Birkbeck, University of London

Department of Computer Science and Information Systems

A Community Analysis

of the UK Government Cats' Twitter Accounts

Pravin Jeyaraj

MSc Data Science

September 2019

Dr Alessandro Provetti

Project Supervisor

Table of Contents

List of Figures

Abstract

There is evidence that various United Kingdom government departments have used cats to catch mice since the 19th Century. Official records go back to 1929 and recently released records indicate that the government cats have always had some form of public following and their behaviour has generated media interest. The current generation of government cats are the first to have a presence on Twitter, and on social media generally. The aim of this project, therefore, is to develop a contemporaneous understanding of the UK government cat community. This achieved by using Python scripts and Twitter's Application Programming Interface to undertake a network analysis of the government cats connected to their friends and followers, a textual analysis of government cat tweets and textual analyses of the locations and profile descriptions of the government cats' followers.

Project Supervisor: Alessandro Provetti

1. Introduction

This chapter introduces the background of our data science project. Watching cats, especially when they are made to do human like things, has become known as one of the most popular pastimes on the internet. More cat pictures are shared online than selfies.[1] In an online survey completed by almost 7,000 Facebook and Twitter users, Jessica Myrick found that the average frequency for looking at internet cats was two to three times a week to daily.[2] It is not surprising, therefore, that the United Kingdom's government's own cats have their own internet presence in the form of Twitter accounts. The existence of government cats predates Twitter and the internet and has a fascinating history.

Motivation: A brief history of the UK government cats

It is not surprising that United Kingdom government departments would use cats to catch rats or mice. National Archives records indicate that there have been government department cats since the 19th century.[3] What seems surprising, however, is the level of political interest and media interest in the actions of the cats.[4] For example, when David Cameron resigned as Prime Minister in 2016, one of the questions that he had to answer at his final Prime Minister's Questions in the House of Commons concerned his feeling about Larry, the Downing Street Cat:

> "The rumour that somehow I don't love Larry, I do…I can't take Larry with me, he belongs to the house [10 Downing Street] and the staff love him very much, as do I."[5]

[1] Porter, 2016
[2] Myrick, 2015, 171
[3] Day, 2016
[4] The Telegraph, 2017; Kennedy, 2016; Scott, 2017
[5] Chambre, 2016

Cameron's response as well as the fact that such an issue was raised by MPs at a time of political crisis, in the aftermath of the referendum on European Union membership, gives an idea of the popularity of Larry the Cat. Tony Blair also experienced a similar question as Prime Minister, with regard to the then Downing Street Cat, Humphrey:

> "Humphrey the cat's abrupt departure from Downing Street six months after Labour won the election took the gloss of Mr Blair's landslide victory. The spin machine insisted the ailing moggy had retired. Opponents claimed he had been exiled or even put down on the orders of the ruthless Alistair Campbell because Cherie Blair hated cats…Mrs Blair, who was reported as finding cats unhygienic, was pictured posing with Humphrey…Smiling for the cameras Mrs Blair gave a not altogether convincing display of affection. Humphrey appeared terrified."[6]

As it turned out, Humphrey was already quite old and he had been "pensioned off to a new home near London".[7] Nevertheless, this didn't stop questions being asked in the House of Commons and proof being demanded that he was still alive.[8] The potential relationship between the Downing Street cat and every Prime Minister since Edward Heath has been a subject of media interest and it has been commented that, when Boris Johnson became Prime Minister in 2019, his first job was to "impress the cat".[9]

Official records of the government cats began in 1929, when applications were made by the respective government department to the Treasury for a "feline upkeep allowance".[10] However,

[6] Millward, 2005
[7] BBC News, 1997
[8] BBC News, 2006
[9] Sleator, 2019
[10] Day, 2016

there is evidence that there has been a Treasury or Downing Street cat since the reign of Henry VII.[11] But perhaps the earliest record of any kind of media or public following for a government cat was Peter the Great, the then Home Office cat, from the 1940s to 1960s:

> "Peter became somewhat of a celebrity, appeared on the BBC in 1958 and pictured in newspapers and magazines, including October 1962's Woman's Realm. Peter's salary became part of the national conversation, with many animal lovers and cat charities incensed that the Home Office only paid him enough for what they considered starvation rations… Much of the Official Cats' file is made up of replies to well wishers and cat-lovers, and these letters give us an insight into Peter's daily life and character. In 1962 for instance we learn that while Peter doesn't have any 'cat-friends' his hobbies include 'pigeon fancying'. "[12]

Even when Peter the Great was put to sleep in 1964, National Archives records indicate that the Home Office received many letters of condolence from fans of Peter from Britain and globally, including one from the New York Transit authority's 'Etti-Cat' and a lot of letters from his fans in Italy.

Aims of the project

The aim of this project has been to develop an understanding of the government cat communities from an exploratory analysis of the government cat Twitter accounts. Given the public following and media coverage of past government cats, it is not surprising that the current occupants for the role of Chief Mouser in various government departments all have Twitter accounts. Tweeting,

[11] Davies, 1997; Van Vechten, 2000
[12] Day, 2016

retweeting, following and liking could be regarded as the social media equivalent of a relationship with fans, akin to fan correspondence in pre-internet times. Since Twitter was only launched in 2006, the current generation of government cats are also the first to have Twitter presences. As indicated in Figure 2 below, some of the government cats appear to have more than one Twitter account; there are five government cats but six Twitter accounts. Research into the phenomenon of internet cats has shown that people will look at cat content as a way of de-stressing or procrastinate, i.e. distract oneself from what they should be doing.[13] In a similar way, it has been commented that media coverage of the government cats are beneficial to the government.[14] I do not think, however, that the same can be said about the Twitter accounts. As indicated in Figure 2, all but one of the Twitter accounts are described in the profile as "unofficial" or "parody". A cursory examination of the content generated for each account indicated that a light-hearted approach was taken, mostly but not completely non-political and written, as if, from the feline point of view. Instead, an analysis of the government cat Twitter accounts would provide an opportunity to develop a contemporaneous profile of the followers of the government cats, without waiting for letters to be released by the government under the 30 years' rule.

The initial constitution of the data collection campaign

Figure 2 provides a tabular summary of the twitter data that is publicly accessible without the use of programming language scripts, i.e. what can be found out about each government cat's twitter page just by human observation. There are five government cats with a Twitter presence but six Twitter pages. There are three Twitter pages for the Foreign Office cat, Palmerston, two Twitter pages for the Treasury cat, Gladstone. The number of followers of each cat, the number of users that each cat is following (friends), the number of tweets posted by each "cat" and the start date of the Twitter

[13] Myrick, 2015, 171
[14] White, 2006

page are the only items of data that are objective, in that it is provided by the system. However, the number of followers, "following" and tweets are still approximate once they reach above a certain level. The remaining items of data – Twitter name, username, profile, related website and location - are subjective in that they are posted by the operator of each Twitter page. Only two Twitter pages have a related website, both of which appear to be fundraising pages for Battersea Dogs and Cats Home, where the cats in question originated. All locations specified suggest that the Twitter pages are operated by people based in the respective government departments in Central London. It is also worth noting that @Number10cat's Twitter page was set up in 2011, but the remainder were set up at some point during 2016. Whilst @Number10cat has the largest number of followers, posted the largest number of tweets and made the largest number of likes, the other Twitter pages indicates that *the relationship between the various objective attributes may not necessarily be linear with respect to time*.

The data available by human observation, as indicated in Figure 2, and the above observations, give rise to a number of options for descriptive and exploratory analysis:

1. The exact number of followers for each government cat;
2. The exact number of users that each government cat is following;
3. The exact number of tweets (or tweets and replies) posted by each government cat;
4. The exact number of likes made by each government cat;
5. Descriptive statistics for the options 1 – 4;
6. The relationship between the start date, the number of followers, following, friends, tweets and likes made;
7. The commonality of followers to governments cats, i.e. which users follow more than one government cats and which followers are the centre of a global network;
8. A textual analysis of the locations of the most recent followers of the government cats, with a view to identifying particular keywords;

9. A textual analysis of the profile descriptions of the most recent followers of the government cats, with a view to identifying keywords;
10. A textual analysis of tweets of the most recent tweets, with a view to identifying particular topics.

Data Flow

The data collection and analysis will primarily be undertaken using Python 2.7 and Twitter's Application Programming Interface (API). This phase was broken down into smaller tasks, as described in the next section. Each task was then broken down into a data collection sub-task and an analysis sub-task. The purpose of this approach was to simplify the testing of the code and to be able to test code without needing to interface with Twitter's system each time. By reducing the need to interface with Twitter's system, this minimised the number of times that Twitter's own imposed limits for data access were breached. As highlighted in Figure 1 below, a data collection program accessed and pulled data from Twitter, converted it into a Pandas dataframe or JSON or CSV format. An Analysis program then accessed the CSV or JSON file and performed various analyses before returning the output to screen or saving in a file.

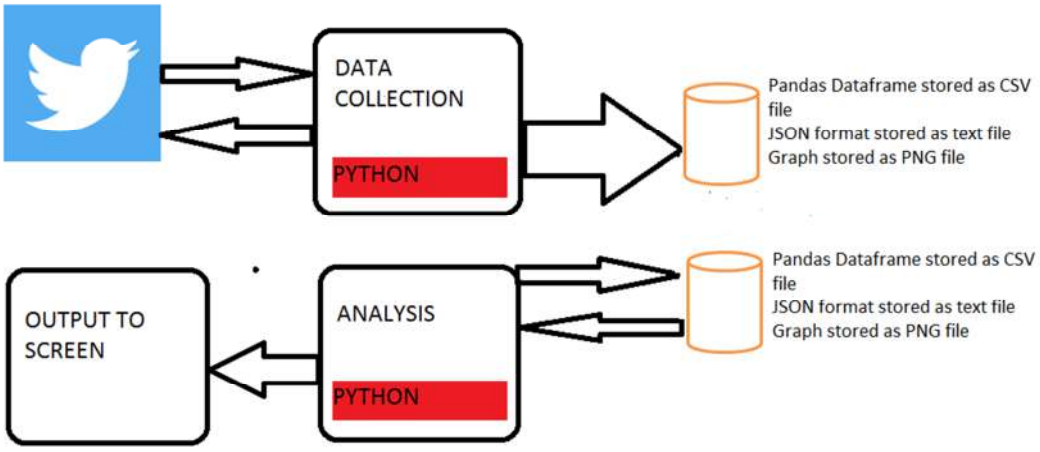

Figure 1: Data flow Diagram of the project

Summary of the Chapters

Chapter 2 focuses on the analysis of different Twitter libraries available in Python 2.7, as the programming approach is influenced primarily by the library used. It provides a justification for using Tweepy.

Chapter 3 examines the basic numerical data available about each government cat Twitter account, e.g. number of followers, friends, tweets and liked tweets and date of creation. This is important as the data that is manually available by inspection is often approximate due to large figures. Obtaining exact data made it possible to undertake an initial analysis to identify any linear relationship. In this case, it was found that there is no real linear relationship between any of the attributes.

Chapter 4 is concerned with an analysis of the UK entire government cat network. It produces a visualisation in graph form as well as numerical data about the number of nodes with certain degree, in-degree and out-degree. In this chapter, the most central users to the global government cat network are identified and information obtained about them.

Chapter 5 focuses on a textual analysis of the tweets of each government cat Twitter account, with a view to identifying regular n-grams used and thus key topics. The analysis also identifies the key users mentioned by each government cat Twitter account and, as a result, enables the visualisation of the relationship between each government cat as a graph. It was found that the high-frequency topics were unique to each government cat account, indicating that each account was operated by a different person.

Chapters 6 and 7 focuses on textual analyses of the profile descriptions and locations of the followers of each government cat. It used the same approach as in Chapter 5. The profile description and location attributes are subjective in that they are provided by Twitter users themselves. They are useful, however, as a way of finding out how users identify themselves rather than being assigned an identity by a third party who may not know them. With regards to location, in particular, it was found that users tended to self-identify by reference to towns, regions or an imaginary place rather than by country.

Cat	Government Department	Twitter name and username	Profile	URL	Location	Joining Date	Followers	Following	Tweets	Likes
Larry	10 Downing Street	Larry the Cat, Esq @Number10Cat	Yes	Yes	Downing Street, Westminster	February 2011	314K	131	16.8K	27.4
Palmerston	The Foreign Office	Palmerston @DiploMog	Yes	Yes	Whitehall	February 2016	90.2K	8	215	25
		Palmerston the Cat @PalmerstonFOCat	Yes	No	Foreign Office, London	April 2016	39.K	765	965	1,516
		Palmerston the FCO Cat @PalmerstonCat	Yes	No	Whitehall, London	April 2016	11K	30	938	2,580
Gladstone	The Treasury	Gladstone @TreasuryMog	Yes	No	Whitehall, London	October 2016	51.2K	21	18.7K	71K
		Gladstone @HMTreasuryCat	Yes	No	Westminster, London	July 2016	21.6K	4	127	142
Ossie and Evie	The Cabinet Office	Evie the Cat @HMCabinetCat	Yes	No	Whitehall	August 2016	41K	6	1,725	23.3K

Figure 2: Twitter pages by observation

2. Comparison of Twitter Libraries for Python

A non-standard library needs to be used to access data from Twitter using its Application Programming Interface (API) using Python. However, there are nine different libraries mentioned he Twitter Developer page.[15] There is thus plenty of choice in terms of Twitter library but no objective analysis of key criteria regarding suitability for particular projects. (There are anecdotal accounts and opinions posted by individual users on personal blogs and forums such as Stack Overflow, Reddit and Quora.) The first task, therefore, was to write a Python script that would measure the length of time to access certain attributes using Twitter's API and the volume of data (number of followers and friends of a target user) that could be accessed. The target user was Palmerston (@DiploMog), which is the official Twitter account of the Foreign and Commonwealth Office cat, Palmerston. @DiploMog has 87.4K followers and follows eight users.

It was originally intended that all nine libraries would be included in the test but this was not possible within the available time and technical capacity. The following libraries were considered:

- Tweepy;
- Twython;
- TweetPony;
- TwitterAPI.

Other common libraries used were:

- Time – to calculate the time taken to access data via the Twitter API;
- Pandas – to store data in a format that was suitable for easy analysis.

[15] 'Twitter Libraries', https://developer.twitter.com/en/docs/developer-utilities/twitter-libraries.html

Method

For each library, a different app (in a Twitter sense) was created on the Twitter Developer platform, thereby providing a different consumer API key, consumer API secret key, access token and access secret token. This was done so that data could be accessed using different libraries without breaching the rate limiting. Through a process of trial and error, it was discovered that the maximum number of followers and friends that could be accessed, without cursoring, was 200.

The code for the comparison task was divided between two Python files, *LibraryTest.py* for data collection and *AnalyseLibraries.py* for analysis, both of which are available in the appendix. The data collection and analysis aspects of the comparison task was divided between two files in order to ensure that the analysis code could be tested without having to interact with the API each time and hence without breaching the rate limit. The actual interaction with Twitter's API, using the various libraries, was handled by *LibraryTest.py*. An empty dataframe was created using pandas. For each library, once a list of followers and friends was captured and the time to capture measured, the data was added to the dataframe. It was also discovered through testing of the code that the time varied between instances of execution. It was decided, therefore, to use a form of bagging to ensure a degree of consistency and stability when it came to analysis of the output; data was captured from Twitter's API ten times for each library, with a view to later calculating the mean time and its standard deviation for each library. (The analysis of the data, including calculation of the mean and standard deviation, was undertaken in *AnalyseLibraries.py*.) Once the data was captured from Twitter and stored in a dataframe, the dataframe was exported to a CSV file, *compare.csv*, for subsequent analysis.

When it came to analyse, the CSV file produced by LibraryTest.py was re-read into a dataframe. The data relating to each Python library were added into new data frames and the mean and standard

deviation for the number of followers, number of friends and the time taken to access was calculated for each new dataframe and added to respective lists. Finally, the resultant lists were incorporated into two dataframes, one for the mean and one for the standard deviation, and the dataframes were plotted into bar chart format.

Findings

The mean time taken to access data using each Python library is shown in Figure 3 and Figure 4

Library	# Followers	# Friends	Time
Tweepy	200	8	2.0681
Twython	6	6	1.5750
TweetPony	200	8	2.2032
TwitterAPI	200	8	1.7580

Figure 3: The average volume of data available and time taken to access

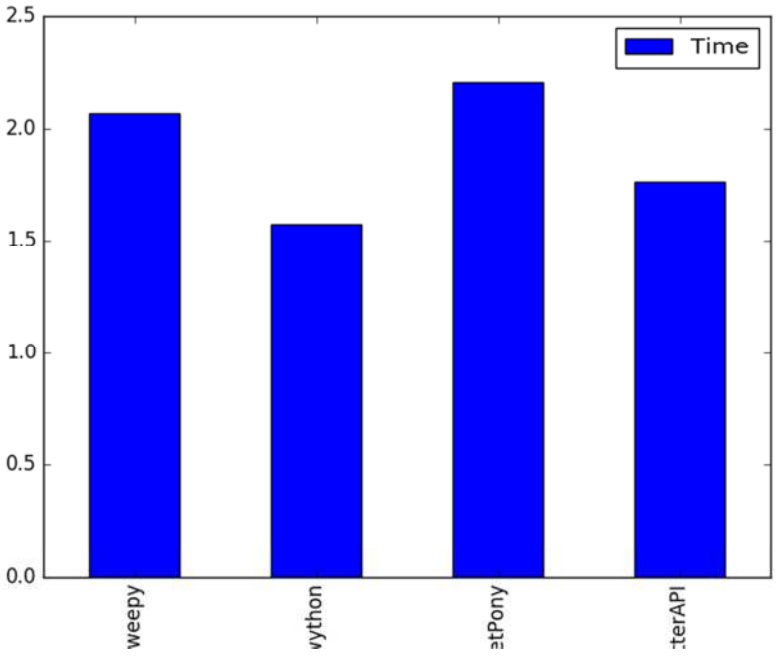

Figure 4: Time taken to access data

Two things immediately stood out. Firstly, using Twython, it was possible to obtain data for only six followers on average, compared to the maximum of 200 that have been obtained with the other libraries. Secondly, the average time to obtain data using Tweepy was longer than TwitterAPI, by 0.3 seconds, and is shorter than TweetPony by 0.8 seconds. Twython took the shortest time, but given that it produced a much smaller dataset, this was perhaps not surprising and not particularly useful. Based on the mean time alone and resulting size of dataset, it appeared that TwitterAPI is the most suitable library to use.

Figure 5: Standard Deviation for volume of data and time taken

Library	# Followers	# Friends	Time
Tweepy	0	0	0.431374
Twython	0	0	0.16815
TweetPony	0	0	0.144580
TwitterAPI	0	0	0.145630

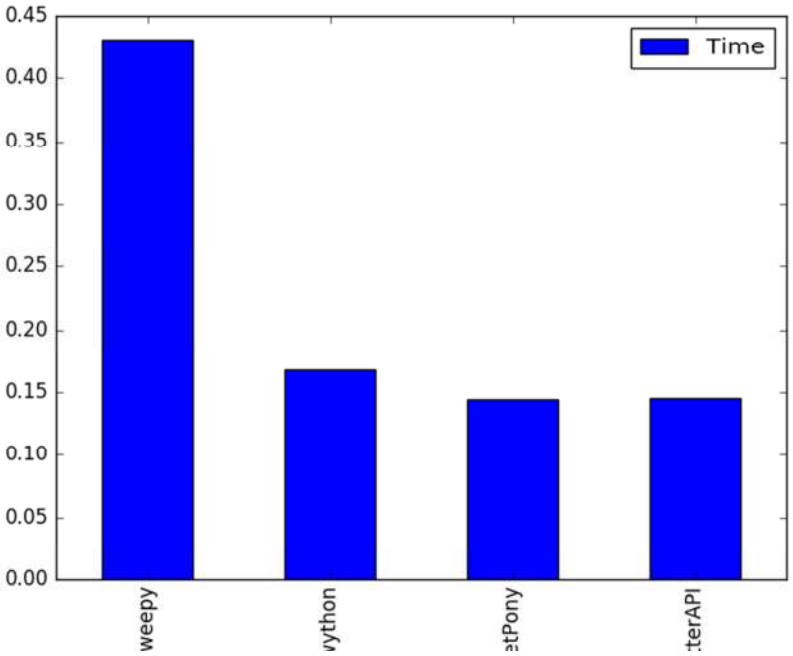

Figure 6: Standard deviation of the time taken to access Twitter

In Figure 5 and Figure 6, it could be seen that the standard deviation for the numbers of followers and number of friends is zero, indicating that the respective mean in Figure 3 was also the actual number of followers and friends obtained in each iteration and not an approximation. With regards to time taken, Tweepy appeared to have three times as greater variability as the other libraries. In this respect, it could be argued that TwitterAPI would be the most effective library in terms of time taken, consistency of time taken over different executions and size of resulting dataset.

However, from an analysis of the code, it could be seen that, with Tweepy or TweetPony, obtaining the followers and friends data from Twitter API can be accomplished in two lines. Using TwitterAPI, on the other hand, involved extra steps before the data can be stored in a dataframe. In this respect, TwitterAPI was more complex to use. Given that Tweepy took less time than TweetPony and was easier to use than TwitterAPI, it could be argued that there was no real benefit in using a library other than Tweepy.

3. Basic Analysis of Government Cats' Twitter pages

In Chapter 1, it was indicated that data such as the number of followers, the number of users the government cat is following, and the number of tweets and likes made by the cat was available from human inspection. However, the volume of data meant that only approximate numbers could be provided this way. In addition, it was observed that the vast majority of the government cats' Twitter pages were created at various points in 2016, so it was not clear whether there was a clear, linear relationship between the aforementioned attributes. The fact that the Twitter Page for @Number10cat was set up in 2011 and had posted the most number of tweets indicated that a linear relationship between the two. However, this was did not appear to the case for the other Twitter pages nor was there any obvious relationship between the duration of the page and the other attributes. As argued in Chapter 1, therefore, this section focused on the following exact data (as opposed to the approximate values made available on the respective profiles):

1. The number of followers for each government cat;
2. The number of users that each government cat is following;
3. The number of tweets (or tweets and replies) posted by each government cat;
4. The number of likes made by each government cat;
5. The time and date of creation of the twitter account;
6. Descriptive statistics for the options 1 – 5;
7. The relationship between the start date, the number of followers, following, friends, tweets and likes made.

The code is stored in

DataCollect_Basic.py and can be seen in the appendix.

Findings

The exact number of followers, friends, tweets and likes made and the exact date of joining Twitter are shown in Table 4 (stored from a Pandas dataframe into a CSV file and copied from Excel).

Gov cat	Followers	Friends	Joined	Likes	Tweets
DiploMog	90574	8	17/02/2016 14:56	31	220
HMCabinetCat	41108	6	04/08/2016 07:06	23302	1725
Number10cat	315770	131	15/02/2011 11:00	27521	16786
PalmerstonFOCat	39792	764	12/04/2016 12:30	1515	965
PalmerstonCat	11006	30	12/04/2016 22:00	2594	941
TreasuryMog	51577	21	14/10/2016 13:47	71159	18756
HMTreasuryCat	21665	4	29/07/2016 09:26	142	127

Figure 7: Summary of Government Cats' Twitter data

From a cursory examination of the above data, and as mentioned in Chapter 1, the Downing Street cat's Twitter account, @Number10cat, has been live for the longest and has the highest number of followers. The remainder of the government cats' Twitter accounts were set up in 2016. This would indicate that the number of followers was directly related to the length of duration of the Twitter account. However, it could also be argued that the @Number10cat account could be seen as an outlier – in the summer of 2019, it had over 300,000 followers, compared to the other government accounts, which each have less than 100,000 followers. Furthermore, if one only looks at the government cat accounts set up in 2016, it is difficult to argue that there is linear relationship between how long an account has been live and the number of followers. The official Foreign Office cat account, @DiploMog, was set up in February 2016 and has almost 91,000 followers, but the second largest number of followers was obtained by @TreasuryMog, one of the unofficial HM Treasury cat account, which was also the last one to be created (in October 2016). In a sense, even within the space of a year, there are still blatant outliers. The inability to say whether there is a linear relationship between the date of creation of an account in seconds (a proxy for duration of an account) and the number of the followers can be seen in Figure 8.

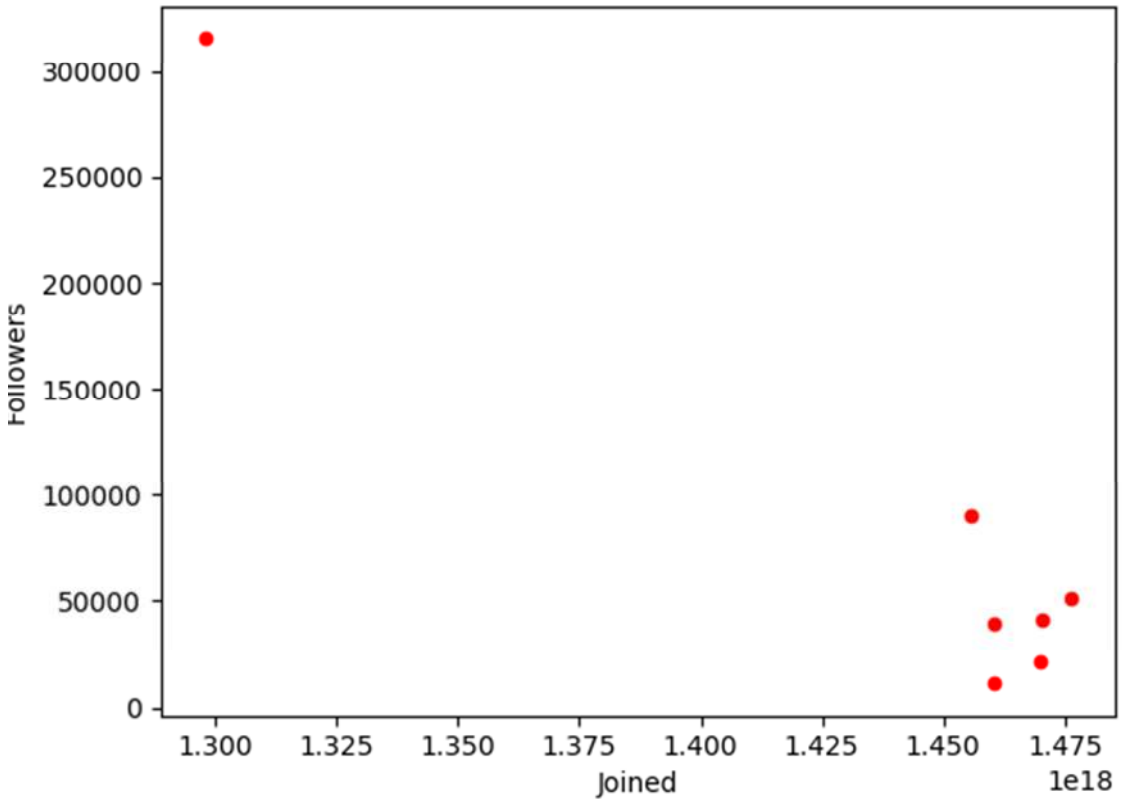

Figure 8: The relationship between the number of followers and the duration of the account in seconds

There also appears to be different level of engagements, which is not necessarily related to the duration of the account. The unofficial account for the Foreign Office cat, @PalmerstonFOCat, has the third largest number of followers, at around 39,000, and was set up in April 2016, but the highest number of friends, i.e. it was the government cat account that was following the highest number of account. At the same time, there were other government cat accounts that posted and liked more tweets in less time. This can be seen Figure 9, Figure 10 and Figure 11.

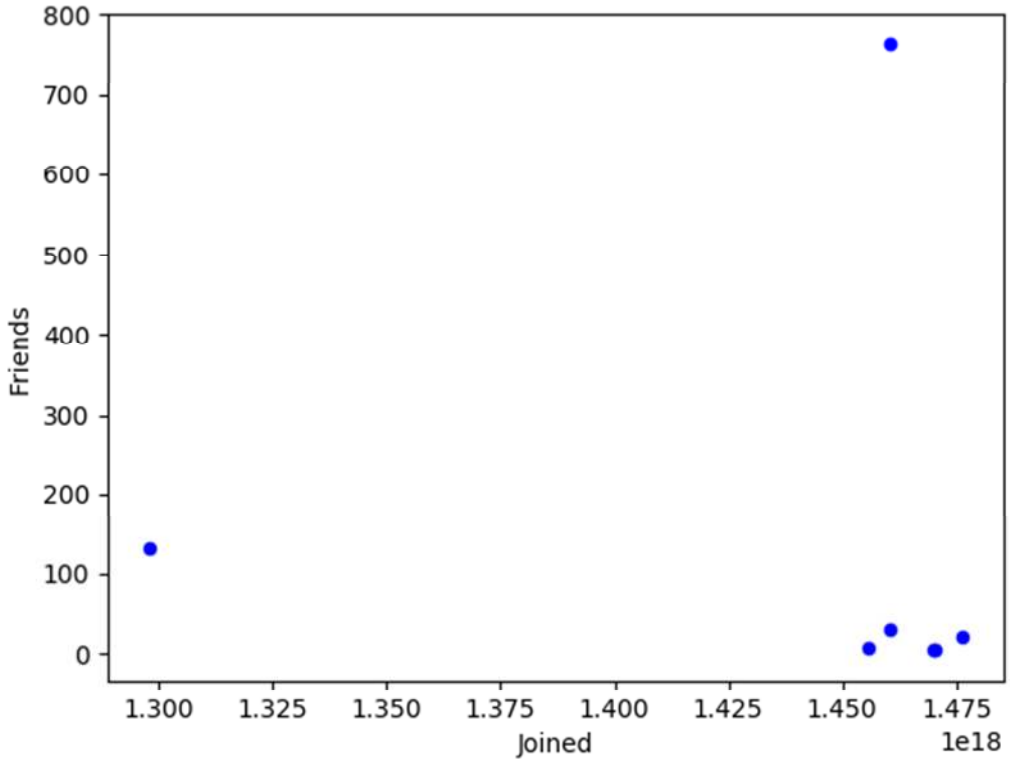

Figure 9: Relationship between duration of Twitter account and number of friends

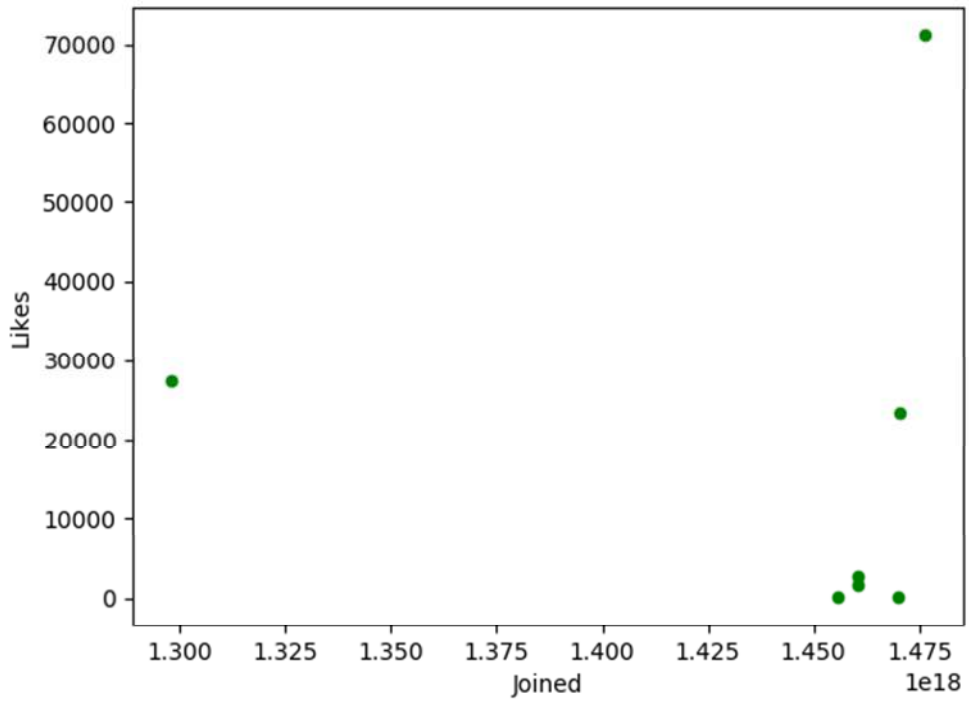

Figure 10: Relationship between duration of Twitter account and number of tweets liked

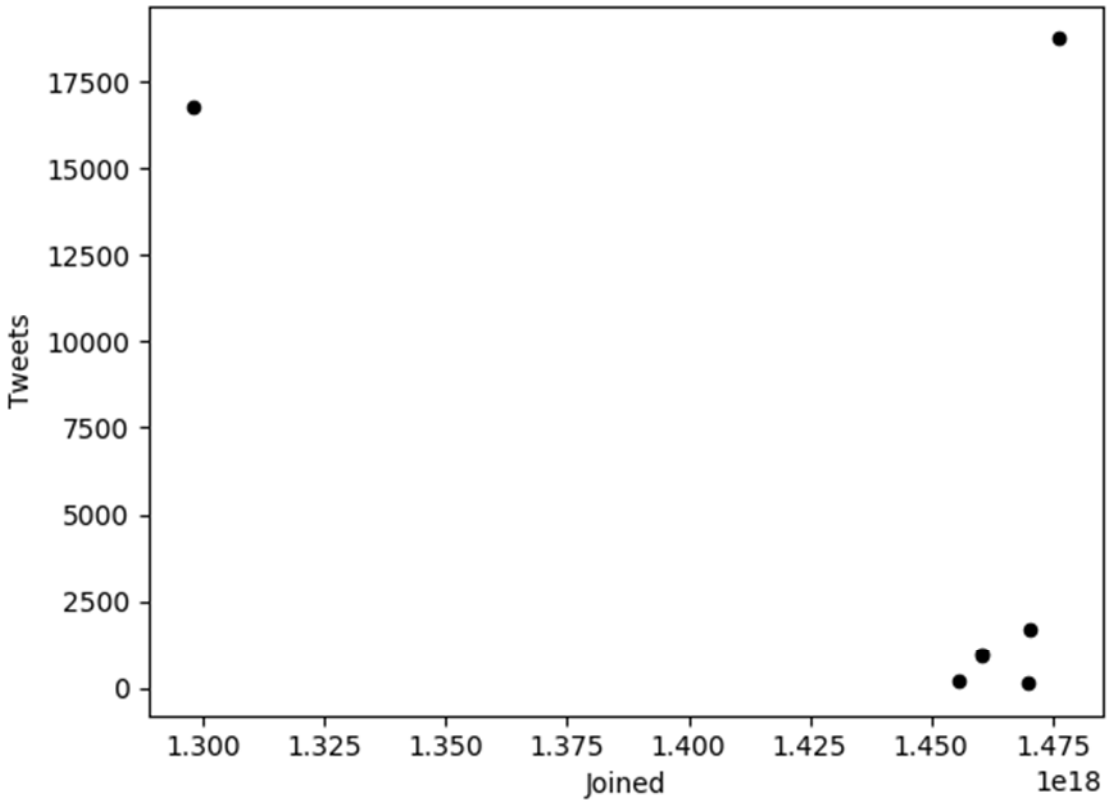

Figure 11: Relationship between duration of account and number of tweets posted

It is not particularly surprising, however, that duration of the government cats' accounts are not necessarily related to the level of engagement. The conventional wisdom, as indicated in Google searches of the phrase "increasing followers on Twitter" is to tweet often, although this alone is not enough. Whilst the longer a Twitter account has been live, the more potential there would be to increase followers, this would generally only be the case if there is content worth following. The data in Figure 7 and the chart in Figure 12 showed that there is no linear relationship between the duration of the account and the number of tweets posted.

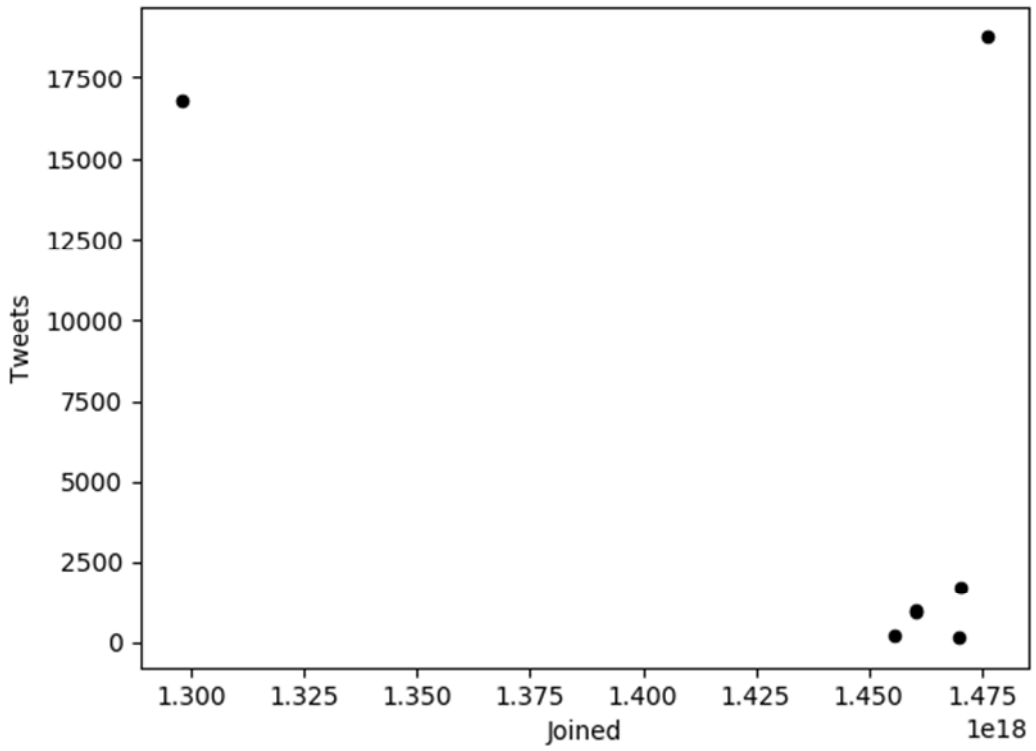

Figure 12: The relationship between duration of Twitter account and number of Tweets

However, when comparing the different Twitter account, the conventional wisdom regarding regular tweeting does not hold. For example, the longest government cat account, @Number10cat, has the largest number of followers but has not posted or liked the most number of tweets. @DiploMog has the second largest number of followers but has posted and liked one of the lowest number of tweets. The relationship (or lack thereof) between the number of tweets and the number of followers, liked tweets and friends can be seen in Figure 7 and Figure 13, Figure 14 and Figure 15.

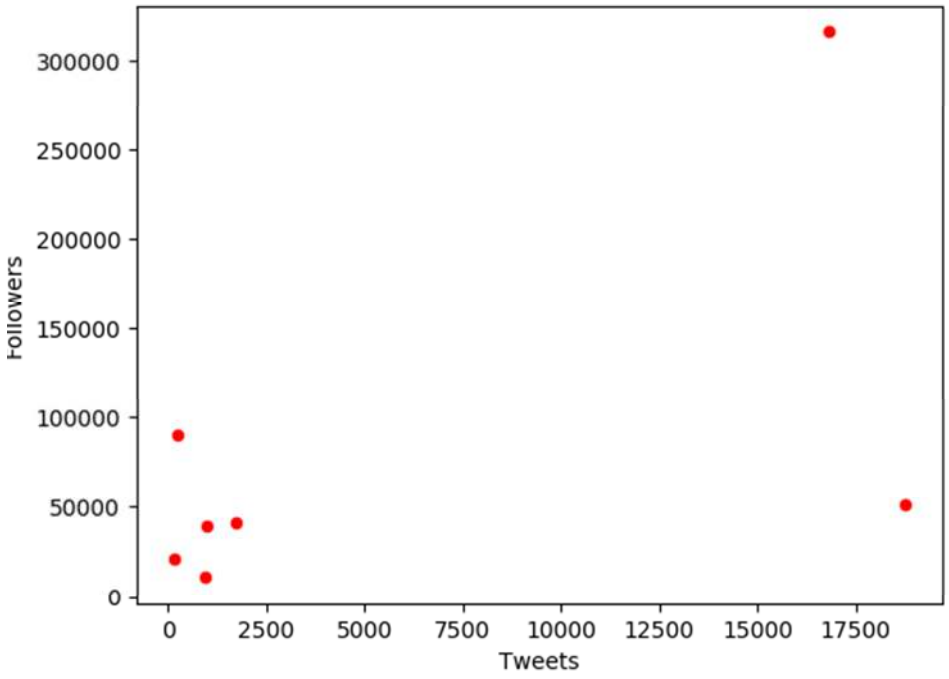

Figure 13: Relationship between the number of Tweets posted and the number of Followers

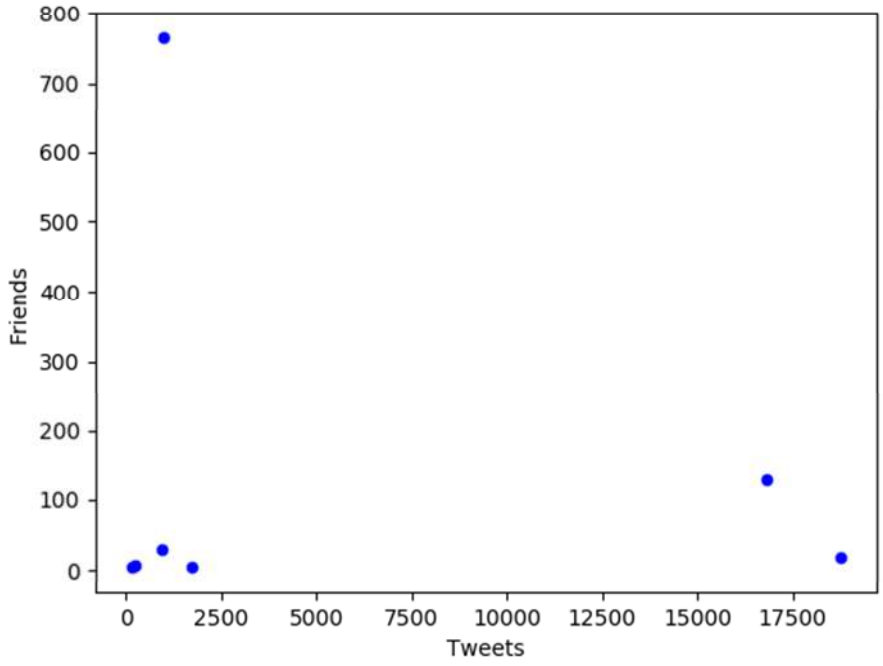

Figure 14: Relationship between the number of Tweets posted and the number of Friends

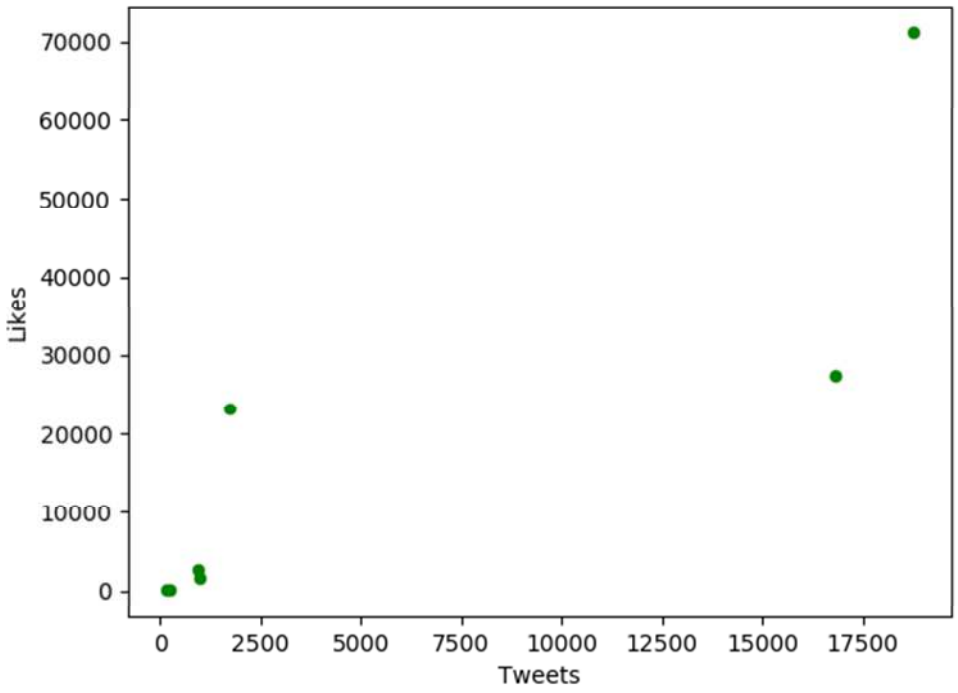

Figure 15: Relationship between number of Tweets posted and number of Tweets liked

The lack of a relationship between the number of tweets by government cat accounts and the number of followers or friends contradicts prior research undertaken by Beevolve, involving 36 million Twitter user profiles.[16]. It is possible that the difference could be down to the small size of the dataset used in this chapter. In Chapters 4 and 5, it became clear that the Twitter accounts of the UK government cats are literally a handful of similar types of accounts for cats or other animals representing public sector organisations. Nevertheless, the inability to visualise a linear relationship between the duration of a Twitter account or the number of tweets posted and the number of followers, friends and tweets liked can be seen in measures of central tendency and dispersion of the number of followers, friends, tweets posted and tweets liked shown in Figure 16. The visibly significant difference between the mean and the median, as well as the size of the standard deviation, for all attributes is indicative of the presence of both relatively very large and very small amounts.

[16] Beevolve, 2012

Attribute	Mean	Median	St. Dev
Followers	81736.14286	41125	106423.098
Friends	137.4285714	19	279.889179
Tweets	5648.428571	965	8323.72985
Likes	18047.28571	2657	26126.3573

Figure 16: Measures of central tendency and dispersion

Since it is not possible to argue that there is a linear relationship between the creation date of the account (as a proxy for duration) or the number of tweets and the other attributes, the question is whether there is a linear relationship between the number of followers, the number of friends and the number of likes.

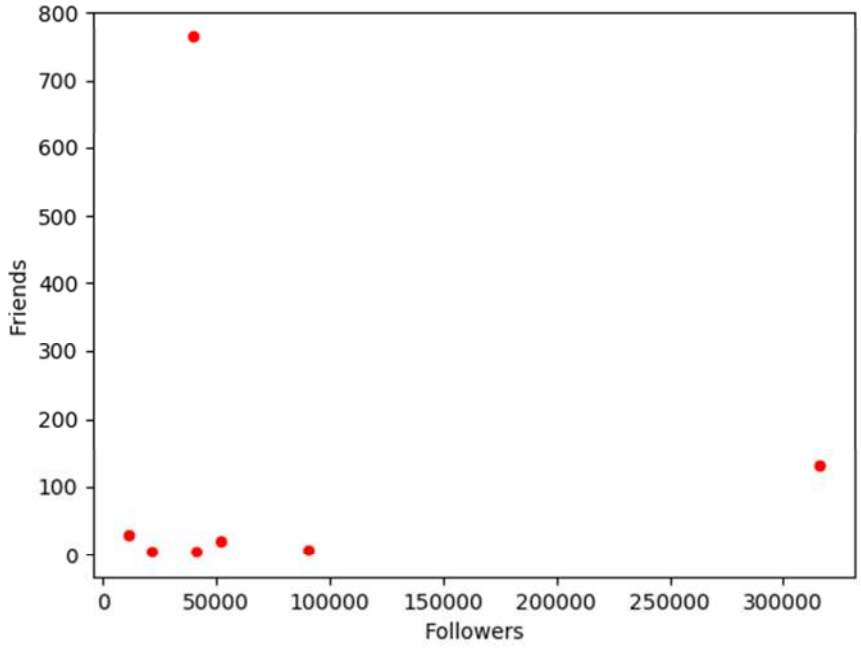

Figure 17: Relationsip between the number of Followers and Friends

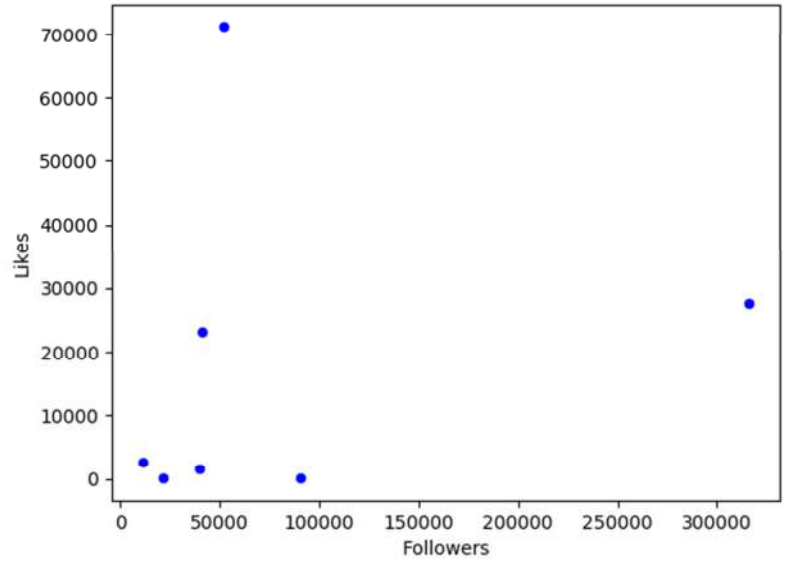

Figure 18: Relationship between number of Followers and Likes

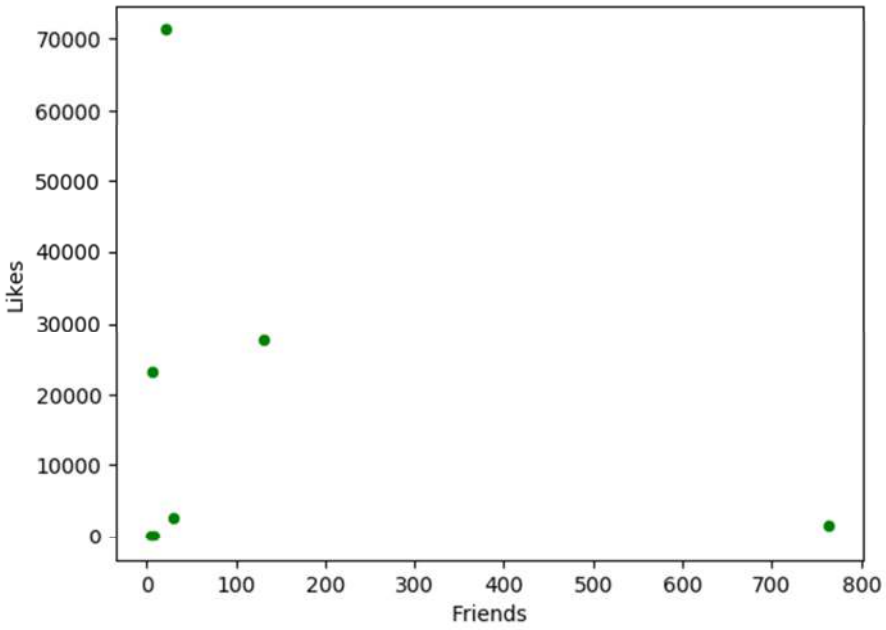

Figure 19: Relationship between number of Friends and Likes

From Figure 17and Figure 18, there does not appear to be any linear relationship between the number of followers and either the number of friends or liked tweets. From Figure 19, there does not appear to be any linear relationship between the number of friends and the number of liked tweets.

4. Network analysis of Government Cats' Followers and Friends

This chapter visualises the relationship between a Twitter user and their followers and friends as form of network. In the first instance, such connections could be viewed as a trade network in the sense of Calderelli and Chessa.[17] The product that is being traded (produced and consumed) is information – more precisely, tweets. In the context of Twitter, followers are consumers of tweets in that they are receiving the information being produced by those that they are following. As a result, those that are being followed are producers to the extent that they are actively tweeting. In the context of this project, the government cat Twitter accounts are, in the first instance, producers of information but they are also consumers in that they are following others and consuming information. The problematic of portraying Twitter relationships as producer-consumer is that it is difficult to measure what it means to consume a tweet. To tweet is an act of production but for that tweet to be consumed, a user would have to first see it in their timeline - this is not always possible due to the structure of Twitter and volumes involved. Another way proposed by Calderelli and Chessa of thinking about a network on Twitter is as a quasi-food web, which is used to show the predator-prey relationships between different species in the same environment or habitat.[18] In this regard, the government cat network is an environment within a larger Twitter environment. The follower - user relationship could be viewed as a predatory relationship in that one has to actively click "follow", in the same way that a predator has to actively hunt or is drawn to its prey. In this respect, the friend of a user could be regarded as its prey. The predator-prey metaphor is imprecise but it offers a good way of viewing the direction of relationships on Twitter.

According to the data obtained on the total number of followers and friends, as shown in Figure 7, there are 572,463 users in the whole government cat Twitter network. This is the sum of the number of

[17] Calderelli and Chessa, 2016, 26-44
[18] Ibid, 2016, 4-25

followers and friends of each government cat plus the six government cat accounts. However, as indicated in Chapter 2, it is not possible to obtain data on all the followers and friends using Twitter's API. The maximum number of friends and/or followers that can be obtained is 200 – that will be the 200 most recent followers and friends.

Assumptions

The following initial assumptions were made:

- Each government cat was at the centre of its own network of followers, so the in-degree of each cat was the same as the number of followers and the out-degree of each follower was 1;
- Each government cat was following other users, so the out-degree of each cat was the same as the number of friends and the in-degree of each friend of a cat was 1.

The initial assumptions were then amended to take into account the following possibilities:

- Some users were following more than one cat;
- Cats were followers of other cats;
- Some of a cat's friends were following the cat and/or other cats

As the focus of this project was on the government cat community, the followers and friends of the followers of the government cats were disregarded save for those that were already followers or friends of a cat. Any Twitter user that is neither a follower nor a friend of a government cat was not viewed as being a member of the government cat community.

The code to produce details about the government cat network was stored in

AnalyseFollowers.py and the code to produce information about the most important users in the network was stored in *AnalyseFollowers2.py*.

Findings

As indicated above, the followers and friends of a government cat make up a local network. In the global network of the seven UK government cats and their 200 most recent followers and friends, there are 1,520 nodes, representing Twitter users, connected by 1,798 edges. In other words, the order of the government cat network is 1,520 and the size of the network is 1,798. That is less than one 1% of the total population size.

A breakdown of the different degrees of each node can be seen in Figure 20 below. As expected, there are no nodes with zero degree, indicating that every node is either a follower or a friend of at least one other user in the network, in keeping with initial assumptions. However, 1,138 nodes have zero in-degree, indicating that no-one is following them. Similarly, 374 nodes have zero out-degree, indicating that they are not following anyone else within the government cat network.

Degree	Degree nodes	In-Degree nodes	Out-Degree nodes
0	0	1138	374
1	1321	361	961
2	125	7	118
3	50	5	46
4	10	2	8
5	5	0	5
6	2	0	3
7	0	0	0
8	0	0	1
19	0	0	1
30	0	0	1

Figure 20: Degree of Nodes in Government Cat Network

The nodes with an in-degree of zero, i.e. no followers within the government cat network, are likely to be on the outside of the network. On the graph in Figure 21, these will probably be the nodes on the outside with directed edges pointing away from them. Similarly, the nodes with an out-degree of zero are likely to be on outer rim, but with directed edges pointing towards them. I would also posit that the 361 nodes with an in-degree of 1 and the 961 nodes with an out-degree of 1 are mostly also on the outer rim.

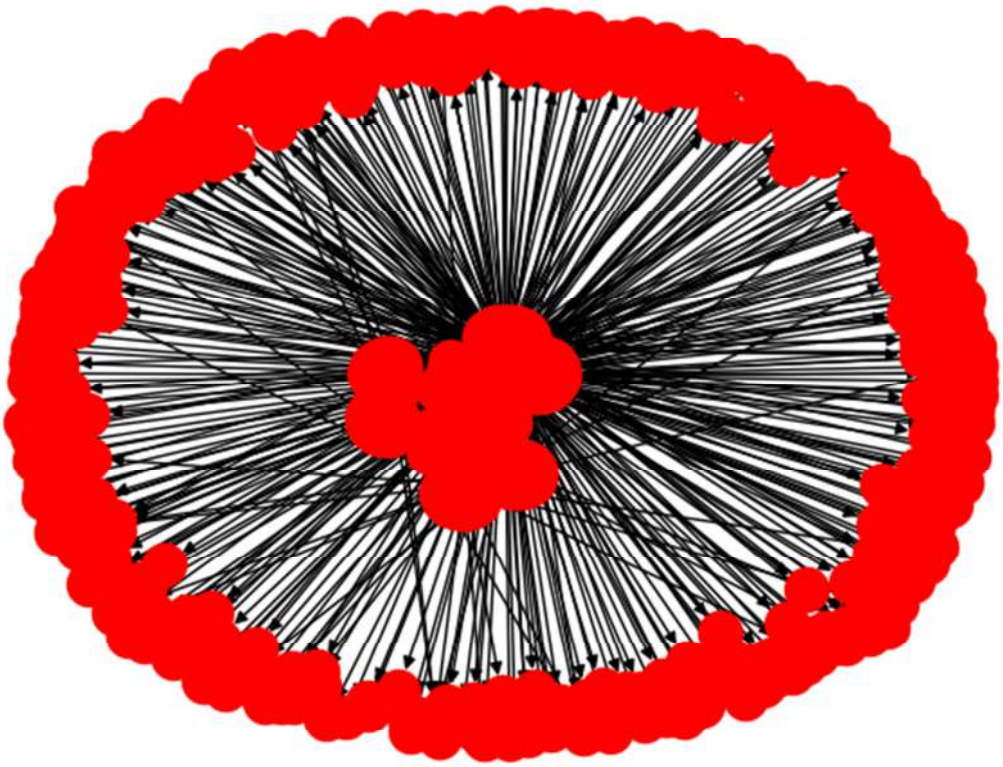

Figure 21: Graph of Government Cat Networkage

However, it is the cluster of nodes in the centre of the graph in Figure 21 that are most relevance, as these indicate intermediate nodes between those on the outer rim. According to degree centrality, the higher the degree of a node, the more important it will be the global network. In the context of a Twitter, it could be argued that those with the largest number of followers (in-degree) and those who are following the largest number of users (out-degree) who will have the greater effect in terms of

information transmission. That does not necessarily mean, however, that those with the largest number of followers will also be following the largest number of users.

Whilst it can be seen from Figure 21 that a relative handful of nodes are more important than the others, it is not clear which nodes they are. However, when examining Figure 22, it can be seen that there is some kind of pattern. Firstly, regardless of whether degree, in-degree or out-degree is being considered, the higher the degree, the lower the number of nodes. Secondly, it is clear that, whether one is looking at from the perspective of degree, in-degree or out-degree, there are between 14 and important nodes, split up in a similar pattern.

Degree	**In-Degree**	**Out-Degree**
Degree of 4: 10 nodes	Degree of 2: 7 nodes	Degree of 4: 8 nodes
Degree of 5: 5 nodes	Degree of 3: 5 nodes	Degree of 5: 5 nodes
Degree of 6: 2 nodes	Degree of 4: 2 nodes	Degree of 6: 3 nodes
		Degree of 8: 1 node
		Degree of 19: 1 node
		Degree of 30: 1 node

Figure 22: Number of most important nodes

When it comes to the degree in general, users with four, five or six followers and friends are important for transmitting information within the government cat network. But when looking in-degree alone, that is the number of followers, users with two, three or four followers are important. For out-degree, that is the number of friends, users with four, five and six friends are important, as well a one user with eight friends, one user with 19 friends and one user with 30 friends. The most important members of the government cat network are listed in Figure 23.

Importance by Degree	Importance by In-Degree	Importance by Out-Degree
Number10cat	Number10cat	CBond78669266
CBond78669266	cabinetofficeuk	amrmt2
amrmt2	Battersea_	rashaan_kashif
rashaan_kashif	BrightonStnCat	Cardi91883944
Cardi91883944	TreasuryMog	@PalmerstonCat
ChynLawrich	CatsProtection	ChynLawrich
danielles_salt	Number10Mouse	@HMTreasuryCat
KingoftheCatzle	HMCabinetCat	danielles_salt
HMCabinetCat	justin_ng	KingoftheCatzle
Shillam	SMcDonaldFCO	Shillam
pictureworks	10DowningStreet	pictureworks
zylanthic	DiploMog	zylanthic
ollycassie	foreignoffice	ollycassie
FlyWestminster	LawrenceDipCat	@TreasuryMog
FluffyDave		FlyWestminster
kwittman		FluffyDave
LawrenceDipCat		@DiploMog
		kwittman
		@HMCabinetCat

Figure 23: Most important members in government cat network

Of the government cats, Number10cat, HMCabinetCat, TreasuryMog and DiploMog are the most important by in-degree. This is to be expected, as they are the government cats with the most followers. Other important users by in-degree whose username is recognisable or notable include:

- Cabinetofficeuk - The Cabinet Office, the respective government department for HMCabinet Cat;
- Battersea_ - Battersea Cats and Dogs Home, where Larry the Cat and others cats were obtained;
- CatsProtection – Cat's Protection;
- 10DowningStreet – the respective government department for Number10cat;
- Foreignoffice – the respective government department for DiploMog
- SMcDonaldFCO – a senior civil servant within the Foreign Office

- BrightonStnCat – the cat for Brighton Station
- Number10Mouse – an unofficial Twitter account for a mouse at Downing Street
- LawrenceDipCat – the cat for the British Embassy in Amman

With regards to out-degree, PalmerstonCat, HMTreasuryCat, TreasuryMog, DiploMog and HMCabinetCat are the most important government cats. PalmerstonCat has the most number of friends but there is no obvious reason why the others are of importance and Number10cat is not. Other notable inclusions are FlyWestminster, an unofficial account for a fly at Westminster. With regards to degree in general, Number10cat and HMCabinetCat are the most important government cats and other notable users are LawrenceDipCat and FlyWestminster.

The problematic with identifying the most important accounts is that the answer differs depending on what version of degree is used. It can be seen from Figure 20 that the number of nodes of a degree N in the undirected network is the sum of the number of nodes of degree N in the directed network (in-degree plus out-degree). It could be posited that the important users as identified by the degree of nodes in the undirected network is useful for a further analysis of the important users. However, this would only provide information in terms of connections but not in terms of information flow. I would therefore argue that, as seen in Figure 24 below, a union (Boolean “or”) of the lists for important nodes by in-degree and out-degree, as shown below, would provide a true reflection of central users within the government cat network. By definition, the union would be the users who are either have a high in-degree or a high out-degree or both.

Union of lists of nodes by in-degree and out-degree
CBond78669266
amrmt2
rashaan_kashif
Cardi91883944
@PalmerstonCat
ChynLawrich
@HMTreasuryCat
danielles_salt
KingoftheCatzle
Shillam
pictureworks
zylanthic
ollycassie
@TreasuryMog
FlyWestminster
FluffyDave
@DiploMog
kwittman
@HMCabinetCat

Figure 24: Union of in-degree and out-degree nodes

Using the Twitter API, more details have been provided about the most important members of the government cats network at the end of this chapter. The correlation between the different attributes of the important users in the network were calculated and shown in Figure 25. As in Chapter 3, the attributes are number of tweets, number of followers, number of friends, number of liked tweets, creation date of account (in seconds), represented as floating point numbers, and the presence of a location and description, represented as '1' for 'present' and '0' for 'not present'.

	Followers	Friends	Tweets	Likes	Created	DescriptionYe	LocationYes
Followers	1	-0.232112431	0.248933099	0.37786356	0.019989791	0.210429705	0.290233185
Friends	-0.232112431	1	0.544966422	0.314131394	-0.496571197	0.168747354	0.086019511
Tweets	0.248933099	0.544966422	1	0.937809939	-0.364853384	0.18059416	0.247553657
Likes	0.37786356	0.314131394	0.937809939	1	-0.199332633	0.132963357	0.257835861
Created	0.019989791	-0.496571197	-0.364853384	-0.199332633	1	-0.255513284	-0.334074491
DescriptionYes	0.210429705	0.168747354	0.18059416	0.132963357	-0.255513284	1	0.396787697
LocationYes	0.290233185	0.086019511	0.247553657	0.257835861	-0.334074491	0.396787697	1

Figure 25: Correlations between attributes of important users

From Figure 25, it can be seen that there is a very weak or weak correlation, defined as less than 0.5, between most of the attributes. Even for the key members of a network, there is no clear link between the number of tweets and the number of followers. This is in keeping with the findings reported in Chapter 3. There is also no clear link between the inclusion or exclusion of a description or location and the number of followers or friends. This would seem to contradict the advice often provided by blogs that a description helps to attract followers. There is a very strong, positive correlation between the number of tweets and number of liked tweets. Intuitively this makes sense, if one considers a tweet and a like as two forms of engagement; the more one tweets, the more likely they are engaging with other people's tweets. There does appear to be an average positive correlation between the number of tweets and the number of friends and an average negative correlation between the number of tweets and time the account was created (as a proxy for duration of account). Since the correlation between tweets and followers is quite weak, it could be argued the average correlation between tweets and friends is because, by following more users, there is more scope for tweets to engage with. The correlation between tweets and time of creation is negative because the smaller the time is, the longer the account has been in existence and so the greater the scope to send more tweets. But since the correlation is only average, it could be posited that some of the important users in the government cat network could be fake account. For the purpose of this project, the term "fake account" is defined to mean one that may not necessarily be operated by a real person. The main, measurable characteristics of a fake account are listed below, but it does not mean that an account that matches these criteria is necessarily fake:

- High follower to tweet ratio (large number of followers, small number of tweets);
- High friend to follower ratio (large number of friends, small number of followers);
- The absence of a description;
- Very high tweet to duration ratio (large average number of tweets per day).

Screen name	Follower to Tweet	Friend to Follower	Description	Tweet to Duration
CBond78669266	inf	8.5	0	0
amrmt2	1	19.32	1	1.38E-12
rashaan_kashif	0.594594595	35.97727273	1	4.38E-12
Cardi91883944	8	6.75	1	5.52E-14
PalmerstonCat	11.53934942	0.002728017	1	5.64E-11
ChynLawrich	1.266666667	14.10526316	1	8.50E-13
HMTreasuryCat	171.2677165	0.0001839	1	7.47E-12
danielles_salt	1	11.5	0	1.15E-13
KingoftheCatzle	3.925	1.785031847	1	8.84E-12
Shillam	0.081218274	3.119318182	1	1.50E-10
pictureworks	0.948849105	1.787061995	1	2.70E-11
zylanthic	inf	12.33333333	1	0
ollycassie	0.16070008	1.504950495	1	8.55E-11
TreasuryMog	2.801499123	0.000360538	1	1.10E-09
FlyWestminster	0.181818182	15	1	2.43E-12
FluffyDave	2.571428571	19.5	0	4.00E-13
DiploMog	409.1666667	8.81E-05	1	1.32E-11
kwittman	0.059576934	4.517509728	1	1.21E-09
HMCabinetCat	23.90144928	0.000145525	1	1.01E-10

Figure 26: Potential characteristics of fake accounts

Figure 26 shows how each of the important followers satisfy the criteria for fake accounts. The larger the follower-tweet ratio or the friend-follower ratio, the more likely an account is to show characteristics of a fake account. A description value of 0 indicates a higher likelihood that an account is fake. Finally, the larger the tweet-duration ratio, the more likely an account is to show characteristics of a fake account. Two users classified as important stand out immediately: @CBond78669266 and @zylanthic. Neither of them appeared to have tweeted at all, as indicated by a zero tweet-to-duration ratio and an infinite follower-to-tweet ratio. Both of these accounts were created in August 2019 and thus were less than one month old at the time of writing. They have one and three followers respectively, while following 17 and 37 users respectively. @CBond78669266 does not have a profile description. This suggesed that both these accounts could be fake. However, @zylanthic does have a profile description – self-identifying as a musician and raconteur – and profile picture, which suggested that the account is fake. I would also point out that two of the government cat accounts, @HMTreasuryCat and @DiploMog, have relatively a higher follower-tweet ratio. This raises questions whether the conventional wisdom for judging whether an account is particularly accurate.

Data obtained about central users in government cat network

This section provides details of the important users in the government cat network, as obtained from Twitter's API using the code in *AnalyseFollowers2.py*. They have been categorised according to stated location.

Location not known

Screenname: CBond78669266
Followers: 1
Friends: 17
Tweets: 0
Likes: 0
Created: 2019-08-12 11:31:16
Location:

Screenname: Cardi91883944
Big beard fat
Followers: 7
Friends: 52
Tweets: 1
Likes: 9
Created: 2019-07-27 13:52:32
Location:

Screenname: ChynLawrich
Gonna to make myself great and write down my life's history
Followers: 19
Friends: 265
Tweets: 15
Likes: 65
Created: 2018-04-18 07:24:02
Location:

Screenname: danielles_salt
Followers: 2
Friends: 29
Tweets: 2
Likes: 6
Created: 2017-09-03 04:23:19
Location:

Screenname: kwittman
Sustainability Ed, Historian, Comm Coll pro, MGCCC, USM, WKU grad. NIU Doc Student. Cubs Fan, Knitter, Liberal, Adoptee, Mississippian. □ RT≠ E. She/her.
Followers: 1016
Friends: 4625

Tweets: 17208
Likes: 40129
Created: 2008-12-06 19:22:53
Location: Southeast of Disorder

Japan

Screenname: amrmt2
研究用アカウント作りました。発言は所属する機関とは一切関係ありません。中近世ヨーロッパ/金工/ジュエリー/Amerbach Kabinett/Wenzel Jamnitzer/Wunderkammer//スイスc・ドイツDE羊肉が好き・
Followers: 16
Friends: 451
Tweets: 12
Likes: 21
Created: 2019-08-13 11:25:47
Location: 南の島に行きたい[south island of Japan]

Pakistan

Screenname: rashaan_kashif
Pakistani American PKUS
Doctor of Medicine
Followers: 44
Friends: 1583
Tweets: 72
Likes: 308
Created: 2016-04-09 05:25:54
Location:

United Kingdom

Screenname: @PalmerstonCat
I'm the top cat of Whitehall..prowling the corridors of power and catching mice for the Diplomatic Service. A long way from Battersea Dog and Cats Home!(parody)
Followers: 10996
Friends: 30
Tweets: 944
Likes: 2661
Created: 2016-04-12 22:00:08
Location: Whitehall, London

Screenname: @HMTreasuryCat
Chief Mouser to the Treasury and new cat on the block. Uncensored and unofficial. Photo credit: Laura Gallant, BuzzFeed.
Followers: 21718
Friends: 4
Tweets: 127

Likes: 142
Created: 2016-07-29 09:26:54
Location: Westminster, London

Screenname: Shillam
Yoga teacher.
Followers: 176
Friends: 549
Tweets: 2143
Likes: 2022
Created: 2009-06-13 07:59:22
Location: www.goodtimesyoga.co.uk

Screenname: pictureworks
Professional Photographer Press/PR/Commercial/Weddings/Landscapes/events, anything as long as it stands still long enough for me to catch it.
Followers: 372
Friends: 663
Tweets: 388
Likes: 571
Created: 2009-08-17 17:35:58
Location: Shrewsbury, Shropshire

Screenname: ollycassie
#CrazyCatLady (https://t.co/lWVLoWnkSm) who abseiled down the Spinnaker Tower on 4/8/19 for @iowcats - https://t.co/OlXzV9zSA9…
Followers: 202
Friends: 303
Tweets: 1257
Likes: 382
Created: 2010-03-30 19:07:27
Location: Isle of Wight

Screenname: @TreasuryMog
I live and work at the Treasury as a mouser but I also have a paw in the finances. Here to help lighten up the political world. Unofficial.
Followers: 51940
Friends: 19
Tweets: 18781
Likes: 71234
Created: 2016-10-14 13:47:06
Location: Whitehall, London

Screenname: FlyWestminster
I'm the fly on all the wall, any wall, in Westminster.

Like all respectable flies, I'm attracted to BS. Fly views are wide-eyed, multiple, rounded, impartial
Followers: 3
Friends: 110
Tweets: 33

Likes: 7
Created: 2019-08-12 17:16:57
Location: Downing Street, London, SW1

Screenname: FluffyDave
Followers: 17
Friends: 349
Tweets: 7
Likes: 6920
Created: 2017-11-28 11:55:24
Location: United Kingdom

Screenname: @DiploMog
Official account of the @foreignoffice Chief Mouser and former resident of @Battersea_. Supporting them through https://t.co/aRd9jlOFAI
Followers: 90713
Friends: 8
Tweets: 220
Likes: 31
Created: 2016-02-17 14:56:00
Location: Whitehall

Screenname: @HMCabinetCat
The newest and only female cat in Government, the Cabinet Office Cat, based in Whitehall. Mother to Ossie. Named after Dame Evelyn Sharp. Parody. Unofficial.
Followers: 41174
Friends: 6
Tweets: 1725
Likes: 23283
Created: 2016-08-04 07:06:54
Location: Whitehall

United States

Screenname: KingoftheCatzle
Snowshoe/Siamese Rescue. Pawsome tweets to keep you feline happy. I'm purrty adorable, too. Loves litterary fiction and classical mewsic. Let's be furends!
Followers: 560
Friends: 1011
Tweets: 117
Likes: 546
Created: 2019-08-01 23:49:52
Location: NYC

Screenname: zylanthic
Raconteur, ENFP, Musician
Followers: 2
Friends: 35
Tweets: 0
Likes: 6

Created: 2019-08-10 22:33:33
Location: Excelsior, MN

5. Main topics of Government cat tweets

In Chapter 4, a visualisation of the global government cat network, based on the most recent 200 followers of each government cat, was produced. As can be seen from Figure 21, there were a handful of Twitter accounts at the centre of the network, with the vast majority around the edges of the network. Those at the centre have either a large number of followers, a large number of friends or both. Some of them appear to satisfy the criteria for what are known as "fake accounts", but the way that this term is used means that it does not really have any significance. Following graphical and statistical analysis, it was seen that there was no clear, linear relationship between number of followers, number of friends, number of tweets and number of likes. Of the most important users – that is, the most central to the network – there appeared to be some diversity.

This chapter focused on a textual analysis of the most recent, 5,000 tweets posted by the government cat accounts. To undertake the textual analysis, the Python library, Textblob, was used to calculate the frequency of each n-grams, where n is one, two or three. Twograms would cover people's names, such as Boris Johnson, and both twograms and threegrams would cover particular phrases that might have been used at the time of data collection, such as "No Deal Brexit" or "do or die". The n-gram frequencies for each government cat Twitter account were stored in dictionaries. During the calculation of the frequencies, any word that was in a stop words list were not counted.[19] The original stop words list consisted of "common English words". However, testing of the code highlighted more words that frequently appeared in the corpus that looked like they should be stop words. Examples of such words include: "I'm" and "A" – the capitalised versions were not in the original stop words list. Other stop words were those that were specific to the Twitter context, such as "RT" or "retweet" or, with regards to descriptions, versions of "All views are my own" or "RTs not endorsements". N-grams were also not

[19] The stop words list was downloaded from https://www.textfixer.com/tutorials/common-english-words.txt and is referenced on the Wikipedia "Stop words" page https://en.wikipedia.org/wiki/Stop_words.

counted if it began with: “http”, “https” or “t.co”, indicating websites or other tweets; “@”, indicating screenname; or “????”, indicated a word from a language that does not use the Latin alphabet. Once the frequencies were calculated, bar charts to show the most frequent n-grams were produced using Matplotlib. In order to make the bar chart easy to read, only the n-grams with frequencies greater than 15 were plotted. This cut-off point was chosen through trial and error. The code was stored in *AnalyseTweets.py*.

Government cats analysis of tweets

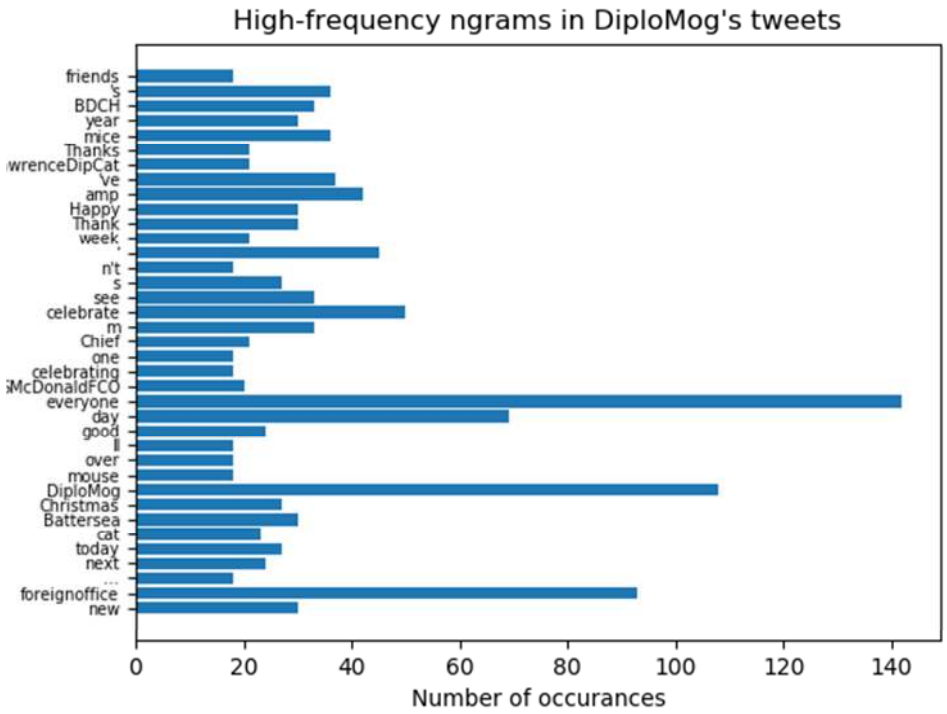

Figure 27: High-frequency n-grams in DiploMog's tweets

@DiploMog is the official Twitter Account for the Foreign Office cat, Palmerston. It is therefore not surprising that two of the most frequently-occuring n-grams. “DiploMog” is the screenname, so it would appear in all tweets and “foreignoffice” as a mention. Other frequent mentions include: @LawrenceDipCat, the cat for the British Embassy in Amman, and @SMcDonaldFCO, who is the permanent-undersecretary for the Foreign Office and the head of the Diplomatic Service. In Chapter 4,

@LawrenceDipCat and @SMcDonaldFCO were also highlighted two important members of the government cat network by in-degree; that is, they have a significant number of followers who are also within the government cat network. @DiploMog also frequently mentions Battersea Dogs and Cats Home a lot, via the mentions of @BDCH and @Battersea. Again, this is not surprising, given that that is where Palmerston as well as other government cats were obtained.

The most frequently mentioned word is “everyone”. From a cursory examination of a Twitter search results of “@DiploMog everyone”, this is due a large number of tweets targeted at all of DiploMog’s followers, in the same way that other Twitter users may use the word “tweeps”. (“Tweeps” seems a bit too informal for an official Twitter account of the Foreign Office.). A lot of tweets also mention “Thanks”, which appear to relate to tweets by @DiploMog’s friends and followers or at least Twitter users who have mentioned @DiploMog. Other high-frequent topics seemed to resolve around celebrations and particular occasions – from a cursory examination of tweets, examples include Christmas, Chinese New Year and annual Twitter anniversaries – and the job of being a government cat (Chief Mouser, Mouser-in-Chief), that is catching mice.

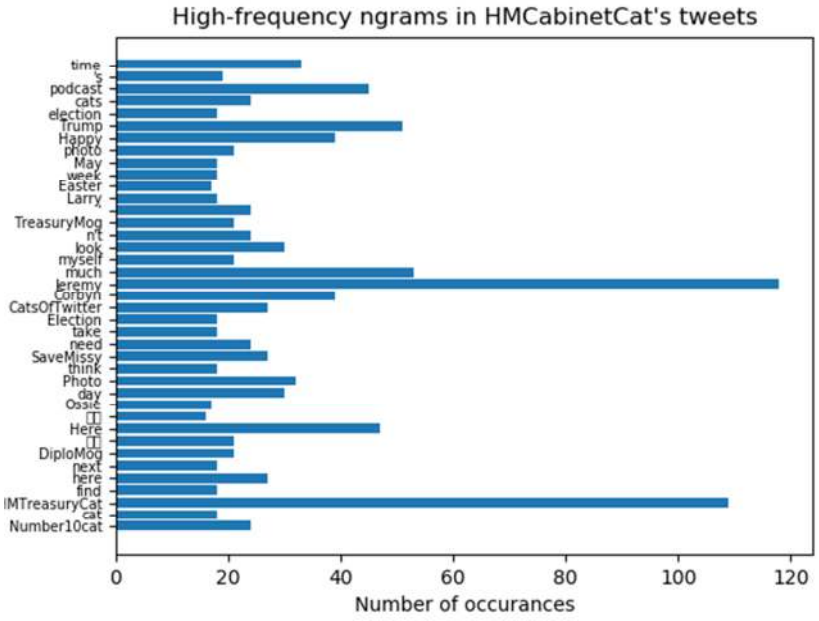

Figure 28: High-frequency n-grams in HMCabinetCat's tweets

The most highly-frequent word used by @HMCabinetCat is “Jeremy”. Given that one of the other highly-frequent words in “Corbyn”, it can be concluded that “Jeremy” refers to the current leader of the Labour Party. This is backed up a cursory examination of Twitter search results and the tone appears to be one of poking fun. Other highly-frequent political mentions are “Trump” (as in the current US President, Donald Trump), “May” (as in the former Prime Minister Theresa May) and “election” (as in the 2017 General Election). What is interesting is that Boris Johnson does not appear to be amongst highly frequent n-grams, but this may be simply down to time – he has only been Prime Minister for a month.

Unsurprisingly, “Cats” is a highly-frequent occurance and cat-related mentions include the government cats Larry (Number10cat), Gladstone (TreasuryMog, HMTreasuryCat) and Ossie. In particular, HMTreasuryCat is the second most-frequent occurrence after Jeremy. HMCabinetCat also frequently used the hashtag #CatsOfTwitter and #SaveMissy, a hashtag related the cat for Tewkesbury Town Hall. The #podcast is used to promote podcasts of HMCabinetCat purring. Anniversaries and occasions are celebrated through the use of the word “Happy”, of which the most frequent is Easter. Finally, a lot of HMCabinet’s tweets appear to contain photos and also mention a credit for the photo.

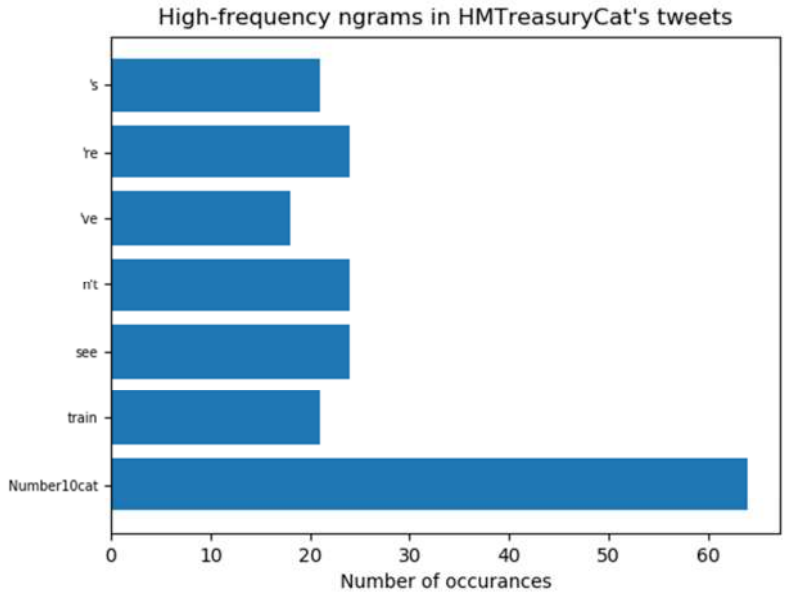

Figure 29: High-frequency n-grams in HMTreasuryCat's tweets

The most frequently occurring n-grams in @HMTreasuryCat's tweets were mentions of Number10cat, the Downing Street cat Twitter account. It also referred to "train" – a cursory examination of Twitter search results indicate that this is most likely to refer to #traingate, which referred to an incident in which Jeremy Corbyn may have misled about the number of available seats on a train. Otherwise, @HMTreasuryCat uses a lot of contractions.

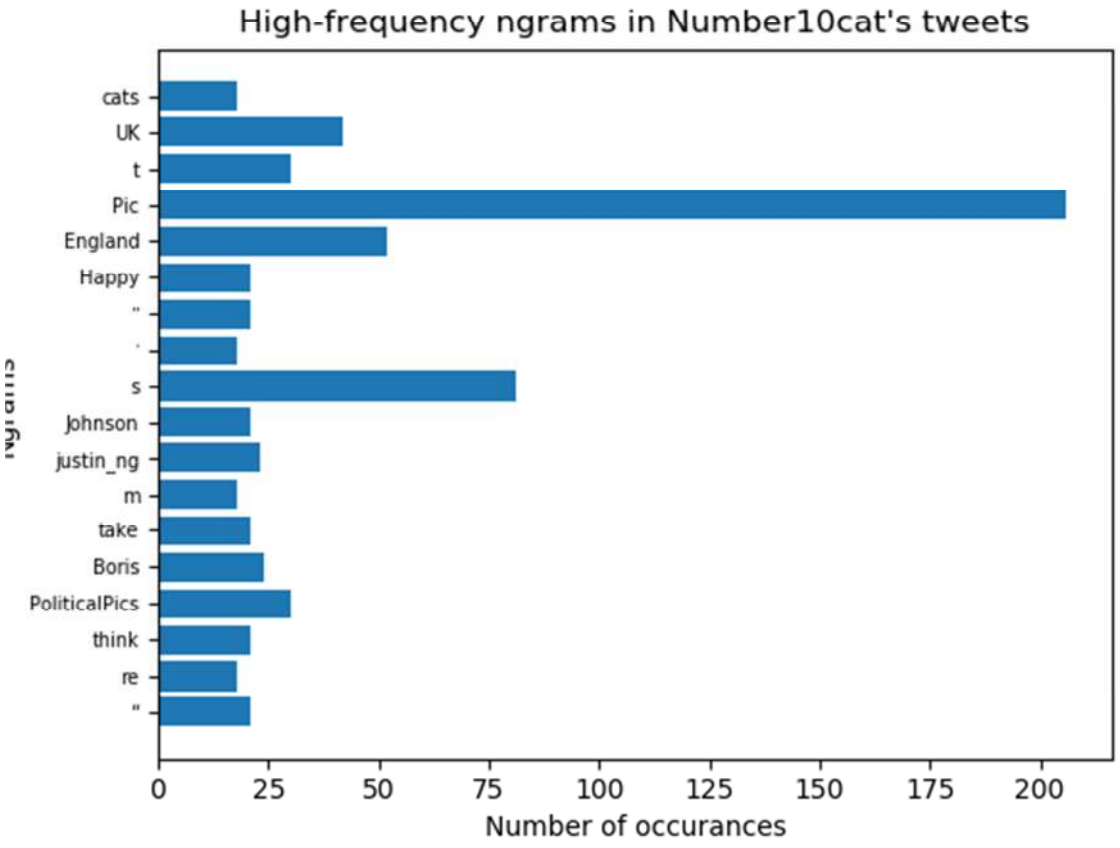

Figure 30: High-frequency n-grams in Number10cat's tweets

The most frequent word in @Number10cat's set of n-grams is "Pic", indicating that most of its tweet are pictures. Also amongst the set of frequent n-grams are @PoliticalPics and @justi_ng, which are the screennames for photo journalists. @PoliticialPics, according to his profile, specialises in the "Weird World of British Politics" and @justin_ng is an Australian freelance photographer based in London. Unlike @HMCabinetCat, @Number10cat frequently mentions Boris Johnson. Other frequent n-grams are "Cats" (appears to be cat pictures), "Happy" (appears to relate to Happy Caturday", UK and England.

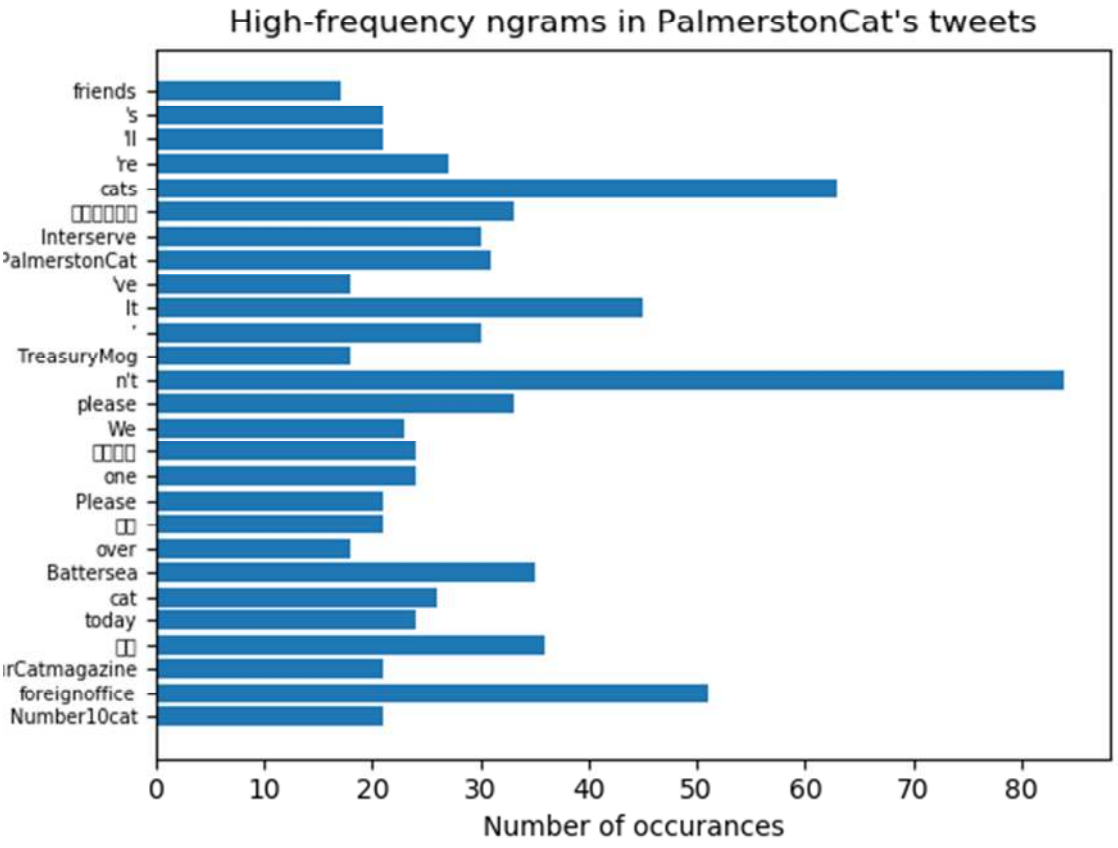

Figure 31: High-frequency n-grams in PalmerstonCat's tweets

@PalmerstonCat appears to use a lot of negative contractions (ending in "n't"). It's other most frequent n-grams are mentions of the @Foreignoffice as well as "Cats". The latter indicates cat content – this is supported by the use of the ngram "cat" and mentioning @YourCatmagazine, the Twitter account for a magazine for cat lovers. Other frequently-used n-grams are Interserve, a contractor for Foreign Office, and mentions of @Battersea, a Twitter account for Battersea Cats and Dogs Home. Finally, the government cats that @PalmerstonCat seems to engage with most are @TreasuryMog and @Number10cat.

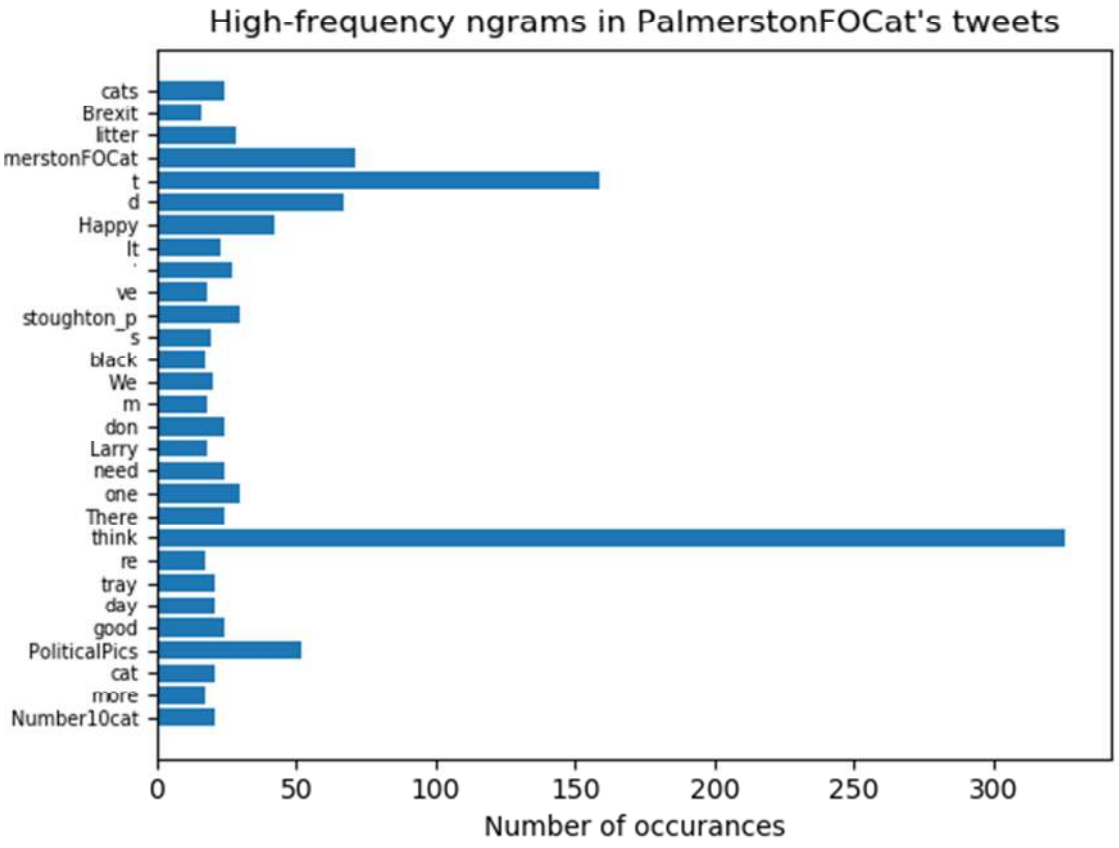

Figure 32: High-frequency n-grams in PalmerstonFOCat's tweets

@PalmerstonFOCat's most frequent ngram is "think", but there does not appear any particular topics to w_hich this ngram relates. For some reason, @PalmerstonFOCat is the only government cat that frequently mentions "Brexit. This is surprising given that it is one of the main, continually-reported political stories. Other frequent n-grams are "Cats", "litter" and "cat", which obviously related to cat content, and "Happy", which relates to greeting for particular anniversaries, such as New Year, Caturday and International Cat Day. The main government cat that @PalmerstonFOCat seems to engage with mostly is @Number10cat (Larry) and other frequent mentions are @PoliticalPics, the account for a photojournalist specialising in British politics, and @Stoughton_p, a British freelance journalist writing on French and Breton culture.

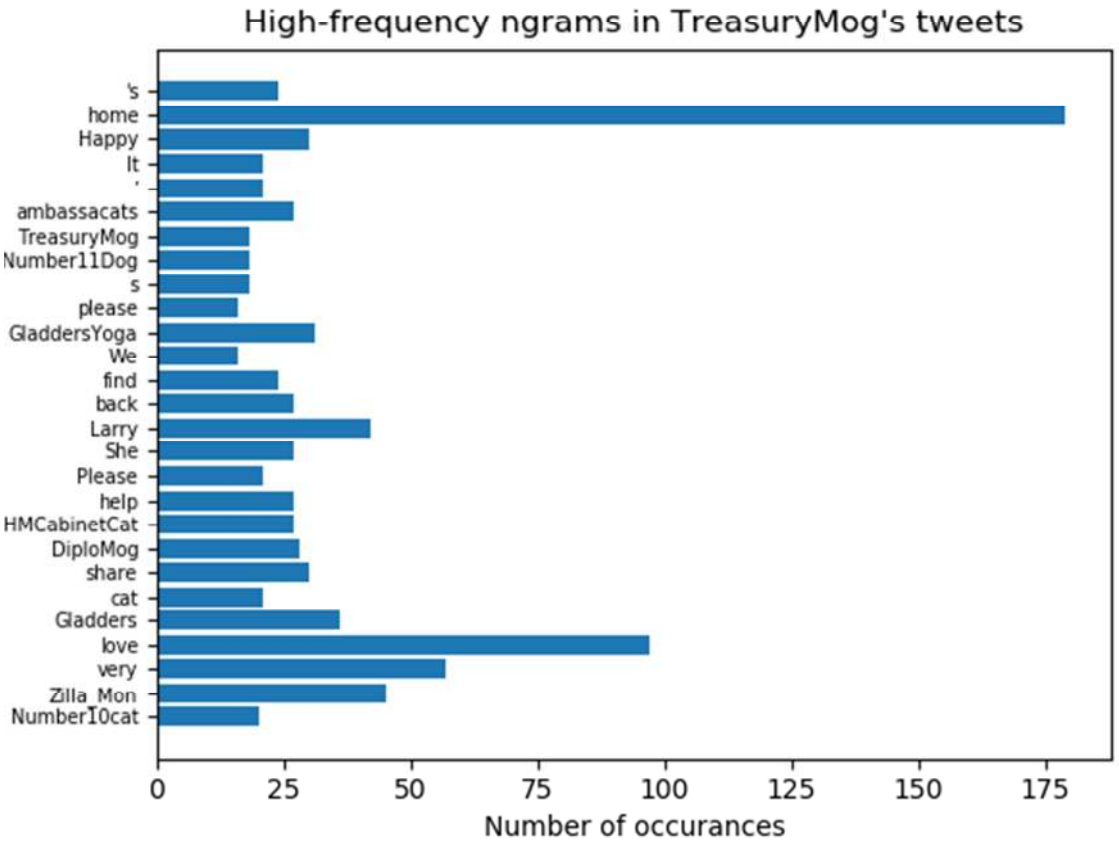

Figure 33: High-frequency n-grams in TreasuryMog's tweets

@TreasuryMog's most frequent ngram is "home", which appears to be used to raise awareness of lost cats or cats needing adoption. Other frequent n-grams are "love", "Happy", which related to greeting for anniversaries and events such as Caturday, and "Cat". The government cats that @TreasuryMog seems to engage with the most are @HMCabinetCat, @DiploMog and @Number10cat. Other frequent mentions are:

- #Ambassacats, a cat –related hashtag;
- @Number11Dog, "the unofficial spokesdog for the Treasury", which was obtained under the previous Chancellor Philip Hammond;
- #GladdersYoga, a hashtag for pictures of Gladstone stretching (as if he is doing Yoga)
- #Gladders, a hashtag got pictures of Gladstone and other cats

- @Zilla_Mon, another cat account

Summary

Government Cat	Gov cat mentions	Other Mentions	Hashtag	Words	Topics
DiploMog	@DiploMog	@foreignoffice @LawrenceDipCat @SMcDonaldFCO @BDCH @Battersea		Everyone Thanks Happy Christmas	Cat Catching mice
HMCabinetCat	Larry Ossie @TreasuryMog @HMTreasuryCat @DiploMog @Number10cat		#CatsOfTwitter #SaveMissy	Happy Easter Photo	Jeremy Corbyn Trump Theresa May 2017 election Cats
HMTreasuryCat	@Number10cat				Traingate (Jeremy Corbyn)
Number10cat		@PoliticalPics @justin_ng		Pic Cats Happy UK England	Boris Johnson
PalmerstonCat	@TreasuryMog @Number10cat	@foreignoffice @YourCatmagazine Interserve @Battersea		Cats	
PalmerstonFOCat	@Number10cat	@PoliticalPics @Stoughton_p		Think Cats Litter Cat Happy	Brexit
TreasuryMeog	@HMCabinetCat @DiploMog @Number10cat	@Number11Dog @Zilla_Mon	#Ambassacats #GladdersYoga #Gladders	Home Love Happy Cat	

There are significant differences in the frequent mentions, hashtags and frequent words used. This suggests that each government cat account is not operated by the same person. However, what each government cat Twitter account has in common is that many of the high-frequency words, such as

"Happy", "Thanks", "Christmas", "Easter", "Love" and cat-related words, could be classed as positive words or words that show positive emotion. This is in keeping with research by David Garcia, Antonios Garas and Frank Schweitzer that indicates that positive words tend to have higher frequency of occurrence in online written communication.[20] In light of Myrick's research that internet cat content tended to be used to increase happiness and reduce stress,[21] it is also not surprising that the government cat twitter feeds would have a degree of positivity. The relationship between the government cats and its most frequent mentions is visualised in Figure 34. There are 18 nodes and 26 edges. The direction of the arrows can be interpreted as "frequently mentions", e.g. @TreasuryMog frequently mentions @Number11Dog. The code is available in *AnalyseTweets2.py*..

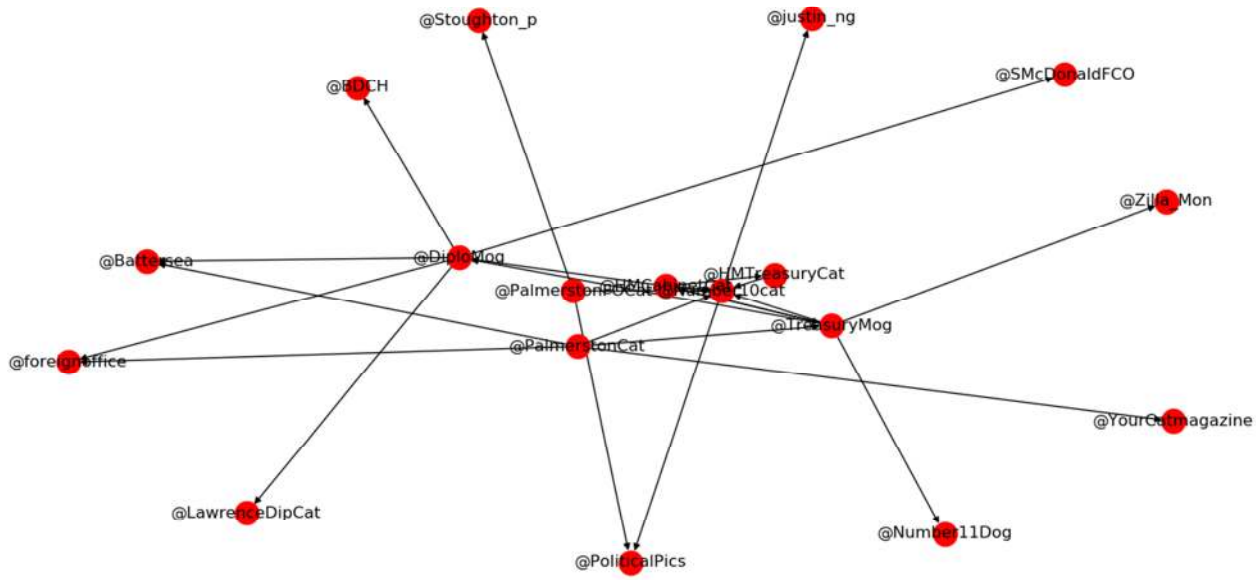

Figure 34: Relationship between Government cats and their highly frequent mentions

[20] Garcia, Garas and Schweitzer, 2012, 4
[21] Myrick, 2015, 174-5

6. Community Analysis of Government Cats' Followers, by Location

In Chapter 5, a textual analysis for the most recent 5,000 tweets of each government cat was undertaken, using the Python library Textblob. It sought to identify the main topics, as indicated by the frequency of particular words. There were topics which were unique to particular government cats but there were also common elements as well. By looking at the frequency of words, it was also possible to identify the particular Twitter accounts that each government account frequently engaged with.

This chapter focuses on a textual analysis of the locations of the 200 most recent followers of each government cat accounts. The textual analysis was undertaken using the same approach used to analyse the government cats' tweets. That is, the frequencies of each n-grams, where n is one, two or three, were calculated. A stop word list was not used, as the aim was to identify how followers self-identify themselves. Once the frequencies were calculated, bar charts to show the most frequent n-grams were produced using Matplotlib. In order to make the bar chart easy to read, only the n-grams with frequencies greater than the average number of occurrences (for each cat) were plotted. In all cases, the average number of occurrences was between two and three, so the average was rounded up to three. The code was stored in *AnalyseLocations.py*.

Assumptions

As mentioned in Chapter 1, the location box on a Twitter user's profile is a free text field. This means that user can essentially type anything they want. Hechl and others discovered that 34% of users in a particular dataset of Twitter profiles gave false or imaginary location names.[22] Ajao stated that less than 14% of users complete the field correctly,[23] by which is probably meant that they do not put their actual

[22] Hechl, Hong, Suh and Chi, 2011
[23] Ajao, 2015, 2

location. In this respect, the location field may not necessarily be the most accurate method for identifying where users are based. For this project, it assumed that the location field could be employed to identify how users describe themselves, particularly in areas where there are regional and national rivalries. For example, as shall be seen later in this chapter, it is interesting to see whether a user self-identifies as living in the United Kingdom, or in England, Scotland, Wales or Northern Ireland.

Findings

The most frequently used n-grams in the locations of @DiploMog's followers were "Here" and "I". From a cursory examination of the dataset, "Here" seems to indicate instances where the follower has stated an "imaginary" location, such as "Here and there" or "Somewhere over the Rainbow". In terms of actual locations, the biggest following appears to be in Somerset, with followings in Bath, Northampton, Leeds, London and Shrewsbury. Some of these have also mentioned "England", "UK" or "United" (perhaps meaning United Kingdom) in the location. Outside of the UK, there is a following in the USA, in particular Virginia (indicated by "VA") and Texas. There is also a following in Rio de Janeiro (in Brazil) and Sydney (in Australia).

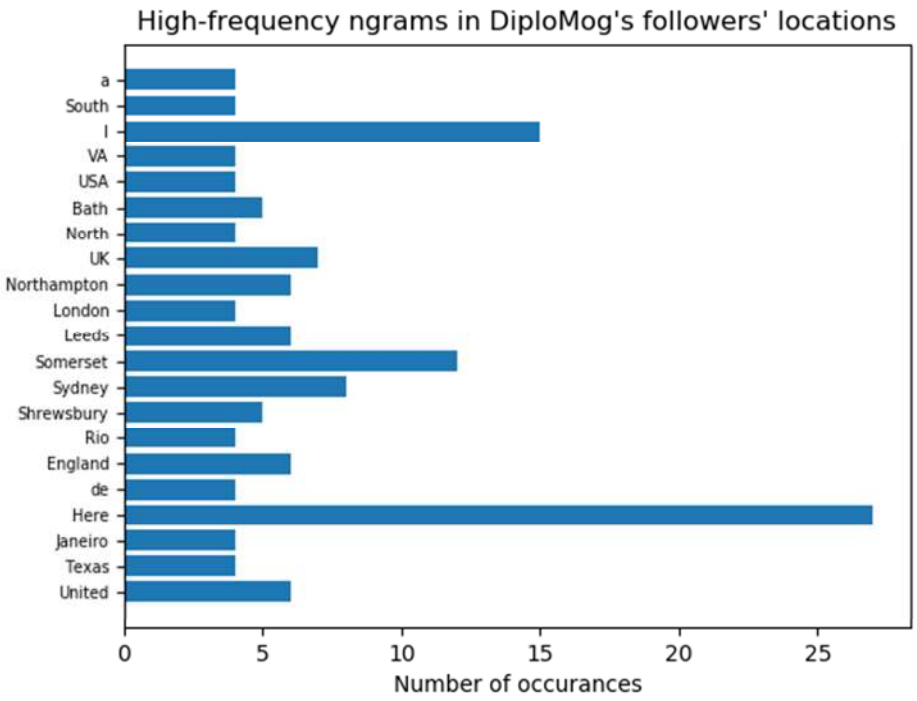

Figure 35: High-frequency n-grams in DiploMog's followers' locations

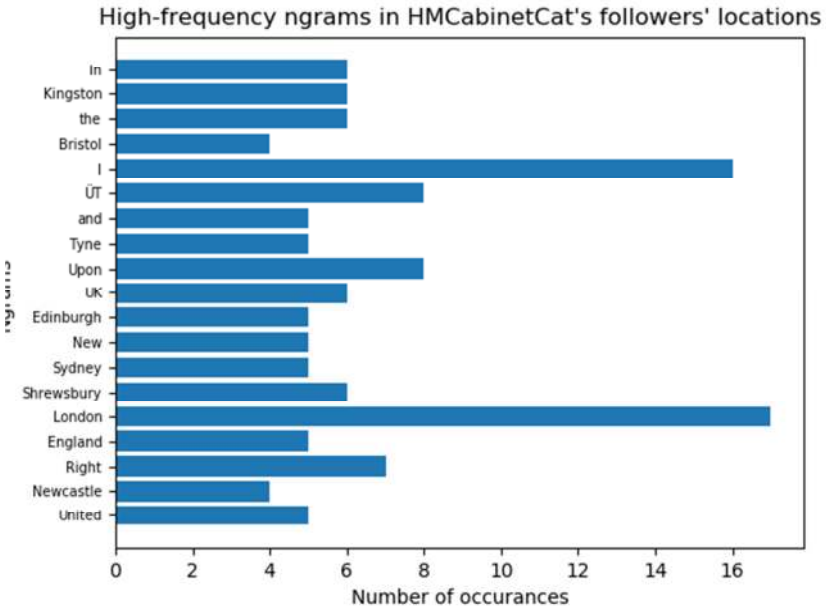

Figure 36: High frequency n-grams in HMCabinetCat's followers' locations

The most frequently occurring location for @HMCabinetCat's followers is London, with followings in Kingston, Bristol, Newcastle-Upon-Tyne, Edinburgh and Shrewsbury. Some of the domestic followers have also indicated "England", "UK" or "United" (probably meaning "United Kingdom"). Outside of the

UK, there appear to be followings in Utah and Sydney.

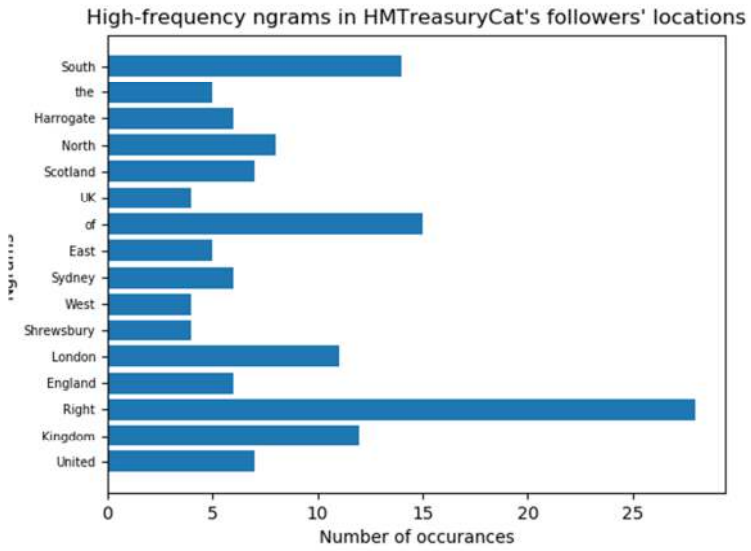

Figure 37: High-frequency n-grams in HMTreasuryCat's followers' locations

The most frequently occurring n-gram amongst the locations of @HMTreasuryCat's followers is "Right". From a cursory examination of the dataset, this appears to refer to an "imaginary" place, such as "Right Here". In terms of actual locations, most of the frequently occurring locations indicate followers in the "United Kingdom", "UK" or "England", although not all of the respective followers will state that specifically in their profile descriptions. Within the UK, there are particular followings in Harrogate, Shrewsbury and London. Outside of the UK, there is a particular following in Sydney.

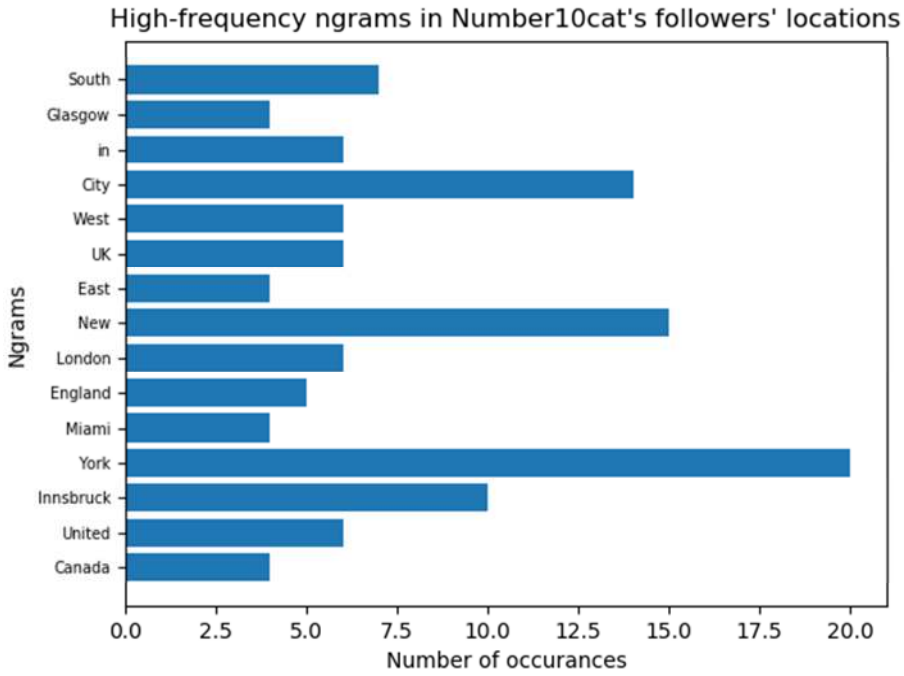

Figure 38: High-frequency n-grams in Number10cat's followers' locations

The most frequently occurring location amongst @Number10cat's followers is "York". Other locations in England, UK or United Kingdom are Glasgow and London. Outside of the UK, there are particular followings in Innsbruck (Austria), Miami (USA) and in Canada. The second most frequently occurring ngram is "New" but from a cursory examination of the dataset, this appears to refer to any location in the world with "New" in the name such as New York, New Delhi, New South Wales. The third most frequent occuring ngram is "City"; from a cursory examination of the dataset, this appears to refer particular cities anywhere in the world.

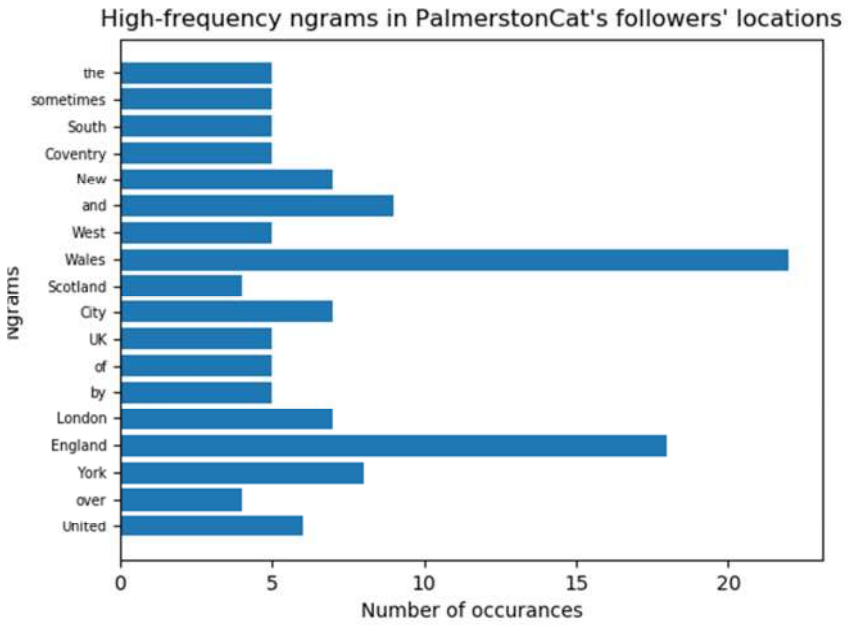

Figure 39: High-frequency n-grams in PalmerstonCat's followers' locations

The most frequently occurring locations for @PalmerstonCat's followers are Wales and England. There is also a following in Scotland, London, York and Coventry and "City" is also a frequently occurring n-gram, indicating followers based in cities anywhere throughout the world.

In Figure 40 below, it can be seen that the most frequently occurring locations for @PalmerstonFOCat's followers are Brussels and the UK (or United Kingdom). Within the UK, there are particular followings in England, Scotland, Leeds and London. Apart from Brussels, outside of the UK, there is a following in Australia, with Sydney as a frequently occuring ngram.

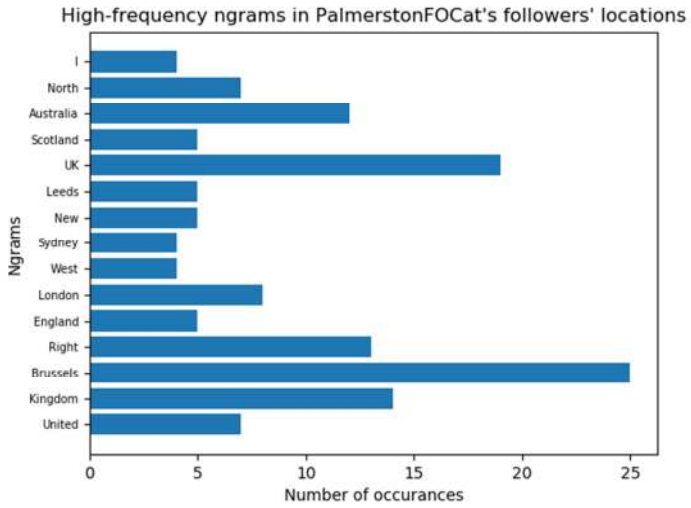

Figure 40: High-frequency n-grams in PalmerstonFOCat's followers locations

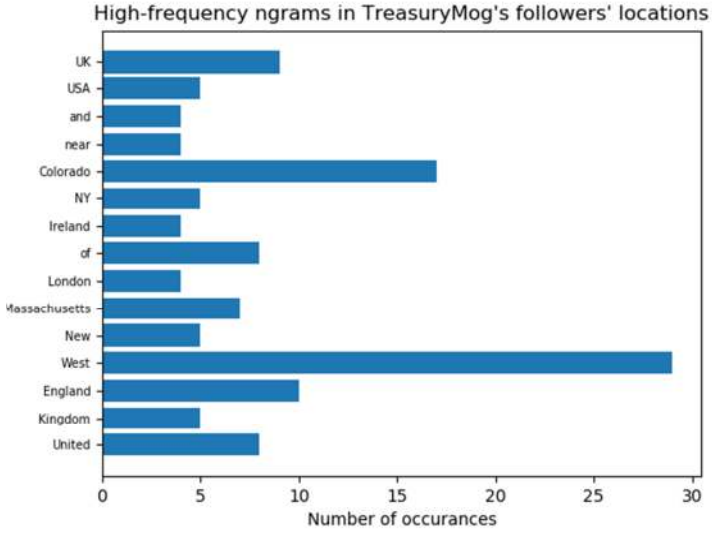

Figure 41: High-frequency n-grams in TreasuryMog's followers' locations

With regard to @TreasuryMog's followers, there is a following in England, the UK and the United Kingdom, with some followers in London. Outside of the UK, there are followers in Ireland and in the USA, particularly Massachusetts and Colorado.

Summary

DiploMog	UK United Kingdom England	 Somerset Bath Northampton Leeds London Shrewsbury	USA Brazil Australia	Virginia Texas Rio De Janeiro
HMCabinetCat	UK United Kingdom England	 London Kingston Bristol Newcastle upon Tyne Edinburgh Shrewsbury	USA Australia	Utah Sydney
HMTreasuryCat	UK United Kingdom England	 Harrogate Shrewsbury London	Australia	Sydney
Number10cat	UK United Kingdom England	 York Glasgow London	Austria USA Canada	Innsbruck Miami
PalmerstonCat	England Wales Scotland	London York Coventry		
PalmerstonFOCat	UK United Kingdom England Scotland	 Leeds London	Belgium Australia	Brussels Sydney
TreasuryMog	England UK United Kingdom	London	Ireland USA	 Massachusetts Colorado

All government cats have a lot of followers within the United Kingdom. They all have a following in London. This appears to be where most of TreasuryMog's followers are concentrated. Between the other government cats, the non-London following is spread throughout the country, from Somerset to

Edinburgh and Glasgow. It appears the followers seem to self-identify more by the town or city rather than by the “United Kingdom”, “England”, “Scotland” and “Wales”. Where the follower has stated the nation, most seem to self-identify as “United Kingdom”, “UK” or “England”. What is interesting, therefore, is that PalmerstonCat has a lot of followers self-identifying as being from “Wales” or “Scotland” and PalmerstonFOCat also has a lot of followers self-identifying as being from “Scotland”.

In terms of global following, all the government cats have followers self-identifying as being from Australia, and in particular Sydney. PalmerstonCat does not appear to have a significant following outside the UK. Apart from HMTreasuryCat and PalmerstonFOCat, all government cats also have followings spread throughout the United States. It is interesting, however, that DiploMog has a following in Brazil, Number10cat has a following in Austria and PalmerstonFOCat has a following in Brussels as it indicates that the government cats have a following beyond the English-speaking countries. During a period of the Brexit process – possibly the final months of the process - PalmerstonFOCat’s following in Brussels is also interesting because it is the seat of the European Union institutions.

7. Community Analysis of Government Cats' followers, by Description

In Chapter 6, a textual analysis of the stated locations for the most recent 200 followers of each government cat was undertaken, using the Python library Textblob. It sought to identify the main areas, both within the United Kingdom and globally, as indicated by the frequency of particular words. It is reasonable and unsurprising to say that most of the followers were within the United Kingdom. However, the followers were spread across the UK and not concentrated only in London. Furthermore, there are followings around the world, including non-English speaking countries.

This chapter focuses on a textual analysis of the profile descriptions of the 200 most recent followers of each government cat accounts. The textual analysis was undertaken using the same approach used to analyse the government cats' tweets and locations. That is, the frequencies of each n-grams, where n is one, two or three, were calculated. Once the frequencies were calculated, bar charts to show the most frequent n-grams were produced using Matplotlib. In order to make the bar chart easy to read, only the n-grams with frequencies greater than 10 (for each cat) were plotted. In all cases, the average number of occurences was between three and four. The code was stored in *AnalyseDescript.py*.

Findings

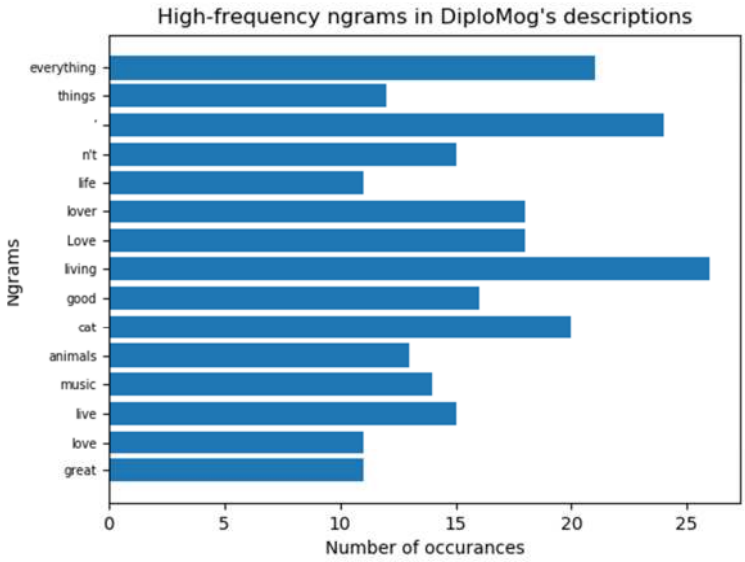

Figure 42: High-frequency n-grams in DiploMog's descriptions

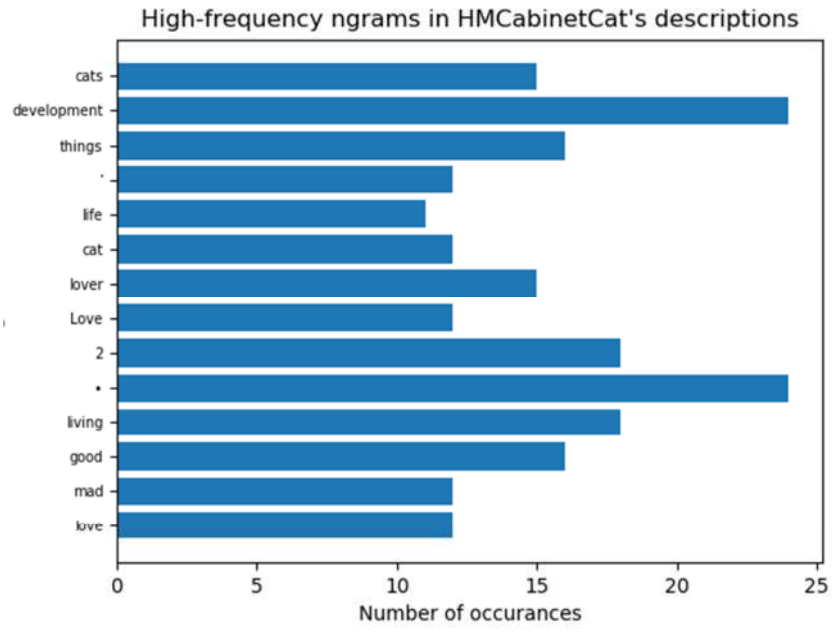

Figure 43: High-frequency n-grams in HMCabinetCat's descriptions

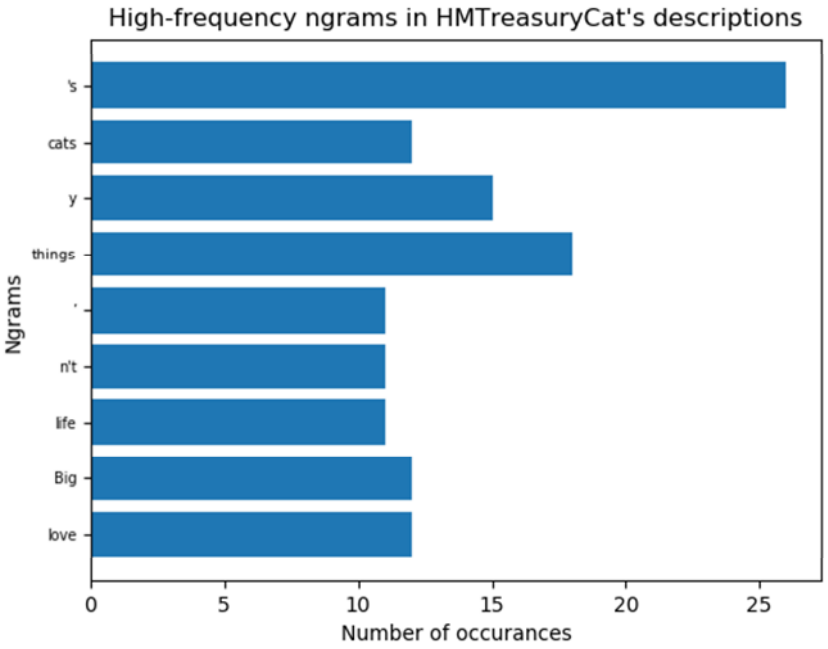

Figure 44: High-frequency n-grams in HMTreasuryCat's descriptions

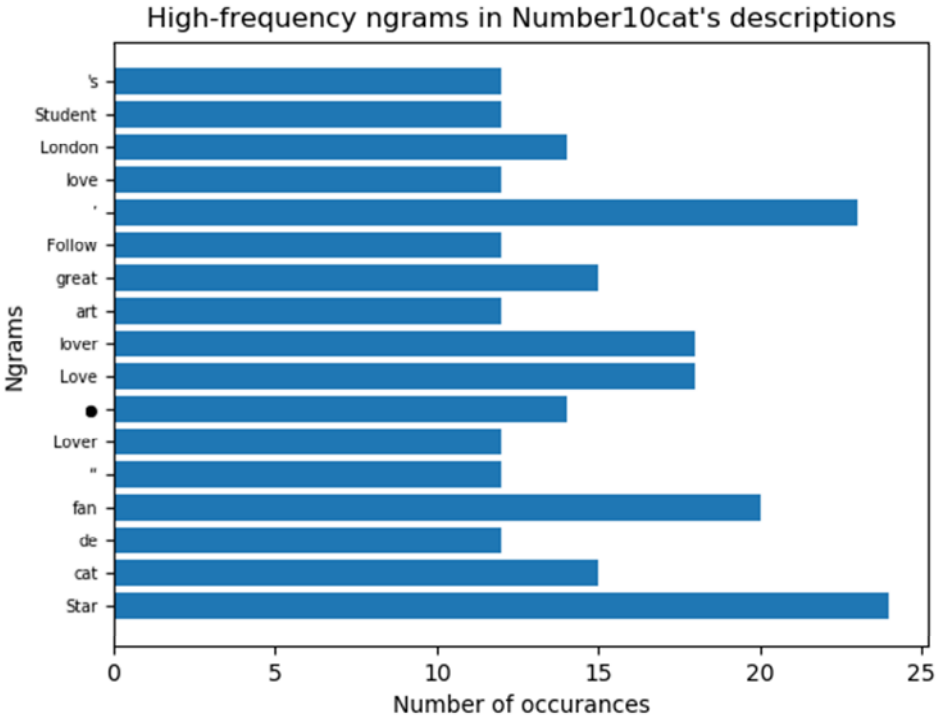

Figure 45: High-frequency n-grams in Number10cat's descriptions

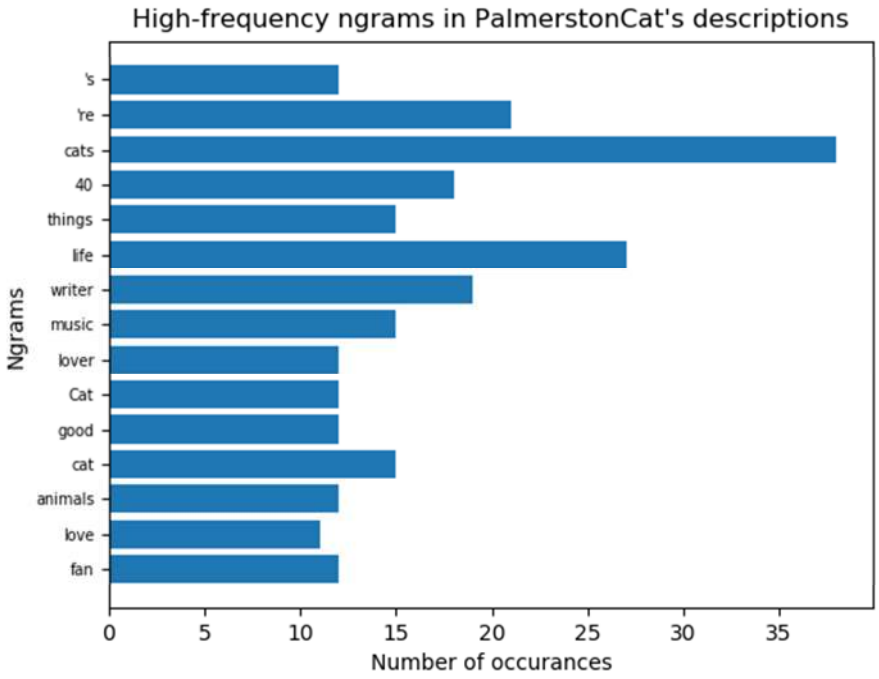

Figure 46: High-frequency n-grams in PalmerstonCat's descriptions

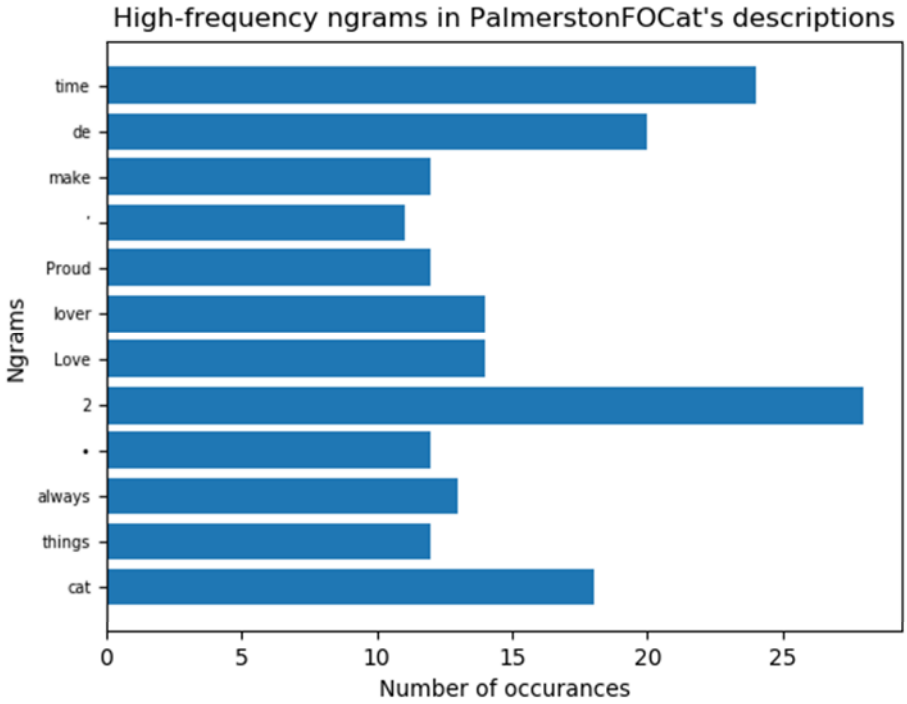

Figure 47: High-frequency n-grams in PalmerstonFOCat's descriptions

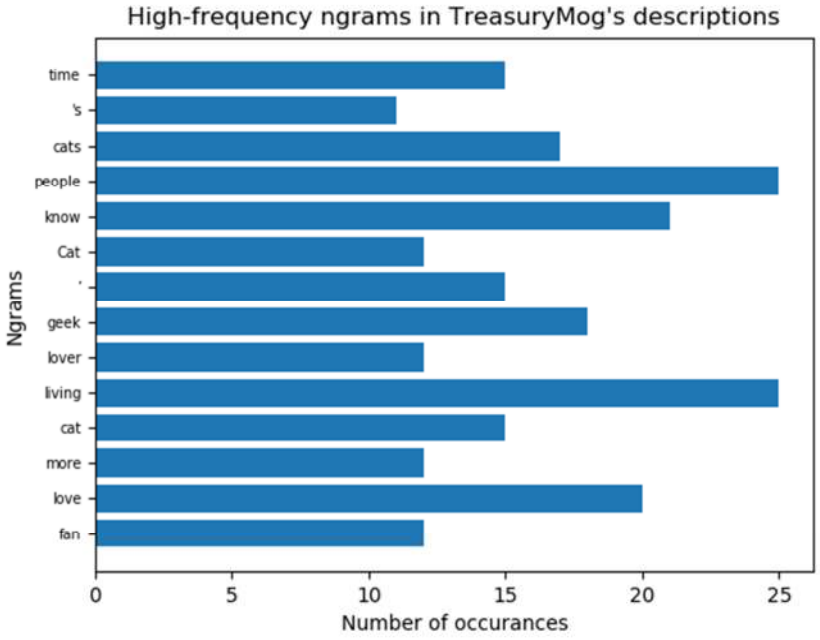

Figure 48: High-frequency n-grams in TreasuryMog's descriptions

Summary

DiploMog	Everything things	Life Living live	Lover Love Love	Good Great	Cat Animals	Music
HMCabinetCat	Things	Life living	Lover Love Love	Good	Cats Cat	Development Mad
HMTreasuryCat	things	Life	Love		Cats	Big
Number10cat			Lover Love lover	Great	Cat	Student London Follow Art Fan Star
PalmerstonCat	things	Life	Lover Love		Cats Cat Cat Animals	Writer Music Fan
PalmerstonFOCat	things		Lover Love		Cat	Time Make Proud always
TreasuryMog		Living	Lover Love		Cats Cat Cat	Time People Know Geek More Fan

Many of the government cats' followers use words such as "everything", "life", "lover", "good" and "cat" and related words, as indicated in the table above. The use of cat-related words is unsurprising, as it indicates an interest in cats. The use of love-related words indicates a self-identification as a lover of some sorts, in the sense of liking something. The main differences seem to be as follows:

- Many of DiploMog's followers are interested in or like music;
- HMCabinetCat has followers involved in some form of development role;
- Number10cat appear to have a lot of followers who describe themselves as students, being based in London, interested in art or possibly Star Wars fans;

- PalmerstonCat seems to have a lot of followers who self-identify as writers or interested in music;
- TreasuryMog seems to have a lot of followers who self-identify as “geeks”.

8. Conclusion and Lessons Learnt

In Chapter 2, four different Python libraries for accessing the Twitter API were compared, using a target user. The number of followers and friends obtained using each library and the time take were measured. The libraries compared were Tweepy, Twython, TweetPony and TwitterAP. The Time library was used to measure the time taken the Pandas library was used to store the data in a suitable format and the Matplotlib library was used to produce bar charts of the results. Since the time taken seemed to vary upon each iteration, a bagging approach was taken where the mean and standard deviation for the time taken was calculated before creating the bar chart. It was found that Twython was the least useful as the maximum number of friends and followers obtained was only six. Tweepy, the oldest library, took less time that TweetPony but more time than TwitterAPI. Tweepy also had the greatest variability in terms of time taken. However, Tweepy and TweetPony were also simpler to use than TwitterAPI. There was therefore no real benefit to using a library other than Tweepy. The main limitation with this chapter was that it was not possible to compare all nine libraries listed on the Twitter libraries page of the Twitter Developer site within the time and technical ccity available, especially as the comparison of libraries was not the primary purpose of the project.

In Chapter 3, a Python program was written to obtain the following details for each government cat:

1. the exact number of followers;
2. the exact number of friends;
3. the exact number of tweets posted;
4. the exact number of tweets liked;
5. the exact time and date of creation of the Twitter account

The libraries used were Tweepy, Matplotlib and Pandas. The above attributes for each government cat were recorded in a dataframe and descriptive statistics – mean, median and standard deviation – were also calculated and recorded in a new dataframe. The results were then plotted in a series of scatterplots in order to identify if there was any linear relationship between the creation date (as a proxy for duration), the number of followers, the number of friends, the number of tweets and the number of likes. From the scatterplots, as well as the initial research cited in Table 1, that there was no clear linear relationship between any of the attributes. Perhaps the main limitation in Chapter 3 was that no attempt to develop a model using linear regression was undertaken. It was judged, however, that this would not have been useful as the dataset was not sufficiently large. For five variables amongst six observations, at least 50 observations would have been needed. For linear regression to be worthwhile, the scope of the project would have had to be expanded beyond central government departments to include other public sector organisations such as local councils and British embassies.

In Chapter 4, a network analysis was undertaken using the Networkx library. An assumption was made that each government cat Twitter account was at the centre of its own (local) network. A directed graph was then produced showing the layout of the government cats and the followers and friends in a single (global) network. A limitation with the graph produced was that it was not possible to label the edges in a readable fashion – it was possible to obtain details of up to 200 followers of each government cat. However, the graph was useful in that it confirmed that the hypothesis that there were a handful of Twitter users at the centre of the network. The number of central users differed slightly depending on the whether one measured the in-degree, the out-degree or the degree of a node. It was judged, however, that the most important users in terms of information flow would have to be those with either a large number of followers (high in-degree) or large number of friends (high out-degree). As a result, the list of important users were then extrapolated by finding the union of both. There were 19 important users. It was then possible to use tweepy to find out the number of followers, friends, tweets and likes for the important users as well as whether they had a profile and/or description. A correlation

matrix was then produced to find out the relationship between the different attributes. It was found that there was only a strong correlation between the number of tweets and the number of likes; all other attributes were weakly or very weakly correlated. Perhaps the limitation with the correlation matrix was that it focused only on the most central users; it's possible that a correlation matrix for all members of the network would have produced different results. For each central user of the network, the following attributes were calculated in order to identify potential "fake" accounts:

- High follower to tweet ratio (large number of followers, small number of tweets);
- High friend to follower ratio (large number of friends, small number of followers); network
- The absence of a description;
- Very high tweet to duration ratio (large average number of tweets per day).

A couple of users did stand out but what this showed was that even if an account showed the most commonly-recognised characteristics of a fake account, that did not necessarily mean that the account was fake. This aspect could be extended with further research on fake accounts. Of course, the main limitation with this network was that it only comprised the 200 most recent followers and friends, but this was a Twitter-imposed restriction.

In Chapters 5, 6 and 7, a textual analysis was undertaken of each government cat's 5,000 most recent tweets and the locations and profile descriptions of the 200 most recent followers. The Textblob library was used to count the number of words, twograms and threegrams and ignoring stopwords or websites. For sake of readability, only n-grams with a count above a certain line was plotted in a bar chart. This line was identified through a process of trial and error. It was discovered that, whether looking at topics of tweets, locations or profile descriptions, the results showed that, for each government cat, there were both common elements and key differences. From the analysis of the

tweets, it was possible to identify the relationship between the government cats and their most frequent mentions as a graph.

9. Bibliography

Ajao, Oluwaseun; Hong, Jun; Liu, Wieru; "A survey of location inference techniques on Twitter", *Journal of Information Science*, 2015, 1-10

BBC News, 'Purr-fect ending fur Humphrey', *BBC*, 25 November 1997, http://news.bbc.co.uk/1/hi/uk/politics/34455.stm

BBC News, 'Downing Street cat Humphrey dies', 20 March 2006, http://news.bbc.co.uk/1/hi/uk_politics/4823834.stm

Beevolve, "An Exhaustive study of Twitter Users Across the World", October 2012, http://www.beevolve.com/twitter-statistics/

Calderelli,Guido; Chessa, Alessandro; *Data Science and Complex Networks* (Oxford University Press: 2016)

Chambre, Agnes, "David the Cameron addresses Larry the Cat rumours", *PoliticsHome*, 13 July 2016, https://politicshome.com/news/uk/politics/dot-commons-diary/77250/david-cameron-addresses-larry-cat-rumours

Davies, Caroline, "More questions over how No 10 handled the kitty", *The Telegraph*, 24 November 1997, published by *Internet Archive Way Back Machine* https://web.archive.org/web/20071205223254/http://www.telegraph.co.uk/htmlContent.jhtml?html=%2Farchive%2F1997%2F11%2F24%2Fnmog124.html

Day, Chris, “The bureaucats at the heart of government”, *National Archives*, 7 June 2016

Garcia, David; Garas, Antonios; Schweitzer, Frank; “Positive words carry less information than negative words”, *EPJ Data Science,* 2012, 1, 3, 1-12

Hechl, B; Hong, L; Suh, B; Chi E.H.; “Tweets from Justin Bieber’s heart: the dynamics of the location, *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems,* AGM 2011, 237-246

Kennedy, Maev, “Mew-turn by Cabinet Office as government gets two new cats”, *The Guardian*, 9 December 2016, https://www.theguardian.com/politics/2016/dec/09/cabinet-office-gets-two-new-cats-uk-government

Millward, David, ‘Humphrey…the Downing Street dossier”, *The Telegraph,* 14 March 2005, https://www.telegraph.co.uk/news/uknews/1485584/Humphrey...-the-Downing-Street-dossier.html

Myrick, Jessica Gall, ‘Emotion, regulation and procrastination, and watching cat videos online: Who watches Internet cats, why and to what effect?’, *Computers in Human Behaviour*,2015, 52, 168-175, https://www.sciencedirect.com/science/article/pii/S0747563215004343 [Last accessed: 14 January 2019]

Porter, Harriet, ‘Why cool cats rule the internet?’, *The Telegraph*, 1 July 2016, https://www.telegraph.co.uk/pets/essentials/why-cool-cats-rule-the-internet/

Scott, Lauren, “A softer side of government: How Larry the cat became a purr-fect political companion on Downing Street”, *CBC*, 22 January 2017, https://www.cbc.ca/news/world/larry-chief-mouser-downing-street-1.3932276

Sleator, Laurence, “The new PM’s first job: Impress the cat”, *BBC*, 24 July 2019, https://www.bbc.co.uk/news/uk-politics-49049852

The Telegraph,”Meet the cats of Westminster: The Government mousers-in-chief”, 4 September 2017, https://www.telegraph.co.uk/women/politics/meet-cats-westminster-government-mousers-in-chief/

‘Twitter Libraries’, https://developer.twitter.com/en/docs/developer-utilities/twitter-libraries.html

Van Vechten, Carl, “The Tiger in the House”, (New York: Bartleby.com, 2000), https://www.bartleby.com/234/6.html

White, Michael, “Humphrey, cat; born 1988, died 2006”, *The Guardian*, 21 March 2006, https://www.theguardian.com/politics/2006/mar/21/1

10. Appendix A: Python code

LibraryTest.py

```
1.  import tweepy
2.  from twython import Twython
3.  import tweetpony
4.  from TwitterAPI import TwitterAPI
5.
6.  import time
7.  import pandas as pd
8.
9.  twitterlib = ["Tweepy", "Twython", "TweetPony", "TwitterAPI"]
10. df = pd.DataFrame()
11. u = "@DiploMog"
12. count = [200]
13.
14.
15. for n in count:
16.
17.     for i in range(10):
18.
19.         # Calculate time for Tweepy
20.
21.         start = time.time()
22.
23.         tweepy_consumer_key = "cr9wJjmmT2CcGXqjsx7aWvsbA"
24.         tweepy_consumer_secret = "QAjnpYxS59MEil5IEpJJzGP0WqDajklMcXxcxiaJLh9doBuQ6g"
25.         tweepy_access_token = "897909419996041218-z27A8n2L6UcmUPuRbZhQBcpAAEk1Jxj"
26.         tweepy_access_token_secret = "iOVhZJSMxV5XPwlePFNwqiPYUEj8QOFCXOAFcM50Q9dDC"
27.
28.         auth = tweepy.OAuthHandler(tweepy_consumer_key, tweepy_consumer_secret)
29.         auth.set_access_token(tweepy_access_token, tweepy_access_token_secret)
30.         api = tweepy.API(auth,wait_on_rate_limit_notify=True)
31.
32.         friends = api.friends(id=u,count=n)
33.         followers = api.followers(id=u,count=n)
34.
35.         end = time.time()
36.
37.         df = df.append({"Count": n,
38.                         "Library": "Tweepy",
39.                         "Time": end - start,
40.                         "Friends": len(friends),
41.                         "Followers": len(followers)},
42.                        ignore_index=True)
43.
44.
45.         # Calculate time for Twython
46.
47.         start = time.time()
48.
49.         twython_consumerkey = '2IiwQDV5c4OeUEA8PI0aqvg5g'
50.         twython_consumersecret = 'RqEDv68Brt3JKVetWAeNFUObP5VeeRloBbQ0V9eqRFCerbas1I'
51.
52.         twitter = Twython(twython_consumerkey, twython_consumersecret, oauth_version=2)
53.         twython_accesstoken = twitter.obtain_access_token()
```

```
54.
55.         twitter = Twython(twython_consumerkey, access_token=twython_accesstoken)
56.
57.         friends = twitter.get_friends_list(screen_name=u,count=n)
58.         followers = twitter.get_followers_list(screen_name=u,count=n)
59.
60.         end = time.time()
61.
62.         df = df.append({"Count": n,
63.                         "Library": "Twython",
64.                         "Time": end - start,
65.                         "Friends": len(friends),
66.                         "Followers": len(followers)},
67.                        ignore_index=True)
68.
69.
70.         # Calculating time for TweetPony
71.
72.         start = time.time()
73.
74.         api = tweetpony.API(consumer_key = "s08vLRIM5VnqCUqFVYTaV3ET9",
75.                             consumer_secret = "q4N5vB5KQrakhjAcWAk8dKOeTn4896cURoAg33A37UjGG2jJxf",
76.                             access_token = "897909419996041218-FIxS4NDoYbgrorjbxns9hYadZDiLg3J",
77.                             access_token_secret = "4p4RMc15bPp5Mknywkc1dY5HbbsJDwSnOGkcw4KLSFyQz")
78.
79.         followers = api.followers(screen_name = u, count=n)
80.         friends = api.friends(screen_name = u, count=n)
81.
82.         end = time.time()
83.
84.         df = df.append({"Count": n,
85.                         "Library": "TweetPony",
86.                         "Time": end - start,
87.                         "Friends": len(friends),
88.                         "Followers": len(followers)},
89.                        ignore_index=True)
90.
91.         # Calculating time for TwitterAPI
92.
93.         start = time.time()
94.
95.         consumer_key = "7bFghx8idOToeNK8YCc1OrwQ5"
96.         consumer_secret = "2GNYr9LBJJXJ3pfJKSlME4H6dXRYUt2kXXSnyRTQedS182Cirs"
97.         access_token = "897909419996041218-VrQAbZABKWP91W5W3cyp6d5j5VSYeJX"
98.         access_token_secret = "v6I9ev1C6FNJLzZFlOnPg7MfYex8KvBP4FuyK7xLq6Es5"
99.
100.             api = TwitterAPI(consumer_key, consumer_secret, access_token, access_token_secret)
101.
102.             followers = api.request('followers/list', {'screen_name':u, 'count':n})

103.             friends = api.request('friends/list', {'screen_name':u, 'count':n})
104.
105.             end = time.time()
106.
107.             followernum = 0
108.             friendnum = 0
109.             for item in followers.get_iterator():
110.                 if 'screen_name' in item:
111.                     followernum +=1
112.             for item in friends.get_iterator():
113.                 if 'screen_name' in item:
114.                     friendnum +=1
```

```
115.
116.                 df = df.append({"Count": n,
117.                                 "Library": "TwitterAPI",
118.                                 "Time": end - start,
119.                                 "Friends": friendnum,
120.                                 "Followers": followernum},
121.                                ignore_index=True)
122.
123.
124.         export_csv = df.to_csv(r"CSV\compare.csv", index = None, header = True)
```

AnalyseLibraries.py

```
1.  import pandas as pd
2.  import matplotlib.pyplot as plt
3.
4.  df = pd.read_csv("CSV\compare.csv")
5.
6.  headings = ["Followers","Friends","Time"]
7.  twitterlib = ["Tweepy","Twython","TweetPony","TwitterAPI"]
8.  FollowersMean = []
9.  FollowersStd = []
10. FriendsMean = []
11. FriendsStd = []
12. TimeMean = []
13. TimeStd = []
14.
15.
16. for i in twitterlib:
17.     for j in headings:
18.         vars()[i + "DF"] = df[df.Library == i]
19.         vars()[j + "Mean"].append(vars()[i + "DF"][j].mean())
20.         vars()[j + "Std"].append(vars()[i + "DF"][j].std())
21.
22. meandf = pd.DataFrame({"Library": twitterlib,
23.                         "Followers": FollowersMean,
24.                         "Friends": FriendsMean,
25.                         "Time": TimeMean})
26.
27. stddf = pd.DataFrame({"Library": twitterlib,
28.                         "Followers": FollowersStd,
29.                         "Friends": FriendsStd,
30.                         "Time": TimeStd})
31.
32. print meandf
33. print stddf
34.
35. meandf.plot(kind = "bar", x = "Library", y = "Time")
36. plt.savefig("TimeMean.png")
37. stddf.plot(kind = "bar", x = "Library", y = "Time")
38. plt.savefig("TimeStd.png")
```

DataCollect_Basic.py

```
import tweepy
import pandas as pd
import matplotlib.pyplot as plt

# Obtain access to Twitter's Application Programming Interace

tweepy_consumer_key = "cr9wJjmmT2CcGXqjsx7aWvsbA"
tweepy_consumer_secret = "QAjnpYxS59MEil5IEpJJzGP0WqDajklMcXxcxiaJLh9doBuQ6g"
tweepy_access_token = "897909419996041218-z27A8n2L6UcmUPuRbZhQBcpAAEk1Jxj"
tweepy_access_token_secret = "iOVhZJSMxV5XPwlePFNwqiPYUEj8QOFCXOAFcM50Q9dDC"

auth = tweepy.OAuthHandler(tweepy_consumer_key, tweepy_consumer_secret)
auth.set_access_token(tweepy_access_token, tweepy_access_token_secret)
api = tweepy.API(auth,wait_on_rate_limit_notify=True)


# Extracting data about Government cat. Output produced to screen and CSV file.

govCats = ["@DiploMog",
           "@HMCabinetCat",
           "@Number10cat",
           "@PalmerstonFOCat",
           "@PalmerstonCat",
           "@TreasuryMog",
           "@HMTreasuryCat"]

countdf = pd.DataFrame(columns = ["Gov cat","Followers","Friends"])

for cat in govCats:
    cat1 = cat[1:len(cat)] # omits '@' character from the username
    user = api.get_user(cat1)
    countdf = countdf.append({"Gov cat": user.screen_name,
                              "Followers": user.followers_count,
                              "Friends": user.friends_count,
                              "Tweets": user.statuses_count,
                              "Likes": user.favourites_count,
                              "Joined": user.created_at},
                             ignore_index=True)

export_csv = countdf.to_csv(r"basicdata.csv", index = None, header = True)

print countdf


# Calculating the mean, median and standard deviation for data for the number
# of followers, number of friends and the number of tweets and likes made.
# Output produced to screen and CSV file.

headings = ["Followers",
            "Friends",
            "Tweets",
            "Likes"]

dfMean = []
dfMedian = []
dfDiff = []
dfStd = []

for column in headings:
    countdf[column] = countdf[column].astype(float)
    dfMean.append(countdf[column].mean(axis = 0))
```

```
    dfMedian.append(countdf[column].median(axis = 0))
    dfStd.append(countdf[column].std(axis = 0))

statsdf = pd.DataFrame({"Attribute": headings,
                        "Mean": dfMean,
                        "Median": dfMedian,
                        "St. Dev": dfStd})

export_csv = statsdf.to_csv(r"basicstats.csv", index = None, header = True)

print statsdf


# Number of tweets converted to numeric, in order to produce scatterplots
# showing relationship to duration of twitter page.

countdf["Joined"] = pd.to_numeric(countdf.Joined)

print "Producing Tweets versus Followers..."
countdf.plot(kind="scatter", x="Tweets", y="Followers", color="red")
plt.savefig("TweetsFollowers.png")
print "Producing Tweets versus Friends..."
countdf.plot(kind="scatter", x="Tweets", y="Friends", color="blue")
plt.savefig("TweetsFriends.png")
print "Producing Tweets versus Likes..."
countdf.plot(kind="scatter", x="Tweets", y="Likes", color="green")
plt.savefig("TweetsLikes.png")
print "Producing Tweets versus Joined..."
countdf.plot(kind="scatter", x="Tweets", y="Joined", color="black")
plt.savefig("TweetsJoined.png")

print "Producing Joined versus Followers..."
countdf.plot(kind="scatter", x="Joined", y="Followers", color="red")
plt.savefig("JoinedFollowers.png")
print "Producing Joined versus Friends..."
countdf.plot(kind="scatter", x="Joined", y="Friends", color="blue")
plt.savefig("JoinedFriends.png")
print "Producing Joined versus Likes..."
       countdf.plot(kind="scatter", x="Joined", y="Likes", color="green")
       plt.savefig("JoinedLikes.png")
       print "Producing Joined versus Tweets..."
       countdf.plot(kind="scatter", x="Joined", y="Tweets", color="black")
       plt.savefig("JoinedTweets.png")


       # Relationship between number of followers, friends and likes.

       print "Producing Followers versus Friends..."
       countdf.plot(kind="scatter", x="Followers", y="Friends", color="red")
       plt.savefig("FollowersFriend.png")
       print "Producing Followers versus Likes..."
       countdf.plot(kind="scatter", x="Followers", y="Likes", color="blue")
       plt.savefig("FollowersLikes.png")
       print "Producing Friends versus Likes..."
       countdf.plot(kind="scatter", x="Friends", y="Likes", color="green")
       plt.savefig("FriendsLikes.png")

       print "End Scatterplots"
```

DataCollect.py

```
import tweepy
import json

tweepy_consumer_key = "cr9wJjmmT2CcGXqjsx7aWvsbA"
tweepy_consumer_secret = "QAjnpYxS59MEil5IEpJJzGP0WqDajklMcXxcxiaJLh9doBuQ6g"
tweepy_access_token = "897909419996041218-z27A8n2L6UcmUPuRbZhQBcpAAEk1Jxj"
tweepy_access_token_secret = "iOVhZJSMxV5XPwlePFNwqiPYUEj8QOFCXOAFcM50Q9dDC"

auth = tweepy.OAuthHandler(tweepy_consumer_key, tweepy_consumer_secret)
auth.set_access_token(tweepy_access_token, tweepy_access_token_secret)
api = tweepy.API(auth,wait_on_rate_limit_notify=True)


govCats = ["@DiploMog",
           "@HMCabinetCat",
           "@Number10cat",
           "@PalmerstonFOCat",
           "@PalmerstonCat",
           "@TreasuryMog",
           "@HMTreasuryCat"]

for cat in govCats:
    print "Reading from", cat, "......"
    friends = api.friends(id=cat,count=200)
    followers = api.followers(id=cat,count=200)
    tweets = api.user_timeline(screen_name=cat,count=5000)
    friends_list = []
    for i in friends:
        friends_list.append(i.screen_name)
    followers_list = []
    location_list = []
    description_list = []
    for i in followers:
        followers_list.append(i.screen_name)
        location_list.append(i.location)
        description_list.append(i.description)
    tweets_list = []
    for i in tweets:
        tweets_list.append(i.text)
    friends_file = "Datasets\\" + cat + "friends.txt"
    followers_file = "Datasets\\" + cat + "followers.txt"
    tweets_file = "Datasets\\" + cat + "tweets.txt"
    location_file = "Datasets\\" + cat + "locations.txt"
    description_file = "Datasets\\" + cat + "descriptions.txt"
    with open(friends_file, "w") as f:
        json.dump(friends_list, f)
    with open(followers_file, "w") as f:
        json.dump(followers_list, f)
    with open(tweets_file, "w") as f:
        json.dump(tweets_list, f)
    with open(location_file, "w") as f:
        json.dump(location_list, f)
    with open(description_file, "w") as f:
        json.dump(description_list, f)
```

AnalyseFollowers.py

```
import pandas as pd
import json
import networkx as nx
import matplotlib.pyplot as plt
import operator

headings = ["@DiploMog",
            "@HMCabinetCat",
            "@Number10cat",
            "@PalmerstonFOCat",
            "@PalmerstonCat",
            "@TreasuryMog",
            "@HMTreasuryCat"]


# Read followers data from text file into dictionary and convert into
# pandas DataFrame

print "Reading followers data..."

rows = {}
for cat in headings:
    file = "Datasets\\" + cat + "followers.txt"
    with open(file) as f:
        data = json.loads(f.read())
    rows[cat] = data


# Read friends data from text file into dictionary and convert into
# pandas DataFrame

rows1 = {}
for cat in headings:
    file = "Datasets\\" + cat + "friends.txt"
    with open(file) as f:
        data = json.loads(f.read())
    rows1[cat] = data


# Assuming each government cat is at the centre of its own network and some users
# follow more than one government cat, create a global network of all government
# cats with followers and friends to identify connecting users.

catGraph = nx.DiGraph()
for cat in rows:
    catGraph.add_node(cat)
    for follower in rows[cat]:
        catGraph.add_node(follower)
        catGraph.add_edge(follower,cat)
for cat in rows1:
    for friend in rows1[cat]:
        if friend not in catGraph:
            catGraph.add_node(friend)
        catGraph.add_edge(cat,friend)

print "Number of users in global network:", catGraph.number_of_nodes()
print "Number of connections in global netwwork:", catGraph.number_of_edges()
print "\n"

num = range(100)
degreedf = pd.DataFrame()
```

```
62.
63. # Calculate number of nodes with n degrees
64. degree = []
65. for n in num:
66.     count = 0
67.     for node in catGraph.degree():
68.         if node[1] == n:
69.             count += 1
70.     degree.append(count)
71.
72.
73. # Calculate number of nodes with n in-degrees
74. in_degree = []
75. for n in num:
76.     count = 0
77.     for node in catGraph.in_degree():
78.         if node[1] == n:
79.             count += 1
80.     in_degree.append(count)
81.
82.
83. # Calculate number of nodes with n out-degrees
84. out_degree = []
85. for n in num:
86.     count = 0
87.     for node in catGraph.out_degree():
88.         if node[1] == n:
89.             count += 1
90.     out_degree.append(count)
91.
92.
93. # Store data about node degrees in a Dataframe and save in CSV file
94. degreedf = pd.DataFrame({ "Degree": num,
95.                           "Degree nodes": degree,
96.                           "In-Degree nodes": in_degree,
97.                           "Out-Degree nodes": out_degree})
98. export_csv = degreedf.to_csv(r"CSV\\" + "degree.csv", index = None, header = True)
99.
100.
101.       #Plot global network of government cats, followers and friends and save as pict
    ure file
102.       nx.draw(catGraph)
103.       plt.show()
104.       plt.savefig("Pictures\\" + "catNetwork.png")
105.       plt.close()
106.
107.
108.       important_degree = []
109.       important_indegree = []
110.       important_outdegree = []
111.       print "\n","most important users by degree"
112.       for node in catGraph.degree():
113.           if node[1] == 4 or node[1] == 5 or node[1] == 6:
114.               important_degree.append(node[0])
115.               print node[0]
116.       print "\n","most important users by in-degree"
117.       for node in catGraph.in_degree():
118.           if node[1] == 2 or node[1] == 3 or node[1] == 4:
119.               important_indegree.append(node[0])
120.               print node[0]
121.       print "\n","most important users by out-degree"
122.       for node in catGraph.out_degree():
123.           if node[1] == 4 or node[1] == 5 or node[1] == 6 or node[1] == 8 or node[1]
    == 19 or node[1] == 30:
124.               important_outdegree.append(node[0])
125.               print node[0]
```

```
126.
127.       print "\n","most important users"
128.       important_common = important_indegree and important_outdegree
129.       for node in important_common:
130.           print node
131.       file = "CSV\\" + "importantNodes.txt"
132.       with open(file,"w") as f:
133.           json.dump(important_common,f)
```

AnalyseFollowers2.py

```
1.  import tweepy
2.  import json
3.  import matplotlib.pyplot as plt
4.  import pandas as pd
5.  import time
6.
7.  tweepy_consumer_key = "cr9wJjmmT2CcGXqjsx7aWvsbA"
8.  tweepy_consumer_secret = "QAjnpYxS59MEil5IEpJJzGP0WqDajklMcXxcxiaJLh9doBuQ6g"
9.  tweepy_access_token = "897909419996041218-z27A8n2L6UcmUPuRbZhQBcpAAEk1Jxj"
10. tweepy_access_token_secret = "iOVhZJSMxV5XPwlePFNwqiPYUEj8QOFCXOAFcM50Q9dDC"
11.
12. auth = tweepy.OAuthHandler(tweepy_consumer_key, tweepy_consumer_secret)
13. auth.set_access_token(tweepy_access_token, tweepy_access_token_secret)
14. api = tweepy.API(auth,wait_on_rate_limit_notify=True)
15.
16.
17. # Reading list of important followers
18.
19. print "Reading list of important followers"
20. print ("\n")
21. file = "CSV\\importantNodes.txt"
22. with open(file) as f:
23.     data = json.loads(f.read())
24.
25. df = pd.DataFrame(columns = ["Followers", "Friends", "Tweets", "Likes", "Created", "Loc
    ation"])
26.
27.
28. for node in data:
29.     results = api.get_user(id=node)
30.     df = df.append( {"Screen name": results.screen_name,
31.                      "Followers": float(results.followers_count),
32.                      "Friends": float(results.friends_count),
33.                      "Tweets": float(results.statuses_count),
34.                      "Likes": float(results.favourites_count),
35.                      "Created": results.created_at,
36.                      "Location": results.location,
37.                      "LocationYes": 1 if results.location != "" else 0,
38.                      "Description": results.description,
39.                      "DescriptionYes": 1 if results.description != "" else 0},
40.                    ignore_index = True)
41.
42. print "Date and time of creation of account:"
43. print df["Created"]
44. print "\n"
45.
46. print "Number of followers:"
47. print df["Followers"]
48. print "\n"
49.
```

```
print "Friends:"
print df["Friends"]
print "\n"

print "Tweets:"
print df["Tweets"]
print "\n"

print "Likes:"
print df["Likes"]
print "\n"

df["Created"] = pd.to_numeric(df["Created"])

columns = ["Followers","Friends","Tweets","Likes","Created","LocationYes","DescriptionYes"]

print df.corr()
export_csv = df.corr().to_csv(r"CSV\\correlation.csv", index = columns, header = True)


fakedf = pd.DataFrame()
fakedf = fakedf.append(df["Screen name"],ignore_index=True)
fakedf = fakedf.append(df["Followers"] / df["Tweets"],ignore_index=True)
fakedf = fakedf.append(df["Friends"] / df["Followers"],ignore_index=True)
fakedf = fakedf.append(df["DescriptionYes"],ignore_index=True)
currentTime = time.time()
currentTime_Days = (((currentTime / 60) / 60) / 24)
created_Days = (((df["Created"] / 60) / 60) / 24)
duration = created_Days - currentTime
fakedf = fakedf.append(df["Tweets"] / duration, ignore_index=True)
fakedf = fakedf.transpose()
print fakedf
export_csv = fakedf.to_csv(r"CSV\\fakeusers.csv", index = ["Follower to Tweet","Friend to Follower","Description","Tweet to Duration"], header = True)
```

AnalyseTweets.py

```
from __future__ import division
import json
import pandas as pd
import numpy as np
import matplotlib
import matplotlib.pyplot as plt
from textblob import TextBlob

headings = ["@DiploMog",
            "@HMCabinetCat",
            "@Number10cat",
            "@PalmerstonFOCat",
            "@PalmerstonCat",
            "@TreasuryMog",
            "@HMTreasuryCat"]

print ("Reading list of tweets...")
rows = {}
for cat in headings:
    file = "Datasets\\" + cat + "tweets.txt"
    with open(file) as f:
```

```
        data = json.loads(f.read())
    rows[cat] = data

print ("Reading stopwords...")
# https://en.wikipedia.org/wiki/Stop_words,
# https://www.textfixer.com/tutorials/common-english-words.txt
file = "Datasets\\stopwords.txt"
with open(file) as f:
    stopwords = f.read()
stopwords = stopwords.split(",")

for cat in headings:
    print "Calculating textblobs for",cat
    cat1 = cat[1:len(cat)]
    vars()[cat1] = {}
    for i in range(len(rows[cat])):
        tweet = rows[cat][i]
        blob = TextBlob(tweet)
        words = blob.ngrams(n=1)
        twograms = blob.ngrams(n=2)
        threegrams = blob.ngrams(n=3)
        for i in range(len(words)):
            if words[i][0] in stopwords or words[i][0][0:4] == "t.co":
                continue
            elif words[i][0][0:4] == "http" or words[i][0][0:5] == "https":
                continue
            elif words[i][0][0] == "@" or words[i][0] == "????":
                continue
            elif words[i][0] in vars()[cat1]:
                vars()[cat1][words[i][0]] += 1
            else:
                vars()[cat1][words[i][0]] = 1
        for i in range(len(twograms)):
            if twograms[i][0] in stopwords or twograms[i][0][0:4] == "t.co":
                continue
            elif twograms[i][0][0:4] == "http" or twograms[i][0][0:5] == "https":
                continue
            elif twograms[i][0] in vars()[cat1]:
                vars()[cat1][twograms[i][0]] += 1
            else:
                vars()[cat1][twograms[i][0]] = 1
        for i in range(len(threegrams)):
            if threegrams[i][0] in stopwords or threegrams[i][0][0:4] == "t.co":
                continue
            elif threegrams[i][0][0:4] == "http" or threegrams[i][0][0:5] == "https":
                continue
            elif threegrams[i][0] in vars()[cat1]:
                vars()[cat1][threegrams[i][0]] += 1
            else:
                vars()[cat1][threegrams[i][0]] = 1

for cat in headings:
    print "Visualising textblobs for",cat
    cat1 = cat[1:len(cat)]
    vars()[cat1 + "highvol"] = {}
    objects = []
    for i in vars()[cat1]:
        if vars()[cat1][i] > 15:
            objects.append(i)
            y_pos = np.arange(len(objects))
            vars()[cat1 + "highvol"][i] = vars()[cat1][i]
    plt.barh(y_pos, vars()[cat1 + "highvol"].values())
    plt.yticks(y_pos,objects,fontsize = 7)
    plt.title("High-frequency ngrams in " + cat1 + "'s tweets")
    plt.xlabel("Number of occurances")
    plt.ylabel("Ngrams")
```

```
    plt.savefig("Pictures\\tweets\\"+cat1+"_wordcount.png")
    plt.close()
```

AnalyseTweets2.py

```
import networkx as nx
import matplotlib.pyplot as plt

headings = ["@DiploMog",
            "@HMCabinetCat",
            "@Number10cat",
            "@PalmerstonFOCat",
            "@PalmerstonCat",
            "@TreasuryMog",
            "@HMTreasuryCat"]


catGraph = nx.DiGraph()
catGraph.add_node("@DiploMog")
catGraph.add_node("@HMCabinetCat")
catGraph.add_node("@HMTreasuryCat")
catGraph.add_node("@Number10cat")
catGraph.add_node("@PalmerstonCat")
catGraph.add_node("@PalmerstonFOCat")
catGraph.add_node("@TreasuryMog")

catGraph.add_node("@foreignoffice")
catGraph.add_node("@LawrenceDipCat")
catGraph.add_node("@SMcDonaldFCO")
catGraph.add_node("@BDCH")
catGraph.add_node("@Battersea")
catGraph.add_node("@PoliticalPics")
catGraph.add_node("@justin_ng")
catGraph.add_node("@Stoughton_p")
catGraph.add_node("@Number11Dog")
catGraph.add_node("@Zilla_Mon")
catGraph.add_node("@YourCatmagazine")

catGraph.add_edge("@DiploMog","@DiploMog")
catGraph.add_edge("@DiploMog","@foreignoffice")
catGraph.add_edge("@DiploMog","@LawrenceDipCat")
catGraph.add_edge("@DiploMog","@SMcDonaldFCO")
catGraph.add_edge("@DiploMog","@BDCH")
catGraph.add_edge("@DiploMog","@Battersea")
catGraph.add_edge("@HMCabinetCat", "@TreasuryMog")
catGraph.add_edge("@HMCabinetCat", "@HMTreasuryCat")
catGraph.add_edge("@HMCabinetCat", "@DiploMog")
catGraph.add_edge("@HMCabinetCat", "@Number10cat")
catGraph.add_edge("@Number10cat", "@PoliticalPics")
catGraph.add_edge("@Number10cat", "@justin_ng")
catGraph.add_edge("@HMTreasuryCat","@Number10cat")
catGraph.add_edge("@PalmerstonCat","@Number10cat")
catGraph.add_edge("@PalmerstonCat", "@TreasuryMog")
catGraph.add_edge("@PalmerstonCat","@foreignoffice")
catGraph.add_edge("@PalmerstonCat","@YourCatmagazine")
catGraph.add_edge("@PalmerstonCat","@Battersea")
catGraph.add_edge("@PalmerstonFOCat","@Number10cat")
catGraph.add_edge("@PalmerstonFOCat","@PoliticalPics")
catGraph.add_edge("@PalmerstonFOCat","@Stoughton_p")
catGraph.add_edge("@TreasuryMog","@HMCabinetCat")
```

```
catGraph.add_edge("@TreasuryMog","@DiploMog")
catGraph.add_edge("@TreasuryMog","@Number10cat")
catGraph.add_edge("@TreasuryMog","@Number11Dog")
catGraph.add_edge("@TreasuryMog","@Zilla_Mon")


print "Number of users in global network:", catGraph.number_of_nodes()
print "Number of connections in global netwwork:", catGraph.number_of_edges()
print "\n"


nx.draw(catGraph,with_labels=True)
plt.show()
plt.savefig("Pictures\\" + "catNetwork2.png")
plt.close()
```

AnalyseLocations.py

```
from __future__ import division
import math
import json
import pandas as pd
import numpy as np
import matplotlib
import matplotlib.pyplot as plt
from textblob import TextBlob

headings = ["@DiploMog",
            "@HMCabinetCat",
            "@Number10cat",
            "@PalmerstonFOCat",
            "@PalmerstonCat",
            "@TreasuryMog",
            "@HMTreasuryCat"]

print ("Reading list of locations...")
rows = {}
for cat in headings:
    file = "Datasets\\" + cat + "locations.txt"
    with open(file) as f:
        data = json.loads(f.read())
    rows[cat] = data

print ("Reading stopwords...")
# https://en.wikipedia.org/wiki/Stop_words,
# https://www.textfixer.com/tutorials/common-english-words.txt
file = "Datasets\\stopwords.txt"
with open(file) as f:
    stopwords = f.read()
stopwords = stopwords.split(",")

for cat in headings:
    print "Calculating textblobs for",cat
    cat1 = cat[1:len(cat)]
    vars()[cat1] = {}
    for i in range(len(rows[cat])):
        location = rows[cat][i]
        blob = TextBlob(location)
        words = blob.ngrams(n=1)
        twograms = blob.ngrams(n=2)
        threegrams = blob.ngrams(n=3)
```

```
        for i in range(len(words)):
            if words[i][0] in vars()[cat1]:
                vars()[cat1][words[i][0]] += 1
            else:
                vars()[cat1][words[i][0]] = 1
        for i in range(len(twograms)):
            if twograms[i][0] in vars()[cat1]:
                vars()[cat1][twograms[i][0]] += 1
            else:
                vars()[cat1][twograms[i][0]] = 1
        for i in range(len(threegrams)):
            if threegrams[i][0] in vars()[cat1]:
                vars()[cat1][threegrams[i][0]] += 1
            else:
                vars()[cat1][threegrams[i][0]] = 1

for cat in headings:
    print "\n"
    print "Visualising textblobs for",cat
    cat1 = cat[1:len(cat)]
    vars()[cat1 + "highvol"] = {}
    objects = []
    xsum = 0
    for i in vars()[cat1]:
        xsum = xsum + vars()[cat1][i]
    average = xsum / len(vars()[cat1])
    print "Average number of occurances of an ngram", average
    for i in vars()[cat1]:
        if vars()[cat1][i] > math.ceil(average):
            objects.append(i)
            y_pos = np.arange(len(objects))
            vars()[cat1 + "highvol"][i] = vars()[cat1][i]
    plt.barh(y_pos, vars()[cat1 + "highvol"].values())
    plt.yticks(y_pos,objects,fontsize = 7)
    plt.title("High-frequency ngrams in " + cat1 + "'s followers' locations")
    plt.xlabel("Number of occurances")
    plt.ylabel("Ngrams")
    plt.savefig("Pictures\\locations\\"+cat1+"_wordcount.png")
    plt.close()
```

AnalyseDescript.py

```
from __future__ import division
import math
import json
import pandas as pd
import numpy as np
import matplotlib
import matplotlib.pyplot as plt
from textblob import TextBlob

headings = ["@DiploMog",
            "@HMCabinetCat",
            "@Number10cat",
            "@PalmerstonFOCat",
            "@PalmerstonCat",
            "@TreasuryMog",
            "@HMTreasuryCat"]
```

```
17.
18. print ("Reading list of descriptions...")
19. rows = {}
20. for cat in headings:
21.     file = "Datasets\\" + cat + "descriptions.txt"
22.     with open(file) as f:
23.         data = json.loads(f.read())
24.     rows[cat] = data
25.
26. print ("Reading stopwords...")
27. # https://en.wikipedia.org/wiki/Stop_words,
28. # https://www.textfixer.com/tutorials/common-english-words.txt
29. file = "Datasets\\stopwords.txt"
30. with open(file) as f:
31.     stopwords = f.read()
32. stopwords = stopwords.split(",")
33.
34. for cat in headings:
35.     print "Calculating textblobs for",cat
36.     cat1 = cat[1:len(cat)]
37.     vars()[cat1] = {}
38.     for i in range(len(rows[cat])):
39.         description = rows[cat][i]
40.         blob = TextBlob(description)
41.         words = blob.ngrams(n=1)
42.         twograms = blob.ngrams(n=2)
43.         threegrams = blob.ngrams(n=3)
44.         for i in range(len(words)):
45.             if words[i][0] in stopwords:
46.                 continue
47.             elif words[i][0] in vars()[cat1]:
48.                 vars()[cat1][words[i][0]] += 1
49.             else:
50.                 vars()[cat1][words[i][0]] = 1
51.         for i in range(len(twograms)):
52.             if twograms[i][0] in stopwords:
53.                 continue
54.             elif twograms[i][0] in vars()[cat1]:
55.                 vars()[cat1][twograms[i][0]] += 1
56.             else:
57.                 vars()[cat1][twograms[i][0]] = 1
58.         for i in range(len(threegrams)):
59.             if threegrams[i][0] in stopwords:
60.                 continue
61.             elif threegrams[i][0] in vars()[cat1]:
62.                 vars()[cat1][threegrams[i][0]] += 1
63.             else:
64.                 vars()[cat1][threegrams[i][0]] = 1
65.
66. for cat in headings:
67.     print "\n"
68.     print "Visualising textblobs for",cat
69.     cat1 = cat[1:len(cat)]
70.     vars()[cat1 + "highvol"] = {}
71.     objects = []
72.     xsum = 0
73.     for i in vars()[cat1]:
74.         xsum = xsum + vars()[cat1][i]
75.     average = xsum / len(vars()[cat1])
76.     print "Average number of occurances of an ngram", average
77.     for i in vars()[cat1]:
78.         if vars()[cat1][i] > 10:
79.             objects.append(i)
80.             y_pos = np.arange(len(objects))
81.             vars()[cat1 + "highvol"][i] = vars()[cat1][i]
82.     plt.barh(y_pos, vars()[cat1 + "highvol"].values())
```

```
83.     plt.yticks(y_pos,objects,fontsize = 7)
84.     plt.title("High-frequency ngrams in " + cat1 + "'s descriptions")
85.     plt.xlabel("Number of occurances")
86.     plt.ylabel("Ngrams")
87.     plt.savefig("Pictures\\descriptions\\"+cat1+"_wordcount.png")
88.     plt.close()
```