

# The Research and Development Of iProwlr

A web application to unify internet users online geographical data to  
create a visual resource for security and privacy education

# CONTENTS

<b>1.0</b>	<b>Executive Summary .....</b>	<b>1</b>
<b>2.0</b>	<b>Market Research .....</b>	<b>1</b>
<b>3.0</b>	<b>Elevator Pitch .....</b>	<b>2</b>
<b>4.0</b>	<b>Market Analysis .....</b>	<b>3</b>
<b>5.0</b>	<b>S.W.O.T Analysis .....</b>	<b>3</b>
5.1	Strengths .....	3
5.2	Weaknesses .....	4
5.3	Opportunities .....	4
5.4	Threats .....	4
<b>6.0</b>	<b>Marketing Strategy .....</b>	<b>4</b>
<b>7.0</b>	<b>Development Process Technologies used .....</b>	<b>5</b>
7.1	Extracting Geographical data .....	5
7.1.1	Untappd .....	5
7.1.2	Strava .....	7
7.1.3	Flickr .....	8
7.2	Machine Learning .....	10
7.2.1	Technology Used .....	10
7.2.2	Training Experiment .....	10
7.2.3	Predictive Model and iProwlr Integration .....	12
<b>8.0</b>	<b>Project Management .....</b>	<b>12</b>
8.1	Gantt Chart .....	13
<b>11.0</b>	<b>Evaluation .....</b>	<b>14</b>
<b>12.0</b>	<b>References .....</b>	<b>15</b>

## **1.0 Executive summary**

Social media is a huge and ever-growing part of our everyday life (Smith & Anderson, 2018) and is all but unavoidable for people. It allows us to communicate with one another, regardless of geographic distances faster and more user-friendly than in the past. It has revolutionized how people digest news (Matsa & Shearer, 2018; Anderson & Caumont, 2014) and information by making it integral with our social lives.

Naturally, with such a huge outreach and user base, social media can come with some inherent risks related to online security and privacy. Social media platforms allow users to post geo-tags in the form of "location check-ins", or geotagged photographs (Gross & Hanna, 2010). These geotags provide a method to update a user's social media followers of recent travels/holidays, or even gamifying visiting places (Sotamaa, 2002), but as we will discuss later, they can carry some considerable privacy concerns. Some websites such as Strava are specifically set up for sharing your exercise information in the form of running trails or cycling routes.

In November 2017 Strava released a global heatmap of every single activity ever uploaded to their platform (Strava, 2018), covering over 3.3 Trillion individual GPS points. But with over 17 Billion miles of routes logged on the heatmap, some problems arose. Soldiers in undisclosed military bases had been logging their exercise routes (Hern, 2018) and military analysts realized that the heatmap had exposed the location of the bases. An obvious security flaw, this raised some important questions. Just how much information do we put out there for the world to see? How can it be used in possible malicious situations? A fundamental lack of understanding by the general population on how their geographical information is stored and how it can be used, gave us the idea to explore the problem in more detail.

We set out to create a website, which can be used as an educational tool, to show just how effectively one's public geographic information can be utilized to track their physical location. Using nothing but the information the user has posted online over a range of social media platforms, we want to give the users a deeper understanding of how much sensitive information they are posting online and how, in the wrong hands, it can be used to track your location.

## **2.0 Market Research**

This idea to accumulate a user's online information and display it in a understandable format. There are several preexisting websites and services which develop on the idea of educating people of their internet habits. Often this involves searching through their online instances and informing them of possible security flaws.

Pipl.com is an information services company with "the world's largest people search engine" (Pipl, 2019). Pipl allows the user to search both offline and online for all instances of a subject in question. A tool on this level is developed mainly for business usage, as such information is beneficial for marketing. The aggregation of an internet users' online information as a tool is what makes pipl.com a competing service to iProwlr. Although, the fact that our service is free and is created for the user to find how their own personal information (specifically geographic data) can be used maliciously is what differs pipl to iProwlr.

Wink, much like iProwlr, is a service that consolidates geographic data publicly available on social media, Facebook specifically (iTools, 2019). Wink uses this publicly available geographic data to search for other Facebook users living in the same area, and along with other search criteria, can refine down the search to people you will possibly know. This tool highlights the easy availability of geographic data for Facebook users all around the world. iProwlr has been developed to make this surprising fact visible and easily understandable for the user. All with the objective of raising the awareness of these concerns with online privacy.

### **3.0 Elevator Pitch**

iProwlr is a web application which focuses exclusively on the geographical data a user makes publicly available. It utilizes several platforms including, but not restricted to, Strava, Untapped and Flickr . iProwlr scans a users' account and extracts any geographical data. Using this data, iProwlr then generates a heatmap overlaid onto a map of the world which can allow a user to see exactly where their movements have been recorded on a global and local scale. The heatmap also allows the user to see trends in their logged movements. Trends are educational for the user, as commonly it is not realized how much information one makes available for public viewing. Furthermore, public information is free to be viewed (or abused) by a 3<sup>rd</sup> party. As well as the heatmap, iProwlr also uses machine learning to make a prediction of a possible time and location at which the user can be found. Such an educational tool we hope will prove to change and modernize the way of which the average internet user will share their information. All in order to boost their privacy and make them less susceptible to threats to their security.

### **4.0 Market Analysis**

iProwlr targets the average internet consumers who use the internet as a tool for communication, socialization and organization. With geographic information, iProwlr aims to make the user more educated of their information availability. Internet users who are perhaps more concerned or curious about online privacy and security concerns will find iProwlr a useful tool. While data breaches keep making the news (Irwin, 2019), many people still happily share valuable data online (Spiegel, 2017). But without a simple, user friendly tool to unify all their online data across numerous platforms, it can be difficult for one to undertake. iProwlr is poised to make this task simpler and more easily digestible for the average consumer. Another potential market iProwlr could be of interest to is the government and law enforcement as it provides a relatively reliable tracking method without breaking any privacy laws.

### **5.0 S.W.O.T Analysis**

#### ***5.1 Strengths:***

Throughout the development stages of iProwlr, it provided practice in a wide array of new skillsets and computing disciplines. After an initial plan of utilizing the public API's fell through due to issues with permissions, we realized that we would have to use a range of different methods for data extraction. All depending on the nature of the website and the format of the geographic data required. This allowed us to expand our knowledge on the use of PHP, Python3 scripting and Python3 libraries. Each new website we wanted to extract data from rewarded us with a plethora of new teachings and quickly we had become proficient at extracting data from websites through a range of methods. On top of this, machine learning was new to us all, and its vital integration into iProwlr meant that we had to add the discipline to our arsenal.

## **5.2 Weaknesses:**

Due to the invasive nature of the methods used to gain the required information from websites some difficulties arose. Strava data protection requires you to be logged in on a paid premium account to be able to download another users GPS data from their activities. Because of this, we developed iProwlr to allow users to manipulate their own data and use machine learning to show a prediction of where they can be found. This is still an important educational tool and fits in iProwlr's agenda. Another weakness iProwlr has is speed. Because of the large number of requests iProwlr makes to a server in a short period of time, we encountered the issue of servers blocking our requests after a limit has been reached within a set time frame. This is server security feature meant that we had to hard code in small delays in order to not appear as a bot and prevent request blocking. These small delays add up and slow down the process considerably. For example, our script that actually predicts the latitude and longitude co-ordinates from the csv data, takes approximately 40 seconds to fully complete a HTTP GET/POST request to our web service.

## **5.3 Opportunities:**

With the subject of cybersecurity and online privacy growing on all forms of media consumption (TV, Radio, Newspapers, Online, etc.), there is a correlating rise in the number of people who are more curious about their online privacy (Byer, 2018). iProwlr serves as an opportunity to provide for this growing market of proactive internet users. This project is also widely scalable to include more online platforms and services as seen appropriate as time progresses. To tackle the growing consumer market (section 4.0) with a free, user friendly and relatively comprehensive tool is a valuable opportunity for iProwlr.

## **5.4 Threats:**

Due to the methods we used to automate the navigating of webpages, if the websites change their layout or start renaming buttons for example, the script we have developed will not be able to navigate and work as it should. Therefore, it is very important for us to keep on top of website updates to ensure the service remains usable. Also, there is the threat of websites changing their policies towards dealing with bots. If servers start further limiting the limit of request speeds or integrates more advanced methods against to prevent automatic logging in and website manipulation, iProwlr might no longer operate within the boundaries of what the server allows. Therefore, it is important to keep updated regarding changes in server policies.

## **6.0 Marketing Strategy**

For iProwlr to reach as much of our target audience (section 2.0) as possible, it must be marketed. Our target audience (*section 2.0*) are the average internet users. People who are perhaps more concerned about online privacy due to the subjects increased media coverage over recent years. It would be redundant to advertise with scientific and technical reports, as the users we are targeting are not in the demographic who will regularly be consuming such channels of information. The most sensible and effective method of delivery would be through the regular channels of which the average internet user consumes information. As these people will most likely be proactive in doing their own basic research on the subject, it would be wise to advertise on pages related to the wider subject of internet privacy and protecting one's online identity. This could include video or banner advertisements on YouTube videos related to the subject, or banners on relevant news articles. Being a proactive target audience help us to be more specific on the pages, articles and media channels we should market our product on.

## 7.0 Development Process & Technologies Used

### 7.1 Extracting Geographical Data:

Due to the nature of iProwlr compatible websites, it was required for us to develop a method to access, locate and extract necessary data. The raw, extracted data we were left with was often in an undesirable format, therefore, we were required to develop an extra formatting step to ensure that all the geographical data is presented in an identical manner. This formatting allows the data to be inputted into our machine learning algorithm successfully.

#### 7.1.1 Untappd:

Untappd is a geosocial networking service and mobile phone application that allows users to check in as they drink beers and share these check-ins and their locations with their friends (Miami new Times, 2012). We decided to implement Untappd compatibility with iProwlr as it allows us to explore the concept of geosocial web platforms and the possible vulnerabilities associated with them. The Geographical data we require from Untappd is in the URL of a 'check in' instance.

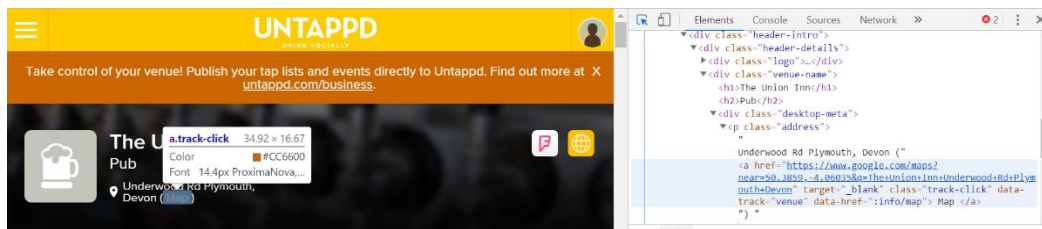


Figure 1

For us to access this data and extract it, a Python script was developed with the libraries: Selenium and BeautifulSoup:

Firstly, the Python library 'Selenium' is used to open the website [www.untappd.com/login](https://untappd.com/login) in an instance of Mozilla Firefox. Selenium is a tool for automating web applications and is an effective way for us to automate the login and navigation process of Untappd

Using Selenium again, we developed a working automated login process in which allows us to login to untapped using a fake 'place holder' account. The reason for using a place holder account instead of logging in with the users specific account is because it allows us to explore what geographical information is made available for the public to access rather than what private information the user has personal access too.

Figure 2

```
13 usernameText = 'textuni'
14 passwordText = 'password'
15
16 user = 'Crosswaite'
17
18 url = 'https://untappd.com/'
19 driver = webdriver.Firefox()
20
21 driver.get(url + 'login')
22
23
24 cj = cookiecutter.CookieJar()
25 br = mechanize.Browser()
26 br.set_handle_robots(False) # ignore robots
27 br.set_handle_refresh(False) # can sometimes hang without this
28 br.addheaders = [('User-agent', 'Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.9.0.1) Gecko/2008071615 Fedora/3.0.1-1.fc9 Firefox/3.0.1')]
29 br.set_cookiejar(cj)
30
31 username = driver.find_element_by_id("username")
32 password = driver.find_element_by_id("password")
33
34 username.send_keys(usernameText)
35 password.send_keys(passwordText)
36
37 driver.find_element_by_class_name("submit-btn").click()
```

The next step was to script a method to navigate to the public page of the user you are wishing to access information from. To do this, Selenium is used again to navigate to the users public page (e.g. [www.untappd.com/user/JohnSmith](http://www.untappd.com/user/JohnSmith)). The previous step to login to a ‘place holder’ account is necessary to be able to access this information.

```
39 user = url + 'user/' + user
40 driver.get(user)
```

Figure 4

```
42 #driver.find
43 containerCount = 0
44
45 while True:
46     pageSoup = soup(driver.page_source, "html.parser")
47     #finds the divs that hold the different activities
48     mainS = pageSoup.find("div", {"id" : "main-stream"})
49     containers = mainS.find_all("div", {"class" : "item"})
50     if len(containers) == containerCount:
51         break
52     containerCount = len(containers)
53     time.sleep(5)
54     try:
55         element = driver.find_element_by_id("branch-banner-iframe")
56         driver.execute_script("arguments[0].style.visibility='hidden'", element)
57     except:
58         print("")
59
60     try:
61         driver.find_element_by_link_text('Show More').click()
62     except:
63         sleep(random.randint(2,3))
64         driver.find_element_by_link_text('Show More').click()
65     time.sleep(random.randint(3,4))
66     print(containerCount)
```

Figure 3

```
pattern = r"<a class=\"user\" (href)=\"\\/user\\/.*?\">.*?(</a>) is drinking (a|an) (<a \\1=\\/b\\..*?\">.*?\\2 by \\4=\\/.*?\">.*?\\2 at \\4=\\/v\\..*?\">.*?\\2\"
if(re.match(pattern, text)):
    geoContainers.append(i)
```

Figure 5

To boost accuracy of the prediction generated by the machine learning algorithm, it is necessary to utilize as many ‘check in’ instances as possible from a user’s account. We used Selenium again and made a section of python script which scrolls to the bottom of the page and clicks ‘show more’, whilst adding all html containers to a list this repeats until it is at the bottom of the feed.

To prevent the manipulation of unnecessary check in instances without an associated location, a RegEx was written to identify check in instances which have a linked location. This was done in the source code (fig 4) to make it more accurate as lazy approached could have been a false positive by the use of the string “at “.

```
with open('untap.csv', 'w', newline='') as csvFile:
    writer = csv.writer(csvFile, delimiter = ',', quotechar='\"', quoting=csv.QUOTE_MINIMAL)

    writer.writerow(['Latitude', 'Longitude', 'Timestamp'])
    for i in containers:
        checkinHtml = ('https://untappd.com/user/Crosswaite' + '/checkin/' + i['data-checkin-id'])
        driver.get(checkinHtml)
        pageSoup = soup(driver.page_source, 'html.parser')
        location = pageSoup.find("p", {"class" : "location"}).a['href']
        dateString = pageSoup.find("p", {"class" : "time"})["data-gregtime"]
        structTime = time.strptime(dateString, "%a, %d %b %Y %H:%M:%S %z")

        if(location not in visitedVenues):
            driver.get(url + location)
            pageSoup = soup(driver.page_source, 'html.parser')
            continue_link = driver.find_element_by_link_text('Map').click()

            curUrl = driver.current_url
            lat = 1
            lon = 1
            visitedVenues[location] = lat + ',' + lon
            time.sleep(random.randint(5,8))
        else:
            latLon = visitedVenues[location]
            index = latLon.find(",")
            lat = latLon[:index]
            lon = latLon[index+1:]
            print(time.mktime(structTime))
            print(lat)
            print(lon)
            writer.writerow([lat, lon, time.mktime(structTime)])

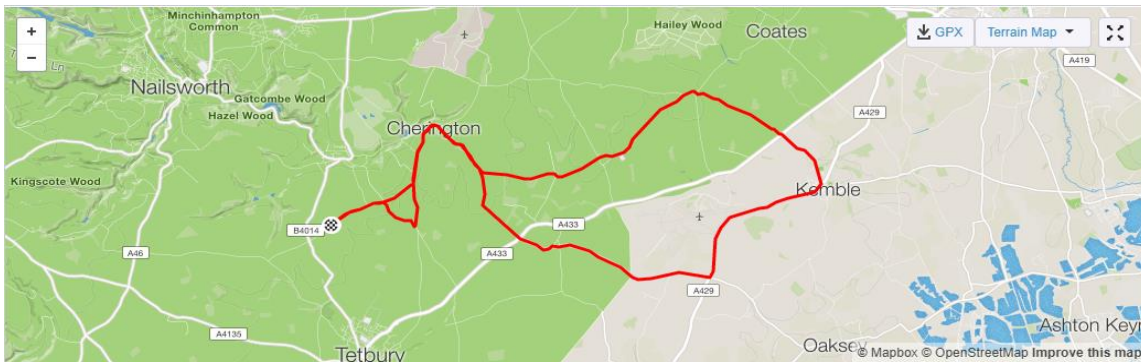
driver.close()
```

Figure 6

Next, the script aggregates all the links of the check in instances and works through the links in each container. If the location doesn't exist in the python dictionary, it opens the google link in a new tab and switches to it where the shorthand URL is replaced by a more accurate one. It then takes that link and closes the window. The selenium URL is then used to write the longitude and latitude to the dictionary with the associative key of the venue name. Then the timestamp, latitude and longitude are written to a list which at the end is written to a .csv file.

### 7.1.2 Strava:

Strava is a website with an associated phone application which allows athletes to use their phone as a GPS tracker to record and upload their activity routes to their profile. We chose to include Strava compatibility in iProwr because of the regularity and reliability of the data the users make public. After the issues exposing US military bases (section 1.0), we instantly recognised the powerful opportunities that a platform like Strava could offer.



*Figure 7 – an example of a mapped activity uploaded to an athletes profile*

In order for us to extract the necessary geographical data from Strava, a Python script was developed which uses the libraries 'Selenium', 'Mechanize' and 'BeautifulSoup' to navigate to and then download the .gpx data. The data contains the crucial latitude and longitude with each uploaded athletes activity (as can be seen by the GPX download button in the top right corner of figure 7).

Selenium was used to navigate to the Strava login page and automatically fill out and submit the users login details. During the development we wanted to use a 'dummy' account to login and then be able to download any users public activity routes. However, downloading the .GPX from a different users account requires you to pay for a premium account.

Once logged into the users account, you are only able to access the last three journeys someone has done on their profile, but if you follow that person and no one else, your friends feed is all of their recorded journeys. Selenium then keeps scrolling down to load up all of these html containers. BeautifulSoup is used to parse the html and list the start time and the activity link.

Mechanize is then used instead of selenium for easier and more controllable downloading of the gpx files. It logs in with the same details and works through the list of activities, downloading the file, then reading the .gpx file using the gpxpy library. Then this was read in line by line to have a list of dictionaries holding the full journey, before deleting the file. This list of locations now needed timestamps added as you don't get the time if downloading someone else's journey. Strava takes a read on average every 3.2 seconds so we set the start time to UNIX timestamp for the start reading and then add 3.2 seconds for each.



### 7.1.3 Flickr:

Flickr is an image hosting social media site for photographers to share their work to a large audience. We chose to integrate Flickr due to the widespread use of Exchangeable Image File Format (EXIF), the location coordinates and timestamp are saved in the EXIF data of an uploaded photograph. EXIF is embedded metadata that is attached with JPEG's and is useful for storing useful information such as the camera setup and setting, and location, but if it is forgotten about or not known by a photographer or just doesn't care, it can pose as a security threat. Kaspersky published an article highlighting this privacy concern. They wrote about how it is an often overlooked feature of JPEGs and gave an example of how it was used to catch a fugitive on the run from the law when VICE news posted a photo of one of their journalists with the fugitive and forgot to remove the EXIF data. The fugitive was subsequently tracked down using the locational data stored the photograph (Kuksov, 2016).

The Selenium library was used again to open up an instance of the Firefox web browser and navigate to the Flickr login page. Once this is loaded, there is further Selenium script to fill out the login fields and submit the form by automating button presses (figure 8).

Flickr displays the EXIF data alongside the photos on the platform. Therefore, BeautifulSoup was used to parse the html and search for the element which presents the photographs EXIF data. The vendor has done most of the difficult part in finding the EXIF data in this case.

```
14 loginUrl = 'https://identity.flickr.com/'
15 url = 'https://www.flickr.com'
16 photosUrl = url + '/photos/'
17 driver = webdriver.Firefox()
18
19 usernameText = "simela@the-first.email"
20 passwordText = "passwordpass"
21 userToRead = "92895616@N06"
22
23 driver.get(loginUrl)
24 time.sleep(1)
25
26 username = driver.find_element_by_id("login-email")
27 username.send_keys(usernameText)
28 driver.find_element_by_class_name("mt-5").click()
29
30 time.sleep(2)
31
32 driver.find_elements_by_class_name("orko-button-primary")
33 driver.find_element_by_id("login-signin").click()
34
35 time.sleep(2)
36
37 password = driver.find_element_by_id("login-passwd")
38 password.send_keys(passwordText)
39
40 driver.find_element_by_id("login-signin").click()
```

Figure 8

```
61 driver.find_element_by_class_name("show-extended-exif").click()
62 time.sleep(2)
63 pageSoup = soup(driver.page_source, "html.parser")
64
65 exifData = str(pageSoup.find("div", {"class": "exif-column-1"}))
66 startI = exifData.find("GPS Time Stamp - ") + 50
67 #12:15:54
68 picTime = exifData[startI:startI+8]
69
```

Figure 9

## 7.2 Machine Learning:

### 7.2.1 Technology Used:

From the conception of iProwlr as a concept, Machine Learning has been an integral system (*section 3.0*). After researching the subject, it was decided that we would use Microsoft's Azure machine Learning platform to develop and integrate a machine learning algorithm into iProwlr. Microsoft defines their Azure machine Learning Studio as a "collaborative, drag-and-drop tool you can use to build, test, and deploy predictive analytics solutions on your data" (Microsoft, 2019). Azure's ease of use, comprehensive features and ease of integration were the main deciding factors along with an integrated web service to host the machine learning. The service would be created by first creating an experiment template, then creating a web service from said template that our Python script would connect to.

### 7.2.2 Training Experiment and Predictive Model:

Firstly, for us to be able to have a functioning predictive algorithm to input data into, it was necessary to design and develop a 'training experiment' (as they are referred to in Azure). This training experiment serves to "teach" our predictive algorithm the steps it requires to complete its task. To achieve this, 2 similar training experiments were developed to build working latitude and longitude prediction algorithms (*figure 10 & figure 11*).

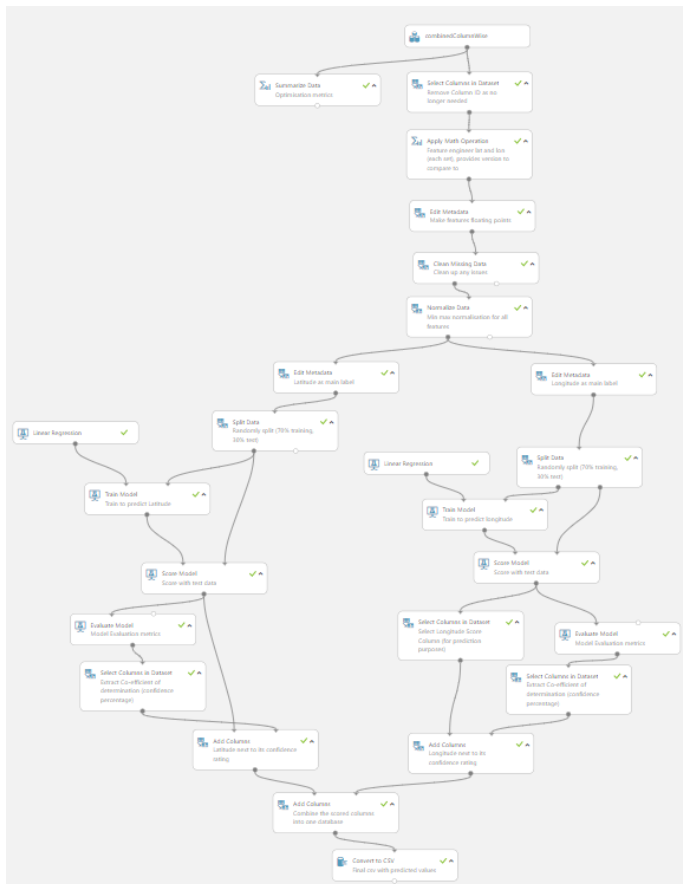


Figure 10 latitude training experiment

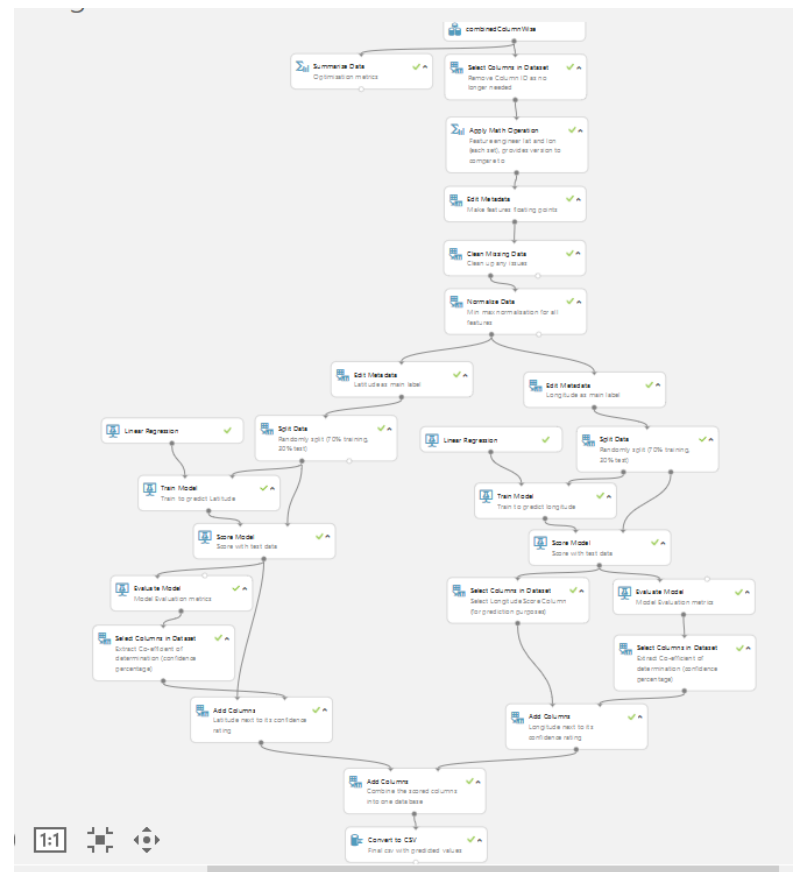


Figure 11 longitude training experiment

The geographical data (longitude, latitude, timestamp) which is used for the training experiments was collected during an extensive period of data collection in which we had to use the social media platforms in their intended manner to generate a large volume of control data. This control data is then used to teach the predictive algorithms (as described in more detail below). Although not used in the final web service, control data ensures the experiment works as intended and helps to identify any potential anomalies in the final dataset.

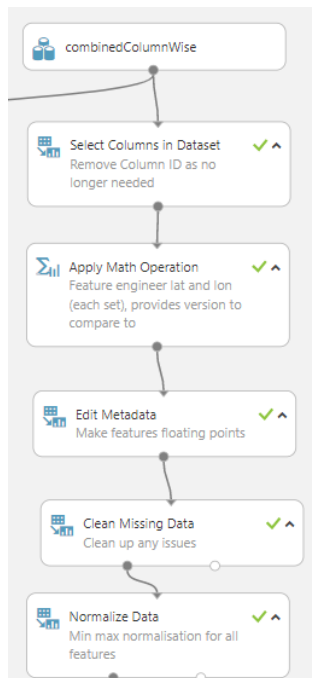


Figure 11

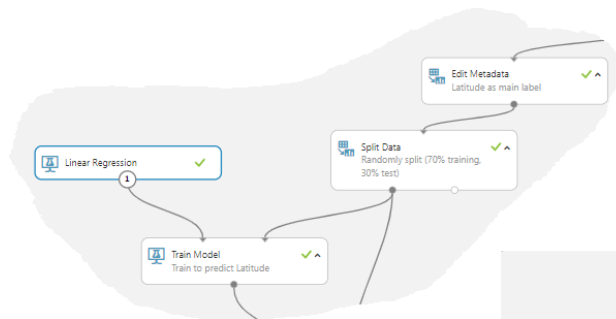


Figure 13

```

1 SELECT
2 [Scored Labels] AS LatitudePrediction,
3 [Coefficient of determination] AS Accuracy
4 FROM t1,t2;

```

Figure 15

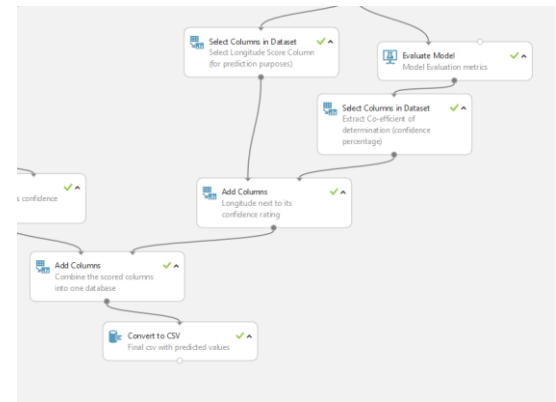


Figure 14

The first step covers the manipulation of the inputted data for the predictive steps later in the algorithm to function properly and be as accurate as possible. Firstly, the control geographical data in the format of a .csv file is simplified by removing unnecessary columns, leaving behind only the longitude, latitude and timestamps. (figure 12)

The Apply Math module squares all the individual values in the dataset by a power of 2. This is done to make a larger dataset and subsequently maximize the accuracy of the predictive algorithm. Squaring that data essentially doubles the dataset scope and grants the algorithm a wider variety of data to work with. (figure 12)

It's necessary to normalize datasets to alter the values of the numeric columns to use a common scale (Malcolm, 2019). This is important for the algorithm to model the data correctly and was important because after we applied the math's function to square the dataset it left us with too big of a gap on the scale between one half of the data and the other. I found MinMax normalization to be the most reliable due to its simple algorithm and universal applicability to all datatypes (figure 12).

Next, the dataset undergoes 2 similar, simultaneous processes to generate a predictive model for latitude and longitude. Each process splits the dataset into two sections for training and testing purposes. After researching extensively and getting advice from subject experts, we chose to split the dataset into 70% for training and 30% for testing (Malcolm, 2019). This is the ratio split that proves the most efficient for our data and is also the standard practice for such an algorithm. (figure 13)

The training data is then put through a training model using linear regression and the results are tested against the test data. A liner regression algorithm is used to establish a linear relationship between our dataset variables and allows model (once trained) to make predictions. (figure 13)

The model is then scored and evaluated to provide a 'coefficient of determination' (confidence rating) for the predicted longitude or latitude, with some concluding clean up modules setting up the data to be combined into one table (figure 14)

Finally, the generated longitude dataset and latitude dataset are recombined and, along with their respective accuracies, are the output for the experiment. After verifying that the experiment works with sample data, the next step is to

implement it as a web service and allow users to send customized datasets to the algorithm. The example csv file shown in figure 12 is now replaced with a webservice input module and a webservice output module is added to the bottom of the algorithm. Azure machine learning creates these for you, the only addition being a SQL script. The script requests the service to only pass the predicted values and accuracy back to the user (the rest is no longer useful). (figure 14 & 15).

We found that although Azure ML was very useful for our purpose and has some excellent documentation for new beginners, we had to create two separate web service's for latitude and longitude respectively. It currently doesn't support the training and evaluation of two or more features. Although this didn't change our results in the end, it did significantly increase the execution time of one of our scripts.

### 7.2.3 iProwlr Integration:

With the machine learning now as a web service, we can use the combined csv file from earlier (containing the Strava, Flickr and Untappd data) and a provided timestamp as inputs for the service. Please see figure 16 for the full script predicting the latitude and longitude values. It contains the final version of the terminal product, very similar to how the final GUI application will function as discussed below. The program takes a timestamp and csv file as input (the csv containing the data from the social media sites) and feeds them both into an input dictionary. It must be in a dictionary format since the html request needs to parse it as a json file, which is easiest to do from a dictionary format. The service returns the predicted latitude and longitude based off the timestamp the user sends to it (as well as the accuracy of each guess). Finally, the program concludes after displaying the data and total accuracy (to be used in our confidence rating).

The script does have its flaws. The input csv must be exactly 3 columns wide (latitude, longitude and timestamp) otherwise it will parse the file incorrectly and return an error message. Furthermore, the accuracy value for the longitude can sometimes be returned as a negative number. This doesn't necessarily mean the result is random, however. The co-efficient of determination ( $R^2$ ) is used as the accuracy in this application and can be inaccurate when working with negative float numbers. Our investigation found that the longitude values it was predicting were within a reasonable range of the original data, so we disregarded the lowest accuracy value because of this.

Figure 16

```

1 import csv
2 import http.cookiejar as cookiejar
3 import json
4 import math
5 import os
6 import requests
7 import urllib
8 from urllib import request
9
10 #Predictor Function
11 def Predictor(url, api_key, inputData, timestamp):
12     body = str.encode(json.dumps(inputData)) # Setup data
13     headers = {'Content-type': 'application/json', 'Authorization': ('Bearer ' + api_key)}
14
15     # Request machine learning response
16     req = urllib.request.Request(url, body, headers)
17
18     try:
19         response = urllib.request.urlopen(req) #Received
20
21         result = response.read() #Open binary
22
23         #Extract relevant data
24         decoded = json.loads(result.decode('utf8'))
25         outputData = decoded['Results'][0]['output'][0]['value']
26         return outputData
27
28     #Error thrown if connection fails
29     except urllib.error.HTTPError as error:
30         print("The request failed with status code: " + str(error.code))
31         print(error.info())
32         print(json.loads(error.read()))
33
34 #mainly start here
35 #Global
36 lat_url = "https://usouthcentral.services.azureml.net/workspaces/997d3d6794a451c868b442bdc628f7b/service"
37 lat_api_key = "
38 lon_url = "https://usouthcentral.services.azureml.net/workspaces/997d3d6794a451c868b442bdc628f7b/service"
39 lon_api_key = "
40 count = 0
41
42 #Extracts user input. Replaced with a form entry in final product
43 timestamp = int(input("Timestamp to base data off of: "))
44 if (type(timestamp) != int):
45     print("Invalid input. Make sure your input is an integer")
46     print("Exiting...")
47     exit(0)
48
49 #Input dictionary. Contains data from csv files. Final product will scan csv files
50 inputData = {
51     "Inputs": {
52         "Inputs": {
53             "ColumnNames": ["Column 0", "Latitude", "Longitude", "Timestamp"],
54             "Values": [{"0": "", "Latitude": "", "Longitude": "", "Timestamp": ""}]
55         }
56     },
57     "GlobalParameters": {}
58 }
59
60 #Insert big csv or json file here containing the location data (no quotations)
61 path = input("Path to csv file: ")
62 r = open(path, 'r') #open and read
63 reader = csv.reader(r)
64 for row in reader:
65     row.insert(0, count) #insert column id
66     inputData['Inputs'][0]['input'][0]['Values'].append(row) #append data from csv to learning input
67     count+=1
68
69 inputData['Inputs'][0]['input'][0]['Values'].pop(1) #pop off column names
70
71 latPredict = Predictor(lat_url, lat_api_key, inputData, timestamp) #lat predictor for lat
72 lonPredict = Predictor(lon_url, lon_api_key, inputData, timestamp) #lon predictor for lon
73
74 #Change accuracy to float value
75 float(latPredict[1])
76 float(lonPredict[1])
77
78 #Use highest accuracy for output
79 if (latPredict[2] > lonPredict[2]):
80     accuracy = latPredict[1]
81 else:
82     accuracy = lonPredict[1]
83
84 #Print out
85 print("Latitude Prediction = %s, Longitude Prediction = %s, Accuracy = %s" % (latPredict[0], lonPredict[0], accuracy))
86 print("Web Prediction Script Complete\n")

```

### 7.3 Webpage & Login:

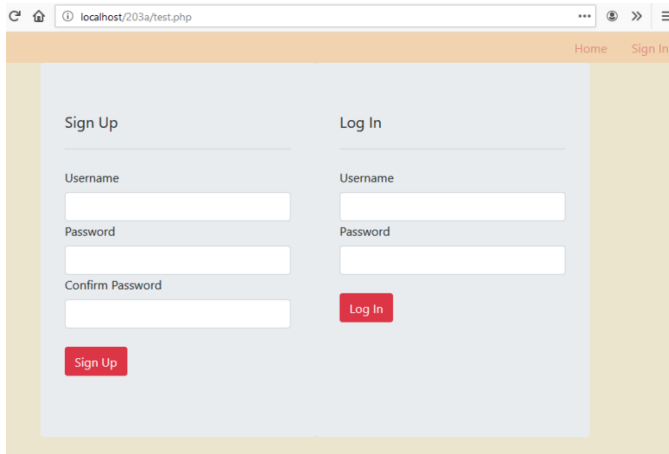


Figure 13

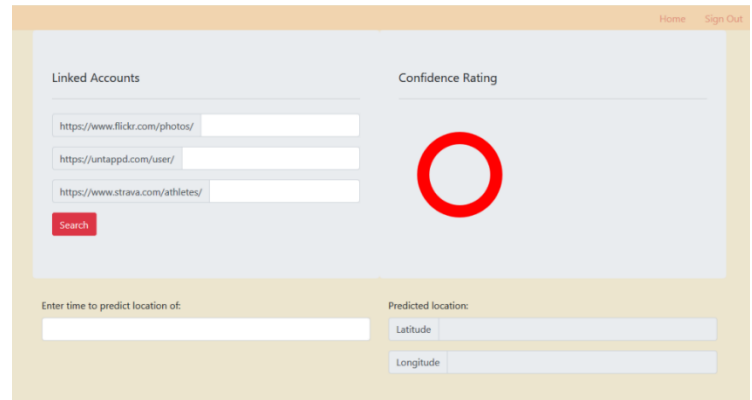


Figure 12

For the frontend of iProwlr, a simple php website was developed which calls the python scripts that then gather and process the data. The main page and login/signup page (Figure 15 & 16) were designed to be as simple and intuitive as possible to ensure ease of use for our target consumer. The values entered into the textboxes are processed by the python script explained in Section 7.1 and the results of the Machine learning algorithm (Section 7.2) are outputted into the Predicted location boxes in the bottom of the page.

A simple login system has been developed with a MySQL database. The database is linked to the webpage and when a user creates a profile, their username and password is saved into the table along with the timestamp of their last login (figure 17)



Table: users	
Columns:	
userid	int(11) AI PK
username	varchar(64)
password	varchar(64)
lastlogin	date

Figure 14

A script was developed to hash the saved passwords in the database in order to a level of security for the user, as saving the passwords in plaintext would be a serious security flaw. (Figure 18)

```
//hashing pass
$pass = password_hash($pass, PASSWORD_BCRYPT);

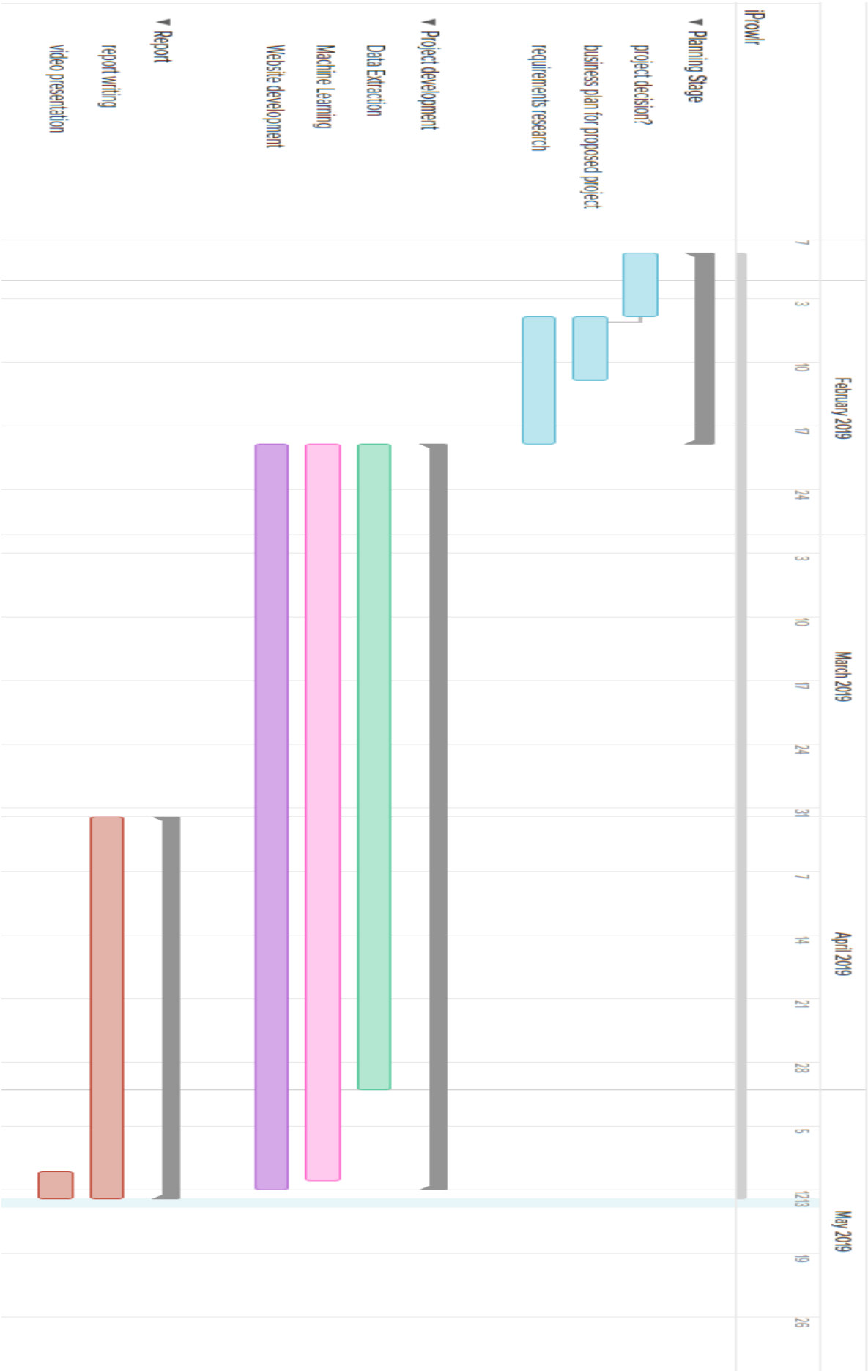
$stmt = $connect->prepare('INSERT INTO users (username, password, lastlogin) VALUES (?, ?, ?)');
$stmt->bind_param("sss", $user, $pass, $time);
$stmt->execute();
```

Figure 15

## **8.0 Project Management**

Tackling the development of a significant project such as iProwlr required us to plan and closely manage the multitude of tasks all happening simultaneously. Upon the conception of iProwlr, a Trello workplace was set up with a list indicating the tasks which had to be completed and who was allocated to each role. We decided on job allocation by compiling the list of the technologies requires for the project and as a group we discussed our strengths, weaknesses and preferences as a developer. From this we efficiently and logically allocated each task to a respective group member. Weekly group meetings and daily communications over Discord allowed us to stay up to date and organized throughout the project. However, this was arguably not the best way to communicate as it created issues around version control. We would sometimes have several Python scripts named the same or like each other, each a completely different version of code. Next time, we would recommend the use of a version control system (e.g. Git) to keep track of the changes in our code and ensure we only have one version of our Python at any one time.

8.1 Gantt Chart:



## **9.0 Evaluation**

iProwlr has proved successful in achieving our mission statement, which was to develop a web service which users can educate themselves on the potential privacy concerns centered around social media. It provides a concise and effective platform for regular internet users to quickly and easily aggregate their publicly available location information and predict their whereabouts at a certain time. Although, it doesn't have as many compatible websites integrated into it yet, the websites that are compatible cover most of the spectrum of common and relevant geolocation logging systems. The techniques used to extract the data from websites are effective at going unnoticed as bots to prevent getting blocked by the servers, although this had caused some speed sacrifices. The methods of data extraction also avoid permission issues requiring users to grant access to their data. iProwlr is easily expandable to integrate many more social media platforms and is a valuable tool for information security and risk management.

## **10.0 References**

Anderson, M. & Caumont, A., 2014. *How social media is reshaping news*. [Online]

Available at: <https://www.pewresearch.org/fact-tank/2014/09/24/how-social-media-is-reshaping-news/>  
[Accessed 23 May 2019].

Byer, B., 2018. *Entrepreneur*. [Online]

Available at: <https://www.entrepreneur.com/article/314524>  
[Accessed 25 May 2019].

Gross, D. & Hanna, J., 2010. *Facebook introduces check-in feature*. [Online]

Available at: <http://edition.cnn.com/2010/TECH/social.media/08/18/facebook.location/index.html>  
[Accessed 24 May 2019].

Hern, A., 2018. *Fitness tracking app Strava gives away location of secret US army bases*. [Online]

Available at: <https://www.theguardian.com/world/2018/jan/28/fitness-tracking-app-gives-away-location-of-secret-us-army-bases>  
[Accessed 13 May 2019].

Irwin, L., 2019. *List of data breaches and cyber attacks in April 2019 – 1.34 billion records leaked*. [Online]

Available at: <https://www.itgovernance.co.uk/blog/list-of-data-breaches-and-cyber-attacks-in-april-2019-1-34-billion-records-leaked>  
[Accessed 28 May 2019].

iTools, 2019. *Wink People Search*. [Online]

Available at: <http://itools.com/tool/wink-people-search>  
[Accessed 13 May 2019].

Kuksov, I., 2016. *What EXIF can tell about the photos you post online | Kaspersky Lab official blog*. [Online]

Available at: <https://www.kaspersky.co.uk/blog/exif-privacy/7893/>  
[Accessed 04 May 2019].

Malcolm, G., 2019. *Introduction to Artificial Intelligence (AI)*. [Online]

Available at: <https://www.edx.org/course/introduction-to-artificial-intelligence-ai-2>  
[Accessed 29 May 2019].

Matsa, K. E. & Shearer, E., 2018. *News Use Across Social Media Platforms*. [Online]

Available at: <https://www.journalism.org/2018/09/10/news-use-across-social-media-platforms-2018/>  
[Accessed 23 May 2019].



Microsoft, 2019. *What is - Azure Machine Learning Studio*. [Online]  
Available at: <https://docs.microsoft.com/en-us/azure/machine-learning/studio/what-is-ml-studio>  
[Accessed 07 May 2019].

Pipl, 2019. *Pipl - People Search*. [Online]  
Available at: <https://pipl.com/>  
[Accessed 13 May 2019].

Smith, A. & Anderson, M., 2018. *Social Media use in 2018*. [Online]  
Available at: <https://www.pewinternet.org/2018/03/01/social-media-use-in-2018/>  
[Accessed 23 May 2019].

Sotamaa, O., 2002. *All The World's A Botfighter Stage: Notes on Location-based Multi-User Gaming*. Tampere, Tampere University Press.

Spiegel, M., 2017. *Are Your Documents Leaking Sensitive Information? Scrub Your Metadata!*. [Online]  
Available at: <https://er.educause.edu/blogs/2017/1/are-your-documents-leaking-sensitive-information-scrub-your-metadata>  
[Accessed 28 May 2019].

Strava, 2018. *Strava Global Heatmap*. [Online]  
Available at: <https://www.strava.com/heatmap#12.01/-4.08856/50.38975/hot/all>  
[Accessed 13 May 2019].