I’m sure that everyone in this room is familiar with Twitter. In case you aren’t, twitter is a social media website that became famous for its 144 character status updates that users could post for the whole world to see. The userbase of Twitter has increased exponentially over the years, and now twitter users publish an astounding 6,000 tweets every second worldwide. There are no strict rules or guidelines as to what users or presidents can tweet about, so with all of this assorted public information flooding in at a constant rate, I believed twitter could be an invaluable tool for Data Science as well as being a social media website. Twitter also has a public API that helps developers catch incoming tweets from the official twitter stream and set to parameters to filter relevant tweets. The information associated with a tweet is not just limited to the body of the tweet and the username, but instead several additional attributes such as timestamps, location by city and country, geo coordinates, likes, and retweets just to name a few. I wanted to work with this information in some way, which led me to think about Machine Learning. With a proper Machine Learning algorithm I could take several thousand tweets and easily process them into something meaningful. My goal for this thesis was to determine if I could build tools to use Twitter and the data associated with tweets to produce a solution to a Data Science problem. To prove this, I needed to narrow down my research into a single particular problem.

My first attempt at deciding on a thesis topic led me to research the use of Machine Learning and Social Media for analyzing personality types. I had found some journals on doing just this, but one in particular seemed to be right for me in terms of how the experiment was done and the total results obtained. *Deep learning for constructing microblog behavior representation to identify social media user’s personality* used a combination of data mining on a Chinese social media website similar to twitter, Machine Learning, and a traditional personality assessment known as Big Five. Participants in this experiment would fill out the personality assessment to create the control variables, and then their social media history from the website was mined, and ran through a word engine to convert the words and sentences into vectors. The SVM was then used to sort this data and produce clearer results which could then be compared to the Big Five results. I had put my focus into this idea for 6 weeks until I realized that my experiment would not be mathematically sound because of my control variable. Personality isn’t something you can measure with conventional tools or methods, and a personality test is far from a perfect indication of a person’s personality. I would need to find a new topic with a more sound control variable.

I’m sure that everyone in this room is also familiar with Influenza, also known as the flu virus. Some symptoms include fever, chills, coughing, congestion, and vomiting, but overall an unpleasant experience for those affected by the virus. In some cases, however, the flu virus can prove deadly and when outbreaks happen, it is important to know how and where to find information related to outbreak locations. Currently, the most official reports of the Flu are done by the Center for Disease Control, as well as World Health Organization. The problem with viewing these records has to do with the speed that Official Influenza Reports are published. As I am reading you this slide, the current weekly report from the CDC is for the week of April 21-28, which is almost a 3 week delay. For those trying to avoid catching this virus, 3 weeks is certainly enough time for an outbreak to develop. My goal was to find some method for gauging the activity of the Flu virus that could provide much faster results, even at the cost of accuracy.

I eventually found myself with another promising paper, but this time instead of personality assessment, I would be attempting a more quantitative goal. The paper I read described using a private twitter database and some machine learning algorithm to find the number of tweets in a certain area mentioning Dengue Fever, comparing them with official reports, and then once their hypothesis was true, used their model to predict future outbreaks that could later be compared with reports as they were published. The team proved successful at mining the data and creating a forecast for Dengue Fever in Brazil, so I wanted to recreate this experiment for myself, but this time using Influenza as my disease and America as my testing ground. There was another journal I had read that did do just this, however, in their experiment they neglected organizing their tweets by hand or with the use of Machine Learning. For my experiment, I wanted my data to have as little noise as possible, and the use of machine learning would ensure that only the tweets I wanted that mentioned the word Flu would be counted. All I would need to do was connect to the twitter API, gather my tweets, and then set my Machine Learning algorithm loose to help me classify which tweets I would need for the data analysis portion.

Connecting to the Twitter API was a less painful experience than my usual CS homework assignments, and writing a script to mine the tweets also proved to be relatively smooth since the tweets were returned from the API as JSON packets. This is a code snippet for my mining algorithm that features the powerhouse method doing most of the work. I placed a filter for the twitter stream to only look for tweets that had the word “flu” in them, and that were also written in English. On the right side is what a tweet looks like in JSON form without the help of my method for cleaning up the data. You can see a lot of the fields I had mentioned before, and every single tweet I gathered returned all of this information. Obviously, a lot of these fields were not relevant to me whatsoever, so I created the method on the left to pull out the necessary data, like the Date, the Tweet itself, the City, and the Country, and then formats it into a list which can be loaded into a CSV as a row for later. It also knows how to deal with emojis, specifically ignoring them instead of crashing spectacularly, and to help my SVM from getting confused I also omitted usernames, retweet usernames, URLs, and new lines. The tweets were then almost ready for the SVM.

I can guess there are probably a few people here that are not familiar with Support Vector Machines, so I will need to clarify a bit. A support vector machine is a supervised learning algorithm that trains itself to create a generalizable function to classify data. Since an svm needs to train, it will need a training set. Given an entire corpus of data, we will need 2 subsets of data: The Training set, the Data Testing set. The training set is ideally chosen at random, which mine was, and then the data is labeled by hand. The svm will then take a small subset out of the training set to use as the validation set for later. The SVM trains itself a model until it can correctly label the training set data. With this model, it then attempts to classify the validation set of data, and assigns each entry with a new label. It then compares the new label generated by the model with the label given by the human to check its accuracy. After this, the model can then either be tweaked or given a larger training set, or it can go ahead with classifying the unlabeled data in the data set.

Going a little deeper, what the SVM is doing with the data it is given involves some math, which varies depending on what kind of data it is working with. After crunching some numbers, it plots the results for the data onto a graph, in this example we have two labels represented by the red and blue points. The SVM then tries to generate a hyperplane to separate each group of data as neatly as possible. To ensure the groups are split as ideally as possible, it relies on the two closest points from each group to tweak the hyperplane equation. The two, or in this case four, closest points are known as the support vectors, and the SVM attempts to find the ideal hyperplane that is maximally equidistant from the support vectors. We can also see some additional points that do not belong to the red or blue group. In this example, these points represent unlabeled data that belongs to the red group and how it was placed within the model. The symbol next to these points represents the slack variable. A value of 0 means it falls in line where it is supposed to be and is given an ideal classification. A slack smaller than 1 means the point lies close to the hyperplane, which is not ideal, but is still labeled correctly. Finally, a slack greater than 1 means that the point is misclassified. The SVM is also able to be tweaked manually by the user, which involves adjusting what is called a C value. The C value represents the tradeoff between accuracy and speed of the SVM. A C value approaching infinity will be incredibly laborious in the placement of the hyperplane and of classifying data, whereas a C value close to 0 will have a very fast, but lazy SVM. Much like a freshman, it will simply answer as quickly as possible with little to no regard for the correctness.

While it may seem simple at first glance, this is an incredibly simple example which only uses 2 dimensions for its data. SVMs are capable of performing this process on many-dimensional graphs, as well as using non-linear Hyperplane functions to curve in between data points as if every point was a support vector, making them incredibly powerful and tweakable tools.

For my svm, I wanted it to classify the tweets as either valid or invalid before I started comparing them to medical reports. For a tweet to be considered valid for this experiment, I was looking for any context that the person tweeting it had a recent personal experience with the flu virus. Invalid tweets were the opposite, and more often than not, were tweets about using the flu as a simile or people arguing about vaccines. My complete training set contained 1,000 tweets, and from there I split the set 90/10 into the training and test set. I was able to achieve mostly 84-88% accuracy when experimenting with training set size, but I reached a high of 90% which I then used to classify my remaining corpus of tweets. Then I simply had to take my results and get rid of all of the invalid tweets and start comparing my data.

For my experiment, I used my mining script to gather tweets for approximately 1 hour each week day in the afternoon. However, there are some days where my mining either had to be cut short, I forgot, or I did not have any free time with my computer to mine. I did manage to mine 20 times and massed a total of around 8.700 tweets. After my SVM went to work, the number of valid tweets I was left with was 2,650 tweets. Using the other information I had gathered about city and country information I was left with 60 tweets. This number may seem small, and it is. Twitter users are able to set a city and country as location data for tweets, however it seemed that many people had turned this feature off, since 8,440 of those tweets had no location data whatsoever, so I made a rather large assumption that tweets being posted in English around 2pm eastern time would be mostly from North America. I was also able to find CDC and WHO data for all but 3 days of mining and these are what my results looked like.

Here are my results graphed with Number of Tweets and Cases vs Days. The graph shows the number of tweets per hour in a day that were marked as valid alongside of points representing confirmed cases of the flu from clinical centers. The large missing gap of twitter data is unfortunately from spring break. Since the CDC reports only came out weekly, I was unable to directly compare my results with daily CDC reports. Also, since I was only able to gather tweets for an hour each day, I took the number of reported cases and divided it by 24. While this may not be the correct method of comparing numbers directly, it does place the CDC points closer to the Tweet points to more easily observe the general trend. Both graphs are decreasing over time in a similar fashion, possibly hinting that they do indeed share a positive correlation with one another.

I do consider this experiment a success, and also proof that Twitter can be used as a powerful data science tool. I was able to successfully connect to twitter and start pulling tweets that were processed into a workable file structure. Then I could use my acquired data to create a training set for my SVM and classify tweets with high accuracy. My findings showed promise, and with more data and work in the future, linear regression techniques could prove that I have found a positive correlation between my variables. This project can easily be altered to track trends of other diseases, or any topic at all. For example, this could be used as a tool to track popularity of candidates for an election, or for tracking mentions of a certain product to determine where advertising or sales could be boosted. The main limitations for this would be that labels or categories would need to be kept relatively simple and unique, and there needs to be enough data in order for the SVM to learn properly, but this method can still apply to a very large number of data science problems.

This experiment had a lot of shortcomings that I simply didn’t have the time or resources to address in this thesis. I was only able to record data on week days for around an hour a day, so it would have been more useful to have a machine constantly running and recording tweets for 24 hours each day. Additionally, the twitter API stream is bottlenecked because of the large influx of tweets coming in, so my script was not able to catch every single tweet that happened to contain the word flu. Having multiple sessions or machines running this could also provide me with more data to paint a better picture of the relation between number of tweets and number of reports. Data location was one of the biggest problems I faced with this lab because so many people have it turned off by default. Certain users have geo location turned on as opposed to location, so the tweets I gather could also use coordinates as well as location to separate tweets from only Americans. If I did have access to more tweets, I could also see if this would make a difference in providing my corpus with enough American tweets to create a useful relation.