Year-Month
2021-01
2021-02
2021-03

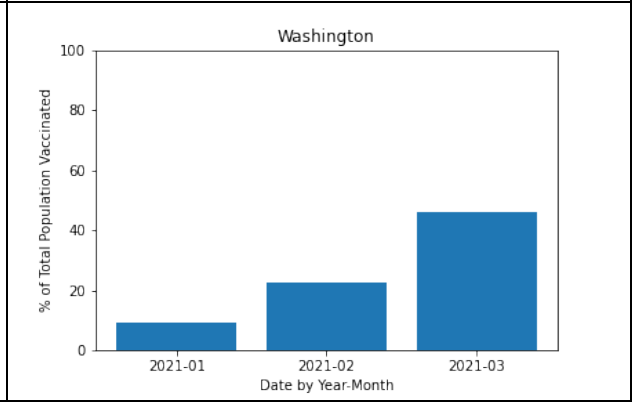

Vermont
Percentage %
Date by Year-Month
% Positive
% Negative
2020-12
2021-01
2021-02
2021-03

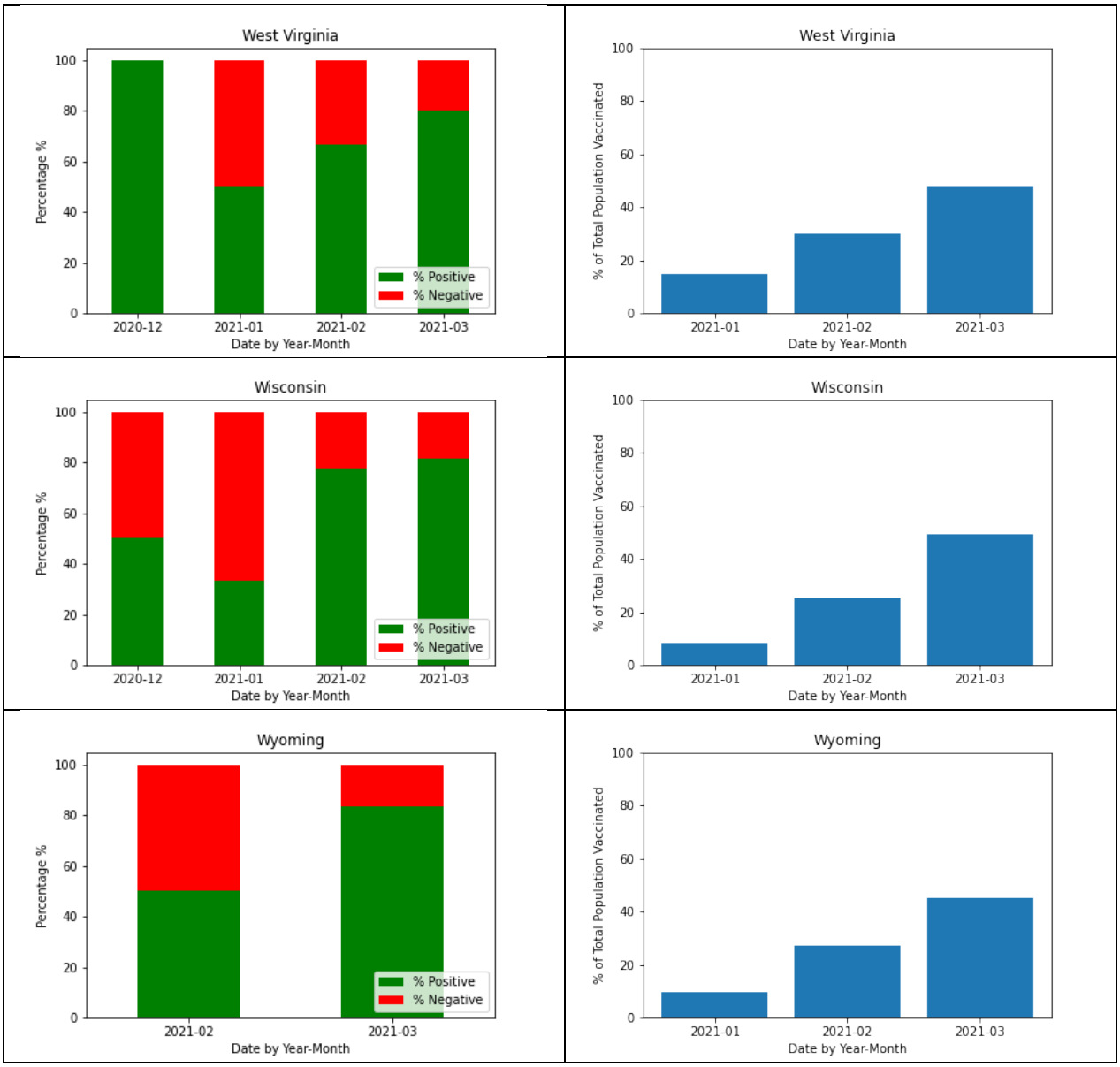

Vermont
% of Total Population Vaccinated
Date by Year-Month
2021-01
2021-02
2021-03


Virginia
Percentage %
Date by Year-Month
% Positive
% Negative
2020-12
2021-01
2021-02
2021-03


Virginia
% of Total Population Vaccinated
Date by Year-Month
2021-01
2021-02
2021-03


Washington
Percentage %
Date by Year-Month
% Positive
% Negative
2020-12
2021-01
2021-02
2021-03


Washington
% of Total Population Vaccinated
Date by Year-Month
2021-01
2021-02
2021-03



Lastly, we will also look at the difference in the overall sentiment among the various states and determine if political view plays a part in it. While we predicted that political view would play a part in the overall sentiment for each state, it seems like political view did not in fact play a huge role as we expected.

As shown from *Fig.15* below, we noticed that most states in US are very positive towards vaccination besides South Dakota, which holds strong negative attitude towards vaccination while Wyoming and Louisiana are pretty neutral towards vaccination. All three states are historically red states, whose voters predominately choose the Republican Party.

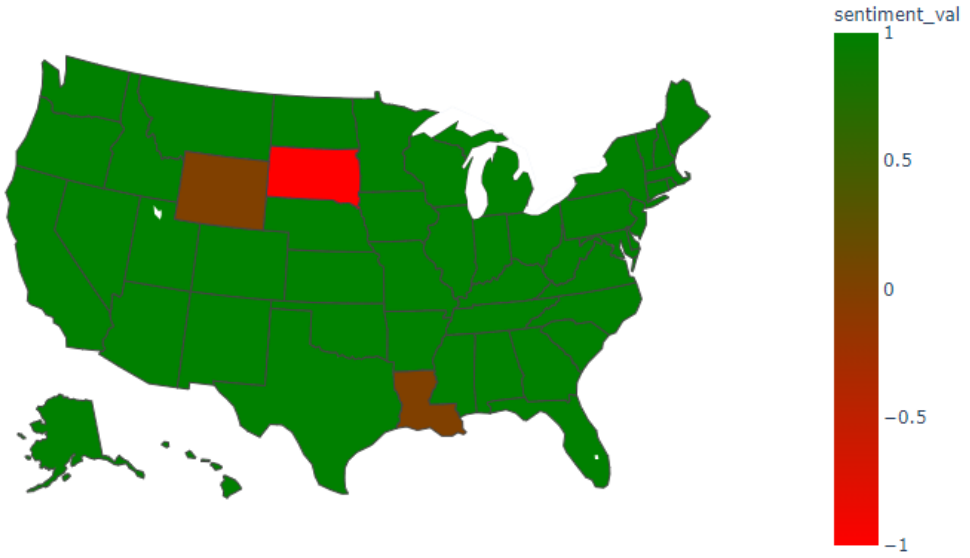


Fig 15: Choropleth Map by Overall Sentiment Value Towards Vaccine

(December 2020 – March 2021)

In *Fig.16*, we can determine which states that are on average, more positive than one another. North Dakota is darkest in colour as it was entirely positive from January to March. This is followed by states such as Nebraska, Indiana and Utah.

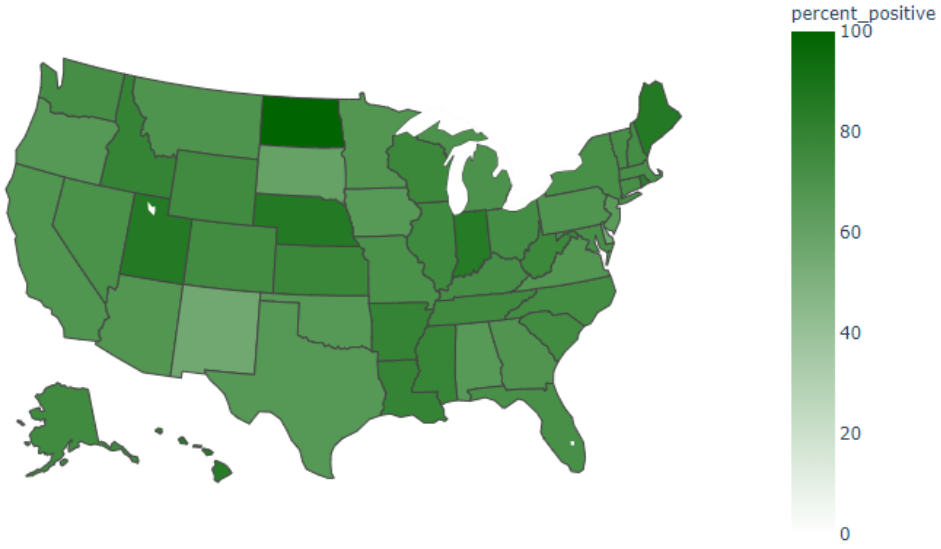


Fig 16: Choropleth Map by Average Percentage of Positive Sentiment Value Towards Vaccine

(December 2020– March 2021)

Limitations

Despite our conclusions, we should note that we had certain limitations when building our model and there are also other mitigating factors that might possibly affect the outcome of our results. For instance, the data that we collected from Twitter might not be the real reflection of how people in each state feels about the vaccine. During pre-processing of the dataset, we noticed that a lot of users did not specify their true location. After excluding these data, it only left us with 6904 rows. Admittedly, Twitter is a very popular social network platform but not everyone tends to use it as it is utilized mainly by the younger generation. Additionally, as Twitter banned President Donald Trump's account, a lot of his followers, which are Republican supporters are either banned or chose to leave the platform. This might contribute to the reason that why most tweets we collected from Twitter are overall positive towards vaccination.

Future Work

Future work can include trying out different NLP models such as the Bag-of-Words model, Term Frequency-Inverse Document Frequency (TF-IDF) model and Word Embedding model (Word2Vec) to see if accuracy of prediction will be improved.

Source of Inspiration

This project was inspired by [12].

6. Distribution of Work

Sections	Member Contribution
Data Preprocessing and Cleaning	Jocelyn, Xiyue
Fine-Tuning BERT: Hugging Face Library	Xiyue
Fine-Tuning BERT: ktrain Library	Jocelyn
Model Evaluation	Jocelyn, Xiyue
Sentiment Prediction of COVID-19 Tweets	Jocelyn, Xiyue
Analyzing Final Results: Graphs, Choropleth Map	Jocelyn, Xiyue

References

[1] “Coronavirus disease (COVID-19),” *World Health Organization*, 12-Oct-2020. [Online]. Available: https://www.who.int/emergencies/diseases/novel-coronavirus-2019/question-and-answers-hub/q-a-detail/coronavirus-disease-covid-19. [Accessed: 30-Mar-2021].

[2] “A Twitter Sentiment Analysis Tool,” *Sentiment140*. [Online]. Available: http://help.sentiment140.com/. [Accessed: 30-Mar-2021].

[3] G. Preda, “All COVID-19 Vaccines Tweets,” *Kaggle*, 27-Mar-2021. [Online]. Available: https://www.kaggle.com/gpreda/all-covid19-vaccines-tweets. [Accessed: 30-Mar-2021].

[4] *Our World in Data*, 28-Mar-2021. [Online]. Available: https://github.com/owid/covid-19-data/blob/master/public/data/vaccinations/us_state_vaccinations.csv. [Accessed: 30-Mar-2021].

[5] P. von Platen, “bert-base-uncased · Hugging Face,” *Hugging Face*, 15-Nov-2018. [Online]. Available: https://huggingface.co/bert-base-uncased. [Accessed: 30-Mar-2021].

[6] C. Sun, X. Qiu, Y. Xu, and X. Huang, “How to Fine-Tune BERT for Text Classification?,” Feb-2020. [Online]. Available: https://arxiv.org/pdf/1905.05583.pdf. [Accessed: 30-Mar-2021].

[7] S.Maiya, “ktrain: A Low-Code Library for Augmented Machine Learning,” July-2020. [Online]. Available: https://arxiv.org/pdf/2004.10703.pdf. [Accessed: 30-Mar-2021].

[8] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,” *ACL Anthology*, Jun-2019. [Online]. Available: https://www.aclweb.org/anthology/N19-1423/. [Accessed: 30-Mar-2021].

[9] L. Smith, “A Disciplined Approach to Neural Network Hyper-Parameters,” Apr-2018. [Online]. Available: https://arxiv.org/pdf/1803.09820.pdf. [Accessed: 30-Mar-2021].

[10] J. Howard and S. Ruder, “Universal Language Model Fine-Tuning for Text Classification,” May-2018. [Online]. Available: https://arxiv.org/pdf/1801.06146.pdf. [Accessed: 30-Mar-2021].

[11] Loshchilov and Hutter, “SGDR: Stochastic Gradient Descent With Warm Restarts,” May-2017. [Online]. Available: https://arxiv.org/pdf/1608.03983.pdf. [Accessed: 30-Mar-2021].

[12] Twhelan, “COVID-19 Vaccine Sentiment Analysis with fastai,” *Kaggle*, 16-Mar-2021. [Online]. Available: https://www.kaggle.com/twhelan/covid-19-vaccine-sentiment-analysis-with-fastai/. [Accessed: 30-Mar-2021].