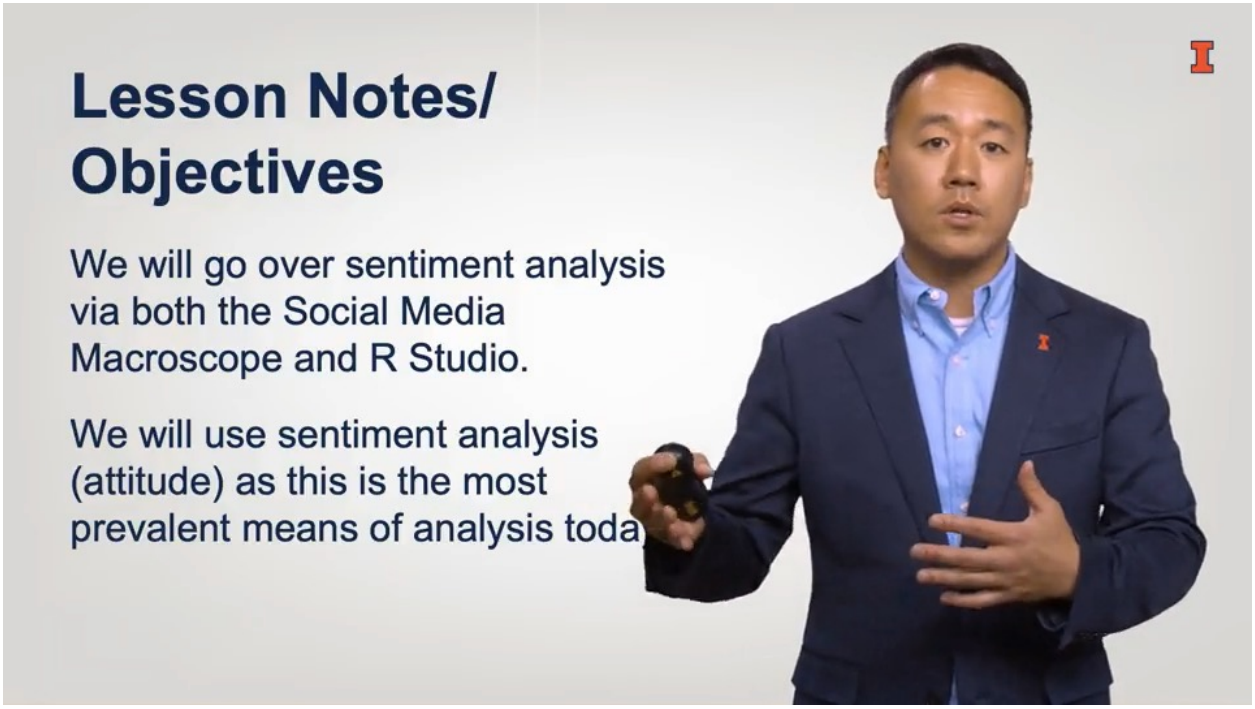
Module 3

Table of Contents

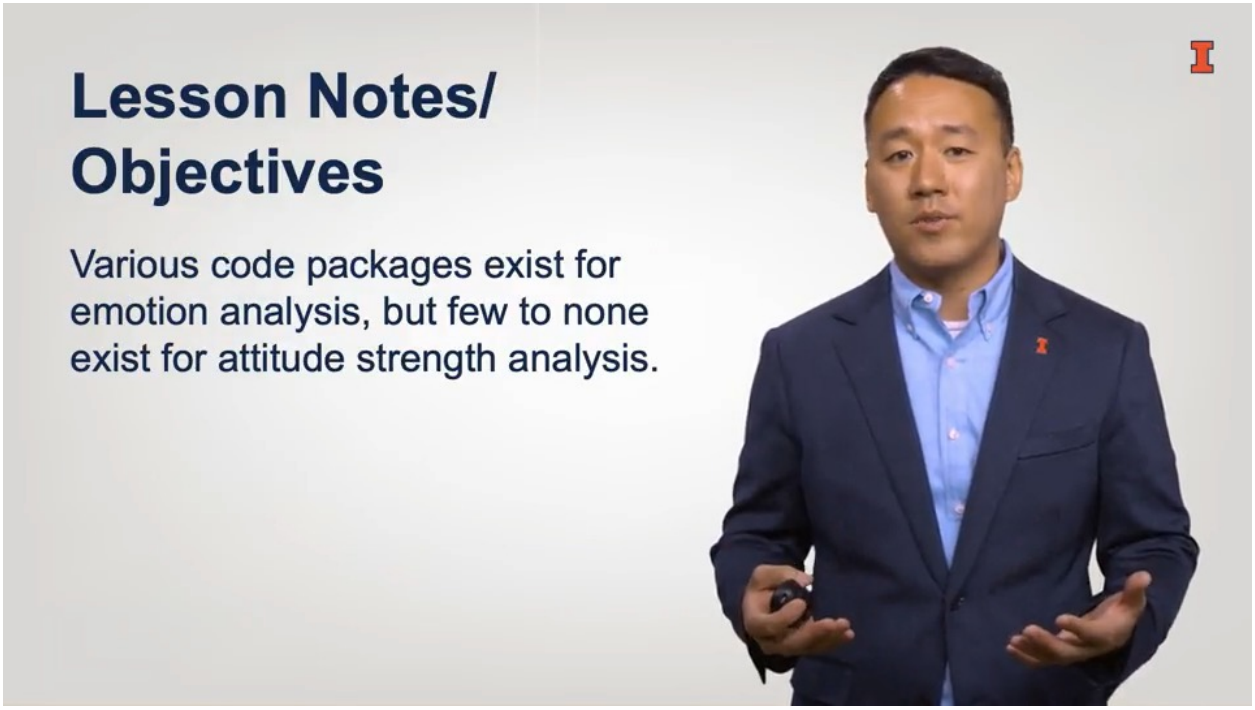
Lesson 1: Text Analysis

Analyzing Social Media Attitude Part 1

Okay, everyone, now that we've looked at analyzing the airline survey dataset for customer satisfaction analysis, we will now switch gears and start looking at analyzing social media data for attitudes. Now some lesson notes and objectives. We will pull Twitter data. That's not the only social media data out there, but that's a dataset that is readily available to pull from and we will use a tool called the Social Media Macroscope. Now, you've already been introduced to the Social Media Macroscope in Module 1, but the reason why we're using the SMM, if I can call that by short, to pull Twitter data is due to API key limitations. You could actually go to Twitter and get your own API keys and try to pull this data yourself without the SMM. The problem is, Twitter has been much more stringent about proving applications for API keys. The way that we're able to get around that is that the SMM has already been approved to pull from Twitter. You can access their Social Media Macroscope and pull Twitter data as long as you have an active Twitter account. You will need an active Twitter account to pull Twitter data. Also, due to Twitter's terms of service rules, you need to pull your own data. Meaning, I've pulled data, but I cannot give that to you as part of this course for you to analyze because that breaks their terms of service. I will show you demos of me pulling data, but you will have to pull your own data to analyze for the various lessons and exercises.

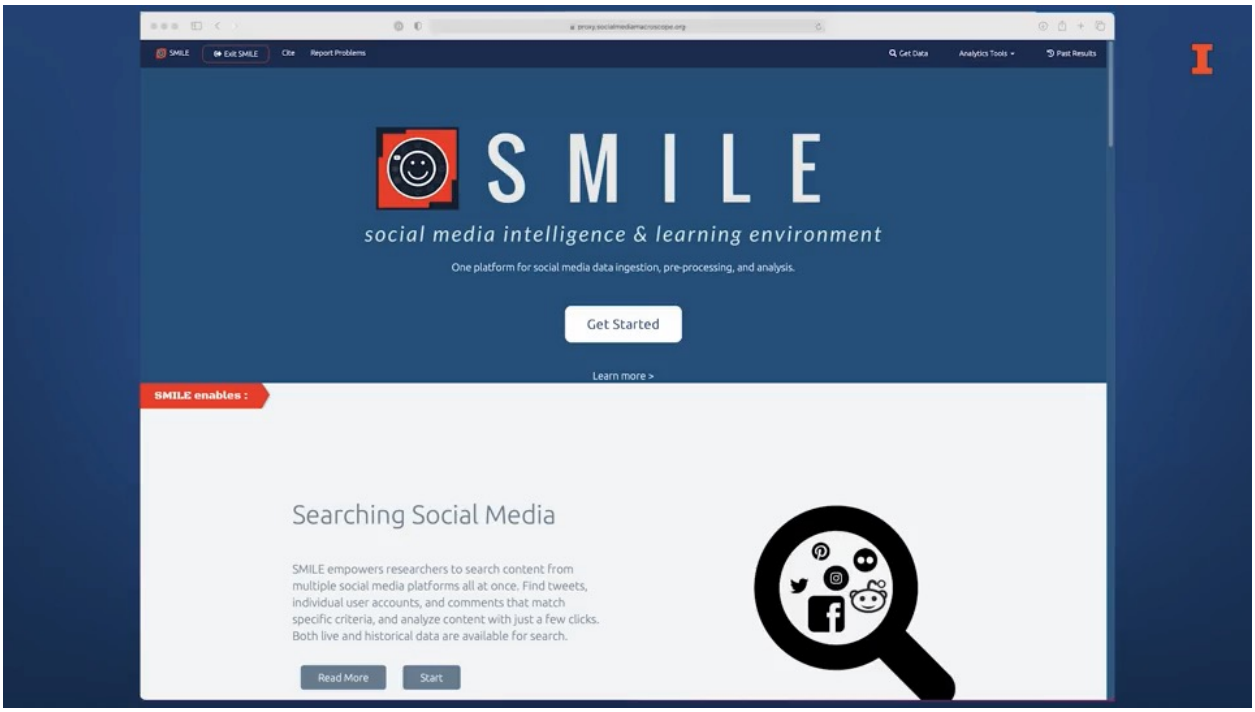


Now what we'll do in this lesson is that we will go over sentiment analysis via the Social Media Macroscope, but then in the subsequent lesson, we will look at sentiment analysis using R Studio. Now we're using sentiment analysis or what is called attitude for these videos as this is the most prevalent means of analysis of customer satisfaction today. But remember in our psychological construct lessons, that it's not just attitude that we should be looking at.

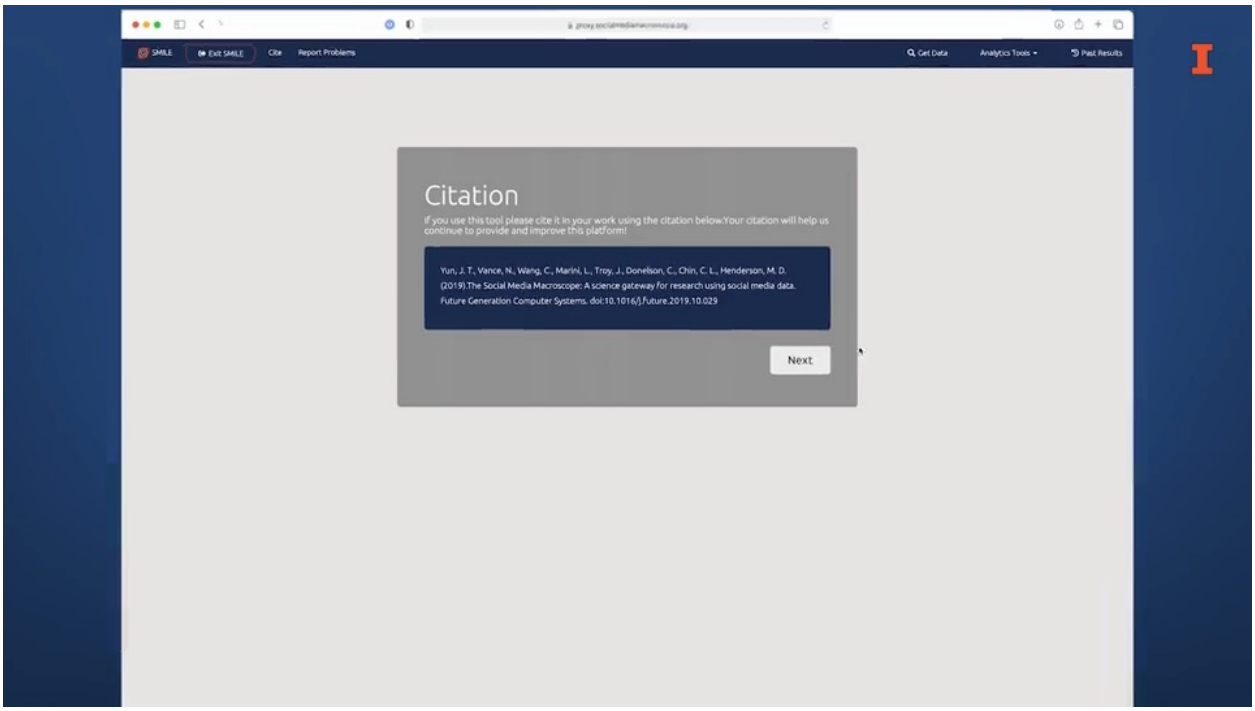


Various code packages exist for a motion analysis, but few to none exist for attitude strength analysis. That's really an open area of research for myself and maybe for you all as we try to computationally

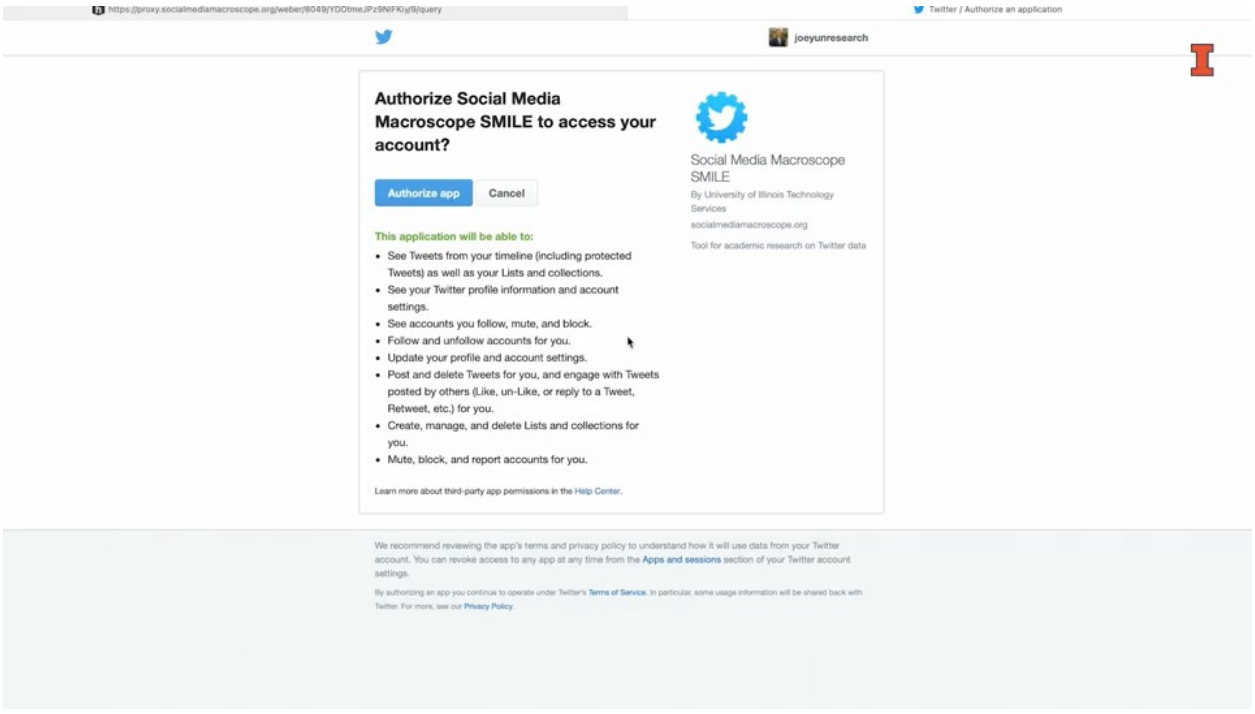
understand people's emotion as well as hopefully in the future, understand people's attitudes strength via their social data. Let's jump into sentiment analysis and let's go right into the Social Media Macroscope right now.



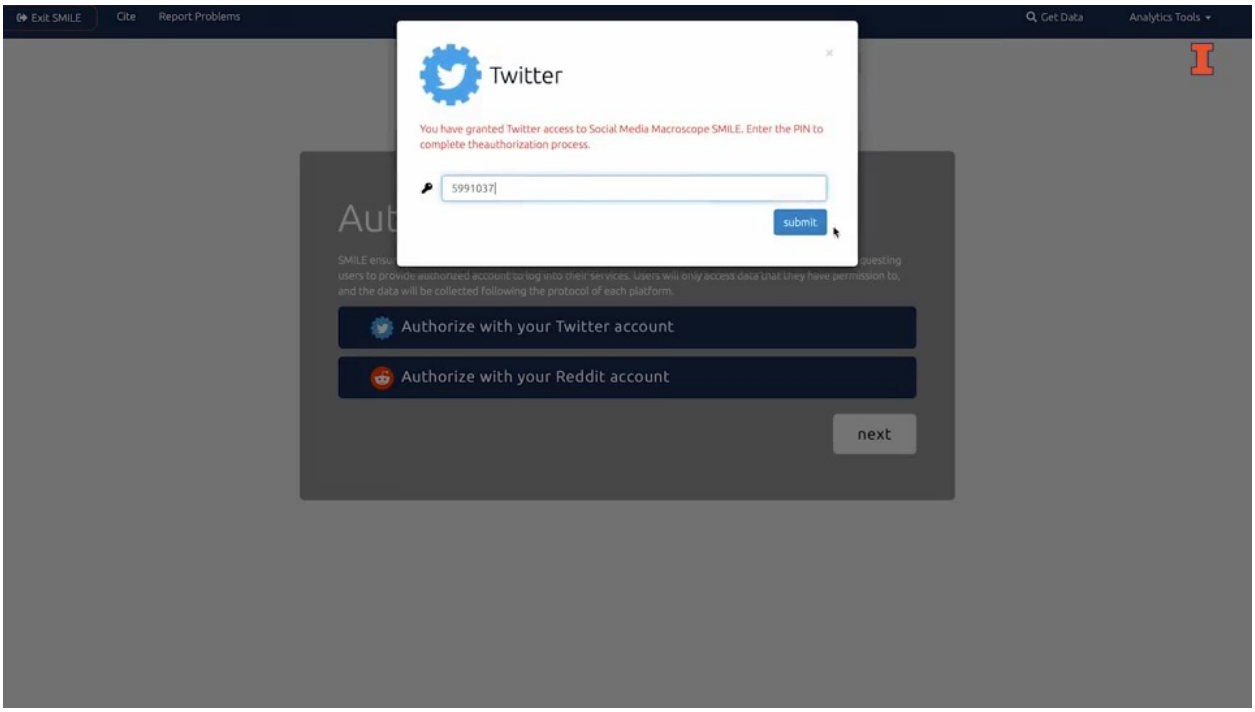
Everybody, we're now in the Smile tool and we are going to search Twitter for a brand. I'm going to search for the brand Apple. I've been talking quite a bit about Apple computing throughout these lessons, but you can choose any brand. All you need to know is their Twitter handle so that you can put in the right search term for Twitter.



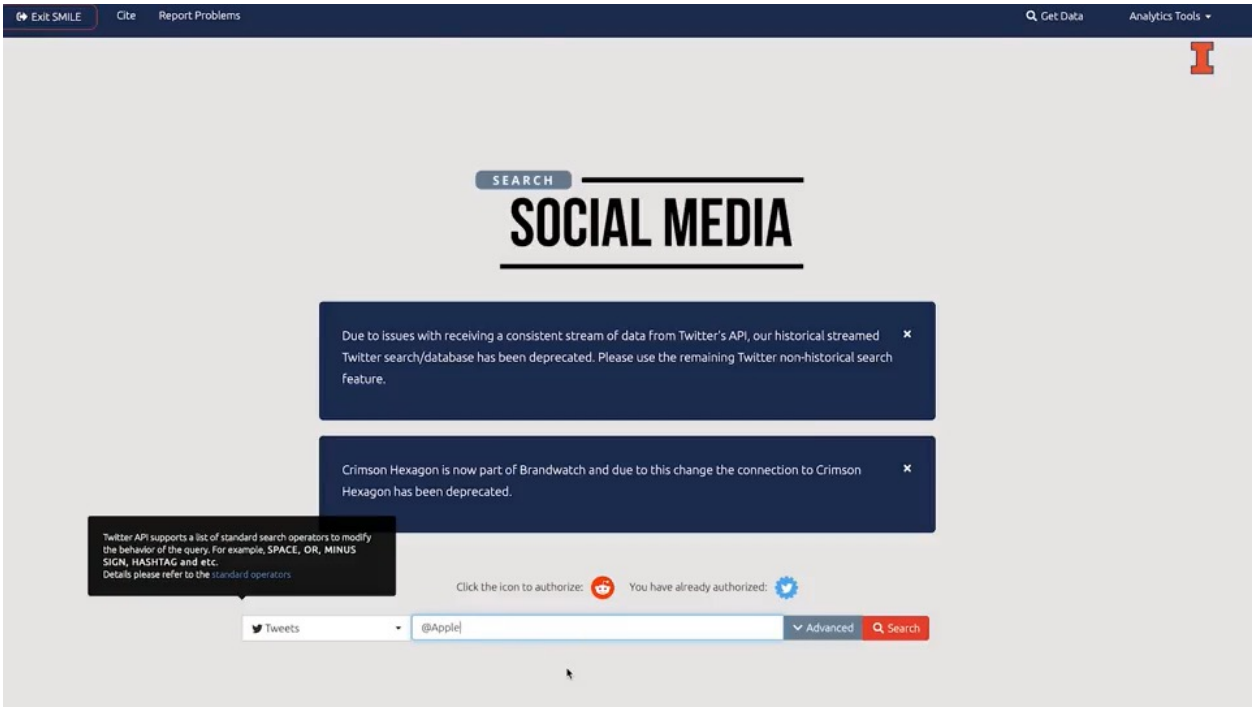
Let's get started here by clicking on "Get started." If you'd like to cite our work, if you're doing research that would help with us to continue to be funded. Click on "Next" after the citation screen. We need to authorize with our Twitter accounts because we're going to pull Tweets.



We'll click on "Authorize." This pops up a permission pop-up from Twitter asking if we want to authorize the Social Media Macroscope Smile tool to access our account so that it can get API keys.

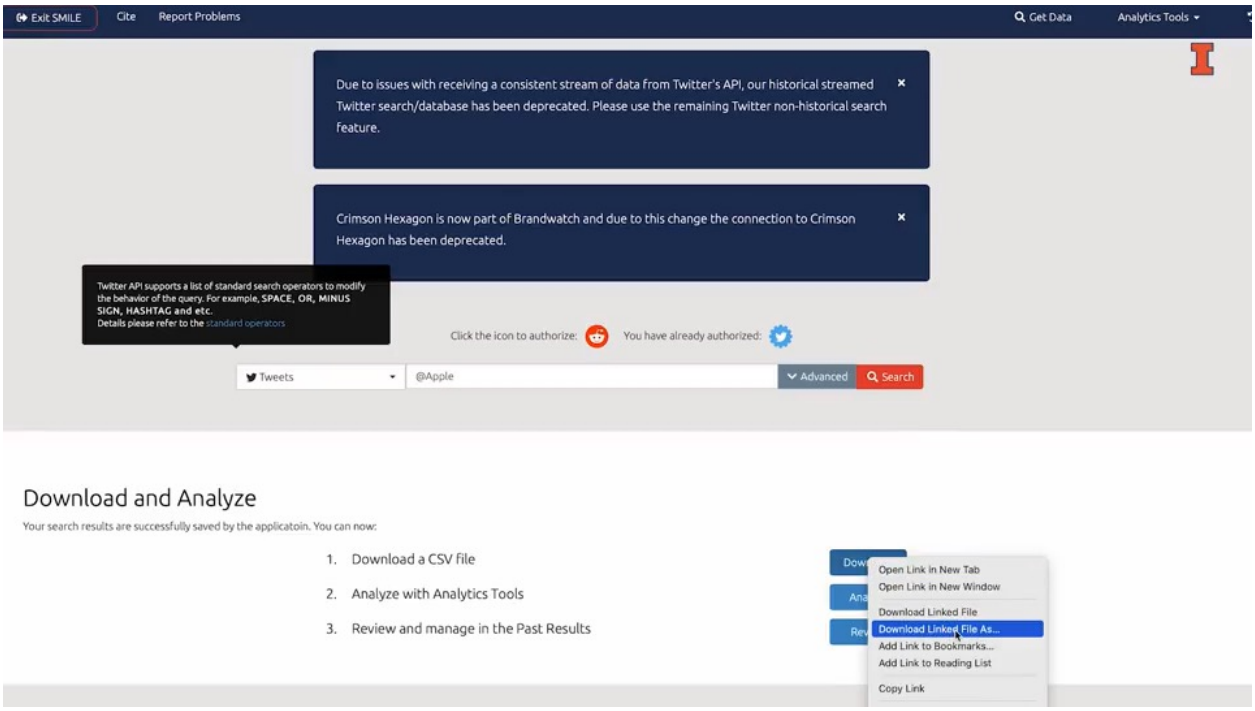


We'll say Authorize app, copy the pin and paste the pin into the Smile pop-up. We'll see a green verified icon here that says that we have been authorized to pull Tweets. Let's click on "Next." Let's select, "Choose tweets" to search. I'm going to put in @Apple.

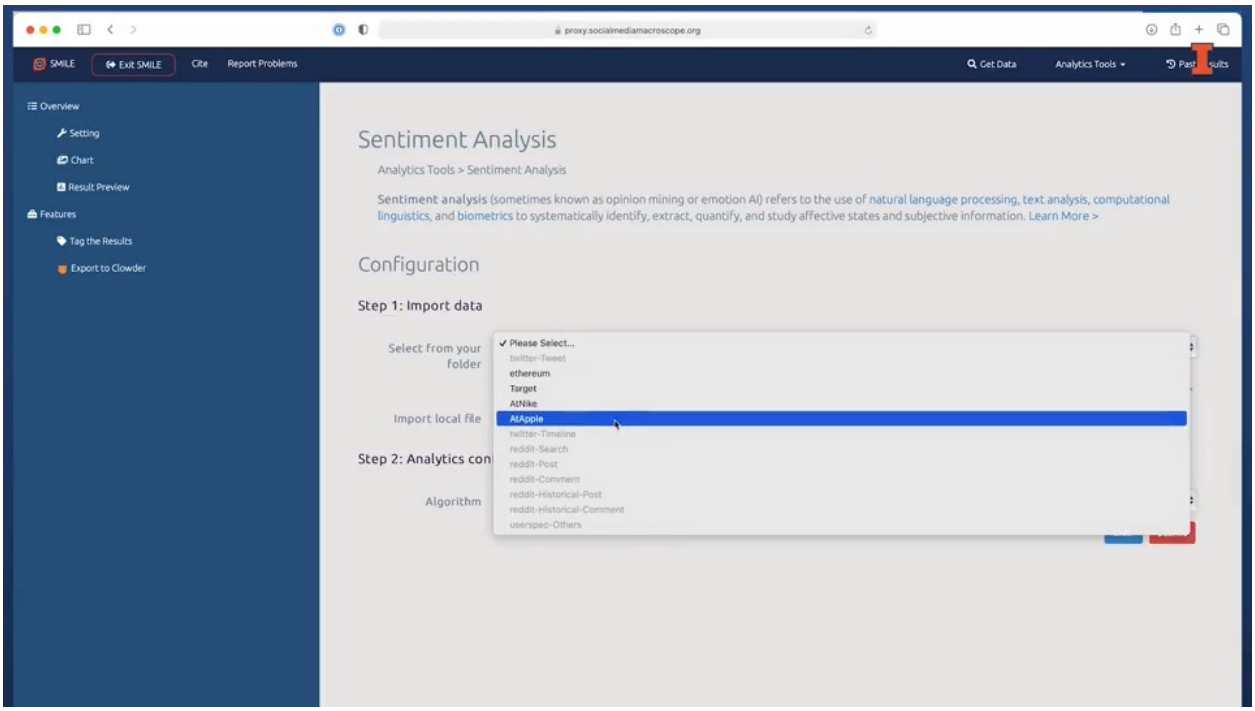


The reason why I'm putting @Apple is because I want to see how people are talking about the brand named Apple so that we can conduct sentiment analysis on that. Then that will enable us to understand some aspects of people's customer satisfaction when it comes to Apple. Now here is a preview. We can see that people are truly talking about Apple computing.

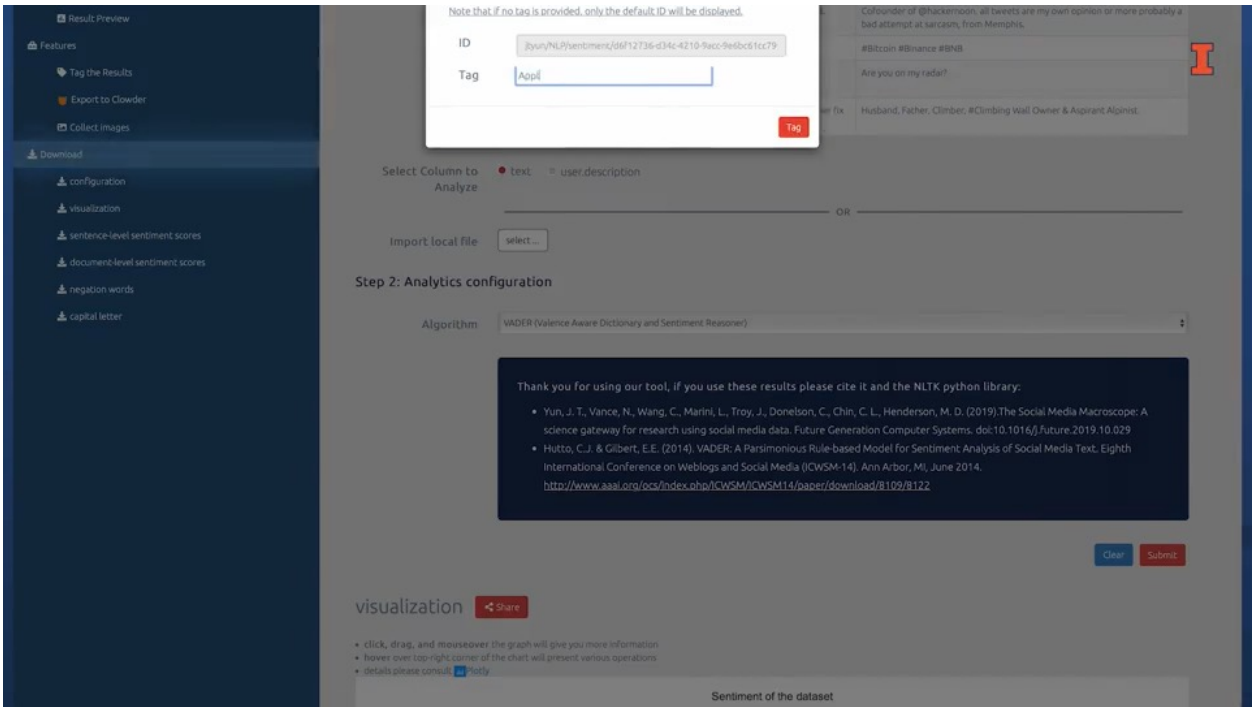
We will save the file name as I'm going to save it as @Apple, click on "Save" here. Now this is pulling against Twitter's API. Using our search term @Apple, downloading the results and bringing them into the Smile tool. Now this process has finished.



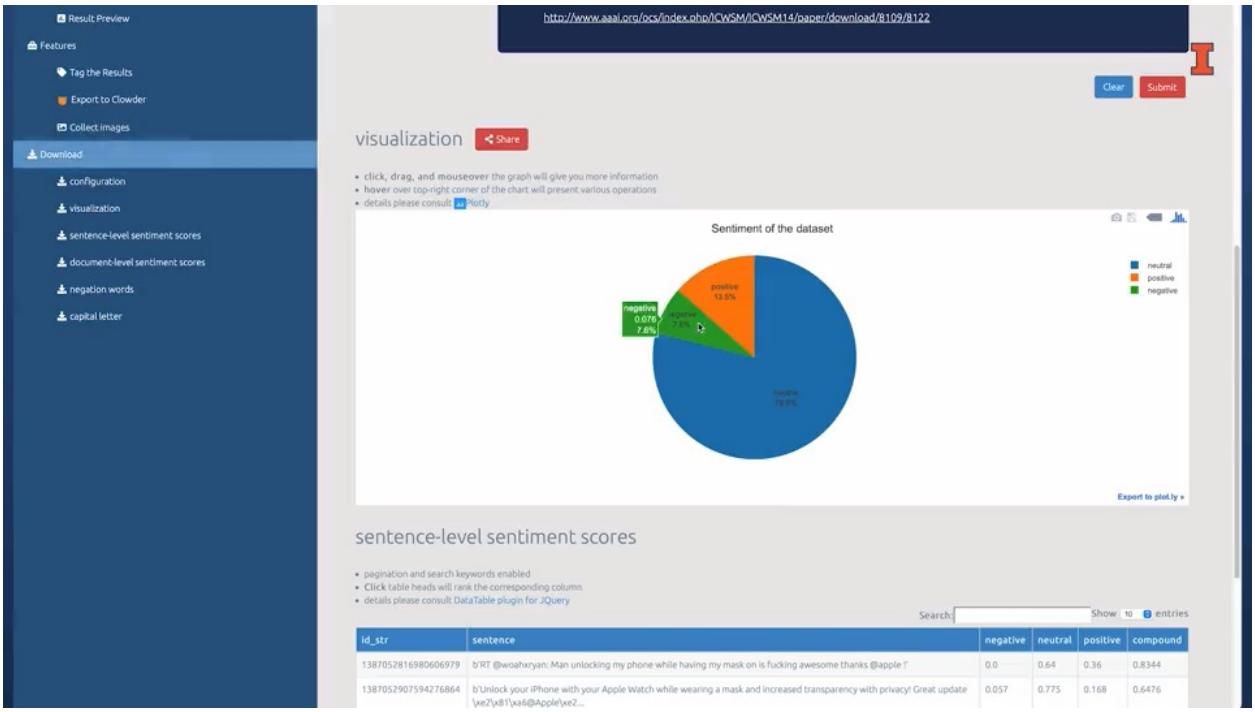
Now what I would like to ask for all of you to do, because we are going to also do sentiment analysis via R, is go to this Download a CSV file portion, and right-click on "Download." If you're on a Mac, you'll see this Download linked file as if you're on a PC, you may see something else that explains how to download. Just choose the download function and save it somewhere. I'm saving it to my desktop. Now let's actually use Smile sentiment analysis functionality to analyze the tweets regarding Apple. You go to an Analytics tools up here at the top right-hand corner.



Click on "Sentiment analysis." We will choose @Apple. It will show us some preview of tweets. Seems correct. Let's choose VADER. VADER is a acronym for valence aware dictionary and sentiment reasoner. It's a sentiment model that was created in 2014 by Hutto and Gilbert. It's specifically built for short social media text. It works really well on tweets. You can actually read the whole paper right here or you could look at the suggested readings that are part of this lesson within this module. We're going to choose VADER, and we're going to click on "Submit." What this is doing is it's running VADER scores for each one of the tweets. Now the way that VADER works is it calculates a different score for negativity, for neutrality, for positivity, and then a compound score. You could actually break it up between any one of those dimensions, but what we'll focus on is looking at the compound score overall. Now Smile is asking us to tag the results.



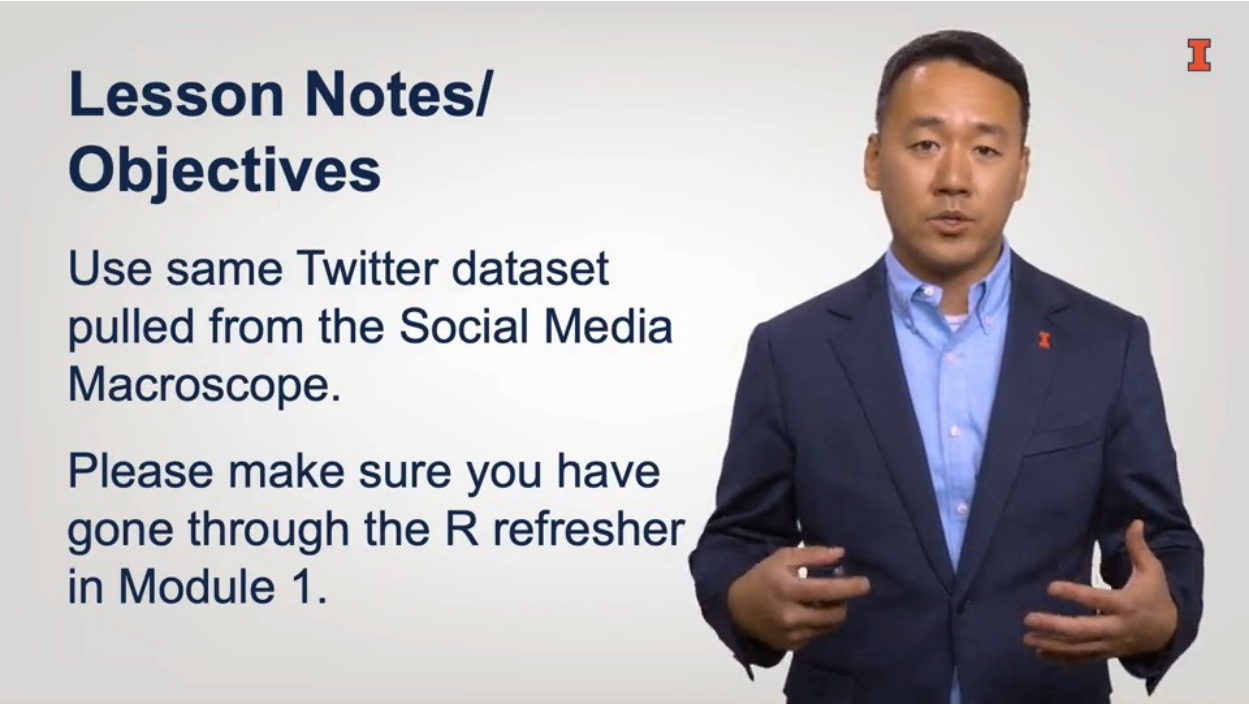
We'll tag it as Apple, we'll tag it as VADER and we'll tag it as sentiments. Click on "Tag." Here's high level overview. Most of the results come out as neutral.



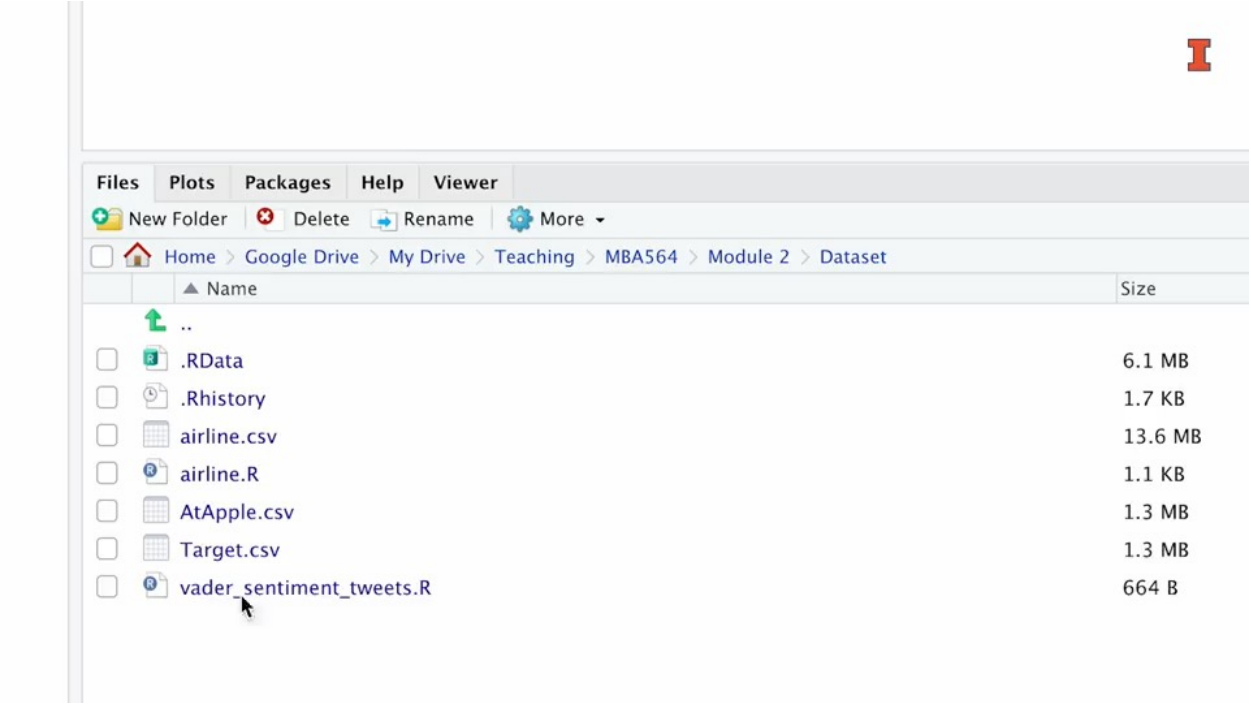
Some are positive, some are negative. We can actually see some results here. You can see a negative score, a neutral score, a positive score, and a compound score. VADER scores things from a negative one to positive one. For example, overall it seems like this score is very positive 0.8344. This is how you run a sentiment analysis by pulling tweets regarding a brand. You could look at each one of these tweets and try to get an idea via the sentiment, people's attitudes. This gives us a little bit of insight into customer

satisfaction. Now when we pull this data into R, in the next lesson, we'll take all the compounds scores or at least a few of the compound scores and work out an average so that we can get more of a high-level view of people's customer satisfaction towards Apple, at least on the conversation within Twitter.

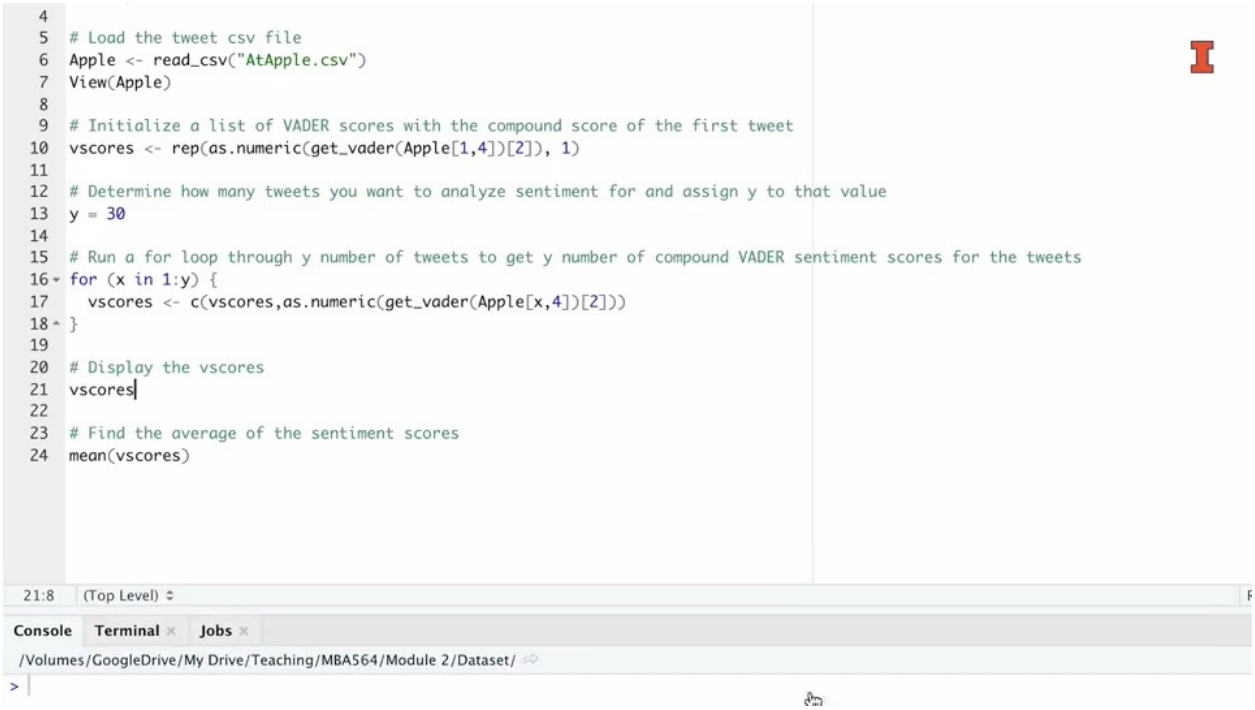
Analyzing Social Media Attitude Part 2



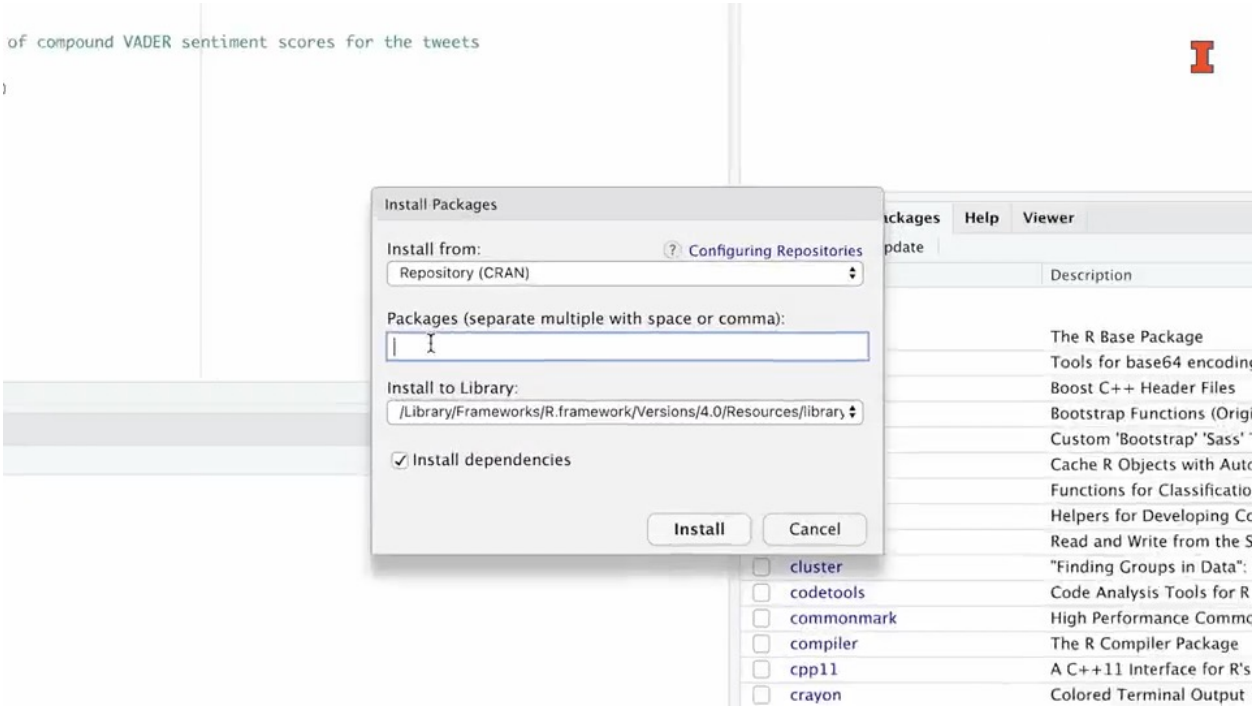
Everyone, I hope you all had fun with the Social Media Macroscope and analyzing sentiment. We will now do the same thing. but in RStudio. You will need the same Twitter dataset that you pulled from the Social Media Macroscope, and in the previous lesson, I explained how you could download a CSV of your Twitter dataset, so you will need that on hand to pull into RStudio. Please make sure that you have gone through the R refresher in Module 1. You've already gone through R in the previous module here. But just to make sure, since we're going into deeper packages, that you will need to make sure you know how to run through R, and pull packages and run those packages in an RStudio for this lesson. With all that said, let's go into sentiment analysis using RStudio.



Okay, everybody, now we are in R studio and what you can see in my working directory here on this right bottom corner is that we have the script entitled vader_sentiment_tweets.R. That is a script that I've written that is provided in your resources for this module, but you also see a CSV file called AtApple.csv That's the CSV file that I created previously in the previous lesson when pulling tweets that mentioned @Apple. Now you're not going to have that CSV file specifically unless you pull Apple tweets and also name it AtApple.csv, but you'll have whatever your CSV is. Here you see I created another CSV and another time called Target.csv, looking at the brand name Target, but again, it's whatever your brand name dot CSV is, is the brand name that you'll be looking at. I cannot provide you my files once again because of Twitter's terms of service rules, and so I cannot provide you tweets that I've downloaded.

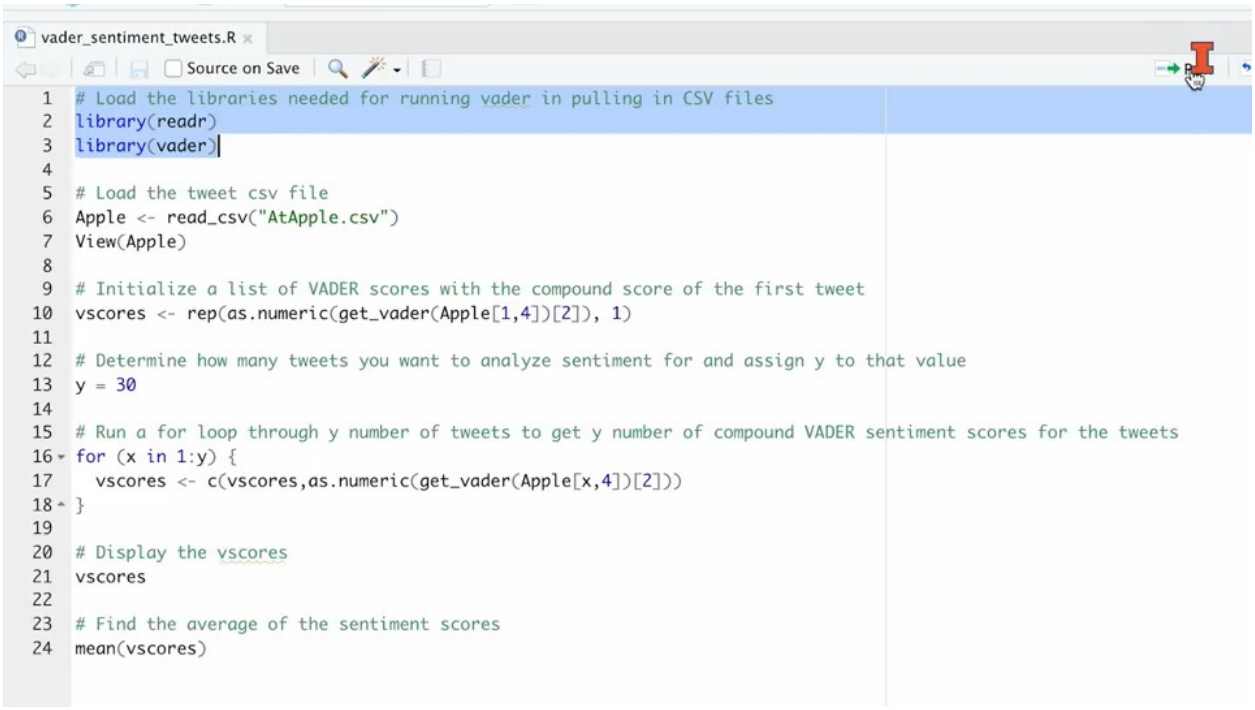


Let's go ahead and let's open the vader_sentiment_tweets.R script here. What I've done here is I've written just a series of commands that get pulling Vader scores for the various tweets that are found in the CSV file.

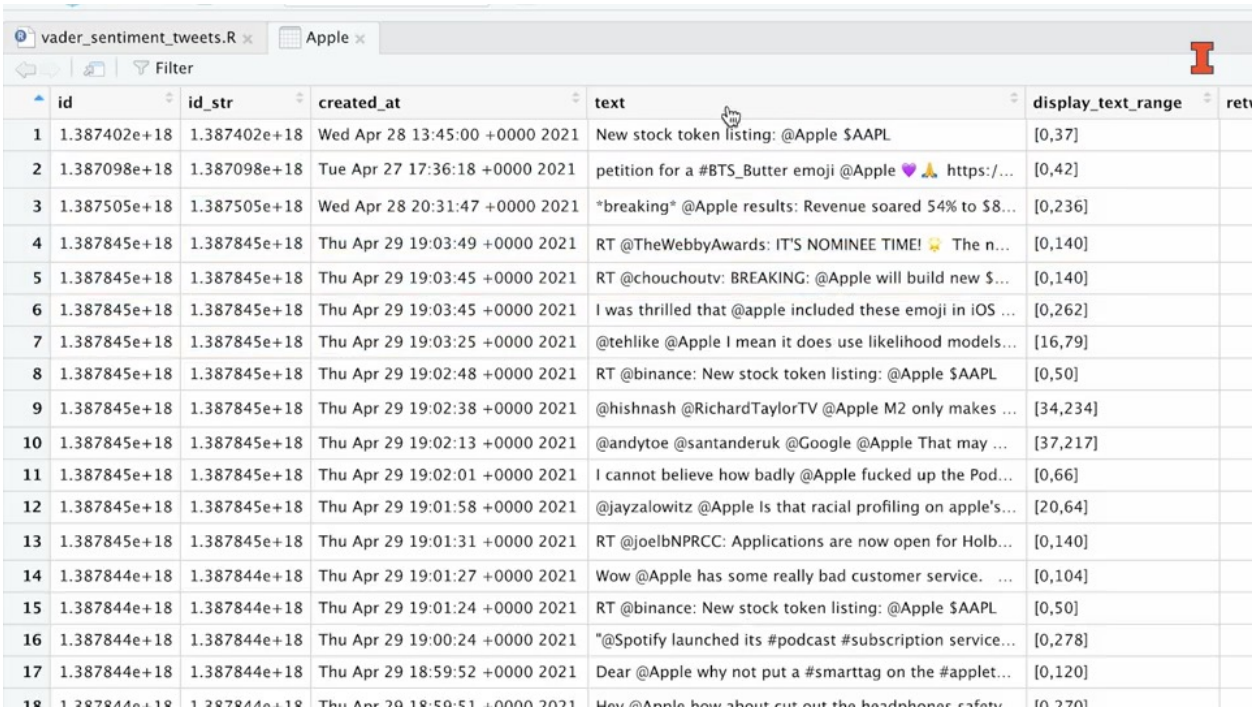


Now the first thing that we actually need to do is we need to load Vader, so if you have never loaded Vader into your RStudio, you can go to this ''Packages'' section and you go to ''Install'', and then you can just type in Vader, and you'll find Vader there and you can click on ''Install''. Then it will just go through the normal R commands to install the package called Vader. I've already done that, so I'm just going to cancel,

but you'll need to do that first. Then once you do that, you should be able to load the libraries. I'm loading two libraries here. One is the library to read in the CSV file, the other is the library to use Vader.

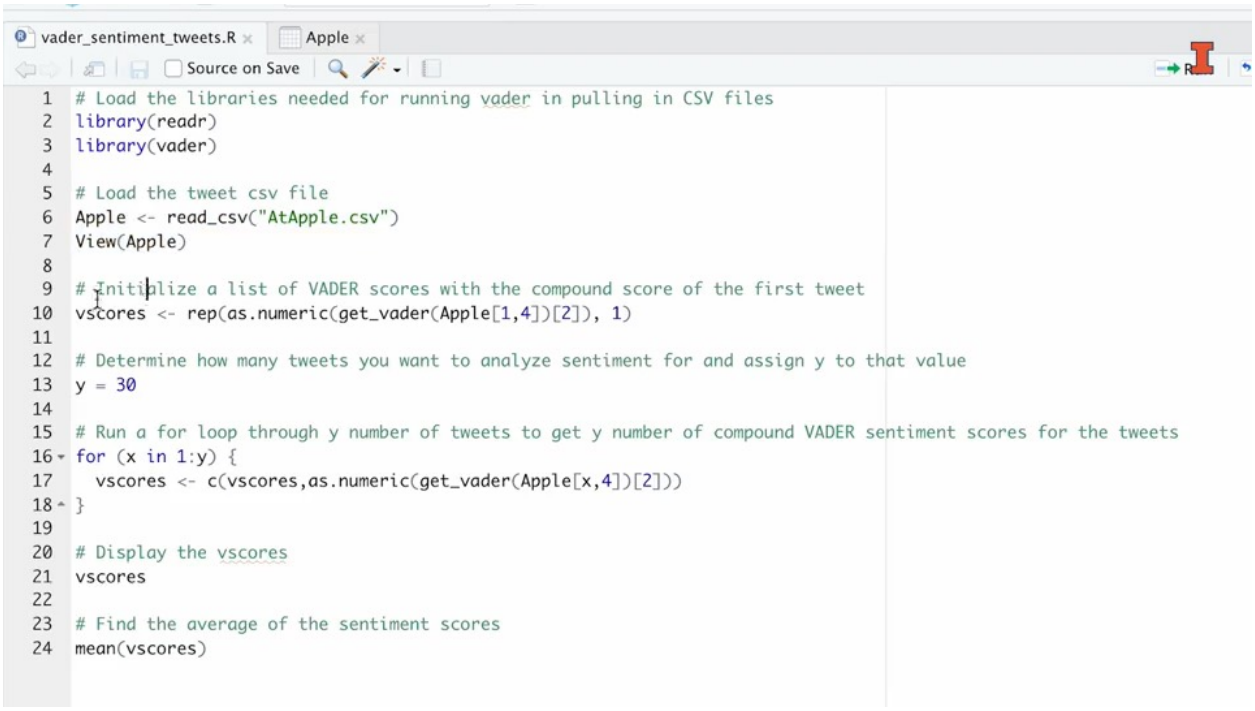


I'll click on ''Run'' for these two chunks here. Now let's load the tweets CSV file again, my At.Apple.csv is in my R working directory right now, so you want to make sure that your CSV is in your R working directory, and I'm going to put that into the data frame variable called Apple, and then I'm going to view that data frame.

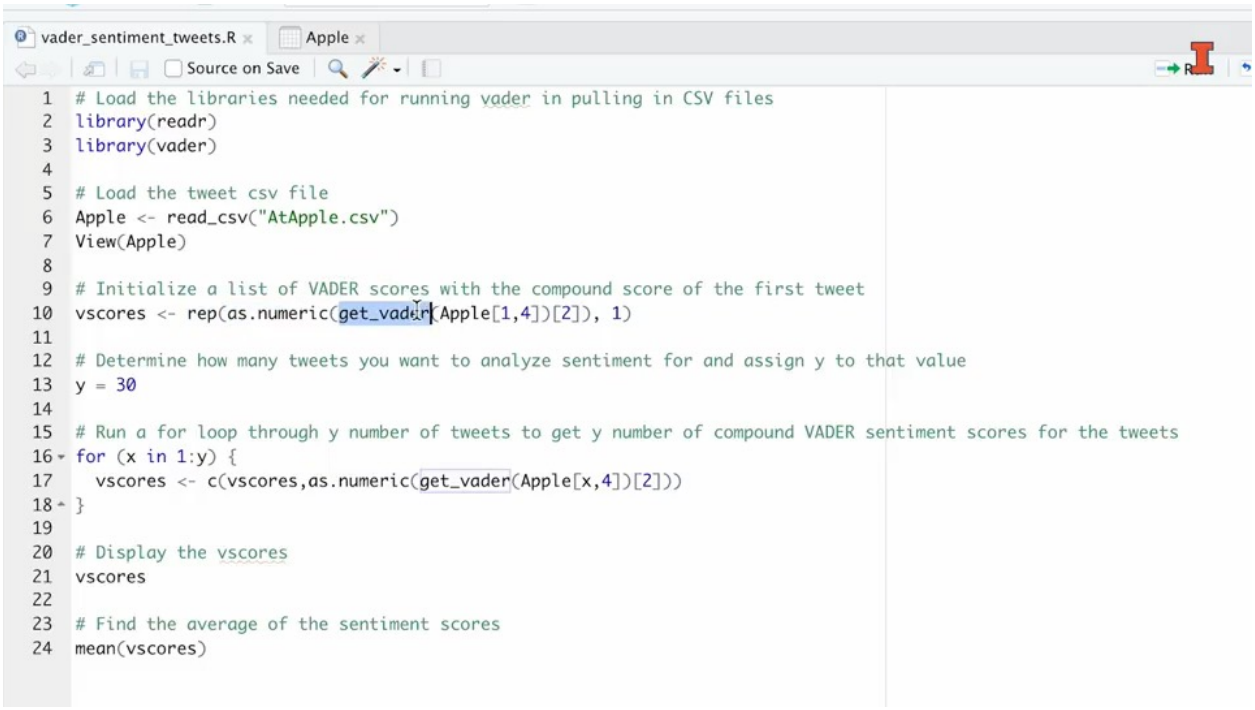


	id	id_str	created_at	text	display_text_range
1	1.387402e+18	1.387402e+18	Wed Apr 28 13:45:00 +0000 2021	New stock token listing: @Apple $AAPL	[0,37]
2	1.387098e+18	1.387098e+18	Tue Apr 27 17:36:18 +0000 2021	petition for a #BTS_Butter emoji @Apple 💜🙏 https:/...	[0,42]
3	1.387505e+18	1.387505e+18	Wed Apr 28 20:31:47 +0000 2021	*breaking* @Apple results: Revenue soared 54% to $8...	[0,236]
4	1.387845e+18	1.387845e+18	Thu Apr 29 19:03:49 +0000 2021	RT @TheWebbyAwards: IT'S NOMINEE TIME! 👑 The n...	[0,140]
5	1.387845e+18	1.387845e+18	Thu Apr 29 19:03:45 +0000 2021	RT @chouchoutv: BREAKING: @Apple will build new $...	[0,140]
6	1.387845e+18	1.387845e+18	Thu Apr 29 19:03:45 +0000 2021	I was thrilled that @apple included these emoji in iOS ...	[0,262]
7	1.387845e+18	1.387845e+18	Thu Apr 29 19:03:25 +0000 2021	@tehlike @Apple I mean it does use likelihood models...	[16,79]
8	1.387845e+18	1.387845e+18	Thu Apr 29 19:02:48 +0000 2021	RT @binance: New stock token listing: @Apple $AAPL	[0,50]
9	1.387845e+18	1.387845e+18	Thu Apr 29 19:02:38 +0000 2021	@hishnash @RichardTaylorTV @Apple M2 only makes ...	[34,234]
10	1.387845e+18	1.387845e+18	Thu Apr 29 19:02:13 +0000 2021	@andytoe @santanderuk @Google @Apple That may ...	[37,217]
11	1.387845e+18	1.387845e+18	Thu Apr 29 19:02:01 +0000 2021	I cannot believe how badly @Apple fucked up the Pod...	[0,66]
12	1.387845e+18	1.387845e+18	Thu Apr 29 19:01:58 +0000 2021	@jayzalowitz @Apple Is that racial profiling on apple's...	[20,64]
13	1.387845e+18	1.387845e+18	Thu Apr 29 19:01:31 +0000 2021	RT @joelbNPRCC: Applications are now open for Holb...	[0,140]
14	1.387844e+18	1.387844e+18	Thu Apr 29 19:01:27 +0000 2021	Wow @Apple has some really bad customer service. ...	[0,104]
15	1.387844e+18	1.387844e+18	Thu Apr 29 19:01:24 +0000 2021	RT @binance: New stock token listing: @Apple $AAPL	[0,50]
16	1.387844e+18	1.387844e+18	Thu Apr 29 19:00:24 +0000 2021	"@Spotify launched its #podcast #subscription service...	[0,278]
17	1.387844e+18	1.387844e+18	Thu Apr 29 18:59:52 +0000 2021	Dear @Apple why not put a #smarttag on the #applet...	[0,120]

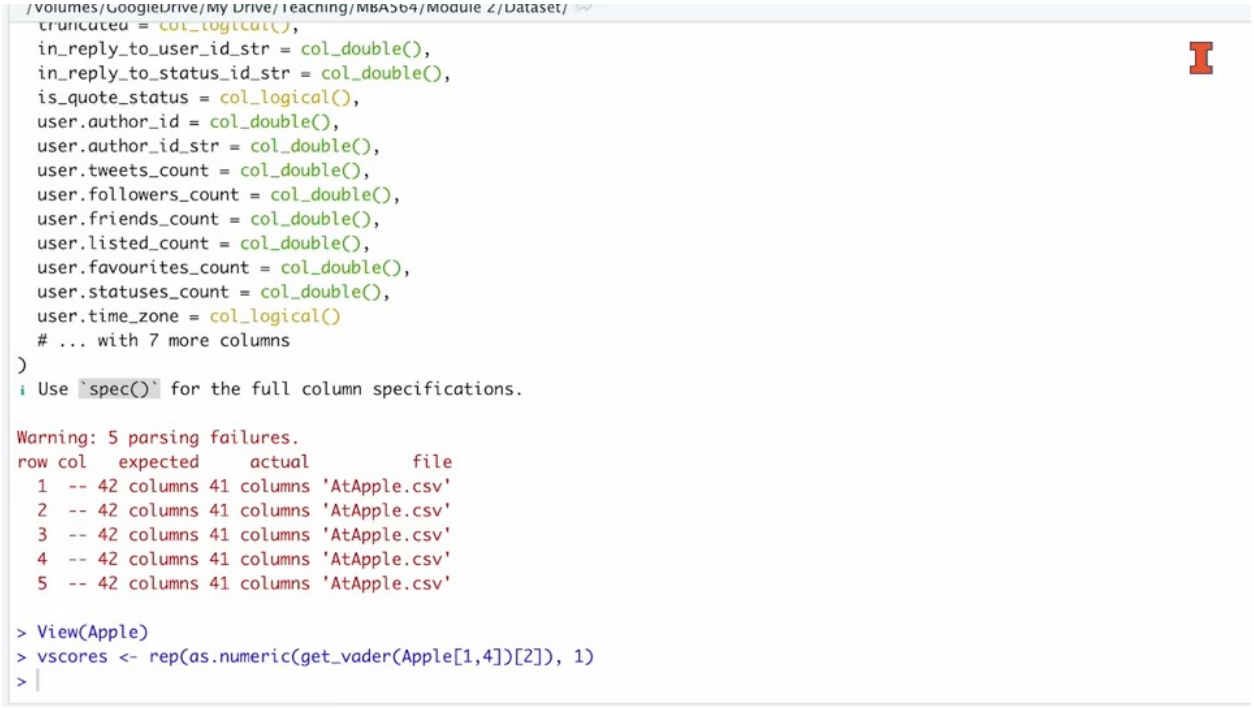
Let's run it. You can see here, here's some tweets, this fourth column right here.



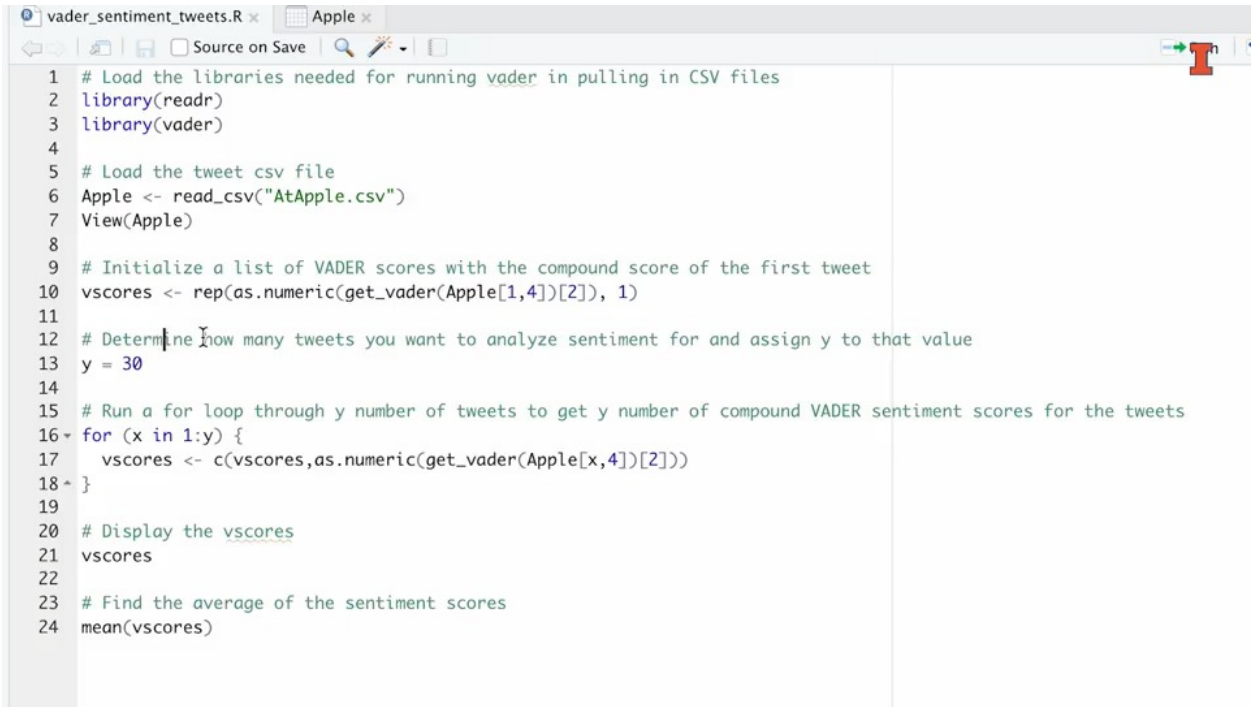
Now that it's loaded, I'd like to initialize a list of Vader scores with the compound score, but I'm just going to use the compound score of the first tweet so that I can initialize a list and have one value within that list. That list is going to be called vscores, or that's short for Vader scores. You can call it whatever you want, but I'm just calling it vscores.



What I'm putting into that is a list representation value as numeric because I want the value to be a number from this get_vader function, and this get_vader function is going to look into my data frame Apple. It's going to look at the first row of the fourth column. That fourth column is a tweet and within this value or the variable that is returned after you run get_vader on Apple 1,4 what you'll actually get is an object that has various scores. It has the positivity score, that neutrality score, that negativity score, and the compound score. What's going on here is that I want the second item, which is the compound score from the get_vader results of running Apple's first row, fourth column. Basically, I just want the compound score of the first tweet that's within the Apple data frame.

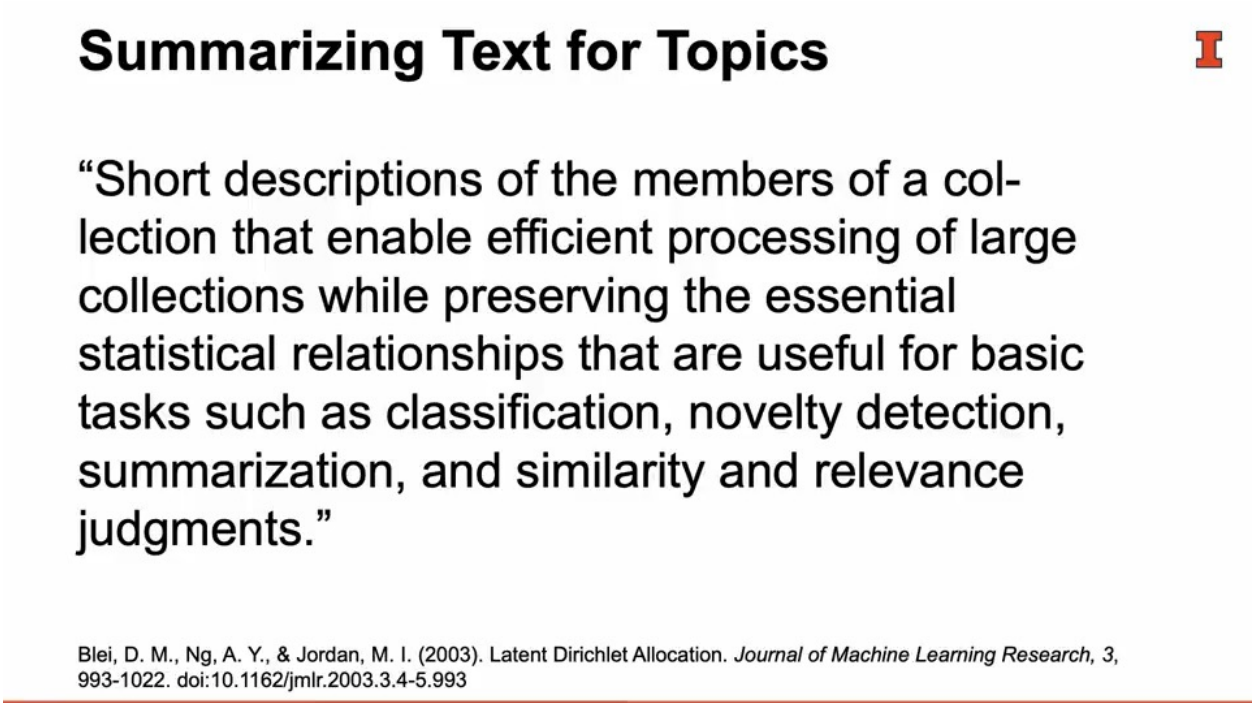


I'm going to run this and I want to store that within vscores. I run it. Once I've run it, I can actually, and this isn't in your R script, but if I wanted to see what's in vscores, I can click on ''vscores'' and what it shows is zero. Now, if this run correctly, what that means is that the first tweet when it runs through Vader, it has a compound score of zero, meaning it's just neutral. Let's make sure that it ran correctly by running some more.



You could run, get_vader across all of the tweets that are in the CSV, but that's going to take an unbelievably long time because there's a lot of tweets that were pulled from Twitter. What I've decided to do with this script is that you can just define a number of tweets, just to test things out as to how many tweets you want to process Vader scores to look at. This variable here, y set to 30, means that we're just going to look at 30 tweets. We're going to run 30 tweets through Vader. Then here's a full loop that is going to run each one of those tweets, and for each of the compounds scores, we're going to store it into vscores. We're going to append that to vscores. Let me run this chunk here. Now let's display the vscores, and if everything is correct, we should see varied scores and we do down here. Now, after running 30 tweets of people talking about @Apple, we can run an average on those compound's sentiment scores by just doing a mean of vscores here. What we find as the result is the result is 0.12, which is slightly positive. What we could say, at least with regards to these 30 results, that when we run vader scores, the customer satisfaction, at least the sentiment scores that are a proxy for customer satisfaction are slightly positive because they have a score of 0.12. All of this is one way or an illustration of how you can use sentiment scores via social media data, namely tweets, to get at customer satisfaction.

Introduction to Text Summarization



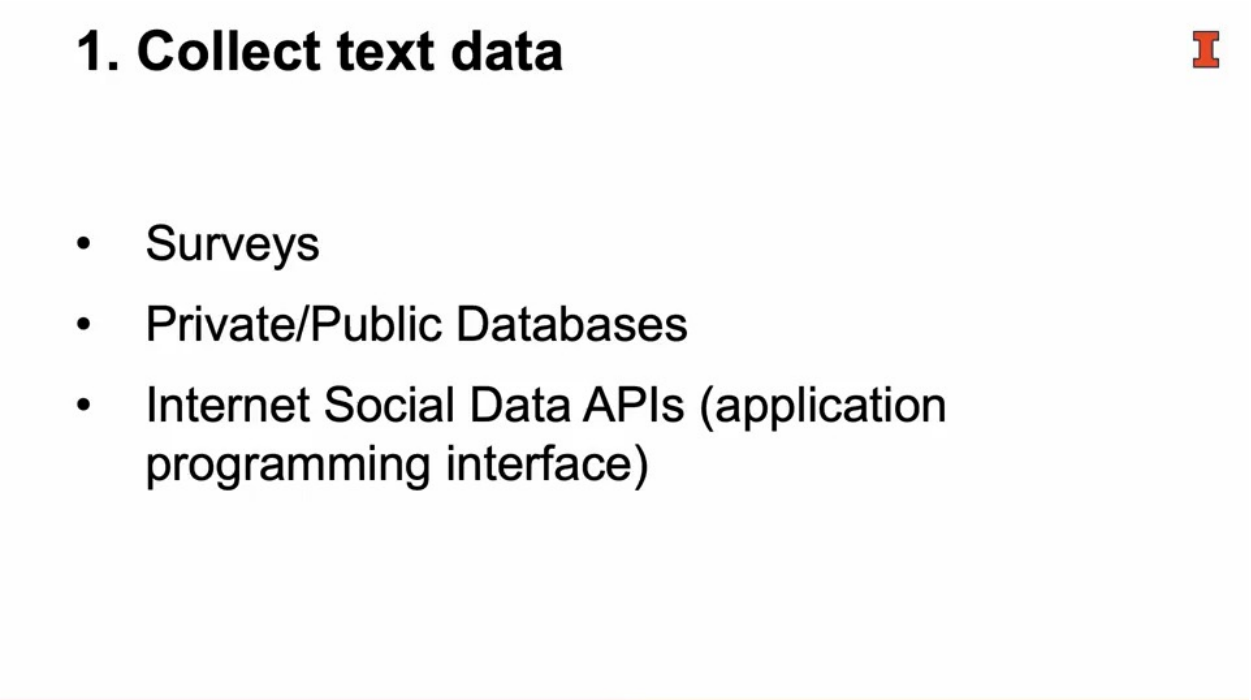
What we're going to do now is we're going to look at text summarization. Specifically what we're doing is we're going over and introduction to text analytics, specifically trying to use computational methods to analyze what people are talking about in aggregate with regards to their text. Now please remember this is just an introduction and so we'll be doing a very high level overview of many of these methods. Just to start with the description, I'm not going to read this whole definition completely, but what we can see in this definition is that we're taking short members of a collection of texts, and we're trying to process it computationally so that we can understand what is being discussed. This can be used for various additional techniques not just with regard to summarization, but you can take these aggregations of text and do things like text classification, novelty detection, so on and so forth.

Text Pre-Processing

Text preprocessing can be defined as bringing "your text into a form that is predictable and analyzable for your task".

Ganesan, K. (2019). All you need to know about text preprocessing for NLP and Machine Learning. KDnuggets. Retrieved from https://www.kdnuggets.com/2019/04/text-preprocessing-nlp-machine-learning.html

Texts preprocessing, which is the first and initial method with regards to analyzing text, can generally be defined as bringing your text into a form that is predictable and analyzable for your task. Here are the steps to preprocess text. What I mean by preprocessing text is that when a computer reads any given body of text that is created by humans, it has to go through a series of steps to prepare the text to be processed computationally. This in itself is a form of text summarization or trying to understand what is being said in the text. I would say it's like a primary or foundational way to get an understanding of the text while using computational methods. There are generally five steps to text preprocessing. First, we collect the text data. Second, we tokenize the text data. Third, we normalize the tokens and I'll go over what it means by token. Forth, we remove stop words, and fifth, we stem and lemmatize those tokens. First, we need to collect the text data.



Now in this course, we've already gone over collecting data from surveys. How you would collect text data from surveys is that you would have some sort of open-ended question in your survey mechanism that would ask something like, do you have any additional comments or how did you feel about this product or what do you think about this brand? Then you give them an open textbox so that they can write that text into the survey response. But you can also obviously collect text data from databases, whether they're private databases within your company or public databases that are available on the Internet, as well as another popular way to collect text data is to collect internet social data. Social media data via what is called an application programming interface. Now I'm not going to go too much into that in this course, but it's basically the way that a social media company would enable you to pull data from them. That's not just text data, it could be image data, it could be whatever they allow you to pull from them. Now this is not an exhaustive list of how to collect text data, but this is three of the more popular ways to collect text data for marketing purposes. The second step is then to tokenize the text. Here is where I'll explain what tokenize means or this concept of tokens. For a computer, a token is basically breaking down text in any way that will enable it to process batches or items. So the computer calls it tokens. Now we could call words tokens because most of the time tokenization is just applied as, let's just do it word for word. To give you an example, let's start with this sentence here. I love Fridays and hate Mondays, but this Monday I turned 21, exclamation point.

2. Tokenize the text

Sentence: I love Fridays and hate Mondays, but this Monday I turn 21!

White-space word tokenization: “I” “love” “Fridays” “and” “hate” “Mondays,” “but” “this” “Monday” “I” “turn” “21!”

If we use whitespace word tokenization, which means that the computer will count a token after each whitespace, then what you have on this slide here is I is a token, love is a token, Fridays is a token, and so on and so forth. You can see that for our example here at the tokens are just simply words that are separated by whitespace. After you've tokenized the text, now you normalize the tokens.

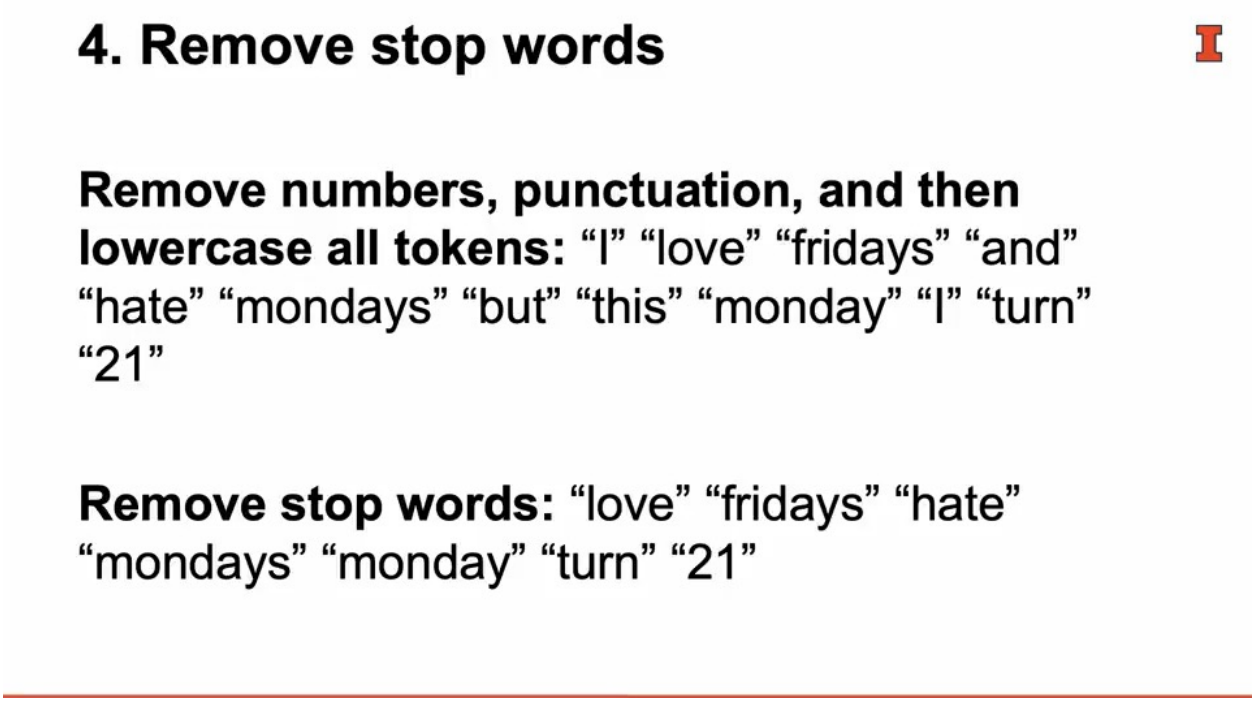
3. Normalize the tokens

White-space word tokenization: “I” “love” “Fridays” “and” “hate” “Mondays,” “but” “this” “Monday” “I” “turn” “21!”

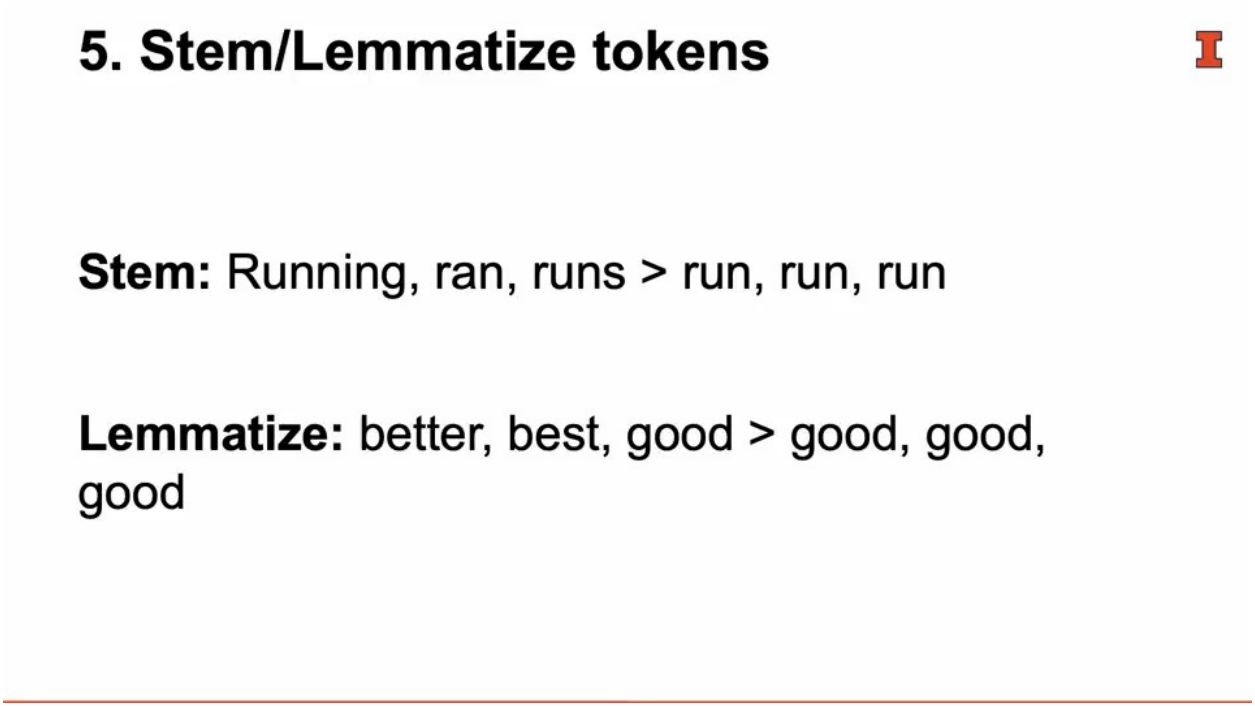
Remove numbers, punctuation, and then lowercase all tokens: “I” “love” “fridays” “and” “hate” “mondays” “but” “this” “monday” “I” “turn” “21”

We start with our whitespace word tokenization at the top of this slide. I, whitespace, love, whitespace, Fridays. Then you go through this process of normalization, which you can actually choose depending on your goals of how you're processing the text how you want to normalize. Many times, people will remove

numbers, they'll remove punctuation, and they'll do lower casing of all the tokens so that everything can be standardized or again, normalized. But I actually highlighted numbers in red here because I noticed in this sentence that that number 21, it's very important to this sentence. It's easy for me to see because we're looking at one sentence. But just so you know, the normalization step, you can make a decision based on your goals as to how you want to normalize. If you look at the normalization on the bottom of the slide, you'll see that punctuation is taken out. After 21, there's no more exclamation point. Everything is lowercased, but we did retain the number 21. Then the fourth step is to remove stop words. I'll just read this definition verbatim. Stop words are the most common words that appear in a text but are no actually important to understanding the topical content of what is being discussed. You see examples here, the word the or with, or to, a, and. Those words usually are not super important to understanding the meaning of a body of text, and they're typically called stop words. Now stop words, there's no defined for sure stop words out there that everybody has to define what their stop words are. This is another chance for a data scientist, you all, to make a decision as to what do you want your corpus of stop words to be? Now of course, there are very popular stop word aggregations on the internet that you can just use, and that's what most people do. But just so you know that stop words are not necessarily predefined and set in stone.

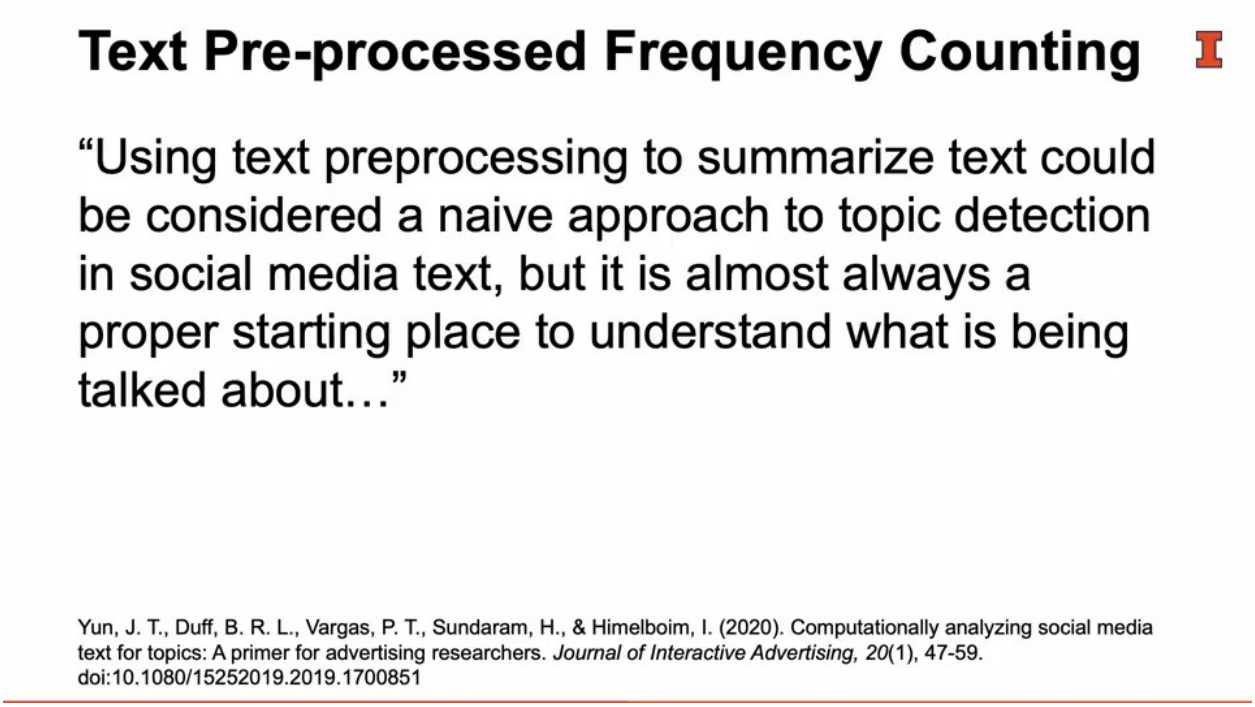


If we remove stop words, you can see that what we started with at the top of this slide after we normalized the tokens, that we removed the stop words at the bottom of this slide, and now we're left with love, fridays, hate mondays, monday, turn, 21. Then the final step is to stem or lemmatized the tokens. Well, what is stemming? You can see these examples here.



You may have various forms of the word run: running, ran, runs. So what stemming does is it will convert all three of those various forums of the word run to just run so that you know that you have three instances of some form of run. Lemmatize is actually taking stemming one step further, where you'll have these words: better, best, good. A computer actually will know through a lemmatization process that all of these are actually meaning the root word, good. Stemming would not get you to good three times from these three words and better, best and good, but lemmatization will do that. Again, this is a decision that you make as to whether you want to stem, or whether you want to lemmatize, or in some senses, you might want to combine both, but that's not necessarily what most people do. They'll make a decision to either stem or lemmatize. I've chosen stemming here. Where we started with love, fridays, hate, mondays, monday, turn, 21, now we have love, friday, hate, monday, monday. You see that now there instead of a Mondays plural and a Monday singular, now we just have to Monday singular, monday, monday, turn, 21.

N-gram Frequency Count and Phrase Mining



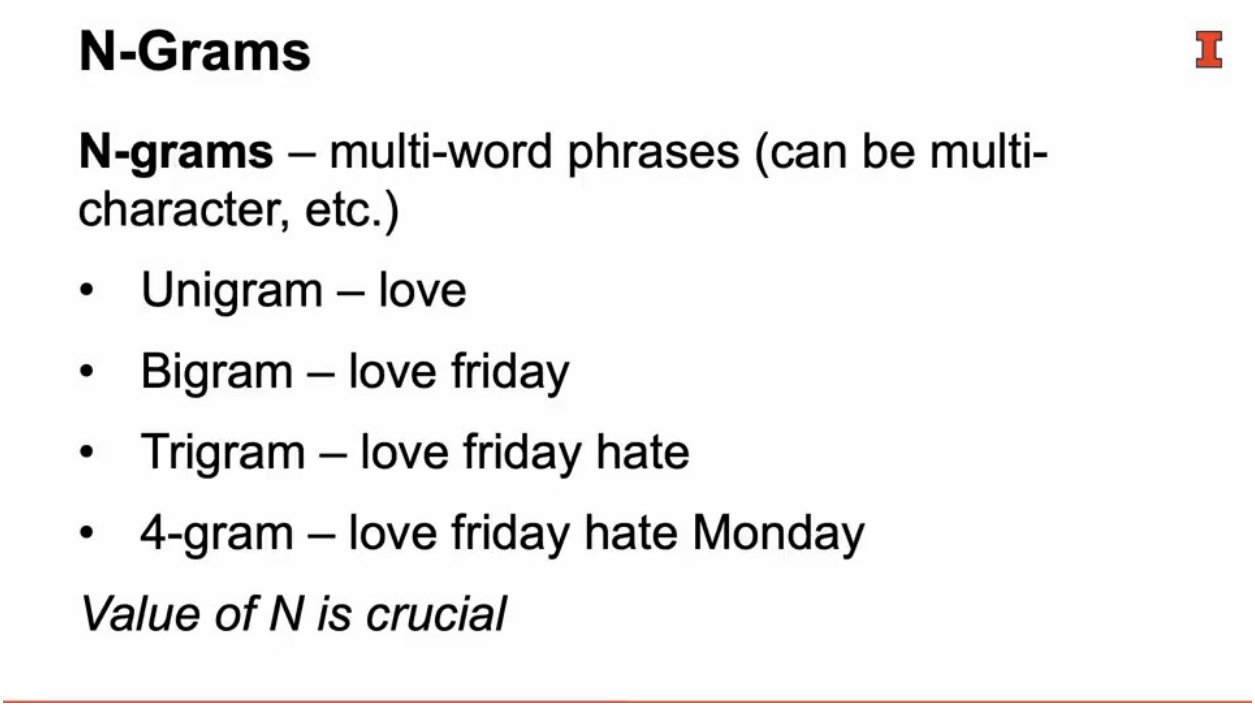
Okay, what will go through next is various methods of text summarization. I had already mentioned in the first lecture using text pre processing as a method of text summarization. But now we'll go into n-gram frequency count which is building on top of the tax free processing as well as we'll go through a concept called phrase mining. So text pre processed frequency counting which is basically taking what we've done previously and just counting the frequency of the words, it could be considered a way of topic detection. Lb it it's a naive approach but it's almost always a proper starting point to understand what is being talked about. And so let's go back to our first example from the first lesson which we had a resulting set of words from pre processing which were love, friday, hate, monday, monday turn, 21.

Text Pre-processed Frequency Counting

"Using text preprocessing to summarize text could be considered a naive approach to topic detection in social media text, but it is almost always a proper starting place to understand what is being talked about…"

Yun, J. T., Duff, B. R. L., Vargas, P. T., Sundaram, H., & Himelboim, I. (2020). Computationally analyzing social media text for topics: A primer for advertising researchers. *Journal of Interactive Advertising, 20*(1), 47-59. doi:10.1080/15252019.2019.1700851

And so the resulting frequency counts come out to be monday is 2 love is 1 friday is 1 so on and so forth. Now you look at this list and let's pretend that somehow we don't remember the first sentence that started all of this. If you look at this list and you see Monday two love Friday hate turn 21 you might be able to piece together. I think they're talking something about days of the week and I'm guessing they're probably thinking hate monday because we all hate Mondays. And then something about 21 maybe a birthday because you see the word turn, you can kind of see how just from this frequency count you start to piece together what is being discussed. But it's not perfect and I guess spoiler alert, no text analytics method at this point is perfect. And so we're just going to have to learn various techniques and try to see what works best for any given situation. But this is pretty simple. We take what we had from pre processing and then we just count the frequency of the tokens which in this case are just words.



Now there is this concept of n-grams which is basically remember I said that the tokens that we chose previously were just white space words. But n-grams is a concept where you can take multi word tokens or rather instead of doing whitespace tokenization you can do some sort of algorithm that will do every other white space. And then you're going to end up with these tokens that are kind of two word phrases, right? And so you see examples here where unigram one g is love but bigram which is two gram is love friday trigram, love friday hate, 4-gram love friday hate Monday. And these would be the tokens that you choose. And so the value of N if you're using n-grams is crucial because it really does affect how you're going to summarize the text in general. Now you can kind of see the difficulty of the n-gram technique even though it's a very widely used technique to tokenize and to look at pre processed text. But the problem with the n-gram technique is that it's really important how you choose N. And sometimes a number of N will work for a certain set of words but it won't work for another set of words. So the us we now have this advanced technique and this is really cutting edge called phrase mining. I'll read this definition verbatim because this work is really in my mind, incredibly important. Phrase mining refers to the process of automatic extraction of high-quality phrases, for example, scientific terms and general entity names in a given corpus for example, research papers and news.

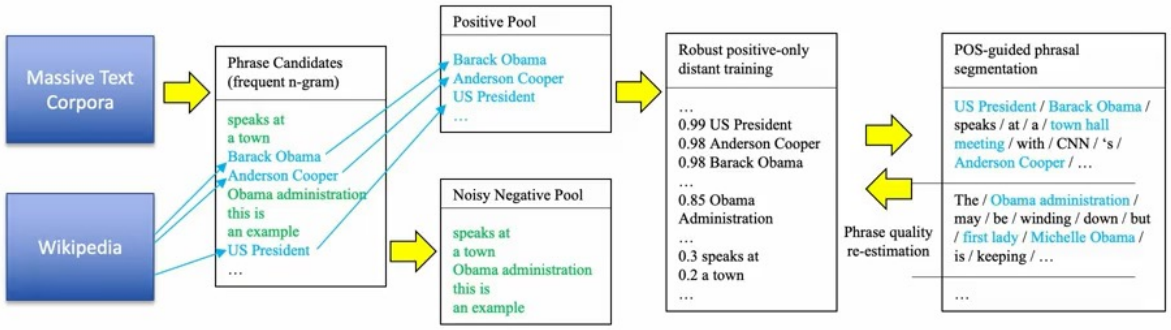
Phrase Mining

"Phrase mining refers to the process of automatic extraction of high-quality phrases (e.g., scientific terms and general entity names) in a given corpus (e.g., research papers and news). Representing the text with quality phrases instead of n-grams can improve computational models for applications such as information extraction/retrieval, taxonomy construction, and topic modeling."

Shang, J., Liu, J., Jiang, M., Ren, X., Voss, C. R., & Han, J. (2018). Automated phrase mining from massive text corpora. *IEEE Transactions on Knowledge and Data Engineering, 30*(10), 1825-1837.

Representing the text with quality phrases instead of n-grams can improve computational models for applications such as information extraction, retrieval, taxonomy, construction and topic modeling. I do want to mention that this is work from Dr Shang when he was a PhD student here at the University of Illinois at Urbana Champagne. But this comes out of the way Hans lab and it's very cutting edge work basically trying to figure out instead of just using the set number N n-grams. Can we have a more intelligent method that can use various forms of N or various values of N to find quality phrases.

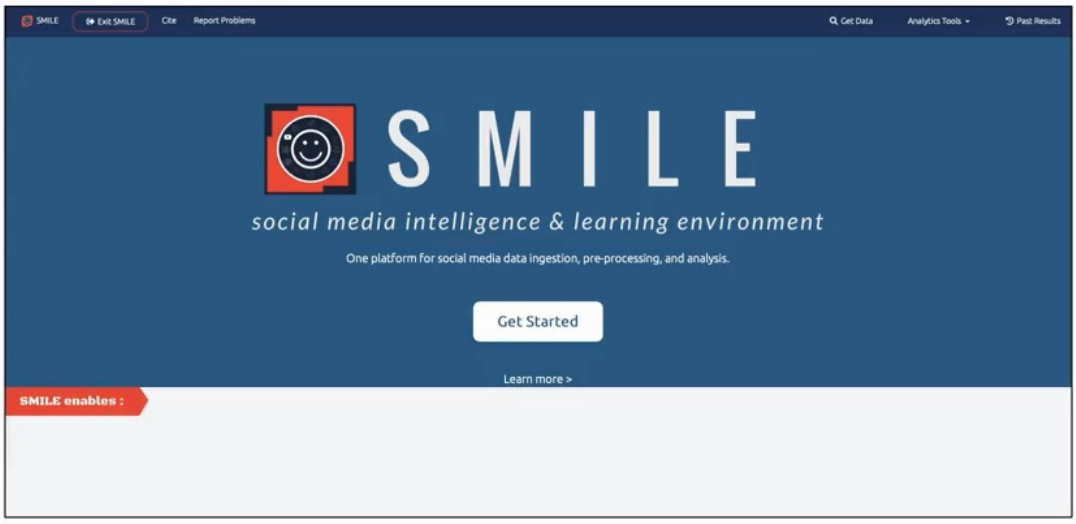
Phrase Mining



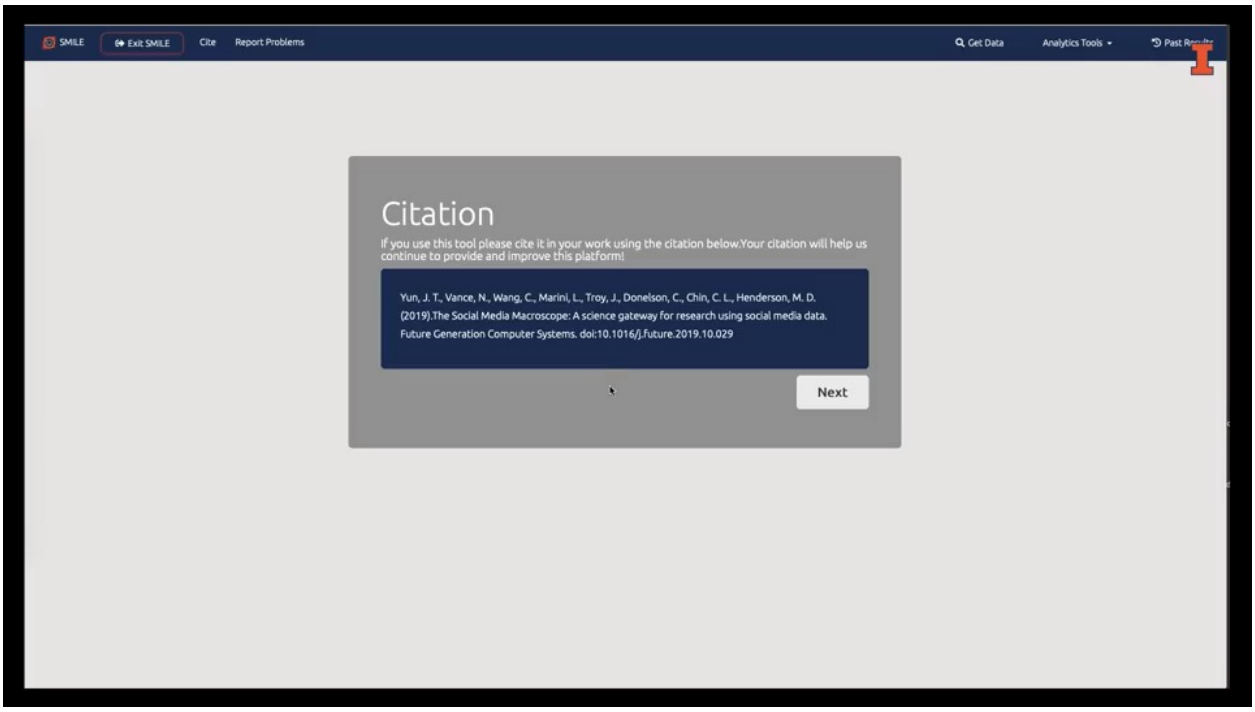
Shang, J., Liu, J., Jiang, M., Ren, X., Voss, C. R., & Han, J. (2018). Automated phrase mining from massive text corpora. *IEEE Transactions on Knowledge and Data Engineering, 30*(10), 1825-1837.
Image taken from open paper: https://arxiv.org/abs/1702.04457

Now this figure looks a bit complicated, but I'm just going to walk us through this because this is basically how Dr Shang's phrase mining works. So on the left of this figure, you have two boxes, you have massive text corpora and you have Wikipedia. Now, let's start first with massive text corpora which is basically you have a huge body of text. Let's for example, take a bunch of news articles. So you have a bunch of news articles that are just a massive bunch of text and then you have Wikipedia. Now this is one of the advancements that Dr Shang brought forward in automated phrase mining, which is the fact that. We have this crowd sourced body of topics really, which is called Wikipedia right where people have created Wikipedia entries that could be a topic or a person so for example, Barack Obama. There is a Wikipedia entry for Barack Obama which has a bunch of definition or description about Barack Obama. And so this thought was how we can try to detect what our quality phrases is to look at what are the titles of articles on Wikipedia. And this was a major intuitive advancement. And so you could take the various phrases that are coming out of the massive text corpora which we see in this second box called phrase candidates frequent n-grams. And these are n-grams that could be of differing sizes. This is where your computational power has to be pretty high because it's kind of calculating n-grams via various values of N. And so you have these words speaks at or I'll just kind of skip around Barack Obama, Anderson Cooper, US President. And basically you'll find that some of these n-grams have Wikipedia entries and some of them don't. And so the ones that do you kind of wake them up saying this is a bit more of a potentially quality phrase. And so in the middle of this figure you have a positive pool box in a noisy negative pool box. And so the quality phrases you put into that positive toolbox Barack Obama, Anderson Cooper, US President. And then the noisy negative pool you have speaks at a town Obama administration, which Obama administration is kind of an interesting example. Because that seems like a quality phrase but it might not necessarily have a Wikipedia entry. And so there is a limitation there is that maybe that's a problem with this method. But again, every method has the pros and cons and then we'll stop right there. And I'm going to jump to the right side of this figure which is the box that says POS-guided phrases segmentation. Which is part of speech guided phrasal segmentation which is taking the actual phrases, actual sentences from let's say these news articles. So the first one is US President Barack Obama speaks at a town hall meeting with CNN's Anderson Cooper. And what computational is being done here is a method that is very common within data science which is there is what are called part of speech taggers POS taggers. And they can identify within a sentence. What's the noun? What's the verb, what's the adjective etc. And so what phrase mining is doing is it's now on this side without looking at Wikipedia. It's segmenting parts of speech and then combining what it finds namely it's going to wait now is very high. So US President Barack Obama Anderson Cooper. And it's going to combine mathematically and we won't go to into the mathematics of this but combine that with the positive pool words that are Wikipedia entries. And then it's going to give a confidence score which is right at the middle or the kind of right middle of this figure in the box called robust positive only distant training. Where it will say with great confidence 0.9999% confidence we believe us president is a quality phrase 98% confidence Anderson Cooper is a quality phrase so on and so forth. Whereas with 30% confidence we think speaks at is a quality phrase or 0.2 or 20% confidence a town is a quality phrase. And you can see right here through this phrase mining exercise which is again very cutting edge that you get a much better picture as compared to choosing a random value of N and using n-grams.

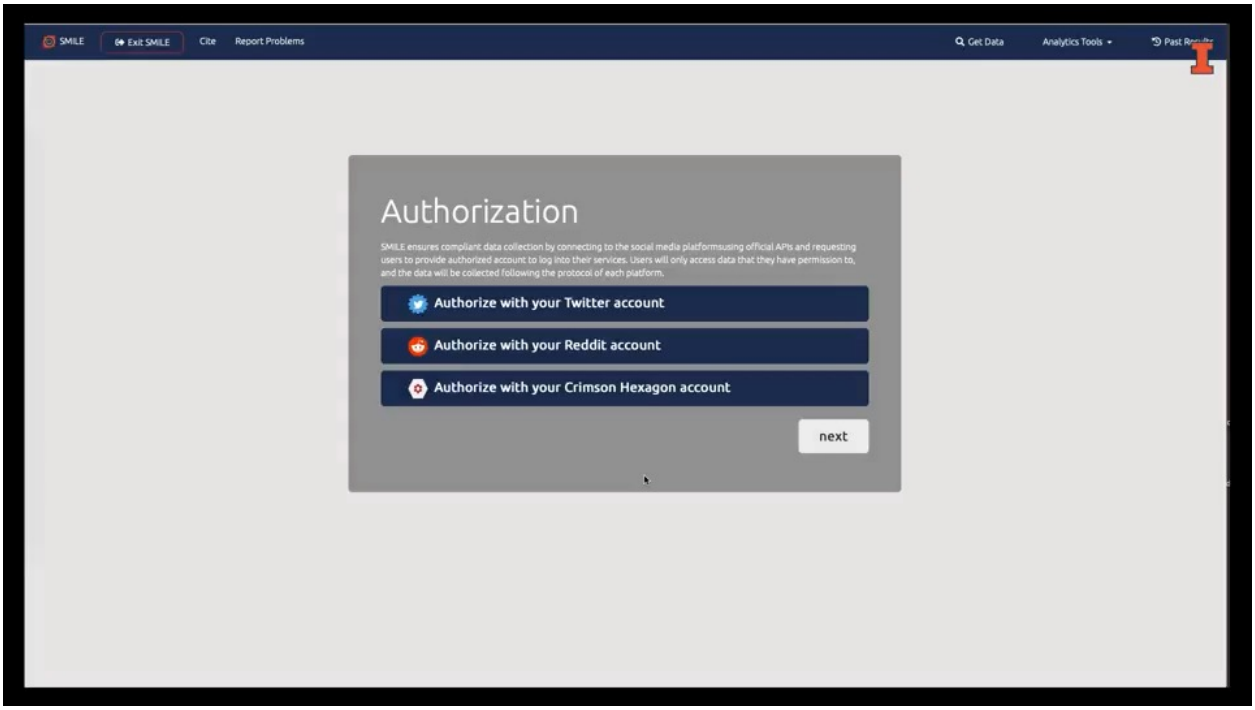
SMILE (within the SMM)



But rather using this method to figure out what is actually being discussed in this body of news articles. So now what we'll do is we'll switch to the social media microscope and specifically its tool called SMILE the social media intelligence and learning environment. And we will actually try text pre processed frequency counting as well as automated phrase mining. >> Now we will see a demonstration of text pre processing as well as phrase mining within the tool SMILE from within the social media microscope. Remember SMILE is the social media intelligence and learning environment. And it is a general social media analytics tool for ingesting in data as well as analyzing that data via data science methods. So the first step is to actually bring in data.

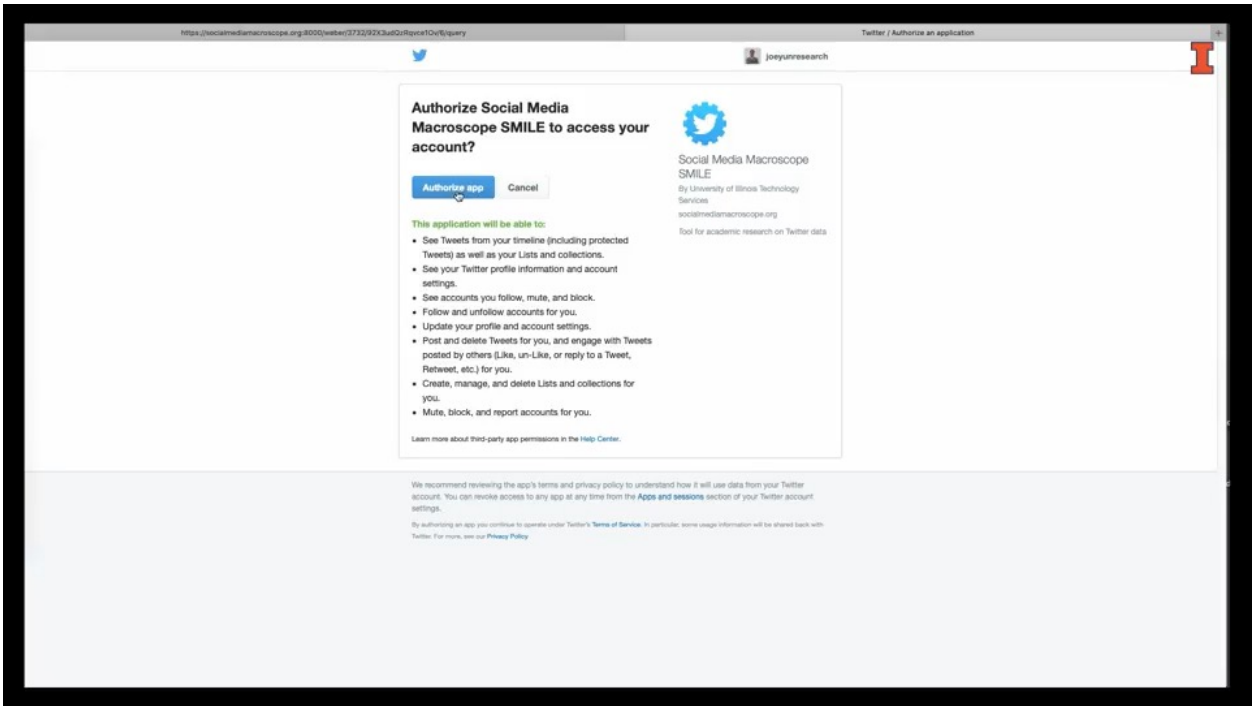


So we'll click on get started here.The screen that we're taking to is a request for citation if you are conducting research using our tools we would greatly appreciate that citation but we click on next. And the first thing that we need to do is bring in some data. Now there's various ways to pull in social data into SMLE one would be via public API and we've got three options here or two options really that our public.



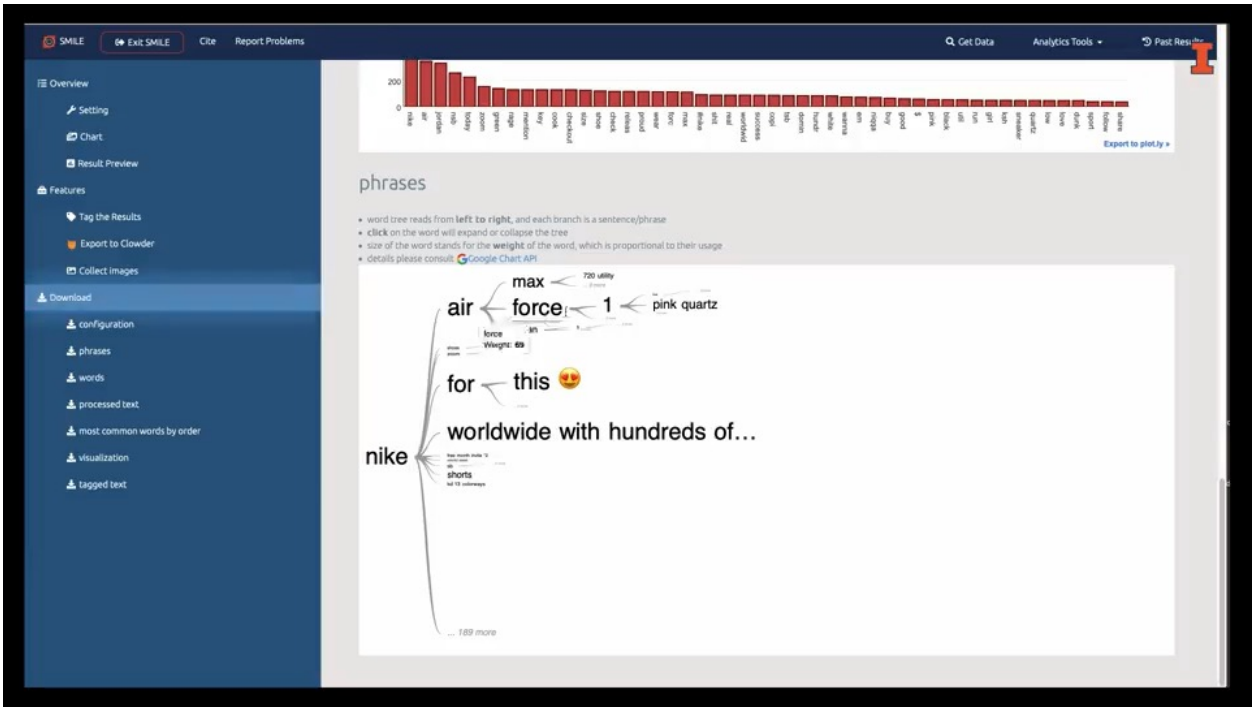
You can pull in via the Twitter public API you can pull in via the Reddit API or if you have a account with a company like crimson hexagon. You could use their private API to pull in data or if you have a CSV with your own data you can use that to import data into SMILE. Please note that once your SMILE session is closed all

the data that you have pulled in will be deleted. So you can rest assured that any outside external data that you bring in via CSV will be securely deleted once you are done using SMILE. So for this demonstration we will authorize with our Twitter account. And one of the things that we have tried to do is make it simple to connect to various public APIs. And so with regards to Twitter you can see a screen like this.

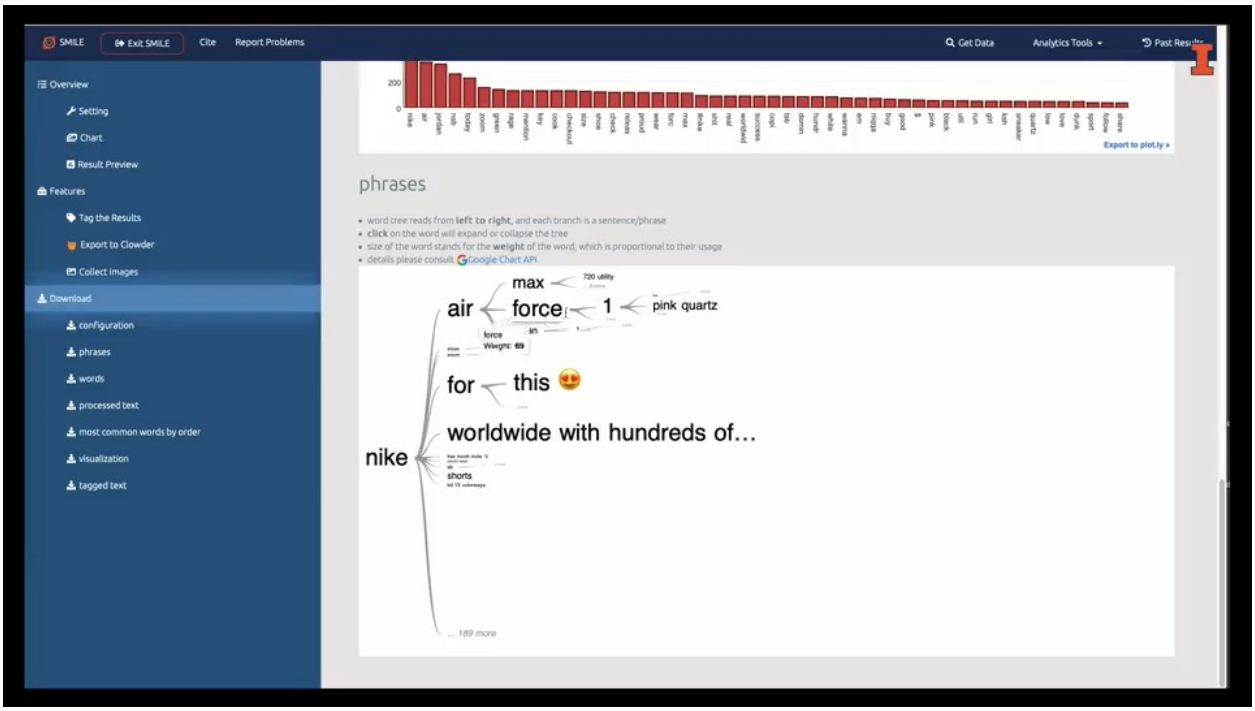


I'm already logged into my twitter account which is joeyunnresearch and it's basically asking me to authorize the app of SMILE to connect to my Twitter account. I will do that. It gives me a key or a pin. I take that pin back into SMILE and submit it. And I am now successfully authorized to pull via Twitter's public API. Now I'll just go with that for now and click on Next and then I'm taken to the search screen for SMILE in which I can select tweets. And say I would like to pull recent tweets from a brand let's say in this case Nike which I've already pulled previously but I will run through this just as an example click on Search. And what this is doing is it will give us a preview of some of the current tweets that are trending for Nike or currently that are being tweeted with regard to the keyword Nike. And then I can save it using a file name such as Nike and click on Save and then that data set will be pulled. Now I've already pulled that data set for time's sake. And so what I can do next is I can go to analytics tools here and I can click on natural language pre processing. Since that's the first technique that we want to go over in this video. I can select my data set which I already previously saved, that's why you see it here Nike. Let me close out this Twitter authorization screen and you'll see some of the preview tweets that are regarding Nike. And you can see right here actually if you wanted to import your own file to do text pre processing you could do it right here but we've got our Nike set that we've already pulled. Then I want to use the pre processed step of stemming. And then I'm going to select currently the only option which is the natural language toolkit part of speech tagger. And again, what is different about the social media microscope is that all the tools are open meaning various things actually. But one of the things that it means when it says that we are open is that all of our methods. They'll be associated papers in which you could click on these links and you can find out more information about how that method works. So you can find specific information from this paper about NLTK. But I will click on Submit and it is going through the tweet data set of Nike that I had previously pulled. And then it is stemming and doing various tokenization techniques and presenting to us a frequency chart. Now I can tag the results and this is throughout the system, I'll say Nike and I'll say pre

processing and I'll tag that result. But now I can scroll down and I can see that the most common tokens that are being discussed for Nike or Nike air Jordan MSNSB, today zoom green etc.



This tool also has a word tree that you could actually interact with and then you can see how these basically phrases or tweets play out through interaction with this word tree. So that's how you run text preprocessing, our natural language process preprocessing via the SMILE tool. Now I will show you how to run automated phrase mining from SMILE. So you'll go to analytics tools just like I just did and click on automated phrase mining and from here we'll go through a very similar process that we just followed where you will click on Nike. And then with regards to this method which is outlined here and the associated paper from Dr Shang is right here you have to select a parameter called minimum support. And basically what this parameter is how many times does a phrase have to occur in the corpus or in the data set of tweets before is even considered for this list of phrases. And so we'll set that threshold somewhat low, we'll say 10 here and I will click on Submit when I click on Submit, it's going to actually ask for my email address. Because this phrase mining takes quite a bit of time to process and then once it is done processing then I will be sent an email that says that. Your phrase mining exercise or algorithm is now complete and you can check the results. So I've already run this.



So click on past results here at the top right I will click here on automated phrase mining and you'll see right here that I ran auto phrase on Nike with a minimum support value of 10. So if I click on this. You'll see the results well and I did not mention this in the previous preprocessing step but you can actually download various aspects of all of these results for all the algorithms withinSMILE from here. But what you see here quickly is a visualization of some of the phrases as well as a preview of the highest phrases that are part of this data set. General Motors, air Jordan, air max, released state, style code Nike SB dunk low so on and so forth. But you can download the complete phrase listing right here or right here, a complete list of phrase extracted so on and so forth. So this is how you use SMILE to conduct text preprocessing as well as phrase mining. Thank you.

LDA Topic Modeling

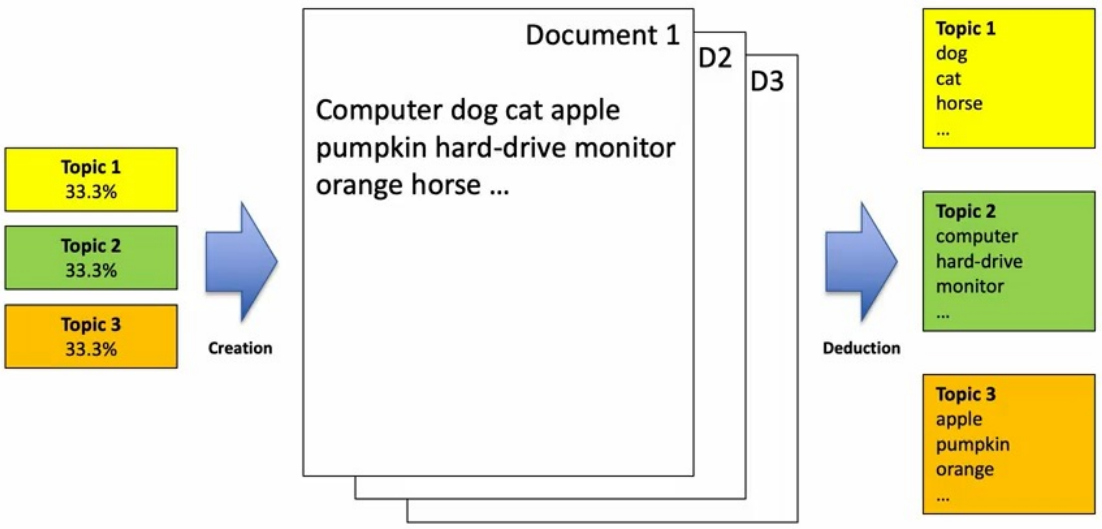
LDA Topic Modeling

"Topic modeling algorithms are statistical methods that analyze the words of the original texts to discover the themes that run through them, how those themes are connected to each other, and how they change over time."

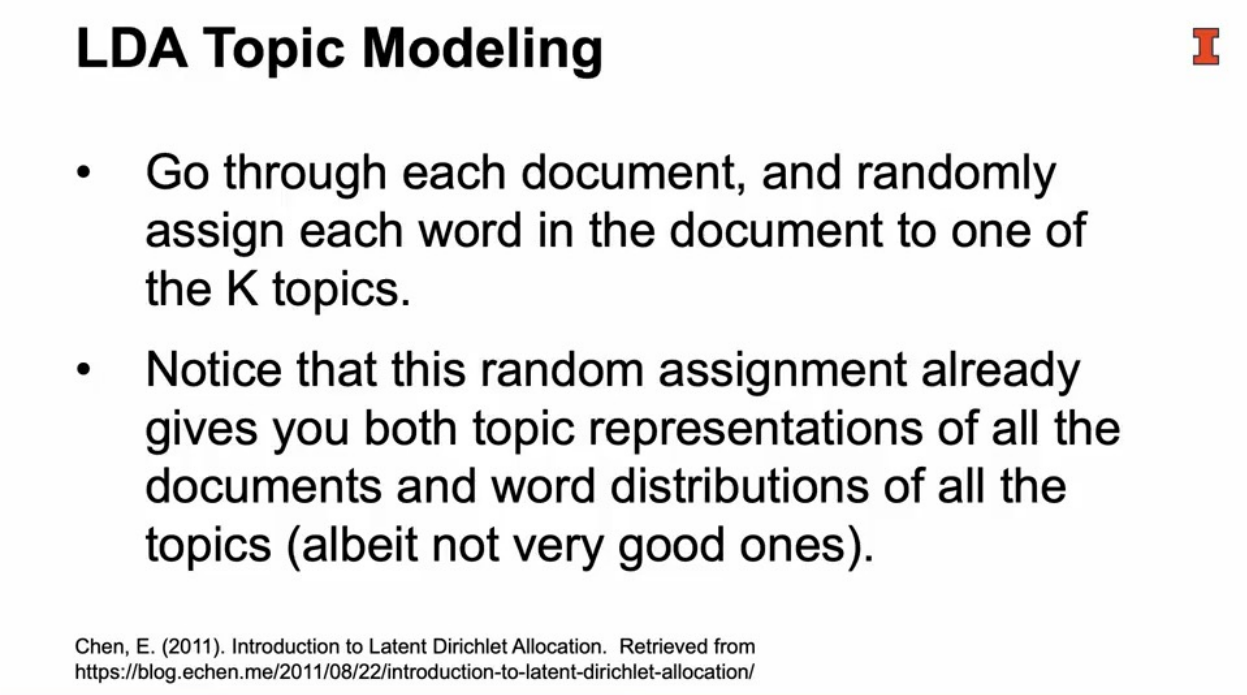
Blei, D. M. (2012). Probabilistic topic models. *Communications of the ACM, 55*, 77-84. doi:10.1109/MSP.2010.938079

Our next topic within text summarization is a very popular way to summarize texts or to detect topics in a body of texts, which is called LDA Topic Modeling. Now, this is from Dr. Blei. Topic Modeling algorithms are statistical methods that analyze the words of the original texts to discover the themes that run through them, how those themes are connected to each other, and how they change over time. Really, Topic Modeling was one of the first accepted or popularized ways to analyze topics of conversation or topics within text in a computational format.

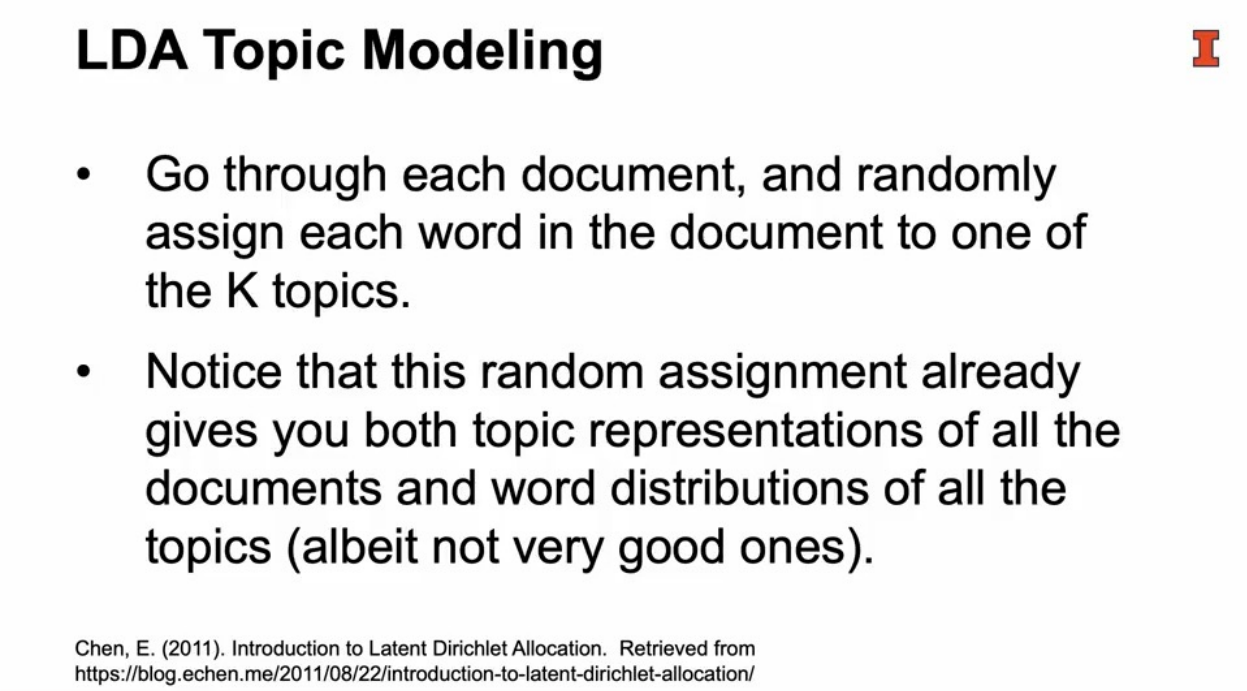
LDA Topic Modeling



What I'm going to try to do from this figure is explained at a high level how LDA Topic Modeling works. I believe the best way to explain how it works is to go backwards as to how a computer envisions how documents are written. Let me say that again. How we're going to explain that is to think about if a computer was writing these documents, how would it be writing these documents when we're considering the LDA Topic Modeling framework? What you see here, both on the left-hand side and the right-hand side are topic boxes. On the right-hand side you can see the words that are included in this topic boxes and then on the left-hand side, you can see the percentages upon which the computer believes that any one of these topics should make up in the documents themselves. For Topic 1 on the right side, you see dog, cat, horse, and so on the left-hand side you see Topic 1 as 33.3 percent. Basically, the computer when it creates these documents it's going to make sure that one-third of the time you're going to see words such as dog, cat, horse, and so on and so forth from Topic 1. Topic 2; computer, hard drive, monitor and then on the left-hand side, 33.3 percent, so again, a third. You can imagine you see that arrow that says creation on the left-hand side, that a computer starts creating these documents, and in this case we have three documents. You see the layers Document 1, D2, D3, and you see that a computer is writing. Now, this obviously doesn't make sense because these aren't sentences and there are conjunctions, and all these various things. But you just see a combination of words that reflect the percentages for each one of these topics, so you see computer, dog, cat, apple, pumpkin, hard drive monitor, orange, horse in Document 1. Now, if you understand that this is how a computer through an LDA Topic Modeling framework would construct a document, then how LDA Topic Modeling works is that with the computers assumption that this is how documents are created, the computer is going to backtrack and try to figure out, how do all these words and all of these documents that you've now given me come out with regards to these percentages, 33.3 percent, and it could obviously be different percentages, as well as what topics they're a part of? Now, even as I just explained that, I just wanted to mention even for me that all these years of conducting LDA Topic Modeling and knowing how this works, it's still confusing. I would like to say that part of the reason why it's confusing is because computers make these jumps that necessarily we as humans don't naturally think like, and so I would just like to mention that if this is a bit complicated, I understand.

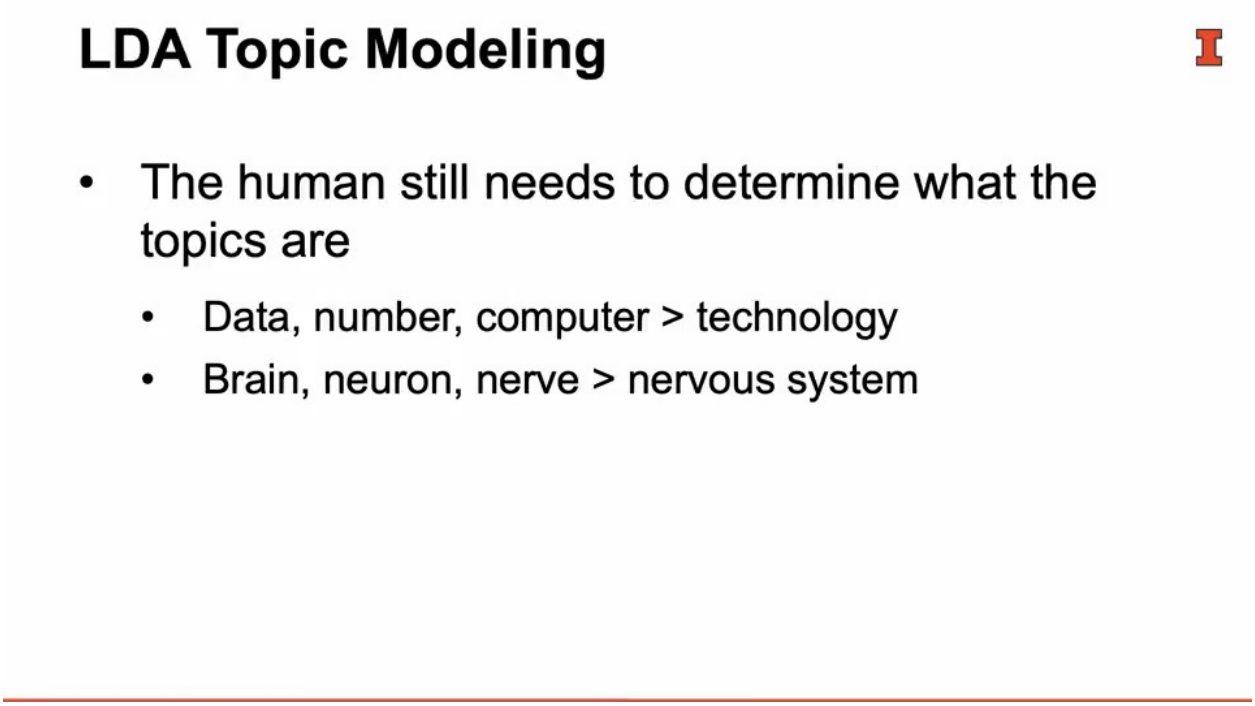


If we assume that this is how computer creates these documents, then the LDA Topic Modeling process is just going backwards, backtracking. The first step in LDA Topic Modeling is to go through each document and randomly assign each word in the document to one of the K topics. Let's just pretend we have three topics like before. For Document 1, Document 2, Document 3 you go word by word and just randomly assign them to Topic 1, Topic 2, Topic 3. Now, you'll see in a second point, notice that this random assignment already gives you both topic representations of all the documents and word distributions of all the topics. We could just stop here, and we would have a horrible calculation of the three different topics and the words that are associated with those topics because we just randomly assigned them. That's why it says, albeit not very good ones. But now, to improve on these calculations, to make sure that the words are actually in the right topics, Topic 1, Topic 2, or Topic 3 for each document, Document 1, Document 2, Document 3. We go through, again, each word. Now, the key is that every time we hit a word, we assume that all the other words are correctly assigned to a topic, and so only the word that we're looking at right now is not correctly assigned.



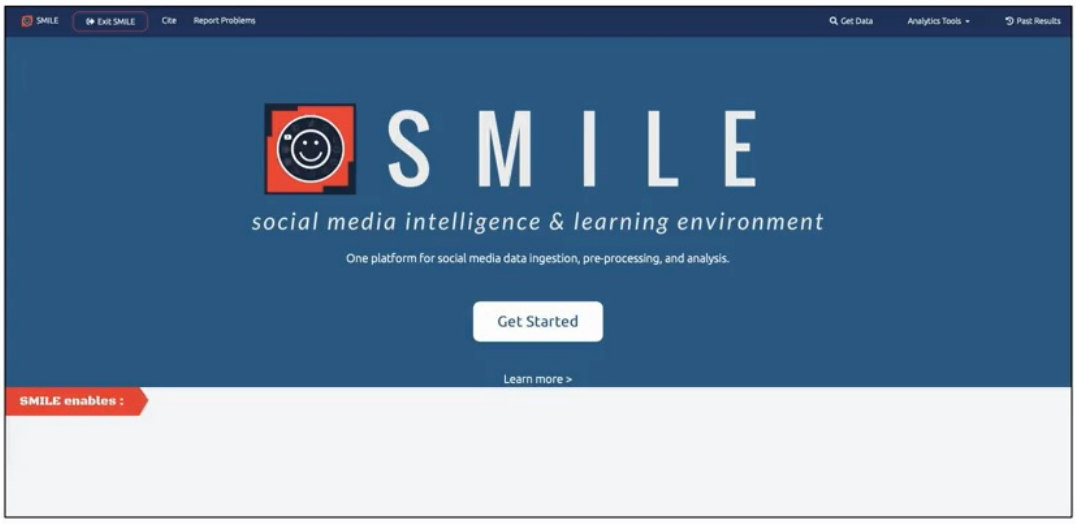
I put the mathematical notations here. You don't have to focus too much on those, but these are what are frequently used. I did want to put them in the slide. But for each topic, we just need to compute two things. We look at with the variable x, the proportion of words in document d, the current document that we are on, that are currently assigned to topic t. What we're asking is, how many of the words that are in the current document are already assigned to Topic 1 or Topic 2 or Topic 3? Then we take for the calculation of variable y, the proportion of assignments to topic t over all the documents that come from this word. The word that we're currently looking at, how many times is that word assigned to Topic 1 or Topic 2 or topic t or Topic 3 across all the documents? Then we do a probability function. X times Y, the probability that topic t, whatever that might be, 1, 2 or 3, generated the word that we are looking at right now. With that probability function, we will then basically assign that word to a new topic or keep it in the current topic. Remember in this previous step, we're assuming that all topic assignments except for the current word in question are correct, and then iteratively updating the assignment of the current word using our model of how documents are generated, what I talked about in the beginning as to how computer would generate these documents. Now this sounds like magic and it sounds like this doesn't even make sense as to how

this works, but what's interesting, and this is what's called actually a Gibbs sampling process. If you want to look a bit more into this, you could do a bit of reading on the mathematical process called Gibbs sampling.



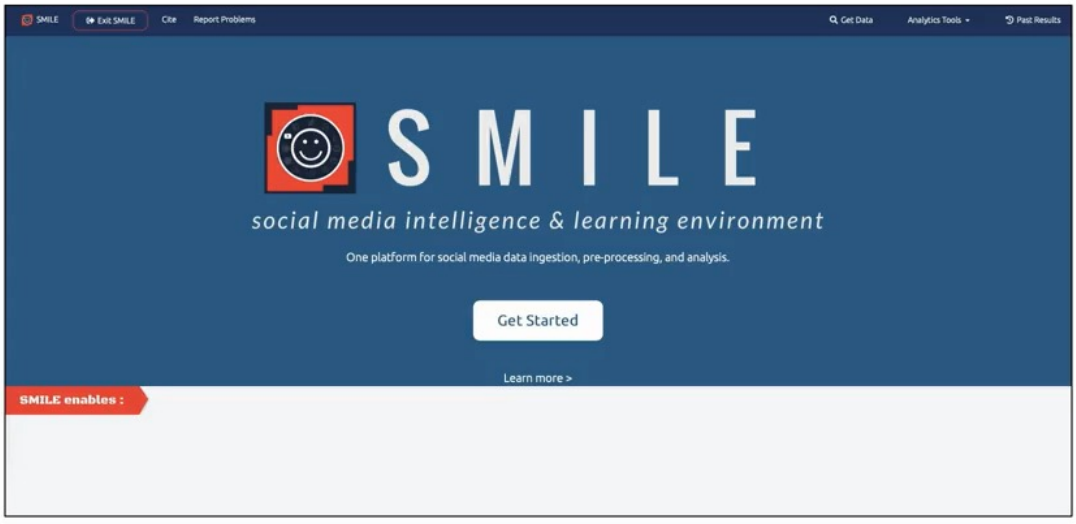
But after you repeat this process a number of times, a large number of time, which for a computer just over, and over again, you eventually reach a rough steady state where your assignments are pretty good so that these words jive together: data, number, computer. Now I do want to make a point that when you do LDA topic modeling many times, even though the computer does numerous, numerous iterations, the words don't look like they come together sometimes. That's again, this process is a probability based process. Sometimes it doesn't come as clean as this. But if it is as clean as this, here's the thing about LDA topic modeling because most people will take this method and say, "I just run LDA topic modeling and then I'm done." That's just not true because the humans still needs to take the step of looking at all the words that are clumped together, that are clustered together, in this case, data, number, and computer, and the human needs to decide what is the topic being discussed. I would say technology. You might actually say something different. You might say computers. Human estimate that call. Or if these words: brain, neuron, and nerve come up as clustered together, then I might say the topic is nervous system. That is in general how LDA topic modeling works.

SMILE (within the SMM)



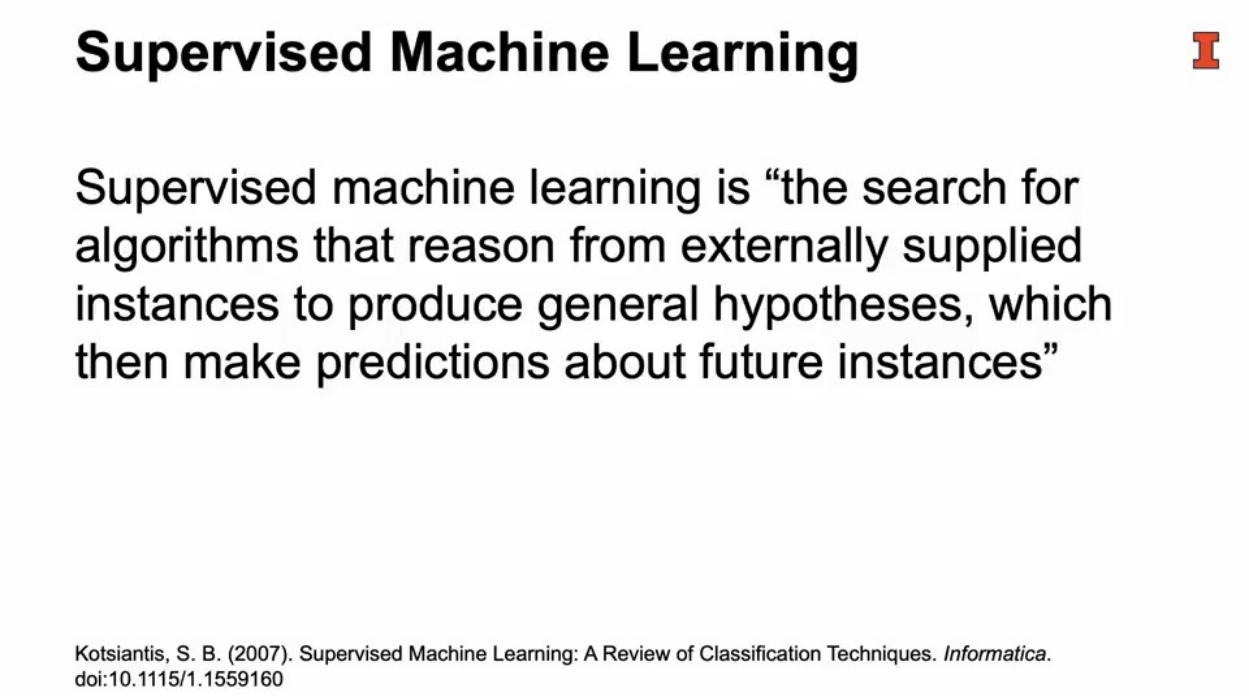
What we'll do now is we'll go into the Social Media Macroscope again within the Smile tool, and we will actually apply running LDA topic modeling on social data. We will now see a demonstration of LDA topic modeling from within the Smile tool within the Social Media Macroscope. Now in the previous video, we pulled a data set, Nike, full of Nike tweets. At this point, we can just go to Analytics Tools here at the top, we can click on "Topic Modelling". From here, just like the previous steps of text preprocessing as well as automated phrase mining, we can select the data set Nike, see a preview, and then we have to choose a number of topics.

SMILE (within the SMM)



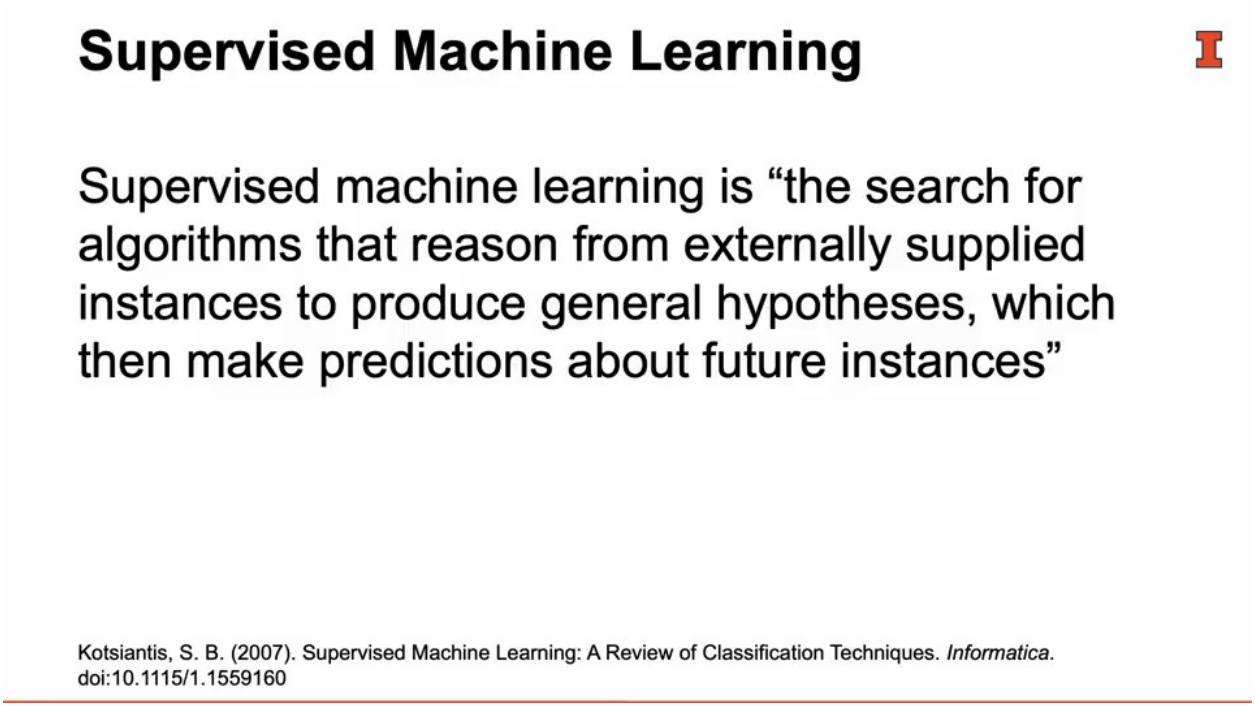
With LDA topic modeling, one of the things that you have to select in the beginning, which is a parameter of this method, is how many topics you believe are within the data set. Now this is a process in which you can calculate via two different scores. One is called the perplexity score, the other one is called the coherent score. We didn't go too much into those specific scores in the lecture that I previously gave, but those are not necessarily the focus of this introductory LDA topic modelling exercise. For now, we will just choose five topics. We will choose the LDA Jensen Package, which you can read more about right here. Click on "Submit". Then once again, just like with automated phrase mining, you need to put your email address, and then click on "Submit". Once you do that, you'll wait, and after a while, once the job is finished, you will get an email saying that your topic modelling has finished, and then you can go to Past Results. You can go to Topic Modeling here on the left-hand side. I've tagged this with topic, Nike, five topics. If I click on these results, once again, we see some downloadables as to various output results for this package. We see a really nice interactive topic modelling visualization, where right here you can see for Topic 1, these are the words that are part of Topic 1, and then Topic 2. I'm clicking on these here, you can see the words. Remember in topic modelling, the goal is to take these words that are statistically related to one another and try to discern what the topic is here. It's not necessarily at first glance very discernible, and then you actually also get four or five topics, the perplexity score, and the coherent score. Long story short, you could actually do an experimentation of four topics or six topics, 7, 8, and you'll get various perplexity and coherent scores. The way that you actually choose what is the maximal number of topics is that you will graph out the perplexity and do a coherent score and try to figure out at what point do those scores seem to level off. That's when you will find the appropriate number of topics to choose for your data set. But again, we're not getting too into the method of LDA topic modeling, but rather this is just an introduction of LDA topic modeling and how to conduct that method via the Smile tool. Thank you.

Machine-Learned Classification and Semantic Topic Tagging

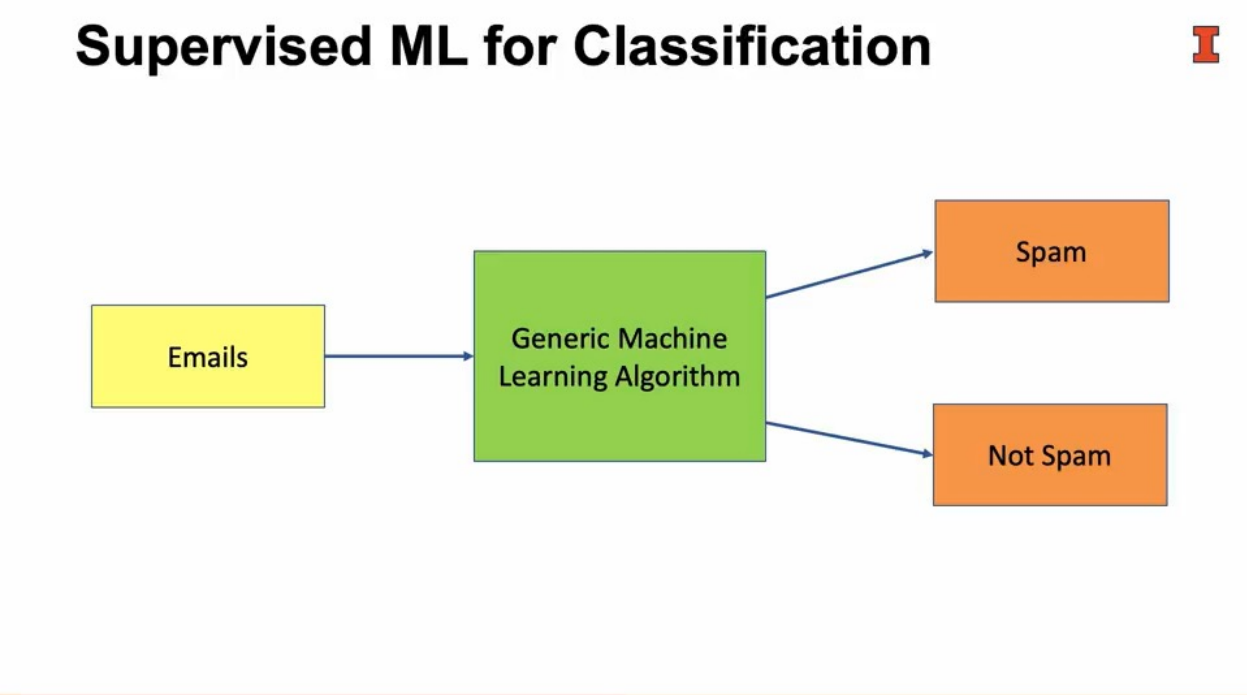


So our last lesson on text summarization will cover machine learning classification and semantic topic tagging. So let's get started. Machine learning classification is a subset of what is called supervised machine learning. And supervised machine learning is the search for algorithms that reason from

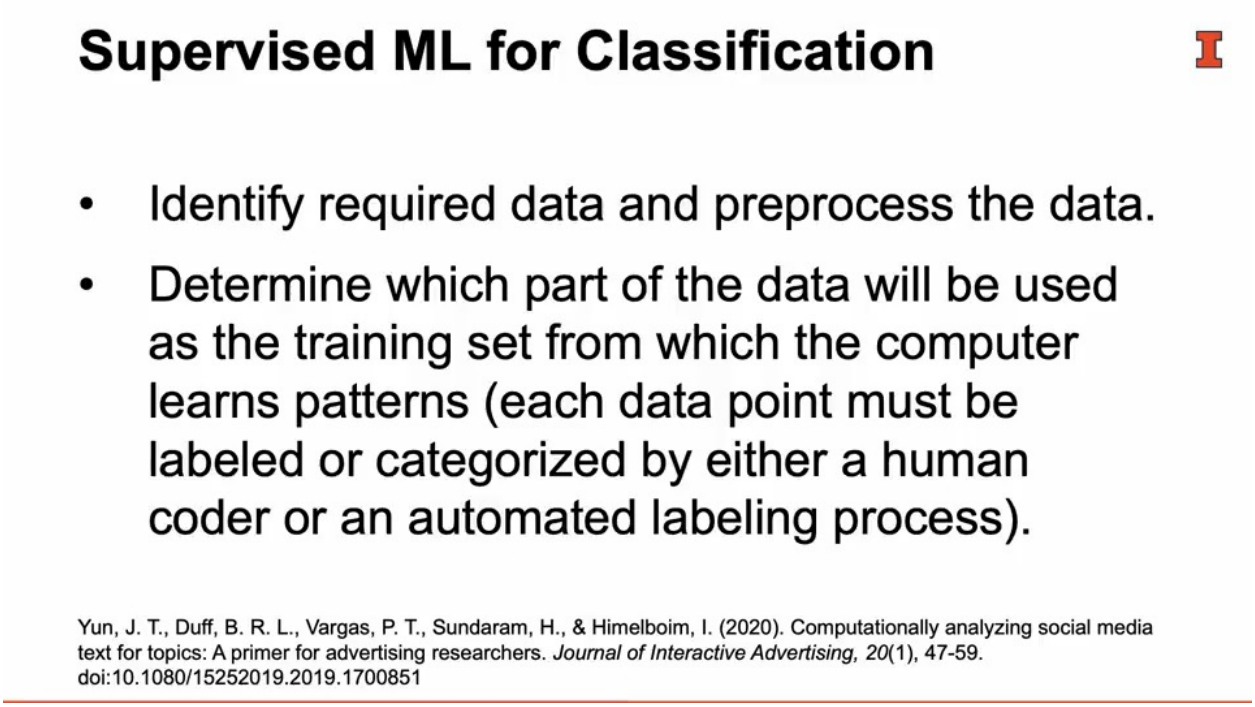
externally supplied instances to produce general hypotheses which then make predictions about future instances. Now that kind of sounds a bit confusing. But basically what this is saying is that supervised machine learning has taken taking intelligence that we have from before and then applying that to predict various things in the future.



And so one of the most common tasks for supervised machine learning is this task of classification. And I'm just going to say that you can think of the word categorization, it doesn't necessarily apply 100%. But in general when you hear the word classification, what we're talking about is how can a computer accurately categorize things with regards to data borrowing from intelligence from the past? Remember supervised machine learning is about taking intelligence from the past and then trying to predict something into the future. And what we're trying to do with classification is we're trying to predict categorization. Now this example, this next example will make everything that I've just said pretty clear, I believe. So let's consider this example of spam detection.



Now all of us use email and let's pretend we all use Gmail, right? And this works for any major email provider but but let's use Gmail for example, because that's what I use for my personal email. With regards to gmail when you're in your email interface on the web browser, you'll see a button that says spam and basically that button is indicating to google that the email that you're looking at right now is spam. Now, what's really interesting is a lot of times nowadays, I never have to click that button. I think this past year, I probably clicked it once or twice, why? Because Google spam detection is really good. But I can tell you because I remember many years ago when Gmail was beta and I joined that beta program that there was quite a few times I was clicking that button spam, but spam detection got intelligent over time. What happened? What happened was supervised machine learning for classification. Basically what was happening, I believe I don't work at Google but I believe what was happening in the background was that in the beginning of Gmail. Every time somebody clicked on spam, what Google did was they took that email that you're looking at and they put it into spam bucket. And then when you didn't click on spam, they would take that email and put it into not spam bucket. And so over time, what they would have is a human crowdsourced categorization of either emails that look like spam or emails that don't look like spam. And so remember supervised Machine learning is all about taking intelligence from the past and trying to predict something for the future. And so once they had enough examples of emails that were classified as spam are categorized as spam and emails that were categorized as not spam. Then they used generic machine learning algorithms or machine learning algorithms and we won't go too much into that in this course. But basically mathematical techniques where they would ask the computer, we've given you all these examples now of spam and not spam. Can you now try to predict any new email as to whether it is spam or not spam? And this is something that machine learning is so good at doing is that when you give it enough examples, intelligence from the past that eventually it can start to very accurately predict categorization in the future. And so that's how supervised machine learning for classification works.



Now the process for supervised machine learning for classification specifically with regard to text data, because we're talking about text analytics in this module, is first we need to identify the required data and pre process that data. So those all those steps of tokenization and normalization, those all apply for machine learning for classification. And so we we run through that pre processing and then we have to determine which part of the data will be used as a training set from which the computer learns patterns. And remember that each data point must be labeled or categorized by either human coder or an automated labeling process. So basically in supervised machine learning you have what's called a training set and what's called the test set. And the training set has to be pre labeled. So remember all that work that we all did for Google that we told them something was either spam or not spam. We could call that whole time training and that's a training set or at least labelling for training. And then the computer can use that intelligence to then future predict spam in the future. Well, that's kind of the same process here. Is that we need to take that data that we've collected the text data and then we need to create this this training set. And then once we have that training set, we take this other set of non labeled data or data that we want to try to predict. And we select various machine learning algorithms so that the computer can mathematically learn patterns to maximize prediction accuracy. And again, we're not going to go too much into those algorithms or not at all actually in this course, but various machine learning courses that you may take in the future. You'll learn quite a bit about the various algorithms that are out there for machine learning.

Supervised ML for Classification

- Select which machine-learning algorithm(s) will be used for the computer to learn patterns within the labeled data training set.
- Allow the algorithm(s) to predict a test set of data that has also been labeled to evaluate how well the algorithm(s) can guess the prelabeled test set.

Yun, J. T., Duff, B. R. L., Vargas, P. T., Sundaram, H., & Himelboim, I. (2020). Computationally analyzing social media text for topics: A primer for advertising researchers. *Journal of Interactive Advertising, 20*(1), 47-59. doi:10.1080/15252019.2019.1700851

Once we've selected the proper set of algorithms, we will use it to predict the test set of data. And that test set of data has also been labelled so that we can assess how well the machine learning algorithm can guess the pre labeled test set. So basically did the machine learning algorithm match the labelling of that test set and how well did it match it. And if it matches it really well, let's say with like 99% accuracy. Now we know that we've got a great spam detection module, we've got a great whatever detection machine learned module for whatever we're trying to categorize. And maybe what we're trying to categorize is topics right, summarizing topics of conversation for text data.

Semantic Topic Tagging

Semantic topic tagging is “the extraction and disambiguation of entities and topics mentioned in or related to a given text.”

Jovanovic, J., Bagheri, E., Cuzzola, J., Gasevic, D., Jeremic, Z., & Bashash, R. (2014). Automated semantic tagging of textual content. *IT Professional, 16*(6), 38-46.

Now, as an applied example for understanding what topics are being discussed in a body of text using a supervised machine learned for classification method. We have semantic topic tagging which is another cutting edge technique. And it's the extraction and disambiguation of entities and topics mentioned in or related to a given text. Now remember with Dr Hans or Dr Chang's automated phrase mining, they use the titles of Wikipedia entries to figure out what our quality phrases for automated phrase mining. Semantic topic tagging is actually taking that to the next level because it's not just taking the titles of Wikipedia entries, but it's taking the content of the Wikipedia entry itself as training data for supervised machine learning for classification. What do I mean by that? Let's just use the example since we used it before of Barack Obama. Barack Obama has a Wikipedia entry, and that entry lists a lot of description about who. Barack Obama is, how he grew up, his, his political career, all of these things. And what supervised machine learning can do is it can take all of those words in the description of the Wikipedia entry for Barack Obama. And it can say this is my pre labeled crowd sourced, text upon which I can use in the future to predict if people are talking about Barack Obama in the text without even mentioning necessarily his name, right? This is a very cutting edge application of supervised machine learning for classification using an already crowdsource pre labeled dataset which is Wikipedia. Now within the smile tool within the social media microscope, we do not have semantic topic tagging installed. So that's something where we're not necessarily going to apply in this class. Although there's various open source packages online that you can look at for semantic topic tagging but we do have a way to do supervised machine learning for classification. I'm not going to necessarily require that of you in this class because that requires quite a bit of human labelling as I mentioned before. And so what I really wanted you to know is that there are these methods for text summarization called supervised machine learning for classification as well as this cutting edge method, semantic topic tagging. But we're not necessarily going to apply that method in this class but feel free if you have any additional questions about that we can discuss.

Still have a ways to go…

Table 6: Results Comparison for One Brand's (FitBit) Topics as Determined by Each Method

Human Coded	Text Preprocessed Summarization	Phrase Mining	Topic Modeling	Semantic Topic Tagging
Charitable Giving	happystep	red carpet	Marketing Fitbit	Fitbit
Customer Service	step	enrollment program	Exercise	Physical fitness
Diet	fitbit	personal trainer	Exercise Encouragement	Physical exercise
Exercise	goal	peanut butter	Healthy Lifestyle	Email
Fashion	fitbitfriend	[Unsupported characters]	Fitbit Customer Service	Health
Health	hear	ultramarathon man		Woohoo (Christina Aguilera song)
Marketing	great	case number		Motivation
Technology	fit	stay tuned		Fun
	job	water resistant		United States dollar
	awesom	losing weight		Billboard 200
	congrat	woody scal		Steps (group)
	make	bay area		With You (Chris Brown song)
	tip	corporate wellness		The Who
	day	slow cooker		Deutsche Mark
	tracker	weight gain		Calorie
	good	continuous heart rate		Heart rate
	workout	weight loss		Today (U.S. TV program)
	share	wireless headphones		Recipe
	work	alta hr		Thanks for Sharing
	love	yoga poses		For Good

https://www.researchgate.net/publication/334002080_Computationally_analyzing_social_media_text_for_topics_A_primer_for_advertising_researchers

Lastly, what I wanted to do is position all of these methods side by side and you see that in this graph that is from an article that I recently published. And what you'll see here is on the left side of the graph. The first column is the the same text and what this is is actually the Twitter timeline for the company Fitbit and you'll see that it's coded via different ways or it summarized via different ways. That first column is a

human coding. So I actually had people look over the tweets of Fitbit and I had them tell me or write down what are the topics being discussed and then I use computational methods. Text pre processing, summarization and the second column phrase mining, topic modeling and semantic topic tagging and what I want us to to look at in this graph. If we consider humans as a gold standard is how different every process computational process is from the human coded summarization. Now if you look at each column, each one has a different strength and weakness. I would say that if you're just doing a word to word comparison, maybe semantic topic tagging is one of the strongest because it has health and the humans coded health. But each one has kind of a different flavor to it that helps us understand the text in better ways that that might actually even add on to what the humans did with regard to their coding process. But long story short no computational method is perfect and we have to remember that. So for me, my strategy is try as many methods as possible and to try to harmonize all the methods together to really get at what are people talking about. The last thing I want to say about text summarization is that if you have a small body of text then there's no reason to use computational methods. We as humans can just read the text and we can have an understanding of what is being discussed. These methods are for when you have let's say thousands or millions of news articles or hundreds of thousands of tweets where no human being is going to be able to process that accurately in a short period of time. And so with that said this concludes our text summarization and, yeah, I hope you had a great time going over this method. Thank you.