



## **Postgraduate Certificate in Software Design with Artificial Intelligence**

### **Data Visualisation - (AL\_KSAIG\_9\_1)**

#### **Major Assignment – Visualising Twitter Data**

Student ID: A00267948

Student Name: Daniel Foth

GIT: <https://github.com/DanielsHappyWorks/Visualising-Twitter-Data>

#### Brief Description:

The assignment should contain your visualisation of the data obtained from David's assignment with

- an introduction explaining your data, its source, etc.
- a section outlining what software you used to create the visualisation
- the visualisations themselves with a short analysis for each visualisation.
- a conclusion bringing all your results together.

## Contents

Introducing the data .....	3
Program Design .....	4
Design Diagram .....	4
The Program Files.....	4
code/Tweet_scraper.py .....	4
code/Model_generator.py .....	4
code/Graph_data.py .....	5
code/Tweet.py .....	5
Visualisations.....	5
Original Data Graphs .....	5
All Data.....	5
Test Data .....	7
Model Prediction Graphs .....	9
Random Forest Classifier.....	9
All Model Data .....	11
Unseen Data Graphs .....	13
Conclusion .....	16

## Introducing the data

The data set is stored as data.csv in the code directory. It contains 513 of the 1500 tweets scraped by the tweet\_scraper.py script using the Twitter API. Some tweets have been removed as they were illegible and manually annotating that many tweets is time consuming, so the data had to be limited.

The tweet data that was saved consists of:

1. *Tweet ID*
2. *Creation Time*
3. *Retweet Count*
4. *Favourite Count*
5. *Source*
6. *Username*
7. *User Location*
8. *Content*
9. *Basic Content* – *this is the same as the content but stripped of special characters so it can be used with a variety of models.*

The two extra columns annotated manually are:

10. *Emotion* – *defines the emotion that stands out most in the tweet*
11. *Polarity* – *a rating of whether the tweet is Positive, Neutral or Negative*

Once models are created and data gets predicted, these get stored as .pkl files in the /output/model\_data directory. They contain the tweet, model prediction, and Vader sentiment analysis prediction.

The unseen data used for testing the model predictions against Vader is always retrieved live. Once predictions are made, they get stored as .pkl files for graphing. Alternatively, they can be opened using the pandas “read\_pickle” function. These have the same properties as data.csv

The topics of interest include:

Politics:

1. *Donald Trump*
2. *Boris Jonson*

State of the world:

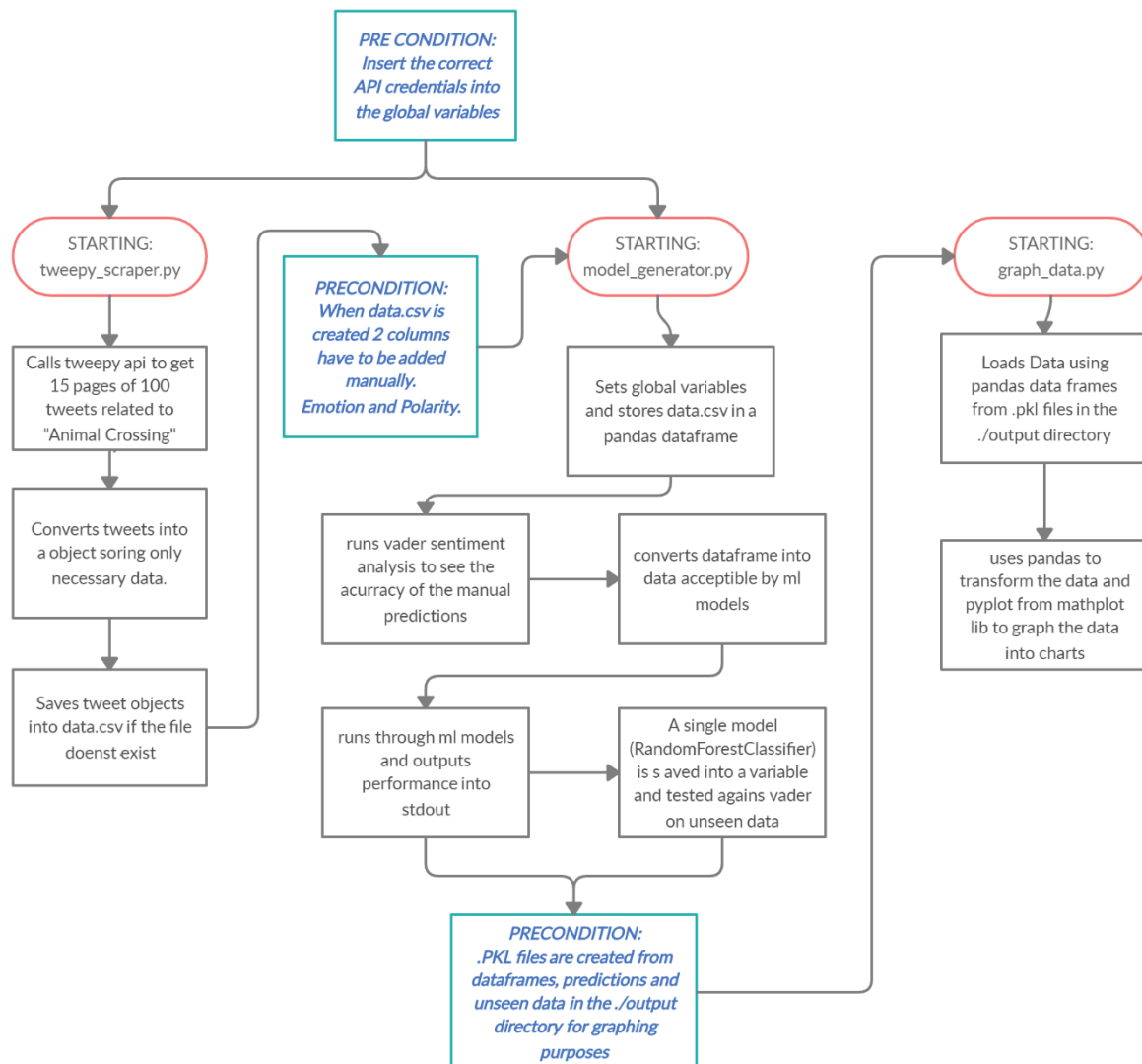
3. *Covid-19*
4. *Isolation*

New Technology

5. *Virtual Reality*
6. *Ps5*
7. *Xbox Series X*

## Program Design

### Design Diagram



This figure describes the general flow of the program from start to finish.

(Created with <https://creately.com/>)

### The Program Files

`code/Tweet_scraper.py`

It is responsible for scraping tweets from the Tweepy API. It can scrape up to 100 tweets per page. When run it will generate a data.csv file with 1500 tweets on the topic of “Animal Crossing” if it doesn’t already exist.

`code/Model_generator.py`

It will create models and run one against multiple categories on unseen data. It uses the data.csv file. It requires that 2 extra columns are added in manually; the Emotion and Polarity column. The predictions made using the scikit learn library, get stored as data frames into .pkl files along with the tweet data for graphing purposes.

code/Graph\_data.py

It graphs the data in the output/ directory and stores the graphs as png files under output/graphs.

The tools used for graphing are pandas data frames to format and audit the data from .pkl files and matplotlib & plotly which are a charting library for python.

Matplotlib & plotly can plot many kinds of graphs including bar, histograms, line, treemap, sunburst and scatter plots.

[code/Tweet.py](#)

The Tweet class processes a line of tweet data from the API into data that can be used as part of the model creation and graphing. See the data section to understand what is extracted.

## Visualisations

From the data retrieved from twitter there are multiple categories of graphs generated. All of which are stored under output/graphs/

These include:

- Original Data Graphs
- Model Prediction Graphs
- Test Data Graphs

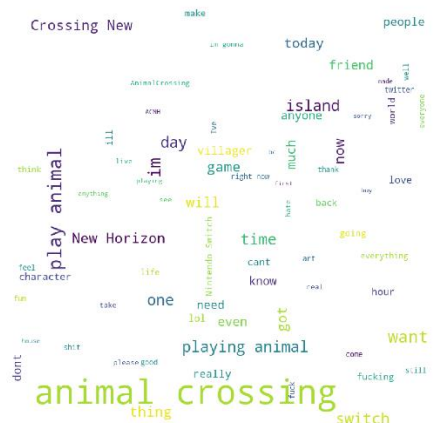
Not all graphs generated will be discussed as this would make the document too long.

## Original Data Graphs

The Original Data graphs consist of 513 rows of data scraped from the Twitter API and prepared for model generation. These graphs are stored in `output/graphs/original_data` and have two categories (`original_` and `test_` both) containing the same graphs.

## All Data

The original\_ graphs visualise the entire data set including the training and test data for the models.



This word cloud isn't perfect as the dataset is very big and these words repeat so often that a lot of others don't even make it into the cloud.

Due to the contrast, it is also a little hard to read when shrunk down into this small a size for the word document. It needs to be opened to be viewed properly.

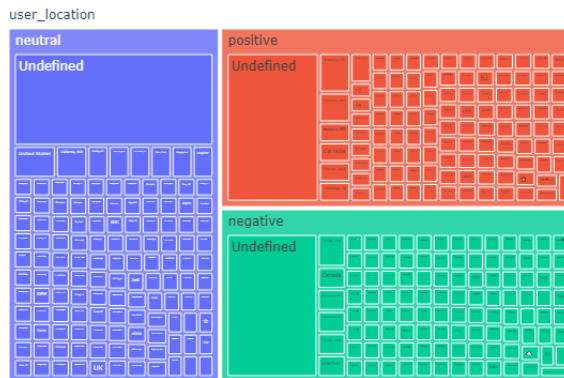
The general language seems to have a positive tone implying that tweets for this

	<p>topic are most likely positive too.</p>
	<p>Here we have a simple line graph that implies that more retweets tend to be proportional to more favourites. This makes sense as the tweet would be seen by more people.</p> <p>Unfortunately, there isn't many tweets above 20 retweets which could be skewing the graph.</p>
	<p>Here we can see the number of users using different apps to post on twitter and comparing it to see if it affects sentiment.</p> <p>Since proportions seem similar in all areas it implies that the device you have doesn't have any impact.</p> <p>We can see that most people use twitter from their iPhone though.</p>
	<p>This treemap is doing a similar job to the sunburst chart above but performs worse as all squares are subdivisions that aren't fully scaled to the count of data in them.</p> <p>To see the interactive chart (treemap and sunburst) with tooltips, run <code>graph_data.py</code> with lines 119 and 104 not commented out.</p>



Like the previous sunburst chart, we're trying to see if location impacts how positive or negative a tweet is. This chart isn't perfect as there are a lot of locations in the dataset, some of which are fake. This makes the chart harder to read.

Open the .svg to see the details better as the image in the document is too small.

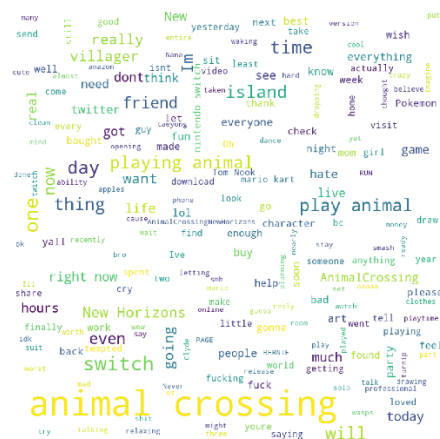


Once again, this treemap is doing a similar job to the sunburst chart above but performs worse as all squares are subdivisions that aren't fully scaled to the count of data in them.

There are too many countries to see this properly without opening the svg or running `fig.show()` within the code. To see the interactive chart (treemap and sunburst) with tooltips, run `graph_data.py` with lines 119 and 104 not commented out.

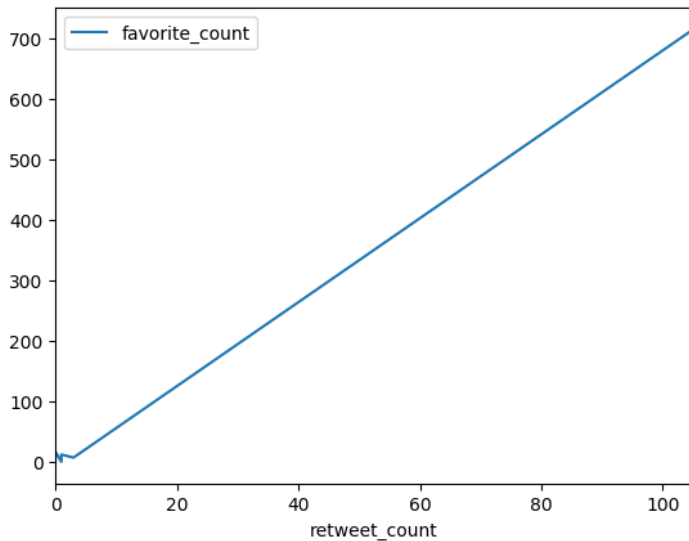
## Test Data

The test\_ graphs visualise the data used to test the models for use on unseen data.



The test data wordcloud is better as it generally shows us more of the less commonly used words.

There are still some issues with the contrast that make it hard to read when small, but it gives us better visibility of what is contained within tweets.

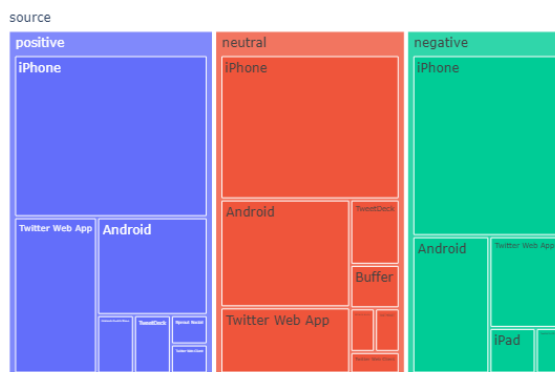


Once again, the same thing is implied, more retweets tend to be proportional to more favourites, but this is hard to evaluate with limited data.



With less data, these sunbursts and treemap graphs are easier to analyse but give us less information about the overall dataset.

In terms of results once again, there is no correlation between device and sentiment.

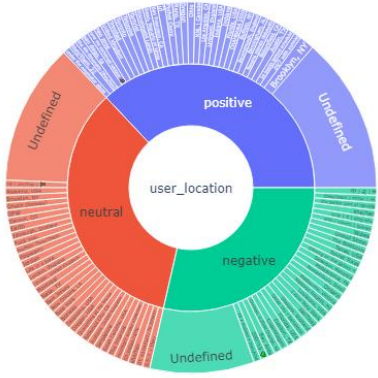
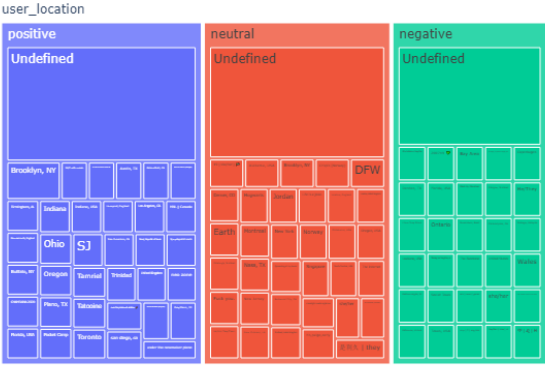


With less data these sunbursts and treemap graphs are easier to analyse but give us less information about the overall dataset.

The sunburst is still more accurate than the treemap but a less crowded treemap can be easier to follow.

In terms of results once again, there is no correlation between device and sentiment.

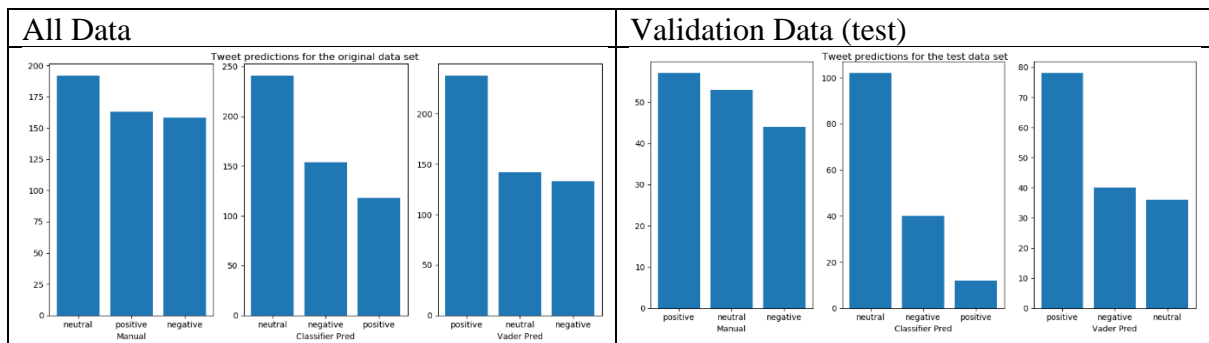


	<p>With less data, these sunbursts and treemap graphs are easier to analyse but give us less information about the overall dataset.</p> <p>In terms of results once again there is no correlation between location and sentiment.</p>
	<p>With less data, these sunbursts and treemap graphs are easier to analyse but give us less information about the overall dataset.</p> <p>The sunburst is still more accurate than the treemap but a less crowded treemap can be easier to follow.</p> <p>In terms of results, once again there is no correlation between location and sentiment.</p>

## Model Prediction Graphs

There is a set of diagrams for the 5 models created in `output/graphs/model`. The one I will focus on is the `random_forest_classifier` as it was chosen for the prediction of unseen data. There is also a set of graphs for the models overall available in `output/graphs/model/all_model`.

## Random Forest Classifier



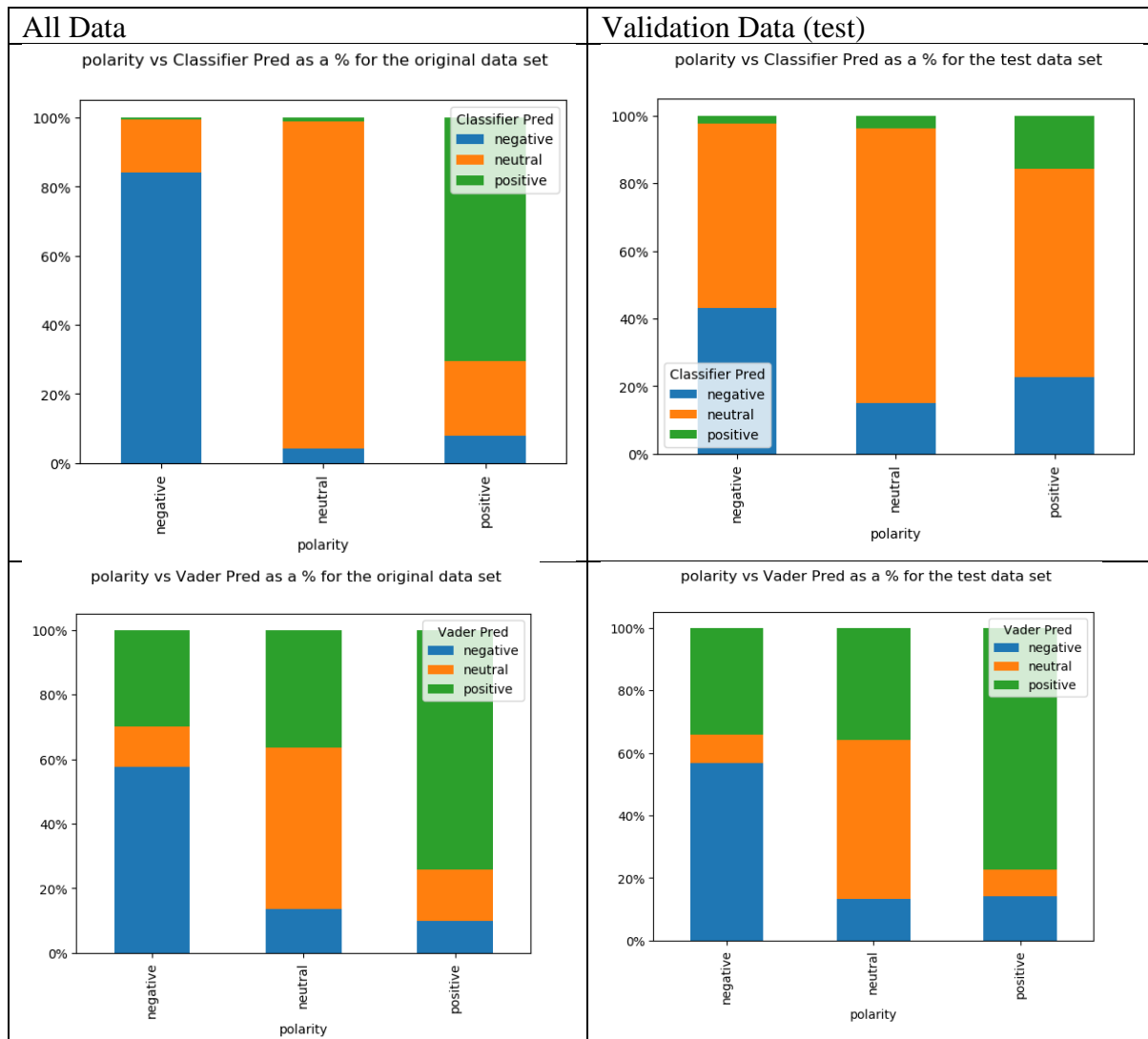
In the above diagrams we can see what kind of predictions were made by each method:

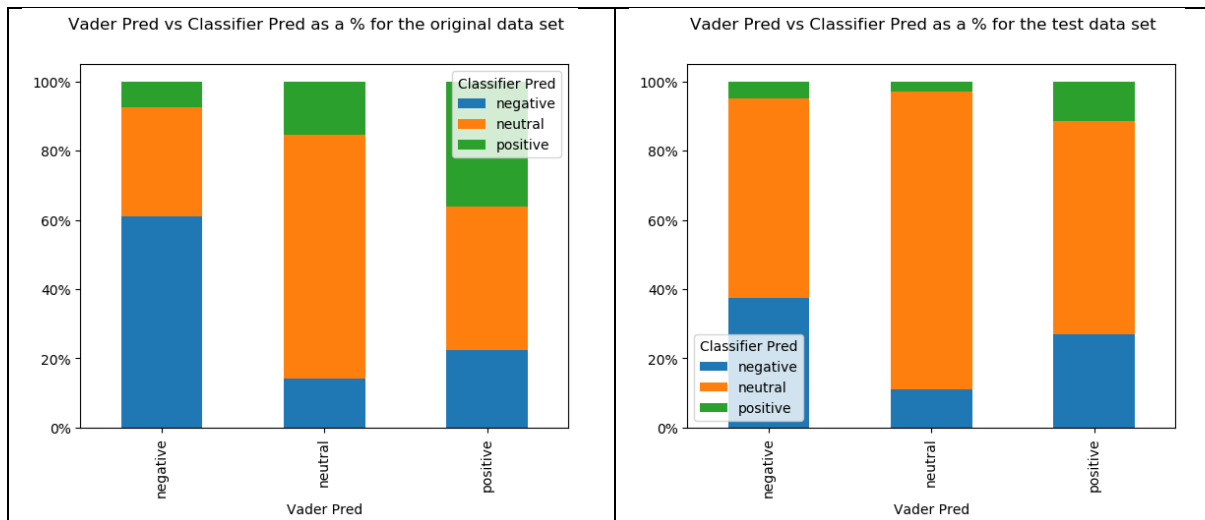
1. Manual -> annotated by me manually
2. Classifier Pred -> annotated by my model based on the RandomForestClassifier from scikit learn

### 3. Vader Pred -> annotated by the Vader Sentiment Analysis tool.

From those diagrams we can tell that my model and the sentiment analysis tool played it safe by mostly categorising tweets as neutral. This is especially true for my model against the test dataset.

These diagrams are far from perfect as each subplot has a different range, making them hard to analyse side by side. Using a percentage value could have solved this issue but it wouldn't have given an exact counter.





These diagrams compare how well the model predicted sentiment based on my manual analysis. It's a well-made plot that's easy to look at but can be very complicated as there is a lot of data in it. Using a % fixes the issue we have with the above graph, making all these graphs easy to compare no matter how big or small the dataset is.

The rows show data:

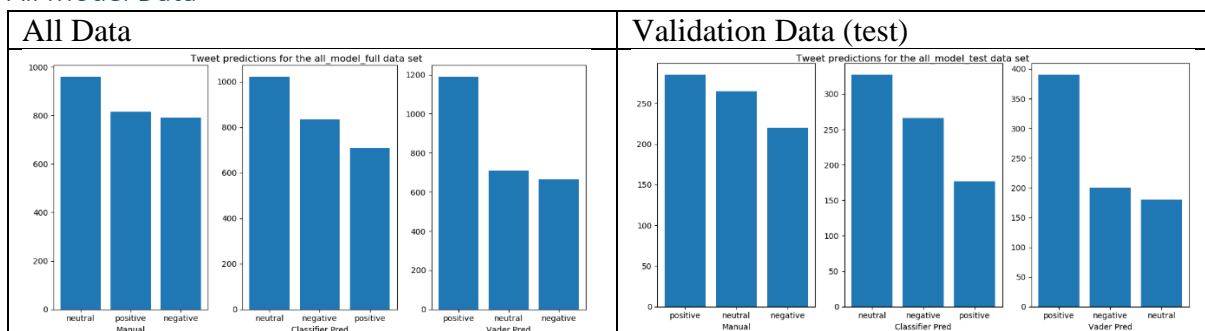
- First row -> manual predictions vs RandomForestClassifier
- Second row -> manual predictions vs Vader Sentiment Analysis
- Third row -> Vader Sentiment Analysis vs RandomForestClassifier

Looking at the first row as an example, from the first graph we can see that a lot of the tweets I marked as negative were predicted as negative for 60% of the tweets by the RandomForestClassifier and a lot of the neutral ones were marked as neutral.

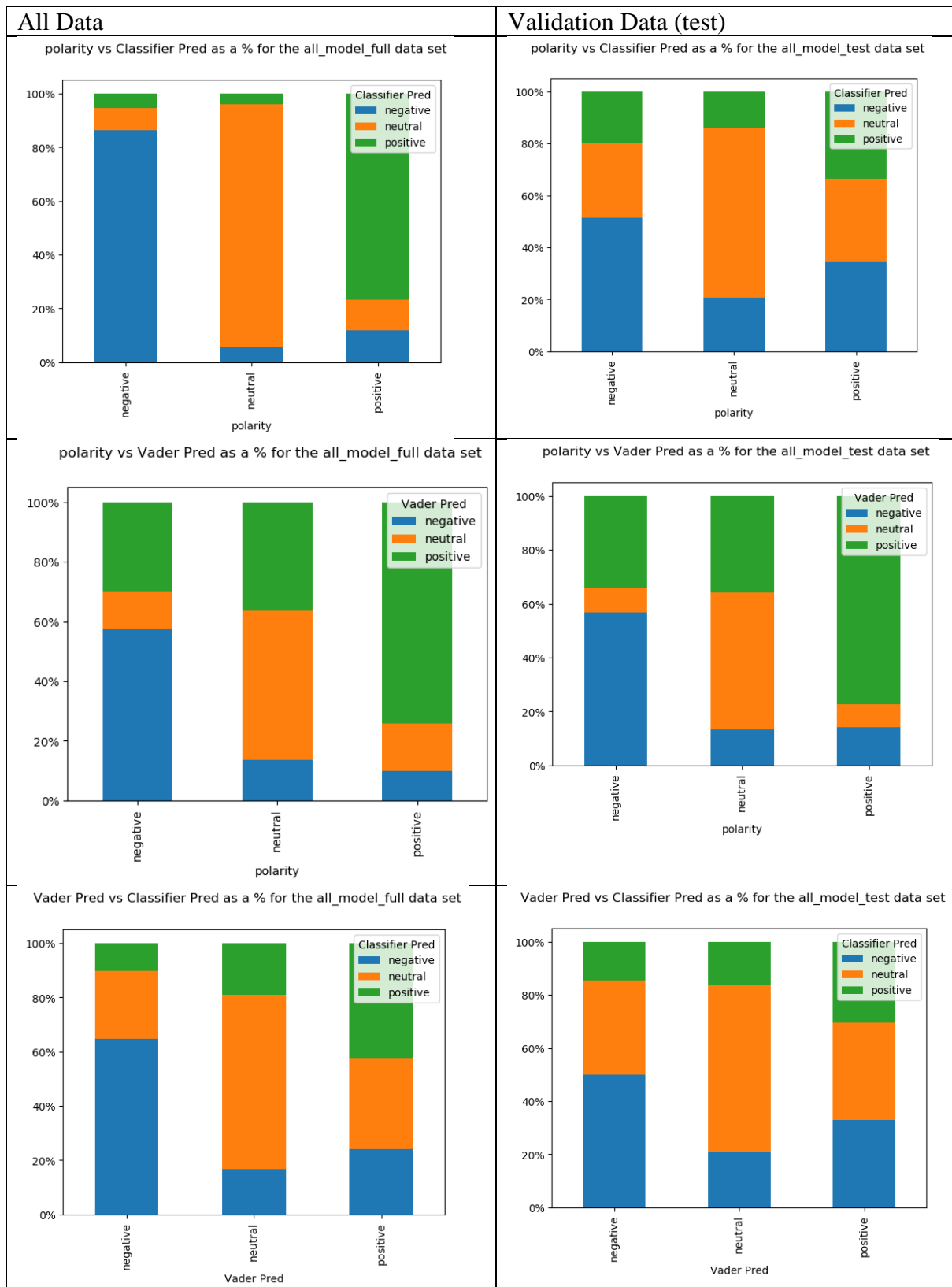
From the second graph overall, if we look at the orange colour, most of the graph is taken by it. From this, we can see that across the board, most predictions were neutral on the dataset.

A lot of different conclusions can be reached using these graphs, making them very versatile. Using a version of this kind of graph with unstacked histograms could make telling the exact percentage of each column a bit easier but this makes it more condensed and cleaner.

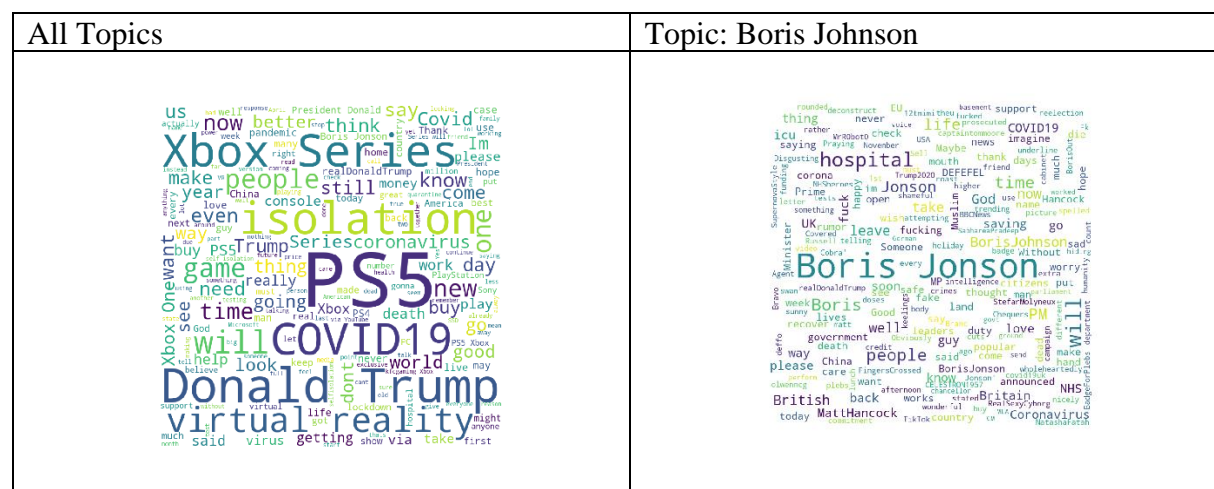
## All Model Data



The evaluation for this graph is like the one in the section above. It has a few issues but gives us a global understanding of how the tweets are classified. In this case, we can see the predictions for each of the five models used while testing. The biggest issue here is that we have no indication of how accurate the predictions are. We only know the numbers.

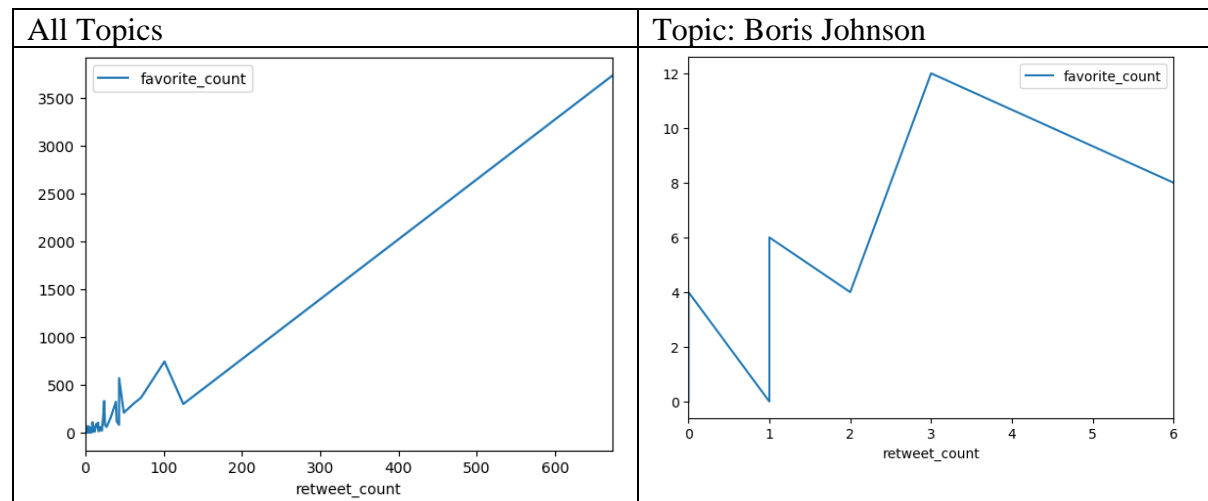


These graphs can once again give us a lot of details. In this case, it is in relation to all five models. When all models are making predictions, the predictions for all the data tend to hover around the same for test data in terms of inaccuracy.



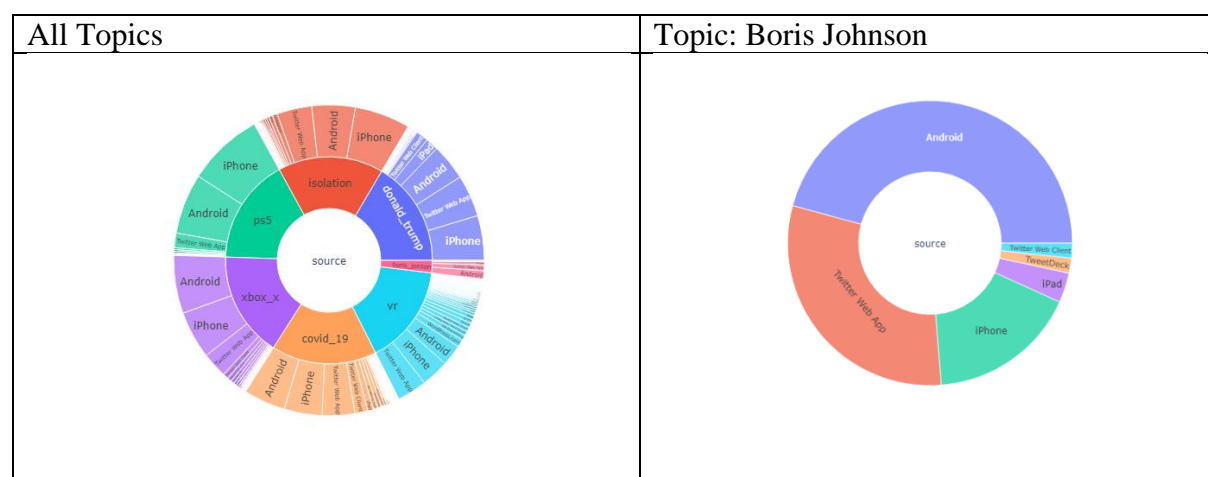
The word cloud that describes all the unseen data doesn't really make much sense as all of the topics are so different but the one about Boris Johnson seems to be on point, especially with him falling ill and it being reflected in the word cloud.

There are still some issues with the contrast that make it hard to read when small, but these two-word clouds are way more readable than the ones above. This seems to be a general issue with the word cloud library as it seems to assign colour randomly.



Form all data, we can see a general upward trend when plotting favourites against retweets but from the Boris Johnson tweets, that seems to only vaguely apply as the number of tweets is a lot more limited.

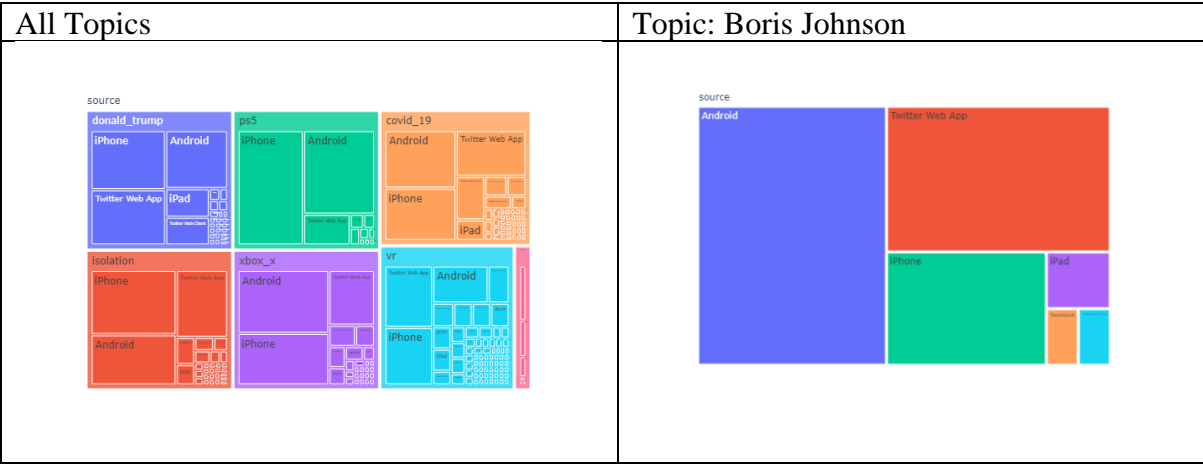
Even with all the unseen tweets, it is hard to tell if this trend is accurate as usually there is an outlier that has way more retweets and favourites than all others. Maybe a graph that can look at the averages could tell us more about this phenomenon.



With less data, these sunbursts graphs are easier to analyse but give us less information about the overall datasets. Here we have two sunburst graphs that represent this very well. One is filled with data about different topics which makes it harder to read. The other one is more limited but very easy on the eyes.

It's good to have access to both as you can use the "all topics" graph to quickly make comparisons but use the more detailed singular chart to zone in on the fields that might not be as visible in the "all topics" graph.

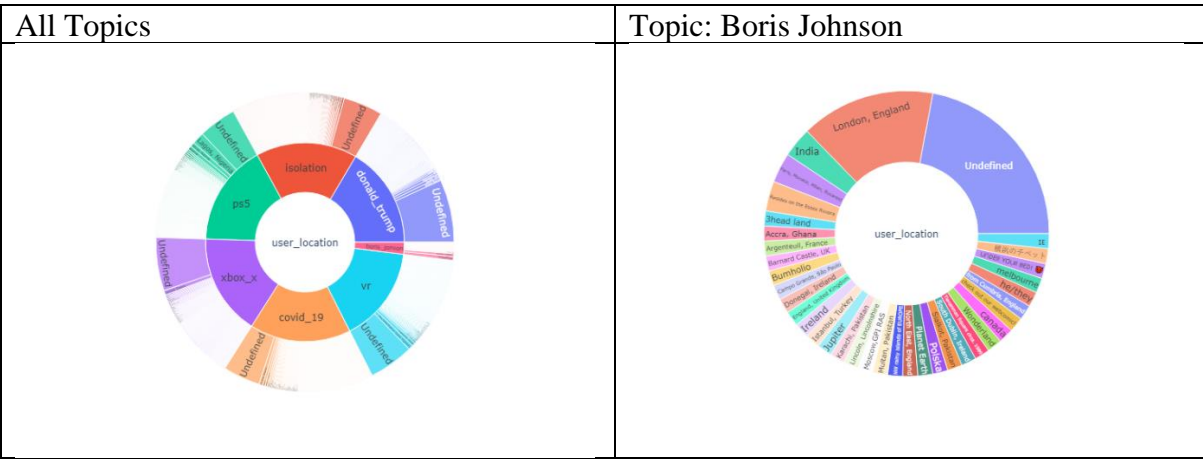
There does seems to be a difference in the devices used by people over different topics. This is a development that we couldn't see with just the one dataset earlier.



With less data, these treemap graphs are easier to analyse but give us less information about the overall datasets. Here we have two treemap graphs that represent this very well. One is filled with data about different topics which makes it harder to read and the other one is more limited but very easy on the eyes.

It's good to have access to both as you can use the “all topics” graph to quickly make comparisons but use the more detailed singular chart to zone in on the fiends that might not be as visible in the “all topics” graph.

There does seems to be a difference in the devices used by people over different topics. This is a development that we couldn't see with just the one dataset earlier.



This is like the source sunburst chart. The only difference is that the “all topics” graph is too cluttered to make sense of as there are too many fields. This makes it less usable.

In terms of location, we can see that the different kinds of tweets tend to be mostly seen in specific locations, for example a lot of tweets about Boris happen in England.





tweets was like based on the word cloud, or how many people use different devices to access Twitter, in this case iPhones being the mostly used.

The sheer number of visualisations made programmatically can make it hard to find your way around the directory full of images but being able to see things in separate visualisations allows for better analysis overall with more freedom to see what does and doesn't work for different kinds of data very quickly. There are some quirks to making these kinds of graphs. One example is sunburst and treemaps which use the plotly library. These are better viewed by running them via code rather than exporting to images, as being able to open them in the browser lets you see them more clearly with the correct tooltips on hover.

To see the interactive chart (treemap and sunburst) with tooltips, run `graph_data.py` with lines 119 and 104 not commented out.